
标准数独技巧教程

Standard Sudoku Technique Tutorial

小向

版本 2019.11.4

注意事项

1. 这是一份技巧教学的文档，难度可能偏大，所以按自己需要看到合适的位置。
2. 教程不能跳着看，否则引用到前文的内容会看不明白。
3. 内容丰富多样化，但建议在阅读的时候配合文字，不要只看图。
4. 本文档完全免费公开，请勿以付费形式转发，违者必究。
5. 如果里面的技巧配图是您自己的，但您不愿意为我们提供版权，请联系我，我将致歉并立即撤下图片。
6. 我的联系方式：

方式	账号
QQ	747507738
微信	暂无
Bilibili (B 站)	sunnie-shine
知乎	张齐天
GitHub	Sunnie-Shine

感谢你的阅读。

目录 Contents

第一篇章 数独简介与直观类技巧	1
Part 1 数独简介	1
1-1 数独规则说明	1
1-1-1 解的唯一性	2
1-1-2 错题	2
1-1-3 约定	2
1-2 坐标表示	2
1-2-1 基本表示手段	3
1-2-2 单元格坐标	3
1-2-3 缩写	3
1-3 术语	3
1-4 独·数之道	4
1-4-1 数独历史和发展	4
1-4-2 最少和最多提示数问题	5
1-4-3 国外各数独大师简介	6
1-4-3-1 欧拉 (Euler Leonhard)	6
1-4-3-2 高乐德 (Wayne Gould)	6
1-4-3-3 鍛治真起 (Kaji Maki)	7
1-4-3-4 西尾彻也 (Nishio Tetsuya)	7
Part 2 排除 (Hidden Single)	8
2-1 宫排除 (Hidden Single in Block)	8
2-2 行列排除 (Hidden Single in Line)	8
2-2-1 行排除 (Hidden Single in Row)	9
2-2-2 列排除 (Hidden Single in Column)	9
2-2-3 为什么会先介绍宫排除，而不是熟悉的行或列的排除？	10
2-2-4 来几个练习题	10
Part 3 唯一余数 (Naked Single)	11
3-1 一个示例	11
3-2 怎么观察？	11
Part 4 区块 (Locked Candidates)	14
4-1 宫区块 (Pointing)	14
4-1-1 宫区块 + 排除	14
4-1-2 宫区块 + 唯一余数	15

4-2	行列区块 (Claiming)	15
4-2-1	行区块 (Claiming in Row)	15
4-2-2	列区块 (Claiming in Column)	16
4-3	组合区块/级联区块 (Cascading Locked Candidates)	17
Part 5	数组 (Subset)	19
5-1	隐性数组 (Hidden Subset)	19
5-1-1	隐性数对 (Hidden Pair)	19
5-1-2	隐性三数组 (Hidden Triple)	20
5-1-3	隐性四数组 (Hidden Quadruple)	21
5-1-4	怎么观察?	21
5-2	显性数组 (Naked Subset)	21
5-2-1	显性数对 (Naked Pair)	22
5-2-2	显性三数组 (Naked Triple)	23
5-2-3	显性四数组 (Naked Quadruple)	23
5-2-4	为什么没有五数组?	24
5-2-5	怎么观察?	24
5-3	数组内区块	25
5-4	死锁数组 (Locked Subset)	25
5-4-1	死锁数组的基本结构形式	26
5-4-2	死锁数组引出的割补法的高级用法	28
Part 6	复合结构技巧 (Complex Pattern)	30
6-1	宫区块 + 行列区块 + 排除	30
6-2	数组 + 区块 + 排除	31
6-3	区块 + 唯一余数	32
6-4	数对 + 区块 + 唯一余数	33
第二篇章	定式候选数技巧	34
Part 1	候选数初步	34
1-1	做题模式	34
1-2	出数和删数	34
Part 2	区块 (Locked Candidates)	35
2-1	宫区块 (Pointing)	35
2-2	行列区块 (Claiming)	36
Part 3	数组 (Subset)	37
3-1	显性数组 (Naked Subset)	37
3-1-1	显性数对 (Naked Pair)	37
3-1-2	显性三数组 (Naked Triple)	38
3-1-3	显性四数组 (Naked Quadruple)	38
3-2	隐性数组 (Hidden Subset)	39
3-2-1	隐性数对 (Hidden Pair)	39
3-2-2	隐性三数组 (Hidden Triple)	39
3-2-3	隐性四数组 (Hidden Quadruple)	40
3-3	区块数组 (Naked Subset With Locked Candidates)	40
3-3-1	区块三数组 (Naked Triple With Locked Candidates)	40

3-3-2 区块四数组 (Naked Quadruple With Locked Candidates)	41
3-4 真·为什么没有五数组?	41
3-4-1 显隐性互补	41
3-4-2 为什么要提显隐性互补?	43
3-5 提一下命名	43
Part 4 鱼/链列 (Fish)	44
4-1 普通链列 (Basic Fish)	44
4-1-1 二链列 (X-Wing)	44
4-1-2 一些术语	45
4-1-3 三链列/剑鱼 (Swordfish)	45
4-1-3-1 一个奇怪的例子	45
4-1-3-2 奇怪的事情发生了	46
4-1-4 四链列/水母 (Jellyfish)	47
4-1-5 普通鱼的互补	47
4-1-6 说一下命名	48
4-1-7 普通鱼的第二个视角: 降维	49
4-2 鱼鳍	50
4-2-1 鳍链列 (Finned Fish)	50
4-2-1-1 鳍二链列 (Finned X-Wing)	51
4-2-1-2 鳍三链列 (Finned Swordfish)	52
4-2-1-3 鳍四链列 (Finned Jellyfish)	54
4-2-1-4 普通鳍鱼的互补	54
4-2-2 退化链列 (Sashimi Fish)	55
4-2-2-1 退化二链列 (Sashimi X-Wing)	55
4-2-2-2 退化三链列 (Sashimi Swordfish)	57
4-2-2-3 退化四链列 (Sashimi Jellyfish)	58
4-2-3 孪生链列 (Siamese Fish)	59
4-2-3-1 孪生二链列 (Siamese X-Wing)	59
4-2-3-2 孪生三链列 (Siamese Swordfish)	60
4-2-3-3 孪生四链列 (Siamese Jellyfish)	61
4-2-4 说一下命名	63
4-3 怎么观察普通链列和鳍链列?	64
4-3-1 普通链列的观察	64
4-3-2 鳍链列的观察	67
4-3-3 最后的例子	69
4-4 来个练习题?	70
Part 5 分支匹配 (Wings)	71
5-1 XY-Wing	71
5-1-1 一个看起来简单的例子	71
5-1-2 为什么不是“有且仅有一格是 4”?	71
5-1-3 “共同对应的地方”是什么意思?	72
5-1-4 老规矩, 说一下观察	72
5-2 XYZ-Wing	73
5-3 WXYZ-Wing	73

5-3-1	普通的 WXYZ-Wing	74
5-3-2	折点残缺的 WXYZ-Wing	75
5-4	VWXYZ-Wing	75
5-4-1	普通的 VWXYZ-Wing	76
5-4-2	折点残缺的 VWXYZ-Wing	76
5-5	为啥英文名这么怪?	77
5-5-1	XY-Wing 还是 XYZ-Wing?	77
5-5-2	“字母-Wing”模式?	77
Part 6	唯一矩形 (Unique Rectangle)	78
6-1	经典唯一矩形 (Basic Unique Rectangle)	78
6-1-1	标准类型 (UR Type 1)	78
6-1-2	原理进一步剖析	80
6-1-2-1	#1: UR 能分属于四个宫吗?	80
6-1-2-2	#2: UR 的结论是出数还是删数?	81
6-1-2-3	#3: 已经形成了致命形式的唯一解题目还往下做, 会怎样?	81
6-1-2-4	#4: 非唯一解的题目使用了 UR, 会怎样?	82
6-1-3	区块类型 (UR Type 2)	82
6-1-4	数组类型 (UR Type 3)	84
6-1-4-1	UR + 显性数对	84
6-1-4-2	UR + 显性三数组	85
6-1-4-3	UR + 显性四数组	86
6-1-4-4	UR + 隐性数对	87
6-1-4-5	UR + 隐性三数组	88
6-1-4-6	UR + 隐性四数组	88
6-1-4-7	数组的显隐性是互补的, 那么 UR 里的数组呢?	89
6-1-5	共轭对类型 (UR Type 4)	89
6-1-5-1	一个示例	89
6-1-5-2	别急, 结论还不知道, 我们来看看问题	89
6-1-5-3	那么, 结论呢?	91
6-1-5-4	有一些想说的	91
6-1-5-5	二链列类型	91
6-1-5-6	隐性唯一矩形 (Hidden Unique Rectangle)	92
6-1-6	其它唯一矩形类型 (UR Other Type)	93
6-1-7	结构简图与小测试	95
6-2	残缺唯一矩形 (Incompleted UR)	98
6-2-1	标准类型 (Incompleted UR Type 1)	98
6-2-2	区块类型 (Incompleted UR Type 2)	100
6-2-3	数组类型 (Incompleted UR Type 3)	101
6-2-4	共轭对类型 (Incompleted UR Type 4)	102
6-3	死锁唯一矩形 (Locked UR)	103
6-3-1	标准类型 (Locked UR Type 1)	104
6-3-2	区块类型 (Locked UR Type 2)	105
Part 7	唯一矩形的形式拓展	106
7-1	唯一环 (Unique Loop)	106

7-1-1	6 格的唯一环 (UL Size 6)	106
7-1-1-1	标准类型 (UL Size 6 Type 1)	106
7-1-1-2	区块类型 (UL Size 6 Type 2)	107
7-1-1-3	数组类型 (UL Size 6 Type 3)	108
7-1-1-4	共轭对类型 (UL Size 6 Type 4)	109
7-1-2	8 格的唯一环 (UL Size 8)	110
7-1-3	10 格的唯一环 (UL Size 10)	110
7-1-4	12 格的唯一环 (UL Size 12)	111
7-1-5	14 格的唯一环 (UL Size 14)	111
7-1-6	原理进一步剖析	112
7-1-6-1	#1: UL 占据的单元格总数必须是偶数格吗?	112
7-1-6-2	#2: UL 最大能占多少个单元格?	112
7-2	拓展矩形 (Extended Rectangle)	113
7-2-1	6 格的拓展矩形 (XR Size 6)	113
7-2-1-1	标准类型 (XR Size 6 Type 1)	113
7-2-1-2	区块类型 (XR Size 6 Type 2)	114
7-2-1-3	数组类型 (XR Size 6 Type 3)	114
7-2-1-4	共轭对类型 (XR Size 6 Type 4)	115
7-2-1-5	变体类型	115
7-2-2	8 格的拓展矩形 (XR Size 8)	117
7-2-3	10 格的拓展矩形 (XR Size 10)	117
7-2-4	12 格的拓展矩形 (XR Size 12)	118
7-2-5	14 格的拓展矩形 (XR Size 14)	118
7-2-6	原理进一步剖析	118
7-2-6-1	#1: 拓展矩形必须涉及偶数个单元格吗?	118
7-2-6-2	#2: 为什么例子左右或上下对应位置的候选数完全一样?	118
7-2-6-3	#3: 拓展矩形最大能涉及多少个单元格?	119
7-3	可规避矩形 (Avoidable Rectangle)	120
7-3-1	标准类型 (AR Type 1)	120
7-3-2	区块类型 (AR Type 2)	121
7-3-3	数组类型 (AR Type 3)	121
7-3-3-1	AR + 显性数对 (AR Type 3 With Naked Pair)	121
7-3-3-2	AR + 隐性三数组 (AR Type 3 With Hidden Triple)	122
7-3-4	隐性可规避矩形 (Hidden AR)	122
7-4	全双值格致死解法 (Bivalue Universal Grave)	124
7-4-1	标准类型 (BUG Type 1)	124
7-4-2	原理进一步剖析	126
7-4-2-1	怎么观察?	126
7-4-2-2	两个不能用 BUG 的例子	127
7-4-3	区块类型 (BUG Type 2 / BUG + n)	128
7-4-3-1	同数 BUG + 3	128
7-4-3-2	异数 BUG + 2	129
7-4-3-3	异数 BUG + 2 的另一则示例	130
7-4-3-4	真数在同一个单元格的 BUG + 2	130

7-4-3-5	异数 BUG + 3	131
7-4-3-6	异数 BUG + 3 的另一则示例	132
7-4-3-7	异数 BUG + 4	132
7-4-4	数组类型 (BUG Type 3)	133
7-4-4-1	BUG + 显性数组 (BUG Type 3 With Naked Subset)	133
7-4-4-2	BUG + 隐性数组 (BUG Type 3 With Hidden Subset)	137
7-4-4-3	为什么 BUG 里的数组没有隐性数对, 却有五数组?	139
7-4-5	共轭对类型 (BUG Type 4)	141
7-4-6	待定 BUG (Almost BUG)	142
7-4-6-1	可能形成 BUG 的 ABUG	142
7-4-6-2	可能形成 BUG + 1 的 ABUG	143
7-4-7	BUG 最少多少个单元格?	144
Part 8	待定数组 (Almost Locked Set)	145
8-1	欠一数组 (Almost Locked Candidates)	145
8-1-1	欠一数对 (Almost Locked Pair)	145
8-1-1-1	基本构型 (ALP Basic Type)	145
8-1-1-2	扩展构型 (ALP Extended Type)	146
8-1-2	欠一三数组 (Almost Locked Triple)	150
8-1-2-1	基本构型 (ALT Basic Type)	150
8-1-2-2	拓展构型 (ALT Extended Type)	150
8-1-3	欠一数组的直观视角	152
8-1-4	欠一数组的互补性	152
8-2	融合待定数组 (Sue de Coq)	153
8-2-1	基础融合	153
8-2-2	数组规格不同的融合	154
8-2-3	“九数组”示例	154
8-2-4	带区块的融合	155
8-2-5	显性转隐性的视角	156
8-2-6	自噬融合	157
8-2-7	从欠一数组到 SDC 的转化	160
8-3	跨区数组基本原理 (Extended Subset Principle)	160
8-3-1	假数组/伪数组 (Subset Counting)	160
8-3-2	跨区数组删数理论和原理	161
8-3-3	SDC 和跨区数组	162
8-4	均衡数组 (Aligned Subset Exclusion)	163
8-4-1	均衡数对 (Aligned Pair Exclusion)	163
8-4-2	均衡三数组 (Aligned Triple Exclusion)	165
8-4-3	均衡四数组 (Aligned Quadruple Exclusion)	166
8-4-4	为什么如此暴力的技巧, 也会放在这里介绍?	166
第三篇章	链	167
Part 1	双强链 (Turbot Fish)	167
1-1	摩天楼 (Skyscraper)	167
1-2	基本链理论和术语	168

1-2-1	一些术语	168
1-2-2	链的呈现方式	168
1-3	原理进一步剖析	169
1-3-1	链的头尾可以同真吗?	169
1-3-2	链的长度有什么结论吗?	170
1-3-3	“强-强-强”也是链?	170
1-4	双线风筝 (Two-string Kite)	171
1-5	多宝鱼 (Turbot Fish Normal Type)	172
1-6	共轭对和强关系的异同	172
Part 2	同数链与异数链	173
2-1	链的基本推论回顾	173
2-2	同数链	173
2-3	异数链	174
2-3-1	双值格链 (XY-Chain)	174
2-3-2	远程数对 (Remote Pair)	175
2-3-3	首尾异数链 (XY-X-Chain)	176
2-3-3-1	先来看看标准版长啥样	176
2-3-3-2	不连续环 (Discontinuous Nice Loop)	177
2-3-3-3	不连续环的变体	177
2-3-3-4	等会儿, 删数和链的头尾也有强弱关系?	178
2-3-4	自噬链 (Cannibalistic AIC)	179
2-3-5	普通的标准链	180
2-3-6	Wing 结构	181
2-3-6-1	XY-Wing	181
2-3-6-2	W-Wing	182
2-3-6-3	Y-Wing? W-Wing?	182
2-3-6-4	M-Wing	183
2-3-6-5	Split-Wing	183
2-3-6-6	Local-Wing	184
2-3-6-7	Hybrid-Wing	184
2-3-6-8	总结	184
2-3-7	谈一谈观察?	185
2-3-7-1	#1: 鱼图题	185
2-3-7-2	#2: 一些真的练习题	186
Part 3	守护者 (Guardian)	188
3-1	守护者的基本逻辑	188
3-2	守护者理论的证明	189
3-3	死环 (Guardian Pair)	189
Part 4	区块链 (AIC With Locked Candidates)	191
4-1	区块节点的引入	191
4-2	区块节点为假?	192
4-3	区块不连续环 (Grouped Discontinuous Nice Loop)	193
4-4	节点重叠 (Node Overlapping)	194
4-4-1	空矩形 (Empty Rectangle)	194

4-4-2	空矩形的疑问	194
4-4-3	节点重叠 (Node Overlapping)	195
4-5	鱼图练习	196
Part 5	待定数组 (ALS)	197
5-1	强 ALS (Strong ALS)	197
5-1-1	同区域异数强关系引入	197
5-1-2	ALS-XZ 和伪数组	199
5-1-3	孪生 ALS-XZ (Siamese ALS-XZ Rule)	200
5-2	链的双向性和强弱关系的新定义	201
5-3	ALS 的互补视角	204
5-4	链式 ALS 结构拓展	205
5-4-1	带双 RCC 的 ALS-双强链法则 (Doubly Linked ALS-XZ Rule)	205
5-4-2	ALS-XY-Wing	207
5-4-2-1	普通的 ALS-XY-Wing 示例	207
5-4-2-2	ALS 区域有重叠的 ALS-XY-Wing	208
5-4-3	ALS-双值格链 (ALS-XY-Chain)	209
5-4-4	ALS-W-Wing	210
5-4-5	死亡绽放 (Death Blossom)	211
5-4-5-1	三花瓣死亡绽放 (Death Blossom With 3 Petals)	211
5-4-5-2	四花瓣死亡绽放 (Death Blossom With 4 Petals)	212
5-4-5-3	五花瓣死亡绽放 (Death Blossom With 5 Petals)	213
5-4-6	带 ALS 的链 (Grouped AIC With ALS)	213
5-4-6-1	普通的例子	213
5-4-6-2	假 SDC 的强关系	214
5-4-6-3	节点重叠 (Node Overlapping)	215
5-5	弱 ALS (Weak ALS)	217
5-5-1	同区域异数弱关系的引入	217
5-5-2	WALS 的利用	219
5-5-3	WALS 的互补视角	220
5-6	毛刺数组 (Burred Subset)	221
5-6-1	显性毛刺数组 (Burred Naked Subset)	221
5-6-2	隐性毛刺数组 (Burred Hidden Subset)	222
5-6-3	毛刺数组节点的真假性	223
5-6-4	毛刺数组的解构 (Deconstruct of Burred Subset)	225
5-6-5	毛刺数组的其它使用方式	227
5-6-5-1	节点重叠	227
5-6-5-2	间接删数的链	228
Part 6	待定致命结构 (Almost Deadly Pattern)	230
6-1	待定唯一矩形 (Almost UR)	230
6-1-1	AUR 里的强弱关系	230
6-1-1-1	构建出 UR 标准类型的强关系	230
6-1-1-2	构建出 UR 区块类型的强关系	231
6-1-1-3	构建出 UR 共轭对类型的强关系	232
6-1-1-4	构建出 UR 死锁形式的弱关系	232

6-1-2	万用 UR (UR of Panacea)	233
6-2	待定拓展矩形和唯一环 (Almost XR & Almost UL)	234
6-2-1	待定拓展矩形 (Almost XR)	234
6-2-2	待定唯一环 (Almost UL)	235
6-3	待定可规避矩形 (Almost AR)	236
6-4	BUG + 2 在链里的运用	238
Part 7	环 (Continuous Nice Loop)	240
7-1	标准环 (Continuous Nice Loop)	240
7-2	环的基本形式和特征	241
7-3	常见技巧的环视角	243
7-3-1	标准链列	243
7-3-2	显隐性数组	244
7-3-3	欠一数对	245
7-3-4	双 RCC 的 ALS-XZ	246
7-4	嵌套结构的环 (Grouped Continuous Nice Loop)	246
7-4-1	ALS 环 (Grouped Continuous Nice Loop With ALS)	246
7-4-2	毛刺数组环 (Grouped Continuous Nice Loop With Burred Subset)	248
7-4-3	AUR 环 (Grouped Continuous Nice Loop With AUR)	249
7-4-4	AUR 环的另一则示例	250
7-4-5	待定拓展矩形环 (Grouped Continuous Nice Loop With XR)	251
7-4-6	空矩形欠一数对 (Empty Rectangle ALP)	252
Part 8	强制链与强制结构	254
8-1	基本的强制链类型	254
8-1-1	区域和单元格强制链 (Region & Cell Forcing Chain)	254
8-1-1-1	区域强制链 (Region Forcing Chain)	254
8-1-1-2	单元格强制链 (Cell Forcing Chain)	255
8-1-2	矛盾强制链 (Contradiction Forcing Chain)	255
8-2	链能看逆向, 那么强制链呢?	256
8-3	致命结构的强制视角	257
8-3-1	强制 UR (Forcing UR)	257
8-3-1-1	UR + 2 / 1 CP	257
8-3-1-2	UR + 4 / 2 CP	258
8-3-1-3	UR + 4 / 3 CP	258
8-3-1-4	残缺 UR + 2 / 1CP	259
8-3-1-5	残缺 UR + 2 / 1CP 的另一则示例	259
8-3-1-6	残缺 UR + 3 / 1CP	260
8-3-1-7	分情况讨论的结构	260
8-3-2	强制拓展矩形和强制唯一环 (Forcing XR & UL)	261
8-3-2-1	强制拓展矩形 (Forcing XR)	261
8-3-2-2	强制 UL (Forcing UL)	261
8-3-3	强制 AR (Forcing AR)	262
8-3-3-1	AR + 3 / 1CP	262
8-3-3-2	带一个强制链的强制 AR	262
8-3-4	直推 UR (Direct Inference UR)	263

8-3-5 直推 AR (Direct Inference AR)	264
8-3-5-1 简易的版本	264
8-3-5-2 跳转很多次的版本	264
8-3-6 UR + n	265
8-3-7 UL + n	266
8-3-8 BUG + n	267
8-3-8-1 BUG + 5	267
8-3-8-2 BUG + 10	267
Part 9 动态链 (Dynamic Chain)	269
9-1 动态链是怎么运作的?	269
9-1-1 一则示例	269
9-1-2 文本表达	270
9-2 嵌套结构的动态链 (Dynamic Grouped Chain)	270
9-2-1 嵌套 ALS 的动态链 (Dynamic Grouped Chain With ALS)	270
9-2-2 嵌套 AUR 的动态链 (Dynamic Grouped Chain With AUR)	271
9-3 一些常见的动态链定式技巧	273
9-3-1 三国争雄 (Tri-W-Wing)	273
9-3-2 毛刺链 (Kraken Region Cell)	274
9-4 动态链的逆向视角	274
9-5 动态环 (Dynamic Continuous (Nice) Loop)	275
9-5-1 一个基本的示例	275
9-5-2 原理进一步剖析	277
9-5-2-1 #1: 动态环的删数是否有定式?	277
9-5-2-2 #2: 英文名里的 Nice 有没有很重要吗?	277
9-5-3 嵌套结构的动态环 (Dynamic Grouped Continuous (Nice) Loop)	278
9-5-3-1 动态环 (Dynamic Grouped Continuous Loop)	278
9-5-3-2 动态环 (Dynamic Grouped Continuous Nice Loop)	278
9-6 最后来个练习?	280
Part 10 构造链与构造结构	281
10-1 什么是构造?	281
10-2 基于致命结构的构造 (Deadly Pattern+)	281
10-2-1 UR + XY-Wing	281
10-2-2 UR + XYZ-Wing	282
10-2-3 UR + SDC	283
10-3 基于链的构造 (AIC+)	283
10-3-1 链的构造原理	283
10-3-2 XY-Wing 构造 (XY-Wing+)	285
10-3-3 XYZ-Wing 构造 (XYZ-Wing+)	286
10-3-4 W-Wing 构造 (W-Wing+)	289
10-3-5 空矩形欠一数对构造 (Empty Rectangle ALP+)	289
10-4 基于强制链的构造——毛刺、毛边	290
10-4-1 毛刺的定义	290
10-4-1-1 第一种毛刺用法	290
10-4-1-2 第二种毛刺用法	292

10-4-2	链列的毛刺用途	293
10-4-2-1	链鱼 (Kraken Fish Type 1)	293
10-4-2-2	对 X-Wing 节点的讨论	294
10-4-2-3	链鱼的推广 (Kraken Fish Type 2)	296
10-4-2-4	节点重叠的链鱼	298
10-4-2-5	四个鱼鳍的三头链	299
10-4-3	毛刺 SDC (Burred SDC)	300
10-4-4	毛刺环 (Burred Continuous Nice Loop)	300
10-4-4-1	环 + 双 RCC 的 ALS-XZ	300
10-4-4-2	环 + ALP	301
10-4-5	毛边的定义	303
10-4-5-1	强毛边 (Strong Inference in Bi-burr)	303
10-4-5-2	弱毛边 (Weak Inference in Bi-burr)	304
10-4-6	一些常见技巧里的毛边	305
10-4-6-1	BUG + 3	305
10-4-6-2	XR + 3	305
10-4-6-3	毛边空矩形欠一数对 (Bi-burred Empty W-Wing)	306
10-4-6-4	毛边环 (Bi-burred Continuous Nice Loop)	307
Part 11	总结	308
第四篇章 包装技巧		309
Part 1	涂色 (Coloring)	309
1-1	单一涂色 (Simple Coloring)	309
1-1-1	涂色规则#1: 色链原则 (Simple Coloring Type 1/Wrap)	309
1-1-2	涂色规则#2: 色分原则 (Simple Coloring Type 2/Trap)	310
1-2	复合涂色 (Multi Coloring)	311
1-3	异数涂色 (Advanced Coloring/3D Medusa)	312
1-3-1	同数异色类型 (3D Medusa Type 1)	312
1-3-2	异数同色类型 (3D Medusa Type 2)	313
1-3-3	异数异色类型 (3D Medusa Type 3)	313
Part 2	袋鼠 (Unknown Cover/Kangaroo)	315
2-1	先来看看一般化的版本	315
2-1-1	辅助数类型	315
2-1-2	数组类型	317
2-1-3	确定值类型	318
2-1-4	什么? 直观?	319
2-2	常见技巧的袋鼠视角	319
2-2-1	欠一数对 (ALP in Unknown Cover)	319
2-2-2	W-Wing (W-Wing in Unknown Cover)	320
2-2-3	死环 (Guardian Pair in Unknown Cover)	320
2-2-4	远程数组 (Remote Subset in Unknown Cover)	321
2-2-4-1	远程数对 (Remote Pair in Unknown Cover)	321
2-2-4-2	远程三数组 (Remote Triple in Unknown Cover)	322
2-2-5	唯一矩形 (UR in Unknown Cover)	323

第五篇章 令人畏惧的技巧.....	325
Part 1 复杂致命结构 (Complex Deadly Pattern)	325
1-1 致命结构的确切定义	325
1-2 致命结构的致命类型探讨	326
1-2-1 数组置换类型	326
1-2-2 行列交换类型	327
1-2-2-1 原理	327
1-2-2-2 互补性	327
1-2-3 数字交换类型	329
1-2-3-1 反转唯一环 (Reverse Unique Loop)	329
1-2-3-2 一个更大的反转唯一环	330
1-2-3-3 反转唯一矩形 (Reverse Unique Rectangle)	330
1-2-3-4 直推反转唯一矩形 (Direct Inference Reverse UR)	331
1-2-3-5 位于大列里的反转唯一环基本使用示例	331
1-2-3-6 位于大列里的反转唯一环	332
1-2-3-7 利用不同真的反转唯一矩形	333
1-2-3-8 另一则利用弱关系的示例	333
1-2-3-9 推导更麻烦的反转唯一矩形	334
1-2-3-10 完全分离的两个反转唯一矩形?	335
1-2-3-11 分离的反转唯一矩形的另一则示例	336
1-2-4 对称翻转类型	336
1-2-4-1 宇宙法 (Gurth's Symmetrical Placement)	336
1-2-4-2 轴对称类型的宇宙法	337
1-3 探长致命结构 (Borescoper's Deadly Pattern)	338
1-3-1 结构的形成	338
1-3-2 证明	339
1-3-3 三个数的探长致命结构 (3-Digit Borescoper's Deadly Pattern)	346
1-3-3-1 标准类型 (3-Digit Borescoper's DP Type 1)	346
1-3-3-2 区块类型 (3-Digit Borescoper's DP Type 2)	346
1-3-3-3 拓展类型 (3-Digit Borescoper's DP Advanced Type)	347
1-3-3-4 毛刺探长致命结构	347
1-3-3-5 嵌套探长致命结构的动态链	348
1-3-4 四个数的探长致命结构 (4-Digit Borescoper's Deadly Pattern)	349
1-3-4-1 标准类型 (4-Digit Borescoper's Deadly Pattern Type 1)	349
1-3-4-2 区块类型 (4-Digit Borescoper's Deadly Pattern Type 2)	349
1-3-4-3 其它类型	349
1-4 淑芬致命结构 (Qiu's Deadly Pattern)	350
1-4-1 区分度理论 (Distinction Theory)	350
1-4-2 构型	353
1-4-3 证明	353
1-4-4 淑芬致命结构的使用	358
1-4-4-1 死锁淑芬致命结构 (Locked Qiu's Deadly Pattern)	358
1-4-4-2 另一个示例	359
1-4-4-3 配合反转拓展矩形的淑芬致命结构	360

1-4-4-4	稍微用一点强制推导的淑芬致命结构	361
1-4-4-5	强制分析带有嵌套淑芬致命结构的思路	361
1-4-4-6	双淑芬致命结构 (Dual Qiu's Deadly Pattern)	362
1-4-4-7	双淑芬致命结构的另外一个示例	363
1-5	致命结构的传递性 (Transmission of DP)	364
1-5-1	先来看一个示例	364
1-5-2	原理进一步剖析	364
1-5-3	传递得到的一些示例	367
1-5-3-1	一个比较简单的传递示例	367
1-5-3-2	用了唯一矩形和拓展矩形的传递示例	368
1-5-3-3	依赖于拓展矩形的稍大一些的致命结构	370
1-5-3-4	一个超大的致命结构	370
1-5-3-5	毛刺致命结构	374
1-6	含有隐性视角的致命结构 (Hidden Deadly Pattern)	375
1-6-1	带有隐性视角的致命结构?	375
1-6-2	又是一则带有一点隐性视角示例	376
1-6-3	综合示例	378
1-6-4	不简单的小练习	380
1-7	无法传递或很难使用传递性的结构	381
1-7-1	唯一矩阵 (Unique Matrix)	381
1-7-1-1	使用	381
1-7-1-2	证明	382
1-7-2	唯一性提示信息覆盖 (Uniqueness Clue Cover)	385
1-7-2-1	使用	386
1-7-2-2	原理剖析	387
Part 2	复杂鱼/链列 (Complex Fish)	389
2-1	鱼的原理	389
2-2	宫内鱼 (Franken Fish)	390
2-2-1	宫内鱼的形成	390
2-2-2	原理进一步剖析	392
2-2-2-1	宫内鱼的残缺情况	392
2-2-2-2	宫内二链列 (Franken X-Wing)?	393
2-2-2-3	宫内四链列的结构	394
2-2-2-4	定义域能重叠吗?	394
2-2-2-5	什么? 鱼也可以转置?	396
2-2-3	一些宫内鱼的示例	397
2-2-3-1	宫内三链列 (Franken Swordfish)	397
2-2-3-2	转置的宫内三链列	398
2-2-3-3	宫内四链列 (Franken Jellyfish)	399
2-2-4	带鱼鳍的宫内鱼构型	401
2-2-5	带鱼鳍的宫内鱼的示例	402
2-2-5-1	鳍宫内二链列 (Finned Franken X-Wing)	402
2-2-5-2	鳍宫内三链列 (Finned Franken Swordfish)	403
2-2-5-3	鳍宫内四链列 (Finned Franken Jellyfish)	404

2-2-5-4	鳍宫内五链列 (Finned Franken Squirmbag/Starfish)	405
2-2-6	鱼鳍的定义要被拓展了?! ——内鳍的形成	407
2-2-7	内鳍宫内鱼的示例	409
2-2-7-1	内鳍宫内三链列 (Finned Franken Swordfish)	409
2-2-7-2	内鳍宫内四链列 (Finned Franken Jellyfish)	410
2-2-7-3	内鳍宫内五链列 (Finned Franken Squirmbag/Starfish)	411
2-3	交叉鱼 (Mutant Fish)	412
2-3-1	交叉鱼的形成	412
2-3-2	原理进一步剖析	413
2-3-2-1	交叉鱼的残缺和带鱼鳍的情况	413
2-3-2-2	交叉鱼的鱼鳍	414
2-3-2-3	交叉四链列 (Mutant Jellyfish)	415
2-3-2-4	交叉二链列 (Mutant X-Wing)?	415
2-3-2-5	同时涉及所有区域类型的交叉鱼	416
2-3-2-6	交叉鱼的转置	417
2-3-3	交叉鱼的示例	417
2-3-3-1	交叉三链列 (Mutant Swordfish)	417
2-3-3-2	交叉四链列 (Mutant Jellyfish)	419
2-3-3-3	外鱼鳍交叉鱼 (Exo-finned Mutant Fish)	421
2-3-3-4	内鱼鳍交叉鱼 (Endo-finned Mutant Fish)	426
2-3-3-5	混合鱼鳍的交叉鱼 (Mix-finned Mutant Fish)	429
2-4	自噬鱼 (Cannibalistic Fish)	432
2-4-1	自噬鱼的形成	432
2-4-2	原理进一步剖析	433
2-4-3	自噬鱼的示例	434
2-4-3-1	自噬三链列 (Cannibalistic Swordfish)	434
2-4-3-2	自噬四链列 (Cannibalistic Jellyfish)	435
2-4-3-3	自噬五链列 (Cannibalistic Squirmbag/Starfish)	436
2-4-3-4	带鱼鳍的自噬鱼 (Finned Cannibalistic Fish)	437
2-5	不饱和鱼 (Unsaturated Fish)	438
2-5-1	不饱和鱼的形成和原理	438
2-5-1-1	秩的概念	438
2-5-1-2	所以, 前文都是在讲秩为 0 的鱼?	438
2-5-1-3	那么, 秩可以为负数吗?	438
2-5-1-4	好了, 秩可以怎么用呢?	439
2-5-2	不饱和鱼的示例	440
2-5-2-1	不饱和四链列 (Unsaturated Jellyfish)	440
2-5-2-2	不饱和五链列 (Unsaturated Squirmbag/Starfish)	444
2-5-2-3	不饱和六链列 (Unsaturated Whale)	453
2-6	最后的介绍	454
2-6-1	为什么同数技巧全都可以叫做鱼?	454
2-6-2	自噬为什么是鱼的内容, 而不是其它技巧的内容?	454
2-6-3	鱼的系统命名规则	454
2-6-4	鱼图的符号规则	455

Part 3	飞鱼导弹 (Exocet)	456
3-1	基础使用	456
3-1-1	4 数初级飞鱼导弹 (4-Digit Junior Exocet)	456
3-1-2	术语词	459
3-1-3	3 数 JE (3-Digit JE)	461
3-1-4	5 数 JE (5-Digit JE)	462
3-1-5	第二类标准 JE (JE Type 2)	462
3-1-6	共轭对类型 JE (JE With Conjugate Pair)	463
3-1-7	只有一个基准单元格的 JE	465
3-1-7-1	使用示例	465
3-1-7-2	代数视角 (JE in Unknown Cover)	466
3-1-8	目标单元格同侧的 JE	467
3-1-9	数组类型 JE (JE With Subset)	468
3-2	额外删数的推论	468
3-2-1	同宫定理 (Same-Box Theorem)	468
3-2-2	T 数对定理 (Target Pair)	471
3-2-3	T 邻/镜面单元格定理 (Mirror Theorem)	473
3-2-3-1	原理	473
3-2-3-2	示例 1: 带死锁区块的 JE	474
3-2-3-3	示例 2: 另一则带死锁区块的 JE	475
3-2-3-4	示例 3: 只有 T 邻定理删数的 JE	475
3-2-3-5	示例 4: 需要 T 邻定理删数的第二类 JE	476
3-2-3-6	示例 5: 依赖于区块和三数组反推的 JE	477
3-2-3-7	示例 6: 同时有共轭对和死锁区块的 JE	478
3-2-4	三链列/剑鱼定理 (Swordfish Theorem)	478
3-2-5	X 致命定理 (Bi-Bi Pattern)	479
3-2-5-1	使用	480
3-2-5-2	证明	481
3-2-5-3	一些使用示例	484
3-2-6	其它结论	485
3-2-6-1	示例 1: 利用邻伴镜面单元格得到的弱关系	485
3-2-6-2	示例 2: 利用占位形成的弱关系	486
3-2-6-3	示例 3: 结构有点分散的链	487
3-3	孪生初级飞鱼导弹 (Siamese JE)	488
3-3-1	基础使用逻辑	488
3-3-2	更多例子	490
3-3-2-1	示例 1: 标准的孪生 JE	490
3-3-2-2	示例 2: 孪生的第二类 JE	490
3-3-2-3	示例 3: 带有 X 致命定理删数的孪生 JE	491
3-3-2-4	示例 4: 确定值也可以算作推导的一部分吗?	492
3-4	高级飞鱼导弹 (Senior Exocet)	493
3-4-1	普通示例	493
3-4-2	退化 SE (Sashimi SE)	494
3-5	复杂飞鱼导弹 (Complex Exocet)	495

3-5-1	宫内飞鱼导弹 (Franken Exocet)	495
3-5-1-1	宫内 JE (Franken JE)	495
3-5-1-2	宫内 SE (Franken SE)	496
3-5-2	交叉飞鱼导弹 (Mutant Exocet)	497
3-5-2-1	交叉 JE (Mutant JE)	497
3-5-2-2	交叉线单元格涉及四个区域的交叉 JE	497
3-5-2-3	交叉 SE (Franken SE)	498
3-6	总结和小练习	499
Part 4	网 (Multi-sector Locked Set)	500
4-1	我们从 SDC 说起	500
4-1-1	度 SDC (Degree-base SDC)	500
4-1-2	段 SDC/多米诺链 (Domino Chain)	501
4-1-3	科尔扎斯环/多米诺环 (Kurzhal's Loop/Domino Loop)	503
4-2	MSLS 的定义和使用	505
4-2-1	MSLS 的定义	505
4-2-2	MSLS 的原理	507
4-2-3	小练习	508
4-2-4	MSLS 的使用	509
4-2-4-1	宫内网和交叉网 (Franken & Mutant MSLS)	509
4-2-4-2	非矩形网 (Non-rectanglar MSLS)	511
4-2-4-3	其它饱和网 (Saturated MSLS)	512
4-2-4-4	不饱和网 (Unsaturated MSLS)	512
4-2-4-5	自噬网 (Cannibalistic MSLS)	515
4-2-4-6	SK 环的拓展	516
4-3	直观 MSLS?	517
4-3-1	逻辑	517
4-3-2	MSLS 的互补性	520
4-4	其它结构跟网的联系	521
4-4-1	鱼直观的原理	521
4-4-1-1	先来回顾一下	522
4-4-1-2	证明	523
4-4-1-3	直观标准鱼视角的互补	526
4-4-1-4	标准鱼的观察	528
4-4-1-5	标准鳍鱼的观察	530
4-4-1-6	直观鱼的示例	532
4-4-1-7	总结	536
4-4-1-8	宫内鱼的直观	536
4-4-1-9	交叉鱼的直观	538
4-4-2	飞鱼导弹和网	539
4-4-3	SK 环和网	541
4-4-3-1	如何转换呢?	541
4-4-3-2	SK 环的直观	541
4-4-3-3	$JE + SK = PLQ$	543
4-4-4	JE 的直观	551

第六篇章 无逻辑技巧和计算机算法	555
Part 1 无逻辑人工技巧 (Mankind Guessing)	555
1-1 试数 (Bowman's Bingo)	555
1-1-1 使用逻辑	555
1-1-1-1 类似于链的基础使用方式	555
1-1-1-2 带有技巧结构的试数推导过程	556
1-1-1-3 配合 UR 的试数逻辑: 试数 UR	557
1-1-2 试数转链	558
1-1-3 链和试数的异同	559
1-1-4 带有试数思想的链	560
1-2 图案叠加删数 (Pattern Overlay Method)	561
Part 2 计算机数独算法简介 (Brute Force)	563
2-1 特雷伯尔表 (Forcing Net)	563
2-2 回溯 (Backtracking for Sudoku)	564
2-3 舞蹈链 (Dancing Links for Sudoku)	564
2-4 Linq 算法 (Linqing for Sudoku)	564
附录	565
Part 1 软件测评和使用	565
1-1 Sudoku Explainer	565
1-2 Hodoku	567
1-3 Xsudoku	567
Part 2 坐标表达	569
2-1 RCB 表示法	569
2-2 K9 表示法	569
Part 3 教程使用符号和图形约定	570
Part 4 算法代码	571
4-1 回溯法算法代码	571
4-2 舞蹈链算法代码	574
4-2-1 数据节点文件 (DataNode.cs)	574
4-2-2 列节点文件 (ColumnNode.cs)	575
4-2-3 舞蹈链文件 (DancingLink.cs)	575
4-2-4 矩阵行节点文件 (DancingLink.MatrixRow.cs)	577
4-2-5 解题器对象 (DancingLinksSolver.cs)	578
4-2-6 主方法调用 (Program.cs)	581
4-3 Linq 解法算法代码	582
Part 5 参考文献	584
Part 6 鸣谢	584

第一章 数独简介与直观类技巧

这一部分的技巧相对比较基础，是入门玩家必须了解和介绍的内容。

Part 1 数独简介

本部分简要概括和介绍数独的基础内容，这些内容将对后续的部分作出铺垫，所以非常重要，我们来看一下。

1-1 数独规则说明

什么是数独？数独是一种益智游戏，在空格里填入数字 1 到 9，使得每一行、每一列和每一个用粗线围起来的 3×3 的九个单元格里，填数都包含 1 到 9 各一个。换句话说，没有重复的数字出现。比如下面这个答案所给出的样子：

9	8	7	4	5	2	6	1	3
3	2	6	1	9	8	7	4	5
5	4	1	7	3	6	8	2	9
6	9	2	5	1	7	4	3	8
4	3	5	8	2	9	1	6	7
7	1	8	3	6	4	5	9	2
8	6	4	9	7	3	2	5	1
1	7	9	2	4	5	3	8	6
2	5	3	6	8	1	9	7	4

可以从图上看，每一行、列和粗线围起来的区域（称为**宫**，**Block**），都有 9 个单元格，并且每一组这样的 9 个单元格都不含有相同的数字，都是 1 到 9 各有一个，不多也不少。这便是数独的规则。规则其实理解起来并不是那么难，对吧。

当然，这里提及一点。规则里说明的“1 到 9 各一个”和“不允许重复”两者应当是等价的说法。换句话说，如果“1 到 9 各一个”这一点满足不了，就得有一个数出现不止一次，也就违背了“不允许重复”的条件。所以一般情况下，很多资料叙述的规则只会提及“1 到 9 各一个”和“不允许重复”的其中一点。

不过我们从规则里看不出什么比较有意义和实质性的东西，所以这里需要继续提及一些内容。

1-1-1 解的唯一性

需要注意的是，每一个题目都只有唯一的**解**（**Solution**）。所谓的解，也就是答案。每一个合适的题目都必须拥有唯一的答案。换言之，每一个空格的填法都只有唯一的一个。但凡拥有某个（或某些）单元格出现多出一种填法的话，都算作多解题，这种题目不是合法的。例如上题，就是一个合格的题目，答案是唯一的；另外，如果一眼就能看到题目所给的数字不满足数独规则，或者是经过一部分推理逻辑得到一些填数并保证这些数字是正确的填入后，此时出现不满足数独规则的情况，那么这种题目叫无解题，即不可能有解的题目。

实际上，唯一解的特殊约定并未定义在题目的整体规则之中，但由于多解和无解题无法找寻结果，以及大多数情况下不能完成而没有意义被排斥，所以唯一解的要求依然被我们算作数独规则之中，作为一条“潜规则”，而这条规则叫做 **Da Rule**，你完全可以认为这个所谓的 **Da Rule** 就指的是唯一解的规则。

如何快速发现题目是多解的呢？目前尚无比较好的办法，要么通过做题经验，要么通过一些经验结论来判断。经验结论将在“致命结构”的技巧部分内详细阐述逻辑及其原理，这里不作过多探讨。

1-1-2 错题

如果在做题过程之中错题了，即发现了行、列和九宫格里出现了重复的数字，这个时候我们怎么去改正呢？一般而言，错题是一件很棘手的事情。只要错题，一般就只有两种解决方案：第一种是通过回忆做题顺序的方式来找到错误所在地方，然后恢复到错误的地方，重新继续完成题目（回溯）。不过这种方法实现起来比较困难，因为回溯需要依赖于记忆做题步骤，而做题步骤太多，才发现错误的话，一般来说都不好解决。我们建议的唯一的办法就是重做此题。

数独是一个很严谨的逻辑推理游戏，如果错题了，还有一种方法，就是通过局部数字置换的方式，将一些数字两两交换，来还原题目。不过这种方法有时候很奏效，但有时候会使得情况变得更糟糕。

1-1-3 约定

我们约定，题中**黑色**的数字表示用于提示做题的数字，**蓝色**的数字表示经过黑色数字推理得到的填数。黑色的数字我们叫做**提示数**或**已知数**（**Given/Hint/Clue**），而蓝色的数字我们叫做**填入数**（**Input**），两者统称**确定值**（**Value**）。后面的题目的图片都使用这种配色呈现出来。

不过，有一部分题目在使用之前尚未找到原题目（即无法还原到原题）的话，那么题目里的所有确定值的数字都将会被默认涂成黑色的。

1-2 坐标表示

我们继续约定，本文档里所有单元格位置的描述和表达均通过**坐标**（**Coordinate**）表示。有如下的一些约定。

1-2-1 基本表示手段

我们使用 **r**、**c**、**b** 和一个 1 到 9 的数字来表示行、列、小九宫格（我们称为**宫**，**Block**），数字是几，那就是顺数第几个。比如 **r1**，就表示顺数第一行；**c6** 则表示顺数的第六列；**b4** 则表示顺数的第四个宫。其中宫的编号方式是从左到右、从上到下。

这个和一般人为理解的逻辑是一样的，所以这里就不用重新作图了。

1-2-2 单元格坐标

我们通过 **rxcy** 的方式表示一个单元格（即第几行和第几列的交集）。比如 **r3c2** 表示第 3 行第 2 列。若要表达某单元格填的是某数的时候，可以使用“**单元格 = 数值**”的形式表达，比如“**r3c2 = 4**”就表示 **r3c2** 填入数字 4；反之使用“**单元格 != 数值**”和“**单元格 <> 数值**”均可表示某个单元格填入的数字不应该是某数，比如“**r3c2 != 3**”就表示 **r3c2** 不应填 3。

不等号在文档里不方便书写，所以采用计算机编程里广泛采用的“**!=**”和“**<>**”来表示不等。

1-2-3 缩写

当需要描述多个单元格的时候，我们可以将相同的部分合并表达。比如如果需要表示 **r3c2**、**r3c3** 和 **r3c4** 的时候，可以将相同的“**r3c**”合并，并将不同的 2、3、4 合并表示，即 **r3c234**；同理合并行也是可以的，比如 **r3c1**、**r4c1** 和 **r7c1** 可以表示为 **r347c1**。

当合并后还可再次合并的时候，也可继续合并，直到最简，比如 **r1c3**、**r1c4**、**r3c3** 和 **r3c4** 四个单元格可以先合并为 **r13c3** 和 **r13c4**，但合并后“**r13c**”是一样的，所以最终的简写方式是 **r13c34**。更大型的结构同理。

但如果怎么合并都无法表示全部的单元格时，就不需要合并了，我们可以使用大括号把所有合并和没有合并的单元格元素表示出来。比如 **r1c1**、**r1c3** 和 **r3c3** 三格可以缩写为 **{r1c13, r3c3}**，单元格元素之间约定使用逗号连接。

更多的合并规则将在后面讲解过程之中逐步说到。

1-3 术语

这部分将解释一部分**术语**词汇（**Terms**）。

- **行、列、宫**：数独的基本元素。
- **区域（Region）**：行、列、宫的统称。
- **解**：题目的答案。合格的题目必须拥有唯一的解。
- **坐标**：表达一个或一组单元格的方式。
- **盘面（Grid）**：整个题目。
- **初盘（Initial Grid）**：在初始情况下的盘面。
- **终盘（Solution）**：答案，和解一个意思。
- **提示数（Given）**：提示完成题目的初始数字，不可以修改。
- **填入数（Modifiable）**：在经过推导得到的数字，这些数字在没有完成题目的任意时

- **确定值 (Value):** 提示数和填入数的统称。

1-4 独·数之道

最初，数独是没有宫的（关于“宫”的详细内容请参看下一章），最早的数独仅有行和列，而规定每一行和每一列内的数字不重复。这在当时是被称为**拉丁方阵**（或**拉丁方**、**拉丁方块**，**Latin Square**）的。由于拉丁方阵，包括如今的数独，它们的规则中并没有涉及任何的代数运算，所以需要填入的 1 到 9 的每一个数字都是可以换成字母或者图片。

Thompe-Élysées. — M. Léon Faure, âgé de cinquante ans, ancien élève de l'École Polytechnique, a été nommé directeur de l'École de la Marine à Toulon. Il a été nommé directeur de l'École de la Marine à Toulon.

Léopold Larba.

DIVERTISSEMENTS QUOTIDIENS

N° 3879 — CARRÉ MAQUEUR DIABOLIQUE
Par M. D. Meynel

Compléter le carré ci-dessous en employant les neuf premiers nombres chacun neuf fois de manière que les horizontales, les verticales et les deux grandes diagonales donnent toujours à l'addition le même total.

7	8	9	1	2	3	4	5	6
3	4	8	9	1	2	5	6	7
5	9	1	2	3	4	6	7	8
2	3	4	5	6	7	8	9	1
1	2	3	4	5	6	7	8	9
9	1	2	3	4	5	6	7	8
8	9	1	2	3	4	5	6	7
6	7	8	9	1	2	3	4	5

Ce carré devra être diabolique, c'est-à-dire que le carré restera magique si l'on place la ligne horizontale ou une colonne verticale à l'issue de toutes les autres.

N° 3565 — MATÉMATIQUES
Par M. Adolphe R.

Solution

Le marchand a vendu 37500 vases; le 50^e jour, le dernier, il a vendu 500 vases.

Solutions fautes

M. Amibryte, un chercheur; Paul et Jules Duplant; Albert Labarre; L. Grollet; C. Gerbaulet.

Les solutions et les envois de problèmes inédits doivent être adressés, dans la huitaine, au rédacteur souscrit.

FÉLIX ANDRÉ.

PROGRAMME D'OPÉRA

OPÉRA, 8 h. 0/0. — **TALMA**.
DÉMAI. — **Relâche.**

FRANÇAIS, 8 h. 1/2. —
OPÉRA-COMIQUE. — **Relâche.**

ODÉON. — **Cigares.**

RENAISSANCE. — **Opéra.**

GYMNASSE. — **Cigares.**

VAUDEVILLE. — **Opéra.**

VARIÉTÉS. — **Cigares.**

1612 年，法国的数学家 Claude-Gaspard Bachet de Méziriac 就给出了 3 阶拉丁方阵的方法。直到 18 世纪，瑞士的大名鼎鼎的数学家欧拉时隔 200 年左右提出了 n 阶拉丁方阵的方法。

1970 年，美国开始发展数独，并更名为填数字（Number Place，日语：ナンバー・プレース，中文为直译），之后流传到日本并开始发扬光大，以数学智力游戏智力拼图游戏

发表。

在 1984 年，一本游戏杂志《パズル通信（つうしん）ニコリ》（谜题通信 Nicolì，直译）正式把它命名为“数字（すうじ）は独身（どくしん）に限（かぎ）る”（数字有唯一的限制）。当时名字就是这么长，但后期由于名字太长了，所以一个叫做鍛治・真起的人就把它简化成了如今的“数（すう）独（どく）”二字。后来一位前任香港高等法院的新西兰籍法官高乐德在 1997 年 3 月到日本东京旅游时，无意中发现了。他首先在英国的《泰晤士报》上发表，不久其他报纸也发表，很快便风靡全英国，之后他用了 6 年时间编写了程序，并将它放在网站上，使这个游戏很快在全世界流行。

中国台湾于 2005 年 5 月由《中国时报》首度引进，并每日连载，亦造成很大的回响。**台湾数独发展协会（Taiwan Sudoku Association，简称 TSA）**亦为世界解谜联盟会员。而中国香港则是于 2005 年 7 月 30 日创刊时引入数独。而中国大陆则是在大约 2007 年正式引入的数独，具体时间无从考究。

目前北京晚报智力休闲数独俱乐部（数独联盟前身）在新闻大厦举行加入世界谜题联合会的颁证仪式，成为世界谜题联合会（简称世智联）的 39 个成员之一。后来更因数独的流行衍生了许多类似的数学智力拼图游戏，例如：**数和（Kakuro）、杀手数独（Killer）、数图（Nonogram）**等。

1-4-2 最少和最多提示数问题

有一个问题，在数独界里一直比较热：一个唯一解的题目最少需要拥有多少个提示数呢？这个问题到目前尚未有一个解析形式的证明，不过计算机通过运算较快的优势，通过加里（Gary）的团队成功得到了结论：**每一个唯一解的数独盘面最少都得有 17 个用于推理解题的提示数**。由于目前不清楚怎么通过解析形式证明这一个结论，所以我们尚不能给出缘由，只能让大家知道这一个结论。不过，目前发现了 17 提示数的唯一解数独题目一共有大约 5 万个题目。

						1		
4								
	2							
				5		4		7
		8				3		
		1		9				
3			4			2		
	5		1					
			8		6			

如图所示，这就是一个只有 17 个提示数，并且具有唯一解的题目。它是目前发现的大约 5 万道题里的其一。当然，也并不是随意写上 17 个提示数，就一定是唯一解的数独题。

另外，“一个唯一解数独题至少需要 17 提示数”可以换一种说法，即这样一个 17 个提示数的题目，每一个提示数都不可缺少，缺少任意一个提示数，都会使得整个题目变成多个答案的题目（多解题）。我们将满足“每一个提示数都不可缺少，缺少任意一个提示数，都会使得整个题目变为多解题”要求的这类题目为**最小题**（或**精简题**，**Minimal Puzzle**）。那么，随之又产生了一个问题：需要最多提示数的精简题，提示数有多少个呢？不过很遗憾的是，这个问题目前没有得到明确结论，不过目前发现了，这个数字至少是 40，下图所示的这个示例即满足这一特征，去掉任意一个提示数，题目都是多解的。

	1	2		3	4	5	6	7
	3	4	5		6	1	8	2
		1		5	8	2		6
		8	6					1
	2				7		5	
		3	7		5		2	8
	8			6		7		
2		7		8	3	6	1	5

但是需要作出警告的是，这个题目难度并不小。不要尝试去解决它。这个题至少得等到您学习到了“链”之后，才可能完成它。不过，如果您愿意尝试一番的话，您可以尝试使用基础的逻辑方式配合试数（试填，发现违背规则的矛盾之后排除原情况的一种技巧）来进行解题。

1-4-3 国外各数独大师简介

接下来将介绍一些国外与数独文化和游戏推广的大师级人物及团队的简单介绍。

1-4-3-1 欧拉（Euler Leonhard）

欧拉（1707 年 4 月 15 日—1783 年 9 月 18 日），是一位瑞士的伟大的数学家和物理学家。欧拉在多个领域其实都有巨大的贡献，比如微积分和图论。另外，我们一直使用的函数表达“ $f(x)$ ”，也是由他引进的。

他也非常爱护家人和小孩，一般晚餐的时候都会和他的小孩一起做数学游戏、读圣经。之后，他于 18 世纪发明了拉丁方（没有宫的数独），这便就是数独的雏形和它的概型。

1-4-3-2 高乐德（Wayne Gould）

高乐德是一名来自新西兰的人，他曾经于 1997 年访日，并彻底改变了他的生活。

1997 年，已经 51 岁的他是香港高等法院的法官，决定退休。在退休的那年，他要去意大利见他的妻子，中途需要在日本转机。等待的过程之中，发现了店内有一个关于数独游戏的书，就决定为了打发时间，就买了一本，并从此爱上了这个游戏。

后面，他向妻子推荐了，于是一家人都爱上了数独游戏。他花了大约 6 年的时间自学编程，并作出了数独游戏程序。根据他自己说的话，他自己都不敢相信自己的努力，眼泪都掉了下来。“早知道编个程序这么难，我就不学了”，他这么调侃到。

后来，他因为特爱数独，所以为了推广，将数独最初免费推给了《泰晤士报》，并为其刊登。之后英国开始逐渐形成流行的数独风。

之后，他又推广到美国。但不幸的是，他推广给《今日美国报》时被拒。但《今日美国报》的一个决策上的错误，最终发现数独越发流行起来后，只好重新从其他渠道引进数独。逐渐地，他也就变成了全球的数独推手。

他的妻子其实比他的做题速度更快一些，而他们一家子人，都非常热爱数独。而很多数独技巧也都由他命名，例如后面会介绍到的 X-Wing 解法。

1-4-3-3 鍛治真起 (Kaji Maki)

1980 年，鍛治真起在一家印刷厂工作，他当时就很喜欢纵横字谜，而他也一直想要出一本杂志，并且当时的确对于日本来说，这样类型的东西的确不多，恰好发现了商机。

于是，他找到了一位女性插画师，和一位热衷于玩智力和解谜游戏的家庭主妇，三个人一共集资了大约三万日元（折合大约 222 元人民币），创办了一本名为 **Nikoli** 的杂志。

当时公司名号都没有，杂志第一次就只印了 500 本，有趣的是，他竟然连价格都没有印在杂志上，于是他就去挨家挨户推广、免费送杂志。送了大概两百本的样子之后，就有一家玩具店，打电话询问他，想用一百日元购买这本杂志，他高兴了起来，从此，他坚定了日本对于谜题的市场需求。

他开始渐渐不断到各大书店推销杂志，也渐渐有了信心，他于三四年后辞掉了这份工作，并真正创办了 **Nikoli** 的出版公司，并直到现在。

非常丰富的想象力的他，开始为数独这款当时没有恰当名字的游戏取名。他从规则联想到一位数、一个人，然后想到单身。最后，得到的一个名字——数字具有唯一的限制。名字太长了，就慢慢缩短到如今的数独二字。于是，于 1984 年，数独游戏的题目开始有了正式的名称进行杂志上的连载。

1-4-3-4 西尾彻也 (Nishio Tetsuya)

西尾彻也 (1954-)，生于日本和歌山县，是数独、谜题界的先驱级人物。他在 80 年代初期就已接触到名为 **Number Place** 的游戏（当时的数独被称为“填数字”，即 **Number Place**），并为日本最早的谜题杂志《**PUZZLER**》发表原创的数独题。此后发明了更加奇特的谜题，例如**数图** (**Nonogram**) 等。

他曾任日本解谜大赛的谜题制作委员长，并在 2006 年举办的第一届**世界数独锦标赛** (**World Sudoku Championship**，一般简称 **WSC**) 荣获第 4 名，解题实力宝刀未老。他被英国泰晤士报和德国《**STERN**》杂志誉为“谜题大师”。有一个数独技巧的名称也来源于他，比如西尾彻也强制链（同数强制链，在后续的内容会提到）。

最近，西尾彻也和科学出版社合作（实则是日本 **Nicoli** 公司和科学出版社合作），并出版了《世界最美超难数独》一套（五本）书籍。

Part 2 排除 (Hidden Single)

在了解了基本的数独元素和约定后，下面我们正式开始讲解数独技巧。

第一个技巧是**排除 (Hidden Single)**。我们可以从规则直接得到这种技巧的推理逻辑。试想一下，规则要求每一个区域必须填入 1 到 9 各一个，这也就意味着每一个区域里必须包含数字 1 到 9，也就意味着每一个区域的 1 到 9 都不可以缺失任意一个数。这便产生了这个技巧的相关内容。

排除有时也称为摒除。

2-1 宫排除 (Hidden Single in Block)

下面我们来看一则示例。

9			4			6	1	3
3	2		1	9		7		
						8		9
				1	7			8
7			3	6				
8								
		9		4	5		8	6
2	5	3			1			4

如图所示，观察 b3，发现 b3 一共有四处空格。不过，可以观察到，数字 8 只有唯一一处可填位置 r3c7。首先，r2c8 不允许填 8 的原因是，r8c8 是 8，这样 c8 上的其它任意位置就不再允许填入数字 8，当然也包括 r2c8；同理，由于 r4c9 是 8 的缘故，这使得 r2c9 不能是 8，毕竟 r4c9 和 r2c9 同列（同列不能出现两个相同数字）。

这样便排除了三种填数可能，故只能使得 r3c7 = 8，毕竟刚才说过，1 到 9 必须都得出现一个，要是数字 8 没有出现在 b3 里，b3 就必须存在有不是 8 的其它数字出现至少两个，这便违背了数独规则，因而 r3c7 还真的是不得不填 8 了。

2-2 行列排除 (Hidden Single in Line)

因为区域分行、列、宫三种，所以既然有宫排除，就应该有行排除和列排除。行排除和列排除统称**行列排除 (Hidden Single in Line)**，不过有时候也叫做线性排除，因为它们的排除效果都是“一整条线”形式的排除。

2-2-1 行排除 (Hidden Single in Row)

3			8	9	1	2		
	8	9	2				1	3
4	1	2	5	3		7	9	8
6	9		1		8	3	7	2
1		8	3		2	9		
2	3				9		8	1
	2	3		1		8		9
9				8	3		2	
8			9	2	5		3	

如图所示，可以观察 r3，发现 5 的位置只能填入到 r3c4。因为 r3 里只有 r3c4 两个单元格是空格。而 r3c6 不能填入 5 是因为 r9c6 是 5，这使得 c6 里的其余任何位置都不能是数字 5，当然也包括了处于 r3 上的 r3c6。所以，r3c4 = 5。

2-2-2 列排除 (Hidden Single in Column)

3			8	9	1	2		
	8	9	2				1	3
4		2		3		7	9	8
6	9		1		8	3	7	2
		8	3		2	9		
2	3				9		8	1
	2	3		1		8		9
9				8	3		2	
8			9	2	5		3	

如图所示，观察到 c8 里填入 1 的位置只有一处是不违背数独规则的：r2c8。其余的单元格都会违背规则，r1c8 填 1 会和 r1 上的 r1c6 的数字 1 冲突；r5c8 不能填入 1 的原因是，与此同宫的 r6c9 也是 1；同样地，r7c8 不能是 1。所以最终填入 1 的位置被确定在了 r2c8。

2-2-3 为什么会先介绍宫排除，而不是熟悉的行或列的排除？

这个问题问得好。虽说宫这个说法和行还有列都相对生疏一些，不过先介绍它的原因很简单：因为很好聚焦。由于宫的“形状”的优势，它比起行和列都容易观察一些，比如，我们一眼就可以看到某宫的哪些位置是空格，而行和列得“扫一遍”才会看完，所以我并不建议你先学会观察行和列排除，再学习宫排除，因此索性把宫排除放在前面了。

至于观察，这里简单说一下。宫排除的观察方式比较轻松一些：通过图上的画线的方式来确定某个宫的某些位置不能填入。

当然了，你在自己做题的时候就用手轻轻画一下就可以了，没必要把线都画出来，人的脑袋是有短暂的记忆功能的，所以你画线的那短暂的若干秒的时间，你是能记住你刚才画了哪些位置的。

行列排除怎么好好观察嘛，这一点我们在后面的内容会讲到。

2-2-4 来几个练习题

下面给出四个题目，给你练手。这几个题目只用排除就可以完成。

9	8	2	1				7				1	7		4	6		5
					6				5				8			3	
3	6		2					4						9			
6	3	7			9						4	8		7		1	
													6				
			8				7	3	2		7		3		1	2	
5					1		4	3				6					
			3							3			9				1
	7				8	2	6	9	8		7	1		2	4		

1	7					3		9		1			2				
						6				7		5	1				
		3		9	1	8	7			9					5	4	
				4				8	7			4	9		6		
		7	1	6	9	2					9			7			
3				5							2		5	8			3
	1	5	4	8		7			5	6					3		
		6											8	7		2	
4		8					2	6				3				7	

Part 3 唯一余数 (Naked Single)

和排除不同，**唯一余数 (Naked Single)** 这种技巧并非针对行、列、宫作排除，而相当于针对单元格作排除。

3-1 一个示例

		5	4	6		9		2
	9		2			7		
2			7					3
5	2			3	7	6	9	4
				4		2		
9	4	7	5	2	6			8
6		9			2			1
		3			4		2	
4		2		7	5	8		

如图所示，仔细观察 r6c9 一格。数数 r6c9 能填哪些数字。数字 1 不行，原因是 r7c9 是 1；数字 2 不行，原因是 r1c9 是 2（或者你看 r5c7 和 r6c5 也可以）；数字 3 不行（r3c9）、数字 4 不行（r4c9）、数字 5 不行（r6c4）、数字 6 不行（r4c7 或 r6c6）、数字 7 不行（r6c3）、数字 9 也不行（r4c1）。

那么，既然 1、2、3、4、5、6、7、9 都不行，那只能填 8 了，所以 r6c9 = 8。这便是这个技巧所得到的结论。

注意，这个题目其实还有两处唯一余数技巧可以使用，请尝试找一下。

唯一余数简称**唯余**。

哦对，你是不是嫌讲解得太简单了？别着急，花式使用唯一余数的内容还在后头呢，所以唯一余数后面还会出现的。

3-2 怎么观察？

观察历来都是比较难受的东西，它一般都不能被好好地阐述出来，因为讲不清楚，所以这里介绍一种能方便解释也方便观察的折中的思维。

为了更好地说明清楚我想要告诉给你的东西，这里我再给出一个例子。

3	1	2	8	5	9	6	4	7	3	1	2	8	5	9	6	4	7
7	5	9	6	2	4	8	1	3	7	5	9	6	2	4	8	1	3
8			7	1	3	2	9	5	8			7	1	3	2	9	5
4		8		7	5	1	3		4		8		7	5	1	3	
			4			7						4			7		
	7					4		8		7					4		8
	9		1	4	6	3				9		1	4	6	3		
			3	9		5	6	1				3	9		5	6	1
6	3	1	5			9		4	6	3	1	5			9		4

如图所示，请你先观察左图，如果把 $r3c1 = 8$ 的结论去掉，你是否能看到它？

我的观察方式是这样的：由于宫排除更容易看到，所以我们着重去找宫内确定值比较多的宫，比如 **b16789** 这种。找这种宫有一个好处是，我们可以数数一下瞄到缺少的数字到底有哪些了，比如 **b1** 缺少的数字是 4、6、8。那么我们就分别去看 **r3c123** 三格，每一个单元格都有能影响到它的一部分单元格，比如 **r3c1**，**r3**、**c1** 和 **b1** 里所有非 **r3c1** 的单元格都能影响到 **r3c1** 的填数。换言之，这些单元格里任意一个单元格一旦出现了数字 **a**，都会使得 **r3c1** 不能填入数字 **a**。

明确这一点后，我们就针对 **r3c1**、**r3c2** 和 **r3c3** 三格，挨个都看一下它对应的这些单元格里，到底有没有可以排除的数字。比如 **r3c1** 的所在列 (**c1**) 里有数字 4 和 6（如右图所示），这恰好能够排除掉 4、6、8 的其中两种情况，所以我们就能够很容易地得到结论 **r3c1 = 8** 了。而我们在观察的时候，也可以利用这一点，来观察唯一余数。

当然，我在前面说到“观察空格比较少的宫”，这一点只是一个经验，如果你做题有经验后，你可能就不会采用我这种笨办法了，而是一眼就能瞄到哪个单元格有哪些填数情况。所以这一点不绝对，仅供参考。

另外，刚才说到的能影响某一个单元格的所有单元格，有一个专门的说法叫做**相关格** (**Peers/Buddies**)，每一个单元格都有 20 个相关格。

如果你对数字不敏感，我推荐给你一个网站：

<http://www.sudokufans.org.cn/finder.php>

这个网站专门用于练习唯一余数的数数操作，它会给你一个区域下的其中八个数字，然后你需要做的，只有找出 1 到 9 里唯独没有出现的数字。它的题目是无尽的，也就是说你随时都可以停止练习，并且这个网站的上方会给出你的做题数据。如图所示。

正确COR[2]错误INC[0]总数TOT[2]正确率ACC[100%]平均T/K[1.686]	
4	<input checked="" type="radio"/> 随机 <input type="radio"/> 横向 <input type="radio"/> 竖向 <input type="radio"/> 宫 <input checked="" type="radio"/> 彩色 <input type="radio"/> 黑白
3	
1	
8	
7	
2	
9	
6	

1

2

3

4

5

6

7

8

9

比如这个题目里，缺少的黑色空格是数字 5。你就只需要对着屏幕按下数字 5 的按键就可以了。不管正确与否，它都会跳转到下一题，并且记录下你的正确情况。

“正确[2]”表示对两题，“错误[0]”表示错误 0 个题目；“总数[2]”表示做了两个题，“正确率[100%]”表示做题目前的正确率是 100%，“平均[1.686]”表示每一题的平均耗时为 1.686 秒。

一般来说，我们需要练习到平均时间在 1.2 秒以下，唯一余数的观察能力就会比较强了。一般比赛的参赛选手的数数练习都在 1 秒内。

Part 4 区块 (Locked Candidates)

区块 (Locked Candidates) 是我们常用的数独技巧。

准确来说，它指的是一个结构。

4-1 宫区块 (Pointing)

接下来为了表示清楚逻辑，我们使用两个典型的示例来介绍这个结构。

4-1-1 宫区块 + 排除

			3	1				8
	7	3	6	9				
					4			3
	2	9						
		4		3		9		
3						8	1	
2			5	6			8	
				4	3	5	2	6
6			2	8	7			

如图所示，可以看到，数字 3 在 b6 里只有两处位置可填：r4c78。其它的情况都通过宫排除给排除掉了。现在我们虽然无法确定 r4c78 里到底是谁填 3，但是我们可以发现，它们既同宫又同行。那么根据数独的规则，我们可以确定，r4 上填入的 3 也只可能是在 r4c78 里，别无其它选择（否则 r4 里但凡出现不是在 r4c78 的数字 3，都会同时使得 r4c78 都不能填入 3，然后 b6 就没办法填 3 了，于是就违背了数独规则，形成矛盾）。这便使得 r4c1 不可填入 3。

这一步的结论很重要。我们着重观察 c1，并使用列排除。发现 c1 上能填入数字 3 的位置仅有 r6c1，所以 r6c1 = 3。

之所以该部分最开始提起的“区块是一个结构”，是因为它实际指的是这个技巧里的 r4c78(3)。由于这个结构产生于宫之中，所以称为宫区块。

我们使用“**单元格坐标(数字)**”的格式来表示单元格里的指定数值，比如 r4c78(3)就表示 r4c78 里的数字 3。这种写法大多都用来表示区块结构。

既然有区块排除，那就有区块唯一余数，下面我们来看第二则示例。

4-1-2 宫区块 + 唯一余数

9						8		
3	2	4	7	8	1	5	6	9
				9		3		
7			1	2	9			
1			3	7	8			5
8	9		5	4	6			3
		1		5				
6	8		9		4		5	2
		9						

如图所示，我们可以通过宫排除可以直接得到 **r79c1** 形成关于 2 的宫区块结构。那么由于 **r79c1** 又是同列的结构，所以这就使得 **c1** 里填入 2 的位置也只能是 **r79c1** 里。那么，与之同列的 **r6c1** 就不能填 2 了。

既然有了结论，我们就可以再通过唯一余数来一个一个数数，最终确定 **r6c1** 只可以填入数字 8，所以 **r6c1 = 8**。

可以看到，宫区块只是把排除升了下级，那么行区块和列区块这种东西又应该是什么样子呢？

4-2 行列区块 (Claiming)

所谓的**行列区块**，和线性排除完全一样，就是产生在行和列的区块。

和之前线性排除类似，有时候也叫它线性区块。

4-2-1 行区块 (Claiming in Row)

下面来看一个示例。

2				1	7	4	5	
5	3	7	4	2				8
4	1	9		5		7	2	3
				4			7	5
1	7						4	6
6	4			7				
		4		6		5	3	7
7				8	4		9	2
			7					4

如图所示。观察 r7，数字 1 的位置只有 r7c46，于是 r7c46(1)便形成了区块结构，恰好它们同宫的关系，使得 r9c5 \neq 1，所以，对 c5 使用列排除，发现 r1c5 是能填入数字 1 的唯一位置，故 r1c5 = 1。

4-2-2 列区块（Claiming in Column）

	3		1	5		2		9
			3	6			5	
7			4	9		6		3
		1	2	7	3	8		
			5	1	9			
	9	3	6	8	4	7		
1								8
3	2			4				
4		9			1		6	

如图所示，我们通过细致的列排除后发现，r125c1 都不能填入数字 5。这样便使得 c1 填 5 的位置只有 r46c1，于是 r46c1 形成了 5 的区块结构。

此时，区块产生于 c1 上，所以是列区块。然后利用唯一余数的方式，对 r6c2 数数，并发现 r6c2 只能填入 9（本应含有 5 这种情况的，但它被 r46c1(5)的区块排除掉了，因为 r46c1 除了同列以外，还同宫）。

那么这个技巧和之前的取名方式类似，既有区块也有唯一余数，所以它就叫区块唯一余数。

有没有发现它很难观察？行列区块和行列排除一样，都是很难观察的东西，那么，我们有一个专门用于观察的辅助技巧：**组合区块**（**Cascading Locked Candidates**）。

4-3 组合区块/级联区块（Cascading Locked Candidates）

		5	2	3	8		9	
			9			6		
			7					
8	2	6			5	9		1
4	5		1	9	2		8	6
1			8			4	5	2
					7			
		4			3			
	8		5	1	9	2		

如图所示，这是一个关于数字 6 的列区块，因为区块的逻辑我们已经完全讲解过了，所以逻辑就自行理解了，这里着重阐述新视角。

我们尝试着将区块的这一列转移到与之排除能排除到的两个宫 b25，你就会发现，b25 都恰好存在 6 的区块，且构成了矩形形状，如图所示。

		5	2	3	8		9	
			9			6		
			7					
8	2	6			5	9		1
4	5		1	9	2		8	6
1			8			4	5	2
					7			
		4			3			
	8		5	1	9	2		

从这个示例可以发现，b2 里 r3c56(6)是一个宫区块，而 b5 里 r6c56(6)也是一个宫区块，且它们构成了矩形形状。

这有什么用途呢？宫区块的推导是对行和列作排除，而两个宫区块我们就可以这么想：

r3c56 和 **r5c56** 两处都存在区块，而构成了矩形，这使得这两个区块的最终填数必然是“错开”的。即 **r3c56** 里要是 **r3c5** 填 6，那 **r5c56** 里的 **r5c6** 才能填 6；反之亦然。这样一来我们就能发现到，**c5** 和 **c6** 的其余单元格都不允许填入 6 了，因为 **c5** 里必然会有数字 6 出现在 **r35c5** 里；同理，**c6** 里也必然有数字 6 出现在 **r35c6** 里。所以，我们聚焦于 **r8c6**，它的填数可能只有 3 和 6，而 6 显然已经不可能了，所以只能是 3，故 **r8c6** = 3。

可以从实际上看出，它的推理逻辑显然跟区块已经没啥太大的关系了，毕竟跨了两个区域后，推理也发生了变化。不过，我之所以说它的诞生旨在解决行列区块和行列排除的观察，就是因为它利用了宫区块的观察角度来代替了行列区块（比如这里的列区块，我们使用了两个宫区块代替掉了）。可能你会问，既然两个宫区块，那显然个数已经比一个列区块要多了，这岂不是不划算？不，相反地，它其实很划算。因为这种区块结构的观察利用到了宫内的区块，而宫区块最终依赖于宫的排除，宫排除是容易聚焦的，所以宫内的结构，区块也好，排除也好，都比行列性的结构要好看很多，甚至有时候，观察到三四个宫排除或宫区块的时间也不一定能看到一个行列排除或行列区块。所以，你可以尝试使用它来代替。

至于为什么级联区块结构能够代替掉行列区块或行列排除，它的理论依据和证明将依赖于非常难的逻辑，按照难度顺序来的话，这里就暂时先不给出证明了。

级联区块技巧的“级联”一词取自微软公司 Access 软件里的**级联更新**（**Cascading Update**）和 UI 设计里**级联样式表**（**Cascading Style Sheet**）的级联，表示“相关联”的意思。有时候这个词也被翻译为“层叠”。

单词 **cascading** 的原形是 **cascade**。

Part 5 数组 (Subset)

现在来看一个新的数独技巧：**数组 (Subset)**。

5-1 隐性数组 (Hidden Subset)

隐性数组是我们常用的第一种数组结构，不过它和区块不太一样的地方是，它有**规格**（有时候也叫做数组的**阶**，**Size/Order**）一说。所以我们这里得分成多个示例来讲解。

5-1-1 隐性数对 (Hidden Pair)

4		1 5 6		9		7	1 3 6	8
		7	8	1		4		
	8			6			5	
8			1	3				7
				7				
1	7			2	8			5
	6	8		5	1		2	4
5	1	3	2	4	9	8	7	6
	4	2		8		5		1

如图所示，仔细观察 r1c38 两格。有什么异样吗？其实这两个单元格的填数并没有图上标注的那么多情况。

仔细观察 r1 里填入 1 和 6 的位置，你就明白了：通过外部 1 和 6 确定值的排除，我们可以快速发现，实际上 r1 里能放下 1 和 6 的地方，恰好都只剩下了橙色的 r1c38 两格。试想一下，假设我们在 r1c3 一格填入 1，那么 r1c8 就必须填入 6，不然 r1 里 6 的位置就没有地方放了；同理：如果 r1c3 填的是 6，那 r1c8 就得是 1 才行，不然 1 就没办法放了。

这样便使得 r1c38 的填数必须得是一个 1、一个 6。自然而然就发现了，其它不是 1 和 6 的数字，比如数字 8 显然就不能填到 r1c38 里了。

接着我们重新对 r1 使用关于数字 8 的行排除，发现数字 8 最终只剩下了 r1c9 一格可以放，所以 r1c9 = 8。

这个技巧的巧妙之处就是，利用了排除法确定了不少一个数字的最终填数位置，而是两个。这种结构从图上来看，把 r1c38 的数字全部标出来后也不能立马确定下格内 1 和 6 是确定了位置的，即必须通过排除。所以这种结构我们称为**隐性数对 (Hidden Pair)**，“隐性”体现在刚才说到的“必须通过排除才能看到，单元格内的填数情况标注出来无法立马确定”，而“数对”则是这个结构在涉及两格的时候，它的名字。

这个技巧称为隐性数对排除，因为用到了隐性数对和排除。

和区块的表示方式类似，我们可以把这个题目里的数对直接记作 $r1c38(16)$ ，其中“16”表示 1 和 6，而不是数字 16，因为在数独里，我们只会用到 1 到 9 的数字，不会超过两位数，所以数字之间无需任何的分隔符号，这一点简写方式和 $c38$ 的 3 和 8 类似。

5-1-2 隐性三数组 (Hidden Triple)

那么我们现在来看看规格涉及三格的例子。

			2		7			1
	3	9		5				4
	9		7		5			3
	5	7	6	8		9		2
					2			
5	2	3			1			
×	×	×	5	4	9	1 2 3 6	1 2 3 6	×
9		4				7 8	7 8	
						5	1 2 3 6	

如图所示，仔细观察 $b9$ ，你会发现数字 1、2、3 的位置只剩下 $\{r8c78, r9c8\}$ 三格可填。

要想在 $b9$ 放下 1、2、3 三个数，而且还各一个，那只能这样三个单元格里，一个 1、一个 2、一个 3。其它的任意情况都会使得三个单元格的填数“不够平衡”，导致有一个数要么多出来一次，要么有的数字根本不会出现。比如这三个单元格里出现了一个 7，那么必然就会使得 1、2、3 的其中有一个数字无法放到 $b9$ 里，使得填法直接违背了“1 到 9 每个数字都得各一个”的规则。所以，我们不能允许这样的事情发生，故只能是一个 1、一个 2、一个 3。

这样便使得 $\{r8c78, r9c8\}$ 三格只能填入 1、2、3 这三种数字。然后转去观察 $r8$ ，可以发现，现在 4 只有一处可以放，即 $r8c5$ ，其它的情况都被排除掉了。所以， $r8c5 = 4$ 。

当然，这个结构因为规格变为 3 的关系，我们称 $\{r8c78, r9c8\}(123)$ 为**隐性三数组 (Hidden Triple)**，当规格超过两格的时候，我们直接称为“ n 数组”，其中 n 表示规格数。当然了，一般建议 n 写成数字的汉字写法，即一般不习惯写成“3 数组”。

5-1-3 隐性四数组 (Hidden Quadruple)

9			1	6	4		8	
	7		9	8	3	2	1	5
8	1	3	2			9	6	4
	8			2				
5					1		7	
		1					4	2
	4		7	1	6			
						1 3 4 5 6 7 8 1 3 4 5 6		1 3 7 8 9 1 3 6
		7	8	9	2			

如图所示，和刚才的逻辑类似，我们可以通过排除法，发现 b9 里填入 1、4、6、7 的地方都只剩下了 r89c79 四个单元格。那么，为了保证 b9 里必须得出现 1 到 9 都各一个的话，四格必须是一个 1、一个 4、一个 6 和一个 7。

这里啰嗦一点。你可能注意到了，它和之前的示例有所不同的地方是，r89c79 四格并不是所有单元格都恰好包含全部的 1、4、6、7 的，比如 r9c7 只有 1、4、6，没有 7，这一点是否会影响我们继续往下推理的逻辑呢？**不影响**。因为它不重要。之所以说它不重要，是因为我们只需要确定下 1、4、6、7 在里面确实是各有一个就行了。在形成结构后，把这四格不是 1、4、6、7 的其余情况全都给它排除掉就可以了，而具体它的内部到底少了哪些数字，从刚才的推理里我们也应当看出，其实我们并不关心这一点；当然了，如果数字少到直接都可以使用排除和唯一余数技巧了的话，那就另说了。

接着观察 r9，就可以发现数字 8 只剩下 r9c4 可以填了，所以 r9c4 = 8。

这个例子用到了规格是四格的情况，所以称为**隐性四数组 (Hidden Quadruple)**。

隐性四数组经常出现像示例里这样，结构里部分单元格缺少一部分数字的情况，但因为它不影响我们的推理，所以我们表述的时候依然可以使用 r89c79(1467)的方式，即使这个表达看起来好像是必须 r8c7、r8c9、r9c7、r9c9 四格必须都得包含全部的 1、4、6、7 一样。

5-1-4 怎么观察？

在示例里，其实我们已经提到了它的观察，它的观察就是利用排除（行、列、宫排除）来寻找隐性数组结构。排除是相对于唯一余数看起来似乎简单一些的技巧，所以我们可以利用这一点来观察到更多的排除结论。

5-2 显性数组 (Naked Subset)

讲完了隐性数组后，可以发现隐性数组用了排除来作为辅助的操作，那么下面要讲的第

二种数组结构——显性数组，可以猜得到，肯定用的是唯一余数了。

5-2-1 显性数对 (Naked Pair)

	9			7		³ 8	2	6
7		8		3		4	9	1
1								5
				5			6	8
6			9	1	8			
	8			2				
5						2		7
8	7	3		4		6	1	9
2	1			6		³ 8	5	

如图所示，仔细观察 **r19c7**，对 **r19c7** 两格使用唯一余数的办法，可以发现，这两个最终都只能填入 3 或 8。它们刚好同列，这就使得 **r1c7** 是 3 的话，**r9c7** 就只能是 8；反之亦然。

不论怎么填，这两个单元格显然是只可以填入 3 和 8 的，而且填的数字还不一样。这样一来，我们就能保证 **c7** 里数字 3 和 8 必须出现在 **r19c7** 里了，进而可以得到 **c7** 里的其余单元格都不能是 3 和 8。

接着我们可以利用这一点，对 **r4** 作行排除，数字 8 此时只有 **r4c9** 可填，所以 **r4c9** = 8。

这个例子优秀的地方在于，**r19c7** 不像刚才的隐性数组一般，它对区域其它单元格作出了填数情况的排除，并非自己涉及的单元格。之所以叫显性数组，就是因为它的结构里没有其它杂七杂八的数字，比如这个例子里，**r19c7** 里只有 3 和 8，很“干净”。这个例子涉及两格，所以叫做**显性数对** (Naked Pair)。

5-2-2 显性三数组 (Naked Triple)

3	9			1 2 6		7		
						6	5	
5		7	1 2 6	1 2 6	8	3	4	9
	4	9	3	8		5		6
6		1		5	4	9	8	3
8	5	3				4		
9			8			1	3	4
		2	9	4		8	6	5
4						2	9	7

如图所示，仔细观察 b2，通过唯一余数技巧数数你就会发现，{r1c5, r3c45}三格只能填入 1、2、6。由于它们同处于一个宫，所以它们的填数一定互不相同；而且刚好三格，填入的数字就只可能是三个不同的数字 1、2、6，也就是一个 1、一个 2、一个 6。

我们此时转去观察 r4c6，由于 r4c6 和 {r1c5, r3c45} 同处于一个宫，所以它的填数就不能是 1、2、6 了。那么，我们对 r4c6 再一次使用唯一余数，最终发现 r4c6 只能是 8，所以 r4c6 = 8。

这个结构和显性数对的思路是类似的，不过结构占了三格，所以我们称为**显性三数组 (Naked Triple)**。

5-2-3 显性四数组 (Naked Quadruple)

5	3	2	7	8	6			
9	7	8	2	4	1		6	
6		1	9	5	3	2	8	7
	2	5	4			6	7	
		3	6	1	7		5	2
7			5					
		4 6 7 9	1					
	4 9 7 9	4 7 9 9	8		5	1		6
		4 6 7 7	3				9	8

如图所示，这个例子有点难。着重观察 b7 里的 {r8c2, r789c3} 四格，这样四格恰都

只包含 4、6、7、9 四种不同的数字，别无其它。由于它们同处于 b7，这使得它们内部的填数是不可以一样的，又因为恰好四格的缘故，所以最终四格的填数只能是一个 4、一个 6、一个 7 和一个 9。

由于 r789c1 和这四格同宫，所以 r789c1 显然是不能填入 4、6、7、9 的任意一个数的。此时我们观察 c1，最终我们确定 r3c1 是填入 6 的唯一一处地方，所以 r3c1 = 6。

这个例子就是**显性四数组**（Naked Quadruple）了。

5-2-4 为什么没有五数组？

我们已经讲完了显性数组和隐性数组，不过你有没有发现，这些数组我们都没有讲到五数组甚至更高规格的数组结构，这未免会让我们引起怀疑。我可以先告诉你一个结论：**规格大于 4 的数组，不论是显性的还是隐性的，理论上都不存在**。至于其原因，我们将在下一篇章的数组里进行详细讲解。

5-2-5 怎么观察？

我们讲完了显性数组，可以发现，显性数组的观察难度比隐性大太多了，首先是我们刚入门学习了唯一余数没多久，觉得唯一余数不如排除好观察；其次是这个技巧依赖于唯一余数的数数操作，所以这使得我们可能更加难以观察到它们。不过实际上，我们有一个稍显轻松的方法。

	8			1	7			6
7	5	3		9		8	4	1
1		6				2		7
6	7	8						4
3			7			1		9
						7	6	8
8		1				4	7	5
5		4		7		9	1	3
9	3	7	5	4	1	6	8	2

例如这个例子而言，橙色两格是一个显性数对，而我们可以从例子里发现，这两格的所属宫（一个 b4，一个 b6）的大部分确定值都是一样的，这使得我们引起警觉：由于大部分提示信息都是一样的，那么某两格的候选数也就应当大概率出现一样的情况。此时我们就找到 r5c3 和 r5c8 两格，除了 6、7、8 相同以外，3 和 9 的确定值也都恰好出现在 r5 上，这便使得 r5c3 和 r5c8 又有两个情况被同时排除掉，这样数下来，r5c38 此时就只能是 1、2、4、5 了。

这个时候，我们又可以借助排除来获取信息：比如 r5c3 和 r5c8 的所在列都有 1 和 4 的确定值的提示信息，那么两格就只剩下 2 和 5 了。这便就形成了显性数对。

所以我们可以总结一下观察方式：找出同一个区域的若干单元格，并大概扫描一下这若干单元格的所在宫（宫比较容易聚焦），来看它们的提示信息是否是真正的大部分都一样（不一定确定值都是完全一样的，只需要大部分相同就好）；如果有这种情况出现，我们就去看每一个单元格各自的所在行列看看是否真正有额外的影响，并最终来确定这若干个单元格究竟是不是显性数组。如果是，那么恭喜你，看看结论到底在哪里；如果不是，也不要灰心，我们依然可以利用这个规则来看看其它地方。

5-3 数组内区块

之前我们学了数组，也学了区块，那么数组能不能和区块一起用呢？我们来看看这个例子。

1	5	4	7		3	2		
	7	2	1					4
	3		4		2			
	4			2	8			
2	8	5 ³	9	1	7		4	6
	1		3	4			2	
	2	5 ³ 8		7	1		9	
7	9	5 ³ 8			4	1		2
	6	1	2	3	9	8	5	7

如图所示，我们可以看到这个示例里，r578c3(358)是一个显性三数组。不过其中我们发现，数字8只能出现在r78c3里，而r78c3又恰好同一个宫，于是b7其余单元格也就不允许填入8了，看起来这个r578c3除了三数组的结论，还带了一个推理很像区块的结论。得到这个结论后，我们就可以利用r9的行排除得到最终的r9c7 = 8了。

这里我们看到，这种数组里还外带了一个区块结论。目前，这种数组尚未拥有新的结构名，我们一般把这种区块称为**数组内区块**，即嵌入数组的区块结构，而整体的技巧则称为**区块数组**（Naked Subset With Locked Candidates）。

5-4 死锁数组（Locked Subset）

当然，有一种数组结构非常特殊，这种数组还有一种专门的观察方式和角度。

5-4-1 死锁数组的基本结构形式

	9			8	1	4	2	5
4	5		9					
	2			4	5			
	4	9					7	8
					7			
8	7					2	6	
			3	6		9	5	2
					9	7		6
9	6	5	2	7				

如图所示，不妨我们观察 b2 和 c4。首先 b2 要填入 1 到 9 各一个，而 c4 也是。它们除了共用 r123c4 三格以外，其它的单元格都不同。但是，既然它们都要填 1 到 9，而且 r123c4 又是共用的，那这不就说明了，r123c56 和 r456789c4 的填数必须得是一一对应的。也就是说，r123c56 里填的是哪 6 个数，r456789c4 就得是哪 6 个数。

这一点很重要。我们再次观察 r456789c4，发现里面只有两个提示数 2 和 3，它们恰好在 r123c56 里都没有出现。r123c56 里还剩两个空，这不就明示这俩空格必须一个 2 一个 3 了吗？换句话说，r456789c4 里的空格就必须得填 1、4、5、8 了。

此时，我们对 b5 作排除，就可以确定 7 的位置了。

可以看到，这个例子里，7 的位置的确定除了依赖于排除，还依赖于刚才区域填数相同的结论。实际上，利用区域填数一致的逻辑在一些其它变体类型的数独题里称为 **Law of Leftover**，中文如果直译则直译成“剩余数法则”，而我比较喜欢叫它割补，因为它的逻辑是割了一个区域补到另一个区域的方式。而实际上，这个例子里产生了两个数对，一个是列上的四数组，一个是宫内的数对。但是……实际上这个所谓的割补法数组指的并不是这两个数组，而是下图给的这个 6、7 数对。

	9		⁶ 7	8	1	4	2	5
4	5		9					
	2		⁶ 7	4	5			
	4	9				7	8	
					7			
8	7					2	6	
			3	6		9	5	2
					9	7		6
9	6	5	2	7				

这个 r13c4(67)的显性数对使用另外一种视角则是这样的：形成数对后，使得 r5c3不能填入 6 或 7，然后通过 b5 的宫排除，就可以确定 7 的位置了。

在技巧资料和一般的文章里，这种数组结构有一个专门的名称，叫做**死锁数组**（Locked Subset）。而割补法并非标准数独的技巧，而是一种叫做**锯齿数独**（Jigsaw Sudoku）的变体类型数独里的一种特殊技巧。

可以从例子里发现，这种数组最大的规格只能是三格，因为它无法跨出区域形成数组结构，否则就无法产生割补的效果了。那么我们不妨再来看一则示例，这个示例就是规格为 3 的情况。

	9			2	1	3	8	
4		2		6		5	1	
	3			⁴ 7	⁵ 8	5	2	
	2		9		³ 6			
			2		4			
			¹ 8	³ 6	7		2	
				¹ 4			9	
	1	7		⁷ 8	3			2
	4	8	5	¹ 9			7	

这个例子就自己推理了，思路和刚才的逻辑完全一样。

5-4-2 死锁数组引出的割补法的高级用法

既然我们说到，死锁数组利用的是割补法的思维，那么下面来看一下，我们如何把割补法用得更加高级一些。

2	8	5	4	7	3	1	6	9
1	7	6	2	8	9	3	5	4
3	9	4	5	1	6	7		
		9					1	
	1	8		4		6	9	
	3	2		9	1	4	7	
		3	1	5	7	9		6
		1	9	6	4		3	7
9	6	7				5	4	1

如图所示，我们可以利用 r4 和 b6 的割补法来得到结论，如果你能找到结论出现在 r4c2 上，那么恭喜你，你完全可以灵活使用这个技巧了；如果不能的话，也不要灰心。我们来看看，这个结构到底如何使用。

首先，我们通过最初的割补法，可以立马得到 r56c789 里的填数一定和 r4c123456 的填数是一样的。那么，我们发现，r56c789 里有 4、6、7，在 r4c123456 里都是没有出现的，所以我们就来看下，它们的填数情况。

2	8	5	4	7	3	1	6	9
1	7	6	2	8	9	3	5	4
3	9	4	5	1	6	7		
		9	×	×	×		1	
	1	8		4		6	9	
	3	2		9	1	4	7	
		3	1	5	7	9		6
		1	9	6	4		3	7
9	6	7				5	4	1

那么，我们先着重观察数字 4 的填法。4 在 r4c123456 里只能放在 r4c12 上，因为 r4c456 都被 b5 这个宫内的 r5c5 的提示数 4 排除掉了，所以我们只能把 4 放在 r4c12

上。

2	8	5	4	7	3	1	6	9	2	8	5	4	7	3	1	6	9
1	7	6	2	8	9	3	5	4	1	7	6	2	8	9	3	5	4
3	9	4	5	1	6	7			3	9	4	5	1	6	7		
		9					1				9					1	
	1	8		4		6	9			1	8		4		6	9	
	3	2		9	1	4	7			3	2		9	1	4	7	
		3	1	5	7	9		6			3	1	5	7	9		6
		1	9	6	4		3	7			1	9	6	4		3	7
9	6	7				5	4	1	9	6	7				5	4	1

另外两则情况则完全同理。6 和 7 的填数位置都仅能放在 r4c14 上。

所以，实际上 4、6、7 只能放在 r4c124 上，这是满足隐性三数组的数组定义的，所以它们构成了隐性三数组的结构，于是我们就可以通过唯一余数，得到最终的填数结论在 r4c2 了。

2	8	5	4	7	3	1	6	9
1	7	6	2	8	9	3	5	4
3	9	4	5	1	6	7		
	4	9					1	
	1	8		4		6	9	
	3	2		9	1	4	7	
		3	1	5	7	9		6
		1	9	6	4		3	7
9	6	7				5	4	1

由于构成了隐性三数组，所以 r4c124 里只能放入 4、6、7，而 r4c2 只能放入 4，所以 $r4c2 = 4$ ，便是这个技巧的结论了。

当然，你也可以发现，6、7 在刚才的推导过程里，已经构成了隐性数对，那么就不用这么麻烦地推导到这里，再通过唯一余数得到结论了。这只是其中的一个参考思路。

Part 6 复合结构技巧 (Complex Pattern)

前面已经讲到了一些复合结构，例如一个排除需要动用区块和数组，这种东西被我们称为**复合结构 (Complex Pattern)**。显然，复合的方式千变万化，也就应该拥有千变万化的组合，形成千变万化的变体。那么现在我们就找出一些非常常见的复合方式，或者是比赛的例题来作讲解。

注意，比赛的题目是不能随意使用的，这里的题目都经过了授权（当然不需要授权的题目就不管它们，需要授权的则必须经过同意和协定来合理合法使用）。

6-1 宫区块 + 行列区块 + 排除

有一些时候，区块之间是可以套用，或者说有递推关系的。比如下面这个例子。

			6					4
								6
	6	2	×	1	4	9	3	
8	2			7	6		9	1
1	5	9	8	4	2	7	6	3
6			1	9			2	
2		6		5		1	4	
4				×	1			
3			4		8			

如图所示，我们先可以观察到 b9 里存在一个 3 的宫区块，于是通过排除，我们可以得到 r8c5 \neq 3。

接着，我们继续观察 c5，发现 c5 此时只能把 3 放在 r12c5 里，所以此时它们形成列区块结构，而因为它们同宫，所以 b2 里其余单元格都不能放下 3。所以此时 r3c4 \neq 3。

最后，正是因为这一点，我们观察 r3，并最终确定了 3 的真正填数位置。通过排除，我们最终确定 r3c9 是唯一的能放 3 的单元格，所以 r3c9 = 3。

这个例子巧妙的地方在于，它连着使用了两个区块，一个是宫区块，而另外一个则是行列区块。

6-2 数组 + 区块 + 排除

		1	3	1	2	3	8	4				7
		6				9						
	5		1	2								3
		4			8	9						
		7					3			1	9	
	2								3	4	6	
9				5	6	2						1
	8	6	3			4	9	2	5			
3	7		4				6					
										3		
2				3	6							

如图所示，这个例子是一个典型的数组+区块+排除的特别厉害的示例。首先，我们可以通过 2、3、9 对 b1 的排除，得到{r1c23, r2c3}(239)形成隐性三数组结构。

在得到数组结论后，r1c23 和 r2c3 就不允许填入不是 2、3、9 的其它数字了。接着我们就可以在 b1 里发现数字 1 的区块结构，位于 r12c1（r3c1 不能填 1，因为 r3c8 是 1）。

那么，得到区块后，我们再对 r6 使用排除法，确定了 1 的填数位置最终只能是 r6c5，所以 r6c5 = 1。

这个例子先使用了数组，然后得到了区块结构；然后区块后又得到了行排除才出现了结论。

6-3 区块 + 唯一余数

现在我们来看一道题目。为了讲解这个技巧，我专门出了此题。

1				8				4
		7			2	8		
	2				4		5	
			3					
7		5		9		4		8
					6			
	1		6				4	
		6	1			5		
3				5				2

我们针对 **r5c5** 作唯一余数的数数操作。其中数字 1 和 2 没有提示信息，而 3 到 9 在 **r5c5** 的相关格里都有对应的提示信息，所以 **r5c5** 只能是 1、2、9 的其一。

然后，我们在 **b2** 里可以发现，1 的填数位置只能是 **r23c5**，所以 **r23c5(1)** 形成区块，致使 **r5c5** \neq 1；而对应的对称的 **b8**，也有一个 2 的区块。所以最终 1 和 2 都不能填入到 **r5c5**，所以 **r5c5 = 9**。

这个题巧妙的地方在于，第一步就必须使用这个技巧，否则要想绕过这个技巧，必须使用死锁数组或三数组结构。所以这一点来说，我还是很自豪的。

6-4 数对 + 区块 + 唯一余数

我们再来看一个比赛的题目。

	6	3	4					
			6					4
		8			7	1	6	9
	8	7					2	3
2 3	2 3		8	7			1 5	1 5
1	5					8	4	
6		2	5			4	9	8
8		5		4 6	9			
				4 6	8	5	7	

这个题目在出了一部分数字后，依然无法继续往下填数。此时得需要一个唯一余数技巧来瓦解题目。

正如图上标注的一样，**r12c5** 是一个 9 区块，**r5c12(23)** 是一个显性数对，**r5c89(15)** 是一个显性数对，**r89c5(46)** 是一个显性数对，再配合 **r5c4** 的确定值 8，我们最终确定了 **r5c5 = 7**。

至此，直观类技巧我们就讲完了。直观的部分，剩下的就得靠你自己了。

第二篇章 定式候选数技巧

从这个篇章开始，我们将为大家介绍到的是候选数技巧。所谓的**候选数**（Candidate），就是我们之前提到的，单元格里的填数情况。

这个板块给出的技巧，都是超过一般比赛难度的技巧，它们也是完成难题的基础。

Part 1 候选数初步

为了叙述方便，我们先把后面要说到的内容全部大体的框架说一下。

1-1 做题模式

首先，题目大致分为如下三种做题模式：

- **直观**（Direct）；
- **局部标记**（局标，Partial Marked）；
- **全部标记**（全标，Full Marked）。

这三种做题模式都有自己的策略。直观是直接通过空白的盘面上手就开始做题的手段；而局部标记就是之前数对里使用的方式，在一部分单元格里标注出候选数来达到辅助做题的目的；而全标就是全盘的所有空格的所有候选数都标注出来。而我们后续的讲解，都是采用全标的形式来解释和描述的。因为后面的技巧一般都较难，所以标注出它们是非常有效的（直观层面一般都做不了）。

1-2 出数和删数

接下来说一下，什么是**出数**（Assignment），什么是**删数**（Elimination）。出数指的是能够得出填数结论的行为，比如之前的技巧，利用区块和数组来达到排除和唯一余数的结论，这种就叫出数结论；而删数，则是针对于候选数而言。得到的推理结论仅仅是排除一个候选数情况的行为（我们称为删除或删减情况，所以叫做删数，而不是“排数”）。

介绍了这么多后，我们开始正式进入候选数技巧的学习。不过为了我们能够更好过渡到候选数层面（全标层面），我们依然先得介绍区块和数组结构。

不过，区块和数组的逻辑已经讲得比较清楚了，这里的全标示例仅给出图片和一些简单的逻辑解析。

Part 2 区块 (Locked Candidates)

我们先来看看区块的全标状态下，到底长啥样。

2-1 宫区块 (Pointing)

5	2	⁴ ₈	3	1	^{7 8} _{4 7 8}	6	9
6	3	1	4	^{7 8}	9	2	5 _{7 8}
7	⁴ _{8 9}	⁴ _{8 9}	5	2	6	⁴ ₈	1 3
^{2 3} ₄	7	6	9	³ _{4 8}	^{1 2} ₈	^{1 3} _{2 3}	5
^{2 3} ₄	¹ ₄	5	^{1 2} _{6 8}	³ _{4 6}	^{1 2} _{7 8}	^{1 3} _{7 8}	^{2 3} ₁
^{2 3} ₉	1	^{2 3} _{8 9}	^{1 2} ₈	³ _{7 8}	5	6	4 ¹ _{7 8}
1	⁶ ₉	^{2 3} ₉	7	⁶ ₈	² ₈	5	³ ₉ 4
^{2 3}	5	7	^{1 2}	9	4	^{1 3}	8 6
8	⁴ _{6 9}	⁴ ₉	¹ ₆	5	3	¹ ₉	7 2

第一则示例是宫区块，因为转变了视角，所以结论就不再是出数的了，而是删数：

宫区块：r5c78(9) => r5c12 <> 9

我们使用推出符号“=>”表示前者可以推出后面的结论，所有后面的结论使用到了这个符号都算作是推出符号。

2-2 行列区块 (Claiming)

2 3 5 8 9	3 6 9	2 6 5 9	4 5	1	7	2 4 5 8 9	2 4 5 6 9	5 6 8
2 5 8	7	4	6	9	3	2 5 8	1	5 8
1	6 9	5 6 9	4 5	2	8	3	4 5 6 9	7
5 7 9	1 9	1 5 7 9	2	3	6	1 5 9	8	4
6	2	1 5 9	8	4	1 9	1 5 9	7	3
4	8	3	7	5	1 9	1 2 9	2 6 9	1 6
2 7	1 4 6	8	3	7 6	2 5	1 4 5 7	4 5	9
7 9	5	7 9	1	8	4	6	3	2
2 3 4 6	1 3 4 6	1 2 6	9	7 6	2 5	1 4 5 7 8	4 5	1 5 8

如图所示，描述如下：

行区块：r8c13(7) => r7c1, r9c13 <> 7

结论里（推出符号后的部分）即使出现多个单元格，也不需要使⤵大括号括起来，例如结论里 r7c1 和 r9c13 指的是同时都可以删除 7，而并不是强调这些单元格的候选数 7，所以此时的大括号是可以省略的（当然也可以写上去）。

Part 3 数组 (Subset)

接下来我们来快速浏览几则数组示例，然后提一些在前面数组没有说到的东西。

不过，因为改变为全标视角后，唯一余数就显得观察起来相当简单，所以与之有所关联的显性数组此时被我调整在了隐性数组之前，先给出例子。

3-1 显性数组 (Naked Subset)

3-1-1 显性数对 (Naked Pair)

先来看显性数对：

⁵ 7	⁵ 8	^{7 8}	1	4	6	^{2 3}	³ 9	^{2 3} 9
4	6	1	³ 9	³ 9	2	7	5	8
2	3	9	7	8	5	¹ 4	¹ 6	¹ 4
9	¹ 4	5	2	6	3	¹ 4	8	7
3	7	6	8	1	4	9	2	5
8	^{1 2} 4	² 4	⁵ 9	⁵ 9	7	6	^{1 3} 4	^{1 3} 4
1	² 5	² 7	6	^{2 3} 5	9	8	4	^{2 3}
6	9	3	4	² 7	8	5	¹ 7	^{1 2}
⁵ 7	^{4 5} 8	⁴ 7	³ 5	^{2 3} 5	1	^{2 3}	³ 6	^{2 3} 6

显性数对：{r7c9, r9c7}(23) => r8c9 <> 2, r9c8 <> 3

3-1-2 显性三数组 (Naked Triple)

2 3 5 9	1	5 6 9	7	2 3 4 5 9	3 5 6	8	2 3 4 6 4	2 3 9
2 3 9	4	6 9	2 3 6	1 2 3 9	8	1 2 6	7	5
2 3 5 8 9	2 5 6 8	7	2 3 4 6	1 2 3 4 5 9	1 3 5 6	1 2 6	2 3 4 6 4	1 2 3 9
1 5 7 8	5 7 8	5 8	4 8	3 4 8	2	9	3 8	6
2 8	9	3	5	6	7	4	1	2 8
4	2 6 8	1 6 8	9	1 3 8	1 3	5	2 3 8	7
1 5 7 8 9	5 7 8	1 4 5 8 9	2 4 5 8	2 5 7 8	5 6	3	2 4 6 8	1 2 8
6	3	4 8	1	2 7 8	9	2 7	5	2 4 8
1 5 7 8	5 7 8	2	3 6 8	3 5 7 8	4	1 6 7	9	1 8

显性三数组: r479c2(578) => r3c2 <> 58, r6c2 <> 8

3-1-3 显性四数组 (Naked Quadruple)

1 3 8	1 4	9	2	4 8	7	4 8	5	6
2 3 5 7 8	2 4 5 6 7	5 6 7 8	4 5 6 8	3 4 6 8	1	2 4 8 9	2 4 9	2 4 8 9
2 8	2 4 6 8	6 8	5	4 6 8	9	7	3	1
4	6 7	6 7 8	7 8	9	2	5	1	3
1 2 7	9	1 7	4 7	5	3	6	8	2 4
2 5 8	2 5	3	1	4 8	6	2 4 9	2 4 9	7
1 7 9	8	2	6	1 7	5	3	4 7 9	4 9
1 5 7 9	1 5 7	1 5 7	3	1 2 7	4	2 8 9	6	2 8 9
6	3	4	9	2 7	8	1	2 7	5

显性四数组: r2c4789(2489) => r2c12 <> 2, r2c25 <> 4, r2c135 <> 8

3-2 隐性数组 (Hidden Subset)

3-2-1 隐性数对 (Hidden Pair)

8	4 5 6	4 5 6	9	4 6	3	1	7	2
7	9	1	5	2	8	4 3 6	3 6 4	3
4 3 6	2	4 3 6	1	4 6	7	9	5	8
4 5 6	8	4 5 6	2	3	1	2 5 6	6 5 9	7 9
2	1	5 3	7	9	6	8	4	5 3
4 3 6	4 6	4 6	2 3 4	8	5	2 3 6	1	3 7 9
9	3	4 5	6	7	2	4 5	8	1
4 5 6	4 5 6	8	3	1	4 9	7	2	4 5 9
1	7	2	8	5	4 9	4 3 9	3 9	6

隐性数对: r46c3(79) => r4c3 <> 456, r6c3 <> 346

隐性数组的候选数视角有一点不好入门。这里简单说一下。

从候选数角度来看, 可以发现 b4 里, 能放 7 和 9 的位置只有 r46c3, 由于同一区域的关系, 于是它们就形成了隐性数对了。

3-2-2 隐性三数组 (Hidden Triple)

7	2 9	3	1	2 8	6	4	5	8 9
4 5 6 9	2 5 6 9	2 6 9	5 8 9	4 8 9	7	1 2 9	3	8 9
4 5 9	1	8	5 9	4	3	6	7	9
1	2 5 6	2 6	3	9	8	7	4 6 4 5 6	6
6 8 9	3 6 7 9	6 7 9	2	5	4	1 3 1 6 8 9	6	6
5 8 9	5 8 9	4	7	6	1	5 3 9 8 9	2	9
2	6 9	1	4	7	5	8	3	6 9
3	6 7 9	6 9	1	2	5 9	4 6 4 5 6 9	1	9
6 8	4	5	6 8	3	9	2	7	1

隐性三数组: r568c2(378) => r5c2 <> 69, r6c2 <> 59, r8c2 <> 69

3-2-3 隐性四数组 (Hidden Quadruple)

3	9	5	4	2	1	7	6	8
7	² ₄	¹ ₈	3	6		¹ ₄	^{1 2} ₄	5
² ₄	² ₄	¹ ₆	7	5		3	^{1 2} ₄	² ₉
8	3	4	1	7	5	2	9	6
5	7	2	6	9	4	8	3	1
6	1	9	² ₈	³ ₈	^{2 3} ₈	^{4 5} ₄	^{4 5} ₄	7
² ₄	² ₉	⁴ ₅	⁶ ₇	⁸ ₉	³ ₈	¹ ₅	^{1 2} ₅	² ₉
1	² ₄	⁵ ₈	⁷ ₈	⁸ ₉	6	⁵ ₉	² ₇	3
² ₉	² ₆	3	5	1	7	⁶ ₉	8	4

隐性四数组: r7c2378(1567)

=> r7c2 <> 248, r7c3 <> 8, r7c7 <> 9, r7c8 <> 2

3-3 区块数组 (Naked Subset With Locked Candidates)

这里罗列两个区块数组的例子。

3-3-1 区块三数组 (Naked Triple With Locked Candidates)

5	¹ ₇	¹ ₇	³ ₆	¹ ₆	³ ₉	4	8	2
² ₄	3		² ₄	⁴ ₈	7	5	¹ ₆	¹ ₆
² ₄	^{1 2} ₄	¹ ₆	² ₄	¹ ₅	² ₆	3	7	9
6	9	¹ ₇	³ ₄	8	5	^{1 2} ₇	^{1 2} ₄	¹ ₄
³ ₄	¹ ₈	5	³ ₄	2	¹ ₄	³ ₆	¹ ₇	¹ ₆
² ₄	^{1 2} ₄	¹ ₈	9	7	¹ ₄	¹ ₆	3	5
1	5	2	7	⁴ ₉	⁶ ₉	8	⁶ ₉	⁴ ₉
³ ₉	8	³ ₉	1	⁴ ₉	⁶ ₉	5	⁴ ₇	⁶ ₉
7	6	4	^{2 3} ₅	³ ₅	^{2 3} ₉	^{1 2} ₉	^{1 2} ₉	8

区块三数组: r278c5(49), r78c5(6)

=> r3c5 <> 4, r13c5, r8c6 <> 6, r19c5 <> 9

3-3-2 区块四数组 (Naked Quadruple With Locked Candidates)

5 6 9	5 6 9	2 3	8	4	6 9	1 3 7	1 2 3
6 9	1	2 3 4	2 3 7	2 3 6	6 9	8	2 3 5
2 3 4 8	4 8	7	1	2 3	5	6	2 3 4
1 3 4	7	6	5	8	2	1 3 4	2 3 4
2 3 5 8 9	5 8 9	2 3 8	4	7	1	3 9	2 3 6
1 2 4 5 8	4 5 8	1 2 4	6	9	3	7	2 4 5 8
6 8	3	5	9	1	4	2	6 8
7	4 8	6 9	2 3 7	2 3 6	6 8	5	1 4 6 8
1 4 6 8	2	1 4 8	3	5	7 8	4 9	3 4 6 8 9

区块四数组: r4c789(14), {r4c789, r5c7}(39) =>
r4c1 <> 14, r5c8 <> 3, r5c9 <> 39, r6c8 <> 4, r6c9 <> 14

3-4 真 · 为什么没有五数组？

那么，回到直观类技巧里遗留的问题：为什么我们不具有五数组，或者更大规格的数组呢？这里我们将为大家解释这一点。

3-4-1 显隐性互补

有没有想过一种证明思路，就是通过显性和隐性的某种特性，使得一旦有显性的时候，隐性就一定不能出现超过规格为 4 的情况呢？

1 9	5 9	2	1 4 5	4 5 7	1 4 5 7	8	3	6	1 9	5 9	2	1 4 5	4 5 7	1 4 5 7	8	3	6
1 3 8	3 8	4	1 3 8	6 9	1 3 8	5	7	2	1 3 8	6 9	1 3 8	6 9	5	7	2	1 3 8	6 9
6	5 8 9	7	2	4 5 8	4 5 8	4	9	1	6	5 8 9	7	2	4 5 8	4 5 8	4	9	1
4 8	1	6	4 5 6 8	2	9	7	4 5 6 8	4 5 6 8	4 8	1	6	4 5 6 8	2	9	7	4 5 6 8	4 5 6 8
4 7 8 9	6 8 9	5	1 3 4 6	3 4 6	1 3 4 6	2	4 8 9	4 8 9	4 7 8 9	6 8 9	5	1 3 4 6	3 4 6	1 3 4 6	2	4 8 9	4 8 9
4 9	6 9	3	7	8	4 5 6 9	1	4 5 9	4 5 9	4 9	6 9	3	7	8	4 5 6 9	1	4 5 9	4 5 9
5	3 8	1	4 8 9	4	2	6	4 8 9	7	5	3 8	1	4 8 9	4	2	6	4 8 9	7
2 7 8	2 7 8	6	4 5 6 8 9	1	4 5 6 7 8	3	4 5 6 8 9	4 5 8 9	2 7 8	2 7 8	6	4 5 6 8 9	1	4 5 6 7 8	3	4 5 6 8 9	4 5 8 9
2 3 7 8	4	9	5 6 8	3 7	3 7 8	1	2 5 6	5 8	2 3 7 8	4	9	5 6 8	3 7	3 7 8	1	2 5 6	5 8

我们找出了一个稍微极端的例子。例如左图，我们可以发现它实际上是一个显性数对结构。不过我们拿出同样的例子，只是切换技巧，可以变成右图这样，图中的例子立马变为了

一个隐性数组。

细数右图的隐性数组，可以发现它的规格是 5，即一个五数组，而我们仅仅是把删数不动，把原来涂了色的数字变为不涂色，而不涂色的候选数变为涂了色。那么形成这样的原因是为什么呢？

一列一共是 9 格，除去两格已经有确定值的关系，所以只剩下 7 个空格。一旦这 7 个格子里有一个显性数组结构，那么必然会存在一个隐性数组，规格呢？规格就是抛开这个显性数组结构涉及的格子之外，剩下的单元格全部构成一个隐性数组，这么大的规格。

你可能会问：这样真的什么时候都可行吗？答案是显然的。我们从另外一个维度来看一则极端的例子。

4 5 8	3 5	1 8	7	9	2	6	1 3 4 4	1 3 4 4	4 5 8	3 5	1 8	7	9	2	6	1 3 4 4	1 3 4 4
7	3 1 2 9	1 2 9	6	8	4	2 3 4	1 2 3 4	5	7	3 1 2 9	1 2 9	6	8	4	2 3 4	1 2 3 4	5
4 8	6 8	1 2 6	3	5	9	1 2 8	7	4 8	4 8	6 8	1 2 6	3	5	9	1 2 8	7	4 8
6	5 9	3	5 9	2	4	1	7	8	6	5 9	3	5 9	2	4	1	7	8
5 8	1	8 9	5 9	7	3	4	2 6	4 2 4	5 8	1	8 9	5 9	7	3	4	2 6	4 2 4
2	4	7	6	8	1	5	3 9	3 9	2	4	7	6	8	1	5	3 9	3 9
9	2 7 8	4	3	1	7	6	5 7 8	2 6	9	2 7 8	4	3	1	7	6	5 7 8	2 6
3	6 7 8	6 8	2	5	7	6	4 7 8	4 9	3	6 7 8	6 8	2	5	7	6	4 7 8	4 9
1	2 7	6 5	8	4	9	2 3 7	2 3 6	2 3 6	1	2 7	6 5	8	4	9	2 3 7	2 3 6	2 3 6

我们来看这一则示例，首先从左图可以看到它是一个隐性数对；而我们对照右图可以发现，剩余四格构成了显性四数组。别急，你是不是看到，四数组里除了 r8c7 有一个 4 以外，其它格子都没有 4 啊？对，这就是这个示例要告诉大家的东西。

它实际上也属于数组，而且是一个合格的显性四数组结构，不过由于受到其它 4 的确定值的影响，4 最终在结构里只有一处可填。这本身没有问题。不过，因为它依然满足四个单元格里只有四种不相同的数字，所以它从这个层面来说，确实是四数组。只是，这里最终我们能确定一点，这一行其余位置的候选数 4 都删除掉后，4 就真的只有 r8c7 一处可以填了！所以除了刚才删数结论外，这种数组还有一个出数结果：r8c7 = 4，于是，四数组降阶为了三数组。

虽说降阶，但是实际上还是合适的。如果我们此时不再把这个四数组看作真正的四数组，而是只看 r8c236 三格，就会发现它实际上就是一个显性三数组，所以此时得到的删数是 r8c7(78)，此时直接利用唯一余数就能够出数了。

所以，总的来说，只要有显性数组，就必然在这个区域里存在与之互补的隐性数组；反之亦然。那么，我们来简单计算一下，为什么没有五数组，或更高规格的数组。

显然，五数组的存在就必然存在与之对应互补的另外一个数组，这个数组的规格必然小于或等于 4。所以，即使我们能够发现这样的数组，结构也存在与之互补的另外一个规格小于 5 的数组，所以，理论上是不存在五数组或更高规格的数组的。

3-4-2 为什么要提显隐性互补？

那么，为什么要提起显隐性互补呢？提起它的原因很简单：为了解决看不到某种数组的关系。因为显隐性互补的存在，我们在观察显性数组的时候，就可以使用隐性数组的互补视角来进行代换。隐性数组仅通过排除可以得到，所以我们更容易发现它；既然发现了隐性数组，也就间接地相当于发现了显性数组。所以，如果你对显性数组不熟悉，可以使用隐性数组来进行代换。这就是互补存在的意义。

3-5 提一下命名

最后说一下数组的命名格式和规范。最开始，数组被称为**链数**，比如三数组最开始被称为三链数。不过链数一词来源于中国台湾的谢道台老师，后面中国大陆也在推广数独，并使用适合大陆习惯的称呼数组一词。所以链数也就变为数组了，实际上它们是一个东西。而在英语里，数组一词有多个翻译：**subset**（子集）、**tuple**（元组）等。**subset** 来自于数学术语 **set**（集合），而 **tuple** 来自于计算机编程的数据结构元组：

```
(string name, int age, bool isGirl) = ("Sunnie", 24, false);
```

即使我们知道，数组在五阶及其以上的时候并不存在，但在数独里，它们依然是被命名的：

- 数对：pair;
- 三数组：triple;
- 四数组：quadruple;
- 五数组：quintuple;
- 六数组：sextuple;
- 七数组：septuple。

数组里没有八数组和九数组，因为互补的最大规格才等于 1 或 0，这在数组里显然是不可能也没必要出现的。

好了，具体内容就说到这里。数组我们就说完了。

Part 4 鱼/链列 (Fish)

我们进入一个新的技巧。这个技巧有一个比较好吃的名字：**鱼或链列 (Fish)**。

4-1 普通链列 (Basic Fish)

4-1-1 二链列 (X-Wing)

8	¹ 6	7	5	3	¹ 6	4	2	9
9	3	5	4	2	7	6	8	1
2	4	¹ 6	9	⁶ 8	¹ 6 8	3	7	5
4	8	3	6	5	2	9	1	7
6	7	2	1	9	3	8	5	4
¹ 5	¹ 5	9	8	7	4	2	3	6
¹ 5	2	¹ 6	3	4	⁵ 6 9	7	⁶ 9	8
3	⁶ 9	8	7	1	⁶ 9	5	4	2
7	⁵ 6 9	4	2	⁶ 8	⁵ 6 8 9	1	⁶ 9	3

如图所示，可以观察到，r18 两行里，能放下 6 的地方现在仅剩 4 个单元格，而且这 4 个单元格又恰好构成矩形的形状。那么不论如何，6 的摆放位置只可以是对角两格的填数情况是一样的。换句话说，要么左上角 r1c2 和右下角 r8c6 同时填 6，要么右上角 r1c6 和左下角 r8c2 同时填 6，没有其它情况。

注意，马上要得到的这一点很重要！虽说我们无法确定 6 的确切位置，但我们可以发现，6 在 c26 里的填数也必须只能在 r18c26 四个单元格里。因为刚才的说明已经得到了，两种情况里必须得有一个情况是对的，我们此时假设我们聚焦于 c2 来看，第一种情况下，c2 里会填入一个 6 在 r1c2 上；而第二种情况下 c2 里会填入一个 6 在 r8c2 上。这使得两种情况里，都会出现一个 6 在 r18c2 里；同理，聚焦于 c6 也是如此：第一种情况下填入到的地方是 r8c6；而第二种情况则填入到 r1c6。

虽说文字有些绕，我觉得你应该能明白我想表达的意思。那么既然这样的结构除了在 r18 里保证出现 6，还能保证 c26 里也各有一个 6 在这四个单元格里，那么很显然，c26 的其余位置都不能再填入 6 了，毕竟这两列的 6 已经只能放在 r18c26 里了。所以其余位置的 6 都可以被删除，于是有 r9c2, r379c6 <> 6。

这个技巧的巧妙之处在于，它的推理逻辑和之前我们说到过的一个技巧：组合区块很类似。在组合区块里，我们也是把区块当成了这样的形式，然后发现只能处于结构对角线的两格的填数情况才是一样的。这种结构实际上被称之为鱼或链列，而鱼和数组一样，也是有规格（阶）一说的，所以这个鱼的规格是涉及两行两列，因此被称为二阶，一般我们称这种结构叫做**二阶鱼或二链列 (X-Wing)**。

4-1-2 一些术语

在鱼的体系里，充满着各种诡异可怕的术语，但不要怕，我们这里提到一部分我们常用的，而复杂和难懂的术语，当后面用到的时候，我们才会提及。

- **定义域 (Defining Set/Base Set)**: 结构被定义下来的位置，比如这个例子里，定义域为 r18 (因为我们发现 r18 里填入 6 的位置只有四格，所以称这种东西叫做定义域)。
- **删除域 (或者叫删数域, Secondary Set/Cover Set)**: 能删除候选数情况的区域，比如这里的 c26 就是这个鱼的删除域。
- **鱼身 (Body)**: 结构涉及的所有单元格，比如前面这个示例下的 r18c26 就是鱼身。

目前就只需要提及这样几个简单的术语。

4-1-3 三链列/剑鱼 (Swordfish)

那么既然鱼也有规格，那么何不把它扩展到三阶的情况呢？我们来看看三阶的情况。

4-1-3-1 一个奇怪的例子

5	¹ ₇	⁴ ₇	6	^{1 3} ₄	^{1 3} ₄	2	9	8
3	2	^{4 6} _{7 9}	¹ _{4 5}	¹ _{4 5}	8	¹ _{4 6}	⁴ ₇	^{4 6}
¹ ₆	¹ ₉	8	7	^{1 2} ₄	^{1 2} ₄	¹ _{4 6}	5	3
2	8	³ _{5 6}	^{1 3} _{4 5}	^{1 3} _{4 5}	9	³ _{4 6}	³ _{4 7}	^{4 6}
9	4	³ ₇	³ ₇	^{2 3} ₇	^{2 3} ₆	8	1	5
¹ ₇	¹ ₆	⁵ ₇	³ _{5 6}	8	^{1 3} _{4 5}	^{1 3} _{4 6}	³ _{4 7}	² ₉
8	6	1	⁴ ₉	³ _{4 9}	7	5	⁴ ₉	2
⁷ ₉	⁵ _{7 9}	⁵ _{7 9}	2	⁴ _{8 9}	⁴ ₉	⁴ ₉	6	1
4	3	2	¹ ₉	6	5	⁹ ₉	8	7

如图所示，变成三阶后，这个例子实际上就比以前的结构看起来要复杂太多了：单元格涉及的数量就从 4 飞升到了 9，涨了一倍还多。没关系，别怕。

由于这个结构涉及恰好 $3 * 3 = 9$ 个单元格（乘号用“*”代替，除号用“/”代替，方便打字，后同），即三行三列，显然这说明了在鱼身里必须填入刚好 3 个 4，而且还不能违背数独规则（即同一个区域里有相同的数字）。那么，你随便放置这些 4，你都会发现，因为是三行三列的结构，所以为了保证行列宫之间不出现重复的冲突，数字只能错开分布，而且 4 的位置也必须得在 c567 的每一列里都恰好要出现一个 4 才行。这是显然的，因为只有三行的关系，每一行放一个 4 还不重复的话，对于列的角度而言，由于列也恰好是三个，所以必须是每一列都有一个 4。这便使得 c567 之中各有一个 4 出现在鱼身里。于是，删除域便形成了：c567，所以 c567 的其余单元格的候选数 4 都将被删除。

这个结构由于是三阶的，所以称为**三阶鱼**或**三链列**，而由于它的技巧英文名叫做

Swordfish，所以这个结构也总被称为**剑鱼**。不过。看起来似乎二链列的逻辑和三链列差别有点大，不过……你可以试试看，用现在这里的三链列的思维套用到二链列上，你就会发现，二链列实际上也可以用这个逻辑来理解，之前只是为了方便理解，直接写上了二链列它自己单独的视角罢了（其实这两种说法是等价的）。

4-1-3-2 奇怪的事情发生了

慢着。大哥，你是不是没有看到 **r9c7** 啊，这不是唯一余数吗（一个单元格就只有一个候选数的时候，就可以称这个单元格是唯一余数了，这和我们之前学到的唯一余数的逻辑是一样的）。很明显，出了这些数字后，岂不是 **4** 就显然不能这么放了，而是这样的：

5	¹ ₇	⁴ ₇	6	³ ₄	¹ ₄	2	9	8
3	2	⁴ ₆ ⁵ ₉	⁴ ₅	8	¹ ₇	⁴ ₆		
¹ ₆	¹ ₉	8	7	² ₄	¹ ₄	⁴ ₆	5	3
2	⁸ ₇	³ ₅ ⁶ ₄	¹ ₇	9	³ ₄	³ ₆ ⁴ ₇	³ ₄	⁶ ₄
9	4	⁶ ₇	3	² ₇	² ₆	8	1	5
¹ ₆	¹ ₅ ³ ₇	³ ₅ ⁶ ₇	8	⁴ ₅	⁴ ₆	³ ₄	² ₆	9
8	6	¹ ₄	³ ₉	7	⁵ ₄	³ ₄		² ₃
⁷ ₅	⁵ ₉	⁵ ₉	2	⁸ ₄	³ ₄	³ ₆	6	1
4	3	2	¹ ₆	⁶ ₅	⁵ ₉	9	⁸ ₇	7

是的。但我想告诉你，这不影响我们使用鱼技巧。换言之，这种**残缺**（Incompleted）的结构依然是可以使用鱼技巧的。这是为什么呢？这是因为我们在刚才三链列的逻辑的时候，并不关心内部是否有残缺，而仅仅是看到结构涉及三行三列后，为了保证每一行列都要出现一个数字 **4**，那么必然要让 **4** 放置在行列互不冲突的三个单元格上，而结构是否残缺并不会影响最终 **4** 要每一列都有一个的要求。所以，这种鱼依然是可行的，并且删除域、定义域依然不会发生变化，只是鱼身少了一格 **r8c3**。

那么，我们再来看两则示例（左图和右图），来好好体会和理解刚才我们说到的残缺。

1 4 5 6	1 5 6	9	1 4 5 8	2	1 8	1 3	5 6	7	6	7 5	9	1 3	7 5	4	8	2
1 5	8	7	1 5	6	3	9	2	4	5	2	8	5 9	4 7	4 3	1	6
1 4 5 6	1 5 6	2	1 4 5 7	1 7	9	1 3	5 6	8	1 3	1 3	4	2	6	8	5	9
9	1 3	5	1 7 6	1 7	3 1	8	4	2	4	3 5 6	5 6	8	1	5 6	2	7
1 3	4	6	1 8	1 8	3	2	5	7	8	7 9	2	5 9	4 7	4 3	1	3
2	7	8	9	5	4	6	3	1	1 5	1 7	9	3	4 7	2	8	6
7	2	1	3	9	5	4	8	6	9	4	1	6	5	3	7	2
5 6	5 6	4	1 8	1 6	8	7	2	9	2	5 6	5 6	7	8	1	9	4
6 8	9	3	2	4	8 6	7	1	5	3	8	7	4	2	9	6	5

4-1-4 四链列/水母 (Jellyfish)

三阶的结构大体的逻辑就可以理解成“三行的结构只出现在不同的三列里”，那么四阶就依葫芦画瓢：“四行的结构只出现在不同的四列里”。不过……这个例子是反过来了。

	3						2				2		1		2		1	2	3		
4	5	6	4		4	6	5	6		5	6	5	6	4		4		4			
7	8	9		9		8	9	7	9	7	8	9	8	9	8	9	7	7	8	9	
	3						2				2				2				2	3	
4	5	6	4		4	6	5	6	1		5	6	4		4		4				
7	8	9	7		9		8	9	7	9	8	9	8	9	7		9	7	8	9	
7	8	9		1	2		3			7	8	9	4		5	6					
	2						1	3	3	1	3	1		2			1	2	4		
4	6	4		4	6		4	5	6	4	5	6	5	6	4		4		4	6	
7		9	7		9		9	7	9	7	8	9	8	9		9		9		9	
	2		3	5			1			4	6	4	6		6		7	8			
4		6					4	6		4	6		6		9		4	6	9		
4		6		8	1		4	6		2			6		9		3	5			
7		9					7	9					9						9		
1	2			2			1	2		3	1	2	3				4		4	5	
4		8	9		9		4	5	6	4	5	6	5	6	4		8	9	7	8	9
1	2			5	7		1	2		4			1	2			6	3			
4							4			4			9		9				4	5	
4			6	3			8		4	5			7			2	1				

如图所示。这一次我们把 c2378 当成定义域。

发现，c2378 里能放置 9 的位置只出现在 r1247 这样不同的四行上。为了我们要在这个 c2378 里每一列都放上一个 9，还得不同行列（不产生冲突），我们发现，唯有把 9 放置在 r1247，每一列都有一个 9 的时候，才不会冲突。所以，r1247 就是这个结构的删除域了。不过这个例子厉害的地方在于，所有红色的数字都可以删除掉，而且除了 r2c5 有一个 1 外，它的删数基本上占满了整行，足足 19 个删数。

这个例子的规格是 4，所以叫**四阶鱼**或**四链列**，英文是 Jellyfish，所以也称**水母**。

4-1-5 普通鱼的互补

在普通鱼里，好像我们也没有说到五阶甚至更高规格的鱼，这一点是为什么呢？难道它

和数组一样？是的。普通鱼也有互补，而正是因为它有互补，所以才不存在五阶甚至更高规格和普通鱼。

如图所示，这是两个普通鱼的互补情况。

6	5	9	1	3	5	4	8	2	6	5	9	1	3	5	4	8	2
5	2	8	5	4	7	3	1	6	5	2	8	5	4	7	3	1	6
1	3	1	3	4	2	6	8	5	9	7	4	2	6	8	5	9	7
4	5	6	5	6	8	1	5	6	2	7	9	4	5	6	5	6	8
8	5	6	2	5	4	7	9	7	1	3	4	5	8	5	6	2	5
1	5	1	5	3	4	7	9	7	3	4	2	8	6	4	5	1	5
9	4	1	6	5	3	7	2	8	9	4	1	6	5	3	7	2	8
2	5	6	7	8	1	9	4	3	2	5	6	7	8	1	9	4	3
3	8	3	4	2	9	6	5	1	3	8	3	4	2	9	6	5	1

如图所示，这是两个互补的普通鱼结构，随便先看哪个例子都行。

比如左图，我们作为“基准”示例来看，它实际上是一个有残缺的三阶普通鱼结构，而删数也都是成立的（这一点我这里就不再说明了，因为这个例子在之前是有出现的，所以就请你自行推理了）。

不过，我们把剩下没有被涂色的单元格里（白色）的所有 5 都找到，然后按照正交（Orthogonal）的形式画出定义域。所谓的正交，就是说，如果原来是行，那与之正交的就是列；如果是列，与之正交的就是行。那么因为“基准”示例的定义域是行，所以我们正交的例子定义域就必须是若干个列。

此时，我们又得到了右图的这个普通鱼，虽然它也是残缺的，但是我们只转变了定义域和删除域的情况，但发现，其实删数却没有丝毫变化，这一点和我们之前数组的显隐性互补很类似，删数是一样的，所以我们才称这两个普通鱼互补。

正是因为这种形式的出现，所以我们才不可能拥有超过 4 阶的鱼结构：盘面整体是 9 阶的，如果某个数字在 n 个正交的行列里都没有出现，则说明这个数字一旦形成鱼结构，那么它的最高规格就不能超过 9 - n。而在这个数里能找到一个 m 阶的鱼，那与之互补的鱼的阶数必然是 9 - n - m，但因为 n 不能超过 4，所以 9 - n - m 这个数是永远都不可以超过 4 的。说起来很学术，实际上对照图上说就很清晰：如果把所有 5 都画出来，然后行列都标出来就会发现，它实际上只占据六行六列。既然普通鱼存在互补，那么两个鱼的规格之和就必须是 6，你不能因为互补，规格之和都超过 6 了，这显然不可能（毕竟 6 行 6 列只能最多填入 6 个这个数进去，规格之和超过 6 就意味着两种普通鱼内，填这个数的总量已经超过 6 个了，我们只能填 6 了，所以这就矛盾了）。

所以，普通鱼的规格也不可能超过 4。

4-1-6 说一下命名

还是老规矩，和数组一样，说一下普通鱼的名称。

因为普通鱼都有自己的名词，是用鱼来表示的，所以每一个阶的鱼都有自己的名字，如下：

- 二链列：X-Wing；
- 三链列：Swordfish（剑鱼）；
- 四链列：Jellyfish（水母）；
- 五链列：Squirmbag/Starfish（海星）；
- 六链列：Whale（鲸鱼）；
- 七链列：Leviathan（海怪，传说里的海中之妖怪）。

其中五链列的 Squirmbag 和 Starfish 是等价的两个说法，意思是一样的，都指的是五链列，用法没有丝毫区别。但是 Squirmbag 这个词语没有中文释义，甚至连直译都没有，不过这个词语是 Squirm（蠕动）和 bag（包）的自然拼接词，至于为什么，我稍后会作出说明。

和数组一样，它没有八链列和九链列，因为刚才的公式告诉我们， $9 - n - m$ 的 n 和 m 不论随便怎么取值，它们不可能存在（即使存在，也变为排除技巧了）。

你可能会很好奇两个地方。第一，为什么二链列有一个自己单独的名字，而不是一个鱼；第二，在之前的逻辑讨论和理论讨论里，为什么数组和普通鱼的关联性会如此大？

先来解决第一个问题。因为在最开始，鱼这个体系的建立是基于三链列的，而不是二链列，二链列在最开始也有自己的视角，所以当发现了鱼体系后，才知道二链列也属于这个体系，因此只是归到了鱼体系里，而没有单独为它取名；而另一方面，之所以这个体系要被称为鱼，是因为从三链列开始，这个结构的长相类似于美国的一种军用双翼飞机，叫 Fairey Swordfish。

现在再来说一下五链列的名称 Squirmbag。这个词语目前都没有比较合适的翻译，它起源于 Stringbag（网袋）一词。和三链列的来源 Fairey Swordfish 一样，Stringbag 是这架飞机的昵称，所以为了保留其“含义”，于是保留了 bag（包、袋子），把 string（串）换成了 squirm（蠕动）。

4-1-7 普通鱼的第二个视角：降维

下面我们来说一下刚才可能会困惑的第二个问题，为什么数组和普通鱼关联会如此之大。实际上，解释这一点只需要一句话：数组和普通鱼是同构的。所谓的同构，就是不同的体系里说法不一样，但代替的事物、核心和逻辑全部是相同的。不过说明这一点我们需要借助一个“工具”——降维。

我们试想一下，如果我们把一个盘面看作是一整个区域 X （即一行或一列），然后把盘面的一行或一列看作是这个区域 X 的一个单元格，然后把一行或一列的一个单元格看成 X 里一个单元格的一个候选数。说白了就是把盘面看成区域，区域看成格，格看成候选数。这么看有啥特殊么？是有的，我们来看看如下这个例子，我想表达什么。

1	6	2	5	4	3	2	7	2
2	7	8	6	2	1	4	3	5
4	3	5	8	2	7	6	2	1
7	2	1	4	5	8	1	6	9
6	4	4	9	1	2	3	5	7
5	5	1	3	7	6	1	2	4
2	1	6	2	3	5	2	4	2
3	4	2	8	5	2	7	1	6
2	7	7	1	6	4	5	2	3

如图所示。这个鱼的定义域是 c349，按列来看。可以发现，2 的位置只能出现于不同的三行里，所以这三行里必须得各有一个 2 才能放满 3 个 2。

这个时候，我们换一个降维视角来看：我们把这个盘面的九个列看成同一个区域的九个不同的单元格（比如这个例子里用到的是 c349，降维就相当于看作是某个区域里的第 3、4、9 个单元格）。然后我们针对于盘面的这几列，每一列都看看 2 的出现位置，c3 里 2 的位置只有第 1 格和第 8 格出现，降维就对应了刚才说到的“某个区域的第 3 格”里只有候选数 1 和 8；同理，c4 里 2 的位置只出现在 r78c4，所以降维后表达的意思是“某个区域的第 4 格”里候选数只有 7 和 8；而 c9 同理，降维后变为候选数 1 和 7。

而单纯观察候选数序列：{18}、{78}、{17}，显然它们是构成三数组的，因为一个区域下的这三个单元格却只包含三种不同的数字 1、7、8，所以内部的填数必须得是一个 1、一个 7 和一个 8。因此我们只需要关心 1、7、8 确实是确实都出现了即可。那么还原回去，因为三数组成立，所以原来的那个普通鱼的删数就完全成立了。按照三数组的逻辑，删数是这个区域里剩余位置的 1、7、8；那么倒回去看普通鱼，这个意思就是“删除第 1、7、8 行其余位置的 2”。

4-2 鱼鳍

之前我们说到了普通鱼结构，不过实际上我们并不可能经常出现类似于之前这样，刚好数字出现于几行几列的情况，有时候还是有点瑕疵的，所以我们就来探讨一下，如果鱼出现了瑕疵，又到底怎么用？

这些瑕疵在鱼里称为**鱼鳍**（简称**鳍**，**Fin**），表示长在鱼身上，又要影响推导结果的候选数。

4-2-1 鳍链列（Finned Fish）

我们循序渐进，看看一般的、带有鱼鳍的鱼到底有哪些用途。

4-2-1-1 鳍二链列 (Finned X-Wing)

先来看一个二阶的。

1	6	5 6	5 6 9	8	1 3	1 2 3	4	2 3	2
2	4	8	3	1	7	7	1	5	7 9
1	4	7	4	1	3	1 2 3	1	3	2
3	6	9	2	4	8	3	5	1	5 6
7	1	8	5	2	6	9	4	3	7
4	6	4 5 6	4 5 6	1	3	1	9	2	5 6
4	6	2	4	6	1	3	9	7	4 5
5	3	7	2	4	8	6	9	1	8
9	4	6	1	3	6	5	3	2 3	2

如图所示，我们发现，对于 **c67** 来说，能够填入 7 的位置，只有五个单元格：**r1c6** 和 **r24c67**。其中，我们当 **r1c6** 不存在，那么剩余结构是什么？那肯定是我们最为完美的二链列结构。但是 **r1c6** 的存在，使得现在不能完美地运用了。但是，我们可以这么去想：

- 当 **r1c6 = 7** 时：**r1c6** 的相关格都不可以填 7，因为但凡相关格的其一填入 7，都会和 **r1c6** 它们共同所在的区域下出现重复数字；
- 当 **r1c6 <> 7** 时：二链列结构形成，所以删数应为 **r24c1234589**。

只可能有上述两种情况的出现，但我们可以发现到的是，两种情况下，都可以删除掉 **r2c45(7)**，所以，**r2c45 <> 7**。

其中的 **r1c6(7)**，我们就称为鳍。

注意，英文名“鳍鱼 (Finned Fish)”之中的“鳍”含义为“带鳍的 (Finned)”，所以采用分词性形容词形式。

还有需要注意的地方是，当鱼鳍成立的时候，只需要讨论它能删除的位置，即它的相关格的数字，此时不关注整体鱼身发生了如何的变化。因为最终的删数是看删除域和相关格的交集。

那么，如何能快速观察和推断到删数情况呢？你可以发现，**r1c6** 的相关格组涉及三个区域：**r1**、**c6**、**b2**。**r1** 和 **c6** 下，都不会和它不存在所形成的二链列的删除的单元格有任何的交集，而只有 **b2** 才有交集。所以，鳍和剩余鱼结构的删数交集只能是宫内。

那么，鳍可以在一个技巧下存在几个呢？因为删除交集只能是宫内，所以最多也只能有 2 个。接下来我们来看一个有两个鱼鳍的例子。

5	^{2 3}	^{2 3}	6	4	8	7	1	9
1	⁴ ₈	⁴ ₈	9	² ₇	² ₇	5	3	6
6	9	7	3	5	1	4	8	2
9	5	¹ ₄	² ₇	^{2 3} ₇	6	¹ ₄ ³ ₇		8
3	7	6	1	8	4	9	2	5
2	⁴ ₈	¹ ₄ ⁸	5	9	³ ₇	6	⁴ ₇	¹ ₃
	^{2 3} _{7 8}	9	4	^{2 3} ₇	5	^{1 2 3} ₈	6	¹ ₃
	^{7 8}	1	² ₇	6	^{2 3} ₇	^{2 3} ₈	9	4
4	6	^{2 3}	8	1	9	^{2 3} ₈	5	7

如图所示，此时影响二链列成立的一共有两个候选数： $r89c7(3)$ ，所以我们灵活处理这两个候选数，我们将 $r89c7(3)$ 看作是一个区块。假设 $r89c7(3)$ 两处都不填，或者是 $r89c7(3)$ 里至少一处要填这样两种情况。

初学这里，我们来思考一下，为什么讨论的时候只分上述两种情况。如果 $r89c7$ 两处都不填 3，就好比这两个单元格“有 0 格填 3”；而与之相反的情况则只可能是“至少 1 格填 3”，所谓的“= 0”和“> 0”（此时我们不需要讨论“< 0”的情况，因为鱼鳍的总个数是完全不可能为负数值的）。

- **当 $r89c7$ 都不是 3：**此时所有影响二链列形成的位置都将不复存在，所以二链列此时成立，删数则是删除域上的 3（当然鱼身所占的 4 格是不能删除 3 的，这里说的是抛开这四个单元格的其它位置）；
- **当 $r89c7$ 里至少有一格填 3：**不论 $r8c7$ 还是 $r9c7$ 填 3，它们形成列区块形式，且它们同处于一个宫里，所以该宫里其余位置都不能填入 3。

整合了两大情况后，我们发现，此时不论哪种情况成立， $r7c9(3)$ 都是可以去掉的，所以 $r7c9(3)$ 便成为了这个题目的删数。

这个例子里，运用了两个鱼鳍，也就是这里的 $r89c7(3)$ ，不过这里，因为有两处鱼鳍，所以我们为了简化分类讨论的情况，我们将其合并归并为一种情况来讨论。

4-2-1-2 鳍三链列（Finned Swordfish）

接下来我们来看一则三阶的、带鱼鳍的普通鱼结构。

9	¹	¹	³	³	³	¹	¹	³	2
²	⁸	⁷ 8	⁷	⁴ 5 6	⁴ 5 6	⁴ 5 6	⁴	⁵ 8	¹
⁷	3	¹	⁶	9	⁴ 5 6	⁴ 5 6	⁴	¹	⁵ 6
²	⁷ 8	⁷ 8	² 3	⁶	1	8	9	³	⁶
⁷	5	4	⁷	⁶	¹	⁶	³	⁷	⁶
8	4	9	²	⁶	7	¹	²	3	5
1	7	3	²	²	⁴ 5 8	⁴ 5 8	⁴ 5	6	²
6	2	5	³	⁸	9	¹	³	7	⁴ 8
4	⁶	2	1	3	⁷	⁶	5	9	⁷ 8
5	9	¹	⁴	⁴	⁸	2	¹	6	3
3	¹	⁶	⁷ 8	⁵ 6	⁵ 6	9	2	¹	4
⁸	⁷ 8	⁷ 8	⁷ 8	⁷ 8	⁷ 8	⁷ 8	⁷ 8	⁷ 8	⁷ 8

如图所示。如果 $r1c6(7)$ 不存在，那么 $r189$ 作为定义域，数字 7 就会出现三链列结构，那么标准的删数就在 $c348$ 里产生了。不过，当 $r1c6(7)$ 的出现，使得我们不得不讨论 $r1c6 = 7$ 的情况。当 $r1c6 = 7$ 时，删除的数字就只有它的相关格里的 7 了。

合并两个情况，我们发现，它们都能删除掉的只有 $r3c4(7)$ ，所以 $r3c4 <> 7$ 。

我们再来看一则示例，这则示例可不太好理解。

6	³	⁴	³	⁴	³	2	1	9	⁵	⁴	³
⁷	⁵	⁷	⁸	⁴	³	⁷	⁵	⁶	⁷	⁸	⁹
⁵	1	2	³	⁴	⁶	8	⁴ 5 6	⁵ 6	⁴ 5 6	³	⁷
⁵	³	⁴	³	⁴	⁶	⁵	⁴ 5 6	¹	²	⁷	⁸
⁵	9	7	2	6	4	1	3	⁵	⁸	⁷	⁹
2	4	³	⁶	1	⁵	⁵	⁵ 6	⁵ 6	⁵ 6	⁸	⁹
1	³	⁶	⁷	⁵	⁵	⁴ 5 6	2	⁴ 5 6	⁸	⁹	⁷
7	2	9	5	3	6	8	4	1	⁵	⁶	⁹
3	⁶	1	⁸ 9	4	7	2	⁵	⁶	⁵	⁶	⁹
4	⁶	5	⁸ 9	1	2	3	⁶	⁹	⁷	⁸	⁹

如图所示，这个例子有些奇怪。如果 $r1c2 <> 5$ ，你就会发现，这个例子和之前的例子都有所不同：鱼身处于 $c1$ 里只有 $r4c1$ 一格了。按照普通鱼的讨论逻辑，为了保证要放 3 个 5， $r4c1$ 此时必须填 5（否则的话，结构整体就只能放到 $r148c89$ 这个矩形区域里，而这个 $3 * 2$ 的区域里是怎么都放不下三个 5 的。所以必须得让 $r4c1 = 5$ （当然，前提是 $r1c2 <> 5$ ，即鱼鳍不成立的时候，才有的结论）。

那么我们此时就不必再去关注其余 5 的摆放位置了。因为我们讨论的两个核心情况已

经出来了：鱼鳍不成立的时候， $r4c1 = 5$ ；鱼鳍成立的时候， $r1c2 = 5$ 。所以完全只需要去看 $r1c2$ 和 $r4c1$ 都能删除的地方，按照普通鱼和鱼鳍相关格的交集这一个层面来讨论的话，删除的仅有 $r23c1(5)$ 。

好了，实际上这个题的删数并不止这些。而且我们从逻辑里就已经看出了端倪，只是在段落的描述里没有点破这一点。那么还有哪里可以删除呢？试试找一下吧。

另外，可以从描述里发现，如果鱼鳍不存在的话，这个结构实际上就降阶变为二链列了（此时只需要看 $r18$ ），所以这一点跟之前显隐性互补的其中一则示例是一样的，虽然是四数组，但我们可以降阶变为三数组。不过，这则示例之所以客观存在，还是因为这里的鱼鳍是客观存在的缘故，它使得结构存在而不会被降阶，毕竟鱼鳍本身也单独算作其中一种分析的情况来讨论。

4-2-1-3 鳍四链列 (Finned Jellyfish)

2 5	1 2 6	9	3	1 2 5	1 5 6	4	8	7
4	7	3	5 9	8	5 6 9	5 6	1 2	1 2
2 5 8	1 2 6	1 2 8	1 2 5	4	7	5 6	3	9
2 8	9	6	7	3	1 8	1 2	5	4
7	3	4 2	1 4 5	9	1 5	1 2	6	8
1	5	4 8	4 8	6	2	7	9	3
6	1 2	5	1 2 8 9	7	1 8 9	3	4	1 2
9	4	1 2	6	1 2	3	8	7	5
3	8	7	1 2 5	1 2 5	4	9	1 2	6

如图所示。这一个示例也是非常难以理解的，因为它残缺太多了，还带有一个鱼鳍。

我们先忽略 $r1c2(2)$ ，看看在 $r1458$ 上会发生什么神奇的事情。当我们忽略 $r1c2(2)$ 时， $r1458$ 所有的 2 构成了四链列结构，因为所有 2 均出现在 $r1458c1357$ 这个矩形里，没有超出这个矩形，且这个矩形恰好是四行四列的，所以为了保证每一行都放下一个 2，而且行列互不影响，所以必须在 $c1357$ 上各放下一个 2，才够放下 4 个 2，于是四链列成立。

当然， $r1c2(2)$ 是客观存在的，所以假设当 $r1c2 = 2$ 的时候，只能删除掉删除域里 $r23c13(2)$ 。所以，这便产生了删数 $r23c13(2)$ 。

4-2-1-4 普通鳍鱼的互补

有没有发现，在之前的示例里，依然没有出现规格大于 4 的、带鱼鳍的普通鱼结构。难道鳍鱼也拥有互补的情况？答案是，是的。不过，这一点的证明我们无法从现有的知识得到，所以证明就不给出了，不过我们可以给出一个示例，让你能明白这一点。

6	^{1 3}	2	^{1 7}	^{3 7}	9	^{5 8}	4	^{5 8}	6	^{1 3}	2	^{1 7}	^{3 7}	9	^{5 8}	4	^{5 8}
8	^{1 3}	5	^{1 4}	^{2 3 4}	6	^{1 2}	9	7	8	^{1 2 3}	5	^{1 4}	^{2 3 4}	6	^{1 2}	9	7
9	4	7	8	5	^{1 2}	6	^{1 2 3}	^{1 3}	9	4	7	8	5	^{1 2}	6	^{1 2 3}	^{1 3}
^{1 2}	9	4	3	8	5	7	^{1 2 6}	^{1 6}	^{1 2}	9	4	3	8	5	7	^{1 2 6}	^{1 6}
3	^{5 7}	8	6	^{2 4 7 9}	^{1 2}	^{1 2 5 9}	^{1 2 3 5 8 9}	^{1 3}	3	^{5 7}	8	6	^{2 4 7 9}	^{1 2}	^{1 2 5 9}	^{1 2 3 5 8 9}	^{1 3}
^{1 2}	^{5 7}	6	^{1 4 7 9}	^{2 4 7 9}	^{1 2}	^{1 2 3 5 8 9}	^{1 2 3 5 8 9}	^{1 3}	^{1 2}	^{5 7}	6	^{1 4 7 9}	^{2 4 7 9}	^{1 2}	^{1 2 3 5 8 9}	^{1 2 3 5 8 9}	^{1 3}
5	6	1	9	^{4 7 8}	^{3 8}	^{3 8}	2	5	6	1	9	^{4 7 8}	^{3 8}	^{3 8}	2	5	6
4	2	3	5	1	^{7 8}	^{8 9}	^{7 8 9}	^{6 8 9}	4	2	3	5	1	^{7 8}	^{8 9}	^{7 8 9}	^{6 8 9}
7	8	9	2	6	3	4	^{1 5}	^{1 5}	7	8	9	2	6	3	4	^{1 5}	^{1 5}

如左图所示，可以看到它是一个很普通的三阶鳍鱼结构，不过，我们依然按照之前的方式，将关注点变为白色的单元格。只是，在转变视角后，选取的定义域要从行变为列，而且鱼鳍的位置依然不能发生任何改变。

确实，我们从白色的单元格里发现到了一处鳍鱼结构，此时鳍鱼是四阶的。不过需要你注意到的地方是，此时删除域 **r4** 上，只有一处 **1** 可以填上去。推理过程里要保证每一个删除域部分都得填入一个 **1** 才合适，所以此时 **r4** 的 **1** 仅能放置到 **r4c1** 上。这一点和之前讲到的三阶鳍鱼的后面这一则示例很相似。不过，删除的数字依然没有发生变动。

4-2-2 退化链列（Sashimi Fish）

之前我们介绍的例子可能已经足够让人崩溃了，不过接下来这一节，理解起来可能更加困难，所以我尽量把细节都讲明白。

4-2-2-1 退化二链列（Sashimi X-Wing）

¹	^{5 6}	^{5 6 9}	4	¹	^{5 6 9}	¹	^{5 6 9}	3	8	2	7
¹	^{5 6}	3	7	2	8	¹	^{6 4}	^{4 6 9}	¹	^{6 4}	^{4 6 9}
¹	^{6 8}	2	^{1 8 9}	4	^{1 6 7 9}	^{1 3 6}	^{3 6 9}	5	¹	^{3 6 9}	^{3 6 9}
9	4	^{2 5}	^{1 5 6}	^{1 3 5 6 7}	^{2 5 6 7}	^{1 3 5 7}	^{1 3 5 7}	^{1 3 5 7}	^{1 3 5 7}	^{1 3 5 7}	8
^{2 5 7 8}	1	6	^{5 8 7}	^{5 8 7}	^{2 5 7}	^{1 3 5 7}	^{1 3 5 7}	4	9	^{5 3}	^{5 3}
3	^{5 7}	^{5 8}	^{1 5 8 9}	^{1 5 8 9}	4	^{1 5 8 9}	^{1 5 8 9}	6	2	^{1 5 8 9}	^{1 5 8 9}
^{2 5 7}	^{5 6 9}	8	^{2 5 9}	^{5 6 9}	4	1	^{5 6 7}	3	^{5 6}	^{5 6}	^{5 6}
^{1 5 7}	^{5 6 9}	^{5 6 9}	^{1 5}	3	2	8	9	^{4 7}	^{4 5 6}	^{4 5 6}	^{4 5 6}
4	^{5 6 9}	3	7	^{5 6 9}	^{5 6 9}	^{5 6 9}	2	8	1	^{5 6 9}	^{5 6 9}

如图所示，我们发现，**r6c8** 按照最初的结构，应该是含有候选数 **7** 的，不过现在它被提示数占据了，结构缺了一个角。那它还能推理吗？

按照最初的分析方式和模式，我们发现，c8 存在完全不同于结构行列的额外的位置：r4c8。如果补齐 r6c8，r4c8 则应该为鳍。于是按照鱼的分析方式和思维来观察这样的结构：

- 如果 r4c8 = 7，则可以删除鱼本拥有的删除域和 b6 的交集单元格（r6c79）。
- 如果 r4c8 <> 7，缺了一个角的二链列成立，但……

关键就在这里，二链列如果缺了角之后，依然能使用吗？

r4c8 <> 7 时，按照底层的分析逻辑，依然分成两种情况来考虑。因为是缺了一个角的二链列，此时按照之前二链列的分析方式，只能有两种填数情况：要么 r6c2 和 r8c8 都填 7，要么 r8c2 单独填 7。单独填 7 意味着它只可能排除掉原本应有的删除域的其中一行（r8），而 r6 它是无能为力的。但是，你会发现一点神奇的现象：当 r4c8 <> 7 时，r8c8 = 7，因为此时 c8 只有两处可填 7 的地方。这也就是说，当 r4c8 <> 7 时，直接自动就进入了 r6c2 和 r8c8 都填 7 的情况（r6c2 填 7 你可以顺次进行推导得到，这是一定的，因为这样的推导过程，和之前标准鱼的假设推导过程类似）。这也就是说，r4c8 不填 7 时，二链列的删数所在区域一定是两行（r68），而不可能只有一行（r8）。

所以，r4c8 不管是不是 7，都能删除掉 r6c79(7)，所以 r6c79 <> 7。

这个技巧理解起来有一丝难度，不过我们可以得到一点：鳍的存在可以使得原本应完美的结构再次缺失一个角。这个缺失的位置必须处于鳍所在的宫。比如这里缺失的是 r6c8(7)，和鳍 r4c8(7)同宫。

结构的特殊性就在于，虽然在推导过程之中，显然 r8c2 是可以单独填 7 的，但是当鳍不成立时，不可能使得它单独成立。

那么，你可能会问了，什么时候可以使得 r8c2 单独填 7？或者换种问法：r8c2(7)在我们上文都没有怎么提到，而我们讨论的情况好像都没有 r8c2(7)的戏份，那么这个候选数是否可以被删除？这个问题问得好，但是我们得有一个思维。我们好像在鳍不成立的时候才考虑到它的，虽说鳍成立这一点跟这个点毫无关系（我们之前就说过，鱼鳍的真假（真假指的是填和不填的两种情况）都跟鱼身里的填数没有任何关系，毕竟最终带有鱼鳍的删数仅仅是看鱼鳍的相关格和删除域的交集。不过此时，因为我们无从找到情况可以讨论，所以只得发现，鱼鳍成立的时候，恰好这个单独的情况就成立了。因为这一点跟鱼鳍有关系，也就是说 r8c2(7)是不可以随意删除的。

最后，如果不看鳍 r4c8(7)，剩余的鱼结构称为**退化鱼或退化链列（Sashimi Fish）**，如果没有鳍的支撑，退化鱼本身就可以直接产生出数效果；不过这个技巧依赖于这个鱼身结构，所以这个技巧索性也就叫退化鱼了，所以退化鱼既可以指代这个鱼身，也可以指代这个逻辑。

对于术语“Sashimi”，有些人认为，退化鱼结构因为结构特殊，必须依靠鳍生存，所以名字里必须添加上 Finned 一词，即 Finned Sashimi Fish；另一些人认为，退化鱼结构本身是一种特别的结构，使用 Sashimi 直接来替代这样的特殊结构就可以了，所以这种观点的人群认为不需要添加 Finned 一词。所以它的英文名可以叫 Finned Sashimi Fish，也可以叫 Sashimi Fish。我个人习惯是，当不产生歧义时，就不用加 Finned 一词。

看吧，就这个例子我讲了都一页多。所以为了你能好好理解，我们再给出一则示例。

1 5 6	5 6 9	4	1 5 6 9	1 5 6 9	3	8	2	7
1 5 6	3	7	2	8	5 6 9	1 6 4	4 6 9	
1 6 8	2	1 8 9	4	1 6 7 9	6 9	1 3 6	5	3 6 9
9	4	2 5	1 5 6 7	1 3 5 6 7	2 5 6 7	1 3 5		8
2 5 7 8	1	6	5 8	5 7	2 5 7	4	9	5 3
3	5 7	5 8	1 5 8 9	1 5 7 9	4	1 5 7	6	2
2 5 6 7	8	2 5 9	5 6 9	4	1	5 6 7	3	5 6
1 5 6 7	5 6 7	1 5	3	2	8	9	4 7	4 5 6
4	5 6 9	3	7	5 6 9	5 6 9	2	8	1

如图所示，和刚才的示例一样，我们把 **r1c6(9)** 补上去后就发现结构完整了。如果我们讨论和分析鱼鳍的时候，因为这里是两个鱼鳍，所以我们得和之前分析普通鳍鱼的时候的分析方式一致，即把两个鱼鳍并在一起，看作一个区块来看，讨论这个区块的填数是两个鳍有一个成立，还是都不成立。

当都不成立的时候，就转化成为了刚才我们提到的例子那样，二链列即使缺了一个角，但逻辑依然是成立的；但因为客观存在的缘故，导致最终的删数产生于删除域和鱼鳍形成的区块的交集，故最终的删数是 **r1c45(9)**。

4-2-2-2 退化三链列（Sashimi Swordfish）

下面我们来看一则三阶的示例。

2 3 4 5	8	2 3 4 5	6	7	1	2 4 5	2 5	9
2 5 6 7	2 5 6 7	2 5	3	9	4	2 5 6 8	1	2 6 8
9	4 6	1	2	8	5	4 6	3	7
2 4 5	1	6	4 5	3	7	2 5 8	9	2 8
2 4 5	9	8	1	4 5	6	7	2 5 9	3
5 7	3	5 7	9	2	8	1	6	4
1	4 6 9		8	4 6	2	3 6 9	7	5
2 3 4 5 6	7	2 3 4 5	4 5	1	9	2 3 6	8	2 6
8	2 5 6 9	2 5 9	7	5 6 9	3	2 6 9	4	1

如图所示，这一则示例里，我们必须补全的是 **r1c2(5)**，当补好 **r1c2(5)** 后，你就会发现，我们利用降维的视角去分析，降维后恰好是 {19}、{59} 和 {15}，构成了三数组，是

成立的，所以这个鱼在忽略鱼鳍的情况下也是成立的。下面我们就来探讨一下，这个退化鱼结构到底又是怎么成立的。

首先，我们要保证在结构里的这三列定义域，每一列都要有一个 5，那么它们互不相同行列的话，此时只可能产生一种情况（因为题目的残缺情况太严重了，所以只有一种情况），即 $r9c2 = r5c5 = r1c8 = 5$ ，不过此时的删除域依然能得到保证。因为这三个不同行列的填法最终使得了它们所在的三行 $r159$ 都有一个 5，所以当鱼鳍不成立的时候，这种结构的删除域整体确实是保证能够删除掉的。然而这看起来好像跟 $\{r9c5, r5c8\}(5)$ 没有任何的关系，但是实际上……鱼鳍成立是可以保证它们能放进去的情况。当鱼鳍成立的时候， $c1$ 就已经存在 5 了，所以我们只需要考虑 $c59$ 上各有一个 5。那么此时你既可以把 $c5$ 的 5 放到 $r5c5$ 上，也可以放到 $r9c5$ 上；同理 $c9$ 也是一样，所以此时鱼鳍成立时， $r9c5$ 和 $r5c8$ 依旧是不可能填入 5 的。不过我们研究的是删数，而删除域已经可以保证能出现了，所以鱼鳍的相关格和删除域的交集即为该技巧的删数，即 $r1c13(5)$ 。

4-2-2-3 退化四链列（Sashimi Jellyfish）

最后来享用一个四链列吧！

5 7 8 9	5 7 8 9	3	4	1	6	2	5 7 8 9	5 7 8 9
2	6	8 9	7 9	5 8 9	3	1	5 7 8 9	4
1	5 7 8 9	4	2 7 9	5 8 9	2 5 8 9	7 8 9	3	6
8 9	4	6	3	7	1	5	8 9	2
5 9	2	1	8	4	5 9	6 7 9	6 7 9	3 7 9
5 8 9	5 8 9	7	6	2	5 9	4	1	8 9
6 7 8 9	7 8 9	5	2 9	3	2 8 9	6 7 8 9	4	1
6 7 8 9	7 8 9	2	1	8 9	4	6 7 8 9	5 6 7 8 9	5 7 8 9
4	1	8 9	5	6	7	3	2	8 9

如图所示，我们如果把 $r9c1(8)$ 补齐（或者补 $r9c2(8)$ 也行，甚至 $r9c12(8)$ 都补上都行），忽略鱼鳍的存在，鱼就不是退化的了，按降维的方式来看，四行看作四格，那么对应的候选数情况就是 $\{1289\}$ 、 $\{18\}$ 、 $\{129\}$ 和 $\{129\}$ ，四数组是成立的，所以鱼结构是成立的。

不过此时需要讨论的情况就太多了。假设鱼鳍不存在，我们要保证 $r1469$ 每一行都有一个 8，首先 $r9$ 就只能放在 $r9c9$ 上，剩下的三行就只剩下 $\{r1c128, r4c18, r6c12\}$ 这么一些单元格可以填入 8 了，可以发现，此时这个形式已经是一个合格的三链列结构，所以三链列的删除域就一定被保证了： $c128$ ，而再算上刚才不得不放在 $r9c9$ 的这一个 8，它对应了 $c9$ ，所以删除域必须是 $c1289$ ，确实是成立的；

然后再来看 $r9c3(8)$ 这个鱼鳍。鱼鳍客观成立的时候，只能保证删除鱼鳍和删除域的交集，所以这个技巧能删的也就只能是这个交集的候选数，而这个交集只有 $r78c12(8)$ ，所以这就是这个技巧的删数。

讲了好半天，希望你能明白我想表达的意思。

4-2-3 孪生链列 (Siamese Fish)

在最开始的普通鱼里，我们讲到过一个示例，而留下了一个问题，说的是段落已经暗示了还有额外删数，只是没有点明删哪里。那么，这个删数的原理和逻辑究竟有着什么样子的特殊之处呢？这便产生了这一节的内容：**孪生鱼**或**孪生链列** (Siamese Fish)。

4-2-3-1 孪生二链列 (Siamese X-Wing)

	1	6	5 ³	5	3	4	8	2
4	5		2	6		7	1	3
2		3		1	4	6	5	9
3	2	7	4				6	1
	6		1	3	2		7	4
1	4			7	6	3	2	8
5	3	4					9	6
				4	5		3	7
						8	4	5

如图所示。仔细观察，这则示例，它不就是之前我们说的退化鱼么？那么删数岂不是都能用退化鱼理解了？是的，我们来看它的两种情况：

	1	6	5 ³	5	3	4	8	2
4	5		2	6		7	1	3
2		3		1	4	6	5	9
3	2	7	4				6	1
	6		1	3	2		7	4
1	4			7	6	3	2	8
5	3	4					9	6
				4	5		3	7
						8	4	5

	1	6	5 ³	5	3	4	8	2
4	5		2	6		7	1	3
2		3		1	4	6	5	9
3	2	7	4				6	1
	6		1	3	2		7	4
1	4			7	6	3	2	8
5	3	4					9	6
				4	5		3	7
						8	4	5

两个情况，每一个情况都对应了一个删数的结论，而且还都是对的。所以我们把结果并在了一起，所以这便是孪生鱼的核心——鱼身大部分不动，鱼鳍的位置变动便产生不同的删数。遂这一个例子就讲完了。是不是有点太简单了？并不，我们来看三阶的情况。

4-2-3-2 孪生三链列 (Siamese Swordfish)

2 4 5 6 8 9	4 5 6 8	4 8 9	1	2 3 6	7	2 6 4 8 9	2 3 6 4 8 9	2 3 4 6
2 4 6 8	1 4 6 7 8	1 2 4 6 7 8	9	2 3 6	5	1 2 6 4 7 8	2 3 6 4 8 9	1 2 3 4 6
2 6 9	1 7	3	4 6 8	2 4	5	2 6 9	1 7	3
2 3 4 6	9	4	2 4 5 8	3 1 5	1 2 3 4 8	1 2 6	7	1 2 3 5 6
3 8	3 7 8	5	2 7 9	1 7 9	6	4	3 9	1 3 3
1	4 3 6 7	2 4	3 5 7	2 5 9	3 4	2 6 9	2 3 5 6 9	8
7	1 3 4 5 8	1 4 8	3 1 5 8	1 2 3 5 6	2 8	2 6 8	2 4 5 6 8	9
5 8 9	5 8	6	5 7 8	4 7 8	1 2 8 9	3	2 5 8 7	2 5
4 5 8 9	3 2	4	5 3 6 7 8	5 6 7 9	3	6	1	4 5 6 7

如图所示，你是否能一眼辨别出这个结构里两处删数到底都是怎么形成的吗？

2 4 5 6 8 9	4 5 6 8	4 8 9	1	2 3 6	7	2 6 4 8 9	2 3 6 4 8 9	2 3 4 6
2 4 6 8	1 4 6 7 8	1 2 4 6 7 8	9	2 3 6	5	1 2 6 4 7 8	2 3 6 4 8 9	1 2 3 4 6
2 6 9	1 7	3	4 6 8	2 4	5	2 6 9	1 7	3
2 3 4 6	9	4	2 4 5 8	3 1 5	1 2 3 4 8	1 2 6	7	1 2 3 5 6
3 8	3 7 8	5	2 7 9	1 7 9	6	4	3 9	1 3 3
1	4 3 6 7	2 4	3 5 7	2 5 9	3 4	2 6 9	2 3 5 6 9	8
7	1 3 4 5 8	1 4 8	3 1 5 8	1 2 3 5 6	2 8	2 6 8	2 4 5 6 8	9
5 8 9	5 8	6	5 7 8	4 7 8	1 2 8 9	3	2 5 8 7	2 5
4 5 8 9	3 2	4	5 3 6 7 8	5 6 7 9	3	6	1	4 5 6 7

拆解开来，结构变成了这样两个不同的退化鱼。那么你可能会问了，是不是孪生鱼最终的拆解必须都得是退化鱼呢？其实不是。来看这一则示例。

6	<div>5</div>	<div>3</div>	<div>3</div>	<div>3</div>	2	1	9	<div>5</div>	<div>4</div>	<div>3</div>
<div>5</div>	1	2	<div>3</div>	<div>3</div>	<div>3</div>	8	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>
<div>5</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>
<div>5</div>	9	7	2	6	4	1	3	<div>5</div>	<div>5</div>	<div>3</div>
2	4	<div>3</div>	1	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>
1	<div>5</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>
7	2	9	5	3	6	8	4	1	<div>3</div>	<div>3</div>
3	<div>3</div>	1	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>
4	<div>3</div>	5	<div>3</div>	1	2	3	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>

如图所示，它能被拆解为下面这样：

6	<div>5</div>	<div>3</div>	<div>3</div>	<div>3</div>	2	1	9	<div>5</div>	<div>4</div>	<div>3</div>
<div>5</div>	1	2	<div>3</div>	<div>3</div>	<div>3</div>	8	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>
<div>5</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>
<div>5</div>	9	7	2	6	4	1	3	<div>5</div>	<div>5</div>	<div>3</div>
2	4	<div>3</div>	1	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>
1	<div>5</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>
7	2	9	5	3	6	8	4	1	<div>3</div>	<div>3</div>
3	<div>3</div>	1	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>
4	<div>3</div>	5	<div>3</div>	1	2	3	<div>3</div>	<div>3</div>	<div>3</div>	<div>3</div>

从左图里可以看出，它看起来好像是退化鱼一般，但需要注意的是，它实际上就是我们之前讲到的那一则示例（见三链列一节的比较奇特的示例），它实际上是一个普通的三阶鳍鱼；而右侧这则示例里，它就是实打实的退化三链列了，因为鱼鳍长在了定义域里只有一格可以填的行上。所以这个整体的孪生鱼，是由一个普通鳍鱼和一个退化鱼并在一起的。

4-2-3-3 孪生四链列（Siamese Jellyfish）

那么，我们来看看这则示例。

5	3	4	2	5	3	1	6	4	2	8	7	2
2	6	1	6	8	7	5	4	2	4	3	9	3
8	4	2	7	3	4	6	5	1	9	8	9	1
4	5	4	6	2	1	7	3	8	9	8	9	4
2	6	1	7	5	4	1	7	6	8	9	4	2
7	3	5	4	1	3	2	3	2	6	5	6	8
1	5	7	5	4	1	3	2	3	2	6	5	6
4	8	4	2	5	6	9	1	7	8	9	5	8
1	5	6	2	4	1	3	7	4	3	5	8	8

如图所示。我们给出拆解的两种情况作为参考：

5	3	4	2	5	3	1	6	4	2	8	7	2
2	6	1	6	8	7	5	4	2	4	3	9	3
8	4	2	7	3	4	6	5	1	9	8	9	1
4	5	4	6	2	1	7	3	8	9	8	9	4
2	6	1	7	5	4	1	7	6	8	9	4	2
7	3	5	4	1	3	2	3	2	6	5	6	8
1	5	7	5	4	1	3	2	3	2	6	5	6
4	8	4	2	5	6	9	1	7	8	9	5	8
1	5	6	2	4	1	3	7	4	3	5	8	8

左图稍微麻烦一些，是一个退化四链列，因为鱼鳍处于 r2 上，当鱼鳍不成立的时候，r2 的数字 9 就只能填入到 r2c3 里，使得结构立马降阶；而右图给出的示例则是普通的四阶鳍鱼。示例的删数和逻辑就不多去讲解了。下面我们再来看一则示例。

5	3	6	8	9	1	² ₄	7	² ₄
9	4	8	2	6	7	1	3	5
2	1	7	4	5	3	8	6	9
3	⁶ ₈	2	7	4	⁶ ₈	9	5	1
¹ ₈	^{5 6} _{8 9}	4	^{1 3} _{5 6 9}	³ ₈	^{5 6} _{8 9}	^{2 3} ₆	² ₈	7
7	^{5 6} _{8 9}	¹ ₉	^{1 3} _{5 6 9}	2	^{5 6} _{8 9}	^{4 6} ₈	⁴ ₆	⁶ ₈
⁴ ₈	7	³ ₉	^{3 6} ₉	1	^{4 6} _{8 9}	5	^{4 6} _{8 9}	⁶ ₈
¹ ₄	2	5	⁶ ₉	7	^{4 6} _{8 9}	^{4 6} ₈	¹ ₄	3
6	^{1 3} _{8 9}	⁵ ₉	³ ₈	³ _{4 5 8 9}	² _{8 9}	7	^{1 2} _{4 8 9}	² ₈

如图所示，这个例子一共有 4 个鱼鳍，可以发现这个例子就是比较难的例子了。我们依然按照拆解的模式把例子进行拆解：

5	3	6	8	9	1	² ₄	7	² ₄
9	4	8	2	6	7	1	3	5
2	1	7	4	5	3	8	6	9
3	⁶ ₈	2	7	4	⁶ ₈	9	5	1
¹ ₈	^{5 6} _{8 9}	4	^{1 3} _{5 6 9}	³ ₈	^{5 6} _{8 9}	^{2 3} ₆	² ₈	7
7	^{5 6} _{8 9}	¹ ₉	^{1 3} _{5 6 9}	2	^{5 6} _{8 9}	^{4 6} ₈	⁴ ₆	⁶ ₈
⁴ ₈	7	³ ₉	^{3 6} ₉	1	^{4 6} _{8 9}	5	^{4 6} _{8 9}	⁶ ₈
¹ ₄	2	5	⁶ ₉	7	^{4 6} _{8 9}	^{4 6} ₈	¹ ₄	3
6	^{1 3} _{8 9}	⁵ ₉	³ ₈	³ _{4 5 8 9}	² _{8 9}	7	^{1 2} _{4 8 9}	² ₈

鱼的内容我们就讲到这里，鱼的体系其实远远不止这一点内容，不过后面的部分难度都相当大，所以我们放在后面才会讲到。

4-2-4 说一下命名

可以看到，给的例子多，称呼它们都不容易，所以我们总结一个叫法。其实也很简单，我们只有三种称呼：标准、鳍、退化和孪生，在这三个词语后加上“n 链列”就可以了。比如鳍三链列（Finned Swordfish）、退化四链列（Sashimi Jellyfish）。如果要说明鳍的个数，直接把鱼鳍个数放在最前面，然后加个“鳍”字。比如三鳍三链列等。

不过，我们习惯把一个鳍的结构称为“单鳍”，两个鱼鳍称为“双鳍”，比如单鳍二链列等等。而且实际上，鱼的命名非常麻烦，所以这一点我们不作出详细说明，后面会提到。

4-3 怎么观察普通链列和鳍链列？

很多人对这个技巧有所畏惧，因为它的结构实在是太庞大了，它不像是数组，哪怕结构较大，起码也在同区域下，现在看不到一会儿也能看到；不过这则结构不同，它涉及多个不同的行列，还要推理，所以看起来似乎非常困难，为了解决链列不方便观察的弊端，一位叫做辉煌背后的衰落的数独玩家发明了一种观察角度，它将鱼结构的观察简化了非常多，但证明过程比较困难，所以这种观察角度的完备性证明过程我们就不在这里说了。

在此之前，我们来看一则示例。如果你能在大约 1 分钟之内观察到这个图里给出的链列（鱼），就说明你已经掌握得比较熟练了。那么我们来看看，我们要讲的观察角度。

4-3-1 普通链列的观察

7	6	4 5	3	1	2	5 8	3	9	4	3	4 5	3
2 3	2	5	3	5	7 8	9	4 5	6	4 5	7 8	3	3
5	3	8	9	4	6	7	5	2	1	5	7	3
2	4	8	1	6	9	4	7 8	5	4	8	2 3	3
9	2	5	7	2 3	5	4 5	8	6	4	8	2 3	1
2	4	5	8	3	4 5	8	6	1	4 5	8	7	9
1	5	3	7	2	5	8	4 5	6	1	4 5	9	3
6	4	5	3	7	9	1	8	2 3	5	2 3	5	3
1	5	8	9	5	8	2	5	3	4 5	7	6	6

如图所示，由于链列只涉及同一个数字，所以我们的观察方法就是通过 1 到 9 逐个查找的方式来搜索是否这个数有普通链列和鳍链列结构。

为了解释我的逻辑，我们从 1 开始。首先我们把盘面里所有的 1 都圈出来，一共是 7 个。然后我们要找出一个矩形区域，这个矩形区域是任意的，随便你怎么选其中的单元格位置，只要它是一个矩形，而且我们的矩形涉及的所有单元格以及所处行列下都没有确定值 1 即可，且还要求矩形的所有单元格里不能有候选数 1。不过，我们需要把矩形找到后，把矩形的信息代入这个公式：

$$\text{NAIN 值} = 9 - (\text{矩形行数} + \text{矩形列数})$$

如果这个所谓的 NAIN 值等于我们要找的这个数字出现的总个数的时候（此处指的是数字 1 的总个数），我们就可以称为该数存在鱼结构，并且鱼身就产生在矩形所在的行列上，而删数，则产生在矩形所在行列以外的其它任何位置上，并且，矩形所占行数和列数的较小者就是这个鱼结构的规格。

对照例子给的图。我们要找的矩形必须是涉及一行一列的，因为带入公式，1 的总个数是 7 个，而 $9 - 2 = 7$ ，所以矩形行数加矩形列数必须是 2 才行，而一行一列代入公式： $1 + 1 = 2$ ，故需要找一行一列的矩形，也就是找一个单元格。而显然，从这个定理里，

我们说过矩形行数和列数的较小者是鱼的规格，所以一行一列要是真能出现，说明鱼结构的规格是 1，这在鱼里显然是不可能的（因为鱼的规格至少为 2。当然了，如果你在找鱼结构之前，没有发现题上还存在“残留”的行列排除的话，那么按照这个定理去找东西，而且要是真的找到了，那么它就对应这个行列排除）。

试想一下，假如我们已经找全了当前盘面下给出的所有 1 的排除，说明我们连矩形规格的情况都不存在了，所以肯定不可能有 1 的鱼。所以我们继续找数字 2。

数字 2 一共有三个确定值，这意味着，要想让 NAIN 值是 3，就必须满足 $9 - 3 = 6$ ，所以矩形一般需要找的是两行四列、三行三列和四行两列的，并且矩形所在的行列和矩形本身都不能含有数字 2 和任何的候选数 2。实际上，你可以仔细寻找一下这种矩形，这个矩形你是找不到的，因为怎么找，始终都会包含一个候选数 2 的单元格。所以 2 看似也没有鱼结构，那么我们去看看数字 3。

数字 3 也是一样。数一下确定值个数：2 个，所以公式 $9 - 7 = 2$ ，我们需要找矩形行数和列数总和为 7 的结构（例如三行四列）。比如我给出一个错误示范：

7	6	^{4 5} ₃	1	2	⁵ ₈	9	⁴ ₈	^{4 5} ₈
^{2 3} _{4 5}	² ₅	1	⁵ ₈	⁵ _{7 8}	9	^{4 5} ₃	6	^{4 5} _{7 8}
³ ₅	8	9	4	6	⁵ ₇	2	1	⁵ ₇
² _{4 8}	1	6	9	⁴ _{7 8}	^{2 3} _{4 8}	³ ₄	5	^{2 3} _{4 8}
9	² _{4 5}	7	^{2 3} _{5 8}	^{4 5} ₈	^{2 3} _{4 5 8}	6	^{2 3} _{4 8}	1
² _{4 5 8}	3	^{4 5} ₈	6	1	² _{4 5 8}	7	² _{4 8}	9
¹ _{5 8}	7	2	⁵ ₈	^{4 5} ₈	6	¹ _{4 5}	9	³ _{4 5}
6	4	³ ₅	7	9	1	8	^{2 3} ₅	^{2 3} ₅
¹ _{5 8}	9	⁵ ₈	² _{5 8}	3	² _{4 5 8}	¹ _{4 5}	7	6

例如这个例子就是错误的寻找方式。虽然这个矩形区域恰好是四行三列，代入公式是正确的（ $9 - (4 + 3) = 2$ ），但是在矩形所在的行列里包含数字 3 的确定值，即此处的 c5 上。所以矩形是无效的，我们只得重新找。

遗憾的是，数字 3 确实没有我们想要的结构，所以只能去找数字 4。

数字 4 的寻找我们就省略了，我们直接来看数字 5（因为我可以明确地告诉你的是，数字 5 确实存在，而数字 4 不存在普通鱼，为了讲解不罗嗦，我们就跳过寻找过程了，逻辑和方式和上面的找 1、2、3 的方式完全是类似的）。

首先，我们发现全盘只有一个数字 5 的确定值，那么根据要求，我们需要寻找的是 $9 - 1 = 8$ 的情况，即行数和列数之和要为 8 的矩形（比如四行四列）。只要我们能找到一个四行四列的区域，且这个矩形不包含候选数 5 或 5 的确定值，且涉及的行列也是的话，那么根据定理，我们就可以立马得到数字 5 存在一个普通的四链列结构。实际上，确实存在，如图所示。

7	6		1	2		9		
		1			9	6		
	8	9	4	6		2	1	
	1	6	9			5		
9		7				6		1
	3		6	1		7		9
	7	2			6		9	
6	4		7	9	1	8		
	9			3			7	6

可以看到这此时寻找的区域是四行四列的，并且单元格以及所处行列的任意位置都不包含数字 5 的确定值，而且矩形涉及的每一个单元格都是确定值（根本不包含候选数 5），所以这个矩形是合格的。那么，我们根据定理可以得到，数字 5 存在一个四链列，且四链列产生在矩形所处行列上，而删数则产生在矩形行列都“扫”不到的地方。

实际上，这个鱼在这里（如左图所示），而如果你看着别扭，我们提供了普通视角（如右图所示）。

7	6		1	2		9		
		1			9	6		
	8	9	4	6		2	1	
	1	6	9			5		
9		7				6		1
	3		6	1		7		9
	7	2			6		9	
6	4		7	9	1	8		
	9			3			7	6

确实完全符合我们的预期，是一个很普通的四链列，而且结构的删数确实也都产生于矩形“扫”不到的地方。这种观察是不是很厉害呢？而且我们还可以从公式里得到一点：如果确定值有 6 个及其以上的时候，根据公式，我们需要找矩形行数 and 列数和为 3 的情况，但此时即使存在鱼，规格也是 1，所以不符合要求。所以我们可以下结论：**存在鱼结构的数字在全盘确定值不能超过 5 个**。5 个就是极限情况，即只能出现二链列。

这种观察角度好在它基本上不关心候选数情况（很少关注候选数，也就是在找矩形的时候，矩形里不含该候选数的格子），这样我们只要去找不是 5 的数字，然后画个矩形就 OK 了，实际上这种观察很简单，而且效率很高，一般而言，如果经常找鱼结构的话，有经验了，

寻找大概一分钟就能把 1 到 9 里所有数字都看一遍，而且能马上分辨出哪些数字有鱼结构，哪些数字没有。

总结一下，我们要找的矩形要满足如下规则：

- 矩形的单元格里不能包含任何的该数的确定值；
- 矩形涉及的单元格里不能存在有含有该数的候选数情况的单元格；
- 矩形所属行列上的任意单元格都不含有该数的确定值；
- 用 9 去减矩形的行数，再减列数为 NAIN 值，该值必须等于该数字的总确定值个数。

要是满足上述的所有要求的矩形，那么矩形就是合格的，且鱼结构存在。

下面我们来看鳍链列（鳍鱼）是怎么观察的。

4-3-2 鳍链列的观察

鳍链列和普通链列的观察规则大部分相同，只有一点稍微不一样：**矩形内可以含有该数的候选数情况的单元格（但是这样的单元格的个数是有限制的，具体什么限制我们将稍后提起）**。而且，退化和孪生链列（退化鱼和孪生鱼）最终都是带鳍的结构，所以它们都可以使用该小节给出的方式来观察。我们来思考如下例子：

1 2	1 2	2	5	1 2		4	1 2	3
7 8	7 8	7		9 7	9		7	
1 2 3	1 2 3	9	2 3	6	4	5	8	1
7	7		7				7	
5	4	2 3	2 3	1 2	8	9	1 2	1
		7	7				7	6
3	5	3	9	7	2	1 3	1 3	4
8		6				8	6	
9		1	6	4	3		5	2
	7 8					7 8		
2 3	2 3		4	8	5	1	3	9
7	7	6				7	7	
	6	9	5	1	3		2	4
7					7	6		
4	1 2 3	2 3	2	8		1 3	1 3	5
	7	7	7		7	6	7	
1 2 3	1 2 3			2	5	1 3	1 3	1
7	7	6	8	4	9	7	7	6

我们可以使用刚才的逻辑来寻找结构，并且矩形里可以含有该候选数的单元格了。那么我们依旧按照刚才的逻辑来寻找。寻找过程因为大部分都是相同的，所以找的过程我们就省略了，直接来看结论。如图所示。

1 2 7 8 1 2 3 7	1 2 7 8 1 2 3 7	2 6 7 6 9	5 2 3 7	1 2 9 7 9 6 4 1 2	4 7 6 5 8 1 2 1 6	1 2 7 6 3
5 3 6 8	4 7	2 3 6 7	9 7	7 2 8	1 3 6 8	1 3 6 4
9 7 8	1 7	6 7	4 7	3 7 8	5 7	2 7
2 3 6 7	2 3 6 7	4 7	8 7	5 7	1 6 7	3 6 9
6 7	9 7	5 7	1 7	3 7	2 6	4 7
4 7	1 2 3 6 7	2 3 6 7	2 7	8 7 9	1 3 6 7	1 3 6 5
1 2 3 6 7	1 2 3 6 7	8 7	4 9	2 7	5 7	1 3 6 7

可以看到示例里，一共有 3 处 2 的确定值，那么我们找的结构必须是行数加列数为 6 的矩形，此处我们找的是三行三列的矩形，符合要求。

不过我们发现了 r8c3 是含有候选数 2 的，那么，我们就可以这么思考了：

- 如果 $r8c3 \neq 2$ ，则矩形区域就不含任何的候选数 2 了，矩形成立，而且还印证了盘面一定存在关于 2 的普通的三链列；
- 如果 $r8c3 = 2$ ，则它只能影响到它的相关格的 2（即只能删除 r8c3 相关格里的所有位置的候选数 2）。

而我们可以从最开始的定理知道，普通鱼产生的删数在矩形所在行列外的其它位置，所以我们就可以得到一个结论了：最终这个鱼的可能删数就是 $\{r7c1, r9c2\}(2)$ 了。当然了，r7c1 不含有 2，所以删数是 r9c2(2)。而实际上，这条鳍鱼确实是存在的，而且也符合我们的预期。

1 2 7 8 1 2 3 7	1 2 7 8 1 2 3 7	2 6 7 6 9	5 2 3 7	1 2 9 7 9 6 4 1 2	4 7 6 5 8 1 2 1 6	1 2 7 6 3
5 3 6 8	4 7	2 3 6 7	9 7	7 2 8	1 3 6 8	1 3 6 4
9 7 8	1 7	6 7	4 7	3 7 8	5 7	2 7
2 3 6 7	2 3 6 7	4 7	8 7	5 7	1 6 7	3 6 9
6 7	9 7	5 7	1 7	3 7	2 6	4 7
4 7	1 2 3 6 7	2 3 6 7	2 7	8 7 9	1 3 6 7	1 3 6 5
1 2 3 6 7	1 2 3 6 7	8 7	4 9	2 7	5 7	1 3 6 7

如图所示。是一个三阶的鳍鱼（鳍三链列）。

那么我们可以了解到的是，实际上这个含有候选数 2 的矩形的单元格 r8c3 实际上最

后被转换为了普通视角的鱼鳍，而鱼鳍在一般层面下是最多只能两个的（在孪生鱼里可能有最多有四个，例如之前的例子那样。不过孪生鱼最终是可以被拆解称两个一般的鳍鱼的，所以实际上还是只有最多两个鱼鳍）。所以，我们找的矩形里最好不要包含三处甚至更多的候选数的单元格，这样会带来很大的麻烦，而且两处包含该候选数的单元格最好同一个宫，这样才符合我们最开始提到的普通视角的鱼的删数逻辑。

那么观察鱼的内容我们讲到这里，我们最后整理并给出六则示例，提供给大家用来作参考和理解使用。

4-3-3 最后的例子

这里陈列六个观察的示范示例，这些例子我们都没有给出普通视角的鱼到底是怎么画出来的，你可以尝试着寻找一下，并且自己把它们画出来。

9	4	5	^{2 3} ₇	^{2 3} ₇	6	1	8	^{2 3} ₇	6	^{1 3} ₇	2	^{1 3} ₇	9	^{5 8} ₈	4	^{5 8} ₈
1	3	² ₇	8	4	9	² ₇	6	5	8	^{1 3} ₄	5	^{1 2} ₄	6	^{1 2} ₅	9	7
8	² ₆	² ₇	5	1	^{2 3} ₇	9	³ ₇	4	9	4	7	8	5	^{1 2} ₆	6	^{1 2 3} ₁
7	² _{5 6}	3	9	² _{5 6}	8	4	1	² ₆	^{1 2} ₇	9	4	3	8	5	7	^{1 2} ₆
² _{5 6}	9	1	4	^{2 3} ₇	^{2 3} ₇	² ₆	³ ₇	8	3	⁵ ₇	8	6	² _{4 7}	^{1 2} ₄	^{1 2} ₅	¹ _{4 5}
² ₆	8	4	^{2 3} ₇	^{2 3} ₆	1	5	9	^{2 3} ₇	^{1 2} ₇	5	6	¹ _{4 7}	² _{4 7}	^{1 2} ₅	^{1 2 3} ₈	¹ _{4 5}
4	7	9	6	8	5	3	2	1	5	6	1	9	⁴ ₇	⁴ _{7 8}	³ ₈	2
^{2 3} ₆	^{1 2} ₆	8	^{1 2 3} ₇	^{2 3} ₇	4	⁷ ₆	5	9	4	2	3	5	1	^{7 8} _{8 9}	⁶ _{7 8}	⁶ _{8 9}
^{2 3} _{5 6}	^{1 2} _{5 6}	² ₆	^{1 2 3} ₇	9	^{2 3} ₇	8	4	⁷ ₆	7	8	9	2	6	3	4	¹ ₅

3	2	^{4 5} ₉	1	^{4 5} ₉	7	6	9	8	³ ₆	5	9	2	8	7	⁴ ₆	1	⁴ ₆
⁴ ₉	⁵ ₉	8	6	2	^{4 5} ₉	7	1	3	2	1	8	³ ₆	4	9	7	³ ₆	5
¹ ₇	¹ ₇	6	3	9	8	2	4	5	4	³ ₆	7	¹ ₆	³ ₆	5	¹ ₆	2	8
6	8	³ ₉	⁴ ₅	⁴ ₅	^{1 2} ₄	⁵ ₃	7	¹ ₄	5	8	³ ₆	7	9	4	1	³ ₆	2
5	⁴ ₉	7	8	3	¹ ₄	¹ ₄	2	¹ ₄	7	³ ₆	4	¹ ₆	2	¹ ₆	³ _{8 9}	5	⁶ ₈
2	⁴ ₉	1	^{4 5} ₉	7	⁴ ₆	⁵ ₃	8	⁴ ₆	¹ ₆	² ₆	^{1 2} ₉	5	3	⁸ ₆	⁶ _{8 9}	4	7
¹ ₇	6	² _{4 5}	² _{4 5}	8	9	¹ ₄	3	^{1 2} ₄	¹ ₃	^{2 3} _{8 9}	^{1 2} ₉	⁴ ₈	6	5	⁴ ₈	7	¹ ₄
⁴ ₉	³ _{5 9}	³ ₉	7	1	² _{4 5}	8	6	² ₄	¹ ₃	³ ₆	4	5	9	7	³ ₈	2	¹ ₃
8	¹ ₇	² ₄	² ₄	6	3	9	5	¹ ₇	³ ₈	³ ₆	7	⁴ ₈	1	2	5	9	³ ₈

4 5	9	7	4 5	6	8	2	3	1	9	5	7	8	1 2	3	4	6	1 2
3	6	4 5	2	1	2	7	4 5	8	6	2	1 2	7	9	5	8	1 2	3
8	1	2	3	4 5	7	4 5	6	9	3	8	1 2	1 2	1 2	6	7	5	9
7	2	6	4 5	4 5	1	4	8	3	8	9	3	1 2	1 2	7	6	2	5
9	4 5	8	6	7	3	1	4 5	2	4	2	1 2	6	5	2	1 3	2	7
1	4 5	3	8	2	4	4 5	7	6	7	1 2	5	3	6	2	1 3	2	2
2	3	1	1 2	4	6	3	9	4 7	1	6	5	3	4	2	7	1	8
6	3	1	1	4	5	3	2	4 7	5	7	2	4	8	1	9	3	6
4	2	8	4	7	3	4	6	1	1 2	3	6	7	2	5	1	8	4

4-4 来个练习题？

我们给出一个题目，这些题目只需要用到排除、唯一余数和本章给出的鱼技巧。慢慢享用吧！

	4	5			6		8	
1				4			6	
						9		
7		3	9		8			
	8				1	5		
	7		6				2	1
		8			4			
				9		8	4	

别看这个题连候选数都没有给你标注出来，实际上这个题使用了我们之前解释的观察技巧就可以发现，它实际上根本就用不上候选数。所以这题我也没有标注候选数，作为挑战，希望你加油。

Part 5 分支匹配 (Wings)

接下来要提到的技巧是一个和之前的逻辑不太相似的技巧，而且这个技巧没有中文名（所以标题的中文名是我胡诌上去的）。

5-1 XY-Wing

5-1-1 一个看起来简单的例子

	3		6		2	1	4	5	9
7		8		7 8					
	4		4	7 8	5	3	6	1	2
5	2	1	6	9	4	3		7 8	7 8
1			5	2	4	6		3	1
	9	8					7 8 9		7 8
2 3		4	7	1	8	9	5	2	6
								4	
1 2	6		5	3	7		2	2	1
		8					8 9	4	8 9
4	5	2	9	7	8	1	6	3	
8	1	3	4	6		2	2	2	5
6	7	9	3	1		2	2	2	
							8	4	4 5
								8	8

如图所示，我们观察到，r1c2 只有两种填数情况：3 和 8。

- 如果 r1c2=3，则 r5c2=4；
- 如果 r1c2=8，则 r2c3=4。

那么，不管是其中哪种情况，最终 r5c2 或 r2c3 **至少一格**会填 4。所以，不论怎么个情况，它们两个共同对应的地方，都不能填入 4 了。所以，r2c1, r6c3 \neq 4。

这个技巧的推导过程就讲完了。不过，你可能会向我提出两个问题：

- 为什么是“至少一格填 4”，而不是“有且仅有一格填 4”？
- “共同对应的地方”到底是什么地方？

那么下面我来一一阐述一下。

5-1-2 为什么不是“有且仅有一格是 4”？

首先是问题 1。我们按照假设来看的话，要么 r1c2 填 3，要么 r1c2 填 8。如果至少一格填 4，意味着还有可能 r5c2 和 r2c3 两格同时为 4，这意味着 r1c2 得同时既填 3 又填 8 的时候，才可能让两格同时为 4。可是一个单元格怎么可能填两个数，这是违背常理的呀。

别忘了。虽然 r1c2 填其中一种情况只能得到其中一个单元格是 4，就比如说，当 r1c2

构型 1 给出的形式是类似于钝角的形状。从 r2c3 开始推理，发现 r3c2 和 r2c7 至少有一个是 z，所以删除共同对应的未知，即交集处：r3c789(z)；

构型 2 则是一个矩形形状。从 r2c3 开始推理，得到 r2c7 和 r6c3 至少有一个是 z，删除交集 r6c7(z)。

5-2 XYZ-Wing

接下来我们来看第二个示例。

1	6	3	5	3	1	5	2	7	9	4	8
2	8	7	4	3	9	6	1	5			
1	1	4	5	1	5	6	8	7	2	3	
6	8	9	3	6	9	8	7	4	5	1	6
5	7	1	6	9	2	8	3	4			
4		6	2	3	8	1	5	7	6	9	
7	1	4	4	8	9	5	3	2	6	1	6
3	2	9	8	1	6	4	5	7			
1		5	6	2	7	4	3		1		9

如图所示。这个例子推理起来和 XY-Wing 没啥两样，区别仅在于 r3c2 这个“拐角”处多了一个候选数 1。如果 1 不存在，则当 XY-Wing 直接用就可以了；而它的客观存在，导致最终的删数只能看 r3c2(1)的相关格和 XY-Wing 删数位置的交集了。所以，实际上找的是 r3c12(1)和 r7c2(1)这三个 1 的交集。

实际上，这个分类讨论的初始单元格 r3c2（在前文里用“拐角”描述的）在这个技巧里就叫**拐点**或**折点**（Pivot）。

这个技巧叫做 XYZ-Wing。不过这个技巧的推理过程非常类似于 XY-Wing，也类似于鱼鳍，所以一些资料也将这种结构称为 Finned XY-Wing，即鳍 XY-Wing。

5-3 WXYZ-Wing

既然刚才已经有了两个不同的示例了，那么我们可以发现，它的结构实际上在慢慢变“大”，那么我们来看，下面的例子。

5-3-1 普通的 WXYZ-Wing

	3 5 7 9	3 5 7 9	1	3 5 9	8	2	4	6	3 5
	4	2 3 5 9	2 3 9	3 5 9	6	3 9	8	7	1
	3 5	8	6	7	4	1	9	2	3 5
5 7	5 7	4	1	3	8	6	9	2	
6	1	9	2	5	7	3	8	4	
8	2 3	2 3	6	9	4	5	1	7	
1	3 9	8	3 9	7	5	2	4	6	
2	4	5	8	1	6	7	3	9	
3 9	6	7	4	2	3 9	1	5	8	

如图所示，和 XYZ-Wing 的假设逻辑完全一样。我们此时得分四种情况来看了：

- 如果 $r2c2 = 2$ ，则 $r2c3 = 3$ ；
- 如果 $r2c2 = 5$ ，则 $r3c1 = 3$ ；
- 如果 $r2c2 = 9$ ，则 $r7c2 = 3$ ；
- 还有一种情况就是 **$r2c2$ 自己填 3**。

可以发现，这四种情况里必须有一个是成立的，因为假设条件就是按 $r2c2$ 来分情况讨论的，所以这意味着 $r2c3$ 、 $r3c1$ 、 $r7c2$ 和 $r2c2$ 四格里必须至少有一个是填 3 的。不过，不管哪一条成立，它们共同对应的位置（ $r1c2$ ）就不允许填 3 了；否则它会同时使得前面叙述的这四个单元格都不能填入 3，进而违背推导的结论。

这个技巧由于涉及了 2、3、5、9 四种不同的数字，所以称为 WXYZ-Wing。

5-3-2 折点残缺的 WXYZ-Wing

我们再来看一则例子。

<div>2 3</div> <div>7 8</div>	<div>5</div> <div>8 9</div>	<div>5 3</div>	<div>3</div> <div>7 8</div>	4	<div>2 3</div> <div>7 9</div>	<div>6</div> <div>5 8</div>	<div>3</div> <div>5 8</div>	1
<div>3</div> <div>7 8</div>	<div>8 9</div>	1	5	6	<div>3</div> <div>7 9</div>	2	<div>3</div> <div>8</div>	4
<div>2 3</div> <div>4 8</div>	<div>4 5</div> <div>8</div>	6	1	<div>2</div> <div>8</div>	<div>2 3</div> <div>5</div>	<div>3</div> <div>5</div>	7	9
6	<div>4 5</div> <div>8</div>	7	<div>3</div> <div>8</div>	1	<div>2 3</div> <div>4 5</div>	9	<div>2 3</div> <div>4 5</div>	<div>2</div> <div>5</div>
<div>3</div> <div>4 8</div>	1	9	6	<div>2</div> <div>8</div>	<div>2 3</div> <div>4 5</div>	<div>3</div> <div>5</div>	<div>2 3</div> <div>4 5</div>	7
<div>4</div> <div>8</div>	2	<div>3</div> <div>5</div>	<div>3</div> <div>7</div>	9	<div>4 5</div> <div>7</div>	<div>3</div> <div>4 5</div>	8	1
5	3	2	9	7	1	4	6	8
9	7	8	4	3	6	1	<div>2</div> <div>5</div>	<div>2</div> <div>5</div>
1	6	4	2	5	8	7	9	3

如图所示,如果我们把 $r4c2$ 补上候选数 3,你就会发现它就是一个正常的 WXYZ-Wing 了,不过这一个单元格不幸的缺少了一个候选数,但它并不会影响我们的推理,大不了只是少一种推理情况:

- 如果 $r4c2 = 4$, 则 $r6c1 = 3$;
- 如果 $r4c2 = 5$, 则 $r6c3 = 3$;
- 如果 $r4c2 = 8$, 则 $r4c4 = 3$ 。

所以不论哪种填法,最终 $r6c13$ 和 $r4c4$ 都会至少有一个 3,所以删除的应当是交集处的 3。

5-4 VWXYZ-Wing

最后我们再来看两个更大规格的例子。这两个例子仅供欣赏,因为我们确实在一般情况下都完全找不到它……

5-4-1 普通的 VWXYZ-Wing

8	9	3	¹ ₄	⁴ ₅	^{1 2} ₅	^{1 2}	7	6
7	5	6	¹	8	^{1 2} ₉	3	^{1 2}	4
1	4	2	6	7	3	8	5	9
5	8	¹ ₇	¹ _{4 7 9}	2	¹ ₉	¹ _{4 6}	¹ _{6 9}	3
4	2	9	¹ ₈	3	6	7	¹ ₈	5
6	3	¹ ₇	¹ _{4 7 8 9}	⁴ ₅	¹ _{5 9}	¹ _{8 9}		2
2	1	4	5	6	8	9	3	7
3	6	8	2	9	7	5	4	1
9	7	5	3	1	4	² ₆	² ₆	8

VWXYZ-Wing:

折点 r6c4(14789)

结构 r4c6(19), r5c4(18), r6c3(17), r6c7(14)

=> r6c4 <> 1

5-4-2 折点残缺的 VWXYZ-Wing

^{2 3}	8	^{2 3} _{4 6 9}	⁴ _{6 9}	5	1	7	³ ₆	³
³ ₉	1	³ _{6 9}	⁶ _{8 9}	2	7	5	4	³ _{8 9}
5	7	⁴ ₆	9	3	⁴ _{6 8}	¹ _{6 8}	¹ _{6 8}	2
4	2	5	7	¹ _{6 8 9}	³ _{6 8 9}	¹ _{3 8}	¹ _{3 8}	¹ _{3 8}
7	³ ₉	8	¹ ₄	¹ _{4 9}	2	¹ _{4 9}	5	6
6	³ ₉	1	5	⁴ _{8 9}	⁴ _{8 9}	⁴ _{8 9}	2	7
^{2 3} ₉	6	^{2 3} ₉	¹ _{4 7 8 9}	³ _{4 6 7 8 9}	5	¹ ₈	¹ _{7 8}	¹ _{4 8}
1	5	³ ₉	⁴ ₈	⁴ _{7 8 9}	⁴ _{6 8 9}	2	³ _{7 8}	³ _{4 8}
8	4	7	2	¹ ₆	³ ₆	¹ ₆	9	5

VWXYZ-Wing:

折点 r7c4(1348)

结构 r1c4(46), r2c4(68), r9c5(16), r9c6(36)

=> r8c4 <> 6

5-5 为啥英文名这么怪？

初学者一定会蒙蔽的地方是，第一，XY-Wing 明明涉及了三种数字，却叫 XY-Wing，而不是 XYZ-Wing；而第二，则是为什么这么草率的在 Wing 前面加各种字母就可以了。

下面就来解释一下。

5-5-1 XY-Wing 还是 XYZ-Wing？

这一点确实是一个棘手但没办法的问题。虽说 XY-Wing 涉及了三种数字，按理说它应当取名为 XYZ-Wing（哪怕把这种结构看作是折点残缺的 XYZ-Wing），但实际上，只有 XY-Wing 这一个技巧脱离了取名的规范和规则，而其它的所有技巧都是遵循“多少种数字就多少个字母”的规范的，哪怕它残缺了。

所以，请记住，XY-Wing 是一个特殊例子即可。

5-5-2 “字母-Wing” 模式？

这个模式是外国人固定的命名模式，因为 XY-Wing 用了 Wing 了，所以干脆就直接在 Wing 前面来改掉字母，以表示这类技巧的通性：分情况讨论（当然了，此时 X-Wing，即二链列技巧就已经不属于这个 Collection 了）。

实际上，带有 Wing 的技巧还有很多，这一点将在后面说到，不过此处提到的所有 Wing 有一个共性，就是 Wing 前面的字母都是按 XY 为基准，向左或向右补充字母。先向右补充到 Z 后，再向左补充字母 W、V、U 等等。

理论上，这种结构可以达到 9 个单元格分支情况，即 RSTUVWXYZ-Wing，不过……五个数的情况 VWXYZ-Wing 就已经不容易发现了，到目前位置，我也仅能找到两则示例，更别说六种数字，甚至更多数字的情况了。

Part 6 唯一矩形 (Unique Rectangle)

接下来，我们又要进入一个大的技巧板块——**致命结构 (Deadly Pattern)**，不过这个板块和鱼一样，体系非常庞大，所以我们先介绍一点基础和实用的内容，然后再在稍微后面一点的地方针对这个技巧的体系进行深层次的拓展。

6-1 经典唯一矩形 (Basic Unique Rectangle)

现在来看一下新的技能。这个技能需要建立于每一个数独题目都只能有唯一的一种填法的这样的一个说法之上。不过结构的原理比较复杂。

6-1-1 标准类型 (UR Type 1)

7	2	¹ 6	¹ 3	4	³ 6	8	9	5
8	4	¹ 5	¹ 5 9 7	¹ 7 9		6	2	3
9	3	5 6	5 8	2	⁶ 8	4	7	1
2	7	8	6	9	5	1	3	4
6	9	4	7	3	1	5	8	2
5	1	3	4	8	2	7	6	9
1	8	9	² 3	6	4	² 3	5	7
3	5	2	¹ 8 7	¹ 7 8		9	4	6
4	6	7	² 3 9	5	³ 9	² 3	1	8

如图所示，这个结构好像是数对，对没有错，这个跟数对很相似，但逻辑却不同。

如果我们暂时不看 **r9c4(9)**，先当它不存在。于是乎，你就会发现一个神奇的现象：在 **r79c47** 这四格上，全部只有候选数 2 和 3。那么，这意味着，**r79c47** 所在的区域上（所在区域有：**r79**、**c47**、**b89**，注意 **b89** 别漏掉了），都会受到数对的影响，得到删数。比如 **r79c4(23)** 则可以删除 **c4** 和 **b8** 内的其余单元格的 2 和 3。

接着，我们来思考一下。如果删了，**r79c47** 四格以外的其余空单元格，除了删了 2 和 3 之后，还是照样没有其它的变动；而这四格就很神奇了。数对内部只有两种填数情况：要么自己填 2，对面填 3；要么自己填 3，对面填 2。所以，这四格既然都是这样的，那必然只能产生如下两种填数状况。

7	2	¹ ₆	¹ ₃	4	³ ₆	8	9	5	7	2	¹ ₆	¹ ₃	4	³ ₆	8	9	5
8	4	¹ ₅	¹ ₅	¹ ₉	⁷ ₇	6	2	3	8	4	¹ ₅	¹ ₅	¹ ₉	⁷ ₇	6	2	3
9	3	⁵ ₆	⁵ ₈	2	⁶ ₈	4	7	1	9	3	⁵ ₆	⁵ ₈	2	⁶ ₈	4	7	1
2	7	8	6	9	5	1	3	4	2	7	8	6	9	5	1	3	4
6	9	4	7	3	1	5	8	2	6	9	4	7	3	1	5	8	2
5	1	3	4	8	2	7	6	9	5	1	3	4	8	2	7	6	9
1	8	9	³ ₆	6	4	² ₅	5	7	1	8	9	² ₆	6	4	³ ₅	5	7
3	5	2	¹ ₈	¹ ₇	⁷ ₈	9	4	6	3	5	2	¹ ₈	¹ ₇	⁷ ₈	9	4	6
4	6	7	² ₅	5	³ ₉	³ ₁	1	8	4	6	7	³ ₅	5	³ ₉	² ₁	1	8

那么，我们这么去想问题：如果说，既然这么两种填法都是这样四格的内部填数情况，而刚好，对于其余单元格下，除了数对删数外，其余候选均不会发生任何变动。那不就意味着，这四格就有两种填数方法，而剩余盘面完全一致，岂不是就两个填法了？对呀，因为这样两种填法，除了对应区域上的 2 和 3 会被删掉外，对剩余盘面就没有其余的任何影响了，这样就说明这四格是可以互换的、并且等效的两种填数情况，其中一种填法对，那另外一种肯定也是对的。

那如果盘面继续往下做，一个唯一答案的题目，怎么可能允许两种填数方式完全不同却不会影响剩余盘面的结构存在呢？所以，这样的结构必然不可以存在。因此，最开始的假设，即那个 9 也就不应该被忽视掉，它才是重中之重！因此，我们可以得到 $r9c4 = 9$ 的特别情况。

总的来说，解释可以理解为这样：一个唯一解的数独题目，只允许让每一个单元格都只有唯一的一种填数可能。若产生了上述 2 和 3 交换的填法，就相当于在这四格里产生了两种填数可能性，即使只有四格可以有两种填法，它也是两种填法啊，唯一解的要求应当是针对全盘的每一个单元格都只有一个填法的，所以很显然地，但凡有一个单元格有两种及其以上的填法，这都是不行的，这样就与唯一解要求矛盾了（即与“唯一的一种填数可能”矛盾）。

当然了，如果 $r9c4$ 还有非 2 和 3 的其余候选数的话，那么就应该说明，这里的 2 和 3 以外的其余候选数，至少都得有一个是正确的：如果全都不是的话，那就说明可以填 2 和 3 咯，那不就形成之前说的矛盾的结构了吗？所以一个通用的结论就是： $r9c4 \neq 23$ 。

这个结构被称为**唯一矩形**（Unique Rectangle，简称 UR）。这里的“唯一”一词，指的是结构必然只能有一种情况，使得最后的盘面也只能是唯一答案；当然了，你叫它“唯一长方形”也没毛病，只是呢，术语用的“矩形”一词。另外，这样会导致内部填数出现多种（2 种甚至更多），但不会对剩余盘面造成其它影响的结构，我们就称为**致命结构**，而唯一矩形就属于致命结构的其中一种，而形成了“对剩余盘面不造成其余影响、长相全一样”的效果的形式，称为**形成了致命形式或形成了致命模式**（has formed a deadly pattern）。那么，上面的结构被称为基本结构，所以也叫唯一矩形的**标准类型或类型 1**（UR Type 1），有时候也叫**第一类唯一性测试**（Uniqueness Test 1）。

6-1-2 原理进一步剖析

为了阐述清楚其原理和逻辑，我们将罗列出新人在学习 UR 的时候常见的几个问题。

6-1-2-1 #1: UR 能分属于四个宫吗？

如果这个结构的四个顶点，分属于四个宫内（之前的结构只存在于 b89，所以称为“分属于两个宫内”），那唯一矩形结构的使用还成立吗？

不可以的哈。根据刚才的证明思路 and 过程，比如这个题：

6		3	1	4	2	5		3	9	8						
8		2	5	3	1	9		2	4	6						
	2	3	4	9	7	8	6	1	5		2	3				
9	8		3	1	5	2		3	6		3	6	4			
5	6	4	8	9		3		2	3	1	2	3	1	2	3	
1		2	3		2	3	6	4		3	9	8	5			
	2	3	1	8	9	6	4	5	7		2	3				
4	5		3	2	7	1	8		3	6	9					
7	9		2	6	5	3	8	4		1	2	6		1	2	6

如图所示，这个题四个顶点就分属四个宫内（b1379）。但是，这样内部的填数互换起来，就和刚才的不太一样了。

6	³	1	4	2	5		9	8	6	³	9	8					
8	₇		5	3	1	9	₇ ² ₆	4	₇ ⁶	₇ ⁶	4	₇ ⁶					
2	4	9	7	8	6	1	5	3	3	4	9	7	8	6	1	5	2
9	8	³	1	5	2	₇ ³ ₆	₇ ³ ₆	4	₇ ³ ₆	₇ ³ ₆	1	5	2	₇ ³ ₆	₇ ³ ₆	4	
5	6	4	8	9	³	₇ ² ₃	₇ ¹ ₂ ³	1	₇ ³	₇ ² ₃	₇ ¹ ₂ ³	1					
1	² ₃	² ₃	6	4	³	9	8	5	₇ ² ₃	₇ ² ₃	6	4	³	9	8	5	
3	1	8	9	6	4	5	7	2	2	1	8	9	6	4	5	7	3
4	5	⁶	2	7	1	8	³ ₆	9	4	5	³ ₆	2	7	1	8	⁶	9
7	9	² ₆	5	3	8	4	¹ ₆	¹ ₆	7	9	⁶	5	3	8	4	¹ ₂ ₆	¹ ₆

你会发现，这样对照起来，你就看得到，b1379 内其余单元格的填数情况是完全不一致的。这就意味着，这个结构并不是唯一矩形。所以，唯一矩形一定分属于两个宫内，分属于四个宫的结构一定不是唯一矩形。

当然，这也是唯一矩形的“矩形”一词的来源：这种违背唯一性要求的技巧只可能是长

方形形状的结构，随便找个直角梯形、甚至是任意的普通的梯形形状的结构肯定已经不能满足推导要求了，毕竟会影响到一部分行或列的填数，而不是完全排除影响的。

所以，为了满足技巧可用性，至少要满足两点：

- 1、技巧涉及的四格必须是分属两个宫内的；
- 2、技巧四格代表的顶点，围成的形状只允许是长方形，而不可以是梯形。

6-1-2-2 #2：UR 的结论是出数还是删数？

我们再来看一则示例。

5	5	3 1	4	8	6	1 3	3	2
7 1		3 1	1	2	7	1 3	5	1
4	4 6	6 9		9		4 6	8 9	5
2	4 6	8	1 5	5 9	1 3	1 3	4 6	7
1 3	9	1 6	2	7	5 3	4 6	8	4 5 6
8	4 6	5	1 6	4 9	1 9	7	2 9	3
4 3	7	2 6	5 6	4 5 9	8	2 9	1	4 5 6
6	8	4 3	7	1	2	5	4 3	9
5 7	1	4 3	8	5 6 9	5 9	2 3	2 3	4 6
9	2 5	2 7	3	5 6	4	1 8	7 6	1 8

如图所示。这个示例里 r2c7 含有 6 个候选数，那么此处和前一个示例有着一区别，但实际上从前面的示例可以看到，它实际上的结论是出数，而这个示例则无法出数，那么，这个结构还能使用 UR 吗？

实际上是可以的，结论就是 r2c7 <> 18。原因得反过来想。因为最初的推导过程之中，我们都是假设这些“额外的数字”不存在，然后再得到结构本身致命，从而得到的结论。这个题也是一样。如果说 r2c7 没有 3、4、6、9 这些候选数的话，即 r2c7 只有 1 和 8 两个候选数的话，这样就会构成关于 1 和 8 的唯一矩形致命形式，所以说，r2c7(3469)至少有一个数是正确的。不管它们之中谁是对的，r2c7 都会有一个填数，而这个数一定都不会是 1 和 8，所以 r2c7 <> 18。

所以实际上，UR 的实际推导结论是删数，而非出数。

6-1-2-3 #3：已经形成了致命形式的唯一解题目还往下做，会怎样？

这一个问题甚至不用去用示意图来说明就可以表达清楚。因为题目是唯一解的，一旦你错误地继续向下做题，就意味着在此之前你删除了本不应该删除的数字（假设那个数不存在才会形成致命形式，导致的两种填法，而违背唯一解的规则。所以反过来说，既然你都假设原本正确的数字不存在了，还要继续往下做，就说明你已经删除了本不该删除的数字了），这只会导致题目无解，虽然你看着 UR 涉及的四个单元格好像确实有两种填法，但实际上已经让整个盘面无解了。

6-1-2-4 #4: 非唯一解的题目使用了 UR，会怎样？

这个问题是问得最好的一个问题，也是最难解释明白的问题，而且这里仅给出理论的说明，没有示意图，因为针对于多解和无解的题目很少去研究和使用的 UR，这本身就是违背了 UR 的使用条件，所以相对于其它技巧，这种“误用”的例子并不多。

实际上，非唯一解的题目出现类似于 UR 形式的结构，如果你对此使用了 UR，那么题目的状态将变得不稳定，即什么情况都可能发生，即变为多解、变为无解、甚至变为唯一解，它们全部都是可能的。多解和唯一解不同的地方在于，它的形式变化多端，多解产生的原因就是在某些数字之间能够互相交换。而如果不受影响的地方使用了 UR，就完全不会更改这个题目的唯一性，即“该多解还多解”；而如果 UR 使用在了产生互换的位置上，那么问题就变得非常棘手了：它可能使得题目唯一解，也可以使得题目无解，因为 UR 的形式涉及的数值和形成多种填数模式的地方还需要继续看是否继续有冲突，所以这一点并不能够立马断言确定下来。所以，题目就可能多解、唯一解和无解，即全部都可能出现。

所以我们说，我们不应对多解题或无解题目使用 UR 技巧，否则变为唯一解题目了，你可能还指着它说，“你看，它是唯一解的题目”，这一点是最能迷惑人的。这一点希望你引起注意。而且，我们不能说一个多解题在使用 UR 后变为唯一解了，就说题目本身是唯一解的。这本身就是错误的，因为初盘是多解的。所以这也就意味着，使用 UR 的前提，必须是你得充分相信出题人或者发题人给你的题目是唯一解的；如果不能保证题目是唯一解的，就不要大胆使用 UR，因为 UR 的使用是有“唯一解”的局限性的，它比较“小气”。

6-1-3 区块类型 (UR Type 2)

4	5	8	9	2	3	6	7	1
2	1	6	4	7	5	³ 9	8	³ 9
³ 9	7	³ 9	1	8	6	4	2	5
¹ 6	9	^{1 2} 6	3	5	8	7	4	² 6
^{5 6} 9	8	^{4 5} 9	7	⁴ 9	2	1	³ 6 9	³ 6 9
7	3	² 4	6	⁴ 9	1	² 9	5	8
8	6	^{1 3} 5 9	^{2 5} 1 3	7	^{2 3} 5 9	³ 9	³ 9	4
⁵ 9	2	⁵ 9	8	6	4	⁵ 9	1	7
¹ 5 3	4	7	^{2 5} 1 3	9	8	³ 6	^{2 3} 6	

如图所示，这个结构就是刚才的结构的拓展版。我们可以看到，如果 r8c13(5)都没有了的话，r38c13 就会构成关于 3 和 9 的唯一矩形，至于推导过程这里就不再过多重复了哈。反正就是有 3 和 9 的数对，然后结构涉及的区域 r38、c13 和 b17 内，都会有这样的 39 显性数对，于是乎，这个结构内就有两种“互换”的填数情况，但都对其余单元格不造成除了 3 和 9 外的任何删数或出数影响，这样来违背唯一解的说法的。因此结构不允许存在，故 r8c13(5)里至少有一个 5 是格里面应该填的数。

那么，不管哪一个格是 5，这两个 5 就形成了一种类似于区块组的结构，那么，b7 和

r8 其余单元格都不应该有 5 的出现了，所以，r7c3, r8c7, r9c1 \neq 5。

另外，这样的结构称为**区块类型或类型 2 (UR Type 2)**，或者**第二类唯一性测试 (Uniqueness Test 2)**。

你以为 UR 就这么简单吗？我们要开始增大讨论的难度了！我们来看另外一则示例。

	5	1	3	2	4	8	1	6	1	6	5	6		5	1	3	2	4	8	1	6	1	6	5	6		
	8	1		2	1	3	1	3	1	5	6	5	6	8	1		2	1	3	1	5	6	5	6			
4	5	6	1	9	7	1	5	6	8	3	2		4	5	6	1	9	7	1	5	6	8	3	2			
2	3		3	1	3	1	3	1	2	3	6	7	8	4	5	6	1	3	1	3	1	2	3	6	7	8	
4	5	6	4	4	5	6	4	5	4	5	6	4	5	4	5	6	4	5	6	4	5	6	4	5	6		
2	3	8	7	4	3	3	2	6	9	5	1		4	5	6	8	7	4	3	3	2	6	9	5	1		
1	4	3	4	5	9	7	8	4	2	3	6	4	6	4	6	1	4	3	4	5	9	7	8	4	2	3	6
7	2	1	4	5	9	3	1	6	8	4	6	9		7	2	1	4	5	9	3	1	6	8	4	6	9	
4	3	5	6	8	1	1	4	5	7	2	4	9		3	5	6	8	1	1	4	5	7	2	4	9		
4	3	1	3	8	6	2	7	5	1	4	4	9		3	1	3	8	6	2	7	5	1	4	4	9		

如左图所示，这个例子里存在一个 XY-Wing 结构，并且我们通过删数后，存在右图的这种疑似的 UR 结构。

在删除掉 r8c1(4)之后，有一个特别的结构就浮出水面了：在 r89c19 处，只有一格只有 3 和 9，其它都是 3、4、9 这三个候选数。如果说，这三个 4 全部没有的话，剩下的所有 3、9 就会构成致命形式。所以，r8c9、r9c19 之中至少一格是 4。不管那个单元格是 4，这三个 4 共同对应到的地方，就不应该是 4，而{r8c9, r9c19}(4)共同对应的地方应该为 r9c78(4)，故结论为 r9c8 \neq 4。

这个结构罕见之处在于，它不能单独直接存在于盘面之中，而必须通过某个删数之后，才会形成，比如这里的 r8c1(4)。虽然看似这个例子跟区块已经无关了，不过它实际上想表示的是类似于区块的“所有区块的候选数不可同时被删除掉”的这一性质，所以这个构型依然被称为区块类型，不过它和区块已经没有什么关系了，只是借用区块这个名字而已。不过，在外国人的命名规范里，这种类型有时候被称为**类型 5 (UR Type 5)**或**第五类唯一性测试 (Uniqueness Test 5)**。

我们再来看一个“类型 5”的例子。

7	5	4	6	9	8	2	4	6	3	1
2	¹ ₄		9	6	¹ ₄	3	5	8	7	
¹ ₃	³ ₄	³ ₆	8	¹ ₄	5	7	9	² ₆	² ₄	
8	¹ ₂	¹ ₂		5	9	4	3	7	6	
6	⁴ ₇	⁴ ₇		3	2	1	8	9	5	
5	9	3	8	7	6	¹ ₄	¹ ₂	² ₄		
¹ ₃	³ ₆	³ ₇	5	¹ ₄	⁴ ₇	⁴ ₆	9	2	¹ ₄	⁶ ₆
9	8	¹ ₆	2	¹ ₄	⁴ ₆	5	7	⁴ ₆	3	
4	² ₇	² ₆	¹ ₇	3	8	¹ ₆	5	9		

如图所示，这个例子里，r7c8(1)和 r8c5(1)不可同时被删掉，否则会形成致命形式，所以至少有一个 1 是对的，所以删除交集，即 r7c4(1)。

6-1-4 数组类型（UR Type 3）

这一节，将讲述 UR 和数组交织的使用逻辑。

6-1-4-1 UR + 显性数对

4	1	6	⁵ ₉	8	⁵ ₉	2	⁵ ₇	³ ₇
7	5	3	1	2	6	9	4	8
2	8	9	4	⁵ ₇	³ ₇	⁵ ₆	1	³ ₆
6	² ₄	² ₄	7	³ ₉	³ ₉	1	8	5
8	9	5	2	1	4	⁶ ₇	3	⁶ ₇
3	7	1	⁵ ₆	⁵ ₆	8	4	9	2
5	² ₄	² ₆	8	⁷ ₉	⁶ ₇	1	3	² ₇
1	3	8	⁵ ₉	⁵ ₉	2	⁵ ₇	6	4
9	² ₆	² ₇	³ ₅	4	³ ₅	8	² ₅	1

如图所示，如果 r7c2(6)和 r7c3(7)同时从盘面上消失，那么结局是什么呢？结局就是 r47c23 会形成唯一矩形致命的形式。但是，r7c2(6)和 r7c3(7)又不能同时都填进去。如果同时都填进去，即 r7c2 = 6 的同时 r7c3 = 7，这样会明显导致 r7c5 没有数字可填，这样就违反了数独的填数规则，所以，r7c2(6)和 r7c3(7)，有且仅有一个数是最终正确的。

然后呢？然后看刚才的 r7c5 啦。如果 r7c2(6)和 r7c3(7)只有一个是对的，那不管

是 6 对还是 7 对，都会和 **r7c5** 形成数对结构嘛。有人会说了，数对还能这么用？当然了，同一个区域下，两个格内只有两种不同候选数，不就叫数对嘛。即使 **r7c2(6)** 对还是 **r7c3(7)** 对，始终都是针对于数字 6 和 7 的情况，而恰好，隔壁 **r7c5** 这个单元格就需要“接纳”一个关于 6 和 7 的单元格来构成数对。不管是只有 6 或 7 的其中一个候选数的单元格，还是有 6 和 7 两个候选数的单元格，依然是一种“数对”，但数对结构并不确定，只能确定其中一格，另一格则确定不了到底是 **r7c2** 还是 **r7c3**，所以我称它为“待定的”数对。那么最后的结论呢，就是 **r7c8 <> 7** 了。

这个结构由于带有一个数组（虽然是“待定的”，但是我们也保留了数组的名词，所以这一类嵌套了数组使用的 UR 类型，称为**数组类型**或**类型 3 (UR Type 3)**，或**第三类唯一性测试 (Uniqueness Test 3)**。

实际上这种数组由于最终位置是不固定的，所以这种数组有一个术语称为**待定数组 (Almost Locked Set)**，不过这里我们不作详细的讨论，因为待定数组可以脱离 UR 单独使用，而且会非常精彩，所以我们此处将不作出详细的原理说明，仅仅是 UR 层面够用即可。

接着，我们来看看显性三数组的示例。

6-1-4-2 UR + 显性三数组

7	3	8	5	9	1	6	2	4
6	9	1	8	² ₄	² ₄	3	5	7
2	5	4	3	6	7	8	9	1
3	8	2	⁴ ₇	⁴ ₅	⁴ ₆	⁴ ₉	1	⁵ ₆
¹ ₅	4	9	¹ ₂	3	² ₆	7	8	⁵ ₆
¹ ₅	6	7	¹ ₄	¹ ₅	8	⁴ ₉	3	2
4	2	3	6	8	5	1	7	9
9	7	5	¹ ₄	¹ ₄	3	2	6	8
8	1	6	² ₇	² ₇	9	5	4	3

如图所示，我们的思维依然将 **r46c4** 两格按照绿色和橙色分成两类。分成两类的好处就是方便和简化讨论的情况总数：

- 两格都填绿色的候选数；
- 一格填绿色的候选数，一格填橙色的候选数；
- 两格都填橙色的候选数。

如果都是绿色的候选数，则 UR 直接形成致命形式，所以这条直接行不通。

如果两格都填橙色的候选数，可以观察到，**c4** 里涂色仅橙色的单元格目前就有 4 个了（算上 **r46c4**，因为它们现在只填橙色候选数），而 **r5c4** 和 **r9c4** 也都只有橙色数字，且只有数字 1、2、7。这样一来，**c4** 里存在四个单元格只能放 1、2、7 三种数字的话，必然就有一格放不了任何数（毕竟只能一个 1、一个 2、一个 7）。所以，这一条也会矛盾。

所以，**r46c4** 里只可能一格是绿色的候选数，而另一个单元格填的是橙色的候选数。这样一来，**r59c4** 两格刚好需要一格与之构成三数组结构，多了不行，少了不够。所以现在就好比产生了一个关于 1、2、7 的显性三数组了。所以 **c4** 除开 **r4569c4** 外的其余单元格都不能填入数字 1、2、7，把它们都删除。不过这个示例有点不幸的是，只有一个删数 **r8c4(1)**。

6-1-4-3 UR + 显性四数组

我们再来看一个带显性四数组的示例。

4	2	5		³ 1	¹ 3	8	⁶ 7	⁶ 7
6	3	8	7	2	5	9	4	1
7	1	9	6	8	4	2	3	5
¹			3	⁵ 1	⁵ 2	6	9	4
⁸ 1	⁷ 8		² 6	4	3	¹ 5	² 7	² 7
⁸ 9	⁷ 8	⁹				⁷ 8	² 7	
5	4	² 6				1	² 7	3
2	6	7	1	4	8	3	5	9
3	5	4	2	6	9	7	1	8
⁸ 9	⁸ 9	1	³ 5	⁵ 7	³ 7	4	² 6	² 6

如图所示，依然按照 **r5c12** 填入绿色候选数和橙色候选数来分析。中途的逻辑就不讨论了，因为前面的思路和这里是一样的。

发现 **r5c12** 必须是一个单元格填入绿色的候选数数值，一个单元格填入的是橙色候选数数值后，就保证了恰好有一个橙色的候选数，随后发现 **r5c369** 都只有候选数 1、2、6、7，此时形成关于 1、2、6、7 的四数组（虽然不确定具体 **r5c1** 是橙色的数还是 **r5c2** 是橙色的数，但保证了 **r5c12** 里起码得有一个 1、2、6、7 的数了，这样就能形成数组）。

于是，我们能删除掉 **r5** 里除开 **r5c12369** 外的其余单元格的候选数 1、2、6、7。

接下来我们来介绍三则带隐性数组的示例。

6-1-4-4 UR + 隐性数对

3 5 8	1 2	5 8	9	3 7 5 6	1 2 3 6	2 3 6	1 2 3 6	4
4	1 2	3 8 9	3 6	3 6	1 2 3 6	5	1 2 3 6	3 6
7	6	5 9	4 5	8	1 2 3 4	2 3	1 2 3 9	3 9
2	9	3 6	4 5	3 5 6	3 4 6	1	3 6	3 6
3 6	7	1	2	3 6	9	4 6	3 6	5
3 5 6	5 8	4	7	1	3 6	9	3 6	2
1	3 5	3 5 6	8	4	3 7	2 3 6	2 3 5 6	3 6
9	4	7	3 6	2	5	8	3 4 6	1
3 5 6	4 5 8	2	1	9	3 7	3 4 6	3 4 5 6	3 6

如图所示，和刚才的例子推理逻辑基本一致，只是这里得反过来看。如果把 **r12c6** 里的 1 和 2 看成橙色的候选数，而其它的所有候选数均算作一组的话，那么针对于 **r12c6** 而言，依然是三种情况：

- 两格都是橙色的候选数数值；
- 一格是橙色的候选数数值；
- 没有一格填入橙色的候选数数值。

那么就讨论一下。如果都是橙色的 1 和 2 的话，显然它会和 **r12c2** 两格形成 UR 的致命形式（毕竟都只剩 1 和 2 了），所以这种情况显然是不可能的。

如果没有一格是 1 和 2，显然也不行。这是因为 1 和 2 是两种不同的数，但 **r1** 如果没有了 **r12c2** 的话，就不再有 1 和 2 的合适的位置放下两个数，可以从图上看出来，现在 **c3** 里只剩下 **r3c6** 可以放 1 或 2，而这一个单元格仅能放一个数，所以 1 和 2 里总会有一个数放不下到 **c3** 上，导致这个数字在 **c3** 里无法出现，便产生了矛盾。所以这种情况依然是矛盾的。

所以，还是只得 **r12c6** 里一格是橙色的填数 1 或 2。这样一来，**r3c6** 和 **r12c6** 的其中一格就形成了 1 和 2 的隐性数对，所以 **r3c6** 就不能填入 1 和 2 以外的其它数字；而 **r12c6** 是选取一个单元格填入 1 或 2，而另外一个单元格则是可以填任意情况的，所以这两个单元格是无法确定填数结果的，所以，总的来说，这个例子的删数只有 **r3c6(34)**。

6-1-4-5 UR + 隐性三数组

1 2 5 6	1 2 5 6	1 3 6	3 8	4 7 8	2 9	1 2 3 7	1 3 5 8
1 2 4	8	1 3 4	9	2 3 7	5	1 2 3 4 7	1 3 4
2 4 5	9	7	1	6	2 8	2 3 4	3 4 5 8
4 5 6 7	4 5 6 7	9	3 4 8	3 5 7	1	3 4 6	3 4 5 6 8
3	1 2 4 5	1 6	4	2 5	2 8 9	4 5 8	7
2 4 5 7	2 4 5 7	8	6	2 3 5 7	2 9	1	3 4 5
8	1 6	1 6	7	9	4	3	5 2
4 7	3	5	2	1	6	4 7	8 9
9	1 4 7	6	2	5	8	3	4 6

如图所示，和刚才的逻辑一样，我们依然讨论 r5c23 的填数情况。中间的逻辑就不推理了，这里给出结论：r5c23 里必须有一个单元格填入的是 1 或 6，而 r5 只剩下两个单元格，可以放 6 和 9 了。这样一来，r5c2368 里必然只存在 3 个单元格能放下 1、6、9，而且是恰好放满，不多不少。所以这三个单元格里就不再可能填 1、6、9 以外的其它候选数了。不过 r5c23 是确定不了填数 1 和 6 的，所以我们只能保证 r5c68 是放 1、6 或者 9 的，所以这个题目能够删除的数字只产生于 r5c68 里，而且删数必须是非 1、6、9 的其余候选数。

我们来看最后一个例子。

6-1-4-6 UR + 隐性四数组

1	9	3	5	2	4	8	7	6
8	5	2 4	3	7	6	9	1 2 4	1
2 4 6	2 4 6	7	1 8	1 9		1 3 4	1 2 5 6	1 3 4 5 6
4 6 7 9	4 6 7 8 9	4 8 9	7 8	5	1	2	3	4 7 9
2 7 9	2 7 8	1	7 8	4	3	6	5 7 8	5 7 9
2 4 7	3	5	9	6	2 8	1 4 7	1 7 8	1 4 7
2 3 4 7	1 2 7 8	2 8 9	1 2	1 3 9	2 9	5	1 7	1 3 6
5	1 2 7	2 9	6	1 3 9	7	1 3 7	4	8
3 7	1	6	4	8	5	1 3 7	9	2

如图所示，这是最后一个例子，也是最复杂的一个例子。我们来看看这个例子到底有何

神奇的地方。

首先还是按照 **r7c89** 来讨论填数情况，并最终得到 **r7c89** 里必须有一个单元格放 6 或者 7，于是，此时发现 **r7** 里恰好有四个单元格能放下 4、6、7、8。所以 **r7c89** 的其中一格和 **r7c123** 一并形成 4、6、7、8 的隐性四数组结构，而 **r7c123** 才是能确定填数的情况的，所以删数将从 **r7c123** 里产生：删除非 4、6、7、8 的候选数。

6-1-4-7 数组的显隐性是互补的，那么 UR 里的数组呢？

既然数组的显隐性互补，那么 UR 里的数组是否也满足互补的要求呢？实际上，是的，我们随意找出一则示例就可以看出来。

例如上述的隐性四数组的例子，我们看互补的情况，这个时候应当看的是剩余的单元格 **r7c456**，不过和显隐性互补不同的是，此时 **r7c89** 依然要看。

在互补后，实际上 **r7** 形成的是 1、2、3、9 的显性四数组，这也是合适的，毕竟 **r7c7** 是确定值被占据了一个单元格。

6-1-5 共轭对类型（UR Type 4）

我们再来看最后一则 UR 的类型。思路看起来简单，但有点难理解。

6-1-5-1 一个示例

5	1	7	² ₆	8	3	4	9	² ₆
9	6	⁴ ₃	² ₇	1	⁴ ₇	8	² ₃	5
⁴ ₃	8	2	9	5	⁴ ₆	7	³ ₆	1
² ₇	9	6	1	² ₇	5	3	4	8
⁷ ₂	² ₃	⁵ ₇	⁵ ₃	4	⁶ ₉	8	⁶ ₉	1
1	4	8	3	⁷ ₆	9	2	5	⁷ ₆
6	3	1	8	9	2	5	7	4
⁴ ₇	⁵ ₇	⁴ ₅	⁷ ₆	3	1	⁶ ₉	² ₈	² ₆
⁷ ₈	2	9	5	4	⁷ ₆	1	⁶ ₈	3

如图所示。可以发现，这个结构里，**r4c15** 是 2、7 的显性数对，而且，这个例子还有一个现象，虽然不是很能引起注意，不过依然很重要的一个推导条件：**r5** 上数字 2 只有 **r5c15** 两处。这一点很重要，为什么呢？

试想一下，**r4c15** 又只能放 2 和 7，而 **r5c15** 里其中一个单元格 2，则另外一格就不允许填 7。不然的话，四格就只有 2 和 7 了，这便形成了致命形式。

6-1-5-2 别急，结论还不知道，我们来看看问题

或许你会问，既然我们都填好了数字，就像图上这样，那凭什么能说明它依然形成致命形式呢？毕竟数字填好了就动不了了，怎么还产生两种填法呢？那么我们来看看。

我们假设最终是 r5c1 是填 2 的，要是 r5c5 是 7 的话，就会产生如下的填数格式：

5	1	7	² ₆	8	3	4	9	² ₆
9	6	³ ₄	² ₇	1	⁴ ₇	8	² ₃	5
³ ₄	8	2	9	5	⁴ ₆	7	³ ₆	1
7	9	6	1	2	5	3	4	8
2	⁵ ₅	³ ₅	4	7	8	⁶ ₉	1	⁶ ₉
1	4	8	3	⁶ ₇	9	2	5	⁶ ₇
6	3	1	8	9	2	5	7	4
⁴ ₈	⁵ ₇	⁴ ₅	⁶ ₇	3	1	⁶ ₉	² ₈	² ₆ ₉
⁸ ₈	2	9	5	4	⁶ ₇	1	⁶ ₈	3

不过，看似填好了，但客观来说，它确实存在另外一种与之匹配的填数格式：

5	1	7	² ₆	8	3	4	9	² ₆
9	6	³ ₄	² ₇	1	⁴ ₇	8	² ₃	5
³ ₄	8	2	9	5	⁴ ₆	7	³ ₆	1
2	9	6	1	7	5	3	4	8
7	⁵ ₅	³ ₅	4	2	8	⁶ ₉	1	⁶ ₉
1	4	8	3	⁶ ₇	9	2	5	⁶ ₇
6	3	1	8	9	2	5	7	4
⁴ ₈	⁵ ₇	⁴ ₅	⁶ ₇	3	1	⁶ ₉	² ₈	² ₆ ₉
⁸ ₈	2	9	5	4	⁶ ₇	1	⁶ ₈	3

虽说填好了，但依然可以产生“置换”，原因很简单：因为它们是填入数，而非提示数，既然不是提示数，数字就能发生变换，毕竟它不固定。这样也算有两种填法。

实际上，官方的解释方式是这样的：即使已经填好一种填法后，因为填数是行、列、宫内成为数对形式的，所以实际上就一定客观存在能与之交换的另外一种填法，这种客观存在的填法不受候选数是否存在的约束。或者换句话说，UR 形成两种填数的情况，并不受候选数的约束，只要四格的填数构成 a、b、a、b 的形式的填法，就必然会产生 a 和 b 置换的另外一种填法，哪怕其中一格不存在其候选数 a 或者 b，防止其置换。

所以，置换是客观存在的，并不受到候选数情况的约束。因此，致命形式依然是能够形成的。

6-1-5-3 那么，结论呢？

还记得推理的结果吗？r5c15 里，因为有一个是 2，所以另外一格就不允许是 7，否则相当于 r5c15 形成了 2 和 7 的数对结构，而上面也是 2 和 7 的数对，将会客观存在两种填法，导致出现致命形式。所以，r5c15 的任意一个候选数 7 都是不允许存在的，也因此，r5c15 两处的 7 都应该被删除。

这个就是我们所谓的**共轭对类型**或**类型 4**了（UR Type 4），也称为**第四类唯一性测试**（Uniqueness Test 4）。

6-1-5-4 有一些想说的

这种删除方式更加新奇。它利用了 2 的填数情况只能位于结构内部的特性，得到了致命形式，从而排除了这么一个假设。这里要介绍一个术语：**共轭对**（Conjugate Pair）。共轭对指的就是这里的这个 r5 的这两数字 2。

共轭对的定义是：在同一区域下，只有两处可填某一个候选数的单元格下的这两个相同的候选数；或是同一个只有两个候选数的单元格下的这两个不同的候选数。

解释起来有点复杂，说白了就是：因为 r5 上只有两处可填 2 的位置 r5c15，所以 r5c1(2)和 r5c5(2)就是共轭对。

共轭对的特征是，两个数有且仅有一个正确。这一点希望你记住，不止是这一节才会用到，后面链以及更靠后的地方，也会使用到这一个特征。

那么，我们再来看另外的使用共轭对的示例类型。

6-1-5-5 二链列类型

1	6	1	2	8	9	3	7	6	5	4
4	5	8	7	6	1	2	3	9		
	6		3	3	4	2	5	8	7	6
8		2	3	4	5	1	6	4	2	7
7	6	5	2	4	9	1	8	3		
1	1	2	4	3	8	7	4	6	2	6
5	4	6	9	7	2	3	1	8		
3	8	1	6	5	4				7	9
2	9	7	1	3	8	5	4	6		

如图所示，可以从示例里看到，所有 4 的填数位置仅出现于 r46c37 四格里。那么这样的话，4 的位置只可能有如下两种填法：

- r4c3 和 r6c7 同为 4；
- r6c3 和 r4c7 同为 4。

这是不是很像是一个二链列结构呢，因为二链列的最终填法也是交叉的两种情况？我们接着来看。我们发现，结构的左下角和右上角（ $r6c3$ 和 $r4c7$ ）两格，都只有 4 和 9 两个候选数。如果左上角和右下角（ $r4c3$ 和 $r6c7$ ）都同时是 4 的话，这必然会使得同为 4 和 9 两个候选数的两个只可以填入 9。这样，就形成了关于 4 和 9 的致命形式。虽然已经填入，但因为终归是自己填入的数字，所以依然可以产生 4 和 9 的交换写法，这样就跟四格都是 4 和 9 两个候选数的唯一矩形致命形式没有本质上的区别了。所以，为了避免致命形式出现，4 的位置只可能是另外一种情况： $r6c3$ 和 $r4c7$ 同为 4。

这个结构因为运用了二链列的思路，所以被称为二链列的类型；而我们在使用二链列的思路的时候，原理依然是借助了“共轭对”的方式的：如果 $r4c3 = 4$ 时，则 $r6c3 \neq 4$ ，而 $r6$ 有 4 的共轭对，所以 $r6c7 = 4$ 。反过来也是一样的：如果 $r6c7 = 4$ 时，则 $r4c7 \neq 4$ ，而 $r4$ 有 4 的共轭对，所以 $r4c3 = 4$ 。所以，它依然归到共轭对里。

这里简单提一下的是，有一些数独分析软件把这种类型归到**类型 6 (UR Type 6)** 或者**第六类唯一性测试 (Uniqueness Test 6)**。这一点不受影响，你知道和了解即可。

6-1-5-6 隐性唯一矩形 (Hidden Unique Rectangle)

最后来介绍一种利用共轭对的新型技巧，这个技巧还有一个单独的技巧名：**隐性唯一矩形 (Hidden Unique Rectangle)**。

8	¹	¹ 5	3	6	9	¹ 5	4	2
2	4	⁷ 6	8	5	1	9	3	⁶
9	3	¹ 5 6	4	2	7	¹ 5	¹ 5 6	8
¹ 6	¹ 9	2	5	4	3	8	7	⁶ 9
5	8	3	9	7	6	4	2	1
⁷ 6	⁷ 9	4	2	1	8	3	⁶ 9	5
¹ 7	5	¹ 7	6	9	4	2	8	3
4	2	8	¹ 7	3	5	6	¹ 9 7 9	
3	6	9	¹ 7	8	2	¹ 5 7	¹ 5	4

首先我们观察到， $r1$ 和 $c3$ 上，都有 5 的共轭对，而且共轭对涉及的单元格刚好是这个结构涉及的其中三个单元格。那这有什么用处呢？

试想一下，5 的共轭对涉及 $r1c3$ 、 $r1c7$ 和 $r3c3$ 这三个单元格，那么就这三格而言，填 5 只可能有以下两种可能：

- $r3r3 = r1c7 = 5$;
- $r1c3 = 5$ 。

如果 $r3r3$ 和 $r1c7$ 同为 5 的话，可以确定的是， $r3c7$ 只能填 1（因为 $r3c7$ 只有 1 和 5 两个候选数）。那么， $r1c3$ 就不能填 1 了。因为 $r1c3$ 如果此时还填 1 的话， $r3c7$ 也是 1，然后剩余两格又是 5，这就构成了 1 和 5 的致命形式。之前说过，虽然已经填入了四

格，看似已经不能发生变动，但是终归是自己填的，而不是提示数，这样就可以交换四格的填数，并产生客观的另外一种填法，就形成了类似于标准类型致命形式。所以此时 **r1c3** 不应为 1。

第二种情况则是 **r1c3** 为 5。那很明显，**r1c3** 都是 5 了，那肯定这格就不能填 1 了。

所以，所有两种情况都能使得 **r1c3** 不可以填 1，所以 **r1c3** \neq 1，所以可以安全地删除掉这个候选数。

这个结构运用到的是行和列两个“维度”上的同数的共轭对，所以依然是共轭对的类型。而至于分类呢，有些分析软件将其归为**类型 7 (UR Type 7)**。但是有趣的是，它有个单独的技巧名——隐性唯一矩形，说明这个唯一矩形的位置是隐藏在里面的，不容易看到。

6-1-6 其它唯一矩形类型 (UR Other Type)

下面给大家陈列一则不属于前面任何一种类型的示例，但和前面的推理思路有很深的关系。

9	1	² 4 5 7	6	² 4 5 7	² 4 5	3	² 4	8
^{2 3} 4 5 6	^{2 3} 4 5	² 4 5 6	³ 4 5	1	8	⁴ 9	7	² 4 9
^{2 3} 4	8	² 4 7	³ 4 7 9	² 4 7 9	^{2 3} 4	1	6	5
^{1 2} 4 5 6	² 4 5	3	^{4 5} 7	^{4 5 6} 7	9	^{4 5} 7	8	^{1 2} 4
^{1 2} 4 5	7	² 4 5 9	8	3	¹ 4 5	6	² 4 5	^{1 2} 4 9
8	^{4 5}	^{4 5 6} 9	2	^{4 5 6} 7	¹ 4 5 6	^{4 5} 7 9	3	¹ 4 9
³ 4 5	9	8	³ 4 5	^{4 5 6} 4 5 6	³ 4 5 6	2	1	7
² 4 5	6	1	^{4 5} 9	² 4 5 8 9	7	^{4 5} 8	^{4 5}	3
7	^{2 3} 4 5	² 4 5	1	² 4 5 8	^{2 3} 4 5	^{4 5} 8	9	6

如图所示，这个结构看起来跟 UR 都没有什么特别大的关系了，不过它依旧能够形成删数，删数就是图中给出的红色的候选数。那么它是如何被删除的呢？我们下面分成两个图为大家呈现出来。

9	1	² _{4 5}	6	² _{4 5}	² _{4 5}	3	² ₄	8	9	1	² _{4 5}	6	² _{4 5}	² _{4 5}	3	² ₄	8
^{2 3} _{4 5 6}	^{2 3} _{4 5}	² _{4 5 6}	³ _{4 5}	1	8	⁴ ₉	7	² _{4 9}	^{2 3} _{4 5 6}	^{2 3} _{4 5}	² _{4 5 6}	³ _{4 5}	1	8	⁴ ₉	7	² _{4 9}
⁴ _{2 3}	8	² _{4 7}	⁴ _{7 9}	³ _{7 9}	² ₄	1	6	5	⁴ _{2 3}	8	² _{4 7}	⁴ _{7 9}	³ _{7 9}	² ₄	1	6	5
^{1 2} _{4 5 6}	² _{4 5}	3	^{4 5} ₇	^{4 5 6} ₇	9	^{4 5} ₇	8	^{1 2} ₄	^{1 2} _{4 5 6}	² _{4 5}	3	^{4 5} ₇	^{4 5 6} ₇	9	^{4 5} ₇	8	^{1 2} ₄
^{1 2} _{4 5}	7	² _{4 5}	8	3	¹ _{4 5}	6	² ₅	^{1 2} _{4 9}	^{1 2} _{4 5}	7	² _{4 5}	8	3	¹ _{4 5}	6	² ₄	^{1 2} _{4 9}
8	^{4 5} ₉	^{4 5 6} ₉	2	^{4 5 6} ₇	¹ _{4 5 6}	^{4 5} _{7 9}	3	¹ _{4 9}	8	^{4 5} ₉	^{4 5 6} ₉	2	^{4 5 6} ₇	¹ _{4 5 6}	^{4 5} _{7 9}	3	¹ _{4 9}
^{4 5} ₃	9	8	^{4 5} ₃	^{4 5 6} _{4 5 6}	³ _{4 5 6}	2	1	7	^{4 5} ₃	9	8	^{4 5} ₃	^{4 5 6} _{4 5 6}	³ _{4 5 6}	2	1	7
² ₅	6	1	⁵ ₉	² _{5 8 9}	7	⁵ ₈	4	3	⁴ ₂	6	1	⁴ ₉	² _{5 8 9}	7	⁴ ₈	5	3
7	^{2 3} _{4 5}	² _{4 5}	1	⁴ _{5 8}	^{2 3} _{4 5}	⁵ ₈	9	6	7	^{2 3} _{4 5}	² _{4 5}	1	⁴ _{5 8}	^{2 3} _{4 5}	⁴ ₈	9	6

左图给出的是示例里 $r8c8 = 4$ 时的情况，而右图则给出的是 $r8c8 = 5$ 的情况。仔细对比两个示例，就可以发现，它实际运用到的是共轭对类型的基本推导思路。

先看左图，当 $r8c8 = 4$ 的时候， $r89c7$ 变为 5、8 的显性数对结构，而由于 $r89c5$ 存在 $c5$ 上关于 8 的共轭对，所以 $r89c5$ 里必须要填入一个 8，那么由于刚才的填数模式，就可以发现， $r89c5$ 里就不能再放入 5 了，否则 $r89c57$ 将会构成关于 5、8 的唯一矩形的致命形式，所以此时应当删除掉的是 $r89c5(5)$ 。当然，由于 $r8c8 = 4$ 的关系，此时 $r8c5$ 根据排除的规则，肯定也可以去掉 $r8c5(4)$ 。

再看右图。当 $r8c8 = 5$ 的时候， $r89c7$ 变为 4、8 的显性数对结构，而由于 $r89c5$ 此时还是存在关于 8 的共轭对，所以 $r89c5$ 里必须有一个 8 的出现，而另外一个则不能是 4，否则 $r89c57$ 将会形成关于 4、8 的唯一矩形的致命形式，所以此时应当删除 $r89c5(4)$ ；当然了，由于排除的关系， $r8c5$ 此时也不能填入 5。

所以，可以对比两种情况就可以发现，结论可以删除的数字里， $r8c5(45)$ 是同时都可以被去掉的，所以 $r8c5 \neq 45$ 是这个推导过程的结论。

这个示例非常有趣的地方在于，它在推导过程里完美诠释了共轭对类型的唯一矩形结构的推理过程，但它又不同于这个类型，因为结构并不是这样简单清晰的样子。

6-1-7 结构简图与小测试

接下来，我们把该文档里的所有提到的构型的结构简图全部给出来，如果你想学习如何观察，那么第一步就是掌握它们的构型。

[illegible]

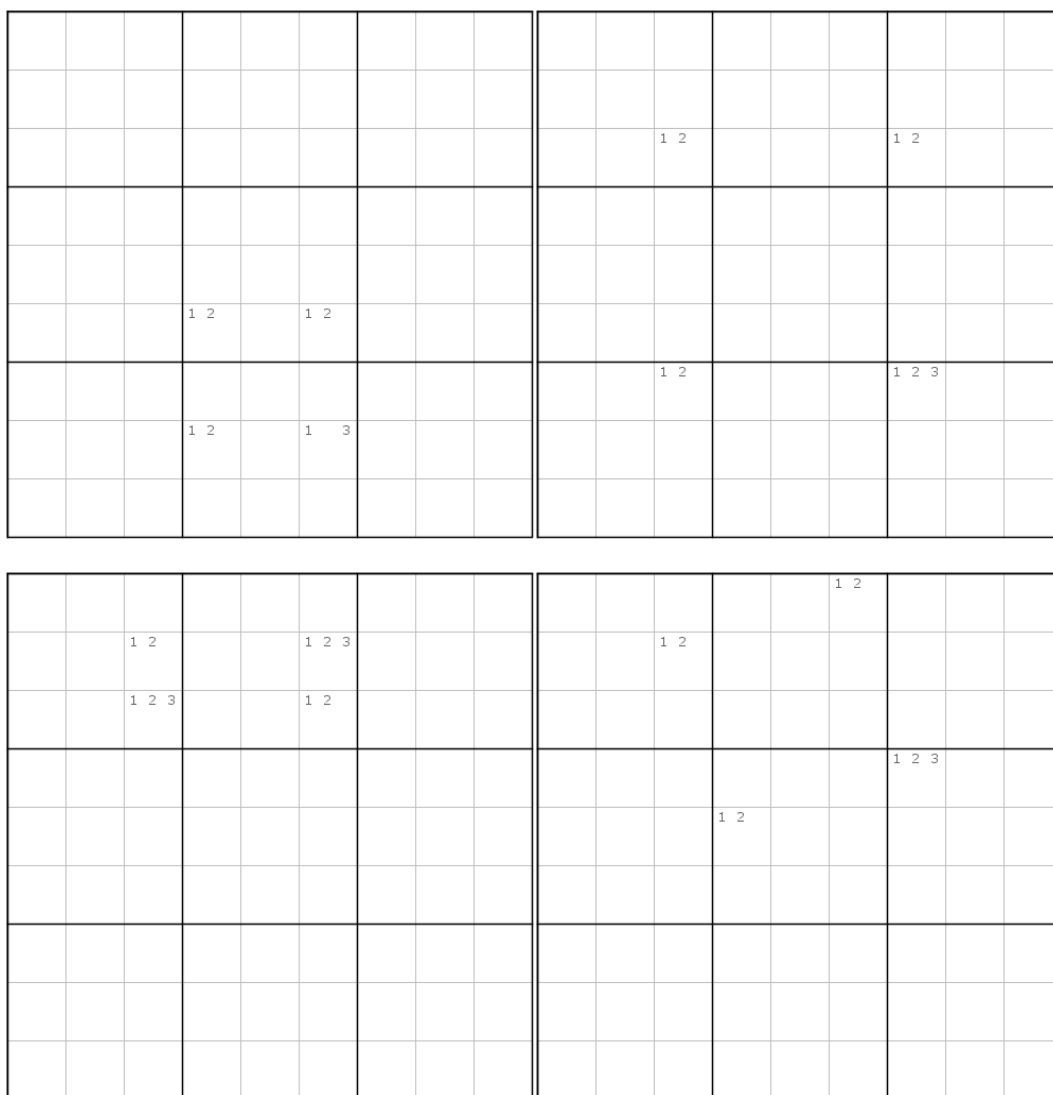
			1 2 3		1 2 3			
			4 5 6		4 5 6			
			7 8 9		7 8 9			
			1 2 3		1 2			
			4 5 6					
			7 8 9					

列出的 7 个简图分别对应类型 1 到 7（标准类型、区块类型、数组类型、共轭对类型、区块类型的推广、二链列类型和隐性唯一矩形）。

注：紫色单元格表示结构涉及的单元格，红色候选数是删数，黄色候选数是结构涉及的额外候选数，绿色候选数是共轭对（结合区域来看，例如类型 7 里，有 3 个 1 被涂绿色，则理解为 r4 和 c4 有 1 的共轭对）。

那么，最后我们来留下一些模型。请仔细判断这些构型到底哪些构型是符合 UR 的构型模式和推理逻辑的（即可以使用 UR 的技巧），而哪些构型不能使用 UR。

			1 2		1 2						1 2			
											1 2			
			1 2								1 2			
					1 2 3						1 2 3			



那么，所有基本的唯一矩形构型（被分类好的）已经全部讲完了。我们不希望大家将分类搞得如此细致（类型 5、6、7），不过我建议大家学习，是为了掌握各种不同的推理思路，达到应有的效果，学习到真正的东西，所以入门时，我依然按照传统的方式，即描述基本构型的方式来讲解 UR 技巧。那么，UR 就讲完了，接下来来看一些以 UR 作为基准进行规格和推导模式推广的技巧。

最后我列出两个例题，提供大家练习，第一题需要使用 UR 的标准类型、区块类型和数组类型；而第二个题需要用到 UR 的标准类型、数组类型和共轭对类型。

	1					4		6	5	3							
4			6		2				2	7			8	3			
		8				2						5	9				
	6		5		4			1			1	6				5	
				6		5				2	4				7	9	
	8		9		7					5				7	1		
2		6		7		9	8						7	8			
						3	7					3	6			1	7
3			4													4	8

题目来自于 SE 软件教学贴的网页（不过现在已经打不开了）。

6-2 残缺唯一矩形 (Incompleted UR)

残缺 UR 是 UR 的一种变种，也是我们需要掌握和理解的一种特殊 UR 形式，它和 UR 一样，依然分为那些类型，只是……

6-2-1 标准类型 (Incompleted UR Type 1)

7	2	6	2	1	4	5	9	3
3	9	4 5	2 5 6 7 8	2 5 7 8	2 5 7 8	1	6 4 7	2
1	2	4 5	5 6 7 8	3	9	4	6 4 7	2
5 9	3	2	5 4 5 7 8	1 5 7 8	6	4 5 7	1	9
5 6	1	8	2 4 5 7	2 5 6 7	3	4 5 7	2	4 5
6 9	4	7	2 5	1 6	8	2 5	1	9
4	5 6	9	2 5 7 8	2 5 7 8	7	3	8	6
8	7	1	4 2	6	3	9	2 4 5	2 4 5
2	5 6	3	4 5 7 8	9	5 7 8	4 7	1	6 8

如图所示。实际上，这个问题变为了这样：如果我们不小心删掉 r3c4(2)之后才开始小心翼翼地使用 UR，那么 UR 还可以使用吗？

唯一矩形的要求是“产生两种互换的填法，但不影响除了可影响到的候选数 2 和 8 外的剩余盘面”。这句话看似相当绕口，不过你如果理解了我之前讲过的唯一矩形的原理，这一点我觉得你是可以理解的。

我们回顾一下共轭对类型里的示例。

5	1	7	² ₆	8	3	4	9	² ₆
9	6	⁴ ₃	² ₇	1	⁴ ₇	8	² ₃	5
⁴ ₃	8	2	9	5	⁴ ₆	7	³ ₆	1
² ₇	9	6	1	² ₇	5	3	4	8
² ₃	⁵ ₇	⁵ ₃	4	² ₆	8	⁶ ₉	1	⁶ ₉
1	4	8	3	⁶ ₇	9	2	5	⁶ ₇
6	3	1	8	9	2	5	7	4
⁴ ₇	⁵ ₈	⁴ ₅	⁶ ₇	3	1	⁶ ₉	² ₈	² ₆
⁷ ₈	2	9	5	4	⁶ ₇	1	⁶ ₈	3

这是之前的例子。我们可以看到，我们固定好 $r5c1(2)$ 后，才能得到删数 $r5c5(7)$ ；反之可以删除 $r5c1(7)$ 。可是，这两种情况都是当 $r5c1(2)$ 或者 $r5c5(2)$ 成立时才能得到的删数，显然不足以作为整体的删数。我坚信总有小伙伴会钻这个牛角尖。

现在我们来尝试解开这个疑问。如果我们就真的固定了 $r5c1(2)$ ，而且还没有了 $r5c1(7)$ ，此时的局面下 $r5c5(7)$ 能否被删除。答案是可行的，因为即使我们只有一种填法，并假设 $r5c5 = 7$ 时，就会产生如左图所示的情形。

5	1	7	² ₆	8	3	4	9	² ₆
9	6	⁴ ₃	² ₇	1	⁴ ₇	8	² ₃	5
⁴ ₃	8	2	9	5	⁴ ₆	7	³ ₆	1
7	9	6	1	2	5	3	4	8
2	⁵ ₇	⁵ ₃	4	7	8	⁶ ₉	1	⁶ ₉
1	4	8	3	⁶ ₇	9	2	5	⁶ ₇
6	3	1	8	9	2	5	7	4
⁴ ₈	⁵ ₇	⁴ ₅	⁶ ₇	3	1	⁶ ₉	² ₈	² ₆
⁸ ₇	2	9	5	4	⁶ ₇	1	⁶ ₈	3

看似填好了，但实际上就 $r4c15$ 四个单元格而言，这个结构是不稳定的。因为四个单元格所在的行、列、宫其实都是一个 2 和 7 的显性数对。而由于这个结构的存在，导致 2 和 7 在内部是可以产生两种互相交换的填数形式的。

就算客观存在的第二种填法里， $r5c5$ 并不包含候选数 2 的情况，但并不会造成任何形成致命形式的影响。形成致命形式是说，结构内部的其中至少两种填法会导致剩余盘面的形式完全一致，而客观存在的 2 和 7 交换的填法确实做到了这一点（可以对照两个图，可以

发现，其余的候选数、提示数、填入数全部是一样的）。所以，它形成了致命形式是毋庸置疑的。

所以，即使没有那种填数情况（候选数根本不足以形成完整的 UR 结构），照样是可以使用的。

所以，最开始给出的结论，虽说少了一个删数情况，但依旧是可以删除的。这种结构由于缺失了一个候选数，所以称为**残缺唯一矩形**（**Incompleted UR**）。残缺 UR 的结构并不影响 UR 的使用。

6-2-2 区块类型（Incompleted UR Type 2）

4 6	3	2	7	1	4 5	6 9	6 8 9	5 6 8 9
1 5			6	2	3	7	1 5	4
1 5 6	4 7	6 5 6	8	4 5	9	3	1 6	2
4 6 8 9	2	4 6 8 9	5	3	4 7 8	1	6 7 9	6 7 9
7	4 5 9	1	2	4 9	6	8	5 9	3
3	6 8 9	5 6 8 9	1	7 8 9	7 8	4	2	5 6 7 9
2	5 6 7 8	3	4	5 6 7 8	1	6 9	6 7 8 9	6 7 8 9
4 5 6 8	1	4 5 6 7 8	9	5 6 7 8	5 7 8	2	3	6 7 8
6 8 9	6 7 8 9	6 7 8 9	3	6 7 8	2	5	4	1

如图所示，如果 r3c1(6)和 r3c8(6)同时消失的话，r23c18 至少存在一种填法导致形成 UR 致命形式，所以 r3c1(6)和 r3c8(6)至少有一个是正确的数字。

不管谁正确，都可以删掉它们的交集（交集上存在一个 6 都会同时使得两个 6 全部消失，从而出错），所以 r3c23 <> 6。

我们接着再来看一个“类型 5”的例子。

1	⁵ 7	⁵ 7 8	4	^{5 6} 8 9	^{5 6} 8	9	3	2
^{8 9}	2	6	^{8 9}	7	3	1	4	5
4	3	⁵ 9	⁵ 9	2	1	6	7	8
⁵ 9	4	3	2	1	⁵ 9	7	8	6
7	¹ 8 9	2	6	^{8 9}	4	3	5	¹ 9
⁵ 8 9	6	¹ 8 9	3	⁵ 8 9	7	4	2	¹ 9
6	⁵ 7 8	⁵ 7 8	1	4	⁵ 8	2	9	3
¹ 5 8 9	¹ 5 8 9	⁵ 8 9	7	3	⁶ 9	⁵ 8	¹ 6	4
3	¹ 9	4	⁵ 8	⁶ 9	2	⁵ 8	¹ 6	7

r1c3(8)和 r7c23(8)这三个 8 不可同时消失，否则将会出现关于 5 和 7 的 UR 致命形式，所以删掉它们三个 8 的交集，于是自然 r8c3 \neq 8。

6-2-3 数组类型（Incompleted UR Type 3）

² 6 9	⁴ 9	5	³ 7	³ 4	³ 6	³ 7	1	8
3	7	1	9	² 5	8	² 5	^{4 6} 4 6	^{4 6} 4 6
8	^{4 6} 2 6	² 6	1	² 4 5 6	² 5 6	² 5	³ 9	³ 9
⁶ 9	⁶ 9	4	5	8	1	3	7	2
5	1	8	^{2 3} 7	^{2 3} 7	^{2 3} 6	⁴ 9	⁴ 9	⁴ 9
7	2	3	4	⁶ 9	⁶ 9	1	8	5
4	⁵ 7	³ 6	8	^{2 3} 5	^{2 3} 5	9	^{2 3} 5 6	1
1	8	² 7	6	³ 5 9	^{2 3} 5 9	4	^{2 3} 5	³ 7
² 6	³ 5	9	^{2 3} 7	1	4	8	^{2 3} 5 6	³ 6

按照原定数组类型的逻辑，我们可以直接得到 r1c1(2)和 r1c2(4)有且仅有一个是正确的，从而会和 r1c457 三格构成 2、3、4、7 的显性四数组，删除其余单元格下的候选数 2、3、4、7。所以 r1c6 \neq 3。

6-2-4 共轭对类型 (Incompleted UR Type 4)

		4	4	5	2	1	3		6
8 9	7		8 9				7 9		
6		3		7	4	3	5	2	8
	9				9				
2		3	5	8	6		4		1
	7					9	7 9		
7	2		3		1	5	8	6	4
		9		9					
1	8	6	2	4		4		5	3
				9	7	9	7 9		
5	4	4	3	6		3		1	2
	9			8 9	7 8	7 9			
4	5	2		3	3	6	1	3	7
			9	8 9				8	
3	6		1	7		3	2	4	5
8 9		8 9			8				
3	1	7	4	5	2	6		3	9
8							8		

c6 有 7 的共轭对出现在结构的 r56c6 之中，而旁边 r56c7 是只能填 7 和 9 的，而 r56c6 又有一格是 7，那另外一格一定不为 9，所以删掉这里的 r5c6(9)。

再来看一下二链列类型（“类型 6”）和隐性唯一矩形（“类型 7”）。

4	9	3	1	6	8	5	7	2
5	2	8		3		3		
	4	7		4		4	7	
1	6	7	9	2	5		3	
						8	4	4
3	5	1		7	2	6	9	
	4	8						8
6	8	9	5		1	3	2	1
	4				4			4
7	4	2	6		1		1	3
			8 9		9			8
8		1	3		3	1	3	1
	7			4		4		4
		7		9	7	9		7
9		1	3		1	3		1
	7			8			7 8	
2		1			1			1
	7			4	4			4
		7	8		7	8		7 8
				5			3	9

老实说，唯一一个不好看的残缺版本就是它了。

首先我们发现，填入 3 的位置，在 r3 和 r6 之中，只有 r3c7 和 r6c9 四格，恰好都在结构内。然后这样四格必然是一个关于 3 的二链列，那么填数最终情况一定是对角两格填 3。

- 如果 r3c7 和 r6c9 填 3，那看起来没毛病；
- 如果 r3c9 和 r6c7 填 3，就有毛病了：这样填的话，会导致 r3c7 和 r6c9 本来只有 3 和 8 的两格都只能填 8，这样就形成了 3 和 8 的 UR 致命形式。所以这种情况不成立。

所以两种情况下，只有 r3c7 和 r6c9 是 3 才是正确的做法，所以 r3c7 和 r6c9 都为 3，自然 r3c9 和 r6c7 就不是 3 了。

残缺到这个样子居然还是 UR。我们只要让我们思路下的 UR 结构能够形成致命形式的写法就可以了。

2 3 6 9	2 3 5 6	2 5 6 9	5 6 9	4	1	8	3 9	7
6 8	7	4	6 8 9	2 3 9	2 3	5	1	2
3 8 9	1	5 9	5 8 9	7	2 5 9	4	6	2 3 9
5	4	3	7	2 9	2 9	6	8	1
1	9	8	3	6	4	2	7	5
2 7	2 6	2 6 7	1	5	8	3 9	4	3 9
4	2 3 5	1	5 9	8	7	3 9	2 3 5 9	6
3 6 9	8	5 6 9	2	1	3 5 6 9	7	3 5 9	4
2 3 6 7	2 3 5 6	2 5 6 7	4	3 9	3 5 6 9	1	2 5 9	8

这是最后一个类型，也称**残缺隐性唯一矩形**（Incompleted Hidden UR）。

- 如果 r3c9 = 2，自然 r3c9 <> 3；
- 如果 r2c9 和 r3c6 都为 2，那么 r2c6 = 3，此时为了防止 UR 致命形式的出现，r3c9 <> 3。

所以 r3c9 一定不为 3。

6-3 死锁唯一矩形（Locked UR）

死锁 UR 基于二链列类型的 UR，不过逻辑就不比之前的简单和死板了。

6-3-1 标准类型 (Locked UR Type 1)

7	4	8	3	5	9	1	2	6
³ _{5 9}	⁵ ₉	1	7	2	6	8	4	³ ₉
³ ₉	2	6	4	¹ ₈	¹ ₈	7	³ ₉	5
2	^{5 6} ₉	³ ₉	¹ _{6 9}	4	¹ ₅	³ _{5 9}	8	7
^{5 6} ₉	7	4	⁶ _{8 9}	3	² _{5 8}	² _{5 9}	¹ _{6 9}	¹ ₉
1	8	³ ₉	⁶ ₉	7	² _{5 9}	² _{5 9}	³ _{6 9}	4
4	³ ₉	2	¹ ₅	¹ ₉	7	6	¹ _{5 3}	8
⁶ ₉	1	7	⁵ _{8 9}	⁶ _{8 9}	3	4	⁵ ₉	2
8	³ _{6 9}	5	2	¹ _{6 9}	4	³ ₉	7	¹ ₃

如图所示。我们观察，2 在全盘只有 r56c67 这四格可填，而候选数 5 的位置关于 r56 来说，也只有 5 个位置。要是 r5c1 \neq 5，5 在 r56 内填数的位置也只有 r56c67 那几格了。试想一下，2 和 5 全被挤在 r56c67 里，会怎么样？全挤进去意味着 2 和 5 都是二链列结构。意味着两个数都是对角两格是一样的数值。那其中一个数是撇对角两格，那剩下另一个数就应该是捺对角两格，这样不就构成 2 和 5 的 UR 致命形式了嘛！所以唯一能防止这样结构出现毛病的，只能让 r5c1 = 5。

所谓的撇对角和捺对角指的是按笔画的方向定义的对角，即撇对角指的是右上和左下两个单元格；而捺对角则指的是左上和右下连个单元格。

这种结构是为了规避 2 和 5 卡死在四格之中形成二链列形式的 UR 结构，它被称为**死锁唯一矩形 (Locked UR)**，而它是标准类型（即类型 1）。

6-3-2 区块类型 (Locked UR Type 2)

1 8	2 3 6	1 2 3 8	2 4 5	6 4 5	4 5 8	4 7	3 4	3 7	4 9
6 8	2 3 6	7	2 4	2 4	9	5	4		1
9	5	4	3	7	1	8	2 6	2 6	
3	4 6	1 9	4 6 9	1 4	7	2	5	8	
1 4 6	2 4 9	1 2 9	5 8	3 5	3 8	1 4 7	1 4 7	4 6 9	
7	8	5	2 9	1 2 9	4 6	1 4	1 4	6 9	3
4 5 8	4 9	3 9	1 4 5	8	2	6	1 4	3	7
2	1	6	7	5 4	3 4	9	8	4 5	
4 5 8	7	3 8	1 4 5 6	9	4 5 6	1 3	1 2 3 4	2 5	

如图所示，这个结构和刚才的差不多，不过这个结构多了些情况。

如果 $r2c8(4)$ 、 $r6c78(4)$ 、 $r79c8(4)$ 都为假时，4 在 $c78$ 的位置就卡死在结构涉及的四格 $r15c78$ 之中了，而本来 7 就只有这四格位置可填，那 4 和 7 都卡死在里面，必然会致命。所以这些数不能同时消失。换句话说，至少一个是正确的数字，所以便可删掉交集，那么 $r5c8 \neq 4$ 就成立了。

这个类型则是区块类型（类型 2）。

目前这个结构只有两种类型的示例，因为不太好找到。

Part 7 唯一矩形的形式拓展

接下来我们来谈论一些关于将唯一矩形形式进行拓展的致命结构。

7-1 唯一环 (Unique Loop)

接着我们来看一个比唯一矩形要大一点的、依然可以形成致命形式的技巧——**唯一环 (Unique Loop, 简称 UL)**。

7-1-1 6 格的唯一环 (UL Size 6)

我们来学习 UL 技巧的第一个示例。

7-1-1-1 标准类型 (UL Size 6 Type 1)

4	8	9	1	7	2	³ 6	³ 6	5
6	5	1	4	_{8 9}	3	_{7 9}	_{7 8}	2
2	7	3	_{8 9}	5	6	₁	₁	4
3	₂ 6	₂ 6	7	1	8	4	5	9
5	9	7	3	6	4	8	2	1
8	1	4	5	2	9	₇	₆	3
9	₂ 6	5	₂ 6	₈	1	_{2 3}	4	7
7	4	8	₂	₃	5	_{1 2 3}	_{1 3}	6
1	3	₂ 6	₂ 6	4	7	5	9	8

如图所示，我们发现，涂绿色的五个单元格{r4c23, r7c2, r9c34}只含有候选数 2 和 6，如果我们此时算上 r7c4 的话，就会有一个神奇的现象发生：

和唯一矩形类似，这个结构涉及 r479c234b478，一共九个区域。如果 r7c4 只有候选数 2 和 6 的话，那么提到的这 9 个区域下就都恰好含有关于 2 和 6 的显性数对的结构。它们和唯一矩形非常接近，一旦排除了这些区域的 2 和 6 后，其余的单元格的任意候选数都不会受到影响，就相当于{r4c23, r7c24, r9c34}六个单元格的 2 和 6 完全被禁锢住了一样，这六个单元格里的 2 和 6 的填数可以随意交换，而其余的单元格却不会受到任何影响。如果题目唯一解，单凭这 6 个单元格来说，就已经产生两种填法了，其它的单元格又都是完全一样的候选数局面，这就必然说明了题目产生了看似双解（两个解）的局面，这和题目唯一解的要求相违背，即产生了致命形式，所以假设错误，即 r7c4 不能只有 2 和 6。那么，“不能只有 2 和 6”的意思就是，r7c4 \neq 26，否则，不管 r7c4 存在候选数 2 还是候选数 6，就一定会产生客观的另外一种填数情况，致使该结构形成 2 和 6 的致命形式。

我们再来熟悉一下这种结构，来看看这种结构能不能拥有和 UR 一样的变体类型。

7-1-1-2 区块类型 (UL Size 6 Type 2)

<small>3</small>	1	6	7	<small>4</small>	<small>3</small>	5	9	2	<small>4</small>	<small>3</small>
<small>8</small>	4	9	2	1	6	<small>3</small>	7	5	<small>8</small>	<small>3</small>
<small>3</small>	5	7	2	9	<small>4</small>	<small>3</small>	1	<small>4</small>	<small>3</small>	6
<small>8</small>	2	7	3	6	<small>1</small>	<small>1</small>	5	8	9	
1	4	5	8	7	9	<small>2</small>	<small>3</small>	<small>2</small>	<small>3</small>	
6	8	9	3	5	2	4	1	7		
5	<small>3</small>	<small>6</small>	<small>4</small>	9	<small>2</small>	<small>3</small>	<small>2</small>	<small>6</small>	<small>4</small>	1
9	<small>3</small>	<small>1</small>	<small>6</small>	5	<small>1</small>	<small>2</small>	8	<small>4</small>	<small>6</small>	<small>2</small>
7	2	<small>1</small>	4	<small>1</small>	6	3	9	5		

如图所示，可以观察到的是，在{r1c19, r2c69, r3c16}六个单元格里，除了 r1c9 和 r3c6 两格含有额外的数字 4 以外，其它剩下的单元格都只含有候选数 3 和 8。

如果 r1c9 和 r3c6 都没有候选数 4 的话，这六个单元格就恰好只有 3 和 8。而这个结构涉及到的是 r123c169b123，也是九个区域。排除掉这 9 个区域下的 3 和 8 的候选数外，其余的单元格并不会受到任何的影响。换句话说，这 6 个单元格的填数进行任意的交换，其余的单元格的局面都是不会发生任何变动的。如果题目唯一解，就不可能出现类似于上述逻辑一样，居然还存在有 6 个单元格有互换的局面。所以 r1c9(4)和 r3c6(4)不能同时都消失于盘面中。所以，这两个 4 至少有一个 4 是成立的，也就是说 r1c9 和 r3c6 里必须至少有一个是填 4 的。所以，它们共同对应的地方 r1c5 和 r3c8 就不能填入 4 了，故删除掉它们。

这便是和区块类型很接近的一则示例，所以它在 UL 里也被称为区块类型。

7-1-1-3 数组类型（UL Size 6 Type 3）

4	1	7	3	2	9	5	8	6
2		5	6		1		2	3
	9					9	7	
2		3	8		5		1	2
	9						7	9
5	8	1	2	9		6	4	
6	2	9	5	4		8		1
3	7	4	1			2	9	5
1	9	3	4		5	7		
7	4	5		3	2			
8	6	2	9	7	1	3	5	4

如图所示，我们仔细观察 **r2c89** 两格。如果两格都没有橙色数字，则显然会出现矛盾：**{r2c89, r4c69, r5c68}**六个单元格只有 3 和 7，并且涉及的 **r245c689b356** 这九个区域必然都会存在 3 和 7 的显性数对形式，结构内部会产生互换，也不会影响到其它任何单元格。所以，此时是形成了致命形式的。

但是如果 **r2c89** 只有橙色的候选数的话，显然也不行：因为此时 **r2c189** 就都只会有候选数 2 和 9 了，三个单元格显然是放不下两种候选数的，就会必然存在一个单元格无法填数，所以这也是矛盾的。

所以，**r2c89** 里必须有一个单元格是填入 3 或 7，而另外一个单元格则必须是 2 或 9，这样的话，不论是 2 还是 9，也不论 2 和 9 放在 **r2c8** 还是 **r2c9** 上，都可以和 **r2c1** 构成一个“待定的”数对结构，这个数对能确定的位置是 **r2c1**，而另外一个单元格则不能从 **r2c8** 和 **r2c9** 里确定下来。不过 **r2** 里显然有 2 和 9 的出现了，因此 **r2** 其余单元格就不能填入 2 和 9 了，删掉它们。

这便是和 UR 的数组类型很相近的 UL 数组类型。

7-1-1-4 共轭对类型（UL Size 6 Type 4）

最后我们来看一则共轭对类型的实例。

6	4	<small>7 5</small>	<small>2</small>	1	<small>5 9</small>	3	<small>2</small>	
8	9	<small>1 5</small>	3	<small>2</small>	<small>5 6 4 6</small>		<small>7 8</small>	<small>1 8 9</small>
2	3	<small>1 7</small>	4	8	<small>6 9</small>	5	<small>7</small>	<small>1 6 4 9</small>
5	7	9	1	3	2	<small>4 6</small>	<small>6 4 8</small>	<small>8</small>
3	6	2	9	4	8	7	1	5
4	1	8	5	6	7	9	3	2
1	5	6	<small>2</small>	<small>2</small>	4	8	9	3
9	8	3	<small>7</small>	<small>7</small>	5	1	2	4
7	2	4	8	9	3	1	5	6

如图所示，我们观察 c8，发现数字 2 存在共轭对，位于 r12c8。这也就表示，r12c8 里必须有且仅有一个单元格是填入 2 的。那么，既然有一个单元格是 2 的话，那么另外一个单元格就不能是数字 7，否则，算上 r1c4、r2c5 和 r7c45，这样六个单元格就只是 2 和 7 了，就必然客观存在两种不同的填数模式，使得形成致命形式。所以，r12c8 里有一个是 2，就必须让另外一个单元格不填 7。和 UR 的共轭对类型的逻辑完全一样，这里也必须删除掉 r12c8 两处的候选数 7。否则但凡“残留”一个 7，就必然能够有一种情况形成致命形式，这样也是我们不想看到的（毕竟形成致命模式就好比是产生了矛盾，所以可以反推得到假设错误）。

看起来 UL 好像只能是 6 格。那么我们来看看更大的例子。

7-1-2 8 格的唯一环 (UL Size 8)

9	1	4	5	2	6	3		
6	2			8	1	4	5	9
5		8	9	4		1	6	2
2		9			8	5	1	
4	8	6	1	5	9	7	2	3
1	5		2			8	9	
3	6	1		9	5	2	4	
	4	5			2	9		1
	9	2		1		6	3	5

如图所示。如果 r9c4 只有候选数 7 和 8 的话，那么{r1c89, r7c49, r8c18, r9c14}这样 8 个单元格涉及的所有区域 r1789c1489b3789 一共 12 个区域里全部都会产生 7 和 8 的显性数对，而其余的单元格均不会受到任何 7 和 8 的影响，所以这 8 个单元格不管随便怎么互换填法，对于其余单元格而言，都是无所谓的，因为怎么换都影响不了它们，使得出现致命形式。所以 r9c4 <> 78。

7-1-3 10 格的唯一环 (UL Size 10)

		3	4			8	7	9
		8		9	3			2
9		2		6	8	5	3	4
		9	8					7
3	2	5		4	7	9		8
	8	7			9			
7	5		2	8	4		9	3
2	9	4	3	1	6	7	8	5
8	3		9	7	5	2	4	

如图所示，如果 r6c4 只有候选数 1 和 6 的话，{r2c78, r4c48, r5c49, r7c37, r9c39}这 10 个单元格的 1 和 6 在所有涉及的区域 r24579c34789b35679 这 15 个区域里都会形成显性数对结构，而随意交换都不会影响到其它的任何单元格，所以形成致命形式，所以矛盾，故 r6c4 <> 16。

7-1-4 12 格的唯一环 (UL Size 12)

2	5	1	4	1	6		3	8
3	1		4	1	2	8	5	6
8		6		3	5	1	2	4
4	8	5	2	6	7	3	9	1
	1		3	5	8	1	2	6
6	2	1	1	1	3	8		5
1	4	8	3	7	9	6	5	2
5	3	2	6	1	1		8	
	6		8	5	2	4	1	3

如图所示，如果 r1c3、r2c4 和 r5c2 三个单元格都不填数字 1 的话，那么涉及了 12 个单元格和 r123589c123479b123479 这 18 个区域里将会产生 7 和 9 的显性数对结构。进而形成了随意互换的致命形式，所以 {r1c3, r2c4, r5c2} 里必须至少有一个单元格是数字 1，那么可以删掉它们的交集，故 r2c2 \neq 1。这个实例套用了 UR 的区块类型。

7-1-5 14 格的唯一环 (UL Size 14)

2		2	9	1	3	1	3	4	5	6
6	4	4	5	2	7		8	1	3	
3	1	5	4	8	6	2	9	7		
2	5	2	7	6	1	3	8	1	3	4
1	3	1	8	9	2	7	6	5		
7	6	8	1	4	5	9	2	1	3	
5	2	6	1	7	8	1	3	4	9	
1	4	1	6	1	3	9	5	7	2	
1		1	2	5	4	6	1	3	8	

这个结构算是很大的结构了，逻辑我就不阐述了，注意，这个要套用 UR 的共轭对类型来理解，r8c3(3) 是共轭对。

7-1-6 原理进一步剖析

7-1-6-1 #1: UL 占据的单元格总数必须是偶数格吗？

关于这一点。是的。如果结构涉及奇数个单元格，而且还要使得每一个区域都有数对的话，这样是完全无法实现的。你可以试着考虑一下，拥有 5 个单元格的唯一环可能长什么样子。

一定要注意的，涉及的区域是要看每一个单元格都占了哪些区域，而不是以数对为单位。

7-1-6-2 #2: UL 最大能占多少个单元格？

这一点不好说明白，而且没有示意图，请你跟着我的描述找出对应的情况。

首先，这个问题的答案是 14。我们先来考虑 18，18 是最大的可能情况，因为全盘一共 9 个宫，每一个宫都有数对的话，就恰好 18 个单元格。但是，一旦这种形式形成了致命结构的构型，那么这个结构的数字就完全可以随意互换了。因为所有宫的这两个数都没有提示信息来排除一种填法，所以，必然是两种填法的，直接成为了双解题。

再少一些，16 个单元格。看似 16 个单元格不违背唯一解的要求，但这种形式的结构必然会出现在 8 个宫里，而剩下的一个宫就一定确定了这个数对涉及的数字，它们的位置。而这显然是做不到的，因为随意放下两个数，整个 UL 都会矛盾，因为它占据的随意两格都会让整个结构的其中一个数对直接无法填数，致使出现无解现象，所以它依然是不可能的。

所以，一个 UL 最多也只能涉及 14 个单元格，即上面的示例那样。而且这也是我收集的唯一一则 14 个单元格的 UL 示例。

7-2 拓展矩形 (Extended Rectangle)

如果说唯一环是从形状进行拓展，那么接下来的结构就是从数字进行拓展。

7-2-1 6 格的拓展矩形 (XR Size 6)

我们先来看看基础的结构。

7-2-1-1 标准类型 (XR Size 6 Type 1)

9	¹	6	7	¹	2	4	3	⁶	5
8	4	2	3	6	5	9	1	7	
¹	5	6	3	¹	5	9	¹	7	²
¹	2	5		7	8	4	6	²	3
¹	5	6	3	¹	5	9	6	²	3
¹	2	5	6	3	¹	5	9	²	3
¹	5	6	3	¹	5	9	6	²	3
¹	2	5	6	3	¹	5	9	²	3
¹	5	6	3	¹	5	9	6	²	3
¹	2	5	6	3	¹	5	9	²	3

如图所示，它把唯一矩形变为了三数的结构了。虽然看起来和唯一矩形很相似，不过推理逻辑和唯一矩形就不太不一样了。

如果 $r3c1$ 只有 1 和 5 的话，我们将会得到 $r357c13$ 这六个单元格有且仅有 1、3、5。我们假设 $r357c1$ 已经填入了数字，就暂时假设为 a_1 、 a_2 和 a_3 ；同理，我们假设 $r357c3$ 填入的数字是 b_1 、 b_2 和 b_3 。

我们先不用去管具体这些字母都代表了什么数字，但我们可以将填入的数值进行下标相同的置换（说白了就是 a_1 和 b_1 交换， a_2 和 b_2 交换， a_3 和 b_3 交换），这样可以让数字对位替换到另外一边，比如针对于 $r5c13$ 而言，上述的交换就相当于把原来 $r5c1$ 填的数字替换到 $r5c3$ 上去；而把 $r5c3$ 的原本的填数换到 $r5c1$ 上。

实际上，这种交换没有任何的意义，但是可以看到，这个结构只涉及 $r357c13b147$ 这 8 个区域，而这 8 个区域都不会因为刚才的交换而改变其它的候选数信息，毕竟刚才的交换，只是这 6 个单元格内部的交换，而交换的数据对于这 8 个区域来说全部都是数对数组形式，这便回到了 UR 的形成致命形式的论证过程：在结构所有涉及的区域上都产生了独赢的数对数组形式，而仅仅是这些数对数组的删数，而不会对其它任何的单元格造成任何填数的影响（其它的单元格的填数状态和候选数数据信息全部是一致的）。如果题目唯一解，就不会产生这种现象：因为唯一解的题目，对于每一个单元格而言，都只能有唯一的一种填数情况，显然这 6 格就已经不满足要求了：其余单元格的候选数状态信息完全是一致的，所以这些位置的填数不论是否是唯一的，就看 $r357c13$ 而言，就无法使得题目唯一解，违背唯一解的要求。所以，原假设不成立，即 $r3c1$ 只能填入 6。

这个技巧将唯一矩形进行了拓展，因而成为**拓展唯一矩形**，简称拓展矩形（**Extended Rectangle**，简称 **XR**，但该技巧很少用简称，除非是一些必须简化名称的地方才会使用）。不过一些资料里依然称这个结构为唯一矩形。

7-2-1-2 区块类型（XR Size 6 Type 2）

4 5 9	3 4	5 9	8	4 5	6	1	7	2
8	1	7	9	2	3	5	4	6
4 5	6	2	1 5	7	1 4 5	9	8	3
2 5 9	7 8	6	2 3 5	1 3 5	7	4	1 2	5 8 9
1	4	5 9	6	8	4	7	3	5 9
2 4 5 9	7 8	3	2 5	1 4 5	7	6	1 2	5 8 9
2 3	5	4	1 2 3	9	1 2	8	6	7
7	2	8	4	6	2 5	3	5 9	1
6	3	1	7	5	8	2	5 9	4

如图所示，如果 r46c9(5)同时消失的话，r46c269 将只剩下 7、8、9，形成拓展矩形的致命形式，所以 r46c9(5)必须至少有一个是对的填数，而由于它们同宫，所以组成区块，删除 r5c9(5)。

7-2-1-3 数组类型（XR Size 6 Type 3）

接下来我们来看一则数组类型的实例。

3 1 3 1 3	4	1 2 3	3	1 2 3	5	1 2	5	6
7 9	8	7 8 9	3	1 3	6	9	7	1 4
2	1 3	4 5 6	5	8	8	5	3	8
4 6 7 9	1	6	1 2 6	5	4	3	8	
7 9	1 2 5	6	2	4	7 8 9	1 5	1 5	3
4 7 9	3	5	3	6	1	4 5 8	5	2
8	1 2 3 4	1 3	2 3	5	3	7	6	1 4
5	9	2	1	3 6	4	3 6	8	7
3 6	7	4	5	3 6 8	3 6	1 2	1 2	9
1	3 6 8	3	7 9	7 9	2	3 6	4	5

如图所示。这个例子希望你自行理解和推理。它只是将拓展矩形和 UR 的数组类型并在一起使用了，你可以试试看。提示一下，这个例子里使用的是带隐性数对的数组类型。

7-2-1-4 共轭对类型 (XR Size 6 Type 4)

1	2	5	8	4	6	5	5	3
5 6	4 5 6	7	1	9	3	2	4 5	4 5 6
9	4	3	2	7	5	4	6	8
4	3	1	5	2	1	2	8	1
3	5 6	5 6	2	3	8	1	4 5 6	4 5 6
8	9	1	4	6	2	3	1 2	5
3	8	3	5	2	4	1	7	9
2	1	9	7	3	8	4 5	6	4 5
7	4 5	4 5	6	1	9	8	3	2

如图所示，我们可以聚焦于 **r46c368** 六个单元格。可以看到，其中 **r46c68** 四个单元格只含有 1、2、7；而 **c3** 存在 **r46c3(1)** 的共轭对。此时我们就可以这么想：因为 **r46c3(1)** 必须有一个是正确的数字，那么这两个单元格的另外一个单元格就不能填入 7，否则两个单元格形成 1 和 7 的显性数对结构，并和旁边 **r46c68** 组成拓展矩形，并形成关于 1、2、7 的致命形式，即 **r4c368** 的填数就可以上下交换，放到 **r6c368** 里；同理，**r6c368** 的填数就换到 **r4c368** 来，于是出现致命形式。所以为了规避致命形式的出现，**r46c3 <> 7**。

7-2-1-5 变体类型

2	6	1	3	4	4	5	1	6	7	5 6
4	7 8	1	6	2	2	5	1	3	2 3	9
5	7 8 9	6	1	2	2	3	6	2 3	8	4
3	4 5	9	2	8	1	3	4	3	6	3
3	6 4	6	8	4	2	3	4	6	4	2 3
1	4 5 6	9	7	2	4	6	8	4	5	9
6	1	4 5 6	9	2	4	6	8	7	4	3
7 8	2	4	6	4	4	6	9	5	4	1
7 8 9	3	4 5 6	9	4	4	6	1	2	4	8 9

如图所示，这一则示例里，它不太像是前面的两则示例，因为它好像是把矩形结构“旋转”了一下，放在了两个宫里。但是推理的逻辑则是完全一样的。

我们假设 **r13c3(9)** 全部消失于盘面里（即当它们不存在），而假设 **r123c3** 的填数分

别是 a_1 、 a_2 和 a_3 ；而假设 $r123c7$ 的填数分别是 b_1 、 b_2 和 b_3 。然后我们发现，这个结构涉及 $r123c37b13$ 七个区域，而这七个区域下，如果我们将下标相同的数值交换一下，只是将这 7 个区域的数组结构内部的填数进行了交换，而不会影响外部的填数情况和候选数信息。所以这便产生了两种填法，违背了唯一解的要求，进而产生矛盾。

所以为了规避这一点，我们的假设就错误了，即 $r13c3(9)$ 不能同时消失于盘面里，它们俩必须至少有一个 9 是对的。

这种构型好比是旋转了矩形，但推理和拓展矩形的思路和逻辑完全一样，所以也算作一种拓展矩形的变体。

我们再来看一则这样长的矩形的示例。

<div>2 3 6</div>	5	<div>2 3 8</div>	<div>1 3 6</div>	<div>1 2 8</div>	9	7	4	<div>1 2 8</div>
1	9	<div>2 3 8</div>	<div>3 8</div>	4	7	<div>2 3</div>	5	6
7	<div>4 3 6 4</div>	<div>2 3 8</div>	<div>1 3 6</div>	<div>1 2 8</div>	5	<div>1 2 3</div>	<div>1 2 3 9</div>	<div>1 2 8 9</div>
<div>3 8</div>	2	9	4	<div>1 8</div>	6	5	7	<div>1 3</div>
<div>4 8</div>	7	6	<div>1 8</div>	5	3	9	<div>1 2 4</div>	<div>1 2 4</div>
<div>4 3 5</div>	1	<div>4 3 5</div>	9	7	2	6	8	<div>4 3</div>
<div>4 3 6 4</div>	<div>3 6</div>	7	2	<div>6 4 9</div>	<div>1 4 8</div>	<div>1 4 8</div>	<div>1 6</div>	5
9	<div>4 6</div>	<div>1 2</div>	5	3	<div>1 4 8</div>	<div>1 2 4 8</div>	<div>1 2 6</div>	7
<div>2 5</div>	8	<div>1 2 5</div>	7	<div>6 4 9</div>	<div>1 4 8</div>	<div>1 2 3 4</div>	<div>1 2 3 6</div>	<div>1 2 9</div>

如图所示。如果 $r89c7$ 两个单元格都只有 1、4、8 的话，显然这两列的填数是可以左右互换的，所以产生了致命形式，这样填是不行的。

但是反过来，如果两个单元格都只有 2 和 3 的话，显然也不行，因为 $c7$ 上还有一个单元格只有 2 和 3，如果 $r89c7$ 都只有 2 和 3 了，则这两个单元格显然就构成了数对形式，而上面 $r2c7$ 就无法填数了，所以也矛盾了。

因此，我们不得不拿出 $r89c7$ 的其中一个单元格，让它只能填入数字 2 和 3。这样一来， $r2c7$ 就只有 2 和 3，与之将产生数对结构，所以 $c7$ 其余单元格都不能填入 2 和 3，删掉它们。

7-2-2 8 格的拓展矩形 (XR Size 8)

	6		6	4	8	5	1	3	2	7
2	5	3	6	4	7	1	9	8		
1		1		8	9	2	3	6	5	4
8	2		5	4	5	9	6	4		3
1		1		6	4	3	4	5	4	2
3	4		5	2	1	5	9	7	8	6
5	8	1	4	5	6	9	2	4		3
5	6		6	2	3	8	4	5	4	1
4	3	9	1	7	2	8	6	6	5	

如图所示，我们假设 r8c1 只有候选数 6 和 7，则就可以按照假设填数的方式，然后产生左右交换，进而对于结构涉及的 r1358c12b147 这些区域里产生“交换也不会影响数对数组”的违背唯一解要求的现象，进而形成致命形式，产生矛盾。所以 r8c1 \neq 67。

7-2-3 10 格的拓展矩形 (XR Size 10)

5	6	5	6	2	1	6	1	4	1	3	5
4	8	1	2	5	3	7	9	6			
7	5	6	3	1	6	8	4	2	1	2	5
5	4	5	4	5	1	6	2	5	6	9	3
2	1	2	6	3	4	1	5	1	2	7	1
2	5	8	9	5	1	6	1	2	6	5	6
1	4	5	6	4	5	8	7	3	4	6	8
3	4	7	6	4	7	8	1	2	5	6	7
2	5	6	5	6	9	4	3	5	6	2	1

和上一则示例的推理思路类似，假设 r67c4(5)同时不存在的时候，填数则可以左右交换，然后产生致命形式，然后矛盾。所以 r67c4(5)至少有一个成立，那么就可以当作一个广义的区块那样，删除所在行的其余位置的 5。

7-2-4 12 格的拓展矩形 (XR Size 12)

3	9	5	2	4	2	4	6	1
1	4	6	7	5	4	3	9	2
8	2	2	9	1	1	7	5	3
5	1		3	2	1	3	1	3
6	3	4	1	5	8	2	9	7
2	1		3	6	1	3	1	3
4	8	3	2	9	1	2	1	2
9	2	1	6	3	4	3	2	3
7	2	2	2	1	3	1	3	4

这一则示例的推理逻辑完全一样，只是这一次是横着看，上下交换。

7-2-5 14 格的拓展矩形 (XR Size 14)

3	6	3	4	3	1	3	1	3	2	5
7	8	7	8	3	1	2	2	3	5	7
4	3	4	8	9	3	6	3	6	9	4
1	2	5	7	3	6	8	3	6	9	4
2	1	3	6	5	6	5	6	6	4	3
3	1	3	7	8	9	7	9	7	8	9
5	6	5	6	4	3	1	6	9	7	2
3	5	6	5	6	1	2	2	5	6	4
7	8	9	7	8	9	1	5	3	1	3
4	5	4	5	7	9	7	9	7	9	6
7	9	7	9	7	9	7	9	7	9	8
5	6	5	6	2	3	4	5	3	9	9

如图所示，最后这一则例子可以发现，结构基本上已经占满了 c12。不过推理方式类似，假设数值，然后左右交换。

7-2-6 原理进一步剖析

7-2-6-1 #1: 拓展矩形必须涉及偶数个单元格吗？

显然，我们可以说明的是，这是一句废话。因为结构不可能缺少任意一个可以形成矩形形状单元格，否则根本没办法进行上下或左右的交换操作。

7-2-6-2 #2: 为什么例子左右或上下对应位置的候选数完全一样？

这个巧合实际上并非是一种巧合，这个结构实际上只是通过对应位置交换而产生的结构，而这个结构的候选数的情况仅仅需要看所处行列的填数信息；而可以仔细观察所处行列，这些数字都是一样的，所以它们的候选数情况基本上是一样的。

7-2-6-3 #3：拓展矩形最大能涉及多少个单元格？

实际上，拓展矩形和唯一环有着些许关联，比如这个问题。答案和唯一环完全一样，也是 14。原因也很相似：如果是 18，则这两行或两列就没有任何的提示信息了。换句话说，这两行的填数就一定能上下或左右直接置换，所以这种情况必然是两种填法。

而如果是 16 的话，则意味着有一对是确定值，上下或左右对应起来；而要使得数字上下可以完全交换，这两个对应的数字就必须是一样的，才能使得剩余的空格上下或左右置换，而这种情况显然不可能，毕竟对应的填数是一样的，这便违背了同一行列有相同数字的规则。

所以，综上，最多的情况是 14。

7-3 可规避矩形 (Avoidable Rectangle)

接下来要说的技巧叫做**可规避矩形** (Avoidable Rectangle, 简称 **AR**)，是一种特殊的唯一矩形，这种结构长相十分别扭，以至于我们都不一定能认为它是一个唯一矩形。正是因为它的结构不太像唯一矩形，所以被单独取名和分类。我们一一来看，它到底有着什么样的用处。

7-3-1 标准类型 (AR Type 1)

8	1	5	4	7	2	9	6	3
4 6 9	3 4 6 9		5	3 6	8	7	1	2
7	6 2 3 7	2 6	1	3 6	9	4 8 4 8		5
4 6 7	2 7	1	8	5	3	4 6 2	9	4 7
3	2 9	4 6 2	6 9	1	7	4 5 6 2 4 5		8
7	6 9	5	8	6 9	2	4	1 6 3	1 7
5	8	3	7	9	6	1 2 4	2 4	1 4
2	6	7	3	4	1	5 8 5 8		9
1	4	9	2	8	5	3	7	6

如图所示，这个结构涉及三个填入数和一个候选数单元格，我们依然能保证 **r7c9** 使用唯一矩形，得到 **r7c9 <> 4**。

乍一看，**r7c9 = 4** 后确实 **r78c59** 四个单元格形成了“唯一矩形”的那种构型，不过这三个数不是已经都填好了吗？为什么依然能删除呢？在之前我们提到过一点，这些是我们填入的数字，既然是填的数，就随时可能发生变化。而且，一旦我们填入了其中一种写法后，就会立马客观产生第二种填法，即 **4** 和 **9** 互换的情况，这种情况在之前说过，是客观存在的，是不受候选数约束的，因为填好了一种写法后，结构所处的行列宫就立马产生了数对形式的东西，比如这里的 **4** 和 **9**，既然是数对，就肯定能发生交换，所以必然能切换到另外一个填法，使得这四个单元格形成致命形式，进而得到假设错误。所以 **r7c9 <> 4**。

可以看到，这个示例实际上并不难理解，难就难在，它混杂在盘面的确定值里，我们平时在寻找的时候，很少能找到这种东西。这还只是它的第一种类型。接下来我们来看它的第二种类型。

7-3-2 区块类型 (AR Type 2)

6 8 9	4 7		5	4 8 9	1	2	3 6 4 6	
2 5 6 9	2 4	5	2 6 4 9	3	3	3	1	8
2 6 8	1	3	2 6 4 8	7	5	9	4 6	
1	5	6	7	3	3	9	4	2
3	8	4	9	6	2	1	5	1
7	9	2	4	1	5	6	8	3
4	2	1	1 3	5	6	1 3	2 3	9
5 8	3	9	1	6	2	4	1	5 6
2	6	1	1 3	7	9	4	2 3	1

如图所示,如果我们假设 r1c23(7)都消失,就会发现 r1c23 两格都变为了唯一余数,使得 r1c2 = 4、r1c3 = 8,这样一来, r15c23 四个单元格就形成了 AR 的致命形式。所以,两格里的 7 不可同时去掉;即至少两个 7 得有一个是填到单元格里的。

这样, r1c23(7)就形成了区块结构,所以 r1 和 b1 里其余位置的 7 都将被删除。

7-3-3 数组类型 (AR Type 3)

接下来我们来看两则数组类型的示例。

因为 AR 不好观察,所以 AR 这个类型的例子非常少。

7-3-3-1 AR + 显性数对 (AR Type 3 With Naked Pair)

1	3	2	3	8	6	4	9	5
5	8	9	2	4	3	1	6	1 3
4	3	3	1	5	9	8	3	2
3	4 5 6	1	8	9	4 5	5 6	2	4 3
8	2	4 5 6	3	4	4 5	1	5 6	9
3	9	4 5	6	2	1	5	4	8
6	4 6	8	9	1	3	2	5	4 6
2	1 3	4 6	5	7	8	9	1 4	1 6
9	1	5	4	6	2	3	8	1 7

如图所示,如果 r89c9 两个单元格都只能填入绿色的候选数的话,则 r89c59 必然形

成可规避矩形的致命形式，所以矛盾；而如果两个单元格都只填橙色的数字的话，b9 里就会存在{r8c89, r9c9}三个单元格都只有候选数 1 和 4，显然这样是填不满三个单元格的，所以这样也是矛盾的。所以 r89c9 里必须只有一个单元格填绿色数字，而另外一个单元格只能填入橙色数字。

那么这样一来，r89c9 里必然会有 1 和 4 的话，就会和 r8c8(14)形成 1 和 4 的数对结构，所以 b9 里的其余单元格都不能放 1 和 4，所以删除 r7c9(4)；另外，这个显性数对还是一个带有区块的显性数对，因为数字 4 只能放在 r8c89 上，所以形成了区块，r8 其余位置都不能填入 4。

7-3-3-2 AR + 隐性三数组 (AR Type 3 With Hidden Triple)

3	7	5	<small>4 6 9</small>	8	<small>4 6</small>	1	2	<small>6 9</small>
4	2	8	<small>6 9</small>	7	1	5	<small>3 6 9</small>	
6	1	9	3	2	5	4	8	7
5	3	<small>1 2</small>	<small>1 2</small>	4	<small>7 6</small>	9	<small>7 6</small>	8
<small>1 7 8 9</small>	<small>8 9</small>	<small>1 6 7</small>	<small>1 3 6 7 8</small>	<small>3 6 7 8</small>	<small>3 7 8</small>	2	5	4
<small>7 8</small>	<small>4 8</small>	<small>2 4 6 7</small>	<small>2 7 8</small>	5	9	<small>3 6 7</small>	<small>3</small>	1
<small>7 8 9</small>	6	<small>4 7</small>	5	1	2	<small>3 8 9</small>	<small>4 9</small>	<small>3 9</small>
2	<small>4 8 9</small>	<small>1 4 7 8</small>	<small>3 6 7 8</small>	<small>3 4 6 7 8</small>	<small>3 8</small>	<small>3 1 3 9</small>	<small>4 6</small>	5
<small>1 8</small>	5	<small>1 4</small>	<small>4 6 8</small>	9	<small>4 6 8</small>	7	<small>1 4 6</small>	2

如图所示，和刚才的逻辑一样，我们可以得到 r7c78 里必须有一个单元格是填入绿色数字（4 或 8）的。而 r7 里所有能放下 4、7、8 三种数字的位置就只剩下 r7c13 和 r7c78 的其中一个单元格了。所以，r7c13 会和 r7c78 的其一构成隐性三数组。

所以，删数就很明显了：r7c13 里的其余非 4、7、8 的候选数。

7-3-4 隐性可规避矩形 (Hidden AR)

接下来提到的是 AR 的最后一种类型，不过这种类型没有和 UR 一样的标准共轭对类型版本，而是这个类型更类似于隐性唯一矩形那样，共轭对有两个，还是呈垂直形式出现的。

6	2	1		7	5	4		

如图所示，我们观察到 r2c15(9)和 r23c5(9)在所在的区域上都是共轭对，即两处必须有一个是正确的数字，所以这三个数字一共就可以分为两种情况：

- r2c1 = r3c5 = 9;
- r2c5 = 9。

当第一种情况下时，为了规避 AR 的致命形式，所以 r2c5 <> 8；而第二种情况，r2c5 已经填好了数字，所以显然不能再填 8，所以 r2c5 <> 8。

所以，综上所述，r2c5 <> 8，即为这个例子的结论。它实际上更类似于 AR 版的隐性 UR 结构，所以有时候也被称为**隐性可规避矩形**（**Hidden Avoidable Rectangle**），不过这种例子出奇的少。

那么，AR 的内容就讲完了。

7-4 全双值格致死解法 (Bivalue Universal Grave)

这一个部分要讲解一个针对于全盘内所有没有填数的单元格的结构，而这一节的原理稍微比较抽象一些，希望你同之前一样，拿出草稿纸写写画画。

7-4-1 标准类型 (BUG Type 1)

5	7	8	1	3	6	2	9	4
¹ ₆	^{1 3} ₆	4	2	8	9	7	^{3 6} ₆	5
^{6 9} ₉	^{3 9} ₉	2	5	7	4	1	8	^{3 6} ₆
4	5	9	3	2	1	8	^{7 6} _{7 6}	⁶ ₆
3	8	1	7	6	5	9	4	2
7	2	6	9	4	8	5	^{1 3 1 3} _{1 3}	^{1 3} _{1 3}
^{1 9} ₉	6	7	8	5	3	4	2	^{1 9} ₉
8	4	5	6	^{1 9} ₉	2	3	^{1 7} ₇	^{1 9} ₉
2	^{1 9} ₉	3	4	^{1 9} ₉	7	6	5	8

如图所示，我们观察全盘，发现一个神奇的现象：全盘除了 **r8c9** 这一格外，其余所有的单元格都有两个候选数，唯独 **r8c9**，有三个候选数。而且，这个结构特殊之处在于，结构所有区域（每一个行、列、宫），均包含规格参差不齐的数组结构，而这样的结构正是由于这个原因，没有任何的矛盾推论。这就很奇怪了，并促使我们想思考一个问题：居然这么神奇的结构，都没有删数的结论，我坚信它是有逻辑得到删数结论的，而这个逻辑到底是怎样的呢？

如果我们去掉 **r8c9** 的候选数 1，先当 **r8c9(1)** 不存在。于是，你会发现一个神奇的现象：所有没有填数的单元格里，恰有两个候选数。

既然所有的空单元格里都是两个候选数，而每一个空格涉及到的所有区域下，所有空单元格都恰好能够构成数组结构。比如 **b3**，所有空单元格 {**r2c8**, **r3c9**} 直接可以构成 36 数对；又如 **c8**，所有空单元格 **r2468** 直接可以构成 1367 四数组，所有区域都是这样。

好像是个废话？好像 **r8c9(1)** 去不去掉也是这样的吧？看我接下来怎么推导哈。

既然每个单元格都是两个候选数，而且都能找到和它一起构成数组的其它单元格，那我们随便选取一格，来假设填数。随便选取一格，比如说 **r2c1**，有 1 和 6 两个候选数。我们假设就会分为 **r2c1 = 1** 和 **r2c1 = 6** 两种填数情况。

又是假设？这么麻烦？我们不需要假设填完全部情况，而是简单思考一下就可以了。试想一下，全盘既然都是两个候选数，那又可以构成数组结构，那么只要有一格填完数字，而每一个单元格都可以找到和它一起构成数组的其余单元格，于是就可以把这个数组填完。比如说 **r2c1**，与之构成数组结构的有 **r2c128** 的 1、3、6 三数组、**r23c12** 的 1、3、6、9 四数组以及 **r237c1** 的 1、6、9 三数组。既然 **r2c1** 填入其中一个数了，那这些数组理所

当然地可以根据这个数推理而得到剩下的部分。由于全盘都是两个候选数的单元格，那 **r2c1** 填入其中一个数值，就必然会形成一种填法。同理，**r2c1** 填入另外一个数值，那就必然会形成另外一种填法。随即形成两个填数完全不一致的盘面结果。

我们再详细阐述一下。就比如一个三数组，又因为这个结构全部单元格都是两个候选的缘故，三数组的话，三个单元格的候选数情况就只可能是这样的：

ab、ac、bc

那么如果{ab}这一格填 a 和填 b，会产生两种不同的情况：

- 情况 1：a、c、b
- 情况 2：b、a、c

每一个单元格在其行、列、宫上都能够找到对应的数组结构，所以每一个行列宫都可以发现这样的形式，直到完成盘面。

每一个两个候选数的单元格，意味着每一个单元格都有两种不同的填法，而假设其中一格，都能使得这样的单元格直接出数。所以全盘的空单元格都可以根据一个单元格，其余单元格会受到此处填入的情况影响，导致其它位置上的填数也会产生变化，即“牵一发而动全身”。

因为有两种填法，所以这样的填法一定会产生两个填法完全不一样的盘面结果。由于全盘都有数组结构的缘故，数组的互换情况就导致了两种完全不同的填数方式。如果其中一个盘面是正确的话，那么另外一个盘面就也应该是正确的。

试想一下，数独只有一个答案，怎么可能出现两个盘面都是正确的的情况？所以两个情况都应该是错的才对。因为刚才我们说到过，只要其中一种填法是对的，那另外一种填法因为数组的互换，也应该是正确的。两个都是对的，这明显不满足数独唯一解的要求呀。所以，这样的结构，即全盘都只有两个候选数、每一格都能够构成数组的特殊结构，就不应当单独存在于盘面之中。所以，最初的假设（假设 **r8c9(1)** 消失于盘面之中）就不正确了。反过来，结论就应为 **r8c9 = 1**。

这个结构称为**全双值格致死解法**（**Bivalue Universal Grave**，简称 **BUG**）。

BUG，全名 **Bivalue Universal Grave**，直译的中文叫“双值-全-坟墓”，所以第一个版本的翻译是“全双值坟墓”，后面发现翻译不大对，并且体现不出本质意思后，就被翻译为了“双候选数致死解法”，后面发现，这个说法还有歧义，随后又被翻译为“全双值格致死解法”，里面的**双值格**（**Bivalue Cell**）指代的是里面有只有 2 个候选数的单元格。

这里介绍一个东西。有多少个候选数就称为“多少值格”，有三个候选数的单元格就称为**三值格**（**Trivalue Cell**），而一般多于两个候选数的单元格都称为**多值格**（**Multivalue Cell**）。

还有一个问题。这个结构为什么叫“致死解法”呢？为什么不叫“致命”而是“致死”？这是因为，在 **r8c9(1)** 去掉之后，结构本身不是出现 **UR**、**UL**、**AR** 以及拓展矩形那样、内部有两种不同的填法，却对剩余盘面不造成其它删数影响的形式，而结构本身内部就已经是无解的了。刚才我们就说到过，如果其中一种填法是正确的，那另外一种填法肯定也是正确的，这就出现两种不同的填法都正确的现象，这违背了唯一解的要求。所以结构的两种填法，它们的内部都是不可能存在正确的填法的，换句话说，怎么填都会出现违背数独规则的矛盾。这一种现象，我们称为“致死”，就不再叫“致命”了。

那么，结构到底是怎么看到这个结论的呢？难道我还要把 r8c9 的所有候选数一个一个地试一遍才知道？当然不是，接下来我就告诉大家怎么看到它。

7-4-2 原理进一步剖析

7-4-2-1 怎么观察？

之前说到两点：

- 只有一个三值格，其余都是双值格；
- 每一个单元格都能找到某个或某些同区域（行、列、宫）的单元格与之构成数组结构。

是满足结构的必要两点。那么，针对这两点，我们可以得到一个便于观察的推论：每一个候选数在其所在的行、列、宫下，均有两个相同的数。文字可能叙述不严谨，来看一下我举例你就知道了。

5	7	8	1	3	6	2	9	4
<small>1</small>	<small>1</small> <small>3</small>		4	2	8	9	7	<small>3</small> <small>6</small> 5
	<small>6</small>	<small>3</small>	2	5	7	4	1	8 <small>3</small> <small>6</small>
4	5	9	3	2	1	8	<small>7</small> <small>6</small>	<small>7</small> <small>6</small>
3	8	1	7	6	5	9	4	2
7	2	6	9	4	8	5	<small>1</small> <small>3</small>	<small>1</small> <small>3</small>
<small>1</small>		6	7	8	5	3	4	2 <small>1</small> <small>9</small>
8	4	5	6	<small>1</small>	2	3	<small>1</small>	<small>1</small> <small>7</small> <small>9</small>
2	<small>1</small>		3	4	<small>1</small>	7	6	5 8

当 r8c9 \leftrightarrow 1 时，每一个数在其所在的行、列、宫上恰好都有两个同样的数。比如 r3c9(3), r3c9(3)所在的行(r3)、列(c9)和宫(b3)内，都恰好有两个 3；又如 r7c1(9)，r7c1(9)所在的行(r7)，列(c1)和宫(b7)内，都恰好有两个 9。

那么，这里提出一个方便后面叙述的术语词：**真数**（True Candidate）或非 BUG 候选数（Non-BUG Candidate）。它们指的是和 BUG 结构无关的候选数，或者说是那个正确的数字，比如盘面之中 r8c9(1)就是真数。当然，后面的题目有可能是多个真数的，不过这多个真数之间的关系并非像这里 r8c9(1)一样直接正确。

所以，我们在找的时候，就把不是出现两次的数字都给它找出来，就可以了。但是，前提是，这个结构在去掉这样的数字之后，是能出现 BUG 致死形式的。

7-4-2-2 两个不能用 BUG 的例子

5	4		^{2 3}	9	²	8	³	1
3	9	2	¹ ₄	¹	8	⁶	5	^{4 6}
	1		³	^{6 4 5}		2	^{4 9}	
	3	4	6	¹	¹	5	9	2
6	2		⁵ ₉	4	⁵ ₉	1	^{7 8}	3
1	5	9	²	²	3	⁶	4	⁶ ₈
² ₉	8	5	⁴ ₉	² ₆	¹ ₄	^{3 1} ₃	7	
² ₉	7	1	8	3	² ₉	4	6	5
4	6	3	¹ ₇	5	^{7 9}	2	¹ ₈	^{8 9}

考虑这个示例。这个盘面可不可以形成一个 BUG 致死形式呢？不可以。为什么呢？因为观察 **r7**，你会发现其中有三个有 29、39、49 这样排布候选数的单元格，这便让数字 9 直接在 **r7** 上出现了 3 次。而它们都是双值格，很明显不可能在这三格上下手寻找真数。所以我们无法让此处 **r7c6** 的其中任意一个候选数去掉之后，形成 BUG 致死形式。因此这个盘面也不可以直接构成 BUG。再来看一个示例。

9	5	2	⁴ ₈	⁴ ₈	1	7	3	6
^{1 3}	^{1 3}	8	5	7	6	2	4	9
4	7	6	9	2	3	1	8	5
7	8	⁴ ₃	⁴ ₃	9	2	5	6	1
6	9	1	7	³ ₈	5	4	2	³ ₈
2	⁴ ₃	5	1	6	⁴ ₈	9	7	³ ₈
⁵ ₃	2	⁴ ₃	6	^{4 5} ₃	9	8	1	7
8	6	9	2	1	7	3	5	4
¹ ₅	¹ ₄	7	³ ₈	⁵ ₈	⁴ ₈	6	9	2

那么这个示例又能不能形成致死形式呢？也不行。因为 **r7c5(3)** 看似在 **r7** 上确实出现了三次，但 **c5** 只有两个 3，所以就这一点而言，如果进行假设时，把 **r7c5(3)** 去掉的话，**c5** 的 3 就直接出数了，破坏了 BUG 原有的模样，所以这也不属于 BUG 的盘面。

7-4-3 区块类型 (BUG Type 2 / BUG + n)

7-4-3-1 同数 BUG + 3

5	4	² 7	6	² 7 8		9	1	3
^{2 3}	9	6	^{2 3}	1	4	7	8	5
³ 7	8	1	³ 5 7	5	9	2	6	4
² 9	1	8	² 9	4	3	5	7	6
^{7 9}	5	² 7	^{8 9}	6	¹ 8	3	4	^{1 2}
4	6	3	⁵ 7	² 5 7	¹ 7	8	9	^{1 2}
8	2	4	1	9	5	6	3	7
6	7	9	4	3	2	1	5	8
1	3	5	^{7 8}	^{7 8}	6	4	2	9

如图所示，全盘仅剩下三个三值格，我们把疑似真数的候选数找出来，它们分别是 $r1c5(7)$ 、 $r3c4(7)$ 和 $r6c5(7)$ 。此时，把它们仨同时去掉，此时盘面的结构就一定会形成 BUG 致死形式（上面罗列的那两点要求）。

导致 BUG 致死形式就意味着，结构不应直接存在于盘面之中；反过来， $r16c5(7)$ 和 $r3c4(7)$ 也就都应该为真数。那三个真数就意味着都是正确的吗？显然不是。多个真数的 BUG 结构，真数应该是“都不可同时消失”，而不应该是“真数全部都正确”。换句话说， $r16c5(7)$ 和 $r3c4(7)$ 里，至少有一个是正确的填数。那么不管谁是正确的，它们都看得到的地方（共同对应的地方），就不应该是正确的。即 $r3c5 \neq 7$ 。否则， $r3c5 = 7$ 会同时让所有的三个真数全部消失，违背推导的结论。

之所以说“不可同时消失”而不是“全部正确”的原因很简单：真数全部消失就使得盘面直接形成 BUG 的致死形式，所以假设错误，所以原假设“同时消失”的相反情况才是结果，即“不可同时消失”。

这个结构和之前介绍的 UR 里的“类型 5”一样，它属于类型 2 是因为可以看成“广义的区块”，所以这个示例属于区块类型。

虽然这里看似在 c5 里，7 出现了四次，已经不满足要求了，不过之前说的出现三次，是针对于每一个真数讲的，如果这一列恰好有两个相同的数字作为真数的时候，这个数确实会在此列出现 4 次。不过，要想找到这些真数，依然得从多值格里去寻找，因为双值格里是根本不可能产生真数的。

而且，看到技巧名后面的“BUG + n”了吗？由于 BUG 的特殊性，所以这个技巧有一个特殊的名字，叫做 BUG + n。这个 n 值表示的是真数的总个数，比如上述例子里，有三个真数，就称为 BUG + 3。当然，我们也可以把标准类型的 BUG 称为 BUG + 1。

实际上，UR 也有类似的逻辑（UR + n），但在 UR 里我们很少这么用，因为它的结构基本上已经固定了是分属于两个宫里的四个构成矩形的单元格，而并不是像 BUG 结构一样，真数的位置可能发生任意的变化。

最后啰嗦强调一下，“真数至少有一个正确”是包含“所有真数同时正确”的情况的，所以分析的时候，不要漏掉情况，否则会导致思维不严谨，从而导致结构结论的判定错误。

接下来我们再来看一些示例。

7-4-3-2 异数 BUG + 2

<small>7 8</small>	1	5	9	6	3	<small>4 8</small>	2	<small>4 7</small>
3	4	2	7	1	8	9	5	6
6	<small>7 8</small>	9	5	2	4	<small>3 1</small>	<small>1 3</small>	
<small>4 9</small>	3	7	6	8	5	2	<small>1 9</small>	<small>1 4</small>
<small>4 2</small>	<small>2 5</small>	6	1	9	7	<small>4 3</small>	8	<small>4 5</small>
<small>8 9</small>	<small>5 8</small>	1	4	3	2	7	6	<small>5 9</small>
1	<small>2 7</small>	8	3	4	6	5	<small>7 9</small>	<small>2 9</small>
5	6	4	2	7	9	1	3	8
<small>2 7</small>	9	3	8	5	1	6	4	<small>2 7</small>

如图所示，我们找到了全盘的所有真数，一个是 $r4c9(9)$ ，而另外一个则是 $r5c9(4)$ 。可以知道的是，真数不可全部同时消失，所以 $r4c9(9)$ 和 $r5c9(4)$ 至少都得有一个数是成立的。那么我们就需要找到一个删数，使得它们都能够排除掉，这样的数即为结果。

可以发现， $r4c9(4)$ 就是这样的数。如果 $r4c9 = 4$ ，显然 $r4c9 \neq 9$ ，而且它也能使得 $c9$ 里其余位置不能填入 4，所以 $r5c9 \neq 4$ ，这样就使两个真数都消失了。虽然它破坏了 BUG 结构，但实际上我们得到的结论“两个真数至少有一个成立”这一点并不依赖于 BUG，所以结论依旧可以使用，而且它的这般推理已经使得真数全消失了，使得出现矛盾。所以， $r4c9 \neq 4$ 。反过来理解也可以：可以发现，所有真数都能排除 $r4c9(4)$ ： $r4c9(9)$ 填入后一定会使得 $r4c9 \neq 4$ （同一个单元格显然不能放下两个数）；而 $r5c9(4)$ 填入后也一定会使得 $r4c9 \neq 4$ （同一列不能填两个 4）。所以呢，由于所有真数全部都能排除掉 $r4c9(4)$ ，故 $r4c9 \neq 4$ 。

这种结构巧妙的地方依然是利用了前一个示例里的逻辑，找到三个真数都能排除的候选数，因此它依然被放到了区块类型里，因为区块类型的核心思想就是“找交集”。这个例子有两个真数，所以称为 BUG + 2。

7-4-3-3 异数 BUG + 2 的另一则示例

2	1	3	4	5	6	8	7	9
		6		3	1	2	4 5	4 5
		7						
4	4 5	5	2		9	6	1	3
		7		7 8				
6	7	4	3	2	5	9	8	1
5	2	1	6	9	8	4	3	7
3	8	9	1			5	6	2
7	3	5 6		4	2	1	9	4 5 6
		5 8		8				
4		6	8	5	1	3	2	4 5 6
		9		7				
1	4 5	2	9	6	3	7	4 5	8

如图所示，这一个示例和刚才的例子的逻辑几乎完全一样，不过它单独被我列出来是因为，这个例子的真数不好被发现。因为 4 和 5 在 c9 都是三次，而这一点并不能断言真数 4 和 5 到底在 r78c9 的哪一个单元格之中。

我们此时就得按行来分析。我们要找的是每一个区域都出现 3 次的数，所以既然列和宫都区分不了，那么只能看行。r7 里数字 5 出现了三次，所以 r7c9(5)才是真数，所以相对地，r8c9(4)是真数。

找到这一点后，我们就可以使用刚才的逻辑来得到结果：r7c9 <> 4、r8c9 <> 5。这一点就自行分析了，我在此处就不需要过多阐述，因为这里的删数逻辑和上一例的是完全一样的。这个技巧依然用了两个真数，所以也是 BUG + 2。

7-4-3-4 真数在同一个单元格的 BUG + 2

4		6	1	4	6	7	2	5	8	3
		9								
5	2	7		3		1		4	6	9
				8						
3	4	6	8	9	4	6	5	1	7	2
		5	6		3	2	1	3		4
7	9			5		8		7	8	
4		1	2			9			3	5 6
8	4 5	3		1 2	4	6	7	9	1 2	5 6
1	3	9		2	2		4	6	5	7
				8	8					
2	7	4		1	6	5		3	9	8
6	8	5	7	3	9	2	4	1		

如图所示，这个例子的逻辑和之前的完全一样，不过它长相特殊，所以我们再次叙述一下这个题目的逻辑。

首先，这个题目的真数有两个，但全部产生于同一个单元格 **r5c4**，真数是 **r5c4(68)**。显然，真数一旦出现，就意味着它们不可同时消失。也就是说这个单元格里要么 **r5c4 = 6**，要么 **r5c4 = 8**。所以 **r5c4** 的填数就跟除了 6 和 8 以外的数字没有关系了，因此可以安全地删除掉它们。

7-4-3-5 异数 BUG + 3

9	1	8	4	²	³	³ ₆	² ₆	5
6	² ₃	4	9	5	8	¹ ₃	¹ ₂	7
5	² ₃	7	6	¹ ₂	¹ ₃	4	8	9
¹ ₄	5	2	7	9	¹ ₆	8	¹ ₄ ₆	3
¹ ₄	8	6	5	3	2	9	7	¹ ₄
7	9	3	8	¹ ₆	4	2	5	¹ ₆
3	7	5	2	8	9	¹ ₆	¹ ₄ ₆	¹ ₆
2	6	9	1	4	5	7	3	8
8	4	1	3	⁷ ₆	⁷ ₆	5	9	2

如图所示，这一示例里，所有真数位于 **r4c8(1)**、**r7c8(6)**和 **r7c9(1)**。它们不可同时消失，即至少有一个数是正确的填数。那么，首先 **r4c8(1)**和 **r7c9(1)**都能立刻删除掉 **r7c8(1)**；而 **r7c8(6)**成立时也能够立马删除掉 **r7c8(1)**(同一个单元格不能放两个数)，所以 **r7c8(1)**显然就是这三个真数都能排除的候选数了，因此 **r7c8 <> 1**。

这个例子是三个真数了，所以是 **BUG + 3**。可以看到，这则示例就比较灵活一些了，真数甚至可以涉及不同的数字，也可能存在删数的结果。

7-4-3-6 异数 BUG + 3 的另一则示例

3	7	9	4	² ₅	² ₅	1	8	6
6	2	5	¹ ₉	8	¹ ₉	4	7	3
1	8	4	7	6	3	5	2	9
9	⁵ ₆	7	¹ ₂ ⁵ ₈	² ₅	4	3	¹ ₆	² ₈
⁴ ₈	⁵ ₆	1	² ₅ ⁸	3	7	9	⁴ ₆	² ₈
⁴ ₈	3	2	6	9	¹ ₈	7	¹ ₄	5
7	9	8	3	4	6	2	5	1
2	4	6	⁵ ₉	1	⁵ ₉	8	3	7
5	1	3	² ₈	7	² ₈	6	9	4

如图所示。这则示例里，有三个真数，但只分属于两个单元格。真数分别是 **r4c4(25)** 和 **r5c4(8)**。它们不可同时被去掉。

那么，显然 **r4c4(8)** 是三个真数都能排除的候选数，因此，**r4c4 <> 8**。

7-4-3-7 异数 BUG + 4

2	7	1	⁴ ₈	6	5	⁴ ₈	9	3
⁴ ₈	9	3	7	1	⁴ ₈	6	5	2
6	⁴ ₅	⁴ ₅ ⁸	9	2	3	1	7	⁴ ₈
1	8	7	3	4	9	5	2	6
⁴ ₈	³ ₄	³ ₄	2	5	7	6	9	8
5	6	9	1	8	2	3	4	7
7	1	6	⁴ ₈	5	⁴ ₈	2	3	9
⁴ ₈	³ ₄	³ ₄	2	9	1	7	6	⁴ ₈
9	2	⁴ ₈	6	3	7	⁴ ₈	1	5

如图所示，这个例子是 **BUG + 4**，即有四个真数，而推导的逻辑在之前已经反复提到了，所以该示例请你自行进行推理。这则示例里的所有真数分别是 **r3c3(4)**、**r8c12(4)** 和 **r8c3(8)**。

7-4-4 数组类型 (BUG Type 3)

因为 BUG 里的真数的位置的特殊性，它的位置是任意的，所以它的数组类型可能比普通 UR 类型的数组类型更难理解透彻。

7-4-4-1 BUG + 显性数组 (BUG Type 3 With Naked Subset)

BUG + 显性数对 (BUG Type 3 With Naked Pair)

4	5	6	1	7	2			3
9	8	7	3	4	5	6	1	2
1	2	3	8	6	9	4	5	7
7	4	5	6	2	8	3	2	1
6	1	4	7	2	3	4	2	5
6	3	2	4	5	1	7	6	5
2	4	4	5	3	6	1	7	9
5	6	1	9	8	7	2	3	4
3	7	9	2	1	4	5	7	6

如图所示，发现该例子里所有的真数一共只有两个：**r4c8(9)**和**r9c8(8)**。它们不可同时被去掉；而如果真数全部都成立（显然是可能的，因为“不可同时消失”包含了“全部都成立”的可能）的话，**r1c8**将无法填入任何一个合适的数字，所以矛盾。所以，实际上，真数必须得只有一个是成立的，不能多，也不能少。

不论是 8 还是 9，都可以和 **r1c8** 形成一个待定的数对结构，我们能确定 **r189c8** 里必然会有一个 8 和一个 9，所以其余位置都不能再填入 8 和 9 了，因此可以删除掉它们。

我们再来看一则示例。

9	2	5	3	6	1	4	7	8
<small>1 4</small>	<small>1 4</small>	3	<small>7 8</small>	<small>7 8</small>	2	5	6	9
<small>6 8</small>	7	<small>6 8</small>	4	9	5	1	2	3
<small>2 5</small>	8	<small>4 7</small>	<small>2 9</small>	3	<small>4 6 7 9</small>	1	<small>5 6</small>	
3	6	9	1	5	7	2	8	4
<small>2 5</small>	<small>1 4</small>	<small>1 4</small>	<small>2 9</small>	<small>4 6</small>	<small>7 9</small>	3	<small>5 6</small>	
<small>4 6</small>	9	<small>4 6</small>	<small>7 9</small>	1	3	8	5	<small>2 7</small>
7	5	<small>1 8</small>	6	<small>4 8</small>	<small>4 8</small>	3	9	<small>1 2</small>
<small>1 8</small>	3	2	5	<small>7 8</small>	9	6	4	<small>1 7</small>

如图所示，这一则示例里一共有三个真数：**r6c36(4)**和**r6c4(2)**。

有三个真数就意味着我们需要讨论四种情况：有 0 个真数成立、有 1 个真数成立、有 2 个成立和有 3 个成立。

显然，有 0 个成立是不可能的，因为它们全部去掉了，使得出现致死形式；全部成立也是不可能的，因为真数里包含相同的数字，还同一行，这样一定会产生矛盾；所以还剩下两种情况，有 1 个或有 2 个真数成立。

如果有两个真数成立的话，那么只可能是 **r6c3** 或 **r6c6** 其一的候选数 4 和 **r6c4(2)** 一同成立了，这是不可能的，因为 **r6** 有一个单元格只有候选数 2 和 4——**r6c5**，如果这两个真数同时成立将立马使得 **r6c5** 无法填入数字。所以这样也是矛盾的。

所以最终的结果就只有一种情况：三者只有一个才是正确的数字。不论是谁成立，都可以和 **r6c5(24)**形成待定的显性数对结构，使得 **r6** 里一定的 2 和 4 一定只会出现在 **r6c3456** 里，所以其余位置都不能够放 2 和 4，删除掉它们。

这两则示例都是通过 BUG 带上待定的显性数对来使用的，即都绑定了数组，所以它们属于数组类型，即类型 3。

接下来我们来看一则带显性三数组的示例。

BUG + 显性三数组 (BUG Type 3 With Naked Triple)

2	9	1	8	5	3	4	6	7
3	4	7	1	2	6	8	5	9
8	5	6	4	9	7	1	2	3
9		6 4	3	1	4 5	5 6		2
5	1		3	2	6	8	9	4
1	2	2	9	7	4 5	3	3	1
4		6 4				5 6		6
	8	2	6	3	1	2	9	5
6	1 2	5	7	4	9	2 3	1 3	8
1	3	9	5	8	2	6	4	1
7								6

如图所示，示例一共存在两处真数，一个是 r6c3(4)，另外一个 r6c7(6)。依然按照有 0 个、1 个、2 个真数是正确的填数这样三种情况讨论，发现有 0 个真数一定违背要求（形成 BUG 致死形式）、有 2 个也会违背要求（使得 r6c19 无法填数，或者可以理解为 r6c1379 四个单元格只剩下 1、4、6 三种不同的候选数，填不满四个单元格）。所以，只有一种情况成立：只有一个真数是对的。

那么，不论 4 还是 6 对，r6c1379 里必然有一个 1、一个 4 和一个 6（构成显性三数组形式），所以删除掉 r6 其余单元格的 1、4、6。

BUG + 显性四数组 (BUG Type 3 With Naked Quadruple)

7 8	1	5	9	6	3	4 8	2	4 7
3	4	2	7	1	8	9	5	6
6	7 8	9	5	2	4	3 1 8 7	1 3	
4	3	7	6	8	5	2	1 4	9 3
2 4	2 5	6	1	9	7	4 3 8	4 5	3
	5 8 9	1	4	3	2	7	6	5 9
1	2 7	8	3	4	6	5	7 9	2 9
5	6	4	2	7	9	1	3	8
2 7	9	3	8	5	1	6	4	2 7

如图所示，这个例子有些复杂的是，它所带的数组的规格有点大。首先，我们依然按照真数有 0、1、2 个成立的情况来分析。有 0 个必然出现致死形式，而有 2 个成立有立马会让 c9 出现五个单元格只能放下 2、4、7、9 四种数，显然是不够放的，有一个单元格必然会空出来，所以也是违背要求的。

所以，只能有一个真数是对的，那么，r14579c9 里会产生一个 2、4、7、9 的显性四数组结构，进而得到删数（c9 里其余单元格的 2、4、7、9）。

BUG + 显性五数组 (BUG Type 3 With Naked Quintuple)

恐怖的是，BUG 的结构过于特殊，使得**显性五数组** (Naked Quintuple) 出现在了结构里。这是我们第一次使用到显性五数组来做题。

7	3	9	1 8	6 8	1 6	2	4	5
1	6	4	3	2	5	8	7	9
8	2	5	9	4	7	3	1	6
3 9	7	8	1 4	3 9	2 4	6	5	1 2
3 6	4	2	1 7	5 6	3 1	9	8	1 7
6 9	5	1	7 8	6 8 9	2 9	4	3	2 7
4	1	6	2	7	3	5	9	8
5	9	3	6	1	8	7	2	4
2	8	7	4 5	5 9	4 9	1	6	3

如图所示，结构里包含 r5c4(1)、r5c5(6)和 r6c5(9)三个真数。按照有 0 个、1 个、

2 个、3 个真数来讨论填数情况：

- **有 0 个真数是对的**是违背要求的，因为这样一定会使得题目形成 BUG 的致死形式；
- **有 3 个真数是对的**是违背要求的，因为全部都对的话，1、2、4、6、9 五个不同的数字将放在 7 个单元格里，这样会剩下两个单元格放不下数字；
- **有 2 个真数是对的**是违背要求的，因为不论哪两个数对，1、2、4、6、9 都无法放下 6 个单元格，总会有一个单元格会空出来。

所以，最终我们就只有一种情况：即三个真数里只有一个是对的。

不论是谁对，这样就恰好会在 r4c46、r5c45 和 r6c5 的其一、以及 r56c6 里产生一个显性五数组结构，所以其余单元格都不能再填入数字 1、2、4、6、9，删掉它们。

那么，显性数组的版本我们就讲完了。接下来我们再来看几则带隐性数组的版本。

7-4-4-2 BUG + 隐性数组 (BUG Type 3 With Hidden Subset)

BUG + 隐性三数组 (BUG Type 3 With Hidden Triple)

3	7	1	4	6	5	9	8	2
² 4	9	^{4 5}	² 7	1	8	⁵ 7	3	6
6	² 5	8	² 7	9	3	1	4	⁵ 7
1	^{5 6} 9	⁵	3	^{7 8}	2	4	⁶ 7	^{8 9}
² 8	3	7	9	4	6	² 5	1	⁵ 8
² 4	² 6	⁴ 9	5	^{7 8}	1	^{2 3} 7	6	³ 8 9
9	1	2	6	3	7	8	5	4
5	8	6	1	2	4	³ 7	9	³ 7
7	4	3	8	5	9	6	2	1

如图所示，隐性数组的讨论可能更为复杂和坎坷一些。我们依然参照刚才的方式，分情况讨论真数的成立性。

如果有 0 个真数成立必然是会矛盾的；全部真数成立，又会使得 3、4、9 在 r6 里找不到三个单元格放下。因为真数都成立之后，能放下 3、4、9 的只剩下了 r6c37 这两个单元格，这显然是不够放下的，就必然会有一个数是放不到这一行里，所以矛盾。

所以最终我们得到，真数依然是只有一个成立的结论。这样一来，3、4、9 刚好放下三个单元格，于是便形成了隐性三数组结构，故得到删数（此时的删数应从 r6c37 里去找，因为 r6c19 两处我们不清楚哪个真数成立，因此不能冒昧去尝试删数）。

这个结构带了一个隐性三数组。

BUG + 隐性四数组 (BUG Type 3 With Hidden Quadruple)

6	2	5	1	1		3	9	3
	9	1	2	3	4	5	6	
	3	4	5	9	6	2	1	2
2	1	3	9		5	7	4	6
2	7	4	1	1	3	2	5	2
4	5	6	4	2	1	8		3
5	4	9	8	7	2	6	3	1
3	8	7	6	4	1	9	2	5
1	6	2	3	5	9	7		

如图所示，这个例子可怕的地方在于，题目一共存在五个真数：**r5c1(48)**、**r5c5(8)**和**r5c9(23)**。如果这种示例我们按照真数为单位来讨论情况的话，就会很难受了。

我们发现，之前的推理过程是不是都是在跟单元格较劲？因为我们讨论数组的成立不外乎就是多一个单元格填不满，少一个单元格填不够。所以此示例使用单元格来分析将会更高效。

当然，前面的示例也都可以这么做，不过前面的例子里每一个真数都单独占据一个单元格，所以看起来是没有区别的。

如果没有单元格能放真数，显然是出现致死形式；而如果三个单元格放的都是真数，1、2、6、9 一共四个数，就放不下 **r5**，因为能放下 1、2、6、9 的地方只剩下 **r5c47** 两格，显然是不够放的，差了两格；如果两个单元格放的都是真数，也不够，因为还差一个单元格。

所以，我们只能让一个单元格填真数，这样的话，1、2、6、9 就刚好四个单元格了，而且恰好形成隐性四数组，所以可以删除的是 **r5c4(4)**和 **r5c7(3)**。

最后，我们再来看一则带有**隐性五数组 (Hidden Quintuple)**的例子。

BUG + 隐性五数组 (BUG Type 3 With Hidden Quintuple)

2 6 4	2	8	4 6	3	9	1	7	5
5 6	1 5	9	1 7 6	2	6	8	3	4
7	1 4	3	1 4 8	5	1 8	6	2	9
2 3	7	1	2 3	6	4	9	5	8
5 3	9	6	3 5 7	8	3	2	4	1
8	2 5	4	9	1	2 5	7	6	3
4	6	5	1 2	9	1 2	3	8	7
9	8	7	3 6	4	3 6	5	1	2
1	3	2	5 8	7	5 8	4	9	6

如图所示，此图里有三个真数，依然按照单元格（或者是真数）为单位来分析填数情况。最终我们依然能确定下来，只有一个真数成立。

那么，剩下两格就必须填入的是 1、4、5、7、8，此时，此列就恰好包含五个单元格，并构成隐性五数组结构，所以，删除 r19c4 里不是 1、4、5、7、8 的其余候选数（当然了，r9c4 这一个单元格没有删数）。

7-4-4-3 为什么 BUG 里的数组没有隐性数对，却有五数组？

为了描述的逻辑结构更为清晰，我们先来说说看，为什么有五数组。

五数组之所以存在于 BUG 里，是因为它的互补并非严格的，和之前的数组不同，BUG 的真数是可以放在任意位置的，这便使得套用数组逻辑的时候，完全可能让真数放同一个单元格，导致数组形成的时候需要少判定一个单元格。我来举例说明这个说法。

6	2	5	1 7	1 8	7 8	4 3	9	4 3	6	2	5	1 7	1 8	7 8	4 3	9	4 3
7 8	9	1	2	3	4	5	6	7 8	7 8	9	1	2	3	4	5	6	7 8
4 7	3	4 8	5	9	6	2 8	1	2 7	4 7	3	4 8	5	9	6	2 8	1	2 7
2 8	1	3	9	6 8	5	7	4	2 6	2 8	1	3	9	6 8	5	7	4	2 6
4 9	7	4 8	1 4	1 6	3 8	2 3	5	2 3	4 9	7	4 8	1 4	1 6	3 8	2 3	5	2 3
4 9	5	6	4 7	2 7	3	1	8	3 9	4 9	5	6	4 7	2 7	3	1	8	3 9
5	4	9	8	7	2	6	3	1	5	4	9	8	7	2	6	3	1
3	8	7	6	4	1	9	2	5	3	8	7	6	4	1	9	2	5
1	6	2	3	5	9	4 8	7	4 8	1	6	2	3	5	9	4 8	7	4 8

如左图所示，这是我们刚才介绍的隐性四数组的示例，实际上，你可以看互补视角，转

变为 2、3、4、8 的显性四数组结构。而实际上，这里不论是显性数组还是隐性数组，都共用了真数涉及的单元格；而真数大多挤在同一个单元格里。如果看互补，就是右图这样，可以发现，实际上它少删除了一个数字，即 **r5c7(3)**，此时这个 3 被当作结构使用了，这一点和我们之前的逻辑并不相同。

下面我们再来说说，为什么没有隐性数对。

隐性数对在之前 **UR** 和 **AR** 甚至是 **UL** 之类的技巧里，都出现过，但是为什么 **BUG** 不行了呢？是没有示例还是根本就不可能存在隐性数对呢？

答案是，根本不可能存在。下面我们来简单证明一下。

首先，我们来思考有两个单元格放真数的情况。如果一个盘面有若干真数分布到两个单元格里，如果是隐性数对，就必须意味着外部还存在一个单元格与之构成隐性数对。例如下图。

			1 2 3					
			1 2					
			1 2					

如果全盘一共有两个真数，且都在 **c4**，一个是 **r3c4(3)**，另外一个为 **r4c4(1)**。

那么为了保证隐性数对的出现，我们必须在 **c4** 的其余单元格里加入一个单元格（此时隐性数对应当是 1 和 2，因为真数所在的单元格的额外候选数只有 1 和 2）。此时我们假定这个单元格在 **r5c4**，且只有 1 和 2，那么 OK，结构成立了。但实际上你可以发现，**r45c4** 是一个显性数对，它的存在破解了 **BUG** 的出现，所以两个单元格有真数的情况肯定是不可能出现了。

那么更多呢？比如三个单元格呢？思路完全是一样的，所以我们就不讨论了。因此，实际上可能的只有一种情况：所有真数都只在一个单元格里出现。

考虑如下情况。如果只有一个单元格能放下真数，那么构成隐性数对的几率就会更大一些，不过……

			1	2	3			
			1	2				

如图所示，假定全盘只有两个真数是 **r4c4(23)**，这样的话，要想构成隐性数对，那么额外的数字 1 就必须和 **c4** 里的另外一个单元格形成隐性数对。这样就 OK 了。不过，真数需要有一个是成立的，这一个单元格必须被占据，所以实际上额外的候选数是必须被我们删除的，所以这显然也不可能构成隐性数对；而从另外一个方面来看，即使 **r4c4(1)** 可以存在，那么构成的两个单元格 **r4c4** 和 **r5c4** 实际上已经自动地形成了隐性数对，而这一点其实跟 BUG 毫无关系。所以，一个单元格放真数也不可能存在隐性数对。

所以，实际上，BUG 的数组类型是不可能带隐性数对来推理的。

7-4-5 共轭对类型（BUG Type 4）

4 7	8	1	3	5	2	4 6	2	6
4 7	5	2	4 9	6	8	1 4	3	3
9	3	6	2 4	7	1	2 4	5	8
1 3	6 9	5	8	1 2	6 9	7	2 3	4
2	4	8	5 7	3	5 7	1 6	1 6	9
1 3	6 9	7	6 9	1 2	4	2 3	8	5
6	7	9	1	8	3	5	4	2
8	1	4	2 5	9	2 5	3 6	7	3 6
5	2	3	7 6	4	7 6	8	9	1

找到全盘的所有疑似真数，然后去掉看是否构成 BUG 致死形式。结果这个例题比较神奇的是，不同的真数可以在同一个单元格内。这个例子有四个真数：**r1c7(26)**和 **r2c7(34)**。

如果它们全部去掉，的确会形成 BUG 致死形式，所以它们都应该是货真价实的真数。

那么，至少都有一个真数是正确的。但是，c7 有 9 的共轭对，意味着 r12c7 有且仅有一格是填 9 的。因为 9 不是真数的缘故，所以其中一格是 9，那另外一格就必然只能填真数了。所以这样一来，就跟这两格不是真数或 9 的其余候选数没有任何关系了，因此可以安全地删除掉它们。

一般来说，共轭对类型的 BUG 结构一般都会产生与之匹配的数组类型（也就是说类型 4 的结构一般都会有类型 3 的身影），比如这个示例，看起来好像没有，实际上你可以对照之前的类型 3 去找找，它实际上带有一个规格稍大一些的数组结构。

而正是因为如此，所以共轭对类型的示例少之又少，这里仅仅找到了一个看起来比较不像数组类型的示例作为介绍。

7-4-6 待定 BUG (Almost BUG)

下面介绍一种新奇的使用模式，叫做**待定 BUG (Almost BUG)**结构。

7-4-6-1 可能形成 BUG 的 ABUG

1	7	4	8	3	2	5	9	6
5	9	3	4	6	1	2	7	8
6	8	2	9	5	7	³ ₄	³ ₄	1
² ₈	6	7	5	^{1 2} _{4 8}		9	^{1 2 3} ₄	³ ₄
⁴ ₈	² ₈	1	9	⁴ ₈	3	6	⁴ ₄	5
² ₄	3	5	^{1 2} ₄	9	6	8	^{1 2} ₄	7
3	² ₄	1	6	² ₈	⁴ ₈	7	5	9
9	² ₄	8	^{1 2} ₄	7	5	^{1 3} ₄	6	³ ₄
7	5	6	¹ ₄	3	9	¹ ₄	8	2

如图所示，这是一个待定的 BUG 盘面。

之所以说它是“待定的”，是因为这里盘面确实不是合格的 BUG 盘面。数字 4 在 r5 里确实出现了三次，也包含了三值格一格 r5c1，但 c1 却没有出现三次 4，宫内也没有；同样地，数字 2 反而在 c1 出现了三次，也包含三值格；但数字 2 在 r5 和 b4 里也没有出现三次，也都只有两次。这已经不满足 BUG 的要求了。但是现在为了用它，我们发现一个神奇的单元格 r5c8：如果 r5c8 = 2，它就会破除掉 4 在 r5 出现三次的影响，而 c8 上的其余 2 也都会被删除，即使此时看起来好像 c8 里的 2 出现了三次。

在一通推导后的删数结论后，盘面上刚才所有涂色的候选数就变成了这样。

1	7	4	8	3	2	5	9	6
5	9	3	4	6	1	2	7	8
6	8	2	9	5	7	³ ₄	³ ₄	1
² ₈	6	7	5	^{1 2} _{4 8}		9	^{1 3} _{4 3}	
⁴ ₈	1	9	7	⁴ ₈	3	6	2	5
² ₄	3	5	^{1 2} _{1 2}	9	6	8	¹ ₄	7
3	² ₄	1	6	² _{8 4 8}		7	5	9
9	² ₄	8	^{1 2} _{1 2}	7	5	^{1 3} _{1 3}	6	⁴ ₃
7	5	6	3	¹ ₄	9	¹ ₄	8	2

细数所有单元格和候选数，全部出现两次，并且所有单元格也均为双值格，至此，形成 BUG 形式。故原假设错误，即最终结论为 $r5c8 \neq 2$ 。

7-4-6-2 可能形成 BUG + 1 的 ABUG

4	¹ ₈	7	⁵ ₈	6	¹ ₅	3	2	9	4	¹ ₈	7	⁵ ₈	6	¹ ₅	3	2	9
6	² ₈	9	² _{7 8}	^{2 3} _{7 8}	^{2 3} _{2 3}	4	5	1	6	² ₈	9	² _{7 8}	^{2 3} _{7 8}	^{2 3} _{2 3}	4	5	1
5	^{1 2} _{1 2}	3	4	^{1 2} _{1 2}	9	6	8	7	5	^{1 2} _{1 2}	3	4	^{1 2} _{1 2}	9	6	8	7
7	9	5	6	^{1 3} _{1 3}	^{1 3} _{1 3}	2	4	8	7	9	5	6	^{1 3} _{1 3}	^{1 3} _{1 3}	2	4	8
8	4	6	² ₇	² ₇	^{2 5} _{2 5}	9	1	3	8	4	6	² ₇	² ₇	^{2 5} _{2 5}	9	1	3
1	3	2	9	4	8	5	7	6	1	3	2	9	4	8	5	7	6
9	6	4	1	5	7	8	3	2	9	6	4	1	5	7	8	3	2
3	5	1	² ₈	² ₈	6	7	9	4	3	5	1	2	8	6	7	9	4
2	7	8	3	9	4	1	6	5	2	7	8	3	9	4	1	6	5

如左图所示，这个盘面是否是一个合格的 BUG 盘呢？不是。因为 $c4$ 里只有三个 2，而真数在此列里会产生两个 ($r2c4(2)$)。这样一来的话， $r5c4$ 无论如何都选不出真数。选 2 的话， $c4$ 就两个 2 了，扣掉两个真数后，此列就只剩下唯一的一个 2，盘面“瓦解”……

所以它不是合格的 BUG 盘应该有的样子。这种差一点的 BUG 又怎么用呢？如果我们尝试这样做：如右图所示，我们率先假设 $r8c4 = 2$ ，那么 $r8c5 = 8$ 是显然的，于是对方 2 和 8 的影响和删数也就顺带得到了。此时删除掉影响的数字后，盘面变成一个标准的 BUG 标准类型的局势：全盘空格都是双值格，只剩下一个三值格；其次，三值格位于 $r2c5$ ，出现三次的数字是 2， $r2$ 、 $c5$ 和 $b2$ 里 2 都是恰好三个。那么，根据 BUG 的使用规则，这个 2 应当是 $r2c5$ 的真数。

还没完呢。我们是通过假设 $r8c4 = 2$ 得到的 BUG 盘面，并得到 $r2c5 = 2$ 的结论

的。那么，这个推理链有何用处呢？ $r2c5 = 2$ 是我们的结论，这就印证了一点：如果题目唯一解，那么题目就可以依靠 $r2c5 = 2$ 这一个结论继续往下做，直到全盘结束，得到答案。而这个题目是我给大家的示例，我能保证题目是唯一解的。所以我们可以这么去认为了：一旦盘面能做到 $r2c5 = 2$ ，那剩下的空格的填数就能立马顺次得到，并出现唯一的结果，而过程无需证明为什么，也不需要你手动根据 $r2c5 = 2$ 后还一个一个把数字填入到单元格里。说清楚明白一些，就是我们一旦得到 $r2c5 = 2$ 的结论，因为题目是唯一解的，所以后续的步骤我们都可以不用做就直到题目已经可以完成了。因此，既然题目能通过假设的 $r8c4 = 2$ 得到 $r2c5 = 2$ 的结论，那么原来的假设 $r8c4 = 2$ 也就必须是成立的。题目应当是唯一解的，这就意味着能推理得到 $r2c5 = 2$ 这一结论的原假设就应该是成立的（毕竟推理是从假设到 $r2c5 = 2$ 来的）。所以， $r8c4 = 2$ 是这个技巧的结论。

那么到这里，致命结构的基础部分就介绍完毕了。接下来我们将进入新的一种数独技巧里面来。

7-4-7 BUG 最少多少个单元格？

一个合格的 BUG 结构一共最少只会涉及多少个单元格呢？这个答案是 11，而这个答案则是通过枚举或构造结构而得到的。一定要注意，BUG 的内部是无解的。

⁴ ₃	2	7	⁴ ₃	1	9	8	6	5	³ ₆	2	8	¹ ₃	9	5	7	¹ ₆	4	
8	9	1	7	5	6	2	4	3	4	7	1	2	6	8	5	9	3	
⁴ ₅	³ ₆	6	⁵ ₃	8	2	⁴ ₃	9	1	7	³ ₆	5	9	7	¹ ₃	4	¹ ₆	8	2
1	4	2	9	7	8	3	5	6	5	4	7	8	¹ ₃	2	¹ ₆	³ ₄	¹ ₆	9
6	5	8	2	3	1	7	9	4	9	8	6	¹ ₃	4	7	¹ ₃	2	5	
7	3	9	⁴ ₅	6	⁴ ₅	1	8	2	1	3	2	6	5	9	4	7	8	
2	1	4	6	9	7	5	3	8	7	1	5	9	8	3	2	4	6	
9	8	⁵ ₃	⁵ ₃	4	2	6	7	1	8	6	3	4	2	1	9	5	7	
⁵ ₃	7	6	1	8	⁵ ₃	4	2	9	2	9	4	5	7	6	8	3	1	

如图所示，这是两个只有 11 个单元格的 BUG + 1 结构。

不过，BUG 最多涉及多少个单元格，目前尚未得到确切的答案，不过我们依然在不断努力寻找最大情况。

Part 8 待定数组 (Almost Locked Set)

在之前我们经常提起数组，还说数组可以出现“待定”的状态，那么现在我们将学习一种新的技巧板块，叫做**待定数组** (Almost Locked Set, 简称 ALS)。

8-1 欠一数组 (Almost Locked Candidates)

有一种区块，叫做待定的区块；而有一种数组，就是从这种区块衍生所得到的数组。那么我们来看看，这个结构究竟是如何的推理过程。

8-1-1 欠一数对 (Almost Locked Pair)

8-1-1-1 基本构型 (ALP Basic Type)

4	2	2	6	6	3	5	1	7
9		9	8	8				
1	3	5	4	2	7	8	6	9
8	6	7	9	1	5	4	2 3	2 3
6	9	3	5	4	8	2	7	1
7	1	8	3	6 9	2	6 9	5	4
2	5	4	1 6 7	6 9	1 6 9	3 6 9	3 8	3 6 8
3	2	1 2	1 2	3	4	7	2 3	5
9	8	9	8	8 9			8	
5	2	1 2	1 2	3	1	1 3	2 3	6
7	8	6	6	7 8 9	6	6	4	8
3	2	1 2	1 2	7 8 9	1	1 3	2 3	6
4	4	6	6	7 8	5	6	9	8

如图所示，如果 $r9c6 = 1$ 的话，就会发现， $r7$ 里只剩下 $r7c3 = 1$ ；而如果 $r9c6 = 6$ 的话， $r7c3 = 6$ ，所以不管 $r9c6$ 填什么， $r7c3$ 就填什么。而 $r9c6$ 只能填入 1 或 6，所以 $r7c3$ 就只能填 1 或 6。

而其它的删数又是怎么回事呢？仔细观察 $r7$ 和 $b8$ 的交集，这三个单元格里也有 1 和 6 的位置。如果 $r9c6$ 是 1 的话，在这三个单元格里，必然也会产生一个 6；因为 $r7$ 和 $b8$ 里都只剩下这几个单元格能放下 6 了；同样地，如果 $r9c6$ 是 6 的话，那只有这几个单元格里能放下 1。换言之，不论怎么填，在 $r7$ 和 $b8$ 里都将产生 1 和 6 的数对结构；而拆开来， $r7c456$ 又像是区块一般。所以， $b8$ 里的其余单元格将不能放下 1 和 6；而 $r7c3$ 则是只能放下 1 和 6，其余候选数也应当删除掉，所以总的结果就是图上那样。

这个结构称为**欠一数对** (Almost Locked Pair, 简称 ALP)，不过实际上，因为也可以拆开来，把 $r7c456$ 视为区块理解，所以这个技巧的英文名称则为 Almost Locked Candidates，而 Locked Candidates 就是区块，Almost 即待定的意思，所以技巧名也直译为“待定区块”。接下来我们来看一种 ALP 的变体。

8-1-1-2 扩展构型（ALP Extended Type）

拓展构型理解起来没有标准类型容易，所以我们将给出比较多的例子提供参考。

	3	3	3	7	2	8	4	1	3
	9	5	6	9					5
1	4	5	6	2	9	3	4	5	6
	4	5	6	1	4	6	4	5	6
6	8	4	5		2	3	2	2	3
	2	3	2	2	3	1	2	3	4
4	4	5	4	5	7	9	7	9	4
2	3	1	4	5	2	3	2	2	3
4	9	7	9	6	5	4	6	7	9
7	2	3	3	4	6	2	3	2	3
	9	8	9	5	1	2	3	5	6
4	2	4	6	5	7	8	7	9	4
5	4	2	3	1	2	3	2	2	3
	9	6	6	8	7	8	9	7	6

如图所示，和刚才的推理大致相同。假设 $r9c7 = 2$ ，则因 $c9$ 只能在 $r4c9$ 放下 2，所以此时 $r4c9 = 2$ ；同理的话， $r9c7 = 7$ ，则 $r4c9 = 7$ 。所以 $r4c9$ 的填数只可能放 2 或者 7；同理，在 $c9$ 和 $b9$ 的交集里，还能放下 2 和 7，这使得 $b9$ 里的其余位置都不能填入 2 和 7，因为 $r9c7$ 填一个数，那么这三个单元格里就必然会产生另外一个数的填数。

那么，为什么可以删除 4 呢？因为 $c9$ 里此时因为 $r4c9$ 删除了除了 2 和 7 的候选数，使得数字 4 在 $c9$ 里形成区块，所以 $b9$ 里不能再填入 4，所以 $b9$ 里其余位置都不能放 4，于是删除掉它们。

这个结构除了 ALP 以外，还带有一个 4 的额外区块结构。不过，实际上这种构型并不仅仅这么简单，我们再来看一些例子。

8	5	1	3	4	2	3	1	2	
	2	1		1	6	2	1	6	9
7	9	9	9	8	5	8	5	6	3
4	2	3	6	1	3	2	3	9	5
3	1	8	7	4	2	5	6	6	5
	5	6	7	5	9	3	6	9	9
2	6	4	1	6	5	1	5	6	7
9	8	7	2	1	4	3	5	6	6
	5	6	3	6	5		7	4	1
1	4	2	5	6	3		8	9	7

如图所示，我们率先可以看到，**r1c6** 只有 1 和 6。我们尝试对 **b3** 进行 1 和 6 的位置的查找。此时可以发现，在 **b3** 里能放下 1 和 6 的地方此时只有 **r23c7**。这一点我们依然使用之前的逻辑，假设 **r1c6** 是 1，然后就可以得到 **r23c7** 里必有一个是 1；而 **r1c6** 是 6，也可以得到 **r23c7** 必有一个是 6。不过此时我们依然不能随意下结论。因为目前来说，我们并没有区域能够产生我们需要的 1 和 6 的数组。

不过我们注意到，**b3** 里只有 **r23c7** 和 **r3c9** 这三个单元格可以填入 5 和 8，这便使得 **r23c7** 和 **r3c9** 里必须有一个 5 和一个 8，那么剩余一个单元格放多少呢？显然只能是 1 或者 6 了。因为刚才我们通过逻辑得到，**r23c7** 里必须得有一个 1 或者 6，但现在这两个单元格里必然会有 5 或者 8 的出现，这使得此时 **r23c7** 里只剩下一个单元格可以放 1 和 6 了。显然，如果 **r23c7** 都放下 5 和 8 的话，1 和 6 就没地方可以放了，所以我们可以得到 **r23c7** 里必须放进去一个 1 和 6。

不过，这样一来，5 和 8 出现在 **r23c7** 和 **r3c9** 的其中两个单元格，而剩下的唯一一个单元格又必须是 1 或者 6 的其一，所以这三个单元格显然就不能填入其它的任何候选数了，所以我们可以安全地删除掉它们；与此同时，**r1** 上由于 **r1c6** 是 1 或者 6 的缘故，而 **r23c7** 会根据这一点要求确定出与之相同的填数，所以 **r1c78** 里必须填入的是与之互斥的另外一个数。换句话说，如果 **r1c6** 是 1，那么根据刚才的要求，**r23c7** 里必须有一个 1，而 **r1c78** 里就必须有一个 6 了，因为 **b3** 必须得有一个 6，而 6 能填的位置此时只剩下 **r1c78** 了，**r23c7** 里虽然还剩下一个单元格，但它被留下来放 5 或者 8 了。

接着我们再来一则和上述逻辑完全一样的例子。

3	2	9	1	1	1	5	1	1
			6	4	8		4	6
			8	7	8		7	8
4 5	4	1	1	3	1	1	6	2
8	8	7	8	6	4	5	8	9
			8	7	8	7	8	
4 5	6	1	1	1	2	1	2	3
8	7	7	8	9	7	8	9	7
8			8	9	7	8	9	7
9	3	4	2	1	1	7	6	5
				8	8			
7	1	6		3		5	4	8
				9	9			2
2	5	8	7	6	4	1	1	3
						9	9	
4 6	7	3	1	1	1	1	2	5
8			6	8	9	8	9	8
			8	9	8	9	8	9
1	9	5	4	2	3	2	7	6
				8		6		
4 6	4	2	5	1	1	3	1	1
8	8			7	8	9	9	8

假设 **r5c4** 是 3，则在 **b2** 里能放下 3 的位置只有 **r2c5**；而当 **r5c4** 是 9 的话，则 **r3c56** 里必须有一个是 9。不过我们此时不能得出结论，就需要进一步的逻辑。

此时我们看到，**b2** 里能放 2 和 5 的位置此时只有 **r2c5** 和 **r3c56** 三个单元格。显然，这三个单元格里肯定得放一个 2 和一个 5，那么还剩下的一个单元格就必须按照刚才的规定放下 3 或者 9 的其一。但是很显然，由于这三个单元格只能放下 2、3、5、9 的数字的其三，跟其它的候选数无关，所以 1、4、6、7、8 出现在这三个单元格的候选数情况都可以删除掉；与此同时，由于 **r5c4** 假设的数字的关系，**r23c4** 里就必须有一个单元格填入 3 和

9 的另外一个数了，否则这个数将放不到 b2 里，导致问题的出现。

所以，c4 上依然可以产生一个 3 和一个 9 在 r235c4 三个单元格之中，因此其余单元格的 3 和 9 都可以删除。

5	7		1	4	1	3	9	3	2
2	2	2	8		3	3	3	4	1
4	1	3	9	2	6	5	7	8	7
8	4	5	5	3	1	7	4	6	2
3	4	5	2	4	5	6	1	5	6
2	6	1	2	3	9	8	4	7	5
9	8	2	1	2	3	1	2	3	5
1	2	6	3	4	5	6	7	8	9
1	2	6	3	4	5	6	7	8	9

如图所示，这一个示例和之前的逻辑也都大致相同。只是这里涉及的额外的那些数字从两个变为 3 个。

首先我们观察到的是 r6c9 只能放入 4 和 7，如果假设它为 4，则观察 c7，可以看到 4 和 7 的位置此时只有 r789c7。而仔细观察，c7 里 r5789c7 四个单元格里必须要放下一个 1、一个 2 和一个 8（因为只有这些单元格可以放下 1、2、8），所以四个单元格里最终只剩下一个单元格，但这个单元格实际上又必须放下 4 或者 7，因为这是我们刚才得到的结论。所以，这四个单元格里显然只能放入 1、2、4、7、8 的其四，故其余的候选数都可以被删除；与此同时，r46c7 的其一必须放下 4 和 7 的、和 r6c9 填数互斥的另外一个数。所以，b6 里一定会有 r46c7 和 r6c9 里又一定会出现 4 和 7，所以删除掉 b6 里其余位置的 4 和 7。

这一些示例都比较难理解，而且比较绕。我们最后列出一些例子，希望你能理解并且尝试自己理解它们。

6	7	9	1 2	1 2	5	8	4	3	4	1	2	6	7	2 3	5	3	2 5 6	8	9
8	4	3	7	6	9	1	2	5	5	3	9	4	2	6	8	1	4 7	2	7
2	1	5	3	3	4	7	6	9	2	7	2	1 2	4 5	1 2	3	4	2 3	5 6	7
3	2	1	9	5	6	4	8	7	1	4	5	7	8	9	6	3	4 7	6 9	7
9	8	7	1 2	1 2	3	5	6	1 2	1 2	6	3	1	4	5	6	2	5	7 8	9
5	6	4	1 2	7	9	3	1 2	1 2	7 8 9	6	8	7	8	9	5	4	5 6	7 9	1 2
7	5	3	6	4	5	6	8	9	3	6	5	5 6	7 8	9	1	2 3	1 3	6	4
1	5	3	2	3	7	6	5	4	9	8	1	4	5	6	7	8	9	2	5 6
4	9	2	1 2	3	1 2	3	1	2	7	6	2	4	5	6	8	7	5 6	8	1

149

8-1-2 欠一三数组（Almost Locked Triple）

8-1-2-1 基本构型（ALT Basic Type）

3	1	5 6	5 6	1	6	1	9	7	4	2
2	1 2	2	2	5	1	3	1 2	6	8	3
4	7 9	7 9	7	4	7	4	7	6	8	9
2	2	2	2	2 3	2 3	3	2	3	1	5
4	6	6	4 6	4	6	4	6	9	1	5
7 8 9	7 8 9	7 8 9	7 8	7	7	8	7	8	9	9
2	2 3	2 3	2	2	2	2	2 3	2 3	6	1
4	5	4 5	4	4 5	4	5	7	8 9	6	1
7 8	7 8	7 8	7	7 9	7	9	7	8 9	6	1
1	5	9	8	6	4	2	2 3	2 3	5	4
7	7	7	7	7	7	7	7	7	7	7
2	2	2	1 2	1 2	1	4 5	3	2	5	4
4 6	5 6	4 5 6	4	4	4 5	7 9	7 9	8 9	7	9
7 8	7 8	7 8	7 9	7 9	7 9	7 9	7 9	7 9	7 9	7 9
		1	4	2	4	7	8	5	3	6
7 8 9	7 8 9	7 8 9	7 9	7 9	7 9	7 9	7 9	7 9	7 9	7 9
2	6	4	2 3	3	3	3	5	1	9	7
8	8	8	6	6	6	6	6	6	6	6
5	3	3	1 3	1 3	1 3	1 3	1 3	1 3	1 3	1 3
7 9	7 9	7 9	7 9	7 9	7 9	7 9	7 9	7 9	7 9	7 9

如图所示，这一个示例好像是扩大了“数组”的规格。我们假设 r1c45 分别填入 1、6、8 的其二，而观察 c6 就会发现，此时填入的这两个数只能放在 r79c6，因为其余位置都没有位置可以放下这两个数了。而在 b2 和 c6 的交集里，由于刚才假设的是 1、6、8 的其二，所以在此交集里，必然会出现的是最后剩下的那一个数。所以，b2 和 c6 都会产生三数组结构，也因此，b2 里的其余单元格都不能填入 1、6、8。

这个结构运用了三数组，所以称为**欠一三数组**（Almost Locked Triple，简称 ALT）。

接下来我们再来看一些类型的示例。

8-1-2-2 拓展构型（ALT Extended Type）

5	1	2 3	2	2	2	2 3	4	9
4	2	2	7 8	7 8	7 8	8	7	2
2 3	2	2 3	9	3	1	5	6	8
6	6	6	9	2	4	5	1 3	1 2 3
8	7 8	7 8	7 8	7 8	7 8	7 8	7	7
1 2	2	1 2	1 2	5	9	6	1	3
4	4 5	4	4	5	9	6	4	5
8	7 8	7 8	7 8	7 8	7 8	7 8	7 8	7
1	6	5	1	6	4	5	3	2
8 9	7 8 9	7 8	7 8	7 8	7 8	7 8	7 9	7
1 2	3	1 2	1 2	2	1 2	1	1	5
4 6	8 9	4 6	4 6	5	5	4 6	5	5
8 9	8 9	7 8	7 8	7 8	7 8	8	7 9	7 8 9
1 2 3	6	1 2 3	1 2	2	7	8	1	3
4	4	4	5	5	5	5	5	5
1 2 3	2	1 2 3	1 2	4	1 2	1 2 3	1	3
8 9	8 9	8	8	8	8 9	7 9	7 9	9
7	2	5	6	3	1 2	1 2	1	4
8 9	8 9	8 9	8 9	8 9	8 9	8 9	8 9	8 9

如图所示，可以发现 **r12c7** 里一定放下的是 2、3、7 的其二。显然，此时这两个数在 **r89c7** 里就不会出现了。但是观察 **b9**，可以发现这两个我们假设的数字此时只能放在 **r7c89** 和 **r8c8** 里，而此时这三个单元格里必须得有一个是 5（因为此时 **b9** 里只有这三个单元格可以放下 5）。所以，这三个单元格此时的形式就只能是一个 5 和两个 2、3、7 的数字，但这些数字跟 1、4、6、8、9 这些数字都没有关系，所以自然都可以删除掉；与此同时，和刚才的逻辑一样，**c7** 也可以删除 2、3、7，是因为 **r89c7** 里必有一个 2、3、7，而这个数恰好必须是和 **r12c7** 的两处填数不同的剩下那个数。

6	4	1	⁵ ₇	⁵ ₇	8	3	2	9
8	7	3	2	9	1	6	4	5
5	9	2	³ ₆	³ ₆	4	1	8	7
³	8	^{5 6} _{9 7}	³ _{5 6}	2	³ _{5 6}	4	⁵ _{7 9}	1
¹ _{7 9}	¹	^{4 5 6}	^{5 6} _{7 8}	^{5 6} _{7 8}	9	2	⁵ ₇	3
^{2 3}	^{2 3}	⁵	4	1	³ _{5 7}	8	⁵ _{7 9}	6
^{1 2} _{4 9}	^{1 2} ₅	^{4 5} _{8 9}	^{5 6} ₈	^{5 6} ₈	² _{5 6}	7	3	² ₄
^{2 3}	6	⁴ ₈	9	³ _{7 8}	^{2 3} ₇	5	1	² ₄
^{2 3}	^{2 3} ₅	7	1	4	^{2 3} ₅	9	6	8

这一个示例和之前的示例一样，所以我们就不作过多的描述了。

另外，欠一数组不包含四数组形式，因为欠一数组实际上可以视为区块处理，而区块是无法涉及四个单元格的，所以，所谓的欠一四数组实际上并不存在（或者说即使有结构，也会被其它技巧代换掉，或者是降阶）。

8-1-3 欠一数组的直观视角

可以看到，这些示例的结构都还比较简单，下面我们来介绍一种可以通过直观层面看到的方式。

					3	5	1	7
1	3	5	4	2	7	8	6	9
8	6	7	9	1	5	4		
6	9	3	5	4	8	2	7	1
7	1	8	3		2		5	4
2	5	4						
		1	6	4	7			5
5							4	
				5	1	6		9

我们假设 $r9c6 = x$ （注意，我们这里利用设未知数 x 的方式来假设，此时 x 是 1 和 6 其一），可以发现，所有 x 值可能的情况都不可能放在 $r7c128$ 上。除了同时在 $b8$ 的 $r7c45$ 外，只剩下 $r7c3$ 一个单元格。所以， $r7c3 = x$ ，故得到 $r7c3$ 里非 x 的删数；同理， $b8$ 里其余位置都不能是 x 。

这个思维视角使用了假设为未知数的方式来间接得到推论，这在数独技巧里称为**代数**，而这个技巧在后面会详细提到。

8-1-4 欠一数组的互补性

实际上，数组有互补，那么欠一数组就可以有互补，因为它们都借助了一点，即观察角度的切换，虽然说是切换，但实际上删数确实一致的，对于这一点，我们在数组和标准链列里已经深刻体会到了。那么欠一数组的互补是如何的呢？我们可以先来看看一些示例。

3	2	1	9	4	5	8	6	7
5	5 6	9	1	6	2	8	1	5
7	5	5 6	4	1	6	3	9	2
7 8	8	7	7	7	6	3	9	2
9	3	5	4	1	6	8	1	2
4	7	8	1	3	6	2	1	5
6	1	2	5	8	9	4	7	3
2	4		1	3	1	3	1	1
5	5	3	1	6	1	1	6	4
1	5	9	7	6	2	5	6	4

如左图所示，这是一个欠一三数组的拓展类型示例，我想，这一点我们将不用过多去介绍它的删数逻辑了。由于结构涉及 **r7** 和 **b9**，所以我们尝试把 **r7** 里的两处放 1、6、7 的单元格进行互补视角的转化，去掉它们，而取而代之的使用删数的单元格（而 **r7** 和 **b9** 的交集，即 **r7c789** 我们目前不用处理）。另外，**b9** 也进行同样的视角转换操作，于是我们就可以转化到右图的形式。

可以看到，右图就将欠一三数组转变为了一个欠一数对，而且删数完全没有发现变化。你可能会问我为什么如此神奇，实际上在视角转化的时候，利用的就是显隐性互补的模式，我们把没涉及到的单元格全部勾选了出来，而把涉及的全部无一例外地都去掉了。

8-2 融合待定数组 (Sue de Coq)

接下来我们来介绍一种跟欠一数组差不多的技巧，这种数组看起来就好像是真的数组。

8-2-1 基础融合

1	5 6	9	2	3 6	7	4 8	3 5	8 3
7	3	2	8	5	4	9	1	6
8	5 6	4	3 9	1	6 9	7 3	2 5	2 3
9	7 8	6	5	4	2	7 8	7 8	1
2	4 8	3	7	9	1	6	4 8	5
5	4 7	1	6	8	3	4 7	2 9	2 9
6	2	8	3 9	3 7	5	1	7 9	4
3	1	7	4	2	6 9	5	8 9	8 9
4	9	5	1	3 6	8	2	3 6	3 7

如图所示，我们观察到 **r1c79** 和 **r36c7** 四个单元格，特别特殊。为什么说它们特殊呢？**c7** 上，有一格是{47}，**b3** 内，有一格是{38}，而在 **c7b3** 内，恰好有 **r13c7** 两格既有{38}（橙色数字）又有{47}（绿色数字）。那么，我们就从这里比较奇怪的 **r13c7** 这两格来思考。

- 如果说 **r13c7** 全都只有橙色数字，那 **r13c7** 就会构成 3、8 数对，于是 **b3** 内，**r1c9** 就无法填数了，所以这样是错误的；
- 如果说 **r13c7** 全都只有绿色数字，那 **r13c7** 就会构成 4、7 数对，于是 **c7** 内，**r6c7** 就无法填数了，所以这样也是错误的。

所以，**r13c7** 有一格只有橙色数字，而另外一格则只有绿色数字。那这样恰好，在 **b3** 内，**r1c9** 和 **r13c7** 其中只有橙色数字的一格构成 38 数对，而在 **c7** 内，**r6c7** 和 **r13c7** 的其中另一格（只有绿色数字的一格）构成 4、7 数对。所以，**b3** 内其余单元格的 3 和 8 都可以删除；**c7** 内其余单元格的 4 和 7 也都可以删除。

是不是很有趣呢？这个结构由于融合了 3 和 8，以及 4 和 7，造成了两个区域的删数。

这个结构称为**融合待定数组**（**Sue de Coq**），简称 SDC。

为什么要叫“待定数组”呢？b3 内，3、8 数对的位置是待定的；c7 的 4、7 数对的位置也是待定的。而“融合”，就指的是既有{38}又有{47}的 r13c7 两格。

de 的 d 小写是为什么？de 是法语，大致相当于英文的单词 of。而这个技巧的名字，来自于一个网友发现了这个技巧，这个网友的网名就叫 Sue de Coq，他谦虚地为这个技巧取名 Two sectors Disjoint Subset，而为了纪念这个网友作出的杰出贡献，就将这个技巧命名为这个网名 Sue de Coq。

最后说明一个东西，我们称{r1c79, r3c7}(38)为一个**待定数组**（**Almost Subset**），因为这三格内，有四种不同的候选数 3、4、7、8，最终的结果就是，这四个不同的候选数的其中三个，将填入到这三个单元格之中，所以是“待定”的；同理，r13c67(47)也是一个待定数组。

那么接下来来看另外一种融合版本，数组规则不一致的。

8-2-2 数组规格不同的融合

1 2	2	1	3	4 5	2	2	6	9
	4	5		7 8	4	7 8		
7	8	5 6	1	4 5	2	2 3	3	2
		9		9	4	6	4	5
2 3	2 3	5 6	2	4 5	2			2
	4 6	7 9	4 6	7 8 9	4 6	1	5	5
			7	9	8	7	8	
4	7	3	8	6	9	5	2	1
9	5	2	4	1	1	6	8	3
		7	4	4	4			
6	1	8	5	2	3	9	4	7
1 2 3	2 3	4	2	1 3	1 2	2 3	1	3
	6		6	8 9	6	7	5	5 6
8			9		8	8	7	8
1 2 3	2 3	7	2	1 3	5	2 3	9	4
	6		6	8		8		
5	9	1	2	1 3	7	2 3	1	2
		6 4	6 4	8		8		6
			8			8		8

如图所示。这次融合的是 r9b9 内 r9c89 两格。根据刚才的分析，如果 r9c89 都只有橙色数字 2、3、8，则 b9 内的数字 2、3、8 不够填满 r8c7 和 r9c789 这四格，所以是错误的；

如果 r9c89 都只有绿色数字 1 和 6，则 r9 内的数字 1、6 不够填满 r9c389 三格，所以依然是错误的。所以 r9c89 一格只有绿色数字，另外一格只有橙色数字。这样一来，b9 内，r8c7 和 r9c789 内就一定会产生 2、3、8 显性三数组；而在 r9 内，r9c389 内就一定会产生 1、6 显性数对。于是 b9 内其余单元格都不能填入 2、3、8，而 r9 内其余单元格也都不能填入 1、6，所以 r7c789<>238、r9c45<>16。

这个版本稍显吃力一些，但分析方法依然是一样的。

8-2-3 “九数组” 示例

我们再来看一则示例，这个示例且先不说如何被找到，你可以试着观察和理解该示例，

这是一个涉及 1 到 9 所有数字的 SDC 结构。如图所示。

1	² _{4 8}	6	9	³ _{8 7}	³ _{2 3}	2	5
		3	¹ ₈	2	5	¹ ₆	4
⁵ ₉	² _{4 5}	² _{4 5}	¹ ₄	³ ₆	¹ ₇	² ₉	8
6	^{1 2 3} ₄	9	7	5	^{1 3} ₄	^{1 2 3} ₈	2
8	^{1 2 3} ₅	² ₅	^{1 2 3} ₄	^{1 3} ₄	9	^{1 2} ₆	7
^{2 3} ₇	^{1 2 3} _{4 7}	² ₇	^{1 2 3} ₄	6	8	5	^{1 2 3} ₉
4	9	¹ ₅	^{1 3} ₈	^{1 3} _{7 8}	³ ₆	^{2 3} ₅	^{2 3} ₆
^{2 3} ₅	6	^{1 2} ₈	5	9	^{1 3} ₈	7	4
⁵ ₇	³ ₅	⁵ _{7 8}	⁴ ₆	⁴ ₈	2	9	³ _{5 6}

如图所示，这是一个涉及 1 到 9 的全部数字的结构，这个结构已经是 SDC 里最大的结构了，无法超过它，因为超过它将一定会在结构里出现相同的数字，导致无法判定。

刚才的若干示例，融合的单元格数量都为两个，如果融合的单元格数量恰好变成三个呢？

8-2-4 带区块的融合

¹ ₄	2	3	7	5	9	¹ ₄	¹ ₄	6
8	7	¹ ₄		2	6	^{1 3} ₄	5	9
9	^{5 6} _{5 6}	^{5 6} _{5 6}	^{1 3} ₈	^{1 3} ₈	^{1 3} _{4 8}	7	2	^{1 3} ₄
² ₆	¹ _{5 6}	^{1 2} _{5 6}	^{1 3} ₈	4	^{1 3} ₈	9	7	¹ _{5 8}
3	¹ _{4 5}	7	¹ _{5 8}	9	6	¹ _{4 8}	¹ _{4 5 8}	2
¹ ₄	¹ _{4 5}	¹ _{4 5}	¹ _{5 8}	2	7	³ ₆	³ _{6 4 5 8}	¹ _{5 8}
5	¹ _{3 9}	³ _{6 9}	4	7	^{1 3} ₈	^{1 2 3} ₆	^{1 3} ₆	¹ _{8 9}
⁷ ₉	⁶ ₄	⁶ ₄	⁶ ₄	⁶ ₄	⁶ ₄	⁶ ₄	⁶ ₄	⁶ ₄
⁷ ₉	² ₆	8	¹ ₃	¹ ₃	5	^{1 2 3} ₄	^{1 3} ₄	¹ _{7 9}

如图所示。这个结构按照原定的逻辑来思考和分析的话，过于复杂，因为一共要讨论七种不同的情况。显然这是麻烦的。那么我们换一个思路来思考这一点呢？

我们把 r89c1 和 r2789c3 这六个单元格内所有出现的候选数进行分类和涂色：我们把 1 和 4 涂成绿色，把 2、6、7 涂成橙色，而 9 单独涂成紫色。

观察 r89c1 两格，这两格都只有橙色数字（2、6、7），它还需要一个单元格只有橙色

数字 2、6、7，填数才能趋于稳定（说白了就是构成 2、6、7 三数组嘛），多了一格都不行。那结构内如果只有 r89c1 有 2、6、7 的话，这样看似可以，但是这同时也意味着 r789c3 这三个单元格内，没有橙色数字。没有橙色数字的话，r2789c3 就只剩下 1、4、9 三种不同的候选数了，而这三个不同的数字却要填入到 r2789c3 四个单元格之中去，这样显然是不够的，所以这样是不可以的。所以，少了一格只有橙色数字，也是不可以的。

同理，看绿色数字（1、4），这样分析也是同样的道理：由于 r2c3 只有绿色数字，那么为了保证结构填数不会出现矛盾，要么不需要其余只有绿色候选数的单元格（0 个），要么刚好只需要 1 个，要么需要 2 个或者更多。

如果是 0 个（也就是说不需要其它单元格只有绿色数字）的话，意味着 r789c3 这三个单元格只有 2、6、7、9 四种不同的数字。转去看 b7，对于 r89c1 和 r789c3 这五格而言，显然不可能只填入四种不同的数字就够的。五格恰好同一个宫，至少要填入五种不同的数字，所以四种显然不够，也就矛盾了；

如果是更多个其它单元格只有绿色数字的话，那么肯定不可以。因为超过一个只有绿色数字的话，算上 r2c3(14)，就有至少三格只有绿色候选数了。绿色候选数只有两种，而要填入到至少三格里面去，这样显然是矛盾的。所以说，也只好恰好需要一个单元格只有绿色候选数，来恰好构成 1、4 数对。

这样一来，r789c3 里有一格被 r89c1 拿去构成 2、6、7 三数组了，又有一格被 r2c3 拿去构成 1、4 数对了，那还剩下一个单元格呢？紫色数字还没上场呢！所以 r789c3 最后剩下的那一个单元格就是 9 了！

所以这样一分析起来，b7 内会产生 2、6、7 三数组，c3 内会产生 1、4 数对，而 r789c3 内一定有一格是 9。所以 b7 内其余单元格不应该再含有候选数{267}、c3 内其余单元格不应该再含有候选数{14}，而 c3 和 b7 内，其余单元格也都不应该含有候选数 9。所以图上的结论就应该为：r46c3 <> 149、r78c2 <> 2679。

8-2-5 显性转隐性的视角

1 5 7 8 9	4	1 5 6 7 9	3	1 6 7 8	5 7 8	2	1 5 6 7 8	3	1 6 7 8	5 7 8	2
3	1 5 7	1 5 6 7 9	4 5 6 4 7	1 6 7	2	8	1 5 6 7	1 5 6 7 9	6	7 9	2
1 2 5 7 8	1 2 5 7 8	1 5 6 7	1 5 6 8	9	5 7 8	3 4	1 2 5 7 8	1 5 6 7	9	5 7 8	3 4
1 4 7 9	1 3 4 7 9	1 3 4 7 9	2	1 4 8	3 4 8 9	5	1 4 7 9	1 3 4 7 9	2 4 7 9	3 6 7 8	5
1 2 5 7 9	1 2 3 5 7	8	1 9	1 9	3 6	4	1 2 5 7	1 2 3 5 7 9	8	1 9	1
1 2 4	6	1 3 4	7	5	4 8	9	1 2 4	6	1 3 4	7	5
6	5 7 8	3 4 5 7	4 5 8 9	4 7 8	3 7 8	1	2 5 7 9	2 4 5 7 8	7 8 9	7 8 9	
4 5 7 8	5 7 8	2	4 5 6 8 9	4 7 8	6 7 8 9	1	4 5 6 7 8	6 7 8 9	6	7 8 9	
1 4 5 7 8	9	1 4 5 7	4 5 6 8	2	4 7 8	3	5 7 8	4 5 6 7 8	3	4 5 6 7 8	3

如左图所示，这个结构涉及了 1、3、4、5、7、8、9 七种数字，按照 SDC 来看，确实属于 SDC 的规范版本：1、3、9 同宫，4、5、7、8 同列，所以七个单元格内的七种数字完全不存在重复的情况，意味着七种数字必然填入到这七个单元格之中，且填数种类不多也不

少。于是可以得到，c6 其余位置的 4、5、7、8 以及 b5 其余位置的 1、3、9 都可以删除，如图红色的删数。不过，这个结构稍大了一些，我们可以尝试“翻转”逻辑，用数组的隐性思维来思考这一点：

如右图所示，这样就把原本的七个单元格变为了只涉及四个单元格了。观察 c6，3 和 9 的位置只存在于 r468c6 三格之中；而观察 b5，4 和 8 的位置只存在于 r4c56 和 r6c6 三格之中，另外，b5 和 c6 的交集上，结构涉及的两格里，候选数 3、9 和 4、8 都有。

如果 r46c6 都是紫色候选数，则 r46c6 可以直接看作 3、9 隐性数对，b5 内 4 和 8 的位置不够填；如果 r46c6 都是蓝绿色候选数，则 r46c6 可以看作 4、8 隐性数对，c5 内 3 和 9 的位置不够填。所以 r46c6 有一格只有蓝绿色候选数，而另外一格则只有紫色候选数。这样一来，只有蓝绿色的单元格就会和 b5 之中的 r4c5 构成 4、8 隐性数对；而另外一格则会和 c6 之中的 r8c6 构成 3、9 隐性数对。所以这样一样可以产生两个独立的数对。我们能确定的是，隐性数对的删数只能有涉及的四格之中，而因为 r46c6 最终哪个只有蓝绿色候选数，哪个只有紫色候选数是不确定的，所以删数只能在构成 4、8 数对和 3、9 数对的另外一格。于是 r4c5 <> 13、r8c6 <> 4578。这样就解决了 SDC 结构过大导致不好观察的问题。这样的观察角度也称为全隐性的 SDC。

另外，因为 SDC 涉及两个区域，所以最终可能产生四种情况：

- 显性、显性
- 显性、隐性
- 隐性、显性
- 隐性、隐性

因为 SDC 是涉及两个区域下的结构，所以完全可以存在显性和隐性同时观察的视角。这种视角就比较别扭了，我们就不展开给大家展示例子了。

8-2-6 自噬融合

4 5 8	1	9	2 5 8	4 5 8	6	3	2 7 8	4 7
4 5 8	7	2	5 6 9 2	4 5 8 9	4 5 8 9	1	6 8 9	4 6 9
4 8	3 4 8	6	8 9	1	7	2 8	2 8 9	5
1	2 4 5 7	4 5 7	3	5 6 7 8	2 5 8	9	2 4 6 7	2 4 6
2 3 4 7	2 3 4 5 9 7	4 5 7	1 5 6 7 9	5 6 7 9	1 2 5 9	2 5 6 7	2 3 4 6 7	8
6	2 3 5 9	8	4	7 5 9	2 5 9	2 5 7	2 3 7	1
2 4 7 8	6	4 5 7	5 7 8 9 7 8 9	4 5 7 8 9	4 5 8 9	1 2 7 8	1 2 7 8 9	2 3 7 9
4 7 8	4 5 8	1	5 6 7 8 9 1	2	4 5 8 9	6 7 8	6 7 8 9	6 7 9
9	2 8	3	1 6 7 8	6 7 8	1 6 8	4	5 7	2 6

如图所示，图上所有红色的数字均可以被删除。首先，我们可以看到，结构一共涉及 8 个单元格，并且在结构里，数字 2、5、9 只有 c4 上出现过；而且数字 1、6、7 也都只在

b8 里出现过。维度这个结构里有一个“不友好的”数字 8：它在 c4 和 b8 这两个区域下都出现了，也就是说，它跨区域出现。跨区域出现也就意味着这个数字很有可能在结构里出现两次，而不像其他的数，因为只有一个区域有这个数，那么这个数只可能出现一次。

既然数字 8 可以出现两次，那么我们就来想想这个 8 最终的使命究竟是如何的。细数一下，刚才我们得到了一些完全不会影响到结构的只出现在一个区域的数字，一共是 6 个这样的数：2、5、9 和 1、6、7。如果一格填一个数，那也才 6 格。结构一共涉及 8 格，这说明还有两个单元格要填入不是这些数字的数字。根据刚才的讨论，我们发现，抛开这些数字不看之后，就只剩下数字 8 了。而且，一共要填满 8 格，空出两格，那两格都得填 8 才行。这也就说明了一点，结构里必须有两个 8。

不过，如何安放这两个 8 呢？很简单，只要这两个 8 它们之间不受影响（比如填到同一行列来违背规则，这就算是受影响了），怎么填都可以。所以，这两个 8，一个必须产生在 r123c4 里，而另外一个 8 则必须产生于 r9c56 里。

那结构里不是还有 r789c4(8)呢？通过刚才的逻辑，我们就明白了，这个 8 是显然不可出现到这里的，所以 r789c4 <> 8；另外，因为 2、5、9 出现在 c4 里，所以必然 2、5、9 都会有，于是这一列其余的单元格的 2、5、9 都可以被去掉；类似地，b8 里的其余单元格的 1、6、7 也都能去掉。不过这个例子告诉了我们，数字 8 显然是在两端都会成为区块结构的（即指的是 r123c4 和 r9c56 里都必须有数字 8），所以对应的 b28 两个宫里，其他单元格的数字 8 也都可以被删除掉（别忘了 r9c56 除了能影响 b8，还能影响 r9 哈，所以 r9 里其他位置的 8 也还是都能被去掉，比如 r9c2(8)）。至此，这个示例的所有删数才全部被找到。

这种结构叫做**自噬 SDC (Cannibalistic SDC)**，**自噬 (Cannibalism)** 实际上是一个指的是结构涉及的候选数因为逻辑也能被去掉的一种特殊情况的术语词汇。

自噬这个词在以后还会用得更多，而这个词本身其实是属于之前提到的鱼这个技巧的范畴（虽然之前的内容没有任何提及到自噬的情况，但千万别否认，因为在后面的较难的板块会提到**自噬鱼 (Cannibalistic Fish)** 结构，因为它比较难，我就把它丢到后面才去讲了），所以这个术语我们就不在这篇文里展开讲解了，也不会将此术语添加到术语表里。到后面的内容的时候，我们再详细讨论这个术语指示的情况。

我们再来看一则示例。

1			1 2				1 2	
7	4	7 8 9	4	3	4	5	4	9
5	6		2	1	4	9	3	2
5	3		8	2	7	8	4	5
5	4	9	4	6	8	1	5	9
4	1	5	6	3	7	2	9	8
2	2	7	4	9	1	3	5 6	5 6
9	3	6	5	8	2	4	1	7
1	5	4	7	2	3	1	4 6	1 6
8		8				9		9
6		1	8	1	5	4	2	3
1 2	2		9	7		5		
7	4	3	9	1	5	6	4 5	4 5

如图所示，这个示例也是一个很好的说明自噬 SDC 的模范示例。

我们首先可以发现，如果抛开 4 和 8 这两个跨区域出现的数字的话，我们可以看到结构里，所有的 1、2、9 只在 b1 里出现，而 7 只在 r1 上出现。我们数一下，结构一共包含 8 个单元格，但现在只有四个单元格填入 1、2、9、7，这使得还有多余的四个单元格我们还得继续确定填数。

幸运的是，4 和 8 是跨区域出现的。如果我们保证 r1 和 c3 这两个区域里分别都有一个 4 和一个 8，就恰好占据 8 个单元格，不多不少。但凡在 4 和 8 里有一个数字少出现一次，那么都会使得结构填不满，或者出现矛盾。举个例子，我们此时假设 r1 只有一个 4，而没有 8 的话，那么为了满足 8 个单元格要填满的要求，1、2、7、9 占了四个位置，它们是不跨区的，就使得它们最多在结构里只可能出现最多一个，我们按最多的情况来算，现在最多填入了四个位置了，但 r1 只有一个 4，也才占据了五个单元格，还有三个单元格，哪怕放入了 4 和 8，也会多余一个单元格，无法填数：1、2、9 已经不行了，7 也是不可以的，而 4 和 8 已经达到了我们假设情况的饱和了，所以此时必然会有一个单元格出现填不了数字的矛盾。

所以，我们为了保证结构必须放满 8 个数字，4 和 8 必须得在 r1 和 c3 两个区域里分别出现各自一次，因此 r1 和 c3 的交汇处 r1c3 是不允许放入 4 和 8 的，否则 4 和 8 在 r1 和 c3 上就会有一个数字少出现一次，使得结构放不满，导致出错。也因此，除了 SDC 的基本删数外，r1c3 也不能填入 4 和 8，把它们删除掉；当然，这个 SDC 肯定是成立的，为了保证结构能充分成立，我们只有唯一的一种填法，即 r1 上是 1、2、7、9，且含有一个 4 和一个 8，c3 上则分配了一个 4 和一个 8。这种唯一的填数模式，恰好保证了 r1 和 c3 上必然会有 4 和 8 的出现，删掉其余位置的 4 和 8，以及 b1 里必然有 1、2、9，c3 上必然有 2 的结果。所以整个 SDC 的删数包含了 r1 其余位置的 4 和 8、b1 其余位置的 1、2、9，c3 上其余位置的 2、4、8。

不过这个例子比较难理解的地方是，它和之前的 SDC 不太一样，这个 SDC 涉及的是 r1 和 c3，即一行一列，而不是标准情况下的一行一宫，或是一列一宫。

8-2-7 从欠一数组到 SDC 的转化

欠一数组和 SDC 是具有非常相似的性质的，你完全可以发现，欠一数组和 SDC 具有相当多的相似的地方。其实我们确实也能找到一种方式，将欠一数组完全转化为 SDC 结构；当然，也可以从 SDC 转回欠一数组。

3	1	5 6	5 6	1	6	9	7	4	2	3	1	5 6	5 6	1	6	9	7	4	2
2	1 2	2		1	3	1 2	6	8	3	4	2	1 2	2	1	3	1 2	6	8	3
4	7 9	7	4	7	4	7	9		9	7	9	7	4	7	4	7	9		9
2	2	2	2	2 3	3	2	2 3	1	5	4	6	4	6	4	6	4	6	4	6
4	6	7 8 9	7 8	7	8	9	7	8	9	7	8	9	7	8	9	7	8	9	7
2	2 3	2 3	2	2	2 3	2	2 3	6	1	4	7	4	5	4	7	4	5	4	7
4	7 8	7 8	4	5	4	7	8 9			7	5	4	7	8 9			7	5	4
1	2 3	9	8	6	4	2	2 3		3	7		5	4	3		7		5	4
2	2	2	2	1 2	1	3	2			4	6	5	6	1 2	1	3	2		
4	6	7 8	7 8	7 9	7 9	9	8 9	7	9	4	6	7 8	7 8	7 9	7 9	9	8 9	7	9
7 8 9		1	4	2	4	7	5	3	6	7 8 9		1	4	2	4	7	5	3	6
2	6	4	2 3	6	3	3	8	1	9	7	8	6	4	2 3	6	3	8	1	9
5		3	3	1	3	1	3	4	2	8		3	3	1	3	1	3	4	2
7	9	7	6	7	9	7	6	7	9	7	6	7	9	7	6	7	9	7	6

如图所示，我们可以发现这是一个欠一三数组结构，而我们直接将 c6 的没涉及的单元格全部都勾出来（此时也算上 r23c6），把它们上色，就可以发现，我们已经成功地转换了。

8-3 跨区数组基本原理 (Extended Subset Principle)

8-3-1 假数组/伪数组 (Subset Counting)

6	1	9	2	5	3	7	4	5
4		5	2	1	6	1	3	9
8	4		3	4	9	1	2	6
5		3	6	1	2	5	4	5
2	4	5	1	3	5	2	6	1
7	9	7	8	7	8	9	8	9
2	5		1	5	6	4	3	5
1		6	4	4	3	5	6	2
3		6	5	5	2	4	6	1
4	5	2	7	6	1	9	8	3

如图所示。我们观察 r5c2359 和 r6c2 这五个单元格。它们有一个神奇的地方是，恰好五格里面刚好只有五种不同的候选数 1、4、5、7、8。这恰好满足数组的定义啊，但是这一定就是一种数组吗？当然不能直接确定了。数组的定义是必须规定于同一区域下的，而这里的五格包含跨区域的单元格，比如这里 r6c2 和 r5c9 两格，就根本没啥联系了。

那我们这么去想这个问题：既然长得很像数组，那就分“内部填数不重复”和“内部填数有重复”来讨论情况吧。

我们发现，数字 1、4、5、7、8 这五种数字里面，只可能有数字 7 可能会重复。因为 7 的位置可以同时填入到 r6c2 和 r5c9 之中，这样它们两格没有直接关系，就可能产生重复。但是，其余的数字都不可能重复，比如 1 的位置就只出现在 r5c39 中，恰好 r5c39 同行（r5），所以这两格一定只可能有其中一格是 1 的；同理，4 的位置也只出现在 r5c23 之中，恰好 r5c23 同行（r5）和同宫（b4），所以 4 在这五格里面就更不可能重复了；那么 5 和 8 的分析方式也是一样的（5 的所有填数可能均同行，8 的填数可能均同宫）。所以唯独数字 7 可能有重复。

那么，如果要把 1、4、5、7、8 这五种数字填入到这五格之中去的话，当内部有重复（就假设此时 r5c9 和 r6c2 同时是 7）时，剩余的 r5c235 三格，就有四种填入的数字（1、4、5、8）可选。这样一来，究竟是其中的哪三种数字被选进去填入了，我们无从得知。所以这方面的情况我们确定不了。但是我们能确定的是，不管重不重复，数字 7 在结构之中都是不可缺少的。换句话说，这四格的数字 7，是不可能全部从盘面之中消失的。同时全被去掉时，这五格里面就只剩下四种数字（1、4、5、8）了，又因为它们一定是不可填数重复的，所以这显然是不够的，所以便产生了矛盾。

于是乎，我们得到了结论：不管内部怎么填数，数字 7 都是不可缺少的，所以删除掉的是四格候选数 7 共同对应到的位置。所以，r5c1 <> 7。

这个结构称为**伪数组**（或**假数组**，Extended Subset Principle/SubSet Counting）。伪数组分析起来相当痛苦，也很费劲，而结论却如此“草率”。那么有没有什么办法更为简单呢？或者说，怎么样观察和使用更快一些呢？

8-3-2 跨区数组删数理论和原理

伪数组的分析方向大体是按照是否是一个“跨区域的数组”来看的。如果是，则内部不重复；而如果不是，则内部一定有重复的数字。然后就去找重复的数字到底是多少。而伪数组有一个特征，这在刚才我们推导原理时，就得到了一点：伪数组假设内部有重复的话，唯一有重复可能的数字一定不可全部从盘面之中消失。所以，在观察和找结论的时候，只要判断出唯一可重复的数字是多少，找到交集，就可以直接删数了。

那么，有没有一眼看出来，是一个跨区域的结构，但是内部一定不重复的呢？有的，比如这个例子。

1 2 3 7 6	2 3 7 8 6	2 3 8	1 2 3 4	1 2 4 6	1 2 4 6	5	1 2 3 9	1 6 9
4	2 3 5 6	2 3 5	9	1 2 5	1 2 5 6	8	1 2 3 4	7
1 2 3 6	9	2 3 5	1 2 3 5	7	8	1 2 3 4	1 2 3 4	1 4 6
5	3 4 7 8	3 4 8 9	1 2 7	1 2 8 9 7	1 2 7	6	1 3 4 7 8 9	1 4 8 9
2 7 9	1	2 8 9	4	6	3	7 9	5	8 9
3 7 9	4 3 7 8	6	1 5 7	1 5 8 9 7	1 5 7	1 3 4 7 8 9	2	2
2 9	2 4 5	2 4 5 9	8	3	1 2 4 5 7	1 2 4 5 7 9	6	1 4 5 9
8	2 4 5 6	7	1 2 5 6	1 2 4 5	9	1 2 4	1 2 4	3
2 3 6 9	2 3 4 5 6	1	2 5 6 7	2 4 5	2 4 5 7	2 4 7 9	2 4 7 8 9	4 5 8 9

如图所示，这是一个 SDC，不过我没有分成橙色和绿色来涂色。因为这样更接近于伪数组的分析方式。

我们使用伪数组的分析方式来看，它是一个跨区七数组，但是巧妙的是，内部并不可能有可以重复的数字，因为 1、2、3、4 同列（c8），而 7、8、9 同宫（b6），所以它是 DDS。按照 DDS 的删数原则，c8 内其余单元格都不可以是 1、2、3、4；而 b6 内的其余单元格都不可以是 7、8、9 了。于是删掉它们。

SDC 的观察或许比较麻烦，因为需要分两个区域来假设其待定数组的情况。但是使用伪数组角度，是不是会稍微轻松一些呢？

融合待定数组（SDC）还有一个名字叫**双区域分布式跨区数组（Two-sector Disjoint Subset）**，就是因为这个原因。

8-4 均衡数组（Aligned Subset Exclusion）

接下来我们来看一种稍微暴力一些的技巧，叫**均衡数组（Aligned Subset Exclusion）**。

8-4-1 均衡数对（Aligned Pair Exclusion）

7	8	3	1 2	6	1 2	1	5	4
5	6	9	7	1 4	1 4	1 3	2	1 3
1	2	4	5	8 9	3	7	6 8 9	6 8 9
2	7	1 2	1 2	1 2 3	5	4	3 1	3 6
4	1	5	1	1 3	7	1 3	6 8 9	3 6 8 9
2	3	1 2	8	1 2	1 2	1	6 7	5
2	4	7	4	5	2	3	1	3 6
3	1	1 2	9	1 2	6	5	4 8	7
6	5	1	4	7	1	2	4 3	3 9

如图所示，观察 **r12c6**。在无计可施的情况下，我们来枚举这两格的所有填数情况。那么，**r12c6** 都是三值格，所以就应该有九种不同的填法。我们一一罗列出来：

r1c6	r2c6	正确情况	错误原因
1	1	×	b2 出现相同数字
	4		
	8	×	r9c6 没数字可填
2	1	×	r1c4 没数字可填
	4		
	8	×	r7c6 没数字可填
9	1		
	4		
	8	×	r3c5 没数字可填

第一列表示 **r1c6** 填的所有可能，第二列表示 **r2c6** 填的所有可能。随后我们发现，有一种特别的情况，使得它所有对应的情况，全部都是错误的。就是那个 **r2c6 = 8**。当 **r2c6 = 8** 的时候，**r1c6** 的所有填数可能与之的组合都不对。正是因为如此，所以 **r2c6 <> 8**。

这个结构称为**均衡数对**（**Aligned Pair Exclusion**），它的逻辑很暴力——枚举。说白了就是一个一个去找。更大的结构就是下面的这个例子了。

8-4-2 均衡三数组 (Aligned Triple Exclusion)

<div>3</div> <div>5 7</div> <div>4</div> <div>6</div> <div>4</div> <div>3</div> <div>4 5</div> <div>1</div> <div>5 8</div> <div>9</div> <td><div>2 3</div><div>2 3</div><td><div>3</div><div>6</div></td></td>	<div>2 3</div> <div>2 3</div> <td><div>3</div><div>6</div></td>	<div>3</div> <div>6</div>
<div>3</div> <div>5</div> <div>4</div> <div>6</div> <div>1</div> <div>2</div> <div>4 5</div> <div>9</div> <div>8</div> <div>7</div> <td><div>3</div><div>6</div></td>	<div>3</div> <div>6</div>	
<div>7 8</div> <div>2</div> <div>9</div> <div>3</div> <td><div>7 8</div><div>6</div><div>4</div><div>5</div><div>1</div></td>	<div>7 8</div> <div>6</div> <div>4</div> <div>5</div> <div>1</div>	
<div>6</div> <div>9</div> <div>5</div> <div>4 5</div> <div>2</div> <div>3</div> <div>1</div> <div>4</div> <div>4 5</div> <td><div>4 5</div><div>7 8</div></td>	<div>4 5</div> <div>7 8</div>	
<div>4</div> <div>8</div> <div>2</div> <div>5</div> <div>6</div> <div>1</div> <td><div>5</div><div>3</div><div>3</div><div>5</div><div>3</div></td>	<div>5</div> <div>3</div> <div>3</div> <div>5</div> <div>3</div>	
<div>1</div> <div>3</div> <div>5</div> <div>4 5</div> <div>4 5</div> <td><div>5</div><div>2</div><div>6</div><div>4 5</div><div>7 8 9</div></td>	<div>5</div> <div>2</div> <div>6</div> <div>4 5</div> <div>7 8 9</div>	
<div>3</div> <div>4</div> <div>4</div> <div>5</div> <div>5</div> <div>2</div> <div>6</div> <div>1</div> <div>4</div> <td><div>3</div><div>9</div></td>	<div>3</div> <div>9</div>	
<div>2</div> <div>5</div> <div>6</div> <div>1</div> <div>3</div> <div>4</div> <div>7</div> <td><div>2</div><div>2</div><div>8 9</div><div>8 9</div></td>	<div>2</div> <div>2</div> <div>8 9</div> <div>8 9</div>	
<div>2 3</div> <div>1</div> <div>4</div> <div>3</div> <div>6</div> <div>9</div> <div>7</div> <td><div>3</div><div>4</div><div>5</div><div>8</div><div>2 3</div><div>4 5</div><div>8</div></td>	<div>3</div> <div>4</div> <div>5</div> <div>8</div> <div>2 3</div> <div>4 5</div> <div>8</div>	

按照刚才的逻辑，我们把 r1c46 和 r2c1 的所有填数情况都枚举一遍。

r1c4	r2c1	r1c6	正确情况	错误原因
4	3	5	×	r2c5 没数字可填
		8	×	r1c3 没数字可填
	5	5	×	r2c5 没数字可填
		8	×	r1c3 没数字可填
5	3	5	×	r1 将出现相同数字
		8		
	5	5	×	r1 将出现相同数字
		8		
7	3	5		
		8	×	r3c5 没数字可填
	5	5		
		8	×	r3c5 没数字可填
8	3	5		
		8	×	r1 将出现相同数字
	5	5		
		8	×	r1 将出现相同数字

随后发现，所有 r1c4 = 4 的情况均是错误的。所以，r1c4 <> 4。

这个就是均衡三数组 (Aligned Triple Exclusion) 了。

8-4-3 均衡四数组 (Aligned Quadruple Exclusion)

1	²	9	3	4	²	²	5	²
	₇				_{7 8}	_{7 8}		_{7 6}
8	5	²	²		⁶	1	4	3
		₇	_{7 9}	₇	_{6 9}			
3	6	4	5	1	²	²		²
				_{7 8}	_{7 8}	_{7 9}	_{7 9}	
⁴	⁴	3	8	5	²	⁴	⁶	1
_{7 9}	_{7 9}			₇	₇	_{7 9}	_{7 9}	
^{4 5}	8	²	6	9	1	²	3	²
₇		₇			_{4 7}		_{4 7}	
6	^{1 2}	^{1 2}	²	3	4	5	8	²
	_{7 9}	₇	₇				_{7 9}	
^{4 5}	¹	¹	¹	8	⁶	3	2	^{4 6}
_{7 9}	_{7 9}	_{7 9}	_{7 9}		₉		₇	₇
2	⁴	8	⁴		3	⁴	⁶	1
	₇		₇	_{9 7}		₇		5
⁴	3	¹	¹	2	5	9	⁶	8
₇		₇	_{6 7}					

可怕的枚举情况。一共我们需要枚举 r7c6、r8c27 和 r9c4 四个的所有填数组合。候选数一共是 $2 * 3 * 3 * 3 = 54$ 种情况！

不过，r8c2 = 7 时，其它三个单元格的所有填数情况全部都会导致矛盾，所以 r8c2 <> 7。

8-4-4 为什么如此暴力的技巧，也会放在这里介绍？

均衡数组一直饱受争议，因为它的推理方式过于暴力，并且不容易观察到。在此为大家介绍一种观察方式。

首先，由于均衡数组的枚举单元格是不定的，所以我们此时按照每一大行和大列的方式搜寻。如果找的是均衡数对，则去找大行和大列存在的双值格（含有两个候选数的单元格），如果这样的双值格较多，就有一定可能产生均衡数对（但如果这样的单元格较少，就不容易得到类似于均衡数对枚举而产生的矛盾现象）；那如果是均衡三数组，则去寻找大量的三值格（含有三个候选数的单元格）和双值格，看是否这样的单元格很多。如果这样的单元格很多，则我们可以知道的是，这样的均衡三数组很有可能出现（当然，也有可能不存在，只是很大程度满足了结构需求了）。

当结构满足要求的时候，这个时候就去看双值格和三值格涉及的所有数字，然后去寻找只有这些候选数的单元格，最后才会针对这样的结构进行枚举推导。而均衡数组经常被改写为待定数组的某些形式，但有时也不一定比均衡数组观察起来方便。

那么，整个待定数组的基本使用技巧就讲到这里，这也象征着进阶技巧已经全部讲完了，接下来我们将进入到链的学习之中。

第三篇章 链

何为链，链是什么？链是一种动态、灵活的证明思路链条。

Part 1 双强链 (Turbot Fish)

1-1 摩天楼 (Skyscraper)

8	¹ ₄	5	6	3	² ₉	⁴ ₉	7	^{1 2} ₉
¹ _{6 9}	⁴ ₆	3	7	² ₄	5	8	¹ _{4 6 9}	^{1 2} _{6 9}
⁶ ₉	2	7	⁴ ₉	8	1	3	⁴ _{6 9}	5
2	3	6	8	9	4	1	5	7
4	8	1	2	5	7	⁶ ₉	3	⁶ ₉
5	7	9	1	6	3	2	8	4
7	^{5 6}	2	3	¹ ₄	^{8 9}	^{4 5 6} ₉	¹ _{4 6 9}	¹ _{6 8 9}
3	^{5 6}	4	⁵ ₉	^{1 2} _{8 9}	² _{8 9}	7	¹ _{6 9}	¹ _{6 8 9}
1	9	8	⁴ ₅	7	6	⁴ ₅	2	3

如图所示，我们观察到一个特别的结构。首先是 r39，r39 上都有关于 4 的共轭对，分别是 r3c48(4)和 r9c47(4)。而奇怪的是，它长得很像之前学习到的一种数独技巧：孪生二链列，而且用这个技巧完全可以得到应有的所有删数。而现在我们用另外一个视角来看这个结构。r3c8(4)一共有两种可能：要么 r3c8 = 4，要么 r3c8 <> 4。那么我们来看一下，当 r3c8 <> 4 时，会怎么样呢？

当 r3c8 <> 4 时，由于 r3(4)共轭对，所以 r3c4 = 4，所以 r9c4 <> 4 了。于是，r9(4)共轭对就用起来了：因为 r9c4 <> 4，所以 r9c7 = 4。那么缩略一下说法，就是当 r3c8 <> 4 时，r9c7 = 4。

而 r3c8 = 4 是另外一种情况，这意味着 r3c8 自己可以直接填 4。

所以综上，一共只有两种填数情况，却导致了 r9c7 和 r3c8 里面至少一个单元格是填 4 的，所以 r9c7(4)和 r3c8(4)共同对应的地方，就不应该填入 4；否则它将使得这两个单元格都不能放 4，出现违背结论的矛盾。

那么，为什么说“至少有一个单元格”，而不是“有且仅有一个单元格”呢？这一点，我们在 Wing 结构一节里面已经提到过一部分原因，这里针对这个题目再阐述一下：虽然看似 r3c8 拥有填 4 和不填 4 的两种填法是“互斥”的两种情况，但是“当 r3c8 <> 4 时，r9c7 = 4”成立，也并不意味着“当 r3c8 = 4 时，r9c7 <> 4”就一定成立。所以 r3c8

= 4 时，也是有可能使得 $r9c7 = 4$ 的。所以我们说 $r9c7$ 和 $r3c8$ 至少有一格是填 4 的。

由于 $r3c8(4)$ 和 $r9c7(4)$ 至少一个单元格填 4，所以不管怎么个情况，它们共同对应的地方，就不应该再含有候选数 4，所以删除掉它们。对应图中的话，就应该为 $r1c7, r7c8 <> 4$ 。

这个技巧称为**摩天楼**（Skyscraper），而摩天楼，则属于**双强链**（Turbot Fish）里面的一个特殊的结构。那么，什么是双强链呢？双强链，顾名思义，也就是有两个强关系的链。那，什么是强关系呢？接下来我们来介绍一下，关于标准链的基本术语及内容。

1-2 基本链理论和术语

1-2-1 一些术语

我们从刚才的示例之中，学习到了双强链的其中一个特殊结构：摩天楼。我们定义出几种说法，为了方便观察和理解：

- **真**（True），指的是某一个候选数直接填入到盘面之中的情况，即用等号=表示的情况；
- **假**（False），指的是某一个候选数从盘面之中消失的情况，即用不等号!=或<>表示的情况；
- **强关系**（也称**强链关系**，Strong Inference）：如果某一候选数为假，则得到另外一个候选数为真的，就称这两个候选数有强关系；
- **弱关系**（也称**弱链关系**，Weak Inference）：如果某一候选数为真，则得到另外一个候选数为假的，就称为两个候选数有弱关系；
- **节点**（Node）：结构的每一个真假讨论的点都是一个节点；
- **链头或链首**（Head），指的是链结构在推导过程之中的第一个节点；
- **链末或链尾**（Bottom），指的是链结构在推导过程之中的最后一个节点；
- **交集或共同作用域**（Intersection），指的是链头和链尾共同对应的地方（或者说，都可以看得到的地方）。

那么来总结一下刚才的推导方式：

分某一个候选数的真假两种填数情况进行讨论，当其为假时，引出一个真和假的交替推导序列，然后得到尾部为真的情况。而因为头部还有可能为真，所以头尾的两个候选数至少有一个为真，因此删除头尾两候选数的交集。

那么，这个“真和假的交替推导序列”就被称为**标准链**（简称**链**，Alternating Inference Chain，简称 AIC）。一般而言，只要我们画出标准链结构，就可以直接在图上体现逻辑，也就可以不需要给不知道逻辑的小伙伴讲解了。

1-2-2 链的呈现方式

链可以用文本形式或图像形式呈现，下面来说明一下，如何使用文本形式表达。

我们使用候选数 1==候选数 2 或候选数 1=候选数 2 的方式，表示两个候选数形成强关系，例如 $r3c4(4)==r3c8(4)$ 或 $r3c4(4)=r3c8(4)$ 都表示这两个数形成了强关系；而使用候选数 1--候选数 2 或候选数 1-候选数 2 的方式表示两个候选数形成弱关系，例如 $r3c4(4)--r9c4(4)$ 或 $r3c4(4)-r9c4(4)$ 都表示这两个数形成了弱关系；而图上的链写作

$$r3c8(4)=r3c4(4)-r9c4(4)=r9c7(4)$$

另外，如果候选数涉及相同数值，那么可以直接写单元格，而省去候选数数值，将它写在最后，比如前面的示例里的文本书写格式可以简化为

$$r3c8=r3c4-r9c4=r9c7(4)$$

下面来说一下如何使用图像呈现链结构。一般来说，我们呈现链结构的时候，将使用箭头画法，来表示推导的顺序和方向。并用实线箭头表示强关系，虚线箭头表示弱关系。如图所示，这是前面示例的链画法。

8	¹ ₄	5	6	3	² ₉	⁴ ₉	7	^{1 2} ₉
¹ _{6 9}	¹ _{4 6}	3	7	² ₄	5	8	¹ _{4 6 9}	^{1 2} _{6 9}
⁶ ₉	2	7	⁴ ₉	8	1	3	⁴ _{6 9}	5
2	3	6	8	9	4	1	5	7
4	8	1	2	5	7	⁶ ₉	3	⁶ ₉
5	7	9	1	6	3	2	8	4
7	^{5 6}	2	3	¹ ₄	^{4 5 6} _{8 9}	¹ _{4 5 6 9}	¹ _{6 8 9}	¹ _{6 8 9}
3	^{5 6}	4	⁵ ₉	^{1 2} _{8 9}	7	¹ _{6 9}	¹ _{6 8 9}	¹ _{6 8 9}
1	9	8	⁴ ₅	7	6	⁴ ₅	2	3

1-3 原理进一步剖析

1-3-1 链的头尾可以同真吗？

显然，这一点在刚才的题目文字里已经说明了。链头假设为假的时候，可以得到链尾为真；而链头还可能为真，所以删去交集。虽说两种假设是互斥的，但不代表结论就必须和假设一样互斥。结论的内容是通过假设得到的；但结论可能通过其它渠道得到，并不是仅通过唯一的方式来得到，而且我们在这个题目的解尚未得到之前，任何情况都是可能存在的，甚至我们可以找到两条链，得到同一个候选数既可以真又可以假。你可能会说，这不是不可能吗？这是可能的，原因在于我们是通过我们自己设定的假设前提才得到的结果，而假设和结论实际上并没有直接的关系。所以，链的头尾可以同真。

从数学的角度出发，高中你一定学过命题与逻辑一节，我们提到过，原命题和否命题的真值无直接关系（也就是说，原命题不管是真还是假，否命题都无法确定其真假性）。而我们可以注意到，**候选数 a 假得到候选数 b 真**和**候选数 a 真得到候选数 b 假**显然是互为否命题的，而它们并没有直接的真假性的关联，所以一个命题即使能够得到，也推不出另外一个命题。

我们来这一则示例里的摩天楼技巧，再对比题目的解来看，就可以发现，链的头尾确实是同真的。

5	4	1	6	2	8	3	7	9	5	4	1	6	2	8	3	7	9
2	3	8	7	¹ ₄		9	5	6	¹ ₄		2	3	8	7	1	9	5
9	6	7	¹ ₅	4	5	3	^{1 2} ₈	^{1 2} ₈	^{1 2} ₄		9	6	7	5	4	3	8
7	^{1 2} ₅	² ₅	4	8	6	^{1 2} ₈	9	3	7	2	5	4	8	6	1	9	3
8	¹ ₂	6	9	3	^{1 2} ₅	4	5	7	8	1	6	9	3	2	4	5	7
4	9	3	^{1 2} ₈	7	5	^{1 2} ₈	6		4	9	3	1	7	5	2	8	6
3	7	4	8	9	^{1 2} ₅	6	^{1 2} ₈	5	3	7	4	8	9	1	6	2	5
6	8	² ₅	3	¹ ₅	7	9	4	^{1 2} ₈	6	8	2	3	5	7	9	4	1
1	² ₅	9	² ₅	6	4	7	3	8	1	5	9	2	6	4	7	3	8

可以看到， $r5c6(2)$ 和 $r9c4(2)$ 在终盘下确实同真，而题目确实是唯一解的。

1-3-2 链的长度有什么结论吗？

在刚才介绍了相关术语之后，我们就可以把之前的逻辑用强弱关系重新解释一遍了。

设 $r3c8(4)$ 为假，则 $r3c8(4)$ 和 $r3c4(4)$ 有强关系， $r3c4(4)$ 和 $r9c4(4)$ 有弱关系， $r9c4(4)$ 和 $r9c7(4)$ 有强关系。

因为逻辑的推导是“假 \rightarrow 真 \rightarrow 假 \rightarrow 真”这样，设开头为假，并且真假交替出现的，所以AIC具有两个基本特性：

- 链必须以强关系开头、强关系结尾；
- 链强弱关系是交替出现的。

那么，还有一个由这两点可以得到的推论。必须强关系开头、强关系结尾，并且强弱关系交替出现，那么结构只可能是“强弱强”、“强弱强弱强”等等了，那么强关系数量始终会比弱关系数量多一个。那么强关系设有 n 个的话，弱关系就得有 $(n-1)$ 个，那么组合形成的AIC就有 $(2n-1)$ 个。因为 n 是正整数，所以 $(2n-1)$ 就应该是一个单数（正的奇数）。所以，AIC的**长度（Length）**，即AIC涉及的强弱关系的总数，就一定是一个单数。而节点数量一定比长度大1，所以节点数应该为 $2n$ 个，则为一个双数（正的偶数）。

那么，双强链，顾名思义，这种结构必须只能有两个强关系；而它具有两个强关系和一个弱关系，所以双强链的长度一定为3。

1-3-3 “强-强-强”也是链？

再来看一则问题。比如示例上的 $c4(4)$ 。因为 $c4(4)$ 也是共轭对，所以它肯定可以满足强关系的定义：当 $r3c4 <> 4$ 时， $r9c4 = 4$ 。那为什么 $r3c9(4)$ 和 $r9c4(4)$ 就必须得是弱关系？

因为我们在使用链的过程之中，强弱关系是交替出现的。换句话说，真假情况是交替出现的，而当AIC的开头 $r3c8(4)$ 假设为假后，推理到 $r3c4(4)$ 时，就已经是真了，所以 $r3c4(4)$ 和 $r9c4(4)$ 使用的是弱关系的定义。虽然，这里确实也满足强关系这一说法。

也就是说，网上给出的一部分资料，写的是“‘强强强’的标准链结构亦成立”，这种说

法是错误的，它犯了本质性的错误。希望你务必引起注意和重视。

在阐述了基本表达方式后，接下来我们继续介绍双强链结构。

1-4 双线风筝 (Two-string Kite)

1	9	2	4	8	3	6	5	8
5	7	8	1 2	1	6	4	3	1 2
4	3	6	1 2 5	1 5	1 2		1	1 2
7	5	3	7 8	7 8 9	7	8 9	7	7 8 9
6	2	4	1	3	1	5	8	5
8	1	9	7	4	5	7	2	3
9	4	5	1 2 7 8	1 5	1 2	3	6	1 5
3	8	1	7	6	4	2	9	7
2	6	5	7	3	9	5	1	4

如图所示，如果 $r1c9(8)$ 为假，则 $r1c5(8)$ 真， $r3c4(8)$ 假， $r7r4(8)$ 真。但是， $r1c9(8)$ 还可以为真。所以两种情况之下， $r1c9(8)$ 和 $r7c4(8)$ 至少一个为真。故删除两个候选数共同对应的位置，即 $r7c9(8)$ 。

链文本表示如下：

```
r1c9=r1c5-r3c4=r7c4(8) => r7c9 <> 4
```

这个结构称为**双线风筝** (Two-string Kite)。

1-5 多宝鱼 (Turbot Fish Normal Type)

1	2	2	6	5	4 5	9	3	4 5
7 8	7	5	5 8	1	1	5	6	7
4 5	7 8	3	9	5	4 5	7	2	
4 5	6	9	2	7	3	1	8	4 5
2 3	2 3	2	8	1	7	2 3	4	3
5 6	9	5 6	9	8 9		5		6 9
2	1 2	4	3	6	9	8	1 2	5
5	7	6	5	4	2	3 1	7	3
3	1 3	7	5	4	2	9		6 9
6	7 8 9	7	5	4	2	9		3
8 9								9
3	5	1	4	2	8	6	7	3
9								9
7	2 3	2	1	3 1	5 6	4	2 5	8
	9	6	9	5 9			9	
2 3	4	8	7	3	5 6	2 3	2 5	1
6 9			5	9	5 6	5	5	

如图所示, 假设 $r3c4(8)$ 假, 可以得到 $r9c7(8)$ 为真, 所以 $r3c4(8)$ 和 $r9c7(8)$ 至少一个为真, 故删除交集。

$r3c4=r7c4-r7c9=r9c7(8) \Rightarrow r3c7 \neq 8$

这个结构称为**多宝鱼** (Turbot Fish (Normal Type))。

这个英文我好像哪里见到过。对, 在上面第一次出现的“双强链”一词, 英文名写的就是这个。双强链和多宝鱼其实是同一个东西, 而刚才的摩天楼和双线风筝, 也都是属于其中的, 不过被单独列出来的原因是, 结构长相更为特殊。

根据技巧的发现者描述, 摩天楼的两个强关系是产生于同两行或两列的, 而双线风筝的两个强关系则产生于某行和某列。如果不满足上述两个要求的, 就是普通的双强链, 或者叫多宝鱼结构。换句话说, 摩天楼的强关系是“平行”的, 而双线风筝的强关系则是“垂直”的, 而强关系既不平行也不垂直, 只能算是普通的双强链 (多宝鱼) 结构。

1-6 共轭对和强关系的异同

那么, 双强链的基本构型我们就学到这里。之所以阐述双强链, 是因为有两大好处: 一来是让你更清晰地掌握链这个数独技巧, 从较短的链开始学习; 二来, 这种结构由于简单, 所以非常好观察, 只需要我们经常去关注共轭对就好。不过, 根据之前的理论, 我们发现, 共轭对就产生于同一个区域下只有两个位置可填的情况, 而定义恰好也满足强关系的。

实际上, 强关系的定义比共轭对更宽泛一些, 共轭对指的是同区域下只有两个位置可填的相同数字, 或是同一双值格下的两个候选数。但跨区域下的不同数字, 共轭对并不包括在内, 而强关系在之后, 我会告诉大家如何去寻找和使用这样的特殊情况。所以你只需要记住一点, 因为目前的强关系的定义可以完全用共轭对实现, 所以我们就用共轭对来寻找吧! 共轭对找起来会很快的哦!

Part 2 同数链与异数链

在学习新内容之前，我们先来回顾之前一节的内容。

2-1 链的基本推论回顾

之前讲了 AIC 的基本要求。不知道你还记得多少呢？是不是看完上一讲就跑出去玩了呢？

我们来回顾一下上一讲讲过的 AIC 的基本满足的特征和要求：

- AIC 开头和结尾都应为强关系；
- AIC 链头节点设为假，得到的链尾节点为真；
- AIC 内的强弱关系交替存在；
- AIC 的长度是一个单数（正奇数），节点数量则是一个双数（正偶数）；
- AIC 的链头和链尾节点至少有一个正确。

我们拿着这样一些特征来看一下现在我们要讲的东西。

2-2 同数链

5 9	1	8	2	3	4 5	6	4 5	7
3	7	4 5 9	4 5 9	6	1	4 9	2	8
2	4 5 9	6	7	4 5 9	8	1	4 5 9	3
1 5 8 9	6	2 9	3	4 5 7 8	4 5	4 7 9	4 9	1 2
4	3	7	1	2	9	8	6	5
1 5 9	5 8 9	2 5 9	4 5 8	4 5 7	6	4 7 9	3	1 2
8 9	2	3	6	7	5	1	4	
6	4 5 8	4 5	4 5 8	1	2	3	7	9
7	4 5 9	1	4 5 9	4 5 9	3	2	8	6

如图所示，假设 $r1c8(9)$ 为假，则由于 $r17(9)$ 和 $b2(9)$ 的共轭对，顺次推理得到 $r2c4(9)$ 为真。所以 $r1c8(9)$ 和 $r2c4(9)$ 至少一个为真。

$$r1c8=r1c1-r7c1=r7c5-r3c5=r2c4(9) \Rightarrow r2c7 \neq 9$$

这个 AIC 的长度超过了原始多宝鱼结构的长度 3，变为了 5。所以它就不是多宝鱼结构了，它就是一种普通的 AIC：**同数标准链**（简称**同数链**，**X-Chain**）。同数链是指，结构内部只涉及同一种候选数的 AIC。不管链有多长都是可以的。

那么，AIC 只能涉及同一种数字吗？当然不是。AIC 分两种：**同数标准链**和**异数标准链**

(简称**异数链**, **Multidigit AIC**)。接下来就会讲解这样的一些结构。

2-3 异数链

接下来我们来看看不同数字之间, 如何形成强弱关系。

2-3-1 双值格链 (XY-Chain)

1 8 9	6 3	1 4 8	3 7 8 9	2 8 9	5 8 9	7 4 9	
	5 7 8 9	2 4 3		4 7 8 9	1 2 3		6 2 3 9
			6 7 8 9		5 4 8 9	1 4 8	
5 7 8 9	8 4 7	1 3 8			4 7 8 9	2 4 8	6 2 3 9
4 7 8 9	2 1 3	9 1 3			6 4 8 9	3 4 8	1 2 3 9
6 1 3 8		7 1 3			2 4 8 9	4 4 8	9 2 3 9
1 8 9	3 4 8	1 3 8	6 7 8 9	2 5 8 9	7 4 8 9		3 1 3 9
			5 7 8 9	1 4 8 9		6 4 8 9	2 1 3 9
2 1 3 8	9 4 8	1 4 8		5 7 8 9	3 4 8 9	7 4 8 9	

如图所示, 这个链特别奇怪, 图中为什么只有弱关系标注呢? 强关系去哪里了呢? 其实, 强关系隐藏在单元格内, 这个链表述如下:

$r6c2(3)=r6c2-r6c4(1)=r6c4-r9c4(5)=r9c4-r8c6(8)$
 $=r8c6-r8c9(9)=r8c9(3)$

$\Rightarrow r8c2 \times 3$

这条链相当长。我们看一下它的逻辑。假设 $r6c2(3)$ 为假, 则 $r6c2$ 只剩下一个候选数了, 所以 $r6c2(1)$ 就为真了。于是乎, $r6c2(1)$ 使得 $r6c4(1)$ 为假, 然后 $r6c4(5)$ 就为真了, 然后依次推理, 直到最后 $r8c9(3)$ 为真。

很奇特的是, 结构的强关系全部产生于单元格之中, 最终可以得到一个结论, 即 $r6c2(3)$ 和 $r8c9(3)$ 至少一个是为真的, 所以删掉交集, 即 $r8c2 \times 3$ 。

这个结构称为**双值格标准链** (简称**双值格链**, **XY-Chain**)。顾名思义, 结构只涉及到双值格, 强弱关系却依然可以传导。

当然了, 之所以提出来, 因为这个结构非常好观察, 对于唯余流玩家来说, 双值格是非常好观察的。

当然了, 这个文本表述还可以写成这样:

$r6c2(3=1)-r6c4(1=5)-r9c4(5=8)-r8c6(8=9)-r8c9(9=3) \Rightarrow r8c2 \times 3$

候选数也是可以直接写入到括号里面的。这样一来, $r6c2(3=1)$ 这样类似的表示方法就可以一下知道是产生于单元格内的强关系 (或是弱关系) 了, 写起来也比之前的方式简单

一些。

2-3-2 远程数对 (Remote Pair)

4	5		1		6	2	3	7
	7	3				1	6	5
2	6	1	5	7	3	4	8	9
3	1	6				5	7	8
7	2	4	8	6	5	9	1	3
		5	7	3	1	6	4	2
5	4	2	3	1	7	8	9	6
1	3		6	5		7	2	4
6		7			2	3	5	1

远程数对 (Remote Pair) 是最有趣的 AIC 结构了。如图所示，假设 $r1c5(8)$ 为假，按照双值格链的推导方式，可以得到 $r8c6(8)$ 为真；或者假设 $r1c5(9)$ 为假也可以：

$r1c5(8=9)-r1c3(9=8)-r8c3(8=9)-r8c6(9=8) \Rightarrow r2c6, r9c5 \neq 89$

$r1c5(9=8)-r1c3(8=9)-r8c3(9=8)-r8c6(8=9) \Rightarrow r2c6, r9c5 \neq 89$

所以这个技巧的头既可以是 8 也可以是 9，所以可以删除两个不同的数。而且，这个技巧还能被拆解为两个摩天楼技巧。

$r1c5=r1c3-r8c3=r8c6(8) \Rightarrow r2c6, r9c5 \neq 8$

$r1c5=r1c3-r8c3=r8c6(9) \Rightarrow r2c6, r9c5 \neq 9$

如图所示。需要注意的是，我们甚至可以认为， $r1c5(89)$ 和 $r8c6(89)$ 的填数一定不同，所以可以构成跨区数对结构。不过，这一点需要你自己理解和推理为什么。

4	5		1		6	2	3	7
	7	3				1	6	5
2	6	1	5	7	3	4	8	9
3	1	6				5	7	8
7	2	4	8	6	5	9	1	3
		5	7	3	1	6	4	2
5	4	2	3	1	7	8	9	6
1	3		6	5		7	2	4
6		7			2	3	5	1

2-3-3 首尾异数链 (XY-X-Chain)

接下来介绍一种 AIC，刚才的 AIC 虽然异数，但头尾起码是同数的，现在介绍的 AIC，连头尾都是异数的了。

2-3-3-1 先来看看标准版长啥样

7	^{2 3}	^{2 3}	5	4	8	6	1	9
1	5	8	7	6	9	^{2 3}	^{2 3}	4
4	9	6	2	1	3	7	5	8
2	7	1	6	5	4	9	8	3
3	8	4	9	7	1	5	6	2
⁵	6	⁵	3	8	2	1	4	7
6	4	²	1	3	²	²	1	5
⁵	^{1 2 3}	^{2 3}	4	9	6	^{2 3}	^{2 3}	1
⁵	¹	³	2	5	4	³	⁷	6

如图所示，链写法如下：

$$r9c4=r9c1(8)-r8c1(8=5)-r8c9=r7c9(5-1)=r7c4(1) \\ \Rightarrow r7c4 \neq 8, r9c4 \neq 1$$

等下，头尾都不一样，那怎么删数呢？

试想一下，之前我们讲的 BUG + 2 里到底是如何删数的。AIC 有个特征，就是首尾节点至少一个正确，这恰好符合之前 BUG + 2 的“真数必须至少有一个是正确的”的要求，所以我们删除的是两个真数都可以看到的地方（即交集），而这个题跟 BUG + 2 基本上没有差别，“即 $r9c4(8)$ 和 $r7c4(1)$ 至少有一个是正确的”是完全一样的思路。所以，我们可以套用之前的逻辑： $r9c4(1)$ 应不应该是这两个节点的交集？如果 $r9c4(8)$ 为真，那 $r9c4$ 自然就不能是 2 了；如果 $r7c4(1)$ 为真，当然也不能让 $r9c4$ 填 1。所以 $r9c4(1)$ 确实应当是这两个节点的交集。当然了，类似地， $r7c4(8)$ 也是这样的。

这个结构是一种特别特殊的 AIC 了，它叫**首尾异数标准链**（简称**首尾异数链**，XY-X-Chain）；另外，如果这种链的删数只有一处而不是像图上有两处的话，这种结构就叫**不连续环**（Discontinuous Nice Loop）。例如下面的这两则示例。

2-3-3-2 不连续环 (Discontinuous Nice Loop)

	3	1	3		1		3	2	1		3
4				6	4	6	8		7	5	6
7	3	1	3		1	2	2	3	1		3
7	8		8	5		6	6	6		8	6
	3				1		6	9	1		3
4		2		6	4	6		5	7	9	6
7	8			9	7	9	7	9	7	8	8
9	6	2	8	3	7	4		1	1		
5	5	3	2	1	4	6	9	7			
1	7	4	5		6	6	8	3	2		
	3	3	1		2	2	6	4			
5	5			6	6	6	4	7	7	8	8
6	9	7	3	4	8	5	2	1			
2	4	8	7	5	1	3	6	9			

如图所示，假设 $r1c8(5)$ 为假，则可以通过该链得到 $r7c8(7)$ 为真。

$$r1c8=r4c8-r4c9(5=4)-r7c9(4=8)-r7c8(8=7) \Rightarrow r1c8 \neq 7$$

不过这种结构显然只能删一个数，即 $r1c8(7)$ 。

2-3-3-3 不连续环的变体

接下来我们来看一个更为“变态”的构型。

1		3	3	4		3	5	8	2	3	2
	7	6	6		6	9			6	9	6
4		3	3	2	1	3	7	1	3	3	1
	5	6	5		6	6		5	6	6	5
	8	8	8		8	9			9		9
	3	3			3	1	3	1	3		
5	6	5	6	2		6		4	7	1	5
8	8	8	8		8	9			9		9
	1	1			1	2		6		4	3
5	4	5	5		5	5		5	4		
7	8	8	7	8	8	9	8	7			
	3	3	3		3		2	3			
5	6	4	6		5	6		9	1	5	6
8	8	8	8		8				8		
2	9		5		5	1	3	5	6	4	6
	7	8	7	8	8			7	7	8	
	3	1	3		3	3	3		3	1	6
5	6	5	6	4	5	6	5	6	9	7	9
7	8	7	8		7	9					
	3				3	6	4	8	1	3	5
7	9	7	9	7	9	7	9	7	6	7	9
	3	3	3		2	3	2	3	3	6	
5	6	5	6	1	5	6			7	8	4
7	9	7	9		9						

如图所示，这条链看起来很奇怪，开头都不好找，而且就图上那样，大多画出来的都是强关系，更难找到链头了。实际上，链表示如下。

$$r4c5=r4c8(2=4)=r4c2-r5c2=r5c6(4=2)=r4c5(2) \Rightarrow r4c5 = 2$$

这条链从 $r4c5(2)$ 开始，然后又转回来了，又到了 $r4c5(2)$ 。而且首尾确实都是强关系，而且也符合链的逻辑，但为什么链结构的结论是出数了呢？我们这么去理解：假设 $r4c5(2)$ 为假，绕了一圈回来发现 $r4c5(2)$ 为真，则说明我们得到了矛盾，进而得到假设错误。而假设是 $r4c5(2)$ 为假，如果它是错误的，那么结论就应该是 $r4c5(2)$ 为真，所以 $r4c5 = 2$ 。

这种结构实际上是一种比较麻烦和难受的链结构，虽然它能直接得到出数，但因形式很特殊，绕了一圈得到了矛盾，这一点和试数非常接近和相似，所以我们不建议大家使用这个技巧，仅理解其逻辑即可。

2-3-3-4 等会儿，删数和链的头尾也有强弱关系？

这个技巧之所以被归为不连续环，是因为它实际上把不连续环的逻辑“反转”了一下。我们先来思考一个问题。链尾和删数结论到底是什么样的关系。我们可以得到，因为链头为假，得到了链尾为真的结果。这便意味着，因为链尾为真，所以才得到了删数结论，所以删数跟链尾实际上为弱关系，即链尾真得到删数假；不过还没完。

如果我们把整条链的过程重新调整一下，从删数开始推的话，因为链头最开始是从“为假”开始推的，所以我们如果还要往后倒退一个节点的话，从删数作为链头的话，我们只得设删数节点为真，然后继续推理，这个结构整体才能持续到链尾；否则到链头节点的时候，就已经和我们最开始的设定相反了，就没办法继续进行了。

4	3	1	3		6	4	6	8	3	2	5	6
	3	1	3	5		1	6	2	2	3	1	3
7	8		8			7	9	7	9	7	9	8
4	3	2			6	4	6	6	5	1		3
7	8		9		9	7	9	7	9	7	9	8
9	6	2	8	3	7	1				4	5	4
5	5	3	2	1	4	6	9	7	9	7	9	8
1	7	4	5			6	9	6	9	8	3	2
5	3		1		6	2	6	2	6	4		7
6	9	7	3	4	8	5	2	1	3	6	9	
2	4	8	7	5	1	3	6	9				

所以如图所示，这个画法的完整版就是上面这样的结构，而从删数节点开始推理，并设它为真，而推了一圈后发现删数节点是为假的，所以与假设违背。由于假设的是“为真”，所以结论就应该为“删数节点为假”，即删掉它。

这一点确实有些绕，不过思路依然是很清晰的。可以从刚才的说明文字里看到，实际上，不连续环一共是两种类型：同一个单元格弱关系开头弱关系结尾（我们把链延长了才得到的效果）和同一个单元格强关系开头强关系结尾，所以我们才说后者也属于不连续环。属于不连续环的原因并不是在于异数，而是推导逻辑和思维方式。另外请记住，删数和链的头尾节点都是形成弱关系的。

讲完了基本的思路后，接着我们再来看一种更诡异的结构。

2-3-4 自噬链（Cannibalistic AIC）

5	2	1	7	3	8	6	4	9
6	4	9	2	1	2	8	3	7
7	8	3	6	4	9	1	5	2
4	7	5	1 2 3	9	1 2 3	2	6	3
1	9	5	2 3	6	2 3	4	7	8
2	3	6	1	8	7	5	9	4
9	1	7	8	5	4	3	2	6
3	5	4	1	2	6	7	9	8
8	6	2	9	7	1	4	8	5

如图所示，链表表述如下：

$r5c7(7=2)-r4c7(2=6)-r4c8=r1c8(6-4)=r5c8(4-7)=r8c8(7)$
 $\Rightarrow r5c8, r8c7 \langle \rangle 7$

这条链看似很普通很简单，不过它有一些奇特。链头是 $r5c7(7)$ ，设定为假，得到链尾 $r8c8(7)$ 为真，所以两端至少有一个为真，于是删除交集。

此时我们会发现删除的交集有一个数是 $r5c8(7)$ ，它是链内的一个节点。为什么这个数也能删掉呢？因为我们利用了链才得到的首尾至少一个为真的结论，而不是因为删数才破坏的链结构，所以结构成立后才产生了删数。

在 SDC 一节里我们提到了自噬，而这种链也具有类似的效果，所以它也就称为了**自噬标准链**（简称**自噬链**，**Cannibalistic AIC**）。

2-3-5 普通的标准链

4 7	2 4 7	2 9 7	6	8	2 5	1 7	1 5 7	3 1 5	1 3 5
6	5	8	7	1	3	9	4	2	
3	1	2 7	4	2 5	9	7 8	6 8	5 8	6
4 5 9	4 8 9	3 5 6 9	1	5	3 5 6	4 6 8	2 5 8 9	7	
4 5 7	4 7 8	1	2	9	7 5 6	3	5 8	4 5 6	
2	3 7 9	3 5 6 9	8	5 7	3 4	1 6	1 5 9	1 5 6 9	
1 7	2 9 7	2 9 7	5	4	1 2 7	1 2 7	6	8	
8	6	4	3	2 7	1 2 7	5	1 2 7	9	
1 5 7	2 7	2 5 7	9	6	8	1 2 4 7	1 2 3 7	1 4 3	

如图所示，链结构如下：

$$\begin{aligned}
 & r4c6(5=6) - r5c6 = r5c9 - r3c9(6) = r3c7(6-8) = r3c8 - r5c8(8=5) \\
 & \Rightarrow r4c8, r5c6 \leftrightarrow 5
 \end{aligned}$$

证明过程和思路我就不再重复了，思路则是完全一致的，根据链的思路往下推导即可。

2-3-6 Wing 结构

Wing 在之前我们已经接触到了一部分结构，接下来我们来看几则也叫做 Wing，但由来的不相同的结构。

这一部分的技巧名字特别不好记，而且是当时创造链体系之前的产物，而现在它们基本上已经被废除，所以这里仅供参考。不过你需要掌握 W-Wing 和 M-Wing。当然，还有 XY-Wing 的额外介绍内容。

2-3-6-1 XY-Wing

实际上，XY-Wing 除了之前的分情况讨论的逻辑以外，我们还能把它视作链结构。我们来看一下这个结构到底是怎么转化为链的。

	3		3	6		2	1	4	5	9
7		8			7	8				
	4		4			5	3	6	1	2
7	9	8	9	8	7	8				
5	2	1		6	9	4	3		7	8
1			5	2	4	6		7	8	9
	9	8	9					3	1	
2	3		3	7	1	8	9	5	4	6
	4									
1	2		6	4	5	3	7	2	2	1
	9			8				8	9	8
4	5	2	9	7	8	1	6	3		
8	1	3	4	6	2	5		2	2	5
6	7	9	3	1	2	5		2	4	4
							8	8	8	8

如图所示，我们将之前的 XY-Wing 的示例改写为链。我们将其中一个分支“反向”，然后形成了一个双值格链：

```
r5c2(4=3)-r1c2(3=8)-r2c3(8=4) => r2c2, r6c3 <> 4
```

实际上，任何的 XY-Wing 都可以使用如上方式进行转化。

2-3-6-2 W-Wing

1 3	6	1 3	5	9	2	4	3	8
7	5	4	1	6	8	2 9	7	2 3
7	9	8	2	7	4	3	6	5 1
6	7	1 5 9	3	1 2 5	5 9	2 9	8	4
1 2 3	1 2	1 5 3 9	2	1 2 5 8	4	7	6	2 3 9
8		9	8 9	8 9				9
2 3	2	3	2	7	6	1	2 3	5
4	8	9	8 9	8 9				
1 2	1 2	7	2	2 3	5	2 3	1 2 3	6
4	4	9	4	8 9	8	8 9	4	
1 2	3	1	6	2	7	2	1 2	2 9
4	8	8 9		8		8 9	4	9
5	2	6	2	2 3	1	2 3	2 3	2 3
	4	9	4	8 9	8	8 9	4	7 9

链表示如下：

$r4c7(9=2)-r2c7=r2c9-r8c9(2=9) \Rightarrow r5c9, r789c7 \neq 9$

假设 $r4c7(9)$ 为假，则由双值格和 $r2c7(2)$ 的共轭对，就可以顺次得到真假情况，最终得到 $r8c9(9)$ 为真。所以链头和链尾至少一个为真，删掉交集。结构的首尾两格是一样的候选数，并带有一个强关系的，就是 W-Wing 结构了。在观察的过程之中，只需要找到两个双值格，候选数也完全一样，再有一个行列上的共轭对，基本上就可以照着结构依葫芦画瓢。

2-3-6-3 Y-Wing? W-Wing?

在中国，Y-Wing 它被当成 W-Wing 的别名；而在外国，Y-Wing 则是被看成 XY-Wing 的别名。这里讲一下为什么中国的 W-Wing 有个别名 Y-Wing。

很多资料上把 W-Wing 技巧名的后面打了个括号写着 Y-Wing，或是 Y-Wing 后给了括号写的 W-Wing。而 Y-Wing 有现如今的结果，系误传所致。

Y 是 XY 的一种简写形式，用来代替两种未知数。有些资料将技巧的英文名写作 Y-Wing，这么说是不正确且不严谨的。因为 Y-Wing 技巧原名而言，被误传的原因是由于 Y-Wing (XY-Wing) 结构内强关系只有三个，所以由此结构拓展出的所有只有三个强关系的链都称为 Y-Wing Style，“Y-Wing Style”中文直译为“Y-Wing 拓展构型”，此时，Y-Wing 拓展构型就包括了 W-Wing。但此时 Y-Wing 并不只是包含 W-Wing 这一种结构，而 XY-Wing 等，均属于此归类。

但是，因为引入技巧的时候，大家不是特别熟悉这一点，刚开始使用的时候并未注意到，也就“将错就错”了；现在中国反而比较习惯于称呼 Y-Wing 一些。

一个解释 Y-Wing 结构的链接是 SudokuWiki 所给出的 [XY-Wing](#) 的链接。

2-3-6-4 M-Wing

7	² _{5 8}	3	² _{5 9}	² _{4 5}	1	⁵ ₉	⁴ _{5 8 9}	6
¹ _{2 4}	¹ _{2 5 6}	² _{4 5 6}	7	² _{4 5}		¹ _{5 9}	3	⁴ _{5 8}
¹ _{4 8}	9	⁴ ₅	6	⁴ ₅	³ ₈	2	¹ _{4 5 8}	7
² _{3 9}	² _{3 6}	² _{6 9}	4	7	5	8	⁶ ₉	1
¹ ₉	¹ ₉	⁶ ₉	7	3	8	2	⁴ _{5 6 9}	⁴ ₅
5	4	8	1	9	6	3	7	2
³ _{8 9}	³ ₈	1	⁵ ₉	6	4	7	2	³ _{5 8}
² _{3 4 8 9}	² _{3 4 5 9}	² _{5 9}	² _{5 9}	¹ _{2 3 5}	³ ₉	¹ _{5 6 8}	¹ _{5 8}	³ _{5 8}
6	² _{3 5}	² ₅	8	¹ _{2 3 5}	7	4	¹ ₅	9

链表示如下:

$r5c2(1=6)-r5c7(6)=r8c7(6-1)=r2c7(1) \Rightarrow r2c2 \neq 1$

2-3-6-5 Split-Wing

¹ ₄	² ₄	¹ ₆	² ₆	5	8	7	9	3
² ₆	9	3	² ₆	7	4	1	8	5
5	7	8	9	3	1	4	6	2
⁷ ₆	1	⁷ ₆	5	8	2	9	3	4
⁴ _{3 4}	³ ₄	5	1	6	9	² ₈	² _{7 8}	
9	8	2	3	4	7	6	5	1
³ ₈	5	4	7	2	6	³ ₈	1	9
² _{3 7}	² ₃	9	⁴ ₈	1	5	² ₃	2	6
⁷ _{1 2 8}	¹ ₂	6	¹ _{4 7}	9	3	5	⁴ _{7 8}	

链表示如下:

$r9c1=r9c8(2)-r5c8(2=7)-r8c8=r8c1(7) \Rightarrow r8c1 \neq 2, r9c1 \neq 7$

2-3-6-6 Local-Wing

9	1 2 7 8	1 2 7 8	4 7	6 7	5	2 7 8	1 7	6 7	3	1 7
4	2 3 7	5 7	1	5 7	6 7	2 3 7	9	5 7	6	8
1 3 5	1 3 5	6	9	3	9	3	4	1 5 7	2	
1 3 5 6	1 3 5 8	9	7	4 5 8	1 3 5 6	2	1 3 4 5	3		
5 7	5 7	4	2	1	6	8	7 9 7	5 3 9		
1 2 5 6	1 2 5	9	4 5 7 8	3	5 8	1 5 7	5 6 7	1 5 7		
8	1 5 6	3	5 6 7	4 6	1 7	5	2	1 4 7	5 9	
7	1 2 5 6	1 2 5	3 5 6	4	2	9	1 3 5	8	1 3 4 5	
1 2 5	4	9	5 7	8	1 2 3 5	1 3 7	1			6

链表示如下：

$r6c4=r1c4(4-6)=r1c7-r2c8=r6c8(6) \Rightarrow r6c8 \neq 4$

2-3-6-7 Hybrid-Wing

5	4 5	6	5	2	4	3	1	3	
7	9		9				8	7 8	
2	3	8	6	1	7	5	4	9	
1	4 5 9 7 9		5 9	8	4	3 7	6	2	
	6 9	2 9	3	7	5	1 2	4	1 8	6
8	7	1	3	4	6	2	9	5	
5 6	2 5	4	8	9	1 2 7	3 1 7	3	7	6
	7 9	1	2	4	6	5 9	8	5 7	3
4	8		2	3	5 9	6	5 7	1	
3	6	5	1	7	8	9	2	4	

链表示如下：

$r6c1(5=6)-r6c9(6=7)-r1c9=r1c1(7) \Rightarrow r1c1 \neq 5$

那么，基本的结构就讲完了，接下来来看总结。

2-3-6-8 总结

下面陈列上述结构的基本链构型。

2-3-7-2 #2: 一些真的练习题

再来看几则题目，这几道题仅通过之前学到的链技巧和直观类技巧就可以完成解题。

1	6	3	2	8	9	7	5	4	1	5	6	8	7	4	9	3	2		
7	9	² ₅	4	⁵ ₆	3	8	² ₆	1	³ ₉	4	7	6	2	³ ₉	1	8	5		
4	8	² ₅	1	⁵ ₇	⁶ ₇	³ ₆	² ₃	⁶ ₉	2	8	¹ ₃	¹ ₅	¹ ₃	¹ ₅	4	7	6		
6	3	9	7	1	4	2	8	5	⁴ ₆	¹ ₂	¹ ₂	¹ ₂	³ ₄	¹ ₂	³ ₆	5	9	⁴ ₇	
8	4	7	5	² ₆	² ₆	¹ ₆	¹ ₆	3	7	3	⁴ ₉	⁴ ₉	⁵ ₉	² ₅	² ₅	6	1	8	
2	5	1	³ ₆	³ ₆	8	4	7	⁶ ₉	8	¹ ₆	5	¹ ₄	¹ ₇	¹ ₆	¹ ₇	3	2	⁴ ₇	
5	2	8	³ ₆	4	1	³ ₆	9	7	⁴ ₅	¹ ₂	¹ ₂	¹ ₂	¹ ₂	¹ ₂	¹ ₂	² ₈	⁵ ₆	3	
9	7	4	8	³ ₆	5	¹ ₃	¹ ₃	2	³ ₆	7	¹ ₂	¹ ₂	¹ ₂	¹ ₂	¹ ₂	² ₈	4	9	
3	1	6	9	² ₇	² ₇	5	4	8	³ ₅	² ₆	² ₃	² ₃	² ₃	² ₃	4	8	7	⁵ ₆	1

2	6	4	9	1	3	5	8	7	⁴ ₉	² ₄	² ₉	² ₆	3	5	1	7	8	⁶ ₉	
⁷ ₉	3	1	⁴ ₇	8	5	⁴ ₆	⁴ ₆	2	8	5	7	6	2	9	3	4	1	⁶ ₉	
⁷ ₉	5	8	⁴ ₇	² ₇	6	3	⁴ ₉	1	1	³ ₉	³ ₆	8	7	4	⁵ ₆	⁵ ₉	2	⁶ ₉	
4	7	9	6	3	¹ ₂	¹ ₂	5	8	5	³ ₇	9	1	6	2	8	⁷ ₃	4	³ ₉	
1	2	6	8	5	4	7	3	9	6	8	1	⁵ ₇	4	⁵ ₇	³ ₂	⁵ ₇	³ ₅	³ ₉	
3	8	5	² ₇	9	¹ ₂	¹ ₂	² ₄	6	⁴ ₂	² ₃	² ₃	² ₃	⁵ ₇	³ ₉	8	1	6	⁵ ₉	
5	9	3	1	² ₇	8	² ₆	² ₆	4	7	1	8	⁴ ₅	³ ₉	⁵ ₆	⁴ ₅	⁶ ₉	2	⁵ ₆	³ ₉
8	4	² ₇	5	6	² ₇	9	1	3	⁴ ₂	² ₃	² ₃	² ₃	² ₄	² ₅	³ ₇	⁴ ₅	⁶ ₉	³ ₈	³ ₉
6	1	² ₇	3	4	9	8	² ₇	5	⁴ ₂	² ₃	² ₃	² ₃	² ₄	² ₅	³ ₇	⁴ ₅	⁶ ₉	1	7

7	6	2	8	1	3	5	9	4	1	7	8	6	⁴ ₂	9	⁴ ₂ ³	5	⁴ ₂ ³
⁴ ₅ ³	³ ₅	³ ₄ ⁵	7	6	9	1	2	8	9	3	4	1	² ₈	5	² ₈	6	² ₈
1	9	1	4	2	5	3	6	7	2	5	6	7	⁴ ₈	3	⁴ ₈ ⁹	1	⁴ ₈ ⁹
¹ ₄	7	6	5	9	8	2	¹ ₄	3	7	9	3	5	² ₈	6	² ₈	4	1
9	⁵ ₈	⁴ ₅ ⁸	1	3	2	⁷ ₆	⁴ ₅ ⁷	⁵ ₆	6	4	1	² ₈	3	7	5	9	² ₈
¹ ₂ ^{3⁵}	² ₃ ⁵	¹ ₅ ³	6	7	4	8	¹ ₅	9	8	2	5	9	1	4	7	3	6
² ₈	² ₈	7	9	5	6	4	3	1	5	6	7	3	² ₈ ⁹	1	⁴ ₈ ⁹	² ₈	² ₄ ^{8⁹}
⁵ ₆	4	9	3	8	1	⁷ ₆	⁵ ₇	2	4	1	² ₉	² ₈	7	5	² ₈ ³	6	² ₈ ⁹
³ ₅ ⁶	1	³ ₅	2	4	7	9	8	⁵ ₆	3	8	² ₉	4	² ₉	6	1	7	5

	<small>1 6 9</small>	2		<small>3 4 7</small>	<small>6 9</small>	<small>1 3 4 6 9</small>	<small>3 8</small>	5	<small>4 6 8 9</small>
	<small>6 9</small>	4	5	2		<small>3 6 9</small>	1	<small>3 6 7</small>	<small>6 8 9</small>
5	<small>1 7</small>	3	8	<small>4 6 7</small>	<small>1 4 6 9</small>	2		<small>6 7</small>	<small>4 6 9</small>
3	9	8	6	5	7	4	1	2	
2	5	7	4	1	8	6	9	3	
4	6	1	<small>2 3</small>	9	<small>2 3</small>	7	8	5	
1	3	9	<small>2 7</small>	<small>4 6 7 8</small>	<small>2 4 6</small>	5	<small>2 6 8</small>	<small>6 8</small>	
8	2	6	1	3	5	9	4	7	
7	4	5	<small>2 9</small>	<small>6 8</small>	<small>2 6 9</small>	<small>3 8</small>	<small>2 3 6</small>	1	

每一题涉及的技巧都不一样，可以尝试练习观察它们。

Part 3 守护者 (Guardian)

3-1 守护者的基本逻辑

9	5	1 4	6	3 7	3 7	2	8	1 4
2 3 4	1 2 3 8	1 2 3 7 8	1 2 4	9	5	1 4	6	3
2 3 7	6	1 2 3 7	1 2	4	8	9	5	3
6	9	1 2 4 7	8	1 2 7	4 7	3	5	
3 1 3 4	5	9	6	8	1 4	2		
8	1 2 3 7	1 2 3 4	5	1 2 3 7	4 7	9	6	
1	4	6	7	5	9	3	2	8
2 3 8	2 3 8	2 3 8	4	6	1	5	7	9
5	7	9	3	8	2	6	1 4	1 4

如图所示，我们发现，如果把 $r2c3(4)$ 和 $r6c7(4)$ 都去掉后， $r25$ 、 $c17$ 和 $b4$ 五个区域就会出现关于数字 4 的共轭对，而且恰好成环。那么这会发生什么样的奇妙情况呢？

因为链需要交替存在，所以我们随便选取一个节点作为链头，比如 $r5c1(4)$ ，向上走链结构。我们之前说过一点。强关系的定义下的两个候选数，是一定满足弱关系的定义的，但该用强关系还是弱关系，取决于关系的交替出现情况。也就是说，共轭对肯定是强关系了，不过以 $r5c1(4)$ 作为链头向上引出 AIC 的话，那么 $r2c1(4)$ 和 $r2c7(4)$ 应为弱关系。随即出现以下链结构：

```
r5c1=r2c1-r2c7=r4c7-r5c8=r5c1(4) => r5c1 = 4
=> r2c1, r5c8 <> 4
=> r4c7, r2c7 = 4 (c7 矛盾)
```

当我们得到 $r5c1 = 4$ 时，同时可以知道的是 $r2c1$ 和 $r5c8$ 都不填 4，于是 $r2$ 和 $b6$ 之中， $r4c7$ 和 $r2c7$ 就应当填 4。此时发现， $r24c7(4)$ 同列，所以矛盾。所以，直接“裸露”出来的五个共轭对首尾成环是完全不可以的，也就是说， $r2c3(4)$ 和 $r6c7(4)$ 不同时为假。所以 $r2c3(4)$ 和 $r6c7(4)$ 至少有一个为真，于是删除交集，故 $r2c7, r6c3 <> 4$ 。

这个技巧称为**守护者 (Guardian)**。很不幸的是，它的逻辑很复杂。接下来我们再来看一则示例。

9	¹ ₆	¹ _{5 6}	¹ ₅	4	2	3	8	⁵ ₆
8	4	¹ _{5 6}	3	9	⁵ ₆	⁵ ₆	¹ _{5 6}	2
7	3	2	¹ ₅	⁵ ₆	8	4	¹ _{5 6}	9
5	7	9	6	2	3	1	4	8
3	2	¹ ₆	4	8	¹ ₅	9	⁵ ₆	⁵ ₆
4	¹ ₆	8	9	⁵ ₆	¹ ₅	⁵ ₆	2	3
6	5	7	8	1	9	2	3	4
2	8	4	⁵ ₆	3	⁵ ₆	⁵ ₆	9	1
1	9	3	2	⁵ ₆	4	8	⁵ ₆	⁵ ₆

如果{r2c7, r3c5, r68c6}(5)全部去掉后,剩下的 r68、c57 和 b8 里存在五个连续的共轭对,并使之成环。显然,这一点是不允许出现的,所以假设矛盾,即 r2c7, r3c5, r68c6 里必须至少有一处是填入 5 的;因此删除交集,即 r2c6 <> 5。

接下来我就来讲一下原理,这会让你觉得技巧寻找和理解起来更加轻松一些。

3-2 守护者理论的证明

那么,为什么说,只有大于 3 且是奇数个共轭对两两连接、首尾成环时,结构必然不存在呢?实际上,我们可以如此思考。

共轭对要求两处填数必须一真一假,那么一旦有一个数为真,那另外一个就必须为假,反之亦然。如果有奇数个的话,顺次推导下去,必然会遇到开头和结尾两处的数同真假。而结构成环,而且还是共轭对,所以这两处必须也要求真假性相反(一个真一个假)。所以这已经违背了要求,所以该结构的内部是无解的(无法找到合适的填法)。

这一点和致死形式类似,但不同。BUG 结构的致死形式确实内部无解,但 BUG 属于致命结构一节的内容,它基于题目唯一解,而此技巧跟唯一解要求无关。

3-3 死环 (Guardian Pair)

我们再来看一则奇特的示例。

2	6	8	7	9	5	3	4	1
9	^{1 3} ₅	¹ ₅	6	³ ₄	³ _{4 8}	2	7	⁵ ₈
⁵ ₇	³ _{5 7}	4	1	^{2 3} ₈	^{2 3}	9	6	⁵ ₈
¹ _{5 6}	^{1 2} _{5 9}	^{1 2} _{5 9}	² _{5 9}	7	^{1 2} _{6 9}	8	3	4
¹ _{4 6}	^{1 2} _{4 9}	3	² _{4 8 9}	² _{4 6}	^{1 2} _{4 6 8 9}	7	5	² ₉
^{4 5} _{7 8}	² _{4 5 7 9}	² _{5 9}	^{2 3} _{4 5 8 9}	^{2 3} _{4 5}	^{2 3} _{4 8 9}	1	² ₉	6
¹ ₄	^{1 2} _{4 9}	6	^{2 3} _{4 9}	8	^{2 3} _{4 9}	5	^{1 2} ₉	7
¹ ₅	8	7	² _{5 9}	² _{5 6}	² _{6 9}	4	^{1 2} ₉	3
3	² _{4 5 9}	² _{5 9}	² _{4 5 9}	1	7	6	8	² ₉

如图所示，感觉略熟悉：感觉有点像是 UR 区块类型的感觉，也有点像是远程数对。它的逻辑是这样的：

如果 $r69c3(5)$ 同假时， $r69$ 和 $c3$ 均会产生 2、9 数对，在删除所有其它位置的 2 和 9 后，2 和 9 构成由 5 个共轭对首尾拼接成环且独立的守护者结构，这种结构显然是不成立的，因为不论怎么填数，始终会存在同一个区域的两个单元格的填数一致，这样便产生了矛盾。所以原假设错误，故 $r69c3(5)$ 不可同为假，即 $r69c3(5)$ 至少一个为真。所以， $c3$ 的其余单元格都不应该填 5，即 $r24c3 \neq 5$ 。

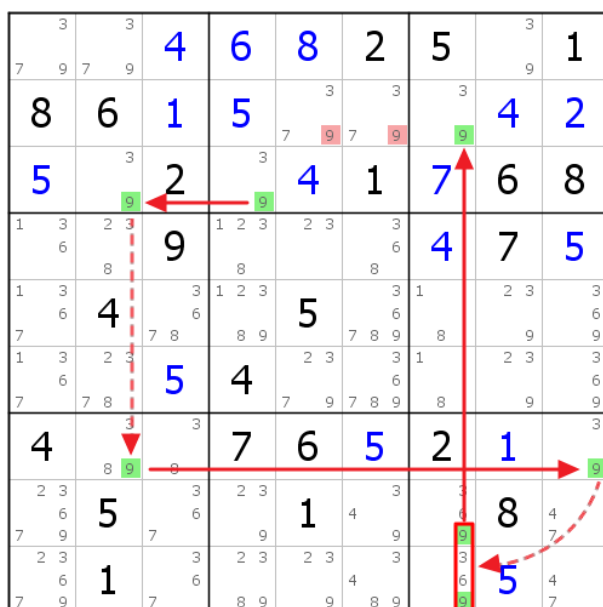
这个结构利用到了数对的删除的特性，随之产生共轭对首尾成环的守护者的出错情况。它使用了两个结构长相一模一样，但候选数不同的守护者结构 2 和 9。这个技巧叫做**死环 (Guardian Pair)**，意味着 $r69c3(5)$ 同假时，内部结构有两个不同数字的守护者合并在一起形成了无法填数的死亡局面。

Part 4 区块链 (AIC With Locked Candidates)

接下来我们来看一种新结构，在此之前，我们必须引入一个新的节点类型：区块。

4-1 区块节点的引入

来看这一则示例。



如图所示，我们可以观察到链头是 r3c4(9)。按照链的逻辑，我们写出文本写法。

$r3c4=r3c2-r7c2=r7c9-r8c7=r2c7(9) \Rightarrow r2c56 \neq 9$

当推理到 r7c9(9) 的时候，按照顺序，这里应该是为真的，但发现 b9 里含有三个候选数 9，按照之前的链的逻辑，我们只可以让其中一个 9 为假，然后继续推理。但实际上，我们不论指定链从 r8c7 继续延伸还是 r9c7 继续，都无法继续推理，因为我们此时仅能让 r8c7 或 r9c7 一个单元格的候选数 9 为假，但 c7 有三处 9，我们不可能因为一处 9 为假就否定了剩下两处的 9，所以我们不得不采用一种逻辑：既然单独看 r8c7 或 r9c7 都不行，干脆我们就直接把它们看作一起，当成一个区块。

假设到 r7c9(9) 为真时，便同时可以使得 r8c7(9) 均为假。此时，当它们都为假后，我们才可以继续向下推理，c7 只有三处 9，其中的两处都为假了，自然只能让 r2c7(9) 为真。那么链得以继续持续到链尾，并且确实拥有删数，且删数是正确的。这条链就此分析完成，并且我们就引入了一种新的节点类型：区块。

不过区块不是整体分析的吗？如此分析已经拆解区块结构了，或者换句话说，实际上前面的逻辑是分了两个“支线”，让 r7c9(9) 同时分别用虚线箭头（弱关系）指向 r8c7(9) 和 r9c7(9)，这并不能说就算真正的区块结构了。那么，既然是节点，就有真假性，而按照之前的逻辑，既然看作整体，那么区块这种类型的节点为真或为假都分别表示哪一种情况呢？下面我们来看看下面这一则示例里，我们会得到什么新鲜的东西。

4-2 区块节点为假？

8	6	5	9	2	1	3	2	2
		7		5			4	5
4	2		9	3	2	6	1	2
	7				5		2	1
3	2		1	4	7	8	5	2
	5						8	7
7	3	8	2	1	9		2	6
			5				4	5
5	4	2	6	8	3			1
							7	9
1	9	6	2	4	7		2	3
			5				5	8
9	8	3	1	6	5		4	7
								2
6	5	4	7	3	2		1	5
	7		8				5	8
2	1	5	7	9	4	6	5	3
		7	8				8	

如图所示，这条链和之前的例子不一样的地方是，它从区块节点开始推导的。

$r3c79=r3c2(5-2)=r2c2-r2c5(2=5) \Rightarrow r2c789 <> 5$

那么，初始状态要求区块节点为假，那么区块节点为假是什么？我们回顾一下之前的理解：区块为真是什么，区块为假又是什么。区块是一种结构，我们人为定义的候选数节点为“真”就表示这个节点就填这个数，这个数是对的；那么相反，为“假”就表示不填，删除这个数。可以看到，它们是互斥的。那么区块节点为真，我们人为规定为真即表示“这个区块成立”，即区块所属的单元格里必须**有一个单元格**就填入这个数。那么为假就取这个说法的相反情况，即“没有单元格填这个数”或“有至少两个单元格填入这个数”

当然，你也可以这么去想：区块为真表示区块成立，那区块为假则表示区块不成立，那么就思考哪些情况属于区块不成立。

而显然，我们这里指代的区块结构是不可能出现“至少两个单元格填入这个数”的情况的，所以只可能是“没有单元格填这个数”。那代入到图里，就表示 $r3c79$ 都不能填 5。那这么一来，链就成立了： $r3c2 = 5$ ，即得到了“区块节点为假”得到“候选数节点为真”的结果，即从假过渡到真，此时这个形式为强关系。那么，后面的逻辑就不难了，并最终得到了 $r2c789(5)$ 全部为假。

从例子里，我们得到了一些结论，下面来总结一下。当区块作为节点时：

- 节点为真表示区块成立，即**区块涉及的单元格里恰好有一个单元格填入该数字，即只有一处候选数为真**；
- 节点为假表示**区块涉及的单元格里，没有一个单元格能放该数字，即每一个候选数全部为假**。

那么此时区块为假的逻辑就可以套用到最开始的那一则示例里理解了： $r89c7(9)$ 区块为假，指的是这两个候选数全部为假。这完全符合我们之前推理的要求。

所以，区块节点的基本逻辑就只有上述两种情况了。

4-3 区块不连续环 (Grouped Discontinuous Nice Loop)

下面我们来看一则区块节点放入不连续环的逻辑。

9	4	3	4 5	7	5	4 5	8	6	1 2	1 2 3
3	6	4	3	7	7	2	2	5	2	2 3
7			8	1	4	6	6	9	7	9
1	2	5 6	7	9	5 6	9	3	7	8	4
5	4	6	4	6	2	1	6	6	1	3
2 3		3	2	3	7	9	7	9	7 8	3
7 8	6	6	7	9	7 8	9	7	9	7 8	4 5
2 3	6	1	2	6	5	6	4	2	2	9
7 8	7 8	7	6	7	7 8	7	7	7 8	7	6
2	6	7	2	6	4	1 2	1 2	3	5	1 2
8			6	9		8 9	9			6
2	6		1	3	7	7	5	4	2	2
8	8 9	8 9	7						6	6
4	5	3	6	1 2	1 2	1 2	1 2	1 2	1 2	7
				8 9	9	9	8 9	9	9	

如图所示，链的表述如下：

$r45c6=r2c6(7)-r1c4(7=5)-r6c4=r6c5(5) \Rightarrow r6c5 \lt \gt 7$

区块为假，指的是无法使得结构构成区块。无法使得区块结构成立，即 $r45c6$ 同时都没有候选数 7。这个时候，由于都没有 7 的缘故，在 $c6$ 之中，7 只能填入到 $r2c6$ 之中。所以此时 $r45c6(7)$ 区块不成立，即 $r45c6(7)$ 区块为假时， $r2c6(7)$ 为真。这样一来， $r45c6(7)$ 就和 $r2c6(7)$ 构成了特别的强关系。于是接着向下推导。

那么，这种结构怎么删数呢？我们可以知道的是，AIC 的逻辑，导致的是链头和链尾至少一个为真，这里应该指的是 $r45c6(9)$ 这个区块和 $r6c5(5)$ 至少一个为真。既不同数，要找的又是一个区块和一个候选数的交集，这怎么找？

区块内有且仅有一处填数位置是为真的，也就是说， $r45c6$ 只有一格是 7，但是我们无法确定哪个是 7。那么链头的确切位置就无法确定，到底是 $r4c6$ 还是 $r5c6$ ；但是链尾处，是唯一确定的候选数 $r6c5(5)$ 。当区块成立时， $b5$ 内一定不可以填 7 了，这包括了 $r6c5(7)$ ；而 $r6c5(5)$ 为真时， $r6c5$ 当然也就不可以填 7。所以 $r6c5 \lt \gt 7$ 。

这个结构是不连续环的拓展版本，因为带了一个区块，所以这个结构也被直接称为**区块不连续环 (Grouped Discontinuous Nice Loop)**。

特别需要引起注意的是，区块节点如果动用首尾异数链的逻辑时，区块并不是所涉及的单元格，每个单元格都能删除候选数，例如此例，删数并不会产生在区块节点上。

4-4 节点重叠 (Node Overlapping)

4-4-1 空矩形 (Empty Rectangle)

我们试想一下, 如果一条链带若干个区块, 而且区块之间有重叠的单元格被共同使用了, 这将如何理解?

8	4 5 6	3	4 5 6	5 6	6	1 2	1 2	1
2	5 6	2	5 6	1	3	5	9	8
1	4 5	4	4 5	2	8	5	6	3
3	4	2	1	5 6	5 6	1 2	1 2	1
4	9	1	5	6	3	2	4	7
2	8	6	1	7	4	1 2	3	5
4 5 6	4 5 6	4	2	6	7	3	1	6
7	2	8	3	6	1	5	4	
6	3	1	6	4	5	6	7	2

如图所示, 如果 $r8c5(9)$ 为假, 则 $r8c7(9)$ 真, 于是将 $r123c7(9)$ 看作区块, 这个区块节点为假, 接着得到 $b3$ 里还剩下 $r1c89(9)$, 它们组成区块结构, 于是可以保证两处有一处为真, 故组成的 $r1c89(9)$ 区块节点可以表示为节点为真。最终该结构的删数就是链头 $r8c5(9)$ 和链尾 $r1c89(9)$ 的交集。显然, 这个区块和候选数的交集只有 $r1c5(9)$, 所以它便是删数。

这个技巧叫做**空矩形 (Empty Rectangle)**, 虽然和“矩形”沾边, 但跟唯一矩形、拓展矩形甚至是可规避矩形里的“矩形”没有任何关系, 它仅仅指的是一个结构, 即宫内 L 形状的五個单元格, 即此处的 $\{r1c789, r23c7\}$ 五个单元格。另外, 空矩形一词还可以用来指代这个链技巧。不过……

4-4-2 空矩形的疑问

可以看到, 实际上结构还是很简单和清晰的, 不过很典型, 因为这个例子里连续用到了区块和区块之间的强弱关系。

那么, 对于空矩形还有一个疑问。如果我把空矩形技巧画成这样下图, 请问这个链是否能正常使用, 并得到一样的删数呢?

8	4 5 6 7 9	3	4 5 6 7 9	5 6 9	6 9	1 2 5 7	1 2 9	1 9
2 5 6 9	5 6 7 9	2 7 9	5 6 7 9	1 7 9	3 7 9	5 7 9	4 7 9	8 7 9
1	4 5 7 9	4 7 9	4 5 7 9	2 7 9	8 7 9	5 7 9	6 7 9	3 7 9
3	4 7 9	4 7 9	2 7 9	1 8 9	5 6 8 9	6 8 9	1 2 6 8 9	1 6 9
4 9	1 9	5 9	6 8 9	3 8 9	2 8 9	4 8 9	1 2 8 9	7 8 9
2 9	8 9	6 9	1 9	7 9	4 9	1 2 9	3 9	5 9
4 5 6 9	4 5 6 9	4 9	2 8 9	6 8 9	7 8 9	3 8 9	1 8 9	1 6 9
7	2	8	3	6 9	1	6 9	5	4
6 9	3	1	6 8 9	4	5	6 8 9	7	2

为了表达清楚，这里也写出文本形式：

`r8c5=r8c7-r123c7=r1c789(9) =?> r1c5 <> 9`

实际上是可行的。虽说这里重叠了一点位置，不过现在要详细阐述此点逻辑。

首先，我们通过之前一样的推理可以到 `r123c7(9)` 处。此时得到的结论是节点为假，即 `r123c7` 没有一个单元格填 9。接着，我们写了 `r1c789(9)`，虽然其中 `r1c7` 是两个区块节点共用的部分，而且此时 `r1c7(9)` 已经为假了。但是这并不妨碍 `r1c789(9)` 这个节点的其它两个单元格为真。因为之前说过，区块为真表示区块结构成立，所谓的成立就指的是区块里有一个单元格填这个数就可以了。显然，`b3` 里现在只剩下 `r1c89` 可以放下 9 了，也就直接意味着 `r1c89` 必然有一个 9 的出现，所以总的来说，`r1c789` 里确实有一个 9 了，所以区块是成立的，也就意味着该区块结构为真。

正因为如此，空矩形才得以存在。实际上，在一些资料和文献之中（包括发现人对于空矩形的介绍）都是采用的重叠的形式介绍的。

4-4-3 节点重叠（Node Overlapping）

接下来我们就来介绍一个节点重叠的普通版示例。

9	2	1	6	4	8	4	5	1	3	7
3	4	1	9	4	5	7	4	2	2	6
1	1	1	1	3	2	3	1	2	3	1
4	4	6	5	6	5	6	4	5	6	8
7	7	7	7	7	7	7	7	7	7	9
1	1	4	3	2	3	2	3	1	2	1
7	7	8	9	8	5	6	5	6	6	9
6	1	1	7	2	1	2	1	2	2	9
8	8	9	8	4	4	4	3	4	8	9
2	5	3	4	4	1	2	1	4	1	5
4	3	2	5	1	9	4	6	1	1	4
7	7	7	7	7	7	7	7	7	7	9
5	1	1	2	4	7	8	1	1	3	1
4	4	6	7	6	7	4	9	7	9	9
8	1	9	6	4	3	4	3	1	5	2
4	4	7	4	7	4	7	4	7	4	7

如图所示，链的表达如下。

$r13c4=r6c4-r5c56=r5c8-r1c8=r1c46(4) \Rightarrow r2c5, r3c6 \neq 4$

可以从示例里得到，链头和链尾都是区块节点，但区块有所重叠：共用了 $r1c4(4)$ 。那么，删数怎么看呢？实际上，删数就是这两个区块的交集，而这两个区块的交集正是 $b2$ 的其余单元格，所以 $r2c5$ 和 $r3c6$ 自然不能够填入 4。

4-5 鱼图练习

下面来看看这个鱼图测试题，和之前一样，“/”表示不能放候选数 a 的地方。这个测试题，你能找到合适的删数 a 的确切位置吗？

	/	/	/	/				
	/	/	/	/	/	/	/	
		/		/	/	/	/	
				/	/	/	/	
/		/	/	/	/	/	/	
/		/	/		/	/		

Part 5 待定数组 (ALS)

在之前，其实我们已经提到了 ALS（即待定数组）的一些基本的使用方式和手段，但实际上待定数组还有很多话题要说，现在我们就来看看，ALS 在链的章节里面会产生哪些有趣的内容和知识。

5-1 强 ALS (Strong ALS)

5-1-1 同区域异数强关系引入

1		2		3	6	2 3	1 2	5	2 3	4
8 9	8		9			8	7		7 8	
4	7		3	6	5	2 3		9	2 3	6
1		2		6	1	3			2 3	9
8	8				8		7		7 8	
7	1	8	2	6	3	9	4	5		
3	4 6	4 6	9	7	5	8	1 2	1 2		
5	9	2	4	1	8	6		3	3	
6	3	1	7	5	4	2	9	8		
	5	4 7 9	1 3	2 3	1 2	3	1 3			6
8 9			8	8 9		4 7	7			
2	4 8	4 9	1 3	3		3		5	1 3	

如图所示，可以看到此时我们把 $r1c2(2)$ 和 $r3c1(1)$ 用强关系连起来，而它们是不同的数值。这是为什么呢？现在我们就来学习一下。

观察紫色的两个单元格 $r1c2$ 和 $r3c1$ ，可以发现此时两个单元格只含有 1、2、8 三种不同的数字。如果此时假设 $r1c2(2)$ 为假的时候，这两个单元格里就没有 2 了，如果再让 $r3c1(1)$ 也一并消失的话，两个单元格就都只剩下唯一的一种候选数 8，使得出错。所以 $r1c2(2)$ 为假时，需要 $r3c1(1)$ 为真才行。

那么，可以发现 $r3c1(1)$ 为真后，显然 $r3c4(1)$ 是为假的，所以形成弱关系；继续推导的话，依然套用刚才的逻辑：观察到 $\{r12c5, r3c4\}$ 三个单元格包含 1、2、3、8 四种数字，而此时的 $r3c4(1)$ 为假，也就意味着这三个单元格里不含有 1。如果此时，这三个单元格的所有 2 也都消失的话，这三个单元格就同时少了两种数字，便只剩下 3 和 8，但这是三个单元格，填入两个数字是显然不够的，所以也会矛盾。所以我们就得到了 $r3c4(1)=r12c5(2)$ 的结论。

于是，这条链成立了：

$$r1c2(2)=r3c1-r3c4(1)=r12c5(2) \Rightarrow r1c6 \neq 2$$

这个例子就分析完毕了。其中可以发现，{r1c2, r3c1}两个单元格包含三种不同的数字，因为最终填数是待定的，所以这两个单元格组成了一个待定数组结构，即一个 ALS 区域；同理，{r12c5, r3c4}三个单元格包含了四种不同的数字，所以填数也是不确定的，因此我们也称为一个 ALS 区域。

可以从例子里看到，实际上，我们利用的 ALS 区域的逻辑是：**n 个单元格里包含(n+1)种不同的候选数，如果少掉其中的两种，就会变为(n-1)种候选数，使得无法填满 n 个单元格，进而出错的结果。**这便是 ALS 的核心。

另外，这个结构的头尾各使用了一个 ALS，而中间用了弱关系连起来，是一个长度为 3 的链。我们把长度为 3，且内部的两个强关系都产生自 ALS 的链称为 **ALS-XZ 法则**（或 **ALS-双强链法则，ALS-XZ Rule**）。X 和 Z 都表示涉及的数值，其中 X 是弱关系涉及的数，而 Z 是链头和链尾两端涉及的数。

实际上，ALS 分两种，强 ALS 和弱 ALS。为了和后续的内容作出区分，我们强制称这个结构为**强 ALS**（**Strong ALS**，但一般都简称为 **ALS**，而它一般不简称为 ~~SALS~~），因为在后面的内容里，ALS 内存在一种与之相反的**弱 ALS**（**Weak ALS**，简称 **WALS**），这种 ALS 只产生弱关系。一般来说，只要我们不强调它和弱 ALS 的逻辑和定义不同，我们都直接简称之为 ALS。

另外，你可能会有一种感觉，之前的假设推理都是“顺序确切的”，也就是推导到这里的时候，一定能确定是某个节点为真（因为是共轭对的关系，前面假后面就必须为真），而此时我们感觉这个结构里两个数之间的这种关系还差一点。比如这两个单元格里同样包含数字 8。如果此时我们假设 r1c2 和 r3c1 两个 8 全消失的话，两个单元格也只剩下数字 1 可填，而且其中 r1c2 还没有候选数了，也照样会出错。那岂不是我还可以得到 $r1c2(2)=\{r1c2, r3c1\}(8)$ 的结论？准确的答案是，是的。强弱关系并不需要确切的顺序得到，如果像是上面这种逻辑，我们照样可以得到一样的结果，因为在强弱关系的叙述文字里，我们并未提到在推导过程里，节点之间必须是明确的顺次推导关系。比如这里，我们完全可以走候选数 8 的方向；当然，这个 8 是两个不同行列的单元格，用起来不方便而已，但客观来说，强关系确实是形成了。

那么，你真的了解 ALS 这个缩写吗？我相信大多数小伙伴在读到这里都是把这个词汇死记硬背记住的。ALS 实际上是 Almost Locked Set 的缩写，而 Locked Set，实际上是数组在早期的英文称呼，现在数组被广泛采用“子集”一词（Subset），所以这个说法应当为 Almost Subset。而为了保持兼容性（保留原始说法以照顾以前的习惯），故这个说法继续采用了 Locked Set 这个古老的说法；当然，你也可以采用 Almost Subset 这种说法。不过，可以看到它们的缩写一个是 ALS，一个则是 AS，差距并不大，而表示同一个东西难免会让人觉得不好理解，所以注意区分和辨别。

5-1-2 ALS-XZ 和伪数组

我们再来看一则示例。

	1			1	3	3	5	
6	9	7	8	2				4
				4	6	4	6	9
6	9	7	8	4	6	4	6	9
5		3	8	4	6	4	6	9
3	9	8	6	5	7	4	1	2
2	5	7	4	1	8	6	9	3
4	6	1	2	3	9	2	3	7
1	3	9	2	4	6	4	6	9
8	2	6	1	3	5	9	4	7
7	4	5	2	6	2	3	2	3

如图所示，首先我们观察到，**r2c169** 三个单元格包含 3、6、8、9 四种数字，现在以 **r2c6(3)** 为假最为链的起点。假设它为假后，对于 **r2c9(8)** 而言，它必须为真，否则 3 和 8 没有了之后，三个单元格里就只剩下了两种数字，填不满导致出错。所以形成了强关系；然后用弱关系连接上 **r1c7(8)** 后，因为 **r1c7** 是双值格的关系，显然 **r1c7(3=8)** 成立，所以整体的链就成立了，删数则是 **r2c6(8)** 和 **r1c7(8)** 的交集。

可以看到，这一则示例如果我们不使用链的视角，把链的线条去掉，并把 8 的涂色也去掉之后，**{r1c7, r2c169}** 四个单元格将可以构成伪数组结构：四个单元格里，除了数字 3 以外，其余的数字都只出现在一个区域里，即它们不可能有重复的填数情况，唯独只有这里的 3。而根据填数情况分析可以发现，3 必须至少有一个要出现，否则填不满四个单元格。所以删除掉数字 3 的交集。

所以，伪数组可以转为 ALS-双强链结构，并且转换方式是，把其中单独跨区的那一个单元格分开单独作为一个 ALS；而剩下的单元格就必定在同一区域了，所以它们作为一个区域，于是我们连上强弱关系就可以形成 ALS-XZ 结构了。

实际上，ALS 最小可以只涉及一个单元格（只要这个单元格是双值格，我们就可以使用 ALS，因为从定义来看，它确实也符合条件，并且可以运用强关系：如果两数同假，则“1 个单元格只包含 0 个候选数”，这一点也都是符合刚才所说的推理过程的；而最多只能到 8 个单元格（因为此时最多包含了 9 种候选数，也就是最大情况），不过实际上，这种结构很少被使用，一般这种结构都得不到什么合适的结论。

5-1-3 孪生 ALS-XZ (Siamese ALS-XZ Rule)

5	2	1	7	3	8		6	4	6	4	9	5	2	1	7	3	8		6	4	6	4	9
6	4	9		1		8	3	7				6	4	9		1		8	3	7			
7	8	3	6	4	9	1	5	2				7	8	3	6	4	9	1	5	2			
4	7			9								4	7			9							
1	9			6								1	9			6							
2	3	6		8	7	5	9					2	3	6		8	7	5	9				
9	1	7	8	5	4	3	2	6				9	1	7	8	5	4	3	2	6			
				2	6											2	6						
						4		5										4		5			

如图所示，我们来看这两则示例，这两则示例实际上是同一个题，而且结构大部分内容都是一样的。

我们先来理解第一个示例，写出它的文本表示形式。

$r89c8(1)=r8c8-r5c8(7)=r46c9(1) \Rightarrow r4c8, r8c9 \neq 1$

在第一个示例里，是以 $r89c8(1)$ 作为区块节点起头，假设为假的，显然， $r89c8$ 两个单元格里只有 1、7、8 三种候选数，而如果 $r89c8(1)$ 区块节点为假，而且 $r8c8(7)$ 也为假的话，则这两个单元格就只剩下 8 这一种数字，显然矛盾了；所以说 $r89c8(1)$ 为假的时候，对于 $r8c8(1)$ 来说就必须为真，所以它们形成强关系；然后， $r5c8(7)$ 和 $r46c9(1)$ 的逻辑同理。

而第二个例子：

$r89c8(8)=r8c8-r5c8(7)=\{r4c9, r5c89\}(8) \Rightarrow r4c8 \neq 8$

可以从例子看到，这里涉及了一个“拐弯的区块”。我们尝试把它看作和之前 BUG 区块类型的那种“广义的区块”一样的逻辑，把它们视为一起，那么这种结构如果为真或为假两种情况确实和普通的区块节点的逻辑是一模一样的。由于三个单元格同宫，所以视为一个节点后，它为假就只能使得其中没有任意一个单元格填入这个数；反之，如果这个节点为真，则也意味着其中只有一个单元格填入这个数，这个说法和区块节点的思维完全一样。所以我们干脆就把它也称为区块节点。

首先， $r89c8(8)=r8c8(7)$ 是显然的（这一点在上一情况已经讲到过）；而对于 $r5c8(7)=\{r4c9, r5c89\}(8)$ ，也是成立的：现在 $r5c8(7)$ 是为假的，如果继续向下推理，如果 $\{r4c9, r5c89\}(8)$ 这个节点也为假的话，就意味着里面没有一个单元格能放下 8，此时 8 也没有了，而细数 $\{r5c8, r456c9\}$ 四个单元格，里面一共是 1、3、4、7、8 五种数字，而此时由于两个节点同假的假设的关系，四个单元格里同时少了全部的 7 和 8，导致只剩下了 1、3、4 三种数，而它必须放在这四个单元格里，这样显然是不够放下的。所以，这样便出现了矛盾。

可以看到，两个情况下，除了开头和结尾的节点不一致以外，弱关系相连的数字 7 是完

全一样的节点，而切换到不同的头尾，就会产生不同的删数，我们称之为**孪生 ALS-XZ**（**Siamese ALS-XZ**），孪生一词已经在链列（鱼）结构里出现过，正好表达的是“大部分结构相同，而不同的地方能导致不同的删数”之意。而实际上，这种结构和孪生链列一样，我们依然可以合并为一个图，如图所示。请你思考一下这个合并的示例如果整体，应该如何推理。

5	2	1	7	3	8		6	4	6	4		9
6	4	9		1		8	3	7				
7	8	3	6	4	9	1	5	2				
4	7			9								
1	9			6								
2	3	6		8	7	5	9					
9	1	7	8	5	4	3	2	6				
				2	6							

$$r89c8(18)=r8c8-r5c8(7)=\{r5c8, r456c9\}(18)$$

$$\Rightarrow r4c8 <> 16, r8c9 <> 1$$

5-2 链的双向性和强弱关系的新定义

现在我们来学习一种新的链的定义。

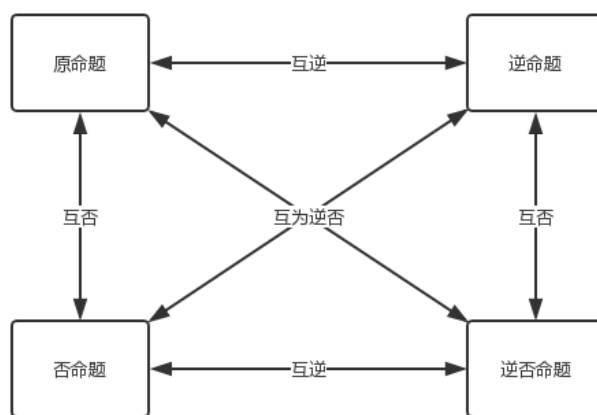
在之前的内容里，我们已经了解到了，链由两种关系构成：强关系和弱关系。它们各自的定义如下。

- **强关系**：如果节点 1 为假时得到节点 2 为真，则称节点 1 和节点 2 有强关系。
- **弱关系**：如果节点 1 为真时得到节点 2 为假，则称节点 1 和节点 2 有弱关系。

不过你可能会想过这样一个问题。为什么节点 1 到节点 2 的顺次逻辑，在结论里就被叙述为“节点 1 和节点 2 之间有强弱关系”了呢？难道不是“节点 1 到节点 2 有强弱关系”吗？

这个问题问得好，我们来看看如何去回答。首先，我们得了解链的新的定义方式。虽说是新的定义方式，但实际上和之前的定义是完全等价的两种不同说法而已。

我们在高中学习过命题和逻辑一章，并了解了原命题、逆命题、否命题和逆否命题，其中逆命题和否命题真值相同（即要么都对，要么都不对）；同理，原命题和逆否命题的真值相同。如图所示。



那么，我们回顾一下强弱关系的定义，拿强关系举例，因为是证明，所以我们把命题的结论从“则称节点 1 和节点 2 有强关系”也替换为“则称为节点 1 到节点 2 有强关系”（改成有方向的）。此时我们取出其中的条件部分。

如果节点 1 为假，则节点 2 为真。

我们如果把它看作原命题，那么它的逆否命题则是把条件和结论都取反后再交换。那么命题就变为了

如果节点 2 为假，则节点 1 为真。

这两种说法明显是互为逆否命题的，所以真值相同，所以这种说法显然也可以替换到原来的强关系的定义里的，即变为

如果节点 2 为假得到节点 1 为真，则称节点 1 到节点 2 有强关系。

显然，这个说法已经把节点 1 和节点 2 互换了方向，但依然可以得到强关系。而这个逆否命题的说法显然对应的应该是

如果节点 2 为假得到节点 1 为真，则称节点 2 到节点 1 有强关系。

所以可以看到，一个条件可以得到两个方向的结论。所以强关系是不受方向的约束的，即正着推和倒着推都是可以得到结论的；同理，弱关系的证明完全是一样的。

所以，强关系和弱关系并不受到方向的约束。而链是完全依赖于强关系和弱关系的，而强关系和弱关系都可以反向推理和理解，所以链是完全可以反向理解的，故链是可以反向来推理和理解的（或者换句话说，链的两个方向推理结果都能得到一致的结论；或者干脆可以直接说链不具有方向性）。

那么，除了能够得到这一点外，我们还能得到什么东西呢？强弱关系的新的定义。

由于强弱关系都可以逆向理解，那么我们同时给出两种写法，看看有什么新的结论：

- 从节点 1 假到节点 2 真。
- 从节点 2 假到节点 1 真。

这分明说的就是两个节点不同假。因为不管两个节点哪个为假，都能推导得到另外一个节点是必须为真的，所以，强关系还可以被叙述为“两个节点不可以同为假”。

同理，我们写出弱关系的两种说法：

-
- 从节点 1 真到节点 2 假。
 - 从节点 2 真到节点 1 假。

可以看到，不论哪个节点为真，都必须得到另外一个节点是为假的。所以，弱关系也可以被改变叙述为“两个节点不可以同为真”。所以，我们就得到了强弱关系的新定义方式：

- 强关系（新定义）：**如果两个节点不同假，则称两个节点有强关系；**
- 弱关系（新定义）：**如果两个节点不同真，则称两个节点有弱关系。**

这个说法有什么用途呢？可以看到，这种说法就可以去验证各式各样的强弱关系了，它并不受到之前共轭对等同在一个区域下的候选数的约束。我们可以利用这一点来得到一些更为奇妙的结果和结论；如果使用之前的逻辑，就不一定能得到，因为从之前的定义来看，对于 ALS 就显得很“无力”，因为 ALS 里的关系的发散的，你可以随便从之前的某个节点往下连接到 ALS 里的任意成立的情况，而这些情况不止一种，即并不是一种确切的情况。所以，这种新的定义模式会更加灵活、高效地把链使用起来。

不过，需要你注意的是，在新的定义里，我们完全可以察觉到一个奇妙的地方。比如强关系，我们仔细看它的定义就可以发现，“节点不同假”包含了一种情况，即“两个节点同真”，因为两个节点一共只包含四种填数可能：**甲真乙真、甲真乙假、甲假乙真、甲假乙假**。而不同假指的是两个节点里不能同时都是假，所以只排除了最后的一种情况，而剩下的三种情况里，是包含“同真”的结果的；同理，因为强弱关系的定义是对称的，所以弱关系还包含“同假”的结果的，所以请你务必重视，不要漏了这个特殊的情况，在某些特殊的时候，这个特殊的条件会对我们做题具有相当大的帮助，例如后面会介绍到的**毛刺（Burr Rule）**和**毛边（Bi-burr Rule）**。

5-3 ALS 的互补视角

接下来，我们来试着理解一下 ALS 的互补视角。

8	3	7	2	9	¹ _{5 6}	¹ 3 ₁	5	4
3 ₆	4	2	¹ _{5 6}	7	¹ _{5 6}	9	8	¹ 3
5	1	9	4	8	3	6	2	7
9	8	3	7	5	2	¹ ₄	¹ _{4 6}	¹ ₆
2	5	6	9	1	4	³ ₈	7	3 ₈
1	7	4	3	6	8	5	9	2
⁴ ₆	² ₇	9	1	8	3	⁵ ₆	⁴ _{5 6}	⁵ ₆
⁴ ₆	³ ₇	3 ₆	5	¹ ₆	2	¹ ₇	¹ _{4 6}	1 _{8 9}
⁷ ₆	² ₆	8	¹ _{5 6}	4	9	¹ ₂	3	¹ _{5 6}

如图所示，我们先把链的文本形式写出来：

$r1c2(3)=r7c2-r7c9(9)=r2c9(3) \Rightarrow r1c7, r2c1 \langle \rangle 3$

可以从例子里发现，图中存在两个 ALS 区域，它实际上就是一个 ALS-XZ，不过从例子里看到，它并未画出之前那样合适的 ALS 区域；相反地，它却画出了原来 ALS 区域所涉及不到的单元格。这是怎么回事？

我们这么思考这个问题。首先我们来分析 $c2$ 的异数强关系 $r1c2(3)=r7c2(9)$ 。我们利用新的强关系定义，如果它们同假，则 $c2$ 放下 3 和 9 的位置只有 $r8c2$ 一处，但 $r8c2$ 显然不能同时放下 3 和 8，所以出现矛盾，所以两个节点不同假，即形成了强关系。这一点是比较好理解的。

同理，我们再来看 $c9$ 的异数强关系 $r7c9(9)=r2c9(3)$ 。如果两个节点同为假，则会发现 3、8、9 在 $c9$ 只能放在 $r58c9$ 两处。显然， $r58c9$ 只有两个单元格，而 3、8、9 是三种数字，显然是放不到两个单元格的，所以这样必然会产生矛盾。因此，强关系也是成立的。

这便是 ALS 的互补观察视角，即通过互补的角度作为占位来处理 and 推导矛盾：选取和原 ALS 涉及单元格互补的单元格，再把 ALS 没有涉及的数字全部圈出来，即构成互补结构。

5-4 链式 ALS 结构拓展

接下来我们来看一下，套用链的 ALS 结构都有哪些拓展构型。

5-4-1 带双 RCC 的 ALS-双强链法则 (Doubly Linked ALS-XZ Rule)

如果我们之前学到的 ALS-XZ 并不够新奇的话，我们现在就来接触一种新的 ALS-XZ，而这种结构更为特殊。

在此之前，我们先来介绍一个术语：**严格共享候选数 (Restricted Common Candidate, 简称 RCC)**，这个词在一些软件和分析 ALS 技巧里经常使用。

我们先来说说 RCC 是什么。RCC 实际上指的是之前 ALS-XZ 里的那个 X，即弱关系连接的候选数。也就是说，一般来说 ALS-XZ 只有一个 RCC (因为 ALS-双强链法则规定了链只有一个弱关系，故只可能有一个 RCC)，例如第一个示例里的 1 和第二个示例里的 8，就是 RCC。

那么，ALS-双强链法则是否可能产生两个 RCC 呢？答案是可以，不过因为从结构上根本不可能实现两个弱关系，那么怎么形成呢？我们来看下面这则示例。

2	1 2	1	2	2	2	4 5 6	3	5 6
4 5	5	4 5	4	6 4	4	8 9	7	5 6
7 9	7 9	7 9	7 9	7 8 9	8 9	7 9	7	7 9
6	2	4	5	5	2 3	2 3	1	8
	7 9	7 9	7 9	4	4	4	7 9	7 9
4 5	8	3	4	6	1	4	4 5 6	2
7 9	7 9		4	7 9	7 9	7 9	7 9	7 9
1 2	1 2	1	9	5	7	3	4 6	1
4	4	4						6
1 3		1	1	3	1 3	5	9	2
4 5	6	5	4	4	4	7 8		
7 8	7 8	7	7	8	8	8		
1 3	1 3	1 3	2	2 3	1 2 3	5	4	1
4 5	5	4 5	8	8	6 4 6	7 8	7	5
7 8 9	7 9	7 9			8	7 8		7 8
	3	3	6	1	1	2	5	4
7	7			8	8 9	8 9		
5	4	8	7	2	2	6	1	3
	9			6	5 6			
1	5	1	2	3	4 6	4 5 6	6	6
5	9	9					7 8 9	7 8 9

如图所示，我们把 r2c239 看作一组 ALS (三个单元格，有 2、4、7、9 四种候选数)，r4c23 看作一组 ALS (两个单元格，有 1、2、4 三种候选数)。这个链的写法如下：

$r2c2(2)=r2c3(4)-r4c3(4)=r4c2(2)$

或

$r2c3(4)=r2c2(2)-r4c2(2)=r4c3(4)$

这个结构可以写成两个不同的链结构。也就是说，这个结构自带了 2 和 4 两个 RCC。根据原定的 ALS-XZ 逻辑，那么可以删除掉的是 c2 内其余单元格的候选数 2，以及 c3 内其余单元格的候选数 4。那这题为啥标注了那么多红色的删数？

回想一下 RCC 的特征：RCC 呈弱关系，意味着两数不可同真。不可同真本应该指的是

“同假”或“有且仅有一个为真”，但是当我们删除掉 c2 内其余单元格的候选数 2 和 c3 的其余单元格的候选数 4 之后，RCC 就不可能再同假了。因为删掉这些数之后，c2 其余位置就不再存在填入 2 的地方，c3 的其余位置也就不存在填入 4 的地方，此时 RCC 变为共轭对。共轭对即一真一假，比如图中 $r24c2(2)$ 这个共轭对，只能是一真一假， $r24c3(4)$ 也是一样的情况。首先，我们根据上面描述出来的两条链，可以看到两个强关系：

- $r2c2(2)=r2c3(4)$
- $r4c2(2)=r4c3(4)$

这意味着 $r2c2(2)$ 和 $r2c3(4)$ 不可同假，同时 $r4c2(2)$ 和 $r4c3(4)$ 也不可同假，而且 $r24c2(2)$ 和 $r24c3(4)$ 又都只能是一真一假。这样的话， $r24c2(2)$ 和 $r24c3(4)$ 这四个候选数的填数情况只可能是以下两种可能了：

- $r2c2(2)$ 和 $r4c3(4)$ 同时为真；
- $r2c3(4)$ 和 $r4c2(2)$ 同时为真。

别无其它可能。这样一来，观察 $r2c239$ 这组 ALS，里面的 $r2c2$ 和 $r2c3$ 其中有且仅有一格被 RCC 数字占据。那么，ALS 内其余两格，则会构成 79 数对。同理， $r4c23$ 这组 ALS，里面的 $r4c2$ 和 $r4c3$ 其中有且仅有一格被 RCC 数字占据，剩下一格就一定是数字 1。也就是说，结构 $r2c239$ （靠上方）这组 ALS 一定会产生 79 数对， $r4c23$ （靠下方）这组 ALS 一定会产生数字 1，所以 $r2$ 的其余位置都不应该填入数字 7 和 9， $r4$ 和 $b4$ 内的其余位置也都不应该填入数字 1，因此删掉它们。

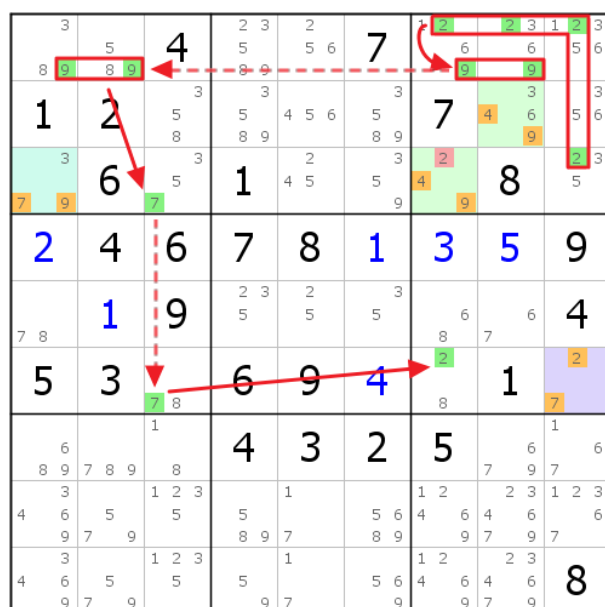
这个结构分析起来非常复杂，因为带有了两个 RCC，并且组成了特别的结构。

它的名字是什么呢？**带有双 RCC 的 ALS-XZ (Doubly Linked ALS-XZ)**；而对比这种说法，最开始的结构因为只有一个 RCC，所以可以被称为**带有单 RCC 的 ALS-XZ (Singly Linked ALS-XZ)**。

5-4-2 ALS-XY-Wing

5-4-2-1 普通的 ALS-XY-Wing 示例

在之前的 Wing 结构一节里,我们介绍了一些 Wing 的链式写法,其中提到了 XY-Wing,那么来看看 ALS 可否和 XY-Wing 交织形成新的东西。



如图所示,链的写法如下:

$$\{r1c789, r3c9\}(2)=r1c78-r1c12(9)=r3c3-r6c3(7)=r6c7(2) \Rightarrow r3c7 \lt \> 2$$

在例子里,为了简化涂色逻辑和思维,而且我们也讲到了互补的视角,所以图上全部都是 ALS 的互补视角的绘制逻辑。

如果 $\{r1c789, r3c9\}(2)$ 和 $r1c78(9)$ 同假的话,则可以发现 b3 里能放下 2、4、9 的位置只有两个单元格,出现矛盾;同理,如果 $r1c12(9)$ 和 $r3c3(7)$ 同假的话,则 b1 里放 7 和 9 的位置只有一个单元格可填,出现矛盾;如果 $r6c3(7)$ 和 $r6c7(2)$ 同假,则 r6 里只有一个单元格能放 2 和 7,也是放不下的,出现矛盾。

所以,所有强关系都成立。不过,为什么它跟 XY-Wing 沾边呢?从这个链涉及的候选数的顺序来看,是 $2=9-9=7-7=2$,和 XY-Wing 的链的写法完全是一样的,所以这个结构就叫做 ALS-XY-Wing。

5-4-2-2 ALS 区域有重叠的 ALS-XY-Wing

我们再来看一则示例。

9	¹ ₅	³ ₅	² ₄	³ ₄	7	¹ ₂	6	8
7	2	⁶ ₈	¹ ₂	¹ ₅	¹ ₅	9	4	3
¹ ₆	¹ ₈	³ ₄	² ₃	² ₃	¹ ₂	² ₇	⁵ ₉	
2	7	¹ ₅	¹ ₅	¹ ₃	¹ ₃	4	8	9
¹ ₄	6	9	7	8	¹ ₄	³ ₅	³ ₅	2
8	⁴ ₅	3	9	⁴ ₅	² ₄	7	1	6
¹ ₄	⁶ ₈	² ₈	¹ ₂	³ ₄	5	³ ₈	9	7
3	⁴ ₈	7	6	² ₄	² ₈	⁵ ₈	² ₅	1
5	¹ ₈	² ₆	¹ ₂	³ ₇	¹ ₂	³ ₆	² ₃	4

如图所示，这个链看起来比较混乱，因为有一部分 ALS 和 ALS 是重叠的。我们先写出文本表达：

$r2c36(8)=r3c6-r5c6(1)=r6c5-r2c5(5)=r3c46(8) \Rightarrow r2c4 \neq 8$

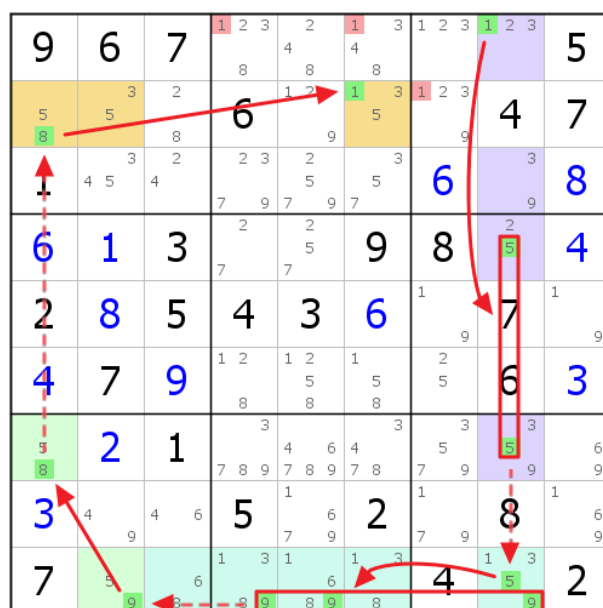
这个示例一共是涉及了三个 ALS 区域，而其中最容易理解和明白的 ALS 区域是{r5c6, r6c56}(1245)和 r2c356(1568)。实际上，b2 里有一个 ALS 区域是{r1c45, r2c56, r3c456}(12345689)。这个区域稍微大了点，涉及了七个单元格，并包含了 8 种数字，但也确实满足 ALS 的要求，所以也是合格的 ALS。

别看数很多，逻辑还是很清晰的。如果 r2c36(8)为假，则 r3c6(1)必须为真，否则紫色的 ALS 区域里将少掉两种数字 1 和 8，使得四个单元格仅填两种数字，显然是不够填的，所以产生矛盾；而第二个 ALS 区域就不用过多分析了；第三个 ALS 因为涉及了 7 个单元格，所以显得很复杂。从之前的逻辑，我们推理到了 r2c5(5)为假，则 r3c46(8)此时必须为真，否则的话，在这个区域里其余位置也没有其它的 5 和 8，所以会少两种数字，7 个单元格就只能填入 6 种数字，导致填数不够的矛盾。所以强关系是成立的。

此时，这个例子特殊的地方仅仅是，ALS 区域有重叠，导致了理解起来的不便；不过针对于这种例子，我们可以尝试使用互补视角来观察：如果 5 和 8 都去掉后，b2 里只有 r2c4 能放下 5 和 8，而显然是不能同时放下两个数的，所以矛盾。

5-4-3 ALS-双值格链 (ALS-XY-Chain)

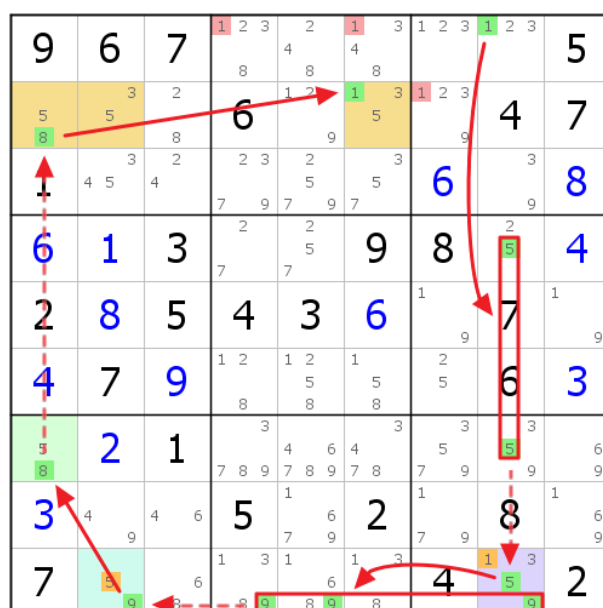
接下来我们再看一则更长的 ALS 链式结构。



如图所示。

$$\begin{aligned} r1c8(1)=r47c8-r9c8(5)=r9c458-r9c2(9)=r7c1-r2c1(8)=r2c6(1) \\ \Rightarrow r1c46, r2c7 <> 1 \end{aligned}$$

这个链内一共有四个 ALS 区域，它们分别是 r1347c8、r9c34568、{r7c1, r9c2} 和 r2c126。分析方式和之前的 ALS-XY-Wing 等结构相同，这里就不再重复。强关系依然采用“不可同时为假”进行分析。不过，如果你觉得这种结构过大的话，可以试试互补视角，不过互补视角后，部分位置就会产生重叠。如图所示。



5-4-4 ALS-W-Wing

接下来再来看一则示例。

6	9	¹ 5 ³	8	¹ 5 ⁷ 5	2	4	³ 5 ⁷
7	¹ 2	4	¹ 2 ⁵ 9	3	² 5 ⁹	8	¹ 5 ⁹ 6
¹ 5 ⁸	¹ 4 ⁸	¹ 5 ⁸	3	7	² 5	6	² 1 ⁵ 9
¹ 4 ⁸	¹ 4 ⁸	¹ 5 ⁸	6	¹ 2 ⁵	4	3	² 1 ⁵
3	6	2	¹ 5	9	8	¹ 5	¹ 4 ⁵ 7
¹ 2 ⁴	¹ 3 ¹	¹ 3 ¹	⁴ 5 ⁷ 9	⁴ 5 ⁷ 9	⁴ 5 ⁹	6	⁴ 5
9	¹ 4	6	⁴ 5	8	⁵ 3	7	¹ 5
² 4	5	7	² 4 ⁹	6	1	⁴ 9	3

如图所示，链如下所示：

$r7c23(8=1)-r7c7=r3c7-r3c1(1)=r35c1(8) \Rightarrow r7c1 \neq 8$

这个链内一共有两个 ALS，并且位于首尾。所以利用刚才的结论，我们可以知道， $r7c23(8)$ 和 $r7c23(1)$ 是强关系。同时为假的时候， $r7c23$ 内只有 3 这一种候选数可填，这显然不成立。结尾的强关系也是一样。

这个结构的写法非常类似于 W-Wing 的结构： $(w=x)-x=x-(x=w)$ ，所以被称为 ALS 版本的 W-Wing，即 **ALS-W-Wing**。

5-4-5 死亡绽放 (Death Blossom)

接下来我们来说一个和 Wing 里分类讨论模式非常相似的 ALS 技巧：**死亡绽放 (Death Blossom)**。

5-4-5-1 三花瓣死亡绽放 (Death Blossom With 3 Petals)

1 2 5 7 9	1 5 9	3	8	4 7	2 4 5	6	7 9 7 9	2
4	5 6 8	2 5 6	2 5 7	2 6	9	2 3 7 8	3	1
1 2 7 8 9	1 6 8 9	1 2 6	1 3 2 7	2 6	1 2 3 6	4	5	2 7 8 9
1 5 9	1 4 5 6 9	1 4 5 6	4 6	3	8	7	2	4 5 9
2 3 5 9	3 4 5 6 9	2 4 5 6	2 6 7	1	4 7	5 9	3 4 8 9 8 9	4 5 8 9
2 3	7	8	9	5	4	2 1 3	1 4 6	3 4 6
1 3 5 8	2	7	4 5	3 8 9	4 5 3	1	5 9	6 4 5 6 9
6	4 5 8	4 5	1	8 9 7	2 4 5	2 5 9	4 7 9	3
1 3 5	1 4 5	3 9	4 5 7	3 4 7	2 4	6	8	1 4 7

如图所示，图中一共使用了两个 ALS 区域：r23c45(23567)和 r239c5(2467)。

我们按 r7c4 作为拐点进行分情况讨论。

- 假设 $r7c4 = 3$ ，则由于 $r3c4 \neq 3$ 的关系，如果此时橙色 ALS 里如果没有 2，则会少两种数字，便出现错误，所以 $\{r2c45, r3c5\}(2)$ 为真；
- 假设 $r7c4 = 4$ ，则 $r9c5 \neq 4$ ，如果此时 $r239c5(2)$ 全部消失，则红色 ALS 里同时少了两种数字，导致不够填的矛盾；
- 如果 $r7c4 = 5$ ，和第一个情况类似，得到橙色 ALS 里无法填数。

所以，由于三种情况都可以得到 2 为真的结果，而 r7c4 只有三种情况，所以 $\{r2c4, r239c5\}(2)$ 里必须至少有一个数字 2 为真，所以删除掉这 4 个 2 的交集。

在死亡绽放里，我们把类似于 Wing 类分支情况的技巧里的拐点或折点称为**花瓣单元格 (Petal Cell)**，这个单元格有几个候选数就称有几个花瓣。比如该示例里有三个候选数，即三个花瓣。

5-4-5-2 四花瓣死亡绽放 (Death Blossom With 4 Petals)

2 3 9	2 4 9	2 4 5	2 3 4	7	1	6	8	3 4 5
8	7	1 2 4 5 6	2 3 4 6	2 3 5	9	2 3 5	1 2 3 5	1 3 4 5
2 3 4	1 2 6	1 2 4 5 6	8	2 3 5	4 5 6	9	1 2 5	7
5	1 4 6	7	9	1 4	2	8	1 3 6	1 3 6
2 4	3	1 2 4	5	6	8	1 4	7	9
4 6	8	9	7	1 4	3	1 4 5	1 5 6	2
1	2 4 6	8	2 3 4 6	2 3 5	7	2 3 5	9	3 5 6
7 9	2 6 9	2 6	2 3 6	8	5 6	1 2 3 7	4	1 5 6
4 7	5	3	1	9	4 6	2	2 6	8

如图所示，我们按 **r9c1** 进行分情况讨论。

- 假设 **r9c1** = 2 时，**r8c23** \neq 2，如果 **r8c6** \neq 5 则在橙色 ALS 里少了两种候选数，导致填数不够，所以 **r8c6** = 5；
- 假设 **r9c1** = 4，则红色 ALS，可以得到 **r8c6** = 5；
- 假设 **r9c1** = 6，则绿色 ALS 里必须让 **r6c7** = 5，否则绿色 ALS 里少两种数字，不够填数，出现矛盾；
- 假设 **r9c1** = 7，则橙色区域和情况 1 一样，**r8c6** = 5。

所以，不管是什么情况，最终必须 **r6c7** 或 **r8c6** 里至少有一个单元格是 5，所以删除其交集，即 **r8c7(5)**。

这个示例带有四个花瓣。一般来说，三个花瓣的示例找起来就比较困难了，所以一般四个花瓣的很少发现，甚至发现不了，所以仅供参考和学习。

5-4-5-3 五花瓣死亡绽放 (Death Blossom With 5 Petals)

1	3	8	1	3	1	5	2	3	1	2	3
7	5		1	3	1	2	4	3	2	3	6
1	9	2	4	3	6	7	8	5	1	8	
1	3	1	3	2	6	4	7	9	1	3	5
1	4	7	9	5	8	3	6	1	2	1	2
5		3	6	3	2	2	1	9	3	4	3
1	3	1	3	7	1	4	9	6	2	5	3
2	1	3	1	3	7	1	5	1	3	6	9
1	4	6	5	6	2	3	2	1	4	7	8

如图所示，这个示例带有 5 个 ALS 区域（分别用红色、橙色、绿色、蓝色和紫色五种颜色来表示，注意 r3c6 被三个 ALS 区域共用），也恰好有 5 个花瓣。不过这个例子依然和之前的推理方式一致，所以我们此处不再重新推理，不过题目很漂亮，可以拿来学习和参考。

5-4-6 带 ALS 的链 (Grouped AIC With ALS)

5-4-6-1 普通的例子

5	5	7	1	2	4	3	4	3	6	8
1	3	1	3	2	4	3	4	3	9	5
1	3	8	6	9	5	7	8	2	1	3
7	2	3	3	1	3	1	3	1	3	1
4	3	6	1	8	7	2	5	9	6	3
5	6	5	6	3	1	3	1	3	1	2
1	6	1	6	4	7	9	5	8	3	2
1	3	7	5	2	3	1	3	1	2	4
2	1	3	3	4	3	1	3	6	4	9

如图所示，这个链的写法如下：

$r3c6(7)=r2c6(7-8)=r8c6-r8c1(8)=r5c1-r5c9(6)=r23c9(7) \Rightarrow r3c8 \times 7$

如图所示，这个例子就不用过多解释了，它的原理和之前的 ALS 完全一样，只是注意一下 c9 的 ALS，此处画出的是它的互补视角：如果同假后，2、6、7 将只能放在两个单元

格里，显然不够填，导致矛盾。

5-4-6-2 假 SDC 的强关系

之前我们说过 SDC 技巧，但是我们从来没有想过，把 SDC 插入到链里使用。我们来看看 SDC 如何放到链里去灵活使用。

先来看第一个示例。

^{2 3} 5	4	6	8	³ 5	7	^{1 2 3} 1	9	^{1 2} 5
8	⁵ 7	³ 7	1	2	9	³ 4	³ 6	^{4 5 6} 7
^{2 3} 5	^{1 2} 5	^{1 2 3} 9	4	³ 5	6	^{2 3} 7	^{2 3} 7	8
^{2 3} 9	² 7	^{2 3} 9	6	4	1	8	5	² 9
1	6	5	2	9	8	7	4	3
² 4	² 8	² 9	5	7	3	^{1 2} 6	^{1 2} 9	^{1 2} 6
7	^{1 2} 5	8	9	¹ 6	5	^{1 2 3} 4	^{1 2 3} 6	^{1 2} 6
^{4 5 6} 9	¹ 5	¹ 4	3	8	2	¹ 6	¹ 7	⁶ 9
² 6	3	^{1 2} 9	7	¹ 6	4	5	8	^{1 2} 6

这是一条长为 1 的链，没错，它只有一个强关系构成，虽然这一点有点别扭，但实际上在链的定义和叙述里，我们明确说到“以强关系开头结尾，并且奇数长度”。这个例子确实强关系开头也是强关系结尾（只是它们都是同一个强关系），而 1 确实也是一个奇数。

这条链的逻辑很简单， $r1c7(1)$ 和 $r8c8(1)$ 不同假。那为什么不同假？如果把它俩去掉，则一个看起来像 SDC 的结构 $\{r13c7, r238c8\}$ 五个单元格仅包含 2、3、6、7，而且全部的数字都不跨区出现（2、3 只出现在 b3 里，6、7 只出现在 c8 上）。这显然是一个 SDC 结构了。不过试想一下，如果所有数字都不跨区，而且还没有一样的数字，就意味着这些数必须有且仅有一个，不多也不少。可是，这一共是五个单元格，却只有 2、3、6、7 四种数，使用 SDC 的跨区数组的逻辑是完全填不满的。是的，这就意味着 SDC 就直接自动出错了。故假设错误，即 $r1c7(1)$ 和 $r8c8(1)$ 必须至少有一个为真。

可以看到，这里的 SDC 实际上是一个假的 SDC，根本就经不起推敲。五个单元格只有四种填数是显然直接出错的，而这种结构只是长相很类似 SDC，所以这里把这个结构称为**假 SDC (Fake SDC)**。

这种假 SDC 在链里很喜欢出现，而且这种结构在使用的时候会非常灵活。而且，这种结构跨区域产生了强关系，非常厉害。

接下来我们再来看一则示例。

7	4	5	4	5	1	6	4	5	2	9	5	3
2 3	1 2	1 2 3	1 2 3		7 8	9	5	7 8	2 3	5 8	6	
9	6	4	5	8	4	3	2	7	5 8	1		
1	9	6	2		7 8	3	5 8	4	7 8			
2	3	4	7	2	1 4	9	1 2	6	7 8			
2 5 8	4 5 7 8	4 5 7 8	6	1 4	7 8	3	1 2	9				
4	1	8	3	5	6	9	7	2				
3 5 6 7	5	9	4 7 8	2	4 7 8	1	6 8	1 3	5 8	3		
2 3 6 7	2 5 7	2 3 5 7	9	7 8	1	5 6 8	5 8	3	4			

如图所示，表示如下。

$r2c6=r1c6-r1c2379=r3c3(5) \Rightarrow r2c123 \neq 5$

最难理解的地方就是这里的 SDC 两端的 5 的节点。我们假设 $r1c6(5)$ 和 $r1c2379(5)$ 同假，则在 $\{r1c2379, r3c3\}$ 里只剩下 2、3、4、8，而且 2、3 只出现在 $r1$ 上，4、8 只能出现在 $b1$ 里。并且涉及五个单元格，但五个单元格无法填入 2、3、4、8 四种数字，所以出现矛盾。故这两个关于 5 的节点不同假。

5-4-6-3 节点重叠 (Node Overlapping)

3 1 3	3	4	2 3	8	1 2	5						
6 4 6	4	7	9	4	7	9	4	7	6	3		
9												
5 6 4	6	2	4 5 7 9	4 5 7	8	1	6	2 3 5	4	9	2	3
9	7											
3	8	5	1	6	2 3 5	4	9	7	6			
5												
3	5	4	2	3	1 3 4 6	1	6	4	6	4	6	
7 8 9		7 9	2	7	3							
1	4	6	4 5 8	4 5 8	4 5 8	7	4	9	1 2 6			
7 8	2 3		1	4	6	9	5	3	4 6			
7	8	9	1	4 5 6	2	7	3	8	4	6		
5 6	9	1	4 5 6	2	7	3	8	4	6			
4	7	6	8	3	9	1 5 6	2	1 5 6	7	7	6	
2	3 6 7	5 7	4 5 6 8	1 4 5 8	1 4 5 6	9	1	4 5 6	7	7	6	

这个示例有些奇葩，链如下所示：

$r3c13(7)=r3c9(7-2)=r6c9-r6c2(2)=r6c23-r46c1=r23c1(7)$
 $\Rightarrow r1c3, r2c2 \neq 7$

这个链头 $r3c13(7)$ 和链尾 $r23c1(7)$ 都是 7 的区块。不过有些特殊的是， $r3c1(7)$ 被两个区块组共用。不过，删数也是存在的。因为 $r3c13(7)$ 和 $r23c1(7)$ 至少一个为真，意味着这三个单元格的候选数 7，至少都有一个位置是为真的（ $r3c1(7)$ 为真，就是这个特殊情况）。删掉交集，也就是 b1 的其余位置的 7 啦。

之前说到过，节点不一定只是候选数，也可以是区块，所以这里的链头和链尾就是两个区块了，它们有部分重叠，所以称为节点重叠。

1 4	3		2	4 7	1 7	9	5	4 7	3 8	6
	3	7 8	6	2	5 7	8	1	4 7	3	
4 5		4 7 9						4 7		9
1 4 5		4 5 7 8		1 4 5 6	3	5 6		2	4 7 8	
8	6	4	5 7	9	3	2	5		1	
9	2	5	1 7 8	4	1 7		6		3	
7	1	3	5 6 8	2 8	2 5 6	4	9	5		
4 5	3		1 5 8	6	1 2 5		1 4 5	2 8	4 5	8
6	5 7	1	9	8	4	3	7 8		2	
2	4 5 9	8	3	1 5 7	1 5 7	6	4 5	4 5 9		

当然了，节点重叠并不一定是区块和区块之间，也可能是 ALS 内部产生的部分之间的重叠。比如这个例子：

$$r48c8(7)=r8c8-r8c5=r6c5-r6c9(8)=r4c8(7) \Rightarrow r12c8 \neq 7$$

其中， $r48c8(7)=r8c8(8)$ 是产生于 $r48c8$ 内的 ALS，而 $r6c9(8)=r4c8(7)$ 则是产生于 $\{r4c8, r6c9\}$ 的 ALS。

这个结构就比起刚才的难观察一些了。

5-5 弱 ALS (Weak ALS)

5-5-1 同区域异数弱关系的引入

首先我们来看一则示例。

		3		2	4	5	6	8	1 2	1	5 6
7	5		4	7	9	5	9	7	7		9
		2		1	3	5	7	1	1 2	3	8
7	5	6	5	7	9	5	4	5	7		
	3		1	2		6	5	3	4	5	6
7	8	7	8	9					7		9
2	1		4		6	1	3	7	5	4	1
	8	9	8	9						8	3
	6	5		1	3	2	1		3	9	1
7			7	8	4		4		8		3
	4	6	1		3	8	5	9		1	6
4	7	7							7	6	7
1			6	1	3	1	3	2	9	1	
4	5	7	8	4	5				4	5	7
7	8	7							7	8	
9	3	4	5	1		8	6	1		2	4
	7			4	5			7		7	5
1		2	4	5	7	7	9	1	3		3
4	5							4	5	8	6
8			8								4
											5

如图所示，观察这条链：

$r1c1=r1c4(3)-r2c46=r2c13(5) \Rightarrow r1c1 \neq 5$

按照原始定义，设 $r1c1(3)$ 为假，则我们可以得到 $r1c4(3)$ 真。根据接下来就推不动了，接下来尝试使用弱关系的新定义来说明是否 $r1c4(3)-r2c46(5)$ 。如果弱关系即“不可同真”，那 $r1c4(3)$ 和 $r2c46(5)$ 是否可以同真。

如果同真，则 $r1c4 = 3$ ，并且 $r2c46(5)$ 区块成立，而 $r2c46(5)$ 区块成立，指的是 $r2c46$ 其中一格是 5。观察涂色的 $r1c4$ 和 $r2c46$ 这三个单元格，在 $b2$ 之中，填入 1 和 9 的位置只可能是这三格。这样就意味着，里面至少得需要两个单元格够 1 和 9 填入。可是，刚才同真后，一格被数字 3 占据了，一格被 5 区块的数字 5 占据了，就只剩下唯一一个单元格了。这样就不够 1 和 9 填入了，所以显然是矛盾的。所以， $r1c4(3)$ 和 $r2c46(5)$ 不可同真。故 $r1c4(3)$ 和 $r2c46(5)$ 是弱关系。

弱关系即推得 $r2c46(5)$ 为假。所以最后得到 $r2c13(5)$ 为真，于是构成区块不连续环。删除的交集就是 $r1c1(5)$ 了，所以 $r1c1 \neq 5$ 。

这个结构则是区块不连续环，并带有一个特别的结构，说它是 ALS，又有些不像，因为它用的是异数弱关系。它就被称为弱 ALS (Weak ALS，简称 WALs)。比如图中的 $\{r1c4, r2c46\}$ 就构成一个 WALs 区域。

我们再来看两则例子。

4	3		3		4	5	5	3	1	2	6			
9		8	8		4	7	9	4	5	7	8	9		
3			3	1	2	9	1	2	3		3			
4	6	7	5	4	6	4	5	1	5	4	4	8	9	
9			8		9	8	9	8	9	9	8	9		
2	4	6	1	4	6	4		3	5	7	4	3		
	8			9	8	9	8	9			8	9		
1		1	2	6	1	2	3	1	2	4	5	1	2	
7		8			7	9		7	9			8	9	
1	3		1	2	5			3	3	3	1	2	3	
7		5	9	8	1	5	4	7	9	7	9	8	9	
1	3	1	2	5	3	1	2	6	1	2	3	1	2	3
4		4	5	5		7	9	7	9	7	9	8	9	
7		8		7	9									
8	1	6	2	5	7	1		3	3	3	4	3	3	
								6	4	6	4	9	9	
5	9	4	3	2	6	8	1	7						
1														
6	3	7	1		1	1		2		6	9	5		
			9	8	9	8	9							

如图所示，这条链比较麻烦，但逻辑很清晰。

$r7c6(9=1)-r7c2=r9c1(1)-r5c1=r5c7(7-6)=r5c8-r9c8(6=9)$
 $\Rightarrow r7c789, r9c456 <> 9$

当我们得到 $r9c1(1)$ 为真时，必须让 $r5c1(7)$ 为假。答案是显然的，因为在 $c1$ 上只有 $r12569c1$ 五个单元格里才能放下 3、4、6、9 四种数字，而当如果 $r9c1(1)$ 和 $r5c1(7)$ 同真时，会占据其中两个单元格，导致还剩下三个单元格，此时 3、4、6、9 是无法完整放进去的，所以会出现矛盾。后面的逻辑将比较清晰，这里就不讨论了。

我们再来看最后一个例子。

1	2	3	6	2 3	1 2	5	2 3	4
8 9	8	9	5	8	7	1	7 8	
4	7	3 6	5	2 3		9	2 3	2 3
1	2		1 3	1 2		3	8	
8	8 6	5	4	7	7	2 3	6	9
7	1	8	2	6	3	9	4	5
3	4	6	9	7	5	8	1 2	1 2
5	9	2	4	1	8	6		3
6	3	1	7	5	4	2	9	8
	5	4	1	3	2 3	1 2	3	6
8 9	7	9	8	8 9		4	7	
2	4	4	1 3	3		6	3	1
8	7	9	8	8 9	7	5	7	

如图所示，我们来看这一则示例。这一则示例里是跨区的两个节点的弱关系。实际上，这个示例并不难，可以看到，如果 $r3c78(3)$ 和 $r9c9(1)$ 两个节点同真，则会导致一个 ALS 区域只能放 2，出现矛盾，所以不同真，即弱关系成立。

不过实际上，这个例子使用的是 ALS 而不是 WALS。

5-5-2 WALS 的利用

实际上，WALS 我们很少去使用它，除非是特殊情况，比如为了缩短链的长度，才会刻意去寻找这种弱关系（当然，ALS 的强关系也会去找，所以它确实不是特别实用）。

那么，WALS 和 ALS 不同，它的使用则更为古怪。

			1	2	3			
			4					
			1	2	3	1	2	3
			4		4			

如图所示，我们先假设图中 b2 内，只有 r1c4 和 r3c45 这三个单元格可以填 1 和 2。这样一来，{r1c4, r3c45}(12)就是一个 WALS 区域，而 3 和 4 里面最多只有一个数可以放到里面。那么，最容易可以得到的一个结论就应该为：这个 WALS 区域内，任一格的 3 和任另外一格的 4 构成弱关系，因为它们不可同真，否则 1 和 2 不够填。比如，这个例子里面，r1c4(3)-r3c5(4)是成立的。

由于弱关系是“不可同真”，所以我们再来思考一下，还有什么结构为真时也会占据一格（或者至少一格）的位置呢？就是区块了。来思考一下，r13c4(3)-r3c5(4)可以吗？答案是可以的。因为同真时，r13c4(3)区块成立，即 r13c4 其中有一格是 3，而同真意味着 r3c5 是 4，那么一定存在两个不同的单元格，使得其中一格是 3、另外一格是 4 了。这样依然会把 1 和 2 挤到一格之内，使得不能填数。所以这个情况依然成立。而且比如 r13c4(3)-r3c45(4)也可以。两个区块同真时，r13c4 一格是 3，r3c45 一格是 4，虽然 r3c4 一格是被两个区块共用的单元格，但很明显，当 r3c4 被 3 或 4 的其中一个数占据时，那另外一个数显然就不能再占据 r3c4 啦，因为 3 和 4 肯定不同格。所以，两个区块为真，依然会占据这个 WALS 里面的其中两个单元格，那剩下一格根本不够填入两种数字 3 和 4，所以矛盾了，故这个弱关系还是成立的。

甚至是更奇葩的弱关系：r13c4(3)-r1c3(4)，这样也是成立的哦！只是这样写法还不如直接写成 r3c4(3)-r1c3(4)。所以在 WALS 区域下，任意两个非“仅填入此区域”的候选数的部分都是弱关系。说白了，“仅填入此区域”指的是这里的数字 1 和 2。

强 ALS

- 弱 ALS

- 规则：在同一区域下，只有 $(n+1)$ 个单元格可以让 n 种候选数填入。
- 弱关系：任意两个不是“仅填入此区域”的候选数部分都是弱关系。

实际上，WALS 也是有互补视角的。ALS 的互补视角更类似于隐性数组，那么 WALS 的互补视角，则更类似于显性数组。

[illegible]

此时可以看到，如果 **r1c4(3)**和 **r2c46(5)**同真的话，**r3c6** 则无法填入任何数字，所以产生了矛盾；而第二个示例的互补，则是这样的：

4	3		3		3	1	2	6
4	9	4 5	8	5	8			
3		7	5	3	1 2	3	3	3
4	6		8	4	6 4 5	9	4	8 9
9				9	8 9	8 9		
2	4	6	1	4	6 4	5	7	4
	8			9	8 9			8 9
1		1 2	6	1 2	3	4	5	1 2
7		8		7	9			8 9
1	3	1 2	9	8	1 5	4	3	1 2 3
	5							
7	3	1 2	5	3	1 2	6	3	1 2 3
4	5	4 5	8	5	7	9	7	8 9
7		8		7	9		8 9	
8	1	6	2	5	7	3	4	3
						9	6	4
5	9	4	3	2	6	8	1	7
	6	3	7	1	4	2		5
				9	8 9			

显然，这个例子看互补视角也更轻松一些。

5-6 毛刺数组 (Buried Subset)

数组这个技巧在我们之前其实介绍过，不过之前的数组（显性、隐性、区块数组甚至是跨区数组）都是可以直接用的。现在来看一个不能直接用的数组形式。

5-6-1 显性毛刺数组 (Buried Naked Subset)

我们先来看一则示例。

4	3		3	2 3	1	2	6	3	5	3
4	7	4	7	9 7		4	8	7 8 9		7 8 9
1	3	1	3	3	3	7	4	3	6	2
4	5 6		5	4 5 6	5		8 9	8 9	8 9	
8		3	2 3	5 6	2 3	2	2 3	4	1	3
	7	9 7		9	5				7	6 9
4	5 6	4 5	4 5 6	7 8	2	1 2	4 6	9	3	5
7		7	8		5 6	8		7 8	7 8	
9	5	1	2	5	3	2	8	6	2	4
	7			8					7 8	
4	3	2	4 5 6	8	7	1	4 5 6	4	5	8 9
			8			8	8		8 9	
4	5	6	9	2 3	8	2	2 3	2 3	2	1
7					8	8		5	4	
2	1	3	3	6	3	1	3	7 8	7 8	3
	4 5	7	4 5	7	4 5	6	7	5	4	6 5 6
1	3				1	2	6	2 3	2	3
7		8		4		5		7 9	7 9	9

如图所示，链如下表示：

$r7c6(7)=r8c6(7-1)=r9c5-r9c1(1)=r9c13(37) \Rightarrow r7c1 <> 7$

按照最开始的逻辑，设 $r7c6(7)$ 为假，则得到 $r8c6(7)$ 真、 $r8c6(1)$ 假、 $r9c5(1)$ 真、 $r9c1(1)$ 假。而此时发现， $r9c1(1)$ 假之后， $r9c13$ 形成了 3、7 显性数对。

换句话说， $r7c6(7)$ 有两种填数情况，其为假时，得到 $r9c13(37)$ 显性数对结构，换句话说， $r9c13(37)$ 显性数对在此时是为真的（结构成立就为真，结构不成立时为假）。所以 $r7c6(7)$ 和 $r9c13(37)$ （即这个显性数对）之中至少有一个为真，于是删除掉它们的交集。那一个显性数对和一个候选数怎么找交集呢？

- $r7c6(7)$ 为真时，可以删的是 $r7c6$ 所在区域下所有的单元格（相关格）内的 7；
- $r9c13(37)$ 为真时，可以删的是 $b7$ 和 $r9$ 内其余单元格的候选数 3 和 7。

所以对于此题来说，它们的交集，只有 $r7c1(7)$ 了，所以 $r7c1 \neq 7$ 。

这个结构是不是很神奇？你会发现，在末端嵌入了一个显性数对，但前提是 $r9c1(1)$ 为假时，这个显性数对才真正成立，所以它可能并不是一个真正的数组，而类似于我们之前学到的鱼鳍的逻辑。所以我们一般称这种结构叫**毛刺显性数对**或显性毛刺数对（**Burred Naked Pair**），**毛刺**（**Burr**）一词用来表示结构多出来了一点点东西，并使这一点为假时，本体结构才真正成立。关于毛刺，我们还将在此链的构造的内容里详细谈论它的基本逻辑和思维方式。

实际上，毛刺和鱼鳍是完全一样的东西，只是鱼鳍一般针对于鱼结构，毛刺则针对于其余的非鱼结构的技巧。而正是因为它们如此接近，所以毛刺一词在英文里用的 **Burr** 一词，也动词化，加上了分词性形容词的后缀 **-ed**（**burred**）。

5-6-2 隐性毛刺数组（Burred Hidden Subset）

2 5 6	1	4	2 5	9	5 7 8	3	5 6
7	2 3 5	3 5 9	6	1 2 3 8	1 3 5 8	1 5 8 9	4
8	3 5 6	3 5 6 9 7	1 3 5 4	1 3 4 5 7	1 3 4 5 7	2	1 5 6 7
5 3	9	5 3 7 8	1 3 7 8	6	2	4	1 3 7 8 9
4 6 4 8	2 6	3 6	1 3 7 8	5	1 3 7 8 9	1 7 8 9	1 2 3 7
2 5	2 3 5 8	1 8	4	3 8	3 7 8 9	5 7 8 9	6 7
4 3 6	7	3 6 8	1 2 3 5 8	1 2 3 4 8	1 3 4 5 8	1 2 4	9
1	4 5 3	5 3	9	4 2 3	6	2 7	3 4 7 8
9	4 8	2	1 3 8	7	1 3 4 8	6	5 1 3

如图所示。链如下所示：

$r46c1=r7c1(3-6)=r7c3(6-8)=r45c3(78) \Rightarrow r45c3 \neq 3$

如果 $r46c1(3)$ 区块为假时， $r7c1(3)$ 为真（之前也提到过这一点，区块和候选数的强关系怎么推导）。于是， $r7c1(6)$ 假、 $r7c3(6)$ 真、 $r7c3(8)$ 假。观察 $c3$ ，我们可以发现， $r7c3(8)$ 为假时， $c3$ 刚好产生 78 隐性数对，在 $r45c3$ 。换句话说，当 $r7c3(8)$ 为假时，

r45c3(78)隐性数对为真。于是也就得到了 r46c1(3)区块和 r45c3(78)隐性数对至少一个为真。

- r46c1(3)区块可以删除的是 c1 和 b4 内的其余位置的候选数 3；
- r45c3(78)隐性数对可以删除的是 r45c3 内其余的候选数。

所以，它们的交集，应该是 r45c3(3)。所以 r45c3 <> 3。

那么，类比于刚才的显性毛刺数组，**隐性毛刺数组**（或**毛刺隐性数组**，**Burred Hidden Subset**）指的是差一点形成隐性数组的情况，比如例子之中的 r457c3(78)（别忘了 r7c3 也有候选数 7 哦）。

5-6-3 毛刺数组节点的真假性

刚才我们接触到的两则示例，都是当节点进入的时候使得毛刺数组为真的结构，我们此时就接触了一种新的节点类型：毛刺数组。

不过，毛刺数组为假是个什么样子呢？所谓的毛刺数组为假，就是让毛刺数组结构的数组本身不成立，即破坏数组。

如何去破坏呢？我们知道，数组的核心本质是“n 个单元格只能填入 n 种数字”和“n 种数字只能放在 n 个单元格里”，那么破坏的方式很简单：要么里面有重复数字（这显然不可能，因为数组结构的内部是不可能出现重复数字的，一旦出现就违背了数独规则）；要么里面包含我们不需要的数字（比如一个毛刺数对里包含候选数 1、2、7，如果 7 不见了就形成了 1 和 2 的数对。那么为了破坏数对，我们在其中填入数字 7，这样就破坏了 1 和 2 的数对结构）。只要出现这两种情况，毛刺数组就被破坏了，也就称毛刺数组为假。

例如我们来看一则毛刺数组为假的示例。

	1			4	3	2	1	2		8
5	6		6	5		7	6	5	5	
3	1	3		1		7	9	6	4	5
5	4		2		8		7	9	7	9
	1		7	1	2	2	6	5	1	
6	4	6			6				4	2
8		8	9		9		9		9	3
	6	5		2	6	1	4	6	4	
7	9		8	9	9		8	9	7	9
4		6	3	5	6	5	6		2	5
7	9	9		9	9		9		7	9
6	2	1		5	6	7	4	6	4	5
9				9				9		8
1		4	8		5	6	2	6	3	5
	7	9		3			7		7	9
3			6	2	2	3		8	1	4
7		7		7			7			
5	3		3	2	3	4	1	5	5	6
8	9	7	8	9	7		7	9	7	9

如图所示，我们从 r5c5(5)开始推理，并假设其为假。按照顺序，我们可以依次得到 r7c5(5)为真、r8c4(5)为假、r8c1(5)为真。此时，当 r8c1(5)为真后，r89c1 不管你怎么填，显然都是无法形成 2 和 7 的隐性数对了，所以该结构一定为假，故 r89c1(27)为假。

接着，当毛刺隐性数对为假的时候，**r4c1(7)**就必须为真了。可这是为什么呢？试想一下，毛刺隐性数对为假之后，由于是2和7的隐性数对的关系，按常理说，在**c1**上的2和7应该只能放在**r89c1**里，但实际是7出现了三次(**r489c1**)，而2只有两次(**r89c1**)，而且此时经过推导，我们已经得到了**r8c1(5)**为真了，此时**r9c1**是不得不填入2的，否则**c1**就放不了2了。那既然**r9c1 = 2**的话，7就只能放在唯一一处地方：**r4c1**了，因为**c1**只有**r589c1**三处位置可以放7。

所以，当毛刺数对节点为假的时候，不得不让这个**r4c1(7)**为真，链才可以继续推导下去。

当然，如果你对逆否命题比较熟悉，你可以尝试使用逆否命题论证：如果我们设定原命题是“如果2和7的隐性数对为假，则必须**r4c1(7)**为真”；那么其逆否命题则是“如果**r4c1(7)**为假，则2和7的隐性数对为真”。这是显然的，因为**r4c1 <> 7**后，**c1**只有两处可以放下2和7，而且都是**r89c1**，所以这两个单元格形成关于2和7的隐性数对，即隐性数对为真。所以逆否命题是成立的；既然逆否命题成立，那原命题也成立，所以原命题的论证结果为真（这实际上是把这个强关系反向来证明了，可以看到，从反向理解这个逻辑反而比正向理解要简单，所以建议大家如果不能理解一些逻辑的时候，尝试使用逆否命题和原命题等价的思维来解释）。

当然，这里再给出第三种理解。我们直接把**r5c2(7)**和**r8c1(5)**用弱关系连起来。因为它们同真的话，**c1**将产生三个单元格只能放下5和7，导致矛盾的出现，所以它们不同真，即形成弱关系，如图所示。

5 6 9	1 6 9	5 9	4	3	2 6 9	1 5 7	2 5 7	8
5	3 1 4	3 9	2	1 7	8 9	6	4 5 7	5 9
6	1 6 8	4 6 8	7	1 2 6 9	2 6	5	1 4 9	3
6 7	5	8 9	2 6 9	1	4 6 8	2 4 7	3	6 7
4	6 7	3 9	5 6 9	5 6 9	6 9	2	5 6 9	1
6 9	2	1	5 6 9	7	4 6	4 5 9	8	5 6 9
1	7 9	4	8	5 6 9	2 6 7	3	5 6 7	2 5 9
3 3 7	3 7	3 7	6	2 7	2 3 7	8	1	4
2 3 5	3 7	3 8	5	2 3 5 6	4	1	5 7	2 5 6 9

5-6-4 毛刺数组的解构 (Deconstruct of Burred Subset)

实际上，不论是显性毛刺数组还是隐性毛刺数组，我们都可以通过解构的方式，把结构本身解构 (Deconstruct) 为一个强或弱的 ALS。我们举例说明一下。

1 2 5 8 9	1 2 5 8	6 9	2 5 6 7	4 5 6	1 4 5 8	3 4 5 8
1 2 5 7	3	4	8	9	2 5	1 2 7 6
2 5 7 8	2 5 8	6 7	3	4 6	1 9	2 7 8 4 5 8
1 2 4 8	1 2 4 8	5	1 2 7	2 6 8	2 6 7 8	1 9 3
1 2 8	9	1 2 3	1 2 5	2 3 5 6 8	2 3 5 6 8	5 6 8 4 7
6	7	1 3	9	4 5 8	4 2 1	5 8
1 2 5 7 9	1 2 5	8	4	2 3 5 7	1 3 6	1 2 9
3	1 2 4	1 2	2 6	2 6 9	7	5 1 2 4 8
2 4 5 7 9	6	7 9	2 5 1	5 8	3 4 8	2 4 8 9

如图所示，这个链比较复杂，嵌入了一个毛刺显性三数组。

首先假设 $r6c56(8)$ 区块节点为假，则 $r6c89(8)$ 为真，于是 $r45c7(8)$ 为假。此时由于 $r245c7$ 里的 8 全部没有了，于是形成显性三数组，所以 $r245c7(156)$ 毛刺显性三数组节点为真，所以数组为真后可以删除 $r7c7(1)$ ，故 $r7c7(1)$ 为假、 $r7c7(3)$ 为真、 $r7c5(3)$ 为假， $r9c6(3)$ 为真， $r9c6(8)$ 为假， $r8c5(8)$ 为真。

所以这个题目的删数在于 $r45c5(8)$ ，这是头尾两个节点的交集。

不过，有些时候，我们不得不将这个三数组解构。因为三数组有些时候我们不一定能看到，而取而代之地使用 ALS 的观察视角。所以实际上，显性三数组将对应一个合格的 ALS 区域，如图所示。

1 2 5 8 9	1 2 5 8	6 9	2 5 6 7	4 5 6	1 4 5 8 3	1 2 4 5 8
1 2 5 7	3 4	8 9	2 5 1	2 5 9	1 2 7	2 4 5 8 6
2 5 7 8	2 5 8 7	6	3 4 6 1	2 6 9	2 4 5 8 7 8	2 4 5 8 3
1 2 4 8	1 2 4 8	5	1 2 7	2 6 8 7 8	9 3	1 2 4 5 8 7
1 2 8	9	1 2 3 8	1 2 5	2 3 5 6 8	4 7	1 2 4 5 8 9
6 7	1 3	9	4 5 6 8 3	4 5 6 8 3	2 1 6	1 2 4 5 8 9
1 2 5 7 9	1 2 5 4	8	4 5	2 3 5 7	1 3 6	1 2 4 5 8 9
3	1 2 4	1 2	2 6 1	2 6 9	7 5	1 2 4 5 8 9
2 4 5 7 9	6	7 9	2 5 7	1 5 8	3 2 8	2 4 5 8 9

如图所示，我们将刚才的毛刺显性三数组节点解构为了一个两个区块节点的强关系： $r24c7(1)=r45c7(8)$ ，而实际上，毛刺显性三数组里我们也并没有完整地使用到任何一个数字（比如原来的数组是 1、5、6，而实际上我们涉及的强关系仅需要 1 和 8 就足够了，5 和 6 根本就没用到），所以我们可以尝试解构，把毛刺数组解构为上述的 ALS 区域的强关系来使用；反之，有时候你也可以使用**构造（Construct）**，将一个 ALS 改为一个合适的数组形式，来丰满结构，使之更容易理解。不过，有些时候 ALS 改写出来的数组可能不是很符合预期，所以我们不建议随时随地都使用构造。

由于对称性的约束，显性毛刺数组对应了强 ALS 区域，那么隐性毛刺数组就对应了 WALs 区域（即弱 ALS）。因为 ALS 用的强关系，而 WALs 区域用的是弱关系，所以对于隐性毛刺数组的解构用得非常少。但是，你依然可以尝试从其它层面对毛刺隐性数组解构，比如拆分成多个单元格，进而得到意想不到的东西，例如下面的这则示例。

8	6	5	1	2	9	4	7	3
4 3 9	4 9	1	7 5 6 8	5 6 8	2 5 6 8 9	5 6 8	5 6 8	5 6 8
3 9	2 7	5 6 8	4 5 6 8	5 6 8	1 5 6 8 9	5 6 8 9	5 6 8	5 6 8
2 9 7 9	5 9	8 3	5 6 7	1 5 6 8	5 6 9	2 5 6 8	4 2 7 8	4 2 7 8
6 5 7 9	3 4	5 6 7 8	4 5 6 7 8	1 5 6 8	5 9	1 7 8	2 7 8	2 7 8
2 5 9	1 4	5 6 7 8	9 7 8	6 8	3 6 7	8 7	2 7 8	2 7 8
7 3 9	8 1	2 5 6	5 6 7	4 5 6 8	3 2 7	4 5 6 8	2 5 6 8	2 5 6 8
1 4 5	6 4 5	3 4 5	7 4 5 6	8 4 5 6	9 4 5 6	7 4 5 6	3 4 5 6	1 4 5 6
4 5	8 4 5 6	9 4 5 6	7 4 5 6	3 4 5 6	1 4 5 6	2 4 5 6	3 4 5 6	1 4 5 6

如图所示，链如下表示：

$$r23c6(38)=r6c6-r6c8=r5c9-r3c9(8)=r3c49(56) \Rightarrow r3c6 \lt \gt 56$$

这个链的精彩之处，是在于链的开头和结尾。链头是 3 和 8 的毛刺隐性数对，而链尾则是 56 毛刺显性数对。那么根据链的证明思路，链头和链尾至少有一个节点成立。所以，5、6 的显性数对和 3、8 的隐性数对的交集为 $r3c6(56)$ ，故 $r3c6 \lt \gt 56$ 。当然，我们依然可以尝试解构（注意下面的解构方式），并得到下面两条链结构：

$$r3c6(38)=r6c6-r6c8=r5c9-r3c9(8)=r3c49(5) \Rightarrow r3c6 \lt \gt 5$$

$$r3c6(38)=r6c6-r6c8=r5c9-r3c9(8)=r3c49(6) \Rightarrow r3c6 \lt \gt 6$$

在上方书写的两种形式的开头，可能你会觉得链的写法里是不是漏掉了 $r2c6$ 这格，其实不然。 $r3c6(38)=r6c6(8)$ 是成立的，即使是两个候选数和一个候选数的特殊强关系。当它们同假时， $r3c6$ 将不会填入 3 或 8， $r6c6$ 也不是 8。这样一来，观察 $c6$ ，你会发现， $c6$ 内填入 3 和 8 的位置，就只剩下了 $r2c6$ 一处，可这样一格是不够填 3 和 8 两种数字的，所以是不行的，强关系就成立了。

另外，结尾的强关系 $r3c9(8)=r3c49(5)$ 和 $r3c9(8)=r3c49(6)$ 则直接利用之前的毛刺显性数组的解构模式进行解构，所以得到两个不同的强关系，所以有不同的删数。当然了，如果你嫌别扭，依然还是可以写作原来的数对形式的。所以实际上，这种形式的毛刺数组就不建议解构理解了，不过链头依然是可以解构的。

5-6-5 毛刺数组的其它使用方式

5-6-5-1 节点重叠

5 8	4	1	3	5 8	9	7 6	2	7 6
3	7	2 5	2 4	2 5 6	4 5 6	8	9	1
2 6 8		2 9	2 8	7	1	3	4	5
5 6	3 6 9	5 7 9	4 5 8 9	3 4 6 8	4 6 7 8	5 6 7 9	1	2
2 5 6	3 6 9	8	5 9	1	2 6 7	4 5 7	5	3 5 6 7
4	1	2 5 7 9	2 5 9	2 3 4 5 6 7 8 9	2 6 7	5 6 7 9	8	6 7
9	5	3	7	3 8	2 8	1	6	4
7	2	6	1	4 5 9	4 5	5	3	8
1	8	4	6	5 9	3	2	7 5	7 9

如图所示，链如下表示：

$$r56c9(39)=r9c9-r8c7=r8c5(9-4)=r6c5-r6c9=r5c9(3) \Rightarrow r5c9 \lt \gt 67$$

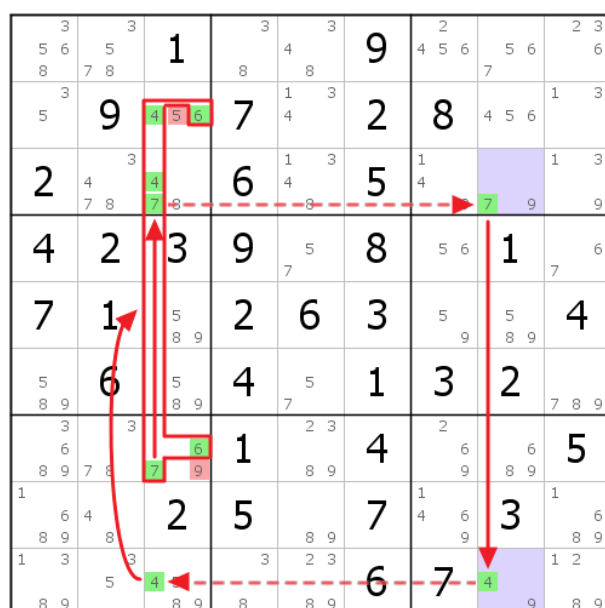
链头是一个 3 和 9 的毛刺隐性数对，而到 $r8c5(4)$ 时为假，而 $r8c5(4)$ 和 $r6c5(3)$ 不可同假，否则 $c5$ 之中只有一处可填 3 和 4 两种数字，这显然是不行的，所以它们不可同假，即 $r6c5(3)$ 应为真。最后得到了 $r5c9(3)$ 为真，所以链头 3 和 9 的毛刺隐性数对和链尾 $r5c9(3)$ 至少有一个节点为真，删掉它们的交集。39 隐性数对成立时，删除 $r56c9$

的其余候选数，而再算上 $r5c9(3)$ 成立的时候，能够删除的位置只有 $r5c9$ 的其余候选数。所以 $r5c9 \nleftrightarrow 67$ 。

这个例子有些别扭的地方是，链头是一个毛刺隐性数组，而链尾，则是这个毛刺隐性数组里涉及的一个候选数，这就好像链尾被链头“吃掉了”一样。

接下来我们再来看一个示例。

5-6-5-2 间接删数的链



如图所示，这也是一个包含节点重叠的链结构，而且和上面那个示例的方式类似，只是上例用的是显性毛刺数组，而这里用的是隐性毛刺数组。链的表述如下：

$$r7c3=r3c3-r3c8(7)=r9c8-r9c3(4)=r237c3(467) \Rightarrow r2c3 \nleftrightarrow 5, r7c3 \nleftrightarrow 9$$

这条诡异的链，就算是写出来了，也会发现 $r2c3 \nleftrightarrow 5$ 。它是怎么删掉的呢？

我们在之前的不连续环里介绍了，节点和删数是弱关系。所以，只要我们找到链头和链尾同时和可能要删的这个节点都是弱关系的话，那么删数就成立了。那么问题就变为了证明 $r7c3(7)-r2c3(5)$ 。

如果你还不能理解为什么“链头和链尾和删数节点是弱关系，那么删数成立可以删”的逻辑的话，你可以这么去想这个问题。针对于此例而言，如果 $r7c3(7)$ 和 $r2c3(5)$ 是弱关系，就意味着两者不可同真，也就意味着，当 $r7c3(7)$ 为真的时候， $r2c3(5)$ 就必然为假。这样就可以说明当 $r7c3 = 7$ 时， $r2c3 \nleftrightarrow 5$ 了。

确实我们有 $r7c3(7)-r2c3(5)$ 。因为它们同真时，观察 $c3$ ，针对于 4、6、7 来说，数字 6 就无位置可填了，就这样矛盾了。所以弱关系得以成立。

刚才说到了，因为弱关系的缘故，所以 $r7c3(7)$ 为真的时候，确实可以确定， $r2c3 \nleftrightarrow 5$ 。所以，因为这样的缘故，本应该产生的 467 隐性三数组也可以删掉 $r2c3(5)$ ，所以 $r2c3 \nleftrightarrow 5$ 也是结论的一部分。

这条链的删数结论的说明甚至用到了隐性毛刺数组才得以解释，像这样的链结构，不能直接得到全部删数的链，则可以称为一种间接删数的链。

至此，ALS 的内容就全部结束了。可以看到，ALS 的例子非常有趣，而且灵活。在我们平时运用和观察之中，将变得非常有用处。

Part 6 待定致命结构 (Almost Deadly Pattern)

我们已经讲完了 ALS 的基本用途，那么接下来我们来说一说，如何把之前学习的致命结构的逻辑套用到链里。

6-1 待定唯一矩形 (Almost UR)

当然，因为这里的唯一矩形并非真正的唯一矩形，和毛刺数组、ALS 类似，它并不是真正的数组结构，需要链的介入才能使用，所以我们把这里的唯一矩形叫做**待定唯一矩形** (**Almost UR**，简称 **AUR**)。

6-1-1 AUR 里的强弱关系

6-1-1-1 构建出 UR 标准类型的强关系

1	5 8	5	2	3	4	6	5	9
2	4 5	6	1 7	1 5	9	3	8	4 5 7
4 7 9	3	4 7 9	7 8	5 8	6	2	1	4 5 7
4 8	7	4 8	1 9	1 9	3	5	2	6
5	1	2	6	4	7	9	3	8
6 9	6 9	3	5	2	8	7	4	1
3	5 8	5	4	2	1	6	5 7	
4 6 4 6	1	3	7 9	5	8	7	9	2
7 8 9	2	5 7 8 9	8 9	6	1	4	5 9	3

如图所示，设 $r8c5(9)$ 为假，我们可以依次得到 $r7c5(9)$ 真、 $r7c23(9)$ 为假。

当此时区块为假时，意味着 $r7c23$ 都不是 5，则 $r17c23$ 形成 UR 的标准类型，故 $r1c3(7)$ 为真。你也可以理解为 $r1c3(7)=r7c23(9)$ ，原因是不同假，同假后导致 $r17c23$ 只有 5、8，构成 UR 的致命形式。

后面的逻辑就不难了。最终我们得到了这个不连续环，并删除了 $r8c5(7)$ 。

这个示例里我们使用了 UR 的致命形式，使得跟 UR 无关的两侧节点互为强关系，这一点很重要，后续的示例都将使用这种思路来解题。

接下来我们再来看一则示例。

6-1-1-2 构建出 UR 区块类型的强关系

8	2			3	1		3	5
5	7	3	8	2	9	1	4	6
	4						2	8
	1		4					
				6			1	9
7				1		5	6	
		2	5	4		6	3	7

如图所示。链表示如下：

$r9c6(1)=r7c6-r78c4(6)=r78c7(8)$

$-r4c7(8=7)-r13c7=r1c8-r1c5=r6c5(7-8)=r78c5(8)$

$\Rightarrow r9c6 \langle \rangle 8$

假设 $r9c6(1)$ 为假，则 $r7c6(6)$ 为真、 $r78c4(6)$ 区块为假。此时注意到，当 $r78c4(6)$ 为假时， $r78c47$ 只包含 2、8、9，且 8 只出现在 $r78c7$ 里。根据 UR 区块类型的逻辑，我们应当得到区块为真的结果，所以 $r78c7(8)$ 此时为真，故形成强关系。接着向下推理，后面的逻辑就不难了。

这个例子是区块不连续环，所以删数是 $r9c6(8)$ 。

这个示例我们用到了 UR 区块类型的基本逻辑和思维，把两个区块节点直接使用强关系连了起来。

6-1-1-3 构建出 UR 共轭对类型的强关系

1	4 6	1	6 4 6		5	9	3 6	3 6	2
8	3	7 8	7	6	7 8	4	7	7	
8		2		6	7 8	4	9	1	5
9	7	6	5	2	3	1	8	4 6 4 6	7
2	1 3	6 4 6		3		5	1	6 4 6 4 6	
1	6		8	4	2		5	2 6	3
1 3	4	5			6			8	
5	3 6	2	6 6	1	2	4	3		6
7	3	1	5	4	8		2 3	2 3	9
	3 6	4	2		3	2	1 3	5	1 6

如图所示，假设 $r2c5(8)$ 为假，则 $r2c5(7)$ 为真、 $r79c5(7)$ 为假，接着我们应当使用强关系使链继续持续下去。此时发现 $r7c2(9)=r79c5(7)$ 。这是因为，一旦两个节点同假后， $b7$ 内能放下 9 的位置只剩下 $r79c3$ ，形成宫区块，而 $c3$ 含有 2 的共轭对，导致 2 只能放在 $r79c3$ 里。如果此时 9 也只能放到 $r79c3$ 的话，则 2 和 9 都只能放在 $r79c3$ ，形成关于 2 和 9 的隐性数对，而此时 $r79c5(7)$ 都为假，也只有 2 和 9，所以 $r79c35$ 此时形成关于 2 和 9 的 UR 的致命形式，所以矛盾。故这个强关系是成立的。

接着向下推理，直到 $r1c2(8)$ ，于是删除交集。

这条链使用到的是 UR 结构外部和内部两个看似完全不相关的节点的强关系：当它们同假时，配合一个共轭对，形成了致命形式。

6-1-1-4 构建出 UR 死锁形式的弱关系

1	4 6	4 8	3	2	5 6	6 8 9	5 8 9	7	2 3	6	5
3		2	3	4	1		5 8	3 3	5 6	5 6	9
7 8	6		7 8		7	3	4	2	6	1	4
	6 4	6 8 9	5		6			7	5	4	2
5 6	3	1		5 6		6 8 9	7		2	5	
5 6	4	6 7 8	9	3	2	4 5 8		5 6	5 6	1	5
2	4	6 4 6		1		6 4 5 9		8	5 6 7 9	3	
4	1		3		5		1 2	3	2 3		6
3	6	9		7	4	3	1 2	2	3	9	8
7 8 9	3	1	2		4	6		1	5	4 5	

如图所示。首先假设 $r9c1(3)$ 为假，则得到 $r7c3(3)$ 为真、 $r7c78(3)$ 区块为假、 $r89c7(3)$ 区块为真。如果此时 $r89c7(4)$ 区块节点也为真，则这两个单元格就产生了两个区块。试想一下。两个区块都成立，而且在同样两个单元格，这意味着什么？这意味着两个单元格里使得同时成立，必须一个是 3，一个是 4。但凡少一个数不出现（比如 3）都不叫区块为真。所以两个区块在同样两个单元格并且为真（例如图上 $r89c7$ ）这种形式，就表示一个关于 3、4 的隐性数对（当然，前提是建立在 $r7c78(3)$ 区块为假的基础上的）。

既然是一个隐性数对，就意味着两个单元格都只有 3 和 4，此时 $r89c57$ 就构成了只有 3 和 4 的 UR 的致命形式，所以矛盾了；所以，这两个区块不能同时为真，即给定的关系 $r89c7(3-4)$ 成立。

接着向上推理，可以得到一个 ALS 区域和一个毛刺显性数对。最后链结束于 $r2c78(36)$ 。所以删数是 $r9c1(3)$ 和 $r2c78(36)$ 的交集，即这里的 $r2c1(3)$ ，所以我们有 $r2c1 <> 3$ 。

这个例子更为特殊的地方是，利用了区块和区块之间不同真，同真便导致形成隐性数对，进而出现致命形式，便用到了弱关系。

6-1-2 万用 UR (UR of Panacea)

下面来介绍一种 AUR 结构，这种 AUR 极为特殊，它的四个顶点都包含一个相同的额外数字。

1 4 5	1 4	3	2 7	2 7	1 2 6	9	6 8	5 6
2	1 7 8		5	9	1 6	4	6 7 8	3
6	9	5 7	4	3	8	2 5 7	1 2 7	1 2 5
1 4 7	1 2 4 7 8	2 4 6 7 8	3	4 8	2 6	2 6 7 8	5	9
4 7	3 5	2 4 6 7 8	2 6 4		9	2 3 6 4 6 7 8	1 2 7	1 2 6
4 9	3 8	2 3 4 6 8 9	1	5	7	2 3 6 4 6 8	2 6	2 6
5 7	2 3 7	2 5 7	9	2 7	4	1	2 7	6 8
4 7 9	6	4 7 9	8	1	5	2 7	3	4 7
8	2 4 7	1	2 7	2 7	3	2 5 6 7	9	2 4 5 6 7

如图所示，首先我们可以利用之前构建区块类型的强关系来得到一个特殊的强关系 $r1c45(6)=r9c45(6)$ ，接着使用弱关系和强关系让链得以延续，最终删除 $r1c8(6)$ 。

这个例子特殊的地方在于， $r19c45$ 这个 AUR 实际上带的两个区块涉及的数字不一定是 6。如果是 6，那么 UR 就是 2 和 7；如果我们把两个 7 区块看成强关系，那 UR 此时涉及的就是 2 和 6；如果是 2 区块的话，则 UR 涉及 6 和 7。这种类型的 AUR 可以根据你的个人喜好，在后面使用到哪种情况，它就可以变为哪种情况，所以这种 AUR 称为**万用 UR**（或**万能 UR**，**UR of Panacea**），其中 Panacea 是万金油的意思，即表示怎样都可以用。

6-2 待定拓展矩形和唯一环 (Almost XR & Almost UL)

使用**待定拓展矩形** (Almost XR) 和**待定唯一环** (Almost UL) 的方式完全和 AUR 类似，所以强弱关系的逻辑就不再介绍了，我们来看下面的示例。

6-2-1 待定拓展矩形 (Almost XR)

2 8	6 3	1	2 9	4 8	6 9	4 8	2 9	7 5
6 8	2 8 9	4	5	6 8 9	7	3	2 8	1
5	2 7	2	3	1	4	2	4	6
2 4	1 2 4 8 9	2	7	6 8 9	2	1	6	5
3	1 5	5	1	2	6	7	9	4
7	4 8	6 9	1 9	3	5	1	6 4 8	2
1	5 7	5	4	2	9	8	3	6
2 4	2 4	8	6	7	3	5	1	9
9	6	3	8	5	1	2 4	2 4	7

如图所示，这是一个待定拓展矩形的示例，我们从 $r1c4(9)$ 开始推理。假设它为假，可以发现 $r1c4(9)=r5c4(1)$ ，否则 $r6c4$ 无法填数；接着 $r5c2(1)$ 为假。此时， $r357c23$ 中只包含 2、5、7、9 四种数字，且 9 只出现在 $r3c23$ 里。为了保证不出现致命形式，所以 $r3c23(9)$ 区块必须成立，所以 $r3c7(9)$ 为假， $r1c7(9)$ 为真。

此时，我们可以断言，头尾的交集即为删数，即 $r1c5(9)$ 。

此示例其实还有一个删数是 $r4c3(2)$ ，不过这个删数的论证较为复杂，且用到靠后的知识点，所以此处不提及。

再来看一个倒置的拓展矩形。

9	¹ _{4 8}	5	2	⁴ ₈	6	^{1 3} _{4 8}	7	^{1 3} _{4 8}
6	3	¹ _{4 7}	¹ _{4 7}	⁴ _{7 8}	9	5	2	¹ _{4 8}
2	¹ _{4 7 8}	¹ _{4 7}	¹ _{4 7}	5	3	¹ _{4 8}	6	9
¹ ₇	¹ ₆	8	9	¹ ₄	2	¹ _{4 6}	3	5
⁷ ₁	¹ ₉	2	6	3	5	7	¹ _{8 9}	¹ ₈
4	¹ ₉	2	6	3	5	7	¹ _{8 9}	¹ ₈
3	5	⁶ ₉	8	¹ _{4 7}	⁴ ₇	¹ _{4 6}	⁴ ₉	2
⁵ ₈	⁶ ₉	⁶ ₉	^{4 5} ₇	2	1	⁴ ₈	⁴ _{5 8}	³ ₈
¹ _{7 8}	¹ _{4 7}	¹ _{4 7}	3	9	^{7 8}	2	¹ _{5 8}	6
¹ _{5 8}	2	3	^{4 5}	6	⁴ ₈	9	¹ _{5 8}	7

如图所示，我们可以看到，当 r78c1(7)和 r79c8(4)同假时，四个单元格将不含有这些数，导致 r789c18 只有 1、5、8，形成拓展矩形的致命形式。所以它们形成强关系。

其它的强弱关系就不用再详细推理了。

6-2-2 待定唯一环（Almost UL）

下面来看一个有趣的示例。

1	² _{8 9}	7	² _{6 9}	⁶ ₈	4	3	5	^{8 9}
⁶ _{8 9}	⁶ _{8 9}	5	1	3	7	^{8 9}	2	4
² _{8 9}	4	3	² ₉	5	^{8 9}	7	1	6
² ₉	² ₉	^{4 6} ₈	^{4 6} ₅	1	7	3	² ₉	² ₉
3	1	^{4 6} ₈	^{4 6} ₇	² ₉	5	^{4 6} ₈	² ₉	^{8 9}
7	5	^{4 6} ₈	^{6 9}	1	3	^{4 6} ₈	2	^{8 9}
⁶ _{8 7 9}	⁶ _{7 8}	2	3	^{7 8}	1	4	9	5
5	^{7 8}	9	4	^{7 8}	6	3	1	⁸
4	3	1	5	9	6	2	8	7

如图所示，左图是 AUR，右图是 AUL。可以看到，左图里 8 和 2 形成强关系是毋庸置疑的，而右图里的 2 和 8 的强关系稍微解释一下。当它们同假的时候，{r4c35, r5c58, r6c38}六个单元格里只包含 4 和 6，而由于这六个单元格的形状在所在的 r456c358b456 里形成了 4、6 的显性数对，所以可以产生交换，而这种环状结构导致了内部一共会产生两种不同的填法，并且这两种完全不同的填法并不会影响到除了 4 和 6 的其它数字信息，所以产生了 UL 的致命形式。

为了规避这种致命形式，r5c5(2)=r6c3(8)必须成立，即它们不同假。

既然两个强关系都找到后，我们使用弱关系把 $r5c3(2)$ 连起来，于是成了下面的这种特殊的不连续环结构：

1	² 8 9	7	² 6 9	⁶ 8 9	4	3	5	⁸ 9
⁶ 8 9	⁶ 8 9	5	1	3	7	⁸ 9	2	4
² 8 9	4	3	² 9	5	⁸ 9	7	1	6
² 9	² 9	⁴ 6	8	⁴ 6	5	1	7	3
3	1	⁴ 6	7	² 6	² 9	5	⁴ 6	⁸ 9
7	5	⁴ 6	⁸ 9	1	3	⁸ 9	⁴ 6	2
⁶ 8	⁷ 8	⁶ 8	2	3	⁷ 8	1	4	9
5	⁷ 8	9	4	² 7 8	² 8	6	3	1
4	3	1	5	9	6	2	8	7

这是一个长度为 3 的链结构，而其中的两个强关系一个诞生于 AUR，一个诞生于 AUL。最后由于链的首尾是同一个节点的关系，导致了这个节点必须为真的结果。所以这个链的结论是 $r5c5 = 2$ 。

6-3 待定可规避矩形 (Almost AR)

待定可规避矩形 (Almost AR, 简称 AAR) 是 AR 的变体，因为 AR 已经比较难观察到了，所以 AAR 更加难观察，不过例子也是非常有趣的，下面罗列出两则关于 AAR 的强弱关系的运用的示例。

3	⁵ 8	6	7	9	² 5	1	² 8	4
⁴ 8	9	⁴ 5	6	1	² 5	² 3	7	³ 8
2	1	7	4	8	3	6	9	5
⁴ 8	⁴ 8	1	3	6	9	7	5	2
5	3	9	2	7	8	4	1	6
7	6	2	1	5	4	³ 8 9	³ 8	³ 8 9
9	2	3	8	4	7	5	6	1
6	7	⁴ 5	⁵ 9	3	1	² 8 9	² 4	⁸ 9
1	⁴ 5	8	⁵ 9	2	6	³ 9	⁴ 3	7

如图所示，这条链写法如下：

$$r1c6(5)=r1c8-r2c9(8=3) \Rightarrow r2c6 \neq 5$$

这条链其实很简单，虽然首尾不同，但我们之前说过，有一种链的删数是需要额外添加弱关系的（在前文里称之为间接删数的链），这条链其实就是这样。

我们设 $r1c6(5)$ 为假时，就会得到 $r2c9(3)$ 为真。很明显，此时 $r1c6(5)$ 是可以删除 $r2c6(5)$ 的，而 $r2c9(3)$ 其实也是可以删除 $r2c6(5)$ 的。很明显，它们并不能同真，否则 $r23c69$ 就会出现关于 3 和 5 的 AR 致命形式。所以 $r2c6 \neq 5$ 必然是成立的。

3	1	² ₇ 6	8	⁴ ₇	² ₉ 5	⁴ ₇ 5 6	⁶ ₉ 4	
5	⁴ ₇	² ₉ 7 8 9	6	⁴ ₇	² ₉ 1 3	¹ ₄ 3	⁶ ₉ 4	³ ₈ 9
	⁴ ₇ 6	⁶ ₉ 7 8 9	³ ₇	1	⁵ ₉ 4 5 6	2	⁴ ₈ 9	³ ₇
¹ ₇	¹ ₉	¹ ₇	2	³ ₆	⁶ ₈	³ ₁ 4	5	
2	3	⁵ ₇ 4	⁴ ₁ 9	¹ ₈	¹ ₈	⁷ ₉	6	
6	8	⁴ ₁	⁵ ₃	7	¹ ₉ 3	¹ ₉	² ₃	
¹ ₇	2	¹ ₇ 3	⁵ ₇ 8	¹ ₆	⁴ ₉ 6	⁶ ₉	⁴ ₉	³ ₇
⁴ ₁	⁵ ₆	⁶ ₉ 7 9	² ₇	³ ₆	3	² ₃ 8	1	
¹ ₈	⁶ ₉	¹ ₈ 3	¹ ₉	² ₄	³ ₆	5	7	

如图所示，链的写法如下：

$$r4c7=r4c5(3-6)=r8c5-r9c7(6=3)-r4c7(3) \Rightarrow r267c7 \neq 3$$

我们来思考一下这个链的 $r4c5(6)-r8c5(6)$ 。如果 $r4c5(6)$ 和 $r8c5(6)$ 同真时，我们就会发现 $r89c57$ 四格形成关于 2 和 6 的 AR 致命形式。所以，它们不可同真。

6-4 BUG + 2 在链里的运用

接下来我们来理解 BUG + 2 这个技巧，在链里的使用。

BUG + 2 在之前我们学到了直接删数的方式，是通过找交集的方式来得到删数的，但总有些时候，BUG + 2 不一定能够一样的交集，那么此时我们就不得不需要依赖于链了。

6	9	2	^{4 5}	3	1	^{4 8}	7	^{5 8}
1	3	8	2	^{4 7}	^{5 7}	9	^{4 5}	6
5	4	7	9	6	8	2	1	3
3	^{5 8}	6	^{4 5}	1	9	^{4 7}	^{2 4}	^{2 7}
4	7	1	8	2	3	5	6	9
2	^{5 8}	9	6	^{4 7}	^{5 7}	3	^{4 8}	1
9	^{2 6}	5	7	8	^{2 6}	1	3	4
8	^{2 6}	3	1	9	4	^{2 6}	^{2 5}	^{2 7}
7	1	4	3	5	^{2 6}	^{6 8}	9	^{2 8}

如图所示，由于 BUG + 2 的特殊性，所有的两个真数必须不同假，否则同时消失会导致 BUG 出现致死形式。所以它们可以形成强关系。

接下来我们使用链，直到这条链一直延伸到 **r4c8(2)**。此时发现链头和链尾处于同一个单元格，此时的删数就应该是 **r4c8** 的其余候选数，所以 **r4c8 <> 8**。

我们再来看一个示例。

^{3 8}	^{2 6}		1	^{2 6}	^{5 3}	4	^{7 9}	^{5 8}
^{4 3}	5	^{4 6}		8	^{3 6}	1	^{7 9}	2
	^{2 7}	1	^{5 7}		4	^{5 6}	3	^{6 8}
1	8	^{5 6}	3	4	^{5 6}	^{5 7}	2	9
^{4 9}	^{4 9}	3	2	1	^{5 7}	8	6	^{5 7}
2	^{7 6}	^{7 5 6}	^{5 9}		^{6 9}	8	3	1
5	1	2	8	3	9	^{7 6}	4	^{7 6}
7	^{4 9}	^{4 9}	6	5	1	2	8	3
6	3	8	4	7	2	9	5	1

如图所示，当 $r3c4(9)$ 和 $r4c6(5)$ 同假时，将导致 BUG 出现致死形式，所以形成强关系是成立的。于是得到了这条链，删数是 $r2c4(9)$ 。

不过这个例子我们可以切换一下视角，使得它的删数变为 $r5c6(5)$ ，如图所示。

	3	2		1	2	3	4		5
8		6	7	9		5	7	9	8
4	3	5	4	6		8	1		2
	2	1	5	7	9	4	5	6	3
8	9	7	1	5	7	9	4	5	6
1	8	5	6	3	4	5	6	2	9
4	9	4	9	3	2	1	5	8	6
2		6	5	6	5	6	8	3	1
5	1	2	8	3	9	7	6	4	7
7	4	9	4	9	6	5	1	2	8
6	3	8	4	7	2	9	5	1	

逻辑推理的过程和刚才的是完全一样的。

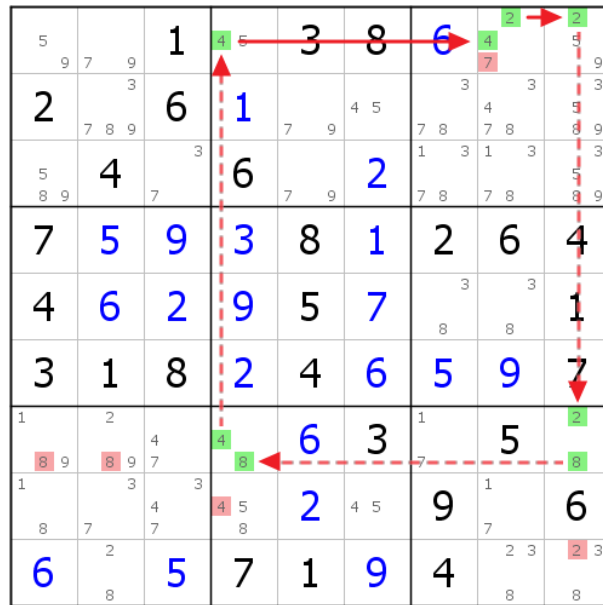
实际上，BUG + 3、BUG + 4 等也都能使用链来删数，不过因为例子比较复杂，而且需要用到新的技巧，所以此处就不再过多提起，到后续的内容里我们会提到。

Part 7 环 (Continuous Nice Loop)

环在链里的应用非常广泛,由于它的结构的特殊性,强弱关系能够首尾拼接形成环结构,甚至完全找不到开头。但正是因为这个特征,导致环的删数非常多,因而受到众多难题玩家的青睐。

7-1 标准环 (Continuous Nice Loop)

在学习技巧之前，我们先来看看环的基本形态。



如图所示。链的写法如下:

$$r1c4(4)=r1c8(4-2)=r1c9(2)-r7c9(2=8)-r7c4(8=4)-r1c4(4)$$

发现了两点神奇的现象：**r1c4(4)**首尾成环了，而且强弱关系是交替的，也就意味着，就图上的情况而言，随便找个蓝色节点都有可能成为开头。于是我们随便选了个节点，比如**r1c4(4)**作为了开头，然后写成了环形。

还有一点是删数，所有红色的都是删数，但这一次，红色的删数比起之前用过的链技巧而言，删数多太多了。那么如何推导呢？反正都是个环结构，随便找个弱关系，把它拆掉。比如我们拆掉最下方的 $r7c4-r7c9(8)$ 的弱关系。然后链变成这样：

5	9	7	9	1	4	5	3	8	6	4	2	2
2			3	6	1		4	5		3	7	3
	7	8	9			7	9		7	8	7	8
5		4		3	6			2	1	3	1	3
	5	8	9	7		7	9		7	8	7	8
7	5	9		3	8	1	2	6	4			
4	6	2		9	5	7		3		3		1
							8	8				
3	1	8		2	4	6	5	9	7			
1		2	4		4		6	3	1		5	2
8	9	8	9	7	8				7			8
1			3		4	5	2	4	5	9	1	6
	8	7	7		4	5				7		
6		2		5	7	1	9	4		2	3	2
		8								8		8

这样就会发现，它就是一个 AIC 了。那么删数就是两个 8 共同对应的区域（r7）的其余 8 了。那我再留下这个弱关系，去断开剩下的随便一个弱关系，也可以得到对应的删数。所以所有删数如最开始的盘面所示。

这种结构特殊之处就在于，链的首尾是完全拼接在一起的，而且链头还不好去找到唯一的那一个位置，这就区别于之前讲到的不连续环了。不连续环虽然名叫不连续环，是有环这个字的，但是我们依然知道它是一种链结构，并且有头有尾，这一点就和现在讲到的这个东西不同了。

那么，这样的特别结构，叫做**标准环**（或**连续环**，简称**环**，**Continuous Nice Loop**）。

最后需要引起注意的是，弱关系也包括单元格内的弱关系，比如图中的 r1c8(2-4) 时候，可以删掉 r1c8(7)，至于原因，把链从 r1c8(2-4) 这里拆开，得到了一条链头为 r1c8(4)，链尾为 r1c8(2) 的 AIC，它们的交集，因为是异数的，而且同一单元格，所以交集只能是 r1c8 内的其余候选数。所以这一点千万别忘记了。

所以说，环有哪些性质可以直接使用呢？

7-2 环的基本形式和特征

标准环具有如下的特征（请最好背下它们）：

- 任意弱关系都能直接拆掉，并得到一条链，从而获得当前的删数效果；
- 任意节点都可以作为整条链的链头；
- 环内的填数情况一共只有两种；
- 环的长度一定为偶数；
- 环里面的所有的相邻节点既可以看作强关系，也可以看作弱关系。

那么，这五点我将一一论证。

第一点，这是环的删数原则和基本思想。所以这一点我就不用说明了（就是拆成链结构，

然后找删数)。

第二点，例如下面两个举例的节点：**r1c4(4)**作为开头和**r7c9(8)**作为开头。

5 9	7 9	1	4 5	3	8	6	4 2	2	5 9
2	7 8 9	6	1	7 9	4 5	7 8	4 7 8	5 8 9	3
5 8 9	4	7	6	7 9	2	7 8	7 8	5 8 9	3
7	5	9	3	8	1	2	6	4	7
4	6	2	9	5	7	8	8	1	3
3	1	8	2	4	6	5	9	7	3
1	2	4	4 5	6	3	1	5	2	7
1	8 9	3	4 5	2	4 5	9	1	6	8
6	2	5	7	1	9	4	2 3	2 3	8

5 9	7 9	1	4 5	3	8	6	4 2	2	5 9
2	7 8 9	6	1	7 9	4 5	7 8	4 7 8	5 8 9	3
5 8 9	4	7	6	7 9	2	7 8	7 8	5 8 9	3
7	5	9	3	8	1	2	6	4	7
4	6	2	9	5	7	8	8	1	3
3	1	8	2	4	6	5	9	7	3
1	2	4	4 5	6	3	1	5	2	7
1	8 9	3	4 5	2	4 5	9	1	6	8
6	2	5	7	1	9	4	2 3	2 3	8

可以从例子里看到，实际上，除了换了链的方向以外，链没有发生任何变动。由于链必须以强关系开头的关系，使得有些以弱关系起头的节点不得不反向绘制链。不过反向绘制并不会影响到什么，因为强弱关系是可以反向的，而且反向后，弱关系开头就变为了强关系开头了，这是因为以弱关系开头的节点和它后一个节点连接；而在反向后，就会和它前面一个节点连接，而它前一个节点和它本身是形成强关系的。

第三点，我们在推理的过程都是最简单的“不填 -> 填 -> 不填 -> 填 -> ……”，这样的周而复始的循环。而环结构之中，由于推理最终回到最初的位置，并且仍旧能够按照原来假设所定的填数状态继续重新推理，所以这一定是其中的一种情况，而根据性质 2，我们知道，任何节点都能做起始点，也就是说，我们不妨将节点切换到与其相邻（它的上一个候选数填数状态或者其下一个候选数填数状态）的位置上去，让它作为链头，这个时候填数状态就能全部交换了。但是，由于结构内就两种填数情况，所以这就说明了，环结构内一定只涉及两种填数。

第四点，强弱关系数量一样，强关系有 n 个，弱关系就有 n 个，那长度自然是 $2n$ 个， $2n$ 是一个偶数。

第五点，根据性质 3，我们可以直到环内仅存在两种填数情况，而两种填数情况显然只可能使得任意相邻的两个节点一真一假，而一真一假的情况是既满足强关系的定义（不同假），也满足弱关系的定义的（不同真）。有些时候，第五点也被说成“**所有强关系都能转为弱关系理解；而所有弱关系都能转为强关系理解**”，因为一真一假的逻辑恰好是同时满足强关系和弱关系的定义的。

不过之前说的不连续环并不满足这些特性，只是一种结构成环而逻辑无环的特别情况罢了。实际上，不连续环里的“环”字的意义和我们此处介绍的环的意义不同。所以，不连续环和连续环实际上就差在了“不”这个字上：不连续环是无法连续推导的，因为它始终是符合链的规则，有头有尾，找交集删数；而连续环，任意节点为头都可以。

7-3 常见技巧的环视角

下面来介绍一些常见技巧的环视角。

7-3-1 标准链列

9	4	3	2	3	2	3	3	3	5	6	1
2	8	4	5	6	4	5	1	4	7	9	3
7	1	5	3	5	9	6	2	4	8		
6	5	3	5	3	2	1	4	7	9		
1	2	3	4	3	6	3	1	3	2	3	5
8	7	8	9	8	7	8	9	8	7	9	8
1	2	3	5	5	4	5	1	3	2	3	6
8	7	8	9	8	7	8	9	8	7	9	8
4	2		9		5	8	6	1	7		
5	3	6	7	1	3	2	3	3	4		
8	9						8	9			
3	3	1	3	3	3	3	3	3	5	2	
8	8	9	4	6	7	9	7	9	8	9	5

9	4	3	2	3	2	3	3	3	5	6	1
2	8	4	5	6	4	5	1	4	7	9	3
7	1	5	3	5	9	6	2	4	8		
6	5	3	5	3	2	1	4	7	9		
1	2	3	4	3	6	3	1	3	2	3	5
8	7	8	9	8	7	8	9	8	7	9	8
1	2	3	5	5	4	5	1	3	2	3	6
8	7	8	9	8	7	8	9	8	7	9	8
4	2		9		5	8	6	1	7		
5	3	6	7	1	3	2	3	3	4		
8	9						8	9			
3	3	1	3	3	3	3	3	3	5	2	
8	8	9	4	6	7	9	7	9	8	9	5

如左图所示，这是一个 X-Wing（二链列）技巧，它的定义域是 c34。我们可以尝试把定义域上的两个单元格视为共轭对，并构成强关系，然后把删除域上同列的两个候选数作为弱关系连起来，最终构成右图的环，这种环的删数和原结构的删数完全一样。而理解方式，只需要变化一点，因为 X-Wing 结构的形式导致了共轭对一定会出现两处，便可立马绘制出环结构。

实际上，只要我们能在这鱼的所有定义域都构成共轭对的形式，那么结构就一定是一个环，比如下面的这个三链列的例子，它的所有定义域区域都会产生共轭对。

1	6	2	5	4	3	2	7	2
2	7	8	6	1	4	3	5	
4	3	5	8	7	6	1		
7	2	1	4	5	8	6	9	
6	4	4	9	1	2	5	7	
5	8	5	3	7	6		4	
2	1	6	3	5	4	8		
3	4	5	8	1	6			
2		7	1	6	4	5	3	

1	6	2	5	4	3	2	7	2
2	7	8	6	1	4	3	5	
4	3	5	8	7	6	1		
7	2	1	4	5	8	6	9	
6	4	4	9	1	2	5	7	
5	8	5	3	7	6		4	
2	1	6	3	5	4	8		
3	4	5	8	1	6			
2		7	1	6	4	5	3	

如图所示，这就是一个三链列，它的环的画法。

7-3-2 显隐性数组

数组之所以也能删除很多数字，是因为它实际上也是一个环。

9	8	^{1 2} ₄	¹ ₄	^{1 2}	6	3	7	5	9	8	^{1 2} ₄	¹ ₄	^{1 2}	6	3	7	5
3	7	6	8	5	² ₉	1	4	² ₉	3	7	6	8	5	² ₉	1	4	² ₉
^{1 2} ₄	¹ _{4 5}	^{1 2} _{4 5}	7	^{1 2 3} ₉	^{2 3} _{4 9}	8	6	² ₉	^{1 2} ₄	¹ _{4 5}	^{1 2} _{4 5}	7	^{1 2 3} ₉	^{2 3} _{4 9}	8	6	² ₉
5	6	9	3	4	7	2	1	8	5	6	9	3	4	7	2	1	8
¹ _{4 8}	¹ ₄	¹ _{4 8}	⁶ ₉	² ₉	² ₉	5	3	7	¹ _{4 8}	¹ ₄	¹ _{4 8}	⁶ ₉	² ₉	² ₉	5	3	7
7	2	3	5	8	1	4	9	6	7	2	3	5	8	1	4	9	6
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
^{1 2} ₄	¹ ₄	¹ ₄	^{1 2} ₄	¹ ₄	¹ ₄	3	9	¹ ₄	^{1 2} ₄	¹ ₄	¹ ₄	^{1 2} ₄	¹ ₄	¹ ₄	3	9	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄	¹ ₄	¹ ₄	3	¹ ₄	2	¹ ₄	3	5	¹ ₄
¹ ₄	¹ ₄	3	¹														

如图所示，这是 1、4 显性数对的环视角。可以发现，由于显性数对是双值格的关系，单元格之间的候选数的关系变为弱关系，而单元格内的候选数关系即为强关系。

同样地，三数组也是如此。只要三数组的三个单元格都是双值格，那么就一定可以画成环，如图所示。

1	6	2	3	5	8	7	9	4	1	6	2	3	5	8	7	9	4
9	8	5	4	6	7	2	1	3	9	8	5	4	6	7	2	1	3
7	3	4	¹ ₉	¹ ₂	⁵ ₈	⁵ ₈	⁶ ₉		7	3	4	¹ ₉	¹ ₂	⁵ ₈	⁵ ₈	⁶ ₉	
8	7	6	² ₉	4	5	1	3	² ₉	8	7	6	² ₉	4	5	1	3	² ₉
² ₄	5	1	8	³ ₉	² ₆	⁴ ₉	⁶ ₇	² ₉	² ₄	5	1	8	³ ₉	² ₆	⁴ ₉	⁶ ₇	² ₉
² ₄	9	3	7	1	² ₆	⁵ ₆	⁴ ₅	8	² ₄	9	3	7	1	² ₆	⁵ ₆	⁴ ₅	8
3	¹ ₂		5	8	¹ ₂	4	6	¹ ₇	3	¹ ₂		5	8	¹ ₂	4	6	¹ ₇
6	¹ ₄	¹ ₂	⁷ ₉	¹ ₂	³ ₄	¹ ₂	³ ₄	5	6	¹ ₄	¹ ₂	⁷ ₉	¹ ₂	³ ₄	¹ ₂	³ ₄	5
5	¹ ₄	8	6	7	¹ ₄	3	2	¹ ₉	5	¹ ₄	8	6	7	¹ ₄	3	2	¹ ₉

除了显性数组外，还有隐性数组结构也可以这么干，不过在之前我们说到，隐性数对实际上是两个数形成了共轭对，只是恰好位于同样两个单元格里，所以我们此处就会使用到这个结论。

3	9	4	6	1 2	1 2	1 2	7	1 2	1 2	3	9	4	6	1 2	1 2	1 2	7	1 2	1 2
1 2	1 2	4	8	1 2	1 2	1 2	6	5	1 2	1 2	1 2	1 2	4	8	1 2	1 2	6	5	1 2
5	1 2	6	7	1 2	1 2	1 2	3	4	9	5	1 2	6	7	1 2	1 2	1 2	3	4	9
2	4	9	3	8	1 2	1 2	5	1 2	6	2	4	9	3	8	1 2	1 2	5	1 2	6
6	2	1	2	5	4	9	9	8	3	6	2	1	2	5	4	9	9	8	3
8	5	3	1 2	1 2	1 2	6	4	1 2	1 2	8	5	3	1 2	1 2	1 2	6	4	1 2	1 2
9	6	5	6	8	2	6	1	3	4	9	6	5	6	8	2	6	1	3	4
1	1	3	2	9	4	1	8	6	5	1	1	3	2	9	4	1	8	6	5
4	1	3	5	6	1	3	2	9	7	4	1	3	5	6	1	3	2	9	7

我们将其中的共轭对视为强关系，而单元格内作为弱关系，就形成了环，并得到删数。

所以我们可以看到，实际上可以删不同位置的数值的情况，一般都是使用环来解决的，只是我们平时很少，甚至不会使用环来作图。

7-3-3 欠一数对

还记得欠一数对吗？欠一数对我们通过了例子来解释，而且还通过了直观层面，假设为 x 和 y 的方式来得到数对形式。不过实际上，正是因为它能通过 x 和 y 的方式来假设，才有了它的环的视角。

4	2	2	6	6	3	5	1	7	4	2	2	6	6	3	5	1	7
1	3	5	4	2	7	8	6	9	1	3	5	4	2	7	8	6	9
8	6	7	9	1	5	4	2	3	2	3	8	6	7	9	1	5	4
6	9	3	5	4	8	2	7	1	6	9	3	5	4	8	2	7	1
7	1	8	3	6	2	6	5	4	7	1	8	3	6	2	6	5	4
2	5	4	1	6	1	6	2	3	3	2	5	4	1	6	1	6	2
3	2	1	2	3	4	7	2	3	5	3	2	1	2	3	4	7	2
5	2	1	2	1	2	1	2	3	4	5	2	1	2	1	2	1	2
4	3	2	1	2	6	4	3	9	4	3	2	1	2	6	4	3	9

如图所示，右图就是左图的环的画法。可以看到，这个例子带了一个区块，而这个区块还是很简单的结构，所以不用过多考虑删数的问题。注意 $r7c3$ 单元格内的弱关系，也是要删数的。

7-3-4 双 RCC 的 ALS-XZ

实际上，这个古怪的玩意儿就是一个环。不过这个环的逻辑在之前已经超纲地提到了（因为它还带有两个 ALS 区域，还要考虑 ALS 区域的删数，所以在前面提到算是“超纲”了），所以我们就不用过多去啰嗦它的基本逻辑。下面的一些复杂的环结构，它们的删数将涉及到这些内容，所以具体的原理我们在下方会提到。

7-4 嵌套结构的环 (Grouped Continuous Nice Loop)

7-4-1 ALS 环 (Grouped Continuous Nice Loop With ALS)

3	4 5 6 7 9	2	6 8	5 6 8	4 5 6 7 8	4 5 7 8	1	4 5 6 8 9
1	4 5 6 8	9	7	4 5 6 8	4 5 8	4 5 6 8	4 5 6 8	4 5 6 8
4 6 7 8	4 5 6 7 9	4 6 7 8 9	2 3 6	1	2 3 4 5 6 8	2 3 4 5 7 8	3 4 5 6 8 9	2 3 4 5 6 8 9
5	2 3 7 9	4 7 9	1	8	2 3 9	6	4 9	2 3 9
2 4 6	2 3 4 6 9	1	5	2 3 6 9	2 3 6 4	2 3 8	7	2 3 4 8 9
2 6	8	3 6 9	7	4	2 3 6 9	1	5 9	2 3 5 9
9	4 6 8	3 4 6 8	3 6	5 6 8	1	4 5 8	2	7
2 7 8	1	3 7 8	4	2 3 5 6 9	2 3 5 6 8 9	3 5 6 8	3 5 6 8	3 5 6 8
4 6 8	2 3 4 6	5	8	6	7	9	4 6 8	1

如图所示，链写法如下：

环：r3c4=r9c4-r9c5(2)=r79c5-r79c4(3)=r3c4(3-2)
=> r3c4 <> 68, r8c56 <> 23, r9c12 <> 2, r58c5 <> 56

由于环可以任意端点起头，对应也可以任意端点结尾，所以我们的最后使用弱关系来表达它产生了循环：把最后一个节点写到链头上去。

我们按照最初的环的视角，来理解一下删数到底是怎么产生的。

首先是 r3c4 <> 68。这一点很清晰，因为 r3c4(2)和 r3c4(3)是弱关系，环结构的其中一个特性是环内的所有弱关系共同对应的位置（交集）都可以直接删除，所以 r3c4 内其余候选数都可删除；

其次是 r8c56 <> 23。这一点是因为 r8c56(23)是分别处于 r9c4(2)和 r9c5(2)的弱关系所处区域和 r79c5(3)和 r79c4(3)的弱关系所处区域。所以 b8 和 r9 上其余位置的候选数 2 都可删除，而 b8 内其余位置的候选数 3 都可删除。当然了，r9c12(2)也就因为刚才说的原因被删掉。

那么，r58c5(56)呢？这个环根本就没涉及 5 和 6，全是 2 和 3。如果你这么想就错了。我们尝试观察 ALS 区域 (r179c5)。之前的环结构内有一个特性：环内只有两种填数

情况。这意味着每两个相邻节点的填数情况真假性一定是相反的。也就是说，**r9c5(2)**和**r79c5(3)**两个节点真假性相反，即：

- **r9c5(2)**为真，**r79c5(3)**为假；
- **r9c5(2)**为假，**r79c5(3)**为真。

只有这样两种情况。那么，如果 **r9c5(2)**为真的话，ALS 区域内必然有一个 5、6 显性数对，出现在 **r17c5**；如果是 **r79c5(3)**为真的话，由于这里是同一区域内的多个候选数 3 为真，所以不管是 **r7c5** 还是 **r9c5** 的候选数 3 为真，剩下另外一格都不应有候选数 3（要保证只能有一个 3 为真，否则违反数独规则），然后和 **r1c5** 构成 5、6 的显性数对。

所以不管怎么样，都会构成 5、6 显性数对，所以 **c5** 内其余单元格都不应有 5 和 6，删除掉它们。这也就是为什么最后还有一个 **r58c5 <> 56** 的原因。

那么，这种环结构嵌入了 ALS，所以我们一般称这种结构就叫**带 ALS 的环（Grouped Continuous Nice Loop With ALS）**。接下来我再举一个例，请自行思考它的所有删数成立的原因。

3	4 5 6	2	6	5 6	4 5 6	4 5	1	4 5 6
1	4 5 6	4	6	9	7	4 5 6	4 5 6	4 5 6
4	6	4 5 6	4	6	1	4 5 6	4 5 6	4 5 6
5	2 3	4	3	1	8	2 3	6	3
2	2 3	1	5	2 3	2 3	2 3	7	2 3
2	6	8	3	7	4	2 3	1	3
9	4	3	3	1	1	4 5	2	7
2	6	1	3	4	2 3	3	3	3
4	2	2 3	5	2 3	7	9	4	1

7-4-2 毛刺数组环 (Grouped Continuous Nice Loop With Burred Subset)

我们再来看带有一个毛刺数组的环结构。

9	4 5	4 5	1 3	6	1 4 5	4 7 8	2	1
	3	2 3	1 2 3		1 2	4 5	1	1
8	4 5	4 5 6	7	4 5	4 5	4 6	4 5	4
2	7	7 8	7	7	7 8 9	7 8 9	7 8	7 8 9
5 6	2	4 5 6	1	8	9	3	4 5	4
7	7	7	7	7	7	7	7	7
	3	3	1 2	6	5	4	1	2
8	7 9	7 8 9	7	7 9	2 3	2	4	4
4	1	5	8	5	5	7	9	6
5 6	5 6	2	4	5	1		1 3	1 3
	9			9 7	7	7 8	7 8	7 8
	6	4 6	3	5	2	4	1	4 6
7	7 9	7 9			7 8	7 8	7 8 9	7 8 9
1	2	4 5 6		6 4	4	4 6	4	2 3
	7 9	9		9 7	7 8	7 8 9	7 8	4
2					3	2	3	7 8 9
7	8	4 6		6 9	4	4 6	4	5
	9	9		9	7	9 7	9 7	

如图所示，链如下所示：

环： $r4c9=r4c4-r3c4(2)=r3c9-r1c7(4)=r16c7(78)-r5c7(7=2)-r4c9(2)$
 $\Rightarrow r2c4 \times 2, r1c9, r2c789, r3c8 \times 4, r3c128 \times 7, r289c7 \times 78$

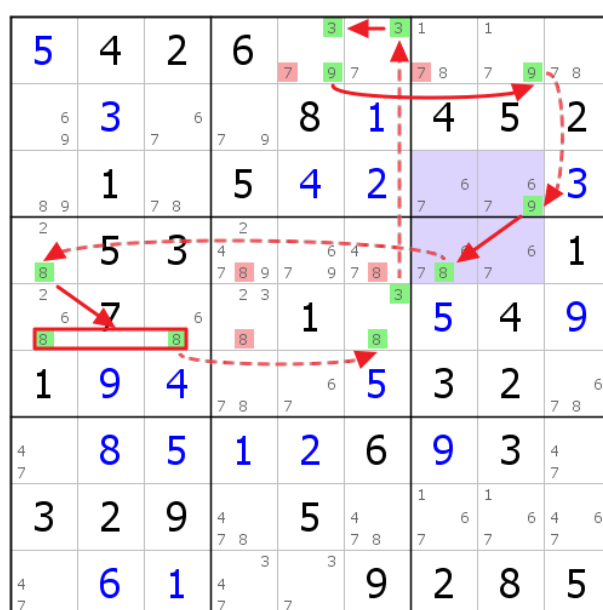
首先，我们能找到所有弱关系的删数： $r2c4(2)$ 、 $\{r1c9, r2c789, r3c8\}(4)$ ；然后是 $r3$ 的 ALS 的删数： $r3c128 \times 7$ 。接着我们来看着重观察这个毛刺显性数对。

当 $r1c7 \times 4$ 时， $r16c7$ 形成 7、8 数对结构，所以 $r5c7 \times 7$ ，链得以继续下去。那么删数怎么看呢？实际上，我们就把它当成真正的数对来删数就可以了。为什么呢？我们只要保证环的两种填法下，都能产生 7、8 数对就可以。

当我们从示例这个环的方向来看，显然是构成数对的，所以这个情况成立；而另外一种填法正好就可以反向来理解。当我们反向理解的时候，假设从 $r5c7(2)$ 开始，设为假。那么 $r5c7(7)$ 为真，此时 $r16c7(78)$ 的毛刺显性数对结构已经被破坏，所以毛刺显性数对节点此时是为假的，但是，数字 8 不得不填在 $r6c7$ 里，因为 $r6c7$ 是双值格，而此时 $r5c7 = 7$ ，所以 $r6c7$ 不得不填 8。这样一来， $c7$ 就依然存在 7 和 8，所以其余位置照样是可以删除 7 和 8 的。

看起来这里只是巧合，但实际上并不是。因为要构成毛刺显性数对，就需要一个单元格是恰好双值格的形式存在（比如 $\{ab\}$ ），或者不是双值格，也要保证两个单元格有额外的数字要一致（比如是 $\{abcd\}$ 或 $\{abd\}$ ），这样才能通过强弱关系来延续。如果一个单元格是 $\{abc\}$ ，而另外一个单元格却是 $\{abd\}$ 的话，这样我们并不能够得到任何合适的结论。特别要引起注意的是，此时的 $\{abc\}$ 和 $\{abd\}$ 的 c 和 d 并非强关系，因为它不是 ALS。

7-4-3 AUR 环 (Grouped Continuous Nice Loop With AUR)



如图所示，这个环稍微有一些复杂，链写法如下：

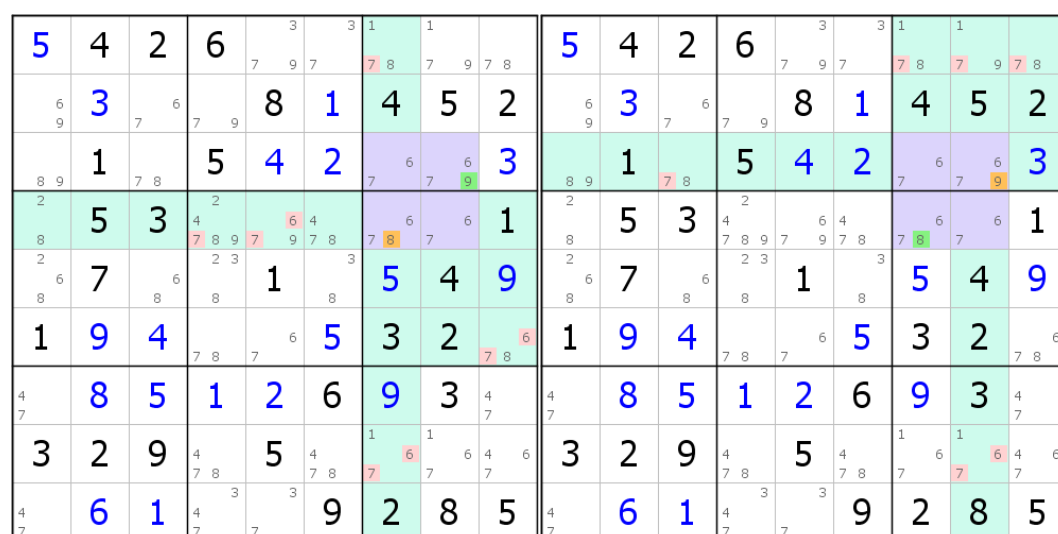
$$\begin{aligned} r4c1=r5c1-r5c6(8=3)-r1c6=r1c5(3-9)=r1c8-r3c8(9)=r4c7-r4c1(8) \\ \Rightarrow r1c57 \times 7, r4c46, r5c4 \times 8 \end{aligned}$$

这些基本的删数，相信我也不需要过多去介绍和解释了。还是环的第一个特性：所有弱关系对应可删（拆成链后删除）。

观察这个 AUR（我们单独看这个 AUR 结构），因为之前说到，相邻两个节点的真假性不同，所以 $r4c7(8)$ 和 $r3c8(9)$ 自然真假性就不一样了。那么就有这样两种情况：

- $r3c8(9)$ 真， $r4c7(8)$ 假；
- $r3c8(9)$ 假， $r4c7(8)$ 真。

因为 AUR 归根结底就是避免致命形式，而内部一定是由产生的数对导致的，我们从这一点来思考就简单一些了。这样一来，我们就分成两个情况作图给大家看。如图所示。



- 如果是左图这种情况 (**r3c8(9)真**)，则 **r4c7(8)**自然应该为假，此时 **r34c7** 和 **r4c78** 都应该是 67 显性数对，可以删除的部分用浅红色标注了出来；
- 如果是右图这种情况 (**r4c7(8)真**)，则 **r3c8(9)**自然就应该为假了，此时 **r3c78** 和 **r34c8** 就会构成 67 显性数对，可以删除的部分用浅红色标注。

我们就发现了一个地方，是两种情况都可以删掉的：

5	4	2	6		3		3	1	1		
	6	3			7	9	7	7	8	7	9
	8	9	1		7	8		4	5	2	
	8	9	5	4	2			6	7	6	3
2	8	5	3		2			6	7	6	1
2	6	7			2	3		7	8	7	
1	9	4			6	5		3	2		6
4	7	8	5	1	2	6		9	3	4	
3	2	9			5			1	6	1	6
4	7	6	1		3			2	8	5	

所以，**r1c7 <> 7** 就是 AUR 在嵌入环里面的特别删数。这个结构叫做**带 AUR 的环** (Grouped Continuous Nice Loop With AUR)。

7-4-4 AUR 环的另一则示例

这个例子的环的长度只有 4，但依然能让我们学习。

2	2	1		2	7		2	3			
4	6	4	6	1	4	5	8	9	5	6	
8	8	3		2	1	2	2	1	2	5	6
2	6	7	8	9	8	9	8	9	8	9	8
7	8	5	4	2	1	3	2	3	1	2	1
5	4	8	9	8	6	4	7	1	2	8	9
1	6	7	8	9	2	3	1	3	3	1	3
4	8	3		4	6	4	4	6	9	7	9
9	4	2	4	5	3	1	2	3	6	8	9
7	8	5		3	1	4	6	4	9	9	7
2	2	5		3	1	2	3	3	3	1	2
4	6	9		4	6	4	5	4	5	6	4
1	2	3		7	8	9	7	8	9	7	8
3	1	2		9	2	2	6	1	2	1	2
7	4	7	8	9	4	5	7	8	9	7	8

如图所示，这个环我们假定从 **r4c7(1)**开始，设该节点为假，则 **r6c7(8)**必须为真，否则同假会导致 **c7** 放 1 和 8 的位置只有 **r2c7** 一个单元格，导致矛盾；接着，**r6c7(8)**为

真时，则必须 $r2c9(6)$ 为假。否则同真将导致 $r6c9$ 无法填数。再接着，当 $r2c9(6)$ 为假后，则必须 $r4c1(1)$ 为真。因为当 $r2c9(6)$ 为假时， $r2$ 出现 6、7 的隐性数对，位于 $r2c13$ ，如果此时 $r4c1(1)$ 为假的话，则 $r4c13$ 变为 6、7 显性数对，此时 $r24c13$ 均只含有 6 和 7，形成致命形式。所以 $r4c1(1)$ 必须为真。

链的推导就到这里，那么删数比较好理解的是 $r4c28(1)$ ，不过其它的呢？首先我们来分析 $r2c7(5)$ 是怎么删除的。实际上，不论 $r4c7(1)$ 成立也好， $r6c7(8)$ 成立也好，由于 $c7$ 只有三个单元格可以放下 1 和 8，而此时如果 $r2c7 = 5$ ，就会让一个和 1、8 不相关的数字占了其中一个单元格，导致剩下两个单元格里无法正常放下 1 和 8（环内的相邻两个节点的真假性相反，即必须一真一假）。所以 $r2c7 \neq 5$ 。

接着我们再来看 $r6c23(8)$ 和 $r5c9(8)$ 。我们可以发现的是，这三个候选数处于 $r6c7(8)$ 节点 $r2c9(6)$ 节点的所在行列上，我们讨论它们俩即可。当 $r6c7(8)$ 为真时，显然可以删除这三个候选数；而当 $r2c9(6)$ 为真时，虽然不明显，但是可以发现，我们用来推矛盾的 $r6c9$ 派上了用场：此时 $r6c9$ 只能放下 8（双值格的关系，其中的 6 被设定的 $r2c9(6)$ 排除了）。所以此时这三个数依然是可以被删除的，所以这三个候选数依旧可以被删掉。

所以这个链的总的删数一共就有这么多。可以发现，链很短，但需要分析很长的时间。

7-4-5 待定拓展矩形环 (Grouped Continuous Nice Loop With XR)

2 6 8	3	1	2 4 6 8	2 4 6 8	2 4 6 8	7 5		
6 8	2 4 8 9	4	5 3	7 3	2 4 6 8	3 1	2 4 6 8	1
5	2 4 7 9	2 4 7 9	3 1	2 4 6 8	2 4 6 8	6 4	8	
2 4 8	1 2 4 8 9	2 4 6 8 9	7 3	2 4 6 8	1 6 8	5 3		
3	1 5	2 5	1 2	4 8	6 7	9 4	8	
7	4 8	6 9	1 9	3 5	1 6 4 8	2		
1	5 7	5 7	4 2	9 8	3 6			
2 4	2 4	8	6 7	3 5	1 9			
9	6	3	8	5	1	2 4	2 4	7

如图所示，这个环的逻辑比较简单，所以就不写文本写法了，不过我们来看看这个待定拓展矩形的删数究竟如何。

我们只提取出待定拓展矩形结构，分情况讨论。依旧是跨结构的两个节点： $r5c2(1)$ 和 $r3c23(9)$ 。我们分别讨论两个情况，如下面的两个图所示。

6 8	3	1	2 9	4 8 9	4 8	2 4 9	7	5	2 8	3	1	2 9	4 8 9	4 8	2 4 9	7	5
6 8		4	5	6 8 9	7	3	2 8	1	6 8	2	4	5	6 8 9	7	3	2 8	1
5	2	7	3	1	4 8	4 9	6	4 8	5	2	7	3	1	4 8	4 9	6	4 8
4 8	4 8 9		6 9	7	4 8 9	4 8	2 1 6	5	3	4 8	1	2	6	7	3	2 1 6	5
3	1	2		4 8	6	7	9	4 8	3	2	5	2	5	1	2	4	6
7	4 8		6 9	1 9	3	5	1 6 4	2	7	4 8		6 9	3	5	1 6 4	2	
1	7	5	4	2	9	8	3	6	1	5	7	4	2	9	8	3	6
4 2	4		8	6	7	3	5	1	9	4 2	4		8	6	7	3	5
9	6	3	8	5	1	4 2	4 2	7	9	6	3	8	5	1	4 2	4 2	7

当 $r5c2(1)$ 为真时，显然我们可以发现 $r5c3 = 2$ 、 $r7c3 = 5$ ；当 $r3c23(9)$ 为真时，可以发现 $r5c23$ 为 2、5 的显性数对，这便使得 $r46c3$ 两个单元格是无论如何都不能放 2 和 5 的，否则它就会使得两种情况都产生矛盾。所以这个例子的额外删数还有 $r4c3(2)$ 。当然， $r46c3(25)$ 是理论上的删数，而题目上并不存在其它的三个候选数，就不用管了。

7-4-6 空矩形欠一数对 (Empty Rectangle ALP)

1		8	4	3	1	3	5	2	7	1	3	3	6
2	1		3	7	1	3	6	9	4	5		3	8
	3	6	5	1	4	3	1	3	1	3	9	2	
6	4		3	1	4	3	2	8	9	7	5		
7	9	8		5	4	3	6	2	4		1		
4		5	2	9	1	7		3	6	8	4	3	6
1	1	3	6	2	8	1	3	1	3	1	3	3	3
8	1	2	3	3	1	3	1	3	1	3	1	2	3
5	1	2	3	3	1	3	1	3	1	3	1	2	3

如图所示， $r3c1$ 只有候选数 3 和 4， $b2$ 里也同时包含 3 和 4 的空矩形结构。

- 如果假设 $r3c1 = 3$ ，则由于空矩形的关系， $r12c4(3)$ 为真，所以 $r4c4 = 4$ ；
- 如果假设 $r3c1 = 4$ ，由于空矩形的关系， $r12c4(4)$ 为真，所以 $r4c4 = 3$ 。

可以看到，只要我们假设出其中一种情况，则得到 $r4c4$ 的填数就一定是另外一种情况，所以我们可以得到 $r3c1$ 和 $r4c4$ 此时形成跨区数对结构，所以 $r3c4$ 和 $r4c1$ 都不能填入 3 和 4。

与此同时，由于空矩形结构的特殊性，在 $r3$ 和 $c4$ 上都会产生 3 和 4，一个位于 $r3c1$ 或 $r4c4$ 两个单元格上，而另外一个单元格则一定在这个空矩形里。所以 $r3$ 和 $c4$ 里的其

可以看到，这个示例里我们通过空矩形得到了一个跨区数对，而更巧妙的是，它甚至还得到了类似于欠一数对的结果，所以这个技巧叫做**空矩形欠一数对**（ALP in Empty Rectangle），有些地方叫它**空矩形 W-Wing** 或直接简称**空 W**，因为它的结构实际上是由两个完全相似的 W-Wing 构成的，如果你把它拆开来看的话。

[illegible]

观察一下推理逻辑，我们可以发现，当我们假设 $r9c1 = 9$ 时，也可以类比上述逻辑，依旧可以得到 $r1c9 = 7$ 的结果。所以我们完全可以得到 $r1c9$ 和 $r9c1$ 是一组关于 7 和 9 的跨区数对结构，于是， $r1c9$ 和 $r9c1$ 本对应得到的地方（如 $r1c1$ 和 $r9c9$ ）显然就不能再放入多余的 7 和 9 了，所以可以删除掉；与此同时， $b9$ 的空矩形区域里，必须有一处区块里是填入 7 的，而另外一组区块里则是填入 9 的，这恰好又使得 $r9$ 和 $c9$ 上产生了关于 7 和 9 的数对结构，所以删除掉其余位置的 7 和 9。所以图上的所有红色候选数均可以被删除。

至此，我们学习了很多关于环的使用套路和方式，环的内容就暂时告一段落，接下来将会有有一个从另外一个维度拓展链的思想：**强制链（Forcing Chain）**。

Part 8 强制链与强制结构

强制链（Forcing Chain）是链的一个重要的思想，为了让链的使用范围更加广泛，便产生了这个思想。

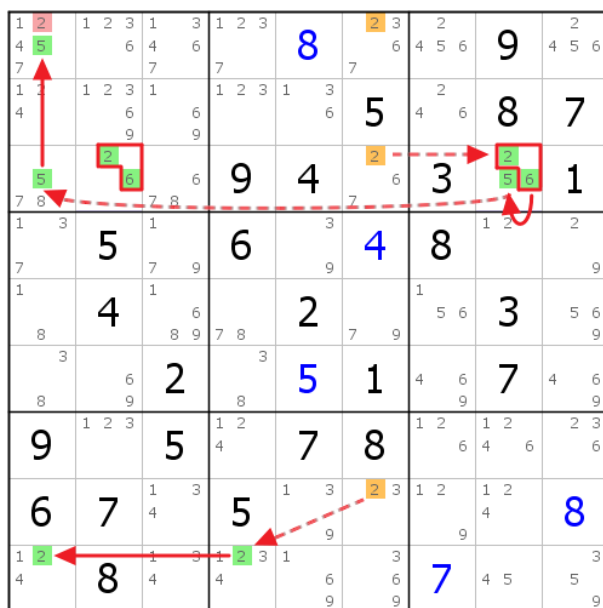
8-1 基本的强制链类型

下面陈列一些基本的、常见的强制链模型。

8-1-1 区域和单元格强制链（Region & Cell Forcing Chain）

区域强制链（Region Forcing Chain）和**单元格强制链**（Cell Forcing Chain）属于一类强制链，这个类型的强制链使用链的思维，将某一个区域所有每个候选数，或某一个单元格的所有候选数情况进行分情况讨论，并以弱关系延伸（即假设每一个分支的头节点为真的形式进行向下推理），直至所有分支的尾节点均能得到一致的结论。如果全部情况都能得到一致的结论，则结论一定成立。不过，区域强制链是按行列宫的某一种候选数进行分情况讨论的，而单元格强制链则是按单元格的所有候选数进行讨论的。

8-1-1-1 区域强制链（Region Forcing Chain）

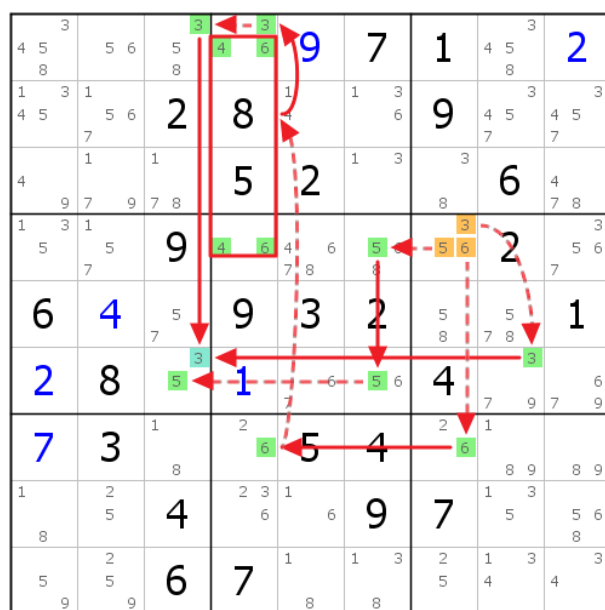


如图所示，可以发现 c6 一共有三个 2，我们将每一个 2 都进行讨论。

- 第一种情况：**r1c6(2)为真**。显然，r1c6 = 2 时，一定有 r1c1 \neq 2；
- 第二种情况：**r3c6(2)为真**，此时有一条强制链：r3c6(2)-r3c28(26)=r3c8-r3c1=r1c1(5)，得到 r1c1(5)为真，所以一定也有 r1c1 \neq 2；
- 第三种情况：**r8c6(2)为真**，此时有一条强制链：r8c6-r9c4(2)=r9c1(2)，得到 r9c1(2)为真，所以 r1c1 \neq 2。

所有情况均能使得 r1c1 \neq 2，所以 r1c1 \neq 2 结论是成立的，可以删除掉它。

8-1-1-2 单元格强制链 (Cell Forcing Chain)



如图所示，按 $r4c7$ 的所有候选数进行分情况讨论。

- 当 $r4c7(3)$ 为真时，则有强制链： $r4c7-r6c8=r6c3(3)$ ，即得到 $r6c3 = 3$ ；
- 当 $r4c7(5)$ 为真时，则有强制链： $r4c7-r4c6=r6c6-r6c3(5=3)$ ，即得到 $r6c3 = 3$ ；
- 当 $r4c7(6)$ 为真时，则有强制链： $r4c7-r7c7=r7c4(6)-r14c4(46)=r1c4-r1c3=r6c3(3)$ ，即得到 $r6c3 = 3$ 。

所有的单元格的情况都能得到 $r6c3 = 3$ ，所以 $r6c3 = 3$ 结论成立。

可以看到，上述的两种推导方式都不是特别好，因为它多少都有一点试数的风格。不过，有些题目不得不使用这类技巧才能完成。所以请读者你在使用之前斟酌，不要随时随地都使用它，这样是不合适的。

8-1-2 矛盾强制链 (Contradiction Forcing Chain)

前面叙述了一种分情况讨论的强制链，接下来要讲到的**矛盾强制链** (Contradiction Forcing Chain) 则是另外一种类型的强制链。这种强制链依赖于一个节点的两两种情况。当两种不同的情况均能得到一致的结论的时候，则结论成立。

1	1	8	2	5	3	5	6	4	5	6	5	6	4	3
7	9	7	9			5	6	9			7		4	6
		6	2	5	3			4	5		1			3
7	9			5	3			5	8		7	8		
4	3	5	7	8	9			6	2	6	9		2	6
								8	8					8
1	6	8	1	4	6	4	7	9	3	1	2	2	6	5
7			7						4	6	7			
1	3	1	1	3		2	2	1	5	4	6		1	
5	6	5	4	6	4	5	4	6				6	4	6
7	9	7	9	7	9	7	8	7	8	8	7	8	7	9
2	5	4	6	1	5	4			3					
									8	9				
1	3	4	1	3	5	8	2	7	2	1	2		6	
5	6				9		9		6	9				
1	5	6	2	1	6	3	5	6	1	5	6		1	
7	8	9	7	9			7	9		8	9		8	9
7	8	6	5	6	4	5	4		1	3		6	2	6

如图所示，按 r9c8(8) 执行两种情况的假设。

- 假设 r9c8(8) 为真，则有强制链：r9c8-r9c1=r8c1(8)，则 r8c1 <> 6；
- 假设 r9c8(8) 为假，则有链：r9c8=r5c8-r5c4=r2c4(8)-r1c5(6)=r8c5(6)，依然可以得到 r8c1 <> 6。

不论真假，都可以得到 r8c1 <> 6，所以 r8c1 <> 6。

另外，这个类型的强制链一般可以改写为普通链的形式，仅需要将其中一种情况反向即可，所以矛盾强制链一般不实用，仅在一些特殊情况（例如嵌套结构里）使用较多。

8-2 链能看逆向，那么强制链呢？

可以看到，强制链和链不一样的是，强制链有至少两个分支，而带有不同的推理情况。那么链可以逆向理解，强制链是否也可以呢？实际上是可以的。我们先来看区域和单元格强制链的大致逻辑：**将所有情况进行分情况讨论，如果全部情况都能得到一致的结论，则结论一定成立。**那么倒过来，则是这样的一种情况：**假设某条件成立，则能推出某一个区域下的所有某一候选数全为假，或者某一个单元格的所有候选数全为假时，产生矛盾，所以原假设错误。**

同理，矛盾强制链的逆向视角则是：**假设某条件成立，则能推出某一个节点存在真假两种情况都成立的矛盾情况，进而得到原假设错误。**

8-3 致命结构的强制视角

实际上，目前仍然有一部分结构依然在使用强制的视角来解题，例如致命结构，这里就列举出一些。

8-3-1 强制 UR (Forcing UR)

强制 UR 是一种利用强制链思想，假设某个待删除的候选数为真，然后通过结构内部的填数逻辑，当得到出现致命形式时即矛盾，进而反证出原假设错误的一种技巧。不过这个技巧同时还依赖于一些共轭对来得到。

8-3-1-1 UR + 2 / 1 CP

1 2 5 9	1 2 5 9	7	4 2 5 9	3	6	8	1 4 9	4 5 9
1 2 5 6 9	1 2 5 6 9	3	4 2 5 9	2 5 9	8	1 4 5 6 9	7	4 5 6 9
8	5 6 9	4	5 9	1	7	2	6 9	3
4 6 9	6 9	2	8	7	1	3	5	4 9
4 9	8	5	3	6	4 2 9	1 4 9	1 2 4 9	7
3	7	1	2 5 9	2 5 9	4 2 5 9	4 2 5 9	6 4 9	8
1 2 5	1 2 5	6	7	4	3	5 9	8	2 5 9
7	3	9	1	8	2 5	4 5 6 9	4 6 9	4 5 6 9
2 5	4	8	6	2 5	9	7	3	1

如图所示，我们假设 $r8c7 = 6$ ，由于结构的两侧 $r6c7$ 和 $r8c8$ 都是双值格的关系，它们将同时得到填入 4 的结果。然后，由于 $r7$ 存在 6 的共轭对的缘故，此时只能使得 $r7c8$ 填入 6。所以此时四个单元格就只填入了 4 和 6，出现了可以互换的形式，即形成了致命形式，所以矛盾，故原假设错误，即 $r8c7 \neq 6$ 。

这个结构显得些许暴力，它是通过假设来得到致命形式的，但还是和无逻辑的暴力不同，它使用了致命形式来得到矛盾。

由于这个结构带有一个共轭对和两个候选数，所以我们一般将这个结构简记为 **UR + 2 / 1 CP**。其中的 CP 是 Conjugate Pair（即共轭对）的缩写。一定要注意的，论证过程并不会借助于外部的任意一个单元格，推导过程只在结构内部进行。只要结构内得到形成致命形式的填法，就可以推出矛盾。

有些地方将 UR + n 的 n 表示为“结构内一共有多少个单元格包含额外的数字”，而不是“结构内包含多少额外的候选数（真数）”，此题恰好有两个单元格，所以依然被记作 UR + 2；但有些地方的记号的结果不太一样，由于本文档里为了统一 UR + n 和 BUG + n 的说法，所以把 n 作为一致的定义来处理。请务必引起重视。

8-3-1-2 UR + 4 / 2 CP

9	1	²	4	²	3	5	6	8
² 5	8	4	² 5 6	² 5 6	1	3	9	7
⁵ 7	6	3	8	9	^{1 2} 7	4	^{1 2}	
² 7 8	3	^{1 2} 5	9	4	² 5	^{1 2} 7 8	1	6
6	⁵ 7	^{1 2} 5	³ 7	8	² 5	4	^{1 3} 7	9
4	9	² 7 8	³ 7	1	6	² 7 8	5	^{2 3}
^{7 8}	4	^{5 6} 7 8	² 5 6	^{2 3} 5 6	9	¹ 7	^{1 3} 7 8	^{1 3} 5
1	2	^{5 6} 7 8	^{5 6} 7	³ 5 6	4	9	³ 7 8	³ 5
3	⁵ 7	9	1	^{5 6} 7	8	⁶ 7	2	4

如图所示，图上有两处删数，我们先来找 $r4c6(2)$ 的原因。假设 $r4c6 = 2$ ，则由于 $r5c6$ 是双值格的关系， $r5c6 = 5$ ，而 $r4$ 存在 5 的共轭对，所以 $r4c3 = 5$ 。与此同似乎，由于 $r5c6 = 5$ ，而 $r5$ 上存在 2 的共轭对，所以 $r5c3 = 2$ 。所以四个单元格形成了关于 2 和 5 的可交换的致命形式，故矛盾。所以 $r4c6 \neq 2$ 。

这种结构被简记为 **UR+4/2CP**。另外，我们还有一个额外的删数，请读者你来自行证明。

8-3-1-3 UR + 4 / 3 CP

^{5 6}	²	²	¹	7	9	3	^{1 2}	^{1 2}
1	9	²	^{4 5} 8	3	⁴ 8	² 4 5 7 8	² 4 6	² 8
³ 7	⁵ 7	³ 7	8	2	6	^{4 5} 7	¹ 4 7	9
4	^{2 3} 7 8	1	^{7 8}	5	^{2 3} 7 8	² 8 9	^{2 3} 9	6
³ 8	5	9	^{4 6} 8	¹ 4 8	^{1 2 3} 4 8	² 4 8	^{2 3} 4	7
³ 7 8	^{2 3} 7 8	^{2 3} 7	^{4 6} 7 8 9	^{4 6} 8 9	^{2 3} 7 8	1	5	^{2 3} 8
2	¹ 7 8	5	3	¹ 4 8 9	¹ 4 7 8	6	¹ 4 7 9	¹ 4
³ 7 8 9	¹ 7 8	³ 7	2	¹ 4 8 9	¹ 4 7 8	⁴ 7 9	^{1 3} 7 9	5
³ 7	6	4	³ 7	¹ 6 9	5	² 7 9	8	^{1 2 3}

如图所示，假设 $r2c8 = 4$ ，则 $c8$ 和 $r2$ 都有 6 的共轭对，所以可以得到 $r1c8$ 和 $r2c3$ 填 6；与此同时，由于 $c3$ 有 4 的共轭对，所以 $r1c3 = 4$ ，此时形成关于 4 和 6 的致命形式，所以矛盾，故 $r2c8 \neq 4$ 。

这个结构被称为 **UR + 7 / 3CP**。这个结构就需要依赖于三个共轭对，所以比较难发现到。

8-3-1-4 残缺 UR + 2 / 1CP

7	4	2	1	³ ₆	8	⁵ ₉	³ ₆	⁵ ₉
9	¹ ₃	5	³ ₆	4	7	¹ ₂	¹ ₂ ⁶	8
¹ ₃	8	6	2	9	5	¹ ₄	¹ ₃ ⁴	7
² ₄	² ₅	3	⁴ ₅	1	9	7	8	6
8	⁷ ₉ ⁷ ₉		³ ₅ ² ₃	6	¹ ₂ ⁴ ₅	¹ ₂ ⁴ ₅	¹ ₂ ⁴ ₅	
⁴ ₅	6	1	8	7	² ₄	3	² ₉ ² ₅	² ₉
³ ₅	³ ₅	4	9	6	1	8	7	² ₄
6	¹ ₂ ⁹	8	7	5	² ₄	¹ ₄ ⁹	¹ ₂ ⁴ ₉	3
¹ ₂	¹ ₂ ⁷	4	4	8	3	6	5	¹ ₂ ⁴ ₉

如图所示，假设 $r7c4 = 4$ ，则两侧都是双值格，故得到 $r7c3 = r9c4 = 9$ ，而 $c3$ 有 4 的共轭对，所以 $r9c3 = 4$ ，此时形成关于 4 和 9 的致命形式，所以矛盾，故 $r7c4 \neq 4$ 。

8-3-1-5 残缺 UR + 2 / 1CP 的另一则示例

1	⁶ ₉	² ₅	² ₅	⁶ ₉	7	4	3	8
⁴ ₅ ⁷	2	8	² ₅	⁶ ₉	3	2	¹ ₆ ⁷	¹ ₇
3	⁶ ₉ ⁷		8	1	4	2	⁶ ₉	5
⁴ ₅	7	⁴ ₅ ⁹	1	2	6	8	⁵ ₉	3
2	¹ ₅	6	9	3	8	7	¹ ₅	4
8	3	¹ ₉	4	7	5	6	2	¹ ₉
9	¹ ₅	¹ ₅	7	4	2	3	8	6
6	8	⁴ ₇	3	5	9	1	⁴ ₇	² ₇
⁴ ₇	² ₄	3	6	8	1	5	⁹ ₇ ⁹	

如图所示，再例如这个示例里，假设 $r2c2 = 2$ ，由于 $r23c7$ 都是双值格的关系，所以一定填入 2 和 9，而 $r3$ 存在 9 的共轭对，所以 $r3c2$ 必然也是 9，所以即使残缺到这个形式，依然能形成致命形式。

8-3-1-6 残缺 UR + 3 / 1CP

2 5		3 1	2 1	6 5	3 4	9		6
1 2	8 7	3 7	1 6	4 5	8 3		2 3 6	5 7
2 4 5	7 4	8 3	7 9	5 3	1 2 7	8	1 3	
3	6	8	9	2	7	5	1	4
9	1	2	4	5	6	3	7	8
7	5	4	3	8	1	9	6	2
1 6	2	3	1 6	7	4	8	5	9
8	4 7	1 7	5	1 3 6	9	1 2 6 4	2 3 1	3 6
4 6	9	5	2 8	1 3 6	2 8	1 7	3 4	1 3

如图所示，这个例子和上面的同理，就留给你自己推理了。

8-3-1-7 分情况讨论的结构

The image displays two 9x9 Sudoku puzzles side-by-side. The left puzzle is a 4-star rated puzzle, and the right puzzle is a 3-star rated puzzle. Both puzzles have a grid with numbers 1-9 and empty cells. The left puzzle has a red box highlighting a 2x2 area in the center (rows 4-5, columns 4-5) and red arrows indicating a cycle of numbers 3, 4, 5, 6. The right puzzle has a red box highlighting a 2x2 area in the center (rows 4-5, columns 4-5) and red arrows indicating a cycle of numbers 1, 2, 3, 4.

如图所示，我们发现 **c6** 上只有两处 7 可以填入，我们尝试分情况讨论 7 的填数。

如果 7 此时放在 **r3c6** 上，那么我们就可以找到上述的强制链结构，并最终得到 **r4c3 <> 7**。于是，**r348c13** 将构成关于 1、2、8 的拓展矩形，且属于共轭对的类型，故我们可以直接删除掉 **r8c13(8)**；而如果 7 此时放在 **r9c6** 上，那么我们则可以直接得到一个 UR 的共轭对类型，并直接得到 **r8c1 <> 8**。

所以，不论两种情况哪一个的成立，我们都可以删除，所以 $r8c1 <> 8$ 。

这个示例看起来似乎跟强制 UR 没有什么关系，但它属于强制思维的推导，所以我们照样列举在此。

8-3-2 强制拓展矩形和强制唯一环 (Forcing XR & UL)

下面, 我们来看两则有关拓展矩形和唯一环的强制推导版本。当然, 这些例子都不容易被观察到, 所以很少出现。

8-3-2-1 强制拓展矩形 (Forcing XR)

4 6 7	1 4 6	3	2	6 4 7	6 9	5 6 7	5 6 9	8	4 6 7	1 4 6	3	2	6 7 9	6 9	5 6 7	5 6 9	8
5	2	1 7 6		3 6 7	9	8	4	1 3 6 7 9	5	2	1 7 6		3 6 7	9	8	4	1 3 6 7 9
4 6 7	9	8	1 4 5 6 7	5	1 3 4 6 7	2		3 6 7 9	6	9	8	1 5 4	5	4	3 6 7	2	6
8	1 3 6	1 6 9	5 6 9	7	2	5 3 6 9	5 3 6 9	4	8	1 3 6	1 6 9		7	2	5 3 6 9	5 3 6 9	4
4 6 7	4 6 7	2	4 5 6 9	6 4 5 6 9	6 4 5 6 9	5 6 8 9	5 6 8 9	1	3 6 7	2	4	6 9	5		7 8 9	6 8 9	1
4 6 7	5	4 6 7 9	8	1	4 6 9	2		3 6 7 9	4 6 7 9	5	4 6 7 9	8	1		3 6 7 9	2	6 7 9
1	4 6 8 9	4 6 9	3	2		6 8 9	6 8 9	7	5	1	4 6 8 9	4 6 9	3	2		6 8 9	7 5
2		6 7 8 9	1 5 6 7 9	4	1 5 6 7 9		6 8 9	6 8 9	3	2		6 7 8 9	6 7 9	5	4	1	6 8 9
3 6 9		3 6	5		6 8 9	8	1	4 2	3 6 9		3 6	5		6 8 9	8	1	4 2

如图所示, 如果我们假设 $r2c6 = 4$, 则我们可以根据左图给出的这些共轭对, 以及 $b5$ 的 5 的候选数分布, 可以得到右图的填数结果。显然, 这样的构型是出现了致命形式的, 因为这样的填数方式显然可以左右交换, 而完全不产生任何影响。所以这样是违背数独规则的, 因此, 我们假设的 $r2c6(4)$ 就是错误的, 应当删除掉它。

8-3-2-2 强制 UL (Forcing UL)

和强制 UR 的思维完全一样, 我们也含有强制 UL 的思路, 不过例子就比较难找到了, 下面为你展示一则这样的示例。

4 5 7	4 5 7	3 6	2 5	8	4 2 3	9	1	5 7
1 5 7		3 5	2	9	1 5 6	8	5 6 7	4
8	4 5 9	1 4 5 9	1 6	1 4 5 6	7	3	5 6	2
2 4 5 9	2 4 5 9	3	8	4 5 7 9	6	1	4 5 7	5
6	8	4 5	3	1 4 5 7	1 4	2 4 5	2 4 5 7	9
4 5 9	1	7	2 5	4 5 9	4 9	6	8	3
1 4 5 7 9	4 5 7 9	1 4 5 8 9	1 7	2	1 8 9	4 5	3	6
3	2 4 7 9	1 4 8 9	1 6 7	1 6 9	5	2 4	2 9	1 8
1 2 5 9	6	1 5 8 9	4	3	1 8 9	7	2 5 9	1 8

如图所示, 如果假设 $r9c1 = 9$, 则由于 $r9$ 和 $c1$ 的 2 的共轭对、 $c8$ 的 9 的共轭对,

以及 2 的直推结果, 我们最终可以得到关于 2 和 9 的 UL 的致命形式, 所以 $r9c1(9)$ 为假, 删掉它。

8-3-3 强制 AR (Forcing AR)

同 UR, 它也具有强制的类型。不过示例不太多。

8-3-3-1 AR + 3 / 1CP

1	9	1 2	7	6	4 3	1	5	2 3
8		8			8	4		
6	2	3	1	5	4	1	8	2
7	7		9		9	4		
1	4	5	1 3	1 3	2	9	6	3
7 8			8	8				7
9	1	4	6	7	5	3	2	8
5 3	5 3	6	2	3	1	7	4	4 9
5 8	5 8		8	8			9	9
2	3	7	4	9	3	5	1	6
	8			8	8			
4 3	6	1	5	1 3	3	2	7	1 4
	8 9		8	8	9			9
4 3	3 1	1 3	2	6	8	4	5	
4 7	7	9	9			9		
1	2	1 2	1	4	7	6	3	1
5 8	5 8	8 9	8					9

如图所示, 假设 $r3c4 = 1$, 则由于 $r2c4$ 是双值格的关系, 得到 $r2c4 = 9$, 而 $r2$ 存在 1 的共轭对, 所以此时 $r2c7 = 1$, 而算上 $r3c7$ 填入的 9, 四个单元格形成可交换的致命形式, 所以产生矛盾, 故 $r3c4 \neq 1$ 。

8-3-3-2 带一个强制链的强制 AR

6	5	7	4		3	4	2	9	1
1		1	8		8	4			
8 9	3	8 9	2	6	7 9	7 9	4	5	
4		2	7 9	1	5	6	3	7	2
8 9									
1	1	6	5	4	8	3	7	2	
9	9								
5	2	4	3	7	6	9	1	8	
3		3	1	9	2	5	6	4	
7 8	7 8	8							
	6	5	4	2	4	1		3	7
8			9		9		8		
	3	4	3	5	1	6	2	3	6
7 8 9		8 9	7 8			8			
2	1	1	6		3	4	5	9	
7 8	8		8	7					

如图所示, 这个例子带有一个强制链, 但不影响推理的逻辑。假设 $r3c7 = 7$, 则 $r2c7$

<> 7, 此时由于 r2c7 和 r3c8 是一组 ALS 的关系, r3c8 不得不填入 3, 否则两个单元格都只会存在 8 这一种候选数, 导致矛盾。但填入 3 后发现 r34c78 四个单元格只有 3 和 7 的出现, 所以产生可以交换的致命形式, 所以矛盾。故 r3c7 <> 7。

8-3-4 直推 UR (Direct Inference UR)

直推 UR (Direct Inference UR) 是一种非常特殊的 UR, 它利用了排除来执行强制 UR 的操作。

4 5 7	3 5 7	4 5 8	1	4 6 8 9	5 6 8 9	2	4 5 7 9	3
6	2 4 5 7	1 2 4 5	4	3	5 7 9	1 4 5 9	8	
1 3 4 5 8	9	1 4 5 8	2	4 8	7	3 4 5	6	1 3
9	2 3 4 5 8	2 4 5 6 8	3 4 6 7 8	2 4 6 8	3 6 8	1	2 5 8 7	2 6
4 8	1	2 4 6 8	4 6 7 8 9	5	6 8 9	6 7 8 9	3	2 6 7 9
3 5 8	2 3 5 8	7	3 6 8 9	1 2 6 8 9	1 3 6 8 9	5 6 8 9	2 5 8 9	4
1 4 8	6	1 4 8 9	5	1 8 9	2	3 4 8 9	7	1 3 9
2	4 8	1 4 8 9	3 6 8 9	7	1 3 6 8 9	4 6 8 9	1 4 8 9	5
5 7	5	3	6 8 9	1 6 8 9	4	6 8 9	1 2 8 9	1 2 6 9

如图所示, 假设 r1c2 = 5, 则由于 r9c12 是双值格的关系, 立马可以得到 r9c12 一定填入的是 5 和 7。不过由于 r9c2 = 7 的关系, 对 b1 作出排除, 可以发现 b1 最终能放入 7 的位置只有 r1c1, 所以 r1c1 = 7, 此时四个单元格就形成了致命形式, 故矛盾, 所以 r1c2 <> 5。

8-3-5 直推 AR (Direct Inference AR)

和上面一样，AR 和 UR 的情况和规格是一样的，所以它依然具有直推的版本。

8-3-5-1 简易的版本

6	¹ _{4 7 8}	5	¹ _{7 8}	2	¹ _{4 7}	3	⁴ ₇	9
9	² _{4 7 8}	³ _{7 8}	5	³ _{4 7}	1	6	² ₇	
² ₇	^{1 2} _{4 7}		9	^{1 3} ₇	6	8	² _{4 7}	5
8	³ ₇	2	4	9	^{1 3} ₇	⁵ ₇	^{1 5} ₇	6
³ ₇	5	9	^{1 2} ₇	6	^{1 2 3} ₇	4	8	¹ ₇
1	6	4	7	5	8	2	9	3
4	9	^{7 8} _{7 8}	3	¹ _{7 8}	5	6	^{1 2} _{7 8}	^{1 2} _{7 8}
^{2 3} ₇	^{2 3} _{7 8}	1	6	^{7 8} _{7 8}	² _{9 7}	⁵ _{9 7}	³ ₅	4
5	^{2 3} _{7 8}	6	^{1 2} ₈	4	^{1 2} _{7 9}		^{1 3} _{7 9}	¹ _{7 8}

如图所示，假设 $r2c3 = 8$ ，则观察 $b2$ ，发现填入 8 的位置只有 $r1c4$ ，所以 $r1c4 = 8$ ，此时 $r12c34$ 形成关于 5 和 8 的致命形式，所以 $r2c3 \neq 8$ 。

8-3-5-2 跳转很多次的版本

³ ₆	8	5	¹ _{4 6 7}	¹ _{4 6 7}	¹ _{4 6 7}	³ _{7 9}	2
1	7	² _{4 5}	8	3	6	⁵ _{4 5}	
^{2 3} ₆	9	² _{4 5 6}	^{1 2} _{4 6}	¹ _{4 6}	8	³ _{4 5 7}	³ _{4 5 7}
² ₄	6	¹ ₉	3	¹ _{4 7 9}	8	5	² ₇
5	^{1 2 3} ₄	^{1 2} ₄	¹ _{4 6}	¹ _{4 6}	¹ _{4 6}	³ _{7 9}	8
8	¹ ₃	7	¹ _{6 9}	5	2	^{1 3} ₉	4
9	² _{4 5}	3	8	^{4 6} _{4 5 6}	² _{4 7}	1	^{4 5} ₇
^{4 6} _{4 5}	8	7	2	¹ _{4 5}	³ _{5 6}	³ _{5 6}	9
7	^{1 2} _{4 5}	^{1 2} _{6 4}	¹ ₃	9	² ₄	8	^{4 5 6} _{4 5 6}

如图所示，假设 $r6c2 = 3$ ，则可以沿着箭头方向作出直推，最终依次得到了 $r5c7 = 3$ 、 $r8c8 = 3$ 、 $r3c9 = 3$ 、 $r1c1 = 3$ 。此时， $r16c12$ 形成了关于 3 和 8 的致命形式，故产生了矛盾，所以 $r6c2 \neq 3$ 。

这个例子跳转了很多次，甚至把全盘的 3 都走了个遍，才得到矛盾的出现。

8-3-6 UR + n

第一种技巧是之前简单说到的 UR + n，将额外的数字全部去掉后，UR 出现致命形式，进而规避的一种解法。但是 UR + n 一般都具有特定的删数类型，诸如 UR + 1（标准类型）、UR + 2（或者 UR + 3 的区块类型）等等。接下来来探讨一些“+ n”的逻辑。

5	7	4	6	8	3	1	9	4	6	2
2	9	3	4	6	7	4	6	8	1	5
1	4	6	8	2	5	9	4	3	4	6
3	2	3	2	1	9	4	6	4	5	6
4	5	6	1	7	3	2	3	1	5	6
9	8	7	1	4	6	4	5	1	4	6
3	6	4	5	6	9	4	8	2	7	5
7	3	1	9	5	6	7	3	2	8	4
7	3	2	3	2	3	1	7	3	5	9
7	8	4	5	4	5	1	7	8	5	9

如图所示。可能你很难看出这是一个 UR 技巧。实际上这是一个 UR + 6，但实际上要用到的只有一个候选数 r4c3(6)。

- 当 r4c3(6)为假的时候，这实际上是一个 HUR（隐性唯一矩形），r9c2 所在行列均含有 2 的共轭对，且 r4c3 是双值格，这两样东西是 HUR 的基本性质和特征，所以此时我们可以立马得到它是 HUR，删数是包含共轭对的 r9c2 的 UR 的另外一个数，即 r9c2(5)。
- 当 r4c3(6)为真时，就需要引用强制链来执行逻辑了：r4c3-r5c2(6=5)，得到 r5c2(5)为真，所以 r9c2(5)此时依然为假。

所以不论 r4c3(6)是否为真，都可以得到 r9c2(5)为假，故 r9c2 <> 5。

这个例子我们暂且称为 UR + 6，虽然其中的 5 个候选数跟该技巧无任何关系（因为 HUR 的要求不需要候选数特别精确，比如共轭对类型下，只需要一个共轭对就可以构成结构，至于共轭对的两个单元格里含有其它什么候选数，跟题目都无关。所以实际上这些数字是无效的，但技巧名要算上（当然，如果你执意称它为“HUR + 1”的话就另说了）。

我们再来看一个例子。

7	4	6	4	3	2	1	3	1	3	8	5	1
2	6	9	4	2	1	5	6	1	5	6	3	4
1	1	3	5	3	4	7	8	7	9	4	5	1
4	1	3	3	9	1	2	3	1	2	3	6	2
5	2	8	9	4	3	4	6	7	3	6	1	8
1	1	3	6	1	5	8	1	2	3	5	4	2
2	4	6	1	4	3	5	2	3	6	6	7	9
3	5	4	7	6	4	7	1	8	9	2	1	8
2	6	8	2	1	3	1	2	3	1	2	3	5

如图所示，这个例子有点麻烦。我们需要按 $r6c6(5)$ 的真假作出两种情况的讨论。

- 当 $r6c6 = 5$ 的时候，我们假设 $r2c6 = 1$ ，你就会发现， $r26c46$ 会形成关于 1 和 5 的致命形式（强制 AR 的逻辑），这里就不推理了，你可以尝试自己找一下共轭对和双值格。
- 当 $r6c6 \neq 5$ 的时候，我们就可以找到一条不连续环，从 $r2c7(1)$ 开始，并得到 $r2c6 = 5$ 的结果，此时 $r2c6$ 依然不能填 1。

所以两种情况下， $r2c6 \neq 1$ 。所以 $r2c6 \neq 1$ 。这个例子使用到了强制 AR 和矛盾强制链的两个思想。

8-3-7 UL + n

7	1	3	9	2	4	5	6	1	3	1	3
4	1	8	5	6	7	3	1	8	8	2	9
2	3	3	5	6	8	1	9	4	5	3	7
5	4	1	7	2	6	3	9	8	3	1	5
3	2	3	4	9	8	7	1	5	1	5	5
8	9	7	3	5	1	2	6	4	3	1	3
9	8	4	5	6	7	1	3	1	3	8	2
1	7	3	9	8	2	5	3	4	5	6	8
3	6	5	6	2	1	4	9	7	5	6	8

如图所示，如果 $r2c23(5)$ 和 $r8c7(5)$ 同假的话， $b1$ 内的 $r3c23$ 将产生 5、6 隐性数对，而 $r8$ 和 $b9$ 也都会产生相应的 5、6 隐性数对。此时 $r3c23$ ， $r8c39$ ， $r9c9$ 五个单元格都只有 5 和 6 两种候选数。而 $r9c2$ 本身就是 5 和 6 的双值格，所以 $\{r3c23, r8c39,$

$r9c29$ 六个单元格形成关于 5 和 6 的致命形式，故矛盾。所以 $r2c23=r8c7(5)$ 是成立的，故删除两端的交集，即 $r2c7(5)$ 。

这个例子有趣的地方是，它借用了外部的逻辑来“死锁”一个 UL 结构，比较类似于之前借用弱关系来死锁住 UR 的形式。

8-3-8 BUG + n

有时候，BUG + n 也无法直接得到删数，于是我们需要链的帮助，所以 BUG + n 也存在强制链视角的版本，接下来就介绍两则示例，其实逻辑很好理解，但不好寻找，所以提供给大家欣赏。

8-3-8-1 BUG + 5

3	2	4		7	5	1	8	4		6		
4		9	7	1	6	2	8	3	4		9	5
8	6	5	3	4	9	7	2	1				
5		1	4	8	9	6		2	1			
	1	4	2	5	3	6		1	2			
7		9	3	2	1	4	5	6	7			
1	3	2	2	1	4	5	6	7				
6		8	9	8	9	1	7	2	4	5	3	
1	4	1	4	3	8	9	5	6	7	2		
2	5	7	4	6	3	9	1	8				

如图所示，我们细数这个题目，一共包含 5 个真数，所以这个题目是 BUG + 5。和 UR + n 不同的是，它里面的真数将会被得到充分使用。

假设真数 $r4c2(1)$ 为真，则有强制链： $r4c2-r6c1(1=9)$ ，则得到 $r6c1 = 9$ ，故可以删除 $r5c2(9)$ ；而其余四个真数均能直接对应删除 $r5c2(9)$ ，所以我们知道，不论真数里是谁成立，或者哪几个成立，我们都能保证 $r5c2(9)$ 是可以通过对应的逻辑删掉的，所以 $r5c2(9)$ 一定是可以删除的。

当然，这个例子你也可以理解为 $(BUG + 4) + 1$ ，当然 BUG + 4 需要打括号，因为此时的 BUG + 4 看作一个整体，而这个整体结构上包含了一个额外的数字，即 $r4c2(1)$ 。即当 $r4c2(1)$ 为假时，BUG + 4 结构成立，删除 $r5c2(9)$ ；而 $r4c2 = 1$ 时，则有强制链可以删除，所以 $r5c2 \neq 9$ 。这样理解也可以。

8-3-8-2 BUG + 10

有时候，寻找 BUG + n 也是一件趣事，当 n 很大的时候，技巧性就很强，虽然实际意义就不大了，但是也非常有趣。例如下面这个例子就是令人抓狂的 BUG + 10，10 个真数都能通过强制链证明得到同一个删数结论。

5	4		3	1	2	9	8	3	7
1 2	1 2	6		3	7	8	4	5	1 6
7	1 3	1 3	8	5	4	6	2	9	1 3
9	5	2	2	6	1	3	3		4
4	2 3	2 3	2	5	3	1	1	6	9
6	1 3	1 3	8	4	9	3	5	8	2
3	1		4	6	8	2	1	1	5
2	7	2	9	1	5	3	4	6	8
1	1	6	5	7	3	4	1	2	6
8		9					9	8	

如图所示，可以发现一些基本的 UR 结构，不过此题没有使用，因为单纯为了有趣才产生了这样的形式。

此题一共包含 10 个真数，分别是 r239c2(1)、r7c7(1)、r5c3(2)、r36c3(3)、r5c2(3)、r8c9(6)和 r5c3(8)。它们全部都能对应到 r7c2(1)，所以 r7c2 <> 1。

Part 9 动态链 (Dynamic Chain)

如果说强制链是链的一种推广形式的话，那么现在要讲到的**动态链 (Dynamic Chain)**就是另外一个层面的链的推广。

动态链强调链可以包含不同的分支，并且分支能够进行汇总，得到想要的结果。这种链可以依赖于 AIC，即强关系开头、强关系结尾，并且强弱关系交替进行的链。

9-1 动态链是怎么运作的？

9-1-1 一则示例

4	¹ ₅	2	6	¹ ₅	3	¹ ₅	7	8
8	¹ ₅	¹ ₉	^{1 2} _{4 5}	7	^{1 2} ₄	3	¹ _{4 5}	6
6	3	7	¹ _{4 5}	8	¹ ₄	2	¹ _{4 5}	4
³ ₇	³ ₇	6	^{1 3} _{4 5}	^{1 3} _{4 5}	¹ ₄	¹ ₅	8	2
⁷ ₃	2	8	^{1 3} ₅	^{1 3} ₅	6	¹ ₅	¹ _{4 5}	9
5	4	¹ ₉	8	2	7	¹ ₉	3	6
9	6	^{1 2 3} _{4 5}	^{1 2 3} ₄	^{1 3} ₄	^{1 2} ₄	8	¹ ₅	7
1	8	3	7	6	5	4	2	9
2	7	¹ _{4 5}	¹ ₄	¹ ₄	8	6	¹ ₅	3

如图所示，我们假设 $r4c2(1)$ 为假，则可以得到 $r6c3(1)$ 为真。此时我们走两个分支方向：

- **方向 1:** 由于 $r6c3(1)$ 真，所以 $r2c3(1)$ 为假， $r2c3(9)$ 为真，可以得到 $r2c6(9)$ 为假。
- **方向 2:** 由于 $r6c3(1)$ 真，所以 $r6c3(9)$ 为假，所以 $r6c7(9)$ 为真， $r1c7(9)$ 假， $r3c8(9)$ 真， $r3c6(9)$ 假。

由于分支起头都是 $r3c6(1)$ 为真，所以不论走哪个分支，最终的结果也应当是同时成立的。所以此时应当由 $r2c6(9)$ 和 $r3c6(9)$ 同时为假。所以此时 $c6$ 只有一处可以填入 9，即 $r4c6$ ，所以城之内 $r4c6 = 9$ 。所以此时链从 $r4c2(1)$ 为假得到了 $r4c6(9)$ 为真，故视作不连续环处理，删除两端的交集，即 $r4c2(9)$ 。

这条链在中途某处出现两个分支方向，并最终通过同时成立的结果来进行归并，并得到最终的结果。我们称带有不同分支的标准链为**动态标准链**（简称**动态链**，**Dynamic AIC**）。当然，强制链分身就带有分支，所以强制链单独有特殊的规定：如果强制链的某个分支上含有不同的分支，此时我们称整个强制链为**动态强制链 (Dynamic Forcing Chain)**。由于动态强制链比强制链更为暴力，所以此处我们就不再举例说明，它的逻辑和动态链其实是完全一

样的效果。

和不连续环一样，它的首尾也是不连续环的删数模式，所以这个链称为**动态不连续环**（**Dynamic Discontinuous Loop**），注意，此处不应书写 Nice 一词。Nice 是否应该书写将在后面进行讨论。

9-1-2 文本表达

和 AIC 一样，动态链依然需要文本的表述格式。不过由于动态链带有分支，所以我们将从分支处用冒号起头，并把每个分支单独换行书写。例如上面的示例的链的表达是

```
r4c2=r6c3(1):
  -r2c3(1)=r2c3-r2c6(9)
  -r6c3=r6c7-r1c7=r3c8-r3c6(9)
=r4c6(9) => r4c2 <> 9
```

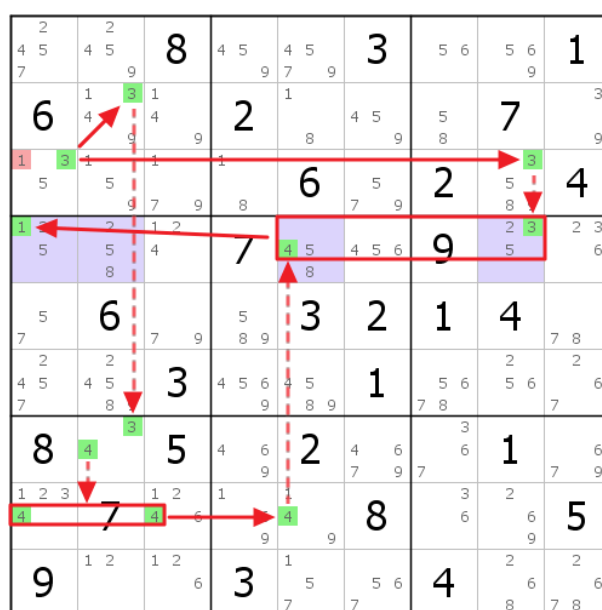
这种格式的书写方法依赖于换行和缩进。如果在不合适的地方多写了缩进，或者是本应该缩进的地方没有缩进，就无法直接判断链的该部分是否应在主线上。

还有一点要注意的是，每行结尾处必须标注出节点涉及的候选数，不能够省略，因为换行后可能为分支路线，我们不能保证它在还原时，应还原为具体是哪一个分支的开头节点的涉及候选数。

另外，由于 r2c6(9)和 r3c6(9)两个分支的末端是弱关系，所以归并的时候，我们将继续使用强关系，故 r4c6(9)应用强关系连接。这也是符合链的规则。

9-2 嵌套结构的动态链（Dynamic Grouped Chain）

9-2-1 嵌套 ALS 的动态链（Dynamic Grouped Chain With ALS）



如图所示，这条链有点小绕。

链的表示如下：

```

r3c1(3):
  =r2c2-r7c2(3=4)-r8c13=r8c5-r4c5(4)
  =r3c8-r4c8(3)
=r4c1(1) => r3c1 <> 1

```

逻辑是这样的：由于链的起始需要设为假，所以设 $r3c1(3)$ 为假，则同时可以得到 $r2c2(3)$ 和 $r3c8(3)$ 为真。两端引出两条不同的链后，分别得到 $r4c5(4)$ 和 $r4c8(3)$ 为假，所以 $r4c1(1)$ 此时必然应为真。所以 $r3c1(3)$ 为假时， $r4c1(1)$ 应为真，所以 $r3c1(3)$ 和 $r4c1(1)$ 至少一个为真，按照不连续环的逻辑，删掉 $r3c1(1)$ 。

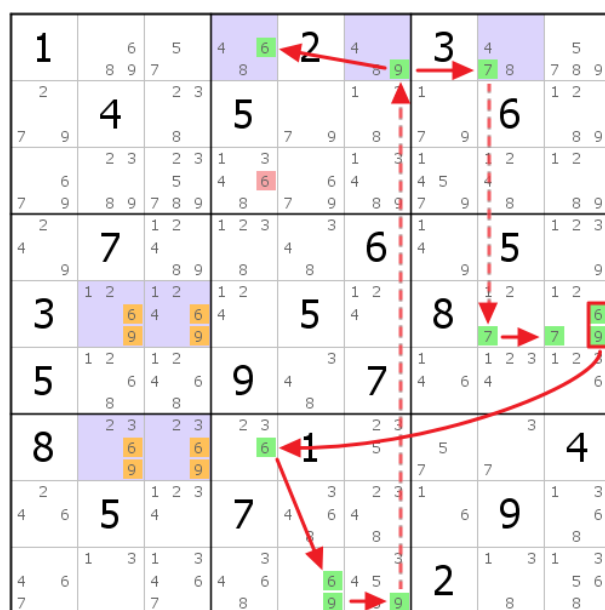
有点疑问。观察 $r4$ ， $r4$ 有多个单元格涂色，但显然不是 ALS，因为数一下单元格数和候选数种类数，发现是相差 2 而不是相差 1 的。

那么，你这么理解一下： $r3c1(3)$ 为假时， $r2c2(3)$ 和 $r3c8(3)$ 同时为真。 $r2c2(3)$ 引出的链得到了 $r4c5(4)$ 为假，而 $r3c8(3)$ 引出的链得到了 $r4c8(3)$ 为假。这也就意味着， $r3c1(3)$ 为假时，可以得到 $r4c5(4)$ 和 $r4c8(3)$ 同为假。如果此时 $r4c1(1)$ 也为假，就意味着这三个数同为假，然后涂色的四格，就都只有 2、5、8 三种候选数，这显然是不够填的，所以 $r4c1(1)$ 此时只能为真。所以它是强关系。

注意一点，这里是 $r4c5(4)$ 和 $r4c8(3)$ 同为假，如果写成 $\{r4c5(4), r4c8(3)\}$ 的话就要注意里面两数是同假的，不要把逻辑搞错，然后推导走向了另外一个错误的方向。比如之前有人就问我，“ $r4c5(4)$ 和 $r4c8(3)$ 这个整体节点为假，应该是破坏 ‘ $r4c5(4)$ 和 $r4c8(3)$ 同时为真’ 的情况，所以应该是 $r4c5(4)$ 和 $r4c8(3)$ 最多有一个为真的意思”。这个说法是错的，因为同为真和同为假并不是对立的。

最后告诉大家，这里嵌入了一个四个单元格，但内部有六种数字的结构。它实际被叫做 **二次待定数组**（Almost ALS，简称 AALS 或 A²LS），就是 ALS 的待定版本。之前没有讲它的原因是这个叫做 AALS 的东西一般不在实战里面出现，或者出现频率极少，而且不好观察到。

9-2-2 嵌套 AUR 的动态链（Dynamic Grouped Chain With AUR）



如图所示，这个链厉害就厉害在它带有一个 AUR。链的表达如下。

```
r7c4(6)=r9c5(6-9)=r9c6-r1c6(9):  
=r1c4(6)  
=r1c8-r5c8(7)=r5c9(7-69)=r7c4(6)  
=> r3c4 <> 6
```

可以看到，这个链在表达上也挺别扭的。假设 $r7c4 \neq 6$ ，可以一直到 $r1c6(9)$ ，此时它为假。

当 $r1c6(9)$ 为假的时候， $r1c468(46789)$ 这个 AALS 区域里就变为了 ALS，因为少了一个数，不过依然不能直接用。此时我们将分两个分支讨论。

- 走**分支 1**，可以得到 $r1c4 = 6$ 的结果，到这里就结束了，我们就不再向下走了；
- 走**分支 2**，可以得到 $r1c8 = 7$ 的结果，然后继续向下得到 $r5c8 \neq 7$ 、 $r5c9 = 7$ 。此时我们这么看。由于 $r5c9(7)$ 为真，所以我们不妨把 $r5c9(69)$ 看作整体，它们均为假，所以类似于区块来说，它们实际上整体节点就是为假的。接着我们发现，如果 $r5c9(69)$ 和 $r7c4(6)$ 同假的话，就会使得 $r57$ 上的所有 6 和 7 都将只能放在 $r57c23$ 里，使得四个单元格变为 6 和 7 的隐性数对，并去掉其它的数字，也就形成了 UR 的致命形式，所以出错了。故它们不同假，即 $r5c9(69)=r7c4(6)$ 是成立的。

不论分支多少成立，我们能保证其中至少有一个分支是对的路线。至于为什么说是至少而不是有且仅有呢，因为在我们推理的过程里的每一个节点都不一定仅通过这一条路径得到结论，它们仅仅是建立在假设成立的前提下可以得到这样的结果罢了，所以其它的情况我们并不清楚。所以至少有一个分支成立，那么结果里至少有一个是对的，即 $r1c4(6)$ 和 $r7c4(6)$ 里有一个是对的，所以删除它们的交集，即 $c4$ 上的其余位置的 6。

此时请注意，我们还漏掉一个情况，就是链头 $r7c4(6)$ 本身成立的情况，但这一点恰好在其中的一条分支上出现了，所以我们不必讨论它，但从严谨和客观的角度出发，实际上应当是取 $r1c4(6)$ 、 $r7c4(6)$ 和 $r7c4(6)$ 的交集，即两个 $r7c4(6)$ ，一个表示分支上的节点，而另外一个则表示链头。

这种“一个头两个尾”的结构有时候被称为**多头链**（Multi-way Chain）。

我们再来看一则示例，不过这则示例需要你自己推理其逻辑。如图所示。

7	1 3	3	4	1 6	3 6	1 3 5 6	1	2
1 5 9	4	8 9	8	1 2 3 6	2 3 6	1 3 5 6	7	5 6 9
1 2 3 8	1 2 3	2 3 6 8	9	7	5	1 3 4 6 8	4 6 8	
3	7	1	6	9	8	2	4 5	4 5
2 9	2 9	4	3	5	1	7 8	6	7 8
6	8	5	2	4	7	9	3	1
2 5 8	2 3 5	2 3	1	2 3 6	9	4 5 6 7	2 4 5 6	4 5 6 7
1 2 5 9	6	7 9	5 7	2 3 6	4	1 5 7	8	5 7 9
4	1 2 5 9	2 7 8 9	5 7	2 6 8	1 6	5 6	1 2 5 9	3

9-3 一些常见的动态链定式技巧

接下来我们来看一些基于动态链的定式技巧。

9-3-1 三国争雄（Tri-W-Wing）

第一个技巧的名字很霸气，而且示例也很新奇。

6 9	5 6 9	3	1 5 6	1 5	2	7	8	4
2	5 6	1	4	7	8	5 6	9	3
4	7	8	6 9	5 6 9	3	5 6	1	2
1	8	6 9	5 6	3	4	2	7	5 6 9
3	6 9	2	7	5 6 9	1	8	4	5 9
7	4	5	2	8	6 9	1	3	6 9
5 8	1	7 6	3	4	5 6	9	2	7 8
6 9	2	4	8	1 6 7 9	6	3	5	1 7
5 8	3	7 9	1 9	2	5 9	4	6	1 7 8

如图所示，它实际上是“三个头”的 W-Wing 结构。由于 W-Wing 的头尾都是一样的形式，所以此处我们随意找一个“头”作为链头开始推导。

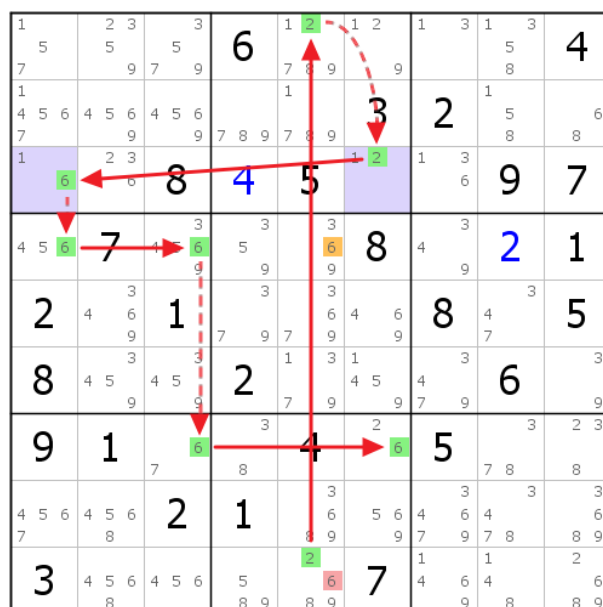
假设 $r4c3(6)$ 为假，则 $r4c3(9)$ 真， $r9c3(9)$ 假。此时可以发现 $r9$ 还剩两个 9，我们分情况进行讨论：假设 $r9c4(9)$ 是成立的，那么向上推导能够得到 $r3c4(6)$ 为真；但如

果假设 $r9c6(9)$ 成立的话，那么向上推导可以得到 $r6c6(6)$ 为真。

所以实际上这个问题被转化为了前一节讲到的三头链问题了，删数是删它们的交集，即 $r4c3(6)$ 、 $r3c4(6)$ 和 $r6c6(6)$ 的交集，即 $r4c4(6)$ 。

这个技巧被叫做**三国争雄**（Tri-W-Wing）。

9-3-2 毛刺链（Kraken Region | Cell）



如图所示。链表示如下：

```
r9c5=r1c5-r3c6(2)=r3c1-r4c1(6):
=r4c5(6)
=r4c3-r7c3=r7c6(6)
=> r9c5 <> 6
```

这条链的删数逻辑是，观察 $r4c5(6)$ ，设它为真时，可以删除 $r9c5(6)$ ；设它为假时，引出链，以 $r9c5(2)$ 起头，得到 $r7c6(6)$ 为真，依然可以删除 $r9c5(6)$ 。所以 $r9c5(6)$ 可以删除。

这个结构叫做**毛刺链**（Kraken Region | Cell）。

9-4 动态链的逆向视角

可以看到，动态链是具有分支的，那么既然有分支，就逃避不了一个问题，如何逆向思考动态链的逻辑。

实际上，动态链的逆向视角采用的是逆否命题来得到。我们先思考，动态链分哪些情况。实际上，动态链之分两种：有一种是内部有分支，但最终汇总了；还有一种是没有汇总，形成了多头链。第一种链实际上逆否不会影响到中间的推导逻辑，因为链的本质是看头尾的删数交集，而中间在删数层面是不关心的（当然，环除外）。所以第一种情况我们不需要细致

的说明。而第二种不同，它是多个头，这就得分方向了。那么我们简单来使用数理逻辑的语言证明一下。

假设这个三头链，链头有一个，链尾有两个。那么它的逻辑是“如果链头为假，则两个链尾至少有一个为真，则删除它们三个端点的交集”。

那么，如果这个链逆向推导呢，逻辑又是如何的？先用逻辑语言把叙述改一下。先设“链头为真”叫 A，“第一个链尾为真”叫 B，“第二个链尾为真”叫 C。则叙述应该是这样：

$$\neg A \rightarrow (B \vee C)$$

若非 A，则 B 或 C。

将链逆向，则就是逆否命题，即

$$\neg(B \vee C) \rightarrow A$$

若非 (B 或 C)，则 A。

然后，借用德·摩根律拆开括号。

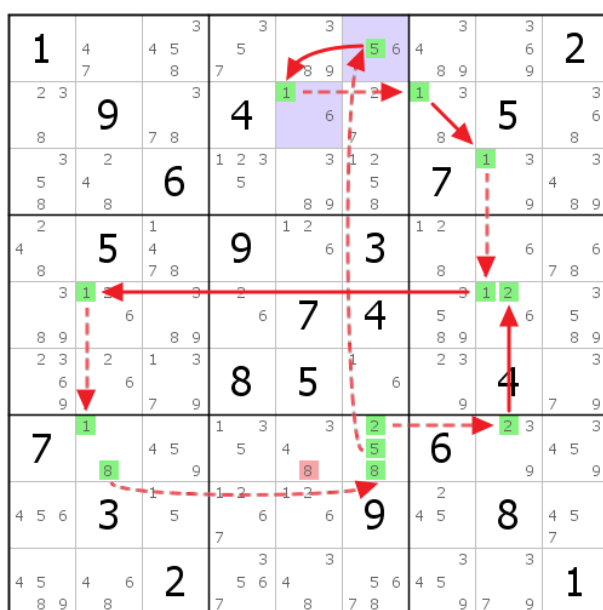
$$\neg B \wedge \neg C \rightarrow A$$

若非 B 且非 C，则 A。

转为自然语言，就是**两个链尾同时为假时，得到链头为真时，则可以删除它们三个端点的交集**。那么，更多情况可以自己试试看，实际上逻辑和推导方式完全和上面的一样。这里就不再讲解了。

9-5 动态环 (Dynamic Continuous (Nice) Loop)

9-5-1 一个基本的示例



如图所示，链写法如下：

动态环：

$r7c2(1=8)-r7c6(8)$ ：

$=r7c6-r7c8=r5c8(2)$

$=r7c6-r1c6(5=6)-r2c5(6=1)-r2c7=r3c8(1)$

$-r5c8=r5c2-r7c2(1) \Rightarrow r7c5 \neq 8$

它的逻辑大致是这样的：设 $r7c2(1)$ 为假，则得到 $r7c6(8)$ 为假时分两种情况讨论，一个是 $r7c6 = 2$ ，另外一个则是 $r7c6 = 5$ 。但不管 $r7c6$ 是数字 2 还是 5，都能得到 $r5c8(1)$ 为假，于是又可以得到 $r5c2(1)$ 为真、 $r7c2(1)$ 为假。这样就首尾拼接成环了。

可是稍微奇怪的是， $r7c6$ 分了两种情况讨论，也就是这里产生了分支，那么，这个环的删数仍然还是所有弱关系对应位置吗？

想一下逻辑，如果把逻辑抽象出来，就是下面这个样子：

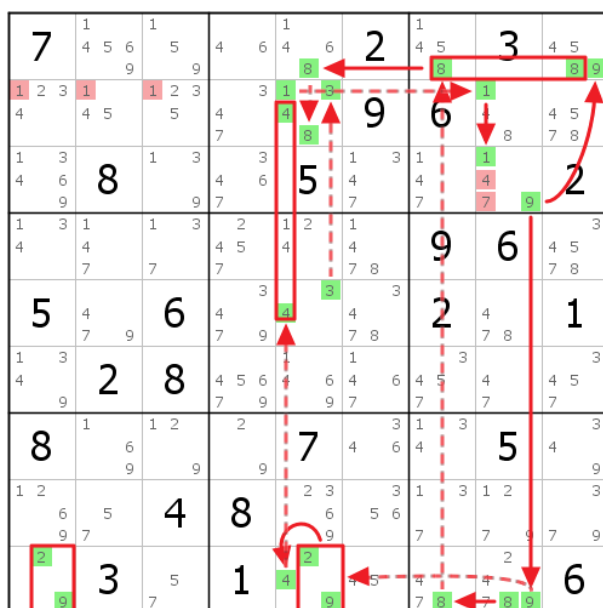
$$A = \begin{cases} B - C \\ D - E \end{cases} = F - A$$

首先，大体下， $F-A$ 部分是可以删除的，不过 $B-C$ 和 $D-E$ 两处是两种不同的情况，它们是“或”的关系（注意“或”的关系是指两个部分至少有一个为真，而不是有且仅有一个为真），所以我们无法判断到底 $B-C$ 还是 $D-E$ 是一直都可以的，也就不能找到分支条件下的删数。所以总体来看，删数只有 $F-A$ 。

对于图上来说，只有 $r5c2(1)-r7c2(1)$ 和 $r7c2(8)-r7c6(8)$ 两处弱关系可以删数。

这个结构称为**动态标准环**（简称**动态环**，**Dynamic Continuous (Nice) Loop**）。有时候也将这种带有两分子环形式的结构称为**双环**（**Double Loop**）。

我们再给出一个示例，请自行推理其推进过程。



这个链的删数特别奇怪，所以请注意实际的推导过程。这个环的理解比较难，希望你能慢慢地、细致地注意到每一个细节。

9-5-2 原理进一步剖析

9-5-2-1 #1: 动态环的删数是否有定式?

动态环的删数实际上在前面已经说了，所以我们可以在这里直接总结出来了。所以答案肯定是有删数的定式的。而且还比较明显。如果是读者你的话，不用看这一节的问题，自己思考一段时间，我想你也可以想到它——**所有主干上的弱关系的区域都可以删数，而分支上的弱关系不可以。**

可以从例子里看到，分支之间是或的关系，所以显然不可以删数，我们无法断言出删数一定能产生在分支上。所以不可以。主干上的弱关系有时候会很多，但有时候会少得可怜，看看上面的例子你就明白我在说什么了。

9-5-2-2 #2: 英文名里的 Nice 有没有很重要吗?

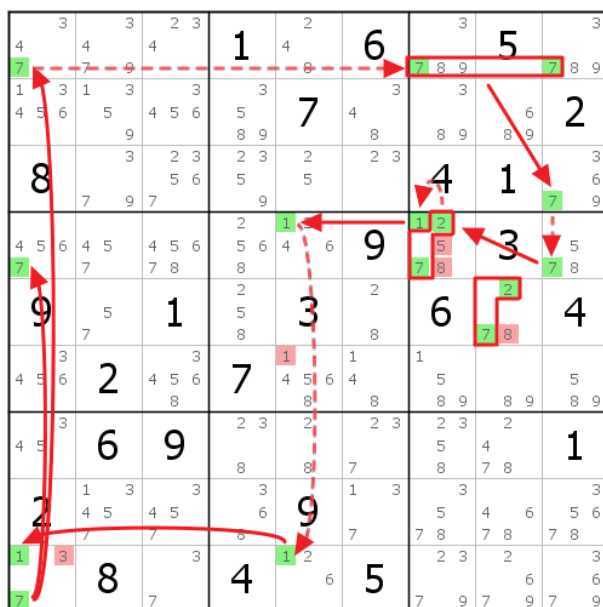
是的。在 **Nice** 的原文描述里提到，“当且仅当环内的每一个弱关系都可以删数时，称为 **Nice Loop**；否则称为 **Loop**”。换句话说，如果我们现在有一个环，不论动态环还是标准环，如果环的每一个弱关系都可以实现删数（即使它在这个题目里找不到删数，但理论上可以删的话），就称为这个环是 **Nice** 的；否则就不能使用 **Nice**。

可以看到，任意一个标准环都一定是 **Nice** 的，因为标准环的特征就是每一个弱关系都可以视为强关系，所以它们所在的区域上一定会出现一个节点包含需要填的数字，所以每个弱关系的对应区域都一定是可以删的；而动态环不同，由于动态环具有分支，所以分支上的弱关系我们并不能保证是否可以删除，所以动态环里不一定所有的环都能加上 **Nice** 一词。实际上，大多数动态环都无法实现全部弱关系都可以删除，只有很少一部分可以实现，所以大部分的动态环都不带 **Nice** 一词，即 **Dynamic Continuous Loop**；而少部分动态环才称为 **Dynamic Continuous Nice Loop**。所以前面给出的示例是不能加 **Nice** 的，因为它还有分支上的弱关系不能删数。

9-5-3 嵌套结构的动态环 (Dynamic Grouped Continuous (Nice) Loop)

说完了基本的动态环的删数原则，我们来看一些嵌套结构的动态环结构。

9-5-3-1 动态环 (Dynamic Grouped Continuous Loop)



如图所示，这个链内嵌入了毛刺隐性数对结构。链的写法如下：

```
r9c5=r9c1(1-7):  
=r4c1-r4c9(7)  
=r1c1-r1c79=r3c9-r4c9(7)  
={r4c7, r5c8}(27)-r4c7=r4c5-r9c5(1)  
=> r9c1 <> 3, r6c5 <> 1, r4c7, r5c8 <> 58
```

这个题的推导和刚才的动态环差不多。删数也是找总体下的弱关系。总体下的弱关系有 $r9c1(1-7)$ 、 $\{r4c7, r5c8\}(27)-r4c7(1)$ 、 $r4c5(1)-r9c5(1)$ 这三个。那么删数也就在这里面去找。需要提到的地方就是这里的 $r4c7$ 和 $r5c8$ 的 2、7 隐性数对的删数了。我们之前说到的是，删数是看两端都能删的地方，虽然这个 2、7 隐性数对确实可以删 $r5c8(8)$ ，但是 $r4c7(1)$ 为什么也能保证 $r5c8(8)$ 可以删呢？因为此时 $r4c7(1)$ 和 $r5c8(8)$ 都是弱关系。如果 $r4c7(1)$ 和 $r5c8(8)$ 同真时，b6 内数字 2 没有位置可填。所以 $r4c7(1)$ 和 $r5c8(8)$ 确实是弱关系，也就确实是可以删的。

这个题目带有一个 ALS，所以称为**嵌套 ALS 的动态环** (Dynamic Grouped Continuous (Nice) Loop With ALS)。同样，此题不加 Nice。

我们再来看一个动态环。

9-5-3-2 动态环 (Dynamic Grouped Continuous Nice Loop)

获取你注意到了，前后两个示例的标题大致一样，就是英文名里，一个有 nice，一个没有。下面我们来尝试理解一下它。

如图所示。这个动态环比较复杂。我们先从 $r2c2(2)$ 开始设为假进行推理。

如果 $r2c2(2)$ 为假，而此时 $r2c28$ 是一个 ALS 区域，所以 $r2c8(3)$ 必须此时为真才行。那么此时，由于 $r2c8 = 3$ 的缘故， $r2c5$ 和 $r2c9$ 都不能是 3，所以此时 $r2c5(3)$ 和 $r2c9(3)$ 是同时为假的。我们先尝试走左边。

假设 $r2c5(3)$ 为假，则 $r23c6(3)$ 区块为真（当然，这里一般是当得到 $r2c8(3)$ 为真后， $r2c56(3)$ 同为假，于是 $r3c6(3)$ 为真，也可以这么理解）， $r5c6(3)$ 为假， $r5c1(3)$ 为真， $r6c2(3)$ 为假；而假设 $r2c9(3)$ 为假，则 $r2c9(7)$ 为真， $r4c9(7)$ 为假， $r6c7(7)$ 为真， $r6c2(7)$ 为假。

此时可以发现，由于刚才假设的情况的成立，我们先得到了 **r2c8(3)**为真的结果，并同时得到了两个分支的走向。可是神奇的地方在于，两个分支竟然最终汇聚到了同一个单元格：**r6c2**。

由于是两个分支的推导同时通过一个结论得到,所以两个不同分支得到的结论就应当是同时成立的,即这里的 **r6c2(3)** 为假和 **r6c2(7)** 为假此时必须是同时成立的,所以, **r6c2** 此时只能放入 2, 故 **r6c2(2)** 此时为真, 于是 **r2c2(2)** 为假, 环便出现了。

那么，可以发现到的是，主线肯定是 $r6c2-r2c2(2)=r2c8(3)$ 这一部分，但剩下的部分，都在分支上。按照原本的推导思路，主线上的所有弱关系是可以删除的，并且由于 ALS 区域处于主线上，所以 ALS 的额外删数： $r2$ 的其余位置的 5 也可以删除掉。不过，这个例子里我们可以删除太多其它的删数了，这些又是为什么呢？试想一下，这个结构带有两个不同的分支，但这两个不同的分支的特殊之处在于，它们以同一个节点延伸出去，这意味着两个分支应该是同时成立的，这一点我已经在前文里说过了。而在环结构的结论里，我们知道，环结构的内部填数只可以有两种情况；而在动态链里，分支里可能有不同的情况，所以不一定只有两种。但仔细观察这个示例就可以发现，这个分支是同时成立的，所以要么分支 1 成立，要么分支 2 成立。我们不必注意分支内部的填数模式，这两种情况我们便可直接作为整体动态环的两种不同的填数模式。

而环只有两种填数模式,且弱关系都可以删除,所以我们可以认为,这两个分支上面的所有弱关系都可以作为删除项,故两个分支的弱关系完全都可以作为这里所说的删除项而删

掉。

可以看到，这个示例里，所有的弱关系全部都可以被删除，所以我们应为这个结构的技巧名加上 nice 一词，即**动态的**（dynamic）、**嵌套结构的**（grouped）、**连续的**（continuous）、**删数完整的**（nice）**环结构**（loop）。

9-6 最后来个练习？

之前我们已经给出了三道鱼图题目了，这里再来一道题。请找出所有不能填入的位置（可以删除的位置）。

/							/
/							/
/							/
/				/			/
/							
							/
			/	/			
/			/	/	/		/
/				/	/		/

Part 10 构造链与构造结构

这一节我们将学习到更奇特的使用，**构造**（Construct）。链的构造不仅仅是一种解题技巧，更是一种观察技巧，它不仅使得我们做题更加有逻辑性，而且还能更快地让我们观察到一些特殊的结构，并加以使用到链之中。但是内容有趣且具有挑战性。

10-1 什么是构造？

构造是一种思想，以某一种技巧作为框架，使得我们寻找的结构基于框架而产生不同的变体形式。比如 W-Wing，我们可以在 W-Wing 里嵌入区块节点，也可以使得 W-Wing 的两头强关系从双值格改为 ALS 区域的强关系。这样风格的内容基于原技巧，而不同于原技巧，这便是构造，例如我们之前实际上已经学到过一些技巧，比如 ALS-W-Wing，这便是 W-Wing 的构造版本；我们还学过死亡绽放，它便是 ALS 版本的分支匹配（即 Wing）逻辑。

接下来我们要讲到的就是基于该原则产生的不同构造逻辑。但能够为了产生丰富的解题思路，我们必须得先重新学习一些我们以前学过的技巧，重新学习它们的新用途。

10-2 基于致命结构的构造（Deadly Pattern+）

我们先来学习一些基本的技巧的构造。

对，你没有看错，构造这个词语实际上并没有对应的技巧名称，而是直接在原技巧名后尾随一个加号“+”来表示构造的技巧。比如下面的这些。

10-2-1 UR + XY-Wing

5 8	1 2	7	3	4	6	1 2 5	9	2 8
6	5 8	4	1	9	2	3	5 7 8	7 8
3	1 2	9	7	5	8	1 2 6	1 6	4
2	5 7	1	9	3	4 7	8	4 5	6
4	3	6	5	8	1	7	2	9
5 7 9	5 7 9	8	2	6	4 7	1 5	1 4 5	3
1	6 8	2 3	4	7	9	2 6	3 6 8	5
7 9	7 6 9	5	8	2	3	4	6 7	1
7 8	4	2 3	6	1	5	9	3 7 8	2 7 8

如图所示，首先我们可以知道，如果 r13c7 的 5 和 6 同假时，就会出现 1 和 2 的 UR 致命形式，所以不同假。所以 r1c7(5)和 r3c7(6)至少一真：如果 r1c7(5)为真，则 r6c7=1；而如果是 r3c7(6)为真，则 r3c8 = 1。

因为两者至少一个为真，也就意味着 $r6c7$ 和 $r3c8$ 至少一个是 1。所以删掉它们的交集，即 $r6c8(1)$ 。

由于结构里嵌套了 XY-Wing 的分支逻辑，所以称为 **UR + XY-Wing**。

10-2-2 UR + XYZ-Wing

6	7	1	8	3	²	²	5	²
9	5	3	^{1 2}	4	7	^{1 2}	6	²
2	4	8	5	6	¹	^{1 3}	^{1 3}	
5	^{1 3}	2	⁴	9	⁴	^{1 3}	^{1 3}	6
	¹	9	3	5	6	^{1 2}	¹	²
⁷	⁸	⁸	6	4	²	9	³	5
⁴	³	5	9	7	⁴	6	2	1
¹	³	9	7	6	2	5	⁴	⁴
¹	2	6	¹	8	5	7	9	3

如图所示，我们把 $r5c89$ 的额外的数字分三种情况讨论：1、2、7。虽然 7 有两个，但是我们依旧可以按照区块的形式来讨论。显然，同假时会导致 UR 出现致命形式，所以必须得有额外的情况成立。所以我们挨个讨论一遍。

- **假设 1 是对的**，那么我们可以得到 $r5c2 = 8$ ；
- **假设 2 是对的**，那么我们可以得到 $r2c9 = 8$ ；
- **假设 7 是对的**，不论哪个 7 正确，都可以得到 $r5c1 = 8$ 。

所以，不论如何我们都可以得到的是， $r5c12$ 和 $r2c9$ 里必须有至少一个 8 出现。所以 $r5c9$ ，即它们的交集，不能出现 8，于是删除掉它。

这个结构称为 **UR + XYZ-Wing**。

10-2-3 UR + SDC

1 5	1 2 3 6	8	4	2 3 5	2 5 6 9	5 6	7	2 5 6 9
9	2 4 6 7	2 4 6 7	2 5 6 7	8	2 5 6 7	3	1	2 4 5 6
4 5 7	2 3 4 6 7	2 3 4 6 7	1	2 3 5	2 5 6 9	8	2 4 9	2 4 5 6 9
2	3 6 8	3 6	5 6 8	4	1	7	3 9	5 9
1 4	9	1 4	2 5	7	2 3 5	1 4 5	6	8
7	1 3 4 6 8	5	6 8	9	3 6	1 4	2 3 4	2 3 4
3	2 4	2 4	7	6	8	9	5	1
6	5	9	3	1	4	2	8	7
8	1 7	1 7	9	2 5	2 5	4 6 4	3 4	3 6

如图所示，我们发现如果 r2c23(67)均为假的话，就会出现 UR 致命形式。而同真的话，就是 6、7 数对，r2c46 没有了 6 和 7，b2 内 2、3、5 的位置要填四格，所以不够。所以，r2c23 只有一格只有{67}，另外一格则为{24}。这样一来，就会和 r2c46 其中一格构成 6、7 数对，剩下一格则和 r13c5 构成 2、3、5 的三数组。所以，删除掉 b2 内其余位置的 2、3、5，删除掉 r2 其余位置的 6、7。

这个结构则是 UR 带了个 SDC，所以我们称为 **UR + SDC**。

10-3 基于链的构造（AIC+）

10-3-1 链的构造原理

思考一下，一般链的头和尾都会有什么关系。显然，链头和链尾至少有一个成立，这是链的原则，但实际上，我们可以立马考虑到的是，既然是至少有一个成立，换个说法也就是不同假，那不就是强关系？是的。所以我们可以把一条链缩写为一个强关系，缩写的原则就是首尾不同假（即原来的说法“至少有一个节点成立”）。所以我们为首尾画出强关系即可，我们先来看一则示例来理解这一点。

3 6	2	7	6 9	1	3	5	8	6 9	4
3	1	3	7	6 9	3	4	6 9	3	1
5	6	5	7	8	9	7	6	4	9
4	1	3	7	6 9	2	8	6 9	7	3
5	3	5	2	7	8	6	7	4	6
1	7	8	6	5	9	3	2	4	6
9	6	4	7	8	2	1	5	3	7
2	4	5	3	6	8	9	7	1	6
6	8	6	1	9	5	7	4	2	3
7	9	3	4	1	2	6	5	8	5

如图所示，我们可以看到这是一个很普通的链，但没有删数，因为链的头尾对应到的是 r4c123，但不巧的是，这几个单元格都没有 6。

不过没关系，我们可以知道的是，链如果是成立的，那么的头尾是为强关系的，那么我们仅需要一个强关系，便得到如下的情况：

3 6	2	7	6 9	1	7	3	5	8	6 9	4
3	1	3	7	6 9	4	3	4	6 9	7	3
5	6	5	7	8	9	7	6	4	9	7
4	1	3	7	6 9	2	8	6 9	7	3	1
5	3	5	2	7	8	6	7	4	6	1
1	7	8	6	5	9	3	2	4	6	8
9	6	4	7	8	2	1	5	3	7	8
2	4	5	3	6	8	9	7	1	6	4
6	8	6	1	9	5	7	4	2	3	7
7	9	3	4	1	2	6	5	8	5	8

可以看到，这个链仅长度为 3，其中的第一个强关系产生于刚才我们得到的链。所以实际上，这条链的头尾的交集现在就已经有了：r2c3(6)，所以 r2c3 <> 6。

这就称为**链的构造**（或**构造链**，Construct of AIC，记作 AIC+）。

但是，好像这样仅仅是简化了链而已（甚至是没有简化，反而增大了逻辑的复杂度，因为一条链被拆成了两条），并没有产生任何实质性的效果。这你就错了，因为构造链并不像看起来这么简单。

实际上，我们在寻找链的时候，经常碰到类似于第一个图上那样的情况：好不容易找到

了一条链，结果发现遗憾的是，它并没有删数，所以我们就此放弃了它。实际上并不需要放弃，而是我们继续通过这一条链的头尾强关系来得到一些新颖的结论，比如嵌入一些技巧结构，就像下面这些技巧一样。

所以不要担心它就像看起来那样无用。

10-3-2 XY-Wing 构造 (XY-Wing+)

XY-Wing 实际上内容并不多，但它的使用在我们平时日常使用之中非常多，它可以使用强制的视角对一个单元格进行逐个的讨论；也可以使用链的视角将其拉伸成一条链，来得到删数；甚至还可以使用伪数组来理解。而且结构非常容易观察，所以我们先要学习的就是关于它的构造。

我们先来思考一下，XY-Wing 到底能得到什么。我们套用 AIC+ 的逻辑，可以得到对应的 XY-Wing+。

1	6	2 3	1 2	7	4	5	1	3	1	3	8
	9		9		9		4	6	4	6	
7	6		3	8	1	2	4	6	5	9	3
1	5 6		1		3			6	1		4 6
7	9	4	5	9	3	8 9	8	6	7	6	2
1	4 5	5	3	9	6	4	1		7	2	
		8				8					
4	7		6	2	4 5	1	9	4	3	4 5	3
	7 8				8						
2	9	1	4	4 5	3	7	8	1	4	6	4 5 6
8	1	4	2	6	4	3	4	2	5	4	9
		9			7		7			7	9
4 5	6	7	4 5	1	2	4	3	8	4	3	
	9									9	
3	2	2	4 5	8	4 5	9	1 2	1	1	4 6	6
	5			7			4	6	4	6	7

如图所示，我们找到了一个 XY-Wing，而它没有删数。不过没关系，没有删数并不影响这个链的成立。既然 XY-Wing 是成立的，那么我们就可以得到的是，链的头尾是称强关系的，所以我们只需要轻松地将 r4c2(5)和 r6c4(5)用强关系连起来就好了。接着，我们的工作就是利用这个强关系来架起桥梁，产生删数。幸运的是，我们确实发现了一个链，如下图所示。

1	6 9	2 3	1 2	7	4 9	5	1 4	3 6	1 4	3 6	8
7	6	3	8	1	2	4 6	5	9	4 7	3 6	
1 7	5 6 9	4	1 5 9	3	8 9	6 8	1 7	6	2	1 6 7	
1 4	5	5 8	3	9	6	4 8	1 4	7	2		
4 7	7 8	6	2	4 5	1	9	4	3 4 5	3		
2	9	1 4	4 5	3	7	8	1 4	6	1 4 5 6		
8	1	2 4 9	6	4 7	3	4 7	2 5	5	4 7 9		
4 5 9	6	7	4 5	1	2	4	3 8	8	4 9		
3	2 5	2 4 5	8	4 7	5	9	1 2 4 6 7	1 4	6 7	6	

可以看到，这条链实际上就是一个多宝鱼（即双强链），不过这个双强链的第一个强关系产生自一个 XY-Wing 结构。

这种形式的 XY-Wing，我们套用了链的逻辑将 XY-Wing “发扬光大”，这样的形式我们称为 **XY-Wing 的构造**（或**构造 XY-Wing**，记作 **XY-Wing+**）。

不过，有些时候我们也可以称为是双强链的构造，不过侧重点不同。如果是双强链的构造，此时我们针对的就不会是前面的 XY-Wing 了，而是后面的这个双强链了，比如之前的 ALS-W-Wing，实际上就是 W-Wing 的构造，而我们可以称为是 ALS 的构造，这一点我们没有必要着重去具体区分到底是谁的构造。

不过，我们一般认为，如果你从技巧的框架出发，那就称为这个技巧的构造。比如 ALS-W-Wing 技巧，我们一般称为是 W-Wing 的构造，因为它更侧重是从 W-Wing 的框架出发的；而上面这一则示例里我们更着重的是从 XY-Wing 的框架出发的，所以称为 XY-Wing 的构造。

10-3-3 XYZ-Wing 构造（XYZ-Wing+）

虽说 XYZ-Wing 跟 XY-Wing 就差了一个“鳍”，但实际上，XYZ-Wing 的构造却比 XY-Wing 要有趣一些。首先我们拿出一则 XYZ-Wing 的示例。

5	² ₈	² ₈	1	7	3	9	4	6
1	7	6	9	4	2	5	3	8
3	4	9	5	6	8	7	2	1
6	³ ₈	⁴ ₈	2	¹ ₆	³ ₄	¹ ₄	9	5
9	³ ₅	7	³ ₆	¹ ₆	³ ₄	¹ ₄	8	2
2	1	⁴ ₅	8	⁵ ₉	⁴ ₉	⁴ ₆	7	3
8	9	1	7	2	5	3	6	4
4	² ₅	² ₅	³ ₆	³ ₉	⁶ ₉	8	1	7
7	6	3	4	8	1	2	5	9

可以看到，这就是一个普通的 XYZ-Wing，不过这个 XYZ-Wing 我们如果试着找到链的开头和结尾的话，就比较麻烦了。我们只能知道的是，我们应当在这 3 个橙色的 3 里寻找链头和链尾，但如何分配链头和链尾就变为了一个比较难受的事情。

实际上，不论怎么分配，都是可以的。例如下面的两种情况：

5	² ₈	² ₈	1	7	3	9	4	6	
1	7	6	9	4	2	5	3	8	
3	4	9	5	6	8	7	2	1	
6	³ ₈	⁴ ₈	2	¹ ₆	³ ₄	7	¹ ₄	9	5
9	³ ₅	7	³ ₆	¹ ₆	³ ₄	¹ ₆	8	2	
2	1	⁴ ₅	8	⁵ ₉	⁴ ₉	⁴ ₆	7	3	
8	9	1	7	2	5	3	6	4	
4	² ₅	² ₅	³ ₆	³ ₉	⁶ ₉	8	1	7	
7	6	3	4	8	1	2	5	9	

5	² ₈	² ₈	1	7	3	9	4	6	
1	7	6	9	4	2	5	3	8	
3	4	9	5	6	8	7	2	1	
6	³ ₈	⁴ ₈	2	¹ ₆	³ ₄	7	¹ ₄	9	5
9	³ ₅	7	³ ₆	¹ ₆	³ ₄	¹ ₆	8	2	
2	1	⁴ ₅	8	⁵ ₉	⁴ ₉	⁴ ₆	7	3	
8	9	1	7	2	5	3	6	4	
4	² ₅	² ₅	³ ₆	³ ₉	⁶ ₉	8	1	7	
7	6	3	4	8	1	2	5	9	

可以看到，这两种形式实际上就是很简单地把其中两个节点放到一起作为一个广义的区块节点处理，然后三个数就能重组为两个节点，形成强关系了；而且，这两种画法实际上也符合链的视角。我们拿左图举例。如果 r5c2(3)和 r45c5(3)同时为假的话，则相当于三个 3 全部去掉，于是在结构里只剩下 1 和 5，虽然是拐弯的，无法形成 ALS，但你可以看到很显然的是，1 和 5 都没有跨区，而占据了三个单元格，这就意味着 1 和 5 只能填入到其中两个单元格里，而总会剩下一个单元格无法填数，这便产生了矛盾。所以，强关系是成立的。

同理，右图的强关系和左图的证明思路完全一样。不过……总会存在一种情况，会用到下面的这种形式的东西。

5	² ₈	² ₈	1	7	3	9	4	6	
1	7	6	9	4	2	5	3	8	
3	4	9	5	6	8	7	2	1	
6	³ ₈	⁴ ₈	2	¹ ₆	³ ₄	7	¹ ₄	9	5
9	⁵ ₆	7	³ ₆	¹ ₅	³ ₄	¹ ₆	8	2	
2	1	⁴ ₅	8	⁵ ₉	⁴ ₉	⁶ ₄	⁶ ₆	7	3
8	9	1	7	2	5	3	6	4	
4	² ₅	² ₅	³ ₆	³ ₉	⁶ ₉	8	1	7	
7	6	3	4	8	1	2	5	9	

如图所示。实际上这种构型依然是成立的。虽说跨区的两个 3 形成一个节点的时候，得考虑是否填数重复的情况，但实际上在 XYZ-Wing 里，我们无需考虑，因为我们考虑的是同假时的情况，而不是同真时的情况。既然是为假，那么显然就只有都不填的这种情况了。

讲完了上述的逻辑后，我们来看 XYZ-Wing+的用法。

6	¹ _{4 5}	¹ ₅	¹ _{5 8}	2	3	⁵ ₉	^{4 5} _{8 9}	7	6	¹ _{4 5}	¹ _{5 8}	2	3	⁵ ₉	^{4 5} _{8 9}	7	
² ₇	9	¹ ₅	¹ _{5 7 8}	¹ _{4 5}	¹ _{5 7 8}	^{2 3} ₅	^{1 2 3} _{4 8}	6	² ₇	9	¹ ₅	¹ _{5 7 8}	¹ _{4 5}	^{2 3} ₅	^{1 2 3} _{4 8}	6	
² ₇	¹ _{4 5}	³ ₈	9	¹ _{4 5 6}	¹ _{5 6}	^{2 3} ₅	^{2 3} _{4 5}	^{1 2 3} ₄	² ₇	¹ _{4 5}	³ ₈	9	¹ _{4 5 6}	¹ _{5 6}	^{2 3} ₅	^{2 3} _{4 5}	^{1 2 3} ₄
9	² ₇	4	¹ ₇	³ ₆	8	¹ ₇	^{1 2 3} ₆	5	9	² ₇	4	¹ ₇	³ ₆	8	¹ ₇	^{1 2 3} ₆	5
3	8	⁵ ₇	2	¹ ₅	4	¹ _{7 9}	^{7 9}	6	3	8	⁵ ₇	2	¹ ₅	4	¹ _{7 9}	^{7 9}	6
1	² ₇	⁵ ₆	6	³ ₇	9	⁵ ₇	^{2 3} ₄	8	1	² ₇	⁵ ₆	6	³ ₇	9	⁵ ₇	^{2 3} ₄	8
8	¹ ₇	³ _{7 9}	¹ _{5 6}	¹ _{5 6}	2	4	⁵ ₇	³ ₉	8	¹ ₇	³ _{7 9}	¹ _{5 6}	¹ _{5 6}	2	4	⁵ ₇	³ ₉
4	6	^{7 9}	⁵ ₈	⁵ _{8 9}	⁵ ₇	⁵ _{8 9}	¹ _{2 3}	1	4	6	^{7 9}	⁵ ₈	⁵ _{8 9}	¹ _{2 3}	1	^{8 9}	^{2 3}
5	¹ ₃	^{1 2 3} ₉	4	7	¹ _{8 9}	6	^{2 3} ₈	^{2 3} _{8 9}	5	¹ ₃	^{1 2 3} ₉	4	7	¹ _{8 9}	6	^{2 3} ₈	^{2 3} _{8 9}

6	¹ _{4 5}	¹ ₅	¹ _{5 8}	2	3	⁵ ₉	^{4 5} _{8 9}	7	6	¹ _{4 5}	¹ _{5 8}	2	3	⁵ ₉	^{4 5} _{8 9}	7	
² ₇	9	¹ ₅	¹ _{5 7 8}	¹ _{4 5}	¹ _{5 7 8}	^{2 3} ₅	^{1 2 3} _{4 8}	6	² ₇	9	¹ ₅	¹ _{5 7 8}	¹ _{4 5}	^{2 3} ₅	^{1 2 3} _{4 8}	6	
² ₇	¹ _{4 5}	³ ₈	9	¹ _{4 5 6}	¹ _{5 6}	^{2 3} ₅	^{2 3} _{4 5}	^{1 2 3} ₄	² ₇	¹ _{4 5}	³ ₈	9	¹ _{4 5 6}	¹ _{5 6}	^{2 3} ₅	^{2 3} _{4 5}	^{1 2 3} ₄
9	² ₇	4	¹ ₇	³ ₆	8	¹ ₇	^{1 2 3} ₆	5	9	² ₇	4	¹ ₇	³ ₆	8	¹ ₇	^{1 2 3} ₆	5
3	8	⁵ ₇	2	¹ ₅	4	¹ _{7 9}	^{7 9}	6	3	8	⁵ ₇	2	¹ ₅	4	¹ _{7 9}	^{7 9}	6
1	² ₇	⁵ ₆	6	³ ₇	9	⁵ ₇	^{2 3} ₄	8	1	² ₇	⁵ ₆	6	³ ₇	9	⁵ ₇	^{2 3} ₄	8
8	¹ ₇	³ _{7 9}	¹ _{5 6}	¹ _{5 6}	2	4	⁵ ₇	³ ₉	8	¹ ₇	³ _{7 9}	¹ _{5 6}	¹ _{5 6}	2	4	⁵ ₇	³ ₉
4	6	^{7 9}	⁵ ₈	⁵ _{8 9}	⁵ ₇	⁵ _{8 9}	¹ _{2 3}	1	4	6	^{7 9}	⁵ ₈	⁵ _{8 9}	¹ _{2 3}	1	^{8 9}	^{2 3}
5	¹ ₃	^{1 2 3} ₉	4	7	¹ _{8 9}	6	^{2 3} ₈	^{2 3} _{8 9}	5	¹ ₃	^{1 2 3} ₉	4	7	¹ _{8 9}	6	^{2 3} ₈	^{2 3} _{8 9}

如左图所示，我们可以看到，这是一个 XYZ-Wing，不过没有删数。不过不用担心，我们继续找到一条链，可以利用上这里面的三个 7 的强关系。确实我们找到了一个不太长的链，它用上了 XYZ-Wing 产生的强关系，只是长相有点丑陋。

$r6c4=r4c46-r4c2=r6c26(7) \Rightarrow r6c8 \neq 7$

我们便得到了正确的删数。

10-3-4 W-Wing 构造 (W-Wing+)

2	6	4	6	9	7	5	4	6	3	1	2	6	4	6	9	7	5	4	6	3	1
5	1	3	6	8	4	7	2	9	5	1	3	6	8	4	7	2	9	5	1	3	6
7	6	9	4	6	2	3	1	5	4	6	4	7	8	9	2	3	1	5	4	6	4
4	8	5	1	9	2	6	3	7	6	4	2	4	8	5	1	9	2	6	3	7	6
1	2	6	7	9	5	3	1	2	4	6	8	4	1	2	6	7	9	5	3	1	2
3	2	1	6	7	4	8	1	2	6	9	5	3	2	1	6	7	4	8	1	2	6
1	6	4	2	8	1	6	7	9	5	3	1	6	4	2	8	1	6	7	9	5	3
1	5	8	3	1	2	9	4	2	1	4	6	7	1	5	8	3	1	2	9	4	2
9	3	1	6	4	5	2	6	2	1	7	8	9	3	1	6	4	5	2	6	2	1

如图所示，这一则示例也是非常清晰的 W-Wing 构造，所以逻辑就不再重复了，你可以自己推理得证。

10-3-5 空矩形欠一数对构造 (Empty Rectangle ALP+)

实际上，之前讲过的一个比较复杂的环结构空矩形欠一数对也是可以具有构造逻辑的。现在我们来看看。

7	2	4	6	3	1	2	2	5
2	2	1	2	6	7	3	1	2
4	5	5	8	9	8	9	8	9
2	3	1	2	4	5	6	7	8
8	1	5	6	7	9	3	4	2
6	4	7	9	8	9	2	1	5
2	2	3	1	5	4	6	7	8
3	5	4	2	4	8	9	6	7
4	5	6	2	7	1	8	9	3
1	7	8	9	3	5	4	2	6

如图所示。我们来阐述一下 $r7c2=r6c6(5)$ 的逻辑。发现到 $r7c56$ 和 $r89c6$ 四个单元格只有 2、4、8、9，而当 $r7c2(5)$ 和 $r6c6(5)$ 同假后，在 $r7c2345$ 和 $r5689c6$ 两个 ALS 里都将产生 4、8、9 的显性三数组，并得到 $r7c4 = r9c6 = 5$ 的结果。但显然，两个单元格同宫，不能填入一样的数字，所以产生了矛盾，所以强关系是成立的。

10-4 基于强制链的构造——毛刺、毛边

为了解决强制链观察困难和逻辑死板的特殊情况，我们发明了一种技巧，叫做**毛刺**（**Burr Logic**）和**毛边**（**Bi-burr Logic**）。

10-4-1 毛刺的定义

先来说说毛刺。毛刺是一些难题技巧里必不可少的包装技巧，它能够把一部分强制链改写为比较优雅的风格，而且逻辑更清晰、做题更容易观察。

我们来看看毛刺到底是如何工作的。

10-4-1-1 第一种毛刺用法

6	5	1	3	2 3	2	1	1 3	4 5	6	5	1	3	2 3	2	1	1 3	4 5
7	8 9	8	7	9 7	9 7	7	4	8	7 8	7	8 9	8	7	9 7	9 7	4	5
1	5	5	2	4	3	5 6	9	1 3	6	5 6	1	5	5	2	4	3	5 6
7	7	8	7	7	7	7	8	8	7 8	7	8	7	7	6	7	5 6	9
5	4	3	6	1	8	2	5	5 6	5 8	2	5	4	3	6	1	8	2
7	9	7	9	7	7	7	7	7	7	7	7	9	7	9	7	7	7
2	4 5	1	4	6	7	6	4	5	6	9	8	4	5	3	4	5	6
7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
2 3	2 3	1	3	8	1 2	1	5	6	4 5	6	8	4	6	1	2	3	4
4 5	5 6	4	6	7	7	7	7	7	7	7	7	7	7	7	7	7	7
8	3	9	5	4	6	7	6	2	4	6	8	3	9	5	4	6	7
7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
1	4	6	1	2	5	1	4	6	3	7	6	1	4	6	2	5	1
9	8 9	8	2	5	1	4	6	3	7	6	9	8 9	8	2	5	1	4
1 2	2	6	5	1	6	6	3	4	6	2	6	1 2	2	6	5	1	6
7	9	7 8 9	7	8 9	7	9	7	8 9	7	8 9	7	9	7	8 9	7	8 9	7
2 3	2 3	6	5	1	6	6	3	4	6	2	6	2 3	2 3	6	5	1	6
4	7	6	4	6	6	4	6	4	6	5	6	4	7	6	4	6	5
7	9	7	9	7	7	9	7	9	7	8 9	7	9	7	9	7	8 9	7

如左图所示，我们发现 r3c789 里，如果没有这个多余的 r3c9(7)，三个单元格将变为一个三数组，那么所在的宫和行上的其余 2、5、6 都可以被删除。

但可恨的地方就是 r3c9(7)是客观存在的，所以我们不能使用这个结论。那么我们就不得不找出一条强制链，来消掉对 7 的影响，而且还能找到和之前删数一样（或者大部分一样）的结论。

幸运的是，我们发现，当 r3c9(7)为真时，此时 r6c9(7)为假，于是产生了一个 XY-Wing 结构，使得它能删除 r23c9(6)；而在之前的三数组里，我们确实也能删除 r2c9(6)，所以，r2c9(6)便是这个技巧的结论。如图所示。

6	5 7 8 9	1 8	3 7 9	2 3 7 9 7	2 5	1 7	1 3 4 8	4 5 7 8
1 5 7	5 7 8	2	4	3 6 7	5 6	9	1 3 6 8	5 6 7 8
5 7 9	4	3	6 7 9	1	8	2 5	5 6	2 5 6
2 4 5 7	1	4 6 7	6 7	2 4 6 7	9	8	4 5	3
2 3 4 5 7	2 3 5 6 7	4 6 7	1 3 6 7	8	1 2 4 6 7	1 5 6 7	1 4 5 9	4 5 6 7 9
8	3 6 7	9	5	3 1 4 6 4 6 7 7	1 6 7	2	4 6 7	
1 4 9	6 8 9	1 8	2	5	1 4 6	3	7	6 8 9
1 2 7 9	2 6 7 8 9	5	1 6 7 8 9	6 7 9	3	4	6 8 9	2 6 8 9
2 3 4 7	2 3 6 4 7 9			4 6 4 6 7 9 7	2 5 6		5 6 8 9	1

这就是毛刺的基本使用方式：当我们发现一个结构多余了一个数或多个数的时候，我们想方设法把它（或它们）看作一个节点，以它为起点，作出一条强制链，沿途产生的所有强关系都是可以直接运用到结论里的。

当然，这个题只有一个弱关系，接了一个 XY-Wing 后，我们利用了 XY-Wing 的构造，产生了首尾的强关系，于是便有了删数。这就是我们上一段文字里说到的“沿途产生的强关系”。不过这个例子只有这一个强关系，所以找到的删数只有一处。有一些示例里的删数比较多，它便使用了更多的强制链沿途产生的强关系。

当然，毛刺结构是可以转为链的写法的，例如这个示例，我们可以将它记作

区块三数组：r3c789(256)=r3c9-r6c9(7)
=XY-Wing: {r3c8(56), r4c8(45), r6c9(46)}
=> r2c9 <> 6

不过，因为结构被嵌入链里，所以最好带上结构名来区分结构是什么。当然，最后的 XY-Wing，我们需要的仅仅是那一条强关系，所以我们甚至可以改写为

区块三数组：r3c789(256)=r3c9-r6c9(7)=(r3c8=r6c9(6)) => r2c9 <> 6

这和之前的毛刺转 ALS 一样，也被称为**解构**（Deconstruct）。当然，区块三数组也可以解构：

r3c89(6)=r3c9-r6c9(7)=(r3c8=r6c9(6)) => r2c9 <> 6

另外，我们称这个多余的数字为**毛刺**（Burr）。例如这个示例里的 r3c9(7)就是整个结构的毛刺。

如果你没有看过瘾，我们再来看一则使用示例。

7	1	2	6	1	3	8	1	2	1
4	6	3	1	2	5	8	4	2	7
4	6	8	5	7	3	4	6	1	9
4	5	9	4	6	4	6	4	3	8
8	2	3	8	2	1	9	4	6	5
3	7	5	8	1	6	4	9	2	3
1	4	5	6	4	9	7	9	8	5
2	5	8	7	3	4	1	5	3	9

如左图所示，如果 r23c1(9)为假，则我们尝试假设此时 r3c6(4)为假，则 r23c4 一定含有数字 4，且 r23c1 是 4、6 数对，那么此时，如果 r23c4(1)和 r3c8(1)同假的话，r23c148 将会构成关于 2、4、6 的拓展矩形的致命形式。为了规避该致命形式的产生，我们不得不设定 r23c4=r3c8(1)关系成立，且该强关系产生于 r3c6(4)为假的时候。于是顺次推导，并可以得到 r3c9 = 9 的结果，此时可以去掉 r3c6(9)；但如右图所示，如果 r23c1(9)为真，则 r46c1(9)同时为假，此时 r46c16 将形成关于 4 和 5 的共轭对类型的 UR 结构。于是此时 r46c6(4)必须为假，所以 r3c6(4)为真，此时一样可以删除 r3c6(9)。

由于两种情况均可以得到一样的结论，所以 r3c6 <> 9 必然成立。

注意右图中的某个箭头上，有一个波浪符号“~”，它用来表示左侧节点为假时得到右侧节点也为假的逻辑。同理，实线箭头上带有一个波浪符号表示左侧节点为真时得到右侧节点也为真的逻辑。这个标注模式是之前从未提到过的，这里需要你注意一下。在后面的内容里我们也会使用这样的写法，并也在附录里给出了这样的标注习惯的意思。

10-4-1-2 第二种毛刺用法

实际上，毛刺的用法还不止这一种，因为它用于链里，如果我们把它放在数组（或伪数组）里，会怎么样呢？

4	2	3	1	4	2	9	7	5	8	4	6
7	2	6	4	6	9	8	5	4	2	6	9
4	6	8	9	5	4	3	4	6	1	4	6
2	1	2	6	6	1	2	8	5	4	3	6
3	1	2	5	9	7	9	6	1	2	4	8
5	6	4	8	9	5	6	7	9	1	3	2
1	5	6	4	6	8	9	7	4	3	6	5
4	5	9	7	4	9	3	1	2	3	4	5
4	5	6	9	2	4	5	9	8	7	9	1

如左图所示，我们找到了一个假的伪数组，之所以说它是假的伪数组，是因为它的内部有 3、4 都可以跨区。我们一般认为的伪数组只能有一个数字可以跨区，这样我们才能找到适合的结论，而这个结构有 3 和 4 都可以跨区，那么我们只能去讨论跨区的最终情况了。

我们拿数字 4 来讨论，如果数字 4 不跨区的话，那么例子里只有 3 可以跨区，而结构涉及的 4 个单元格里恰好只有 3、4、6、9 四种数字，所以我们可以利用伪数组的结果直接得到，这三个 3 的交集就是删数，例如图中的 r8c46 和 r9c4，都处于这个交集上（当然，图里只标了一个结论删数，所以其它的都没标）。

接着看右图。当数字 4 跨区的话，显然只能把它们放到 r7c8 和 r9c5 上才可以实现跨区。所以放上去后，我们可以发现到一个强制 UR 结构：r39c45(34)。由于其中的一个单元格已经被我们刚才的假定前提 4 给覆盖了，所以我们基于这一点，可以发现 r9c4 如果填入 3 的话，这四个单元格就只能放 3 和 4：首先我们通过 r9c4(3)得到 r3c4(4)为真（双值格）；然后由于 r3 存在 3 的共轭对，所以 r3c5(3)必须为真，此时四个单元格只有 3 和 4 了。由于全部只能放下 3 和 4，所以形成了致命形式，故假设不成立，即 r9c4 <> 3。

而在第一种情况下（不跨区），r9c4(3)依然是可以删除的，所以 r9c4 <> 3 即为结论。

这则示例看起来好像跟毛刺没有什么关系，因为毛刺是结构本身多出来一坨东西，但实际上，我们可以按照 r9c5(4)是否成立来进行讨论，并基于伪数组这一点来产生不同的讨论情况。

毛刺的两种用法就说完了，接下来我们来看一些其它的经典示例。

10-4-2 链列的毛刺用途

10-4-2-1 链鱼（Kraken Fish Type 1）

链鱼（也叫**链链列**、**远程鳍链列**、**远程鳍鱼**，**Kraken Fish**，由于属于鱼的构造，所以也可以简记为**鱼+**或**链列+**）是 Hodoku 里规定的一种技巧，在结构里存在一个鱼鳍，不过这个鱼鳍不像是之前的鱼鳍一般可以直接对应删数，而是添加一条链来得到删数。

1	4	6	7	2	3	4	5	4	5	6	8	9	4	6	9	1	4	6	7	2	3	4	5	4	5	6	8	9	4	6	9						
2	2	2	2	3	1	3	1	4	5	7	5	6	8	9	4	5	6	8	9	7	1	3	4	5	6	8	9	4	5	6	8	9					
5	4	9	3	6	8	7	4	3	1	2	5	6	8	9	4	5	6	8	9	7	1	3	4	5	6	8	9	4	5	6	8	9					
4	7	1	5	8	1	6	2	5	6	9	3	2	3	5	6	9	4	7	1	5	8	1	6	2	5	6	9	3	2	3	5	6	9				
2	2	5	8	6	3	4	9	1	2	5	7	2	5	7	2	5	7	2	5	7	2	5	7	2	5	7	2	5	7	2	5	7	2	5	7		
9	3	1	2	5	7	1	6	2	6	4	8	2	6	4	8	2	6	4	8	2	6	4	8	2	6	4	8	2	6	4	8	2	6	4	8		
3	4	2	6	4	1	5	8	4	6	9	7	4	6	9	3	4	2	6	4	1	5	8	4	6	9	7	4	6	9	3	4	2	6	4	1		
2	6	1	9	4	7	2	6	4	6	3	2	3	5	6	2	6	4	6	3	2	3	5	6	2	6	4	6	3	2	3	5	6	2	6	4	6	
7	2	6	5	4	5	8	1	3	2	3	5	6	8	2	6	4	6	3	2	3	5	6	8	2	6	4	6	3	2	3	5	6	8	2	6	4	6
7	6	5	4	5	8	1	3	2	3	5	6	8	2	6	4	6	3	2	3	5	6	8	2	6	4	6	3	2	3	5	6	8	2	6	4	6	

如左图所示，假设 r3c2(4)不存在，则 X-Wing 在 r39 上作为定义域是成立的。所以删除域上的 4 全部都可以删。不过由于 r3c2(4)客观存在的缘故，我们只能把它作为毛刺

进行讨论。

假设 $r3c2 \neq 4$ 显然可以删数；而 $r3c2 = 4$ 时，我们可以找到一条强制链： $r3c2(4-9)=r3c7-r12c9(9=4)$ 。由于沿途的所有强关系都可以使用，所以我们依次来寻找强关系末端的结果是否可以对应到删数上。

首先是 $r3c7(9)$ ，当它为真时，原本的删数都是 4，跟 9 无关，所以这个强关系没有删数；接着是 $r12c9(4)$ ，可以看到此时我们可以发现 $r12c9(4)$ 可以对应删数 $r1c7(4)$ ，所以 $r1c7 \neq 4$ 便是这个结构的结果。

当然，很显然的是，毛刺可以写成一条普通的链，所以这个示例可以把它写成一条普通的链：

$r3c2(4-9)=r3c7-r12c9(9=4) \Rightarrow r1c7 \neq 4$

这样写的好处是可以更直观地把结构用链表达出来。不过，我们就需要介绍一种新的节点类型：X-Wing（以及更大型的链列结构）。这则示例是链鱼的第一种类型。

在讲第二种类型之前，我们先要处理一个遗留的问题：X-Wing 作为节点是什么样的东西。

10-4-2-2 对 X-Wing 节点的讨论

和之前介绍过的节点类型不同的是，X-Wing 的结构非常大，这使得我们在讨论真假的时候，不得不思考如何去放置这些数字。

首先，考虑一下 X-Wing 节点为真的情况。

		1				1		
		1				1		

如图所示。显然 X-Wing 节点为真，就意味着对角线两端的两个单元格（左上 $r3c3$ 和右下 $r7c7$ ，或者是右上 $r3c7$ 和左下 $r7c3$ ）两处的填数模式是一样的，即有两个位置要填这个数 1，而两个位置不填这个数 1。

那么 X-Wing 节点为假则是这个说法的对立面。X-Wing 一共涉及四个节点，所以严谨的方式是，我们需要讨论的情况有如下五种：

- 节点里有 4 个单元格都填这个数；
- 节点里有 3 个单元格都填这个数；
- 节点里有 2 个单元格都填这个数；
- 节点里有 1 个单元格都填这个数；
- 节点里有 0 个单元格都填这个数（即没有位置放这个数）。

别怕，讨论的情况看起来很多，但实际上很多情况是完全不可能出现的。显然，X-Wing 里完全不可能出现 3、4 个节点都填这个数字的情况；而一下我们就排除了近一半的情况；而填入 2 个显然也不可能，因为怎么放置 2 个，只要放置的位置合理不产生违背数独规则的东西，都是 X-Wing 为真的情况，而我们要讨论 X-Wing 为假，所以它不属于这种情况。

所以一下我们就只剩下两种情况了。思考一下，X-Wing 可能存在“有 0 个单元格填这个数”的可能吗？如果你说是，那么你就错了。由于鱼鳍的存在，那么鱼鳍必然只会存在在定义域上，而只有一个鱼鳍的情况显然是不可能出现在两个定义域区域上的。如果没有位置填这个数，这就会使得有一个定义域区域下放不下这个数字。比如上图里，定义域我们人为固定为 r37，假设鱼鳍在 r3 上，那么如果属于“有 0 个填入”的情况的话，r7 就无法找到合适的位置能够放下 1 了，因为 r37 是定义域，定义域规定了结构必须产生于这些区域，即区域里必须至少有一个位置是放这个数的。

所以，在五种情况里，只有一种情况是符合 X-Wing 为假的，那就是“四个节点里有一个放这个数”，而这个数还必须位于刚才我们说到的、鱼鳍不在的定义域区域上。所以综上所述，我们就讨论了 X-Wing 节点真假的情况了：

- **X-Wing 节点为真：**恰有两处对角位置上填入这个数字；
- **X-Wing 节点为假：**有且仅有一处位置可以填入这个数字。

那么我们再次回到原来的题目上去理解。

1	4	6	7	2	3	4	5	4	3	5	6	4	6
2	6	2	2	3	4	1	3	1	3	7	3	4	6
8	8	8	9	8	9	9	4	5	8	9	8	9	8
5	4	9	4	3	6	8	7	4	3	1	2	1	2
4	7	1	5	8	1	6	2	5	6	5	6	9	3
2	2	2	6	3	4	9	1	2	5	2	5	7	7
8	8	8	6	3	4	9	1	6	2	6	4	8	8
9	3	1	2	5	7	1	6	2	6	4	8	8	8
3	4	6	2	1	5	8	4	2	6	7	4	6	9
2	6	1	9	4	2	6	4	6	5	6	5	6	5
7	8	2	2	7	2	6	4	6	8	8	8	2	7
7	6	5	4	5	6	3	4	6	8	8	6	1	7

如图所示。我们率先就得假设 X-Wing 节点（r39c37(4)）为假（图上没有画出 X-Wing 整体）。显然，当 X-Wing 节点为假时，只有 r9c37 里有一处是 4，而 r3c37 里都不能是 4，也因此我们确定了 r3c2 是 4。因为它是 r3 上的鱼鳍，r3 只有 r3c237 三处可以填入 4，现在 X-Wing 节点规定 r3c37(4)都为假了，显然只有 r3c2(4)为真了。所以，

X-Wing 节点为假后，得到了 $r3c2(4)$ 为真，于是链得以往下继续理解。

X-Wing 节点的逻辑就讨论完了。那么剑鱼（三链列）和水母（四链列）呢？虽然结构更大，但讨论逻辑依旧完全一样，首先我们可以得到的是，三链列最多占据 9 个单元格，而三个定义域上最多只允许我们能放下 3 处链列涉及的这个候选数，所以 4 到 9 的情况完全不用我们考虑，因为这样必然是有重复的，所以问题直接简化到讨论 0 到 3 个位置填数的情况；同理，四链列也是一样，最多我们也只需要讨论的是 0 到 4 个位置填数的情况。

不过，实际上，管它填多少个，你都会发现，在节点为假时，填数位置一定不会出现在鱼鳍所处的定义域区域下。这便有了我们上面那样的结果。所以，不论链列的规格是多大的，我们总能得到相似的结论，这便是链列这个东西神奇的地方所在。

另外，可以从图上看到，前面的例子里毛刺的讨论过程跟鱼鳍完全一样，它们唯一的差别是，鱼鳍可以直接对应删数，而毛刺不能。实际上，鱼鳍和毛刺是一样的东西，毛刺由中国的数独玩家探长（昵称）发现，并由他命名了这个技巧。从他本人口中得到的是，毛刺应当和鱼鳍是一样的，只是使用场合不同：鱼鳍一般用于鱼（链列）结构，而毛刺则针对于其它的非鱼（链列）技巧。

10-4-2-3 链鱼的推广（Kraken Fish Type 2）

接下来我们来看一下链鱼结构的推广，到底该怎么用。

3	^{1 2} ₄	6	^{1 2} ₅	¹ _{5 8}	9	² _{4 5 7}	² _{7 8}	² ₈
8	7	² ₉	^{2 3} ₅	6	4	^{2 3} ₅	1	^{2 3} ₉
¹ ₄	^{1 2} _{4 9}	5	7	^{1 3} ₈	1	² _{4 6}	^{2 3} _{8 9}	² _{6 8 9}
¹ ₉	^{5 6} ₉	3	¹ ₉	8	¹ _{7 9}	^{5 6} ₇	² _{7 9}	4
2	⁴ _{6 9}	¹ _{4 8 9}	^{1 3} ₆	^{1 3} _{4 7}	1	^{1 3} _{6 7}	³ _{7 8 9}	5
7	^{4 5} ₉	⁴ ₈	^{1 3} ₅	^{1 3} _{4 5 9}	2	^{1 3} ₁	6	³ _{8 9}
¹ _{4 5 9}	^{1 2} _{4 5 6 7 9}	^{1 2} _{4 7 9}	¹ _{5 6}	¹ ₇	3	8	² ₅	^{1 2} ₆
¹ ₅	8	¹ ₇	9	2	^{5 6} ₇	³ ₆	4	^{1 3} ₆
¹ _{5 6}	^{1 2} _{5 6}	^{1 2 3} ₁	4	¹ _{5 8}	¹ _{5 6 8}	9	^{2 3} ₅	7

如图所示，我们可以看到一个链列结构，不过它带有一个鱼鳍，而删数却跟鱼的涉及数值完全不一样。我们来看一下，我们如何运用这个鱼结构。

3	^{1 2} ₄	6	^{1 2} ₅	¹ ₈	9	² _{4 5}	² _{7 8}	² ₈	3	^{1 2} ₄	6	^{1 2} ₅	¹ ₈	9	² _{4 5}	² _{7 8}	² ₈
8	7	² ₉	^{2 3} ₅	6	4	^{2 3} ₅	1	^{2 3} ₉	8	7	² ₉	^{2 3} ₅	6	4	^{2 3} ₅	1	^{2 3} ₉
¹ ₄	^{1 2} _{4 9}	5	7	¹ ₈	^{1 3} ₈	^{2 3} _{4 6}	^{2 3} _{6 8 9}	^{2 3} _{6 8 9}	¹ ₄	^{1 2} _{4 9}	5	7	^{1 3} ₈	¹ ₈	^{2 3} _{4 6}	^{2 3} _{6 8 9}	^{2 3} _{6 8 9}
¹ _{5 6 9}	3	¹ ₉	8	¹ _{7 9}	^{1 5 6}	^{1 2} _{7 9}	² _{7 9}	4	¹ _{5 6 9}	3	¹ ₉	8	¹ _{7 9}	^{1 5 6}	^{1 2} _{7 9}	² _{7 9}	4
2	¹ _{4 6 4}	¹ _{8 9}	^{1 3} _{6 4}	¹ _{7 9}	^{1 3} _{6 4}	^{1 3} _{7 8 9}	5	¹ _{4 6 4}	2	¹ _{4 6 4}	¹ _{8 9}	^{1 3} _{6 4}	¹ _{7 9}	^{1 3} _{6 4}	^{1 3} _{7 8 9}	5	¹ _{4 6 4}
7	¹ _{4 5}	¹ _{8 9}	^{1 3} _{5 4}	^{1 3} _{5 4}	^{1 3} _{5 4}	^{1 3} _{5 4}	2	¹ _{4 5}	7	¹ _{4 5}	¹ _{8 9}	^{1 3} _{5 4}	^{1 3} _{5 4}	^{1 3} _{5 4}	^{1 3} _{5 4}	2	¹ _{4 5}
¹ _{4 5}	^{1 2} _{4 5 6 4}	^{1 2} _{4 5 6 4}	¹ ₅	¹ ₇	3	8	² _{5 6}	^{1 2} _{4 5}	¹ _{4 5}	^{1 2} _{4 5 6 4}	^{1 2} _{4 5 6 4}	¹ ₅	¹ ₇	3	8	² _{5 6}	^{1 2} _{4 5}
¹ ₅	8	¹ ₃	9	2	¹ _{5 6}	^{1 3} _{5 6}	4	¹ ₃	¹ ₅	8	¹ ₃	9	2	¹ _{5 6}	^{1 3} _{5 6}	4	¹ ₃
¹ _{5 6}	^{1 2} _{5 6}	^{1 2 3}	4	¹ ₈	¹ _{5 6 8}	9	^{2 3} _{5 8}	7	¹ _{5 6}	^{1 2} _{5 6}	^{1 2 3}	4	¹ ₈	¹ _{5 6 8}	9	^{2 3} _{5 8}	7

如左图所示，链表述如下：

$$r4c5-r179c5(5)=r7c5-r7c3=r8c3(7) \Rightarrow (r4c5 = 5 \Rightarrow r8c3 = 7)$$

这条链以弱关系开头了，显然用了强制链的思想。逻辑是这样的：当 $r4c5(5)$ 为真时， $r8c3(7)$ 为真。很显然， $r8c3(7)$ 为真了，自然 $r8c3(1)$ 为假。

而如右图所示，链表述如下：

$$r8c3=r8c6(7-5)=r4c6(5-6)=r4c1(6-1)=r45c3(1) \Rightarrow (r4c5 \neq 5 \Rightarrow r8c3 \neq 1)$$

那这条链想说明什么呢？如果 $r4c5(5)$ 为假时， $r8c3$ 照样不为 1。首先观察到 $r8c3(7)$ 为链头，引出一条区块不连续环，直至 $r45c3(1)$ ，这样就可以删除 $r8c3(1)$ 了。但是有一个小问题， $c6(5)$ 并不是强关系。那为什么用强关系了呢？

因为这是一个为假的二链列。当这个为假时，它就是毫无瑕疵的二链列结构了，而之前环结构一章里，二链列结构可以被看作环的形式，而弱关系下的所有数字可删，从而把弱关系改变为强关系的形式。这里就是利用了这一点：为假，二链列成立，二链列又是一个环，所以所有二链列内的弱关系均可被看成强关系使用，所以这才成立的。只是要注意一点，这只是为假时，才成立，并不是所有情况下都成立，别忘了这个大前提。

然后按照这个区块不连续环的逻辑，当为假，链结构成立，并最终得到了 $r45c3(1)$ 为真的结论，所以此时 $r8c3(1)$ 照样为假。那么，不论为真的真假， $r8c3(1)$ 都可以删除。所以 $r8c3 \neq 1$ 。

10-4-2-4 节点重叠的链鱼

4	2	1	5	3	6	1		
1		3	4	1	9	5	8 9	7 8 9
7 8	8	7		7	2		1 2	2 3
5	5	9	6	8	1	4	7	3
1 2	4	1 2 3	9	1 2	7	5	2	6
8			7	8	9	1	8	8
6	5	7	4	8	3	9	2	1
1 2	7	9	1	6	5	3	4	6
8			8		8	8	8	8
2	1	2	3	4	2	2	6	4 5
5	7 9		8	7 8	8	8 9	8 9	8 9
	3	6	2	1	1	3	1	3
7 9		9	8	4	4		6	4
2 3	5 6	4	1	7	5		9	9
5			6	6	1	2 3	5	2 3
			8	8	8	7	8	8

如图所示，链表述如下：

$r1c6=r7c6(7-4)=r5c6(4-3)=r5c2-r2c2(3)=r23c3(36-7)=r17c36(7)$
 $\Rightarrow r1c9 \lt;> 7$

那么简单叙述一下它的逻辑：当 $r1c6(7)$ 为假时，顺次得到 $r7c6(7)$ 真、 $r7c6(4)$ 假、 $r5c6(4)$ 真、 $r5c6(3)$ 假、 $r5c2(3)$ 真、 $r2c2(3)$ 假，然后 $b1$ 内产生 36 隐性数对，所以 $r23c3(36)$ 隐性数对为真， $r23c3(7)$ 均为假。然后，因 $r23c3(7)$ 均为假，所以观察 $c36(7)$ ，二链列结构成立。

所以最终链的删数为 $r1c6(7)$ 和 $r17c36(7)$ 的交集。 $r1c6(7)$ 和 $r17c36(7)$ 都能删除的是 $r1$ 其余位置的候选数 7，所以 $r1c9 \lt;> 7$ 。

这个是单候选数和二链列的节点重叠结构。

10-4-2-5 四个鱼鳍的三头链

4	² 6	⁶ 9	1	⁵ 8	² 6	⁶ 9	⁷ 8	3
² 7	5	² 7	¹ 4	9	3	² 4	² 8	¹ 6
² 6	8	² 6	7	¹ 4	² 6	5	² 9	¹ 6
² 6	7	² 8	² 8	5	⁴ 8	1	² 8	⁹ 9
9	¹ 6	² 8	¹ 8	7	¹ 8	3	5	4
5	¹ 8	4	¹ 8	3	² 9	6	⁷ 8	⁹ 9
8	⁴ 9	5	¹ 9	2	⁴ 7	3	⁷ 9	⁶ 9
² 7	² 6	² 9	3	8	⁴ 7	⁶ 9	1	5
1	3	⁶ 9	⁴ 5	⁶ 9	⁴ 5	8	⁴ 9	2

如图所示，链表表述如下：

```
{r8c7, r9c8}(49)=r8c7-r17c27(6)
=r7c9(6)
={r1c5, r7c45}-r9c5(6)=r9c8(9)
=> r7c9 <> 6
```

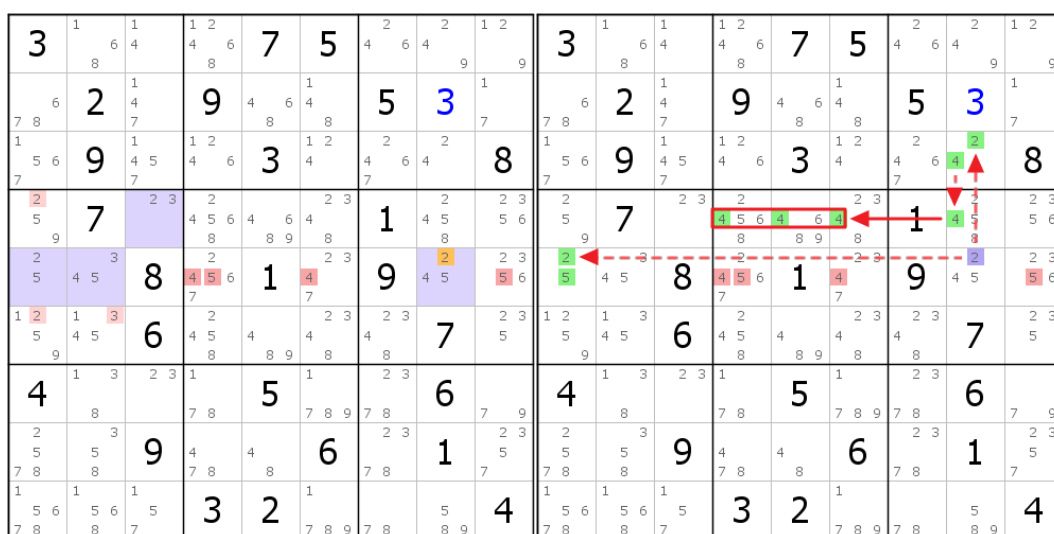
首先，{r8c7, r9c8}(49)的毛刺显性数对为假时，r8c7(6)为真（之前说明过这一点，这里不再赘述）。然后当 r8c7(6)为真时，r17c27(6)的二链列结构（如果暂时先不看 r1c5、r7c45 这四个鱼鳍的话）必然无法成立。因为 r8c7(6)为真必然会排除二链列结构同一列上的两处填数位置，二链列必然是对角两格同时为真的，这样一排除必然导致无法这么去填数，从而形成弱关系。那么，当二链列结构为假时，那只能让鱼鳍至少其一为真了。此时分两路观察。

如果 r7c9(6)为真，那必然 r7c9(9)为假；而如果剩下三个鱼鳍至少一个为真的话，都会使得 r9c5(6)为假，于是观察 r9c58 这个 nALS 区域，最终得到 r9c8(9)为真。这样的话，r7c9(9)也应为假。

这样就有两个链尾。这是一个三端点链。而链头则是{r8c7, r9c8}(49)的毛刺显性数对，而它照样是可以删除 r7c9(9)的，所以 r7c9 <> 9。

这个例子比较麻烦的是，它带有四个鱼鳍，但实际上我们在使用的时候还是当作一个整体使用的。

10-4-3 毛刺 SDC (Burred SDC)

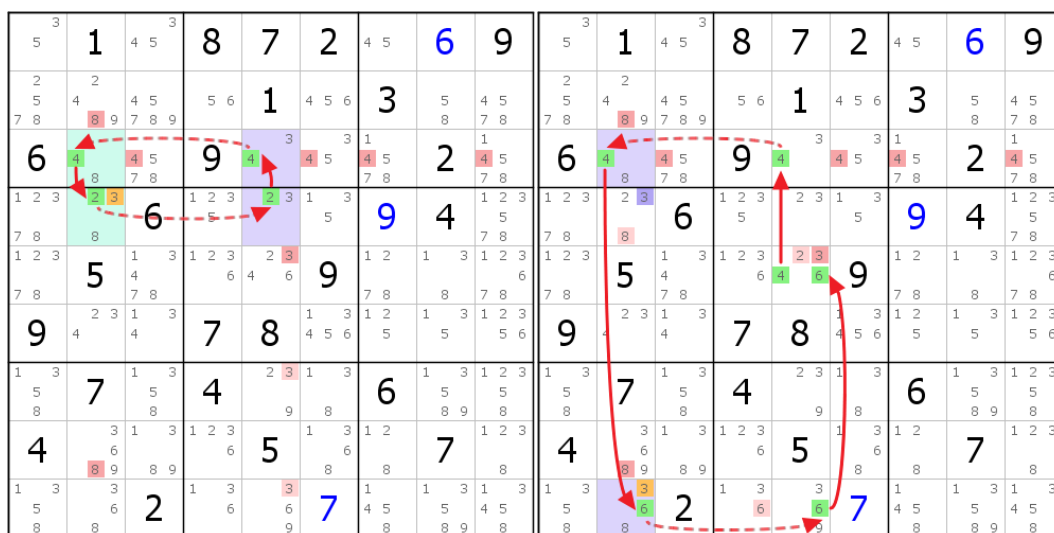


如左图所示，如果我们尝试去掉 $r5c8(2)$ ，那么 $r4c3$ 和 $r5c128$ 四个单元格将构成 SDC 结构，所以可以删的数字全部已经标注了出来。

不过，因为它是客观存在的，所以我们视作毛刺，假设为真并讨论其逻辑，此时就会引出一条动态强制链，并走了两个分支。一个分支将走向 $r5c1(5)$ ，而另外一个分支则走向了 $r4c456(4)$ 。显然，它们都是通过毛刺 $r5c1(5)$ 为真得到的，所以它们将同时成立，故删数就是这些 4 和 5，和左图删数的交集，即 $r5c46(4)$ 和 $r5c49(5)$ 。

10-4-4 毛刺环 (Burred Continuous Nice Loop)

10-4-4-1 环 + 双 RCC 的 ALS-XZ



如左图所示，我们尝试把 $r4c2(3)$ 看作毛刺，假设为假的话， $r34c24$ 将会构成一个双 RCC 的 ALS-XZ，当然它也算一个普通的环结构，不过删数理解起来也和之前的方式一样：首先删除弱关系的交集，然后再来看 ALS 区域涉及的数字，组成数组，删除交集。所以左图的所有删数我们都使用了红色标注出来了。

再看右图，当 $r4c2(3)$ 为真的时候，我们不得不讨论真的时候的情况，此时，我们利

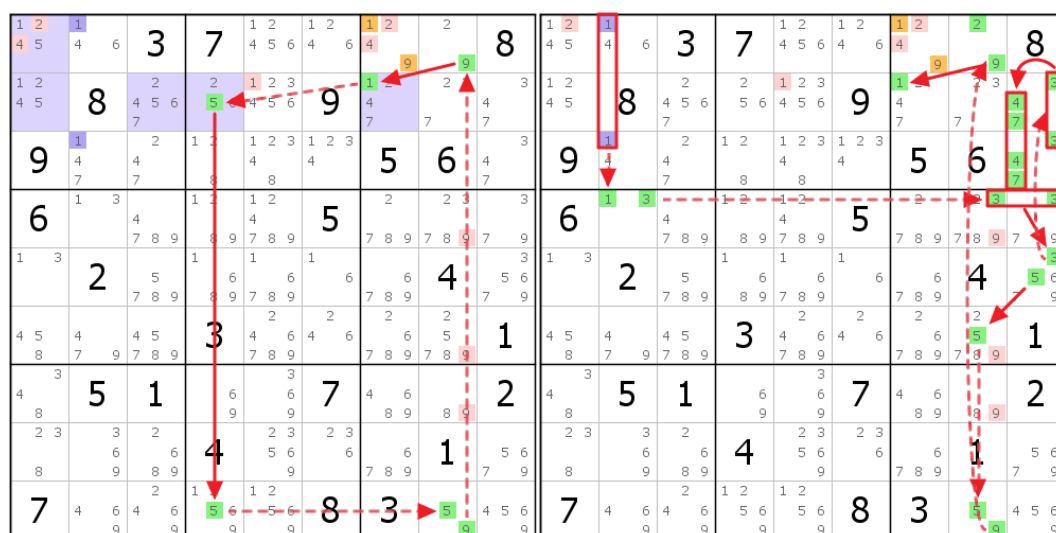
用 $r4c2(3)$ 为真的特性，向下排除，得到 $r9c2 <> 3$ ，此时，右图给出嵌套 ALS 结构的环就成立了，删数也全部都标注出来了。

可以从示例里面看到，它们两个环的删数是有部分相同的，所以我们取出相同部分（深红色），就是这个毛刺环结构的删数了。

可能你会问我，到底这里说的毛刺环指的是左边还是右边的环。实际上都可以，这一点并不重要，不过我更喜欢称呼的毛刺环指的是右边这个，因为右边毛刺为真的时候得到的结果，作为强调来说，确实右侧的重要性更大。

10-4-4-2 环 + ALP

下面要讲到的例子是最后一个毛刺的示例，不过示例非常精彩，而且富有挑战性。



如左图所示，我们尝试把 $r13c2(1)$ 看作毛刺的话，那么就会产生这样的环：

环： $r2c4=r9c4-r9c8(5=9)-r1c8(9)=r2c7(1)-r2c4(5)$

可以看到，把以前学到的所有知识点代入，可以理解到大部分的逻辑，但 $r2c7(1)-r2c4(5)$ 是很难立刻明白原因的，你可以尝试想想看，如果你能独立明白它，那么恭喜你，你对以前的知识已经真正做到了灵活运用。如果没有，也没有关系，下面我们来讲解一下左图这个弱关系到底是怎么产生的。

首先，我们要明白链的基本构造原理是基于**头尾至少有一个为真，形成强关系**的。所以图上我们可以率先得到 $r2c4(5)$ 和 $r2c7(1)$ 不同假，即至少有一个为真。那么我们就不妨考虑一下这两个节点的真假情况。

如果两个节点同真，则观察 $b1$ ，由于毛刺此时假设为假，所以放下 1 的位置只有 $r1c1$ 这一处，而 5 本身也只有 $r1c1$ 一处位置。说白了， 1 和 5 此时被压在 $r1c1$ 里。显然两个单元格是无法填入一个数字的，所以出现矛盾，故两个节点不同真。

但是，两个节点一共只有四种组合：真真、真假、假真、假假。显然假假不允许（强关系定义不同假），真真也不允许（刚才证明得到 1 和 5 无法放进去）。所以，只有两种情况，即一真一假。既然是一真一假，那么会发生什么事情呢？

第一，这两个节点可以形成弱关系了。因为两个节点的关系是一真一假，既然是一真一

假，那么这种情况就既满足强关系的定义，也满足弱关系的定义（一真一假的情况恰好处于两个定义的交集上），所以两个节点的关系既可以按强关系处理，也可以按弱关系处理。而此时，我们当然喜欢让结构更完美，即环比链更好（删数更多），所以我们会优先将其当作弱关系处理，所以弱关系就成立了。待会儿我们还会用到这个弱关系。

第二，会让涂色的五个单元格{r1c1, r2c1347}构成关于 1 和 5 的欠一数对结构（ALP）。这一点很神奇，可事实确实如此。不论 r2c4(5)为真，还说 r2c7(1)为真，我们都可以确定 b1 里能放下 1 和 5 的位置都只能放在 r1c1 里，所以 r1c1 实际上只能放 1 和 5，也因此，欠一数对就成立了。

既然得到了两个结论，那么环的删数就很明确了：首先是我们之前知识能够删除的地方：r1c7(24)、r467c8(9)。那么 r1c1(24)和 r2c5(1)则是由于刚才的欠一数对成立时，产生的删数（欠一数对成立会同时使得 b1 和 c2 上产生 1、5 的待定的显性数对），所以删除其余位置的 1 和 5。当然，就 b1 而言，r1c1 只能填入 1 和 5，所以 2 和 4 当然就删掉了。所以总的来说，第一个图能得到这样的结论。

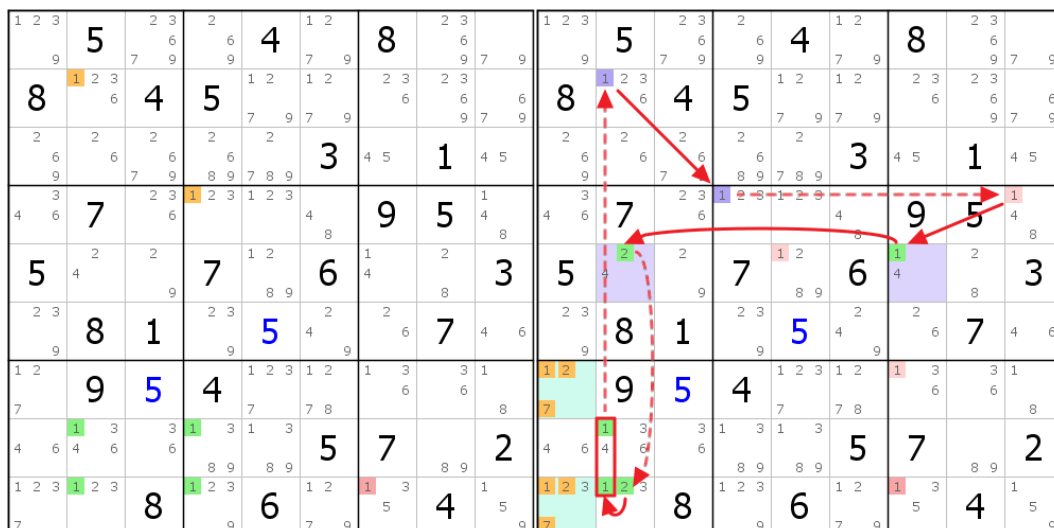
我们再来看右图。右图看起来特别复杂，但实际上我们只要掌握一点就好：当毛刺引入强制链（即设为真）的情况时，我们只需要找出所有强关系处能对应到的删数，而且这个删数在原情况（毛刺设为假）时是可以删掉的的话，那就可以作为整个技巧的删数结论了。所以，我们就来尝试找出结果。我们引入了一个动态强制链，下面简单叙述一下路径。首先从毛刺 r13c2(1)为真出发，得 r4c2(1)假、r4c2(3)真、r4c89(3)假、r45c9(3)真。此时分两个分支走向：第一个分支是走上面 r23c9(3)假、r23c9(47)毛刺显性数对真结束；第二个分支则是走下面 r5c9(5)假、r6c8(5)真、r9c8(5)假、r9c8(9)真、r1c8(9)假。此时走两个分支方向：由于 r1c8(9)假的缘故，所以 b1 存在 ALS 区域，使得 r1c8(9)=r2c7(1)成立；走另外一个方向则是 r1c8(9)假的缘故，只得使 r1c8(2)真（双值格）。于是，我们在路径上分别得到了 r1c8(2)真、r2c7(1)真、r23c9(47)毛刺数对真、r9c8(9)真，所以它们对应到的删数 r1c1(2)、r1c7(24)、r467c8(9)都是这个动态强制链路径上能排除的数字情况。而对应左图，只有 r1c1(4)不能删，其它的数字全部能删。所以，整个技巧里，除了 r1c1(4)不可以删以外，右图里的所有删数全部都可以作为结论成立。即，整个题目的删数如下图所示。

1 2	1	3	7	1 2	1 2	1 2	2	8
4 5	4 6			4 5 6	4 6	4	9	
1 2	8	2	2	1 2 3	9	1 2	2 3	3
4 5		4 5 6	5 6	4 5 6		4	7	4
9	1	2	1 2	1 2 3	1 2 3	5	6	3
	4	4	8	4	4			4
	7	7		8				7
6	1 3	4	1 2	1 2	5	2	2 3	3
		7 8 9	8 9	7 8 9		7 8 9	7 8 9	7 9
1 3	2	5	1	6	6	6	4	3
		7 8 9	8 9	7 8 9		7 8 9		5 6
4 5	4	4 5	3	4	2	2	2	1
8	7 9	7 8 9	7 8 9	7 8 9	4 6	6	5	
	3			3		7 8 9	7 8 9	8 9
4	5	1	6	6	7	4 6	8 9	2
			9	9				
2 3	3	2	4	2 3	2 3	6	1	5 6
8		6		5 6		7 8 9		7 9
7	4	6	1 2	1 2	8	3	5	4 5 6
		9	5 6	5 6			9	9

10-4-5 毛边的定义

如果毛刺是“技巧 + 1”的话，那么毛边则是“技巧 + 2”。看起来差别不大，但实际上在使用过程之中，我们并非是将这两个特殊的部分作为强制链的形式进行挨个讨论的，而是构成强弱关系来讨论。

10-4-5-1 强毛边（Strong Inference in Bi-burr）



如左图所示，可以看到，在 c24 上一共有 6 个 1。如果我们把 r2c2(1)和 r4c4(1)抛开不算的话，恰好就构成了一个二链列结构。但是由于客观的存在，导致我们没办法使用这个技巧，所以我们只能加入链的思维。

此时由于是两个毛刺，所以我们不得不分四种情况讨论：**真真、真假、假真、假假**。虽然看起来很多，但实际上我们依旧可以简化逻辑。首先，它们同为假的时候，二链列的结构就成立了，所以我们把它单独算成一种情况；而剩下三种，我们可以发现，**真真、真假、假真**这三种情况可以直接一并称为“不同假”。而不同假即为强关系，所以我们还有一种情况，就是直接把它们用强关系相连，接着就有了右图这样的不连续环结构：**r5c7(1)=r5c2-r9c2(2)=r8c9-r2c2=r4c4-r4c9=r5c7(1)**。可以看到，这个结论可以直接得到 **r5c7 = 1** 的结果，而原来的二链列可以删除 r5c7(1)对应到的 r9c7(1)，所以，r9c7(1)即该技巧的结论。

可以发现，这一则示例我们利用了四种情况来简化为两种：同假形成结构和强关系。这种构造形式我们就称为**毛边（Bi-burr）**，可以从英文名发现，它就是两个毛刺（Bi-前缀加上 burr）。“边”暗示两个节点连接在一起形成链结构的一条边。

可以从示例所分的四种情况里发现，既然我们可以划分为强关系和同假，那么是否也可以划分为弱关系和同真呢？答案是肯定的，因为弱关系对应不同真，而恰好跟同真是互斥的情况。

10-4-5-2 弱毛边 (Weak Inference in Bi-burr)

3	4		5	1	8		3	2		3	4		5	1	8		4	6	7	3	2
9	7						7	6			5	3					4	5	1		5
5	8		6	9		4	3	2		4	5	8		1		7					3
2		4	5	1		6	4	3		6	4	5	8		9		5				3
1		9	7																		
4	6			1		5	8			2		2						2			
5	3		2			2				5	8	9	7		9		1				6
6		1	5	8		2	7	4		1	5	9	8		3	1	3				
7	9	4	3		5	6	8		1	5	6			2	1	5	6				
1		5	2			6	5	6		1	5	6	9								

如左图所示，如果我们假设 $r4c2(1)$ 和 $r6c7(1)$ 同真和不同真，会怎么样呢？假设它们不同真，我们首先可以得到的结果是，不论是同假也好还是有一个填入也好，我们都有 $r4c2(7)=r6c7(6)$ ，说明这一点也很简单：当两个 1 同假时，它们就直接填进去了，显然不同假，所以这个强关系成立；如果有一个为真，则另外一个单元格必须填入 6 或者 7 这个数字，这便使得两个单元格的 6 和 7 不同假了，所以依然是强关系。于是，我们利用上这个强关系，我们可以寻找到一个连续环结构。

不过有一个问题。我们这个例子在讲弱毛边，显然应当是同真和不同真的情况，但是左图却用的是强关系（即不同假）。不同假和同真并非是互斥的，所以我们证明思路是否出现了问题和漏洞呢？其实不然。这一点就得用到环的特性了。之前环里有一个特殊的特性，环内的任意关系都可以切换，即强关系可以视为弱关系；而弱关系则可以视为强关系。既然环内的任意相邻两个节点都是一真一假的话，那么显然我们就可以把这里 $r4c2(1)$ 和 $r6c7(1)$ 作为弱关系来看待。所以实际上并没有出现问题。

接着我们来看右图。由于左图我们是当作不同真处理了，所以右图必须是同真的情况。可以看到，当两个 1 同真后，对 $r7$ 排除，就可以得到 $r7c27(1)$ 为假的结果，于是我们就有了 $r7c27(1)=r7c7(9)$ 的强关系。接着向上连接，最终可以得到 $r2c9(7)$ 为真。所以实际上，这个例子我们可以通过 $r4c2(1)$ 和 $r6c7(1)$ 同真而得到 $r2c9(7)$ 为真。

可是例子删什么数字呢？左图是一个环，可以删除的是两个数字： $r2c2(7)$ 和 $r8c7(6)$ ，而在右图我们得到了 $r2c9(7)$ 为真的结果，所以 $r2c2(7)$ 和它是同行的，所以可以删除掉，而 $r8c7(6)$ 就不行了。所以这个例子最终可以删除的数字只有 $r2c2(7)$ 。

你也别小看了它，因为这个题目就这一个技巧就可以破解。

可以发现，弱毛边和强毛边的使用有一点点小的区别是，弱毛边在处理同真的情况的时候，我们利用的方式一般有两种：一种是通过像上图这样的排除，进而得到想要的强关系；而另外一种则是通过结构的占位来得到强关系。所以弱毛边用起来比强毛边更灵活。

接下来我们再来看看一些特殊的结构或技巧都如何利用毛边进行解题。

10-4-6 一些常见技巧里的毛边

10-4-6-1 BUG + 3

9	1	3	2	7	8	4	6	1	3	5	4	6	9	1	3	2	7	8	4	6	1	3	5	4	6
5	4	3	7	1	4	6	9	2	8	4	6	9	5	4	3	7	1	4	6	9	2	8	4	6	9
8	6	1	4	2	3	5	7	1	4	6	9	8	6	1	4	2	3	5	7	1	4	6	9	8	6
4	2	6	5	9	1	8	7	3	5	4	6	9	4	2	6	5	9	1	8	7	3	5	4	6	9
3	1	8	4	5	2	7	9	6	4	5	8	3	1	8	4	5	2	7	9	6	4	5	8	3	1
7	9	5	8	4	6	3	2	1	7	9	5	8	4	6	3	2	1	7	9	5	8	4	6	3	2
1	4	3	9	5	2	6	7	8	1	4	3	9	5	2	6	7	8	1	4	3	9	5	2	6	7
2	5	9	6	7	8	1	4	3	2	5	9	6	7	8	1	4	3	2	5	9	6	7	8	1	4
6	7	4	8	1	3	5	9	2	6	7	4	8	1	3	5	9	2	6	7	4	8	1	3	5	9

如图所示，这是一个 BUG + 3 结构，其中三个真数分别是 r2c9(4)、r7c9(8)和 r9c6(4)。为了好用毛边结构，我们将两个 4 作为毛边来看待，并使用强毛边来推导。

假设 r2c9(4)和 r9c6(4)同假，则由于 BUG + n 要求所有真数不同假，所以只得让 r7r9(8)为真，于是引出一条强制链，得到 r7c2(4)为真，所以此时可以删除 r7c6(4)。

接着我们假设 r2c9(4)和 r9c6(4)不同假，则它们形成强关系，于是可以看到一个环结构，依然可以得到删数 r7c6(4)。

所以不论哪种情况都可以删除 r7c6(4)，所以它就是这个技巧的结论。

10-4-6-2 XR + 3

3	8	1	2	3	5	2	2	4	6	4	6	4	6	4	6	4	6	4	6	4	6	4	6	4	6
5	4	5	2	6	1	2	9	8	3	7	9	5	6	4	8	3	7	9	5	6	4	8	3	7	9
9	2	3	7	4	6	8	5	4	6	9	8	5	4	6	9	8	5	4	6	9	8	5	4	6	9
2	1	4	3	5	6	4	7	9	8	3	5	6	4	7	9	8	3	5	6	4	7	9	8	3	5
1	5	7	8	9	4	6	2	3	1	5	7	8	9	4	6	2	3	1	5	7	8	9	4	6	2
4	6	3	1	8	9	7	5	2	4	6	3	1	8	9	7	5	2	4	6	3	1	8	9	7	5
4	7	8	9	5	6	2	3	1	4	6	9	8	5	7	2	3	1	4	6	9	8	5	7	2	3
1	3	2	4	5	6	7	8	9	1	3	2	4	5	6	7	8	9	1	3	2	4	5	6	7	8

如图所示，如果 r5c6(9)为真，则直接可以删除 r2c6(9)；当它为假的时候，我们就引出了一条毛边 r3c4(6)=r8c6(1)，按照图上给出的方向进行推理，最终我们就可以得

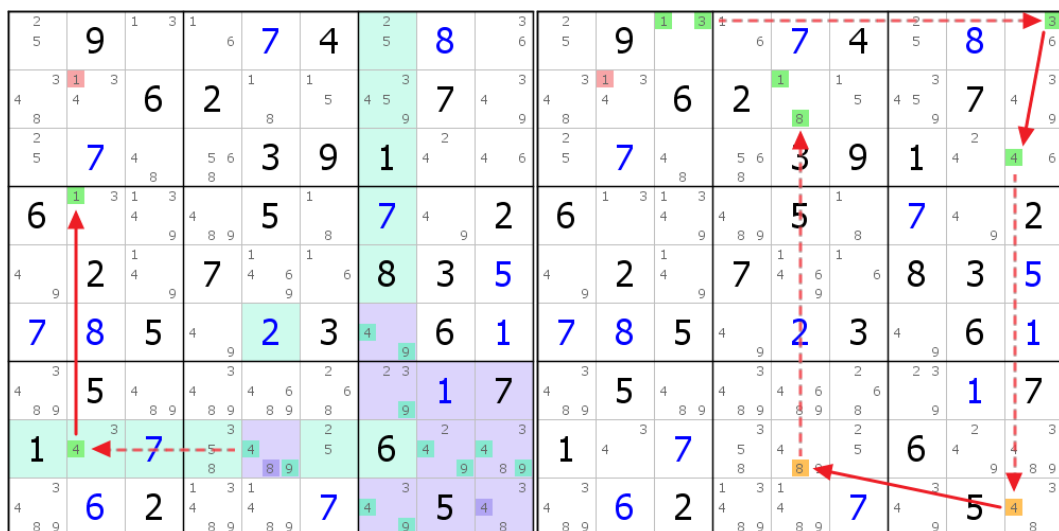
到删数结果。需要阐述的地方是上面的 ALS 区域和假 SDC 区域。

ALS 区域里，如果给出的 5 和 6 同真，则会导致 ALS 区域没有 5 和 6，致使填数少掉两种，2、7、9 又无法填满四个单元格，导致出错。当然，这里你也可以看互补的视角：**r2** 无法填入 6（1、3、4、8 是确定值，而原视角涉及的是 2、7、9，所以互补视角只涉及 5 和 6，而 5 把 6 该填的位置占了）；而假 SDC 区域里，如果 5 和 9 同假，则会使得区域里只有 3 和 6，且都不跨区，所以填数一定不相同，而三个单元格必须要填入三种不同的数字，而此处只剩下 3 和 6 两种数字可填，所以产生了矛盾。

当然，这个例子你也可以叫毛刺，如果你强调的是 **r5c6(9)** 的真假讨论的话。之所以称为毛边，是考虑到拓展矩形的真数的填数情况而设定的。如果同假，则 **r5c6(9)** 为真，而如果不同假，就形成强关系。所以其实这两种说法只是针对的侧重点不同罢了，没有必要纠结。

10-4-6-3 毛边空矩形欠一数对 (Bi-burred Empty W-Wing)

下面我们来看一点精彩的例子：毛边和空矩形欠一数对（当然，这个技巧也叫空矩形 W-Wing）到底会交织出什么样的逻辑呢？

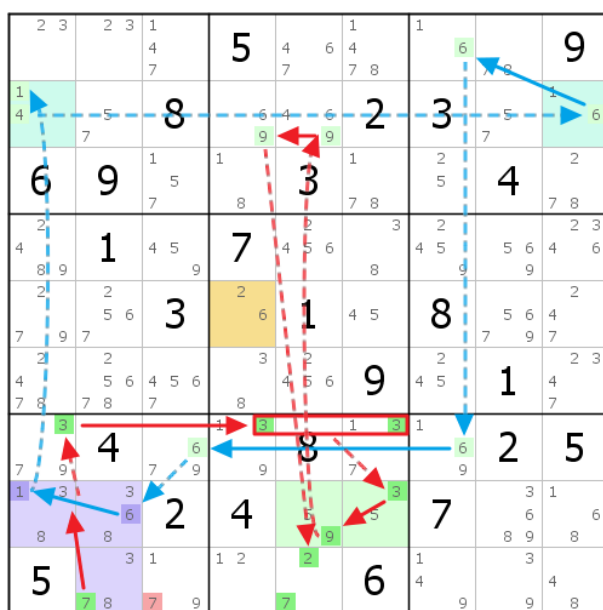


如左图所示。如果我们尝试让 **r8c5(8)** 和 **r9c9(4)** 同假，此时我们就可以得到 **b9** 的空矩形区域和两个关于 4 和 9 的双值格，于是空矩形欠一数对就这样诞生了。所以，**r8c2(4)** 便是此时的删数。注意，这是当 **r8c5(8)** 和 **r9c9(4)** 同假时才得到的删数，并不是全局的删数。接着，由于 **r8c2(4)** 为假，那么由于 ALS 的关系，**r4c2(1)** 为真。所以，此时可以删除掉 **r2c2(1)**。

接着看右图。由于同假是一种情况，那么反面情况自然就是不同假，即强关系了。所以我们必须利用上 **r8c5(8)=r9c9(4)**。于是我哦们能够找到上图的链结构，注意 **b3** 里有一个 ALS 区域。

可以看到，不论哪种情况，我们都可以得到 **r2c2(1)** 是删数的结论，所以 **r2c2(1)** 即为删数。

10-4-6-4 毛边环 (Bi-burred Continuous Nice Loop)



如图所示，这个例子可能也算毛边里比较难理解的例子了，但逻辑依旧很清晰。如果 $r9c2(7)$ 为假，则我们可以观察 AALS 区域 $\{r8c12, r9c2\}$ 里，立刻可以找到一个强关系 $r8c1(1)=r8c2(6)$ ，这样我们就可以利用这个强关系，进而形成一个环结构（蓝色的强弱关系画出来的这部分）。

得到这个结构后，我们就得到了环，所以顺带也就得到了这个环的一个删数 $r7c1(3)$ 。而之前说过，删数和结构是成弱关系的，所以它们可以用弱关系连起来。所以实际上我们开始的这两个强关系大致是这样：设 $r9c2(7)$ 为假，则 $r9c2(7)=\text{环结构}-r7c1(3)$ 。接着继续向下推导，我们可以得到一条完整的链，一直到 $r9c4(7)$ 。

这里需要阐述一下 $r2c4(9)-r9c5(2)$ 。如果它们同真，则 $c4$ 上 2 和 6 将无法填数，因为都被挤入 $r5c4$ 一个单元格里，显然它们是不能同时填入这个单元格的，所以矛盾。

最后，由于整条链结构（红色）成立，所以删数也就成立了： $r9c3(7)$ 。需要注意的是，如果我们把 $r9c2(7)$ 和 $r9c5(7)$ 用弱关系连起来的话，那这条链就已经变为了一个完整的环，不过遗憾的是，这个环只有 $r9c3(7)$ 一个删数，所以我把它画成了链。反正只有一个删数，就没必要画成环了。

Part 11 总结

至此，毛刺毛边的内容就全部结束了，而且，构造也就结束了，而且，链的内容我们就全部讲完了。

可以看到的是，链里给出的所有例子都非常的精彩，而且删数也非常值得去反复推敲，所以在自己做题的时候，希望你们能做到“记、观、用、活”这四个点。

记就是记忆，能记住这些链结构都怎么用，为啥这么用(why)，什么时候这么用(when)，在哪里可以用(where)，如何用(how)。

观就是观察，当我们能够熟练到快速理解和看懂整体逻辑的时候，我们就可以尝试自己去找了。因为熟练看懂逻辑时，你就已经掌握了众多链的使用方式，也就顺带拥有了如何寻找链的一些基础经验，我们就可以通过这些基础经验来寻找链结构。

用就是使用，当能做到观察到后，我们应当学会如何去使用它。这一点不同于观察，使用链和观察链是两回事，当观察到的时候，不一定能找到删数。而且一些环结构，它的删数有时候隐藏得很深（比如嵌入 AUR 之类的东西）。这个时候就必须得自己去逐个验证和得到，然后熟练于心。

活就是灵活，当我们能够做到熟练使用教程文档给出的使用方式后，就可以尝试去找一些教程里没有给出的强弱关系的证明方式和逻辑。我们在学习这些东西的时候，可以看到技巧是非常多，也非常复杂的。我们在教程文档里不可能也没有必要对每一个类型的技巧都给出一个示例，这样是不现实的，所以在做题的时候，我们利用我们学到的东西，可以得到一些我们学不到的东西，比如我们学习了 UR，我们可以将 UR 同时通过数和形进行拓展，进而得到新的致命结构（这一点将在后面的内容介绍到）；又比如我们学习了链的强弱关系的证明方式和过程，它用到了逆否命题，我们就可以通过一些新的东西，比如借助完全不相关的逆命题或否命题来论证一些我们从未尝试过的新的玩意儿；再比如我们学习了链列，我们就可以将它进行推广，放到环里，形成动态环来找删数，实际上链列也是一个环，除了之前我们说到的标准连续环结构，是必须让定义域区域缩减为强关系才能形成，那么一般情况下，它实际上就是一个动态的环结构，而且它的删数就恰好是所有弱关系都可以用到。

这些东西，都不应该全部由我给大家介绍，而是希望大家通过贯穿整个教程，来达到灵活运用效果。如果你做到了，那么恭喜你，大部分的数独题目都已经对你来说不难了；但是如果你做不到，或者接近于做到，那么也没有必要灰心和难受，成功没有捷径可走，但有工具可以帮你快速到达成功。只要你能灵活掌握，成功就不远了。

下一个部分将为你介绍两种不同于链的包装模式，它们基于共轭对和一些零散的结构。

第四篇章 包装技巧

本篇章将为你介绍的是两个包装技巧，这两种包装模式不同于链，而又和链很相似。这两种包装技巧有一个很新颖，也有一个很老旧，但都是数独发展这么多年来沉淀的厉害的技巧。

Part 1 涂色 (Coloring)

涂色 (Coloring) 是一种古老的技巧，它利用大量的共轭对来枚举盘面里的填数情况，进而得到一些矛盾来排除一些数字。

由于这个技巧稍微比较暴力一些，但运用了共轭对，所以我们此处作为简单描述，你只需要明白这个技巧的原理即可。这个技巧比较“守旧”，所以看起来有一些暴力，所以你如果不喜欢使用它，大可以跳过本节内容。不过这节内容介绍的技巧在一些软件里依然被使用到，例如 Hodoku。

1-1 单一涂色 (Simple Coloring)

1-1-1 涂色规则#1：色链原则 (Simple Coloring Type 1/Wrap)

2	3	¹ ₄	7	¹ ₄	6	8	9	5
5	9	7	2	8	¹ ₄	¹ ₄ 3	¹ ₄ 3	6
6	¹ ₈	¹ ₈	9	3	5	¹ ₄ 2	7	¹ ₄ 2
3	¹ ₈ 2	6	¹ ₄	9	¹ ₈ 2	5	¹ ₄ 2	7
7	5	¹ ₈	3	6	¹ ₈ 2	¹ ₄ 2	¹ ₄ 2	9
4	¹ ₂	9	5	7	¹ ₂	6	8	3
8	7	2	¹ ₄	5	3	9	6	¹ ₄
9	4	5	6	¹ ₂	7	¹ ₂ 3	¹ ₂ 3	8
1	6	3	8	² ₄	9	7	5	² ₄

如图所示，我们发现盘面之中有很多关于 1 的共轭对。比如 r17、c46、b28。我们知道，共轭对指的是两端有且仅有一个为真的两数。那么我们不妨将共轭对分成两种颜色，以标记出两种不同的填数情况，然后再针对这一种已经标好的填数情况，再观察其他共轭对是否有联系。比如图中 r1(1)和 b2(1)的共轭对就有联系，如果将 r1c3(1)涂第一种颜色，那么 r1c5(1)就只能涂另外一种颜色，那么 b2 里，r2c6(1)还是得涂第一种颜色（要保持共轭对两端涂色不同，表示不同填数情况）。

随之涂好一些候选数颜色后，我们发现，r2c6(1)和 r3c9(1)涂色不同，而且跨区。

这意味着这两个数有且仅有一个是对的。所以，r2c78(1)就不能有填数，否则将会和 r2c6(1)和 r3c9(1)其中一个重复，违背数独规则；当然，r3c23 <> 1 也是一样的思路。

这种结构使用的是最为基础的形式，称为**色链**（Simple Coloring Type 1），因为它用到共轭对串联起来的效果，就像链一样，把强弱关系串联起来。英文也称为 **Simple Coloring Trap**。

1-1-2 涂色规则#2：色分原则（Simple Coloring Type 2/Trap）

6	5	9	2	2	2	1	3	
			4	4	7	7	8	7
4	4	1	8	9	3	6	2	5
2		3	1	6	5		4	9
4	2	5	5	9	6	3	1	4
		7	8					7
3	6	4	7	1	2	5	9	4
		8						8
9	1	5	3	5	4		6	2
		7		8		7	8	
7	9	4	6	1	4	2	5	3
		8						
5	4	3	2	2	3	9	8	1
			4	4	7			
1		3	5	5	3	4	7	6
	8	2	8	9	9			

涂色法很有趣的地方在于，它还有更奇妙的逻辑。

如图所示，我们找出所有关于 8 的共轭对。分别图上不同的颜色，并按照刚才色链的方式，颜色要交替涂。随后我们发现，比如 c5, r6c5(8)涂第一种颜色，r9c5(8)就得涂第二种颜色。而走另外一个方向，r6 上，r6c7(8)也只能涂第二种颜色，r3c7(8)第一种颜色，r1c9(8)第二种颜色，r1c6(8)第一种颜色，c7c6(8)第二种颜色。

这个时候我们发现，r7c6(8)和 r9c5(8)是同一种涂色，这意味着同一种填数情况发生在同一个宫内，这明显是违背数独规则的，所以所有同一种涂色下的候选数全部删除。

这个技巧称为**色分**（Simple Coloring Type 2）。确实很实用，不过有一点暴力就是了。英语里也称为 **Simple Coloring Wrap**，不过我更喜欢叫 **Simple Coloring Map**，因为 Map 有涉及面较广的含义。

1-2 复合涂色 (Multi Coloring)

3	4	9	5	1 2	6	1 2	7	8
5 8	5 8	7	1 2	3	9	1 2 6	4	1 6
1	2	6	7	8	4	3	9	5
7	6	1 2	9	1 2	3	8	5	4
9	1 3 8	1 8	1 6	4	5	7	2	1 3 6
4	1 3 5	1 2 5	1 2 6	7	8	1 6	3 6	9
6	9	3	8	5	2	4	1	7
2	1 8	1 4 8	4 3	9	7	5	3 6 8	3 6
5 8	7	4 5 8	4 3	6	1	9	3 8	2

复合色链法比起色链法更难一些。

如图所示，将一些共轭对涂色，不过，b3 有 3 个 1，没办法形成共轭对，所以我们只能为两边的两种情况分别涂上两套不同的颜色。然后 b3 内的 1 就用“不同真”的方式连起来。

此时，图中有四种不同的涂色方式（绿浅、绿深、橙浅、橙深），也就是说图中有四种不同的填数情况，只是，b3(1)不同真，所以意味着“绿浅”和“橙浅”不同真。

- 如果是**绿浅为真**，那么由于绿浅和橙浅不同真的关系，橙色的深浅两种颜色（情况）下，只得橙深为真，因此可以删除 r5c23(1)；
- 如果是**绿深为真**，可以删除 r5c23(1)。

就绿色的情况而言，至少有一个情况成立，但都可以删除 r5c23(1)，所以 r5c23 <> 1。

这个思路就更加复杂一些了，用到了两种不同的涂色方案，并使用了 b3(1)这里用“不同真”的类似于弱关系的東西连接起来了。这种技巧称为**复合涂色 (Multi Coloring)**。

1-3 异数涂色 (Advanced Coloring/3D Medusa)

前面我们讲到了基本的涂色模式，不过它们仅针对于同数的共轭对而言。接下来我们来看一种运用于异数之间的共轭对（或双值格）的一种涂色技巧。由于涂色涉及同色和异色的删数模式，所以针对于这个技巧而言，就会有四种删数模式，不过同数和同色组合的这种类型是无法删数的，所以不存在，故只剩下三种类型。

1-3-1 同数异色类型 (3D Medusa Type 1)

3	4	6	2	<div>1 5 8</div>	<div>1 8</div>	7	<div>5 8</div>	9
8	9	5	<div>4 7</div>	<div>3 6</div>	<div>3 7</div>	2	1	<div>4 3</div>
2	7	1	<div>4 8</div>	<div>3 5</div>	<div>3 8</div>	9	<div>4 5 8</div>	<div>6 4 8</div>
6	3	2	5	<div>7 8</div>	4	9	<div>7 8</div>	1
9	5	8	1	<div>7 3</div>	2	<div>3 6</div>	4	<div>3 6 7</div>
7	1	4	<div>3 6 8</div>	<div>9 3</div>	<div>3 6 8</div>	<div>3 8</div>	2	5
<div>1 5</div>	8	3	9	2	<div>1 6 7</div>	<div>1 4 5 6</div>	<div>5 7</div>	<div>4 6 7</div>
<div>1 5</div>	2	9	<div>7 8</div>	<div>4 6</div>	<div>1 7 8</div>	<div>1 5 6 8</div>	3	<div>7 8</div>
4	6	7	<div>3 8</div>	<div>1 3 8</div>	5	<div>1 8</div>	9	2

如图所示，我们可以找到一大堆双值格和共轭对，并分散构成两种情况。

可以看到， $r3c5(3)$ 和 $r5c5(3)$ 都涂了颜色，但颜色不同，故属于两种不同的情况，而共轭对保证了必须有一种情况成立，所以这两个 3 里必须有一个数为真，所以显然， $r9c5$ 就不可能填 3 了，因为 3 属于哪种情况都无法填到这里来；同理， $r3c7(3)$ 的删数原理也是一样。

由于这个技巧从数、色、形三个维度对结构进行扩展，所以这个技巧也被叫做**立体美杜莎 (3D Medusa)**，所谓的立体就取自三维立体结构之意。

1-3-2 异数同色类型 (3D Medusa Type 2)

7	5	1	9	² ₄	² ₆	² ₄	3	² ₆	8
3	2			5	1	4	7	9	
4	9				7	5		1	
1	6	⁴ ₇	5	9	⁴ ₈	⁷ ₈	3	2	
5	⁴ ₇	2	1	⁴ ₈	3	⁷ ₈	9	6	
9	8	3		7		1	4	5	
6	¹ ₃		7	¹ ₃		2	8	4	
8	¹ ₇		4	¹ ₂		6	5	3	
2	⁴ ₅					9	1	7	

如图所示，我们针对于共轭对和双值格进行涂色，随即发现 **r9c6** 出现了两个相同涂色的单元格。显然，既然属于同一种情况，那么它们的所有填数真假情况就得是一样的。但是显然，**r9c6** 不可能让 5 和 8 都为真，所以只能都为假。所以所有橙色的数字必须全为假，故这些数字全都可以删除。

1-3-3 异数异色类型 (3D Medusa Type 3)

³ ₅	³ ₅	³ ₅	4	³ ₅	6	8	2	1	
8	² ₃	4		² ₃	1	6	7	9	
¹ ₆	¹ ₂		² ₃	8	7	4	5	3	
2	¹ ₅	⁵ ₆	⁶ ₉	¹ ₅	8	¹ ₅	3	4	
¹ ₆	¹ ₃	8	² ₃	¹ ₂	4	7	9	² ₅	
9	4		² ₃	¹ ₂	5	¹ ₂	6	8	
7	6	² ₃	1	4	9	² ₃	8	² ₅	
4	8	² ₃	5	6	² ₃	9	1	7	
³ ₅	9	1	8	7	² ₃	² ₃	4	6	

如图所示，我们涂色后发现，在 **r4c4** 里，同一个单元格具有两种不同的涂色，这意味着这一个单元格只能是 6 或者 9。所以，这一个单元格不能填入 7，把 7 删除掉即可。

这个逻辑看起来似乎比较好理解，所以我们就不多啰嗦了。不过，这种类型还有一种删数的形式。

	3	2 3	2		1 2		6	3	
4	4 5	8	4 5	9	5			4 5	4 5
7 8	8				8		7 8	7 8	7 8
4	4 5		1	3	5	6	2	4 5	9
7 8	8				8		7 8	7 8	
	3		2	2		1	1	3	
	6	9	5	5	4	7	5	5	5
8			8	8			8	8	8
2		3		7		3			1
		6				4	5	5	
			6				8	8	
1	3		9		1 2				6
7			8	2	5	3	4	9	
				5					
5	1			6	1		7	2	3
	4								
		4		8	9				
1	1 2		2	4	7	3	1	6	2
	8 9	5	5				5		5
		8					8 9	8	8
4	6	2		1	2		3	4	2
8	5 6	8	7		6	9		5 8	4 5
				8	8		8	8	8
1	1 2		3	2	2		1	1	2
4	6	4		8	6	5		4	4
	8 9	8			8		8 9	7 8	7 8

如图所示，可以看到 **r1c2(3)**（或者 **r5c1(3)**）和 **r3c1(6)**具有不同的涂色。既然两种不同的涂色就表示两种情况里必须成立一个，所以 **r3c1(3)**是不允许填入的，否则它将直接破坏两种情况，导致无法找到合适的填数情况，出现矛盾，所以删除 **r3c1(3)**。

这种结构很类似于不连续环的逻辑，对吧。

至此，涂色的基本内容就介绍完了。下面我们来看另外一种包装技巧：袋鼠。

Part 2 袋鼠 (Unknown Cover/Kangaroo)

袋鼠 (Unknown Cover/Kangaroo)，原称为**代数法**，指的是假设某个单元格填入的未知数来获取一定填数逻辑的技巧。袋鼠和代数谐音，所以也称为袋鼠。但由于 **Algebra** (代数) 一词跟这里的代数的使用不太相似，所以没有启用这个名称，所以本文档采用谐音叫法，即袋鼠。

不过，袋鼠这种技巧更像是一种解题视角，因为做题过程之中我们会通过这种独特的视角看到我们平时完全看不到的东西。

2-1 先来看看一般化的版本

2-1-1 辅助数类型

3	5	1	6	² x ₈	7	9	² 4 ₈	² 4 ₈
² 8 ₉	² 8 ₉	6	² 4 ₈ 9	5	² 4 ₈ 9	7	3	1
4	7	² a ₈ 9	1	² y ₈ 9	3	6	² x ₈	5
5	² 4 ₈	² 3 ₉	² 4 ₈ 9	7	1	8	² 4 ₈ 9	6
² 8 ₉	6	7	3	² a ₈ 9	² 4 ₈ 9	1	5	² 4 ₈ 9
1	² 4 ₈ 9	² 8 ₉	² 4 ₈ 9	6	5	3	7	² 4 ₈ 9
7	¹ 8 ₉	5	¹ 8 ₉	4	6	2	¹ 8 ₉	3
6	¹ 2 ₈ 9	4	7	3	² 8 ₉	5	¹ 8 ₉	¹ 8 ₉
² a ₈ 9	² 3 ₈ 9	² 3 ₈ 9	5	² 1 ₈ 9	² 4 ₈ 9	4	6	7

如图所示，我们优先假设 $r5c5 = a$ ，然后分别假设 $r1c5$ 和 $r3c5$ 为 x 和 y ，就可以发现， x 不包含候选数 9，而对应地， a 和 y 里必须有一个数字是 9。所以， $r3c8$ 不含数字 9，而且 $r3$ 缺少的数字和 $c5$ 缺少的数字完全一样，这就说明了 $r3c3$ 只能是含有数字 9 的 a ，而 $r3c8$ 只能是没有 9 的 x 。我们得到了 $r3c3 = r5c5 = a$ 之后，可以进一步使用 $c1$ 的排除，得到 $r9c1 = a$ 。

3	5	1	6	x ₈ ²	7	9	² _{4 8}	² _{4 8}
² _{8 9} b	² _{8 9} c	6	² _{4 8 9}	² _{4 8 9} 5	7	3	1	
4	7	² _{8 9} a	1	² _{8 9} y	3	6	x ₈ ²	5
5	^{2 3} _{4 9}	^{2 3} ₉	² _{4 9}	7	1	8	² _{4 9}	6
² _{8 9} c	6	7	3	² _{8 9} a	² _{4 8 9} 4	1	5	² _{4 9} b
1	² _{4 8 9}	² _{8 9}	² _{4 8 9}	6	5	3	7	² _{4 9}
7	¹ _{8 9}	5	¹ _{8 9}	4	6	2	¹ _{8 9}	3
6	¹ _{8 9} b	4	7	3	² _{8 9}	5	¹ _{8 9}	¹ _{8 9}
² _{8 9} a	^{2 3} _{8 9}	^{2 3} _{8 9}	5	1	² _{8 9} b	4	6	7

接着，我们得到 $r9c1$ 是 a 后，我们继续对 $b8$ 执行假设。假设 $r9c6 = b$ ，则根据 $b7$ 的宫排除，得到 $r78c2$ 里必须要有一个 b ，就相当于形成了一个以未知数形式呈现的区块结构。

由于 $r2c2$ 只有 2、8、9，而既然 $r2c2$ 不能是 a ，也不能是 b ，那我们只好假设 $r2c2$ 填的是 c 。此时我们就确定了 a 、 b 、 c 是 2、8、9 的数字，互不相同，这样一来也顺便得到了 $r2c1 = b$ 、 $r5c1 = c$ 。

接着我们观察 $r5$ ，此时可以发现， b 只能放在 $r5c9$ ，而 $r5$ 一共有 2、4、8、9 四种数字没填，其中 a 、 b 、 c 也都被我们填入到盘面里，所以只能 $r5c6 = 4$ 。

可以看到，这个示例里，我们借助了功能不大的辅助数，来得到 $r9c1 = a$ 的结果后，继续向下推导，并得到最终的结果。

这就叫做袋鼠，利用假设未知数的形式来代入试填，并得到结论的方式。这个例子还能继续往下通过袋鼠标注全盘，进而得到 a 、 b 、 c 的具体数值结果，这里就不讲了，你可以尝试继续做下去。在完成后就会发现，实际上这里辅助的 x 和 y 也是有用处的，只是此处得到结论后我们没有继续往下做了，所以看起来没有什么用途。

接着我们来看第二种用法。

2-1-2 数组类型

1	7	³ _{6 9}	³ ₉ a	2	8	4	³ ₆	5
8	³ ₉ a	5	4	³ _{6 9} 6	³ _{6 9} b	7	2	1
^{2 3} ₆	4	^{2 3} ₆	7	1	5	9	8	³ ₆
^{2 3} _{5 9}	^{2 3} ₉	7	6	³ ₉	1	^{2 3} ₅	4	8
³ _{5 6 9}	8	4	2	7	³ ₉	³ _{5 6}	1	³ _{6 9}
^{2 3} _{6 9}	1	^{2 3} _{6 9}	5	8	4	^{2 3} ₆	7	^{2 3} _{6 9}
7	5	8	³ ₉ b	³ _{6 9}	2	1	³ _{6 9}	4
4	³ _{6 9}	1	8	5	7	^{2 3} ₆	³ _{6 9}	^{2 3} ₆
^{2 3} ₉	^{2 3} _{6 9}	^{2 3} ₉	1	4	³ _{6 9}	8	5	7

如图所示，假设 $r2c2 = a$ (a 是 3、9 之一)，则显然 $r2c56$ 里只能是 6 和 9，鉴于例子用不上 6 作为假设，我们就不假设为未知数了，那么 $r2c56$ 只能填入 6 或 b 。于是 $r1c4 = a$ 、 $r7c4 = b$ 。

1	7	³ _{6 9}	³ ₉ a	2	8	4	³ ₆	5
8	³ ₉ a	5	4	³ _{6 9} 6	³ _{6 9} b	7	2	1
^{2 3} ₆	4	^{2 3} ₆	7	1	5	9	8	³ ₆
^{2 3} _{5 9}	^{2 3} ₉	7	6	³ ₉	1	^{2 3} ₅	4	8
³ _{5 6 9}	8	4	2	7	³ ₉	³ _{5 6}	1	³ _{6 9}
^{2 3} _{6 9}	1	^{2 3} _{6 9}	5	8	4	^{2 3} ₆	7	^{2 3} _{6 9}
7	5	8	³ ₉ b	³ _{6 9} a	2	1	³ _{6 9}	4
4	³ _{6 9}	1	8	5	7	^{2 3} ₆	³ _{6 9}	^{2 3} ₆
^{2 3} ₉	^{2 3} _{6 9} a	^{2 3} ₉	1	4	³ _{6 9} 6	8	5	7

接着，我们以 $r2c2 = a$ 向下推理。由于排除操作，我们最终确定 a 只能放在 $b7$ 里的 $r9c13$ 里。接着使用排除，得到 $b8$ 此时只能把 a 放在 $r7c5$ 里（ $r7c4$ 是上一轮推导得到的 b ，显然 a 和 b 不能填入一样的数字）。

那么观察 $b8$ ，已经出现 a 和 b 了，而 a 和 b 又是 3 和 9，所以 $r7c45$ 是 3、9 的数对结构，故只有 $r9c6 = 6$ 了。

这一则示例也是比较精彩的，它破解了一些需要链的棘手情况。

2-1-3 确定值类型

9	¹ ₇ a	8	^{1 2} ₇	3	5	6	² _{4 7}	² _{4 7}
¹ _{4 7}	2	¹ _{4 7}	9	6	¹ ₇ a	3	5	8
6	3	5	² ₇	4	8	1	² ₇	9
¹ _{4 5 7}	6	2	^{1 3} _{4 7}	8	9	⁴ ₇	^{1 3} ₅	¹ ₅
¹ _{4 5 7}	8	3	¹ _{4 6 7}	^{1 5} ₇	^{1 2} _{4 7}	⁴ ₇	9	^{1 2} _{5 6}
¹ _{4 5 7}	9	¹ _{4 7}	^{1 3} _{4 6 7}	^{1 5} ₇	^{1 2} _{4 7}	8	^{1 2 3} _{5 6}	^{1 2} _{5 6}
8	¹ _{4 7} b	6	5	2	¹ _{4 7} c	9	¹ _{4 7}	3
3	5	¹ ₇ a	¹ _{4 7} b	9	6	2	8	¹ _{4 7}
2	¹ _{4 7} c	9	8	¹ ₇ a	3	5	6	¹ _{4 7}

如图所示，假设 $r1c2 = a$ ，并假设 $r7c2 = b$ 、 $c9c2 = c$ （此时 a 、 b 、 c 是 1、4、7，且互不相同）。于是，我们根据排除可以得到 $b7$ 里 a 的位置在 $r8c3$ ；同理， $r2$ 上填 a 的位置也可以确定，在 $r2c6$ 。接着，由于 a 是 1 和 7 的其一（不能是 4），而 $r9$ 恰好缺少 1、4、7，且此时已经 $r9c2 = c$ 了，那么只能把可以是 4 的 b 放在 $r9c9$ ，所以 $r9c5 = a$ 。而且，由于 $r7c2 = b$ 、 $r9c5 = a$ ，所以它们的交集 $r7c6$ 就只可以是 c 了，且 $r8c4 = b$ 。

9	¹ ₇ a	8	^{1 2} ₇	3	5	6	² _{4 7}	² _{4 7}
¹ _{4 7}	2	¹ _{4 7}	9	6	¹ ₇ a	3	5	8
6	3	5	² ₇	^{b=4} ₄	8	1	² ₇	9
¹ _{4 5 7}	6	2	^{1 3} _{4 7}	8	9	⁴ ₇	^{1 3} ₅	¹ ₅
¹ _{4 5 7}	8	3	¹ _{4 6 7}	^{1 5} ₇	^{1 2} _{4 7}	⁴ ₇	9	^{1 2} _{5 6}
¹ _{4 5 7}	9	¹ _{4 7}	^{1 3} _{4 6 7}	^{1 5} ₇	^{1 2} _{4 7}	8	^{1 2 3} _{5 6}	^{1 2} _{5 6}
8	¹ _{4 7} b	6	5	2	¹ _{4 7} c	9	¹ _{4 7}	3
3	5	¹ ₇ a	¹ _{4 7} b	9	6	2	8	¹ _{4 7}
2	¹ _{4 7} c	9	8	¹ ₇ a	3	5	6	¹ _{4 7}

接着，观察 $b2$ 。由于 $r8c4(b)$ 对 $b2$ 的排除，就会发现 $b2$ 里没有位置可以填下 b 了。那岂不是做错了？没有。回想一下， a 、 b 、 c 是 1、4、7，而 4 还有一个确定值位于 $b2$ 里，那就是 $r3c5$ 。既然 b 放不到这些空格上，那么这个 4 就必须是 b 的数值结果了，也就是 $b = 4$ 。否则 b 将没有位置放，出现矛盾。所以，我们就确定了 $b = 4$ 的结论，刚才假设的所有位置，只要是 b 的地方，都可以被替换为 4，即 $r7c2 = r8c4 = r9c9 = b = 4$ 。

(r9c9 因为在推理里用不上, 所以没有标注)。

可以看到这一则示例用法更为灵活, 它直接把填数的结果给替换成了真实的数值。

2-1-4 什么? 直观?

可以看到, 前面的示例除了简单枚举一下单元格的填数情况以外, 基本上就用不上候选数了, 所以这个技巧基本上都用不着候选数。而实际上, 这个技巧本来也就是为了直观而生的, 一般来说, 这个技巧都使用直观视角解题; 当然, 有些时候可以配合候选数全标形式来辅助数数操作。

而且可以发现, 这些盘面存在一定的共性: 只要是涉及三到四种数字在盘面里确定值比较少的, 或者说候选数比较多的, 我们就可以尝试使用袋鼠, 因为这类型的题目使用袋鼠总能得到我们想要的结果, 毕竟这些数字之间的提示信息比较多, 但我们在实际做题过程之中, 使用的之前的技巧并没有真正用到它们。

2-2 常见技巧的袋鼠视角

在数独里, 有一些技巧依然可以使用袋鼠的视角进行观察和理解, 而这种理解思维比起普通版的话, 要更新颖一些。而且在上一节里, 我们说到了, 袋鼠一般可以用于直观层面, 所以我们下面的示例都不标注候选数, 你可以尝试来理解和感受一下, 直观的袋鼠到底是什么样的。

2-2-1 欠一数对 (ALP in Unknown Cover)

6	8	9	4	2			3	
2			7	8	9			
			6	3		9	2	8
	6	1			4	2	8	
		8	1		2			
			8	6		3	9	1
8			2	4				9
			5		6	8		3
	7	6		1	8	5	4	2

如图所示。假设 $r9c1 = a$, 则通过唯余操作, 得到 a 是 $\{39\}$ 之一 (即 a 是 3 和 9 的其一)。对 b4 进行排除, 由于 r6 上有 3 和 9 的确定值, 所以不管 a 是多少, 最终 a 只能填到 r5c2。而由于 a 是 3 和 9 的其一, 所以观察 r5 可以发现填入 2 的位置只剩下 r5c6, 故 $r5c6 = 2$ 。

如果你打开软件的候选数开关, 查看全盘的全标候选数的话, 你就会发现, 这就是一个完美的、正常的欠一数对。所以, 它是欠一数对的袋鼠视角。

2-2-2 W-Wing (W-Wing in Unknown Cover)

	1	2	9		8	6	3	4
				1	4	5	2	9
	9	4		6		1	8	7
		3	1			9	4	5
9		1	4			2		8
2	4				9	3		1
1				4		8	9	
8	2	7	6	9	1	4	5	3
4		9	8		5	7	1	

假设 $r3c6 = a$ ，则 a 是 2 和 3 的其一。对 $b5$ 进行关于 a 的排除，我们发现，因为 $r6$ 上已经有 2 和 3 的确定值，所以 2 和 3 只能填到 $r45c56$ 。而因为 $r3c6 = a$ ，所以 $r45c5$ 有且仅有一格是 a 。接着观察到， $r9c5 = \{23\}$ ，而因为 $r45c5 = a$ ，所以 $r9c5$ 只能填 b （其中 b 是不同于 a 的、属于 $\{23\}$ 的另一个数）。这个时候，我们确定了 $r3c6$ 和 $r9c5$ 其实是一个跨区数对。所以， $r7c6 \neq ab$ ，而它们是 $\{23\}$ ，所以 $r7c6$ 只能是非 2 和非 3 的另外的数，即 $r7c6 = 7$ 。

全标之后你就会发现，它就是一种特殊的 W-Wing 结构（可能中间带有区块，但是 W-Wing 可以走两个方向）。

2-2-3 死环 (Guardian Pair in Unknown Cover)

2	6	8	7	9	5	3	4	1
9			6			2	7	
		4	1			9	6	
				7		8	3	4
		3				7	5	
						1		6
		6		8		5		7
	8	7				4		3
3				1	7	6	8	

如图所示，我们假设 $r9c9 = a$ ，则 a 是 $\{29\}$ 的其一。如果 $r9c9$ 是 a 的话，就可以

得到 $r5c9 = b$ 、 $r6c8 = a$ (b 不同于 a 且均是 2 和 9 其一)。而观察 $b1$ ，我们发现 $b1$ 有确定值 2 和 9，所以 $r2c3 \neq a$ ，所以观察 $c3$ ，只可能是 $r4c3 = a$ 。

此时我们可以得到， $r5c9$ 和 $r4c3$ 是关于 2、9 的跨区显性数对。所以 $r5c2 \neq 29$ ；而最后观察 $c3$ ，填入 1 的位置只剩下一处： $r2c3$ ，所以 $r2c3 = 1$ 。

2-2-4 远程数组 (Remote Subset in Unknown Cover)

实际上，远程数对由于涉及的数字很简单，所以袋鼠依然可以运作。而且，袋鼠可以用于多种数字，所以不只是远程数对，远程三数组等等也都可以出现。

2-2-4-1 远程数对 (Remote Pair in Unknown Cover)

1	7	8	6		9		5	
9	3	4	1	5	a	6	b	7
2	5	6	7		3		1	
7	9	3	5	6	b	a	4	1
6	4	1	a	3	7	5	9	b
8	2	5	9	1	4	7	3	6
5	6	7	3		1		a	
4	1		b	7	5		6	
3	8		4		6	1	7	5

如图所示。我们假设 $r7c8 = a$ ，通过唯余操作，得到 a 是 {28} 之一，可以以此得到 $r2c6$ 、 $r4c7$ 、 $r5c4$ 都是 a ； $r2c8$ 、 $r4c6$ 、 $r5c9$ 、 $r8c4$ 都是 b (b 不同于 a 且都是 {28} 之一)。所以，图中就有相当多的跨区数对了，这里就不啰嗦有哪些了，删掉所有红色单元格上的 2 和 8，然后与此同时可以得到 $r7c5 = 9$ 。

2-2-4-2 远程三数组 (Remote Triple in Unknown Cover)

3	4	8	9		7			5
5	2	6	1	3	8	4	9	7
9	7	1		4	5	8	3	
7	a	2			9	5		3
4	8	9	5	7	3			
b	5	3				7		9
a	3	4	7			9	5	8
2	b	5	8	9	4	3	7	a
8	9	7	3	5	a	b	2	4

如图所示，我们先假设 $r4c2 = a$ (a 是 $\{16\}$ 其一)，于是可以连续得到一些填数，此时其实已经形成了远程数对，得到 $r6c6 \leftrightarrow 16$ (即填 2) 的结果，不过这里我们还有结论，所以不着急出数。

3	4	8	9	b c	7		a	5
5	2	6	1	3	8	4	9	7
9	7	1		4	5	8	3	b c
7	a	2			9	5		3
4	8	9	5	7	3		a	b c
b	5	3				7		9
a	3	4	7	b c		9	5	8
2	b	5	8	9	4	3	7	a
8	9	7	3	5	a	b	2	4

接着，由于 $b8$ 只有 1、2、6 三个数没有填了，那么我们假设 a 、 b 、 c 是 1、2、6，且互不相同。基于这一点，我们显然可以知道， $r7c5$ 是 b 和 c 的其一 (图上的 “|” 表示或的意思，也就是 b 和 c 的其一)；同理，我们依然可以假设 $c9$ 上剩余两格分别都是 b 和 c 的其一。

此时，继续观察 $b5$ 和 $b6$ 。由于 $b5$ 可以确定 a 的位置形成了区块，所以 $b6$ 也可以形成 a 的区块结构；同样， $b3$ 也有 a 的区块，此时它们构成复合区块结构，所以 $r1c5 \leftrightarrow a$ ，故也只能填入 b 和 c 的其一。此时观察 $c5$ ，可以发现有两处单元格都是 b 和 c 的其一，而由于两个单元格同列，所以它们构成数对。

此时并没有结束。由于 $r4c1 = a$ 的关系，三个单元格的交集 $r4c5$ 是这个跨区三数组的交集（把 a 也算在结构内，就是三个单元格填 a 、 b 、 c 三种数字了，所以可以看作跨区的三数组），所以 $r4c5 \in abc$ ，于是这一个单元格就出数了。于是题目瓦解。

可以看到，这一种利用方式也是非常神奇的，除了远程数对的链式结构以外，还有现在发现的“叠加影响”形成的**远程三数组（Remote Triple）**结构。

2-2-5 唯一矩形（UR in Unknown Cover）

这个示例比较麻烦一些，们依然使用候选数视角给大家标注，方便对比。

1	³ 5 8 9	^{5 6} 9	^{2 3} 6 8	7	² 5	³ 5 8	^{2 3} 4 6	⁴ 8 6
³ 7 8	4	^{5 6} 7	^{2 3} 6 8	^{5 6} 8	1	³ 5 7 8	9	² 6 7 8
2	⁵ 7 8	^{5 6} 7	³ 6 8	4	9	^{1 3} 5 7 8	³ 6 7 8	¹ 6 7 8
	¹ 7 9	^{7 9} 7 9	8	4	3	6	2	¹ 7 9
5	6	2	9	1	7	4	8	3
4	¹ 7 9	³ 6 8	3	5	2	8	¹ 6 7 9	⁶ 7 9
⁶ 7 9	⁵ 7 9	1	7	^{5 6} 8	3	⁴ 7 9	² 4 8 9	² 4 8 9
³ 7 9	^{2 3} 6 8	^{4 5 6} 7 9	² 6	8	^{4 5} 7 9	³ 1 7 9	² 7 9	² 7 9
³ 7 8	^{2 3} 7 8	⁴ 7	1	9	⁴ 6	^{2 3} 7	⁵ 6	5

如图所示，我们假定 $r4c1 = a$ （显然， a 是 7 和 9 的其一），同样我们设定字母 b 和 c 属于 $\{179\}$ 之一，并假设它们互不相同（此时一定要注意，因为三个数互不相同，而 a 不包含数字 1，所以在 b 和 c 里，一定有一个数字是 1）。那么，因为 $r4c1$ 是 a 的关系， $r4c29$ 和 $r6c2$ 都只剩下 1 和 $\{79\}$ 之一；换句话说，它们的填数实际上就是 b 或 c 的其一。

1	³ 5 8 9	^{5 6} 9	^{2 3} 6 8	7	² 5	³ 5 8	^{2 3} 4 6	⁴ 8 6
³ 7 8	4	^{5 6} 7	^{2 3} 6 8	^{5 6} 8	1	³ 5 7 8	9	² 6 7 8
2	⁵ 7 8	^{5 6} 7	³ 6 8	4	9	^{1 3} 5 7 8	³ 6 7 8	¹ 6 7 8
^a 7 9	¹ 7 9	^b 7 9	8	4	3	6	2	¹ 7 9
5	6	2	9	1	7	4	8	3
4	¹ 7 9	^c 7 9	3	5	2	8	¹ 6 7 9	^{!b} 7 9
⁶ 7 9	⁵ 7 9	1	7	^{5 6} 8	3	⁴ 7 9	² 4 8 9	² 4 8 9
³ 7 9	^{2 3} 6 8	^{4 5 6} 7 9	² 6	8	^{4 5} 7 9	³ 1 7 9	² 7 9	² 7 9
³ 7 8	^{2 3} 7 8	⁴ 7	1	9	⁴ 6	^{2 3} 7	⁵ 6	5

显然，我们通过刚才的假设，因为此时的 b 和 c 可以认为是等价的两个变量的表示符号，所以随便假设 $r4c2$ 是 b 还是 c 都行。那么我们假设为 b 继续推导。

由于 $r4c2$ 是 b ，所以 $r4c9$ 只能是 c （字母 a 和 b 在这一行已经用过了，这三个字母代表的是三种完全不同的数字，所以只能是 c 了）。同理 $r6c2$ 也只能是 c 了。我们此时着眼于 $r46c29$ 四个单元格上。显然，如果 $r6c9$ 是 b 了，即使字母代表的数字可变，但当前状态下， $r46c29$ 构成了关于 b 和 c 的唯一矩形，形成了致命形式，所以 $r46c29$ 此时应当删除数字 b 。可是 b 是多少呢？这一点我们可不知道。不过，我们可以通过假设的字母所给出的范围来间接确定删除的数字可以是哪个。

因为 b 和 c 在此时的假设下是等效的，所以这个唯一矩形的构型构成了关于 b 和 c 的致命形式（因为 b 和 c 不相同，显然它们形成了致命形式）。那么此时删除的数字就变为 c 了。所以确实确定不了吗？还没完呢。别忘了最开始的假设，我们以为的 b 和 c 里一定有一个数字是 1，这也就意味着我们刚才所有可能成立的两种假设下，暗含了一点： $r6c9$ 不应该是 1。因为 $r6c9$ 一定只能填入的是 b 和 c ，仅此两种情况；而它俩一定有一个数字是 1，所以 1 是我们一定可以确定可以被删除掉的数字。所以， $r6c9 \neq 1$ ，这就是这个题目的结论（删数如前一个图所示）。

我们再来看一则示例。

7	5	¹ ₄	¹ ₄	8	2	3	¹ ₆	¹ ₄	⁶ ₉
¹ ₄	2	8	6	¹ ₃	¹ ₃		¹ ₉		5
¹ ₆	¹ ₆	3	¹ ₄	5	¹ ₇	8	2	¹ ₄	¹ ₇
8		³ ₇	2	⁵ ₇	4	³ ₅	1	³ ₅	⁶ ₉
¹ ₃	¹ ₄	5	¹ ₃	¹ ₃	8	⁴ ₆	7	² ₉	
¹ ₃	¹ ₄	6	2	¹ ₃	¹ ₅	⁴ ₅	⁵ ₉	³ ₅	8
¹ ₅	¹ ₆		8	2	¹ ₅	⁵ ₆	4	¹ ₇	³ ₆
¹ ₄	8	7	3	¹ ₆	¹ ₄	⁵ ₆	2	¹ ₅	¹ ₆
2	¹ ₆	¹ ₄	¹ ₅	7	¹ ₅	⁵ ₆	8	¹ ₆	³ ₉

如图所示，如果我们尝试假设 $r6c1 = 3$ ，则由于 $b6$ 的 3 的候选数分布情况，可以得到 $r5c5 = 3$ 。此时，我们发现 $r5c4$ 只有候选数 1 和 9，不论我们假设它填入什么数，最终 $r5c1$ 和 $r6c5$ 都会得到与之对应的另外一个数。如图所示，我们通过代数法，可以轻松得到 $r56c15$ 形成关于 3 和 b 的致命形式。所以假设的候选数 $r6c1(3)$ 应当被删除。

当然，还有很多结构都可以采用袋鼠的视角进行观察和推导，不过这里就不再一一列举了。

至此，本篇章的内容就结束了，下一个篇章将是本文档里最难的一部分内容。

第五篇章 令人畏惧的技巧

本篇章将为大家介绍的是数独技巧里最难的四大技巧板块：致命结构、鱼（链列）、飞鱼导弹和网。这四个板块的难度可能超乎想象，可能刚开始学习，学会了，第二天就忘掉逻辑和证明了。所以这些技巧都值得反复去推敲，而且作用很大。

Part 1 复杂致命结构 (Complex Deadly Pattern)

1-1 致命结构的确切定义

在之前的内容里，我们简单描述了有关致命结构技巧的基本原理和运用，而在之前，它具有两种不同的证明致命形式的方式，一种是类似于数组置换产生两种填法的方法，例如唯一矩形和唯一环；另外一种则是类似于某一个行或列的涉及的单元格左右或上下对应交换形成致命形式的，例如拓展矩形。

那么，致命结构的确切定义呢？我们从这两种类型里提取出相同的方式：**当所涉及的结构内部的填法相对于盘面而言不是唯一的填法的（即包括多种填法和无解填法，多种填法以 UR 为代表，无解填法以 BUG 为代表，即怎么填最终都会矛盾），而结构对盘面其余任何位置都是孤立的，即不会导致任何外界其它单元格的填数影响的，称为致命结构。**

那么，以此我们将产生出非常多的变体形式的致命结构。接下来我们就来看看到底都有什么类型。

1-2 致命结构的致命类型探讨

1-2-1 数组置换类型

思考一下 UR 形成致命形式的原理：内部可以发生交换，进而产生两种不同的、不造成外部影响的填数模式。那么我们试着把 UR 和 UL 这两种都是一样的变换形式的技巧结合在一起，看它能形成什么样的形状。

7	5 8 9	5 9	6	2 5 9 8	2 5 9 8	3	4	1								
1	4	6	5 3	5 3 1	2 5 7	9	2 5									
2	1 5 9	1 5 9	4	1 5	6	8	5 7									
1	6 9	7	4	2 5 9	3	2 5 6	8									
3	4 8 9	8 9	6	2 5 9	1	2 3 5 7	2 5									
5	3 6	2	8	1	7	4 9	6 4 5						4 9		4 9	
1	6 9	2	1 5 9	8	4 6 7	5 9	3		6 9				4 6 4 9			
3	6 9	7	5 3	2 3 5	2 3 5	4 6	8	1	6 9				4 6		4 9	
4	1 5 3	8	1 5 7	9	2 5 7	2 5	6									

如左图所示，观察 c1，我们发现 6 存在共轭对，这意味着，r78c1 有且仅有一格是 6，那么，另外一格都不应为 9。那么我们接下来就要说明的就是，为什么另外一个单元格不应该是 9。

如果 r78c1 有一个是 6，而且此时另一格为 9 的话，算上 r6c79、r7c67、r8c69 这六个单元格的话，结构一定是致命的。我们此时把 r78c1 看成 6、9 数对。我们看成 6、9 数对后，观察结构所在的所有区域（r678c1679b6789）。因为结构不是 UR 那样规则的结构，所以我们只能尝试去利用假设的方式来看。r6、c79、b69 内都有 4、9 数对、c6、b8 内都有 4、6 数对、c1、b7 内 6、9 数对、r78 有 4、6、9 三数组，即如右图所示。

显然，结构里必定会产生两种相反的填数模式。但是显然，管它怎么填，这个结构涉及的各个区域上都已经产生了固定的数对或更高阶的数组。所以不管怎么交换实际上都是没有实质性效果的，完全影响不到其它位置。但是，就内部有不同的填数模式而言，就相当于已经产生了两种形式的填数模式了。而纵观整个盘面，我们把刚才结构存在的区域下的所有数组删数都清除掉，并把两种填数情况分裂为两个盘面对比来看的话，那么它们的剩余盘面则是完全一致的。换句话说，这两种不同的填法都直接可以得到同一个“剩余盘面”。数独唯一解，要求每一个单元格都只可以有一种填法（也就是整体的题目的每一个单元格来说，答案上只可能是唯一的一个数，别无其他可能），这样一来，结构就形成致命形式了（虽然不知道它这种致命形式到底叫什么名字）。

这种结构实际上叫做**双值格致命结构**（Bi-value Deadly Pattern），因为它对应的致命形式，每一个单元格都是双值格。那么有没有**多值格致命结构**（Multi-value Deadly Pattern）呢？是有的，不过我们这一节不介绍。

同时，我们得到了一个结论：**如果一个盘面存在像上图这样的全双值格结构，并且形成**

不影响外界，但内部可发生交换的形式，那么题目一定不是唯一解的。

1-2-2 行列交换类型

1-2-2-1 原理

3	9	5	2 7 8	4 7 8	2 6 4	6 8	6 8	1
1	4 6	7	5	4 8	3	9	2 6 4	2 8
8	2 4 6	2 6	9	1 4	1 6	7	5	3
5	1 7		3 4	2		1 4	3 8	6
6	3	4	1	5	8	2	9	7
2	1 7		3 4	6		1 4	3 8	4 5
4	8	3	2 7	9	1 2 5	1 5 6	1 2 6 5	2 5
9	2 5	1	6	3 7	4	5 7	2 3 7	8
7	2 5 6	2 6	2 8	1 3 8	1 2 5	1 5	4	9

如图所示，如果 $r6c7(5)$ 假的话，那么观察 $r46c234678$ 这十二个单元格，横着看都是 134789 六数组，完全一样的。既然都是一模一样的六数组，很明显， $r4$ 的填数和 $r6$ 的填数就可以完全交换一下，也就是说， $r4c2$ 和 $r6c2$ 交换、 $r4c3$ 和 $r6c3$ 交换、 $r4c7$ 和 $r6c7$ 交换，诸如这样的方式。

这样把所有的六格的填数交换一下，上面的填数放到下面去，下面的填数放到上面去，这也完全对全盘没有丝毫的影响。所以，我们可以认定它是一种致命结构。

1-2-2-2 互补性

实际上，这一类型的致命结构是有互补视角的。我们回顾一下之前提到的一个问题：为什么结构上下对应的候选数会如此相似。比如上一题里，结构 $r4c2$ 和 $r6c2$ 的候选数情况完全一样， $r4c3$ 和 $r6c3$ 也一样，其它也都是（除了删数的单元格 $r6c7$ 这一组不一样以外）。

我们在之前的解答里，是这么说的：由于结构只是涉及行列的上下或左右对应位置的交换，所以它跟宫没有任何影响。而且实际上，在这些单元格的所处行列上，确实具有完全相同的数字作出了排除，所以候选数是一样的。

那么，我们参照这一点，换做互补的视角来看看。如图所示。

3	9	5	² 4 7 8	² 4 7 8	² 6 4 8	² 6 4 8	² 6 4 8	1
1	² 4 6	7	5	² 4 8	3	9	² 6 4 8	²
8	² 4 6	² 6 4	² 4 9	¹ 4 9	^{1 2} 6 9	7	5	3
5	¹ 7	² 8 9	³ 4	2	¹ 7 9	^{1 3} 4 8	^{1 3} 8	6
6	3	4	1	5	8	2	9	7
2	¹ 7	² 8 9	³ 4	6	¹ 7 9	^{1 3} 4 5 8	^{1 3} 8	⁴ 5
4	8	3	² 7 9	¹ 7 9	^{1 2} 5 7 9	¹ 5 6	^{1 2} 6 7	² 5
9	² 5	1	6	³ 7	4	³ 5	^{2 3} 7	8
7	² 5 6	² 6	^{2 3} 8	^{1 3} 8	^{1 2} 5	^{1 3} 5 6	4	9

我们把 r46 提取出来，把之前涉及的 12 个单元格抛开不看，只看剩余的位置，即确定值。此时我们可以发现，当 r6c9 = 5 的时候，r46 的所有确定值信息完全一样：都是 2、5、6。显然，既然出现了一样的填数信息了，那么 r46 剩下的单元格就一定是完全一样的数字，那么上下就必然能产生对应位置互换的方法了，这样就产生了两种填数方法，故形成了致命形式，所以 r6c9 <> 5。

所以，可以看到这个例子里，删数是互补视角上的 r6c9(5)，而原视角里删数则是 r6c7(1348)，即其它都删了，除了 5 没删。这就形成了鲜明的互补视角。所以，如果我们发现一个结构比较大的话，或者说一个疑似致命结构比较大的话，可以尝试使用互补视角来解决观察的问题。

那么，我们从这个视角里产生一个特殊的结论：**如果同一个并排三宫里的两行或者两列没有任何的确定值信息，题目一定是多解的。**所以，如果你看到这样的结构，那么题目一定是多解题（当然，其余位置如果产生显然的矛盾，这一种情况另说）。

接着，我们再来看一种更神奇的逻辑。

1-2-3 数字交换类型

1-2-3-1 反转唯一环 (Reverse Unique Loop)

	3		3		1 2 3	1 2 3	1 2 3	1		4	6
5		5		5	7		7		8 9		
7 8 9		8 9	7								
	6	4	1	8			9		2	3	2 3
7					7	6			5	5	
	3		3								
6			2	5		1 3	4		7		
8 9	8 9					6			8 9		8
			4	6		1 2 5			3	1 5	1 2
8 9	8 9				7		7				7
	5 6		5 6	3		1 2			2	1	
7					7		4	8	5	5 6	9
1	2		5 6	9		5 3		3	4	5 6	
			7			7		7		8	7 8
2 3		7			1 2 3	1 2 3			5	6	1 3
			8 9			8 9				8 9	4
	1 3				1 3	1 3			6	1	
4			8 9		7	7 8 9			8		2
											5
2 3	1 3		5 6		4	1 2 3	1 2 3		7	1 3	1 3
5 6						8 9				8 9	8

如图所示。此时我们换个思维。全盘只有这样涂色的五个单元格有确定值 8 和 9。如果说 $r6c9(8)$ 此时为真的话， $r2c46$ 、 $r5c69$ 、 $r6c49$ 就会构成关于 8 和 9 的确定值版本的唯一环结构。

那么我们现在要说明，确定值版本的唯一环结构在此时是致命的。由于是确定值，而确定值就可能包含提示数信息，所以显然不能直接交换。而如果我们尝试对于所有没有填数的单元格，又都含有候选数 8 和 9 的地方全部随便填入一种方式，只要不违背数独规则即可，然后对这些位置全部都互换一下。

我们假设此时盘面的解唯一的话，那另外一个盘也应当有唯一的解，而且是完全一样的解（除了这里 8 和 9 互换的地方不同外）。试想一下，一道唯一解的题会不会存在这样，有局部的数值进行互换之后，剩下却完全一致的盘面对应同一个解的情况？当然不存在。所以这便形成了致命形式。

我们将这种利用确定值形成唯一环（或唯一矩形）形式来反向证明出致命形式的模式，叫做**反转致命形式**（Reverse Deadly Pattern），它实际上包含两种情况，一种是前一个示例给出的拓展矩形的互补视角，称为**反转拓展矩形**（Reverse Extended Rectangle）；而另一种就是这一节讲到的**反转唯一矩形**（Reverse Unique Rectangle）或**反转唯一环**（Reverse Unique Loop）。

请注意，在观察确定值的时候，一定要找到全盘所有的这些数字，不要单纯认为其中只要一部分形成了唯一矩形或唯一环就算可以了，实际上这是不对的，因为其它没有被勾选出来的这些确定值可能给剩余的单元格一些提示信息，出数或删除数的结论都是可以存在的，所以我们一定要找全盘的所有这两种确定值。

那么我们接下来来看一些反转唯一矩形的例子。

1-2-3-2 一个更大的反转唯一环

2	9	1	6	8	5	3	7	4
4	7	8	3	9	1	6	5	2
6	5	3	4	7	2		1	1
8		7	9	5	3	1		6
1			7	2	6			
9			8	1	4	7		
7	8				9	2		
			2		7			

如图所示，如果 $r7c3(9)$ 为真，则涂色的 8 个单元格将会构成唯一环形状的结构，而全盘所有的 8 和 9 都已经被涂出来，构成此形式。

显然，这样就会导致剩余盘面的 8 和 9 的填数位置没有其它的可确定的提示信息，使得 8 和 9 显然会出现两种不同的填入方式，使得出现多种填法，违背唯一解的要求。所以 $r7c3 \neq 9$ 。

1-2-3-3 反转唯一矩形（Reverse Unique Rectangle）

1	7	1	2	4	5	3	1	6
6	3	2	1	8	9	4	5	7
1	5	4	3	7	6	1	2	8
3		7	5	6		1	2	8
1	2	1	2	4	9	1	2	3
5	1	2	1	9	3	4	6	7
7	1	2	3	6	5	1	2	4
1	2	6	9	4	1	2	8	7
1	2	4	8	5	7	9	3	1

全盘只有三个位置有确定值 1 和 2。如果 $r1c3(1)$ 为真，就会构成确定值版本的唯一矩形，全盘别无其他关于 1 和 2 的提示信息，故 1 和 2 的填数无法继续判断和确定。所以这样的结构必然是致命的。

1-2-3-4 直推反转唯一矩形 (Direct Inference Reverse UR)

1 4 8	4 8	2	1 4 7 8	9	5	1 4 7 8	6	3
5	4 8 9	6 4 8 9	3	1 4 7 8	1	2	1 4 7 8 9	8 9
3	7	1 4 8 9	2	1 4 8	6	1 4 8	5	
1 2 4 8	2 4 5 8 9	7	1 4 8	1 4 5 8	3	1 4 8	1 2 4 8 9	6
6	3	1 4 8 9	1 4 7 8	2	1 7 8 9	5	1 4 8 9	8 9
1 2 4 8	2 4 5 8 9	1 4 8 9	6	1 4 5 8	1 8 9	3	1 2 4 8 9	7
2 8	1	5	9	6 7	4	6 7	3	2 8
9	2 8	3	5	1 7 8 6	1 2 7 8	1 7 8 6	1 7 8	4
7	4 6 4 6	1 8	3	1 2 8	9	5	1 2 8	

如图所示，如果 $r1c12(4)$ 都为假后，我们将立即产生一个**直推反转唯一矩形 (Direct Inference Reverse Unique Rectangle)**，从 $r1c2(8)$ 开始，使得 $b7$ 里只有 $r7c1$ 可以填入 8，且 $r1c1$ 只能是 1，此时全盘所有填入确定值 1 和 8 就确定下来，而且构成了唯一矩形的形状。

而之前说过，这种结构是必然会使得剩余盘面的 1 和 8 的填数可以产生置换，所以产生多种填法，进而违背数独的唯一解的要求。所以，我原假设 $r1c12(4)$ 形成区块，即至少有一个是成立的，当作区块，删除 $b1$ 和 $r1$ 的其余位置的候选数 4。

1-2-3-5 位于大列里的反转唯一环基本使用示例

1 7 8	3 6	6	3 6	4 7 9	2	1 4 6	3 7 8	5	8 9
1 7 8	3	4	9	5	1	1	2 3 7 8	2 3	6
2	5 6 7 8	5 6	6 7 9	6	3	4	1	8 9	
4 7	6	3	2 4 7	6	1	8	2 4 6	5	2 4
4 5	2 5	8	3	9	4	2 6	7	1	
9	1	2 4 7	6 4 6	4 7	6	5	2 8	2 4 8	3
4 7 8	3 5 8	2 5	2 3 4 5	2 4	1 4 5	9	1 2 3 8	6	7
4 7	3 5 6	2 5 6	1	8	4 5 6 7	4 7	2 6	9	2 4
4 7 8	6	9	2 4 6	4 7	6	3	1 4 6	2 8	5

如图所示，首先发现全盘的 6 和 7 只有 $b369$ 里存在。当 $r2c7 = 7$ 时，6 和 7 的所有提示信息将构成反转唯一环的结构，使得其余任何一个地方都没有 6 和 7 的提示信息，

即使题目能够继续完成下去，但到最终，6 和 7 都无法继续往下，因为此时的 6 和 7 已经没有任何提示信息可以让它们往下完成了，这个时候，所有空位上可以放入 6 的地方也可以放入 7，使得产生多种不同的填数结果。而题目是唯一解的，肯定不会允许这样的情况出现，所以这种结构应当是予以避免的，故原本的假设错误，故 $r2c7 \neq 7$ 。

1-2-3-6 位于大列里的反转唯一环

有些时候，反转唯一环的使用不像是上面的一些示例一样，它可能仅仅在并排的三个宫里。

² 9	1	³ 9	4	7	8	6	5	² ³
² 7		³ 7	4	¹ 9	5	6	¹ 9	² ³
5	8	6	2	³ 9	¹ ³ 9	¹ 4	¹ 4	¹ 7
4		³ 9	7	8	1		5	2
6	2	³ 9	5	⁴ 9	³ 9	7	8	¹ 4
8	5	1	6	⁴ 9	² 9		3	⁷ 9
¹ 7		⁷ 9	2	¹ 7	8	4	3	6
3	6	8		² 7	5	¹ ² 7	¹ 7	4
¹ 7	4	5	3	6	¹ ² 7		9	8

如图所示，这个结构便是这样。请仔细观察左侧 $b124578$ 里的 8 和 9，可以发现到的是，8 出现了很多，但 9 一个都没有。我们这么去思考：由于 $b124578$ 并不存在任何 8 和 9 的提示信息，而其中的所有 8 都只是我们填入的数字，这并不足以说明它一定是题目的一部分。如果 $r2c7(1)$ 和 $r5c9(1)$ 同假，则两个单元格都只能填入 9，使得 $b369$ 里 8 和 9 的确定值构成了唯一环的类似形状。

这样的形状，我们应当知道的是，8 和 9 显然在这个结构所在的 $r259c789b369$ 里全部都存在 8 和 9 的信息，而其余位置的 8 和 9 就完全跟这里所给出的 8 和 9 无关。换言之，剩下要填入的 8 和 9 肯定只能放在 $b124578$ 里，而这些位置的 8 和 9 肯定跟这里的 8 和 9 的提示信息已经脱离了关系。由于 $b369$ 的 8 和 9 构成的形状，导致剩下的这些 8 和 9 都完全无法通过这些仅有的提示信息得到新的结论，使得 $b124578$ 此时应当放入的 8 和 9 的位置完全不能确定。即使我们随意在 $b124578$ 里找到了一种不违背数独规则的填数模式，把 8 和 9 都填上了，但由于刚才说到的，8 和 9 在 $b124578$ 里没有任何的提示数，我们就无法保证这些 8 和 9 一定就是对的，所以最终这些填好的 8 和 9 的位置就可以产生置换，即把 8 换为 9，而把 9 改为 8。这样的修改显然是不会产生矛盾的，于是便产生了两种填法，违背了数独的唯一解里要求的每一个单元格仅有一种填数的相关内容。所以，最初的假设，即 $r2c7(1)$ 和 $r5c9(1)$ 不应同假，所以它们构成了类似于致命结构里的区块类型的删数模式，即其中必须至少有一个 1 是对的，所以删除掉它们的交集，即 $r3c9(1)$ 。

仔细揣摩这个推理思路就可以发现，它和之前的反转唯一环不太一样，但又有相似的推导思路，不过更加灵活了。

1-2-3-7 利用不同真的反转唯一矩形

3			5 6	5 6	2		6	1	4	4
	5 6	1		5 6	3	4	7		5 6	5 6
	5 6	2	4		5 6	1	8		5 6	3
4	4		7	1	6	5	3	2	4	
	5 6	3		5 6	4	8	2		6	7
1	4		2	7	9	3		4 5 6	4 5 6	4 5 6
4	6		1	2	3		6	4 5 6	4 5 6	8
2	4		8		5	1	4	6	4	6
	6	5	3	8	7	4	2	1		

如图所示，我们尝试假设 $r7c2(9)$ 区块为假作为开头，则 $r6c2(6)$ 为真。显然，全盘只有这三处有 6 和 9，如果此时 $r4c2 = 9$ ，则构成反转唯一矩形的致命形式。所以 $r4c2(9)$ 为假。接着，后面的逻辑就不用多作解释了。

1-2-3-8 另一则利用弱关系的示例

1	7	8		2 3	2 3	9		2 3	4	5
	4		4 5		2 3	1		2 3	8	7
	6		5 6		2 3	7	4	8		2 3
	2		3	1		5		2 3	4	7
8	9	5	4		7	1	6		2 3	8
	4	6	4	3	7	9	1		2 3	5
	7		2	4		6		6	5	9
5	1	9		4		2 3	6	8		2 3
	3	8	6		9	1		2 3	5	4

如图所示。链的推导就不过多阐述了，我们来看 $r4c6(6)-r7c1(3)$ 的成立的原因。如果它们同真，必然会导致 $r4$ 上填入 2、3 的位置被挤入到 $r4c12$ 里，而此时 $r7c1(3)$ 为真， $r7c2$ 是 2，那么 $r47c12$ 就只有 2 和 3，构成类似于唯一矩形的环状结构，而全盘所有确定值 2 和 3 已经被我们确定下来，如果这么填数的话，2 和 3 将不再存在提示信息，导致 2 和 3 在余下的部分里可以互换形成两种填法，进而违背唯一解的要求。

所以规避这个内容，我们认定 $r4c6(6)-r7c1(3)$ 是成立的，所以引出了这一条不连

续环，得到了删数。

1-2-3-9 推导更麻烦的反转唯一矩形

1 4	1 8	3 8	9	5	4 7 8	2	6	1 7	4 7	3
2	1 5 8	3 8	5 8	3 7 8	1 4	6	4 7 8	3 7	1 5	9
1 4 5 6	7	3 5 6	1 4	3	4	9	2 4	8	2 4 5	3
5 7 8	9	2 5 7 8	4 7 8	1	4 5 7 8	3	6	4 7 8	2	
1 7 8	6 8	3 6	4	2	3 7 8 9	3 7 8	5	7 9	7 8	
5 7 8	6 8	2 5 6	3 7 8	6 9	4 5 7 8 9	3 7 8	2 4 5 6	2 7 9	1	
9	4	5 7 8	7 8	2	5 7 8	1	3	6		
3	2 5 6	2 5 6	6 9	5 7 8 9	1	2 7 8	4	2 5 7 8		
5 7 8	6 8	1	3	4 5 7 8	4 5 6 7 8	9	2 5 7	2 5 7 8		

如图所示，假设 $r1c8(1)$ 为假，则 ALS 区域 $\{r1c8, r2c7\}$ 里必然得到 $r2c7(4)$ 为真，所以 $r1c9(4)$ 为假。不过由于刚才假设 $r1c7(1)$ 为假，所以 $r1c8(7)$ 是为真的，因此 $r1c9$ 此时不会含有数字 7，那么 $r1c9$ 只能填入 3 了；接着，当 $r1c9(3)$ 为真时， $r1c2(3)$ 为假，而刚才假设 $r1c8(7)$ 为真的缘故，所以显然此时 $r1c2(8)$ 必须为假。否则全盘所有确定值 7 和 8 就固定下来，而构成唯一矩形的形状，导致 7 和 8 的剩余盘面将产生可交换的情况，导致非唯一填法，违背唯一解的要求。所以 $r1c2(8)$ 为假，故 $r1c2$ 此时只能使得 $r1c2(1)$ 为真。

所以 $r1c28$ 里必须有一个 1 是成立的，所以删除它们的交集，即 $r1c1(1)$ 。

实际上可以看出，这是一个环结构，不过麻烦的地方在于，它的内部嵌套了众多的分支，所以这里就不讨论其它地方的删数了（你可以尝试找找看，到底还有没有额外的删数）。

那么，我们根据这一个致命形式，可以得到一个结论：**如果全盘没有出现至少两种数字的任何提示信息，则题目一定是多解的。**

1-2-3-10 完全分离的两个反转唯一矩形？

下面我们来看一种比较特殊的情况，不过它稍微需要一点链。

6	1		7		4		
5	4					7	6
3	7	8		9	2	1	5
7					6	2	
2	8		7	3		5	
9			2				7
4	2	7	9		5		
1	9	5		7		4	2
8	6	3	2		7	9	

如图所示。我们先尝试认为链结构是对的，并认为处于 r56c26 的 3 和 8 会构成反转唯一矩形的致命形式，稍后我们再作出原因的说明。我们先来看链结构：假设 r6c2(3)为假，则按照链结构，顺次得到 r6c2(5)为真、r6c5(5)为假、r1c5(5)为真、r1c6(5)为假，并最终得到 r12c6(28)毛刺显性数对为真。

当链头 r6c2(3)为真时，我们认为反转唯一矩形成立，则可以删除 r6c6(8)；如果链尾 r12c6(28)为真时，显然也可以删除 r6c6(8)，所以 r6c6 <> 8。

下面我们来思考，这样的结构为什么依然是成立的。为什么我们之前说，“不允许在结构之外出现这些数字的任意提示信息”，但这个例子里却出现了，但依旧可行呢？这是因为，外部出现的 3 和 8，依旧完全构成了反转唯一矩形的形式，即构成了唯一矩形一般分属于两个宫的构型。这样的话，即使有这些 3 和 8 的提示信息，它对外部的其余 3 和 8 也是不会产生任何影响的。所以，这两个反转唯一矩形实际上是毫无关系，但都对外部不产生其余的影响，故这两个分离的反转唯一矩形此时都是完全成立的。

那么，既然成立了，那么推导逻辑和删数原理也就全部正确了，我们就不再作出说明，下面我们来看一则带强制思维的、这样分离的反转唯一矩形的例子。

1-2-3-11 分离的反转唯一矩形的另一则示例

下面来看最后一个比较难的例子。

4	3	4 5	4 5	2	1 2	7	6	1
1	6	5	5	4	9	2	8	3
4	4	2	6	1	3	1	4	5
2	1	3	6	9	5	4	4	7 8
6	9	4	5	1	2	2	1	7 8
8	5	7	1	4	7	9	3	6
5	4	2	4	2	6	4	1	3
4	4	6	1	7	4	8	5	4
4	3	1	5	4	3	6	4	2

如图所示，如果我们尝试假设 $r4c5 = 4$ ，则根据图上给出的链的顺序，可以分别得到 $r5c7 = 4$ 、 $r6c3 = 4$ 、 $r2c5 = 4$ 。而此时可以发现，当这些数字填入后，全盘所有的 4 和 9 分为了两个反转唯一矩形的构型出现在盘面上。刚才我们说到，这样的两个完全分离的反转唯一矩形互不影响，而也都不会影响到外部的其余 4 和 9，导致其余 4 和 9 的提示信息无法继续得到，故产生无法填完 4 和 9，即产生多个填数方法的致命形式。

所以，为了规避这个形式的出现， $r4c5 \neq 4$ ，这便是这个例子的结论。

1-2-4 对称翻转类型

1-2-4-1 宇宙法（Gurth's Symmetrical Placement）

1	3	7	6	5	8	1 2	1 2	4
1	1 2	1 2	2	3	4	1 2	7	6
4	2	2	1	2	7	5	2	3
2	1	6	3	4	5	6	4	8
8	1	6	2	1 2	1	2 3	2 3	5
5	7	4	8	4	6	2 3	4	1
6	1	5	8	4	2	1 3	1 3	7
3	4	1 2	7	6	1	1 2	1 2	2
7	1 2	1 2	5	8	3	4	6	2

如图所示，我们将所有 1 和 2 的确定值构成一组，3 和 6 一组，4 和 7 一组，5 和 8 一组，9 此时没有合适的构成一组。那么你就会发现，这个盘面有一个神奇的现象：拿 1 和 2 一组来说，所有 1 的中心对称的位置恰好都是 2，而反过来也是一样；3 和 6 也是这样，所有 3 的中心对称的单元格上一定是 6，其它的也都是这样。

那么我们就可以思考一个问题了。既然确定值的信息是完全中心对称的，那么我们随意在一处地方找到一种技巧，那么与之中心对称的位置就一定会找到与之对应的同一种技巧。比如我们在 r3c4 上发现了一个环，那么 r67c67 就一定存在一个环。因为提示信息是中心对称的，所以如果有 1 的技巧，就在对应中心对称的地方找得到 2 的技巧，因为 1 和 2 是中心对称的，其它的技巧也都同理。

而 r5c5 是这个盘面的对称中心。如果题目是唯一解的，r5c5 就不应当放“可以构成一组的两种数字的其中一种”。比如说，r5c5 如果是 1 的话，那与之中心对称的那一个单元格就必然是 2。可是 r5c5 已经是对称中心了，就意味着它的中心对称的位置就是它自己。所以显然要使得 $r5c5 = 1$ 就必须 $r5c5 = 2$ ，这样是根本不可能的，所以 1 和 2 都不可能放进去。同理，3 和 6 是一组显然不能同时填入；4 和 7 一组不能，5 和 8 一组也不能。所以 r5c5 只能填入唯一的一个没有配对的数字 9 了，所以 $r5c5 = 9$ 。

这种技巧利用了盘面的高度对称性来进行解题，所以是一种非常强大的技巧，但这种技巧也依赖于高度对称的盘面，所以不对称的盘面显然就无法使用这个技巧了。这种技巧叫做**宇宙法**（**Gurth's Symmetrical Placement**），中文名称是由中国数独/谜题国家队选手邱言哲命名的，因为宇宙一词的英文为 **Universe**，它的词根就是类似于全盘、全局的意思。

我们再来看一种对称类型：轴对称。

1-2-4-2 轴对称类型的宇宙法

1	4		6	3	5	4	3		2
	7	8	9	7	8	9			
4	5	6	2	5	6	1	6	9	3
7			7	8		8		4	5
4	5	6	4	5	3	7	4	6	1
9			9				8		5
5	3	1	5	4	1	3	1	2	6
9			9	5	8	9		8	7
8	1	2	1	6	1	2	6	5	3
7	9	7	9		6	9	7	8	4
5	3	6	2	3	2	3	2	8	9
7		7		5	4		7		1
4	6	3	7	6	2	8	5	1	6
7			7	9				4	
4	5	4	5	1	9	3	6	4	2
7		7	8					7	5
2	5		5	6	4	7	1	6	3
	8	9		8				9	

全盘所有确定值下，4 和 7 一组、5 和 8 一组、6 和 9 一组，剩下的 1、2、3 各自单独一组。

注意，这里叙述的是单独一组，不是一起的。

关于捺对角线对称的话，r4c4、r5c5、r6c6 下就只能为 1、2、3，不能是其他数字

(这三格恰好在对称轴上)。否则的话，4 和 7 一组，比如 **r4c4** 是 7 的话，为了满足对称性，**r4c4** 还应是 4 才可以，这样是不符合数独规则的，同样的，6 和 9 也不行，5 和 8 也不行，就只能是 1、2、3 了。

虽然它们各自一组，但我们依然无法确定到底 **r4c4** 是 1，是 2，还是 3。所以我们只能大致知道，这三格一定都只有候选数 1、2、3。不过呢，由于三格恰好同宫（**b5**），所以这三格构成三数组，可以确定删除 **b5** 内的其余候选数 1、2、3 以及这三格内的所有其余候选数。于是，盘面瓦解。

至此，我们就讲完了基本的致命结构基本变换类型，这些变换类型既相似又有各自明显独特的地方。不过，我们之前提到的内容都还算比较容易。接下来我们来看的几个技巧就比较困难了，虽然都是数组置换类型的致命结构。

1-3 探长致命结构 (Borescoper's Deadly Pattern)

1-3-1 结构的形成

现在我们来看一种新的致命结构，我们先来看下如下的三个例子。

						1 2	1 2	
					1 2		1 2	
					1 2	1 2		

如图所示，我们可以看到，这就是一个普通的 UL 结构，只涉及 6 个单元格。我们把这个例子进行推广。我们尝试把结构改写一下，让这个结构涉及三个数，变为下面左图这样，而下面右侧的这个结构就是它的更高规格的形式。

						1 2 3	1 2 3				1 2 3	1 2 3	
					1 2 3	1 2 3	1 2 3				4	4	
					1 2 3	1 2 3				1 2 3	1 2 3	1 2 3	
										4	4	4	
										1 2 3	1 2 3	1 2 3	
										4	4	4	

不过看起来这个结构似乎并不太像是一个致命结构，现在我们来证明之。

1-3-2 证明

我们拿出结构，针对于 **b68** 的四个结构涉及的单元格作为假设条件。由于两个单元格显然只能填入两种数字，但结构本身是可以出现三种数的（稍后将给出证明），所以我们不妨就按照所有的情况一一进行枚举，看看是否全部情况都可以导致致命形式的出现，或者根本不存在这种形式。

首先，结构涉及三种数字，那么 **b68** 两侧放置的数对就只可能有两种情况：两侧是一样的数对和两侧有一个数不一样（比如 1、2 和 1、2，以及 1、2 和 1、3），显然不会有其它情况了。

先考虑第一种情况。

						1 2	1 2	
					1 2	1 2 3	1 2 3	
					1 2	1 2 3		

如右图所示，这是第一种情况（当然，2、3 和 2、3 以及 1、3 和 1、3 的情况和这里假设的情况是一样的，所以其它两种就不用再去证明了）。那么，可以从图里看到，**b9** 是有一个三数组的，所以我们针对于 1 和 2 而言来假设分布情况，那么就只有三种情况。

- 1 和 2 横着放在 r7c78 上；
- 1 和 2 竖着放在 r78c7 上；
- 1 和 2 斜着放在 r7c8 和 r8c7 上。

显然，第一种情况将会直接在 r67c78 上产生关于 1 和 2 的 UR 的致命形式，所以这种情况将被我们排除掉；第二种情况也是如此，在 r78c67 上产生关于 1 和 2 的 UR 的致命形式，所以只可能是第 3 种填法，即把 1 和 2 “斜着”放进去。但是斜着放也产生了致命形式，是 UL 的致命形式，所以三种情况全部出现致命形式，故第一种情况是可以产生致命形式的。

可能你在这里会问我，为什么在证明的时候不能使用 UR。比如这里可以看到结构里产生了 r67c78 以及 r78c67 的 UR 的区块类型，则可以删除两处的 3，一样形成了致命形式，可是我并没有在前文里这么去证明它，这是为什么呢？因为我们是在证明致命形式是否出现，所以我们在结构里不要使用任何致命结构的技巧，否则结构因为可能的非唯一填数情况而使用致命形式使得结构本身变得不稳定（多种填法、唯一填法或没有合适的填法）。

接下来我们来看第二种情况的证明。

						1 2	1 2	
					1 3	1 2 3	1 2 3	
					1 3	1 2 3		

如图所示，这是第二种类型，我们依旧按照 b9 里按横着、竖着和斜着三种情况来放置 1 和 2（当然这里的数字不同，你也可以假设为放置 1 和 3，因为情况等价，所以这里只需要一种情况即可）。首先横着是不允许的，因为一定会出现 UR 的致命形式；而竖着又会让 c7 出现三个 1 和 2 的双值格产生无效填法，所以这种也是不可以的；那么 1 和 2 只能斜着放进去。

当 1 和 2 斜着放进去的时候，显然可以得到 r7c7 = 3 的结果，于是 r78c6 都可以出数，进而产生了连锁反应，导致这 7 个单元格全部可以出数，但只有唯一的填法，如下图所示。

						2	1	
					1	3	2	
					3	1		

但遗憾的是，**r78c67** 产生了关于 **1** 和 **3** 的致命形式，所以这种情况依然是会形成致命形式的，所以，第二种情况也会形成致命形式。

所以，两种情况都会产生致命形式，所以这个结构本身就是一个致命结构。

可以发现到的是，结构在证明过程之中，位于 **b9** 的三个单元格只要是和 **b68** 两侧的数对同行列的话，都是允许的，所以这个结构实际上有四种不同的情况，如下面四个图所示。

						1 2 3	1 2 3					1 2 3	1 2 3	
						1 2 3	1 2 3	1 2 3				1 2 3	1 2 3	1 2 3
						1 2 3	1 2 3					1 2 3		1 2 3

							1 2 3	1 2 3							1 2 3	1 2 3	
					1 2 3			1 2 3					1 2 3	1 2 3			
					1 2 3	1 2 3	1 2 3						1 2 3	1 2 3	1 2 3		

其中的任意一种情况都是一个合格的致命结构。

接下来我们尝试证明四个数的形式是一个致命结构。由于涉及四个数，所以分情况就有三个了：两侧是一样的数（例如 1、2 和 1、2）、两侧有一个数不同（例如 1、2 和 2、3）和两侧涉及的两个数都不同（例如 1、2 和 3、4）。

我们一个一个来看，首先是第一种情况。

										1 2	1 2						
						1 2	1 2 3	1 2 3	1 2 3	4	4						
						1 2	1 2 3	1 2 3	1 2 3	4	4						

如图所示，我们依然按照前文的说法分成三种情况，b9 的 1 和 2 横着放、竖着放和斜着放。横着放和竖着放显然都会出现错误，一个会出现致命形式，而另外一个则会使得列上出现三个 1 和 2 的双值格。所以 1 和 2 显然只能斜着放，但是斜着放就变为了之前三个数的情况，形成了 UL 的致命形式，所以这个情况必然产生致命形式。

接着看第二种情况。

						1 2	1 2	
					2 3	1 2 3	1 2 3	
					4	4	4	
					2 3	1 2 3	1 2 3	
					4	4	4	

如图所示,这是第二种情况,我们此时可以这么想这个问题。我们把 4 随意放在 **r7c78** 里的任意一个单元格里,这便使得剩下的三个单元格构成了 1、2、3 的三数组结构,而配合两侧的 1、2 数对和 2、3 数对,结构形成了之前的致命形式(这里就不证明了,前文已经证明了这个结构),所以这个会产生致命形式的。

最后我们再来看一下第三种情况,即两侧的数字没有相同的,我们拿 1 和 2 以及 3 和 4 来举例说明。

						1 2	1 2	
					3	1 2 3	1 2 3	
				4	4	4	4	
					3	1 2 3	1 2 3	
				4	4	4	4	

如图所示,这是第三种情况,我们照样按照之前的分析方式,对 **b9** 里的 1 和 2 作出填数的放置情况的分析:横着放出现 UR 致命形式;竖着放则会在 **c7** 或 **c8** 里出现三个 1 和 2 的双值格,出现矛盾;所以此时依然只能斜着放。不过斜着有两种情况,一种是 **r7c7** 和 **r8c8** 是 1 和 2,另外一种则是 **r7c8** 和 **r8c7** 是 1 和 2。我们按其中一种情况来分析即可。假定, **r7c7** 和 **r8c8** 是 1 和 2 的话,则有如下图所示的情况。

						1 2	1 2	
					4	3	1 2	4
					4	3	4	3
					4	3	1 2	

似乎看起来并不像是一个会产生致命形式的东西，我们随意填入一种情况，而这种情况看似不会造成交换，但实际上 **b9** 内发生了轮换，导致产生了两种填法，所以我们任意假设出其中的这两种填法，如下图所示。

					1	2		
				3	2	4		
				4	3	1		

可以看到，**b9** 里的 1、2、3、4 的填法按宫内进行了“顺时针旋转”，即都向它逆时针上相邻那一个单元格移动了一下，而与此同时，两侧的数对也因为刚才的交换而发生了变换。不过，交换是合法的，因为我们找到了一种填数方式，使得这种填数方式和原本的填数方式完全一样：结构所处的所有区域下都产生了一致的数组结构，并未发生实质性的、有影响的修改。比如我们这种置换形式，**c7** 上原本是 1、2、3，而轮换了之后，**c7** 是 2、3、1，依然还是这些数，所以 **c7** 产生的两种填法对这个区域而言并未造成任何的影响。同理其余的区域也是一样的道理。

那么，既然结构涉及的任意区域都不会产生改变，我们就可以认为第二种情况是第一种情况的置换后的结果。如果题目是唯一解的，则显然对于这 8 个单元格而言就产生了两种填法。而且严谨的是，这里应当说是**至少两种**填法，因为这个结构还可以逆时针轮换 **b9** 里的数字。所以形成了致命形式。

故第三种情况也会产生致命形式。所以三种情况全部矛盾，故 4 个数字的结构依旧是一个致命结构。

证明看起来很难受，但是实际上结构很简单，我们这里把结构罗列出来。

							1 2 3	1 2 3						1 2 3	1 2 3		
														4	4		
						1 2 3	1 2 3	1 2 3					1 2 3	1 2 3	1 2 3		
													4	4	4		
						1 2 3	1 2 3						1 2 3	1 2 3	1 2 3		
													4	4	4		

这两种形式的致命结构称为**探长致命结构**（**Borescoper's Deadly Pattern**），由中国数独玩家探长（英文 **Borescoper**，为昵称）发现，并冠名该技巧。

另外需要啰嗦一下的是，这个技巧的结构内部的候选数是可以消失一些的，而消失的这些候选数并不影响结构形成致命形式。这也是显然的，因为结构本身的完整版是形成致命形式的，所以少掉一些情况，只是少一些填数方案，但我们说过，形成致命形式是客观的，只要能发现一种填法使得形成致命形式，就必然存在与之互换的另外一种形式，所以它并不会影响到什么。

当然了，消失一些候选数使得结构直接被破解，出现排除或者唯一余数之类的技巧的话，这种情况就另说了。

下面我们来讲述一下，关于这两种致命结构的基本使用。

1-3-3 三个数的探长致命结构 (3-Digit Borescoper's Deadly Pattern)

1-3-3-1 标准类型 (3-Digit Borescoper's DP Type 1)

2	4 6 9	1	6 4 5 6 9	5	3	7	8
8	7 6 4 5 9		2 6 7	3	4 5 9	1	4 5 9
3	4 7 9 4 5 9		4 7 8 9	1	4 5 9	6	2
6	4 8 9	3	1 2 5 8 8	7	4 5 8 9		4 5 9
1	5	2	6 8	9	4	7	3 6
7	4 8 9 4		3 2 5 6 8		1	2 4 5 6 8 9	
4	3	8	2 7 9 7	6	2 9	5	1
9	1	6	5 3 8		2 8	4	7
5	2	7	4 1 8 9		6	3 8 9	3 9

如图所示，我们寻找到一个三数的探长致命结构，为了规避致命形式的出现，所以 **r4c7** <> 459。

1-3-3-2 区块类型 (3-Digit Borescoper's DP Type 2)

5 3 8	2	5 3 6	1	7	4 3 6 8	5 3 8	4 3 8	9
5 3 8	7	4	3 8	9	2	5 3 8	6	1
9	6 8	1	3 5 6 8	3 6 4 5 6 8		2 4 3 8	7	
7	9	2	4	5	3 6	3 6	1	8
5 3 8	1	5 3 6	9	2 3 6 8	3 6	4	7	2 3 6
4	6 8	3 6 8	2 3 6 8	1	7	9	5	2 3 6
2	5	9	7	3 6	1	3 6 8	3 8	4
1	3	8	5 6	4	9	7	2	5 6
6	4	7	2 3 5 8	2 3 8	3 5 8	1	9	3 5

如图所示。如果此时 **r15c3(6)** 均不存在的话，将使得结构只有 3、5、6，出现探长致命结构的致命形式。所以为了规避致命形式的出现，**r15c3(6)** 至少有一个是对的，所以构成类似于区块的形式，故 **r6c3** <> 6。

1-3-3-3 拓展类型 (3-Digit Borescoper's DP Advanced Type)

1	1 2	6	5	2	3	2	9	4
8	7			8		7		
5	2	5	1 2	6	9	4	2 3	1 2 3
8	7	7	8			7	6	6
4	3	9	1 2	6	7	1	5	8
			6			6		1 2
2	4 5		3	1	8	5 6	4	6
		8	7	8	7	8	5	6
7	5	1	2	6	4	5 6	3	5 6
	9					9		8
6	4 5		3	1 2	8	2	1	2
	9	8	8	8	8	5	4	5
3	6	2	7	8	1	7	8	9
1	1		4	9	3	5 6	8	2
5	7						7	6
9	8	5	4	5 6	2	1	3	3
	7					7	6	6

如图所示,我们发现,3和7都仅出现在 r46c46 里。如果 b5 里所有 8 也只放在这四个单元格里的话,那必然 3、7、8 就只可能出现在这其中的三个单元格里,但是通过之前的说明里我们知道,不管怎么安放 3、7、8,配合 r46c3 和 r7c46 的话,都必然形成关于 3、7、8 的致命形式,所以我们不得不删除这些 8。

可是,为什么要全都删除呢?你可以反向思考这一点。b5 里除了 r46c46 四个单元格外,还有五个单元格,这些单元格里必须得有一个 8 的出现,所以很自然就可以去掉 r46c46 的 8 了,因为 b5 里一定会有 8 在这五个单元格里。

当然,这个例子还有一个删数,是 r1c5(8),是因为我们刚才说到的那五个单元格里,只有其中两个单元格包含候选数 8,而且它们恰好形成区块,所以 b5 和 c5 上其余位置的 8 也可以删除。

接下来我们来看一些嵌套该结构的链的示例。

1-3-3-4 毛刺探长致命结构

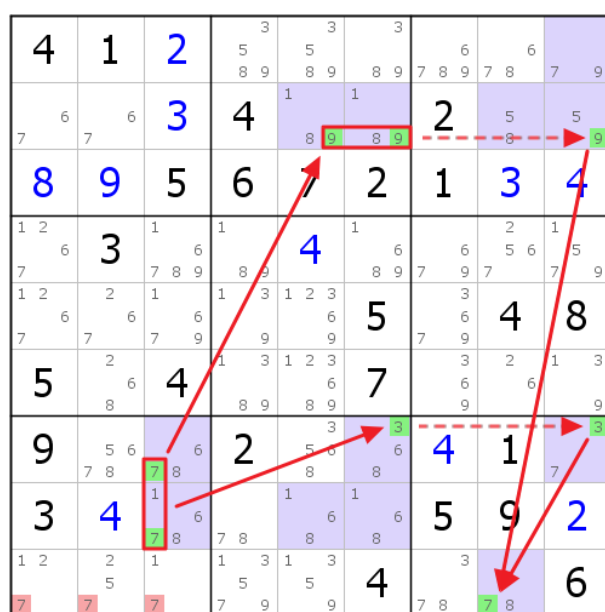
8	3	4	3	2	2	4	5	6	1	4	2
7	6	7		7	6	4	5	9	7	5	
2	5	6	1	2	6	3	4	5	6	4	9
7		7	8	7	8	7	8	7	8	7	8
2	5	9	1	4	5	4	5	6	2	3	5
		7	8	7	8	7	8	7	8	7	8
1	3	1	3	1	3	1	3	1	3	1	3
5	7	8	9	7	8	9	7	8	9	7	8
9	5	8	6	5	8	6	5	8	6	5	8
6	4	7	3	3	1	2	5	1	3	7	8
4	2	9	5	1	7	8	3	6	1	7	8
1	3	1	3	6	7	8	9	7	8	9	7
7	1	3	5	2	3	1	2	1	3	2	5
8				8				8			

来看一则用法比较灵活的毛刺。我们将 $r6c5(9)$ 视作毛刺。当它为假的时候，我们可以引出一条区块不连续环，删数此时是 $r5c6(5)$ 。不过毛刺为真的时候，我们一定有 $r6c5 = 9 \Rightarrow r5c6 = 1$ 的逻辑。考虑反证法，假设 $r6c5(9)$ 为真但 $r5c6 \neq 1$ 的话， $b5$ 里将产生 1、5 的隐性数对，而与此同时，由于 $r6c5(9)$ 的占位，导致 3、7、8 全部挤入到 $r56c4$ 里，导致必然有一个数填不进去，出现矛盾。所以这个逻辑是对的。所以，我们就相当于得到了：

- $r6c5(9)$ 为假 $\Rightarrow r5c6(1)$ 为真；
- $r6c5(9)$ 为真 $\Rightarrow r5c6(1)$ 为真。

所以 $r5c6(1)$ 不论任何情况都为真，所以它为真。这一点很像是矛盾强制链里的思维，不过矛盾强制链是证否。

1-3-3-5 嵌套探长致命结构的动态链



如图所示。假设 $r7c3$ 里不含有 7，则 $\{r2c56, r7c36, r8c356\}$ 将产生一个疑似探长致命结构的形式。我们此时分两种情况讨论。

如果此时 $r2c56(9)$ 为真，则 $r2c9(9)$ 为假，此时观察 $\{r2c89, r9c8\}$ 三个单元格，如果 $r9c8(7)$ 也不存在的话，那所有数字都不跨区，三个单元格只含有两种不同的数字，将无法填满，产生矛盾，故 $r9c8(7)$ 必然为真才行；

如果此时 $r7c6(3)$ 为真，则 $r7c9(3)$ 为假，于是观察 $\{r1c9, r2c89, r7c9, r9c8\}$ 五个单元格，如果此时 $r9c8(7)$ 为假的话，此时里面不含有重复的数字，而且每一个数字都不再跨区，五个单元格填不满，所以会矛盾，所以 $r9c8(7)$ 为真。

显然，上述两种情况里至少有一个是为真的，否则都为假，而 $r7c3(7)$ 为假，此时 $\{r2c56, r7c36, r8c356\}$ 七个单元格将只有 1、6、8，形成探长致命结构的致命形式。所以必须保证至少有一个情况成立，但最终它们的结果都是 $r9c8(7)$ 为真。所以这个链的整体思路就是设 $r7c3(7)$ 为假时一定有 $r9c8(7)$ 为真，故删除头尾的交集，即 $r9c123(7)$ 。

不过，这个题目 UR 比较多，实际上图里还有一个 UR 的毛刺删数，和一个 UR 的数组

类型，你可以找找看，虽然题目里的 UR 跟这里的探长致命结构没有什么特别的关系……

1-3-4 四个数的探长致命结构（4-Digit Borescoper's Deadly Pattern）

接下来我们来看一些四数的探长致命结构。

1-3-4-1 标准类型（4-Digit Borescoper's Deadly Pattern Type 1）

9	1 2	4	5 8	7 8	5 7 8	1	2	3
6	1 2	7	3	9	4	1	2	8
8	5	3	1	6	2	9	7	4
2	6 9	5	4 6 9	1	3	8	4 6	7
7	4	6 9	6 8 9	5	6 8 9	3	1	2
1	3	8	2	4 7	6 7	5 6	4 5 6	9
3	6 8	1 2	7	4 8	5 6 8	2	9	1 5 6
4	6 8 9	6 9	5 6 8	2	1	7	3	5 6
5	7	1 2	4 6 9	3	6 4 9	2	8	1 6

如图所示，当 $r6c8(4)$ 为假时，将使得 $\{r12c278, r6c78\}$ 产生四数探长致命结构的致命形式，所以 $r6c8 = 4$ ，删掉其余的候选数。

1-3-4-2 区块类型（4-Digit Borescoper's Deadly Pattern Type 2）

3	2	6	9	1	5	3	4	7
1 4 5	1 4	1 4 5 9	8	3	7	2	6 9	5 6
7	3	5 9	6	2	4	5 8	3 9	1
1 3 4 5	6	1 3 4 5	2 4 7	5 8	1 3 8	9	2 3 5 7	2 3 4 5
4 5 9 7 9	8	2 4 7	6	3 9	1	7	2 3 5 7	2 3 4 5
1 3 4 5 9 7 9	2	4 7	5 9	1 3 9	4 5 6	8	4 5 6	3
2	1 4 8	1 4	3	7	6 8	4 5 6	1 5 6	9
1 3 6 9	7	5	4	2	3 6	1 3 6	8	
6 8	5	4 3	1	8 9	6 8 9	7	2 3 4	2 3

如图所示，如果让 $r45c89(5)$ 全部消失，则 $\{r45c389, r9c89\}$ 形成关于 2、3、4、7 的致命形式，所以 $r45c89(5)$ 至少有一个是对的，那么删除 $b6$ 其余位置的 5。

1-3-4-3 其它类型

1	3	3	1	6	4	4	5	6	9	4	5	6	5	6	2
		8	8	7	8		8			7	8	7	8		
2	4	7	3		5	6	1		5	6	5	6	5	6	9
5	9		6	4	4	6	2	1	3	4	6		7	8	
6	1	3	9	7	4			2		2					5
9		2		5	4		6	3	1	4			7	8	
4			5	1	2	3		4	6	9	4	6	7	8	
	3	2	3	2	4		1	4		9	2		5	6	3
8		2		9	6	3	5			4					1
1	3	3	1	6	2	9	7		5	6	5	6			3
4		5	6	4	6					8		8			8

如图所示。当 r1c57(4)去掉后，结构变为一个只带三个数，但在 b3 占了四个单元格的探长致命结构。可以发现，这种结构实际上就类似于之前讲到的三数探长致命结构的进阶类型：b3 里任取 r12c78 的其中三个单元格，结构就一定出现致命形式。所以这个题也一样。我们任取其中三个单元格，都会保证里面出现一个数字 7，而这个数字 7 一定是对的，否则结构必然出现关于 5、6、8 的致命形式，所以可以删除的是这个 7 区块对应的 r1 和 b3 的其余位置的 7；而显然，4 在 r1 的其余位置也可以删除，否则结构只有 5、6、7、8，必然形成致命形式。

1-4 淑芬致命结构 (Qiu's Deadly Pattern)

接下来讲一个技巧名比较接地气的致命结构：**淑芬致命结构 (Qiu's Deadly Pattern)**。这个技巧依然由之前发现宇宙的邱言哲整理和命名。

1-4-1 区分度理论 (Distinction Theory)

在正式讲解技巧之前，我们先来说一个术语：**区分度 (Distinction)**。区分度是用于致命结构的论证的，我们规定，一个致命结构一旦形成致命形式，那么这个结构的区分度此时是为 0 的；反过来说，一旦发现致命结构的区分度为 0，则结构一定是形成了致命形式。

不过这一点在拓展矩形里用得比较多，接下来的内容将以区分度来介绍它，因为它在一定程度上依赖于拓展矩形的结构。

		1	2					
		3	1					

如图所示，这样的结构区分度为 1，因为 r7 和 r8 的填数总会有一处是不同的。因为我们给定了确定值里，r7 和 r8 里都有 1，所以上下两行都不会再填入 1，这是它们的共性；而 2 和 3 不同，2 在 r7 里有确定值，但 r8 里没有，这意味着剩下的空格里，r7 可以放下 3，而 r8 里可以放下 2，它们是一组不同的数字，而其它的数（4 到 9 则都一定 r78 都有），所以不同的数只有 2 和 3 一组，因此我们称此时的区分度为 1。

寻找区分度就是看结构上下两行（或者为左右两列）上会有多少组不同的数字。

		1	2			4		
		3	1			5		

如图所示，这个是区分度为 2 的结构，因为 1 的确定值在上下两行都出现，所以空格不会出现 1 了；而其余的数字，6、7、8、9 在 r78 两行里都会“成对”出现，所以实际上只有 2、3、4、5 要特殊考虑。而上下两行的确定值可以构成两组这样的情况，比如 2 和 3 一组，4 和 5 一组（当然 2 和 5，3 和 4 也行）。所以我们就称这样的结构的区分度为 2。

或者换句话说，区分度看的是上下两行（或左右两列）的不同确定值的种类数，再除以 2。但需要注意的是，**区分度为 0 要能产生致命形式的话，那么确定值的分布就必须得上下**

对应，或左右对应。比如上述示例里，**r78** 上的所有确定值都是上下对应的：上面有一个确定值，那么它的下方必然也有一个确定值；反之亦然。如果结构的确定值是“错开分布”的，那么我们就无法断言其区分度是多少，因为此时还得继续看，上下对应的另外那一个空格的候选数情况到底如何。比如下面这个例子。

		1	2					
		4	3	1				

如图所示，这个结构并非上下对应起来都有确定值，但是我们可以发现，**r7c3** 和 **r8c4** 都是 1，那么 **r78** 的剩余空格里一定不会有 1，而 5、6、7、8、9 都是共同拥有的，所以只需要分析 2、3、4。

而实际上，在最终填入后我们就可以发现到的是，**r8c3** 只能是 3 或 4 的其中一个，而它们都不是 2。如果是 2 的话，那么上下两行的确定值都能“一一配对”，所以区分度直接变为了 0，形成了致命形式。

这一点你也可以从候选数角度来理解：剩余的单元格都只能填入 3 到 9，所以 **r7** 的一种填法可以上下对应交换放到 **r8** 里；而 **r8** 里的也可以放回到 **r7** 里面，导致出现两种填法。

那么，显然 **r8c3** 只能是 3 或 4，但都不是 2，所以区分度为 1，即有一组填数是不同的。

在了解了这一点的内容后，我们来看这个致命结构的长相。

1-4-2 构型

实际上，这个构型非常的可怕，因为在致命结构的内容里，这个技巧的结构算上非常大的了。

1	2	3	1	2	3				
4			4						
1	2	3	1	2	3				
4			4						
1	2	3	1	2	3				
4			4						

如图所示，它的结构是这样的，它需要满足如下的条件（有一个条件在图上不好标注出来，只能用文本叙述）：

- 结构一定占据两行（或两列，暂且称为 L1 和 L2）和两个额外的单元格（暂且称为 C1 和 C2）；
- C1 和 C2 必须同宫且同行（或者同宫且同列，具体情况看的是结构占据的是 L1 和 L2 的分布是按行分布还是按列分布），但不能在 L1 和 L2 内；
- 和 C1 和 C2 对应应在 L1 和 L2 里的四个单元格（这个“对应”指的是结构涉及的单元格是和 L1 和 L2 呈平行状态，然后上下对应的四个单元格，即这里的 C1 和 C2 是 r5c12，则上下对应起来的位置是 r78c12 这四个单元格）必须只能包含 C1 和 C2 涉及的这四种数字；
- L1 和 L2 的区分度为 1，即有一对数字不相同。

实际上这个结构看起来特别复杂，甚至动用了符号来叙述，可见这个结构的恐怖。那么接下来我们来看一下这个结构为什么是一个致命结构。

1-4-3 证明

下面我们来给出这个技巧的证明。

由于规定两行（或者两列）的区分度为 1，而涉及的数字却没有任何的限定，所以我们不得不分三种情况讨论。如上图所示，假定 r5c12 里填入的是 a 和 b 的话，那么有如下三种情况：

- r78 的确定值含有 a 和 b（即都有）；
- r78 的确定值里只包含 a 和 b 的其中一个；
- r78 的确定值里不含有 a 和 b 的任意一个（全是其它数字）。

下面我们就按照这三种情况一一进行讨论。

为了方便理解，我们把确定值固定为只有两个数，并且上下对应。那么第一种情况就对应这样的结构：

1 2 3 4	1 2 3 4							
2 3 4	2 3 4							1
1 3 4	1 3 4							2

r7 和 r8 按照数独规则都必须填入 1 到 9 各一个，而已经出现的 1 或 2，导致剩下 8 个单元格只能填入剩余 8 种数字。就 r7 而言，显然只能填的是 2 到 9，而其中 r7c12 又填入了 2、3、4 的其二，所以 r7c345678 里只能是 5、6、7、8、9 和一个 2、3、4 的剩余的那个数；与此同时，r8 上也是一样的道理：因为出现了 2，而且 r8c12 里必须是 1、3、4 的其二，所以 r8c345678 里只能是 5、6、7、8、9 和一个 1、3、4 的剩余的那个数。

对比两行的最终填数结果就可以发现，5、6、7、8、9 是它们一样的数字，而不一样的只有 1、2、3、4 的剩余那个数不同。我们尝试随意填入一种情况来解释后面的逻辑：

1 2 3 4	1 2 3 4							
2 3 4	2 3 4	5	6	7	8	9	3	1
1 3 4	1 3 4	8	5	9	4	6	7	2

如图所示，这是其中一种情况。我们假设 r7c12 里是 2 和 4 的话，那很显然 r7c345678 里多出来的 1 到 4 的那个数就只能是 3 了；同理，r8 假设 r8c12 是 1 和 3 的话，那

r8c345678 里多出来的就只能是 4 了。

这也就是区分度为 1 的基本逻辑。似乎看起来好像区分度为 2，因为 r78c345678 里有一组不同的数，而且 r78c9 又是一组不同的数。但实际上并不是这样。这两行此时区分度是“抵消”为 0 了，即形成了致命形式。为什么呢？我们尝试把这两行的非确定值的剩余单元格执行上下交换。

2	1									1	2						
4	2	5	6	7	8	9	3	1	2	3	8	5	9	4	6	7	1
1	3	8	5	9	4	6	7	2	4	1	5	6	7	8	9	3	2

如左图所示，我们把之前 r5c12 和 r78c12 的填数也确定下来看看到底能否交换。不过很显然的是，我们把 r78c345678 上下的对应位置执行交换，而剩余的 r78c12 进行顺时针轮换，再带上 r5c12 配合下面轮换的操作进行置换，就会发现，此时变为右图这样。

显然，两种填法都是可以的，而结构涉及的所有区域：r578c123456789b4789 的所有填数全部都没有发生变化。比如 c2 上，原本的填法是 1、2、3，而交换后是 2、3、1，数字依然只有 1、2、3 这三种数字，并没发生变化；同理，其余的所有区域也恰好全部满足要求。所以，这便形成了致命形式。故第一种情况是致命的。

实际上，我们完全不必去交换 r78c345678 的数字，我们完全可以只动 r578c12 的填数，只要保证 c12 的填数要和交换前的填数是一样的就可以了。然后就会发现，这种构造形式更容易得到一种合适的填法。

接下来我们来看第二种情况。

1	2	3	1	2	3			
4			4					
2	3		2	3				1
4			4					5
1	2	3	1	2	3			
4			4					

如图所示，显然数字 5 是 1、2、3、4 所涉及不到的数字。我们用来作为第二个情况的确定值的数值。

对于 $r7$ 而言，因为 $r7c12$ 是 2、3、4 其二，而 $r7c9$ 是 1，所以 $r7c345678$ 里只能放下的是 6、7、8、9 和 2、3、4 的剩余那个数和一个 5；同理，对于 $r8$ 而言， $r8c12$ 是 1、2、3、4 的其二，而 $r8c9$ 是 5，所以 $r8c345678$ 里只能是 6、7、8、9 和 1、2、3、4 的其二（六个单元格，显然 6、7、8、9 只有四种数字，所以剩余两个单元格又不能放 5，故只能是 1、2、3、4 的其二）。

我们对比两种情况，可以发现，6、7、8、9 是它们都有的，所以我们可以不用看它们，而特殊的数字（即不同的）只有 5，以及 1、2、3、4 的其中一组不同的数。由于此时 $r7$ 是填入的 2、3、4 的剩余一个数和 5，而 $r8$ 是 1、2、3、4 的其二，所以它们会如何呢？我们依然画出其中的构造出来的一种情况。

1	2							2	1		
2	4	5	3	6	7	8	9	1	3	2	5
3	1	6	8	9	2	4	7	5	1	4	6

如左图所示，这是其中的一种填法，可以看到，橙色的数字就是我们说的不相同的数字组合。我们先保证 $c12$ 的填数要一样，然后去修改其余位置的填数，就可以发现我们构造出来了一种填数形式，变为右图这样。

此时可以看到，实际上结构涉及的所有区域 **r578c123456789b4789** 的每一个区域上的填数都是一样的。比如 **b8** 里，原来的填法是 2、3、6、7、8、9，而在产生了交换后，依然是 2、3、6、7、8、9。其余的区域也是一样的。所以实际上，这种形式依然形成了致命形式。

虽然其中有一些数字没有发生位置的变换，但其它的单元格依然有交换。唯一解必须保证每一个单元格都不能有超出一种的填数情况，所以但凡出现有一个单元格能有两种填法，都应当是错误的。所以其中的一部分单元格不产生交换也没有关系。

接着，我们来看第三种情况。

1 2 3 4	1 2 3 4							
1 2 3 4	1 2 3 4							5
1 2 3 4	1 2 3 4							6

如图所示，我们依然按照之前的逻辑，假设 **r578c12** 的填数依旧是 1 和 2，那么我们依然可以得到一些合适的结果。

就 **r7** 而言，**r7c12** 一定是 1、2、3、4 的其二，而 **r7c9** 是 5，所以 **r7c345678** 只能填入 7、8、9 和 1、2、3、4 的其二以及一个 6；而就 **r8** 而言，**r8c12** 是 1、2、3、4 的其二，而 **r8c9** 是 6，所以 **r8c345678** 只能填入 7、8、9 和 1、2、3、4 的其二和一个 5。

可以发现，**r78** 里都会填入 7、8、9，所以我们就不用管了。所以我们可以按照上述的方式构造出来一个填法。

1	2									2	1						
2	3	7	8	9	6	1	4	5	4	2	7	8	3	6	1	9	5
4	1	5	2	3	7	8	9	6	1	3	5	2	9	7	8	4	6

如左图所示，我们随便找到一种填数方式，并按 **c12** 执行交换，先保证 **c12** 的填数不能发生变化，然后去更换其余位置的填数，并最终找到一种填法，如右图所示。

可以看到，这种构型依然是使得结构出现致命形式的，因为我们找到了两种不同的填数方式。所以，它依然是致命的，所以第三种情况也被我们证明了。

所以，我们利用了反证法，得到了三种情况全部产生了致命形式，所以这个结构是一个合格的致命结构。

1-4-4 淑芬致命结构的使用

1-4-4-1 死锁淑芬致命结构（Locked Qiu's Deadly Pattern）

4 7		3 6	4 9	3	4 7	6 7	4 6	2	1	8	5
4 7	8		6 7	5	9	1	4 7	6 8	4 6	2	3
4 8		1	2	4 8	6 5	3	4 6	9 7	4 9	7	
3	8		7 9	5	2	1	7 9		4	6	
2 5	2 5		6 7	4 7	6 8	4 7	6 8	9	3	1	7 8
1	4		6 7	9 7	8	3	7 8	6 9	2 7	5	8 9
4 7	5 9	5 7	9 7	1	2	4 7	6 8	4 7	6 8	4 5	3 6
4 7	2 9	2 5	3 7	1	4	4 7	6 8	4 7	6 8	2 5	3 6
6		2 7	8	3	4 7		5	4 7		9	1

如图所示，我们观察到，在 **r78** 两行内，确定值上下对应，并且区分度一定是 1，而发现 **r78c12** 里，在 **b7** 里的 5、7、9 均只能放在这四个单元格上。如果此时 2 也只能放在这里面的话，四个单元格将会产生 2、5、7、9 的四数组结构，就形成了淑芬致命结构的致命形式。所以我们不可以让 2 放在其中的任意一处地方，故 **r8c12** \neq 2。

1-4-4-2 另一个示例

2	5 6 9	5 6 8	4	3	7	5 8 9 1 5 8	1 6 9 3
4	5 3 9	7	1 6 8 9	1 8 9	1 6 8	2	5 8 9 3
6 8 9 1	3 6 8	5	2	4	7	3 6 9	
7	6 8 2	1 6 8 9	4	1 6 8	3	6 9 5	
1	4	5 6 2	5 9 3	7	6 9 8		
3	5 6 8	9	6 8 7	5 6 8	1	2 4	
5 6 8	2	1 6 8	3	1 5 8 9	9	5 8 4 7	
5 8 9 7	4	1 8 2	1 5 8 9	6	1 3 1 3 5 8 9		
5 8 9	3 1 3 9 8	7	6	4	5 8 9 1 5 8	2	

如图所示，在 b5 里的 1、6、8 只能放在 r46c46 里。如果此时 5 也在里面的话，就会使得 1、5、6、8 形成四数组，而 c46 此时的区分度为 1，即立马形成淑芬致命结构的致命形式，所以我们不可以让任意一个数字 5 放在 r46c46 里，于是删除掉它们。

再来看一种变种。

1	2	<div>6 8 9</div>	<div>6 7 9</div>	<div>6 8 9</div>	4	3	5	<div>6 7 8</div>
<div>6 9</div>	7	3	1	<div>6 8 9</div>	5	<div>2 6 8 9</div>	<div>6 9</div>	4
4	<div>5 8</div>	<div>5 6 8 9</div>	<div>6 7 9</div>	2	3	<div>6 7 8 9</div>	<div>6 7 9</div>	1
<div>2 3 5 9</div>	<div>3 6 5 6</div>	<div>2 7 9</div>	8	<div>3 6 7 9</div>	<div>6 9</div>	4	1	<div>2 6 7</div>
8	1	4	5	<div>6 7</div>	2	<div>6 7 9</div>	<div>6 7 9</div>	3
<div>2 3 9</div>	<div>3 6 7 9</div>	<div>2 7 9</div>	4	<div>3 6 7 9</div>	1	5	8	<div>2 6 7</div>
<div>3 5</div>	<div>3 2 5 8</div>	<div>2 5 8</div>	<div>2 6</div>	1	7	<div>2 6 8</div>	4	9
<div>2 6</div>	9	1	3	4	8	<div>2 6 7</div>	<div>2 6 7</div>	5
7	4	<div>2 6 8</div>	<div>2 6 9</div>	5	<div>6 9</div>	1	3	<div>6 8</div>

如图所示，如果 r7c2 \neq 8，则 r7c12 一定是 3、5，而这两个数在 b4 和 r46 交集的这六个单元格上只能放在 r46c12 里，这就说明了其余两个单元格不管填入的是什么，由于 r46 的区分度为 1，而 r46c12 里必须填 4 个不同的数字，所以不管是 3、5 还有哪两个数字，总会形成淑芬致命结构的致命形式。所以为了规避结构的出现，r7c2 \neq 8。

1 2 3 5	1 2 3 1	1 3 1	1 3 1	1 3 1	7 8 9 6	4 7 8 9	3 1 8 9	1 3 9	1 3 4	1 3 1	1 3 1
1 3 1	6 7 8 9	4 7 8 9	2 7 8 9	1 3 1	1 3 1	5 7 8	1 3 1	1 3 1	6 7 8 9	4 7 8 9	2 7 8 9
7	8	9	5	3	1 4 1	6	2	7	8	9	5
9	1 3 1	5 5	1 3 1	1 3 1	4 5 8	6	2	7	1 4 5	8	8
8	4	2	1 3 7	1 3 9	1 3 7	6	3	1 5 1	8	9	5
1 3 5	7	6	2	1 4 5	1 3 8	5 8 1	4 9	1 3 1	5 8	9	5
1 2 4	9	1 3 7	1 3 8	1 3 1	4 5 7	5 7	1 2 4	6	1 2 4	9	5
1 2 3 4	1 2 3 7	1 3 8	6 7 8	1 3 1	4 5 7	1 3 9	1 2 4	1 3 1	4 5 9	6	5
6	5	1 4 7	9	2	1 4 7	8	3	6	5	1 4 7	9

首先，我们观察 **b1** 里存在一个 ALS 区域 $\{\mathbf{r1c23}, \mathbf{r2c1}\}(1235)$ ，如果 2 和 5 同时为假，则显然三个单元格无法凑够三种数字，此时只有 1 和 3，所以出错，所以 $\mathbf{r1c2}(2)$ 和 $\mathbf{r1c3}(5)$ 不同假（或者看互补形式，图中标注的 $\mathbf{r1c1}(25)$ ，同假导致 2 和 5 同时挤进 $\mathbf{r1c1}$ 导致出错）。

而当两个数同真时，一样有 $r1c8 \neq 4$ ，否则 $r12$ 的确定值上下对应起来，且全部出现的 2、4、5、6 都可以找到一一配对的形式，所以此时区分度为 0，即出现反转拓展矩形的致命形式（看候选数就是， $r12$ 的空格里必须都填 1、3、7、8、9，而随便填出一种合适的状态后，上下可以对应位置交换形成两个不同的填法），所以 $r1c8 \neq 4$ 是成立的，如右图所示。

360

1-4-4-4 稍微用一点强制推导的淑芬致命结构

3	4	6	2	1	8	7	5	9
² 8	9	² 8	^{4 5}	^{4 5}	7	1	6	3
5	7	1	3	9	6	2	8	4
4	^{1 2} 8	^{2 3} 8 9	6	^{2 3} 5 8	^{1 2 3} 5	⁵ 8	³ 9	7
² 8 9	6	^{2 3} 8 9	7	^{4 5} 8	² 4 5	⁵ 8	³ 9	1
¹ 7 8	5	³	9	^{3 1} 8	³	4	2	6
6	^{1 2}	⁵ 7 9	¹ 5	^{2 3} 5	^{2 3} 5 9	³ 9	4	8
^{7 8 9} 1 2	3	^{4 5} 7 9	^{4 5} 8	^{4 5} 7	^{4 5} 9	6	1	2
^{8 9}	⁸	⁴ 9	¹ 8	6	⁴ 9	³ 9	7	5

如图所示，假设 $r6c1(1)$ 为假的话，则 $r4c2(1)$ 为真。此时发现 $r45$ 的区分度为 1 了。如果此时我们让 $b4$ 里的所有 2、8、9 卡在 $r4c3$ 和 $r5c13$ 里的话，结构将立刻形成淑芬致命结构的致命形式，所以 $r6c1(8)$ 必须为真。所以， $r6c1(18)$ 里必须有一个为真，所以跟 7 无关，删掉 $r6c1(7)$ 。

这个例子有一点超纲，需要学习了传递性质后才能更快地理解这个示例。不过你可以类比之前的证明方式来对这个结构进行唯一性的验证，看看它是否会形成致命形式（如果条件都达到了的话）。实际上给出之前的条件后，一样是可以论证的，只是稍微慢一点。

1-4-4-5 强制分析带有嵌套淑芬致命结构的思路

3	1	^{4 5}	2	7	8	⁴ 5	9	6
² 5 6	² 8	² 4 5	1	^{5 6}	^{5 6} 9	⁴ 5	7	^{4 5} 3
^{5 6}	9	7	³ 6	4	³ 5 6	8	1	2
^{1 2} 8	4	⁹ 8	^{1 2} 7	^{8 9}	⁷ 3	6	5	
^{1 2}	6	³ 9	^{1 2} 5	⁴ 2	^{1 3} 7	8		
7	5	² 8	^{4 6} 3	⁴ 2 6	9	² 4	1	
² 4 5	^{2 3} 8	6	9	² 5 4 5	^{2 3} 1	^{2 3} 5	7	
² 4 5	^{2 3} 1	7	² 5 4 5	^{2 3} 6	^{2 3} 8	^{2 3} 9		
9	7	² 5 6	³ 8	² 5 6	1	^{2 3} 4		

这个示例稍微复杂了一些，我们稍微来看看。

先看左图。我们尝试假设出两种填数模式，关于 $c4$ 的数字 9。由于 $c4$ 上数字 9 的填数位置只有 $r4c4$ 和 $r7c4$ 两处可填，恰好是两种情况，那么我们构造毛刺技巧，来利用上两种不同的 9 的填数位置的情况。

当 9 放在 r7c4 时，我们尝试找出从 r2c3(2)为假的链结构。当 r2c3(2)为假时，r9c3(5)此时必须也为假。观察 r2c3(2)，如果它为假，那么 r2 存在数字 8 的共轭对在 r2c12 上，而此时如果让 r2c3 <> 2 的话，2 也只能放在其中，导致 r2c12 构成 2、8 的隐性数对结构；而此时发现 r78 的区分度为 1，这是淑芬致命结构的一大要求，所以此时淑芬致命结构即将构成。唯一避免结构的发生的地方只有 r9c3。如果 r9c3(5)为真的话，就意味着 r9c3 无法填入 2 或者 8，使得 2 和 8 在 b7 里只能放在 r78c12 之中，而这是淑芬致命结构的最后一大形成条件，所以致命形式便形成了。所以，为了避免该致命形式的形成，我们不得不规避 r9c3 = 5 的要求，故 r9c3 <> 5。后面就是普通的推导模式了。

而当 9 放在 r4c4 时，我们有右图这样的强制链结构，并最终得到 r2c8 = 5 的结论。

不过，两种情况至少有一个是成立的，但它们都能得到 r2c3 <> 5 的结论，故 r2c3 <> 5 便是这个示例的真正结论。

1-4-4-6 双淑芬致命结构 (Dual Qiu's Deadly Pattern)

		4	5	4	2	3	6	1	8
7	9			7					
	2	1 2			1	1			
		4	8	6	4 5	4 5	9	3	4
7					7	7			7
6	1		3	9	1	1	2	5	
	4				4	4			4
					7 8	7 8			7
3		5	4	1	9	2	8		6
	7							2	5 6
1		5		8	4 5 6	4 5 6	3	4	
	7		9					7	9
2								2	
	9	8	6	3	4 5	4 5	4 5		1
					7	7		9	
5		2	1 2		1		1		
		6		4	4	6	9	4	8
				7	7				3
4	3	7	2		1	1			
					6	6	5	6	5 6
					8	8		9	9
8		1		5	3	1			
	6				4	6	7	4	6
		9							2

实际上，结构其实是不需要那么标准的，里面的那个“数对”就可以替换成任何一个区分度为 1 的东西，比如上图这样。之所以这么说，是因为区分度为 1 依然可以保证产生的最终两个对应位置一定是这样的数对形式，进而使得产生淑芬致命形式的那个“数对”。

如图所示，如果 r2c1(2)为真，则 r23 的区分度为 1，这便使得剩余的 r23c29 里随便找出一个上下对应的两个单元格都不可能是相同的填数，再配合 c56（此时 c56 本来就是区分度为 1 的结构）就一定会产生淑芬致命结构的致命形式。所以 r2c1 <> 2。

这个结构实属罕见，所以为了纪念这个结构，取了一个名字叫**双淑芬致命结构** (Dual Qiu's Deadly Pattern)。

1-4-4-7 双淑芬致命结构的另外一个示例

5 6 7 8 9	5 6 7 8 9	4	5 6 7 9	1 2	3 6 7 8 9	3 8 9
1 2 6 7 8 9	2 5 6 7 8 9	1 5 8 9	3	5 6 7 8 9	5 6 7 8 9	4 2 6 7 8 9
3	2 6 7 8 9	8 9	4	6 7 8 9	6 7 8 9	1 5 2 8 9
2 5 8 9	3 6	2 5 8 9	4 1	2 8	2 8 9	7
4	2 5 7 8 9	5 8 9	2 6 7 9	5 6 7 8 9	5 6 7 8 9	3 1 2 6 8 9
1 2 7 8 9	2 7 8 9	1 8 9	2 6 7 9	3 7 8 9	3 7 8 9	2 6 8 4 5
5 8 9	1 7	5 9	3 5 9	4	2 5 8	2 3 8 6
5 6 8 9	4 5 6 8 9	2 8 9	1 7 9	3 5 6 7 9	3 5 6 7 9	5 7 8
5 6 8 9	4 5 6 8 9	3 8	2 7	5 6 7	9	1 7

如图所示，如果 $r2c1(1)$ 为真，则 $r23$ 的区分度为 1；而此时 $c56$ 区分度本身就为 1，所以如果 $r2c1(1)$ 为真，则必然能找到其中上下对应的两个单元格填入的是 $r23c56$ 的其中四种数字，而此时 $c56$ 区分度为 1，所以形成淑芬致命结构的致命形式。故 $r2c1(1)$ 为假，删掉。

最后我们再给一个双淑芬致命结构，长相跟上面这个差不多，不过，这个题如果不用这个技巧，难度可能会比较大。

5 7	2 3 7	9	8	3 5 6 7	4	1 2 6	1 2 3 6	1 2 3 6
1	2 3 4 7 8	2 3 8	2 3 7 9	3 6 7 9	2 3 7 9	5	4 6 8 9	4 8
4 5 8	2 3 4 8	6	1 2 3 5 9	3 5 9	1 2 3 5 9	2	2 3 4 8 9	7
4 7 9	3 1 4 6 7 9	5 5	1 3 5 7 9	2	1 3 5 7 9	8	1 3 6 7 9	1 3 6
7 8 9	7 8 9	3 1 5 8	4	3 5 7 9	6	1 2 7 9	1 2 3 7 9	1 2 3
2	3 1 7 9	3 6	1 3 7 9	8	1 3 7 9	4 5	1 3 6	3 6
3 6	5 4	2 3 7	3 7	8	1 2 6 7	1 2 6 7	9	9
3 6	1 7	2 3 5 9	4	2 3 5 9	2 3 6 8	2 6 8	5 8	5 8
8 9	8 9	8	6 1	5 7	3	4 7	4 5	4 5

如图所示。如果 $r4c1(4)$ 为真，则 $r46$ 和 $c46$ 的区分度均为 1，此时形成淑芬致命结构的致命形式。所以 $r4c1(4)$ 为假。

那么，淑芬致命结构的内容就介绍到这里。接下来我们来看的是一个方便使用的全新理论：**传递性 (Transmission)**。

1-5 致命结构的传递性 (Transmission of DP)

在之前的技巧里，我们发现了不少的解题思维和方法，不过这些结构都过于庞大，如果每一次我们都尝试去论证一番的话，可能会耗费很多时间，而且也做不到致命结构的灵活性，接下来我们就来看一下一种新的视角，用来论证致命结构的是否：**传递性 (Transmission)**。想要使用好传递性，是一件非常困难而且富有挑战性的事情。

1-5-1 先来看一个示例

7	5 8 9	5 9	6	2 5 9	2 5 8	3	4	1
1 8	4	6	5 7	3 5 8	1 7	2 5	9	2 5
2	1 5 9	3 5 9	5 7 9	4	1 5	6	8	5 7
1 9	1 6 9	7	4	2 5 9	3	2 5	2 5 6	8
3 8 9	3 8 9	4	2 5 9	6	2 5	1	2 3 5	2 5 7
5	3 6	2	8	1	7	4 9	3 6 4	9
1 6 9	2	1 5 9	1 5 7	8	4 6 4	9 7	5	3
3 6 9	7	5 9	2 3 5	2 3 5	4 6	8	1	4 9
4	1 5	3 8	1 5 7	3 5	9	2 5	2 5	6

如图所示，这个结构就是之前我们说到的那则示例。如果 **r78c1** 只能是 6 和 9 的话，这便会使得这个结构涉及的所有区域（行、列、宫）里都会产生一个数对或三数组结构，而显然这个结构形成的形式是可以产生交换的，而且交换的填数模式并不会影响什么，因为填数还是那些数字，而交换并不会影响到数对和三数组本质的变化，所以这样便产生了致命形式。

而我们的传递性是这样的一个方式来“降解”结构。我们发现，{**r6c79**, **r7c7**, **r8c9**}(**49**)是四个只有 4 和 9 的单元格，如果我们补充两个单元格在诸如 **r78c5** 的位置，让它们也只能放下 4 和 9，这一定会使得原本四个单元格和我们新添加的两个单元格共同构成的六个单元格形成 UL 的致命形式。所以，我们可以这么去想这个问题。既然我们能够让这个结构形成致命形式，那么我们就去掉这四个单元格，而与之让结构补上 **r78c5(49)**，然后再去论证剩下的这一部分是否能够形成致命形式即可。

接着，我们再来看，剩余的六个单元格（注意，此时原本只剩下四个单元格，但由于刚才我们补充了 **r78c5(49)**，此时我们也算进去）将构成拓展矩形的致命形式，所以原结构也可以说明是致命的。

这个说法听起来相当奇怪，我们甚至随意占用了一些我们结构完全没用到的单元格，却用来辅助证明了致命形式是否会出现。虽说是这样，不过我们依旧可以给出简要的证明。

1-5-2 原理进一步剖析

刚才我们提到，这种借用其它单元格作出辅助证明的方式确实有点别扭，但实际上它确

实奏效了，是巧合还是真正可以这么用呢？实际上，这种变换形式是完全不依赖盘面的。也就是说，我们在论证结构的致命与否时，只需要看结构本体所在区域即可，而其它的位置都是不用我们管的，因为我们知道，致命结构本身就不依赖于其它单元格。

我们尝试把结构剥离出来，不过这个时候我们反过来思考这个问题，从一个拓展矩形（双值格版的）和一个唯一环开始入手。

6				4		4	6	
9					9			
6				4		4	6	
9					9			

如图所示，我们剥离出了这个结构，此时我们可以尝试通过这个结构来构造出更大的致命结构。

试想一下，要想使得整个结构本身是形成致命形式的，就必须让结构的每一个区域不会收到额外数字的影响。为了让其它数值不会受到丝毫影响，我们只得去往结构里去“加”一个致命结构。这样一来，由于都是致命结构的原因，它们各自的区域显然是互不影响的。

此时，我们依赖于这个结构里的 4 和 9，添加上一个额外的 4 和 9，使之构成唯一环，如图所示。

						4		4
							9	9
6				4		4	6	4
9					9			9
6				4		4	6	
9					9			4
								9

如图所示，我们就构造出了一个结构，它们共用了 **r78c5(49)**。不过很显然，这样的构造是不合理的，因为此时 **r78** 里产生了 **4** 和 **9** 的数对，直接就相当于破坏了这个结构整体。那么我们需要消除这种破坏，于是需要删除掉一些单元格，让结构依然是致命形态，还不会影响其它单元格。

此时我们考虑 **r78c5(49)**。这是它们共用的地方，如果删除掉，我们既能保证行上依然还是合适的数组形式，而其余位置也不会受到影响。太棒了！我们开始删除 **r78c5(49)** 这一部分。

						4		4
							9	9
6					4	6	4	
9							9	
6					4	6		4
9								9

这样，我们构造的致命结构就出现了，其余的单元格不会受到影响，因为两边最初的结构都是单独的致命结构，它们独自本身就不会影响到其余位置，而剥离掉的 **r78c5(49)** 也都是共用的、多出来的那一部分，所以结构不会因为删掉它们俩而变得不稳定；相反，它变得稳定了。

那么，我们体会到，实际上结构都是通过从小到大构造形成的，所以反过来，我们可以

通过补充一部分原本需要的单元格，然后再把补充的那一部分给它剔除掉，只要剩余的部分是致命的，那么原结构必然也一定是致命的，因为我们剔除掉和补充的部分恰好组合形成一个合适的致命结构，而这些致命结构也都是通过我们之前的文字全部证明过了的。

现在我们来看一部分可以通过传递得到论证的示例。

1-5-3 传递得到的一些示例

1-5-3-1 一个比较简单的传递示例

3	9	4	8	7	6	5	2	1
7	2	5	1	9	3	8	4	6
6	8	1	2	4	5	7		
5	¹ ₄		³ ₆	³ ₄	¹ ₆	8	9	7
8	7	9	⁴ ₃	¹ ₅	⁶ ₂	⁴ ₆	¹ ₃	⁵ ₉
2	¹ ₄		³ ₆	7	¹ ₅	⁶ ₉	⁴ ₆	¹ ₃
4	⁵ ₆	7		⁶ ₉	2	1	3	8
1	⁵ ₆	8		⁶ ₉	3	7	2	⁵ ₉
9	3	2	5	8	4	1	6	7

这一则示例相信我们不用剥离出结构也能理解。我们可以看到，r35c89 还需要补充两个单元格即可构成拓展矩形的致命形式，而我们就尝试补充到 r7c89 上，这是因为结构还有 r7c9，这样我们就只需要补充一个单元格。

这样补充好后，我们去掉 r35c89 和 r7c9 五个单元格，现在剩下了 r78c248（注意 r7c8 是我们才补充上去的单元格）。

显然，这样的结构是出现了关于 5、6、9 的拓展矩形的致命形式的，所以原结构是致命的。

1-5-3-2 用了唯一矩形和拓展矩形的传递示例

8	1	3	1	3	2	4	5	9	1	6	5	6
9	7	1	4	2	5	3	6	1	4	8	1	5
5	4	7	6	8	9	1	4	7	2	3		
1	9	7	4	2	3	6	5	8				
2	8	4	5	6	7	3	9	1				
3	6	5	9	1	8	2	7	4				
3	2	8	1	3	5	9	1	4	7	6		
4	5	1	3	1	3	7	2	1	8	6	9	
7	1	6	9	1	6	8	4	5	3	2		

如图所示，这一则示例比较麻烦，我们给出简图。

	1	3	1	3								
3	6	3	6									
3	6			1	3							
		1	3	1	3							
	1	6		1	6							

如图所示，我们可以很明显地看到，r6c12 显然是一个可能构成的 UR，我们尝试把 r6c12 和 r7c1 这三个单元格去掉，给它“补”到 r7c2 上，于是变为这样。

	1	3	1	3							
		3			1	3					
		6				6					
			1	3	1	3					
						6					
	1				1						
		6				6					

接着，我们关注 **r18c23**，它也是一个看起来很像是 **UR** 的形式，我们尝试对这部分进行修改，把 **r1c23** 和 **r8c3** 三个单元格去掉，补到 **r8c2** 上。

[illegible]

可以看到，此时这个结构虽然看起来别扭，但实际上确实是一个拓展矩形，而且它形成了致命形式，所以原结构是致命的。

1-5-3-3 依赖于拓展矩形的稍大一些的致命结构

3	4	¹ 5	9	² 5	¹ 2	¹ 2	6	
² 7		¹ 8		⁸ 7	² 1		⁷ 8	
⁷ 2	9		6			3	5	4
⁷ 2	6	¹ 5	² 5	4	3	¹ 2	9	⁷ 8
5	3	6	4	7	9	8	1	2
9	2	4	3	1	8	5	7	6
1	8	7	² 5	² 5	6	9	4	3
8	5	3	1	6	4	7	2	9
6	7	2	8	9	5	4	3	1
4	1	9	7	3	2	6	8	5

如图所示，这个结构依旧可以通过之前的说法执行论证。不过这个示例就不画图了。我们挑战一下。首先，我们关注于 **r23c1356**，把这个疑似致命结构的形式，去掉结构涉及的六个单元格，而补充到 **r3c56** 上，注意，补充的数字一定要和原结构要匹配，比如 **r3c5** 显然是补充 2 和 8 的，而 **r3c6** 则只能补充 1 和 7。

接着，我们发现，补充后，剩下一共八个单元格，此时依然是构成拓展矩形的致命形式的，所以原结构是致命结构。

1-5-3-4 一个超大的致命结构

我们来看一则非常大型的致命结构。

⁴ 5	6	7	³ 9	8	⁴ 9	1	² 5	² 3
8	3	2	5	6	1	4	9	7
⁴ 5	9	1	³ 7	2	⁴ 7	³ 5	⁶ 8	³ 5
7	8	9	1	4	3	⁵ 6	² 5	² 5
6	1	5	² 7	9	² 7	8	3	4
3	2	4	8	5	6	7	1	9
¹ 2	5	6	² 9	7	8	³ 9	4	¹ 3
9	7	3	4	1	5	2	6	8
¹ 2	4	8	6	3	² 9	⁵ 9	7	¹ 5

如图所示，这个结构是一个致命结构，而这个结构可以删除的是 **r1c9(35)**，说明 **r1c9** 如果只有 3 和 5 则会形成致命形式，所以我们把结构剥离出来，看看它是如何形成致命形

式的。

4 5			3		4			3
			9		9			5
4 5			3		4	3		3
			7		7	5 6		5 6
			2		2			
			7		7			
1 2			2			3		1 3
			9			9		
1 2					2	5		1
					9	9		5

如图所示，我们剥离出了结构，不过这一点有一些让我们难下手，因为我们并没有比较好的视角，从哪里开始并不好确定，毕竟结构很大。没有关系，我们可以观察 **b3**。**b3** 只有三个单元格，这让我们想到，要让它也是致命结构的一部分，显然是三数探长致命结构的一部分才行，所以我们尝试在它的两侧都添加一个 3、5、6 的单元格。当然，只要结构是致命的，我们就可以随意添加数值，所以我们完全可以不用添加完整的 3、5、6，而是直接以数对的形式添加就可以了。比如，我们在 **r13c3** 处添加一个 3、5 数对，而 **r4c79** 上也添加 3、5 数对，随之去掉 **r1c9** 和 **r3c79** 三个单元格。

4 5		3	3		4			
		5	9		9			
4 5		3	3		4			
		5	7		7			
						3		3
			2		2	5		5
			7		7			
1 2			2			3		1 3
			9			9		
1 2					2	5		1
					9	9		5

添加 3 和 5 的原因很简单，就是为了保证后面推导可以顺利进行。如果添加 3 和 6，可以看到 **b9** 给出的数字是 1、3、5、9，如果多出了 6 就不好处理了；同理 **b12** 的结构数值分布也是这样的道理。

接着，我们可以先走下面来处理。我们如果在 **r79c2** 上添加一个 1、9 的数对，那么

r4c79、r79c279 这八个单元格将构成关于 **1、3、5、9** 的探长致命结构的致命形式，所以我们尝试去掉这六个原本存在的单元格，而都添加补充到 **r79c2** 上，结构变为了这样。

4 5		5	3	3	4			
				9	9			
4 5		5	3	3	4			
			7		7			
			2		2			
			7		7			
1 2	1			2				
		9		9				
1 2	1				2			
		9			9			

接着，我们发现，**r79c12** 可以补充为一个拓展矩形，所以我们再次借用 **b9**，使得 **r79c12** 构成拓展矩形的一部分，比如我们尝试为 **r79c8** 补充一个 **2** 和 **9** 的数对，于是结构就变为了这样。

4 5		5	3	3	4			
				9	9			
4 5		5	3	3	4			
			7		7			
			2		2			
			7		7			
			2				2	
			9				9	
					2		2	
					9		9	

如图所示，我们此时又发现了，我们如果此时为 **r6c46** 补充 **2** 和 **9** 的数对，**r6c46、r7c48** 和 **r8c68** 将构成关于 **2** 和 **9** 的 UL 的致命形式，所以我们把 **2** 和 **9** 补过去。

4	5		3	3		4			
			5	9		9			
4	5		3	3		4			
			5	7		7			
				2		2			
				7		7			
				2		2			
				9		9			

其实已经可以发现，结构开始渐渐变小了。我们接着往下走。可以发现，**r56c46** 只涉及 2、7、9，如果我在下面补充一个 7 和 9 的数对，这四个单元格将变为一个拓展矩形的一部分，所以我们在下方添加一个数对，然后补过去。

4	5		3	3		4			
			5	9		9			
4	5		3	3		4			
			5	7		7			
				7	9	7	9		

差不多了，我们继续发现 **r13c13** 是一个疑似关于 3、4、5 的拓展矩形的一部分，所以我们补充一部分，使之为拓展矩形的一部分。所以我们继续让结构变为我们需要的那样。

			3		4		3	
			9		9		4	
			3		4		3	
			7		7		4	
			7	9	7	9		

可以看到，b2 里涉及的是 3、4、7、9，而 r13c8 是只有 3 和 4 的，而 r7c46 只有 7 和 9，这显然是一个探长致命结构的致命形式。所以不用再分解了，而也可以论证得到的是，原结构是一个致命结构。

最后我们来看一则带链的致命结构的示例。

1-5-3-5 毛刺致命结构

1	1	5	4	2	7	6	1	3	1	1	5	4	2	7	6	1	3
8 9	8 9						8 9		8 9	8 9						8 9	
3	1	6	5	8		4	1	2	3	1	6	5	8		4	1	2
7	9	7				6	9		7	9	7				6	9	
6	4	2	3	1		5	7	8 9	6	4	2	3	1		5	7	8 9
8 9									8 9								
6	6	4	9	3	5	1	2	7	6	6	4	9	3	5	1	2	7
8	8								8	8							
2	5		6	1		3	4		2	5		6	1		3	4	
7	9	7	8	9		8	9		7	9	7	8	9		8	9	
1	3	7	4			6	5		1	3	7	4			6	5	
9									9								
5	6	8	1		3	2	4	6 9	5	6	8	1		3	2	4	6 9
7	9								7	9							
4	3		6	5		7		1	4	3		6	5		7		1
			9				8 9					9				8 9	
1	6	2	1			4	3	5	1	6	2	1			4	3	5
7	9		7	9	8	7	9		7	9		7	9	8	7	9	

如左图所示，假设 r8c4(6)为假，则可以得到 r5c1(7)=r6c1(1)的结论，此时我们使用该强关系可以得到一条不连续环，并最终得到 r6c1 <> 9；但由于 r8c4(6)为真，则可以引出强制链，进行直推可以得到，r6c7(9)为真的结果，此时我们依然可以删除 r6c1(9)。所以，r6c1 <> 9 即为所得。

可以看到，这一则示例里我们可以使用毛刺的方式来得到我们想要的结论，不过它借助了这种比较复杂的致命结构（不过实际上可以看到，这种致命结构论证起来比起之前的示例要简单一些）。

1-6 含有隐性视角的致命结构 (Hidden Deadly Pattern)

实际上我们已经学会了大部分的致命结构了，不过依然有一些结构论证起来非常复杂，而通过之前的方式依然是很难找到切入点，甚至无法找到切入点。下面将选取一部分这样的形式作为讲解的内容。

致命结构除了之前的逻辑以外，我们还可以为其嵌入隐性视角，这样的结构称为**隐性致命结构 (Hidden Deadly Pattern)**。这种结构看起来复杂，证明起来，更复杂。

1-6-1 带有隐性视角的致命结构？

2	9	1	6	8	5	3	7	4
4	7	8	3	9	1	6	5	2
6	5	3	4	7	2		1	1
8	²	7	9	5	3	1	²	6
1	⁴	³	4 5	7	2	6	⁵	³
9	^{2 3}	²	5 6	8	1	4	7	⁵
	³	1 2	²	1	5	6	8	4
7	8	4	6	1	5	4	³	9
	³	1	6	4	6	9	2	4

如图所示，这个结构是一个致命结构，不过它有点奇怪的是，以往我们理解致命结构都是通过显性的层面来思考的，不过这一次我们需要隐性来理解了。可以发现到的是，r35 的所有填入 8 和 9 的位置只有 r35c789，那么 r3 里放下 8 和 9 的地方就只有 r3c789 里选其中两个单元格；同理，r5 也是一样。

如果 r9 里放下 8 和 9 的位置也只能是在同 r35 的对应位置 r9c789 里的话，会发生什么现象？显然，8 在里面必须有三处要放 8，而 9 也是。不管怎么放 8 和 9，你都会发现，8 和 9 的放置最终都会形成唯一矩形或唯一环的致命形式，即 8 和 9 可以交换的局面。所以，r9c789 里不能有任何一处可以放 9 的地方，所以 r9c789 <> 9。

这一则示例我们可以看到，如果 r9c789 是 r9 里唯一能放下 9 的三处地方的话，那么不管 8 和 9 的放置方式如何，最终都将致命，所以这一种使用方式非常的神奇和新颖。

1-6-2 又是一则帶有一點隱性視角示例

这个例子我们先来说说结论。如左图所示，如果 **r4c1(2)**和 **r7c4(1)**同时去掉的话，我们画出来的这些单元格将会组成一个致命形式，于是我们能够找到一条链，并使用刚才的强关系，得到一条不连续环，并得到删数。

那么，这个致命结构是如何产生的呢？我们可以将这个结构的构型提取出来。

				2	2			
			4	4	4			
			2	2	2			
			4	4	4			
4			2	2				
7			4	4				
			7	7				
4			3	3				
7			4	4				
			7	7				

如图所示。我们可以看到，这个结构的下方 **r79c145** 只有 2、3、7、9，我们完全可以使用传递性，将这些单元格传递变为一个 2、4 的数对。

				2		2		
				4		4		
			2	2	2			
			4	4	4			
				2		2		
				4		4		

别急，我知道你肯定有一处疑问。**r6c45** 如果是 2 和 4 的话，岂不是影响到 **r4c456** 的 2 和 4 的填数了？看起来确实是，不过我们需要注意的是，这里宫内的 2 和 4 实际上和它是不构成直接的影响的关系的，因为 **r6c45** 的这个 2 和 4 实际上并不一定在这个位置，这只是我们在传递的时候，为了固定数字，并持续向下推导时选择好的一处地方而已，所以我们不能直接说这个 2 和 4 是可以影响到 **r4** 的三处 2 和 4 的单元格的。

实际上在之前的示例里，我们完全也可以把一些传递后的数对或者数组结构安放在看似会冲突的地方上去，这一点的原因和上面我解释这里 2 和 4 的冲突的原因是一样的，因为传递后产生的数对或数组结构的位置本身就是不固定的，固定下来也仅仅是为了更好地向下推导其逻辑。但是固定下来也不代表它一定就在这里，因为我们只是拿出了构型来推理，可以很明显地发现，原盘面多数情况下，在我们传递好的位置上是有确定值的，所以实际上就安放传递后的数组结构的位置这一点来看，实际上只要能 and 原结构保持致命形式的出现，放在哪里其实并没有什么关系，只是要注意，不要把传递的结构和原结构拿来混淆，否则有时候完全区分不开，导致无法继续推导，或者错误地使用了该传递性。

以前的示例里，我们可以发现的是，传递后的结构的位置都是放在一些完全不受影响的地方，这一点就是为了避免错误地理解我们此处提到的可能产生混淆的内容。

接着我们就可以使用刚才第一则示例的逻辑了。我们不论怎么安放 **r4** 的 2 和 4 的位置，由于结构 2 和 4 在 **r36** 的 2 和 4 的位置是固定的，所以这三处 2 和 4 的安放只会使得这些位置要么形成 UR 的致命形式，要么形成 UL 的致命形式。所以，整个结构实际上是致命的。因此，最初得到的强关系得以论证，即我们原定要消失的数字并不可以同时被去掉。

1-6-3 综合示例

1	2	3	7 8	9	4	7 8	5	6
7	4	5 6	5 6	1	5 6	2	3	4
8	4	5 6	5 6	1	3	5 6	2	4
9	1	4	3	2	8	6	7	5
6	7 8	7 8	4	5	1	2	1	3
5	7 8	7 8	9	6	1	2	1	3
2	7 8	5	6	4	3	7 8	9	1
4	9	1	5	8	7	3	6	2
3	7 8	6	2	1	9	5	4	7 8

最后我们来看一则比较麻烦的示例。如图所示，这是一个致命结构，如果 **r6c8(4)** 去掉的话，它将形成致命形式。现在我们来证明思考一下这是为什么。

我们先把结构抽象出来。

	5 6	5 6		5 6				
	5 6	5 6			5 6			
	7 8	7 8		5	1		1	
	7 8	7 8		7	5		8	
					1		1	
				7	6	6	8	
	5	5						
	7 8	7 8						
		6						
	7 8	7 8						

如图所示。注意，**b14** 里的 5 和 6 还有 7 和 8 是隐性的视角，仅这些单元格可以填入，并非是提取过程之中出现错误。

首先，我们观察到，**b5** 里的数字是 1、5、6、7，所以我们完全可以在它的两侧补上相应的结构，使之形成一个合格的四数探长致命结构的致命形式，为了题目更简单，我们在结构靠下方补上 5、6 数对，而右侧补上 1、7 数对，变为下面这个图里展示的这样。

	5 6	5 6	5 6					
	5 6	5 6			5 6			
	7 8	7 8				1		1
	7 8	7 8				8		7
	7 8	7 8				1		1
	7 8	7 8				8		7
	5	5						
	7 8	7 8						
				5 6	5 6			
	6	6						
	7 8	7 8						

接着，我们可以看到，**r56c89** 是 1、7、8，显然可以为左侧补充 7、8 使之构成拓展矩形的致命形式，而 5、6 可以发现可以补充 5、6 构成唯一环形式，所以我们补充上匹配的 5、6 数对和 7、8 数对，于是结构变为这样。

	5 6	5 6					5 6	
	5 6	5 6					5 6	
	7 8	7 8	7 8					
	7 8	7 8	7 8					
	5	5						
	7 8	7 8						
	6	6						
	7 8	7 8						

如图所示。可以发现，此时结构就比较清晰了。我们这样去思考这个问题。由于此时 **r23c8(56)** 是我们传递的结果，而 **r56c4(78)** 也是。虽然它们的位置并非必须要放在这里，但由于相对位置是不能发生改变的，即要保证原始传递过来的部分是可以还原回去形成致命形式的，所以现在的这个部分的相对位置是不允许发生任何的改变的。这也就是说，**r23c8(56)** 完全可以调整到 **r23c4(56)** 来，甚至是 **r23c1(56)** 来，但必须是 **r23** 里同列的单元格；同理，**r56c4(78)** 也是这样的道理。

那么基于这一点，我们注意 **b1**，发现 **b1** 里的 5 和 6 此时就只能斜着放了。因为横着放 5 和 6 会导致 **r23c8(56)** 传递出来的单元格的其一无法填数，导致出错；而竖着填就会立马配合 **r23c8(56)** 形成关于 5、6 的致命形式，所以数字 5、6 此时在 **b1** 里只能斜着放

进去。同理；b4 里的 7、8 也是一样的道理。

那么，既然只能斜着放，那么为什么我们不能视作唯一环的一部分呢？拿 r23c8(56) 来说，配合 b1 里 5、6 斜着放的特性，只要 c23 里出现同行的 5、6 数对，就算是形成 UL 的致命形式了，那么我们可以利用这一点，把这个隐性的视角通过传递性，转化到 c23 上某一同行的两个单元格上；同理，7 和 8 对于 b4 也是一样，所以我们传递的最终版本变为这样。

		7 8		7 8					
		5 6		5 6					
		5		5					
		7 8		7 8					
		6		6					
		7 8		7 8					

很显然，这是一个涉及四行两列的拓展矩形结构，它是致命结构，所以原结构是一个致命结构。故假设的 r6c8(4)为假就是错误的，故 $r6c8 = 4$ 。

1-6-4 不简单的小练习

那么下面我们来一些小练习。请问，这些结构是否是致命结构？或者，嵌套了这些结构的链是否使用正确？如果是，请说明其原因和该结构对应的结论；如果不是，请说明理由。

6	5	4 5	8	3	4 5	4 5	1	4	2	6	1	2	5	4	7 8	3
4 5	1	2	7	9	6	3	4 5	3	4	5	8	3	1	2	3	1
4 5	3	3	1	2	4 5	4 5 6	4 5	4	6	7	9	4	8	6	2	1
7	9	8	8	1	2	7	9	7	9	4	6	7	9	8	6	9
1	2	4 5	6	4 5	8	2	4 5	3	4	7	9	1	2	3	1	2
4 5	5	3	2	1	7	4 5 6	4 5	1	4	6	8	9	4	6	7	8
4 5	2	6	9	1	3	4 5 6	4 5	2	1	2	4	6	7	8	9	6
7	7	8	8	4	5	8	7	7	8	9	6	7	8	9	6	6
2	4	7	5	8	9	1	3	6	1	3	4	5	6	7	8	9
3	6	1	4	7	2	9	8	5	7	9	6	8	9	6	4	5
8	5	5	3	6	1	7	4	4	1	5	6	8	3	5	7	9

3	1 2	7	5	8	4	1 2	6	1 2	3	1	2	4 6	5	4 6	4 6	9	8	
1 2	4	5		7		1 2 3	8	1 2 3	7	5	5	4	2 6	1	8	2 6	5 6	3
8	6	9	2	3	1	7	5	4	7	5	8	6	3	4	2	1	4 5	4
5	1 2	6			3	4	7	1 2	4	2	5	6	3	6	6	8		1
9	7	8	1	2		1 2 3		2 3	6	3	1	8	2	4				5
1 2	3	4	7	5		1 2	6	9	1 2	8	7		1	4	3	5	4	2
7	9	2		1		5	4	6	5	9	4	8	7	6	3	2	5	1
6	8	3				2	1	7	1	5	3	8	2			5	6	4
4	5	1			7	8			2	6	7	5	4	1	3	8		4

1-7 无法传递或很难使用传递性的结构

下面来介绍一些不太好使用传递性和根本就不能使用传递性来证明其结构是致命结构的例子。

1-7-1 唯一矩阵（Unique Matrix）

我们先来看示例，看看它长啥样。

1-7-1-1 使用

1 5 6	7	9	1 2 5 6	2 5 6	2 5 6	4	3	8
2	4 3	4 3	4 6	9	8	7	5	1
1 4 5	1 8	4 5 8	1 4 5	3	7	6	2	9
8	6	2	7	1	9	5	4	3
4 5 7	4 5 3	5 3	5 6	8	4 5 6	1	9	2
4 5	9	1	3	4 5 2	4 5 2	8	7 6 7	6
9	2 8	4 5 6 7 8	2 5 6 4 5 6	2 4 5 6 4 5 6	3	1	4 5 6 7	
1 4 5 6 7	1 2 4 5 6 7	8	3	9	7 6 4 5			
3	4 5	4 5 6 7	9	4 5 6 7	1	2	8	4 5 6 7

如图所示。我们假设 r7c3(8)和 r78c5(2)同假，则 r789c359(4567)将构成一个合格的致命结构。这个结构叫做**唯一矩阵**（Unique Matrix），它和唯一矩形的技巧名就差一个字，但意义完全不一样。

1-7-1-2 证明

这个结构使用传递性将非常难解释，所以我们只能使用普通的枚举方法来证明这个结构是否是致命结构。由于结构涉及的数字是四种，而且是三阶的，所以我们尝试对结构里填入的数字分别假设为 a 、 b 、 c 和 d 来简化描述，并假设出四种情况：没有 d 、一个 d 、两个 d 和三个 d 。

	1 2 3 4			1 2 3 4			1 2 3 4	
	1 2 3 4			1 2 3 4			1 2 3 4	
	1 2 3 4			1 2 3 4			1 2 3 4	

如图所示，我们就假设图里的 $d = 4$ 。下面开始枚举每一种情况。

假设结构没有 4 的话，显然是致命的，因为任取结构涉及的两列，都会发现构成了拓展矩形的致命形式（竖着放的那种形态）。

假设结构有一个 4 的话，显然也是致命的，因为结构只能放 1 个 4，那么剩余的两列里必然也只有 1、2、3，使得依然形成了拓展矩形的致命形式。

假设结构里有两个 4 和三个 4 的话，证明过程就开始复杂起来了。所以我们尝试尽量用图片的形式呈现出来。先来看只有两个 4 的情况。

	4			1 2 3			1 2 3	
	1 2 3			1 2 3			1 2 3	
	1 2 3			1 2 3			4	

如图所示，假设我们只有两处 4，显然这两处 4 的位置是可以随意放的，所以我们找了一个比较对称的、方便观察的地方放下了 4。接着，我们可以观察剩下的七个单元格。显然，我们对 r456c5 的填法是随意的，所以我们就尝试直接放上 1、2、3；同理，r5c258 也是对称的，所以这里其实可以随意放 1、2、3，都是同样的情况。所以我们找出一种填法，如下图所示。

	4			1			2	
	1			2			3	
	2			3			4	

至于 r4c8 和 r6c2，是根据直接的排除效果得到的，不过显然，r46c28 构成了 UR 的致命形式，所以这种情况就错误了。

因为刚才我们的证明过程之中，虽然很多填法我们都没有直接实现，但因为结构是对称的，所以怎么去换数字的位置，实际上都是等效的，所以我们没有给出其它情况的证明。接着我们考虑有三个 4 的情况。

	4		1 2 3			1 2 3		
	1 2 3			4		1 2 3		
	1 2 3		1 2 3				4	

如图所示，我们找到三处填入 4 的位置，显然，和刚才一样，4 的位置是可以随意放置的，所以我们找一个比较对称的、好观察的填法。

接着，我们只需要处理剩余的六个单元格，看它们是否填入后形成整个结构的致命形式即可。而和刚才一样，我们不得不又开始随意假设情况，比如此时我们按 **r46c5** 填入 1 和 2 来看接下来的情况（2、3 和 1、3 的逻辑和这里的假设是等效的，所以不用再去证明 2、3 和 1、3 的组合的情况）；同理，**r5c28** 也是一样的道理。

	4			1			2 3	
	2			4			1	
		1 3		2			4	

接着，我们可以尝试看下 **r4c8** 和 **r6c2** 的填数情况。如果 **r6c2** 是 1，则 **r456c25** 构成拓展矩形的致命形式；同理，**r4c8** 如果是 2 的话也会这样，所以我们直接确定 **r4c8 = r6c2 = 3** 的结果。

	4			1		3		
	2			4		1		
	3			2		4		

但很显然，填入 3 后就可以看到，r46c28 构成了关于 3、4 的 UR 的致命形式（这已经回到了之前的证明的其中一步了），所以这种情况也不成立。所以我们就证明了，所有情况全部都出现致命形式，故整个结构是一个致命结构。

接下来我们继续来看一则使用示例，相信你通过证明后，就可以更快地得到结果了。

6	9	1	4	³	8	5	2	³
⁴	³	5	⁴	³	1	6	2	⁸
8	2	³	⁷	⁹	5	1	4	6
²	7	6	8	¹	9	3	5	¹
⁴	³	⁴	8	5	7	2	6	¹
²	1	5	3	²	6	7	8	⁴
⁴	⁹	³	2	5	4	⁶	1	8
5	8	⁴	⁷	⁶	1	⁴	3	2
1	⁴	⁶	2	⁶	8	⁴	7	5

如图所示，如果 r5c1(3)为假的话，结构 r456c159 将构成唯一矩阵的致命形式，所以 r5c1(3)为真，删掉 r5c1 的其余候选数。

1-7-2 唯一性提示信息覆盖（Uniqueness Clue Cover）

接下来要介绍一个比较难的唯一性技巧，叫做**唯一性提示信息覆盖**（Uniqueness Clue Cover，简称 UCC）。在外国的资料上，这个技巧的介绍也很少，其中提到了一句话，这个技巧是通过计算机编程验证，得到的删数，不过现在我们将尝试用逻辑解释这个技巧的删数原理，所以打起精神来，这个技巧将非常具有挑战性。

1-7-2-1 使用

1 2 3	1 2	3	2 3	4 5	2 3	1	1 3	1 3
4 5	4 6	4 6	5	4 5	4 5 6	4 6	5 6	5 6
7 8	7 8 9	7 8 9	7 8 9	7 8 9	9	7 8	7 8 9	9
1 3	1	3	3	3	3	1	1 3	
4 5	4 6	4 6	5	4 5	4 5 6	4 6	5 6	2
7 8	7 8 9	7 8 9	7 8 9	7 8 9	9	7 8	7 8 9	
2 3	2	3	2 3		1	4 6	5 6	3
4 5	4 6	4 6	5	4 5		7 8	7 8 9	5 6
7 8	7 8 9	7 8 9	7 8 9	7 8 9				9
2	2	6	1	5	3	5	2	4
7 8	7 8 9			8 9		9	7	5 6
2	2		5	6	1	7	3	1 2
4	4	8 9			4	8 9		1
4	3	4 6	1	2	4 5	1	6	1 5 6
7		7 9	5 9		9	7	7 9	8
1 3	1		1 3	6	3	5	1 3	1 3
4	4		7 9			9	8	
7 8	7 8		1 2 3	1			1 2 3	
6	5	3	7	8	9	1 2 3		4
9	1	3	4	1	2 3	1 2	1 2 3	7
	8	8		5	5	8 6	8	

如图所示，观察第一大行（**r123**），提示数非常少，这甚至让我们开始怀疑，这样的数独题还能保证唯一解吗？题是唯一解的，这请你放心。不过我们就是需要利用这一点，来得到填数。

1 2 3	1 2	3	2 3	4 5	2 3	1	1 3	1 3
4 5	4 6	4 6	5	4 5	4 5 6	4 6	5 6	5 6
7 8	7 8 9	7 8 9	7 8 9	7 8 9	9	7 8	7 8 9	9
1 3	1	3	3	3	3	1	1 3	
4 5	4 6	4 6	5	4 5	4 5 6	4 6	5 6	2
7 8	7 8 9	7 8 9	7 8 9	7 8 9	9	7 8	7 8 9	
2 3	2	3	2 3		1	4 6	5 6	3
4 5	4 6	4 6	5	4 5		7 8	7 8 9	5 6
7 8	7 8 9	7 8 9	7 8 9	7 8 9				9
2	2	6	1	5	3	5	2	4
7 8	7 8 9			8 9		9	7	5 6
2	2		5	6	1	7	3	1 2
4	4	8 9			4	8 9		1
4	3	4 6	1	2	4 5	1	6	1 5 6
7		7 9	5 9		9	7	7 9	8
1 3	1		1 3	6	3	5	1 3	1 3
4	4		7 9			9	8	
7 8	7 8		1 2 3	1			1 2 3	
6	5	3	7	8	9	1 2 3		4
9	1	3	4	1	2 3	1 2	1 2 3	7
	8	8		5	5	8 6	8	

我们先假定下方六个宫 **b456789** 已经通过普通逻辑全部填满了数字。如果题目唯一解，上面的 **b123** 则应是可以继续推导得到填数的。不过 **b123** 内只有两个提示数，明显无法直接得到填数，不管是什么技巧，都是一样。

我们从唯一余数下手来思考这个问题。我们发现，**b123** 因为只有两个提示数，所以如果要唯一余数的话，至少要 8 个完全不同的数字，才能出一个数字。我们假定下面六个宫全部已经填好了数字，那么，如果可以出唯一余数，此时能出唯一余数的地方，只有 **r3c789**。为什么呢？下面有 6 个数可以提供唯一余数的数数操作，而 **r3c789** 恰好又可以数出两个数来，这样就可以有 8 个数字，就可以产生唯一余数解法，得到这三格的填数。但仔细思考

来看，**r3c78** 其实是不行的。为什么呢？根据 **r2c9(2)**，可以得到 **b69** 里填入 2 的位置，只能在 **r456789c78** (**c78b69**，涂橙色的区域)。我们就针对 **r3c78** 的其中一格来说，比如这里的 **r3c7**，能够影响这里候选数排除的数字有 **r3c6**、**r2c9** 和 **r456789c7**，共计 8 个数。可是 **r456789c7** 里有一个 2，而 **r2c9** 也是 2，这样就带来了一个很遗憾的结果：有两个 2，所以有效的信息最多只有 7 个，明显是无法唯一余数出数的。

同理，**r3c8** 也是一样的道理。所以其实，实际上只有 **r3c9** 可以通过如此的逻辑出数（当然前提还是最开始说到的，下面的六个宫全部已经填好了数字，而且必须是完全不同的 8 个数字，才有办法唯一余数）。

那么，此时我们观察 **c9**，**c9** 此时有一格可以出数，是 **r1c9 = 1**。为什么呢？如果 **r456789c9** 里有一格是数字 1 的话，这样就会和 **r3c6(1)** 是一样的数字，**r3c9** 此时就不能通过唯一余数得到出数；此时就没有位置可以出数了。这是因为，由于 **r3c78** 不能出数，**r3c9** 如果还不能出数的话，那就真的没办法再找到位置出数了，此时这一行就会产生多种填法，毕竟现在已经没有其他的提示信息了，没办法确定接下来填哪里，或者其他的逻辑删数。所以不能让 **r456789c9** 里有数字 1；显然，**r3c9** 也不能是 1 (**r3c6(1)** 的排除)，所以，**c9** 里只有 **r1c9** 可以填入数字 1，所以 **r1c9 = 1**。

回到最初的逻辑，同样我们可以通过观察 **r2**，此时可以出得到 **r1c6 = 2** 的结果，这里的逻辑是类似的。

1 2 3	1 2	3	2 3		2 3	1	1 3	1 3
4 5	4 6	4 6	5	4 5	4 5 6	4 6	5 6	5 6
7 8	7 8 9	7 8 9	7 8 9	7 8 9	9	7 8	7 8 9	9
1 3	1	3	3	3	3	1	1 3	
4 5	4 6	4 6	5	4 5	4 5 6	4 6	5 6	2
7 8	7 8 9	7 8 9	7 8 9	7 8 9	9	7 8	7 8 9	
2 3	2	3	2 3		1		3	3
4 5	4 6	4 6	5	4 5		4 6	5 6	5 6
7 8	7 8 9	7 8 9	7 8 9	7 8 9		7 8	7 8 9	9
2	2	6	1	5	3	5	2	4
7 8	7 8 9			8 9		9	7	5 6
2	2		5	6	1		1 2	1
4	4		8 9	4	8 9	7	3	9
			1	5	2	4 5	1	5 6
4 7	3	4 6	5	2	9	7	6	7 9
1 3	1		1 3	6		3	1 3	1 3
4	4		7 9			9	8	
7 8	7 8		1 2 3	1			1 2 3	
6	5	3	7	7	7	8	9	4
9	1	3	4	1	5	2 3	1 2	1 2 3
	8	8				5	6	6
						8	8	7

逻辑大致和上方的逻辑同样，只是换了下坐标，其他的推导思维都不变。

注意，这两个填数是互不冲突的，是独立分开的两个结论，也就是说，**r1c6 = 2** 和 **r1c9 = 1** 的出数结论是不讲先后顺序的。得到这两个数之后，余盘瓦解。

1-7-2-2 原理剖析

这种神乎其技的技巧，不免会让人觉得比较痛苦，因为它的填数模式看起来似乎跟致命结构完全没有关系，但实际上，你可以感觉到，上述的逻辑里有一些瑕疵。比如，为什么我们非得使用唯一余数技巧不可呢？如果使用其它的技巧，比如链，难道不行吗？

答案是否定的。这里只能用唯一余数。排除不行，链等等技巧更不行。排除是显然不够

数字的，一个合格的排除法，针对于 **b123** 来说，如果像是 **b1** 的话，就至少需要 4 个完全一样的数字作排除，才可能得到唯一的填数位置。然而 **r2c9** 和 **r3c6** 是不同的数字，下面最多只能保证两个数字，上面也只能保证其中一个数字作排除，所以只有三个数字，完全不可能得到填数结论。

b23 为什么也不行呢？就拿 **b2** 来说，要出一个数字，需要至少 3 个数字才能“卡死”一个数字的填数位置。而下面的数字 **1** 或 **2** 的填数位置是我们无法直接确定的，换言之，如果看数字 **2** 的话，能够使用排除操作，并得到填数的位置只可能是 **r1c6**，并且需要 3 个数字 **2** 共同对 **b2** 排除，才可能出这个数。然而，你怎么知道下方数字 **2** 的位置不能在 **r456789c6** 里？如果在这里没，情况就更加糟糕，3 个数字 **2** 都不能保证出数了，然而就需要至少 4 个 **2**，就回到了 **b1** 的排除逻辑上（数字 **2** 不够那么多，显然是不行的），所以 **b2** 是无法排除的。**b3** 也是同理的。

所以排除操作是无法得到填数结果的。那链为什么也不行呢？链针对于这样提示数少的地方，是无法起到重要作用的，链的结果只能是删数，而由于 **b123** 提示数非常少的缘故，可以利用的强弱关系也会非常少（可以说甚至没有），根本无法指望用链来得到删数。

其他技巧肯定不行的，这里就不用多说了，毕竟，除了唯一性技巧（致命结构）以外，都能转为强弱关系表示的画法。链都不行，更别说其他的结构了。

Part 2 复杂鱼/链列 (Complex Fish)

在前面的基础部分里我们介绍了鱼的基本使用和定义，不过可以看到的是，这种例子结构非常大，导致可能出现的频率不高，所以不怎么实用；其次是结构不好推理，虽说它和数组是同构的，性质和数组完全一致，除了表达上不一致以外。

这一个部分我们将为你介绍的是鱼的完整体系，这个体系将鱼的内容完整地推广到任意同数情况，所以我们经常可以听到一些小伙伴的评价：“鱼 = 所有同数技巧”。

不过在进入正文之前，我们需要先介绍一下后续经常用到的鱼的基本定义以及鱼的删数原则，在介绍完毕之后，我们才会正式地进入鱼的篇章。

2-1 鱼的原理

首先，我们再次回顾一下鱼的一些常见术语词汇。

- **定义域**（或**基准域**，**Defining Set** 或 **Base Set**）：定义鱼结构推导的区域集合。
- **删除域**（或**删数域**、**覆盖域**，**Secondary Set** 或 **Cover Set**）：可以提供删除鱼结构的区域的集合。

那么，我们可以这么去思考一个鱼结构的删数逻辑，例如下图。

5	¹ ₇	⁴ ₇	6	^{1 3} _{4 4}	^{1 3} _{4 4}	2	9	8
3	2	^{4 6} _{7 9}	¹ _{4 5 9}	¹ _{4 5 9}	8	¹ _{4 6 7}	^{4 6} ₇	^{4 6} ₇
¹ ₆	¹ ₉	8	7	^{1 2} _{4 9}	^{1 2} _{4 9}	¹ _{4 6}	5	3
2	8	³ _{5 6 7}	^{1 3} _{4 5 7}	^{1 3} _{4 5 7}	9	³ _{4 6 7}	^{4 6} ₇	^{4 6} ₇
9	4	³ _{7 6}	³ ₇	^{2 3} _{7 6}	^{2 3} _{7 6}	8	1	5
¹ ₇	¹ _{6 7}	⁵ _{7 6}	8	^{1 3} _{4 5 7}	^{1 3} _{4 6 7}	³ _{4 6 7}	2	9
8	6	1	³ _{4 9}	³ _{4 9}	7	5	³ ₄	2
⁷ _{7 9}	⁵ _{7 9}	⁵ _{7 9}	2	³ _{4 8 9}	³ _{4 9}	³ _{4 9}	6	1
4	3	2	¹ ₉	6	5	⁹ ₉	8	7

如图所示，我们回顾一下之前的推导逻辑，我们可以发现，r368 这三行的 4 的填数恰在同样的三列里，这使得我们无论如何放置数字 4 的位置，但由于 r368 里每一行都必须放一个 4 的缘故，使得三行肯定会放下三处 4，但它们被框在同样的 c567 三列里，所以恰好我们安放 4 的位置就只能在 c567 的不同行列的三处，但也都肯定可以保证，c567 里，每一列都会出现一个 4 在 r368c567 的其中三个单元格里。

所以，c567 既然可以保证每一列都会有 4 在 r368c567 里，使得其余位置都不能放

可以从刚才的示例的逻辑里精简出来，定义域的每一个区域必须保证填数“必须填入一个”；而删除域则保证的是“最多有一处可填入”。

再来回顾这个例子，定义域 **r368** 即保证 **r3**、**r6** 和 **r8** 每一个区域都必须放下一个 **4** 在 **r3c567**、**r6c567** 和 **r8c567** 里；删除域 **c567** 则保证 **c5**、**c6** 和 **c7** 每一个区域最多都只有一个 **4** 要填入到 **r368c5**、**r368c6** 和 **r368c7** 里。相信你应该能够明白这一点了。

显然，我们找到了三处删除域和三处定义域。鱼要求，**如果鱼的定义域和删除域的区域数相等，并且删除域给定的所有区域都能够完整包含到定义域能放下数字的所有位置的话，那么删除域的非涉及单元格都可以删除。**这句话有些绕，我们来理解一下这句话的逻辑。

那么，鱼的整体内容就讲到这里。我们现在来看鱼的正文内容。

2-2-1 宫内鱼的形成

[illegible]

390

候选数。

我们移动其中一列，使得和右侧的一列拼起来，所以结构变为右图这样。当然，逻辑还是原本的三链列的逻辑。

现在，我们开始变换结构的形状。我们现在尝试把其中的 **r2** 变换到 **b3** 里，这一点很奇怪，但也很特殊。变换后，结果变为下图这样。

						x	x	/
						x	x	/
						x	x	/
/	x	/	/	/	/	x	x	/
/	x	/	/	/	/	x	x	/

如图所示。如果结构变为这样，还能推理和执行删数逻辑吗？我们尝试使用之前我们得到的逻辑进行解释。首先，定义域区域数量为 **3: r58b3**，这保证了这三个区域必须填入三个 **x** 到里面去；而删除域是我们规定的，也恰好是三个区域，而这三个区域都完全恰好包括了所有我们原本定义域里圈出来的所有 **x**。我们知道，从理论上可以明白一点，既然删除域能完整容纳下定义域原本所有的候选数 **x**，而且定义域区域数和删除域区域数一样，那不论怎么放置这些 **x** 的位置，由于不能违背数独规则，即行列宫内不含重复数字，那么 **x** 的位置还要在每一个定义域里恰好放一个，那显然每一个删除域里，就必须每一个区域都含有一个 **x**。否则，候选数 **x** 必定会在某一个定义域区域里出现两次。毕竟我们只能填入这么多 **x**，如果少一个删除域区域不出现 **x**，那就必然会让定义域区域多一个位置填 **x**，而定义域区域数只有那么多，所以多余的填数位置必然就会挤到同一个区域里，导致违背数独规则。所以这个结构的删数，删除域的每一个区域的其余位置（涂色单元格并不含任何符号的位置）就是删数的区域了。

从理论上听起来很空泛，所以我们针对于实际情况来对这个结构作出分析。首先，我们先来按照特殊的定义域区域 **b3** 来讨论填数，我们把它划分为 **r123c7(x)**和 **r123c8(x)**两部分。首先，如果 **r123c7** 里有一个单元格填 **x**，则由于 **r123c7(x)**区块的影响，此时在 **r58** 里必然会产生一个二链列结构在 **r58c28**。**r58c28** 显然是可以被删除 **c28** 的，因为此时它是一个完美的二链列结构。而由于假设 **r123c7(x)**区块的成立，此时 **c7** 是由于这个区块得到 **x** 的删数的，所以我们可以把 **c7** 算上，所以删除域是 **c278**；接着，如果假设 **r123c8(x)**区块成立，则下面的 **r58** 依然会产生一个完美的二链列结构，删除域此时是 **c27**，但由于 **r123c8(x)**区块的成立，所以 **c8** 其余位置也是不能放下 **x** 的，所以我们把 **c8** 也算上，整体的结构删除的地方还是 **c278**。

所以，两种情况肯定会至少有一种是对的，但它们都可以删除 c278 的其余位置的 x，所以我们完全可以认为，这个结构整体是可以删除 c278 其余位置的 x 的，或者换句话说，把它看作一种新的鱼结构，而定义域是 r58b3，删除域则是 c278。

我们把这种定义域区域里同时包含行宫或列宫组合的，或者是删除域区域里同时包含行宫或列宫组合的鱼结构称为**宫内鱼**（或**宫内链列**，**Franken Fish**）。比如这个例子里，定义域含有 r58b3，是同时含有行和宫两种区域类型的，所以称为宫内鱼。

2-2-2 原理进一步剖析

因为宫内鱼的结构有一些奇怪，而且会有区域被压缩到宫内，所以讨论起来就会有一些地方比较麻烦。所以我们进一步来分析这种结构到底可以如何变换。

2-2-2-1 宫内鱼的残缺情况

						/	x	/
						x	/	/
						/	/	/
/	x	/	/	/	/	x	x	/
/	x	/	/	/	/	x	x	/

如图所示，这个结构是否是一个正确的宫内鱼结构呢？答案是肯定的。从理论的角度来说，删除掉原本宫内鱼完整版的一部分候选数是不影响鱼的使用的，即使它们会变得各种奇形怪状，这是因为，删除的部分并不会影响到定义域区域数和删除域区域数相同的这一个特征。而我们之前从理论上和实际上都详细阐述到了“只要它们的数量相同，并且删除域区域包含所有定义域区域里圈出来的所有候选数 x，就一定是可以删数的”这一个看起来似乎很棒也很神奇的结论。

而从实际上来说，想要说明这一点也很简单，我们依然通过 b3 来分情况讨论。只是这一次，b3 结构变得更简单了，这使得 b3 本身该有的两个区块都直接变为了两个单独的候选数。但实际上，它们的排除效果是不受影响的，所以实际上删数依然可以得到保证。

之前，我们学到了退化鱼结构，这种鱼结构甚至只能依赖于鳍才能存在。我们先不考虑鱼鳍的事情，我们尝试在这个残缺得看似无法再残缺的结构再删除一些 x。如果 r5c7(x) 和 r8c8(x) 此时也一并消失的话，如果看这个结构，实际上是可以存在的，只是巧妙的是，此时的每一个区域都仅剩下了两处可以填入 x 的位置，所以这样它们是可以构成一个环结构的，删数也就恰好是删除域所在的这三个区域。

2-2-2-2 宫内二链列 (Franken X-Wing) ?

现在我们来考虑一下,二链列是否存在对应的宫内二链列形式。我们照着之前的逻辑,依葫芦画瓢看看。

							x	x	/
/	x	/	/	/	/	/	x	x	/
/	x	/	/	/	/	/	x	x	/
/	x	/	/	/	/	/	/	/	/
/	x	/	/	/	/	/	/	/	/
/	x	/	/	/	/	/	/	/	/

如左图所示，结构是这样的，我们尝试把结构的 **x** 移动到 **r28c78** 里，然后把 **r2** 转换到 **b3** 内，就可以变为右图这样。

但实际上，在中途的转换时就可以发现，它已经变为了一个级联区块结构，所以宫内二链列实际上已经脱离了实际的应用，所以它只存在于理论之中。

当然，我们继续观察右图，即使变为了真正的宫内二链列，也会发现，实际上它等效于 **c9** 和 **r8** 的两个关于 **x** 的区块结构，删数是都可以删掉的，因此依然不存在于实际的运用里。当然，实际上它也可以像是一般的宫内鱼那样，变为残缺的情况，比如下面这样。

					/	x	/
					x	/	/
					/	/	/
/	/	/	/	/	x	x	/

那么它是环，是宫内二链列，还是级联区块呢？这就你自己来分析它了。

2-2-2-3 宫内四链列的结构

实际上，四链列因为比三链列更大，所以肯定会存在宫内的版本。但是宫内四链列比较特殊的地方在于，它具有四个定义域区域，这样就可以有更多的区域转变到宫内，例如我们可以把一个行改为宫，或者同时把两个行改为宫，例如下面的这两种不同的情况。

						x	x	/	/	x	x					x	x	/
						x	x	/	/	x	x					x	x	/
						x	x	/	/	x	x					x	x	/
/	x	/	x	/	/	x	x	/										
/	x	/	x	/	/	x	x	/	/	x	x	/	/	/	/	x	x	/
/	x	/	x	/	/	x	x	/	/	x	x	/	/	/	/	x	x	/

至于如何推理变为这样，我就不再过多阐述了。

2-2-2-4 定义域能重叠吗？

我们还有一种构型依然可能出现在盘面里。由于宫内鱼的结构有宫的区域，这就导致了可能出现行或宫重叠的现象。

/	x	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/
/	x	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/

如图所示，这个结构就是我刚才说的重叠的情况。显然，这种结构里把原本 b3 的宫移动到了 b6，但是其它的并未发生变化。

可是，仔细观察就可以发现，r5 已经产生了出数的结论，所以这样的结构在三阶的情况下只有理论上是存在的。另外，这里也会产生一个问题。如果区域重叠了，那么 r5c78

如果也含有 x ，那应该怎么考虑逻辑呢？毕竟当 **r5c78** 其一填入 x 后，这将同时使得两个定义域区域同时拥有填数，而我们却只填入了一个数就满足了两个定义域区域的要求。这种现象，我们将在后面的内容讨论到，我们大可直接将这个重叠的部分（即这里的 **r5c78**）视为“不能放候选数 x ”，即图上画的“/”符号。

从理论上说明，我们可以发现，此处的结构依然存在三个定义域区域和删除域区域，也确实满足删除域区域全部覆盖了所有定义域里的 x 。那么这就保证了， x 在定义域里只能放 3 个，而且还要想行列宫不产生违背数独规则的情况，那么只能在每一个删除域区域上也都要保证一个 x 的出现，而这一点内容已经在前文叙述过，这里就不作出说明了。

可以类比一下这个情况，我们将结构提升到四阶，看看宫内四链列是否存在重叠区域的现象。实际上，宫内四链列确实存在重叠的现象。例如下面的这种情况。

/	/	x	x	/	/	x	x	/
						x	x	/
						x	x	/
/	/	x	x	/	/	/	/	/
/	/	x	x	/	/	x	x	/

如图所示，这种结构就是典型成立的，首先定义域区域 **r6** 和 **b6** 有重叠部分，而重叠部分我们目前要求的是不能有 x 。而这样一来，就不存在之前重叠区域里产生的“填入一个 x 同时满足两个区域”的特殊情况，就不必过多去探讨。并且这样放置的话，定义域区域保证要放下 4 个 x ，删除域区域也是 4 个，这样就满足了删数的原理，所以删除域依然是可以成立的。

不过，这个示例如果从假设层面，就显得很麻烦了，而理论是可以保证正确性的，所以我们就不再探讨它的实际推导为何正确了。

2-2-2-5 什么？鱼也可以转置？

别着急，在进入示例讲解之前还要看一种情况。

						x	x	/		/					x	x	
						x	x	/		/					x	x	
						x	x	/		/					x	x	
										/					/	/	
/	x	/	/	/	/	x	x	/		x					x	x	
										/					/	/	
										/					/	/	
/	x	/	/	/	/	x	x	/		x					x	x	
										/					/	/	

仔细观察两个鱼图，可以发现它们涉及的单元格都是一样的，但第一个的定义域变为了第二个的删除域，而第一个的删除域就变为了第二个的定义域。那么，变换到后面的这个情况，那么它是否成立，并且删数可以正常删除呢？

实际上，这种结构依然是可以的。从理论上说明，这样的行为并没有引起定义域区域数和删除域区域数的改变，也没有引起“全覆盖”要求的改变，所以删数依旧是成立的；而从实际逻辑推理上来理解，你可以考虑随意放置 x，最后必然在 c278 里每一列都能找到一处 x 的放置，分属于 r58b3 的每一个区域。

将定义域和删除域交换的行为，在鱼结构里称为**转置**（Transpose）。当然，这只在理论上才会出现，在实际解题的结构之中，结构是无法转置的，它只有互补，这一点我们在后面的内容会作出详细的说明。

实际上，这种结构依然叫做宫内鱼，因为在宫内鱼的定义里保证了，定义域或者删除域的其中一个定义里出现行宫和列宫组合就满足了要求。这就是为什么，宫内鱼的定义必须说明必须定义域或删除域的其一，而并不是单纯只有定义域的原因。

那么，讲到了这么多的构型，现在我们来看一些示例。

2-2-3 一些宫内鱼的示例

现在来看一些有关宫内鱼技巧的例子，来熟悉一下宫内鱼的使用。

2-2-3-1 宫内三链列（Franken Swordfish）

下面来看若干**宫内三链列（Franken Swordfish）**技巧的例子。

1 4 7	8	9	3 6	1 4 7	1 4 7	5	2
1 4 5 7	1 4 5	4 7	8	1 4 7	2	6	3 9
2	3	6	1 4 7	9	5	8	1 4 7
3	6	4 7	1 2 4 7	5	8	9	1 2 4 7
1 4 5 7	1 4 5	8	1 2 4 7	1 4 7	3	1 2 4 7	6
1 4 7	1 4	2	6	3	1 4 7	8	5
6	4 9	3	1 4 7	2	1 4 7	5	8
4 9	2	1	5	8	6	4 7	3
8	7	5	9	1 4	3	2	6 1 4

如图所示，可以看到这个结构的定义域是 **r34b9**，删除域是 **c489**。显然，定义域区域数和删除域区域数一样多，而且删除域全部包含了定义域里的所有 **1**。这样一来，放下 **1** 的位置不论怎么安排，每一个删除域区域都能保证有一个 **1** 的出现。所以所有删除域的其余位置的 **1** 都可以被删除。

5	4	9	2 3 6	7	2 3 6	8	1	2 3 6
2 3 6	1	3	1 2 3 6	1	5	9	4	2 3 6
2 3 6	1	3	9	4	1 2 3 6	2	3	5
1	2	4 5	4 7 8	3	7 8 9	4 5 6	6	4 7 8 9
3	8	6	1 4	5	1	7	2	3
7	5	3 4 5	4 8	2	6 8 9	4 5 6	3	1
8	5	3	1 2 3 6	1	4	1 2	6	2 3 6
3	6	2	5	1	1 3 7 8	1 4	7 8 9	4 7 8 9
4	7	1	2 8	9	2 8	3	5	2 8

接下来是第二则示例。这个结构的定义域为 **r19b5**，删除域则是 **c469**。和刚才一样，定义域区域数和删除域区域数一样多，都是 **3**，而且删除域区域完全覆盖了所有定义域涉及的 **6**，所以删除域的每一个其余位置的 **6** 都可以被删除。

2 8	2 8	1 4	3	7	1 4	5	6	9	1 3	7	8	9	2	4	6	1 3	5		
5	4	6	9	8	2	4	6	1	3	7	1 2 3	5	1 2	6	1 3	1	9	4	
1 3	1 3	6	7	9	1 5	1 5	6 4	2 2	2 4	2 4	9	4	1 6	5	1 6	2	3 3		
4	2 3	5	8	6	5 3	9	2 3	2 5	1	7	8	6	9	7	3	5	4	1 2	1 2
1 2 3	7	6	1 4	5	1 3	1 8	9	2 4	5 8	2 3	4	5	9	4	1	2	3		
9	1 5	1 3	2	1 4	5 3	7	6	4 5	8	5	2	3	4	8	1	7	6	9	
1 8	9	2	7	1 4	5 8	3	4 8	1 4	5 8	6	6	8 9	4	2	1 7	5	1 3	1 3	
7	1 4	5 8	1 3	1 8	6	1 2	2 3	4 8	9	2 3	1 2	3	1	5	7 8 9	4	6		
6	1 4	5 8	1 3	1 8	9	1 2	2 3	4 8	5	2 3	4	8 9	1	4	6	9	1 2	1 2	

如图所示，我想让你来理解这两个示例。这两则示例的逻辑和上两个示例的逻辑是完全一样的，所以我并不需要阐述什么东西，靠你自己了！

2-2-3-2 转置的宫内三链列

接着来看一则利用基本构型转置了的结构。

4 5 6	4 5 6	8	9	1	2	7	3	5 6	
2	5 9 7	3	6	5 8 7	5 8 7	1	4	5 8 9	
1 5 7	1 5 6 9	3 6 7 9	3 5 7 8	4	5 7 8	2 6	2 5 6 8	2 5 6 8 9	
9	1 5 6	4	2	5 6 3	5 6 3	8	7	1 6	
1 5 7	1 5 6 8	6 7	4	9	5 6 7 8	3	1 2 6	1 2 6	
7 8	6	3	2	7 8	7 8	1	5	9	4
4 3	4 6 8 9	6 9	5 7 8	2	5 6 7 8	4	6	1 5 8	1 5 6 8 9
4 8	7	1	5 8	5 8	6	9	2 4	2 5 8	3
3 6 8	2	5	1	3 7 8	4	9	6 8	7 8	6

如图所示，这个鱼结构的定义域是 r689，而删除域是 c15b9。

2-2-3-3 宫内四链列 (Franken Jellyfish)

宫内三链列已经够难找了，**宫内四链列 (Franken Jellyfish)** 的例子就更难找到了，不过实际上它还是广泛存在于题目里的，所以这里给出了一些情况。

7	6	2	³ ₉	1	5	8	³ ₉	4
4	1	9	³ ₈ 6	³ ₈	7	³ ₆	2	5
3	5	8	2	4	4	6	1	7
⁵ ₈ 6	2	³ ₅ 6	4	³ ₅ 8	³ ₉	7	³ ₅ 6	1
⁵ ₈ 6	9	4	³ ₅ 6	7	1	³ ₆	³ ₄ 5	2
1	7	4	³ ₅ 6	³ ₆ 5	2	9	4	8
⁵ ₆	8	1	⁵ ₆	7	4	³ ₁ 3	2	¹ ₆ 9
9	3	1	⁶ ₉	5	2	8	4	¹ ₆ 7
2	4	7	¹ ₉	6	¹ ₉	5	8	3

如图所示，这就是一个宫内四链列的示例。可以看到，这个结构的构型非常类似于之前说到的四阶情况的重叠构型。显然，这个结构此时是成立的，定义域区域和删除域区域一样多，都是 4 个，而且也满足了“全覆盖”的要求；并且，r6 和 b6 重叠的单元格上也都没有影响填数的额外的候选数 6，所以结构是成立的。

² ₅ 8	² ₄ 5	6	7	² ₄ 8	² ₄ 5	3	9	1
1	⁵ ₇ 8	9	3	1	4	5	6	2
3	¹ ₇ 2	2	1	¹ ₅ 6	² ₈ 9	5	4	7
¹ ₅ 6	¹ ₇ 2	2	⁵ ₈ 6	3	⁵ ₇ 9	² ₆ 8	4	1
⁵ ₆ 8	⁵ ₇ 9	7	2	⁴ ₆ 8	⁴ ₅ 6	⁶ ₇ 9	1	3
4	² ₅ 8	3	⁵ ₇ 9	1	² ₆ 8	² ₇ 9	5	6
² ₇ 9	3	1	4	² ₆ 8	8	⁴ ₉	7	5
² ₇ 9	² ₇ 9	4	9	5	3	1	² ₈	6
² ₇ 9	6	5	¹ ₄	¹ ₇ 2	2	² ₄ 8	3	⁸ ₉

如图所示，这是另外一则示例，它和上一个示例不同，它没有重叠，所以直接不用考虑重叠的那个额外条件。删数是完全成立的，这里就不再过多说明了。

那么，我们再给出四则示例，不过得你自己来思考了。不过它们的构型都差不多。

3	4	5	1	6	1	2	7	1	1	2	2	1	2	1	2	3	2	4
2	8	6	1	3	1	3	1	3	1	4	4	5	4	5	9	7	8	2
1	5	7	4	3	2	2	3	3	3	4	6	4	5	8	9	4	7	5
6		3	4	5	1	2	1	2	4	6	4	5	6	9	3	4	5	7
5	7	4	2	1	6	9	3	4	7	8	8	7	8	7	8	4	5	6
4	2	1	4	9	3	5	6	7	8	9	3	4	5	6	7	8	9	5
7	6	2	1	3	5	4	9	1	2	3	4	5	6	7	8	9	5	6
9	1	3	2	3	1	2	1	2	1	2	1	2	1	2	3	4	5	6
4	1	5	4	5	6	7	8	9	1	2	3	4	5	6	7	8	9	5
5	4	8	4	5	6	7	8	9	1	2	3	4	5	6	7	8	9	5

4	5	3	1	9	6	5	3	7	5	3	1	2	1	2	3	1	2	7
3	2	6	4	5	4	4	5	5	8	1	9	8	7	4	5	6	4	5
5	9	7	1	3	1	2	3	1	5	6	5	6	8	9	1	2	3	4
9	6	2	1	2	5	1	3	4	7	8	9	1	2	3	4	5	6	7
2	5	8	3	4	7	9	1	6	7	8	9	1	2	3	4	5	6	7
7	1	4	3	6	9	2	5	4	3	5	1	2	3	4	5	6	7	8
1	2	3	2	5	6	1	5	4	5	6	1	2	3	4	5	6	7	8
6	7	5	4	5	4	4	5	2	9	1	8	7	6	5	4	3	2	1
1	4	9	1	5	2	7	5	6	5	6	3	7	6	5	4	3	2	1

如图所示，这些示例都是之前我们提到过的构型。所以逻辑我们就完全不用再去说明它了。第一个和第二个都是重叠的情况，第三个是原结构的转置的情况，第四个则是带有两个宫的定义域的宫内四链列结构。

那么，有没有更高阶的宫内鱼结构呢？由于结构的特殊性，它的互补并不像之前数组和标准鱼构型那样，规格会发生变化，所以，它是可能出现五链列以及更高的情况的，不过例子就不好找了，所以这里就不再列举了，后面将会广泛出现这种情况的示例。

2-2-4 带鱼鳍的宫内鱼构型

接下来我们来探讨宫内鱼的鱼鳍的存在性和存在的类型。

						x	x	/								I	x	F
						x	x	/								x	I	F
						x	x	/								I	I	F
/	x	/	/	/	/	x	x	/	F	x	F	/	/	/		I	x	F
/	x	/	/	/	/	x	x	/	F	x	F	/	/	/		x	I	F

如左图所示，这是一个标准的宫内三链列的结构。思考一下标准鱼的鱼鳍的逻辑：标准鱼的逻辑里，我们必须找到找到一个鱼鳍，能够删数的前提是，这个鱼鳍必须挨着定义域里的某个候选数 x ，这样我们才能保证删数时，这个长在外面的鱼鳍，它所在的行列宫可以对应到删除域的某处上。

比如上面左边的这个鱼图里，首先我们肯定必须要保证的是，鱼鳍得长在“/”处的地方，因为这是鱼鳍的规定，鱼鳍必须应该能够影响结构的成立；其次，鱼鳍还得能够对应到删除域上的某处。所以， $r5c456$ 就不可能了，如果 $r5c456$ 某处可以是 x 的话，这样删除域的任意一个地方都无法对应到，所以导致无法继续推理执行删数逻辑；同理， $r8c456$ 也是一个道理。所以这个结构最终能够出现鱼鳍的地方就是右图里标注“F”字母的地方；当然，为了可以对照，图中标注了“I”字母，来表示这个单元格是可以提供残缺情况的（即标注了“I”的单元格可以没有 x 候选数，即可以改写为“/”符号）。

同样地，宫内四链列也都可以带有鱼鳍，只是逻辑和上述的逻辑是一样的，所以可以类比，就不作阐述了。

下面我们来看一些有关鱼鳍的宫内鱼的示例。

2-2-5 带鱼鳍的宫内鱼的示例

2-2-5-1 鳍宫内二链列 (Finned Franken X-Wing)

接下来我们来看一些关于带有鱼鳍的宫内鱼的示例。

5	¹	¹ 3	2	¹	6	³	7	4
	8	8		9		9		
4	6	9	8	7	4	1	2	5
2	¹	7		5	8			
	4	6						
1	1	1 2		6	1	2 3	4	9
	5	5						
	8	8						
1	1	1 2		8	1	2 3	1	3
	4	4 5 6						
7	3	1		2	1	5 6	1	8
		5 6						
	6	5 6	4	1	2	3		7
	8	8						
1		1		6	3	4	8	2
3	2	5 6	7	4 5	8	4 5 6	5 6	1

如图所示，我们把 r1 和 b8 的所有 9 圈出来。如果就这样去找删除域的话，就会发现，r1c7(9)和 r8c6(9)都是不好找删除域的地方。因为它们相对于 r1 和 b8 而言，都是“单独”出现的，也就是说，我们可以明显把 c5 勾出来，这样其中的三个 9 都可以被覆盖到，但 r1c7(9)和 r8c6(9)却被单独列出来，要增加删除域就必须得为它们各自单独增加一个删除域区域才行。不过这样，显然就变为了三个删除域区域，已经超过了两个删除域区域的要求。

所以我们可以采用一种方法，把其中一个 9 看作鱼鳍，然后另外一个看作鱼的一部分，这样的话，鱼鳍就不用单独作出一个删除域区域，因为它的推理不依赖于鱼本体的结构。我们在推理鱼鳍的时候，都是采用“如果鱼鳍为真，则删除鱼鳍所在的相关单元格的位置的这个数；而鱼鳍为假，则鱼结构成立”，但这种说法里，并未提到鱼鳍为真的时候，鱼到底是怎样的，所以我们不必去考虑它跟鱼本体结构的关系，所以也没必要单独分配一个删除域区域；而另外一个数就作为鱼的一部分，分配一个删除域区域即可。所以，如图所示的结构就是这样处理的：把 r8c6(9)看作鱼鳍，而 r1c7(9)看作鱼的一部分，这样的话，定义域和删除域的区域个数就相同了，而且鱼鳍也可以对应到删除域区域的 r8c7(9)上，所以 r8c7<> 9 就是这个例子的结论。

实际上，可以看到，这个实际就是一个同数的区块链结构。我们把删除域视作弱关系的区域，定义域看作强关系的区域，鱼鳍看作链头的话，就会发现它就是一个普通的链结构，所以一般来说，这样的结构我们都可以转为链来书写和观察。如果你觉得鳍宫内二链列 (Finned Franken X-Wing) 不容易看鱼的话，请你尝试使用链的视角来找它们。

2-2-5-2 鳍宫内三链列 (Finned Franken Swordfish)

3 6 4 6 9	5	7	4 3 1 9	1 4 9	8	2
7	8 1	4	6	2	4 9	3 5
3 4 9	2 3 2 3 9	4 5 9	3 4 5 8	1 5 8 9	1 4 9	7 6
5 3	9	4	6	7	5 3	2 1 8
2	3 7 7 8	1	8 3	4	6	5 9
5 8	1	6	5 9	2	5 8 9	3 4 7
6 8 9	2 3 6 8 9	2 3 4	1 2 3	7	5	2 9 4 3
4	2 3 5	2 3	8	9	5 3	7 6 1
1	2 3 5 7	2 3 7 9	2 3 4 5	3 4 5	6	8 2 9 4 3

如图所示，如果我们忽略 $r1c1(3)$ ，那么在 $r15b7$ 里的所有 3 将构成一个标准的宫内三链列的结构，删除域则是 $c235$ 。但是由于 $r1c1(3)$ 客观存在，我们只能删除掉删除域和 $r1c1(3)$ 的相关格的交集，即只有 $r23c23(3)$ 四处的候选数 3 可以删除（如果这些单元格里含有候选数 3 的话）。所以，就图上而言，结论是 $r3c23 \neq 3$ 。

1 4	6 2	7	3	8	5	9	1 4 6	1 6
3	1 4 5 6	1 4 5	7 9	1 6 9	1 6	2 5 8	4 5 6 8	2 5
9	8	1 5	2	1 6	4	1 5 6	1 5 6	7
7	1 4 5 6	1 2	4 5	4 5 3	8	1 2 3 5 6	9	1 2 3 5 6
1 4 5	1 4 5 6	8	4 5 7	2	3 6	1 5 6	1 5 6	1 5 6
5 6	3	2	1	5 6 7	6 9 7	2 5 6 7 8	5 6 8	4
8	1 4 5 7	1 4 5	6	1 4 5 7	1 3 7	1 3 5	2	9
1 4 5	1 4 5 7	1 4 5 9	4 5 7	1 4 5 7	2	1 3 5 6	1 3 5 6	8
2	1 5	6	8	1 5 3	9	4	7	1 5 3

如图所示，这一则示例的逻辑也是一样。假设 $r4c9(6)$ 为假，则 $c19b5$ 将形成一个宫内三链列。而 $r4c8(6)$ 的客观存在，导致删数范围缩小到 $r56c78(6)$ ，所以 $r56c78 \neq 6$ 。

4 5 6	4 5 6	1	4 5	3 1	1 3	7	9	2
7	4 5 8 9	2	4 5 8	1 4 5 9	6	1 4	3	1 4 5
3	4 5 9	4	2	1 4 5 9	7	6	1 4 5	8
4 5 6	4 5 6 7 9	3	4 5 6	8	1 4 5 9	1 4	2	1 4 7 9
2	4 9	8	4	3	7	1 4 9	5	1 4 9
4 5 6	1	4 7 9	4 5 6	2	4 5 9	3	8	4 7 9
1	4 7	5	9	3	2	8	4 7	6
8	2	4 7	1	6	4 5	9	4 5 7	3
9	3	6	7	4 5	8	2	1 4 5	1 4 5

如图所示，如果 r1c13 和 r3c3 的候选数 4 全部消失，则结构 r137 的所有 4 将构成一个宫内三链列，删除域则完全成立；但鱼鳍有三个，所以这个示例必须要找的是删除域和这三个鱼鳍都能删除掉的地方，即只有 r2c2(4)了，所以 r2c2 <> 4。

2-2-5-3 鳍宫内四链列（Finned Franken Jellyfish）

1 2	1	9	1	7	4 5	1 3	6	1 2 3
8	5 8		4 8			4 5 8		4
1 2	6	7	9	1 4 5 8	3	1 4 5 8	1 2	1 2 4
3	4	5 8	1	6	2	1 5 8	7	9
9	1 3 7	6	2	1 4	8	1 4 7	1 3	5
1	2	4	5	3	9	1	8	6
5	1 3 8		7	1 4	6	9	1 2 3 4	1 2 3
4	9	1	3	2	7	6	5	8
		3 5 8	6	9	1	2	4	3
6	5 7 8	2	4 8	4 5 8	4 5	1 3 7	9	1 3 7

如图所示，这个示例极其不好看，因为残缺的现象非常严重，所以例子也比较难理解，所以不要分心，看看下面的逻辑，仔细思考其中的说法。

不过不要紧，因为我们按照理论来分析，如果此时我们不看 r1c2(1)的话，定义域和删除域区域数量是一样的，都是 4 个，而且删除域也确实做到了全覆盖的要求，所以它是一个合格的宫内四链列结构。只是，当鱼鳍被删除后，结构的有一些删除域上就只有一处 1 可以填了，比如 c9。我们说过，结构一经成立的话，删除域的每一个区域里也都会含有一处位置填入候选数 x，但如果此时只有一处可以放，那它就应当就是 x 的填数位置。那么是不是说明 r9c9(1)在 c9 此时只有这一处可以填，就一定说明 r9c9 = 1 呢？当然不是了，

因为这是基于鱼鳍为假时候的推导。所以鱼鳍为真的时候，我们并未作出任何对于鱼结构内部的推导逻辑，而仅仅是用来删数了，所以 $r9c9$ 此时是不是 1，我们并不知道。

那么为什么不把 $r1c2(1)$ 看作结构的一部分，而非要当它为鱼鳍不可呢？原因是，如果把 $r1c2(1)$ 看成结构的一部分，这个结构就一定要多一个删除域来覆盖 $r1c2(1)$ ，我们要实现全覆盖的要求就必须保证每一个候选数都尽量覆盖完全。所以这个结构我们只能把它看作鱼鳍，这样，鱼鳍因为它不属于结构的一部分，我们在推理和判断剩余结构是否是宫内鱼的时候就不需要去考虑它了。

那么，这里就可以详细阐述一下 **Finned**（带鱼鳍的鱼）和 **Sashimi**（退化鱼）的用法区别了。在前文里，**Finned** 和 **Sashimi** 是很好区分的，比如在标准鱼里，只要结构必须依靠鱼鳍才能存在，去掉鱼鳍后结构立马被降解的，就是 **Sashimi** 形式，即退化鱼；而其它的情况均为 **Finned**。

而宫内鱼里结构就显得很复杂，所以有些时候总会混淆带鱼鳍的普通鱼和退化鱼，到底什么时候该叫什么。这里可以这么想这个问题。只要把鱼鳍去掉后，存在某一个定义域区域里只有一处位置可以放下 x 的，就叫退化鱼（**Sashimi**），其它的情况全部都是带鱼鳍的普通鱼结构（**Finned**），所以上述例子看似好像结构残缺过于严重，就会不由地去怀疑它是否是退化鱼，实际上，它是看的删除域只有一处位置，而并非定义域只有一处，所以它依然用 **Finned** 一词。

2-2-5-4 鳍宫内五链列（Finned Franken Squirmbag/Starfish）

之前说过，宫内鱼是存在五阶的形式的，现在我们就来看看这里的示例。

8	4	5	3	1	4	5	6	2	7	4	5	6	9
1	4	6	4	7	8	3	5	5	5	4	2	4	6
4	6	4	5	2	4	5	6	4	7	8	1	3	9
7	6	4	5	8	5	3	1	2	5	9	2	5	9
5	2	8	9	7	6	1	4	3	8	9	5	8	9
1	9	1	3	8	9	8	9	5	2	4	6	7	5
4	4	5	8	9	8	9	1	3	4	5	2	6	7
3	7	1	2	4	5	6	5	9	8	4	5	9	9
2	4	5	6	4	5	7	8	3	4	5	9	1	9

如图所示，如果将 $r6c4(5)$ 和 $r8c5(5)$ 视为不存在，那么剩下的部分就存在一个定义域为 $c2358b5$ ，删除域则是 $r13479$ 的**宫内五链列**（**Franken Squirmbag/Starfish**）。但由于鱼鳍的客观存在，所以我们只能删除掉删除域以及这两处 5 的相关格交集，所以只有 $r79c4(5)$ 可以删除。

接下来来看一则目前例子里最麻烦的一个示例。这个例子是一个**孪生宫内五链列**（**Siamese Finned Franken Squirmbag**）。如果你能明白这一则示例，就说明你对鱼的掌握已经能灵活运用了。

1	2	3		3	1	2	3		2	3		2		3	1	3		3		3
4	5	6		4		6			5				9	7	9	8	9		6	
7	8	9		8	9	7		9	7		9		9		7	8	9		9	
2	3			3		2	3		2	3		6	1			3		4	7	
5				8	9		9		5						8	9				
8	9								9											
1	3			3	1	3			3			4	8		1	3		5	2	
6				6					7	9					6					
7	9				9	7		9	7	9					9					
2	3			3	2	3		1	3	1				3	2	3		3		
6				6				4	6			6		4	5				5	
8	9				8	9		9	7	8	9		8	9	7		9		9	
3				1	5			4						2		3		3	6	
8	9							7	8	9		8	9		4	7	9	7		
2	3			7	4				3			5			3	2	3			
6								6				6			6			1	8	
9								9				9			9					
1	3			3	1	3		1	2			1	2			3		3		3
4				4					6					6		5	6			5
7	9				9	7		9	8	9		8	9		9	7	8	9	7	9
1	3			2	8			1				7	5			3		3		
								6							6			6	9	4
								9							9			9		
				5	6				3	4						2				1
7	9							8	9						7	8	9			

如图所示。我们细数所有 **r23568** 的 3，如果要定删除域，最少也得有 6 个才行。因为多出来的 **r23c23(3)**和 **r6c6(3)**必须单独为其增加一个宫内的删除域，才可以实现全覆盖的规则。

不过，显然这样定义域区域和删除域区域就不一样多了。我们就尝试去这么想。如果我们此时把 **r23c23(3)**看作鱼的一部分，而且我们此时把 **r6c6(3)**看作鱼鳍的话，鱼鳍因为它并不作为推导鱼结构内部逻辑的一部分，所以我们不必单独为其分配一个删除域，所以这样的话，定义域和删除域个数恰好此时就相同了，那么删除域的所有位置都可以删除；不过带有鱼鳍，所以删除域内的删数还要进一步看是否和鱼鳍能够对应上。所以最终的删数此时是 **r4c4(3)**，这个位置是鱼鳍和删除域都能对应的位置；

不过，我们现在切换视角，把 **r23c23(3)**视作鱼鳍，而 **r6c6(3)**看作鱼的一部分，此时因为换过来看并不会影响总体结构的删除域区域个数，所以删除域依旧是 5 个，鱼此时依然成立，不过删数就变为了删除域和 **r23c23(3)**能够对应到的位置，所以此时的删数只有 **r1c1(3)**。

可以看到，两则观察视角是同时推理的，并不分先后顺序，所以两个删数是可以同时得到删除的，那么 **r1c1(3)**和 **r6c6(3)**就是这个示例的结论。这个结构和之前我们讲到的孪生鱼非常接近，因为它也是采用切换视角，大部分数值都是作为鱼的一部分看待，而少部分作为鱼鳍进行视角的切换，并对应不同的删数。

2-2-6 鱼鳍的定义要被拓展了?! ——内鳍的形成

相信，你看到这个标题的时候，一定是一头雾水的，不过，请你回顾一下宫内鱼有一种特殊情况，即定义域区域存在重叠的时候。

这种时候我们当时告诉大家的是，鱼的结构不要覆盖到两个定义域区域重叠的地方，这样讨论会比较复杂，因为重叠的地方，只需要填入一次数字，就同时满足了两个定义域区域，所以是非常难搞定的一件事。鱼的定义域区域数和删除域区域数要一致，是为了后面能够更好地推导鱼结构，能够快速经过这个通过理论证明过的结论，来得到全部删除域的能够删除的数。但重叠区域不同的地方在于，它会影响到的是两个定义域区域，这便影响到了定义域区域数和删除域区域数相同，并且全覆盖就可以删数的说法。

通过理论证明，我们保证了每一个候选数都能不重叠，才可以得到删数的结论，我们就拿宫内四链列和重叠后依旧可以删数的宫内四链列作出一个对比。

						x	x	/									
						x	x	/	/	/	x	x	/	/	x	x	/
						x	x	/									
/	x	/	x	/	/	x	x	/							x	x	/
															x	x	/
/	x	/	x	/	/	x	x	/	/	/	x	x	/	/	/	/	/
/	x	/	x	/	/	x	x	/	/	/	x	x	/	/	x	x	/

如左图所示，它是一个标准的宫内四链列结构，而且每一处定义域区域都没有重叠的地方；而右图给出的，r6 和 b6 就产生了重叠。

思考一下。如果 r6 和 b6 重叠的其中 r6c78 两个单元格包含候选数 x 的话（即相当于把“/”符号替换为“x”），则鱼的删数逻辑还成立吗？显然就不能了。当我们不把 x 放到 r6c78 的时候，就等同于右图的情况，删数此时是成立的；但我们把 r6c78 里放一个 x 的话，这样 r6 和 b6 两个区域都有了 x，但我们只填了一个，那么，剩余的两个定义域区域只能填入两个 x，这样算起来一共只有三处 x，而涉及了四个定义域区域。而删除域一共有 4 个区域，我们只有三个 x，而且删除域区域没有重叠，虽然删除域区域是全覆盖了这些 x 的，但由于定义域里只有 3 个 x，那么删除域一共 4 个区域，怎么可能每一个区域都能“兼顾”到呢？所以，此时推导就不对了。

那么，我们应该如何去处理这种重叠部分，来清除掉它对我们结构本体的影响呢？鱼鳍就是一个很好的选择。鱼鳍就是这样的一种存在，它会直接影响鱼本体的存在和形成，而它又独立于鱼的本体推导。那么我们不妨把右图里的 r6c78(x) 看作鱼鳍。

当它们不存在的时候，删数必然是删除域上的所有位置的删数（当然这不包括构建鱼结构的候选数）；而当它们为真的时候，显然只能是找这个鱼鳍和删除域共同对应的部分。不过遗憾的是，r6c78(x) 是两个鱼鳍，而此时如果找和删除域的交集的话，显然是不能找到

的。对照图上而言，鱼鳍分属到删除域的其中两列上，它们是并行的关系，要同时能删除，显然是做不到的，所以此时这个结构是没有删数的。

那么，如何使得这个结构拥有删数呢？答案就是，**r6c78** 里只有一处是鱼鳍就可以了，而另外一个单元格并不含有这个候选数数值，即其中一个单元格画上“**x**”，而另外一个单元格画上“/”，这样就可以得到删数了。比如 **r6c7** 含有 **x**，那么删数就是 **c7** 的删除域区域的 **x**；而如果只有 **r6c8** 含有 **x**，那么删数就是 **c8** 的删除域区域的 **x**。这样我们就完全解决了结构无法删数和无法推导的问题。

接下来我们降阶，看一下宫内三链列重叠时，鱼鳍应当放哪里。

/	x	x	/	x	/	/	/	/
/	x	x						
/	x	x						
/	x	x	/	x	/	/	/	/

如图所示，这就是一个重叠了定义域区域的宫内三链列，重叠的其中 **r9c23** 含有 **x**。和刚才的思维逻辑一样，此时的鱼是无法删数的，因为重叠的地方有两处，我们不得不看作鱼鳍处理，但删数又无法直接对应到同一处或多处上，所以此时是不允许的。那么我们削减一个位置，比如我们尝试把 **r9c2(x)** 改为“/”，结构就可以删数了：删数是 **c3** 这个删除域区域的 **x**。

/	x	x	/	x	/	/	/	/
/	/	x						
/	/	/						
/	@	/	/	x	/	/	/	/

最后，我们来说一下定义。如果鱼鳍长在定义域区域的交集的，我们称为**内鱼鳍**（简称**内鳍**，**Endo Fin**）；而其余位置上的鱼鳍，此时我们约定称之为**外鱼鳍**（简称**外鳍**，**Exo Fin**）以作区分。它们实际上都属于鱼鳍的范畴；当然，至于如何对一个鱼结构命名，这一点我们后面再说，现在都直接视作鳍鱼处理就好。之所以叫内鳍，就是因为在全覆盖的时候，内鳍其实也是被覆盖了的，而外鳍必须单独分配一个删除域区域才可以覆盖。

在鱼图里，内鱼鳍我们使用“@”（at 符号）或“E”来表示，而外鱼鳍则使用“F”（有些时候也用“&”，即 and 符号）来表示。那么接下来我们就来看一些有关内鱼鳍的宫内鱼结构的示例。

2-2-7 内鳍宫内鱼的示例

2-2-7-1 内鳍宫内三链列（Finned Franken Swordfish）

	4 5	4 5 8	3	2	6 8 9	4 6 8	1	7
6	1 3	2	7	4	1 8	3 8	5	9
1 3	7	1 4 8	5	6 8 9	1 6 8 9	3 4 6 8		2
1 3	5 6	9	4	1 3 6	1 3 5 6	2 6 7		8
1	8	1 4 5 6 7	2	1 6 9	1 5 6 7 9	4 6 7 9	3	1 6
2	1 3 4 6	1 3 4 6	1 6 8 9	1 3 6 8 9	7	5 4 9	1 6	
5	1 2 3 6	1 3 8	8 9	7	1 3 8 9	2 8 9	6	4
7 8	2 6	6 7 8	6 8 9	5	4	1 7 8 9	2	3
4	9	1 3 6 7 8	1 6 8	1 3 6 8	2	7 8	7 8	5

如图所示，这个结构想必我们在刚才才推理过，所以删数比较容易确定。如果内鱼鳍不

成立，则宫内三链列是成立的（只是，在之前的内容里提到过，宫内三链列如果含有重叠的定义域的话，重叠部分不能有填数，就会使得结构直接降解。所以这种结构必须依赖于鱼鳍而存在）；而鱼鳍 $r9c3(3)$ 的存在只能使得删数确定在 $c3$ 的其余位置的 3 ，所以，这个例子的删数只有 $r3c3(3)$ 。

2-2-7-2 内鳍宫内四链列（Finned Franken Jellyfish）

4 8 9	4 7 8 9	6	3 9 7	3 5 7	5 7 9	2	1 4 8	
4 8 9	1	4 7 8	2	6 7 9		3 4 8		5
2 5	3	2 5	1	8	4	9	7	6
7	4 5 6 8	2 3 4 5 6 8	3 4 9	1 3 5 9	1 2 5 9	1 4 8	2 6 8 9	1 2 4 8 9
1	4 5 8	4 5 8	4 9 7	5 7 9	6	4 7 8	3 2 7 8 9	4 7 8 9
2 3 4 6	4 6	9	8	1 3 7	1 2 7	5	2 4 6	1 2 4 7
3 5 6 8 9	4 5 6 8 9	4 5 8	7	1 4 9	1 8 9	1 4 8	2 4 5 8 9	1 2 3 4 8 9
4 5 8 9	2	4 5 7 8	6	1 4 9	3	1 4 7 8	4 5 8 9	1 7 8 9
4 8 9	4 7 8 9	1	5	2		6	4 8 9	4 7 8 9

这个示例是一个比较标准的带内鳍的宫内四链列。如果 $r1c9(4)$ 为假，则宫内四链列就无需考虑重叠的部分，所以定义域区域数和删除域区域数相等，也全覆盖了，所以删数是完全成立的；不过由于内鳍的存在，此时的删数只能在 $c9$ ，所以 $r4578c9(4)$ 都是可以删除的。

我们再来看一则示例。

8	5	1 4 9	6	1 4 9	7	1 4	2	3
1 4 9	1 2 7	1 2 6 7 9	1 3 8	1 3 5 8 9	4 5 8	1 4 6 7 8	4 6 8	1 6 7
3	1 7	1 6 4 6	2	1 4 8	4 8	1 4 6 7 8	5	9
7	1 2 6	1 2 4 5 6	1 8	1 2 4 6 8	9	1 4 5 6 8	3	1 4 6
1 4 5 9	1 2 3 6	1 2 3 4 5 6	1 3 8	1 2 3 7 8	2 6 8	1 5 6 7 8 9	4 8	1 6 7
1 9	8	1 9	5	1 7	4 6	1 7 9	4 6	2
6	4	5	9	2 5	1	2 3 7	7	8
1 5 7	1 3 7	1 3 5 7	4 8	2 4 5 6 8	2 4 5 6 8	2 3 7	9	4 6
2	9	8	7	4 6	3	4 6	1	5

如图所示，这个示例和刚才的推导过程完全一样。如果鱼鳍 $r5c1(4)$ 不成立，则定义

域区域数和删除域区域数相同，都是 4 个，而且全覆盖，所以删数是成立的；而鱼鳍的客观存在影响了结构，导致删数只能在 r5 这个删除域上，所以删数只有 r5c568(4)。

2-2-7-3 内鳍宫内五链列 (Finned Franken Squirmbag/Starfish)

6		2	3	⁴ _{7 8}	9	1	5	⁴ _{7 8}
¹ ₇	3	¹ ₈	2	^{4 5} _{7 8}	⁵ ₈	² _{7 8 9}	⁴ _{7 8 9}	6
9	5	4	⁷ ₈	1	6	² _{7 8}	3	² _{7 8}
2	6		5		4	3		1
4	1		⁶ _{7 8 9}	2	3	5	⁶ _{7 8 9}	
5		3	1	⁶ _{7 8 9}	² _{7 8}	² _{7 8}	^{4 6} _{7 8 9}	² _{8 9}
3	⁷ ₉	⁷ ₉	4	⁶ _{7 8 9}	2	⁷ _{8 9}	1	5
8	4	5	⁷ ₉	3	1	6	² _{7 9}	² _{7 9}
¹ ₇		2	⁶ _{7 8 9}	^{5 6} _{7 8}	⁵ ₈	4		3

最后的这个例子比较复杂，因为它既有外鳍也有内鳍。

如图所示，定义域区域 c1 和 b7 含有重叠部分，所以 r9c1(7)我们考虑为内鱼鳍，而为了保证鱼能够更好地推理，我们又把 r1c5(7)视作外鱼鳍处理，这样的话，如果它们同时都不存在，则结构包含五个定义域区域和五个删除域区域，它们个数是相等的，而且也做到了全覆盖，所以删除域的删数全部可以删除；而鱼鳍的存在打破了删数规则，所以我们此时必须找的是删除域、外鱼鳍 r1c5(7)和内鱼鳍 r9c1(7)的共同对应的位置。

显然，我们为了分析更严谨，我们不得不分三种情况考虑。如果 r1c5(7)成立，可以删到 r9c5(7)；如果 r9c1(7)成立，则也可以删除 r9c5(7)；而且 r9c5(7)也在删除域上，所以 r9c5(7)就是这个鱼的删数结论。

那么，宫内鱼结构的逻辑就全部讲完了。接下来我们要看到的是，另外一种鱼结构的变体类型，它的形状就更加奇怪了，导致这种结构的名字的直译版本很“草率”地就叫做**变异鱼 (Mutant Fish)**，那么实际上这个结构叫什么？我们下一节继续。

2-3 交叉鱼 (Mutant Fish)

现在我们来学习一种新的鱼的结构，叫做**交叉鱼**（或**交叉链列**，**Mutant Fish**）。这种结构非常麻烦的地方是，它一点都不好看。

2-3-1 交叉鱼的形成

/	x	/	/	/	/	/	x	x	/	x	/	/	/	/	x	x
															x	
															x	
						/	x	x						/		
						/	x	x						/		
						/	x	x						/		
															x	
															x	
/	x	/	/	/	/	/	x	x	/	x	/	/	/	/	x	x

如左图所示，它是一个宫内鱼，我们把其中宫的定义域区域移动到了中间去。现在，我们把 **b6** 补到 **c7** 上。结构就变为了右图这样。注意到的是，此时的删除域也同时发生了变动。虽然删数的单元格没有发生变动，但是可以看到，由于改变了定义域区域的位置，如果此时还把 **c89** 作为删除域区域的话，这个结构的 **r456c89** 就删不了了。为什么呢？

我们可以尝试把右图看作一个同数的环结构，即

$$r23c7=r78c7-r9c89=r9c2-r1c2=r1c89-r23c7(x)$$

这个环结构显然是不可能删除 **r456c89** 的，因为环的性质只能删除弱关系的对应区域，而 **r456c89** 并未在弱关系的区域内。

当然了，你也可以使用最普通的假设，把 **c7** 拆分为两种情况：**r23c7(x)** 区块成立和 **r78c7(x)** 区块成立两种情况进行讨论，可不管怎么讨论，**c456c89(x)** 跟我们讨论的填数位置无关，或者并不能够在 **r456c89** 里找到任意一个单元格，是两个区块不论谁成立都可以删掉的，所以 **r456c89(x)** 是删不掉的。

不过，上述两种情况都是为了检验实际情况，我们依旧可以通过理论说明为什么删不掉。首先我们通过这个图示给出的样子找到了 3 个定义域区域和 3 个删除域区域，显然它们的个数是相同的，并且删除域也完整覆盖了所有定义域里所有 **x** 的可能填数位置，这说明删除域的删数应当都是成立的。不过请你注意的是，删除域区域此时只能包含 **b39**，如果选定为 **c89**，则我们立马就会发现，**r2378c7(x)** 并未覆盖到，此时又得考虑如何去调整删除域区域来覆盖它们，所以这样是不可以的。

那么，我们就称右图这样的结构为交叉鱼。从定义上讲，当定义域或删除域的其中一方同时包含行和列的组合的，就可以叫交叉鱼。比如上述的示例里，定义域是 **r19c7**，同时含有行和列，所以称为交叉鱼；当然，定义里也规定，删除域同时包含行和列的组合，也算作

是交叉鱼。那么，宫内鱼和交叉鱼有什么区别呢？宫内鱼只能有行宫和列宫的组合，而交叉鱼则是包含行列的组合。

最后，我们可以看出，图上规定的宫内鱼其实是完全可以转换为交叉鱼的视角的；而交叉鱼也可以转换宫内鱼的视角。它们实际上是可以互相转换的。但，这仅仅是这样的一些简单的情况，有很大一部分的交叉鱼都是无法转换回宫内鱼的；当然，宫内鱼也存在一些情况无法转换为交叉鱼的情况，这一点我们将在后续的示例里逐渐为大家展示出来；而且，交叉鱼涉及行和列，就必然会产生交叉，导致定义域必然重叠的现象，所以交叉处的单元格得视为内鱼鳍看待（当然，最好不要使得结构的该位置包含可能会影响推导的这个候选数）。

2-3-2 原理进一步剖析

2-3-2-1 交叉鱼的残缺和带鱼鳍的情况

宫内鱼我们分析过了残缺和带鱼鳍的情况，那么交叉鱼由于变换了某一定义域区域的位置，那么残缺情况又应当如何分析呢？而鱼鳍的位置呢？

我们拿出一个**交叉三链列**（**Mutant Swordfish**）作为示例给大家展示。

/	x	/	/	/	/	/	x	x
						x		
						x		
						/		
						/		
						/		
						x		
						x		
/	x	/	/	/	/	/	x	x

如图所示，这是一个刚才我们得到的交叉三链列，它的定义域是 **r19c7**，删除域则是 **c2b39**。我们为了让残缺的部分去掉后也能继续推导，而且不影响结构的成立，我们此时可以考虑 **b39** 里的区块。

我们分别去掉 **r1c89(x)**、**r23c7(x)**、**r78c7(x)**和 **r9c89(x)**区块各自的其中一个单元格，把它们画成“/”，再来看看，它是否成立呢？

/	x	/	/	/	/	x	/
						x	
						/	
						/	
						/	
						/	
						x	
/	x	/	/	/	/	/	x

可以看到，就算是变为了这样，结构依旧是可以保证成立的，因为定义域区域和删除域区域数始终没有发生变化，而且全覆盖依旧是做到了。当然，从实际上分析，比如我们使用环的视角，可以发现，它仅仅是修改了环内的区块节点，而变为了候选数节点，但是弱关系区域并未发生任何的变动，所以删数照样是成立的。

2-3-2-2 交叉鱼的鱼鳍

那交叉鱼会有鱼鳍吗？如果有，那鱼鳍的位置呢？实际上，交叉鱼的鱼鳍，不论是外鱼鳍还是内鱼鳍，都非常好分析，因为随便选取一个原来给的完整版的鱼图里，除了 **r19c456** 的其它任何画“/”的地方都可以作为鱼鳍，因为它们均可以对应到删数；而 **r19c7** 两处是定义域区域的交集处，它们可以视作内鳍，此时这个内鳍依旧可以对应到 **b3** 或者 **b9** 的删除域区域的，所以交叉鱼的内鳍并不像是宫内鱼那样“傲娇”，必须要有重叠才会有内鱼鳍，而且还要找交集。因为交叉鱼生来就一定会有行列交叉，所以内鳍一定是可能产生在交集处的，就根本不存在定义域区域不会重叠的情况。所以它的所有删数在图中都标注了出来。

F	x	F	/	/	/	F	I	x
						x		
						I		
						x		
						I		
F	x	F	/	/	/	F	x	I

当然，需要你注意到的地方是，r19c7 在图上标注的是 F，但是你应该知道，它是内鱼鳍。

2-3-2-3 交叉四链列 (Mutant Jellyfish)

实际上，交叉四链列的构型也很简单，只需要分析其中一个宫，并改写为行或列就可以了。比如下面的这个示例，改写起来其实非常简单。

						x	x	/							/
						x	x	/							/
						x	x	/							/
/	x	/	x	/	/	x	x	/	/	x	/	x	/	/	x
															x
/	x	/	x	/	/	x	x	/	/	x	/	x	/	/	x
															x
/	x	/	x	/	/	x	x	/	/	x	/	x	/	/	x
															x

左图是原本的带有一个宫的定义域区域的宫内四链列结构，而改写好的交叉四链列的情况如右图所示。可以看到，它和之前的变换逻辑完全一样，仅仅是移动了某一处宫，改写为了列。

当然，不仅仅带有一个宫的可以这么干，带有两个宫的结构依然可以这么干。

/	x	x				x	x	/	/						/
/	x	x				x	x	/	/						/
/	x	x				x	x	/	/						/
									x						x
/	x	x	/	/	/	x	x	/	/	x	x	/	/	/	x
									x						x
									x						x
/	x	x	/	/	/	x	x	/	/	x	x	/	/	/	x
									x						x

可以看到，其实依旧是为列这么简单。不过，删除域全部都变为了宫。

2-3-2-4 交叉二链列 (Mutant X-Wing) ?

实际上，**交叉二链列 (Mutant X-Wing)** 也是理论上存在而已，实际上并不存在。在之前的宫内二链列的内容里我们其实已经介绍了它的逻辑等效于区块，所以交叉二链列仅仅变了一个定义域区域而已，所以结构的删数实际上和宫内二链列是一样的，所以就不作出讨论

了。

						x	x	/							/
						x	x	/							/
						x	x	/							/
/	/	/	/	/	/	x	x	/	/	/	/	/	/	x	/
															/

实际上，就是两个区块，一个行区块，一个列区块。

2-3-2-5 同时涉及所有区域类型的交叉鱼

接下来我们再来看一种变种的交叉鱼结构，它的定义域恰好涉及一行一列和一个宫。

		/				x	/	/
		/				x	/	/
		/				/	x	x
		/						
		/						
		/						
x	x	/	/	/	/	x	/	/
		/						
		/						

如图所示，这种结构依旧属于交叉鱼，因为定义域确实同时含有了行和列的组合，而宫内鱼结构要求不能有行和列的组合。可以看到，这种结构比较神奇的地方在于，我们找到的三个定义域区域和三个删除域区域都是可行的，而且确实全覆盖了，所以删数也是都成立的，只是结构长相比比较奇怪罢了。

当然，如果你实在是不愿意使用理论说明，可以采用环结构视角来看。这个示例可以讲结构连起来形成一个同数带区块的连续环结构，不过意义和之前说到的标准版本的交叉鱼的构型是一样的，删数也是产生于弱关系上。

2-3-2-6 交叉鱼的转置

当然了，交叉鱼也是存在转置的，不过……

	x					/	x	x
	/					x	/	/
	/					x	/	/
	/							
	/							
	/							
	/					x	/	/
	/					x	/	/
	x					/	x	x

如图所示，这就是交叉鱼的转置后的样子，我们尝试把它的定义域看作 **c2b39**，删除域则改为 **r19c7**。可以发现，从理论上就可以明白，定义域和删除域区域个数不会因为转置而发生改变，所以删数照样是成立的。

下面我们来看交叉鱼的基本示例。因为交叉鱼有一部分无法转换为宫内鱼视角，所以交叉鱼的示例就比较多了，而且还比较难。

2-3-3 交叉鱼的示例

2-3-3-1 交叉三链列（Mutant Swordfish）

	3	5	1	6	8	2	3	4	2	3
7	9	7				5	9	7	9	
4	3	8	4	5	1	2	3	1	2	3
7	9		7		9			4	9	7
4	3	2	6		7	4	9	8	5	1
	9									
6	4	8	1	2	3	5	1	2	3	5
			9	7	6	8	4	1	2	3
1	2	3	1	5			7	9	1	2
1	2	1		5	1	2	3	6	3	8
7	7		7		9		9			
5	9	4	2	3	8	1	2	7	1	3
8	1	3	2	3	1	2	1	2	1	3
		4			4		5	6	1	3
1	4		4	7	1	4	5	9	3	2
7		6			9			2	8	4

如图所示，这是一个**交叉三链列（Mutant Swordfish）**结构。可以看到，这种交叉三链

列残缺的现象非常严重。不过，它依然没有影响我们的删数。

接下来我们再来看两则交叉三链列的示例。

4	9	4	6	7	6	4	2 3	2 3	6	1	5
1	5	2	5	6	1	1	1	1	8	3	4
1	3	1	3	1	1	1	5	6	2	7	8
4	4	4	8	9	8	9	5	6	7	8	9
2 3	3	3	5	6	5	4	6	1	7	4	6
5	9	9	9	9	8	9	9	9	9	8	9
1 2 3	1	3	6	8	6	4	6	4	5	4	6
7	9	9	9	9	7	9	7	9	9	9	9
1	3	1	3	4	2	3	5	1	3	1	3
5	6	6	6	6	7	9	8	8	8	8	9
7	9	9	9	9	9	9	9	9	9	9	9
1	4	1	4	2	3	8	5	1	5	6	6
4	9	9	9	9	9	9	9	9	9	9	9
6	5	1	3	1	1	4	4	1	3	2	1
8	7	1	3	1	1	2	6	1	3	7	8
9	9	9	9	9	9	9	9	9	9	9	9

如图所示，这个结构便是我们在构型里讲到的，同时涉及全部区域类型的交叉三链列结构。这种构型神奇的地方就在于，例子高度对称，定义域和删除域总体构成的是一个完全中心对称的形状，非常漂亮。

3	2	8	7	1	4	1	1	6	5
1	5	9	4	6	1	8	2	7	3
1	6	7	5	2	3	8	1	4	4
8	1	6	4	7	5	1	9	3	2
2	3	1	1	1	6	5	4	7	7
7	4	5	1	3	2	6	1	8	9
6	8	3	2	5	7	4	2	5	1
1	4	5	1	2	6	3	2	5	8
1	1	1	2	1	1	7	2	5	6
4	5	9	9	9	9	9	9	9	9

如图所示，这是另外一则和上一则构型一样的示例。这则示例也不必过多叙述逻辑，依旧是满足定义域和删除域区域数相同，以及全覆盖要求。

2-3-3-2 交叉四链列 (Mutant Jellyfish)

接下来我们来看一些交叉四链列 (Mutant Jellyfish) 的例子。

4	1 2 7	1 3	2 3 6	2 7	3 6 9	2 3 9	8	5
9	2 3	5	8	2 3	4	1	6	7
6	2 3 7	8	5	2 3 7	1	9	2 3	4
2 3 5	9	2 3 6	7	4	5 8	6	1 3 8	1 2 3
2 3	8	7	9	1	6	5	4	2 3
1 3 5	4	1 3 6	2 3	2 3 8	5 8	6	7	9
1 2 3	1 6	4	1 3 6	5 9	7	2 3 8	5 9	8
8	1 3 6	9	4	5 6 3	2	7	1 3 5	1 3 6
7	5	1 2 3	1 3	3 8 9	3 8 9	4	1 2 9	6

如图所示，这个结构选取了 **r238c7** 作为定义域，圈出了所有 3 后，又接着找到了 **c25b39** 作为结构的删除域，也满足了全覆盖的要求，区域数也相同，都是 4 个。所以，我们就保证了删数的成立。

1	6	7	9	4 8	2 5	2 4	5 8	3
5	3 8 9	2	4 7 8	1	3 7	6	4 7 8	8 9
4	3 8 9	3 8	5 7 8	3 7 8	2 3 7	1 2 9	5 7 8	1 5
3 6	2	9	4 7	4 7	8	1 3 5	3 5 6	1 5
7	1 5	6 8	1 3 5	2	1 3 5	3 9	6 8	4
3 8	1 4 5	4 5	6	3 9	1 5	7	2	8 9
6 8	4 5 7	4 5	1 3 7 8	3 7 8	1 3 7	3 6 9	4 9	2
2	4 7	1	3 7	5	7 9	8	4 9	6
9	3 8	3 6	2	6 8	4	5	1	7

如图所示，这个结构和上一则的构型相同，只是这一则示例可能残缺情况要比上一则要严重一点。

接着。

1 3 4 6 9	1 5 4 5 6 9	1 3 4 5 6 9	2 7 4 6 8 9	2 7 5 6 8 9	1 2 3 4 5 9	3 1 2 3 4 5 9	1 2 3 4 5 9
4 6 9	2 4 5 6 9	1 4 5 6 9	3 4 5 6 8 9	3 4 5 6 8 9	4 5 9	7 4 5 8 9	4 5 8 9
1 3 4 7 9	1 5 7 9	8 7 9	2 4 9	2 4 5 9	2 5 9	6 4 9	1 2 3 4 5 9
1 2 7 6 9	4 7 9	1 6 7 8	2 3 7 8	2 3 7 8	1 2 7 8	2 3 9	5 6 8 9
5 1 8	1 6 8	2 3 8	2 3 8	1 2 8	2 3 8	4 6 8	7 4 5 8 9
2 6 7 8 9	3 7 9	6 7 8 9	2 4 5 6 7 8	2 4 5 6 7 8	2 4 5 6 7 8	1 4 9	2 6 8 9
1 3 4 7 9	1 5 7 9	2 7 9	3 6 7 9	3 6 7 9	8 4 9	3 1 2 3 4 5 9	1 2 3 4 5 9
1 3 7 8 9	6 7 9	1 3 7 9	5 7 9	3 8 9	4 7 9	2 1 3 9	1 3 9
4 3 7 8 9	5 4 5 7 8 9	3 4 5 7 8 9	2 3 7 8 9	1 2 7 8 9	2 3 7 8 9	4 6 7 8 9	4 5 6 7 8 9

如图所示，这一则示例的定义域只涉及两行和两个宫，所以它是宫内鱼而不是交叉鱼？并不，因为删除域涉及的是三列一行，按照定义规定，删除域也满足要求的话，也应当算交叉鱼。所以实际上，这个示例是原构型的转置后的样子。

1 4 7 9	2 4 7 9	3 4 8 9	5 4 8 9	6 4 8 9	4 7 8 9	4 8 9	4 9
2 4 5 7	2 4 5 7 9	2 4 7 9	3 4 7 8	1 4 8 9	4 5 7 8 9	6 4 9	1 4 9
6 4 5 7 9	8 4 5 7 9	1 4 7 9	1 4 7 9	2 1 3 4 5 9	1 3 4 5 9	1 3 4 5 9	1 3 4 5 9
4 5 7 8	3 4 7 8	1 4 7 8	5 4 7 8	2 4 7 8	4 8 9	1 4 8 9	6 4 8 9
9 4 7 8	2 4 7 8	6 4 7 8	1 4 7 8	3 4 8 9	4 8 9	2 3 4 8 9	5 1 2 4 9
2 4 5 8	1 2 4 5 8	1 2 4 5 8	6 4 8 9	5 4 8 9	4 8 9	7 1 2 4 9	1 2 4 9
2 3 4 8	2 3 4 8 9	5 4 8 9	2 4 8 9	6 4 8 9	1 4 8 9	2 3 4 9	7 2 3 4 9
2 3 4 8	6 4 8 9	1 2 4 5 8	1 2 4 5 8	2 3 4 5 8	4 5 8 9	2 3 4 5 8 9	2 3 4 5 8 9
2 3 4 7	1 2 4 7	1 2 4 7	1 2 4 5	9 1 2 4 5	6 4 5 7 8 9	2 3 4 5	8 2 3 4 5

如图所示，这个示例给出了一个涉及两行两列定义域，而删除域确实涉及一行一列两个宫的结构。这个结构虽说长相特殊和奇怪，但是删数确实是正确的，它满足之前提到的两大要求。所以红色的 2 都可以删除。

接下来我们再来看一则和这个构型一样的示例。

1			1	1		1	1	
4 5 6	5 6	4 5 6	5	5 6	3	2	4	8
7 9	9	7 9	9	9			7	
8		2 3	2 3	1 2	4	1 3	5	9
	6	6	6	6		6		
1	2 3	2 3	1 2			1 3	1 3	3
4 5 6	5 6	4 5 6	5	8	6 4	6 4	4	6
7 9	9	7 9	9	9	9	7	7	7
4 5 6	1	2	2 3	2 3	8	4	3	3
		4 5 6	4	4 6	7	9	4	7
3	7	4	6	8	4	5	2	1
		9	9	9				
4	2	8	1 2	1 2 3		5	6	3
			4	4	7 9			4
5 6	5 6	3 3	4 5	7	2	1 3	1 3	2 3
9	8 9	5 6	9		8	4 6	4	4 6
2	4		6	3	9	1	6	5
		7				7		
5	3	1	6	4 5	2	3	3	2 3
7 9	8 9				8	4 7	4 9	4 7

如图所示，这个结构便是和上面那一则示例构型一样的例子。这则示例里依旧满足鱼能够删数的两大要求 9，所以结构是成立的。

2-3-3-3 外鱼鳍交叉鱼（Exo-finned Mutant Fish）

外鱼鳍交叉三链列（Exo-finned Mutant Swordfish）

接下来我们来看一些有关带有鱼鳍的交叉鱼的示例。和宫内鱼结构一样，我们至少从三阶开始介绍。

2	1 3	2	5	6	1 4	1 3	8	7
	4	9				4		
1 3	7	6	8	1 4	2	1 3	1 3	9
4						4 5	4 5	
8	5	1 4	3	9	7	6	1 2	1 2
							4	4
1 4	2	3	6	8	1 4 5	7	1 4 5	1 4 5
					9			
5	6	1 4	1 2	1 2	1 4	1 3	1 2 3	8
		7 9	7 9		9	4	4	
1 4	8	1 4	1 2	1 2	3	9	6	1 2
		7	7	4 5				4 5
1 2	1 4	8	1 2	3	1 5	1 4 5	7	1 4 5 6
					9			
1 3	1 3	1 5	4	7	8	2	9	1 5 6
7	9	1 2	1 2	1 2	6	8	1 4 5	3
		4 5		5				

如图所示，这个示例里给出了两行一列的定义域。不过，如果我们把 r5c7(4)算作鱼的一部分，就会发现它将会单独占据一个删除域区域，这样非常“不划算”，而且无法分配删除域区域能够合适地让每一个候选数都能被覆盖到，而且只需要三个删除域区域就做不到。所以，我们只得让它作为结构的鱼鳍。

当它为假时，其余位置就全覆盖了，而且恰好结构形成一个交叉三链列，删数是成立的；

而当它为真的时候，它只能删除掉它自己的相关单元格的 4，当然这也包括了 $r5c3(4)$ ，所以，不论鱼鳍的真假，删除的地方总含有 $r5c3(4)$ ，所以 $r5c3 \neq 4$ 。

外鱼鳍交叉四链列（Exo-finned Mutant Jellyfish）

6	1 2 3 9	2 9	8	1 2 9	1 3 9	4	7	2 5 9
4	1 2 3 7 9	2 7 9	1 2 3 7 9	5 7 9	6	1 3 7 9	5 8 9	2 5 8 9
1 2 9	1 2 7 9	8	1 2 7 9	5 7 9	4	1 3 7 9	6	1 2 7 9
2 8 9	2 7 8 9	3	4	1 3 8 9	5 7 8 9	5 9	2 5 9	6
2 9	2 7 9	6 7 9	1	5 6 7 9	3	5 6 7 9	8	4 7 5 9
5	4 6 7 8 9	4 6 7 9	6 7 9	6 8 9	2	1	3	7 9
7	1 4 8 9	6 4 8 9	6 9	1 3 5 6 9	5 8 9	1 3 5 6 8 9	2	5 6 8 9
1 2 8 9	1 2 8 9	6 8 9	6 9	5 6 8 9	7	1 3 5 6 8 9	3	5 6 8 9
3	5	2 6 9	2 6 9	2 8 9	4	7	6 8 9	1

如图所示，我们尝试选取 $r38c57$ 四个区域里的所有 5 作为鱼的定义域，就会发现，我们怎么分配删除域， $r2c7(5)$ 都会比较特殊。要么为其单独分配一个删除域区域，但是这样就会让结构多出一个删除域区域，怎么也无法找到恰好四个删除域区域能满足全覆盖的要求。所以，我们不得不把 $r2c7(5)$ 作为鱼鳍处理。

假设它为假，则交叉四链列结构就全覆盖了，区域数也是一致的，所以满足要求，删除域的其余位置的 5 都可以删除；但是由于鱼鳍的存在，我们只能删除掉鱼鳍可以对应的删除域的单元格上的 5，即此时只有 $r2c468(5)$ 可以删除，所以 $r2c468 \neq 5$ 。

7	6	5 9	1 2 8	1 8	3	4	1 2 5 9	1 2 5 9
1 5	1 4 9	3	1 2 4 7	6	1 2 4 7 9	1 2 7 9	8	1 2 5 9
8	1 4	2	5	1 4 7 9	1 4 7 9	1 3 7 9	1 3 7 9	6
1 2 7	1 2 7	8	1 2 7	5 9	5 9	6	4	3
3	1 2 7	4	6	1 7 8	1 2 7 8	1 2 7 9	5 7 9	1 2 5 9
9	5	6	1 2 3 4	1 3 4	1 2 4	8	1 2 7	1 2
4	2 9	7 9	1 3 7	1 3 7	6	5	2 3 9	8
6	3	5 7	7 8	2	5 7 8	1 9	1 9	4
2 5	8	1	9	4 5 3	4 5	2 3 9	6	7

如图所示，这一则示例和上一则的推导方式完全一样，只是这一次结构带有两个鱼鳍，

而且残缺比较严重，所以看起来结构甚至有点不太像是一个鱼。

不过不要紧，如果我们把 $r4c1(2)$ 和 $r5c7(2)$ 去掉，结构确实满足了区域数一致和全覆盖要求，所以删数肯定是可以保证的，只是鱼鳍的存在只能删除掉两个鱼鳍和删除域的交集，所以此时只能删除掉的是 $r5c2(2)$ 。

外鱼鳍交叉五链列 (Exo-finned Mutant Squirmbag/Starfish)

之前说过，交叉鱼存在一部分结构不方便转回宫内鱼的形式，所以交叉鱼的规格上限并非 4，比如下面的这几则示例，都是规格至少为 5 的交叉鱼。

4 6 9	8	3 5 6 9	2 3 4 5 6	1	2 5 6 9	2 4 5 6	7	4 5
4 6 7	5 6 7	3 4 5 6	2 3 4 5 6	3 5 6	8	9 4 5 6	2 4 5 6	1
2	1	4 5 6 9	7	5 6 4 5 6 9	4 5 6 9	3	8	
4 6 7	2 5 7	2 4	8	9 4 5	3	1	6	
8	5 6 9	4 5 6	1	7	3	4 5 9	2	
3	5 9	1	4 5 6	2	4 5 6 7 8	4 5 8 9 7	4 5	
5	4	2 6 8	2 3 6	3 6 8	7	1 8	2 6 8	9
1 6	2 6	7	9	5 6 8	1 2 5 6	2 4 5 6 8	2 4 5 6 8	3
1 6 9	3	2 6 8 9	2 5 6	4	1 2 5	2 5 6 7 8	2 5 6 8 7	5

如图所示，这个结构的定义域为 $r358c49$ ，而删除域则是 $r6c378b2$ 。如果我们不把 $r1c9(4)$ 视作鱼鳍的话，就无法找到全覆盖的情况，所以我们只能当它为鱼鳍。

当 $r1c9(4)$ 为假的时候，**交叉五链列** (Mutant Squirmbag/Starfish) 结构成立，所有删除域上其余位置的 4 都可以删掉；而当鱼鳍为真的时候，只能删除掉鱼鳍能对应到的删除域上的位置的 4，所以只有 $r1c7(4)$ 和 $r2c8(4)$ 可以删除。

3	1	5	2	4	7	4	6	4
2	7	4	8	6	3	5	4	4
8	4	6	4	1	5	7	2	4
7	6	4	1	5	5	4	4	4
4	4	3	5	5	2	6	4	4
1	5	5	6	3	4	1	8	7
4	5	3	4	2	1	4	5	6
4	5	4	1	4	6	3	7	2
6	2	4	3	4	1	9	4	4

如图所示，如果去掉 $r2c3(4)$ 和 $r4c7(4)$ ， $r2c1247$ 五个区域的所有剩余的 4 可以作为定义域的数字，然后选出五个删除域 $r378b34$ 做到全覆盖的要求。所以删除域上的其余的 4 都可以删除。但是， $r2c3(4)$ 和 $r4c7(4)$ 也可以至少一个成立，这也是一种情况。所以我们删除的位置只有它们的交集对应应在删除域上的位置的 4，所以只有 $r4c3(4)$ 。

7	6	4	1	4	3	3	4	2
4	4	8	2	4	6	7	4	4
1	4	2	4	7	5	6	9	4
2	1	4	5	4	6	9	7	8
4	8	4	7	9	2	1	4	4
9	7	1	4	4	3	2	4	5
3	5	1	3	2	7	4	1	9
4	2	7	9	4	1	5	6	3
6	4	4	4	5	8	1	2	7

如图所示，如果将 $r4c3(4)$ 和 $r8c2(4)$ 作为鱼鳍处理的话， $r4c1249$ 作为定义域，而 $r2359b5$ 作为删除域，交叉五链列就成立了。而鱼鳍的客观存在只能保证删数只可以产生在交集上，所以删数只有 $r9c3(4)$ 。

最后我们再给出两则带有三个鱼鳍的示例。因为示例非常不好观察到，所以仅仅作作为参考和欣赏，提供理论逻辑的学习。

2 8	4 8	1	2 5	2 3 5 8	7	2 3 4 8	6	9	2	1 6 9	3	1 6 9	5	1 3	8	4	7
2 7 8	4 7 8	9	1 2 8	1 2 3 8	6	2 3 4 8	3	5	7 6	1 4 7	6 9	5	1 4 7	6 9	2	3	6 9
3	5	6	9	4	2 8	1	7	2 8	8	4 6 9	6 4 9	3	2	4 6 9	7	1	6 9
5 7 9	6 7 9	3 7 9	4 5 7	8	1 7 9	4 3 7	5 6	2	4 6	7 6	4 7	8	4 7	1	9	5	2
1	3 6 8 9	4 5 8	2 6	1 2	4 9	5 6	5 8 9	7	3 4 6	3 7	5	4 6 7	4 6 7	2 4	6 4	8	1
6 7 8 9	2	4 8	1 7 9	1 6	5	3 8	1 3 8 9	1 4 6	1 7	9	2	3	8	5	4	7 6 9	6 9
5 7 8 9	9	2	5 7	6	1	5 7 8	4	3	4 8	7	5	3 6 9	6	3 7	1	2	6
4	6 7 8	5 7 8	3	2 7 8	2 8	9	1 5 8	1 2 6	5 8	1 8	2 9	6	7 8 9	7	9	4	6
5 7 8	1	3	4	2 7 8	9	2 7 8	5 6 8	2 6	9	2 7	6	4 7	1 4 7	3 4	1 3	5	8

如图所示，这两则示例都比较难以观察到。所以，提供理解就可以了。

同样，下面给出的两则有关六链列和七链列的例子都是比较大型的结构，它们也不容易观察到，所以仅供参考。

外鱼鳍交叉六链列（Exo-finned Mutant Whale）

接下来我们来看一个有关于**交叉六链列**（**Mutant Whale**）的带鱼鳍的版本例子。

4 6	5	1 2 7	4 6	9	1 3	8	1 2 3 7	1 3
3	1 2 7	9	8	2 7	5	6	1 2 7	4
4 8	6 4 8	1 2 6	4 6 7	2 4	1 3	5	1 2 3 9 7 9	1
1 2 4	1 2 4	6	9	3	8	4 7	1 4 7	5
1 4	4	8	2	5	7	3	1 4 6	6 9
5	3 7 9	3 7	1	6	4	2	7 9	8
1 2 6	1 2 3 7 8	1 2 3 6	1 2 3 5	4 5	4	2	4 6 7	3 6
9	7 8	4	3	7 8	6	1	5	2
2 6	2 3 7	2 3 5	4 5	1	2	4 9	8	3 6

如图所示，如果把 $r3c4(7)$ 去掉，则这个结构的定义域为 $r268c14b9$ ，一共涉及六个区域，而删除域则是 $r79c258b4$ ，也是六个区域，也满足了全覆盖的要求，所以鱼结构是成立的；而鱼鳍的存在，导致删数只能找能对应的地方，所以此时可以对应到的地方有 $r3c258(7)$ 和 $r1c5(7)$ ，但是很显然的是，删数只有 $r3c5(7)$ ，因为只有这一个单元格含有候选数 7。

外鱼鳍交叉七链列 (Exo-finned Mutant Leviathan)

9	4	8	³ ₅ 6	³ ₅ 6	2	7	³ ₆	1
7	² ₆	3	8	1	⁴ ₆	9	² ₄ 6	5
² ₆	¹ ₅	1	³ ₄	9	7	³ ₄	8	² ₆
² ₄ 6	² ₅ 6	¹ ₅ 6	7	² ₅ 6	³ ₆	¹ ₄ 5	³ ₄ 5 6	8
1	³ ₈	7	⁵ ₆	4	⁶ ₈	2	⁵ ₆	9
² ₄ 6	² ₅ 6	¹ ₅ 6	⁵ ₉	² ₅ 6	³ ₈	1	³ ₄ 5	³ ₄ 5 6
8	7	¹ ₄ 6	¹ ₉	3	⁴ ₆	¹ ₅ 6	¹ ₅ 6	² ₆
3	¹ ₉	2	¹ ₆	7	5	8	⁶ ₉	4
5	¹ ₆	⁴ ₉	2	⁶ ₈	⁴ ₆	¹ ₆	7	3

如图所示，如果不看 r1c5(6)和 r5c6(6)，则 r158c1379 将构成一个标准的**交叉七链列 (Mutant Leviathan)**，而删除域则是 r346c48b79。删数的逻辑就不阐述了。

2-3-3-4 内鱼鳍交叉鱼 (Endo-finned Mutant Fish)

内鱼鳍交叉三链列 (Endo-finned Mutant Swordfish)

下面我们来看一些含有内鱼鳍的示例。

³ ₆ 4	³ ₆	5	7	³ ₄	¹ ₉	¹ ₄	8	2
7	8	1	⁴ ₉	6	2	⁴ ₉	3	5
³ ₄	² ₃	² ₃	³ ₄ 5	³ ₄ 5	¹ ₈	¹ ₄	7	6
³ ₅	9	4	6	7	³ ₅	2	1	8
2	³ ₇	³ ₈	1	³ ₈	4	6	5	9
⁵ ₈	1	6	⁵ ₉	2	⁸ ₉	3	4	7
⁶ ₈ 9	³ ₆	³ ₈ 9	² ₄ 3	1	7	5	² ₉	³ ₄
4	² ₅	² ₃	8	9	³ ₅	7	6	1
1	³ ₇	³ ₇ 9	² ₄ 3	³ ₄ 5	6	8	² ₉	³ ₄

如图所示。如果将 r1c1(3)去掉，排除 r1 和 c1 重叠的影响，交叉三链列结构就会成立，删数就在删除域的位置上。不过，r1c1(3)客观存在，所以当作内鱼鳍，删除掉的是这个位置和删除域区域的交集，所以此时只有 r3c23(3)可以删除。

内鱼鳍交叉四链列 (Endo-finned Mutant Jellyfish)

2 5	7	8	1	4 2	5 6	3 5	4 3 5 6	3 5 9
3	9	5 6	4 5	7	8	2	1 4 6	1 5
1	2 5 6	4	3	9	2 5 6	8	5 6	7
2 4 5	4 6	2 5 6 9	7	1 8	3	5 9	1 8	2 5 9
8	1	3	2	5	9	4	7	6
7	2 5	2 5 9	6	1 8	4	3 5 9	1 3 5 8	1 2 3 5 9
6	4 3	1	9	2 4	2 5	7	5 3	8
9	5 8	7	5 8	3	1	6	2	4
4 2 5	4 3	2 5	4 8	6	7	1	9	5 3

如图所示。这个结构里 r7 和 b9 将产生交集，而且结构的定义域涉及两行两宫，而删除域涉及一行三列。将 r7c8(5) 视作内鱼鳍，当它不存在的时候，鱼将忽略重叠的影响，删数成立；否则删除掉的是直接对应的地方，所以只有 c8 的删除域区域是有效的。

内鱼鳍交叉五链列 (Endo-finned Mutant Squirmbag/Starfish)

9	2 4 6	3 5 6	7	1	3 6	2 3 5	8	4 5
2 3	2 4 6	8	5	3 6	9	2 7	2 3 4 7	1
7	1	3 5	2	8	4	9	3 5	6
6	5	1 3	9	4	2 3	2 7 8	1 2 3 7 8	3
2 3	2 9	1 3 9	6	7	8	1 5	3 4 5	4 5
4 8	7 8	4 7	1 3	5	1 2 3	2 3	6	9
5	8 9	2	4	3 6	7	1 3 6 8	1 9	3
1	6 7 8 9	6 9	3	2	5	4	7 9	7 8
4 8	3	4 7	1 8	9	1 6	5 6 7	5	2

如图所示，忽略 c5 和 b2 的重叠的影响，把 r2c5(3) 视作内鱼鳍处理。那么其余位置将产生定义域为 r34c57b2、删除域则是 r7c36b36 的交叉五链列结构。但是鱼鳍的存在只能保证 r2c389(3) 的删数是有效的，所以 r2c8 <> 3。

8	2						5	
	3	4		2	6	9		
6	7		1		3		2	
	1	6		2	9			
			4	1		6		
			6	7		1		
	2	3					7	6
	6	3	8			9	2	
9	5		2			3		4

如图所示,这个示例也是比较神奇的例子。它有两个内鱼鳍,不过逻辑就不过多叙述了,在前文的解析里, 它的逻辑已经说过非常多次了。

内鱼鳍交叉六链列 (Endo-finned Mutant Whale)

最后来看一个比较大的结构。

4	3			1		7		
					4	3		
	5	1		3	6			
5	6		2					
		3		5		8		
					7		9	5
			8	7	5	9	1	
	9		4					
		5		9				4

如图所示,如果我们尝试把 r1c3(2)和 r8c5(2)视为内鱼鳍,则这个结构的定义域为 r157c5b18, 删除域则是 r2c1689b7, 而删数就找内鱼鳍在删除域区域内的交集即可。

2-3-3-5 混合鱼鳍的交叉鱼 (Mix-finned Mutant Fish)

混合鱼鳍的交叉五链列 (Mix-finned Mutant Squirmbag/Starfish)

最后来看一些奇形怪状的结构。这些奇形怪状的结构同时含有外鳍和内鳍，所以结构的要求也比较大，一些小于或等于四阶的结构基本上都找不到，或者很少有这种情况的出现，因为它们可能被其它的鱼结构代替掉了，所以这里的例子至少都是五阶的。

1	6		2	3	5		1	6	4	1
7		7 8				8 9				8 9
1	5 6	5	6	2	4	7	1	6	1 3	1 3
	8 9	8 9	8					6	5 6	5 6
3	4 5	4		6	6	1	2	5	8 9	7
	8 9	8		8	9			8 9		
	6	1				2	5	3	4	3
4	7 8		4	7 8	7			7	9	9
4 5		2	4	4 5	1 3		1	6	3 1	3
7 8			7 8	7 8	7 9	8 9	4	6	6 4	
4 5	4 5	3	4 5 6	1 3	6	4 5 6	8	2	1 3	
7	7		7	7					4	
2	4	3	5	1	3	3	4	7	7 8	6
	8 9			7				9	9	
4	6 4	3	4	6	9	2	4 5 6	3	1	
7 8	8	7 8						4	7 8	4 5
4	6	1	4	6	4 5 6	8	4 5 6	3	5	2
7	9		7					7	9	

如图所示，如果我们不算 $r9c1(9)$ ，也忽略掉 $r7$ 和 $c7$ 重叠处 $r7c7(9)$ 的影响的话，那么一个交叉五链列就能被找到，删数也是成立的（定义域为 $r347c17$ ，删除域则为 $r2c258b6$ ）。而鱼鳍的存在，导致删数只能在删除域的鱼鳍对应的交集上。所以，最终能够删除的位置只有 $r9c8(9)$ 。

	1 2	1	2	2		1	2	4	3
7 8	7	5 8 9	5 8 9	5 8	6 9	4	4	9	
	2	3	1	2	4	7	1	6	1 2
8		8 9	8 9	8 9					9
4	6 4	2	4	1	2 3	3	8	7	5
		9	9			6 9			
1	4 5	4 5	4 5	6	2	3	4	4	
	7 9	8 9	8 9				8	7	
4	3	4 5	6	4 5	1 3	1 3	9	1 2	1 2
7 8	7	7	8	8	8	4		4	
4	3		2	7	1 3	1 3	5	1	6
	8 9	4	9		8	4		4	
5	6	7	2 3	1 2 3	8	1	2	1 2	
		4	4			4	4	9	9
2	8	1	6	9	5	7	3	1 2	
	4	4						4	
2	1 2	3	2	7	1	6	5	8	
9	4		4						

如图所示，如果我们忽略 $r2$ 和 $c7$ 的交集的影响，并且忽视掉 $r5c8(2)$ 的话，一个交叉五链列就成立了，定义域为 $r258c57$ ，删除域则是 $r17c19b2$ 。所以，总的删数只有

r1c8(2)，因为它是两个鱼鳍在删除域上的交集。

3	1			6	2	4		2
		7 8 9	5 8 9		5 9		5	5 8
2	6		3	3	4	1	5	9
		7 8	5 8	5 8			7 8	
	4	5	7	2	1	6	3	2
								8
5		6		2 3	8		1	4
	2		9	7		2	9	
4	7		1	2			6	3
		8 9		5		8 9		
1		3	6	4				
	2				2		5	5 8
		8 9		7	7 8 9	8 9		
	5	2	4		6	3		1
				7 8 9			8 9	
6		1	2	5			4	7
		8 9		8		5 8 9		
		3						
			4	5		2	6	
7 8 9	8		5 8 9	1	7 9	5 8 9		

如图所示，这一则示例我并不打算讲解其中的逻辑，因为它的示例和之前的推导过程完全一样。不过我可以给出提示是，如果忽略掉涂紫色和橙色的 9，定义域为 r8c15b59，删除域则是 r347c6b7。

混合鱼鳍的交叉六链列（Mix-finned Mutant Whale）

最后两则示例也是比较难看到的，仅供参考。

4			1	3	1	3	1	3	1	5	6	7	2
		5	9	5	6	9		5	8	9			
	1	5	6	7	8	1	5	4	2	1	5	6	9
3		2	2	1	5	7	6	4	1	5	8		
		5	6	9	5	9							
8	3	1	4		6	7		2	2	6		5	
					9								
	2	2	2	1 2 3	8	1	3	1		1	4	6	4
	5	6	4		9		6			9		7	
	2	2	2	1 2	1	5	8	3	1	4	6	7	
	6	4	9	7	9								
9		5	4	7	2		3	1	3	1	5	8	6
							8						
	1	2	8	3	6	5	9	7	1	2	1	4	
7	6		2	1	3	1	3	4	1	2	3	1	2
			5		8				5	8			9

如图所示，这个结构的定义域是 r368c2b18，删除域则是 r7c14589（如果忽略掉鱼鳍的影响的话）。

1			2	4	3	5		8
3	2	5		1	8			
		4	9	5	1	2	3	
5	4					8		
	3	2	8	5		4		
	1	8	4			5	9	
2	5	1	3		7	8		
4			5	8			3	
		3		2				5

如图所示，这个结构的定义域是 r37c367b2，删除域是 r2468b19（如果忽略掉鱼鳍的影响的话）。

2-4 自噬鱼 (Cannibalistic Fish)

不论是不是鱼技巧，在之前的叙述里都有提到过一个词汇：**自噬 (Cannibalism)**。只要和这个术语词沾边的结构，都是可以删除掉结构本身的一部分的情况。

它的英文名 Cannibalism 的意思是“自相残杀”，所以在术语词里，我们把这个词语翻译为“自我吞噬”，简称“自噬”，暗示“吃掉”自己身上的某一个部分，而这个标题里的英文单词 Cannibalistic 是原词语的形容词。

在前文的叙述里，我们说过，自噬其实来自于鱼结构。那么下面就我们来看看，自噬到底是怎么回事。

2-4-1 自噬鱼的形成

让我们来思考一种结构，既然定义域可以有重叠，那么，删除域有重叠，会如何呢？

		/				x	/	/			/				x	/	/
		/				x	/	/			/				x	/	/
		x				/	x	x			x				*x	x	x
		/									/						
		/									/						
		/									/						
x	x	/	/	/	/	x	/	/	x	x	/	/	/	/	x	/	/
		x									x						
		x									x						

如左图所示，这是一个之前介绍到的，全部区域类型都涉及了的交叉鱼结构。不过我们发现，删除域区域 **r3** 和 **c7** 实际上是拥有重叠的，只是图上画不出来，在 **r3c7**，这个单元格确实属于 **r3** 和 **c7** 的重叠位置，但是由于它在定义域上，所以我们无法为其图上删除域的颜色。不过，我们现在拓展一下鱼结构，把它当作这个鱼的删数结论。或者换句话说，这里的 **r3c7(x)** 实际上也是一个正确的删数，而且它甚至位于定义域上。不过，这得如何证明和说明它呢？

我们通过反证法来说明。假设我们让 **r3c7 = x**，会发生什么情况。首先是 **r3** 和 **c7** 这两处删除域区域，由于 **r3c7 = x** 的关系，**r3c3** 和 **r7c7** 此时都不能放 **x** 了，于是 **c3** 和 **r7** 两个区域能放入 **x** 的位置此时挤入 **b7** 这一个宫里。显然，这样是放不下的，因为只有一个宫，而这个宫却要放下两处 **x**，这是肯定不够放的。所以，原来的假设是错误的，也就是说，即使 **r3c7** 含有候选数 **x**，我们也必须删掉它，否则结构就不能正常放下应该拥有的三个 **x**。

我们称 **r3c7(x)** 这一处候选数，本身存在于定义域里，但由于填入后必然会出现矛盾导致结构出错的这一类候选数叫做**自噬删数 (Cannibalistic Elimination)**；而我们把带有自噬删数的鱼称为**自噬鱼 (Cannibalistic Fish)**。

在早期的文献里，自噬删数也被称为鱼鳍，即称**自噬鱼鳍**（简称**自噬鳍**，**Cannibalistic Fin**）。算作鱼鳍的原因很简单，它确实影响了结构的成立，因为它占据了定义域的一处候选数位置。不过这种影响并不大，因为它的填入就必然导致了矛盾的出现，所以此时就已经应当把它删除掉了。所以它的影响不足以大到像是内鱼鳍和外鱼鳍一般。所以本文档此时不将其算作鱼鳍类别。

另外，自噬删数我们在鱼图里使用“*x”表示，x 暗示结构在定义域区域上，表示这个单元格含有这个候选数；而“*”实际上表示的是鱼图的删数。在前面所有鱼图里，我们都没有用“*”表示删数，是因为我们可以通过涂色来说明删数位置，就没有在每一个单元格都写上该符号。而且，书写这个符号过多，就会导致它会和“x”符号混淆不清。

2-4-2 原理进一步剖析

从上一个鱼图示例里看到，似乎最开始我们给出“删除域区域交集”的信息点跟删数实际上的关系并不大。所以这仅仅是巧合？实际上并不是。我们再举一个例子来说明这一点。

	/	/				/		
	/	/				/		
	x	x				x		
	x	x				/		
	x	x				/		
	*x	*x				x		
	/	/				/		
	/	/				/		
	/	/				/		

如图所示，这个结构的定义域为 **c237**，删除域则为 **r36b4**。显然，如果我们忽视 **r6c23(x)** 的话，它满足了全覆盖要求，而且区域数也一样，所以删数是肯定成立的。

不过，**r6c23(x)** 的存在，会使得结构怎么样呢？假设我们让 **r6c2 = x**，显然它占据了两处删除域区域 **r6** 和 **b4**，以及一处定义域区域 **c2**。那么，我们还需要放置的 **x** 的位置全部被挤入到唯一的一处删除域区域 **r3** 里了，此时结构出现错误，所以 **r6c2 <> x**；同样地，如果你假设 **r6c3 = x**，照样会这样出错。

对比上一个鱼图示例，和这里的示例，可以看到，它们的删数的推导过程大致相同的地方在于，最后都会使得后续要放的 **x** 被挤入到同一个删除域区域里。比如第一个鱼图示例里，可能的 **x** 被挤入到 **b7** 这个删除域区域之中；而这个示例里，可能的 **x** 被挤入到 **r3** 这个删除域区域之中。这难免也太巧合了吧，为什么会发生这种现象？

原因是这样的。当我们一旦在删除域区域的交集上放 **x** 后，如果这个数在定义域某处上，这个时候它会同时影响到一个定义域区域和两个删除域区域，这是毋庸置疑的，因为它处于删除域区域的交集上，显然影响了两处删除域区域，而它也在定义域区域上，自然也会影响到一个定义域区域。这样一来，它就会使得原本 **n** 个定义域区域和 **n** 个删除域区域恰

好成立的结构立马变为只有 $(n - 1)$ 个定义域区域和 $(n - 2)$ 个删除域区域。

接下来的内容会比较绕，希望你能一句一句地分析和理解。 $(n - 1)$ 个定义域区域保证这个结构还需要填入 $(n - 1)$ 处 x ，而删除域保证这些区域里最多只能放下 $(n - 2)$ 处 x ，注意，此时是**最多**能放这么多个 x 。试想一下，由于原结构是全覆盖的，但现在只剩下最多可以放 $(n - 2)$ 个 x ，但定义域又必须规定此时还必须填 $(n - 1)$ 个，显然 $(n - 1)$ 是比 $(n - 2)$ 要大的，这不是矛盾了吗，最多能放的个数居然比必须规定要放的个数还要少，这怎么可能？所以矛盾了。

上述的两则示例，在这个理论里，都表示 $n = 3$ 时的情况，实际上，当 $n = 4$ 甚至更高，也都符合这个理论，因为鱼的定义域和删除域的定义是这么规定的：定义域要求每一个区域都**恰好填入**一个 x ；而删除域要求每一个区域都**最多填入**一个 x ，而我们仅通过这个定义就能得到这个结论，所以它本身就是带有普遍和广泛意义的，跟 n 数值多少是无关的。当然了，这里的 n 肯定不能小于 2，否则 $(n - 2)$ 就变为了一个负数，就没有意义了。

所以，我们只需要掌握的是，只要原鱼结构是成立的（满足区域数一样多，以及全覆盖的要求即可），那么删除域一旦存在交集，那么交集处的元素就可以删除，不论它在哪里。

这里就需要你注意一个地方了。如果删除域区域的交集如果不处于定义域上，这显然是可以删的，因为毕竟它在删除域区域上，而且又不受定义域区域填数的限制和影响，本来就可以删除，它这这就是一个普通的、平凡的删数；而如果删除域区域的交集在定义域上的话，那么就需要理论的依托才可以删数。所以，实际上它们都是可以删的。只是需要请你注意它们的区别，虽然都是可以删除的，但删数原因不同：一个是普通的删数，而一个是理论支持得到的矛盾而引起的删数。

2-4-3 自噬鱼的示例

下面我们来看一些自噬鱼的示例。

2-4-3-1 自噬三链列（Cannibalistic Swordfish）

9	1	6	8	4 5	2	4 5	3	3
4	2	5	9	1	3	8	6	1
8	5	3	4 6	1	4 5 6	1	4 5	9
5	4	7 9	2	3 6 9	8	7	6	1
2	6	8	1	4 6	4 6	9	2 3	5
2	6	3	5	7 9	1	7	6	4
3	5	2	6	2	9	1	2	1
1	9	4	3	2	5	7	8	6
2	6	2	1	2	4	3	5	9

如图所示，这是一个带有自噬删数的交叉三链列结构。首先我们来分析自噬删数的删数原因。由于它同时位于两个删除域区域和一个定义域区域上，如果贸然让此处填入 7，就会

使得两个删除域区域 **r3** 和 **c5** 受到填数影响，一个定义域区域 **b2** 受到影响。显然，整个结构需要放置三处 **7**（因为有三个定义域），但由于删除域区域目前只有一个，而定义域区域区域却有两个，删除域使得结构最多只能让结构上再放 **1** 个 **7** 进去，而定义域却要求结构必须放 **2** 个 **7**，这显然冲突了。所以 **r3c5 <> 7**；另外，由于这个数已经删除后，剩余的部分就成了一个普通的交叉三链列结构，所以删数全部都属于正常的删数情况，这里就不用阐述了。

2-4-3-2 自噬四链列（Cannibalistic Jellyfish）

接下来来看一些带有自噬删数的四链列结构。

2 3	5 6	1	2 3	7	9	4 2 3	5 6	4	3
9	2 5 8	2 3 8	6	4	3 8	1 2 3 8	1 5 8	3	7
4	7	2 3 6 8	1	2 3 8	5	2 3 8 9	3 6 8 9	3	
5	9	2 7 8	2 3 7 8	2 3 8	6	3 8	4	1	
1	2 4 8	2 4 7 8	2 3 4 7 8	5	3 7 8	3 8 9	3 8 9	6	
6	3	4 8	4 8	9	1	5	7	2	
3 1 7	6	3 6	9	3 6 8	4	1 3 7 8	2	5	
8	1 4 5 6	3 4 5 6	3 5 7	3 6	2	1 3 4 7	1 3	9	
2 3 7	2 4 5	9	3 5 7 8	1	3 7 8	6	3 8 4	3 8	

如图所示，我们尝试把 **c2689** 作为定义域，而 **r259b3** 作为删除域，显然可以看到的是，**r2c8** 位于 **r2** 和 **b3** 的交集处，而且它确实在定义域区域 **c8** 上。如果 **r2c8 = 8**，则定义域区域变为三个，而删除域区域则变为两个。显然，最多只能放 **2** 个 **8** 和**恰好放入 3 个 8** 的说法是互相矛盾的，所以矛盾。所以，**r2c8 <> 3**。其它的删数就不用再次作出介绍了，因为都是普通的宫内四链列的基本删数。

1 3	1	1 3	2	7	2	1 2 3	3	1 2 3
4 6	5	4 5 6	4 6	8 9	5 6	4 5	4	4 5
9	9	9	8 9	7	8 9	9	8 9	8 9
4 6	2	4 5 6	1	4 5 6	3	4 5	7	4 5
9	9	9	9	8	9	9	9	8 9
1 3	1	5	8	2	2	3	1 2 3	1 2 3
4 6	7 9	7 9	4	4 5	5	6	4	4 5
7 9	7 9	9	9	9	9	9	9	9
1 2	6	4	1	2 3	2 3	1 2	2 3	2 3
7 8 9	7 8 9	7 9	6	6	6	7 8	9	6
7 8 9	8	9	7 8 9	8	8	7 8	9	8 9
5	1	1	6	2 3	1 2	2 3	3	7
8	8	6	4 6	4 6	6 4	4 6	8	6
2	3	6	2	2	2	1	2	6
7 8 9	7 8 9	7 9	7 8 9	8	7 8	9	1	8 9
1 3	1	2	3	3	6	8	3	1 3
4 6	7 9	7 9	7 9	7 9	7 9	4	6	4 5 6
7 9	7 9	9	9	9	9	9	9	9
1 3	6	1 3	5	3	4	1 3	2	1 3
7 8 9	7 8 9	7 9	7 9	8	9	7 9	9	9
4 6	5	4 5	2 3	1	2	3	3	3
7 8 9	7 8 9	7 9	7 8 9	7 8 9	7 8 9	7 9	4 5 6	7 9

如图所示，这一则示例和上面的类型页非常类似，而逻辑也是一样的，就不必作出阐述了。希望你能够独立理解。

2-4-3-3 自噬五链列（Cannibalistic Squirmbag/Starfish）

1 2 3	3	1 2 3	2 3	2	3	1 3	3	3
4 5 6	4	6	5	6	6	6	6	9
7 8 9	8 9	7 9	7 9	9	7 9	8 9	8 9	9
2 3	3	2 3	2 3	6	1	3	4	7
5	8 9	8 9	9	5	9	8 9	9	9
1 3	3	1 3	3	4	8	1 3	5	2
6	6	6	7 9	7 9	7 9	6	9	9
7 8 9	8 9	8 9	9	7 8 9	8 9	7 9	7 9	9
2 3	3	2 3	1 3	1	3	2 3	3	3
6	6	6	4 6	4 6	4 5	4 5	5	9
8 9	8 9	8 9	7 8 9	8 9	7 9	7 9	7 9	9
3	1	5	4	2	4	3	6	6
8 9	8 9	8 9	7 8 9	8 9	7 9	7 9	7 9	9
2 3	7	4	3	5	3	2 3	1	8
6	6	6	6	6	6	9	9	9
7 9	7 9	7 9	7 9	7 9	7 9	7 9	7 9	9
1 3	3	1 3	1 2	1 2	6	3	3	3
4 6	4	6	8 9	8 9	8 9	5 6	6	5
7 8 9	7 8 9	7 8 9	7 8 9	7 8 9	7 8 9	7 8 9	7 8 9	9
1 3	2	8	1	7	5	3	6	4
9	9	9	6 9	9	9	6	6	9
7 9	5	6	8 9	3	4	7 8 9	2	1

这一个示例是我个人最喜欢的一个例子了，因为它的删数对于结构的每一处都是可以删掉的，非常整齐。

如图所示，这个结构的定义域为 **c23569**，删除域则是 **r147b15**。可以看到，这样看结构的话，必然 **b1** 和 **r1** 会有重叠，而 **b5** 和 **r4** 会有重叠，而重叠的位置也都恰好位于定义域区域上。假设这些重叠的位置但凡有一处填入 9，比如 **r1c2 = 9**，则定义域区域少一个，删除域区域数少两个，产生矛盾；其它的候选数也都是如此。

2-4-3-4 带鱼鳍的自噬鱼 (Finned Cannibalistic Fish)

4 7	8	3	5 7 9	1	2	6	4 5 7 9	4 7 9
1 4 7	5	1 7 9	8		6	2	1 4 7 9	3
	2	6	4	3	5 7	1 5 9	1 5 9	8
1 7		2 3 2	6	4	1 7	3 8 9	2 8 9	5
	9	1 4 7	5	2	8	3	1 4 7	6
	6	2 3 4 7	1 2 4 8	1 5 7	5	9	1 2 4 8	2 4
	3	2 7 9	2 7	5 7 9	6	4	5 8 9	1
	8	4 7 9	6	1 5 7 9	5	1 5 7	4 5 9	3
	5	1 4 9	1 4	3	2	8	7	6

最后来介绍一个带有其它鱼鳍和自噬删数的鱼结构。

如图所示，这个结构的定义域为 **r35b567**，删除域则是 **r6c2368**。可以发现，定义域区域 **r5** 和 **b6** 产生了重叠，且重叠处存在候选数，即 **r5c8(7)**。我们不得不把它视为内鱼鳍来看。假设它不存在的话，那么带有自噬删数的鱼结构就成立了，注意，删除域区域 **r6** 和 **c8** 是有重叠的，**r6c8** 也确实恰好在定义域区域上，所以它被看作自噬删数。

如果内鱼鳍 **r5c8(7)** 为假，则这个自噬删数可以删除，原因是会导致定义域区域少一个，删除域区域少两个的矛盾；而其它的删除域区域上的删数也都是可以正常删除掉的。但是内鱼鳍的客观存在，所以最终删数就只能是内鱼鳍能够对应到的位置，即 **c8** 这个删除域区域了。

可以看到，此时 **c8** 上包含了刚才的自噬删数 **r6c8(7)**，那么此时它能否被删除呢？答案自然是肯定的。因为在鱼鳍为假的时候，我们通过了自噬的原理来删掉了它；而当鱼鳍为真的时候，显然它也作为同一区域的其它候选而存在，当然也可以被删掉。所以其实这里的 **r6c8(7)** 是可以被删除的。

2-5 不饱和鱼 (Unsaturated Fish)

接下来将为你介绍的是最后一种鱼结构，之所以放在最后讲，是因为这种结构难度非常大，而且有一些基本定式无法满足，所以删数的方式和原理有一些别扭。

试想一下，我们之前学到的鱼结构，定义域区域数和删除域区域数是相同的，即一样多的，这种结构我们完全可以拿着它就可以用来删数，因为毕竟这种结构删数的原理已经通过理论严格证明了，只要满足全覆盖和区域数一致的要求，就可以删数。那么，有没有可能让定义域区域数比删除域区域数多，或者删除域区域数比定义域区域数多，但依旧全覆盖的情况呢？那么如果有，又应该如何删数呢？本节就会讨论全覆盖要求下，定义域区域数和删除域区域数不一样多时候的删数情况。这种鱼结构称为**不饱和鱼 (Unsaturated Fish)**。

2-5-1 不饱和鱼的形成和原理

在进入正文技巧内容的学习之前，我们先要掌握一个理论概念——**秩 (Rank)**。这个词汇实际上来自于线性代数里的矩阵的秩。这个词被套用到鱼结构里，又有什么特殊的用法和用途呢？

2-5-1-1 秩的概念

我们在之前的内容里，提到过一个情况：自噬。在自噬的内容里，我们说到过，**当鱼的定义域和删除域区域总数相同，而且还全覆盖的时候，鱼是可以删数的；而如果鱼的某一部分位于两个删除域区域的交集上，而位于一个定义域区域的交集的话，假设它填入后，它会同时使得两个删除域区域消失，一个定义域区域消失，致使定义域区域数少于删除域区域数，这样的话，填数就不再平衡了，因为定义域必须保证放入 $(n-1)$ 个数，而删除域却必须保证最多放 $(n-2)$ 个数。这个最多放入的总个数竟然比必须放入的总个数还要少，所以出现了矛盾。**

那么我们尝试把这种说法包装一下。我们使用删除域区域总数减去定义域区域总数，减下来的这个差值称为鱼结构的秩。那么上述内容就变为这样：**当某个结构满足全覆盖要求，而且秩为 0 的时候，鱼就可以删数了；而如果某一处位置填入后导致秩为 -1 ($(n-2)-(n-1)=-1$) 的时候，出现矛盾。**

这便是秩的一个基本的叙述模式。可以看到，有了这个说法之后，前面的说辞就会简化一些了。

2-5-1-2 所以，前文都是在讲秩为 0 的鱼？

是的，前文介绍的内容都是秩为 0 的鱼结构，因为删除域区域数和定义域区域数是相同的，所以减下来的差值必然为 0。

但请注意，前文所说的是“区域数相同，则秩为 0”，而秩为 0 的结构不代表一定能在所有删除域进行删数。我们之前发现的所有秩为 0 的鱼结构是可以删数的，这是因为结构涉及了全覆盖，且每一个候选数都只被使用过一次，但有些时候，某个（或某些）候选数将会被重复使用，此时的分析将会更复杂一些，而这一点在后续的一些例题里会出现。

2-5-1-3 那么，秩可以为负数吗？

可以从刚才的说法里看到，一个结构是不允许秩为负数的，哪怕刚才说到的 -1 ，也是不允许的；比如秩为 -2 则表示的是定义域区域数只有 $(n-1)$ 个，但是删除域区域数却只有

$(n - 3)$ 个。定义域保证结构还必须填入 $(n - 1)$ 个 x ，但删除域保证最多还只能放 $(n - 3)$ 个 x 。显然，只要后面这个数值（即最多能放置的个数）如果比前面这个数（即必须放置的）要小，不管小多少，就一定会矛盾。因为我们无法达成这里给出的要求。

那么用秩的说法就是，**一个鱼结构的秩必须大于或等于 0 才有效，即使无法删数，或者其它原因没有删数，但起码不会像是秩为负数那样立刻矛盾。**

2-5-1-4 好了，秩可以怎么用呢？

所以，我们可以基于这一点，产生一些有关秩这个理论的一些基础用法。我们把鱼作一个推广。之前我们要求的鱼的定义域区域数要必须等于删除域区域数，是为了保证鱼内部的所有删除域都是有效的。但如果我们没有必要删除那么多，或者目的并非是找这种删数多的鱼的时候，或者甚至是根本找不到删数多的鱼结构的时候，我们可以试试找找，删除域区域数比定义域区域数多的鱼结构。

当删除域区域数比定义域区域数多，那么这个差值，即鱼的秩一定就会大于 0，虽然我们并不能得到结论，但是起码它不是负数，就意味着结构是可能成立的，只是删数不知道在哪里罢了。

那么总的来说，不饱和鱼结构的使用方式是这样的。**我们如果找到一个结构，经过全覆盖要求后发现删除域区域比定义域区域多，就算找到了不饱和鱼结构。此时只要我们找到某一处位置，假设它填入后，能使得鱼的剩余结构的秩变为负数时，我们就可以认为结构产生矛盾了，故原假设错误，删掉它。**

实际上，在有时候不一定能立刻找到秩为负数的矛盾，因为有些时候结构秩看起来为 0，实际上并不是真正的 0。这些结构都非常麻烦，而且大多数例子直接使用秩来理解都不是很方便，反而还不如通过代入试填来推导矛盾来得快。所以在有些时候，试填和利用同数技巧来获得结构内部的矛盾的过程，称为**饱和性测试（Saturation Test）**。虽说试填有一些暴力，但依旧通过逻辑层面推导，所以在逻辑层面，我们是可以接受的。下面我们将拿出众多的示例给大家分析。

那么我们尝试使用这个方式来理解一些常见的不饱和鱼结构吧。

2-5-2 不饱和鱼的示例

2-5-2-1 不饱和四链列（Unsaturated Jellyfish）

示例 1

9	¹ ₈	¹ ₄	⁶	5	3	2	⁴ ₈	7	¹ ₄	⁶
¹ _{7 8}	5	2	6	4	¹ ₇		⁸ ₉	3	¹ ₉	
¹ _{4 7}	¹ ₆	3	9	8	¹ ₇		5	⁴ ₆	2	
5	4	8	3	9	6	2	1	7		
² ₆	² ₉	7	8	1	5	3	⁴ ₆	⁴ ₉	⁶	⁶
3	¹ ₆	¹ ₉	7	2	4	⁶ ₉	8	5		
¹ ₄	¹ ₃	¹ ₄	2	6	8	7	5	⁴ ₃	³ ₉	⁶
⁶	³ ₈	5	4	7	9	1	2	³ ₆		
² ₇	² ₇	⁴ ₉	1	5	3	⁴ ₆	⁴ ₉	⁶	8	

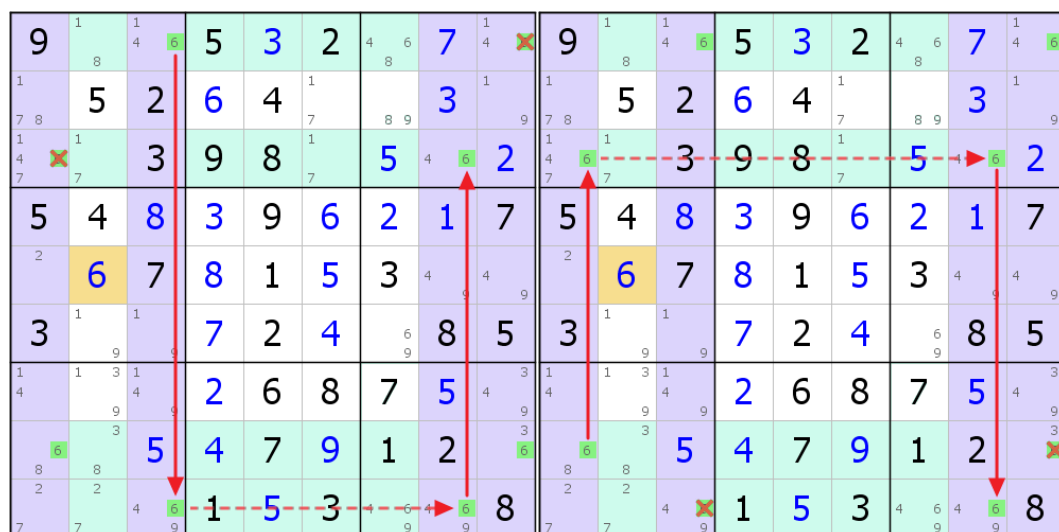
如图所示，我们尝试找到 c1389 的所有数字 6，并寻找新的区域来执行全覆盖要求。可以发现，此时我们找到了 r13589b4 一共 6 个区域，才覆盖完所有的数字。一般来说，我们都会去删除域上找删数，所以此时我们对于每一个删除域上的含有候选数 6 的单元格执行假设操作。

此时发现，假设 r5c2 = 6 的时候，我们将有下图这样的构型。

9	¹ ₈	¹ ₄	⁶	5	3	2	⁴ ₈	7	¹ ₄	⁶
¹ _{7 8}	5	2	6	4	¹ ₇		⁸ ₉	3	¹ ₉	
¹ _{4 7}	¹ ₆	3	9	8	¹ ₇		5	⁴ ₆	2	
5	4	8	3	9	6	2	1	7		
² ₆	⁶	7	8	1	5	3	⁴ ₉	⁴ ₉		
3	¹ ₉	¹ ₉	7	2	4	⁶ ₉	8	5		
¹ ₄	¹ ₃	¹ ₄	2	6	8	7	5	⁴ ₃	³ ₉	⁶
⁶	³ ₈	5	4	7	9	1	2	³ ₆		
² ₇	² ₇	⁴ ₉	1	5	3	⁴ ₆	⁴ ₉	⁶	8	

如图所示，这样的构型好像看似没有什么问题，因为定义域区域依旧是 4 个，而删除域区域则变为了 4 个。虽然变少了，但是删除域区域数依旧和定义域区域数一样多，起码它并非变为秩小于 0 的情况，即定义域区域比删除域区域数量还多的情况。不过请你注意一个

地方。由于我们假设到 $r5c2 = 6$ 的时候，结构剩下的部分变为了一个看起来很像是四链列的构型。看起来虽然没有什么异样，但请你仔细思考一下，这个结构真的存在吗？我们可以发现到的是，不论你怎么放置这些填数，任取其中三处填入，最终都必然会有一个定义域区域无法填数，这便产生了矛盾。可你会问，这是为什么呢？



我们尝试尽量使用简单的思路来解释。我们可以通过结构找到两条摩天楼结构，且它们共用一个定义域区域作为强关系产生的区域。之所以这么做，是为了后面证明出现矛盾作出铺垫。

我们可以观察图上的两种填数模式，就可以发现，此时已经产生了不对劲。左图的删数和右图的删数能够对应到 $c9$ 的唯一的两处填数位置，而这两处位置都被删除了。这便使得 $c9$ 就不能填数了。因为链是同时可以成立的，所以删数也都是同时成立的。而删数导致了 $c9$ 无法填入 6，所以出错。

那么既然错误了，这便就说明了这样的结构不存在，就得到了原本的假设 ($r5c2 = 6$ 的说法) 自然就是错误的了，所以 $r5c2 \neq 6$ 。

可是你会思考，这也只能说明这一个例子才能满足这个要求，而其它的例子不一定。实际上，其它的例子也都符合这种要求，只要我们能形成上述形状的构型的话。由于结构的特殊性，结构只会涉及 8 个单元格，而且 8 个单元格分属于四行四列，且两个一组，出现在同一个并排三宫里。这句话有点绕，具体的意思指的是，比如 $c13$ 这两处定义域区域，它们都处于 $b147$ 这样并排的三个宫里；同理， $c89$ 也是一样。这样一来，我们还有 $c1389$ 都是共轭对这个条件，这就使得结构产生了非常多的摩天楼，并且能互相删除自己的某一个端点。当我们给出的链合适时，删除的位置就应当会排除掉同一个定义域区域下的所有情况，使得结构出错。

可能你还是有问题的地方在于，定义域区域数和删除域区域数此时是一致的，那么秩为 0，根据理论，秩为 0 的结构是可以直接把删除域区域的所有位置都看作删数的地方来删数的，可这个例子怎么就不行了呢。实际上，我们在论证摩天楼存在和删数的时候，用到了过多的定义域区域来论证：首先，我们用到了四个定义域区域，是结构本身存在的；而 $b13$ 是隐藏的，因为 $b13$ 在 6 排除了 $r1389c2(6)$ 的删数。所以，定义域区域比删除域区域数还多 2 个，所以秩为 -2，导致矛盾的出现，而 $b13$ 实际上我们在结构论证矛盾（就是刚才摩天楼技巧的互相删数那里）是用到了它的，要不你自己独立想一想？

接下来我们再来看一则不饱和四链列的示例。

示例 2

4 7	8	3	5 7 9	1	2	6	4 5 7 9	4 7 9
1 4 7	5	1 7 9	8	7 9	6	2	1 4 7 9	3
2	6	1 7 9	4	3	5 7	1 5 9	5 7 9	8
1 7	2 3	2 8	6	4	1 7	3 8 9	2 8 9	5
9	1 4 7	5	2	8	3	1 4 7	1 4 7	6
6	2 3 4 7	1 2 4 8	1 5 7	5 7	9	1 4 8	1 2 7 8	2 4 7
3	2 7 9	2 7 9	5 7 9	6	4	5 8 9	2 5 8 9	1
8	4 7 9	2 6	1 5 7 9	5 7 9	1 5 7	4 5 9	3	2 4 9
5	1 4 9	1 4	3	2	8	7	6	4 9

如图所示，这个结构看起来比较简单，但删数很“离奇”：不饱和鱼按道理是通过假设得到的删数结论，而这个例子却巧合地删除了四个数字之多。下面我们就来挨个进行分析和讨论。

假如 $r6c2 = 7$ ，则 $c2$ 和 $b4$ 两个删除域区域就会由于填数假设而直接被去掉，但这个假设并不会影响到任何定义域区域（当然， $b7$ 此时可以出数，但是我们考虑通过秩的逻辑来说明，所以这里我们不继续往下假设）。

显然，定义域区域数依旧是 4 个，但删除域区域数变为了 3 个（原本有 5 个，但 $c2$ 和 $b4$ 消失，所以只剩下 3 个）。但是，此时删除域区域数比定义域区域数还少，秩为-1。我们说过，秩为负的时候，结构出错，所以这个结构便产生了矛盾。故 $r6c2 \neq 7$ 。

而其余三个删数，是通过普通鱼的视角得到的，这一点其实在之前的例子里已经说明了它们的删数缘由。

4 7	8	3	5 7 9	1	2	6	4 5 7 9	4 7 9
1 4 7	5	1 7 9	8		6	2	1 4 7 9	3
	2	6	4	3	5 7	1 5 9	1 5 9	8
1 7		2 3 2	6	4	1 7	3 8 9	2 8 9	5
	9	1 4 7	5	2	8	3	1 4 7	6
	6	2 3 4 7	1 2 4 8	1 5 7	5	9	1 2 4 8	2 4 7
	3	2 7 9	2 7	5 7 9	6	4	5 8 9	1
	8	2 4 7	6	1 5 7 9	5	1 5 7	4 5 9	3 4
	5	1 4 9	1 4	3	2	8	7	6 4 9

如图所示。如果我们把定义域区域 **b6** 的视角改变到 **r4** 的话，定义域就不饱和鱼所给的定义域完全一样了。

当然了，至于为什么可以这么切换视角，你可以尝试自己论证这一点，你可以通过秩在切换视角后依旧不产生变化这一点来论证它。

示例 3

1 3	5	1 3	9	2	7	6	8	4
2 4	8	4 6	3 4 5 6	1	2 3 5	7	9	
2 4 6 4 7	9	4 5	8	1	2 3 5 6	7	9	3
1 3 7	2 3 4 7	5	1 7	4 7	3 8	9	6	2 3
2 4 7 9	2 3 4 7 9	1 3 7 9	3 7	4 6 9	5 6 9	2 3 4 5 8	1 4 5	2 3 8
1 3 4	6	8	2	4 9	5 9	3 4 5	1 4 5	7
4 3 6 9	4 9	3 9	2	8	1	4 9	7	4 5 5 6
5	1	4 6	3	7	2 4 9	2 8	2 9	2 6 8
8	4 9	7	6	5	2 4 9	2 4	3	1

我们再来看一则示例，这一则示例则更像是第一例里那样的删数逻辑样式。

如图所示，细数定义域区域和删除域区域，定义域区域一共 4 个，分别是 **b2356**；而删除域区域则是 **r23456**，一共 5 个区域。如果 **r5c23(3)** 其中有一个为真，都会使得结构只剩下 **{r2c57, r3c69, r4c59, r6c67}(3)**，不论如何填入 3，都会出现错误。因为原本 **r789c5679** 都是不含有候选数 3 的，这使得 3 只能放在上面的单元格里；而由于 **r5c23(3)** 其一成立的关系，使得 3 的位置变为类似于之前第一则示例的样子，导致无法填数。当然，通过秩来解释就是，结构涉及的定义域和删除域个数发生了变化，导致差值为负

数，出现了矛盾。

当然，这个题更为简单的理解方式就是转化为之前看到过的一定矛盾的构型：我们尝试把 b2356 的定义域改变为 c5679，就会发现，实际上结构就变为了之前那样的矛盾的构型，你可以自己试试看。

2-5-2-2 不饱和五链列（Unsaturated Squirmbag/Starfish）

示例 1

7	1	6	4 5	5	2	8	4 5	4 5	7	1	6	4 5	5	2	8	4 5	4 5
2	4	8	4	8	1	9	4 5	8	7	6	4 5	2	4	3	1	9	4 5
3	8	9	5	7	6	2	4	3	8	5	6	4	7	3	2	4	1
5	6	2	4	8	1	4	5	9	7	5	6	2	4	8	1	4	5
1	3	5	6	7	8	9	4	5	6	7	8	9	4	5	6	7	8
4	7	8	9	5	6	1	5	6	3	2	5	6	8	9	4	7	8
3	6	4	6	4	2	6	7	1	5	4	8	9	8	9	1	5	4
1	8	5	1	2	2	4	9	6	7	3	1	5	6	7	3	1	5
6	4	6	7	3	5	5	6	4	1	2	6	4	6	7	3	5	5

如左图所示，我们需要找到 r345689b1 一共 7 个删除域区域才能实现全覆盖。所以此时的鱼的秩为 $7 - 5 = 2$ 。假设 r3c1 = 8，则结构将会消失两个删除域区域 b1 和 r3，而定义域区域不会消失。

似乎看起来秩变为 0 了。实则不然。它同时还影响着 b47 两个宫，使得在 b47 里填入 8 的位置分别只能在 {r4c3, r56c2} 和 {r8c3, r9c2} 里。那么，这样一来，仔细观察 c45 填入 8 的地方，就可以发现，不论 b6 里填入 8 的位置在哪里，左侧 c23 和 c45 填 8 的位置都能形成和之前示例完全一样的构型，而这种构型在之前都已经讲得比较清楚了，所以这种结构不用再去推导，就知道它一定是错误的。所以，原假设错误。

我们再来看一些这样的示例。

示例 2

4	3	4	6	2	3	5	7	9	3	1
8	1	5	5	9	8	1	3	4	6	2
3	1	6	6	2	1	3	3	3	5	4
7	9	7	9	9	8	8	8	8	8	8
4	5	4	5	6	5	6	7	9	2	1
2	6	7	4	8	1	5	6	9	3	3
3	6	9	8	1	5	2	3	4	7	6
1	3	5	6	9	7	8	4	6	2	4
7	5	6	2	8	7	3	4	6	1	5
7	5	6	7	5	6	4	1	9	5	6
7	7	7	7	7	7	7	7	7	7	7

如图所示，这个结构看起来好像秩为 0，对吧。实际上你可以看到，如果照着结构给出的样子的话， $r1c4(6)$ 和 $r9c6(6)$ 是无法覆盖到的。所以，这个结构的删除域实际上是 $r34678b28$ ，一共 7 个删除域区域。

我们尝试对两个删数逐个进行讨论。假设 $r7c8(6)$ 为真，则会如何呢？

4	3	4	6	2	3	5	7	9	3	1
8	1	5	5	9	8	1	3	4	6	2
3	1	6	6	2	1	3	3	3	5	4
7	9	7	9	9	8	8	8	8	8	8
4	5	4	5	6	5	6	7	9	2	1
2	6	7	4	8	1	5	6	9	3	3
3	6	9	8	1	5	2	3	4	7	6
1	3	5	6	9	7	8	4	6	2	4
7	5	6	2	8	7	3	4	6	1	5
7	5	6	7	5	6	4	1	9	5	6
7	7	7	7	7	7	7	7	7	7	7

我们画出这样的一个推导过程。注意着不是链，所以没有用实线虚线表达强弱关系。显然，这样画出来后，打勾的就是我们假设填入的地方，打叉的则是不能填入的地方。那么我们再次针对剩余的部分执行推导。

结构因为填入了 $r6c9(6)$ ，所以少了一个定义域区域，而删除域区域少了几个呢？ $r6$ 是肯定会消失的，而原假设 $r7c8(6)$ 使得 $r7$ 删除域区域消失，所以少了两个。现在依然没有矛盾，我们还得继续去发现。

4	3	4	6	2	3	5	7	9	3	1
8	1	5	5	9	1	3	4	6	2	7
3	1	6	6	2	1	3	3	5	4	
7	9	7	9	2	8	8	5	4		
4	5	4	5	6	7	9	2	1	8	
2	6	7	4	8	1	5	3			
3	8	1	5	2	4	7	4	7	6	
1	3	5	2	4	7	6	5			
5	6	2	8	1	4	7	9	5		
7	5	6	4	1	9	5	8	7	2	

接着我们观察左图，图中有一个在 c34 上的摩天楼，于是得到 $r3c56 \neq 6$ 的结果，此时观察 c5，发现 6 有出数，于是变为右图的状态。此时就可以发现，c6 无法填入数字 6，出现错误。所以，原本的假设 $r7c8(6)$ 是错误的，应当删除掉。

别急，这个示例还有一个 $r8c1(6)$ 的删数我们还没有说明。

4	³	4	6	2	³ 8	5	7	9	³ 8	1	
8	1	5		³ 5	9	1	³	4	6	2	7
³ 7	³ 9	1	⁶ 7	³ 9	⁶ 2	1	³ 6	³ 6	³ 8	5	4
4	5	³ 4	5	6	5	³ 6	7	9	2	1	8
2		⁶ 9	7	4	8	1	5		⁶ 9	3	
³ 9	8	1	5	2	³ 6	4	7	4	7	⁶ 9	
1	3	⁵ 9	⁷ 8	⁶ 4	⁶ 2	4	7	⁶ 8	9	⁵ 6	
⁶ 7	2	8		³ 4	³ 5	³ 1	4	⁷ 9	⁵ 9		
⁵ 7	⁵ 7	4	1	9	⁵ 8	⁶ 7	³ 8	³ 7	⁶ 6	2	

如图所示，似乎设定 $r8c1(6)$ 为真，更难分析得到结论。是的，确实它比较复杂，因为我们假设到的第一步就遇到了瓶颈：我们好像无法继续推导了，因为没有一处可以出数。不过我们不能灰心，实际上是有办法解决掉这个瓶颈的。

4	3	4	6	2	3	5	7	9	3	1	
8	1	5	5	3	9	1	3	4	6	2	7
3	1	6	3	2	1	3	3	5	4		
7	9	7	9	6	8	8	8	5	4		
4	5	4	5	6	5	6	7	9	2	1	8
2	6	7	4	8	1	5	6	3			
3	8	1	5	2	6	7	7	6			
1	3	5	9	7	8	4	6	2	7	8	9
6	2	8	7	4	3	5	3	1	4	7	9
5	5	4	1	9	5	6	3	2	3	6	7

4	3	4	6	2	3	5	7	9	3	1	
8	1	5	5	3	9	1	3	4	6	2	7
3	1	6	3	2	1	3	3	5	4		
7	9	7	9	6	8	8	8	5	4		
4	5	4	5	6	5	6	7	9	2	1	8
2	6	7	4	8	1	5	6	3			
3	8	1	5	2	6	7	7	6			
1	3	5	9	7	8	4	6	2	7	8	9
6	2	8	7	4	3	5	3	1	4	7	9
5	5	4	1	9	5	6	3	2	3	6	7

先看左图，可以看到，结构里有一个同数环结构。而删数是通过环结构的定义得到的，即弱关系对应的区域，所以结构里可以删除的地方是 $r3c6(6)$ 和 $r7c4(6)$ 。在删除了它们后，我们再来看右图。右图里由于删除了刚才给出的两处地方，所以剩余的部分里可以找到两个摩天楼，得到删数 $r3c5(6)$ 和 $r7c5(6)$ 。

此时就可以发现问题了：定义域区域 $c5$ 里无法找到合适的地方填 6 了，所以出现了矛盾，所以原假设 $r8c1(6)$ 错误，应当删除掉。

看了这么多例子了，我相信你应该有所感受。我们寻找删数矛盾，总是通过试填的方式来证明矛盾的出现，矛盾的出现也都只是利用结构内部（即定义域里）的所有候选数而已，并未涉及外部的任何其它的候选数。这也算作一种约定俗成的方式。

实际上，我们完全可以利用外部的数字来推导矛盾，但涉及的东西就比较多了，而且显得比较暴力。就好比 UR 里，我们通过强制 UR 的思路可以得到，实际上它也仅仅是利用了 UR 结构内部的候选数和共轭对来推导矛盾的，并没有涉及外部的其它任何数字。

我们再来看一则相对熟悉的例子。

示例 3

4	1	5	7	5	3	6	5	2
2	3	5	5	6		5	4	5
7	5	6	2	4 5	4	5	1	3
9	4 5	1	3	4 5	2	5	6	5
	5	2	5	7	4	5	9	
	7	4 5	4 5	4 5	2	5	3	1
1	4		8	2	5	3	7	6
	6	7	4	3		1	2	5
	2	3	6	1	7	5	9	4

如图所示。我们假设 $r3c2(5)$ 或 $r6c3(5)$ 其一成立的话，它们的位置看起来似乎是影响差不多的。比如 $r3c2(5)$ 如果为真，则显然会使得 $r12c3(5)$ 和 $r45c2(5)$ 同假，而 $r6c3(5)$ 也会导致一样的情况出现。所以我们可以断言，如果 $r3c2(5)$ 可以删除，那么 $r6c3(5)$ 也可以删除，我们就不必两个候选数都推理一遍了。

我们假设 $r3c2(5)$ 为真，则可以得到下面图中给出的样子。

4	1		7	5	3	6	5	2
2	3		5	6		5	4	5
7	5	6	2	4	4		1	3
9	4	1	3	4 5	2	5	6	5
		2	5	7	1	4	5	9
	7	4 5	4 5	4 5	2	5	3	1
1	4		8	2	5	3	7	6
	6	7	4	3		1	2	5
	2	3	6	1	7	5	9	4

看起来定义域区域是 5 个，删除域区域也是 5 个，但结构出错了吗？确实是出错了。我们尝试观察 $b9$ ，任意一处填数都会使得上方给出的 $r1245$ 四行定义域区域出现类似于之前遇到过很多次的出错构型。所以，整体这个结构必然是错误的。这便得到了最开始假设的矛盾，故 $r3c2(5)$ 是可以删除的；当然， $r6c3(5)$ 也是一样的。

示例 4

和上面差不多的出错方式的示例还有下面这一个。

[illegible]

如左图所示，如果我们尝试假设 **r3c3(4)** 为真，则由于 **r4** 共轭对的关系，直接可以得到 **r5c4(4)** 为真，于是依然出现了如右图所示的出错构型。

所以，这些例子都还是比较清晰和神奇的。我们再来看一则利用鱼鳍推导的。

示例 5

9	8	7	6	5	1 2	4	3	1 2
6	1 2 3	1 3	1 2	4	1 2	8	2 5	1 2 5
1 2 5	1 2 5	4	1 2 3 8 9	1 2 3 8	7 9	1 2 9	6	7
8	1 3	9	1 2	6	1 2 4	5	2 4	2 3 4
1 3 5	1 3 4 5	2	1 5 8 9	1 7 8	4 8 9	3 7 9	4 8 9	6
5	4 5 7	6	2 5 7 8 9	2 7 8	3	2 7 9	1 4 8 9	
4	2 3 7	5	1 2 3 7 8	1 2 3 7 8	6	1 2 3 7	2 7 8 9	1 2 3 8 9
1 2 3 7	9	4	1 2 3 7 8	5	6	2 7 8	1 2 3 8	4 5
1 2 3 7	6	8	1 2 3 7	9	1 2 7	1 2 3 7	4 5	4 5

如图所示，这个结构饱和五个定义域区域和五个删除域区域，不过含有四个鱼鳍，而且四个鱼鳍不都能直接得到删数的成立。不过我们依然可以尝试利用试填的方式来得到矛盾。首先，两个外鱼鳍是可以直接对应删数的，所以我们就不再讨论它们了。接着我们来讨论两个内鱼鳍的情况。

9	8	7	6	5	1 2	4	3	1 2	9	8	7	6	5	1 2	4	3	1 2
6	1 2 3	1 3	1 2	4	1 2	8	2 5	1 2 5 9	6	1 2 3	1 3	1 2	4	1 2	8	2 5	1 2 5 9
1 2 5	1 2 5	4	1 2 3	1 2 3	8 9	1 2	6	7	1 2 5	1 2 5	4	1 2 3	1 2 3	8 9	1 2	6	7
8	1 3	9	1 2	6	1 2	5	2 4	2 3	8	1 3	9	1 2	6	1 2	5	2 4	2 3
1 3 5	1 3 5	2	1 5	2	8 9	1 3	4	8 9	1 3 5	1 3 5	2	1 5	2	8 9	1 3	4	8 9
5	4 5	6	7 8 9	3	2	7 9	1	4	5	4 5	6	7 8 9	3	2	7 9	1	4
4	2 3	5	1 2 3	1 2 3	6	1 2 3	2	1 2 3	4	2 3	5	1 2 3	1 2 3	6	1 2 3	2	1 2 3
1 2 3	9	1 3	4	1 2 3	5	6	2	1 2 3	1 2 3	9	1 3	4	1 2 3	5	6	2	1 2 3
1 2 3	6	8	1 2 3	9	1 2	1 2 3	4 5	4 5	1 2 3	6	8	1 2 3	9	1 2	1 2 3	4 5	4 5

如左图所示，假设 $r5c5(7)$ 成立，则我们可以依次得到 $r4c7(7)$ 和 $r8c1(7)$ 为真的结果，由于 $r8c1(7)$ 的成立，所以此时是可以删除 $r6c1(7)$ 的。

如右图所示，假设 $r8c5(7)$ 成立，则排除掉一些情况，我们发现剩余部分里， $r59$ 形成了一个二链列结构，而这个二链列依旧可以删除 $r6c1(7)$ 。

所以实际上，两个内鱼鳍依旧可以对应到删数，所以删数 $r6c1(7)$ 不论哪个鱼鳍成立都可以删除掉；而且鱼结构的删除域也包含 $r6c1(7)$ ，所以它可以安全地被删除。

这一则示例之所以放在不饱和鱼里讲解，是因为这一则示例也是通过不饱和鱼的结构包装得到的构型。如果你在做题过程之中能发现这样的形式的结构，我们就可以通过利用鱼鳍的方式来讨论，把结构变为秩为 0 的饱和鱼结构，这样看起来逻辑比起之前的示例要清晰一些。

示例 6

4	6	8	5	7	9	3	2	4	6	1
1	4	6	3	4 5 6	2	5 6	5 6	5 6	5 8 9	7
7	2	6	4	1	4 5 6	5 6	5 6	3	4 5 8 9	3
2	1 2	5	6	7	4 5 6	4 5 6	5 6	3	4 5 8 9	8
4	6	3	4	2	5 6	4 5 6	1 2	1	5 6	7
8	1 2	5	6	4	2 3	4 5 6	1 2	5 6	4 5 8 9	9
3	1 2	7	1 2	2	5 6	5 6	4	5 8	5 8	9
5	7	9	8	2	1	7	9	4	3	6
4	2	4	6	8	3	2	7	1	5	9

如图所示，这是一个交叉五链列，看起来还不错，不过多了两个内鱼鳍。我们尝试逐个进行讨论。

4	8	5	7	9	3	2	4	1	4	8	5	7	9	3	2	4	1
1	4	3		2		5	6	7	1	4	3		2		5	6	7
7	2	4	1			5	6	7	7	2	4	1			5	6	7
2	3	7	2	3	1	2	3	1	2	3	7	2	3	1	2	3	1
4	3	7	2	3	1	2	3	1	2	3	4	3	7	2	3	1	2
8	1	2	3	4	5	6	7	8	8	1	2	3	4	5	6	7	8
3	1	2	3	4	5	6	7	8	3	1	2	3	4	5	6	7	8
5	2	3	4	5	6	7	8	9	5	2	3	4	5	6	7	8	9
4	2	3	4	5	6	7	8	9	4	2	3	4	5	6	7	8	9

如图所示，这是两个讨论情况。可以看到的是，不论哪种情况成立，删数都是可以删掉的，因为最终我们都会在 b46 里发现一组级联区块结构。级联区块可以当作 X-Wing 使用，而它们都会删除 r4 这一个区域的 6。

示例 7

5	6	2	1	3	4	8	5	6	5	6	2	1	3	4	8	5	6
8	1	4	7	9	3	2	5	6	8	1	4	7	9	3	2	5	6
3	7	5	6	8	2	4	1	3	3	7	5	6	8	2	4	1	3
2	4	1	9	6	5	3	2	3	2	4	1	9	6	5	3	2	3
5	6	3	5	6	2	4	8	1	5	6	3	5	6	2	4	8	1
2	3	5	6	7	1	9	4	2	2	3	5	6	7	1	9	4	2
1	5	6	7	8	9	1	2	3	1	5	6	7	8	9	1	2	3
4	5	6	7	8	9	1	2	3	4	5	6	7	8	9	1	2	3
5	6	2	3	4	5	6	7	8	5	6	2	3	4	5	6	7	8

如左图所示，我们找到一个交叉五链列结构，不过删除域区域比较多，要想覆盖完整就需要 r1678b2368 一共 8 个区域才行。不过不要紧，删除域区域再多也不害怕，因为删数一定是在删除域区域上的。那么假设我们让 r6c9(5)为真，会如何呢？

如右图所示，我们按照顺序推导，就会发现，b3 里同时出现两处填入 5 的地方，这便是产生了矛盾。所以原来的假设是错误的。

那么，我们给出了这么多例子，下面再给出两则示例，希望你能够自己理解它们。

示例 8

2	4 5 8	4 5 7	4 5 8	1	9	4 5 8	6	5 7 8
1 4 5 6	4 5 6 8	1 4 5 7	4 5 8	5 8	3 4 6 8	2	4 5 7	1 5 7 8 9
1 4 5 6	9	3	7	2	4 6 8	4 5 8	4 5	1 5 8
3	1	5 9	5 6 8	4	2 6 8	7	5 9	2 5 8
4 5 7 9	4 5	8	1 2 5	3 5	1 2 3 7	6	2 3 5	2 5 9
5 6 7	5 6	2	5 3 8	9	3 7 8	5 3 8	1	4
4 9	2 3 9	4 9	2 3	7	5	1	8	6
1 5 8	2 3	6	1 4 8	3 8	1 2 3 4 8	9	2 4 5 7	2 5 7
1 5 8	7	1 5	9	6	1 2 4 8	4 5	2 4 5	3

这一则示例看起来比较麻烦，但实际上并不难。

示例 9

2	1 8 9	4	5	1 7 8	6 7 8	1 8 9	1 6 9	3
6	1 5 8	7	1 8	3	9	4	2 5	2 5 8
1 5 8	3	1 8 9	1 6 7 8	2 4	2 4	1 6 8 9	5 6 7 9	6 7 8
7 8	2	3	6 7 8	4 5 8	4 5 6 7 8	8 9	5 6 9	1
1 8	1 8	6 8	5	2	9	3	7	4 6 8
4	6 7 8 9	6 8 9	1 6 7 8	1 5 7 8	5 7 8	2	3	5 6 8
3	4	1 6	9	2 5 7	2 5 7	1 6	8	2 7
1 5 7	1 5 7	2	4	6	7 8	3	1 7	9
9	6 7 8	6 8	3	2 7 8	1	5	2 6 7	4

这一则也是一样。

2-5-2-3 不饱和六链列 (Unsaturated Whale)

接下来我们来看一些不饱和六链列的示例。

示例 1

9	4	6	4	6	3	2	8	7	5	1	6	9	4	6	4	6	3	2	8	7	5	1	6
1 2	2	1	5 6	1	7	1	6	1	4	3		1 2	2	1	5 6	1	7	1	6	1	4	3	
3	6	7	6	4	6	4	6	5	2	1	6	3	6	7	6	4	6	4	6	5	2	1	6
1 2	3	4	5 6	4	5 6	4	5 6	1 2	4	2	8	1 2	3	4	5 6	4	5 6	4	5 6	1 2	4	2	8
2	2	4	5 6	8	4	5 6	4	5 6	1	3	2	2	2	4	5 6	8	4	5 6	4	5 6	1	3	2
7	2	1	4	5	4	5	8	9	6	2	5	7	2	1	4	5	4	5	8	9	6	2	5
5	7	9	2	1	5	3	3	6	3	4		5	7	9	2	1	5	3	3	6	3	4	
6	1	7	9	5	8	9	4	5	3	2	3	6	1	7	9	5	8	9	4	5	3	2	3
4	5	4	5	3	2	7	6	5	1	9		4	5	4	5	3	2	7	6	5	1	9	

如左图所示，这个示例看起来很漂亮，因为结构高度对称。假设 $r6c5(9)$ 为真，则变为右图这样。

右图看起来似乎很麻烦，但实际上，它可以转化为之前我们就讲到过很多次的矛盾的构型。我们随意拿出一处定义域区域的两个情况来进行讨论，就可以发现讨论后一定会出现矛盾，这里我们就不重复给出示例的推导过程了，你可以自己试试看。

示例 2

9	1	4	2	3	8	6	7	5	9	1	4	2	3	8	6	7	5
6	2	3	7	4	6	5	1	4	6	2	3	7	4	6	5	1	4
5	6	5	4	5	4	6	9	1	5	6	5	4	5	4	6	9	1
1	3	2	3	2	4	3	1	4	1	3	2	3	2	4	3	1	4
1	5	2	3	1	2	3	1	2	1	5	2	3	1	2	3	1	2
4	6	1	2	5	8	1	5	2	4	6	1	2	5	8	1	5	2
2	4	5	8	3	1	5	3	6	2	4	5	8	3	1	5	3	6
3	4	3	6	1	4	9	5	2	3	4	3	6	1	4	9	5	2
1	5	7	9	2	3	8	3	6	1	5	7	9	2	3	8	3	6

如图所示，假设 $r5c2(2)$ 为真，则会产生矛盾。这一则示例就自己理解了，我们不再重复叙述它的具体逻辑。

2-6 最后的介绍

鱼的内容就全部说完了，最后我们需要介绍一些理论的内容，以便编写电脑程序，或者书写参考文档时使用。

2-6-1 为什么同数技巧全都可以叫做鱼？

这一点我们其实看过众多例子了，所以我们不必过多叙述就能明白这一点。二阶的鱼结构我们可以转化变为链结构，例如双强链，或者带有区块的双强链结构；而三阶的甚至更高阶的，我们都可以通过鱼鳍的包装来得到推导；而包装比较麻烦的，可以通过远程试填来得到矛盾，或者甚至不用包装鱼鳍，直接上手就开始找内部的矛盾。所以，所有同数技巧其实都可以包装形成鱼结构。

2-6-2 自噬为什么是鱼的内容，而不是其它技巧的内容？

这一点我们也不必去过多叙述了。自噬就是通过删除域区域的交集而产生的一种特殊删数。如果它还在定义域区域上，那么删除它的原因就是因为原本结构的秩为 0，而因为填入后，定义域区域少一个，而删除域区域少两个，秩变为-1，出现了矛盾。

不过可以通过鱼结构的理论知道，当鱼结构本身的秩就已经不是 0 的时候，我们并不难妄自下结论，说某个处于删除域交集上的候选数依旧是自噬删数。这样是不对的，因为它即使填入，鱼结构的秩也不能保证变为多少；更何况有一大部分鱼结构的秩并不是结构得到 0 就必然能够得到删除域区域全删的结论的。

2-6-3 鱼的系统命名规则

我们先来说说饱和鱼结构（即定义域区域和删除域区域数相同，且全覆盖的情况）的命名规则。

我们有如下的顺序来修饰一个鱼：

（鳍化修饰符） 鳍数量修饰符 鳍名修饰符 形状修饰符 规格修饰符

解释如下：

- **鳍化修饰符**：表示鳍影响到结构的类型。有三种鳍化类型：**标准**（Basic）、**退化**（Sashimi）和**孪生**（Siamese）。当为“标准”时，应当省略不写。且“退化”一词的位置应调整至鳍名修饰符之后。
- **鳍数量修饰符**：表示有多少个鳍。如果有多种鳍，则按照外鳍、内鳍、自噬鳍的顺序排列。如果同一种鳍只有一个或两个的，则使用**单**（Single）和**双**（Double）修饰，超过两个的，则使用汉字数字**三**（Triple）、**四**（Quadruple）、**五**（Quintuple）等表示。
- **鳍名修饰符**：表示鳍的名字。有两种鳍：**外鳍**（Exo-fin）、**内鳍**（Endo-fin）。拼写时应按照此顺序排列，并配合鳍数量修饰符描述。
- **形状修饰符**：表示形状。一共有三种形状：标准、宫内、交叉。
- **规格修饰符**：表示规格。规格取自于结构的定义域区域数。一般一共有六种规格（范围在 2 到 7 之间），使用汉字数字后添加“链列”二字即可。超过其规格的，仍使用数字，但结构本身一般不符合其标准；英文里一般有六种规格（范围在 2 到 7 之间）：**X-Wing**、**Swordfish**、**Jellyfish**、**Squirmbag**（或**Starfish**）、**Whale**、**Leviathan**。超过其规格的，使用“**数字 + 连字符 + Fish**”的写法，如**8-Fish**，但结构本身一般不符合其标准。

合格的命名可以是如下的示例：

- 双外鳍单内鳍退化宫内五链列
- 四外鳍三链列

当然，你也可以使用 **D\S 描述**（**D\S Description**）来描述鱼结构。如：

- rrb\ccc 鱼（三阶、宫内）
- ccbb\rrrc 鱼（四阶、交叉）

其中用 **r** 表示一个行区域，**c** 表示一个列区域，**b** 表示一个宫区域。反斜杠左侧的部分是鱼的定义域涉及的区域的类型集合，右边则是删除域区域涉及的区域集合。

当然，也可以直接将定义域区域和删除域区域、结构涉及的数字、甚至是鱼鳍体现在其中。涉及的数字应写在 **D\S** 描述的前面，而鱼鳍应写在 **D\S** 描述之后，用空格隔开，且使用字母 **f**、**ef** 前缀表示属于外鳍还是内鳍，并且以“外鳍、内鳍”的顺序排放。如：

- 2 r19b26\c578b1 fr3c6 fr9c3
- 1 r9c4b6\r47c8 fr9c3
- 9 c29b6\r468 efr4c9

这样可以直接看出规格、形状。其中的 **fr3c6** 就是指明有一个外鳍位于 **r3c6**。第一个示例涉及的候选数是 2，而第二个示例涉及的则是 1。当详细描述时，末尾的“鱼”字可以被省略。

当鱼的定义域区域数少于删除域区域数时，称为**不饱和鱼**（**Unsaturated Fish**）。这个术语词已经在之前提到过了。

当不饱和时，删除域区域数大于定义域区域数，此时应为鱼名称最前面添加“不饱和”一词，规格按照定义域区域数计算。比如定义域区域有 4 个，而删除域区域有 5 个，则我们认为结构是一个四链列，而不是五链列。

2-6-4 鱼图的符号规则

鱼图里一般想要表达出意思，需要不同的符号标识来表达。下面罗列出所有需要用到的符号。

- 永假符号 **/**：放到单元格里，表示该单元格不能放入鱼涉及的这个数；
- 鱼结构符号 **x**：表示当前单元格可能填入这个数字；
- 删数符号 *****：表示当前单元格可以通过结构删除该单元格的这个数字；
- 外鱼鳍符号 **F** 或 **δ**：表示当前单元格的这个数字是作为外鱼鳍出现的；
- 内鱼鳍符号 **E** 或 **@**：表示当前单元格的这个数字是作为内鱼鳍出现的；
- 自噬符号 **C** 或 ***x**：表示当前单元格的这个数字是作为自噬删数出现的；
- 重删数符号 ******：表示当前单元格在一个饱和鱼结构不论鱼鳍是否成立，都能删除的地方；
- 残缺符号 **I** 或 **!**：表示当前单元格是结构不一定必需涉及的地方，它可以不存在，结构照样成立，且删数成立；
- 未知符号 **?**：表示当前单元格的删数是否成立，并不清楚。

Part 3 飞鱼导弹 (Exocet)

接下来我们将进入第三个高难的技巧板块：**飞鱼导弹 (Exocet)**，而这个技巧在之前的所有技巧内容里都没有出现过，难度可见一斑。

在早期文献里，这个技巧被翻译为**鱼雷**和**潜水艇**，不过潜水艇在早期文献里是一个错误翻译。

3-1 基础使用

3-1-1 4 数初级飞鱼导弹 (4-Digit Junior Exocet)

1 2 3 5 7 8	2 3 5 6 7	1 3 5 6 8	4	5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 7 8 9	2 3 5 8 9
4	2 3 5 6 7	3 5 6	2 5 6 7	8	9	1	2 3 6 7	2 3 5
2 5 7 8	9	5 6 8	2 5 6 7	3	1	2 5 6 7	2 4 6 7 8	2 4 5 8
1 2 3 4	2 3 4	1 3 4 9	1 2 4 7 8	1 4 7 9	5	2 3 7 9	1 2 3 7 8 9	6
1 2 5	8	1 5 9	3	1 6 7 9	2 6 7	4	1 2 7 9	1 2 9
6	2 3 4	7	1 2 8	1 4 9	2 4 8	2 3 9	5	1 2 3 8 9
5 7	3 4 5 6 7	1 4 5 6	9	1 4 5 6 7	3 4 6 7	8	2 3 4 6 7	2 3 4 5
5 7 8	3 4 5 6 7	2	1 5 6 7 8	1 5 6 7	3 4 6 7 8	3 5 6 9	1 3 4 6 9	1 3 4 5 9
9	3 4 5 6 8	3 4 5 6 8	5 6 8	2	3 4 6 8	3 5 6 4	3 1 4 6	3 4 6 7

如图所示，这个结构看起来涂色的单元格非常多，所以看起来结构挺复杂的。我们来看看怎么去推理。首先我们注意到，**r456789c247** 这 18 个单元格里，填入 2、5、6、7 的位置最多只有两个。我们换一种比较容易理解的说法：在 **r456789c247** 这样 18 个单元格里，最多能够放下两个 2。这是显然的，因为 2 在这 18 个单元格里只出现在 **r46c247** 这样 6 个单元格之中。显然，这 6 个单元格分属于 **r46** 两行上，所以显然只可能填入最多两个 2，任何第三个数字 2 的填入都会在这 6 个单元格的其中任意一行里产生重复。同理地，我们可以发现 5、6、7 也是一样的道理，如下面给出四个图所示，我们分别把 2、5、6、7 四种不同的数字的填数情况的位置全部都标注出来了。

1 2 3 5 7 8	2 3 5 6 7	1 3 5 6 8	4	5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 7 8 9	2 3 5 8 9	1 2 3 5 7 8	2 3 5 6 8	4	5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 7 8 9	2 3 5 8 9
4	2 3 5 6 7	3 5 6	2 5 6	8	9	1 2 3 5 6	2 3 6 5	2 3 2 3	4	2 3 5 6 7	3 5 6	2 5 6	8	9	1 2 3 5 6	2 3 2 3
2 5 7 8	9	5 6 8	2 5 6	3	1	2 5 6 7 8	2 4 6 7 8	2 4 5 8	2 5 7 8	9	5 6 8	2 5 6	3	1	2 5 6 7 8	2 4 5 8
1 2 3 4 9	2 3 4 9	1 3 4	1 2 7 8	1 7 9	5	2 3 7 9	1 2 3 7 8 9	6	1 2 3 4 9	2 3 4	1 2 7 8	1 3 7 9	5	2 3 7 9	1 2 3 7 8 9	6
1 2 5	8	1 5 9	3 7 9	1 7 6	2 7 9	4 7 9	1 2 7 9	1 2	1 2 5	8	1 5 9	3 7 9	2 7 6	4 7 9	1 2 7 9	1 2
6	2 3 4	7	1 2 8	1 4	2 9 8	2 3 9	5 8 9	1 2 3 8 9	6	2 3 4	7	1 2 8	1 4	2 9 8	2 3 9	5 8 9
3 5 7	1	4 5 6 7	9	4 5 6 7	4 6 7	8	2 3 4 5	2 3	3 5 7 8	1	4 5 6 7	9	4 5 6 7	4 6 7	8	2 3 4 5
9	4 5 6 8	4 5 6 8	3 5 6	2	4 6 8	3 5 6	1 3 4 6	7	9	4 5 6 8	4 5 6 8	3 5 6	2	4 6 8	3 5 6	1 3 4 6

1 2 3 5 7 8	2 3 5 6 7	1 3 5 6 8	4	5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 7 8 9	2 3 5 8 9	1 2 3 5 7 8	2 3 5 6 8	4	5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 7 8 9	2 3 5 8 9
4	2 3 5 6 7	3 5 6	2 5 6	8	9	1 2 3 5 6	2 3 6 5	2 3 2 3	4	2 3 5 6 7	3 5 6	2 5 6	8	9	1 2 3 5 6	2 3 2 3
2 5 7 8	9	5 6 8	2 5 6	3	1	2 5 6 7 8	2 4 6 7 8	2 4 5 8	2 5 7 8	9	5 6 8	2 5 6	3	1	2 5 6 7 8	2 4 5 8
1 2 3 4 9	2 3 4 9	1 3 4	1 2 7 8	1 7 9	5	2 3 7 9	1 2 3 7 8 9	6	1 2 3 4 9	2 3 4	1 2 7 8	1 3 7 9	5	2 3 7 9	1 2 3 7 8 9	6
1 2 5	8	1 5 9	3 7 9	1 7 6	2 7 9	4 7 9	1 2 7 9	1 2	1 2 5	8	1 5 9	3 7 9	2 7 6	4 7 9	1 2 7 9	1 2
6	2 3 4	7	1 2 8	1 4	2 9 8	2 3 9	5 8 9	1 2 3 8 9	6	2 3 4	7	1 2 8	1 4	2 9 8	2 3 9	5 8 9
3 5 7	1	4 5 6 7	9	4 5 6 7	4 6 7	8	2 3 4 5	2 3	3 5 7 8	1	4 5 6 7	9	4 5 6 7	4 6 7	8	2 3 4 5
9	4 5 6 8	4 5 6 8	3 5 6	2	4 6 8	3 5 6	1 3 4 6	7	9	4 5 6 8	4 5 6 8	3 5 6	2	4 6 8	3 5 6	1 3 4 6

很显然，它们都只出现在最多两行上，所以最多只可能允许填入两次。

接着，我们转变一下观察和推理的视角。c247 一共三列，每一列都必须出现完整的一组数字 1 到 9 序列，所以 c247 一共三列，就必须出现三次完整的 1 到 9 序列。但 2、5、6、7 这四种数字在这 18 个单元格里只能最多出现两次在里面，那么与之互补的剩余的 9 个单元格里，就必须出现至少一个 2、一个 5、一个 6 和一个 7，才能满足此要求，如图所示。

1 2 3	2 3	1 3	4	5 6	2	2 3	2 3	2 3
5	5 6	5 6			6	5 6	6	5
7 8	7	8		7	7	9	7 8 9	8 9
4	2 3	3	2	5 6	8	9	1	2 3
	5 6	5 6	5 6				6	5
	7		7				7	
2	9	5 6	2	3	1	2	2	2
5		5 6	5 6			5 6	4	6 4 5
7 8		8	7			7	7 8	8
1 2 3	2 3	1 3	1 2	1	5	2 3	1 2 3	6
	4	4		4		7	9	7 8 9
1 2		1	1		2		1 2	1 2
5	8	5	3	6	6	4	7	9
	2 3		1 2	1	2	2 3		1 2 3
6	4	7	8	4	8	9	5	8 9
	3	3		3		2 3	2 3	2 3
5	1	4 5 6	9	4 5 6	4 6	8	4	6 4 5
7								
	3	3	1	1	3	3	1	3
5	4 5 6	2	5 6	4 5 6	4 6	5 6	4	6 4 5
7 8	7		7 8	7	7 8	9	9	9
	3	3	1	3	3	1	3	
9	4 5 6	4 5 6	5 6	2	4	5 6	4	6
		8	8		8			

抛开 r1c4、r2c7 和 r3c2 不看（因为它们都是确定值，无法填入其它数字了），也就是说，在剩余的六个单元格里，必须至少有一个单元格里是填入 2 的，有一个是 5，有一个是 6，还有一个地方是 7。

接着，我们把视线放到 r1c56 上。恰好，r1c56 只有 2、5、6、7 这四种候选数。我们假设最终这两个单元格放的是 2、5、6、7 的其二（为了方便表达，我们假设最终的填数为 a 和 b，其中 $a, b \in \{2567\}$ ），把 a 和 b 试填放到结构里，就会发现，此时能够排除掉一些填数位置。

1 2 3	2 3	1 3	4	ab	2	2 3	2 3	2 3
5	5 6	5 6			6	5 6	6	5
7 8	7	8		7	7	9	7 8 9	8 9
4	ab	3	2	5 6	8	9	1	2 3
	5 6	5 6	5 6				6	5
	7		7				7	
2	9	5 6	2	3	1	ab	2	2
5		5 6	5 6			5 6	4	6 4 5
7 8		8	7			7	7 8	8
1 2 3	2 3	1 3	1 2	1	5	2 3	1 2 3	6
	4	4		4		7	9	7 8 9
1 2		1	1		2		1 2	1 2
5	8	5	3	6	6	4	7	9
	2 3		1 2	1	2	2 3		1 2 3
6	4	7	8	4	8	9	5	8 9
	3	3		3		2 3	2 3	2 3
5	1	4 5 6	9	4 5 6	4 6	8	4	6 4 5
7								
	3	3	1	1	3	3	1	3
5	4 5 6	2	5 6	4 5 6	4 6	5 6	4	6 4 5
7 8	7		7 8	7	7 8	9	9	9
	3	3	1	3	3	1	3	
9	4 5 6	4 5 6	5 6	2	4	5 6	4	6
		8	8		8			

如图所示，我们一不小心就排除掉了 r1c27 和 r23c4 四个单元格填入 a 和 b 的情况了。而在和之前说到的那 18 个单元格互补的 r123c247 之中，仅剩下两个单元格还能放下 a 和 b。比如说，假如 a 和 b 是 2 和 5，那么 r2c2 和 r3c7 里必须有一个 2 和一个 5。这是显然的，因为 6 和 7 我们不知道放哪里，这一点没有约束，但 2 和 5 已经在 r1c56 里形成了数对形式，导致了删数的影响，最终只能放到 r2c2 和 r3c7 里。而这两个单元格显然

不能填入一样的数字，是因为 2、5、6、7 必须在这 9 个单元格里都至少出现一次，而 6、7 我们可以随意保证（刚才的假设 r1c56 是 2 和 5 跟 6 和 7 是无关的，所以 6 和 7 是可以随意放的），而 2 和 5 就只能在这两个单元格里了，而都得出现至少一次，就必须让两个单元格填入一个 2 和一个 5 才可以满足这个要求了。因此，这两个单元格里一个是填入 2，另外一个填入 5 的。当然，这是假设，所以换回 a 和 b 的话，r2c2 和 r3c7 里就必须有一个 a 和一个 b 的出现，而且 a 和 b 还是不同的数字。

不管 a 和 b 填入的数字是什么，我们都能清楚地明白，r2c2 和 r3c7 里都不能填和 2、5、6、7 无关的其余任何数字了，所以 r2c2(3) 是应当被删除的，因为这个候选数一旦成立，就会占据其中一个单元格使得填入 a 和 b 的位置不够，进而出错。

这个结构推导分了非常多个小的步骤：首先是观察下方的 18 个单元格，确定 2、5、6、7 的填数都是最多两个的；然后是上面互补的 9 个单元格里至少出现各自一个；接着是发现上方有两处单元格里恰好只有 2、5、6、7，于是假设它们是填入 2、5、6、7 其二，然后用未知数 a 和 b 试填到单元格里，并最终发现 a 和 b 只能放到我们规定的 r2c2 和 r3c7 之中的，所以两个单元格里必须都是 2、5、6、7 的数字，别无其它，所以无关的候选数都能被删除掉。

这个结构有一点点用到了鱼的东西，比如 2、5、6、7 放入 c247 的情况就有一些类似于鱼的定义域，因为鱼的定义域规定，一个区域内填入某个数字是恰好一次的，所以这个看起来就好像是一个涉及 2、5、6、7 四种数字的剑鱼，而且恰好位置都在 c247 上。而这个技巧被称为**飞鱼导弹 (Exocet)**也跟鱼有关，可见结构和鱼的相关性是非常大的。而目前我们遇到的例子是基础版本的，所以按照分类，它应属于**初级飞鱼导弹 (Junior Exocet, 简称 JE)**；而又因为结构涉及的 2、5、6、7 四种数字，所以又可以直接称为**4 数 JE**。

这个结构涉及到了非常多的单元格，所以我们不得不提及结构涉及的一些术语词。

3-1-2 术语词

为了介绍简单，我们给出一些示意图，来表达我们需要表述的内容。

1 2 3 5 7 8	2 3 5 6 7	1 3 5 6 8	4	5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 5 7 8 9	2 3 6 5 8 9	1 2 3 5 7 8	2 3 5 6 8	4	5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 5 7 8 9	2 3 6 5 8 9
4	2 3 5 6 7	3 5 6 7	2 5 6 7	8	9	1	2 3 6 5 7	2 3 6 5	4	2 3 5 6 7	3 5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 5 7	2 3 6 5 7 8 9	2 3 6 5 8 9
2 5 7 8	9	5 6 8	2 5 6 7	3	1	2 5 6 7	4 6 4 5 7 8 9	2 2 4 5 8	2 5 7 8	9	5 6 8	2 6 7	2 3 5 6 7 9	2 2 4 6 4 5 7 8 9	2 2 4 6 4 5 8	2 2 4 5 8
1 2 3 4	2 3 4	1 3 4	1 2 7 8	1 4 7 9	5	2 3 7 9	1 2 3 7 8 9	6	1 2 3 4	2 3 4	1 2 4	1 3 4	1 2 4	1 4 7 9	5	2 3 7 8 9
1 2 5	8	1 5 9	3	1 6 7 9	2 6 7	4	1 2 7 9	1 2 9	1 2 5	8	1 5 9	3	1 6 7 9	2 6 7	4	1 2 7 9
6	2 3 4	7	1 2 8	1 4 9	2 4 8	2 3 9	5	1 2 3 8 9	6	2 3 4	7	1 2 8	1 4 9	2 2 3 9	5	1 2 3 8 9
5 3 7	1	4 5 6 7	9	4 5 6 7	4 6 7	8	2 3 4 6 4 5 7 8 9	2 3 4 5	5 3 7	1	4 5 6 7	9	4 5 6 7	4 6 7	8	2 3 4 6 4 5 7 8 9
5 3 7 8	4 5 6 7	2	1 5 6 7 8	4 5 6 7	4 6 7 8	5 6 9	3 1 3 1 3 4 6 4 5 9	3 1 3 1 3 4 5 9	5 3 7 8	4 5 6 7	2	1 5 6 7 8	4 5 6 7	4 6 7 8	5 6 9	3 1 3 1 3 4 6 4 5 9
9	4 5 6 8	4 5 6 8	5 6 8	2	4 6 8	5 6 4 6	7	9	9	4 5 6 8	4 5 6 8	2	5 6 8	4 6 8	5 6 4 6	7

先来看左图，左图框起来的 18 个单元格统称为**交叉线单元格**（有时候也简称**交叉单元格**，**Crossline Cells**）。右图给出的两个单元格叫做**基准单元格**（**Base Cells**），一般都有两个单元格构成，而且可以通过示例里发现，交叉线单元格里推导所涉及到的数字，在基准单元格

里都找得到，而且基准单元格里也没有其它多余的候选数情况。

1 2 3 5 7 8	2 3 5 6 7	1 3 5 6 8	4	5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 7 8 9	2 3 5 8 9	1 2 3 5 6 7	4	5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 7 8 9	2 3 5 8 9
4	2 3 5 6 7	3 5 6	2 5 6	8	9	1	2 3 6 5	2 3 2 3	1 2 3 5 6 7	4	2 3 5 6 7	3 5 6	2 5 6	8	9
2 5 7 8	9	5 6 8	2 5 6	3	1	2 5 6	2 4 6 4 5 7 8	2 8	1 2 3 5 6 7	2 5 7 8	9	5 6 8	2 5 6	3	1
1 2 3 4 9	2 3 4	1 3 4	1 2 9 7 8	5	2 3 7 9	1 2 3 7 8 9	6	1 2 3 7 8 9	1 2 3 4 9	2 3 4	1 3 4	1 2 9 7 8	5	2 3 7 9	1 2 3 7 8 9
1 2 5	8	1 5 9	3	6 7 9	2 6	4	1 2 7 9	1 2 9	1 2 5	8	1 5 9	3	6 7 9	2 6	4
6	2 3 4	7	1 2 8	1 4	2 9	5	1 2 3 8 9	1 2 3 8 9	1 2 3 6 4	7	1 2 8	1 4	2 9	5	1 2 3 8 9
3 5 7	1	4 5 6 7	9	4 5 6 7	3 4 6	8	2 3 4 6 4 5 7	2 3 8	3 5 7 8	1	4 5 6 7	9	4 5 6 7	8	2 3 4 5
5 7 8	3 4 5 6 7	2	1 5 6 7 8	1 4 5 6 7	3 4 6	5	3 1 3 1 3 5 6 4 6 4 5 9	3 1 3 1 3 5 6 4 6 4 5 9	5 7 8	3 4 5 6 7	2	1 5 6 7 8	3 4 5 6 7	8	2 3 4 5
9	4 5 6 8	3 4 5 6 8	1 5 6 8	2	4 6 8	5	3 1 3 5 6 4 6 8	3 1 3 5 6 4 6 8	9	4 5 6 8	3 4 5 6 8	2	1 5 6 8	3 4 6 8	7

左图给出的两个单元格是最终我们确定一个 a 和一个 b 的地方，它们被称为**目标单元格 (Target Cells)**，一般也是有两个的；而右图给出的，目标单元格两侧的绿色的四个单元格称为**镜面单元格 (Mirror Cells)**。好像镜面单元格在之前的示例里并没有真正用到，所以它用得很少，不过有用到的时候我们会提及到。

1 2 3 5 7 8	2 3 5 6 7	1 3 5 6 8	4	5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 7 8 9	2 3 5 8 9
4	2 3 5 6 7	3 5 6	2 5 6	8	9	1	2 3 6 5	2 3 2 3
2 5 7 8	9	5 6 8	2 5 6	3	1	2 5 6	2 4 6 4 5 7 8	2 8
1 2 3 4 9	2 3 4	1 3 4	1 2 9 7 8	5	2 3 7 9	1 2 3 7 8 9	6	1 2 3 7 8 9
1 2 5	8	1 5 9	3	6 7 9	2 6	4	1 2 7 9	1 2 9
6	2 3 4	7	1 2 8	1 4	2 9	5	1 2 3 8 9	1 2 3 8 9
3 5 7	1	4 5 6 7	9	4 5 6 7	3 4 6	8	2 3 4 6 4 5 7	2 3 8
5 7 8	3 4 5 6 7	2	1 5 6 7 8	1 4 5 6 7	3 4 6	5	3 1 3 1 3 5 6 4 6 4 5 9	3 1 3 1 3 5 6 4 6 4 5 9
9	4 5 6 8	3 4 5 6 8	1 5 6 8	2	4 6 8	5	3 1 3 5 6 4 6 8	3 1 3 5 6 4 6 8

最后再来看这个图，能够受到基准单元格填数影响的这几个单元格被称为**逃逸单元格 (Escape Cells)**。它也用得比较少，只是在推导过程里一般会提及到而已。

还有一些其它的术语词，比如**邻伴单元格**（简称**伴单元格**，**Companion Cells**）和它的**镜面单元格 (Companion's Mirror Cells)**，不过这些我们都在具体用得到的时候才提及，因为它们用得非常少。

下面我们再来看一则示例。

3-1-3 3 数 JE (3-Digit JE)

2 4 5 8	2 4 6 7	2 4 5 6 7 8	1 2 6 7 8 9	1 6 7 8	2 4 5 8 9	1 2 4 8	3
2 3 4	2 3 4	1	2 3 7 8 9	3 7 8 9	5	6	2 4 9
2 3 5	9	2 5 6 8	1 2 3 6 8	4	1 3 6 8	2 5 8	7
1 2 3 4	1 2 3 4	2 6 8	1 3 6 7 8	1 3 6 7 8	9	2 3 4 7	5
1 2 3 4	1 2 3 4	2 6 9	1 3 6 7 8	5	1 3 6 7 8	2 3 4 9	8
1 3 8 9	5	6 8 9	4	1 3 6 7 8	2	3 1 6 7 9	1 6 9
1 4 5	8	4 5 7	1 3 6 7	2 6 7	1 3 4 5 7	9	4 5 6 7
2 4	2 4	3	5	6 7 8 9	4 6 7 8	1	2 4 6 7
6	1 2 4	2 4 5 7	1 3 7 8 9	1 3 7 8 9	2 3 4 5 7 8	2 3 4 8	2 4 5 7

如图所示，这一则示例比之前那一个示例简单一些的是，基准单元格里面只有 1、3、6 三种数字。

逻辑类似地，我们可以发现 **r123789c258** 里面能放下最多两个 1、两个 3 和两个 8，所以在 **r456c258** 里，至少得有一个 1、一个 3 和一个 8。

我们假设基准单元格里放的是 **a** 和 **b** 两种不同的数字（显然，**a** 和 **b** 是 1、3、8 的其二），那么，如果放进去后，**r5c46** 就形成了关于 **a** 和 **b** 的数对，使得 **r456c258** 里最终能放 **a** 和 **b** 的位置只剩下 **r4c2** 和 **r5c8** 两个单元格。

因为 **r4c2** 和 **r5c8** 两个单元格只能是 **a** 和 **b** 的关系，而 **a** 和 **b** 又必须是 1、3、8 的关系，所以这两个单元格显然是不能放入其它和 1、3、8 无关的数，所以删除掉它们，故 **r4c2 <> 24**。

这个例子推导起来比较简单和轻松，因为基准单元格里只有三种数字，而最初的示例是四种数字。但不管是多少种数字，推导的逻辑都是大致上一样的，因为在推导交叉线单元格里的时候，数字之间是没有什么关系的。

下面我们来看一种惊人的示例，它涉及五种不同的数字。

3-1-4 5 数 JE (5-Digit JE)

2 3 4 6	4 6	5	1 3 4 6	1 6 9	7	2 3 6 9	1 3 6 9	8
8	1	3	3 5 6	2	3	4	5 6 7 9	3 5 6 7 9
9	4 6 7	4 7	8	1 5 6	1 3 4	2 5 6 7	1 3 5 6	2 3 5 6
1 2 7 8 9		6	1 2 7	1 7 8 9	5	3	7 8 9	4
1 5 7	3	1 7 9	1 6 7	4	1 8 9	5 6 7 8 9	2	5 6 7 9
2 4 5	4 5 7 8 9	4 7 9	2 3 6 7	6 8 9	2 3 8 9	1	5 6 7 8 9	5 6 7 9
1 3 4 5	4 5 9	8	1 2 4 5 7	1 5 7	6	2 5 7 9	4 5 7 9	2 3 7 9
4 5 6 7	2	4 9 7	4 5 7	3	4 8 9	5 6 7 8 9	4 5 6 7 8 9	1
7	4 5 6	1 3	9	1 5 8	1 2 4 8	2 5 6 8	3 4 5 6 8	2 3 5 6

如图所示，这一则示例我想要让你自己来尝试推理，我不给出解释。它涉及 5、6、7、8、9 五种数字，而删数是图上已经给出了的 r2c9(3)。

3-1-5 第二类标准 JE (JE Type 2)

下面我们再看看第二种类型的 JE，它和第一类的 JE（之前说到的都是第一类的）有什么区别。

9	8	2 3 4 5	7	1 2 3 5	1 2 3 5	6	1 2 4	1 2 4
7	1 3 4	2 3 4	6	1 2 3 8 9	1 2 3 8 9	5	1 2 4	1 2 4 8 9
2 5 7	1 5 6	2 5 6	2 8 9	4	1 2 5 8 9	2 7 8	3	1 2 7 8 9
6	4 7 9	3 8	5	1 2 3 7	1 2 3 4	9	1 2 4 7	1 2 3 4 7
4 5 7	3 2	4 5 7 9	3 4	1 3 6 8 9	1 3 6 8 9	3 4	1 4 5 6 7 8	1 4 5 6 7 8
4 5 7	3 4	5 1	2 3 4 8 9	2 3 6 8 9	2 3 6 8 9	2 3 7 8	2 4 5 6 7 8	2 3 4 5 7 8
1	4 7 9	3 4 7 9	2 3 4 8 9	2 3 6 8 9	2 3 6 8 9	2 3 7 8	2 4 5 6 7 8	2 3 4 5 7 8
2 3 4 8	3 4 5 7	2 3 4 5 7	1 8 9	2 3 5 8	2 3 5 8	2 3 7 8	9	6
2 3 4 8	3 4 5 7	2 3 4 5 7	2 3 5 8	5	7	1	2 4 8	2 3 4 8

如图所示，观察到基准单元格里涉及 2、3、4、8，我们就尝试在交叉线单元格里找 2、3、4、8 的填数情况。

显然，r123456c147 里，能放下最多两个 2、3、4，而 8 却在 r356 都有，这是否意味着 8 是例外，可以出现三次呢？

仔细观察一下 8 的分布情况，我们发现 8 只能在 c47 里出现，所以实际上也算只能出现最多两个。所以这一次我们不能横着看了，需要竖着看。

这样我们就确定了 2、3、4、8 只能最多两次的情况。接着后面的逻辑就一样了，设基准单元格填入 a 和 b，然后得到目标单元格 r7c4 和 r8c1 是 a 和 b，所以不能涉及 2、3、4、8 以外的数字，删掉它们。

我们再给两则示例，希望你能独立理解。

9	8	1 2 3	7	1 2 3	1 2 3	6	1 3 1 2 3	9	8	1 2 3	7	1 2 3	1 2 3	6	1 3 1 2 3
5	2 3	1 2 3	2 3	1 2 3	1 2 3	3	1 3 1 2 3	5	2 3	1 2 3	2 3	1 2 3	1 2 3	3	1 3 1 2 3
1 2	2 3	7	2 3	6	4	9	7 8 9	1 2	2 3	7	2 3	6	4	9	7 8 9
7	6	4	6	8	1 2 3	5	1 2 3	7	6	4	6	8	1 2 3	5	1 2 3
3	2 5	9	2 5 6	1 2 5 6	1 2 5	5	1 3 1 3	3	2 5	9	2 5 6	1 2 5 6	1 2 5	5	1 3 1 3
1 4	6	1 5	8	1 4 5	1 4 5	2	1 3 1 3	1 4	6	1 5	8	1 4 5	1 4 5	2	1 3 1 3
1 2	2 1 2	2	2 5 6	3	1 2	5	1 1	1 2	2 1 2	2	2 5 6	3	1 2	5	1 1
7 8	7	7 8	4 5 6	9	7 8	7 8 9	7 8 9	7 8	7	7 8	4 5 6	9	7 8	7 8 9	7 8 9
4	2	1 2 3	2 3	2	4 5	6	4 5 3	4	2	1 2 3	2 3	2	4 5	6	4 5 3
7 8	7 8	7 8	9	9	4 5	9	7 8 9	7 8	7 8	7 8	9	9	4 5	9	7 8 9
4	4 5	5	1	4 5	4 5	6	4 5 3	4	4 5	5	1	4 5	4 5	6	4 5 3
7 8	7	9	7 8	9	9	9	7 8 9	7 8	7	9	7 8	9	9	9	7 8 9
2	2 3	2 3	2 3	2 3	2 3	3	3	2	2 3	2 3	2 3	2 3	2 3	3	3
4	6	4 5	9	5 6	4 5	7	8	4	6	4 5	9	5 6	4 5	7	8

认真对比两个示例，实际上它们是同一个题，只是恰好有两处删数而已。这个示例的逻辑还会在后续提到。

下面我们来看一些有关于 JE 的变体。

3-1-6 共轭对类型 JE (JE With Conjugate Pair)

	3		3		3										
5 6	4 5	4	6	5 6	4 5										
7 8	8 9	7	9	7 9	8 9	7 8									
1	1		1 2	1 2	1 2										
5 6	4 5	4	6	5 6	4 5										
7 8	8 9	7	9	7 9	8 9	7 8									
1	1		1 2	1 2	1 2										
5 6	5	2	3	5 6	5										
7 8	8 9	9	8	8 9	7 8										
2 3	2 3	1	8	2 3	2 3										
7	4	9	6	6 4	7										
2 3	6	4	3	1 2	7	1 2 3									
5		9	4	4	4 5	8									
2 3	2 3		3	1 2	1 2 3	9									
5	4 5	4		4	5 6	7									
7 8	8	7		6	7	9									
1 2 3	1 2 3	8	5	1 2 3	1 2 3										
6		9	7	9	7										
9	1 2 3		3	1 2	4	1 2 3									
4	7	5		7	7 8	5									
			1 2	1 2 3	6	1 2 3									
			9	8 9	6	9									

如图所示，最初的推理逻辑都是一样，发现基准单元格里涉及 6、7、9，所以我们就在交叉线单元格里去看 6、7、9 的分布，发现都是最多出现两次。所以 r347c789 里，6、

7、9 都是最少有一次出现。

然后发现，基准单元格填入的是 6、7、9 的其二，假设为 a 和 b，那么 a 和 b 最终就只能放到 r4c8 和 r7c89 里。显然，三个单元格是足够放下一个 a 和一个 b 了，而且还有一个额外的数字，随意是多少。但实际上并不是这样。

我们仔细一点就会发现，r7 上 r7c89 出现了 4 的共轭对，这便使得最终 4 必须放在 r7c89 的其一里，所以实际上，在 r4c8 和 r7c89 里，有一个 a、一个 b 和一个 4，而且必须在 r7c89 里，a 和 b 的位置还是很紧张的。所以，这三个单元格里和 6、7、9 无关的其余候选数都能被删除掉。

这一个示例把目标单元格从两个变为了三个，因为它还带有一个共轭对的使用。

我们再来看两则这样的示例。

1 2 3	1 3	7	1 2 3	5	9	1 2 3	6	1 2 3
5	6	1 2 3	1 2 3	1 2	2 3	1 2 3	1 2 3	1 2 3
8	4	1 2 3	6	1 2	2 3	1 2 3	1 2 3	1 2 3
1 3	1 3	8	5	2	2 3	4	1 2 3	1 2 3
1 3	2	1 3	3	4	3	1 3	7	1 3
4 6	5	3	2 3	2	2 3	2 3	2 3	8
1 2 3	1 3	6	7	1 2	2	9	8	1 2 3
1 2 3	1 3	4	9	1 2	2	5	1 2 3	1 2 3
1 2	1	1 2	1 2	3	1 2	1 2	1 2	7

这一个示例的逻辑和刚才的差不多，所以这一个就自己理解。

下面我们来看一个更为奇特的示例。

4 7 8	4 7 8	3	1 2 4 5 7 8	1 2 4 5 7 8	9	1 2 4 7 8	6	1 2 5 7 8
5	4 6 7 8 9	4 6 7 8 9	1 2 3 4 5 7 8	1 2 4 6	1 2 6 7 8	1 2 3 4 7 8 9	1 2 3 4	1 2 3 7 8
2	1	4 6 7 8 9	4 5 7 8	4 5 6 7 8	5 6 7 8	4 7 8 9	3 4 5	5 7 8
4 6 8	2 4 5 6 8	4 5 6 8	1 2 4 5	1 2 4 5 6	3	1 2 8	7	9
4 3 7 8	2 3 4 5 7 8	1	1 2 4 5	2 5	2 5	6	2 3 5	2 3 5 8
3 6 7 9	2 3 5 6 7 9	5 6 7 9	1 2 5 7	8	1 2 5 6 7	1 2 3 5	1 2 3 5	4
1 6 7 8	5 6 7 8	5 6 7 8	1 2 5 8	3	4	1 2 7	9	1 2 6 7
1 3 4 6 9	3 4 5 6 9	4 5 6 9	1 2 5	7	1 2 5	1 2 3 4	8	1 2 3 6
1 3 4 7 8	3 4 7 8	2	6	9	1 8	5	1 3 4	1 3 7

如图所示，这一则示例带有两个共轭对。首先我们依然还是可以看到，交叉线单元格里 4、7、8 的填数位置都最多只有两个，所以在 r123c347 里，4、7、8 都是最少有一个出现。

假设基准单元格填入 4、7、8 的其二是 a 和 b，那么根据 a 和 b 的数对的结构排除，我们得到了 r23c47 里必须要放下 a 和 b。而四个单元格里，c4 出现了 3 的共轭对，而 c7 出现了 9 的共轭对，且都在刚才说到的 r23c47 里，所以这四个单元格里必须是一个 a、一个 b、一个 3 和一个 9，而且 3 只能放在 r23c4 里，9 只能放在 r23c7 里。所以，和 4、7、8 无关的数字都能被删除，因为位置是刚好够的，容不下其它数字的放入。

3-1-7 只有一个基准单元格的 JE

3-1-7-1 使用示例

9	2 3 5 6	2 4 6	1 2 3 4 5 6	8	1 3 4 5	4 5 6	1 4 5 6	7
3 4 5 6	1	7	3 4 5 6	3 4 5 6	3 4 5 6	4 5 6 8 9	2	5 8
2 4 5 6	2 5 6	8	1 2 4 5 6	1 2 4 5 6	1 4 5	3 4 5 6	1 4 5 6	5
2 4 5 6	2 5 6	3	4 5 8	4 5 8	4 5 8 9	1	4 5 6 7 8	2 5 8
1 4 5 6	5 6 7 9	1 4 6	1 3 4 5	1 3 4 5	2	4 5 6 7 8	3 4 5 6	5 8
8	2 5	1 2 4	1 3 4 5	7	6	4 5 8 9	3 4 5	9
1 2 3	2 3	1 2	9	1 2 3	1 3 4	2 4 5	1 3 5	6
1 2 3 6	4	1 2 6 9	1 2 3 7 8	5	1 3 7 8	2 7 8 9	1 3 7 8 9	1 2 3 8
7	2 3 8 9	5	1 2 3 8	6	1 3 8	2 8 9	1 3 8 9	4

之前我们就说过，涉及的数字的种类数的多少都无所谓，所以我们最少可以到多少呢？

	2 3	6	4	1 2	5	1 2	1 2 3	1 2 3
7 8 9	7 8 9			7 8 9	7 8	5	5	5
1	2	4	2	2	6	3	9	7
	8	8	5	8			4 5 6	
4	5	3	1 2	1 2	6	1 2	1 2 3	1 2 3
7 8 9	7 8 9	7 8 9	7 8 9	7 8 9	7 8	4 6	4 6	4 6
4 5 6	6	4 5	2	2	4 6	3	1 2	1 2
7 8 9	7 8 9	7 8 9	7 8	7 8	7 8	4 5 6	4 5 6	4 5 6
2	3	3	3	3	1	4 5 6	8	9
7	6	4 5	7	4 6	7	7		
4 6	3 1 3	2 3	5	9	1 2	1 2	1 2	
7 8	7 8	7 8	7 8	7 8	4 6	4 6	4 6	6
5 6	4	5	1 3	1 3	5	1 2	1 2 3	8
7 9	7 9	7 9	7 9	7 9	7	5 6	5 6	9
3	6	5	1	1	2	1	1	1
7 8 9	7 8 9	7 8 9	7 8 9	7 8 9	7 8 9	4 5 6	4 5 6	4 5 6
5	1	2	6	3	4 5	4 5	7	3
8 9	8 9		4	8 9	8		4 5	

3-1-8 目标单元格同侧的 JE

1 2 3	1 2 3	7	9	5	8	1 2 3	3	1 2
4	6	4	6	4	6	4	6	4
5	4	1 2	2	1 2 3	1 3	1 2 3	3	1 2
		9	6	6	6	7	8 9	7 8
6	8	1 2 3	2	7	1 3	5	3	1 2
		9	4	4	4		9	
1 2	1 2	1 2	3	1	1	2	2	2
8 9	5	5 6	6	6	4 5 6	4 6	4 5 6	4 5 6
3	5	5 6	4 5 6	6	2	4 6	1	4 5 6
7 8 9	7 9	7 8	7 8	7 8	7 8	7	7 8	7 8
1 2 3	1 2 3	4	5 6	9	1	8	3	2
7	7	7	7	7	7	7	5 6	5 6
4	5	5	1	3	5	4	2	4 5 6
7 9	7 9	7 9	7 9	7 9	7 9	7 9	7 9	7 9
1 2 3	1 2 3	8	2	4	5 6	9	1	5 6
7	7	7	7	7	7	7	7	7
1 2	6	1 2	2	2	5	1	4 5	3
4	4	5	5	8	7	4	8	
7 9	7 9	7 9	7 9	7 9	7 9	7 9	7 9	

这则示例的逻辑和之前的示例都差不多，只是唯一的一处特点是，它的目标单元格位于同一侧。

3-1-9 数组类型 JE (JE With Subset)

<div>2 4</div>	<div>1 4</div>	5	<div>1 2 3 4</div>	6	7	<div>1 2 3 4</div>	8	9
<div>2 4 6 8</div>	9	<div>1 2 3 4</div>	<div>1 2 3 4 8</div>	5	<div>1 2 3 4</div>	<div>7 4</div>	<div>1 3 6 4</div>	<div>1 2 3 4 6</div>
<div>2 4 6 8</div>		<div>1 2 3 4</div>	<div>1 2 3 4 8 9</div>	<div>2 4</div>	<div>1 2 3 4</div>	<div>1 2 3 4</div>	<div>1 3 6</div>	5
1	<div>5 6 4</div>	<div>2 4 6 8</div>	<div>2 4 6 8</div>	<div>2 7 8</div>	9	<div>4 5 7</div>	<div>5 4</div>	<div>3 8</div>
7	<div>5 6 4</div>	<div>4 9</div>	<div>1 4 6 8</div>	3	<div>1 4 6</div>	<div>1 4 5 9</div>	2	<div>1 4 8</div>
<div>2 4 9</div>	3	8	5	<div>2 7</div>	<div>1 2 4</div>	6	<div>1 7 9</div>	<div>1 4</div>
<div>5 8 9</div>	<div>7 8</div>	<div>7 9</div>	<div>2 3 6 9</div>	1	<div>2 3 5 6</div>	<div>2 3 9</div>	4	<div>2 3 6</div>
3	2	<div>1 9</div>	<div>6 9</div>	4	8	<div>1 5 9</div>	<div>1 5 6 9</div>	7
<div>4 5 9</div>	<div>1 4</div>	6	7	<div>2 9</div>	<div>2 3 5</div>	8	<div>1 3 9</div>	<div>1 2 3</div>

如图所示，根据最初的 JE 的推理过程，我们最终由于 $r1c12$ 是 a 和 b （其中 a 和 b 是 1、2、4 的其二）的关系，得到 $r2c4$ 和 $r3c47$ 里必须有一个 a 和一个 b 。我们算上 $r3c5$ 可以发现， $b2$ 里能放 8 和 9 只有 $r2c4$ 和 $r3c45$ 这三个单元格。如果 $r23c4$ 都没有 8 和 9，那么 $b2$ 就只有一处可以放 8 和 9，而显然一个单元格是容不下两处填 8 和 9 的要求的，所以这样是不行的；如果 $r23c4$ 都是 8 和 9 的话，根据 JE 的要求， a 和 b 又无法正常放到 $r2c4$ 和 $r3c45$ 里了，因为这三个单元格必须保证一个 a 和一个 b ，换句话说，这三个单元格里必须有两个单元格是 1、2、4，所以这么填也不行。

所以，最终我们就只能让 $r23c4$ 里有一个单元格是 8 或者 9。那么， $b2$ 里就必然会产生一个关于 8 和 9 的隐性数对结构，是 $r23c4$ 的其一和 $r3c5$ 构成的。这样一来，我们就能保证的是， $r23c4$ 里有一个单元格是 8 或者 9，而另外一个单元格则是 1、2、4 的其一，而且 $r3c5$ 只能是 8 或 9 其一， $r3c7$ 只能是 1、2、4。所以删除掉与这些数字无关的其余候选数。

3-2 额外删数的推论

我们可以从之前的示例里看到，它们的推导过程很清晰，但总感觉少了一些什么。下面我们来看看这些引起我们思考的地方。

3-2-1 同宫定理 (Same-Box Theorem)

第一个我们要介绍的内容就是同宫定理。同宫定理是一种神奇的定理，但使用范围有限。我们先来看看它到底是怎么用的。

1 5 8	1 5 8	7	1 5 6 8 9	1 5 6 8 9	4	1 5 3 8 9	2	3 5 6 8 9
9	1 2 4 5 8	1 2 4 5 8	1 2 5 6 8	1 2 3 5 6	2 3 6	7	1 3 4 6	4 5 6 8
6	3	1 2 4 5 8	7	1 2 5	2 8 9	1 5 8 9	1 4	4 5 8 9
1 2 3 5 7	1 2 5 6	1 2 3 5	2 5 6 9	8	2 3 6 7 9	1 2 3 9	1 3 4	2 3 4 7 9
4	2 6 8	2 3 8	2 6 9	2 3 6 7 9	1	2 3 8 9	5	2 3 7 8 9
1 2 3 5 7 8	1 2 5 8	9	4	2 3 5 7	2 3 7	6	1 3 7	2 3 7 8
1 2 5 8	1 2 5 8 9	6	3	1 2 7 9	2 7 8 9	4	2 7 9	2 5 7 9
1 2 3	7	1 2 3 4	1 2 6 9	1 2 4 6 9	5	2 3 9	8	2 3 6 9
2 3 5 8	2 4 5 8 9	2 3 4 5 8	2 6 8 9	2 4 6 7 9	2 6 7 8 9	2 3 5 9	3 6 7 9	1

如图所示，这个结构是一个标准的 JE，我们已经不用再去叙述逻辑了，删数就是这里浅红色 r2c4(26)和 r3c7(9)。现在我们尝试来看看它还能得到什么结论。

我们尝试观察 b5689 里的所有 1、5、8 的确定值的分布。我们发现 1 和 8 是同宫的，而 5 是单独处于 b6 和 b8 的。同宫定理是：**假设基准单元格里涉及的数字是 a、b、c 三种不同的数字，那么去尝试找交叉线单元格所属的六个宫里，和两个基准单元格不同行列的剩下四个宫里，a、b、c 的确定值的分布。如果其中两种数字处于同宫的分布，那么基准单元格里就一定是一组数字，即，基准单元格就是这两个数字构成的数对形式。**

这句话有些绕，我们一步一步分析。首先，1、5、8 就是这个定理说到的 a、b、c 三种数字，然后观察交叉线单元格分属的六个宫 b456789，然后发现，基准单元格所在行列包含 b47，所以剩下我们要观察的四个宫是 b5689。然后发现 1、5、8 的确定值里，1 和 8 是同宫的，所以基准单元格就是关于 1 和 8 的数对。于是，删除掉 r1c12(5)。

这个定理非常好用，但具有一定的局限性。目前从定理来看，首先基准单元格必须是两个，而且必须涉及的是三种不同数字，如果是四个甚至更多，是都不行的。而且就目前来看，我们无法给出这个定理的证明。而且，使用上也有局限性。

5 6 7	5 6	1	5 6 7 9	3	8	2 5 6 7 9	2 4 5 7 9	2 4 6 7 9
4	9	5 6 7 8	2	1 7	5 6	5 6 7	3	6 7 8
3 5 6 8	2	5 6 7 8	5 6 7 9	4 7 9	4 5 6 9	1	5 7 8 9	6 7 8 9
3 5 6 8 9	5 6 7	2	1	7 8 9	5 9	4	7 8 9	3 6 7 8 9
3 6 4 8 9	6	3 7 8	3 7 8 9	2 7 8 9	2 3 9	2 3 6 9	1	5
1 3 5 4 8 9	1 3 7	3 7 8	5 7 8 9	6	2 3 4 5 9	2 3 7 9	2 7 8 9	2 3 7 8 9
7	8	9	4	5	1 2	2 3	6	1 2 3
1 2 5 6	1 5 6	4	3	1 2 9	1 2 6 9	8	2 5 7 9	1 2 7 9
1 2 3 5 6	1 3 5 6	3 5 6	6 8 9	1 2 8 9	7	2 5 9	2 4 5 9	1 2 4 9

如图所示，我们发现 5 和 7 同宫（b8），但这个示例依旧不能直接使用。我们所认为的同宫，必须是对角分布都需要存在同宫的要求。比如这一则示例里，b8 是同宫的，但与之对角的 b6 却不是 5 和 7 同宫的；而 b6 却只有 5 没有 7。所以这一个示例无法使用同宫定理得到结论。

1 5 8	1 5 8	7	1 5 6 8 9	1 3 5 6 8 9	4	1 3 5 8 9	2	3 5 6 8 9
9	1 2 4 5 8	1 2 4 5 8	1 2 5 6 8	1 2 3 5 6	2 3 6 8	7	1 3 4 6 4 5 6 8	
6	3	1 2 4 5 8	7	1 2 5 9	2 8 9	5 8 9	1 4 9	4 5 8 9
1 2 3 5 7	1 2 5 6	1 2 3 5	2 5 6 9	8	2 3 6 7 9	1 2 3 9	1 3 4 7 9	2 3 4 7 9
4	2 6 8	2 3 8	2 6 9	2 3 6 7 9	1	2 3 8 9	5	2 3 7 8 9
1 2 3 5 7 8	1 2 5 8	9	4	2 3 5 7	2 3	6	1 3 7	2 3 7 8
1 2 5 8	1 2 5 8 9	6	3	1 2 7 9	2 7 8 9	4	7 9	2 5 7 9
1 2 3	7	1 2 3 4	1 2 6 9	1 2 4 6 9	5	2 3 9	8	2 3 6 9
2 3 5 8	2 4 5 8 9	2 3 8	2 6 8 9	2 4 6 7 9	2 6 7 8 9	2 3 9	3 6 7 9	1

对比之前的示例，1 和 8 是同宫的，而处于对角的两个宫 b5 和 b9 都是同宫的，所以它才满足了同宫定理的要求。

所以，这一点很重要。

至于这个定理的论证，我们不会在这里提到，因为它的思路比较复杂。我们还有一个定理，它的逻辑和论证将在后面给出，而且它恰好包含了同宫定理的内容。

3-2-2 T 数对定理 (Target Pair)

1 2 3 5 7 8	2 3 5 6 7	1 3 5 6 8	4 5 6 7	ab 5 6 7	2 6 7	2 3 5 6 7 9	2 3 6 7 8 9	2 3 5 8 9
4 5 6 7 8	alb 2 3 7	3 5 6 8	2 5 6 7	8 5 6 7	9 5 6 7	1 5 6 7	2 3 6 7 8	2 3 5 8
2 5 7 8	9 5 6 8	5 6 8	2 5 6 7	3 5 6 7	1 5 6 7	alb 2 3 7	2 6 7 8	2 4 5 8
1 2 3 4 9 7 8	2 3 4 9	1 3 4 9	1 2 4 7 8	5 7 9	2 3 7 9	1 2 3 7 8 9	6	
1 2 5	8 5 9	1 5 9	3 6 7 9	2 6 7	4 7 9	1 2 9	1 2 9	
6 4 2 3	7 4 2 3	1 2 8	1 4 9	2 8	2 3 9	5 8 9		
5 7	1 4 5 6	3 4 5 6	9 4 5 6	4 7	3 6	8 4 6 5	2 3 4 5	2 3 5 6
5 7 8	3 4 5 6 7	2 4 5 6 7 8	1 5 6 7 8	1 4 5 6 7	3 4 6 7 8	3 5 6 9	1 3 4 6 9	1 3 4 5 9
9 4 5 6 8	3 4 5 6 8	3 4 5 6 8	1 5 6 8	2 4 6 8	3 5 6 8	1 3 4 6 7	2 3 5 6 7 9	7

第一个最容易想到的推论，就是目标单元格的跨区数对了。由于最后两处的填数是不同的，所以两个单元格只能一个 **a** 一个 **b**，它们便形成了关于 **a** 和 **b** 的跨区数对，只是删数我们还不知道，因为 **a** 和 **b** 是未知数。如果我们后续的推导里可以得到 **a** 和 **b** 究竟是什么的时候，我们就可以顺理成章地得到跨区数对的删数了。

因为同宫定理目前不能使用在涉及四种数字的情况下，所以我们也无法使用同宫定理来进行删数。但是，如果只有三种数，那么就可以得到结论了：

4 5 8	4 5 8	1	4 6 7 8	3 9	4 6 7 9	2 9	5 6 7 8 9	2 5 6
2 9	3 9	2	1 6 7 8	1 7 8	5 7 8	1 7 8	6 7 8	4
6 4 5 8	7 4 5 8	7	1 2 4 8	2 9	1 4 9	3 5 8 9	1 2 5	
1 3 4 5 9	4 5 6 4 5 6 9	3 4 5 6 9	1 3 4 5 7	1 7 9	8 7	1 4 7	2 7	1 3 6
1 2 3 4 5 8	2 4 5 6 8	2 3 4 5 6 8	1 2 3 4 5 7	1 2 4 7	1 3 4 7	1 7 8	1 3 6 7 8	9
1 2 3 4 8 9	7 4 8 9	2 3 4 8 9	1 2 3 4 8	6 4 9	1 3 4 9	5 8	1 3 7	1 3
4 5 7 9	4 5 6 4 5 6 9	3 4 5 6 9	1 3 4 6 7	1 7	2 7	1 7 9	1 3 5 9	8
2 3 5 7 8	2 5 6 8	2 3 5 6 8	9 7 8	1 7 8	1 3 7	1 2 7	4 7	1 2 3 5
2 3 4 7 8 9	1 4 8 9	2 3 4 8 9	3 4 7 8	5 4 7	3 4 7	6 7 9	3 7	2 3

如图所示，我们可以利用同宫定理得到的是，基准单元格里是 2 和 9，那么根据 JE 的理论，我们还可以得到的是目标单元格里也是 2 和 9，因为基准单元格是 **a** 和 **b**，那么目标单元格里也是 **a** 和 **b**，而且填数还不一样。

那么，由于是 2 和 9 的跨区数对的关系，r1c46 和 r3c89 就显然不能放其它的 2 和 9

了，删掉它们。

这个定理跟目标单元格有关，而目标单元格 **target** 的首字母是 T，所以称为 **T 数对 (Target Pair)**。

我们再来看一则比较难理解的例子。

9	8	1 2	7	1 2 3 4 5	1 2 3 4 5	6	1 2 3 5	2 3 5
5	1 7	1 2 6 7	9	1 2 3 8	1 2 3	4	1 2 3 7 8	2 3 7 8
1 2 7	4	3	1 2 5 8	1 2 5 8	6	1 2 8	9	2 5 7 8
6	1 3 9	1 2 8 9	5 8	7	1 2 3 5	1 2 3 8 9	1 2 3 4 5 8	2 3 4 5 8 9
1 2 3 8	5	4	1 2 3 8	6	9	7	1 2 3 8	2 3 8
1 2 3 7 8	1 3 7 9	1 2 7 8 9	4	1 2 3 5 8	1 2 3 5	1 2 3 8 9	1 2 3 5 8	6
4	1 3 7 9	1 7 8 9	6	1 2 3 9 7	1 2 3	5	2 3 7 8	2 3 7 8 9
3 7 8	2	5 6 7 8 9	3 5	3 4 5 9 7	3 4 5 9 7	3 8 9	3 4 6 7 8	1
1 3 7	1 3 6	1 3 5 6 7 9	1 2 3 5	1 2 3 4 5 9	8	2 3 9	2 3 4 6 7 9	2 3

如图所示，我们可以根据基本的 JE 的删数形式得到目标单元格的删数。不过还不够，此时我们着重去看伴单元格 **r4c1** 和 **r6c4** 的镜面单元格 **r4c23** 和 **r6c56**。

9	8	1 2	7	1 2 3 4 5	1 2 3 4 5	6	1 2 3 5	2 3 5
5	1 7	1 2 6 7	9	1 2 3 8	1 2 3	4	1 2 3 7 8	2 3 7 8
1 2 7	4	3	1 2 5 8	1 2 5 8	6	1 2 8	9	2 5 7 8
6	1 3 9	1 2 8 9	5 8	7	1 2 3 5	1 2 3 8 9	1 2 3 4 5 8	2 3 4 5 8 9
1 2 3 8	c 5	4	d	6	9	7	a/b	1 2 3 8
1 2 3 7 8	a	7 9 7 8 9	4	1 2 3 5 8	1 2 3 5	1 2 3 8 9	1 2 3 5 8	6
4	1 3 7 9	1 7 8 9	6	1 2 3 9 7	1 2 3	5	2 3 7 8	2 3 7 8 9
3 7 8	2	5 6 7 8 9	3 5	3 4 5 9 7	3 4 5 9 7	3 8 9	3 4 6 7 8	1
1 3 7	1 3 6	1 3 5 6 7 9	1 2 3 5	1 2 3 4 5 9	8	2 3 9	2 3 4 6 7 9	2 3

我们发现，**r5** 仅剩下四个单元格。我们小范围使用代数法，假设 **r5c89** 是 **a** 和 **b**，那么 1、2、3、8 的剩下两个数假设为 **c** 和 **d**，此时只能放在 **r5c14** 里，我们可以随意假设 **c** 和 **d** 的位置，反正都是一样的。

a 和 **b** 也是一样，目标单元格有两个，随便哪一个是 **a** 都行，这并不影响后续的推导。那么此时，我们可以得到上图这样的填数模式。显然，我们必然得到了 **r4c4** 是 **b**，而 **r5c1**

是 c, r6c1 是 a, 而发现 r4c23 里只有 1、2、3、8 和额外的数字 9。因为 a、b、c 都出现了, 那么这两个单元格里只能放下一个 d 和一个 9, 否则的话, 两个单元格只剩下 1、2、3、8, 而其中三种数都不能放入 r4c23 里, 所以必须让 9 放进去, 故 r4c23 里存在一个 9 区块; 同理, 下面的 r6c56 也应当通过类似的结论得到含有 5 区块的结论。

显然, 由于 5 区块和 9 区块的成立, 图中对应的区域里自然就可以删除掉这些数字, 这里就不画出来了。

3-2-3 T 邻/镜面单元格定理 (Mirror Theorem)

和目标单元格有关的还有一个定理, 我们称为 T 邻定理, 或镜面单元格定理, 因为它跟镜面单元格有关。

3-2-3-1 原理

2 4 5 8 9	2 4 5 8 9	1	2 4 6 7 8	3	4 6 7 9	2 9	5 6 7 8 9	2 5 6
2 9	3	2 9	1 2 6 7 8	5	1 2 7 8 9	1 7 8 9	6 7 8 9	4
6	2 4 5 8 9	7	1 2 4 8	2 9	1 4 9	3	1 5 8 9	1 2 5
1 3 4 5 9	2 4 5 6 9	3 4 5 6 9	1 3 4 5 7	8	1 4 7	2 7	1 3 6	
1 2 3 4 5 8	2 4 5 6 8	2 3 4 5 6 8	1 2 3 4 5 7	1 2 4 7	1 3 4 7	1 7 8	1 3 6	9
1 2 3 4 8 9	7	2 3 4 8 9	1 2 3 4	6	1 3 4 9	5	1 3 8	1 3
3 4 5 7 9	2 4 5 6 9	3 4 5 6 9	1 3 4 6 7	2	1 7	1 3 7 9	5 7 9	8
2 3 5 7 8	2 5 6 8	2 3 5 6 8	9	1 7 8	1 3 7	1 2 7	4	1 2 3 5 7
2 3 4 7 8 9	1	2 3 4 8 9	3 4 7 8	5	3 4 7	6	3 7 9	2 3

如图所示, 我们通过之前的定理可以确定的删数, 能做到这里。不过, 我这次把镜面单元格也涂上了颜色, 这意味着镜面单元格也有结论, 不过是什么结论呢?

我们知道基准单元格一定是 2 和 9 了, 那么 r1c7 和 r3c5 就一定是 2 和 9, 不过究竟哪一个单元格是 2, 哪一个是 9 我们目前还不知道。不过请思考一下。假如 r1c7 填入的是 2, 那么 r3c5 就是 9, 那么会如何呢?

如果 r1c7 是 2, 而 r2c13 里必有一个 2, 这就使得了 r3c5 的两个镜面单元格 r3c46 必须有一个是填 2 的, 因为 2 在 b2 里只剩下这两个单元格可以填入 2 了; 同理, 由于 r3c5 是 9, 所以 b3 里此时能放 9 的位置只有 r1c89, 所以 r1c7 的镜面单元格 r1c89 里必须有一个 9。

这就是 T 邻定理的内容: **如果我们确定了基准单元格一定是填入 a 和 b 的, 而某一个目标单元格填入了数字 a, 那么它自己的镜面单元格就必须有一个单元格是填入 b 的; 或者把 a 和 b 换过来也可以。**

那么, 这个定理能不能反过来用呢? 当然可以。根据逆否命题, 如果镜面单元格里不包含我们需要的数字 a 和 b 的话 (取原命题的反面情况), 那么对应的目标单元格里就不能说

明填入的数字一定是 **a** 和 **b**，此时的 **JE** 就不一定成立了。因为原命题和逆否命题等价，所以这样理解是完全可以的。

这个内容有什么用处呢？我们来看一些示例。

3-2-3-2 示例 1：带死锁区块的 JE

9	8	1 2	7	1 2 3 4 5	1 2 3 4 5	6	1 2 3 5	2 3 5
5	1 7	1 2 6 7	9	1 2 3 8	1 2 3	4	1 2 3 7 8	2 3 7 8
1 2 7	4	3	1 2 5 8	1 2 5 8	6	1 2 8	9	2 5 7 8
6	1 3 9	1 2 8 9	1 2 3 5 8	7	1 2 3 5	1 2 3 8 9	1 2 3 4 5 8	2 3 4 5 8 9
1 2 3 8	5	4	1 2 3 8	6	9	7	1 2 3 8	2 3 8
1 2 3 7 8	1 7	3 9	1 2 7 8 9	4	1 2 3 5 8	1 2 3 8 9	1 2 3 5 8	6
4	1 7	3 9	1 7 8 9	6	1 2 3 9	1 2 3 7	5	2 3 7 8 9
3 7 8	2	5 6 7 8 9	3 5	4 5 9	4 5 9	3 8 9	3 4 6 7 8	1
1 3 7	1 6	3 5 6 7 9	1 2 3 5	1 2 3 4 5	8	2 3 9	2 3 4 6 7 9	2 3

如图所示。仔细观察这个结构，它有点不像是之前熟悉的 **JE** 结构，因为基准单元格里含有的数字 **9** 在交叉线单元格里有确定值的出现。我们之前的推导逻辑里，基准单元格里涉及的所有数字在交叉线单元格里都没有确定值的出现的。

显然，这个结构的数字 **9** 出现了这样的情况，我们就无法当作 **JE** 来使用，不过我们可以换一种思考方式：正难则反。假设基准单元格 **r4c23** 里不含有候选数 **9** 的话，那么 **JE** 就明显是成立了，因为 **1、2、3、8** 四种数字在交叉线单元格里最多都只能出现两个，所以在 **c147** 的剩余部分 **r456c147** 里，**1、2、3、8** 都最少需要出现一次。假设 **r4c23** 填入的是 **a** 和 **b**（此时 **a** 和 **b** 是 **1、2、3、8** 的其二，且不相同），那么我们可以顺理成章地得到目标单元格 **r5c4** 和 **r6c7** 里都只能是 **a** 和 **b**。而 **1、2、3、8** 显然就是这里 **a** 和 **b** 的候选情况，所以自然就删掉和 **1、2、3、8** 无关的数字。

不过可以从图上看，**r6c7** 似乎把 **1、2、3、8** 这些需要的数字全删掉了，说明上述推导有错。那么错误在哪里呢？错误在于，我们说过，如果 **JE** 成立，那么就可以尝试使用 **T** 邻定理来得到镜面单元格的结论。显然，如果 **r5c4** 填入 **a**，而 **r6c7** 填入 **b** 的话，那么 **r5c4** 的镜面单元格 **r5c56** 里就必须有一个是数字 **b**（且 **b** 一定是 **1、2、3、8** 的其一）。不过可以从图上看，**r5c56** 都是确定值，且是 **6** 和 **9**，跟 **1、2、3、8** 全部无关。这就说明了矛盾的出现：原本我们要求 **JE** 成立后，一定会得到镜面单元格必须是 **1、2、3、8** 的其一，结果却出现的是 **6** 和 **9**，这显然不对，所以原假设错误，即 **r4c23** 里必须有一个 **9**，即形成了 **9** 区块。

那为什么删除的数字是 **r6c7(1238)** 呢？显然，我们可以通过 **9** 的确定值看出，**b6** 此时能放 **9** 的位置只有 **r6c7**，所以 **r6c7 = 9**。

类似于这种使用的，还有一些示例。

3-2-3-3 示例 2：另一则带死锁区块的 JE

1 3 4	2	1 3 8 9	1 3 4 6 8	5	3 6 8	1 3 4 6 7 9	1 3 4 6 8	1 3 4 6 7 8 9
1 3 4	5	6	7	1 3 4 8	9	1 3	2	1 3 4 8
7	1 3 4	1 3 8 9	1 2 3 4 6 8	1 2 3 4 8	2 3 6 8	5	1 3 4 6 8	1 3 4 6 8 9
2	1 3 8 9	1 3 5 8 9	1 5 8	6	4	1 3 9	7	1 3 8 9
6	1 3 4 7 8	1 3 7 8	1 2 8	9	2 7 8	1 2 3 4	5	1 2 3 4 8
1 4 8 9	1 4 7 8 9	1 5 7 8 9	3	1 2 8	2 5 7 8	1 2 4 9	1 4 8	6
1 3 8 9	1 3 6 7 8 9	4	2 5 6 8 9	2 3 8	2 3 5 6 8	1 2 3 6 7	1 3 6	1 2 3 5 7
5	1 3 6 8	1 2 3 8	2 4 6 8	7	2 3 6 8	1 2 3 4 6	9	1 2 3 4
3 9	3 6 7 9	2 3 7 9	2 4 5 6 8 9	2 3 4	1	8	3 4 6 7	2 3 4 5

如图所示。我们尝试假设 **r1c46** 没有候选数 **6**，那么交叉线单元格就不需要看 **6** 的相关填数位置，此时 **1、3、4、8** 都最多出现两次，所以 **r123c158** 里至少出现一次 **1、3、4、8**。而假设 **r1c46** 填入 **a** 和 **b**（此时 **a** 和 **b** 是 **1、3、4、8** 的其二，且不相同），那么最终 **a** 和 **b** 只能放在 **r2c1** 和 **r3c8** 里。

不过根据 T 邻定理，我们可以看到 **r2c1** 的镜面单元格含有 5 和 6 的确定值，但它们跟 1、3、4、8 无关，所以出现矛盾。所以，原本假设错误，即 **r1c46(6)** 区块是成立的，故我们可以利用这个区块，得到 **r3c8 = 6** 的结论。

3-2-3-4 示例 3：只有 T 邻定理删数的 JE

1 4 7	2 3 9	1 2 3 9	4 2 3 9	5	6	1 2 7	8	2 9
4 6 9	5	1 2 6 9	7	1 2 4	1 4	1 2 6	4	3
1 4 7	2 3 6 9	8	2 3 4	1 2 3 4	1 3 4	1 2 5 6	1 2 4 5	2 4 5 6 9
2	3 6 8	1 5 6	4 3 8	1 3 4 6 7 8	1 3 4	9	1 3 4 5	4 5 8
1 6 8 9	4	1 3 6 9	5	1 3 6 8	1 3 8	1 2 3 8	1 2 3	7
1	3 7 8	1 3 5 7	4 3 8	9	2	1 3 5 8	6	4 5 8
3	2 6 7 8 9	4	2 6 8 9	2 6 7 8	5 7 8 9	2 5 6 8	2 5 9	1
5	2 6 7 8 9	2 6 7 9	1	2 3 6 7 8	3 7 8 9	4	2 3 9	2 6 8 9
6 8 9	1	2 6 9	2 3 4 8 9	2 3 4 6 8	3 4 5 8 9	2 3 5 6 8	7	2 5 6 8 9

如图所示，我们可以通过基本的推理逻辑得到目标单元格 **r7c4** 和 **r8c9** 填入的是 2、6、8、9 的其二，不过此时依然没有删数。不过我们不能气馁，因为这个例子隐藏了一个巨

大的 T 邻定理的使用方法。

可以发现，假如我们设 **r9c13** 填入 **a** 和 **b**，且 **r7c4** 是 **a**、**r8c9** 是 **b** 的话，那么根据 T 邻定理可以得到，**r8c8** 一定是 **a**，而 **r7c56** 里有一个是 **b**。那么，**r8c8** 为啥一定是 **a** 呢？因为 **r8c9** 的镜面单元格 **r8c78** 里必须有一个 **a**，这是 T 邻定理规定的结论；而其中的 **r8c7** 是 4 的确定值，而它并不在 2、6、8、9 其中，所以能放 **a** 的地方就只剩下 **r8c8** 了。

所以，**r8c8** 是 **a**。换言之，**r8c8** 不应当是 2、6、8、9 外的数字，删掉它们。此时又可以发现，**r8c8** 填入的是 **a**，根据代数法的原理，原本假设为 **a** 的地方，就必须是填入 **a** 的，所以目标单元格 **r7c4** 就一定也是这些候选数。**r8c8** 只有 2 和 9，那么 **r7c4** 也只能有 2 和 9，删除掉 6 和 8。

3-2-3-5 示例 4：需要 T 邻定理删数的第二类 JE

9	8	1 2 3 4 5	7	1 2 3 4	2 3 4	6	1 2 3 4	1 4 5
7	1 2 3 5	1 2 3 4 5	1 2 3 4	1 2 3 4 6 8	2 3 4 8	1 3 4 8	9	1 4 5 8
1 2 3 4	1 2 3 4	6	1 2 3 4 9	5	2 3 4 8 9	1 3 4 7 8	1 2 3 4 8	1 4 7 8
1 2 5 6 8	4	1 2 5	1 2 9	1 2 6 7 9	2 5 6 7 9	1 7 8 9	1 8	3
1 3 5	1 3 5 9	7	8	1 3 4	4 5 9	2 9	6	1 4 9
1 2 3 6 8	1 2 3 6 9	1 2 3 4 9	1 2 3 4 9	1 2 3 4 6 7 9	2 3 4 6 7 9	5	1 4 8	1 4 7 8 9
1 3 4	1 3 7	9	6	4 7 8	3 4 7 8	1 3 4 8	5	2
1 2 3 4 6	1 2 3 6	8	5	2 3 4 9	2 3 4 9	1 3 4 9	7	1 4 6 9
2 3 4 5 6	2 3 5 6	2 3 4 5	2 3 4 9	2 3 4 7 8 9	1	3 4 8 9	3 4 8	4 6 8 9

如图所示，它的逻辑和前面这一则示例推导逻辑完全一样，不过用到的是第二类 JE 的论证出现至少两次的逻辑。推导只有这一点不同，所以这里就不再给出示例的说明了，请你自己分析。

需要注意的是，删数一定要是同步的。比如 T 邻定理得到的候选数是怎么样的，可以根据代数法代入回原本的目标单元格，得到候选数一致的结论，这一点别漏掉了。

3-2-3-6 示例 5：依赖于区块和三数组反推的 JE

1 2 5 6 7	2 3 5 6 7	1 2 3 5 7	4 7 8	1 3 5 6 7 8	1 6 7 8	5 6 8 7 8	5 6 8 7 8	9 1 6 7 8
4 7 9	5 6 7	1 5 6 7	1 7 8	1 6 7 7 8	9 7 8	2 5 6 7	3 5 6 7	4 1 6 7 8
1 2 5 7 9	8 4 5 7 9	1 3 5 7 9	1 2 3 7 8	1 4 7 8	3 4 5 7 8	5 6 8 9 7 8	1 4 5 8 9	2 1 8 7 8
8 4 7	3 4 7	1 3 4 7	5 7 8	9 4 6 7	1 4 6 7	3 4 6 7	1 4 6 7	2 4 6 7
1 2 5 9	2 3 4 5 9	1 2 3 4 5 9	1 2 6 8	1 4 6 8	1 4 6 8	3 4 5 6 8 9	7 4 5 6 8 9	1 3 6 8
3 4 7	2 4 6 7	2 4 7	9 3 6 7	1 4 6 7 8	1 4 6 7 8	4 6 7 8	2 4 6 8	5 3 6 7
5 6 7 9	4 5 6 7 9	8 4 5 7 9	3 6 7	4 6 7	2 4 6 7	1 4 6 7	4 6 9 7	3 6 7
2 6 7 9	1 4 7 9	2 4 7 9	3 6 7 8	3 4 6 7 8	5 4 6 7 8	3 4 6 8 9	2 4 6 8 9	3 6 7 8

如图所示，我们发现交叉线单元格里含有数字 5，为了 JE 结构可以成立，我们尝试忽略它，假设基准单元格 r3c78 不含有候选数 5，看看会如何。

此时发现，由于忽略了 5 的影响后，JE 是可以成立的，所以可以得到的是目标单元格 r1c1 和 r2c4 只能填入和 r3c78 一样的两种数字（假设为 a 和 b，则此时 a 和 b 是 1、6、7 的其二）。

接着，我们发现，由于 r3c78 没有候选数 5 了，所以 r3c13(5)区块成立，所以 b1 其余位置都不能放 5；而且，根据 T 邻定理可以得到，r2c5 此时是只能填入和 r1c1 一样的数字的，不管是什么数，但可以肯定的是，此时 r1c1、r2c45 和 r3c78 都只能是 1、6、7 了。不过，问题出现了。由于刚才 5 区块的成立，使得 r2c23 不能放入 5，此时 r2c2345 只有 1、6、7 三种数字，但需要填入四个单元格，这显然是不合逻辑的，出现了矛盾。所以原本的假设错误，即 r3c78(5)区块应当是成立的。所以删除掉对应的删数，即 r3 和 b3 其余位置的 5。

3-2-3-7 示例 6：同时有共轭对和死锁区块的 JE

1 2 3	1 3	7	1 2 3	5	9	1 2 3	6	1 2 3
5	6	1 2 3	1 2 3	1 2	2 3	1 2 3	1 2 3	1 2 3
8	4	1 2 3	1 2	2 3	1 2 3	1 2 3	1 2 3	1 2 3
1 3	1 3	8	5	2	2 3	4	1 2 3	1 2 3
1 3	2	1 3	3	4	4 6	4 6	7	1 3
4 6	3	5	3	2 3	2	2 3	2 3	8
1 2 3	1 3	6	7	1 2	2	9	8	1 2 3
1 2 3	1 3	4	9	1 2	2	5	1 2 3	1 2 3
1 2	1	1 2	1 2	3	1 2	1 2	7	

如图所示，首先我们可以通过 JE 直接得到目标单元格内的删数。接着我们可以观察 r3c7 的镜面单元格 r3c9 有什么结论。

我们按照期望先假设 r23c7 填入的是 b 和 7（7 是共轭对导致的），而 r2c4 假设填入的是 a。那么 r23c7 究竟哪一个是 7，哪一个是 b，我们不清楚，所以分两种情况分析。

假如 $r2c7 = b$ ，那么此时形成的目标单元格就位于同一侧了，而此时 r1c12 这一对基准单元格填入的肯定是 a 和 b，所以我们把 a 提出来，针对于 b3 作出排除，可以发现 a 只能放到 r3c89 里，而 b3（或者说 r3）上存在 5 的共轭对，这意味着 r3c89 里必须有一个是 a，另外一个 5。而 r3c89 填入的数字跟 a（1、2、3 其一）和 5 以外的数字无关，所以可以删除掉它们；

假设 $r3c7 = b$ ，那么此时更容易地确定到 a 的填数位置在 r3c89，因为这里可以直接利用 T 邻定理得到结论。所以照样 r3c89 是 a 和 5，跟其余的候选数无关，删除掉它们。

3-2-4 三链列/剑鱼定理（Swordfish Theorem）

当然，还是前面这个示例那道题。除了我们可以删除这些以外，还有一些删数，是通过三链列得到的，所以这个定理称为**剑鱼定理**或者**三链列定理**（Swordfish Theorem）。

4 5 8	4 5 8	1	2 7 8	3 6 9	4 6 7 9	2 9	5 6 7	2 5 6
2 9	3	2 9	1	1	5	1	1	6
6	4 5 8	7	1 2 4 8	2 9	1	3 5 8 9	1	2 5
1 3 4 5 9	4 5 6 9	4 5 6 9	1 3 4 5 7	1 2 3 4 5 7	8	1 3 4 5 7	2 6	1 3 6
1 2 3 4 5 8	2 4 5 6 8	2 3 4 5 6 8	1 2 3 4 5 7	1 2 3 4 5 7	1 3 4	1 3 4	1 3 6	9
1 2 3 4	7	4	1 2 3 4	6	1 3 4	5	1 3 8	1 3
4 5 7 9	4 5 6 9	4 5 6 9	1 3 4 6 7	2	1 3 5 7 9	8	1 3 5 7 9	8
2 3 5 7 8	2 5 6 8	2 3 5 6 8	9	1 3 7 8 9	1 3 6	4	1 2 3 5	1 2 3 7
4 2 3 7 8 9	1	4	2 3 4 8 9	5	4 3 7	6	3 2 3 7	2 3

如图所示，图上给出了关于 2 和 9 的三链列结构。注意， $r1c7(2)$ 和 $r3c5(2)$ 是可以
通过弱关系连起来的，同理， $r1c7(9)$ 和 $r3c5(9)$ 也是一样。为什么呢？因为它们不同真，
同真就违背了 JE 的原则——目标单元格的填数是不同的。那么，形成剑鱼后，我们就可以
通过 2 和 9 找到各自额外的删除域的删数，至于 $r1c7$ 和 $r3c5$ 弱关系连起来的对应区域，
我们已经通过跨区数对删除了，就是跨区数对可以删掉的那些 2 和 9。所以这个定理能够删
除的数字如下。

4 5 8	4 5 8	1	4 6 7 8	3 9	4 6 7	2 9	5 6 7 8 9	2 5 6
2 9	3	2 9	1	6	5	1	1	6
6	4 5 8	7	1 2 4 8	2 9	1	3 5 8	1	5
1 3 4 5 9	4 5 6 9	4 5 6 9	1 3 4 5 7	1 2 3 4 5 7	8	1 3 4 5 7	2 6	1 3 6
1 2 3 4 5 8	2 4 5 6 8	2 3 4 5 6 8	1 2 3 4 5 7	1 2 3 4 5 7	1 3 4	1 3 4	1 3 6	9
1 2 3 4	7	4	1 2 3 4	6	1 3 4	5	1 3 8	1 3
4 5 7 9	4 5 6 9	4 5 6 9	1 3 4 6 7	2	1 3 5 7 9	8	1 3 5 7 9	8
2 3 5 7 8	2 5 6 8	2 3 5 6 8	9	1 3 7 8 9	1 3 6	4	1 2 3 5	1 2 3 7
4 2 3 7 8 9	1	4	2 3 4 8 9	5	4 3 7	6	3 2 3 7	2 3

显然，这个定理依旧需要先得到跨区数对的结论。

3-2-5 X 致命定理 (Bi-Bi Pattern)

下面来说一个得到比较困难的定理：**X 致命定理 (Bi-Bi Pattern)**。这个定理将同宫定理
进行了推广，但证明逻辑就会麻烦一些。

这个定理的英文名里的 Bi 是数字 2 的含义的前缀，读作[baɪ]。

3-2-5-1 使用

2 5 6 7 8	1 4 5 6 8 7	1 2 4 5 6 7	2 6 4 8 7	2 6 4 7 8	1 2 4 6 7 8	3 7	9
2 6 4 7 8 9	4 6 8 9 7	2 4 6 7	2 3 6 4 8 9 7	2 3 6 4 7	1 4 6 7	2 4 6 7	5
2 6 4 7 9	1 4 6 9	3 3	2 6 5 9	5 6 7 9	4 6 7 9	8	1 2 4 6 7
2 3 5	3 4 5	8	1 3 5	9 5	3 5 7	1 2 3 4 5 7	1 2 4 6 7
3 5 6 9	7	4 5 6	1 3 5 6 8	1 3 6	2 5 6 7 8	1 3 4 5 9	1 3 4 8 9
1 5 6 9	3 5 6 9	2 5 6	4 7	3 6	3 5 6 7 8	2 3 5 7 9	2 7 8 9
3 6 7	1 6 7	3 6	9	1 2 3 6	8 4 6 7	1 2 3 4 7	1 2 3 4 7
3 5 7 8	2 5 7	1 5	1 3 5 9	1 3 4	3 4 5 9	6	1 3 4 7 8 9
4 5 6 8	1 5 6 8	3 5 6	1 5 6	2 3 6	3 5 6 9	1 2 3 9	1 2 8 9

如图所示，我们只看 JE 的构型，可以通过基本的推导思路得到 $r7c9 \neq 3$ 的结论。因为这里我们着重解释该定理，所以我们就暂时不管这个删数，去看基准单元格到底有什么删数。

下面我们为了方便描述和表达，我们给出 X 区域的定义：

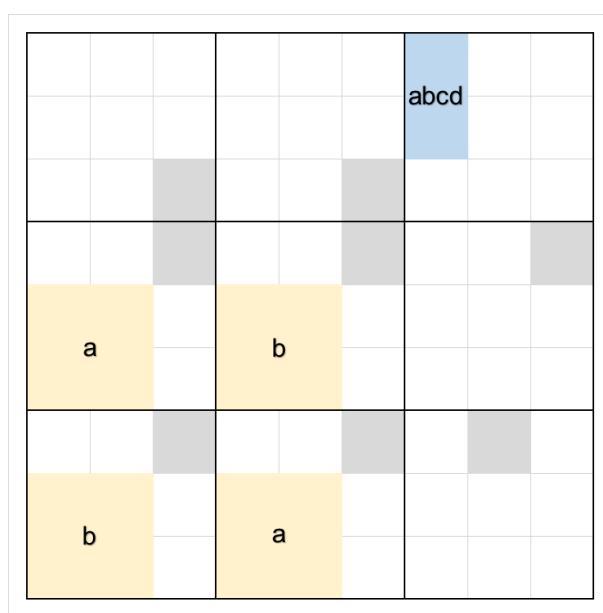
X 区域 (X-Region/Dual Region)：交叉线单元格里，以确定值为轴心，与交叉格分部方向正交的单元格排开，并排开基准格和交叉格能够直接对应的两个宫外的剩下四个宫的格子外，剩下的 16 格。例如图示里，所有浅红色（含深绿色和深青色）的 16 格为一个 X 区域。

该定理如下：**X 区域里，如果某两种数字的提示数能出现在对角的两个宫里，那么这两个数就一定不能组合产生于基准格里。**举个例子，数字 7 出现于对角两个宫，而另外一个对角宫里 2 和 4 都是对角出现的，则就使得如果 7 出现到组合里，那另外一格填数必须不能是 2 或 4，否则 2、7 和 4、7 组合均满足推论的要求，这就会使得基准单元格一定不允许为 2、7 和 4、7 组合。

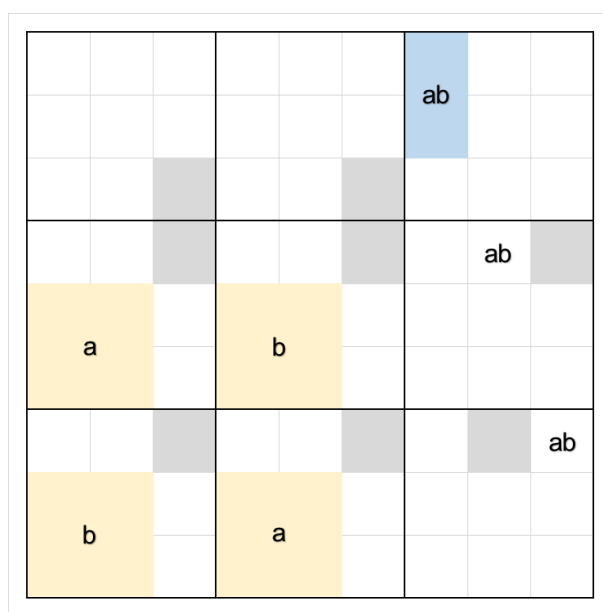
接着因为基准格 $r12c7$ 里有一个单元格含有候选数 1，而另外一格又没有这个数字。如果这一个单元格填入 7，则另外一格由于没有 1 的关系，只剩下 2 和 4 的填数情况，这必然形成了 2 和 7 以及 4 和 7 的组合。所以这必然是不允许的。所以假设错误，故 $r1c7$ 不能填入 7。

这个定理用起来非常神奇，而且非常匪夷所思，它把确定值和候选数层面直接关联起来了，而且也有固定的删数模式和原理。下面我们来看看这个定理如何证明。

3-2-5-2 证明



如图所示，我们构造出简图。为了满足题目所证明的需求，我们就构造出 X 致命原则的主要形成条件：**a** 和 **b** 各自得在对角宫出现，并且不是相同的对角两宫；基准格就假设为 **a**、**b**、**c** 和 **d** 即可（假设 **a**、**b**、**c** 也是类似的证明）。



现有如图的推理情况的其一。我们继续向下推导。由于 **r4c8** 只能填 **a** 和 **b** 了，所以该行里 **r4c12** 和 **r4c45** 里，**a** 和 **b** 的填数位置显然不可同真（即 **r4c12(b)** 和 **r4c45(a)** 形成弱关系）。注意，**a** 和 **b** 的提示数此时已经在 X 区域里表示到图上了，所以图中的 **a** 和 **b** 的位置显然分别只能填入到 **r4c45** 和 **r4c12** 里面了。

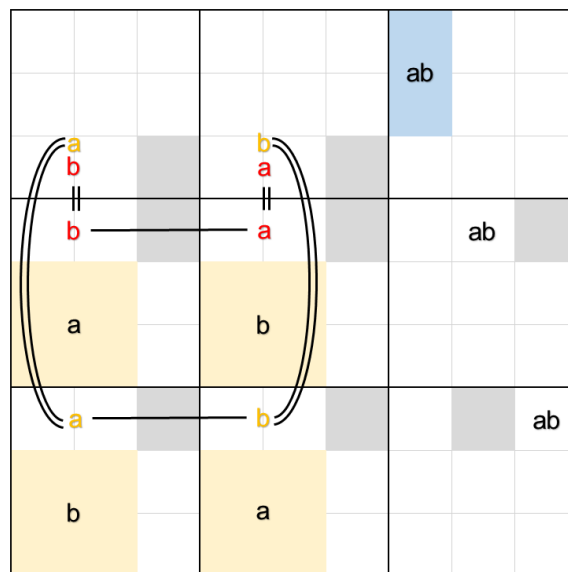
同样地，下面的 **r7c12** 和 **r7c45** 也有此类似的关系。此时我们使用链表示出来：

The diagram shows a 4x4 grid illustrating the construction of a Latin square. The grid is divided into four 2x2 quadrants. The top-left quadrant contains a 2x2 grid of yellow cells with 'a' and 'b' in the bottom row. The top-right quadrant contains a 2x2 grid of white cells with 'ab' in the top-right cell. The bottom-left quadrant contains a 2x2 grid of yellow cells with 'b' and 'a' in the bottom row. The bottom-right quadrant contains a 2x2 grid of white cells with 'ab' in the bottom-right cell. A horizontal line connects the 'b' in the top-left quadrant to the 'a' in the top-right quadrant. A horizontal line connects the 'a' in the bottom-left quadrant to the 'b' in the bottom-right quadrant. The grid is surrounded by a thick black border.

此时继续观察该图。由于下面 **r89c45** 里有一个 **a**，而上面 **r4c45** 里也能有 **a**，那请问 **r34c45** 四格里 **a** 的位置能有几个呢？是不是只有一个了！交叉格里只能有最多两个 **a**，上面 **r4c45** 里如有 **a**，那上面的 **r3c45** 就不能有 **a**，不然 **r34c45** 都有 **a** 的话，就一定会有一个 **a** 和下面 **r89c45** 里的 **a** 的提示数重复，违背要求。因此 **r4c45(a)** 和 **r3c45(a)** 不可同为真；另外一方面，能不能同为假呢？如果同为假的话，那 **a** 要保证恰好两个的地方只有 **r3c12** 和 **r7c12** 了。之所以此处说恰好两个而不是至少两个的原因很简单：结论一个 **a** 和一个 **b** 的结果已经得到，这使得此时填入 **a** 的位置在交叉格里必须得是两个，毕竟 **r4c8** 和 **r7c9** 里已经有一个 **a** 了。

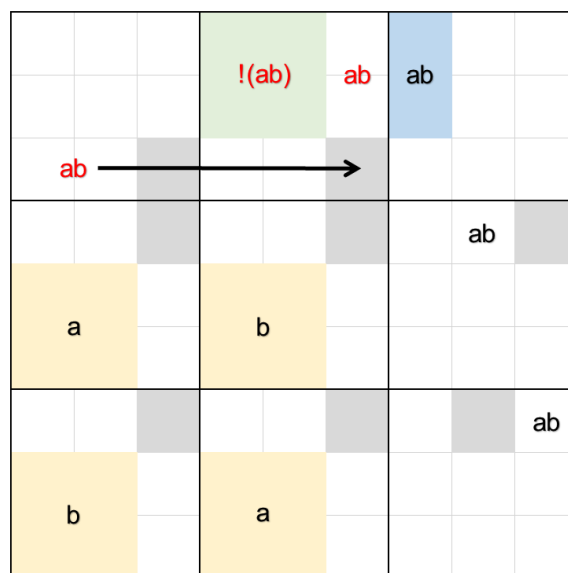
那既然 **r37c12** 里有 **a** 的话，那可不可能填两个呢？显然不能。因为 **r37c12** 同时都可以“对应到”**r56c12** 的 **a** 的提示数上，如果 **r37c12** 同为 **a**，显然会有一个 **a** 会和 **a** 的提示数矛盾。这下，**r3c45(a)** 和 **r4c45(a)** 更不能同为假。既然不能同为真，也不能同为假，那么 **r3c45(a)** 和 **r4c45(a)** 只能是一个真和一个假（即它同时满足强关系也同时满足弱关系）。

继续。我们此处为了能够继续推理，我们使用 **r3c45(a)**和 **r4c45(a)**的强关系，为了能把链串起来。刚才的结论只能得到这么一点结论吗？显然不是，我们来看看真正能连上的位置：



所有强弱关系的证明方式，都可以从刚才的证明逻辑之中得到，这里就不一一证明了。

试想一下。这样构成了两条链，一条是“内线”，一条是“外线”。 b 假得 a 真， a 假又得 b 真，又恰好涉及的是同样的两个部分 $r3c12$ 和 $r3c45$ ，那么这就只可能说明一点： a 和 b 是挨着的，而不是让 $r3c12$ 和 $r3c45$ 一边一个。既然 a 和 b 是挨着的，我们就可以粗略地叙述为“ a 和 b 在此处的填数是同宫的”。如果同宫，则表示这里形成了显然的 a 和 b 的数对。那么形成数对又会怎么样呢？



如图所示，假设我们的 a 和 b 的数对在第一个宫挨着的地方，根据排除效果，显然 $r3c6$ 就不能是 a 和 b 了；而根据刚才的原理， $r12c45$ 此时是不可以为 a 和 b 的。那此时 a 和 b 就只可以填到 $r12c6$ 里。至此 $r12c6$ 形成 a 和 b 的数对。

此时再结合上 $r12c7$ 的 a 和 b ，此时 $r12c67$ 形成关于 a 和 b 的唯一矩形的致命形式，所以矛盾了。证毕。

如果你把这个证明逻辑套用到三个数的 JE 结构，同宫定理是适用的，原理可通过 X 致命定理的原理直接证明得到。

2 4 5 8	2 4 6 7	2 4 5 6 7 8	1 2 6 7 8 9	1 6 7 8 9	1 6 7 8	2 4 5 8 9	1 2 4 8	3
2 3 4 8	2 3 4 7	1	2 3 7 8 9	3 7 8 9	5	6	2 4 8	2 4 9
2 3 5 8	9	2 5 6 8	1 2 3 6 8	4 6 8	1 3 6 8	2 5 8	7	1 2 5
1 2 3 4 8	1 2 3 4 6 8	2 4 6	1 3 6 7 8	1 3 6 7 8	9	2 3 4 7	5	1 2 4 6 7
7	1 2 3 4 6 9	2 4 6 9	1 3 6 9	5	1 3 6 9	2 3 4 9	1 2 3 4 6	8
1 3 8 9	5	6 8 9	4	1 3 6 7 8	2	3 7 9	1 3 6 7	1 6 9
1 4 5	8	4 5 7	1 3 6 7	2 6 7	1 3 4 6 7	3 4 5 7	9	4 5 6 7
2 4 9	2 4 7	3	5	6 7 8 9	4 6 7 8	1 4 6 7	2 4 6 8	2 4 6
6	1 2 4 7	2 4 5 9	1 3 7 8 9	1 3 7 8 9	1 3 4 7 8	2 3 4 5 7 8	2 3 4 8	2 4 5 7

1、3、6 里，1 和 3 的组合，以及 1 和 6 的组合是不可以同时出现的，否则形成上述的致命形式（当然也可能中途就没办法填数导致死亡了），所以三个数的组合只剩下 3 和 6 是可以的，而正是因为组合只能是 3 和 6 的关系，这就看起来好像 3 和 6 恰好同宫，就可以用了，所以这便通过 X 致命定理简单地证明了同宫定理。

下面我们来看一些例子。

3-2-5-3 一些使用示例

5 6 7	5 6 7	1	5 6 7 9	3	8	2 5 6 7 9	2 4 5 7 9	2 4 6 7 9
4	9	5 6 7 8	2	1 7	5 6	5 6 7	3	6 7 8
3 5 6 8	2	5 6 7 8	5 6 7 9	4 7 9	4 5 6 9	1	5 7 8 9	6 7 8 9
3 5 6 8 9	5 6 7	2	1	7 8 9	5 9	4	7 8 9	3 6 7 8 9
3 6 4 6 8 9 7	3 7 8	3 7 8 9	2 7 8 9	2 3 7 8 9	2 3 9	2 3 6 9	1	5
1 3 1 3 5 4 5 8 9 7	3 5 7 8	3 5 7 8 9	6	2 3 4 5 9	2 3 7 9	2 7 8 9	2 3 7 8 9	2 3 7 8 9
7	8	9	4	5	1 2	2 3	6	1 2 3
1 2 5 6	1 5 6	4	3	1 2 9	1 2 6 9	8	2 7 9	1 2 7 9
1 2 3 5 6	1 3 5 6	3 5 6	6 8 9	1 2 8 9	7	2 5 9	2 4 5 9	1 2 4 9

如图所示，我们发现，5 和 6 是不能形成组合放到 r1c12 里的，因为 b5689 里 5 和 6 的确定值的分布构成了 X 形状，这样就保证了 X 致命定理的出现。所以 r1c12 不能是 5 和 6 的数对。

那么, $r1c2 \neq 56$ 就很好理解了。假设 $r1c2 = 5$, 则 $r1c1$ 只能放入 5 或者 6, 使其形成 5、6 数对, 必然出错; $r1c6 = 6$ 时也是一样。所以 $r1c12 \neq 56$ 。

1	4 5 7	2 3 4 7	3 4 6 8	2 5 6 8	3 4 6 8	5 6 7 8	5 7 8	9
4 5 9	3 4 5	3 4 5	6 7	1 5 8 9	1 3 4 8 9	1 5	2 5 8	3
2 3 5 7	8	3 7 9	1 2 6 9	1 2 3 5 6 9	1 3 6 9	4 1 7	5 5 6	3
2 4 8 9	1 2 4	1 8 9	1 4 6 8 9	7 5	1 2 6 9	3 4 6 8	2 4 6 8	6
3 4 7 8 9	1 3 4 7	5 7 8 9	1 4 6 8 9	2	1 6 7 9	1 4 7 8 9	4 6 7 8	6
2 4 7 8 9	6	1 4 7 8 9	3 4 7 8 9	1 8 9	1 4 8 9	1 2 5 7 9	1 4 5 7 8 9	2 4 5
3 4 5 7	9	1 3 4 7	1 2 6	1 2 3 6 7	1 3 6	8 4 5 7	2 3 4 5 7	2 3
6 7	5 7	3 7 8	2 8 9	4 7 8 9	3 7 8 9	2 3 5 7 9	5 7 9	1
4 7 8	3 4 7	1 3 4 7	2 5	1 3 8 9	1 3 7 8 9	3 7 9	6 4 7	3

如图所示。r5c8(7)删数的原理就不再赘述了。我们发现, b1379 里, 9 和 1、8 组合都会形成 X 致命形式, 所以基准单元格 r6c56 里不能是 1、9 数对和 8、9 数对。

如果 $r5c6 = 9$, 那么 $r5c5$ 此时只能放入 1 或 8, 而这都是不行的, 所以 $r5c6 \neq 9$ 的结论就直接通过这个定理得到了。

3-2-6 其它结论

最后我们来看看其它结论。这些结论一般不能直接用来删数, 而需要使用链技巧嵌套使用, 下面我们拿出三则示例, 给大家阐述 JE 里会形成哪些结论。

3-2-6-1 示例 1: 利用邻伴镜面单元格得到的弱关系

2 3 4 8	1 2 3 4	1 2 3 4 5	9	2 3 4 5 6 7	1 2 3 4 6 7	1 2 3 5 6	8
6 7	4 8	1 2 3 4 5	7	1 2 3 4 5 6 8	1 2 3 4 5 6 8	9 5	1 2 3 4 5 6 8
5	1 3 4 8 9	1 2 3 4 8 9	7	1 2 4 6 8	2 3 4 6 8	1 2 3 4 6	1 3 6
2 3 4 9	3 4 5 9	7 6	1 5	5 9	8 1 5 9	1 3 5 9	1 5 9
1	5 6 9	5 9	8 7	4 7	3 6 7	5 6 7 9	2
2	5 6 8 9	2 5 8 9	3	2 5 7 9	1 6 7	4 7 9	1 5 6 7 9
3 7 9	3 5 9	2 3 5 9	8	2 3 5 6 7	1 2 3 6 7	1 2 3 6 7 9	4
4 7 9	2 4 5 9	4 5 9	1	4 5 6 7	3 6 7	8 7 9	3 6 7 9
8	1 3 4	6 9	4 7	2 7	2 3 5 7	1 2 3 7	1 3 7

如图所示，我们可以通过结构找到这一条链：

$r1c12(3)=\{r1c1(2), r1c2(1)\}$ 的其一-- $r2c89(12=35) \Rightarrow r1c78, r2c3 \langle \rangle 3$

首先我们来思考第一个强关系到底是如何形成的，我们尝试把 JE 结构看作成立的话，那么交叉线单元格里，1、2、4 都最多出现两次，但 3 可以出现三次。为了让其成立，我们忽略 3 的影响，设链头 $r1c12(3)$ 为假，这使得基准单元格里没有 3。这样的话，JE 就可以成立了。不过，我们可以发现， $r1c12$ 必须填入 1、2、4 的其二，而由于 4 的客观存在，这会使得 $r1c12$ 里肯定有一个 4，而另外一个单元格填入的则是 1 或 2，所以 $r1c12(3)$ 和 $r1c12$ 里的 1 或者 2 是不同假的，换句话说，当 $r1c12(3)$ 为假的时候， $r1c12$ 里的 1 或者 2 就必须有一个数是为真的。

接着，由于 JE 成立，1、2、4 在目标单元格 $r2c4$ 和 $r3c7$ 里就必须是不同的填数，不过很显然，如果此时让 $r2c7$ 的镜面单元格 $r2c89$ 里填入一个 1 或者一个 2，都会同时使得目标单元格无法放入 1 或者 2，导致两个目标单元格都只能放入 4，这是不允许的，所以到这里， $r2c89(12)$ 就必须为假。

当 $r2c89(12)$ 为假的时候， $r2c89$ 就会产生一个关于 3 和 5 的显性数对，链到此结束。所以可以发现，实际上整个链长度为 3，但每一个强弱关系都值得深思。那么删数就是头尾的交集了，所以删除掉的只有图上标注出来的那些数字 3。

3-2-6-2 示例 2：利用占位形成的弱关系

2 3 4	1 2 3	1 2 3	4 5	9	2 3 4	1 2 3	1 2	8
6	7	1 2 3	4 5	9	5 6 7	1 2 3	1 2 3	1 3
5	1 3	1 2 3	4 5	7	2 3 4	1 2 3	1 2 3	1 3
2 3	4 5	7	6	1 2	5	8	1 3	1 3
1	5 6	5	8	5	4	7	6	2
2	5 6	2	1 2	3	2	1	6	4
3 1	3 1	2 3	8	2 3	1 2 3	1 2 3	1 2 3	4
4 3	2	4 5	4 5	1	3	6	8	3
8	1 3	6	9	2	2 3	5	1 2 3	1 3

如图所示，还是上一道题，不过我们把 3 已经删除掉了，现在来看看之后的逻辑。实际上我们可以发现，数字 4 在交叉线单元格里只能最多填一个，这便使得如果 JE 成立后，两个目标单元格都必须填 4 才能填满三次，但实际上这是不允许的。如果 $r1c12$ 是填入 4 和 a (a 是 1、2 其一)，那么目标单元格里怎么说也得有 a 的一个位置，但此时两个目标单元格都是 4，导致 a 无法放进去，所以出错。

接着后面就好推理了：所以 $r1c78(47)$ 的毛刺隐性数对成立，删除 $r1c78(6)$ ，使得 $r1c6(6)$ 为真。所以实际上，删数是看 $r1c12(3)$ 和 $r1c6(6)$ 的交集，即区块不连续环删除了 $r1c6(3)$ 。

3-2-6-3 示例 3：结构有点分散的链

[illegible]

这个链有点奇怪，所以我把不同的节点的候选数用不同的颜色涂起来了。这个链首先从 b6 的 2、7、9 入手。首先发现 $\{r4c9, r56c7\}(279) = \{r4c9, r6c7\}(3)$ 的关系成立，得到 **r4c9** 和 **r6c7** 里至少有一个填入 3；接着得到 **r4c7(3)** 为假的结论。当然 **r4c7(3)** 为假后，我们就可以发现，1、3、4、6 在交叉线单元格里就必须填入最多两次，满足了 JE 本应该满足的要求，所以 JE 此时成立，也就是说，目标单元格里就必须是 1、3、4、6，所以此时是可以删除 **r3c7(9)** 的。

所以实际上，这个链可以这么描述。

$$\{r_4c_9, r_5c_7\}(279) = \{r_4c_9, r_6c_7\} - r_4c_7(3) = \{r_1c_{12}, r_2c_4, r_3c_7\}(1346) \Rightarrow r_3c_7 \leq 9$$

那么，基本的一些定理就说完了，下面来看一个示例，可以通过基本的定理删除掉这些候选数。请自行理解，并尝试找到每一个删数的真正删数原因。

9	4 5 6	4 5 6	3	2	2 3	8	1	5 6	1 2	4 5 6	9	8	1 2 3	7	1	5	1 2 3	6	1	2	4 5	1	4 5
4 6	1	4 6	7	2	8	5	4	6	2	4 6	5	2 3	6	2 3	4	1 2 3	6	2	1 2	7	8 9	1	8 9
2	5 6	5 6	8	3	4	1	2	5 6	9	5 6	7	1 2	6	2	5 6	1 2	6	2	5	7 8	3	1	5
1 3	5 6	5 6	1	2	4	3	1	5 6	1	5 6	3	1 2	4	6	2	1	6	2	4 5	7	4 5	6	4 5
7	1 3	7 8	8	7 8 9	5	1 3	7	5 6	7 8 9	8	5	4 6	7	1 2	8	1	3	9	1	4 5	6	4 5	
1 3	4 5	9	4 5	6	1	3	1	4 5	1	2 3	4	1	6	4	6	1	6	5	4	3	1	4	6
7	2	8	8	7 8 9	5	7 8 9	7	5	7 8	8	2	8	7 8	9	7 8	9	9	7	7 8	7	9	2	9
8	4 5	4 5	1	2	6	3	1	4 5	1	5 6	4	2	3	1	4	5	3	4	5	9	1	2	3
1	4 6	3	9	7	5	1	7	4	7	5	1	4	6	5	6	7 8	7 8	4	5	6	8	9	7
1 2	4 5 6	4 5 6	7	1	2	3	1	4 5 6	1	5 6	8	2	3	2 3	3	2 3	2 3	9	1	2	4 5	4 5	6

3-3 孪生初级飞鱼导弹 (Siamese JE)

在我们描述了之前的内容后，我们来看一种类似于孪生鱼的飞鱼导弹结构：**孪生初级飞鱼导弹 (Siamese JE)**。

3-3-1 基础使用逻辑

4 5 6 9	5 6 8	4 6 8	1 6 7	1 4 5 6 7 9	1 5 7 9	1 3 7 9	2	1 3 4 5 8 9
1	7	4 6 8	4 6 9	2	5	4 3 9	4 5 8 9	4 5 3 8 9
2 4 5 9	2 5	3	8	1 4 5 7 9	1 5 7 9	6	4 5 7 9	1 4 5 9
2 3 5 6	1 2 3 5 6 8	1 2 8	1 2 6	1 5 6 8 9	4	2 3 9	5 6 9	7
2 5 6 7	1 2 5 6	9	3	1 5 6 7	1 2 5	8	4 5 6 9	2 4 5 6
2 3 5 6 7	4	2 6 7 8	2 6	5 6 7 8 9	2 5	2 3 9	1	2 3 5 6 9
2 4 7	1 2 6	5	1 2 7 9	3 4	3 6 7 8 9	1 2 9	4 7 8 9	1 2 4 8 9
2 3 4 6 7	1 2 3 6 4 6	1 2 7	1 2 9	1 4 7 8 9	1 2 7 8 9	5 4 6 4 6 7 8 9	3 1 2 3 8 9	3 1 2 3 8 9
8	9	1 2 4 6 7	5	1 4	1 2 4	1 2 3 4 6 4 6 7	5	3 1 2 3 4 6 4 6 7

4 5 6 9	5 6 8	4 6 8	1 6 7	1 4 5 6 7 9	1 5 7 9	1 3 7 9	2	1 3 4 5 8 9
1	7	4 6 8	4 6 9	2	5	4 3 9	4 5 8 9	4 5 3 8 9
2 4 5 9	2 5	3	8	1 4 5 7 9	1 5 7 9	6	4 5 7 9	1 4 5 9
2 3 5 6	1 2 3 5 6 8	1 2 8	1 2 6	1 5 6 8 9	4	2 3 9	5 6 9	7
2 5 6 7	1 2 5 6	9	3	1 5 6 7	1 2 5	8	4 5 6 9	2 4 5 6
2 3 5 6 7	4	2 6 7 8	2 6	5 6 7 8 9	2 5	2 3 9	1	2 3 5 6 9
2 4 7	1 2 6	5	1 2 7 9	3 4	3 6 7 8 9	1 2 9	4 7 8 9	1 2 4 8 9
2 3 4 6 7	1 2 3 6 4 6	1 2 7	1 2 9	1 4 7 8 9	1 2 7 8 9	5 4 6 4 6 7 8 9	3 1 2 3 8 9	3 1 2 3 8 9
8	9	1 2 4 6 7	5	1 4	1 2 4	1 2 3 4 6 4 6 7	5	3 1 2 3 4 6 4 6 7

如图所示。先看左图，我们把 r7c12 作为基准单元格来推导。显然，最终我们得到的结论就是 r8c4 和 r9c7 只能放 1、2、4、7 的其二，所以删掉其余的候选数。

不过，这并不是这个题目的结束。我们尝试换一个角度，我们就会发现，我们把基准单元格改变为 r9c56 的话，那么根据交叉线单元格的分布，最终确定目标单元格位于 r7c7 和 r8c3，而此时这两个单元格也应该是只能放入 1、2、4、7 的其二的，所以删除掉其余的候选数。

这种结构就只有这些删数吗？其实依旧没有完，我们继续向下推理。

4 5 6 9	5 6 8	4 6 8	1 6 7	1 4 5 6 7 9	1 5 7 9	1 3 7 9	2	1 3 4 5 8 9
1	7	4 6 8	4 6 9	2	5	4 3 9	4 5 8 9	4 5 3 8 9
2 4 5 9	2 5	3	8	1 4 5 7 9	1 5 7 9	6	4 5 7 9	1 4 5 9
2 3 5 6	1 2 3 5 6 8	1 2 8	1 2 6	1 5 6 8 9	4	2 3 9	5 6 9	7
2 5 6 7	1 2 5 6	9	3	1 5 6 7	1 2 5	8	4 5 6 9	2 4 5 6
2 3 5 6 7	4	2 6 7 8	2 6	5 6 7 8 9	2 5	2 3 9	1	2 3 5 6 9
2 4 7	1 2 6	5	1 2 7 9	3 4	3 6 7 8 9	1 2 9	4 7 8 9	1 2 4 8 9
2 3 4 6 7	1 2 3 6 4 6	1 2 7	1 2 9	1 4 7 8 9	1 2 7 8 9	5 4 6 4 6 7 8 9	3 1 2 3 8 9	3 1 2 3 8 9
8	9	1 2 4 6 7	5	1 4	1 2 4	1 2 3 4 6 4 6 7	5	3 1 2 3 4 6 4 6 7

如图所示，此时我们可以发现到的是，和 **c347** 里面的交叉线单元格互补的九个单元格是 **r789c347**，而排除掉刚才删数和占位的单元格，还剩下没用的单元格是 **r7c4** 和 **r9c3**，而实际上，这两个单元格恰好位于基准单元格 **r7c12** 和 **r9c56** 的删数交集上。试想一下，如果 **r7c12** 填入的是 **a** 和 **b** 的话，那么 **r9c56** 还能否是 **a** 和 **b** 呢？我们的答案是不能。因为一组基准单元格是对应一组目标单元格的，而一组目标单元格需要保证数字放入是合适的，就必须保证这一组的数字在目标单元格里必须要填进去，否则其余位置是放不了数的。举个例子。假如 **r7c12** 是 **a** 和 **b**，那么对应的目标单元格是 **r8c4** 和 **r9c7**，它们就必须是一个 **a** 和一个 **b**；接着如果我们再看 **r9c56** 这一组基准单元格的话，它对应的目标单元格是 **r7c7** 和 **r8c3**。如果它们也是 **a** 和 **b** 的话，显然，由于两种不同的 JE 视角用到了同样的三列，如果 **a** 和 **b** 完整了，那么 1、2、4、7 里不是 **a** 和 **b** 的剩下两种数字（假设为 **c** 和 **d**）就无法填到剩下的单元格里了。也就是说，**r9c56** 里必须是 **c** 和 **d** 的组合。当然，如果 **a** 和 **b** 的其一能放到 **r9c56** 里，这样的情况也是不允许的，否则剩下的三种数字里必然有一种数字填不进去。

所以，我们暂且假设 **r7c12** 是 **a** 和 **b**，而 **r9c56** 是 **c** 和 **d**。显然，**a**、**b**、**c**、**d** 是四种不同的数字，且都是 1、2、4、7 里的数字（即一个字母对应一种数字），那么刚才说到的“删数交集”**r7c4** 和 **r9c3** 就派上用场了：因为四种数字不同的关系，而且四个单元格是跨区的，所以它们形成了一个关于 1、2、4、7 的跨区四数组，所以 **r9c3** 和 **r7c4** 是不能放 1、2、4、7 的，所以这两个单元格要删除掉 1、2、4、7 这四种数字。

当然，我们可以从图示里看到，推导逻辑需要要求基准单元格是相同的数字种类。如果不同，推导显然是进行不下去的，或者说，有些时候可以进行，但需要依赖于其它的逻辑，所以此处我们就可以不作过多考虑了。

另外，孪生 JE 有时候也称为 JE4。在官方的 JE 技巧文档里，它使用的是 JE 加上一个数字来表示这个技巧的名字，其中这个数字就表示目标单元格的总数量。例如最初的 JE 逻辑需要两个目标单元格，所以叫 JE2；而后续发现的只需要一个基准单元格和一个目标单元格的，称为 JE1；而带有一组共轭对的称为 JE3。

这个推导逻辑便是这个技巧需要告诉给你的思路。不过，由于结构的特殊性，它也产生了一些各式各样的使用模式，下面我们来看一些情况。

3-3-2 更多例子

我们再来看看一些其它的例子。

3-3-2-1 示例 1：标准的孪生 JE

1 5 6 7 8	2	3	1 4 6 7	5 6 7 8	1 4 5 6 7	5 6 7 8	9
4	5 7 8	5 6 7 8	2 6 7 9	2 5 6 7 8	1 5 6 7 8	2 3 5 6 7 8	3 5 6 7 8
1 5 6 7 8	9	1 5 6 7 8	1 2 6	3 5 6 7 8	1 2 5 6 7 8	4 5 6 7 8	5 6 7 8
2	5 7	5 6 7	8	1 5 6 7 9	3 5 6 7 9	4 5 6 7 9	4
1 5 6 4 5	1 3 4 5	1 4 5 6	2 3 6 9	2 5 6	7	8	1 3 5 6 9
9	1 3 5 7 8	1 5 6 7 8	3 6	4 5 6	3 5 6 7	1 3 5 6 7	2
3	4 5 7 8	2 4 5 7 8	4 6 7	9	4 6 8	2 4 5 6 7 8	1
1 5 7 8	6	1 2 4 5 7 8 9	1 2 3 4 7	2 7 8	1 2 3 4 8	2 3 4 5 7 9	3 5 7 8
1 4 7 8	1 4	1 2 4 7 8 9	5	2 6 7 8	1 2 3 4 6 8	2 3 4 6 7 9	3 6 7 8

如图所示，这个例子有一点不对称，但依旧不影响推理，而且有一组 JE 的模式是无法通过基础的逻辑删数的，即 r1c5 和 r3c9 这一组目标单元格，按理说应当删除 5、6、7、8 外的其余候选数，但这两个单元格只有 5、6、7、8，所以删除不了数字。不过配合孪生 JE 技巧就可以得到额外的删数了。

3-3-2-2 示例 2：孪生的第二类 JE

我们知道，孪生 JE 是两个 JE 合并得到的，所以我们可以把之前的示例进行合并，得到一个孪生 JE。

9	8	1 2 3	7	1 2 4 5	1 2 3 4 5	6	1 3 4	1 2 3
5	2 3 7	1 2 3 6	2 3 4 6 9	1 2 4 6 9	1 2 3 4 6 9	3 4 6 7 8 9	1 3 4	1 2 3 7 8 9
1 2 6 7	2 3 7	4	2 3 6 9	8	1 2 3 4 6 9	3 7 9	5	1 2 3 7 9
3	5 7	9	2 5 6	1 2 5 6	1 2 5 6	5 7 8	1 6 7 8	4
1 4 7	6	5 7	8	1 4 5 7 9	1 4 5 7 9	2 5 7 8	1 3 4 7 9	5 7 9
1 2 4 7 8	2 4 5 7	1 2 5 7 8	2 4 5 6 9	3	1 2 4 5 7 9	5 7 8 9	1 6 7 8 9	5 7 8 9
2 4 7 8	1	2 3 5 7 8	2 3 4 5 9	6	2 4 5 9	3 4 5 7 8 9	3 4 7 8 9	3 5 7 8 9
4 7 8	4 5 7	3 5 7 8	1	4 5 9	4 5 9	4 5 7 8 9	2	6
4 2 6	2 3 4 5 9	2 3 5 6	2 3 4 5 9	7	8	1	3 4 9	3 5 9

如图所示，这一则示例的推导逻辑相信一定不用多说了，我们只阐述下孪生 JE 的额外删数。

我们可以论证得到的是，r8c56 和 r9c89 填入的数字不同，所以它们的交集上，即此处的 r8c7 和 r9c4 就不可以填入 3、4、5、9 了。所以，删除掉它们。

3-3-2-3 示例 3：带有 X 致命定理删数的孪生 JE

2 3 8	2 3 4 8	1	2 3 4 5 8	3 4 5 7	2 3 4 5 7	2 3 4 5 7 8	9	6
2 3 6 4 8 9	2 3 6 4 8 9	2 3 4 8 9	1	3 4 7 9	2 3 4 7 9	2 3 4 7 8	2 4 5 7 8	2 3 4 5 7 8
5	7	2 3 4 8 9	2 3 4 8	3 4 6 4 9	2 3 4 6 4 9	1	2 4 8	2 3 4 8
1 2 3 5	1 2 3 5	7	9	8	2 3 4 5	6	1 2 4	1 2 4
1 2 3 8 9	1 2 3 8 9	6	7	1 3 4	2 3 4	5	1 2 4 8	1 2 4 8 9
4	1 2 5 8 9	2 8 9	2 5	1 5 6	2 5 6	2 7 8	3	1 2 7 8 9
1 2 3 7	1 2 3 4	5	6	3 4 7	8	9	1 2 4 7	1 2 3 4 7
1 3 6 4 7 8 9	1 3 6 4 8 9	3 4 5 8 9	3 4 5	2 4 5 7 9	3 4 5 7 9	3 4 7 8	1 4 5 6 7 8	1 3 4 5 7 8
2 3 6 4 7 8 9	2 3 6 4 8 9	2 3 4 5 8 9	3 4 5	3 4 5 7 9	1	2 3 4 7 8	2 4 5 6 7 8	2 3 4 5 7 8

如图所示，我们可以按照 r456789c347 作为交叉线单元格，找到匹配的两组基准单元格。而 r1c7 和 r3c3 是两组基准单元格的交集，所以删除掉 2、3、4、8。

1 2 7	1 2 4	5	1 7 8 9	1 2 4 7 8 9	2 4 7 8 9	1 2 4 7 8	3	6
1 2 3 7	1 2 3 6 4 7	1 2 4 6 7	1 7 8	5 7 8	2 4 6 7 8	1 2 4 7 8	4 7 8 9	1 2 4 8 9
8	9	1 2 4 6 7	1 3 6 7	1 2 3 4 7	2 3 4 6 7	5	4 7	1 2 4
4	1 2 3 5 8	1 2 7 8	3 5 6 7 9	2 3 7 9	2 3 5 6 7 9	1 3 7 8	5 7 8 9	1 3 5 8 9
1 2 3 7	1 2 3 5	9	3 7	8	2 3 5 7	6	4 5 7	1 3 4 5
3 5 6 7	3 5 6 8	6 7 8	4	3 7 9	1	3 7 8	2 5 8 9	3 5 8 9
2 5 6 9	7	4 6 8	2 5 6 8 9	3 4 9	3 4 5 8 9	2 3 4 8	1 4 5 8	2 3 4 5 8
1 5 9	1 4 5 8	1 4 8	2	1 3 4 9	3 4 5 8 9	3 4 8	6	7
1 2 5	1 2 4 5 8	3	1 5 7 8	6	4 5 7 8	9	4 5 8	2 4 5 8

这一则示例出了孪生 JE 的删数外，r3c9(4)也可以删除。

首先，我们针对于 r1c12 作为基准单元格来看 JE 结构，发现 1 和 2 的确定值分布在下面对应 X 区域里呈现对角宫分布的形式，所以 1 和 2 这一组组合是无法放到 r1c12 里的，即 r1c12 一定不是 1、2 构成的数对。不过，根据孪生 JE（即 JE4），我们可以得到

的是，四个基准单元格最终的填数是完全不同的。但是，1 和 2 此时只能放在所有这四个基准单元格的其中三个单元格里：**r1c12** 和 **r3c9**。由于 **r1c12** 不能同时是 1 和 2 的缘故，所以 **r3c9** 必须是 1 或者 2 的其一，否则 1 或者 2 的其一将填不到四个基准单元格里，导致矛盾的出现，所以 **r3c9** 不能填和 1、2 无关的其余候选数，故 **r3c9** \neq 4。

下面我们来思考一个问题：假定单元格 **r3c9** 此时还含有候选数 7，那么能否删除呢？答案是肯定的，因为 7 是 1、2、4、7 的其一，它不会影响孪生 JE 结构的形成，所以刚才的结论都会成立，所以删数肯定是成立的；但如果 **r3c9** 还含有和 1、2、4、7 无关的候选数的话，可能就不行了，因为 **r3c9** 作为基准单元格的其一，是不允许含有其它和结构无关的候选数的；否则孪生 JE 都是无法形成的，更别提删数是否成立了。

下面我们来看一个富有挑战性的示例。

3-3-2-4 示例 4：确定值也可以算作推导的一部分吗？

9	8	1 2 3 5	7	1 2 3 4	1 2 3 4	6	2 3 4 5	1 2 4 5
7	1 3 6	1 2 3 6	8	5	1 2 3 4 6	9	2 3 4	1 2 4
1 2 3 5 6	1 3 5 6	4	1 2 6	1 2 3 6	1 2 3 6 9	1 2 5 7	2 3 5 7 8	1 2 5 7 8
6	1 4 5	1 2 5	1 2 4 5	1 2 4 7	8	1 2 4 5 7	9	3
1 2 3 4	7	1 2 3 5 9	1 2 4 5 6	1 2 4 6 9	1 2 4 6 9	8	2 4 5	1 2 4 5
1 2 4	1 4 5 9	1 2 5 8 9	3	1 2 4 7 9	1 2 4 7 9	1 2 4 5 7	2 4 5 7	6
5	1 4 6 7 8	1 6 7 8	9	1 2 4 6 7 8	1 2 4 6 7	3	2 4 6 7 8	2 4 7 8
1 3 4 8	2	1 3 6 7 8 9	1 6 7 8	1 3 4 6 7 8	1 3 4 6 7	4 5 7	4 5 6 7 8	4 5 7 8 9
4 3 8	4 3 6 9 7 8 9	3 3 6 4 7 8 9	2 2 6 4 7 8	2 3 4 6 7 8	5	2 4 7	1	2 4 7 8 9

如图所示。我们发现，在交叉线单元格里是含有一个确定值 5 的，但是在之前的所有示例里面，确定值都不作为我们考虑填入多少次的基本推导逻辑。不过我们仔细分析这个题目就会发现，如果我们尝试把 5 算作一部分，而基准单元格 **r4c23** 和 **r5c89** 也不必忽略这个 5，会怎么样？

在交叉线单元格里，5 确实最多只能填入两次，首先，我们算上 **r7c1** 的这个确定值 5，因为它确实占据了 **c147** 的其中一列填 5 的权利，而剩下两列就必须填入两个 5 了。而实际上，在交叉线单元格里，能放 5 的地方就只有 **r38c7** 两处了，而且它们恰好同列。同列就意味着这两处单元格只能最多有一个可以放 5，所以算上 **r7c1** 的这个 5，恰好可以叫做“最多两次 5”。

所以后面的逻辑就可以发现，1、2、4、5 最多两次确实是成立的，所以后面的逻辑就不用多说了，根据孪生 JE 的删数模式进行就可以了。

这个例子巧妙的地方就在于，它利用了一次确定值。

3-4 高级飞鱼导弹 (Senior Exocet)

下面我们来看高级版的飞鱼导弹——**高级飞鱼导弹 (Senior Exocet, 简称 SE)** 结构。

3-4-1 普通示例

1 2 3 4	1 3 4	7	1 2 3 4 6	1 4 5 6	2 3 5 6	1 2 3 4 5	9	8
6	1 3 4 8	1 2 3 4	9	1 4 5 8	2 3 5 8	7	2 3 5	1 2 4 5
5	1 3 4 8 9	1 2 3 4 9	1 2 3 4 8	1 4 7 8	2 3 7 8	6	2 3	1 2 4
1 7 8 9	1 5 7 9	1 5 9	2 6 8	3	2 6 7 8 9	1 2 5 8	4	1 2 5 6 7
4 3 7 8	2	4 5 3	4 6 8	4 6 7 8	1	5 3 8 7 8	3 5 6 7 8	9
1 3 4 7 8 9	1 3 4 7 9	6	5	4 7 8 9	2 7 8 9	1 2 3 8 7 8	2 3 7 8	1 2 7
2 3 7 9	3 5 6 7 9	2 3 5 9	3 6 8	5 6 8 9	4	2 5 8	1	2 5 6 7
1 4 7 9	1 4 5 6 7 9	1 4 5 9	1 6 8	2	5 6 8 9	4 5 8 7 8	5 6 7 8	3
1 2 3 4	1 3 4 5 6	8	7	1 5 6	3 5 6	9	2 5 6 4 5 6	2 4 5 6

如图所示，我们如果把 **r1c12** 作为基准单元格的话，就会发现目标单元格此时仅剩下唯一一个单元格：**r3c4** 了。这是否意味着结构直接出错了呢？并不是这样，因为 **r6c7** 是这个时候“居然”空的。

之所以说“居然”，是因为之前的逻辑里，我们可以发现一个经验上的东西，即确定值在交叉线单元格的分布基本上是占满了其中的两行（列）甚至是三行（列）。而这个例子里却缺少了一个单元格 **r6c7** 没有被确定值占据掉。我们尝试从 JE 的原理入手分析这个结构。

假如 **r6c7** 不放入基准单元格 **r1c12** 涉及的候选数 1、2、3、4 会怎么样呢？这显然就意味着，1、2、3、4 在交叉线单元格里都是最多出现两次的情况的，而这便会使得与之互补的 **r123c347** 里必须至少一个 1、至少一个 2、至少一个 3 和至少一个 4 的出现才能够满足 **c347** 填够三次 1、2、3、4 的要求。

可是如果我们一旦假设 **r1c12** 填入的是 a 和 b 后，就会发现，其中有一个数必然放不到 **r123c347** 里，这是因为此时目标单元格仅剩下唯一一个单元格：**r3c4** 了。如果 **r3c4** 是 a 的话，那 b 放哪里呢？那如果 **r3c4** 是 b 的话，那么 a 又放哪里呢？这便使得结构出现了矛盾：a 和 b 里会有一个数填不进去。所以，我们最初的假设，**r6c7** 不放入 1、2、3、4，这个说法就是错误的，故 **r6c7** 只能是 1、2、3、4 的其一，故删除掉和 1、2、3、4 无关的其余候选数，故 **r6c7** \neq 8。那么，我们就完全可以把 a 和 b 的其一放到 **r6c7** 里作为目标单元格了，这样一来，**r3c4** 如果是 a，那么 **r6c7** 就是 b，这样便满足了要求。所以，**r3c4** \neq 8 实际上也是成立的。

这就是这个示例想要告诉你的推导逻辑：它尝试把目标单元格向下调整到了交叉线单元格里，但依旧不影响我们的推导，这种结构作为飞鱼导弹的升级版，自然就被取名叫做**高级**

飞鱼导弹（Senior Exocet）了。

3-4-2 退化 SE（Sashimi SE）

下面我们来看一点稍微不一样的构型。

4 5 6	2	1	2 3	2 3	3	2 3	7	2 3
4 5 6	4 6	1	4 5 6	5 6	4 5 6	4 5	7	5 6
3	9	2	2	2	2	8	1 2	1 2
		4 5 6	4 5 6	5 6	4 5		4	5 6
8	2	2	1	2 3	3	9	2 3	2 3
4	4 6	4 5 6	4 5 6	5 6	4 5		4	5 6
1	1	9	3	5 6	1	7	1 2	1 2
4 5 6	4 6			5 6	5 6		8	5 6
1	1	3	5	5	4	2	1 3	1 3
5	7 8	7 8	7 8 9	8 9			5	8 9
2	1 3	5 6	5 6	5 6	1	1 3	1 3	4
	6	8	8 9	7 8 9	7 8 9	5	8 9	
1	1 2	3	7	2	5	6	1 2	1 2
4	4			5	4 5		4	8 9
	9	8		8 9	8 9		8 9	8 9
4	2	2	2	1	4	3	5	2 3
6 4 6	4 6 4	4 6 4	4 8 9	8 9	4 8 9	4	1 2 3	1 2 3
7 9 7	7 8	7 8	8 9		8 9		4	8 9
1	5	2	2	2 3	6	1 2 3	1 2 3	7
4		4	8 9	8 9		4	8 9	

如图所示。我们发现，如果把 **r456789c347** 作为交叉线单元格，而把 **r1c12** 作为基准单元格的话，里面居然包含了基准单元格的候选数 6。这未免也太神奇了吧。显然，它想要直接告诉我们，基准单元格里必有一个 6 的出现。如果基准单元格没有 6 的话，JE 此时是成立的，那么 2、4、5 作为我们推导的数字，却发现目标单元格此时就只有 **r2c4** 一个单元格，那么 **r1c12** 里填入的两种数字的其中一种就必然无法放到 **r123c347** 里，导致矛盾的出现。所以，我们不得不把这个 6 也算作是目标单元格处理，这样，JE 此时是成立的，而且它还直接暗示了我们，**r1c12** 必须有一个 6，故 **r1c12(6)** 是一个区块结构，删除掉 **r1** 和 **b1** 其余位置的 6。当然，**r2c4 <> 6**，因为目标单元格的填数必须是不同的，它对应着“基准单元格的填数是不同的”这一个要求。

这种结构也属于 SE 的一种，但由于它的其中一个目标单元格直接变为了确定值来暗示填数结论，我们把这种结构使用鱼里的“退化”一词来表示，所以这个结构叫做**退化 SE（Sashimi SE）**。

3-5 复杂飞鱼导弹 (Complex Exocet)

更为复杂的示例就是把 JE 和鱼的变体类型结合起来了。不过实际上，JE 这种技巧结构相对较大，而且很难发现各种变体，所以这样的示例并不多见到，这里仅仅拿出几则示例阐述逻辑即可。

3-5-1 宫内飞鱼导弹 (Franken Exocet)

3-5-1-1 宫内 JE (Franken JE)

1 3 4 6	7	1 3 4	1 6 4 6 8 8	2	1 3 5 9	5	9
1 2 3 5 6	8	1 2 3 5	7	5 6 9	1 2 3 6	1 2 3 6	4
1 2 4 5 6	9	1 2 4 5	1 6 3	4 5 6 7	1 2 8	1 2 7 6	
2 3 5 8 9	2 3 5	6	2 8 9	1 4 7 8 9	2 4 5 7 8	2 4	2 3 7 8
1 2 3 8	4	1 2 3 8	5 6 7 8	6 6 7 8	1 2 3 7 8	9 7 8 6	1 2 3 6
1 2 5 8 9	1 2 5	7	2 6 4 6 8 9 8 9	3	1 2 4 5 8	1 2 4 6	1 2 6
7	2 3 5	4 5 8 9	3 8 9	5 8 9 1	6 4 2 3	2 3 8	2 3 8
1 3 4 5 8	1 3 5	1 3 4 5 8 9	3 6 8 9	2 5 6 8 9	1 3 4 8 9	7 1 3 8	
1 2 3 8	6	1 2 3 8 9	4 7 8 9 7 8 9		1 2 3 8 9	1 2 3 8 9	5

如图所示，这个结构长得比较类似于之前鱼里的宫内鱼结构。我们发现，数字 1、2、3 在交叉线单元格里都最多只能放入两个进去。因为 1 在所有的交叉线单元格里都只出现在 c13 上，而 2 和 3 也是。所以显然，只有两个区域就只可能最多出现两次。

接着，我们假设基准单元格 r12c7 填入是 a 和 b 的其二，那么根据基本的 JE 的理论，我们应当知道的是，在 r59c89 四个单元格里，必须出现 a 和 b，否则 a 或者 b 的其一就无法放入结构涉及的 r59b1 里凑够三次 1、2、3，使得出现矛盾。所以，r5c9 和 r9c8 里必须有一个 a 和一个 b，于是删除和 1、2、3 无关的其余候选数。

这一则示例有趣的地方是，它涉及的交叉线单元格补全区域就是两行一宫，所以和之前的鱼技巧一样，它被命名为**宫内 JE (Franken JE)**。而实际上我们可以发现，寻找填数最多的次数也就是在寻找交叉线单元格里填入这些数字的删除域区域。删除域区域要求的是“区域内最多能放入一个”，这一点恰好符合我们寻找 JE 交叉线单元格的宗旨。

下面我们来看一个比较难理解的例子。

3-5-1-2 宫内 SE (Franken SE)

1 2 5 6 7	2 3 5 6 7	1 2 3 5 7	4	1 3 5 6 7 8	1 7 8 6	5 6 7 8 8	1 5 6 8	9
4	5 6 7	1 5 7	1 6 7 8	1 5 6 7 8	9	2	3 1 7 8	6
1 5 6 7 9	8	1 5 7 9	1 3 6	2 7	1 6	5 6 7	1 5 6	4
1 2 5 7 9	2 4 5 7 9	6	1 2 4 7 8	3	4 5 8 9	1 4 5 8 9	1	
8	3 4 7	1 3 4 7	5	9	1 4 6	3 4 6	1 4 6	2
1 2 5 9	2 3 4 5 9	1 2 3 4 5 9	1 2 6 8	1 4 6	1 4 6	3 4 5 6 8 9	7	1 3 6 8
3	2 4 7	2 6 7	9	1 4 7 8	1 6 7 8	4 6 7 8	2 6 8	5
5 6 7 9	4 5 6 7 9	8	3 6 7	3 4 6	2	1	4 6 9	3 6
2 6 7 9	1	2 4 7 9	3 6 7 8	3 4 6	5	3 4 6	2 6 8 9	3 6 7 8

如图所示。这个示例有一点奇怪，它不能用 X 致命定理，但可以删除基准单元格的数字。我们先尝试不管这个 1，我们看看 1 是否满足交叉线单元格的填数要求。

如果 1 存在，那么基准单元格涉及的是 1、6、7、8，我们就都得看一遍。发现，r2 和 b3 存在交集，而交叉线单元格 r2c9 处于这个交集上。如果它能填入 1，则它会影响到 r2 和 b3 两个区域的填数。所以我们先不考虑它填 1 的情况。如果不填 1，那么结构最多还能放两次 1，因为 1 此时仅在 c38 出现。如果我们考虑这个 1 出现在基准单元格 r13c6 里的话，那么最终 1 要凑够四个区域填入四个的要求，由于目前只能在里面填入两个，这使得 1 不得不放到目标单元格 r7c5 上。那么还有一个位置呢？r2c9 是不允许此时填 1 的，这被我们刚才的假设所约束。所以 1 无法填满四个，出现矛盾。但如果我们放宽政策，假设 r2c9 可以是 1 的话，那么我们必然会优先考虑让 r2c9 填入 1，但实际上，你可以发现，不顾你怎么放 1，在结构里最多也只能出现三个，因为我们假设 r2c9 = 1 后，1 还可以填的位置只剩下 r5 和另外一个目标单元格 r7c5 里才可以放 1 了，而这最多只能放入两个 1，依然凑不够四个 1。

所以，实际上，不论 r2c9 到底是不是 1，只要我们假设基准单元格里放 1，就无法保证 1 能够填满四个。所以基准单元格里是不允许填入 1 的。

同时，由于这个条件的成立，剩余的 6、7、8 在交叉线单元格里都最多出现三次的缘故，使得剩下的可以填入 6、7、8 的位置仅有 r7c5 和挤进 r2 的单元格 r2c9，因此，我们不得不让 r2c9 也算作目标单元格，故 r2c9 <> 1 且 r7c5 <> 14。

3-5-2 交叉飞鱼导弹 (Mutant Exocet)

3-5-2-1 交叉 JE (Mutant JE)

2 3 4 5	2 3 5 7 8 9	1	2 3 4 5 7	2 4 5 9	3 4 5 7 9	2 4 6 7 8	2 4 6 7 9	2 6 8 9
6	2 7 9	4 7 9	2 4 7	8	4 7 9	3	5	1
2 3 4 5 9	2 3 5 7 8 9	3 4 5 7 8 9	1	2 4 5 9	6	2 4 7 8	2 4 7 9	2 8 9
8	3 5 6 9	5 9	3 4 5 6	7	4 5 6	2 6	1	2 5 6
1 3 5	4	5 7	3 5 6 8	1 5 6	2	9	3 6 7	5 6 8
1 3 5	3 5 6 7	2	9	1 5 6 8	1 3 5 8	6 7 8	3 6 7	4
2 4 9	1	4 8 9	2 4 6 8	3	4 8 9	5	2 4 6 9	7
2 3 4 5 9	2 3 5 8 9	6	2 4 5 7 8	1 2 4 5 9	1 4 5 7 8 9	1 2 4	2 4 9	2 9
7	2 5 9	4 5 9	2 4 5 6	1 2 4 5 6 9	1 4 5 9	1 2 4 6	8	3

如图所示，这一则示例是介绍宫内 JE 技巧的那一例例子经过修改的题目。发现 2、4、9 在 r2c15 的交叉线单元格里最多都只能放入两次，那么在剩余的 r789c15 里，2、4、9 都至少出现一个。假设基准单元格 r8c89 填入 a 和 b，则发现 a 和 b 只剩下唯一的两个单元格 r7c1 和 r9c5 可填，所以 r7c1 和 r9c5 必须是一个 a 和一个 b，否则其中一个将无法填数出现矛盾。所以这一则示例的删数就是 r9c5(156)了。

这一则示例利用到的是交叉鱼的思想，观察的交叉线单元格涉及的是行和列区域，所以称为**交叉 JE (Mutant JE)**，如果通过之前鱼的删除域区域思想来看的话，交叉线单元格的分布里，可以尝试把 r13 作为“删除域区域”来看待。

3-5-2-2 交叉线单元格涉及四个区域的交叉 JE

8	3	4	1 5 6 9	6 9	2	7	1 5 6 9	5 6 9
1	5	9	4 6 8	7	4 6 8	2 8	3	2 6 8
2 6	7	2 6	1 5 8 9	3 9	1 3 8	4	1 5 9	5 8 9
7	2 4 5 8	5 8	3	1	9	6	2 4 5 8	2 4 5 8
3	4 6 5 8	5 8	2	4 6	7	5 8 9	5 9	1
2 4 6	9	1	4 6 8	5	4 6 8	2 3	7	2 3 4
2 4 6	8	2 6	1 4 6 7 9	2 3 4 6 9	5	1 2 3 9	2 4 6 7 9	2 3 4 6 7 9
5	1 2 4 6	7	1 4 6 9	8	1 3 4 6	1 2 3 9	2 4 6 9	2 3 4 6 9
9	1 2 4 6	3	1 4 6 7	2 4 6	1 4 6	1 2 5	8	2 4 5 6 7

如图所示，这一则示例比较难理解。我们尝试把交叉线单元格所属的区域扩展到四个，即 **r2c258**。由于是四个区域，所以推导有些许不同。我们发现，2、4、6 在交叉线单元格里都最多可以出现三次，所以剩余补全四个区域的 **r789c258** 里就必须至少出现 2、4、6 各一个。

我们先不着急向下推导逻辑，我们先来看看最多三次是什么情况。首先是数字 2。数字 2 恰好可以用 **b3** 和 **r4** 覆盖，所以最多只能放两个。数字 4 可以在 **r45** 和 **b3** 里各放置一个，所以最多可以出现三个。数字 6 可以在 **r5** 和 **b23** 里各放置一个，所以最多也是三个。只不过特殊的地方是，2 只能最多两个。

所以我们假设到 **r7c13** 填入的是 **a** 和 **b**，那么剩余 **a** 和 **b** 的放置位置只剩下 **r8c8** 和 **r9c5** 两处。为了保证 **a** 和 **b** 都能放进结构里，**r8c8** 和 **r9c5** 就必须得是 **a** 或者 **b**。不过有一点不同，由于 2 在交叉线单元格里只能最多放两个，所以如果 **a** 或者 **b** 的其一是 2 的话，那么 **r8c8** 和 **r9c5** 里就必须都是 2，才能满足填入 **r2c258** 四个区域需要恰好四个 2 的要求；不过如果 **a** 和 **b** 跟 2 无关，那么我们就不需要考虑这一点了，因为 2 的位置可以随意放置在 **r789c258** 的其它位置上，这样也能凑够两次。

3-5-2-3 交叉 SE (Franken SE)

1	4	3	3	3	4	6	2	3	6	4	6	5
2	2	3	8	7	8	9	7	8	9	2	3	2
4	4	5	6	5	3	1	4	3	9	4	7	7
7	8	7	8	7	8	9	7	8	9	7	8	7
2	9	2	3	5	3	3	1	2	3	1	2	3
4	7	8	9	4	5	6	4	5	6	7	8	9
7	8	9	7	8	9	7	8	9	7	8	9	7
2	1	2	3	6	7	8	9	1	2	1	2	3
4	7	8	9	4	5	6	7	8	9	7	8	9
6	1	1	1	3	2	1	3	4	5	1	7	8
7	8	9	7	8	9	7	8	9	7	8	9	7
3	1	2	1	2	4	7	8	9	5	1	2	6
7	8	9	7	8	9	7	8	9	7	8	9	7
2	1	2	3	7	1	3	3	1	3	1	2	3
4	4	6	7	5	9	4	5	4	9	5	6	8
9	2	1	2	3	1	3	3	1	3	1	2	3
4	8	9	8	8	9	7	8	9	7	8	9	7
5	1	3	1	3	2	3	6	1	3	1	3	4
8	8	9	7	8	9	7	8	9	7	8	9	7

如图所示，这个结构就不必我给大家阐述逻辑了，希望你能自己理解这个示例。而且这一则示例还有删数，不过我没有在图上标注出来。

3-6 总结和小练习

至此，JE 的内容就全部结束了。虽然例子说完了，但是用法还是千变万化的，我们不可能做到全部都说一遍，所以后面的路还需要你自己走。下面我们来看一些有趣的例子，这些例子可能之前出现过，也可能没出现过，不过我会给出许多删数，请自行思考删数的来源，以及是否删数给全了。

9	8	1 2 3 4 5	7	1 2 6 4	3	1 2 3 4 6	3	1 2 4 5 6	9	8	1 2 3 4 6	7	1 2 3 4	2 3	1 2 3 4	2	1 3 5 6
6	1 2 3 5	1 2 3 4 5	1 2 3 4	1 2 3	3	7	3	1 2 4 5	1 2 4 5 6	1 3 4 5 6	7	1 2 3 4	6	5	1 2 3 4	2	1 3 8 9
1 2 3 4	1 2 3 4	7	1 2 3 4	5	4	8	9	8	9	1 2 3 4	9	1 2 4	1 2 3 4	1 2 3 4	7	2	1 3 8 9
1 2 3 7 8	4	1 2 3 8	1	6	5	2 3 6	3	2	6	8	7 8	9	8	4	2	1 2 6	3
1 2 3 7	1 2 3 6	9	8	1	6	4	6	5	4	6	5	3	2	6	7 8	9	4
5	5 6	5	4	6	3	2	4	6	1	4	6	7 8	9	7	1	2	1
1 2 3 4	1 2 3 7	6	5	2	3	9	4	7 8	1	4	7 8	3	4	5 6	7 8	9	2
1 2 7 8	1 2 5	1 2 7	2	4	6	3	1	5 6	7 8	9	8	6	5	4	6	7	6
4 5 7 8	5	4 5 8	3	6	1	4	6	2	4 5 6 7 8 9	1	4	6	2	4 5 6 7 8 9	1	4	6

9	8	1 2 4 5	7	1 2 3 4	6	1 3 5	3	1 2 3 5	9	8	1 2 4	7	6	2 3 4 5	3	1 2 3 5	1 3 4 5
1 2 7	2 3 7	6	1 2 3 8	5	1 3 8	4	3	1 2 3 7 8 9	1 2 4 5	1 2 3 4 5	6	1 2 4	1 2 3 5	4	9	1 2 3 7 8	1 3 4 5
1 2 4 5	2 3 7	1 2 4	1 2 3 4	1 2 3 4	9	1 3 5 6	3	1 2 3 7 8 9	2	1 2 4	3	1 2 5	1 2 4	2	4	1 2 4	6
3	6	1 5	8	1 4	2	1 4	5	9	4	1 3 6	1 9	6	1 9	5	2	1 3 7 8 9	1 3 4 5
1 2 7	4	1 2 5	6	1 2 3 7	8	1 3 5	3	1 3 7 9	2	1 2 8	7	3	1 2 4	2	6	1 5	8
1 2 7	2	8	1 2 3 4	1 2 3 4	5	1 3 4	3	6	2 3 5	1 2 4	1 2 4	8	2	4 5 7	3	6	1 3 4 5
4 5 6 7 8	1	4 5 7	3	4 8 9	2	5 6 9	4 5 6 8 9	4 5 8 9	1	2 3 6	2 8	2 3 6	2 3 7	2 3 7	4	9	3
4 5 7 8	5	3	4 5 8 9	6	2	4 5 8 9	4 5 8 9	4 5 8 9	7 8 9	6	4	5	6	1	3	7 8 9	6
2 4 5 6	2	2	3	3	3	3	5 6 9	1	7	7 8 9	6	4	5	1	5 6 7 8	5	2

Part 4 网 (Multi-sector Locked Set)

欢迎进入本文档里的最难的一部分：**网**（**Multi-sector Locked Set**，简称 **MSLS**）。

4-1 我们从 SDC 说起

先来看看 SDC 基础版

4-1-1 度 SDC (Degree-base SDC)

我们可以从之前的 SDC 的讲解里看到，SDC 是一种由两个区域构成的整体跨区的数组结构。那么 SDC 能否拓展到涉及更多个区域呢？

1	3	2	2	2	4	5	6	1	6
9		5	4	4	7	8	9	7	8
8	7	2	6	4	1	9	4	5	3
1	4	5	5	3	3	5	5	6	2
9		7	7	8	9	7	8		7
2	1	1	8	4	6	4	5	4	5
5	3	6	9	2	3	4	5	6	7
7	8	8	1	7	7	1	4	5	6
3		6	1	5	4	6	7	8	9
7	8	7		7	9	8	7	8	9
4	1	1	3	3	2	6	6	8	5
	9	7	7	7	6	8	8	9	
5	3	5	3	4	3	2	1	2	6
7		9	7	4	7	6		9	
6	2	8	5	1	5	7	3	4	
		9							

如图所示，我们观察这个结构，它涉及了三个区域：**r5**、**c8** 和 **b6**，一共涉及 **r5c289**、**r2c24** 和 **r4c9** 这六个单元格和六种不同的候选数 4、5、6、7、8、9。

首先我们对这样六个单元格按照所属区域涂上颜色：我们发现六格之中，4 和 5 只出现于同一列上，所以我们暂用绿色标注；6 和 8 只出现于同一行上，所以我们暂用橙色标注；7 和 9 则只出现于宫内，所以图上紫色。

此时我们会发现，六格有六种不同的候选数，并且每一种候选数都不会跨区域出现。所以，我们可以这样思考这个问题：结构在 **b5** 内，有多个涂色的单元格一共有三个：**r4c8** 和 **r5c89**。因为上方的 **r2c8** 只有候选数 4 和 5，所以还需要一格，候选数也只有 4 和 5，才可以不出现矛盾（如果三格只有 4 和 5 则无法填满三格；如果只有一格有 4 和 5，则 4 和 5 在 **c8** 上总有其中一格数不会出现，于是矛盾）。于是 **r45c8** 其中一格只能有候选数 4 和 5；同理，**r5c2** 因为只有候选数 6 和 8，所以需要且仅需要一格只有候选数 6 和 8，才不会出现矛盾。所以 **r5c89** 其中一格只能有候选数 6 和 8。再观察 **b6**，结构涉及于 **b6** 内还有一格只有候选数 7 和 9，那么还需要且仅需要一格只有 7 和 9 才不会出现矛盾。

这样一来，r4c8 和 r5c89 这三格有一格只有 7 和 9、有一格只有 6 和 8、有一格只有 4 和 5。这样刚好用完这三格，所以这样看来，c8 总会出现 45 数对、r5 总会出现 68 数对、b6 总会出现 79 数对，于是删除 c8、r5、b6 其余位置各自出现的数对而得到的删数，图中红色的均为删数。

这样的结构就是 SDC 的拓展，把原本涉及的两个区域变为涉及了三个区域，所以称为**三区域分布式跨区数组**(Three-sector DDS)，另也可以直接叫作**二度 SDC**(SDC Degree 2)。注意，结构涉及 n 个区域，则称为“(n-1)度 SDC”。最基本的 SDC 也可以被称为**一度 SDC**(SDC Degree 1)，千万不要搞错了名字。

可以看到，这样的结构最多只能到达二度，因为盘面只可能出现行、列、宫三种区域类型，所以不可能出现第四个维度，使得结构变为三度的。

4-1-2 段 SDC/多米诺链 (Domino Chain)

除了这样从涉及区域来拓展结构的，还可以直接将推导情况接起来形成链条形式。

	5 6	2 5 6	4	8	9	1 2 6	1 5 7	3 6 7	1 3 5 6
7	5 6	1	2 5 6 7 8	2 5 6	3	2 6	5 7 8 9	6 7 8 9	4
	5 6 9	3	5 6 8	7	4 5 6 8	1 4 6	2	6 8 9	1 5 6 8 9
1 3 4 6	4 6 8		1 6	2 3 4 6 7 9	4 6 7 8	2 3 6 7 8	1 4 6 8 9	5	1 2 6 8 9
2	4 5 6 8	9	4 5 6 8	1	4 5 6 8	3	6 8	7	
1 3 4 5 6	7	1 5 6	2 3 4 5 6 8 9	2 3 4 5 6 8	4 5 6 8	2 3 6 8	1 2 6 8 9		
1 5 6 7	2 5 6	3	1 6	3 6 7	9	5 7 8	4	5 8	
8	4 5 6 9 7	5 6	4 6	2	4 6 7	5 7 9	1	5 9	3
1 4 7	2 4 9 7	1 2	1 3 4	8	5	6	2 3 7 9	2 3 9	

如图所示，我们从 r5c468 起推。r5c46 其中一格只能有 6 和 8，和 r5c8(68)构成 68 数对（r5c46 都没有 6 和 8 和都只有 6 和 8 都是矛盾的，这一点和上面的思路是一致的，这里就不阐述了）。因为 r5c46 其中一格只有 6 和 8，所以另一格就不会有 6 和 8、只有 4 和 5。此时 r46c5 其中一格也只能有 4 和 5，构成 45 数对；而 r46c5 其中一格只有 4 和 5，所以另外一格则只有 6 和 7，此时和 r7c5 形成 67 数对。所以推导过程可以得到 r5 形成 68 数对、b5 形成 45 数对、c5 形成 67 数对，于是其余单元格对应数对的删数就都可以删除了。

这个结构和刚才的结构不太一样的地方是，多度 SDC（多区域分布式跨区数组）是“发散形”的，而这个结构则是“链条形”的。所以这个结构也可以看作多段不同的待定数组共同构成的结构，并称为**多段 SDC 或多米诺链 (Domino Chain)**，是不是很像多米诺骨牌那样一个一个顺次倒下的感觉呢？另外，上图由三段构成，所以是**三段 SDC**。

8	4 5	5	3	4	3	1	6	4 5	2
1 2 3	1 2	7 9	7 9	7	7	2 3	4	4 5	3
4	4	6	6	4	5	4	8 9	7 8	7 8
2 3	2	5	3	2 3	2 3	2 3	1	4 5	5
4 5	4 5	7	9	7	8	8	7 8	7 8	7 8
2	2	1	5	2 3	2 3	2 3	2	6	4
7	7	7 8	6	6	6	6	8 9	7 8 9	7 8 9
4 5	3	5	4	6	8	4	5	1	5 6
7	7	7	7	9	9	9	9	7	6
9	4 5 6	5	1 2	1 2	7	3	2 3	5 6	8
1	3	1	2	6	5	4	3	1	3
7	7 8 9	2	7 8	7 9	7 9	7 9	8	8	6
1 3	1	5	3	1 2	2	2	2 3	1	3
5	5	8 9	8 9	8	4	7	8	6	8
6	1	4	3	1	2	2	2	5	9
7	7	7	7	7	8	8	8	8	8

如图所示，c7 之中有两格只有候选数 4、8、9，则还需要一个单元格只有候选数 4、8、9 才不会出现矛盾；发现 r45c7 其一是 4、8、9 就可以，于是 r45c7 另外一格只有候选数 2 和 5，于是还需要恰好一格只有候选数 2 和 5 才不会出现矛盾，此时发现 r6c89 其中一格只有 2 和 5 就可以，于是另外一格则只有候选数 6 和 8，此时需要 r6c23 其中一格只有候选数 6 和 8 才不会矛盾；r6c23 其中一格只有候选数 6 和 8，所以另外一格则只有候选数 2、4、5，于是和 b4 内只有候选数 2、4、5、7 的 r4c1、r5c13 三格共同构成 2457 四数组。

所以总的来说，c7 总会出现 489 三数组、b6 总会出现 25 数对、r6 总会出现 68 数对、b4 总会出现 2457 四数组。所以各自对应的区域下的数组的删数就都可以删除掉了。

这个结构在推导时被分为四个部分，所以称为**四段 SDC**。

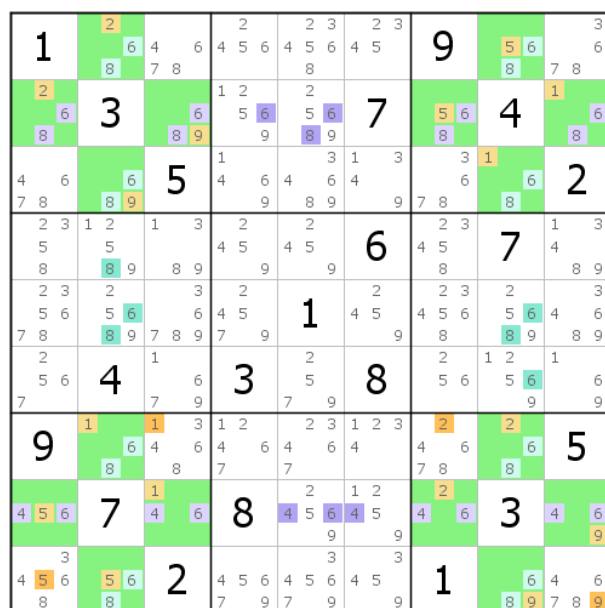
那么，这里再来一个练习，请自行观察。这个结构有些复杂——**六段 SDC**。

1	3	3	3	2	1 2	5	2 3	4
7	7	8	8 9	8 9	2	2	8 7	9
4	4 5 6	4 5 6	3	4	6	1	7	9
1	3	2	9	4	6	7	5 6	5
4	4	4	4	8	8	8	8	8
5	3	1	2	6	4	7	2	2
6	8	2	5	7 9	7 9	1	4	3
4	3	7	1	2 3	2	5	5	6
9	9	9	9	8	8	8	8	8
2 3	3	3	2	8	1 2	9	6	1 2
4 5	4 5	4 5	4	4	7	4 5	2	5
7	7	7	7	7	7	7	7	7
2	1	4 5 6	4	6	4	3	4 5	2
7 9	7 9	7 9	7 9	7 9	7 9	7 9	7 8	7 8
8	6 4	6	5	1 2	1 2	4	2 3	1 2
7	7	7	7	7	7	7	7	7

大致的推导过程是从 r3、b2、c5、b8、r7、b7 的顺序推导的哦！就提示到这里了！当然了，从推导过程之中，你也可以发现，**多段 SDC 的推导是不论方向的，所以倒着推导也是可以的。**

它非常像是链，但和链不同的是，链如果不首尾拼接形成环的话，是无法删除那么多数字的，而多段 SDC 虽然很像是链的形式顺次推导，但是它在不封闭的情况下也是可以删除很多数字的哦。那么，多段 SDC 封闭起来构成环路会怎么样呢？

4-1-3 科尔扎斯环/多米诺环 (Kurzhal's Loop/Domino Loop)



如图所示，我们只看绿色的单元格。我们这么去想，任找一点，比如 r2c13，不妨假设其中一格只有候选数 68，则 r2c79 其中一格只有候选数 6 和 8，构成 68 数对，而另一格则只有 1 和 5，与 r13c8 构成 15 数对，另外一格则是 68，和 r79c8 其中一格形成 68 数对，另外一格则只有 29，和 r8c79 其中一格构成 29 数对，而 r8c79 另外一格则只有候选数 4 和 6，则会和 r8c13 其中一格构成 46 数对，则另外一格是 15，会和 r79c2 其中一格构成 15 数对，另外一格则只有候选数 6 和 8，会和 r13c2 其中一格构成 68 数对，而另外一格则只有候选数 2 和 9，会和 r2c13 其中一格构成 29 数对。而最开始 r2c13 假设的其中一格是候选数 6 和 8，那另外一格必然肯定只有候选数 2 和 9 了。

这样一来就会构成涉及 r28c28b1379 八个区域的环形八段 SDC，且各自区域上恰会构成一个简单的数对。这样的结构是多米诺链的环形版本，所以直接可以成为**多米诺环 (Domino Loop, 简称 DL)**，或称为**科尔扎斯环 (SK Loop, 简称 SK 环)**。

结构是由一位叫 Stephen Kurzhal's 的人发现的技巧，所以为了纪念他，采用了他的名字直接称呼这样的环，即 S.K. Loop。

另外，我们可以对这样的多段 SDC 的推导过程利用数学符号的方式简写，而不是用上面复杂的文本描述。

```
r2c1379(68) -> {r13c8, r2c79}(15) ->
r1379c8(68) -> {r79c8, r8c79}(29) ->
r8c1379(46) -> {r79c2, r8c13}(15) ->
```

`r1379c2(68) -> {r13c2, r2c13}(29)`

另外，多米诺环还有一个示例。

9	8	1 2 3	7	1 2 3 4 6	2 3 5 6	1 2 4 5	1 3 4 5 6	2 3 5 6
7	1 2	6	1 2 3 4 5	1 2 3 4	2 3 5	8	1 3 4 5	2 3 5
1 2 3	5	4	1 2 3 9	1 2 3 6	2 3 6	1 2 9	1 3 6	2 3 6
6	1 2 4	1 2 5	8	1 2 3	2 3 5	3	4 5 7	5 6 9
1 3 4 5 8	7	9	7	9	7	9	7	9
1 3 4 5 8	7	9	7	9	7	9	7	9
2 3 5	8	7	9	7	9	7	9	7
1 2 4 5	3	1 2 5	6	1 2 3	2 3 5	7	1 5	2 5
1 2 4	1 2 4	1 2 6	7	8	9	1 2 3	6	9
2 5	2	2	5	6	7	8	9	1
1 2 4	1 2 4	1 2 6	7	8	9	1 2 3	6	9
2 5	2	2	5	6	7	8	9	1
8	7	9	7	8	9	7	8	9

它的表述如下。

`r4c5689(7) -> {r4c89, r56c7}(459) ->`
`r5689c7(6) -> {r7c89, r89c7}(125) ->`
`r7c5689(8) -> {r7c56, r89c4}(249) ->`
`r5689c4(3) -> {r4c56, r56c4}(125)`

4-2 MSLS 的定义和使用

我们正式进入网的学习。不过首先，我要拿出一个大家基本上都熟悉的例子：被网上“戏称为”世界最难题的数独题目。

4-2-1 MSLS 的定义

8	1 2 4 6	2 4 5 6 9	2 3 4 7	1 2 3 5 7	1 2 3 4 7	1 3 5 6 9	4 5 7 9	1 3 4 5 6 7 9
1 2 4 5 9	1 2 4	3	6	1 2 5 7 8	1 2 4 8	1 5 8 9	4 5 7 8 9	1 4 5 7 9
1 4 5 6	7	4 5 6 4	3 8	9	1 3 4 8	2	4 5 8	1 3 4 5 6
1 2 3 4 6 9	5	2 4 6 9	2 3 8 9	2 3 6 8	7	1 6 8 9	2 4 6 9	1 2 4 6 9
1 2 3 6 9	1 2 3 8 6	2 6 9	2 3 8 9	4	5	7	2 8 9	1 2 6 9
2 4 6 7 9	2 4 6 8 7 9	2 4 6 7 9	1 8	2 6 8 9	2 6 8 9	5 6 8 9	3	2 4 5 6 9
2 3 4 5 7	2 3 4	1	2 3 4 7 9	2 3 7	2 3 4 9	3 5 9	6	8
2 3 4 6 7	2 3 4 6	8	5	2 3 6 7 9	2 3 4 6 9	3 9	1	2 3 7 9
2 3 5 6 7	9	2 5 6 7	2 3 7 8	1 2 3 6 7 8	1 2 3 6 8	4	2 5 7	2 3 5 7

这个题把它的全部候选数都标注上去的话，就是这样的，不过这个题目，在前面学习到的技巧之中，是无法解决掉的，也包括飞鱼导弹（或者我自己没找到）。

不卖关子了，接下来我们直接来看这个例子用什么网怎么操作。

首先我们标记出来一系列单元格，如下图所示。

8	1 2 4 6	2 4 5 6 9	2 3 4 7	1 2 3 5 7	1 2 3 4 7	1 3 5 6 9	4 5 7 9	1 3 4 5 6 7 9
1 2 4 5 9	1 2 4	3	6	1 2 5 7 8	1 2 4 8	1 5 8 9	4 5 7 8 9	1 4 5 7 9
1 4 5 6	7	4 5 6 4	3 8	9	1 3 4 8	2	4 5 8	1 3 4 5 6
1 2 3 4 6 9	5	2 4 6 9	2 3 8 9	2 3 6 8	7	1 6 8 9	2 4 6 9	1 2 4 6 9
1 2 3 6 9	1 2 3 8 6	2 6 9	2 3 8 9	4	5	7	2 8 9	1 2 6 9
2 4 6 7 9	2 4 6 8 7 9	2 4 6 7 9	1 8	2 6 8 9	2 6 8 9	5 6 8 9	3	2 4 5 6 9
2 3 4 5 7	2 3 4	1	2 3 4 7 9	2 3 7	2 3 4 9	3 5 9	6	8
2 3 4 6 7	2 3 4 6	8	5	2 3 6 7 9	2 3 4 6 9	3 9	1	2 3 7 9
2 3 5 6 7	9	2 5 6 7	2 3 7 8	1 2 3 6 7 8	1 2 3 6 8	4	2 5 7	2 3 5 7

这样我们就标注了 20 个单元格。我们之前学到的 SDC 的拓展，不论是多段 SDC 还是

这里我们来看这 20 个单元格的所有候选数，这些候选数能不能像之前那些 SDC 那样，每一个候选数都可以明确地找到自己所属于的区域呢？

[illegible]

那怎么去选择每一个候选数的所在区域呢？比如 **r1c5(7)**，因为行上只有唯一一个单元格可能有 7 的填数位置，按照 **SDC** 拓展版本那样，如果一个区域下只有唯一一处有这个数，就完全无法构成合适的结构。所以这样看是不行的，换言之，只能将候选数安排到列上。

506

8	1 2	2	2 3	1 2 3	1 2 3	1 3	1 3	1 3
4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6
1 2	1 2	3	6	1 2	1 2	1 2	1 2	1 2
4 5	4 5	4 5	4 5	4 5	4 5	4 5	4 5	4 5
9	9	9	9	9	9	9	9	9
1	7	4 5 6	4	9	1 3	2	4 5 6	1 3
4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6
1 2 3	1 2 3	2	2 3	2 3	2 3	1	2	1 2
4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6
9	9	9	9	9	9	9	9	9
1 2 3	1 2 3	1 2 3	1 2 3	1 2 3	1 2 3	1 2 3	1 2 3	1 2 3
4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6
9	9	9	9	9	9	9	9	9
2	2	2	2	2	2	2	2	2
4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6
9	9	9	9	9	9	9	9	9
2 3	2 3	2 3	2 3	2 3	2 3	2 3	2 3	2 3
4 5	4 5	4 5	4 5	4 5	4 5	4 5	4 5	4 5
7	7	7	7	7	7	7	7	7
2 3	2 3	2 3	2 3	2 3	2 3	2 3	2 3	2 3
4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6
7	7	7	7	7	7	7	7	7
2 3	2 3	2 3	2 3	2 3	2 3	2 3	2 3	2 3
4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6	4 5 6
7	7	7	7	7	7	7	7	7

结论是成立的，而且红色的删数是可以保证正确性的，我们可以安全删除掉它们。比如刚才我们确定了 **r8** 上有 3 和 6，所以 **r8** 结构涉及的区域单元格的候选数 3 和 6 均可以被删除。

那，又是为什么可以这么确定删数呢？外围的数字又是个什么意思呢？

4-2-2 MSLS 的原理

下面我们用简单的几句话阐述一下它到底为什么成立。

我们利用鱼的删数原理思路可以进一步地推广，换句话说，鱼只会涉及行、列、宫，因为鱼只能是同数的；而网涉及不同数字，所以还会比鱼多出一种衡量区域：单元格。

我们这么去想这个问题。我们尝试把外围的数字看作“广义的删除域”，而把单元格看作“广义的定义域”。定义域意味着，每一个定义域下只可能有且仅有唯一一个正确的数；删除域意味着，每一个删除域下最多都只能有一个正确的数。那么 20 个单元格都被看作定义域的话，就有 20 个定义域，即恰有 20 个数字正确；而我们把外围的数字，挨个在所在区域上算作对应的删除域区域的话，也会有 20 个删除域。删除域下数字最多只有一个正确的，比如说，我们把 **r1** 上涉及的两个删除域区域 **r1(1)**、**r1(3)** 和 **r1(6)** 都写出来，这就表示 **r1** 上最多只有一个位置是 1、最多一个位置是 3、最多一个位置是 6。

而 20 个删除域区域就意味着，最多有 20 个数填入结构之中，而定义域保证，结构恰有 20 个数。所以这么一来，就没有多出来的数了。换句话说，这些数最终都是全部会填入到结构之中的。所以，删除域下的数字（所在区域的其余候选数）均可删除。

这下知道为什么网的英文名的直译是“多区域（跨区）数组”了吧。因为它的结构不同于 SDC 的拓展，

另外再提及一些术语。

- **定义单元格**（简称**定义格**，**Defining Cells**）：即结构涉及的单元格，每一个单元格都称为一个定义格。即定义网结构的单元格。也称为**网定义域**（**Defining Sets For MSLS**）。
- **链接点**（**Link**）：比如上述示例之中，标注在盘面外的数字。位于行上的，称为**行链接**

点 (Row Link)，位于列上的，称为**列链接点** (Column Link)，当然也存在**宫链接点** (Block Link)。

- **网定义域** (Defining Sets For MSLS)：同定义格。
- **网删除域** (Secondary Sets For MSLS)：链接点所在的所有区域，称为网删除域。

4-2-3 小练习

下面我将会给出一个题目，这个题目里存在一共 3 个不完全相同的网结构（当然网结构的位置不同，删数就不一定完全一致了）。我会给出其中的一个网，并且给出对应的删数在那里，以及所有提供提示的涂色信息，而剩下的两个网里，一个我只给出删数和定义单元格的位置，但不给出链接点的分配情况的涂色信息；而还有一个仅仅给出定义单元格的位置，不给出涂色信息和删数，希望你能够独立通过前文的描述完成推理。

9	8	1 2	7	1 2 4 5	3	6	3 1 2 3
5	1 2 6	4	1 2 3 6	1 2 6	3	1 2 3	1 2 3
1 2 6	3	1 2 6	1 2 4 5 6	8	4 5 6 9	1 2 4 5	1 2 7 8 9
7	1 2 6	7	2 4 5 6	9	4 5 6 8	3	1 4 6
1 2 4 6	7	1 2 8	5	4 6 8	4 6 7 8	2	1 7 8 9
1 3 4 6 8	5	2 3 6 8 9	4 6 8	2 3 4 6	1	4 6 7 9	5
2 3 4 6 8	2 3 6 8 9	2 3 6 8 9	2 3 4 6	2 4 6	1	4 6 7 9	5
1 2 4 6	1 2 4 5 6	9	3	4 5 6 7	8	5 6 7	2 6
3 6 7 8	5 6 7 8 9	3 1 8	1 5 6 7	2 4 5 6	5	3 5 6 7 9	4
2 3 4 6 8	2 3 6 8 9	2 3 4 5 6	4 5 6 8	4 5 6 7 8	2 5	1 7 9	2 3 6 9

这是第一个网结构，看起来好像要补充 r1c1 才能算是完整的 4 * 5 的结构，但这个例子里由于 r1c1 被提示数 9 占据了，所以不能算上它，而剩下的部分恰好满足 19 个单元格和 19 个链接点的分配，而且同样满足了全覆盖的要求，所以网是成立的。

9	8	1 2	7	1 2 4 5	3 4 5	6	3 4 5	1 2 3
5	1 2 6	4	1 2 3 6	1 2 6	3 6 9	1 2 7 9	3 7 8 9	1 2 3
1 2 7	6 3	1 2 6 7	1 2 4 5 6	8	4 5 6 9	1 2 4 5 9	4 5 9	1 2 9
1 2 4	6 7	1 2 6 8	2 4 5 6 8	9	4 5 6 8	3 4 6 8	1 6 8	6
1 3 4 6 8	4 6 9	5	4 3 6 8	4 6 7	3 4 6 7 8	1 4 7 9	2 6 7 8 9	1 6
2 3 4 6 8	2 6 9	2 3 6 8 9	2 3 6 8	2 4 6 7	1 4 6 7 8	4 7 9	4 6 7 8 9	5
1 2 4	1 2 6 7	1 2 6 7	9	3	4 5 6 7	8	5 6 7	2 6
3 7 8	6 5 6 9	3 7 8 9	1 5 6 8	1 5 6 7	2 5 6 7	5 7 9	3 5 6 7 9	4
2 3 4 6 7 8	2 4 5 6 9	2 3 6 7 8 9	2 3 6 7 8 9	2 3 6 7 8 9	2 3 6 7 8 9	1 4 5 6 7 8 9	2 5 6 7 8 9	3 6 9

这个网的示例里只有 16 个单元格，但依旧构成了网，希望你能找到链接点的情况。

9	8	1 2	7	1 2 4 5	3 4 5	6	3 4 5	1 2 3
5	1 2 6	4	1 2 3 6	1 2 6	3 6 9	1 2 7 9	3 7 8 9	1 2 3
1 2 7	6 3	1 2 6 7	1 2 4 5 6	8	4 5 6 9	1 2 4 5 9	4 5 9	1 2 9
1 2 4	6 7	1 2 6 8	2 4 5 6 8	9	4 5 6 8	3 4 6 8	1 6 8	6
1 3 4 6 8	4 6 9	5	4 3 6 8	4 6 7	3 4 6 7 8	1 4 7 9	2 6 7 8 9	1 6
2 3 4 6 8	2 6 9	2 3 6 8 9	2 3 6 8	2 4 6 7	1 4 6 7 8	4 7 9	4 6 7 8 9	5
1 2 4	1 2 6 7	1 2 6 7	9	3	4 5 6 7	8	5 6 7	2 6
3 7 8	6 5 6 9	3 7 8 9	1 5 6 8	1 5 6 7	2 5 6 7	5 7 9	3 5 6 7 9	4
2 3 4 6 7 8	2 4 5 6 9	2 3 6 7 8 9	2 3 6 7 8 9	2 3 6 7 8 9	2 3 6 7 8 9	1 4 5 6 7 8 9	2 5 6 7 8 9	3 6 9

最后是同样这个题目的第三个网结构。

那么，下一节我们将继续讲解，网结构的复杂变体类型。

4-2-4 MSLS 的使用

4-2-4-1 宫内网和交叉网（Franken & Mutant MSLS）

你知道吗？之前讲到的题目，在完成基本网的删数后，继续往下做，会得到两三个数对的删数，然后可以出数，然后题目依然比较难（虽然网的使用有降低题目的难度，不过依然会比较难）。

不过，有一些思维可以帮助我们在同一个技巧之中找到更多的删数。诸如标准鱼删除不了的数，可以借助它的变型版本，诸如宫内鱼和交叉鱼来为宫建立定义域区域，然后删数。

网其实也是由宫内和交叉版本的。

我们依然拿出之前的示例作出解释。

8	1 2	2	2 3	1 2 3	1 2 3	1 3	1 3	1 3
	4 6	4 5 6	4	5	4	5 6	4 5	4 5 6
1 2	1 2			1 2	1 2	1	4 5	1
4 5	4			5	4	8	7 8 9	4 5
	9			7 8	8	8 9	7 8 9	7 9
1			3		1 3		1	3
4 5 6	7	4 5 6	4		4	4 5	4 5 6	
			8		8	8		
1 2 3	5	4 6	2 3	2 3	6	7	1 2	4 6
4 6		9	8 9	8		8 9	8 9	9
1 2 3	1 2 3	2	2 3	4	5	7	2	1 2
	6		6				8 9	6
	9	8	8 9	8 9			8 9	9
2	2	2	2	2	2	3	2	2
4 6	4 6	4 6	4 6	4 6	4 6	5 6	4 5 6	4 5 6
7 9	8	7 9	8	8 9	8 9	8 9	9	9
2 3	2 3	1	2 3	2 3	2 3	3	6	8
4 5	4		4	7 9	4	9	9	
7					9			
2 3	2 3	8	5	2 3	2 3	3	1	2 3
4 6	6 4	6		6 4	6	9	7 9	
7				7	4 6	9		
2 3	2 3	2 3	2 3	1 2 3	1 2 3	2 3	2 3	2 3
5 6	5 6	5 6	5 6	6	6	5	5	5
7		7 8	7 8	8	8	7	7	

如图所示，我们只去看绿色的单元格。我们按照网的思路，对全部绿色单元格内的候选数作出全覆盖策略：每一个数都找一个它所属的行和列。发现的结果如下所示：

- r3(45)、r6(68)、r9(257)
- c2(24)、c4(38)、c7(59)
- b1(16)、b5(29)、b7(36)、b9(3)

结构一共涉及了 10 个区域（r369c247b1579），而恰好使用了 20 个单元格，且一共包含 20 个不同的数字分配情况（称为链接点）。所以我们可以知道，20 个单元格恰只能填入这 20 个不同的数字，不存在多余的数。所以可以删除共同对应的区域下的对应候选数。比如，我们因为可以确定 c2 的 2 和 4 一定填入到结构之中（r12678c2），所以 c2 的其余单元格一定不可填入 2 和 4。所以删除它们。

图中所有不在结构之中的深色候选数（红、青、紫）均为删数。为了让读者看图更为清晰和明白，使用了颜色作为区分。

那么，根据鱼的命名规范，标准网其实也属于一种交叉网，因为交叉（Mutant）一词被定义为定义域区域或删除域区域同时涉及行和列的结构。而很明显，链接点按之前的说明，可以被视作为删除域区域。所以它肯定是一种交叉的结构。这里为了规避理解，网的宫内和交叉被定义为：

- 链接点只涉及行和列的，称为**标准网**（Basic MSLS）；
- 链接点涉及宫，但不同时涉及行和列的，称为**宫内网**（Franken MSLS）；
- 链接点同时涉及行、列、宫的，称为**交叉网**（Mutant MSLS）。

那，这里给出的是交叉网，因为行、列、宫全部都涉及到了。那么，宫内网结构有哪些呢？最基础的 SDC 就可以使用宫内网的理解思路来思考。

1	5 6	9	2	3 6	7	4 8	3 5	8 3
7	3	2	8	5	4	9	1	6
8	5 6	4	3 9	1	6 9	3 7	2 5	2 3
9	7 8	6	5	4	2	3 7	8	1
2	4 8	3	7	9	1	6	4 8	5
5	4 7	1	6	8	3	4 7	2 9	2 9
6	2	8	3 9	3 7	5	1	7 9	4
3	1	7	4	2	6 9	5	8 9	8 9
4	9	5	1	3 6	8	2	3 7	3

这个是之前的例子。你可以使用网的视角：

涉及 4 个单元格，且恰好有 4 个链接点。3 和 8 是宫链接点，4 和 7 是列链接点，恰好实现了全覆盖。所以可以删除对应区域的对应候选数。

不过说起来，最上面的图甚至可以看成是一个多段 SDC（多米诺环），这就好像是部分交叉鱼可以看作是环一样。

4-2-4-2 非矩形网（Non-rectangular MSLS）

由于网结构类比于鱼结构来说，定义域区域变为了单元格（定义格），所以形态就不一定非要呈现鱼那样的矩形状态了。

2 3	2	2 3	2	1 2	1	1 2 3	3	4
5 7	5 7	5 8	6 7	5 6 7	6 7	5 6 7	5 6	
2 4	1	2 5	2 4	2 4 5 6	3	2 5 6	9	2 5
6	2 4 5	2 3 5	2 4 7	9	1 4 7	8	3 5 7	1 2 3
1 2 4	2 4 6	1 2 6	3	4 6 7	5	2 4 6 7	4 6 7	2 7 8 9
2 4 5	2 4 5 6	7	1	4 6 8	4 6 8 9	2 3 4 5 6	3 4 5 6	2 3 5 8 9
4 5 9	3	5 6 8 9	4 6 7 8 9	4 6 7 8	2	4 5 6 9	1	5 7 8 9
1 2 3 5	2 5 6	4	2 6 8 9	1 2 3 6	1 6 8 9	7	3 5 8	1 3 5 8 9
1 3 7	9	1 3 6	5	1 3 4 6 7 8	1 4 6 7 8	1 3 4	2	1 3 8
8	2 5 7	1 2 3 5	2 4 7 9	1 2 3 4 7	1 4 7 9	1 3 4 5 9		6

如图所示，结构涉及 21 个单元格，其中的 E5 就是结构“多出来”的部分。之所以“多出来”三个字打了引号，是因为结构需要它们，它们并不是多余的，但对于矩形形状来说，就多了一块。

你可以尝试自行寻找结构的链接点。只要链接点实现了全覆盖，并且所有恰有 21 个链接点，就是可以的。答案并不一定是唯一的。

4-2-4-3 其它饱和网（Saturated MSLS）

饱和一词本出现在鱼结构里，不过我们套用到这里来了。不过这并不重要，我们下一节才会讲到较难的部分，所以这里仅表示一个意义：定义格数和链接点数是一样的。

7	4 5	1 4 5	1 3 4 5	6	4 5	2	1 3
	3	2	1 2	5	1 3	7	1 3
4	6 4	6 4	8 9	7	2	4 5 6	8 9
4	3	1	4 5	2	7	5 6	4
4	8 9	6	7 9	6	1 3	2	5 6
5	4	2	4 5	6	4	1	3
2	3	4 5	6 7	7	1 5	4 5	1 5
4	6	4 5 6	9	8	4 5	3	2
1	8	5	4	2	5	7 9	6

如图所示，这个网的删数实际上很简单，因为我们细数结构里的数字的分布情况，就可以发现，定义格一共 8 个，链接点也有 8 个，所以全部链接点对应的区域都可以删数，即图中所有标注在结构外部的涂色候选数。

4-2-4-4 不饱和网（Unsaturated MSLS）

在鱼结构里，我们知道了，秩可以轻松判定一个鱼结构是否是饱和的。如果鱼结构不饱和，我们计算的秩就永远得不到 0 的结果；即使我们发现，有一部分情况满足等于 0 但删数依旧不是完整的时候，它们还具有自己特殊的使用逻辑和套路，我们如果把这个内容脱光到网里，到底会如何呢？

之前讲过一点关于鱼结构的术语——秩的内容。我们简单复习一下：

鱼的秩等于删除域区域数减去定义域区域数。一般来说，一个符合要求的鱼结构，秩必然大于等于 0（小于 0 是不够填数的，直接矛盾）。那么网结构呢？其实，网也是有秩这一个说法的。实际上，所有数独技巧只要被包装为强弱区域的形式时，就均有秩一说，不过我们不过多说明这一点，这一点内容可以类比于鱼和网来自行理解和操作。

网也是一样，不过公式改为了这样：如果我们用字母 D 表示定义格数，L 表示链接点数时，网的秩有如下公式：

$$\text{Rank} = L - D$$

所以我们可以知道，上面的所有题目之中，结构的秩均为 0（定义格数和链接点数一样）。那么，是否存在类比于鱼结构的情况——链接点数多于定义格数呢？当然存在了。我们来看一则运用很灵活的例子。

[illegible]

如图所示, 结构的链接点情况表示如下:

- $r_2(345)$ 、 $r_4(345)$ 、 $r_9(36)$;
- $c_1(27)$ 、 $c_2(1)$ 、 $c_4(2)$ 、 $c_6(17)$ 、 $c_9(127)$ 。

一共是 16 个定义格和 17 个链接点。秩为 $17 - 16 = 1$ 。由于秩不为 0，所以不可以像之前那样直接得到删数。那么原本可能存在的删数（图上不属于定义格外的候选数，红色和紫色）都是可能删掉的数字。橙色数字则是删掉了这些可能删的数后，产生的 b9 的 127 三数组的删数。

先不管那么多。我们先来思考一个问题。红色和紫色的删数之中，但凡有一个是对的，会如何？如果但凡有一个删数是对的，都会排除掉当前所在区域下的一个链接点，这样就变成了秩为 Θ 的网，所以其他的候选数（红色和紫色）都可以删掉了。

注意，这一步只是我们的猜想和假设，不代表这些数就一定在这一步就可以删掉了哈。这里特别强调一下。看能不能删数是需要找到与之矛盾的情况出现，才可以确定原假设错误，这样才可以直接删数。

[illegible]

接着我们发现，如果我们删掉了红色和紫色的候选后，橙色的肯定可以删除了（127 数组），于是，观察 **r89**，我们可以发现，数字 1、2、7 恰好只剩下六格可填：**r89c359**。于是这六格一定形成了一个拓展矩形结构的致命形式。这样一来就矛盾了。所以原假设错误了，故可以删除掉这些候选数。

另外，我们可以将上述结构表述为 **16|17网**或**+1网**（+1表示结构的秩为1， $17 - 16 = 1$ ）。前者表示定义格数，后者表示链接点数，中间用反斜杠表示。

9	2	<div><div>1</div><div>6</div></div>	<div><div>1</div><div>4</div></div>	<div><div>3</div></div>	8	<div><div>1</div><div>4</div><div>6</div></div>	5	<div><div>1</div><div>4</div><div>3</div></div>	7
3	<div><div>1</div><div>4</div><div>8</div></div>	<div><div>1</div><div>5</div><div>8</div></div>	9	<div><div>2</div><div>4</div><div>5</div></div>	7	<div><div>1</div><div>2</div><div>4</div></div>	<div><div>1</div><div>2</div><div>4</div></div>	<div><div>1</div><div>2</div><div>3</div><div>9</div></div>	6
<div><div>1</div><div>4</div><div>7</div></div>	<div><div>1</div><div>4</div><div>6</div><div>7</div></div>	<div><div>1</div><div>5</div><div>6</div><div>7</div></div>	<div><div>1</div><div>3</div><div>4</div></div>	<div><div>2</div><div>3</div><div>4</div><div>5</div><div>6</div></div>	<div><div>1</div><div>2</div><div>3</div><div>4</div><div>5</div><div>6</div></div>	8	<div><div>1</div><div>2</div><div>3</div><div>4</div></div>	<div><div>1</div><div>2</div><div>3</div><div>4</div><div>9</div></div>	<div><div>1</div><div>3</div><div>4</div><div>9</div></div>
1	1	4	<div><div>1</div><div>3</div><div>8</div></div>	<div><div>2</div><div>3</div><div>5</div><div>6</div><div>9</div></div>	<div><div>1</div><div>2</div><div>3</div><div>5</div><div>6</div><div>9</div></div>	<div><div>1</div><div>2</div><div>6</div><div>9</div></div>	<div><div>1</div><div>2</div><div>8</div><div>9</div></div>	<div><div>1</div><div>8</div><div>9</div></div>	
6	<div><div>1</div><div>8</div><div>9</div></div>	<div><div>1</div><div>8</div></div>	7	<div><div>2</div><div>4</div><div>9</div></div>	<div><div>1</div><div>2</div><div>4</div></div>	3	<div><div>1</div><div>2</div><div>4</div><div>8</div><div>9</div></div>	5	
2	5	3	<div><div>1</div><div>4</div><div>8</div></div>	<div><div>4</div><div>6</div><div>9</div></div>	<div><div>1</div><div>6</div></div>	<div><div>1</div><div>4</div><div>6</div><div>7</div><div>9</div></div>	<div><div>1</div><div>4</div><div>7</div><div>8</div><div>9</div></div>	<div><div>1</div><div>4</div><div>9</div></div>	
<div><div>4</div><div>7</div></div>	3	2	5	1	9	<div><div>4</div><div>7</div></div>	6	8	
5	<div><div>1</div><div>4</div><div>7</div><div>6</div></div>	<div><div>1</div><div>6</div><div>7</div></div>	2	<div><div>4</div><div>7</div></div>	8	<div><div>1</div><div>4</div><div>7</div><div>9</div></div>	<div><div>1</div><div>3</div><div>4</div><div>9</div></div>	<div><div>1</div><div>3</div><div>4</div><div>9</div></div>	
8	<div><div>1</div><div>4</div><div>7</div></div>	9	6	<div><div>3</div><div>4</div><div>7</div></div>	<div><div>3</div><div>4</div></div>	<div><div>1</div><div>4</div><div>7</div></div>	5	2	

如图所示，我们可以得到一个强关系：

$$r8c35=r79c7(7) \Rightarrow r8c78 \neq 7$$

如果这样两个节点同假时，结构就会同时少两个链接点，原本是 $8 \setminus 9$ 的结构，就会变为 $8 \setminus 7$ ，于是秩变为 $7 - 8 = -1 < 0$ ，所以矛盾。所以 $r8c35(7)$ 和 $r79c7(7)$ 至少有一个成立，删除 $r8c78(7)$ 。

4-2-4-6 SK 环的拓展

显然，在前文的描述里，我们找到了一种这样的结构，而现在我们来看一个长相是多米诺环，但实际上删数不只是多米诺环基本删数的构型。

9	2	4 5	2 3	3	3	4 5	4	6	1
2	7 8	7 8	5 6	5 6	8	8	7	5 6	8
5	3	1	1 2	1 2	5 6	5 6	4	7	5 6
8	8	8	8 9	8 9	8 9	8 9	8 9	8	8
4 5	1	6	1	1 3	1 3	2	4	4 5	3
7 8	7 8	7 8	5	5	5	8 9	8 9	8	8
4 6	5	1	3	2	1	1	4	6	4 6
7 8	7 8 9	7 8 9	4	9	4	7 8 9	8 9	7 8	2 3
2 3	1 2	1 3	1	6	1	1 3	1 2	2 3	4
7 8	7 8 9	7 8 9	4 5	9	5	7 8 9	4	7 8	4
2 3	1 2	1 3	1	7	8	1 3	5	2 3	4 6
4 6	6 4	9	4	9	4	4	9	4 6	9
5	2	1	1 3	1 3	6	1	4 5	4 5	7 8
7 8	7 8 9	7 8 9	4 5	8 9	8 9	8 9	8	7 8	7 8
5 6	4	5	1 2	1	5 6	5 6	3	5 6	2
8	8 9	8 9	8 9	8 9	8 9	8 9	8	8	8
1	6	5	2	2 3	3	4 5	4	9	9
7 8	7 8	7 8	4 5 6	4 5	5 6	7 8	7 8	7 8	7 8

如图所示。这个结构是一个多米诺环，不过请仔细观察删数，你可以发现，b1379 里都多出了候选数 8 的删数，而 8 并未在 b1379 里存在链接点。

我们这么去思考这个问题。数字 8 在整体结构内是必然会出现 4 个的，且分属于 b1379 里，换言之，为了保证数独规则不违背，b1379 里就得各有一个 8 的出现，位于这个多米诺环涉及的 16 个单元格里。那么这么去分析的话，8 的删数就很好理解了：因为 b1379 的每一个宫都会存在 8，所以 b1379 里显然是无法放入其它的 8 的，所以删掉它们。

4-3 直观 MSLS?

实际上，大型结构依旧是可以直观的，不过这一点内容很少被提到，是因为结构本身比较庞大，找到候选数情况的网都是很少的，所以对于这一方面的研究并不多。

4-3-1 逻辑

需要弄清楚 MSLS 结构到底是如何直观的，我们就必须思考一个全局性的问题：如何证明得到一部分单元格和另外一个部分的单元格填数是一样的。虽然看起来这个说法好像跟 MSLS 结构没什么关系，但这是直观 MSLS 结构的核心。

不过需要你注意的是，可以直观的 MSLS 结构一般只会针对于一部分特别标准的结构，因为只有这么一些结构才能够快速直观，而各种奇形怪状的结构，我们一般都不会尝试去直观它们，因为它们的结构在候选数视角里都很少被观察到，而且直观技巧必须要求结构是可以快速观察的，我们基于这一点才会有各种奇妙技巧的直观。

		6		7		9		
	1				2			3
3			8	9				
2					4			5
	3							1
		7				8	6	
			9					
				6		1		
5					7			4

如图所示，这是我们直观网结构的一则示例。我们首先为确定值进行分组，细心一点可以发现，我们选出的数字里，橙色的只包含 1 到 5，而紫色的则只包含 6 到 9；而且我们还发现，1 到 5 的这些数字全部都出现在同样的几行里，而紫色的只出现在几列里（当然了，我们也可以反着看：紫色的出现在几行里，而橙色的出现在几列里，这两种视角并不影响什么）。这一点我们将会马上作出说明。

		6		7		2		
	1				2			3
3			3	2				
2					4			5
	3							1
		7				3	6	
			2					
				6		1		
5					7			4

我们尝试把刚才说的东西的其中一种说法按照画线的方式呈现出来，如图所示。请注意一点。图中 **r3c1** 是含有数字 3 的，但为什么我们没有把它涂色为橙色，也算在结构里呢，毕竟它确实在 **c1** 上？这是因为，它目前处于紫色和橙色的交叉处。如果为其分配橙色的话，虽然这种画线模式可以把它合理地安排到，但我们刚才说过的另外一种视角就不是那么容易能画上去：它就会单独占据 **r3** 一行。这并不是我们想看到的；另外，**r9c6** 的确定值 7 也没有涂色，这也是显然的，因为这个数并不是我们结构需要的 1 到 5 的其一。因为橙色我们只勾选了 1 到 5，跟 7 无关，我们不应选中它。

我们尝试思考一下逻辑。我们按照橙色为基准，橙色一共占据了 4 列，所以一共是 36 个单元格，而这 36 个单元格一共需要填入 4 套完整的 1 到 9 的序列。而其中 **r13678c1269** 这样 20 个单元格是属于橙色和紫色线条的交叉处的。我们把这一部分暂时就直接称为“交叉处”。

显然，我们可以利用上述完全一样的说法看出来的是，紫色所占据的 **r13678** 五行里必须填入 5 套完整的 1 到 9 序列。

我们考虑一下交叉处的情况。因为交叉处是紫色和黄色区域共有的部分，这使得我们拥有如下两个式子：

$$\text{紫色区域填数} = \text{交叉处} + \text{剩余紫色区域部分}$$

$$\text{橙色区域填数} = \text{交叉处} + \text{剩余橙色区域部分}$$

我们通过第一个式子可以移项得到：

$$\text{交叉处} = \text{紫色区域填数} - \text{剩余紫色区域部分}$$

然后完整代换到下面这一个式子里：

$$\text{橙色区域填数} = \text{紫色区域填数} - \text{剩余紫色区域部分} + \text{剩余橙色区域部分}$$

现在，我们尝试去用几套 1 到 9 替换上面的式子：

$$4 \text{ 套 } 1 \text{ 到 } 9 = 5 \text{ 套 } 1 \text{ 到 } 9 - \text{剩余紫色区域部分} + \text{剩余橙色区域部分}$$

然后移项

$$\text{剩余紫色部分} = 1 \text{套} 1 \text{到} 9 + \text{剩余橙色部分}$$

现在，我们把剩余紫色部分和剩余橙色部分都用图呈现出来。

		6		7		9		
	1				2			3
3			8	9				
2					4			5
	3							1
		7				8	6	
			9					
				6		1		
5					7			4

我们已经接近于真相了：删数马上就会出现。我们可以发现到的是，最初我们选定的紫色和橙色确定值全部都处于我们涂色的部分里面。这就是为什么我们需要让你选择的数字要放入到非交叉处的地方的原因：因为后续我们需要依赖于这一点得到结论。

现在，我们尝试细数一下我们最初选定的确定值。我们发现，紫色的 25 个单元格里，一共包含 14 个空单元格没有填入数字；而橙色的 16 个单元格里，一共有 5 个单元格没有填入数字。现在，我们借用刚才我们得到的公式来得到结论：

$$\text{剩余紫色部分} = 1 \text{套} 1 \text{到} 9 + \text{剩余橙色部分}$$

既然剩余紫色部分需要一套完整的 1 到 9 和剩余橙色部分的填数构成，那么我们细看结构就会发现，橙色里除了含有一个 7，是不含有额外的 6、7、8、9 的。而且我们发现，紫色区域里也有 7，我们知道的是，等式两边都含有数字 7，所以我们大可使用消去的方式抵消掉这两个 7。这便是在说，橙色目前只有 15 个单元格，而且没有任何 6、7、8、9 了；而紫色区域现在只剩下 24 个单元格了。由于紫色部分目前是必须要含有一组完整的 1 到 9 序列的，所以我们先排除这一组 1 到 9 的影响，即从紫色区域的确定值里划掉一组 6、7、8、9，而且可以不管其它 1 到 5 放置的位置，因为这并不重要。现在，我们再看结构，由于我们刚才在紫色区域通过抵消的操作已经划去了其中一个 7，而且又划掉了一组 6、7、8、9，此时我们还剩的确定值只剩下五个：6、6、8、9、9，而个数也刚好可以完全放入到橙色区域的空格里面。显然，我们是需要它们的，因为橙色区域里必须包含紫色已经有的数字，而我们发现橙色里有 1、2、3、4、5，而没有 6、7、8、9（我们刚才已经抵消了 r9c6 的确定值 7，这里是处于忽略状态的），这就是我们需要优先考虑的地方：正是因为橙色部分没有任何 6、7、8、9，所以我们为了保证等式的成立，我们不得不先把 6、7、8、9 这些根本没出现在橙色区域的数字先填入到里面去，不然 6、7、8、9 在橙色区域并不出现，而紫色里却包含了 6、7、8、9，而且不止一组 6、7、8、9，就不可能满足上述的等式的要求，这便是我要说明的东西。

所以，由于我们不得不优先考虑 6、6、8、9、9 放入到橙色区域里，所以其它的 1、2、3、4、5 就完全无法放到橙色区域里，因为橙色区域里填入 6、6、8、9、9 后就把所有 16 个单元格占满了，再也容不下其它数了。所以，这个直观网的结论是：橙色区域的空单元格里删除所有的 1、2、3、4、5，而紫色区域的空单元格里删除所有的 6、7、8、9。

“紫色区域的空单元格里删除所有 6、7、8、9”这一点在上述内容里没有作出说明，不过很容易就可以理解，因为紫色和橙色的区域的填数目前来说是“互补”的，我们可以得到橙色的结论，紫色部分的结论肯定就可以类比得到，所以这里我们就不再作出说明了。

而实际上，这个结构对应的候选数视角如下图所示。

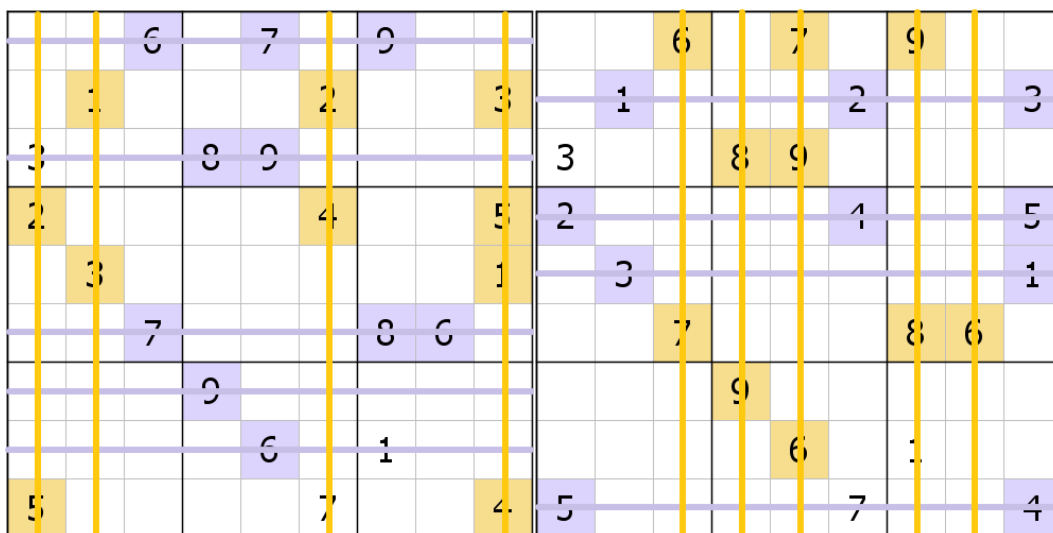
4 8	2 4 5 8	6	1 3 4 5	7	1 3 5	9	1 2 4 5 8	2 8
4 7 8 9	1	4 5 8 9	4 5 6 4 5	2	4 5 6 4 5	3		
3	2 4 5 7	2 4 5	8	9	1 5 6	2 4 5 6 4 5	1 2 4 5	2 6
2	6 8 9	1 8 9 7	1 3 1 3 6	4	3 7	5		
4 8 9	3	4 5 8 9 7	2 5 6 5 8	5 6 8 9	4 7	1	4 7 9	
1 4	4 5 9	7	1 2 3 5	1 2 3 5	1 3 5 9	8	6	2 9
1 4 6 7 8	2 4 6 7 8	1 2 3 4 8	9	1 2 3 4 5 8	1 3 5 8	2 3 5 6 7 8	2 3 5 7 8	2 6
4 7 8 9	2 4 7 8 9	2 3 4 8 9	2 3 4 5	6	5 3 8	1	2 3 5 7 8 9	2
5	2 6 8 9	1 2 3 8 9	1 2 3 1 2 3	7	2 3 6 8 9	4		

它确实符合我们删数的预期效果。

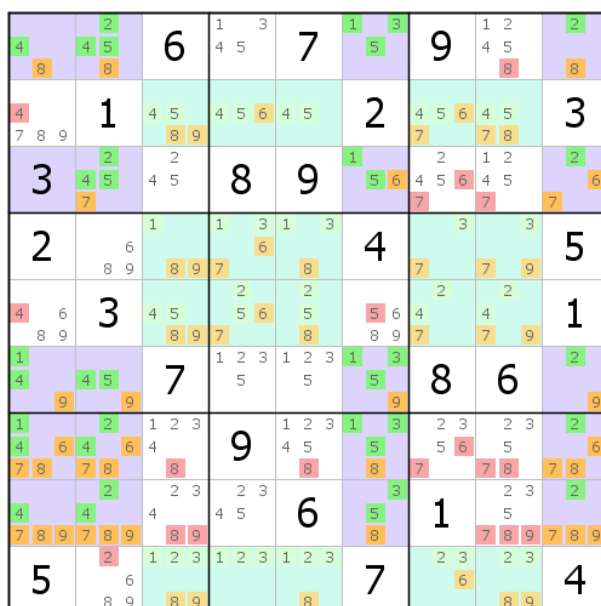
这就是直观网结构想要告诉给你的内容。我们通过“两个完全不相关的填数区域”构造出了填数完全一致，或者多出一组或少一组 1 到 9 的结论，来间接得到填数的关系，然后通过占位推导得到的删数。

4-3-2 MSLS 的互补性

实际上，MSLS 这么大的结构依然是具有互补的。只要我们尝试在最初的假设推导的时候，换做另外一个视角，即把紫色的用列表示，而橙色的用行表示，就会呈现出另外一个对应的网结构，如右图所示（左图是原本的推导视角）。



而对应的候选数视角的网是这样的，如下图所示。



图中，紫色单元格是一组网结构，而淡绿色单元格又是一组网结构。

可以从图上看，删数完全没有发生变化，只是把网的位置变了一下，但并不影响删数，因为换了一下后，原本作为行链接点的地方全部变为了列链接点，而列链接点都变为了行链接点。这便是网的互补。

4-4 其它结构跟网的联系

下面我们来阐述一些其它技巧跟网结构的关联。

4-4-1 鱼直观的原理

第一个我们需要讲到的内容就是鱼结构。实际上，鱼结构的直观我们已经在很前面的内容里说过了，但我们并没有证明之。现在我们尝试把之前的理论进行证明，实际上，之前的理论我们都是通过网结构才能得到证明。

4-4-1-1 先来回顾一下

我们先回顾一下推导过程。

7	6	^{4 5} ₃	1	2	⁵ ₈	9	⁴ ₈	^{4 5} ₃
^{2 3} _{4 5}	² ₅	1	⁵ ₈	⁵ _{7 8}	9	^{4 5} ₃	6	^{4 5} _{7 8}
³ ₅	8	9	4	6	² ₇	2	1	⁵ ₇
² _{4 8}	1	6	9	⁴ _{7 8}	^{2 3} _{4 5}	³ ₄	5	^{2 3} _{4 8}
9	² ₅	7	^{2 3} _{5 8}	^{4 5} ₈	^{2 3} _{4 5}	6	^{2 3} _{4 8}	1
² _{4 5}	3	^{4 5} ₈	6	1	² _{4 5}	7	² _{4 8}	9
¹ _{5 8}	7	2	⁵ ₈	^{4 5} ₈	6	¹ _{4 5}	9	³ _{4 5}
6	4	³ ₅	7	9	1	8	^{2 3} ₅	^{2 3} ₅
¹ _{5 8}	9	⁵ ₈	² _{5 8}	3	² _{4 5}	¹ _{4 5}	7	6

如下方左图所示，我们尝试在结构里找到 $m \times n$ 的矩形区域，需要满足如下的要求：

- 矩形区域涉及的结构里不能含有我们需要找的鱼所涉及的数字的确定值；
- 矩形区域所处的所有行列区域上都不允许含有我们需要找的鱼所涉及的数字的确定值；
- 设矩形区域以外，鱼涉及的数字的确定值总个数为 k ，而矩形涉及大小为 $m \times n$ ，需要满足 $m + n + k = 9$ 。

7	6	^{4 5} ₃	1	2	⁵ ₈	9	⁴ ₈	^{4 5} ₃
^{2 3} _{4 5}	² ₅	1	⁵ ₈	⁵ _{7 8}	9	^{4 5} ₃	6	^{4 5} _{7 8}
³ ₅	8	9	4	6	² ₇	2	1	⁵ ₇
² _{4 8}	1	6	9	⁴ _{7 8}	^{2 3} _{4 5}	³ ₄	5	^{2 3} _{4 8}
9	² ₅	7	^{2 3} _{5 8}	^{4 5} ₈	^{2 3} _{4 5}	6	^{2 3} _{4 8}	1
² _{4 5}	3	^{4 5} ₈	6	1	² _{4 5}	7	² _{4 8}	9
¹ _{5 8}	7	2	⁵ ₈	^{4 5} ₈	6	¹ _{4 5}	9	³ _{4 5}
6	4	³ ₅	7	9	1	8	^{2 3} ₅	^{2 3} ₅
¹ _{5 8}	9	⁵ ₈	² _{5 8}	3	² _{4 5}	¹ _{4 5}	7	6

这个示例就满足上述的三点要求：矩形区域里没有 5，矩形区域所在行列上的任何一点也都没有 5，而在区域外的其它部分含有一个 5 ($k = 1$)，且满足矩形区域大小 (4×4) 代入的公式： $4 + 4 + 1 = 9$ 。所以这样就有鱼结构的存在了。而且实际上，我们确实能找到鱼结构，如右图所示。

下面我们来证明一下这个内容。

4-4-1-2 证明

我们认为下方图上这样的结构就有鱼结构的存在了。

								4
5			3		2		7	
8			6		5		9	
						4		
9			2		6		1	

这是一则框架图，并不是一个唯一解的题目。不过可以从结构上发现，它确实存在鱼。那么我们用直观视角叙述的内容是什么呢？

这个图上有有一个关于数字 4 的三阶的标准鱼（也称三链列和剑鱼）。

我们需要记住如下的术语和公式。

术语：

- **域内**：直观标准鱼结构需要有一个“阵列”的构造出现，例如图上的 r247c1468 这样 12 个单元格。这么 12 格我们称为区域内，简称域内。
- **域内确定值**（简称**域内数**）：在域内的该数的确定值集合，比如图上的域内不包含任何该数字，所以图上并没有域内数。
- **架子**：区域内单元格横向或纵向涉及的所有区域，r2、r4、r7、c1、c4、c6、c8 这 7 个区域的所有单元格直接称为架子。
- **域外**：盘面除了架子外的剩余单元格。
- **域外确定值**（简称**域外数**）：在域外的该数确定值集合，比如图上的 r1c9(4)和 r5c7(4)统称为区域外提示数。
- **域外内净次数**（Net Extraterritorial and Intraterritorial Numbers，简称 **NAIN**）：该数值等于区域外提示数个数减去区域内该数总个数得到。比如该图里，区域外有 2 个 4，区域内则没有 4（0 个 4），则域外内净次数等于 $2 - 0 = 2$ 。

公式：

- 如果设架子一共由 n 个区域构成，且域内包含 k 个要找的鱼涉及的这个数，区域外提示数一共有 m 个，则当且仅当 **NAIN** 值等于“ $9 - \text{架子区域数}$ ”的时候，才有标准鱼的出现（或带鱼鳍的标准鱼、带链的鱼鳍标准鱼等等），且鱼的定义域产生于架子上，而删除域产生于域外。

我们先把公式用数学语言描述一下，然后以便后面的理解和对照。

$$\text{域外内净次数 (NAIN)} = 9 - \text{架子区域数 (n)}$$

我们为了证明简单一些，使用移项和拆解后的公式：

$$\text{域外数个数 (m)} - \text{域内数个数 (k)} + \text{架子区域数 (n)} = 9$$

由于 n 是架子区域总数，我们拆解为两个分量 a 和 b （分别表示横向架子区域数和纵向架子区域数），这样就可以表示和体现出架子是涉及 a 行 b 列的。那么一定有

$$\text{横向架子区域数 (a)} + \text{纵向架子区域数 (b)} = n$$

而

$$\text{域内单元格总数} = a * b$$

我们现在来数一下域外单元格总个数。因为架子区域是 a 行 b 列的，所以域外单元格总数应当是

$$\text{域外单元格总数} = (9 - a) * (9 - b)$$

而架子涉及的是 $a * b$ 个单元格，所以可以从式子上得到，架子外单元格总数就比架子本身要多出来 $(9 - a) * (9 - b) - a * b$ 个单元格。先把这个结论记下来，一会儿会用到，现在我们切换一下内容，来看结构。

								4
5			3		2		7	
8			6		5		9	
						4		
9			2		6		1	

如图所示，可以从图上看，所有的橙色单元格就是所有的域外单元格了。

现在，橙色单元格加上淡绿色单元格，刚好凑齐 c23579 五列，因为是五列，所以填数一定是数字 1 到 9 各有 5 个的（每一列都有 1 到 9 各一个，所以五列就各 5 个）。

再横着看。紫色单元格就是我们所有的域内单元格了。紫色单元格加上淡绿色的单元格，刚好凑齐三行，所以填数一定是 1 到 9 各有 3 个的，即：

$$\text{橙色单元格} + \text{淡绿色单元格} = 5 \text{ 套 } 1 \text{ 到 } 9$$

紫色单元格 + 淡绿色单元格 3套 1到9

我们针对于这两个式子作减法运算：

橙色单元格 - 紫色单元格 = 2套 1到9

再移项。

橙色单元格 = 2套 1到9 + 紫色单元格

这样就意味着，橙色包含所有紫色单元格的所有填数，还外带两套完整的 1 到 9 序列。

例如我们现在来看图上的数字 4。

								4
5			3		2		7	
8			6		5		9	
						4		
9			2		6		1	

域内（紫色）单元格是不存在数字 4 的（即此时 $k = 0$ ），而从式子里可以看到，域外（橙色）单元格具有两套 1 到 9 和紫色单元格的所有数字，而紫色里又没有 4，那么只能说明 2 套 1 到 9 里的 2 个 4 将是橙色单元格里所有的 4 了。换句话说，橙色就不应当再出现其余任何的 4。

刚才只是举例说明我想表达的意思，现在来推广到一般化的情况。域外单元格（橙色）的总数就比域内（紫色）单元格的总数本身要多出来 $(9 - a) * (9 - b) - a * b$ 个单元格，那么域外单元格的填数比域内的填数会多出来多少套完整的 1 到 9 呢？显然就是用这个数除以 9 即可，则有：

$$\begin{aligned}
 \frac{(9-a)(9-b)-ab}{9} &= \frac{81-9b-9a+ab-ab}{9} \\
 &= \frac{81-9(b+a)}{9} \\
 &= 9-(b+a) \\
 &= 9-n
 \end{aligned}$$

并最终我们得到了一个公式： $9 - n$ ，即橙色比紫色多出 $(9 - n)$ 套 1 到 9， n 还记得指代的是什么呢？ n 就是架子区域数了（因为 a 和 b 分别代指的是横向架子区域数和纵向架子区域数，所以加起来就是总的架子区域数）。这意味着什么呢？如果架子不包含我们寻

找的标准鱼结构所涉及的数字，而此时区域外单元格的部分（橙色）就包含 $(9 - n)$ 套 1 到 9，外加上架子内（紫色）的所有确定值。但如果紫色区域内不包含我们要找的数的话，那么数字就只可能出现在 $(9 - n)$ 套 1 到 9 的这一项里。当 $(9 - n)$ 恰好等于我们在区域外找到的这个数字的总个数时，区域外就不会再存在其余的这个数字了的填数位置了。

那么此时看下刚才“具象化”后的公式：

$$\text{橙色单元格} + \text{淡绿色单元格} = 5 \text{ 套 } 1 \text{ 到 } 9$$

既然橙色（域外）里没有这个寻找的数字的容身之地，那么这个数就只可能出现在淡绿色单元格里。如果域内里没有我们费尽千辛万苦寻找的数字，而必须又得需要 5 个这个数，现在域外是不能再填这个数了（域外有 2 个了），而为了保证数字不出现违背数独规则的现象，还剩下 3 个这个数，它们就只可能出现在不会直接和提示数冲突的地方（即淡绿色格子）。而又要保证行列上不要违背数独规则“重复”，那么数字就只能允许填入到它们应当放到的位置上。于是，标准鱼便形成了。

这一段如果不好理解，你可以对照图上来看。淡绿色的区域是 3×5 的区域，而提示数 4 的摆放位置一定是会和淡绿色具有同列的单元格的，毕竟这些 4 一定都在橙色单元格里，而橙色单元格又是和淡绿色单元格“正交的”，这就使得它就会连续排除两个在淡绿色填入 4 的位置。所以 3×5 的“5”就被压缩为了 $5 - 2 = 3$ 。这个式子的“2”，指的就是这个 4 在域外的提示数的个数了（两个 4）。那剩下来不就刚好 3×3 了吗？刚好我们从计算就得到了，必须要填 3 个，那么 3×3 的区域要填 3 个数，这不就已经满足了标准鱼的规则了吗？所以标准鱼一定会形成的原因我们就说清楚了。

所以说，只要出现最开始提到的公式，满足这个等式要求，那么就一定出现标准的鱼结构（当然，带鱼鳍的我们稍后会详细叙述其直观层面的推理逻辑）。

请注意，上面的证明使用的是 $k = 0$ 的情况，当 $k < 0$ 的时候，证明方式完全一样，结论也成立，此处就不赘述了，但不代表公式的 k 可以被忽略。

4-4-1-3 直观标准鱼视角的互补

在我们的推导和证明的过程之中，我们难免会遇到一些棘手但没有必要纠结的问题，比如鱼结构存在互补，但直观视角如何是思考互补的内容。实际上，这个答案很简单。我们来看一些示例。

7	6	4 5	1	2	5 8	9	4	4 5
2 3	2	1	5 8	5	9	4 5	6	4 5
5	8	9	4	6	7	2	1	5
4	1	6	9	4	4 5	5	4	2 3
9	2	7	2 3	4 5	4 5	6	4	1
4	3	4 5	6	1	4 5	7	4	9
1	7	2	5	4 5	6	1	9	4 5
6	4	5	7	9	1	8	2 3	2 3
1	9	5	2	3	4 5	7	6	8

如图所示，这个示例在我们之前已经出现过了。我们来看这个结构对应的两个互补的鱼结构。

7	6	4 5	1	2	5 8	9	4	4 5
2 3	2	1	5 8	5	9	4 5	6	4 5
5	8	9	4	6	7	2	1	5
4	1	6	9	4	4 5	5	4	2 3
9	2	7	2 3	4 5	4 5	6	4	1
4	3	4 5	6	1	4 5	7	4	9
1	7	2	5	4 5	6	1	9	4 5
6	4	5	7	9	1	8	2 3	2 3
1	9	5	2	3	4 5	7	6	8

如图所示，这个就是这个鱼结构对应的两个完全不同的候选数视角，不过一个是按列删数，一个则是按行删数。

我们再来看一则有关标准鳍鱼的示例。

1 2 7 8 1 2 3 7	1 2 7 8 1 2 3 7	2 7 2 3 6	5 2 3 6	1 2 9 7 2 3 1 2	4 7 1 2 3	1 2 7 1 2 3 7	1 2 7 1 2 3 7	2 7 2 3 6	5 2 3 6	1 2 9 7 2 3 1 2	4 7 1 2 3	1 2 7 1 2 3 7	3 7 1 2 3
5 7	4 7	2 3 6	2 3 6	1 2 9 7	8 7	9 7	1 2 9 7	2 3 6	2 3 6	1 2 9 7	8 7	9 7	1 2 9 7
3 6 8	5 8	3 6 8	9 8	7 8	2 8	1 3 6	1 3 6	4 8	3 6 8	9 8	7 8	2 8	1 3 6
9 7	1 7	6 7	4 7	3 7	5 7	2 7	9 7	2 3 6	2 3 6	4 7	8 7	5 7	1 7
7 6 7	9 7	5 7	1 7	3 7	2 7	4 7	8 7	7 6 7	9 7	5 7	1 7	3 7	2 7
1 2 3 6	1 2 3 6	2 3 6	2 3 6	1 3 6	1 3 6	7 9 7	7 9 7	5 6	1 2 3 6	1 2 3 6	2 3 6	2 3 6	1 3 6
1 2 3 6	1 2 3 6	2 3 6	2 3 6	1 3 6	1 3 6	7 9 7	7 9 7	5 6	1 2 3 6	1 2 3 6	2 3 6	2 3 6	1 3 6
1 2 3 6	1 2 3 6	2 3 6	2 3 6	1 3 6	1 3 6	7 9 7	7 9 7	5 6	1 2 3 6	1 2 3 6	2 3 6	2 3 6	1 3 6

如图所示,这也是之前的示例,我们可以发现,我们直接将整体的矩形对应的行换成列,就达到了互补的效果。当然,右图呈现的看起来是一个孪生鱼结构,似乎不完全是互补的。但实际上,我们针对于左图的结构,右图是它的互补的构型,这一点总是没有问题的。

所以鱼结构的直观视角里,互补仅仅是把矩形对应的行列区域(即架子)换一下而已。

4-4-1-4 标准鱼的观察

实际上,在前面已经提到了很多有关观察的内容了,不过没有说原理之前,理解起来是不那么容易的,这里我们再次阐述一下如何以直观视角观察鱼结构。

我们再次把提到的公式照搬过来(为了方便你看,所以我就复制一遍过来,就不用你往上翻了):

$$NAIN = 9 - \text{架子区域数}$$

为了凑公式,我们必须凑齐所有合适的数字才行。因为鱼是同数技巧,所以我们观察只需要从1到9挨个数一遍数字的位置即可。

1 7 4	6 9	8 7	5 7 4	6 9	7 9	2 7	3 7	4 7
5 4 7	4 9	4 9	3 4 7	6 9	2 4 7	1 5 7	7 6	8 9
2 4 7	2 3 4	2 3 4	8 1 4	1 4	1 4	5 6 7	6 5 7	9 9
8 1 4	1 2 4	1 2 4	6 1 4	5 4	7 4	9 4	3 4	3 4
2 7	2 7	6 7	5 7	9 7	2 3 7	3 7	4 7	1 7
3 1 7	4 9	1 9	1 4	1 4	8 7	6 5 7	5 2 7	2 7
9 4 7	8 4	4 7	2 4	5 4	6 4	3 4	1 5 7	5 6
2 4 7	1 2 3 4	1 2 3 4	1 4	1 3 4	1 3 4	8 4	2 5 6	5 6
2 4	1 2 3 4	1 2 3 4	7 4	8 4	1 3 4	9 4	2 5 6	5 6

注意，即使候选数单元格包含该候选数，也不算作信息点带入到公式中。因为公式本身并不包含候选数是否包含的这一点的。

现在我们找一个 3×4 的区域，如图所示：

1	6 7 9	8	5	6 7 9		2	3	4
5	4 6 9	4 6 9	3	4 6 9	2	1	7	8
2 4 7	2 3 4 7	2 3 4 7	8	1 4 7	1 4 7	5	6	9
8	1 2 4	1 2 4	6	1 2 4	5	7	9	3
2 7	2 6 7	5	9	2 3 7	3 7	4	8	1
3	1 4 7	1 4 7	1 4	1 4 7	8	6	5	2
9	8	4 7	2	4 5	6	3	1	5 7
2 4 7	1 2 3 4 5 6 7	1 2 3 4 5 6 7	1 4	1 3 4 5 9	1 3 4 9	8	2 4 7	5 6 7
2 4	1 2 3 4 5 6	1 2 3 4 6	7	8	1 3 4	9	2 4	5 6

529

1	6 7 9	8	5	6 7 9		2	3	4
5	4 6 9	6 4 9	3	4 6 9	2	1	7	8
2 4 7	2 3 4 7	2 3 4 7	8	1 4 7	1 4 7	5	6	9
8	1 2 4	1 2 4	6	1 2 4	5	7	9	3
2 7	2 7 6	6 2 7	5	9	2 3 7		3 4	8 1
3	1 4 7	1 4 9	1 4 7	1 4 7	8	6	5	2
9	8	4 7	2	4 5 7	6	3	1	5 7
2 4 7	1 2 3 4 5 6	1 2 3 4 6	1 4	1 3 4 5 9	1 3 4 9	8	2 4	5 6 7
2 4	1 2 3 4 5 6	1 2 3 4 6	7	8	1 3 4	9	2 4	5 6

这和我们的预期是完全一样的。

4-4-1-5 标准鳍鱼的观察

9	4	5		2	3	2	3	6	1	8		2	3
			7		7						7		
1	3			2				8	4	9		2	
			7								7		
8		2	6		6	5	1	2	3	9		3	4
			7								7		
7		2	5	6	3	9	2	5	6	8	4	1	
													6
	2	5	6	9	1	4		2	3	2	3	2	6
							7		7		7		3
	2	6	8	4		2	3	2	3	1	5	9	
							7		7				2
4	7	9	6	8	5	3	2	1					
	2	3	1	2	6	8	1	2	3	2	3	4	7
	2	3	1	2	5	6	2	1	2	3	9	2	3

先来看这个例子。如图所示。我们从直观层面找到了这么一个东西。首先，数字 7 在域外有两个，所以为了保证和为 9，我们选择找 3 * 4 的区域。

恰好，我们找到了该“矩阵”， $NAIN = 2 - 0 = 2$ （注意，架子内是有一个候选数 7 的。这里是不是 7 我们是不确定的）。

此时讨论它是否是 7。如果 $r9c6 = 7$ ，则直接通过它所在行列宫执行排除删数即可；如果 $r9c6 \neq 7$ ，由于所有 7 已经被我们找完了，而且域内和这些 7 不同行列，所以公式也满足，标准鱼成立。于是删数存在于域外，故此时只需要去找到删数的具体宫（b8）和删除域的交集，即为删数。

9	4	5	^{2 3}	^{2 3}	6	1	8	^{2 3}
1	3	²	8	4	9	²	6	5
8	²	²	5	1	^{2 3}	9	³	4
7	²	3	9	²	8	4	1	²
²	9	1	4	^{2 3}	^{2 3}	²	³	8
²	8	4	^{2 3}	^{2 3}	1	5	9	^{2 3}
4	7	9	6	8	5	3	2	1
^{2 3}	^{1 2}	8	^{1 2 3}	^{2 3}	4	⁶	5	9
^{2 3}	^{1 2}	²	^{1 2 3}	9	^{2 3}	8	4	⁶

而实际上，我们能找到关于 7 的鳍三链列，如图所示。

9	4	5	^{2 3}	^{2 3}	6	1	8	^{2 3}
1	3	²	8	4	9	²	6	5
8	²	²	5	1	^{2 3}	9	³	4
7	²	3	9	²	8	4	1	²
²	9	1	4	^{2 3}	^{2 3}	²	³	8
²	8	4	^{2 3}	^{2 3}	1	5	9	^{2 3}
4	7	9	6	8	5	3	2	1
^{2 3}	^{1 2}	8	^{1 2 3}	^{2 3}	4	⁶	5	9
^{2 3}	^{1 2}	²	^{1 2 3}	9	^{2 3}	8	4	⁶

那么，鳍鱼的直观有什么特殊结论吗？其实是有的。试想一点，我们在直观层面是没有标注候选数的，而此时我们标注的架子是不区分行列“方向”的，它不像鱼会区分。所以，此时我们有一个结论是这样的：

我们使用这个图给大家解释下。由于 7 的位置不能确定行还是列上存在标准鱼，我们的结论是：以这个不知道 7 到底是否成立的单元格（即这里的 **r9c6**）为“拐点”，所在宫内同行列的一共 5 个单元格（即 **r78c6**、**r9c45** 和 **r9c6** 五个单元格）里必有一个单元格是数字 7。

当然，这个题上 **r78c6** 是被确定值占据了，因而没有 7。我刚才的结论是推广到一般情况下的。而且，实际上这 5 个单元格叫做“关于 **r9c6** 的空矩形区域”，简称“空矩形”。

至于为什么会这样，说起来也很简单：因为直观层面不区分鱼的“方向”，所以既然删

数成立，那么结构必然产生于架子里，而架子里一定会有鱼身的存在，所以不能保证具体在哪里，只能保证**拐角的这个十字形状下的 5 个单元格（空矩形区域）里的这个数字必须有一个是为真的**，这就是当观察直观版本的鲑鱼时，它所拥有的结论。

4-4-1-6 直观鱼的示例

下面阐述一些稍微难一些的直观鱼的示例。这些直观鱼的示例在之前都是没有说到的，因为在之前的内容里，我们并没有把技巧升华到这个难度。

以下的例子均和上面的逻辑完全一样，所以这里不再细述其逻辑。如果需要介绍，下面还是会有注释文字的。

示例 1：标准鱼

5 8 9	5 8	2 8 9	1 2 8 6	7 8	1 3 6 8	1 8	1 3 5 9	4 9	5 8 9	5 8	2 8 9	1 2 8 6	7 8	1 3 6 8	1 8	1 3 5 9	4 9
6 4 5 8	4 5 8	2 8 9	1 2 8	2 3 8	1 3 8	3 8	1 3 7 9	1 3 7	6 4 5 8	4 5 8	2 8 9	1 2 8	2 3 8	1 3 8	1 3 7 9	1 3 7	4 9
7 4 6 8	1 4 6 8	3 4 6 8	4 8 9	5 8	9 8	2 6 9	8 7 9	6 7	7 4 6 8	1 4 6 8	3 4 6 8	4 8 9	5 8	9 8	2 6 9	8 7 9	6 7
2 4 5 6 4	3 4 5 6 4	1 4 5 6 4	6 5 9	7 5 9	5 7 9	1 7 9	1 3 7 9	8 7 9	2 4 5 6 4	3 4 5 6 4	1 4 5 6 4	6 5 9	7 5 9	5 7 9	1 7 9	1 3 7 9	8 7 9
5 8	9 8	1 8	6 7 8	3 7 8	1 7 8	6 7 8	5 7 8	4 7	1 8	2 8	5 8	7 8	4 7	3 7 8	1 7 8	2 8	9 7
1 8 9	2 7 8	5 6 8 9	7 8 9	8 8 9	4 6	3 6	6 7 8	9 7 8	1 8	2 7 8	5 6 8 9	7 8 9	8 8 9	4 6	3 6	6 7 8	9 7 8
4 7 8	3 7 8	6 8	1 5 6	9 5 6	1 5 6	3 5 6	1 7 8	1 2 7	4 7 8	3 7 8	6 8	1 5 6	9 5 6	1 5 6	3 5 6	1 7 8	1 2 7

示例 2：孪生鱼

1 2 7 8	1 2 7 8	2 7 6	5 9 7 9	4 7 6	3 7 6	1 2 6 3	5 9 7 9	4 7 6	3 7 6	1 2 6 3	5 9 7 9	4 7 6	3 7 6	1 2 6 3	5 9 7 9	4 7 6	3 7 6
1 2 3 7	1 2 3 7	9 7	2 3 7	6 7	4 7	5 7	8 7	1 7	1 2 3 7	1 2 3 7	9 7	2 3 7	6 7	4 7	5 7	8 7	1 7
5 7	4 7	2 3 6	2 3 6	1 2 8	9 7 6	1 2 6	8 7 6	1 2 6	5 7	4 7	2 3 6	2 3 6	1 2 8	9 7 6	1 2 6	8 7 6	1 2 6
3 8	5 6	3 6	9 7	7 8	2 8	1 3 6	1 3 6	4 7	3 8	5 6	3 6	9 7	7 8	2 8	1 3 6	1 3 6	4 7
9 7 8	1 7 8	6 7 8	4 7 8	3 7 8	5 7 8	2 7 8	9 7 8	1 7 8	9 7 8	1 7 8	6 7 8	4 7 8	3 7 8	5 7 8	2 7 8	9 7 8	1 7 8
2 3 7	2 3 7	6 7	4 7	8 7	5 7	1 7	3 7	9 7	2 3 7	2 3 7	6 7	4 7	8 7	5 7	1 7	3 7	9 7
6 7	9 7	5 7	1 7	3 7	2 7	4 7	8 7	5 7	6 7	9 7	5 7	1 7	3 7	2 7	4 7	8 7	5 7
4 7	1 2 3 7	2 3 6	2 7	8 7	6 7 9	1 3 7	1 3 7	5 7	4 7	1 2 3 7	2 3 6	2 7	8 7	6 7 9	1 3 7	1 3 7	5 7
1 2 3 7	1 2 3 7	6 7	8 7	4 7	5 7	1 3 7	1 3 7	6 7 9	1 2 3 7	1 2 3 7	6 7	8 7	4 7	5 7	1 3 7	1 3 7	6 7 9

1 2 7 8	1 2 7 8	2 7	5	1 2 9 7	4	1 2 7 6	3
1 2 3 7	1 2 3 7	9	2 3 7	6 4 7	5 8 7	1	
5	4	2 3 7	2 3 7	1 2 7	8 9 7	1 2 7	6
3 6 8	5	3 6 7	9	7 2 8	1 3 7	1 3 7	4
9	7 8 7	1	6 4 7	3 7	5 2 7		
2 3 7	2 3 7	4	8 5 7	1 7	3 6 7	3 6 7	9
7 6 7	9 5 7	1 3 7	3 6 7	2 4 7	8 7		
4	1 2 3 7	2 3 7	2 7	8 7	1 3 7	1 3 7	5
1 2 3 7	1 2 3 7	8 4 7	2 9	5 7	1 3 7	1 3 7	6

1 2 7 8	1 2 7 8	2 7	5	1 2 9 7	4	1 2 7 6	3
1 2 3 7	1 2 3 7	9	2 3 7	6 4 7	5 8 7	1	
5	4	2 3 7	2 3 7	1 2 7	8 9 7	1 2 7	6
3 6 8	5	3 6 7	9	7 2 8	1 3 7	1 3 7	4
9	7 8 7	1	6 4 7	3 7	5 2 7		
2 3 7	2 3 7	4	8 5 7	1 7	3 6 7	3 6 7	9
7 6 7	9 5 7	1 3 7	3 6 7	2 4 7	8 7		
4	1 2 3 7	2 3 7	2 7	8 7	1 3 7	1 3 7	5
1 2 3 7	1 2 3 7	8 4 7	2 9	5 7	1 3 7	1 3 7	6

横着看，前面两个图是鲮鱼的第一种看法；后面两个图是鲮鱼的第二种看法，因为它们共用大部分域内单元格，而且从题上可以看出，它们使用了相同的鱼身，所以是孪生的鱼结构。

示例 3：鱼和环的交织

6	1 2 3 9	1 2 3 9	7	5 8 7	2 5 8 7	4	2 3 8	5 8 9 7
2 3 5	4	2 3 9	5 3 8	1	2 5 8 7	2 3 8	6	5 8 9 7
2 3 5	7 8	4 6 4 6	9	2 3 7	1 5 8			
7	1 2 3 5 8	1 2 3 4	9	4 5 8 7	6	2 3 8	1 3 7	
2	1 2 8 9	1 2 8 9	4 6 7	7 3 8	1 2 8 9	5 4 7		
3 1 8 9	3 1 5 8	1 3 4 6	2	4 5 6 8	1 3 4 5 6	3 1 8	7	
4	2 3 8 9	2 3 7 9	1 3 8 9	3 1 6 8	5	3 1 6 8	1 3 7	
1	3 8 9	5	4 6 4 6	7	3 8	4 6 7	2	
3 8	6	3 1 4 5 8	2	1 4 5 8	1 3 7 8	9	1 3 4 8	

6	1 2 3 9	1 2 3 9	7	5 8 7	2 5 8 7	4	2 3 8	5 8 9 7
2 3 5	4	2 3 9	5 3 8	1	2 5 8 7	2 3 8	6	5 8 9 7
2 3 5	7 8	4 6 4 6	9	2 3 7	1 5 8			
7	1 2 3 5 8	1 2 3 4	9	4 5 8 7	6	2 3 8	1 3 7	
2	1 2 8 9	1 2 8 9	4 6 7	7 3 8	1 2 8 9	5 4 7		
3 1 8 9	3 1 5 8	1 3 4 6	2	4 5 6 8	1 3 4 5 6	3 1 8	7	
4	2 3 8 9	2 3 7 9	1 3 8 9	3 1 6 8	5	3 1 6 8	1 3 7	
1	3 8 9	5	4 6 4 6	7	3 8	4 6 7	2	
3 8	6	3 1 4 5 8	2	1 4 5 8	1 3 7 8	9	1 3 4 8	

6	1 2 3	1 2 3	7	3	2	4	2 3	3
2 3		9	5	8	5	8	8	5
5	4	2 3	3	1	2	2 3	2 3	6
2 3	7	8	4 6	4 6	9	7 8 9	7 8	3
5						2 3	1	5
7	1 2 3	1 2 3	1	4 5	1	4 5	2 3	1 3
5		4	8	9	6	8	4	4
8	2	1 2	1	7	3	1 2	5	1
8 9	8 9	4 6	4 6	8	8 9	8 9	8 9	8 9
3	1	3	2	4 5 6	1	3	3	7
8 9	8 9	4 6	8	8	8 9	4 6	8	
4	2 3	2 3	1	3	1	5	3	1 3
	8 9	7 9	8 9	6	8	7 8	8	
1	3	5	4 6	4 6	7	3	4 6	2
	8 9	8 9	8 9	8	8	8	8	
3	6	3	1	3	1	3	9	1 3
8		7	4 5	8	4 5	7 8	4	8
			2	1	1	3		
			8	8	7 8			

这个需要解释下为什么。看第一个图，如果两个 4 同假，则域内不存在 4 的任何提示信息，算上域外的提示数 4， $NAIN = 3 - 0 = 3$ ，且此时恰好满足等式，即产生关于 4 的鳍鱼结构（这里不能说是鱼，因为 r8c8 的 4 是不知道是否存在的）。

刚才鳍鱼最后得到一个结论：十字形状的五個单元格里必须有一个 4 为真。但是观察这样的 5 个单元格（r789c8 和 r8c79），能填入 4 的位置只剩下十字架的中间这一个单元格 r8c8 了，所以当 r5c3 <> 4 的时候，必须得有 r8c8 = 4 的结论。因而强关系成立。

而右图的 6 的强关系则是另外一种情况。因为域外有 4 个 6 的确定值了，所以此时看这个结构已经不成立了。不过……此时因为 r8c8 = 4 的关系，r8c8 <> 6 也是事实。如果此时 r5c3 = 6 的话，会怎么样呢？

从原理上推进逻辑，如果域内有这个数，那么此时 $k = 1$ ，根据公式要得到 9，域外就必须有 4 个。确实盘面“履行了诺言”，出现了 4 个确定值 6。那么此时鱼是成立的，因而 r5c3=6 是应当成立的；否则如果 r5c3 没有 6，从原理上推进推理过程，6 将放不满架子该放的总个数，于是便产生了矛盾（此处就自己推一下，我就不再作图说明了）。

总之，链写法如下：

$$r5c3=r8c8(4-6)=r5c3(6)-r5c3(4) \Rightarrow r5c3 <> 129$$

这个链构成了环，并且成立。

此时留下一个问题：这环除了能删除基本的删数外，还有一些删数客观存在，你知道在哪里吗？

示例 4：鱼和直推 UR

1 2	1 2	1 2		7	1	2	2	1
8 9	9	6	6		4	5	5 6	4 5 6
	1 2	1 2	1 2	1 2	8	8 9	8	8
4	7		6	3	2	2	5 6	9
1 2 3	1 2 3			1 2	5	6	7 8	
8	7 9	5		1	4	4	2	1
1 2	1 2			9	8		7 8	8
1 2	4	6	8	3	1 2	7	5	6 4 6
9	9						9	
5	2 3	2 3		6	2		1	7 8
	9	9			8	4	7 8 9	
7	1	1	6	1	5	9	4 6	3 2
4	6					5 8	8	
1 2 3	1 2 3	7	4	6	2 3	2	2	3
	9				9	8 9	8 9	5 8
2 3	5	4	8	7	2 3	1	2	3
	9				9		6 9	6
6	8	2 3	2	5	1	2	4	3
	9		9			7 9		7

如图所示，因为推导逻辑顺序不好标注到图上，所以没有箭头。现在我们用文本进行叙述。

我们假设 $r1c3 = 1$ ，此时应当有 $r26c3 \neq 1$ ，且 $r6c3 = 6$ 的结论。针对于 $r26c3 \neq 1$ ，它是架子上的单元格，如果两处都没有 1，所以架子没有 1，此时公式里的 $NAIN = 3 - 0 = 3$ ，于是恰好满足公式，所以删除域的 1 应当都可以被去掉，所以图中标注青色的数字 1 都是此时可以删除的数字。

接着，当这些 1 去掉后，结合刚才 $r6c3=6$ 的结论，我们可以确定，b4 里只有一处可以填 1，即 $r6c1 = 1$ 。然后根据直推 UR 的逻辑， $r12c3$ 由于 $r6c3 = 6$ 的关系，这两个单元格也都不能填 6。所以 b1 里填 6 的位置只有一处可填： $r1c2 = 6$ 。

此时，结合刚才的所有步骤，我们得到 $r1c2 = 6$ 、 $r1c3 = 1$ 、 $r6c2 = 1$ 、 $r6c3 = 6$ ，此时形成 UR 的致命形式，所以矛盾。所以最初的假设（ $r1c3 = 1$ ）为假，即 $r1c3 \neq 1$ ，为该题结论。

示例 5：另外一个环

1 5 7 9	2	5 7 9	8	1 5 7 9	1 3 5 7 9	3 5 7 9	4	6
4 5 7 9	3 7 9	4 5 7 9	3 6	2 5 6	3 7	1 5 6	8	5 7 9
6	1 3 8 9	5 7 8 9	1 3	5 9	4	2 3 5 9	2 3 5	2 5 7 9
2 7	5	2 3 7	1 2 3 6	1 3 6	9	4	1 2 6	1 2 8
4 8 9	6 8 9	1	4 6	7	2 6	3 5 6	2 5 6	3
2 4	3 6 4	2 3 9	5	1 4	3 8	1 2 3 6	2 6	7 9
3	7	6	9	1 5	1 2 5	8	1 2 5	4
1 2 5	4	2 5	7	3 6	8	3 6	9	1 2 5
2 5 8 9	1 8 9	2 8 9	2 3 4	1 3 4 5 6	2 3 5 6	7	3 6	2 5

如图所示，链表示如下：

$r9c5 = \{r46c5, r4c4\} - r5c46(6) = r5c4 - r9c4 = r9c5(4-6)$
 $\Rightarrow r9c6 \langle \rangle 135, r4c4, r5c17 \langle \rangle 2, r6c6 \langle \rangle 26$

首先我们来简单说一下。假设 $r9c5 \langle \rangle 6$ 了，则鳍鱼立马就形成了，还记得我刚才的逻辑吗？十字形的 5 个单元格里必须得有一个 6，所以此时 $r46c5$ 和 $r4c4$ 里必须出一个 6。所以， $r9c5(6) = \{r46c5, r4c4\}(6)$ 强关系成立。接着，由于三格有一个 6，所以 $r5c46$ 不能填 6。注意到 ALS 区域 $r5c46$ ，所以 $r5c46(6) = r5c4(4)$ 成立。后面就是基本逻辑了。

形成环后，删数逻辑就不用我一个一个说了。此时请注意 ALS 的删数比较特殊，2 也可以删掉。

4-4-1-7 总结

本文介绍了直观鱼的基本公式，以及公式的证明。公式如下：

$$NAIN = 9 - \text{架子区域数}$$

当满足该公式的时候，题目必然存在标准鱼结构。而这样的观察视角对于出题层面和解题层面都具有很大的帮助。

对于鳍鱼，还存在一个结论：**如果架子存在没有确定的空单元格，且该单元格的候选数包含我们需要寻找的标准鱼所涉及的数字，则鳍鱼成立（前提是必须要满足公式再来谈这一点哈），以这个单元格为“拐角”，向该宫以行列方向延伸的十字形状的 5 个单元格里，必须有一个格填入的是这个数。**

4-4-1-8 宫内鱼的直观

下面我们拿出简短的几句话内容来阐述一下，宫内鱼和交叉鱼的直观。

在我们之前直观的视角里，我们只要能够找到一个矩形区域的结构，便可得到存在鱼结

2 5 8	2 4 5 8	6	7	4 8	4 5	3	9	1	2 5 8	2 4 5 8	6	7	4 8	4 5	3	9	1
1 5 7	1 4 5 7	9	3	1 4 8	4 5	5 7 8	6	2	1 5 7	1 4 5 7	9	3	1 4 8	4 5	5 7 8	6	2
3	1 2 5 7	2	1	5 6 8	6 9	5 7 8	4		3	1 2 5 7	2	1	5 6 8	6 9	5 7 8	4	
1 2 5 6 8	1 2 5 6 8	2		5 6 8	3	5 6 7 9	2	6 5 8	4	1 2 5 6 8	2		5 6 8	3	5 6 7 9	2	6 5 8
5 6 8	5 6 8	7	2	4 6 8	4 5 6 9	6 8 9	1	3	5 6 8	5 6 8	7	2	4 6 8	4 5 6 9	6 8 9	1	3
4	2 5 8	3		5 6 8	7 9	6 8 9	1		4	2 5 8	3		5 6 8	7 9	6 8 9	1	
2 9	3	1	4 6	2 6	8	4 9	7	5	2 9	3	1	4 6	2 6	8	4 9	7	5
2 7 8	2 7 8	4 9	5	3	1	2 8	6		2 7 8	2 7 8	4 9	5	3	1	2 8	6	
6	5						3		6	5						3	

可以看到，我们套用之前的方式完全是一样的操作。但实际上，我们从经验上就可以看出，这种操作也只能针对于只有一个宫的宫内鱼结构才可以奏效，而且它借助的是和这个唯一的宫元素旁边无关的候选数引出的毛刺。

而实际上，两个示例都可以看到，借助毛刺并没有什么作用。因为借助的毛刺已经完成了大部分鱼结构的观察。如果你能找到两个示例的毛刺为真的情况（即左图这样），显然宫内鱼结构的构型就差不太远了。所以，宫内鱼的直观视角完全就是“费力不讨好”；当然了，在一些极端情况下，它们还是有用处的，虽然用处不太大。

4-4-1-9 交叉鱼的直观

下面我们来看交叉鱼结构的直观视角。我们按照老规矩，先看交叉三链列。

	3	5	1	6	8	2 5		3	4	2 3
7	9	7			1 2 3	5	9	7		9
4	3	8	4 5		1 2 3	5	3	2		6
7	9		7		9	4	9	7	9	
4	3	2	6		3	7	4	8	5	1
	9				9					
6	4	8	1 2	3	5	3	5	1 2	2	
1 2 3	1	3	9	7	6	8	4	1 2	9	7
1 2	1		5	1 2	5	4	1 2	6	3	8
7	7	7	7	9	9	9	9			
5	9	4	2 3	8	1 2	7	1 3	6	4	3
8	1	3	2 3	1 2	4	6	1 3	7	4	3
1 4		6	4	1	5	9	3	2	8	4 5

如图所示。如果 $r2c5(5)$ 为假，则由于 2×5 的矩形内不含候选数 5，而也满足直观鳍鱼结构的要求，所以删数应当为域外的候选数 5。所以这个示例想要告诉我们的就是这一点。它依然引出了强制链的思维。我们再来看一下交叉四链列的例子。

4	1 2	1 3	2 3	2	3	2 3	8	5
9		5	8		4	1	6	7
6		8	5		1	9		4
	9		7	4			1 3	1 2 3
	8	7	9	1	6	5	4	2 3
	4						7	9
		4			7			8
8		9	4		2	7		
7	5					4		6

可以从图中看出，当引出强制链的时候，最终得到了一个二链列结构，并得到了删数。而实际上你也可以看到，它已经包含了大部分原本交叉四链列结构的候选数了，这种观察视角并没有起到实质性的作用。

4-4-2 飞鱼导弹和网

实际上，飞鱼导弹和网是有着密不可分的关系的，不过在飞鱼导弹里我们只字未提，原因很简单：因为讲解飞鱼导弹的时候还没学到网。那么现在来看看，飞鱼导弹到底和网有着什么关系。

首先我们拿出一则最标准飞鱼导弹的示例给大家看。

3 1 3 1 3	3 3	2	1 3 1 3 1	3 3 3	2	1 3 1 3 1	3 3 3	2	1 3 1 3 1
4 5 6 4 5 6 4	4 5 7 8		4 5 4 5 9	4 5 6 4 5 6 4	4 5 7 8	4 5 4 5 9	4 5 6 4 5 6 4	4 5 7 8	4 5 4 5 9
2 3 1 3	9	6	1 2 3 8	2 3 1 3	9	6	1 2 3 8	2 3 1 3	9
4 5 4 5			4 5 7	4 5 4 5			4 5 7	4 5 4 5	
2 3	7	1	9	2 3	7	1	9	2 3	7
4 5 8				4 5 8				4 5 8	
3 4 5 3	6	9	8	3 4 5 3	6	9	8	3 4 5 3	6
4 5 7				4 5 7				4 5 7	
4 5 2				4 5 2				4 5 2	
7 9				7 9				7 9	
1 4 5 3	8	7		1 4 5 3	8	7		1 4 5 3	8
9				9				9	
2 3 1 3 1 2 3	2 3 1 2 3 1 3	1 2 3 4	1 2 3 5	2 3 1 3 1 2 3	2 3 1 2 3 1 3	1 2 3 4	1 2 3 5	2 3 1 3 1 2 3	2 3 1 2 3 1 3
7 8 9 8 9 7	5 6 8 9 7 8	7 8	7 8	7 8 9 8 9 7	5 6 8 9 7 8	7 8	7 8	7 8 9 8 9 7	5 6 8 9 7 8
2 3 1 3 1 2 3	2 3 1 2 3 1 3	1 2 3 4	1 2 3 5	2 3 1 3 1 2 3	2 3 1 2 3 1 3	1 2 3 4	1 2 3 5	2 3 1 3 1 2 3	2 3 1 2 3 1 3
4 6 4 6 4	6 8 9 7 8	6	6	4 6 4 6 4	6 8 9 7 8	6	6	4 6 4 6 4	6 8 9 7 8
7 8 9 8 9 7	8 9 7 8 7 8	7 8	7 8	7 8 9 8 9 7	8 9 7 8 7 8	7 8	7 8	7 8 9 8 9 7	8 9 7 8 7 8
2 3 1 3	5	4	9	2 3 1 3	5	4	9	2 3 1 3	5
6				6				6	

这个示例实际上你在前文可以找到它。这则示例的大致逻辑是，如果我们假设右图之中所有红色的删数里，任意一个是为真的，那么都会使得结构里的任意一个链接点消失，使得结构变为0网（秩为0的网），此时的所有网内的可能删数就全部成立了；但此时如果让这些删数全部成立的话，必然会使得r89里所有1、2、7全部挤入到r89c359里，结构变为1、2、7的拓展矩形的致命形式，所以出现矛盾。

从经验上来看，我们一般都会将 JE 原本涉及的数字作为一种链接点类型，而剩余的数
字，则设为另外一种类型的链接点，并且一般而言，计算 JE 里涉及的数字的出现次数时，
如果按照行来看，那么这些数字在转为网的时候，也都将使用行链接点视角；同样地，如果
是列的话，就转为列链接点的视角。

实际上很多时候，JE 都可以通过这个方式转为网的视角，而具体为什么可以这样转换，
而且删数是否和 JE 本身可以删除的数字（含可推论带来的删数）一样呢？很抱歉的是，JE
的结构过于庞大，使得结构的变化形式非常多，所以不能轻易给出删数原理和结论是否一致。
比如，上图给出的是一个+1 网，而稍后我们就会介绍一个转换为网视角后变为+2 网的例子，
它带有 16 个单元格，却是 18 个链接点。

不过，这个示例的 JE 视角里（左图），我并没有标注出潜在的推论删数，那么到底最终
右图给出的这些删数是否和 JE 的所有删数是完全一样的呢？实际上并不是。网的视角的删
数比 JE 的删数要多一些，你可以尝试推导一下。

2 4 5 8	2 4 6 7	2 4 5 6 7 8	1 2 6 7 8 9	1 6 7 8 9	2 4 5 8 9	1 2 4 8	3
2 3 4	2 3 4	1 8	2 3 7 8 9	5 7 8 9	6 4	2 4	2
2 3 5	9 5 6 8	2 5 6 8	1 2 3 6	4 6	1 3 5	2 7	1 2 5
1 2 3 4	1 2 3 4	1 3 6	1 3 7 8	9 7 8	2 3 4	5 7	1 2 4 6
7 4	1 2 3 4	2 6	1 3 6	5 6	2 3 4	1 2 3 6	8
1 3 8 9	5 8 9	6 8 9	4 7 8	1 3 7 8	2 7 9	1 3 7	1 6
1 4 5	8 4	2 7	1 3 7	2 4 6	4 5 7	9 4 5 6	
2 4	2 4	3 9 7	5 7 8 9	1 7 8 9	4 6	2 4 6	2
6 4	1 2 4	2 4 5	1 3 7 8 9	1 3 7 8	2 3 4 5	2 3 8	2

2 4 5 8	2 4 6 7	2 4 5 6 7 8	1 2 6 7 8 9	1 6 7 8 9	2 4 5 8 9	1 2 4 8	3
2 3 4	2 3 4	1 8	2 3 7 8 9	5 7 8 9	6 4	2 4	2
2 3 5	9 5 6 8	2 5 6 8	1 2 3 6	4 6	1 3 5	2 7	1 2 5
1 2 3 4	1 2 3 4	1 3 6	1 3 7 8	9 7 8	2 3 4	5 7	1 2 4 6
7 4	1 2 3 4	2 6	1 3 6	5 6	2 3 4	1 2 3 6	8
1 3 8 9	5 8 9	6 8 9	4 7 8	1 3 7 8	2 7 9	1 3 7	1 6
1 4 5	8 4	2 7	1 3 7	2 4 6	4 5 7	9 4 5 6	
2 4	2 4	3 9 7	5 7 8 9	1 7 8 9	4 6	2 4 6	2
6 4	1 2 4	2 4 5	1 3 7 8 9	1 3 7 8	2 3 4 5	2 3 8	2

如图所示，这个示例带有 16 个单元格，但拥有 18 个链接点。显然此时我们就不能尝
试使用“预备删数”的逻辑来推导了。不过，我们可以试着改变一下思路，或许你可以发现
这个技巧实际上在转换为网的时候，依然可以删数。不过，我希望你能独立思考它，虽然我
也知道这一点思考和得到结果的过程很艰辛，也富有挑战性。

4-4-3 SK 环和网

下面我们来看一个更奇特的技巧：**SK 环**。这个技巧可以被看作一个网。

4-4-3-1 如何转换呢？

1	2	6	4	2	2 3	2 3	9	5	6	3
	8		7	4 5 6	4 5 6	4 5		8		
2	3			1 2	2	7	5	6	1	6
	6		6	5	5 6		5	6	8	
8			8 9	9	8 9		8			
4		6	5	1	3	1 3		1	6	2
7		8 9		4	6 4	6 4		8		
				9	8 9	9 7				
2 3	1 2	1 3	2	2	2 3	2 3	4 5	7	1 3	
5	5		8 9	4 5	4 5	6	8		8 9	
8	8 9	8 9		9	9					
2 3	2	3	2	2	2	2 3	2 3	2	3	
5 6	5 6	6	4 5	4 5	1	4 5	4 5 6	5 6	4 6	
7 8	8 9	7 8 9	7 9	9			8	8 9	8 9	
2	1	6	2	5	8	5 6	5 6	5 6	6	
5 6	4	7 9	3	7 9					9	
7										
9	1	1 3	1 2	2 3	1 2 3		2	2	5	
	8	4 6	4 6	4 6	4	4 6	7 8	8		
		8	7	7						
4 5 6	7	1	2	2	1 2	4 5 6	4 6	3	4 6	
		4	5 6	4 5	9					
				9	9					
4	5 6	5 6	2	4 5 6	4 5 6	4 5	1	6	4 6	
8	8	8		7 9	7 9	9		8 9	7 8 9	

如图所示，这便是很好的说明。可以看到，我们把行、列、宫链接点分别使用绿色、橙色和紫色标注，可以发现，恰好可以实现全覆盖，并满足 0 网的规则：链接点总数和单元格总数一样。所以，这个示例的删数便是所有链接点对应区域的这些数字了。

不过，我们介绍 SK 环结构并没有如此简单。

4-4-3-2 SK 环的直观

事实上，由于结构过于特殊，我们甚至可以通过直观层面去分析 SK 环结构。

首先，我们观察候选数视角的 SK 环，可以看到候选数视角的 SK 环里涉及的行列链接点比较整齐，而宫链接点则显得不是特别引人注目。不过我们依然可以发现到，行列链接点涉及的数字不总是一样的，但大致的分布是两个数字一组，且以宫链接点（紫色）作为行链接点和列链接点的桥梁。而且，宫链接点在对角分布的两个宫里，数字是一样的。比如 b19 里的宫链接点都是 2 和 9，而 b37 里则都是 1 和 5，这是必然吗？

1	2 6 8	4 7	2 4 5 6	2 4 5 6	2 4 5 6	9	5 6 8	3
2 6 8	3	6 8 9	1 2 5 6 9	2 5 6 8 9	7	5 6 8	4	1 6 8
4 7	6 8 9	5	1 4 6 9	3 4 6 8 9	1 3 4 7 9	3 1 6 8	2	
2 3 5 8	1 2 5 8 9	1 3 8 9	2 4 5 9	2 4 5 9	6	2 3 4 5 8	7	1 3 4 8 9
2 3 5 6 7 8	2 5 6 8 9	3 6 7 8 9	2 4 5 7 9	1	2 4 5 9	2 3 4 5 6 8	2 5 6 8 9	3 4 6 8 9
2 5 6 7	4	1 6 7 9	3	2 5 7 9	8	2 5 6	1 2 5 6 9	1 6 9
9	1 6 8	1 3 6 8	1 2 4 6 7	2 3 4 6 7	1 2 3 4	2 4 6 7 8	2 6 8	5
4 5 6	7	1 4 6	8	2 4 5 6 9	1 2 4 5 9	2 4 6	3	4 6 9
4 5 6 8	3 5 6 8	2	4 5 6 7 9	3 4 5 6 7 9	1	6 8 9	4 6 7 8 9	

我们反过来，看看结构的直观模式。为了不影响你继续看候选视角，这一次我不打算把候选数全部都隐去。可以看到的是，我们以 **r28c28** 作为“轴心”，**b1379** 四个宫的剩余提示数全部都作为结构的“支柱”。如果我们只要这些数字，那么 SK 环的框架就已经形成了。

1	2 4 6 8	4 6 7 8	2 3 4 5 6 7 8	2 3 4 5 6 7 8	2 3 4 5 6 7 8	9	5 6 7 8	3 6 7 8
2 6 7 8	3	6 7 8 9	1 2 5 6 7 8 9	1 2 5 6 7 8 9	1 2 5 6 7 8 9	5 6 7 8	4	1 6 7 8
4 6 7 8	4 6 8 9	5	1 3 4 6 7 8 9	1 3 4 6 7 8 9	1 3 4 6 7 8 9	3 1 6 7 8	2	
2 3 4 5 6 7 8	1 2 4 5 6 7 8 9	1 3 4 6 7 8 9	1 2 3 4 5 6 7 8 9	1 2 3 4 5 6 7 8 9	1 2 3 4 5 6 7 8 9	2 3 4 5 6 7 8	1 2 5 6 7 8 9	1 3 4 6 7 8 9
2 3 4 5 6 7 8	1 2 4 5 6 7 8 9	1 3 4 6 7 8 9	1 2 3 4 5 6 7 8 9	1 2 3 4 5 6 7 8 9	1 2 3 4 5 6 7 8 9	2 3 4 5 6 7 8	1 2 5 6 7 8 9	1 3 4 6 7 8 9
2 3 4 5 6 7 8	1 2 4 5 6 7 8 9	1 3 4 6 7 8 9	1 2 3 4 5 6 7 8 9	1 2 3 4 5 6 7 8 9	1 2 3 4 5 6 7 8 9	2 3 4 5 6 7 8	1 2 5 6 7 8 9	1 3 4 6 7 8 9
9	1 4 6 8	1 3 6 8	1 2 3 4 6 7 8	1 2 3 4 6 7 8	1 2 3 4 6 7 8	2 4 6 7 8	2 6 7 8	5
4 5 6 8	7	1 4 6 8	4 5 6 8 9	4 5 6 8 9	4 5 6 8 9	4 6 8	3	4 6 8 9
4 5 6 8	3 4 5 6 8	2	4 5 6 7 8 9	3 4 5 6 7 8 9	3 4 5 6 7 8 9	1	6 8 9	4 6 7 8 9

此时我们仅尝试观察 **b1379** 里跟 SK 环有关的单元格，你就会发现，结构的链接点完全没有发生变化，不过因为结构的干扰几乎不存在了（因为是图示，所以我们把所有其它位置的确定值都拿掉了），所以可能链接点会看似变多，但依旧是满足 0 网的要求的。

1		2				2 3	2 3	2 3	9		5 6		3	6
	4		6	4	6	4 5 6	4 5 6	4 5 6		7 8			7 8	
2			8			7 8	7 8	7 8						
	2					1 2	1 2	1 2					1	
	6	3			6	5 6	5 6	5 6		5 6			6	
7 8					7 8 9	7 8 9	7 8 9	7 8 9	7 8				7 8	
4	6		4		6	1 3	1 3	1 3		3				
7 8			8 9			4 6	4 6	4 6		6			2	
			8	9		7 8 9	7 8 9	7 8 9	7 8		7 8	6		
2 3	1 2			1	3	1 2 3	1 2 3	1 2 3	2 3	1 2			1	3
4 5 6	4 5 6			4	6	4 5 6	4 5 6	4 5 6	4 5 6	5 6			4	6
7 8				8 9	7 8 9	7 8 9	7 8 9	7 8 9	7 8	7 8 9			7 8 9	
2 3	1 2			1	3	1 2 3	1 2 3	1 2 3	2 3	1 2			1	3
4 5 6	4 5 6			4	6	4 5 6	4 5 6	4 5 6	4 5 6	5 6			4	6
7 8				8 9	7 8 9	7 8 9	7 8 9	7 8 9	7 8	7 8 9			7 8 9	
2 3	1 2			1	3	1 2 3	1 2 3	1 2 3	2 3	1 2			1	3
4 5 6	4 5 6			4	6	4 5 6	4 5 6	4 5 6	4 5 6	5 6			4	6
7 8				8 9	7 8 9	7 8 9	7 8 9	7 8 9	7 8	7 8 9			7 8 9	
9		1			3	1 2 3	1 2 3	1 2 3	2		2			
	4		6		4	4 6	4 6	4 6		6			5	
			8			7 8	7 8	7 8	7 8		7 8	6		
					8	7 8	7 8	7 8						
	4	5	6			1 2	1 2	1 2		2			3	
				1		4 5 6	4 5 6	4 5 6	4	6			4	6
			8			8 9	8 9	8 9		8			8 9	
	3					3	3	3	1					
4	5	6		4	5	4 5 6	4 5 6	4 5 6			6		4	6
				8		7 8 9	7 8 9	7 8 9			7 8 9		7 8	

比如这个示例里，我们把图上的链接点都标注了出来。不过，细数一下链接点总数，就可以发现，此时结构涉及一共 20 个链接点，因为四个链接点是我们变为结构形式时补充上去的，此时是 16\20 网，所以无法直接得到 SK 环原本的删数。

仔细看一下，这个结构多出来的四个链接点分别在哪里：一个是 c8 的 7、一个是 r2 的 7、一个是 c2 的 4，一个是 r8 的 8，而仔细观察原本的题目里，这些数字在所在的区域下全部都有确定值，它们把这些数字的影响都排除掉了，所以 SK 环就直接成立了。

那么，SK 环的直观具有哪些特性呢？首先结构是分属于四个宫的，并且这四个宫是类似于田字形分布的，并不是错开分布的；而且四个宫里，每一个宫都只有三个确定值，且确定值里，对角分布的两个宫里，抛开所谓的轴心不看，剩下的数字则是完全一样的。这就是 SK 环在直观的时候需要满足的架构。

1	2	4	6	4	6	2 3	2 3	6	8	5	6	3	6
2	8	7	8	7	8	4 5	4 5	7	8	7	8	6	6
3	6	7	8	9	7	8	9	8	9	7	8	5	6
4	6	4	6	4	6	1 2	1 2	3	3	1	5	6	2
7	8	8	9	7	8	5 6	5 6	7	8	9	7	8	6
2 3	1 2	1	3	1	3	7 8 9	7 8 9	4	5	6	4	6	3
4 5 6	4 5 6	4	6	4	6	4 5 6	4 5 6	4	6	4	6	4	6
7 8	8 9	7	8	9	7	8	9	7	8	9	7	8	9
2 3	1 2	1	3	1	3	2 3	1 2	1 2	2 3	1 2	1	3	3
4 5 6	4 5 6	4	6	4	6	4 5 6	4 5 6	4	6	4	6	4	6
7 8	8 9	7	8	9	7	8	9	7	8	9	7	8	9
7	2	4	6	4	6	1 2	1 2	3	3	1	5	6	2
2	8	7	8	7	8	5 6	5 6	7	8	9	7	8	6
3	6	7	8	9	7	8	9	8	9	7	8	5	6
4	6	4	6	4	6	1 2	1 2	3	3	1	5	6	2
7	8	8	9	7	8	5 6	5 6	7	8	9	7	8	6
2 3	1 2	1	3	1	3	7 8 9	7 8 9	4	5	6	4	6	3
4 5 6	4 5 6	4	6	4	6	4 5 6	4 5 6	4	6	4	6	4	6
7 8	8 9	7	8	9	7	8	9	7	8	9	7	8	9

如图所示，这便是 SK 环的一个基本的示例。

4-4-3-3 JE + SK = PLQ

可以从之前学习到的 SK 环和 JE 技巧来看，它们的直观层面是比较相似的。不过 JE 我

他们没有讲述直观视角（严谨地说，是只讲了 JE 里只有一个基准单元格的特殊 JE 构型，因为它比较容易直观，可以使用代数法的视角），但实际上 JE 可以转为网的视角，所以使得结构涉及的确定值分布也是近乎类似的。

要是我们把这两个技巧放到一起呢？

1	4	7	9	7	8	9	6	4	6	4	6	4	5	6	5	5	3
4	7	8	5	7	8	3	1	2	3	7	8	9	6	4	5	6	3
7	8	6	7	9	2	1	3	1	3	1	3	1	3	4	1	2	3
2	1	2	3	1	3	5	1	2	3	1	2	3	1	3	8	4	2
4	7	8	9	6	7	8	9	6	7	8	9	6	7	8	9	6	7
9	1	2	3	1	3	4	1	2	3	8	4	1	2	3	6	1	2
4	5	8	6	1	3	1	2	3	4	1	2	3	4	5	6	7	8
5	6	7	8	9	6	7	8	9	6	7	8	9	6	7	8	9	6
3	2	1	4	5	6	7	8	9	6	7	8	9	6	7	8	9	6
7	8	9	6	7	8	9	6	7	8	9	6	7	8	9	6	7	8
6	7	8	9	6	7	8	9	6	7	8	9	6	7	8	9	6	7
5	6	7	8	9	6	7	8	9	6	7	8	9	6	7	8	9	6
4	5	6	7	8	9	6	7	8	9	6	7	8	9	6	7	8	9
3	4	5	6	7	8	9	6	7	8	9	6	7	8	9	6	7	8
2	3	4	5	6	7	8	9	6	7	8	9	6	7	8	9	6	7
1	2	3	4	5	6	7	8	9	6	7	8	9	6	7	8	9	6

如图所示，这是同一个题目的两个不同的技巧。一个是 JE（左图），一个是 SK 环（右图）。显然，它们的删数不都一样（当然了，JE 如果使用“潜规则”，那么可能会有一样的删数效果，不过我们就不考虑了，这不是我们这里要说的重点）。

现在，我们尝试把这些得到的删数都去掉，看看还有没有结论出现。首先，我们得到它们之后，r5 会出现一个 1、2、3、4 的四数组，这是很容易发现的，我们就不用额外作图说明了。

接着我们要看另外一处结论。

1	4	7	9	7	8	9	6	4	6	4	6	4	5	6	5	5	3
4	7	8	5	7	8	3	1	2	3	7	8	9	6	4	5	6	3
7	8	6	7	9	2	1	3	1	3	1	3	1	3	4	1	2	3
2	1	2	3	1	3	5	1	2	3	1	2	3	1	3	8	4	2
4	7	8	9	6	7	8	9	6	7	8	9	6	7	8	9	6	7
9	1	2	3	1	3	4	1	2	3	8	4	1	2	3	6	1	2
4	5	8	6	1	3	1	2	3	4	1	2	3	4	5	6	7	8
5	6	7	8	9	6	7	8	9	6	7	8	9	6	7	8	9	6
3	2	1	4	5	6	7	8	9	6	7	8	9	6	7	8	9	6
7	8	9	6	7	8	9	6	7	8	9	6	7	8	9	6	7	8
6	7	8	9	6	7	8	9	6	7	8	9	6	7	8	9	6	7
5	6	7	8	9	6	7	8	9	6	7	8	9	6	7	8	9	6
4	5	6	7	8	9	6	7	8	9	6	7	8	9	6	7	8	9
3	4	5	6	7	8	9	6	7	8	9	6	7	8	9	6	7	8
2	3	4	5	6	7	8	9	6	7	8	9	6	7	8	9	6	7
1	2	3	4	5	6	7	8	9	6	7	8	9	6	7	8	9	6

如图所示，如果我们只讨论这几个单元格，依然拥有结论。首先我们通过基础的 JE 逻辑，假设 r5c46 是 a 和 b，那么 r4c2 = a、r6c8 = b 的话，显然此时位于 r5 的 1、

像是图上这样，通过 SK 和 JE 结合，得到的跨区四数组，我们称之为**固定产生的构型四数组**（**Pattern Locked Quadruple**，简称 **PLQ**）。不过，不同的 SK 环的提示数分布具有不同的 PLQ 的结论，下面我们来试试看一个示例，这个示例就拥有另外一处 PLQ。

1			4 5	2 3		2 3		4	2 3	6	1		4 5	2 3		2 3		4	2 3	6
	7 8 9		7 8 9			7		8 9				7 8 9						8 9		
4	6	3	4		1 2		4	2		5	1 2		4 6		1 2		2		5	1 2
7	9		7	9		7		9				7	9				9			9
4 5	6			6	1 3	1	4 5	6	5 6		1 3		4 5 6		1 3		5 6		1 3	4 5 6
8 9			8 9										8 9						8 9	
	6	1 2	6	1		5	1 2	4	2	6	3	1 2		6		5	1 2	4	2	6
7 8 9	7 8 9		7 8 9	7 8 9			7		8 9			8 9				8 9				8 9
5 6	1 2		6	5	1 2		6	3	1 2	6		3	1 2		6	4 5 6	1 2	6	4 5 6	1 2
7 8 9	7 8 9		7 8 9	7 8 9		7		8 9		7 8 9		8 9				7 8 9		7 8 9		8 9
5 6	3	1 3			8	1 2	6	9	2	5 6		6	5	1 2		3	2	1 2	6	5
7		7				7			7								7			
4	3				6	2 3		2	3		1	2	3		5	8 9		2	3	5
7 8 9	7 8 9		7 8 9			7 8				8 9					8 9					8 9
7	8				1 2 3		6		6		4		8		2 3		6		8	
7 8						7 8														
2	1		1	3	1 3	1	4 6	4 5 6	5 6		8		5 6		6	8 9		6		7
			8 9										8 9							

1		4 5	2 3	4 5	2 3	2 3	2	6
	7 8 9	7 8 9	7 9	7	7	8 9	8 9	
4 6	3	4	1 2	6 8	1 2	4	5	1 2
7 9		7 9	7 9	7	9	9		4
4 5 6		6	1 3	1	1 3	1	1	1 3
8 9		8 9	4 6	4 5 6	5 6	7	8 9	4 8 9
	1 2	1	5	1 2	6	4	2	1 2
6	7 8 9	6	7 8 9	7			6	8 9
7 8 9	7 8 9	7 8 9	1 2	6	3	1 2	2	1 2
5 6		6	1 2	6	7	6	4 5 6	6
7 8 9	7 8 9	7 8 9	7		7		7 8 9	4 5 8 9
3	4	1 3	8	1 2	6	9	2	1 2
7	5 6	7	7	7			5 6	7
4 3		6	2 3	4 5	2 3	1	2	2 3
7 8 9	7 8 9		7	7	7 8		8 9	5 8 9
3	5	1 3	1 2 3	6	1 2 3	2 3		2 3
7 8		7 8	7		7 8	6	8	8
2	1	1 3	1 3	1	1 3	3		
	8 9	8 9	4 6	4 5 6	8	5 6	6	7
						8 9	8 9	

545

1		4 5	2 3	2	2 3	3	2	6
	7 8 9	8 9	4	4 5	7	4	8 9	
4	6	3	1 2	6	1 2	6	5	1 2
7	9		7	9	7	9		4
4 5		6	1 3	1	1 3	7	1	3
8 9	8 9	2	4	6	4 5 6	5 6	8 9	8 9
	1 2	1	5	1 2	6	2	3	1 2
6	7 8 9	7	7	7	7	8 9	8 9	
5	1 2	6	1 2	6	1 2	6	1 2	6
8 9	7	8 9	7	8 9	7	8 9	7	8 9
3	5 6	1	1 3	1 2	6	2	1 2	1 2
7	7	4	5	7	6	5 6	7	5
4	3		6	2 3	2	2 3	1	2
8 9	7 8 9		7	4 5	5	7 8	8 9	5
	3	1	1 2	1 2	6	2 3	2 3	2 3
7 8	5	7 8	7	6	7	6	8	8
2	1	3	1 3	1	1 3	3		7
	8 9	8 9	4	6	4 5 6	5 6	8 9	

现在我们要尝试解决，这四个删数到底是什么情况。我们先假设 $r4c2 = 7$ 。如果 $r4c2 = 7$ ，则意味着 JE 的基准单元格 $r5c46$ 里必然会有数字 7 的出现，即形成 7 区块。不过观察 $b6$ 就会发现，由于提示数的分布，这样的填法导致 $b6$ 无法放入 7，出现矛盾。所以 $r4c2 \neq 7$ ；同理，我们假设 $r5c2 = 7$ ，就会使得 $b6$ 仅能让 $r6c8 = 7$ ，而它也是目标单元格的其一，所以依然可以得到基准单元格里含有 7 的结论。但这也是不可能的，因为 $r5c2 = 7$ 使得 $r5$ 上不允许再放入 7 了，所以出现矛盾。

同样地， $r56c8(2)$ 的删数也是完全一样的假设模式，所以我们可以得到图上的这四个删数结论。

1		4 5	2 3	2	2 3	3	2	6
	7 8 9	8 9	4	4 5	7	4	8 9	
4	6	3	1 2	6	1 2	6	5	1 2
7	9		7	9	7	9		4
4 5		6	1 3	1	1 3	7	1	3
8 9	8 9	2	4	6	4 5 6	5 6	8 9	8 9
	1 2	1	5	1 2	6	2	3	1 2
6	7 8 9	7	7	7	7	8 9	8 9	
5	1 2	6	1 2	6	1 2	6	1 2	6
8 9	7	8 9	7	8 9	7	8 9	7	8 9
3	5 6	1	1 3	1 2	6	2	1 2	1 2
7	7	4	5	7	6	5 6	7	5
4	3		6	2 3	2	2 3	1	2
8 9	7 8 9		7	4 5	5	7 8	8 9	5
	3	1	1 2	1 2	6	2 3	2 3	2 3
7 8	5	7 8	7	6	7	6	8	8
2	1	3	1 3	1	1 3	3		7
	8 9	8 9	4	6	4 5 6	5 6	8 9	

在得到了上述的删数后，SK 环的结构和盘面变为如图所示的这样。

现在，我们需要打开脑洞，思考 $r28c46$ 这四个单元格会有如何的结论。实际上，这四个单元格是一个关于 1、2、6、7 的跨区四数组，即是一组 PLQ。这是为什么呢？

我们仔细观察 SK 环里，提示数的分布情况，就会发现，1、2、6、7 恰好是我们涉及的结构数字，而且每一个宫的提示数列出来是 1 和 2、6 和 7、2 和 6 以及 1 和 7。这样四组分布情况有一个看得出但说不出的特性。

我们为了解释这种特性，我们就将视角精细化。仔细观察 SK 环转换为网视角后的宫链接点，就会发现一个神奇的地方，每一个宫的宫链接点，都只有两个，而且有一个宫链接点必然占据了三个单元格，而还有一个则只占据两个单元格。且两个宫链接点必然会共用一个单元格。这些细节是我们稍后都会用到的。

思考一下。四个宫的宫链接点的分布如此“对称”，这意味着结构本身是高度对称的，这就是我们想说的特性：提示数的分布是高度对称的。

如果分布是高度对称的话，我们可以类比宇宙法的思路。由于高度对称的结构形成，必然可以得到的是，我们不能够去破坏这种对称性，否则结构会立马失去平衡，导致填数无法正确放入，出现矛盾。所以，我们为了保证结构依旧是完整和正确的，我们只能使得 r28c46 这四个可能会影响到结构的单元格是一个四数组。如果这四个单元格只填入两种候选数，显然是矛盾的，因为形成了 UR 的致命形式；如果只填入三种候选数，显然这样的填数模式影响到了高度对称的宫链接点，使得结构立刻变为不稳定状态，使得结构出现类似于宇宙法的矛盾，所以，四个单元格必须是四个完全不同的数字。

那么，这一点有什么用处呢？我们尝试枚举出所有填数情况。我们拿出 r2c46 进行枚举，r2c46 一共存在 6 种不同的填数情况：

- 1 和 2；
- 1 和 6；
- 1 和 7；
- 2 和 6；
- 2 和 7；
- 6 和 7。

我们首先从 1 和 2 这一对情况说起。

1		4 5	2 3	2	2 3	3	2	6
	7 8 9	8 9	4	4 5	7	4	8 9	
4 6	3	4	1 2	8	1 2	2	5	1 2
7	9	7	6	7	6	9	4	9
4 5		6	2	1 3	1 3	7	1	3
	8 9	8 9		4 6	4 5 6	5 6	8 9	8 9
	6	1	5	1 2	6	4	3	1 2
7 8 9	7	7 8 9		7		8 9	8 9	
5	1 2	6	5	1 2	6	4 5	1 2	6
8 9	7	8 9	7	7	6	8 9	7	8 9
3	4	1 3	8	1 2	6	9	2	1 2
7	6	7	7	7		5 6	1 2	6
4	3		6	2 3	2	2 3	1	3
	8 9	7 8 9	7	4	4 5	5	8 9	5
	3	1 3	1 2	6	1 2	6	2 3	2 3
7 8	5	7 8	7	9	7	8	4	8
2	1	4	1 3	1	1 3	3	6	7
	8 9	8 9	4 6	4 5 6	5 6	8 9	8 9	

如图所示，假设 r2c46 是 1 和 2 的数对，那么 r8c46 只能是 6 和 7 的数对，于是观察 b39，由于刚才说到的那种分布模式，最终我们可以根据这种模式直接确定此时 1、2、6 的位置，那么，r56c8 同时只能填入 7，显然是违背数独规则的。

下面我们来看 1 和 6 的组合。

1		4 5	2 3	2	2 3	3	2	6
	7 8 9	8 9	4 5	7 9	7	5	4	8 9
4 6	3	4	1 2	8	1 2	2	5	1 2
7 9		7 9	6		6	4		9
4 5		6	2	1 3	1 3	7	1	3
8 9	8 9	8 9		4 6	4 5 6	5 6	8 9	4
	1 2	1	5	1 2	4	2	3	1 2
7 8 9	7	6	7 8 9	7	6	8 9		8 9
5	1 2	6	1 2	3	1 2	4 5	1 2	4 5
8 9	7	8 9	7	6	7	8 9	7	6
3	5 6	4	1 3	8	1 2	9	2	1 2
7	5 6	7	5	7	6	5 6	7	6
4	3		6	2 3	2	2 3	1	2
8 9	7 8 9	7 8 9	4	4 5	7	5	8 9	5
3		1 3	1 2	6	1 2	2 3	4	2 3
7 8	7 8	7 8	7	6	7	6	8	8
2	1	4	3	1 3	1 3	5	6	7
	8 9	8 9	4 6	4 5 6	5 6	8 9	8 9	

如果假设好 1 和 6 的话，我们可以发现，此时并未产生任何的矛盾。我们不能认为这种情况是错误的。所以我们先暂时放一边。

1		4 5	2 3	2	2 3	3	2	6
	7 8 9	8 9	4 5	7 9	7	5	4	8 9
4 6	3	4	1 2	8	1 2	2	5	1 2
7 9		7 9	6		6	4		9
4 5		6	2	1 3	1 3	7	1	3
8 9	8 9	8 9		4 6	4 5 6	5 6	8 9	4
	1 2	1	5	1 2	4	2	3	1 2
7 8 9	7	6	7 8 9	7	6	8 9		8 9
5	1 2	6	1 2	3	1 2	4 5	1 2	4 5
8 9	7	8 9	7	6	7	8 9	7	6
3	5 6	4	1 3	8	1 2	9	2	1 2
7	5 6	7	5	7	6	5 6	7	6
4	3		6	2 3	2	2 3	1	2
8 9	7 8 9	7 8 9	4	4 5	7	5	8 9	5
3		1 3	1 2	6	1 2	2 3	4	2 3
7 8	7 8	7 8	7	6	7	6	8	8
2	1	4	3	1 3	1 3	5	6	7
	8 9	8 9	4 6	4 5 6	5 6	8 9	8 9	

然后发现，1 和 7 的组合依然是不行的。和上述逻辑类似，我们可以发现，r56c8 都只能是 7。

1			2 3	2	2 3	3	2	6
	7 8 9	4 5	4 7	4 5	7 5	4 8 9	8 9	
4 6	3	4	1 2	8	1 2	2	5	1 2
7 9		7 9	6		6	4		4 9
4 5		6	2	1 3	1 3	1		3
8 9	8 9			4 6	4 5 6	5 6	7	4 8 9
	1 2	6	5	1 2	6	2	3	1 2
7 8 9	7	7 8 9		7		8 9		8 9
5	1 2	6	1 2	6	1 2	6	4 5	6 4 5
8 9	7	8 9	7		7	8 9	7	8 9
3	4	1 3	8	1 2	9	2	1 2	1 2
5 6	7	5	7	6	7	5 6	6	5
	3		6	2 3	2	2 3	1	2
4 8 9	7 8 9		4 7	4 5	7 5	7 8	8 9	5 8 9
	3	1 3	1 2	9	1 2	2 3	4	2 3
7 8	5	7 8	7	6	7	6		8
2	1		1 3	1	1 3	3		7
	8 9	4	4 6	4 5 6	5 6	8 9	6 8 9	

接下来我们发现，如果 r2c46 是 2 和 6 的数对的话，b17 会出现矛盾，如图所示，逻辑和上面完全一样，这里就不作出说明了。

1			2 3	2	2 3	3	2	6
	7 8 9	4 5	4 7	4 5	7 5	4 8 9	8 9	
4 6	3	4	1 2	8	1 2	2	5	1 2
7 9		7 9	6		6	4		4 9
4 5		6	2	1 3	1 3	1		3
8 9	8 9			4 6	4 5 6	5 6	7	4 8 9
	1 2	6	5	1 2	6	2	3	1 2
7 8 9	7	7 8 9		7		8 9		8 9
5	1 2	6	1 2	6	1 2	6	4 5	6 4 5
8 9	7	8 9	7		7	8 9	7	8 9
3	4	1 3	8	1 2	9	2	1 2	1 2
5 6	7	5	7	6	7	5 6	6	5
	3		6	2 3	2	2 3	1	2
4 8 9	7 8 9		4 7	4 5	7 5	7 8	8 9	5 8 9
	3	1 3	1 2	9	1 2	2 3	4	2 3
7 8	5	7 8	7	6	7	6		8
2	1		1 3	1	1 3	3		7
	8 9	4	4 6	4 5 6	5 6	8 9	6 8 9	

接下来是 2 和 7 的数对组合。显然这种情况依旧没有矛盾。

1			4 5	2 3	2	2 3	3	2		6
	7 8 9		8 9	4	7 9	4 5	7	5	4	8 9
4 6	3	4		1 2	6	8	1 2	2		5
7 9		7 9		7		7	6	4		1 2
4 5		6	2	1 3	1	4	6	4 5 6	5 6	7
8 9	8 9	8 9		4	6	4 5 6	9	1	5 6	8 9
	1 2	6	1	5	1 2	6	4	2	6	3
7 8 9	7	7 8 9	7 8 9	7	7	7	8 9	8 9	8 9	1 2
5	1 2	6	5	1 2	6	3	1 2	6	4 5	7
8 9	7	8 9	5 8 9	7	6	7	6	4 5	7	8 9
3	4	1 3	8	1 2	6	9	2	1 2	6	1 2
5 6	7	5	7	7	6	7	5 6	7	6	5
4	3		6	2 3	2	2 3	1	2		3
8 9	7 8 9	7 8 9		4	4 5	5	7 8	8 9	5	8 9
3		1 3	1 2	6	7	1 2	6	2 3	6	2 3
7 8	5	7 8	7 8	7	6	9	7	8	4	8
2	1		3	1 3	1	1 3	5	3		7
	8 9		8 9	4	6	4 5 6	5 6	8 9	6	8 9

6 和 7 的组合也是错误的。

这六个图我没有去掉本来删掉了的 $r45c2(7)$ 和 $r56c8(2)$ ，是因为使用的枚举完全没有用到这个地方的删数逻辑，所以为了示意更明确、更完整，就没有在图上去掉这些原本已经删除掉的数字。

这样我们通过枚举，就把所有情况都走了一遍，最终发现，只有 1 和 6 的组合，或是 2 和 7 的组合是满足要求的。当然，这也同时说明了一点， $r5c46$ 不能是 1 和 6 的数对，也不能是 2 和 7 的数对，否则将会和 PLQ 给出的其中两个单元格形成 UR 的致命形式。

1			4 5	2 3	2	2 3	3	2		6
	7 8 9		8 9	4	7 9	4 5	7	5	4	8 9
4 6	3	4		1 2	6	8	1 2	2		5
7 9		7 9		7		7	6	4		1 2
4 5		6	2	1 3	1	4	6	4 5 6	5 6	7
8 9	8 9	8 9		4	6	4 5 6	9	1	5 6	8 9
	1 2	6	1	5	1 2	6	4	2	6	3
7 8 9	7	7 8 9	7 8 9	7	7	7	8 9	8 9	8 9	1 2
5	1 2	6	5	1 2	6	3	1 2	6	4 5	7
8 9	7	8 9	5 8 9	7	6	7	6	4 5	7	8 9
3	4	1 3	8	1 2	6	9	2	1 2	6	1 2
5 6	7	5	7	7	6	7	5 6	7	6	5
4	3		6	2 3	2	2 3	1	2		3
8 9	7 8 9	7 8 9		4	4 5	5	7 8	8 9	5	8 9
3		1 3	1 2	6	7	1 2	6	2 3	6	2 3
7 8	5	7 8	7 8	7	6	9	7	8	4	8
2	1		3	1 3	1	1 3	5	3		7
	8 9		8 9	4	6	4 5 6	5 6	8 9	6	8 9

1			4 5	2 3	2	2 3	3	2		6
	7 8 9		8 9	4	7 9	4 5	7	5	4	8 9
4 6	3	4		1 2	6	8	1 2	2		5
7 9		7 9		7		7	6	4		1 2
4 5		6	2	1 3	1	4	6	4 5 6	5 6	7
8 9	8 9	8 9		4	6	4 5 6	9	1	5 6	8 9
	1 2	6	1	5	1 2	6	4	2	6	3
7 8 9	7	7 8 9	7 8 9	7	7	7	8 9	8 9	8 9	1 2
5	1 2	6	5	1 2	6	3	1 2	6	4 5	7
8 9	7	8 9	5 8 9	7	6	7	6	4 5	7	8 9
3	4	1 3	8	1 2	6	9	2	1 2	6	1 2
5 6	7	5	7	7	6	7	5 6	7	6	5
4	3		6	2 3	2	2 3	1	2		3
8 9	7 8 9	7 8 9		4	4 5	5	7 8	8 9	5	8 9
3		1 3	1 2	6	7	1 2	6	2 3	6	2 3
7 8	5	7 8	7 8	7	6	9	7	8	4	8
2	1		3	1 3	1	1 3	5	3		7
	8 9		8 9	4	6	4 5 6	5 6	8 9	6	8 9

下面我们使用涂色。由于我们确定最终 PLQ 只有两种填数模式，我们就分两种情况进行涂色。第一种情况是左图这样，我们可以根据这种涂色模式，得到 $r19c28$ 只有 8 和 9，于是形成一个特殊的结构，看起来像是 X-Wing，而且是绑定的 X-Wing。我们通过这个结构，可以直接断定， $r19$ 不能再填入其它的 8 和 9 了，因此，此时的 $r1c37 <> 9$ ， $r9c37 <> 8$ 。

同理，第二种情况是右图这样。我们此时可以得到， $r37c28$ 形成类似于刚才的绑定的

X-Wing 结构，不过此时需要借助 b28 的共轭对。由于关于 8 和 9 的结构成立，使得此时 $r3c4 \neq 9$, $r7c6 \neq 8$ 。由于 b2 存在 9 的共轭对，而 b8 存在 8 的共轭对，所以 $r1c4 = 9$ 、 $r9c6 = 8$ 。这样一来，我们依然可以得到 r1 和 r9 的其余位置不能放入 9 和 8，所以此时 $r1c37 \neq 9$ ，而且 $r9c37 \neq 8$ 。

由此可见，我们通过费力的枚举，最终得到了应有的“回报”，这四个删数是我们预料不到的地方。

4-4-4 JE 的直观

又是一个优秀的标题。怎么，想让我直观 JE 这种破结构？当然是的。我们来看看 JE 是怎么转换视角的。

3	1	3	1	3	3	3	2	1	3	1	3	1
4	5	6	4	5	6	4	4	5	7	8	2	4
8	8						8	7	8			7
2	3	1	3				3	1	2	3		1
4	5	4	5				4	5	7			4
		9					6			8		7
2	3			2	3			2	3			2
4	5	7		4			1	5	8	9		4
												6
4	5	3		4	5		6	2	3			1
7							9	4	5			2
		6								8	1	2
											5	7
4	5		2				1			1		1
7	9		7				4	5	6	8	7	4
												6
												3
1			4	5		3		2		2		2
						9	8	7		5	6	4
												9
2	3	1	3	1	2	3		2	3	1	2	3
7	8	9		8	9	7		5	6			7
												4
2	3	1	3	1	2	3		2	3	1	2	3
4	6	4	6	4				6				1
7	8	9		8	9	7		8	9	7		7
												5
2	3	1	3					2	3			1
7												9
												8

我们依然拿出最最标准的示例给大家。因为 JE 的变体类型结构实在是太多了，所以很多时候，推论的删数不一定在任意时刻都成立，所以为了阐述这里的直观视角，我选择了一个最基础的示例。这则示例在前文里已经被提及了两次，但我依旧喜欢拿出这个示例来解释一些东西，因为它最基础。

我们参照网的直观视角，将 JE 用直观的方式呈现出来。

4 5 6 8	3 1 3 4 5 6 4	1 3 4	3 1 3 4 5 6 4	3 1 3 4 5 6 4	2 1 3 4 5 6 4	1 3 4	1 3 4	1 3 4
2 3 4 5	1 3 4 5	9	3 1 3 4 5 6 4	6 1 3 4 5 6 4	3 1 3 4 5 6 4	1 2 3 4 5 6 4	8 1 2 4 5 6 4	1 2 4 5 6 4
2 3 4 5 8	7 4	1	2 3 4 5	3 1 3 4 5 6 4	9 1 3 4 5 6 4	2 3 4	2 3 4	6 1 2 4 5 6 4
3 1 3 4 5 6 4	4 5 7	6	2 3 4 5	9 1 3 4 5 6 4	8 1 3 4 5 6 4	1 2 3 4 5 6 4	1 2 4 5 6 4	1 2 4 5 6 4
4 5 7 9	2 4 7	1	4 5 6 8	1 5 8	4 5 6 8	1 4 7	1 5 7	3 1 2 4 5 6 4
1 4 5 9	3 8 9	7	2 3 5	3 1 3 4 5 6 4	2 6 4	2 5 6 4	2 5 6 4	2 5 6 4
2 3 7 8 9	1 3 6 6	1 2 3 7 8 9	2 3 5 6	1 2 3 5 6	1 3 5 6	1 2 3 6 7	4 1 2 7 8 9	1 2 7 8 9
2 3 4 6 7 8 9	1 3 4 6 7 8 9	1 2 3 4 6 7 8 9	2 3 5 6	1 2 3 5 6	1 3 5 6	5 1 2 3 7 8 9	1 2 7 8 9	1 2 7 8 9
2 3 7	1 3 6	5	2 3 6	4 1 3 7 6	9 1 3 7 6	1 2 3 7 6	8 1 2 7 6	8 1 2 7 6

如图所示，具体形成这样的形式我们就不再详细描述了，这一步其实很简单，直接画线分配填数位置就可以了。而根据上述的图上的分配形式，现在我们可以得到的是：

交叉格 + 紫色单元格 = 4 套 1 到 9

交叉格 + 橙色单元格 = 5 套 1 到 9

所以

橙色单元格 = 1 套 1 到 9 + 紫色单元格

如图所示（上述三个公式配合这个图一起理解）。注意，此时图上已经没有 b9 的涂色，因为 b9 恰好是一套 1 到 9，我们完全可以拿来用于抵消公式里的“1 套 1 到 9”这部分。

4 5 6 8	3 1 3 4 5 6 4	1 3 4	3 1 3 4 5 6 4	3 1 3 4 5 6 4	2 1 3 4 5 6 4	1 3 4	1 3 4	1 3 4
2 3 4 5	1 3 4 5	9	3 1 3 4 5 6 4	6 1 3 4 5 6 4	3 1 3 4 5 6 4	1 2 3 4 5 6 4	8 1 2 4 5 6 4	1 2 4 5 6 4
2 3 4 5 8	7 4	1	2 3 4 5	3 1 3 4 5 6 4	9 1 3 4 5 6 4	2 3 4	2 3 4	6 1 2 4 5 6 4
3 1 3 4 5 6 4	4 5 7	6	2 3 4 5	9 1 3 4 5 6 4	8 1 3 4 5 6 4	1 2 3 4 5 6 4	1 2 4 5 6 4	1 2 4 5 6 4
4 5 7 9	2 4 7	1	4 5 6 8	1 5 8	4 5 6 8	1 4 7	1 5 7	3 1 2 4 5 6 4
1 4 5 9	3 8 9	7	2 3 5	3 1 3 4 5 6 4	2 6 4	2 5 6 4	2 5 6 4	2 5 6 4
2 3 7 8 9	1 3 6 6	1 2 3 7 8 9	2 3 5 6	1 2 3 5 6	1 3 5 6	1 2 3 6 7	4 1 2 7 8 9	1 2 7 8 9
2 3 4 6 7 8 9	1 3 4 6 7 8 9	1 2 3 4 6 7 8 9	2 3 5 6	1 2 3 5 6	1 3 5 6	5 1 2 3 7 8 9	1 2 7 8 9	1 2 7 8 9
2 3 7	1 3 6	5	2 3 6	4 1 3 7 6	9 1 3 7 6	1 2 3 7 6	8 1 2 7 6	8 1 2 7 6

然后，结构就成了这样。显然，此时橙色和紫色直接就是填数完全一样的构型了。不过，光是得到这个结论还不够，因为它并不足以得到删数结果。因为我们细数确定值就可以发现，

橙色单元格含有 8 个空格，但紫色的确定值只有 6 个，显然会空出两个单元格，依然我们是无法确定填什么的。不过别担心，我们还有一些情况是可以立马排除的。

细看结构，由于原本 JE 给出的基准单元格只有 1、2、7，而如果让 r24c9 都填入 1、2、7 的话，显然是不允许的，因为四个单元格都只有 1、2、7 是显然矛盾的。所以 r24c9 只能有一个单元格是 1、2、7，而另外一个单元格则不允许是 1、2、7。第二，基准单元格还不允许让 r89c35 全都是 1、2、7，否则四个单元格配合 r89c9 构成 1、2、7 的拓展矩形的致命形式。所以 r89c35 里至少有一个数也不是 1、2、7。所以，我们就可以确定 1、2、7 的最终填入位置了。

3	1	3	1	3	3		2	1	3	1	3	1
4 5 6	4 5 6	4	4 5 6	7 8	5	7 8		4	5	7	9	4 5
8	8							7	9	7		9
2 3	1	3		3	6	4 5	3	1 2 3	8	1 2		
4 5	4 5		9	4 5				4		4 5		
								7		7		
2 3		2 3		3	9	4	2 3	2 3		2 3		6
4 5	7	4		1	5	8		4	5			
8												
3	3		6	2 3	9	1 3		8	1 2	1 2		
4 5	4 5			4 5		4 5			5	4 5		
7									7	7		
4 5	2	4		4 5 6	5	1	4 5 6	1	4	6	5 6	3
7	9	7		8	8	8	8	7	7	9		
1	4 5	3	8	7	2 3	3	4 5 6	2	2	2	2	
	9				5			4	6	5 6	4 5	9
										9		
2 3	1	3	1 2 3	2 3	1 2 3	1	3	1 2 3		1 2		
6	6		7	5 6	5	5 6		6	4			
7 8 9	8 9			8 9	7 8	7 8		7		7		
2 3	1	3	1 2 3	2 3	1 2 3	1	3		1 2 3	1 2		
4 6	4	6	4	6	6	6		5	6	6		
7 8 9	8 9		7	8 9	7 8	7 8			7	7		
2 3	1	3		2 3		1	3		1 2 3			
6	6		5	6		6		9	6			8
7												

如图所示，我们不管 r89c35 里哪三个单元格是 1、2、7，也不用管 r24c9 哪一个单元格是 1、2、7，这都不重要，因为我们还有两个单元格是完全可以确定下填入 1、2、7 的位置的，即 r2c7 和 r3c8，这两个单元格是一定只能是 1、2、7 的，否则 1、2、7 不够填入到橙色单元格里，使得橙色还有的其余位置必然会出现矛盾，要么四个单元格填入 1、2、7 三种数字出错，要么就是拓展矩形的致命形式导致的出错。

这就是 JE 的直观。虽说是直观，但依旧需要一点点候选数的视角来辅助我们的逻辑推理，而它恰好借用了完整的宫，来抵消多余的 1 套 1 到 9；但比较遗憾的是，显然这种结构只能是分布呈现类似于同宫定理需要满足的样子。如果是四种数字的话，由于确定值的分布不再像这样简单，所以不便推理，很少出现这样的构型；而其它的变种就更难说了。

下面我们来尝试理解一些示例。如图所示。另外，需要你自行判断逻辑的结论，以及删数到底是如何形成的，或是能否通过直观视角删除这些标注出来的候选数。如果可以删除，请自行说明理由。不过标注的删数都是通过基础的 JE 的删数模式得到的删数，不代表直观视角可以删除。

9 4 5 6 7	2 4 5 6 7	3 4 5 6 7	3 7	2 6 7	2 3 7	8 1 5 6 4 5 6	1 2 3 4 5 6 7	1 2 3 4	1 3 4	7	1 2 3 4 5 6 7	1 4 5 6 7	2 3 5 6 7	1 2 3 4 5 7	9	8
2 4 5 6 7	1 4 8 7	2 4 8 7	2 7 8 9 7	2 7 8 9 7	5 7 8 9 7	3 4 6 9 7	2 4 6 7	6 1 3 4	1 2 3 4	9 1 4 5 8	2 3 5 8 7	7 2 3 5	2 3 1 2 4 5			
2 5 6 8	5 6 8	3 5 6 8	4	1 8 9	2 8 9	2 5 6 9	5 6 9	5 1 3 4	1 2 3 4	1 2 3 4	1 4 8 7	6 2 3 7	6 2 3 7	1 2 3 4		
1 3 5 6 7	5 6 7 8	5 6 8	2 7 8 9	4 7 8 9	1 3 5 6 7	1 3 5 6 7	1 3 5 6 7	1 7 8 9	5 7 9	1 5 9	2 6 8	3 2 6 7 8 9	4 1 2 5 8	1 2 3 4	4 5 6 7	9 1 2 3 4
1 3 5 6 7	2 1 6 8	1 6	5 7 8 9	1 3 5 6 7	1 3 5 6 7	4 7 8 9	1 3 5 6 7	4 7 8	2 4 5 3	4 6 8	4 6 7 8	1 5 8	5 6 7 8	3 4 5 6	9 1 2 3 4	
1 3 4 5 7	9 4 5 8	1 4 5 8	6 7 8	1 3 5 6 7	1 3 5 6 7	1 2 3 4 5 7	1 2 3 4 5 7	1 3 4 5 7	1 3 4 5 7	6 4 8 9	5 4 8 9	4 7 8 9	7 8 9	1 2 3 4	2 3 4 5	1 2 3 4
8 4 5 7	3 4 6 9	1 4 6 9	5 7 8 9	1 7 8 9	6 7 8 9	1 3 4 5 7	1 3 4 5 7	2 3 4 5 7	3 5 6 8	2 3 5 6 8	3 5 6 8	4 5 6 8	2 5 6 8	1 5 6 8	1 5 6 8	3 2 3 4 5 6
1 2 4 5 6 7	3 4 5 6 7	7 4 5 6 7	1 8	3 8	1 2 8	1 4 5 6 8	1 4 5 6 8	1 2 3 4 5 6 7	1 3 4 5 6 7	8 4 5 6 7	7 1 5 6 7	5 6 3 5 6 7	9 2 5 6 7	2 5 6 7	2 5 6 7	4 5 6 7

5 6 7	5 6 7	1 7	5 6 7 9	3 7	8 7 9	2 5 6 7 9	2 4 5 7 9	2 4 6 7 9	2 3 7 8 9	6 7 8 9	4 7 8 9	1 2 7 8 9	5 7 8	1 2 5 8	1 2 3 5	1 2 3 5
4 5 6 7 8	9 5 6 7 8	2 5 6 7 8	1 7	1 5 6	5 6 7 8	3 7 8 9	3 7 8	6 7 8	1 7 8 9	5 7 8 9	6 7 8 9	1 7 8 9	2 6 8	3 4 6	2 4 6	7 4 6
3 5 6 8	2 5 6 8	3 5 6 8	5 6 7 9	4 7 9	4 5 6 9	1 7 8 9	5 7 8 9	6 7 8 9	4 7 8 9	5 7 8 9	6 7 8 9	7 7 8 9	8 7 8	1 2 4 6	1 2 3 4 6	1 2 3 4 6
3 5 6 8 9	3 5 6 8 9	2 5 6 8 9	1 7 8 9	5 9	4 7 8 9	3 7 8 9	4 7 8 9	6 7 8 9	1 7 8 9	5 7 8 9	6 7 8 9	7 7 8 9	8 7 8	3 4 5 6	1 2 4 5 6	1 2 4 5 6
3 6 4 8 9	3 6 4 8 9	3 6 4 8 9	2 7 8 9	2 3 7 8 9	2 3 7 8 9	1 7 8 9	5 7 8 9	6 7 8 9	1 7 8 9	5 7 8 9	6 7 8 9	7 7 8 9	8 7 8	1 4 5 6	2 4 5 6	3 4 5 6
1 3 5 4 8 9	1 3 5 4 8 9	1 3 5 4 8 9	6 7 8 9	2 3 4 5	2 3 4 5	2 3 4 5	2 3 4 5	2 3 4 5	2 3 4 5	2 3 4 5	2 3 4 5	2 3 4 5	2 3 4 5	1 2 4 5 6	1 2 3 4 5 6	1 2 3 4 5 6
7 1 2 5 6	8 1 2 5 6	9 1 2 5 6	4 1 2 5 6	5 1 2 5 6	6 1 2 5 6	1 2 5 6	2 3 5 6	2 3 5 6	2 3 5 6	2 3 5 6	2 3 5 6	2 3 5 6	2 3 5 6	1 2 4 5 6	1 2 3 4 5 6	1 2 3 4 5 6
1 2 3 5 6	1 3 5 6	3 5 6	6 8 9	7 8 9	7 8 9	2 5 6 9	2 4 5 9	1 2 4 5 9	1 2 4 5 9	1 2 4 5 9	1 2 4 5 9	1 2 4 5 9	1 2 4 5 9	1 2 4 5 9	1 2 4 5 9	3 4 5

至此，我们就把所有网的内容介绍完毕了。可以发现，很多技巧的直观视角都是依赖于网结构的，而在之前的内容并没有说明，把它放到最后一个技巧来进行介绍并不为过。

第六篇章 无逻辑技巧和计算机算法

从此处开始，我们将开始阐述最后一个数独技巧的理论部分：**没办法的办法**（**Last Resort**）。它们一般是在学习的技巧都无法完成解题后，产生的一大数独技巧。它们一般毫无逻辑可言，或者说逻辑层面来讲，不适合人脑的思维和观察，诸如后续会介绍到的**回溯法**（**Backtracking for Sudoku**）。不过，这一类题目在比赛和其他的很多时候，使用起来可能比起相对于人脑可接受的逻辑技巧而言要好使一些，所以也受到竞赛流派的玩家喜爱。

当然了，这类技巧的使用与否，取决于你的目的。如果你只是为了完成题目呢，就不用过于在意技巧；而如果你想享受做题的过程，请尽量使用逻辑层面的技巧来解题。

Part 1 无逻辑人工技巧（Mankind Guessing）

1-1 试数（Bowman's Bingo）

我们要阐述一种无逻辑技巧：**试数法**（**Bowman's Bingo**）。它的使用比较轻松。

1-1-1 使用逻辑

1-1-1-1 类似于链的基础使用方式

8	²	5	6	7	²	3	4	1
4	6	1	^{2 3}	⁵	^{2 3}	²	7	²
^{2 3}	^{2 3}	7	4	1	²	6	²	²
²	^{1 2}	²	7	^{4 5}	^{1 2}	^{1 2}	3	6
^{2 3}	^{1 2}	6	²	9	^{1 2}	7	^{1 2}	4
7	4	²	^{2 3}	^{5 6}	^{1 2 3}	^{1 2}	9	⁵
6	²	4	¹	3	7	9	^{1 2}	²
²	7	²	¹	^{4 6}	^{4 6}	^{1 2}	5	3
1	⁵	3	⁵	2	8	4	6	7

如图所示，我们随意设定一点的填数为真，比如这里设定 $r4c2 = 2$ 为真，然后顺次可以发现 $r6c3 = 8$ 、 $r6c9 = 5$ 、然后使用 $r6c9 = 5$ 对 $r4$ 作排除，可以得到 $r4c6 = 5$ ，然后对 $r5$ 作排除，得到 $r5c1 = 5$ ，于是 $r5$ 内只有 $r5c2$ 可以填 3，于是 $r5c2 = 3$ 。

随机发现 $c2$ ，发现 $c2$ 内 1 没有填数位置了。所以矛盾，因此最开始的假设是错误的，即应当是 $r4c2 \neq 2$ 。

8		5	6	7		3	4	1
4	6	1					7	
		7	4	1	5	6		
	2		7	4	5	1	3	6
5	3	6		9		7		4
7	4	8				9	5	
6		4		3	7	9		
	7					5	3	
1		3		2	8	4	6	7

这一幅图，是刚才的试数的填数显示过程，你可以仔细看一下。

我们再来看一则示例。

1-1-1-2 带有技巧结构的试数推导过程

1	2	4	3	5	6	7	8	9
1	3	1	3	1	3	1	3	1
1	3	1	3	1	3	1	3	1
1	3	1	3	1	3	1	3	1
1	3	1	3	1	3	1	3	1
1	3	1	3	1	3	1	3	1
1	3	1	3	1	3	1	3	1
1	3	1	3	1	3	1	3	1
1	3	1	3	1	3	1	3	1

如图所示，我们尝试假设 $r1c8 = 5$ ，就会顺着箭头发现 3、4、9 的三数组结构，由于 $r1c8 = 5$ 和三数组的关系， $r3c8$ 不能再放下 5 和 3、4，所以 $r3c8$ 只能填入 8。

1	6	2	5 6	4	6	3	1	5	1	5 6
8	3	1	3	7	9	5	2	7	9	7 8 9
1	6	4	6	7	9	5	2	1	3	1 3
8								4		6
3		9	4 5 6	3	1	7	2 3	5	2 3	4 5 6
6			8	6	4	7	9	8	4	5 6
5	1	3	1	3	6			3	2	3
7								9		
4		6		3	6	9			3	7
7	9	8	2	3	4	7	5	6	4	1
1	3	1	3	1	3	7	9	1	3	2
8	6	4	5	6	8			5		5 8
2	1	5	6	1	6	2	5	3	1	5
7	9	7	7	8	9	8	3	1	5	7 8 9
2	1	3	1	3	2	2	4	1	3	6
7	9	7	5	8	9	8	8	7	9	5 8 9

接着我们继续向下试数。得到 $r3c8 = 8$ 后，我们会按照箭头依次得到 $r3c4 = 6$ 、 $r4c4 = 2$ ，于是 $r4c89$ 构成 3、4 的显性数对。

此时配合最初的三数组就可以发现，此时四个单元格只有 3、4、9 三种候选数，显然是不够填的，所以出现矛盾，故原本的假设错误，即 $r1c8 \neq 5$ 。

1-1-1-3 配合 UR 的试数逻辑：试数 UR

下面我们来看看试数的 UR 结构。

2	5 6	3	4 5 6	1	3	1	2	1 2	5	7
8				4	8	9	4	5	8	7
5	8 9	4	5	1	5	2	7	6	3	4
7		1	6	4	8	4	9	5	4	5
2	5 6	8	5 6	7	1	4	1	4	1 2	5 6
9										
4	1 2		3	8	6	1 2 3	7	1 2	5	9
2 3	1 2	6	7	2 3	1	4	3	5	4	6
9										
5 6	4 5 6	4 5 6		3	1	3	8	1	3	1
1	7	2	2 3	9	7	9	5	6	4	8 9
3										
8	9	8	4	1	1 2	5	1	6	1	6
				7			7	8	8	

如左图所示，我们假设 $r3c9 = 5$ ，则 $r3c8 = 2$ ，于是根据两个填数结论共同得到 $r3c2 = 4$ ，于是 $r2c2 = 5$ 。此时观察 $b2$ ，发现 $b2$ 此时只有 $r1c5$ 可以放 4，所以 $r1c5 = 4$ ，于是根据 $r1c5$ 和 $r3c2$ 都填入的 4 得到 $b3$ 的 4 的位置只能在 $r2c9$ 。你也可以结合右图的推导过程进行分析。

但发现，由于假设 $r3c9 = 5$ ，得到了 $r23c29$ 构成了关于 4 和 5 的 UR 的致命形式，所以假设错误，故 $r3c9 \neq 5$ 。

这一则示例是配合了试数和直推的逻辑进行 UR 的论证，所以称为**试数 UR**。

1-1-2 试数转链

当然了，有一部分简单的试数法，可以修改为链写法。

8	²	5	6	7	²	3	4	1
4	6	1	^{2 3}	⁵	^{2 3}	²	7	²
^{2 3}	^{2 3}	7	4	1	²	6	²	²
²	^{1 2}	²	7	^{4 5}	^{1 2}	3	6	²
^{2 3}	^{1 2 3}	6	9	^{1 2}	7	^{1 2}	4	²
7	4	²	^{2 3}	^{5 6}	^{1 2 3}	9	^{1 2}	²
6	²	4	¹	3	7	9	^{1 2}	²
²	7	²	¹	^{4 6}	^{4 6}	^{1 2}	5	3
1	⁵	3	⁵	2	8	4	6	7

如图所示，这就是刚才的试数法的链写法。可以看到的是，实际上它被转换为了一个不连续环。不过第二则示例想要改成试数，就有点棘手了：因为假设 $r1c8 = 5$ 的时候，需要借用这个假设才能得到 $r3c8$ 的填数结论，相当于多出了一个分支，但分支不长。如果画成动态链，会有些“小题大做”。

1-1-3 链和试数的异同

很多人在看了链这种技巧之后，发现它越发是试数，于是就有读者会考虑，链是不是就是一种特殊的试数过程呢？其实不然。

我们回顾一下试数的过程：试数是随机找点，假设为真，然后一格一格向单元格内推导填数，最终发现类似于“同一区域内有相同的数字出现”、“同一区域内没有某个数字可以填入”、“某一个单元格没有数字可填”、“某一个单元格同时填入两个数”之类的矛盾，从而推翻原定假设的过程。

我们再回顾一下链的过程：链也是随机找点，假设为假，然后按照真假性依次推理（注意，链的中途是可以插入其他技巧的，相邻的节点可以同真假，并且是以候选数为单位），最终得到某个相同的数为真，或者是同一区域下的另外一个数为真，从而删除头尾的交集的过程。

它们相同点在于，试数和链都使用真假来推导，而不同的是，试数可以发散寻找。也就是说，试数在某个情况下，不仅只按照“一条路试到尾”的过程，还可以按照两个甚至多个方向试数，在方向上灵活性很强；而链虽然有动态链形式，但链最初的结构，仅仅只是为了一个方向上进行假设和推导的。但是，由于链的节点是以候选数为单位的，所以推导的时候必然可以非常灵活。任意两个节点可以同真、同假或是一真一假。

试数的过程可以任意进行扩散，但如果题目稍难，试数就变得非常困难和复杂。而且，目前在出题层面上，有一种**消去后门机制（Backdoor Deletion）**，可以去掉里面称为“后门”的东西，防止题目在找到这样的“后门”后，使得题目大大降低难度。而链就很好地避免了这一点。链是候选数层面的推导，只要避免“后门”出现在某种技巧的删数结论下，就完全可以避免试数成功。

之所以产生动态链，是因为在链的推导过程之中，一个方向上的推导已经无法完整得到删数了，此时将借助两个、甚至多个方向上的情况来导致矛盾。所以，强制链、动态链都不是试数。但是，试数和链是可以等价转化的。这也就是说，链可以转化为试数过程；而试数，也可以转化为链的写法。可这并不代表它们就是一个东西，推导层面、方式和证明手段的不同就决定了两种东西的本质区别。

链非常容易观察。因为链的强弱关系决定了两个数字之间具有什么样的位置关系和填数关系，毕竟链是一条路到头的推导。

最后，在计算机编程过程之中，这样的试数过程会被简化为从 **r1c1** 开始按从左到右、从上到下的顺序依次试数，直到推导出矛盾之后，原假设为假。这样的过程称之为**回溯法（Backtracking）**，有些地方也被称为**阿里阿德涅之线（Ariadne's Thread）**。我们之后会介绍到。

后门分两种，一种是**魔术格（Magic Cell）**，指的是数独盘面里，如果得到某单元格的填数，就能把一道难题的难度大幅度降低，那么这个单元格就叫做魔术格；另外一种，则是针对候选数层面的“魔术候选数”，不过这个说法类比于魔术格可以理解：如果某个候选数的真假结论被得到，就能大幅度降低题目的难度的话，那么这个候选数就可以称为所谓的“魔术候选数”。不过术语里并没有这个词，只是用于类比和描述它罢了。

1-1-4 带有试数思想的链

当然了，因为试数比起链来说，要好观察一些，所以这样的思想往往还被用于链之中。

2	5 8	6	1 8	4	1 5 9	3	7 8 9
4 5 8	1	4 5 8	2 3 6	2 6 7 9	3 6 7 8 9	2 5 6 7 8 9	4 6 7 8 9
4 8	9	7	5	1 2 6	1 3 6 8	1 2 6	4 6 8
9	2 3 7	2 3	1 2 6	8	5	4	3 6 7
5 6 7 8	5 7 8	5 8	1 2 4 6	1 2 4 6 7	1 6 7	1 6 7 8 9	3 6 7 8 9
6 7 8	4	1	9	3	6 7	2 6 7 8	5
1 5 7 8	5 7 8	5 8 9	1 6 8	1 5 6 9	2	3	4 7 9
1 4 5 7 8	2 3	2 3 4 5 8 9	1 3 4 8	1 4 5 9	1 3 8 9	2 5 7 9	6 7 9
4 5	6	2 3 4 5 9	7	4 5 9	3 9	8	1

如图所示，我们不标注出链的写法。这里简单阐述一下逻辑。

我们设定链头 $r8c9 \neq 2$ ，于是可以得到 $r78c9$ 的 7、9 数对，于是得到 $r1c9$ 只能填 2，于是 $r1c4 = 1$ ；于是 $r4c4 \neq 1$ 后，发现 $r4$ 出现 2、3、6、7 的四数组结构。所以 $r4c8 = 1$ 。

而刚才试数的过程之中，我们得到了 $r1c9 = 8$ ，并且由于 $r4c8 = 1$ ，所以 $r3c8$ 不能填 1、不能填 8、只能填 2，所以我们就得到了 $r8c9 \neq 2$ 时， $r3c8 = 2$ ，于是形成思路链条，删掉它们两格的交集的 2，图中即 $r23c9$ 和 $r9c8$ 的 2，都是可以删掉的。

这样的思维，基本的逻辑是链的假设，在中间插入了试数的风格，所以称为带有试数思想的链；当然了，反过来就称为带有链思想的试数法，不过我们就不作出介绍了。

1-2 图案叠加删数 (Pattern Overlay Method)

今天我们来讲一下，一种同数的无逻辑技巧。

4	9	3	6	5	1	8	2	7
6	2	1	2	7	4	5	3	9
2	7	2	2		3	6	4	1
5	3	7	1	1		2	6	4
1	6	4	7	3	2	9	8	5
9	2	2	5	4	6	7	1	3
2	2	2	3	1		1	7	2
5	4	5	5	6		4		6
8	8	8	8	8		8		8
7	1	6	4	2	5	3	9	8
3	2	9	1	1	7	1	5	2
4	8		8	8		4		6

如图所示，我们发现盘面内的数字 8 的填数位置均使用绿色标注。随后我们发现，填入 8 的情况只有如下 7 种情况：

a	b	c	d
e	f	g	

随后，我们可以清楚地发现，数字 8 一定不出现在 r4c14 和 r6c4 里。既然所有情况都不涉及这三格，所以这三格一定不会是 8，可以安全删除这些数字。如图所示。

4	9	3	6	5	1	8	2	7
6	² 8	1	² 8	7	4	5	3	9
² 5 8	7	² 5 8	² 8 9		3	6	4	1
⁵ 8	3	7	¹ 5 8 9	¹ 8 9		2	6	4
1	6	4	7	3	2	9	8	5
9	² 5 8	² 5 8	⁵ 8	4	6	7	1	3
² 5 8	² 4 5 8	² 5 8	3	¹ 6 8 9		¹ 4	7	² 6
7	1	6	4	2	5	3	9	8
3	² 4 8	9	¹ 8	¹ 6 8	7	¹ 4	5	² 6

这种技巧称为**图案叠加删数**（Pattern Overlay Method/Template，简称 **POM**）。这一点也就是 Hodoku 软件里给定的暴力技巧：**Template Delete** 和 **Template Set**。其中后者则是这个技巧的出数版本。

另外，这种技巧全部都能转换为鱼结构的使用。一般针对于一些同数情况下稍显简单的数字的排列情况，这种情况下的数据就不会有很多的组合情况，所以这种技巧在这样的情况下使用起来反而会快速一些。

Part 2 计算机数独算法简介 (Brute Force)

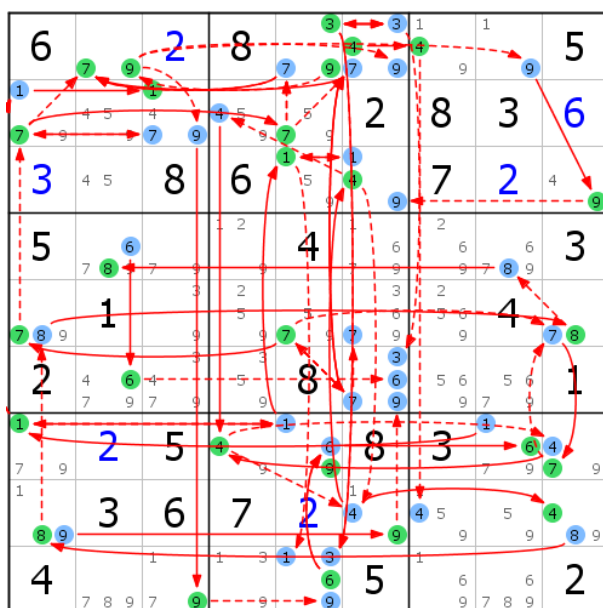
今天我们来介绍一些基本的计算机编程算法。它对于计算机的解题来说也是相当重要，但由于它的算法也显得很暴力，所以这一类算法也被数独界称为 **Brute Force**。我们来看一下。

不过教程为了面向广大读者，我们不会详细给出每一种思路的算法的源代码。

2-1 特雷伯尔表 (Forcing Net)

这一种方法名字比较多，考虑到直译“强制网”会和网技巧名称产生冲突，所以没有使用这个翻译模式，它的英文名有 **Forcing Net**、**Trebor's Tables** 和 **Tabling** 三种，都是等价的意思，而在编程里，它更靠近表格存储的运算，所以这里我们采用的是表格的名称的直译，即**特雷伯尔表**。你可以把这个技巧被理解为动态强制链组，即在强制链组内包含动态分支的现象，甚至在分支上还带有分支的情况。

这一类的算法通常复杂度都相当大。如图所示。



这样的算法包含两种计算思维：**矛盾型** (Contradiction) 和**验证型** (Verity)。由于显得比较暴力，所以这一点这里提一下即可。另外，它的源代码可以参看如下两个链接：

- [TablingSolver.java](#)
- [TableEntry.java](#)

如果教程被打印下来，此时上述的两个超链接就无法查看了，此时请参考如下的在 GitHub 上的 Hodoku 代码的仓库：

<https://github.com/Sunnie-Shine/Hodoku>,

并找到上述两个文件即可。

2-2 回溯 (Backtracking for Sudoku)

回溯是一种比较死板的思维。它尝试为 `r1c1` 的第一个候选数 (`r1c1(1)`) 假设为真, 然后尝试以这个填法来尝试 `r1c1` 所在的行列宫是否存在重复。如果重复了, 就说明了数字填错了, 我们就可以反推出 `r1c1(1)` 是错误的; 然后我们再次尝试 `r1c1 = 2`, 然后依然看 `r1c1` 的所在行列宫里是否存在重复。如果不重复的话, 我们就继续假设 `r1c2` 的填数 (从 1 开始)。

以这样的形式, 直至得到整个盘面, 所以这个算法是显得很暴力的, 但它作为数独的基本算法之一, 需要编程学习的朋友们记住和灵活使用, 我们将代码列到了本教程的附录之中。

2-3 舞蹈链 (Dancing Links for Sudoku)

舞蹈链 (Dancing Links for Sudoku, 简称 **DLX**, 取 Dancing 的 D、Links 的 L 和 [ks] 组合的发音 X), 是一种算法, 同时也可以看作是一种数据结构, 因为它在计算过程之中, 使用到的是一种特殊的链表结构——**双向循环十字链表** (**Bidirectional Orthogonal Linked List**)。它为普通链表的使用增加了更加有趣的使用: 双向和十字。它不仅仅可以完美解决数独的“行列宫不重复”的情况, 还能解决类似于这样的其他类型的谜题, 例如**八皇后问题** (**Eight-queen Puzzle**) 等。不过八皇后问题依然可以通过回溯得到解。不过, 这个算法的实现原理比较复杂, 和之前的 **Forcing Net** 一样, 我们将不在此作出探讨, 不过我们会在附录里列出舞蹈链解决数独问题的算法代码 (C#语言版本, 熟悉 Java 的可以类比, 并给出 Java 和 C#不同的地方, 方便你进行转换)。

2-4 Linq 算法 (Linqing for Sudoku)

在 C# 语言里, 我们拥有一种特殊的语法和函数库: Linq。Linq 是 Language Integrated Query, 即语言集成查询的缩写, 取了 L、in、q 三部分凑起来的, 读作 /lɪŋk/ (读音类似于单词 link)。

从名字上可以看出, 它实际上在编程语言的层面上实现了 SQL 那一套的查询操作, 不过是轻量级的; 而本质上, Linq 提供的是可使用 `foreach` 循环的 `IEnumerable<T>` 和 `IEnumerator` 对象的一些扩展方法, 用于遍历集合, 以完成类似于 SQL 的增、删、改、查的操作。因此在 C# 里, 我们将使用 Linq 函数库来操作程序变量的行为为 **Linq to Objects**, 即**应用于对象的 Linq**。

数独里, 我们可以使用 Linq 算法, 即采用 Linq 的查询修改功能来为一个盘面的信息进行修改和查询。它的大体思路是, 逐个查询盘面表示的信息为空的单元格, 并尝试使用单元格的所有候选数挨个替换它, 然后迭代执行的一种思维方式。

至此, 整个教程就全部结束了。后面的附录给出了教程的一些附带内容。

附录

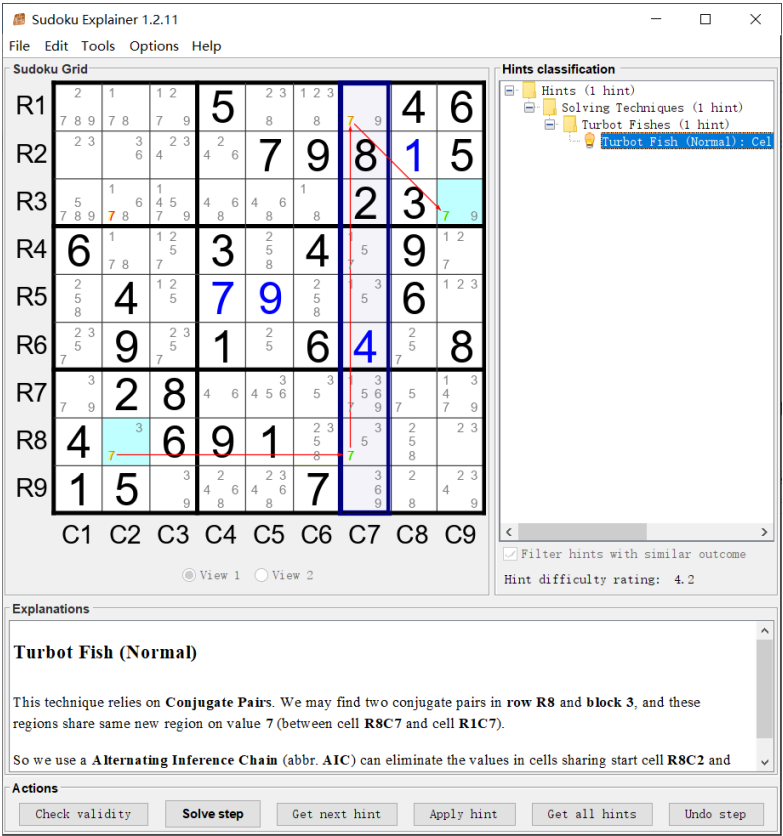
此处罗列一些比较常见，且对数独分析和做题有用处的东西。

Part 1 软件测评和使用

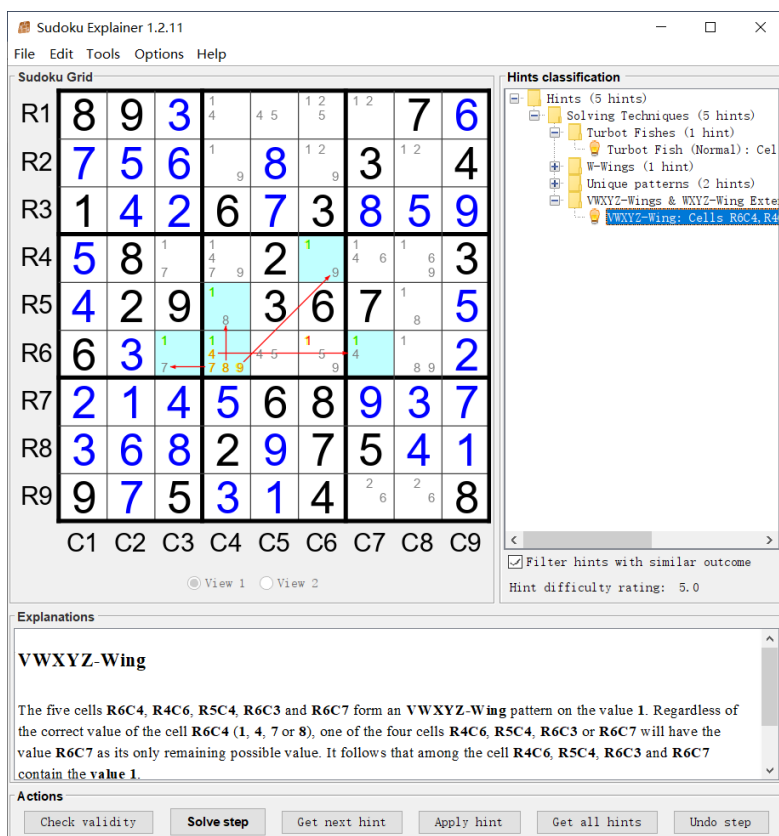
下面陈列一些比较常见的数独分析软件。

1-1 Sudoku Explainer

Sudoku Explainer 是一款用于分析数独难度系数的软件。



如图所示，界面大致像是上面这样。你可以给你发现的题目作出难度系数的分析，但需要题目是唯一解的。否则将会提示“题目不唯一解”。



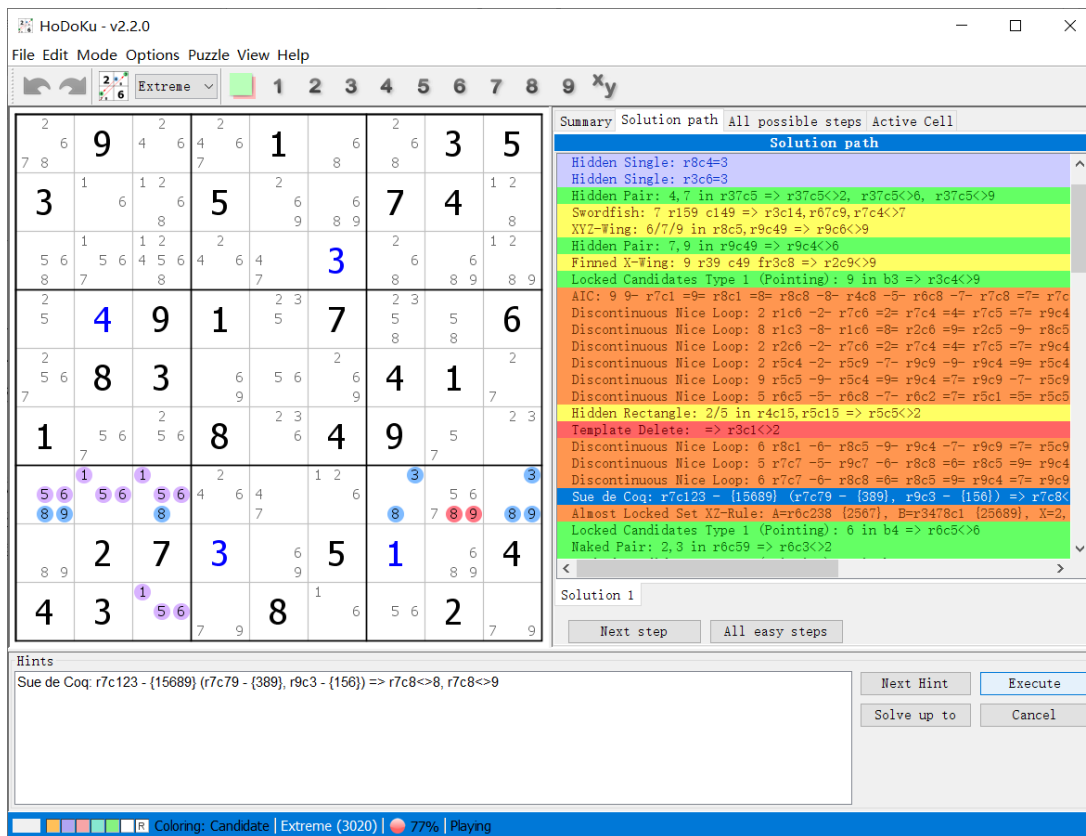
另外，我为其添加了很多技巧，例如 Wing 的增强：XYZ-Wing 的拓展：只有三个分支，但涉及四个数的 Wing，即之前说到的 WXYZ-Wing 的残缺构型；还添加了双强链技巧。

我把 Sudoku Explainer 增强了一部分的功能，并放到了 GitHub 上，它的下载地址是：<https://github.com/Sunnie-Shine/NewerSudokuExplainer>。里面的 jar 文件就是这个程序了，剩下的则是我在重新增强软件的时候，它的源代码。

遗憾的是，这个软件的官网已经不能访问了。

1-2 Hodoku

Hodoku 是另外一款分析数独题目的软件。同样地，它也需要题目是唯一解的，但提供的分析技巧比 Sudoku Explainer 要多一些。

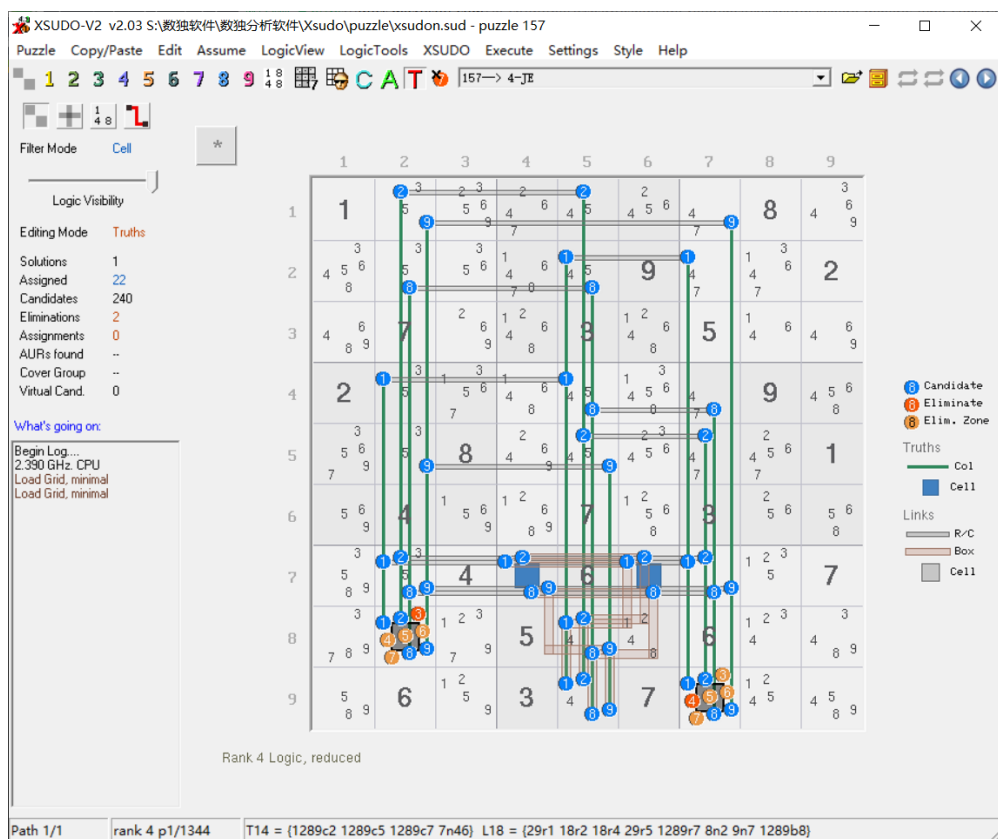


如图所示。它的安装包我依然放到了 GitHub 上，不过这个软件我并没有作出增强的处理，它的地址是 <https://github.com/Sunnie-Shine/Hodoku>。你可以下载或克隆代码，或是直接提取出 `msi` 文件，它是 Hodoku 软件的安装包。

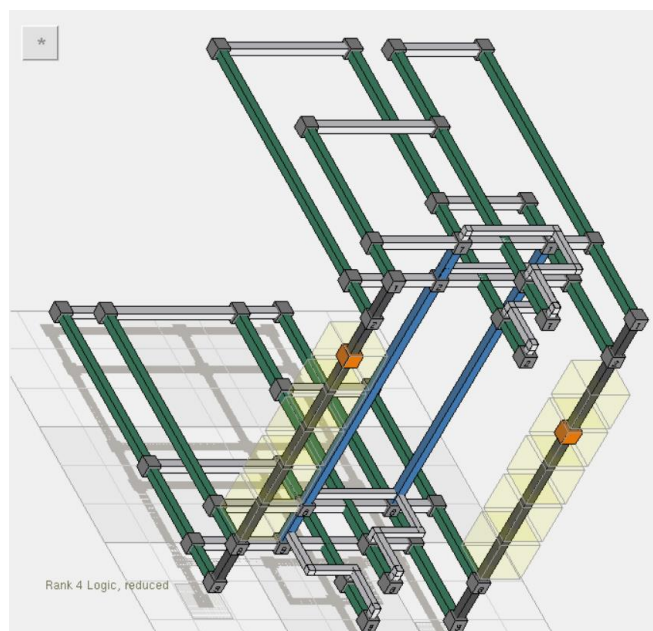
这个软件的官网是 <http://hodoku.sourceforge.net/en/>。

1-3 Xsudo

Xsudo 软件是一个界面比较经典的软件，它能够分析很多甚至 Hodoku 都做不到的技巧，诸如 SK 环、JE 等。但这些技巧并不是软件自带的，需要自定义技巧的分析过程，通过告诉软件强弱关系的方式来得到目的。



如图所示，这演示了一个 JE 技巧在 Xsудо 软件里的画法。你也可以使用软件尝试查看这个软件的三维构型。



如图所示。不过这个软件我没有得到源代码，所以请参看这个软件的官方网站并下载：
<http://sudokuone.com/index.html>。

Part 2 坐标表达

在数独里，我们具有一些约定俗成的东西，比如坐标。坐标是用于表达一个或一系列单元格的最为直接的方式，但它依然具有一些规范。

2-1 RCB 表示法

RCB 表示法是国际比较通用的方法，通过 **r**、**c**、**b** 三个字母表示行、列、宫，然后在它们各自的后面加上 1 到 9，表示第几行、第几列、第几宫。例如 **r2c3** 表示第 2 行第 3 列；当然，如果出现多个单元格，我们可以尝试简写它们。比如 **r2c1**、**r2c3** 和 **r2c6** 可以简写为 **r2c136**，因为它们同时在 **r2** 上；跟进一步地，如果两次简化后还能继续简化的，比如 **r1c12** 和 **r2c12** 同在 **c12** 上，所以我们还可以简化为 **r12c12**。

如果表示候选数，我们常见的方式是，在单元格后加上括号和一个数字，表示这个单元格的候选数几。比如 **r1c2(3)** 表示 **r1c2** 上的候选数 3；另外，同一个单元格的候选数也可以合并，例如 **r1c2(36)** 表示 **r1c2** 的候选数 3 和 6。

当然，有时候我们也会把候选数前置到最前面，去掉括号。比如 **36r1c2** 依然表示 **r1c2** 的候选数 3 和 6。

在表示某个单元格的出数或删除数结论的时候，我们分别使用 “=” 和 “<>” 表示出数或删除数。删数除了使用 “<>” 外，还可以使用 “!=” 表示。比如 **r1c2 <> 3** 表示 **r1c2** 需要删除候选数 3，**r2c8 != 29** 表示 **r2c8** 需要删除 2 和 9，而 **r1c1 = 1** 表示 **r1c1** 填入数字 1。如果使用单元格表达的话，我们可以将其简写为 **3r1c2**、**!29r2c8** 和 **1r1c1**。但这种表达有时候会产生二义性，例如 **3r1c2** 就具有二义性：它既可以表示 **r1c2 = 3**，也可以表示 **r1c2(3)**，所以在一定环境下才能使用。

2-2 K9 表示法

K9 表示法将一些常见的坐标继续简化了，这也是中国比较常见的表达方式。

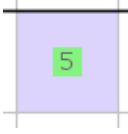
我们使用字母 **A** 到 **I** 分别表示第 1 到 9 行（有些地方因为避免 **I** 和 1 写法相近而把 **I** 换为 **J**），用数字 1 到 9 表示 1 到 9 列。比如 **A3** 表示第 1 行第 3 列（即 **r1c3**），同样，它的简写模式和上述 RCB 表示法很类似：**AC3** 表示 **A** 行和 **C** 行（即第 1 行和第 3 行）的第 3 列。

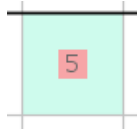

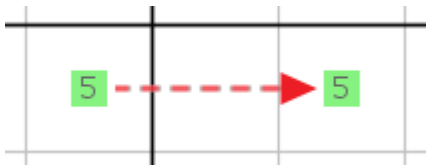

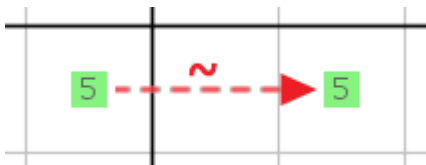
如果涉及候选数，和 RCB 表示法不同的是，它只能把候选数放到后面，并且用括号括起来。比如 **A3(4)** 等价于 RCB 表示法的 **r1c3(4)** 或者 **4r1c3**。

Part 3 教程使用符号和图形约定

此处罗列出教程里使用到的所有符号，以及图片里约定好的图形。

符号	意思	示例
{单元格}	表示一系列单元格	{r1c23, r2c2}
(候选数数值)	表示一个或一系列候选数	r2c3(6)
单元格 = 数值	表示单元格能填入什么数	r1c2 = 3
!= 或 <>	删除什么候选数	r1c2 <> 3 或 r1c2 != 3
=> 结论	表示可得到什么结论	... => r1c2 <> 5
候选数 1 => 候选数 2	从候选数 1 为真，能得到 候选数 2 为真	r1c2(3) => r2c3(5)
定义域\删除域	表示鱼内的定义域和删除 域写法	rrc\cbb
*	表示乘号	1 * 2
/	表示除号	16 / 4

图形	解释
	表示一个删数的涂色。
	表示定义域的涂色。

	表示删除域的涂色。
	表示一个强关系（即从左侧候选数为假，能得到右侧候选数为真）。
	表示一个弱关系（即从左侧候选数为真，能得到右侧候选数为假）。
	表示一个从左侧候选数为真，得到右侧候选数为真的过程。
	表示一个从左侧候选数为假，得到右侧候选数为假的过程。

Part 4 算法代码

下面列举在前文提到或使用到的一些算法源代码。

4-1 回溯法算法代码

由于回溯是相对比较简单的算法，所以我们使用 C 语言进行描述。它的代码如下。

```

/* 导入头文件 */
#include <stdio.h>
#include <stdlib.h>

/* 函数声明 */
// 显示盘面。
int show_grid(int grid[9][9]);

```

```

// 查看盘面所在的行列处的相关格是否重复（是否违反数独规则）。
int is_valid(int row, int col, int grid[9][9]);
// 回溯递归执行算法。
void backtracking(int n);

/* 测试数据 */
int test_grid[9][9] =
{
    { 4, 2, 5, 0, 0, 8, 3, 6, 7 },
    { 0, 6, 0, 0, 0, 4, 1, 2, 0 },
    { 0, 8, 0, 0, 0, 2, 0, 0, 0 },
    { 0, 4, 0, 0, 0, 5, 9, 1, 2 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 1, 9, 4, 0, 0, 6, 7, 5 },
    { 0, 3, 0, 0, 0, 0, 0, 8, 0 },
    { 0, 0, 6, 0, 0, 0, 0, 9, 0 },
    { 1, 9, 4, 0, 0, 0, 0, 5, 0 }
};

/* 函数部分 */
// 主函数。
int main(void)
{
    show_grid(testGrid);
    backtracking(0);

    system("pause");
    return 0;
}

// 显示盘面。
int show_grid(int grid[9][9])
{
    int i, j;
    for (i = 0; i < 9; i++)
    {
        for (j = 0; j < 9; j++)
            printf("%d ", grid[i][j]);
        printf("\n");
    }
    printf("*****\n");
}

// 验证盘面是否唯一解。

```

```

int is_valid(int row, int col, int grid[9][9])
{
    int number = grid[row][col];
    int key[9] = { 0, 0, 0, 3, 3, 3, 6, 6, 6 };
    int i, j;

    for (i = 0; i < 9; i++)
    {
        if (i != row && grid[i][col] == number ||
            i != col && grid[row][i] == number)
            return 0;
    }

    for (i = key[row]; i < key[row] + 3; i++)
        for (j = key[col]; j < key[col] + 3; j++)
            if ((i != row || j != col) && grid[i][j] == number)
                return 0;
    return 1;
}

// 回溯法解题。
void backtracking(int n)
{
    int i;
    if (n == 81)
        show_grid(test_grid);
    else
    {
        int row = n / 9, col = n % 9;

        if (test_grid[row][col] != 0)
        {
            backtracking(n + 1);
            return;
        }

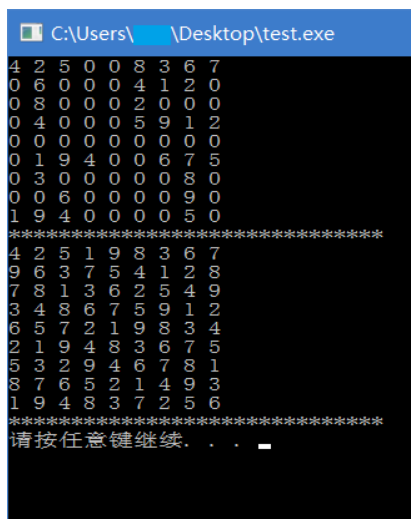
        for (i = 0; i < 9; i++)
        {
            test_grid[row][col]++;
            if (is_valid(row, col, test_grid))
                backtracking(n + 1);
        }

        test_grid[row][col] = 0;
    }
}

```

```
}  
}
```

测试的结果如下。



可以从思路发现，回溯法无法验证题目的唯一性，但它可以验证题目是否有解。如果题目无解，则没有任何的结论的输出；如果有解，它就会尝试输出一个解，并退出程序（不过你可以考虑更改“n == 81”条件内执行的语句来更改得到解时候的执行逻辑。

不过由于 C 语言不支持异常，所以我们无法控制输出固定个数解的操作。不过在其它的高级编程语言里，我们可以采用异常机制，当输出的解的总数超出一定限度的时候，让该递归函数抛出递归跳出的异常（例如 Java 的 [InterruptedException](#)）来跳出执行。

4-2 舞蹈链算法代码

这里列出舞蹈链算法的源代码，为了思路更为清晰，此处将使用 C# 语言描述。

4-2-1 数据节点文件（DataNode.cs）

```
// 引用类型后缀符号 '?' 表示此处的变量可能会传入空值 null。  
public class DataNode  
{  
    public DataNode(int id, ColumnNode? column)  
    {  
        (Id, Column, Left, Right, Up, Down)  
        = (id, column, this, this, this, this);  
    }  
  
    public int Id { get; set; }  
    public ColumnNode? Column { get; set; }  
    public DataNode Left { get; set; }  
    public DataNode Right { get; set; }
```

```

    public DataNode Up { get; set; }
    public DataNode Down { get; set; }
}

```

4-2-2 列节点文件 (ColumnNode.cs)

```

// sealed class 相当于 Java 的 final class。
// 冒号是派生自什么类或接口，相当于 extends 和 implements。
// 构造函数上的冒号和 base 表示调用基类（父类、超类）构造器。
public sealed class ColumnNode : DataNode
{
    public ColumnNode(int id) : base(id, null)
    {
        Column = this;
        Size = 0;
    }

    public int Size { get; set; }
}

```

4-2-3 舞蹈链文件 (DancingLink.cs)

```

// partial 关键字表示该类还有代码分布在其它文件和其它部分里。
public partial class DancingLink
{
    public DancingLink(ColumnNode root) => Root = root;

    public ColumnNode Root { get; set; }

    public ColumnNode CreateLinkedList(int[,] grid)
    {
        // 类似于 Java 的断言 (assertion)，但这里仅仅是作出假设，
        // 以此禁用编译器对 grid 可能为 null 而直接调用时抛出异常
        // 而产生一个警告。
        // o is null 等价于 (object)o == null。
        // 由于 C# 有重载运算符的机制，所以重载后不保证能判定 null 值，
        // 进而产生异常，所以会被翻译为 o 变量强制转为 object 后调用
        // 最初的基类 object 的等号运算符。
        // 为了防止潜在的运算符重载产生的异常，请尽量使用 is 运算符。
        Contract.Assume(!(grid is null));

        var columns = new List<ColumnNode>();
        for (int columnIndex = 0; columnIndex < 324; columnIndex++)

```

```

    {
        var col = new ColumnNode(columnIndex)
        {
            Right = Root,
            Left = Root.Left
        };
        Root.Left.Right = col;
        Root.Left = col;
        columns.Add(col);
    }

    for (int x = 0; x < 9; x++)
    {
        for (int y = 0; y < 9; y++)
        {
            if (grid[x, y] == 0)
            {
                for (int d = 0; d < 9; d++)
                {
                    // 单元格为空，所以要添加所有的候选数到表里。
                    FormLinks(columns, x, y, d);
                }
            }
            else
            {
                // 当前位置有提示数，所以只需要加入这个候选数即可。
                int d = grid[x, y] - 1;
                FormLinks(columns, x, y, d);
            }
        }
    }

    return Root;
}

private void LinkRow(MatrixRow d)
{
    d.Position.Right = d.Column;
    d.Position.Left = d.Block;
    d.Column.Right = d.Row;
    d.Column.Left = d.Position;
    d.Row.Right = d.Block;
    d.Row.Left = d.Column;
    d.Block.Right = d.Position;

```

```

        d.Block.Left = d.Row;
    }

    private void LinkRowToColumn(DataNode section)
    {
        var col = section.Column;
        if (!(col is null))
        {
            col.Size++;
            section.Down = col;
            section.Up = col.Up;
            col.Up.Down = section;
            col.Up = section;
        }
    }

    private void FormLinks(
        List<ColumnNode> columns, int x, int y, int d)
    {
        var position = new DataNode(
            x * 81 + y * 9 + d, columns[x * 9 + y]);
        var row = new DataNode(
            x * 81 + y * 9 + d, columns[81 + x * 9 + d]);
        var column = new DataNode(
            x * 81 + y * 9 + d, columns[162 + y * 9 + d]);
        var block = new DataNode(
            x * 81 + y * 9 + d,
            columns[243 + (3 * (x / 3) + y / 3) * 9 + d]);
        var matrix_row = new MatrixRow(
            position, row, column, block);

        LinkRow(matrix_row);
        LinkRowToColumn(matrix_row.Position);
        LinkRowToColumn(matrix_row.Row);
        LinkRowToColumn(matrix_row.Column);
        LinkRowToColumn(matrix_row.Block);
    }
}

```

4-2-4 矩阵行节点文件 (DancingLink.MatrixRow.cs)

```

public partial class DancingLink
{
    private class MatrixRow

```

```

{
    public MatrixRow(
        DataNode position, DataNode row,
        DataNode column, DataNode block)
        => (Position, Row, Column, Block)
        = (position, row, column, block);

    public DataNode Position { get; set; }
    public DataNode Row { get; set; }
    public DataNode Column { get; set; }
    public DataNode Block { get; set; }
}
}

```

4-2-5 解题器对象 (DancingLinksSolver.cs)

```

public sealed class DancingLinksSolver : BruteForceSolver
{
    private int _solutionCount;
    // readonly 关键字表示该变量仅在初始化时候赋值 (构造器中),
    // 其它任何位置都不允许为其赋值。可类比于 final 字段。
    private readonly Stack<DataNode> _answerNodesStack
        = new Stack<DataNode>();
    private ColumnNode? _root;
    // 这个 Grid 类型在这里没有列出源代码, 不过你可以自己实现。
    // 它仅仅是对一个二维 int 类型数据的封装。
    private Grid? _resultGrid;

    public Grid? Solve(int[,] gridArray)
    {
        var dlx = new DancingLink(new ColumnNode(-1));
        _root = dlx.CreateLinkedList(gridArray);

        try
        {
            // null 包容运算符放到变量和引用类型表达式末尾,
            // 表示产生 null 值是预期操作。否则编译器将在此处产生警告,
            // 提示用户在此处产生了 null 值。
            Search();
            return _resultGrid!;
        }
        catch { return null; }
    }
}

```



```

private void Search()
{
    if (_solutionCount > 1)
        throw new MultipleSolutionsException();

    Contract.Assume(!(_root is null));

    if (_root.Right == _root)
    {
        // All columns were removed!
        _solutionCount++;
        // out 变量表示该变量在执行操作的时候，只用于赋值，
        // 可类比于 C 语言的指针来返回多个返回值时的操作。
        RecordSolution(_answerNodesStack, out _resultGrid);
    }
    else
    {
        var c = ChooseNextColumn();
        Cover(c);

        for (var r = c.Down; r != c; r = r.Down)
        {
            _answerNodesStack.Push(r);
            for (var j = r.Right; j != r; j = j.Right)
                Cover(j.Column!);
            Search();
            r = _answerNodesStack.Pop();
            c = r.Column!;

            for (var j = r.Left; j != r; j = j.Left)
                Uncover(j.Column!);
        }

        Uncover(c);
    }
}

private void Cover(DataNode column)
{
    column.Right.Left = column.Left;
    column.Left.Right = column.Right;
    for (var i = column.Down; i != column; i = i.Down)
    {
        for (var j = i.Right; j != i; j = j.Right)

```

```

        {
            j.Down.Up = j.Up;
            j.Up.Down = j.Down;
            j.Column!.Size--;
        }
    }
}

private void Uncover(DataNode column)
{
    for (var i = column.Up; i != column; i = i.Up)
    {
        for (var j = i.Left; j != i; j = j.Left)
        {
            j.Column!.Size++;
            j.Down.Up = j;
            j.Up.Down = j;
        }
    }
    column.Right.Left = column;
    column.Left.Right = column;
}

private ColumnNode ChooseNextColumn()
{
    Contract.Assume(!(_root is null));

    int size = int.MaxValue;
    var nextColumn = new ColumnNode(-1);
    var j = _root.Right.Column;
    while (j != _root)
    {
        if (j!.Size < size)
        {
            nextColumn = j;
            size = j.Size;
        }
        j = j.Right.Column;
    }
    return nextColumn;
}

private void RecordSolution(
    Stack<DataNode> answer, out Grid result)

```

```

{
    Contract.Assume(!(answer is null));

    var resultArray = new int[9, 9];
    // Linq 语句, 将语句执行的结果
    // (可用 foreach 循环的 IEnumerable<T> 对象)
    // 导入到新创建的 List<T> 对象里。
    // 注意, C# 的 int 和 Java 的 int 不同, Java 的 int 和系统名
    // Integer 不是等价的, 而 C# 里, int 和系统名 System.Int32
    // 是等价的, 所以可以直接把这个关键字放在泛型参数上。
    var idList = new List<int>(from k in answer select k.Id);
    idList.Sort();

    var gridList = new List<int>(
        from id in idList select id % 9 + 1);
    for (int i = 0, x = 0, y = 0; i < 81; i++)
    {
        resultArray[y, x++] = gridList[i];
        if ((i + 1) % 9 == 0)
        {
            y++;
            x = 0;
        }
    }

    var grid = new Grid(resultArray);
    if (grid.SimpleValidate()) // 最终验证一下是否违背数独规则。
        result = grid;
    else throw new NoSolutionException();
}
}

```

4-2-6 主方法调用 (Program.cs)

```

// static class 和 Java 意义不同, 这里的静态类指的是类里只有静态成员,
// 并非是一个访问修饰符关键字。
/// <summary>
/// 本项目的主类。
/// </summary>
internal static class Program
{
    /// <summary>
    /// 主方法, 程序的主入口点。
    /// </summary>

```

```

private static void Main()
{
    // 由于代码排版的关系，此处只能选择折行的方式把一整个字符串拆解成
    // 三部分显示，再拼接起来。
    // Parse 方法是 Grid 类里提供的字符串转 Grid 类型的方法，
    // 前文说过 Grid 是二维整型数组的封装，所以代码应当比较好写。
    var grid = Grid.Parse(
        "060000005307040900020300007" +
        "006470000000905000000012700" +
        "100009060009060403600000020");

    Console.WriteLine(
        new DancingLinksSolver().Solve(grid));
}
}

```

4-3 Linq 解法算法代码

在数独里，我们甚至可以使用 Linq 轻松完成一个题目的解题，而且比纯回溯要更快一些。我们可以尝试使用 Linq 自带的查询和替代功能，为一个数独盘面（字符串形式）进行解题。代码如下（使用 C# 8 提供的语法）。

```

/// <summary>
/// 使用 Linq 解题。
/// </summary>
/// <param name="grid">
/// 表示题目的字符串，`0` 表示空格，`1` 到 `9` 表示提示数。
/// </param>
/// <returns>
/// 返回题目的解的字符串。当题目多解或无解时，将返回 <c>null</c> 值。
/// C# 8 提供了不可空引用类型的特性，当项目被设置为引用类型不可空时，
/// 我们必须在引用类型后添加可空类型符号 `?` 表示该类型变量是可以为
/// <c>null</c> 数值的。
/// </returns>
/// <remarks>
/// <para>注意参数字符串不带有任何的换行符号 `r` 或 `n`。</para>
/// <para>
/// 使用的时候还需要注意引入 <see href="System.Linq"/>
/// 命名空间。
/// </para>
/// </remarks>
public string? Solve(string grid)
{

```

```

var leafNodesOfMoves = new[] { grid };
SolveStrings(leafNodesOfMoves); // 使用本地方法。
var result = leafNodesOfMoves.Length != 1
    ? null // 这个数组变量的长度如果不为 1, 则表示多解或无解。
    : leafNodesOfMoves[0];
return result is null ? null : result;

#region 本地函数
// 摘要:
//     使用 Linq 解题的函数。
// 参数:
//     leafNodesOfMoves: 解的存放数组。
static void SolveStrings(string[] leafNodesOfMoves)
{
    // 这个本地函数里只有一个 while 循环。
    while (leafNodesOfMoves.Length > 0
        && leafNodesOfMoves[0].IndexOf('0') != -1)
    {
        leafNodesOfMoves = (
            from partialSolution in leafNodesOfMoves
            let index = partialSolution.IndexOf('0')
            let column = index % 9
            let groupOf3 = index - index % 27
                + column - index % 3
            from searchLetter in "123456789"
            where !(
                from emptyToCheck in Values.DigitRange
                let isInRow = partialSolution[
                    index - column + emptyToCheck]
                    == searchLetter
                let isInColumn = partialSolution[
                    column + emptyToCheck * 9]
                    == searchLetter
                let isInBlock = partialSolution[
                    groupOf3 + emptyToCheck % 3
                    + (int)Math.Floor(emptyToCheck / 3f) * 9]
                    == searchLetter
                where isInRow || isInColumn || isInBlock
                select emptyToCheck).Any()
            select $"{partialSolution.Substring(0, index)}{
                searchLetter}{
                    partialSolution.Substring(index + 1)}"
        ).ToArray();
    }
}

```

```
}  
#endregion  
}
```

可以从代码里看出，实际上它就是在不断替换字符为 0（0 表示空格）的地方，然后得解的操作。

Part 5 参考文献

此处我们陈列出这份教程里参考的资料（或提供资料的网站）。

- [1] <http://sudopedia.enjoysudoku.com>
- [2] <http://www.math.ie/checker.html>
- [3] <http://forum.enjoysudoku.com>
- [4] <https://tieba.baidu.com/f?kw=数毒>
- [5] <https://zhuanlan.zhihu.com/SunnieSudoku>
- [6] <http://www.sudocue.net/glossary.php>
- [7] <http://www.sudokufans.org.cn>
- [8] <https://www.bilibili.com/video/av11911189/>

Part 6 鸣谢

感谢参与本文档编写和给予帮助的小伙伴（列举名称不分任何先后顺序）：

- 小钰（030667073250）；
- 楠竹（024627632154）；
- 念兹在兹（014657434307）；
- 探长（0153654376）；
- 辉煌背后的衰落（04032426026）；
- 浮世论℃（024461354775）；
- 北航虎（02150357116）；
- 幸福时光（02622275656）；
- 无极噬（015547033555）。

感谢大家的支持。另外，教程到此就全部结束了，以后的路还需要你们自己走。