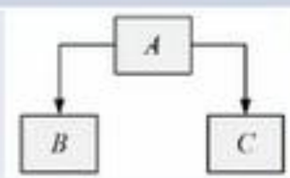
第6讲 约束问题求解

尹慧琳，yinhuilin@tongji.edu.cn
同济大学 电子与信息工程学院

第6讲 约束问题求解

- 约束满足问题（Constraint Satisfaction Problems，CSPs）
  - 问题：被定义为其状态必须满足若干约束和限制的一组对象
  - 求解：通过识别违反约束的变量/值的组合以排除不可行空间、聚焦搜索空间，提高搜索效率
  - 可行集（feasible set）：满足约束条件的所有候选解的集合

CSPs是人工智能和运筹学共同的研究课题

	标准的搜索问题	约束满足问题
状态	采用原子式表示	采用因子式表示：是一组变量及值
内部结构	无 	有 

第 6 讲 约束问题求解

第6讲 约束问题求解

6.1 约束满足问题

- 有很多智力游戏属于CSP，如：
  - 八皇后难题（Eight queens puzzle）
  - 数独（Sudoku）
  - 纵横字谜（Crosswords）
  - 算式谜（Cryptarithmetic）
  - 不等式（Futoshiki）
  - 数和（Kakuro, Cross sums）
  - 逻辑谜题（Logic puzzle）

1	3	4	5	2	6
2	5	6	1	4	3
5	1	2	6	3	4
6	4	3	2	5	1
3	6	5	4	1	2
4	2	1	3	6	5

6×6

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

9×9

	12	2	1			3				11	13	9	14		10
6	3	7		1	12					9	10	4	16	15	11
	11			9			10			3	16	13	2	1	
8	15	9		16	13	14	11	6	1	4		3	12	7	5
13			11	14	6		5	12	4		7	2	1	8	3
4	1					15	3			14	9	16	11	12	7
	14		16			12	7	11	2			10	9	13	
7			15	11		10	9	8	13	16	3	6		4	14
2		13	9	10			1	5	3			15	4		16
5		11	6	3	8	9	4		14	10	15	12	7		
12	10		3	6			2	16		7	4	1	8	5	9
		15		12	11	5	16	2				14		3	6
	13		12	5	16	1								14	4
10	5		4	2	9	8	13	3	7	6		11	15	16	12
	6		2		3	11		4	15	5	12	8			
11		3	14			4	12			8	1	5		9	2

16×16

SEND + MORE = MONEY

海上明月 × 9 = 月明海上

第6讲 约束问题求解

6.1 约束满足问题

- CSP的定义：三元组 $\langle X, D, C\rangle$
  - X: 变量的集合，$\mathrm{X}=\{X_1, \dots, X_n\}$
  - D: 值域的集合，$\mathrm{D}=\{D_1, \dots, D_n\}$，其中$D_i$是由变量$X_i$的可能取值$\{v_1, \dots, v_k\}$组成的集合。
  - C: 描述变量取值的约束，也是一个集合 $C=\{C_1, \dots, C_m\}$。
- CSP的约束（Constraint）
  - 概念：对可能世界中的变量进行赋值的条件组合
  - 构成：$C_j \in C$是一个二元组$\langle t_j, R_j\rangle$
    - 作用域t_j：$t_j \subset X$是具有 k 个变量的作用域
    - 约束关系 R_j：对应于值域 D_i 定义了作用域中 k 个变量取值应满足的关系。

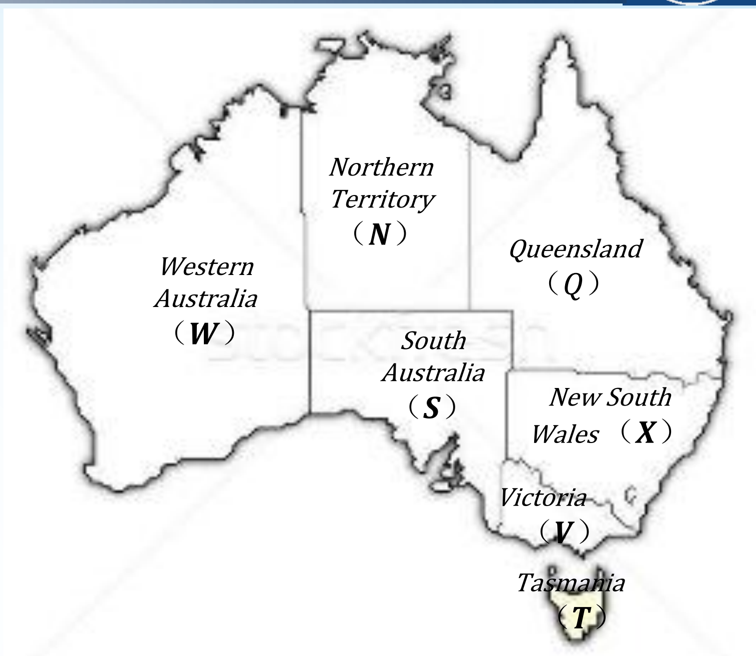
第6讲 约束问题求解

6.1 约束满足问题

- CSP举例：地图着色问题

澳大利亚境内分为7个州和行政区，彼此的位置和相邻关系如图所示。现用三种不同的颜色为各区域涂色，要求相邻的区域颜色不能相同。

- 变量：7个区域
  $X = \{W, N, Q, X, V, S, T\}$
- 值域：三种颜色
  $D_i = \{r, g, b\}$
- 约束：相邻区域颜色不相同
  $C = \{S \neq W, S \neq N, S \neq Q, S \neq X, S \neq V, W \neq N, N \neq Q, Q \neq X, X \neq V\}$



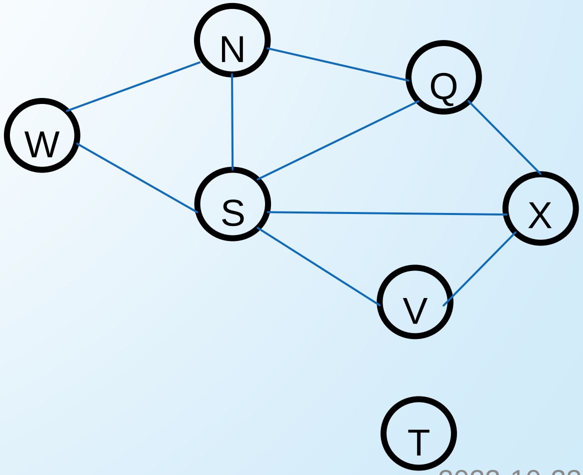
第6讲 约束问题求解

6.1 约束满足问题

- CSP举例：关于约束的表达

  CSP定义中约束的表示C_i是有序对$\langle t_j, R_j\rangle$

  - 隐式：$\langle (S,W), S \neq W\rangle$，简化为$\{S \neq W\}$
  - 显式：展开所有可能的组合有
    $\{(r,g),(r,b),(g,r),(g,b),(b,r),(b,g)\}$
  - 约束图
    - 节点：对应于问题的变量
    - 连线：表示两者间有约束

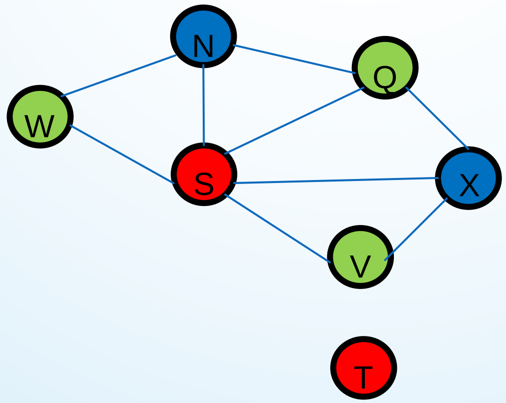


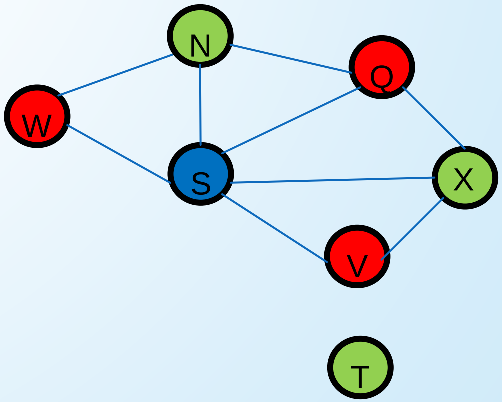
第6讲 约束问题求解

6.1 约束满足问题

- CSP的解：
  - 状态：由对部分或全部变量的一个赋值来定义，如$\{X_i = v_i, X_j = v_j, \ldots\}$
    - 相容 / 合法 / 一致 的赋值：满足所有的约束条件
    - 完整赋值：指每个变量都已赋值
  - 解：相容的、完整的赋值

可行集

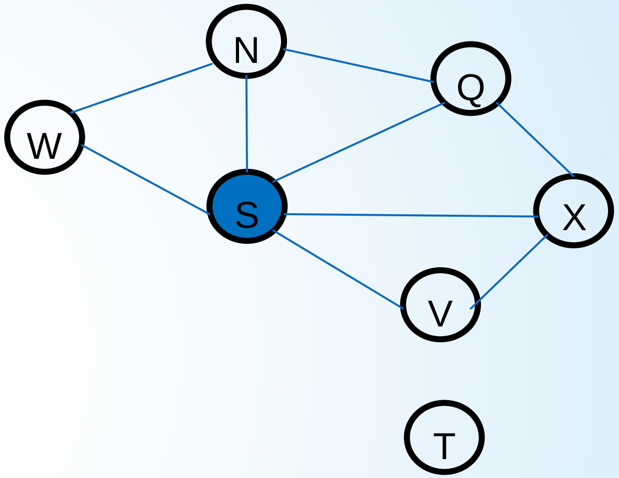




第6讲 约束问题求解

6.1 约束满足问题

- CSP举例：关于求解的分析
  - 若按标准搜索问题处理
    - 搜索空间：大小?
    - 目标状态：
  - 若按启发式搜索处理
    - 启发式函数的设计
  - 若按CSP问题处理
    - 通过识别约束是否满足，可快速消除庞大的搜索空间



第6讲 约束问题求解

6.2约束满足问题实例化

- *数独P167*

	1	2	3	4	5	6	7	8	9
A			3		2		6		
B	9			3		5			1
C			1	8		6	4		
D			8	1		2	9		
E	7								8
F			6	7		8	2		
G			2	6		9	5		
H	8			2		3			9
I			5		1		3		

$$\begin{array}{cccc} A_1, & A_2, & \cdots, & A_9 \\ B_1, & B_2, & \cdots, & B_9 \\ & \vdots & & \\ I_1, & I_2, & \cdots, & I_9 \end{array}$$

$$D_i = \{1, 2, \cdots, 9\}$$

$$\forall \neq (A_1, A_2, \cdots, A_9), \cdots, \forall \neq (I_1, I_2, \cdots, I_9)$$
$$\forall \neq (A_1, B_1, \cdots, I_1), \cdots, \forall \neq (A_9, B_9, \cdots, I_9)$$
$$\forall \neq (A_1, A_2, A_3, \cdots, C_3), \cdots, \forall \neq (G_7, G_8, G_9, \cdots, I_9)$$

第6讲 约束问题求解

6.1 约束满足问题实例化

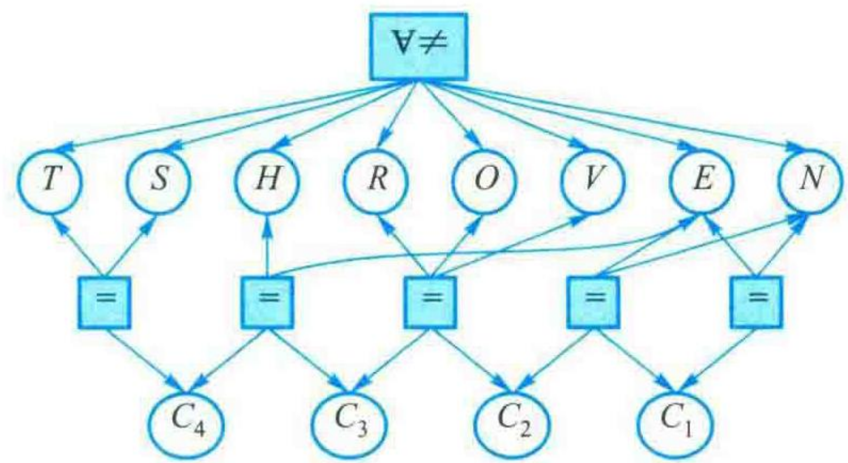
- *算式谜 P168*

$$THREE+THREE+ONE=SEVEN$$

$$X=\{T,S,H,R,O,V,E,N,C_1,C_2,C_3,C_4\} \qquad D_i=\{0,1,2,\cdots,9\}$$

$$\begin{aligned}
&\forall\neq(T,S,H,R,O,V,E,N)\\
&E+E+E=N+10\cdot C_1\\
&E+E+N+C_1=E+10\cdot C_2\\
&R+R+O+C_2=V+10\cdot C_3\\
&H+H+C_3=E+10\cdot C_4\\
&T+T+C_4=S
\end{aligned}$$

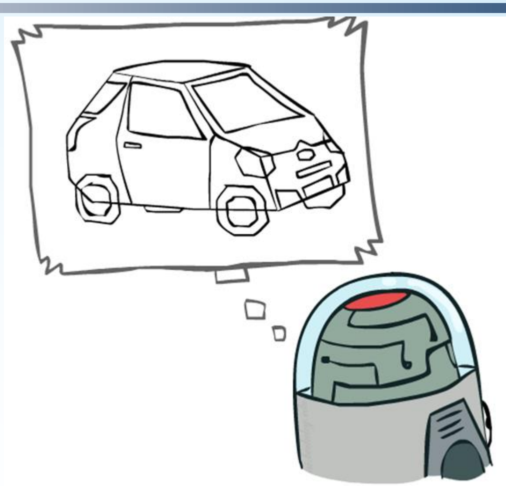
(a)



(b)

第6讲 约束问题求解

6.1 约束满足问题



- 举例：汽车组装调度
  - 变量：对应15个任务的开始时间

$$X = \{Axle_F, Axle_B,$$ ——安装两个车轴

$$Wheel_{RF}, Wheel_{LF}, Wheel_{RB}, Wheel_{LB},$$ ——安装四个车轮

$$Nuts_{RF}, Nuts_{LF}, Nuts_{RB}, Nuts_{LB},$$ ——拧紧每个车轮的螺母

$$Cap_{RF}, Cap_{LF}, Cap_{RB}, Cap_{LB},$$ ——安装四个轮毂罩

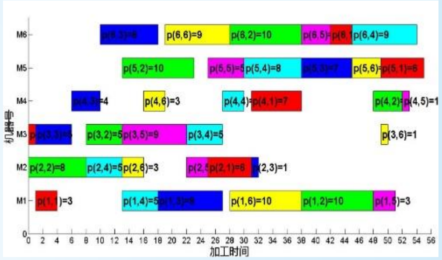
$$Inspect\}$$ ——检查最后的组装

- 约束：
  - 资源约束：只有一个车轴夹具
  - 工艺约束：工艺流程规定了每个任务的执行时间；其中Inspect任务的用时为3分钟
  - 时间约束：组装过程的总完成时间是30分钟
- 值域：Di={1,2,3,...,27}

第6讲 约束问题求解

6.1 约束满足问题

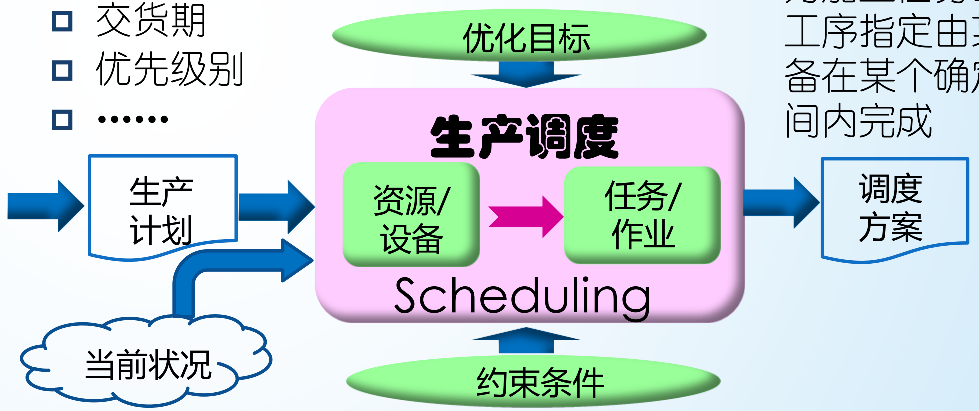
- 举例：作业车间调度
  - Single machine
  - Parallel machine
  - Flow-shop
  - Job-shop
  - Open-shop
  - FMC/FMS
  - Reentrant
  - Stochastic
  - ……



- 时间相关
- 效率相关
- 平衡相关
- ……

为加工任务的所有工序指定由某台设备在某个确定的时间内完成

- 产品序列
- 交货期
- 优先级别
- ……



- 设备状况
- 在制品状况
- 缓存区状况
- ……

- 工艺约束（加工流程）
- 时间约束
- 资源约束
- ……

第6讲 约束问题求解

6.1 约束满足问题

- CSP问题的一般化
  - 变量 Variable：表示问题中某些特征的符号
  - 值域 Domains

值域	离散 Discrete	连续 Continuous
有限 Finite	地图着色问题	*广泛存在* 如：线性规划问题
无限 Infinite	整数或字符串集合	

  - 约束 Constraints

	一元约束	二元约束	全局约束
线 性 Linear	$\langle(S), S \neq \mathrm{b}\rangle$	$S \neq W$	$Alldiff(F,T,U,W,R,O)$
非线性 Nonlinear	*尚无有效的通用求解算法*		

- CSP问题的求解
  - 推理方法：约束传播
  - 搜索方法：回溯搜索、局部搜索

第 6 讲 约束问题求解

第6讲 约束问题求解

6.2 约束传播

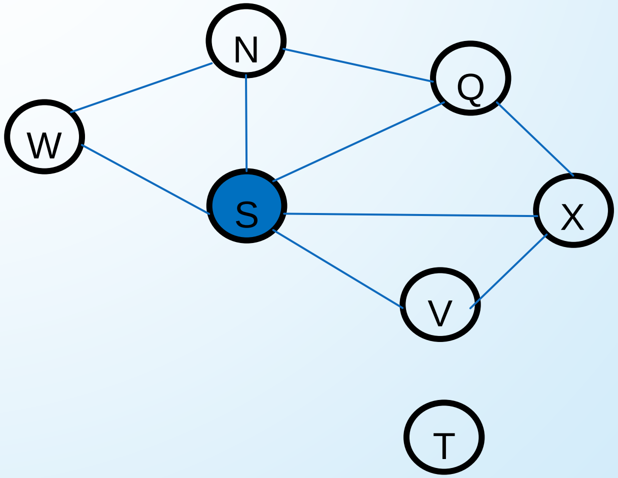
- 约束传播 Constraint Propagation

基于变量赋值的约束，对变量的合法取值范围进行限制，并影响到与此变量有约束关系的其它变量的取值，以此类推。从而缩减问题的空间。

- 核心思想：局部相容性（local consistency）

将一个问题转化成等价但更易于求解的约束问题，不断使相应的变量、值域满足局部相容性约束的相关条件。

- 局部相容性的类型
  - 节点相容
  - 弧相容
  - 路径相容
  - k-相容



第6讲 约束问题求解

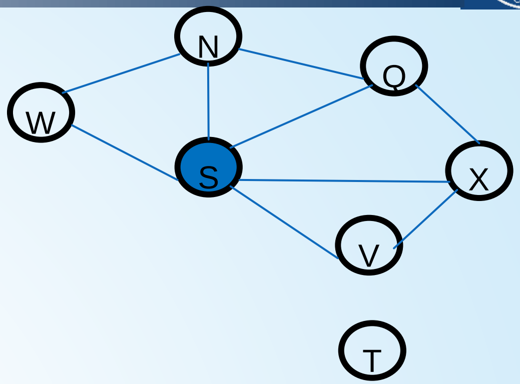
6.2 约束传播：局部相容性

- 节点相容
  - 定义：如果单个变量的值域中的所有取值满足它的一元约束，则称此变量是节点相容的。
  - 举例：若地图着色问题中，S区域的人不喜欢绿色，即$\langle (S), S \neq g \rangle$，则变量S的值域空间由原来的三元缩小为二元 {r, b}
- 弧相容
  - 定义：如果CSP中某变量值域中的所有取值满足该变量的所有二元约束，则称此变量是弧相容的。如果每个变量相对其他变量都是弧相容的，则称该网络是弧相容的。
  - 举例：约束$S \neq W$ 可以显式地展开为：

$$\langle (S, W), \{(r, g), (r, b), (g, r), (g, b), (b, r), (b, g)\} \rangle$$

第6讲 约束问题求解

6.2 约束传播：局部相容性



- 路径相容
  - 定义：指两个变量的集合$\{X_i, X_j\}$对于第三个变量X_m是相容的。即：对于$\{X_i, X_j\}$的每一个相容赋值$\{X_i = a, X_j = b\}$，X_m都有合适的取值同时使得$\{X_i, X_m\}$和$\{X_m, X_j\}$是相容的。被称为路径相容。
  - 举例：两色澳大利亚地图着色问题
    - $\{W, S\}$的相容赋值有二个：$\{W = r, S = b\}$和$\{W = b, S = r\}$
    - 分析$\{W, S\}$对N的相容路径，可知不存在。所以此题无解
- k-相容
  - 定义：对于任意k-1个变量的相容赋值，第k个变量总能被赋予一个和前k-1个变量相容的值，则这个CSP就是k相容的。
    - k=1，等同于节点相容
    - k=2，等同于弧相容
    - k=3，等同于路径相容

第 6 讲 约束问题求解

6.1 约束满足问题

6.2 约束传播

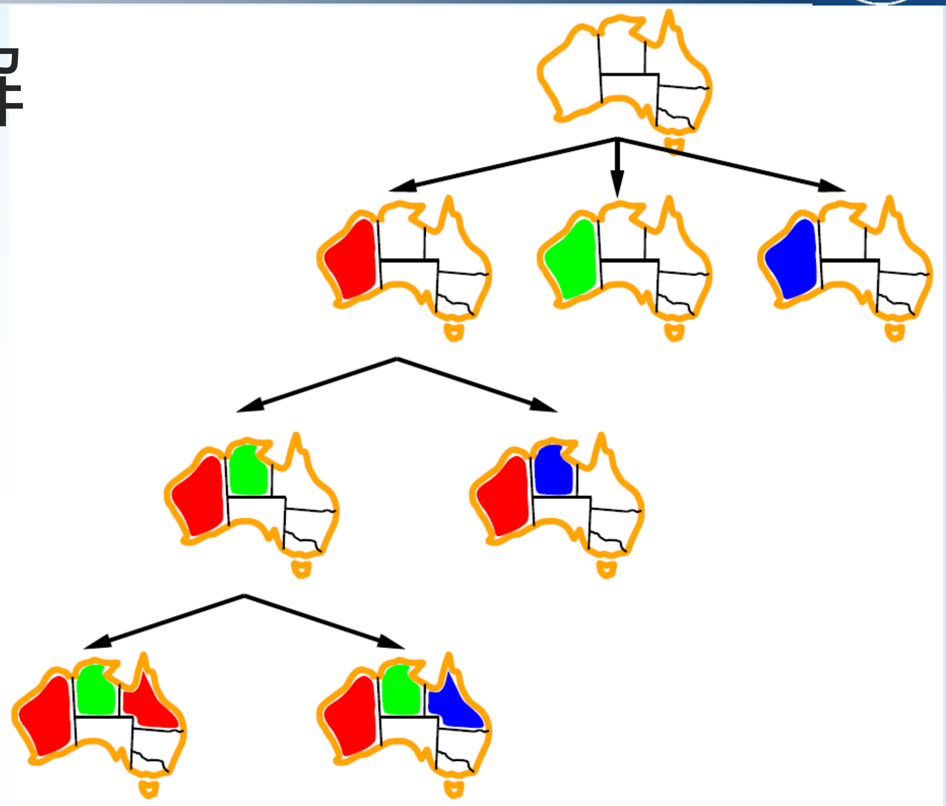
6.3 回溯搜索

6.4 局部搜索

6.5 问题的结构

第6讲 约束问题求解

6.3 回溯搜索



- 回溯搜索：

  是一种通用的深度优先搜索算法，用于查找问题空间上的解

- 回溯搜索的步骤：
  - 以深度优先方式递进地寻找候选解；
  - 一旦确定该候选不是一个合法的解，就立即将其抛弃，从原路返回；
  - 再以深度优先方式寻找下一个候选

第6讲 约束问题求解

6.3 回溯搜索

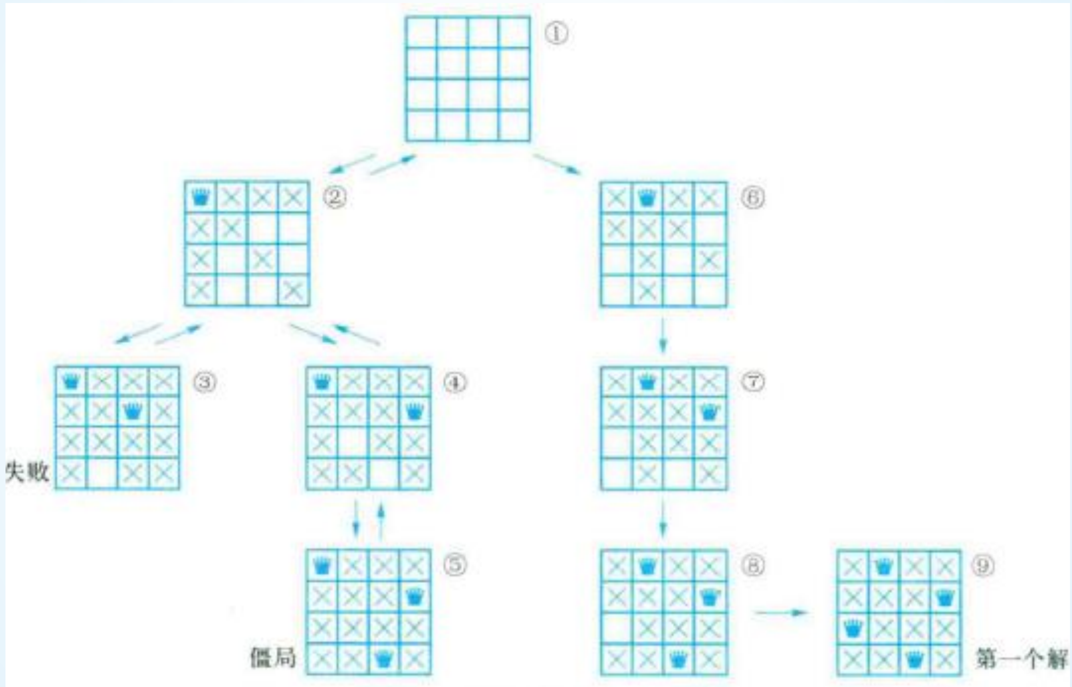
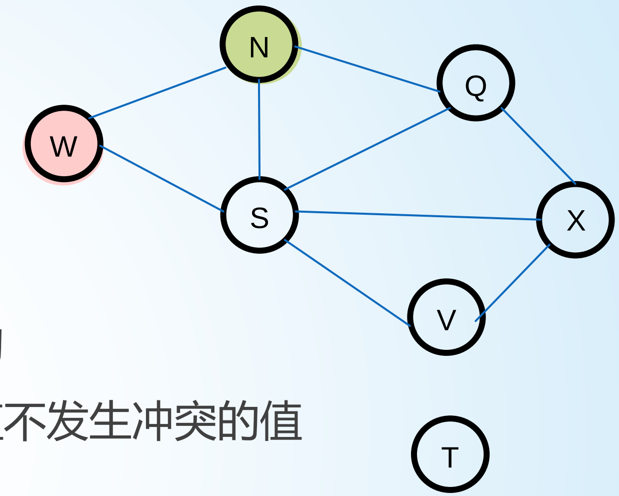


图 6.8 四皇后回溯搜索过程

- 只允许部分候选解存在，并且能够对是否有效解进行快速测试。
- 比枚举快，因为通过单次测试消除大量无意义的候选解。

第6讲 约束问题求解

6.3 回溯搜索：改进 引入启发式



- 变量和取值顺序
  - CSP不同于经典搜索问题的特殊性：
    - 每次一个变量，变量的选择是可交换的
    - 变量的赋值时，只需要考虑与前面赋值不发生冲突的值
- 引入启发式
  - 最少剩余值（MRV）启发式

    尽量选择合法取值最少的变量。（再比如算式谜，数独）
  - 度启发式

    通过选择参与其他未分配变量的最大约束数，来减少未来选择的分支因子。

第6讲 约束问题求解

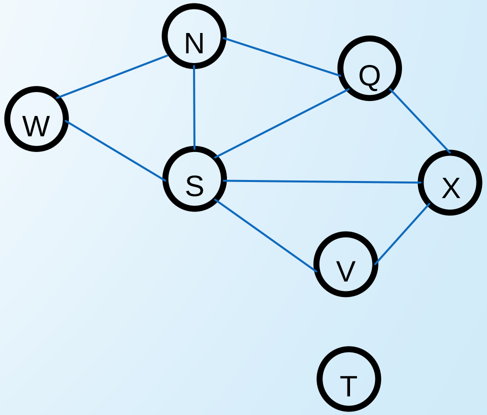
6.3 回溯搜索：改进 交叉搜索和推理

- 搜索中的推理

在搜索过程中进行前向检查（forward checking），去掉不满足约束条件的值

- 举例：具有前向检查的回溯搜索

	W	N	Q	X	V	S	T
初始值域	r g b	r g b	r g b	r g b	r g b	r g b	r g b
W=r	r	g b	r g b	r g b	r g b	g b	r g b
Q=g	r	b	g	r b	r g b	b	r g b
V=b	r	b	g	r	b		r g b

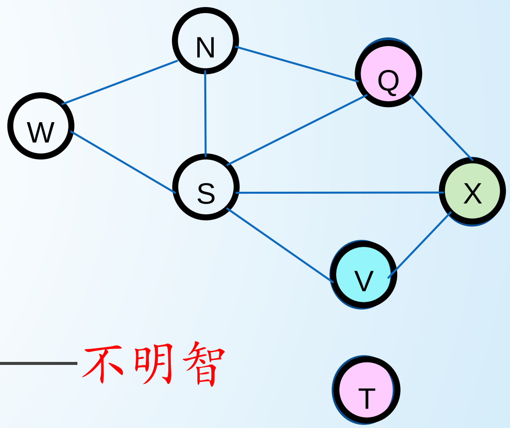


第6讲 约束问题求解

6.3 回溯搜索：改进 智能回溯

- 基本 时序回溯

回溯算法的策略：当算法搜索失败时，回到先前的变量并尝试不同的值。



举例分析：

- 变量赋值顺序： {Q, X, V, T, S, W, N}
- 当 {Q=r, X=g, V=b, T=r, S=?} 时
- S 不存在合法赋值，搜索失败
- 按照时序回溯，则重新为T尝试其他赋值 ——不明智

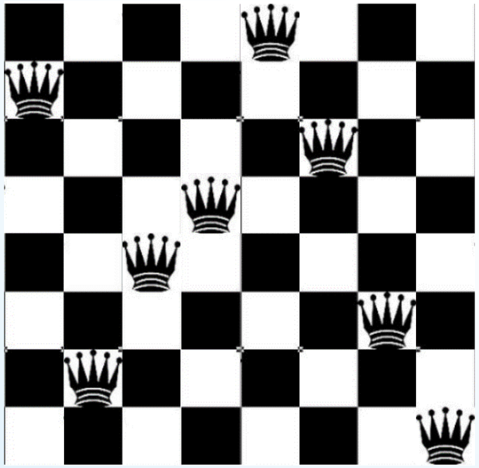
- 改进 智能回溯
  - 冲突集：与赋值失败的变量相冲突的变量集合
  - 冲突指导的回跳（回跳法 Back jumping）：回溯到发生冲突集中时间最近的变量赋值，即对 V 尝试重新赋值

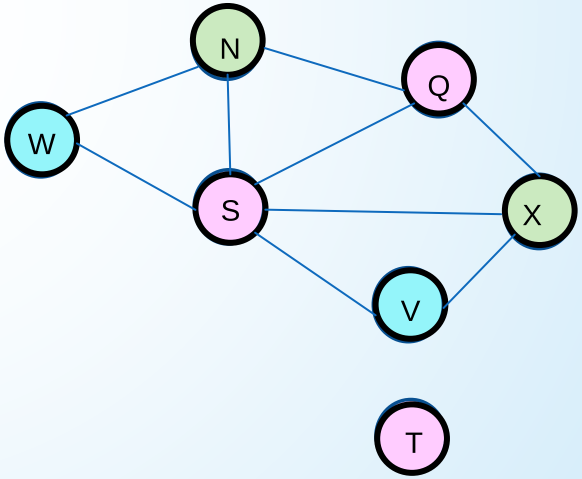
第 6 讲 约束问题求解

第6讲 约束问题求解

6.4 局部搜索

- CSP的局部搜索
  - 初始状态：对变量集合中的每个变量都赋一个值——全态形式化
  - 搜索过程：一次改变一个变量的取值，直到找到满足所有约束的解为止
- 举例：



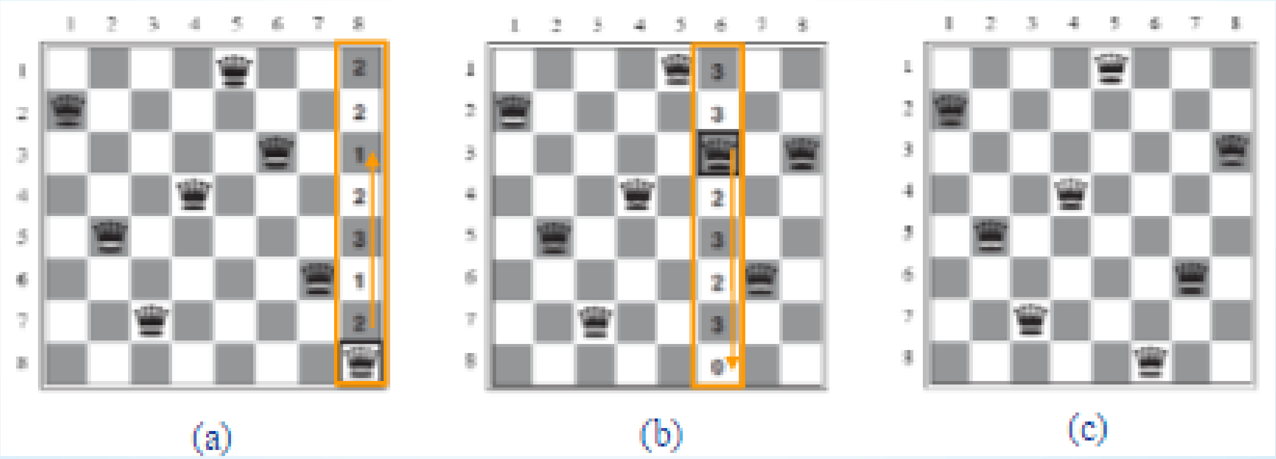


- 关键
  - 如何为变量选择新的赋值，以消解变量间的约束冲突？

第6讲 约束问题求解

6.4 局部搜索

- 最少冲突启发式（Min-conflicts heuristic）
  - 变量选择：随机选择任意一个冲突变量
  - 值的选择：选择导致与其它变量呈现最少冲突的新值



第6讲 约束问题求解

6.4 局部搜索

【小结】局部搜索与回溯搜索

- 回溯搜索

  （1）在约束的导引下寻找可行解

  （2）因为状态的定义方式与约束传播是相同的，所以二者可以结合使用

- 局部搜索

  在不可行解的基础上，不断消除不满足约束的冲突，最终达成可行解。

第 6 讲 约束问题求解

6.1 约束满足问题

6.2 约束传播

6.3 回溯搜索

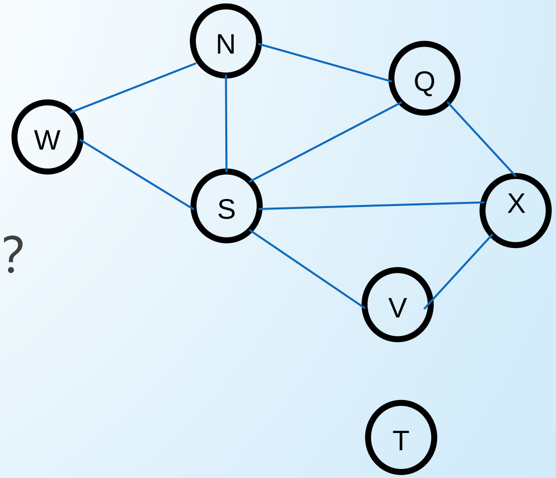
6.4 局部搜索

6.5 问题的结构

第6讲 约束问题求解

6.5 问题的结构

- CSP问题的结构化表示：约束图
- 问题分解：
  - CSP的复杂性，与约束图的结构密切相关
  - 现实世界的问题可以被分解为许多子问题（复杂性科学中的“分解与集成”）
- 独立子问题：
  - 约束图的独立组件
  - 独立性的判定：约束图中是否存在连通组件?
  - 假设CSP的每个组件对应一个子问题CSP_i，若赋值S_i是CSP_i的一个解，则$\cup_i S_i$是$\cup_i CSP_i$（即CSP）的一个解。



第6讲 约束问题求解

6.5 问题的结构

- 独立子问题 解的复杂度：
  - 将一个具有*n*个变量的CSP进行分解，设每个子问题仅有*c*个变量，则可以分解为*n/c*个子问题。
  - 若值域中元素个数为*d*，则每个子问题可以通过 d^c 个步骤进行求解。因此，求解该CSP的复杂度是 $O((n/c)\ d^c)$，与n呈线性关系。若不分解，复杂度为$O(d^n)$，与n是指数关系。

比如 n=80，d=2，c=20

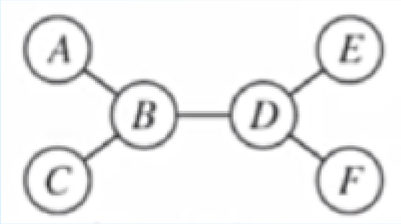
2^{80}个节点，每秒处理一千万个节点，需要四十亿年

$4*2^{20}$个节点，需要0.4秒

第6讲 约束问题求解

6.5 问题的结构

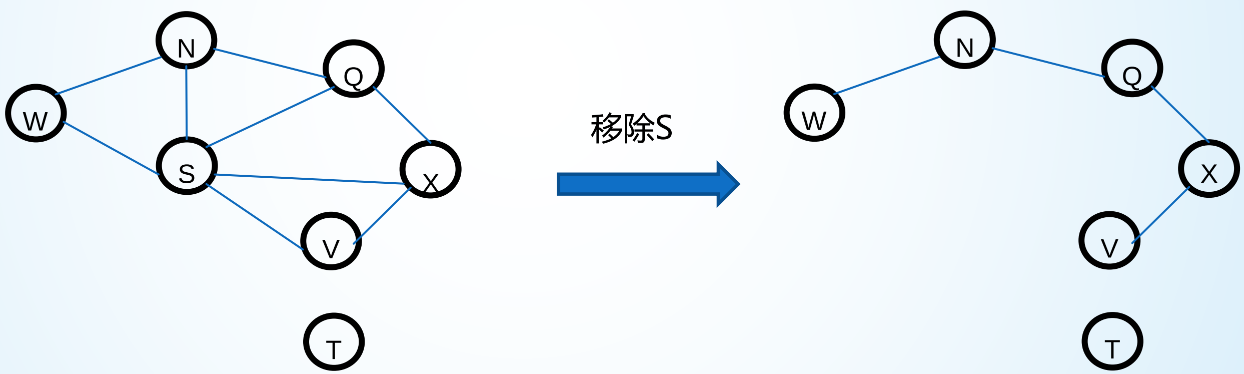
- 树：无圈的连通图；n个节点的树有n-1条边；任意2点间有且仅有一条链/路径
- 树结构CSP问题
  - 概念：约束图具有树结构的CSP
  - 特点：任意一个树结构的CSP，可以在变量个数的线性时间内求解
- 树结构CSP的求解
  - 先挑选任意变量作为树的根节点
  - 选择变量顺序（**拓扑排序**），则每个变量在树中出现在父节点之后



第6讲 约束问题求解

6.5 问题的结构

- 简化约束图为树结构：割集调整
  - 从CSP的变量中选择子集S（环割集 cycle cutset），使得剩下的约束图能够形成一棵树（剩余树 remaining tree）。

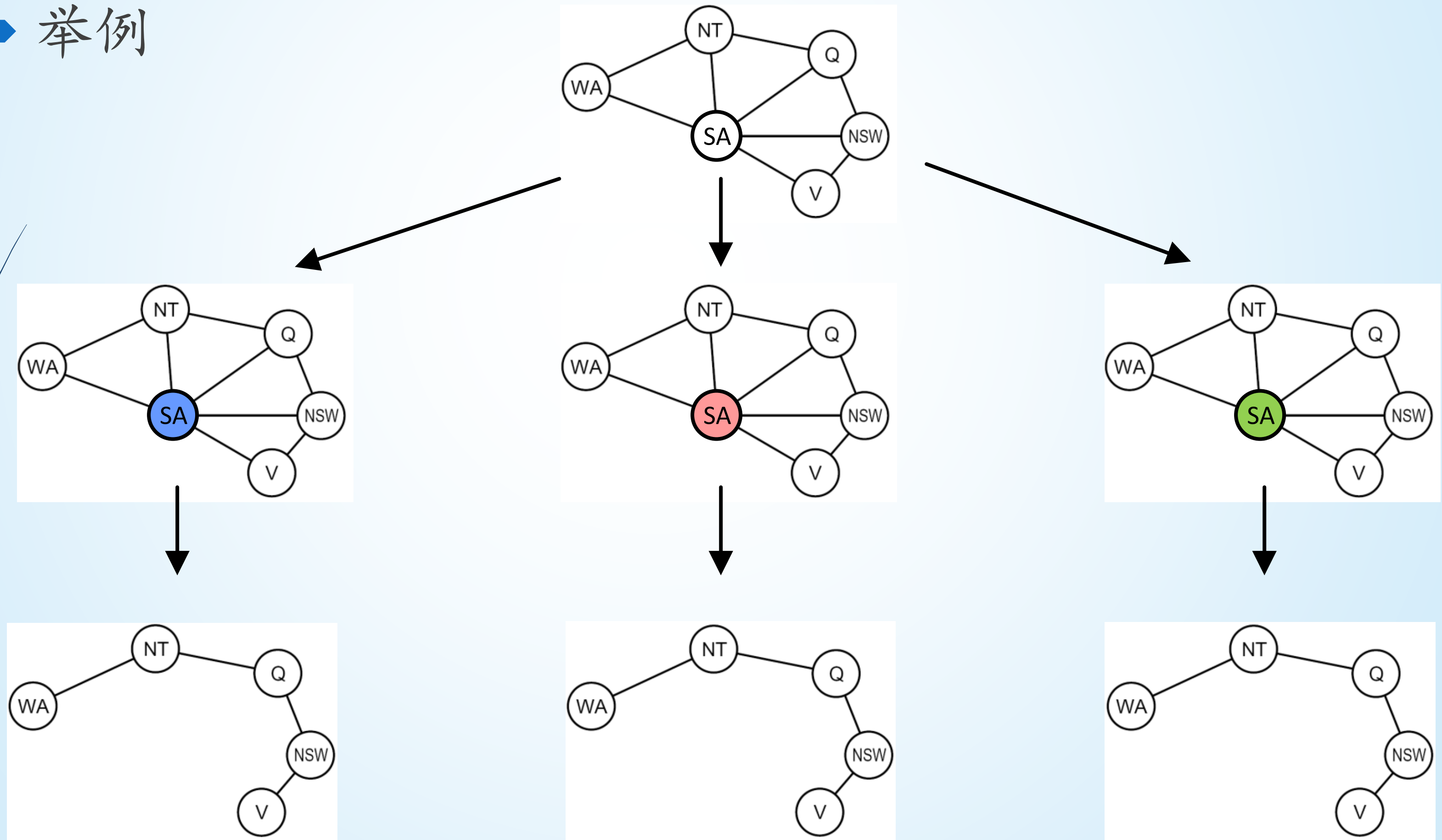


  - 对于满足S所有约束的S中变量的每个可能赋值
    - 从CSP剩余变量的值域中删除与S赋值不相容的值
    - 若剩余树有一个解，则将它与S的赋值一起返回
  - 继续尝试S上的其他所有可能赋值

第6讲 约束问题求解

6.5 问题的结构

- 举例



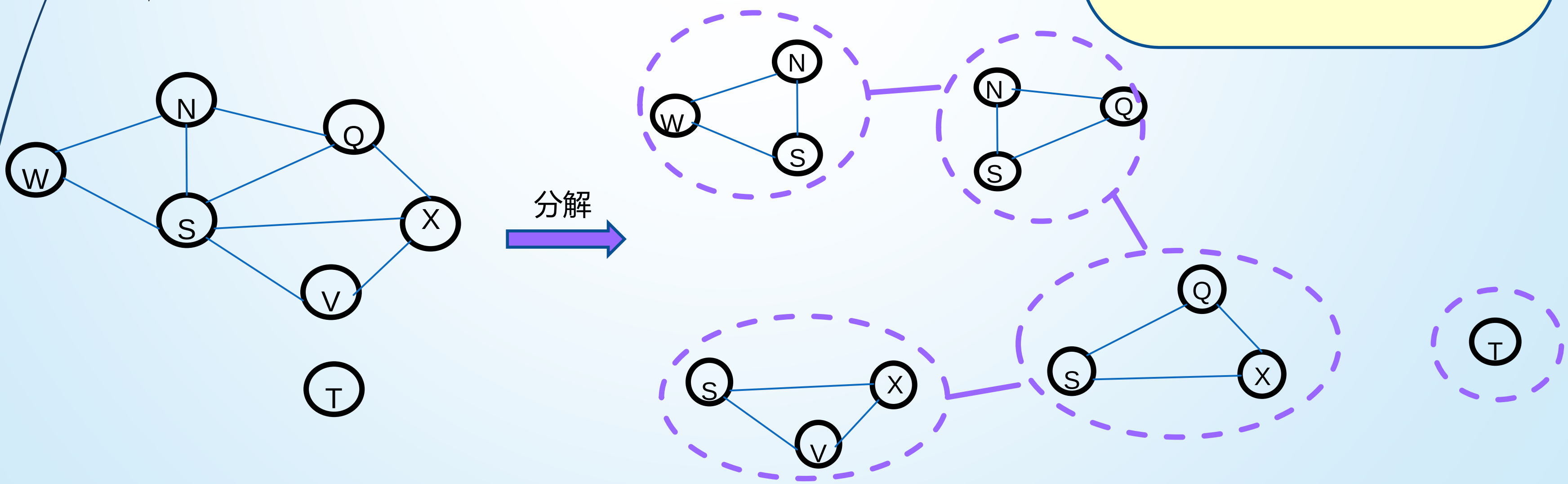
第6讲 约束问题求解

6.5 问题的结构

- 简化约束图为树结构：树分解
  - 将约束图分解为相关联的子问题
  - 分别独立求解各子问题
  - 把子问题的求解结果合并，形成问题的最终解
- 举例

树分解的条件

- 原始问题中的每个变量至少在一个子问题中出现
- 如果两个变量在原问题中由约束相连，则它们至少同时出现在一个子问题中（连同它们的约束关系）
- 如果一个变量出现在树中的两个子问题中，则它必须出现在连接这两子问题的路径上的所有子问题里

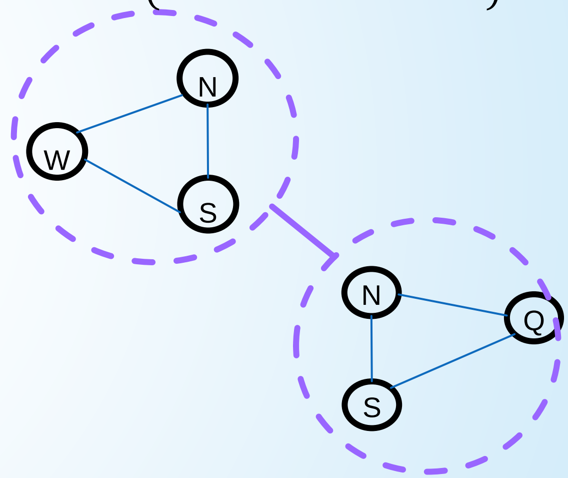


第6讲 约束问题求解

6.5 问题的结构

- 简化约束图为树结构：树分解
  - 分解子问题
  - 求解子问题
    - 分别独立求解各子问题，如果其中任何一个无解，则整个问题无解
    - 如果所有子问题均有解，则进一步构造整个问题的完整解
  - 构造完整解
    - 把每个子问题视为一个“巨型变量”，它的值域是这个子问题的所有解的集合
    - 用树算法来求解连接这些子问题的约束（即上层抽象问题）。子问题的约束**要求它们的共享变量要取相同的值**
    - 满足所有上层抽象问题的解，即为原问题的最终解。

$$\left\{\begin{matrix} \{W=r,S=b,N=g\} \\ \{W=r,S=g,N=b\} \\ \{W=b,S=r,N=g\} \\ \{W=b,S=g,N=r\} \\ \{W=g,S=b,N=r\} \\ \{W=g,S=r,N=b\} \end{matrix}\right\}$$



$$\left\{\begin{matrix} \{N=r,S=b,Q=g\} \\ \{N=r,S=g,Q=b\} \\ \{N=b,S=r,Q=g\} \\ \{N=b,S=g,Q=r\} \\ \{N=g,S=b,Q=r\} \\ \{N=g,S=r,Q=b\} \end{matrix}\right\}$$

作业

将以下算式谜游戏问题当作CSP求解：

SEND + MORE = MONEY