The Structure of Problems

School of Electronic and Computer Engineering
Peking University

Wang Wenmin

Contents

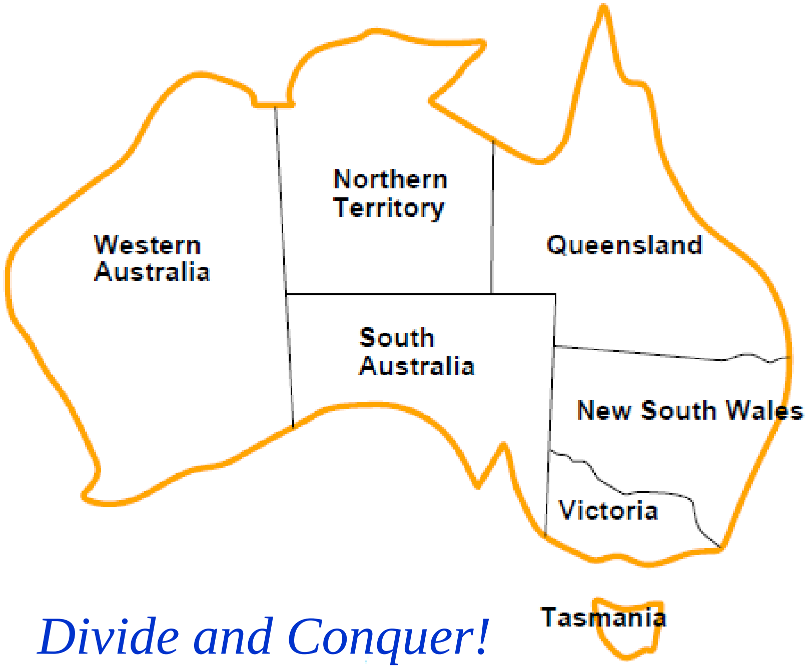
Decomposing Problem 问题分解

- The structure of problem as represented by constraint graph can be used to find solutions.
  由约束图所表征的问题结构，可以用于寻找解。
- The complexity of solving a CSP is strongly related to the structure of its constraint graph.
  求解一个CSP问题的复杂性，与约束图的结构密切相关。
- The problem in the real world can be decomposed into many sub-problems.
  现实世界的问题可以被分解为许多子问题。

Example:

Coloring Tasmania and coloring the mainland are independent sub-problems.

对塔斯曼尼亚着色与澳洲大陆着色是相互独立的子问题。



Divide and Conquer!

Independent Sub-problems 独立子问题

- They are identifiable as connected components of constraint graph.

  独立子问题可被标识为约束图的联接组件。

- Suppose a graph of n variables can be broken into sub-problems of only c variables: each worst-case solution cost is $O((n/c)\cdot d^c)$, linear in n.

  设n个变量的图可分解为仅有c个变量的子问题：每个最坏解的代价是O((n/c)・dᶜ)，n的线性关系。

- Without the decomposition, the total work is $O(d^n)$.

  如果不分解，则总的运行是O(dn)。

  - E.g., assuming n = 80, d = 2, c = 20, search 10 *million nodes/sec*.

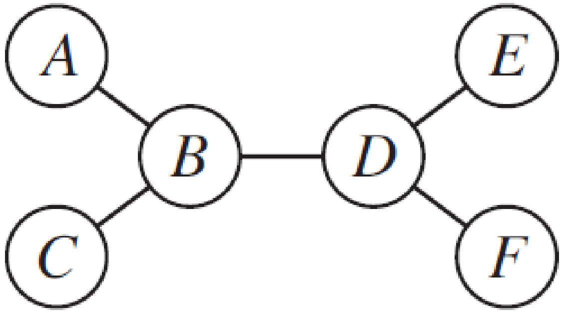
    例如， 假如 n = 80, d = 2, c = 20, 每秒搜索1千万个节点

    - Original problem: $d^n = 2^{80} = 4$ *billion years*;
    - 4 sub-problems : $(n/c)\cdot d^c = (80/20)\cdot 2^{20} = 0.4$ *seconds*.

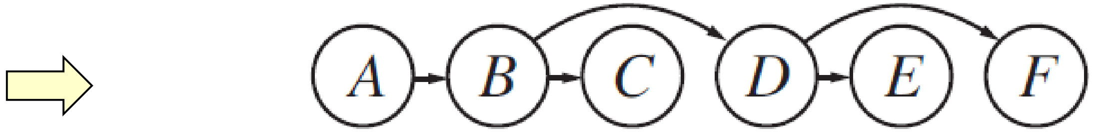
    原始问题：dn = 280 = 40亿年；4个子问题：(n/c)・dc = (80/20)・220 = 0.4秒。

Tree-structured Problems 树结构问题

- Any *tree-structured CSP* can be solved in *time linear* in the number of variables.
  任何树结构的CSP都可以用变量数中的时间线性加以解决。
- The method to solve a tree-structured CSP: first pick any variable to be root of tree, and choose an ordering (called a topological sort).
  求解树结构CSP的方法：先挑选任意变量作为树的根，然后再选择一个排列（称为拓扑排序）。



The constraint graph of a tree-structured CSP
树结构CSP的约束图



A linear ordering of the variables consistent with the tree with *A* as the root
一种以*A*为根与该树的变量一致的线性排列

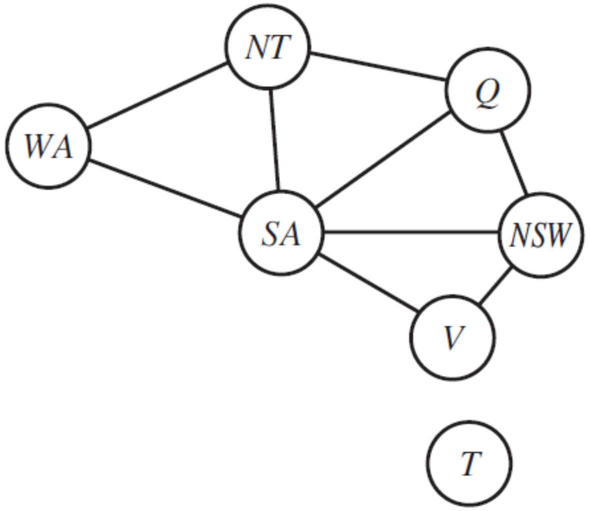
For a tree-structured CSP, it can be solved in $O(n \cdot d^2)$ time.
对于一个树结构的CSP，可以在$O(n \cdot d^2)$时间内得到解。

Algorithm to Solve Tree-structured CSPs 求解树结构CSPs的算法

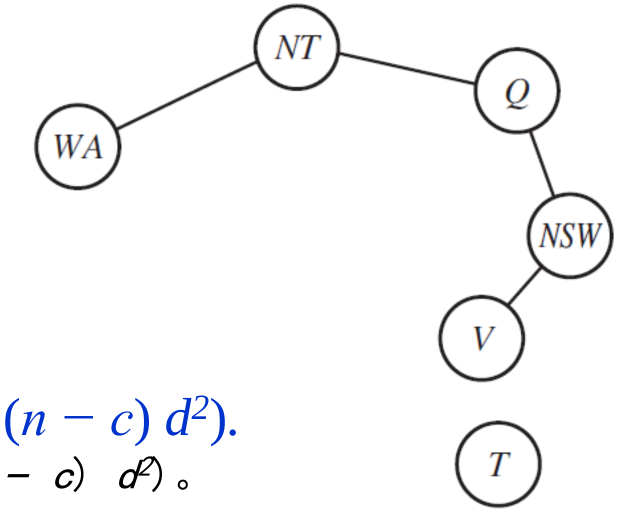
```
function TREE-CSP-SOLVER(csp) returns a solution, or failure
    inputs: csp, a CSP with components X, D, C
    n ← number of variables in X
    assignment ← an empty assignment
    root ← any variable in X
    X ← TOPOLOGICAL-SORT(X, root)
    for j = n down to 2 do
        MAKE-ARC-CONSISTENT(PARENT(X_j), X_j)
        if it cannot be made consistent then return failure
    for i = 1 to n do
        assignment[X_i] ← any consistent value from D_i
        if there is no consistent value then return failure
    return assignment
```

Reduce Constraint Graphs to Tree Structures 简化约束图为树结构

- 1st approach: cutset conditioning 第1种途径：割集调节
  - It can reduce a general CSP to a tree-structured one, and is quite efficient if a small cutset can be found.
    可以将一个通用CSP问题简化为一个树结构CSP，并且若能够找到一个小割集则相当有效。
- Conditioning: Instantiate a variable, prune its neighbors‘ domains.
  调节：对一个变量进行实例化，剪去它的相邻范畴。



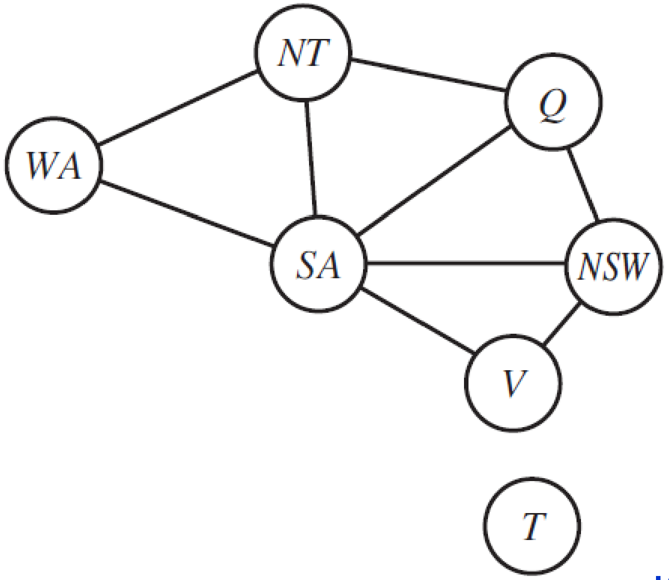
Remove *SA*
移除*SA*



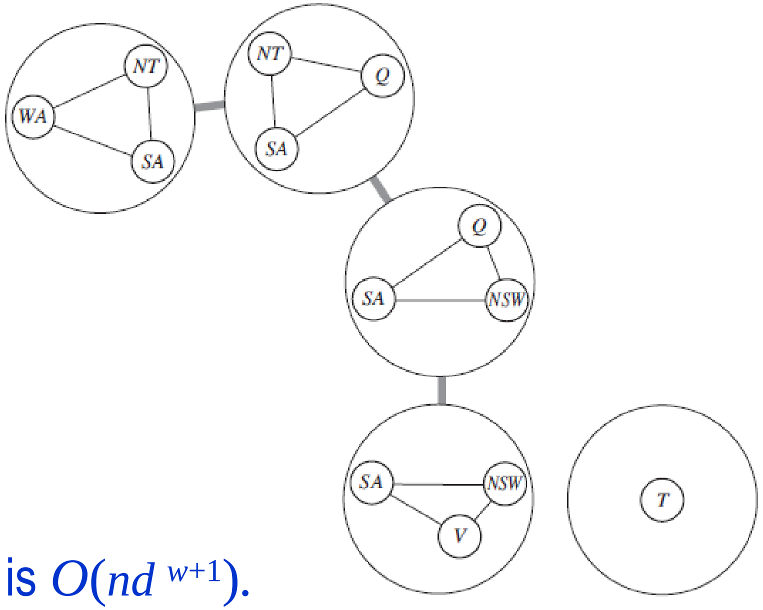
If cutset has size c, then run time is $O(d^c\,(n-c)\,d^2)$.
若割集的尺寸为c，则运行时间是$O(d^c\,(n-c)\,d^2)$。

Reduce Constraint Graphs to Tree Structures 简化约束图为树结构

- 2nd approach: tree decomposition 第2途种径：树分解
  - This techniques transform CSP into a tree of sub-problems and are efficient if the tree width of the constraint graph is small.
    这种技法将CSP转换成一棵子问题树，并且当该约束图的树宽较小时很有效。



Decomposition
分解



If a graph has tree width w, then run time is $O(nd^{w+1})$.
若一个图的树宽为w，则运行时间是$O(nd^{w+1})$。

Thank you for your attention!

PoAI