第11章 算法能力的极限

*Steven*

**1. 简述：**

**（1）问题变换（或问题化简）和复杂性归约的基本思想。**

**（2）求解问题下界的意义；求解问题下界的几类主要方法。**

**（3）P、NP和NP完全的概念。**

**（4）证明问题NP完全性的基本思路。**

(1)

**问题变换：**

A问题难以设计算法，所以对问题B建立一个与A相同的下界，通过以下三步将问题A变换为B：

1. 将问题A的输入变换为问题B的适当输入
2. 解出问题B
3. 把问题B的输出变换为问题A的一个正确解

其中，若用时间能完成变换的第1和第3步，且，则称问题A是时间可变换到问题B的。

**复杂性归约：**

若A的下界为,，则为B的一个计算时间下界。

若B的上界为, ，则为A的一个计算时间上界。

(2)

**求解问题下界的意义：**确定某问题任意算法的最佳效率，以及确定该问题的效率极限。

**主要方法：**

1. 渐进效率类型
2. 根据同样问题的其他求解算法来看算法的效率

(3)

P：P类判定问题：这些问题可以用一个确定性算法在多项式时间内判定或解出

NP：NP类判定问题：这些问题可以用一个确定性算法在多项式时间内检查或验证它们的解

NP完全：一个判定问题是NP完全的，当且仅当

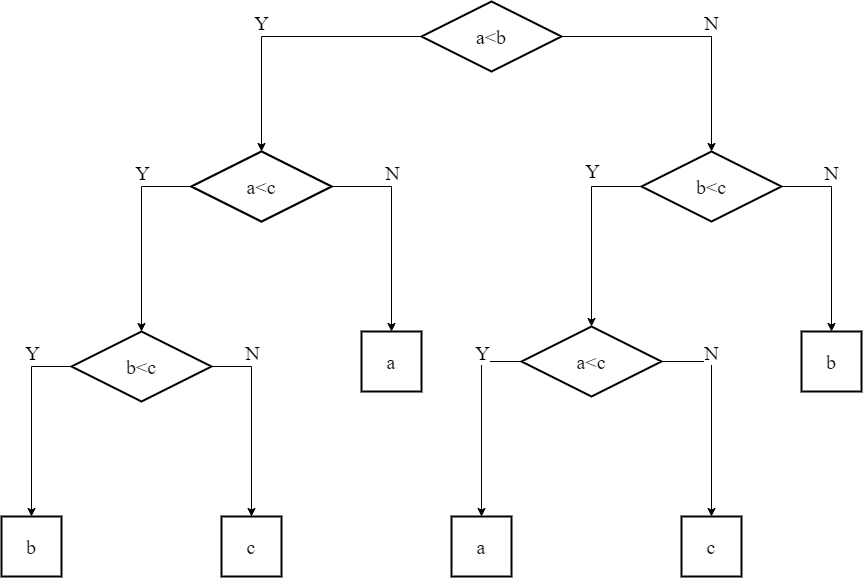
对于NP中的每一个问题

(4)

<1> 存在一个NP完全问题

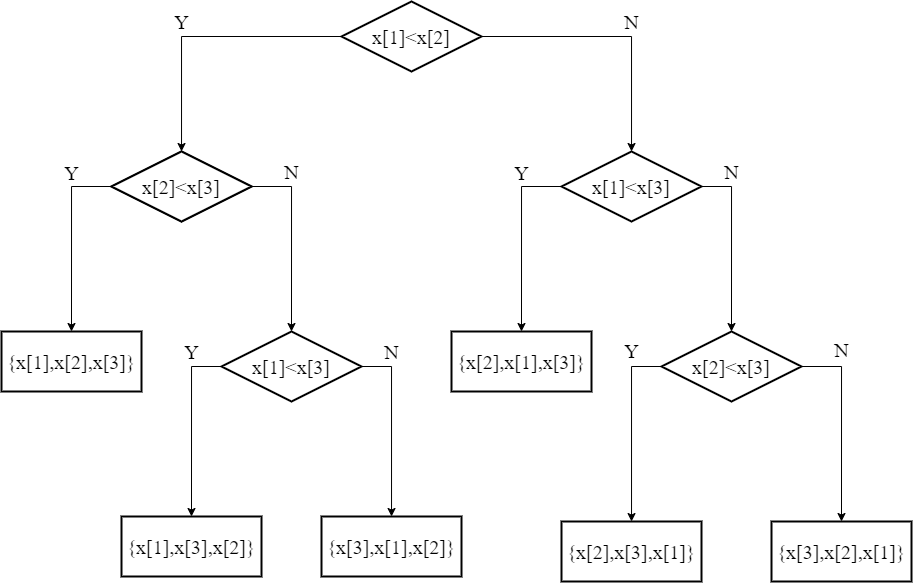
<2> 对于，证明

**2. 考虑在三元素集合中“求中值”问题，为求解该问题的算法画一棵决策树。**



**3. 对于“三元素的基本冒泡排序”问题，画出其决策树，并基于决策树分别求出最坏情况和平均情况下的键值比较次数。**

（以下输出结果按照升序排列）



最坏情况的比较次数是3次，即n次，最好情况的比较次数是2次。平均比较次数为 次。

**4. 定义“背包问题”的判定版本，并且简要描述问题的一个多项式时间的算法，它能够检验某个特定解是不是问题的一个解（可以假设一个特定解代表了检验算法的一个合法输入）。**

背包问题的判定版本：给定一个特定解x，要判断其重量和是否小于背包容量，以及价值之和是否大于某一下界。

给定一个特定解x，要判断其重量和是否小于背包容量，以及价值之和，只需要进行2\*(|x|-1)次加法（|x|是x的元素个数），时间复杂度为O(2|x|-1)，能在多项式时间内完成判断，所以0-1背包问题的判定问题是NP问题。