算法设计与分析

NP完全性

主要内容

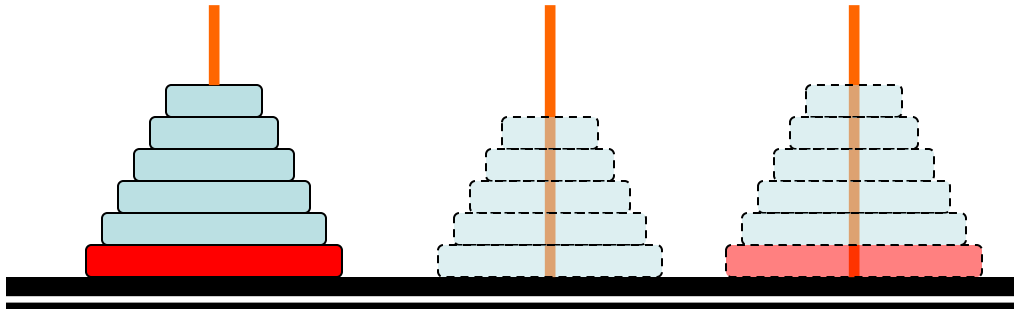
- 计算模型
- 判定问题
- P类问题
- NP类问题
- NP完全问题

不同时间复杂度对应运行时间对比

Algorithm	1	2	3	4	
Time function(ms)	$33n$	$46n \lg n$	$13n^2$	$3.4n^3$	2^n
Input size(n)	Solution time				
10	0.00033 sec.	0.0015 sec.	0.0013 sec.	0.0034 sec.	0.001 sec.
100	0.0033 sec.	0.03 sec.	0.13 sec.	3.4 sec.	4×10^{16} yr.
1,000	0.033 sec.	0.45 sec.	13 sec.	0.94 hr.	
10,000	0.33 sec.	6.1 sec.	22 min.	39 days	
100,000	3.3 sec.	1.3 min.	1.5 days	108 yr.	
Time allowed	Maximum solvable input size (approx.)				
1 second	30,000	2,000	280	67	20
1 minute	1,800,000	82,000	2,200	260	26

经典问题回顾-Hanoi Tower

- 对于N个盘子的问题，如果使用递归进行求解，需要2^N-1次移动
- N=64时…….



两大类问题（非严格分类）

- 多项式时间
  - 排序：O(nlgn)
  - Ordered searching: O(logn)
  - 最大元、最小元：O(n)
- 非多项式时间
  - 旅行商问题： $O(n!)$
  - 0-1背包问题：$O(2^n)$

字符串与语言

- 字母表: 任意一个有限集. 常用记号Σ, Γ
- 符号: 字母表中的元素
- 字符串: 字母表中符号组成的有限序列, 如asdf, 通俗地说即单词
- 串的长度, 例: abcde的长度为5
- 串的连接, 例:abc与de的连接是abcde
- 串的反转, 例:abcde的反转是edcba
- 空词: 记为ε, 长度为0
- 语言: 一些字符串的集合

语言举例与字典序

- 取字母表 $\Sigma = \{0,1\}$, Σ上的语言举例:
- $A=\{0,00,000,000000\}$
- $B=\{1,01,11,001,011,101,111,\ldots\}$
- $C=\{\varepsilon,0,1,00,11,000,010,101,111,\ldots\}$
- Σ上所有字符串的字典序排列

  $\varepsilon,0,1,00,01,10,11,000,\ldots$

将计算问题转化成语言

- 计算理论中总是将计算问题转化成语言:
- “一数是否是2的倍数”-----{ 以0结尾的01串 }
- “一串是否含有子串010”---{ 含有010的01串 }
- “图是否连通” ----------{ <G> | G是连通图 }
  - 其中<G>是图G编码成的字符串

计算的模型--自动机理论

- 有限自动机(FA)
  - 存储量极小的计算机
  - 在文本处理,编译程序及硬件设计中有应用
- 下推自动机(PDA)
  - 带一个栈的计算机
  - 在程序设计语言和人工智能中有应用
- 图灵机(TM)
  - 有无限可改写存储的计算机
  - 能解决实际计算机所能解决的一切问题
- FA和PDA是简化的图灵机
- 一般将以图灵机作为计算的理论模型

计算的复杂性

- 算法理论中，关注
  - 从计算的观点看，要解决的问题的内在复杂性如何？
  - 它是易的还是难的？
- 复杂性的度量: 耗费的时间, 耗费的存储

P与NP是否相等

- P问题是容易解决的问题
- NP问题是容易验证的问题
- $P \subseteq NP$,但是$P=NP$还是$P \subset NP$(真包含)?
- 它是21世纪的七大数学难题
- 复杂性的应用: 密码学， 编码与解码

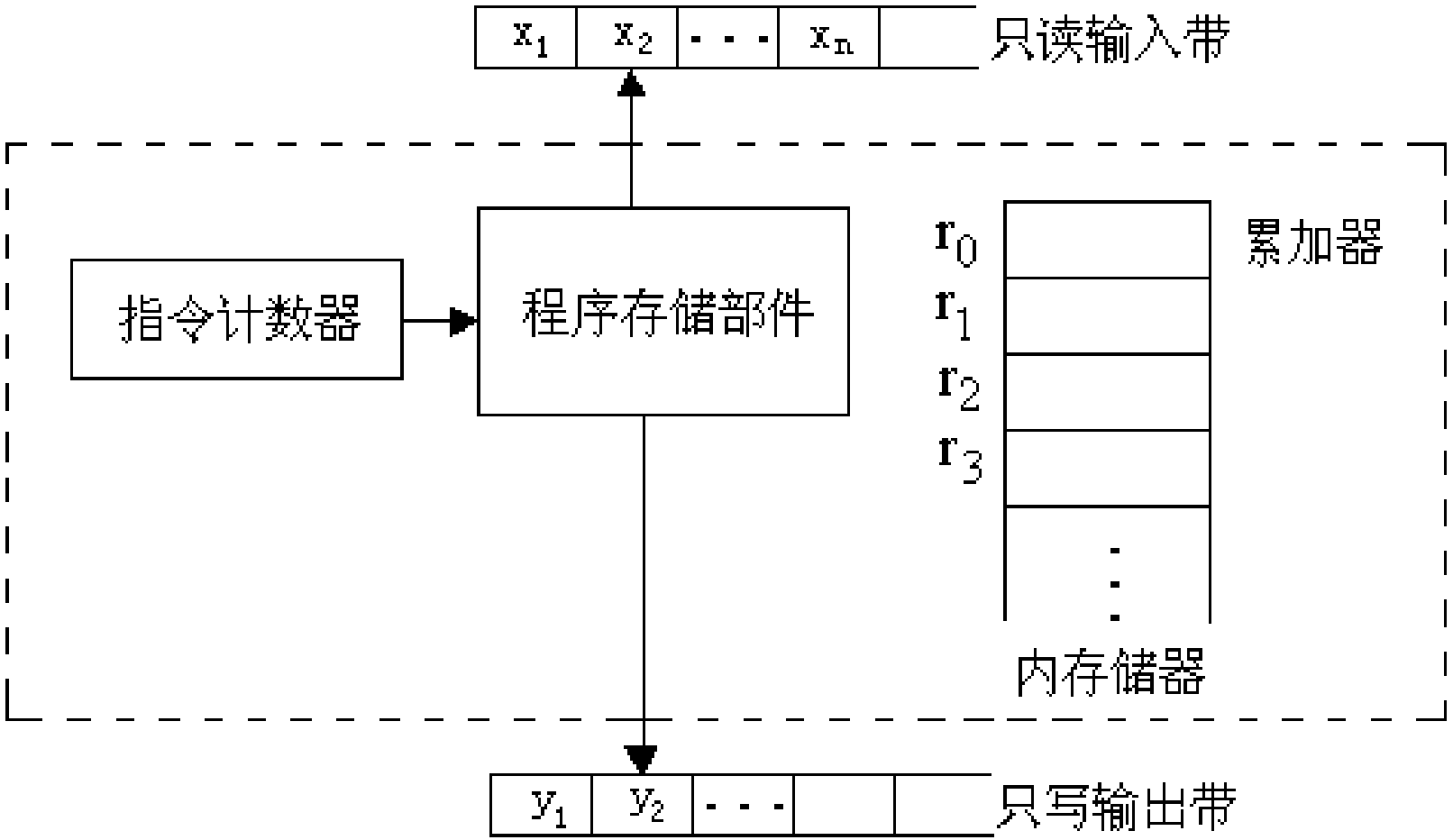
计算模型

- 计算模型是计算机科学的重要组成部分
- NP-完全概念的原始定义需要借助非确定性Turing机等计算模型的概念
- 计算模型研究中有不少好的思想可供借鉴

计算模型

- RAM
- RASP
- 确定性Turing机
- 非确定性Turing机

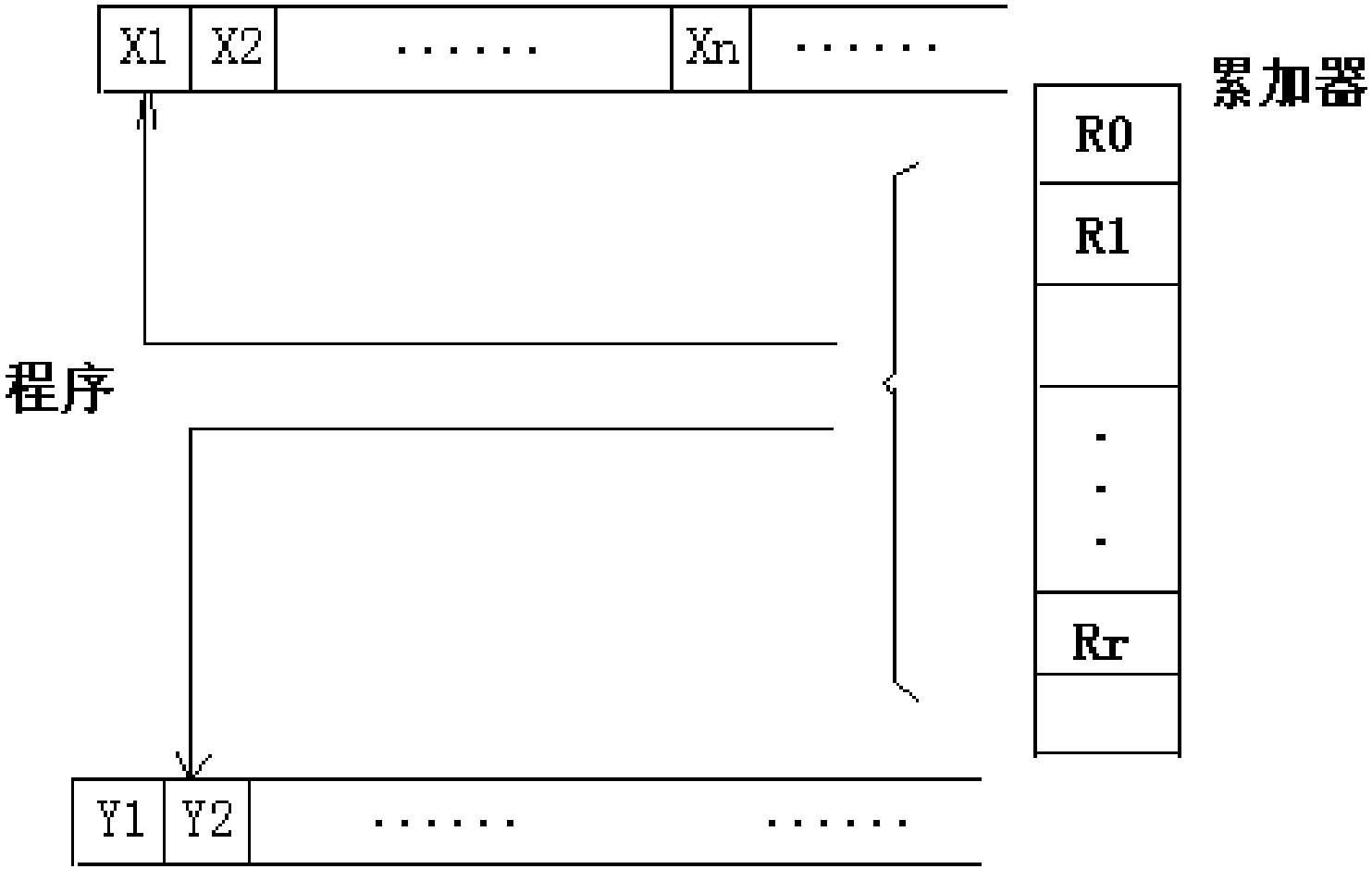
Random Access Machines（RAM）



Random Access Machines（RAM）

- 一个RAM程序定义了从输入带到输出带的一个映射。可以对这种映射关系作2种不同的解释
  - 把RAM程序看成是计算一个函数
  - 把RAM程序当作一个语言接受器

Random Access Stored Program



Random Access Stored Program

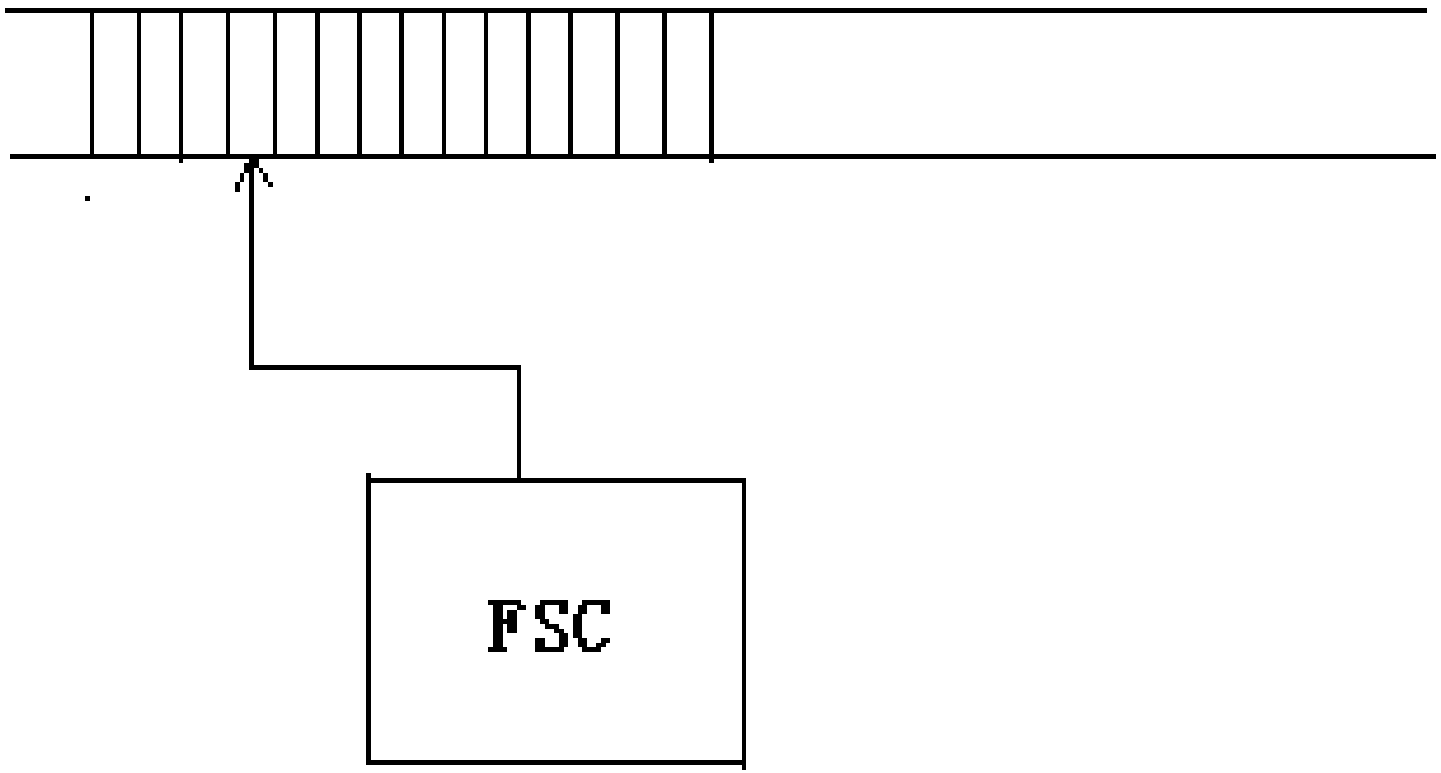
- 存储程序模型RASP
  - RASP的整体结构类似于RAM，所不同的是RASP的程序是存储在寄存器中的。每条RASP指令占据2个连续的寄存器。第一个寄存器存放操作码的编码，第二个寄存器存放地址。

Turing机

- 1936年，Alan Turing提出一个问题：
  - 什么是计算？什么问题是可计算的？
  - Turing 认为计算是一个人拿一支笔在一张纸上进行的操作，输入是眼睛看到的符号，根据脑中的规则在纸上擦掉或写上一些符号；再用眼睛看下面的符号，根据规则进行擦写的工作；重复上述工作，直到这个人认为可以结束为止。此时，最后写下的符号就是所要的结果。
- 特点：在这个（计算）过程中，只用到了有穷多个符号
- Turing 根据这个过程构造出了一个计算模型，称之为Turing机

Turing机

单带Turing机模型



Turing机

- Turing发现该模型可以实现非常复杂的计算
- Turing称：凡是Turing机可以计算的问题就是可计算的，否则就是不可计算的
- 由此引出可计算性理论的研究

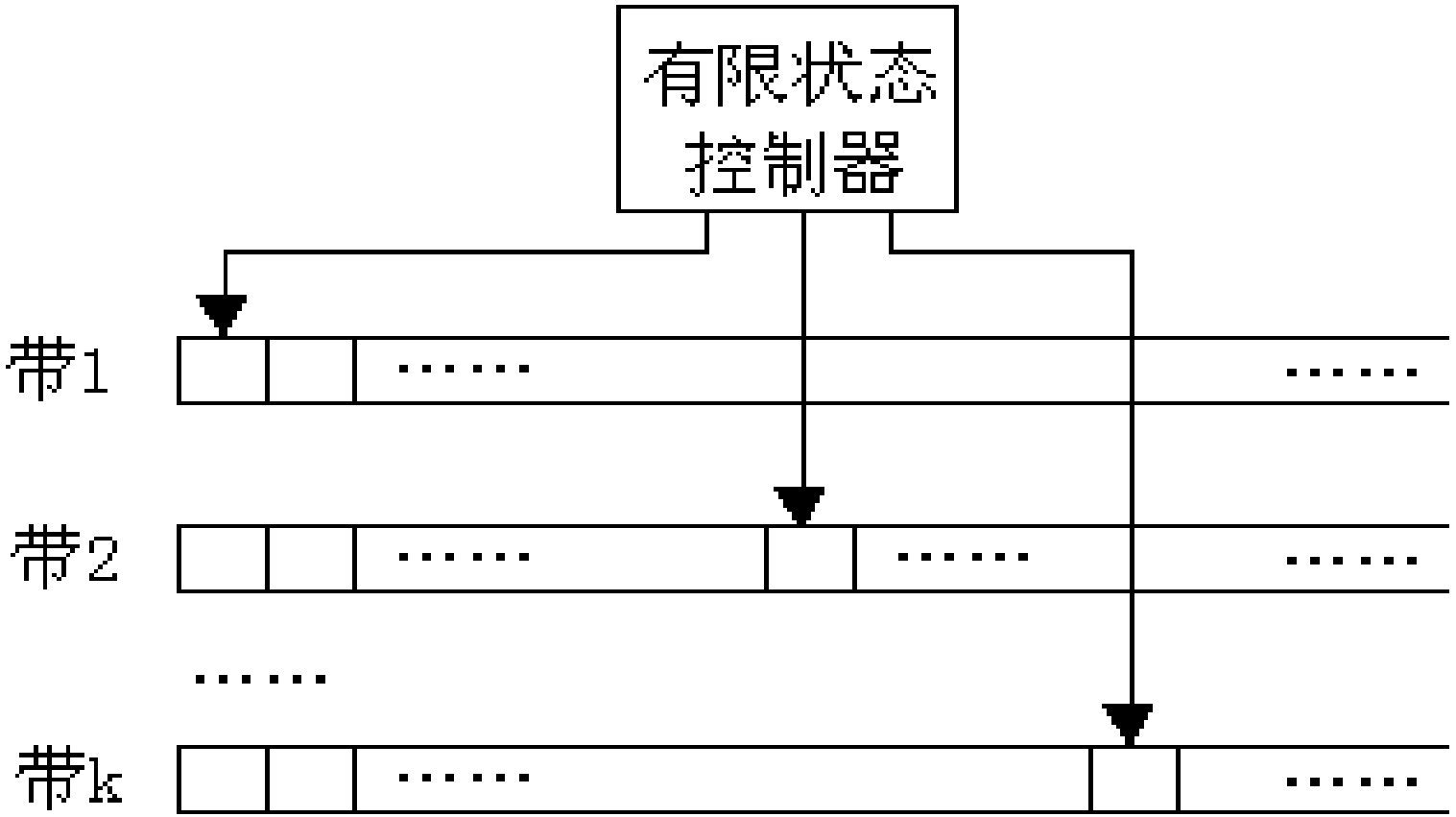
Turing机

- Church-Turing Thesis（丘奇-图灵论题）
  - 任何合理的计算模型都是相互等价的（计算范围相同）。
  - 合理：单位时间内可以完成的工作量，有一个多项式的上限。
- 迄今为止所有被提出的合理计算模型均满足该论题

Turing机的形式

- 单带双向Turing机
- 单带单向Turing机
- 多带双向Turing机
- 多带单向Turing机
- 已经证明，这些Turing机的计算能力（在多项式意义下）相同

Turing机



Turing机

- 根据有限状态控制器的当前状态及每个读写头读到的带符号，图灵机的一个计算步可实现下面3个操作之一或全部。
  - (1)改变有限状态控制器中的状态。
  - (2)清除当前读写头下的方格中原有带符号并写上新的带符号。
  - (3)独立地将任何一个或所有读写头，向左移动一个方格(L)或向右移动一个方格(R)或停在当前单元不动(S)。

Turing机的形式

- k带图灵机可形式化地描述为一个7元组(Q，T，I，δ，b，q_0，q_f)，其中:
- (1)Q是有限个状态的集合。
- (2)T是有限个带符号的集合。
- (3)I是输入符号的集合，$I \subseteq T$。
- (4)b是唯一的空白符，$b \in T-I$。
- (5)q_0是初始状态。
- (6)q_f是终止(或接受)状态。
- (7)δ是移动函数。它是从$Q \times T^k$的某一子集映射到$Q \times (T \times \{L，R，S\})^k$的函数。

Turing机的形式

- 与RAM模型类似，图灵机既可作为语言接受器，也可作为计算函数的装置。

非确定性Turing机

- 一般地说，将可由多项式时间算法求解的问题看作是易处理的问题，而将需要超多项式时间才能求解的问题看作是难处理的问题。
- 有许多问题，从表面上看似乎并不比排序或图的搜索等问题更困难，然而至今人们还没有找到解决这些问题的多项式时间算法，也没有人能够证明这些问题需要超多项式时间下界。
- 在图灵机计算模型下，这类问题的计算复杂性至今未知。
- 为了研究这类问题的计算复杂性，人们提出了另一个能力更强的计算模型，即非确定性图灵机计算模型，简记为NDTM(Nondeterministic Turing Machine)。
- 在非确定性图灵机计算模型下，许多问题可以在多项式时间内求解。

非确定性Turing机

- 一台k带DTM（确定性Turing机）根据其当前所在状态及k个读写头当前读到的字符唯一地确定下一步的动作
- 与DTM不同的是，NDTM的每一步动作允许有若干个选择

P类与NP类语言

- P类和NP类语言的定义：
  - P={L|L是一个能在多项式时间内被一台DTM所接受的语言}
  - NP={L|L是一个能在多项式时间内被一台NDTM所接受的语言}

NP-C类语言

- 定义（狭义，Karp）：称满足下述2条的语言L_0是NP-C的：
  - 1)$L_0 \in NP$;
  - 2)$\forall L \in NP$，都有$L \leq_p L_0$。

P类问题和NP类问题

- NP类问题
  - 一个判定问题中的每一个实例编码为一个符号串ω，使得该实例的回答为Yes当且仅当它的编码ω在多项式时间里被一台NDTM所接受，则称该判定问题属于NP类
- 对于P类问题也可以类似地进行定义
- NP-完全性问题的定义（依赖于语言）：
  - 若某个判定问题进行编码后，所对应的语言L0是NP-C的，则称该问题是NP-C的

最优化问题和判定问题

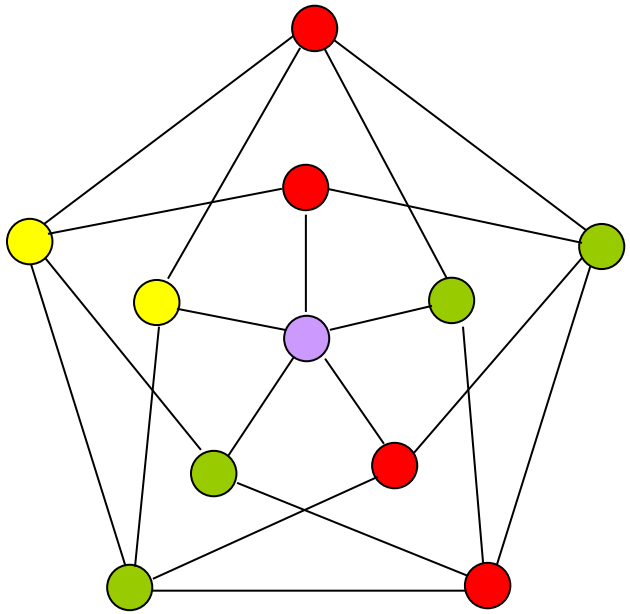
- 判定问题：回答是“Yes”或“No”
- 最优化问题（不是Yes-No问题）可以与一个判定问题相对应
  - 比如，最优化问题：团集问题(CLIQUE)：任给一个无向图G，找出G中最大的团集。团集：点的集合，满足：任两点之间均有边相连
  - 对应的判定问题：G中是否有k-团（k个顶点的团集）？

例：Graph Coloring

- Given a undirected graph G and a positive integer k, is there a coloring of G using at most k colors?

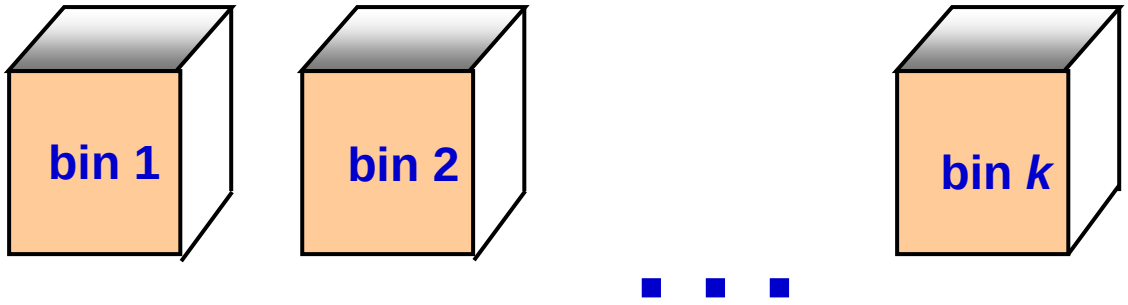
problem size: (n,k)

proposed solutions: k^n



例：Bin Packing

- Given k bins each of capacity one, and n objects with size s_1, …, s_n, (where s_i is a rational number in (0,1]). Do the n objects fit in k bins?



problem size: *n*

Proposed solutions: *n*!

例：Knapsack

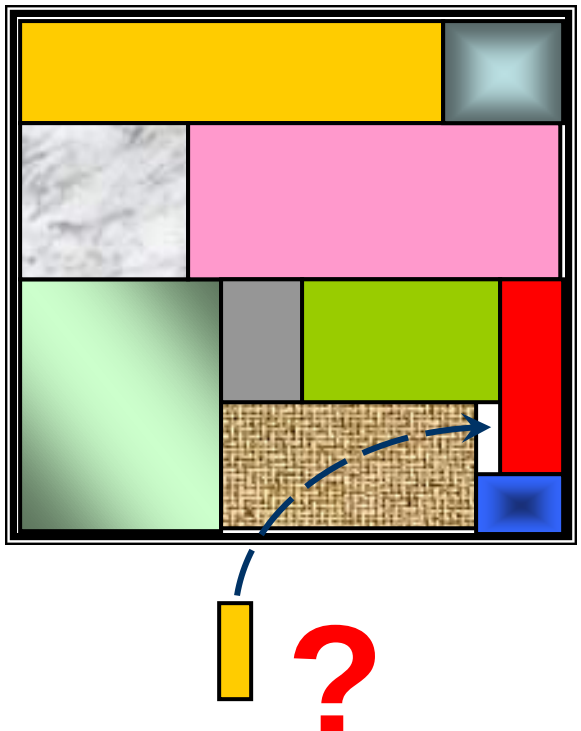
- Given a knapsack of capacity C, n objects with sizes s_1, …, s_n and "profits" p_1, …, p_n, and a positive integer k. Is there a subset of the n objects that fits in the knapsack and has total profit at least k?

例：Subset Sum

- Given a positive integer C and n objects with size s_i each. Is there a subset of the objects whose sizes add up to C exactly?

problem size: *n*

Proposed solutions: 2^n



例：CNF-Satisfiability

- Given a CNF(conjunctive normal form) formula, is there a truth assignment that satisfies it?

p	q	$\alpha(p,q)$
T	T	T
T	F	F
F	T	F
F	F	T

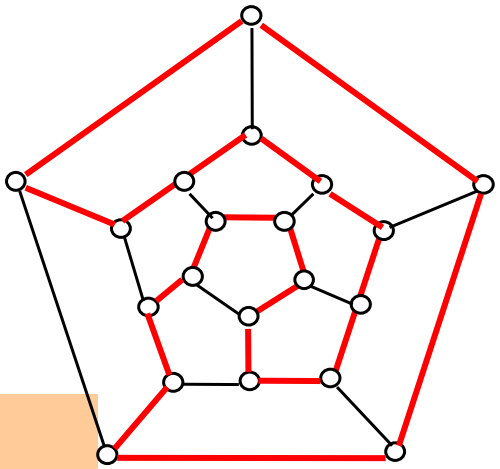
$(p \vee \sim q) \wedge (\sim p \vee q)$

problem size: variable number(?)

proposed solutions: 2^n

例：Hamiltonian Problems

- Hamiltonian cycles or Hamiltonian paths
  - Given a undirected graph G. Does G have a Hamiltionian cycle of Hamiltonian path?



problem size: *n*

proposed solutions: *n*!

例：TSP

- Given a complete, weighted graph and an integer k, is there a Hamiltonian cycle with total weight at most k?

problem size: $n=|V_G|$

proposed solutions: $n!$

“证书”（certificate）

- 定义：若集合S包含判定问题A的所有解，则称S是A的证书集，S中的元素称为A的一个证书（certificate，注意证书不一定是解）

“证书”（certificate）

- k-团问题
  - 给定无向图G=(V,E)，顶点集V的任何一个k顶点子集V’就是k-团问题的一个证书
  - 如果Q 是k-团问题的解，则一定有$Q \in S$

“证书”（certificate）

- Hamilton路径问题
  - 给定无向图G=(V,E)，任何一个由n个互不相同的顶点构成的序列 就是H路径问题的一个证书
  - 如果图G有Hamilton路径，则该路径一定属于S

“证书”（certificate）

- SAT问题
  - 给定一个n元的布尔表达式$f(x_1,x_2,\ldots,x_n)$，则n元组$(\alpha_1, \alpha_2,\ldots, \alpha_n)$ (α_i或为0或为1，已确定)为该问题的一个证书
  - 若$f(x_1,x_2,\ldots,x_n)$可满足，则使$f(x_1,x_2,\ldots,x_n)$为1的n元组含在其中

P类问题和NP类问题

- 若判定问题A满足：1、有证书集S；2、存在一个算法F，对于S中的每一个证书α，F都能够在多项式时间里验证α是否为A的一个解，则称$A \in NP$
- 基于以上定义，通常称
  - NP类问题是多项式时间可验证的
  - P类问题是多项式时间可解的

k-团问题的验证

- 对于任给的一个证书V’（k顶点子集即|V’|=k），只要逐一检查V’中的任意两点之间是否有边
- 这样的点共有k(k-1)/2个，若每个点对间均有边相连，则V’是一个k-团，否则不是
- 验证算法时间为$\Theta(k^2)$，故是多项式时间可验证的

Hamilton路径问题的验证

- 对于任给的一个证书即序列，只要逐一检查序列中相邻点之间是否有边相连
- 若n-1个相邻点对均有边相连，则该序列是Hamilton路径，否则不是
- 验证算法时间为$\Theta(n)$，故是多项式时间可验证的

SAT问题的验证

- 对于任给的一个证书即n元组($\alpha_1, \alpha_2, \ldots, \alpha_n$)($\alpha_i$或为0或为1，已确定)，只要把这n个具体的0或1值代入布尔表达式$f(x_1,x_2,\ldots,x_n)$，进行逻辑运算后即可知$f(\alpha_1, \alpha_2, \ldots, \alpha_n)$的值为0还是为1
- 若$f(x_1,x_2,\ldots,x_n)$的长度为m，则计算$f(\alpha_1, \alpha_2, \ldots, \alpha_n)$的时间为$\Theta(m)$，故是多项式时间可验证的

NP完全（NP Complete）问题

- 非形式定义，如果一个问题属于NP类，且与NP类中的任何问题是一样“难”的，则说它属于NP-C类，也称其是NP完全的（NP-Complete）
- NP完全问题是难处理的，迄今为止，未发现有NP完全问题的多项式时间求解方案

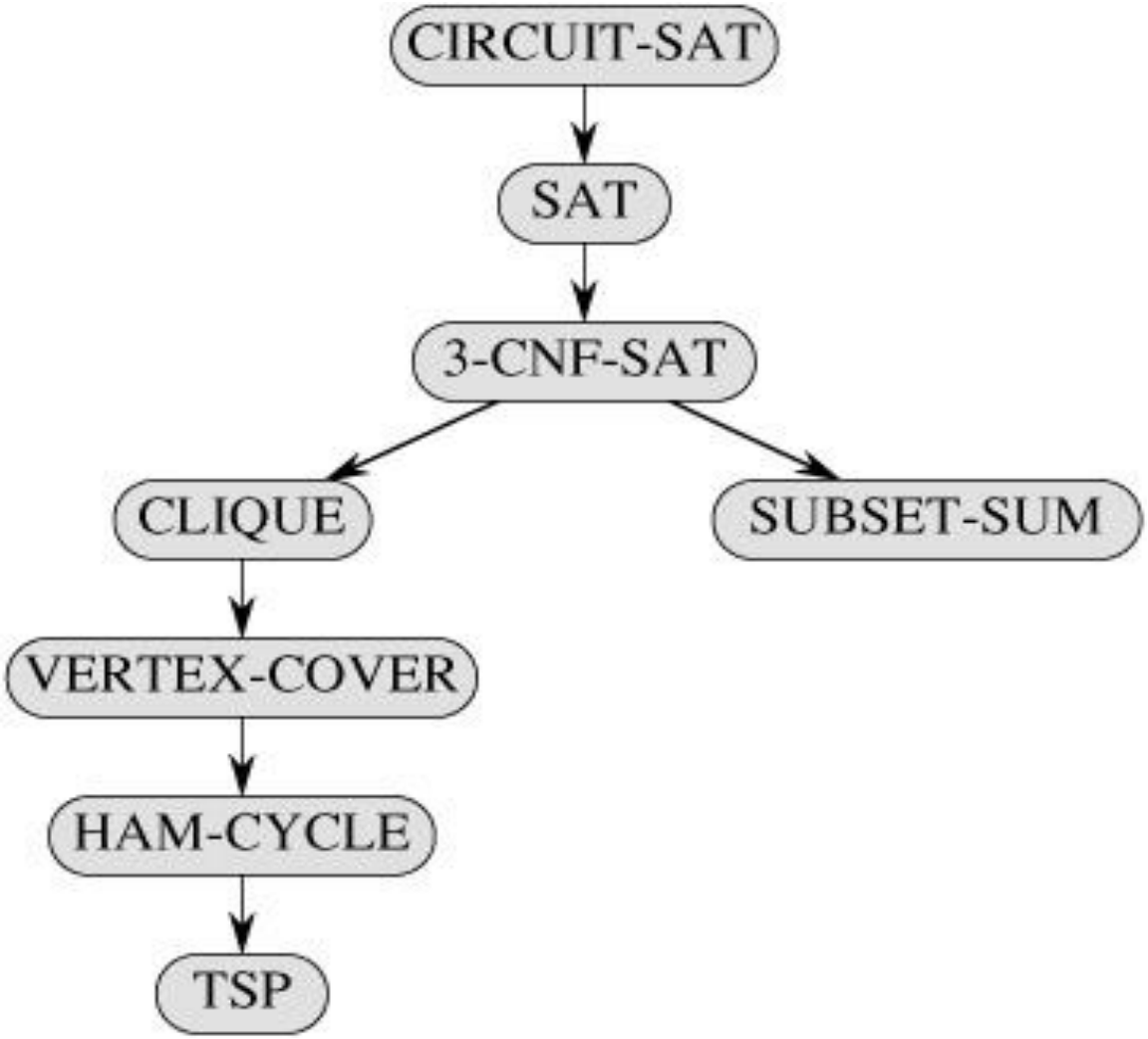
问题的多项式变换（规约）

- 设IA是判定问题A的任一实例，B是另一判定问题，若存在一个从A到B的映射f满足
  - ∀IA∈A，实例IA的答案为‘Yes’ *iff* f(IA)（与 IA相对应的B问题的实例）的答案为‘Yes’
  - ∀IA∈A，若IA的输入规模为n，则对应实例f(IA)的输入规模不超过PA(n)，这里PA(n)是一个与判定问题A有关的多项式
  - ∀IA∈A，若IA的输入规模为n，则对应实例f(IA)的输入在多项式时间p(n)内可计算
- 则称问题A可多项式变换为B，记作$A \leq_p B$，($\leq$p亦称Karp规约)

多项式变换(规约)的作用

- 若A≤pB且问题B是多项式时间可判定的，则问题A也一定是多项式时间可判定的
- 要证明一个判定问题B是NP-C的，除了要证明B是多项式时间可验证的（从而B属于NP类），还要找一个NP-C问题A，证明A可以在多项式时间里变换为B，且A的任一实例回答为‘Yes’ iff 与之对应的B的实例回答为‘Yes’

NP完全问题的实例


CIRCUIT-SAT
SAT
3-CNF-SAT
CLIQUE
SUBSET-SUM
VERTEX-COVER
HAM-CYCLE
TSP

k-团问题

- 给定一个无向图G=(V，E)和一个正整数k，判定图G是否包含一个k团
- 证明思路
  - 3 SAT是NP完全问题，证明3 SAT≤p CLIQUE（k-团问题）

Vertex Cover，顶点覆盖

- 顶点覆盖的最优化问题：在一个无向图G中，找一个顶点数最少的顶点集，满足：任一条边的两个顶点中至少有一个在此集合中
- 顶点覆盖的判定问题：无向图G中是否存在顶点数为k的顶点覆盖？
- 证明思路
  - 先证明其属于NP，再证明CNF-SAT≤pVC

子集和问题(Subset-Sum)

- 有一个数集$A=\{a_1,a_2,\ldots,a_n\}$及一个目标数S，问A中是否能找出一个子集A’，使得A’中元素之和为S？
- 证明思路
  - 首先证明其是NP问题， 再证明3CNF-SAT$\leq_p$ Subset-Sum

限制法

- 要证明问题$\pi \in$NP-C，先证明$\pi \in$NP，再找一个已知的NP-C问题π'，证明π'是π的特例（在π上加了一些限制），从而立即有$\pi' \leq_p \pi$

击中集（Hitting Set，HS）

- 设F是一个由S的某些子集构成的集合族。给定正整数k，问是否存在S的子集S’满足S’属于S，|S’|≤k，使得S’包含F中每一个子集里的至少一个元素？（满足条件的子集S’称为F的击中集。）

背包问题（Knapsack）

- 有物品$u_1,u_2,\ldots,u_n$，大小为$S_1, S_2,\ldots, S_n$，价值为$C_1, C_2,\ldots, C_n$，有一个背包，其容量B $< \sum_{i=1}^{n} S_i$，问如何选取物品，使得背包中装入物品的价值总和为最大？
- 相应的判定问题
  - 给定一个正整数k，问是否存在一种选取方法$\{i_1,i_2,\ldots,i_p\}$=A’ $\{1,2,\ldots,n\}$，使得$\sum_{j=1}^{p} S_{ij} \leq$B，而$\sum_{j=1}^{p} C_{ij} \geq$k？

NP-hard类问题的求解方法

- 当n不太大时，可使用a)动态规划法 b)分枝限界法 c)回溯法
- 求近似解。在误差允许的范围之内找一个解，该近似解可以在多项式时间里得到
- 用启发式算法求解，根据具体问题设计启发式搜索策略，在理论上往往缺乏严格的证明，用实验数据说明算法很有效
- 智能优化算法，常常能获得很好的结果。但有偶然性，与最优解的误差难以给出