浙江大学 ZheJiang University

组合优化

浙江大学数学系　谈之奕

浙江大学
ZheJiang University

计算复杂性初步

3SAT

浙江大学 ZheJiang University

组合优化

- **3SAT：在SAT问题的描述中，限定任一子句包含文字数不超过3**

任取SAT问题的一实例

$$
\begin{aligned}
&(l_{i_{11}} \vee l_{i_{12}} \vee \cdots \vee l_{i_{1l_1}}) \\
&\wedge (l_{i_{21}} \vee l_{i_{22}} \vee \cdots \vee l_{i_{2l_2}}) \\
&\wedge \cdots \wedge (l_{i_{k1}} \vee l_{i_{k2}} \vee \cdots \vee l_{i_{kl_k}})
\end{aligned}
$$

任取3SAT问题的一实例

$$
\begin{aligned}
&(l_{i_{11}} \vee l_{i_{12}} \vee l_{i_{13}}) \\
&\wedge (l_{i_{21}} \vee l_{i_{22}} \vee l_{i_{23}}) \\
&\wedge \cdots \wedge (l_{i_{k1}} \vee l_{i_{k2}} \vee l_{i_{k3}})
\end{aligned}
$$

- **2SAT** $\in \mathcal{P}$

Krom MR. The Decision Problem for a Class of First-Order Formulas in which all Disjunctions are Binary. *Mathematical Logic Quarterly*, 13, 15-20, 1967.

$\mathrm{SAT} \le_m^p \mathrm{3SAT}$

浙江大学 ZheJiang University

组合优化

- 任取SAT问题一实例 $F_{SAT} = c_1 \wedge c_2 \wedge \cdots \wedge c_m$，构造3SAT问题的实例 F_{3SAT}
  - 若 c_i 中所含文字数不超过3，则 F_3 也包含子句 c_i
  - 若 $c_i = l_{i_1} \vee l_{i_2} \vee \cdots \vee l_{i_k}$，则令

$$C_i = (l_{i_1} \vee l_{i_2} \vee y_{i1}) \wedge (\neg y_{i1} \vee l_{i_3} \vee y_{i2}) \wedge (\neg y_{i2} \vee l_{i_4} \vee y_{i3})$$

$$\wedge \cdots \wedge (\neg y_{i,k-4} \vee l_{i_{k-2}} \vee y_{i,k-3}) \wedge (\neg y_{i,k-3} \vee l_{i_{k-1}} \vee l_{i_k})$$

其中 $y_{i1}, y_{i2} \cdots y_{i,k-3}$ 是新变量且不出现在别处。F_3 中包含 C_i 中所有子句

$\mathrm{SAT} \le_m^p \mathrm{3SAT}$

浙江大学 ZheJiang University

组合优化

- 若 F_{SAT} 答案为“是”，则存在一种赋值，使得任一子句 c_i 为真，因此 c_i 中文字至少有一个为真，不妨设为 l_{i_j}
- 存在一种赋值，使得 F_{3SAT} 中每个子句均为真，F_{3SAT} 答案也为“是”

$$c_i = l_{i_1} \vee l_{i_2} \vee \cdots \vee \overset{1}{l_{i_j}} \vee \cdots \vee l_{i_k}$$

F_{SAT} 中原有变量赋值保持不变

$$\begin{aligned}
C_i = {} & (l_{i_1} \vee l_{i_2} \vee \overset{1}{y_{i1}}) \wedge (\neg \overset{0}{y_{i1}} \vee l_{i_3} \vee \overset{1}{y_{i2}}) \wedge \cdots \wedge (\neg \overset{0}{y_{i,j-4}} \vee l_{i_{j-2}} \vee \overset{1}{y_{i,j-3}}) \\
& \wedge (\neg \overset{0}{y_{i,j-3}} \vee l_{i_{j-1}} \vee \overset{1}{y_{i,j-2}}) \wedge (\neg \overset{0}{y_{i,j-2}} \vee \overset{1}{l_{i_j}} \vee \overset{0}{y_{i,j-1}}) \wedge (\neg \overset{1}{y_{i,j-1}} \vee l_{i_{j+1}} \vee \overset{0}{y_{i,j}}) \\
& \wedge (\neg \overset{1}{y_{i,j}} \vee l_{i_{j+2}} \vee \overset{0}{y_{i,j+1}}) \wedge \cdots \wedge (\neg \overset{1}{y_{i,k-4}} \vee l_{i_{k-2}} \vee \overset{0}{y_{i,k-3}}) \wedge (\neg \overset{1}{y_{i,k-3}} \vee l_{i_{k-1}} \vee l_{i_k})
\end{aligned}$$

$\text{SAT} \le_m^p \text{3SAT}$

组合优化

- 若F_{3SAT} 答案为“是”，则存在一种赋值，使得 F_{3SAT} 中每个子句均为真
- 将该赋值限制到 F_{SAT} 中的变量，若存在一子句 c_i 为假，则 c_i 中所有文字均为假
- C_i中有子句值为假，矛盾，故c_i中所有子句均为真，F_{SAT} 答案也为“是”

$$c_i = \overset{0}{l_{i_1}} \vee \overset{0}{l_{i_2}} \vee \cdots \vee \overset{0}{l_{i_j}} \vee \cdots \vee \overset{0}{l_{i_k}}$$

$$C_i = (\overset{0}{l_{i_1}} \vee \overset{0}{l_{i_2}} \vee \overset{1}{y_{i1}}) \wedge (\overset{0}{\neg y_{i1}} \vee \overset{0}{l_{i_3}} \vee \overset{1}{y_{i2}}) \wedge (\overset{0}{\neg y_{i2}} \vee \overset{0}{l_{i_4}} \vee \overset{1}{y_{i3}}) \wedge \cdots$$

$$\wedge \cdots \wedge (\overset{0}{\neg y_{i,k-5}} \vee \overset{0}{l_{i_{k-3}}} \vee \overset{1}{y_{i,k-4}}) \wedge (\overset{0}{\neg y_{i,k-4}} \vee \overset{0}{l_{i_{k-2}}} \vee \overset{1}{y_{i,k-3}}) \wedge (\overset{0}{\neg y_{i,k-3}} \vee \overset{0}{l_{i_{k-1}}} \vee \overset{0}{l_{i_k}})$$

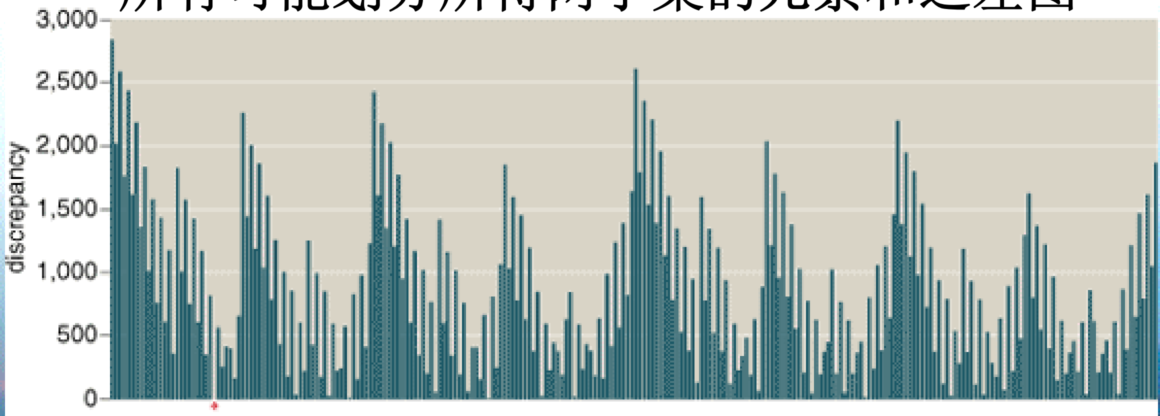
划分问题

- 划分问题（**Partition**）
  - 给定一正整数集 $A=\{a_1,a_2,\cdots,a_n\}$，问是否存在子集 A_1, A_2，使得 $A=A_1\cup A_2$，$A_1\cap A_2=\phi$，且满足 $\sum_{a_i\in A_1} a_i=\sum_{a_i\in A_2} a_i=\frac{1}{2}\sum_{j=1}^{n} a_j$

$$A=\{484,114,205,288,506,503,201,127,410\}$$

$$A_1=\{410,506,503\}, A_2=\{484,114,205,288,201,127\}$$

所有可能划分所得两子集的元素和之差图



COMPUTING SCIENCE

THE EASIEST HARD PROBLEM

Brian Hayes

One of the cherished customs of childhood is choosing up sides for a ball game. Where I grew up, we did it this way: The two chief bullies of the neighborhood would appoint themselves captains of the opposing teams, and then they would take turns picking other players. On each round, a captain would choose the most capable (or, toward the end, the least inept) player from the pool of remaining candidates, until everyone present had been assigned to one side or the other. The aim of this ritual was to produce two evenly matched teams and, along the way, to remind each of us of our precise ranking in the neighborhood pecking order. It usually worked.

None of us in those days—not the hopefuls waiting for our name to be called, and certainly not the two thick-necked team leaders—recognized that our scheme for choosing sides implements a greedy heuristic for the balanced number partitioning problem. And we had no idea that this problem is NP-complete—that finding the optimum team rosters is certifiably hard. We just wanted to get on with the game.

And therein lies a paradox: If computer scientists find the partitioning problem so intractable, how come children the world over solve it every day? Are the kids *that* much smarter? Quite possibly they are. On the other hand, the success of playground algorithms for partitioning might be a clue that the task is not always as hard as that forbidding term "NP-complete" tends to suggest. As a matter of fact, finding a hard instance of this famously hard problem can be a hard problem—unless you know where to look. Some recent re-

jobs into two sets with equal running time will balance the load on the processors. Another example is apportioning the miscellaneous assets of an estate between two heirs.

So What's the Problem?

Here is a slightly more formal statement of the partitioning problem. You are given a set of *n* positive integers, and you are asked to separate them into two subsets; you may put as many or as few numbers as you please in each of the subsets, but you must make the sums of the subsets as nearly equal as possible. Ideally, the two sums would be exactly the same, but this is feasible only if the sum of the entire set is even; in the event of an odd total, the best you can possibly do is to choose two subsets that differ by 1. Accordingly, a perfect partition is defined as any arrangement for which the "discrepancy"—the absolute value of the subset difference—is no greater than 1.

Try a small example. Here are 10 numbers—enough for two basketball teams—selected at random from the range between 1 and 10:

2 10 3 8 5 7 9 5 3 2

Can you find a perfect partition? In this instance it so happens there are 23 ways to divvy up the numbers into two groups with exactly equal sums (or 46 ways if you count mirror images as distinct partitions). Almost any reasonable method will converge on one of these perfect solutions. This is the answer I stumbled onto first:

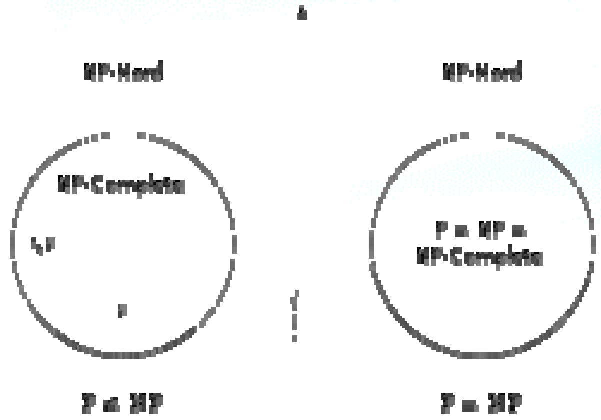
(2 5 3 10 7) (2 5 3 9 8)

Both subsets sum to 27.

Hayes B. The easiest hard problem. *American Scientist*, 90(2), 113-117, 2002.

$\mathcal{NP}$ –难

- 至少和$\mathcal{NP}$ –完全问题一样难的问题称为$\mathcal{NP}$ –难（$\mathcal{NP}$ –hard）问题
  - 在 $\mathcal{P} \neq \mathcal{NP}$ 的假设下，$\mathcal{NP}$ –难问题也没有多项式时间算法
  - 若一优化问题的判定形式是$\mathcal{NP}$ –完全的，则该优化问题是$\mathcal{NP}$ –难的



子问题

组合优化

- 在某问题Π的实例构成上增加一些限制就得到的一个子问题（**subproblem**）Π'
  - Π'的实例集包含在Π的实例集中，Π'的答案为“是”（“否”）的实例集恰为Π的答案为“是”（“否”）的实例集与Π'的实例集之交
- 子问题的计算复杂性
  - 若$\Pi\in\mathcal{P}$，则$\Pi'\in\mathcal{P}$，但反之不然
  - 若Π'是$\mathcal{NP}$-完全的，则Π也是$\mathcal{NP}$-完全的，但反之不然
  - 希望寻找“最特殊”的$\mathcal{NP}$–完全子问题和“最一般”的$\mathcal{P}$子问题

子集和问题

- 子集和问题（**Subset Sum**）
  - 给定正整数集 $A=\{a_1,a_2,\cdots,a_n\}$和数 B，问是否存在子集$A_1 \subseteq A$，使得 $\sum_{a_i \in A_1} a_i = B$
  - 取 $B=\frac{1}{2}\sum_{j=1}^{n} a_j$，划分问题成为子集和问题的子问题
- 子集和问题的优化形式

子集和是$\mathcal{NP}$ 一完全问题

  - 求子集 $A_1 \subseteq A$，使得$\sum_{a_i \in A_1} a_i \le B$且$\sum_{a_i \in A_1} a_i$尽可能大
  - 若背包问题物品 j的价值与大小均为a_j，容量为 B，则子集和问题优化形式成为背包问题优化形式的子问题

背包问题判定形式是$\mathcal{NP}$ 一完全问题

子集和问题

组合优化

- 子集和问题的优化形式（极小化）　　极小化背包的定义与应用
  - 求子集 $A_1 \subseteq A$，使得 $\sum_{a_i \in A_1} a_i \geq B$ 且 $\sum_{a_i \in A_1} a_i$ 尽可能小
- 第 K 个最大子集和问题
  - 给定正整数集 $A=\{a_1,a_2,\cdots,a_n\}$ 和数 B，问是否存在 A 的 K 个子集 $A_1,A_2,\cdots,A_K$，使得 $\sum_{a_i \in A_i} a_i \leq B, i=1,\cdots,K$
  - 第 K 个最大子集和问题可能不属于 $\mathcal{NP}$
    - 实例规模：$n+\max_{i=1,\cdots,n} \log a_i + \log B + \log K$
    - 可行解的规模：Kn

强$\mathcal{NP}$-完全

组合优化

- 设Π为一$\mathcal{NP}$-完全问题，且存在多项式函数p，使得Π的某个所有实例满足$\mathrm{Max}(I) \le p(\mathrm{size}(I))$的子问题$\Pi$'是$\mathcal{NP}$-完全的，则称$\Pi$为强$\mathcal{NP}$-完全（**strongly $\mathcal{NP}$-complete**）的
- 在$\mathcal{P} \ne \mathcal{NP}$的假设下，任何强$\mathcal{NP}$-完全问题不存在伪多项式时间算法
  - 若Π存在伪多项式时间算法A，其时间复杂性为

$$f(I) = O(\mathrm{poly}(\mathrm{size}(I), \mathrm{Max}(I)))$$

  - 用A求解Π的子问题Π'，其时间复杂性为

$$f(I) = O(\mathrm{poly}(\mathrm{size}(I), \mathrm{Max}(I))) = O(\mathrm{poly}(\mathrm{size}(I), p(\mathrm{size}(I)))) = O(\mathrm{poly}(\mathrm{size}(I)))$$

  - A为Π'的多项式时间算法，与Π'的$\mathcal{NP}$-完全性矛盾

强$\mathcal{NP}$−完全性的证明

- 问题是$\mathcal{NP}$−完全的，且其所有实例均满足 $\mathrm{Max}(I) \le p(\mathrm{size}(I))$
  - **Hamilton**图问题、**SAT**问题
- 归约证明某问题的$\mathcal{NP}$−完全性时所构造的该问题的实例均满足 $\mathrm{Max}(I) \le p(\mathrm{size}(I))$
  - **TSP**问题的判定形式
- 按强$\mathcal{NP}$−完全问题的定义证明
  - 寻找多项式函数 p 和所有实例满足 $\mathrm{Max}(I) \le p(\mathrm{size}(I))$ 的子问题，通过归约证明该子问题是$\mathcal{NP}$−完全的
- 用伪多项式时间归约从一个已知强$\mathcal{NP}$−完全问题出发证明某问题的强$\mathcal{NP}$−完全性

伪多项式时间归约

浙江大学 ZheJiang University

组合优化

- 设有判定问题 Π_1, Π_2，若对 Π_1 的任一实例 I_1，可在 $O(\mathrm{poly}(\mathrm{size}(I_1), \mathrm{Max}(I_1)))$ 时间内构造出 Π_2 的一个实例 $I_2 = f(I_1)$，使得 实例构造的时间不太长
  - I_1 的答案为“是”当且仅当 I_2 的答案也为“是” 可以反馈
  - $\mathrm{Max}(I_2) \le p_1(\mathrm{size}(I_1), \mathrm{Max}(I_1))$，这里 p_1 为某个二元多项式 最大数没有显著增大
  - $\mathrm{size}(I_1) \le p_2(\mathrm{size}(I_2))$，这里 p_2 为某个多项式 实例规模没有显著缩小

则称 Π_1 可伪多项式时间归约到 Π_2，记为 $\Pi_1 \le_m^{pp} \Pi_2$

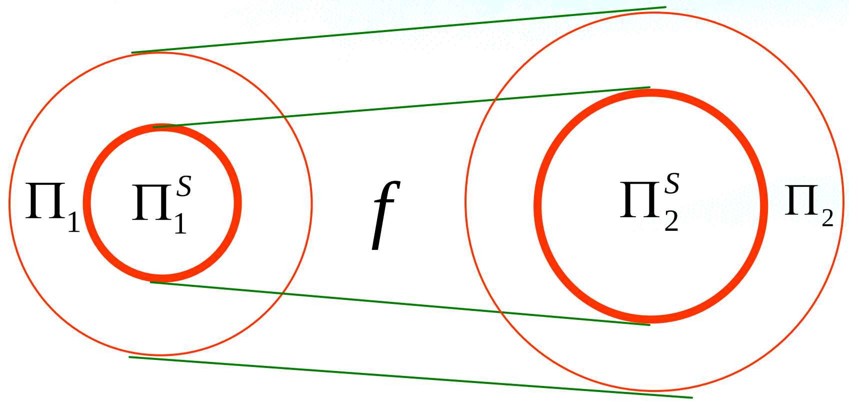
伪多项式时间归约

- 若Π_1为强$\mathcal{NP}$−完全问题，$\Pi_2 \in \mathcal{NP}$且$\Pi_1 \le_m^{pp} \Pi_2$，则Π_2也是强$\mathcal{NP}$−完全问题
  - 由于Π_1为强$\mathcal{NP}$−完全问题，存在Π_1的子问题Π_1^S，其实例集$\mathcal{I}_1$中所有实例I_1满足$\mathrm{Max}(I_1) \le p(\mathrm{size}(I_1))$，且$\Pi_1^S$是$\mathcal{NP}$−完全的
  - 将以$\mathcal{I}_2 = f(\mathcal{I}_1)$为实例集的$\Pi_2$的子问题记为$\Pi_2^S$
  - 由于构造$f(\mathcal{I}_1)$可在$O(\mathrm{poly}(\mathrm{size}(I_1), \mathrm{Max}(I_1))) = O(\mathrm{poly}(\mathrm{size}(I_1)))$时间内完成，$f$也是$\Pi_1^S$到$\Pi_2^S$的多项式时间归约，即$\Pi_1^S \le_m^p \Pi_2^S$，因此$\Pi_2^S$是$\mathcal{NP}$−完全的
  - 若能证明$\mathcal{I}_2$中所有实例I_2满足$\mathrm{Max}(I_2) \le p'(\mathrm{size}(I_2))$，则由强$\mathcal{NP}$−完全问题的定义可知$\Pi_2$是强$\mathcal{NP}$−完全的

伪多项式时间归约

组合优化

$$
\begin{aligned}
\mathrm{Max}(I_2) &\le p_1(\mathrm{size}(I_1), \mathrm{Max}(I_1)) \\
&\le p_1(\mathrm{size}(I_1), p(\mathrm{size}(I_1))) \\
&\le p_1'(\mathrm{size}(I_1)) \\
&\le p_1'(p_2(\mathrm{size}(I_2))) \\
&\le p'(\mathrm{size}(I_2))
\end{aligned}
$$



$\mathrm{Max}(I_1) \le p(\mathrm{size}(I_1))$ $\mathrm{Max}(I_2) \le p'(\mathrm{size}(I_2))$?

$$
\boxed{\begin{gathered}
\mathrm{Max}(I_2) \le p_1(\mathrm{size}(I_1), \mathrm{Max}(I_1)) \\
\mathrm{size}(I_1) \le p_2(\mathrm{size}(I_2))
\end{gathered}}
$$

上述两个条件容易验证，实践中多数构造都能满足

若 $\mathrm{size}(I_1) = \Omega(2^{\mathrm{size}(I_2)})$，即 $\mathrm{size}(I_2) = O(\log(\mathrm{size}(I_1)))$，构造的实例规模显著缩小，从而 $\mathrm{Max}(I_2) \le p'(\mathrm{size}(I_2))$ 未必成立，未能找到满足要求的 $\mathcal{NP}$−完全的子问题

第一个强 $\mathcal{NP}$－完全问题

3-划分问题

- **3-划分（3-partioning）**
  - 给定由 $3m$ 个正整数组成的集合 $A=\{a_1,a_2,\cdots,a_{3m}\}$ 和正整数 B，其中 $\frac{B}{4}<a_j<\frac{B}{2}, j=1,2,\cdots,3m, \sum_{j=1}^{3m}a_j=mB$，问 A 是否可划分成 m 个互不相交的集合 $A_1,A_2,\cdots,A_m$，使得对任意的 $i, 1\le i\le m, \sum_{a_j\in A_i}a_j=B$

6 8 10 17 19

3 5 7 9 11 12 13

划分

10 13 17

3 7 11 19

5 6 8 9 12

划分

103 108 119

105 112 113

106 107 117

109 110 111

3-划分

普通$\mathcal{NP}$-完全

- 若Π为$\mathcal{NP}$-完全问题，且存在伪多项式时间算法，则Π不可能是强$\mathcal{NP}$-完全的，此时称Π为普通意义下的$\mathcal{NP}$-完全问题（$\mathcal{NP}$-**complete in the ordinary sense**）
  - 背包、划分均为普通意义下的$\mathcal{NP}$-完全问题
- 至少和强$\mathcal{NP}$-完全问题一样难的问题称为强$\mathcal{NP}$-难（**strongly** $\mathcal{NP}$-**hard**）问题
  - **TSP**为强$\mathcal{NP}$-难问题

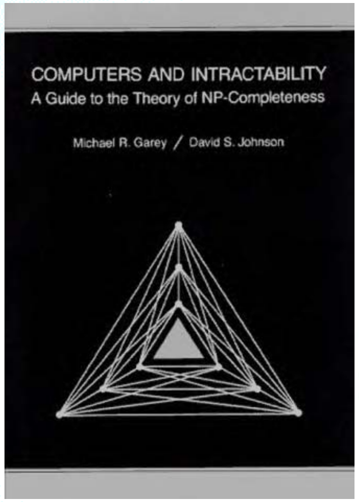
空间复杂性

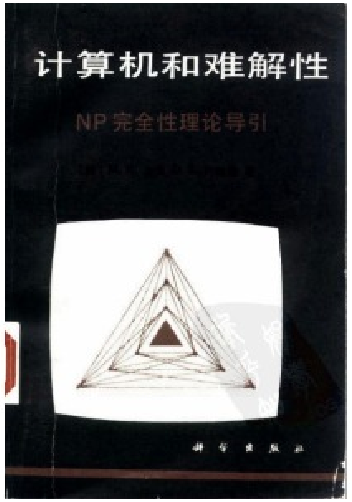
- 算法的空间复杂度（**space complexity**）是关于实例规模 n 的一个函数 $f(n)$，它表示用该算法求解所有规模为 n 的实例中算法使用空间量最多的那个实例算法使用的空间量
- 空间复杂度类
  - $\mathcal{PSPACE}$，$\mathcal{NPSPACE}$，$\mathcal{EXPSPACE}$，$\mathcal{PSPACE}$ -完全

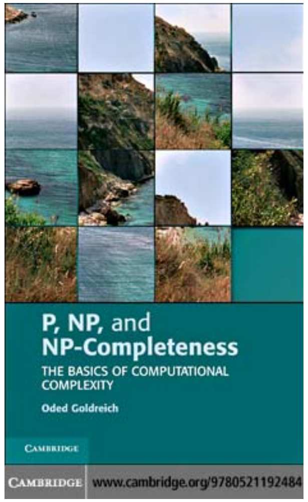
$$\mathcal{P} \underset{?}{\subsetneq} \mathcal{NP} \underset{?}{\subsetneq} \mathcal{PSPACE} \underset{?}{\subsetneq} \mathcal{EXP} \underset{?}{\subsetneq} \mathcal{NEXP} \underset{?}{\subsetneq} \mathcal{EXPSPACE}$$

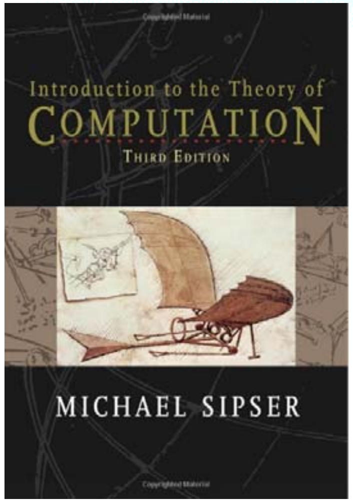
$$\mathcal{PSPACE} = \mathcal{NPSPACE}$$

参考资料







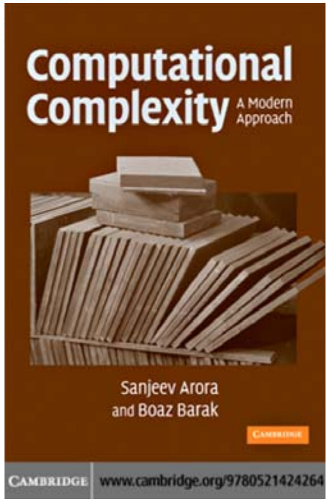


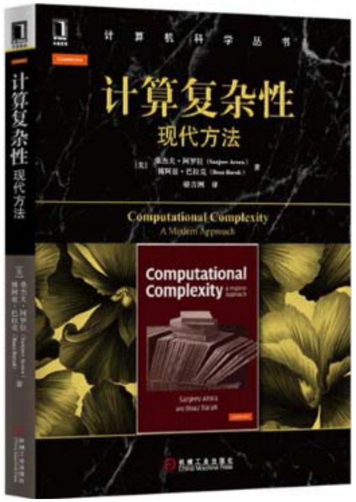
Garey MR, Johnson DS. *Computers and intractability: a guide to the theory of NP-completeness*. Freeman, 1979.（中译本：计算机和难解性：NP 完全性理论导引．张立昂，沈泓译，科学出版社，1990.）

Goldreich O. *P, NP, and NP-Completeness: The basics of Computational Complexity*. Cambridge University Press, 2010.

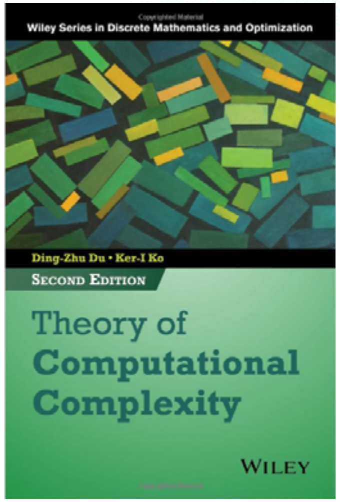
Sipser M. *Introduction to the Theory of Computation* (3rd). Cengage Learning, 2012. (中译本：计算理论导引. 段磊、唐常杰译, 机械工业出版社，**2015**）

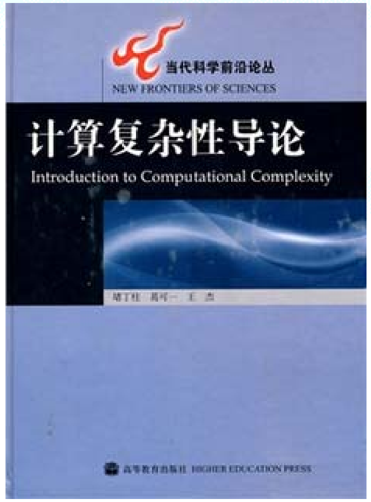
参考资料





Arora S, Barak B. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.（中译本：计算复杂性：现代方法. 骆吉洲译，机械工业出版社，2015.）





Du DZ, Ko KI. *Theory of Computational Complexity*. Wiley, 2011.
堵丁柱，葛可一，王杰，计算复杂性导论，高等教育出版社，2002.

$\mathcal{NP}$ 优化问题汇编

浙江大学 ZheJiang University

组合优化

A compendium of NP optimization problems

Editors:
[Pierluigi Crescenzi], and [Viggo Kann]
Subeditors:
Magnús Halldórsson (retired)
Graph Theory: Covering and Partitioning, Subgraphs and Supergraphs, *Sets and Partitions*.
Marek Karpinski
Graph Theory: Vertex Ordering, *Network Design*: Cuts and Connectivity.
Gerhard Woeginger
Sequencing and Scheduling.

This is a continuously updated catalog of approximability results for NP optimization problems. The compendium is also a part of the book Complexity and Approximation. The compendium has not been updated for a while, so there might exist recent results that are not mentioned in the compendium. If you happen to notice such a missing result, please report it to us using the web forms.

You can use web forms to report new problems, new results on existing problems, updates of references or errors.

http://www.nada.kth.se/~viggo/problemlist/compendium.html

$\mathcal{NP}$ 优化问题汇编

MAXIMUM KNAPSACK

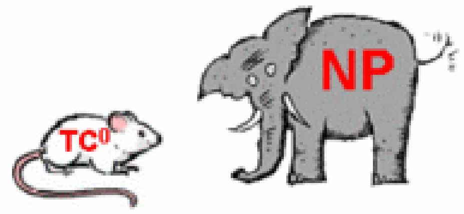
- INSTANCE: Finite set U, for each $u \in U$ a size $s(u) \in Z^+$ and a value $v(u) \in Z^+$, a positive integer $B \in Z^+$.
- SOLUTION: A subset $U' \subseteq U$ such that $\sum_{u \in U'} s(u) \leq B$.
- MEASURE: Total weight of the chosen elements, i.e., $\sum_{u \in U'} v(u)$.

- *Good News*: Admits an FPTAS [266].
- *Comment*: The special case when $s(u)=v(u)$ for all $u \in U$ is called MAXIMUM SUBSET SUM. The corresponding minimization problem where $\sum_{u \in U'} s(u) \geq B$ also admits an FPTAS, as well as several other variations of the knapsack problem [191].
- *Garey and Johnson*: MP9

Complexity Zoo

组合优化

- **There are now 535 classes and counting**



All Classes

Complexity classes by letter: Symbols - A - B - C - D - E - F - G - H - I - J - K - L - M - N - O - P - Q - R - S - T - U - V - W - X - Y - Z

Lists of related classes: Communication Complexity - Hierarchies - Nonuniform

Symbols

$0\text{-}1\text{-}NP_C$ - $1NAuxPDA^p$ - 2-EXP - 3SUM-hard - $\#AC^0$ - #L - #L/poly - #GA - #P - #W[t] - ⊕EXP - ⊕L - ⊕L/poly - ⊕P - $\oplus SAC^0$ - $\oplus SAC^1$

A

A_0PP - AC - AC^0 - $AC^0[m]$ - AC^1 - ACC^0 - AH - AL - ALL - ALOGTIME - AlgP/poly - Almost-NP - Almost-P - Almost-PSPACE - AM - AM_{EXP} - AM ∩ coAM - AM[polylog] - AmpMP - AmpP-BQP - AP - APP - APX - ATIME - AUC-SPACE(f(n)) - AuxPDA - AVBPP - AvgE - AvgP - AW[P] - AWPP - AW[SAT] - AW[*] - AW[t] - AxP - AxPP

B

βP - BH - $BP_d(P)$ - BPE - BPEE - $BP_HSPACE(f(n))$ - BPL - BP•NP - BPP - BPP^{cc} - BPP_k^{cc} - BPP^{KT} - BPP/log - BPP/mlog - BPP//log - BPP/rlog - BPP-OBDD - BPP_{path} - BPQP - BPSPACE(f(n)) - BPTIME(f(n)) - BQNC - BQNP - BQP - BQP/log - BQP/poly - BQP/mlog - BQP/mpoly - BQP/qlog - BQP/qpoly - BQP-OBDD - BQPSPACE - BQP_{CTC} - $BQP_{tt}/poly$ - BQTIME(f(n)) - k-BWBP

NL: Nondeterministic Logarithmic-Space

Has the same relation to L as NP does to P.

In a breakthrough result, was shown to equal coNL [Imm88] [Sze87]. (Though contrast to mNL.)

Is contained in LOGCFL [Sud78], as well as NC^2.

Is contained in UL/poly [RA00].

Deciding whether a bipartite graph has a perfect matching is hard for NL [KUW86].

NL can be defined in a logical formalism as SO(krom) and also as FO(tc), reachability in directed graph is NL-Complete under FO-reduction.

https://complexityzoo.uwaterloo.ca/

线性规划和
整数规划简介

数学规划

组合优化

- 若干个变量在满足一些等式或不等式限制条件下，使一个或多个目标函数取得最大值或最小值
  - 满足所有约束条件的点称为可行点（解）（**feasible point**），可行点的集合称为可行域（**feasible region**），记为 S
  - 可行解 $\mathbf{x}^*$ 称为一（极小化）数学规划问题的最优解（**optimal solution**），若对任意 $\mathbf{x}\in S$，$f(\mathbf{x}^*)\le f(\mathbf{x})$；相应地 $f(\mathbf{x}^*)$ 称为最优值
- 单目标数学规划

$$
\begin{aligned}
\min \quad & f(\mathbf{x}) \\
s.t. \quad & g_j(\mathbf{x})\ge 0 \quad j=1,\cdots,s \\
& h_l(\mathbf{x})=0 \quad l=1,\cdots,t \\
& \mathbf{x}\in\mathbb{R}^n
\end{aligned}
$$

目标函数

min 极小
max 极大

subject to:
以下为约束条件

变量取值范围约束

等式约束

不等式约束

数学规划分类

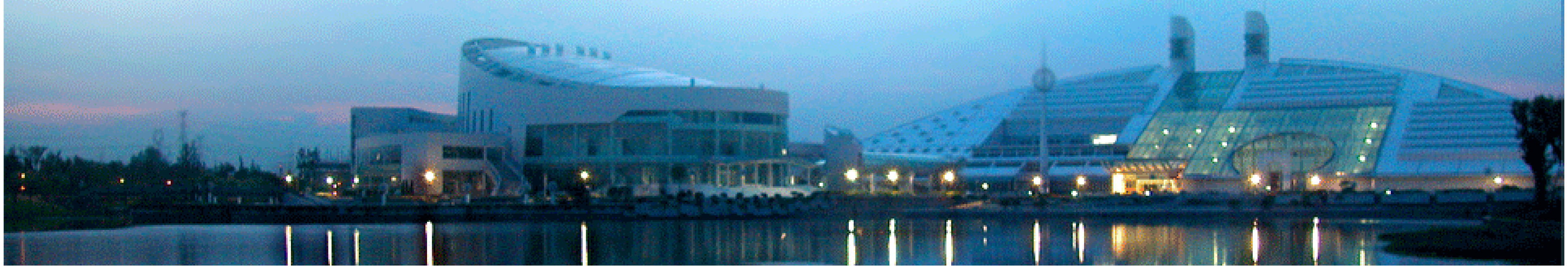
组合优化

- 线性规划与非线性规划
  - 线性规划：f, g_i, h_j 均为线性函数
  - 非线性规划：f, g_i, h_j 至少有一个是非线性函数
- 整数规划：至少有一个决策变量限定取整数值
  - 混合整数规划（**Mixed Integer Programming, MIP**）：部分决策变量取整数值
  - **0-1**规划：所有决策变量都取 **0** 或 **1**

$$\begin{aligned} \min \quad & f(\mathbf{x}) \\ s.t. \quad & g_j(\mathbf{x}) \geq 0 \quad j = 1, \cdots, s \\ & h_l(\mathbf{x}) = 0 \quad l = 1, \cdots, t \end{aligned}$$

$$x_i \in \mathbb{Z}$$

$$x_i \in \{0,1\}$$



线性规划

浙江大学 ZheJiang University

组合优化

- 任何线性规划总可通过适当变形变为标准形标准型

$$
\begin{aligned}
\min\ & c_1x_1 + c_2x_2 + \cdots + c_nx_n \\
s.t.\ & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\
& a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\
& \cdots \\
& a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \\
& x_1, \cdots, x_n \geq 0
\end{aligned}
$$

$$
\begin{aligned}
\min\ & \mathbf{cx} \\
s.t.\ & \mathbf{Ax} = \mathbf{b} \\
& \mathbf{x} \geq \mathbf{0}
\end{aligned}
$$

价格系数向量

系数矩阵

右端向量

单纯形法

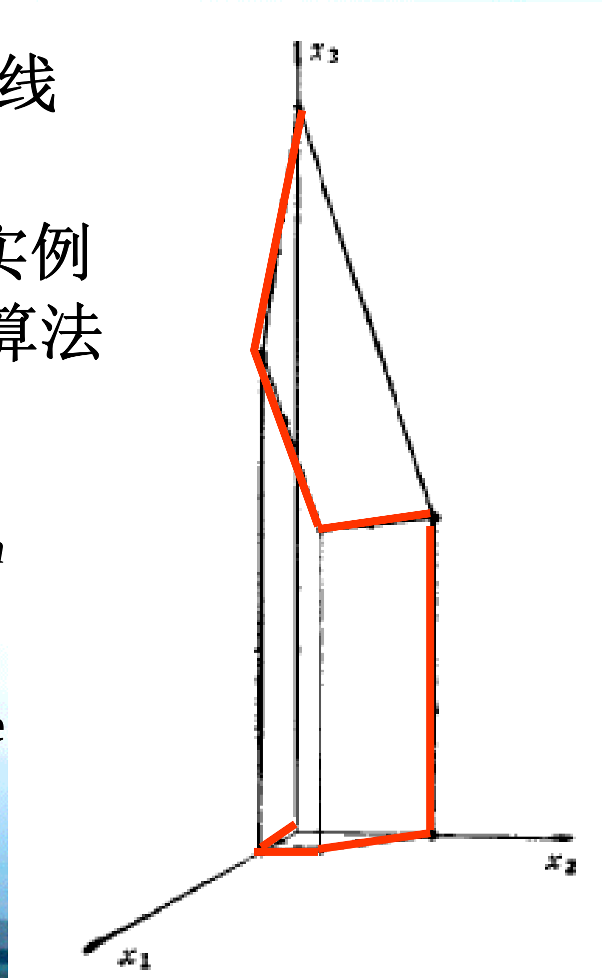
- **1947年Dantzig提出了求解线性规划的单纯形法**
- **1967年得到的Klee-Minty实例说明单纯形法是指数时间算法**

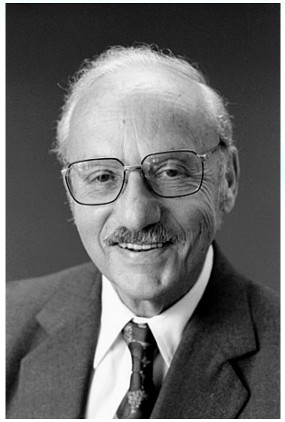
$$\max \sum_{i=1}^{m} 10^{m-i} x_i$$

$$s.t.\ \ 2\sum_{i=1}^{j-1} 10^{j-i} x_i + x_j \le 100^{j-1},\ j = 1,\ \cdots,\ m$$

$$x_i \ge 0,\ i = 1, \cdots, m$$

Klee V, Minty GJ, How good is the simplex algorithm? In *Inequalities – III* (Shisha O, Eds.), Academic Press, 159–175, 1972





George Bernard Dantzig (1914-2005)
美国运筹学家

多项式时间算法

组合优化

- 1979年，**Khachiyan** 给出了求解线性规划的第一个多项式时间算法——椭球法（**Ellipsoid algorithm**），说明线性规划是多项式时间可解的
- 1984年，**Karmarkar** 给出了实际效果更好的线性规划多项式时间算法——内点法（**Interior Point Method**），在数学规划领域产生了深远的影响

Narendra Karmarkar (1957-)
印度数学家

Leonid Genrikhovich Khachiyan (1952-2005)
苏联数学家

Khachiyan L, A polynomial algorithm in linear programming. *Doklady Akademiia Nauk SSSR*, 244, 1093-1096, 1979

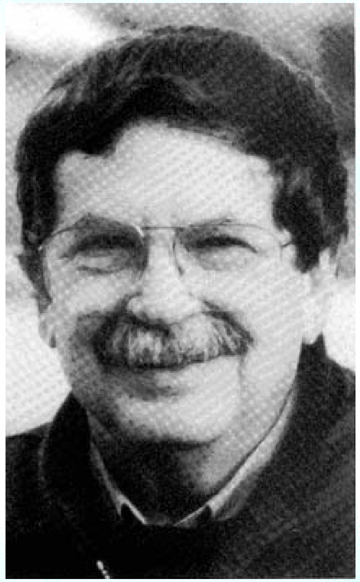
Karmarkar NK, A new polynomial-time algorithm for linear programming. *Combinatorica*, 4, 373–395, 1984

The Mathematical Sputnik

组合优化

- The New York Times of November 7, 1979 announced an event which its readers could easily believe had the importance of the launching of Sputnik. "A surprise discovery by an obscure Soviet mathematician has rocked the world of mathematics and computer analysis ... Apart from its profound theoretical interest… the theory of codes could eventually be affected by the Russian discovery, and this fact has obvious importance to intelligence agencies everywhere".
- In England, the Guardian broke the story three days earlier, under the headline, " Soviet Answer to 'Traveling Salesmen'. "

Lawler EL. The great mathematical Sputnik of 1979. *The Mathematical Intelligencer*, 2, 191-198, 1980.

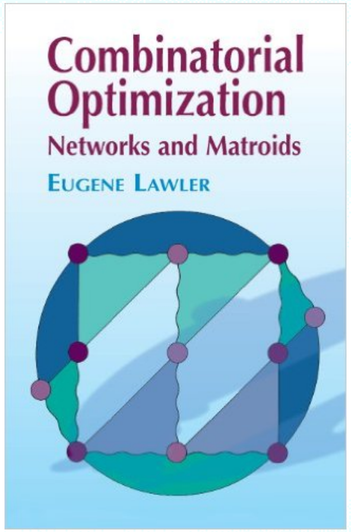


Eugene Leighton (Gene) Lawler (1933-1994)
美国运筹学家

The Mathematical Sputnik

组合优化

- Khachiyan emphatically did not discover a polynomial bounded algorithm for the TSP…What Khachiyan did do was to answer a much smaller question in complexity theory. Linear programming had been known to be a problem in $\mathcal{NP}$. It had not been shown to be $\mathcal{NP}$-complete…Khachiyan squeezed linear programming into $\mathcal{P}$ and thereby resolved the issue.
- Only much later, on March 21, did the Times print a retraction,…But the headline read " A Russian's Solution in Math Questioned" and the subhead read " Americans Who Studied Khachiyan Linear Programming Method Express Doubt on Scope. " That made it sound as though poor Khachiyan had exaggerated the importance of his work, and Western mathematicians had cut him down to size.



Lawler E. *Combinatorial Optimization: Networks and Matroids*, Dover Publications, 1976

整数线性规划的$\mathcal{NP}$ −完全性

组合优化

- 划分问题 $\le_m^p$ 整数线性规划
  - 任取划分问题的实例 $A=\{a_1,a_2,\cdots,a_n\}$，考虑整数规划 $\sum_{j=1}^{n} a_j x_j = \frac{1}{2}\sum_{j=1}^{n} a_j, x_j \in \{0,1\}$
  - 整数规划有可行解当且仅当划分问题实例答案为“是”
- 整数线性规划 $\in \mathcal{NP}$
  - 若线性不等式组 $\mathbf{Ax} \ge \mathbf{b}$ 有整数解，必存在一整数解，其规模不超过实例规模的多项式

Papadimitriou CH. On the complexity of integer programming. *Journal of the ACM*, 28, 765-768, 1981

整数线性规划算法

- **1958年，Gomory给出了求解整数线性规划的割平面法（cutting plane method）**
- **1960年，Land和Doig给出了求解整数规划的分支定界法（Branch and Bound ）**

Gomory RE, Outline of an Algorithm for Integer Solutions to Linear Programs, *Bulletin of the American Mathematical Society*, 64, 275–278, 1958

Land A, Doig A, An automatic method of solving discrete programming problems, *Econometrica* 28, 497–520, 1960

Ralph Edward Gomory (1929-)
美国运筹学家

左上:Ailsa Land
左下:Alison Doig

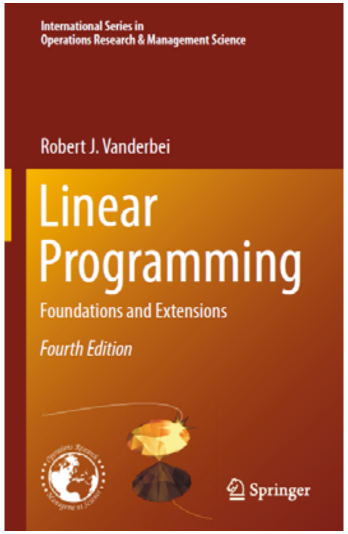
松弛

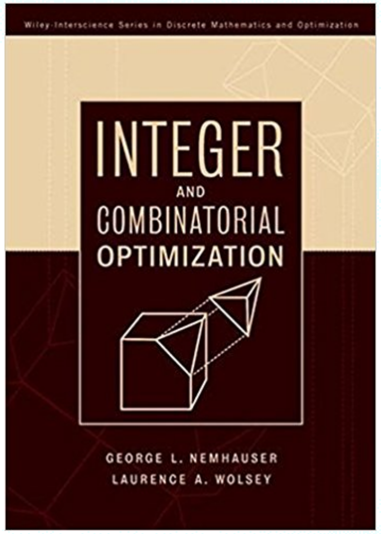
- 设有整数线性规划（**IP**），去除决策变量取整数约束后所得线性规划记为（**LP**），称（**LP**）为（**IP**）的松弛（**relaxation**）
  - （**IP**）的可行域包含于（**LP**）的可行域中
  - （**IP**）的可行解也是（**LP**）的可行解，但反之不然
  - （**IP**）的最优值不优于（**LP**）的最优值
  - 若（**LP**）的最优解为整数解，则它也是（**IP**）的最优解

$$
\begin{aligned}
& \min \ \ \mathbf{cx} \\
(\mathrm{IP}) \ \ & s.t. \ \ \mathbf{Ax} = \mathbf{b} \\
& \mathbf{x} \in \mathbb{Z}_+^n
\end{aligned}
$$

$$
\begin{aligned}
& \min \ \ \mathbf{cx} \\
(\mathrm{LP}) \ \ & s.t. \ \ \mathbf{Ax} = \mathbf{b} \\
& \mathbf{x} \in \mathbb{R}_+^n
\end{aligned}
$$

参考资料





黄红选，韩继业，数学规划，清华大学出版社，**2006**

Vanderbei RJ, ***Linear Programming: Foundations and Extensions*****, Springer, 2014**

Schrijver A. ***Theory of Linear and Integer Programming*****. Wiley, 1998.**

Wolsey LA, Nemhauser GL. ***Integer and Combinatorial Optimization*****, Wiley, 1999**


浙江大学
ZheJiang University
谢 谢