第10讲 问题的复杂度分析 (下)

罗国杰

gluo@pku.edu.cn

2025年春季学期

PKU-04833060

算法设计与分析『实验班』

复习：“对手论证”

https://en.wikipedia.org/wiki/Three-card_Monte#/media/File:Three_Card_Monte.jpg

几种选择问题的复杂度下界

问题	算法	最坏情况	问题下界	最优性
找最大	Findmax	$n-1$	$n-1$	最优
找最大最小	FindMaxMin	$\lceil 3n/2 \rceil - 2$	$\lceil 3n/2 \rceil - 2$	最优
找第二大	锦标赛	$n+\lceil \log n \rceil - 2$	$n+\lceil \log n \rceil - 2$	最优
找中位数	Select	$O(n)$	$3n/2-3/2$	阶最优
找第k小	Select	$O(n)$	$n+\min\{k,n-k+1\}-2$	阶最优

找最大最小问题的FindMaxMin算法和复杂度下界

输入：n 个数的数组 L

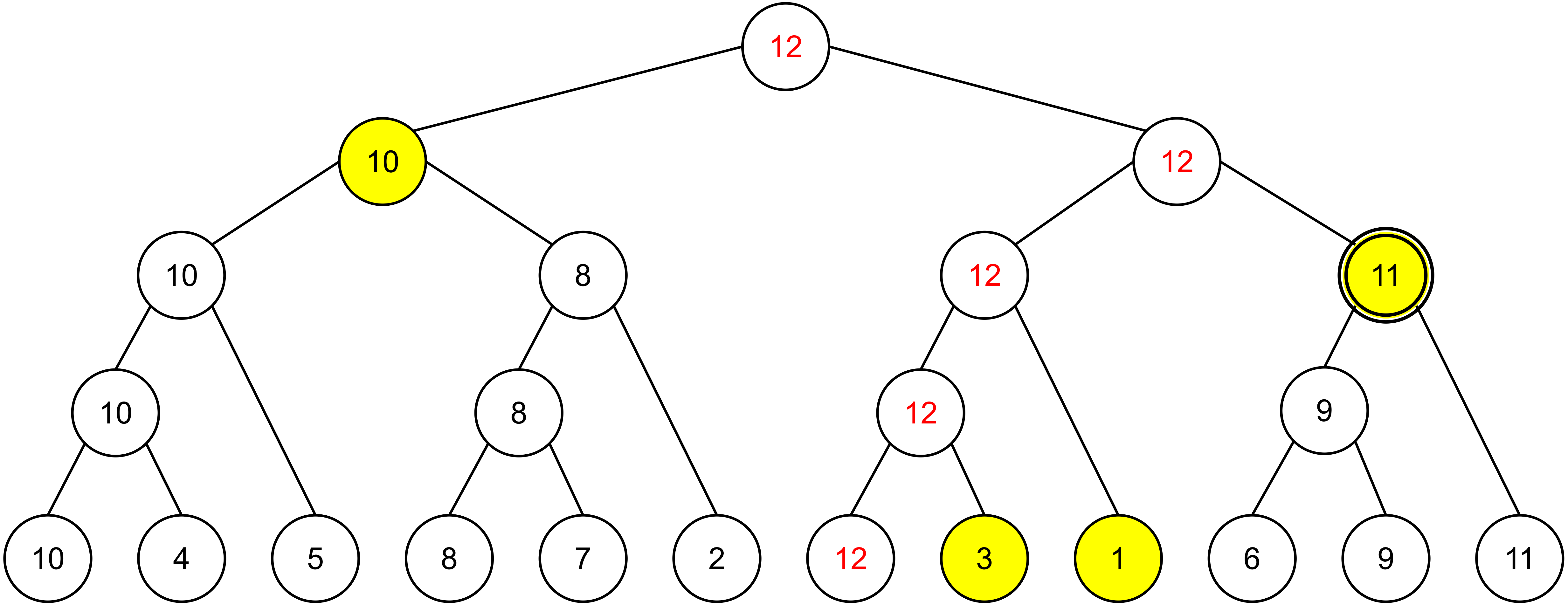
输出：max, min

1. 将 n 个元素两两一组分成 $\lfloor n/2 \rfloor$ 组
2. 每组比较，得到 $\lfloor n/2 \rfloor$ 个较小和 $\lfloor n/2 \rfloor$ 个较大
3. 在 $\lceil n/2 \rceil$ 个较小中找最小 min
4. 在 $\lceil n/2 \rceil$ 个较大中找最大 max

- 总比较次数：$W(n) = \lfloor n/2 \rfloor + 2 \times (\lceil n/2 \rceil - 1) = \lceil 3n/2 \rceil - 2$

- 对手论证思路：最大最小至少需要2(n-1)个信息，构造任意算法的“对手”，使最多$\lfloor n/2 \rfloor$次操作增加2个信息，至少还需2(n-1)-2$\lfloor n/2 \rfloor$次操作集齐2(n-1)个信息；总共至少需要$\lfloor n/2 \rfloor$+2(n-1)-2$\lfloor n/2 \rfloor$=$\lceil 3n/2 \rceil$-2次操作。

找第二大问题：锦标赛/胜者树算法



找第二大问题的复杂度下界

- 设 K 为直接与 *max* 结点的比较次数，则
- 确定最大需要淘汰 N-1 个元素
- 确定第二大需要淘汰 $K-1$ 个元素
- 至少用 $N+K$-2 次比较

- 对手论证思路：构造任意算法的“对手”，使 K 最大化。定义算法比较时两个数的赋值，使每次比较在胜者树中对应的子树尽可能平衡，从而 $K \geq \lceil \log n \rceil$。总共至少需要 $N+\lceil \log n \rceil-2$ 次比较。
- 结论：锦标赛算法使找第二大的最优算法。

找中位数问题：对手论证

定理 设n为奇数，任何通过比较运算找n个数的中位数(median)的算法在最坏情况下至少做 $3n/2-3/2$ 次比较

证 为找到中位数，必须要得到 $n-1$ 个信息单位：$(n-1)/2$ 个数比 median 大/小。

对手论证思路：针对算法构造输入，使得“非关键”的比较次数达到$(n-1)/2$次。

首先定义x与y关键的比较（获得信息单位）与非关键的比较.

关键的比较：建立 x 与 median 的关系的比较.

* 得到x比median大的信息：$\exists y$ ($x>y$ 且 $y \geq$ median)，x满足上述条件的第一次比较
* 得到x比median小的信息：$\exists y$ ($x<y$ 且 $y \leq$ median)，x满足上述条件的第一次比较

（比较时 y 与median的关系可以不知道）

非关键的比较：当$x>$median, $y<$median，这时$x>y$的比较不是关键的。

找中位数问题：对手论证的输入构造方法

1. 分配一个值给中位数 median；
2. 如果 A 比较 x 与 y，且 x 与 y 没有被赋值，那么赋值 x,y 使得 $x>\text{median}$, $y<\text{median}$;
3. 如果 A 比较 x 与 y, 且 $x>\text{median}$, y 没被赋值，则赋值 y 使得 $y<\text{median}$;
4. 如果 A 比较 x 与 y, 且 $x<\text{median}$, y 没被赋值，则赋值 y 使得 $y>\text{median}$;
5. 如果存在 $(n-1)/2$ 个元素已得到小于 median 的值，则对未赋值的全部分配大于 median 的值；
6. 如果存在 $(n-1)/2$ 个元素已得到大于 median 的值，则对未赋值的全部分配小于 median 的值.
7. 如果剩下1个元素则分配 median 给它.

找中位数问题：对手论证的构造实例

1. 初始 构造median=4
2. 比较x_1,x_2，构造$x_1>x_2$ $x_1=7$，$x_2=1$
3. 比较x_3,x_4，构造$x_3>x_4$ $x_3=5$, $x_4=2$
4. 比较x_5,x_6，构造$x_5>x_6$ $x_5=6$, $x_6=3$
5. 构造$x_7=4$
6. 比较$x_1>x_3$
7. 比较$x_3>x_7$
8. …

非关键比较

关键比较

找中位数问题：复杂度分析

元素状态　N：未分配值；S：得到小于median值；
L：得到大于median值

比较前的状态	分配策略
N, N	一个大于median，一个小于median
L, N 或 N, L	分配给状态N的元素的值小于median
S, N 或 N, S	分配给状态N的元素的值大于median

这样赋值的输入使得 A 在这个输入下所进行的上述比较都是非关键的. 这样的比较至少有$(n-1)/2$个. 因此总比较次数至少为

$$(n-1)+(n-1)/2 = 3n/2-3/2$$

结论：Select算法在阶上达到最优.

几种选择算法的总结

问题	算法	最坏情况	问题下界	最优性
找最大	**Findmax**	$n-1$	$n-1$	最优
找最大最小	**FindMaxMin**	$\lceil 3n/2 \rceil-2$	$\lceil 3n/2 \rceil-2$	最优
找第二大	锦标赛	$n+\lceil \log n \rceil-2$	$n+\lceil \log n \rceil-2$	最优
找中位数	**Select**	$O(n)$	$3n/2-3/2$	阶最优
找第k小	**Select**	$O(n)$	$n+\min\{k,n-k+1\}-2$	阶最优

图论问题的对手论证例子

- 假设图用邻接矩阵表示（基于查询边 e 的算法类）
- 例子：判断图的边集是否为空
  - 对手论证思路：维护图 E 和 G，使得在检查所有边之前，可以构造 G ≠ E
- 例子：判断图是否连通
  - 对手论证：根据算法的查询，同时维护两个图 Y 和 M
    - Y (yes) 只包含确定在图里的边、M (maybe) 包含可能在图里的所有边
  - 初始令 Y 为空图、M 为完全图；按右侧算法维护 Y 和 M
  - 性质
    - Y 是 M 的子图。M 总是连通的。
    - 如果 M 有环，环上的边都不属于 Y（环上的边不会进入 else 分支）。Y 无环（同理）。
    - 若 Y ≠ M，则 Y 是不连通的。（反证：若连通，则 Y 是树；加任一在 M 而不在 Y 的边都能成环，违反第2行）

```
HIDECONNECTEDNESS(e):
  if M \ {e} is connected
      remove (i, j) from M
      return 0
  else
      add e to Y
      return 1
```

J. Erickson, “Algorithms: Lecture 13 Adversary Arguments,” Fall 2019. https://jeffe.cs.illinois.edu/teaching/algorithms/

复习：决策树

- 给定 27 枚硬币
- 已知其中有 26 枚重量相同、有1枚偏重
- 用天平称重，找出偏重的硬币
- 试证明：任意基于比较的算法在最坏情况至少需要 3 次称重

元素唯一性问题（Element Uniqueness）

- 问题：给定 $X=(x_1, x_2, ..., x_n)$，是否 $\forall i,j$ 有 $x_i \neq x_j$
- 决策树模型，$T(n) = \Omega(n \log n)$
  - 树高 ≥ log (|{YES,NO}|) = log (2) = 1，至少比较1次?
  - 改进的数叶子方法
    - 定义在 YES 叶子 L_k 停机的 X 的集合为 $S(L_k)$；$S(L_k)$ 是个凸集
    - 如果 Z 是 YES 实例 X 的非平凡置换（$\exists i,j$ 使 $x_i > z_i$ 且 $x_j < z_j$），则 X 和 Z 不属于同一个 $S(L_k)$
    - 因此，YES 叶子数 ≥ n!，树高 ≥ n log n
- 代数计算树模型，$T(n) = \Omega(n \log n)$（略）
  - Michael Ben-Or, “Lower bounds for algebraic computation trees,” STOC '83.
  - Andrew C.-C. Yao, “Lower bounds for algebraic computation trees with integer inputs,” FOCS '89.

通过归约确认问题计算复杂度的下界

问题 P，问题 Q

问题 Q 的复杂度已知 $\Omega(g(n))$

存在变换 f 将 Q 的任何实例 I 转换成 P 的实例 $f(I)$（I 和 $f(I)$ 规模同阶）

解 Q 的算法：$T_Q(\mathrm{n}) = T_1+T_P(n)+T_3$

1. 将 Q 的实例 I 变成 $f(I)$，时间 T_1
2. 用解 P 的算法作为子程序解 $f(I)$，得 $s(f(I))$，时间 $T_P(n)$
3. 将解 $s(f(I))$ 变换成原问题的解 $s'(I)$，时间 T_3

假设 T_1 和 T_3 相对 $T_P(n)$ 是次要的（例如 $g(n)$ 至少线性，T_1和T_3线性）

解 P 的算法可以解 Q. 且时间的阶一样，因此 P 至少与 Q 一样难. $Q \le_l P$

$$T_1+T_P(n)+T_3= T_Q(n) =\Omega(g(n))$$

归约：最邻近点对与唯一性问题

- P问题与Q问题：

  P: 平面直角坐标系中n个点的最邻近点对问题

  Q: 元素的唯一性问题：给定 n 个数的集合S，判断 S 中的元素是否存在相同元素. 时间 $\Omega(n\log n)$.

- 变换 f:

  Q的实例：$x_1, x_2, \ldots, x_n$，变成点$(x_1,0),(x_2,0),\ldots,(x_n,0)$

- 解Q算法：

  1. 利用求最邻近点对算法 P 计算最短距离 d.
  2. if d=0 then return “No”
  3. else return “Yes”

结论：计算平面直角坐标系中n个点的最邻近点对问题的时间是 $\Omega(n\log n)$，其中算法以比较为基本运算

归约：最小生成树与唯一性问题

- P问题与Q问题：

  P: 平面直角坐标系中n个点的最小生成树问题；

  Q: 元素的唯一性问题　$\Omega(n\log n)$

- 变换 f:

  Q的实例：$x_1, x_2, \ldots, x_n$，变成X 轴上的n个点,

- 解Q算法：

  1. 利用求最小生成树算法P构造树T，确定T的最短边e.
  2. 检测e的长度是否为0
  3. if $|e|=0$ then 不唯一，else 是唯一的.

- 结论：计算平面直角坐标系 n点最小生成树时间是$\Omega(n\log n)$，其中算法以比较为基本运算

决策树 vs 布尔电路

- 决策树模型：“数据访问”的下界
  - 比较操作的数目
- 布尔电路模型：“数据变换”的下界
  - 电路的深度和门数

Reference: A. Chakrabarti, “CS85: Lower Bounds in Computer Science,” Dartmouth College, Spring 2008.

布尔电路：定义

- n变量布尔电路是包含以下指定类型节点的有向无环图
  - 输入节点：入度为零的节点，带标记 x1, ¬x1, x2, ¬x2, …, xn, ¬xn
  - 输出节点：出度为零的节点
  - 门节点：AND门、OR门，入度为二；NOT门，入度为一
- 布尔电路的值可以通过归纳定义
- 布尔电路的大小 = 边的数目 = Θ(门的数目)
- 布尔电路的深度 = 从输入到输出的最长路径

Reference: A. Chakrabarti, “CS85: Lower Bounds in Computer Science,” Dartmouth College, Spring 2008.

布尔电路：阈值函数下界

- 定理：任意实现 n 变量阈值函数 THR_n^2 的布尔电路至少有 2n-3 个门
  - n 变量 k 阈值函数 THR_n^k：当且仅当至少有 k 个变量为 1 时函数值为 1，否则为 0
  - 容易证明更松的门数下界 n-1
- 证明（门消元 + 数学归纳法）
  - 归纳基础：当 n=2 时， $THR_2^2 = AND_2$，门数 1 = 2n-3
  - 假设 C 是 THR_n^2 的最小电路，则 C 不包含以 (xi,xi) 或 (xi, ¬xi) 或 (¬xi, ¬xi) 为输入的门
  - 选择某个以 zi（xi 或 ¬xi）和 zj（ xj 或 ¬xj ）为输入的门，可以证明 zi 或 zj 至少有一个值被两个门使用。因为 zi 和 zj 的不同取值组合，我们可以得到 n-2 个变量的函数 THR_{n-2}^2、 THR_{n-2}^1 和 THR_{n-2}^0。如果 zi 和 zj 的值只被一个门使用，我们只能得到两个不同的阈值子电路，矛盾
  - 假设 xi（或者 ¬xi）的值被至少两个门使用，设置 xi=0 将消掉两个门，得到 THR_{n-1}^2 子电路，门数比原电路少2
  - 根据归纳假设，上述 THR_{n-1}^2 子电路至少有 2(n-1)-3=2n-5 个门。因此原电路至少有 2n-3 个门

布尔电路：多项式电路类vs多项式算法类

- 计算 f: {0,1}*→{0,1}，即识别语言 $L_f \subseteq \{0,1\}^*$，我们需要一族电路 $\langle C_n \rangle_{n=1}^{\infty}$，其中 C_n 计算输入长度为 n 的函数
  - 没有**一致性**要求的布尔电路甚至能“计算”不可判定问题，例如输入以一进制编码的停机问题
- 定理 P ⊆ P/poly：如果 L ∈ P，则存在一族多项式大小的布尔电路，即 $\exists \langle C_n \rangle_{n=1}^{\infty}$ 使得 $\forall n \forall x \in \{0,1\}^n\ C_n(x) = 1 \Leftrightarrow x \in L$ 且 C_n 的大小为关于 n 的多项式
  - 证明思路：构造布尔电路，模拟识别 L 的图灵机；电路大小是 O(图灵机运行时间^2)
  - M. Sipser, “Introduction to the Theory of Computation (2nd ed.),” 2006 第九章
- 曾被认为有希望证明 P ≠ NP
  - 选择你喜欢的 NPC 问题 L
  - 根据以上定理，如果 P = NP，则存在多项式大小的电路计算 L
  - 证明计算 L 的电路大小的下界是超多项式，则可证明 P ≠ NP

Reference: A. Chakrabarti, “CS85: Lower Bounds in Computer Science,” Dartmouth College, Spring 2008.

布尔电路：香农定理

- 香农定理：几乎所有函数 f: {0,1}^{n}→{0,1} 至少需要 $\Omega(2^n/n)$ 门的布尔电路来计算
- 证明:
  - 假设布尔电路不包含NOT门。只要输入节点包含所有的变量和它们的非，则可使用德摩根律消除非输入节点外的NOT门
  - 布尔电路的节点可作拓扑排序，电路的门数记为 s，则每个门的类型有2种选择、每个门的输入有不超过 s^2 种选择。因此，大小为 s 的电路不超过 $(2s^2)^s \le s^{3s}$
  - 每个电路计算唯一的布尔函数，所以能用不超过 $2^n/(10n)$ 个门计算的函数不超过

$$\left(\frac{2^n}{10n}\right)^{3\cdot\frac{2^n}{10n}} \le \frac{2^{3\cdot 2^n/10}}{(10n)^{3\cdot 2^n/10n}} = 2^{\frac{3\cdot 2^n}{10}-\frac{3\cdot 2^n}{10n}\log 10n} = 2^{2^n\left(\frac{3}{10}-\frac{3\log 10n}{10n}\right)} < 2^{2^n\cdot\frac{3}{10}}$$

  - 布尔函数 f: {0,1}^{n}→{0,1} 共有 2^{2^n} 个，因此 $\lim_{n\to\infty}\frac{2^{2^n\cdot\frac{3}{10}}}{2^{2^n}}=0$

Reference: A. Chakrabarti, “CS85: Lower Bounds in Computer Science,” Dartmouth College, Spring 2008.

布尔电路：多项式电路类vs多项式算法类

- 计算 f: {0,1}*→{0,1}，即识别语言 $L_f \subseteq \{0,1\}^*$，我们需要一族电路 $\langle C_n \rangle_{n=1}^{\infty}$，其中 C_n 计算输入长度为 n 的函数
- 定理 P ⊆ P/poly：如果 L ∈ P，则存在一族多项式大小的布尔电路，即 $\exists \langle C_n \rangle_{n=1}^{\infty}$ 使得 $\forall n \forall x \in \{0,1\}^n\ C_n(x) = 1 \Leftrightarrow x \in L$ 且 C_n 的大小为关于 n 的多项式
- 曾被认为有希望证明 P ≠ NP
  - 选择你喜欢的 NPC 问题 L
  - 根据以上定理，如果 P = NP，则存在多项式大小的电路计算 L
  - 证明计算 L 的电路大小的下界是超多项式，则可证明 P ≠ NP
  - 然而，2008年前已证明的最大下界是 5n-o(n) [Iwama et al., FOCS’01]

Reference: A. Chakrabarti, “CS85: Lower Bounds in Computer Science,” Dartmouth College, Spring 2008.

本讲总结

- 寻找最优算法的途径
  - 1. 设计算法；2. 分析问题下界；
  - 3-∞. 努力设计算法和分析下界，直至两者相遇
- 问题复杂度下界
  - 平凡下界
  - 直接计算最少运算次数（例：无序数组找最大)
  - 决策树（例：检索问题）
  - 对手论证（例：选择问题）
  - 归约
- 非决策树模型：代数计算树、布尔电路，等等