上海师范大学
Shanghai Normal University

《算法设计与分析》

10-NP 完全问题 (NP-complete Problem)

杨启哲

上海师范大学信机学院计算机系

2025 年 11 月 13 日

主要内容

上海师范大学 Shanghai Normal University

- P 类问题与 NP 类问题
- NP 完全问题
- 其他跟 NP 类问题相关的问题类

P 类问题与 NP 类问题

可以高效解决的问题

目前我们已经学习了很多问题的算法，很多问题可以在多项式时间内解决:

- 排序问题
- 图的最短路径问题
- 最长公共子序列问题
- . . .

而有些问题则我们还没有方法可以在多项式时间内解决:

- 输出所有的排列。
- 背包问题。

一般来说，我们称可以在多项式时间内解决的问题为可以高效解决的问题，把目前没有看起来未来也不太可能有多项式时间算法的问题称为不可以高效解决的问题。

Sissa 和 Morre

传说中，国际象棋游戏是印度人 Brahmin Sissa 发明的，其目的是为了娱乐和教导他的国王。当心存感激的国王问 Sissa 想要什么赏赐时，充满智慧的 Sissa 向国王提出了这样的请求：请在棋盘的第一个格子里放上一粒稻谷、第二个格子放上两粒，第三个四粒，以此类推，每次在下一个格子里放上前一格加倍数量的稻谷，直到 2^{63}。国王立刻就答应了，当然国王就领略到了指数级增长的威力。因为满足该请求一共需要：

$$2^{64} - 1 = 18446744073709551615$$

足够将整个印度国土覆盖好几层！

为什么多项式时间被认为是高效的算法？(II)

Moore 定律

1965 年，计算机芯片行业的先驱 Gordon E. Moore 注意到，集成电路上的晶体管数量每两年翻一番，于是他预测这一规律将会持续下去。经过适当的修正，如今这一规律被表述为，每 18 个月集成电路上的晶体管数量将会翻一番，即著名的 Moore 定律。

表面上看，这一定律对于多项式时间算法应该是产生反向刺激的，毕竟如果算法是指数级的，为什么不能等到 Moore 定律使它的实现变为可能？但事实恰恰相反:

问题实例	1975	1985	1995	2023
$O(2^n)$	25	31	38	50
$O(n^2)$	25	2500	25×10^4	1×10^8
$O(n^6)$	25	50	100	350

多项式时间算法的能力的提升将是指数级的，而指数时间算法的能力只能以多项式的时间进行提升！

我们先关注一类特殊的问题：**判定问题**。

定义 1 **[判定问题 (decision problems)].**

一个判定问题指的是答案只有 0 或 1 的问题。

例 2.

- 给定一个整数数列，其中是否存在相同的元素?
- 给定一个无向图 $G = (V, E)$ 和整数 k，是否可以对其顶点进行 k 染色?
- 给定一个无向图 $G = (V, E)$ 和整数 k，是否存在一个包含至少 k 个顶点的团?

判定问题与优化问题 (I)

很多问题都有判定版本和优化版本，例如：

团 (Clique)

判定问题：团

输入：无向图 $G = (V, E)$ 和整数 k。

问题：是否存在一个包含至少 k 个顶点的团？

优化问题：最大团

输入：无向图 $G = (V, E)$。

问题：G 中的最大团的大小。

顶点染色

判定问题：顶点染色

输入：无向图 $G = (V, E)$ 和整数 k。

问题：是否可以对其顶点进行 k 染色？

优化问题：色数 (chromatic number)

输入：无向图 $G = (V, E)$。

问题：G 的色数，即将其顶点染色使得相邻的顶点颜色不同的最小颜色数。

上海师范大学 Shanghai Normal University

判定问题的算法经常可以用来解决优化问题:

用团的判定问题解决优化问题!

假设我们有一个算法 A 可以解决团的判定问题，那么我们可以设计如下的优化算法:

算法 opt–A

输入: 无向图 $G=(V,E)$。

输出: G 的最大团的大小。

1: $\mathrm{min} \leftarrow 1$, $\mathrm{max} \leftarrow |V|$, $k \leftarrow \mathrm{min}$
2: **while** $\mathrm{min} \leqslant \mathrm{max}$ **do**
3:     $\mathrm{mid} \leftarrow \lfloor (\mathrm{min}+\mathrm{max})/2 \rfloor$
4:     **if** $A(G, \mathrm{mid})$ **then**
5:         $\mathrm{min} \leftarrow \mathrm{mid}+1$
6:     **else**
7:         $\mathrm{max} \leftarrow \mathrm{mid}-1$
8: **return** max

我们现在来介绍 P 类问题的概念:

定义 3 **[确定性算法].**

设 A 是求解问题 P 的一个算法，如果对于问题 P 的任何一个实例，在整个执行过程中每一步都只有一种选择，那么我们称 A 是一个确定性算法，也就是说对于同样的输入，A 的输出从来不会被改变。

定义 4 **[P 类问题].**

P 类问题由这样的判定问题组成，其解（是/不是）可以用确定性算法在运行多项式步内，比如 $O(n^k)$ 内得到，这里 k 是一个与 n 无关的常数。

P 类问题举例

关于 P 类问题

- 我们之前学习的大部分算法都是确定性的，并且落在 P 类问题中。
- 这里我们没有使用通常的方式，比如利用图灵机这样的计算模型去定义 P 类问题，而是直接使用了抽象的”算法“概念。

P 类问题举例

- 给定一个整数数列，其中是否存在相同的元素？
- 给定一个带非负权重的有向图 $G=(V,E)$ 和正整数 k 及两个特殊的顶点 s,t，问 s 到 t 是否存在一条长度至多为 k 的路径？
- 给定一个无向图 $G=(V,E)$，其是否可以被二染色？

P 类问题有一个很好的性质：其**补问题**也在 P 类问题中。

定义 5 **[补问题].**

设 P 是一个判定问题，其补问题 $\overline{\mathrm{P}}$ 定义为：对于 P 的任何一个实例，其补问题的答案是 P 的答案的否定，即如果 P 回答 1，$\overline{\mathrm{P}}$ 回答 0;P 回答 0，$\overline{\mathrm{P}}$ 回答 1。

例 6.

图的 k-染色问题的补问题定义如下：给定一个无向图 $G = (V, E)$ 和整数 k，是否不可对其顶点进行 k 染色?

定理 7.

如果 P 是一个 P 类问题，那么其补问题 $\overline{\mathrm{P}}$ 也是一个 P 类问题。

容易被验证的问题

我们已经介绍了 P 类问题，这是一类可以被高效解决的问题，但是有些判定问题我们可能没有高效的算法去解决，但是我们可以高效的验证一个解是否正确，比如：

- 给我们一个图上的 k-染色方案，我们可以在多项式时间内验证其是否正确。
- 给我们图上的一条路径，我们可以在多项式时间内验证其是否是一条哈密顿路径。

我们把这样的判定问题称为**可以被高效验证的**问题。事实上，这就是著名的 NP 类问题.

上海师范大学 Shanghai Normal University

为了更准确的定义 NP 类问题，我们需要引入非确定型算法的概念：

非确定型算法

对于一个输入 x，一个非确定型算法由下列两个阶段组成：

- 猜测阶段：这个阶段，算法任意的产生一个字符串 y，这个字符串可能对应输入实例的一个解，也可能什么意义也没有。这一阶段唯一的要求就是，y 能在多项式步数内生成，即 $O(|x|^k)$ 步，其中 k 是一个与 x 无关的非负整数。
- 验证阶段：在这个阶段，一个确定性算法验证两件事：
    - y 是否是合法的，即 y 是否是一个合法的字符串，如果不合法则回答 NO。
    - y 是否是 x 的一个解，如果是正确的，则回答 YES，否则回答 NO。

我们称一个非确定型算法是正确的，当且仅当存在一个导致回答 YES 的猜测 y。如果整个算法的执行时间是多项式的，则称该算法是一个多项式时间的非确定型算法。

非确定性算法举例 (I)

以图的 k-染色问题为例，我们可以设计一个非确定型算法：

- **猜测阶段**：我们对图上的顶点任意的给予一个赋予一个颜色；从而得到一个对应的赋色方案。
- **验证阶段**：我们检查这个赋色方案是否合法，即相邻的顶点是否有相同的颜色，如果不合法则回答 NO；如果合法，我们检查这个赋色方案是否是一个 k-染色方案，如果是则回答 YES，否则回答 NO。

算法的两个阶段都可以在多项式时间内完成，因此这是多项式时间的非确定性算法。

非确定型算法举例 (II)

上海师范大学 Shanghai Normal University

再来考虑图的哈密顿路径问题，我们也可以设计一个类似的非确定型算法：

- **猜测阶段**：我们任意的生成对图上顶点的一个排列。
- **验证阶段**：我们检查这个排列是否是图的一个哈密顿路径，如果是则回答 YES，否则回答 NO。

算法的两个阶段都可以在多项式时间内完成，因此这是多项式时间的非确定性算法。

NP 类问题

上海师范大学 Shanghai Normal University

定义 8 **[NP 类问题].**

NP 类问题由这样的判定问题组成，其解（是/不是）可以用多项式时间的非确定性算法在运行多项式步内，比如 $O(n^k)$ 内得到，这里 k 是一个与 n 无关的常数。

NP 类问题

- 所有的 P 类问题也都是 NP 类问题。
- 图的 k-染色问题:
  给定无向图 $G = (V, E)$ 和整数 k，是否可以对其顶点进行 k 染色?
- 图的哈密顿回路问题:
  给定无向图 $G = (V, E)$，问其是否存在一条哈密顿回路?
- SAT 可满足性问题:
  给定一个 n 个变量的合取范式，是否存在一组布尔值使得该合取范式为真?

我们现在来思考一下这两类问题的异同：

- 共同点：
  1. P 类问题和 NP 类问题都是判定问题。
  2. P 类问题和 NP 类问题都是具有多项式时间属性的。
- 不同点：
  1. P 类问题可以用确定性算法解决，NP 类问题可以用非确定性算法解决。
  2. P 类问题可以在多项式时间内求得答案，NP 类问题只能在多项式时间内验证。

问题 $NP = P$?

现在我们可以知晓计算机理论中的核心问题 $NP = P$? 究竟代表的是什么，即 P 类问题和 NP 类问题是否相等。

NP 完全问题

什么是 NP 完全问题

上海师范大学 Shanghai Normal University

同样是 NP 类问题，有些问题看起来比另一些问题更难，比如：

- 数列中是否存在相同的元素？
- 无向图中是否存在哈密顿路径？

这两个问题都是 NP 类问题，但是第二个问题看起来比第一个问题更难，因为前者可以在多项式时间内解决，是一个 P 类问题，而后者目前还没有多项式时间的算法。

为了更好的区分 NP 类问题中的难度，我们引入了NP 完全问题的概念，即 NP 问题类中最难的问题。

怎么定义**最难**？

问题归约

如果存在两个判定问题 Π_1, Π_2，如果现在存在一个确定性算法 B，其行为如下：

- 对于问题 Π_1 的任何一个输入 x，B 可以在多项式时间内将其转化为问题 Π_2 的输入 y。
- $\Pi_1(x)$ 回答是 YES 当且仅当 $\Pi_2(y)$ 回答是 YES

那我们称 Π_1 可以多项式时间归约到 Π_2，记作 $\Pi_1 \propto_{poly} \Pi_2$。

问题归约的意义

事实上 $\Pi_1 \propto_{poly} \Pi_2$ 其实意味着 Π_1 不会比 Π_2 更难，因为我们可以用 B 将 Π_1 转化为 Π_2，然后用 Π_2 的算法去解决 Π_1。

定义 9 **[NP 困难问题 (NP—hard Problem)].**

一个判定问题 Π 被称为是 NP 困难问题，如果对于 NP 类问题 Π'，都有 $\Pi' \propto_{poly} \Pi$。

定义 10 **[NP 完全问题 (NP—complete Problem)].**

一个判定问题 Π 被称为是 NP 完全问题，如果其满足：

- Π 是一个 NP 问题。
- Π 是一个 NP 困难问题，即对于 NP 类问题 Π'，都有 $\Pi' \propto_{poly} \Pi$。

可满足性问题 (I)

我们现在来回顾下**可满足性问题 (Satisfiability Problem)**，这也是第一个著名的**存在实际意义的**NP 完全问题。

回顾一下，一个布尔公式 f 被称为合取范式如果其具有如下的形式：

$$f = (x_1 \vee x_2) \wedge (x_2 \vee \neg x_3 \vee x_4) \wedge (x_1 \vee x_2 \vee \neg x_3 \vee \neg x_4)$$

公式 f 被称为是可满足的，如果存在一组布尔值使得 f 为真。

判定问题：可满足性问题 (SAT)

输入： 一个 n 个变量的合取范式 f。

问题： f 是否可满足？

定理 11 [Cook–Levin 定理].

SAT 是 NP 完全的。

怎么去证明一个问题是 NP 完全问题

我们现在介绍了第一个 NP 完全问题:SAT，对于其他的问题，我们如何证明其是 NP 完全的?

- 按照定义，我们要证明所有的 NP 困难问题都可以多项式时间归约到该问题。

幸好，多项式归约具有好的性质:**传递性**，让我们可以简化证明的方式。

Shanghai Normal University

定理 12.

如果 $\Pi_1 \propto_{poly} \Pi_2$ 且 $\Pi_2 \propto_{poly} \Pi_3$，则 $\Pi_1 \propto_{poly} \Pi_3$。

证明. 由定义，存在多项式 p_1，使得对于 Π_1 的任何一个输入 x，我们可以在多项式时间内将其转化为 Π_2 的输入 y，且 $|y| \leqslant cp_1(|x|)$。这里 c 是一个与 x 无关的常数；同样的，存在多项式 p_2，使得对于 Π_2 的任何一个输入 y，我们可以在多项式时间内将其转化为 Π_3 的输入 z，且 $|z| \leqslant cp_2(|y|)$。

因此存在一个算法 B，使得对于 Π_1 的任何一个输入 x:

- B 可以在 $q(cp_1(|x|))$ 的时间内将其转化为 Π_3 的输入 z，z 的大小关于 x 是多项式的。
- $\Pi_1(x) = 1$ 当且仅当 $\Pi_3(z) = 1$。

即：$\Pi_1 \propto_{poly} \Pi_3$。 □

推论 13.

如果 $\Pi_1 \propto_{poly} \Pi_2$ 且 Π_1 是 NP 完全的，则 Π_1 也是 NP 完全的。

证明 NP 完全问题的方法

由上述推论，我们可以得到证明一个问题 Π 是 NP 完全问题的方法：

---

1. 证明该问题是 NP 问题。
2. 找出一个已知的 NP 完全问题 Π'，证明该问题可以多项式归约到 Π，即 $\Pi' \propto_{poly} \Pi$.

一个例子

我们现在来证明团问题是 NP 完全问题:

判定问题: 团

输入: 无向图 $G = (V, E)$ 和整数 k。

问题: 是否存在一个包含至少 k 个顶点的团?

定理 14.

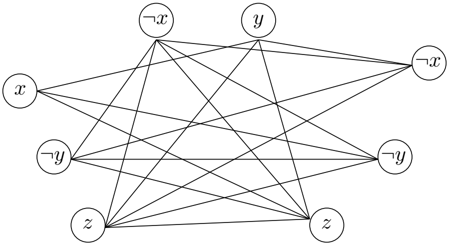
团问题是 NP 完全的。

团问题是 NP 完全的证明. 我们将 SAT 问题归约到团问题上。令 f 是一个 n 个变量 m 个子句的合取范式，$f = C_1 \wedge C_2 \wedge \cdots C_m$，构造如下的图 $G = (V, E)$:

- 对于每个子句的每个文字 (x or$\neg x$) 我们都在图中定义一个顶点来表示。
- $E = \{(x_i, x_j) | (x_i \neq \neg x_j) \wedge x_i$ 和 x_j 在不同的子句$\}$。

显然这是一个多项式时间内可以完成的归约。

$$f = (x \vee \neg y \vee z) \wedge (\neg x \vee y) \wedge (\neg x \vee \neg u \vee z)$$



引理 15.

f 是可满足的当且仅当 G 有一个大小为 m 的团。

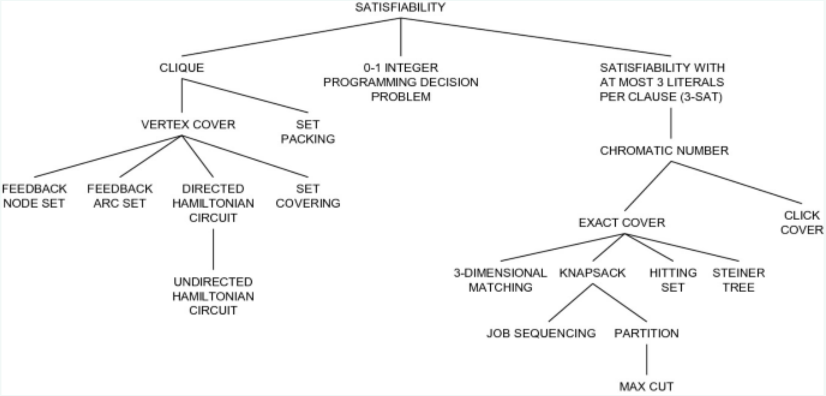
引理证明. 一个大小为 m 的团集对应了在 m 个子句中对其中 m 个文字的真值指派。两个文字之间有边等价于这两个文字的真值指派没有矛盾，从而：

$$
\begin{aligned}
& f\text{是可满足的} \\
\Leftrightarrow\quad & \text{存在一个真值指派使得 } m \text{ 个子句中都至少有一个文字为真} \\
\Leftrightarrow\quad & \text{图 } G \text{ 存在一个大小为 } m \text{ 的团。}
\end{aligned}
$$

回到原来的问题，由上述引理，我们有：$SAT \propto_{poly} Clique$，从而团问题是 NP-难的，注意到其显然是可以多项式验证的，因此团问题是 NP 完全问题。 □

Karp 的 21 个 NP 完全问题

1971 年 Cook 发表了对第一个实际的 NP 完全问题的证明 (Cook—Levin 定理) 之后，Karp 在 1972 年发表了”Reducibility Among Combinatorial Problems” 这篇文章，其中列举了如下的 21 个 NP 完全问题：



3-可满足性问题 (3–SAT)是在可满足性问题的基础增加**每个句子至多有 3 个文字**的限制。

定理 16.

3–SAT 问题是 NP 完全的。

证明. 3–SAT 问题在 NP 是显然的，因为可以在多项式时间内验证一组赋值是否使其为真。

下面我们证明：SAT $\propto_{poly}$ 3–SAT. 给定一个 SAT 的实例 φ，我们来构造一个与其同真假的 3–SAT 实例 φ：

- 如果 φ 没有超过 3 个文字的句子，则直接令 $\varphi' = \varphi$。
- 如果 φ 中存在一个句子超过 3 个文字 φ_i，令 $\varphi_i = a_1 \vee a_2 \vee \ldots \vee a_k$，定义下式：

$$\phi_i = (a_1 \vee a_2 \vee z_1) \wedge (\neg z_1 \vee a_3 \vee z_2) \wedge \cdots \wedge (\neg z_{k-3} \vee a_{k-1} \vee a_k)$$

ϕ_i 为可满足当且仅当 φ_i 是可满足的。将所有的 φ_i 都替换成上述的 ϕ_i，所得到的新的公式 φ' 即是 3–SAT 的实例。

□

问题 17 **[顶点覆盖问题 (Vertex Cover)].**

给定无向图 $G = (V, E)$ 和正整数 k，是否存在大小为 k 的子集 $V' \subseteq V$，使得对于任意的边 $(u, v) \in E$，都有 $u \in V'$ 或 $v \in V'$?

定理 18.

顶点覆盖问题是 NP 完全的。

非常显然，这个问题**是个 NP 问题**，所以我们重点还是**证明其是 NP-难的**。

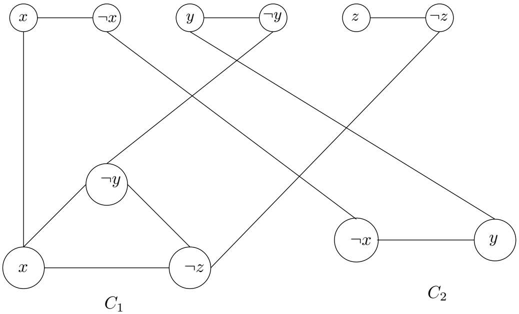
可满足性问题到顶点覆盖问题的归约 (I)

我们现在将 SAT $\propto_{poly}$ VertexCover。

给定 SAT 的实例 I，设其具有 m 个子句和 n 个布尔变元 $x_1, \dots, x_n$，子句 C_i 包含的文字数位 n_i。我们构造一个顶点覆盖问题的实例 $I' = (G_I, k)$ 如下：

- 对于每个布尔变元 x_i，G 包含一对顶点 $x_i, \neg x_i$，并且存在一条边相连。
- 对于每个具有 m 个文字的子句，G 包含一个大小为 m 的团集，每个顶点代表其中一个文字。
- 对于团集中的每个文字，我们跟一开始构造的顶点中相同的文字连一条边。
- 令 $k = n + \sum_{j=1}^{m}(n_j - 1)$.

$$f = \underbrace{(x \vee \neg y \vee \neg z)}_{C_1} \wedge \underbrace{(\neg x \vee y)}_{C_2}$$



可满足性问题到顶点覆盖问题的归约 (II)

引理 19.

G_I 存在一个大小为 k 的顶点覆盖当且仅当公式 I 是可满足的。

证明.

$\Leftarrow$ 如果存在一个真值指派使得 I 是可满足的，则若该指派选择将 x 赋值为真，则将 x 放入 k 个顶点集合中，否则放入 $\neg x$；此外对于子句所代表的团，我们选取其中 $|C|_i - 1$ 个点作为集合中的元素（任意排除一个使其赋值为真的点），不难证明，这样挑出来的 k 个顶点是图中的一个定点覆盖。

□

证明.

$\Rightarrow$ 如果图中存在一个大小为 k 的顶点覆盖，注意到：

1. 每个团集至少有 $n_i - 1$ 个顶点。
2. 每个 $x, \neg x$ 至少要包含一个点。

从而每个 $x, \neg x$ 恰好包含一个点。我们断定选择 $x, \neg x$ 的情况给予了公式的一个成真赋值，这是因为：

- 一个具有 n 个文字的子句与 $x, \neg x$ 恰好连了 n 条边。
- 如果对 $x, \neg x$ 的选择没有与该子句相同的文字的话，这 n 条边至少会有一条边没有被覆盖到。

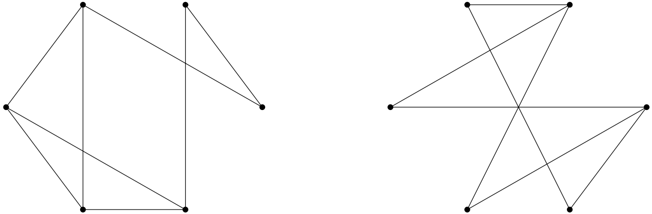
$\square$

团问题到顶点覆盖问题的归约

SAT 到 VertexCover 的归约还是比较麻烦的，我们尝试换个问题进行考虑。下面说明 Clique $\propto_{poly}$ VertexCover

---

回顾一下**补图**的概念：



引理 20.

图 G 有一个大小为 k 的团当且仅当其补图有一个规模为 $|V|-k$ 的顶点覆盖。

其他跟 NP 类问题相关的问题类

与 P 类问题类似，我们也可以定义NP **类问题的补**：

定义 21 **[co—NP 问题].**

co—NP 问题是由其的补问题属于 NP 类问题的判定问题组成的集合。

类似 NP 完全问题，我们有：

定义 22 **[co—NP 完全问题].**

问题 Π 是 co—NP 完全的，如果：

1. Π 是 co—NP 问题。
2. 对于任何一个 co—NP 问题 Π'，$\Pi' \propto_{poly} \Pi$.

判断一个公式是否是重言式 (tautology) 便是 co–NP 问题:

- 其等价于判断其否定是否是不可满足的。

进一步，我们可以发现其是一个 co–NP 完全问题。事实上，我们有:

定理 23.

问题 Π 是 NP 完全的，当且仅当其补问题 $\overline{\Pi}$ 是 co–NP 完全的。

注意到，**完全问题**可以理解成在该类问题中**最难的**问题，所以:

定理 24.

如果存在一个问题 Π 使得其和其补问题都是 NP 完全的，则 $\mathrm{NP} = \text{co–NP}$

NPI 问题 (I)

让我们再把焦点放到 NP 问题里。

- P 类问题在其中。
- 我们已经证明了存在一些 NP 完全问题，即 NP 问题里最难的那种问题。

那是否会存在一个 NP 问题，其既不是 P 问题，也不是 NP 完全问题?

NPI 问题 (II)

上海师范大学 Shanghai Normal University

我们将这种问题称作 NPI 问题 (NP–intermediate)，显然如果存在这么一个问题，则 $NP \neq P$。 下面的定理告诉我们，如果 $P \neq NP$，则这样的问题一定存在：

定理 25 [ladner theorem].

如果 $P \neq NP$，则一定存在一个 NPI 问题。

推论 26.

$P = NP$ 当且仅当不存在 NPI 问题。

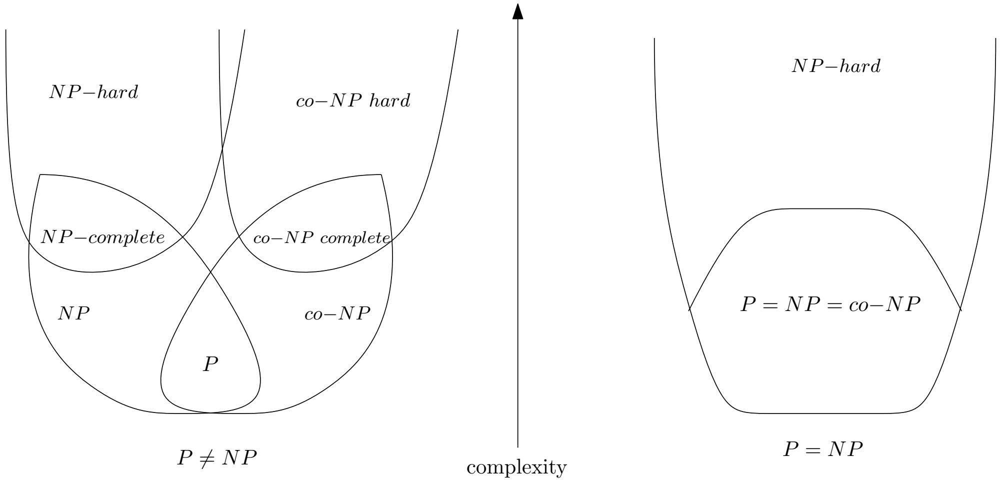
关于 NPI 问题

事实上，关于实际上的 NPI 问题研究者也一直在努力寻找，也曾有过若干的怀疑对象：

1. **判定素数**. 但这一问题在 21 世纪初被证明具有多项式时间算法。
2. **图同构问题**. 目前最好的结论是伪多项式时间算法，可能是目前被认为最接近能证明是在 NPI 里的问题.

事实上，这一难度应该与证明 $P \neq NP$ 是一样的，即可能目前所有已有的证明方法都不能成功。

我们上述复杂性类的关系可以如下图所示:



本章总结

- P 类问题与 NP 类问题
  - 判定问题和搜索问题的概念、区别与联系
  - P 类问题与 NP 类问题
- NP 完全问题
  - 问题归约
  - NP 完全问题的定义
  - NP 完全问题的证明方法、举例
- 其他复杂性类
  - co—NP 类问题
  - NPI 类问题
  - 复杂性类之间的关系