NP 完全性

多项式相关

设 $f,g:\mathbb{N}\to\mathbb{N}$。如果存在多项式 p 和 q，使得对任意的 $n\in\mathbb{N}$，$f(n)\le p(g(n))$，$g(n)\le q(f(n))$，则称函数 f 和 g 是多项式相关的。

- 例如：$nlog(n)$ 和 n^2 是多项式相关的，而 $log(n)$ 和 n 不是多项式相关的。

易解的和难解的

设 A 是求解问题 Π 的算法，在用 A 求解 Π 的实例 I 时，首先要把 I 编码成二进制的字符串作为 A 的输入，称 I 的二进制编码的长度为 I 的规模，记作 $|I|$。如果存在函数 $f:\mathbb{N}\to\mathbb{N}$，使得对任意的规模为 n 的实例 I，A 对 I 的运算在 $f(n)$ 步内停止，则称算法 A 的时间复杂度是 $f(n)$。以多项式为时间复杂度的算法称作 **多项式时间算法**。有多项式时间算法的问题称作 **易解的**，不存在多项式时间算法的问题称作是 **难解的**。

NOTE:

- 这里定义的计算复杂度是用编码的输入规模来衡量的。之前用到的时间复杂度都是用自然参数（如顶点数，边数等）直接衡量。这两者不一定相同，但是一般是多项式相关的。
- 一般来说，输入规模与具体的编码方式有关。但是可以证明采用 k 进制编码 $(k\ge 2)$ 时，编码长度与二进制编码的长度相差不超过 $\lceil log_2 k\rceil$ 倍；用二进制进行不同方式的编码最终得到的编码长度也一般是多项式相关的。
- 算法的运行时间还和选取的操作指令集相关。但是同样可以证明合理的指令集得到的指令长度只有常数的倍数差距。
- 总之，算法的是不是多项式时间的和选取的编码方式和操作指令集无关。

P 和 NP

定义

1. 判定问题是指输出只有 yes 和 no 的任务。、
   - 比如哈密顿回路问题：任给无向图 G，问 G 是哈密顿图吗?

2. 所有多项式时间可解的判定问题组成的问题类称作 P 类。

3. 所有多项式时间可验证的判定问题组成的问题类称作 NP 类
   - 多项式时间可验证是指随机取一个解，可以在多项式时间内验证这个解是否为可行解。比如在哈密顿回路问题中：随机取一些顶点，验证这些顶点是否构成哈密顿回路。

关系

- $P\in NP$：事实上验证的算法可以直接取成求解的算法。当求解的结果和要验证的解不同是，返回 no，否则返回 yes。这样得到的验证算法当然是多项式时间的。

NP 完全性

多项式时间变换

设判定问题 $\Pi_1=\langle D_1,Y_1\rangle$，$\Pi_2=\langle D_2,Y_2\rangle$。如果函数 $f:D_1\to D_2$ 满足条件：

1. f 是多项式时间可计算的。
2. 对所有的 $I\in D_1$，$I\in Y_1\Leftrightarrow f(I)\in Y_2$

则称 f 是 Π_1 到 Π_2 的多项式时间变换，并称 Π_1 可以多项式时间变换到 Π_2，记作 $\Pi_1\le_p\Pi_2$。

$\le_p$ 的性质：

1. $\le_p$ 具有传递性。即设 $\Pi_1\le_p\Pi_2$，$\Pi_2\le_p\Pi_3$，则 $\Pi_1\le_p\Pi_3$
2. $\Pi_1\le_p\Pi_2$，则 $\Pi_2\in P$ 蕴含 $\Pi_1\in P$。
3. $\Pi_1\le_p\Pi_2$，若 Π_1 难解，则 Π_2 难解。

因此多项式时间变换提供了一种衡量判定问题之间难度的手段 —— 如果 $\Pi_1 \leq_p \Pi_2$ ，则相对于多项式时间 Π_1 不会比 Π_2 更难。

NP 难和 NP 完全性

定义：

如果对所有的 $\Pi^{'} \in NP$, $\Pi^{'} \leq_p \Pi$ ，则称 $\Pi^{'}$ 是 NP 难的，如果 Π 是 NP 难的且 $\Pi \in NP$ ，则称 Π 是 NP 完全的。

性质：

1. 如果存在 NP 难的问题 $\Pi \in P$ ，则 $P = NP$
2. 如果存在 NP 难的问题 $\Pi^{'}$ ，使得 $\Pi^{'} \leq_p \Pi$ ，那么 Π 是 NP 难的。

于是想要证明一个问题 Π 是 NP 难的，只需要找到一个合适的已知 NP 难的问题 $\Pi^{'}$ ，并证明 $\Pi^{'} \leq_p \Pi$ 即可。证明了 Π 是 NP 难的之后，我们就有很大把握可以认为这个问题是难解的，因此可以放弃寻找多项式时间的算法。