# Chapter 1 计算复杂性

---



## 最大数：

1.最大数是一个数值，不是存储数值用的字节数

2.最大数可以是规模的指数函数，也可以是多项式函数

TSP最大数是$B=\max\{n,\max_{i,j}c_{i,j}\}$,是实例规模$n+log_2(L)$的一个指数函数

## Euler 图

给定一个图与边，Euler图问题是指是否存在一条通过所有边且每条边仅通过1次的通路（注意不是环路，因为并不要求起点终点相同）。给定一个图，判断它是否为Euler图的问题称为Euler图问题。

含有n个顶点图，用邻接矩阵存储的实例规模是$n^2$，最大数是n，最大数是实例规模的多项式函数

## 时间复杂度：

用算法执行过程中所需要的加，乘，比较，赋值等基本运算次数表示算法所用的时间。

一般来说，算法的时间复杂度是关于实例规模$n$的一个函数$f(n)$，它表示用该算法求解所有规模为n的实例中所需的基本运算次数最多的那个实例的基本运算次数。

## 多项式时间算法

如果算法的时间复杂度是实例规模的多项式函数，也就是说$f(n) = O(poly(n))$，那么称它为多项式时间算法，不能这样限制时间复杂度的算法称为指数时间算法。

## 伪多项式时间算法

如果某个算法的时间复杂度是实例规模n与最大数B的二元多项式函数$f(n) = O(poly(n, B))$,但是它不是实例规模n的一元多项式函数，则称它为伪多项式时间算法

## 以背包问题为例：

n件物品，物品j的价值为$p_j$，大小为$w_j$,背包容量为C

### 实例规模

规模是什么呢？（注意存储的时候对一个数存储一般是$\lfloor log_2 k \rfloor + 1$(向下取整后加一)+1再一个加一表示结束的时候的占位符，一般是个空格）

首先对所有的价值进行区分：

$\sum_{i=1}^{n} (\lfloor log_2(p_j) \rfloor + 2 + \lfloor log_2(w_j) \rfloor + 2) + \lfloor log_2(C) \rfloor + 2$

如果我们取最大数为L，那么实例规模大概就能看成是$n + \lceil log_2 L \rceil$

最大数是实例规模的指数函数，最大数是L

### 算法的时间复杂性举例：

这个时候，比如说

时间复杂性为$O(n^2), O(nlogB)$都是多项式时间算法，因为里面的都是实例规模的多项式函数

复杂度为$O(n!), O(2^n B), O(n2^B)$等的算法都是指数时间算法

而复杂度为$O(nL), O(Llogn), O(L)$等的算法都是伪多项式时间算法（实例规模与最大数的多项式函数，注意最大数不是存储字节，是数）

一般称多项式算法为高效算法

而如果最大数不是很大的话，那么伪多项式时间算法也是高效的。比如图问题实例规模是$n^2$，最大数是n（有向无权图）

### 指派问题的问题规模

n个任务m个人，$p_i j$

实例规模仍然为$n + log_2 L$

## P，NP问题

### P

确定性图灵机多项式时间可解问题类称为P类，简记为P

非确定性图灵机多项式时间可解问题类称为NP，这类问题中，图灵机的状态转移函数可能是多值的，确定性图灵机是一种特殊的非确定性图灵机。

一般在$P \neq NP$空间下进行问题讨论。

### NP类定义：

基于非确定性算法的NP类定义：

对于一个判定问题，如果存在一**非确定性算法**，使得对任何一个**答案为是的实例**，这个算法可以：

1.猜想出这个实例的一个可行解（也就是说实例有解）

2.可行解的规模不超过实例规模的多项式

3.能在实例规模的多项式时间内验证猜想是否是正确的。

则称该问题属于NP类

例子：

**证明TSP的判定形式$\in NP$**

首先，可以猜想出TSP实例的一个可行解$\pi$，然后给这个$\pi$标记为是

其次，$\pi$可以按照经过城市的标号用n个数字来表示，比如$12347651$这种，因此可行解的规模为$nlogn$

实例规模为$n+log_2 L$。可行解规模肯定小于$n^2$，因此确实是实例规模的多项式函数。

多项式时间内验证是显然的

这样看来，也有可能是问题不在NP类中。也就是说有不可判断问题

而P问题中，答案为是的实例所对应的解可以直接求出来，因此$P \subset NP$

那么，NP类中最难的问题是什么问题呢？

## NP-C

NP类中最难的问题的子集称为NP-完全类问题,一般如果NP-C类问题中有一个问题有多项式时间算法，那么NP类的所有问题都有多项式时间算法

## 归约

假设有两个判定问题$A_1, A_2$,如果对于$A_1$的任意实例$I_1$,我们都可以在**多项式时间内构造** 出$A_2$的一个实例$I_2$，使得$I_1$的答案为是，当且仅当$I_2$的答案也为是，那么$I_1$可以多项式归约到$I_2$

一般来说，这个意味着求解问题$A2$并不会比$A_1$容易。因为此时如果我们解出了A1，那么只能得到A2部分解，如果解出了A2，那么A1就能接出来。

归约问题具有传递性

## Hamilton圈

经过图的所有顶点**恰好一次** 的**圈** 称为Hamilton圈。存在Hamilton圈的图称为Hamilton图

Hamilton图问题(HC)：

判断图G是否为一Hamilton图

跳马图的Hamiltonian图问题

## 一个简单的Hamilton图归约：

$HC <_m^p TSP$

任给一个$HC$问题的实例$I_HC: G = (V, E)$

构造TSP判定形式的实例$I_{TSP}$

城市数目$n = |V|$

城市间距离：如果图中有边，距离为1，无边距离为2

整数阈值$M = |V|$

$I_{HC}$的答案为是当且仅当$I_{TSP}$的答案也为是

（怎么证明呢？思路是证明G中存在一个Hamilton圈当且仅当存在路称不超过M的环游）

因为是充要条件，要证明2边

如果存在一个Hamilton圈，则该圈的环游总长度恰好为V

假设环游总长度不超过V，那么因为环游需要经过|V|个城市，所以两个城市的边长肯定都是1，因此所对应的G中两个顶点肯定有边连接，因此所有这些边组成了一个Hamilton圈

## NP-C问题：

假设$A \in NP$,且对于任意的$A' \in NP$,A'可以归约到A，那么称A为NP-完全问题。

所有NP完全问题的集合称为NP-完全类，记作$NP - C$。

$NP - C$类是NP类的一个子类，包含了NP类中最难的问题

因此，如果NP-C类中有一个问题有多项式时间算法，那么NP类中的所有问题都有多项式时间算法

我们可以看到，从定义证明NP-C是困难的。但是我们可以用归约的传递性来证明

NP-完全性判定定理：

如果$A \in NP$,且存在某个$A^C \in NP^C$,$A^C \leq_m^p A, A \in NP - C$

因此我们需要寻找第一个NP-完全问题

## SAT问题

Boolean变量

若干Boolean变量用运算符与括号按合取顺序排列，而合取范式就是指若干个子句的合取。

可满足性问题：

给定一个合取范式，是否存在变量的一种赋值，使得表达式的值为真

首先SAT属于NP

一个问题实例，可行解规模肯定不超过问题实例的多项式函数，而验证一定是多项式时间内验证的，因此$SAT \in NP$

SAT问题是第一个NP-完全问题

Karp归约可以得到，3SAT，整数规划，背包问题，划分问题，HC问题都是NP-完全问题

因此有了这些之后我们就可以对NP完全问题进行证明

问题A的NP-完全性证明

首先证明A是NP，然后证明有一个NP完全问题可以归约到A

3SAT问题：

任意子句包含的文字数目不超过3

$SAT <=_m^p 3SAT$

构造一个3SAT实例：

若$c_i$中所含的文字数不超过3，那么$F_3$也含有子句$c_1$

如果$c_i = l_{i1} \cup l_{i2} \cup ... \cup l_{ik}$

构造新子句
$C_i = (l_{i1} \cup l_{i2} \cup y_{i1}) \cap (!y_{i1} \cup l_{i3} \cup y_{i2}) \cap (!y_{i2} \cup l_{i4} \cup y_{i3}) \cap ... \cap (!y_{i,k-4} \cup l_{i_{l-2}} \cup y_{i,,k-3}) \cap (!y_{i,k-3} \cup l_{i_{k-1}} \cup l_{i_k})$

y系列都是不出现在别处的新变量

如果FSAT的答案是是，那么存在一种赋值使得任何一个子句是真。那么$c_i$至少有一个文字是真的，不妨设$l_{i_j}$.

那么存在一种赋值使得$C_i$也为真

存在。

反过来，如果这个F3SAT的实例答案为真，那么存在一个赋值，使得F3SAT每个子句都是真的。假设此时存在一个ci是假的，那么ci中所有都是假的，证明此时F3SAT有个子句也是假的。如何证明呢？

因为在当前赋值下要满足2个条件，一个是所有的子句都得是真的，另一个是对ci的所有l都是假的

那么我们只有这种赋值了。。

此时一定会出现问题，也就是说此时必会存在Ci是假的，那么在当前赋值下F3SAT的实例答案为假，矛盾，假设错误，成立

## 划分问题：

给定一个正整数集合A，问是否可以把A划分成两个不交的子集，使得每个子集元素和是A的元素和的一半。

至少和NP完全问题一样难的问题称为NP难问题。因此NP-C也是NP-hard问题。如果一个问题是NP-h，那么一定存在一个NP-C可以归约到它。

## 子问题：

在某一个问题的实例上增加一些限制得到了子问题。

子问题一定不会比原问题难。也就是说如果原问题是NP-C，那么子问题不一定是NP-C

## 强NP-C问题

（这个问题的提出主要是把原问题的范围扩大到伪多项式时间算法，即在强NP-C问题下，连伪多项式都不存在）

假设A是一个NP-C问题，且存在多项式的函数p，使得A的某个所有实例满足$Max(I) \leq p(size(I))$的子问题A‘

**是NP完全的**，那么称A是强NP完全的。

在P不等于NP的假设下，任何强NP-C问题不存在伪多项式时间算法。

我们可以用伪多项式时间归约来从一个强NP来证明另外一个问题的强NP

## 伪多项式时间归约

假设我们有判定问题 A1，A2，如果对A1的任意一个实例I1，可以在$O(poly(size(I_1)), Max(I_1))$的时间内构造出A2的一个实例I2,使得：

I1的答案为是当且仅当I2的答案也为是

$Max(I_2) <= p_1(size(I_1), Max(I_1))$

(最大数的规模没有显著增大)

$size(I_1) \leq p_2(size(I_2))$

（实例的规模也没有显著减小）

那么A1可以伪多项式归约到A2，$A_1 \leq_m^{pp} A_2$

伪多项式时间归约：

假设A1是强NP-C，$A_2 \in NP, A_1 \leq_m^{pp} A_2$，那么$A_2$也是强NP-C问题

因为$A_1$是强NP-C问题，那么存在一个子问题$A_1^s$，使得它的实例集中的所有实例$I_1$满足

$Max(I_1) \leq p(size(I_1))$,且子问题是NP-C的

因为$I_2 = f(I_1)$的构造可以在伪多项式，此时是多项式的时间内构造，因此f是一个多项式归约

也就是说$A_2^s$这个东西可以被多项式归约到，因此此时$A_2^s$是NP-C的

只要能证明$A_2^S$中的所有实例都满足最大数小于实例规模，那么我们可以证明A2是强NP——C

怎么证明呢？这里就需要用到所谓的伪多项式归约的2个公理了，也就是实例规模不会显著变小，最大数不会显著变大就行：

$Max(I_2) <= p_1(size(I_1), Max(I_1))$

(最大数的规模没有显著增大)

$size(I_1) \leq p_2(size(I_2))$

（实例的规模也没有显著减小）

我们的目的是$Max(I_2) <= p(size(I_2))$

$Max(I_2) <= p1(size(I_1), Max(I_1)) <= p1(size(I_1), p(size(I_1)) <= p_1'(size(I_1)) <= p_2'(size(I_2))$

因此这里就证明出来了

如果我们构造的实例规模比如说

$size(I_2) = O(log(size(I_1)))$

此时不一定成立，也就是实例规模显著缩小了。

因此我们需要找第一个强NP-C问题

3-划分问题

NP-C中不是强NP——C的称普通NP-C，也就是存在伪多项式时间算法

背包，划分都是普通意义下的NP-C问题。

而TSP是强NP——C问题，至少和NP——C一样难的问题称强NP-难问题

## 线性规划与整数规划

满足所有约束条件的点称为可行解，可行点的集合称为可行域

单目标：

$$\begin{gathered} min\ f(x) \\ s.t.\ g_j(x) \geq 0, j=1,\ldots,s \\ h_l(x)=0, l=1,\ldots,t \\ x \in R^n \end{gathered}$$

最优值

线性规划与非线性规划

整数规划

0-1规划

线性规划与标准型

单纯形法（求解线性规划问题，指数时间算法）

多项式时间算法（椭球法，内点法

整数线性规划是NP完全的

归约为划分问题

划分问题$\leq_m^p$整数线性规划

任取划分问题实例$A=\{a_1,a_2,\ldots,a_n\}$,考虑整数规划
$\sum a_j x_j = 1/2 \sum a_j, x_j \in \{0,1\}$.

整数规划存在可行解iff划分问题实例答案为是

然后我们需要证明整数线性规划是个NP问题

NP问题：

猜想出一个可行解

证明可行解不超过实例规模多项式

在多项式时间内验证

如果线性不等式组存在整数解，那么必然有一个整数解的规模不超过实例规模多项式

割平面法，分支定界法

松弛：