# 第一章 算法分析与NP-完全问题

## §1.1 算法分析

### 一．程序运行时间的测量

影响程序运行时间的因素：

1.程序的输入的长度

2.编译程序生成目标代码的质量

3.计算机指令的性质和速度

4.算法的时间复杂性

### 二．评价算法运行时间的标准

运行时间作为输入长度的函数T(n)

#### 1.最坏运行时间：

算法对具有长度n的任何输入的最长运行时间。

#### 2.平均运行时间：

即在“平均”输入下，算法的运行时间。通常我们假设给定长度的各种输入概率相同。平均运行时间是在这个假设下，运行时间的数学期望值。

### 三．记号

1.记号

设g(n)是一给定函数，用表示函数的集合：

存在正的常数使得当有。我们写，表示。

例1：。

取

当时，。

2.记号O

设g(n)是一给定函数，用表示函数的集合：

存在正的常数C和使得当，有

。我们写表示。显然有

。

例2：同例1，有，同时也有,

但是不成立。

3.记号

设g(n)是一给定函数，用表示函数的集合：

, 使得当有

。

4.记号o

设g(n)是一给定函数，用表示函数的集合：

，存在常数 使得当,

有.

例3: ，但。

5.记号

设g(n)是一给定函数，用表示函数的集合：

对于任意正的常数C，存在常数，使得当

，有。

例4：。

### 四．运行时间增长率的比较

例5：有两个算法运行时间分别为和是否的算法比

的算法好？

设

当时，算法2比算法1运行得快。当n充分大时，

故当n充分大时，算法1比算法2快。

例6：有四个算法运行时间增长率如下：（见图1.1）

## §1.2 算法分析技术

### 一．计算程序运行时间

1.加法规则：

如果分别是两段程序的运行时间，，那么程序段的运行时间为

，时间复杂度为

这因为存在正的常数，使得当时，

；当时，。令，当时，

。

2.乘法规则：

如果，那么为

。乘法规则主要用于循环结构的时间分析。

\*

### 二．例子：

PROCEDURE Bubblesort(VAR A : array[1..n]of integer);

VAR

i, j, temp : integer;

BEGIN

FOR i := 1 TO n-1 DO

FOR j := n DOWNTO i+1 DO

IF A[j-1] > A[j] THEN

BEGIN

temp := A[j-1];

A[j-1] := A[j];

A[j] := temp;

END

END;

算法分析：

(4), (5), (6)为 O(1)+O(1)+O(1) = O(max(1, 1, 1)) = O(1);

(3)取最坏情况，执行条件判断O(1)，语句内部O(1), 结果为O(1);

(2)执行循环控制条件为O(1)，内部O(1)，共循环n-i次，由乘法规则

(1)将各次循环的时间加起来：

## §1.3 确定图灵机

### 一．确定图灵机

1.构成：确定型单带图灵机(DTM)由有限状态控制器，读写头和一条带组成：这条带由标有的带方格的双向无穷序列构成。（见图1.2）

2.DTM定义：

一个DTM(程序)包括：

(1)有穷的带符号集，包括输入符号子集和一个特殊的空白符#；

(2)有穷状态集合Q, 包括一个特殊的初始状态和两个特殊的停机状态；

(3)转移函数：

设。

DTM的输入放在从1到|x|的带方格中。初始时，所有其它方格中放空白符#。DTM初始处于状态，读写头指向带方格1.

当前有穷状态控制器处于状态q, 读头所指方格上符号为s。这时状态改为q’，读头所指的方格写上s’，再根据决定读写头左移()或右移()一格。

运行到某一步，当时，DTM给出答案“是”并停机；当时，DTM给出答案“否”并停机。

3.例子：给出一个DTM

q 0 1 #

设我们定义DTM的格局表示当前图灵机处于q状态，带上的符号串为，当前读写头指向a所在的这个方格。

上例中DTM对于输入10100的计算过程可表示如下：

4.DTM接受的语言

接受x当且仅当输入x时，M停机在状态。

M识别的语言为：

在上例中：

DTM输入x后，有三种可能：

(1)停机在状态；

(2)停机在状态；

(3)永远不停机。

\*对应算法的DTM程序对任何输入，它都停机。

### 二．判定问题

1.定义：由于任何事物都可以用0和1编码，一个判定问题可以描述如下：已知

则给出答案“否”。

2.例子：

给定一个正整数N，问是否有正整数m使N = 4m?

在标准编码方案下，整数N用二进制数表示。一个整数可被4整除，当且仅当它的二进制数最后两位是00。故上例中的DTM可解本判定问题。

### 三．P问题类

1.图灵机的时间复杂性

确定型图灵机对输入x的时间复杂性指的是，从开始到停机为止的运行步数。

如果存在多项式p，使得，则图灵机(程序)

M叫做多项式时间的图灵机(程序)。

2.P问题类

P = {L | 有多项式时间的DTM程序M使得}。

## §1.4 非确定型图灵机

### 一．非确定图灵机的定义

一个非确定型图灵机NDTM M包括：

(1)与DTM相同

(2) 与DTM相同

(3) 转移函数

\*解释的多值性，一次可做多种运算。

### 二．非确定图灵机的另一种定义

NDTM也可以看作除了多一个“猜想模块”外，其它与DTM相同。这个“猜想模块”带有“猜想头”，可对带写入“猜想”。

计算分为两个阶段，一是猜想阶段，一是检验阶段。对任一输入，NDTM首先给出(无穷多种猜想中的)一个猜想，然后按DTM的动作验证该猜想是否答案。检验阶段若停在状态，则回答“是”。检验阶段也可能停在状态或不停机。

### 三．NP问题类

NDTM对给定的输入字符串x有无穷多种可能的计算，对中每一个可能的猜想字符串有一个计算。如果这些计算中至少有一个是接受的计算(停在状态)，则称这个NDTM接受x。

NDTM识别的语言为：

NDTM M接受字符串x所需的时间定义为：M关于x的所有接受计算中猜想阶段和检验阶段直到进入停机状态时为止所需的步数的最小值。

。

若存在多项式p，使得是一个多项式时间的NDTM。

NP = {L| 存在多项式时间的NDTM M, 使得}。

\*NP问题类是多项式时间可验证解的问题类。

## §1.5 P问题与NP问题的关系

### 一．P与NP的关系

#### 1.

确定图灵机是特殊的非确定图灵机，故。

#### 2.用确定图灵机解决NP问题

##### 定理1.1：如果，那么存在一个多项式p使得能用时间复杂性为的确定型算法解决。

证明：设非确定图灵机字母表为，且。对输入x，其长度为

n。若非确定型图灵机在不超过q(n)步内对x给出肯定的判断，则NDTM给出的猜想长度不超过q(n)。由于猜想每位的字母有k种可能，故猜想的全体不超过个。每个猜想在步内检验完毕，故全体检验完毕的时间复杂度以为上界。适当选取多项式p，则

。

\*人们普遍认为

\*但以上结论无法证明

### 二．P,NP与co-NP的关系

#### 1.co-NP的定义

由NP类的定义，我们不知是否蕴含L的补集。定义

。

#### 2.关系的几种可能性

(见图1.4)

注意有以下结论：

1.

2.如果

\*人们认为情形(a)可能性最小，(d)可能性最大。

### 三．多项式变换和NP完全性

#### 1.多项式变换

从语言到语言的多项式变换是满足下述两个条件的函数 :

(1)存在计算f的多项式DTM;

(2)。

如果存在一个从的多项式变换，则记作。

#### 2.多项式变换的几个性质

##### 定理1.2：如果，那么。

证明：设的字母表，是从的多项式变换。设为计算f的多项式时间的DTM，为识别的多项式时间的DTM。构造识别的多项式时间的DTM 如下：对输入

首先按方法计算，然后按方法确定是否。因为当且仅当，故是识别的DTM。如果是限制运行时间的多项式函数，那么，的运行时间为，它不超过的某个多项式。证毕。

##### 定理1.3：若，那么。

证：设分别是语言的字母表，是从到的多项式变换，是从的多项式变换。定义为：，那么，f是从的多项式变换。显然，当且仅当。可以用类似定理1.2的证明，证明可用多项式时间的DTM计算f。

#### 3.多项式等价性

若，则称是多项式等价的。

多项式等价性是一个等价关系。P类是其中的一个等价类。

#### 4.NP完全类(NPC)

如果语言，并且对所有其它语言，则L称为是。

\*如果判定问题，并且对所有其它判定问题，则NP-完全的。

NP-完全问题是NP中最难的问题。

\*NP与P的关系图可描述如图1.5

\*如果，则NP\P。

#### 5.NP-完全问题的证明方法

##### 定理1.4：如果属于NP，是NP-完全的，且，那么也是NP-完全的。

证明：因为，只要证：即可。由于是

NP-完全，，且。由的传递性，。

\*定理1.4给出了证明一个问题是NP-完全的方法：

(1)

(2) 某个已知的NP-完全问题可多项式变换到。