吴 铤

2003年12月18日

- 时间复杂度

- 1 时间复杂度
- 2 P类

- 1 时间复杂度
- 2 P类
- 3 NP类

- 1 时间复杂度
- 2 P类
- 3 NP类
- 4 NP完全

### 定义1 (时间复杂度)

令M是在所有输入上都停机的确定型图灵机,M的运行时间或时间复杂度是指函数 $f: \mathbb{N} \to \mathbb{N}$ ,函数值f(n)表示M在所有长度为n的输入上运行所经过的最大步数。

如果f(n)是M的运行时间,则称M在时间f(n)内运行,M是f(n)时间图灵机。

- 最坏情况分析: 某特定长度的所有输入上的最长运行时 间
- 平均情况分析: 某特定长度的所有输入上的运行时间的 平均值



令M是在所有输入上都停机的确定型图灵机,M的运 行时间或时间复杂度是指函数 $f: \mathbb{N} \to \mathbb{N}$ , 函数值f(n)表 示M在所有长度为n的输入上运行所经过的最大步数。



令M是在所有输入上都停机的确定型图灵机,M的运 行时间或时间复杂度是指函数 $f: \mathbb{N} \to \mathbb{N}$ , 函数值f(n)表 示M在所有长度为n的输入上运行所经过的最大步数。

如果f(n)是M的运行时间,则称M在时间f(n)内运 行,M是f(n)时间图灵机。



# 定义1(时间复杂度)

令M是在所有输入上都停机的确定型图灵机,M的运 行时间或时间复杂度是指函数 $f: \mathbb{N} \to \mathbb{N}$ , 函数值f(n)表 示M在所有长度为n的输入上运行所经过的最大步数。

NP类

如果f(n)是M的运行时间,则称M在时间f(n)内运 行,M是f(n)时间图灵机。

- 最坏情况分析: 某特定长度的所有输入上的最长运行时 间

令M是在所有输入上都停机的确定型图灵机,M的运 行时间或时间复杂度是指函数 $f: \mathbb{N} \to \mathbb{N}$ , 函数值f(n)表 示M在所有长度为n的输入上运行所经过的最大步数。

如果f(n)是M的运行时间,则称M在时间f(n)内运 行,M是f(n)时间图灵机。

- 最坏情况分析: 某特定长度的所有输入上的最长运行时 间
- 平均情况分析: 某特定长度的所有输入上的运行时间的 平均值



设函数 $f,q:\mathbb{N}\to\mathbb{R}^+$ , 若存在正整数 $c,n_0$ , 使得 

$$f(n) \le cg(n),$$

NP类



设函数 $f,q:\mathbb{N}\to\mathbb{R}^+$ , 若存在正整数 $c,n_0$ , 使得 

$$f(n) \le cg(n),$$

NP类

- $f(n) = 5n^3 + 2n^2 + n + 1$ :  $f(n) = O(n^3)$



设函数 $f,g: \mathbb{N} \to \mathbb{R}^+$ ,若存在正整数 $c,n_0$ ,使得 当 $n \geq n_0$  时,有

$$f(n) \le cg(n),$$

NP类

- $f(n) = 5n^3 + 2n^2 + n + 1$ :  $f(n) = O(n^3)$
- $f(n) = 3n \log_2 n + 5n \log_2 \log_2 n$ :  $f(n) = O(n \log n)$
- 多项式界:  $O(n^c)$
- 指数界:  $O(2^{(n^{\delta})}), \delta$ 是大于零的实数。



设函数 $f,q:\mathbb{N}\to\mathbb{R}^+$ , 若存在正整数 $c,n_0$ , 使得 

$$f(n) \le cg(n),$$

NP类

- $f(n) = 5n^3 + 2n^2 + n + 1$ :  $f(n) = O(n^3)$
- $f(n) = 3n \log_2 n + 5n \log_2 \log_2 n$ :  $f(n) = O(n \log n)$



设函数 $f,g:\mathbb{N}\to\mathbb{R}^+$ ,若存在正整数 $c,n_0$ ,使得当 $n\geq n_0$ 时,有

$$f(n) \le cg(n),$$

NP类

- $f(n) = 5n^3 + 2n^2 + n + 1$ :  $f(n) = O(n^3)$
- $f(n) = 3n \log_2 n + 5n \log_2 \log_2 n$ :  $f(n) = O(n \log n)$
- 多项式界:  $O(n^c)$
- 指数界:  $O(2^{(n^{\delta})}), \delta$ 是大于零的实数。



设函数 $f,q:\mathbb{N}\to\mathbb{R}^+$ , 若存在正整数 $c,n_0$ , 使得 

$$f(n) \le cg(n),$$

NP类

- $f(n) = 5n^3 + 2n^2 + n + 1$ :  $f(n) = O(n^3)$
- $f(n) = 3n \log_2 n + 5n \log_2 \log_2 n$ :  $f(n) = O(n \log n)$
- 多项式界:  $O(n^c)$



设函数 $f,g: \mathbb{N} \to \mathbb{R}^+$ ,若存在正整数 $c,n_0$ ,使得 当 $n \geq n_0$  时,有

$$f(n) \le cg(n),$$

NP类

- $f(n) = 5n^3 + 2n^2 + n + 1$ :  $f(n) = O(n^3)$
- $f(n) = 3n \log_2 n + 5n \log_2 \log_2 n$ :  $f(n) = O(n \log n)$
- 多项式界:  $O(n^c)$
- <mark>指数界</mark>:  $O(2^{(n^{\delta})}), \delta$ 是大于零的实数。



时间复杂度 0000000 大O和小o记法

### 定义3

设函数 $f, g: \mathbb{N} \to \mathbb{R}^+$ , 若

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0,$$

则称
$$f(n) = o(g(n))$$
。

$$\sqrt{n} = o(n)$$

$$n = o(n \log(\log n))$$

$$n \log n = o(n^2)$$

设函数 $f,g:\mathbb{N}\to\mathbb{R}^+$ , 若

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0,$$

则称f(n) = o(q(n))。

- $\sqrt{n} = o(n)$

时间复杂度 0000000 大O和小o记法

设函数 $f, g: \mathbb{N} \to \mathbb{R}^+$ , 若

$$\lim_{n\to\infty}\frac{f(n)}{g(n)}=0,$$

则称f(n) = o(q(n))。

- $\sqrt{n} = o(n)$
- $n = o(n \log(\log n))$

### 设函数 $f,g:\mathbb{N}\to\mathbb{R}^+$ ,若

$$\lim_{n\to\infty}\frac{f(n)}{g(n)}=0,$$

则称
$$f(n) = o(g(n))$$
。

- $n = o(n \log(\log n))$
- $n \log n = o(n^2)$

时间复杂度 ○●○○○○○ 大*〇*和小*o*记法

M对于输入串w,执行

- 扫描带。如果在1的右侧发现0,则拒绝; (O(n))
- 2 如果带上既有0也有1,则重复以下操作
  - 扫描带 删除一个()和一个1:
- 图 如果所有1/0都被删除后还有0/1,则拒绝,否则接受。 (O(n))

即 $A \in TIME(n^2)$ ,其中TIME(t(n))表示为由O(t(n))时间的图灵机判定的所有语言构成的集合,称为时间复杂类。



NP类

# 例1 (语言 $A = \{0^k 1^k | k \ge 0\}$ 对应图灵机M的算法分析)

### M对于输入串w,执行

- 扫描带。如果在1的右侧发现0,则拒绝; (O(n))
- ② 如果带上既有0也有1,则重复以下操作  $(n/2 \cdot O(n) = O(n^2))$ 
  - 扫描带,删除一个0和一个1;
- 图 如果所有1/0都被删除后还有0/1,则拒绝,否则接受。 (O(n))

即 $A \in TIME(n^2)$ ,其中TIME(t(n))表示为由O(t(n))时间的图灵机判定的所有语言构成的集合,称为时间复杂类。



- 扫描带。如果在1的右侧发现0,则拒绝;(O(n))



- Ⅰ 扫描带。如果在1的右侧发现0,则拒绝;(O(n))
- 2 如果带上既有0也有1,则重复以下操作



- 扫描带。如果在1的右侧发现0,则拒绝;(O(n))
- 2 如果带上既有0也有1,则重复以下操作
  - 扫描带、删除一个0和一个1;



- 扫描带。如果在1的右侧发现0,则拒绝;(O(n))
- 2 如果带上既有0也有1,则重复以下操作

- 扫描带、删除一个0和一个1:
- 3 如果所有1/0都被删除后还有0/1,则拒绝,否则接受。



NP类

- 扫描带。如果在1的右侧发现0,则拒绝;(O(n))
- 2 如果带上既有0也有1,则重复以下操作

- 扫描带、删除一个0和一个1:
- 3 如果所有1/0都被删除后还有0/1,则拒绝,否则接受。



NP类

- 扫描带。如果在1的右侧发现0,则拒绝;(O(n))
- 2 如果带上既有0也有1,则重复以下操作  $(n/2 \cdot O(n) = O(n^2))$ 
  - 扫描带、删除一个0和一个1:
- 3 如果所有1/0都被删除后还有0/1,则拒绝,否则接受。



- 扫描带。如果在1的右侧发现0,则拒绝;(O(n))
- 2 如果带上既有0也有1,则重复以下操作  $(n/2 \cdot O(n) = O(n^2))$ 
  - 扫描带、删除一个0和一个1:
- 3 如果所有1/0都被删除后还有0/1,则拒绝,否则接受。 (O(n))



NP类

M对于输入串w,执行

- 扫描带。如果在1的右侧发现0,则拒绝;(O(n))
- 2 如果带上既有0也有1,则重复以下操作  $(n/2 \cdot O(n) = O(n^2))$ 
  - 扫描带、删除一个0和一个1:
- 3 如果所有1/0都被删除后还有0/1,则拒绝,否则接受。 (O(n))

即 $A \in TIME(n^2)$ ,其中TIME(t(n))表示为由O(t(n))时间



NP类

M对于输入串w,执行

- 扫描带。如果在1的右侧发现0,则拒绝; (O(n))
- 2 如果带上既有0也有1,则重复以下操作  $(n/2 \cdot O(n) = O(n^2))$ 
  - 扫描带、删除一个0和一个1:
- 3 如果所有1/0都被删除后还有0/1,则拒绝,否则接受。 (O(n))

即 $A \in TIME(n^2)$ ,其中TIME(t(n))表示为由O(t(n))时间 的图灵机判定的所有语言构成的集合, 称为时间复杂类。



- 扫描带。如果在1的右侧发现0, 则拒绝;
- 2 如果带上既有0也有1,则重复以下操作

  - 再次扫描带,从第一个0开始,隔一个删除一个0;
  - 再次扫描带, 从第一个1开始, 隔一个删除一个1;
- B 如果带上没有0和1,则接受,否则拒绝。

NP类

- 扫描带。如果在1的右侧发现0,则拒绝;
- 2 如果带上既有0也有1,则重复以下操作
  - 扫描带,检查剩余的0,1的总数是奇数还是偶数。若是奇数 则拒绝·
  - ❷ 再次扫描带,从第一个0开始,隔一个删除一个0;
  - 再次扫描带, 从第一个1开始, 隔一个删除一个1;
- B 如果带上没有0和1,则接受,否则拒绝。

NP类

- 扫描带。如果在1的右侧发现0、则拒绝;

NP类

 $M_1$ 对于输入串w,执行

- Ⅱ 扫描带。如果在1的右侧发现0, 则拒绝;
- 2 如果带上既有0也有1,则重复以下操作
  - 扫描带,检查剩余的0,1的总数是奇数还是偶数。若是奇数,则拒绝;
  - ② 再次扫描带,从第一个0开始,隔一个删除一个0;
  - 圆 再次扫描带,从第一个1开始,隔一个删除一个1;
- B 如果带上没有0和1,则接受,否则拒绝。

时间复杂度

○○○●○○○ 分析算法

NP类

- 扫描带。如果在1的右侧发现0,则拒绝;
- 2 如果带上既有0也有1,则重复以下操作
  - 扫描带,检查剩余的0,1的总数是奇数还是偶数。若是奇数,则拒绝;
  - ☑ 再次扫描带,从第一个0开始,隔一个删除一个0;
  - B 再次扫描带,从第一个1开始,隔一个删除一个1;
- 3 如果带上没有0和1,则接受,否则拒绝。

### 例2 (语言 $A = \{0^k 1^k | k > 0\}$ 对应图灵机 $M_1$ 的算法分析)

NP类

- 扫描带。如果在1的右侧发现0、则拒绝;
- 2 如果带上既有0也有1、则重复以下操作
  - Ⅰ 扫描带, 检查剩余的0,1的总数是奇数还是偶数。若是奇数、 则拒绝;
  - 2 再次扫描带、从第一个0开始、隔一个删除一个0;

- 扫描带。如果在1的右侧发现0、则拒绝;
- 2 如果带上既有0也有1、则重复以下操作
  - Ⅰ 扫描带, 检查剩余的0,1的总数是奇数还是偶数。若是奇数、 则拒绝:
  - 2 再次扫描带、从第一个0开始、隔一个删除一个0;
  - 3 再次扫描带、从第一个1开始、隔一个删除一个1;

NP类

- 扫描带。如果在1的右侧发现0、则拒绝;
- 2 如果带上既有0也有1,则重复以下操作
  - 扫描带, 检查剩余的0.1的总数是奇数还是偶数。若是奇数, 则拒绝:
  - 2 再次扫描带、从第一个0开始、隔一个删除一个0;
  - 3 再次扫描带、从第一个1开始、隔一个删除一个1;
- 3 如果带上没有0和1,则接受,否则拒绝。

- 0 0 0 0 0 0 1 1 1 1 1 1 1 (6个)
- 0 0 0 1 1 1 (3<sup>↑</sup>)
- 0 1 (1个)

个数序列为奇、偶、奇、奇 反转编码后为(1101)2=13

- 每个步骤所需时间O(n)
- 循环次数 $1 + \log_2 n$
- $M_1$ 运行时

闰 $O(n) + (1 + \log_2 n)O(n) = O(n \log n), A \in \text{TIME}(n \log n)$ 

个数序列为奇、偶、奇、奇 反转编码后为 $(1101)_2=13$  运行时间分析:

- 每个步骤所需时间O(n)
- 循环次数1 + log<sub>2</sub> n
- $M_1$ 运行时

闰 $O(n) + (1 + \log_2 n)O(n) = O(n \log n), A \in TIME(n \log n)$ 

个数序列为奇、偶、奇、奇 反转编码后为 $(1101)_2=13$  运行时间分析:

- 每个步骤所需时间O(n)
- 循环次数1 + log<sub>2</sub> n
- $M_1$ 运行时

闻 $O(n) + (1 + \log_2 n)O(n) = O(n \log n)$ , $A \in \text{TIME}(n \log n)$ 

- (13个)
- 0 0 0 0 0 0 1 1 1 1 1 1 1 (6↑)

- (13个)
- 0 0 0 0 0 0 1 1 1 1 1 1 1 1 (6**个**)

- 0 0 0 0 0 0 1 1 1 1 1 1 1 (6**个**)
- 0 0 0 1 1 1 (3↑)
- 0 1 (1个)

个数序列为奇、偶、奇、奇 反转编码后为(1101)<sub>2</sub>= 13 运行时间分析:

- 每个步骤所需时间O(n)
- 循环次数 $1 + \log_2 n$
- *M*<sub>1</sub>运行时

 $|\mathbb{H}O(n) + (1 + \log_2 n)O(n) = O(n \log n), A \in \text{TIME}(n \log n)$ 

- (13个)
- 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 (6个)
- 0 0 0 1 1 1 (3**个**)

- (13个)
- 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 (6个)
- 0 0 0 1 1 1 (3**个**)
- 0 1 (1↑)

- (13个)
- 0 0 0 0 0 0 1 1 1 1 1 1 1 1 (6**个**)
- 0 0 0 1 1 1 (3**个**)
- 0 1 (1个)

- (13个)
- 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 (6个)
- 0 0 0 1 1 1 (3**个**)
- 0 1 (1个)

个数序列为奇、偶、奇、奇 反转编码后为 $(1101)_2 = 13$ 

- (13个)
- 0 0 0 0 0 0 1 1 1 1 1 1 1 1 (6**个**)
- 0 0 0 1 1 1 (3**个**)
- 01 (1个)

个数序列为奇、偶、奇、奇 反转编码后为 $(1101)_2 = 13$ 

- (13个)
- 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 (6个)
- 0 0 0 1 1 1 (3**个**)
- 01 (1个)

个数序列为奇、偶、奇、奇 反转编码后为 $(1101)_2 = 13$ 运行时间分析:

NP类

- 00000011111111 (6个)
- 0 0 0 1 1 1 (3**↑**)
- 0 1 (1**个**)

个数序列为奇、偶、奇、奇 反转编码后为(1101)<sub>2</sub>= 13 运行时间分析:

- 每个步骤所需时间*O*(*n*)
- 循环次数 $1 + \log_2 n$
- *M*<sub>1</sub>运行时
- $|\mathbb{I}|O(n) + (1 + \log_2 n)O(n) = O(n \log n), \quad A \in \mathrm{TIME}(n \log n)$

时间复杂度

○○○○ 分析算法

- 00000011111111 (6个)
- 0 0 0 1 1 1 (3**个**)
- 0 1 (1**个**)

个数序列为奇、偶、奇、奇 反转编码后为(1101)<sub>2</sub>= 13 运行时间分析:

- 每个步骤所需时间*O*(*n*)
- 循环次数1 + log<sub>2</sub> n
- *M*<sub>1</sub>运行时

时间复杂度

○○○○ 分析算法

- (13个) lacksquare
- 0 0 0 0 0 0 1 1 1 1 1 1 1 (6个)
- 0 0 0 1 1 1 (3**个**)
- 0 1 (1个)

个数序列为奇、偶、奇、奇 反转编码后为 $(1101)_2 = 13$ 运行时间分析:

- 每个步骤所需时间*O*(*n*)
- 循环次数1 + log<sub>2</sub> n
- *M*<sub>1</sub>运行时

 $\operatorname{Id} O(n) + (1 + \log_2 n)O(n) = O(n \log n), A \in \operatorname{TIME}(n \log n)$ 

时间复杂度

分析算法

- (13个)
- 0 0 0 0 0 0 1 1 1 1 1 1 1 (6个)
- 0 0 0 1 1 1 (3**个**)
- 01 (1个)

个数序列为奇、偶、奇、奇 反转编码后为 $(1101)_2 = 13$ 运行时间分析:

- 每个步骤所需时间*O*(*n*)
- 循环次数1 + log<sub>2</sub> n
- *M*<sub>1</sub>运行时 [闰 $O(n) + (1 + \log_2 n)O(n) = O(n \log n), A \in TIME(n \log n)$

时间复杂度

分析算法



# 例3 (语言 $A=\{0^k1^k|k\geq 0\}$ 对应双带图灵机 $M_2$ 的算法分析)\_\_\_\_

### $M_2$ 对于输入串w,执行

- 扫描带。如果在1的右侧发现0,则拒绝;
- 2 扫描带1上的0,直至第一个1时停止。将0复制到带2上;
- 图 扫描带1上的1直至输入的末尾。
  - 每次从带1上读到一个1, 就在带2上输出一个0;
  - 如果在读完1之前所有的0都被删除,则拒绝。
- 如果所有的0都被删除,则接受,否则拒绝。

在双带图灵机模型下, $A \in TIME(n)$ ,即其是线性时间。

注意: A的复杂度与选取的计算模型有关。



- 扫描带。如果在1的右侧发现0、则拒绝;



NP类

# 例3 (语言 $A = \{0^k 1^k | k \ge 0\}$ 对应双带图灵机 $M_2$ 的算法分析)

- 11 扫描带。如果在1的右侧发现0,则拒绝;
- 2 扫描带1上的0,直至第一个1时停止。将0复制到带2上;
- 3 扫描带1上的1直至输入的末尾。

- 4 如果所有的0都被删除,则接受,否则拒绝。
- 在双带图灵机模型下, $A \in TIME(n)$ ,即其是线性时间。 注音·A的复数度与进取的计算模型有关



 $M_2$ 对于输入串w,执行

- 1 扫描带。如果在1的右侧发现0,则拒绝;
- 2 扫描带1上的0,直至第一个1时停止。将0复制到带2上;
- 3 扫描带1上的1直至输入的末尾。
  - 每次从带1上读到一个1, 就在带2上输出一个0;
  - 如果在读完1之前所有的0都被删除,则拒绝。
- 如果所有的0都被删除,则接受,否则拒绝。

在双带图灵机模型下, $A \in TIME(n)$ ,即其是线性时间。



NP类

# 例3 (语言 $A = \{0^k 1^k | k \ge 0\}$ 对应双带图灵机 $M_2$ 的算法分析)

 $M_2$ 对于输入串w,执行

- 1 扫描带。如果在1的右侧发现0,则拒绝;
- 2 扫描带1上的0, 直至第一个1时停止。将0复制到带2上;
- 3 扫描带1上的1直至输入的末尾。
  - 每次从带1上读到一个1, 就在带2上输出一个0;
  - 如果在读完1之前所有的0都被删除,则拒绝。
- 如果所有的0都被删除,则接受,否则拒绝。

在双带图灵机模型下, $A \in TIME(n)$ ,即其是线性时间。

◆ロ > ◆園 > ◆夏 > ◆夏 > 夏 めのの

NP类

# 例3 (语言 $A = \{0^k 1^k | k \ge 0\}$ 对应双带图灵机 $M_2$ 的算法分析)

 $M_2$ 对于输入串w,执行

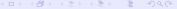
- 1 扫描带。如果在1的右侧发现0,则拒绝;
- 2 扫描带1上的0, 直至第一个1时停止。将0复制到带2上;
- 3 扫描带1上的1直至输入的末尾。
  - 每次从带1上读到一个1, 就在带2上输出一个0;
  - 如果在读完1之前所有的0都被删除,则拒绝。
- 如果所有的0都被删除,则接受,否则拒绝。

在双带图灵机模型下, $A \in TIME(n)$ ,即其是线性时间。 注音· 4的复杂度与进取的计算模型有关



NP类

- 扫描带。如果在1的右侧发现0、则拒绝;
- 2 扫描带1上的0、直至第一个1时停止。将0复制到带2上;
- 3 扫描带1上的1直至输入的末尾。
  - 每次从带1上读到一个1、就在带2上输出一个0;
  - 如果在读完1之前所有的0都被删除,则拒绝。
- 如果所有的0都被删除,则接受,否则拒绝。



NP类

 $M_0$ 对于输入串w,执行

- 扫描带。如果在1的右侧发现0、则拒绝;
- 2 扫描带1上的0、直至第一个1时停止。将0复制到带2上;
- 3 扫描带1上的1直至输入的末尾。
  - 每次从带1上读到一个1、就在带2上输出一个0;
  - 如果在读完1之前所有的0都被删除,则拒绝。
- △ 如果所有的0都被删除,则接受,否则拒绝。

在双带图灵机模型下,  $A \in TIME(n)$ , 即其是线性时间。



 $M_0$ 对于输入串w,执行

- 扫描带。如果在1的右侧发现0、则拒绝;
- 2 扫描带1上的0、直至第一个1时停止。将0复制到带2上;
- 3 扫描带1上的1直至输入的末尾。
  - 每次从带1上读到一个1、就在带2上输出一个0;
  - 如果在读完1之前所有的0都被删除,则拒绝。
- 如果所有的0都被删除,则接受,否则拒绝。

在双带图灵机模型下,  $A \in TIME(n)$ , 即其是线性时间。

注意: A的复杂度与选取的计算模型有关。



#### 定理1

### 设t(x)是一个函数, $t(n) \ge n$ , 则

- 每个t(n)时间的多带图灵机和某个 $O(t^2(n))$ 时间的单带图灵机等价。
- 每个t(n)时间的非确定型单带图灵机和某个 $2^{O(t(n))}$ 时间的确定型单带图灵机等价。
- 设N是非确定型图灵机,且是判定机,N的运行时间是函数 $f: \mathbb{N} \to \mathbb{N}$ ,其中f(n)是在任何长度为n的输入上所有计算分支中的最大步数。

NP类

#### 定理1

设t(x)是一个函数,  $t(n) \ge n$ , 则

- 每个t(n)时间的多带图灵机和某个 $O(t^2(n))$ 时间的单带图灵机等价。
- 每个t(n)时间的非确定型单带图灵机和某个 $2^{O(t(n))}$ 时间的确定型单带图灵机等价。
- 设N是非确定型图灵机,且是判定机,N的运行时间是函数 $f: \mathbb{N} \to \mathbb{N}$ ,其中f(n)是在任何长度为n的输入上所有计算分支中的最大步数。

NP类

#### 定理1

设t(x)是一个函数,  $t(n) \ge n$ , 则

- 每个t(n)时间的多带图灵机和某个 $O(t^2(n))$ 时间的单带图灵机等价。
- 每个t(n)时间的非确定型单带图灵机和某个2<sup>O(t(n))</sup>时间的确定型单带图灵机等价。
- 设N是非确定型图灵机,且是判定机,N的运行时间是函数 $f: \mathbb{N} \to \mathbb{N}$ ,其中f(n)是在任何长度为n的输入上所有计算分支中的最大步数。

#### 定理1

设t(x)是一个函数,  $t(n) \geq n$ , 则

■ 每个t(n)时间的多带图灵机和某个 $O(t^2(n))$ 时间的单带 图灵机等价。

NP类

- 每个t(n)时间的非确定型单带图灵机和某个 $2^{O(t(n))}$ 时间 的确定型单带图灵机等价。
- 设N是非确定型图灵机, 且是判定机, N的运行时间是 函数 $f: \mathbb{N} \to \mathbb{N}$ , 其中f(n)是在任何长度为n的输入上所 有计算分支中的最大步数。



$$P = \bigcup_{k} \text{TIME}(n^k).$$



类。即

### 定义4 (P类)

P是确定型单带图灵机在多项式时间内可判定的语言

$$P = \bigcup$$

$$P = \bigcup_k \mathrm{TIME}(n^k).$$

### 定义4 (P类)

P是确定型单带图灵机在多项式时间内可判定的语言 类。即

$$P = \bigcup_{k} \text{TIME}(n^k).$$

# P问题示例:

- $PATH = \{\langle G, s, t \rangle | G$ 是具有从s到t的有向路径的有向图}
- $\blacksquare RELPRIME = \{\langle x, y \rangle | x, y$ 互素

#### 定义4 (P类)

P是确定型单带图灵机在多项式时间内可判定的语言 类。即

$$P = \bigcup_{k} \text{TIME}(n^k).$$

P问题示例:

- $PATH = \{\langle G, s, t \rangle | G$ 是具有从s到t的有向路径的有向图}
- $RELPRIME = \{\langle x, y \rangle | x, y \subseteq \mathbb{R} \}$

### 定义4 (P类)

P是确定型单带图灵机在多项式时间内可判定的语言 类。即

$$P = \bigcup_{k} \text{TIME}(n^k).$$

P问题示例:

- $PATH = \{\langle G, s, t \rangle | G$ 是具有从s到t的有向路径的有向图}
- $RELPRIME = \{\langle x, y \rangle | x, y$ 互素}

- 对于有m个节点的有向图G, PATH问题的暴力算法所需时间为 $m!=\Theta(2^{m\log m})=\Theta(m^m)$
- 多项式时间算法M:
  - $\blacksquare$  在结点s上做标记;
  - 重复以卜步骤,直全个冉有新结点被标记
    - 扫描G所有的边。如果找到边(a,b),其中a已被标记而b未被标记,则标记b
  - 如果t被标记,则接受,否则拒绝。



- 对于有m个节点的有向图G, PATH问题的暴力算法所需 时间为 $m! = \Theta(2^{m \log m}) = \Theta(m^m)$



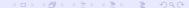
- 对于有m个节点的有向图G, PATH问题的暴力算法所需 时间为 $m! = \Theta(2^{m \log m}) = \Theta(m^m)$



- 对于有m个节点的有向图G, PATH问题的暴力算法所需 时间为 $m! = \Theta(2^{m \log m}) = \Theta(m^m)$



- 对于有m个节点的有向图G, PATH问题的暴力算法所需 时间为 $m! = \Theta(2^{m \log m}) = \Theta(m^m)$
- 多项式时间算法M:

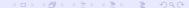


P中的问题举例

- 对于有m个节点的有向图G, PATH问题的暴力算法所需 时间为 $m! = \Theta(2^{m \log m}) = \Theta(m^m)$
- $\blacksquare$  多项式时间算法M:
  - 1 在结点*s*上做标记:



- 对于有m个节点的有向图G, PATH问题的暴力算法所需 时间为 $m! = \Theta(2^{m \log m}) = \Theta(m^m)$
- $\blacksquare$  多项式时间算法M:
  - 1 在结点*s*上做标记:
  - 2 重复以下步骤, 直至不再有新结点被标记



- 对于有m个节点的有向图G, PATH问题的暴力算法所需 时间为 $m! = \Theta(2^{m \log m}) = \Theta(m^m)$
- $\blacksquare$  多项式时间算法M:
  - 1 在结点*s*上做标记:
  - 2 重复以下步骤, 直至不再有新结点被标记
    - 扫描G所有的边。如果找到边(a,b), 其中a已被标记而b未被标 记,则标记b



- 对于有m个节点的有向图G, PATH问题的暴力算法所需 时间为 $m! = \Theta(2^{m \log m}) = \Theta(m^m)$
- $\blacksquare$  多项式时间算法M:
  - 1 在结点s上做标记;
  - 2 重复以下步骤, 直至不再有新结点被标记
    - 扫描G所有的边。如果找到边(a,b),其中a已被标记而b未被标 记,则标记b
  - 3 如果t被标记,则接受,否则拒绝。



#### 证明.

对于输入的 $\langle x, y \rangle$ , 执行

1 重复以下操作,直至y=0

- $x \leftarrow x \mod y$
- $\mathbf{Q}$  父换x, y
- 2 如果x = 1则接受,否则拒绝。



#### 证明.

- I 重复以下操作,直至y=0
  - $\mathbf{x} \leftarrow x \mod y$
  - $\mathbf{Q}$  父换x, y
- 2 如果x = 1则接受,否则拒绝。



#### 证明.

- 重复以下操作,直至y=0
  - $\mathbf{x} \leftarrow x \mod y$
  - 交換x, y
- 2 如果x = 1则接受,否则拒绝。



#### 证明.

- **1** 重复以下操作,直至y=0
  - $x \leftarrow x \mod y$
  - 交换x, u



#### 证明.

- **1** 重复以下操作,直至y=0
  - $x \leftarrow x \mod y$
  - 交換x, y



#### 证明.

- **1** 重复以下操作,直至y=0
  - $x \leftarrow x \mod y$
  - 交換x, y
- $\mathbf{p}$  如果x = 1则接受,否则拒绝。



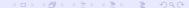
# 定理4 (每个上下文无关语言都是P的成员)

- 确定长度为1的生成方法
- 对于长子串 $w_1 \cdots w_k$ ,将其分为两个短子 串 $w_1 \cdots w_t, w_{t+1} \cdots w_k$ 
  - 若有规则 $A \to BC$ ,而B, C分别可以生  $\overrightarrow{b}_{w_1} \cdots w_t$ ,  $w_{t+1} \cdots w_k$ ,则A生成 $w_1 \cdots w_t$



# 定理4 (每个上下文无关语言都是P的成员)

- 确定长度为1的生成方法
- 对于长子串 $w_1 \cdots w_k$ ,将其分为两个短子  $= w_1 \cdots w_t, w_{t+1} \cdots w_k$ 
  - 若有规则 $A \to BC$ ,而B, C分别可以生



# 定理4(每个上下文无关语言都是P的成员)

- 确定长度为1的生成方法



# 定理4 (每个上下文无关语言都是P的成员)

- 确定长度为1的生成方法
- 对于长子串 $w_1 \cdots w_k$ ,将其分为两个短子 串 $w_1 \cdots w_t$ , $w_{t+1} \cdots w_k$ 
  - 若有规则 $A \rightarrow BC$ ,而B,C分别可以生成 $w_1 \cdots w_t$ ,则A生成 $w_1 \cdots w_k$



# 定理4(每个上下文无关语言都是P的成员)

- 确定长度为1的生成方法
- 对于长子串 $w_1 \cdots w_k$ ,将其分为两个短子
  - 若有规则 $A \rightarrow BC$ ,而B, C分别可以生 成 $w_1 \cdots w_t, w_{t+1} \cdots w_k$ ,则A生成 $w_1 \cdots w_k$



- 2 对i = 1 to n 运行nv次、v为变元数
- 3 对每个变元A,

P中的问题举例

- 4 检查是否有规则 $A \rightarrow w_i$ 。若有,则把A放入Table(i, i)
- 5 对 $\ell = 2$  to n 运行 $n^3$ 步
- 6 对i=1 to  $n-\ell+1$
- 7  $\Rightarrow i = i + \ell - 1$
- 8  $\forall k = i \text{ to } i - 1$
- 9 对每条规则 $A \rightarrow BC$
- 10 若Table(i, k)包含B, Table(k + 1, j)包含C, 则把A放入Tabel(i, j)
- 若S属于Table(1, n),则接受,否则拒绝



# 对于上下文无关语言L,设G是生成L的乔姆斯基范式的CFG, 即其中的规则形式为 $A \to BC$ , $A \to a$ , 则对于输入 $w = w_1 \cdots w_n$

- 2 对i = 1 to n 运行nv次,v为变元数
- 3 对每个变元A,
- 4 检查是否有规则 $A \rightarrow w_i$ 。若有,则把A放入Table(i, i)
- 5 对 $\ell = 2$  to n 运行 $n^3$ 步
- 6 对i=1 to  $n-\ell+1$
- 7  $\Rightarrow i = i + \ell - 1$
- 8  $\forall k = i \text{ to } i - 1$
- 9 对每条规则 $A \rightarrow BC$
- 10 若Table(i, k)包含B, Table(k + 1, j)包含C, 则把A放入Tabel(i, j)
- 若S属于Table(1, n),则接受,否则拒绝



# 对于上下文无关语言L,设G是生成L的乔姆斯基范式的CFG, 即其中的规则形式为 $A \to BC$ , $A \to a$ , 则对于输入 $w = w_1 \cdots w_n$

- 2 对i = 1 to n 运行nv次,v为变元数
- 3 对每个变元A,
- 4 检查是否有规则 $A \rightarrow w_i$ 。若有,则把A放入Table(i, i)
- 5 对 $\ell = 2$  to n 运行 $n^3$ 步
- 6 对i=1 to  $n-\ell+1$
- 7  $\Rightarrow i = i + \ell - 1$
- 8  $\forall k = i \text{ to } i - 1$
- 9 对每条规则 $A \rightarrow BC$
- 10 若Table(i, k)包含B, Table(k + 1, j)包含C, 则把A放入Tabel(i, j)
- 若S属于Table(1, n),则接受,否则拒绝





- $\blacksquare HAMPATH =$  $\{\langle G, s, t \rangle | G$  是包含从s 到t 的哈密尔顿路径的有向图 $\}$

- $\blacksquare HAMPATH =$  $\{\langle G, s, t \rangle | G$  是包含从s 到t 的哈密尔顿路径的有向图 $\}$
- $COMPOSITES = \{x | x = pq,$ 整数 $p,q > 1\}$

- $\blacksquare HAMPATH =$  $\{\langle G, s, t \rangle | G$  是包含从s 到t 的哈密尔顿路径的有向图 $\}$
- $COMPOSITES = \{x | x = pq, \text{ $\mathbb{Z}$ $\mathfrak{Y}$}, q > 1\}$

- $\blacksquare HAMPATH =$  $\{\langle G, s, t \rangle | G$  是包含从s 到t 的哈密尔顿路径的有向图 $\}$
- $COMPOSITES = \{x | x = pq,$ 整数 $p,q > 1\}$
- $\blacksquare SUBSET SUM = \{\langle s, t \rangle | s = \}$  $\{x_1, \dots, x_k\}$ 且存在 $\{y_1, \dots, y_l\} \subseteq s$ ,满足 $\sum y_i = t\}$



- $\blacksquare HAMPATH =$  $\{\langle G, s, t \rangle | G$  是包含从s 到t 的哈密尔顿路径的有向图 $\}$
- $COMPOSITES = \{x | x = pq,$ 整数 $p,q > 1\}$
- $\blacksquare SUBSET SUM = \{\langle s, t \rangle | s = \}$  $\{x_1, \dots, x_k\}$ 且存在 $\{y_1, \dots, y_l\} \subseteq s$ ,满足 $\sum y_i = t\}$

# 多项式可验证性



#### 定义5

■ 语言A的验证机是一个算法V, 其中

其中c被称为A的证书。

- 如果语言A有一个多项式时间验证机,则称其是多项式
- 具有多项式时间可验证的语言类被称为NP(非确定型多



#### 定义5

■ 语言A的验证机是一个算法V, 其中

其中c被称为A的证书。

- ■如果语言A有一个多项式时间验证机,则称其是多项式 可验证的。
- 具有多项式时间可验证的语言类被称为NP(非确定型多



#### 定 义5

■ 语言A的验证机是一个算法V, 其中

其中c被称为A的证书。

- ■如果语言A有一个多项式时间验证机,则称其是多项式 可验证的。
- 具有多项式时间可验证的语言类被称为NP/非确定型多 项式时间)。

# 在非确定型多项式时间内判定HAMPATH问题的非确 定型图灵机:

NP类

对于输入的 $\langle G, s, t \rangle$ , 其中G是m阶有向图

对于输入的 $\langle G, s, t \rangle$ , 其中 $G \in \mathbb{R}_m$ 阶有向图

- **I** 写出m个数构成的序列 $p_1, \dots, p_m$ ,其中 $p_i$ 从 $\{1, \dots, m\}$ 中 非确定地选取:



对于输入的 $\langle G, s, t \rangle$ , 其中 $G \in \mathbb{R}_m$ 阶有向图

- **I** 写出m个数构成的序列 $p_1, \dots, p_m$ ,其中 $p_i$ 从 $\{1, \dots, m\}$ 中 非确定地选取:
- □ 检查序列的重复性。如果有重复,则拒绝:



对于输入的 $\langle G, s, t \rangle$ , 其中G是m阶有向图

- **I** 写出m个数构成的序列 $p_1, \dots, p_m$ ,其中 $p_i$ 从 $\{1, \dots, m\}$ 中 非确定地选取:
- □ 检查序列的重复性。如果有重复,则拒绝:
- 3 检查 $p_1 = s, p_m = t$ 是否都成立, 否则拒绝;



对于输入的 $\langle G, s, t \rangle$ , 其中 $G \in \mathbb{R}_m$ 阶有向图

- **I** 写出m个数构成的序列 $p_1, \dots, p_m$ ,其中 $p_i$ 从 $\{1, \dots, m\}$ 中 非确定地选取:
- □ 检查序列的重复性。如果有重复,则拒绝:
- 3 检查 $p_1 = s, p_m = t$ 是否都成立,否则拒绝;
- **4**  $\forall i \in \{1, \dots, m-1\}$ ,检查 $(p_i, p_{i+1})$ 是否是G中的边。如果 都是,则接受,否则拒绝。

一个语言在NP中的充要条件是其能够被某个非确定型 多项式时间图灵机(NTM)判定。

NP类

- 一个语言在NP中的充要条件是其能够被某个非确定型 多项式时间图灵机(NTM)判定。
  - - 对于长为n的输入w:
      - 非确定地选择长度不超过nk的字符串c
      - 2 在输入 $\langle w,c\rangle$ 上运行V;
      - 3 如果V接受,则接受,否则拒绝。

一个语言在NP中的充要条件是其能够被某个非确定型 多项式时间图灵机(NTM)判定。

NP类

- 多项式时间NTM构造如下: 对于长为n的输入w:



一个语言在NP中的充要条件是其能够被某个非确定型 多项式时间图灵机(NTM)判定。

- 多项式时间NTM构造如下: 对于长为n的输入w:
  - $\blacksquare$  非确定地选择长度不超过 $n^k$ 的字符串c:



一个语言在NP中的充要条件是其能够被某个非确定型 多项式时间图灵机(NTM)判定。

- 多项式时间NTM构造如下: 对于长为n的输入w:
  - $\blacksquare$  非确定地选择长度不超过 $n^k$ 的字符串c:
  - 2 在输入 $\langle w,c\rangle$ 上运行V;



一个语言在NP中的充要条件是其能够被某个非确定型 多项式时间图灵机(NTM)判定。

- 多项式时间NTM构造如下: 对于长为n的输入w:
  - $\blacksquare$  非确定地选择长度不超过 $n^k$ 的字符串c:
  - 2 在输入 $\langle w,c\rangle$ 上运行V;
  - 3 如果V接受,则接受,否则拒绝。



- 对于输入⟨w,c⟩

- 对于输入⟨w,c⟩
  - 在输入w上模拟N, 而把c的每个符号看成是对每次非确定 选择的描述;



- 对于输入⟨w,c⟩
  - 在输入w上模拟N,而把c的每个符号看成是对每次非确定 选择的描述;
  - 若*N*的该计算分支接受,则接收,否则拒绝。



$$NP = \bigcup_{k} \text{NTIME}(n^k)$$

$$P \subseteq NP$$

- NTIME $(t(n)) = \{$ 能被O(t(n))时间NTM判定的语言 $\}$

NP类

- NTIME $(t(n)) = \{$ 能被O(t(n))时间NTM判定的语言 $\}$
- $\blacksquare NP = \bigcup \text{NTIME}(n^k)$

- NTIME $(t(n)) = \{$ 能被O(t(n))时间NTM判定的语言 $\}$
- **N** $P = \bigcup \text{NTIME}(n^k)$
- $P \subseteq NP$

- NTIME $(t(n)) = \{$ 能被O(t(n))时间NTM判定的语言 $\}$
- $\blacksquare NP = \bigcup \text{NTIME}(n^k)$
- $P \subseteq NP$
- $P \stackrel{?}{=} NP$ , 或  $P \stackrel{?}{\subset} NP$

$$w \in A \iff f(w) \in B$$

- $SAT = \{ \langle \phi \rangle : \phi$ 是可满足的布尔公式 \}

$$w \in A \iff f(w) \in B$$

- $SAT = \{ \langle \phi \rangle : \phi$ 是可满足的布尔公式 \}
- $3SAT = \{\langle \phi \rangle : \phi$ 是可满足的3CNF公式}

$$w \in A \iff f(w) \in B$$

- $SAT = \{ \langle \phi \rangle : \phi$ 是可满足的布尔公式 \}
- $3SAT = \{\langle \phi \rangle : \phi$ 是可满足的3CNF公式}
- 对于语言A, B, 若存在多项式时间可计算函 数 $f: \Sigma^* \to \Sigma^*$ , 使得

$$w \in A \iff f(w) \in B$$

则称A可以多项式时间映射可归约到B,记为 $A <_P B$ 。



- $SAT = \{ \langle \phi \rangle : \phi$ 是可满足的布尔公式 \}
- $3SAT = \{\langle \phi \rangle : \phi$ 是可满足的3CNF公式}
- 对于语言A, B, 若存在多项式时间可计算函 数 $f: \Sigma^* \to \Sigma^*$ , 使得

$$w \in A \iff f(w) \in B$$

则称A可以多项式时间映射可归约到B,记为 $A <_P B$ 。

■ 若 $A <_P B$ 且 $B \in P$ ,则 $A \in P$ 

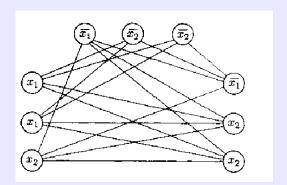
- ■点:对应于每个变量(或其否定)
- 边:连接所有的点对,除非
  - 同一个三元组内的节点无边相连
  - 相反标记的两个节点无边相连

- 点: 对应于每个变量(或其否定)

- 点: 对应于每个变量(或其否定)
- 边: 连接所有的点对, 除非

- 点: 对应于每个变量(或其否定)
- 边: 连接所有的点对, 除非
  - 同一个三元组内的节点无边相连

- 点: 对应于每个变量(或其否定)
- 边:连接所有的点对,除非
  - 同一个三元组内的节点无边相连
  - ■相反标记的两个节点无边相连



## 定义6 (NP完全)

#### 若语言B满足

- $B \in NP$
- NP中每个A都可以多项式时间归约到B

则称B是NP完全的。



## 定义6 (NP完全)

若语言B满足

- $B \in NP$
- NP中每个A都可以多项式时间归约到B

则称B是NP完全的。

- 若B是NP完全的  $\exists B \in P$ , $\exists P \in P$



NP类

## 定义6 (NP完全)

#### 若语言B满足

- $B \in NP$
- NP中每个A都可以多项式时间归约到B

#### 则称B是NP完全的。

- 若B是NP完全的  $\exists B \in P$ , $\exists P \in P$
- 若B是NP完全的,且 $B <_P C$ ,其中 $C \in NP$ , 则C是NP完全的。



#### 定理7 (库克-列文定理: SAT是NP完全的)

- $\blacksquare SAT \in NP;$
- 设A是任意NP问题,N是在时间 $n^k$ 内判定A的非确定型 图灵机。
- 画面:  $n^k \times n^k$ 表,其中每行表示N在输入w上的某个计算公式的投票

				起始格局
				第2个格局
				第*个格局
				第nk个格局



#### $\blacksquare SAT \in NP;$

- 设A是任意NP问题,N是在时间m内判定A的非确定型图灵机。
- 画面:  $n^k \times n^k$ 表,其中每行表示N在输入w上的某个计算公式的数量

				起始格局
				第2个格局
				第*个格局
				第nk个格局

## 定理7 (库克-列文定理: SAT是NP完全的)

- $\blacksquare SAT \in NP;$
- 设A是任意NP问题,N是在时间 $n^k$ 内判定A的非确定型

				起始格局
				第2个格局
				第*个格局
				第nk个格局

## 定理7 (库克-列文定理: SAT是NP完全的)

- $\blacksquare SAT \in NP;$
- 设A是任意NP问题,N是在时间 $n^k$ 内判定A的非确定型 图灵机。

				起始格局
				第2个格局
				第*个格局
				第nk个格局

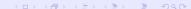
## 定理7 (库克-列文定理: SAT是NP完全的)

- $\blacksquare SAT \in NP;$
- 设A是任意NP问题,N是在时间 $n^k$ 内判定A的非确定型 图灵机。
- 画面:  $n^k \times n^k$ 表, 其中每行表示N在输入w上的某个计 算分支的格局

#	$q_0$	$w_1$		$w_n$	]		J	#	起始格局
#								#	第2个格局
÷	÷	÷	i	÷	:	:	:	÷	第*个格局
#								#	第nk个格局



- N在w上的每个对应画面对应N在w上的一个计算分支



- *N*是否接受*w*等价于*N*在*w*上是否有接受画面(即某行是 接受格局)



- *N*是否接受*w*等价于*N*在*w*上是否有接受画面(即某行是 接受格局)
- 归约函数  $f: \diamond C = Q \cup \Gamma \cup \{\#\}$ ,其中 $Q, \Gamma$ 分别表示状态 集和带字母表



- *N*是否接受*w*等价于*N*在*w*上是否有接受画面(即某行是 接受格局)
- 归约函数f: 令 $C = Q \cup \Gamma \cup \{\#\}$ ,其中Q, $\Gamma$ 分别表示状态 集和带字母表
  - 变量:



- *N*是否接受*w*等价于*N*在*w*上是否有接受画面(即某行是 接受格局)
- 归约函数f: 令 $C = Q \cup \Gamma \cup \{\#\}$ ,其中Q, $\Gamma$ 分别表示状态 集和带字母表
  - 变量:
    - 对每个 $i, j, 1 < i, j < n^k$ 以及每个 $s \in C$ ,定义变量 $x_{i,i,s}$



- *N*是否接受*w*等价于*N*在*w*上是否有接受画面(即某行是 接受格局)
- 归约函数f: 令 $C = Q \cup \Gamma \cup \{\#\}$ ,其中Q, $\Gamma$ 分别表示状态 集和带字母表
  - 变量:
    - 对每个 $i, j, 1 < i, j < n^k$ 以及每个 $s \in C$ ,定义变量 $x_{i,i,s}$
    - $x_{i,i,s} = 1 \iff$  单元cell[i,j]包含s



- *N*是否接受*w*等价于*N*在*w*上是否有接受画面(即某行是 接受格局)
- 归约函数  $f: \varphi C = Q \cup \Gamma \cup \{\#\}$ ,其中 $Q, \Gamma$ 分别表示状态 集和带字母表
  - 变量:
    - 对每个 $i, j, 1 < i, j < n^k$ 以及每个 $s \in C$ ,定义变量 $x_{i,i,s}$
    - $x_{i,i,s} = 1 \iff$  单元cell[i,j]包含s
  - $\triangle \vec{\Xi} \phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accent}$



■ 
$$\phi_{cell}$$
:  $\bigwedge_{1 \leq i,j \leq n^k} \left[ \left( \bigvee_{s \in C} x_{i,j,s} \right) \wedge \left( \bigwedge_{s \neq t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \right) \right]$ 
■ 保证对于每个单元 $cell[i,j]$ ,在所有 $s_{i,j,s}$ ,  $s \in C$ 中有且仅有

$$\bullet \phi_{cell} \colon \bigwedge_{1 \leq i,j \leq n^k} \left[ \left( \bigvee_{s \in C} x_{i,j,s} \right) \wedge \left( \bigwedge_{s \neq t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \right) \right]$$

- 保证对于每个单元cell[i,j],在所有 $s_{i,j,s}$ ,  $s \in C$ 中有且仅有一个是1
- - 保证第一行是*N在w*上的起始格局
- $lackbox{ } \phi_{accept} : \bigvee_{1 \leq i,j \leq n^k} x_{i,j,q_{accept}}$ 
  - 保证接受格局出现在画面中



$$\bullet \phi_{cell} \colon \bigwedge_{1 \leq i,j \leq n^k} \left[ \left( \bigvee_{s \in C} x_{i,j,s} \right) \wedge \left( \bigwedge_{s \neq t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \right) \right]$$

- 保证对于每个单元cell[i,j], 在所有 $s_{i,j,s}$ ,  $s \in C$ 中有且仅有 一个是1

- $\bullet \phi_{cell} \colon \bigwedge_{1 \le i,j \le n^k} \left| \left( \bigvee_{s \in C} x_{i,j,s} \right) \wedge \left( \bigwedge_{s \ne t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \right) \right|$ 
  - 保证对于每个单元cell[i,j], 在所有 $s_{i,j,s}$ ,  $s \in C$ 中有且仅有 一个是1
- $\phi_{start}: x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\bot} \wedge \cdots$

- $\bullet \phi_{cell} \colon \bigwedge_{1 \leq i,j \leq n^k} \left[ \left( \bigvee_{s \in C} x_{i,j,s} \right) \wedge \left( \bigwedge_{s \neq t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \right) \right]$ 
  - 保证对于每个单元cell[i,j],在所有 $s_{i,j,s}$ ,  $s \in C$ 中有且仅有一个是1
- $\phi_{start}$ :  $x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\_} \wedge \cdots \wedge x_{1,n^k-1,\_} \wedge x_{1,n^k,\#}$ 
  - 保证第一行是N在w上的起始格局
- $\blacksquare \varphi_{accept}: \bigvee_{1 \leq i, i \leq n^k} x_{i,j,q_{accept}}$ 
  - 保证接受格局出现在画面中



- $\bullet \phi_{cell} \colon \bigwedge_{1 \leq i,j \leq n^k} \left[ \left( \bigvee_{s \in C} x_{i,j,s} \right) \wedge \left( \bigwedge_{s \neq t} \left( \overline{x_{i,j,s}} \vee \overline{x_{i,j,t}} \right) \right) \right]$ 
  - 保证对于每个单元cell[i,j],在所有 $s_{i,j,s}$ ,  $s \in C$ 中有且仅有一个是1
- $\phi_{start}$ :  $x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\bot} \wedge \cdots \wedge x_{1,n^k-1,\bot} \wedge x_{1,n^k,\#}$ 
  - 保证第一行是N在w上的起始格局
- $\blacksquare \phi_{accept}: \bigvee_{i \in A} x_{i,j,q_{accept}}$ 
  - 保证接受格局出现在画面中



- $\bullet \phi_{cell} \colon \bigwedge_{1 \le i,j \le n^k} \left| \left( \bigvee_{s \in C} x_{i,j,s} \right) \wedge \left( \bigwedge_{s \ne t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \right) \right|$ 
  - 保证对于每个单元cell[i,j], 在所有 $s_{i,j,s}$ ,  $s \in C$ 中有且仅有 一个是1
- $\phi_{start}: x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\bot} \wedge \cdots \wedge x_{1,n+3,\bot} \wedge x_{1,n+3,\bot} \wedge \cdots \wedge x_{1,n+3,\bot} \wedge x_{1,n+3,\bot} \wedge \cdots \wedge x_{1,n+3,\bot} \wedge \cdots$  $\cdots \wedge x_{1,n^k-1} \wedge x_{1,n^k,\#}$ 
  - 保证第一行是*N*在*w*上的起始格局
- $\phi_{accept}$ :  $\bigvee_{1 \leq i,j \leq n^k} x_{i,j,q_{accept}}$



- $\bullet \phi_{cell} \colon \bigwedge_{1 \le i,j \le n^k} \left| \left( \bigvee_{s \in C} x_{i,j,s} \right) \wedge \left( \bigwedge_{s \ne t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \right) \right|$ 
  - 保证对于每个单元cell[i,j], 在所有 $s_{i,j,s}$ ,  $s \in C$ 中有且仅有 一个是1

- $\phi_{start}: x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\bot} \wedge \cdots \wedge x_{1,n+3,\bot} \wedge x_{1,n+3,\bot} \wedge \cdots \wedge x_{1,n+3,\bot} \wedge x_{1,n+3,\bot} \wedge \cdots \wedge x_{1,n+3,\bot} \wedge \cdots$  $\cdots \wedge x_{1,n^k-1} \wedge x_{1,n^k,\#}$ 
  - 保证第一行是*N*在*w*上的起始格局
- $\phi_{accept}: \bigvee_{1 \leq i, j \leq n^k} x_{i,j,q_{accept}}$



- $\bullet \phi_{cell} \colon \bigwedge_{1 \le i,j \le n^k} \left| \left( \bigvee_{s \in C} x_{i,j,s} \right) \wedge \left( \bigwedge_{s \ne t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \right) \right|$ 
  - 保证对于每个单元cell[i,j], 在所有 $s_{i,j,s}$ ,  $s \in C$ 中有且仅有 一个是1

- $\phi_{start}: x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\bot} \wedge \cdots \wedge x_{1,n+3,\bot} \wedge x_{1,n+3,\bot} \wedge \cdots \wedge x_{1,n+3,\bot} \wedge x_{1,n+3,\bot} \wedge \cdots \wedge x_{1,n+3,\bot} \wedge \cdots$  $\cdots \wedge x_{1,n^k-1} \wedge x_{1,n^k,\#}$ 
  - 保证第一行是*N*在*w*上的起始格局
- $\phi_{accept}: \bigvee_{1 \leq i, j \leq n^k} x_{i,j,q_{accept}}$ 
  - 保证接受格局出现在画面中



- *φ*<sub>move</sub>: 保证每一行都是由上一行按照规则合法转移得 到的格局
- 手段: 合法2×3窗口
- 例: 设字母表为a,b,c,  $\delta(q_1,a) = \{(q_1,b,R)\}, \delta(q_1,b) = \{(q_2,c,L),(q_2,a,R)\}$

- $\bullet$   $\phi_{move}$ : 保证每一行都是由上一行按照规则合法转移得 到的格局

$$a q_1 b$$

- ullet  $\phi_{move}$ : 保证每一行都是由上一行按照规则合法转移得到的格局
- 手段: 合法2×3窗口
- 例: 设字母表为a,b,c,  $\delta(q_1,a) = \{(q_1,b,R)\}, \delta(q_1,b) = \{(q_2,c,L),(q_2,a,R)\}$

$$\begin{array}{c|cccc} a & q_1 & b \\ \hline q_2 & a & c \end{array}$$

$$\begin{bmatrix} a & q_1 & b \\ a & a & q_2 \end{bmatrix}$$



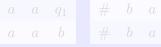
$$\begin{bmatrix} a & b & a \\ a & a & a \end{bmatrix}$$

$$a$$
  $q_1$   $b$ 

- $\bullet$   $\phi_{move}$ : 保证每一行都是由上一行按照规则合法转移得 到的格局
- 手段: 合法2×3窗口
- 例: 设字母表为a,b,c,  $\delta(q_1,a) = \{(q_1,b,R)\}, \delta(q_1,b) =$  $\{(q_2, c, L), (q_2, a, R)\}$

$$\begin{array}{c|cccc} a & q_1 & b \\ \hline q_2 & a & c \end{array}$$

$$\begin{bmatrix} a & q_1 & b \\ a & a & q_2 \end{bmatrix}$$



$$\begin{bmatrix} a & b & a \\ a & a & a \end{bmatrix}$$

$$a q_1 b$$

- 手段: 合法2×3窗口
- 例: 设字母表为a,b,c,  $\delta(q_1,a) = \{(q_1,b,R)\}, \delta(q_1,b) =$  $\{(q_2, c, L), (q_2, a, R)\}$
- b ba $q_1$ a $q_1$ # baa $q_2$  $q_2$ a

- $\bullet$   $\phi_{move}$ : 保证每一行都是由上一行按照规则合法转移得 到的格局
- 手段: 合法2×3窗口
- 例: 设字母表为a,b,c,  $\delta(q_1,a) = \{(q_1,b,R)\}, \delta(q_1,b) =$  $\{(q_2, c, L), (q_2, a, R)\}$
- ba $q_1$  $q_1$ # ba c aa $q_2$  $q_2$ a
- b b ba bbaa $q_1$ c b b $q_2$ aa $q_1$ aa

■ 如果顶行是起始格局,表中每个窗口都是合法的,那么每一行都是由上一行合法转移得到。

- - 窗口合法 =  $X_{i,j-1,a_1} \wedge X_{i,j,a_2} \wedge \cdots \wedge X_{i+1,j+2,a_6}$
- 变量数目:  $O(n^{2k})$ 
  - $\blacksquare$  单元数 $n^{2k}$
  - lacksquare 甲兀天联的发量数为|C|,与输入子符串长度尢关

- 如果顶行是起始格局, 表中每个窗口都是合法的, 那么 每一行都是由上一行合法转移得到。
- $\phi_{move} = \bigwedge_{1 \le i < n^k, 1 \le j \le n^k} ((i, j)$ 窗口都是合法的)

- 如果顶行是起始格局, 表中每个窗口都是合法的, 那么 每一行都是由上一行合法转移得到。
- $\phi_{move} = \bigwedge_{1 \le i < n^k, 1 \le j \le n^k} ((i, j)$ 窗口都是合法的)

窗口合法 = 
$$X_{i,j-1,a_1} \wedge X_{i,j,a_2} \wedge \cdots \wedge X_{i+1,j+2,a_6}$$

■ 如果顶行是起始格局, 表中每个窗口都是合法的, 那么 每一行都是由上一行合法转移得到。

NP类

•  $\phi_{move} = \bigwedge_{1 \le i < n^k, 1 \le j \le n^k} ((i, j)$ 窗口都是合法的)

窗口合法 = 
$$X_{i,j-1,a_1} \wedge X_{i,j,a_2} \wedge \cdots \wedge X_{i+1,j+2,a_6}$$

- 变量数目: O(n<sup>2k</sup>)

- 如果顶行是起始格局, 表中每个窗口都是合法的, 那么 每一行都是由上一行合法转移得到。
- $\phi_{move} = \bigwedge_{1 \le i < n^k, 1 \le j \le n^k} ((i, j)$ 窗口都是合法的)

窗口合法 = 
$$X_{i,j-1,a_1} \wedge X_{i,j,a_2} \wedge \cdots \wedge X_{i+1,j+2,a_6}$$

- 变量数目: O(n<sup>2k</sup>)
  - 单元数n<sup>2k</sup>



■ 如果顶行是起始格局, 表中每个窗口都是合法的, 那么 每一行都是由上一行合法转移得到。

NP类

•  $\phi_{move} = \bigwedge_{1 \le i < n^k, 1 \le j \le n^k} ((i, j)$ 窗口都是合法的)

窗口合法 = 
$$X_{i,j-1,a_1} \wedge X_{i,j,a_2} \wedge \cdots \wedge X_{i+1,j+2,a_6}$$

- 变量数目: O(n<sup>2k</sup>)
  - 单元数n<sup>2k</sup>
  - 单元关联的变量数为|C|,与输入字符串长度无关



- 如果顶行是起始格局, 表中每个窗口都是合法的, 那么 每一行都是由上一行合法转移得到。
- $\phi_{move} = \bigwedge_{1 \le i < n^k, 1 \le j \le n^k} ((i, j)$ 窗口都是合法的)

窗口合法 = 
$$X_{i,j-1,a_1} \wedge X_{i,j,a_2} \wedge \cdots \wedge X_{i+1,j+2,a_6}$$

- 变量数目: O(n<sup>2k</sup>)
  - 单元数n<sup>2k</sup>
  - 单元关联的变量数为|C|,与输入字符串长度无关



- $lackbox{} \phi_{cell} : O(n^{2k})$
- $\bullet$   $\phi_{start}:O(n^k)$
- lacksquare  $\phi_{move}$ : $O(n^{2k})$
- $lackbox{\bullet} \phi_{accept} : O(n^{2k})$

#### 推论1

- 3SAT是NP完全的;
- CLIQUE是NP完全的;

- $\bullet$   $\phi_{cell}:O(n^{2k})$
- $lackbox{\bullet} \phi_{start} : O(n^k)$
- lacksquare  $\phi_{move}$ : $O(n^{2k})$
- $lackbox{\bullet} \phi_{accept} : O(n^{2k})$

### 推论1

- 3SAT是NP完全的;
- CLIQUE是NP完全的;

- $\phi_{start}:O(n^k)$
- $\bullet$   $\phi_{move}$ : $O(n^{2k})$
- lacksquare  $\phi_{accept}$ : $O(n^{2k})$

- 3SAT是NP完全的;
- CLIQUE是NP完全的;

- $\phi_{start}:O(n^k)$
- $\phi_{move}:O(n^{2k})$
- lacksquare  $\phi_{accept}$ : $O(n^{2k})$

- 3SAT是NP完全的;
- CLIQUE是NP完全的;

- $\bullet \phi_{cell} : O(n^{2k})$
- $\phi_{start}:O(n^k)$
- $\bullet$   $\phi_{move}$ : $O(n^{2k})$
- $\bullet$   $\phi_{accept}$ : $O(n^{2k})$

- 3SAT是NP完全的;
- CLIQUE是NP完全的;

- $\bullet$   $\phi_{cell}:O(n^{2k})$
- $\phi_{start}:O(n^k)$
- $\bullet$   $\phi_{move}:O(n^{2k})$
- $\phi_{accept}:O(n^{2k})$

- 3SAT是NP完全的;
- CLIQUE是NP完全的;

- $\bullet$   $\phi_{cell}:O(n^{2k})$
- $\phi_{start}:O(n^k)$
- $\phi_{move}:O(n^{2k})$

- 3SAT是NP完全的;
- CLIQUE是NP完全的;

- SUBSET-SUM∈NP:
- $3SAT \le_P SUBSET SUM$ :



- SUBSET-SUM $\in$ NP;
- $3SAT \le_P SUBSET SUM$ :



- SUBSET-SUM∈NP;
- $3SAT <_P SUBSET SUM$ :
  - **②** 设 $\phi = c_1 \land c_2 \land \cdots \land c_k, \phi$ 中的变量为 $x_1, \cdots, x_l, 其中<math>c_i$ 为
  - 变量 $x_i$ 对应一对数 $y_i, z_i$ :



#### 证明.

- SUBSET-SUM∈NP;
- $3SAT \le_P SUBSET SUM$ :
  - 设 $\phi = c_1 \wedge c_2 \wedge \cdots \wedge c_k$ ,  $\phi$ 中的变量为 $x_1, \cdots, x_l$ , 其中 $c_i$ 为 初等和
  - 变量 $x_i$ 对应一对数 $y_i, z_i$ :

200

- SUBSET-SUM∈NP:
- $3SAT <_P SUBSET SUM$ :
  - 设 $\phi = c_1 \wedge c_2 \wedge \cdots \wedge c_k$ ,  $\phi$ 中的变量为 $x_1, \cdots, x_l$ , 其中 $c_i$ 为 初等和
  - 变量 $x_i$ 对应一对数 $y_i, z_i$ :
    - $y_i, z_i$  十进制的左边部分由1和随后的l i 个0组成
    - 右边部分: 当初等和 $c_i$ 包含 $x_i$ 时,  $y_i$ 的第j位是1;



- SUBSET-SUM∈NP:
- $3SAT <_P SUBSET SUM$ :
  - **•** 设 $\phi = c_1 \wedge c_2 \wedge \cdots \wedge c_k$ ,  $\phi$ 中的变量为 $x_1, \cdots, x_l$ , 其中 $c_i$ 为 初等和
  - 变量 $x_i$ 对应一对数 $y_i, z_i$ :
    - $y_i, z_i$ 十进制的左边部分由1和随后的l i个0组成
    - 右边部分: 当初等和 $c_i$ 包含 $x_i$ 时,  $y_i$ 的第i位是1;





- SUBSET-SUM∈NP:
- $3SAT <_P SUBSET SUM$ :
  - 设 $\phi = c_1 \wedge c_2 \wedge \cdots \wedge c_k$ ,  $\phi$ 中的变量为 $x_1, \cdots, x_l$ , 其中 $c_i$ 为 初等和
  - 变量 $x_i$ 对应一对数 $y_i, z_i$ :
    - $y_i, z_i$ 十进制的左边部分由1和随后的l i个0组成
    - 右边部分: 当初等和 $c_i$ 包含 $x_i$ 时,  $y_i$ 的第i位是1; 当初等和 $c_i$ 包含 $\overline{x_i}$ 时, $z_i$ 的第i位是1
    - 其余位记为0



- SUBSET-SUM∈NP:
- $3SAT <_P SUBSET SUM$ :
  - 设 $\phi = c_1 \wedge c_2 \wedge \cdots \wedge c_k$ ,  $\phi$ 中的变量为 $x_1, \cdots, x_l$ , 其中 $c_i$ 为 初等和
  - 变量 $x_i$ 对应一对数 $y_i, z_i$ :
    - $y_i, z_i$ 十进制的左边部分由1和随后的l i个0组成
    - 右边部分: 当初等和 $c_i$ 包含 $x_i$ 时,  $y_i$ 的第i位是1; 当初等和 $c_i$ 包含 $\overline{x_i}$ 时, $z_i$ 的第i位是1
    - 其余位记为0



$$y_1 = 100010$$

$$z_1 = 100000$$

$$y_2 = 10001$$

$$z_2 = 10000$$

$$y_3 = 1000$$

$$z_3 = 1011$$

$$y_4 = 100$$

$$z_4 = 110$$

- $y_1 = 100010$
- $z_1 = 100000$

- $y_1 = 100010$
- $z_1 = 100000$
- $y_2 = 10001$
- $z_2 = 10000$

- $y_1 = 100010$
- $z_1 = 100000$
- $y_2 = 10001$
- $z_2 = 10000$
- $y_3 = 1000$
- $z_3 = 1011$

- $y_1 = 100010$
- $z_1 = 100000$
- $y_2 = 10001$
- $z_2 = 10000$
- $y_3 = 1000$
- $z_3 = 1011$
- $y_4 = 100$
- $z_4 = 110$

- $y_1 = 100010$
- $z_1 = 100000$
- $y_2 = 10001$
- $z_2 = 10000$
- $y_3 = 1000$
- $z_3 = 1011$
- $y_4 = 100$
- $z_4 = 110$

- $y_1 = 100010$
- $z_1 = 100000$
- $y_2 = 10001$
- $z_2 = 10000$
- $y_3 = 1000$
- $z_3 = 1011$
- $y_4 = 100$
- $z_4 = 110$

- $y_1 = 100010$
- $z_1 = 100000$
- $y_2 = 10001$
- $z_2 = 10000$
- $y_3 = 1000$
- $z_3 = 1011$
- $y_4 = 100$
- $z_4 = 110$

- $y_1 = 100010$
- $z_1 = 100000$
- $y_2 = 10001$
- $z_2 = 10000$
- $y_3 = 1000$
- $z_3 = 1011$
- $y_4 = 100$
- $z_4 = 110$

- $y_1 = 100010$
- $z_1 = 100000$
- $y_2 = 10001$
- $z_2 = 10000$
- $y_3 = 1000$
- $z_3 = 1011$
- $y_4 = 100$
- $z_4 = 110$

- $y_1 = 100010$
- $z_1 = 100000$
- $y_2 = 10001$
- $z_2 = 10000$
- $y_3 = 1000$
- $z_3 = 1011$
- $y_4 = 100$
- $z_4 = 110$

- $y_1 = 100010$
- $z_1 = 100000$
- $y_2 = 10001$
- $z_2 = 10000$
- $y_3 = 1000$
- $z_3 = 1011$
- $y_4 = 100$
- $z_4 = 110$

- $y_1 = 100010$
- $z_1 = 100000$
- $y_2 = 10001$
- $z_2 = 10000$
- $y_3 = 1000$
- $z_3 = 1011$
- $y_4 = 100$
- $z_4 = 110$



$$g_i = h_i = 10^{k-j}$$
:  $g_1 = h_1 = 10, g_2 = h_2 = 1$ 

$$S = \{y_1, z_1, \cdots, y_l, z_l, g_1, h_1, \cdots, g_k, h_k\}$$

$$t = 1^l 3^k$$
:  $t = 100033$ 



$$g_i = h_i = 10^{k-j}$$
:  $g_1 = h_1 = 10, g_2 = h_2 = 1$ 

$$S = \{y_1, z_1, \cdots, y_l, z_l, g_1, h_1, \cdots, g_k, h_k\}$$



$$g_i = h_i = 10^{k-j}$$
:  $g_1 = h_1 = 10, g_2 = h_2 = 1$ 

NP类

$$S = \{y_1, z_1, \cdots, y_l, z_l, g_1, h_1, \cdots, g_k, h_k\}$$

$$t = 1^l 3^k$$
:  $t = 100033$ 



- 对每个初等和 $c_i$ ,取一对数 $g_i, h_i$ :
  - $q_i = h_i = 10^{k-j}$ :  $q_1 = h_1 = 10, q_2 = h_2 = 1$
- $S = \{y_1, z_1, \cdots, y_l, z_l, g_1, h_1, \cdots, g_k, h_k\}$



$$g_i = h_i = 10^{k-j}$$
:  $g_1 = h_1 = 10, g_2 = h_2 = 1$ 

$$S = \{y_1, z_1, \cdots, y_l, z_l, g_1, h_1, \cdots, g_k, h_k\}$$

$$t = 1^l 3^k$$
:  $t = 100033$ 



- 对每个初等和 $c_i$ ,取一对数 $g_i, h_i$ :
  - $q_i = h_i = 10^{k-j}$ :  $q_1 = h_1 = 10, q_2 = h_2 = 1$
- $S = \{y_1, z_1, \cdots, y_l, z_l, g_1, h_1, \cdots, g_k, h_k\}$
- $t = 1^{l}3^{k}$ : t = 100033



- 对每个初等和 $c_i$ ,取一对数 $g_i$ ,  $h_i$ :
  - $q_i = h_i = 10^{k-j}$ :  $q_1 = h_1 = 10, q_2 = h_2 = 1$
- $S = \{y_1, z_1, \dots, y_l, z_l, g_1, h_1, \dots, g_k, h_k\}$
- $t = 1^{l}3^{k}$ : t = 100033

- 在φ的成真赋值中,

$$g_i = h_i = 10^{k-j}$$
:  $g_1 = h_1 = 10, g_2 = h_2 = 1$ 

- $S = \{y_1, z_1, \dots, y_l, z_l, g_1, h_1, \dots, g_k, h_k\}$
- $t = 1^{l}3^{k}$ : t = 100033

- 在φ的成真赋值中,
  - = 若 $x_i = 1$ ,则取 $y_i$

- 对每个初等和 $c_i$ ,取一对数 $g_i$ ,  $h_i$ :
  - $q_i = h_i = 10^{k-j}$ :  $q_1 = h_1 = 10, q_2 = h_2 = 1$
- $S = \{y_1, z_1, \dots, y_l, z_l, g_1, h_1, \dots, g_k, h_k\}$
- $t = 1^{l}3^{k}$ : t = 100033

- 在φ的成真赋值中,
  - = 若 $x_i = 1$ ,则取 $y_i$
  - $\blacksquare$  若 $x_i = 0$ ,则取 $z_i$

- 对每个初等和 $c_i$ ,取一对数 $g_i$ ,  $h_i$ :
  - $a_i = h_i = 10^{k-j}$ :  $q_1 = h_1 = 10, q_2 = h_2 = 1$
- $S = \{y_1, z_1, \dots, y_l, z_l, g_1, h_1, \dots, g_k, h_k\}$
- $t = 1^{l}3^{k}$ : t = 100033

- 在φ的成真赋值中,
  - = 若 $x_i = 1$ ,则取 $y_i$
  - $\blacksquare$  若 $x_i = 0$ ,则取 $z_i$
- 所选数之和的前i位的每一位是1

- 对每个初等和 $c_i$ ,取一对数 $g_i$ ,  $h_i$ :
  - $a_i = h_i = 10^{k-j}$ :  $q_1 = h_1 = 10, q_2 = h_2 = 1$
- $S = \{y_1, z_1, \dots, y_l, z_l, g_1, h_1, \dots, g_k, h_k\}$
- $t = 1^{l}3^{k}$ : t = 100033

 $\phi$ 可满足 $\Longrightarrow$  S的某个子集和为t

- 在φ的成真赋值中,
  - = 若 $x_i = 1$ ,则取 $y_i$
  - $\blacksquare$  若 $x_i = 0$ ,则取 $z_i$
- 所选数之和的前i位的每一位是1
- 所选数之和的后k位为1,2,3:



- 对每个初等和 $c_i$ ,取一对数 $g_i$ ,  $h_i$ :
  - $a_i = h_i = 10^{k-j}$ :  $q_1 = h_1 = 10, q_2 = h_2 = 1$
- $S = \{y_1, z_1, \dots, y_l, z_l, g_1, h_1, \dots, g_k, h_k\}$
- $t = 1^{l}3^{k}$ : t = 100033

 $\phi$ 可满足 $\Longrightarrow$  S的某个子集和为t

- 在φ的成真赋值中,
  - = 若 $x_i = 1$ ,则取 $y_i$
  - $\blacksquare$  若 $x_i = 0$ ,则取 $z_i$
- 所选数之和的前l位的每一位是1
- 所选数之和的后k位为1,2,3:
- 取适当的q,h使得后k位的每一位都是3



对于
$$\phi = (x_1 \vee \overline{x_3} \vee \overline{x_4}) \wedge (x_2 \vee x_2 \vee \overline{x_3})$$
的成真解

释
$$x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 1$$
有:

■ 
$$\Re z_1 = 100000, y_2 = 10001, z_3 = 1011, y_4 = 100$$

$$z_1 + y_2 + z_3 + y_4 = 1111112$$

$$z_1 + y_2 + z_3 + y_4 + g_1 + h_1 + g_2 = 111133 = t$$

对于
$$\phi = (x_1 \vee \overline{x_3} \vee \overline{x_4}) \wedge (x_2 \vee x_2 \vee \overline{x_3})$$
的成真解

释
$$x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 1$$
有:

- $\mathfrak{P}_{z_1} = 100000, y_2 = 10001, z_3 = 1011, y_4 = 100$

对于 $\phi = (x_1 \vee \overline{x_3} \vee \overline{x_4}) \wedge (x_2 \vee x_2 \vee \overline{x_3})$ 的成真解

释 $x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 1$ 有:

- $\mathfrak{P}_{z_1} = 100000, y_2 = 10001, z_3 = 1011, y_4 = 100$
- $z_1 + y_2 + z_3 + y_4 = 1111112$

对于 $\phi = (x_1 \vee \overline{x_3} \vee \overline{x_4}) \wedge (x_2 \vee x_2 \vee \overline{x_3})$ 的成真解

释 $x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 1$ 有:

- $\mathfrak{P}_{z_1} = 100000, y_2 = 10001, z_3 = 1011, y_4 = 100$
- $z_1 + y_2 + z_3 + y_4 = 1111112$
- $\mathbf{\mathfrak{P}}_{g_1,h_1,g_2}$

对于 $\phi = (x_1 \vee \overline{x_3} \vee \overline{x_4}) \wedge (x_2 \vee x_2 \vee \overline{x_3})$ 的成真解

释 $x_1 = 0$ ,  $x_2 = 1$ ,  $x_3 = 0$ ,  $x_4 = 1$ 有:

- $\mathfrak{P}_{z_1} = 100000, y_2 = 10001, z_3 = 1011, y_4 = 100$
- $z_1 + y_2 + z_3 + y_4 = 1111112$
- $\mathbf{\mathfrak{P}}_{g_1,h_1,g_2}$
- $z_1 + y_2 + z_3 + y_4 + q_1 + h_1 + q_2 = 111133 = t$

- 如果 $y_i \in T$ ,则令 $x_i = 1$ ,否则令 $x_i = 0$
- 由于后k列的每一列和为3,从而子集中的 $y_i$ 或 $z_i$ 必定至少提供1
- 如果是 $y_i$ ,则 $x_i$ 出现在 $c_j$ 中且赋值是1,从而 $c_j$ 被满足
- 如果是 $z_i$ ,则 $\overline{x_i}$ 出现在 $c_j$ 中且 $x_i$ 赋值为0,从而 $c_j$ 满足
- 计算量 $O(n^2)$

- 如果 $y_i \in T$ ,则令 $x_i = 1$ ,否则令 $x_i = 0$

- 如果 $y_i \in T$ ,则令 $x_i = 1$ ,否则令 $x_i = 0$
- 由于后k列的每一列和为3,从而子集中的 $y_i$ 或 $z_i$ 必定至 少提供1

- 如果 $y_i \in T$ ,则令 $x_i = 1$ ,否则令 $x_i = 0$
- 由于后k列的每一列和为3,从而子集中的 $y_i$ 或 $z_i$ 必定至 少提供1
- 如果是 $y_i$ ,则 $x_i$ 出现在 $c_i$ 中且赋值是1,从而 $c_i$ 被满足

- 如果 $y_i \in T$ ,则令 $x_i = 1$ ,否则令 $x_i = 0$
- 由于后k列的每一列和为3,从而子集中的y;或z;必定至 少提供1

NP类

- 如果是 $y_i$ ,则 $x_i$ 出现在 $c_i$ 中且赋值是1,从而 $c_i$ 被满足
- 如果是 $z_i$ ,则 $\overline{x_i}$ 出现在 $c_i$ 中且 $x_i$ 赋值为0,从而 $c_i$ 满足



- 如果 $y_i \in T$ ,则令 $x_i = 1$ ,否则令 $x_i = 0$
- 由于后k列的每一列和为3,从而子集中的y;或z;必定至 少提供1
- 如果是 $y_i$ ,则 $x_i$ 出现在 $c_i$ 中且赋值是1,从而 $c_i$ 被满足
- 如果是 $z_i$ ,则 $\overline{x_i}$ 出现在 $c_i$ 中且 $x_i$ 赋值为0,从而 $c_i$ 满足
- 计算量O(n²)