计算理论
第三章 计算复杂性

教材:

[S] 唐常杰等译, Sipser著, 计算理论导引, 机械工业.

参考资料:

[L] Lewis等著, 计算理论基础, 清华大学.

第三章 计算复杂性

- 1. 时间复杂性

  $\{ 0^k1^k \mid k \geq 0 \}$的时间复杂性分析

- 2. 不同模型的复杂性关系

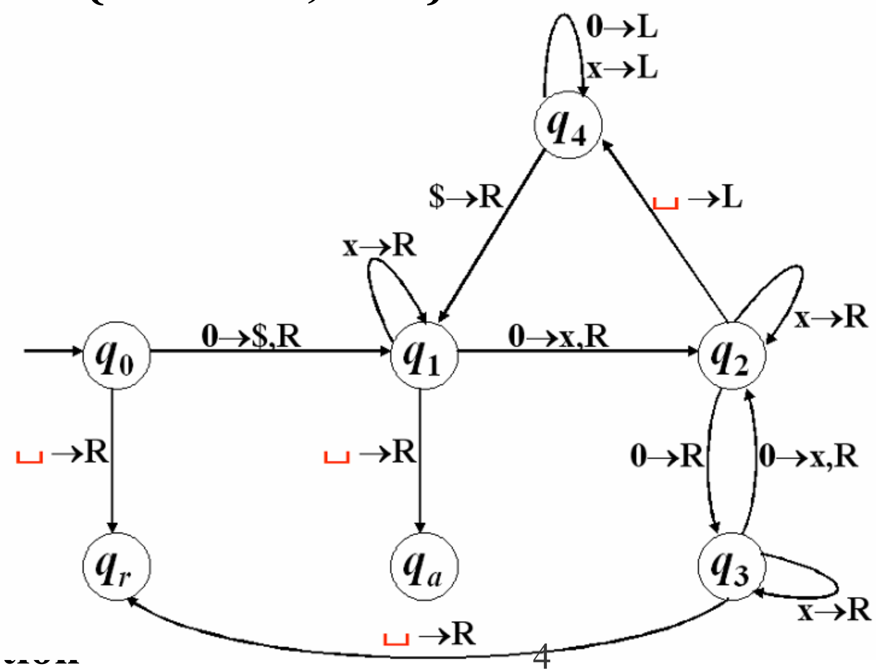
  单带与多带 确定与非确定

- 3. P类与NP类
- 4. NP完全性及NP完全问题

——第7章 时间复杂性

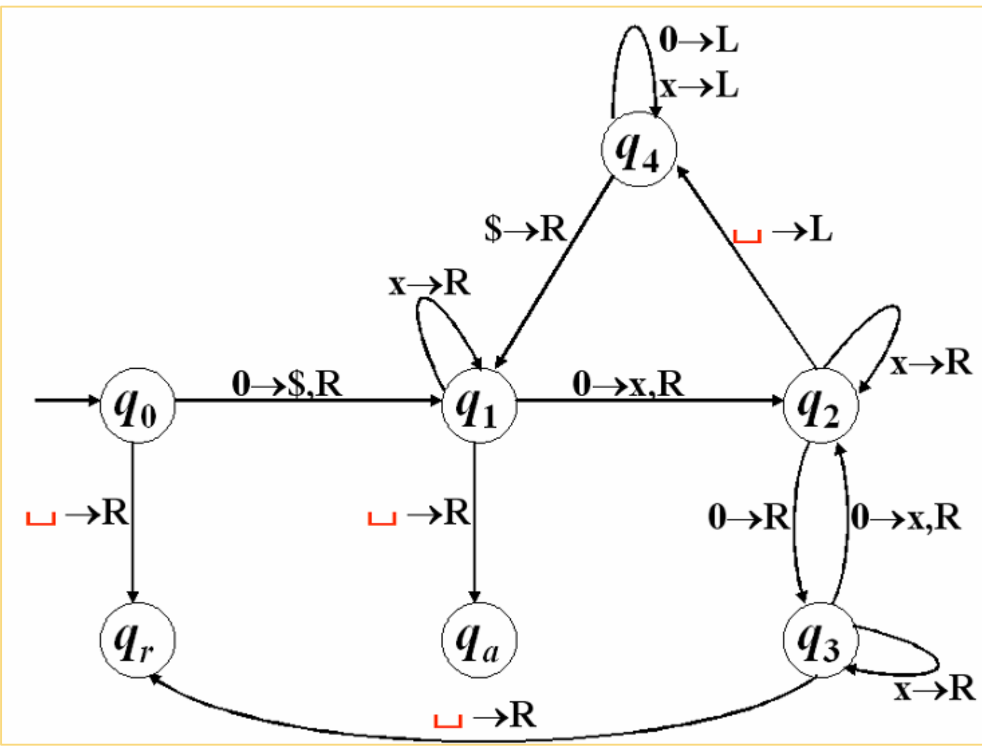
3.1. 时间复杂性

- 判定器M的运行时间或时间复杂度是 f:N→N, $f(n)$是M在所有长为n的输入上运行的最大步数.
- 若$f(n)$是M的运行时间, 则称
  M在时间$f(n)$内运行 或 M是$f(n)$时间图灵机
- 例:Σ={0}, C={0^k:k=2^n,n≥0} 图灵可判定语言



时间复杂性

- 例:$\Sigma=\{0\}$，$C=\{0^k:k=2^n, n\ge 0\}$
- 图灵可判定语言



$q_0$00_
$\$q_1 0$_
xq_2_
$\$q_4 x$_
$q_4$$x_
$\$q_1 x$_
xq_1_
$x_$q_a$

$f(2) = 7$

$q_0$000_
$\$q_1 00$_
$xq_2$0_
$\$x0q_3$_
$x0_$q_r$

$f(3) = 4$

$q_0$0000_
$\$q_1 000$_
$xq_2$00_
$\$x0q_3 0$_
$x0x$q_2$_
$\$x0q_4 x$_
$xq_4$0x_
$\$q_4 x0x$_
$q_4$$x0x_
$\$q_1 x0x$_
...
$q_4$$xxx_
...
$xxx_$q_a$

$f(4) = 21$

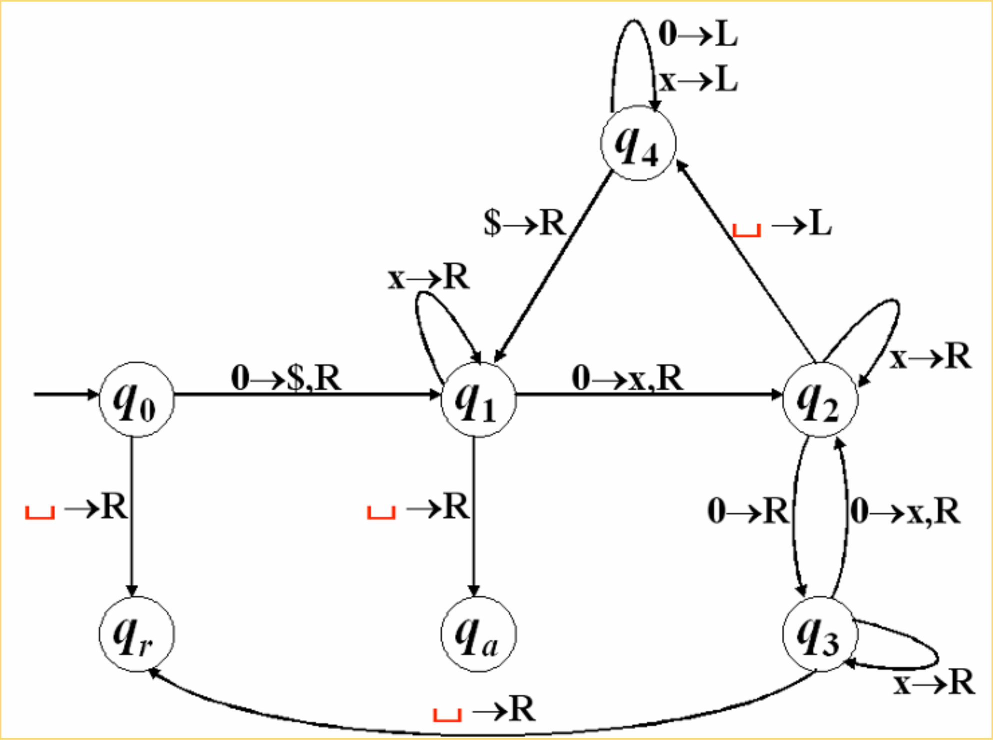
$f(2t+1) = 2t+2$

$f(2^k) = (2k+1)2^k+1$

$f(4) =21$

时间复杂性

- 判定器M的运行时间或时间复杂度是f:N→N,
- $f(n)$是M在所有长为n的输入上运行的最大步数.



$f(1) = 2$, $f(2) = 7$

$f(3) = 4$, $f(4) = 21$

…

$f(2t+1) = 2t+2$,

$f(2^k) = (2k+1)2^k+1$,

$n+1 \le f(n) \le 3n\log n$

$f(n) = O(n \log n)$

大O与小o记法

对于函数$f,g:\mathrm{N}\to\mathrm{R}^+$,

记$f(n)=O(g(n))$,若存在$c>0$使得

$$\lim_{n\to\infty}\frac{f(n)}{g(n)}\le c$$

$f(n)=O(g(n))$

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

记$f(n)=o(g(n))$,若

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

$f(n)=o(g(n))$

$f(n)<cg(n)$

$n+1\le f(\mathrm{n})\le 3\mathrm{nlogn}$

$f(n)=O(n\log n)$

大O与小o记法

- $f(n)=o(g(n))$:
  - 1) $\sqrt{n}=o(n)$
  - 2) $n=o(n\log\log n)$
  - 3) $n\log\log n=o(n\log n)$
  - 4) $n\log n=o(n^2)$
  - 5) $n^2=o(n^3)$

记$f(n)=o(g(n))$,若

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

多项式界, 指数界

- 多项式界：　$n^{O(1)}/n^{c}$(c>0，常数)
  - ¶ $n^{0.1}$, $n^{0.99}$, n, $n^{1.1}$, n^{2}, $n^{2.57}$, n^{3}, n^{10}, n^{100}, ……

- 指数界: $2^{n^{O(1)}}$/ $2^{n^{\delta}}$(δ>0, 常数)
  - ¶ $2^{n^{0.1}}, 2^{n^{0.5}}, 2^{n^{1}}, 2^{n^{2}}, 2^{n^{10}}, n!, n^{n}$

$$n! = o(n^n)$$

$$n! = \omega(2^n)$$

$$\log n! = \Theta(n \log n)$$

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$

多项式与指数的区别

- 多项式与指数在增长率上有巨大差异，
  - 当n=1000时，
  - n^3=10^9
  - 2^n = 1.07150861E301 >宇宙中原子数（10^78～10^82）
- 多项式有稳定性
  - 对加,减,乘,除,合成封闭
  - 计算模型互相模拟的开销是多项式的

2^{1000}=

10715 08607 18626 73209 48425 04906 00018
10561 40481 17055 33607 44375 03883 70351
05112 49361 22493 19837 88156 95858 12759
46729 17553 14682 51471 45285 69231 40435
98457 75746 98574 80393 45677 74824 23098
54210 74605 06237 11418 77954 18215 30464
74983 58194 12673 98767 55916 55439 46077
06291 45711 96477 68654 21676 60429 83165
26243 86837 20566 80693 76

2^{32}=1024*1024*1024*4
=4294967296

图灵机M_1

讨论语言$A = \{ 0^k1^k \mid k \geq 0 \}$的复杂性:

M_1="对输入串w:

(1)扫描带,如果在1的右边发现0，则拒绝. $2n=O(n)$

(2)如果0和1都在带上，就重复下一步. $2n=O(n)$

(3)扫描带，删除一个0和一个1. $2n=O(n)$

(4)如果带上同时没有0和1，就接受." $n=O(n)$

000111
*00111
$00x11
$$0xx1
$$$xxx
accept
12+7×3+3=36

000011
*00011
$000x1
$$00xx
$$$0xx
reject
12+9×2+4=34

001100
*01100
reject
5

时间分析: $f(6) = 42$(平均), $f(n) \geq 1$,

(1) $2n=O(n)$, (4) $n=O(n)$,

$\{$ (2) $2n=O(n)$ + (3) $2n=O(n)$ $\} \times (n/2) = O(n^2)$

所以M_1的运行时间是$O(n^2)$.

$f(6) = 42$

时间复杂性类

- **定义**: 对于函数t:N→N，时间复杂性类 **TIME(t(n))** 定义为:

  TIME(t(n)) = { L | 存在O(t(n))时间图灵机判定L}

例：因为M_1是时间$O(n^2)$图灵机,

所以A = $\{0^k1^k:k\geq 0\} \in$ TIME(n^2).

是否存在更快的TM判定A呢?

图灵机M_2

M_2=“对输入串w:

1)扫描带,若1的右边有0,则拒绝.

2)若0,1都在带上,重复以下步骤.

3) 检查带上0,1总数的奇偶性,

若是奇数, 就拒绝.

4) 再次扫描带,

第1个0开始,隔1个0删除1个0;

第1个1开始,隔1个1删除1个1.

5)若带上同时没有0和1,则接受.

否则拒绝.”

0000011111
*000011111
$0x0xx1x1x
$xx0xxxx1x
$xxxxxxxxx
accept
20+20×3+10
=70

000111
*00111
$0xx1x
$xxxxx
accept
12+12×2+6
=30

0001111
*001111
$0xx1x1
reject

00111111
*0111111
$0x1x1x1
$xxxx1xx
$xxxx1xx
reject

00011111
*0011111
$0xx1x1x
$xxxxx1x
reject

图灵机M_2

M_2="对输入串w:

1)扫描带,若1的右边有0,则拒绝. $O(n)$

2)若0,1都在带上,重复以下步骤. $O(n)$

3) 检查带上0,1总数的奇偶性, 若是奇数,就拒绝. $O(n)$

4) 再次扫描带, 第1个0开始,隔1个0删除1个0; 第1个1开始,隔1个1删除1个1. $O(n)$

×log n

5)若带上同时没有0和1,则接受. 否则拒绝." $O(n)$

总时间: $O(n\log n)$

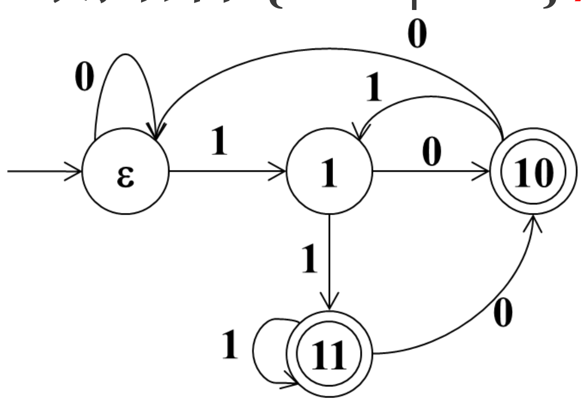
$\{0^k1^k | k\geq 0\} \in \mathrm{TIME}(n\log n)$

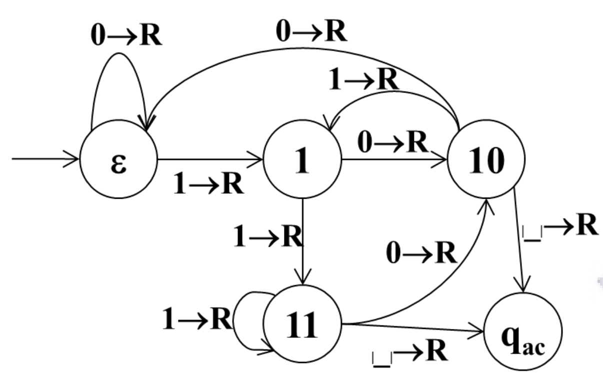
- 由M_2知道$A\in\mathrm{TIME(nlogn)}$.
- 有没有更快的单带确定TM识别A? 没有!
- 对于单带确定图灵机, 由
- 定理: 时间$o(n\log n)$的单带图灵机判定的语言是正则语言.

$\mathrm{TIME}(o(n\log n)) \subseteq$ 正则语言类 $\subseteq \mathrm{TIME}(n) \subseteq \mathrm{TIME}(o(n\log n))$

正则语言类 $= \mathrm{TIME}(n) = \mathrm{TIME}(o(n\log n))$

非正则语言 $\{0^k1^k \mid k\geq 0\} \notin \mathrm{TIME}(o(n\log n))$





第三章 计算复杂性

- 1. 时间复杂性

  $\{ 0^k1^k \mid k\geq 0 \}$的时间复杂性分析
- 2. 不同模型的复杂性关系

  单带与多带 确定与非确定
- 3. P类与NP类
- 4. NP完全性及NP完全问题

——第7章 时间复杂性

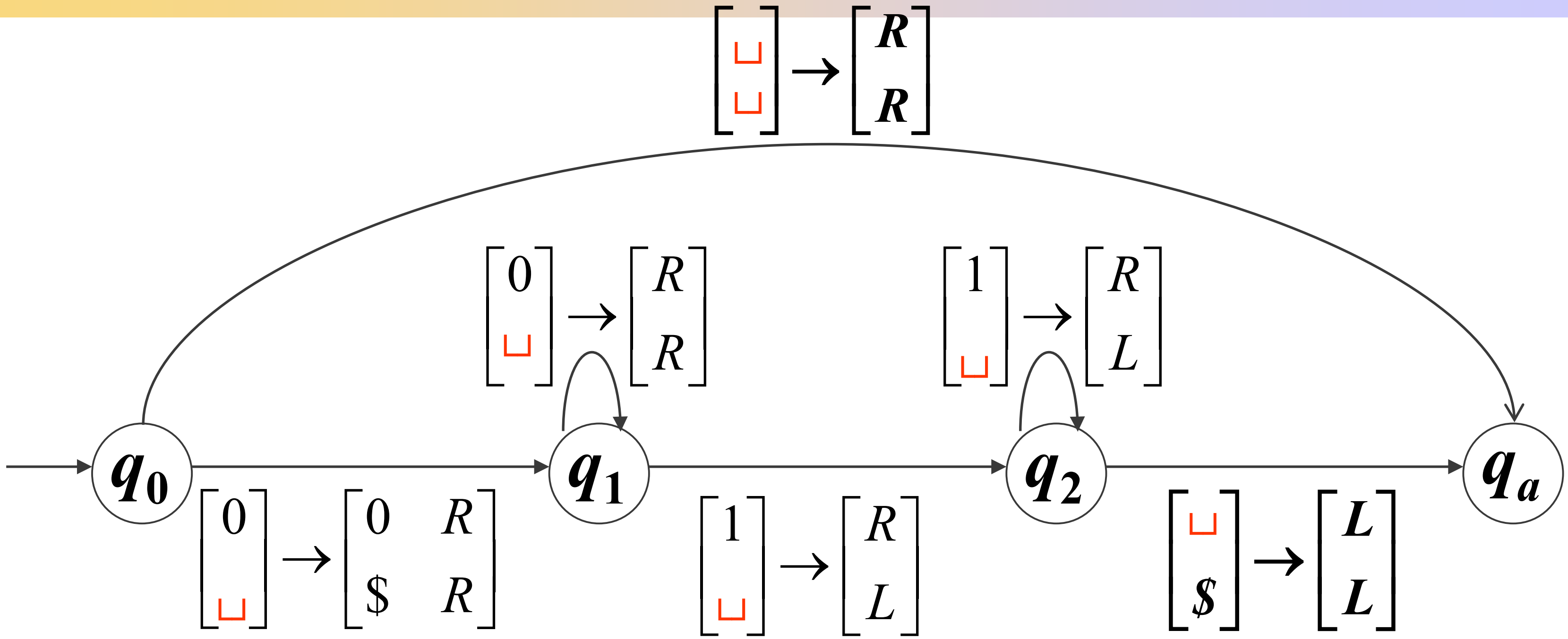
单带与多带图灵机复杂性关系

$\{ 0^k1^k \mid k\geq 0 \}$ 有 $O(n)$ 时间双带图灵机

M_3=“对输入串w:

1) 扫描1带,如果在1的右边发现0,则拒绝.
2) 将1带的1复制到2带上.
3) 每删除一个1带的0就删除一个2带的1.
4) 如果两带上同时没有0和1,就接受.”

$\{0^k1^k|k\geq 0\}$的O(n)时间双带TM (补充)



0*	0	0	1	1	1	
*						q_0

0	0*	0	1	1	1	
$	*					q_1

0	0	0*	1	1	1	
$		*				q_1

0	0	0	1*	1	1	
$			*			q_1

0	0	0	1	1*	1	
$		*				q_2

0	0	0	1	1	1*	
$	*					q_2

0	0	0	1	1	1	*
$*						q_2

0	0	0	1	1	1*	
$*						q_a

单带与多带图灵机复杂性关系

定理:设函数$t(n)\geq n$, 则每个$t(n)$时间多带图灵机和某个$O(t^2(n))$时间单带TM等价.

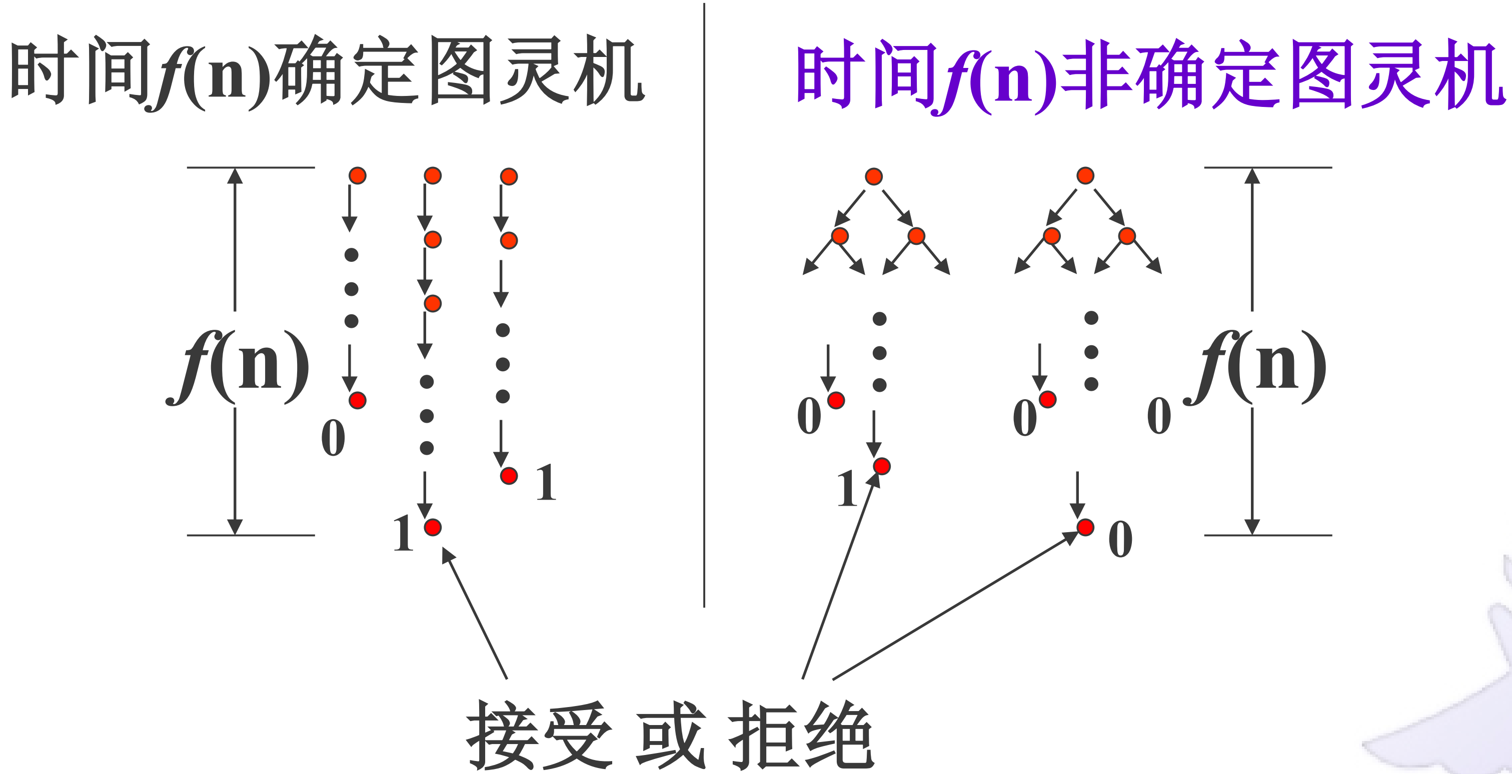
- 证明思路:
  - 用单带图灵机模拟多带图灵机,
  - 每步模拟需要$O(t(n))$ 时间
  - 总共模拟$t(n)$步
  - 所以总的模拟时间$O(t^2(n))$

#	0	0	1	0	#	a	a	b	b	#	b	a		

…………

非确定判定器的运行时间

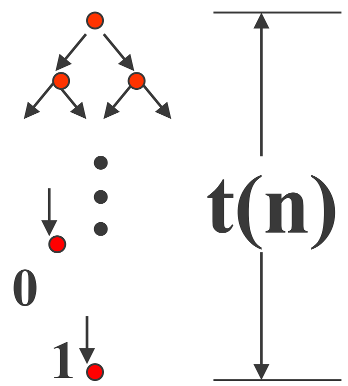
- 定义: 对非确定型判定器N, 其运行时间f(n)是在所有长为n的输入上, 所有分支的最大步数.



确定性与非确定性图灵机复杂性关系

定理: 设$t(n)\geq n$, 则每个$t(n)$时间**NTM** 都有一个$2^{O(t(n))}$时间单带**DTM**与之等价.

- 证明思路:
  - 设b是所有节点的最大分支数
  - 每个分支模拟需要$O(t(n))$时间,
  - 总共模拟$b^{t(n)}$个分支



定理:设$t(n)\geq n$, 则$\mathbf{NTIME}(t(n)) \subseteq \mathbf{TIME}(2^{O(t(n))})$

第三章 计算复杂性

- 1. 时间复杂性

  $\{ 0^k1^k \mid k\geq 0 \}$的时间复杂性分析

- 2. 不同模型的复杂性关系

  单带与多带 确定与非确定

- 3. P类与NP类

- 4. NP完全性及NP完全问题

——第7章 时间复杂性

多项式时间

- 运行时间相差多项式可以认为是小的

  相差指数可以认为是大的.

- 有关素性测试: Prime = { p | p是素数 }
  - 如何编码? 一进制,二进制,十进制?
  - 典型的指数时间算法来源于蛮力搜索.
  - 有时通过深入理解问题可以避免蛮搜.
  - 2001年Prime被证明存在多项式时间算法. AKS算法

P类

- 定义: P是单带确定TM在 多项式时间内可判定的问题,即

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

- P类的重要性在于:
  - 1）对于所有与单带确定TM等价的模型, P不变.
  - 2）P大致对应于在计算机上实际可解的问题.

    实际算法往往是$O(n)$, $O(n^2)$,$O(n^3)$, 几乎没有$O(n^{100})$.

- 研究的核心是一个问题是否属于P类.

Polynomial time

一些P类问题

- **Path：有向图中s到t是否有路径**
- **两个整数互素问题(m, n)(m>n)**
  - ¶ 最大公因子为1
  - ¶ 欧几里德算法，辗转相除法
- **模p指数运算 $a^b \bmod p$**
- **素性测试**

- 以增加空间复杂性来减小时间复杂性

P类

- 如何判断是否属于P类
  - 算法在长为n的输入上运行时，所需步骤为多项式上界
  - 算法的每一步都在多项式时间内完成。

- M="对于输入<G, s, t>, G是包含顶点s和t的有向图,

1次 1) 在顶点s上做标记.

2) 重复如下步骤, 直到不再有顶点被标记.

*m*次 3) 扫描G的所有边。如果找到一条边(a, b)，a被标记而b未被标记，则标记b.

1次 4) 若顶点t已标记, 则接受; 否则, 拒绝."

总的执行次数：m+2

各个步骤都能在多项式时间内完成。

NP类

- 定义:NP类是单带非确定TM在多项式时间内可判定的问题,即

$$NP = \cup_k NTIME(n^k)$$

$$EXP = \cup_k TIME(2^{O(n^k)})$$

$$P \subseteq NP \subseteq EXP$$

$$P \subset EXP$$

Nondeterministic **P**olynomial time

NP问题举例

- HP = {<G,s,t>|G是包含从s到t的哈密顿路径的有向图}
- CLIQUE={<G, k>|G是有k团的无向图}

- 目前没有快速算法
- 但其成员是可以快速验证的（有多项式时间验证机）.
- 快速验证的特点:
  1. 只需要对语言中的串能快速验证.
  2. 验证需要借助额外的信息:证书,身份证.
- 注意:HP的补可能不是可以快速验证的.

NP类

- 定义: NP= { L | 语言L有多项时间验证机}
  - 解的长度是输入规模的多项式
  - 验证解所花费时间是输入规模的多项式
  - 候选解个数是输入规模的指数

CLIQUE∈NP

- 团:无向图的完全子图(所有节点都有边相连).

  CLIQUE = { <G, k> | G是有k团的无向图 }

- 证明1： 构造验证机 TM V

  V=“ 对输入<<G, k>,c>:

  1) 检查c是否是G中k个顶点的集合;

  2) 检查G是否包含连接c中顶点的所有边;

  3) 若两项检查都通过,则接受;

  否则,拒绝.”

  V是多项式时间图灵机

  #

CLIQUE∈NP

- 团:无向图的完全子图(所有节点都有边相连).

  CLIQUE = { <G, k> | G是有k团的无向图 }

- 证明2：构造单带非确定TM N
- N="对于输入<G, k>,这里G是一个图:

  1)非确定地选择G中k个节点的子集c.

  2)检查G是否包含连接c中节点的所有边.

  3)若是,则接受;

  否则,拒绝."

哈密顿路径问题HP∈NP

- HP={ <G,s,t> | G是包含从s到t的哈密顿路径的有向图}
- 证明：P时间内判定HP的NTM:

N_1="对于输入<G,s,t>:

1)非确定地选G的所有节点的排列$p_1,\ldots p_m$.

2)若$s=p_1,t=p_m$,且对每个i, (p_i,p_{i+1})是G的边,则接受;否则拒绝."

#

子集和问题

- 子集和问题:

  SUBSET-SUM= { <S,t> | S={$x_1,x_2,\ldots,x_k$}, 且存在
  {$y_1,y_2,\ldots,y_l$}$\subseteq${$x_1,x_2,\ldots,x_k$}, 使得 $\Sigma y_i=t$ }
  注: {$x_1,x_2,\ldots,x_k$}, {$y_1,y_2,\ldots,y_l$}是多重集。

- 例: <{4,11,16,21,27},25> $\in$ SUBSET-SUM,

  4+21=25

SUBSET-SUM∈NP

- 证明: 验证机DTM V
  - V="对输入<<S, t>, c>:
  - 1) 检查S是否包含c中所有数;
  - 2) 检查c是否总和为t的数的集合;
  - 3) 若两项检查都通过,则接受;

    否则,拒绝." #

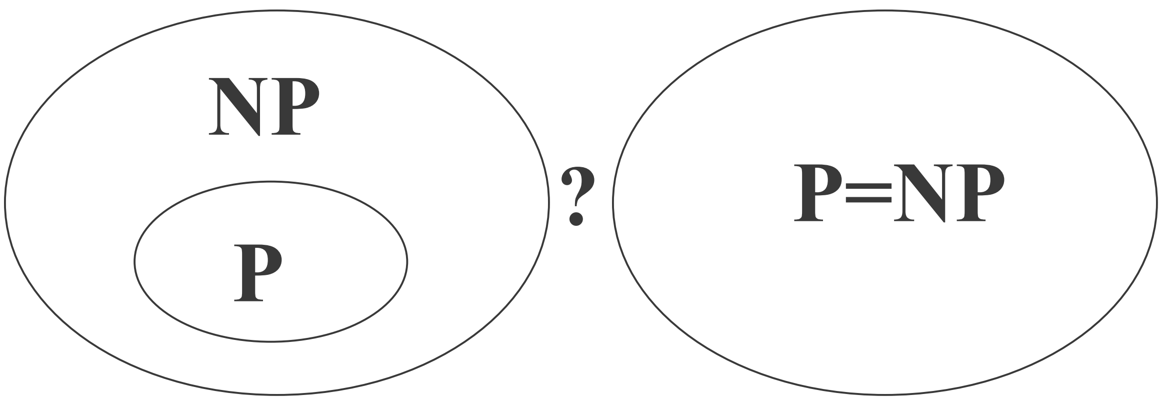
P与NP

- P=成员资格可以快速判定的语言类.
- NP=成员资格可以快速验证的语言类.

显然有 $P \subseteq NP$

但是否有 $P = NP$ ？

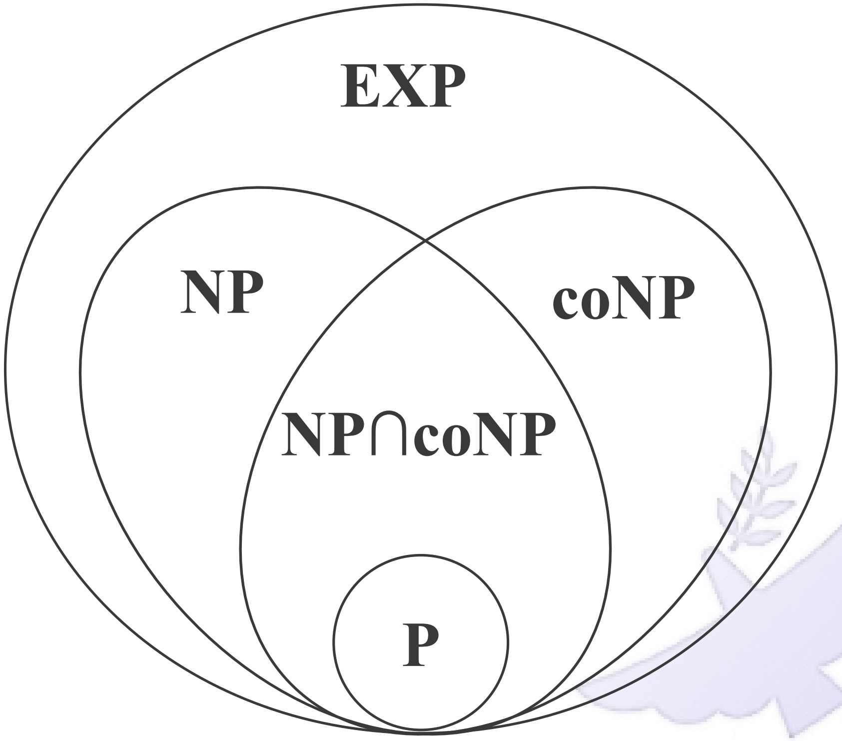
看起来难以想象, 但是现在没有证明.



当代数学与理论计算机共同的难题.

coNP

- coNP=NP中语言的补语言
  - 例: $\mathrm{CLIQUE}^c \in \mathrm{coNP}$
- NP =? coNP是个难题
- $P \in NP \cap coNP$



第三章 计算复杂性

- 1. 时间复杂性

  $\{ 0^k1^k \mid k\geq 0 \}$的时间复杂性分析

- 2. 不同模型的复杂性关系

  单带与多带 确定与非确定

- 3. P类与NP类

- **4. NP完全性及NP完全问题**
  - **NP完全性的定义**
  - **SAT是NP完全问题**
  - 一些NP完全问题

NP完全性

- Cook(美)和Levin(苏联)于1970's证明：

  NP中某些问题的复杂性与整个NP类的复杂性相关联, 即：

  若这些问题中的任一个找到P时间算法,则P=NP.

  这些问题称为NP完全问题.

- 理论意义:两方面
  - 1)研究P与NP关系可以只关注于一个问题的算法.
  - 2)可由此说明一个问题目前还没有快速算法.

可满足性问题SAT

- 布尔变量: 取值为1和0(True, False)的变量.
- 布尔运算: AND($\wedge$),OR ($\vee$),NOT ($\neg$). 布尔公式.
  - 例: $\phi_1 = ((\neg x) \wedge y) \vee (x \wedge (\neg z))$, $\phi_2 = (\neg x) \wedge x$
- 可满足性问题: 给定一个布尔公式, 确定这个公式是否可满足
  - 称ϕ可满足, 若存在布尔变量的0,1赋值使得ϕ=1.
  - 例: $(\neg x \vee (\neg y \wedge x)) \wedge (x \vee y)$
  - x=1,y=0是可满足赋值

SAT= { <ϕ> | ϕ是可满足布尔公式}

合取范式可满足性问题

- 文字: 变量或变量的非,如x或$\neg$x.
- 子句:由$\vee$连接的若干文字,如$x_1 \vee (\neg x_2) \vee x_3 \vee x_4$.
- 合取范式(cnf):由$\wedge$连接的若干子句,如
  - $((\neg x_1) \vee x_2 \vee (\neg x_3)) \wedge (x_2 \vee (\neg x_3) \vee x_4 \vee x_5) \wedge ((\neg x_4) \vee x_5)$
- k-cnf (conjunctive normal form)
  - 每个子句的文字数不大于k: 3cnf, 2cnf

可满足性问题SAT

- 可满足性问题:

  SAT = { <ϕ> | ϕ是可满足的布尔公式 } NP完全

- 二元可满足性问题:

  2SAT = { <ϕ> | ϕ是可满足的2cnf } $\in$ P

- 三元可满足性问题:

  3SAT = { <ϕ> | ϕ是可满足的3cnf } NP完全

二元可满足问题2SAT∈P

- 1. 当2cnf中有子句是单文字x, 则反复执行(直接)清洗
  - 1.1 由x赋值，删去含x的子句
  - 1.2 删去含$\neg x$的文字

  若清洗过程出现相反单文子子句, 则清洗失败并结束
- 例如：
  - $(x_1\vee x_2)\wedge(x_3\vee\neg x_2)\wedge(x_1)\wedge(\neg x_1\vee\neg x_2)\wedge(x_3\vee x_4)\wedge(\neg x_3\vee x_5)\wedge(\neg x_4\vee\neg x_5)\wedge(\neg x_3\vee x_4)$
  - $\Rightarrow (x_3\vee\neg x_2)\wedge(\neg x_2)\wedge(x_3\vee x_4)\wedge(\neg x_3\vee x_5)\wedge(\neg x_4\vee\neg x_5)\wedge(\neg x_3\vee x_4)$
  - $\Rightarrow (x_3\vee x_4)\wedge(\neg x_3\vee x_5)\wedge(\neg x_4\vee\neg x_5)\wedge(\neg x_3\vee x_4)$

二元可满足问题2SAT∈P

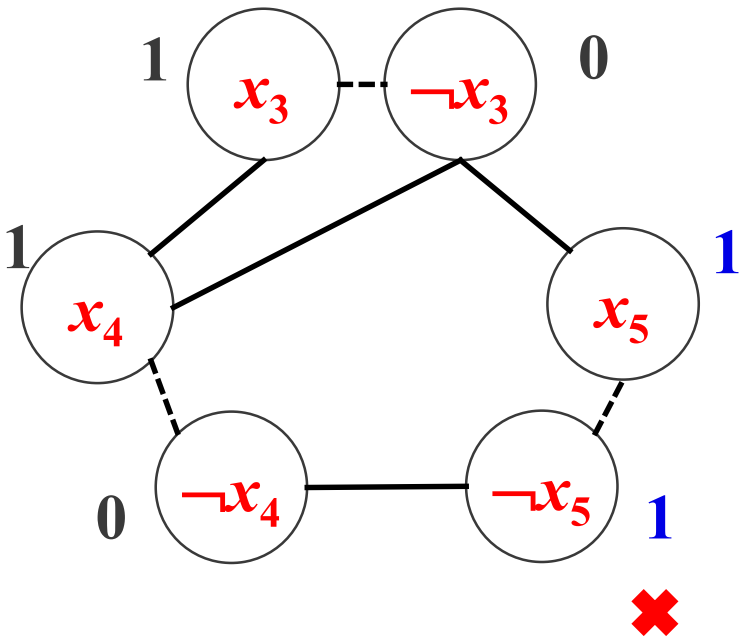
- 2. 若无单文字子句, 则
  - 针对变量赋（真/假）各清洗一次

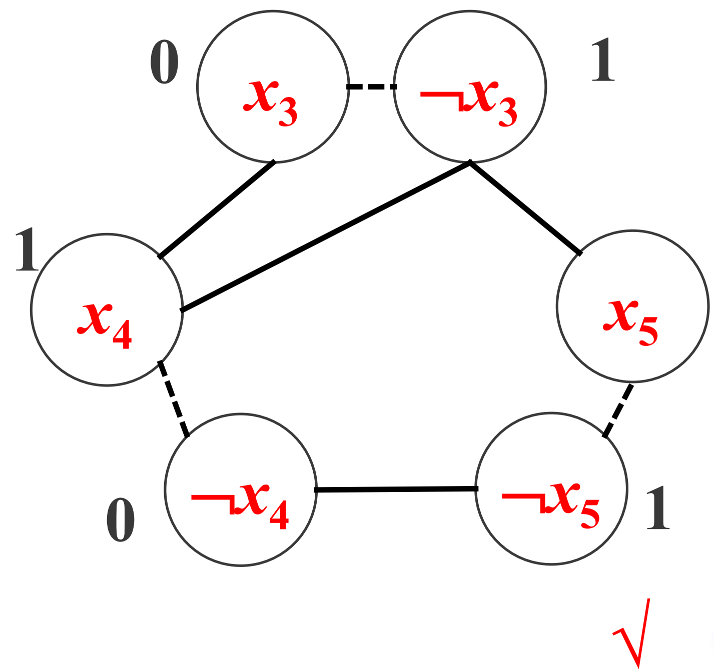
若两次都清洗失败, 则回答不可满足.

- 例如:
  - $(x_3 \vee x_4) \wedge (\neg x_3 \vee x_5) \wedge (\neg x_4 \vee \neg x_5) \wedge (\neg x_3 \vee x_4)$
  - $x_3=1 \Rightarrow (x_5) \wedge (\neg x_4 \vee \neg x_5) \wedge (x_4) \Rightarrow (\neg x_4) \wedge (x_4)$ 失败
  - $x_3=0 \Rightarrow (x_4) \wedge (\neg x_4 \vee \neg x_5) \Rightarrow (\neg x_5) \Rightarrow \varnothing$ 成功
- 3. 若成功清洗后有子句剩下, 则
  - 继续2.
  - 否则, 回答可满足.

二元可满足问题2SAT∈P

- $(x_3 \vee x_4) \wedge (\neg x_3 \vee x_5) \wedge (\neg x_4 \vee \neg x_5) \wedge (\neg x_3 \vee x_4)$





3SAT∈NP

- 三元可满足性问题:

  3SAT = { $\langle\phi\rangle$ | ϕ是可满足的3cnf }

- P时间内判定3SAT的NTM:

  N="对于输入$\langle\phi\rangle$, ϕ是一个3cnf公式,

  1)非确定地选择各变量的赋值T.

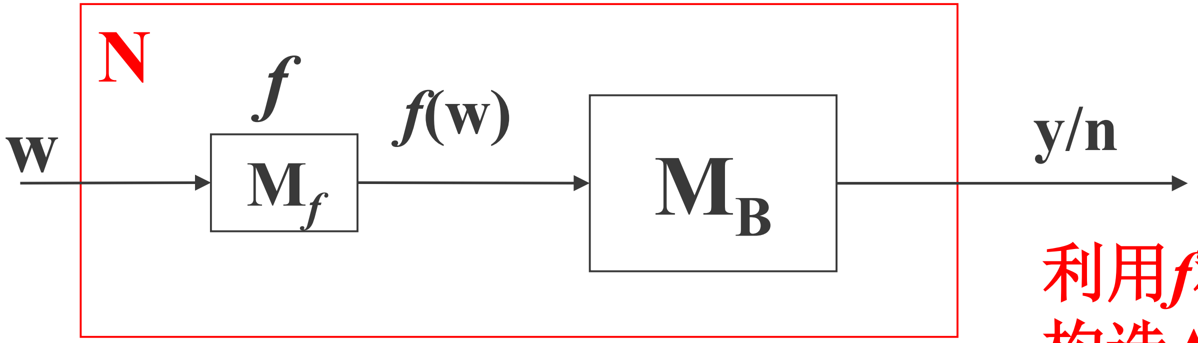
  2)若在赋值T下 ϕ=1, 则接受;否则拒绝."

- 第2步在公式长度的多项式时间内运行.

多项式时间映射归约

- 定义:多项式时间可计算函数$f:\Sigma^*\to\Sigma^*$.
  若∃多项式时间图灵机, ∀w输入, M停机时带上的串为$f(w)$ 。
- 定义:称A可多项式时间映射归约到B ($A\le_P B$),
  若∃多项式时间可计算函数$f:\Sigma^*\to\Sigma^*$,
  $$\forall w\in\Sigma^*,\ w\in A \Leftrightarrow f(w)\in B.$$
  函数f称为A到B的多项式时间归约.

通俗地说: f将A的实例编码在多项式时间内转换为B的实例编码.



利用f和B的判定器构造A的判定器N

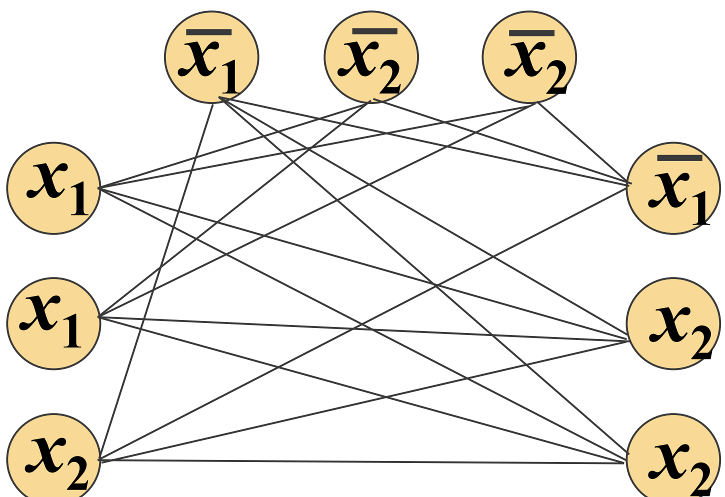
定理: 3SAT $\leq_P$ CLIQUE

- 3SAT = {<ϕ> | ϕ是可满足的3cnf公式 }
- CLIQUE = { <G, k> | G是有k团的无向图 }.
- 证明:设$\phi=(a_1\vee b_1\vee c_1)\wedge\ldots\wedge(a_k\vee b_k\vee c_k)$,有k个子句.

令 f(ϕ) = <G,k>, G有k组节点,每组3个，每个子句的变量是一组;

同组节点无边相连, 相反标记无边相连.

- 例: $f((x_1\vee x_1\vee x_2)\wedge(\overline{x_1}\vee\overline{x_2}\vee\overline{x_2})\wedge(\overline{x_1}\vee x_2\vee x_2))$ = <G,3>



需证:<ϕ>∈3SAT

⇔

<(G, k)>∈CLIQUE

$\forall\phi,\ \phi\in 3SAT \Leftrightarrow f(\phi)\in CLIQUE$

- 例: $f((x_1\vee x_1\vee x_2)\wedge(\overline{x_1}\vee\overline{x_2}\vee\overline{x_2})\wedge(\overline{x_1}\vee x_2\vee x_2)) = \langle G,3\rangle$
- ∃变量赋值($x_1=0,\ x_2=1$)使得$\phi=1$
- $\Leftrightarrow$ ∃k团(每组挑一个顶点得到k团, 非同组非相反)
- $\Leftrightarrow f(\phi)$ ($\langle G, 3\rangle$) $\in$ CLIQUE.

$f(\phi)$在$|\langle\phi\rangle|$的多项式时间内可计算:

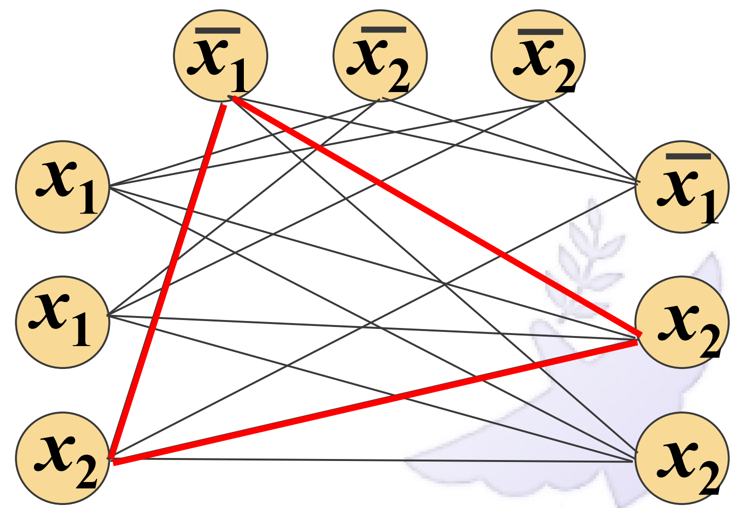
设ϕ的长度3k=O(k),

则G的顶点数3k=O(k),

G的边数是$O(k^2)$)

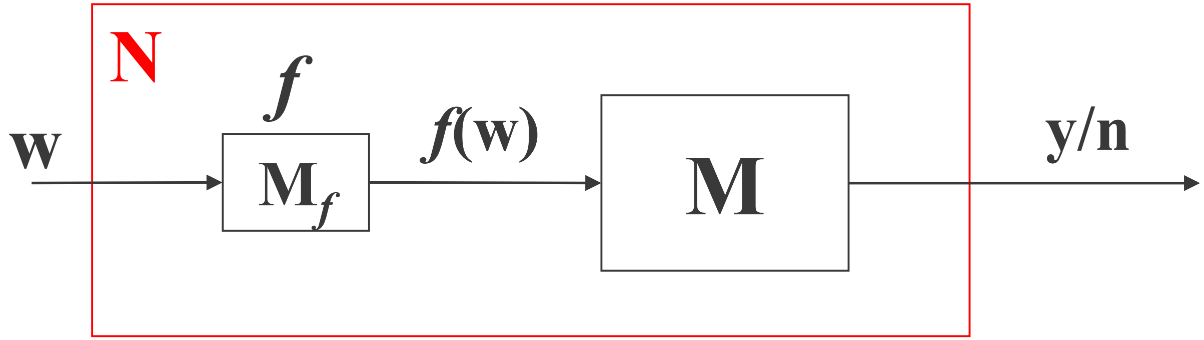
可见 $f(\phi) = \langle G, k\rangle$的构造

可在k的多项式时间内完成.



归约引理

- 归约引理：若$A\le_P B$且$B\in P$, 则$A\in P$
- 证明: 设$f:\Sigma^*\to\Sigma^*$是A到B的P时间映射归约,
  B有P时间判定器M, 则
  N=“输入w, 计算M(f(w)), 输出M的运行结果”
  在多项式时间内判定A.
- 问题: 若f是n^a时间归约, M是n^b时间判定器, 则N时间?
  设$|w|=n$, 则$|f(w)|\le n^a$, 则M(f(w))时间 $\le n^{ab}$.



$$\forall w\in\Sigma^*,\ w\in A \Leftrightarrow f(w)\in B.$$

NP完全性（NPC）

- 定义:语言B称为NP完全的(NP-complete), 若它满足:

  1) $B \in NP$;

  2) $\forall A \in NP$, 都有$A \le_p B$.

- 定理1(归约引理): 若$A \le_p B$ 且 $B \in P$ 则 $A \in P$.
- 定理2: 若B是NPC的, 且$B \in P$, 则P=NP.

  证明: $\forall A \in NP$, $A \le_p B$ 且 $B \in P \Rightarrow A \in P$

- 定理3: 若B是NPC的, $B \le_p C$, 且$C \in NP$, 则C是NPC.

  证明: $\forall A \in NP$, $(A \le_p B)$ 且 $(B \le_p C) \Rightarrow A \le_p C$

C-L定理

- **Cook-Levin定理:**

  对任意$A \in NP$都有$A \leq_p SAT$.

- **推论**: 若$SAT \in P$, 则 $NP = P$.

- 若3SAT是NPC 且 $3SAT \leq_p CLIQUE \Rightarrow$ CLIQUE是NPC

Cook-Levin定理的证明步骤

- 定义:语言B称为NP完全的(NP-complete), 若它满足:

  1) $B \in NP$;

  2) $\forall A \in NP$, 都有$A \le_P B$.

- Cook-Levin定理: SAT是NP完全问题.
- 证明步骤:

1. $SAT \in NP$(易证)

2. $\forall A \in NP$, $A \le_P SAT$

Cook-Levin定理的证明步骤

- 1. SAT∈NP(易证)
- 证明：P时间内判定SAT的NTM:

  N="对于输入< ϕ>, ϕ是一个布尔公式,

  1)非确定地选择各变量的赋值T.

  2)若在赋值T下 ϕ=1, 则接受;否则拒绝."

- 第2步在公式长度的多项式时间内运行.

$\forall A \in NP$, 都有 $A \le_p SAT$

- 证明思路: 将字符串w对应到布尔公式ϕ，$\phi = f(w)$

  利用图灵机接受的形式定义进行证明.

- 过程: 任取$A \in NP$, 设N是判定A的n^k时间NTM.

  $\forall w(|w|=n)$, N接受w

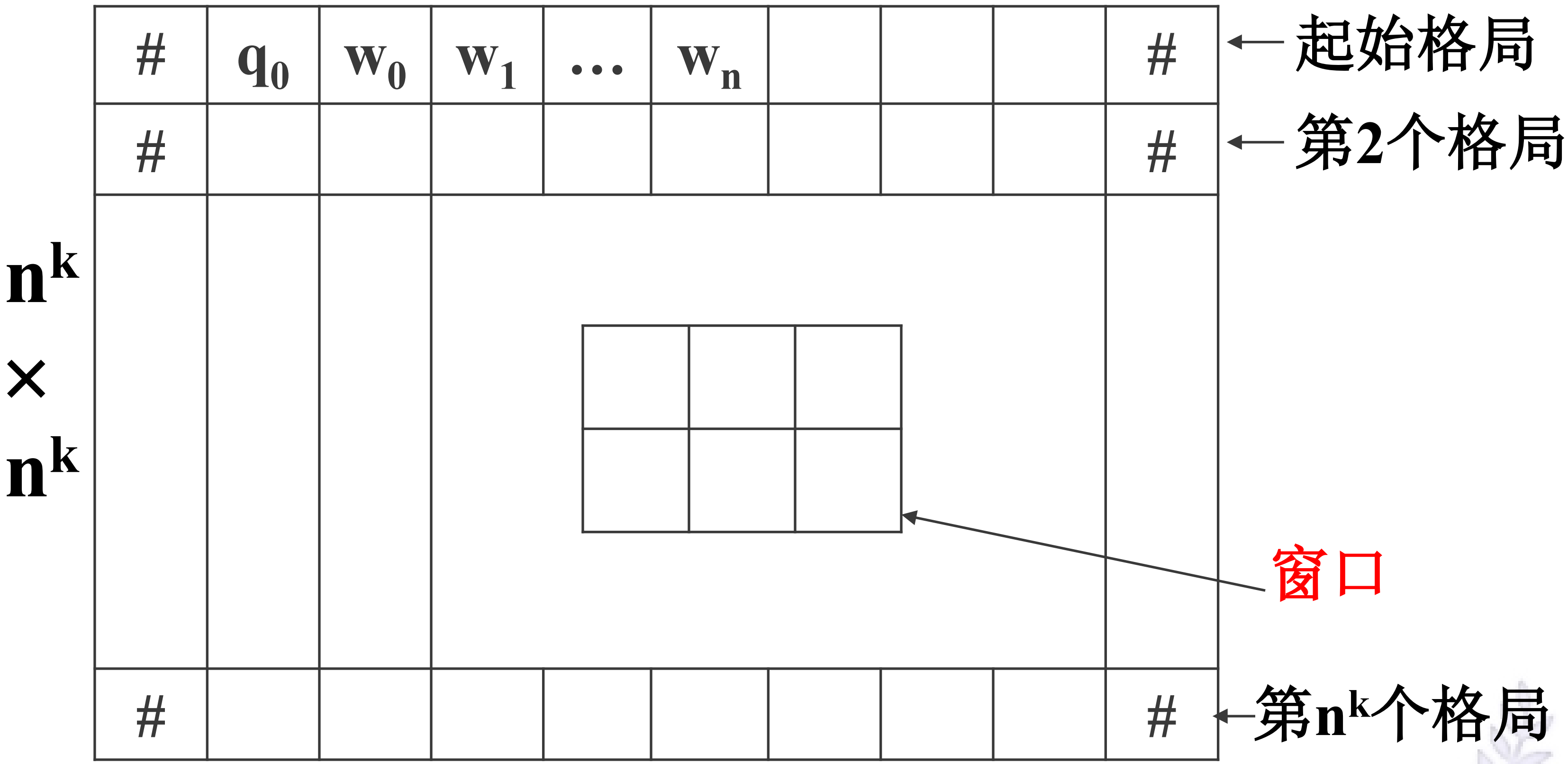
  $\Leftrightarrow$ N对w有长度小于n^k的接受格局序列

  $\Leftrightarrow$ 该序列能填好N在w上的画面(一个$n^k \times n^k$表格)

  $\Leftrightarrow$ $\phi = f(w)$可满足($|\langle\phi\rangle| = O(n^{2k})$)

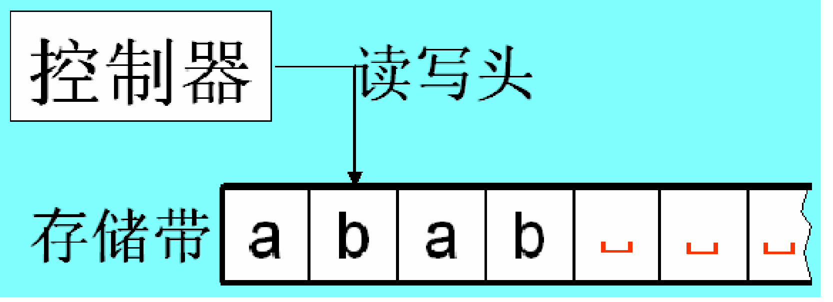
- 结论: SAT是NP完全的

N接受w⇔能填好N在w上的表



能填好表: 第一行是起始格局
上一行能产生(或等于)下一行
表中有接受状态

回忆图灵机(TM)形式化定义



TM是一个7元组$(Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$

1) Q是状态集.

2) Σ是输入字母表,不包括空白符␣.

3) Γ是带字母表,其中␣$\in\Gamma$, $\Sigma\subset\Gamma$.

4) δ: $Q\times\Gamma\rightarrow Q\times\Gamma\times\{L,R\}$是转移函数.

5) $q_0\in Q$是起始状态.

6) $q_a\in Q$是接受状态.

7) $q_r\in Q$是拒绝状态, $q_a \neq q_r$.

回忆图灵机格局的定义

- 描述图灵机运行的每一步需要如下信息:

控制器的状态; 存储带上字符串; 读写头的位置.

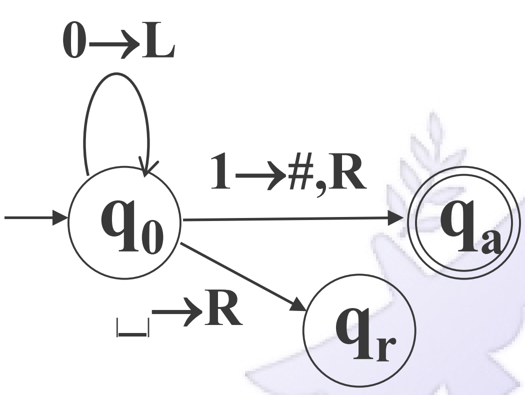
- 定义: 对于图灵机$M=(Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$,
- 设$q\in Q$, $u,v\in\Gamma^*$, 则格局 uqv 表示

1) 当前控制器状态为q ;

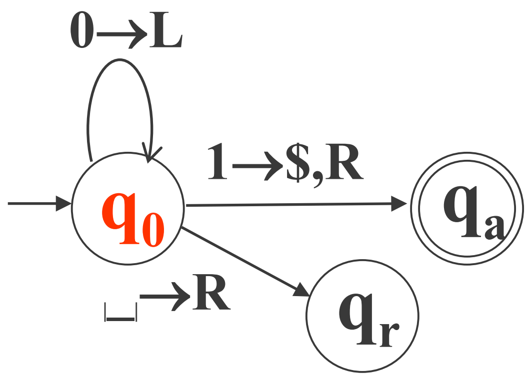
2) 存储带上字符串为uv(其余为空格);

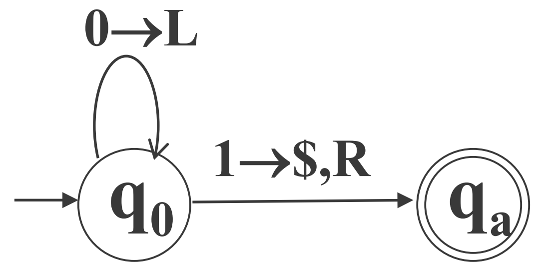
3) 读写头指向v的第一个符号.

- 起始格局, 接受格局, 拒绝格局.



格局演化举例





省略拒绝状态

q_0 0 1

q_0 0 1

…

循环

q_0 1 0

$ q_a 0

接受

q_0 _ _

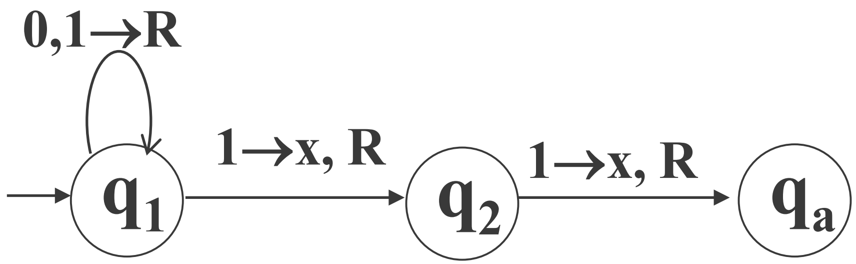
_ q_r _

拒绝

#	q_0	1	0	_	_	#
#	$	q_a	0	_	_	#

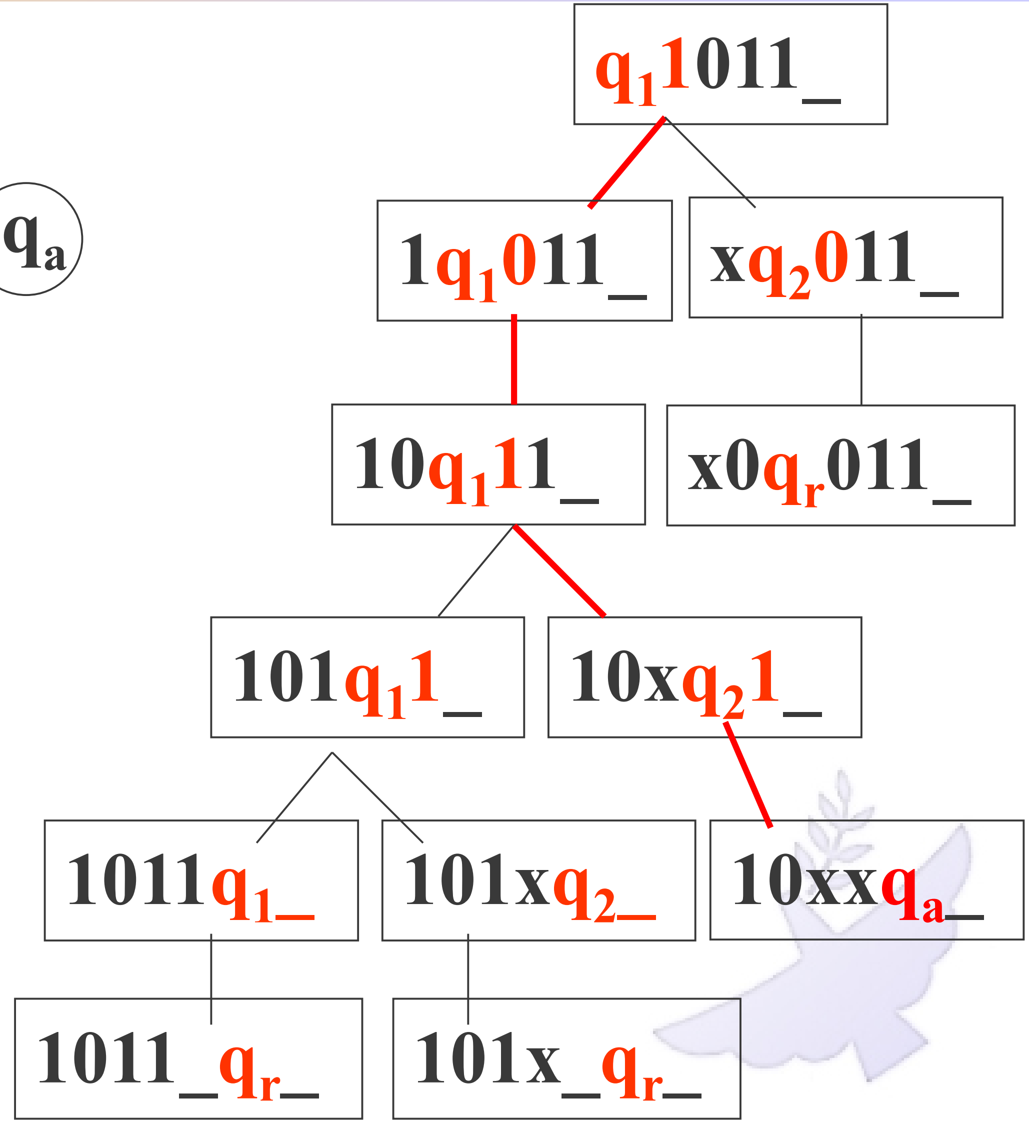
#	q_0	1	0	_	_	#
#	$	q_a	0	_	_	#
#	$	q_a	0	_	_	#
#	$	q_a	0	_	_	#

N接受w⇔能填好N在w上的表



NTM

#	q_1	1	0	1	1	_	#
#	1	q_1	0	1	1	_	#
#	1	0	q_1	1	1	_	#
#	1	0	x	q_2	1	_	#
#	1	0	x	x	q_a	_	#
#	1	0	x	x	q_a	_	#
#	1	0	x	x	q_a	_	#
#	1	0	x	x	q_a	_	#



构造布尔公式$\phi=f(w)$

- 能填好画面 $\Leftrightarrow$ $\phi=f(w)$可满足

  $f(w)=\langle\phi\rangle$, $\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accept}$.

- 对于任意赋值:
  - 1. $\phi_{cell}=1$ $\Leftrightarrow$ 每格有且只有一个符号;
  - 2. $\phi_{start}=1$ $\Leftrightarrow$ 第一行是起始格局;
  - 3. $\phi_{accept}=1$ $\Leftrightarrow$ 表格中有接受状态;
  - 4. $\phi_{move}=1$ $\Leftrightarrow$ 每行由上一行格局产生.
- $\forall w \in A \Leftrightarrow f(w)=\langle\phi\rangle \in SAT$， 即 $A \leq_m SAT$
- 若$|\langle\phi\rangle|$是$|w|$的多项式, 则有$A \leq_P SAT$

构造ϕ_{cell}:$\phi_{cell} = 1 \Leftrightarrow$ 每格有且只有一个符号

- ϕ的变量: $x_{i,j,s}$, $i,j=1,\ldots,n^k$, $s\in Q\cup\Gamma\cup\{\#\}$ //全体符号
- $x_{i,j,s}$: 第i行第j列是否填了符号s

$$\phi_{\text{cell}} = \bigwedge_{1\le i,j\le n^k}\{[\bigvee_{s} x_{i,j,s}]\wedge[\bigwedge_{s\ne t}(\overline{x_{i,j,s}\wedge x_{i,j,t}})]\}$$

$\bigvee_{s} x_{i,j,s} = 1$ $\Leftrightarrow$ (i,j)格中至少有一个符号

$\bigwedge_{s\ne t}(\overline{x_{i,j,s}\wedge x_{i,j,t}}) = 1$ $\Leftrightarrow$ (i, j)格中至多有一个符号

例: $(x_{i,j,1}\vee x_{i,j,2}\vee x_{i,j,3})\wedge(\overline{x_{i,j,1}\wedge x_{i,j,2}})\wedge(\overline{x_{i,j,1}}\vee\overline{x_{i,j,3}})\wedge(\overline{x_{i,j,2}}\vee\overline{x_{i,j,3}})$

- ϕ_{cell} 长$O(n^{2k})$

构造ϕ_{start}

$$\phi_{start} = x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \cdots \wedge x_{1,n^k,\#}$$

- 长$O(n^k)$
- $\phi_{start} = 1 \Leftrightarrow$ 第一行是起始格局;

#	q_0	w_0	w_1	…	w_n				#	起始格局
#									#	第2个格局

$n^k \times n^k$

构造ϕ_{accept}

$$\phi_{\text{accept}} = \bigvee_{1 \le i, j \le n^k} x_{i,j,q_{\text{accept}}}$$

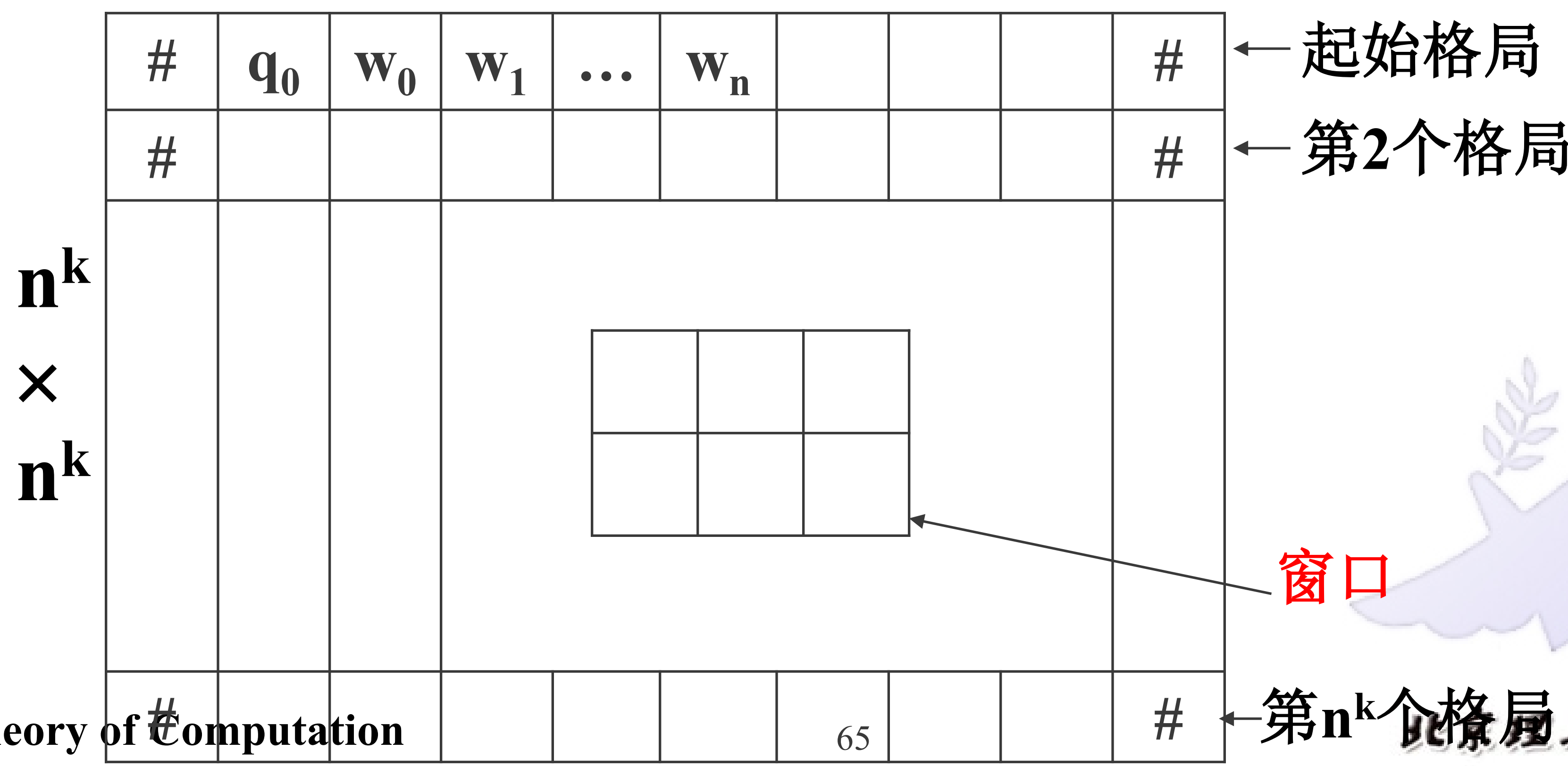
- 长$O(\text{n}^{2\text{k}})$
- $\phi_{\text{accept}} = 1 \Leftrightarrow$ 表格中有接受状态

构造ϕ_{move}

$\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accept}$.

ϕ_{move}确定表的每行是上一行的合法结果.

只需判断每个2×3窗口是否“合法”.



合法窗口

设$\delta(q_1,a)=\{(q_1,b,R)\}$，$\delta(q_1,b)=\{(q_2,c,L),(q_2,a,R)\}$

合法窗口

a	q_1	b
q_2	a	c

a	q_1	b
a	a	q_2

d	a	q_1
d	a	b

#	b	a
#	b	a

a	b	a
a	b	q_2

b	b	b
c	b	b

非法窗口

a	b	a
a	a	a

a	q_1	b
q_1	a	a

a	q_1	b
q_1	a	q_1

窗口$x_{i,j}$是合法的：

$$\bigvee_{\substack{a_1,a_2,\cdots,a_6 \\ \text{是合法窗口}}} [x_{i,j-1,a_1} \wedge \cdots \wedge x_{i+1,j+1,a_6}]$$

合法窗口有常数个

设$\delta(q_1,a)=\{(q_1,b,R)\}$，$\delta(q_1,b)=\{(q_2,c,L),(q_2,a,R)\}$

合法窗口

a	q_1	b
q_2	a	c

a	q_1	b
a	a	q_2

d	a	q_1
d	a	b

#	b	a
#	b	a

a	b	a
a	b	q_2

b	b	b
c	b	b

$$\phi_{\text{move}} = \bigwedge_{1\le i,j\le n^k} \{ \bigvee_{\substack{a_1,a_2,\cdots,a_6 \\ \text{是合法窗口}}} [x_{i,j-1,a_1} \wedge \cdots \wedge x_{i+1,j+1,a_6}] \}$$

$O(n^{2k})$

- N的一个转移函数规则对应常数个合法窗口
- 与N的转移函数无关的合法窗口有常数个

$A \le_p SAT$, SAT是NPC

$O(n^{2k})$ $$\phi_{accept} = \bigvee_{1 \le i,j \le n^k} x_{i,j,q_{accept}}$$

$O(n^{2k})$ $$\phi_{cell} = \bigwedge_{1 \le i,j \le n^k} \{[\bigvee_{s} x_{i,j,s}] \wedge [\bigwedge_{s \ne t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}})]\}$$

$O(n^k)$ $$\phi_{start} = x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \cdots \wedge x_{1,n^k,\#}$$

$O(n^{2k})$ $$\phi_{move} = \bigwedge_{1 \le i,j \le n^k} \{ \bigvee_{\substack{a_1,a_2,\cdots,a_6 \\ \text{是合法窗口}}} [x_{i,j-1,a_1} \wedge \cdots \wedge x_{i+1,j+1,a_6}]\}$$

证明：

(1) $f(w) = <\phi> = < \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accept} >$

(2) $w \in A \Leftrightarrow <\phi> \in SAT$,

(3) 令$|w| = n$, 则 $|<\phi>| = O(n^{2k})$

所以$A \le_p SAT$, SAT是NPC。

推论:3SAT是NP完全的

只需将前面的ϕ改造为3cnf公式.

$$\phi = \phi_{\text{cell}} \wedge \phi_{\text{start}} \wedge \phi_{\text{move}} \wedge \phi_{\text{accept}}.$$

$$\phi_{\text{start}} = x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \cdots \wedge x_{1,n^k,\#}$$

cnf

$$\phi_{\text{accept}} = \bigvee_{1\le i,j\le n^k} x_{i,j,q_{\text{accept}}}$$

cnf：一个子句

$$\phi_{\text{cell}} = \bigwedge_{1\le i,j\le n^k} \{[\bigvee_{s} x_{i,j,s}] \wedge [\bigwedge_{s\ne t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}})]\}$$

cnf

$$\phi_{\text{move}} = \bigwedge_{1\le i,j\le n^k} \{ \bigvee_{\substack{a_1,a_2,\cdots,a_6 \\ \text{是合法窗口}}} [x_{i,j-1,a_1} \wedge \cdots \wedge x_{i+1,j+1,a_6}]\}$$

不是cnf

ϕ_{move}的改造

$$\phi_{\text{move}} = \bigwedge_{1\le i,j\le n^k} \{ \bigvee_{\substack{a_1,a_2,\cdots,a_6 \\ \text{是合法窗口}}} [x_{i,j-1,a_1} \wedge \cdots \wedge x_{i+1,j+1,a_6}] \}$$

分配律 $(a\wedge b) \vee c = (a\vee c) \wedge (b\vee c)$

例如：$(a\wedge b)\vee(c\wedge d)\vee(e\wedge f) = (a\vee c\vee e)\wedge(a\vee c\vee f)\wedge\ldots$

长度由2×3变为3×2^3.

设合法窗口有M个, 则ϕ_{move}原长度是$6Mn^{2k}$,

改造为cnf范式后, ϕ_{move}长度是$M\times6^M\times n^{2k}$.

推论:3SAT是NP完全的

- **缩短子句长度为3:**

给定赋值T $a_1 \vee a_2 \vee \ldots \vee a_k = 1 \quad \Leftrightarrow$

$\Leftrightarrow \exists$z赋值, 在T下 $(a_1 \vee a_2 \vee \ldots \vee a_{k\text{-}2} \vee \mathrm{z}) \wedge (\neg \mathrm{z} \vee a_{k\text{-}1} \vee a_k) = 1$

1个k-文字子句 变为 k-2个3-文字子句

例如: $a_1 \vee a_2 \vee a_3 \vee a_4 = 1 \Leftrightarrow (a_1 \vee a_2 \vee \mathrm{z}) \wedge (\neg \mathrm{z} \vee a_3 \vee a_4) = 1$

改造为cnf范式后, $|\phi_{move}|$: $M \times 6^M \times n^{2k}$.

推论:3SAT是NP完全的

$$\phi_{\text{start}} = x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \cdots \wedge x_{1,n^k,\#}$$

$$\phi_{\text{accept}} = \bigvee_{1\le i,j\le n^k} x_{i,j,q_{\text{accept}}}$$

$$\phi_{\text{cell}} = \bigwedge_{1\le i,j\le n^k} \{[\bigvee_{s} x_{i,j,s}] \wedge [\bigwedge_{s\ne t}(\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}})]\}$$

$$\phi_{\text{move}} = \bigwedge_{1\le i,j\le n^k} \{ \bigvee_{\substack{a_1,a_2,\cdots,a_6 \\ \text{是合法窗口}}} [x_{i,j-1,a_1} \wedge \cdots \wedge x_{i+1,j+1,a_6}]\}$$

1个k-文字子句 变为 k-2个3-文字子句

- $|\phi_{\text{accept}}|$: $n^{2k} \to 3(n^{2k}-2)$.
- $|\phi_{\text{cell}}|$: $(|S|+|S|^2)n^{2k} \to (3(|S|-2)+|S|^2)n^{2k}$.
- $|\phi_{\text{move}}|$: $3\times(M-2)\times6^M\times n^{2k}$.

ϕ的长度是多项式的，SAT$\le_p$ 3SAT, 所以3SAT是NPC的.

改造为cnf范式后, $|\phi_{\text{move}}|$: $M\times6^M\times n^{2k}$.

小结

- **Cook-Levin定理**: 对任意$A \in NP$都有$A \le_p SAT$.
- 定义:语言B称为NP完全的(NP-complete), 若它满足:

  1) $B \in NP$;

  2) $\forall A \in NP$, 都有$A \le_p B$.
- **定理1(归约引理)**: 若$A \le_p B$ 且 $B \in P$ 则 $A \in P$.
- **定理2**: 若B是NPC的, 且$B \in P$, 则$P = NP$.
- 推论: 若$SAT \in P$, 则 $NP = P$.
- **定理3**: 若B是NPC的, $B \le_p C$, 且$C \in NP$, 则C是NPC.
- SAT是NPC 且$SAT \le_p 3SAT \Rightarrow$ 3SAT是NPC
- 3SAT是NPC 且 $3SAT \le_p CLIQUE \Rightarrow$ CLIQUE是NPC

第三章 计算复杂性

- 1. 时间复杂性
- 2. 不同模型的复杂性关系
- 3. P类与NP类
- 4. NP完全性及NP完全问题
  - NP完全性的定义
  - SAT是NP完全问题
  - 一些NP完全问题

一些NP完全问题

SAT

↓

3SAT

HP(哈密顿路径)

↓

UHP(无向图哈密顿路径)

↓

UHC(无向图哈密顿圈)

↓

TSP(旅行售货员问题)

KS

(子集和, 01背包)

CLIQUE

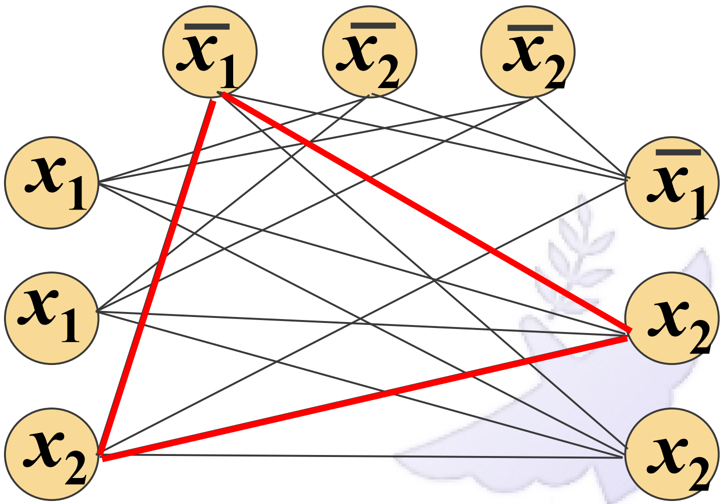
HP是NPC

- HP={ <G,s,t> | G是有向图, 有从s到t的哈密顿路径 }
- 证明思路：$3SAT \leq_P HP$
- 构造 $f(\phi)$ = <G,s,t> 使得 ϕ可满足⇔ G有从s到t的HP
- 任取3cnf公式$\phi = (a_1 \vee b_1 \vee d_1) \wedge \ldots \wedge (a_k \vee b_k \vee d_k)$,
  - 其中a, b, c为文字x_i或者$\neg x_i$。
  - n个变量$x_1, \ldots, x_n$,
  - k个子句$c_1=(a_1 \vee b_1 \vee d_1), \ldots, c_k=(a_k \vee b_k \vee d_k)$,

HP是NPC($3SAT \le_p HP$)

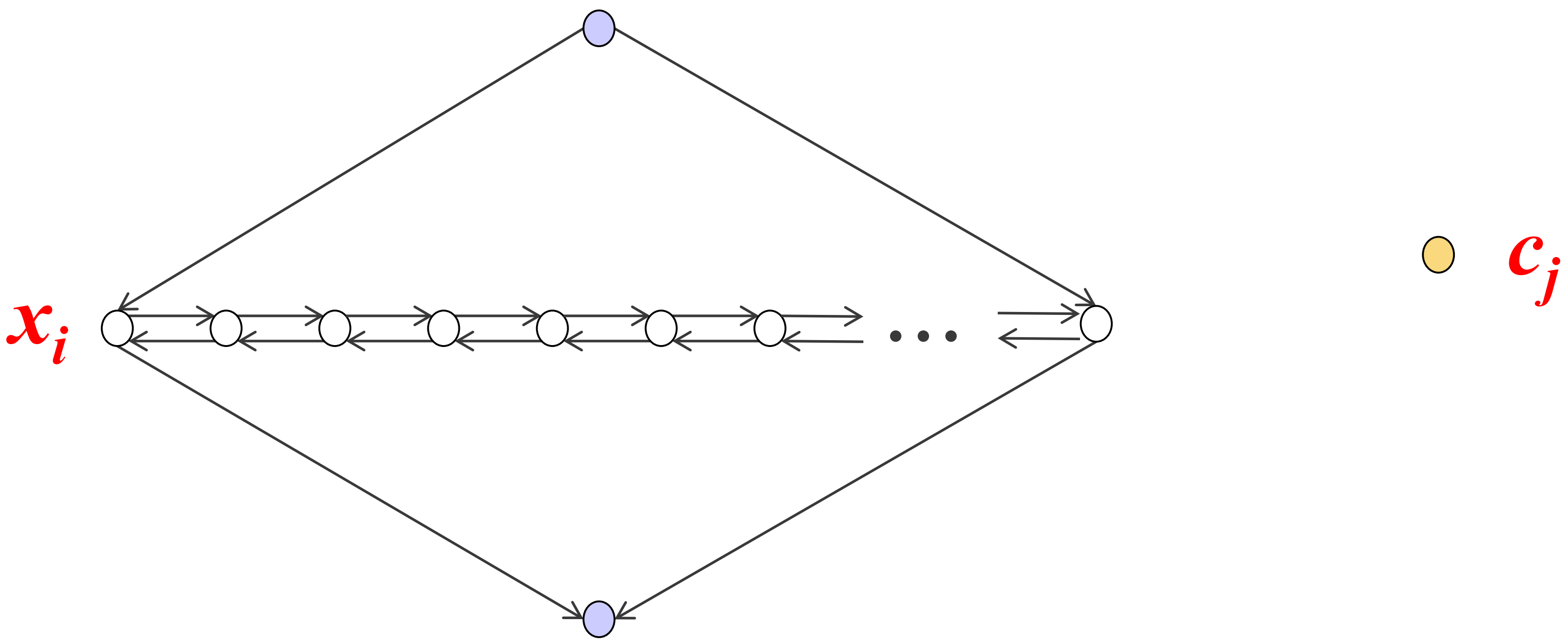
- 3cnf公式$\phi = (a_1 \vee b_1 \vee d_1) \wedge \ldots \wedge (a_k \vee b_k \vee d_k)$
- 构造 $f(\phi) = $ <G,s,t> 使得 ϕ可满足$\Leftrightarrow$ G有从s到t的HP
- 一般由3cnf公式构造的图有
  - 变量构件
  - 子句构件
  - 联接构件

  如右图3SAT到CLIQUE归约



变量构件和子句构件

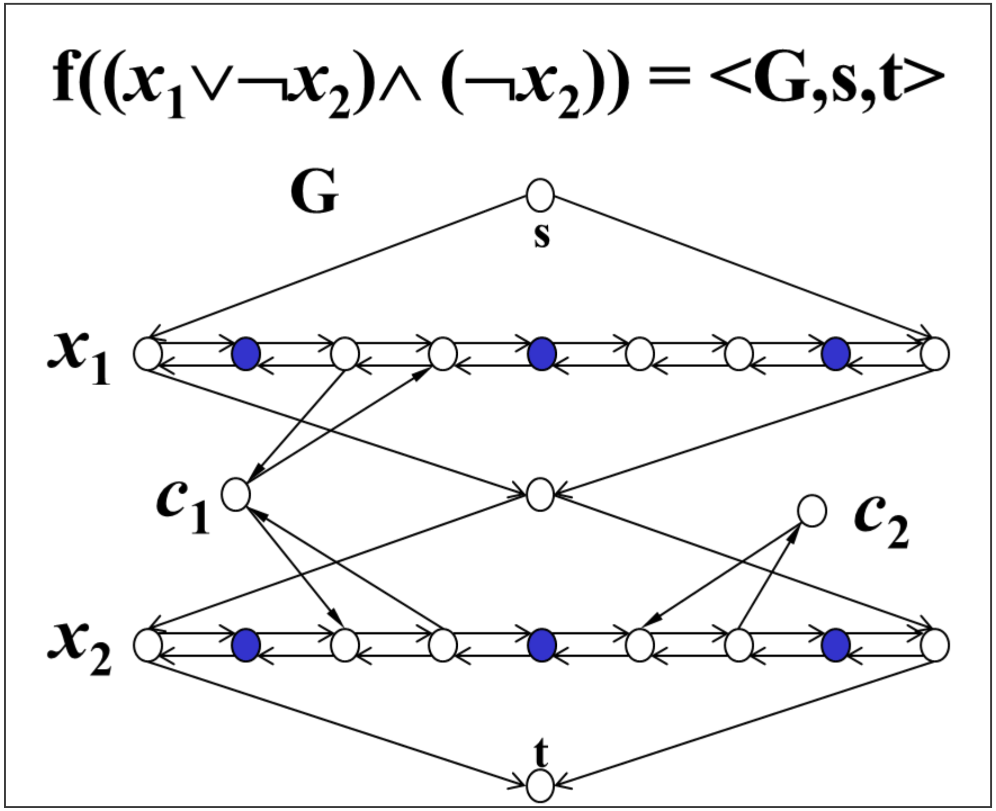
$f((x_1 \vee \neg x_2) \wedge (\neg x_2)) = <G, s, t>$



变量x_i表示为一个钻石结构　　　　子句c_j表示为一个节点

联接构件

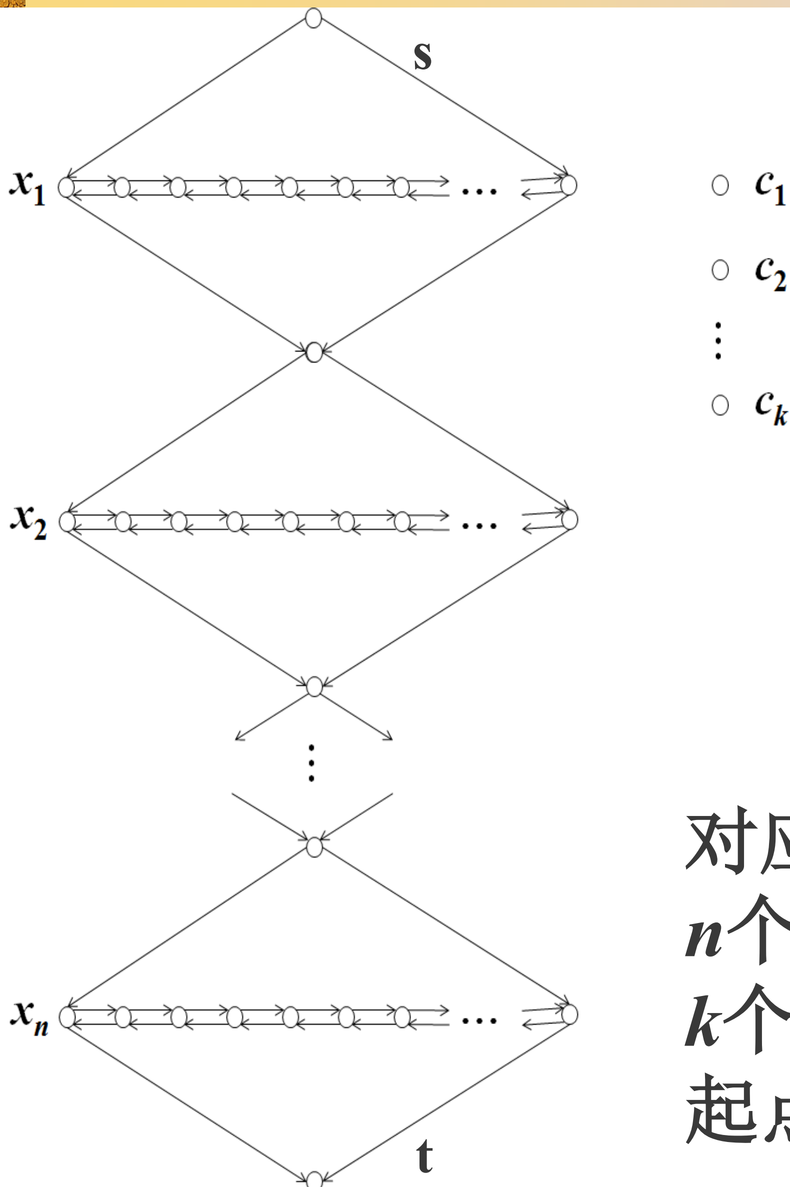
$f((x_1 \vee \neg x_2) \wedge (\neg x_2)) = \langle G, s, t \rangle$



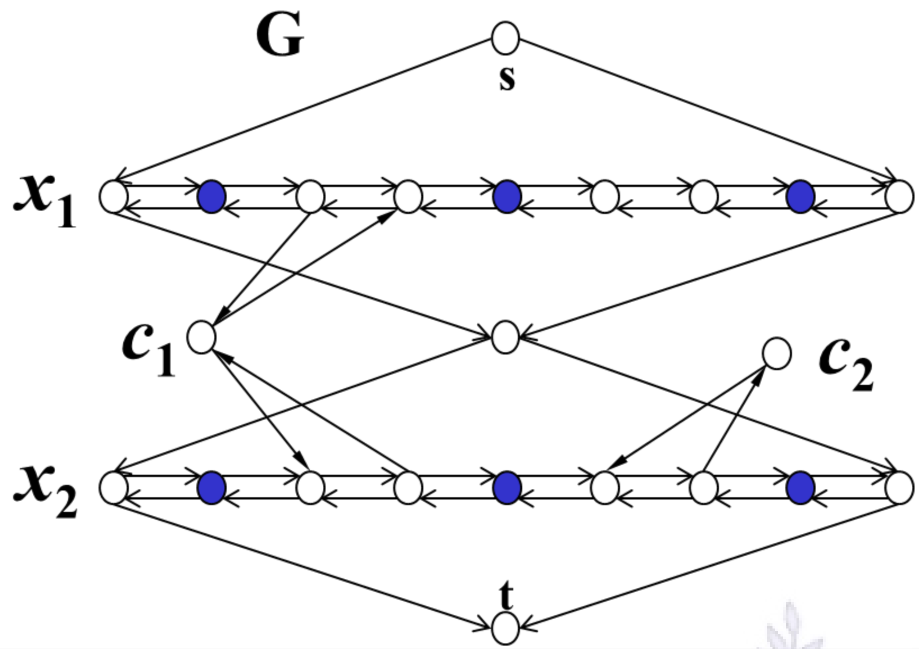
左-右式路径：x_i=1

右-左式路径：x_i=0

图G的总体结构



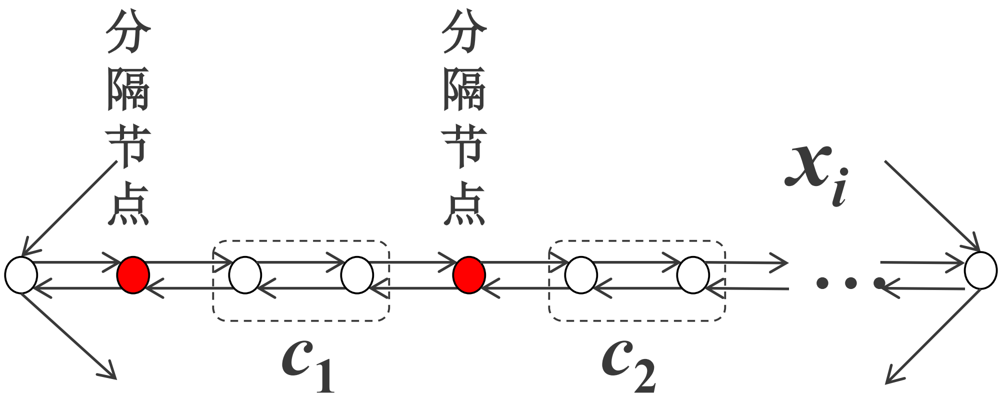
$$f((x_1 \vee \neg x_2) \wedge (\neg x_2)) = \langle G,s,t \rangle$$

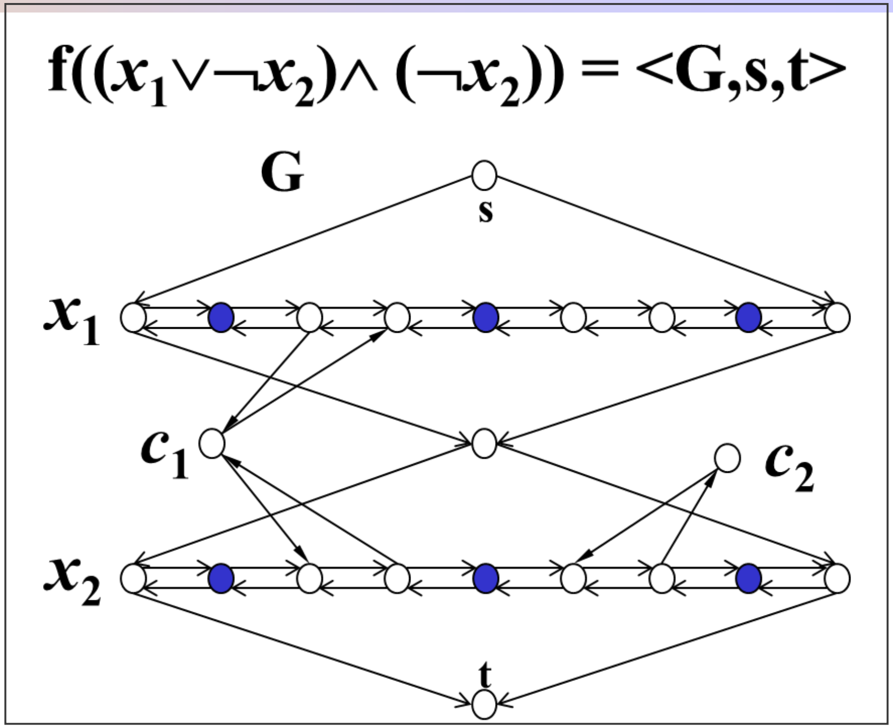


对应
n个变量$x_1, \ldots, x_n$,
k个子句$c_1, \ldots, c_k$,
起点s, 终点t

这个图有哪些哈密顿路径?

钻石构件中的水平节点



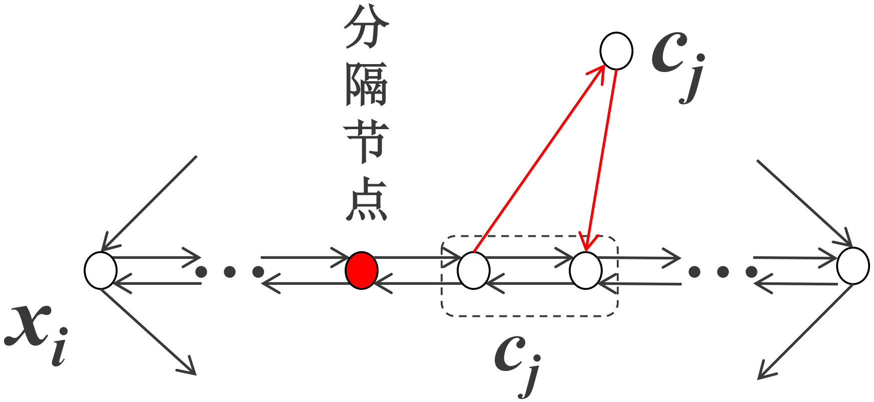


n: 变量个数, k:子句个数

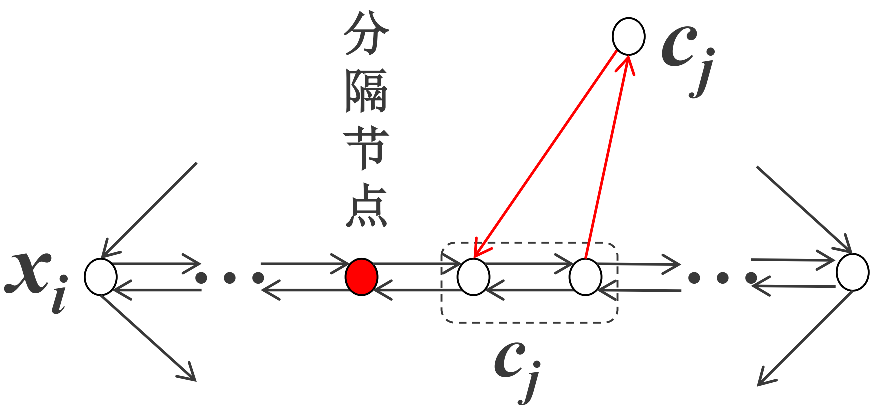
水平行除两端的两个节点外有$3k+1$个节点

- 每个子句对应一对节点(共$2k$个)
- 用分隔节点隔开($k+1$个)

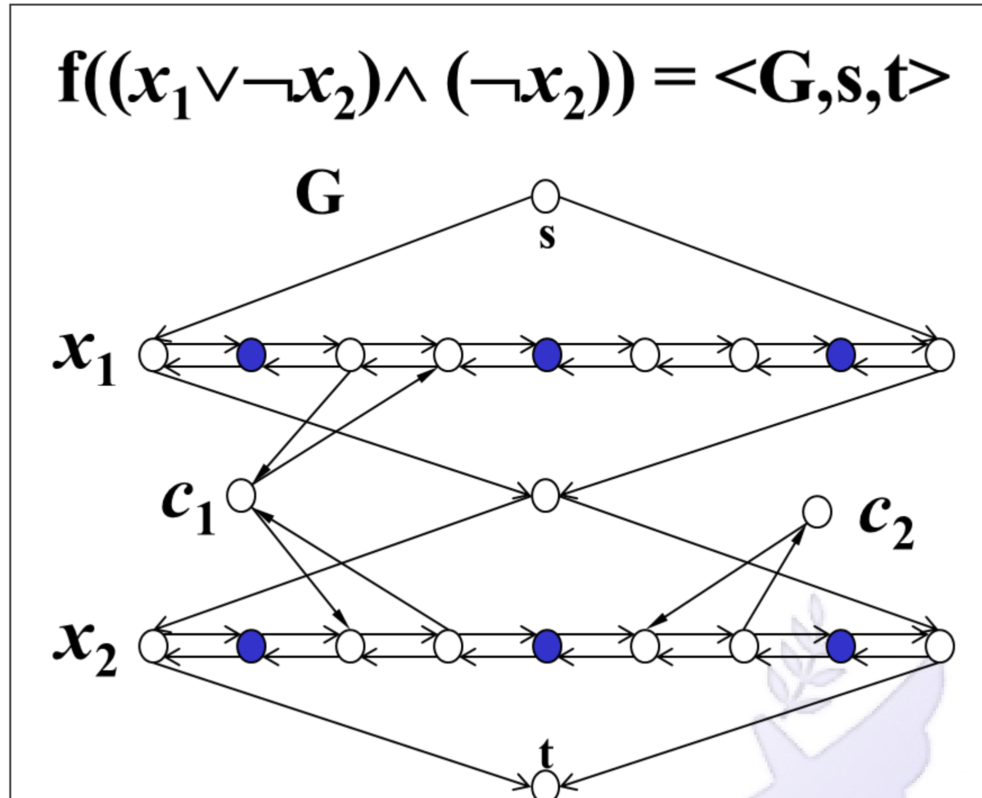
变量与子句构件的连接



当子句c_j含有文字x_i时添加的边
左-右式路径可以通过，c_j满足



当子句c_j含有文字$\neg x_i$时添加的边
右-左式路径可以通过，c_j满足



ϕ可满足赋值对应<G,s,t>正规路径

- ϕ可满足 $\Rightarrow$ G有如下从s到t哈密顿路径

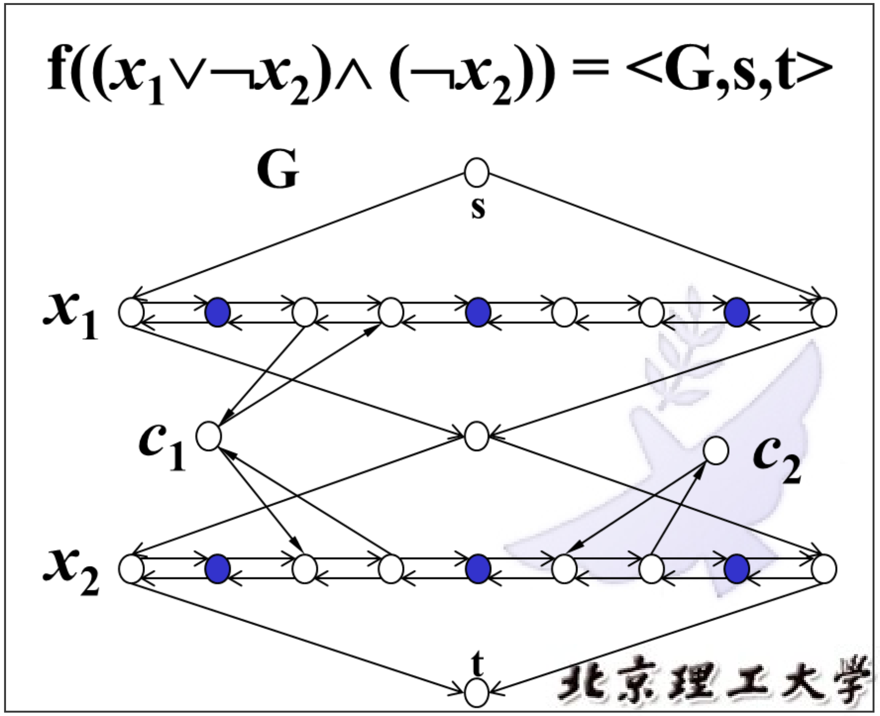
  从上至下

  赋值1的变量左-右式通过钻石

  赋值0的变量右-左式通过钻石

  c_j选一真文字经过一次

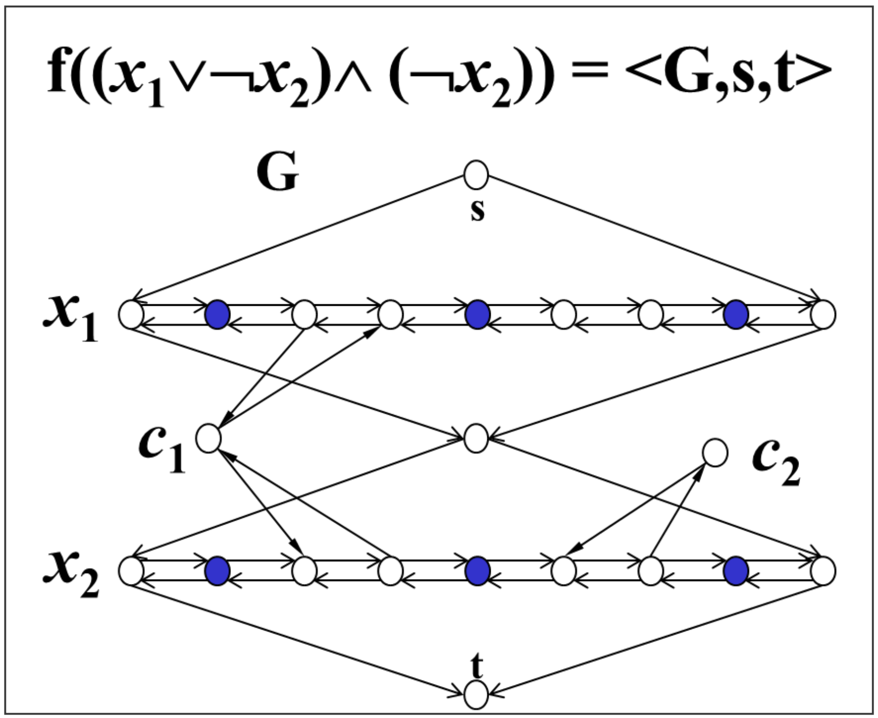
  称这种路径为正规路径



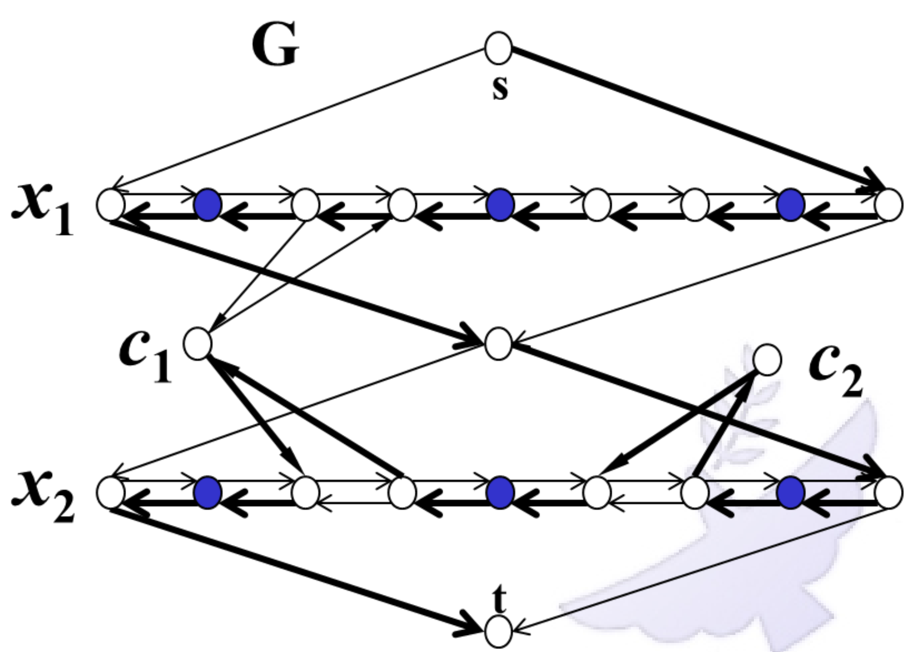
ϕ可满足赋值对应<G,s,t>正规路径

ϕ可满足 $\Rightarrow$ G有如下从s到t哈密顿路径

- 正规路径：从上至下 • 赋值1的变量左-右式通过钻石
- 赋值0的变量右-左式通过钻石 • c_j选一真文字经过一次



$x_1=0,\ x_2=0,$



ϕ可满足赋值对应<G,s,t>正规路径

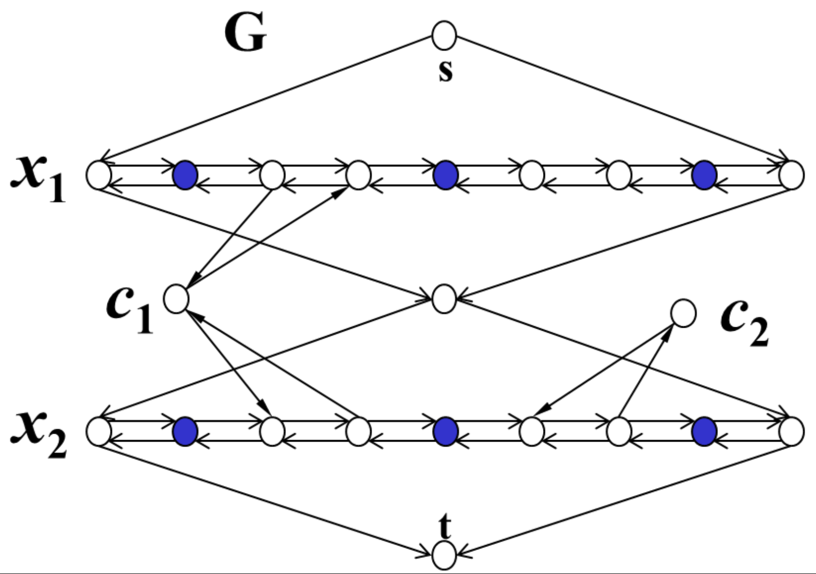
ϕ可满足 $\Rightarrow$ G有如下从s到t哈密顿路径

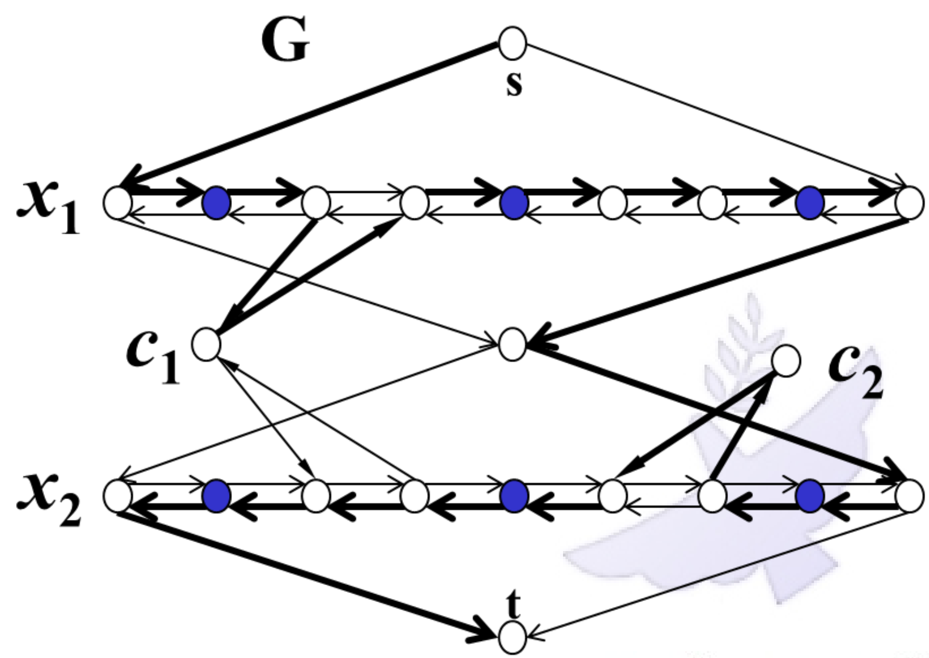
• 从上至下 • 赋值1的变量左-右式通过钻石 • 赋值0的变量右-左式通过钻石

• c_j选一真文字经过一次 • 称这种路径为正规路径

$x_1=1,\ x_2=0,$

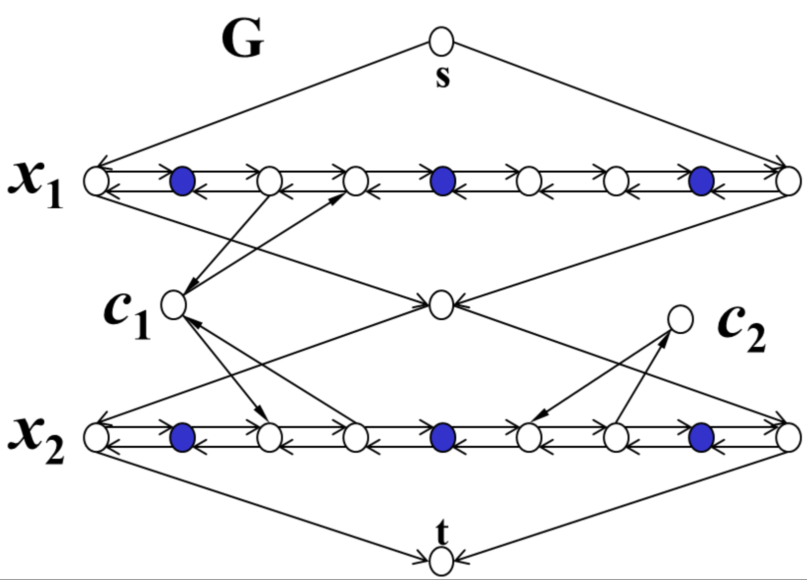
$\mathrm{f}((x_1\vee\neg x_2)\wedge(\neg x_2)) = \langle G,s,t\rangle$

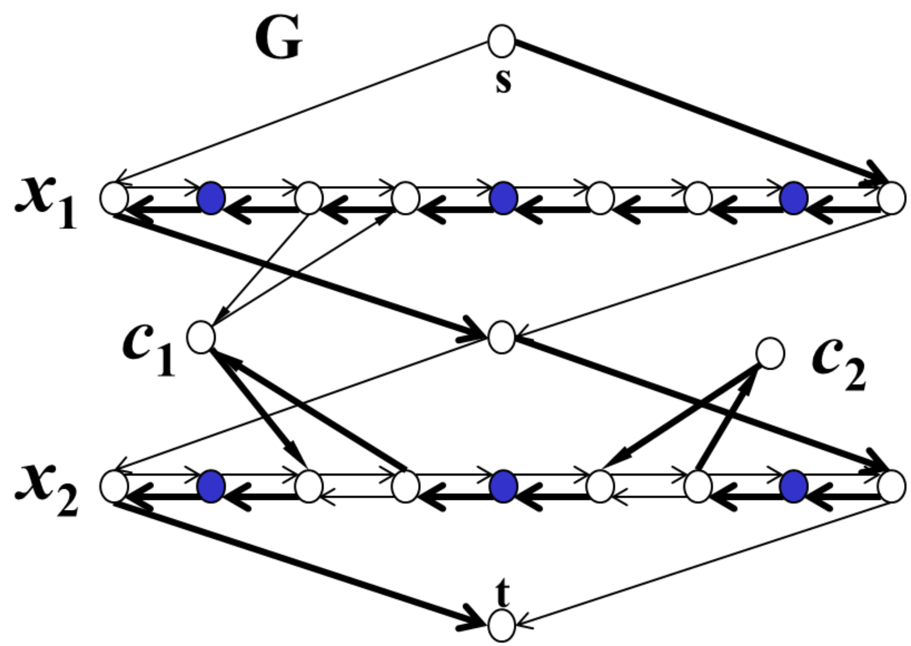




<G,s,t>正规路径对应ϕ可满足赋值

$f((x_1 \vee \neg x_2) \wedge (\neg x_2)) = <G,s,t>$





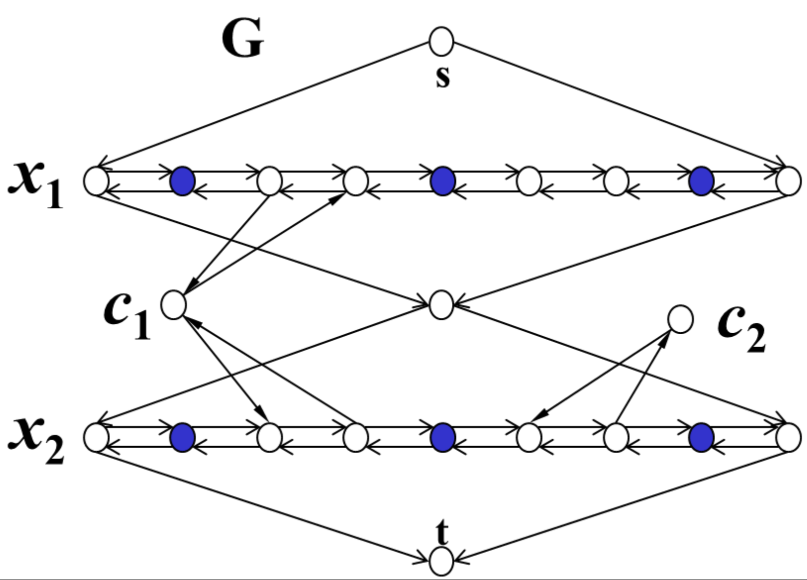
- 由左-右 或 右-左 式穿过钻石可确定变量赋值,
- c_j被穿过说明在对应变量赋值下$c_j=1$,

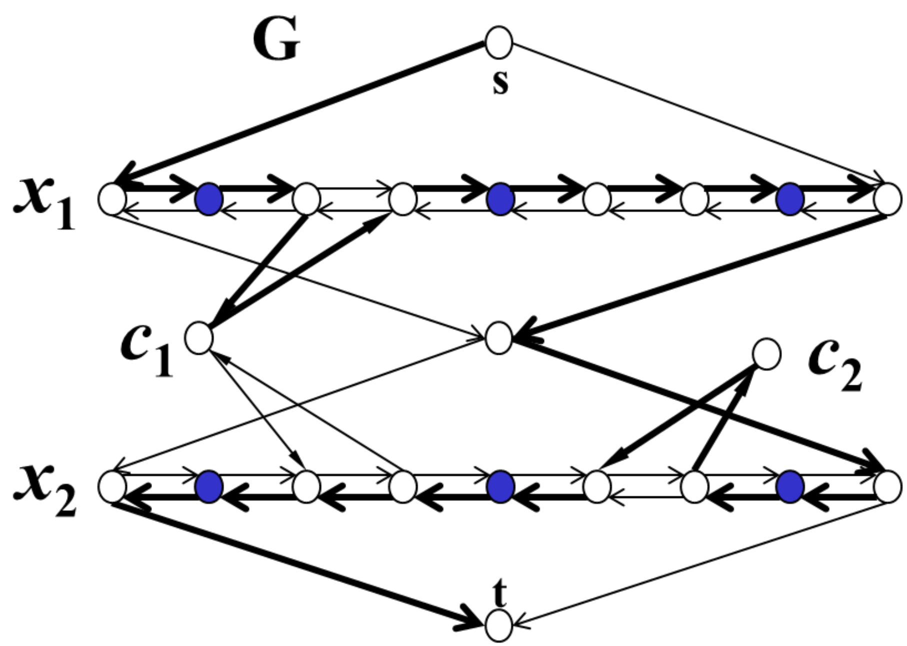
则公式ϕ可满足

正规路径对应$x_1=0$, $x_2=0$.

<G,s,t>正规路径对应ϕ可满足赋值

$$f((x_1 \vee \neg x_2) \wedge (\neg x_2)) = \langle G,s,t \rangle$$





- 由左-右 或 右-左 式穿过钻石可确定变量赋值,
- c_j被穿过说明在对应变量赋值下c_j=1,

则公式ϕ可满足

正规路径对应x_1=1, x_2=0.

所以**3SAT$\leq_p$HP**，**HP是NPC**

一些NP完全问题

SAT

3SAT

HP(哈密顿路径)

KS

(子集和, 01背包)

CLIQUE

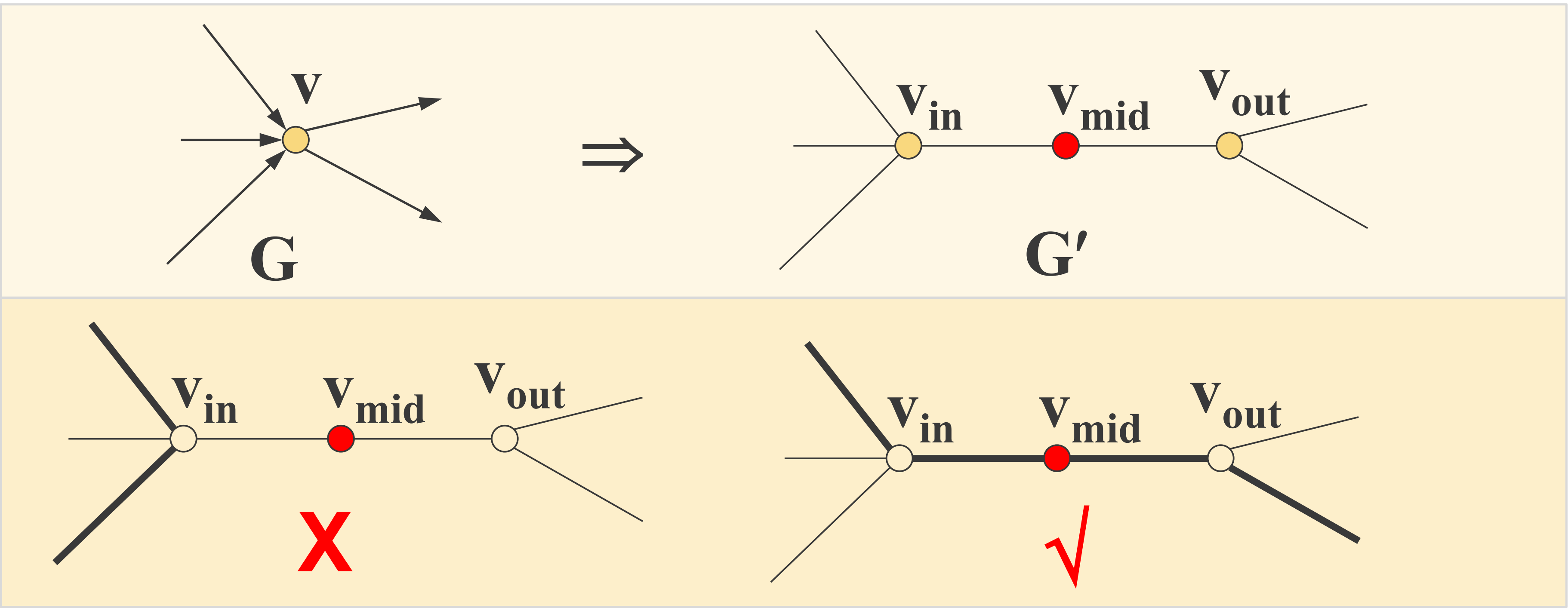
UHP(无向图哈密顿路径)

UHC(无向图哈密顿圈)

TSP(旅行售货员问题)

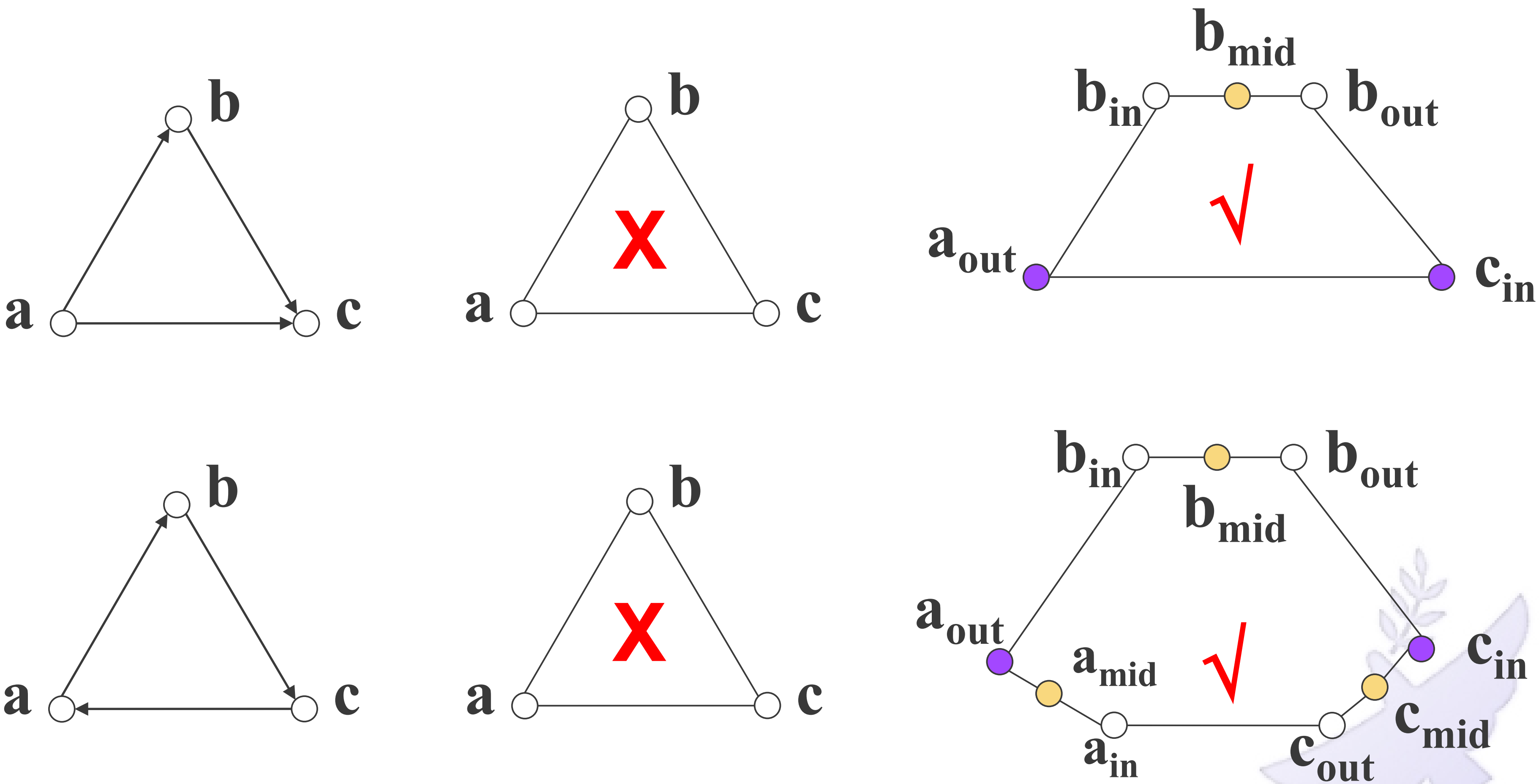
无向图哈密顿路径问题UHP是NPC

- HP = {<G,s,t> | G是有从s到t哈密顿路径的有向图 }
- UHP = {<G,s,t>| G是有从s到t哈密顿路径的无向图 }
- 证明: HP$\leq_p$UHP, 映射归约如下 f(<G,s,t>) = <G′,s_{out},t_{in}>
- s对应s_{out}, t对应t_{in}, 其它每个节点v对应v_{in},v_{mid},v_{out}



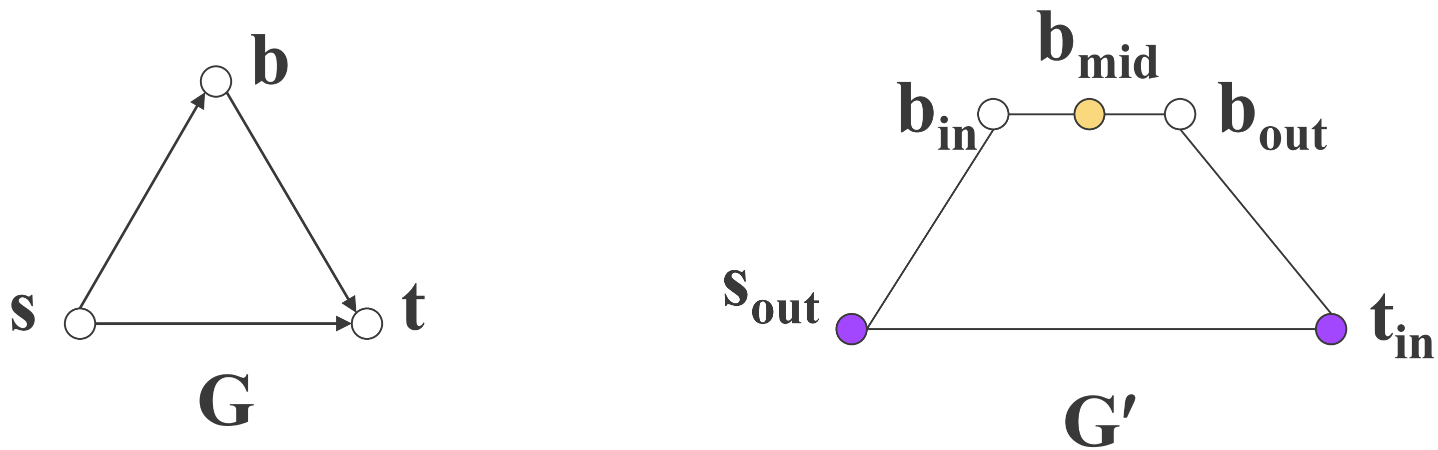
$HP \leq_P UHP$

映射归约 $\langle G,a,c\rangle \rightarrow \langle G',a_{out},c_{in}\rangle$ 举例



HP$\leq_P$UHP

- $f(<G,s,t>) = <G',s_{out},t_{in}>$
- G′有从s_{out}到t_{in}的H通路 $\Rightarrow$ G中有从s到t的H通路
- G中有从s到t的H通路 $\Rightarrow$ G′有从s_{out}到t_{in}的H通路



UHC是NP完全的

- UHC = {<G>| G是有哈密顿回路的无向图 }
- 证明：(1) UHC∈NP

  构造多项式时间内判定UHC的非确定图灵机N:

  N="对于输入<G>, G是无向图,

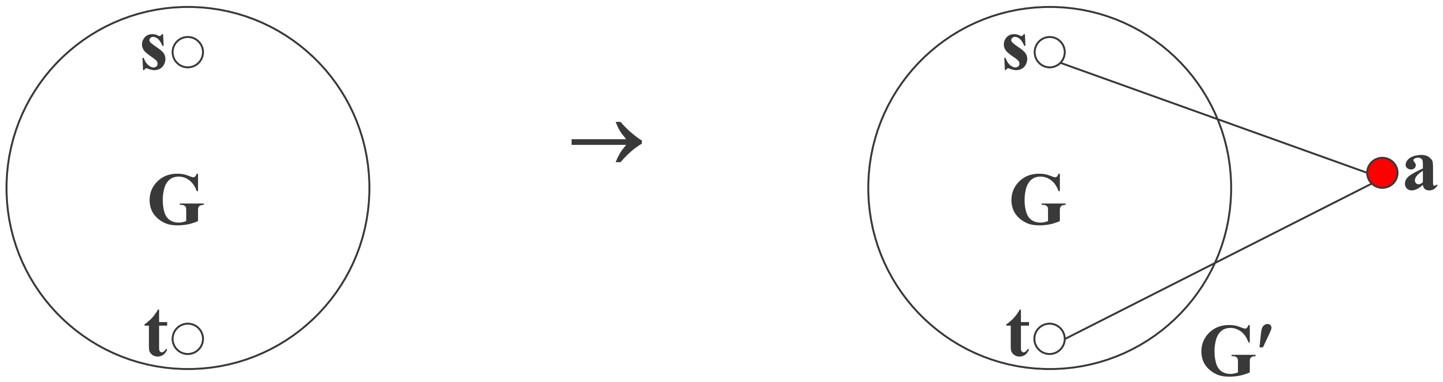
  1)非确定地选择G所有节点的一个排列$v_1,v_2,\ldots, v_n$.

  2)若$(v_1,v_2,\ldots,v_n,v_1)$是G的路径, 则接受; 否则拒绝."

  第2)步可在多项式时间内完成。
- (2) $UHP\leq_p UHC$
- 由(1), (2)和UHP是NP完全的, 得UHC是NP完全的

UHP$\leq_p$UHC

- UHP = {<G,s,t>| G是有从s到t哈密顿路径的无向图 }
- UHC = {<G>| G是有哈密顿回路的无向图 }
- 证明: 映射归约如下 <G,s,t> → <G′>



<G,s,t> → <G′> 增加一个节点两条边, 多项式时间

G有从s到t的哈密顿路径 ⇔ G′有哈密顿回路

TSP是NP完全的

- TSP={<G,s,w,b> | 无向图G有 从s出发回到s, 权和≤b 的哈密顿回路 } //将TSP修改成判定性问题
- (1) TSP∈NP. 构造多项式时间内判定TSP的NTM:

  N="对于输入<G,s,w,b>, G是无向图,s是节点, w是权, b≥0,

  1)非确定地选择G所有节点的排列$s,v_2,\ldots, v_n$.

  2)若$(s,v_2,\ldots,v_n,s)$是G的路径, 且路径权和≤b, 则接受;

  3)否则拒绝."
- (2) $UHC \leq_P TSP$
- 由(1), (2)和UHC是NP完全的, 得TSP是NP完全的

UHC $\le_p$ TSP

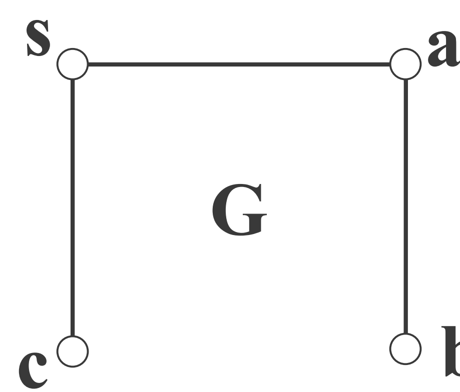
- UHC = { <G> | G是有哈密顿回路(HC)的无向图 }
- TSP = { <G,s,w,b> | G有s出发费用≤b的哈密顿回路 }
- *f*(<G>) = <G′,s,w,n>
- 设G=(V, E), V={$v_1,\ldots,v_n$},

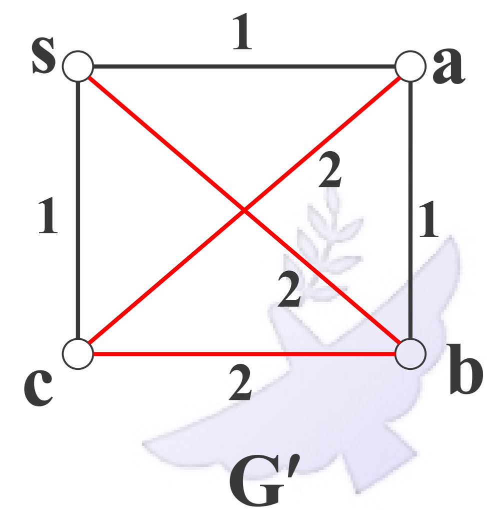
  令G′=(V, V×V), s∈V

  定义权w:

$$w[v_i,v_j]=\begin{cases}0 & 若 i=j \\ 1 & 若 (v_i,v_j)\in E \\ 2 & 其它\end{cases}$$

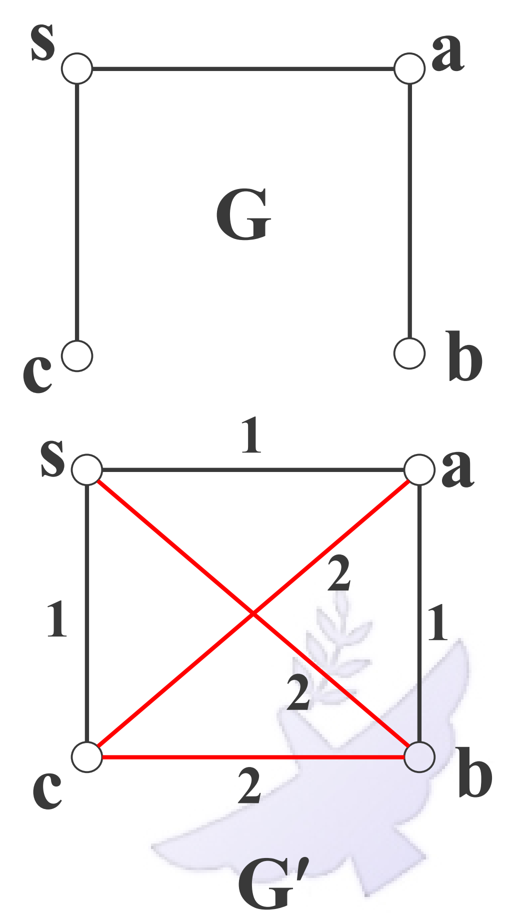
- *f*(<G>)边数=n², 多项式时间可计算





UHC $\le_p$ TSP

- $f(<G>) = <G,s,w,b>$
- 证明：G有HC ⇔ G′有s出发费用≤n的HC
- 1）G′有s出发费用≤n的HC
  - ⇒ 该回路上的边都在G中
  - ⇒ G有HC
- 2）G有HC
  - ⇒ G′有s出发费用≤n的HC



一些NP完全问题

SAT

↓

3SAT

↓

HP(哈密顿路径)　　KS (子集和, 01背包)　　CLIQUE

↓

UHP(无向图哈密顿路径)

↓

UHC(无向图哈密顿圈)

↓

TSP(旅行售货员问题)

0-1背包(knapsack)问题是NPC

- KS = { < S, t > | t 等于S中一些数的和 } //改为判定性问题
- KS是NPC的。证明：
  - 1) KS∈NP
  - 2) 3SAT $\leq_p$ KS
- 设ϕ是3cnf公式, 构造 $f(<\phi>) = < S, t >$

  设ϕ有n个变量$x_1,\ldots,x_n$, k个子句$c_1,\ldots,c_k$,

  构造KS的一个实例<S, t>，使得

  该实例包含加起来等于目标t的子集T当且仅当ϕ可满足。

[S]中称为子集和问题.

3SAT $\leq_P$ KS

- 设ϕ是3cnf公式, 构造 $f(<\phi>) = < S, t >$

  设ϕ有n个变量$x_1,\ldots,x_n$, k个子句$c_1,\ldots,c_k$, 构造

  数集 $S = \{ y_1,\ldots,y_n, z_1,\ldots,z_n, g_1,\ldots,g_k, h_1,\ldots,h_k\}$,
  - y_i和z_i的高n位分别表示变量x_i和$\neg x_i$。
  - g_j和h_j的低k位对应子句c_j,

  ¶ S中所有数十进制表示:
  - 根据ϕ构造数的高n位和低k位
  - S中数每位是0或1;

  ¶ t的低k位都是3, 高n位都是1.

	高n位	低k位
	x_1 x_2 x_3 … x_n	c_1 c_2 c_3 … c_k
y_1		
…		
z_1		
…		
g_1		
…		
h_1		
t	1 1 1 1 … 1	3 3 3 … 3

$y_1,\cdots,y_n,z_1,\cdots,z_n,g_1,\cdots,g_k,h_1,\cdots,h_k,t$的构造

- 所有数十进制表示, 根据ϕ构造每个数的高n位和低k位
- S中数每位是0或1; t的低k位都是3, 高n位都是1.
- 构造见下表. 总位数≤$(n+k+1)^2$.

		$x_1 \quad x_2 \quad \cdots \quad x_n$	$c_1 \quad c_2 \quad \cdots \quad c_k$
S	$y_1 \cdots y_n$	$yx_{ij}=\begin{cases}1 & i=j\\0 & else\end{cases}$	$yc_{ij}=\begin{cases}1 & 若c_j中有x_i\\0 & else\end{cases}$
	$z_1 \cdots z_n$	$zx_{ij}=\begin{cases}1 & i=j\\0 & else\end{cases}$	$zc_{ij}=\begin{cases}1 & 若c_j中有\neg x_i\\0 & else\end{cases}$
	$g_1 \cdots g_k$	0	$gc_{ij}=\begin{cases}1 & i=j\\0 & else\end{cases}$
	$h_1 \cdots h_k$	0	$hc_{ij}=\begin{cases}1 & i=j\\0 & else\end{cases}$
	t	1 1 … 1	3 3 … 3

- yx区: 单位阵
- zx区: 单位阵
- gc区: 单位阵
- hc区: 单位阵
- yz行c_j列≤3个1

归约举例1

$$f(\langle (x_1 \vee \neg x_2) \wedge (\neg x_2) \rangle) = \langle \{1010,100,1000,111,10,1,10,1\},1133 \rangle$$

	$x_1 \quad x_2 \quad \dots \quad x_n$	$c_1 \quad c_2 \quad \dots \quad c_k$
y_1 … y_n	$yx_{ij} = \begin{cases} 1 & i=j \\ 0 & else \end{cases}$	$yc_{ij} = \begin{cases} 1 & 若c_j中有x_i \\ 0 & else \end{cases}$
z_1 … z_n	$zx_{ij} = \begin{cases} 1 & i=j \\ 0 & else \end{cases}$	$zc_{ij} = \begin{cases} 1 & 若c_j中有\neg x_i \\ 0 & else \end{cases}$
g_1 … g_k	0	$gc_{ij} = \begin{cases} 1 & i=j \\ 0 & else \end{cases}$
h_1 … h_k	0	$hc_{ij} = \begin{cases} 1 & i=j \\ 0 & else \end{cases}$
t	$1 \quad 1 \quad \dots \quad 1$	$3 \quad 3 \quad \dots \quad 3$

	x_1	x_2	c_1	c_2
y_1	1 单位阵	0	1	0
y_2	0	1	0	0
z_1	1 单位阵	0	0	0
z_2	0	1	1	1
g_1	0	0	1 单位阵	0
g_2	0	0	0	1
h_1	0	0	1 单位阵	0
h_2	0	0	0	1
t	1	1	3	3

y_1行c_1列是1, 因为c_1含x_1; y_1行c_2列是0, 因为c_2不含x_1;
y_2行c_1列是0, 因为c_1不含x_2; y_2行c_2列是0, 因为c_2不含x_2;
z_1行c_1列是0, 因为c_1不含$\neg x_1$; z_1行c_2列是0, 因为c_2不含$\neg x_1$;
z_2行c_1列是1, 因为c_1含$\neg x_2$; z_2行c_2列是1, 因为c_2含$\neg x_2$.

归约举例2

$$f(\langle(x_1\vee\neg x_2)\rangle) = \langle\{101,10,100,11,1,1\},113\rangle$$

	x_1 x_2 … x_n	c_1 c_2 … c_k
y_1 … y_n	$yx_{ij} = \begin{cases}1 & i=j\\0 & else\end{cases}$	$yc_{ij} = \begin{cases}1 & \text{若}c_j\text{中有}x_i\\0 & else\end{cases}$
z_1 … z_n	$zx_{ij} = \begin{cases}1 & i=j\\0 & else\end{cases}$	$zc_{ij} = \begin{cases}1 & \text{若}c_j\text{中有}\neg x_i\\0 & else\end{cases}$
g_1 … g_k	0	$gc_{ij} = \begin{cases}1 & i=j\\0 & else\end{cases}$
h_1 … h_k	0	$hc_{ij} = \begin{cases}1 & i=j\\0 & else\end{cases}$
t	1 1 … 1	3 3 … 3

	x_1	x_2	c_1
y_1	1	0	1
y_2	0	1	0
z_1	1	0	0
z_2	0	1	1
g_1	0	0	1
h_1	0	0	1
t	1	1	3

ϕ可满足$\Leftrightarrow$ f(<ϕ>)$\in$KS(knapsack)

f (<$(x_1 \vee \neg x_2) \wedge (\neg x_2)$>) = <{1010,100,1000,111,10,1,10,1},1133 >

		x_1	x_2	c_1	c_2
	y_1	1	0	1	0
	y_2	0	1	0	0
S	z_1	1	0	0	0
	z_2	0	1	1	1
	g_1	0	0	1	0
	g_2	0	0	0	1
	h_1	0	0	1	0
	h_2	0	0	0	1
	t	1	1	3	3

- 取赋值x_1=0, x_2=0, (可满足)
  对应选 z_1, z_2,
  添g_1, g_2, h_1, h_2, 得和t

ϕ可满足$\Leftrightarrow f(\langle\phi\rangle)\in$KS(knapsack)

$$f(\langle(x_1\vee\neg x_2)\wedge(\neg x_2)\rangle) = \langle\{1010,100,1000,111,10,1,10,1\},1133\rangle$$

S

	x_1	x_2	c_1	c_2
y_1	1	0	1	0
y_2	0	1	0	0
z_1	1	0	0	0
z_2	0	1	1	1
g_1	0	0	1	0
g_2	0	0	0	1
h_1	0	0	1	0
h_2	0	0	0	1
t	1	1	3	3

- 取赋值x_1=1, x_2=0, (可满足)

  对应选 y_1, z_2,

  添 g_1, g_2, h_2, 得和t

ϕ可满足$\Leftrightarrow$ f(<ϕ>)$\in$KS(knapsack)

f (<$(x_1 \vee \neg x_2) \wedge (\neg x_2)$>) = <{1010,100,1000,111,10,1,10,1},1133 >

S

	x_1	x_2	c_1	c_2
y_1	1	0	1	0
y_2	0	1	0	0
z_1	1	0	0	0
z_2	0	1	1	1
g_1	0	0	1	0
g_2	0	0	0	1
h_1	0	0	1	0
h_2	0	0	0	1
t	1	1	3	3

- 取赋值x_1=0, x_2=1, (不可满足) 对应选 z_1, y_2, 得不到 t
- 取赋值x_1=1, x_2=1, (不可满足) 对应选 y_1, y_2, 得不到 t

ϕ可满足$\Rightarrow$ f(<ϕ>)$\in$KS

f (<ϕ>) = <S, t >

<$(x_1\vee\neg x_2)\wedge(\neg x_2)$>

成真赋值$x_1=1, x_2=0$

S

	x_1	x_2	c_1	c_2
y_1	1	0	1	0
y_2	0	1	0	0
z_1	1	0	0	0
z_2	0	1	1	1
g_1	0	0	1	0
g_2	0	0	0	1
h_1	0	0	1	0
h_2	0	0	0	1
t	1	1	3	3

若ϕ有满足赋值(<ϕ>$\in$3SAT)

则 对每个x_i,

若$x_i=1$, 则选数y_i;

若$x_i=0$, 则选数z_i.

第x_i列的和是1.

对每个c_j,

已选数c_j列和t_j,$1\le t_j\le 3$

若=1, 则选g_j, h_j;

若=2, 则选g_j;

若=3, 则不用选

已选数第c_j列的和是3

即可选出子集T,其和 = t

即 f(<ϕ>)$\in$KS

ϕ可满足$\Leftarrow f(<\phi>)\in$KS

$<(x_1\vee\neg x_2)\wedge(\neg x_2)>$

成真赋值$x_1=1, x_2=0$

S

	x_1	x_2	c_1	c_2
y_1	1	0	1	0
y_2	0	1	0	0
z_1	1	0	0	0
z_2	0	1	1	1
g_1	0	0	1	0
g_2	0	0	0	1
h_1	0	0	1	0
h_2	0	0	0	1
t	1	1	3	3

若$f(<\phi>)\in$KS

即存在子集T, 其和 = t

则 子集中前n位，对第x_i列,

因为y_i, z_i只有1个

若有y_i, 则令$x_i=1$;

若有z_i, 则令$x_i=0$.

子集中后k位，对第c_j列,

因为第c_j列的和是3

gh行c_j列和$t_{ghj}\le 2$

yz行c_j列和t_{yzj}, $1\le t_{yzj}\le 3$

子句c_j在当前赋值下为1

即ϕ有满足赋值

3SAT $\leq_P$ KS

- 归约可以在多项式时间内完成
  - 表格大小=(2n+2k+1)(n+k)
  - 每一格的内容可以根据ϕ直接得到
  - 全部构造时间是O((n+k)²)

	$x_1 \quad x_2 \quad \dots \quad x_n$	$c_1 \quad c_2 \quad \dots \quad c_k$
y_1 … y_n	$yx_{ij}=\begin{cases}1 & i=j\\ 0 & else\end{cases}$	$yc_{ij}=\begin{cases}1 & \text{若}c_j\text{中有}x_i\\ 0 & else\end{cases}$
z_1 … z_n	$zx_{ij}=\begin{cases}1 & i=j\\ 0 & else\end{cases}$	$zc_{ij}=\begin{cases}1 & \text{若}c_j\text{中有}\neg x_i\\ 0 & else\end{cases}$
g_1 … g_k	0	$gc_{ij}=\begin{cases}1 & i=j\\ 0 & else\end{cases}$
h_1 … h_k	0	$hc_{ij}=\begin{cases}1 & i=j\\ 0 & else\end{cases}$
t	1 1 … 1	3 3 … 3

END

计算理论总结

- 计算模型
  - 有限自动机 非确定有限自动机 正则表达式
  - 正则语言 泵引理
  - 图灵机 图灵可判定语言 图灵可识别语言
- 可计算理论
  - 停机问题非图灵可判定,
  - 停机问题的补不是图灵可识别
- 计算复杂性
  - **P, NP, EXP, NP**完全

计算理论第7章作业

7.22 令HALF-CLIQUE = { <G> | G是无向图, 包含结点数至少为m/2的完全子图, m是G的结点数}。证明HALF-CLIQUE是NP完全的。

说明：书上的答案只是要点，考试时需要给出完整的答案。

证明：

(1) HALF-CLIQUE∈NP

构造如下非确定图灵机

N= "对于输入<G>, G是无向图,有**m**个顶点

(a) 非确定地产生一个**m/2**个顶点的子集

(b) 若这个子集中的任意两个顶点之间都有边相连，则接受；否则，拒绝"。

因为N的语言是**HALF-CLIQUE**，且N是在多项式时间运行，所以**HALF-CLIQUE∈NP**。

计算理论第7章作业

(2) 证明CLIQUE可以多项式时间映射归约到HALF-CLIQUE.
对任意<G,k>，其中G是一个无向图，k是一个正整数。构造函数f(<G,k>) = G'。
设G有m个顶点。按如下方式构造G'：
若k=m/2，则G=G'；
若k>m/2，则在G中增加2k-m个新顶点，这些新顶点都是孤立点，得到G'；
若k<m/2，则增加m-2k个新顶点，这些新顶点之间两两都有边相连，新顶点与G的所有顶点之间也都相连。
首先，f可在多项式时间内计算完成。
其次证明f是CLIQUE到HALF-CLIQUE的映射归约，即证明G有k团⇔G'(设有m'个顶点)有m'/2个顶点的团：
若G有k团，当k=m/2时，G'=G, m'=m，则G'也有k=m'/2团；当k>m/2时，m'=2k，G'中也有k=m'/2团；当k<m/2时，m'=2m-2k，G中的k团加上新添的m-2k个顶点形成m-k=m'/2团。
若G'有m'/2团，当k=m/2时，G'=G, m'=m，则G也有k=m'/2团；当k>m/2时，m'=2k，G中也有k=m'/2团；当k<m/2时，m'=2m-2k，G'中的m-k团至多有m-2k个新添顶点，去掉新添顶点至少还有k个顶点，所以G中有k团。

由(1)和(2)，HALF-CLIQUE是NP完全问题。