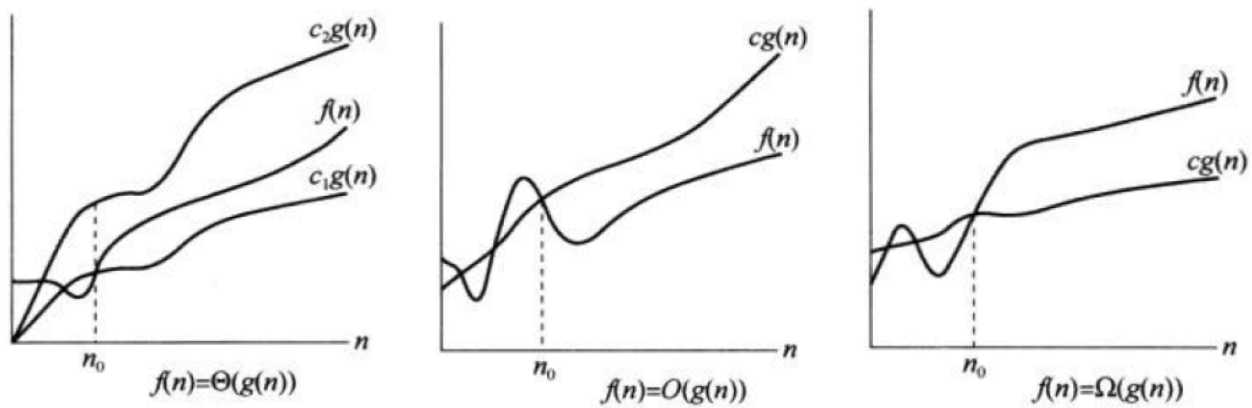
课程回顾

- ■插入排序分析
  - ➢循环不变式、算法执行时间
- ■渐近表示法
  - ➢Θ记号
  - ➢O记号
  - ➢Ω记号
  - ➢o记号
  - ➢ω记号



$$\lim_{n\to\infty}\frac{f(n)}{g(n)}=\begin{cases}0 & f(n)=o(g(n)),f(n)=O(g(n))\\ a,a>0 & f(n)=\Theta(g(n)),f(n)=O(g(n)),f(n)=\Omega(g(n))\\ \infty & f(n)=\omega(g(n)),f(n)=\Omega(g(n))\end{cases}$$

课程回顾 (续)

- 函数比较（传递性、自反性、对称性、转置对称性）
- 标准记号及常用函数（单调性、取整、模运算、多项式、指数、对数、阶乘、多重函数、多重对数函数、斐波那契数）

标准记号与常用函数

■阶乘

- ➢ $n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n = \prod_{i=1}^{n} i$，并定义0!=1
- ➢ $n! = \begin{cases} 1, & n = 0 \\ n \cdot (n-1)!, & n > 0 \end{cases}$

➢斯特林(Stirling)近似公式

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \left(1 + \Theta\left(\frac{1}{n}\right)\right)$$

标准记号与常用函数 (续)

■阶乘

➢斯特林(Stirling)近似公式

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \left(1 + \Theta\left(\frac{1}{n}\right)\right)$$

➢对所有$n \geq 1$，有 $n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\alpha_n}$

其中 $\frac{1}{12n+1} < \alpha_n < \frac{1}{12n}$

➢ $n! = o(n^n) \quad n! = \omega(2^n) \quad \lg(n!) = \Theta(n \lg n)$

标准记号与常用函数 (续)

■多重函数

- 使用记号$f^{(i)}(n)$表示函数$f(n)$重复i次作用于n上：

$$f^{(i)}(n) = \underbrace{f(f(f...f}_{i}(n)))$$

其中，定义$f^{(0)}(n)=n$

- $$f^{(i)}(n) = \begin{cases} n, & i = 0 \\ f(f^{(i-1)}(n)), & i > 0 \end{cases} \quad (i \in \mathbb{Z})$$

标准记号与常用函数 (续)

■多重对数函数

- 使用记号$\lg^* n$表示多重对数，定义如下：

$$\lg^* n = \min\{i \geq 0 : \lg^{(i)} n \leq 1\}$$

通俗理解：对于值n，至少要作用几次lg函数才能得到不超过1的结果？

- 例如：$\lg^*2=1$，$\lg^*4=2$，$\lg^*16=3$，$\lg^*65536=4$，$\lg^*(2^{65536})=5$
- 多重对数是一个增长非常慢的函数
- 可探测的宇宙中原子数目估计约为$10^{80} \ll 2^{65536}$，因此很少遇到使得$\lg^* n>5$的输入规模n

标准记号与常用函数 (续)

■斐波那契数

- ➢每个斐波那契数都是两个前面的数之和：

$$
\begin{aligned}
F_0 &= 0 \\
F_1 &= 1 \\
F_i &= F_{i-1} + F_{i-2}, \quad i \geq 2
\end{aligned}
$$

- ➢产生的序列为：0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...

标准记号与常用函数 (续)

■斐波那契数

➢斐波那契数与黄金分割率ϕ

$$F_i = \frac{\phi^i - \hat{\phi}^i}{\sqrt{5}} = \left\lfloor \frac{\phi^i}{\sqrt{5}} + \frac{1}{2} \right\rfloor$$

斐波那契数以指数形式增长

$\hat{\phi}$是ϕ的共轭数，它们是方程$x^2-x-1=0$的两个根：

$$\phi = \frac{1+\sqrt{5}}{2} \approx 1.618$$

$$\hat{\phi} = \frac{1-\sqrt{5}}{2} \approx -0.618$$

本章内容

- ■算法定义及基本概念（教材Chapter 1）
- ■算法描述（教材Chapter 2）
- ■函数增长及渐近记号表示（教材Chapter 3）
- ■标准记号与常用函数（教材Chapter 3）
- ■NP完全性理论（区分并理解P/ NP/ NPC/ NP-Hard 几类问题）（教材Chapter 34）

NP完全性理论

- 从计算的观点来看，要解决的问题的内在复杂性如何？它是“易”计算的还是“难”计算的？
- 知道了一个问题的计算时间下界，就知道了对于该问题能设计出多有效的算法，从而可以较正确地评价对该问题提出的各种算法的效率，进而确定对已有算法还有多少改进的余地
- 许多情况下，要确定一个问题的内在计算复杂性是很困难的

NP完全性理论 (续)

- **多项式时间的算法**：对于规模为n的输入，在最坏情况下的运行时间是$O(n^k)$，其中k为某一确定常数

$$\underbrace{\boxed{O(1) < O(n) < O(n\lg n) < O(n^2)}}_{\text{多项式级别}} < \underbrace{\boxed{O(2^n) < O(n!) < O(n^n)}}_{\text{超多项式级别}}$$

NP完全性理论 (续)

- 一般来说，我们认为在多项式时间内可解的问题是易处理的问题，在超多项式时间内解决的问题是不易处理的问题
- “NP完全”(NPC) 问题：迄今为止，既没有人找出求解这类问题的多项式时间算法，也没有人能够证明对这类问题不存在多项式时间算法

判定问题与优化问题

- 最优化问题 (optimization problem)：每一个可行解都有一个关联的值，我们希望找出一个具有最佳值的可行解
    - 例如：SHORTEST-PATH问题：已知无向图G以及顶点u和v，要找出u和v之间经过边数目最少的一条路径。在无权、无向图中的单点对间最短路径问题
- 判定问题 (decision problem)：问题的答案是简单的“是”或“否”（或者更形式化地，答案是“1”或“0”）
    - 例如：SHORTEST-PATH相关的判定问题PATH：判定给定的无向图G、顶点u和v、一个整数k，在u和v之间是否存在一条至多包含k条边的路径

判定问题与优化问题 (续)

- NPC类问题只局限于判定问题
- 通常通过对优化的值强加一个界，就可以将一个给定的优化问题转化为一个相关的判定问题
- 试图证明最优化问题“不易处理”时，就可利用该问题与相关的判定问题之间的关系
  - 判定问题要“更容易一些”，至少“不会更难”
  - 如果能够证明某个判定问题是困难问题，相当于证明了其相关最优化问题也是困难的

P类问题

- P类问题 (Polynomial Problem)：在多项式时间内可以解决的问题
  - 例如：排序问题可在$O(n^2)$时间复杂度内解决

NP类问题

- NP类问题 (Non-deterministic Polynomial Problem)：在多项式时间内可以被验证的问题
  - 非确定性算法：猜测 + 验证，猜测阶段是非确定性的，给出问题解的一个猜测；验证阶段是确定性的，验证阶段给出解的正确性
  - 设算法A是解一个判定问题Q的非确定性算法，如果算法A的验证阶段可以在多项式时间内完成，则称算法A是一个多项式时间非确定性算法，也称问题Q是非确定性多项式时间可解的（NP问题）

NP类问题 (续)

- NP类问题 (Non-deterministic Polynomial Problem)：在多项式时间内可以被验证的问题
  - 解可能不好找，但可猜测一个解，很容易验证是否正确
  - $P \subseteq NP$
  - P类问题是否是NP类问题的真子集，目前是一个开放问题

NPC类问题

- 如果一个NP问题和其他任何NP问题一样“不易解决”，那么我们认为这一问题是NPC类问题或称之为NP完全问题
- 如果任何NP完全问题可以在多项式时间内解决，那么所有NP问题都有一个多项式时间算法
- 若能够确定一个NP完全问题，就可以提供充分的论据说明其不易处理性
  - 采用近似算法或解决某种易处理问题的特例，而不是寻找求的问题精确解的快速算法

归约 (reduce)

- 多项式时间归约算法 (reduction algorithm)，将A的任何实例α转化成B的某个实例β：
  - 转换操作需要多项式时间
  - 两个实例的解是相同的（α的解是“是”iff β的解也是“是”）

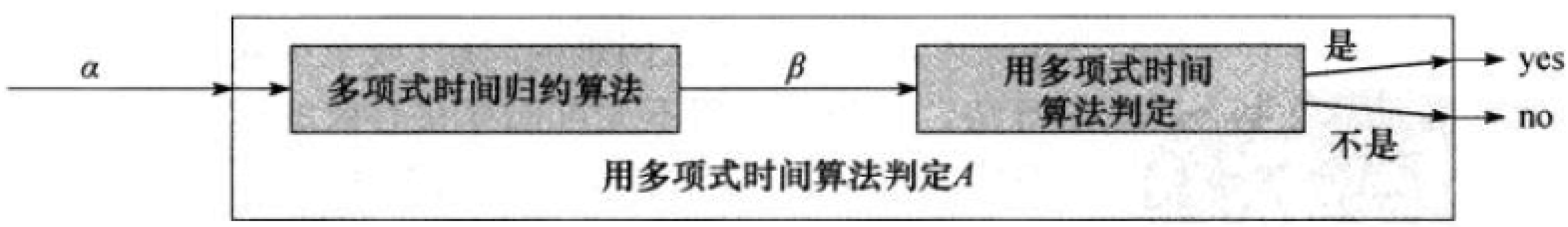


图 34-1 在给定另一个问题 B 的多项式时间判定算法后，如何利用多项式时间归约算法在多项式时间内解决判定问题 A。将 A 的实例 α 在多项式时间内转换为 B 的实例 β，在多项式时间内解决 B，再将 β 作为 α 的解

归约 (reduce，续)

- 将对问题A的求解“归约”到对问题B的求解，可利用问题B的“易求解性”来证明A的“易求解性”
- 相反的，假设有判定问题A，已知它不可能存在多项式时间算法，若有一个多项式时间归约，将A的一个实例转化为一个B的实例，则B也不可能存在多项式时间算法
- A的求解归约到B的求解（或者说问题A归约到问题B），则问题B难于问题A，或者说A并不比B更难解决
  - A的任何实例都可以被“容易地重新描述”为B的实例，B的实例的解也是A的实例的解

归约 (reduce，续)

- 例：求解关于未知量x的线性方程问题可以转化为求解二次方程问题
  - 实例$ax+b=0$可变换为$0x^2+ax+b=0$

线性方程求解 $\xrightarrow{\text{归约到}}$ 二次方程求解

特例问题$ax+b=0$

易解决

泛化问题$0x^2+ax+b=0$

难解决

NPC类问题 (续)

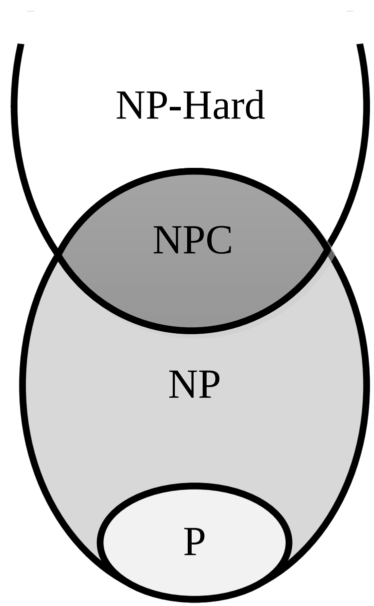
- 若对于问题Q，满足：
  1. Q$\in$NP
  2. 每个NP问题都可以多项式时间归约到Q

     或：任意一个NPC问题都可以多项式时间归约到Q

  那么问题Q是一个NP完全问题
- 若问题Q满足了第二条，则称为NP-Hard问题

NP完全性理论 (续)

- 定理34.4 (p628) 如果任何NP完全问题是多项式时间可求解的，则P=NP。等价地，如果存在某一NP中的问题不是多项式时间可求解的，则所有NP完全问题都不是多项式时间可求解的。
- 目前认为P≠NP



Karp的21个NP完全问题

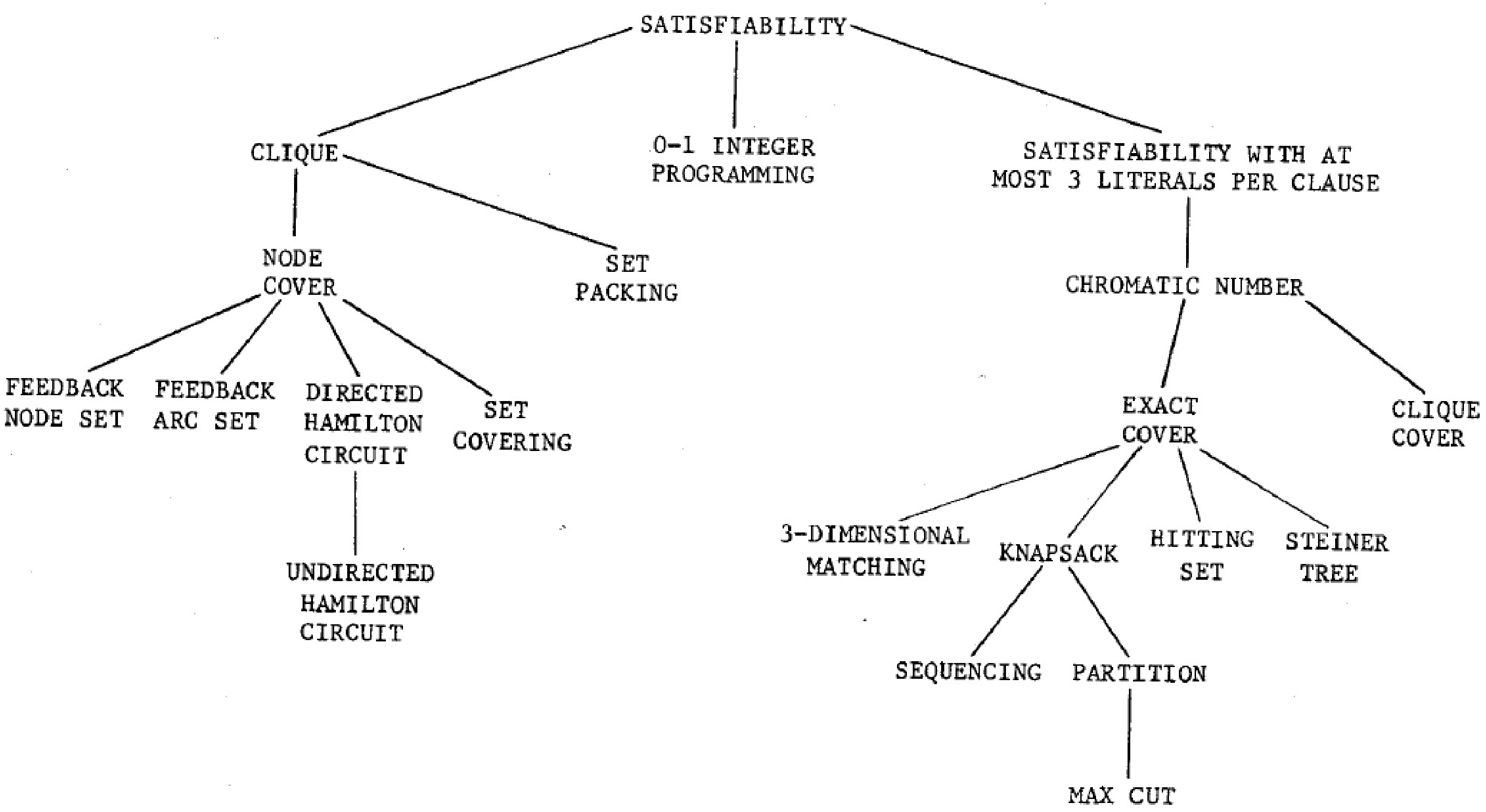


FIGURE 1 - Complete Problems

典型的NPC问题

■合取范式的可满足性问题*k*-CNF-SAT

➢布尔表达式 $(x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_2 \vee \neg x_3)$

- 布尔变量x_i，取值为0或1
- 布尔连接词：例如∧（与，合取）、∨（或，析取）、¬（非）
- 括号

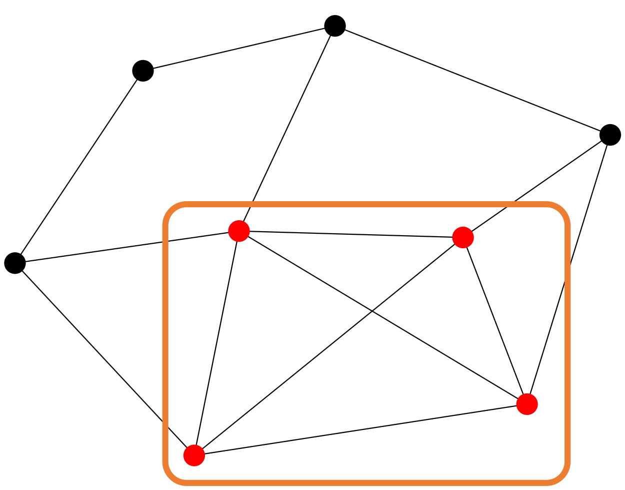
➢合取范式 (Conjunctive Normal Form, CNF)

- 布尔表达式用合取连接若干个析取子句
- 若每个子句恰好有*k*个布尔变量或其否定形式，则称为*k*合取范式（*k*-CNF）
- 2-CNF-SAT∈P，但3-CNF-SAT∈NPC $\Omega\left(2^n\right)$

典型的NPC问题 (续)

■团问题CLIQUE

- ➢给定一个无向图G=(V, E)和一个正整数k，判定图G是否包含一个k团，即是否存在$V' \subseteq V$和|V'|=k，且对任意u,w∈V'，有(u,w)∈E。
- ➢朴素算法：找出所有k个顶点的子图并验证



$$\Omega\left(C_{|V|}^{k} C_{k}^{2}\right) = \Omega\left(k^2 C_{|V|}^{k}\right)$$

k为常数，则为$\Omega(|V|^k)$

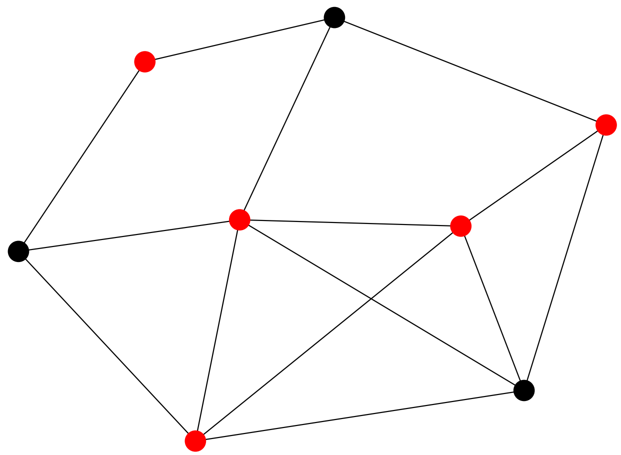
k通常与|V|相关，则为超多项式级别

存在一个规模k为4的团

典型的NPC问题 (续)

■顶点覆盖问题VERTEX-COVER

- 给定一个无向图$G=(V, E)$和一个正整数k，判定是否存在$V' \subseteq V$和$|V'|=k$，使得对任意$(u,v)\in E$有$u\in V'$或$v\in V'$，如果存在，就称V'为图G的一个大小为k的顶点覆盖
- 即E中每条边关联的两个顶点至少有一个在V'中



$$\Omega\left(|E|C_{|V|}^{k}\right)$$

存在一个规模k为5的顶点覆盖

典型的NPC问题 (续)

■子集和问题SUBSET-SUM

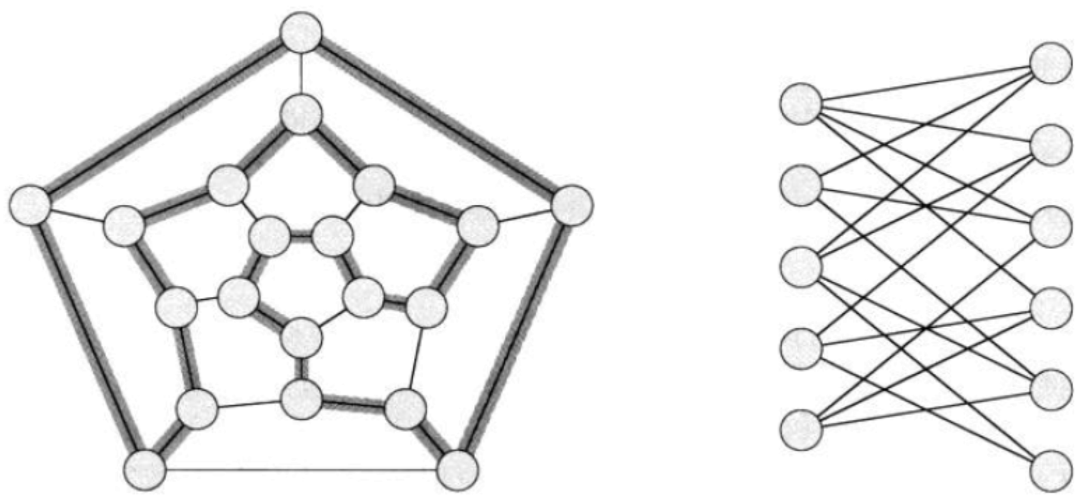
- 给定一个正整数有限集*S*和一个整数目标*t* >0，判定是否存在*S*的一个子集$S' \subseteq S$，使得*S*'中的整数的和为*t*
- 例：*S*={1, 2, 7, 14, 49, 98, 343, 686, 2409, 2793, 16808, 17206, 117705, 117993}, *t* =138457，则子集*S*'={1, 2, 7, 98, 343, 686, 2409, 17206, 117705}是该问题的一个解

$$\Omega\left(2^n\right)$$

典型的NPC问题 (续)

■哈密顿回路问题HAM-CYCLE

- ➢给定无向图$G=(V, E)$判定其是否含有一条哈密顿回路
- ➢无向图$G=(V, E)$中的一条哈密顿回路是通过V中每个顶点的简单回路



（a） （b）

图 34-2 (a)一个表示正十二面体中顶点、边、面的图，其中哈密顿回路以阴影边示出。(b)一个包含奇数个顶点的二分图。任何这样的图都是非哈密顿图

典型的NPC问题 (续)

■旅行商问题TSP (Traveling Salesman Problem)

- ➢一个售货员必须访问n个城市，售货员希望恰好访问每个城市一次，并最终回到出发城市。售货员从城市i到城市j的旅行费用为一个整数$c(i, j)$，旅行所需的全部费用是他旅行经过的各边费用之和，售货员希望整个旅行费用最低。判定问题为：旅行费用不超过k。
- ➢给定一个无向完全图$G=(V, E)$及定义在$V\times V$上的一个费用函数c和一个整数k，判定G是否存在经过V中各顶点恰好一次的回路，使得该回路的费用不超过k

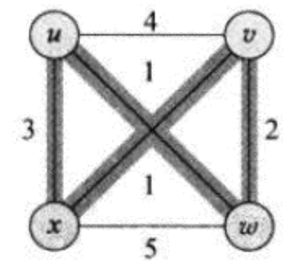


图 34-18　旅行商问题的一个实例。阴影覆盖的边表示费用最低的旅行路线，其费用为 7

本章小结

- 算法定义、算法 vs. 程序
- 问题求解步骤
- 算法描述（伪代码）
- 插入排序分析：正确性分析、算法执行时间（最好、最坏、平均情况）
- 渐近表示法（Θ/O/Ω/o/ω记号）及函数比较
- 标准记号及常用函数（单调性、取整、模运算、多项式、指数、对数、阶乘、多重函数、多重对数函数、斐波那契数）
- NP完全性理论（P/NP/NPC/NP-Hard，典型的NPC问题）