算法分析与设计

近似算法

主要内容

- 近似算法
- 近似算法的性能
- 几个NPC问题的近似算法

近似算法

- 迄今为止，所有的NP完全问题，均未能找到多项式时间的算法，故当问题规模较大时，求得最优的精确解的可能性很小
- 在此情况下，往往退而去求比最优精确解稍差一点的解作为问题的近似答案

近似算法的性能

- 近似算法一般都比较简单，但设计近似算法时必须关注所设计的算法所得到的近似解与最优解之间的差距到底有多大

近似算法的性能

- 若一个最优化问题的最优值为c^*，求解该问题的一个近似算法求得的近似最优解相应的目标函数值为c，则将该近似算法的近似比定义为$\max\{c^*/c, c/c^*\}$
- 在通常情况下，近似比是问题输入规模n的一个函数$\rho(n)$，即 $\max\{c^*/c, c/c^*\} \le \rho(n)$

近似方案

- 常数近似比的近似算法
- 多项式时间近似方案
  - PTAS, Polynomial Time Approximation Scheme
- 完全多项式时间近似方案
  - FPTAS, Fully Polynomial Time Approximation Scheme

装箱问题（Bin Packing）

- 设有n个物体$u_1,u_2,\ldots,u_n$，每个物体的体积不超过1。另外，有足够多的、体积为1的箱子。箱子、物体均是长方体且截面相同
- 问如何装箱，使得所用箱子数最少？

装箱问题（Bin Packing）

- First-Fit（FF）算法
  - 从排在最前面的箱子开始，对每个箱子剩余的体积逐一进行检查，一旦碰到第一个能够装进当前物体的箱子时，就立即把该物体装入这个箱子。对每个物体反复执行上述程序
- 算法的最坏时间复杂度：$O(n^2)$
- 用FF算法不能保证所获得的解是最优解

装箱问题（Bin Packing）

- 分析
  - 记号I：表示某一问题的任一实例
  - OPT(I)：表示该实例的最优解
- FF算法满足：对于任何装箱实例I，都有FF(I)≤2OPT(I)
- 更为准确地，对所有装箱问题的实例I，都有$FF(I) \le \lceil 17/10\ OPT(I) \rceil$，且存在OPT(I)任意大的实例I，使得$FF(I) \ge 17/10(OPT(I)-1)$

装箱问题（Bin Packing）

- Next-Fit（NF）算法
  - 先把第一个空箱置为当前箱。然后依次把物品$u_1,u_2,\ldots,u_n$按下列方式装箱：若当前所指的箱子里放得下u_i，则把u_i放入箱中；若放不下则把u_i放入下一个空箱，把当前指针指向（放u_i的）该箱
- 算法的最坏时间复杂度：O(n)（因为对每个物品只检查当前的箱子）

装箱问题（Bin Packing）

- 分析
- NF算法满足：对于任何装箱实例I，都有 $NF(I) \le 2OPT(I)-1$

装箱问题（Bin Packing）

- Best-Fit（BF）算法
  - FF算法的修改：在已装有物品的箱子中，找一个既能放下u_i、又使得其剩余空间最小的箱子来放u_i
  - 表面上看起来该算法要比FF法更能充分利用空间，但实际上，Johnson等人证明了BF法在最坏情况下的性能，本质上与FF法相同

装箱问题（Bin Packing）

- FFD（First-Fit Decreasing）算法
  - 先将所有物品从大到小排序，然后再使用FF法
- 分析
  - 对一切装箱实例I，有FFD(I)≤⌈4/3 OPT(I)⌉，当OPT(I)=3k+1时，有FFD(I)≤⌊4/3 OPT(I)⌋
  - 对所有装箱问题的实例I，有FFD(I)≤ 11/9OPT(I)+1 (1990)

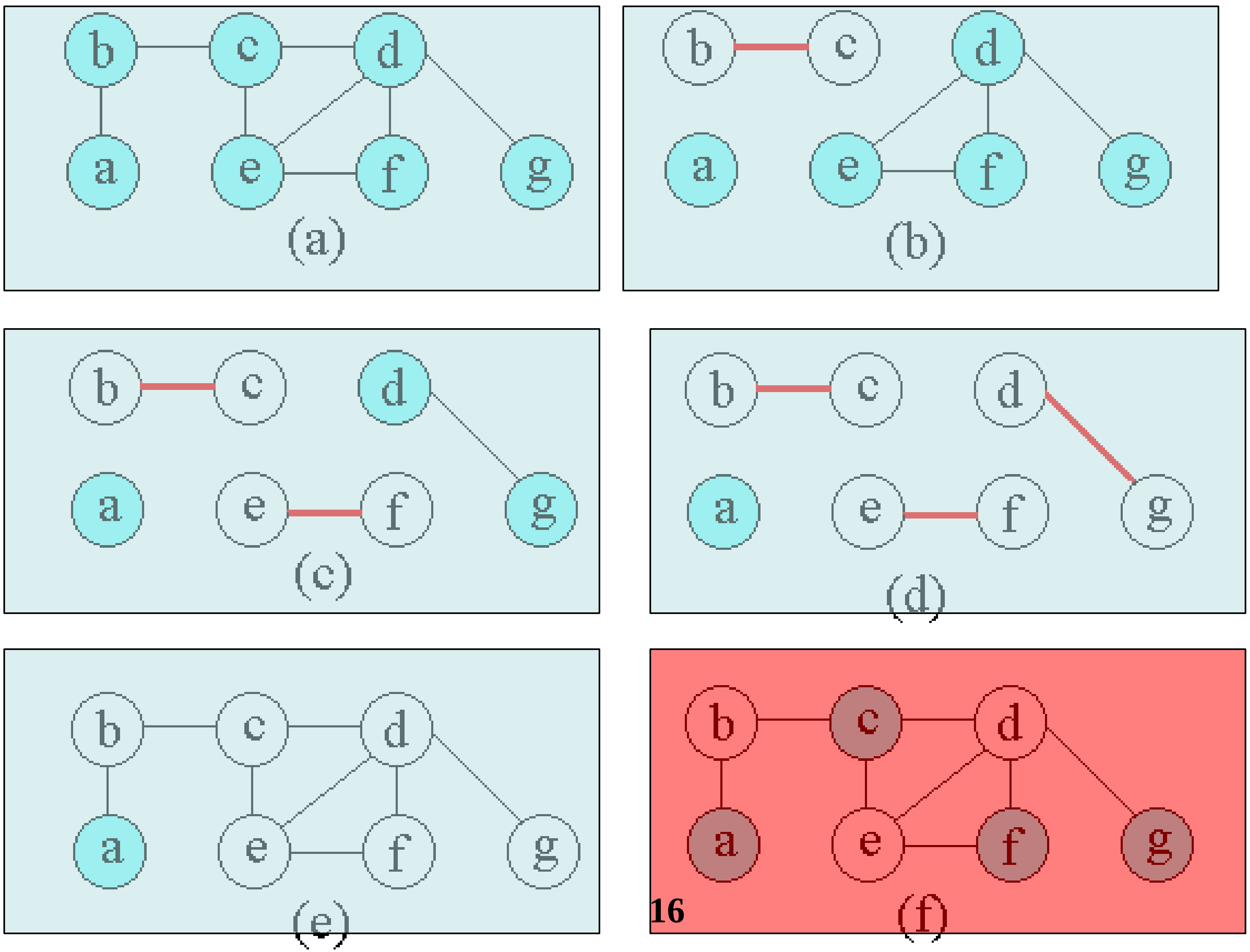
顶点覆盖问题

- 问题描述
  - 无向图G=(V,E)的顶点覆盖是它的顶点集V的一个子集V'$\subseteq$V，使得若(u,v)是G的一条边，则v$\in$V'或u$\in$V'
  - 顶点覆盖V'的大小是它所包含的顶点个数|V'|
  - 顶点覆盖问题就是要求在一个给定的无向图中，找出一个具有最小规模的顶点覆盖

顶点覆盖问题

```
VertexSet approxVertexCover ( Graph g )
{
cset=∅；
   e1=g.e；
   while (e1 != ∅)
{
     从e1中任取一条边(u,v)；
     cset=cset∪{u,v}；
     从e1中删去与u和v相关联的所有边；
   }
   return cset
}
```

顶点覆盖问题



旅行商问题

- 旅行商问题（TSP）简单描述
  - 给定一个完全无向图G=(V,E)，其每一边(u,v)∈E有一非负整数代价c(u,v)
  - 要找出G中具有最小代价的哈密尔顿回路
- TSP的特殊性质
  - 代价函数c往往具有三角不等式性质，即对任意的3个顶点u,v,w∈V，有：$c(u,w) \le c(u,v)+c(v,w)$
  - 当图G中的顶点就是平面上的点，任意2顶点间的代价就是这2点间的欧氏距离时，代价函数c就具有三角不等式性质

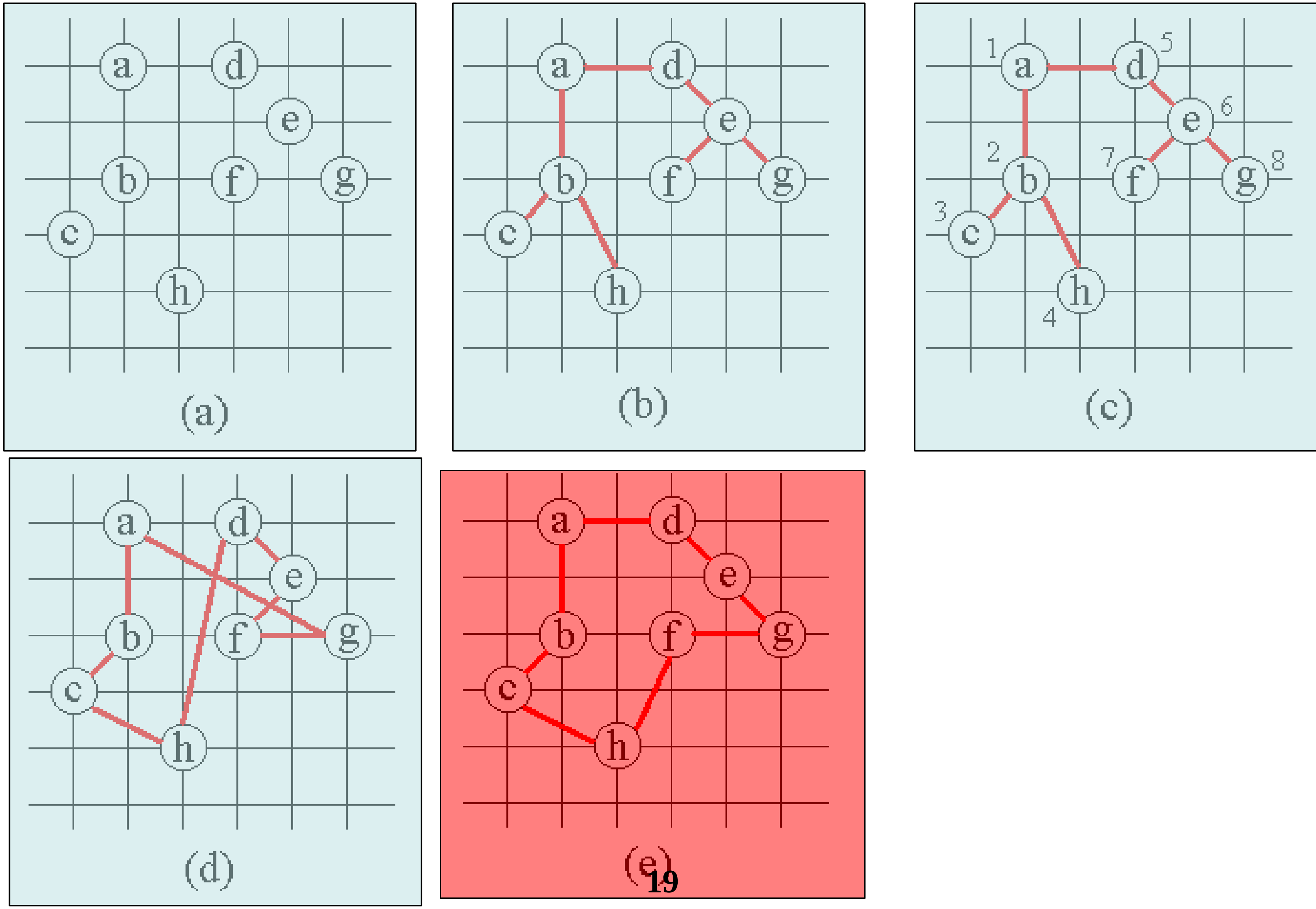
旅行商问题

- 满足三角不等式性质的旅行商问题
  - 对于给定的无向图G，可以利用找图G的最小生成树的算法设计找近似最优的旅行售货员回路的算法

```
void approxTSP (Graph g)
{
        (1)选择g的任一顶点r；
        (2)用Prim算法找出带权图g的一棵以r为根的最小生成树T；
        (3)前序遍历树T得到的顶点表L；
        (4)将r加到表L的末尾，按表L中顶点次序组成回路H，作为计算结果返回；
}
```

  - 当代价函数满足三角不等式时，算法找出的路径的代价不会超过最优路径的代价的2倍

旅行商问题



旅行商问题

- 一般的旅行商问题
  - 在代价函数不一定满足三角不等式的一般情况下，不存在具有常数近似比的解TSP问题的多项式时间近似算法，除非P=NP
  - 换句话说，若P≠NP，则对任意常数ρ>1，不存在近似比为ρ的求解旅行商问题的多项式时间近似算法