# 《算法设计与分析》复习提要

陈卫东（chenwd@scnu.edu.cn）

2021-12

**提纲**



## 一、算法分析基础知识

**1**、设函数 $f(n)=n^{10}$ 和 $g(n)=2^n$，则 $f(n)$和 $g(n)$有如下关系（　　）。

**A.** $f(n)=O(g(n))$　　**B.** $f(n)=\Omega(g(n))$　　**C.** $f(n)=\Theta(g(n))$　　**D.** 以上都不对

**2**、某递归算法的时间复杂度 $T(n)$满足递归方程：$T(n)=2T(n/2)+bn$（$n\geq 2$）且 $T(1)=a$，其中 $n=2^k$, $a$, $b$, $k$ 均为正整数。$T(n)$可用$\Theta$表示为（　　）。

**A.** $\Theta(n^2)$　　**B.** $\Theta(n\log n)$　　**C.** $\Theta(n\log^2 n)$　　**D.** $\Theta(n^2\log n)$

**3**、快速排序的最坏情况时间复杂度是多少？什么情况下会出现最坏情况运行时间？使用快速排序时可采用哪些方法尽量避免出现最坏情况运行时间？

——快速排序的最坏情况时间复杂度为$\Theta(n^2)$，当待排序序列比较有序时快速排序会出现最坏情况运行时间。可采用如下两种方法之一来尽量避免快速排序出现最坏情况运行时间：排序前将序列洗牌，打乱其较有序的状态；每次执行划分过程前在序列中随机挑选一个元素交换到序列前面作为划分元素。

**4**、分析下列冒泡排序的时间复杂度。

**算法**　REC**BUBBLESORT**
**输入:**　$n$ 个元素的数组 $A[1\ldots n]$。
**输出:**　按照非降序排列的数组 $A[1\ldots n]$。

```
1.  sorted←false
2.  sort(n)

过程 sort(i)                    //对 A[1…i]排序
1.  if i ≥2 and not sorted then
3.      for j←1 to i–1
4.          if A[j]>A[j+1] then
5.              交换 A[j]与 A[j+1]
6.              sorted ←false
7.          end if
8.      end for
9.      sort(i–1)               //递归对 A[1…i–1]排序
10. end if
```

——设上述递归算法中元素比较的最多次数 $T(n)$满足的递归方程为：

$T(n)=T(n-1)+n-1$，　$n\geq 2$；　　$T(1)=0$。

使用展开法解得 $T(n)=1+2+\ldots.+n-1=\Theta(n^2)$。

**5**、假设图 $G=\langle V,E\rangle$用邻接矩阵表示，$n=|V|$，$m=|E|$。分析图的深度优先遍历算法 DFS 的时间复杂度。

（1）算法 DFS 中有哪些基本操作？各基本操作耗费时间是多少？

（2）请根据基本操作的耗费时间，分析得出算法 DFS 的时间复杂度。

**算法**　DFS
**输入**：有向或无向图 $G=(V,E)$。
**输出**：在相应的深度优先搜索树中对顶点的前序和后序。

```
1. predfn ← 0; postdfn ← 0
2. for 每个顶点 v∈V
3.     标记 v 未访问
4. end for
5. for 每个顶点 v∈V
6.     if  v 未访问 then dfs(v)
7. end for
```

**过程**　dfs(v)

```
1. 标记 v 已访问
2. predfn ← predfn + 1
3. for 每条边(v,w)∈E
4.     if w 标记为未访问 then dfs(w)
5. end for
6. postdfn ← postdfn + 1
```

——（1）算法 DFS 中基本操作包括：**访问结点**、**检测结点**（检测一个结点 $v$ 就是根据 $v$ 找到其相邻结点）。图中每个结点都恰好访问 1 次，访问结点的总次数为 $n$，耗费时间为$\Theta(n)$；图中每个结点间断的需要检测多次，检测时即扫描相应结点的邻接矩阵的一行，因此检测总耗费时间为$\Theta(n^2)$。

（2）算法 DFS 的时间复杂度主要由访问结点的时间、检测结点的时间来决定，其他运算耗费时间不超过这两部的耗费时间。故算法 DFS 时间复杂度为$\Theta(n)+\Theta(n^2)=\Theta(n^2)$。

## 二、算法设计基本方法

**1**、假定 $n=2^k$（$k$ 为正整数）。设计一个分治算法在数组 $A[1..n]$中找**最大元素**和**第二大元素**，要求比较次数为 $3n/2-2$。

（1）用文字简要描述递归形式分治算法的基本思路（不用写出完整代码）。

（2）请给出算法中比较次数 $T(n)$满足的递归方程，并证明 $T(n)=3n/2-2$。

——（1）一个分治算法描述如下：当序列 $A[1..n]$中元素的个数 $n=2$ 时，通过直接比较即可找出序列的第 2 大元素。当 $n>2$ 时，递归求出序列 $A[1..n/2]$和 $A[n/2+1..n]$中的第 1 大元素 $x_1,x_2$ 和第 2 大元素 $y_1,y_2$。然后，通过两次比较即可在四个元素 $x_1,y_1$，$x_2,y_2$ 中找出 $A[1..n]$的最大元素 $x$ 和第 2 大元素 $y$。关键的比较运算描述如下：

| | |
|---|---|
| **If** $x1\geq x2$ **Then**<br>　$x \leftarrow x1$<br>　**If** $y1\geq x2$ **Then** $y \leftarrow y1$<br>　**Else** $y \leftarrow x2$ **EndIf** | **Else**<br>　$x \leftarrow x2$<br>　**If** $y2\geq x1$ **Then** $y \leftarrow y2$<br>　**Else** $y \leftarrow x1$ **EndIf**<br>**EndIf** |

（2）算法时间复杂度满足如下递归方程：

$T(n)=2T(n/2)+2$（$n>2$）；$T(2)=1$。

因为 $n=2^k$（$k$ 为正整数），所以，

$T(n)=T(2^k)=2T(2^{k-1})+2=2^2T(2^{k-2})+2^2+2=\cdots$

$=2^{k-1}T(2)+2^{k-1}+\cdots+2^3+2^2+2=2^k+2^{k-1}-2=3n/2-2$。

**2、**考虑在 $n$ 个互不相同元素的数组 $A[1\ldots n]$中找出所有前 $t$ 个最小元素的问题（这 $t$ 个元素不要求由小到大有序）。这里 $t$ 不是一个常量，而是作为输入数据的一部分。我们可以很容易用排序算法解决此问题并返回 $A[1\ldots t]$，然而耗费时间为 $O(n\log n)$。请设计一个时间复杂度较低的算法求解该问题。

（注意，你可以调用选择第 $k$ 小元素算法 SELECT($A$, $low$, $high$, $k$)求解，该算法用于在数组元素 $A[low\ldots high]$中返回第 $k$ 小元素，其时间复杂度为$\Theta(n)$。但不能直接使用 $t$ 次调用算法 SELECT 来解决问题，因为这将耗费时间 $\Theta(tn)=O(n^2)$ （这里 $t$ 不是常量），不符合题意。）

——（1）算法思路：先调用算法 SELECT($A$,1,$n$,$t$)在数组 $A[1\ldots n]$中得到第 $t$ 小元素，然后用第 $t$ 小元素作为划分元素，对数组 $A[1\ldots n]$进行一趟划分，使得小于等于划分元素的元素都交换到划分元素前面即可。

（2）调用算法 SELECT($A$,1,$n$,$t$)需要时间$\Theta(n)$，划分算法需扫描一遍数组 $A[1\ldots n]$耗时为$\Theta(n)$，因此算法时间复杂度为$\Theta(n)$。

**3、（1）**简述 Dijkstra 算法的基本思路。

**（2）** 在图 1 中应用 Dijkstra 算法能求出从 ***s*** 到其它所有结点的最短路径吗？为什么？

**（3）**考虑用 Dijkstra 算法求出图 2 中从 ***s*** 到其它所有结点的最短路径。请给出主要计算过程及计算结果。

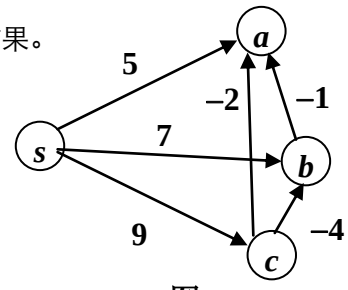



图 1

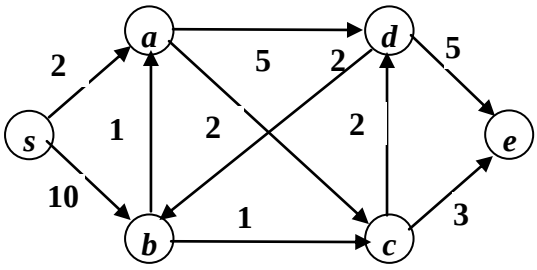



图 2

——（1）Dijkstra 算法是一个贪心算法，用于求图中某点（这里是结点 $a$）到其它所有点之间的最短路。其贪心准则是，按照路长由短到长依次构造这 $n$-1 条路。令$\lambda[u]$表示结点 $a$ 只经过 $X$ 中的结点到达 $u$ 的最短路长，则每次选择$\lambda[u]$最小的 $u$ 加入到 $X$ 中，即结点 $a$ 到 $u$ 的最短路长为$\lambda[u]$。

（2）因图 1 中有些边的权重为负值，所以应用 Dijkstra 算法不能正确求出从 ***s*** 到其它所有结点的最短路径。

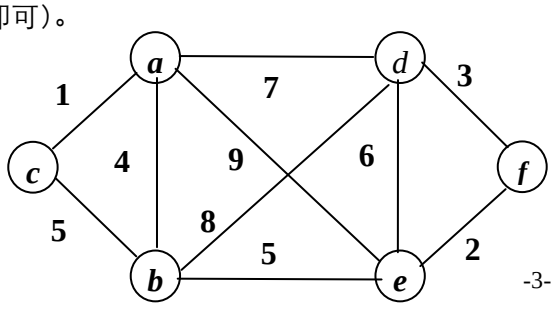
（3）计算过程及计算结果如下表所示。

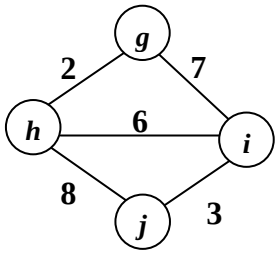
| 迭代次数 | 集合 X 中的元素 | λ[s] | λ[a] | λ[b] | λ[c] | λ[d] | λ[e] | 最短路径及其路长 |
|---|---|---|---|---|---|---|---|---|
| 初始化 | S | - | **2** | 10 | ∞ | ∞ | ∞ | s→a （长度为 2） |
| 1 | s，a | - | - | 10 | **4** | 7 | ∞ | s→a→c （长度为 4） |
| 2 | s，a，c | - | - | 10 | - | **6** | 7 | s→a→c→d （长度为 6） |
| 3 | s，a，c，d | - | - | 8 | - | - | **7** | s→a→c→e （长度为 7） |
| 4 | s，a，c，d，e | - | - | **8** | - | - | - | s→a→c→d→b (长度为 8) |
| 5 | s，a，c，d，e，b | - | - | - | - | - | - | |

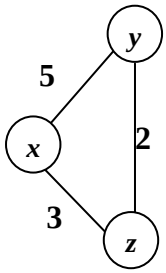
**4、**给定带权无向图 $G$，如果 $G$ 是连通图，则 $G$ 的最小成本生成森林就是 $G$ 的最小生成树；如果 $G$ 不连通，则 $G$ 的各个连通分支的最小生成树组成图 $G$ 的最小成本生成森林。

**（1）**最小生成树算法有 Prim 算法和 Kruskal 算法。这两个算法中哪一个算法更适合求解上述最小成本生成森林？ 简述原因。

**（2）**带权无向图 $G$ 的图形如下，试求解其最小成本生成森林（画出最小成本生成森林即可）。

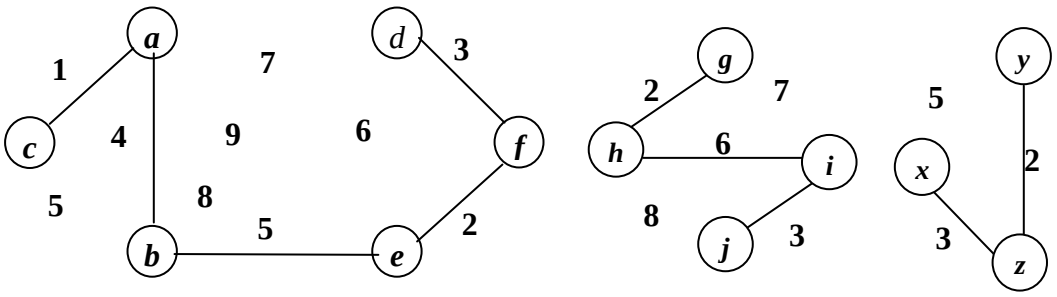


——（1）对于不连通图，Prim 算法从某点开始依次在该点所在连通分支中加边，最终能得到该连通分支的最小生成树；而 Kruskal 算法按照边权由小到大依次考虑边的加入，能求得所有连通分支的最小生成树。因此，Kruskal 算法更适合求解最小成本生成森林问题。

（2）图 $G$ 的最小成本生成森林如下图所示。




**5、【最大间隔聚类问题】**给定非负带权无向图 $G=\langle V,E\rangle$及正整数 $k(1\le k\le|V|)$，试将 $G$ 划分为 $k$ 个非空子图 $G_1,G_2,\ldots,G_k$使得任意两子图的最小间距最大。这里两子图的间距是指跨于这两子图之间的最小权的边的权值。

——贪心算法(可用最小生成树问题的 Kruskal 算法求解，在添加后 $k-1$ 条边之前停止即可)。

**6、**无向图 $G=\langle V,E\rangle$是二部图是指该图的点集 $V$ 能划分成非空的两部分 $V_1$ 和 $V_2$ 使得图 $G$ 中任何边的两个端点分别属于 $V_1$ 和 $V_2$。考虑用图的**深度优先遍历**方法设计一个算法判别无向图 $G=\langle V,E\rangle$是否是二部图。要求：

**（1）**用文字简要描述算法的基本思路（不用写代码）。

**（2）**给出算法正确性的证明。

**（3）**如果要求利用**宽度优先**遍历图的方法设计算法判别图 $G=\langle V,E\rangle$是否是二部图呢?

——（1）基于深度优先遍历图的算法思路如下： 通过从某点开始对图进行深度优先遍历来依次给图中相邻节点分别着黑、白两种颜色（节点的“访问”即为节点的“着色”）。在此过程中如果不存在有两个端点同色的**回边**，则断定该图不是二部图，否则该图是二部图。

（2）上述算法使得图中所有边被分成“树边”和“回边”，且所有树边的两端点是不同色的。

如果图 $G$ 是二部图$\langle V_1\cup V_2,E\rangle$，则其所有边都跨于 $V_1$ 和 $V_2$ 之间，且算法中给 $V_1$ 中所有点是同一种颜色，给 $V_2$ 中所有点是另一种颜色。因此，不存在有两个端点同色的回边。

反之，若不存在有两个端点同色的回边，下证图 $G$ 是二部图。将 $G$ 中所有节点按颜色归类：所有黑色点的集合记作 $V_1$，所有白色点的集合记作 $V_2$。既然 $G$ 中树边都跨于 $V_1$ 和 $V_2$ 之间且没有两端点同色的回边，说明所有边都跨于 $V_1$ 和 $V_2$ 间，即图 $G$ 是二部图。

（3）采用宽度优先遍历图 $G$，得到宽度优先生成树。如果宽度优先生成树的同层节点没有边出现，则图 $G$ 为二部图，否则图 $G$ 不是二部图。在图 $G$ 的宽度优先生成树中，$G$ 中任何边必跨于相邻层节点之间或同层节点之间，前者是树边，后者是横跨边。若宽度优先生成树的同层节点没有边，则图 G 为二部图。反之，若图 $G$ 是二部图，则必有其宽度优先生成树同层节点之间无边。算法正确性证毕。

**7、**使用动态规划法求解下列 0-1 背包问题实例：物品个数 $n=4$，重量 $W=\{2,3,4,5\}$，利润 $P=\{3,4,5,7\}$，背包承重量 $C=9$。

**（1）**给出递推计算的公式。

**（2）**用表格或图展示出主要计算过程，并指出问题的解。

——（1）递推计算公式：

令 V[i][j]表示在物品 1,2,…,i 中挑选物品装入容量是 j 的背包中的最大价值。则对于 1≤i≤n, 1≤j≤C，有如下递推计算公式：

V[i][0]=0, V[0][j]=0,

V[i][j]=V[i-1][j] （j<W[i]），

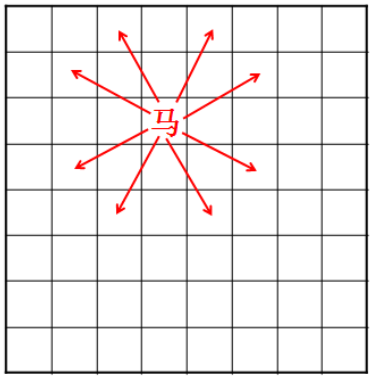
V[i][j]=max{V[i-1][j],V[i-1][j-W[i]]+P[i]}（j≥W[i]）。

（2）计算过程：

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 2 | 0 | 0 | 3 | 4 | 4 | 7 | 7 | 7 | 7 | 7 |
| 3 | 0 | 0 | 3 | 4 | 4 | 7 | 8 | 9 | 9 | 12 |
| 4 | 0 | 0 | 3 | 4 | 5 | 7 | 8 | 10 | 11 | 12 |

由此得最优装包价值为 12。最优的装包方法为：（0, 0, 1, 1）或 (1,1,1,0)。

**8、【马周游问题】**在 8×8 棋盘上马按照跳日字的规则能从一格跳到另一格。任给马在棋盘上的起始位置，请问是否存在一种马能跳到棋盘中每个格子恰好一次且最后回到起始位置的周游棋步？如果存在，请找出一种这样的周游棋步。




（1） 用图论知识说明：若马从棋盘正中心位置出发能有周游棋步，则马从棋盘的任何位置出发都有周游棋步。

（2）考虑用回溯法求解马周游问题。请给出解向量形式和搜索树类型，描述剪枝操作。

（3）可采取哪些措施来提高回溯算法的求解效率？

——（1） 将棋盘中每个格子看作一个点，如果马能从一个格子跳到另一个格子，则这两个格子对应的点之间连边，由此得到一个无向图。通过这种方法可将马周游问题建模为无向图的哈密尔顿圈问题。由图论知识可知，一个无向图中存在哈密尔顿圈当且仅当从任何点出发都有哈密尔顿圈。这表明若马从棋盘正中心位置出发能有周游棋步，则马从棋盘的任何位置出发都有周游棋步。

（2）用回溯法求解马周游问题时，其解向量形式可为$(x_1,x_2,\ldots,x_n)$，**其中** $1\le x_i\le 8$，$(1\le i\le n)$。搜索树是一个棵 64 层高的**满 8 叉树**。剪枝操作为：**第 $i$ 步朝 $x_i$ 方向走可行 $\Leftrightarrow$ $x_i$ 方向是未跳过的空格**。

（3）可采取如下措施来提高回溯算法的求解效率：

(i) 先求出马从棋盘中心出发的一条周游棋步，并保存起来。然后根据这条周游棋步通过位置的变换容易得到从任何初始位置出发的周游棋步。

**(ii)** 在回溯法的搜索过程中，每次优先选择往跳入机会的方向格跳过去。

**9、**工厂因为开展新业务打算明年年初新购置某台设备,并要做好后续 5 年该设备的更新计划,决定每年是重新购置该设备还是继续维护现有设备.已经预测今后 5 年该设备的购置费和维护费分别如下表 1 和表 2 所示(购置价格和维护费用的单位: 万元).请设置一个该设备的 5 年更新方案,使得在此期间该设备的购置费和维护费的总和最小.

表 1 每年的设备购置费用(单位: 万元)

| 第 1 年 | 第 2 年 | 第 3 年 | 第 4 年 | 第 5 年 |
|---|---|---|---|---|
| **22** | **22** | **24** | **24** | **26** |

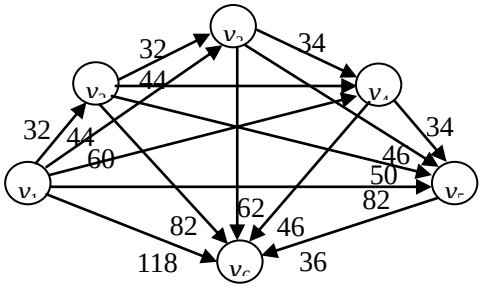
表 2 设备维护费用(单位: 万元)

| 0-1 年 | 1-2 年 | 2-3 年 | 3-4 年 | 4-5 年 |
|---|---|---|---|---|
| **10** | **12** | **16** | **22** | **36** |

——根据上述表格易看出,有多种可供选择的设备更新方案。例如，对于每年购置新设备的方案， 五年购置费为 22+22+24+24+26=118，维护费为 10×5=50，总费用为 168；对于隔年购置新设备的方案，五年购置费为 22+24+26=72，维护费为 10+12+10+12+10=54，总费用为 126。

我们可把这个问题转化为图的最短路径问题来求解。

令 $v_i$ 表示第 $i$ 年初设备是新购置的状态($i$=1,2,3,4,5,6)。对于任意 $i<j$, $v_i$ 到 $v_j$ 连有向边，表示第 $i$ 年初新购置设备使用一直使用到第 $j$ 年才又重新购置，其权重表示第 $i$, $i$+1, … , $j$-1 年的设备购置及维护总费用。每条有向边的权重可以根据上述表格数据计算出来。例如，有向边<$v_1$,$v_4$>的权重表示从第 1 年购置新设备使用到 3 年底的总费用，即 $w_{14}$=22+10+12+16=60，有向边<$v_2$,$v_6$>的权重表示从第 2 年购置新设备使用到 5 年底的总费用，即 $w_{26}$= 22+10+12+16+ 22=82，得到一个有向赋权图 (如下图所示)。原寻找总费用最少的设备更新方案问题转为求从 $v_1$ 到 $v_6$ 的最短路径问题。 我们采用 Dijkstra 算法可以获得两条最短路径：$v_1 \to v_3 \to v_6$, $v_1 \to v_4 \to v_6$ 分别表示第 1 年和第 3 年初需购置新设备的方案和第 1 年和第 4 年初均购置新设备的方案，总费用均为 106 万元。




## 三、NP 完全性理论基本概念

**1**、如何理解求解问题的具体算法的复杂度、问题的复杂度?

**2**、说明语言类 **P**、**NP**、**NP**-complete、**NP**-hard 的含义。假定 **P≠NP**，画图表示其包含关系。

**3**、什么是问题间的归约? （多项式时间）归约在建立问题的复杂度方面有何作用?

**4**、证明一个问题是 **NP** 难度有哪些常用方法?

**5**、如何对付 **NP** 难度问题?

**6**、指出下面关于 **P≠NP** 错误证明的错处，并说明你认为它是错误的理由。

证明：考虑 **SAT** 的一个算法：“在输入 ϕ 上，尝试变量的所有可能的赋值，若有满足 ϕ 的就接受”。该算法显然需要指数时间。所以 **SAT** 有指数时间复杂度，因此 **SAT** 不属于 **P**。因为 **SAT** 属于 **NP**，所以，**P** 不等于 **NP**。

——错在混淆 SAT 问题的时间复杂度和其一个具体算法时间复杂度。

**7**、$n$ 皇后问题可以归约为在一个 $n^2$ 点无向图中找一个 $n$ 点独立集的问题（如何归约？）。

**既然图的独立集问题是 NP 难度的，能否由该归约得出 *n* 皇后问题是 NP 难度的？请简要说明原因。**

——不能。注意归约方向不对。其实皇后问题属于 **P** 类，因为存在有多项式时间求解算法（递归构造出解）。

## 四、NP 完全性理论的基本方法（归约技术）

——为了证明某问题 ***A*** 属于 **NP**-*hard* 类，只需将一个 **NP**-*hard* 问题归约到 ***A*** 即可。（复习“NP 完全性证明举例.doc”中的典型问题的归约证明）

**1**、图 $G=\langle V,E,W\rangle$中给定起点和终点 $s,t\in V$ 和中间节点集 $D\subset V$，要求找出从 $s$ 到 $t$ 且中间须经过 $D$ 中每节点最少 1 次的最短径路。如何求解?

——该问题可求解如下：用 Dijkstra 算法或 Floyd 算法求出$\{s,t\}\cup D$中任意两点间的最短路径长度，然后以$\{s,t\}\cup D$为点集且以这些最短路长为边权构成一个新图 $G^*$。原问题归结为在 $G^*$中求一条经过 $G^*$中每个点至少一次的 $s$-$t$ 最短路径。

**2、证明: Partition ≤ SubsetSum 且 SubsetSum ≤ Partition，即Partition ≡ SubsetSum。**

——归约要点：Partition ≤ SubsetSum是显然的，因为Partition 是 SubsetSum的特例。

SubsetSum ≤ Partition 证明见“NP完全性理论讲义--英文课件08IntractabilityI.pdf中第72页”。

**3、试根据Partition、SubsetSum的NP完全性，证明负载均衡问题是NP完全的。**

——要点：证明Partition ≤ 负载均衡问题，或 SubsetSum ≤负载均衡问题即可。

**4、证明: 点覆盖问题≤Roman-Subset 问题**

**【点覆盖问题】**给定无向图 $G=\langle V,E\rangle$，求最小规模点子集 $S\subseteq V$ 满足 $S$ 能覆盖 $E$ 中所有边。

**【Roman-Subset 问题】**给定一个有向图 $G=\langle V,E\rangle$，求最小规模点子集 $S\subseteq V$ 满足 $G$ 中任何回路上都至少有一点在 $S$ 中。

——归约要点：给定**点覆盖问题**的一个实例，即一个无向图$G=\langle V,E\rangle$。构造**Roman-Subset问题**的实例如下：将$G$中每条无向边$(u,v)\in E$变成两条有向边$<u,v>$和$<v,u>$，所得有向图记作$G^*=\langle V,E^*\rangle$（$G$中每条边变为长度为2的回路）。显然，$G$的一个$k$点覆盖也是$G^*$的一个$k$点Roman Subset。反之，对于$G^*$的一个$k$点Roman Subset一定与每条长度为2的回路（原图$G$中的每条边）有共同点，即它必是$G$的一个$k$点覆盖。

**5、证明: （有向图）Hamilton圈问题≤Hamilton路问题，且Hamilton路问题≤Hamilton圈问题，即Hamilton路问题 ≡ Hamilton圈问题.**

——归约要点：给定Hamilton圈问题的一个实例，即有向图$G=\langle V,E\rangle$。图$G$中任取一点$v$,用两新点$v_1$和$v_2$来替换$v$使得进入$v$的边都进入$v_1$,离开$v$的边都从$v_2$离开。所得图$G'$即为Hamilton路问题实例。显然，图$G$中存在Hamilton圈当且仅当图$G'$中存在一条从$v_2$到$v_1$的Hamilton路。

给定Hamilton路问题的一个实例，即有向图$G=\langle V,E\rangle$。现在添加新点$s$和$t$，且添加从$s$到$V$中每个点的边，添加$V$中每个点到$t$的边，最后再添加一条从$t$到$s$的边，所得图$G'$为Hamilton圈问题实例。显然，图$G$中存在Hamilton路当且仅当图$G'$中存在一条Hamilton圈。

# 五、NP-hard 问题的算法设计

——快速精确算法、近似算法、启发式算法（局部搜索、禁忌搜索、模拟退火、...）

**1、 最大割问题的简单近似算法**

**Step 1.** 初始化：将图 $G=\langle V, E\rangle$点集 $V$ 分为两部 $V_1=V$ 和 $V_2=\varnothing$，并置当前割 Cut=∅。

**Step 2.** 重复下列操作直到 Cut 不能改进为止：将某点从一部移动到另一部来改进 Cut。

试证该算法是 2-近似算法。

——当算法终止时，点集 $V$ 的两部 $V_1$ 和 $V_2$ 中任一点 $v$ 在部内相邻点数 $dn(v)$不超过部间相邻点数 $di(v)$，即 $di(v)\geq dn(v)$。注意，$d(v)=di(v)+dn(v)$且$\sum d(v)=2m$（$m$ 是边数）。因此，$\sum di(v)\geq m$。又易知，$\mathbf{O(I)}\leq m$，$\mathbf{A(I)}=|\text{Cut}|=(\sum di(v))/2\geq m/2$。因此，$\mathbf{O(I)}/\mathbf{A(I)}\leq 2$。

**2、【负载平衡问题】**设有$n$个独立的作业$J_1, J_2,\ldots, J_n$需要处理，其中作业$J_i$需要$t_i$个机时（$i=1,2,\ldots,n$）。现只有$p$台完全相同的机器$\boldsymbol{M}_1, \boldsymbol{M}_2,\ldots, \boldsymbol{M}_p$可以同时使用，任何一项作业可在任

一台机器上加工。现在需要对这些作业进行安排使得所有机器的负载尽可能的均衡。该问题是**NP**难度的问题。下面是求解负载平衡问题的一个简单的贪心算法：

**算法 Greedy-Balance**

1. 初始化： 所有的作业没有被安排
   对第$i$台机器$M_i$，令$T_i$=0，$A(i)$=∅
   //$T_i$表示机器$i$上所有作业的处理总时间，$A(i)$表示分配给机器$i$的作业集合
2. 按处理时间$t_i$的非增次序给作业排序，不妨设 $t_1 \geq t_2 \geq \ldots \geq t_n$
3. **FOR** $j$ =1 ,…, $n$ **DO**
4. 设机器$M_i$达到最小值$\mathbf{min}_k\, T_k$
5. 把作业$j$分配给机器$M_i$
6. $A(i) \leftarrow A(i) \cup \{j\}$
7. $T_i \leftarrow T_i + t_j$
7. **END-FOR**

**试证明该算法是3/2-近似算法。**

——**证明思路:** (1) 如果负载最大的机器上只有一个作业，则显然贪心解值A(I)=O(I)，即A(I)/O(I)=1<3/2；(2) 如果负载最大的机器上至少有2个作业，根据贪心算法思路必有最后加入作业的处理时间$t_j \leq t_{p+1}$ ($j \geq p+1$). 注意到O(I) ≥2 $t_{p+1} \geq 2t_j$. 因此，考虑到该机器上的负载A(I)分为两部分：

最后一项作业之前的负载 A(I)- $t_j \leq (\Sigma_{1\leq i\leq n}\, t_i - t_j)/p$，即有 $A(I) \leq \Sigma_{1\leq i\leq n}\, t_i/p + (1-1/p)\, t_j$；

最后一项作业的处理时间 $t_j \leq$ O(I)/2. (注意到$\Sigma_{1\leq i\leq n}\, t_i/p \leq$ O(I))

因此，有 A(I) ≤ O(I)+(1−1/$p$) O(I)/2= O(I)(3/2− 1/(2$p$)) < O(I)3/2. 即有 A(I)/O(I)<3/2.

**3、【两机调度问题（两机负载均衡问题）】**设有 $n$ 个独立的任务 $J_1, J_2, \ldots, J_n$，需要尽快加工完成。现只有 2 台完全相同的机器可以同时使用，当然，任何一个任务可在任一台机器上加工。问如何调度才能使得整个加工工作及早结束？设任务 $J_i$ 需要 $t_i$ 个机时（$i$=1,2,…,$n$）。该问题是 **NP** 难的，请通过设计该问题的一个多项式近似方案来证明它属于 **PTAS**。

——参见近似算法授课讲义。

**4、【广告策略问题】**给定图 $G=\langle V, E\rangle$,路径集 $P_1, P_2, \ldots, P_t$，试找出图中一个最少元素的点集 $S$, 满足在 $S$ 中所有点上放置广告能使得每一条路径 $P_i(1 \leq i \leq t)$上至少有一个结点放置了广告。试给出该问题的一个近似算法。

——将广告策略问题归约到集合覆盖问题，然后用集合覆盖问题的贪心近似算法求解广告策略问题即可。

**5、（1）** 证明：一个图 $G=\langle V, E\rangle$是二着色当且仅当该图是二部图。

**（2）**如何判断一个图 $G=\langle V, E\rangle$是否为二部图？ ——图的宽度优先遍历

**（3）**现有一算法$A$能枚举图的所有极大独立集，试由该算法设计求解图3着色问题的算法。

**6、【影响最大化问题】**给定有向图 $G=\langle V, E\rangle$，其中每个结点 $v \in V$ 有一个阈值 $w(v) \in [0,1]$，以及参数 $k(1 \leq k \leq |V|)$，试在图中找出 $k$ 个点的子集 $S$ 使得 $S$ 作为信息源能使信息在图中得到最大范围的传播。试给出求解该问题的一个启发式算法。

——贪心启发式算法、迭代改进型启发式算法（局部搜索、禁忌搜索、模拟退火、…）