《算法设计与分析》2019 期末考试真题

尹达恒

2020/12/20

东 南 大 学 考 试 卷（A 卷）

课程名称	算法设计基础	考试学期	2019-2020-2	得分	
适用专业	计算机	考试形式	开卷	考试时间长度	150 分钟

（可携带纸质教材、课件、讲义、笔记）

1. 判断题（共 10 分，每小题 2 分）
   a) $T(n) = n + 7T(n/5) = \theta(n)$ ....( )
   b) P 类问题可多项式时间验证一个解 ....( )
   c) K 团问题可多项式时间内规约($\leq_p$)到集合覆盖问题的实例..........................( )
   d) 近似算法的近似比可能小于 1..( )
   e) 随机算法是可能得到最优解的 ....( )

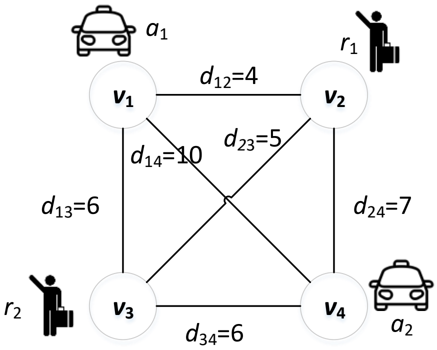
2. 给定n个班会活动$A = \{a_1, a_2, \ldots, a_n\}$，以及两个教室，每个班会活动$a_i$可表示为$[s_i, f_i]$，即开始时间和结束时间。请设计算法安排尽量多的班会活动到两个教室中，使得任意两个安排的班会活动不冲突。（共 15 分）

3. 给定一个正整数数组$A[1,2,\ldots,n]$，现要从中选出一些数，满足数组中任意相邻的 3 个数最多有一个可被选中（即对任意i，$A[i-1], A[i], A[i+1]$三个数最多可被选中一个）。请设计一算法使得选出的数总和最大。（共 10 分）

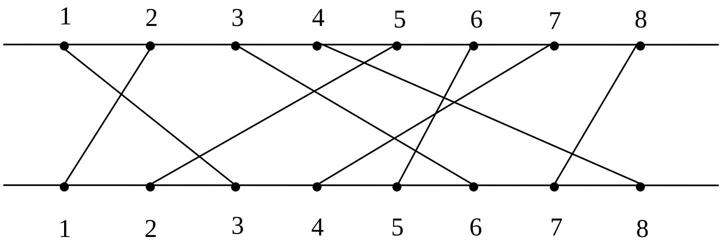
4. 考虑这样一个出租车派单问题：给定一个网络$G = (V, D)$，其中$\forall v_i \in V$表示节点，$D = \{d_{ij}\}, v_i, v_j \in V$表示任意两个节点之间的行程距离，满足三角不等式，即$\forall v_i, v_j, v_k$，有 $d_{ij} + d_{jk} \geq d_{ik}$。假设现在有 n 辆出租车 $A = \{a_1, a_2, \ldots, a_n\}$ 和 m 个乘客 $R = \{r_1, r_2, \ldots, r_m\}$，其中$n \geq m$。每辆出租车$a_k \in A$的节点位置定义为$v(a_k)$，每个乘客$r_l$的位置定义为$v(r_l)$。现在出租车平台公司希望为每个乘客安排一辆出租车，目标是最小化空载距离和，即所有的出租车到乘客的空载距离总和最小。如图所示，有两辆出租车a_1和a_2，他们所在节点分别是v_1和v_4，同时有两个乘客r_1和r_2，他们所在节点分别是v_2和v_3。如果将a_1分配给r_1，a_2分配给r_2，那么空载距离和为：$d_{12} + d_{34} = 10$。
   现在某平台提出一种贪心方案，步骤如下：
   Step1: 对于任意的乘客r_l以及任意的出租车a_k，如果他们之间的距离，$d_{v(a_k)v(r_l)}$最短，则匹配成功，即将出租车a_k分配给乘客r_l；
   Step2：移除 Step 1 中匹配成功的出租车和乘客；
   Step3: 重复 Step 2~3，直到所有的乘客分配完毕。
   如图所示，首先匹配成功的是乘客r_1和出租车a_1，因为他们之间的距离$d_{v(a_1)v(r_1)} = 4$最短。然后匹配成功的是乘客r_2和出租车a_2。
   试问该贪心方法是否为最优方法？如果不是，请给出一个反例并且设计最优方法；如果是，请给出证明。（共 10 分）

姓名______ 学号______

线 封 密



5. 某电路板两侧分别有n个焊点，分别记做焊点$1,2,\ldots,n$，如图所示。根据电路设计图，现在需要将顶层的焊点i $(1 \le i \le n)$与底层的焊点$\pi(i)$用导线联通，即需要n条直线$(i,\pi(i))$ $(1 \le i \le n)$来连接n对焊点。
   两条直线 $(i,\pi(i))$与$(j,\pi(j))$相交，如果$i < j$ 但$\pi(i) > \pi(j)$成立的话。反之亦然。两条直线的交点称为交叉点。如图所示的例子中，一共有 9 个交叉点。
   请设计一个分治算法为任意给定的n对焊点计算总共的交叉点个数。你设计的算法复杂度不能高于$O(n\log n)$。（共 15 分）



6. X 数轴上从左到右有n个不等间距的点$a[1,2,\ldots,n]$，给定一根长度为L的绳子，求绳子最多能覆盖其中的几个点。（共 15 分）
   请设计一个$O(n^2)$时间的算法。
   请问是否存在$O(n)$时间的算法？请尝试说明要点。

7. 假定 0/1 背包问题中，有 3 个背包，每个背包容量分别为C_1，C_2，C_3，给定 n个物品$A = \{a_1, a_2, \ldots, a_n\}$，每个物品$a_i$可表示为$(v_i, w_i)$，即价值和重量。请设计一动态规划方法将物品装入这三个背包，使得每个背包装入物品重量不超过各自容量，且装入物品的总价值最大。（共 15 分）

8. 给定n个物体$A = \{a_1, a_2, \ldots, a_n\}$，请分析随机取出两个物体的概率，并设计一个算法以等概率取出两个物体，给出算法思想及伪代码。（共 10 分）

1 题解 2

1.1 问题分析

该问题是一个典型的贪心算法问题：

- 最优子结构：一个在 $[0, t]$ 时间段内最优的活动安排方案，去掉最后一个活动后，在 $[0, t - t_{\text{最后一个活动}}]$ 时间段内也必然是最优方案
- 贪心选择性：每次必然选择结束时间最早的方案，因为结束时间最早意味着有更多的机会安排其他活动

因此，在每次选择时，只需要在可以在任一教师安排的活动中选择结束时间最早的活动即可。

1.2 算法伪代码

见算法 1。

Algorithm 1: 题解 2 算法伪代码

Function *QuickSort(S)* **begin**
  Input: 班会活动集合 S
  Output: 两个教室中的活动集合 S_1、S_2
  将 S 按结束时间排序;
  for $i \in \{1, 2, 3, ..., |S|\}$ **do**
    if $S[i]$ 开始时间晚于 S_1 最后一个活动的结束时间 **then**
      将 $S[i]$ 安排到 S_1;
    else if $S[i]$ 开始时间晚于 S_2 最后一个活动的结束时间 **then**
      将 $S[i]$ 安排到 S_2;
    end
  end
  return S_1, S_2
end

2 题解 3

2.1 问题分析

该问题是一个典型的动态规划问题：

- 最优子结构：一个选择了 $A[k]$ 且在数组 $[1, k]$ 位内最优的选择方案，，在 $[1, k-2]$ 位内也必然是最优方案
- 重叠子问题：$[1, k]$ 位内最优方案等于下列方案中最优的方案：
  - $[1, k-3]$ 位内的最优方案加 $A[k]$
  - $[1, k-2]$ 位内的最优方案
  - $[1, k-1]$ 位内的最优方案

因此，有最优方案 $F(A[1:k])$ 递推公式：

$$F(A[1:k]) = max\{F(A[1:k-3]) + A[k], F(A[1:k-2]), F(A[1:k-1])\}$$

2.2 算法伪代码

见算法 2。

3 题解 4

3.1 问题分析

首先，该方法不是最优方法。如果 $d_{12} = 2$、$d_{24} = 1$、$d_{34} = 2$，且其余路径长度都满足 $d_{ij} = d_{ik} + dkj$，即点 v_1, v_2, v_4, v_3 一字排开，那么算法首先会将 a_2 分给 r_1，之后会将 a_1 分给 r_2，代价总和为 6，然而最优方案应该是 a_1 分给 r_1、a_2 分给 r_2，代价总和为 4。

该问题是一个典型的二部图最大匹配问题：

- 构建二部图：计算每个出租车到每个乘客之间的距离，作为二部图的边权值
- 计算二部图：计算二部图的最小匹配方案

3.2 算法伪代码

见算法 3。

Algorithm 2: 题解 3 算法伪代码

Function *QuickSort(A)* **begin**

  Input: 数组 A

  Output: 最大的总和值 S

  初始化数组 B，长度和数组 A 相同;

  if 数组 A 长度为 *0* **then return** *0*;

  $B[1] = A[1]$;

  if 数组 A 长度为 *1* **then return** $B[1]$;

  $B[2] = max\{B[1], A[2]\}$;

  if 数组 A 长度为 *2* **then return** $B[2]$;

  $B[3] = max\{B[1], B[2], A[3]\}$;

  if 数组 A 长度为 *3* **then return** $B[3]$;

  for $i \in \{4, ..., |A|\}$ **do**

    $B[i] = max\{B[i-3] + A[i], B[i-2], B[i-1]\}$;

  end

  return $B[|A|]$

end

Algorithm 3: 题解 4 算法伪代码

Function *QuickSort(A)* **begin**

end

4 题解 5

4.1 问题分析

此问题可以通过分治求解，对于上部的焊点 T_i 和下部的焊点 B_i，可以通过如下方式求它们的相交次数：

- 分治：分别递归地计算 $T_i, i \in [1, n/2]$ 和 $T_i, i \in [n/2+1]$ 的相交点个数，并对 $T_i, i \in [1, n/2]$ 和 $T_i, i \in [n/2+1]$ 所连接的下部焊点编号进行排序

- 组合：对于 $T_i, i \in [1, n/2]$，将其对应的上部焊点编号与 $T_i, i \in [n/2+1]$ 对应的上部焊点编号比较，$T_i, i \in [1, n/2]$ 上部编号较大的与 $T_i, i \in [n/2+1]$ 上部编号较小的相交，求出相交数量；对于 $T_i, i \in [n/2+1]$ 同理。最终求出相交点数之和即最终结果。

4.2 算法伪代码

见算法 4。

5 题解 6

5.1 问题分析

显然，对于 n 个点，它们的间距有 $n-1$ 个，将其组织为一个数组 A，原覆盖问题就可以看作选择这个数组中的连续几个值，显然，总共有 $(n-1)+(n-2)+\cdots+2+1 = n(n-1)/2$ 种选法遍历所有这些选法就能得到 $O(n^2)$ 时间的算法。

但实际上这 $n(n-1)/2$ 种选法并不需要全部遍历，可以从第一个间距开始增加，如果长度超过绳长度，就将开头处的距离减去，直到增加到最后一个间距，时间复杂度为 $O(n)$。

5.2 算法伪代码

$O(n^2)$ 时间的算法见算法 5。$O(n)$ 时间的算法见算法 6。

Algorithm 4: 题解 5 算法伪代码

Function $F(T)$ **begin**

  Input: 上部焊点对应的下部焊点编号 T，$T_i = j$ 表示上部焊点 i 与下部焊点 j 相连

  Output: 相交焊点数 S，所有的下部焊点编号排序结果 B

  $S_1, B_1 = F(\{T_i | i \in [1, |T|/2]\})$;

  $S_2, B_2 = F(\{T_i | i \in [|T|/2 + 1, |T|]\})$;

  $S = S_1 + S_2$;

  $B = \{\}, k = 1, t = 0$;

  repeat

    if $B_1[i] > B_2[j]$ **then**

      $B[k] = B_2[j]$;

      $j = j + 1$;

      $t = t + 1$;

    else

      $B[k] = B_1[i]$;

      $i = i + 1$;

      $S = S + t$;

      $t = j - 1$;

    end

    $k = k + 1$;

  until;

  return S, B

end

Algorithm 5: 题解 6 算法伪代码

Function $F(T)$ **begin**
  Input: 点的间距列表 A，绳子长度 L
  Output: 覆盖的最大点数 m
  $m = 0;$
  for $i \in [1, |A|]$ **do**
    $S = 0, k = 1;$
    for $j \in [i, |A|]$ **do**
      $S = S + A[j];$
      $k = k + 1;$
      if $S <= L \wedge k > m$ **then**
        $m = k;$
      end
    end
  end
  return m
end

6 题解 7

6.1 问题分析

此问题可以是典型的动态规划问题。对于物品 a_i 和剩余 C_1、C_2、C_3 的三个背包，背包方案的最优解必然是下面三个中价值最大的那个：

- 物品 a_i 不放入任何背包，C_1、C_2、C_3 中装入剩下的物品
- 物品 a_i 放入背包 1，$C_1 - w_i$、C_2、C_3 中装入剩下的物品
- 物品 a_i 放入背包 2，C_1、$C_2 - w_i$、C_3 中装入剩下的物品
- 物品 a_i 放入背包 3，C_1、C_2、$C_3 - w_i$ 中装入剩下的物品

由此可得最优解 $M(i, C_1, C_2, C_3), i \in [1, n]$ 的递推关系式：

$$M(i, C_1, C_2, C_3) = max \begin{Bmatrix} M(i-1, C_1, C_2, C_3) \\ M(i-1, C_1 - w_i, C_2, C_3) + v_i \\ M(i-1, C_1, C_2 - w_i, C_3) + v_i \\ M(i-1, C_1, C_2, C_3 - w_i) + v_i \end{Bmatrix}$$

Algorithm 6: 题解 6 算法伪代码

```
Function F(T) begin
    Input: 点的间距列表 A，绳子长度 L
    Output: 覆盖的最大点数 m
    m = 0;
    for i ∈ [1, |A|] do
        S = 0, k = 1;
        for j ∈ [i, |A|] do
            S = S + A[j];
            k = k + 1;
            if S > L then
                break;
            end
            if S <= L ∧ k > m then
                m = k;
            end
        end
    end
    return m
end
```

6.2 算法伪代码

见算法 7。

Algorithm 7: 题解 7 算法伪代码

Function M **begin**
  Input: 物品列表 $A = \{(v_i, w_i)|i \in [1, n]\}$，背包容量 C_1, C_2, C_3
  Output: 能装入的物品总价值 M
  使用全局变量 $\boldsymbol{M}$ 存储中间值;
  if $\boldsymbol{M}[i][(C_1, C_2, C_3)]$ 已计算 **then**
    return $\boldsymbol{M}[i][(C_1, C_2, C_3)]$;
  end
  $M = M_1 = M_2 = M_3 = M(i-1, C_1, C_2, C_3)$;
  if $C_1 > w_i$ **then**
    $M_1 = M(i-1, C_1 - w_i, C_2, C_3) + v_i$
  end
  if $C_2 > w_i$ **then**
    $M_2 = M(i-1, C_1, C_2 - w_i, C_3) + v_i$
  end
  if $C_3 > w_i$ **then**
    $M_3 = M(i-1, C_1, C_2, C_3 - w_i) + v_i$
  end
  $M = max\{M, M_1, M_2, M_3\}$;
  return M;
end

7 题解 8

7.1 问题分析

显然，随机取出两个物体的取法有 C_n^2 种，每种概率为 $1/C_n^2$

7.2 算法

先随机取出一个物体，再在剩下的物体中随机取出一个物体。