上海师范大学 Shanghai Normal University

《算法设计与分析》

13-复习 (Review)

杨启哲

上海师范大学信机学院计算机系

2024 年 12 月 21 日

主要内容

- 课程总结
- 考试内容

课程总结

《算法设计与分析》究竟上了什么课?

上海师范大学 Shanghai Normal University

1. 什么是算法?
2. 算法的设计方法。
3. 算法的分析方法。
4. 算法复杂性的基本理论。

什么是算法

上海师范大学 Shanghai Normal University

算法

算法是用于求解某一问题的一系列计算过程，其包含：

- 输入输出。
- 有穷的指令集。

一般来说，我们需要关注算法的终止性、正确性和复杂性。

求解问题

1. 设计一个算法。
2. 算法是正确的么？
3. 算法的复杂性如何？可以有更好的算法么？

算法的设计方法

- 穷举算法.
- 归纳算法
  - 基数排序、多项式求值、生成排列等。
- 分治算法
  - 二分查找、归并排序、快速排序、矩阵乘法、快速傅里叶变换等。
- 贪心算法
  - Dijkstra 算法、Prim 算法、Kruskal 算法、Huffman 编码等。
- 动态规划算法
  - 最长公共子序列、矩阵连乘、背包问题、所有点对最短路径问题等。

在不同数据结构的算法

- 堆和不相交数据集
  - 堆排序、Union–Find 算法。
- 图结构
  - 图的遍历，DFS、BFS。
  - 有向无环图 (DAG) 的拓扑排序、强连通分量算法等。

不同建模下的算法处理

- 流网络
  - 最大流问题、最小割问题，Ford—Fulkerson 算法、MCA 算法、、Edmonds—Karp 算法等。
- 线性规划
  - 单纯形法等。

算法分析方法

- 分析的基本方法
  - 大 O 记号、大 Ω 记号、大 Θ 记号。
  - 时间复杂性、空间复杂性。
- 递归算法的分析
  - 递推式的求解、主定理 (Master Theorem)。

算法复杂性的基本理论

- P 类问题和 NP 类问题
  - 非确定性算法的概念。
- 归约的概念。
  - 归约的方向。
- NP 完全问题。
  - 证明方法。
- 其他复杂性类。
  - co−NP 类问题等。

考试内容

考试概况

上海师范大学 Shanghai Normal University

考试安排

- 考试地点：**奉贤 3 教楼 122**
- 考试时间：**2025 年 1 月 7 日 (周二) 10:45–12:15**，共 90 分钟。

试卷构成

$$\text{试卷} = 20 \text{ 分判断题} + 40 \text{ 分算法设计题} + 40 \text{ 分综合题}$$

- 判断题一共 4 道，每道 5 分，共 20 分。
- 算法设计题一共 3 道，共 $10 + 15 + 15 = 40$ 分。
- 综合题一共 3 道，共 $10 + 15 + 15 = 40$ 分。

判断题举例

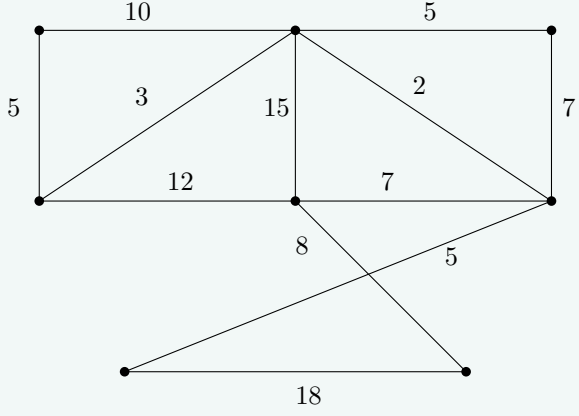
1. 假设某个算法的运行时间满足递推关系 $T(n) = T(n/2) + O(n)$，则该算法的时间复杂度为 $O(n)$。
2. 如果我们能够将一个问题归约到 3-SAT 问题，那么该问题一定是 NP 完全问题。
3. Dijkstra 算法对于有负权边的图是不正确的。
4. Ford – Fulkerson 算法对于有无理数权重的边也可以获得正确的结果。

解 1.

1. 正确。2. 错误。3. 正确。4. 错误。

问题 2.

1. 设计算法求出一个图的最小生成树。
2. 用你设计的算法求出下列图的最小生成树。



解 3.

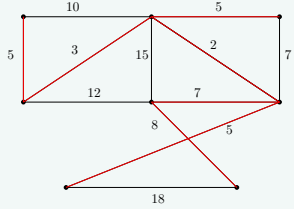
1. 利用 Kruskal 算法:

   输入: 含权连通无向图 $G=(V,E)$, $V=\{1,2,\ldots,n\}$

   输出: G 生成的最小生成树所组成的边集 T

   1: 按非降序的权重 E 进行排序，得到 $E=\{e_1,e_2,\ldots,e_m\}$
   2: $T=\emptyset$
   3: **for** $i=1$ to m **do**
   4: **if** $T\cup\{e_i\}$ 不成圈 **then**
   5: $T=T\cup\{e_i\}$
   6: **return** T

2. 最小生成树如下所示:



综合题举例 (I)

上海师范大学 Shanghai Normal University

问题 4.

QUADEQ 是这样的一类问题。给定 n 个取值为 $0,1$ 的变量 $v_1,\dots v_n$ 和 m 个类似如下的方程：$\sum_{i,j\in[n]} a_{ij}v_iv_j = b$，其中 $a_{ij},b \in \{0,1\}$，这里的加法是在 mod 2 意义下的加法。问是否存在一个解使得所有的方程都成立？

- 证明这是一个 NP 问题。
- 将 3−SAT 问题归约到 QUADEQ 问题，从而证明 QUADEQ 问题是 NP 完全问题。

解 5.

1. QUADEQ 是一个 NP 问题，对于其任何一个解，代入即可验证其是否是使得方程组成立的解，这个过程只需要多项式时间，所以其是一个 NP 问题。
2. 对于 3−SAT 的一个实例公式 f，假设其有 n 个变量 m 个子句，对于每一个子句 $C_i = x \vee \neg y \vee z$，我们构造如下的二次方程：

$$u_{C_{i1}} u_x + u_{C_{i2}}(1 - u_y) + u_{C_{i3}} u_z = 1$$

这里 $u_{C_{ij}}$ 是自由的变元。通过这个变换我们可以将一个 3−SAT 问题转化为一个具有 m 个方程至多 $n + 3m$ 个变量的 QUADEQ 问题实例。这个过程可以在多项式时间内完成。注意到：

- 如果 f 有解，则我们可以对应构造出一个 QUADEQ 问题实例的解。
- 如果 QUADEQ 问题实例有解，则我们可以对应构造出一 f 的解。

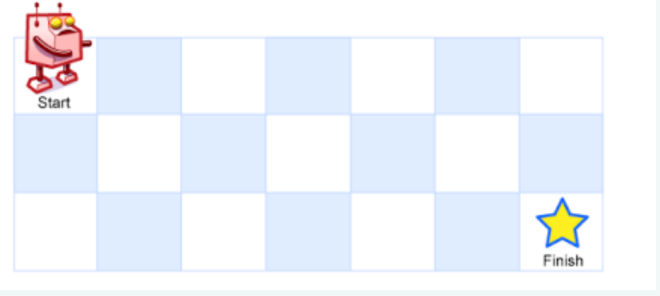
从而我们完成了对 3−SAT 问题到 QUADEQ 问题的多项式时间归约，即证明了 QUADEQ 问题是 NP 完全问题。

问题 6.

一个机器人位于一个 $m \times n$ 网格的左上角（起始点在下图中标记为 Start ）。

机器人每次只能向下或者向右移动一步。机器人试图达到网格的右下角（在下图中标记为 Finish ）。

设计算法求出总共有多少条不同的路径，并给出你的算法的时间复杂性。



解 7.

定义状态 $\mathrm{dp}[\mathrm{i}][\mathrm{j}]$ 表示从起点到达 (i, j) 的路径数目，那么我们有状态转移方程：

$$\mathrm{dp}[\mathrm{i}][\mathrm{j}] = \mathrm{dp}[\mathrm{i}-1][\mathrm{j}] + \mathrm{dp}[\mathrm{i}][\mathrm{j}-1]$$

从而我们的算法可以如下设计：

输入: $\mathrm{m}, \mathrm{n} \in \mathbb{N}^+$

输出: $\mathrm{dp}[\mathrm{m}][\mathrm{n}]$

初始化 $\mathrm{dp}[0][0] = 1, \mathrm{dp}[\mathrm{i}][0] = 1, \mathrm{dp}[0][\mathrm{j}] = 1$
for $\mathrm{i} = 1$ to m **do**
    for $\mathrm{j} = 1$ to n **do**
        $\mathrm{dp}[\mathrm{i}][\mathrm{j}] = \mathrm{dp}[\mathrm{i}-1][\mathrm{j}] + \mathrm{dp}[\mathrm{i}][\mathrm{j}-1]$
return $\mathrm{dp}[\mathrm{m}][\mathrm{n}]$

算法的时间复杂性为 $\mathrm{O}(\mathrm{mn})$，空间复杂性为 $\mathrm{O}(\mathrm{mn})$。

关于答题的一些额外说明 (I)

对于大题的回答，我希望大家给的是一个清晰的思路，而不是一个完整的代码。具体来说，你可以选择使用如下的伪代码形式：

输入: $m, n \in \mathbb{N}^+$
输出: $dp[m][n]$

初始化 $dp[0][0] = 1, dp[i][0] = 1, dp[0][j] = 1$
for $i = 1$ to m **do**
    for $j = 1$ to n **do**
        $dp[i][j] = dp[i-1][j] + dp[i][j-1]$
return $dp[m][n]$

当然对于你中间使用的变量，需要你给出一些解释，比如上面的 $dp[i][j]$ 想表达的意思。

关于答题的一些额外说明 (II)

也可以是如下描述思路的形式：

BFS 可以求出最短路径的长度，我们可以在 BFS 的过程中记录下每个顶点的前驱节点，从而可以得到从 u 到 v 的所有最短路径。具体来说：

1. 对每个节点我们定义一个数组 $P(u)$，初始 $P(u) = 1$，其余都为 0.
2. 从 u 开始对图进行 BFS，与一般不同的是，每次访问到一条边 (s, t) 时，不管是否 t 访问过，更新 $P(t)$ 为 $P(s) + P(t)$.
3. 当某一轮访问到 v 的时候，这一轮全部结束后输出 $P(v)$ 的值。

但是请注意，忽略描述的部分一定是方便实现的，比如有 n 个顶点，你遍历每个顶点；或者说是大家都知道的基本概念，比如 BFS 等，像下述的描述，你需要给出一个更为清晰的思路：

- 枚举图中所有的圈，找出权重最小的那一个。

关于答题的一些额外说明 (III)

上海师范大学 Shanghai Normal University

但我不希望看到的是如下的基于任何一种语言的代码：

```cpp
int lengthOfLongestSubstring(string s) {
    // 哈希集合，记录每个字符是否出现过
    unordered_set<char> occ;
    int n = s.size();
    // 右指针，初始值为 -1，相当于我们在字符串的左边界的左侧，还没有开始移动
    int rk = -1, ans = 0;
    // 枚举左指针的位置，初始值隐性地表示为 -1
    for (int i = 0; i < n; ++i) {
        if (i != 0) {
            // 左指针向右移动一格，移除一个字符
            occ.erase(s[i - 1]);
        }
        while (rk + 1 < n && !occ.count(s[rk + 1])) {
            // 不断地移动右指针
            occ.insert(s[rk + 1]);
            ++rk;
        }
        // 第 i 到 rk 个字符是一个极长的无重复字符子串
        ans = max(ans, rk - i + 1);
    }
    return ans;
}
```

这种回答的风险是如果你的代码上有细节问题导致我没理解，你的回答可能会被扣分。

一个完课的调查

最后，这是关于这门课完课的一个调查问卷，希望同学们可以积极参与，表达大家的想法和建议，你们的意见能够帮助我更好的改进这门课程。谢谢!

- 《算法设计与分析》完课调查
- 问卷的二维码:

祝大家考试顺利！提前祝大家新年快乐！