期末复习题3-4

詹博华　2021/11/17

问题#3

- 我们称一个子序列$y_1, y_2, \ldots, y_n$为交替序列，如果每个相邻的y_i, y_{i+1}, y_{i+2}都满足$y_i < y_{i+1} > y_{i+2}$或$y_i > y_{i+1} < y_{i+2}$。也就是说，如果$y_i < y_{i+1}$，则$y_{i+1} > y_{i+2}$，反之亦然。给定一个序列$x_1, x_2, \ldots x_n$，我们想要找到这个序列的最长的交替子序列。
- 例如，如果初始序列是：

$$x_1 = 13, x_2 = 93, x_3 = 86, x_4 = 50, x_5 = 63, x_6 = 4,$$

这个序列的最长交替子序列的长度为5，由$13, 93, 50, 63, 4$组成。

- 使用动态规划设计寻找最长交替子序列的算法。

问题#3解答

- 我们定义子问题$DP(i,b)$，其中$1 \leq i \leq n$，b是一个布尔值。如果b为真，则$DP(i,b)$代表最长的终止于x_i的交替子序列，其中最后一步是上升的。如果b为假，$DP(i,b)$代表最长的终止于x_i的交替子序列，其中最后一步是下降的。如果子序列的长度为1，则我们定义它既是上升的也是下降的。例如$DP(5,\mathrm{True}) = 4$，因为子序列x_1, x_2, x_4, x_5的长度为4，并且最后一步上升。$DP(5,\mathrm{False}) = 3$，因为子序列x_1, x_2, x_5的长度为3，并且最后一步下降。

问题#3解答

a. 对于以上序列，计算$DP(i,b)$，对于每个$1 \le i \le 6$和$b \in \{\text{True}, \text{False}\}$，写入以下表中：

$$x_1 = 13, x_2 = 93, x_3 = 86, x_4 = 50, x_5 = 63, x_6 = 4$$

	$i = 1$	$i = 2$	$i = 3$	$i = 4$	$i = 5$	$i = 6$
$b = \text{True}$	1	2	2	2	4	1
$b = \text{False}$	1	1	3	3	3	5

问题#3解答

b. 写出$DP(i,b)$的递归关系。

$$DP(i, \textsc{True}) = 1 + \max_{1 \le j < i \text{ and } x_i > x_j} DP(j, \textsc{False})$$
$$DP(i, \textsc{False}) = 1 + \max_{1 \le j < i \text{ and } x_i < x_j} DP(j, \textsc{True})$$

c. 写出$DP(i,b)$递归关系的基本情况（$i = 1$）。

$$DP(i, \textsc{True}) = 1 \qquad \text{if } x_i = \min\{x_1, \ldots, x_i\}$$
$$DP(i, \textsc{False}) = 1 \qquad \text{if } x_i = \max\{x_1, \ldots, x_i\}$$

问题#3解答

d. 如果采用自底向上的计算方式，给出一个合适的计算顺序。

   按照i从小到大的顺序，计算每个$DP(i, \text{True})$和$DP(i, \text{False})$。

e. 如果计算了所有的$DP(i, b)$，如何计算最长交替子序列的长度？

   需要取表中所有值的最大值。

问题#3解答

f. 将以上步骤放到一起，给出计算最长交替子序列长度的算法，使用伪代码表示。分析算法的时间复杂度。

for i = 1 **to** n
  for b = {True, False}
    if $DP(i,b)$ in base case:
      compute $DP(i,b)$ using c)
    else
      compute $DP(i,b)$ using b)
return maximum value in $DP(i,b)$

复杂度分析：
由于b)和c)中的计算需要线性时间，整个计算需要$O(n^2)$时间。

问题#4

- 给定一个不带括号的，由加法和乘法组成的表达式，找出如何添加括号，使得表达式的取值最大。
- 例如，如果提供的表达式是$6 + 0 \cdot 6$，则应该添加括号为$(6 + 0) \cdot 6$，结果为36，而不是$6 + (0 \cdot 6)$，结果为6。再比如，如果提供的表达式是$0.1 \cdot 0.1 + 0.1$，则应该添加括号为$(0.1 \cdot 0.1) + 0.1$，结果为0.11，而不是$0.1 \cdot (0.1 + 0.1)$，结果为0.02。
- 使用动态规划设计多项式时间的算法，在给定表达式之后，找到最好的添加括号的方法。假设输入的格式为$x_0, o_0, x_1, o_1, \ldots o_{n-1}, x_n$，其中每个$x_i$是一个数字，每个$o_i$是加号或乘号。分析算法的时间复杂度。

问题#4解答

- 设$DP[i,j]$为加括号后$x_i, o_i, \ldots x_j$的可能最大值。
- 递归关系为：

$$DP[i,j] = \max_{k=i}^{j-1} \Big(DP[i,k]\ o_k\ DP[k+1,j] \Big).$$

- 基本情况为：

$$DP[i,i] = x_i.$$

- 计算顺序可以按照$j-i$从小到大的方式。

问题#4解答

- 伪代码

```
for i = 0 to n
  DP[i,i] = x_i
for l = 0 to n
  for i = 0 to n − l
    j = i + l
    compute DP[i,j] using recurrence
return DP[0,n]
```

- **时间复杂度：** 每个子问题的计算$O(n)$，一共$O(n^2)$个子问题，因此一共$O(n^3)$。

问题#4解答

- **出题时忽略的一个细节：** 以上答案假设所有涉及的数字都是正数，因此最大化$a+b$和$a\times b$只需要最大化a和b。但如果a和b可能是负数，以上假设不成立。
- **解决方案：** 除了维护$DP_{\max}[i,j]$为$x_i,o_i,\ldots x_j$加括号后的最大值以外，还维护$DP_{\min}[i,j]$为$x_i,o_i,\ldots x_j$加括号后的最小值。在递归关系中，考虑$DP[i,k]$和$DP[k+1,j]$分别为最大值和最小值的四种情况。算法的时间复杂度依然是$O(n^3)$。