作业 6

Algorithm 1 问题 1

实现思路是: 贪心算法，从左至右考虑可选择的 interval ，如果一次选择中有多个可选择的 interval ，则选择 $R[i]$ 最大的

伪代码如下:

Let $option = [0, 0]$,

while $option.second \neq X.right$ **do**

  $temp = []$

  for i in L **do**

    if $L[i] < option.second$ **then**

      $temp.append(i)$

    end if

  end for

  Select i for i in $temp$ such that $max(R[i])$

  $option = [0, R[i]]$

end while

正确性证明：首先一定要在 option 的范围选，不然无法覆盖 X，要证明的是每次都要选 R[i] 最大的点，显然若在某次选择中，不选择 R[i] 最大的点 a，则存在 R[i] 最大的点 b，则 a，b 最有端点可能有空隙，空袭里的点被 b 考虑不被 a 考虑，所以 a 的选择会更少，所以选 b 一定优于 a

时间复杂度：实际上，可以先对 L 和 R 排序，这样每次循环只要不满足 option 的条件就退出，最多只需要考虑每个 interval 一次，为排序的时间复杂度 O（nlgn）

Algorithm 2 问题 2

先对 R 排序，然后从左向右考虑，每次选择 R 中最小的 interval（i），以 R[i] 设置一个点，然后从删除 R 中所有符合条件的点，循环直到 R 为空

对 R 按照右端点升序排序对数组 R 进行排序

$points \leftarrow []$ 初始化空的点集

while R 不为空 **do**

  $i \leftarrow$ 最小的 interval 在 R 中的索引选择 R 中最小的 interval

  在 $R[i]$ 的右端点处添加一个点 p 设置点 p 在 $R[i]$ 的右端点处

  $points \leftarrow points \cup \{p\}$ 将点 p 添加到点集 $points$

  从 R 中删除所有与 $R[i]$ 相交的 interval 删除与 $R[i]$ 相交的 interval

end while

return $points$ 返回点集 $points$ 作为最小点集

正确性证明: 显然如果排序后, 以本算法划分出的点集 set1, 若存在更优解 set2, 考虑其中任意一个点 b, 选 set1 中的点 a, 使 a>b 且 a 尽可能小, 即 set1 中最小的在 b 右边的点. 显然,b 能覆盖的点 a 也能覆盖, 但是可能存在 a 覆盖的点 b 不能覆盖, 所以选 a 一定优于 b

时间复杂度: 排序后考虑每一个点只要考虑一次, 所以为排序的时间复杂度 O(nlgn)

Algorithm 3 问题 2

问题 1: 初始化 cnt = 0, 遇到左括号 cnt+1, 右括号 cnt-1, 从左至右遍历 w, 过程中始终有 cnt>=0 且最后 cnt==0 则 w is balanced, 反之相反

证明: 若一个 w 为 imbalanced, 则一定存在数量不匹配的左右括号, 存在多余的左括号, 会在最终由 cnt>0 , 存在多余的右括号, 在中途由 cnt<0 都被会算法检测

问题 2: 查看 w 是否为 (x) 的形式, 即检查 w[1] 和 w[-1], 若满足, 则 ans = len(w)-2 否则,w 为 xy 形式, 利用问题 1 的算法, 找出第一个满足 cnt=0 的点, 从 0 到这个点则为 x, 则 ans = max(len(x),len(w)- len(x))

证明: 算法自证