研究生算法课课堂笔记

上课日期：2014.12.8　　第(3)节课

组长：赵玮泽　　组员：刘欢欢

组员：吴昊泽　　组员：罗杨成

注意：请提交 Word 格式文档。

2014.12.8 第三节课笔记

作业的第二题其实解法很多，但同时符合下述三点的解法并不多：

1. 时间复杂度：$\theta(n^2)$
2. 空间复杂度：$\theta(n)$
3. 能够反向追踪（作业题中不要求，但希望同学们掌握）

解法一：

一维动归＋遍历

opt(j) 表示前 j 天当中最优的数据处理量，遍历的是最后一次 reboot 是哪一天；

w(i+1, j): 表示第 i+1 天到第 j 天连续工作的数据处理量。

状态转移方程如下：

$opt(j) = \max\{opt(i-1) + w(i+1, j)\}$ ←（也可以在此处区分一下是否有过 reboot，两种情况取 max）

其中 $1 \le j \le n$, $1 \le i < j$,

$opt(0) = 0$，

$opt(1) = \min\{x_1, s_1\}$，

$g(j) = i$,　// g 数组记录何时 reboot

return opt(n)

重点在于怎么计算 w(i+1, j)：

i. 每算一遍 w(i+1, j)时间代价是 O(n)，总的时间复杂度将是 $O(n^3)$；
ii. $w(i+1, j) = w(i+1, j-1) + \min\{x_j, s_{j-i}\}$，计算 w(i+1, j)时间代价降到单位时间，总时间复杂度是 $\theta(n^2)$，但空间复杂度也是 $\theta(n^2)$。

解法二：

反向遍历的一维动归。

令 opt（i）表示从第 i 天起到第 n 天的数据处理量，并且在第 i-1 天重启过。第 i 天之后最近的一次重启是在第 j 天。

状态转移方程如下：

$opt(i) = \quad max_{1\leq j\leq n} \{ opt(j+1) + w(i , j-1) \}$

其中 $i <= j < n$

$w(i , j)$ 表示从第 i 天到第 j 天的数据处理量，如果 $j<i$ ，那么 $w(i , j) = 0$。

状态转移方程表示：如果第 i-1 天重启过，且第 i 天以后最早的一次重启是在第 j 天（j>=i），那么从第 i 天到第 n 天的数据处理量 opt(i)等于第 j+1 天到第 n 天的数据处理量 opt(j+1)加上第 i 天到第 j-1 天的数据处理量 w(i,j-1)。遍历 j 使得 opt(i)取值最大。

初始值：$opt(n) = min\{ x_n, s_1 \}$

返回值：opt(1)

时间复杂度：i 和 j 构成两层嵌套的循环，所以时间复杂度为 $O(n^2)$。

空间复杂度：使用一个一维数组来保存 opt(i)的值，所以空间复杂度为 O(n)。

这里需要说明的是 w(i , j-1)并不需要额外的数组来记录，也不需要额外的循环来计算。因为在对 j 迭代的过程中，opt(i)只依赖 w(i, j-1)，使用一个变量记录该值即可。示例代码如下：

```
for i = n to 1:
        ……
        val = min{ x_i , s_1 }
        for j = i+1 to n-1:
                val += min{ x_j , s_{j-i+1} }
                ……
        End for
End for
```

反向追踪：通过反向追踪能够还原出具体在哪些天重启了电脑。w(i, j-1)在反向追踪的过程中以上述代码相同的方式计算出来，不需要额外的空间记录该值。

解法三：

$opt(i,j)$ 定义为从开始到第 i 天最优的数据处理量，假设 j 天前重启过。

$$opt(i,j)=max\begin{cases} opt(i-1,j-1)+min\{x_i,s_j\}, j\neq 0 \\ max_{0<k<=i-1}\{opt(i-1,k)\}, j=0 \end{cases}$$

范围：$\begin{array}{l} 1\le i\le n \\ 0\le j\le i \end{array}$

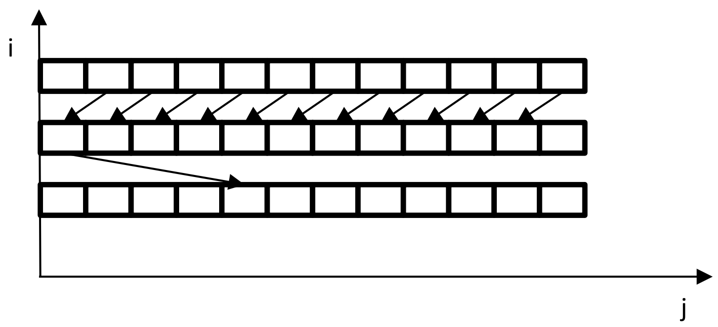
初始条件：$opt(1,j)=min\{x_1,s_1\}$

返回值：$max_{0<j\le n}\{opt(n,j)\}$

时间复杂度为：$O(n^2)$

空间复杂度为：$O(n)$

依赖关系：（见下图）$j>0$ 时，向左下角对角线依赖；$j=0$ 时，在下面一行中选择除了第一个之外最大的。



追踪方式：可以使用滚动数组优化，因为只依赖下面一行，所以保留两行即可。这样的话反向追踪需要额外的一个数组，记录上图中的依赖关系即可。

解法四：

$opt(i,j)$ 定义为从第 i 天到第 n 天的最优解，假设在 j 天前重启：

转移方程：

$$opt(i,j)=\max\begin{cases} opt(i+1,j+1)+\min\{x_i,s_j\} \\ opt(i+1,1) \end{cases}$$

范围：$1\le i\le n, 0\le j\le i$ ，定义 $s_0=0$

初始化：

$opt(n,j)=\min(x_n,s_j)$

返回结果：

$opt(1,1)$，有人认为应返回 $\max(opt(1,1),opt(1,0))$，实际上 $opt(1,1)$ 包含了 $opt(1,0)$ 的情况，因为 $opt(1,1)=\max\{opt(2,1),opt(2,2)+\min(x_1,s_1)\}$，而 $opt(2,1)$ 和 $opt(1,0)$ 表示相同的情况。

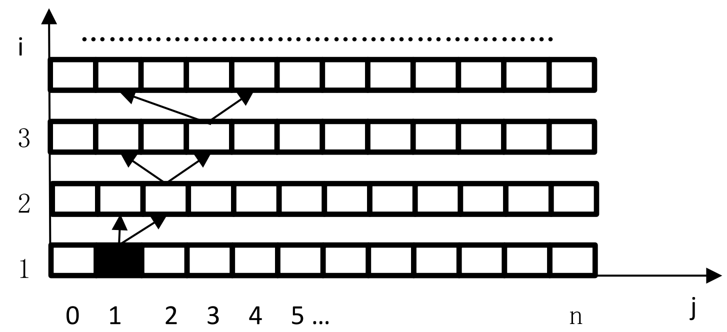
时间复杂度：

$O(n^2)$

空间复杂度：

$O(n)$ 由于求解 $opt(i,j)$ 只需 $i+1$ 行的信息，从而可以用滚动数组实现

依赖关系：如下图



追踪方法：

我们只需记录 $opt(i,1), i=1,2,...,n$，

对于第一天，我们比较$opt(1,1)$和$opt(2,1)$如果，两个相等，说明第一天重启，工作量为 0；

否则，$opt(1,1)$由$opt(2,2)$而来，第i天工作量为$\min(x_1, s_1)$

假设第i我们追踪到$opt(i, j)$，比较它和$opt(i+1,1)$，如果相等，说明第i天工作量为 0；

否则，$opt(i, j)$由$opt(i+1, j+1)$而来，第i天工作量为$\min(x_i, s_j)$