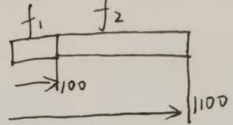


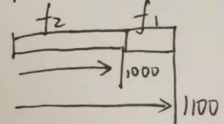
13.2 d

14.2 b

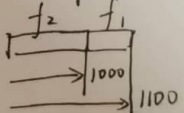
14.3 c

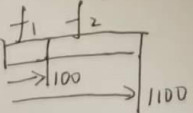
4.

4. (1). $f_1: \square S_1=100 \quad p_1=0.01$
 $f_2: \square S_2=1000 \quad p_2=0.99$
长度由小到大排列: 
$$T(\pi) = 100 \times 0.01 + 1100 \times 0.99 = 1090$$

若但排列: 
$$T(\pi)' = 1000 \times 0.99 + 1100 \times 0.01 = 1001 < T(\pi)$$

故长度由小到大不是最优排列

(2). $f_1: \square S_1=100 \quad p_1=0.1$
 $f_2: \square S_2=1000 \quad p_2=0.9$
访问概率由高到低: 
$$T(\pi) = 1000 \times 0.9 + 1100 \times 0.1 = 1010$$

但排列: 
$$T'(\pi) = 100 \times 0.1 + 1100 \times 0.9 = 1000 < T(\pi)$$

故不是最优排列

(3) 算法: 首先计算每个文件的长度和访问概率的商, 然后对序列按照 s_i/p_i 升序排序。

正确性证明: 令 G 为贪心算法产生的调度, 令 O 为任意最优调度。如果 $G=O$, 那么贪心是最优的, 我们就完成了。否则, O 必须包含至少一个反转, 即至少有

一对未按 s_i/p_i 升序排序调度的文件，记为 f_i 、 f_j 。也就是说，设 f_i 和 f_j 是调度 O 中的前两个连续任务，使得 $s_j/p_j < s_i/p_i$ 。设 i 文件前有长度 s ，则 O 的总成本 $T_1 = (s+s_i)*p_i + (s+s_i+s_j)*p_j$ 。在 O 中交换 f_i 和 f_j ，交换后的总成本 $T_2 = (s+s_j)*p_j + (s+s_i+s_j)*p_i$ ， $T_1 - T_2 = s(p_j - p_i) > 0$ ，即交换后成本会降低，是更优的策略。以此类推，贪心算法 G 就是最优策略

运行时间： $O(n \log n)$ ，为排序所需时间复杂度

5. (a) 假设当前存在一个刺穿集，其中存在不是某个区间的右端点(b_i)的刺穿点。则将第一个不是某个区间的右端点(b_i)的刺穿点 x_i 变为 x_i' ，使其位于 x_i 所刺穿的区间中最小的右端点 b_i 处，则 x_i' 没有超出任何一个 x_i 所刺穿的区间，并且还可能会因为经过某个区间的左端点而使刺穿的区间数增加。则相应的刺穿集中点的个数要么不变，要么因为 x_i' 刺穿的区间增加而减少。以此类推，对于任何一组区间，都存在一个最小尺寸的刺穿集，其中每个刺穿点都是某个区间的右端点(b_i)。

(b) 算法思想：将所有右端点按从小到大的顺序排序，然后每两个右端点合为一组，选择组内深度更大的端点作为该组的端点(若深度相等则选择更小的端点)，得到 $n/2$ 个端点，然后再两两分组以此类推，最终得到深度最大的端点。

MD 启发式算法不是最优的

$$(c) T = n \log n + (k-1)n/k * \log((k-1)n/k) + \dots + 2n/k * \log(2n/k) + n/k * \log(n/k)$$

(d)

