San Skulrattanakulchai

Gustavus Adolphus College

Oct 16, 2015

Weighted Activities Scheduling

GT: Ch 12.3

Problem

We are given n intervals (activities) numbered by the positive integers from 1 to n. Each interval i has an integral start time s_i, an integral finish time f_i, and a positive integral benefit (weight) b_i. Each interval $(s_i, f_i]$ is open at its lower bound and closed at its upper bound. A non-overlapping subset of the given intervals is called a *schedule*. An *optimum schedule* is a schedule that maximizes the total benefit. We wish to find an optimum schedule.

Example

Let intervals (0, 4], (1, 3], (2, 4], and (3, 6] be given with corresponding benefits of 1, 2, 2, and 3 units, respectively. An optimum schedule chooses intervals (1, 3] and (3, 6], achieving a total benefit of 2+3 = 5 units.

Input Format

We assume data is input as three arrays: the start times $S[\cdot]$, the finish times $F[\cdot]$, and the benefits $B[\cdot]$, each of length n. For the time being, assume the finish times are in non-decreasing order, that is, $F[1] \le F[2] \le F[3] \le \cdots \le F[n]$. For each interval i, where $1 \le i \le n$, define the predecessor of i, written p_i to be

$$p_i := \max\{\, 0, j : 1 \le j < i \text{ and interval } j \text{ does not overlap with interval } i \,\}.$$

Recurrence

Define $m(i)$, for all $1 \le i \le n$, to be the maximum benefit achievable by scheduling intervals 1 through i only. For convenience we set $m(0) := 0$.

Fix i. Consider an optimum schedule S using only intervals 1 through i. Schedule S either includes interval i or it doesn't. If it does, then all intervals from p_i+1 through $i-1$ cannot belong to S since all these intervals overlap with interval i, and $S \setminus \{i\}$ must be an optimum schedule that achieves $m(p_i)$. If it does not, then S is also an optimum schedule

that achieves $m(i-1)$. One of these two cases must hold, and by definition of $m(i)$ it must be true that $m(i) = \max\{\, m(p_i) + b_i,\ m(i-1)\,\}$.
Therefore, $m(i)$ satisfies the recurrence

$$m(i) = \begin{cases} 0 & \text{if } i = 0, \\ \max\{\, m(p_i) + b_i,\ m(i-1)\,\} & \text{if } 1 \leq i \leq n. \end{cases}$$

Algorithm

Step 1 Sort the intervals so that the finish times are in non-decreasing order.

Step 2 Compute p_i for all $1 \leq i \leq n$ and store these predecessor values in a table.

Step 3 Fill in the table of $m(\cdot)$ values, from left-to-right (i.e., from 0 to n).

Step 4 Retrieve an optimum schedule from the information stored in the $m(\cdot)$ table and the predecessors table.

Running Time

Step 1 takes time $O(n \log n)$ using an efficient sorting algorithm.

Step 2 takes time $O(n \log n)$, e.g., by binary search.

Step 3 takes time $O(n)$.

Step 4 takes time $O(n)$.

Therefore, time for the whole algorithm is $O(n \log n)$.

Notes

1. We present this problem in a general setting. It applies whenever we have one shareable resource that can be used by only one activity at any given point in time. The corresponding problem in GT is the Telescope Scheduling problem.

2. A special case of this problem where all the interval benefits are equal can be solved by a greedy algorithm.