Task Scheduling Problem

November 15, 2020

Introduction

- Also called interval scheduling problem
- Motivated by applications in resources scheduling
- We are given a set $\mathcal{S} = \{a_1, a_2, \ldots, a_n\}$ of n activities that are to be scheduled on some resource, which can serve only one activity at a time. Each activity a_i has a time interval specified by *start time* s_i and a *finish time* f_i. Two activities are *compatible* if their intervals doesn't overlap. The problem is to schedule as many compatible activities as possible on the resource.

An example

Consider the following set $\mathcal{S}$ of activities with their start and finish times.

a_i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	9	9	10	11	12	14	16

The subset $\{a_3, a_9, a_{11}\}$ consists of mutually compatible activities, but it is not a maximum subset.

The subset $\{a_1, a_4, a_8, a_{11}\}$ is a largest subset of mutually compatible activities.

Another largest subset is $\{a_2, a_4, a_9, a_{11}\}$.

The greedy algorithm

Which of the following parameter can be used for greedy choice?

1. Start time
2. Finish time
3. Shortest activity ($f_i - s_i$)

A few trials show that the finish time is the best.

Exercise. Show that the other parameters may not generate optimal solutions (provide counter examples)

The greedy algorithm

1. Sort the activities by finish time and set $\mathcal{R} = \phi$
2. Pick the first activity a_i in the sorted list and add to $\mathcal{R}$
3. Remove all activities that are not compatible with a_1
4. Repeat steps 3 and 4 until the list is finished

Consider the following activities again.

a_i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	9	9	10	11	12	14	16

The list is already sorted in the increasing order of finish time. We add a_1 to R and delete all incompatible activities $(a_2,\ a_3,\ a_5,\ a_{10})$. Then add a_4 to R and delete the incompatible activities $(a_6,\ a_7)$, then add a_8 and delete a_9, and finally add a_{11} to R.

Optimal substructure and greedy choice

1. **Greedy choice** : There exists an optimal solution containing the greedy choice (a_1).

   Let R be an optimal solution and if R contains a_1, we are done. otherwise let a_k be the first activity in R, then we can remove a_k and safely add a_1 because $f_1 \leq f_k$. The new R is also optimal .

2. **Optimal substructure**: The optimal solution can be made from the greedy choice plus an optimal solution to the remaining sub problem.

   Proof is left as an exercise.