Network Flow For Scheduling?

Muhammad Islam

December 10, 2022

1 Introduction

Testing

2 Background

The topic explored in this paper is inspired by the real-world problem of time management experienced by most ordinary people including students, employees, etc. Usually, an individual has several tasks or assignments which they wish to complete by a certain time called a deadline, while attending to several events such as sleep, lunch, dinner, classes/professional meetings, social gatherings, etc. Since events are unavoidable with fixed starting times and ending times, the time an individual has to complete tasks and assignments is limited to the time unoccupied by events called "free time". There are two main aspects of this problem that are worth noting:

- The time needed to complete each task, called its "duration", need not be the same.
- The deadlines of each task need not be the same.

These two aspects are important because if they didn't hold true the solution to this problem would have been simple: divide the free time available before the deadline by the number of tasks and allocate the resulting time for each task, with the order in which the tasks are performed being irrelevant. However, because not all tasks take the same amount of time to complete, an equal amount of time wouldn't be ideal. Of course, the solution to the problem would still be trivial: assign a weight corresponding to the time needed for each task divided by the minimum amount of time needed for a task out of all tasks; divide the free time available before the deadline by the total weight of all tasks; then finally allocate time for each task corresponding to the product of the weight of the task and the previous result.The order in which the tasks are performed would also be irrelevant in this situation.

Yet, because not all tasks have the same deadline, not only would the order matter but there is no single fixed amount of free time. For tasks with earlier deadlines, the amount of free time is less than tasks with later deadlines even if the task with the earlier deadline may need more time to complete.

I formalize the problem as follows:

□

Table 1: Figure 1

Define a task to be a record with a name, a duration, m, and a deadline f and an event to be a record with a name, start time, and end time. Given a set of tasks T of size n sorted by their deadlines, a set of events E, and a start date, d_0, allocate time, l, for each task, t_i, such that $\frac{l}{m}$ is maximized for all tasks.

Before examining this problem, I will first demonstrate an equivalent graphical representation of the problem, comparing it with that of a flow network. I will then explore a variant of the problem in which l is maximized rather than $\frac{l}{m}$. I will then finally propose an algorithm for the problem and generalize it to similar networks.

3 Findings

3.1 Representation of the Problem

To represent this problem, we can first calculate the amount of free time for each $t_i \in T$ between t_i and t_i other than the first task. For the first task, t_1, the amount free time is calculated between the start date and t_i. Let D represent the set of all of these amounts. We can make a square n by n table where the columns are for D and the rows are for T. Observe that for each D_i, we can only allocate tasks for all $t_j \mid j \geq i$.

4 Aspect 2 in detail