Information Processing Letters 47 (1993) 165–166
Elsevier

14 September 1993

A note on "Scheduling unit-time tasks with integer release times and deadlines" *

George Steiner and Scott Yeomans

Management Science and Information, Systems Area, Faculty of Business, McMaster University, Hamilton, Ontario, Canada

Communicated by F. Dehne
Received 3 January 1993

Abstract

Steiner, G. and S. Yeomans, A note on "Scheduling unit-time tasks with integer release times and deadlines", Information Processing Letters 47 (1993) 165–166.

Frederickson considered the problem of scheduling n unit-time tasks on m processors in which the tasks have integer release times and deadlines. Using appropriate data structures, he presented an $O(n)$ time algorithm for the problem. However, it can be shown that this algorithm could conceivably construct an infeasible schedule. We show how this problem can be resolved while the linearity of the algorithm is preserved.

Keywords: Analysis of algorithms; deadlines; release times; scheduling; unit-time tasks

1. Introduction

Frederickson [1] considers the problem of non-preemptively scheduling n unit-time tasks, T_1, $T_2, \ldots, T_n$ on m identical processors, where each task has an integer release time r_i and an integer deadline d_i, such that no task starts prior to its release time or completes beyond its deadline. The problem can be solved by scheduling according to the earliest deadline rule (see [2]) and Frederickson demonstrates how this solution technique can be implemented to run in $O(n)$ time. However, Frederickson does not account for a possible infeasibility condition and the algorithm could potentially construct a schedule which is not feasible. A test for this condition can be added to the algorithm and the modification of either determining the maximum number of tasks which may be feasibly scheduled or declaring that no feasible schedule exists could be incorporated into the procedure in constant time; thereby preserving its linearity.

Correspondence to: G. Steiner, Michael G. DeGroote School of Business, McMaster University, 1280 Main Street West, Hamilton, Ontario, Canada L8S 4M4.

* This research was supported in part by the Natural Sciences and Engineering Research Council under grant A1798.

2. Frederickson's scheduling algorithm

Frederickson's algorithm consists of three phases.

The first phase is a sorting phase using a table and stack structure. It associates with each meaningful location r' in the table a list of tasks with release time r'.

In the second phase, the schedule is partitioned into *compact sections* where each section starts at some release time, r', and extends until the number of tasks released during the section is less than or equal to the number of time units available within the section.

In the third phase, each compact section is treated as an individual scheduling problem and the release times and deadlines of the tasks within each section are transformed and reindexed so that the first release date occurs at time 0 and the transformed deadlines, $\hat{d}_i$, are such that $\hat{d}_i < \hat{d}_j$ implies that $i < j$. A sequence of INSERT(i) and EXTRACT-MIN instructions is then constructed and executed for an off-line minimum problem and the schedule is extracted (see [1] for more details).

The number of EXTRACT-MIN instructions equals the number of tasks within the compact section. INSERT instructions are placed in front of the appropriate EXTRACT-MIN instructions to correspond to when the various tasks are released. If task T_i is removed by the jth EXTRACT-MIN instruction, then it will be scheduled in time step $r' - 1 + \lceil j/m \rceil$ (where r' is the time at which the compact section starts) on processor $1 + (j - 1) \bmod m$.

It is possible, however, that even though the number of release times within a compact section is greater than or equal to the number of tasks, that the deadlines of a subset of the jobs in the compact section would be such that the cardinality of this subset exceeds the number of starting times available and that no feasible schedule is possible. This can be best be demonstrated using a simple example.

Assume that there are five tasks to be scheduled on a single processor, where each task, i, is denoted by the triple (i, r_i, d_i). These tasks are: (1, 1, 2), (2, 1, 3), (3, 1, 3), (4, 2, 5) and (5, 4, 6). Using the method of Frederickson, phase 1 and phase 2 will place all of these tasks into the same compact section. The instruction sequence for the applicable off-line minimum problem is: INSERT(1), INSERT(2), INSERT(3), EXTRACT-MIN, INSERT(4), EXTRACT-MIN, EXTRACT-MIN, INSERT(5), EXTRACT-MIN, EXTRACT-MIN, which will construct the schedule 1-2-3-4-5. This schedule is not feasible since task 3, which has a deadline of 3, will complete at time 4. The infeasibility occurs because tasks 1, 2 and 3 all have release time 1 and must all be completed by time 3, thereby requiring that three tasks be scheduled into two starting times.

3. Modification of Frederickson's scheduling algorithm

Frederickson's algorithm can be modified in one of two ways. Firstly, if the above infeasibility condition is encountered then the algorithm could halt, declaring that no feasible schedule exists for the set of tasks. Alternatively, if the infeasibility condition is encountered, the algorithm could be modified to construct a sequence for the maximum number of jobs which can be feasibly scheduled. Define an EXTRACT-MIN instruction to be *feasible* if the task selected by it can be scheduled prior to its deadline. Then these modifications can be implemented in the following way.

Modification 1 would stop the algorithm if an infeasible EXTRACT-MIN were encountered. That is, if the first $j - 1$ EXTRACT-MIN instructions are feasible and the index i selected by the jth EXTRACT-MIN is such that $d_i < \lceil j/m \rceil$, then the jth EXTRACT-MIN is not feasible since task T_i cannot be feasibly scheduled.

Modification 2 would state that if index i is removed by the jth feasible EXTRACT-MIN, then, letting $k = \max\{\lceil j/m \rceil, r_i\}$, task T_i will be scheduled in time $r' - 1 + k$. Conversely, if index i is removed by an EXTRACT-MIN which is not feasible, then task T_i will not be scheduled.

Either of the above modifications could be performed in constant time, thus preserving the linear time nature of Frederickson's original algorithm.

References

[1] G. Frederickson, Scheduling unit-time tasks with integer release times and deadlines, *Inform. Process. Lett.* **16** (4) (1983) 171–173.

[2] M. Garey, D. Johnson, B. Simons and R. Tarjan, Scheduling unit-time tasks with arbitrary release times and deadlines, *SIAM J. Comput.* **10** (2) (1981) 256–269.