Exact Algorithms for the General Coupled Task Scheduling Problem*

[Extended Abstract]

József Békési
Department of Applied Informatics, Gyula Juhász Faculty of Education University of Szeged
H-6701 Szeged, POB 396, Hungary
bekesi@jgypk.u-szeged.hu

Gábor Galambos
Department of Applied Informatics, Gyula Juhász Faculty of Education University of Szeged
H-6701 Szeged, POB 396, Hungary
galambos@jgypk.u-szeged.hu

Michael N. Jung
Institut für Informatik, Fakultät für Mathematik und Informatik Universität Heidelberg
Im Neuenheimer Feld 368, D-69120 Heidelberg, Germany
michael.jung@iwr.uni-heidelberg.de

Marcus Oswald
Institut für Informatik, Fakultät für Mathematik und Informatik Universität Heidelberg
Im Neuenheimer Feld 368, D-69120 Heidelberg, Germany
marcus.oswald@iwr.uni-heidelberg.de

Gerhard Reinelt
Institut für Informatik, Fakultät für Mathematik und Informatik Universität Heidelberg
Im Neuenheimer Feld 368, D-69120 Heidelberg, Germany
gerhard.reinelt@iwr.uni-heidelberg.de

ABSTRACT

The coupled task problem is to schedule jobs on a single machine where each job consists of two subtasks and where the second subtask has to be started after a given time interval with respect to the first one. The problem has several applications and is NP-hard. In this paper we present a branch-and-bound algorithm for this problem and compare its performance with integer programming models.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

1. INTRODUCTION

In this presentation we consider the so called Coupled Task Scheduling problem. This problem can be defined as follows: we are given n jobs and each job consists of two distinct sub-tasks. The sequence of the sub-tasks is fixed and there is a fixed length delay-time between the two parts. We can denote job J_i by a triple (a_i, L_i, b_i), where the values represent the processing time of the first sub-task, the delay time between the two sub-tasks and the processing time of the second sub-task, respectively. It is required that the second sub-task must be scheduled *exactly* $L_i + a_i$ units after having started the first one. During the delay time the machine is idle, so it is possible to schedule other sub-tasks in this interval. The aim is to schedule the given n coupled-tasks on one machine in such a way that no two job tasks can overlap. We want to minimize the latest finishing time of the jobs. (As usual we call this as the makespan and we denote it by $C_{\max}$). Preemption, i.e., interrupting a task and resuming it later is not allowed.

The general problem (1|Coup-Task|$C_{\max}$) first was investigated by Shapiro [4]. He discussed the practical applications of the problem and presented 3 heuristics for this hard problem. It is proved that this problem is $\mathcal{NP}$-hard (see [3]). Unfortunately, the author did not analyze the heuristics in detail, only experimental results showed their efficiencies. Orman and Potts investigated the problem from complexity point of view (see [2]). Later Sherali and Smith gave a mathematical formulation for the problem [5]. In this presentation we deal with the optimal solution of the general problem. We present a new combinatorial branch-and-bound algorithm, which works without using linear programming or other similar techniques. In addition to this we introduce new 0-1 programming formulations and compare the models computationally to the one given by Sherali and Smith.

*This work was supported by the European Union and the European Social Fund through project " Supercomputer, the national virtual lab" grant no.: TAMOP-4.2.2.C-11/1/KONV-2012-0010.

2. THE BRANCH-AND-BOUND METHOD

The branch-and-bound method works in a combinatorial way. This means that the bounds of the nodes of the B&B tree are calculated from the combinatorial structure of the schedule. Each node represent a subset of the jobs, where the sub-tasks are ordered in a given way. Child nodes are created by inserting a new job to the current list in all possible ways. To decide the feasibility of a given permutation of the sub-tasks a feasibility test algorithm is developed. In case of a feasible schedule the output of the procedure is a set of time windows representing the possible starting times of the sub-tasks. Infeasible schedules are cut-off immediately from the tree. For node selection the *dive-and-best* strategy is applied. Its advantage that it finds good feasible solutions fast and it tries to compute tight bounds. For selecting the next node to be processed the *best-bound-first* strategy is used, i.e. the child node with the best lower bound is selected. For larger problems the tree may contain huge number of nodes. To improve the efficiency of the algorithm different kind of bounding procedures can be applied. In this case we use four specific methods, which can effectively decrease the computational time by pruning the tree. The proposed methods are based on the analysis of the gaps in the schedule and they give specific lower bounds on the makespan. Besides bounding it is also important to exclude nodes with symmetric properties from the tree. This is done by introducing blocks and lexicographic ordering of the schedules. Computational studies prove that this branch-and-bound method is more effective than the mathematical programming approaches.

3. THE INTEGER PROGRAMMING FORMULATIONS

Our presentation contains different 0-1 programming formulations. Here we present the time-indexed formulation, which is common for modeling machine scheduling problems. The basic ideas of this method were already presented in [1].

To get the optimal schedule we want to minimize M, which is an upper bound on the makespan. To do this we define an optimization problem, which decides if all jobs can be scheduled in the interval $[0, M]$. Then we can calculate the optimal value of M by a binary sarch. To give the optimization problem, we need binary variables y_{jt} and z_{jt} for $j = 1, ..., n$ and $t = 0, ..., M$. The interpretation of these variables is the following:

$$y_{jt} = \begin{cases} 1 & \text{job } j \text{ is started at time } t, \\ 0 & \text{otherwise} \end{cases}$$

and

$$z_{jt} = \begin{cases} 1 & \text{a sub-task of job } j \text{ occupies the machine in time interval } [t, t+1], \\ 0 & \text{otherwise.} \end{cases}$$

There is a connection between the variables y_{jt} and z_{jt}, namely

$$z_{jt} = \sum_{s=t-a_j+1}^{t} y_{js} + \sum_{s=t-a_j-l_j-b_j+1}^{t-a_j-l_j} y_{js}, \text{ for } 0 \le t \le M.$$

Note that we ignore the variables y_{js} with $s < 0$ here. Using these notations the corresponding optimization problem is the following:

$$\max \sum_{j=1}^{n} \sum_{t=0}^{M-1} y_{jt}$$

subject to

$$\sum_{t=0}^{M-1} y_{jt} \le 1, \text{ for } j = 1, 2, ..., n \quad (1)$$

$$\sum_{j=1}^{n} z_{jt} \le 1, \text{ for } t = 0, 1, ..., M-1 \quad (2)$$

$$\sum_{j=1}^{n} z_{j0} = 1 \quad (3)$$

$$y_{jt} = 0, \text{ for } t = M - a_j - l_j - b_j + 1, ..., M-1 \quad (4)$$
$$j = 1, ..., n, \quad (5)$$

where inequalities (1) ensure that every job is scheduled at most once and inequalities (2) ensure that no two sub-tasks overlap. Using (3) we enforce that the schedule starts as early as possible and because of constraints (4) each job is started only if it can be completely processed.

According to computational studies this formulation can provide good results for some special instances.

4. REFERENCES

[1] M. Elshafei, H.D. Sherali, J.C. Smith 2004, *Radar pulse interleaving for multi-target tracking*, Naval Research Logistics 51 (4), 72-94.

[2] A.J. Orman, C.N. Potts 1997, *On the Complexity of Coupled-Task Scheduling*, Discrete Applied Mathematics 72, 141-154.

[3] A.H.G. Rinnooy Kan 1976, *Machine Scheduling Problems*, Martinus Nijhoff, The Hague.

[4] R.D. Shapiro 1980, *Scheduling Coupled Tasks*. Naval Research Logistics Quarterly, 20, 489-498.

[5] H.D. Sherali, J.C. Smith 2005, *Interleaving Two-Phased Jobs on a Single Machine with Application to Radar Pulse Interleaving*. Discrete Optimization 2, 348-361.