A New Formulation for Scheduling Unrelated Processors under Precedence Constraints

Nelson Maculan
Universidade Federal do Rio de Janeiro
COPPE - Engenharia de Sistemas e Computação
Caixa Postal 68514
Rio de Janeiro 21945, Brazil
e-mail: maculan@cos.ufrj.br

Stella C.S. Porto
Universidade Federal Fluminense
Dept. of Telecommunication Engineering
Rua Passos da Pátria 156
Niterói 24210, Brazil
e-mail: stella@caa.uff.br

Celso Carneiro Ribeiro
Pontifícia Universidade Católica
Departamento de Informática
Rua Marquês de São Vicente 225
Rio de Janeiro 22453-900, Brazil
e-mail: celso@inf.puc-rio.br

Cid Carvalho de Souza
Universidade Estadual de Campinas
Departamento de Ciência da Computação
Caixa Postal 6065
Campinas 13081, Brazil
e-mail: cid@dcc.unicamp.br

PUCRioInf-MCC37/95

December 1995

Abstract:

We give a new formulation for the problem of task scheduling into unrelated processors under precedence constraints. This formulation has a polynomial number of variables and does not require that the processing times be integer valued.

Keywords: Parallel processing, scheduling, unrelated processors, precedence constraints, make-span.

Resumo:

Apresenta-se uma nova formulação para o problema de escalonamento de tarefas em processadores não relacionados sob restrições de precedência. Esta formulação usa um número polinomial de variáveis e não necessita que os tempos de processamento assumam valores inteiros.

Palavras-chaves: Processamento paralelo, escalonamento, processadores não relacionados, restrições de precedência, tempo de execução.

1 Motivation

Let $T = \{t_1, \ldots, t_n\}$ be a set of partially ordered tasks and $P = \{p_1, \ldots, p_m\}$ a set of processors. We are also given an (acyclic directed) precedence graph $G = [T, A]$ associated with the set of tasks [1, 4], such that $(t_i, t_j) \in A$ if and only if task t_i must be executed before task t_j. Each task has to be assigned to exactly one processor, in which it is entirely executed without preemption. For each task $t_j \in T$ and each processor $p_k \in P$, we denote by d_{jk} the total processing time of task t_j in case it is assigned to processor p_k. Three situations may occur:

- identical processors: $d_{j,k_1} = d_{j,k_2}, \quad \forall t_j \in T, \quad \forall p_{k_1}, p_{k_2} \in P$
- uniform processors: $d_{j_1,k_1}/d_{j_1,k_2} = d_{j_2,k_1}/d_{j_2,k_2}, \quad \forall t_{j_1}, t_{j_2} \in T, \quad \forall p_{k_1}, p_{k_2} \in P$
- unrelated processors: arbitrary processing times $d_{j,k}, \quad \forall t_j \in T, \quad \forall p_k \in P$.

The task scheduling problem under precedence constraints consists in finding an optimal assignment of the tasks in T to the processors in P minimizing the makespan, i.e. the maximum completion time among all tasks in T. The minimization of the makespan on two uniform processors (problem $Q2 \mid\mid C_{max}$ in the notation of [7]) is already NP-hard [5, 6].

An application of this problem arises in the context of scheduling tasks of parallel programs. Parallel programs can be represented as a set of interrelated tasks which are sequential units. In a heterogeneous multiprocessor system, we not only have to determine how many, but also which processors should be allocated to an application, as well as which processors are going to be assigned to each task. Greedy algorithms for processor assignment of parallel applications modeled by task precedence graphs in heterogeneous multiprocessor architectures have been proposed by Menascé and Porto [8], while Porto and Ribeiro [10, 11] have studied sequential and parallel algorithms based on the tabu search metaheuristic.

In this paper, we give a new formulation for the problem of task scheduling into unrelated processors under precedence constraints. In the next section, we recall the existing formulation for this problem, based on the discretization of the schedule horizon into unit time-periods. Following, we present the new formulation with a polynomial number of 0-1 variables. Some final remarks are drawn in the last section.

2 Existing Formulation

We remind in this section the only known formulation [2] for this problem, given by Blazewicz et alii [3] in the context of deterministic models for scheduling under resource constraints. For ease of notation, we also define the sets of indices $N = \{1, \ldots, n\}$ and $M = \{1, \ldots, m\}$ associated, respectively, with the sets T of tasks and P of processors. Let t_H denote a known heuristic schedule length, called the time horizon, and assume that t_H and all processing times d_{jk} are integer valued. Notice that a trivial upper bound for t_H is given by $\sum_{j=1}^{j=n} \text{minimum}_{k=1,\ldots,m}\{d_{jk}\}$. If the time horizon is splitted into t_H unit time-periods, we may define the following class of 0-1 variables:

$$x_{kjt} = \begin{cases} 1, & \text{if task } t_j \text{ scheduled to processor } p_k \text{ is completed in the } t\text{-th time period,} \\ 0, & \text{otherwise,} \end{cases}$$

for all $j \in N$, $k \in M$, and $t = 1, \ldots, n_H$. Without loss of generality, task t_n is assumed to be last task to be executed, i.e., the unique terminal task without successors. Associated with each task t_j are its critical-path determined early finish time e_j and late finish time ℓ_j, which may be precomputed by forward and reverse passes through the precedence graph. Again, 0 and t_H are valid lower and upper bounds for e_j and ℓ_j, respectively, for all $t_j \in T$.

We denote by $\Gamma(j)$ the set of the indices of the immediate predecessors of task t_j, i.e. $\Gamma(j) = \{i \in N \mid (t_i, t_j) \in A\}$. This scheduling problem, consisting in the minimization of the makespan C_{max} under precedence constraints, may be formulated in the following way:

$$\text{minimize} \quad C_{max} = \sum_{k=1}^{m} \sum_{t=e_n}^{\ell_n} t \cdot x_{knt} \tag{1}$$

subject to:

$$\sum_{k=1}^{m} \sum_{t=e_j}^{\ell_j} x_{kjt} = 1, \quad \forall j \in N \tag{2}$$

$$\sum_{k=1}^{m} \sum_{t=e_j}^{\ell_j} (t - d_{jk}) \cdot x_{kjt} - \sum_{k=1}^{m} \sum_{t=e_i}^{\ell_i} t \cdot x_{kit} \geq 0, \quad \forall j \in N, \quad \forall i \in \Gamma(j) \tag{3}$$

$$\sum_{j=1}^{n} \sum_{q=t}^{t-d_{jk}-1} x_{kjq} \leq 1, \quad \forall t = 1, \ldots, t_H, \quad \forall k \in M \tag{4}$$

$$x_{kjt} \in \{0, 1\}, \quad \forall k \in M, \quad \forall j \in N, \quad \forall t = 1, \ldots, n_H, \tag{5}$$

where equalities (2) ensure that each task is completed and assigned to exactly one processor, inequalities (3) express precedence constraints (no task may be started unless all its predecessors have already completed their execution), and inequalities (4) state that each machine can be used by at most one task in each time-period.

Regardless of more practical issues concerning its solvability as a large scale integer programming problem, this formulation already has two main drawbacks. First, it requires that the processing times be integer valued. Second, a non-polynomial number of 0-1 variables is used, due to the splitting of the schedule horizon into unit time-periods. In the next section, we present a more suitable formulation, with a polynomial number of variables.

3 Formulation with a Polynomial Number of Variables

We use the same notation already introduced in the previous sections. We define a new set of 0-1 variables:

$$y_{jk}^{s} = \begin{cases} 1, & \text{if task } t_j \text{ is the } s\text{-th task executed by processor } p_k, \\ 0, & \text{otherwise,} \end{cases}$$

for all $j \in N$, $k \in M$, and $s = 1, \ldots, n$. For every task $t_j \in T$, we denote by $w_j \geq 0$ the starting time of its execution. Then, the problem may be formulated as follows:

$$\text{minimize} \quad C_{max} \tag{6}$$

subject to:

$$\sum_{k=1}^{m} \sum_{s=1}^{n} y_{jk}^{s} = 1, \quad \forall j \in N \tag{7}$$

$$\sum_{j=1}^{n} y_{jk}^{1} \leq 1, \quad \forall k \in M \tag{8}$$

$$\sum_{j=1}^{n} y_{jk}^{s} \leq \sum_{j=1}^{n} y_{jk}^{s-1}, \quad \forall k \in M, \quad \forall s = 2, \ldots, n \tag{9}$$

$$w_j \geq w_i + \sum_{k=1}^{m} \sum_{s=1}^{n} d_{ik} \cdot y_{ik}^{s}, \quad \forall i \in \Gamma(j), \quad \forall j \in N \tag{10}$$

$$w_j \geq w_i + d_{ik} - \alpha \cdot [2 - (y_{ik}^{s} + \sum_{r=s+1}^{n} y_{jk}^{r})], \quad \forall k \in M, \ \forall s = 1, \ldots, n-1, \ \forall (i,j) \in N \times N \tag{11}$$

$$C_{max} \geq p_j + \sum_{k=1}^{m} \sum_{s=1}^{n} d_{jk} \cdot y_{jk}^{s}, \quad \forall j \in N \tag{12}$$

$$y_{jk}^{s} \in \{0, 1\}, \quad \forall j \in N, \quad \forall k \in M, \quad \forall s = 1, \ldots, n \tag{13}$$

$$w_j \geq 0, \quad \forall j \in N, \tag{14}$$

where $\alpha \gg 0$ is a sufficiently large penalty coefficient.

Equations (7-8-9-10) play a role similar to that of (2-3-4) in the previous formulation. Equalities (7) ensure that each task is assigned to exactly one processor. Inequalities (8-9) state that each processor can not be simultaneously used by more than one task. The first of those (8) means that at most one task will be the first one to be executed by any given processor $p_k \in P$. Notice that for a task to be the s-th one (with $s \geq 2$) assigned to processor p_k, then there must be another one assigned as the $(s-1)$-th task of this same processor p_k, as given by inequalities (9). Inequalities (10) express precedence constraints (no task may be started unless all its predecessors have already completed their execution), while constraints (12) define the maximum completion time, i.e. the makespan.

We now turn our attention to constraints (11), which define the sequence of starting times of the set of tasks assigned to the same processor. They express the fact that task t_j must start at least d_{ik} time units after the beginning of task t_i, whenever it is executed after task t_i in the same processor p_k, i.e. $y_{ik}^{s} = \sum_{r=s+1}^{n} y_{jk}^{r} = 1$ for some $s = 1, \ldots, n-1$. Notice that these constraints where not necessary in the previous formulation, since the task starting times were defined by the variables x_{kjt} themselves.

4 Final Remarks

The success of integer programming methods is known to be very dependent on the bound given by the linear relaxation provided by the formulation. To improve this bound, we seek for stronger inequalities to add to the model. The formulation given in the previous section may be improved further.

One way to measure the "strength" of an inequality is to compute the amount of (affine independent) integer solutions satisfying it at equality. The larger this amount is, the stronger is the inequality (see [9] for details). Inequalities (11) may be strengthened, in such a way that the number of integer solutions satisfying these inequalities at equality increases. To do so, we add new terms to these inequalities, in a process usually called lifting. Assuming that i, j, k, and s are fixed, we notice that for a solution to verify inequality (11) at equality, tasks t_i and t_j have to be respectively the $(s-1)$-th and s-th tasks executed by processor p_k and, moreover, this processor cannot remain idle between the execution of these two tasks. Lifting inequalities (11) leads to the stronger valid inequalities (11') below:

$$w_j \geq w_i + d_{ik} - \alpha \cdot [2 - (\sum_{r=1}^{s} y_{ik}^r + \sum_{r=s+1}^{n} y_{jk}^r)] + \sum_{l \neq i \neq j, l=1}^{n} d_{lk}(y_{lk}^s + y_{lk}^{s+1} + \textstyle\sum_{r=s+1}^{n} y_{jk}^r - 1),$$
$$\forall k \in M, \quad s = 1, \ldots, n-1, \quad \forall (i,j) \in N \times N \qquad (11')$$

We do not claim that any of the formulations presented above are directly amenable to be solved by standard integer programming techniques, such as branch-and-bound or branch-and-cut. However, as already mentionned, the first formulation already has two main drawbacks. First, it requires that the processing times be integer valued. Second, a non-polynomial number of 0-1 variables is used, due to the splitting of the schedule horizon into unit time-periods. However, both of these difficulties are overcame by the new formulation.

We have developed in this paper a new, more suitable formulation for the problem of task scheduling into unrelated processors under precedence constraints. This formulation makes use of only a polynomial number of 0-1 variables. Moreover, the assumption of integer valued processing times is no longer necessary. Current research and further extensions of this work consist in finding good relaxations of this formulation, which could generate sharper lower bounds for the optimal value of the makespan; solving practical applications on regularly structured precedence graphs, corresponding to real-life processor configurations; and its application to the problems already solved by Porto and Ribeiro [10, 11], which would allow the evaluation of the quality of the approximate solutions already obtained by tabu search.

References

[1] D.P. Bertsekas and J.N. Tsitsiklis, *Parallel and Distributed Computation*, Prentice-Hall, Englewood Cliffs, 1989.

[2] J. Blazewicz, *Personnal communication*, 1993.

[3] J. Blazewicz, W. Cellary, R. Slowinski, and J. Weglarz, *Scheduling Under Resource Constraints - Deterministic Models*, J.C. Baltzer AG, Basel, 1986.

[4] E.G. Coffman and P.J. Denning, *Operating Systems Theory*, Prentice-Hall Inc., New Jersey, 1973.

[5] M.R. Garey and D.S. Johnson, "Strong NP-Completeness Results: Motivation, Examples and Implications", *Journal of the ACM* 25 (1978), 499–508.

[6] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Company, San Francisco, 1979.

[7] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, "Sequencing and Scheduling: Algorithms and Complexity", Report NFI 11.89/03, Eindhoven Institute of Technology, Department of Mathematics and Computer Science, Eindhoven, 1989.

[8] D.A. Menascé and S.C.S. Porto, "Processor Assignment in Heterogeneous Parallel Architectures", *Proceedings of the IEEE International Parallel Processing Symposium*, 186–191, Beverly Hills, 1992.

[9] G.L. Nemhauser and L.A. Wolsey, *Integer and Combinatorial Optimization*, John Wiley and Sons, 1988.

[10] S.C. Porto and C.C. Ribeiro, "A Tabu Search Approach to Task Scheduling on Heterogeneous Processors under Precedence Constraints", *International Journal of High Speed Computing* 7 (1995), 45–71.

[11] S.C. Porto and C.C. Ribeiro, "Parallel Tabu Search Message-Passing Synchronous Strategies for Task Scheduling under Precedence Constraints", *Journal of Heuristics* 1 (1995).