XXX Simpozijum o operacionim istraživanjima
SYM-OP-IS 2003
Hotel PLAŽA, Herceg-Novi, 30.09.-03.10.2003.

FORMULACIJA PROBLEMA RASPOREDJIVNJA ZADATAKA NA VIŠE PROCESORA KORIŠĆENJEM MATEMATIČKOG PROGRAMIRANJA

A MATHEMATICAL PROGRAMMING FORMULATION FOR THE MULTIPROCESSOR SCHEDULING PROBLEM WITH COMMUNICATION DELAYS*

Tatjana Davidović
[1]Matematički institut SANU, Beograd, p.f. 367. email:tanjad@mi.sanu.ac.yu
Nelson Maculan
COPPE-Systems Engineering, Federal University of Rio de Janeiro, P.O. Box 68511
21941-972 Rio de Janeiro, Brazil, email: maculan@cos.ufrj.br
Nenad Mladenović[1]
email: nenad@mi.sanu.ac.yu

Rezime – *U radu se formuliše problem statičkog rasporedjivanja zavisnih zadataka na višeprocesorske sisteme proizvoljne arhitekture uz prisustvo komunikacija kao problem matematičkog programiranja. Najpre se razmatra slučaj rasporedjivanja na identične procesore (homogeni višeprocesorski sistem), a zatim se ukratko razmatra proširenje na heterogeni slučaj.*

KLJUČNE REČI: STATIČKA RASPODELA, HOMOGENI PROCESORI, KOMUNIKACIONO KAŠNJENJE, LINEARNO PROGRAMIRANJE.

Abstract - *A mathematical programming formulation for static scheduling of dependent tasks onto homogeneous multiprocessor system of an arbitrary architecture with communication delays is given. Heterogeneous processors case is briefly discussed.*

KEY WORDS: STATIC SCHEDULING, HOMOGENEOUS MULTIPROCESSORS, COMMUNICATION DELAYS, LINEAR PROGRAMMING.

1. INTRODUCTION

Parallel computing is very attractive research field for more than forty years now [5]. One of the main problems is scheduling of modules (tasks) to processors, i.e. definition of distribution and order of tasks among processors. There is a bulk of papers considering different variant of scheduling problems, a recent survey can be found in [1]. This problem is known to be NP-hard in majority of cases, although there are several special cases that can be solved in polynomial time. Therefore, heuristic methods for finding solutions of good quality are welcome. Such suboptimal solution can be used as upper bound for an exact solution method based on branch-and-bound or branch-and-cut, but in that case it is useful to have mathematical formulation in order to derive lower bounds by relaxation. There are several possible ways to formulate scheduling problems: using graph representation for the program and/or multiprocessor architecture [3,6,11], via sets of instructions [9], or by the formal, mathematical programming based definitions.

In the literature one can find mathematical formulation for different variants of scheduling problems. For example, in [10] the formulation for scheduling independent tasks is given. Scheduling of dependent tasks without communication is modelled in several different ways [2,8,12]. In [7] the authors considered the problem of scheduling dependent tasks onto heterogeneous multiprocessor architecture, but with neglected communication time. To the best of our knowledge, up to now no model involving communications was developed.

* This research has been supported in part by NSF Serbia. The first and the third authors have been supported by grant no. 1583.

Moreover, in existing models, it was always assumed that processors are completely connected.

In this paper we suggest the mathematical programming formulation of Multiprocessor Scheduling Problem with Communication Delays (MSPCD). The problem consists of scheduling dependent tasks with communication delays onto homogeneous, arbitrary connected multiprocessor architecture. The extension to heterogeneous case is briefly discussed.

The paper is organized as follows: in the next section combinatorial, graph representation based description of the MSPCD is given. Section 3 contains mathematical programming formulation of the problem. Possible extensions to the heterogeneous multiprocessor architecture are discussed in section 4, while section 5 concludes the paper.

2. PROBLEM DESCRIPTION

The Multiprocessor Scheduling Problem With Communication Delays (MSPCD) is as follows: tasks (or jobs) have to be executed on several processors; we have to find where and when each task will be executed, such that the total completion time is minimum. A duration of each task is known as well as precedence relations among tasks, i.e. what tasks should be completed before some other could begin. In addition, if dependent tasks are executed on different processors, the data transferring time (or communication delay) that are given in advance are also considered.

The tasks to be scheduled are represented by a directed acyclic graph (DAG) [3,6,11] defined by a tuple $G = (M, E, C, L)$ where $M = \{1, \ldots, n\}$ denotes the set of tasks (modules); $E = \{e_{ij}, \ |i, j \in M\}$ represents the set of communication edges; $C = \{c_{ij}, e_{ij} \in E\}$ denotes the set of edge communication costs; and $L = \{l_1, \ldots, l_n\}$ represents the set of task computation times (execution times, lengths). The communication cost $c_{ij} \in C$ denotes the amount of data transferred between tasks i and j if they are executed on different processors. If both tasks are scheduled to the same processor the communication cost equals zero. The set E defines precedence relation between tasks. A task cannot be executed unless all of its predecessors have completed their execution and all relevant data is available. Task preemption and redundant execution are not allowed.

The multiprocessor architecture $A = \{1, 2, \ldots, p\}$ is assumed to contain p identical processors with their own local memories which communicate by exchanging messages through bidirectional links of the same capacity. This architecture is modelled by a *distance matrix*} [3,4]. The element (k, l) of the distance matrix $D = [d_{kl}]_{p \times p}$ is equal to the minimum distance between the nodes k and l. Here, the minimum distance is calculated as the number of links along the shortest path between two nodes. It is obvious that distance matrix is symmetric with zero diagonal elements.

The scheduling of DAG G onto A consists of determining the index of the associated processor and starting time instant for each of the tasks from the task graph in such a way as to minimize some objective function. The usual objective function (that we shall use in this paper as well) is completion time of the scheduled task graph T_{max} (also referred to as makespan, response time or schedule length). The starting time of a task i depends on the completion times of its predecessors and the amount of time needed for transferring the data from the processors executing these predecessors to the processor that has to execute the task i. Depending on multiprocessor architecture the time that is spent for communication between tasks i and j can be calculated in the following way

$$\gamma_{ij}^{kl} = c_{ij} \cdot d_{kl} \cdot ccr, \qquad (0)$$

where it is assumed that task i will be executed on processor l, task j on processor k and ccr represents the Communication-to-Computation-Ratio which is defined as the ratio between time for transferring the unit amount of data and the time spent for performing single computational operation. This parameter is used to describe the characteristics of multiprocessor system. In message passing systems ccr usually has a large value because communication links are very slow. For shared-memory multiprocessors the communication is faster since it consists of writing data from main (electronic) memory of one processor into global (also fast) memory and then into main memory of another processor. If the tasks are scheduled to the same processor, i.e. $k=l$, the amount of communication is equal to zero since $d_{kk}=0$.

3. MATHEMATICAL FORMULATION

The informal, graph representation based definition of the MSPCD is given in the previous section. The formal definition of similar problem is given in [7]. In that paper the problem of scheduling dependent tasks onto heterogeneous multiprocessor architecture is considered, but assuming that communication cost can be neglected, i.e. $c_{ij}=0$. Here we will adopt this formulation for the MSPCD, assuming communication costs to be significant and multiprocessor architecture consists of identical processors connected in arbitrary way.

Let us denote the set of immediate predecessors of task j by $Pred(j)$, i.e. $Pred(j) = \{i \in M \mid e_{ij} \in E\}$.

Let

$$y_{jk}^{s} = \begin{cases} 1, & \text{if } j \text{ is the } s^{\text{th}} \text{ task executed on proc.} k, \\ 0, & \text{otherwise,} \end{cases}$$

$\forall j \in M, k \in$ A .

Another set of variables we shall use in the model represents the starting time of task j and it is denoted by t_j, while T_j denotes the completion time of task j, i.e. $T_j = t_j + l_j$.

The MSPCD then can be formulated as follows

$$\text{minimize} \quad T_{max} \tag{1}$$

subject to:

$$\sum_{k=1}^{p}\sum_{s=1}^{n} y_{jk}^{s} = 1, \quad \forall j \in \{1,2,...,n\} \tag{2}$$

$$\sum_{j=1}^{n} y_{jk}^{1} \le 1, \quad \forall k \in \{1,2,...,p\} \tag{3}$$

$$\sum_{j=1}^{n} y_{jk}^{s} \le \sum_{j=1}^{n} y_{jk}^{s-1}, \forall k \in \{1,2,...,p\}, \quad \forall s \in \{2,...,n\} \tag{4}$$

$$t_j \ge t_i + \sum_{k=1}^{p}\sum_{s=1}^{n} l_i \cdot y_{ik}^{s} + \sum_{k=1}^{p}\sum_{s=1}^{n}\sum_{l=1}^{p}\sum_{r=1}^{n} \gamma_{ij}^{kl} \cdot y_{ik}^{s} \cdot y_{jl}^{r}, \quad \forall i \in Pred(j), \quad \forall j \in \{1,2,...,n\} \tag{5}$$

$$t_j \ge t_i + l_i - \alpha \cdot \left[2 - \left(y_{ik}^{s} + \sum_{r=s+1}^{n} y_{jk}^{r} \right) \right], \quad \forall k \in \text{A}, \quad \forall s \in \{1,2,...,n-1\}, \quad \forall i, j \in M \tag{6}$$

$$T_{\max} \ge t_j + \sum_{k=1}^{p}\sum_{s=1}^{n} l_j \cdot y_{jk}^{s}, \forall j \in \{1,2,...,n\} \tag{7}$$

$$y_{jk}^{s} \in \{0,1\}, \forall j \in \{1,2,...,n\}, \forall k \in \{1,2,...,p\} \quad \forall s \in \{1,2,...,n\} \tag{8}$$

$$t_j \ge 0, \quad \forall j \in \{1,2,...,n\} \tag{9}$$

where α 0 is a sufficiently large penalty coefficient and γ_{ij}^{kl} represent the amount of communication between tasks i and j as defined by (0).

Equations (2) ensure that each task is assigned to exactly one processor. Inequalities (3)-(4) state that each processor can not be simultaneously used by more than one task. (3) means that at most one task will be the first one at k, while (4) ensures that if some task is the s^{th} one ($s \ge 2$) scheduled to processor k then there must be another task assigned as $(s\text{-}1)^{\text{th}}$ to the same processor. Inequalities (5) express precedence constraints together with communication time required for tasks assigned to different processors, while constrains (7) define the objective function value (maximum completion time, makespan, $T_{max} = \max_j \{T_j\}$).

Constraints (6) define the sequence of the starting times of the set of tasks assigned to the same processor. They express the fact that task j must start at least l_i time units after the beginning of task i whenever j is executed after i on the same processor k. If tasks i and j are not assigned to the same processor and they are mutually independent, the previous reasoning does not hold: even being r^{th} ($r>s$) task on some other processor l, j may start before i completes its execution. The last term of inequalities (6), i.e. $\alpha \cdot \left[2 - \left(y_{ik}^{s} + \sum_{r=s}^{n} y_{jk}^{r}\right)\right]$, is added to cover that case.

Mathematical formulation of MSPCD given by (1)-(9) belongs to mixed integer bilinear programs: Variables y_{jk}^{s} are of 0-1 type and variables t_i are continuous. However, it is possible to linearize this model by introducing new set of 0-1 variables $z_{ijkl}^{sr} = y_{ik}^{s} \cdot y_{jl}^{r}$.

The variables z_{ijkl}^{sr} have to satisfy following conditions

$$y_{ik}^{s} \ge z_{ijkl}^{sr} \tag{10}$$

$$y_{jl}^{r} \ge z_{ijkl}^{sr} \tag{11}$$

$$y_{ik}^{s} + y_{jl}^{r} - 1 \le z_{ijkl}^{sr} \tag{12}$$

$$z_{ijkl}^{sr} \ge 0 \tag{13}$$

4. HETEROGENEOUS PROCESSORS CASE

When considering heterogeneous multiprocessor system, new difficulty arises. First, l_i has to be substituted by l_{ik} to describe different computational capabilities of the processors. Then, different capacity of communication links ν_q, (q=1,…,p) has to be introduced for each processor. This means several things, first, *ccr* is becoming a vector, having different value for different processors. In addition, the right hand side of the formula (0) is becoming more complex. It is not enough just to consider the distance between tasks since the communication time is varying along the exact route data have to travel along, because of the different corresponding communication capacities. This will force

d_{kl} to be substituted by the term containing sum of the communication capacities of all the processors along the route. Instead of the sum, we can use maximum value to estimate upper bound for communication delay guided by the following reasoning. Communication time η_q, is in reverse relation with the communication capacity ν_q: larger capacity of communication links yields smaller communication time. Moreover, for each pair of processors k and l, the communication time is determined by the communication capacity of the slower processor, i.e. $\nu_{kl}=\min\{\nu_k, \nu_l\}$ and consequently, communication time is $\eta_{kl}=\max\{\eta_k, \eta_l\}$. To simplify the description, we can approximate the communication time with the largest value among all the processors, i.e. set $\eta=\max\{\eta_q, 1\leq q\leq p\}$ which means that we will always overestimate the required communication time, i.e. the optimal solution. If we introduce η into consideration we obtain

$$d_{kl} \rightarrow \sum_{m=1}^{d_{kl}} \eta_m \geq \max_{1\leq q\leq p} \eta_q \cdot d_{kl} = \eta \cdot d_{kl}.$$

Finally, for the heterogeneous case we have: equation (0) is substituted with

$$\gamma_{ij}^{kl} \geq c_{ij} \cdot d_{kl} \cdot \eta \cdot \max_{1\leq q\leq p} ccr_q,$$

and also, in (5), (6) we have to substitute l_i with l_{ik} and in (7) l_j with l_{jk}.

5. CONCLUSION

Complete linear programming formulation for scheduling dependent tasks onto homogeneous multiprocessor architecture of an arbitrary structure with taking into account communication delays is proposed. The extension to heterogeneous multiprocessor system is briefly discussed.

The proposed formulation is extension of existing models proposed for the variants of scheduling problem without communication. It can be used for determining the lower bounds when developing exact solution methods for multiprocessor scheduling problem with communication delays.

REFERENCES

[1] J. Blazewicz, M. Drozdowski, K. Ecker. Management of resources in parallel systems. In J. Blazewicz, *et al.* edts., *Handbook on Parallel and Distributed Processing*, pages 263--341. Springer, 2000.

[2] E. H. Bowman. The schedule-sequencing problem. *Operationals Research*, 7(5):621--624, 1959.

[3] T. Davidović. Exaustive list--scheduling heuristic for dense task graphs. *YUJOR*, 10(1):123--136, 2000.

[4] G. Djordjević and M. Tošić. A compile-time scheduling heuristic for multiprocessor architectures. *The Computer Journal*, 39(8):663--674, 1996.

[5] T. C. Hu. Parallel sequencing and assembly line problems. *Oper. Res*, 9(6):841--848, Nov. 1961.

[6] Y.-K. Kwok and I. Ahmad. Efficient scheduling of arbitrary task graphs to multiprocessors using a parallel genetic algorithm. *J. Parallel and Distributed Computing*, 47:58--77, 1997.

[7] N. Maculan, C. C. Ribeiro, S. C. S. Porto, and C. C. de Souza. A new formulation for scheduling unrealted processors under precedence constraints. *RAIRO Recherche Operationelle*, 33:87--90, 1999.

[8] A. S. Manne. On the job--shop scheduling problem. *Operationals Research*, 8(2):219--223, 1960.

[9] D. A. Menascé and S. C. S. Porto. Processor assignment in heterogeneous parallel architectures. In *Proc. of the IEEE Int. Parallel Processing Symposium*, 186--191, Beverly Hills, 1992.

[10] M. Queyranne and A. Schulz. Polyhedral approaches to machine scheduling. Technical report, Report 1994--408, Technical University of Berlin, 1994.

[11] G. C. Sih and E. A. Lee. A compile-time scheduling heuristic for interconnection-constrained heterogeneous processor architectures. *IEEE Trans. on Parallel and Distributed Systems*, 4(2):175--187, February 1993.

[12] H. M. Wagner. An integer linear-programming model for machine scheduling. *Res. Log. Quart.*, 6(2):131--140, 1959.