Non-approximability Results for the Hierarchical Communication Problem with a Bounded Number of Clusters

Eric Angel, Evripidis Bampis, Rodolphe Giroudeau

Non-approximability Results for the Hierarchical Communication Problem with a Bounded Number of Clusters*

(Extended Abstract)

Eric Angel, Evripidis Bampis, and Rodolphe Giroudeau

LaMI, CNRS-UMR 8042, Université d'Evry Val d'Essonne
523, Place des Terrasses
F–91000 Evry France
{angel, bampis, giroudea}@lami.univ-evry.fr

Abstract. We study the hierarchical multiprocessor scheduling problem with a constant number of clusters. We show that the problem of deciding whether there is a schedule of length three for the hierarchical multiprocessor scheduling problem is $\mathcal{NP}$-complete even for bipartite graphs i.e. for precedence graphs of depth one. This result implies that there is no polynomial time approximation algorithm with performance guarantee smaller than 4/3 (unless $\mathcal{P} = \mathcal{NP}$). On the positive side, we provide a polynomial time algorithm for the decision problem when the schedule length is equal to two, the number of clusters is constant and the number of processors per cluster is arbitrary.

1 Introduction

For many years, the standard communication model for scheduling the tasks of a parallel program has been the *homogeneous communication model* (also known as the *delay model*) introduced by Rayward-Smith [12] for unit-execution-times, unit-communication times (UET-UCT) precedence graphs. In this model, we are given a set of identical processors that are able to communicate in a uniform way. We wish to use these processors in order to process a set of tasks that are subject to precedence constraints. Each task has a processing time, and if two adjacent tasks of the precedence graph are processed by two different processors (resp. the same processor) then a communication delay has to be taken into account explicitly (resp. the communication time is neglected). The problem is to find a trade-off between the two extreme solutions, namely, execute all the tasks sequentially without communications, or try to use all the potential parallelism but in the cost of an increased communication overhead. This model has been extensively studied these last years both from the complexity and the (non)-approximability point of views [7].

* This work has been partially supported by the APPOL II (IST-2001-32007) thematic network of the European Union and the GRID^2 project of the French Ministry of Research.

B. Monien and R. Feldmann (Eds.): Euro-Par 2002, LNCS 2400, pp. 217–224.

In this paper, we adopt the *hierarchical communication model* [1,3] in which we assume that the communication delays are not homogeneous anymore; the processors are connected in *clusters* and the communications inside the same cluster are much faster than those between processors belonging to different clusters. This model captures the hierarchical nature of the communications in todays parallel computers, composed by many networks of PCs or workstations (NOWs). The use of networks (clusters) of workstations as a parallel computer has renewed the interest of the users in the domain of parallelism, but also created new challenging problems concerning the exploitation of the potential computation power offered by such a system. Most of the attempts to model these systems were in the form of programming systems rather than abstract models [4,5,13,14]. Only recently, some attempts concerning this issue appeared in the literature [1,6]. The one that we adopt here is the *hierarchical communication model* which is devoted to one of the major problems appearing in the attempt of efficiently using such architectures, the *task scheduling problem.*

The proposed model includes one of the basic architectural features of NOWs: the hierarchical communication assumption i.e. a level-based hierarchy of the communication delays with successively higher latencies.

The hierarchical model. In the *precedence constrained multiprocessor scheduling problem with hierarchical communication delays*, we are given a set of multiprocessor machines (or clusters) that are used to process n precedence constrained tasks. Each machine (cluster) comprises several identical parallel processors. A couple (c_{ij}, ϵ_{ij}) of communication delays is associated to each arc (i, j) of the precedence graph. In what follows, c_{ij} (resp. ϵ_{ij}) is called intercluster (resp. interprocessor) communication, and we consider that $c_{ij} \geq \epsilon_{ij}$. If tasks i and j are executed on different machines, then j must be processed at least c_{ij} time units after the completion of i. Similarly, if i and j are executed on the same machine but on different processors then the processing of j can only start ϵ_{ij} units of time after the completion of i. However, if i and j are executed on the same processor then j can start immediately after the end of i. The communication overhead (intercluster or interprocessor delay) does not interfere with the availability of the processors and all processors may execute other tasks.

Known results and our contribution. In [2], it has been proved that there is no hope (unless $\mathcal{P} = \mathcal{NP}$) to find a ρ-approximation algorithm with ρ strictly less than 5/4, even for the simple UET-UCT ($p_i = 1; (c_{ij}, \epsilon_{ij}) = (1, 0)$)case where an unbounded number of bi-processor machines, denoted in what follows by $\bar{P}(P2)$ is considered, $(\bar{P}(P2)|prec; (c_{ij}, \epsilon_{ij}) = (1, 0); p_i = 1|C_{\max})$. For the case where each machine contains m processors, where m is a fixed constant (i.e. for $\bar{P}(Pm)|prec; (c_{ij}, \epsilon_{ij}) = (1, 0); p_i = 1|C_{\max})$, a $\frac{4m}{2m+1}$-approximation algorithm has been proposed in [1]. However, no results are known for arbitrary processing times and/or communication delays. The small communication times (SCT) assumption where the intercluster communication delays are smaller than or equal to the processing times of the tasks, *i.e.* $\Phi = \frac{\min p_i, \ i \in V}{\max c_{kj}, \ (k,j) \in E} \geq 1$, have been

adopted in [3], where, as in [1], the interprocessor communication delays have been considered as negligible. The authors presented a $\frac{12(\Phi+1)}{12\Phi+1}$-approximation algorithm, which is based on linear programming and rounding. Notice that for the case where $c_{ij} = \epsilon_{ij}$, *i.e.* in the classical model with communication delays, Hanen and Munier [10] proposed a $\frac{2(1+\Phi)}{2\Phi+1}$-approximation algorithm for the problem with an unbounded number of machines.

In this paper, we consider for the first time the case where the number of clusters is bounded and more precisely we examine the non-approximability of the problem with two clusters composed by a set of identical processors $(P2(P)|prec; (c_{ij}, \epsilon_{ij}) = (1,0); p_i = 1|C_{max})$. In Section 2, we prove that the problem of deciding whether there is a schedule of length three is $\mathcal{NP}$-complete even for bipartite graphs i.e. for precedence graphs of depth one. This result implies that there is no polynomial time approximation algorithm with performance guarantee smaller than 4/3 (unless $\mathcal{P} = \mathcal{NP}$). In Section 3, we provide a polynomial time algorithm for the decision problem when the schedule length is equal to two, the number of clusters is constant and the number of processors per cluster is arbitrary.

2 The Non-approximability Result

In this section, we show that the problem of deciding whether an instance of $P2(P)|bipartite; (c_{ij}, \epsilon_{ij}) = (1,0); p_i = 1|C_{max}$ has a schedule of length at most three is $\mathcal{NP}$-complete. We use a polynomial time reduction from the $\mathcal{NP}$-complete problem balanced independent set (BBIS) problem [15].

Definition 1. Instance of BBIS: *An undirected balanced bipartite graph* $B = (X \bigcup Y, E)$ *with* $|X| = |Y| = n$, *and an integer* k.
Question: *Is there in* B, *an independent set with* k *vertices in* X *and* k *vertices in* Y *?*

If such an independent set exists, we call it *balanced independent set of order k*. Notice that, the problem remains $\mathcal{NP}$-complete even if $k = \frac{n}{2}$, n is even (see [15]). In what follows, we consider BBIS with $k = \frac{n}{2}$ as the source problem.

Theorem 1. *The problem of deciding whether an instance of* $P2(P)|bipartite; (c_{ij}, \epsilon_{ij}) = (1,0); p_i = 1|C_{max}$ *has a schedule of length at most three is* $\mathcal{NP}$*-complete.*

Proof. It is easy to see that the problem $P2(P)|bipartite; (c_{ij}, \epsilon_{ij}) = (1,0); p_i = 1|C_{max} \in \mathcal{NP}$. The rest of the proof is based on a reduction from $BBIS$.

Given an instance of $BBIS$, i.e. a balanced bipartite graph $B = (X \cup Y, E)$, we construct an instance of the scheduling problem $P2(P)|bipartite; (c_{ij}, \epsilon_{ij}) = (1,0); p_i = 1|C_{max} = 3$, in the following way:

– We orient all the edges of B from the tasks of X to the tasks of Y.

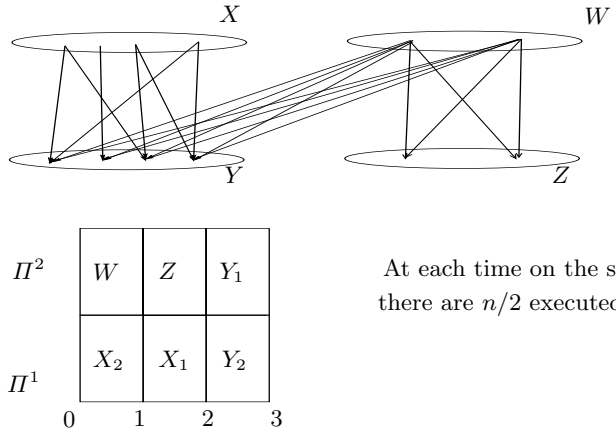

Fig. 1. The precedence graph and an associated schedule corresponding to the polynomial reduction $BBIS \propto P2(P)|bipartite; (c_{ij}, \epsilon_{ij}) = (1, 0); p_i = 1|C_{max}$.

– We add two sets of tasks: $W = \{w_1,\ w_2, \ldots,\ w_{n/2}\}$ and $Z = \{z_1,\ z_2, \ldots,\ z_{n/2}\}$. The precedence constraints among these tasks are the following:

$$w_i \rightarrow z_j,\ \forall i \in \{1,\ 2, \ldots,\ n/2\},\ \forall j \in \{1,\ 2, \ldots,\ n/2\}.$$

– We also add the precedence constraints:

$$w_i \rightarrow y_j,\ \forall i \in \{1,\ 2, \ldots,\ n/2\},\ \forall j \in \{1,\ 2, \ldots, n\ \}.$$

– We suppose that the number of processors per cluster is equal to $m = n/2$, and that all the tasks have unit execution times.

The construction is illustrated in the first part of Figure 1. The proposed reduction can be computed in polynomial time.

Notation: The first (resp. second) cluster is denoted by Π^1 (resp. Π^2).

- Let us first consider that B contains a balanced independent set of order $\frac{n}{2}$, call it (X_1, Y_1) where $X_1 \subset X$, $Y_1 \subset Y$, and $|X_1| = |Y_1| = n/2$. Let us show now that there exists a feasible schedule in three units of time. The schedule is as follows.
  - At $t = 0$, we execute on the processors of cluster Π^1 the $n/2$ tasks of $X - X_1 = X_2$, and on the cluster Π^2 the $n/2$ tasks of W.
  - At $t = 1$, we execute on Π^1 the $n/2$ tasks of X_1 and on Π^2 the $n/2$ tasks of Z.
  - We execute at $t = 2$ on the cluster Π^2 the $n/2$ tasks of Y_1 and on the cluster Π^1 the $n/2$ tasks of $Y - Y_1 = Y_2$.

  The above way of scheduling the tasks preserves the precedence constraints and the communication delays and gives a schedule of length three, whenever there exists in B a balanced independent set of order $\frac{n}{2}$.

- Conversely, we suppose that there is a schedule of length three. We will prove that any schedule of length three implies the existence of a balanced independent set $(X'_1,\ Y'_1)$, in the graph B, where $X'_1 \subset X$, $Y'_1 \subset Y$ and $|X'_1| = |Y'_1| = n/2$.
  We make four essential observations. In every feasible schedule of length at most three:
  1. Since the number of tasks is $3n$ there is no idle time.
  2. All the tasks of W must be executed at $t = 0$, since every such task precedes $\frac{3n}{2}$ tasks, and there is only $\frac{n}{2}$ processors per cluster (n in total). Moreover, all the tasks of W must be executed on the same cluster. Indeed, if two tasks of W are scheduled at $t = 0$ on different clusters, then no task of Z or Y can be executed at $t = 1$. Thus, the length of the schedule is greater than 3 because $|Z \bigcup Y| = \frac{3n}{2}$.
     Assume w.l.o.g. that the tasks of W are executed on Π^1.
  3. No task of Y or Z can be executed at $t = 0$.
     Let X'_2 be the subset of X executed on the processors of cluster Π^2 at $t = 0$. It is clear that $|X'_2| = \frac{n}{2}$, because of point 1.
  4. No task of Y or Z can be executed at $t = 1$ on Π^2. Hence, at $t = 1$, the only tasks that can be executed on Π^2, are tasks of X, and more precisely the tasks of $X - X'_2 = X'_1$.

  Let Y'_1 be the subset of tasks of Y which have a starting time at $t = 1$ or at $t = 2$ on the cluster Π^1. This set has at least $\frac{n}{2}$ elements and together with the $\frac{n}{2}$ elements of X'_1, they have to form a balanced independent set in order the schedule to be feasible.

Corollary 1. *The problem of deciding whether an instance of* $P2(P)|bipartite; (c_{ij}, \epsilon_{ij}) = (1, 0); p_i = 1; dup|C_{max}$ *has a schedule of length at most three is* $\mathcal{NP}$*-complete.*

Proof. The proof comes directly from the one of Theorem 1. In fact, no task can be duplicated since otherwise the number of tasks would be greater than $3n$, and thus the schedule length would be greater than three.

Corollary 2. *There is no polynomial-time algorithm for the problem* $P2(P)|bipartite; (c_{ij}, \epsilon_{ij}) = (1, 0); p_i = 1|C_{max}$ *with performance bound smaller than* $\frac{4}{3}$ *unless* $\mathcal{P} = \mathcal{NP}$*.*

Proof. The proof is an immediate consequence of the Impossibility Theorem (see [9,8]).

3 A Polynomial Time Algorithm for $C_{max} = 2$

In this section, we prove that the problem of deciding whether an instance of $Pk(P)|prec; (c_{ij}, \epsilon_{ij}) = (1, 0); p_i = 1|C_{max}$ has a schedule of length at most two is polynomial by using dynamic programming. In order to prove this result, we show that this problem is equivalent to a generalization of the well known problem $P2||C_{max}$.

Theorem 2. *The problem of deciding whether an instance of* $Pk(P)|prec; (c_{ij}, \epsilon_{ij}) = (1,0); p_i = 1|C_{max}$ *has a schedule of length at most two is polynomial.*

Proof. We assume that we have $k = 2$ clusters. The generalization for a fixed $k > 2$ is straightforward. Let π be an instance of the problem $Pk(P)|prec; (c_{ij}, \epsilon_{ij}) = (1,0); p_i = 1|C_{max} = 2$. We denote by G the oriented precedence graph, and by G^* the resulting non oriented graph when the orientation on each arc is removed. In the sequel we consider that G has a depth of at most two, since otherwise the instance does not admit a schedule of length at most two. It means that $G = (X \bigcup Y, A)$ is a bipartite graph. The tasks belonging to X (resp. Y), i.e. tasks without predecessors (resp. without successors), will be called source (resp. sink) tasks. In the sequel we assume that G does not contain any tasks without successors and predecessors, i.e. isolated tasks. We shall explain how to deal with these tasks later.

Let denote W_j the j-th connected component of graph G^*. The set of tasks which belong to a connected component W_j will be called a *group of tasks* in the sequel.

Each *group of tasks* constitutes a set of tasks that have to be executed by the same cluster in order to yield a schedule within two time units. Consequently the following condition holds: there is no feasible schedule within two time units, if there exists a *group of tasks* W_j such that $|W_j \bigcap X| \geq m+1$, or $|W_j \bigcap Y| \geq m+1$. Recall that m denotes the number of processors per cluster.

The problem of finding such a schedule can be converted to a variant of the well known $P2||C_{max}$ problem. We consider a set of n jobs $\{1, 2, \ldots n\}$. Each job j has a couple of processing times $p_j = (p_j^1, p_j^2)$. We assume that $\sum_{j=1}^{n} p_j^1 \leq 2m$ and $\sum_{j=1}^{n} p_j^2 \leq 2m$. The goal is to find a partition $(S, \overline{S})$ of the jobs such that the makespan is at most m if we consider either the first or second processing times, i.e. determine $S \subset \{1, 2, \ldots n\}$ such that $\sum_{j \in S} p_j^1 \leq m$, $\sum_{j \in S} p_j^2 \leq m$, $\sum_{j \in \overline{S}} p_j^1 \leq m$ and $\sum_{j \in \overline{S}} p_j^2 \leq m$.

Now, to each *group of tasks* W_j we can associate a job with processing times $p_j^1 = |W_j \bigcap X|$ and $p_j^2 = |W_j \bigcap Y|$. The Figure 2 presents the transformation between the problem $P2(P)|prec; (c_{ij}, \epsilon_{ij}) = (1,0); p_i = 1|C_{max}$ and the variant of $P2||C_{max}$.

The problem $P2||C_{max}$ can be solved by a pseudo-polynomial time dynamic programming algorithm [11]. In the sequel we show that there exists a polynomial time algorithm for the problem we consider.

Let us define $I(j, z_1, z_2) = 1$, with $1 \leq j \leq n$, $0 \leq z_1, z_2 \leq m$, if there exists a subset of jobs, $S(j, z_1, z_2) \subseteq \{1, 2, \ldots, j-1, j\}$, for which the sum of processing times on the first (resp. second) coordinate is exactly z_1 (resp. z_2). Otherwise $I(j, z_1, z_2) = 0$.

The procedure basically fills the $0-1$ entries of a n by $(m+1)^2$ matrix row by row, from left to right. The rows (resp. columns) of the matrix are indexed by j (resp. (z_1, z_2)). Initially we have $I(1, p_1^1, p_1^2) = 1$, $S(1, p_1^1, p_1^2) = \{1\}$, and $I(1, z_1, z_2) = 0$ if $(z_1, z_2) \neq (p_1^1, p_1^2)$. The following relations are used to fill the matrix:

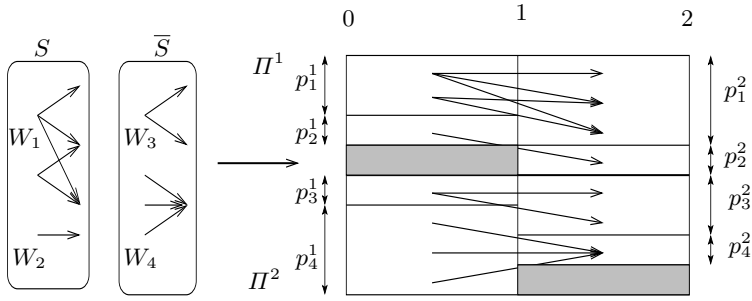

Fig. 2. llustration of the transformation with $m = 4$ (idle time is in grey).

- If $I(j, z_1, z_2) = 1$ then $I(j+1, z_1, z_2) = 1$. Moreover $S(j+1, z_1, z_2) = S(j, z_1, z_2)$.
- If $I(j, z_1, z_2) = 1$ then $I(j+1, z_1 + p^1_{j+1}, z_2 + p^2_{j+1}) = 1$. Moreover $S(j+1, z_1 + p^1_{j+1}, z_2 + p^2_{j+1}) = S(j, z_1, z_2) \bigcup \{j+1\}$.

Now, we examine the last row of the matrix, and look for a state (n, m_1, m'_1) such that $I(n, m_1, m'_1) = 1$, with $|X| - m \leq m_1 \leq m$ and $|Y| - m \leq m'_1 \leq m$. It is easy to see that the instance π admits a schedule within two time units if and only if there exists such a state. From such a state (n, m_1, m'_1) we can find a schedule of length at most two in the following way. Let W (resp. $\overline{W}$) the set of *group of tasks* associated with jobs in $S(n, m_1, m'_1)$ (resp. $\overline{S(n, m_1, m'_1)}$). The $m_1 \leq m$ source (resp. $|X| - m_1 \leq m$ sink) tasks of W are scheduled on the first cluster, during the first (resp. second) unit of time. The $m'_1 \leq m$ source (resp. $|Y| - m'_1 \leq m$ sink) tasks of $\overline{W}$ are scheduled on the second cluster, during the first (resp. second) unit of time.

In the case where the graph G contains a set of isolated tasks, we remove those tasks from set X, compute the previous matrix, and look for the same state as before. The instance π admits a schedule within two time units if and only we can fill the gaps of the previous schedule with the isolated tasks.

For $k > 2$ clusters we consider the $Pk||C_{max}$ scheduling problem in which each job has a couple of processing times. The goal is to find a partition $(S_1, \ldots, S_{k-1}, \overline{S_1 \cup \ldots \cup S_{k-1}})$ of the jobs such that the makespan is at most m if we consider either the first or second processing times. As before this problem can be solved by a pseudo-polynomial time dynamic programming algorithm using the states $(j, z_1, z_2, \ldots z_{2(k-1)})$, with $1 \leq j \leq n$ and $1 \leq z_i \leq m$, $i = 1, \ldots, 2(k-1)$. We have $I(j, z_1, z_2, \ldots z_{2(k-1)}) = 1$ if there exists a partition $(S_1, \ldots, S_{k-1})$ of jobs such that $\sum_{j \in S_{2l+1}} p^1_j = z_{2l+1}$ and $\sum_{j \in S_{2l+2}} p^2_j = z_{2l+2}$ for $0 \leq l \leq k-2$.

Let us now evaluate the running time of the overall algorithm for a problem instance with m processors per cluster (m is part of the input of the instance).

Lemma 1. *The complexity of the algorithm is equal to* $\mathcal{O}(nm^{2(k-1)})$.

Proof. Each state of the dynamic programming algorithm is a tuple $(j, z_1, z_2, \ldots z_{2(k-1)})$, with $1 \leq j \leq n$ and $1 \leq z_i \leq m$, $i = 1, \ldots, 2(k-1)$.

The number of such states is $\mathcal{O}(nm^{2(k-1)})$ and the computation at each state needs a constant time.

References

1. E. Bampis, R. Giroudeau, and J.-C. König. A heuristic for the precedence constrained multiprocessor scheduling problem with hierarchical communications. In H. Reichel and S. Tison, editors, *Proceedings of STACS*, LNCS No. 1770, pages 443–454. Springer-Verlag, 2000.
2. E. Bampis, R. Giroudeau, and J.C. König. On the hardness of approximating the precedence constrained multiprocessor scheduling problem with hierarchical communications. Technical Report 34, LaMI, Université d'Évry Val d'Essonne, to appear in RAIRO Operations Research, 2001.
3. E. Bampis, R. Giroudeau, and A. Kononov. Scheduling tasks with small communication delays for clusters of processors. In *SPAA*, pages 314–315. ACM, 2001.
4. S.N. Bhatt, F.R.K. Chung, F.T. Leighton, and A.L. Rosenberg. On optimal strategies for cycle-stealing in networks of workstations. *IEEE Trans. Comp.*, 46:545–557, 1997.
5. R. Blumafe and D.S. Park. Scheduling on networks of workstations. In *3d Inter Symp. of High Performance Distr. Computing*, pages 96–105, 1994.
6. F. Cappello, P. Fraignaud, B. Mans, and A. L. Rosenberg. HiHCoHP-Towards a Realistic Communication Model for Hierarchical HyperClusters of Heterogeneous Processors, 2000. to appear in the Proceedings of IPDPS'01.
7. B. Chen, C.N. Potts, and G.J. Woeginger. A review of machine scheduling: complexity, algorithms and approximability. Technical Report Woe-29, TU Graz, 1998.
8. P. Chrétienne and C. Picouleau. Scheduling with communication delays: a survey. In P. Chrétienne, E.J. Coffman Jr, J.K. Lenstra, and Z. Liu, editors, *Scheduling Theory and its Applications*, pages 65–90. Wiley, 1995.
9. M.R. Garey and D.S. Johnson. *Computers and Intractability, a Guide to the Theory of NP-Completeness*. Freeman, 1979.
10. A. Munier and C. Hanen. An approximation algorithm for scheduling dependent tasks on m processors with small communication delays. In *IEEE Symposium on Emerging Technologies and Factory Automation*, Paris, 1995.
11. M. Pinedo. *Scheduling : theory, Algorithms, and Systems*. Prentice Hall, 1995.
12. V.J. Rayward-Smith. UET scheduling with unit interprocessor communication delays. *Discr. App. Math.*, 18:55–71, 1987.
13. A.L. Rosenberg. Guidelines for data-parallel cycle-stealing in networks of workstations I: on maximizing expected output. *Journal of Parallel Distributing Computing*, pages 31–53, 1999.
14. A.L. Rosenberg. Guidelines for data-parallel cycle-stealing in networks of workstations II: on maximizing guarantee output. *Intl. J. Foundations of Comp. Science*, 11:183–204, 2000.
15. R. Saad. Scheduling with communication delays. *JCMCC*, 18:214–224, 1995.