V. J. RAYWARD-SMITH
D. REBAINE

Open shop scheduling with delays

Informatique théorique et applications, tome 26, nº 5 (1992), p. 439-447
<http://www.numdam.org/item?id=ITA_1992__26_5_439_0>

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 26, n° 5, 1992, p. 439 à 448)

OPEN SHOP SCHEDULING WITH DELAYS (*)

by V. J. RAYWARD-SMITH ([1]) and D. REBAINE ([2])

Communicated by K. APT

Abstract. – *The concept of interprocessor delay is introduced to the open shop model. Delays are uniform if they are always the same for any job and between any pair of machines. Scheduling an open shop with uniform delays is shown to be NP-complete even for two machines. However, if all tasks are unit execution time and the delays are uniform then a polynomial algorithm to solve the decision problem is exhibited. If the delays are nonuniform, the problem remains NP-complete.*

Keywords: Open shop; discrete scheduling; interprocessor delay.

Résumé. – *Le concept des temps interopératoires est introduit dans un atelier open-shop. Ces derniers sont dits uniformes s'ils sont identiques pour chaque tâche et quelle que soit la paire de machines utilisée. L'ordonnancement d'un atelier open shop avec des temps interopératoires uniformes se révèle NP-complet, même pour deux machines. Cependant, si les durées d'exécution des tâches sont unitaires et les temps interopératoires sont uniformes, un algorithme polynomial est disponible pour résoudre le problème de décision correspondant. Si les temps interopératoires sont non uniformes, le problème reste NP-complet.*

Mots clés : Ordonnancement; atelier open shop; temps interopératoires.

1. INTRODUCTION

The usual model of open shop scheduling involves a number, $m \in Z^+$, of machines and a set $J = \{1, 2, \ldots, n\}$ of jobs. Each job $j \in J$ has to be processed on each machine and thus comprises m tasks, $t_{1j}, t_{2j}, \ldots, t_{mj}$, where t_{ij} has to be processed on machine i. For each task, t_{ij}, there is an associated length, $l_{ij} \in Z^+$, denoting the time taken by machine i to process that task. In open shop scheduling the tasks of any job can be processed in any order and we seek a nonpreemptive schedule of all the jobs which minimizes some measure, most usually the overall completion time.

Thus, we seek a schedule σ: $\{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\} \to Z^+$ such that

(*) Received February, 1991, revised September 1991.
([1]) School of Information Systems, University of East Anglia, Norwich NR4 7TJ, UK.
([2]) Institut d'Informatique, Université de Tizi Ouzou, Algérie.

0988-3754/92/05 439 10/$3.00/© AFCET-Gauthier-Villars

(1) $\sigma(i, j) > \sigma(i, k) \Rightarrow \sigma(i, j) \geqq \sigma(i, k) + l_{ik}$ for $1 \leqq i \leqq m$, $1 \leqq j$, $k \leqq n$ {this ensures pre-emption is unnecessary},

(2) $0 \leqq \sigma(h, j) - \sigma(i, j) < l_{ij} \Rightarrow h = i$ for $1 \leqq h$, $i \leqq m$, $1 \leqq j \leqq n$ {this ensures two tasks of one job are never processed at the same time}, and

(3) subject to (i) and (ii), the completion time

$$C = \max \{ \sigma(i, j) + l_{ij} \mid 1 \leqq i \leqq m,\ 1 \leqq j \leqq n \}$$

is minimized.

Open shop scheduling is shown to be *NP*-complete in [Gonzalez and Sahni, 1976] even for the case $m=3$. It is *NP*-complete in the strong sense for arbitrary m [Lenstra, 1977]. However, for the special case $m=2$, and, similarly, for pre-emptive scheduling on an arbitrary number of machines, there exist polynomial time algorithms [Gonzalez and Sahni, 1976].

In this paper, we introduce the concept of an interprocessor time delay. This models the situation where the machines are remote and there is a time delay when a job is transferred from one machine to another. Job shop scheduling with such delays has been discussed in [Hwang *et al.*, 1989; Rayward-Smith, 1987 *a*; Rayward-Smith, 1987 *b*].

Let d_{jhi} denote the time delay encountered in transferring job j from machine h to machine i. Condition (2) above is then replaced by

$$(2') \qquad 0 \leqq \sigma(h, j) - \sigma(i, j) < l_{ij} + d_{jih} \Rightarrow h = i$$

and any schedule satisfying (1) and (2′) is called valid. The delays are *symmetric* if $d_{jhi} = d_{jih}$ for all $1 \leqq j \leqq n$, $1 \leqq i$, $h \leqq m$, and *uniform* if $d_{jhi} = d$ for all $1 \leqq j \leqq n$, $1 \leqq i$, $h \leqq m$.

By setting $d_{jhi} = 0$ for all $1 \leqq j \leqq n$, $1 \leqq i$, $h \leqq m$, it follows immediately from the *NP*-completeness of open shop scheduling that open scheduling with uniform delays is *NP*-complete for any fixed $m \geqq 3$. In section 2, we prove the stronger result that open shop scheduling with delays is *NP*-complete even with uniform delays and $m=2$. Moreover, for UET (unit execution time) tasks, open shop scheduling with symmetric delays on an arbitrary number of machines is *NP*-complete. In Section 3, we present a polynomial time algorithm to solve the decision problem in the special case where m is arbitrary and all delays are uniform but all the tasks are UET. Section 4 is our conclusion.

2. *NP*-COMPLETENESS RESULTS

The decision problem associated with scheduling a two processor open shop with uniform delays is as follows.

Open shop with two machines and uniform delays (02*D*)

Instance: An open shop with two machines; a set of n jobs, each job, j, having an associated pair of positive integers, (l_{1j}, l_{2j}), where l_{ij} is the time to process the job on machine i $(i=1, 2)$; a uniform delay, $d \in Z^+$ and a bound $L \in Z^+$.

Question: Is there a valid open shop schedule of the n jobs on the two machines with uniform delay d and completion time $\leqq L$?

In this section, we will prove 02*D* is *NP*-complete but before we do so, we need to establish a lemma.

For each job, $j \in J$, a schedule is going to process one task earlier than the other – such a task is called a first task. The remaining task is a second task.

LEMMA 1: *There exists an optimal schedule where for each machine* $i=1, 2$, *there exists a time* T_i *such that for all* $t \leqq T_i$, *machine i processes first tasks and for all* $t > T_i$, *machine i processes second tasks.*

Proof: Consider an optimal schedule which does not satisfy the statement of the lemma. Without loss of generality, assume machine 1 processes the second task of job j immediately before the first task of job k. These can be interchanged still obtaining a valid optimal schedule. Repeating this argument establishes the lemma.

A schedule that satisfies the condition of Lemma 1 is called a *staged schedule.*

THEOREM 1: 02*D is NP-complete.*

Proof: $02D \in NP$ since we can guess a schedule, σ, and, in polynomial time, check whether it is valid.

The theorem is thus established by proving $\Pi \ \alpha \ 02D$ for some known *NP*-complete problem, Π. In this case, we choose PARTITION, proved *NP*-complete in [Karp, 1972].

Partition

Instance: Finite set, A, and a size $s(a) \in Z^+$ for each $a \in A$.

Question: Is there a subset $A' \subseteq A$ such that

$$\Sigma\{s(a) \mid a \in A'\} = \Sigma\{s(a) \mid a \in A - A'\}?$$

Without loss of generality, we can assume all instances of PARTITION have $\Sigma\{s(a) \mid a \in A\} = 2M$ for some integer, M; otherwise the instance is trivially a *NO* instance. Moreover, by introducing a scaling factor if necessary, we can safely assume each $s(a) \geqq 2$. We then define $f: D_{\text{PARTITION}} \to D_{02D}$ by mapping each a of size $s(a)$ into job j_a. The pair associated with job j_a is $(s(a), s(a))$, so the job needs the same amount of processing time on each machine. We also introduce four other jobs which we will call the capital jobs, denoted by V, W, X, Y. The associated pairs are $(1, M)$, $(1, M)$, $(M, 1)$ and $(M, 1)$, respectively. Thus, given an instance, I, of PARTITION, the constructed instance $f(I)$ of $02D$ comprises these $n+4$ jobs, a uniform delay $d = 2M$ and a bound $L = 4M + 2$. We need to establish that f is a polynomial transformation. Clearly f can be computed in polynomial time so all we need show is that $I \in Y_{\text{PARTITION}}$ iff $f(I) \in Y_{02D}$.

$I \in Y_{\text{PARTITION}} \Rightarrow f(I) \in Y_{02D}$: Assume A' is such that

$$\Sigma\{s(a) \mid a \in A'\} = \Sigma\{s(a) \mid a \in A - A'\} = M.$$

Then a valid schedule for f with delay $2M$ and of length $4M+2$ is given in figure 1. Since this satisfies the bound, $f(I) \in Y_{02D}$.

0 1 M+1 2M+1 3M+1 4M+1 4M+2

V	elements in A'	Y	X	elements in A-A'	W
X	elements in A-A'	W	V	elements in A'	Y

Figure 1

$f(I) \in Y_{02D} \Rightarrow I \in Y_{\text{PARTITION}}$: Consider the valid schedule of $02D$ which completes in time $\leqq L = 4M + 2$. Since both machines 1 and 2 have tasks of length $4M+2$ to process, the schedule must have no idling whatsoever. By Lemma 1, we can assume this optimal schedule is a staged schedule.

In order that the schedule can be completed in time $4M+2$, T_1 and T_2 must both satisfy

$$T_i \leqq 4M+2-d-1=2M+1.$$

Also, since we have no idling and no second task can possibly be processed before time $2M+1$, $T_i \geqq 2M+1$. Hence we deduce $T_1=T_2=2M+1$.

The only tasks of size 1 are tasks associated with capital jobs. Thus if a first task is to complete at time $2M+1$ it must be the large task associated with a capital job. Similarly if a second task is to start at time $2M+1$, it also must be the large task associated with a capital job. The only valid schedule must thus have the capital tasks filling the shaded area of the Gantt chart of figure 2.

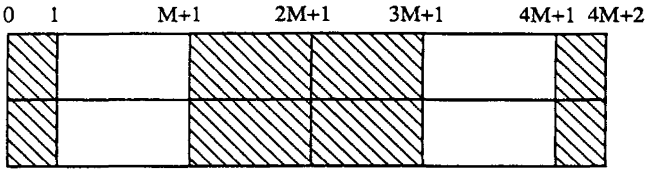

Figure 2

The remaining jobs must be processed at the unshaded times. Let $A'=\{a \mid j_a$ is a first job on machine $1\}$. Then $\Sigma\{s(a) \mid a \in A'\}=M$ so $I \in Y_{\text{PARTITION}}$.

Thus, we have established a polynomial transformation from PARTITION to $02D$ and the proof of the theorem is completed.

When the tasks are all UET and the delays are uniform, we can solve the decision problem in polynomial time (*see* Section 3). However, if the tasks are UET and the delays are nonuniform, the problem remains *NP*-complete. Formally stated, the decision problem is as follows.

Open shop with UET tasks and nonuniform delays *(OmDU)*

Instance: An open shop with m machines; a set of n jobs, each job, j, requiring unit processing on each $i\,(1 \leqq i \leqq m)$; delays $d_{jhi} \in Z^+$ $(1 \leqq j \leqq n,\ 1 \leqq h,\ i \leqq m)$; and a bound $L \in Z^+$.

Question: Is there a valid open shop schedule of the n jobs on the m machines with the given delays and completion time $\leqq L$?

THEOREM 2: *OmDU is NP-complete even if the delays are symmetric.*

Proof: The proof uses a simple polynomial transformation from Hamiltonian Path, proved *NP*-complete in [Karp, 1972].

Hamiltonian Path (HP)

Instance: Graph $G=(V, E)$.

Question: Does G contain a Hamiltonian path?

Given an instance I in D_{HP}, the constructed instance of *OmDU* has $m=|V|$ machines and just one job, j_1, comprising m UET tasks. The symmetric delays are defined by $d_{1ij}=0$ if $(v_i, v_j)\in E$ and m otherwise. It is then routine to show that there exists a schedule of length m iff there exists a Hamiltonian path in G. Since the transformation is polynomial time and the scheduling problem is clearly in *NP*, the proof is complete.

3. UET TASKS

In this section we consider the case where we have m machines, all n tasks are UET and the delays are uniform, *i.e.* $d_{jih}=d$ for all $1\leqq h,\ i\leqq m,\ 1\leqq j\leqq n$. In this special case, we are able to exhibit a polynomial time algorithm to solve the corresponding decision problem.

LEMMA 2: *If $m(d+1)\leqq n$ then the optimal schedule involves no idling and can be obtained by a simple allocation algorithm.*

Proof: We define $\sigma:\{1, 2, \ldots, m\}\times\{1, 2, \ldots, n\}\rightarrow Z^+$ by

$$\sigma(i, j)=(j-1+(i-1)(d+1))\ mod\ n.$$

Such a schedule is based on a repeated cyclic shift of the allocation on machine 1 by d. It is easy to confirm that σ is valid provided $m(d+1)\leqq n$ and has completion time of n. Figure 3 illustrates a schedule constructed using this function with $m=4$, $d=2$ and $n=12$. The schedule is presented using a Gantt chart, G, related to σ by $G[i, k]=j$ iff $\sigma(i, j)=k$.

Now, we consider the case where $f=m(d+1)-n>0$. We introduce f fictitious jobs which we number $n+1,\ n+2,\ \ldots,\ n+f$. We then schedule these $n+f$ jobs using the following algorithm to define σ.

machine \ time	0	1	2	3	4	5	6	7	8	9	10	11
1	1	2	3	4	5	6	7	8	9	10	11	12
2	10	11	12	1	2	3	4	5	6	7	8	9
3	7	8	9	10	11	12	1	2	3	4	5	6
4	4	5	6	7	8	9	10	11	12	1	2	3

Figure 3

```
t := 0;
for p := 0 to m−1 do
for q := 0 to d do
    begin
    for r := 0 to m−1 do
        begin
        i := (r+p) mod m+1;
        j := r+mq+1;
        σ(i, j) := t
        end;
    t := t+1
    end
end
```

The range of σ defined by this algorithm is clearly $\{0, \ldots, n+f-1\}$. Moreover, it is relatively easy to check that σ is valid according to the

machine \ time	0	1	2	3	4	5	6	7	8
1	1	4	7	3	6	9	2	5	8
2	2	5	8	1	4	7	3	6	9
3	3	6	9	2	5	8	1	4	7

Figure 4

definitions of Section 1. In figure 4, we illustrate its use when $m=3$, $d=2$ and $n+f=9$.

We can now prove

LEMMA 3: *If $m(d+1)>n$ then there exists a simple scheduling algorithm of optimal length $(m-1)(d+1)+\lceil n/m \rceil$.*

Proof: We schedule $n+f=m(d+1)$ jobs using the scheduling algorithm described above. Every task of a fictitious job is replaced by an idle period. The length of the resulting schedule is taken to be $\max\{\sigma(i,j)+1 \mid \leqq i \leqq m, 1 \leqq j \leqq n\}$ rather than $\max\{\sigma(i,j)+1 \mid 1 \leqq i \leqq m, 1 \leqq j \leqq +f\}$, and this will equal $(m-1)(d+1)+\lceil n/m \rceil$. Any schedule of n jobs on m machines must involve some job not having any of its m tasks completed until time $\lceil n/m \rceil$. The earliest completion time of such a job is thus $\lceil n/m \rceil + (m-1)(d+1)$. Hence a schedule that achieves this length is optimal.

machine \ time	0	1	2	3	4	5	6	7
1	1	4		3			2	5
2	2	5		1	4		3	
3	3			2	5		1	4

Figure 5

Figure 5 illustrates an optimal schedule of 5 jobs on $m=3$ machines when $d=2$. It is constructed from the schedule of figure 4.

Since a constructed schedule in the case where $m(d+1)>n$ involves either allocating activities or idling instructions to $m^2 d$ machine time slots, the best possible algorithm to construct the Gantt chart is $O(m^2 d)$. This is the order of the above algorithm.

The following is now a trivial result.

THEOREM 3: *OmDU with uniform delays is in P.*

Proof: The algorithm to solve the problem is as follows.

if (if $m(d+1)\leqq n$ then $L \leqq n$ else $L \leqq (m-1)(d+1)+n$ div m) then YES else NO.

4. CONCLUSION

We have extended the open shop model to include interprocessor delays. Even when the delays are uniform, finding optimal schedules is *NP*-hard for any fixed $m \geqq 2$. It is trivial to show this result also holds for nonuniform delays.

However, in the special case when delays are uniform and all the tasks are UET, the length of the optimal schedule can be found using a simple formula.

When the tasks are UET but the delays are nonuniform, the decision problem is *NP*-complete for arbitrary m. It is open whether this result holds for fixed m. However, if $d = \max\{d_{jih}: 1 \leqq j \leqq n,\ 1 \leqq i,\ h \leqq m\}$ and $m(d+1) \leqq n$ then the algorithm of Lemma 2 can be used to find an optimal schedule in pseudo-polynomial time, $O(nm)$.

REFERENCES

T. Gonzalez and S. Sahni, Open Shop Scheduling to Minimize Finish Time, *J. A.C.M.*, 1976, 23, pp. 665-679.

J. J. Hwang, Y. C. Chow, F. D. Angers and C. Y. Lee, Scheduling Precedence Graphs in Systems with Interprocessor Communications Times, *S.I.A.M. Comput*, 1989, 18, pp. 244-257.

R. M. Karp, Reducibilities Among Combinational Problems in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher Eds, *Plenum Press*, New York, 1972.

J. K. Lenstra, Private Communication, cited in M. R. Garey and D. S. Johnson Eds, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, San Francisco, 1979.

V. J. Rayward-Smith, The complexity of Preemptive Scheduling Given Interprocessor Communication Delays, *Inform. Proc. Letters*, 1987 *a*, 25, pp. 123-125.

V. J. Rayward-Smith, UET Scheduling with Unit Inerprocessor Communication Delays, *Discrete Appl. Math.*, 1987 *b*, 18, pp. 55-71.