Belgian Journal of Operations Research, Statistics and Computer Science Vol. 40 (1-2) 00

SCHEDULING ON A BATCH MACHINE WITH JOB COMPATIBILITIES

Mourad BOUDHAR [a] and Gerd FINKE [b]

[a] Institut de Mathématiques, Université USTHB, B.P. 32,
Bab-Ezzouar, El-Alia 16111, Alger, Algérie
E-mail: mourad.boudhar@imag.fr

[b] Laboratoire Leibniz, Institut Imag, 46 avenue Félix Viallet,
38031 Grenoble Cedex, France
E-mail: gerd.finke@imag.fr

Abstract

We consider the problem of minimizing the makespan on a single batch processing machine, in which jobs are not all compatible, i.e. there are at least two jobs that can not be processed simultaneously in the same batch. The capacity of the batch processing machine can be finite or infinite. The processing time of a batch is given by the processing time of the longest jobs in the batch. We prove NP-hardness of the general problem and present polynomial algorithms for several special cases.

Keywords

Batch scheduling, batch processing machine, compatibility graph.

1 Introduction

The batch scheduling is an extension of other scheduling models that refers to batches of jobs to be processed together (material heated together in a furnace, material painted together, material rolled together, material transported together by an AGV, etc...). In these cases, the jobs in the same batch have to be compatible (similar physical or chemical properties, forms, colors, weights, etc...).

The principal motivation for batch scheduling is the scheduling of burn-in operations in the semiconductor industry. The final stage in the production of integrated circuits is the burn-in operation, in which chips are loaded onto boards which are then placed in an oven and exposed to high temperatures. The purpose of burn-in operations is to subject the chips to thermal stress for an extended period of time in order to bring out latent defects.

In this paper, we consider the problem of scheduling multiple independent jobs $J_1, \ldots, J_n$ (n is the number of jobs) on a single batch processing machine $B1$ in order to minimize the makespan C_{max} (final completion time). We suppose that there exists a relation of compatibility between jobs, where two jobs are said to be compatible if they can be processed simultaneously in the same batch. This relation is represented by a graph $G = (V, E)$ where V is the set of jobs and any pair of jobs is in E if and only if they are compatible. The capacity of the batch processing machine can be finite (it can process up to b jobs simultaneously) or infinite (it can process any number of jobs simultaneously). Job J_i has the processing time p_i and the release date r_i. The processing time of a batch is equal to the maximum processing time of any job assigned to it. All jobs in a batch must be available and start and finish at the same date. Preemption is not allowed.

This new combination of batch processing with a compatibility graph defined for the jobs, appears to be an interesting concept relating scheduling and graph theory.

There is no literature discussing batch scheduling problems of these types. Brucker et al. [8] have proved that the problem with any processing times, any release dates, $b = 2$ and where all jobs are compatible (graph G is complete) is NP-hard in the strong sense. In their paper, the authors do not introduce the notion of compatibility graphs. They have shown that the equivalent mirror image problem of minimizing the maximum lateness is NP-hard in the strong sense. Some related very special cases, e.g. the IF (incompatible families) structures have previously been treated [23]. The notion of a compatibility graph is not introduced.

For other optimality criteria see [11, 18, 20, 23] for the IF (incompatible families) structures and [8, 10, 17, 18] for the CF (compatible families) structures.

This article is organized as follows. In section 2 we give the problem classification and present an illustrative example. In section 3, problems with general compatibility graphs are considered. In section 4, we analyze problems with special compatibility graphs. Section 5 is a brief conclusion.

2 Classification, notations and illustrative example

The usual three-field notation $\alpha/\beta/\gamma$ is modified to include batch processing machines. Throughout this article, the performance criterion $\gamma = C_{max}$ is used. We set $\alpha = Bm$ to indicate m parallel batch processing machines. The vector β is enriched by the following two parameters β_1' and β_2' (which are given at the beginning of the $\beta - field$): $\beta_1' \in \{\emptyset, G = (V, E)\}$ where $\emptyset$ refers to the complete graph and G is a given graph); $\beta_2' \in \{b, b = k, b \geq n\}$ characterizes the batch capacity where b means variable capacity, $b = k$ is the constant capacity of size k and $b \geq n$ refers to an infinite capacity. Also, for any schedule of jobs on batch processing machines, let $rb_j = \max_{J_i \in B_j}\{r_i\}$ be the release date of batch B_j, $pb_j = \max_{J_i \in B_j}\{p_i\}$ the processing time of batch B_j, db_j the starting time of batch B_j, Cb_j the completion time of batch B_j, D_i the starting time of job J_i and C_i the completion time of job J_i (which is equal to Cb_j if $J_i \in B_j$).

Example 1 *Let us process 5 jobs J_1, J_2, J_3, J_4 and J_5 on a single batch processing machine. The processing times are given in table 1.*

J_i	J_1	J_2	J_3	J_4	J_5
p_i	1	1	3	4	1

Table 1. Example 1

First case: Problem $B1/b = 2/C_{max}$ (all jobs are compatible)
We obtain the following optimal solution:
The number of batches is equal to 3;
$B_1 = \{J_1, J_2\}$; $pb_1 = max\{1, 1\} = 1$;
$B_2 = \{J_3, J_4\}$; $pb_2 = max\{3, 4\} = 4$;
$B_3 = \{J_5\}$; $pb_3 = 1$;
$C_{max} = 6$.
The schedule is in figure 1.

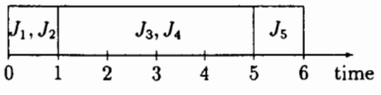


Figure 1: Optimal schedule of first case

Second case: Problem $B1/G = (V, E), b = 2/C_{max}$
where $V = \{1, 2, 3, 4, 5\}$ and $E = \{(1, 2), (1, 3), (2, 3), (2, 4), (4, 5)\}$ (see figure 2)

We obtain the following optimal solution:
The number of batches is equal to 3;
$B_1 = \{J_1, J_3\}$; $pb_1 = max\{1, 3\} = 3$;
$B_2 = \{J_2, J_4\}$; $pb_2 = max\{1, 4\} = 4$;
$B_3 = \{J_5\}$; $pb_3 = 1$;
$C_{max} = 8$.
The schedule is in figure 3.

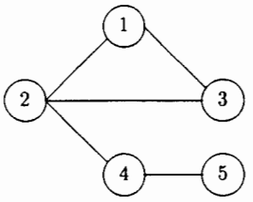


Figure 2: The graph G=(V,E)

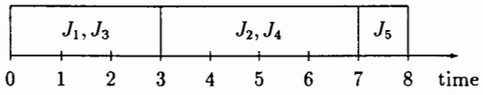


Figure 3: Optimal schedule of second case

Third case: Problem $B1/G = (V, E), b = 2, r_i/C_{max}$ with the same compatibility graph (figure 2) and the release dates given in table 2.

J_i	J_1	J_2	J_3	J_4	J_5
r_i	0	1	1	5	5

Table 2. Release dates of example 1

We obtain the following optimal solution:
The number of batches is equal to 3;
$B_1 = \{J_1\}$; $pb_1 = 1$; $rb_1 = 0$;
$B_2 = \{J_2, J_3\}$; $pb_2 = max\{1, 3\} = 3$; $rb_2 = max\{1, 1\} = 1$;
$B_3 = \{J_4, J_5\}$; $pb_3 = max\{4, 1\} = 4$; $rb_3 = max\{5, 5\} = 5$;
$C_{max} = 9$.
The schedule is in figure 4.

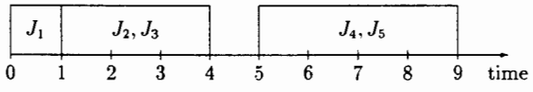


Figure 4: Optimal schedule of third case

It should be noted that the number of batches is not fixed, but should be determined.

3 General compatibility graphs

The problem $Bm/G = (V, E), b = 1/C_{max}$ is equivalent to the parallel machine problem $Pm//C_{max}$. Consequently, these batch scheduling problems are NP-hard for $m \geq 2$. It re-

mains the class of single batch machine scheduling problems with $b \geq 2$. Note that the case $b = 1$ coincides with the classical single machine scheduling problems and does not provide any new insight.

Theorem 1 *[6] The problem* $B1/G = (V, E), b = k, p_i = 1/C_{max}$ *with* $k \geq 3$ *is NP-Hard.*

Efficient heuristics and exact algorithms based on job orderings are presented, with numerical results, in [6] to solve the problem $B1/G = (V, E), b = k, p_i = 1/C_{max}$ where $k = 3, 4, \ldots, \infty$. Extensions to arbitrary processing times, to solve the problem $B1/G = (V, E), b = k/C_{max}$ where $k = 3, 4, \ldots, \infty$, are also discussed.

Let us analyze the batch capacity $b = 2$.

Let A be the vertex-edge incidence matrix of the graph $G = (V, E)$ where

$$a_{ij} = \begin{cases} 1 & \text{if the edge } j \text{ is incident to the vertex } i \\ 0 & \text{if not} \end{cases}$$

for $i = 1 \ldots n$ and $j = 1 \ldots q$ (q is the number of edges).

Theorem 2 *The problem* $B1/G = (V, E), b = 2/C_{max}$ *reduces to the maximum weight matching.*

Proof. Let $c_j = max\{p_{s_j}, p_{t_j}\}$ be the cost of edge j, if the edge j is incident to the vertices s_j and t_j.

The linear model, corresponding to this problem, is:

$$\min C_{max} = (\sum_{j=1}^{q} c_j x_j) + \sum_{i=1}^{n} (1 - \sum_{j=1}^{q} a_{ij} x_j) p_i$$

$$\text{subject to} \begin{cases} \sum_{j=1}^{q} a_{ij} x_j \leq 1 \quad \text{for } i = 1, \ldots, n \\ x_j \in \{0, 1\} \quad \text{for } j = 1, \ldots, q \end{cases}$$

where

- $x_j = \begin{cases} 1 & \text{if the jobs connected by the edge j are in the same batch} \\ 0 & \text{if not} \end{cases}$
- $\sum_{j=1}^{q} c_j x_j$: the sum of the batch processing times with two jobs.
- $\sum_{i=1}^{n} (1 - \sum_{j=1}^{q} a_{ij} x_j) p_i$: the sum of batch processing times with one job.
- $\sum_{j=1}^{q} a_{ij} x_j \leq 1$ indicates that each job is, at most, in one batch.

We have

$$\begin{aligned}
(\sum_{j=1}^{q} c_j x_j) + \sum_{i=1}^{n}(1 - \sum_{j=1}^{q} a_{ij}x_j)p_i &= (\sum_{i=1}^{n} p_i) - (\sum_{i=1}^{n}\sum_{j=1}^{q} a_{ij}x_j p_i - \sum_{j=1}^{q} c_j x_j) \\
&= (\sum_{i=1}^{n} p_i) - (\sum_{j=1}^{q}\sum_{i=1}^{n} a_{ij}p_i x_j - \sum_{j=1}^{q} c_j x_j) \\
&= (\sum_{i=1}^{n} p_i) - \sum_{j=1}^{q}(\sum_{i=1}^{n} a_{ij}p_i - c_j)x_j \\
&= (\sum_{i=1}^{n} p_i) - \sum_{j=1}^{q}(p_{s_j} + p_{t_j} - \max\{p_{s_j}, p_{t_j}\})x_j \\
&= (\sum_{i=1}^{n} p_i) - \sum_{j=1}^{q} \min\{p_{s_j}, p_{t_j}\}x_j.
\end{aligned}$$

$\sum_{i=1}^{n} p_i$ is a constant and greater than $\sum_{j=1}^{q} \min\{p_{s_j}, p_{t_j}\}x_j$. Then, minimizing C_{max} is equivalent to maximizing $\sum_{j=1}^{q} l_j x_j$ where $l_j = \min\{p_{s_j}, p_{t_j}\}$ if the edge j is incident to the vertices s_j and t_j. Hence the linear model reduces to the maximum weight matching problem. □

The following algorithm solves the problem $B1/G = (V, E), b = 2/C_{max}$.

Algorithm A1;
Begin

1- From the graph $G = (V, E)$, construct a new valued graph $H = (V, E)$ where each edge in E is valued by $min\{p_i, p_j\}$ if the edge is incident to the vertices i and j.

2- Find a maximum weight matching in the graph H.

3- Form the batches:

- for each set of the matching, process the corresponding two jobs in the same batch.
- Other jobs are processed as single job batches.

4- Execute the batches in an arbitrary order.

5- The makespan is the sum of all processing times minus the value of the maximum weight matching.

End

The maximum number of possible edges in G is $\frac{n(n-1)}{2}$ and the best known algorithm for the maximum weight matching is in $O(n^{2.5})$. Hence, also the algorithm A1 runs in $O(n^{2.5})$.

Corollary 1 *The problem $B1/G = (V, E), b = 2, p_i = 1/C_{max}$ can be reduced to the maximal cardinality matching.*

Proof. In the above proof, we have $\min\{p_{s_j}, p_{t_j}\} = 1$. Thus minimizing C_{max} is equivalent to maximizing $\sum_{j=1}^{q} x_j$ and the linear model is equivalent to the maximal cardinality matching problem. □

The following algorithm solves the problem $B1/G = (V, E), b = 2, p_i = 1/C_{max}$.

Algorithm A2;
Begin

1- Find a maximal cardinality matching in the graph G.

2- Form the following batches:

- for each set of the matching, process the corresponding two jobs in the same batch.
- Other jobs are processed in single job batches.

3- Schedule the batches in an arbitrary order (without idle time).

4- The makespan is the number of jobs minus the value of the maximal cardinality matching.

End

Also the algorithm A2 runs in $O(n^{2.5})$.

These problems become difficult if one has different release times r_i, even if the processing times are unitary.

Theorem 3 *The problem* $B1/G = (V,E), b = 2, r_i, p_i = 1/C_{max}$ *is NP-Hard in the strong sense.*

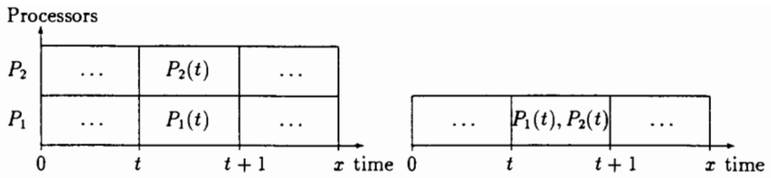
Proof. Let OCR be the decision problem corresponding to the scheduling problem under resource constraints $P2/res.11, r_i, p_i = 1/C_{max}$, defined as follow: given two processors P_1 and P_2, n jobs $J_1, \ldots, J_n$ with identical processing times equal 1 and release dates $r_1, \ldots, r_n$, and m resources $R_1, \ldots, R_m$ (each job J_i requires for its execution at most one unit of each of the resources $R_1, \ldots, R_m$). Question: Can one schedule jobs $J_1, \ldots, J_n$ on the processors P_1 et P_2 with a makespan less or equal to x ? OCR is NP-Complete in the strong sense [1, 2].

Let $R_j(i) = \begin{cases} 1 & \text{if the job } J_i \text{ requires the resource } R_j \\ 0 & \text{if not} \end{cases}$

We prove that OCR reduces polynomially to the corresponding problem OL: jobs are $J_1, \ldots, J_n$ (the same), $p_i = 1$ for $i = 1, \ldots, n$, r_i for $i = 1, \ldots, n$ (the same release dates), $G = (V,E)$ where $V = \{1, \ldots, n\}$ and $(i,j) \in E$ if and only if $R_k(i) + R_k(j) \leq 1$ for $k = 1, \ldots, m$ (i.e. the two jobs J_i and J_j do not require a same resource), $y = x$. This reduction is obviously polynomial.

If OCR has a solution with a makespan less or equal to x, then there exists an allocation of jobs $J_1, \ldots, J_n$ on processors P_1 and P_2 such that: $C_{max} \leq x, D_i \geq r_i$ for $i = 1, \ldots, n$ and $R_k(P_1(t)) + R_k(P_2(t)) \leq 1$ for $t = 0, \ldots, x-1$ and $k = 1, \ldots, m$ (where $P_j(t)$ indicates the job allocated to processor P_j at time t, if $P_j(t) = \emptyset$ then $R_k(P_j(t)) = 0$ for $k = 1, \ldots, m$).

We construct a solution for OL as follow (see figure 5): if a job is processed at time t then we form the batch $B_t = \{P_1(t), P_2(t)\}$ (the jobs $P_1(t)$ and $P_2(t)$ are compatible because $R_k(P_1(t)) + R_k(P_2(t)) \leq 1$ for $k = 1, \ldots, m$).



a/ Solution of OCR b/ Corresponding solution of OL

Figure 5: Relation between the solutions of OL and OCR

Then OL has a solution with a makespan less or equal to x.

If OL has a solution with a makespan less or equal to x, we construct a solution for OCR as follows: The jobs belonging to the batch B_j and processed at time t, are dispatched on the two processors P_1 et P_2 at time t (see figure 5), then the constraints of release dates are respected and, also, the resource constraints are respected as the jobs are compatible. We obtain a solution for OCR with a makespan less or equal to x. □

4 Special compatibility graphs

The problem $B1/G = (V, E), b \geq n, p_i = 1/C_{max}$ can be solved in polynomial time if, and only if, the problem of partitioning the graph G into the minimum number of cliques is polynomial. In fact, if s is the minimum number of cliques, then one has obviously $C_{max} = s$.

We give below a list of graphs where this problem is polynomial:

- circular-arc graphs [14].
- chordal graphs [13].
- comparability graphs [16].

and the special cases:

- split graphs (the graph and its complementary graph are chordal graphs).
- permutation graphs (the graph and its complementary graph are comparability graphs).
- interval graphs (the graph is a chordal graph and its complementary graph is a comparability graph).

In particular the case where G is an interval graph has interesting applications. In [7] an industrial problem from the sheet metal industry is given.

Let GSK3=(V,E) be a graph containing no complete subgraphs with 3 vertices (a bipartite graph is a GKS3 graph). For these graphs, the sets in each partitioning contain at most 2 vertices. Then the problem $B1/GSK3 = (V,E), b/C_{max}$ can be solved in polynomial time by the algorithm $A1$.

Let assume that one knows already a minimal partition of the graph into cliques of size b or less. Let $B_1, \ldots, B_s$ be a feasible minimum partitioning, i.e. $\mid B_k \mid \leq b$ and s is minimal. Let us denote this compatibility graph by $G(K_b^1, \ldots, K_b^s)$. Even in this case, the problem turns out to be difficult in the presence of different release times.

Theorem 4 *The problem $B1/G(K_b^1, \ldots, K_b^s), b, r_i, p_i = 1/C_{max}$ is NP-hard.*

Proof. Let CLIQUE be the following decision problem: let a graph be given $H = (V,E)$ and an integer $K > 0$. Question: Does H contain a clique of size K (a complete subgraph of K vertices)? This problem is NP-complete [12].

We prove that CLIQUE reduces polynomially to the corresponding problem OL: $n = n_1 + n_2$ where $n_1 = \mid V \mid$ and $n_2 = (K-1) \mid V \mid$, and $J = \{J_1, \ldots, J_{n_1}\} \cup \{J_{n_1+1}, \ldots, J_n\}$ (To each vertex i of V we associate a job J_i and add $(K-1) \mid V \mid$ jobs). Set
$p_i = 1$ for $i = 1, \ldots, n$,
$r_i = 0$ for $i = 1, \ldots, n_1$, $r_i = 1$ for $i = n_1 + 1, \ldots, n$,
$b = K$,
$G = (V_1 \cup V_2, E_1 \cup E_2 \cup E_3)$ (see figure 6) where,
$V_1 = \{1, \ldots, n_1\}$, $V_2 = \{n_1 + 1, \ldots, n\}$,
$(i,j) \in E_1$ if and only if $(i,j) \in E$ $(E_1 = E)$,
$(i,j) \in E_2$ if and only if $i \in V_2$, $j \in V_2$ and $i \neq j$
(The subgraph $K_{n_2} = (V_2, E_2)$ is a clique of size n_2) and
$(i,j) \in E_3$ if and only if $i \in V_1$ and $j \in V_2$.
$y = \mid V \mid$.

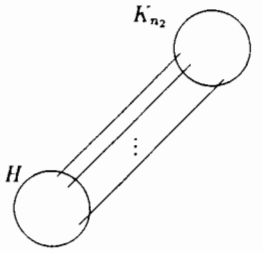


Figure 6: Construction of G

This construction is polynomial.

Since $n_1 = \mid V \mid$, $n_2 = \mid V \mid (K-1)$ and K_{n_2} is a complete graph, the graph G has a partition into $\mid V \mid$ cliques of size K (each clique is composed of one vertex of H and $K-1$ vertices of K_{n_2}). This partition is optimal because the number of vertices in the graph is equal to $K \mid V \mid$.

If CLIQUE has a solution, then there exists a clique of size K in the graph H. One constructs a solution for OL as follows. Assume, without loss of generality, that vertices of H are ordered as follows $V = \{1,\ldots,K\} \cup \{K+1,\ldots,\mid V \mid\}$ where vertices $1,\ldots,K$ form a clique. The batches are (see figure 7):

$B_1 = \{J_1,\ldots,J_K\}$,
$B_j = \{J_{K+(j-1)}\} \cup \{J_{n_1+(j-2)(K-1)+1},\ldots,J_{n_1+(j-1)(K-1)}\}$ for $j = 2,\ldots,n_1-K+1$
$B_j = \{J_{w+(j-n_1+K-2)K+1},\ldots,J_{w+(j-n_1+K-1)K}\}$ for $j = n_1-K+2,\ldots,n_1$
where $w = n_1 + (n_1-K)(K-1)$

with processing times:

$Cb_1 = 1$ and
$Cb_j = j$ for $j = 2,\ldots,n_1$.

The batches are processed in non-decreasing order of their indices.

$C_{max} = \max_{1\le i\le n}\{C_i\} = n_1 = \mid V \mid$.

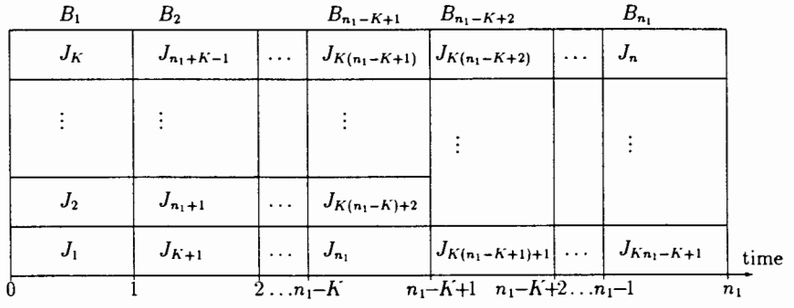


Figure 7: Construction of a solution of OL

If OL has a solution with makespan less or equal to $n_1 = \mid V \mid$, then in each batch there are exactly K jobs, because the number of jobs is equal to $n_1 + (K-1)n_1 = Kn_1$ and the processing time of any job is equal to 1. As the jobs J_i $(n_1+1 \le i \le n)$ have a release date equal to 1, they will be processed between the date 1 and the date n_1. The number of these jobs is equal to $n_1(K-1)$. Between the date 1 and the date n_1, one has to process $n_1 - K$ jobs among the remaining jobs J_i $(1 \le i \le n_1)$, because $(n_1-1)K = n_1(K-1) + (n_1-K)$. Finally, the number of jobs to be processed between the date 0 and the date 1 is K and forms necessarily a clique. Therefore CLIQUE has a solution. □

5 Conclusion

In this paper, we have established new complexity results for the problem of minimizing the makespan on a single batch processing machine, in which jobs are not all compatible. Table 3 summarizes the problem types and their complexity status.

Problem	Complexity
$B1/G = (V, E), b = k, p_i = 1/C_{max}$ $(k \geq 3)$	NP-hard
$B1/G = (V, E), b = 2/C_{max}$	$O(n^{2.5})$
$B1/G = (V, E), b = 2, p_i = 1/C_{max}$	$O(n^{2.5})$
$B1/GSK3 = (V, E), b/C_{max}$	$O(n^{2.5})$
$B1/G = (V, E), b = 2, r_i, p_i = 1/C_{max}$	NP-hard
$B1/G = (V, E), b \geq n, p_i = 1/C_{max}$	
for general graphs	NP-hard
for circular-arc graphs	$O(n^3)$
for chordal graphs	$O(n^2)$
for comparability graphs	$O(n^3)$
$B1/G(K_b^1, \ldots, K_b^s), b, r_i, p_i = 1/C_{max}$	NP-hard

Table 3. Summary of results

References

[1] J. Blazewicz, J. Barcelo, W. Brucker, and H. Rőck. Scheduling tasks on two processors with deadlines and additional resources. *European Journal of Operational Research*, 26:364-370, 1986.

[2] J. Blazewicz, W. Cellary, R. Slowinsky, and J. Weglarz. Scheduling under resource constraints - deterministic models. *Annals of Operations Research*, 7, 1986.

[3] M. Boudhar and G. Finke. Etude de quelques problèmes d'ordonnancement sur des machines à traitement par batchs avec des contraintes d'incompatibilité entre les tâches. (problèmes difficiles). *Rapport de recherche 003/99 (Institut de Mathématiques, USTHB. Alger)*, 1999.

[4] M. Boudhar and G. Finke. Etude de quelques problèmes d'ordonnancement sur des machines à traitement par batchs avec des contraintes d'incompatibilité entre les tâches. (problèmes polynomiaux). *Rapport de recherche 004/99 (Institut de Mathématiques, USTHB, Alger)*, 1999.

[5] M. Boudhar and G. Finke. Scheduling on a batch processing machine with capacity equal to two. In *Fourteenth Conference on Quantitative Methods for Decision Making (ORBEL'14)*. Mons (Belgium), 20-21 January 2000.

[6] M. Boudhar and G. Finke. Scheduling on batch processing machines with constraints of compatibility between jobs. In *Second Conference on Management and Control of Production and Logistics (MCPL'2000)*, Grenoble (France), 5-8 July 2000.

[7] N. Brauner, C. Dhaenens-Flipo, M-L. Espinouse, G. Finke, and H. Gavranovic. Decomposition into parallel work phases with application to the sheet metal industry. In *International Conference on Industrial Engineering and Production Management (IEPM'99)*, Glasgow, 12-15 July 1999.

[8] P. Brucker, A. Gladky, H. Hoogeveen, M.Y. Kovalyov, C. Potts, T. Tautenhahn, and S. Van De Velde. Scheduling a batching machine. *Journal of Scheduling*, 1:31–54, 1998.

[9] P. Brucker and S. Knust. Complexity results of scheduling problems. *page web: http://www.mathematik.uni-osnabrueck.de/research/OR/class/.*

[10] V. Chandru, C.Y. Lee, and R. Uzsoy. Minimizing total completion time on batch processing machines. *International Journal of Production Research*, 31:2097–2121, 1993.

[11] G. Dobson and R.S. Nambinadom. The batch loading and scheduling problem. *Research Report, Simon School of Business Administration, University of Rochester NY*, 1992.

[12] 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.

[13] F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques and maximum independent set of chordal graph. *SIAM Journal of Computing*, 1:180–187, 1972.

[14] F. Gavril. Algorithms on circular-arc graphs. *Networks*, 4:357–369, 1974.

[15] F. Jolai Ghazvini. *Ordonnancement sous contrainte de groupage (machine à traitement par batch).* Thèse de Doctorat, E.N.S.G.I.(I.N.P.G.), Grenoble (France), 1998.

[16] M.C. Golumbic. The complexity of comparability graph recognition and coloring. *Computing*, 18:199–208, 1977.

[17] D.S. Hochbaum and D. Landy. Algorithms and heuristics for scheduling semiconductor burn-in operations. *Research Report ESRC 94-8, University of California, Berkeley, USA*, 1994.

[18] W. Kubiak and F. Jolai Ghazvini. Minimizing earliness/tardiness criteria on a batch processing machine with job families. In *Second Annual International Conference on Industrial Engineering*, volume 2, pages 785–790, San Diego, California, USA, 22-15 November 1997.

[19] C.Y. Lee and R. Uzsoy. Minimizing makespan on a single batch processing machine with dynamic job arrivals. *International Journal of Production Research*, 17:219–236, 1999.

[20] S.V. Mehta and R. Uzsoy. Minimizing total tardiness on a batch processing machine with incompatible job families. *I.I.E. Transaction on Scheduling and Logistics*, 31:165–178, 1998.

[21] C.N. Potts and Y.K. Kovalyov. Scheduling with batching: A review. *European Journal of Operational Research*, 120:228–249. 2000.

[22] M. Sakarovitch. *Optimisation combinatoire: programmation discrète.* Hermann, 1984.

[23] R. Uzsoy. Scheduling batch processing machines with incompatible job families. *International Journal of Production Research*, 33:2685–2708, 1995.