A new mixed integer linear programming model for minimizing makespan on a single batch processing machine with release times and non-identical job sizes

Renan Spencer Trindade
Programa de Pós-graduação em Informática - Universidade Federal de Santa Maria
Av. Roraima nº 1000 - Cidade Universitária - Bairro Camobi - Santa Maria - RS - 97105-900
renanspencer@gmail.com

Olinto César Bassi de Araújo
Colégio Técnico Industrial de Santa Maria - Universidade Federal de Santa Maria
Av. Roraima nº 1000 - Cidade Universitária - Bairro Camobi - Santa Maria - RS - 97105-900
olinto@ctism.ufsm.br

Felipe Martins Müller
Programa de Pós-graduação em Informática - Universidade Federal de Santa Maria
Av. Roraima nº 1000 - Cidade Universitária - Bairro Camobi - Santa Maria - RS - 97105-900
felipe@inf.ufsm.br

Marcia Helena Costa Fampa
Instituto de Matemática and COPPE - Universidade Federal do Rio de Janeiro
Cidade Universitária, CT, Bl. H, Sala 319-03, C.P. 68511, Rio de Janeiro - RJ - 21941-972
fampa@cos.ufrj.br

RESUMO

Problemas de programação de máquinas de processamento de bateladas com o objetivo de minimizar o tempo total de processamento são extensivamente estudados na literatura científica, motivados principalmente por testes de confiabilidade na indústria de semicondutores. Neste trabalho é considerado o problema de sequenciamento de bateladas em máquina única com tempos de liberação e com tarefas de tamanhos diferentes $(1|r_j, s_j, B|C_{max})$. O tamanho total de uma batelada não pode ultrapassar a capacidade da máquina e o tempo de processamento de uma batelada é igual ao tempo de processamento mais longo das tarefas alocadas na batelada. Um novo modelo matemático é proposto com uma formulação que inclui cortes de simetria. Resultados computacionais evidenciam que o novo modelo matemático claramente supera o modelo clássico da literatura.

PALAVRAS CHAVE. Programação inteira mista, dimensionamento e programação de lotes, máquina de processamento em lotes.
Área Principal: Otimização Combinatória, Programação Matemática, PO na indústria.

ABSTRACT

Scheduling problems on batch processing machines aiming to minimize makespan are widely exploited in the scientific literature, mainly motivated by reliability testing in the semiconductor industry. We consider the scheduling problem of minimizing the makespan in a single batch machine with release times and non-identical job sizes $(1|r_j, s_j, B|C_{max})$. The total size of a batch cannot exceed machine capacity and the processing time of a batch is equal to the largest job processing time assigned to the batch. A new mathematical model is proposed with a formulation which includes symmetry breaking cuts. The computational results show that the new mathematical model clearly outperforms the classical model from the literature.

KEYWORDS. Mixed integer linear program. Batch scheduling. Batch Processing machine.
Main Area: Combinatorial Optimization, Mathematical Programming, OR in Industry

1. Introduction

Batch processing machines can perform the same task in a group of products simultaneously, aiming to avoid settings in equipment and facilitate material handling. These machines are used in many different industry segments, such as, shoe manufacturing (Fanti et al., 1996), aircraft (Zee, Van Harten and Schuur, 1997), metal (Ram and Patel, 1998; Mathirajan, Chandru and Sivakumar, 2004), and furniture manufacturing (Yuan et al. 2004). As there are many types of the problem associated with batch processing machinery, this study is motivated by reliability tests in the semiconductor industry, specifically by operations called burn-in, presented in Uzsoy (1994).

During burn-in operation, integrated circuits (IC) are submitted to a thermal stress for a long period of time and then submitted to integrity tests that may identify latent defects leading to their discard. This is done by maintaining the circuit at a constant temperature, usually in an industrial oven at about 120°C. Another similar procedure, described in Chung Tai and Pearn (2009), is to use batch processing machines in thermal aging tests for evaluating thin film transistor liquid crystal displays (TFT-LCD). In this case, the industrial oven is generally set at a temperature of 55°C and the TFT-LCD panels tested have different assembly times, not all being available at the moment the test procedure starts. As the assembly times are known, the release times of the products can be defined a priori.

Each IC or panel to be tested is called a job and requires a minimum time inside the oven (processing machine). This minimum time is also settled a priori, considering the supplier requirements. Jobs cannot be processed directly on the machine, they need to be placed on a tray, called batch, which contains a limited number of jobs. A job may stay inside the oven longer than the minimum required time, but it cannot stay a shorter period of time. This feature allows a machine to process jobs with different processing times. The processing time of the batch is then equal to the longest processing time among all jobs assigned to the batch. In case of jobs with non-identical release times, the batch can only start to be processed when all the tasks assigned to it are available. This is due to the fact that once the batch processing starts, it cannot be interrupted until the test is completed.

Processing times of burn-in operations can be quite long when compared to other test operations. For example, it is possible to have 120 h of burn-in operation, against 4-5 h for other test operations. The process may require three months to produce a chip. TFT-LCD panels tests usually take 6 hours and a machine can handle about 450 parts of panels. Therefore, burn-in tests consist of a bottleneck in the final test operation and efficient scheduling of these operations aims to maximize productivity and reduce flow time in the stock, which constitutes a great concern for management.

Several works address problems of designing and scheduling batch processes in a single machine with non-identical job sizes. Heuristic methods have been applied when release times are not considered, as in Uzsoy (1994) and Ghazvini and Dupont (1998). A branch-and-bound method is proposed in Dupont and Dhaenens-Flipo (2002). A mixed integer linear programming formulation is presented in Melouk, Damodaran and Chang (2004). The application of branch-and-price method is proposed by Parsa, Kashan and Karimi (2010). Metaheuristics are also proposed, such as, simulated annealing in Melouk, Damodaran and Chang (2004), a genetic algorithm in Kashan, Karimi, and Jolai (2006), Tabu Search in Meng and Tang (2010) and GRASP in Damodaran, and Ghrayeb Guttikonda (2013).

This paper addresses the minimizing makespan on a single batch processing machine with release times and non-identical job sizes. The problem is categorized as $1|r_j, s_j, B|C_{max}$, on the three-field system proposed by Graham, R. L, et al. (1979). From our best knowledge, there are only two papers in the literature that propose the application of metaheuristics to the solution of this problem, namely: hybrid genetic algorithms in Chou, Chang and Wang (2005) and ant colony in Xu, Chen and Li (2012). Also, a mathematical programming formulation for the problem is presented in Xu and Li Chen (2012). In Xu, Chen and Li (2012) the problem is proven NP-hard, based on

the fact that problem $1|r_j, s_j, B|C_{max}$ is equivalent to problem $1|s_j, B|C_{max}$ in the particular case where all jobs have release time equal to zero $r_j = 0, \forall j \in J$, and this last problem was proven NP-hard in Uzsoy (1994).

This work proposes a new mixed integer linear programming (MILP) formulation for the $1|r_j, s_j, B|C_{max}$ problem which includes symmetry breaking cuts. A similar approach is found in Köhler, Fampa and Araújo (2012) applied to sofware clustering problem. Symmetry breaking cuts are proposed for other specific problems such as synchronous optical network design problem in Sherali and Smith (2001), layout problem in the fashion industry in Degraeve, Gochet and Jans (2002), packing and partitioning orbitopes in Kaibel and Pfetsch (2007) and lot-sizing problems in Jans (2009).

The problem addressed in this work is solved by CPLEX, considering two MILP formulations: the one presented here and the other presented in Xu, Chen and Li (2012). Comparisons and numerical results are presented.

The paper is divided into five sections. Section 2 presents the MILP formulation for the problem proposed in Xu and Li Chen (2012). In Section 3, we propose our new MILP formulation for the problem, which includes symmetry breaking cuts. In Section 4, we discuss our experiments and present numerical results. In Section 5 we present our final remarks.

2. MILP formulation for $1|r_j, s_j, B|C_{max}$ on the literature review

This problem can be formally defined as: given a set J of jobs, each element $j \in J$ has a release time r_j, a processing time p_j and a size s_j. Each job $j \in J$ must be assigned to a batch $k \in K$, not exceeding the capacity limit B of the machine. The batches must be scheduled on the processing machine. The processing time of each batch $k \in K$ is defined as $P_k = \max\{p_j : j \text{ is assigned to } k\}$. Each job $j \in J$ becomes available at time r_j, which means that jobs with different release times are considered in this problem. The release time of batch $k \in K$ is defined as $R_k = \max\{r_j : j \text{ is assigned to } k\}$. Jobs cannot be divided between batches. It's also not possible to add or remove jobs from the machine during the batch processing. The goal is to design and schedule the batches $k \in K$ so that the processing time of the machine C_{max} (makespan) is minimized. The number of batches used depends on the number of jobs and the considered processing machine capacity. In the worst case, the number of batches is equal to the number of jobs, $|K| = |J|$.

The following MILP formulation for the problem is proposed in Xu, Chen and Li (2012).

Parameters

s_j : size of job j
p_j : processing time of job j
r_j : release time of job j
B : capacity of the machine

Decision variables

$$x_{j,k} = \begin{cases} 1 & \text{if job } j \text{ is assigned to batch } k\ ; \\ 0 & \text{otherwise.} \end{cases}$$

P_k : processing time of batch k
S_k : starting time of batch k

Model 1

$$Min\ S_{|K|} + P_{|K|} \tag{1}$$

$st.$

$$\sum_{k \in K} x_{j,k} = 1 \qquad \forall j \in J \tag{2}$$

$$\sum_{j \in J} s_j x_{j,k} \leq B \qquad \forall k \in K \tag{3}$$

$$P_k \geq p_j x_{j,k} \qquad \forall j \in J, \forall k \in K \tag{4}$$

$$S_k \geq r_j x_{j,k} \qquad \forall j \in J, \forall k \in K \tag{5}$$

$$S_k \geq S_{k-1} + P_{k-1} \qquad \forall k \in K : k > 1 \tag{6}$$

$$P_k \geq 0 \qquad \forall k \in K \tag{7}$$

$$S_k \geq 0 \qquad \forall k \in K \tag{8}$$

$$x_{j,k} \in \{0, 1\} \qquad \forall j \in J, \forall k \in K \tag{9}$$

The objective function (1) minimizes the total processing time (makespan). In the model, the makespan is defined as the time required to complete the last batch. Constraint (2) states that each job is assigned to only one batch. Constraint (3) states that each batch does not exceed the machine capacity. Constraint (4) determines the processing time of batch k. Constraints (5) and (6) determine the time when the batch k starts to be processed. Constraints (7), (8) and (9) define the domains of the variables.

Model 1 has a large number of symmetric solutions, which particularly occur in two situations, presenting batches with identical configurations and same value for C_{max}.

The first situation occurs when the number of batches used in a solution is less or equal to $|K|$. In this case, the model allows the same solution having its batches represented by different indexes. More specifically, a batch may be represented by any index k', such that $k'' < k' < k'''$, where k'' is the index of the immediately preceding batch and k''' is the index of the batch immediately after k'. Note that the order in which the batches are sequenced in the processing machine is always the same. Figure 1 illustrates this situation showing an example with three symmetric solutions, where each solution is composed by batches "A", "B" and "C" sequenced in the machine always in the same order. Model 1 identifies these solutions as three different solutions, in which the batch "B" (doted lines) can be represented by the indexes k_2, k_3 or k_4. The makespan of the three solutions has the same value.

The second situation occurs when a modification in the order in which the batches are processed does not change C_{max}. This can happen when a whole group of adjacent batches are released at the time when the machine starts to process, i.e., the machine is not idle when these batches start to be processed.

Figure 2 shows two examples where the permutation of the batches order does not affect the value of C_{max}. In Case 1 the two batches k and k' have the same release times, and exchanging their orders clearly does not affect C_{max}. In Case 2 the batch k' is released before batch k. Furthermore, the machine is not idle at the moment these two batches are released because of the processing of the batch k''. In this configuration, the exchange in the orders of the batches k and k' also does not affect the value of the makespan C_{max}.

3. A new MILP formulation for $1|r_j, s_j, B|C_{max}$

The idea underlying the new mathematical formulation for $1|r_j, s_j, B|C_{max}$ is to take the jobs sorted by non-decreasing order of the release time, i.e., set $r_1 \leq r_2 ... \leq r_{|J|}$. The symmetries exemplified in the previous section can be handled by defining the decision variables as

$$x_{j,k} = \begin{cases} 1 & \text{if job } j \text{ is assigned to batch } k\ ; \\ 0 & \text{otherwise.} \end{cases} \qquad ,\forall k \in K, \forall j \in J : j \leq k \tag{10}$$

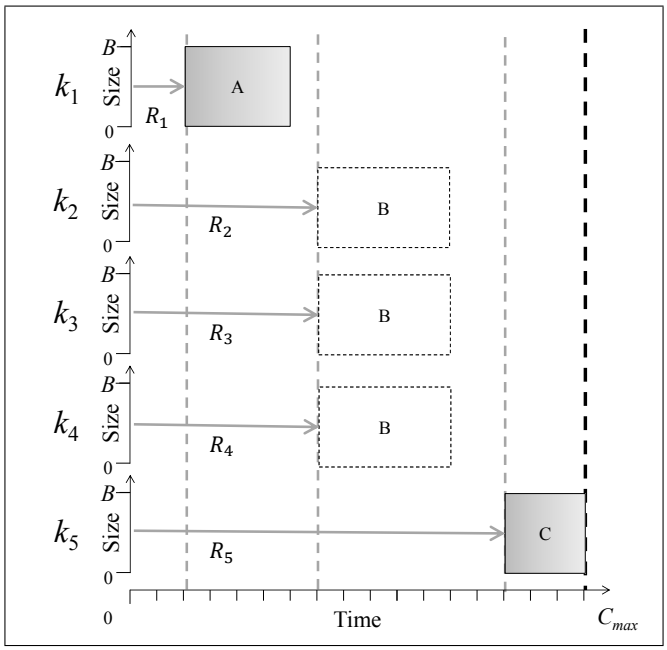

Figure 1: Symmetric solutions where batches are represented by different indexes.

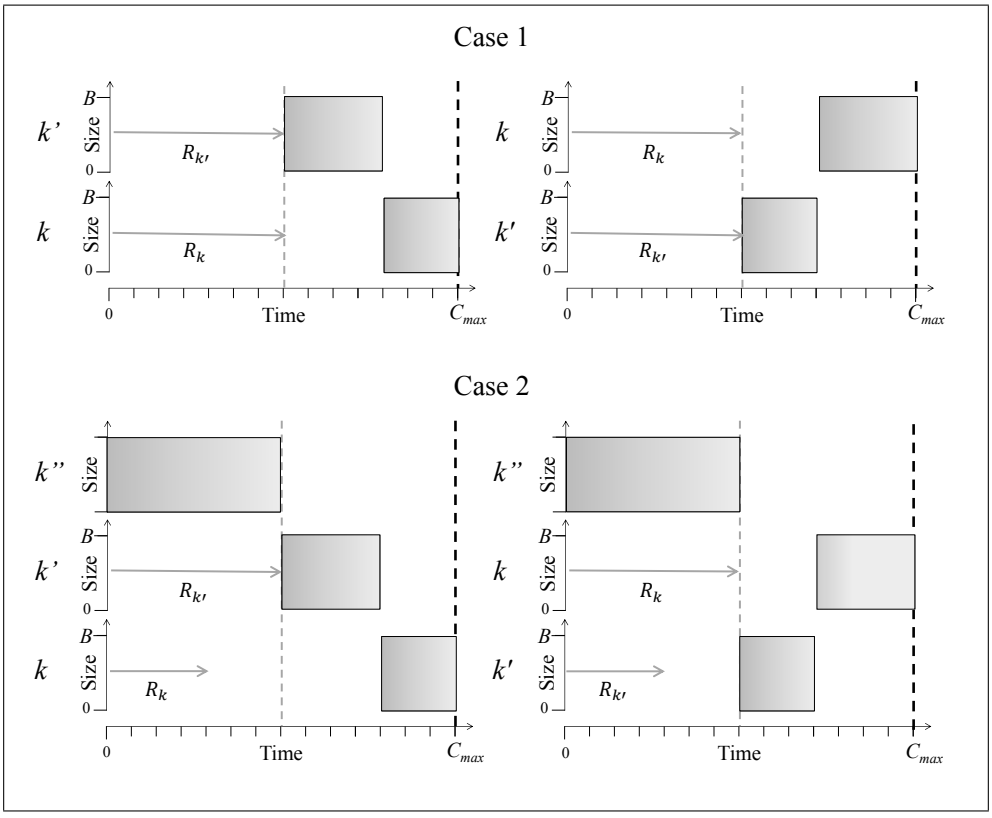

Figure 2: Cases where exchanging the order of batches does not change the makespan.

From this definition, it follows that a job can only be assigned to a batch if the index of the job is less than or equal to the index of the batch. If in addition, it is enforced that batch k can only be used if job k is assigned to it, then any solution to problem $1|r_j, s_j, B|C_{max}$ will be represented by sequential batches in the same order as the jobs.

Proposition 1 ensures that no optimal solution is cut off when the problem is conceived in this way.

Proposition 1. *Any optimal solution for* $1|r_j, s_j, B|C_{max}$ *can be represented considering batches sequenced in non-decreasing order of their release times.*

Proof. Let S represent an optimal schedule for $1|r_j, s_j, B|C_{max}$ in which there are at least two batches k and k', such that the batch k is sequenced after the batch k' and the respective release times are such that $R_k < R_{k'}$.

If all the batches in S are processed without inserted idle time (see Case 2 in Figure 2), then it is possible to construct a new schedule S' in which batch k precedes batch k', with same makespan of S.

On the other hand, if the schedule S contains an inserted idle time immediately before batch k', the exchange of the batches k and k' will decrease the makespan value, and in this case the schedule S is not optimal. □

Considering the parameters and the variables P_k and S_k from Model 1, together with the variables x defined in (10), the proposed new mathematical model is given by:

Model 2

$$Min\ S_{|K|} + P_{|K|} \tag{11}$$

$st.$

$$\sum_{k \in K: j \leq k} x_{j,k} = 1 \qquad \forall j \in J \tag{12}$$

$$\sum_{j \in J: j \leq k} s_j x_{j,k} \leq B x_{k,k} \qquad \forall k \in K \tag{13}$$

$$x_{j,k} \leq x_{k,k} \qquad \forall j \in J, \forall k \in K : j < k \tag{14}$$

$$P_k \geq p_j x_{j,k} \qquad \forall j \in J, \forall k \in K : j \leq k \tag{15}$$

$$S_k \geq r_k x_{k,k} \qquad \forall k \in K \tag{16}$$

$$S_k \geq S_{k-1} + P_{k-1} \qquad \forall k \in K : k > 1 \tag{17}$$

$$P_k \geq 0 \qquad \forall k \in K \tag{18}$$

$$S_k \geq 0 \qquad \forall k \in K \tag{19}$$

$$x_{j,k} \in \{0, 1\} \qquad \forall j \in J, \forall k \in K : j \leq k \tag{20}$$

The objective function (11) minimizes the total processing time (makespan). As in Model 1, the makespan is defined as the time required to complete the last batch. Constraint (12) is a valid inequality which determines that each job is assigned to only one batch, with non smaller index than the job index. Constraint (13) states that each batch does not exceed the machine capacity. Disaggregated constraint (14) imposes that each job can only be assigned to an open batch. Constraint (15) determines the processing time of batch k. Constraints (16) and (17) determine the time when the batch k starts to be processed. Constraint (16), in particular, is a symmetry cut that avoids the use of a batch that does not contain the job of the same index. Constraints (18), (19) and (20) define the domains of the variables.

4. Computational experiments

This section will present how the tests were performed, how the instances were generated and the results obtained comparing both models discussed above, say, Model 1 and Model 2.

4.1. Instances

The set of instances were generated according to the methodology proposed by Chou, Chang and Wang (2005) and presented by Xu, Chen and Li (2012), differing only in the way to fix the upper bound on tthe release time. For each job j, an integer processing time p_j, an integer release time r_j and an integer job size s_j were generated from the respective uniform distribution depicted in Table 1. In total were generated 280 instances, 20 for each of the 14 different combinations of number and size of the jobs. We also address results for instances with more than 100 jobs. This appears for the first time in the literature for $1|r_j, s_j, B|C_{max}$, using either exact or heuristic methods.

The following steps were considered to generate the release times of the tasks for each instance:
1. Set size and processing time of the jobs;
2. Find an initial solution using LPT-BFF heuristic setting all releasing times to zero. In the Batch First Fit Lowest Processing Time heuristic, the tasks are sorted in a non-decreasing order of their processing times and each task is allocated to the first batch it fits, considering the maximum capacity of the batch;
3. Set C equal to C_{max} obtained by LPT-BFF heuristic;
4. Then the release times are uniformly random generated in the interval [0, C].

Table 1: Parameter settings.

Number of jobs:	10, 20, 50, 100, 200, 300, 500
Release time:	r_1: [0, C]
Processing time:	p_1: [8, 48]
Jobs size:	s_1: [1, 15] s_2: [15, 35]
Machine capacity:	40

4.2. Comparisons of Model 1 and Model 2

We run the computational experiments using CPLEX 12.5 with default parameter settings on an Intel Quad-Core Xeon X3360 2.83 GHz processor, 8GB of RAM. The time limit to run each instance was fixed in 1800 seconds.

Each line of Table 2 presents the average results for 20 instances of each possible combination. The first two columns present the identification of the problem. The following columns present the average makespan C_{max}, the average of total execution time $T(s)$, the average of time to best solution found $T_B(s)$ and the average of CPLEX duality gap, for Model 1 and Model 2, respectively.

Both models were efficient to solve reduced size instances. Optimal solutions were found for all instances with 10 jobs in at most 0.20 seconds, $T_B(s)$ columns. However, looking the whole set of results, Model 2 overcomes Model 1, both in execution time and in solution quality. Observe that Model 1 could not prove optimality for all instances with more than 20 jobs. On the other hand, using Model 2 optimal solutions were found for all instances with 10 and 20 jobs in at most 0.32 seconds, $T_B(s)$ columns. Considering 50 and 100 jobs Model 2 continues to overcomes Model 2 in both criteria.

Table 2: Computational results for Models 1 and Model 2.

		Model 1				Model 2			
Number of jobs	s_j	C_{max}	$T(s)$	$T_B(s)$	Gap	C_{max}	$T(s)$	$T_B(s)$	Gap
10	[1, 15]	117.80	0.16	0.09	0.00	117.80	0.02	0.02	0.00
10	[15, 35]	316.95	0.57	0.20	0.00	316.95	0.01	0.00	0.00
20	[1, 15]	193.80	192.34	27.78	0.03	193.80	0.33	0.32	0.00
20	[15, 35]	560.70	270.13	7.97	0.44	560.70	0.01	0.01	0.00
50	[1, 15]	396.50	1800.00	1674.20	40.64	389.45	456.99	129.53	0.63
50	[15, 35]	1351.80	1800.00	1725.32	75.01	1298.55	0.06	0.06	0.00
100	[1, 15]	920.05	1800.00	1744.30	94.50	760.45	1744.24	327.94	5.51
100	[15, 35]	3368.70	1800.00	1640.57	94.90	2578.35	0.39	0.36	0.00
200	[1, 15]	2884.10	1800.00	177.67	98.33	1576.75	1800.00	1761.36	15.38
200	[15, 35]	9257.15	1800.00	82.61	99.40	5049.35	1.19	1.17	0.00
300	[1, 15]	4355.70	1800.00	1785.86	99.94	2526.05	1800.00	1388.04	21.49
300	[15, 35]	13707.75	1800.00	1754.98	99.86	7483.25	1.99	1.93	0.00
500	[1, 15]	7152.60	1800.00	1800.00	100.00	5805.50	1800.00	1179.36	43.76
500	[15, 35]	23320.85	1800.00	1800.00	100.00	12589.80	3.50	3.43	0.00

Table 2 also shows that instances of type "s_1" demand more computational effort for both models. This can be explained by the reduced size of the tasks compared with the capacity of the machine that allows more possible batch configurations increasing the solution space to be explored. Considering this fact we can conclude that "s_1" instances are more difficult than "s_2".

Model 2 found optimal solution of all instances "s_2" with $T(s)$ lower than 3.5 seconds. For instances "s_1" with more than 50 jobs Model 2 cannot find optimal solutions for the whole set, showing the worst performance for 500 jobs where the average gap was 43.76%.

Observing the average of total execution time $T(s)$ and the average of time to best solution found $T_B(s)$ for all instances "s_2" with more than 200 jobs, we see that most of the computational effort is spent to improve the objective function value and not proving optimality, since the best solution reported is found in a time closer to the upper bound of 1800 seconds.

5. Conclusions

In this paper, we propose a new mixed integer linear programming (MILP) formulation for the scheduling problem of minimizing the makespan on a single batch machine with release times and non-identical job.

To our knowledge the only mathematical programming model for this problem presented in the literature is also a MILP formulation, which contains in its feasible set a large number of symmetric solutions. Due to the symmetry, the branch-and-bound algorithm when applied to this MILP, behaves very inefficiently, solving equivalent subproblems many times, and, therefore, the solution of even moderate-sized instances may become very challenging.

Motivated by this realization, the MILP formulation that we propose in this work incorporates symmetry breaking cuts. The idea behind the symmetry cuts is based in rules that should be satisfied by the indexes assigned to different batches, and considers the jobs sorted by non-decreasing order of the release time. The formulation presented was proven to be correct, i.e., we proved that any optimal solution of the problem belongs to its feasible set.

Finally, we have shown with our numerical experiments, the strength of these symmetry cuts, when comparing the results obtained with the new proposed formulation to the previous one. For the first time in the literature, proven optimal solutions were presented for instances with more than 100 jobs. In the time limit of 30 minutes we could prove optimality for instances of up to 500 jobs, while only instances with up to 20 jobs were solved to optimality by using the previous MILP formulation.

To our knowledge the results presented in this work constitute the state of the art for exact solution algorithms for this NP-Hard problem, and as future work, we can mention the study of pre-processing procedures, for example to reduce the possible number of batches in the solution.

References

Chou, F.-D., Chang, P.-C., and Wang, H.-M. (2005), A hybrid genetic algorithm to minimize makespan for the single batch machine dynamic scheduling problem, *The International Journal of Advanced Manufacturing Technology*, 31(3-4), 350-359.

Chung, S. H., Tai, Y. T., and Pearn, W. L. (2009), Minimising makespan on parallel batch processing machines with non-identical ready time and arbitrary job sizes, *International Journal of Production Research*, 47(18), 5109-5128.

Damodaran, P., Ghrayeb, O., and Guttikonda, M. C. (2013), GRASP to minimize makespan for a capacitated batch-processing machine, *The International Journal of Advanced Manufacturing Technology*, 68(1-4), 407-414.

Degraeve, Z., Gochet, W., and Jans, R. (2002), Alternative formulations for a layout problem in the fashion industry, *European Journal of Operational Research*, 68(1-4), 143(1), 80-93.

Dupont, L., and Dhaenens-Flipo, C. (2002), Minimizing the makespan on a batch machine with non-identical job sizes: an exact procedure, *Computers & Operations Research*, 29(7), 807-819.

Fanti, M. P., Maione, B., Piscitelli, G., and Turchiano, B. (1996), Heuristic scheduling of jobs on a multi-product batch processing machine, *Int J Prod Res*, 34(8), 2163-2186.

Ghazvini, F. J. and Dupont, L. (1998), Minimizing mean flow times criteria on a single batch processing machine with non-identical jobs sizes, *International Journal of Production Economics*, 55, 273-280.

Graham, R. L., Lawler, E. L., Lenstra, J. K. and Rinnooy Kan, A. H. G. (1979), Optimization and approximation in deterministic sequencing and scheduling?: a survey, *Proceedings of the Advanced Research Institute on Discrete Optimization and Systems Applications of the Systems Science Panel of NATO and of the Discrete Optimization Symposium co-sponsored by IBM Canada and SIAM Banff*, Aha. and Vancouver, 5, 287-326.

Jans, R. (2009), Solving Lot-Sizing Problems on Parallel Identical Machines Using Symmetry-Breaking Constraints, *INFORMS Journal on Computing*, 21(1), 123-136.

Kaibel, V., and Pfetsch, M. (2007), Packing and partitioning orbitopes, *Mathematical Programming*, 114(1), 1-36.

Kashan, A. H., Karimi, B., and Jolai, F. (2006), Effective hybrid genetic algorithm for minimizing makespan on a single-batch-processing machine with non-identical job sizes, *International Journal of Production Research*, 44(12), 2337-2360.

Köhler, V., Fampa, M., and Araújo, O. (2012), Mixed-Integer Linear Programming Formulations for the Software Clustering Problem, *Computational Optimization and Applications*, 55(1), 113-135.

Mathirajan, M., Sivakumar, A.I., and Chandru, V. (2004), Scheduling algorithms for heterogeneous batch processors with incompatible job families, *J Intell Manuf*, 15, 787-803.

Melouk, S., Damodaran, P., and Chang, P.-Y (2004), Minimizing makespan for single machine batch processing with non-identical job sizes using simulated annealing, *International Journal of Production Economics*, 87(2), 141-147.

Meng, Y., and Tang, L. (2010), A tabu search heuristic to solve the scheduling problem for a batch-processing machine with non-identical job sizes, *In 2010 International Conference on Logistics Systems and Intelligent Management (ICLSIM)*, (Vol. 3, pp. 1703-1707). IEEE.

Parsa, N. R., Karimi, B. and Kashan, H. A. (2010), A branch and price algorithm to minimize makespan on a single batch processing machine with non-identical job sizes, *Computers & Operations Research*, 37(10), 1720-1730.

Ram, B. and Patel, G. (1998), Modeling furnace operations using simulation and heuristics, *Proc 1998 winter simulation conference*, 957-963.
Sherali, H. D., and Smith, J. C. (2001), Improving Discrete Model Representations via Symmetry Considerations, *Management Science Publication*, 47(10), 1396-1407
Uzsoy, R. (1994), Scheduling a single batch processing machine with non-identical job sizes, *International Journal of Production Research*, 32(7), 1615-1635.
Xu, R., Chen, H., and Li, X. (2012), Makespan minimization on single batch-processing machine via ant colony optimization, *Computers & Operations Research*, 39(3), 582-593.
Yuan, J. J., Liu, Z. H., Ng, C.T., and Cheng, T.C.E. (2004), The unbounded single machine parallel batch scheduling problem with family jobs and release dates to minimize makespan, *Theor Comput Sci*, 320(2-3), 199-213.
Zee, D. J. Van Der., Van Harten, A., and Schuur P. C. (1997), Dynamic job assignment heuristics for multi-server batch operations - A cost based approach, *Int J Prod Res*, 35(11), 3063-3093.