The Ninth International Symposium on Operations Research and Its Applications (ISORA'10)
Chengdu-Jiuzhaigou, China, August 19–23, 2010

A Row-and-Column Generation Method to a Batch Machine Scheduling Problem

Gongshu Wang Lixin Tang

Liaoning Key Laboratory of Manufacturing System and Logistics,
The Logistics Institute, Northeastern University, Shenyang, China, 110819

Abstract This paper studies a batch machine scheduling problem in which decisions on grouping jobs into batches and scheduling batches on parallel machines are jointly made. To address the problem, we model it as two-stage set-partitioning (**TSSP**) type formulation with exponential number of rows and columns. To find solutions, we propose a row-and-column generation method. The proposed method starts with a restricted version of the linear relaxation of (**TSSP**), and enlarges the model through iteratively generating needed rows and columns. The integer solution is obtained by solving the final integer programming with generated rows and columns. Experimental results show the efficiency of the proposed method.

Keywords Batching; Scheduling; Large-scale optimization; Row-and-column generation

1 Introduction

A batch machine is a machine that can handle a group of jobs simultaneously. The study of batch machine scheduling problems is significant from both theoretical and practical points of view. From a practical point of view, it is important because we can find many examples of using batch machine in the real world. The motivation for this work comes from a real steel production of reheating ingots on soaking pit where a batch of ingots with the same and similar reheating manner is simultaneously reheated. From a theoretical point of view, the batch machine scheduling problem is a generalization of the discrete machine scheduling problem, and some interesting results of the former can be extended to the latter.

Batch machine scheduling problems have received considerable attention in the literature. The problem of batching and scheduling of certain kinds of batch processors is discussed in [1], where an integer programming formulation for this problem is presented, a lower bound from a partial LP relaxation is generated, a polynomial algorithm to solve a special case is provided, and a set of heuristics are tested on the general problem. New complexity results on the parallel batch scheduling problem subject to release dates are presented in [2-4]. An unbounded model of the single machine parallel batch scheduling problem with release dates is considered in [5]. For literature review on the algorithms and complexity results for batch machine scheduling problems, see [6-8].

Correspondence to: L.X. Tang (tanglixin@ise.neu.edu.cn), G.S. Wang (wanggongshu@ise.neu.edu.cn)

The problem considered in this paper is abstracted from the actual steel production, and it distinguishes previous batch machine scheduling problems in several aspects. First, jobs can be grouped together to form a batch only if the dissimilarity of jobs' attributes is within a given tolerance. Second, the concept of core job is introduced in this study, and the processing manner of a batch is that of the core job. Finally, the problem looks at a trade-off between total weighted completion time and total cost of dissimilarity of jobs' attributes in each batch.

Solving scheduling problems on discrete machine by the column generation method has received considerable attention [9-13] because of the excellent lower bound obtained from the linear relaxation of set-partitioning formulation. However, our problem can not be modeled as the general set-partitioning type formulation because the problem decisions require that jobs need to be partitioned into batches and meanwhile batches need to be partitioned into single-machine schedules. As a consequence, the standard column generation method can not solve our problem directly. To overcome the mentioned difficulties, we model our problem as two-stage set-partitioning type formulation, and extend the standard column generation to a generalization form called row-and-column generation to solve it.

This paper is organized as follows. In Section 2, we describe the problem and model it as two-stage set-partitioning type model. A row-and-column method is developed in Section 3 to solve the model. Computational experiments and results are reported in Section 4. Finally conclusion is given in Section 5.

2 Problem Description And Mathematical Formulation

2.1 Problem Description

There are n independent jobs that have to be scheduled on m identical parallel batch machines. Each job $j \in \{1,\ldots, n\}$ is associated with a processing time p_j, a weight w_j to identify its importance and urgency and a volume v_i to represent its physical size. Without loss of generality, we assume that the job parameters p_j, w_j and v_j are integral. Each batch machine is available from time zero onwards and can process up to a batch of jobs simultaneously such that total volumes of jobs in the batch do not exceed the given capacity of the machine V. Also, each job j has an attribute to indicate it is preferred to be processed at what manner (for example, each ingot is specified with a preferred re-heating curve). Through each job is specified with a preferred process manner, it can also be processed at other similar manner to increase the flexibility of grouping jobs into a batch. Once a batch is formed, there is a core job whose attribute all jobs in this batch will be produced according to. For a pair of jobs (i, j), $i \neq j \in \{1,\ldots, n\}$, it is associated with a dissimilarity cost q_{ij}. To guarantee the product quality, each job j is associated with a subset of assignable jobs $i \in L_j$ such that the dissimilarity cost between each job belonging to L_j and the core job j is not large than a given tolerance. The problem is to group jobs into a set of batches and sequence those batches on a set of machines to minimize total dissimilarity cost and total weighted completion time.

Our problem is a generalization of the discrete parallel machine scheduling problem to minimize the total weighted completion time in which the volumes of all

jobs are identical and a machine can handle up a job every time. And as it is known that the latter is NP-hard [9-10], our problem is also NP-hard.

2.2 Two Stage Set-partitioning Type Formulation

According to problem description, a batch is defined by a core job j and a set of other non-core jobs belonging to L_j, such that total volumes of jobs do not exceed the machine capacity V. We define B^j as the set of all possible batches corresponding to core job j, and $B=B^1 \cup B^2 \cup \cdots \cup B^n$. Associated with each batch $b \in B$ is an incidence vector $\boldsymbol{a}_b$ whose ith component a_{ib} equals to 1 if job i is included in batch b, and 0 otherwise. Given a batch $b \in B_j$, the processing time of the batch is $p(b)=p_j$, the weight of the batch is $w(b)=\sum_{i \in L_j} a_{ib} w_i + w_j$ and the dissimilarity cost for the batch is $q(b)=\sum_{i \in L_j} a_{ib} q_{ij}$.

A single-machine schedule s is defined by a sequence of batches $(b_1,\cdots,b_r)$. The total weighted completion time of a single-machine schedule s can be calculated as $c(s)=\sum_{l=1}^{r}\sum_{k=1}^{l} w(b_l)p(b_k)$. We define S as the set of all possible single-machine schedules. Associated with each $s \in S$ is an incidence vector $\boldsymbol{A}_s$ whose bth component A_{bs} equals to 1 if schedule s covers the batch b, and 0 otherwise.

We introduce binary decision variable λ_b to determine whether a batch $b \in B$ is selected or not, binary decision variable μ_s to determine whether a schedule $s \in S$ is selected or not. Let ω_1, $\omega_2 \in [0,1]$ be the weights of two terms of a scalar objective such that $\omega_1+\omega_2=1$. With these notations, the problem can be modeled as the following two-stage set-partitioning type model, where jobs are partitioned into batches at the first stage and all active batches are partitioned into single-machine schedules at the second stage.

[**TSSP**]

$$\min \quad \omega_1 \sum_{b \in B} q(b)\lambda_b + \omega_2 \sum_{s \in S} c(s)\mu_s \tag{1}$$

subject to

$$\sum_{b \in B} a_{ib}\lambda_b = 1, \quad \forall i \in \{1,...,n\}, \tag{2}$$

$$-\lambda_b + \sum_{s \in S} A_{bs}\mu_s = 0, \quad \forall b \in B, \tag{3}$$

$$\sum_{s \in S} \mu_s \le m, \tag{4}$$

$$\lambda_b \in \{0,1\}, \quad \forall b \in B, \tag{5}$$

$$\mu_s \in \{0,1\}, \quad \forall s \in S. \tag{6}$$

The objective (1) is to minimize the total dissimilarity cost of active batches and total weighted completion time of the selected single-machine schedules. Constraints (2) ensure that each job is exactly covered by one active batch. Constraints (3) impose that if a batch is active, it must be contained in a single-machine schedule. Constraint (4) guarantees that no more than m schedules can be selected for there are m machines available. Constraints (5)-(6) define the value range of the variables.

3 Solution Method

3.1 The Framework of Row-and-Column Generation

Since the number of all possible batches, $|B|$, and the number of all possible single-machine schedules, $|S|$, are both exponentially large, the number of rows, $|B|+n+1$, and the number of columns, $|B|+|S|$, are also exponentially large, and thus it is impractical to solve (**TSSP**) directly. The standard column generation method can not solve (**TSSP**) as well, because we do not know in advance which batches should be active in an optimal solution and uncertainty stretches in both directions: rows and columns. To overcome the mentioned difficulty, we modify and extend the standard column generation method to a row-and-column generation method to solve the formulation.

By relaxing binary variables λ_b and μ_s to continuous variables, the linear relaxation of (**TSSP**), denoted as (**LTSSP**), is obtained. The proposed method starts with a restricted master problem (**RMP**) which contains only a small subset of columns and rows of (**LTSSP**), and then enlarges (**RMP**) through iteratively generating needed rows and columns by dynamic programming. Because row-and-column generation deals with (**LTSSP**), it only obtains an estimated lower bound for (**TSSP**). To obtain feasible solution of (**TSSP**), an integer programming with generated rows and columns in (**RMP**) is solved by the integer optimization solver of CPLEX. The whole framework of the proposed method is shown in Fig. 1.

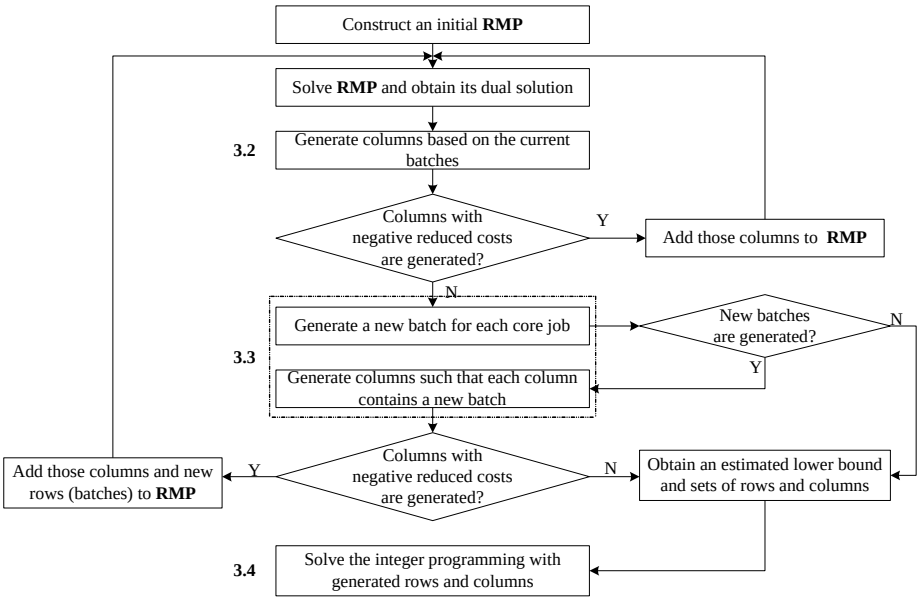

Fig 1. The flow chart of row-and-column generation

3.2 Generating Columns

Let $\boldsymbol{\rho}$ be vectors of dual variables corresponding to constraints (3) and θ be the dual variable corresponding to constraint (4). The reduced cost of a column μ_s, $s \in S$ is given by the following formula:

$$\sigma_s = \omega_2 c(s) - \sum_{b \in B} A_{bs} \rho_b - \theta \tag{18}$$

In the standard column generation, column with negative reduced cost is

generated by solving a pricing subproblem to minimize objective (18). However, this approach does not suit our model, because the current (**RMP**) only contains a subset of batches $\bar{B} \subset B$, and the values of some dual variables corresponding to rows in constraints set (3) with indices belonging to $B \setminus \bar{B}$ are unknown. However, the column μ_s with negative reduced can also be generated based on the current batches, $\bar{B}$, through solving the following modified pricing subproblem.

$$\min_{s \in S} \left(\omega_2 c(s) - \sum_{b \in \bar{B}} A_{bs} \rho_b - \theta \right) \tag{19}$$

The pricing subproblem (19) is to find a subset of batches in $\bar{B}$ and a schedule of those batches on a machine such that the total weighted completion time of batches in the schedule, $\omega_2 c(s)$, minus the total dual variable value of these batches, $\sum_{b \in \bar{B}} A_{bs} \rho_b$, is minimized. With a set of given batches, the optimal sequence of batches is given by sequencing the batches in non-increasing order of the ratio $w(b)/p(b)$. Based on this property, the pricing subproblem (19) can be solved by a dynamic programming.

1) Re-index batches in $\bar{B}$ such that $w(b_1)/p(b_1) \geq w(b_2)/p(b_2) \geq \ldots \geq w(b_{|\bar{B}|})/p(b_{|\bar{B}|})$.

2) Let $f(l, t)$ be the minimum reduced cost for all feasible single-machine schedules that consist of batches from the set $\{b_1, \cdots, b_l\}$ in which the last batch is completed at time $t \leq P(l) = \sum_{k=1}^{l} p(b_k)$.

3) Initial values: $f(0,0)=0$, $f(l, t)=\infty$ for $l \in \{1, 2, \ldots, |\bar{B}|\}$, $t \leq P(l)$.

4) Recursive relation: for $l \in \{1, 2\ldots, |\bar{B}|\}$, $t \leq P(l)$, $f(l, t) = \min \{f(l\text{-}1, t), f(l\text{-}1, t\text{-}p(b_l)) + w(b_l)t - \rho_{b_l}\}$.

5) The problem (19) is solved by computing: $f^* = \min \{ f(|\bar{B}|, t) \mid 0 \leq t \leq P(|\bar{B}|)\}$.

If $f^*-\theta<0$, the corresponding column μ_s with negative reduced cost is found, and it will be added to (**RMP**) which is re-optimized by simplex method. However, the fact that $f^*-\theta \geq 0$ can not guarantee the optimality conditions of (**LTSSP**), because $\bar{B}$ contains only a subset of all possible batches, columns with negative reduced cost may exist if $\bar{B}$ is enlarged.

3.3 Generating Rows

In this section, we discuss the case of generating new batches (rows are also generated because each of constraints set (3) is associated with a batch). The motivation of generating new batches is to obtain the descending solution of (**RMP**) such that the corresponding new λ_b variables become basic. From the constraints (3), we know that if a new λ_b variable becomes basic variable, there exists a new schedule $s \in S \setminus \bar{S}$ in which the batch b is included such that the corresponding variable μ_s will also become the basic variable. Based on this fact, batch and schedule need to simultaneously generate for obtaining the descending solution of (**RMP**). Because the choices on generating new batch are difficult decisions, we take the idea from [14] that assume one new batch should be generated at a time. Using this assumption, for a core job j, we first generate a new batch $b^* \in B^j \setminus \bar{B}^j$, and then generate a new batch $s \in S \setminus \bar{S}$ that contains the new batch b^*.

Let $\boldsymbol{\alpha}$ be vectors of dual variables corresponding to constraints (2) of (**LTSSP**), the reduced cost of the column λ_{b^*} corresponding to a new batch b^* is given by the

following formula:

$$\sigma_{b^*} = \omega_1 q(b^*) - \sum_{i=1}^{n} a_{ib^*}\alpha_i + \rho_{b^*} \tag{20}$$

Since the new λ_{b^*} variable is expected to become a basic variable, we have $\sigma_{b^*} = 0$ which also indicates $\rho_{b^*} = \sum_{i=1}^{n}(\alpha_i - \omega_1 q_{ij})a_{ib^*}$.

The new batch b^* will be included in a new column $\mu_s, s \in S \setminus \bar{S}$, and therefore the reduced cost of new column μ_s is given by the following formula:

$$\sigma_s = \omega_2 c(s) - \sum_{b \in \bar{B}} A_{bs}\rho_b - \rho_{b^*} - \theta \tag{21}$$

To minimize σ_s, it is likely that ρ_{b^*} has maximal value. Thus, the following knapsack problem is solved to generate a new batch $b^* \in B^j$.

$$\max \quad \rho_{b^*} = \sum_{i \in L_j}(\alpha_i - \omega_1 q_{ij})a_{ib^*} + \alpha_j \tag{22}$$

subject to

$$\sum_{i \in L_j} a_{ib^*} v_i + v_j \le V , \tag{23}$$

$$a_{ib^*} \in \{0,1\} , \forall i \in L_j . \tag{24}$$

After the new batch $b^* \in B^j$ has been generated, update $\bar{B} = \bar{B} \cup \{b^*\}$, and then generate a new schedule that contains batch b^* by the dynamic programming present in the previous section with a slight modification of recursion i.e., $f(l^*, t) = f(l^*\text{-}1, t\text{-}p(b_{l^*}) + w(b_{l^*})t - \rho_{b_{l^*}}$, for $b_{l^*}=b^*$, $t \le \mathrm{P}(l^*)$ and $f(l, t) = \min\{f(l\text{-}1, t), f(l\text{-}1, t\text{-}p(b_l)) + w(b_l)t - \rho_{b_l}\}$ for $l \ne l^*$, $t \le \mathrm{P}(l)$.

Once the new generated schedule s containing the new batch b^* has negative reduced cost, two columns λ_{b^*} and μ_s and a row with index b^* in the constraints set (3) are added to (**RMP**). Note that for each core job $j \in \{1, \ldots, n\}$, we try to generate a new batch and a corresponding new schedule in the row generation procedure.

3.4 Getting Integer Solution

With generated rows and columns of the final (**RMP**), an integer programming is obtained by re-imposing binary requirements on λ_b and μ_s variables. The integer programming can be solved by CPLEX with an integer solution which is probably closed of the optimal solution of (**TSSP**).

4 Computational Results

To test the performance of the algorithm and study the characteristics of the solution, we conducted a computational experiment on a range of test problems. Our algorithm was implemented in C++ using Microsoft Visual C++ 6.0 compiler and ran in Windows XP on a HP Compaq PC with 3.0 GHz CPU and 1 GB RAM.

The test problem instances were generated at random as follows. The processing time of each job was randomly generated from discrete uniform distribution U[10, 30]. The weight of each job was randomly generated from discrete uniform distribution U[10, 100]. The volume of each job was randomly generated from discrete uniform distribution U[1, 3]. The value of the attribute of each job (va_i) was

randomly generated from discrete uniform distribution U[10, 40]. If the absolute value of dissimilarity between two attributes $|va_i - va_j|$ is large than 15, we regard that two jobs are incompatible, i.e., $i \notin L_j$, otherwise, the dissimilarity cost $q_{ij} = |va_i - va_j|$.

Three parameters are chosen to represent the problem structure as described below: number of jobs $n \in \{20, 30, 50\}$, number of machines $m \in \{2, 4\}$, capacity of the machine $V \in \{8, 12\}$. Form the combination of parameter levels 12 problem scenarios were selected, and for each scenario, 25 different problem instances were randomly generated. Thus totally 300 instances were used in the experiment.

The computational results are reported in Table 1. The headers of the columns in the table are interpreted as follow. Columns (1)-(3) represent the problems structure in terms of number of jobs, number of machines and capacity of the machine. Column (4) and column (5) present the average and maximum integrality gaps respectively between the integer feasible solution of (**TSSP**) which is obtained by solving the integer programming with generated rows and columns) and the solution of the final (**RMP**) which is obtained by row-and-column generation. Column (6) and column (7) present the average and the maximum computational times in seconds respectively for the problems.

From Table 1, we observe that the average and maximum gaps between the integer feasible solution of (**TSSP**) and the solution of the final (**RMP**) are within 1% and 2% for all instances. Because the solution of the final (**RMP**) provides an estimated lower bound on (**TSSP**), this result indicates that near-optimal solution of (**TSSP**) is found. As the number of machines increases, the computational time reduces because having more available machines can reduce the job competition for machines and make the problem become easier. For the test instances, our algorithm can find near-optimal solutions with reasonable computational time. It indicates that proposed row-and-column generation method has great potential for solving the similar large-scale optimization problems.

Table 1
Computational Results of the Test Problems

Problem			Gap		CPU time (s)	
n	m	V	Avg.	Max.	Avg.	Max.
20	2	8	0.65%	1.28%	3.32	10.38
20	2	12	0.53%	1.36%	3.06	11.82
20	4	8	0.73%	1.19%	1.22	4.39
20	4	12	0.52%	1.25%	1.05	3.62
30	2	8	0.76%	1.97%	15.27	38.93
30	2	12	0.55%	1.20%	12.75	30.59
30	4	8	0.93%	1.56%	6.36	20.46
30	4	12	0.87%	1.43%	5.12	14.09
50	2	8	0.85%	1.78%	240.29	335.93
50	2	12	0.91%	1.92%	193.35	276.32
50	4	8	0.99%	2.02%	94.37	189.51
50	4	12	0.87%	1.74%	85.82	156.43

5 Conclusion

In this paper, we study a batch machine scheduling problem in which decisions on grouping jobs to batches and scheduling batches on parallel machines are jointly made. To address this problem, we model it as two-stage set-partitioning type

formulation. To find solutions, we propose a row-and-column generation method in which rows and columns are both dynamically generated. The proposed method assumes that one new batch should be generated at a time in each iteration, it may not generate all needed rows. As a consequence, it is a near-optimal method. However, the results show good quality solutions, which are probably close to the optimal, suggesting the application of such method to other similar problems. Further research topic can focus on the optimality conditions for the linear relaxation of (**TSSP**) and embedding the row-and-column generation in the branch-and-bound framework to find optimal solution.

Acknowledges

This research was supported by Doctoral Program Foundation of Institutions of Higher Education of China (Grant No. 20090042120038), and the Fundamental Research Funds for the Central Universities of China (Grant No. N090104002 and N090304014).

References

[1] G. Dobson, R. S. Nambimadom, The batch loading and scheduling problem, Oper. Res., 2001, 49(1): 52–65.
[2] T. C. E. Cheng, Z. H. Liu, W. C. Yu, Scheduling jobs with release dates and deadlines on a batch processing machine, IIE. Trans., 2001, 33: 685–690.
[3] Z. H. Liu, W. C. Yu, Scheduling one batch processor subject to job release dates, Disc. Appl. Math., 2000, 105: 129–136.
[4] Z. H. Liu, J. J. Yuan, T. C. E. Cheng, On scheduling an unbounded batch machine, O. R. Lett., 2003, 31: 42–48.
[5] W. H. Li, J. J. Yuan, Single machine parallel batch scheduling problem with release dates and three hierarchical criteria to minimize makespan, machine occupation time and stocking cost, Int. J. Prod. Econ., 2006, 102: 143–148.
[6] S. Webster, K. R. Baker, Scheduling groups of jobs on a single machine, Oper. Res., 1995, 43: 692–703.
[7] P. Brucker, A. Gladky, H. Hoogeveen, M. Y. Kovalyov, C. N. Potts, T. Tautenhahn, S. L. van de Velde, Scheduling a batching machine, J. Scheduling., 1998, 1: 31-54.
[8] C. N. Potts, M. Y. Kovalyov, Scheduling with batching: a review, Euro. J. Oper. Res., 2000, 120: 228–249.
[9] Z. L. Chen, W. Powell, Solving parallel machine scheduling problems by column generation, INFORMS. J. Comp., 1999, 11(1):78–94.
[10] J. M. van den Akker, J. Hoogeveen, S. van de Velde, Parallel machine scheduling by column generation, Oper. Res., 1999, 47: 862–872.
[11] Z. L. Chen, C. Lee, Parallel machine scheduling with a common due window, Euro. J. Oper. Res., 2002, 136: 512–527.
[12] C. Y. Lee, Z. L. Chen, Scheduling Jobs and Maintenance Activities on Parallel Machines, Nav. Res. Logis., 2000, 47: 145-165.
[13] Z. L. Chen, W. Powell, Exact algorithms for scheduling multiple families of jobs on parallel machines, Nav. Res. Logis., 2003, 50: 823–840.
[14] E. J. Zak, Row and column generation technique for a multistage cutting stock problem, C & OR, 2002, 29: 1143-1156.