Preemptive Jobs Scheduling on Parallel Machines with Setup Times and Renewable Resources

Tomasz Śliwiński, Eugeniusz Toczyłowski
Warsaw University of Technology,
Institute of Control & Computation Engineering,
00-665 Warsaw, Poland

Abstract. A two-stage algorithm for scheduling preemptive jobs on parallel machines with minimum makespan criterion and requirements for limited renewable resources and existence of sequence dependent setup times is investigated. In the first stage of the algorithm a set of best elementary feasible plans is obtained through column generation. For the second stage we compare genetic algorithms for sequencing elementary plans, where various approximate criterions for calculation minimum makespan are used.

1 Introduction

Problem of scheduling jobs on parallel machines with the presence of renewable resource constraints and setup times is of considerable importance in many process industries such as chemical, pharmaceutical, food, etc.

Machines working in parallel constitute a cell where simultaneous processing of different jobs (production of different item types) is possible. Set of jobs K is to be processed by the set of heterogenous machines L working in parallel. The jobs can be divided in time and space, i.e. each job can be split and processed on many machines, it can also be interrupted at any time and later resumed on the same or different machine. The machine cannot process more then one job at the time.

Machines are not fully independent - we consider the case where there is a common renewable resource shared between the machines. Examples of such a resources are workers or tools needed for production and available in a limited number. We consider a general case, when the amount of the resource allocated for processing a job depends on that job and on the machine.

The change of the job processed on a given machine entails the setup/changeover to occur. Its duration depends on the previous and the following jobs and on the machine.

Please use the following format when citing this chapter:

Śliwiński T. and Toczylowski E., 2008, in IFIP International Federation for Information Processing, Volume 257, Lean Business Systems and Beyond, Tomasz Koch, ed.; (Boston: Springer), pp. 29–39.

The goal is to minimize the total length of the schedule (makespan) preserving all the required constraints.

2 Two-phase approach

The difficulty of the problem is related to the presence of resource constraints and setup times in a single model. The problem of scheduling jobs on a single machine with changeovers is NP-hard. Increasing the number of machines and adding common renewable resource constraints makes it even more difficult to solve. Some heuristics for such problems were presented in [1, 2, 3, 5, 6]. The approximate two-phase algorithm presented in this paper extends the structural approach proposed by Toczyłowski [7, 8].

Modern production systems are designed to minimize setup times, which then represent not more than a few percent of the total production time. This justifies the two-phase approach where in the first phase only production planning with resource availability problem is considered and changeovers are left aside for detailed scheduling performed in the second phase.

In the presented approach production planning can be seen as an asynchronous, multistage decision process, where decision horizon is divided into many elementary stages. Each of them spans a different period of time when the state of the process, i.e. assignment of jobs to machines and resource allocation, is constant. The single stage will be called elementary production plan. The special structure of the problem makes it possible to decompose it to subproblems in which resource constraints only for one elementary production plan are taken into consideration. This approach based on the Danzig-Wolf decomposition and called column generation technique greatly simplifies and accelerates the optimization process. The result of this phase is the set of elementary production plans used later as the starting point for detailed scheduling.

Unfortunately, first phase column generation algorithm that creates optimal set of elementary production plans does not take into account neither the setup times between consecutive plans nor even the detailed schedules. Moreover, the sequence of the plans is fully random with respect to the setup times. This is why the second phase is needed. There are two problems that can be formulated in this phase. One is to sequence elementary plans by minimizing the setup times/costs, and the other is to calculate the detailed schedule based on the sequenced plans. Sometimes, the two above problems are not distinguishable, which means that detailed schedules are needed during sequencing to produce reliable results. Of course this approach greatly increases the complexity of the problem, so in most cases only an approximate measure of the objective is used during sequencing whereas the detailed scheduling is performed as the final step.

3 Production planning

Let us introduce the following notation.

Indices
k – job $(k \in K)$
l – machine $(l \in L)$
β – elementary production plan $(\beta \in B)$
Inputs
p_{lk} – processing time of the entire job k on machine l
α_{lk} – amount of renewable resource allocated for job k on machine l
W – renewable resource available
Decision variables
y_β – duration of β
v_{lk}^β – binary variable equal 1 if and only if job k is processed on machine l in elementary production plan β

Set of elementary production plans minimizing the makespan can be obtained by solving the following problem:

$$\min \sum_{\beta \in B} y_\beta \tag{1}$$

$$\sum_{\beta \in B} \left(\sum_{l \in L} \frac{1}{p_{lk}} v_{lk}^\beta \right) y_\beta = 1, \quad k \in K \tag{2}$$

$$\sum_{k \in K} v_{lk}^\beta \le 1, \quad l \in L, \beta \in B \tag{3}$$

$$\sum_{l \in L} \sum_{k \in K} \alpha_{lk} v_{lk}^\beta \le W, \quad \beta \in B \tag{4}$$

$$v_{lk}^\beta \in \{0, 1\}, \quad l \in L, \ k \in K, \beta \in B \tag{5}$$

$$y_\beta \ge 0, \quad \beta \in B \tag{6}$$

Constraints (2) ensure, that each job will be completed. Constraints (3) and (4) apply to only one elementary plan. First of them prevent processing more than one task on a single processor. Second, ensure that the renewable resource constraints are not violated.

The problem (1)–(6) is a difficult quadratic programming problem with number of variables growing exponentially with its size. Its structure, however, makes it possible to solve it effectively by applying Danzig-Wolfe decomposition and thus the column generation method. This kind of decomposition can be used due to the existence of two groups of the constraints. First group (2) binds all the variables y_β present in the objective function. Second group (3)–(4) applies to only one column of the constraints matrix (2).

3.1 Master problem

Applying Danzig-Wolfe decomposition to problem (1)–(6) one gets the following master problem:

$$\min \sum_{\beta \in B} y_\beta \tag{7}$$

$$s.t. \sum_{\beta \in B} (\sum_{l \in L} \frac{1}{p_{lk}} v_{lk}^{\beta}) y_{\beta} = 1, \quad k \in K \tag{8}$$

$$y_{\beta} \geq 0, \quad \beta \in B \tag{9}$$

One should notice, that in contrary to the original problem, v_{lk}^{β} are not variables but constant parameters of linear constraints (8). Constraints matrix must include all the columns corresponding to all feasible combinations of these parameters, thus, the number of variables y_{β} can be enormous. However, not all columns have to be hold in the computer memory, instead, they can be handled implicitly with the column generation scheme. We only need to identify the best column during the pricing and to generate selected column for pivoting.

3.2 Pricing problem

New column generated during the pricing problem should minimize the reduced cost of the corresponding variable y_{β}. For the problem (7)–(9) the reduced cost is given by the formula

$$y_{0\beta} = 1 - \sum_{k \in K} (\sum_{l \in L} \frac{1}{p_{lk}} v_{lk}^{\beta}) \pi_k = 1 - \sum_{k \in K} \sum_{l \in L} (\frac{\pi_k}{p_{lk}}) v_{lk}^{\beta},$$

where π is a vector of dual variables of the current basic solution corresponding to constraints (8). The column generated during pricing defines assignment of jobs to machines and thus one elementary production plan. Feasibility of that plan can be ensured by additional constraints

$$\sum_{k \in K} v_{lk}^{\beta} \leq 1, \quad l \in L \tag{11}$$

$$\sum_{l \in L} \sum_{k \in K} \alpha_{lk} v_{lk}^{\beta} \leq W \tag{12}$$

$$v_{lk}^{\beta} \in \{0,1\}, \quad l \in L, \; k \in K \tag{13}$$

The pricing problem can be extended with any suitable constraints that apply to single elementary plan, for example, one can define maximum number of machines ($L_{\max}$) that can process given job $\sum_{l \in L} v_{lk}^{\beta} \leq L_{\max}$

4 Detailed scheduling

Detailed scheduling can be seen as two overlapping problems. One is sequencing of elementary production plans and the other is detailed scheduling based on the given sequence. In our work we decided to separate sequencing and detailed scheduling by applying approximate sequence quality measures during sequencing. In a perfect case optimal sequence in the sense of the applied measure should result in the optimal final schedule. Unfortunately, there is no simple way of constructing a measure, that ensures optimal final detailed schedule and is computationally

efficient. That is why we decided to apply approximate sequence quality measures resulting in reasonable good final schedules.

For the given sequence of elementary production plans the quality measure can be evaluated with one of the three methods differing in computational effort: (i) total changeover time, (ii) critical path method, and (iii) critical path method with job reallocation. Measures (ii) and (iii) can be used to determine relatively accurate final schedules.

Total changeover time. Processing a job on a single machine in one elementary production plan will be later called *operation*. In the system with one processor finding the optimal sequence of elementary plans corresponds to minimizing the total changeover time. But in general, in the systems with multiple parallel machines the shortest total changeover time doesn't guarantee the best detailed schedule. The reason for that is the need for full time access to the common resource and the resulting synchronization between schedules on different machines and in different elementary plans.
The experiments showed the total changeover time can be considered as a very good quality measure of the elementary plans sequence. Computational efficiency is its most important property.

Critical path method. Hindi and Toczyłowski [4] developed an approach where detailed schedule for a given sequence of elementary plans can be found using critical path method. The idea is to build a task graph taking into account precedence relationships between operations and changeovers. The task graph is constructed in a way that prevents overlapping of the neighboring elementary plans which would result in the violation of the resource constraint. Its structure is defined by precedence relations for all operations according to the sequence of the elementary plans they belong to. The simplest way of establishing a consistent and sufficient set of precedence relationships is to consider as followers for each operation the set of operations, one for each machine, that follow it soonest.

Precedence relations for sample sequence of elementary plans are shown in Fig.1. If one associates the processing times with the nodes and changeover times with the horizontal edges, the resulting graph is a potential task graph. Hence, the minimum makespan (critical path) and the earliest starting times of each operation can be easily determined by finding the longest path.

Critical path method can be used as an approximate quality measure for sequencing. It is more accurate then total changeover time and still very effective computationally. The major drawback of this method is poor density of the resulting schedule – machines have to wait for all operations of the previous elementary plan to complete. Toczyłowski [9] proposed significant improvement to this method by allowing shifting of operations between elementary plans.

Critical path method with job reallocation. In this approach we are allowed to shift small portions of the job between elementary plans the given job was assigned to, thus reducing idle time of the processors. An efficient way to achieve this is to construct an LP model corresponding to the task graph with variable duration of

operations. Unfortunately, this approach doesn't take into account the possibility of removing empty operations, i.e. operations with resulting duration of 0. In such a case, an approximate approach is suggested. All empty operations together with corresponding changeovers are removed and the new LP model is constructed. This procedure is repeated as long as one can find at least one empty operation in the results of the LP model. Let us introduce the following notation:

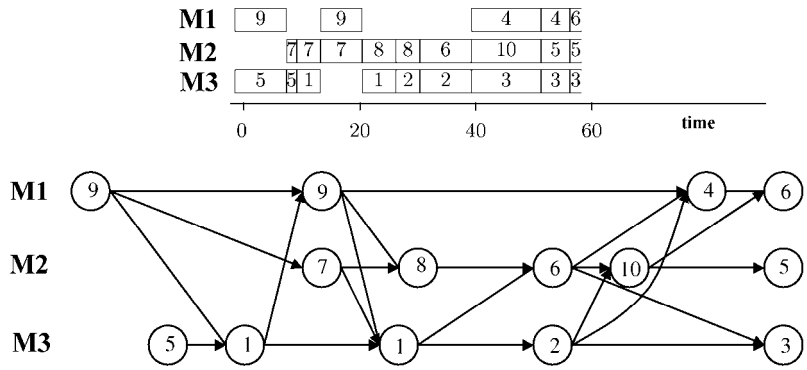

Fig. 1. Sequence of elementary production plans and the corresponding precedence graph

- O – set of all operations if the precedence graph $(n \in O)$
- O^k – set of all operations job k consists of
- l_n – machine where the operation n is processed
- k_n – job the operation n belongs to
- p_n – processing time of the whole job k_n on machine l_n $(p_n = p_{l_n k_n})$
- x_n – portion of job k_n corresponding to operation n
- s_n – moment when the operation n starts
- e_n – moment when the operation n ends
- F^n – the set of followers of n in the precedence graph
- f^n – direct follower of operation n on the same machine
- B – set of operations without followers
- c_{nm} – changeover time between two consecutive operations n and m on the same machine

Now we can formulate LP problem corresponding to the critical path method with job reallocation.

$$\min T \tag{14}$$

$$e_n = s_n + x_n p_n, \quad n \in O \tag{15}$$

$$e_n \le s_m, \quad n \in O, m \in F^n \tag{16}$$

$$e_n \le c_{nf^n} + s_{f^n}, \quad n \notin B \tag{17}$$

$$e_n \le T, \quad n \in B \tag{18}$$

$$\sum_{n \in O^k} x_n = 1, \quad k \in K \tag{19}$$

$$0 \le x_n \le 1, \quad n \in O \tag{20}$$

The objective (14) expresses the makespan. Constraints (15) and (16) ensure correct precedence relations between operations in the consecutive elementary plans. (17) inserts changeovers to the schedule and (19) ensures that each job in the resulting schedule will be completed.

5 Sequencing of elementary production plans

The problem of sequencing elementary production plans is of permutational nature. For each sequence one can compute some quality measure. Finding optimal sequence equals finding permutation minimizing this measure.

In our work we decided to use genetic algorithm as a basic approach to the elementary plans sequencing problem. The major drawback of this method is the need for a large number of objective computations during the optimization process. As the result, genetic algorithms are best suited for applications where the objective value can be obtained relatively easy. We also decided to use this approach because genetic algorithms do not need special knowledge about the problem, what in this case is of great importance. Simpler problem of scheduling jobs on a single machine with total changeover time as the quality measure can be modeled as the traveling salesman problem. However, when critical path method with or without job reallocation is used as the quality measure, the problem considered here lays far beyond the TSP problem.

Basic scheme of the genetic algorithm applied in our experiments is following. First, a set of the random feasible solutions (sequences) is generated. Then they are evaluated with one of the quality measures. Basing on the resulting values, some of the solutions are selected for mutation and crossover. Changed or totally new solutions are evaluated again and the procedure repeats.

An example of sequencing and detailed scheduling is shown in Figure 2. For the production plan from Figure 1 the optimal sequence of elementary plans was computed using the genetic algorithm. The sequence is optimal (suboptimal) according to the final detailed schedule. That means the detailed schedule had to be computed for all sequences that were constructed by the genetic algorithm. The optimal sequence is shown in Fig. 2a. The detailed schedule for the given sequence computed using the critical path method is shown in Fig. 2b. The final detailed schedule computed using the critical path method with job reallocation is shown in Fig. 2c.

6 Computational experiments

Computational experiments were carried out for the first and for the second phase of the presented algorithm. All tests were based on the randomly generated problems. The generation procedure worked as follows. First, processing times p_{lk} and required resources α_{lk} were generated as random numbers uniformly distributed in the intervals, respectively, [10, 30] and [6, 10]. Next, the amount of the common renewable resource was determined in such a way that on average only 75% of

machines could operate. For problems where there are fewer jobs than machines, this resource constraint was set to the level, where only 75% of jobs were processed at the same time. Finally, for each machine – previous job – following job combination changeover times were generated as random numbers uniformly distributed in the interval [1.2, 2.0], which represent 6 to 10 percent of the average processing time p_{lk}. All computations were performed on a PC with the Pentium 1.7 GHz, employing the CPLEX 9.1 package.

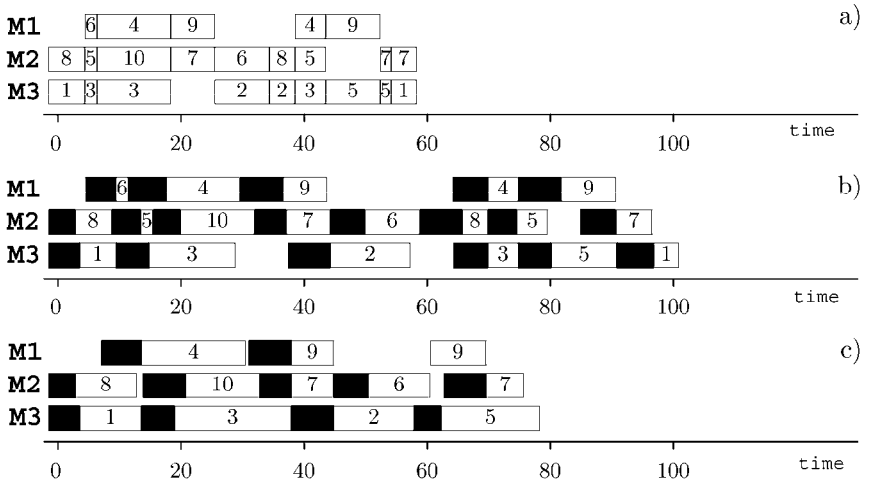

Fig. 2. The sequence minimizing critical path with job reallocation: a) the sequence, b) detailed schedule for the given sequence computed using critical path method, c) detailed schedule computed using critical path method with job reallocation

A series of tests were performed to evaluate the computational efficiency of the first phase. We tested solution times for different numbers of jobs and machines and different levels of common resource available.

The results presented in Tab. 1 are the average times for 10 random problems with freely available common resource. In Tab. 2 are shown results for problems with active common resource constraint. The index states for the number of tests (among 10) for which the timeout of 200 seconds occurred.

We also performed computational tests for the second phase of the algorithm. The purpose of the experiments was to compare sequencing algorithm using two approximate quality measures (total changeover time and critical path method) and one exact quality measure (critical path method with job reallocation). The resulting values are the average for 10 random problems with 10 machines, 40 jobs and active common resource constraint. We used following parameters for the genetic algorithm: mutation probability 0.06, crossover probability 0.5, population size 50.

Table 1. Solution times in the first phase for problems with freely available common resource

number of machines	number of jobs			
	10	20	30	40
10	0.0	0.2	0.3	0.6
20	0.0	0.3	0.8	1.5
30	0.0	0.4	0.7	2.1
40	0.1	0.4	0.7	2.7

Table 2. Solution times in the first phase for problems with restricted common resource

number of machines	number of jobs			
	10	20	30	40
10	0.2	1.6	2.2	5.1
20	0.8	5.3	8.6	17.0
30	1.0	9.7	20.2	41.3
40	0.8	^{9}82.3	53.5	^{2}61.7

In the first experiment the convergence of the sequencing algorithm was tested. The results presented in Fig. 3 were determined as if the sequencing algorithm stopped after a given number of seconds and for the best sequence found so far final detailed schedule was computed using critical path method with job reallocation. This explains why plots corresponding to approximate measures are not monotonic.

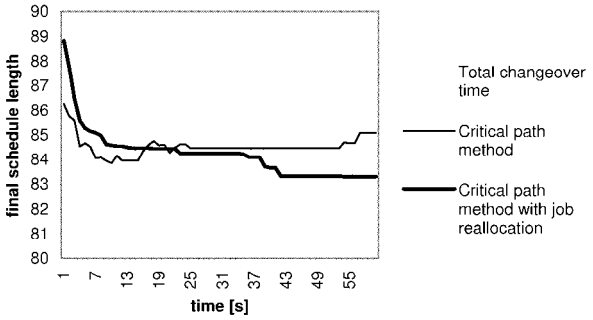

Fig. 3. Convergence of the sequencing algorithm in time for different sequence quality measures

In the second experiment the convergence of the sequencing algorithm was tested as a function of the number of objective computations. The tests were carried out up to 5000 objective computations. The test problems are exactly the same as in the previous experiment. The results are shown in Fig. 4.

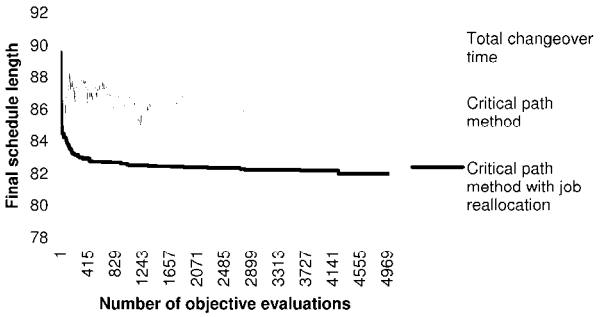

Fig. 4. Convergence of the sequencing algorithm as the function of objective computations for different sequence quality measures

One can notice the algorithm employing the exact quality measure, i.e. critical path method with job reallocation, performed much better than algorithms with approximate quality measures. This was achieved, however, in the expense of the computing time – the algorithm with the approximate quality measure ended after a few seconds while the algorithm with the exact measure needed over half an hour to complete.

7 Concluding remarks

Presented here structural algorithm for preemptive scheduling of jobs on parallel machines with changeovers and resource constraints integrates effectively different elementary optimization techniques including column generation method, evolutionary algorithms, linear programming and specialized heuristics. The results of the computational experiments indicate high efficiency of the total changeover time used as the sequence quality measure in applications with limited computation time. When this is not a problem the best results can be obtained using critical path method with job reallocation.

References

1. Dobson G., Karmarkar U., Rummel J.: Batching to minimize flow times on parallel heterogeneous machines. Mgmt Sci. 35 (1989) 607-613.

2. Figielska E.: Preemptive scheduling with changeovers: using column generation technique and genetic algorithm. Computers & Industrial Engineering 37 (1999) 81-84.

3. Figielska E., Toczyłowski E.: Algorytm szeregowania podzielnych operacji na równoległych procesorach przy ograniczeniach zasobów zużywalnych

i odnawialnych. Optymalizacja w zagadnieniach kombinatorycznych, Wyd. WSM (1997) 18-31.

4. Hindi K. S., Toczyłowski E.: Detailed Scheduling of Batch Production in a Cell with Parallel Facilities and Common Renewable Resources. Computers and Industrial Engineering 28 (1995) 839-850.

5. Oliff M.D.: Disaggregate planning for parallel processors. IIE Trans 19 (1987) 215-219.

6. So K.: Some heuristics for scheduling jobs on parallel machines with setups. Mgmt Sci. 36 (1990) 467-475.

7. Toczyłowski E.: Niektóre metody strukturalne optymalizacji do sterowania w dyskretnych systemach wytwarzania. WNT (1989).

8. Toczyłowski E.: Algorithms for preemptive scheduling of independent tasks in the presence of general renewable and consumable resources. Zeszyty Nauk. AGH, Automatyka 59 (1991) 163-172.

9. Toczyłowski E.: Preemptive Scheduling of independent Tasks in the Presence of Setup Times and Renewable Resources. 17th International Symposium on Mathematical Programming ISMP'2000, 7-11 August 2000, Atlanta, USA.