A Lot Scheduling Problem on a Single Machine with Indivisible Orders

Wen-Hung Kuo and Dar-Li Yang
Department of Information Management
National Formosa University
Yunlin, Taiwan, R.O.C.
Email:{whkuo,dlyang}@nfu.edu.tw

***Abstract* — In this paper, a lot scheduling problem on a single machine with indivisible orders is studied. The objective is to minimize the total completion time of all orders. We use a binary integer programming approach to solve this problem. The binary integer programming approach with run time limit is considered as one heuristic method. As compared to a lower bound, it turns out the average performance of the method is really good.**

Keywords- lot scheduling; single machine; total completion time; indivisible order; integer programming

I. INTRODUCTION

Generally, there are two main production processes in a production system, that is, continuous production and batch production. Here, we are interested in batch production. In the literature, there are two categories of batch scheduling problems. One is batch scheduling with divisible batch sizes. Naddef and Santos [1] studied a single machine problem with batching jobs. The objective is to minimize the total completion times. They showed that the greedy algorithm solves the problem if jobs are all of one type. They also provided a heuristic for the problem with various job types. Coffman et al. [2] considered a single machine job shop in which subassemblies of two different types are made and then assembled into products. They provided an efficient algorithm for minimizing the total flow time of the products. Dobson et al. [3] considered batch jobs in the multiple-machine scheduling problem. The objective is to minimize the mean flow times. They proposed an efficient algorithm for computing the optimal solution for single product case. Hou et al. [4] studied a lot scheduling problem with orders which can be split. Orders are grouped into lots and then processed. The objective is to minimize the total completion time of all orders. They showed that this problem can be solved in polynomial time.

The other is batch scheduling with indivisible batch sizes. Shallcross [5] studied a problem of batching identical jobs on a single machine. He presented an algorithm to minimize the sum over all jobs of the batched completion times. Mosheiov et al. [6] addressed a classical minimum flow-time, single-machine, batch-scheduling problem. They introduced a simple rounding procedure for Santos and Magazine's solution [7], which guarantees optimal integer batches. Mor and Mosheiov [8] studied an identical parallel-machine scheduling problem with identical job processing times and identical setups. They showed that the solution is given by a closed form, consisting of identical decreasing arithmetic sequences of batch sizes on the different machines.

In a factory, products are usually made according to customers' orders. This production approach is called MTO (make to order). Since different orders may contain different quantities, two production strategies are applied in the batch production, especially when the lot size of the batch production is fixed. Also, in this particular situation, the production time of each lot is fixed no matter how many quantities in the lot. Therefore, from the viewpoint of efficiency, one order may be divided into several lots to fill up each lot. The study presented by Hou et al. [4] is based on this viewpoint. However, from the viewpoint of management, one order is not divided into different production lots because the products of the same order are finished at the same time and then delivered to the customer. Based on this viewpoint, in this paper, we study the problem given by Hou et al. [4] but orders are restricted to be indivisible.

The rest of the paper is organized as follows. In the second section, a description of the problem is given. Next, the integer programming formulation is provided. Computational experiments are given in the fourth section. Final section is the conclusion.

II. PROBLEM DESCRIPTION

There are n orders (O_i, i = 1, 2, …, N) to be grouped into lots and then be processed on a single machine. Each order has its own size (σ_i, i = 1, 2, …, N). The size of each order is no more than one lot's capacity (K). On top of that, every order is indivisible. It means products of each individual order have to be processed in the same lot. The orders in the

 ISBN: 978-1-61208-419-0

same lot have the same processing time (*t*). Therefore, all orders in the same lot have the same completion time.

The machine can handle at most one lot at a time and cannot stand idle until the last lot assigned to it has finished processing. The objective is to minimize the total completion time ($\sum C_{O_i}$) of all orders. Thus, using the three-field notation, this scheduling problem is denoted by $1/lot, indivisible/\sum C_{O_i}$.

III. INTEGER PROGRAMMING FORMULATION

The problem is conjectured to be NP hard [9]. Therefore, we use the following binary integer programming approach to solve this problem.

Let $X_{i[q]} = 1$ if the *i*th order is assigned to the *q*th lot, and 0 otherwise. Since the processing time of each lot is *t*, the completion times of the first lot, the second one, etc., are *t*, 2*t*, …, respectively. Thus, the total completion time of all orders is $t\sum_{q=1}^{N}\sum_{i=1}^{N} X_{i[q]}q$. Then, a binary integer programming (BIP) formulation to solve the proposed problem is developed as follows.

$$\text{Minimize} \quad t\sum_{q=1}^{N}\sum_{i=1}^{N} X_{i[q]}q \tag{1}$$

$$\text{Subject to} \sum_{q=1}^{N} X_{i[q]} = 1 \quad i = 1,2,...,N \tag{2}$$

$$\sum_{i=1}^{N} \sigma_i X_{i[q]} \le K \quad q = 1,2,...,N \tag{3}$$

$$X_{i[q]} \in \{0,1\} \quad i = 1,2,...,N,\ q = 1,2,...,N \tag{4}$$

The objective is to minimize the total completion time of all orders which is shown in (1). Equation (2) ensures that each order is only assigned to one lot. Equation (3) limits the total sizes of orders that are assigned to the same lot to the lot capacity (*K*). Finally, (4) guarantees that variable $X_{i[q]}$ is either 0 or 1.

IV. COMPUTATIONAL EXPERIMENTS

The above binary integer programming approach can solve the proposed problem, but it is time-consuming when it comes to a large problem. Considering the efficiency of the BIP, the run time limit of the BIP is set to 3600 seconds. Also, in order to evaluate the performance of the BIP, it is tested in the computational experiments which are conducted based on the following parameter set.

Order number *N* is equal to 20, 30, 40, 50, 60, 70, 80, 90, 100.

Lot capacity *K* is equal to 15, 30.

Order size σ_i is uniformly distributed over [1,5], [1,10] ($\sigma_i \overset{d}{=} \text{U}(1,5)$, $\sigma_i \overset{d}{=} \text{U}(1,10)$)

There are $9 \times 2 \times 2 = 36$ problem types. For each problem type, 30 test problems are generated. Each test problem is solved by BIP and LP, respectively. BIP and LP are solved by using a computer program coded in LINGO 11.0 with 4GB of memory available for working storage, running on a personal computer Intel(R) Core(TM) i7-2600 CPU @ 3.4GHz. To evaluate the performance of the computational results, we have to come up with a lower bound (LB) and then compare these percentage errors ($100*(BIP - LB)/LB$) in different test problems.

Obviously, one lower bound can be obtained from the solution of a variant of the original problem by changing the original problem to the one in which orders are divisible and can be processed in different lots. Therefore, we only need to change (4) as follows.

$$X_{i[q]} \ge 0 \quad i = 1,2,...,N,\ q = 1,2,...,N \tag{4'}$$

Then, since the problem becomes a Linear Programming (LP) problem, we take much less time to solve the problem than the original one. The lower bound is also tight because the solutions of the original problem and its variant can happen to be the same (integers).

The average and maximal percentage errors of each problem type for the BIP solutions and also the number of optimal solutions obtained within 3600 seconds are shown in the following table.

From Table 1, we have the following observations:

(1) For $N = 20$, the optimal solutions for all generated test problems can be found within 3600 seconds.

(2) The larger the lot capacity is or the smaller the order size range is, the more optimal solutions you can obtain within the run time limit.

(3) Average percentage errors of all problem types are less than 2.5, it means that the performance of the binary integer programming with run time limit is really good, especially, in the problem type with parameters *K* = 30 and $\sigma_i \overset{d}{=} \text{U}(1,5)$.

 ISBN: 978-1-61208-419-0

(4) Most maximal percentage errors of all problem types are less than 4.5, it implies that the BIP performs well in most test problems, even in the worst situations.

TABLE1. COMPUTATIONAL RESULTS.

	σ_i =1~5, K =15			σ_i =1~5, K =30			σ_i =1~10, K =15			σ_i =1~10, K=30		
	Error (%)			Error (%)			Error (%)			Error (%)		
N	avg	max	opt. no.	avg	max	opt. no.	avg	max	opt. no.	avg	max	opt. no.
20	0	0	30	0	0	30	0	0	30	0	0	30
30	0	0	30	0	0	30	1.05	9.23	26	0	0	30
40	0.38	3.44	26	0	0	30	1.30	10.08	25	0.15	4.46	29
50	1.08	3.04	13	0	0	30	1.31	10.86	24	0.13	3.93	29
60	1.38	2.77	7	0	0	30	1.34	7.52	23	0.28	3.19	27
70	1.59	2.37	2	0	0	30	1.06	10.24	25	0	0	30
80	1.23	2.38	6	0.56	1.71	18	1.49	8.01	22	0.09	2.63	29
90	0.91	2.54	10	0.57	1.44	15	1.30	7.27	23	0.18	3.15	28
100	1.47	2.06	1	0.86	1.32	3	2.46	6.55	17	0.08	2.47	29

σ_i : order size, K: lot capacity, N: order number

However, some of them in the problems with parameters K = 15 and $\sigma_i \overset{d}{=} \mathrm{U}(1,10)$ are greater than 10, even though their average percentage errors are less than 2.5. The performance of BIP in such problems is not robust. Therefore, it is worthwhile to come up with other heuristics with better performance.

V. CONCLUSION

In this paper, we studied a single-machine lot scheduling problem with indivisible orders. The problem is conjectured to be NP-hard. Therefore, a binary integer programming approach is given to solve the problem. Considering the efficiency of the BIP, the run time limit is set. Also, compared to the lower bound, it turns out the average performance of the BIP within run time limit is really good for all test problems. The maximal percentage errors of the BIP with run time limit are a little greater than 10 in one situation. Therefore, it is worthwhile to find other heuristics with better performance in the future.

ACKNOWLEDGEMENT

This research was supported in part by the National Science Council of Taiwan, Republic of China, under grant number NSC-102-2221-E-150-043-MY2.

REFERENCES

[1] D. Naddef and C. Santos, “One-pass batching algorithms for the one-machine problem,” Discrete Applied Mathematics, vol. 21, pp. 133–45, 1988.

[2] E. D. Coffman, A. Nozari, and M. Yannakakis, “Optimal scheduling of products with two subassemblies on single machine,” Operations Research, vol. 37, pp. 426–36, 1989.

[3] G. Dobson, U. D. Karmarkar, and J. L. Rummel, “Batching to minimize flow times on parallel heterogeneous machines,” Management Science, vol. 35, pp. 607–13, 1989.

[4] Y. T. Hou, D. L. Yang, and W. H. Kuo, “Lot scheduling on a single machine,” Information Processing Letters, vol. 114, pp. 718–722, 2014.

[5] D. F. Shallcross, “A polynomial algorithm for a one machine batching problem,” Operations Research Letters, vol. 11, pp. 213–218, 1992.

[6] G. Mosheiov, D. Oron, and Y. Ritov, “Minimizing flow-time on a single machine with integer batch sizes,” Operations Research Letters, vol. 33, pp.497–501, 2005.

[7] C. Santos and M. Magazine, “Batching in single operation manufacturing systems,” Operations Research Letters, vol. 4, pp. 99–103, 1985.

[8] B. Mor and G. Mosheiov, “Batch scheduling of identical jobs on parallel identical machines,” Information Processing Letters, vol. 112, pp. 762–766, 2012.

[9] M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-completeness. W. H. Freeman, New York, 1979.

 ISBN: 978-1-61208-419-0