6th International Conference on Industrial Engineering and Industrial Management.
XVI Congreso de Ingeniería de Organización. Vigo, July 18-20, 2012

Minimizing the makespan on parallel batch scheduling with stochastic times

Sahraeian R.[1], Samaei F[2]., Rastgar I[3].

Abstract This paper addresses to the problem of batch scheduling in a parallel machine environment with size dependent setup time and release time and minimizing the makespan (C_{max}). The jobs' processing times, setup times and release times are stochastic for better depiction of the real world. The problem is NP-hard. Therefore, in this paper we compare different heuristics with a special simulation approach and analyze the obtained results.

Keywords: Parallel machines, Batch scheduling, Simulation, Stochastic times

1.1 Introduction

In this paper we compare different heuristics for the batch processing machine (BPM) problem scheduling (each batch is a group of different jobs) on a set of identical parallel machines. The BPM have been widely utilized such as semiconductor, printing, chemical industries. The processing time of batch is the longest processing time among the jobs in the batch, and also longest release time of jobs in batch is equal the batch release time. Setup time for specific batch depends to size of batch. The BPM problems are often bottleneck because of their long processing times; thereby in this scheduling problem makespan is an important objective.

[1] Rashed Sahraeian (✉)
Department of Industrial Engineering, Shahed University, Tehran, Iran
e-mail: Sahraeian@shahed.ac.ir
[2] Farshid Samaei
Department of Industrial Engineering, Shahed University, Tehran, Iran
[3] Iman Rastgar
Department of Industrial Engineering, Shahed University, Tehran, Iran

BPM scheduling problems have been studied extensively in recent years. Chandru, et al. (1993) presented heuristics and exact methods for parallel batch processing machine and single batch processing machine problems in order to minimum sum of the compellation times. Chang, et al. (2004) provided simulated annealing (SA) for problem under study and presented a mathematical formulation for minimizing makespan. Kashan, et al. (2008) proposed a Genetic Hybrid which that used a local search, and presented a lower bound for minimizing makespan. Cheng, et al. (2011) provided mixed integer programming method for minimizing makespan and sum of compellation times of jobs and presented polynomial time algorithm. All mentioned papers have deterministic times but stochastic times are much closer to the real world. So, in this paper, the jobs' processing times, setup times and release times are assumed stochastic for better depiction of the real world. The simulation has been populated over the past three decades. The reason for this is that the simulation model can allowed to become quite complex (Kelton, et al. 2004). The BPM problem is NP-hard so we design a special simulation approach for stochastic BPM model. The remainder of this paper is structured as follows.

In the next section, the problem definition is presented. Section 3 deals with simulation model. Section 4 details a brief overview of heuristics. The computational results are presented in Section 5. Finally, a discussion of the results is explained in the last section.

1.2 Problem definition

The problem under study with using the standard three field notation presented by Graham, et al. (1979) can be denoted by $P_m|batch, r_j|C_{max}$. In a BPM, all jobs are divided to different batches. Jobs allocated in a batch are processed simultaneously in start time and then released from the machine together in finish time. The BPM is able to process a number of jobs as long as the sum of job sizes in the batch is not greater than to the capacity of the machine. The BPM problem considered in this paper can be described as follows:

1. There are *n* jobs to be processed by *m* identical parallel batch processing machines.
2. All jobs have arbitrary release time and size.
3. The machines are available at zero time.
4. The jobs' processing times and release times are stochastic for better depiction of the real world.
5. The processing time of a batch is defined with the longest processing time of all the jobs allocated in the batch.
6. Once a batch is processed by a machine, it cannot be Interrupted, i.e., no preemption is allowed.

7. No jobs can be introduced or removed from a batch while the batch is being processed.
8. All the jobs are considered equal in importance.
9. The performance measure is makespan. Our objective is to minimize the makespan.

1.3 Simulation approach

As digital computers appeared in the 1950s and 1960s, people began writing computer programs in general purpose-procedural language like FORTRAN to do simulations of complicated systems. This approach was highly customizable and flexible, but also painfully tedious and error-prone (Kelton, et al. 2004). We apply simulation by programming in MATLAB language because our model is complicated and it framework not kinds of simulation that many people do so special-purpose simulation language like GPSS and SIMAN or high-level simulators like Arena not flexible enough to simulate our model. Discrete-event simulation is used to simulate the real model which is a popular simulation technique, and applicable to a large variety of problems in the real world. In discrete-event simulation, the operation of a system is represented as a chronological sequence of events. Each event occurs at an instant in time and marks a change of state in the system (Stewart, 2004).

Many mechanisms have been proposed for carrying out discrete-event simulation; among them are the event-based, activity-based, process-based and three-phase approaches (Pidd, 2004). The proposed simulation model applies activity-based mechanism. In the model, jobs are entities and machines are recourses according to Kelton, et al. (2004) definition. Fig. 1.1 shows activity diagram of the simulation model.

Two phase structure has defined which in the first phase batch sequence based on each heuristic method has been driven and in the second phase discrete-event simulation is applied. Fig. 1.2 shows the basic model of the phase two.

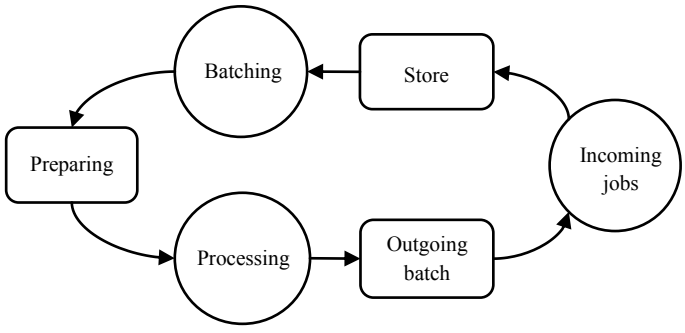

Fig. 1.1 Activity diagram

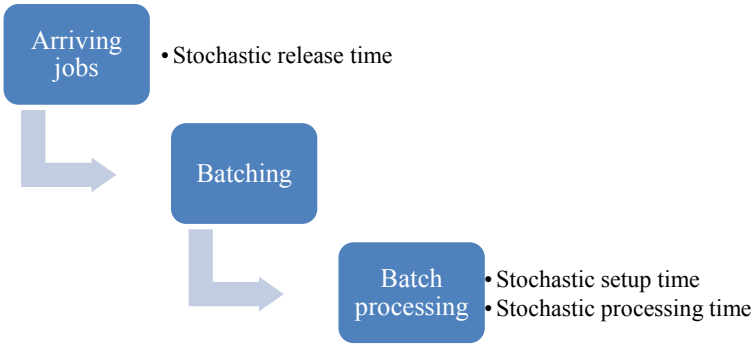

Fig. 1.2 second phase of simulation model

Verification plays a very important role in the model results. We use two strategies for verification our model. First one is applying deterministic data instead of stochastic data for the release, processing and setup times; this allowed us to predict the system's behavior. Second strategy is checking outputs in order to match as close as possible with the real data. The verification of simulated model shows valid simulation has done.

1.4 Heuristic algorithms

The BPM belongs to the class of NP-hard problems. Because of its complexity, the instances with a large number of jobs cannot be solved to optimality within a reasonable time. Therefore, many heuristic and meta-heuristic algorithms have been developed in order to find a near optimal solution in reasonable computational time. In this section, different heuristics are presented. The Marginal Cost heuristic (Damodaran and Velez-Gallego, 2010) is applied to batching arriving jobs. Then four flowing heuristics applies to schedule batches.

1.4.1 ERT

In the ERT (Equalization of Runout Time) rule, the batches are listed in non-decreasing order of their release times (break ties using LPT-Longest Processing Time first), and following that order, each batch is assigned to the first available machine.

1.4.2 ERT-LPT

In the ERT–LPT rule, the batches are listed in non-decreasing order of their release times (break ties using LPT), and following that order, the batches are assigned to the first available machine until the minimum machine release time is greater than or equal to the maximum batch release time. At this point, the batches are assigned to the first available machine by applying the LPT rule.

1.4.3 ERT-SPT

In the ERT–LPT rule, the batches are listed in non-decreasing order of their release times (break ties using LPT), and following that order, the batches are assigned to the first available machine until the minimum machine release time is greater than or equal to the maximum batch release time. At this point, the batches are assigned to the first available machine by applying the SPT (Shortest Processing Time first) rule.

1.4.4 LECT

In LECT rule, the earliest completion time of each batch (i.e., $C_j = r_j + P_j$) is computed. The batches are then listed in non-increasing order of C_j, and following that order, each batch is assigned to the first available machine.

1.5 Computational results

The simulation has been coded in MATLAB 7.1. The presented heuristics have been modeled and compared using the proposed simulation model. All three stochastic times take value of different uniform distributions. We used uniform distributions because of their high variances which ensuring that the presented heuristics are being tested under unfavorable conditions Weng, et al. (2001).

Eight test problems according to Table 1.1 randomly generated. Random generation is based on Kashan, et al. (2008) paper. For each problem, all approaches have been applied and for each case 10000 runs have been applied.

Table 1.1 Test problems

Test Problem	Machine	Job
1	2	10

Test Problem	Machine	Job
2	2	25
3	2	50
4	2	100
5	4	10
6	4	25
7	4	50
8	4	100

The number of runs or replications was obtained by following the steps that Banks, et al. (2004) recommended in order obtaining good confidence intervals. We decided on the tolerable half width that we want and substituted the appropriate values in the following equation which based on $100(1-\alpha)\%$ confidence *t* distribution:

$$H = t_{\frac{\alpha}{2},R-1}\frac{S}{\sqrt{R}} \tag{1.1}$$

where S^2 is sample variance and *R* is the number of runs. Suppose that an error criterion ε is specified; a sample size R must be chosen such that $R \geq R_0$ and $H \leq \varepsilon$. The error criterion is $\varepsilon = 0.001$, and the confidence coefficient is $1-\alpha = 0.99$.

Table 1.2 illustrates the relative performance of four heuristics. The ERT-LPT method significantly outperformed the other methods for all test problems. The relative performance for method *k* was calculated as follows:

$$RP_k = \frac{c_k}{c_{\min}} \tag{1.2}$$

where C_k is the makespan of heuristic k and C_{min} refers to the minimum makespan between all four heuristics.

Table 1.2 Relative performance obtained from simulation

Test Problem	ERT	ERT-LPT	ERT-SPT	LECT
1	0.9647	1	0.9277	0.9484
2	0.9708	1	0.9542	0.9601
3	0.9771	1	0.9329	0.9552
4	0.9669	1	0.9662	0.9648
5	0.9843	1	0.9558	0.9512
6	0.9692	1	0.9236	0.9209
7	0.9798	1	0.9614	0.9643
8	0.9654	1	0.9293	0.9265

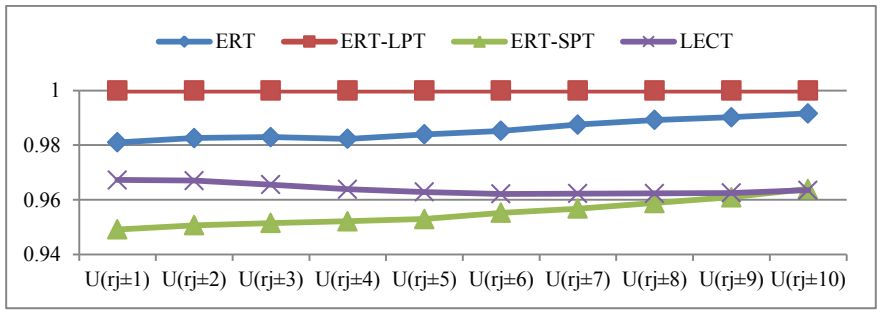

Fig. 1.3 the sensitivity of release time

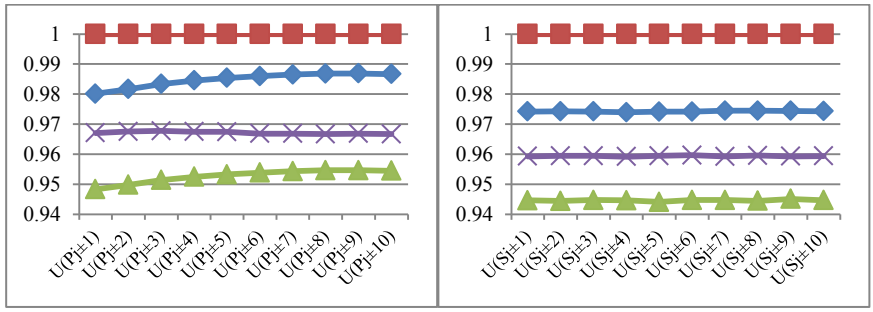

Fig. 1.4 the sensitivity of setup time (left) and the sensitivity of processing time (right)

1.6 Output analysis

For better understanding the behavior of model, the sensitivity analysis has been done for test problem 3. Fig. 1.3 shows the sensitivity of stochastic release time which processing time and setup time considered deterministic. As can be seen, the performance of methods will be close by increasing the interval of uniform distribution. Fig. 1.4 shows the sensitivity of stochastic processing time and setup time, respectively. It is obvious that stochastic processing time and setup time do not have significantly influence which stochastic release time have. Fig. 1.5 shows the sensitivity of all stochastic times together.

1.7 Conclusions

In this paper, the special simulation approach developed to simulating the batch processing machine model. It used to analyze the performance of four heuristics. The ERT-LPT method significantly outperformed the other methods for all test

problems. The four heuristics presented in this paper were also tested in a deterministic model, and the results obtained were similar to the stochastic model in the sense that ERT-LPT significantly outperformed the other methods. It is worth noting here that this procedure is first strategy of verification.

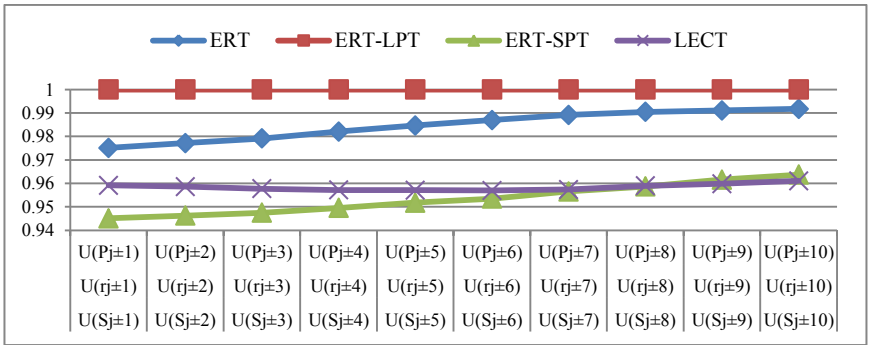

Fig. 1.5 the sensitivity of interval of uniform distribution for all stochastic times

1.8 References

Banks, J., Carson, J., Nelson, B. L., and Nicol, D. (2004). *Discrete-Event System Simulation*, 4th edition, Prentice Hall.

Chandru, V., Lee, C., and Uzsoy, R. (1993). Minimizing total completion time on batch processing machine. *International Journal of Production Research*, vol. 31, pp. 2097-2121.

Chang, P., Damodaran, P., and Melouk, S. (2004). Minimizing makespan on parallel batch processing machines. *International Journal of Production Research*, vol. 42, pp. 4211–4220.

Cheng, B., Yang, S., Hu, X., and Chen, B. (2011). Minimizing makespan and total completion time for parallel batch processing machines with non-identical job sizes. *Applied Mathematical Modeling*, pp. 1010-1016.

Damodaran, P., and Velez-Gallego, M. C. (2010). Heuristics for makespan minimization on parallel batch processing machines with unequal job ready times. *The International Journal of Advanced Manufacturing Technology*, vol. 49, pp. 1119-1128.

Graham, R., Lawler, E., Lenstra, J., and Rinnooy Kan, A. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, vol. 5, pp. 287–326.

Kashan, A., Karimi, B., and Jenabi, M. (2008). A hybrid genetic heuristic for scheduling parallel batch processing machines with arbitrary job sizes. *Computers & Operations Research*, vol. 35, pp. 1084–1098.

Kelton, D., Sadowski, R., and Sturrock, D. (2004). *Simulation with Arena*, 3rd edition, New York: McGraw-Hill.

Pidd, M. (2004). *Computer Simulation in Management Science*, 5th edition, Wiley.

Stewart, R. (2004). *Simulation: The practice of model development and use.* Wiley.

Weng, M., Lu, J., and Ren, H. (2001). Unrelated parallel machine scheduling with setup consideration and a total weighted completion time objective. *International Journal of Production Economics, 70*(3), 215-226.