1999年度日本オペレーションズ・リサーチ学会
秋季研究発表会

SCHEDULING ALGORITHMS AND COMPUTATIONAL EXPERIMENTS FOR MULTIPLE BATCH PROCESSORS

M. Mathirajan, Vijay Chandru and K.N. Krishnaswamy

Department of Management Studies
Indian Institute of Science
Bangalore 560 012
Email: msdmathi@mgmt.iisc.ernet.in

EXTENDED ABSTRACT

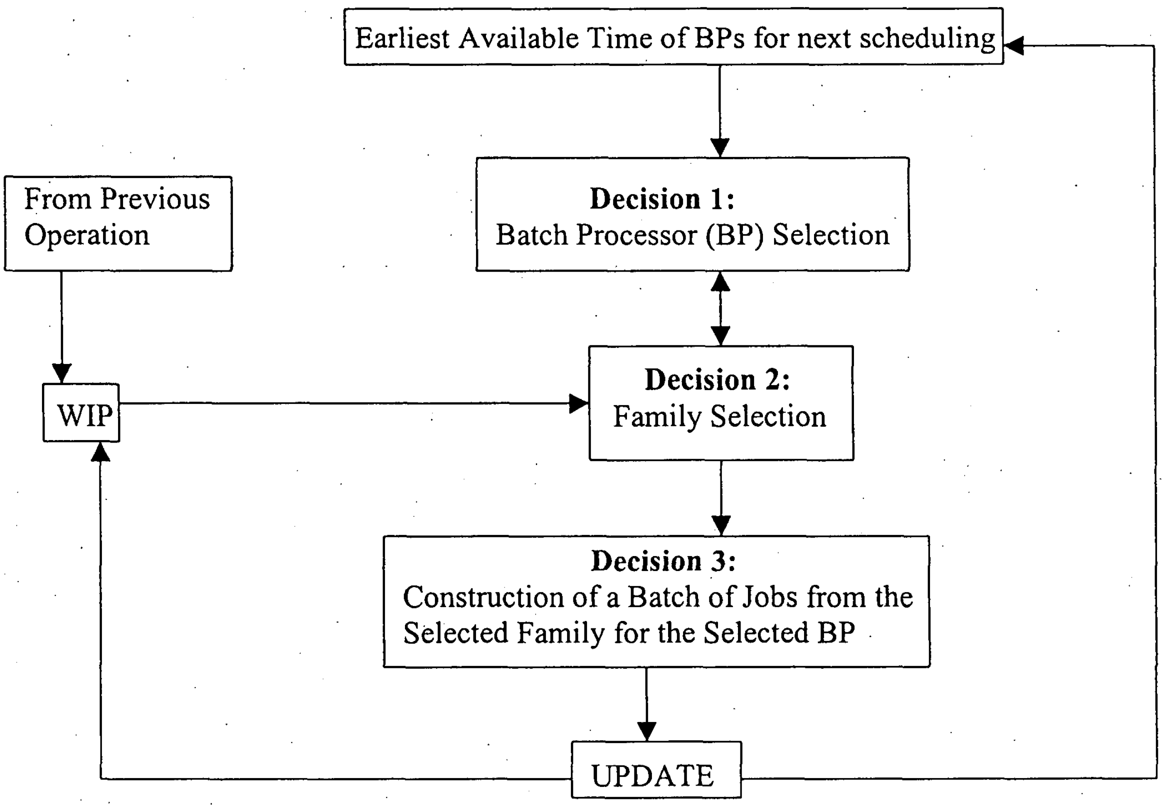
Batch processor (BP) scheduling is a complex problem encountered in industries where a batch processor is a part of the manufacturing line. Batch processors include such facilities as (a) burn-in ovens in the final testing stage of semiconductor manufacturing, (b) diffusion or oxidation tubes stages in semiconductor wafer fabrication, (c) heat-treatment operations in steel and ceramic industries and (d) melting furnace in foundries. The scheduling of batch processors has received attention in scheduling research only recently.

In this paper we address a deterministic problem of scheduling [a PNIBP-INCJF-NIJS] a **P**arallel and **N**on-**I**dentical **B**atch **P**rocessor (PNIBP) with **INC**ompatible **J**ob **F**amiliess (INCJF)* along with **N**on-**I**dentical **J**ob **S**izes (NIJS). The problem addressed in this paper is an extension of the problems dealt by Uzsoy (1995), Van Der Zee et al. (1997) and Mehta and Uzsoy (1998), where all batch processors are identical and all jobs are identical in size. The problem described in this paper was identified at the heat-treatment operation in the post-casting stage of a steel foundry production system.

The problem of scheduling any PNIBP-INCJF-NIJS has three inter-related sets of decisions at time “t’, which is schematically represented in Figure 1.

* Due to the chemical nature of the process, it is not possible to process jobs with different recipes together in the same batch. Thus all jobs requiring the same recipe can be viewed as a jab family, where all jabs of the same family require the same processing time. We shall call these job families incompatible since jobs of different families cannot be processed together.

Figure 1 : Scheduling Problem of PNIBP–INCJF-NIJS


Earliest Available Time of BPs for next scheduling
From Previous Operation
Decision 1: Batch Processor (BP) Selection
Decision 2: Family Selection
WIP
Decision 3: Construction of a Batch of Jobs from the Selected Family for the Selected BP
UPDATE

From our earlier research on this problem (Mathirajan et al. (1998a, 1998b)), it followed that the decision on 'family selection' is of central importance in efficient scheduling of PNIBP-INCJF-NIJS. So, the main goal of this research is to develop the best 'family-selection' heuristics. With this premise, the complexity of the problem is discussed first and then the development and testing of four heuristic algorithms for scheduling a PNIBP-INCJF-NIJS are dealt with in this paper.

The proposed heuristic algorithms, in brief, are as follows: (a) The heuristics for 'batch processor selection' is based on the "capacity" of batch processors available; (b) the heuristics for 'family selection' is based on computing an index (family-wise) and selecting the family which has a minimum index [the index is computed using processing time of a family, number of jobs in the family, weight of each job and the priority of each job in the family]; and (c) the heuristics for 'construction of a batch' is based on selecting a set of feasible jobs from the top of a sorted-list of jobs [sorting the list of jobs available in the selected family is primarily based on priority of a job in ascending order and within the priority, is based on "weight" of a job in descending order].

The analysis presented here assumes a fixed heuristics for BP selection and for batch construction. We carry out a comparision of four heuritstics for "family selection". These four heuristics are based on (1) the weighted average priority of a job; (2) the weighted average weight of a job; (3) the simple average priority of a job; and (4) the simple average weight of a job, respectively.

All algorithms were implemented and tested on a 200 MHz Pentium-based PC. Programs were written in Turbo C++ to implement all the four algorithms; and also for the procedure proposed for generating problem instances randomly.

The four heuristic algorithms were compared among themselves in order to pick the best solution for a given problem instance. Note that it is difficult to obtain an exact solution for these batch scheduling problems within the reasonable compuational time since the problem is intractable as shown in [Mathirajan et al. (1998a)]. For each algorithm, a separate computational experiment was carried out using 100 randomly generated problem instances to test their relative performance. The analysis of relative performance of the heuristic algorithms is based on three-performance measures: (1) Overall Flow Time [OUT], (2) Average Utilization of Batch Processors [AUBP} and (3) Average Waiting Time Per Job [AWTPJ]. The results are shown below in Table 1.

Table 1: Performance measure wise ranking of the heuristic algorithms

Heuristic Algorithm [HA]	Ranking of the Heuristic Algorithms		
	Overall Flow Time [OFT]	Average Utilization of Batch-Processors [AUBP]	Average Waiting Time Per Job [AWTPJ]
I	3 [23]	3 [08]	1 [65]
II	4 [16]	4 [03]	2 [35]
III	2 [28]	2 [24]	3 [00]
IV	1 [50]	1 [65]	3 [00]

Note: Values given in [ ] indicates the number of times the algorithm resulted best-result out of 100 problem instances)

The "Best case" and the "Worst case" analyses of the results of the experiments shown in Table 1 indicates that the heuristic algorithm IV (with the 'family selection' criterion based on "the simple average weight of a job") is likely to perform better with respect to the performance measures: OFT and AUBP and the heuristic algorithm I (with the 'family selection' criterion based on "the weighted average priority of a job") is likely to perform better in terms of the performance measure: AWTPJ.

From the above we conclude that the heuristic algorithm IV will be preferred if the requirements of both customer and the producer are primary. But, if the management prefers greater control over WIP, the heuristic algorithm I may be more suitable.

In Table 1. the performance measures are treated separately and the heuristic algorithms are ranked accordingly. We also ran a comparative test based on a composite index as a single measure. The composite-index is based on different weighted combination of the performance measures considered in Table 1. The results of this analysis are shown below in Table 2.

Table 2: Best Algorithm based on Combined-Index as a Performance Measure

Weightage in % for Performance Measures OFT:ABPU:AWTPJ	Composite-Index as a performance measure for HA:				BEST Algorithm
	I	II	IIII	IV	
10 : 20 : 70	50.9	28.0	08.0	16.5	I
10 : 80 : 10	25.7	16.6	24.8	46.5	IV
10 : 30 : 60	46.7	26.1	10.8	21.5	I
40 : 10 : 50	38.0	20.3	12.4	31.0	I
70 : 20 : 10	16.7	08.8	22.4	55.5	IV
60 : 10 : 30	26.6	13.9	17.2	44.0	IV
60 : 20 : 20	22.4	12.0	20.0	49.0	IV
20 : 40 : 40	36.8	21.0	16.0	33.0	I
30 : 10 : 60	43.7	23.5	10.0	24.5	I

The results reported in Table 2 are consistent with those of Table 1. Thus the preferred choice of heuristic algorithms is confirmed by the use of composite-index. That is, for scheduling a **P**arallel and **N**on-**I**dentical **B**atch **P**rocessor with **INC**ompatible **J**ob of **F**amilies along with **N**on-**I**dentical **J**ob **S**izes (PNIBP-INCJF-NIJS), the heuristic algorithm IV will be preferred if the requirements of both customer and the producer are primary. But, if the management prefers greater control over WIP, the heuristic algorithm I may be more suitable.

We would like to add that this is a part of the on-going research consultancy with a large steel casting foundry in South India. The scheduling algorithms that are being developed will be employed by the foundry in scheduling batch processors (melting and heat-treatment furnaces) of castings.

REFERENCES

1. **Mathirajan, M., Vijay Chandru, and Krishnaswamy, K.N. (1998a).** Scheduling multiple batch processors in a steel foundry. Presented in the International Conference on Operational Research For a Better Tomorrow, New Delhi, INDIA, 1998.

2. **M.Mathirajan, Vijay Chandru, K.N.Krishnaswamy and Reha Uzsoy (1998b).** A Schema of Taxonomy and Notation for the Deterministic Batch Processors Scheduling. 31st Annual Convention of ORSI, AGRA, INDIA, December 1998.

3. **Mehta, S.V., and Uzsoy, R. (1998).** Minimizing total tardiness on a batch processing machine with incompatible job families. IIE Transaction, Vol. 30, 165-178.

4. **Uzsoy, R. (1995).** Scheduling batch-processing machine with incompatible job families. International Journal of Production Research, Vol. 33, No. 10, 2685-2708.

5. **Van Der Zee, D.J., Van Harten, A., and Schuur, P.C. (1997).** Dynamic job assignment heuristics for multi-server batch operations – A cost based approach. International Journal of Production Research, Vol. 35, No. 11, 3063-3093.