Fuzzy Optimization and Decision Making, 4, 141–147, 2005.

Single Machine Batch Scheduling Problem with Resource Dependent Setup and Processing Time in the Presence of Fuzzy Due Date

KANTHEN K HARIKRISHNAN harii@tmsk.uitm.edu.my
Department of Operational Research, Faculty of Information Technology and Quantitative Science, University Technology MARA, 40200 Shah Alam, Selangor, Malaysia

HIROAKI ISHII
Graduate School of Information Technology & Physical Sciences, Osaka University, Japan

Abstract. We consider a batch scheduling problem on a single machine which processes jobs with resource dependent setup and processing time in the presence of fuzzy due-dates given as follows:

1. There are n independent non-preemptive and simultaneously available jobs processed on a single machine in batches. Each job j has a processing time and a due-date.
2. All jobs in a batch are completed together upon the completion of the last job in the batch. The batch processing time is equal to the sum of the processing times of its jobs. A common machine setup time is required before the processing of each batch.
3. Both the job processing times and the setup time can be compressed through allocation of a continuously divisible resource. Each job uses the same amount of the resource. Each setup also uses the same amount of the resource.
4. The due-date of each job is flexible. That is, a membership function describing non-decreasing satisfaction degree about completion time of each job is defined.
5. Under above setting, we find an optimal batch sequence and resource values such that the total weighted resource consumption is minimized subject to meeting the job due-dates, and minimal satisfaction degree about each due-date of each job is maximized. But usually we cannot optimize two objectives at a time. So we seek non-dominated pairs i.e. the batch sequence and resource value, after defining dominance between solutions.

A polynomial algorithm is constructed based on linear programming formulations of the corresponding problems.

Keywords: batching, bi-criteria, fuzzy due-date, linear programming, minimum satisfaction degree, resource allocation, scheduling

1. Introduction

In this paper, we introduce a fuzzy constraint about the due-dates of each job that are processed on a single machine in batches. As we know the motivation for batching may instead be the capacity of the machine to process several jobs at once. For example, imagine that jobs must be placed in an oven, for a heat-treat or burn-in operation. The oven has a finite capacity, so that several of the jobs can be processed simultaneously. As in baking cookies, a group of jobs processed together is called a batch, and we call this a batch processing model. Typically, the capacity of the oven is related to the weight, size, or the number of jobs in a batch. Further each customer have their own deadlines which to be followed by the supplier on time. A fuzzy due date reflects on some flexibility in the deadlines of each customer to cookies. Some customers urge the cookie maker to deliver cookies in time, e.g. hotels, air flights, but the other can wait and little delay is permittable.

Processing in all batches require a machine set up time commonly. Both job processing times and setup time can be compressed through allocation of a continuously divisible resource (Cheng et al 2001). Resources such as energy, manpower and utensils used in the above example may reduce or compress the job processing time and set up time. But increase of resource makes the manager unsatisfied economically though it may decrease the maximum completion time. But if we can decrease the maximum completion time greatly by the small additional cost, the decision maker decides to add the resource. Therefore, in this paper, we find an optimal batch sequence and resource values such that the total weighted resource consumption is minimized subject to meeting the job due-dates, for the proposed model. Recently Cheng et al (1998), considered a related bi-criterion problem but it does not deal with batch scheduling. Further fuzzy due date in our model is a generalization of lateness.

Studies on the batching problems have been carried out by many researchers, i.e., (Dobson et al 1987, Naddef and Santos, 1988, Shallcross 1992, Sung and Joo 1997, Potts and Kovalyov 2000, Muthusamy et al 2000). Similarly researches treating the resource dependent job processing and setup time are found in Cheng and Kovalyov (1995), Cheng et al (1998), Cheng et al (2001), Harikrishnan (2001). Section 2 formulates the bi-criteria scheduling problem and Section 3 introduces an idea of a non-dominated pair. Section 4 proposes a polynomial time algorithm to find an optimal batch sequence and resource values such that the total resource consumption is minimized subject to meeting all the due-dates of all jobs and minimal satisfaction degree among due-dates of all jobs is to be maximized. Section 5 summarizes this paper.

2. Problem Formulation

We consider the following bi-criteria scheduling problem.

1. There are n independent non-preemptive and simultaneously available jobs, $\{J_1, J_2, \ldots, J_n\}$ to be processed on a single machine in batches. The machine can handle only one job at a time and cannot process any jobs whilst a set-up is performed. Each job J_j has a processing time p_j and a fuzzy due-date $\tilde{D}_j$. The due-date, $\tilde{D}_j$ of each job is flexible and $\tilde{D}_j$ is represented by the following membership function which reflects the satisfaction degree of due date $\tilde{D}_j$ for each job.

$$\mu_j(C_j) = \begin{cases} 1 & C_j \leq d_j \\ 1 - \frac{C_j - d_j}{t_j} & d_j < C_j < d_j + t_j \\ 0 & d_j + t_j \leq C_j \end{cases}$$

where t_j, d_j are non-negative constants and C_j is the completion time of each job J_j (Han 1994). Processing of all jobs in a batch l is completed together upon the completion of the last job in the batch i.e. if job J_j belongs to the batch l, it should hold

$$C^l \leq d_j + (1 - \alpha_j)t_j$$

where C^l is the batch completion time of batch l and $\alpha_j = \mu_j(C_j)$. Then

$$C^l = sl + \sum_{j=1}^{B_i} p_{\pi(j)},$$

where $B_i = \eta_1 + \eta_2 + \cdots + \eta_l$, η_l is the size of batch i and $\pi(j)$ is the j-th processed job in a schedule π. Note that a common machine setup time s preceding the processing of each batch is assumed.

2. Both the job processing times and the setup time can be compressed through an allocation of a continuously divisible resource. Each job uses the same amount of the resource. Each setup also uses the same amount of the resource. If the set up time is compressed by using an amount of x of a continuously divisible resource to perform the setup, then

$$s = s_{\max} - ax$$

where $s_{\max}$ is the value of the setup time when $x = 0$ and $a > 0$ is the value of the setup time reduction per unit consumption of the resource. Let $s_{\min}$ be the minimum setup time. It is assumed that

$$0 < s_{\min} < s_{\max} \text{ and } 0 \leq x \leq x_{\max} \stackrel{\Delta}{=} \frac{s_{\max} - s_{\min}}{a}.$$

Similarly the job processing times can be compressed if an amount y of the same or another continuously divisible resource is used to process the jobs:

$$p_j = b_j - e_j y, \quad j = 1, 2, \ldots, n$$

where $b_j > 0$, $e_j > 0$, $0 \leq y \leq y_{\max} \leq \min\left\{\frac{b_j - b_{\min}}{e_j} \mid \quad j = 1, 2, ..., n\right\}$, and $b_{\min} > 0$ is a technological restriction which defines the minimum job processing time.

The total resource consumption will be $W = vkx + wy$ where k is the (unknown) number of batches in the schedule, $v \geq 0$ and $w \geq 0$ where v, w are unit compression costs per unit time by using resources. All data are assumed to be integers. The values x and y can be real numbers.

3. There is a polynomial algorithm discussed by Cheng (2001) to find simultaneously the resource values x and y, and a feasible batch sequence with respect to the fixed deadlines d_j i.e. $C^l \leq d_j$ for each job J_j scheduled in the batch l so as to minimize the total resource consumption $W = vkx + wy$ where k is the number of batches in the schedule, $v \geq 0$ and $w \geq 0$ through a linear programming formulation:

Let an optimal batch sequence be expressed as

$$\{1, \ldots, j_1\}, \quad \{j_1 + 1, \ldots, j_2\}, \ldots, \{j_{k-1} + 1, \ldots, n\}.$$

Linear programming problem with two variables x and y:

$$\begin{aligned}
\text{Minimize} \quad & W = vkx + wy \\
\text{subject to} \quad & 0 \leq x \leq x_{\max}, \quad 0 \leq y \leq y_{\max}, \\
& l(s_{\max} - ax) + \sum_{r=1}^{j} (b_r - e_r y) d_i, \quad (l, i, j) \in \mathrm{N},
\end{aligned}$$

where $\mathrm{N} = \{(1, 1, j_1), (2, j_1 + 1, j_2), \ldots, (k, j_{k-1} + 1, n)\}$.
Each triple $(l, i, j) \in N$ satisfies $1 \leq l \leq i \leq j \leq n$, $1 \leq l \leq k \leq n$ and it corresponds to the batch $\{i, i + 1, \ldots, j\}$ sequenced at l th place in the derived optimal batch sequence.

4. As due-dates of each job is flexible in our model, we seek the optimal batch sequence considering two objectives i.e., minimum satisfaction degree $\mu_{\min}$ among the fuzzy due-dates of all jobs to be maximized and the total resource consumption i.e., $W = vkx + wy$ to be minimized.

3. Non-dominated Schedule

Note that without loss of generality, we can set all satisfaction degrees α_j of fuzzy due-dates on all jobs to be same value since $\min_j \alpha_j$ is to be maximized and so feasible solution can be described as (α, x, y) where α is the common satisfaction degree. Usually we cannot optimize both objectives at a time and so we seek some non-dominated solutions defined as follows.

First we define a vector $V^X = (V_1^X, V_2^X)$ with respect to feasible solution $X = (\alpha, x, y)$.

V_1^X indicates the minimal satisfaction degree α and V_2^X is total resource consumption under the optimal batch schedule corresponding to X. Vector $V^{X_1} = (V_1^{X_1}, V_2^{X_1})$ dominates vector $V^{X_2} = (V_1^{X_2}, V_2^{X_2})$ means: $V_1^{X_1} \geq V_1^{X_2}$ and

$V_2^{X_1} \leq V_2^{X_2}$, and at least one inequality holds without equality. Solution X is called non-dominated if there exists no solution that dominates X. Further corresponding pairs (resource values, batching sequence) is called non-dominated pair. From the results in Cheng (2001), if we fix $X = (\alpha, x, y)$, we can obtain the corresponding optimal batch sequence by HL procedure in Cheng (2001). If there exists a feasible batch sequence satisfying the due-dates, we can assume without loss of generality that it processes jobs in non-decreasing order of due-dates (EDD rule) (see Lawler and Moore (1969)).

Procedure HL: Without loss of generality, we assume, jobs are ordered in the increasing order of due dates. The first batch is initiated with the setup time only. At the beginning of iteration j, $j = 1, \ldots, n$, jobs $1, \ldots, j-1$ are assumed to have been assigned in batches. Job j is assigned as follows:

> If the jobs of the last batch can be completed by their deadlines with the addition of j to this batch, then do so. Otherwise, if job j can be completed by its deadline by starting a new batch, then do so. If neither is possible, then no feasible schedule exists.

4. Solution Procedure

If we fix the minimal satisfaction degree α, then each corresponding due-date is $d_j + (1 - \alpha_j)t_j$. Then applying the solution procedure in Cheng (2001), we can obtain the optimal batch schedule which minimizes W. Now let $\bar{\alpha} = 1 - \alpha$. Then $\bar{\alpha}$ denotes maximal dissatisfaction level with respect to the fuzzy due-date. Further let $d_j'(\bar{\alpha}) =^{\Delta} d_j + \bar{\alpha}t_j = (d_j + (1 - \alpha)t_j)$. Then $d_j(\bar{\alpha})$ is a parameterized due-date of each job J_j. An order of these parameterized due-dates changes as $\bar{\alpha}$ increases in the interval [0,1]. By changing the job index if necessary, we assume that $d_i \leq d_j$ for $i < j$ without loss of generality. When $d_i = d_j$ we assume that $t_i \leq t_j$ for $i < j$.

We seek the pair wise order changing point $\bar{\alpha}_{ij}$ with respect to $d_i(\bar{\alpha})$ and $d_j(\bar{\alpha})$ for $i < j$.

Then

$$\bar{\alpha}_{ij} = \begin{cases} \frac{d_j - d_i}{t_i - t_j} & \text{if} \quad d_j - d_i < t_i - t_j \\ 1 & \text{if} \quad t_i - t_j \leq d_j - d_i \end{cases}$$

where $\bar{\alpha}_{ij}$ corresponds to the intersection of the piecewise linear functions which describe the satisfaction degrees for jobs J_i and J_j (if such an intersection exists), otherwise $\bar{\alpha}_{ij}$ is set to a default value 1.

Now we sort $\bar{\alpha}_{ij}$ and let the result be

$$0 = \bar{\alpha}_o < \bar{\alpha}_1 < \bar{\alpha}_2 < \cdots < \bar{\alpha}_q < \bar{\alpha}_{q+1} = 1$$

where q is the number of different $\bar{\alpha}_{ij}$ between 0 and 1. Divide the interval [0, 1] into subintervals, and write $\Delta_i = [\bar{\alpha}_i, \bar{\alpha}_{i+1})$, $i = 0, 1, 2, \ldots, q-1$, $\Delta q = [\bar{\alpha}_q, 1)$. Note that in each interval, order of corresponding due dates is not changed and so processing according to EDD i.e. the order of due dates, is optimal. As certain processing order π_i of jobs always becomes optimal inside each subinterval Δ_i, the feasibility of π_i at $\alpha = (\bar{\alpha}_i + \bar{\alpha}_{i+1})/2$, $i = 1, 2, \ldots, q$ is checked by using the solution procedure suggested by T.C. Edwin Cheng [3]. Since there exists infinitely many non-dominated pairs (non-dominated solutions), we only seek at most one non-dominated pair (non-dominated solution) for each subintervals i.e. at the middle point of each subintervals, we seek at most $(q+1)$ non-dominated pairs (non-dominated solutions). Now we are ready to describe our solution algorithm.

Solution Algorithm

Step 0: Construct subintervals Δ_i, $i = 0, 1, \ldots, q$. Set $u = 0$, and $XS = DS = \phi$.

Step 1: Set $\bar{\alpha} = (\bar{\alpha}_u + \bar{\alpha}_{u+1}/2)$. For due-dates $d_1(\bar{\alpha})$, $d_2(\bar{\alpha}), \ldots, d_n(\bar{\alpha})$, apply the solution procedure in Cheng (2001) and obtain optimal solution $X^u = (\alpha, x^u, y^u)$ and corresponding optimal batch sequence S^u. If there exists no solution in DS dominating X^u, then set $\mathrm{DS} = \mathrm{DS} \cup \{X^u\}$, $XS = XS \cup ((x^u, y^u), S^u)\}$ and go to step 2. Otherwise go to step 2 directly.

Step 2: If $u \neq q$, set $u = u + 1$ and return to step 1. Otherwise terminate. XS gives non-dominated pairs.

Complexity:

Note that $q = O(n^2)$ and therefore, the sorting needs $O(n^2 \log n)$ executions. The calculation of modified due dates is $O(n^2)$. By applying the solution procedure in Cheng (2001) for each modified due -date, above algorithm solves the problem by at most $O(n^9 \log n)$ times as the solutions procedure's complexity is $O(n^7)$. So its complexity is polynomial. Further when applying the solution procedure in Cheng (2001), except the first time, it need not be started from scratch. So if suitably applied, our algorithm can be executed in the same time complexity as the solution procedure in Cheng (2001).

5. Conclusion

Polynomial time algorithm have been presented to solve variants of the single machine batch scheduling problem with setup and job processing times dependent on continuously divisible resource as we can find in Cheng et al (1995), (1998) and (2001). But our model discussed one step further by presenting flexibility in due dates of each job. We have investigated the single machine batching scheduling problem with fuzzy due-date constraints and proposed the polynomial time algorithm to solve the problem. But the algorithm may need some refinement not only in algorithmic sense but also in actual situations. Our approach to fuzzified scheduling models is relatively new. We should endeavor to pursue this direction to other classical batch scheduling models with resource constraints and construct more actual schedules applicable to real situations, such as the optimal operations of processing disposals in the disposal center.

Acknowledgments

This research is supported by JSPS Ronpaku Program No: VCC-10144 and Research Project of Otemongakuin University 2003.

Note

1. Corresponding author. Department of Operational Research, Faculty of Information Technology and Quantitative Science, University Technology MARA, 40200 Shah Alam, Selangor, Malaysia. Tel.: +603-5543-5432.

References

Cheng, T. C. E. and M. Y. Kovalyov. (1995). "Single Machine Batch Scheduling with Deadlines and Resource Dependent Processing Times", *Operations Research Letters* 21, 243–249.

Cheng, T. C. E., Adam Janiak, and M. Y. Kovalyov. (1998). "Bicriterion Single Machine Scheduling with Resource Dependent Processing Times", *SIAM Journal Optimization*, 8 (2), 617–630.

Cheng, T. C. E., Adam Janiak, and M. Y. Kovalyov. (2001). "Single Machine Batch Scheduling with Resource Dependent Setup and Processing Times", *European Journal of Operational research* 135, 177–183.

Dobson, G., U. S. Karmarkar, and J. L. Rummel. (1987). "Batching to Minimize Flow Times on One Machine", *Management Science* 33, 784–799.

Sangsu Han, Hiroaki Ishii, and Susumu Fujii. (1994). "One Machine Scheduling Problem with Fuzzy Due-Dates", *European Journal of Operational Research* 79, 1–12.

Kanthen K. Harikrishnan and H. Ishii. (2001). "Some Resource Constrained Scheduling Problems", *Asia Pacific Management Review* 6, 477–484.

Lawler E. L. and J. M. Moore. (1969). A Functional Equation and its Applications to Resource Allocation and Sequencing Problems", *Management Science* 16, 77–84.

Kanesan Muthusamy, Hiroaki Ishii, Teruo Masuda, and Shintaro Mohri. (2002). "Single Machine Batching Problem to Minimize the Sum o Completion Times with Number of Batches and Batch Size Limitations", *Science Mathematicae Japonicae* 55, 369–381.

Naddef D. and C. Santos. (1988). "One-Pass Batching Algorithms for the One-Machine Problem", *Discrete Applied Mathematics* 21, 133–145.

Potts, C. N and M. Y. Kovalyov. (2000). "Scheduling with Batching: A Review", *European Journal of Operational Research* 120, 228–249.

Shallcross, D. F. (1992). "A Polynomial Algorithm for a One Machine Batching Problem", *Operations Research Letters* 11, 213–218.

Sung C. S. and U. G. Joo. (1997). "Batching to Minimize Weighted Mean Flow Time on a Single Machine with Batch Size Restrictions", *Computers and Industrial Engineering* 32, 333–340.