International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 12
April 2018

Minimize The Batch Delivery In Machine Scheduling

P. Ramanatha Reddy[1,2], David Aluri[3], Dr. B. Yedukondalu[4]

[1]*Research Scholar, K L University, Vaddeswaram, Andhra Pradesh, India*
[2]*Department of Mechanical Engineering, Nalla Malla Reddy Engineering College, JNT University, Hyderabad, India*
[3]*Department of Mechanical Engineering, Nalla Malla Reddy Engineering College, JNT University, Hyderabad, India*
[4]*Associate Dean-Student Affairs, Associate Professor, Dept. of Mechanical Engineering, K L University, Vaddeswaram, Andhra Pradesh*

ABSTRACT

This paper is to Minimize the Batch Delivery in Machine Scheduling, study a problem in which a set of jobs has to be batched as well as scheduled for processing on a single machine. Each delivery batch has a capacity and incurs cost. A constant machine set-up time is required before the first job of each batch is processed. A schedule specifies the sequence of batches, where each batch comprises a sequence of jobs. The batch delivery time is defined as the completion time of the last job in a batch. The earliness of a job is defined as the difference between the delivery time of the batch to which it belongs and the job completion time. The objective is to find a coordinated production and delivery schedule that minimizes the total flow time of jobs plus the total delivery cost and to minimize the sum of the total weighted job earliness and mean batch delivery time.

Keywords: *Scheduling, Make span, Total Weighted Tardiness*

1. Introduction

Scheduling is the process of arranging, controlling and optimizing work and workloads in a production process or manufacturing process. Scheduling is used to allocate plant and machinery resources, plan human resources, plan production processes and purchase materials. It is an important tool for manufacturing and engineering, where it can have a major impact on the productivity of a process. In manufacturing, the purpose of scheduling is to minimize the production time and costs, by telling a production facility when to make, with which staff, and on which equipment. Production scheduling aims to maximize the efficiency of the operation and reduce costs. In some situations, scheduling can involve random attributes, such as random processing times, random due dates, random weights, and stochastic machine breakdowns. In this case, the scheduling problems are referred to as stochastic scheduling. In this section, we introduce the notation to be used in the paper:

2. Notation

n: number of jobs

m: number of machines

$J = \{J_1 \dots J_n\}$: job set to be processed;

P_i : processing time of job J_i;

R : number of batch deliveries;

$\alpha(R)$: delivery cost function, a no decreasing function of R;

$C_{max} = \max \{c_i\}$: makespan of a schedule;

$G(\pi) = \sum_i^n = I F_i + \alpha(R)$: total penalty of π

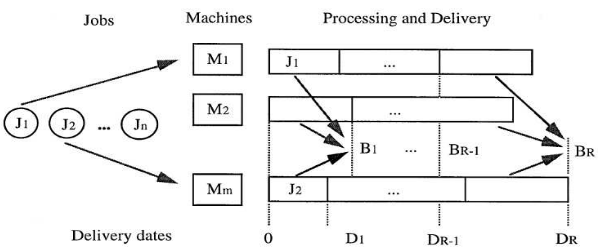

Fig.1. Problem descriptions

B_l : batch l;

b_l : number of jobs in B_l;

$^D l$: delivery date of B_l;

C_i : completion time of job J_i

F_i: flow tome of J_i, which is equal to the batch delivery

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 12
April 2018

date on which J_i is delivered

$\pi =< B_{1,\ldots\ldots} B_R >$: a schedule;

Adopting the three-field notation introduced by Graham et al. [11], we denote our problem as $P_m/bd/(\sum F_i + \alpha(R))$.

3. PROBLEM DEFINITION

NP-COMPLETENESS AND DYNAMIC PROGRAMMING

In this section, we consider the complexity issues of the problem. First, it is interesting to note that, when the delivery cost is negligible, $P_m/bd/(\sum F_i + \alpha(R))$ simply reduces to the classical parallel machine scheduling problem $Pm//\sum C_i$ it is well known that $Pm//\sum C_i$ is solved by the generalized shortest processing time (SPT) rule: schedule the jobs in the order of non decreasing processing times, and assign each job to the earliest available machine [12].

Next, we give a simple proof for the NP-completeness for the problem under study. Assume that the batch delivery cost is so large that all jobs must be delivered in one batch. i.e. R=1. Then $P_m/bd/(\sum F_i + \alpha(R))$ is equivalent to the classical parallel machine scheduling problem Pm//Cmax. Since Pm//Cmax has been shown to be NP-complete in the ordinary sense when m is fixed, and NP-complete in the strong sense when m is arbitrary [13].

Theorem 1. Even when R"1, $P_m/bd/(\sum F_i + \alpha(R))$ is NP-complete in the ordinary sense when m is fixed and m≥2, and NP-complete in the strong sense when m is arbitrary.

The following lemma establishes several properties for an optimal schedule for the problem.

Argument 1 :-. There exists an optimal schedule π^* $= < B_{1,\ldots\ldots} B_R >$ for Pm/bd, R $\leq U/(\sum F_i + \alpha(R))$ in which

i. There is no idle time before each job;

ii. All jobs assigned to the same machine are scheduled in the SPT order;

iii. B_l contains all jobs which finish processing in the time interval $(D_{l-1}, D_l]$, l=……, R.

Proof

- Trivial.
- Assume that jobs J_i and J_j are assigned to the same machine and J_j follows J_i immediately such that $p_i \geq p_j$ in π^*. Let π' be a schedule obtained by swapping J_i and J_j. It is easy to show that $G(\pi^*) \geq G(\pi')$, regardless of whether J_i and J_j are delivered in the same batch or not.
- Let us number all jobs in the order of their completion time in π^*. Assume that the batches are numbered in accordance with the numbers of their last jobs. It is clear that, without loss of generality, we can assume $D_1 <\ldots\ldots< D_R$ Let J_j be the first job in B_R. If there are any jobs between J_j and J_n (the last job in B_R) which are assigned to other batches, then there must be at least one batch, say B_l, such that $C_j \leq D_l < D_R$. Let π' be a schedule obtained by simply assigning J_j to B_l. It is obvious that $G(\pi^*) \geq G(\pi')$, a contradiction. Following the same argument with the jobs in batch B_{R-1} and so on, we can show that there exists an optimal schedule in which all batches consist of a number of jobs which finish processing contiguously. Since all jobs which processing at D_l, l=1…… R, can all be assigned to B_l in any optimal schedule, we have shown that B_l contains all jobs which finish processing in the time interval $(D_{l-1}, D_l]$. h

Based on Lemma 1, we can develop a dynamic programming algorithm to solve the problem. Let; **U** be an upper bound for the number of batch deliveries, and P $= \sum^n_{i=1} p_i$. The algorithm is formally described as follows.

Algorithm -1:

a) Renumber the jobs in the SPT order i.e. $p_1 \leq p_2 \ldots . \leq p_n$.

b) Define H_R (j, $t_{1,\ldots\ldots\ldots}$,tm,$D_{1,\ldots\ldots\ldots}$,D_R) as the minimum total flow if we have scheduled jobs J_1 up to J_j such that the total processing time of the jobs assigned to

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 12
April 2018

machine u is t_u u=1……m and the delivery date is D_l for batch B_I I= 1,……m

recursive relation : for j = 0,…….n, t_u = 0,….,P , u = 1,……,m, $D_t = D_{l-1}+1$,….P, I = 1,….,R, D_0 = 0 and R = 1,….,U,

c) H_R (j,t_1,…….,t_m,D_1,…..,D_R) = min {X_u}, → **(1)**

Where

i. X_u = H_R (j-1,t_1,…….,t_u-p_j,…..t_m,D_1,…..,D_R) +F_j → **(2)**

ii. F_j = {D_t /$D_{t-1} < t_u \le D_t$, D_0 = 0, l = 1,…….R}. →**(3)**

d) (initial conditions: for each t_k =0,…..P, u=1,…m, D_t =D_{l-1}+1,…..P, I=1,…,R, D_0 = 0, and R = 1,….,U, H_R (j,t_1,…..,t_m, D_1,…..,D_R)
{If j=0, t_1=t_2 =…….t_m =0 ∞ otherwise

e) Optimal solutions: G^* = min { H_R (n,t_1,…..,t_m, D_1,…..,D_R) + α(R)} over all t_u = 0,….,P,R, D_0 = 0 and R = 1,…U.

Argument 2: Algorithm -1 solves the problem Pm/bd, R ≤ U/ ∑ (F_i + α(R)) in O(nm;U^2 P^m+u-1) time.

Proof: Due to Lemma 1, there exists an optimal schedule with jobs assigned to each machine in the SPT order, and each job in assigned to the first batch after the completion of the job. If J_j is assigned to machine u, then C_j =t_u, and if $D_{l-1} < t_u \le D_l$, then J_j is assigned to B_l , and so F_j = D_l . This justifies Eqs. (2) and (3). Since H_R (j, t_1,…….t_m,D_1…..D_R) is determined by the minimum assignment by definition, we have justified the validity of the recursive relations. So the algorithm PMBD-1 solves the problem Pm/bd, R≤ U/ (∑F_i + α (R)).

The time complexity of the algorithm can be established as follows. Since only m-1 of the values t_1,….t_m are independent, the number of different states of the recursive relations is at most n P^{m+u-1} for R =1,….U For each state, the right hand side of Eq. (1) can be calculated in O(mU) time. Thus, the overall computational complexity of Algorithm PMBD-1 is O(nm;U^2 P^{m+U-1}).

Argument 2 implies that the problem Pm/bd, R≤ U/ (∑F_i + α (R)). is not strongly NP-complete for any constant m and U ;. But it is not clear whether Pm/bd, R≤ U/ (∑F_i + α (R) is strongly NP-complete or pseudo polynomially solvable for a constant m and an arbitrary U.

POLYNOMIALLY SOLVABLE CASES

In this section, we first consider a special case where the job assignment is predetermined. It is evident that the problem reduces to an optimal batching problem in this case. This special case characterizes the practical scenario where each machine is dedicated to a special group of jobs. According to argument 1, we can provide a backward dynamic programming algorithm to solve the optimal batching problem as follows.

(a) Schedule the jobs on each machine in the SPT order, and then renumber all the jobs in accordance with the job completion times.

(b) Define H_R (j, l) as the minimum total completion time of the jobs J_j,…….J_n when they are assigned to the delivery batches B_1,……B_R.

(c) Recursive relations: For R = 1,………,n, I = R,…..1, and j = n,… l, H_R (j,l) = min { H_R (k, I+1) + (k-j) C_{k-1} }
→**(4)**

(d) Initial conditions: for each j = 1,…..,n l= 1,…..R, and R = 1,…..n,
$H_{R(J,L)}${if j= n+1 and I = R+1; ∞ otherwise

(e) Optimal solutions: G(π^{*}) = min { H_R(I,1) + α(R)} 1 ≤ R ≤ n

While the optimality of the algorithm can be easily justified, it is also not difficult to see that the time complexity of the algorithm is O (n4).

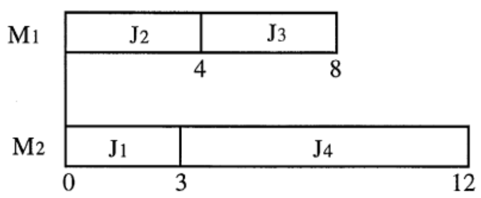

Fig.2. Example 1

Now we present a numerical example for the special case to demonstrate the optimality of the algorithm.

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 12
April 2018

Example 1: Consider the instance with J = {$J_{1,\ldots\ldots}$ J_4}, m=2, and α(R) = 7R. Assume that J_2 and J_3 are assigned on M_1, J_1 and J_4 are assigned on M_2, and all jobs are sequenced in the SPT order on each machine (as shown in Fig. 2). Now using algorithm -2 to solve the instance, we have the following results:

When R = 1, we have

H_1 (4, 1) = H_1(5, 2) + C_4 = 12,
H_1 (3, 1) = H_1(5, 2) + 2C_4 = 24,
H_1 (2, 1) = H_1(5, 2) +3C_4 = 36,
H_1 (1, 1)=H_1(5, 2) + 4C_4 = 48, and so
And G(π_4^*) =7R +H_4 (1, 1) = 55.

When R=2, we have

H_2 (4, 2) = H_2(5, 2) + C_4 = 12,
H_2 (3, 2) = H_2(5, 3) + 2C_4 = 24,
H_2 (2, 2) = H_2(5, 3) +3C_4 = 36,
H_2 (1, 1) = min {H_2(4, 2) + 3C3,
H_2 (3, 2) + 2 C_2 , H_2(2, 2) +C_3} =32
And G(π_2^* = 7R + H_1 (1, 1) = 46.

When R = 3, we have

H_3 (4, 3) = H_3(5, 4) + C_4 = 12,
H_3 (3, 3) = H_3(5, 4) + 2C_4 = 24,
H_3 (2, 2) = min {H_3(4, 3) +2C_3
H_2 (3, 3) = H_2(3, 3) + C_2} = 28,
H_3 (1, 1) = H_3(2, 2) + C_1 = 31.

Thus
G(π_3^*) =7R +H_3 (1, 1) + C_1 = 52.

When R=4, we have
H_4 (4, 4) = H_4(5, 5) + C_4 = 12,
H_4 (3, 3) = H_4(4, 4) + C_3 = 20,
H_4 (2, 2) = H_4(3, 3) +C_4 = 24
H_4 (1, 1) = H_4(2, 2) + C_1 = 27,
And so
G(π_1^*) =7R +H_1 (1, 1) = 55.
Hence, we obtained an optimal schedule π^* = < B_1,B_2 > with B_1 = {J_1, J_2 } and B_2 = {J_3,J_4 }

We now consider the special case with identical processing times, Pm/bd, P_i = P / (∑F_i + α (R)). Let n_o = [n/m], g = n_o m – n. let n_u be the number of jobs processed on machine u under a specific schedule. Then we have the following lemma.

Argument 3: There exist an optimal schedule for Pm/bd,P_i /(∑ F_i + α(R)) in which n_o -1 ≤ n_u , ≤ n_0, u = 1,….m**.**

Proof: Suppose there exists an optimal schedule π^* in which the condition is not satisfied. According to Lemma 1, we can assume that there is no inserted idle time in π^*. Then there must be a pair of machines u and v such that $n_u \geq n_v$ +2 and the last job on machine u is also the last job of the last batch delivery. It is clear that moving the last job on machine u to the last position on machine v will not increase the total penalty. Repeating this process, we can obtain a desired optimal schedule.

Let π^* = < $B_{1\ldots\ldots\ldots}$,B_R > be an optimal schedule with R batch deliveries for the problem **P_i /(∑ F_i + α(R))** Let b_l = b_l / m, l = 1,……..R-1. We have

Argument 4: There exists an optimal schedule with R batch deliveries for Pm/bd, P_i = p /(∑ F_i + α(R)) in which b_l , l = 1,……..R-1 is integral

Proof: According to Lemma 1, we can assume that in π^* there is no inserted idle time and B_l contains all jobs which finish processing in the time interval (D_{l-1},D_l], l=1…….,R-1. It is evident that (D_l.D_{l-1})/p is integral, l=1……..,R. Since $b_{l;=}$ (D_l - D_{l-1})m/p, l=1,…….,R-1, b_l is also integral.

Now assume that π_R^* satisfied arguments 3 and 4. Let b_l = [b_l/m]. We have

Argument 5: There exists an optimal schedule with R batch deliveries for **Pm/bd, P_i = p /(∑ F_i + α(R))** in which [b_k – b_l] ≤ 1 for any pair of k and I, where k = 1,……,R, I = 1,…………,R.

Proof: We first show that changing the sequence of B_l, I= 1,….,R-1, will not cause any increase in the total penalty. It is not difficult to see that the sequencing of batch deliveries B_l, I = 1,…….,R-1, is equivalent to the classical single machine total weighted flow time scheduling problem, denoted as 1// ∑ w_l c_l , with w_l = b_l

m and $p_l = b_l$ p. It is well known that the total weighted flow time is minimized by sequencing the jobs in the weighted shortest processing time (WSPT) order [14]. Since

$P_1 / w_1 = \ldots = p_{R-1} / w_{R-1} = p/m$, B_i , I=1,………, R -1, can be sequenced in an arbitrary order. Now, we can prove the lemma by showing

$[b_{l-1} - b_l] \leq 1$ I = 2,……R → **(5)**

We assume that $b_l > 1, l = 2, \ldots\ldots R$. . By definition, we have

$D_{l-1}\, b_{l-1} + D_l\, b_l \leq (D_{l-1} - p)(b_{l-1} - m) + (D_l + p)(b_l + m)$, → **(6)**

$D_{l-1}\, b_{l-1} + D_l\, b_l \leq (D_{l-1} + p)(b_{l-1} + m) + (D_l - p)(b_l - m)$. → **(7)**

Then we can easily obtain the desired results.

Now, assume that there are some batches such that b_l =1. Note that b_l may be less than m in this case. Since it is trivial when $b_{l-1} = 1$ (or $b_{l+1} = 1$), we suppose $b_{l-1} > 1$ (or b_{l-1} = 1). From (6) or (7), we can easily show that $b_{l-1} \leq 2$ or ($b_{l-1} \leq 2$ or ($b_{l+1} \leq 2$), and thus (5) holds again.

This completes the proof

Based on these results, we can easily construct an optimal schedule with R batch deliveries

$\pi^* = < B_1, \ldots, B_R >$ such that

where $b_0 = [\, n_0 / R\,]$ and $h = b_o R - n_o$. The associated total penalty can be calculated as

$G(\pi_R^*) = \alpha(R) + \sum \quad {}_{-}1)^2 + h\,(b_o - 1) p\, b_o\,(R - h)\, m + \sum \quad {}_0 p$

$= \alpha(R) + (Rb_o - h + b_o)(Rb_0 - h)\, mp/2 - h\,(b_o - 1) mp/2 - g n_0\, p$

Now, we can construct a simple algorithm to solve the problem as follows.

Algorithm -3:

1. $n_o := [n/m]$: $g := n_{om} - n$; $R^* = 0$; $G^* := \infty$
2. for R =1 to n_0 do being $b_o := [\, n_o / R]$; $h := b_0 R - n_0$;

$G(\pi_R^*) = \alpha(R) + (Rb_o - h + b_o)(Rb_o - h) mp/2 - h(b_o - 1) mp/2 - g n_0\, p$:

if $G^* > G(\pi_R^*)$ then $R^* := R$; $G^* := G(\pi_R^*)$;

It is clear that Algorithm-3 solves the problem Pm/bd, $P_i = p / (\sum F_i + \alpha(R))$ in $O(n/m)$ time. It should be pointed out that, the algorithm, although efficient, is not actually polynomial when each batch delivery has an equal delivery cost c, i.e. $\alpha(R) = cR$.

The following numerical example demonstrates the optimality of the algorithm.

Example 2: Consider the instance with $J = \{J_1, \ldots\ldots, J_7\}$ $m = 2$, $\alpha(R) = 7R$, $P_i = P = 2$ i = 1,…….7 From Lemma 3, we know that there exists an optimal schedule in which all jobs are sequenced as shown in Fig. 3. It is clear that $n_o = 4$, $g = 1$. Using algorithm PMBD-3 to solve the instance, we have the following results.

When $R = 1$, we have $b_o = 4$ and $h = 0$, and so $G(\pi^*_1) = 7R + 28p = 63$.

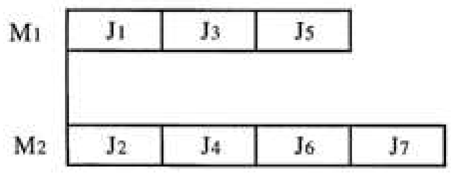

Fig.3. Example 2

When $R = 2$, we have $b_o = 2$ and $h = 0$, and so $G(\pi_2^*) = 7R + 20p = 54$.

When $R = 3$, we have $b_o = 2$ and $h = 2$, and so $G(\pi_3^*) = 7R + 18p = 57$.

When $R = 4$, we have $b_o = 1$ and $h = 0$, and so $G(\pi_4^*) = 7R + 16p = 60$.

Now we can obtain an optimal schedule

$\Pi^* = < B_1, B_2 >$ with $B_1 = \{ J_1, J_2, J_3, J_4 \}$ and $B_2 = \{ J_5, J_6, J_7 \}$

4. Conclusion

In this paper, the parallel machine scheduling with batch delivery costs have been studied It is shown that the problem to minimize the sum of the total flow time and delivery cost is NP-complete in the strong sense. A dynamic programming algorithm is then provided to solve the problem. The algorithm is pseudo polynomial

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 12
April 2018

when the number of machines is constant and the number of batches has a fixed upper bound. A two polynomial time algorithms to solve the special cases where the job assignment is given or the job processing times are equal is provided. There are a number of issues which are of interest for further research. First, it is interesting to investigate the open problem posed in the paper, i.e. whether it is pseudopolynomially solvable or strongly NP-complete when the number of machines is constant and the number of batches is arbitrary. It is also interesting to investigate polynomial time algorithms for the special case where the job processing times are equal and the batch delivery cost function is linear. Another interesting issue is to develop elective heuristics to solve the general problem, and it is evident that a viable strategy is to combine the *list scheduling* procedure for the classical parallel machine scheduling problems [15] with the optimal batching algorithm proposed in this paper.

REFERENCES

1. S. Webster, K.R. Baker, Scheduling groups of jobs on a single machine, Operations Research 43 (1995) 692-703.

2. M.M. Liaee, H. Emmons, Scheduling families of jobs with setup times, International Journal of Production Economics 51 (1997) 165-176.

3. T.C.E. Cheng, J.N.D. Gupta, G. Wang, A review of #owshop scheduling research with setup times, Production and Operations Management (1999), to appear.

4. T.C.E. Cheng, H.G. Kahlbacher, Scheduling with delivery and earliness penalties, Asia-Paci"c Journal of Operational Research 10 (1993) 145-152.

5. T.C.E. Cheng, V.S. Gordon, Batch delivery scheduling on a single machine, Journal of the Operational Research Society 45 (1994) 1211-1215.

6. T.C.E. Cheng, V.S. Gordon, M.Y. Kovalyov, Single machine scheduling with batch delivery, European Journal of Operational Research 94 (1996) 277-283.

7. T.C.E. Cheng, M.Y. Kovalyov, B.M.T. Lin, Single machine scheduling to miHenimize batch delivery and job earliness penalties, SIAM Journal on Optimization 7 (1997) 547-559.

8. J.W. Hermann, C.-Y. Lee, On scheduling to minimize earliness-tardiness and batch delivery costs with a common due date, European Journal of Operational Research 70 (1993) 272-288.

9. Z.-L. Chen, Scheduling and common due date assignment with earliness-tardiness penalties and batch delivery costs, European Journal of Operational Research 93 (1996) 49-60.

10. J. Yuan, A note on the complexity of single-machine scheduling with a common due date, earliness-tardiness, and batch delivery costs, European Journal of Operational Research 94 (1996) 203-205.

11. .L. Graham, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: A survey, Annals of Discrete Mathematics 5 (1979) 287-326.

12. R.W. Conway, W.L. Maxwell, L.W. Miller, Theory of Scheduling, Addison-Wesley, Reading, MA, 1966.

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

14. W.E. Smith, Various optimizers for single-stage production, Naval Research Logistics Quarterly 3 (1956) 59-66.

15. R.L. Graham, Bounds for certain multiprocessing anomalies, Bell System Technical Journal 45 (1996) 1563-1583.

About the authors:

P. Ramanatha Reddy is presently a research scholar at K L University and he is working as Assistant Professor at Nalla Malla Reddy Engineering College, Hyderabad, India.

David Aluri got his M.E(Automation & Robotics) from Osmania University, Hyderabad and is presently working as Assistant Professor at Nalla Malla Reddy Engineering College, Hyderabad, India.

3 Dr. B. Yedukondalu is presently Associate Dean-Student Affairs, hostels and Associate Professor, Dept. of Mechanical Engineering, Robotics & Mechatronics Research Group Head.