ANZIAM J. 47(2005), 265–275

FLOWSHOP/NO-IDLE SCHEDULING TO MINIMISE THE MEAN FLOWTIME

LAXMI NARAIN[✉ 1] and P. C. BAGGA[2]

(Received 27 June, 2003; revised 18 April, 2004)

Abstract

This paper deals with n-job, 2-machine flowshop/mean flowtime scheduling problems working under a "no-idle" constraint, that is, when machines work continuously without idle intervals. A branch and bound technique has been developed to solve the problem.

1. Introduction

The nonpreemptive, flowshop sum of the completion times scheduling problem is that of scheduling n jobs to be processed by 2 machines. Job i, $i = 1, 2, \dots, n$, consists of at most 2 operations $(O_{i,1}, O_{i,2})$. Operation $O_{i,j}$, which precedes $O_{i,j+1}$, has to be processed uninterrupted for $p_{i,j}$ time units on machine M_j, $j = 1, 2$. Here $p_{i,j}$ is a nonnegative integer, where $p_{i,j} = 0$ if $O_{i,j}$ is zero (that is, has zero processing time) and positive otherwise. Two operations of the same job cannot be processed simultaneously and a machine may process at most one job at a time. The problem is to find the operation sequence on each machine that obeys the problem constraints and minimises the sum of completion times. The problem is designated $n/2/F/\sum C_i$, where F stands for flowshop discipline and C_i is the completion time of job i. (For notation and classification of scheduling problems we follow Rinnooy-Kan [15].)

Minimisation of elapsed time and minimisation of mean completion time have been extensively studied in the literature [1–17]. Conway, Maxwell and Miller [6] gave a fundamental theorem for flowshop scheduling that states that an optimal schedule exists for an $n/m/F/\delta$ problem (where δ is any regular measure of performance) with the same processing order on the first two machines [6, page 81]. Ignall and Schrage [9] solved an $n/2/F/\sum C_i$ problem, Szwarc [16] discussed the problem minimising

[1] Department of Mathematics, University of Delhi, Delhi, India; e-mail: laxmi_narain_2004@yahoo.com.

[2] First line of address, Second line of address, etc.

https://doi.org/10.1017/S1446181100010026 Published online by Cambridge University Press

the mean flowtime when the processing times of jobs on the machines satisfy certain conditions.

Adiri and Pohoryles [1] studied flowshop/sum of completion times scheduling problems, working under a "no-wait" or "no-idle" constraint for dominating machines. The no-wait constraint prescribes that the jobs work continuously without waiting times between consecutive machines, that is, a job needs to go through a sequence of machines without any delay between successive operations of the job. This means the difference between the completion time of a job's last operation and the starting time of its first operation is equal to the sum of its operation times on all machines. The no-wait problem has been extensively studied in the scheduling literature [1–5, 8, 14, 17]. Adiri and Pohoryles [1], Rajendran and Chaudhuri [14], and van der Veen and van Dal [17] studied the no-wait problem with the objective of minimising total flowtime. Adiri and Pohoryles [1], and van der Veen and van Dal [17] provided polynomial time algorithms for special no-wait flowshops while Rajendran and Chaudhuri [14] provided two heuristic algorithms for generic no-wait flowshops.

Aldowaisan and Allahverdi ([2] and [3]) studied the problem of minimising total flowtime with simultaneous consideration of no-wait and separate setup times. They also developed optimal solutions for special flowshops and a heuristic solution for the general two-machine flowshop problem for sequence-independent setup times. In [3] they developed optimal solutions for two special flowshops and a dominance relation for the general two-machine flowshop problem for sequence-dependent setup times.

Under the no-idle constraint, machines work continuously without idle intervals, that is, machines should not remain idle once they start processing the first job. The no-idle constraint arises in real-life situations, when machines may be rented to complete an assignment. Minimisation of the total rental cost of machines would be the criterion in these types of situations. The total rental cost of the machines will be at a minimum when idle times on the machines are also at a minimum. Under the no-idle constraint each machine is required to be taken on rent for the time equal to the sum of the processing times of all the jobs on it. Therefore, the total rental cost of the machines will be minimised under the no-idle constraint.

Adiri and Pohoryles [1] studied special no-idle flowshop problems with the objective of minimising total flowtime and of providing polynomial bounded algorithms. Narain [12] studied the flowshop problem with the objective being minimum total rental cost. The total rental cost is minimised when idle time on all the machines is zero. He provided an algorithm to find the earliest time at which a machine should be taken on rent (or when it should start processing the first job) such that the machine works continuously without any idle intervals until the last job of the schedule is completed on it. Narain and Bagga [13] provided an optimal solution for a three-machine flowshop problem under a no-idle constraint with the objective being minimum total elapsed time.

https://doi.org/10.1017/S1446181100010026 Published online by Cambridge University Press

The present paper deals with general n-job, 2-machine flowshop problems under a no-idle constraint with the objective function being the minimum sum of the completion times of all the jobs. We assume that a set of jobs are ready for processing at time zero, and hence total (mean) completion time and total (mean) flowtime are equivalent. A branch and bound technique has been developed to solve the problem. A numerical example is also given to demonstrate our technique.

2. Notation

Here M_j denotes machine j, $j = 1, 2$, $p_{i,j}$ represents the processing time of job i on machine M_j, $Z_{i,j}$ is the completion time of job i on machine M_j and $I_{i,j}$ is the idle time of machine M_j for job i.

We define $H_j(J_r)$ to be the earliest time at which machine M_j should start processing the first job of schedule J_r such that machine M_j works continuously without a break until the last job is completed on it. Here $Z'_{i,j}$ is the completion time of job i on machine M_j when machine M_j starts processing jobs at time $H_j(J_r)$ and $I'_{i,j}$ is the idle time of machine M_j for job i when machine M_j starts processing jobs at time $H_j(J_r)$. The number of jobs $i = 1, 2, \dots, n$.

3. Complexity of flowshop/no-idle/sum of completion time problem

Garey, Johnson and Sethi [7] proved that a 3-partition problem is reducible to an $n/2/F/\sum C_i$ problem. However, as their constructed instance of an $n/2/F/\sum C_i$ problem is also a no-idle problem, they at the same time proved the unary NP-completeness of an $n/2/F/\sum C_i$ problem and an $n/2/F$, no-idle$/\sum C_i$ problem. Moreover, only minor modification of the proof is needed for proving the NP-completeness of these problems where operations with zero processing times are prohibited. Specifically, replacement of zero processing times on machine M_1 by operations with infinitesimal processing times $\epsilon > 0$ and the shifting of all processes of length ϵ to the beginning of the schedule on machine M_1, proves the unary NP-completeness of $n/2/F$, $p_{i,j}>0/\sum C_i$ and $n/2/F$, $p_{i,j}>0$, no-idle$/\sum C_i$ problems.

4. Theorems

Without loss of generality we can consider the job sequence $J_r = 1, 2, \dots, r$ in the proof of the theorems.

THEOREM 4.1. *If machine M_2 starts processing jobs at time $H_2(J_r) = \sum_{i=1}^{r} I_{i,2}$, then $Z_{r,2}$, the completion time of job r on machine M_2, will remain unaltered.*

https://doi.org/10.1017/S1446181100010026 Published online by Cambridge University Press

PROOF. Let $Z'_{i,2}$ be the completion time of job i on machine M_2 when it starts processing jobs at time $H_2(J_r)$. The proof of this theorem is based on mathematical induction.

For $r = 1$, the completion time of job 1 on machine M_2 when it starts processing jobs at time $H_2(J_1)$,

$$Z'_{1,2} = H_2(J_1) + p_{1,2} = \sum_{i=1}^{1} I_{i,2} + p_{1,2} = p_{1,1} + p_{1,2} = Z_{1,2}.$$

Therefore the result holds for $r = 1$.

Assume that the result folds for $r = m$. For $r = m + 1$,

$$\begin{aligned} Z'_{m+1,2} &= \max\{Z_{m+1,1}, Z'_{m,2}\} + p_{m+1,2} \\ &= \max\{Z_{m+1,1}, Z_{m,2} + I_{m+1,2}\} + p_{m+1,2} \\ &= \max\{Z_{m+1,1}, Z_{m,2} + \max\{Z_{m+1,1} - Z_{m,2}, 0\}\} + p_{m+1,2} \\ &= \max\{Z_{m+1,1} + \max\{Z_{m+1,1}, Z_{m,2}\}\} + p_{m+1,2} \\ &= \max\{Z_{m+1,1}, Z_{m,2}\} + p_{m+1,2} \\ &= Z_{m+1,2}. \end{aligned}$$

Therefore the result holds for $r = m + 1$ also. Hence by mathematical induction this theorem holds for all r, where $r = 1, 2, \dots, n$. □

THEOREM 4.2. *If machine M_2 starts processing jobs at time $H_2(J_r) = \sum_{i=1}^{r} I_{i,2}$, then there is zero idle time on machine M_2 for partial schedule J_r, where $J_r = 1, 2, \dots, r$.*

PROOF. Let $I'_{i,2}$ be the idle time of machine M_2 for job i of schedule J_r when machine M_2 starts processing jobs at time $H_2(J_r)$. Let $Z'_{i,2}$ be the completion time of job i on machine M_2 when it starts processing jobs at time $H_2(J_r)$.

By the method of mathematical induction, we shall show that the idle time of machine M_2 for job $k = 1, 2, \dots, r$ is zero, when it starts processing jobs at time $H_2(J_r)$.

For $k = 1$,

$$\begin{aligned} H_2(J_r) &= \sum_{i=1}^{r} I_{i,2} = I_{1,2} + \sum_{i=2}^{r} I_{i,2} \\ &= Z_{1,1} + \sum_{i=2}^{r} I_{i,2} \geq Z_{1,1}, \quad \text{since} \quad \sum_{i=2}^{r} I_{i,2} \geq 0. \end{aligned} \tag{4.1}$$

From (4.1), $I'_{1,2} = \max\{Z_{1,1} - H_2(J_r), 0\} = 0$. Therefore the result holds for $k = 1$.

https://doi.org/10.1017/S1446181100010026 Published online by Cambridge University Press

Let the result hold for $k = m$. This implies that

$$Z'_{m-1,2} \geq Z_{m,1} \tag{4.2}$$

and $I'_{m,2} = 0$.

For $k = m+1$, $Z'_{m,2} = \max\{Z_{m,1}, Z'_{m-1,2}\} + p_{m,2}$. From (4.2),

$$\begin{aligned}
Z'_{m,2} &= Z'_{m-1,2} + p_{m,2} = H_2(J_r) + \sum_{i=1}^{m-1} p_{i,2} + p_{m,2} = \sum_{i=1}^{r} I_{i,2} + \sum_{i=1}^{m} p_{i,2} \\
&= \sum_{i=1}^{m} I_{i,2} + \sum_{i=m+1}^{r} I_{i,2} + \sum_{i=1}^{m} p_{i,2} \\
&= \sum_{i=1}^{m} I_{i,2} + \sum_{i=1}^{m} p_{i,2} + I_{m+1,2} + \sum_{i=m+2}^{r} I_{i,2} \\
&= Z_{m,2} + \max\{Z_{m+1,1} - Z_{m,2}, 0\} + \sum_{i=m+2}^{r} I_{i,2} \\
&= \max\{Z_{m,2} + Z_{m+1,1} - Z_{m,2}, Z_{m,2}\} + \sum_{i=m+2}^{r} I_{i,2} \\
&= \max\{Z_{m+1,1}, Z_{m,2}\} + \sum_{i=m+2}^{r} I_{i,2} \\
&\geq Z_{m+1,1} + \sum_{i=m+2}^{r} I_{i,2} \\
&\geq Z_{m+1,1} \quad \text{since} \quad \sum_{i=m+2}^{r} I_{i,2} \geq 0.
\end{aligned} \tag{4.3}$$

From (4.3), $I'_{m+1,2} = \max\{Z_{m+1,1} - Z'_{m,2}, 0\} = 0$. Therefore the result holds for $k = m+1$ also. Hence by mathematical induction this theorem holds for all k, where $k = 1, 2, \ldots, r$. □

THEOREM 4.3. *If $T = \sum_{i=1}^{n-r} (n - r - i + 1) p_{i,k}$, then T is the minimum for that sequence in which jobs are arranged in increasing order of $p_{i,k}$.*

PROOF. Consider the following sequences S and S':

$$S : 1, 2, \ldots, j-1, j, j+1, j+2, \ldots, n-r,$$
$$S' : 1, 2, \ldots, j-1, j+1, j, j+2, \ldots, n-r.$$

https://doi.org/10.1017/S1446181100010026 Published online by Cambridge University Press

If T is the corresponding value for sequence S' then

$$T' = \sum_{i=1}^{j-1}(n-r+1-i)p_{i,k} + (n-r+1-j)p_{j+1,k} + \left(n-r+1-\overline{j+1}\right)p_{j,k} + \sum_{i=j+2}^{n-r}(n-r+1-i)p_{i,k}.$$

Sequences S will be preferable to sequence S' if job j precedes job $j+1$, that is, if $T \le T'$, that is,

$$\begin{aligned}\sum_{i=1}^{j-1}&(n-r+1-i)p_{i,k} + (n-r+1-j)p_{j,k} \\ &+ \left(n-r+1-\overline{j+1}\right)p_{j+1,k} + \sum_{i=j+2}^{n-r}(n-r+1-i)p_{i,k} \\ &\le \sum_{i=1}^{j-1}(n-r+1-i)p_{i,k} + (n-r+1-j)p_{j+1,k} \\ &+ \left(n-r+1-\overline{j+1}\right)p_{j,k} + \sum_{i=j+2}^{n-r}(n-r+1-i)p_{i,k},\end{aligned}$$

that is,

$$\begin{aligned}(n-r+1-j)p_{j,k} &+ \left(n-r+1-\overline{j+1}\right)p_{j+1,k} \\ &\le (n-r+1-j)p_{j+1,k} + \left(n-r+1-\overline{j+1}\right)p_{j,k},\end{aligned}$$

that is,

$$\begin{aligned}(n-r-j)p_{j,k} &+ p_{j,k} + (n-r-j)p_{j+1,k} \\ &\le (n-r-j)p_{j+1,k} + p_{j+1,k} + (n-r-j)p_{j,k},\end{aligned}$$

that is, $p_{j,k} \le p_{j+1,k}$. The proof is therefore complete. □

5. Branch and bound technique

Let us denote $a_i = p_{i,1}$, $b_i = p_{i,2}$ and $C_i = Z_{i,2}$. Consider any partial schedule J_r having r jobs. The sum of completion times (which is equivalent to the mean flowtime) for this schedule can be divided into

$$\sum_{i\in J_r} C_i + \sum_{i\in \bar{J}_r} C_i. \tag{5.1}$$

https://doi.org/10.1017/S1446181100010026 Published online by Cambridge University Press

Under the no-idle constraint, machine M_j, $j = 1, 2$, works continuously without idle intervals. The idle time for machine M_1 is always zero and the idle time for machine M_2 can be made zero by applying Theorem 4.2. By Theorem 4.2, for the partial schedule J_r, the earliest time at which machine M_2 should start the processing of the first job of J_r such that M_2 works continuously without a break is

$$H_2(J_r) = C_{l_r} - \sum_{k=1}^{r} b_{l_k},$$

where $l_1, l_2, \dots, l_r$ represent the $1^{\text{st}}, 2^{\text{nd}}, \dots, r^{\text{th}}$ job of partial schedule J_r.

Now $C_{l_1} = H_2(J_r) + b_{l_1}$ and $C_{l_2} = H_2(J_r) + \sum_{k=1}^{2} b_{l_k}$. Continuing,

$$C_{l_r} = H_2(J_r) + \sum_{k=1}^{r} b_{l_k}.$$

Therefore

$$\begin{aligned}\sum_{i=1}^{r} C_{l_i} &= rH_2(J_r) + b_{l_1} + \sum_{k=1}^{2} b_{l_k} + \dots + \sum_{k=1}^{r} b_{l_k} \\ &= rH_2(J_r) + \sum_{k=1}^{r} (r+1-k) b_{l_k} \\ &= r\left(C_{l_r} - \sum_{k=1}^{r} b_{l_k}\right) + \sum_{k=1}^{r} (r+1-k) b_{l_k},\end{aligned}$$

that is,

$$\sum_{i \in J_r} C_i = r\left(C_{l_r} - \sum_{i \in J_r} b_i\right) + \sum_{k=1}^{r} (r+1-k) b_{l_k}.$$

Hence the lower bound for the first sum of (5.1) can be defined as

$$S_r = r\left(C_{l_r} - \sum_{i \in J_r} b_i\right) + \sum_{k=1}^{r} (r+1-k) b_{l_k}. \tag{5.2}$$

To obtain the lower bound for the second sum of (5.1) we define

$$A_r = (n-r) \sum_{j \in J_r} a_j + \sum_{k=1}^{n-r} (n-r-k+1) a_{i_k} + \sum_{k=1}^{n-r} b_{i_k}, \tag{5.3}$$

where $i_1, i_2, \dots, i_{n-r}$ is a permutation of jobs in $\tilde{J}_r$, and

$$B_r = (n-r) \max\left\{C_{l_r}, \sum_{j \in J_r} a_j + \min_{i \in \tilde{J}_r} a_i\right\} + \sum_{k=1}^{n-r} (n-r-k+1) b_{j_k}, \tag{5.4}$$

https://doi.org/10.1017/S1446181100010026 Published online by Cambridge University Press

where l_r is the last job and $j_1, j_2, \dots, j_{n-r}$ is a permutation of jobs in $\bar{J}_r$.

In Equation (5.3), $(n-r)\sum_{j\in J_r} a_j + \sum_{k=1}^{n-r} b_{i_k}$ is constant. Therefore A_r is at a minimum when $\sum_{k=1}^{n-r}(n-r-k+1)a_{i_k}$ is at a minimum. By Theorem 4.3, $\sum_{k=1}^{n-r}(n-r-k+1)a_{i_k}$ is minimised for that schedule in which jobs are arranged in increasing order of the processing times of the jobs on machine M_1. Let us denote this by $\hat{A}_r$.

In Equation (5.4), $(n-r)\max\left\{C_{l_r}, \sum_{j\in J_r} a_j + \min_{i\in\bar{J}_r} a_i\right\}$ is constant. Therefore B_r is minimised when $\sum_{k=1}^{n-r}(n-r-k+1)b_{j_k}$ is at a minimum. By Theorem 4.3, $\sum_{k=1}^{n-r}(n-r-k+1)b_{j_k}$ is the minimum for that schedule in which jobs are arranged in increasing order of the processing times of the jobs on machine M_2. Let us denote this by $\hat{B}_r$.

Therefore a lower bound for any partial schedule J_r under a no-idle constraint is given by

$$\mathrm{LB}[J_r] = S_r + \max\left(\hat{A}_r, \hat{B}_r\right). \tag{5.5}$$

6. Example

Consider a 5-job, 2-machine sequencing problem whose processing times are given in Table 1.

TABLE 1. Processing times for a 5-job, 2-machine sequencing problem.

Jobs	1	2	3	4	5
Machine M_1	3	12	9	11	2
Machine M_2	5	15	2	6	20

The lower bound for the partial schedule $J_r = 1$ is computed using (5.5) as follows:

$$\begin{aligned}\mathrm{LB}[1] = 3+5+\max[&4\times3+4\times2+3\times9+2\times11+12+(15+2+6+20),\\ &8\times4+4\times2+3\times6+2\times15+20]\\ = 8+\max[&12+8+27+22+12+43, 32+8+18+30+20]\\ = 8+\max[&124, 108] = 8+124 = 132.\end{aligned}$$

Similarly, the lower bounds for the partial schedules $J_r = 2, 3, 4$ and 5 are 190, 144, 158 and 160 units respectively.

The minimum value of the lower bound is for vertex $J_r = 1$. Therefore $J_r = 1$ is the branching node.

https://doi.org/10.1017/S1446181100010026 Published online by Cambridge University Press

For $J_r = 12$,

$$\begin{aligned} \mathrm{LB}[12] &= 2 \times 10 + 2 \times 5 + 15 + \max[3\times15+3\times2+2\times9+11+(2+6+20), \\ &\qquad\qquad 3\times30+3\times2+2\times6+20] \\ &= 20 + 10 + 15 + \max[45 + 6 + 18 + 11 + 28, 90 + 6 + 12 + 20] \\ &= 45 + \max[108, 128] = 45 + 128 = 173. \end{aligned}$$

Similarly, the lower bounds for the partial schedules $J_r = 13$, 14 and 15 are 143, 155 and 153 units respectively.

The minimum value of the lower bound is for $J_r = 13$. Therefore $J_r = 13$ is the branching node.

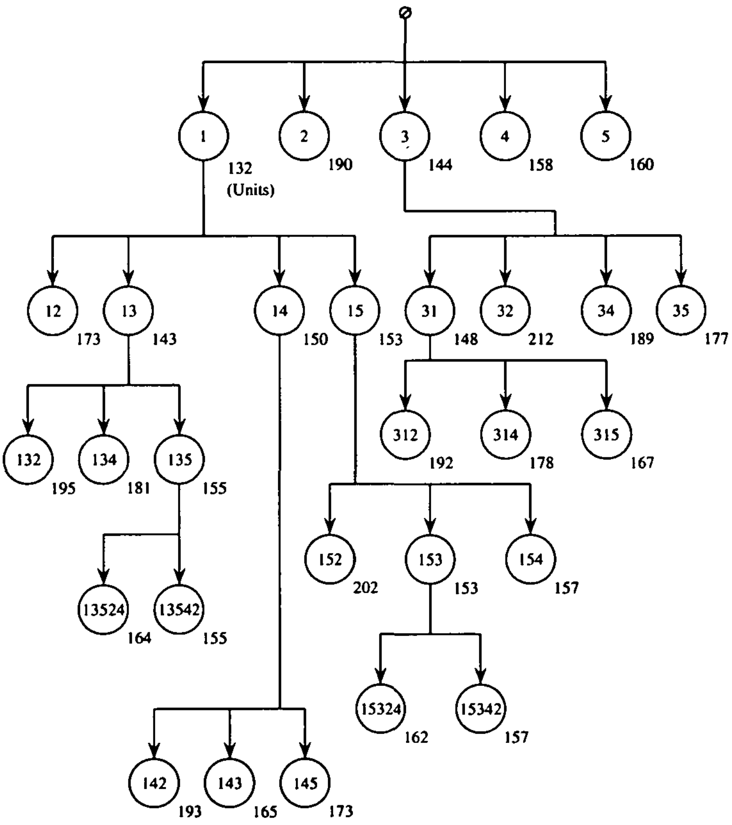

FIGURE 1. Scheduling tree.

Continuing in this way, the branch and bound technique is applied for evaluations of relevant lower bounds and a scheduling tree is obtained as in Figure 1.

https://doi.org/10.1017/S1446181100010026 Published online by Cambridge University Press

Hence the optimal sequence is 1–3–5–4–2 and the minimum sum of completion times subject to no-idle intervals on the machines is 155 units.

7. Conclusion

The problem of a two-machine flowshop, where the machines work continuously without any break after starting the processing of the first job, has been considered with respect to minimising the total (mean) flowtime.

The $n/2/F$, no-idle/$\sum C_i$ problem is NP-complete. The first machine has no idle time, that is, the first machine always works continuously without any break after starting the processing of the first job. The second machine may have idle times. For the second machine to work continuously, the processing of the first job on it should be delayed. Theorems 4.1 and 4.2 provide an expression for the earliest time at which processing on the second machine must be started such that the second machine works continuously without any break until the last job is completed on it. The branch-and-bound algorithm of this paper provides an optimal solution. In general, a minimum $n(n+1)/2$ nodes must be created.

Adiri and Pohoryles [1] considered an $n/m/P$, no-idle/$\sum C_i$; $m \geq 2$, problem with an increasing or decreasing series of dominating machines and provided polynomial bounded algorithms. The $n/m/F$, no-idle/$\sum C_i$; $m > 2$, problem is still open.

Acknowledgements

The authors are very grateful to the referees for their constructive comments and valuable suggestions for improving an earlier version of this paper.

References

[1] I. Adiri and D. Pohoryles, "Flowshop/no-idle or no wait scheduling to minimize the sum of completion time", *Naval Res. Logist. Quart.* **29** (1982) 495–504.

[2] T. Aldowaisan and A. Allahverdi, "Total flowtime in no-wait flowshops with separated setup times", *Comput. Oper. Res.* **25** (1998) 757–765.

[3] A. Allahverdi and T. Aldowaisan, "No-wait and separate setup three-machine flowshop with total completion time criterion", *Int. Trans. Oper. Res.* **7** (2000) 245–264.

[4] A. Allahverdi and T. Aldowaisan, "Minimizing total completion time in a no-wait flowshop with sequence dependent additive changeover times", *J. Oper. Res. Soc.* **52** (2001) 449–462.

[5] C. Chen, V. Neppalli and N. Aljaber, "Genetic algorithms applied to the continuous flowshop problem", *Comput. Indust. Engng.* **30** (1996) 919–929.

https://doi.org/10.1017/S1446181100010026 Published online by Cambridge University Press

[6] R. W. Conway, W. L. Maxwell and L. W. Miller, *Theory of scheduling* (Addison-Wesley, Massachusetts, 1967).

[7] M. R. Garey, D. S. Johnson and R. Sethi, "The complexity of flowshop and jobshop scheduling", *Math. Oper. Res.* **1, 2** (1976) 117–129.

[8] N. G. Hall and C. Sriskandarajah, "A survey of machine scheduling problems with blocking and no-wait in process", *Oper. Res.* **44** (1996) 510–525.

[9] E. Ignall and L. Schrage, "Applications of the branch-and-bound technique to some flowshop scheduling problems", *Oper. Res.* **13** (1965) 400–412.

[10] S. M. Johnson, "Optimal two and three stage production schedules with setup times included", *Naval Res. Logist. Quart.* **1** (1954) 61–68.

[11] E. L. Lawler and D. E. Wood, "Branch and bound method: A survey", *Oper. Res.* **14** (1966) 699–719.

[12] L. Narain, "Minimizing total hiring cost of machines in $n \times m$ flowshop problem", *J. Decis. Math. Sci.* **7** (2002) 23–32.

[13] L. Narain and P. C. Bagga, "Minimizing total elapsed time subject to zero idle time of machines in $n \times 3$ flowshop problem", *Indian J. Pure Appl. Math.* **34** (2003) 219–228.

[14] C. Rajendran and D. Chaudhuri, "Heuristic algorithms for continuous flowshop problem", *Naval Res. Logist. Quart.* **37** (1990) 695–705.

[15] A. H. J. Rinnooy-Kan, *Machine scheduling problems: classification, complexity and computations* (Nijhoff Publishing, The Hague, 1976).

[16] S. Szwarc, "The flowshop problem with mean completion time criterion", *AIIE Trans.* **15** (1983) 172–176.

[17] J. A. A. van der Veen and R. van Dal, "Solvable cases of the no-wait flowshop scheduling problem", *J. Oper. Res. Soc.* **42** (1991) 971–980.

https://doi.org/10.1017/S1446181100010026 Published online by Cambridge University Press