Jurnal Teknik Industri, Vol. 19, No. 1, Februari 2018, pp. 73-81
ISSN 1978-1431 print / ISSN 2527-4112 online
https://doi.org/10.22219/JTIUMM.Vol19.No1.73-81

Improve Algoritma Hodgson Untuk Meminimasi Jumlah *Job* Terlambat Pada Penjadwalan *Flow shop*

Dian Setiya Widodo
Program Studi Teknik Manufaktur, Politeknik 17 Agustus Surabaya
Jl. 45, Gedung N&P, Semolowaru, Sukolilo, Surabaya, Jawa Timur 60118
Korespondensi penulis, surel: diansetiyawidodo@yahoo.com

Abstract

Since Johnson published a paper in 1954, the problem of job scheduling has received the attention of hundreds of practitioners and researchers, as one of the most studied topics in Operation research. Several heuristic studies have discussed the problem of flow shop scheduling to minimize the completion time (makespan). In this paper, we consider the problem of pure flow shop scheduling to minimize the number of jobs of tardy. We have developed a Hudson algorithm for minimization solution the number jobs of tardy. Hodgson's improved Heuristic algorithm was tested and compared to the EDD priority rule. The Numerical experimental results show the new algorithm provides a better solution than priority EDD. The improving Hodgson algorithm gives a minimum number of tardy jobs.

Keywords*: Flow shop, The number jobs of tardy, Hodgson*

Abstrak

Sejak Johnson menerbitkan makalah pada tahun 1954, masalah penjadwalan pekerjaan telah mendapat perhatian dari ratusan praktisi dan peneliti, sebagai salah satu topik yang paling banyak dipelajari dalam Penelitian Operasi. Beberapa penelitian heuristik telah membahas masalah penjadwalan flow shop meminimalkan waktu penyelesaian (makespan). Pada makalah ini, kami mempertimbangkan masalah penjadwalan pure flow shop untuk meminimalkan jumlah job yang terlambat. Kami telah mengembangkan algoritma hodgson untuk menghasilkan solusi minimum jumlah job yang terlambat. Algoritma Heuristik improve hodgson ini diuji dan dibandingkan dengan dengan aturan prioritas EDD. Hasil percobaan numerik menunjukkan Algoritma improve hodgson lebih baik dari aturan EDD. Algoritma improve hodgson menghasilkan minimum jumlah pekerjaan terlambat.

Kata kunci: *Flow shop, jumlah job terlambat, Hodgson*

1. Pendahuluan

Penjadwalan merupakan pengaturan waktu pada mesin yang mencakup kegiatan mengalokasikan fasilitas dan peralatan. Penjadwalan berguna untuk menentukan urutan pekerjaan [1]. Fungsi penjadwalan yaitu memproses produk sesuai dengan waktu yang diinginkan, memprediksi kesiapan setiap sumber daya yang diperlukan [2]. Berdasarkan aliran produksi, penjadwalan di klasifikasi menjadi *flow shop* dan *job shop*. Harto, *et al.* [3] menyatakan bahwa pada penjadwalan *job shop* umumnya menggunakan algoritma aktif dan non delay. Husen, *et al.* [4] menyatakan bahwa pada penjadwalan *job shop* dapat menggunakan algoritma simulated annealing. Penjadwalan *flow shop* adalah penjadwalan *di mana* semua job yang akan dikerjakan melalui urutan proses operasi yang sama [1]. Umumnya performansi dari penjadwalan yaitu untuk meminimalkan *flow time, makespan, maximum tardiness*, *job tardy* dan *lateness* [5]. Pengertian dari flow time itu sendiri ialah waktu yang dibutuhkan sebuah pekerjaan untuk berada di dalam sebuah sistem produksi. Sedangkan waktu yang dibutuhkan untuk menyelesaikan seluruh pekerjaan (*job*) adalah *makespan* [6]. Dalam

Diterima 6 Februari, 2018; Direvisi 17 Februari, 2018; Disetujui 18 Februari, 2018

penjadwalan sendiri terdapat beberapa kategori berdasarkan sistem produksi diantaranya ialah *single machine*, *flow shop* dan *job shop*.

Aturan-aturan prioritas dasar pada penjadwalan produksi umumnya menggunakan metode *Shortest Processing Time* (SPT), *Longest Processing Time* (LPT), *Earliest Due date* (EDD), Aturan *Slack* [2] [7]. Beberapa metode tersebut dapat digunakan untuk beberapa penyelesaian solusi tertentu sebagai contoh yaitu meminimumkan *mean flow time, makespan, tardiness, mean tardiness, maximum tardiness*, dan *number of tardy job*. Metode SPT digunakan untuk meminimalkan mean *flow time*. Selanjutnya jika untuk meminimalkan *makespan* dan *mean flow time* dapat menggunakan metode LPT lalu dilanjut dengan metode SPT [8]. Jika untuk mengurangi *tardiness* dapat menggunakan aturan slack. Jika untuk mengurangi *mean tardiness* dapat menggunakan metode SPT, EDD, dan slack lalu dilanjut menggunakan algoritma genetika [9].

Beberapa algoritma pure *flow shop* yang telah diteliti adalah Johnson [10], Campbell, *et al.* [11], Gupta [12], Dannenbring [13], Logendran *and* Nudtasomboon [14], Pour [15], Nawaz, *et al.* [16] [17], *simulated annealing* [4] dan *Branch and Bound* [18]. Penelitian tersebut umumnya menggunakan kriteria minimasi *makespan*. Makespan adalah total wa ktu yang dibutuhkan untuk menyelesaikan seluruh *job* [5][7]. Selain itu ada beberapa penelitian yang mambahas tentang minimasi *job* yang terlambat antara lain Moore [19] membuat algoritma untuk minimasi jumlah *job* yang terlambat. Algoritma ini yang mengawali cikal bakal algoritma hodgson. Ho *and* Chang [20] telah melakukan penelitian untuk meminimasi jumlah *job* yang terlambat pada masalah *single* stage parallel mesin. Lodree Jr, *et al.* [21] telah melakukan penelitian untuk meminimasi jumlah *job* yang terlambat pada masalah *flow shop* dinamis. Azizoglu, *et al.* [22] telah melakukan penelitian untuk meminimasi dua kriteria performansi *maximum earliness* dan *number of tardy jobs* pada *single* mesin. M'Hallah *and* Bulfin [23], M'Hallah *and* Bulfin [24] telah melakukan penelitian untuk meminimasi Total bobot jumlah *job* yang terlambat pada masalah *single machine*. Li *and* Chen [25] telah melakukan penelitian minimasi *job tardy* untuk masalah *single stage* mesin paralel dengan *due date* seragam. Aydilek, *et al.* [26] telah melakukan penelitian meminimasi jumlah *job* yang terlambat untuk satu mesin dengan waktu proses yang tidak diketahui. Umumnya mengurangi *number of tardy job* dapat menggunakan metode algoritma Hodgson [27].

Beberapa penelitian untuk menimimasi jumlah *job* yang terlambat telah banyak dilakukan. Sayangnya, penelitian tentang minimasi menimimasi jumlah *job* yang terlambat kebanyakan masih dipakai untuk masalah *single* mesin atau *single* stage. Pada penelitian terdahulu, algoritma Hodgson merupakan algoritma heuristik terbaik dalam menghasilkan solusi minimasi menimimasi jumlah *job* namun khusus untuk kasus *single machine* [19]. Dalam penelitian ini, kami mengembangkan algoritma heuristik Hodgson pada penjadwalan produksi *flow shop* guna menemukan solusi minimasi jumlah *job* terlambat. Tujuan dari artikel ini adalah mengembangkan algoritma heuristik Hodgson pada penjadwalan produksi *flow shop* guna menemukan solusi minimasi jumlah *job* terlambat. Algoritma yang telah dikembangkan (*improve* Hodgson) dibandingkan dengan aturan prioritas EDD.

2. Metode Penelitian

Pada dasarnya algoritma Hogdson ini digunakan pada penjadwalan *single machine*. Dalam penelitian ini algoritma Hodgson digunakan untuk kasus *flow shop n-*

machine. Beberapa bagian algoritma telah dimodifikasi oleh peneliti. Tahapan penjadwalan *flow shop* untuk meminimasi jumlah *job* adalah sebagai berikut :

1. Setiap pekerjaan j dihitung total waktu proses (Tj) sesuai dengan persamaan 1. t_{ij} Menunjukkan waktu proses yang diperlukan pekerjaan j di mesin i.

$$Tj = \sum_{i=1}^{m} t_{ji} \quad (1)$$

2. Urutkan *Job* sesuai dengan *due date* yang paling kecil.
3. Lakukan penjadwalan. Hitung *lateness* berdasarkan persamaan 2 Jika tidak ada *job* yang terlambat maka urutan *job* tersebut sudah optimal. Jika ada yang terlambat lanjut ke langkah 4.

$$Lj = Cj - dj \quad (2)$$

Jika Lj ≤ 0, artinya saat penyelesaian memenuhi batas akhir.
Jika Lj > 0, artinya saat penyelesaian melewati batas akhir (Terlambat).

4. Jika *job* j pertama kali terlambat (saat penyelesaian melewati batas akhir (Lj > 0)), maka cari *job* a sebelum *job* j yang mempunyai waktu total pengerjaan paling lama. Kemudian hilangkan *job* a tersebut, untuk dikerjakaan setelah semua *job* tidak ada lagi yang terlambat setelah semuanya diproses
5. Lakukan penjadwalan kembali seperti langkah 3 tanpa *job* yang terlambat sebelumnya, lakukan langkah 4 dan 5 hingga *job* tidak ada yang terlambat atau hanya menyisakan 1 *job* yang terlambat diurutan paling belakang.
6. Letakkan semua *job* yang dihilangkan karena keterlambatan tadi dalam urutan paling akhir di urutan penjadwalan yang sudah terbentuk.
7. Hitung performansi kriteria jumlah *job* yang terlambat berdasarkan persamaan 3

$$Nt = \sum_{j=1}^{n} Nj \quad (3)$$

Nj = 1 jika Lj > 0
Nj = 0 jika Lj ≤ 0

3. Hasil dan Pembahasan

Percobaan numerik dilakukan dengan melakukan penjadwalan 10 *job flow shop* 2 mesin. Data percobaan numerik dapat dilihat pada Tabel 1.

Tabel 1 Data Percobaan Numerik

Job	Mesin 1	Mesin 2	*Due Date*
1	5	1	15
2	4	2	12
3	5	1	13
4	3	3	14
5	6	2	11
6	7	1	17
7	3	2	20
8	2	2	18
9	5	1	17
10	4	2	22

Langkah-langkah penyelesaian meminimasi jumlah *job* yang terlambat pada masalah *flow shop* adalah sebagai berikut:

1. Menghitung total waktu proses (Tj) setiap pekerjaan j. total waktu proses (Tj) dihitung sesuai dengan persamaan 1. Rekapitulasi perhitungan total waktu proses dapat dilihat pada Tabel 2. Contoh perhitungan Tj dapat dilihat berikut ini.
   $T_1 = t_{11} + t_{12}$
   $= 5 + 1 = 6$

Tabel 2 Rekapitulasi perhitungan total waktu proses

Job	Mesin 1	Mesin 2	Tj
1	5	1	6
2	4	2	6
3	5	1	6
4	3	3	6
5	6	2	8
6	7	1	8
7	3	2	5
8	2	2	4
9	5	1	6
10	4	2	6

2. Urutkan *Job* sesuai dengan *due date* yang paling kecil. Urutan berdasarkan *due date* terkecil adalah $J_5, J_2, J_3, J_4, J_1, J_6, J_9, J_8, J_7, J_{10}$.
3. Melakukan penjadwalan berdasarkan urutan *due date* terkecil dan menghitung *lateness* berdasarkan persamaan 2 berdasarkan urutan *job* pada langkah 2. Penjadwalan dan perhitungan *lateness* dapat dilihat pada Tabel 3. Contoh perhitungan Tj dapat dilihat berikut ini.

   $L_5 = C_5 - d_5$

   $= 6 - 11 = -5$

Tabel 3 Penjadwalan dan perhitungan lateness

					Mulai		Selesei		
Job	Mesin 1 (tj1)	Mesin 2 (tj2)	Tj	dj	M1	M2	M1	M2	Lj
5	4	2	6	11	0	4	4	6	-5
2	3	2	5	12	4	7	7	9	-3
3	5	4	9	13	7	12	12	16	3
4	4	7	11	14	12	16	16	23	9
1	1	1	2	15	16	23	17	24	9
6	7	3	10	17	17	24	24	27	10
9	2	1	3	17	24	27	26	28	11
8	2	2	4	18	26	28	28	30	12
7	1	2	3	20	28	30	29	32	12
10	1	1	2	22	29	32	30	33	11

4. *Job* 3 merupakan *job* pertama kali terlambat (saat penyelesaian melewati batas akhir (Lj > 0)). Selanjutnya mencari *job* yang mempunyai waktu total waktu pengerjaan paling

lama pada *job* 3 dan urutan sebelum. Berdasarkan *lateness* positif pertama urutan pekerjaan adalah J_5, J_2 dan J_3. Waktu terbesar adalah pada *job* 3. Kemudian hilangkan *job* 3 tersebut.

5. Melakukan penjadwalan kembali seperti langkah 3 hingga *job* tidak ada yang terlambat atau hanya menyisakan 1 *job* yang terlambat diurutan paling belakang. Rekapitulasi iterasi dan urutan *job* dapat dilihat pada Tabel 4.

Tabel 4 Rekapitulasi iterasi dan urutan *job*

Iterasi Ke	Urutan *Job*										*Job* yang dihilangkan
Iterasi 1	5	2	3	4	1	6	9	8	7	10	-
Iterasi 2	5	2	4	1	6	9	8	7	10		3
Iterasi 3	5	2	1	6	9	8	7	10			4
Iterasi 4	5	2	1	9	8	7	10				6

6. Letakkan semua *job* yang dalam urutan paling akhir. Sehingga urutan *job* adalah $J_5, J_2, J_1, J_9, J_8, J_7, J_{10}, J_3, J_4, J_6$

7. Hitung performansi kriteria urutan *job* adalah $J_5, J_2, J_1, J_9, J_8, J_7, J_{10}, J_3, J_4, J_6$ persamaan 3. Rekapitulasi perhitungan performansi *improve* hodgson dapat dilihat pada Tabel 5. Contoh perhitungan *job* yang terlambat sebagai berikut.

   $N_5 = 0$

Tabel 5 Rekapitulasi performansi algoritma *improve* Hodgson

Job	Mesin 1 (tj1)	Mesin 2 (tj2)	Tj	dj	Mulai		Selesai		Lj	Nj
					M1	M2	M1	M2		
5	4	2	6	11	0	4	4	6	-5	0
2	3	2	5	12	4	7	7	9	-3	0
1	1	1	2	13	7	9	8	10	-3	0
9	2	1	3	14	8	10	10	11	-3	0
8	2	2	4	15	10	12	12	14	-1	0
7	1	2	3	17	12	14	13	16	-1	0
10	1	1	2	17	13	16	14	17	0	0
3	5	4	9	13	14	19	19	23	10	1
4	4	7	11	14	19	23	23	30	16	1
6	7	3	10	17	23	30	30	33	16	1
Total Nt										3

Perhitungan algoritma *improve* Hodgson juga dibandingkan dengan aturan prioritas *Earliest Due date* (EDD). Hasil perhitungan penjadwalan aturan prioritas EDD dapat dilihat pada Tabel 6. Hasil Tabel 6 perhitungan penjadwalan produksi aturan prioritas menunjukkan bahwa penjadwalan dengan aturan prioritas EDD menghasilkan 8 dari 10 *job* terlambat. Sedangkan pada penjadwalan *improve* hodgson menghasilkan 3

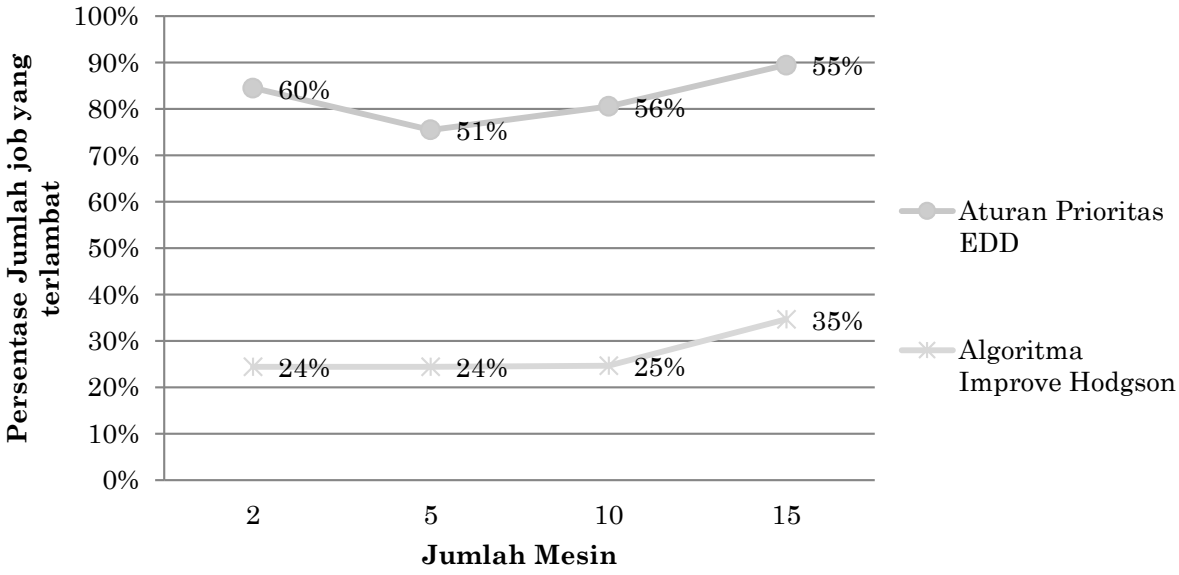
dari 10 *job* terlambat. Percobaan ini menunjukkan bahwa algoritma *improve* hodgson efektif dalam meminimasi jumlah *job* terlambat pada kasus *flow shop*.

Tabel 6 Perhitungan penjadwalan aturan prioritas EDD

Job	Mesin 1 (tj1)	Mesin 2 (tj2)	Tj	dj	Mulai M1	Mulai M2	Selesei M1	Selesei M2	Lj	Nj
5	4	2	6	11	0	4	4	6	-5	0
2	3	2	5	12	4	7	7	9	-3	0
3	5	4	9	13	7	12	12	16	3	1
4	4	7	11	14	12	16	16	23	9	1
1	1	1	2	15	16	23	17	24	9	1
6	7	3	10	17	17	24	24	27	10	1
9	2	1	3	17	24	27	26	28	11	1
8	2	2	4	18	26	28	28	30	12	1
7	1	2	3	20	28	30	29	32	12	1
10	1	1	2	22	29	32	30	33	11	1
Total Nt										8

Kami juga melakukan percobaan numerik untuk membuktikan perbandingan algoritma usulan dengan aturan prioritas EDD. Percobaan dilakukan pada masalah *flow shop* untuk 2, 5, 10 dan 15 *stage*/ mesin. Setiap *stage* kami lakukan percobaan sebanyak 5. Rekapitulasi percobaan numerik yang telah dilakukan dari perbandingan antara hasil algoritma baru ini dengan algoritma *Earliest Due date* (EDD) dapat dilihat pada

Tabel **7**. Persentase *job* yang terlambat dari jumlah *job* di tiap percobaan dapat dilihat pada Gambar 1. Percobaan tersebut menunjukkan bahwa algoritma usulan dapat meminimasi jumlah *job* yang terlambat.



Gambar 1 Persentase *job* yang terlambat terhadap jumlah mesin

Tabel 7 Rekapitulasi percobaan numerik

Percobaan 1	Jumlah stage/mesin	Jumlah *Job*	Jumlah *Job* yang terlambat	
			Improve Hogson	Prioritas EDD
1	2	10	2	8
2	5	8	3	5
3	10	8	4	5
4	15	6	3	4
5	2	15	6	10
6	5	13	4	8
7	10	13	5	10
8	15	11	6	8
9	2	20	5	11
10	5	18	1	8
11	10	18	2	8
12	15	16	6	7
13	2	25	6	13
14	5	23	7	11
15	10	23	3	13
16	15	21	5	11
17	2	30	4	14
18	5	28	5	11
19	10	28	3	11
20	15	26	2	10

4. Simpulan

Dalam penelitian ini di simpukan bahwa pengembangan algoritma *improve* Hodgson yang dilakukan untuk meminimasi jumlah *job* yang terlambat memberikan hasil penjadwalan *flow shop* yang lebih baik dari aturan prioritas *Earliest Due date* (EDD). Algoritma *improve* Hodgson menghasilkan jumlah pekerjaan terlambat yang lebih minimum dibandingkan dengan aturan EDD. Diharapkan penelitian ini dapat membuka peluang untuk penelitian-penelitian selanjutnya untuk mengembangkan algoritma pada kasus *flexible* maupun *hybrid flow shop* agar menghasilkan solusi yang lebih baik lagi terutama dalam meminimasi jumlah *job* yang terlambat..

Daftar Notasi

Tj : Total waktu proses
t_{ij} : Waktu proses yang diperlukan pekerjaan j di mesin i.
Lj : *Lateness* pekerjaan j.
Cj : *Completion time* pekerjaan j
dj : *due date* pekerjaan j
Nj : pekerjaan j terlambat =1 jika Lj > 0, selain itu 0
Nt : Jumlah pekerjaan yang terlambat

Referensi

[1] K. R. Baker *and* D. Trietsch, Principles of sequencing *and* scheduling: John Wiley & Sons, 2013.
[2] M. L. Pinedo, Scheduling: theory, algorithms, *and* systems: Springer Science & Business Media, 2012.
[3] S. Harto, A. K. Garside, *and* D. M. Utama, "penjadwalan produksi menggunakan algoritma jadwal non delay untuk meminimalkan makespan studi kasus di cv. Bima mebel," Spektrum Industri, vol. 14, 2016.
[4] M. Husen, I. Masudin, *and* D. M. Utama, "Penjadwalan Job Shop Statik Dengan Metode Simulated Annealing Untuk Meminimasi Waktu Makespan," Spektrum Industri, vol. 13, 2015.
[5] K. R. Baker, Introduction to sequencing *and* scheduling: John Wiley & Sons, 1974.
[6] M. Pinedo, Planning *and* scheduling in manufacturing *and* services vol. 24: Springer, 2005.
[7] D. M. Utama, "Analisa Perbandingan Penggunaan Aturan Prioritas Penjadwalan Pada Penjadwalan Non Delay N Job 5 Machine," Research Report, vol. 1, 2017.
[8] C. Rajendran, "Heuristic algorithm for scheduling in a flowshop to minimize total flowtime," International Journal of Production Economics, vol. 29, pp. 65-73, 1993.
[9] Y. Li, W. Ip, *and* D. Wang, "Genetic algorithm approach to earliness *and* tardiness production scheduling *and* planning problem," International Journal of Production Economics, vol. 54, pp. 65-76, 1998.
[10] S. M. Johnson, "Optimal two-and three-stage production schedules with setup times included," Naval Research Logistics (NRL), vol. 1, pp. 61-68, 1954.
[11] H. G. Campbell, R. A. Dudek, *and* M. L. Smith, "A heuristic algorithm for the n job, m machine sequencing problem," Management science, vol. 16, pp. B-630-B-637, 1970.
[12] J. N. Gupta, "Heuristic algorithms for multistage flowshop scheduling problem," AIIE Transactions, vol. 4, pp. 11-18, 1972.
[13] D. G. Dannenbring, "An evaluation of flow shop sequencing heuristics," Management science, vol. 23, pp. 1174-1182, 1977.
[14] R. Logendran *and* N. Nudtasomboon, "Minimizing the makespan of a group scheduling problem: a new heuristic," International Journal of Production Economics, vol. 22, pp. 217-230, 1991.
[15] H. D. Pour, "A new heuristic for the n-job, m-machine flow-shop problem," Production Planning & Control, vol. 12, pp. 648-653, 2001.
[16] M. Nawaz, E. E. Enscore, *and* I. Ham, "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem," Omega, vol. 11, pp. 91-95, 1983.
[17] I. Masudin, D. M. Utama, *and* F. Susastro, "Penjadwalan Flowshop Menggunakan Algoritma Nawaz Enscore Ham," Jurnal Ilmiah Teknik Industri, vol. 13, pp. 54-59, 2014.
[18] Z. Lomnicki, "A" branch-and-bound" algorithm for the exact solution of the three-machine scheduling problem," OR, pp. 89-100, 1965.
[19] J. Moore, "Sequencing n jobs on one machine to minimize the number of tardy jobs," Management Science, vol. 15, pp. 102-109, 1968.

[20] J. C. Ho *and* Y.-L. Chang, "Minimizing the number of tardy jobs for m parallel machines," European Journal of Operational Research, vol. 84, pp. 343-355, 1995.
[21] E. Lodree Jr, W. Jang, *and* C. M. Klein, "A new rule for minimizing the number of tardy jobs in dynamic flow shops," European Journal of Operational Research, vol. 159, pp. 258-263, 2004.
[22] M. Azizoglu, S. Kondakci, *and* M. Köksalan, "Single machine scheduling with maximum earliness *and* number tardy," Computers & Industrial Engineering, vol. 45, pp. 257-268, 2003.
[23] R. M'Hallah *and* R. Bulfin, "Minimizing the weighted number of tardy jobs on a single machine," European Journal of Operational Research, vol. 145, pp. 45-56, 2003.
[24] R. M'Hallah *and* R. Bulfin, "Minimizing the weighted number of tardy jobs on a single machine with release dates," European Journal of Operational Research, vol. 176, pp. 727-744, 2007.
[25] S.-S. Li *and* R.-X. Chen, "Single-machine parallel-batching scheduling with family jobs to minimize weighted number of tardy jobs," Computers & Industrial Engineering, vol. 73, pp. 5-10, 2014.
[26] A. Aydilek, H. Aydilek, *and* A. Allahverdi, "Algorithms for minimizing the number of tardy jobs for reducing production cost with uncertain processing times," Applied Mathematical Modelling, vol. 45, pp. 982-996, 2017.
[27] C.-C. Wu, W.-C. Lee, *and* T. Chen, "Heuristic algorithms for solving the maximum lateness scheduling problem with learning considerations," Computers & Industrial Engineering, vol. 52, pp. 124-132, 2007.