JURNAL ILMIAH TEKNIK INDUSTRI
ISSN: 1412-6869 (Print), ISSN: 2460-4038 (Online)
journal homepage: http://journals.ums.ac.id/index.php/jiti/index
doi: 10.23917/jiti.v18i1.7683

Pengembangan Algoritma *Hybrid Flowshop Three-Stage* Dengan Mempertimbangkan Waktu Setup

Dana Marsetiya Utama[1a],Annisa Kesy Garside[1], Wahyu Wicaksono[1]

Abstract. *Hybrid flowshop scheduling is one topic that is often reviewed by researchers at this time. Hybrid flowshop scheduling is the development of problems from pure flowshop. Flowshop problems have one machine at each stage. In this problem, each stage of operation has a machine that is arranged in parallel. This article aims to discuss the issue of hybrid flowshop scheduling at three stages to minimize makespan. Some previous studies discussed scheduling problems by considering setup time. However, such research is generally for the problem of pure flowshop. Therefore, a new algorithm is proposed to solve the problem. The proposed algorithm is developed from the Pour heuristic algorithm. Several experiments were conducted to determine the performance of the proposed algorithm. This study uses ten numerical experiments. This experiment uses the number of jobs varying from 5 jobs to 50 jobs. The results of numerical experiments show that the proposed algorithm has better performance compared to some other algorithms. The proposed method produces an effective solution if it is used to solve problems with a large number of jobs..*

Keywords: *production scheduling, hybrid flowshop three stages, Pour algorithm, setup time.*

Abstrak. *Penjadwalan hybrid flowshop adalah salah satu topik yang sering dikaji oleh peneliti saat ini. Penjadwalan hybrid flowshop merupakan pengembangan masalah dari pure flowshop. Berbeda dengan kasus flowshop yang memiliki satu mesin di tiap tahapan. Pada masalah ini, tiap tahapan operasi memiliki mesin yang dususun secara paralel. Artikel ini bertujuan untuk membahas permasalahan penjadwalan hybrid flowshop three stage untuk meminimasi makespan. Beberapa penelitian terdahulu membahas masalah penjadwalan dengan mempertimbangkan waktu setup. Namun, penelitian tersebut umumnya untuk masalah pure flowshop. Oleh karena itu, Algoritma baru diusulkan untuk menyelesaikan masalah tersebut. Algoritma usulan dikembangkan dari algoritma heuristik Pour. Beberapa percobaan dilakukan untuk mengetahui performansi dari algoritma usulan. Penelitian ini menggunakan 10 percobaan numerik. Percobaan ini menggunakan jumlah job bervariasi dari 5 job sampai 50 job. Hasil percobaan numeric menunjukkan algoritma usulan memiliki perfomansi yang lebih baik dibandingkan dengan beberapa algoritma lain. Metode usulan menghasilkan solusi yang efektif, apabila digunakan untuk menyelesaikan masalah dengan jumlah job yang besar.*

Kata Kunci: *penjadwalan produksi, hybrid flowshop three stages, algoritma Pour, waktu setup.*

I. PENDAHULUAN

Penjadwalan produksi memiliki arti perencanaan urutan pengerjaan *job* untuk mendapatkan performa yang lebih efektif dan efisien (Harto dkk., 2016; Husen dkk., 2015; Ong, 2013). Penjadwalan *hybrid flowshop* memiliki definisi sama dengan penjadwalan *pure flowshop*. Namun, *hybrid flowshop* memiliki beberapa mesin paralel pada *stage* tertentu (Marichelvam dkk., 2014). Penjadwalan produksi merupakan permasalahan yang masuk ke dalam lingkup *non-polinomial problem* (NP) (Abdel-Basset dkk., 2018). Sehingga, jumlah *job* akan berpengaruh pada waktu komputasi yang dibutuhkan (Utama, 2019; Utama dkk., 2019).

Terdapat beberapa penelitian terdahulu yang telah membahas tentang permasalahan *hybrid flowshop* dengan waktu setup terpisah. Ebrahimi dkk. (2014) menggunakan algortima genetika untuk meminimasi *makespan* permasalahan *hybrid flowshop* dengan *dependent setup time*. Ying dan Lin (2018) telah melakukan penelitian mengenai permasalahan *no-wait hybrid flowshop with setup time* menggunakan algoritma *metaheuristic* TPM dengan fungsi tujuan minimasi *makespan*. Gómez-Gasquet dkk. (2012) mengembangkan disain dari algortima *metahuristic genetic* untuk meminimasi *makespan* dengan *dependent setup time*. Zandieh dan Karimi (2011) telah melakukan penjadwalan produksi *hybrid*

[1] Jurusan Industri, Fakultas Teknik, Universitas Muhammadiyah Malang, Jl. Raya Tlogomas No. 246, Malang, Jawa Timur 65144

[*] email: dana@umm.ac.id

Diajukan: 01-02-2019 Diperbaiki: 19-06-2019
Disetujui: 25-06-2019

flowshop dengan waktu setup terpisah menggunakan algoritma metaheuristik genetika untuk meminimasi *makespan*.

Selain itu, beberapa penelitian terdahulu membahas tentang permasalahan *pure flowshop* dengan *setup time* terpisah. Allahverdi dan Al-Anzi (2006) membahas algoritma *branch and bound* pada kasus *flowshop three stage* dengan fungsi tujuan minimasi *makespan*. Ruiz dan Stützle (2008) menggunakan algoritma heuristik IG (*Iterated Greedy*) untuk meminimasi *makespan* dan *tardiness*. Lin dkk. (2009), dan Proust dkk. (1991), telah melakukan penjadwalan *pure flowshop* dengan waktu setup dan removel terpisah menggunakan algoritma DFFP. Aldowaisan dan Allahverdi (2004) dan Allahverdi dan Aldowaisan (2000) telah melakukan penjadwalan produksi *pure flowshop* dengan waktu setup terpisah menggunakan algoritma heuristic *no-wait* untuk meminimasi total *complication time*. Sedangkan beberapa algoritma metaheuristik yang sering dipakai untuk menyelesaikan kasus penjadwalan adalah *simulated annealing* (Firdaus dkk., 2015; Husen dkk., 2015), *Artificial Immune System* (Nasution dkk., 2017), dan CEGA (Widodo, 2017).

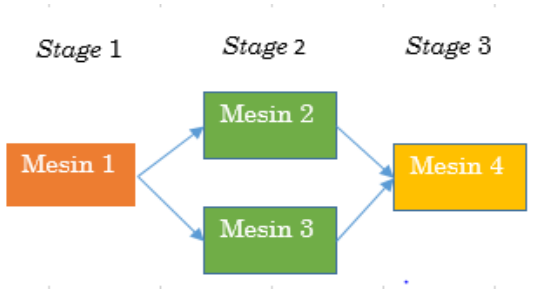
Selain itu, ada algoritma heuristik yang sering digunakan dalam menyelesaikan permasalahan *pure flowshop.* Algoritma tersebut diantaranya adalah algoritma CDS, dan NEH (Masudin dkk., 2014; Utama, 2018). Algoritma tersebut efektif memberikan *makespan* yang efisien. Namun saat ini, Pour (2001) telah mengusulkan algoritma baru pada proses produksi *flowshop* untuk meminimasi *makespan*. Algoritma Pour memberikan hasil yang lebih baik dibandingkan CDS dan NEH. Soetanto dkk. (2005) telah melakukan perbandingan penjadwalan produksi dengan menggunakan algoritma heuristik Pour dan algoritma optimasi *mixed integer programing* (MIP) untuk meminimasi *makespan.* Hasil percobaan mereka menunjukan algoritma heuristik Pour tidak kalah baik dibandingkan MIP. Sulaksmi dkk. (2014) juga telah melakukan penjadwalan produksi dengan menggunakan algoritma heuristik Pour pada industri manufaktur konveksi dimana hasil *makespan* dari algoritma heuristik Pour lebih baik di bandingkan dengan algoritma yang digunakan oleh perusahaan.

Pada penelitian-penelitian terdahulu, masih sangat minim algoritma heuristik untuk meminimasi *makespan hybrid flowshop three-stage* dengan mempertimbangkan waktu setup. Beberapa penelitian mengembangkan algoritma *hybrid flowshop* dengan dasar algoritma CDS. Selain itu, algoritma pour adalah ada salah satu algoritma yang bagus untuk dikembangkan. Pada penelitian sebelumnya, algoritma heuristik Pour hanya digunakan pada permasalahan *pure flowshop* saja. Algoritma ini belum pernah dikembangkan untuk masalah *hybrid flowshop* dengan mempertimbangkan waktu setup. Sehingga, tujuan artikel ini adalah mengembangkan algoritma heuristik Pour untuk menyelesaikan permasalahan *hybrid flowshop three stage* dengan mempertimbangkan waktu setup. Permasalahan pada *hybrid flowshop* ini terdapat 1 mesin pada *stage* pertama, 2 mesin pada *stage* 2 dan 1 mesin pada *stage* 3.

II. METODE PENELITIAN

Definisi Masalah

Dalam permasalahan *hybrid flowshop with seperated setup time* ini, terdapat n *job* yang akan diproses melalui 3 *stage.* Semua *job* akan diproses dengan urutan yang sama pada *stage* 1, 2 dan terkadang ada urutan yang berbeda pada *stage* 3. Permasalahan *hybrid flowshop with seperated setup time* ini termasuk ke dalam kategori NP (*non-polinomial problem*). Tujuan dari penjadwalan *hybrid flowshop with seperated setup time* ini adalah untuk mengurutkan urutan pengerjaan *job* yang akan menghasilkan *makespan* minimum. Skema permasalahan *hybrid flowshop three stage* dapat dilihat pada Gambar 1.



Gambar 1. Contoh skema permasalahan *hybrid flowshop three stage*

Asumsi

Beberapa batasan pada permasalahan *hybrid flowshop with seperated setup time* adalah: (1) Setiap *job* akan mengalami proses setup,

(2) Urutan pada *stage* 1 dan *stage* 2 selalu sama, (3) Setiap mesin hanya dapat memproses 1 item ada waktu yang sama, (4) Tidak adanya *interupsi job,* (5) Setiap mesin akan memulai proses pada t=0, (6) Waktu setup terpisah dengan waktu proses, (7) *Job* ke-u akan diproses ketika mesin ke-m selesai memproses *job* ke-(u-1).

Formula *completation time hybrid flowshop* with *setup time* dapat dilihat pada persamaan 1-10.

$C_{1,1} = t_{1,1} + s_{1,1}$ ....(1)

$C_{u,1} = C_{u-1,1} + s_{u,1} + t_{u,1}$; u=2,...n ....(2)

$C_{1,2} = C_{1,1} + s_{1,2} + t_{1,2}$ ....(3)

$C_{2,3} = C_{2,1} + s_{2,3} + t_{2,3}$ ....(4)

$C_{u,2/3} = C_{u-1,1} + s_{u,2/3} + t_{u,2/3}$; u=3,...n ....(5)

$C_{1,4} = C_{1,2} + s_{1,4} + t_{1,4})$ ....(6)

$C_{u,4}) = C_{u-1,2/3} + s_{u,4} + t_{u,4}$; u=2,...n ....(7)

makespan = max ($C_{u,4}$) ....(8)

Notasi dari *hybrid flow shop three stage* adalah sebagai berikut:

u : indeks dari pekerjaan, $u = 1,2 \ldots, n$.

k : indeks dari mesin, $k = 1,2,3,4$.

n : jumlah pekerjaan.

$t_{u,k}$: waktu proses pekerjaan u pada mesin k.

$s_{u,k}$: waktu setup pekerjaan u pada mesin k.

Wtu : waktu total operasi pekerjaan u.

Cu,k : waktu penyelesaian job u pada mesin k.

Persamaan (1) digunakan untuk melakukan perhitungan waktu selesai *job* 1 pada mesin 1. Persamaan (2) digunakan untuk menghitung waktu selesai *job* ke-2 sampai ke-n pada mesin 1. Persamaan (3) digunakan untuk menghitung waktu selesai *job* ke-1 di mesin 2. Persamaan (4) digunakan untuk menghitung waktu selesai *job* ke-2 di mesin 3. Persamaan (5) digunakan untuk menghitung waktu selesai *job* ke-3 sampai ke-n pada mesin 2 atau 3. Pekerjaan ke-3 sampai ke-n ditugaskan hanya di salah satu mesin 2 atau 3. Persamaan (6) digunakan untuk menghitung waktu selesai *job* ke-1 pada mesin 4. Persamaan (7) digunakan untuk melakukan perhitungan *job* ke-2 sampai ke-n pada mesin 4. Persamaan (9) digunakan untuk melakukan perhitungan nilai *makespan*.

Pengembangan Algoritma Usulan

Pengembangan algoritma heuristik Pour yang telah dilakukan terletak pada perhitungan nilai *increasing time*. Jika pada algoritma *pure* Pour, nilai *increasing time* didapat dengan menjumlahkan secara kumulatif waktu proses terendah hingga terbesar. Namun, pada algoritma usulan nilai *increasing time* didapat dengan melakukan penjumlahan kumulatif dari total waktu setup dan waktu proses. Adapun tahap-tahap pengerjaan algoritma usulan sebagai berikut:

a. Menentukan *job* secara random sebagai urutan awal (berlaku untuk semua *job*).
b. Melakukan penjumlahan waktu proses dan waktu setup pada Persamaan (9).
c. $Wtu = \sum_{k=1}^{3} t_{u,k} + s_{u,k}$ ....(9)
d. Menghitung *increasing time (It)* pada *wt* dengan mengkumulatifkan waktu terkecil hingga terbesar.
e. Menghitung *sum increasing time* (SIC) untuk setiap *job* yang ada.
f. Mengurutkan SIC dari waktu terkecil hingga terbesar sebagai jadwal urutan sementara.
g. Setelah didapatkan urutan sementara, maka hitunglah total waktu penyelesaian seluruh *job*.
h. Ulangi langkah 1-6 untuk setiap *job* yang ada sampai didapatkan *makespan* paling minimal, yang akan ditempatkan sebagai urutan pertama dari urutan *job*.
i. Ulangi langkah 1-6 untuk mencari urutan *job* ke-2 sampai ke-n dengan nilai *makespan* paling minimal.

III. HASIL DAN PEMBAHASAN

Contoh Perhitungan Algoritma Usulan

Data contoh perhitungan kasus dengan 4 *job* 3 *stage* dapat dilihat pada Tabel 1. Pada *stage* 2 terdapat 2 buah mesin identik, namun

Tabel 1. Data contoh perhitungan

Job	*Stage* 1		*Stage* 2		*Stage* 3	
	setup	proses	*setup*	proses	*setup*	proses
1	3	2	1	18	1	5
2	2	5	2	15	2	3
3	3	2	2	16	1	5
4	2	3	1	15	2	3

pada *stage* 1 dan *stage* 3 hanya memiliki 1 buah mesin.

Langkah 1 :

Pada langkah ke-1 tambahkan *setup time* dan *processing time* pada setiap *job.* Hasil perhitungan dapat dilihat pada Tabel 2.

Tabel 2. Total waktu setup dan waktu proses

job/stage	1	2	3
1	5	19	6
2	7	17	5
3	5	18	6
4	5	16	5

Langkah 2 :

Pada langkah ke-2 lakukan perhitungan *increasing time* pada setiap *stage* dengan menganggap waktu pada *job* pertama sama dengan nol. Hasil perhitungan *increasing time* per *stage* dapat dilihat pada Tabel 3.

Tabel 3. Hasil perhitungan *increasing time*

job/stage	1	2	3
1	0	0	0
2	17	33	5
3	5	51	16
4	10	16	10

Langkah 3 :

Pada langkah ke-3 hitung total waktu *increasing time* pada setiap *job* dan urutan dari yang terkecil hingga yang terbesar. Hasil perhitungan dapat dilihat pada Tabel 4.

Tabel 4. *Total increasing time per job*

job	jumlah
1	0
2	55
3	72
4	36

Berdasarkan Tabel 5 maka didapatkan urutan pengerjaan *job* dengan urutan *job* 1 – *job* 4 – *job* 2 – *job* 3 dengan hasil *makespan* sebesar 49 menit.

Langkah 4 :

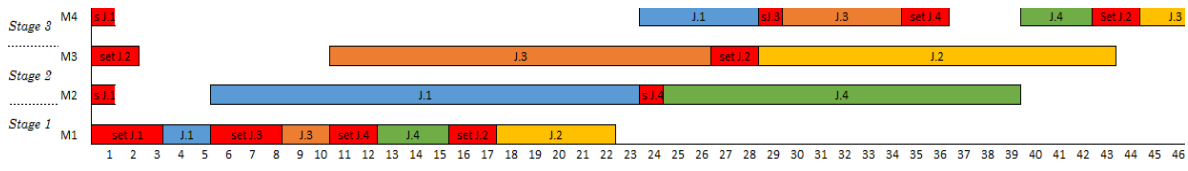
Lakukan pengulangan dari langkah 2 sampai langkah 3 dengan *job* 2 dan seterusnya yang diasumsikan nol. Rekapitulasi pengulangan langkah 2 sampai langkah 3 dapat dilihat pada Tabel 5.

Tabel 5. Rekapitulasi solusi alternatif algoritma usulan

Urutan ke	Urutan *job*	*Makespan*
1	*job* 1 - *job* 4 - *job* 2 - *job* 3	49
	job 2 - *job* 4 - *job* 3 - *job* 1	51
	job 3 - *job* 4 - *job* 2 – *job* 1	49
	job 4 - *job*2 - *job*3 - *job* 1	51
2	*job* 1 - *job* 4 - *job* 2 – *job* 3	49
	job 1 – *job* 2 - *job* 4 – *job* 3	50
	job 1 - *job* 3 – *job* 4 – *job* 2	47
3	*job* 1 – *job* 3 – *job* 4 – *job* 2	47
	job 1 – *job* 3 – *job* 2 – *job* 4	48

Tabel 6. Hasil *makespan* uji performansi

Experiment	Jumlah *Job*	*Makespan*		
		Pour	DFFP	FCFS
1	5	64	65	67
2	10	130	131	135
3	15	181	182	184
4	20	193	197	205
5	25	302	302	309
6	30	373	382	379
7	35	434	439	443
8	40	522	525	530
9	45	539	543	550
10	50	627	628	630



Gambar 2. *Gantt chart* urutan terbaik algoritma usulan

Dari Tabel 6 dapat diketahui *makespan* terbaik sebesar 47 menit dengan urutan *job* 1 – *job* 3 – *job* 4 – *job* 2. *Gantt chart* urutan terbaik dari algoitma usulan terlihat pada Gambar 2.

Uji Performansi

Untuk memperkuat algoritma usulan penulis, maka penulis melakukan 10 kali percobaan pada permasalahan *three stage* dengan *setup time* dengan jumlah *job* yang berbeda-beda. Data acak *processing time* dengan distribusi Uniform (2:6) dan data acak *setup time* dengan distribusi Uniform (1:2) pada mesin 1, data acak *processing time* dengan distribusi Uniform (15;20) dan data acak *setup time* dengan distribusi Uniform (5:8) pada mesin 2, dan data acak *processing time* dengan distribusi Uniform (4:8) dan data acak *setup time* dengan distribusi Uniform (3:5) pada mesin 3. Hasil perhitungan *makespan* dapat dilihat pada Tabel 6 dan waktu komputasi pada Tabel 7.

Tabel 7. Waktu komputasi uji performansi

Exp	jumlah *job*	*makespan*		
		Pour	DFFP	FCFS
1	5	0,044	0,015	0
2	10	0,146	0,046	0
3	15	0,406	0,031	0
4	20	0,546	0,031	0
5	25	1,109	0,046	0
6	30	1,350	0,031	0
7	35	2,030	0,015	0
8	40	2,900	0,031	0
9	45	3,290	0,190	0
10	50	5,180	0,031	0

Setelah mendapatkan hasil *makespan* untuk masing-masing *numerical experiment* yang dilakukan maka akan dilakukan pengukuran *performance* dar ialgoritma usulan melalui perhitungan *efficiency index* (EI) dan *relative error* (RE) untuk mengetahui apakah algoritma usulan lebih baik dari algoritma lainnya. Tabel 8 dan Tabel 9 merupakan hasil perhitungan *efficiency index* (EI) dan *relative error* (RE).

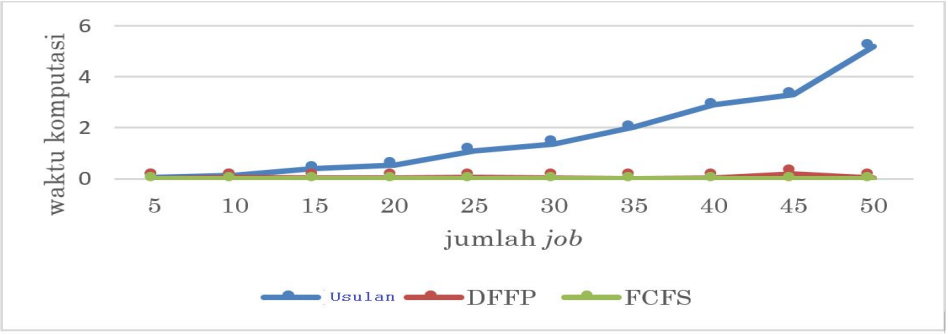
Dari Tabel 8 dapat diketahui bahwa rata-rata *efficiency index* algoritma usulan dengan algoritma DFFP adalah sebesar 0,99. Karena nilai tersebut < 1, maka performansi algoritma usulan lebih baik dari algoritma DFFP. Rata-rata *efficiency index* algoritma usulan dengan algoritma FCFS (*First Come First* Serve) adalah sebesar 0,97. Karena nilai tersebut <1, maka performansi algoritma usulan lebih baik dari algoritma FCFS (*First Come First Serve*). Dari Tabel 9 dapat diketahui bahwa rata-rata *relative error* algoritma usulan dengan algoritma DFFP adalah sebesar 1%. Karena nilai tersebut >0, maka algoritma usulan lebih baik dari algoritma DFFP. Rata-rata *relative error* algoritma usulan dengan algoritma FCFS (*First Come First* Serve) adalah sebesar 2%. Karena nilai tersebut >0, maka algoritma usulan lebih baik dari algoritma FCFS (*First Come First Serve*). Berdasarkan hasil yang diperoleh dari perhitungan *efficiency index* (EI) dan *relative error* (RE), secara keseluruhan dapat dikatakan bahwa algoritma usulan memiliki performansi yang lebih baik dibanding algoritma DFFP dan algritma FCFS (*First Come First Serve*) karena algoritma usulan dapat memberikan ruang solusi alternatif yang lebih banyak.

Tabel 8. Perhitungan *efficiency index*

Numerical experimant	*Efficientcy index*	
	Algoritma usulan dengan DFFP	Algoritma usulan dengan FCFS
5 *job*	0,98	0,95
10 *job*	0,99	0,96
15 *job*	0,99	0,98
20 *job*	0,98	0,94
25 *job*	1,00	0,97
30 *job*	0,97	0,98
35 *job*	0,98	0,98
40 *job*	0,99	0,98
45 *job*	0,99	0,98
50 *job*	0,99	0,99
Rata-rata	0,99	0,97

Tabel 9. Perhitungan *relative error*

Numerical experiment	*Relative error*	
	Algoritma usulan dengan DFFP	Algoritma usulan dengan FCFS
5 *job*	2%	5%
10 *job*	1%	4%
15 *job*	2%	2%
20 *job*	7%	6%
25 *job*	0%	2%
30 *job*	2%	2%
35 *job*	0%	1%
40 *job*	1%	2%
45 *job*	1%	2%
50 *job*	0%	0%
Rata-rata	1%	2%



Gambar 3. Grafik perbandingan waktu komputasi

Analisa Algoritma Usulan

Algoritma usulan adalah algoritma heuristik Pour yang telah dikembangkan. Hasil perbandingan waktu komputasi antara algoritma usulan dengan algoritma DFFP dan FCFS dapat dilihat pada Gambar 3. Gambar 3 menunjukan bahwa semakin besar job yang digunakan, maka membutuhkan waktu komputasi yang lebih besar. Algoritma ini memberikan ruang solusi yang besar sehingga membutuhkan waktu komputasi yang sedikit lebih lama. Jumlah ruang solusi yang dihasilkan akan berdampak juga pada hasil *makespan* yang didapat. Semakin banyak ruang solusi alternatif yang diciptakan maka peluang mendapatkan hasil yang lebih optimal akan lebih besar.

IV. KESIMPULAN

Pengembangan algoritma heuristik Pour adalah salah satu pendekatan baru yang berguna untuk menyelesaikan masalah pada *hybrid flowshop three-stage with separated setup time*. Pengembangan ini dilakukan untuk meminimasi fungsi tujuannya, yaitu minimasi *makespan*. Algoritma Pour adalah algoritma yang diciptakan oleh Pour pada tahun 2001 yang digunakan untuk menyelesaikan kasus *pure flowshop* dan tidak mempertimbangkan waktu setup. Saran untuk penelitian selanjutnya algoritma ini dapat dijadikan sebagai tahapan awal pada algoritma *metaheuristic*.

DAFTAR PUSTAKA

Abdel-Basset, M.; Manogaran, G.; El-Shahat, D.; Mirjalili, S. (2018). "A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem". *Future Generation Computer Systems, 85*, 129-145.

Aldowaisan, T.; Allahverdi, A. (2004). "New heuristics for m-machine no-wait flowshop to minimize total completion time". *Omega, 32* (5), 345-352.

Allahverdi, A.; Al-Anzi, F.S. (2006). "A branch-and-bound algorithm for three-machine flowshop scheduling problem to minimize total completion time with separate setup times". *European Journal of Operational Research, 169* (3), 767-780.

Allahverdi, A.; Aldowaisan, T. (2000). "No-wait and separate setup three-machine flowshop with total completion time criterion". *International Transactions in Operational Research, 7* (3), 245-264.

Ebrahimi, M.; Ghomi, S.F.; Karimi, B. (2014). "Hybrid flow shop scheduling with sequence dependent family setup time and uncertain due dates". *Applied Mathematical Modelling, 38* (9-10), 2490-2504.

Firdaus, M.; Masudin, I.; Utama, D.M. (2015). "Penjadwalan flowshop dengan menggunakan simulated annealing". *Spektrum Industri, 13* (1), 27-40.

Gómez-Gasquet, P.; Andrés, C.; Lario, F.-C. (2012). "An agent-based genetic algorithm for hybrid flowshops with sequence dependent setup times to minimise makespan". *Expert Systems with Applications, 39* (9), 8095-8107.

Harto, S.; Garside, A.K.; Utama, D.M. (2016). "Penjadwalan produksi menggunakan algoritma jadwal non delay untuk meminimalkan makespan studi kasus di CV. Bima Mebel". *Spektrum Industri, 14* (1), 79-88.

Husen, M.; Masudin, I.; Utama, D.M. (2015). "Penjadwalan job shop statik dengan metode simulated annealing untuk meminimasi waktu makespan". *Spektrum Industri, 13* (2), 115-124.

Lin, S.-W.; Gupta, J.N.; Ying, K.-C.; Lee, Z.-J. (2009). "Using simulated annealing to schedule a flowshop manufacturing cell with sequence-dependent family setup times". *International*

Journal of Production Research, 47 (12), 3205-3217.

Marichelvam, M.K.; Prabaharan, T.; Yang, X.S. (2014). "A discrete firefly algorithm for the multi-objective hybrid flowshop scheduling problems". *IEEE Transactions on Evolutionary Computation, 18* (2), 301-305.

Masudin, I.; Utama, D.M.; Susastro, F. (2014). "Penjadwalan flowshop menggunakan algoritma Nawaz enscore HAM". *Jurnal Ilmiah Teknik Industri, 13* (1), 54-59.

Nasution, R.; Garside, A.K.; Utama, D.M. (2017). "Penjadwalan job shop dengan pendekatan algoritma artificial immune system". *Jurnal Teknik Industri, 18* (1), 29-42.

Ong, J.O. (2013). "Penjadwalan non-delay melalui mesin majemuk untuk meminimumkan makespan". *Spektrum Industri, 11* (2), 185-195.

Pour, H.D. (2001). "A new heuristic for the n-job, m-machine flow-shop problem". *Production Planning & Control, 12* (7), 648-653.

Proust, C.; Gupta, J.; Deschamps, V. (1991). "Flowshop scheduling with set-up, processing and removal times separated". *The International Journal Of Production Research, 29* (3), 479-493.

Ruiz, R.; Stützle, T. (2008). "An iterated greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives". *European Journal of Operational Research, 187* (3), 1143-1159.

Soetanto, T.V.; Palit, H.C.; Munika, I. (2005). "Studi perbandingan performance algoritma heuristik Pour terhadap mixed integer programming dalam menyelesaikan penjadwalan flowshop". *Jurnal Teknik Industri, 6* (1), 79-85.

Sulaksmi, A.; Garside, A.K.; Hadziqah, F. (2014). "Penjadwalan produksi dengan algoritma heuristik Pour (Studi kasus: Konveksi One Way–Malang)". *Jurnal Teknik Industri, 15* (1), 35-44.

Utama, D.M. (2018). *Pengembangan Algoritma NEH Dan CDS Untuk Meminimasi Consumption Energy Pada Penjadwalan Flow Shop*. Paper presented at the Prosiding SENTRA (Seminar Teknologi dan Rekayasa), Malang.

Utama, D.M. (2019). "An effective hybrid sine cosine algorithm to minimize carbon emission on flow-shop scheduling sequence dependent setup". *Jurnal Teknik Industri, 20* (1), 10.

Utama, D.M.; Widodo, D.S.; Wicaksono, W.; Ardiansyah, L.R. (2019). "Metaheuristics algorithm for minimizing energy consumption in the flow shop scheduling problem". *International Journal of Technology, 10* (2), 320-331.

Widodo, D.S. (2017). "Pengembangan Cross Entropy-Genetic Algorithm (CEGA) pada penjadwalan model flow shop untuk meminimalkan makespan". *Teknoterap, 1* (1), 1–11.

Ying, K.-C.; Lin, S.-W. (2018). "Minimizing makespan for no-wait flowshop scheduling problems with setup times". *Computers & Industrial Engineering, 121*, 73-81.

Zandieh, M.; Karimi, N. (2011). "An adaptive multi-population genetic algorithm to solve the multi-objective group scheduling problem in hybrid flexible flowshop with sequence-dependent setup times". *Journal of Intelligent Manufacturing, 22* (6), 979-989.