Available *online* at : http://josi.ft.unand.ac.id/

Jurnal Optimasi Sistem Industri

| ISSN (Print) 2088-4842 | ISSN (Online) 2442-8795 |

Studi Kasus

Penjadwalan Flow shop Untuk Meminimasi *Total Tardiness* Menggunakan Algoritma *Cross Entropy*–Algoritma Genetika

Dana Marsetiya Utama, Leo Rizki Ardiansyah, Annisa Kesy Garside

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

ARTICLE INFORMATION

Received: March 16, 19
Revised: October 29, 19
Available online: October 31, 19

KEYWORDS

Scheduling, total tardiness, flow shop, cross entropy, genetic algorithm.

CORRESPONDENCE

Phone: +6282132044774
E-mail: dana@umm.ac.id

A B S T R A C T

Flow shop scheduling problems much studied by several researchers. One problem with scheduling is the tardiness. Total tardiness is the performance to minimize tardiness jobs. it is the right performance if there is a due date. This study proposes the Cross-Entropy Genetic Algorithm (CEGA) method to minimize the mean tardiness in the flow shop problem. In some literature, the CEGA algorithm is used in the case of minimizing the makespan. However, CEGA not used in the case of minimizing total tardiness. CEGA algorithm is a combination of the Cross-Entropy Algorithm which has a function to provide optimal sampling distribution and Genetic Algorithms that have functions to get new solutions. In some numeric experiments, the proposed algorithm provides better performance than some algorithms. For computing time, it is affected by the number of iterations. The higher the iteration, computing requires high time.

PENDAHULUAN

Penjadwalan memiliki kaitan erat dengan penentuan urutan proses pengerjaan [1]. Ukuran kinerja penjadwalan bertujuan mendapatkan urutan proses pengerjaan yang lebih baik [2]. Masalah dalam penjadwalan terjadi karena keterbatasan fasilitas dan jumlah mesin. Akibatnya, job mengalami antrian saat akan diproses karena mesin dalam keadaan sibuk [3-5]. Penjadwalan merupakan aktivitas perencanaan produksi untuk mengatur waktu dan sumberdaya [6-8]. Penjadwalan selalu mempertimbangkan tenaga kerja, peralatan produksi dan fasilitas [9]. Pada penjadwalan *pure flowshop*, *job* akan diproses pada setiap mesin dengan urutan proses yang sama. Penjadwalan termasuk dalam kasus *NP-hard* [10, 11]. Kasus *NP-hard* ini menggambarkan semakin besar problem maka waktu komputasi akan semakin lama. *Flowshop scheduling problem* merupakan kasus job yang memiliki proses yang sama [12].

Mean tardiness merupakan performasi penjadwalan untuk meminimasi keterlambatan [13]. Algoritma heuristik yang populer untuk masalah penjadwalan *flow shop* adalah NEH [14] dan CDS [15]. Beberapa Algoritma metaheuristik yang populer adalah *Simulated Annealing* (SA) [16,17], Algoritma Genetika (GA), dan *Immune system* [18]. Pada beberapa penelitian terdahulu telah diulas tentang maslah penjadwalan *flow shop*. Li *et al*. [19], Onwubolu dan Mutingi [20], Min dan Cheng [21], Hamidinia *et al.* [22], Etiler *et al*. [23], Le Riche dan Haftka [24], dan Gunawan [25] menggunakan GA. Mereka menggunakan fungsi tujuan meminimasi *tardiness*. Hasil penelitian mereka menunjukan GA efektif dan efisien cocok untuk menyelesaikan masalah penjadwalan dalam meminimasi *tardiness.* Saputro and Yento [26] juga menggunakan GA untuk menyelesaikan penjadwalan non deterministik. Pada penelitian lain, GA digunakan untuk menyelesaikan penjadwalan *hybrid flow shop and job shop* untuk meminimasi *tardiness and makespan* [27, 28]. GA dapat digunakan pada penjadwalan *multi-processor* dengan tujuan untuk meminimasi *completion time* [29]. Menurut De Boer *et al.* [30], algoritma *Cross Entropy (CE)* merupakan prosedur pemecahan permasalahan stokastik untuk memberikan solusi yang optimal. Pada penelitian lain, CE memberikan hasil yang optimal pada penjadwalan yang memiliki keterbatasan sumber daya [31]. Dalam penelitian Caserta *et al*. [32], CE dinilai efektif dalam menyelesaikan produksi *knapsack* dengan menambahkan waktu setup. CE juga dapat digunakan dalam memecahkan masalah *hybrid multi item* dengan waktu setup [33].

Menurut Buliali *et al*. [34], GA sangat efektif dalam penjadwalan. GA dapat memberikan solusi yang lebih baik dari sebelumnya. Pada penelitian yang dilakukan oleh Widodo [9], Bashori [35], dan Bashori *et al*. [36], algoritma *Cross Entropy Genetic Algorithm* (CEGA) digunakan untuk meminimasi *makespan.*

DOI: 10.25077/josi.v18.n2.p133-141.2019

CEGA dinilai efektif karena bisa melakukan minimasi waktu pengerjaan. Pada penelitian lain, metode CEGA lebih baik daripada metode *Genetic Algorithm-Simulated Annealing* (GASA) dibuktikan dengan *makespan* yang paling kecil [37,38]. Menurut Nurkhalida and Santosa [39], Hanka and Santosa [40], serta Widyadana and Pamungkas [41], CEGA dapat menghasilkan solusi yang kompetitif dibandingkan dengan metode CE murni maupun *Simulated Annealing* terutama untuk ukuran permasalahan kecil dan sedang. Menurut Rahmawati and Santosa [42] dan Widodo *et al*. [43], kualitas solusi yang dihasilkan CEGA menghasilkan kinerja yang lebih baik.

Beberapa penelitian terdahulu telah banyak yang menggunakan algoritma CEGA, namun, fokus penelitian tersebut adalah minimasi *makespan.* Saat ini, belum ada penelitian untuk meminimasi total *tardiness* menggunakan *CEGA.* Tujuan penelitian ini adalah menggunakan algoritma CEGA pada pola aliran *pure flow shop* untuk meminimasi total keterlambatan (*total tardiness*). Tujuan penelitian ini juga mengetahui parameter terbaik dalam meminimasi total *tardiness* menggunakan metode CEGA. Algoritma CEGA diharapkan mampu untuk memberikan solusi yang lebih baik dalam meminimasi total *tardiness.* Sehingga algoritma CEGA dapat digunakan sebagai algoritma untuk meminimasi total *tardiness* pada kasus *pure flow shop.*

METODE

Penelitian ini menggunakan integrasi algoritma CEGA. GA merupakan algoritma populer yang digunakan para peneliti. Algoritma Genetika diinspirasi dari yang memanfaatkan proses seleksi alamiah yang dikenal dengan proses evolusi. Dalam proses evolusi, individu secara terus-menerus mengalami perubahan gen. Proses evolusi yang dimaksud adalah proses mutasi dan cross over. CE memiliki fungsi memberikan distribusi sampling yang optimal. GA memiliki fungsi untuk mendapatkan solusi baru dengan menambahkan update parameter pindah silang, *cross over*, dan mutasi.

Definisi Masalah

Permasalahan pada penjadwalan *flow shop* ini adalah keterlambatan penyelesaian pekerjaan. Untuk penyelesaian pekerjaan dikerjakan pada urutan mesin yang sama. Beberapa asumsi pada penelitian ini ditetapkan sebagai berikut:

a) Setiap job_z hanya bisa diproses satu kali pada setiap mesin tanpa ada pengulangan proses, dimana z = 1, ...,n
b) Setiap mesin M_x hanya bisa memproses satu *job* dalam satu waktu, dimana x = 1, ...,m
c) Setiap *job* membutuhkan waktu proses pada setiap mesin
d) Urutan pengerjaan *job* diproses pada mesin dengan urutan yang sama
e) Disetiap mesin hanya memproses satu job disetiap produksinya
f) Tidak terjadi interupsi job
g) Saat melakukan proses produksi, mesin dimulai pada waktu ke-0
h) Jobz akan dikerjakan ketika mesin ke x selesai memproses job (z-1)

Formula yang digunakan dalam permasalahan *flow shop* sebagai berikut:

$$C(j_1,M_1) = t(j_1,M_1) \quad (1)$$
$$C(j_1,M_x) = C(j_1,M_{x-1}) + t(j_1,M_x),\ x = 2, ...,m \quad (2)$$
$$C(j_z,M_1) = C(j_{z-1},Mx) + t(j_z,M_1),\ z = 2, ...,n \quad (3)$$
$$C(j_z,M_x) = \text{Max}\ ((C(j_{z-1},M_x),(C(j_z,M_{x-1})) + t(j_z,M_x),\ z = 2, ...,n ,\ x = 2, ...,m. \quad (4)$$
$$Tj_z = \text{Max}\{0,(C(j_z,M_x) - d)\} \quad (5)$$
$$TT = \sum_{z=1}^{n} Tjz \quad (6)$$

Formula *flow shop* dijelaskan sebagai berikut. Persamaan (1) menunjukan *Completion time job* urutan ke-1 pada mesin ke-1. Completion time ini dihitung dari waktu proses pada *job* urutan ke-1 pada mesin ke-1. Persamaan (2) menunjukan *Completion time job* urutan ke-1 pada mesin ke-x. Persamaan ini dihitung dengan menjumlahkan *completion time job* ke-1 pada mesin sebelumnya dan waktu proses *job* urutan ke-1 pada mesin ke-x. Persamaan (3) menunjukan *Completion time job* urutan ke-z pada mesin ke-1. Persamaan ini dihitung dengan menjumlahkan *completion time job* pada urutan sebelumnya dimesin ke-1 dan waktu proses *job* urutan ke-z pada mesin ke-1. Persamaan (4) menunjukan *Completion time job* urutan ke-z pada mesin ke-x. Persamaan ini dihitung dengan mempertimbangkan nilai maksimum *completion time job* diurutan sebelumnya dimesin ke-x dengan *completion time job* diurutan z dimesin sebelumnya. persamaan ini dijumlahkan dengan waktu proses *job* urutan ke-z pada mesin ke-x. Persamaan (5) menunjukan *Tardiness job* pada urutan ke-z. jika *completion time job* urutan ke-z pada mesin ke-x dikurangi dengan d hasilnya sama dengan kurang dari 0, maka tidak terjadi keterlambatan. Dan apabila hasilnya lebih dari 0 maka terjadi keterlambatan. Dan Persamaan (6) menunjukan *Total Tardiness.* Persamaan ini dihitung dengan menjumlahkan hasil dari *tardiness job* ke-z

Cross Entropy

Metode CE memiliki tujuan untuk menghasilkan urutan kearah solusi yang optimal dari hasil iterasi. Metode CE awalnya diterapkan untuk simulasi kejadian langka (*rare-event*), lalu dikembangkan untuk beberapa kasus seperti optimasi kombinatorial, optimasi kontinu, *machine leaarning*, dan beberapa kasus lain [38]. Metode CE termasuk dalam keluarga teknik Monte Carlo yang bisa digunakan untuk menyelesaikan kasus estimasi maupun optimasi. Cara kerja dari algoritma CE adalah bagaimana mengambil sampel random untuk mendapatkan urutan dari solusi yang bagus. Dengan distribusi ini akan di-generate kandidat solusi baru. Distribusi ini akan terus di-update berdasarkan kandidat solusi yang lebih baik. Algoritma akan terus mengulang skenario yang sama hingga distribusi sampel mengarah pada area solusi optimal [39].

Metode CE melibatkan prosedur iterasi, dimana tiap iterasi dapat dipecah menjadi dua fase: (1) Melakukan pembangkitan sampel random(x) dengan menggunakan mekanisme atau distribusi tertentu. (2) Memperbaharui parameter (ν) dari mekanisme random berdasarkan data sampel elite untuk menghasilkan sampel yang lebih baik pada iterasi berikutnya. Sampel elite adalah berapa persen dari sampel yang kita pilih untuk memperbaiki atau mengupdate parameter ν yang digunakan. Proses tersebut secara matematis dapat ditulis sebagai berikut [30]:

1. Tentukan nilai N, yaitu banyaknya sampel, ν0, ρ dan α.
2. Bangkitkan sampel sebanyak N dengan mekanisme tertentu, memanfaatkan parameter ν0.
3. Evaluasi sampel ini dengan memasukkan ke dalam fungsi tujuan. Lalu urutkan nilai fungsi tujuan. Ini dilakukan dengan cara memasukan nilai x yang dibangkitkan ke dalam fungsi tujuan f (x). Jika ada N sampel maka didapatkan sebanyak N nilai f. Lalu kita urutkan nilai f ini dari yang terbesar ke yang terkecil (untuk kasus maksimasi lakukan sebaliknya). Pilih 1 − ρ persentil dari N sampel x yang menghasilkan nilai f

DOI: 10.25077/josi.v18.n2.p133-141.2019

terkecil. Ini bisa dijelaskan, misalnya N =10, ρ =0.1, maka 1 − 0.1=0.9 persentil dari 10 sampel adalah titik sampel ke 9 dan ke 10 setelah nilai f diurutkan. Kedua sampel ini yang disebut sampel elite. Ingat bahwa yang kita perlukan adalah nilai x-nya bukan nilai f-nya. Nilai f digunakan untuk mencari nilai x mana yang memberikan f terkecil. Sejumlah nilai x terbaik ini gunakan untuk mengupdate parameter ν.

4. Memperbaharui γt secara adaptif.
5. Untuk ν yang sudah diupdate, gunakan untuk membangkitkan nilai x yang baru. Kemudian masukkan ke dalam γt dan νt−1 yang telah ditetapkan.

Update parameter vektor ν bisa dilakukan dengan formula $\hat{v}_t = \alpha\hat{v}_t + (1-\alpha)\hat{v}_{t-1}$. Untuk memperbaiki performansi generasi dari νt, diperkenalkan parameter *smoothing*. Parameter smoothing (α) nilai nya berkisar antara $0 \leq \alpha \leq 1$. Kecenderungan CE apabila terdapat dua atau lebih solusi optimal akan berfluktuasi antara solusi sebelum fokus pada satu titik solusi. Penggunaan α yang tepat diharapkan dapat membantu algoritma CE agar lebih cepat mencapai konvergensi pada solusi optimal. Penggunaan α tampak dalam persamaan berikut:

1. Algoritma utama CE untuk optimasi
2. Tentukan parameter awal ν = u, α dan ρ.Tetapkan iterasi = 1.
3. Bangkitkan sampel random X1,..., XN dari fungsi probabilitas distribusi tertentu f (−; u) dan pilih sampel (1−ρ) *quantile* dari performansi setelah diurutkan.
4. Gunakan sampel yang sama untuk memperbarui nilai parameter
5. Aplikasikan persamaan 2.3. untuk memuluskan vektor ν = u.Kembali ke langkah 2 dengan nilai parameter yang baru, tetapkan it = it +1.
6. Jika *stopping criteria* sudah dipenuhi, berhenti.
7. Perlu dicatat bahwa *stopping criteria*, vektor solusi awal, ukuran sampel N dan nilai ρ harus dinyatakan secara spesifik dari awal iterasi. Parameter ν di-*update* hanya berdasarkan sejumlah (1 − ρ) sampel terbaik. Sampel yang digunakan untuk mengupdate parameter ini dinamakan sampel elite.

Genetic Algorithm

Konsep GA lahir berdasarkan teori evolusi Darwin. Konsep tersebut menggambarkan seleksi alam dimana yang kuat maka menang. Konsep evolusi biologis ini pertama kali dicetuskan oleh Rechenberg dalam salah satu penelitiannya. GA secara resmi diperkenalkan di Amerika Serikat pada tahun 1975 oleh John Holland di Universitas Michigan. Perkembangan performansi dari sistem komputasi yang terus berkelanjutan membuat konsep ini menarik untuk beberapa jenis optimasi. Dengan mempertahankan daftar solusi yang menjanjikan pada setiap tahap, dan iterasi algoritmik bertujuan untuk menghasilkan yang lebih baik dengan mencari jenis lingkungan khusus, sehingga algoritma ini akan menyimpan solusi terbaiknya agar tidak hilang selama proses iterasi berlangsung.

GA menggabungkan dua urutan terpilih. Pada prinsipnya, suatu GA dapat menggabungkan lebih dari dua urutan yang ada, karena kandidat baru dapat dilihat sebagai keturunan dari yang sudah ada, terminologi tersebut dipastikan berasal dari evolusi dan genetika. Jadi, di setiap generasi, kita mulai dengan para orangtua (urutan), yang merupakan pembawa gen terbaik dari generasi sebelumnya. Pasangan orang tua dipilih biasanya secara acak untuk menghasilkan keturunan. Setiap orangtua memberikan kontribusi gen (*subsequences*) kepada keturunannya, dan mutasi (perubahan acak) juga dapat terjadi. Algoritma genetika berakhir setelah jumlah generasi yang ditentukan, tetapi aturan penghentian lainnya dapat diberlakukan. Gen yang terkuat dipilih sebagai solusi [3].

Menurut Rao *et al.* [44], prosedur aplikasi GA pada kasus optimasi adalah sebagai berikut :

1. Tentukan parameter awal berupa ukuran populasi *m*, parameter *crossover* P_{ps}, parameter mutasi P_m, nilai standar deviasi yang diijinkan $(s_f)_{max}$ sebagai kriteria konvergensi, dan jumlah generasi/iterasi maksimum (i_{max}) sebagai kriteria konvergensi kedua.
2. Bangkitkan sebanyak *m random* populasi, masing-masing terdiri dari kromosom dengan panjang $l = nq$. Evaluasi nilai *fitness* F_i untuk semua kromosom.
3. Lakukan tahap reproduksi
4. Lakukan operasi *crossover* mengacu pada parameter *crossover* P_{ps}
5. Lakukan operasi mutasi dengan parameter mutasi P_m
6. Evaluasi nilai F_i, i = 1,2, ..., m dari populasi yang baru. Tentukan standar deviasi dari nilai *fitness m* kromosom.

Periksa status konvergensi algoritma maupun proses. Jika $s_f \leq (s_f)_{\max}$ maka kriteria konvergensi tercapai dan proses dapat dihentikan. Jika sebaliknya, maka lanjutkan ke langkah Periksa jumlah generation. Jika $i \geq i_{\max}$ maka proses dapat dihentikan. Jika sebaliknya maka set $i = i$+1 dan teruskan ke langkah 3.

Metode GA merupakan metode penggabungan dari yang didasarkan pada mekanisme seleksi alam dan genetika alam untuk mendapatkan ruang solusi optimal. Prosedur metode GA ditunjukkan pada Gambar 1: Menentukan nilai parameter inisial yang meliputi: jumlah populasi (N), parameter pindah salah (P_{ps}) dan parameter mutasi (P_{mt}). Berikut penj lasan mengenai penentuan parameter inisial: Jumlah populasi (N) pada algoritma ini berupa urutan prioritas dari semua jon yang akan dijadwalkan. Semakin banyak jumlah *job*, maka semakin banyak jumlah populasi awal yang harus dibangkitkan. Nilai inisial untuk parameter pindah silang (P_{ps}). Nilai inisial untuk parameter mutasi (P_{mt}).

Membangkitkan populasi awal secara acak sebanyak N kromosom. Dimana kromosom merupakan urutan *job* sepanjang n. Mengevaluasi semua kromosom dalam populasi dengan menghitung nilai *fitness* sesuai fungsi tujuan. Kromosom dengan nilai *fitness* terkecil akan disimpan dalam solusi terbaik sementara. Tahap selanjutnya yaitu menyeleksi kromosom-kromosom dengan menggunakan mekanisme *Roullette Wheel (rw)*. Dengan membangkitkan bilangan acak sebanyak N dimana $w \in [0,1]$. Untuk menentukkan kromosom yang terpilih menjadi induk yaitu dengan menggunakan persamaan 7 dan 8.

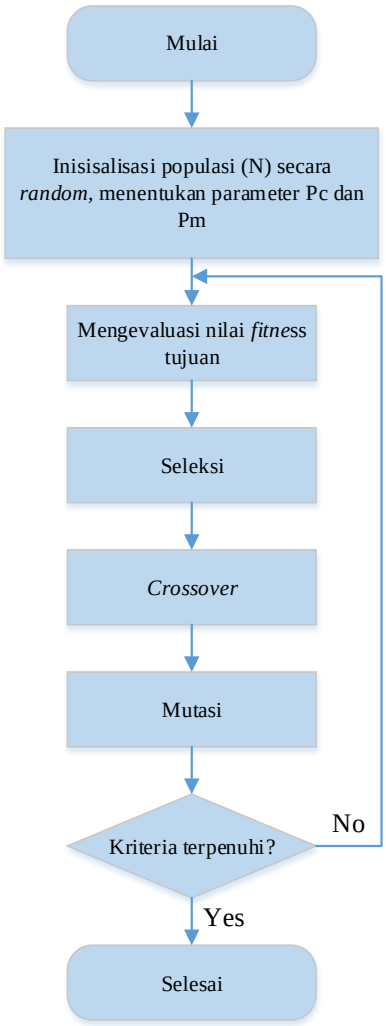
$$c_1 > rw_j \tag{7}$$

selanjutnya,

$$c_{j-1} < rw_j < c_j \tag{8}$$

Crossover atau pindah silang (kawin silang), dilakukan dengan melibatkan persilangan sepasang kromosom induk untuk membentuk kromosom baru/anak (*offspring*). Kemungkinan suatu kromosom mengelami proses *crossover* berdasarkan dari probabilitas *crossover* (Pc). Dalam proses *crossover* akan terjadi pertukaran sebagian gen dari dua kromosom induk. Dengan harapan mendapatkan kromosom anak dengan komposisi gen

yang baik. Metode yang digunakan adalah metoda 2-*point order cross over*.



Gambar 1. Prosedur *Genetic Algorithm* [42]

Mutasi dimaksudkan untuk memunculkan individu baru yang berbeda dengan individu yang sudah ada. Mutasi dilakukan dengan menggunakan tiga mekanisme, yaitu *swap mutation* (menukar), *flip mutation* (membalik), dan *slide mutation* (menggeser). Di dalam satu populasi akan terdapat *Pm x N* populasi yang dimutasi. Untuk parameter mutasi (*Pm*) bernilai setengah dari nilai parameter pindah silang.

Algoritma Cross Entropy Genetic Algorithm

Menurut Widodo [9], algoritma CEGA merupakan penggabungan antara algoritma CE dan GA. Fungsi dari algoritma CE adalah untuk memberikan distribusi sampling yang optimal yang terus di*update* sampai menemukan solusi otimal. Fungsi dari algoritma GA adalah menghindari solusi terjebak sehingga solusi terbaik akan terus dimunculkan pada setiap iterasi agar mendapatkan solusi baru yang optimal. Algoritma CEGA memiliki 12 tahapan. Langkah-langkah penjadwalan menggunakan metode CEGA. Prosedur algoritma CEGA dapat dilihat pada Tabel 1.

Tabel 1. Prosedur Algoritma CEGA

	Prosedur CEGA
1	Tentukan sampel yang akan dibangkitkan (N), rho, alpha, P_ps, dan kriteria ε
2	Pengacakan sampel secara keseluruhan sebanyak (N)
3	Hitung nilai fungsi tujuan untuk keseluruhan sampel
4	*While* (P_ps – P_ps_lama) > ε
5	Tentukan jumlah sampel elit (rho*N)
6	Hitung nilai LFR pada setiap iterasi dengan persamaan (7)
7	Hitung *update cross over* parameter dan Probabilitas mutasi sesuai persamaan (8)-(10)
8	Penyeleksian induk dengan metode *roulette wheel*
9	*Cross over*
10	Mutasi
11	Ulangi langkah ke-3
12	***End of while***

Berikut adalah formula yang digunakan algoritma CEGA:

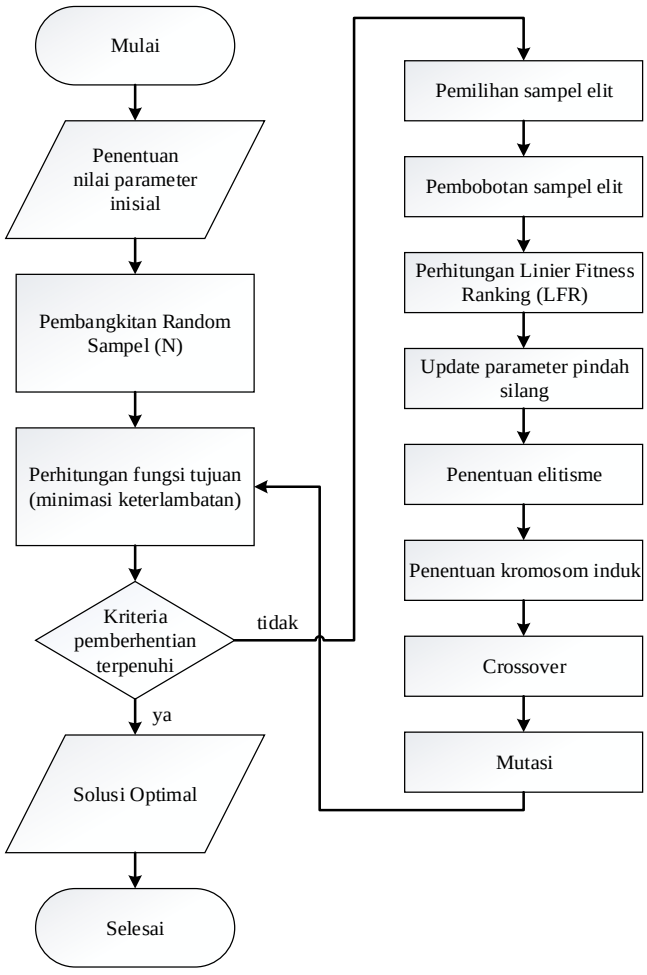
$$LFR(I(N\text{-}i+1)) = F_{max(TT)}\text{-}(F_{max(TT)}\text{-}F_{min(TT)})*((i\text{-}1)/(N\text{-}1)) \quad (9)$$

$$P_{ps(i)}=(1-\alpha)*u+(P_{ps(i+1)}*\alpha) \quad (10)$$

$$u = \frac{\overline{Z}_{e\,(TT)}}{2 * Z_{best\,(TT)}} \quad (11)$$

$$P_m = \frac{P\_ps}{2} \quad (12)$$

Gambar 2 menunjukkan *flowchart* CEGA yang dimodifikasi dari Widodo *et al.* [43].



Gambar 2. *Flowchart* CEGA yang dimodifikasi dari Widodo, et al. [43]

DOI: 10.25077/josi.v18.n2.p133-141.2019

HASIL DAN PEMBAHASAN

Dalam melakukan pengujian eksperimen CEGA, alat bantu hitung yang digunakan adalah Matlab R14 pada komputer windows 8.1 dengan *processor* AMD x64 RAM 4GB. Dalam perhitungan ini digunakan data skala kecil, sedang dan besar. Hal ini bertujuan agar dapat mengetahui efektifitas dari algoritma. Penelitian ini membandingkan algoritma CEGA dengan CE dan GA. Hasil perhitungan total *tardiness* dengan menggunakan alat bantu software Matlab R14 dapat dilihat pada Tabel 1 dan 2, sedangkan untuk waktu kompilasinya dapat dilihat pada Tabel 3 dan 4.

Algoritma CEGA rata-rata memberikan hasil perhitungan penjadwalan yang lebih baik. Pada penelitian terdahulu, algoritma CEGA juga memberikan hasil yang lebih optimal [37,39,40,45]. Ini membuktikan penelitian yang dilakukan oleh Rahmawati and Santosa [42] serta Widodo et al. [43]. Mereka menyatakan kualitas solusi yang dihasilkan CEGA menghasilkan kinerja yang lebih baik. Untuk waktu komputasi dipengaruhi oleh jumlah iterasinya pada setiap proses. Semakin banyak iterasi maka, semakin lama waktu komputasinya akan tetapi akan memberikan ruang solusi yang lebih besar. Algoritma CE dan GA rata-rata memiliki waktu komputasi yang lebih cepat dari pada algoritma CEGA. Untuk algoritma GA cenderung memberikan hasil total *tardiness* yang lebih tinggi dari pada algoritma CEGA (tabel 6). Untuk algoritma CE terkadang memberikan hasil yang sama jika dibandingkan dengan algoritma CEGA. Untuk hasil perbandingannya dapat dilihat pada Gambar 3. Tabel 5 menunjukkan hasil perhitungan rata-rata algoritma.

Tabel 1. Perbandingan Total *Tardiness* (CA dan CE)

Kelompok Job	*Job*	Mesin	GA	CE								
				rho = 0,1			rho = 0,2			rho = 0,3		
				alpha								
				0,4	0,6	0,9	0,4	0,6	0,9	0,4	0,6	0,9
Kecil	5	5	503	333	333	346	341	333	320	333	373	338
		10	449	213	184	217	216	198	196	199	175	213
		15	524	377	319	353	319	319	377	296	424	315
	10	5	529	401	415	401	419	419	419	419	439	419
		10	2615	1764	1804	1764	1804	1815	1812	1764	1882	1858
		15	12598	10403	10067	10403	10471	10369	10510	10897	10868	9946
Sedang	20	5	9642	6227	5454	6293	5863	5806	6487	5553	6177	6021
		10	14572	12418	11433	12508	11361	11934	13727	11223	11392	12448
		15	29817	25838	26313	26448	26137	26401	26354	25402	26229	26605
	25	5	16728	13518	13916	14743	12764	13199	13503	13004	13212	13577
		10	22615	19116	18954	19152	18477	19280	19416	18862	19706	19428
		15	34182	29912	30809	31641	29630	28417	30766	28724	30660	30541
Besar	35	5	154272	109196	102548	99205	105760	105750	100463	111068	107263	101748
		10	178241	156642	163099	149504	156330	158494	147599	157503	159494	154616
		15	315623	291750	295481	279321	297216	292068	288083	289552	285405	280093
	45	5	189275	155355	163099	149504	156642	163099	163099	151277	157520	156786
		10	208152	177432	178759	181033	174387	175597	180889	175785	176682	176457
		15	534172	500982	498850	467509	497671	493017	472726	506190	490780	486450

Tabel 2. Perbandingan Total *Tardiness* (CEGA)

Kelompok Job	*Job*	Mesin	CEGA								
			rho = 0,01			rho = 0,02			rho = 0,03		
			alpha								
			0,4	0,6	0,9	0,4	0,6	0,9	0,4	0,6	0,9
Kecil	5	5	333	333	333	333	333	333	333	333	333
		10	198	199	198	175	198	198	198	198	198
		15	315	315	315	315	315	315	315	315	315
	10	5	401	409	396	396	399	396	396	401	396
		10	1764	1810	1764	1871	1764	1755	1810	1810	1764
		15	9946	10685	9946	9946	10403	9946	10868	9946	9946
Sedang	20	5	5938	5866	5362	6243	5877	5579	7214	6194	5662
		10	12678	11772	11170	11921	12119	10863	12163	12134	11529
		15	26771	26460	25417	25853	26526	25746	25852	26338	25413
	25	5	14057	13615	12433	13524	13782	12647	13791	14169	13007
		10	20507	19803	18477	20275	19669	19429	20156	19403	18912
		15	30176	31346	29416	32042	31391	28601	30867	29308	29191
Besar	35	5	101876	103113	106829	101148	99003	100600	102094	100636	104446
		10	156786	157520	159917	151277	153844	151879	149499	147599	155355
		15	280517	286766	291791	281214	286294	289508	279386	281331	287703
	45	5	154616	159494	147599	157503	156330	156642	163099	149504	156642
		10	181009	181089	179653	179935	180429	179310	180522	174387	177670
		15	467060	469115	509590	464150	477244	477668	445375	461143	470210

Tabel 3. Perbandingan Total Waktu Komputasi (CE dan GA)

Kelompok Job	*Job*	Mesin	GA	CE								
				rho = 0,1			rho = 0,2			rho = 0,3		
				alpha								
				0,4	0,6	0,9	0,4	0,6	0,9	0,4	0,6	0,9
Kecil	5	5	0,078	0,296	0,167	0,102	0,171	0,186	0,127	0,218	0,203	0,194
		10	0,098	0,156	0,184	0,088	0,199	0,198	0,109	0,155	0,178	0,078
		15	0,086	0,183	0,179	0,093	0,198	0,177	0,110	0,231	0,193	0,093
	10	5	0,1094	0,2969	0,1904	0,0625	0,8175	0,4844	0,0938	0,2656	0,1875	0,1875
		10	0,1094	0,2813	0,3125	0,1250	0,3125	0,3438	0,1250	0,4531	0,1250	0,1719
		15	0,156	0,3906	0,4375	0,1406	0,5781	0,5	0,1094	0,6094	0,2969	0,2344
Sedang	20	5	0,652	0,8906	11,094	0,25	0,5313	0,8594	0,2031	10,313	0,6094	0,7031
		10	0,2188	15,781	0,4219	0,1719	0,8750	1	0,2344	0,6406	0,75	1
		15	0,0938	10,156	0,4219	0,2031	0,7813	10,781	0,25	0,6875	0,75	0,7969
	25	5	0,2344	42,031	15,781	0,8651	25,625	36,719	0,7969	34,063	23,281	41,875
		10	0,25	53,750	29,844	0,7188	45,156	38,125	10,781	40,313	34,531	30,938
		15	0,1875	43,125	19,531	0,6719	28,438	5	13,438	29,844	28,281	22,813
Besar	35	5	0,8906	149,844	78,906	41,094	181,875	159,219	48,760	195,469	129,375	118,438
		10	0,3478	98,125	67,969	22,031	12,75	81,563	3,75	160,781	169,844	47,344
		15	11,406	173,906	141,250	60,781	114,844	103,594	89,688	138,438	128,281	112,344
	45	5	0,2588	89,798	60,781	161,771	17,031	169,844	22,031	98,125	60,781	171,125
		10	11,094	138,438	132,031	107,969	155,938	127,969	115,781	128,594	120,156	116,250
		15	2,593	82,968	58,703	18,718	131,187	78,078	37,375	146,890	127,969	563,125

Tabel 4. Perbandingan Total Waktu Komputasi (CEGA)

Kelompok Job	*Job*	Mesin	CEGA								
			rho = 0,01			rho = 0,02			rho = 0,03		
			alpha								
			0,4	0,6	0,9	0,4	0,6	0,9	0,4	0,6	0,9
Kecil	5	5	0,359	1,106	0,703	0,349	0,75	0,5	0,289	0,233	0,872
		10	0,435	0,25	0,609	0,531	0,792	0,543	0,359	0,412	0,452
		15	1,278	0,281	1,327	1,498	0,801	1,402	0,356	1,447	0,571
	10	5	0,4688	0,3125	16,719	0,2344	0,7500	16,094	0,2344	0,4375	30938
		10	0,4688	0,8438	30,156	0,5156	0,8750	29,531	0,5156	0,8125	33,438
		15	0,5469	0,8750	32,188	0,5469	0,8281	34,063	0,5313	0,8438	31,719
Sedang	20	5	0,9375	14,063	27,344	0,3750	14,375	31,250	0,5	11,719	52,188
		10	0,9375	0,6094	28,750	0,4219	14,688	27,813	0,4219	0,75	58,906
		15	0,8438	0,8438	25,625	0,5313	14,531	29,375	0,5781	0,8175	53,906
	25	5	15,469	11,875	59,375	0,8438	24,375	44,531	0,7969	12,031	86,904
		10	14,688	15,625	48,750	11,250	20,625	50,781	0,7188	1,25	88,281
		15	15,938	11,406	49,375	0,8281	15,625	4,75	0,7656	12,969	84,844
Besar	35	5	1,8906	27,188	154,855	1,75	26,563	90,938	18,438	29,219	31,750
		10	17,031	26,875	81,875	16,094	24,688	83,281	17,031	25,469	84,375
		15	34,375	49,219	171,563	32,344	51,094	173,125	35,938	49,688	108,906
	45	5	35,938	51,094	49,688	112,344	35,943	32,344	170,553	48,209	34,265
		10	34,219	47,188	151,875	32,813	5,25	150,469	36,563	47,188	155,313
		15	69,831	104,531	33	71,094	105,313	130,313	86,719	99,531	121,094

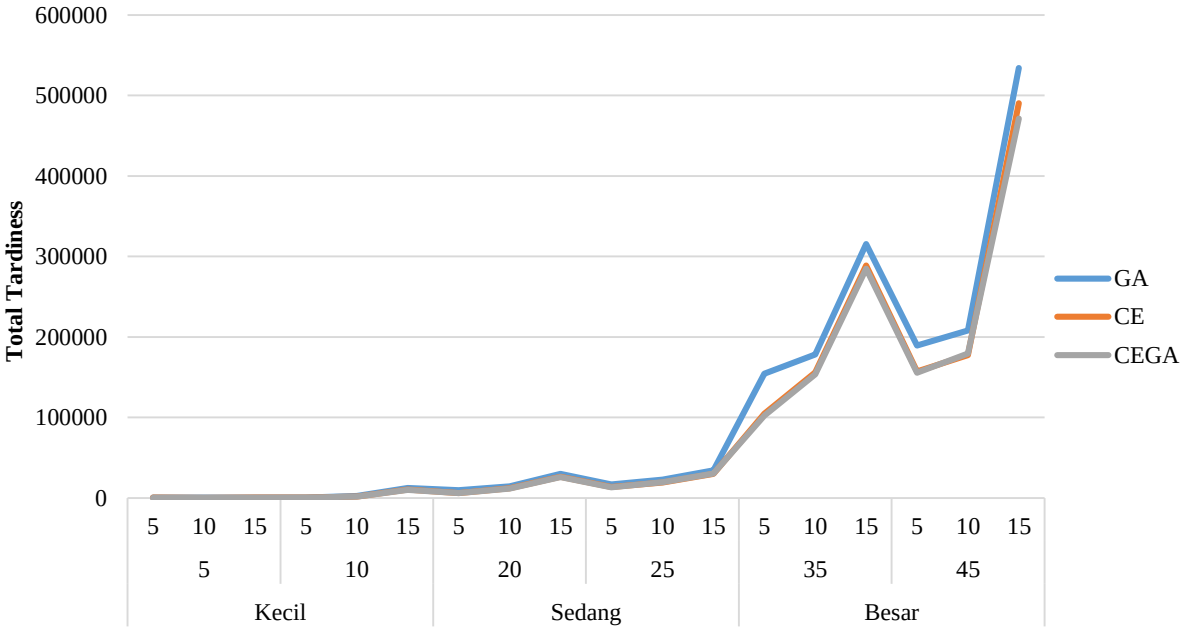
Tabel 5. Hasil Perhitungan Rata-rata Algoritma

Kelompok Job	*Job*	Mesin	GA	CE	CEGA
Kecil	5	5	503	339	333
		10	449	201	196
		15	524	344	315
	10	5	529	417	399
		10	2615	1807	1790
		15	12598	10437	10181
Sedang	20	5	9642	5987	5993
		10	14572	12049	11817
		15	29817	26192	26042
	25	5	16728	13493	13447
		10	22615	19155	19626
		15	34182	30122	30260

DOI: 10.25077/josi.v18.n2.p133-141.2019

Tabel 5. Hasil Perhitungan Rata-rata Algoritma (Lanjutan)

Kelompok Job	*Job*	Mesin	GA	CE	CEGA
Besar	35	5	154272	104778	102194
		10	178241	155920	153742
		15	315623	288774	284946
	45	5	189275	157376	155714
		10	208152	177447	179334
		15	534172	490464	471284



Gambar 3. Rekapitulasi Perhitungan

KESIMPULAN

Penelitian ini menunjukkan bahwa Algoritma CEGA efektif untuk menyelesaikan masalah minimasi *total tardiness*. Hasil percobaan menunjukan bahwa algoritma usulan memberikan performansi yang lebih baik dari algoritma CE dan GA. Kasus ini diimplementasikan pada penjadwalan *pure flow shop* dengan fungsi tujuan untuk meminimasi total keterlambatan penyelesaian *job*. Algoritma usulan menghasilkan waktu komputasi yang relatif lebih tinggi dibandingkan algoritma CE dan GA. Untuk penelitian selanjutnya disarankan untuk meningkatkan kinerja dari algoritma agar waktu komputasi yang diperlukan semakin kecil.

DAFTAR PUSTAKA

[1] D. M. Utama, "Analisa Perbandingan Penggunaan Aturan Prioritas Penjadwalan Pada Penjadwalan Non Delay N Job 5 Machine," Prosiding SENTRA (Seminar Teknologi dan Rekayasa), vol. 2, pp. 19-23, 2017.

[2] 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, pp. 79-88, 2015. https://doi.org/10.12928/si.v14i1.3706.

[3] K. R. Baker and D. Trietsch, Principles of sequencing and scheduling: John Wiley & Sons, 2013.

[4] M. L. Pinedo, Scheduling: theory, algorithms, and systems: Springer, 2016.

[5] R. Ginting, "Penjadwalan Mesin," Yogyakarta: Graha Ilmu, 2009.

[6] D. M. Utama, "Algoritma LPT-Branch and Bound Pada Penjadwalan Flexible Flowshop untuk Meminimasi Makespan," PROZIMA (Productivity, Optimization and Manufacturing System Engineering), vol. 2, pp. 20-26, 2018.

[7] D. M. Utama, T. Baroto, D. Maharani, F. R. Jannah, and R. A. Octaria, "Algoritma ant-lion optimizer untuk meminimasi emisi karbon pada penjadwalan flow shop dependent sequence set-up," 2019, vol. 9, pp. 69-78, 2019-06-28 2019.

[8] D. M. Utama, A. K. Garside, and W. Wicaksono, "Pengembangan algoritma Hybrid Flow shop Three-Stage Dengan Mempertimbangkan Waktu Setup," Jurnal Ilmiah Teknik Industri, vol. 18, pp. 72-78, 2019. https://doi.org/10.23917/jiti.v18i1.7683.

[9] D. S. Widodo, "Pengembangan Cross Entropy-Genetic Algorithm (CEGA) Pada Penjadwalan Model Flow Shop Untuk Meminimalkan Makespan," TEKNOTERAP, vol. 1, pp. 1–11, 2017.

[10] D. M. Utama, "An Effective Hybrid Sine Cosine Algorithm to Minimize Carbon Emission on Flow-shop Scheduling Sequence Dependent Setup," 2019, vol. 20, pp. 62-72, 2019-02-26 2019. https://doi.org/10.22219/JTIUMM.Vol20.No1.62-72.

[11] D. M. Utama, D. S. Widodo, W. Wicaksono, and L. R. Ardiansyah, "A New Hybrid Metaheuristics Algorithm for Minimizing Energy Consumption in the Flow Shop Scheduling Problem," International Journal of Technology, vol. 10, pp. 320-331, 2019. https://doi.org/10.14716/ijtech.v10i2.2194.
[12] A. Firmansyah, D. Utomo, and M. Irawan, "Algoritma Genetika Ddengan Modifikasi Kromosom Untuk Penyelesaian Masalah Penjadwalan Flowshop," J. Sain dan Seni, vol. 1, 2016.
[13] A. K. Garside, D. M. Utama, and M. R. Arifin, "Penjadwalan produksi flowshop menggunakan algoritma branch and bound untuk meminimasi mean tardiness," 2018, vol. 3, pp. 1-6, 2018-08-11 2018.
[14] I. Masudin, D. M. Utama, and F. Susastro, "Penjadwalan Flowshop Menggunakan Algoritma Nawaz Enscore HAM," Jurnal Ilmiah Teknik Industri, 2014.
[15] D. M. Utama, "Pengembangan Algoritma NEH Dan CDS Untuk Meminimasi Consumption Energy Pada Penjadwalan Flow Shop," 2018, p. 8, 2019-01-10 2019.
[16] 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. https://doi.org/10.12928/si.v13i2.2689.
[17] M. Firdaus, I. Masudin, and D. M. Utama, "Penjadwalan Flowshop Dengan Menggunakan Simulated Annealing," Spektrum Industri, vol. 13, 2015. https://doi.org/10.12928/si.v13i1.1836.
[18] R. Nasution, A. K. Garside, and D. M. Utama, "Penjadwalan Job Shop Dengan Pendekatan Algoritma Artificial Immune System," Jurnal Teknik Industri, vol. 18, pp. 29-42, 2017. https://doi.org/10.22219/JTIUMM.Vol18.No1.29-42.
[19] 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. https://doi.org/10.1016/S0925-5273(97)00124-2.
[20] G. C. Onwubolu and M. Mutingi, "Genetic algorithm for minimizing tardiness in flow-shop scheduling," Production planning & control, vol. 10, pp. 462-471, 1999. https://doi.org/10.1080/095372899232993.
[21] L. Min and W. Cheng, "Genetic algorithms for the optimal common due date assignment and the optimal scheduling policy in parallel machine earliness/tardiness scheduling problems," Robotics and computer-integrated manufacturing, vol. 22, pp. 279-287, 2006. https://doi.org/10.1016/j.rcim.2004.12.005.
[22] A. Hamidinia, S. Khakabimamaghani, M. M. Mazdeh, and M. Jafari, "A genetic algorithm for minimizing total tardiness/earliness of weighted jobs in a batched delivery system," Computers & Industrial Engineering, vol. 62, pp. 29-38, 2012. https://doi.org/10.1016/j.cie.2011.08.014.
[23] O. Etiler, B. Toklu, M. Atak, and J. Wilson, "A genetic algorithm for flow shop scheduling problems," Journal of the Operational Research Society, vol. 55, pp. 830-835, 2004. https://doi.org/10.1057/palgrave.jors.2601766.
[24] R. Le Riche and R. T. Haftka, "Optimization of laminate stacking sequence for buckling load maximization by genetic algorithm," AIAA journal, vol. 31, pp. 951-956, 1993. https://doi.org/10.2514/3.11710.
[25] H. Gunawan, "Aplikasi Algoritma Genetik untuk Optimasi Masalah Penjadwalan Flow-Shop," IPB (Bogor Agricultural University), 2003.
[26] N. Saputro and Y. Yento, "Pemakaian Algoritma Genetik Untuk Penjadwalan Job Shop Dinamis Non Deterministik," Jurnal Teknik Industri, vol. 6, pp. 61-70, 2005.
[27] C. Yu, Q. Semeraro, and A. Matta, "A genetic algorithm for the hybrid flow shop scheduling with unrelated machines and machine eligibility," Computers & Operations Research, vol. 100, pp. 211-229, 2018. https://doi.org/10.1016/j.cor.2018.07.025.
[28] J. F. Gonçalves, J. J. de Magalhães Mendes, and M. G. Resende, "A hybrid genetic algorithm for the job shop scheduling problem," European journal of operational research, vol. 167, pp. 77-95, 2005. https://doi.org/10.1016/j.ejor.2004.03.012.
[29] P. K. Muhuri, A. Rauniyar, and R. Nath, "On arrival scheduling of real-time precedence constrained tasks on multi-processor systems using genetic algorithm," Future Generation Computer Systems, 2018. https://doi.org/10.1016/j.future.2018.10.013.
[30] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," Annals of operations research, vol. 134, pp. 19-67, 2005. https://doi.org/10.1007/s10479-005-5724-z.
[31] M. Krisnawati, "Penyelesaian Permasalahan Penjadwalan Aktivitas Proyek dengan Batasan Sumber Daya Menggunakan Metode Cross Entropy," Dinamika Rekayasa, vol. 10, pp. 1-5, 2014.
[32] M. Caserta, E. Q. Rico, and A. M. Uribe, "A cross entropy algorithm for the Knapsack problem with setups," Computers & Operations Research, vol. 35, pp. 241-252, 2008. https://doi.org/10.1016/j.cor.2006.02.028.
[33] M. Caserta and E. Q. Rico, "A cross entropy-Lagrangean hybrid algorithm for the multi-item capacitated lot-sizing problem with setup times," Computers & Operations Research, vol. 36, pp. 530-548, 2009. https://doi.org/10.1016/j.cor.2007.10.014.
[34] J. L. Buliali, D. Herumurti, and G. Wiriapradja, "Penjadwalan Matakuliah dengan Menggunakan Algoritma Genetika dan Metode Constraint Satisfaction," JUTI: Jurnal Ilmiah Teknologi Informasi, vol. 7, pp. 29-38, 2008. https://doi.org/10.12962/j24068535.v7i1.a59.
[35] H. Bashori, "Upaya Meminimasi Makespan dengan Penerapan Algoritma Cross Entropy pada Penjadwalan Flow Shop," Widya Teknika, vol. 23, 2015.
[36] H. Bashori, P. Pratikto, and S. Sugiono, "Penjadwalan Flow Shop Dengan Penerapan Cross Entropy-Genetic Algorithm (Cega) Untuk Meminimasi Makespan," Journal of Engineering and Management in Industrial System, vol. 3, 2015. https://doi.org/10.21776/ub.jemis.2015.003.01.6.
[37] M. Budiman, "Pendekatan Cross Entropy-Genetic Algorithm Untuk Permasalahan Penjadwalan Job Shop Tanpa Waktu Tunggu Pada Banyak Mesin," Surabaya: Institut Teknologi Sepuluh Nopember, 2010.
[38] B. Santosa, M. A. Budiman, and S. E. Wiratno, "A Cross Entropy-Genetic Algorithm for m-Machines No-Wait Job-ShopScheduling Problem," Journal of Intelligent Learning Systems and Applications, vol. 3, p. 171, 2011. https://doi.org/10.4236/jilsa.2011.33018.
[39] L. Nurkhalida and B. Santosa, "Pendekatan Cross Entropy-Genetic Algorithm Pada Permasalahan Multi Objective Job Shop Scheduling," ed: UPT. Perpustakaan Institut Teknologi Sepuluh Nopember Surabaya, 2012.
[40] M. Hanka and B. Santosa, "Pengembangan Algoritma Hybrid Cross Entropy-Genetic Algorithm Pada Permasalahan Multiobjective Job Shop Scheduling Untuk Minimasi Makespan Dan Mean Flow Time," Tugas Akhir: Institut Teknologi Sepuluh Nopember, 2013.
[41] I. G. A. Widyadana and A. Pamungkas, "Perbandingan Kinerja Algoritma Genetika Dan Simulated Annealing Untuk Masalah Multiple Objective Pada Penjadwalan Flowshop," Jurnal Teknik Industri, vol. 4, pp. 26-35, 2004.
[42] N. Rahmawati and B. Santosa, "Penerapan Algoritma Hybrid Cross Entropy-Genetic Algorithm Dalam Penyelesaian Resource-Constrained Project Scheduling Problem," Prosiding SENIATI, vol. 3, pp. 37-1-5, 2017.
[43] D. S. Widodo, P. B. Santoso, and E. Siswanto, "Pendekatan Algoritma Cross Entropy-Genetic Algorithm Untuk

 DOI: 10.25077/josi.v18.n2.p133-141.2019

Menurunkan Makespan Pada Penjadwalan Flow Shop," Journal of Engineering and Management in Industrial System, vol. .https://doi.org/10.21776/ub.jemis.2014.002.01.6

[44] S. S. Rao, T.-S. Pan, and V. B. Venkayya, "Optimal placement of actuators in actively controlled structures using genetic algorithms," AIAA journal, vol. 29, pp. 942-943, 1991. https://doi.org/10.2514/3.10683.

[45] P. D. Puspitasari and B. Santosa, "Penjadwalan Truk pada Sistem Cross Docking dengan Penyimpanan Sementara dengan Algoritma Hybrid Cross Entropy-Genetic Algorithm," Surabaya: Teknik Industri, Institut Teknologi Sepuluh Nopember, 2011.

NOMENKLATUR

LFR	=	*Linear Fitness Ranking*
I	=	Indeks *job* disetiap iterasi
N	=	Jumlah sampel acak *random*
$F_{max(TT)}$	=	1/Z(1), diambil dari nilai TT minimum
$F_{min(TT)}$	=	1/Z(N), diambil dari nilai TT maksimum
P_ps	=	Parameter pindah silang
ε	=	Koefisien penghalusan
u	=	*Update cross over* parameter
$Z_{e(TT)}$	=	Rata – rata dari nilai TT
$Z_{best(TT)}$	=	Nilai terbaik dari keseluruhan TT
Pm	=	Probabilitas mutasi
ε	=	Parameter pemberhentian
t	=	Waktu proses
z	=	Jumlah *job*
x	=	Jumlah mesin
d	=	*Duedate*
T_{jz}	=	*Tardiness* pada *job*
TT	=	Total *Tardiness*
$C(j_z,M_x)$	=	Waktu Penyelesaian Job z di mesi x
$t(j_z,M_x)$	=	Waktu operasi Job z di mesi x