Journal of Applied Informatics and Computing (JAIC)
Vol.4, No.1, Juli 2020, pp. 13~16
e-ISSN: 2548-6861

Penerapan Algoritma *Elephant Herding Optimization* (EHO) pada Masalah *Hybrid Flowshop Scheduling* (HFS)

Ahmad Kamsyakawuni [1]*, Khurnia Palupi [2]*, Agustina Pradjaningsih [3]*
*Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Jember
kamsyakawuni.fmipa@unej.ac.id [1], khurniapalupi@gmail.com [2], agustina.fmipa@unej.ac.id [3]

Article Info

Article history:

Received 2020-01-04
Revised 2020-02-03
Accepted 2020-02-04

Keyword:

Elephant Herding Optimization (EHO), *Hybrid Flowshop Scheduling* (HFS), *Makespan.*

ABSTRACT

The industry is the driving force for the economy in Indonesia. One of the problems faced by industrial companies in the production process is determining the production schedule. The production schedule that is not according to the specified target can cause losses to the company. Scheduling is the allocation of resources to carry out a set of work at a specified time. The problem solved in this article is hybrid flowshop scheduling (HFS); companies in bread making will be using it. A solution to solve the HFS problem using elephant herding optimization (EHO) algorithm. For the company to complete the production process by minimizing makespan, effective scheduling is needed, taking into account the number of parallel machines. This article experimented with nine (9) jobs and seven (7) stages of the production process. The results of this article makespan 11.270 seconds using the MATLAB software. EHO algorithm program on HFS problem can minimize time for 36 minutes 39 seconds.

I. PENDAHULUAN

Industri menjadi salah satu motor penggerak perekonomian di Indonesia [1]. Salah satu masalah yang dihadapi oleh perusahaan industri dalam proses produksi adalah menentukan jadwal produksi. Jadwal produksi yang tidak sesuai dengan target yang ditentukan dapat menyebabkan kerugian pada perusahaan. Penjadwalan dalam produksi penting dilakukan agar memaksimalkan produksi dan menghindarkan perusahaan dari proses produksi yang tidak sesuai target. Penjadwalan adalah kegiatan pengalokasian sumber daya atau mesin yang ada untuk menjalankan sekumpulan tugas dalam jangka waktu tertentu. Dalam proses produksi terdapat tiga elemen penjadwalan yaitu *job*, operasi dan mesin [2].

Hybrid flowshop scheduling (HFS) merupakan penjadwalan proses produksi paralel yang memiliki mesin identik dengan urutan proses produksi searah [3]. HFS terdiri atas *n job* ($J\{1, 2, ..., n\}$) diproses pada *k-stage.* Terdapat m_i mesin yang identic pada *stage i*. *Job j* harus diproses secara serentak pada $size_{ij}$ mesin paralel yang identik pada *stage* i selama t_{ij} satuan waktu. Tujuannya adalah mencari urutan penjadwalan yang meminimumkan *makespan* ($C_{\max}$), dimana t_{ij} adalah waktu proses *job* j pada *stage* i dan $size_{ij}$ banyaknya mesin yang dibutuhkan untuk memproses *job j* pada *stage i* [4].

Agar perusahaan dapat menyelesaikan proses produksi sesuai target dengan meminimumkan *makespan*, diperlukan solusi dengan penjadwalan yang efektif. Solusi permasalahan HFS dapat dicari dengan menggunakan metode metaheuristik. Metode metaheuristik bersifat general, tidak bergantung pada jenis permasalahan sehingga dapat diterapkan berbagai macam permasalahan [5]. Metode metaheuristik dapat memberikan kemungkinan solusi yang mendekati solusi optimal [6]. Salah satu algoritma yang dapat menyelesaikan masalah optimasi adalah algoritma *elephant herding optimization* (EHO) yang terinspirasi dari perilaku menggiring kelompok gajah. Algoritma EHO dibandingkan dengan algoritma lain, yaitu *biogeography-based optimization* (BBO), *differential evolution* (DE) dan *genetic algorithm* (GA) pada masalah optimasi fungsi. Berdasarkan hasil perbandingan solusi yang didapat dengan algoritma EHO mendekati optimal dibandingkan ketiga algoritma ini [7].

Pratiwi (2019) telah menyelesaikan HFS pada perusahaan *spring bed* dengan algoritma *migrating birds optimization* (MBO). Hasil penelitian didapat algoritma MBO efektif dalam menyelesaiakan HFS karena *makespan* yang dihasilkan lebih optimal dari *makespan* perusahaan [8].

Regita (2019) telah berhasil meneliti masalah *multiple constraints knapsack* 0-1 yang diselesaikan dengan menggunakan algoritma EHO. Hasil penelitian didapat algortima EHO efektif mendapatkan keuntungan lebih optimal dari metode simpleks [9]. Berkaitan dengan hal tersebut, artikel ini menggunakan algoritma EHO pada permasalahan optimasi lain. Sehingga akan membahas lebih lanjut penerapan algoritma EHO pada masalah HFS. Artikel ini bertujuan mendapatkan solusi penjadwalan *job* dengan *makespan* minimum sekaligus pembuatan program menggunakan *software* MATLAB. Hasil *makespan* menggunakan algoritma EHO akan dibandingkan dengan *makespan* awal penjadwalan perusahaan. Manfaatnya memberikan alternatif penyelesaian penjadwalan dengan meminimasi *makespan*.

Evaluasi performasi algoritma diujikan pada *benchmark problems*. Belum adanya informasi mengenai solusi optimal pada *benchmark problems* atau pada data primer diakomodasi dengan penggunaan *lower bound* (LB), yaitu batas bawah (nilai minimum). Kriteria yang umum digunakan pada masalah HFS adalah *makespan*. Nilai *Lower Bound* dapat dihitung menggunakan persamaan (1)

$$LB = \max_{i \in M}\left\{\min_{j \in J}\{\sum_{l=1}^{i-1} t_{lj}\} + \max\left\{\left(\left\lceil \frac{1}{m_i}\sum_{j \in J} t_{ij} \cdot size_{ij}\right\rceil\right), \left(\sum_{j \in A_i} t_{ij} + \left\lceil \frac{1}{m_i}\sum_{j \in B_i} t_{ij} \cdot size_{ij}\right\rceil\right)\right\} + \min_{j \in J}\{\sum_{l=i+1}^{k} t_{lj}\}\right\} \quad (1)$$

dimana,

$$A_i = \left\{j \mid size_{ij} > \frac{m_i}{2}\right\}, \quad B_i = \left\{j \mid size_{ij} = \frac{m_i}{2}\right\}$$ [10]

Perbandingan antara *makespan* dan *lower bound* dilakukan dengan menghitung persentase deviasi atau *Percentage Deviation Algorithm* (PDA). Perhitugan persentase deviasi adalah sebagai berikut

$$\text{PDA}(l) = 100\ \% \times \frac{C_{\max}(l) - LB}{LB} \quad (2)$$

Tahap pemecahan masalah optimasi dengan menggunakan algoritma EHO sebagai berikut:

1) *Clan Updating Operator*

Posisi baru setiap gajah dalam kelompok c_i dipengaruhi oleh posisi *matriacrh* c_i. Posisi gajah *j* dalam masalah HFS sebagai urutan *job* diperbarui berdasarkan persamaan 3.

$$x_{new,c_{i,j}} = x_{c_{i,j}} + \alpha\left(x_{best,c_i} - x_{c_{i,j}}\right) r \quad (3)$$

dengan,

$x_{new,c_{i,j}}$: posisi baru untuk gajah *j* di kelompok c_i
$x_{c_{i,j}}$: posisi lama untuk gajah *j* di kelompok c_i
α : faktor skala yang menentukan *matriarch* c_i
x_{best,c_i} : gajah terbaik di kelompok c_i (*matriarch* c_i)
r : bilangan random pada interval [0,1]

Matriarch sebagai urutan *job* yang menghasilkan makespan terkecil memperbarui posisinya dengan persamaan (2.4).

$$x_{new,c_{i,j}} = \beta\, x_{center,c_i} \quad (4)$$

dengan,

β : faktor yang menentukan pengaruh x_{center,c_i}
x_{center,c_i} : pusat kelompok dihitung berdasarkan persamaan (5)

$$x_{center,c_i} = \frac{1}{n_{c_i}} \sum_{j=1}^{n_{c_i}} x_{c_{i,j}} \quad (5)$$

dengan,

x_{center,c_i} : pusat kelompok c_i
n_{c_i} : jumlah gajah dalam kelompok c_i
$x_{c_{i,j}}$: posisi gajah *j* di kelompok c_i

2) *Separating Operator*

Gajah jantan akan meinggalkan kelompoknya dan hidup sendiri ketika dewasa. Proses memisahkan diri dapat dimodelkan dalam *separating operator*. *Fitness* adalah kemampuan yang dimiliki setiap gajah. Pada permasalahan ini *makespan* digunakan sebagai tolak ukur kemampuan yang dimiliki setiap gajah. Gajah dengan *fitness* terkecil dengan kata lain posisi gajah yang menghasilkan *makespan* terbesar menerapkan *separating operator* yang ditunjukan dalam persamaan (6).

$$x_{worst,ci} = x_{min} + (x_{max} - x_{min} + 1)\, rand \quad (6)$$

dengan,

x_{max} : batas atas dari posisi individu gajah
x_{min} : batas bawah dari posisi individu gajah
$x_{worst,ci}$: individu gajah terburuk kelompok
$rand$: bilangan random dalam interval [0,1] [3]

II. METODE PENELITIAN

A. Data Penelitian

Data yang digunakan dalam artikel ini berupa data *job*, data waktu proses produksi, data mesin dan tahapan produksi pada perusahaan yang bergerak dibidang pembuatan roti. Artikel ini melakukan percobaan menggunakan 9 *job* dan 7 tahap proses produksi (*stage*). Uji parameter dengan nilai yang digunakan *nClan* dan nc_i adalah 10, 15, 20. Uji parameter dengan nilai yang digunakan α dan β adalah 0,01; 0,5; 0,9. Setiap kombinasi parameter menggunakan 10 kali percobaan dengan maksimum iterasi 500.

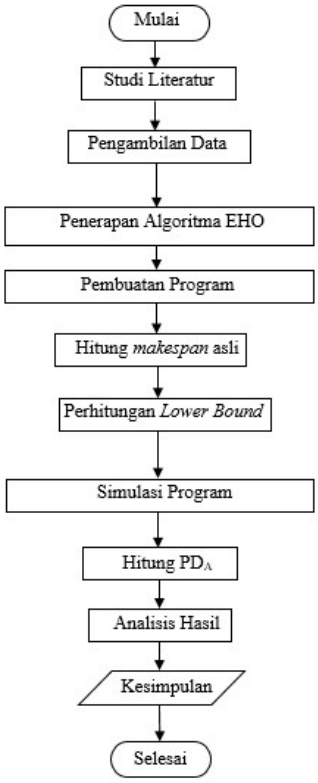
B. Skema Langkah Penelitian

Langkah-langkah yang dilakukan adalah

1) *Mengumpulkan dan mempelajari literatur yang berkaitan dengan algoritma EHO dan masalah HFS*
2) *Pengambilan Data*: Data yang digunakan berupa data *job*, data waktu proses produksi, data mesin, dan tahapan produksi. Data yang digunakan sembilan *job* tujuh *stage*.
3) *Penerapan algoritma EHO*
4) *Pengecekan kriteria pemberhentian dan memilih solusi*: Jika iterasi telah mencapai iterasi maksimal, maka proses dihentikan dan solusi terbaik didapat.

5) *Pembuatan Program*: Pembuatan program penerapan algoritma EHO pada masalah HFS dilakukan menggunakan *software* MATLAB.
6) *Simulasi Program*: Melakukan simulasi pada program yang telah dibuat.
7) *Analisis Hasil*: Hasil penjadwalan akan dibandingkan dengan *lower bound*, untuk mengetahui nilai PDA menggunakan persamaan (2). Semakin kecil nilai PDA maka semakin optimal atau lebih efektif penjadwalan pada algoritma tersebut. Hasil penjadwalan program nantinya akan dibandingkan dengan hasil *makespan* awal penjadwalan perusahaan.
8) *Kesimpulan*: Membuat kesimpulan dari hasil yang telah diperoleh.

Gambar 1 merupakan skema langkah penelitian.



Gambar 1. Skema langkah penelitian

C. Penerapan Algoritma EHO pada Masalah HFS

Langkah – langkah penerapan algoritma EHO pada masalah HFS dengan data yang telah diidentifikasi adalah sebagai berikut:

1) *Inisialisasi populasi awal*: Inisialisasi populasi awal dilakukan dengan cara membangkitkan secara acak (*X*) pada selang [-4, 4], tahap selanjutnya (*X*) diurutkan dari nilai terkecil didapat (*Y*) sebagai posisi awal berupa urutan *job* dan menghitung total waktu produksi (*makespan*) masing-masing *job*.
2) *Menyimpan Matriarch*: Menentukan *matriarch* yaitu solusi awal dengan nilai *makespan* terkecil dalam setiap kelompok.
3) *Clan Updating Operator*: Pada solusi awal akan dilakukan *clan updating operator* dengan setiap posisi (*X*) diperbarui menggunakan persamaan (3) dan posisi *matriarch* menggunakan persamaan (4). Solusi baru tersebut akan dihitung nilai *makespan*.
4) *Separating Operator*: Posisi (*X*) dengan *makespan* terbesar dalam setiap kelompok akan menerapkan *separating operator* menggunakan persamaan (6). Didapat posisi baru dan menghitung nilai *makespan*.
5) *Updating Matriach*: Jika *matriarch* baru memiliki *makespan* lebih kecil dari *matriarch* lama maka akan menggantikan *matriarch* lama. Jika tidak, maka *matriarch* lama dipertahankan.

III. HASIL DAN PEMBAHASAN

Perhitungan HFS dilakukan dengan bantuan software MATLAB. Setelah dilakukan pengujian parameter (Tabel I), dapat diketahui beberapa parameter yang perlu diperhatikan untuk mencari solusi optimal. Simulasi akhir dengan nilai parameter yang digunakan nClan = 125; nCi = 15; α = 0,9; β = 0,9 dengan menggunakan dua iterasi maksimal yaitu, 500 dan 1000. Hasil dari simulasi program dapat dilihat pada Tabel II dan Tabel III.

TABEL I
UJI PARAMETER 9 JOB DAN & STAGE ITERASI MAKSIMUM 500

No.	Nilai Parameter				Rata-rata *Makespan*	Rata-rata Iterasi	Rata-rata Waktu Komputasi
	nClan	nc_i	α	β			
1	10	10	0,5	0,5	11.350,7	145,8	147,91105
2	15	10	0,5	0,5	11.334,4	156,3	189,21
3	20	10	0,5	0,5	11.321,3	152,8	266,92539
4	20	15	0,5	0,5	11.279	163,5	335, 4813
5	20	20	0,5	0,5	11.274,7	253,9	485,70427
6	15	15	0,01	0,01	11. 417	260,5	316,67408
7	15	15	0,5	0,5	11.315	193,7	288,72076
8	15	15	0,01	0,9	11.329,7	248,1	299,3128
9	15	15	0,5	0,9	11.325,7	211	263,1874
10	15	15	0,9	0,9	11.330	165	293,6641

TABEL II
SIMULASI AKHIR ITERASI MAKSIMUM 500

No.	Makespan	Waktu komputasi	PDA	Iter	Urutan job
1	11270	2229,7492	35,1319	32	8-4-1-2-6-3-9-5-7
2	11270	2165,4557	35,1319	39	8-4-1-2-6-3-9-5-7
3	11270	2160,6316	35,1319	16	8-4-1-2-6-3-9-5-7
4	11270	2560,8959	35,1319	19	8-4-1-2-6-3-9-5-7
5.	11270	3367,9719	35,1319	3	8-4-1-2-6-3-9-5-7
6	11270	2954,2226	35,1319	28	8-4-1-2-6-3-9-5-7

7	11270	2262,8068	35,1319	23	8-4-1-2-6-3-9-5-7
8	11270	2168,0995	35,1319	37	8-4-1-2-6-3-9-5-7
9	11270	2168,5286	35,1319	7	8-4-1-2-6-3-9-5-7
10	11270	2176,3957	35,1319	335	8-4-1-2-6-3-9-5-7
Rata-rata		2421, 476	35,1319	54	

TABEL III
SIMULASI AKHIR ITERASI MAKSIMUM 1000

No.	Makespan	Waktu komputasi	PDA	Iter	Urutan job
1	11270	4594,7852	35,1319	379	8-4-1-2-6-3-9-5-7
2	11270	4593,4616	36,1319	540	8-4-1-2-6-3-9-5-7
3	11270	4559,72	35,1319	35	8-4-1-2-6-3-9-5-7
4	11270	4589,0976	35,1319	321	8-4-1-2-6-3-9-5-7
5.	11270	4398,6221	35,1319	77	8-4-1-2-6-3-9-5-7
6	11270	4603,6599	35,1319	246	8-4-1-2-6-3-9-5-7
7	11270	4583,9461	35,1319	95	8-4-1-2-6-3-9-5-7
8	11270	4495,5496	35,1319	285	8-4-1-2-6-3-9-5-7
9	11270	4723,8971	35,1319	64	8-4-1-2-6-3-9-5-7
10	11270	4625,2816	35,1319	9	8-4-1-2-6-3-9-5-7
Rata-rata		4576,802	35,1319	206	

Berdasarkan uji parameter dapat dilihat pada Tabel I, hasil yang diperoleh bahwa parameter mempengaruhi pencapaian solusi optimal. Parameter *nClan* dan nc_i semakin besar maka solusi yang dibangkitkan semakin banyak sehingga kandidat solusi baru yang muncul lebih banyak. Dengan demikian kemungkinan didapatkan solusi optimal lebih tinggi. Semakin besar nilai parameter α dan β yang diberikan tidak menjamin dapat lebih baik dalam pencarian solusi. Hal ini karena hasil perhitungan parameter α dan β menentukan urutanan *job* yang mempengaruhi *makespan*. Iterasi maksimal tidak begitu mempengaruhi hasil yang didapat jika populasi yang dibangkikan cukup besar, semakin banyak iterasi maka waktu penyelesaian program semakin lama. Hasil penelitian parameter mempengaruhi *makespan* yang dihasilkan dari urutan *job*. Jumlah populasi yang dibangkitkan sangat mempengaruhi hasil yang didapat. Setelah dilakukan uji simulasi akhir pada program hasil *makespan* terbaik dapat dilihat pada Tabel II dan Tabel III. Penjadwalan HFS dengan 9 *job* 7 *stage* menghasilkan solusi optimal yaitu 11.270 detik dengan rata-rata konvergen terbaik 54; rata-rata waktu komputasi terbaik 2421,476 detik; dan PDA sebesar 35,1319%. Waktu optimal perusahaan memproduksi roti selama 3 jam 7 menit 50 detik dengan urutan *job* 8-4-1-2-6-3-9-5-7. *Makespan* sesuai urutan awal perusahaan yaitu 13.469 detik artinya perusahaan produksi roti selama 3 jam 44 menit 29 detik dengan urutan *job* 1-2-3-4-5-6-7-8-9 dan PDA 61,4988%. Pengerjaan masalah HFS dengan menggunakan program algoritma EHO dapat meminimalisir waktu selama 36 menit 39 detik. Sehingga algortima EHO efektif menyelesaikan masalah HFS karena *makespan* yang dihasilkan program lebih optimal dari *makespan* awal perusahaan.

IV. KESIMPULAN

Berdasarkan hasil dan pembahasan dapat ditarik kesimpulan EHO efektif dalam menyelesaikan masalah HFS karena hasil yang diperoleh mendekati nilai optimal dilihat berdasarkan hasil 10 kali percobaan dengan nilai PDA 35,1319% dan menghasilkan *makespan* yaitu 11.270 detik. Parameter yaitu, kelompok gajah (*nClan*) = 125, jumlah gajah dalam kelompok (nc_i) = 15; α = 0,9; β = 0,9; dan maksimum iterasi menggunakan dua iterasi maksimal yaitu, 500 dan 1000.

DAFTAR PUSTAKA

[1] Latief, M. N. 2018. Indonesia Bangun Manufaktur sebagai Penggerak Perekonomian. Anadolu Agency. https://www.aa.com.tr/id/ekonomi/indonesia-bangun-manufaktur-sebagai-penggerak-perekonomian/1048439 [Diakses pada 2 Mei 2019].

[2] Ginting, R. 2009. *Penjadwalan Mesin.* Yogyakarta: Graha Ilmu.

[3] Engin Orhan dan Engin Batuhan. 2017. Hybrid flowshop with multiprocessor task scheduling based on earliness and tardiness penalties. *Journal of Enterprise Information Management*

[4] Ogus, C. dan M. F. Ercan. 2005. A Genetic Algorithm for Hybird Flowshop Scheduling with Multiprocessor Tasks. *Journal of Scheduling.* 8(4): 323-351.

[5] Blum, C. dan A. Roli. 2003. Metaheuristics in Combinatorial Optimization. *ACM Computing Surveys.* 35(3): 268-308.

[6] Hillier, F. S. dan G. J. Lieberman. 2010. *Introduction to Operations Research*: *Ninth edition*. Standford University.

[7] Wang, G., S. Deb, dan L. S. Coelho. 2015. Elephant herding Optimization. *Conference Paper*: 2015 3*rd International Symposium on Computational and Business Intelligence.* IEEE

[8] Pratiwi, Y. E. 2019. Penyelesaian Masalah Hybird Flowshop Scheduling (HFS) dengan Algoritma Migrating Birds Optimization (MBO). *Skripsi*. Jember: Jurusan Matematika FMIPA Universitas Jember.

[9] Regita, Y. D. 2019. Penerapan Algoritma Elephant Herding Optimization (EHO) pada Permasalahan Multiple Constraints Knapsack 0-1. *Skripsi*. Jember: Jurusan Matematika FMIPA Universitas Jember.

[10] Komaki, G. M, Teymourian Ehsan dan Kayvanfar Vafid. 2015. Minimising makespan in the two-stage assembly hybrid flowshop scheduling problem using artificial immune systems. *International Journal of Production Research*