ISSN 2338-3925

PENDEKATAN ALGORITMA *CROSS ENTROPY-GENETIC ALGORITHM* UNTUK MENURUNKAN *MAKESPAN* PADA PENJADWALAN *FLOW SHOP*

Dian Setiya Widodo[1*], Purnomo Budi Santoso[2], Eko Siswanto[3]

[1,2,3] Universitas Brawijaya, Fakultas Teknik, Malang 65145, Indonesia

Abstract *In the flow shop scheduling each job will go through every machine in the same order. The goal of this research is to complete a series of jobs in order to obtain the optimal makespan for stick skewers rounding machine. It is become necessities because the company has no planning and scheduling, while the demand exceeds the capacity Thus, optimize the scheduling is needed to be done. One way to optimize the scheduling is to minimize makespan. Based on the results of the research Cross Entropy Algorithm-Genetic Algorithm (CEGA) can provide the optimal solution. This is evident by a comparison technique that enumeration techniques obtained similar results with makespan 1982 seconds, but when compared to the company, CEGA method is better with makespan efficiency of 12.18%. The use of CEGA with the MATLAB has more advantages that simplify and speed in performing the calculations to can optimal solution.*

Keywords *Flow shop scheduling, makespan, cross entropy algorithm-genetic algorithm*

1. PENDAHULUAN

Masalah penjadwalan muncul apabila pada saat yang sama terdapat sekumpulan pekerjaan yang harus dihadapi dengan terbatasnya mesin atau fasilitas produksi yang tersedia [1]. Salah satu usaha yang dilakukan untuk tercapainya penjadwalan yang optimal adalah dengan meminimalkan total waktu penyelesaian serangkaian *job* (*makespan*). Salah satu jenis penjadwalan adalah *flow shop scheduling*. Penjadwalan *flow shop* merupakan jenis penjadwalan produksi dimana setiap *job* akan melalui setiap mesin dengan urutan yang sama. Penjadwalan produksi *flow shop* tujuan utamanya yaitu untuk menyelesaikan serangkaian pekerjaan (*job*) berdasarkan urutan proses dengan tujuan untuk mendapatkan nilai *makespan* yang optimal.

Perusahaan mesin pembulat lidi tusuk sate ini merupakan perusahaan yang berdiri sejak tahun 2008 bergerak pada bidang pembuatan mesin tusuk sate. Dimana *spare part* mesin tusuk sate di produksi oleh perusahaan sendiri, dalam proses pembuatan *spare part* roda gigi mesin tusuk sate aliran proses produksi yang dipakai yaitu model *flow shop*.

* *Corresponding author*: Dian Setiya Widodo
diansetiyawidodo@yahoo.com
Published online at http:JEMIS.ub.ac.id

Permasalahan yang dihadapi perusahaan yaitu tidak adanya perencanaan penjadwalan dan tingginya permintaan *spare part* roda gigi mesin pembulat lidi tusuk sate yang melebihi kapasitas produksi mengakibatkan perusahaan harus dapat mengoptimalkan penjadwalan *job*. Oleh karena itu untuk mengoptimalkan penjadwalan dibutuhkan metode yang efektif untuk dapat meminimasi *makespan* sehingga akan meningkatkan produksi dan kebutuhan komponen tersebut dapat terpenuhi.

Salah satu metode yang dapat dipakai untuk mengatasi permasalahan penjadwalan yaitu dengan pendekatan *metaheuristik*. *Metaheuristik* yaitu metode pendekatan yang didasarkan pada metode *heuristic* yaitu suatu metode untuk mencari solusi dengan pendekatan komputasi yang memadukan interaksi antara prosedur pencarian lokal dan strategi yang lebih tinggi untuk menciptakan proses yang mampu keluar dari titik-titik *local* optimum dan melakukan pencarian diruang solusi untuk menemukan solusi global dengan cara mencoba secara iteratif untuk memperbaiki kandidat solusi yang diinginkan [9]. Berbagai macam yang tergolong dalam metode *metaheuristik* salah satu diantaranya yaitu algoritma *cross entropy* dan *genetic algorithm*.

Metode algoritma *Cross Entropy* termasuk teknik yang cukup baru tujuannya adalah untuk

ISSN 2338-3925

menghasilkan urutan solusi yang memusat dengan cepat kearah solusi yang optimal dari hasil iterasi. Metode *Cross Entropy* Awalnya diterapkan untuk simulasi kejadian langka (*rare-event)*, lalu dikembangkan untuk beberapa kasus seperti optimasi kombinatorial, optimasi kontinyu, *machine learning*, dan beberapa kasus lain [9].

Sedangkan *Genetic Algorithm* merupakan teknik pencarian yang didasarkan pada mekanisme seleksi alam dan genetika alami. Genetik Algoritma berangkat dari himpunan solusi (populasi) yang dihasilkan secara acak. Masing-masing individu dalam populasi disebut *crommosom* yang merupakan representasi dari suatu solusi [2].

Tujuan dalam penelitian ini adalah menerapkan metode algoritma *Cross Entropy-Genetic Algorithm* (CEGA) untuk mendapatkan nilai *makespan* yang optimal. Dengan mengoptimalkan *makespan* diharapkan dapat meminimasi waktu proses sehingga akan meningkatkan *efisiensi utilitas* produksi. Penggunaan metode CEGA ini dipilih karena berdasarkan hasil penelitian terdahulu yang telah dilakukan oleh [3] dengan menggunakan metode CEGA pada kasus *job shop* yang bertujuan meminimasi *makespan* menunujukkan performansi yang lebih baik berdasarkan nilai makespan dan waktu komputasi jika dibandingkan dengan algoritma lain seperti *genetic algorithm-simulated annealing*, dan *hybrid tabu search*. Sehingga berdasarkan hasil tersebut maka pada penelitian ini akan menerapkan metode CEGA dengan mengembangkannya untuk diterapkan pada kasus *flow shop scheduling*.

Telah banyak penelitian yang dilakukan mencari penyelesaian untuk mendapatkan hasil optimal terhadap permasalahan untuk meminimasi *makespan* pada kasus penjadwalan *flow shop* seperti yang telah dilakukan oleh [4] yang menggunakan metode *genetic algorithm*, [7] memakai metode *linier programming*, [8] memakai metode *ant colony-tabu search*, dan [10] yang menggunakan metode algoritma *heuristic poor* dan *mixed integer programming*. Sedangkan penelitian yang saya lakukan ini menggunakan metode lain yang akan menggabungkan metode algoritma *cross entropy-genetic algorithm* (CEGA) pada kasus *flow shop scheduling* yang bertujuan untuk meminimasi makespan.

2. METODE PENELITIAN

Algoritma *Cross Entropy* (CE)

Pada dasarnya Cara kerja dari algoritma *Cross Entropy* dapat digambarkan sebagai berikut: strategi perolehan informasi oleh algoritma ini adalah bagaimana mengambil sampel *random* yang tepat dari ruang lingkup permasalahan dengan mendapatkan gambaran distribusi dari solusi yang bagus. Distribusi ini akan terus *di-update* berdasarkan kandidat solusi yang lebih baik. Selanjutnya sampel akan dibangun berdasarkan rata-rata distribusi yang muncul dari solusi yang bagus. Algoritma akan terus mengulang skenario yang sama hingga distribusi sampel mengarah pada area solusi optimal [6].

Genetic Algorithm

Pada umumnya cara kerja *Genetic Algorithm* dapat dijelaskan sebagai membangkitkan populasi secara *random* dimana populasi terdiri dari individu-individu yang dibangkitkan secara *random*. Selanjutnya masing-masing individu akan dievaluasi berdasarkan nilai *fitness* dari masing-masing individu acak tadi, kemudian dipilih individu dengan nilai *fitness* terbaik (semakin tinggi nilai fitness maka semakin besar peluang individu tersebut terpilih menjadi kromosom induk). Individu terpilih ini (kromosom induk) selanjutnya akan saling ditukar untuk membentuk populasi baru. Pertukaran individu terdiri dari dua jenis mekanisme, yaitu *cross over* (kawin silang) dan mutasi.

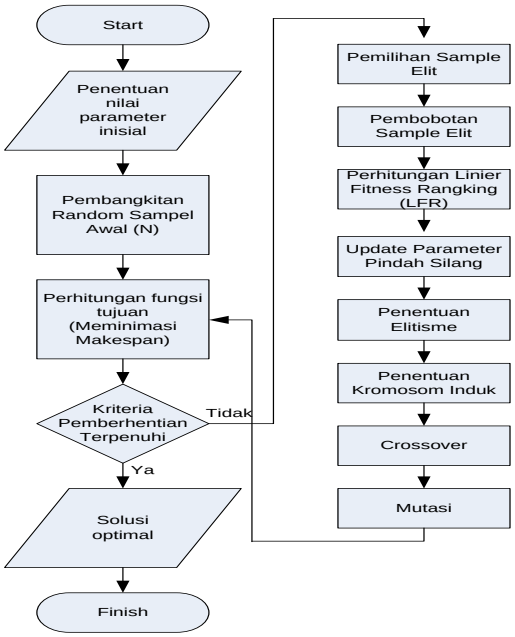
Pengembangan Algoritma *Cross Entropy-Genetic Algorithm* (CEGA)

Pengembangan algoritma *Cross Entropy* ini nantinya akan digabungkan dengan *Genetic Algorithm*. Tujuan pengembangan ini adalah memperluas pencarian solusi pada sampel elit *Cross Entropy* (CE) ketika berada di local optimal dengan menambahkan mekanisme pindah silang dan mutasi pada *Genetic Algorithm* (GA), hal tersebut berfungsi untuk menghindari kemungkinan pencarian solusi terjebak di area lokal optimal karena mekanisme GA dapat menghasilkan kromosom baru yang memiliki sifat tetap menjaga nature kromosom pada populasi awalnya.

Berikut ini tahapan pengembangan metode Algoritma *Cross Entropy-Genetic Algorithm* (CEGA) yang diusulkan di kutip dari penelitian yang pernah dilakukan oleh [6] dapat ditunjukkan pada Gambar 1.

Dalam penelitiaan ini juga akan melakukan perbandingan antara perhitungan dengan teknik enumerasi, Algoritma *CEGA*, dan metode perusahaan. Hal tersebut dilakukan unuk mengetahui *performance* dari algoritma CEGA.

ISSN 2338-3925



Gambar 1. *Flowchart* Algoritma *Cross Entropy-Genetic Algorithm (CEGA)*

3. HASIL DAN PEMBAHASAN

Perhitungan Dengan *Enumerasi*

Pada perhitungan teknik *enumerasi* ini digunakan untuk menghitung *makespan* secara manual dengan bantuan *Microsoft excel*. Informasi yang digunakan sebagai *input* yaitu waktu standart proses pengerjaan *job*. Berikut ini waktu standart pengerjaan *job* akan disajikan pada Tabel 1.

Tabel 1. Waktu Standart Proses Pengerjaan *Job* Satuan Detik

JOB	WAKTU PROSES (Detik)			
	Mesin Cutting	Mesin Las	Mesin Bubut	Mesin Hobing
1	585	913	1135	1258
2	870	0	1256	1445
3	983	0	1184	1009
4	1392	1149	1634	2569

Sumber: Data Perusahaan

Keterangan:
*Job*1= Roda gigi ukuran Ø 65 mm
Job 2= Roda gigi ukuran Ø 80 mm gigi tajam
*Job*3= Roda gigi ukuran Ø 80 mm gigi tumpul
*Job*4= Roda gigi ukuran Ø 300 mm

Penyelesian yang akan dilakukan dengan teknik *enumerasi* ini untuk mendapatkan urutan pengerjaan *job* yang optimal sehingga akan didapatkan nilai fungsi tujuan (nilai *makespan*) yang minimum. Adapun urutan proses penyelesaian permasalahan dengan teknik enumerasi ini adalah sebagai berikut:

a) Tahap 1: Pembangkitan secara acak urutan *job*
Pada tahap ini dilakukan pembangkitan secara acak semua urutan *job* yang mungkin akan terjadi.

b) Tahap 2: Perhitungan nilai *makespan*
Pada tahap ini setiap urutan jadwal (*job*) yang dibangkitkan pada tahap 1 akan dilakukan perhitungan *makespan*.

c) Tahap 3: Pemilihan solusi terbaik
Pada tahap ini semua urutan penjadwalan *job* yang telah dihitung nilai *makespan* pada tahap sebelumnya kemudian akan dipilih solusi terbaik yaitu beradasarkan nilai *makespan* terkecil.

Dari data waktu standart pengerjaan *job* kemudian dilakukan perhitungan dengan *enumerasi*. Pada penelitian ini dimana terdapat 4 *job* yang akan di jadwalkan maka total kemungkinan penjadwalan urutan *job* yaitu 4!=2x3x4=24 total kemungkinan urutan *job*. Sehingga hasil perhitungan *makespan* akan ditunjukkan pada Tabel 2.

Tabel 2. Solusi *Makespan* Masing-masing Urutan *Job*

No	Urutan Job	Makespan (Detik)
1	1 2 3 4	9276
2	2 1 3 4	9182
3	4 2 1 3	10456
4	4 3 2 1	10456
5	3 4 1 2	10430
6	3 2 4 1	9855
7	3 2 1 4	9182
8	4 3 1 2	10456
9	4 2 3 1	10456
10	4 1 2 3	10456
11	4 1 3 2	10456
12	3 4 2 1	10430
13	3 1 2 4	9182
14	3 1 4 2	9757
15	2 3 4 1	9855
16	2 3 1 4	9182
17	2 4 3 1	9881
18	2 4 1 3	9881
19	2 1 4 3	9208
20	1 3 2 4	9276
21	1 3 4 2	9757
22	1 2 4 3	9208
23	1 4 2 3	9783
24	1 4 3 2	9783

Berdasarkan hasil perhitungan dengan teknik *enumerasi* diatas menunjukkan terdapat 4 solusi alternativ penjadwalan *job* yang memiliki nilai *makespan* minimum sebesar 9182 detik yaitu pada urutan *job* 3214, *job* 3124, *job* 2314, dan *job* 2134.

Perhitungan Dengan Algoritma *Cross Entropy-Genetic Algorithm (CEGA)*

Pada bagian ini akan diberikan penjelasan mengenai proses perhtungan algoritma CEGA dalam 1 (satu) iterasi dalam menyelesaikan contoh kasus yang sama dengan *enumerasi* yang telah dilakukan pada bagian sebelumnya. Adapun tahapan-tahapan untuk melakukan perhitungan dengan CEGA sebagai berikut:

ISSN 2338-3925

1) Tahap 1: Inisialisasi Parameter

Pada tahap inisialisasi parameter algoritma *Cross Entropy-Genetic Algorithm* yang akan dipakai pada contoh perhitungan ini adalah:

(a) Jumlah sampel yang dibangkitkan (N) = 6
(b) Parameter kejarangan (rho (ρ))= 0.01
(c) Koefisien penghalusan (alpha (α))= 0.6
(d) Parameter pindah silang (P_ps) = 1
(e) Parameter pemberhentian (beta (β))= 0.0001

2) Tahap 2: Pembangkitan Sample

Bentuk sampel yang mewakili prioritas urutan *job* dari seluruh operasi pengerjaan *job* ini dibangkitkan secara *random*. Sehingga dalam penenilitian ini akan dilakukan pembangkitan bilangan *random* yang membentuk 6 sampel sebagai berikut:

X1= 3-2-1-4 X4= 4-2-1-3
X2= 1-2-3-4 X5= 1-4-2-3
X3= 1-3-4-2 X6= 2-4-3-1

Sampel yang di ambil secara *random* berupa urutan pengerjaan *job*, dimana sampel X1 urutan pengerjaan *job* yaitu *job* 3, *job* 2, *job* 1, dan *job* 4, serta sampel X2 urutan pengerjaan *job* yaitu *job* 1, *job* 2, *job* 3, dan *job* 4. Begitu juga untuk sampel berikutnya.

3) Tahap 3: Perhitungan Fungsi Tujuan

Pada tahap perhitungan fungsi tujuan akan dihitung berdasarkan nilai *makespan*, berikut ini diberikan contoh perhitungan satu sampel dan untuk sampel lain dihitung dengan langkah yang sama. Sampel yang digunakan adalah sampel pertama (X1) = 3 – 2 – 1 - 4. Urutan pertama dari sampel 3-2-1-4 adalah *job* ke 3.

Untuk menentukan nilai waktu mulai dan waktu selesai pada proses perhitungan fungsi tujuan menggunakan syarat sebagai berikut:

a) Jika operasi tersebut tidak memiliki operasi prasyarat (*job*) dan operasi pendahulu (mesin), maka letakkan operasi dengan waktu mulai = 0.
b) Jika tidak terdapat operasi prasyarat (*job*) namun terdapat operasi pendahulu (mesin), maka waktu mulai operasi = waktu selesai operasi pendahulu.
c) Jika tidak terdapat operasi pendahulu (mesin) namun terdapat operasi prasyarat (*job*), maka waktu mulai operasi = waktu selesai operasi prasyarat (*job*).
d) Jika terdapat operasi prasyarat (*job*) dan operasi pendahulu (mesin), maka waktu mulai operasi = waktu selesai terlama diantara operasi prasyarat (*job*) dan operasi pendahulu.

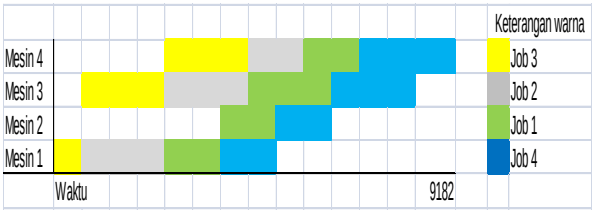
Dimana pada operasi prasyarat memiliki hubungan dalam satu *job*, sedangkan operasi pendahulu memiliki hubungan dalam operasi mesin. Berikut ini hasil perhitungan *makespan* pada penjadwalan *job* untuk sampel X1 yaitu 3 – 2 – 1 – 4, didapatkan total *makespan* sebesar 9182 detik.Untuk hasil perhitungannya dapat dilihat pada Table 3.

Tabel 3. Nilai Fungsi Tujuan Pada *Job* Sampel X1.

URUTAN JOB	WAKTU PROSES (Detik)				MULAI (Detik)				SELESAI (Detik)			
	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
3	983	0	1184	1009	0	983	983	2167	983	983	2167	3176
2	870	0	1256	1445	983	1853	2167	3423	1853	1853	3423	4868
1	585	913	1135	1258	1853	2438	3423	4868	2438	3351	4558	6126
4	1392	1149	1634	2569	2438	3830	4979	6613	3830	4979	6613	9182
MAKESPAN (Detik)									9182			

Karena rumus menghitung Z (*Makespan*) adalah $Z = \max_{1 \le j \le n} (F_j)$ ……(pers. 1)

maka nilai maksimum dari *flow time* (F_j) yaitu 9182 detik. Sehingga nilai *makespan* (Z) sebesar 9182 detik. Maka pada Gambar 2 akan ditunjukkan model gambar *gantt chart* pada sampel X1(3-2-1-4).



Gambar 2. Gambar *Gantt Chart* Sampel X1.

Untuk hasil perhitungan *makespan* pada ke enam sampel dapat dilihat pada Tabel 4.

Tabel 4. Rekapitulasi Perhitungan *Makespan* Untuk Semua Sampel

No	Sampel	Urutan Job	Makespan
1	X1	3 - 2 - 1 - 4	9182
2	X2	1 - 2 - 3 - 4	9276
3	X3	1 - 3 - 4 -2	9757
4	X4	4 - 2 - 1 - 3	10456
5	X5	1 - 4 - 2 -3	9783
6	X6	2 - 4 - 3 - 1	9881

4) Tahap 4: Penentuan Sampel Elit

Untuk penentuan sampel elit maka nilai *makespan* dari semua sampel kemudian di urutkan dari yang terkecil hingga terbesar. Pada tahap inisialisasi sebelumnya telah dipilih nilai *rho* sebesar 0.01, maka rumus jumlah sampel elit adalah *N x rho* = 6 x 0.01 = 0.06 ≈ 1 [6]. Sehingga dari Tabel 5 akan diambil satu sampel teratas yang memiliki makespan terkecil sebagai sampel elit

Yaitu terdapat pada sampel ke-1 (X1) dengan urutan prioritas *job* yaitu 3-2-1-4.

Tabel 5. Rekapitulasi Pengurutan *Makespan*

No	Sampel	Urutan Job	Makespan
X1	X1	3 - 2 - 1 - 4	9182
X2	X2	1 - 2 - 3 - 4	9276
X3	X3	1 - 3 - 4 -2	9757
X4	X5	1 - 4 - 2 -3	9783
X5	X6	2 - 4 - 3 - 1	9881
X6	X4	4 - 2 - 1 - 3	10456

5) Tahap 5: Pembobotan Sampel Elit

Nilai pembobotan sampel elit ini diperoleh dari evaluasi terhadap nilai terbaik pada iterasi sebelumnya [6]. Karena dalam perhitunngan penentuan sampel elit hanya terdapat 1 sampel elit, maka bobot sampel elit tersebut = 1.

6) Tahap 6: Perhitungan Linier Fitness Rangking (LFR)

Tahap perhitungan nilai LFR ini digunakan untuk pemilihan induk pada proses pindah silang. LFR diperoleh melalui rumus [6]:

LFR(I(N-i+1)) = Fmax-(Fmax-Fmin)*((i-1)/(N-1)) .....(pers. 2)

dengan Fmax = 1/Z(1) = 0,00010891 dan Fmin = 1/Z(N) = 1/Z(6) = 0,00009564, dari hasil *makespan* yang sudah di urutkan maka LFR untuk ke enam sampel adalah:

X1 = X1 → LFR = 0,00010891-((0,00010891-0,00009564)*((1-1)/(6-1))) = 0.0001089087

X2 = X2 → LFR = 0,00010891-((0,00010891-0,00009564)*((2-1)/(6-1))) = 0.0001066971

X3 = X3 → LFR = 0,00010891-((0,00010891-0,00009564)*((3-1)/(6-1))) = 0.0001044854

X4 = X5 → LFR = 0,00010891-((0,00010891-0,00009564)*((4-1)/(6-1)))= 0.0001022738

X5 = X6 → LFR = 0,00010891-((0,00010891-0,00009564)*((5-1)/(6-1))) = 0.0001000622

X6 = X4 → LFR = 0,00010891-((0,00010891-0,00009564)*((6-1)/(6-1))) = 0.0000978505

7) Tahap 7: Update Parameter Pindah Silang

Update parameter diperlukan untuk mendapatkan nilai parameter yang *update* untuk evaluasi kriteria pemberhentian. Pada setiap iterasi akan dilakukan *update* parameter pindah silang. Semakin besar nilai parameter pindah silang maka semakin banyak jumlah sampel yang akan mengalami pindah silang. Maka rumus *update* parameter pindah silang adalah [6]:

$P_{ps(i)}=(1-\alpha)*u+(P_{ps(i+1)}*\alpha)$ …… (pers. 3)

dengan*u* adalah

$$u = \frac{\overline{Z_e}}{2 * Z_{best}} \quad \text{..... (pers. 4)}$$

Dimana Z_e = objektif pada sampel elite dan Z_{best} = objektif terbaik pada tiap iterasi. Maka nilai parameter pindah silang (P_ps) adalah

$$u = \frac{\overline{Z_e}}{2 * Z_{best}} = \frac{9182}{2 * 9182} = 0{,}5$$

$P_{ps(i)}$= (1-α)*u+($P_{ps(i+1)}$*α)
= (1-0,6)*0,5+(1*0,6)
= 0,80

8) Tahap 8: Elitisme

Pada tahap *elitisme* ini bertujuan untuk menyimpan sampel dengan nilai fungsi tujuan terbaik pada setiap iterasi. Sampel ini nantinya akan muncul kembali pada populasi sampel di iterasi berikutnya. Oleh karena itu, untuk menjaga agar individu bernilai *fitness* tertinggi tersebut tidak hilang selama evolusi [6]. Jumlah sampel yang di elitisme sebanyak satu sampel karena dari hasil perhitungan sampel elit didapatkan nilai 1. Pada kasus ini, maka sampel yang di *elitisme* adalah X1 =3-2-1-4.

9) Tahap 9: Pemilihan Induk Pindah Silang

Pemilihan dua buah kromosom sebagai induk yang akan dipindah silangkan dilakukan secara proporsional sesuai dengan dengan nilai *fitness*-nya. Dengan menggunakan mekanisme *roulettee wheel*, dilakukan pemilihan induk 1 dari sampel elit dan induk 2 dari sampel keseluruhan.

1) Pemilihan induk 1

Dalam pemilihan induk 1 ini akan di ambil dari nilai sampel elit yang pada perhitungan sebelumnya memiliki nilai fungsi tujuan terbaik, dari mekanisme elitisme didapatkan pada populasi sampel X1=3-2-1-4.

2) Pemilihan induk 2

Pada pemilihan induk 2 ini akan dipilih berdasarkan nilai evaluasi dari sampel keseluruhan. Pada proses ini, menggunakan nilai LFR dari sampel yang sedang dievaluasi. Apabila nilai perbandingan antara kumulatif LFR dan total LFR lebih besar dari nilai *random* yang dibangkitkan, maka sampel tersebut menjadi induk 2.

Total LFR = 0.0001089087 + 0.0001066971 + 0.0001044854 + 0.0001022738 + 0.0001000622 + 0.0000978505
= 0.0006202777

Hasil kumulatif LFR total dibandingkan nilai *random* yang dibangkitkan (0.4772) adalah:

X1=0.0001089087/0.0006202777=0.176< 0.4772

X2=(0.0001089087+0.0001066971)/ 0.0006202777
= 0.348 < 0.4772

ISSN 2338-3925

X3=(0.0001089087+0.0001066971+0.0001044854) /0.0006202777= 0.516 >0. 4772
X4=(0.0001089087+0.0001066971+0.0001044854 +0.0001022738)/0.0006202777=0.681>0.4772
X5=(0.0001089087+0.0001066971+0.0001044854 +0.0001022738+0.0001000622)/0.0006202777 = 0.842 > 0.4772
X6=(0.0001089087+0.0001066971+0.0001044854 +0.0001022738+0.0001000622+0.0000978505)/0.0006202777= 1 > 0.4772

Berdasarkan hasil perhitungan diatas, diketahui sampel X1 dijadikan induk 1 sedangkan sampel X3, X4, X5, dan X6 ditetapkan sebagai induk 2.

10) Tahap 10: *Cross Over* (Pindah Silang)

Cross over merupakan tahapan dari algoritma genetik yang menyilangkan 2 induk untuk membentuk kromosom baru untuk menghasilkan individu baru yang lebih baik. Metode yang digunakan adalah 2-*point order cross over*. Dari hasil langkah pemilihan induk pindah silang diperoleh bahwa X1 ditetapkan sebagai induk 1 dan X3, X4, X5, dan X6, sebagai induk 2.

Jika dalam pembangkitan bilangan *random* diperoleh bilangan *random* (R) lebih kecil dari parameter pindah silang (P_ps) atau R < P_ps, maka akan dilakukan pindah silang terhadap kedua kromosom induk tersebut. Namun jika pembangkitan bilangan *random* (R) lebih besar dari parameter pindah silang (P_ps) atau R > P_ps, maka tidak terjadi pindah silang terhadap kedua kromosom induk.

Pada sampel X2 dan X3 didapatkan hasil bilangan *random* yang dibangkitkan nilainya lebih kecil dari P_{ps} (0.80) sehingga dilakukan pindah silang antara induk X1 dan X3. Untuk menentukan bagian dari sampel induk yang akan ditukar, maka dilakukan mekanisme sebagai berikut: dengan membangkitkan dua bilangan *random*, diperoleh bilangan 0.0713 dan 0.3772 (Tabel 6). Nilai *random* tersebut kemudian dikonversi menjadi nilai bulat, dan digunakan sebagai pembatas bagian sampel yang akan di pindah antar induk.

r_i = ceil (*random***n*) ……(pers. 5)
r1 = ceil(0.0713*4) = 1 maka p1 = 1
r2 = ceil(0.3772*4) = 2 maka p2 = 2

Induk 1 = 3 | 2 |1 4
Induk 2 = 1 | 3 |4 2

Kemudian angka yang didalam kurung ditukar antar induk menjadi seperti berikut:

Calon anak 1 = …**3** … …
Calon anak 2 = …**2** … …

Untuk mengisi titik-titik pada calon_anak_1, maka diperiksa satu per satu operasi pada induk1. Jika terdapat operasi pada induk 1 yang belum tercantum pada calon_anak_1, maka operasi tersebut akan mengisi titik-titik. Hal ini dilakukan secara berurutan. Sehingga sampel anakan hasil pindah silang menjadi:

Anak 1 = 2-3-1-4 → menggantikan X2 lama
Anak 2 = 1-2-4-3 → menggantikan X3 lama

Kemudian ulangi dengan mengevaluasi urutan sampel ke 4, 5, dan 6. Dengan perhitungan yang sama dengan awal, diketahui induk1 = X1 dan induk2 = X4, X5 dan X6. Dengan *random* lebih kecil dari P_{ps} (0.80), maka terjadi pindah silang. Induk1 dan induk2 digunakan untuk mengganti urutan ke- 4, ke- 5 dan ke-6 untuk hasil selengkapnya dapat dilihat pada Tabel 7.

Tabel 6. Proses *Cross over*

Induk Cross Over	Urutan Job	Random	r_i = ceil (*random***n*) dan n=4	Ceil Yg dicross over	Hasil Cros over	Keterangan
1	2	3	5	6	7	8
X1 dg X3	3-2-1-4	r1 0.0713	0.2852 = 1	3 | 2 | 1-4	2-3-1-4	Sampel X2
	1-3-4-2	r2 0.3772	1.5088 = 2	1 | 3 | 4-2	1-2-4-3	Sampel X3
X1 dg X4	3-2-1-4	r1 0.2413	0.9652 = 1	3 | 2 | 1-4	3-4-1-2	Sampel X4
	1-4-2-3	r2 0.4272	1.7088 = 2	1 | 4 | 2-3	1-2-3-4	-
X1 dg X5	3-2-1-4	r1 0.4028	1.6112 = 2	3-2 | 1-4	2-4-3-1	Sampel X5
	2-4-3-1	r2 0.9616	3.8464 = 4	2-4 | 3-1	2-3-1-4	-
X1 dg X6	3-2-1-4	r1 0.7425	2.97 = 3	3-2-1 | 4	2-1-4-3	Sampel X6
	4-2-1-3	r2 0.9862	3.9448 = 4	4-2-1 | 3	2-1-3-4	-

Tabel 7. Hasil *Cross Over*

Urutan	Sampel	Random (R)	P_ps	Kesimpulan	Hasil Cros over	Keterangan
X1	X1 = 3-2-1-4	0.9169	0.80	Tidak dilakukan Cros over	3-2-1-4	Diambil dari Kromosom Induk X1
X2	X2 = 1-2-3-4	0.6928	0.80	Cros over	2-3-1-4	Hasil cross over induk X1 dan X3
X3	X3 =1-3-4-2	0.0906	0.80	Cros over	1-2-4-3	Hasil cross over induk X1 dan X3
X4	X5 = 1-4-2-3	0.8643	0.80	Cros over	3-4-1-2	Hasil cross over induk X1 dan X4
X5	X6 = 2-4-3-1	0.6952	0.80	Cros over	2-4-3-1	Hasil cross over induk X1 dan X5
X6	X4 = 4-2-1-3	0.4795	0.80	Cros over	2-1-4-3	Hasil cross over induk X1 dan X6

11) Tahap 11: Mutasi

Mutasi dimaksudkan untuk memunculkan individu baru yang berbeda dengan individu yang sudah ada. *Probabilitas* mutasi (Pm) akan menentukan kromosom mana yang akan mengalami perubahan gen, semakin besar nilai *probabilitas* mutasi maka semakin banyak kromosom dalam populasi yang akan mengalami mutasi [9]. Proses mutasi akan dipilih secara *random* dan *gen* pada site tersebut akan diubah nilainya. Angka *random* akan dibangkitkan dengan batasan 0 sampai 1. Jika angka *random* (R) tersebut lebih kecil dari parameter mutasi (Pm) maka digit gen akan diganti, dan jika angka *random* (R) tersebut lebih besar dari parameter mutasi (Pm) maka digit gen tidak akan diganti.

Nilai parameter mutasi ditentukan dengan rumus sebagai berikut [5]:

ISSN 2338-3925

$$P_m = \frac{P\_ps}{2} = \frac{0.80}{2} = 0.40 \qquad \text{.....(pers.6)}$$

Setelah didapatkan nilai parameter mutasinya sebesar 0.40, sedangkan N sebesar 6. Maka jumlah sampel kromosom yang akan mengalami mutasi adalah:

Na = Pm x N= 0.40 X 6= 2.4 ≈ 3; sehingga ada 3 sampel kromosom yang akan mengalami mutasi.

Untuk mengetahui sampel kromosom mana yang akan mengalami mutasi maka akan dibangkitkan bilangan *random*. Berikut ini hasil pembangkitan bilangan *random* pada kromosom yang akan mengalami proses mutasi ditunjukkan pada Tabel 8.

Tabel 8. Penentuan Mutasi Pada Kromosom

Sampel	Hasil Cros over	Random mutasi	Probabilitas Mutasi (Pm)	Keterangan
1	2	3	4	5
X1	X1 tetap			
X2	2 - 3 - 1 - 4	0.9741	0.40	R > Pm maka tidak dilakukan mutasi
X3	1 - 2 - 4 - 3	0.3231	0.40	R < Pm maka dilakukan mutasi
X4	3 - 4 - 1 - 2	0.3249	0.40	R < Pm maka dilakukan mutasi
X5	2 - 4 - 3 - 1	0.2366	0.40	R < Pm maka dilakukan mutasi
X6	2 - 1 - 4 - 3	0.6955	0.40	R > Pm maka tidak dilakukan mutasi

Dari Tabel 8. maka didapatkan kromosom X3, X4, dan X5 yang akan mengalami mutasi, maka tahap selanjutnya yaitu melakukan mutasi pada gen ke 3 sampel kromosom tersebut. Berikut perhitungan mutasi akan diberikan pada contoh sampel kromosom X3:

rand1= 0.0668; rand2 = 0.8591; n = 4

I,J= ceil ($n*r_{ij}$) ……..(pers. 7)

I = ceil (4*0.0668) = 0.26 ≈1

J = ceil (4*0.8591) =3.43 ≈ 4

Jika nilai:

k =1 : maka dilakukan *flip mutation* (membalik)

k =2 : maka dilakukan *swap mutation* (menukar)

k =3 : maka dilakukan *slide mutation*(menggeser)

rand K = 0.6252

k= ceil (rand K*3) …….(pers. 8)

k= ceil (0.6252*3) = 1,87 ≈ 2 berarti dilakukan *swap mutation*

X3= 1 – 2 – 4 – 3 dilakukan *swap mutation*

Untuk mengetahui proses mutasi yang terjadi pada X4 dan X5, dilakukan langkah yang sama seperti pada perhitungan X3. Sehingga didapatkan proses gen yang akan dimutasi pada sampel X4 dan X5 adalah sebagai berikut:

X4= 3 – 4 – 1 – 2 dilakukan *swap mutation*

X5= 2 – 4 – 3 – 1 dilakukan *flip mutation*

Setelah dilakukan mutasi sampel X3, X4, dan X5 berubah menjadi seperti pada Table 9.

Tabel 9. Proses Hasil Mutasi

Sampel	Hasil Cros over	Random mutasi	Probabilitas Mutasi (Pm)	Keterangan	Random (r1,r2)		Nilai I,J dengan n=4		Random K	Nilai K dengan k=3	Proses Mutasi	Hasil mutasi
1	2	3	4	5	6		7=(Kolom 6*n)		8	9=(Kolom 8*k)	10	11
X1	X1 tetap											X1= 3 - 2 - 1 - 4
X2	2 - 3 - 1 - 4	0.9741	0.40	R > Pm maka tidak dilakukan mutasi								X2= 2 - 3 - 1 - 4
X3	1 - 2 - 4 - 3	0.3231	0.40	R < Pm maka dilakukan mutasi	r1	0.0668	I	0.26 = 1	0.6252	1.87 = 2	Swap mutation (menukar)	X3= 1 – 3 – 4 – 2
					r2	0.8591	J	3.43 = 4				
X4	3 - 4 - 1 - 2	0.3249	0.40	R < Pm maka dilakukan mutasi	r1	0.0891	I	0.35 = 1	0.5336	1.60 = 2	Swap mutation (menukar)	X4= 3 – 1 – 4 – 2
					r2	0.7289	J	2.91 = 3				
X5	2 - 4 - 3 - 1	0.2366	0.40	R < Pm maka dilakukan mutasi	r1	0.2024	I	0.80 = 1	0.3229	0.96 = 1	Flip mutation (membalik)	X5= 2 – 3 – 4 – 1
					r2	0.6676	J	2.67 = 3				
X6	2 - 1 - 4 - 3	0.6955	0.40	R > Pm maka tidak dilakukan mutasi								X6= 2 - 1 - 4 - 3

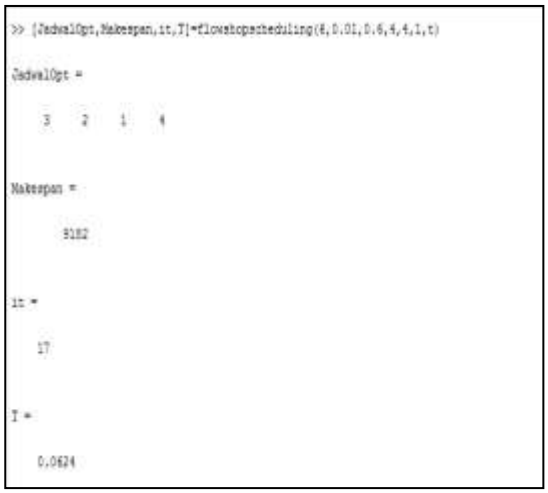
12) Tahap 12: Menghitung Nilai Fungsi Tujuan Dari Populasi Baru

Dari hasil proses mutasi akan diperoleh populasi sampel dengan anggota yang baru yang terdiri dari sampel hasil elitisme, sampel hasil pindah silang, dan sampel hasil mutasi. Berikut ini rekap hasil perhitungan nilai fungsi tujuan dari populasi baru terdapat dalam Tabel 10.

Tabel 10. Rekap Hasil Perhitungan *Makespan* Populasi Baru

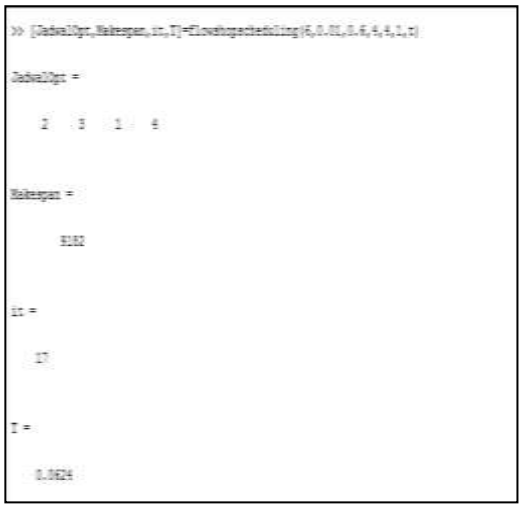
Sampel ke	Urutan Job	Makespan (Detik)
X1	3 - 2 - 1 - 4	9182
X2	2 - 3 - 1 - 4	9182
X3	1 - 3 - 4 -2	9757
X4	3 - 1 - 4 - 2	9757
X5	2 - 3 - 4 - 1	9855
X6	2 - 1 - 4 - 3	9208

Untuk menguji hasil CEGA manual maka berikut ini ditampilkan hasil *output* penjadwalan CEGA *flow shop* dengan menggunakan bantuan software.



Gambar 3. Hasil *Ouput Coding* CEGA Solusi Optimal1

 ISSN 2338-3925



Gambar 4. Hasil *Ouput Coding* CEGA Solusi Optimal 2

Metode Yang Dipakai Perusahaan

Sesuai permasalahan yang dihadapi perusahaan dimana dalam melakukan penjadwalan produksi pihak perusahaan tidak memiliki metode penjadwalan, pada kondisi sekarang ini metode yang dipakai perusahaan ini yaitu dengan cara intuisi, sehingga untuk menyelesaikan urutan *job* dalam suatu kegiatan penjadwalan produksi dapat berubah sewaktu-waktu.

Oleh karena itu untuk memilih metode penjadwalan perusahaan akan dipilih model pengerjaan *job* yang paling sering dilakukan oleh perusahaan. Dimana model pengerjaan urutan *job* yang sering dilakukan yaitu *Job* 4-2-1-3. Sehingga didapatkan hasil perhitungan total *makespan* sebesar 10456 detik. Untuk mengetahui hasil perhitungan *makespan* dapat dilihat pada Tabel 11.

Tabel 11. Total *Makespan* Metode Perusahaan

URUTAN JOB	WAKTU PROSES (Detik)				MULAI (Detik)				SELESAI (Detik)			
	M1	M2	M3	M4	M1	M2	M3	M4	M1	M2	M3	M4
4	1392	1149	1634	2569	0	1392	2541	4175	1392	2541	4175	6744
2	870	0	1256	1445	1392	2541	4175	6744	2262	2541	5431	8189
1	585	913	1135	1258	2262	2847	5431	8189	2847	3760	6566	9447
3	983	0	1184	1009	2847	3830	6566	9447	3830	3830	7750	10456
MAKESPAN (Detik)									10456			

Dari hasil perhitungan *makespan* sebelumnya telah didapatkan bahwa total *makespan* metode usulan (CEGA) sebesar 9182 detik, sedangkan metode perusahaan didapatkan total *makespan* sebesar 10456 detik. Maka besar parameter *performance efisiensi* adalah sebagai berikut:

$$Efisiensi = \frac{Z_{Makespan\text{-}Perusahaan} - Z_{Makespan\text{-}CEGA}}{Z_{Makespan\text{-}Persahaan}} x100\% \quad ..(pers.9)$$

$$Efisiensi = \frac{10456\,\text{det}\,ik - 9182\,\text{det}\,ik}{10456\,\text{det}\,ik} x100\%$$

$$Efisiensi = 12{,}18\%$$

Berdasarkan hasil perhitungan nilai *efisiensi*, didapatkan nilai sebesar 12,18% maka algoritma CEGA menunjukkan lebih *efisien* dari segi *makespan* dibanding dengan metode yang diterapkan oleh perusahaan.

4. KESIMPULAN

Berdasarkan hasil dengan melakukan perhitungan iterasi satu pada CEGA, diperoleh hasil optimal yang sama dengan hasil pada perhitungan *enumerasi*. Yaitu didapatkan nilai *makespan* sebesar 9182 detik pada urutan *job* 3-2-1-4, dan *job* 2-3-1-4. Hal tersebut menunjukkan bahwa terdapat beberapa alternativ untuk mendapatkan jadwal yang optimal.

Begitu pula dengan algoritma CEGA yang dijalankan dengan *software* Matlab. Dari hasil *running* dengan Matlab didapatkan nilai minimum *makespan* sebesar 9182 detik, dengan kondisi optimal pada iterasi ke 17, dan waktu komputasi yaitu sebesar 0,0624 detik. Sedangkan perbandingan nilai *makespan* antara metode CEGA dengan perusahaan didapatkan *efisiensi* sebesar 12,18%.

DAFTAR PUSTAKA

[1] **Baker, K**., 1974, "*Introduction to Sequencing and Scheduling", John Wiley and Sons Inc., Singapore.*

[2] **Berlianti, I., dan Arifin, M**., 2010, "Teknik-teknik Optimasi *Heuristik*", Graha Ilmu. Yogyakarta.

[3] **Budiman M.A**., 2010, "Pendekatan *Cross Entropy-Genetic Algorithm* Untuk Permasalahan Penjadwalan *Job Shop* Tanpa Waktu Tunggu Pada Banyak Mesin", Surabaya: Institut Teknologi Sepuluh Nopember.

[4] **Etiler, O., Toklu, B., Atak, M., dan Wilson, J**., 2004, "*A Genetic Algorithm For Flow Shop Scheduling Problems", Journal of the Operational Research Society*.

[5] **Hanka, M. K. R., dan Santosa, B**., 2013. "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.

ISSN 2338-3925

[6] **Nurkhalida, L., dan Santosa, B**., 2012,"Pendekatan *Cross Entropy-Genetic Algorithm* Pada Permasalahan *Multi Objective Job Shop Scheduling",*UPT. Perpustakaan Institut Teknologi Sepuluh Nopember Surabaya.

[7] **Purnama, I. L. I**., 2000, "Implementasi *Integer Linear Programming* Pada Perhitungan *Makespan* Dalam Penjadwalan Produksi", Jurnal Teknologi Industri, Vol. IV.

[8] **Sahputra, I. H., Octavia, T., dan Chandra, A. S**., 2009, "*Tabu Search* Sebagai *Local Search* Pada Algoritma *Ant Colony* Untuk Penjadwalan *Flow Shop*", Jurnal Teknologi Industri, Vol. IV.

[9] **Santosa B., dan Willy, P**., 2011, "*Metoda Metaheuristik Konsep dan Implementasi",* Guna Widya, Surabaya.

[10] **Soetanto, T. V., Palit, H. C., dan Munika, I**., 2004, "Studi Perbandingan *Performance* Algoritma *Heuristik Pour* Terhadap *Mixed Integer Programming* Dalam Menyelesaikan Penjadwalan *Flow Shop",* Jurnal Teknik Industri Vol. 6.