BAB II
LANDASAN TEORI

2.1 *Layout*

Layout adalah penyusunan dari elemen-elemen desain yang saling berhubungan kedalam sebuah bidang, sehingga membentuk susunan artistik. Tujuan utama layout adalah menapilkan elemen gambar dan teks agar menjadi komunikatif dalam sebuah cara yang dapat memudahkan pembaca menerima informasi yang disajikan (Gavin Amborse dan Paul Harris, 2005). Tata letak mesin dengan jalur aliran satu baris ada dalam konfigurasi yang berbeda, seperti garis lurus, berbentuk U, garis yang menyerupai ular dan melingkar. Meskipun sistem penanganan bahan yang sudah modern sering kali menjadi jalur aliran konfigurasi yang kompleks, jalur aliran satu baris merupakan jalur aliran yang masih populer di industri. Karena kemudahan konstruksi dan pengawasannya, *layout* mesin dengan jalur satu baris merupakan tata letak mesin yang paling umum digunakan (Ho dan Moodie,1998).

Tata letak mesin pada pabrik merupakan salah satu masalah dari studi tata letak peralatan ini. Algoritma optimal digunakan untuk memperoleh tata letak peralatan yang optimal, tetapi lebih banyak mengkonsumsi waktu. Aplikasi lainnya adalah algoritma heuristic sub-optimal yang dikategorikan oleh Kusiak dan Heragu, 1987 dalam Miao Tzu Lin, 2008 menjadi empat jenis: *construction algorithms, improvement algorithms, hybrid algorithms, and graph theoretic algorithms*. Beberapa tahun ini, adanya perkembangan perbaikan algoritma komputer, banyak peneliti yang mengusulkan bahwa meta-heuristik algoritma sama seperti *simulated annealing*, pencarian tabu dan algoritma genetika mirip dengan algoritma optimal untuk menentukan diterimanya solusi yang optimal dalam waktu yang wajar. Algoritma optimal membutuhkan banyak waktu, sedangkan algoritma *heuristic sub*-optimal membutuhkan kualitas solusi dengan mempertimbangkan keterbatasan waktu dan kemampuan algoritma.

Layout dengan jalur aliran satu baris merupakan masalah tata letak mesin yang diidentifikasi sebagai NP-masalah yang lengkap (Suresh dan Sahu, 1993 dalam Miao Tzu Lin, 2008). Solusi keakuratan perhitungan tidak dapat menyelesaikan masalah ini. Kompleksitas masalah-masalah seperti meningkat secara eksponensial dengan jumlah perangkat. Misalnya, sebuah sistem yang terdiri dari mesin N akan terdiri dari solusi ruang dengan ukuran N. Untuk

pengaturan perangkat dalam sistem, jumlah solusi yang mungkin adalah sama dengan jumlah permutasi elemen N (Ficko et al., 2004 dalam Miao Tzu Lin, 2008).

Hollier, 1963 dalam Miao Tzu Lin, 2008 memperkenalkan empat metode *flow-line analysis* (FLA) untuk masalah empat jenis mesin tunggal dimana hanya ada satu jenis mesin saja yang diperbolehkan dalam aliran garis. Bragalia, 1996 dalam Miao Tzu Lin, 2008 mengusulkan kombinasi *simulated annealing* dan genetik algoritma untuk meminimalkan total *backtracking* dalam proses pembuatan pesanan pada mesin yang digunakan dalam satu jalur. Hal ini diasumsikan bahwa lokasi mesin disesuaikan dengan fasilitas ruangan yang ada. Prosedur tata letak dua fase adalah menggabungkan analisis aliran baris dan *simulated annealing* disarankan oleh Ho dan Moodie (1998). Empat metode FLA dimodofikasi oleh Hollier deangan fase pertama (metode 1,2, 3 dan 4) dan memperkenalkan dua fase baru dalam metode FLA (metode 5 dan 6). Metode 5 adalah jalur yang cocok untuk aliran dua arah dalam satu dan metode 6 cocok untuk aliran satu arah dalam satu garis. Ponnambalam dan Ramkumar, 2001 dalam Miao Tzu Lin, 2008 mengusulkan dua metode terbaik, yaitu FLA metode 5 dan metode FLA 6 dan menggabungkannya dengan algoritma genetika untuk mengurangi biaya penanganan bahan dengan desain tata letak yang efisien. Ficko et al.,2004 dalam Miao Tzu Lin, 2008 mempresentasikan model *Flexible Manufacturing System* (FMS) ke dalam baris tunggal atau baris berganda dengan order berbasis algoritma genetika. Dalam studi tata letak mesin, pemilihan fungsi objektif telah difokuskan untuk meminimasi waktu bergerak, meminimasi total jarak bergerak, minimasi jarak bergerak antara bagian, meminimasi biaya bergerak bahan dan meminimasi waktu pengembalian (Heragu dan kusiak murni, 1988; Kumar et al, 1995; Ho dan Moodie, 1998; dan Sarker et al,.1998 dalam Miao Tzu Lin, 2008)

2.2 *Line Balancing*

Keseimbangan lintasan (*line balancing*) adalah lintasan produksi dimana material berpindah secara kontinyu dengan laju rata-rata yang sama melalui sejumlah stasiun kerja, tempat dilakukannya pekerjaan perakitan (Elsayed, 1994). Definisi klasik dari masalah lintasan keseimbangan perakitan (Hoffman, 1990) meliputi:

- set (n) tugas masing-masing sesuai dengan waktu (t_i) penyelesaiannya;
- set hubungan utama diantara masing-masing tugas, dan
- waktu siklus *C*.

Masalahnya adalah untuk menetapkan tugas yang sedemikian rupa sehingga dapat meminimalkan jumlah stasiun kerja (N) pada baris tanpa melanggar kendala yang diutamakan atau tanpa memiliki jumlah waktu tugas disetiap stasiun kerja tanpa melebihi waktu siklus. Perbedaan antara waktu siklus dan jumlah tugas disetiap satu stasiun kerja disebut sebagai "waktu kendur" atau s_j. Total waktu kendur (S) adalah jumlah dari sj s semua stasiun kerja. S juga bisa dihitung dengan mengalikan waktu siklus dengan jumlah stasiun kerja dan mengurangi total waktu tugas (T). Secara teoritis total kendur minimum (S*) adalah $N*C-T$.

Tujuan *line balancing* adalah untuk memperoleh suatu arus produksi yang lancar dalam rangka memperoleh utilisasi yang tinggi atas fasilitas, tenaga kerja dan peralatan melalui penyeimbangan waktu kerja antar *workstation*, dimana setiap elemen tugas dalam suatu kegiatan produksi dikelompokkan sedemikian rupa dalam beberapa stasiun kerja yang telah ditentukan, sehingga diperoleh keseimbangan waktu kerja yang baik (Elsayed, 1994). Permulaan munculnya persoalan *line balancing* berasal dari ketidakseimbangan lintasan produksi yang berupa adanya *work in process* pada beberapa *workstation*.

Menangani munculnya persoalan masalah *line balancing*, mengisyaratkan mendahulukan persoalan utama, waktu tugas dan waktu siklus yang ditangani. Berdasarkan ketiga aspek, ada tiga pendekatan yang berbeda untuk menangani masalah *line balancing*. Pendekatan pertama adalah untuk menemukan waktu siklus seminimum mungkin sesuai dengan hak yang lebih tinggi dan waktu tugas yang ditetapkan. Pendekatan kedua adalah untuk mencari minimal jumlah stasiun kerja untuk memberikan waktu siklus dengan dibatasi oleh hak yang lebih tinggi. Pendekatan ketiga adalah untuk memvariasikan waktu tugas dengan memegang hubungan hak yang lebih tinggi secara konstan dan waktu siklus yang tetap (Hoffman, 1990).

Jalur perakitan dikatakan sempurna/seimbang, jika total waktu kendur (yaitu jumlah dari waktu diam semua stasiun sepanjang garis) adalah nol (Baybars, 1986 dalam Miao Tzu Lin, 2008). Dalam situasi nyata, sangat sulit untuk mencapai keseimbangan yang sempurna karena laju produksi masing-masing stasiun kerja tidak sama. Waktu kendur dapat terjadi sebagai hasil perturbasi baris yang disebabkan oleh ketidakhadiran operator, kerusakan mesin dan perbaikan, variasi *material handling* dan juga berbagai kinerja operator.

2.3 Istilah-istilah Dalam *Line Balancing*

a) *Precedence diagram*

Merupakan gambaran secara grafis dari urutan kerja operasi kerja, serta ketergantungan pada operasi kerja lainnya yang tujuannya untuk memudahkan pengontrolan dan perencanaan kegiatan yang terkait di dalamnya. Adapun tanda-tanda yang dipakai sebagai berikut: (Elsayed, 1994).

1) Simbol lingkaran dengan huruf atau nomor di dalamnya untuk mempermudah identifikasi dari suatu proses operasi
2) Tanda panah menunjukkan ketergantungan dan urutan proses operasi. Dalam hal ini, operasi yang berada pada pangkal panah berarti mendahului operasi kerja yang ada pada ujung anak panah
3) Angka diatas simbol lingkaran adalah waktu standar yang diperlukan untuk menyelesaikan setiap operasi

b) *Asssamble product*

Adalah produk yang melewati urutan *workstation* dimana setiap *workstation* (WS) memberikan proses tertentu hingga selesai menjadi produk akhir pada perakitan akhir (Elsayed, 1994)..

c) *Work elemen*

Elemen operasi merupakan bagian dari seluruh proses perakitan yang dilakukan (Elsayed, 1994).

d) Waktu operasi (Ti)

Adalah waktu standar untuk menyelesaikan suatu operasi (Elsayed, 1994).

e) *Workstation* (WS)

Adalah tempat pada lini perakitan dimana proses perakitan dilakukan. Setelah menentukan interval waktu siklus, maka jumlah stasiun kerja efisien dapat ditetapkan dengan rumus berikut: (Elsayed, 1994)

$$\boldsymbol{Kmin} = \frac{\sum_{i=1}^{n} \boldsymbol{ti}}{\boldsymbol{C}} \quad \text{.....(1)}$$

Di mana:

Ti : waktu operasi/elemen (I=1,2,3,…,n)

C :waktu siklus stasiun kerja

N : jumlah elemen

K_{min} : jumlah stasiun kerja minimal

f) *Cycle time* (CT)

Merupakan waktu yang diperlukan untuk membuat satu unit produk satu stasiun. Apabila waktu produksi dan target produksi telah ditentukan, maka waktu siklus dapat diketahui dari hasil bagi waktu produksi dan target produksi. Dalam mendesain keseimbangan lintasan produksi untuk sejumlah produksi tertentu, waktu siklus harus sama atau lebih besar dari waktu operasi terbesar yang merupakan penyebab terjadinya *bottle neck* (kemacetan) dan waktu siklus juga harus sama atau lebih kecil dari jam kerja efektif per hari dibagi dari jumlah produksi per hari, yang secara matematis dinyatakan sebagi berikut: (Elsayed, 1994)

$$\boldsymbol{ti\ max} \leq \boldsymbol{CT} \leq \frac{\boldsymbol{P}}{\boldsymbol{Q}} \quad \text{.....(2)}$$

Dimana:

ti max : waktu operasi terbesar pada lintasan

CT : waktu siklus (*cycle time*)

P : jam kerja efektif per hari

Q : jumlah produksi per hari

g) *Station time* (ST)

Jumlah waktu dari elemen kerja yang dilakukan pada suatu stasiun kerja yang sama (Elsayed, 1994).

h) *Idle time* (I)

Merupakan selisih perbedaan antara *cycle time* dan *station time* atau CT dikurangi ST (Elsayed, 1994).

i) *Balance delay* (D)

Sering disebut *balancing loss*, adalah ukuran dari ketidakefisiensinan lintasan yang dihasilkan dari waktu menganggur sebenarnya yang disebabkan karena pengalokasian yang kurang sempurna di antara stasiun-stasiun kerja. *Balance delay* ini dinyatakan dalam persentase. *Balance delay* dapat dirumuskan: (Elsayed, 1994)

$$\boldsymbol{D = \frac{(n\,x\,C) - \sum_{i=1}^{n} ti}{(n\,x\,C)}\,x\,100\%} \quad \text{.....(3)}$$

Dimana:

n : jumlah stasiun kerja

C : waktu siklus terbesar dalam stasiun kerja

$\sum ti$: jumlah waktu operasi dari semua operasi

ti : waktu operasi

j) *Line efficiency* (LE)

Adalah rasio dari total waktu di stasiun kerja dibagi dengan waktu siklus dikalikan jumlah stasiun kerja yang dirumuskan sebagai berikut: (Elsayed, 1994)

$$\boldsymbol{LE = \frac{\sum_{i=1}^{K} STi}{(K)(CT)}\,x100\%} \quad \text{.....(4)}$$

Dimana:

STi : waktu stasiun dari stasiun ke-1

K : jumlah (banyaknya) stasiun kerja

CT : waktu siklus

k) *Smoothness index* (SI)

Adalah suatu indeks yang menunjukkan kelancaran relatif dari penyeimbangan lini perakitan tertentu yang dirumuskan sebagai berikut: (Elsayed, 1994)

$$\mathbf{SI} = \sqrt{\sum_{i=1}^{K} (STi\ max - STi)^2} \quad \text{.....(5)}$$

Dimana:

STi max : maksimum waktu di stasiun

STi : waktu stasiun di stasiun kerja ke-i

l) *Output production* (Q)

Adalah jumlah waktu efektif yang tersedi dalam suatu periode dibagi dengan *cycle time* yang dirumuskan sebagai berikut: (Elsayed, 1994)

$$\boldsymbol{Q = \frac{T}{CT}} \quad \text{.....(6)}$$

Dimana:

T : jam kerja efektif penyelesaiaan produk

C : waktu siklus terbesar

2.4 Metode-Metode *Line Balancing*

2.4.1 Metode *Kilbridge-Wester Heuristic*

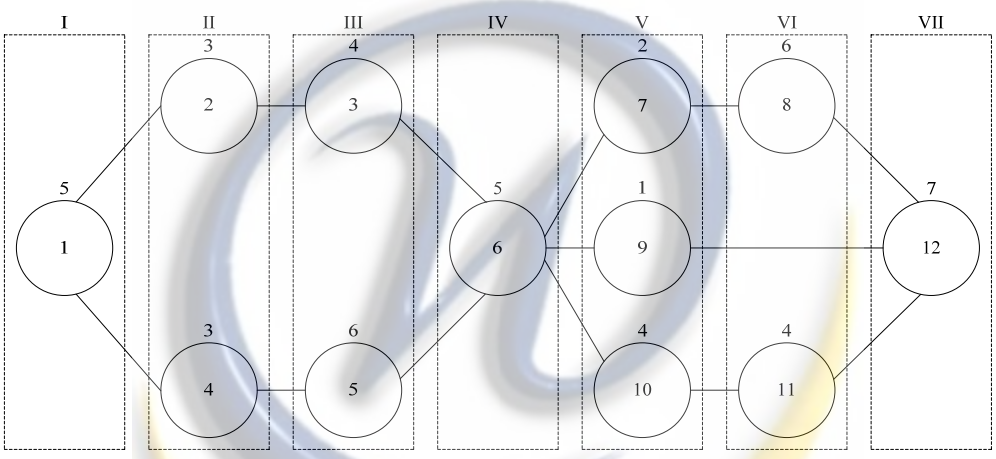
Langkah-langkah proses perhitungan menggunakan metode *Kilbridge-Wester* antara lain: (Elsayed, 1994)

1. Gambar *precedence* diagram.
2. Bagi elemen-elemen kerja dalam diagram tersebut ke dalam kolom-kolom. Kolom I adalah elemen-elemen kerja yang tidak memiliki elemen kerja pendahulu (*predecessor*). Kolom II adalah elemen-elemen kerja pendahulu di kolom I. Kolom III adalah elemen-elemen kerja dengan elemen kerja pendahulu di kolom III dan seterusnya.
3. Tentukan waktu siklus (CT) dari bilangan prima waktu total elemen kerja dan tentukan jumlah stasuin kerja.
4. Tempatkan elemen-elemen kerja ke stasiun kerja sedemikian sehingga total waktu elemen kerja tidak melebihi waktu siklus. Hapus elemen kerja yang sudah ditempatkan dair daftar elemen kerja.
5. Bila penempatan suatu elemen kerja mengakibatkan total waktu elemen kerja melebihi waktu siklus, maka elemen kerja tersebut ditempatkan di stasiun kerja berikutnya.
6. Ulangi langkah 3 dan langkah 5 sampai seluruh elemen kerja ditempatkan.

Contoh kasus pada gambar 2.1: (Elsayed, 1994).

Solusi

Elemen 1	pada kolom I
Elemen 2 dan 4	pada kolom II
Elemen 3 dan 5	pada kolom III
Elemen 6	pada kolom IV
Elemen 7, 9 dan 10	pada kolom V
Elemen 8 dan 11	pada kolom VI
Elemen 12	pada kolom VII



Gambar 2.1 *Precendence* Diagram Kasus 1

(Sumber: Elsayed, 1994)

Tabel 2.1 Waktu Proses Kasus 1

Elemen Kerja (i)	Waktu Proses (Ti)
1	5
2	3
3	4
4	3
5	6
6	5
7	2
8	6
9	1
10	4
11	4
12	7 **(*t max*)**

(Sumber: Elsayed, 1994)

Perkalian angka dasar yang mungkin untuk 50 adalah 2 x 5 x 5. Batasan waktu siklus yang diberikan adalah 7 ≤ CT ≤ 50. Semua hasil kombinasi perkalian angka dasar untuk waktu siklus akan menjadi *feasible* waktu siklus atau *infeasible* waktu siklus.

feasible waktu siklus	***infeasible*** waktu siklus
$C_1 = 50$	$C_4 = 2$
$C_2 = 5$ x5 = 25	$C_5 = 5$
$C_3 = 2$ x 5 = 10	-

Memisalkan keseimbangan baris menggunakan CT = 10. Tahap pertama memberikan penugasan elemen-elemen kerja pada *workstations* terlihat pada tabel 2.2 dan elemen kerja yang saling berhubungan terlihat pada tabel 2.3. Selanjutnya mengelompokkan elemen kerja ke dalam *worksation* (WS). Elemen 1 terpilih menjadi yang pertama karena elemen 1 merupakan elemen pendahulu. Oleh karena itu, elemen 1 dapat ditandai pada *workstation* 1. Salah satu elemen antara elemen 2 dan 4 mempunyai waktu proses yang sama yaitu 3 dapat ditugaskan pada *workstation* 1. Elemen 2 yang ditugaskan pada *workstation* 1 sehingga jumlah waktu kerja pada *workstation* 1 adalah 8 detik ≤ CT. elemen 4 tidak dapat ditugaskan pada *workstation* 1 karena bila elemen 4 ditugaskan pada *workstation* 1, waktu kerja pada *worksation* 1 melebihi CT. Maka elemen 4 ditugaskan pada *worksation* 2 yang diikuti oleh elemen-elemen selanjutnya untuk menempatkan elemen-elemen pada *workstations.* Penempatan lain elemen-elemen pada *workstations* terdapat pada tabel 2.4.

Tabel 2.2 Penugasan *Workstation*

WS	Elemen i	Ti	Waktu WS	Kumulatif Waktu WS
I	1	5	5	5
II	2	3	6	11
	4	3		
III	3	4	10	21
	5	6		
IV	6	5	5	26
V	7	2	7	33
	9	1		
	10	4		
VI	8	6	10	43
	11	4		
VII	12	7	7	50

(Sumber: Elsayed, 1994)

Line efficient (LE) $= \frac{50}{6\ x\ 10}$ x 100% = 83.3%

Smoothness index (SI)= $\sqrt{4 + 1 + 1 + 9 + 9}$ = 4.89

Tabel 2.3 Elemen Pendahulu Untuk Setiap Elemen Kerja

Elemen Kerja i	Elemen Pendahulu	Ti
1	0	5
2	1	3
3	2	4
4	1	3
5	2	6
6	5	5
7	6	2
8	7	6
9	6	1
10	6	4
11	7	4
12	11	7

(Sumber: Elsayed, 1994)

Hasil uji coba tabel 2.4, secara perlahan dapat dilakukan perpindahan elemen-elemen kerja antar *workstations* untuk menghasilkan keseimbangan terbaik antar *workstations*. Perpindahan yang dapat terjadi terlihat pada tabel 2.5. Waktu siklus yang diberikan adalah 9 (CT=9). Hasil LE dan SI untuk CT = 9 adalah

Line efficient (LE) $= \frac{50}{6 \times 9}$ x 100% = 92.6%

Smoothness index (SI)= $\sqrt{1+1+1+1}$ = 2

Tabel 2.4 Elemen-elemen Penugasan Untuk *Workstations* (CT=10)

WS	Elemen i	Ti	Waktu WS (ST)	CT - ST
I	1	5	8	1
	2	3		
II	4	3	9	1
	5	6		
III	3	4	9	1
	6	5		
IV	7	2	7	3
	9	1		
	10	4		
V	8	6	10	0
	11	4		
VI	12	7	7	3

(Sumber: Elsayed, 1994)

Tabel 2.5 Elemen-elemen Penugasan Untuk *Workstations* (CT=9)

WS	Elemen i	Ti	Waktu WS (ST)	CT - ST
I	1	5	8	1
	2	3		
II	4	3	9	0
	5	6		
III	4	4	9	0
	6	5		
IV	7	2	8	1
	8	6		
V	10	4	8	1
	11	4		
VI	9	1	8	1
	12	7		

(Sumber: Elsayed, 1994)

2.4.2 Metode *Helgeson-Birnie (Positional-Weight Technique)*

Metode ini sesuai dengan namanya yang dikemukakan oleh Helgeson dan Birnie. Langkah-langkah dalam metode ini adalah sebagai berikut: (Elsayed, 1994).

1. Buat *precedence diagram* untuk setiap proses.
2. Tentukan bobot posisi untuk masing-masing elemen kerja yang berkaitan dengan waktu operasi untuk waktu pengerjaan yang terpanjang dari mulai operasi permulaan hingga sisa operasi sesudahnya.
3. Membuat *rangking* tiap elemen pengerjaan berdasarkan bobot posisi dilangkah 2. Pengerjaan yang mempunyai bobot terbesar diletakkan pada rangking pertama.
4. Tentukan waktu siklus (CT).
5. Pilih elemen operasi dengan bobot tertingg i, alokasikan ke suatu stasiun kerja. Jika masih layak (waktu stasiun < CT), alokasikan operasi dengan bobot tertinggi berikutnya, namun lokasi ini tidak boleh membuat waktu stasiun (ST) > CT.
6. Bila alokasi suatu elemen operasi membuat waktu stasiun > CT, maka sisa waktu ini (CT – ST) dipenuhi dengan alokasi elemen operasi dengan bobot paling besar dan penambahannya tidak membuat ST < CT.
7. Jika elemen operasi yang jika dialokasikan untuk membuat ST < CT sudah tidak ada, kembali ke langkah 5.

Contoh kasus pada gambar 2.1:

Solusi

Memposisikan elemen berdasarkan hasil perhitungan bobot untuk setiap elemen yang terlihat pada tabel 2.6. contoh perhitungan bobot untuk operasi elemen 6 = {(5+2+6+7), (5+1+7), (5+4+4+7)} = 20. Urutan bobot operasi masing-masing elemen terlihat pada tabel 2.7. Dengan mengikuti langkah 4, 5 dan 6 diperoleh penugasan elemen-elemen pada *workstations* yang terlihat pada tabel 2.8.

Tabel 2.6 Urutan Bobot Operasi Elemen Kasus 1

Elemen Kerja (i)	***Positional Weight* (PW)**
1	34
2	27
3	24
4	29
5	25
6	20
7	15
8	13
9	8
10	15
11	11
12	7

(Sumber: Elsayed, 1994)

Tabel 2.7 Urutan Bobot Operasi Elemen Berdasarkan *Positional Weight*

Peringkat	**Elemen Kerja (i)**	***Positional Weight* (PW)**
1	1	34
2	4	29
3	2	27
4	5	25
5	3	24
6	6	20
7	7	15
8	10	15
9	8	13
10	11	11
11	9	8
12	12	7

(Sumber: Elsayed, 1994)

Tabel 2.8 Elemen-elemen Penugasan Untuk *Workstations* (CT=10)

WS	Elemen i	Ti	Waktu WS (ST)	CT - ST
I	1	5	8	2
	4	3		
II	2	3	9	1
	5	6		
III	3	4	9	1
	6	5		
IV	7	2	6	4
	10	4		
V	8	6	10	0
	11	4		
VI	9	1	8	2
	12	7		

(Sumber: Elsayed, 1994)

Line efficient (LE) $= \frac{50}{6\ x\ 10}$ x 100% = 83.3%

Smoothness index (SI) $= \sqrt{4 + 1 + 1 + 16 + 4}$ = 5.09

2.4.3 Metode *Moodie-Young*

Metode *Moodie-Young* memiliki dua tahap analisis. Fase (tahap) satu adalah membuat pengelompokan stasiun kerja berdasarkan matriks hubungan antar-*task*, tidak dirangking seperti metode *Helgeson-Birnie*. Fase dua, dilakukan revisi pada hasil fase satu (Elsayed, 1994).

Fase satu: Elemen pengerjaan ditempatkan pada stasiun kerja yang berurutan dalam lini perakitan dengan menggunakan aturan *largest-candidate*. Aturan *largest-candidate* terdiri atas penempatan elemen-elemen yang ada untuk tujuan penurunan waktu. Dari sini, bila dua elemen pengerjaan cukup untuk ditempatkan di stasiun, salah satu yang mempunyai waktu yang lebih besar ditempatkan pertama. Setelah masing-masing elemen ditempatkan, ketersediaan elemen dipertimbangkan untuk tujuan pengurangan nilai waktu untuk penugasan selanjutnya. Sebagai pemisalan, matriks P menunjukkan pengerjaan pendahulu masing-masing elemen dan matriks F pengerjaan pengikut untuk tiap elemen untuk tiap prosedur penugasan.

Fase dua: Pada fase dua ini mencoba untuk mendistribusikan waktu nganggur (*idle*) secara merata (sama) untuk tiap-tiap stasiun melalui mekanisme jual dan transfer elemen antar stasiun. Langkah-langkah pada fase dua ini adalah sebagai berikut: (Elsayed, 1994).

1. Menentukan dua elemen terpendek dan terpanjang dari waktu stasiun dari penyeimbangan fase satu.
2. Tentungan setengah dari perbedaan kedua nilai tujuan (GOAL).
3. GOAL = ($ST_{max} - ST_{min}$) / 2.
4. Menentukan elemen tunggal dalam ST_{max} yang lebih kecil dari kedua nilai GOAL dan yang tidak melampaui elemen pengerjaan terdahulu.
5. Menentukan semua penukaran yang mungkin dari ST_{max} dengan elemen tunggal dari ST_{min} yang mereduksi ST_{max} dan mendapatkan ST_{min} akan lebih kecil dari 2 x GOAL.
6. Lakukan penukaran yang ditunjukkan oleh kandidat dengan perbedaan mutlak terkecil antara kandidat tersebut dengan GOAL.
7. Bila tidak ada penukaran atau transfer yang dimungkinkan antara stasiun terbesar dan terkecil, mengusahakan penukaran antara *rank* pada pengerjaan berikut: N (stasiun ranking ke N memiliki jumlah waktu *idle* terbesar), N-1, N-2, N-3, …, 3, 2, 1.
8. Bila penukaran masih tidak mungkin, lakukan pembatasan dengan nilai GOAL dan ulangi langkah satu hingga enam.

Contoh kasus pada gambar 2.1:

Solusi

Pertama-tama membuat struktur matriks P dan F yang terlihat pada tabel 2.9. kolom 1 pada matriks P adalah daftar semua nomor elemen. Kolom 2, 3 dan 4 berisi matrik yang mendahului elemen kerja yang ada pada kolom 1. Kolom 2,3 dan 4 pada matrik F berisi elemen yang mengikuti elemen pendahulu pada colum 1. Apabila tidak ada elemen yang mendahului atau tidak ada elemen yang mengikuti, maka kolom elemen kerja pada matrik F atau matrik P diberi angka nol.

Tabel 2.9 Matrik P dan F Kasus 1

Elemen	Matriks P			Elemen	Matriks F		
1	0	0	0	1	2	0	0
2	1	0	0	2	3	0	0
3	2	0	0	3	6	0	0
4	1	0	0	4	5	0	0
5	4	0	0	5	6	0	0
6	3	5	0	6	7	9	10
7	6	0	0	7	8	0	0
8	7	0	0	8	12	0	0
9	6	0	0	9	12	0	0
10	6	0	0	10	11	0	0
11	10	0	0	11	12	0	0
12	8	9	11	12	0	0	0

(Sumber: Elsayed, 1994)

Pemecahan masalah pada tahap pertama adalah memilih elemen pada matrik P yang mempunyai nilai matriks nol semua karena elemen yang dimaksud tidak mempunyai elemen pendahulu. CT=10 dan penugasan elemen-elemen ke *workstations* terlihat pada tabel 2.10.

Line efficient (LE) $= \frac{50}{6\ x\ 10}$ x 100% = 83.3%

Smoothness index (SI) $= \sqrt{4 + 1 + 1 + 9 + 9}$ = 4.89

Beralih ke fase 2 untuk memperbaiki solusi pada fase 1.

GOAL = $\frac{10-7}{2}$ = 1.5

Perpindahan elemen 7 ke *workstation* V dan elemen 9 ke *workstation* 6 (elemen 7 merupakan pendahulu elemen 8 dalam *worksatation* V). perpindahan tersebut akan mengurangi waktu proses maksimum menjadi 9, sehingga CT=9.

Line efficient (LE) $= \frac{50}{6\ x\ 9}$ x 100% = 92.6%

Smoothness index (SI) $= \sqrt{4}$ = 2

Tabel 2.10 Penugasan Pada Fase 1

WS	Elemen i	Ti	Waktu WS (ST)	CT - ST
I	1	5	8	2
	2	3		
II	4	3	9	1
	5	6		
III	3	4	9	1
	6	5		
IV	10	4	10	0
	11	4		
	7	2		
V	8	6	7	3
	9	1		
VI	12	7	7	3

(Sumber: Elsayed, 1994)

2.4.4 Metode *Rank and Assign Heuristic (RA)*

Metode *rank and assign heuristic* serupa dengan metode *Immediate Update First-Fit Heuristic* (IUFF) dan proses pengerjaannya menyerupai metode *Helgeson-Birnie*. Langkah-langkah metode *rank and assign heuristic* sebagai berikut: (Elsayed, 1994)

1. Hitung fungsi *score* setiap elemen kerja.
2. Urutkan semua elemen kerja berdasarkan nilai fungsi *score*-nya dari yang terbesar.
3. Kelompokkan elemen kerja-elemen kerja pada stasiun kerja dengan memperhatikan *precedence* diagram dan batasan CT.
4. Bila dimungkinkan untuk melakukan perpindahan atau pertukaran elemen kerja, lakukanlah perpindahan atau pertukaran elemen kerja dengan memperhatikan *precedence* diagram dan batasan CT.

Contoh kasus pada gambar 2.1:

Solusi

Membandingkan hasil metode *Helgeson-Birnie* dengan metode *rank and assign heuristic.* Metode RA menghasilkan bobot peringkat elemen seperti yang terlihat pada tabel 2.11. Contoh elemen 1 = (5 + 3 + 4) + (5 + 2 + 6 + 7) + (5 + 1 + 7) + (5 + 4 + 4 + 7) + (5 + 3 + 6) + (5 + 2 + 6 + 7) + (5 + 1 + 7) + (5 + 4 + 4 + 7) = 132

Tabel 2.11 Urutan Bobot Operasi Elemen Berdasarkan *Score* Metode RA

Peringkat	**Elemen Kerja (i)**	***Score***
1	1	132
2	4	62
3	2	60
4	5	59
5	3	58
6	6	53
7	7	15
8	10	15
9	8	12
10	11	11
11	9	8
12	12	7

(Sumber: Elsayed, 1994)

Penugasan elemen-elemen kerja ke *worsktations* menggunakan metode RA seperti pada tabel 2.12 berikut ini:

Tabel 2.12 Elemen-elemen Penugasan Untuk *Workstations* (CT=10)

WS	Elemen i	Ti	Waktu WS (ST)	CT - ST
I	1	5	8	2
	4	3		
II	2	3	9	1
	5	6		
III	3	4	9	1
	6	5		
IV	7	2	6	4
	10	4		
V	8	6	10	0
	11	4		
VI	9	1	8	2
	12	7		

(Sumber: Elsayed, 1994)

Line efficient (LE) $= \frac{50}{6\ x\ 10}$ x 100% = 83.3%

Smoothness index (SI) $= \sqrt{4 + 1 + 1 + 16 + 4}$ = 5.09

2.4.5 Metode *Mathematical Programming Formulation*

Masalah keseimbangan lintasan yang akan dibahas terdiri dari N pekerjaan, setiap pekerjaan i mempunyai durasi waktu sebesar ti, sekumpulan pekerjaan terdahulu yaitu Pi, dan sekumpulan pekerjaan sesudahnya yaitu Si. Setiap pekerjaan ditugaskan pada satu stasiun kerja j dalam sekumpulan stasiun kerja M. Dengan demikian solusi optimal dari masalah ini dapat diselesaikan dengan metode optimasi yang menggunakan formulasi-formulasi matematis sebagai berikut: (Elsayed, 1994)

$$Minimize\ Z = \sum_{j=1}^{M} f_j X_j + cC \quad \text{.....(7)}$$

Kendala

$$\sum_{i \in N} t_i X_{ij} - C \leq 0, j \in M \quad \text{.....(8)}$$

$$\sum_{i \in M} X_{ij} = 1, i \in N \quad \text{.....(9)}$$

$$X_{ik} \leq \sum_{j=1}^{k} Xhj, k \in M, j \in N, h \in P_i \quad \text{.....(10)}$$

$$\sum_{i \in Wj} Xij - \|Wj\| Xj \leq 0, J = 1, 2, \ldots, M \quad \text{.....(11)}$$

$$\sum_{i=Ea}^{La} iX_{ai} - \sum_{j=Eb}^{Lb} jX_{bj} \leq 1 \quad \text{.....(12)}$$

Dimana :

fj = waktu proses dalam satu stasiun kerja j

c = waktu siklus.

Xij = variable biner 0/1, dimana I menunjukkan pekerjaan dan j menunjukkan stasiun kerja. Xij bernilai 1 jika pekerjaan i ditugaskan pada stasiun j dan sebaliknya bernilai 0.

Xj = variable biner 0/1. xj bernilai 1 jika stasiun j ada dan sebaliknya bernilai 0.

ti = waktu proses pekerjaan i.

Wj = Kumpulan pekerjaan yang akan ditugaskan pada stasiun kerja;
|Wj| adalah bagian dari Wj

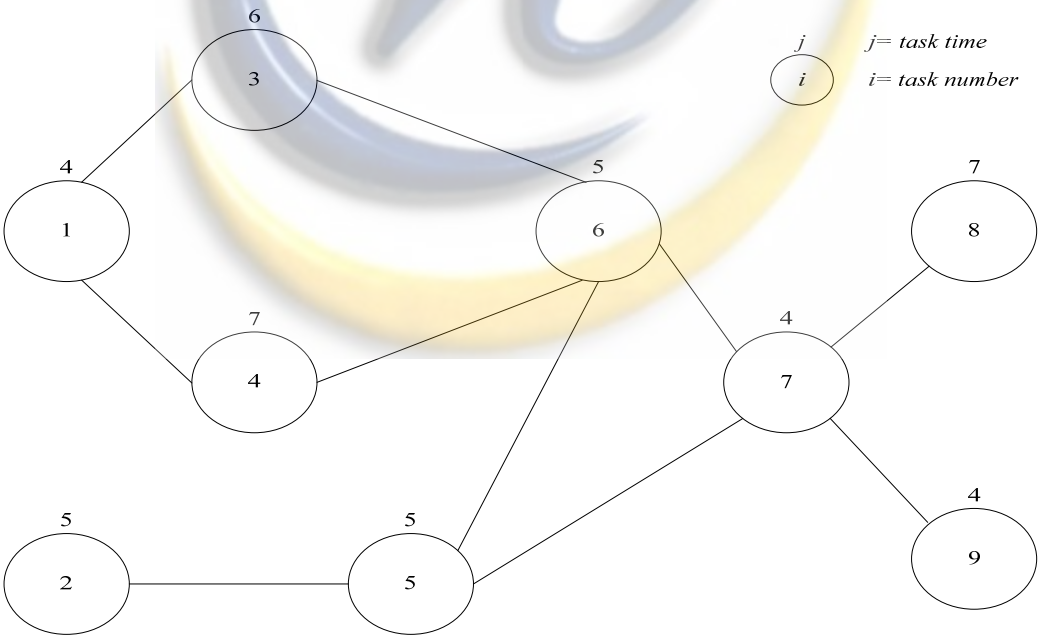
Ei = Stasiun kerja pertama yang dapat menerima penugasan

Li = Stasiun kerja terakhir yang dapat menerima penugasan.

Pembatas (8) menyatakan bahwa waktu stasiun kerja tidak melebihi waktu siklusnya. Pembatas (9) memaksa satu pekerjaan hanya dapat ditugaskan pada satu stasiun kerja. Pembatas (10) menyatakan bahwa kondisi precedence tidak terlanggar. Pembatas (11) memaksa penyediaan stasiun kerja paling minimum. Pembatas (12) menyatakan batas-batas waktu siklus yang diperbolehkan (batas bawah dan batas atas dari waktu siklus).

Contoh kasus pada gambar 2.2:

Masalah garis perakitan yang diberikan dalam Deckro adalah untuk meberikan penilaian produksi dengan waktu siklus C_1 = 15 dan C_2 = 20 yang digunakan. Jumlah maksimum *workstation* yang digunakan adalah 9 untuk batasan *layout*. Koefisiensi untuk fungsi objektif adalah $f_1 = f_2 = f_3$ = 15 dan $f_4 = f_5$ = 20 dan c = 3. Elemen 6 dapat menggantikan elemen 3 dan posisinya dekat dengan elemen 3. Tujuannya adalah meminimasi jumlah *workstation,* penugasan elemen-elemen ke *workstations* dan meminimasi waktu siklus. Hubungan elemen pendahulu dan waktu elemen kerja seperti pada gambar 2.2 dan tabel 2.13 berikut ini:



Gambar 2.2 *Precendence* Diagram Kasus 2

(Sumber: Elsayed, 1994)

Tabel 2.13 *Element Time* dan *Precedence Requirement*

Elemen Kerja	*Element Time*	*Direct Predecessor*	Ei	Li	*Assignment of Zero-One Variable*
1	4	-	1	4	$X_{11}, X_{12}, X_{13}, X_{14}$
2	5	-	1	4	$X_{21}, X_{22}, X_{23}, X_{24}$
3	6	1	1	4	$X_{31}, X_{32}, X_{33}, X_{34}$
4	7	1	1	4	$X_{41}, X_{42}, X_{43}, X_{44}$
5	5	2	1	4	$X_{51}, X_{52}, X_{53}, X_{54}$
6	10	3,4,5	2	4	X_{62}, X_{63}, X_{64}
7	6	5,6	2	5	$X_{72}, X_{73}, X_{74}, X_{75}$
8	1	7	2	5	$X_{82}, X_{83}, X_{84}, X_{85}$
9	4	7	2	5	$X_{92}, X_{93}, X_{94}, X_{95}$

(Sumber: Elsayed, 1994)

Solusi

Formulasi masalah

$$\text{Minimasi} = \quad 15X_1 + 15X_2 + 15X_3 + 20X_4 + 20X_5 + 3C$$

Kendala

Cycle time constraints

$$4X_{11} + 5X_{21} + 6X_{31} + 7X_{41} + 5X_{51} - C <= 0$$

$$4X_{12} + 5X_{22} + 6X_{32} + 7X_{42} + 5X_{52} + 10X_{62} + 6X_{72} + X_{82} + 4X_{92} - C <= 0$$

$$4X_{13} + 5X_{23} + 6X_{33} + 7X_{43} + 5X_{53} + 10X_{63} + 6X_{73} + X_{83} + 4X_{93} - C <= 0$$

$$4X_{14} + 5X_{24} + 6X_{34} + 7X_{44} + 5X_{54} + 10X_{64} + 6X_{74} + X_{84} + 4X_{94} - C <= 0$$

$$6X_{75} + X_{85} + 4X_{95} - C <= 0$$

Completion constraints

$$X_{11} + X_{12} + X_{13} + X_{14} = 1$$

$$X_{21} + X_{22} + X_{23} + X_{24} = 1$$

$$X_{31} + X_{32} + X_{33} + X_{34} = 1$$

$$X_{41} + X_{42} + X_{43} + X_{44} = 1$$

$$X_{51} + X_{52} + X_{53} + X_{54} = 1$$

$$X_{62} + X_{63} + X_{64} = 1$$

$$X_{72} + X_{73} + X_{74} + X_{75} = 1$$

$X_{82} + X_{83} + X_{84} + X_{85} = 1$

$X_{92} + X_{93} + X_{94} + X_{95} = 1$

Precedence constraints

Elemen 1 mendahului elemen 3

- $X_{11} + X_{31} <= 0$
- $X_{11} - X_{12} + X_{32} <= 0$
- $X_{11} - X_{12} - X_{13} + X_{33} <=0$
- $X_{11} - X_{12} - X_{13} - X_{14} + X_{34} <=0$

Elemen 1 mendahului elemen 4

- $X_{11} + X_{41} <= 0$
- $X_{11} - X_{12} + X_{42} <= 0$
- $X_{11} - X_{12} - X_{13} + X_{43} <= 0$
- $X_{11} - X_{12} - X_{13} - X_{14} + X_{44} <= 0$

Elemen 2 mendahului elemen 5

- $X_{21} + X_{51} <= 0$
- $X_{21} - X_{22} + X_{52} <= 0$
- $X_{21} - X_{22} - X_{23} + X_{53} <= 0$
- $X_{21} - X_{22} - X_{23} - X_{24} + X_{54} <= 0$

Elemen 3 mendahului elemen 6

- $X_{31} - X_{32} + X_{62} <= 0$
- $X_{31} - X_{32} - X_{33} + X_{63} <= 0$
- $X_{32} - X_{32} - X_{33} - X_{34} + X_{64} <= 0$

Elemen 4 mendahului elemen 6

- $X_{41} - X_{42} + X_{62} <= 0$
- $X_{41} - X_{42} - X_{33} + X_{63} <= 0$
- $X_{42} - X_{42} - X_{33} - X_{34} + X_{64} <= 0$

Elemen 5 mendahului elemen 6

- $X_{51} - X_{52} + X_{62} <= 0$
- $X_{51} - X_{52} - X_{53} + X_{63} <= 0$
- $X_{51} - X_{52} - X_{53} - X_{54} + X_{64} <= 0$

Elemen 5 mendahului elemen 7

- $X_{51} - X_{52} + X_{72} <= 0$
- $X_{51} - X_{52} - X_{53} + X_{73} <= 0$
- $X_{51} - X_{52} - X_{53} - X_{54} + X_{74} <= 0$
- $X_{51} - X_{52} - X_{53} - X_{54} - X_{75} <= 0$ (*redundant*)

Elemen 6 mendahului elemen 7

- $X_{62} + X_{72} <= 0$
- $X_{62} - X_{63} + X_{73} <= 0$
- $X_{62} - X_{63} - X_{64} + X_{74} <= 0$
- $X_{62} - X_{63} - X_{64} + X_{75} <= 0$ (*redundant*)

Elemen 7 mendahului elemen 8

- $X_{72} + X_{82} <= 0$
- $X_{72} - X_{73} + X_{83} <= 0$
- $X_{72} - X_{73} - X_{74} + X_{84} <= 0$
- $X_{72} - X_{73} - X_{74} - X_{75} + X_{85} <= 0$

Elemen 7 mendahului elemen 9

- $X_{72} + X_{92} <= 0$
- $X_{72} - X_{73} + X_{93} <= 0$
- $X_{72} - X_{73} - X_{74} + X_{94} <= 0$
- $X_{72} - X_{73} - X_{74} - X_{75} + X_{95} <= 0$

Workstation constraints

$X_{11} + X_{21} + X_{31} + X_{41} + X_{51} - 5X_1 <= 0$

$X_{12} + X_{22} + X_{32} + X_{42} + X_{52} + X_{62} + X_{72} + X_{82} + X_{92} - 9X_2 <= 0$

$X_{13} + X_{23} + X_{33} + X_{43} + X_{53} + X_{63} + X_{73} + X_{83} + X_{93} - 9X_3 <= 0$

$X_{14} + X_{24} + X_{34} + X_{44} + X_{54} + X_{64} + X_{74} + X_{84} + X_{94} - 9X_4 <= 0$

$X_{75} + X_{85} + X_{95} - 3X_5 <= 0$

Zoning constraint

$2X_{62} + 3X_{63} + 4X_{64} - X_{31} - 2X_{32} - 3X_{33} - 4X_{34} <= 1$

Pemecahan masalah tentang pemrograman linier menggunkana LINDO atau SAS? OR dalam memberikan penugasan elemen-elemen ke *worsktations* yang terlihat pada tabel 2.14. Waktu siklus yang optimal adalah 20 dan jumlah *workstations* yang optimal adalah 3 dan fungsi objektifnya adalah 105.

Tabel 2.14 Penugasan Elemen Ke *Workstations*, CT=20

WS	Elemen i	Ti	Waktu WS (STk)	CT - STk
1	1	4	17	3
	3	6		
	4	7		
2	2	5	20	0
	5	5		
	6	10		
3	7	6	11	9
	8	1		
	9	4		

(Sumber: Elsayed, 1994)

Line efficient (LE) $= \frac{48}{3\ x\ 20}$ x 100% = 80%

Smoothness index (SI) $= \sqrt{9 + 0 + 81}$ = 9.48

Ketika koefisien fungsi objektifitas berubah untuk f_1 sampai f_5 menjadi 1 dan c =3, *cycle time* optimal adalah 15, jumlah *workstations* menjadi 4 dan fungsi objektifitas menjadi 49. Penugasan optimal eleme-elemen kerja ke *workstations* terlihat pada tabel 2.15 berikut ini:

Tabel 2.15 Penugasan Elemen Ke *Workstations*, CT=15

WS	Elemen i	Ti	Waktu WS (ST)	CT - ST
1	1	4	15	0
	2	5		
	3	6		
2	4	7	7	8
3	5	5	15	0
	6	10		
4	7	6	11	4
	8	1		
	9	4		

(Sumber: Elsayed, 1994)

Line efficient (LE) $= \frac{48}{4\ x\ 15}$ x 100% = 80%

Smoothness index (SI) $= \sqrt{0 + 64 + 0 + 16}$ = 8.94

2.4.6 *The Single-Row Machine Layout Problem by Genetic Algorithm*

Genetic Algorithm (GA) adalah metode probabilistik untuk mencari teknik dasar dalam bekerja terhadap suatu pekerjaan dari genetika alam dan prinsip-prinsip evolusi. Ini pertama kali diusulkan oleh Holland, 1975 dalam Miao Tzu Lin, 2008 dan telah digunakan disejumlah aplikasi optimasi yang beragam. GA menggunakan pencarian acak yang diarahkan untuk mencari solusi global optimal. Mereka banyak unggul dari teknik gradien turunan karena mereka memiliki kemampuan untuk menemukan solusi global optimal untuk fungsi objektif multimodal. Dengan demikian, GA cocok untuk aplikasi pada fungsi optimasi nonlinier dan masalah pemrograman nonlinier (Man et al, 1996; Srinivas dan Patnaik, 1996 dalam Miao Tzu Lin, 2008).

GA bekerja dengan populasi individu yang mewakili solusi potensial dalam suatu masalah. Setiap individu biasanya diwakili oleh rangkaian karakter tunggal. Pada setiap iterasi algoritma, nilai kesesuaian *f (i)* dihitung untuk masing-masing individu saat ini. Berdasarkan fungsi kesesuaian, sejumlah individu yang dipilih merupakan orang tua yang potensial. Dua individu baru bisa diperoleh dari dua orang tua dengan memilih titik acak di sepanjang tali, membelah kedua rangkaian pada saat itu dan kemudian bergabung dengan bagian depan satu orangtua ke bagian belakang orang tua lain dan sebaliknya, proses ini biasanya disebut penyebrangan. Pada tingkat ini operasi diberi nilai khas antara 0,6-1,0 (Man et al., 1996 dalam

Miao Tzu Lin, 2008). Individu juga bisa berubah melalui mutasi acak ketika unsur-unsur dalam sebuah rangkaian diubah langsung pada probabilitas yang lebih kecil dengan nilai khas kurang dari 0,1 (Man et al, 1996 dalam Miao Tzu Lin, 2008). Proses penyebrangan dan mutasi secara kolektif disebut sebagai reproduksi. Hasil akhirnya adalah sebuah populasi baru (atau generasi berikutnya) dan mengulangi seluruh proses. Seiring waktu, algoritma ini mengarah untuk konvergensi dalam populasi dengan variasi yang lebih sedikit dan lebih sedikit antara individu. Ketika GA bekerja dengan baik, populasi menyatu untuk solusi menyelesaikan masalah optimasi yang baik yang mendasar dan individu terbaik dalam populasi dari sekian banyak generasi mungkin akan dekat dengan optimum global.

Singkatnya, sebuah GA bekerja sebagai berikut: (Miao Tzu Lin, 2008)

- Buat generasi awal.
- Mengevaluasi kesesuaian setiap individu pada generasi awal.
- Lakukan langkah-langkah berikut sampai kondisi penghentian benar:
    - Membuat individu baru dari individu kawin dalam arus pembangkit menggunakan operator genetik yang dipilih, *crossover* dan mutasi;
    - Mengevaluasi kesesuaian yang baru yang dibuat oleh setiap individu;
    - Menciptakan generasi baru dengan memasukkan yang baru dan menghapus yang lama individu dalam generasi sekarang.
- Kembali kepada individu terbaik (*s*).

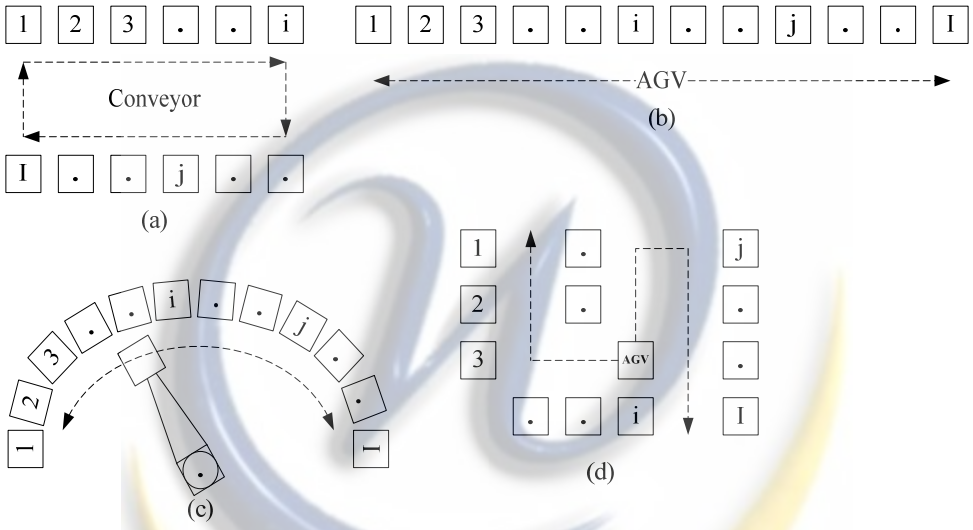
Mendiskusikan GA untuk masalah *Assembling Line Balancing* (ALB) dalam hal lima komponen: representasi, inisialisasi, evaluasi, operator dan parameter.

Model Matematika *From-To Matrix*

Tata letak mesin yang digunakan pada industri manufaktur umumnya adalah *single-row* seperti yang ditunjukkan pada gambar 2.3. Jarak antara stasiun sebagai *from-to matrix* ditampilkan dalam persamaan 13: (Miao Tzu Lin, 2008)

$$T_{IxI} = [tij] = \begin{bmatrix} t11 & \cdots & t1j & \cdots & t1I \\ \vdots & & \vdots & & \vdots \\ ti1 & \cdots & tij & \cdots & tiI \\ \vdots & & \vdots & & \vdots \\ tI1 & \cdots & tIj & \cdots & tII \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(13)$$

I merupakan jumlah stasiun kerja, t_{ij} merupakan jarak dari stasiun kerja i ke stasiun kerja j (meter)



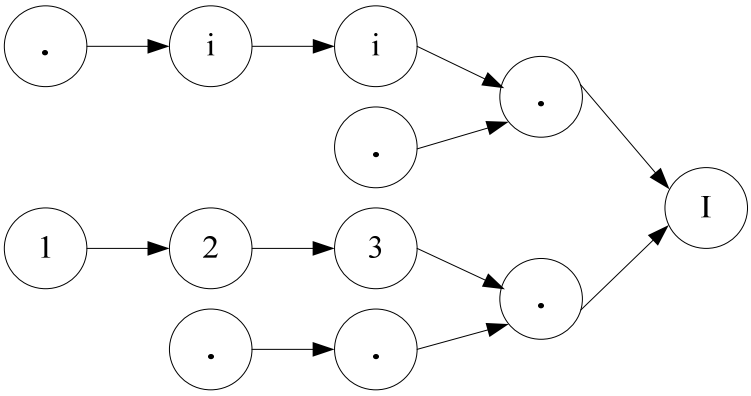
Gambar 2.3 Bentuk *Layout* Mesin

(Sumber: Miao Tzu Lin, 2008)

Precedence Diagram dan *Matrix*

Diagram didahulukan dengan lingkaran dan arah panah untuk menunjukkan hubungan antara stasiun kerja ditunjukkan oleh gambar 2.4. Hubungan antara stasiun kerja sebagai prioritas pada *precendence matrix* ditunjukkan dalam persamaan 14: (Miao Tzu Lin, 2008)

$$P_{IxI} = [pij]_{IxI} = \begin{bmatrix} P11 & \cdots & P1j & \cdots & P1I \\ \vdots & & \vdots & & \vdots \\ Pi1 & \cdots & Pij & \cdots & PiI \\ \vdots & & \vdots & & \vdots \\ PI1 & \cdots & PIj & \cdots & PII \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(14)$$



Gambar 2.4 *Precedence Diagram*

(Sumber: Miao Tzu Lin, 2008)

Algoritma Jarak Perpindahan Antar Pemotongan Bahan

Berdasarkan *from-to matrix* dan *precedence matrix*, diperoleh total jarak *matrix* seperti yang ditunjukkan pada persamaan 15 dan total jarak D ditunjukkan pada persamaan 16 (Miao Tzu Lin, 2008):

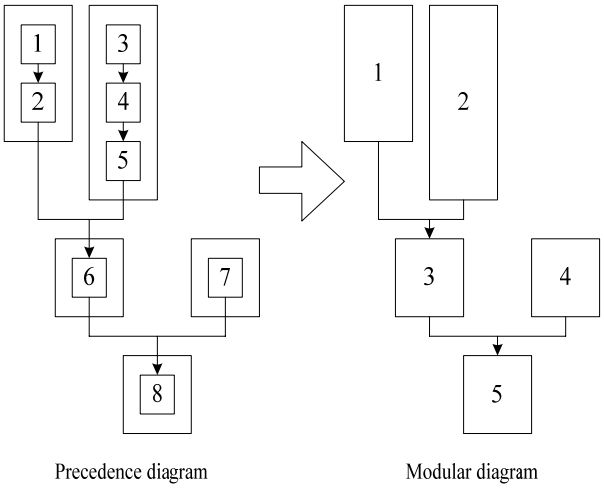
$$M_{IxI} = [mij]_{IxI} = [tij\ X\ pij]_{IxI} \quad \text{.....(15)}$$

$$D = \sum_{i=1}^{I} \sum_{j=1}^{I} mij \quad \text{.....(16)}$$

Hierarchical Order-Based Genetic Algorithm

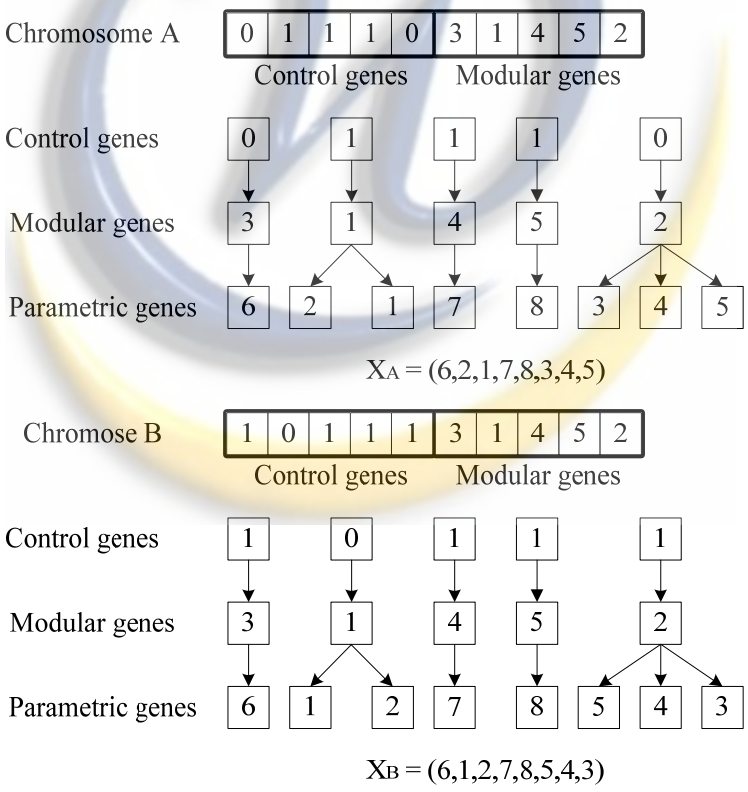
Kromosom (*Chromosomes*)

Seperti yang ditunjukkan bagian kiri pada gambar 2.5, menurut manufaktur urutan baris sub-perakitan, tata letak mesin yang berdekatan memiliki minimal jarak yang modular. Garis sub-perakitan diurutan baru dan ketertiban ditampilkan dibagian kanan dari diagram modular pada gambar 2.5. Hirarkis algoritma genetika berdasarkan pesanan adalah kromosom terdiri dari gen kontrol dan modular gen yang digunakan untuk memperoleh gen parametrik seperti yang ditunjukkan pada gambar 2.6.



Gambar 2.5 *Precendence Relationship of Modular Order*

(Sumber: Miao Tzu Lin, 2008)



Gambar 2.6 *Chromosomes Of The Hierarchical Order-Based Genetic Algorithm*

(Sumber: Miao Tzu Lin,2008)

Populasi Awal (*Initial Population*)

Sebuah kolam populasi kromosom dapat ditetapkan secara acak pada awalnya. Sebuah konvensional algoritma genetika biner diaplikasikan pada gen kontrol. Gen-gen modular diadopsi algoritma genetika berbasis pesanan untuk mengidentifikasi daftar *order-based* seperti satu, dua, tiga,. . . , I. Posisi kromosom dipilih sesuai dengan nomor acak dari daftar *order-based* dan kemudian dihapus dari daftar. Prosedur ini diulang sampai daftar *order-based* dikosongkan.

Fungsi Kebugaran (*Fitness Function*)

Penelitian ini mengkonversi jarak total menjadi fungsi fitness dengan menggunakan jumlah waktu positif yang mempunyai jumlah kebalikan dari fungsi objektif seperti yang ditunjukkan dalam persamaan 17 (Miao Tzu Lin, 2008); satu dengan fungsi *fitness* yang lebih besar dianggap optimal.

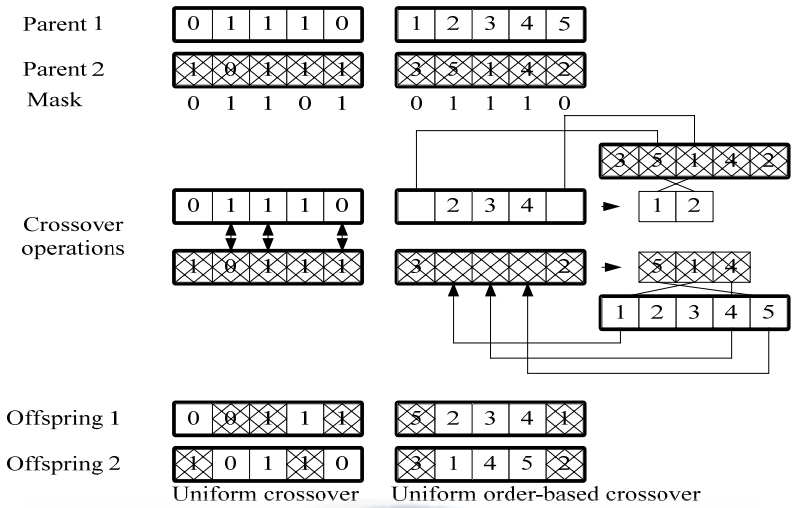
$$\text{Fit } (D) = 1{,}000 \text{ X } \frac{1}{D} \qquad \text{.....(17)}$$

Roulette Wheel Selection

Roulette Wheel Selection adalah salah satu teknik yang paling umum digunakan untuk mekanisme pemilihan proporsional. Kami menempatkan fungsi *fitness* dalam pesanan dan membagi nilai fungsi individual dengan nilai fungsi total. Rasio yang diperoleh adalah digambarkan sebagai roda rolet dan kromosom terkuat dan menempati interval yang terbesar. Penomoran dari interval yang lebih besar lebih mungkin untuk dipilih (Man et al.,2000 dalam Miao Tzu Lin, 2008).

Penyeberangan (*Crossover*)

Tidak semua kromosom dimasukkan ke dalam *crossover*, sehingga definisi Pc (probabilitas *crossover*) dengan nilai khas antara 0,6 ~ 1 (Man et al, 2000 dalam Miao Tzu Lin, 2008). Ini menghasilkan keturunan dari *parents* berdasarkan topeng *crossover* yang diolah secara acak. Gen-gen kontrol mengadopsi *uniform crossover*. Operasi ini ditunjukkan dalam bagian kiri gambar 2.7. Gen-gen modular mengadopsi *uniform order-based crossover*. Operasi ini ditunjukkan pada bagian kanan gambar 2.7 sebagai berikut:

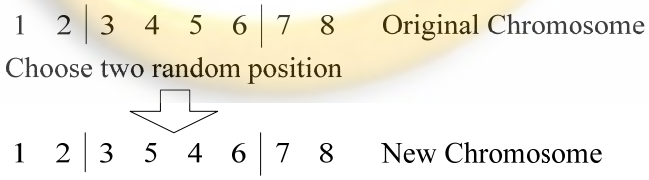


Gambar 2.7 *Crossover Of The Hierarchical Order-Based Genetic Algorithm*

(Sumber: Miao Tzu Lin, 2008)

Mutasi (*Mutation*)

Definisi Pm (probabilitas mutasi) dengan nilai khas kurang dari 0,1 digunakan untuk melarikan diri dari optimum lokal (Man et al, 2000 dalam Miao Tzu Lin, 2008). Gen-gen kontrol mengadopsi mutasi biner algoritma genetika konvensional. Gen-gen modular digunakan untuk memperebutkan sebuah sub-daftar mutasi seperti yang ditunjukkan pada gambar 2,8. Kami memilih secara acak dua bit dari bit *string* dan kromosom secara acak dari daftar yang tercatat. (Chan et al, 1998 dalam Miao Tzu Lin, 2008).



Gambar 2.8 *Scramble Sub-list Mutation*

(Sumber: Miao Tzu Lin, 2008)

Pemutusan Kondisi (*Termination Condition*)

Biasanya, kondisi penghentian ditampilkan dalam tiga aspek. Bila beberapa operasi waktu yang diperlukan dari kali evolusi, algoritma generasi diatur ke mengakhiri. terminasi juga inisiat ketika fungsi tujuan mencapai target nilai. Ketika kelompok ini ditunjukkan dengan

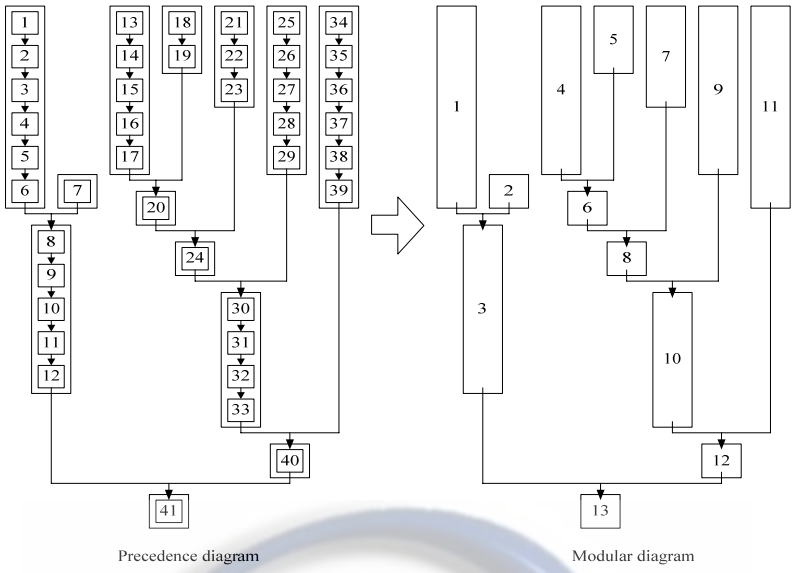
homogenitas, konvergensi disajikan dan itu adalah dekat dengan solusi optimal lokal. Penelitian ini mengadopsi pemutusan generasi algoritma.

Hasil dan Pembahasan

Untuk menguji efektifitas masalah ini, kami melakukan penelitian mengenai produksi kemeja pria yang sama dengan penelitian genetik algoritma (Chan et al, 1998 Miao Tzu Lin, 2008). Gambar 2.9 menunjukkan contoh diagram hubungan didahulukan. Kromosom dan struktur tersebut ditunjukkan pada gambar 2.10. Tugas masing-masing stasiun ditabulasi dalam tabel 2.16. Contoh kondisi ditunjukkan sebagai berikut: (Miao Tzu Lin, 2008)

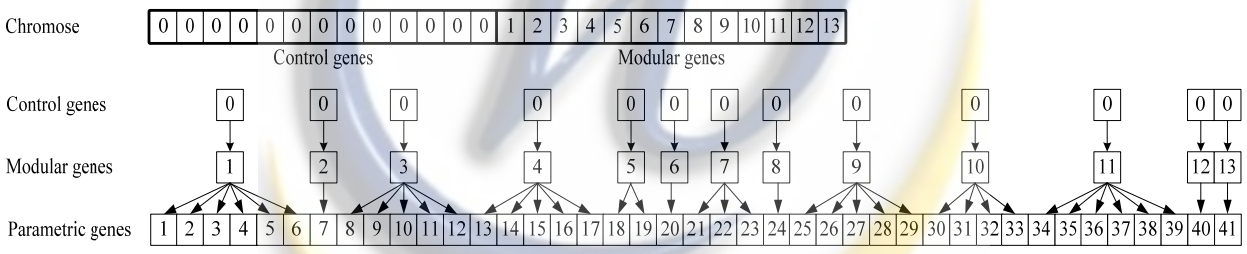
- Ukuran populasi: 50.
- Probabilitas crossover (Pc): 0,6.
- Probabilitas mutasi (Pm): 0,008.
- Pemutusan Kondisi: Pemutusan sampai generasi 200.

Menurut urutan proses kerja, tata letak mesin terdaftar sebagai (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41). Namun, sebelum menerapkan algoritma yang diusulkan dan memasukkan urutan persamaan, kami mengakuisisi total jarak antara potongan pemotongan, D1 ¼ 70 dan nilai kebugaran, Fit (D1) ¼ 14,29. Dengan menerapkan algoritma genetika berdasarkan pesanan sampai generasi 200, kami mengakuisisi mesin layout sebagai (1, 41, 40,39, 3, 4, 2, 5, 7, 8, 9, 10, 25, 26, 15, 16, 17, 21, 22, 23, 20, 19, 18, 24, 32, 11, 12, 13, 14, 27, 28, 29, 31, 6, 36, 38, 37, 35, 34, 33, 30), sehingga total jarak antara potongan pemotongan, D2 ¼ 68 dan nilai fitness, Fit (D2) ¼ 14,7. Setelah penerapan algoritma genetik hirarkis order-based, kromosom gen kontrol pada gambar 2.11 disajikan sebagai (0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1); gen modular sebagai (1, 3, 2, 7, 4, 9, 5, 6, 8, 10, 12, 13, 11). Dari gen kontrol dan gen modular kami mengakuisisi urutan perbaikan tata letak mesin sebagai (1, 2, 3, 4, 5, 6, 12, 11, 10, 9, 8, 7, 23, 22, 21, 17, 16, 15, 14, 13, 25, 26, 27, 28, 29, 18, 19, 20, 24, 30, 31, 32, 33, 40, 41, 39, 38, 37, 36, 35, 34), sehingga total jarak antara memotong buah, D3 ¼ 55, dan nilai kebugaran, Fit (D3) ¼ 18.18. Jarak total dan kebugaran nilai algoritma generasi diakuisisi oleh dua metode ditunjukkan seperti pada gambar 2.12 dan penilaian keefektifan ditingkatkan ditunjukkan seperti pada gambar 2.13.



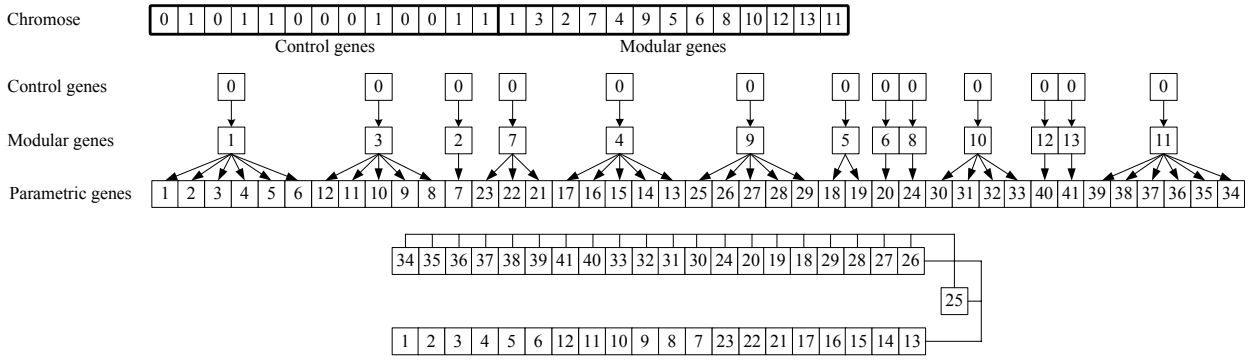
Gambar 2.9 *Precendence Diagram* Produksi Kemeja Pria

(Sumber: Miao Tzu Lin, 2008)



Gambar 2.10 *Chromosomes Hierarchical Order-Based Genetic Algorithm* Produksi Kemeja Pria

(Sumber: Miao Tzu Lin, 2008)



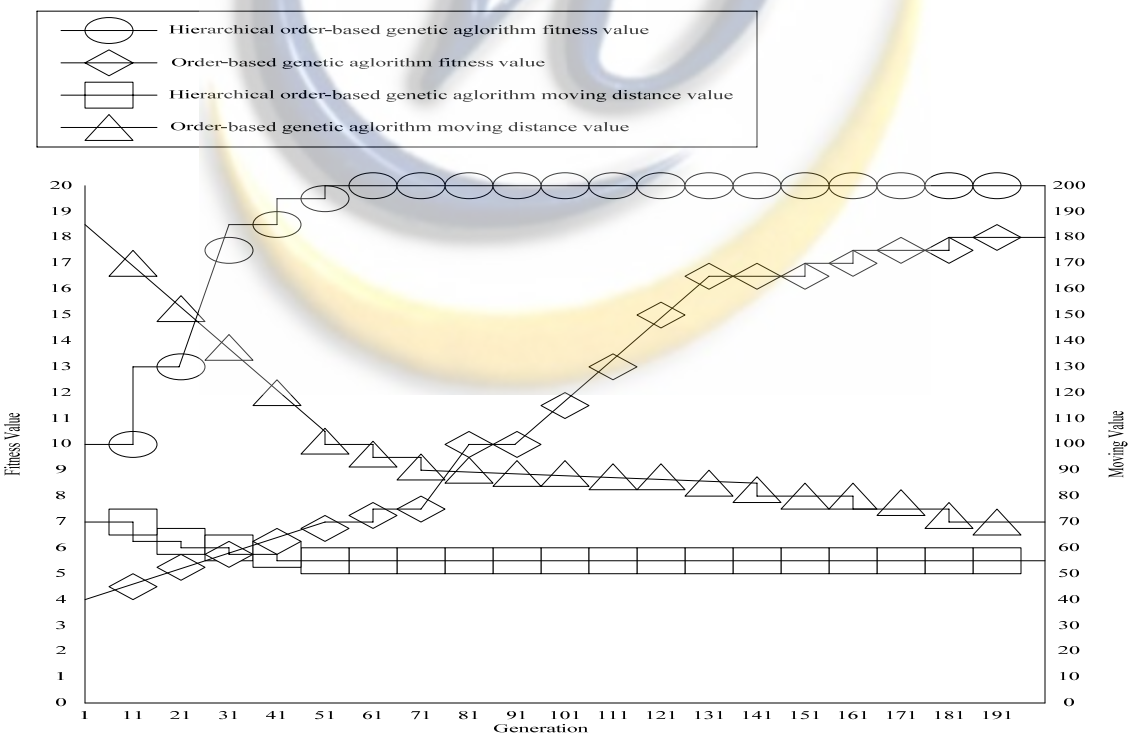
Gambar 2.11 Perbaikan *Layout* Mesin Berbentuk U

(Sumber: Miao Tzu Lin, 2008)

Tabel 2.16 Tugas Setiap Stasiun Kerja Dalam Memproduksi Kemeja Pria

Task No.	Task name	Task No.	Task name
1	Spot fuse collar fall	22	Set yoke label
2	Top fuse collar fall	23	Set yoke label
3	Sew collar stay pocket	24	Join shoilder
4	Runstitch collar band	25	Set sleeve under placket
5	Trim, tum and press collar fall	26	Set sleeve top placket
6	Topstitch collar band	27	Finish sleeve placket
7	Hem collar band	28	Sew sleeve placket buttonhole
8	Attach collar band	29	Sew sleeve placket button
9	Turn and press collar band	30	Set sleeve
10	Topstitch collar band	31	Topstitch armhole
11	Sew collar band buttonhole	32	join side seam
12	Sew collar band button	33	Hem bottom
13	Sew centre front placket	34	Hem cuff
14	Hem right front edge	35	Runstitch cuff
15	Trim neckline	36	Turn and press cuff
16	Sew centre front buttonhole	37	Topstitch cuff
17	Sew right front button	38	Sew cuff buttonhole
18	Hem pocket mount	39	Sew cuff buttonhole
19	Crease pocket	40	Set cuff
20	Set pocket	41	Set and close collar
21	Sew yoke pleats		

(Sumber: Miao Tzu Lin, 2008)



Gambar 2.12 Nilai Kebugaran dan Nilai Jarak Perpindahan Tiap Generasi

(Sumber: Miao Tzu Lin, 2008)

Tabel 2.17 Efektifitas Penilaian Dari Perbaikan *Layout* Mesin Berbentuk U

No	U-shape machine layout	Total distance *D* (m)	Assessment of improve effectiveness	Remark (fitness value)
1	Machine layout according to manufacturing order	*D* 1 = 70		Fit (*D* 1) = 14.29
2	Order-based genetic algorithm	*D* 2 = 68	improved by 29 percent than *D* 1	Fit (*D* 2) = 14.7
3	Hierarchical order-based genetic algorithm	*D* 3 = 55	improved by 21.4 percent than *D* 1; improved by 19.1 percent than *D* 2	Fit (*D* 3) = 18.18

(Sumber: Miao Tzu Lin, 2008)

Perbandingan *Hierarchical Genetic Algorithm Order-Based* dan *The Order-Based Genetic Algorithm*

Ketika membandingkan dua metode algoritma, kami menemukan suatu pemusatan/titik temu dari *the order-based genetic algorithm* pada generasi 178 dan *hierarchical genetic algorithm order-based* pada generasi 39 yang menunjukkan perpindahan jarak yang lebih singkat, nilai *fitness* yang lebih baik, pemusatan yang lebih cepat dan solusi optimal yang akurat dari *hierarchical genetic algorithm order-based*.