

**LABORATORIUM SISTEM PRODUKSI
PROGRAM STUDI TEKNIK INDUSTRI
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS KATOLIK PARAHYANGAN
BANDUNG
2017**

MODUL 5

PENGATURAN TATA LETAK LANTAI PRODUKSI DAN PERHITUNGAN BIAYA MATERIAL HANDLING

A. Tujuan Praktikum

1. Praktikan mampu menerapkan metode dasar dalam pengaturan mesin.
2. Praktikan mampu menggunakan *software* dalam pengaturan mesin.
3. Praktikan mampu melakukan perhitungan biaya *material handling*.

B. Landasan Teori

1. Klasifikasi Algoritma

Sebagian besar algoritma yang digunakan untuk merancang *layout*, dapat dikategorikan berdasarkan tipe *input* data yang dibutuhkan. Terdapat dua metode perancangan *layout* berdasarkan *input*-nya :

a. Metode Kuantitatif

Metode ini membutuhkan kriteria keputusan yang dapat diukur seperti biaya *material handling*, waktu tempuh pembeli, atau frekuensi pengangkutan seperti yang terdapat dalam *from-to-chart* (FTC). Contoh metode yang menggunakan data kuantitatif antara lain: CRAFT, MCRAFT, BLOCPAN, MIP, LAYOPT, COFAD.

b. Metode Kualitatif

Metode ini hanya menggunakan kriteria kualitatif seperti alasan keamanan atas bahaya kebakaran (misalnya bagian oven dengan bagian pengelasan) dan alasan lain yang tampak melalui sebuah *Activity Relationship Chart* (ARC). Contoh metode yang menggunakan *input* data kualitatif adalah: MIP, ALDEP, CORELAP, LAYOPT.

Selain pembagian berdasarkan tipe *input*, pembagian algoritma juga dapat dikelompokkan berdasarkan fungsi tujuannya. Terdapat dua fungsi tujuan yang biasanya mendasari sebuah perancangan (Tompkins et. al., 2003), yaitu:

a. *Distance-based objective*

Metode ini merupakan pendekatan yang memperhitungkan jarak dan aliran material dari suatu bagian ke bagian yang lain. Untuk melakukan pengolahan data lebih lanjut, akan lebih tepat bila data yang digunakan adalah FTC. Tujuan dari pendekatan ini adalah meminimasi biaya per satuan waktu untuk pergerakan antar departemen. Apabila dinyatakan secara matematis, maka fungsi tujuan *distance-based* adalah sebagai berikut:

$$\text{Min } Z = \sum_{i=1}^m \sum_{j=1}^m f_{ij} \cdot C_{ij} \cdot d_{ij} \quad (\text{Pers. 1})$$

Dengan:

f_{ij} = aliran dari departemen i ke departemen j,

C_{ij} = biaya yang dibutuhkan untuk memindahkan 1 unit barang dari departemen i ke departemen j,

d_{ij} = jarak dari departemen i ke departemen j.

b. *Adjacency-based objective*

Pendekatan yang dilakukan bertujuan memaksimalkan nilai dari *adjacency score* sebagai berikut :

$$\text{Min } Z = \sum_{i=1}^m \sum_{j=1}^m f_{ij} \cdot x_{ij} \quad (\text{Pers. 2})$$

Dengan:

f_{ij} = aliran dari departemen i ke departemen j,

x_{ij} = 1 , bila departemen i dan departemen j bersebelahan,

x_{ij} = 0 , bila departemen i dan departemen j tidak bersebelahan.

Dapat dilakukan perhitungan efisiensi pengaturan tata letak yang dihasilkan, dengan menggunakan rumus:

$$Z = \frac{\sum_{i=1}^m \sum_{j=1}^m f_{ij} \cdot x_{ij}}{\sum_{i=1}^m \sum_{j=1}^m f_{ij}} \quad (\text{Pers. 3})$$

Terdapat beberapa teknik yang dapat digunakan dalam penentuan *layout* secara terkomputerisasi, antara lain: CRAFT, BLOCPAN, MIP, LOGIC, dan MULTIPLE.

Pembagian algoritma yang ketiga, didasarkan pada format yang digunakan merepresentasikan *layout*. Terdapat dua buah format yang dapat digunakan dalam merepresentasikan *layout*. Kedua format tersebut adalah (Tompkins et. al., 2003):

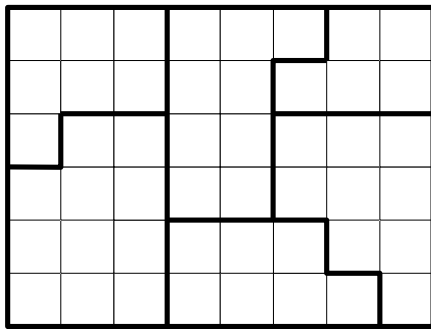
a. Representasi Diskrit (*Discrete Representation*)

Sebagian besar algoritma penentuan *layout*, menggunakan representasi diskrit ini dalam menggambarkan *layout*-nya. Melalui algoritma ini, komputer diijinkan untuk menyimpan dan memanipulasi *layout* sebagai sebuah matriks. Dengan jenis representasi ini, area dari setiap departemen dibulatkan ke jumlah *grid* bulat yang terdekat. Apabila ukuran *grid* terlalu besar, departemen dengan luas yang kecil akan memiliki *grid* dengan jumlah yang terlalu sedikit.

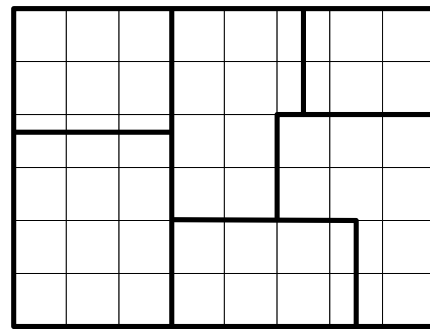
Ukuran dari *grid* menentukan resolusi yang dihasilkan akan semakin baik dan memberikan fleksibilitas bentuk departemen. Namun semakin kecil untuk ukuran *grid*, beban perhitungan yang harus dilakukan juga semakin besar. Oleh karena itu, penentuan ukuran *grid* secara tepat merupakan hal yang penting dalam algoritma dengan representasi diskrit. Contoh bentuk *layout* yang dihasilkan menggunakan representasi diskrit dilihat pada Gambar 1.

b. Representasi Kontinu (*Continuous Representation*)

Representasi ini tidak harus memiliki struktur *grid* yang tepat garis. Secara teoritis, bentuk representasi ini akan lebih mudah. Namun pengimplementasiannya pada komputer akan lebih sulit. Bentuk representasi ini tidak boleh digunakan untuk bentuk departemen yang *rectangular*, kecuali apabila bentuk departemen tersebut 'd disesuaikan' untuk mengakomodasi bentuk bangunan yang *nonrectangular*. Contoh bentuk *layout* yang dihasilkan menggunakan representasi kontinu dapat dilihat pada Gambar 2.

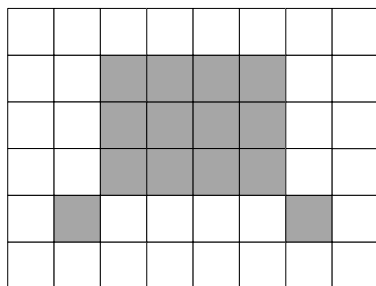


Gambar 1. Contoh Bentuk
Representasi *Layout* Secara Diskrit

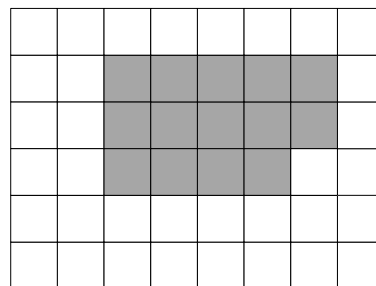


Gambar 2. Contoh Bentuk
Representasi *Layout* Secara Kontinu

Bentuk departemen memegang peranan penting dalam *computerized layout algorithms*. Departemen sendiri merupakan entitas yang tidak dapat dipisahkan dalam perancangan sebuah *layout*. Sebuah algoritma penentuan *layout* seharusnya tidak melakukan pemisahan (*split*) sebuah departemen menjadi dua atau lebih bagian. Namun apabila sebuah departemen terlalu besar, maka akan dipertimbangkan untuk melakukan pembagian terhadap departemen ini. Gambaran perbedaan departemen yang *split* dan *unsplit* dapat dilihat pada Gambar 3 dan Gambar 4.



Gambar 3. Contoh Departemen *Split*



Gambar 4. Contoh Departemen *Unsplit*

Selain berdasarkan tiga kriteria di atas, algoritma yang digunakan untuk merancang *layout*, dapat dikategorikan berdasarkan fungsi utamanya, yaitu :

1. *Layout Improvement*

Algoritma tipe *improvement*, biasanya diawali dengan *layout* awal yang disediakan oleh analis, yang kemudian diolah untuk meningkatkan pencapaian fungsi tujuannya, melalui perubahan '*incremental*' pada *layout*.

2. *Layout Construction*

Algoritma tipe *construction* ini biasanya melakukan pengembangan *layout* dari nol (tidak ada *layout* awal yang diberikan). Algoritma ini dapat dibagi menjadi dua sesuai dengan asumsi yang digunakan. Dua asumsi yang digunakan ialah:

i. Dimensi bangunan telah diberikan.

Jenis algoritma ini cocok digunakan ketika suatu operasi diaplikasikan ke suatu bangunan yang sudah ada.

ii. Dimensi bangunan tidak diberikan.

Jenis algoritma ini cocok digunakan ketika suatu operasi diaplikasikan ke tanah kosong (belum terdapat bangunan).

2. Metode *Graph-Based*

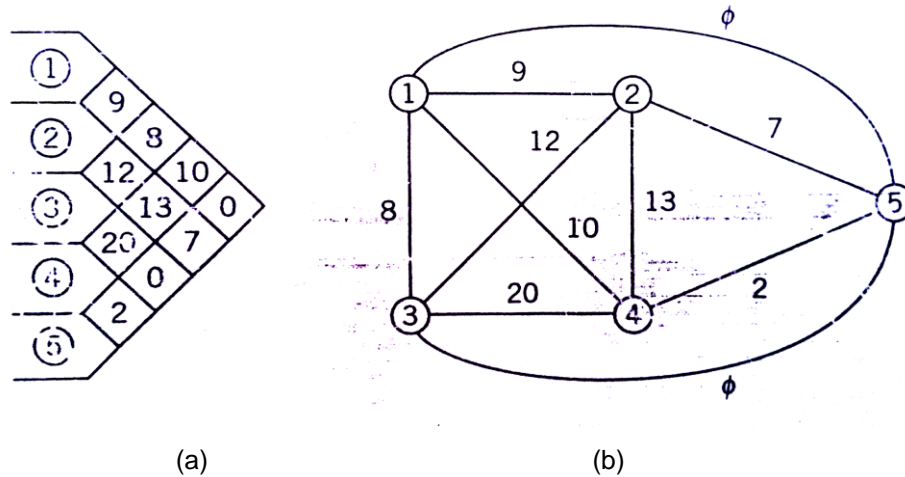
Metode *graph-based* merupakan algoritma tipe *layout construction*. Metode ini didasarkan pada *graph theory* dan biasanya digunakan dengan *adjacency-based objective* (Krejcirik, 1960s; Seppanen dan Moore, 1970s). Sebelum mendeskripsikan metode untuk menentukan *adjacency graph*, pertama-tama perlu dilakukan pengamatan sebagai berikut (Tompkins et. al., 2010):

1. *adjacency score* tidak memperhitungkan jarak maupun hubungan selain bagi departemen yang bersebelahan,
2. spesifikasi dimensi departemen tidak dipertimbangkan; panjang batas antara departemen yang bersebelahan tidak dipertimbangkan,
3. busur tidak saling berpotongan (*planarity*),
4. *adjacency score* sangat sensitif terhadap pemberian bobot.

Hubungan antardepartemen dapat direpresentasikan dalam bentuk *relationship chart* dan *relationship diagram*. Contoh *relationship chart* dan *relationship diagram* dapat dilihat pada Gambar 5. Pada praktikum Modul 5 ini, nilai pada *relationship chart* merupakan frekuensi perpindahan material antardepartemen yang didapat dari *From to Chart* (Modul 3).

Langkah iterasi pembuatan *adjacency graph* adalah sebagai berikut:

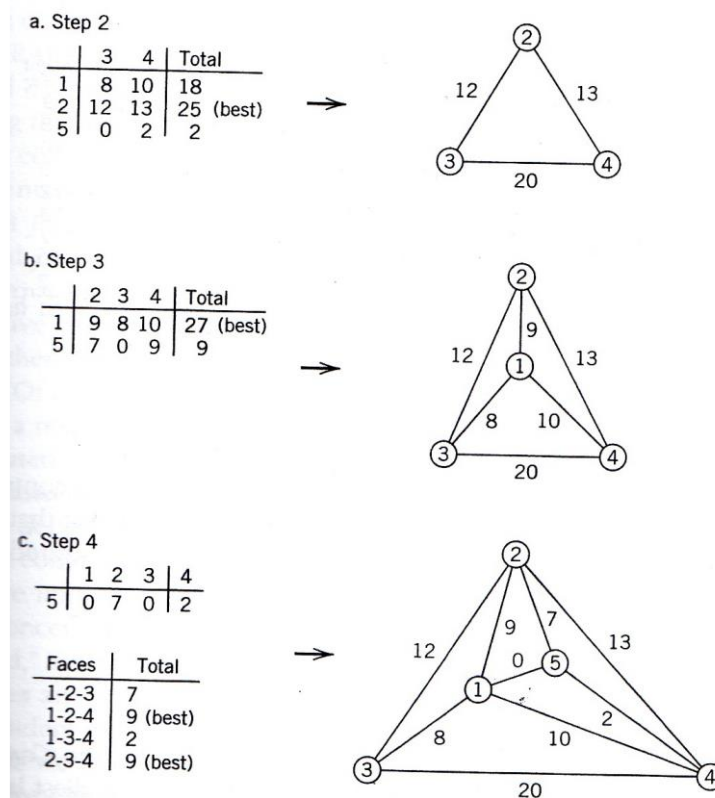
1. Pilih pasangan departemen dengan bobot terbesar,
2. Pilih departemen ke tiga yang akan masuk berdasarkan jumlah bobot terbesar sesuai dengan hubungannya dengan dua departemen awal,



Gambar 5. Contoh (a) *Relationship Chart* (b) *Relationship Diagram*

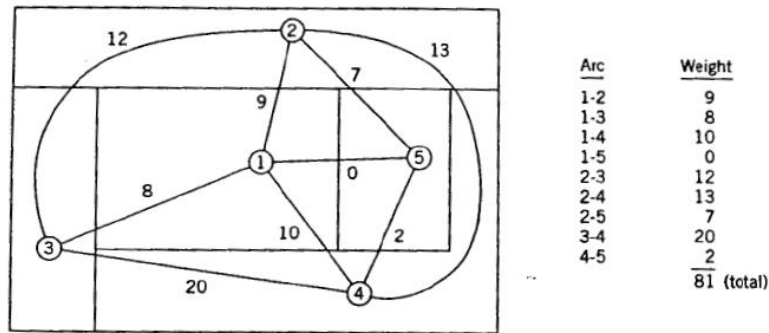
(sumber: Tompkins et. al., 2010)

3. Pilih departemen ke empat yang masuk ke dalam *face* (daerah yang dibatasi oleh grafik).
Ulangi untuk sisa departemen lainnya,
4. Bentuk *adjacency graph*,
5. Buat *block layout* yang sesuai dengan *adjacency graph* tersebut.



Gambar 6. Langkah Metode *Graph-Based*

(sumber: Tompkins et. al., 2010)



Gambar 7. Block Layout berdasarkan Adjacency Graph
(sumber: Tompkins et. al., 2010)

3. CRAFT

Metode CRAFT (*Computerized Relative Allocation of Facilities Techniques*) menggunakan data aliran yang terdapat FTC sebagai *input* dan membutuhkan *initial layout*. Untuk menghitung *layout cost*, digunakan *distance-based function* seperti pada Persamaan 1.

Pada metode CRAFT, penggambaran departemen tidak harus dalam bentuk *rectangular*, dan *layout* ditampilkan dalam bentuk *discrete*. Pengolahan selanjutnya dilakukan dengan pertukaran antar departemen. Yang perlu diperhatikan dalam metode ini adalah departemen yang ditukar haruslah bersebelahan atau memiliki luas area yang sama. Untuk melakukan pertukaran terdapat beberapa cara yang bisa digunakan, antara lain (Tompkins et. al., 2003):

1. *Two way*,
2. *Three way*,
3. *Two way then three way*,
4. *Three way then two way*.

Jarak yang diperhitungkan dalam metode CRAFT merupakan jarak dari *centroid* suatu departemen, ke *centroid* departemen lain yang ingin dihitung biayanya. Metode ini dapat mengakomodasi bentuk departemen yang tidak *rectangular*, sehingga memungkinkan meng-*capture layout* awal secara akurat. Hal ini merupakan kelebihan dari metode CRAFT. Namun, ada juga efek negatif dari hal ini, yaitu *layout* akhir yang dihasilkan seringkali memiliki bentuk yang kurang bagus atau kotak utuh. Hal ini menjadi kelemahan dari metode CRAFT.

Selain kelemahan dan kelebihan dari metode CRAFT yang telah disebutkan diatas, masih terdapat beberapa kelebihan dan kelemahan lainnya dari metode ini. Kelebihan tersebut antara lain (Tompkins et. al., 2003):

1. Dapat menggunakan *fixed layout departments*,
2. Tiap departemen bisa diatur supaya berbentuk *rectangulars*.

Sedangkan kelemahan lain dari metode ini adalah :

1. Bersifat sangat heuristik,

2. Hasil akhir sangat bergantung pada *initial layout*,
3. Pertukaran departemen hanya bisa dilakukan terhadap pasangan departemen yang bersebelahan atau berluas sama.

Agar area akhir yang dihasilkan secara keseluruhan bisa berbentuk kotak, terkadang digunakan *dummy departments*. Departemen *dummy* merupakan sebuah departemen yang tidak memiliki aliran atau interaksi dengan departemen lainnya, tetapi membutuhkan sejumlah area sesuai dengan yang ditentukan oleh perencana *layout*. Secara umum, departemen *dummy* memiliki beberapa kegunaan, antara lain (Tompkins et. al., 2003):

1. Mengisi ketidakteraturan bangunan.
2. Merepresentasikan penghalang atau area kosong pada pabrik (seperti tangga, eskalator, pilar).
3. Merepresentasikan area tambahan pada pabrik.
4. Alat bantu dalam mengevaluasi lokasi pada gang pada *layout* akhir.

Implementasi terkini dari CRAFT, dinamakan sebagai MICRO-CRAFT atau sering disingkat sebagai MCRAFT. Pada dasarnya, metode ini serupa dengan CRAFT, hanya saja MCRAFT dapat melakukan pertukaran dua departemen yang tidak bersebelahan.

4. MCRAFT

Pembuatan *layout* awal ini dilakukan dengan bantuan metode MCRAFT. Metode MCRAFT dipilih karena metode MCRAFT memiliki kesamaan karakteristik algoritma dengan metode CRAFT, yaitu sama-sama menggunakan teknik *pairwise exchange* untuk mendapatkan solusi optimal. Langkah pertama dalam membuat *layout* awal pada metode MCRAFT adalah menghitung jumlah kotak yang digunakan untuk merepresentasikan setiap departemen (*number of unit area template*). Setelah menghitung *number of unit area template*, kemudian ditentukan jumlah *band*, *sweep width*, dan ukuran bangunan yang akan digunakan. *Band* adalah lintasan besar yang membagi suatu area menjadi beberapa bagian. *Sweep width* adalah lebar pergerakan dalam penempatan departemen.

5. WinQSB

Layout awal pabrik yang diperoleh dari metode MCRAFT akan digunakan sebagai *input* untuk metode CRAFT dengan menggunakan *software WinQSB Facilities Location and Layout*. Untuk melakukan iterasi dengan menggunakan *software WinQSB* dibutuhkan *input* berupa FTC dan koordinat pengaturan mesin awal.

Software WinQSB mencari *total cost* minimum berdasarkan iterasi, dimana pada setiap iterasi dilakukan penukaran antara dua buah departemen, sehingga pada setiap iterasinya *total cost* akan selalu menjadi lebih kecil. Iterasi akan terus dilakukan hingga terjadi fenomena *split*

pada sebuah departemen tertentu. Fenomena *split* adalah fenomena dimana suatu departemen terpisah atau terpecah menjadi dua atau lebih bagian.

C. *Input dan Output Modul*

Input modul 5:

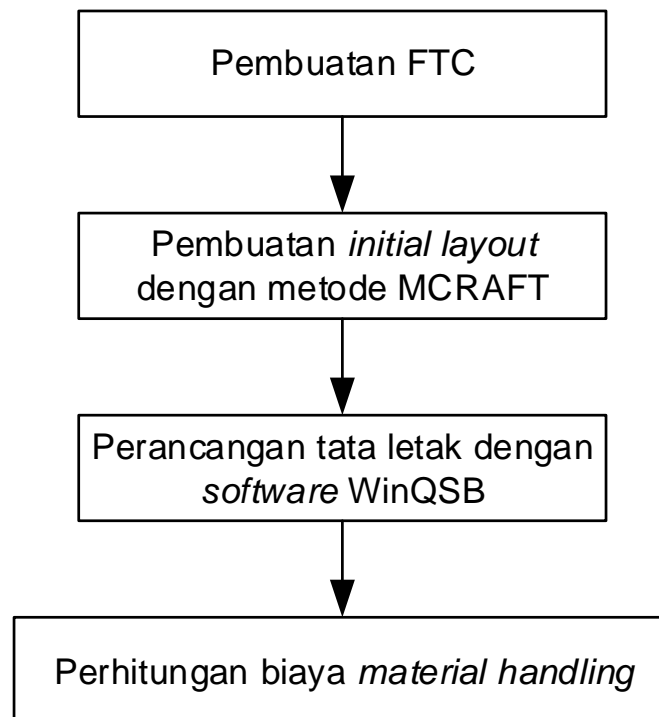
- Luas lantai *process layout* (Modul 1), luas lantai *GT Layout* (Modul 2), tabel data aliran material dan FTC (Modul 3), luas warehouse dan storage (modul 4)

Output modul 5:

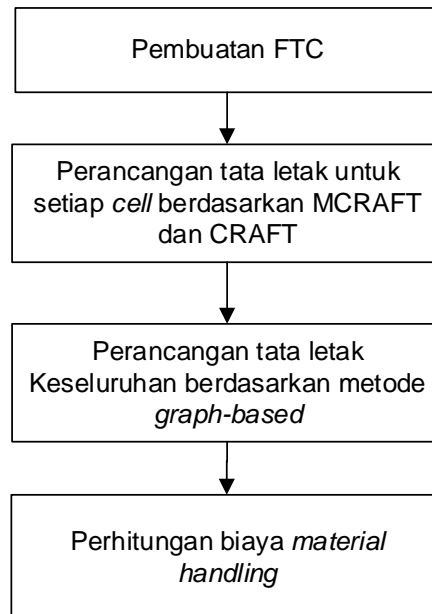
- Hasil akhir *process layout* dan *GT layout*.
- Hasil perhitungan *cost material handling* untuk *procees layout* dan *GT layout*

D. *Prosedur Pelaksanaan Praktikum*

Prosedur praktikum Modul 5 dapat dilihat pada Gambar 8 (*layout proses*) dan Gambar 9 (*layout GT*):



Gambar 8. Prosedur Praktikum Modul 5 untuk *Layout Proses*

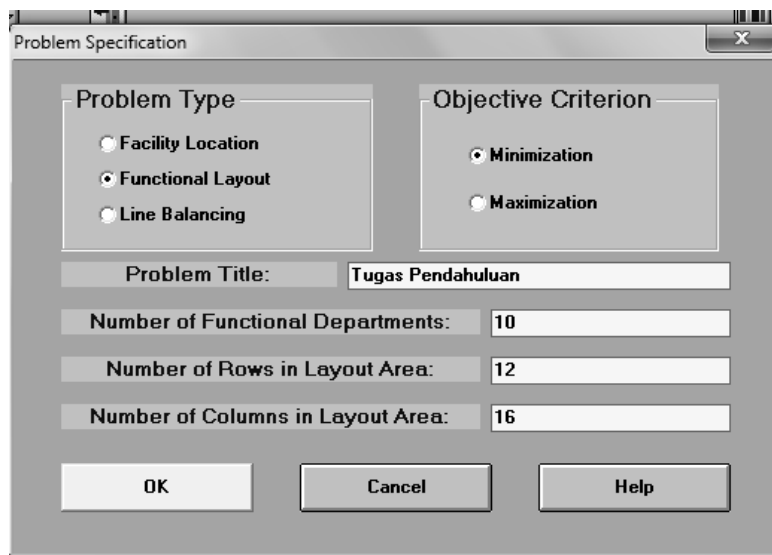


Gambar 9. Prosedur Praktikum Modul 5 untuk *Layout GT*

Langkah-langkah pengerjaan modul 5:

1. *Layout Proses (CRAFT)*:

- a. Membuat *layout* awal dengan metode MCRAFT dengan jumlah baris pada *band* sebanyak 3, dengan jumlah baris dan kolom ditentukan masing-masing kelompok sesuai luas lantai pabrik. Masukkan departemen sesuai urutan dari *storage* hingga *warehouse*. Gunakan ukuran 1 *unit area template* = 16 m².
- b. Menginput data ke dalam program WinQSB. Cara *input* data :
 - i. Buka program WinQSB, pilih *Facility Location and Layout*.
 - ii. Klik *File* → *New Problem*,
 - iii. akan keluar kotak dialog sebagai berikut :



Gambar 10. Tampilan Awal WinQSB untuk *Facility Location and Layout*

- *Problem Type*: klik pada *Functional Layout*,
- *Objective Criterion*: klik pada *Minimization*,
- *Problem Title*: ketik judul masalah yang ingin diselesaikan (Contoh: Modul 5),
- *Number of Functional Departments*: isi dengan jumlah departemen yang ada mulai dari *storage* sampai *warehouse*.
- *Number of Rows in Layout Area*: jumlah baris dari MCRAFT yang dibuat,
- *Number of Columns in Layout Area*: jumlah kolom dari MCRAFT yang dibuat,

iv. Klik OK

c. Masukkan data FTC.

d. Masukkan koordinat dari setiap departemen (Gambar 11).

Contoh :

(x,y) : x menandakan baris, y menandakan kolom

CS : (1,1)-(2,3),(3,1)-(4,2)

P : (1,4)-(2,5),(3,3)-(4,5)

JPR : (5,1),(5,2)-(6,5)

WB : (6,1),(7,1)-(8,5)

IR : (9,1),(9,4)

	1	2	3	4	5	
1	CS	CS	CS	P	P	1
2	CS	CS	CS	P	P	2
3	CS	CS	P	P	P	3
4	CS	CS	P	P	P	4
5	JPR	JPR	JPR	JPR	JPR	5
6	WB	JPR	JPR	JPR	JPR	6
7	WB	WB	WB	WB	WB	7
8	WB	WB	WB	WB	WB	8
9	IR	IR	IR	IR		9
	1	2	3	4	5	

Gambar 11. Contoh *input* Koordinat Setiap Departemen

e. Lakukan iterasi hingga memperoleh *layout* terbaik.

- i. Klik *Solve and Analyze* → *Solve the Problem*,
 - ii. *Solution Options*: klik pada *Improve by Exchanging 2 / 3 departments*,
 - iii. *Distance Measure*: klik *Rectilinear Distance*,
 - iv. *check* pada *Show Exchange Iteratio*,
 - v. akan keluar tampilan seperti pada Gambar 12,
 - vi. Klik kotak yang ditunjukkan oleh huruf a pada Gambar 12 untuk memperoleh *layout* dari setiap iterasi. **Save layout untuk setiap iterasi!**
 - vii. **Bandingkan total cost antara metode *exchanging 2 dan 3 departments* kemudian pilih layout final yang memiliki total cost terkecil**
- f. Lakukan *massage* untuk *layout* hasil dari software WinQSB.

Facility Location and Layout

File Iteration Scale Utilities Window Help

a

Initial Layout for run

r \ c	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	A	A	A	A	6	6	6	6	6	6	6	6	5	5	5	5	5	1	1	1
2	A			A	6							6	5			5	1			
3	A			A	6	6	6	6	6	6	6	6	5	5	5	5	1			
4	A			A	7	7	7	7	3	3	3	3	2	2	2	2	1	1	1	1
5	A			A	7			7	3			3	2				2	2	2	2
6	A			A	7			7	3			3	2							2
7	A			A	7	7	7	7	3			3	2							2
8	A		A	B	B	B	B	B	3			3	2							2
9	A		A	B				B	3	3	3	3	2	2	2	2	2	2	2	2

Gambar 12. Contoh *Layout* Awal Permasalahan

2. *Layout* GT (*Graph Based*)

- a. Untuk *layout* GT, buat *initial layout* tata letak fabrikasi untuk setiap cell yang ada dengan menggunakan MCRAFT, lalu dilanjutkan dengan metode CRAFT (sama dengan pembuatan *layout process*).
- b. Buat layout secara keseluruhan (*Storage – Cell 1 – Cell 2 – Cell 3 – Assembly – Inspeksi – Warehouse*) dengan metode *Graph Based*. Bentuk layout hasil dari poin a tidak boleh diubah.
- c. Hitung biaya *material handling* untuk *Layout* GT.

3. Perhitungan *Material handling*

- a. Lakukan pembuatan *layout* dengan skala 1:400 menggunakan *Visio*.
- b. Lakukan proses *massage* untuk seluruh departemen yang tidak berbentuk persegi/persegi panjang.
- c. Buatlah lahan kosong(gang) yang digunakan untuk *material handling*. Lebar gang sesuai referensi yang ada.

- d. Tentukan titik *centroid* untuk setiap departemen yang ada. Koordinat titik (0,0) berada di ujung kiri bawah *layout*.
- e. Tentukan pintu masuk dan keluar dari setiap departemen yang ada, pastikan pintu masuk dan keluar dapat dilalui oleh material handling yang dimiliki. Pintu masuk dan keluar memiliki akses terhadap gang.
- f. Hitung jarak secara *rectilinear* antar departemen yang memiliki aliran produksi.
- g. Lakukan perhitungan *cost Material Handling* dengan cara mengalikan jarak antar departemen dengan jumlah frekuensi perpindahan.
- h. Lakukan perhitungan waktu perpindahan untuk antar departemen yang ada.

E. Aturan dan Keterangan :

1. Ketentuan yang perlu diperhatikan pada saat melakukan *massage*:
 - a. Tidak terdapat kotak kosong atau *dummy* di dalam sebuah departemen, pengecualian untuk departemen yang berada pada sisi luar (boleh digunakan *dummy* asalkan *dummy* tersebut berada pada batas luar departemen).
 - b. Luas departemen tidak berubah (harus tetap sama setelah di-*massage*).
 - c. Bentuk 'L' kurus tidak diperkenankan.
2. Ketentuan yang perlu diperhatikan pada saat melakukan pengisian tipe metode:
 - a. Kecepatan bergerak *material handling* 5 kmph. Akselerasi dan deselerasi diabaikan.
 - b. Biaya variabel *material handling* per jam ditentukan berdasarkan gaji operator *material handling* per bulan. Perhitungan *cost* pada modul ini hanya dalam bentuk satuan waktu, sedangkan dalam satuan uang akan dilakukan pada modul 7.
3. *Storage* dan *warehouse* harus mempunyai minimal **satu sisi bebas**.
4. Untuk pembuatan laporan praktikum, gambar *process layout* hasil MCRAFT pada Microsoft Visio, dengan ukuran 1 kotak (1 x 1) cm = 16m².

F. Target Asistensi

1. *Layout* proses
2. *Layout* GT
3. Perhitungan biaya *material handling* untuk *process* dan GT *layout*

G. Daftar Pustaka

- Tompkins, J. A., White, J. A., Bozer, Y. A., dan Tanchoco, J. M. A. (2003) *Facilities Planning*, 3rd ed., Hoboken, NJ: John Wiley & Sons, Inc.
- Tompkins, J. A., White, J. A., Bozer, Y. A., dan Tanchoco, J. M. A. (2010) *Facilities Planning*, 4th ed., Hoboken, NJ: John Wiley & Sons, Inc.