BAB II
TINJAUAN PUSTAKA

Pada bagian landasan teori diuraikan berbagai teori atau referensi yang terkait dan menunjang permasalahan yang diteliti. Bab ini bertujuan untuk mendukung permasalahan yang diteliti serta mendukung hasil penelitian.

2.1 Penelitian Terdahulu

Penelitian yang telah dilakukan terdahulu tentang penjadwalan produksi dapat digunakan sebagai referensi penelitian ini. Berikut merupakan uraian penelitian terdahulu yang berkaitan dengan penjadwalan produksi:

1. Khalili *et al* (2010) membandingkan hasil penjadwalan produksi yang memiliki 2 fungsi tujuan (*bi-criteria*) untuk meminimalkan makespan dan keterlambatan (*tardiness)* pada 3 *staged flowshop problem*. Peneliti membandingkan hasil penjadwalan dengan metode *Electromagnetism-like mechanism* (EM) dengan *Simulated Annealing* (SA) dan penjadwalan heuristik meliputi SPT, NEH, (g/2,g/2) Jhonson'rule, EWDD dan SLACK. Hasil dari perbandingan menunjukkan bahwa *Electromagnetism-like mechanism* memiliki makespan dan keterlambatan paling rendah dibandingan dengan penjadwalan SA dan heuristik.
2. Pratiwi (2014) melakukan penelitian pada perusahaan penghasil kasur spons, PT. Dwisutra Setia Agung Surabaya. Pada penelitian ini mengananalisa dan memberikan penjadwalan usulan dengan metode *Integer Linear Programming*. Permasalahan pada penelitian ini, penjadwalan yang digunakan kurang optimal dan pada stage ke-5 dan ke-6 dari penelitian ini memiliki 2 mesin yang memiliki waktu proses yang berbeda, sehingga terdapat mesin yang *idle* pada selama proses produksi berlangsung. *Idle* ini menyebabkan bertambahnya waktu *makespan* dan memperlama waktu proses produksi. Hasil dari penelitian ini menunjukkan bahwa waktu *makespan* terendah diperoleh dari penjadwalan usulan dengan metode *Integer Linear Programming*.
3. Panneerselvan *et al* (2010) membandingkan hasil penjadwalan produksi pada *Unrelated Parallel Machinines n-stage m-machines* dengan fungsi tujuan meminimalkan *makespan.* Penelitian ini membandingkan penjadwalan metode heuristik dengan penjadwalan model *Integer Linear Programming*. Hasil yang diperoleh dibandingkan dengan menggunakan ANOVA. Hasil dari ANOVA

menunujkan bahwa penjadwalan dengan metode Heuristik tidak memiliki perbedaan yang signifikan dengan penjadwalan model *Integer Linear Proggramming* pada *Unrelated Parallel Machinines n-stage m-machines* meskipun hasil yang diperoleh *Integer Linear Proggramming* memiliki nilai makespan yang lebih rendah dibandingkan metode heuristik.

4. Wildan (2014), dalam penelitiannya bertujuan untuk menentukan jadwal produksi yang memiliki nilai makespan optimal. Penjadwalan dilakukan pada 2 mesin pengisian paralel identik di PT. Pertamina Production Unit Gresik. Dalam penelitian ini, dipertimbangkan masalah *sequence dependent* setup time yang dibuat dalam bentuk matriks. Metode yang digunakan adalah *Mixed Integer Linear Programming* dengan *software* LINGO 14.0 serta dihasilkan urutan pengerjaan produk yang optimal untuk masing-masing lintasan dengan nilai makespan sebesar 453,5 jam dan nilai utilitas lintasan sebesar 0,91.

Perbandingan penelitian sebelumnya dengan penelitian sekarang dapat dilihat pada Tabel 2.1.

Tabel 2.1
Penelitian Terdahulu dan Penelitian yang Sedang Dilakukan

Peneliti	Objek Penelitian	Metode	Hasil Penelitian
Khalili *et al* (2010)	3 *staged flowshop problem*	*Electromagnetism-like mechanism* (EM)	*Electromagnetism-like mechanism* memiliki makespan dan keterlambatan paling rendah dibandingan dengan penjadwalan SA dan heuristik.
Pratiwi (2014)	Produksi kasur spos di PT. Dwisutra Setia Agung Surabaya	*Integer Linear Programming (ILP)*	Waktu *makespan* terendah diperoleh dari penjadwalan usulan dengan metode *Integer Linear Programming.*
Panneerselvan *et al* (2010)	*Unrelated Parallel Machinines n-stage m-machines*	*Integer Linear Programming (MILP)*	Metode Heuristik tidak memiliki perbedaan yang signifikan dengan penjadwalan model *Integer Linear Proggramming* pada *Unrelated Parallel Machinines n-stage m-machines.*
Wildan (2014)	Mesin Pengisian paralel identik di PT. Pertamina Production	*Mixed Integer Linear Programming (MILP)*	Menjadwalkan 2 mesin paralel identik dengan *sequence dependent setup time* (dalam matriks) dan menghasilkan urutan pengerjaan produk yang optimal untuk kedua lintasan dengan nilai *makespan* terkecil.
Penelitian ini (2018)	Produksi plastik di PT. Kencana Tiara Gemilang	*Mixed Integer Linear Programming (MILP)*	Pengembangan model *Mixed Integer Linear Programming* memerlukan adanya parameter, variabel keputusan, fungsi objektif dan fungsi kendala yang diformulasikan dalam bentuk

Peneliti	Objek Penelitian	Metode	Hasil Penelitian
			matematis.Terdapat adanya 8 fungsi kendala yang digunakan dalam memodelkan penjadwalan model *Mixed Integer Linear Programming.*

2.2 Penjadwalan Produksi

Menurut Haming & Nurnajmuddin (2012:71), penjadwalan merupakan suatu proses mengorganisir, memilih dan menggunakan sumber daya yang tepat untuk mengalokasikan semua aktivitas yang penting pada sebuah prosedur sesuai dengan output yang diinginkan dan pada waktu yang diinginkan, walaupun ketepatan waktu merupakan kepuasan pelanggan dan adanya keterbatasan hubungan antara aktivitas dan sumber daya.

Menurut Ginting (2009:15), penjadwalan adalah pengurutan pembuatan atau pengerjaan produk secara menyeluruh yang dikerjakan pada beberapa buah mesin. Sedangkan menurut Baker (1974:2), penjadwalan (*scheduling*) merupakan proses pengalokasian sumber daya untuk memilih sekumpulan tugas dalam jangka waktu tertentu. Definisi ini dijabarkan dalam dua arti. Pertama, penjadwalan merupakan sebuah fungsi dalam pengambilan keputusan untuk menentukan jadwal yang paling tepat. Kedua, penjadwalan merupakan inti dari teori yang berisi kumpulan prinsip, model, teknik dan kesimpulan logis dalam pengambilan keputusan.

Secara umum tujuan penjadwalan (Baker, 1974:10) adalah:

1. Meningkatkan produktivitas mesin, yaitu mengurangi waktu menganggur.
2. Mengurangi persediaan barang setengah jadi dengan cara mengurangi jumlah rata-rata pekerjaan yang menunggu dalam antrin suatu mesin karena mesin tersebut sibuk.
3. Mengurangi beberapa keterlambatan pada pekerjaan yang mempunyai batas *completion time* (waktu penyelesaian) sehingga meminimasi *penalty cost* (biaya keterlambatan).

2.2.1 *Input* Penjadwalan Produksi

Pekerjaan-pekerjaan yang berupa alokasi kapasitas untuk *order*, penugasan prioritas *job* dan pengendalian jadwal produksi membutuhkan informasi terperinci, dimana informasi-informasi tersebut menyatakan input dari sistem penjadwalan (Kusuma, 2001:18).

Pada penjadwalan produksi, kebutuhan-kebutuhan kapasitas dari semua *order* yang dijadwalkan dalam hal macam dan jumlah keseluruhan sumber daya yang digunakan.

Untuk produk tertentu, informasi ini bisa diperoleh dari lembar kerja operasi (berisi kebutuhan-kebutuhan komponen, sub komponen dan bahan pendukung). Kualitas dari keputusan penjadwalan sangat dipengaruhi oleh ketepatan estimasi *input*. Oleh karena itu, pemeliharaan catatan terbaru tentang status tenaga kerja dan peralatan yang tersedia dan perubahan kebutuhan kapasitas yang diakibatkan perubahan desain produk/ proses menjadi sangat penting (Haming & Nurnajmuddin, 2012:72).

2.2.2 *Output* Penjadwalan Produksi

Untuk memastikan bahwa suatu aliran kerja yang lancar melalui tahapan produksi, maka sistem penjadwalan harus membentuk aktivitas-aktivitas *output* sebagai berikut:

1. Pembebanan *(loading)*

   Pembebanan melibatkan penyesuaian kebutuhan kapasitas untuk *order* yang diterima/ diperkirakan dengan kapasitas yang tersedia. Pemebebanan dilakukan dengan menugaskan *order* pada fasilitas-fasilitas, operator-operator dan peralatan tertentu.
2. Pengurutan (*sequencing*)

   Pengurutan ini merupakan penugasan tentang *order* mana yang diprioritaskan untuk diproses terlebih dahulu bila suatu fasilitas harus memproses banyak *job*.
3. Prioritas *Job* (*dispatching*)

   Dispatching merupakan prioritas kerja tentang *job* mana yang diseleksi dan diprioritaskan untuk diproses.
4. Pengendalian Kinerja Penjadwalan

   Pengendalian kinerja penjadwalan dapat dilakukan dengan:
   a. Meninjau kembali status semua *order* pada saat tertentu melalui sistem tertentu.
   b. Mengatur kembali urutan-urutan, misalnya: *expediting* semua *order* yang jauh dibelakang atau mempunyai prioritas utama.
   c. *Up-Dating* jadwal, sebagai refleksi kondisi operasi yang terjadi dengan merevisi prioritas-prioritas.

2.2.3 Notasi dan Istilah Penjadwalan Produksi

Menurut Pinedo (2008:22), istilah-istilah yang digunakan dalam penjadwalan produksi adalah sebagai berikut:

1. Setiap *job* i {=1,2,3,….,n} yang dijadwalkan pada mesin j {1,2,3,..,m}. proses pengerjaan *job* i di mesin j disebut dengan O_{ij}.

2. *Processing time* (waktu proses), P_{ij} yaitu lamanya waktu yang harus dihabiskan *job* i di mesin j untuk memproses operasi O_{ij}.
3. *D*ue *date* (waktu tenggat), d_{ij} yaitu batas *completion time job* i yang telah ditentukan. Apabila *completion job* diluar waktu ini, maka dikenakan pinalti pada *job* tersebut.
4. *Release date* (waktu lepas), r_i adalah waktu ketika *job* i masuk ke sistem, yaitu waktu paling awal job i bisa mulai diproses. Biasanya $r_i = 0$.
5. *Start tim* (waktu mulai), s_{ij} adalah waktu mulai diprosesnya *job* i pada mesin j.
6. *Completion time* (waktu penyelesaian), C_{ij} adalah waktu penyelesaian pemrosesan *job* i pada mesin j.
7. *Flow time* (alir waktu), $F_{ij} = C_{ij} - r_i$ adalah rentang waktu antara tugas tersedia untuk diproses dengan ketika tugas tersebut selesai.
8. Makespan C_{max} yaitu waktu penyelesaian dari seluruh *job.*
9. *Lateness* (waktu keterlambatan), $L_i = C_i - d_i$ adalah selisish antara waktu penyelesaian j*ob* i dengan waktu tenggatnya. *Lateness* baru dapat dihitung setelah *job* i selesai menjalani semua proses dan dapat bernilai negatif dan positif.
10. *Tardiness* (waktu keterlambatan positif), $T_i = max\ (L, 0)$, adalah besarnya keterlambatan penyelesaian *job* i.
11. *Earliness* (waktu keterlamabatan negatif), $T_i = min\ (L, 0)$, adalah besarnya keterlambatan penyelesaian *job* i.

2.2.4 Kriteria Evaluasi Penjadwalan

Keberhasilan suatu penjadwalan dapat diukur dengan besaran yang melibatkan informasi dari pekerjaan-pekerjaan yang dijadwalkan. Hal tersebut dapat dicapai dengan menentukan kriteria jadwal yang diinginkan. Menurut Conway (1967:47), penentuan kriteria jadwal dapat didasarkan atas karakteristik pekerjaan dan peralatannya sebagai berikut:

1. Kriteria berdasarkan atribut tugas. Dilihat dari hubungannya dengan tugas yang ada, misalnya minimasi *flow time*, minimasi *lateness*, minimasi jumlah *job* yang terlambat.
2. Kriteria berdasarkan atribut *shop*. Dilihat dalam hubungannya dengan *shop* (alat atau mesin), seperti: maksimal utilitas *shop*, minimasi waktu *set up* mesin, penyeimbang beban mesin.

Jika terdapat n pekerjaan yang dijadwalkan, maka tingkat keberhasilan dapat dinilai dari besaran-besaran berikut (Baker, 1974:12):

1. *Makespan* (waktu penyelesaian seluruh pekerjaan)

   $C_{max} = Max\{Ci\}$ (2-1)

2. *Mean Flow Time* (waktu penyelesaian rata-rata)

   $\bar{F} = \frac{1}{n}\sum_{i=1}^{n} F_i$ (2-2)

3. *Mean Lateness* (rata-rata keterlambatan)

   $\bar{L} = \frac{1}{n}\sum_{i=1}^{n} L$ (2-3)

4. *Mean Tardiness* (rata-rata keterlambatan non negatif)

   $\bar{T} = \frac{1}{n}\sum_{i=1}^{n} T$ (2-4)

5. *Maximum Flow Time* (maksimal waktu penyelesaian seluruh pekerjaan)

   $F_{max} = \max_{1\le i\le n}\{F_i\}$ (2-5)

6. *Maximum Lateness* (maksimum keterlambatan)

   $L_{max} = \max_{1\le i\le n}\{L_i\}$ (2-6)

7. *Minimum Tardiness* (maksimum keterlambatan non negatif)

   $L_{min} = \max_{1\le i\le n}\{T_i\}$ (2-7)

8. *Number of Tardy Jobs* (jumlah pekerjaan yang terlambat)

   $N = \sum_{i=1}^{n} \delta_i$ (2-8)

2.3 Jenis Penjadwalan Produksi

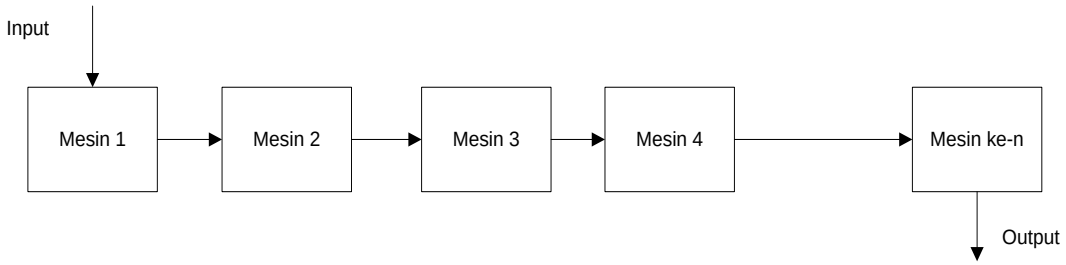
Penjadwalan pada umumnya dibagi menjadi beberapa macam sesuai dengan karakteristiknya. Penjadwalan dibagi berdasarkan mesin yang digunakan dalam proses dan berdasarkan aliran prosesnya.

1. Berdasarkan mesin yan digunakan dalam proses
   a. Penjadwalan pada mesin tunggal (*single machine*)
   b. Penjadwalan pada mesin jamak (*m machine*)
2. Berdasarkan pola aliran proses

a. Penjadwalan *Flowshop*

Proses produksi dengan aliran *flowshop* berarti proses produksi dengan aliran identik dari satu mesin ke mesin yang lain atau dengan kata lain semua produk diproses melalui aliran dan mesin yang sama. Susunan suatu proses produksi jenis *flowshop* dapat diterapkan dengan tepat untuk produk-produk dengan desain yang stabil dan berproduksi dalam jumlah besar, sehingga investasi dengan tujuan khusus yang digunakan dapat segera kembali.

Menurut Baker (1974:137), pada tipe *Flowshop* terbagi lagi menjadi *pure flowshop* dan *general flowshop*. Pada *pure flowshop* semua pekerjaan mengalir pada lini produksi yang sama dan tidak adanya variasi dalam proses produksinya. Sedangkan untuk *general flowshop* terdapat variasi dalam proses produksinya, sehingga dimungkinkan terdapat pekerjaan yang tidak harus di kerjakan di semua mesin.

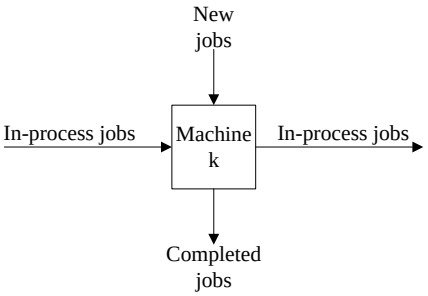


Gambar 2.1 Skema *flow shop*
Sumber: Baker (1974:137)

Gambar 2.1 adalah contoh aliran *job* pada tipe penjadwalan *flowshop* mulai dari mesin 1 hingga ke mesin n.

b. Penjadwalan *Job Shop*

Proses produksi dengan aliran *job shop* berarti proses produksi dengan pola aliran atau rute pada tiap mesin yang spesifik untuk setiap pekerjaan dan mungkin berbeda untuk tiap *job*. Gambar 2.2 merupakan contoh aliran *job* pada tipe penjadwalan *job shop*.



Gambar 2.2 Skema *job shop*
Sumber: Baker (1974:178)

c. Penjadwalan *Flexible Flowshop*

Dimana tiap *job* mengambil rute yang sama melewati aliran proses seperti *flow shop*, tetapi dalam prosesnya terdapat jumlah mesin yang lebih dari satu tipe dan hanya digunakan dalam satu *job* saja. Keuntungan dalam menggunakan aliran proses ini adalah semakin singkatnya pekerjaan, karena menggunakan mesin lebih dari satu.

d. *Re-entrant Flowshop*

Dimana tiap job dapat mengunjungi kembali mesin sebelumnya dan dapat dilakukan berkali-kali.

2.4 Metode Penjadwalan Produksi

Masalah penjadwalan produksi dapat diselesaikan dengan menggunakan metode heuristik yang terdiri dari 2 jenis, yaitu metode heuristik dan metode meta-heuristik. Algoritma heuristik menyususn satu per satu solusi dari masalah penjadwalan. Alagoritma ini memilih mesin-mesin atau *job-job* atau operasi-operasi mana yang harus dijadwalkan terlebih dahulu. Algoritma heuristik yang sering digunakan untuk menyelesaikan penjadwalan yaitu *priority dispatching rule*.

Priority dispatching rule adalah suatu aturan penjadwalan yang mengatur *job* mana pada suatu antrian job pada suatu mesin yang harus diproses terlebih dahulu berdasarkan prioritas-priorotas tertentu. Jadi, ketika suatu mesin telah selesai melakukan proses pada suatu *job* maka berdasarkan *priority dispatching rule*, dipilih satu *job* yang memiliki prioritas tertinggi untuk diproses di mesin tersebut (Pinedo, 2012:127).

Berikut adalah beberapa aturan yang merupakan *basic priority dispatch rules* (Pinedo, 2012:129):

1. *The Service in Random Order* (SIRO) *rule*

   Menurut aturan prioritas ini, setiap kali mesin tidak melakukan pekerjaan, pekerjaan berikutnya dipilih secara acak dari *job* dalam antrian untuk diproses. Aturan ini dilakukan tidak untuk mengoptimalkan apapun.

2. *The Earliest Release Date first* (ERD) *rule*

   Aturan ini terkenal juga dengan aturan *First-Come_First-Served* yang berdasarkan waktu kedatangan *job* atau pesanan dari pelanggan. Sehingga *job* yang pertama kali dating, dikerjakan terlebih dahulu dan begitu seterusnya. Aturan ERD digunakan untuk meminimalkan variasi lama waktu tunggu dari pekerjaan pada mesin.

3. *The Earliest Due Date first* (EDD) *rule*

   Menurut aturan ini, urutan penjadwalan dilakukan berdasarkan *due date* setiap *job*. Aturan ini mengabaikan waktu proses dari setiap *job* dan waktu kedatangan *job*. Sehingga *job* yang memiliki waktu *due date* paling awal memiliki prioritas yang paling tinggi untuk diproses di sebuah mesin. Aturan ini biasanya digunakan untuk meminimumkan keterlambatan pada *job.*

4. *The Minimum Slack Time first* (MST) *rule*

   Menurut aturan ini, *job* diurutkan berdasarkan waktu *slack* yang paling kecil. Pada saat mesin telah selesai memproses suatu *job*, maka dihitung waktu *slack* yang tersisa ($d_i - p_i - t, 0$) dari tiap-tiap *job* yang ada dalam antrian, dimana t merupakan waktu sekarang. *Job* yang memiliki waktu *slack* terkecil memiliki prioritas tertinggi untuk

diproses pada mesin selanjutnya. Aturan ini berkaitan dengan *due date* sehingga dapat digunakan untuk meminimalkan keterlambatan *job.*

5. *The Weighted Shortest Processing Time first* (WSPT) *rule*

   Shortest Processing Time (SPT) *rule* adalah aturan yang mengurutkan *job* berdasarkan pada lamanya waktu proses setiap *job.* Jadi *job* yang memiliki waktu proses paling singkat diproses terlebih dahulu dan kemudian dilanjutkan oleh *job* lainnya sampai pada *job* yang memiliki waktu proses paling lama. Aturan ini digunakan untuk menyeimbangkan beban kerja antar mesin yang disusun secara paralel. Sedangkan *The Weighted Shortest Processing Time fisrt* (WSPT) *rule* merupakan aturan pengurutan yang setiap kali mesin tidak melakukan pekerjaan dengan rasio tertinggi bobot (w_j) dari waktu pemrosesan (p_j) maka dijadwalkan berikutnya, yaitu pekerjaan dalam urutan w_j/p_j. Aturan ini cenderung untuk meminimalkan jumlah dari waktu penyelesaian terbobot, yaitu w_jC_j. dalam pengaturan mesin tunggal, dengan n pekerjaan yang tersedia pada waktu nol, aturan WSPT tidak meminimalkan w_jC_j. ketika semua bobot adalah sama, aturan WSPT dapat menurun ke aturan SPT.

6. *The Longest Processing Time first* (LPT) *rule*

   Aturan ini memerintahkan pekerjaan dalam urutan penurunan pengolahan waktu. Ketika ada mesin paralel, aturan ini cederung untuk menyeimbangkan beban kerja atas mesin. Hal ini menguntungkan untuk mempertahankan pekerjaan dengan waktu pemrosesan singkat untuk nanti, karena pekerjaan ini berguna pada akhir untuk menyeimbangkan beban kerja. Setelah penugasan pekerjaan ke mesin telah ditentukan, pekerjaan pada setiap mesin tertentu dapat *resequenced* tanpa mempengaruhi keseimbangan beban kerja.

7. *The Shortest Setup Time first* (SST) *rule*

   Aturan yang mempriorotaskan operasi dengan waktu proses operasi paling sedikit.

8. *The Least Flexible Job first* (LFJ) *rule*

   Aturan ini digunakan ketika ada sejumlah mesin non-identik secara paralel dan pekerjaan tunduk pada kendala kelayakan mesin, dimana *job* yang memiliki fleksibilitas pengerjaan yang paling tinggi, dalam artian dapat dikerjakan di jenis mesin manapun, diprioritaskan terlebih dahulu.

9. *The Critical Path* (CP*) rule*

   Aturan Jalur kritis digunakan dengan operasi yang ada batasan didahulukan. Aturan ini memilih sebagai pekerjaan berikutnya.

10. *The Largest Number of Successors (*LNS*) rule*

    Aturan yang memprioritaskan operasi dengan jumlah successor operasi paling banyak.

11. *The Shortest Queue at Operation (*SQNO*) rule*

    Aturan yang digunakan di *job shop*. Aturan ini memprioritaskan jumlah antrian terkecil untuk diproses terlebih dahulu.

12. *The Priority Scheduling (*PS*) rule*

    Aturan yang digunakan yang memprioritaskan operasi dengan jumlah pembobotan tertentu. Pembobotan yang terbesar yang dikerjakan terlebih dahulu.

2.5 *Linear Programming*

Linear Programming merupakan salah satu alat riset operasi yang paling efektif untuk digunakan (Taha, 2007:11). *Linear programming* adalah metode matematis yang mengalokasikan sejumlah sumber daya yang terbatas untuk menghasilkan tujuan tunggal yaitu memaksimumkan keuntungan atau meminimumkan biaya. Sumber daya dapat berupa uang, tenaga kerja, bahan mentah, kapasitas mesin, waktu, ruangan atau teknologi. Pokok pikiran yang utama dalam menggunakan *linear programming* adalah merumuskan masalah dengan jelas dari sejumlah informasi yang tersedia, lalu menerjemahkan masalah tersebut menjadi bentuk model matematis untuk mendapatkan solusi terhadap masalah yang dihadapi. Langkah-langkah dalam pembuatan linear programming adalah sebagai berikut (Aminudin, 2005:6-8):

1. Mendefinisikan masalah

   Pendefinisian dan penggambaran masalah harus dibuat dengan jelas. Pendefinisian masalah merupakan langkah yang penting sehingga sebaiknya melibatkan manajemen maupun anggota organisasi lainnya.

2. Memformulasikan modem matematis

   Model adalah gambaran abstrak dari suatu masalah. Ketepatan dalam memformulasikan sangat dipengaruhi oleh asumsi yang digunakan. Asumsi model harus realistis. Asumsi model yang realistis menjadi faktor kesulitan dalam membuat model. Setelah pendefisinian masalah, langkah selanjutnya adalah membuat formulasi model matematis yang terdiri dari tiga tahap yaitu:

   a. Menentukan variabel yang tak diketahui (variabel keputusan) dan nyatakan dalam simbol matematis.
   b. Membuat fungsi tujuan yang ditunjukkan sebagai suatu hubungan linier (bukan perkalian) dari variabel keputusan.

c. Menentukan kendala-kendala masalah tersebut dan mengekspresikan dalam persamaan atau pertidaksamaan yang juga merupakan hubungan linier dari variabel keputusan. Kendala mencerminkan keterbatasan sumber daya masalah tersebut.

3. Mengukur validitas

   Untuk dapat memastikan apakah terdapat korelasi antara model matematis dengan kejadian sebenarnya diperlukan pengujian-pengujian terhadap model.

4. Implementasi keputusaan

   Langkah terakhir dalam suatu kajian *linear programming* adalah pelaksanaan atau implementasi dari solusi yang dihasilkan dengan disetujui oleh pembuat keputusan.

Permasalahan *linear programming* pada dasarnya memiliki lima karakteristik utama yaitu:

1. Masalah *linear programming* pada umumnya berupaya untuk memaksimumkan keuntungan atau meminimumkan biaya. Upaya maksimasi keuntungan atau minimasi biaya disebut fungsi tujuan (*objective function*) dari *linear programming*. Fungsi tujuan ini terdiri dari variabel-variabel keputusan (*decision variables*).
2. Terdapat keterbatasan atau kendala-kendala untuk mencapai tujuan yang ditentukan dalam *linear programming*. Kendala-kendala tersebut dirumuskan dalam fungsi-fungsi kendala (*constrains function*) yang terdiri dari variabel-variabel keputusan yang menggunakan sejumlah sumber daya yang terbatas. Jadi, *linear programming* akan menghasilkan fungsi tujuan dengan memperhatikan fungsi-fungsi kendala yang ada.
3. Memiliki sifat linearitas. Sifat linearitas ini berlaku untuk semua fungsi tujuan dan fungsi-fungsi kendala. Sebagai contoh, apabila satu unit produk A dapat menghasilkan keuntungan, katakanlah Rp200.000 maka apabila kita memproduksi dua unit produk A memberikan keuntungan Rp400.000 (2 x Rp200.000), produksi tiga unit A menghasilkan keuntungan Rp600.000 (3 x Rp200.000) dan seterusnya. Demikian pula untuk penggunaan sumber-sumber daya. Misalnya untuk sumber daya tenaga kerja, katakanlah untuk memproduksi satu unit produk A membutuhkan tiga jam kerja, maka untuk menghasilakan dua unit produk A membutuhkan enam jam kerja (2 x 3 jam kerja) sedangkan produksi 3 unit produk A membutuhkan sembilan jam kerja (3 x 3 jam kerja).
4. Memiliki sifat homogenitas. Sifat homogenitas ini berkaitan dengan kehomogenan sejumlah sumber daya yang digunakan. Misalnya dalam proses produksi, semua

produk X dihasilkan oleh mesin-mesin yang identik, tenaga kerja yang memilki kemampuan sama.

5. Memiliki sifat *divisibility*. Sifat *divisibility* dibutuhkan karena *linear programming* mengasumsikan bahwa nilai dari variabel-variabel keputusan maupun penggunaan sumber daya dapat dibagi ke dalam pecahan-pecahan. Jika pembagian ini tidak mungkin dilakukan terhadap variabel keputusan, sebagai contoh dalam industri mobil dan furniture nilai kuantitas produksi diukur dalam bilangan bulat, maka modifikasi terhadap *linear programming* harus dilakukan. Bentuk modifikasi dari linear programming ini disebut *integer linear programming*.

Berikut ini salah satu contoh pemodelan *integer linear programming* yang digunakan pada kasus penjadwalan dengan n-kriteria non-permutasi (Komaki *et al*, 2015:353):

Indeks

J = *Job* j dimana j= 1,..,N

i = *Stage* i dimana i=1,...,M

Parameter

$t_{i,j}$ = waktu proses *job* j pada *stage* i

$s_{i,k,j}$ = waktu *setup* dari *job* k menuju *job* j pada *stage* i

$r_{i,}j$ = waktu rute dari *job* j menuju *stage* i. $r_{i,j}$ bernilai 1 jika *job* j dikerjakan pada *stage* i, selain itu bernilai 0

U = nilai berukuran besar

g_j = tahapan *job* yang diinginkan pada *stage* terakhir

w_1, w_2, w3 = pembobotan pada fungsi tujuan f_1, f_2 dan f_3

d_j = *due* date untuk *job* j

Variabel Keputusan

$C_{i,j}$ = waktu *completion* time pada *job* j di *stage* i

P*i,k,j* = pendahulu *job* j menuju *job* k di stage i. Pi,k,j bernilai 1 jika *job* k harus dikerjakan terlebih dahulu sebelum *job* j, selain itu bernilai 0

o_j = tahapan *job* sekarang (tanpa mempertimbangkan prioritas) pada *stage* terakhir

Fungsi Tujuan

$$\text{Min } Z = w_1 \, x \, f_1' + w_2 \, x \, f_2' + w_3 \, x \, f_3' \quad (2\text{-}9)$$

Fungsi Kendala

$$w_1 + w_2 + w_3 = 1 \quad (2\text{-}10)$$

$$f_1 = \text{Max}\{Cj\} \quad (2\text{-}11)$$

$$f_2 = \frac{1}{n}\sum_{i=1}^{n} T \quad (2\text{-}12)$$

$$f_3 = \sum_{j=1}^{n} |o_j - g_j| \quad (2\text{-}13)$$

$$f_1^{'} = (f_1 - f_{1,min}) / (f_{1,max} - f_{1,min}) \quad (2\text{-}14)$$

$$f_2^{'} = (f_2 - f_{2,min}) / (f_{2,max} - f_{2,min}) \quad (2\text{-}15)$$

$$f_3^{'} = (f_3 - f_{3,min}) / (f_{3,max} - f_{3,min}) \quad (2\text{-}16)$$

$$C_{i,j} \geq r_{i,j} \text{ x } t_{i,j} \quad (2\text{-}17)$$

$$C_{i+1,j} \geq C_{i,j} + r_{i,j} \text{ x } t_{i,j} \quad (2\text{-}18)$$

$$r_{i,j} \text{ x } (C_{i,j} - C_{i,k}) + U \text{ x } P_{i,j} \geq r_{i,j} \text{ x } (t_{i,j} + s_{i,k,j}) \quad (2\text{-}19)$$

$$r_{i,j} \text{ x } (C_{i,k} - C_{i,j}) + U \text{ x } (1 - P_{i,j}) \geq r_{i,j} \text{ x } (t_{i,j} + s_{i,k,j}) \quad (2\text{-}20)$$

$$f_1 \geq C_{M,j} \quad (2\text{-}21)$$

Persamaan (2-9) digunakan untuk memperoleh nilai optimal yang diperoleh dari waktu *makespan*, *tardiness* dan perbedaan dengan jadwal sebelum. Persamaan (2-10) menjelaskan total keseluruhan pembobotan bernilai 1. Persamaan (2-11) menghitung total waktu *makespan.* Persamaan (2-12) menghitung nilai rata-rata dari *tardiness.* Persamaan (2-13) menghitung nilai perbedaan penjadwalan *existing* dan usulan. Persamaan (2-14), (2-15) dan (2-16) menujukan nilai kemungkinan minimum dan maksimun dari fungsi tujuan. Persamaan (2-17) memastikan dan menghitung nilai penyelesain suatu *job* j pada *stage* i. Persamaan (2-18) menghitung penambahan waktu penyelesaian pada *stage* sebelumnya. Persamaan (2-19) dan (2-20) memastikan bahwa *job* dikerjakan pada setiap *stage.* Persamaan (2-21) memastikan bahwa nilai *maksepan* yang diperoleh adalah nilai *makespan* keseluruhan.

2.6 *Gantt Chart*

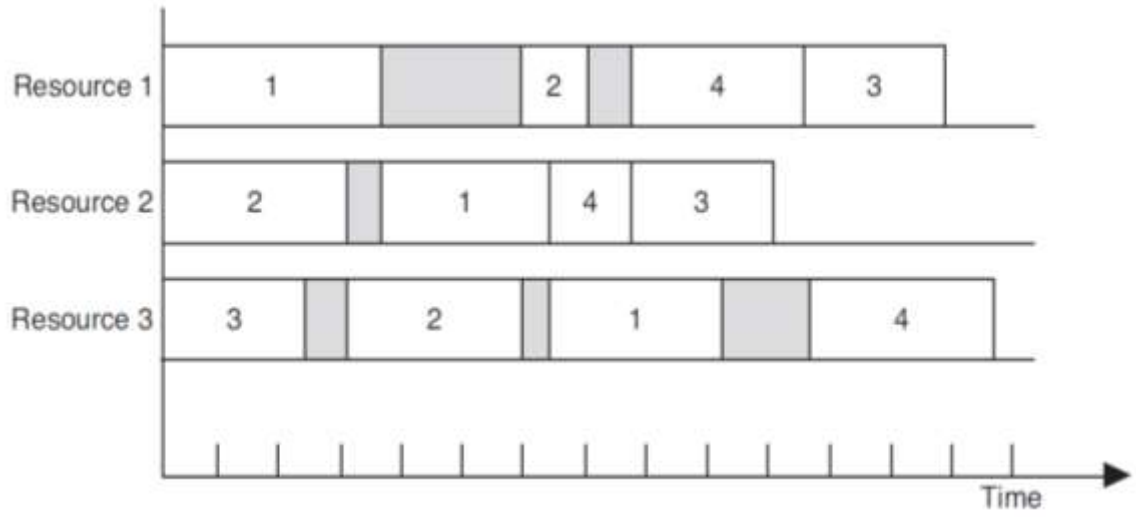
Gantt chart merupakan grafik hubungan antara alokasi sumber daya dengan waktu. *Gantt chart* digunakan untuk berbagai tujuan yang berhubungan dengan pembebanan dan penjadwalan. Tujuan dari *Gant chart* adalah untuk mengorganisasi dan secara visual menampilkan penggunaan sumber daya aktual atau yang diinginkan dalam sebuah kerangka kerja waktu. *Gantt chart* terdiri dari sumbu X dan sumbu Y. Sumbu X merupakan durasi pengerjaan tugas yang dapat dinyatakan dalam jam, hari, minggu, bulan atau lainnya sesuai dengan kebutuhan (Nisa, 2008:94). Sedangkan sumbu Y merupakan tugas atau sumber daya yang digunakan dalam kegiatan tersebut.

Menurut Ginting (2009:93), dari *gantt chart* kemudian ditentukan urutan *(sequence)* dari *job* yang memberikan kriteria penjadwalan terbaik, misalnya waktu pemrosesan tersingkat, utilitas mesin dan *idle time* minimum.

Menurut Pinedo (2009:360), ada beberapa model/ *interface* yang tersedia untuk mempermudah visualisasi dari penjadwalan secara detail antara lain:

1. *Gantt chart*: menunjukkan kapasitas yang tersedia dalam jangka pendek (hari atau minggu) ketika ada sejumlah pekerjaan/ *job* (dua puluh atau tiga puluh).
2. Dispatch *list*: menunjukkan daftar pekerjaan di urutan mana pekerjaan tersebut diproses pada suatu mesin, namun interface ini tidak memberikan pandangan yang baik dari jadwal yang relatif terhadap waktu.
3. *Capacity bucket*: menunjukkan angka/ numerik pada setiap segmen waktu, utilitas setiap mesin dan baik untuk perencanaan jangka medium atau panjang.
4. *Throughput diagram*: menunjukkan secara kumulatif dari waktu ke waktu mengenai jumlah total pesanan yang diterima, total yang dihasilkan dan total yang dikirimkan.

Model yang paling sederhana dan paling banyak dipakai adalah *gantt chart*. *Gantt chart* dipopulerkan penggunaannya oleh Henry Gantt pada tahun 1990-an. Pada umumnya, *gantt chart* terdiri dari garis vertikal dengan sumbu y dan garis horisontal dengan sumbu x. Seperti pada Gambar 2.3, sumbu y mewakili sumber daya tertentu, sedangkan sumbu x menunjukkan skala waktu. *Gantt chart* tersebut juga menunjukkan kapan *job* mulai hingga selesai dan juga menunjukkan perkiraan waktu *idle* untuk setiap sumber daya.



Gambar 2.3 Gantt chart
Sumber: Baker (1974:3)

Dengan *gantt chart*, lebih mudah menemukan informasi tentang jadwal yang diberikan dengan menganalisis hubungan geometris. Selain itu, juga dapat mengatur ulang tugas atau

job pada grafik untuk memperoleh informasi komparatif tentang jadwal alternatif. Dengan cara ini, *gantt chart* berfungsi sebagai bantuan untuk mengukur kinerja dan membandingkan jadwal serta untuk memvisualisasikan masalah sejak awal.

Halaman ini sengaja dikosongkan