BAB II
TINJAUAN PUSTAKA

Pada bab ini diuraikan mengenai landasan teori dan acuan yang digunakan dalam menyelesaikan permasalahan penelitian. Tinjauan pustaka yang digunakan sebagai pedoman agar pelaksanaan penelitian dapat terfokus pada tujuan yang ingin dihasilkan dan bersumber dari buku, jurnal ilmiah, internet, penelitian serta sumber lainnya yang relevan. Dalam bab ini juga dijelaskan mengenai tinjauan pustaka dari penelitian terdahulu yang mendukung penelitian ini sehingga akan didapat perbandingan dan acuan garis besar sebelum dilaksanakannya penelitian.

2.1 Penelitian Terdahulu

Beberapa penelitian sebelumnya yang berkaitan dengan penjadwalan dengan menggunakan metode Algoritma *Tabu Search* akan dijelaskan sebagai berikut :

1. Noversada, Meiriza dan Wijayanti (2006) melakukan penelitian yang bertujuan untuk membuktikan dan memberikan penjelasan mengapa algoritma *Tabu Search* sangat cocok untuk diterapkan pada lingkungan *job shop* dan memberikan hasil yang paling optimal dibandingkan dengan metode-metode yang lain. Hal ini disebabkan oleh fleksibilitasnya yang bisa menyaingi prosedur klasik lainnya. Saat ini banyak sekali permasalahan dari berbagai bidang yang bisa menggunakan implementasi algoritma *Tabu Search* untuk mendapatkan solusi optimal. Salah satu contoh permasalahan yang juga dapat diselesaikan dengan algoritma *Tabu Search* adalah permasalahan *job shop scheduling* dimana permasalahan ini bertujuan untuk meminimasi waktu *makespan*. Hasil ini didukung oleh beberapa ahli riset bahwa ada beberapa kondisi dimana algoritma ini dapat bekerja dengan lebih baik lagi. Caranya adalah dengan melakukan *Aspiration Criterion, Resizing The Tabu List, dan Restoring The Best Known* dalam langkah iterasinya.
2. PT. Indonesian Steel Tube Work (ISTW) merupakan perusahaan penanaman modal asing (PMA) Jepang yang bergerak di bidang pembuatan pipa baja las dengan sistem produksinya yang berupa *make to order*. Permasalahan yang dihadapi oleh perusahaan adalah terdapat banyak keterlambatan dalam pemenuhan *due date* pesanan terutama bila terjadi *rush order* atau *order sisipan*. Sebelumnya, kebijakan penjadwalan produksi perusahaan lebih berpatokan pada pengelompokan diameter pipa yang dipesan oleh konsumen. Penjadwalan ini kelihatannya akan memudahkan

pengaturan waktu *set-up* mesinnya, namun pada kenyatannya akan menghasilkan banyak keterlambatan. Oleh karena itu, penelitian ini mengusulkan penjadwalan ulang dengan menggunakan gabungan metode *Heuristic Interchange* dan metode Wilkerson-Irwin. Hasil yang didapat dari penelitian ini adalah perbaikan performasi jadwal ditandai dengan penurunan total *tardiness* yang tadinya berjumlah 1806 hari berkurang menjadi 930 hari, *mean tardiness* juga berkurang dari 9 hari menjadi 5 hari, *max. tardiness* yang semula 60 hari berkurang menjadi 41 hari saja, demikian yang terakhir untuk jumlah *job* yang terlambat juga berkurang dari 66 *job* menjadi 42 *job* saja. (Nurkertamanda, Fanani, dan Wardhani: 2009).

3. Rudyanto dan Arifin (2010) melakukan penelitian di *Home Industry* CV. Eko Joyo yang memproduksi paving *block*. Sebelumnya, perusahaan melakukan penjadwalan produksi secara konvensional, sehingga hal ini dapat menyebabkan banyak keterlambatan pada lantai produksi dan *order* selesai melewati target waktu *due date*-nya. Metode yang diusulkan dalam penelitian ini adalah metode *Earliest Due Date* (EDD) karena merupakan metode penjadwalan produksi yang menghasilkan *max. tardiness* yang paling minimum. Usulan ini terbukti menghasilkan *tardiness* maksimum yang lebih sedikit dibandingkan penjadwalan yang dilakukan secara konvensional (tanpa sistem informasi penjadwalan produksi dengan metode EDD). Hasil perbandingan dari kedua metode ini didukung dengan perhitungan menggunakan uji hipotesis *paired comparation (t-test)* yang mengindikasikan bahwa terdapat perbedaan yang signifikan dari nilai rata-rata *maximum tardiness* antara penjadwalan metode konvensional dan metode EDD.

Dari beberapa tinjauan pustaka dari referensi penelitian terdahulu maka pada Tabel 2.1 ditampilkan ringkasan perbandingan dari masing-masing penelitian.

Tabel 2.1 Perbandingan Penelitian Terdahulu dengan Penelitian Sekarang

	Noversada, dkk	**Nurkertamanda., dkk**	**Rudyanto & Arifin**	**Penelitian Sekarang**
Judul Penelitian	Algoritma Tabu *Search* dan Penggunaannya dalam Penyelesaian *Job Shop Schedulling Problem*	Penerapan Kebijakan Ulang pada Ruang Lingkup Single Machine untuk Meminimasi Total *Tardiness*	Penerapan Metode Earliest Due Date pada Pejadwalan Produksi Paving pada CV. Eko Joyo	Perbaikan Prioritas Penjadwalan Pada Lingkungan Produksi Fleksibel *Job Shop* (FJSP) untuk Meminimasi Total *Tardiness*.

Tabel 2.1 Perbandingan Penelitian Terdahulu dengan Penelitian Sekarang (Lanjutan)

	Noversada, dkk	**Nurkertamanda., dkk**	**Rudyanto & Arifin**	**Penelitian Sekarang**
Metode	Algoritma *Tabu Search*	*Heuristic Interchange* Nyrendra dan Wilkerson Irwin	EDD, *paired t-test*	EDD, FCFS, *Pairwise Interchange Heuristic*, dan *Tabu Search*
Objek Penelitian	Lingkungan *Job-Shop*	PT. Indonesia Steel Tube Works, *Job-shop*	CV. Eko Joyo, *job-shop*	PT. Barata Indonesia (Persero), FJSP
Tujuan Penelitian	Minimasi *Makespan*	Meminimasi total *tardiness*, Mean *tardiness*, max *tardiness* dan *number of tardy jobs*.	Minimasi *maximum tardiness*	Meminimasi total *tardiness* dan biaya penalti dari *job-job* yang ada.

2.2 Tinjauan Mengenai Konsep Penjadwalan Produksi

Baker (1974) mengatakan bahwa di dalam penjadwalan terdapat beberapa batasan yang mungkin terjadi sesuai dengan kondisi nyatanya, yaitu:

1. Adanya keterbatasan kapasitas pada masing-masing *resources* yang tersedia
2. Adanya pembatasan penggunaan teknologi untuk masing-masing order yang ada sehingga tugas dapat dikerjakan sesuai dengan teknologi yang dimiliki.

Sehingga dalam menyelesaikan masalah penjadwalan, *Baker* membagi permasalahan menjadi dua garis besar yaitu dengan menggunakan *resources* yang manakah tugas dapat dikerjakan dan kapankah *job* dapat dikerjakan. Dengan kata lain, esensial dari permasalahan penjadwalan adalah memberikan solusi untuk keputusan pengalokasian dan pengurutan *job*. *Scheduling* (penjadwalan) merupakan proses penugasan kapan pekerjaan harus dimulai dan diselesaikan, sedangkan *sequencing* (pengurutan) merupakan proses pengaturan urutan atas pekerjaan-pekerjaan yang harus diselesaikan tersebut. Karena eratnya hubungan diantara kedua istilah ini, maka biasanya dalam penggunaan kata *scheduling* (penjadwalan), pengertian *sequencing* sudah tercakup didalamnya.

2.3 Definisi dan Tujuan Penjadwalan

Secara umum masalah penjadwalan dapat dijelaskan sebagai berikut. Jika ada n *job* $\{j_1, j_2, j_3, ..., j_n\}$ harus diproses pada m mesin $\{m_1, m_2, m_3, ..., m_n\}$ Proses pengerjaan *job* J_1 disebut dengan operasi O_{ij}. Waktu yang diperlukan untuk memproses operasi O_{ij}

pada mesin M_j adalah t_{ij}. Beberapa *Job* mungkin memiliki saat pengerjaan paling awal atau saat kedatangan *job* ke *Shop* yang disebut *release date*, r_{ij} yang mungkin tidak sama dengan 0, dan juga batas saat penyelesaian yang disebut *due date*, d_{ij}. Permasalahan penjadwalan adalah menentukan urutan produksi yang memberikan solusi terbaik dengan kriteria sebagai berikut:

1. Memenuhi *technological constraint* yang ada, dengan kata lain merupakan jadwal yang *feasible*.
2. Penentuan mesin yang akan digunakan (pengalokasian mesin) untuk menyelesaikan suatu proses produksi.
3. Penentuan waktu pemakaian mesin tersebut (pengurutan).
4. Memenuhi beberapa kriteria pengukuran performansi, seperti minimasi *makespan*, minimasi banyaknya *job* yang terlambat, dan sebagainya.

2.3.1 Definisi Penjadwalan

Conway (1967) mendefinisikan penjadwalan sebagai "*Scheduling is the task of assigning each operation to a specific position or The time scale of the specific machine".* Sedangkan, menurut Baker (1974) penjadwalan produksi memiliki definisi sebagai proses pengalokasian sumber-sumber untuk memilih sekumpulan tugas dalam jangka waktu tertentu. Definisi yang diberikan Baker (1974) mengandung dua arti, yaitu:

a. Penjadwalan merupakan fungsi pengambilan keputusan yaitu menentukan jadwal (nilai praktis).
b. Penjadwalan merupakan suatu teori, yaitu sekumpulan prinsip-prinsip dasar, model-model, teknik-teknik, dan kesimpulan-kesimpulan logis dalam proses pengambilan keputusan yang memberikan pengertian dalam fungsi penjadwalan (nilai konseptual).

Selain itu, penjadwalan merupakan proses pengorganisasian, pemilihan, dan pemberian waktu dalam penggunaan sumber daya untuk melaksanakan aktivitas yang diperlukan dalam menghasilkan *output* yang diinginkan, dengan memenuhi waktu yang ditetapkan dan kendala-kendala hubungan antara waktu dan aktivitas. Keputusan yang dibuat dalam penjadwalan meliputi:

a. Pengurutan pekerjaan (*sequencing*)
b. Waktu mulai dan selesai pekerjaan (*timing*)

c. Urutan proses suatu pekerjaan (*routing*)

Persoalan penjadwalan timbul apabila beberapa pekerjaan akan dikerjakan secara bersamaan, sedangkan sumber yang dimiliki terbatas. *Input* dari suatu penjadwalan mencakup jenis dan banyaknya *part* yang akan dioperasi, urutan ketergantungan antar operasi, waktu proses untuk masing-masing operasi, serta fasilitas yang dibutuhkan oleh setiap operasi. Sedangkan *output* dari penjadawalan meliputi *dispatch list*, yaitu daftar yang menyatakan urutan pemrosesan *part* serta waktu mulai dan selesai pemrosesan *part*.

2.3.2 Tujuan Penjadwalan

Tujuan penjadwalan adalah sebagai berikut:

1. Menurut Baker (1974), tujuan penjadawlan umumnya adalah sebagai berikut:
   a. Meningkatkan produktifitas mesin, yaitu dengan mengurangi waktu mesin menganggur.
   b. Mengurangi persediaan barang setengah jadi dengan jalan mengurangi jumlah rata-rata pekerjaan yang menunggu dalam antrian suatu mesin karena mesin tersebut sibuk.
   c. Mengurangi keterlambatan suatu pekerjaan. Setiap pekerjaan mempunyai batas waktu (*due date*) penyelesaian, jika pekerjaan tersebut diselesaikan melewati batas waktu yang ditentukan maka pekerjaan tersebut dinyatakan terlambat. Dengan metoda penjadwalan maka keterlambatan ini dapat dikurangi, baik waktu maupun frekuensi.
2. Bedworth (1987) mengidentifikasikan beberapa tujuan dari aktivitas penjadwalan, adalah sebagai berikut:
   a. Meningkatkan penggunaan sumber daya atau mengurangi waktu tunggunya, sehingga total waktu proses dapat berkurang dan produktivitas dapat meningkat.
   b. Mengurangi persediaan barang setengah jadi atau mengurangi sejumlah pekerjaan menunggu dalam antrian ketika sumber daya yang ada masih mengerjakan tugas yang lain. Teori Baker mengatakan, jika aliran kerja suatu jadwal konstan, maka antrian yang mengurangi rata-rata waktu alir akan mengurangi rata-rata persediaan barang setengah jadi.

c. Mengurangi beberapa kelambatan pada pekerjaan yang mempunyai batas waktu penyelesaian sehingga akan meminimalisasi *penalty cost* (biaya keterlambatan).

d. Membantu pengambilan keputusan mengenai perencanaan kapasitas pabrik dan jenis kapasitas yang dibutuhkan sehingga penambahan biaya yang mahal dapat dihindarkan.

Sehingga secara sederhana, penjadwalan yang baik seharusnya simpel, mudah dimengerti dan dapat dilaksanakan oleh pihak manajemen dan oleh siapapun yang menggunakannya. Aturan-aturan penjadwalan seharusnya cukup kuat tetapi mempunyai tujuan yang rea*list*is sehingga cukup fleksibel untuk memecahkan masalah yang tidak terprediksi sebelumnya dan membolehkan satu perencanaan ulang.

2.4 Elemen penjadwalan

Penjadwalan memiliki elemen-elemen penting serta hubungan sebagai berikut:

1. *Job*

   Job didefinisikan sebagai pekerjaan yang harus diselesaikan untuk mendapatkan suatu produk. *Job* terdiri dari beberapa operasi yang harus dikerjakan (minimal 1 operasi). Manajemen melalui perencanaan yang telah dibuat/berdasarkan pesanan dari pelanggan yang memberikan *job* kepada lantai kerja pabrik untuk dikerjakan. Informasi yang dipunyai oleh suatu *job* ketika datang ke lantai kerja pabrik adalah operasi-operasi yang harus dilakukan didalamnya (dari bagian *engineering*) saat *job* harus diselesaikan dan pada saat *job* mulai dapat dikerjakan.

2. Operasi

   Operasi adalah himpunan bagian penyusun *job*. Untuk menyelesaikan suatu *job*, operasi dalam *job* diurutkan dalam suatu urutan pengerjaan tertentu. Urutan tersebut ditentukan pada saat perencanaan proses. Suatu operasi baru dapat dikerjakan apabila operasi atau proses yang mendahuluinya sudah dikerjakan terlebih dahulu. Matriks *routing* berisikan informasi mengenai urutan pengerjaan dan jenis mesin yang digunakan dalam setiap operasi. Setiap operasi mempunyai waktu proses.

3. Mesin

   Mesin adalah sumber daya yang diperlukan untuk mengerjakan proses penyelesaian suatu *job*. Setiap mesin hanya dapat memproses satu tugas pada saat tertentu.

2.5 *Output* dan *Input* Sistem Penjadwalan

2.5.1 *Output* Sistem Penjadwalan

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-order* yang diterima atau diperkirakan dengan kapasitas yang tersedia. Pembebanan dilakukan dengan menugaskan *order-order* pada fasilitas-fasilitas, operator-operator, dan peralatan tertentu.
2. Pengurutan (*sequencing*)

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

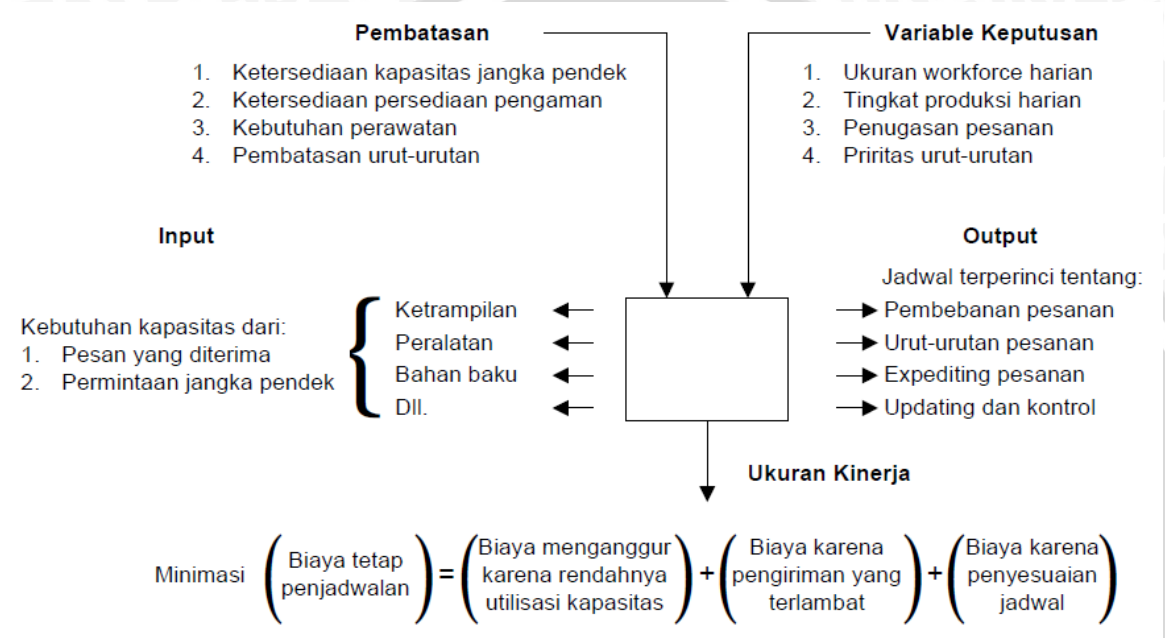
   Prioritas *job* merupakan prioritas kerja tentang *job-job* mana yang diseleksi dan diprioritaskan untuk diproses.
4. Pengendalian kinerja penjadwalan, dilakukan dengan:
   a. Meninjau kembali status *order-order* pada saat melalui sistem tertentu.
   b. Mengatur kembali urutan-urutan.
5. *Up-dating* jadwal, dilakukan sebagai refleksi kondisi operasi yang terjadi dengan merevisi prioritas-prioritas.

2.5.2 *Input* Sistem Penjadwalan

Pekerjaan-pekerjaan yang berupa alokasi kapasitas untuk *order-order*, penugasan prioritas *job*, dan pengendalian jadwal produksi membutuhkan informasi terperinci, dalam informasi-informasi tersebut akan menyatakan *input* dari sistem penjadwalan. Pada bagian ini, kita harus menentukan kebutuhan-kebutuhan kapasitas dari *order-order* yang dijadwalkan dalam hal macam dan jumlah sumber daya yang digunakan. Untuk produk-produk tertentu, informasi ini diperoleh dari lembar kerja operasi (berisi keterampilan dan peralatan yang dibutuhkan, waktu standar, dll) dan BOM (berisi kebutuhan-kebutuhan akan komponen, *sub* komponen, dan bahan pendukung). Kualitas dari keputusan-keputusan penjadwalan sangat dipengaruhi oleh

ketepatan *estimasi input-input* diatas. Oleh karena itu, pemeliharaan catatan terbaru tentang status tenaga kerja dan peralatan yang tersedia dan perubahan kebutuhan kapasitas yang diakibatkan perubahan desain produk atau proses menjadi sangat penting.

Bila digambarkan, maka elemen-elemen *output-input*, prioritas-prioritas dan ukuran kinerja dari sistem penjadwalan akan tampak seperti pada gambar dibawah ini:



Gambar 2.1 *Input-Output* Sistem Penjadwalan
Sumber : Nasution (2003 : 176)

2.6 Istilah-istilah Umum Dalam Penjadwalan

Istilah yang umum digunakan dalam penjadwalan adalah sebagai berikut:

1. Waktu proses (*processing time*),

   t_i (2-1)

   Waktu proses merupakan estimasi lamanya waktu yang dibutuhkan mesin ke-*k* untuk menyelesaikan operasi ke-*j* dari pekerjaan (*job*) ke-*i*, kadang-kadang didalamnya sudah tercakup waktu yang dibutuhkan untuk persiapan dan pengaturan mesin (waktu *set-up*).

2. Waktu siap (*ready time*), menunjukkan saat pekerjaan ke-*i* dapat dikerjakan,

   r_i, (2-2)

3. Batas waktu penyelesaian (*due date*), merupakan batas waktu yang diperbolehkan untuk menyelesaikan suatu pekerjaan.

   D_i, (2-3)

4. Waktu menunggu (*waiting time*), adalah waktu tunggu pekerjaan *i* dari saat pekerjaan siap dikerjakan sampai saat operasi pendahulu selesai.,

   w_i, (2-4)

5. Waktu penyelesaian (*completion time*), merupakan rentang waktu mulai dari awal (*t=0*) sampai pekerjaan *i* selesai dikerjakan.

   C_i, (2-5)

6. Waktu tunggal (*flow time*), adalah waktu antara saat dimana pekerjaan *i* telah siap untuk dikerjakan sampai pekerjaan selesai.

   F_i, (2-6)

7. *Makespan* (*Ms*), merupakan jangka penyelesaian suatu penjadwalan (penjumlahan seluruh waktu proses). *max Ms* = Keterlambatan (*lateness*), L_i adalah perbedaan antara *completion time* dengan *due date*, sehingga bisa (+) atau (-).

   $Li = Ci - Di$ (2-7)

   $L_i < 0$ (*negatif*), saat penyelesaian memenuhi batas *due date* atau ketika *job* dapat diselesaikan lebih cepat.

   $Li > 0$ (*positif*), saat penyelesaian melampaui batas *due date* (*tardy job*)

8. Kelambatan (*tardiness*), adalah keterlambatan penyelesaian suatu pekerjaan dari saat *due date*.

   $T_i = max \{0, L_i\}$: hanya melihat L yang > 0 (positif) (2-8)

9. *Slack Time,* adalah waktu sisa yang tersedia bagi suatu pekerjaan (waktu proses– *dua date*).

   $SL_i = d - t$ (2-9)

10. *Set-up Time,* adalah waktu yang dibutuhkan untuk kegiatan persiapan sebelum pemrosesan *job* dilaksanakan.

    S_i (2-10)

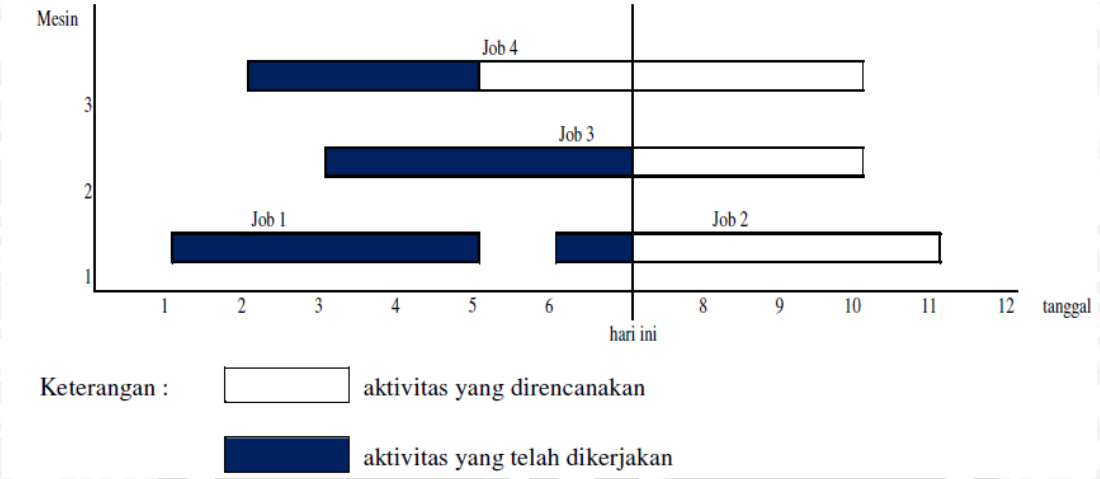
11. *Arrival Time,* adalah saat *job* mulai berada di *shop floor*.

    a_i (2-11)

12. *Delivery Date*, adalah saat pengiriman *job* dari *shop floor* ke proses berikutnya atau ke konsumen.

    d_i (2-12)

13. *Gantt Chart,* merupakan peta visual yang menggambarkan *loading* (beban mesin) dan *scheduling* (urutan pemrosesan *job* dan saat mulai dan saat selesai suatu *job*). Contoh *Gantt Chart* dapat dilihat pada Gambar 2.2.

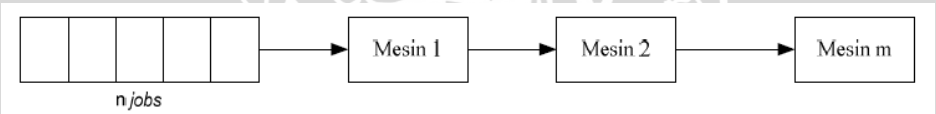


Gambar 2.2. Contoh *Gantt Chart.*
Sumber : Andriana (2010)

2.7 Penjadwalan N *Job* M Mesin

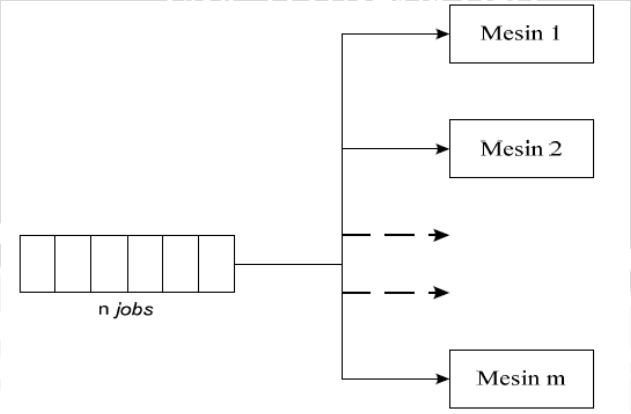
Pada model pertama klasifikasi masalah penjadwalan, terdapat penjadwalan untuk mesin ganda atau penjadwalan n *job* m mesin. Model ini terbagi lagi untuk mesin seri dan paralel, untuk mesin seri pekerjaan harus dikerjakan pada beberapa mesin secara berurutan, sedangkan mesin paralel tiap pekerjaan hanya dikerjakan pada satu mesin.

- Menggunakan mesin seri.



Gambar 2.3 n *job* m mesin pada mesin seri.

- Menggunakan mesin paralel.



Gambar 2.4 n *job* m mesin pada mesin paralel.

2.8 Penjadwalan Dalam Sistem Produksi *Flexible Job Shop*

Penjadwalan mempunyai metode yang berbeda-beda untuk setiap tipe sistem produksi karena setiap sistem mempunyai karakteristik yang berbeda satu dengan yang lainnya. Demikian pula dengan sistem produksi *flexible job shop*. Ciri khas persoalan *flexible job shop* adalah aliran pekerjaan dalam *shop* tidak searah (*unidirectional*) dan terdapat beberapa *work center* dengan pengaturan mesin yang paralel di setiap *work center*-nya.

Pinedo (2008) menjelaskan bahwa pengertian dari *flexible job shop* adalah generalisasi dari sistem *job shop* dan lingkungan mesin yang paralel. Dibandingkan dengan *m* mesin dalam urutan seri, dalam sistem ini terdapat *c work center* yang berisi beberapa mesin identik yang disusun secara paralel. Masing-masing *job* memiliki urutan tertentu untuk dikerjakan; *job j* membutuhkan proses di masing-masing *work center* hanya pada satu mesin dan semua mesin bisa melakukannya. Jika ada sebuah *job* yang dalam prosesnya mengharuskan melewati *work center* lebih dari sekali waktu, maka *β-field* berisi urutan dari sirkulasinya.

Penjadwalan *flexible job shop* adalah pengurutan pekerjaan untuk lintas produk yang tidak beraturan (tata letak pabrik berdasarkan proses). Mahmudy, dkk (2013) mengatakan bahwa penjadwalan pada proses produksi tipe *flexible job shop* lebih sulit dibandingkan penjadwalan *job Shop* biasa. Hal ini disebabkan oleh 2 alasan, yaitu:

1. Memilih mesin dimana setiap operasi bisa dikerjakan (*routing problem*).
2. Menyusun rute dari operasi-operasi ke mesin (*scheduling problem*).

Faktor-faktor tersebut diatas menghasilkan sangat banyak kemungkinan kombinasi dari pembebanan (*loading*) dan urutan-urutan (*sequencing*). Perhitungan dari identifikasi dan evaluasi jadwal-jadwal yang mungkin menjadi sangat sulit sehingga banyak perhatian diarahkan pada riset penjadwalan *flexible job shop*. Selain itu, persiapan suatu penjadwalan *job shop,* penyesuaian dan pembaharuannya membutuhkan investasi yang sangat besar.

Mahmudy, dkk (2013) memberikan pengertian secara rinci mengenai *Flexible Job Shop* dalam lingkungan manufaktur dimana terdapat beberapa *job J* yang independen yang harus diselesaikan dengan menggunakan *M machines*. Setiap *job J* mempunyai beberapa operasi order yang *non-preemptable*. Untuk setiap operasi *k* dari *job j* ($O_{j,k}$) ada beberapa set alternatif mesin ($M_{j,k}$) sehingga operasi dapat diproses. Waktu yang dibutuhkan untuk memproses operasi $O_{j,k}$ di mesin *m* ($m \epsilon M_{j,k}$) dinotasikan

sebagai $T_{j,k,m}$. Mendukung penyelesaiannya maka dalam pengerjaannya menggunakan asumsi sebagai berikut:

1. Semua *job* siap untuk diproses pada saat t = 0.
2. Semua operasi hanya bisa diproses oleh satu mesin.
3. Setiap mesin bisa mengerjakan paling banyak satu operasi pada suatu waktu tertentu.
4. Mesin selalu tersedia (tidak pernah rusak)
5. Waktu *set-up* dari operasi sudah termasuk ke dalam waktu operasi (T*j,k,m*).

2.9 Sistem Pengendalian Situasi Produksi

Berikut bebrapa anggapan mengenai klasifikasi terhadap situasi produksi:

1. Tujuan pengklasifikasian adalah untuk memisahkan antara berbagai situasi kontrol produksi yang berbeda.
2. Perbedaan dalam sistem kontrol dapat dijelaskan oleh karakteristik situasi produksi dalam hubungannya dengan lingkungan.
3. Pemisahan didasarkan pada sifat *order* pelanggan dan aturan yang dimainkan dalam proses produksi.

Perbedaan yang mendasar antara situasi-situasi produksi pada perusahaan adalah saat menerima *order* pelanggan relatif terhadap produksi produk akhir. Faktor penentu suatu situasi produksi adalah *lead time*. Jika kesedian waktu menunggu konsumen lebih kecil dari *lead time*, maka organisasi tersebut harus memelihara persedian produk jadi. Jika konsumen mentolerir menunggu beberapa hari, organisasi akan lebih memilih strategi merakit untuk dipesan atau memproduksi untuk dipesan.

Pembagian klasifikasi situasi produksi, sebagai berikut (Nasution, 2003):

1. *Make to stok*

   Mengubah komponen tingkat rendah dan bahan mentah keseluruhan menjadi produk akhir untuk mengantisipasi *order* pelanggan.
2. *Assemble to order*

   Mengubah komponen tingkat rendah dan bahan mentah menjadi level manufaktur tertentu dan membentuk *order* pelanggan bila menerima pesanan.
3. *Make to order*

   Sangat sedikit atau sama sekali tidak memiliki material tingkat rendah hingga pesanan pelanggan diterima.

4. *Engineer to order*

   Sangat sedikit mengetahui tentang apa yang akan diproduksi hingga pesanan diterima dan membuat spesifikasi engineering-nya. Aktivitas pada sistem produksi dan sistem kontrol produksi seluruhnya dikendalikan oleh pesanan pelanggan.

2.10 Metode Penjadwalan

Berikut merupakan beberapa aturan metode yang digunakan dalam penelitian ini menurut Pinedo (2009):

2.10.1 *Earliest Due date* (EDD)

Metode *earliest due date* (EDD) mengurutkan jadwal berdasarkan waktu *due date* setiap *job* yang dikerjakan. Aturan ini mengabaikan waktu kedatangan dan total waktu proses setiap *job*. Artinya *job* yang memiliki *due date* paling awal dibandingkan *job-job* yang lainnya maka akan diprioritaskan untuk dikerjakan terlebih dahulu pada suatu mesin. Aturan ini cenderung digunakan untuk meminimumkan maksimum *lateness* pada *job-job* yang ada dalam antrian.

2.10.2 *First Come First Serve* (FCFS)

Merupakan metode penjadwalan yang memberikan prioritas tertinggi pada operasi yang masuk S_t (stasiun ke-t) lebih dahulu. Sedangkan menurut Ginting (2009) metode ini merupakan aturan penjadwalan yang menjadwalkan *job* yang datang akan diproses lebih dulu sesuai dengan waktu kedatangannya. Jadi, *job* yang datang pertama juga akan dijadwalakan pertama dalam daftar jadwal.

2.11 Algoritma *Tabu Search*

Pinedo (2009) menyebutkan bahwa algoritma *Tabu Search* dapat digunakan sebagai salah satu alternatif penyelesaian untuk permasalahan di lingkungan *Job Shop* dengan jumlah *job* yang sedikit. Menurut Glover dan Laguna (1986) dalam buku Berlianty dan Arifin (2010), algoritma Tabu *Search* merupakan suatu algoritma yang menuntun setiap tahapannya agar dapat menghasilkan kriteria aspirasi yang paling optimum tanpa terjebak ke dalam solusi awal yang ditemukan selama tahapan berlangsung. Sehingga maksud dari algoritma ini adalah mencegah terjadinya

perulangan dan ditemukannya solusi yang sama dalam satu terasi yang akan digunakan lagi pada iterasi yang selanjutnya (tabu!).

Metode ini mempunya dua macam *tools* yaitu *adaptive memory* dan *responsive exploration*. Dengan adanya *adaptive memory* pada tabu *search* ini menuntun suatu prosedur yang mampu dalam melakukan pencarian solusi yang diinginkan dengan lebih ekonomis dan efektif, sedangkan *responsive exploration* lebih menekankan pada tahapan tiap proses yang harus dilalui selama proses pencarian itu berlangsung, dimana pada setiap tahapan tersebut dipunyai suatu variabel keputusan yang akan menuntun pada tahapan selanjutnya sampai akhir proses pencarian dihentikan.

Tujuan akhir dari semua algoritma optimalisasi adalah mendapatkan dengan cepat solusi optimal yang terdekat. *Tabu Search* sudah terbukti cukup efektif untuk menyelesaikan persoalan tersebut, namun beberapa pelaku riset telah menyatakan bahwa ada beberapa kondisi dimana algoritma tersebut bisa bekerja dengan lebih baik. Beberapa langkah yang bisa membuat algoritma *Tabu Search* lebih optimal antara lain :

1. *Aspiration Criterion.*

   Ini adalah sebuah fungsi yang mengontrol, kapan saja dia bisa mengabaikan sebuah *tabu-state* dari sebuah langkah. *Aspiration* criterion, dalam gambaran *general*, akan menerima sebuah tabu *move* jika *cost* dari solusi tersebut akan bisa lebih baik daripada *cost* dari solusi terbaik yang sudah pernah dikunjungi.

2. *Resizing the Tabu List*.

   Ini adalah teknik untuk memodifikasi panjang dari tabu *list*. Pada kebanyakan kasus, tabu *list* akan diperpendek jika solusi yang lebih baik ditemukan, dan sebaliknya, akan diperpanjang jika solusi yang lebih buruk ditemukan.

   Asumsi utama dibalik teknik ini adalah ketika sebuah solusi yang memenuhi telah ditemukan, kemungkinan masih ada solusi lain dalam beberapa langkah selanjutnya yang juga memenuhi. Menambah jumlah dari pergerakan *neighboring* yang valid akan membuat algoritma bisa menemukan solusi-solusi yang lebih baik dengan kemungkinan yang lebih besar.

3. *Restoring the Best Known Solution.*

   Salah satu cara untuk menghindari pemborosan waktu untuk memeriksa solusi yang kurang optimal adalah dengan mengatur ulang *current solution* sebagai *the best known solution* (solusi yang terbaik yang dari seluruh solusi yang sudah diperiksa) secara perodik.

2.11.1 Konsep Dasar Tabu *Search*

Memori pada *Tabu Search* mempunyai dua sifat, yaitu *explicit* dan *attributive* (Glover & Laguna, 1986 dalam buku Berlianty & Arifin, 2010):

1. *Explicit memory* menyimpan *complete solution* yang umumnya menghabiskan alokasi ruang memori dan waktu, sehingga untuk menghindari hal ini, *complete solution* dikurangi sehingga hanya terdiri dari *elite solution* yang dikunjungi selama pencarian.
2. *Attributive memory* menyimpan informasi tentang atribut dari solusi yang ditemukan yang mungkin dapat berubah dari satu solusi ke solusi lain.

Sedangkan penggunaan dari memori *Tabu Search* ini mengacu pada *adaptive memory* seperti yang telah dijelaskan sebelumnya. Struktur memori dalam *Tabu Search* menggunakan empat prinsip utama yaitu:

1. *Recency* atau yang lebih lengkap dikenal sebagai *recency based memory*, memori yang tetap menjaga struktur terbaik dari solusi awal yang ditemukan selama proses pencarian pada setiap iterasinya, sehingga apabila pada suatu terasi ditemukan struktur/solusi yang lebih baik maka solusi ini akan tetap dipertahankan sampai ditemukannya solusi baru yang lebih baik lagi. Agar *recency* dapat bekerja dengan baik, maka didukung dengan Tabu *List* yang berisi Tabu *Active*. Tabu *List* adalah suatu set memori yang memberikan informasi tentang struktur dari solusi awal yang pernah diambil, sehingga tabu *list* akan menghindari digunakannya struktur yang sama untuk menghasilkan suatu solusi jika struktur tersebut pernah dipakai. Sedangkan tabu *active* adalah tabu *list* yang berada di urutan paling akhir, artinya bahwa solusi itu merupakan solusi dari terasi yang paling akhir yang sedang dikunjungi. Untuk membatasi *range* atau jumlah dari tabu *list* sendiri dinamakan dengan Tabu *Tenure*.
2. *Frequency*, menyediakan sebuah tipe informasi yang merupakan kumpulan informasi yang telah direkam oleh *recency based memory*. Sehingga *frequency* dan *recency* dapat saling melengkapi untuk membentuk suatu informasi permanen guna mengevaluasi pergerakan/*move* yang terjadi.
3. *Quality* adalah kemampuan untuk membedakan solusi terbaik yang dikunjungi selama pencarian atau iterasi berlangsung.

4. *Influence* mempertimbangkan efek yang terjadi dari pemilihan solusi yang dipilih selama pencarian berlangsung, tidak hanya kualitas saja yang dipertimbangkan melainkan juga strukturnya.

2.11.2 Mekanisme *Tabu Search*

Tabu *search* dalam penyelesaiannya harus melewati setiap tahap tertentu yang telah diatur (Glover, 1986 dalam Berlianty & Arifin, 2010), sebagai berikut:

1. Membangkitkan solusi awal

   Yaitu menentukan acuan awal atau kondisi optimal solusi sebagai pembanding ketika proses *Tabu Search* dimulai.
2. Menentukan kriteria aspirasi

   Kriteria aspirasi ini fungsinya sebagai tujuan atau *goal* yang akan dicapai.
3. Melakukan *move*

   Ada beberapa macam *move* yang dapat dipilih selama proses pencarian berlangsung yaitu :

   a. *Local Search*, yang terdiri dari dua macam, yaitu :
      - *Insertion*, yaitu memilih secara acak satu bagian struktur untuk dipindah ke bagian yang lain.
      - *Swap*, yaitu memilih secara acak dua bagian struktur untuk selanjutnya ditukar posisinya.

   b. *Neighborhood Search*

      Untuk pencarian dengan teknik ini kemungkinan atribut dari struktur akan dipindah-pindah. Permutasi *n-charge neighborhood* mengambil n elemen matriks solusi. Perubahan yang dipakai oleh dua *neighborhood* dengan melakukan *swap* elemen matriks atau kombinasi elemen itu dengan menukar elemen lain dalam matriks.
4. Untuk menghindari terulangnya langkah yang diambil, maka dilakukan tabu *Test*. Tabu *Test* memanfaatkan tabu *list* yang sudah ada, tabu *list* berisi atribut solusi-solusi yang telah dikunjungi sebelumnya. Tujuan sebenarnya dari tabu *list* bukan hanya sekedar untuk menghindari perulangan langkah yang telah diambil, tetapi lebih kepada agar tidak mundur. Untuk mencegah perulangan, daftar solusi yang telah dicapai disimpan di dalam sebuah tabel. Bagaimanapun situasi perulangan jarang terjadi, karena telah dikombinasikan dengan beberapa *neighborhood*, dimana

akan memperluas *search space* dan membuat kemungkinan mengulang solusi yang telah dikunjungi menjadi hampir tidak mungkin. Solusi yang telah dimasukkan ke dalam tabel akan dinamakan sebagai tabu. Kondisi pengecualiannya hanya untuk *blockage situation*. Jika kondisi ini terjadi maka semua jalur akan menjadi tabu juga. Untuk menghindari ini maka digunakan jalur paling awal dalam tabu *list*.

5. *Alternative move* yang lolos tabu *Test* masih harus melewati *aspiration Test* apakah bisa melewati *aspiration threshold* atau tidak, jika tidak maka diteruskan ke iterasi selanjutnya.
6. Jika alternatif *move* mempunyai *aspiration* kriteria yang lebih baik daripada *aspiration threshold* maka dilakukan eksekusi terhadap alternatif *move* tersebut dan memperbarui memori yang tidak relevan.
7. Jika aturan berhenti sudah memenuhi syarat berhenti, maka pencarian berhenti.

Langkah-langkah penyelesaian dari algoritma *Tabu Search* akan dijelaskan sebagai berikut (Pinedo, 2009).

Langkah 1

Atur nilai k=1

Pilih satu urutan inisiasi S_1 menggunakan metode heuristik tertentu

Atur $S_0 = S_1$

Langkah 2

Pilih satu jadwal kandidat S_c dari solusi tetangga S_k

Jika *move* dari $S_k \rightarrow S_c$ terlarang oleh mutasi dalam *tabu-list*

Maka atur $S_{k+1} = S_k$ dan lanjut ke langkah 3

Jika *move* dari $S_k \rightarrow S_c$ tidak terlarang oleh mutasi manapun dalam *tabu-list*

Maka atur $S_{k+1} = S_c$ dan

masukkan mutasi kebalikannya dalam *tabu-list* yang paling atas

Pindah semua masukan dalam *tabu-list* satu posisi ke bawah dan hapus masukan paling akhir di dalam *tabu-list*

Jika $G(S_c) < G(S_0)$, atur $S_0 = S_c$;

Lanjut ke langkah 3

Langkah 3

Tambahkan nilai k dengan 1

Jika nilai k = N maka BERHENTI

Jika tidak, kembali ke langkah 2.