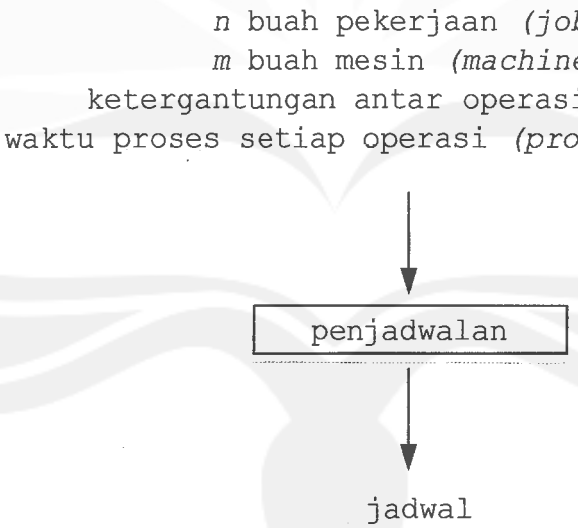
BAB 3

DASAR TEORI

3.1. Penjadwalan Produksi Secara Umum

Penjadwalan adalah suatu usaha menggunakan waktu agar diperoleh efisiensi terhadap waktu yang sesuai dengan kebutuhan. Penjadwalan merupakan proses pembuatan keputusan yang berperan sangat penting dalam industri manufaktur dan jasa (Pinedo, 1999).

Permasalahan dalam penjadwalan adalah bagaimana mengalokasikan pekerjaan ke mesin, pada kondisi mesin mempunyai kapasitas dan jumlah yang terbatas.



Gambar 3.1. Persoalan Penjadwalan

Masalah penjadwalan muncul apabila ada sekumpulan pekerjaan yang harus diproses dalam waktu yang sama sedangkan mesin yang tersedia jumlahnya terbatas namun

tetap bisa mencapai hasil optimal seperti yang diharapkan. Salah satu usaha yang dilakukan untuk mengatasi kendala tersebut adalah dengan menjadwalkan pekerjaan-pekerjaan tersebut sehingga dapat digunakan secara optimal. Secara umum masalah penjadwalan dapat dinyatakan sebagai suatu persoalan seperti pada gambar 3.1. di atas.

Penjadwalan produksi memiliki beberapa tujuan umum, antara lain :

a. Mengurangi penumpukan barang setengah jadi pada lintasan produksi *(waiting line)*
b. Meningkatkan produktivitas mesin dengan mengurangi waktu tunggu mesin *(idle time)*
c. Mengurangi keterlambatan karena waktu proses suatu pekerjaan telah melampaui jatuh temponya
d. Mengurangi waktu siklus produksi sehingga biaya produksi dapat ditekan

3.1.1. Jenis-Jenis Penjadwalan

Menurut jenisnya, penjadwalan dapat dibedakan menjadi dua, yaitu :

a. Penjadwalan maju *(forward scheduling)*

   Penjadwalan maju adalah penjadwalan operasi dari saat mulai, bergerak searah dengan pergerakan waktu, sampai seluruh operasi terjadwal. Biasanya digunakan bila yang diinginkan adalah menyelesaikan *job* secepat-cepatnya, atau untuk mengantisipasi bila saat pelaksanaan jadwal ada banyak kemungkinan gangguan di waktu yang akan datang.

b. Penjadwalan mundur *(backward scheduling)*

   Penjadwalan mundur adalah penjadwalan operasi dari *due-date*, bergerak berlawanan arah dengan pergerakan

waktu, sampai seluruh operasi terjadwalkan. Biasanya digunakan apabila yang diinginkan adalah menyelesaikan *job* tepat saat *due-date*nya.

3.1.2. Persoalan Umum dari Penjadwalan *Flow Shop* dan *Job Shop*

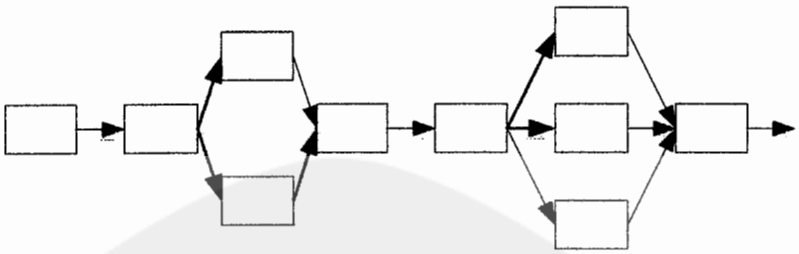
Pada proses produksi dan industri manufaktur sering ditemui istilah teori penjadwalan. Asumsi yang harus dimulai dalam persoalan pada proses produksi dan industri manufaktur adalah adanya *n part* yang akan diproses melalui *m* mesin (m_1, m_2, m_3,...m_n), sedangkan proses pekerjaan pada mesin disebut sebagai operasi. Simbol bagi operasi ini adalah n_i.

Pada *flow shop*, kendala teknologi yang dihadapi adalah setiap pekerjaan harus diproses melewati mesin dengan aturan yang khusus yaitu setiap pekerjaan harus melewati mesin yang ada secara berurutan. Pada *job shop*, setiap pekerjaan tidak perlu melewati mesin secara berurutan. Setiap operasi memerlukan waktu yang pasti, yang disebut dengan waktu proses dan disimbolkan dengan t_{ij}. Secara konvensi, di dalam t_{ij} telah dimasukkan waktu yang diperlukan untuk mengatur mesin, waktu untuk *set up* mesin sebelum melakukan pekerjaan serta waktu transportasi pekerjaan untuk sampai ke mesin. Setiap pekerjaan harus diketahui waktu prosesnya secara pasti.

Berikut ini merupakan diagram pola aliran proses dari *flow shop* dan *job shop* :

a. *Flow shop*

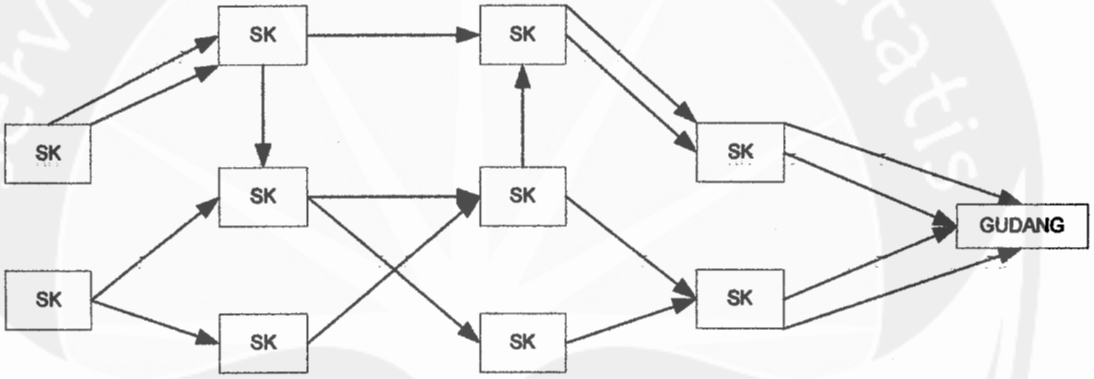
Setiap pekerjaan memiliki pola aturan proses yang sama.



Gambar 3.2. Pola Aliran Sistem Produksi *Flow Shop*

b. *Job shop*

Setiap pekerjaan memiliki aturan proses tersendiri sesuai dengan jenis pekerjaan.



Gambar 3.3. Pola Aliran Sistem Produksi *Job Shop*

3.1.3. Tipe Jadwal pada Sistem Produksi *Job Shop*

Pada penjadwalan *job shop*, karakteristik pekerjaan yang diselesaikan harus melewati beberapa mesin *(routing)* dan masing-masing rute yang ditempuh tiap-tiap pekerjaan berlainan.

Karakteristik persoalan penjadwalan *job shop* ialah penggunaan mesin oleh lebih dari satu pekerjaan sehingga ada keterbatasan waktu penggunaan. Akibatnya mungkin akan ada antrian pekerjaan dan keadaan ini lebih diperumit dengan adanya batas waktu *(due date)* tiap pekerjaan.

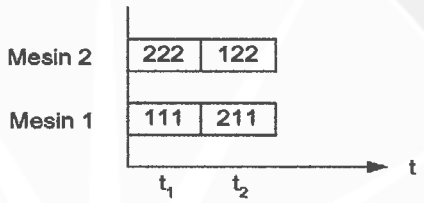
Beberapa definisi tentang jadwal pada proses produksi *job shop* adalah :

a. Jadwal *feasible*

Suatu jadwal dikatakan *feasible* jika seluruh operasi dari semua *job* telah ditugaskan dan ketentuan *routing* operasi telah dipenuhi (atau dengan kata lain tidak ada *overlap* antar operasi).

b. Jadwal semi aktif

Adalah sekumpulan jadwal *feasible* di mana tidak satu pun operasi dapat dikerjakan lebih awal tanpa mengubah susunan operasi pada mesin. Contoh jadwal semi aktif dapat dilihat pada gambar 3.4.

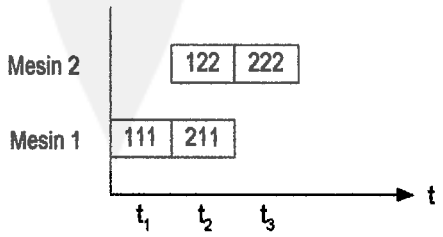


Gambar 3.4. Jadwal Semi Aktif

Pada gambar 3.4., jika 211 (*job* 2 operasi 1 di mesin 1) dikerjakan lebih dahulu maka akan mengubah susunan operasi pada mesin 1.

c. Jadwal aktif

Adalah sekumpulan jadwal *feasible* di mana tidak satu pun operasi dapat dipindahkan lebih awal tanpa menunda operasi yang lain. Contoh jadwal aktif dapat dilihat pada gambar 3.5.



Gambar 3.5. Jadwal Aktif

Pada gambar 3.5., jika 211 ditukarkan dengan 111 di mesin 1 maka akan mengakibatkan jadwal 222 dan 122 terpaksa ditukar pula.

d. Jadwal *non delay*

Adalah kumpulan jadwal *feasible* di mana tidak satu pun mesin dibiarkan menganggur jika pada saat yang bersamaan terdapat operasi yang memerlukan mesin tersebut.

3.1.4. Aturan Prioritas *(Priority Dispatching)*

Prosedur *dispatching* lebih umum digunakan dalam praktek masa kini, dimana prosedur ini dapat dengan mudah diadaptasikan untuk kedatangan *job* yang dinamik, kerusakan mesin, atau faktor-faktor lain yang dapat membuat status produksi melebihi waktu yang tersedia (Baker, 1974).

Teknik *priority dispatching* menentukan aturan prioritas untuk memilih satu operasi di antara operasi-operasi yang mengalami konflik pada mesin *m* pada setiap tahap. Aturan prioritas digunakan untuk memilih operasi mana yang akan dilakukan terlebih dahulu. Penyelesaian konflik penjadwalan dapat dilakukan dengan berbagai aturan prioritas tertentu sehingga dapat merupakan algoritma heuristik tertentu.

Beberapa contoh aturan prioritas pekerjaan, yaitu :

a. R *(Random)*

Job yang akan diproses dipilih secara acak tanpa memperhatikan aturan tertentu.

b. FCFS *(First Come First Serve)*

Aturan ini mempertimbangkan waktu kedatangan *job*. *Job* yang mempunyai waktu kedatangan lebih awal akan dikerjakan lebih dulu.

c. SPT *(Short Processing Time)*

Aturan ini bertujuan untuk mengurangi *work-in-process inventory*, rata-rata *flow time* dan rata-rata *job lateness*. Urutan *job* pada satu *work center* didasarkan pada waktu proses *job*. *Job* dengan waktu proses lebih kecil akan dikerjakan lebih dulu.

d. EDD *(Earliest Due Date)*

Aturan ini akan bekerja dengan baik jika dihubungkan dengan kriteria keterlambatan *job* karena aturan ini memberikan prioritas pada *job* yang memiliki *due date* lebih dekat untuk dikerjakan lebih dahulu.

e. MWKR *(Most Work Remaining)*

Aturan ini mempertimbangkan sisa waktu proses sampai pekerjaan tersebut diselesaikan. Pekerjaan dengan sisa waktu terbesar dipilih untuk diproses.

3.1.5. Istilah Dasar Penjadwalan

Beberapa istilah yang sering digunakan dalam menyelesaikan masalah penjadwalan antara lain :

a. *Processing Time* (t_i)

Waktu proses adalah estimasi waktu proses suatu pekerjaan hingga pekerjaan itu selesai dikerjakan.

b. *Due date* (d_i)

Due date adalah batas waktu penyerahan pekerjaan selesai kepada pemesan. Apabila *due date* ini dilanggar maka akan dikenakan penalti dengan memperhitungkan kelambatannya.

c. *Completion Time* (C_i)

Adalah waktu pekerjaan ke-*i* dimulai saat *t=0*, sampai semua pekerjaan *i* selesai.

d. *Flow Time* (F_i)

Adalah waktu antara saat mulai sampai saat selesai suatu pekerjaan.

e. *Lateness* (L_i)

Adalah selisih antara waktu penyelesaian (C_i) dan pekerjaan dengan *due date*-nya. Pekerjaan akan bernilai positif *lateness* apabila diselesaikan setelah *due date*-nya dan bernilai negatif *lateness* bila diselesaikan sebelum *due date*-nya.

f. *Tardiness* (T_i)

Jika suatu pekerjaan cepat selesai maka akan mempunyai negatif *lateness* tetapi bernilai *tardiness=0* dan jika suatu pekerjaan bernilai positif *lateness* maka akan bernilai positif *tardiness*.

g. *Slack Time* (SL_i)

Adalah ukuran dari perbedaan antara sisa waktu dari *due date* dan *processing time*.

h. *Makespan (M)*

Adalah waktu antara saat mulai sampai saat selesai operasi terakhir dari seluruh operasi yang ada.

i. *Number of Tardy (NT)*

Adalah jumlah pekerjaan yang terlambat selesai.

3.2. Algoritma *Ant System*

3.2.1. Sekilas Tentang Koloni Semut

Semut adalah serangga sosial yang hidupnya berkoloni. Perilaku semut ditentukan oleh keselamatan dari keseluruhan koloni karena semut secara individu tidaklah begitu berguna. Koloni semut telah diketahui mampu untuk menemukan jalur terpendek dari sarangnya menuju sumber makanan dan kembali lagi ke sarangnya.

Perilaku *foraging* (mencari makan) dan *recruiting* (mendapatkan makanan) merupakan fenomena kerjasama antar sesama semut yang menarik untuk dipelajari. Perilaku ini menjelaskan bagaimana semut menjelajahi berbagai tempat untuk mencari sumber-sumber makanan, kemudian menemukan jalan pulang kembali ke sarangnya untuk memberitahukan penemuan sumber makanan tersebut kepada semut-semut lain di koloninya. Dalam melakukan tindakannya tersebut, semut menggunakan media komunikasi berupa jejak feromon (suatu substansi kimia yang dilepaskan dan diendapkan di tanah untuk kemudian dijadikan petunjuk). Selama mencari sumber makanan, tiap semut mengendapkan fraksi feromonnya untuk menandai jalur yang dilaluinya sebagai indikasi bagi semut lainnya untuk menemukan sumber makanan tersebut. Semakin banyak semut yang mengikuti jejak feromon tersebut maka semakin menarik jejak tersebut untuk diikuti. Proses tersebut dikarakterisasi sebagai *positive feedback loop*, yaitu probabilitas seekor semut untuk memilih suatu jalur akan meningkatkan jumlah semut yang sebelumnya telah memilih jalur yang sama.

Akumulasi feromon bertindak sebagai *distributed memory* (memori terdistribusi) yang digunakan bersama oleh seluruh semut lainnya dan menandai probabilitas jalur yang paling sering dilalui oleh semut antara sarang dan sumber makanannya.

Dalam perjalanannya menuju sumber makanan, semut-semut menjumpai rintangan yang memicu semut-semut tersebut untuk memilih arah perjalanannya secara random. Jika suatu jalur memiliki fraksi feromon yang diendapkan, maka akan mempengaruhi semut berikutnya untuk memilih jalur tersebut, kemudian berhadapan dengan rintangan yang

sama. Hal ini terjadi karena probabilitas feromon yang ada pada jalur tersebut lebih berpengaruh dan bersifat memiliki daya tarik bagi semut lainnya daripada jalur baru yang tidak memiliki endapan feromon sama sekali.

Fakta lain yang perlu diperhatikan adalah bahwa fraksi feromon juga akan menguap seiring berjalannya waktu, bahkan untuk jalur yang lebih panjang memungkinkan fraksi feromon yang ada akan menguap semuanya. Semakin pendek jalur maka semakin sering dilalui oleh semut-semut dari koloni yang selanjutnya juga akan meningkatkan jumlah feromon yang ada di jalur tersebut. Kondisi *positive feedback* dengan akumulasi dan penguapan feromon seperti ini akan secara otomatis menghilangkan probabilitas bagi semut-semut berikutnya untuk memilih jalur yang paling panjang.

3.2.2. Algoritma *Ant System* untuk Penjadwalan *Job Shop*

Algoritma *ant system* merupakan suatu teknik probabilistik untuk menyelesaikan masalah komputasi dengan menemukan jalur terbaik melalui grafik, yang mengambil filosofi sistem biologi alamiah dari perilaku semut dalam menemukan jalur dari koloninya menuju makanan. Algoritma *Ant System (AS)* diperkenalkan oleh Moyson dan Manderick dan secara luas dikembangkan oleh Marco Dorigo sejak tahun 1996. Algoritma ini menggunakan populasi agen yang bekerja sama dan berkomunikasi dengan memori terdistribusi yang digunakan bersama-sama untuk selanjutnya diaplikasikan pada permasalahan optimasi kombinatorial. Berbagai permasalahan yang dapat diselesaikan dengan algoritma semut antara lain *travelling salesman problem, Quadratic Assignment Problem*

(QAP), job shop scheduling problem, pengaturan jalur kendaraan, *graph coloring* dan *network routing.*

Prinsip dasar dari algoritma *ant system* adalah mendapatkan populasi ℓ semut tiruan *(artificial ants)* yang secara siklik membangun solusi dari suatu permasalahan kombinatorial. Algoritma *ant system* menerjemahkan definisi dari permasalahan optimasi ke dalam sebuah *graph* (grafik) yang menggambarkan pergerakan semut-semut sepanjang setiap *branch* (cabang) dari satu node ke node lainnya, kemudian membuat jalur yang merepresentasikan solusi. Dimulai dari *initial node* (node awal), setiap semut memilih node berikutnya dari jalur yang ada berdasarkan *State Transition Rule* (aturan transisi status) sebagai berikut:

$$pij\,(t) = \frac{\left[\tau_{ij}(t)\right]^{\alpha} \cdot \left[\frac{1}{d_{ij}}\right]^{\beta}}{\sum_{j\varepsilon\ allowed\ \ nodes} \left[\tau_{ij}(t)\right]^{\alpha} \cdot \left[\frac{1}{d_{ij}}\right]^{\beta}} \qquad (3.1)$$

dengan $p_{ij(t)}$ adalah probabilitas perpindahan semut dari $node_i$ ke $node_j$ pada waktu ke-t, $\tau_{ij(t)}$ adalah intensitas jejak antara $node_i$ dan $node_j$ pada waktu ke-t, dan d_{ij} merupakan jarak heuristik antara $node_i$ dan $node_j$. Parameter α berpengaruh terhadap probabilitas jumlah feromon yang besarnya $\alpha \geq 0$, sedangkan parameter β akan mempengaruhi jarak heuristik yang nilainya $\beta \geq 0$. Menurut aturan tersebut, node-node dengan jumlah feromon tinggi dan dekat dengan aktual node (dalam hal jarak heuristik) akan memiliki probabilitas yang lebih tinggi untuk dijadwalkan.

Setelah semua semut mengunjungi setiap node dan menghasilkan solusi lengkap maka dilakukan feromon *update* sebagai berikut :

$$\tau_{ij}(t+n) = (1-\rho).\tau_{ij}(t) + \rho.\Delta\tau_{ij}(t+n) \tag{3.2}$$

$$\Delta\tau_{ij}(t+n) = \begin{cases} \dfrac{Q}{f_{evaluasi}(best\_so\_far)} \\ 0, \ otherwise \end{cases} \tag{3.3}$$

dengan $\tau_{ij}(t+n)$ adalah feromon yang telah di-*update* pada akhir tiap siklus, ρ sebagai koefisien evaporasi (penguapan) dan Q adalah jumlah feromon per unit jarak. Kedua persamaan di atas menjelaskan dua hal yang terjadi, yaitu :

a. Fraksi feromon pada semua *edges* pada *graph* pasti menguap.
b. Terjadi penambahan feromon pada *edges* yang menjadi *best-so-far solution*.

Pada permasalahan *job shop*, data-data yang diperlukan berupa data *routing* mesin dan waktu proses masing-masing operasi. Data *routing* mesin dideskripsikan dalam sebuah matriks T, sedangkan data waktu proses dispesifikasikan dalam matriks P. Sebagai contoh, untuk kasus 2x3, maka matriks T dan P adalah sebagai berikut :

$$T = \begin{bmatrix} M_1 & M_2 & M_3 \\ M_2 & M_3 & M_1 \end{bmatrix} \qquad P = \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \end{bmatrix}$$

Pada matriks T, baris matriks merepresentasikan *job* yang menjelaskan sekuens mesin yang akan dijadwalkan sedangkan kolom matriks menunjukkan operasi ke berapa. Kedua matriks tersebut kemudian dibuat *graph* yang menerjemahkan permasalahan *job shop*. *Graph* terdiri dari sejumlah node

dan *edges* seperti pada gambar 3.6. Maksimum jumlah node ditentukan oleh persamaan berikut ini :

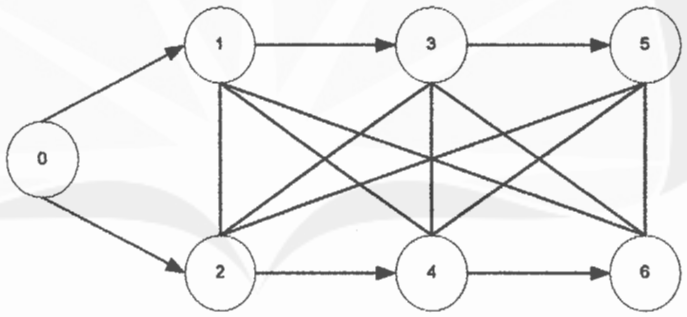
$$Node_{maksimum} = (n \times m) + 1 \quad (3.4)$$

dengan n menunjukkan banyaknya *job* dan m adalah banyaknya mesin. Sementara jumlah *edges* ditentukan oleh :

$$edges = \frac{|O|.(|O|-1)}{2} + n \quad (3.5)$$

$$|O| = n \times m \quad (3.6)$$

$|O|$ mengindikasikan jumlah operasi dalam *job shop*.



Gambar 3.6. *Graph* untuk kasus 2x3

Semua *graph* diawali dengan sebuah *initial node* yang dilambangkan dengan node 0. Node-node pada *graph* merepresentasikan operasi-operasi pada matriks T. Oleh karena itu, node 1 (*job* 1 operasi 1 (O_{11})) menjelaskan elemen T_{11} yang berarti setara dengan M_1. Node satu dangan node lainnya terhubung dengan *edges*. Node-node milik *job* yang sama terhubung secara horisontal dan memiliki satu arah (node 1-3-5 dan node 2-4-6), yang mencerminkan *technological order of processing a job*. Sementara *edges* lainnya terhubung dua arah (*bidirectional*). Demi kesederhanaan pemahaman, maka *edges* yang terhubung

bidirectional diwakili oleh *edges* tak berarah/garis tebal. Masing-masing *edges* memiliki pasangan nilai $\{\tau_{ij}, d_{ij}\}$ yang merepresentasikan jumlah feromon dan jarak heuristik antar node. Waktu proses dari operasi yang dialamatkan pada node j dapat diasumsikan sebagai jarak heuristik antara node i dan node j. Nilai dari jarak heuristik ini dapat dilihat pada matriks P. Hal yang perlu diperhatikan untuk *bidirectional edges* adalah nilainya yang tidak simetris. Artinya, jarak untuk percabangan dari node i ke node j adalah tidak sama dengan jarak percabangan dari node i ke node j dengan arah yang berlawanan. Oleh karena itu, jumlah feromon untuk *bidirectional edges* harus didefinisikan untuk kedua arah tersebut.