

BAB II

TINJAUAN PUSTAKA

2.1 Penjadwalan Produksi

Penjadwalan merupakan proses pengorganisasian waktu pelaksanaan suatu kegiatan operasional. Secara umum, tujuan utama penjadwalan adalah untuk mengoptimalkan waktu proses, mengurangi waktu tunggu pelanggan, menekan jumlah persediaan, serta meningkatkan efisiensi penggunaan fasilitas, tenaga kerja, dan peralatan. Penyusunan jadwal mempertimbangkan berbagai kendala yang ada. Selain itu, penjadwalan bertujuan untuk meminimalkan keterlambatan dari tenggat waktu yang telah ditentukan, guna memastikan kesesuaian dengan jadwal yang telah disepakati bersama konsumen. Penjadwalan distribusi juga berperan dalam meningkatkan produktivitas mesin dan mengurangi waktu tidak produktif atau waktu menganggur (Nugroho & Ekoanindiyo, 2017).

Dalam konteks industri, penjadwalan produksi memegang peran penting sebagai bentuk pengambilan keputusan yang bertujuan untuk mengatur dan mengoordinasikan waktu pelaksanaan kegiatan produksi secara optimal. Salah satu aspek krusial dalam sistem produksi adalah memastikan bahwa setiap pekerjaan dapat dijadwalkan dan diselesaikan tepat waktu sesuai dengan kebutuhan pelanggan. Untuk mencapai hal tersebut, diperlukan perencanaan penjadwalan yang terstruktur dan efisien. Penjadwalan yang baik dapat mengurangi waktu menganggur pada mesin dan tenaga kerja, meminimalkan jumlah barang dalam proses (*work-in-process*), serta meningkatkan efisiensi keseluruhan. Keberhasilan sistem penjadwalan ini umumnya diukur dari seberapa kecil nilai *makespan* yang dihasilkan dalam proses produksi (Syabani & Setiafindari, 2022).

2.2 Tujuan Penjadwalan

Kegiatan produksi memerlukan penjadwalan yang terstruktur agar proses berjalan lancar dan penggunaan sumber daya menjadi lebih efisien. Penjadwalan

produksi dirancang untuk mencapai beberapa tujuan utama, antara lain sebagai berikut (Utama, 2023):

- a) Meminimasi waktu menganggur (*idle time*) pada mesin, sehingga produktivitas mesin dapat meningkat.
- b) Meminimasi persediaan produk WIP (*work in process*) yang disebabkan oleh antrian proses pada suatu mesin. Pengurangan persediaan ini bisa diatasi dengan pengurangan jumlah rata-rata *job* yang menunggu pada sebuah antrian mesin.
- c) Meminimasi keterlambatan (melampaui *due date*), melalui cara seperti berikut :
 - 1) Meminimasi maksimum keterlambatan .
 - 2) Meminimasi total *job* yang terlambat.
- d) Meminimasi biaya pada produksi.
- e) Meminimasi risiko denda dengan cara pemenuhan *due date*, karena ketika terjadi keterlambatan dapat dikenakan denda sesuai dengan kesepakatan.

2.3 Jenis-Jenis Penjadwalan

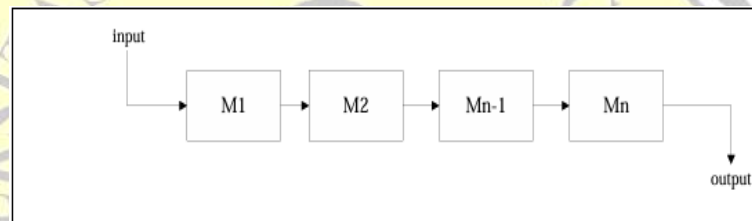
Pada penjadwalan berdasarkan sistem produksi terdapat beberapa jenis penjadwalan yaitu *flowshop*, *job shop* dan *heuristic schedule generation*. Permasalahan yang membedakan antara *job shop* dan *flowshop* adalah pola aliran kerja yang tidak memiliki tahapan-tahapan proses yang sama. Untuk melakukan penjadwalan dengan efektif, diperlukan informasi mengenai waktu proses pada setiap mesin serta jenis pekerjaannya. Data tersebut dapat diperoleh melalui pengukuran waktu kerja serta pengamatan terhadap operator di bagian tertentu. Setelah informasi mengenai jenis pekerjaan dan waktu proses tiap mesin tersedia, barulah proses penjadwalan dapat dilaksanakan. Berikut ini merupakan tipe penjadwalan produksi (Barokah et al., 2016):

1) Penjadwalan Produksi *Flowshop*

Penjadwalan *flowshop* merupakan pola alir dari N buah *job* yang melalui proses yang sama (searah). Model aliran produksi jenis ini sangat cocok

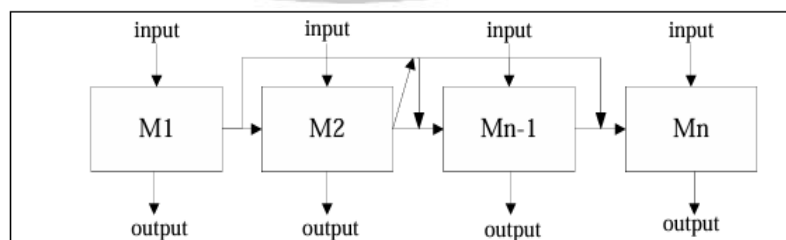
diterapkan pada produk-produk dengan desain yang konsisten dan diproduksi dalam jumlah besar (volume tinggi), sehingga investasi pada peralatan khusus (*special purpose*) dapat segera menghasilkan pengembalian. Terdapat dua pola aliran *flowshop* yaitu *pure flowshop* dan *general flowshop*. Berikut ini merupakan macam-macam pola aliran *flowshop* :

- a. *Pure flowshop* merupakan sistem produksi di mana semua pekerjaan mengikuti jalur proses yang seragam. Mesin-mesin dalam sistem ini diatur sesuai dengan urutan tahapan proses, dan setiap *job* harus melewati satu proses pada setiap tahapan secara berurutan. Berikut pola aliran *pure flowshop* :



Gambar 1. Pola Aliran *Pure Flowshop*
(Sumber : Barokah et al., 2016)

- b. *General flowshop* memiliki pola aliran kerja yang lebih fleksibel dibandingkan *pure flowshop*. Ciri khas dari sistem ini adalah alur proses yang selalu bergerak ke arah kanan (maju), namun urutan operasinya tidak harus selalu berurutan. Meskipun demikian, sistem ini tidak mengizinkan adanya aliran balik dalam proses produksinya. Berikut pola aliran *general flowshop* :



Gambar 2. Pola Aliran *General Flowshop*
(Sumber : Barokah et al., 2016)

2.4 Istilah-Istilah Penjadwalan

Pemahaman yang lebih mendalam mengenai konsep penjadwalan produksi memerlukan pengenalan terhadap beberapa istilah umum yang sering digunakan dalam proses penjadwalan. Berikut ini beberapa istilah umum yang sering digunakan dalam penjadwalan, antara lain (Sidabutar et al., 2020):

1. Waktu Proses (*Processing Time*) adalah estimasi durasi yang dibutuhkan untuk menyelesaikan suatu pekerjaan atau tugas produksi.
2. *Makespan* merupakan total waktu yang dibutuhkan untuk menyelesaikan seluruh *job* dari awal hingga akhir.
3. *Due date* adalah tenggat waktu yang dijanjikan kepada pelanggan untuk menyerahkan produk yang telah selesai.
4. *Completion Time* adalah waktu yang menunjukkan kapan suatu tugas selesai, dihitung sejak awal pengerjaan tugas pertama pada waktu $t = 0$.
5. *Lateness* menunjukkan selisih antara waktu penyelesaian (*completion time*) dan batas waktu yang telah ditentukan (*due date*) untuk sebuah tugas.
6. *Tardiness* adalah ukuran keterlambatan suatu tugas. Jika tugas selesai setelah *due date*, nilainya positif, sedangkan jika selesai lebih awal nilainya bisa dianggap nol (atau negatif dalam konteks tertentu).
7. *Early* menunjukkan bahwa suatu tugas diselesaikan sebelum batas waktu yang ditentukan (*due date*).
8. *Flow Time* merupakan total waktu yang dibutuhkan sejak sebuah tugas tersedia untuk dikerjakan hingga selesai, termasuk waktu tunggu dan waktu proses.
9. *Slack* adalah selisih waktu antara *due date* dan waktu proses suatu tugas, menunjukkan seberapa longgar waktu penyelesaiannya.
10. *Heuristic* adalah pendekatan pemecahan masalah yang dirancang untuk menghasilkan solusi yang cukup baik, meskipun tidak selalu menjamin solusi optimal.
11. *Ready Time* mengacu pada waktu ketika suatu pekerjaan sudah siap untuk

dimulai atau dijadwalkan dalam proses produksi.

2.5 Elemen Penjadwalan

Penjadwalan produksi melibatkan tiga elemen penting yang perlu diperhatikan untuk memastikan kelancaran dan efisiensi proses. Elemen-elemen tersebut adalah sebagai berikut (Utama, 2023):

1. *Job*, adalah sebuah pekerjaan yang diproses untuk menghasilkan suatu produk. Ada beberapa proses operasi yang harus dilakukan (minimal 1 operasi) untuk menjadikan *job* sebagai sebuah produk.
2. Operasi, adalah sebuah himpunan proses dari sebuah *job*. Sebuah operasi diurutkan menggunakan teknik tertentu untuk menyelesaikan sebuah *job*. Sebuah operasi bisa dikerjakan ketika operasi pendahulunya selesai dikerjakan lebih dulu. Urutan sebuah pekerjaan dan jenis mesin untuk setiap operasi diinformasikan dalam sebuah *matriks routing*.
3. Mesin, adalah sebuah *resource* (sumber daya) yang digunakan untuk menyelesaikan sebuah *job*. Sebuah mesin hanya bisa melakukan satu tugas pada periode waktu tertentu (Utama, 2023). Agar proses penyelesaian *job* berjalan dengan lancar, mesin harus memenuhi tiga kondisi berikut (Darmadi, 2019):
 - a) Setiap *job* mengikuti *routing* yang telah ditentukan.
 - b) Setiap mesin hanya dapat memproses satu *job* dalam satu waktu, dan pemrosesan tersebut tidak boleh dihentikan.
 - c) Waktu yang dibutuhkan untuk menyelesaikan setiap operasi telah ditentukan sebelumnya.

2.6 Hybrid Flowshop dengan Mesin Paralel

Hybrid flowshop (HFS), yang juga dikenal sebagai *flexible flowshop*, merupakan pengembangan dari model *flowshop* dan sistem mesin paralel, di mana suatu operasi dapat dilakukan oleh salah satu mesin paralel yang tersedia. Secara umum, HFS dapat diartikan sebagai sekumpulan pekerjaan (*job*) yang harus diproses di beberapa pusat kerja (*work center*), dengan setiap *work center* terdiri

dari sejumlah mesin paralel. Setiap *job* menjalani serangkaian operasi yang harus melewati work center dalam urutan yang telah ditentukan. Mesin-mesin dalam satu work center dianggap setara, sehingga *job* dapat diproses oleh mesin manapun di dalamnya (Pratiwi et al., 2018).

Praktik industri sering menghadapi masalah penjadwalan *Hybrid Flowshop* (HFS), terutama pada perusahaan yang menggunakan mesin ganda di setiap tahapan produksinya. HFS merupakan suatu sistem di mana terdapat n pekerjaan (*job*) yang harus diselesaikan melalui m tahapan proses (*stage*) secara berurutan, dengan tujuan untuk mengoptimalkan fungsi objektif tertentu. Pada *Hybrid Flowshop* (HFS), meskipun beberapa tahap hanya memiliki satu mesin, sistem ini dikategorikan sebagai HFS jika setidaknya ada satu tahap dengan lebih dari satu mesin paralel. Setiap *job* harus melewati seluruh tahapan secara berurutan, dengan satu mesin yang memproses *job* pada setiap tahap. Mesin-mesin di tiap tahap bisa identik, homogen, atau heterogen. Pengelolaan antrian antar tahapan dapat dilakukan dengan aturan seperti *First Come First Served* (FCFS) atau aturan penjadwalan lainnya sesuai kebutuhan. (Anwar et al., 2024).

2.7 Ant Colony Optimization (ACO)

Algoritma semut atau *Ant Colony Optimization* (ACO) merupakan algoritma pencarian berdasarkan probabilitas, di mana probabilitas yang digunakan merupakan probabilitas dengan bobot sehingga butir pencarian dengan bobot yang lebih besar akan berakibat memiliki kemungkinan terpilih lebih besar pula (Gandhi & Widyawati, 2019). *Ant Colony Optimization* (ACO) termasuk dalam kategori *Swarm Intelligence*, yang merupakan salah satu paradigma pengembangan untuk menyelesaikan masalah optimasi. Inspirasi untuk memecahkan masalah tersebut berasal dari perilaku kelompok atau kawanan serangga. ACO umumnya digunakan untuk menyelesaikan masalah optimasi diskrit dan permasalahan kompleks yang melibatkan banyak variabel (Karjono et al., 2016).

Perilaku semut dalam mencari dan menemukan sumber makanan dari sarangnya, kemudian kembali ke sarangnya, menjadi inspirasi bagi ditemukannya algoritma semut. Kemampuan semut untuk menemukan solusi yang baik dari berbagai solusi yang ada disebabkan oleh penggunaan informasi lokal berupa feromon yang ditinggalkan setiap kali semut melewati suatu jalur. Semakin banyak semut yang melewati jalur tersebut, semakin banyak feromon yang tertinggal, sehingga semut berikutnya akan lebih cenderung memilih jalur tersebut berdasarkan jumlah feromon yang ada (Gandhi & Widyawati, 2019). Jalur-jalur yang pendek akan mempunyai ketebalan feromon dengan probabilitas yang tinggi dan membuat jalur tersebut akan dipilih dan jalur yang panjang akan ditinggalkan. Jalur feromon membuat semut dapat menemukan jalan kembali ke sumber makanan atau sarang mereka. Berikut ini formula untuk pemilihan jalur sementara (deterministik) (Karjono et al., 2016):

$$v = \max \{ [\tau_{i,j}] \cdot [\eta_{i,j}^{\beta}] \} \dots \dots \dots (1)$$

Seekor semut akan berjalan dari simpul i menuju simpul j dengan probabilitas, Berikut ini merupakan formula probabilitas pemilihan jalur (Karjono et al., 2016):

$$P_{i,j} = \frac{(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})}{\sum (\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})} \dots \dots \dots (2)$$

Dimana:

$\tau_{i,j}$ = Jumlah *pheromone* pada sisi i,j

α = Parameter pengontrol pengaruh $\tau_{i,j}$

$\eta_{i,j}$ = Desirability sisi i,j (Biasanya $1/d_{i,j}$, dimana d adalah jarak)

β = Parameter pengontrol pengaruh $\eta_{i,j}$

Adapun formula yang digunakan untuk penambahan dan penguapan *pheromone* (Karjono et al., 2016):

$$\tau_{i,j} = (1-\rho)\tau_{i,j} + \Delta\tau_{i,j} \dots \dots \dots (3)$$

Dimana:

$\tau_{i,j}$ = Jumlah *pheromone* pada sisi i,j

ρ = Tingkat penguapan *pheromone*

$\Delta\tau_{i,j}$ = Jumlah *pheromone* dihasilkan

Berikut ini merupakan formula untuk update *pheromone* (Karjono et al., 2016):

$$\tau_{t,v} \leftarrow (1-\alpha) \cdot \tau_{t,v} + \alpha \cdot \Delta\tau_{t,v} \dots \dots \dots (4)$$

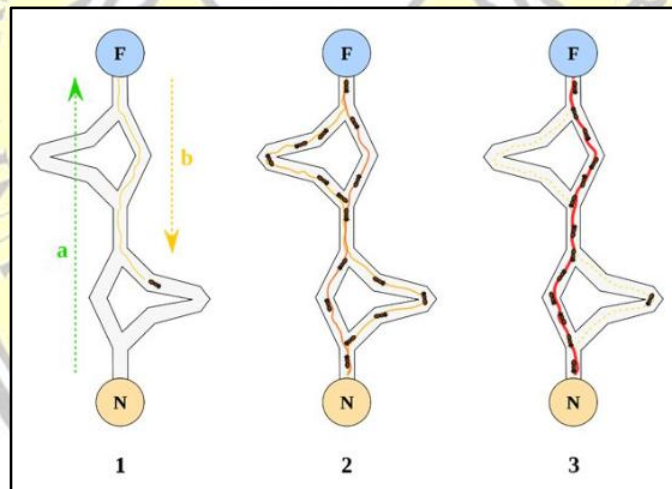
Dimana :

$\tau_{t,v}$ = Nilai *pheromone*, sebelum pembaruan

α = Tingkat pembaruan *pheromone*

$\Delta\tau_{t,v}$ = Jumlah *pheromone* yang ditambahkan

Berikut ini adalah prinsip dasar *Ant Colony Optomization* (ACO) (Ihsan et al., 2024):



Gambar 3. Prinsip Dasar ACO

(Sumber : Ihsan et al., 2024)

Proses ini akan dilakukan oleh semut-semut yang memperkuat jalur yang lebih pendek dengan menambah jumlah feromon, sehingga jalur terbaik akan semakin terekspos. Algoritma ACO memiliki beberapa kelebihan, yaitu (Ihsan et al., 2024):

1. Algoritma ACO menggunakan metode *backtracking* yang baik, sehingga

dapat untuk mencapai solusi terbaik

2. Algoritma ACO memiliki sistem kerjasama yang baik. Hal ini dapat ditunjukkan dengan efektivitas kerjasama antar koloni semut dan mencari solusi dengan solusi terbaik
3. Menggunakan struktur yang lebih luas dalam algoritma ACO, yang dapat membantu untuk menemukan solusi yang dapat diterima pada tahap proses
4. Algoritma ini dapat diterapkan pada versi yang sama untuk berbagai masalah optimasi kombinatorial

Adapun kelemahan dari algoritma ini adalah tingkat kerumitan cukup tinggi, *run time* nya juga cukup lama (Ihsan et al., 2024). Berikut adalah langkah-langkah kerja dari algoritma ini (Karjono et al., 2016):

- 1) Pada awalnya, semut berkeliling secara acak.
- 2) Ketika semut-semut menemukan jalur yang berbeda misalnya sampai pada persimpangan, mereka akan mulai menentukan arah jalan secara acak.
- 3) Sebagian semut memilih berjalan keatas dan sebagian lagi akan memilih berjalan kebawah.
- 4) Ketika menemukan makanan mereka kembali ke koloninya sambil memberikan tanda tangan dengan jejak feromon.
- 5) Karena jalur yang ditempuh lewat jalur bawah lebih pendek, maka semut yang bawah akan tiba lebih dulu dengan asumsi kecepatan semut-semut adalah sama
- 6) Feromon yang ditinggalkan oleh semut di jalur yang lebih pendek aromannya akan lebih kuat dibandingkan feromon di jalur yang lebih panjang.
- 7) Semut-semut lain akan lebih tertarik mengikuti jalur bawah karena aroma feromon lebih kuat.

2.8 Google Colab

Google Colab, singkatan dari *Google Colaboratory*, adalah layanan *cloud computing* yang disediakan oleh Google untuk mendukung pengembangan dan

penelitian ilmiah. *Colaboratory*, atau “Colab”, merupakan produk dari Google *Research* yang memungkinkan pengguna untuk menulis dan mengeksekusi kode *Python* secara langsung melalui browser. *Platform* ini sangat cocok digunakan dalam bidang *machine learning*, analisis data, dan pendidikan (Wilyani et al., 2024).

Google Colab berfungsi sebagai sebuah *Integrated Development Environment* (IDE) berbasis cloud untuk pemrograman *Python*, di mana proses komputasi dijalankan melalui server milik Google yang dilengkapi dengan perangkat keras berperforma tinggi. Dari sisi perangkat lunak, Google Colab telah menyediakan hampir semua pustaka (*library*) yang umum digunakan dalam berbagai bidang pengembangan dan riset (Guntara, 2023).

Python sendiri dikenal sebagai bahasa pemrograman tingkat tinggi yang populer karena sintaksisnya yang sederhana dan fleksibel. Bahasa ini dikembangkan oleh Guido van Rossum dan telah digunakan secara luas dalam berbagai keperluan, seperti pengembangan perangkat lunak, analisis data, kecerdasan buatan (*artificial intelligence*), dan pengembangan web. Kombinasi antara kemudahan penggunaan *Python* dan infrastruktur Google Colab menjadikannya alat yang ideal bagi pemula maupun profesional dalam berbagai bidang industri teknologi (Wilyani et al., 2024).

2.9 Metode Taguchi

Metode Taguchi merupakan pendekatan sistematis yang digunakan untuk peningkatan kualitas, dengan fokus utama pada optimalisasi rancangan produk dan proses. Tujuan dari metode ini adalah untuk menghasilkan produk atau proses yang tidak sensitif terhadap pengaruh variabel gangguan atau *noise*, sehingga dikenal dengan istilah *robust design*. Pendekatan ini memungkinkan sistem bekerja secara konsisten meskipun terdapat variasi dalam kondisi lingkungan atau proses (Halimah & Ekawati, 2020).

Metode ini digunakan dalam rekayasa dan perbaikan kualitas melalui perancangan eksperimen (*Design of Experiments/DoE*) guna mengidentifikasi faktor-

faktor yang paling dominan memengaruhi karakteristik kualitas dalam suatu proses. Dengan demikian, karakteristik tersebut dapat dikendalikan secara lebih efektif. Hasil dari penerapan metode ini adalah ditemukannya kombinasi optimal antara unit produk dan unit proses, yang mampu memberikan hasil yang seragam dan stabil, sekaligus menekan biaya produksi (Halimah & Ekawati, 2020).

Salah satu kontribusi terbesar dari Taguchi adalah pengembangan desain eksperimen menggunakan *orthogonal array*, yaitu bentuk modifikasi dari desain *fractional factorial*, yang mempermudah pelaksanaan eksperimen dalam jumlah lebih sedikit namun tetap informatif. Selain itu, metode ini juga memperkenalkan transformasi respon ke dalam bentuk rasio signal terhadap *noise* (*S/N ratio*), yang digunakan untuk mengevaluasi stabilitas serta performa sistem secara kuantitatif. Meskipun demikian, metode Taguchi juga memiliki kelemahan, khususnya terkait risiko pengambilan kesimpulan yang kurang akurat akibat jumlah eksperimen yang dipangkas. Oleh karena itu, dalam penerapannya, perlu dilakukan validasi dan analisis yang cermat untuk memastikan bahwa kesimpulan yang diambil tetap dapat diandalkan (Halimah & Ekawati, 2020). Taguchi membagi karakteristik kualitas menjadi 3 kategori, yaitu (Setiawan et al., 2018):

a. Perhitungan Rasio S/N

Rasio Sinyal terhadap *Noise* (S/N) digunakan untuk mengevaluasi pengaruh level faktor terhadap kualitas respon. Dalam penelitian ini digunakan dua karakteristik S/N, yaitu:

- 1) Nominal is *Best*, suatu produk dikatakan baik apabila pada karakteristik kualitas tertentu, nilainya mendekati nilai target yang telah ditentukan. Berikut ini rumus yang digunakan:

$$S/N = 10 \log \left(\frac{\bar{y}^2}{S^2} \right) \dots \dots \dots (5)$$

Dimana:

$\bar{y}^2$ = Kuadrat rata-rata respon

S^2 = Variansi respon

- 2) *Larger the better*, Suatu produk memiliki kualitas yang baik apabila memiliki nilai yang semakin tinggi pada karakteristik kualitas tertentu.

Berikut rumus yang digunakan:

$$S/N = -10 \log \left(\frac{1}{n} \sum_{i=1}^n \frac{1}{y_i^2} \right) \dots \dots \dots (6)$$

Dimana:

n = Jumlah ulangan

y_i = Nilai pengamatan ke- i

- 3) *Smaller is Better*, Suatu produk dikatakan berkualitas baik apabila pada karakteristik kualitas tertentu, memiliki nilai yang semakin rendah.

Adapun rumus yang digunakan, sebagai berikut:

$$S/N = -10 \log \left(\frac{1}{n} \sum_{i=1}^n y_i^2 \right) \dots \dots \dots (7)$$

Dimana:

n = Jumlah ulangan

y_i = Nilai pengamatan ke- i

