

LEMBAR PENGESAHAN

Proposal Proyek Akhir dengan judul :

OPTIMASI PENJADWALAN FLEXIBLE JOB-SHOP MENGGUNAKAN ALGORITMA GENETIKA

Optimization Of Flexible Job-Shop Scheduling Using Genetic Algorithm

oleh :

TODO TUA SITORUS

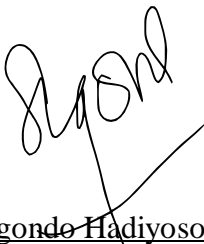
6705160108

Telah diperiksa dan disetujui untuk diajukan sebagai syarat mengambil
Mata Kuliah Proyek Akhir
pada Program Studi D3 Teknik Telekomunikasi Universitas Telkom

Bandung, 23 Oktober 2020

Menyetujui,

Pembimbing I



Sugondo Hadiyoso, S.T., M.T.

NIP. 13870076

Pembimbing II



Tita Harvanti, S.T., M.T.

NIP. 20950009



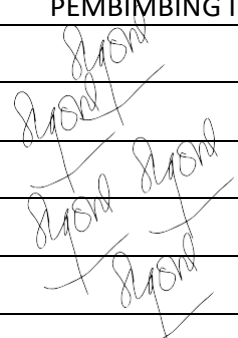
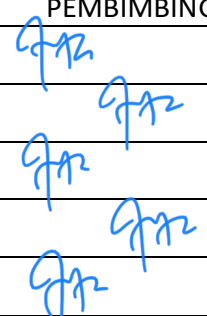
UNIVERSITAS TELKOM
FAKULTAS ILMU TERAPAN
KARTU KONSULTASI
SEMINAR PROPOSAL PROYEK AKHIR

NAMA / PRODI : Todo Tua Sitorus / D3TT NIM : 6705160108

JUDUL PROYEK AKHIR :
OPTIMASI PENJADWALAN FLEXIBLE JOB-SHOP MENGGUNAKAN ALGORITMA GENETIKA

CALON PEMBIMBING : I. Sugondo Hadiyoso, S.T., M.T.

II. Tita Haryanti, S.T., M.T.

NO	TANGGAL	CATATAN HASIL KONSULTASI	TANDA TANGAN CALON PEMBIMBING I
1	14/10	BAB 1 (SELESAI)	
2	16/10	BAB 2 (SELESAI)	
3	18/10	BAB 3 (SELESAI)	
4	20/10	BAB 4 (SELESAI)	
5	23/10	FINALISASI PROPOSAL	
6			
7			
8			
9			
10			
NO	TANGGAL	CATATAN HASIL KONSULTASI	TANDA TANGAN CALON PEMBIMBING II
1	14/10	BAB 1 (SELESAI)	
2	16/10	BAB 2 (SELESAI)	
3	18/10	BAB 3 (SELESAI)	
4	20/10	BAB 4 (SELESAI)	
5	23/10	FINALISASI PROPOSAL	
6			
7			
8			
9			
10			

**OPTIMASI PENJADWALAN FLEXIBLE JOB-SHOP
MENGUNAKAN ALGORITMA GENETIKA**

Optimization Of Flexible Job-Shop Scheduling Using Genetic Algorithm

PROPOSAL PROYEK AKHIR

Diajukan sebagai syarat untuk mengambil Mata Kuliah Proyek Akhir

oleh :

TODO TUA SITORUS

6705160108



**D3 TEKNIK TELEKOMUNIKASI
FAKULTAS ILMU TERAPAN
UNIVERSITAS TELKOM
2020**

LEMBAR PENGESAHAN

Proposal Proyek Akhir dengan judul :

**OPTIMASI PENJADWALAN FLEXIBLE JOB-SHOP MENGGUNAKAN
ALGORITMA GENETIKA**

Optimization Of Flexible Job-Shop Scheduling Using Genetic Algorithm

oleh :

TODO TUA SITORUS

6705160108

Telah diperiksa dan disetujui untuk diajukan sebagai syarat mengambil
Mata Kuliah Proyek Akhir
pada Program Studi D3 Teknik Telekomunikasi Universitas Telkom

Bandung, 23 Oktober 2020

Menyetujui,

Pembimbing I

Sugondo Hadiyoso, S.T., M.T.

NIP. 13870076

ABSTRAK

Permasalahan penjadwalan *Flexible Job-Shop* adalah kelanjutan dan perluasan dari permasalahan penjadwalan *Job-Shop* klasik. *Flexible Job-Shop* banyak diaplikasikan di industri, seperti pada manufaktur semikonduktor, perakitan mobil, dan tekstil, dimana sekelompok mesin digunakan untuk mengerjakan satu operasi. *Flexible Job-Shop* saat ini adalah model penjadwalan yang umum di lingkungan industri modern, namun seringkali karena tingkat kompleksitas yang tinggi perlu dilakukan optimasi agar mengurangi konflik di antara jalur produksi tersebut

Algoritma Genetika adalah salah satu algoritma optimasi yang banyak diaplikasikan pada berbagai permasalahan penelitian di dunia nyata, salah satunya adalah permasalahan penjadwalan produksi, dimana perlu dirumuskan tujuan objektif dengan berbagai kondisi batasan atau konflik antara *job*, operasi dan mesin yang harus dicari solusi dengan mencari nilai-nilai yang dianggap paling terbaik, termasuk dalam permasalahan penjadwalan *Flexible Job-Shop*.

Kata Kunci: *Job-Shop*, *Flexible Job-Shop*, Algoritma Genetika, Optimasi

DAFTAR ISI

LEMBAR PENGESAHAN.....	i
ABSTRAK	ii
DAFTAR ISI	iii
BAB I PENDAHULUAN	1
1.1 Latar Belakang.....	1
1.2 Tujuan dan Manfaat	2
1.3 Rumusan Masalah.....	2
1.4 Batasan Masalah	2
1.5 Metodologi	3
BAB II DASAR TEORI.....	5
2.1 Pengertian Penjadwalan	5
2.2 Jenis-jenis Aliran Produksi.....	5
2.3 Penjadwalan Flexible Job-Shop.....	6
2.4 Algoritma Genetik	7
2.4.1 Pengertian Algoritma Genetik.....	7
2.4.2 Struktur Umum Algoritma Genetik.....	8
2.4.3 Algoritma Genetika Komponen-komponen Utama.....	8
2.4.4 Istilah-istilah dalam algoritma genetik.....	10
2.4.5 Aplikasi Algoritma Genetik.....	11
2.4.6 Operator Genetik.....	12
2.4.7 Parameter Genetik	14
BAB III MODEL SISTEM	16
3.1 Asumsi	16
3.2 Parameter	17
3.3 Fungsi Objektif dan Kendala	18

BAB IV BENTUK KELUARAN YANG DIHARAPKAN	20
4.1 Keluaran yang Diharapkan.....	20
4.2 Jadwal Pelaksanaan.....	21
DAFTAR PUSTAKA	22

BAB I

PENDAHULUAN

1.1 Latar Belakang

Permasalahan penjadwalan *Flexible Job-Shop* adalah kelanjutan dan perluasan dari permasalahan penjadwalan *Job-Shop* klasik. Pada umumnya, di dalam model *Job-Shop* klasik, semua *job* mengikuti jalur produksi yang tetap, walau tidak harus sama dalam setiap *job*, dan setiap satu operasi dari setiap *job* harus diproses di satu mesin yang telah ditentukan. Penjadwalan *Job-Shop* klasik digunakan untuk menentukan urutan dari setiap operasi tersebut pada setiap satu mesin untuk mengoptimalkan satu atau beberapa kegiatan produksi. Namun sebaliknya, jika ada dua atau lebih mesin yang digunakan untuk memproses satu operasi, maka model tersebut dinamakan *Flexible Job-Shop*.

Flexible Job-Shop banyak diaplikasikan di industri, seperti pada manufaktur semikonduktor, perakitan mobil, dan tekstil, dimana sekelompok mesin digunakan untuk mengerjakan satu operasi. *Flexible Job-Shop* saat ini adalah model penjadwalan yang umum di lingkungan industri modern, namun seringkali karena tingkat kompleksitas yang tinggi perlu dilakukan optimasi agar mengurangi konflik di antara jalur produksi tersebut

Algoritma Genetika adalah salah satu algoritma optimasi yang banyak diaplikasikan pada berbagai permasalahan penelitian di dunia nyata, salah satunya adalah permasalahan penjadwalan produksi, dimana perlu dirumuskan tujuan objektif dengan berbagai kondisi batasan atau konflik antara *job*, operasi dan mesin yang harus dicari solusi dengan mencari nilai-nilai yang dianggap paling terbaik, termasuk dalam permasalahan penjadwalan *Flexible Job-Shop*. Dalam Algoritma Genetika ada beberapa tahap yang dilakukan yaitu proses *crossover*, proses *mutation*, dan terakhir digunakan proses *tournament selection* untuk mendapatkan probabilitas yang terbaik sebagai hasil dari optimasi.

1.2 Tujuan dan Manfaat

Adapun tujuan dari Proyek Akhir ini, sebagai berikut:

1. Merancang solusi permasalahan penjadwalan *Flexible Job-Shop* untuk jalur produksi industri manufaktur
2. Menerapkan algoritma genetika dalam mengatasi permasalahan penjadwalan *Flexible Job-Shop*
3. Mendapatkan hasil optimasi penjadwalan *Flexible Job-Shop*

1.3 Rumusan Masalah

Adapun rumusan masalah dari Proyek Akhir ini, sebagai berikut:

1. Bagaimana merancang solusi permasalahan penjadwalan *Flexible Job-Shop* untuk jalur produksi industri manufaktur?
2. Bagaimana penerapan algoritma genetika dalam mengatasi permasalahan penjadwalan *Flexible Job-Shop*?
3. Bagaimana mendapatkan hasil optimasi penjadwalan *Flexible Job-Shop*?

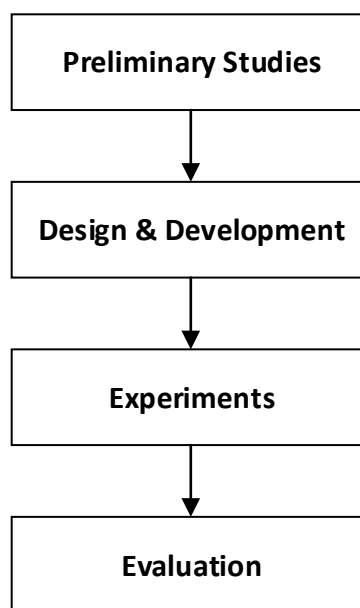
1.4 Batasan Masalah

Dalam Proyek Akhir ini, dilakukan pembatasan masalah sebagai berikut:

1. Contoh permasalahan yang digunakan pada penelitian dibatasi hanya pada permasalahan *Flexible Job-Shop* yang diajukan oleh Brandimarte dan Dauzere.
2. Tidak mempertimbangkan dari posisi atau tata letak mesin.
3. Tidak melibatkan waktu transportasi bahan dari satu mesin ke mesin lainnya dalam operasi.
4. Tidak melibatkan perhitungan sumber daya energi yang dibutuhkan oleh setiap mesin dalam operasionalnya.
5. Mesin-mesin yang dilibatkan dalam perhitungan dianggap selalu dalam kapasitas maksimal (tidak mempertimbangkan kondisi *fatigue*)
6. Bahan-bahan yang digunakan dalam operasi dalam setiap *job* dianggap selalu tersedia tepat waktu, keterlambatan (*lateness*) tidak dilibatkan dalam perhitungan

7. Tidak melibatkan perhitungan perubahan batas waktu produksi (*due date*) yang berubah di tengah proses produksi, *due date* dianggap selalu tidak ada perubahan dan tepat waktu.
8. Tidak mempertimbangkan adanya kemungkinan pengurangan waktu proses atau perlambatan (*tardiness*) di setiap operasi dari setiap *job* yang diakibatkan adanya faktor manusia atau hal lainnya yang memiliki dampak pada *tardiness*

1.5 Metodologi



Gambar 1 Metodologi Penelitian

Metodologi yang akan digunakan pada penelitian ini disesuaikan dengan gambar di atas dimana tahapannya adalah sebagai berikut :

1. Tahap *Preliminary Studies* (Studi Pendahuluan)

Studi pendahuluan atau biasa disebut studi literatur digunakan oleh peneliti untuk mengumpulkan dan mempelajari semua bahan referensi yang disesuaikan dengan topik penelitian yang didapat dari berbagai sumber, yang dianggap dapat mendukung metodologi yang akan digunakan.

2. Tahap *Design & Development* (Rancang Bangun)

Pada tahap ini, peneliti akan mengajukan rancangan solusi permasalahan penjadwalan untuk analisis dan pengukuran selama pemodelan dan simulasi dengan menggunakan algoritma genetika. Solusi yang diusulkan akan mempertimbangkan kembali, sesuai dengan hasil simulasi yang dilakukan setelah percobaan, sesuai dengan kontribusi penelitian yang ingin dicapai.

3. Tahap *Experiment* (Percobaan)

Pada tahap ini, peneliti akan menerapkan algoritma genetika untuk mencari hasil optimasi pada permasalahan penjadwalan *Flexible Job-Shop*, membuat model dan melakukan simulasi dengan bantuan perangkat lunak.

4. Tahap *Evaluation* (Evaluasi)

Pada tahap ini, peneliti akan membahas dan mendeskripsikan hasil evaluasi percobaan yang dikumpulkan dari setiap tahap percobaan sesuai batasan permasalahan, kemudian dilakukan pengujian dari setiap hasilnya.

BAB II

DASAR TEORI

2.1 Pengertian Penjadwalan

Pengertian penjadwalan adalah aktivitas perencanaan untuk menentukan kapan dan dimana setiap operasi sebagai bagian dari pekerjaan secara keseluruhan harus dilakukan pada sumber daya yang terbatas, serta pengalokasian sumber daya pada suatu waktu tertentu dengan memperhatikan kapasitas sumber daya yang ada.

Ada beberapa pengertian penjadwalan menurut beberapa ahli :

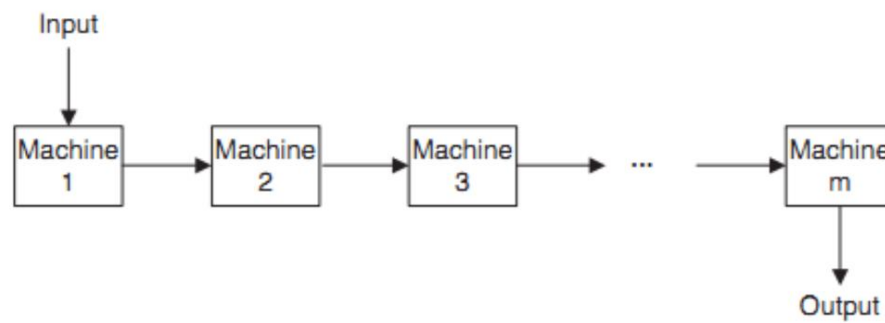
- Menurut Arifin & Rudyanto (2010), penjadwalan produksi adalah proses alokasi sumber daya dan mesin untuk menyelesaikan semua pekerjaan dengan mempertimbangkan batasan-batasan yang ada.
- Menurut Baker & Trietsch (2009), penjadwalan merupakan proses alokasi mesin-mesin yang ada untuk menjalankan tugas dalam jangka waktu tertentu.
- Menurut Pinedo (2016), penjadwalan adalah proses pengambilan keputusan yang digunakan untuk industri manufaktur dan jasa yang berhubungan dengan alokasi sumber daya untuk mengerjakan tugas dengan tujuan mengoptimalkan satu atau lebih tujuan.

Penjadwalan dibutuhkan untuk meminimasi distribusi tenaga kerja operator dan mesin agar lebih efektif. Hal ini sangat penting untuk pengambilan keputusan dalam proses produksi.

2.2 Jenis-jenis Aliran Produksi

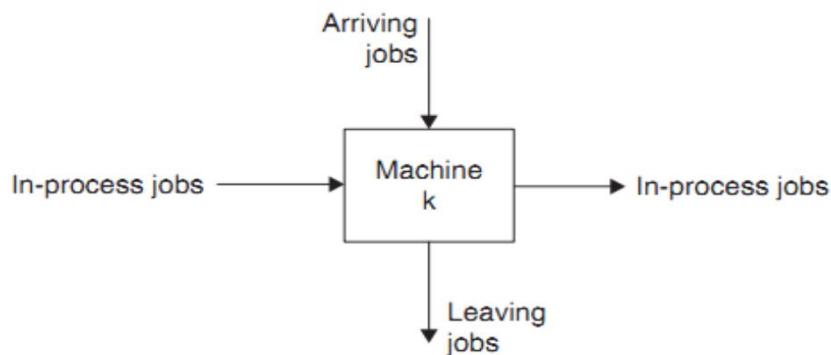
Jenis-jenis aliran produksi yang dimiliki oleh perusahaan menurut Baker & Trietsch (2009) yaitu :

1. Aliran Flow-shop, merupakan rantai produksi yang memproses suatu produk dengan urutan proses yang sama terhadap semua komponen produk mulai dari bahan mentah hingga menjadi barang jadi. Sehingga jika suatu produk telah selesai diproses pada suatu mesin dan sedang dalam proses pengerjaan pada mesin lainnya, produk tersebut tidak dapat diproses kembali pada mesin sebelumnya.



Gambar 2.1 Pola Aliran Flow-Shop

2. Aliran *Job-Shop*, dimana setiap order dapat melalui urutan proses yang berbeda-beda dengan mesin yang berbeda pula. Karena setiap order memiliki urutan proses dan mesin yang berbeda maka memungkinkan untuk masing-masing stasiun kerja memproses beberapa item yang berbeda juga. Dengan artian beberapa pekerjaan dapat diproses beberapa kali di mesin yang sama.



Gambar 2.2 Pola Aliran Job-Shop

2.3 Penjadwalan Flexible Job-Shop

Menurut Huang (2018), Penjadwalan *Flexible Job-Shop* adalah kelanjutan dan perluasan dari penjadwalan *Job-Shop* klasik. Pada umumnya, di dalam model *Job-Shop* klasik, semua *job* mengikuti jalur produksi yang tetap, walau tidak harus sama dalam setiap *job*, dan setiap satu operasi dari setiap *job* harus diproses di satu mesin yang telah ditentukan. Penjadwalan *Job-Shop* klasik digunakan untuk menentukan urutan dari setiap

operasi tersebut pada setiap satu mesin untuk mengoptimalkan satu atau beberapa kegiatan produksi. Namun sebaliknya, jika ada dua atau lebih mesin yang digunakan untuk memproses satu operasi, maka model tersebut dinamakan *Flexible Job-Shop*.

Flexible Job-Shop digunakan tidak hanya untuk menentukan urutan operasi pada mesin, tetapi juga menetapkan setiap operasi ke mesin, menurut Huang (2018) algoritma *Flexible Job-Shop* dapat dikategorikan ke dalam algoritma *polynomial non-deterministic / non-polynomial NP-hard*. Algoritma ini akan menjadi semakin rumit karena adanya kompleksitas penjadwalan pada setiap mesin.

Algoritma *non-polynomial* dibagi dua kelas yaitu *NP-hard* dan *NP-complete*. Suatu permasalahan dalam *NP-complete* bersifat dapat dipecahkan dalam waktu *polynomial* jika dan hanya jika seluruh permasalahan *NP-complete* juga dapat dipecahkan dalam waktu *polynomial*. Jika sebuah permasalahan *NP-hard* dapat dipecahkan dalam waktu *polynomial*, maka seluruh permasalahan *NP-complete* dapat dipecahkan dalam waktu *polynomial*. Seluruh permasalahan *NP-complete* merupakan permasalahan *NP-hard* belum tentu menjadi permasalahan *NP-complete* (Horowitz, 2008)

2.4 Algoritma Genetik

Algoritma genetik merupakan evolusi atau perkembangan dunia komputer dalam bidang kecerdasan buatan (*artificial intelligent*), yang sebenarnya terinspirasi oleh teori Darwin dimana individu yang kuat adalah pemenang dalam kompetisi dilingkungannya.

2.4.1 Pengertian Algoritma Genetik

Algoritma Genetika adalah algoritma pencarian heuristik yang didasarkan atas mekanisme evolusi biologis. Keberagaman pada evolusi biologis adalah variasi dari kromosom antar individu organisme. Variasi kromosom ini akan mempengaruhi laju reproduksi dan tingkat kemampuan organisme untuk tetap hidup. pada dasarnya ada 4 kondisi yang sangat mempengaruhi proses evaluasi yaitu :

1. kemampuan organisme untuk melakukan reproduksi
2. keberadaan populasi organisme yang bisa melakukan reproduksi
3. keberagaman organisme dalam suatu populasi
4. perbedaan kemampuan untuk *survive*

Individu yang lebih kuat (fit) akan memiliki tingkat survival dan tingkat reproduksi yang lebih tinggi jika dibandingkan dengan individu yang kurang fit. Pada kurun waktu tertentu (sering dikenal dengan istilah generasi), populasi secara keseluruhan akan lebih banyak memuat organisme yang fit.

Algoritma Genetika pertama kali dikembangkan oleh John Holland dari universitas Michigan (1975). John Holland mengatakan bahwa setiap masalah yang berbentuk adaptasi (alami maupun buatan) dapat diformulasikan dalam terminologi genetika. Algoritma genetika adalah simulasi dari proses evolusi Darwin dan operasi genetika atas kromosom.

2.4.2 Stuktur Umum Algoritma Genetik

Pada algoritma ini teknik pencarian dilakukan sekaligus atas sejumlah solusi yang mungkin yang dikenal dengan **populasi**. Individu yang terdapat dalam satu populasi disebut istilah **kromosom**. Kromosom ini merupakan suatu solusi yang masih berbentuk simbol. Populasi awal dibangun secara acak, sedangkan populasi berikutnya merupakan hasil evolusi kromosom-kromosom melalui iterasi yang disebut dengan istilah **generasi**. Pada setiap generasi, kromosom akan melalui proses evaluasi dengan menggunakan alat ukur yang disebut dengan fungsi **fitness**. Nilai fitness dari suatu kromosom akan menunjukkan kualitas kromosom dalam populasi tersebut. Generasi berikut dikenal dengan istilah anak (*offspring*) terbentuk dari gabungan 2 kromosom generasi sekarang yang bertindak sebagai induk (**parent**) dengan menggunakan operator penyilangan (**crossover**). Selain operator penyilangan, suatu kromosom juga dapat dimodifikasi dengan menggunakan operator mutasi.

Populasi generasi yang baru dibentuk dengan cara menyeleksi nilai fitness dari kromosom induk (parent) dan nilai fitness dari kromosom anak (offspring), serta menolak kromosom-kromosom yang lainnya sehingga ukuran populasi (jumlah kromosom dalam suatu populasi) konstan. Setelah melalui beberapa generasi, maka algoritma ini akan konvergen ke kromosom terbaik.

2.4.3 Algoritma Genetika Komponen-komponen Utama

Ada 6 komponen utama dalam algoritma genetika, yaitu :

1. Teknik Penyandian
Teknik penyandian disini meliputi penyadian gen dari kromosom. Gen merupakan bagian dari kromosom. Satu gen biasanya akan mewakili satu variabel. Gen dapat dipresentasikan dalam bentuk: string bit, pohon, array bilangan real, daftar aturan, elemen permutasi, elemen program, atau representasi lainnya yang dapat diimplementasikan untuk operator genetika.
2. Prosedur Inisialisasi
Ukuran populasi tergantung pada masalah yang akan dipecahkan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian harus inisialisai terhadap kromosom yang terdapat pada populasi tersebut. Inisialisasi kromosom dilakukan secara acak, namun demikian harus tetap memperhatikan domain solusi dan kendala permasalahan yang ada.
3. Fungsi Evaluasi
Ada 2 hal yang harus dilakukan dalam melakukan evaluasi kromosom, yaitu: evaluasi fungsi objektive (fungsi tujuan) dan konversi fungsi objektive kedalam fungsi fitness. Secara umum, fungsi fitness diturunkan dari fungsi objektive dengan nilai tidak negatif. Apabila ternyata fungsi objektive memiliki nilai negatif, maka perlu ditambahkan suatu konstanta C agar nilai fitness yang terbentuk menjadi tidak negatif.
4. Seleksi
Seleksi ini bertujuan untuk memberikan kesempatan reproduksi yang lebih besar bagi anggota populasi yang paling fit.
5. Operator Genetika
Ada 2 operator genetika, yaitu:
 - a. Operator untuk melakukan rekombinasi, yang terdiri dari: rekombinasi bernilai biner (*crossover*).
 - b. Mutasi. mutasi bernilai biner
6. Penentuan parameter
Yang disebut parameter disini adalah parameter kontrol algoritma genetika, yaitu: ukuran populasi (*popsiz*e), peluang *crossover* (P_c), dan peluang mutasi (P_m). Nilai parameter ini ditentukan juga berdasarkan permasalahan yang akan dipecahkan. Ada beberapa rekomendasi yang bisa digunakan, antara lain:

a) Untuk permasalahan yang memiliki solusi cukup besar, De Jong merekomendasikan untuk nilai parameter kontrol:

$$(\text{popsize}; P_c; P_m) = (50; 0,6; 0,001)$$

b) Bila rata-rata fitness setiap generasi digunakan sebagai indikator, maka Grefenstette merekomendasikan:

$$(\text{popsize}; P_c; P_m) = (30; 0,95; 0,01)$$

c) Bila fitness dari individu terbaik dipantau pada setiap generasi, maka usulannya adalah:

$$(\text{popsize}; P_c; P_m) = (80; 0,45; 0,01)$$

Ukuran populasi sebaiknya tidak lebih kecil dari 30, untuk sembarang jenis permasalahan.

2.4.4 Istilah-istilah dalam algoritma genetik

Algoritma genetik merupakan algoritma pencarian yang bekerja berdasarkan mekanisme seleksi alam dan genetika. Pada genetika, kromosom terdiri dari gen-gen. Tiap gen mempunyai sifat tertentu (*allele*), dan posisi tertentu (*locus*). Satu atau lebih kromosom bergabung membentuk paket genetik yang disebut *genotif*. Interaksi *genotif* dengan lingkungannya disebut *fenotif*. Pada algoritma genetik, kromosom berpadanan dengan string dan gen dengan karakter. Setiap karakter mempunyai posisi (*locus*) dan arti tertentu (*allele*). Satu atau lebih string bergabung membentuk struktur (*genotif*), dan apabila struktur tersebut di-*decode*-kan akan diperoleh salah satu alternative solusi (*fenotif*).

Tabel 2.1 Tabel Susunan Gen Genetik

Ilmu genetik	Algoritma genetik
Kromosom	String
Gen	Karakter
Allele	Nilai karakter
Locus	Posisi dalam individu
Genotif	Struktur
Fenotif	parameter

2.4.5 Aplikasi Algoritma Genetik

Sejak dirintis oleh John Holland, Algoritma Genetik telah dipejari, diteliti dan diaplikasikan secara luas pada berbagai bidang. Algoritma Genetik banyak digunakan pada masalah praktis yang berfokus pada pencarian parameter-parameter optimal. Hal ini membuat banyak orang mengira bahwa Algoritma Genetik hanya bisa digunakan untuk masalah optimasi. Pada kenyataannya, Algoritma Genetik juga memiliki performansi yang bagus untuk masalah-masalah selain optimasi.

Keuntungan penggunaan Algoritma Genetik terlihat dari kemudahan implementasi dan kemampuan untuk menemukan solusi yang “bagus” (bisa diterima) secara cepat untuk masalah-masalah berdimensi tinggi. Algoritma Genetik sangat berguna dan efisien untuk masalah dengan karakteristik sebagai berikut :

- a. Ruang masalah sangat besar, kompleks, dan sulit dipahami.
- b. Kurang atau bahkan tidak ada pengetahuan yang memadai untuk merepresentasikan masalah ke dalam ruang pencarian yang lebih sempit.
- c. Ketika metode-metode konvensional sudah tidak mampu menyelesaikan masalah yang dihadapi.
- d. Solusi yang diharapkan tidak harus paling optimal, tetapi cukup “bagus” atau bisa diterima.
- e. Terdapat batasan waktu, misalnya dalam *real time system* atau sistem waktu nyata.

Algoritma Genetik telah banyak diaplikasikan untuk menyelesaikan masalah dan pemodelan dalam bidang teknologi, bisnis, dan entertainment, seperti :

a) Optimasi

Algoritma Genetik digunakan untuk optimasi numerik dan optimasi kombinatorial seperti Travelling Salesman Problem (TSP), perancangan IC, Job Shop Scheduling, optimasi suara dan video.

b) Pemrograman Otomatis

Algoritma genetik digunakan untuk melakukan proses evolusi terhadap program komputer untuk merancang struktur komputasional, seperti cellular automata dan sorting networks

c) Machine Learning

Algoritma genetik berhasil diaplikasikan untuk memprediksi struktur protein. Algoritma genetik juga berhasil dalam perancangan neural networks (jaringan syaraf tiruan).

d) Model Ekonomi

Algoritma genetik digunakan untuk memodelkan sistem-sistem inovasi, memprediksi produksi dan laba perusahaan.

2.4.6 Operator Genetik

Operator genetik digunakan untuk mengkombinasikan (modifikasi) individu dalam aliran populasi untuk menghasilkan individu pada generasi berikutnya. Ada tiga operator genetik yaitu, *crossover* dan mutasi.

2.4.6.1 Crossover

Pada *crossover* akan dipilih secara acak dua individu dan tempat pertukaran, dimana kromosom yang ditandai diantara kedua tempat pertukaran akan bertukar tempat satu sama lain. Proses *crossover* akan membangkitkan offspring baru dengan mengganti sebagian informasi dari parents (orang tua atau induk).

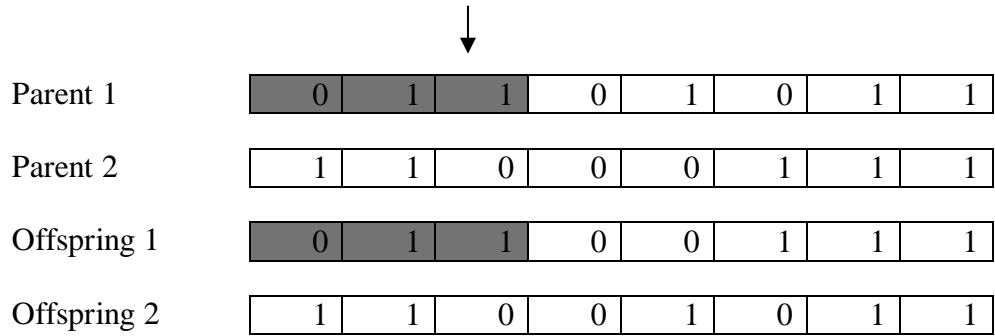
Tujuan dari proses *crossover* adalah untuk menambahkan keanekaragaman individu dalam populasi dengan mengawinkan individu-individu pada populasi sehingga menghasilkan keturunan berupa individu-individu baru untuk ditempatkan pada populasi selanjutnya. Ada beberapa tipe *crossover*, antara lain :

1. *One-cut-point-crossover*

Pada tipe ini, akan dibuat satu titik *crossover* dimana individu yang dihasilkan akan diambil dari bilangan biner parent pertama dari awal sampai titik *crossover* dan sisanya dari parent kedua. Algoritmanya adalah :

- a. Memilih site secara random dari parent pertama
- b. Isi disebelah kanan site pada parent pertama ditukar dengan parent kedua untuk menghasilkan offspring.

Contoh :

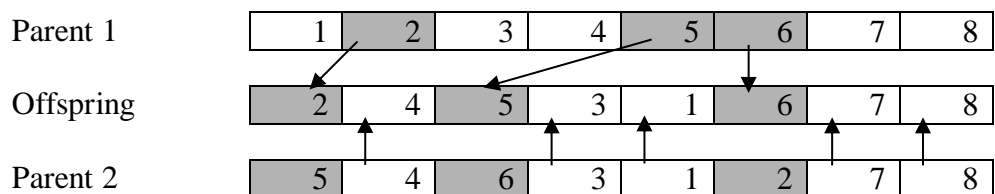


Gambar 2.3 Contoh Single Point Crossover

2. Order-based crossover

Pada tipe ini, offspring yang dihasilkan hanya satu hasil dari kombinasi kedua parent.

Contoh :



Gambar 2.4 Contoh Order Based Crossover

2.4.6.2 Mutasi

Mutasi menciptakan individu baru dengan melakukan modifikasi satu atau lebih gen dalam individu yang sama. Mutasi berfungsi untuk menggantikan gen yang hilang dari populasi selama proses seleksi serta menyediakan gen yang tidak ada dalam populasi awal. Sehingga mutasi akan meningkatkan variasi populasi. *Shif mutation* dilakukan cara :

- Menentukan dua site secara random
- Site pertama ditempatkan ke site kedua, untuk selanjutnya digeser ke kiri seperti terlihat pada gambar berikut.

Parent	1	2	3	4	5	6	7	8
Offspring	1	2	7	4	5	6	3	8

Gambar 2.5 Contoh Shift Mutation

2.4.7 Parameter Genetik

Parameter-parameter genetik berguna dalam pengendalian operator-operator genetik. Pemilihan penggunaan nilai-nilai parameter genetik sangat berpengaruh terhadap kinerja algoritma genetik dalam menyelesaikan suatu masalah. Parameter-parameter genetik yang digunakan antara lain :

1. Ukuran populasi

Ukuran populasi mempengaruhi ukuran efektivitas dan kinerja algoritma genetik. Tidak ada aturan yang pasti tentang berapa nilai ukuran populasi. Apabila ukuran populasi kecil berarti hanya tersedia sedikit pilihan untuk *crossover* dan sebagian kecil dari domain solusi saja yang dieksplorasi untuk setiap generasinya. Sedangkan apabila terlalu besar, kinerja algoritma genetik akan menurun. Penelitian menunjukkan ukuran populasi besar tidak mempercepat pencarian solusi. Disarankan ukuran populasi berkisar antara 20 – 30.

2. Jumlah generasi

Jumlah generasi berpengaruh terhadap banyaknya iterasi yang akan dikerjakan dan domain solusi yang akan dieksplorasi untuk setiap generasinya. Semakin besar jumlah generasi berarti semakin banyak iterasi yang dilakukan, dan semakin besar solusi yang dieksplorasi. Sedangkan semakin kecil jumlah generasinya, maka akan semakin kecil iterasi yang akan dilakukan, dan semakin kecil pula solusi yang dieksplorasi untuk tiap generasinya.

3. Probabilitas *crossover* (P_c)

Probabilitas *crossover* akan mengendalikan operator *crossover* dalam setiap generasi dalam populasi yang mengalami *crossover*. Semakin besar nilai probabilitas *crossover*, akan semakin cepat struktur individu baru terbentuk ke dalam populasi. Sedangkan apabila nilai probabilitas *crossover* terlalu besar, individu yang merupakan kandidat solusi terbaik mungkin akan dapat hilang lebih cepat pada generasi selanjutnya. Disarankan nilai probabilitas *crossover* berkisar antara 80 % - 95 %.

4. Probabilitas mutasi (P_m)

Probabilitas mutasi akan mengendalikan operator mutasi pada setiap generasi. Peluang mutasi yang digunakan biasanya lebih kecil daripada peluang *crossover*. Pada seleksi alam murni, mutasi jarang sekali muncul. Oleh karena itu, operator

mutasi pada algoritma genetik juga tidak selalu terjadi. Untuk itulah nilai peluang mutasi dibuat lebih kecil untuk setiap generasi. Disarankan nilai probabilitas mutasi kecil berkisar antara 0.5 % - 1 %.

BAB III

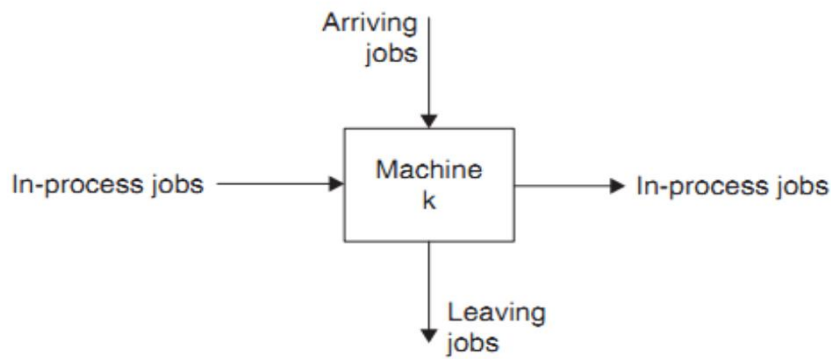
MODEL SISTEM

Dalam model sistem penjadwalan *Flexible Job-Shop* dalam penelitian ini ada satu set independen *job* yang terdiri dari $\{J_1, J_2, \dots, J_n\}$ yang harus diproses pada satu set mesin $\{M_1, M_2, \dots, M_n\}$. Dalam setiap *job* dapat terdiri dari satu atau lebih operasi, dan setiap *job* tersebut bisa saja berbeda dalam setiap jalur produksi, kemudian yang menjadi acuan utama sekaligus meningkatkan kompleksitas penjadwalan, yaitu pada *Flexible Job-Shop*, setiap operasi dapat ditangani atau diproses oleh satu atau lebih mesin, dan waktu pemrosesan setiap operasi tersebut sangat tergantung pada setiap mesin yang bekerja untuk menyelesaikan operasi tersebut.

3.1 Asumsi

Beberapa asumsi yang akan digunakan untuk merancang sistem penjadwalan *Flexible Job-Shop* dalam penelitian ini adalah sebagai berikut :

1. Setiap *job* tidak bergantung satu sama lain di antaranya, dan semuanya dilakukan secara terpisah dimulai dari awal mula *job* tersebut mulai diproses.
2. Setiap *job* memiliki prioritas sama untuk mulai diproses
3. Interupsi pada setiap *job* tidak diperbolehkan atau dianggap tidak akan pernah ada
4. Untuk setiap operasi, pasti akan tersedia satu atau lebih mesin yang mampu memprosesnya, tidak ada operasi yang terbengkalai atau tidak dapat diproses
5. Sebuah operasi tidak diperbolehkan diproses jika proses operasi sebelumnya belum selesai dilakukan, walaupun ada mesin yang menganggur atau diam
6. Setiap mesin dapat memproses paling banyak satu operasi pada setiap saat, satu mesin tidak dapat memproses lebih dari satu operasi.



Gambar 3.1 Model Sistem Penjadwalan Flexible Job-Shop

3.2 Parameter

Beberapa parameter yang digunakan dalam perancangan sistem, adalah sebagai berikut:

m = total jumlah mesin di lantai produksi

M = satu set mesin

M_i = mesin ke- i ($i = 1, \dots, m$)

n = total jumlah *job*

J = satu set *job*

J_j = *job* ke- j ($j = 1, \dots, n$)

h_j = jumlah operasi asosiasi dengan *job* (J_j) ke- j ($j = 1, \dots, n$)

O_{jh} = operasi ke- h asosiasi dengan *job* (J_j) ($j = 1, \dots, n$, and $h = 1, \dots, h_j$)

Ω_{jh} = sub dari 1 set mesin yang dibutuhkan untuk memproses operasi (O_{jh})

m_{jh} = jumlah mesin yang dibutuhkan untuk memproses operasi (O_{jh})

P_{ijh} = waktu yang dibutuhkan untuk memproses operasi (O_{jh}) oleh mesin (M_i)

S_{jh} = waktu mulai operasi (O_{jh}) atau waktu rilis

C_{jh} = waktu penyelesaian operasi (O_{jh})

d_j = tanggal jatuh tempo (*due date*) *job* (J_j)

C_j = waktu penyelesaian *job* (J_j)

C_{max} = waktu penyelesaian maksimum untuk semua *job* (*makespan*)

H = bilangan bulan positif besar

T_0 = total jumlah semua operasi

Variabel keputusan yang digunakan dalam perancangan sistem, adalah sebagai berikut :

$$x_{ijh} = \begin{cases} = 1 \\ = 0 \end{cases} \frac{\text{jika operasi } O_{jh} \text{ diproses oleh mesin } M_i}{\text{lainnya}}$$

$$x_{ijh} = \begin{cases} = 1 \\ = 0 \end{cases} \frac{\text{jika operasi } O_{jh} \text{ mendahului } O_{kl} \text{ diproses oleh mesin } M_i}{\text{lainnya}}$$

3.3 Fungsi Objektif dan Kendala

Asumsi-asumsi perlu dilakukan dalam perancangan sistem yang dibuat, asumsi-asumsi itu terdiri dari fungsi objektif dan batasan-batasan / kendala (*constraints*). Fungsi objektif yang digunakan dalam penelitian ini bertujuan untuk meminimasi *makespan* (C_{max}), *bottleneck* beban kerja mesin (W_m), dan total beban kerja mesin (W_t), yaitu yang dirumuskan sebagai berikut :

$$C_{max} = \min \left(\max_{1 \leq j \leq n} (C_j) \right) \quad (1)$$

$$W_m = \min \left(\max_{1 \leq i \leq m} \sum_{j=1}^n \sum_{h=1}^{h_j} P_{ijh} x_{ijh} \right) \quad (2)$$

$$W_t = \min \left(\sum_{i=1}^m \sum_{j=1}^n \sum_{h=1}^{h_j} P_{ijh} x_{ijh} \right) \quad (3)$$

Selanjutnya adalah *constraint* (batasan / kendala) yang digunakan dalam penelitian ini sesuai dengan fungsi objektif yang digunakan di atas, adalah sebagai berikut :

$$c_{jh} - s_{jh} - \sum_{\{i: O_{jh} \in \Omega_{jh}\}} (P_{ijh} x_{ijh}) = 0 \quad \forall j, h = 1, \dots, h_j \quad (4)$$

$$\sum_{i=1}^{m_{jh}} x_{ijh} = 1 \quad j = 1, \dots, n; h = 1, \dots, h_j \quad (5)$$

$$c_{jh} \leq s_{j(h+1)} \quad j = 1, \dots, n; h = 1, \dots, h_j - 1 \quad (6)$$

$$c_{jh} - c_{kl} + H y_{ijkhl} + H(1 - x_{ijh}) + H(1 - x_{ikl}) \geq P_{ijh} \quad (7)$$

$$\forall i, (j, h), (k, l): O_{jh} \in \Omega_{jh}, O_{kl} \in \Omega_{kl}$$

$$s_{jh} + x_{ijh} P_{ijh} \leq c_{jh} \quad i = 1, \dots, m; j = 1, \dots, n; h = 1, \dots, h_j \quad (8)$$

$$c_{jh_j} \leq C_{max} \quad j = 1, \dots, n \quad (9)$$

Constraint (4) menunjukkan bahwa perbedaan antara waktu penyelesaian dan waktu mulai pengoperasian O_{jh} di mesin M_i sama dengan waktu pemrosesannya di mesin M_i , artinya operasi tersebut harus diselesaikan tanpa gangguan sama sekali sebelum operasi lain dimulai, sesuai dengan asumsi (3). *Constraint* (5) sesuai dengan asumsi (4), menyajikan fakta bahwa operasi harus dilakukan ditetapkan hanya ke satu mesin di antara sub set dari mesin alternatif pada suatu waktu. Untuk menjamin asumsi (5), *constraint* (6) dan (7) menangani secara seksama urutan operasi yang telah ditentukan terkait dengan setiap pekerjaan. *Constraint* (8) memastikan asumsi (6), yaitu masing-masing mesin hanya dapat melakukan paling banyak satu operasi pada a waktu. *Constraint* (9) menjelaskan bahwa waktu penyelesaian setiap operasi tidak boleh lebih dari C_{max} . *Constraint* (10) menentukan domain variabel keputusan.

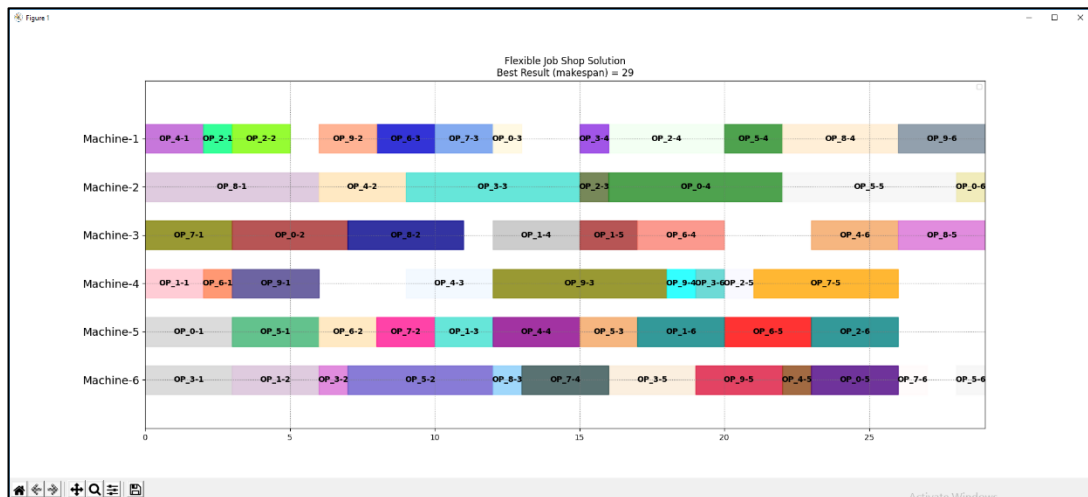
BAB IV

BENTUK KELUARAN YANG DIHARAPKAN

4.1 Keluaran yang Diharapkan

Bentuk keluaran yang diharapkan dari model sistem dalam penelitian ini adalah hasil penjadwalan *Flexible Job-Shop* yang memberikan hasil optimasi terbaik menurut waktu proses (*makespan*), jumlah Generasi Genetik, dan waktu yang dibutuhkan oleh Algoritma Genetik untuk mendapatkan hasil

Kemudian hasil penjadwalan tersebut juga harus dapat menggambarkan urutan pekerjaan di setiap mesin sesuai setiap operasi yang telah dijadwalkan yang ditampilkan keluarannya dalam bentuk Gantt Chart seperti contoh gambar di bawah ini.



Gambar 4.1 Contoh Keluaran dalam Bentuk Gantt Chart

Gantt Chart di atas menggambarkan hasil penjadwalan *Flexible Job-Shop* yang telah dilakukan optimasi menggunakan Algoritma Genetik sesuai dengan model sistem yang telah dirancang dalam penelitian ini

4.2 Jadwal Pelaksanaan

Jadwal kegiatan dibuat untuk mengetahui kegiatan yang dilakukan dengan merinci setiap tahap dalam prosedur penelitian yang akan dilakukan pada waktu yang ditentukan. Waktu penelitian tersebut dibuat dalam bentuk jadwal kegiatan sebagai berikut :

Tabel 1.1 Jadwal Kegiatan

No	Kegiatan	Bulan (Tahun 2020 - 2021)				
		Nov	Des	Jan	Feb	Mar
1	Studi Literatur					
2	Realisasi					
3	Pengujian					
4	Analisis					
5	Pembuatan Laporan					

DAFTAR PUSTAKA

- Abd Elazeem, M., Ali Osman, M., & Ali Hassan, M. (2011). Optimality of the flexible job shop scheduling problem. *African Journal of Mathematics and Computer Science Research*, 4(10)(Academic Journals), 321-328.
- Agarwal, A. K., & Kumar, D. (2020). JOB SHOP AND FLEXIBLE JOB SHOP SCHEDULING PROBLEMS:. *INTERNATIONAL JOURNAL OF INNOVATIONS IN ENGINEERING RESEARCH AND TECHNOLOGY [IJIERT]*, 7(5).
- Habibi, Y., Swalaganata, G., & Yustita, A. D. (2017). PENYELESAIAN MULTI-OBJECTIVE FLEXIBLE JOB SHOP SCHEDULING PROBLEM MENGGUNAKAN HYBRID ALGORITMA IMUN. *Jurnal Teknosains*, 6, No. 2(Pascasarjana Universitas Gadjah Mada).
- Huang, X., Guan, Z., & Yang, L. (2018). An effective hybrid algorithm for multi-objective flexible job-shop scheduling problem. *Advances in Mechanical Engineering*, 10(9), 1-14.
- Li, X., Peng, K., Gao, L., Xie, J., & Li, H. (2019). Review on flexible job shop scheduling. *IET Collaborative Intelligent Manufacturing*, 1(2).
- Pezzella, F., Morganti, G., & Ciaschetti, G. (2018). A genetic algorithm for the Flexible Job-shop Scheduling Problem. *Computers & Operations Research*, 35, 3202–3212.