Penerapan Simulated Annealing Untuk Memperbaiki Konvergensi Prematur Pada Algoritma Genetika Dalam Penjadwalan Produksi

TUGAS AKHIR

Sebagai syarat untuk memperoleh gelar sarjana S-1 di Program Studi Informatika, Jurusan Teknik Informatika, Fakultas Teknik Industri, Universitas Pembangunan Nasional "Veteran" Yogyakarta

Disusun oleh:

Archan Habib Sya'bana

123170024

PROGRAM STUDI INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK INDUSTRI
UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"
YOGYAKARTA
2021

Daftar Isi

Daftar Isi ..... **ii**
Daftar Gambar ..... **iv**
Daftar Tabel ..... **v**
BAB I **1**
1.1. Latar Belakang 1
1.1. Rumusan Masalah 3
1.2. Batasan Masalah 3
1.3. Tujuan Penelitian 3
1.4. Manfaat Penelitian 4
1.5. Tahapan Penelitian 4
1.6. Sistematika Penulisan 5
BAB II **6**
2.1. Tinjauan Studi 6
2.2. Tinjauan Pustaka 17
2.2.1. Penjadwalan 17
2.2.2. Jenis Aliran Proses Produksi 18
2.2.3. Ukuran performansi Penjadwalan 20
2.2.4. Penjadwalan *Flowshop* 20
2.2.5. Perhitungan Fungsi Tujuan 21
2.2.6. *Gantt Chart* 21
2.2.7. Algoritma Genetika 22
2.2.8. Komponen Utama Algoritma Genetika 24
2.2.9. *Simulated Annealing* 27
2.2.10. Proses *Simulated Annealing* 29
BAB III **31**
3.1. Metode Penelitian 31
3.2. Pengumpulan Data 31
3.3. Analisis Kebutuhan 33
3.3.1. Kebutuhan Fungsional 34
3.3.2. Kebutuhan Nonfungsional 34
3.4. Pembuatan Model 35
3.4.1. *Hybrid* Algoritma Genetika dan *Simulated Annealing* 35
3.4.3. Proses *Simulated Annealing* 42
3.5. Proses Desain 44

3.5.1. Perancangan Sistem 45
3.5.2. Perancangan Proses 46
3.5.3. Perancangan Antarmuka 48
3.5.4. Jadwal Penelitian 50
Daftar Pustaka 51

Daftar Gambar

Gambar 2.1 Pola aliran pure flowshop **19**
Gambar 2.2 Pola aliran general flowshop **19**
Gambar 2.3 Pola aliran Jobshop **20**
Gambar 2.4 Gantt Chartt **22**
Gambar 2.5 Ilustrasi Struktur pada Algoritma Genetika **23**
Gambar 2.6. Interpretasi kromosom **24**
Gambar 2.7. PMX *Crossover* **26**
Gambar 3.1. Tahapan Penelitian **31**
Gambar 3.2. *Flowchart Hybrid* Algoritma Genetika dan *Simultaed Annealing* **35**
Gambar 3.3. kromsom 2 dan kromosom 4 ..... **40**
Gambar 3.4. Menentukan titik *crossover* **40**
Gambar 3.5. Hasil Menukar Gen **40**
Gambar 3.6. Relasi Pemetaan **40**
Gambar 3.7. Kromosom Hasil *Crossover* **40**
Gambar 3.8. Kromosom 4 ..... **41**
Gambar 3.9. Proses Mutasi **41**
Gambar 3.10. Hasil Mutasi Pada Kromosom 4 ..... **41**
Gambar 3.12. Flowchart Simulated Annealing **42**
Gambar 3.13. Pemindahan Gen **43**
Gambar 3.14. Hasil *Insertion* **43**
Gambar 3.15. Model Pengembangan Waterfall **45**
Gambar 3.16. Arsitektur Perangkat Lunak **46**
Gambar 3.17. *Flowchart* Sistem **46**
Gambar 3.18. DFD Level 0 ..... **47**
Gambar 3.19. DFD Level 1 ..... **47**
Gambar 3.20. DFD Level 2 proses 1 ..... **48**
Gambar 3.21. DFD Level 2 proses 2 ..... **48**
Gambar 3.22. User interface menu penjadwalan **49**
Gambar 3.23. User interface menu tambah data **49**
Gambar 3.24. User interface menu parameter **50**

Daftar Tabel

Tabel 2.1 State of the art 10
Table 2.2. Pencarian Nilai Makespan dengan *I job* dan *j* mesin 25
Tabel 3.1. Proses Potong 32
Tabel 3.2. Proses Jahit 32
Tabel 3.3. Proses Sablon / Bordir 32
Tabel 3.4. Data Pekerjaan dan Waktu Proses Tiap Mesin 33
Table 3.5. Spesifikasi Perangkat Keras 34
Table 3.6. Spesifikasi Kebutuhan Perangkat Lunak 34
Tabel 3.7. Perhitungan *Makespan* Kromosom 1 Populasi Awal 36
Tabel 3.8. Perhitungan *Makespan* Kromosom 2 Populasi Awal 36
Tabel 3.9. Perhitungan *Makespan* Kromosom 3 Populasi Awal 37
Tabel 3.10. Perhitungan *Makespan* Kromosom 4 Populasi Awal 37
Tabel 3.11. Perhitungan *Makespan* Kromosom 5 Populasi Awal 38
Tabel 3.12. Hasil Perhitungan Nilai Objektif 38
Tabel 3.13. Hasil Perhitungan *Invers* 38
Tabel 3.14. Perhitungan Probabilitas Kromosom 39
Tabel 3.15. Populasi Setelah Seleksi 39
Tabel 3.16. Populasi Setelah Proses Crossover 40
Tabel 3.17. Penentuan Mutasi 41
Tabel 3.18. Populasi Setelah Proses Mutasi 41
Tabel 3.19. Perhitungan Nilai Objektif Setelah Insertion 43
Tabel 3.20. Jadwal Penelitian 50

BAB I
PENDAHULUAN

1.1. Latar Belakang

Dalam bidang industri, perdagangan dan berbagai aktivitas lain pastinya memerlukan suatu perencanaan yang baik pada tiap kegiatannya, dengan tujuan untuk memperoleh keuntungan yang tinggi dengan tidak mengabaikan kualitas suatu produk dan efisiensi sumber daya. Penjadwalan produksi biasanya berkaitan dengan pengurutan proses pembuatan atau pengerjaan produk secara menyeluruh terhadap sejumlah mesin yang tersedia. *Flowshop Scheduling Problem* (FSP) sendiri adalah *Scheduling Problem* dengan kondisi dimana setiap pekerjaan harus diproses tepat satu kali pada setiap mesin, dengan urutan mesin yang dilalui setiap pekerjaan harus sama Setiya Widodo (2014) .Hampir semua permasalahan yang dihadapi industri yang tipe produksinya flowshop adalah masalah penjadwalan produksi, terlebih ketika menentukan urutan pekerjaan (jobs) yang akan dikerjakan dengan efisien. Kegagalan dalam menyusun urutan pengerjaan jobs pada mesin akan menyebabkan keterlambatan dalam penyelesaian jobs dan rendahnya utilitas mesin yang akan menyebabkan tingginya biaya produksi (Firdaus et sal., 2015).

Penjadwalan produksi sangat penting pada perusahaan yang menggunakan sistem *make to order*, dimana produk baru akan diproduksi sesuai permintaan dari konsumen, karena waktu yang diperlukan dalam menyelesaikan suatu pekerjaan sangat bergantung terhadap penjadwalan yang dipakai saat proses produksi, selain itu sulitnya melakukan estimasi waktu yang dibutuhkan untuk memproduksi suatu pesanan juga merupakan permasalahan yang sering terjadi. Hal tersebut yang menyebabkan due date yang dijanjikan kepada customer tidak sesuai dengan kemampuan produksi perusahaan sehingga mengakibatkan terjadinya keterlambatan.

Penelitian tentang permasalahan penjadwalan *flowshop* telah banyak dilakukan, penelitian tersebut dilakukan dengan menggunakan beberapa teknik dan algoritma untuk mendapatkan hasil yang semaksimal mungkin. Seperti pada penelitian Raghavan (2015) yang menggunakan *decision tree* (DT) , algoritma *Scatter Search* (SS) dan DT Govindan et al.(2017), algoritma *Golden Ball* (GBA) Sayoti & Essaid Ri (2016), *Tabu Search* Grabowski & Wodecki (2004), Gao et al. (2013),*Simulated Annealing* (SA) Low (2005) Firdaus et al. (2015), Pamungkas (2002), *Constructive Greedy* (CG) dan *Stochastic Greedy* (SG) Ancău (2012) , SA dan TS Lin & Ying (2009), Algoritma Genetika (GA) Etiler et al. (2004),GA dan TS Minmei (2006), dan *Hybrid* GA Wei et al. (2018), (Dai, Tang, Giret, et al., 2013).

Dari beberapa metode-metode yang dapat digunakan untuk menyelesaikan permasalahan penjadwalan *flowshop*, penelitian Sayoti dan Essaid yang menggunakan algoritma Golden Ball dalam menyelesaikan penjadwalan *flowshop*, menjelaskan bahwa GBA yang merupakan algoritma yang terinspirasi dari strategi bermain sepak bola, terlihat bahwa metode tersebut dapat bekerja dengan baik pada kasus penjadwalan *flowshop* dengan cepat dan mendekati optimal akan tetapi algoritma tersebut hanya dapat menyelesaikan permasalahan penjadwalan *flowshop* dengan baik pada permasalahan berskala kecil Sayoti & Essaid Ri (2016). Kemudian pada penelitian Raghavan yang menggunakan metode decision tree menjelaskan bahwa metode DT mudah digunakan dan dirasa efiesien akan tetapi dalam penyelesaian masalahan flow shop ukuran pohon dalam DT bisa sangat besar karena seiring bertambahnya pekerjaan yang ada ukuran pohon juga meningkat Raghavan (2015). Begitu juga

Pada peneltian Govindan yang menggunakan scatter search yang dikombinasikan dengan algoritma DT, penerapan DT bertujuan agar mendapatkan populasi awal yang baik yang kemudian dimasukkan ke SS yang nantinya akan menghasilkan solusi makespan yang optimal / mendekati optimal, akan tetapi ukuran dari pohon yang dihasilkan bisa sangat besar Govindan et al.(2017).selanjutnya pada penelitian Ancău yang menggunakan dua metode yaitu metode CG dan SG hasil dari algoritma bandingkan dengan algoritma NEH (Nawaz, Enscore, Ham) dan terlihat bahwa CG memiliki hasil yang hampir sama dengan NEH, sedangkan SG lebih unggul dari kedua algoritma tersebut, akan tetapi ukuran komputasi yang dihasilkan SG relatif tinggi Ancău (2012), kemudian terdapat beberapa penelitian yang menggunakan algoritma tabu search seperti pada penelitian Grabowski & Wodecki yang menggunakan TS dalam menyelesaikan permutasi didalam *flowshop* dengan kriteria makespan, algoritma tabu search sendiri merupakan algoritma pencarian lokal yang dalam kinerjanya menggunakan tabu list dimana berfungsi agar solusi-solusi yang telah dievaluasi akan disimpan sehingga akan terhindar dari lokal optimum Grabowski & Wodecki (2004). TS juga digunakan dalam penelitian Gao, TS pada penelitian tersebut digunakan untuk menyelesaikan flow shop terdistribusi hasil dari penelitian tersebut mengatakan bahwa TS memiliki kelebihan dibandingan algoritma NEH, VND, dan GA_LS dalam menyelesaikan permasalahan flow shop terdistribusi akan tetapi TS memiliki waktu yang lebih lama dalam prosesnya dibandingkan dengan NEH dan VND Gao et al. (2013). Kemudian ada bebrapa penelitian yang menggunakan (2018)GA dalam permasalahan *flowshop* , pertama GA diimplementasikan untuk menyelesaikan permasalahan *flowshop* dalam penelitian Reeves (1995) setelah penelitian tersebut kemudian banyak peneliti yang menggunakan GA untuk menyelesaikan berbagai macam permasalahan *flowshop* seperti pada beberapa penelitian yang menggunakan GA dalam menyelesaikan *hybrid flowshop* dengan beberapa kriteria Yu et al. (2018), Wang & Liu (2013), begitu juga dengan Rahman yang menggunakan GA untuk permaslahan dalam menyelesaikan permasalahan penjadwalan flowshop dalam sistem produksi make to stock (Rahman et al., 2015).

Dari beberapa penelitian yang sudah dilakukan terlihat bahwa tiap metode memiliki keunggulan dan kelemahan masing-masing, variasi yang dilakukan pada saat pengujian juga berpengaruh pada solusi yang diberikan tiap algoritma . dalam penelitian Dai menjelaskan bahwa banyak algoritma yang dikembangkan untuk menyelesaikan permasalahan yang diangkat yaitu *Flexible Flow Shop* seperti algoirtma genetika, simulated annealing (SA), particle swam optimization, ant colony optimization dan beberapa algoritma lainnya , dari beberapa algoritma tersebut GA dapat memberikan hasil solusi yang optimal/mendekati optimal dalam waktu yang cepat, akan tetapi GA memiliki kekurangan yang fatal yaitu kemungkinan besar GA dapat terjebak dalam *local optimum* atau kovergensi prematur Dai, Tang, Giret, et al.(2013), selain itu pada penelitian Wei mengatakan bahwa diantara algoritma lain dalam menyelesaikan flow shop GA memiliki kemampuan pencarian global dan keceptan yang baik akan tetapi GA lemah dalam pencarian lokal sehingga diperlukan algoritma pencarian lokal untuk memperbaiki kelemahan GA tersebut Wei et al. (2018). penelitian tentang GA juga dilakukan oleh Li dan Gao yang menggunakan hybrid GA dan TS pada permasalahan job shop, TS digunakan untuk meningkatkan kinerja GA dimana GA yang memiliki keunggulan dalam pencarian global (explorasi) dan TS yang unggul pada pencarian local (exploitasi), maka dengan hybrid GA dan TS membuat kemampuan pencarian yang

efektif dan dapat menyeimbangkan intensifikasi dan diversifikasi dengan sangat baik (Li & Gao (2016)Hybrid GA dan TS juga digunakan untuk menyelesaikan permasalahan *flexible flowshop* Sukkerd & Wuttipornpun (2016). Dalam permasalahan konvergensi premature pada GA, Dai mengatakan bahwa SA memiliki kemampuan yang baik untuk mengatasi GA dari local optimum/konvergensi prematur , GA digunakan untuk mendapatkan solusi yang optimal atau mendekati optimal pada ruang solusi yang ada (explorasi), dan kemudian SA digunakan untuk mencari solusi yang lebih baik berdasarkan solusi yang telah didapatkan (exploitasi) Dai, Tang, Giret, et al.(2013), kelebihan SA juga terlihat pada penelitian Lin dan Ying yang membandingan algoritma SA dan TS dan terlihat bahwa SA lebih unggul dari pada TS dalam menyelesaikan permasalahan penjadwalan flowshop Lin & Ying (2009), algoritma GA dan SA juga telah digunakan untuk menyelesaikan masalah *Job Shop* Yin (2013), *Open Shop* Andresen et al. (2008) dan *Flexible Flow Shop* Dai, Tang, Giret, et al(2013), (Dai, Tang, Zheng, et al., 2013).

Maka dari itu ,pada penelitian ini akan menggunakan metode *Hybrid* Algoritma Genetika-Simulated Annealing untuk untuk meminimalkan makespan pada penjadwalan produksi bertipe *flowshop*. Dalam penelitian ini akan menggunakan data dari Hafki Project yang merupakan sebuah industri yang bergerak dibidang pembuatan pakaian yang memiliki sistem *flowshop* pada produksinya.

1.1. Rumusan Masalah

Sesuai dengan uraian latar belakang yang sudah dijelaskan di atas, rumusan masalah dalam penelitian ini adalah sebagai berikut:

Pengaruh penerapan algoritma Simulated Annealing untuk menyelesaikan permasalahan lokal optimum / konvergensi prematur yang teradapat di Algoritma Genetika dalam permasalahan penjadwalan Flow Shop

1.2. Batasan Masalah

Batasan masalah yang ada di dalam penelitian ini adalah sebagai berikut:

a. Data yang digunakan dalam penelitian ini adalah data-data yang berkaitan dengan proses produksi Hafki Project.
b. Terdapat beberapa aturan dalam flowshop menurut (Sayoti & Essaid Ri, 2016):
    1. diasumsikan setiap sumber daya atau bahan mentah yang dibutuhkan untuk proses produksi selalu tersedia.
    2. Urutan mesin yang dilalui setiap pekerjaan harus sama.
    3. semua pekerjaan (*job*) daam keadaan siap diproses (*ready time)* dalam waktu 0 dan bersifat *independent* terhadap yang lain.
    4. tidak ada pekerjaan yang diproses pada waktu yang sama dan di mesin yang sama.
    5. setiap mesin yang digunakan diasumsikan selalu dalam keadaan siap dipakai tanpa adanya gangguan.

1.3. Tujuan Penelitian

Tujuan yang ingin dicapai dari penelitian ini adalah sebagai berikut:

a. Mengetahui performa algoritma Simulated Annealing dalam memperbaiki konvergensi prematur pada Algoritma Genetika.

b. Menerapkan Algoritma Genetika dalam permasalahan penjadwalan produksi bertipe flow shop.

1.4. Manfaat Penelitian

Manfaat yang ingin dicapai dari penelitian yang ingin dilakukan ini adalah sebagai berikut:

a. Manfaat bagi peneliti, peneliti dapat menerapkan ilmu yang didapat selama perkuliahan dan dapat mengimplementasikan Algoritma Genetika untuk menyelesaikan permasalahan penjadwalan produksi.
b. Manfaat penelitian ini bagi industri, dengan adanya penelitian ini maka industri akan dipermudah dalam merencanakan proses produksi terutama pada proses penjadwalan.

1.5. Tahapan Penelitian

Pada penelitian yang akan dilakukan ini, terdapat beberapa tahapan yang akan dilakukan yaitu sebagai berikut:

a. Study Literatur
   Dalam penelitian ini tahap pertama yang dilakukan adalah melakukan *study literature*. *Study literature* dilakukan untuk mencari referensi dari penelitian sebelumnya. Study literature dapat dicari dari jurnal-jurnal yang membahas penelitian serupa.
b. Pengumpulan Data
   Setelah *studi literature* dilanjutkan dengam melakukan pengumpulan data, data yang akan digunakan dalam penelitian ini adalah data yang bersumber dari Hafki Project.
c. Analisis Sistem
   Selanjutnya adalah melakukan analisis kebutuhan perangkat lunak yang ada dibuat di dalam penelitian ini.
d. Pembuatan Model
   Tahap selanjutnya adalah pembuatan model. Pada tahap ini dilakukan pembuatan model untuk permasalahan yang dianggkat dengan menggunakan algoritma dan teknik yang sudah dipilih.
e. Implementasi Perangkat Lunak
   Selanjutnya, setelah model yang dibuat dirasa memiliki performa yang bagus, model tersebut diimplementasikan dalam bentuk perangkat lunak yang bisa digunakan oleh pengguna. Dalam pembuatan perangkat lunak ini, menggunakan metodogi *waterfall*.
f. Pengujian dan Evaluasi Perangkat Lunak
   Setelah perangkat lunak selesai dibuat, dilakukan pengujian perangkat lunak untuk memastikan perangkat lunak yang dibuat berjalan normal tanpa ada kendala.
g. Kesimpulan dan Saran
   Setelah semua tahap dilakukan, didapatkan kesimpulan dari penelitian yang sudah dilakukan tentang bagaimana performa algoritma Simulated Annealing dalam menangani permasalahan konvergensi prematur pada Algoritma Genetika.

1.6. Sistematika Penulisan

Penelitian ini disusun berdasarkan sistematika penulisan yang terdiri dari 5 bab yaitu:

BAB 1 PENDAHULUAN

Pada BAB I ini, membahas latar belakang penelitian ini dilakukan, rumusan masalah yang ada di dalam penelitian ini, batasan masalah, tujuan, dan manfaat penelitian ini dilakukan, serta sistematika penulisan laporan mengenai penelitian yang dilakukan.

BAB II TINJAUAN PUSTAKA

Dalam BAB II ini, berisi tentang landasan teori mengenai obyek penelitian dan metode yang akan dilakukan di dalam penelitian ini, kemudian juga membahas penelitian-penelitian serupa yang sudah dilakukan sehingga menjadi referensi penulis dalam melakukan penelitian ini.

BAB III METODE PENELITIAN

Pada BAB III ini berisi penjelasan tentang metode yang akan digunakan oleh penulis di dalam melakukan penelitian ini. Metode-metode yang dipilih nantinya akan digunakan untuk menyelesaikan permasalahan pada kasus yang sedang diteliti, yaitu penjadwalan produksi.

BAB IV HASIL DAN PEMBAHASAN

Pada BAB IV berisi tentang penjelasan hasil dari tiap tahap penelitian yang sudah dilakukan oleh penulis dengan menggunakan metode yang sudah dijelaskan pada bab sebelumnya. Penjelasan hasil penelitian akan berisi evaluasi performa model yang sudah dibuat dengan menggunakan algoritma yang dipilih.

BAB V PENUTUP

Bab ini akan berisi kesimpulan hasil dari penelitian yang sudah dilakukan oleh penulis. Kemudian penulis juga menambahkan informasi seperti kekurangan dari penelitian yang sudah dilakukan ditambahkan dengan saran yang bisa dilakukan pada penelitian yang akan datang, dapat berupa saran mengenai perbaikan metode supaya dalam penelitian yang akan datang dapat menghasilkan hasil yang lebih baik.

BAB II
KAJIAN LITERATUR

2.1. Tinjauan Studi

Penjadwalan produksi merupakan suatu proses dalam perencanaan dan pengendalian produksi serta pengalokasian sumber daya dengan memperhatikan kapasitas sumber daya yang ada, pengalokasian sumber daya yang ada secara tepat dan efisien sangat diperlukan didalam industri. Penjadwalan flowshop merupakan suatu kegiatan perencanaan produksi yang melibatkan beberapa pekerjaan, jenis mesin dan waktu proses pekerjaan pada tiap mesin. Permasalahan penjadwalan flowshop adalah suatu penjadwalan yang bertujuan untuk mencari urutan dari beberapa pekerjaan yang diproses pada beberapa mesin. Dengan ketentuan urutan mesin yang dilalui setiap pekerjaan harus sama. Tujuan dari penjadwalan flowshop adalah untuk menjadwalkan pekerjaan sedemikian rupa untuk meminimalkan waktu penyelesaian semua pekerjaan (makespan). Terdapat beberapa aturan pada flowshop yaitu diasumsikan setiap sumber daya atau bahan mentah yang dibutuhkan untuk proses produksi selalu tersedia. semua pekerjaan (*job*) daam keadaan siap diproses (*ready time)* dalam waktu 0 dan bersifat *independent* terhadap yang lain, tidak ada pekerjaan yang diproses pada waktu yang sama dan di mesin yang sama dan setiap mesin yang digunakan diasumsikan selalu dalam keadaan siap dipakai tanpa adanya gangguan Sayoti & Essaid Ri (2016),. Penelitian tentang permasalahan penjadwalan flowshop telah banyak dilakukan, penelitian tersebut dilakukan dengan menggunakan beberapa teknik dan algoritma untuk mendapatkan hasil yang semaksimal mungkin. Salah satu penelitian yang sudah dilakukan adalah penelitian yang dilakukan oleh Sayoti dan Essaid yang menggunakan algoritma *Golden Ball* dalam menyelesaikan penjadwalan flowshop, algoritma GBA algoritma ini pertama diusulkan oleh (Osaba et al., 2013), kemudian versi terbaru dari GBA diterbitkan pada tahun 2014 oleh penulis yang sama , GBA merupakan algoritma yang terinspirasi dari strategi bermain sepak bola, terdapat 4 fase pada algoritma GBA yaitu fase inisialisasi, fase pelatihan(training), fase kompetisi, dan fase tranfer. Pada dasarnya konsep dari algoirtma ini adalah dengan membagi solusi awal ke dalam kelompok kemudian setiap kelompok mewakili satu tim, Setiap tim bekerja secara mandiri dan bersaing dengan tim lain untuk mendapatkan solusi terbaik. selanjutnya algoritma GBA dibandingan dengan algoritma lain yaitu algoritma Palmer, CDS, dan NEH terlihat bahwa algoritma GBA lebih unggul dalam menyelesaikan permasalahan penjadwalan flowshop, dalam pengujiannya tiap algoritma akan digunakan untuk menyelesaikan penjadwalan flowshop dengan beberapa variasi pada jumlah pekerjaan dan jumlah mesin terlihat bahwa GBA lebih unggul dilihat dari makespan yang dihasilkan. Hasil pengujian pada penelitian ini mengatakan bahwa GBA dapat bekerja dengan baik pada kasus penjadwalan flowshop dengan cepat dan mendekati optimal akan tetapi algoritma tersebut hanya dapat menyelesaikan permasalahan penjadwalan flowshop dengan baik pada permasalahan berskala kecil saja (Sayoti & Essaid Ri, 2016).

kemudian pada penelitian Ancău yang mengusulkan dua metode yaitu metode CG dan SG untuk menyelesaikan permasalahan penjadwalan flowshop, dalam pengujiannya kedua algoritma tersebut diujikan pada empat kelompok masalah benchmark oleh Prof. E.Halaman Taillard agar telihat bagaimana kinerja dari CG dan SG maka dilakukan perbadingan juga dengan algoritma NEH, maka terlihat bahwa CG dan NEH memiliki kemampuan yang sama

dalam menyelesaikan penjadwalan flowshop dilihat dari makespan yang dihasilkan, sedangkan untuk SG lebih unggul dari CG dan NEH, akan tetapi dalam prosesnya SG memerlukan komputasi yang relatif besar sehingga memerlukan waktu yang lebih lama dari pada NEH penelitian ini juga mengatakan bahwa algoritma yang diusulkan tersebut cocok untuk permasalahan penjadwalan flowshop dengan ukuran sedang atau besar (Ancău, 2012).

selanjutnya terdapat beberapa penelitian yang menggunakan algoritma tabu search seperti pada penelitian Grabowski & Wodecki yang menggunakan TS dalam menyelesaikan permutasi didalam flowshop dengan kriteria makespan dimana algoritma tabu search merupakan algoritma pencarian lokal yang dalam kinerjanya menggunakan tabu list dimana berfungsi agar solusi-solusi yang telah dievaluasi akan disimpan sehingga akan terhindar dari lokal optimum, pada penerapannya algoritma TS ini dikembangkan oleh penulis dengan ditambahkan suatu parameter, dalam pengujiannya pertama algoritma yang diusulkan dibandingan dengan algoirtma TS yang dikembangkan lainnya dan terlihat bahwa algoritma yang diusulkan lebih unggul dilihat dari kecepatannya dari pada 2 algoritma TS yang dikembangkan lainnya, kemudian algoritma yang diusulkan dibandingan dengan beberapa algoritma lainnya dimana pada pengujian kedua ini ditujukan untuk mengetahui bagaimana performa metode yang diusulkan dalam menyelesaikam kasus permutasi flowshop dilihat dari makespan yang dihasilkan, terlihat dalam 30 kali pengujian algoritma yang diusulkan unggul 24 dari algoritma pertama dan 19 dari algoritma kedua sendangkan dengan algoritma ketiga hanya unggul 4 kali percobaan. (Grabowski & Wodecki, 2004). TS juga digunakan dalam penelitian Gao, TS pada penelitian tersebut digunakan untuk menyelesaikan flow shop terdistribusi hasil dari penelitian tersebut mengatakan bahwa TS memiliki kelebihan dibandingan algoritma NEH, VND, dan hybrid GA dan VND dalam menyelesaikan permasalahan flow shop terdistribusi akan tetapi TS memiliki waktu yang lebih lama dalam prosesnya dibandingkan dengan NEH dan VND (Gao et al., 2013)

Kemudian pada penelitian Raghavan yang menggunakan metode decision tree dalam menyelesaikan permutasi penjadawalan flowshop, Raghavan mengatakan bahwa banyak metode yang dipakai dalam menyelesaikan permasalahan flowshop dengan memberikan solusi yang cepat akan tetapi tidak ada yang mendemonstrasikan atau menjelaskan tentang bagaimana metode tersebut diimplementasikan, sehingga membutuhkan waktu yang lama dalam proses pengkodeannya apalagi dalam permasalahan yang kompleks maka dari itu banyak peneliti yang berusaha mencari algoritma yang efisien dan praktis dalam menyelesaikan permasalahan flowshop, Raghavan mengatakan bahwa keuntungan DT adalah mudah digunakan dan efiesien selain itu aturan-aturannya juga mudah dibuat dan dipahami, pada pengujiannya algoritma tersebut digunakan untuk memecahkan 60 masalah benchmark dari Taillard (Taillard, 1993), kemudian DT dibandingkan dengan beberapa algoirtma yaitu Liu and Reeves heuristics (LS), SA, Deepak Laha's (DL) composite heuristics H-1, H-2, setelah pengujian terlihat bahwa rata-rata yang dihasilkan DT unggul dari SA dan LS selain itu dalam penyelesaian masalahan flow shop ukuran pohon dalam DT bisa sangat besar karena seiring bertambahnya pekerjaan yang ada ukuran pohon juga meningkat (Raghavan, 2015). Begitu juga Pada peneltian Govindan yang menggunakan scatter search yang dikombinasikan dengan algoritma DT, penerapan DT bertujuan agar mendapatkan populasi awal yang baik yang kemudian akan dilanjutkan ke SS, meskipun begitu seperti yang dikatakan Raghavan ukuran pohon yang dihasilkan DT bisa sangat besar (Govindan et al., 2017).

Kemudian pada penelitian Firdaus yang menggunakan algoritma SA untuk menyelesaikan penjadwalan flowshop , SA sendiri merupakan suatu algoritma yang bekerja berdasarkan analogi dari proses annealing yang merupakan suatu proses dalam pembentukan kristal dalam suatu materi. Proses dari SA diawali dengan dipilihnya suatu solusi awal, yang mengidentifikasikan kondisi materi sebelum proses dimulai. Gerakan bebas dari atom pada suatu materi, direpresentasikan dalam bentuk modifikasi terhadap solusi awal/solusi sementara agar dapat diproses oleh SA. Pada saat awal proses SA, ketika parameter suhu (T) diatur tinggi, solusi sementara yang sudah ada diperbolehkan untuk mengalami modifikasi dengan bebas, Kebebasan dari modifikasi tersebut secara relatif diukur berdasarkan nilai fungsi tertentu yang berguna untuk mengevaluasi seberapa optimal solusi sementara yang dihasilkan. Bila nilai fungsi dari hasil evaluasi solusi yang dimodifikasi ini membaik, solusi hasil modifikasi tersebut akan digunakan sebagai solusi selanjutnya. Tetapi ketika nilai fungsi evaluasi hasil modifikasi memburuk, ketika temperatur annealing masih tinggi, solusi yang lebih buruk ini masih mungkin diterima. Dalam tahapan selanjutnya saat temperatur sedikit demi sedikit dikurangi, maka kemungkinan untuk menerima langkah modifikasi yang tidak memperbaiki nilai fungsi evaluasi semakin berkurang. Sehingga kebebasan untuk memodifikasi solusi semakin berkurang, hingga akhirnya diperoleh solusi yang mendekati solusi optimal, dalam teknik pengacakannya Firdaus menggunakan metode swap (menukar), flip (membalik), dan slide (menggeser). Setelah itu dilakukang pengujian dengan mengubah parameter-parameter suhu awal (T) dan faktor pereduksi suhu (fp) dan dihasilkan bahwa dengan parameter suhu (T) 100 dan parameter faktor penurunan suhu (fp) 0.5 menghasilkan makespan sebesar 2.636 menit ≈ 44 jam yang merupakan hasil terbaik dari beberapa parameter yang telah dicoba. Dan kemudian dilakukan perbandingan dengan penjadwalan yang dilakukan oleh perusahaan yang menghasilkan makespan sebesar 2917 menit ≈ 49jam. Sehingga makespan yang dihasilkan penjadwalan dengan metode SA lebih baik dengan selisih waktu 5 jam hampir mendekati 1 hari jam dari kondisi sebelumnya. Selain kriteria makespan juga didapat idle time dari metode SA lebih baik yaitu 2.81 % pada tiap mesin(Firdaus et al., 2015) Kemudian SA juga digunakan pada penelitian Lin dan Ying yang menggunakan hybrid SA dan TS ,hasil dari algoritma tersebut lalu dibandingkan dengan hasil dari SA dan TS , dari 40 kali percobaan terlihat bahwa TS unggul sebanyak 3 kali (7.5%) ,SA unggul sebanyak 17 kali(42.5%) dan algoritma yang diusulkan unggul 28 kali (70%) (Lin & Ying, 2009).

Selanjutnya banyak peneliti yang menggunakan GA dalam menyelesaikan kasus penjadwalan produksi GA sendiri merupakan algoritma yang diadaptasi dari prosedur evolusi genetik dimana tujuannya sendiri adalah mencari individu yang terbaik yang berhasil bertahan dalam proses evolusi.GA bekerja dengan satu set solusi yang dikodekan yang sesuai dengan suatu permasalahan, satu set solusi tersebut disebut populasi. Setiap solusi direpresentasikan dari serangkaian gen, disebut dengan kromosom. Setiap kromosom akan dievaluasi dan nilai obyektif dari sebuah solusi/kromosom dinamakan fitness, kemudian untuk mendapatkan solusi terbaik dilakukan seleksi pada tiap solusi yang ada, kemudian dalam proses regenerasi solusi terdapat proses yang disebut crossover dan mutasi , skema evolusi akan selalu berulang sampai suatu kriteria pemberhentia terpenuhi (Yu et al., 2018).pertama GA diimplementasikan untuk menyelesaikan permasalahan flow shop dalam penelitian (Reeves, 1995) pada penelitian tersebut GA digunakan untuk menyelesaikan permasalahan flow shop dengan parameter makespan , Reeves mengatakan bahwa GA dan SA memiliki kemampuan yang hampir sama

untuk menyelesaikan permasalahan flow shop akan tetapi GA lebih unggul dalam menyelesaikan kasus flow shop ukuran besar. setelah penelitian tersebut kemudian banyak peneliti yang menggunakan GA untuk menyelesaikan berbagai macam permasalahan flow shop seperti pada beberapa penelitian yang menggunakan GA dalam menyelesaikan *hybrid flowshop* dengan beberapa kriteria (Yu et al., 2018) (Wang & Liu, 2013), begitu juga dengan Rahman yang menggunakan GA untuk permasalahan dalam menyelesaikan permasalahan penjadwalan flowshop dalam sistem produksi make to stock (Rahman et al., 2015). Seperti yang dikatakan pada beberapa penelitian yang menggunakan GA dalam menyelesaikan permaslahan produksi banyak yang mengatakan bahwa GA unggul dari beberapa metode lainnya akan tetapi banyak penelitian yang mengatakan bahwa GA memeiliki kekurangan yaitu seringnya GA terjerbak pada konvergensi premature dan beberapa mengatakan bahwa GA lemah pada pencarian lokal (Dai, Tang, Giret, et al., 2013)(Li & Gao, 2016)(Wei et al., 2018), sehingga banyak peneliti yang berusaha memperbaiki kelemahan dari GA tersebut, seperti pada penelitian Dai yang menjelaskan bahwa banyak algoritma yang dikembangkan untuk menyelesaikan permasalahan yang diangkat yaitu *Flexible Flow Shop* seperti algoirtma genetika, simulated annealing (SA), particle swam optimization, ant colony optimization dan beberapa algoritma lainnya , dari beberapa algoritma tersebut GA dapat memberikan hasil solusi yang optimal/mendekati optimal dalam waktu yang cepat, akan tetapi GA memiliki kekurangan yang fatal yaitu kemungkinan besar GA dapat terjebak dalam local optimum atau kovergensi premature (Dai, Tang, Giret, et al., 2013). Pada penelitian Dai mengatakan bahwa SA memiliki kemampuan yang baik untuk mengatasi GA dari local optimum/konvergensi premature sehingga Dai megusulkan algortima GA dan SA , GA digunakan untuk mendapatkan solusi yang optimal atau mendekati optimal pada ruang solusi yang ada (explorasi), dan kemudian SA digunakan untuk mencari solusi yang lebih baik berdasarkan solusi yang telah didapatkan (exploitasi), Kemudian pada pengujiannya GA-SA digunakan untuk menyelesaiakan permaslahan *flexible flowshop* dengan 12 pekerjaan,3 tahap, dan jumlah mesin parallel tiap tahapan yaitu 3,2, dan 4. Pengujian dilakukan dengan beberapa percobaan dengan mengubah parameter bobot makespan dan konsumsi energi ,setelah dilakukan pengujian maka dihasilkan bahwa GA-SA dapat meningkatkan rasio rata-rata energi total menjadi 85.13% pada konsumsi energi yang menganggur , nilai konsumsi energi yang mengganggur menurun menjadi 6.64 dan konsumsi energi total menjadi 260.36 sehingga konsumsi energi menggangur menurun sebesar 12.72% maka terlihat bahwa GA-SA dapat bekerja dengan baik dan efisien (Dai, Tang, Giret, et al., 2013). Masih terkait dengan permaslahan penjadwalan produksi hybrid GA dan TS juga diusulkan pada penelitian Li dan Gao yang digunakan untuk menyelesaikan permasalaghan flexible jobshop dalam hasil percobaan menunjukkan bahwa hybrid GA dan TS yang diusulkan telah mencapai peningkatan yang signifikan untuk menyelesaikan permaslahan flexible jobshop terlepas dari keakuratan solusi dan waktu komputasi , pada penelitian tersebut juga mengatakan dengan ditambahkannya algoritma pencarian local pada GA dapat memberikan kemampuan pencarian yang efektif dan dapat menyeimbangkan intensifikasi dan diversifikasi dengan sangat baik (Li & Gao, 2016).Selain itu hybrid GA dan TS juga digunakan pada penelitian Sukkerd untuk menyelesaikan permasalahan *flexible flowshop* pada penelitian tersebut mengatakan bahwa dengan TS maka dapat meningkat kualitas populasi sebelum operasi GA diterapkan sehingga diharapkan dapat memberikan hasil yang lebih baik, dalam pengujiannya metode tersebut dibandingkan dengan GA dan TS dan

terlihat bahwa hybrid GA dan TS lebih unggul dari kedua algoritma tersebut meskipun perbedaan tidak signifikan, selain itu dilihat dari waktu komputasinya hybrid GA dan TS membutuhkan waktu yang lebih lama daripada GA dan TS (Sukkerd & Wuttipornpun, 2016). kemudian pada penelitian Wei mengatakan bahwa diantara algoritma lain dalam menyelesaikan flow shop GA memiliki kemampuan pencarian global dan keceptan yang baik akan tetapi GA lemah dalam pencarian lokal sehingga diperlukan algoritma pencarian lokal untuk memperbaiki kelemahan GA tersebut, untuk mengatasi permasalahan tersebut Wei mengusulkan hybrid GA dan SA selain itu Wei menambahkan metode MME yang merupakan kombinasi dari algoritma memetic dan NEH pada awal pembentukan populasinya, kemudian dalam pengujiannya algoritma yang diusulkan digunakan untuk memecahakan permaslahan Talliard benchmarks dan dibandingkan dengan metode hybrid algoritma genetika, iterated greedy yang ditingkatkan, algoritma memetic dan beberapa algoritma lainnya dan keunggulan metode yang diusulkan terlihat jelas. Dalam pengujian hingga permasalahan dengan ukuran 500×20 , metode yang diusulkan mengalami peningkatan sebesar 31,7%, 30,5%, 80,6%, 36,1%, dan 71,9% dibandingkan dengan algoritma-algoritma yang dibandingkan (Wei et al., 2018).

Tabel 2.1 State of the art

No.	Penulis	Judul	Metode	Hasil Penelitian
1.	Fatima Sayoti dan Mohammed Essaid Riffi	Golden Ball Algorithm for solving Flow Shop Scheduling Problem	Golden Ball Algorithm	Penelitian ini menggunakan metode *Golden Ball* untuk menyelesaikan permaslahan flowshop , metode GBA yang digunakan pada penelitian ini merupakan metode GBA yang telah diperbarui, dalam pengujiannya metode GBA digunakan untuk menyelsaikan permasalhan flowshop dengan beberapa variasi mesin dan perkerjaan (*job)* . untuk mengetahui kinerja dari algoritma GBA, algoritma tersebut dibandingkan dengan metode Palmer, CDS, dan NEH untuk menyelesaikan permasalahan yang sama. Pada penelitian ini terlihat bahwa GBA lebih unggul dari pada ketiga algoritma yang dibandingkan dilihat

Tabel 2.1 State of the art

				dari maskepan yang dihasilkan, meskipun begitu GBA memiliki kelemahan yaitu metode tersebut hanya bekerja dengan baik pada permasalahan flowshop berskala kecil.
2.	Mircea ANCĂU	On Solving Flowshop Scheduling Problems	Constructive Greedy, Stochastic Greedy	Penelitian ini mengusulkan dua metode yaitu *constructive greedy* (CG) dan *stochastic greedy* (SG). Dalam pengujiannya kedua algoritma tersebut digunakan untuk menyelesaikan empat kelompok masalah *benchmark* oleh Prof.E.Halaman Taillard, pada pengujiannya dua metode tersebut juga dibandingan dengan meotde NEH (Nawaz, Enscore, Ham). Hasil dari pengujian tersebut terlihat bahwa CG memiliki kemampuan yang hamper sama dengan NEH, sedangkan SG memiliki hasil yang lebih baik daripada CG dan NEH dalam menyelsaikan permasalahan *flowshop* dilihat dari makespan yang dihasilkan, akan tetapi SG memiliki kelemahan karena dalam prosesnya SG memerlukan komputasi yang relatif besar sehingga memerlukan waktu yang lebih lama dalam prosesnya.

Tabel 2.1 State of the art

3.	Jozef Grabowskia , Mieczyslaw Wodeckib	A very fast tabu search algorithm for the permutation flow shop problem with makespan criterion	Tabu Search	Penelitian ini mengusulkan algoritma *Tabu Search* yang telah dikembangkan, pada pengujiannya algoritma yang diusulkan dibandingkan dengan algoritma *Tabu Search* yang telah dikembangkan lainnya yang dilihat dari kecepatan dan makespan yang dihasilkan. Hasil dari pengujian pertama terlihat bahwa algoritma yang diusulkan lebih unggul dalam kecepatan dan komputasinya dibandingkan 2 algoritma lainnya, kemudian pada pengujian kedua algoritma tersebut hanya unggul dalam 4 dari 30 percobaan dari 3 algoritma lainnya.
4.	O.Etiler, B.Toklu, M.Atak, J.Wilson	A genetic algorithm for flow shop scheduling problems	*Genetic Algorithm*	Penelitian ini menggunakan algoritma genetika untuk menyelesaikan permasalahan *flowshop*, dalam implementasinya penelitian ini menggunakan teknik *crossover LOX* (*linier order crossover*) dan pada mutasinya menggunakan dua teknik yaitu *exchange mutation* dan *shift mutation* dalam pengujiannya algoritma yang diusulkan dibandingan dengan algoritma NEH dalam menyelesaikan permaslahan *flowshop* dengan beberapa variasi, dan mmberikan hasil bahwa algoritma yang diusulkan

Tabel 2.1 State of the art

				unggul 139 kali dari 230 percobaan, algoritma yang diusulkan juga dibandingkan dengan alogritma genetika dari chen dan terlihat algoritma genetika yang diusulkan unggul 78% dan algoritma genetika chen unggul 38%.
5.	Balasundaram Rathinam, Kannan Govindan, Baskar Neelakandan, Siva Sankar Raghavan	Rule Based Heuristic Approach For Minimizing Total Flow Time In Permutation Flow Shop Scheduling	Decision Tree	Penelitian ini menggunakan metode DT untuk menyelesaikan permasalahan *flowshop* , peniliti mngusulkan DT karena DT dirasa efisien dan praktis sehingga dalam penggunaannya DT lebih mudah di implementasikan. Dalam pengujiannnya DT diujikan dengan menyelesaikan 60 masalah *benchmark* dari Taillard. Kemudian DT bandingkan dengan LS,SA, dan DL dan terlihat bahwa DT lebih unggul dari LS dan SA akan tetapi dalam prosesnya pohon yang dihasilkan DT bisa semakin besar seiring bertambahnya perkerjaan (job) yang ada.
6.	Watcharapan Sukkerd dan Teeradej Wuttipornpun	Hybrid genetic algorithm and tabu search for finite capacity material requirement planning system in flexible flow shop with assembly operations	Hybrid Genetic Algorithm and Tabu Search	Penelitian ini mengusulkan metode hybrid GA dan TS untuk menyelesaikan permasalahan *flexible flowshop* pada penelitian tersebut mengatakan bahwa dengan adanya TS maka dapat meningkat kualitas populasi sebelum operasi GA diterapkan sehingga

Tabel 2.1 State of the art

				diharapkan dapat memberikan hasil yang lebih baik, dalam pengujiannya ketiga metode tersebut dibandingkan dengan metode terdahulu dan terlihat bahwa perbedaan makespan yang dihasilkan lebih unggul dalam kisaran 4,60–4,83% , jika dibandingan 3 algoirtma tersebut hybrid GA dan TS lebih unggul dari kedua algoritma tersebut meskipun perbedaan tidak signifikan, selain itu dilihat dari waktu komputasinya hybrid GA dan TS membutuhkan waktu yang lebih lama daripada GA dan TS.
7.	Hongjing Wei , Shaobo Li, Houmin Jiang, Jie Hu dan Jianjun Hu.	Hybrid Genetic Simulated Annealing Algorithm for Improved Flow Shop Scheduling with Makespan Criterion	Hybrid Genetic Algorithm and Simuated Annealing	Penelitian ini menggunakan metode hybrid GA dan SA dalam menyelesaikan permasalahan *flowshop* , selain itu pada penelitian ini juga menerapkan metode MME yang bertujuan agar dapat memperoleh populasi awal yang baik. Pada penelitian ini mengatakan bahwa GA memiliki kemampuan pencarian global dan keceptan yang baik akan tetapi GA lemah dalam pencarian lokal sehingga diperlukan algoritma pencarian lokal untuk memperbaiki kelemahan GA tersebut maka ditambahkan SA, pada pengujiannya

Tabel 2.1 State of the art

				algoritma yang diusulkan digunakan untuk memecahkan permasalahan Talliard benchmarks dan terlihat bahwa dalam pengujian hingga permasalahan dengan ukuran 500 × 20 , metode yang diusulkan mengalami peningkatan sebesar 31,7%, 30,5%, 80,6%, 36,1%, dan 71,9% dibandingkan dengan 5 algoritma lainnya..
8.	Xinyu Li, Liang Gao	An Effective Hybrid Genetic Algorithm and Tabu Search for Flexible Job Shop Scheduling Problem	*Hybrid* Genetic Algorithm and Tabu Search	Pada Penelitian ini menggunakan metode *Hybrid* algoritma genetika dan *tabu search* untuk menyelesaikan permaslahan penjadwalan produksi bertipe *flexible jobshop*, dalam implementasinya TS digunakan setelah proses GA selesai pada tiap iterasinya , solusi terbaik yang didapatkan GA akan diproses oleh TS sehingga diharapkan dapat mendapatkan hasil yang lebih baik, dalam pengujiannya metode yang diusulkan dibandingkan dengan metode-metode lain terdahulu dalam menyelesaiakan permasalahan *benchmark*, dan hasilnya metode yang diusulkan dapat memiliki peningkatan yang signifikan terlepas dari akurasi solusi dan waktu komputasi.

Tabel 2.1 State of the art

9.	Min Dai , Dunbing Tang, Adriana Giret , Miguel A. Salido , W.D. Li	Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm	Genetic Algorithm and Simulated Annealing	Penelitian ini mengusulkan metode GA dan SA untuk menyelsaikan permaslahan *flexible flowshop*, pada penelitian ini mengatakan bahwa banyak algoritma yang dapat digunakan seperti algoirtma genetika, simulated annealing (SA), particle swam optimization, ant colony optimization dan beberapa algoritma lainnya , dari beberapa algoritma tersebut GA dapat memberikan hasil solusi yang optimal/mendekati optimal dalam waktu yang cepat, akan tetapi GA memiliki kekurangan yang fatal yaitu kemungkinan besar GA dapat terjebak dalam lokal optimum atau kovergensi premature, peneliti juga mengatakan bahwa SA memiliki kemampuan yang baik untuk mengatasi permasalhan GA tersebut. Kemudian pada pengujiannya GA-SA digunakan untuk menyelesaiakan permaslahan *flexible flowshop* dengan 12 pekerjaan,3 tahap, dan jumlah mesin parallel tiap tahapan yaitu 3,2, dan 4. Pengujian dilakukan dengan beberapa percobaan dengan mengubah parameter bobot untuk makespan dan konsumsi

Tabel 2.1 State of the art

				energi, dan dihasilkan bahwa GA-SA dapat meningkatkan rasio rata-rata energi total menjadi 85.13% pada konsumsi energi yang menganggur , nilai konsumsi energi yang mengganggur menurun menjadi 6.64 dan konsumsi energi total menjadi 260.36 sehingga konsumsi energi menggangur menurun sebesar 12.72%..
10.	S.-W. Lin dan K.-C. Ying	Applying a hybrid simulated annealing and tabu search approach to non-permutation flowshop scheduling problems	Tabu Search, Simulated Annealing, Hybrid Simulated Annealing-Tabu Search	Penelitian ini menggunakan metode hybrid SA dan TS untuk menyelesaikan permasalahan *flowshop* , kemudian pada pengujiannya algoritma SA,TS dan hybrid SA dan TS digunakan untuk mnyelesaikan permasalahan benchmark Demirkol (1998) yang berisi 600 permasalahan secara acak. Setelah diujikan terlihat dari 40 kali percobaan TS unggul sebanyak 3 kali (7.5%) ,SA unggul sebanyak 17 kali(42.5%) dan algoritma yang diusulkan unggul 28 kali (70%), sehingga dapat dilihat bahwa dalam menyelesaikan permaslahan *flowshop* SA lebih unggul dari pada TS.

2.2. Tinjauan Pustaka

2.2.1. Penjadwalan

Pengertian dari penjadwalan sendiri merupakan suatu kegiatan dalam melakukan pengaturan waktu pada mesin yang mencakup kegiatan mengalokasikan fasilitas dan peralatan. Penjadwalan berguna untuk menentukan urutan pekerjaan Baker & Trietsch (2013). Sedangkan

menurut Ginting ,penjadwalan merupakan pengurutan dalam pembuatan atau pengerjaan produk secara menyeluruh yang diproses pada beberapa buah mesin. Dengan begitu masalah pengurutan akan melibatkan pengerjaan sejumlah komponen yang sering disebut dengan istilah 'job'. Sedangkan job sendiri masih merupakan komposisi dari sejumlah elemen-elemen dasar yang disebut aktivitas atau operasi, aktivitas atau operasi ini membutuhkan alokasi sumber daya tertentu selama periode waktu tertentu yang sering disebut dengan waktu proses. Penjadwalan merupakan alat ukur yang baik bagi perencanaan agregat. Pesanan-pesanan aktual pada tahap ini akan ditugaskan pertama kalinya pada sumberdaya tertentu (fasilitas, pekerja, dan peralatan), kemudian dilakukan pengurutan kerja pada tiap-tiap pusat pemrosesan sehingga dicapai optimalitas utilisasi kapasitas yang ada. Terdapat dua pola aliran pada penjadwalan produksi yaitu flow shop dan job shop, pada pola flow shop mempunyai tipe aliran yang memiliki urutan tertentu yang sama, flow shop sendiri terbagi menjadi dua yaitu pure flow shop dan general flow shop, pada pure flow shop setiap pekerjaan yang dilakukan pada proses produksi memiliki kesamaan dalam urutan produksinya atau tidak adanya variasi, sedangkan general flow shop dimungkinkan adanya variasi antara pekerjaan atau pekerjaan yang dilakukan tidak harus dikerjakan di setiap mesin. Sedangkan job shop setiap pekerjaan memiliki pola aliran kerja yang berbeda(Ginting, 2009).

Menurut Bedtttorth (1987) yang dikutip dari Ginting (2009). mengidentifikasikan beberapa tujuan dari aktivitas penjadwalan adalah sebagai berikut:

1. Mengoptimalkan penggunaan sumber daya atau meminimalkan waktu tunggunya, yang bertujuan agar total waktu proses dapat berkurang, sehingga produktivitas dapat meningkat.
2. Dapat mengurangi persediaan barang setengah jadi atau mengurangi pekerjaan-pekerjaan yang menganggur dalam suatu antrian pada saat sumber daya masih dalam keadaan mengerjakan tugas yang lain. Teori barker menjelaskan jika aliran ketrja suatu jadwal konstan , maka antrian yang mengurangi rata-rata waktu alir dapat mengurangi rata-rata dari persediaan barang setengah jadi.
3. Mengurangi beberapa kelambatan pada pekerjaan yang mempunyai batas waktu penyelesaian sehingga akan meminimisasi *penalty cost* (biaya kelambatan).
4. Mengurangi beberapa keterlambatan pada pekerjaan yang memiliki batas waktu peneyelesaian (deadline) sehingga penalty cost (biaya keterlambatan) akan terminimalisir.
5. Dapat membantu dalam pengambilan keputusan dalam perencanaan kapasitas pabrik dan jenis kapasitas yang diperlukan sehingga penggunaan biaya dapat lebih optimal.

2.2.2. Jenis Aliran Proses Produksi

Adapun jenis –jenis aliran proses produksi secara umum yang dimiliki banyak perusahaan :

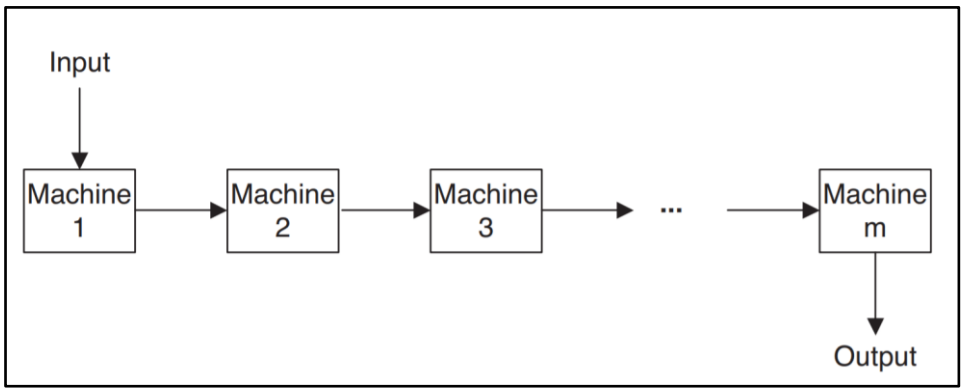
1. Flowshop

Pada aliran ini merupakan aliran ketika lantai produksi memproses produknya dengan urutan yang sama dari bahan awal hingga produk setengah jadi atau produk jadi (selesai). Pada saat suatu produk sudah diproses oleh mesin yang sudah dilalui, maka produk tersebut tidak bisa diproses kembali pada mesin tersebut.

Kemudian terdapat beberapa Aliran Flowshop yang dijelaskan (Baker & Trietsch, 2013):

a. Pure Flowshop

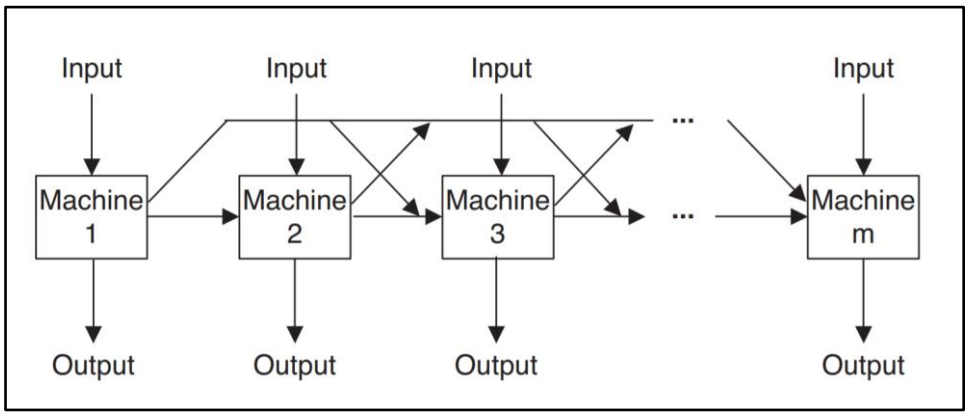
Semua Jenis pekerjaan melalui urutan proses yang sama dan memilki pola aliran yang identik.



Gambar 2.1 Pola aliran pure flowshop

b. General Flowshop

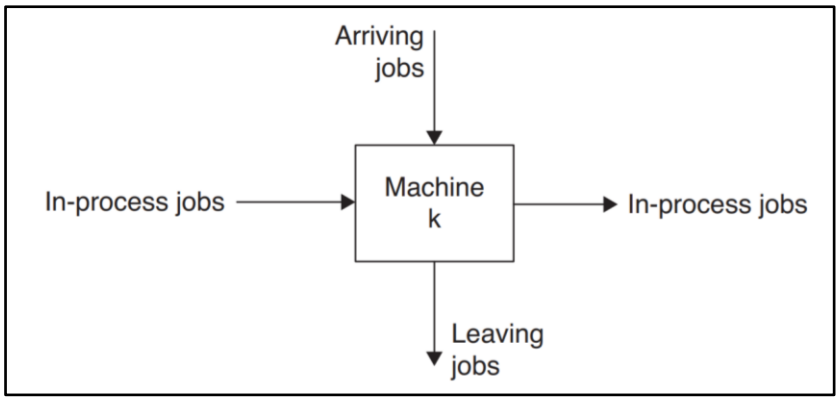
Pola aliran proses tidak identik. Dalam aliran proses ini pekerjan bisa memiliki lebih sedikit dari banyaknya operasi yang ada, operasinya tidak selalu membutuhkan mesin yan berdekatan atau operasi yang berurutan, dan proses operasi tidak selalu dimulai di mesin awal dan atau berakhir di mesin terakhir.



Gambar 2.2 Pola aliran general flowshop

2. Job Shop

Aliran Proses Jobshop Menurut Baker & Trietsch (2013) berbeda denga flowshop. aliran proses pada jobshop tidak searah. maksudnya adalah ada beberapa pekerjaan yang memiliki proses yang berbeda urutannya atau biasa disebut random serta dalam prosesnya dapat diproses lebih dari satu kali pada mesin yang sama.



Gambar 2.3 Pola aliran Jobshop

2.2.3. Ukuran performansi Penjadwalan

Ukuran Performansi merupakan tujuan dari penjadwalan akan hasil yang diinginkan . Kriteria ukuran performansi yang digunakan untuk mengevaluasi penjadwalan mesin dapat diklarifikasikan menjadi dua Baker & Trietsch (2013) yaitu:

1. Kritria berdasarkan atribut tugas.
   a. *Completion time*, yaitu saat dimana pengerjaan suatu job sudah selesai pada suatu stasiun kerja.
   b. *Mean flow time*, merupakan waktu rata-rata dihabiskan untuk menyelsaiakan pekerjaan dilantai pabrik.
   c. *Mean weight flow time*, hampir sama dengan mean flow time , akan tetapi disini juga mempertimbangkan prioritas pengerjaan setiap job.
   d. Maksimum *lateness*, yaitu besarnya simpangan maksimum atau selisi dari waktu penyelesaian semua pekerjaan (*job*) yang dijadwalkan terhadap batas waktu penyelesaian job tersebut.
   e. *Mean tardiness*, merupakan nilai rata-rata keterlambatan seluru pekerjaan (*job*) yang dijadwalkan.
   f. *Mean weight tardiness*, yaitu rata-rata keterlambatan seluru pekerjaan (*job*) yang akan dijadwalkan dengan adanya faktor prioritas pengerjaan masing-masing job.
2. Kriteria berdasarkan atribut pabrik.
   a. Utilitas mesin, merupakan rasio dari sejumlah waktu proses yang dibebankan pada mesin dengan rentang waktu untuk menyelesaikan seluruh tugas pada semua mesin.
   b. Minimasi makespan , yaitu jangka waktu penyelesaian seluruh pekerjaan (*job*) yang akan dijadwalkan yang merupakan jumlah dari seluruh proses.
   c. Pemenuhan *due date*, yaitu merupakan penyelesaian pekerjaan sesuai dengan batas waktu yang ditentukan oleh pelanggan dimana harus selalu dilakukan produsen untuk pempertahankan pelangannya

2.2.4. Penjadwalan *Flowshop*

Penjadwalan Flowshop adalah tipe penjadwalan yang pergerakan memiliki pola yang berurutan dari mesin awal kemudian melewati rangkaian stasiun kerja hingga mesin terakhir sebelum pekerjaan diselesaikan Baker & Trietsch (2013). Kondisi yang terdapat dalam penjadwalan *flowshop* , antara lain :

1. Sekumpulan pekerjaan (*job)* multi-operasi yang tidak terkait dan siap untuk diproses diwaktu nol. (Setiap pekerjaan membutuhkan beberapa operasi, dan setiap operasi membutuhkan mesin yang berbeda)
2. Tidak ada 2 operasi atau lebih pada pekerjaan (*job*) yang sama yang berjalan secara simultan.
3. *No Pre-emption*, setiap operasi pada suatu mesin harus dikerjakan sampai selesai sebelum mesin tersebut melaukan operasi lain.
4. *No Cancellation*, ketika job sudah diproses oleh mesin maka job tersebut harus diproses hingga selesai.
5. Job deskripsi sudah diketahui sebelumnya.
6. Tidak ada kerusakan mesin dan mesin selalu tersedia dalam proses produksi.

Dalam model mesin tunggal untuk menemukan urutan yang optimal, perlu dilakukan pemeriksaan setiap kemungkinan yang dapat terjadi pada urutan pengerjaan tiap pekerjaan (*jobs)* sehingga bisa dikatakan dalam pengecekannya terdapat permutasi n! sesuai dengan jumlah pekerjaan. Begitu pula pada masalah flow shop, ada n! urutan pekerjaan yang berbeda untuk setiap mesin, dan berpotensi sebanyak (n!)m kemungkinan. Maka dari itu akan sangat membantu apabila terdapat pendekatan-pendekatan agar mempermudah dalam pencarian urutan yang optimal pada penjadwalan *flowshop*.

2.2.5. Perhitungan Fungsi Tujuan

Pada perhitungan fungsi tujuan akan dihitung berdasarkan nilai makespan. Dalam menentukan nilai waktu mulai dan waktu selesai pada proses perhitungan fungsi tujuan menggunakan syarat sebagai berikut (Setiya Widodo et al., 2014):

a. Jika operasi tersebut tidak memiliki operasi prasyarat (*job*) dan operasi pendahulu (mesin), maka letakkan operasi dengan waktu mulai = 0.
b. Jika tidak terdapat operasi prasyarat (*job*) namun terdapat operasi pendahulu (mesin), maka waktu mulai operasi = waktu selesai operasi pendahulu.
c. Jika tidak terdapat operasi pendahulu (mesin) namun terdapat operasi prasyarat (job), maka waktu mulai operasi = waktu selesai operasi prasyarat (*job*).
d. Jika terdapat operasi prasyarat (*job*) dan operasi pendahulu (mesin), maka waktu mulai operasi = waktu selesai terlama diantara operasi prasyarat (*job*) dan operasi pendahulu.

Dimana pada operasi prasyarat memiliki hubungan dalam satu pekerjaan (*job*), sedangkan operasi pendahulu memiliki hubungan dalam operasi mesin, Maka rumus menghitung Z (Makespan) ditunjukkan pada persamaan (2.1).

$$Z = \max_{1 \le j \le n} (F_j) \quad \text{.....(2.1)}$$

Keterangan

Z = Makespan

J = Pekerjaan (*job*)

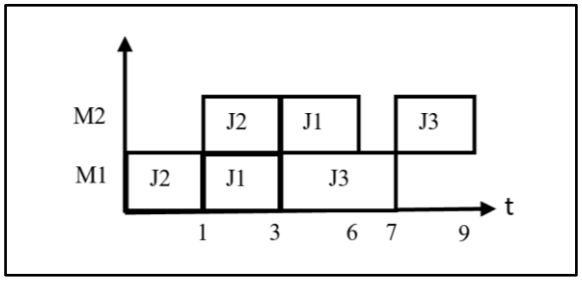
n = Jumlah Pekerjaan

F_j = Nilai maksimum dari Flowtime

2.2.6. *Gantt Chart*

Dengan menggunakan *Gantt* chart maka dapat mempermudah dalam perhitungan *makespan* , *Gantt chart* diperkenalkan pertama kali oleh Hendry Gantt pada tahun 1961. *Gantt*

chart merupakan representasi grafis dari *job-job* yang harus diselesakan dan digambarkan dalam bentuk batang dan analog dengan waktu dan penyelesaian *job* tersebut, berikut ilustrasi dari *gantt chart* :



Gambar 2.4 Gantt Chartt

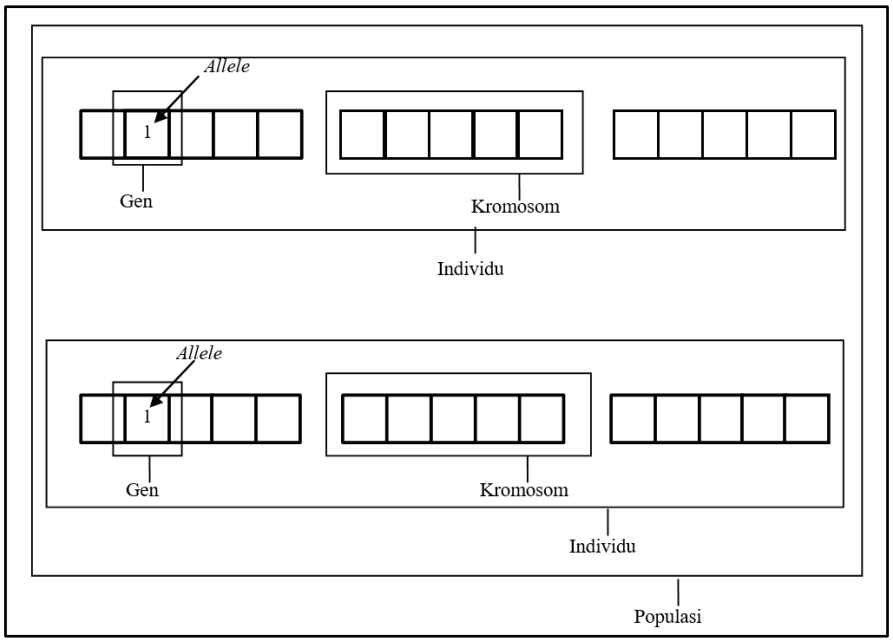
Pada Gambar 2.1 digambarkan dengan sumbu horizontal sebagai waktu dan sumbu vertical sebagai mesin yang digunakan. Maka, *Machine Oriented Gantt Chart* adalah diagram *gantt* yang berorientasi pada mesin.

Kelebihan dalam mengunakan diagram gantt adalah (Ginting, 2009):

a. Dalam situasi awal mengenai penggunaan diagram gantt memungkinkan evaluasi lebh awal mengenai penggunaan sumber daya seperti yang telah direncanakan
b. Kemajuan pekerjaan mudah diperiksa pada setiap waktu karena sudah tergambar dengan jelas
c. Semua pekerjaan diperlihatkan secara grafis dalam suatu diagram yang mudah dipahami.

2.2.7. Algoritma Genetika

Algoritma genetika merupakan algoritma yang dikembangkan oleh John Holland dari University of Michigan. Algoritma genetika merupakan algoritma yang bekerja berdasarkan pada prinsip seleksi alam dan teknik evolusi. Algoritma genetika terdiri atas penelitian dan aplikasi yang memperhatikan analogi-analogi ilmiah, analisis matematika, dan perhitungan komputasi untuk mencari solusi atas masalah yang terjadi dari berbagai latar belakang bidang studi. Algoritma genetika memiliki kemampuan untuk melakukan proses peniruan dalam mekanisme evolusi alam. Dalam pencarian populasi dilakukan dengan memunculkan kemungkinan-kemungkinan solusi agar dapat mendapatkan solusi terbaik. Dari suatu populasi awal ,individu-individu melakukan beberapa operasi genetic tertentu. Kemudian daya tahan setiap individu (*fitness*) tersebut dinilai untuk menentukan apakah individu tersebut bisa dipertahankan untuk generasi selanjutnya atau tidak (Ginting, 2009).



Gambar 2.5 Ilustrasi Struktur pada Algoritma Genetika

a. Genotype (gen), merupakan sebuah nilai yang menyatakan satuan dasar yang membentuk suatu arti tertentu dalam satu kesatuan gen yang dinamakan kromosom.
b. Allele, merupakan nilai dari gen.
c. Kromosom, merupakan gabungan gen-gen yang membentuk nilai tertentu.
d. Individu, menyatakan satu nilai atau keadaan yang menyatakan salah satu solusi yang mungkin dari permasalahan yang diangkat.
e. Populasi, merupakan sekumpulan individu yang akan diproses bersama dalam satu siklus proses evolusi.
f. Generasi, menyatakan satu siklus proses evolusi atau satu iterasi didalam algoritma genetika.

Dalam teori genetik, gen menjadi operator pembawa sifat individu pada kromosom. Spesifikasi bentuk susunan gen pada sepanjang kromosom tersebut dinamakan *Schemata*. Susunan genetik dari individu tersebut akan mengalami perubahan dan perubahan tersebut terjadi secara bertahap dari suatu generasi ke generasi berikutnya. Perubahan susunan genetic pada individu dipengaruhi oleh sifat-sifat proses genetik yang terdiri dari:

a. Reproduksi
   Merupakan proses terjadinya peniruan terhadap individu induk oleh individu lain berdasarkan besarnya kemampuan individu tersebut untuk bertahan hidup (*fitness*).
b. Pindah Silang (*Crossover*)
   Dalam pindah silang, individu terlebih dahulu mencari pasangannya secara random lalu baru dilakukan pindah silang, terdapat beberapa macam teknik pindah silang yang dapat digunakan.
c. Mutasi (*mutation*)
   Mutasi merupakan perubahan bagian kromosom yang disebabkan oleh aberasi kromosom (penyimpangan kromosom) perbuahan bagian kromosom yang terjadi seperti inversi, tranlokasi, duplikasi dan terdapat beberapa perubahan lainnya.

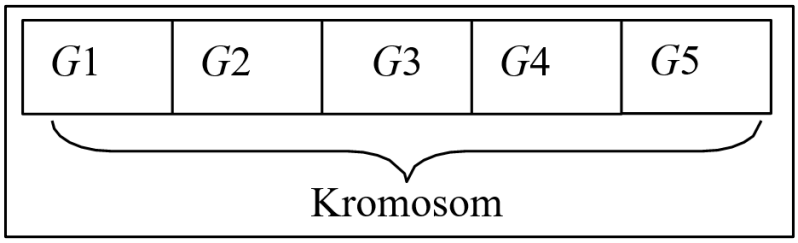
Ketiga sifat proses genetik ini diterapkan pada suatu populasi, sehingga individu tersebut melakukan perubahan atas mekanisme yang terjadi dengan tetap melakukan tugas-tugas dalam sistem dengan baik.

Proses selanjutnya adalah dengan melakukan seleksi untuk mendapatkan susunan genetic yang terbaik sebagai generasi terpilih. Maka dari itu sangat penting dalam menentukan perubahan posisi pada keseluruhan populasi sehingga perlu diperhatikan posisi ‘siapa’ mengganti ‘siapa’. Dengan begitu peartanyaan yang muncul adalah bagaimana mengetahui bahwa bagian-bagian kromosom yang berisi kode-kode genetic akan menghasilkan perubahan yang diinginkan dalam lingkungan yang dinamis. Salah satu alternatif yang dilakukan dengan melakukan perhitungan nilai *fitness* (Ginting, 2009).

2.2.8. Komponen Utama Algoritma Genetika

a. Teknik Pengkodean

Teknik pengkodean disini meliputi pengkodean gen dari kromosom. Satu gen biasanya akan mewakili satu variable. Gen dapat direpresentasikan dalam bentuk: string bit, pohon, array, bilanan real, daftar aturan, elemen permutasi, elemen program, atau reprentasi lainnya yang dapat diimplementasikan untuk operator genetika.Dalam permasalagan *flowshop* urutan dari pekerjaan (*job*) digunakan sebagai solusi kromosom. Jumlah gen dalam kromosom sama dengan jumlah pekerjaan yang akan diproses. Jika ada 10 pekerjaan diproses, p1 = [3, 5, 8, 7, 9, 6, 4, 2, 1, 10] dapat diambil sebagai salah satu kromosom dalam populasi. Nomor di kromosom menunjukkan ID pekerjaan yang sedang dikerjakan, dan posisi di kromosom menunjukkan urutan pemrosesan pekerjaan Wei et al. (2018). Kromosom akan diinterpretasikan pada Gambar 3.2 berikut.



Gambar 2.6. Interpretasi kromosom

Dimana G1,G2,…,G5 = job

b. Membangkitkan Populasi Awal

Membangkitkan populasi awal adalah proses membangkitkan sejumlah individu secara acak atau melalui prosedur tertentu. Ukuran untuk populasi tergantung pada masalah yang akan diselesaikan dan jenis operator genetika yang akan diimplementasikan. setelah ukuran populasi ditentukan, kemudian dilakukan pembangkitan populasi awal. Syarat-syarat yang harus dipenuhi untuk menunjukkan suatu solusi harus benar-benar diperhatikan dalam pembangkitan setiap individunya. Dalam permasalahan *flowshop* ini pembangkitan populasi awal dilakukan secara acak (Dai, Tang, Giret, et al., 2013).

c. Evaluasi Nilai *Fitness*

Nilai fitness adalah nilai yang menyatakan baik tidaknya suatu solusi (individu). Nilai fitness ini yang dijadikan acuan dalam mencapai nilai optimal, Algoritma Genetika bertujuan mencari individu dengan nilai fitness yang terbaik. Jika permasalahannya adalah meminimumkan seperti pada permaslahan *flowshop* dengan kriteria makespan maka *fitness* yang baik yang memiliki nilai paling minimum.Setiap kromosom dalam populasi akan dihitung nilai objektif (makespan) dengan menggunakan persamaan (2.1) dan menggunakan cara yang ditunjukkan pada Tabel 2.2 berikut.

Table 2.2. Pencarian Nilai Makespan dengan *I job* dan *j* mesin

M / J	M1		M2		⋯	Mj	
	S_{ij}	E_{ij}	S_{ij}	E_{ij}	⋯	S_{ij}	E_{ij}
J1	0	$t_{1,1}$	$E_{1,1}$	$S_{1,2}+t_{1,2}$	⋯	$E_{1,j-1}$	$S_{1,j}+t_{1,j}$
J2	$t_{1,1}$	$S_{2,1}+t_{2,1}$	$\max\{E_{1,2}, E_{2,1}\}$	$S_{2,2}+t_{2,2}$	⋯	$\max\{E_{1,j}, E_{2,j-1}\}$	$S_{2,j}+t_{2,j}$
⋮	⋮	⋮	⋮	⋮	⋯	⋮	⋮
J_i	$t_{j-1,1}$	$S_{i,1}+t_{i,1}$	$\max\{E_{i-1,2}, E_{i,1}\}$	$S_{i,2}+t_{i,2}$	⋯	$\max\{E_{i-1,j}, E_{i,j-1}\}$	$S_{i,j}+t_{i,j}$

d. Seleksi

Seleksi kromosom dilakukan agar memperoleh kromosom-kromosom unggulan yang memiliki nilai fitness yang baik untuk dipindah silangkan (*crossover*). Seleksi kromosom pada model ini dilakukan dengan teknik roda rolet (*roulette wheel*) seperti pada penelitian Dai, Tang, Giret, et al. (2013).Proses seleksi bertujuan untuk memilih individu-individu yang akan dipilih untuk proses persilangan dan mutasi, sehingga akan diperoleh calon induk yang baik. Induk yang baik akan menghasilkan keturunan yang baik. Langkah pertama dalam seleksi yaitu pencarian nilai fitness. Masing-masing individu dalam suatu wadah seleksi akan menerima probabilitas reproduksi yang tergantung pada nilai objektif dirinya sendiri terhadap nilai objektif dari semua individu dalam wadah seleksi tersebut. Pada tahap seleksi ini akan menggunakan metode *Roullete wheel seletion.* Pada proses seleksi kromosom ini diterapkan prinsip elitisme. berikut tahap dari seleksi roda rolet:

1. Hitung nilai fitness dari masing-masing individu.
2. Hitung total fitness semua individu
3. Hitung probabilitas masing-masing individu
4. Dari probalitas tersebut, dihitung jatah masing-masing individu pada angka 1 sampai 100
5. Dibangkitkan bilangan random antara 1 sampai 100
6. Dari bilangan random yang dihasilkan, ditentukan individu mana yang terpilih dalam proses seleksi

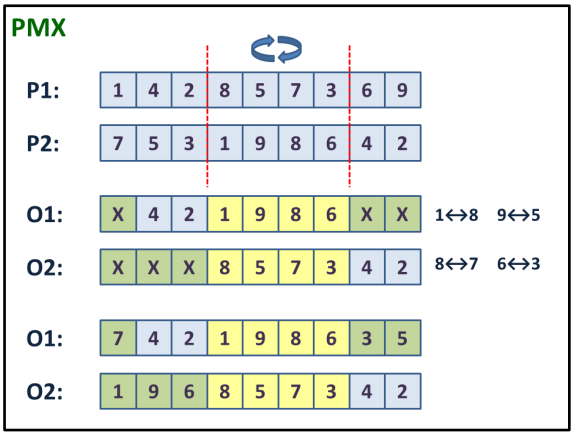
roda rolet akan diputar sebanyak n kali, dimana n adalah jumlah individual atau kromosom dalam populasi. Dari setiap putaran, kromosom yang terpilih akan menjadi *parents* bagi generasi berikutnya.

e. Pindah Silang (*Crossover*)

Crossover merupakan proses di dalam algoritma genetika yang bekerja untuk meng-gabungkan dua kromosom parent menjadi kromosom baru (offspring) pada suatu waktu. Sebuah kromosom yang mengarah pada solusi baik dapat diperoleh melalui proses crossover pada dua buah kromosom. *Crossover* memainkan peran penting dalam pertukaran informasi antar kromosom. Dengan *crossover* yang efektif dapat memberikan kombinasi solusi yang baik di kromosom dan mempercepat prosedur pencarian, tidak semua kromosom parent mengalami crossover, banyaknya pasangan induk yang akan mengalami persilangan akan ditentukan dengan nilai Pc yaitu probabilitas crossover. (Schaffer et al., 1989) menyebutkan nilai Pc optimal terletak pada range 0,75-0,95. pada permasalahan ini akan memakai PMX (*Partially Mapped Crossover*). Operator PMX merupakan operator crossover paling populer untuk mengoperasikan permutasi. Dengan cara memilih dua titik crossover pertama dan menukar sub-bagian kromosom di antara keduanya, dan kemudian mengisi kromosom secara parsial. Menurut Holland telah

dibuktikan bahwa PMX mampu memberikan skema terbaik dan mampu mereproduksi diri lebih baik dari operator crossover lainnya (Minmei et al., 2006).

Metode *crossover* ini dimulai dengan membangkitan bilangan random kemudian jika bilangan random (R) lebih kecil dari parameter pindah silang (P_ps) atau R < P_ps, maka akan dilakukan pindah silang terhadap kedua kromosom induk tersebut. Namun jika pembangkitan bilangan random (R) lebih besar dari parameter pindah silang (P_ps) atau R > P_ps, maka tidak terjadi pindah silang terhadap kedua kromosom induk ketika dua kromosom induk mengalami *crossover* maka dilakukan pengambilan dua buah angka random antara 1 dan banyak job dalam string Pamungkas (2002). Misalkan crossover diberlakukan pada dua jadwal P1 dan P2 berikut ini, angka random yang muncul adalah 4 dan 8, maka:



Gambar 2.7. PMX *Crossover*

f. Mutasi (*mutatiom*)

Mutasi merupakan perubahan bagian kromosom yang disebabkan oleh aberasi kromosom (penyimpangan kromosom) perbuahan bagian kromosom yang terjadi seperti inversi, tranlokasi, duplikasi dan terdapat beberapa perubahan lainnya. Mutasi dimaksudkan untuk memunculkan individu baru yang berbeda dengan individu yang sudah ada. Probabilitas mutasi (Pm) akan menentukan kromosom mana yang akan mengalami perubahan gen, semakin besar nilai probabilitas mutasi maka semakin banyak kromosom dalam populasi yang akan mengalami mutasi Nilai Pm yang sering pakai pada implementasi Algoritma Genetika berada pada range 0,001 dan 0,05 (Davis, 1991 dalam Ismail & Irhamah, 2008).

Proses mutasi akan dipilih secara random dan gen pada site tersebut akan diubah nilainya. Angka random akan dibangkitkan dengan batasan 0 sampai 1. Jika angka random (R) tersebut lebih kecil dari parameter mutasi (Pm) maka akan terjadi mutasi, dan jika angka random (R) tersebut lebih besar dari parameter mutasi (Pm) maka tidak terjadi mutasi. Dalam permaslahan ini akan menerapakan metode *swap mutation* seperti pada penelitian Minmei et al. (2006), Setiya Widodo et al. (2014). Bila terpilih terkena operator mutasi, langkah berikutnya melakukan swap mutation yaitu dengan mengambil dua angka random misal d1 dan d2. Kemudian job d1 dan job d2 pada individu tersebut bertukar posisi. Misal jadwal A = 3 1 2 4 5, d1 = 3 dan d2 = 4. Maka job 3 bertukar tempat dengan job 4 menjadi A’ = 4 1 2 3 5.

g. Elitisme

Elitisme adalah proses untuk menyalin dan menempatkan individu atau kromosom-kromosom terbaik pada populasi baru dengan tujuan agar individu-individu terbaik ini tidak hilang atau rusak karena proses pindah silang atau mutasi sehingga individu-individu terbaik akan tetap muncul di populasi berikutnya (Azmi et al., n.d.).

h. Kriteria Pemberhentian

Mengulangi proses dari langkah c dan langkah selanjutnya sampai proses pencarian berhenti ketika iterasi yang ditentukan terpenuhi.

2.2.9. *Simulated Annealing*

Menurut Santosa & Willy (2011) yang dikutip dari Firdaus et al. (2015) *Simulated Annealing* Merupakan suatu algoritma yang bekerja berdasarkan analogi dari proses annealing proses annealing sendiri merupakan suatu proses dalam pembentukan kristal dalam suatu materi. Proses dari SA diawali dengan dipilihnya suatu solusi awal, yang mengidentifikasikan kondisi materi sebelum proses dimulai. Gerakan bebas dari atom pada suatu materi, direpresentasikan dalam bentuk modifikasi terhadap solusi awal/solusi sementara agar dapat diproses oleh SA. Pada saat awal proses SA, ketika parameter suhu (T) diatur tinggi, solusi sementara yang sudah ada diperbolehkan untuk mengalami modifikasi dengan bebas, Kebebasan dari modifikasi tersebut secara relatif diukur berdasarkan nilai fungsi tertentu yang berguna untuk mengevaluasi seberapa optimal solusi sementara yang dihasilkan. Bila nilai fungsi dari hasil evaluasi solusi yang dimodifikasi ini membaik, solusi hasil modifikasi tersebut akan digunakan sebagai solusi selanjutnya. Tetapi ketika nilai fungsi evaluasi hasil modifikasi memburuk, ketika temperatur annealing masih tinggi, solusi yang lebih buruk ini masih mungkin diterima. Dalam tahapan selanjutnya saat temperatur sedikit demi sedikit dikurangi, maka kemungkinan untuk menerima langkah modifikasi yang tidak memperbaiki nilai fungsi evaluasi semakin berkurang. Sehingga kebebasan untuk memodifikasi solusi semakin berkurang, hingga akhirnya diperoleh solusi yang mendekati solusi optimal.

Metode SA mengikuti proses pendinginan secara perlahan dari baja / metal yang mendidih untuk mencapai nilai minimum fungsi dalam permasalahan minimasi. Proses pendinginan ini ditiru dengan cara menentukan parameter yang serupa dengan suhu lalu mengontrolnya dengan menggunakan mengunakan konsep distribusi probabilitas Boltzmann. Distribusi probabilitas Boltzmann sendiri menyatakan bahwa energi (E) dari suatu system dalam keseimbangan panas pada suhu T terdistribusi secara probalistik mengikuti rumus

$$P(E) = e^{-E/kT} \quad \text{.....(2.2)}$$

Dimana P(E) merupakan peluang mencapai tingkat energi E, T dan k adalah konstanta Boltzmann. Pada persamaan ini menjelaskan bahwa apabila proses pencairan solusi mengikuti distribusi probabilitas Boltzmann konvergensi dari algoritma SA dapat diatur dengan menggunakan temperatur T. Dalam minimasi fungsi, misalkan solusi yang sekarang adalah x dan nilai fungsinya f(x), mirip dengan status energi pada system thermodinamika, energi Ei pada status xi adalah

$$E_i = f_i = f(x_i) \quad \text{.....(2.3)}$$

Menurut kriteria Meropolis, probabilistic titik solusi berikutnya adalah xi+1 bergantung pada perbedaan status energi atau fungsi tujuan dia dua titik (status) diberikan oleh

$$P[E_{i+1}] = \min\{1, e^{-\Delta E/kT}\} \quad \text{.....(2.4)}$$

Dimana

$\Delta E=E_{i+1}-E_i=\Delta f=f_{i+1}-f_i=f(x_{i+1})-f(x_i)$……..……(2.5)

Titik baru bisa ditemukan dengan menerapkan distribusi probabilitas Boltzmann tersebut. Untuk faktor Boltzmann, k, bisa diberi nilai 1. Jika ΔE 0, maka P[Ei+1] = 1 sehingga titik xi+1 selalu diterima. Ini adalah pilihan yang masuk akal dalam konteks minimasi fungsi, yaitu jika f(xi+1) f(xi) maka xi+1 pasti diterima. Dilain pihak jika ΔE 0, maka nilai f(xi+1) akan lebih besar (lebih buruk) dari f(xi). dalam optimasi konvensional jika ini terjadi maka xi+1 akan ditolak. Tetapi dalam SA ini masih diterima dengan Probabilitas.

$P[E_{i+1}] = e^{-\Delta E/kT}$ ……...(2.6)

Dari persamaan diatas bisa dilihat bahwa pada temperatur T tinggi maka peluang menerima xi+1 dengan ΔE yang lebih besar akan besar. Dengan menurunnya temperatur T probabilitas untuk menerima titik xi+1 yang lebih buruk dari titik sebelumya akan mengecil. Sehingga jika temperatur semakin rendah (semakin dekat ke titik optimal), peluang suatu solusi xi+1 dengan nilai f lebih besar dibanding pada titik xi akan semakin kecil.

Dalam proses algoritma genetika-simulated annealing, individu baik yang dihasilkan oleh GA dikirim ke SA untuk perbaikan. SA dapat menghindari jatuh ke optimal lokal karena dapat menemukan beberapa kemungkinan solusi. Namun, efisiensi pencarian SA tidak tinggi. Oleh karena itu, harus ada beberapa parameter yang terkait dengan SA dipelajari, termasuk struktur *neighborhood*, temperature, laju pendinginan, dan kriteria pemberhentian. Faktor-faktor ini memainkan peran penting dalam kinerja SA dan harus diterapkan dengan hati-hati berikut penjelasan faktor-faktor dari SA (Dai, Tang, Giret, et al., 2013):

1. Struktur *Neighborhood*

   Struktur Neighborhood secara langsung akan mempengaruhi efisiensi pencarian lokal. Dalam penlitian ini akan mengadopsi dua Struktur Neighborhood yaitu Two points exchange dan Insertion (Wei et al., 2018) berikut penjelasannya:

   a. Two points exchange. Tukar gen di dua posisi yang dihasilkan secara acak. Misalnya, pengkodean kromosom adalah “154837629. Posisi gen yang dihasilkan secara acak adalah 3 dan 6. Kemudian, kromosom baru yang diproduksi dengan pertukaran gen pada kedua posisi tersebut adalah “15783429”.
   b. Penyisipan. Pilih dua posisi gen secara acak. Kemudian, gen dengan jumlah posisi yang lebih besar dimasukkan ke posisi gen sebelumnya dengan jumlah posisi yang lebih kecil. Misalnya, kode satu kromosomnya adalah "15783429", dua nomor posisi yang dipilih secara acak adalah 2 dan 4, dan yang baru kromosom yang diperoleh setelah operasi penyisipan adalah "18573429"

2. Temperatur

   Suhu awal SA harus diatur dengan benar karena suhu awal terlalu tinggi akan menyebabkan pemborosan waktu perhitungan, dan suhu awal yang terlalu kecil menyebabkan efisiensi pencarian berkurang.

3. Laju pendingan

   Laju pendinginan merupakan fungsi yang menganalogikan seberapa cepat pencaipaian solusi akhir dan digunakan dalam proses penurunan temperatur. Berikut persamaannya (Firdaus et al., 2015) :

   $T_{baru}=T_{awal}$ x c……...……(2.6)

Dimana T_{awal} = temperatur awal
T_{akhir}= temperatur akhir
c = faktor reduksi suhu (0 < c < 1)

4. Kriteria pemberhentian

Di SA kriteria pemberhentian tergantung pada parameter-parameter yang dipakai yaitu meliputi jumlah iterasi dan temperatur dalam pemberhentian siklus saat penurunan suhu apabila siklus sudah mencapai batas iterasi maka penurunan suhu dilakukan begitu juga pada parameter temperatur apabila temperatur memiliki nilai yang lebih kecil dari kriteria pemberhentian suhu maka kriteria pemberhentian terpenuhi (Firdaus et al., 2015).

2.2.10. Proses *Simulated Annealing*

Menurut Santosa & Willy (2011) yang dikutip dari Firdaus et al. (2015), Algoritma SA dapat dijelaskan secara ringkas sebagai berikut. Algoritma Simulated Annealing dimulai dengan suatu vektor solusi x1 (iterasi i = 1) dan temperature T dengan nilai yang cukup tinggi. Bangkitkan vector solusi baru secara random yang dekat dari titik sekarang dan hitung perbedaan nilai fungsi tujuannya sama dengan persamaan (2.4) apabila fi+1 lebih kecil dari fi atau Δf memiliki nilai negatif, terima titik fi+1 sebagai titik solusi baru. Sebaliknya, jika nilai positif, probabilitas menerima xi+1 sebagai solusi baru adalah $e^{-\Delta f/kT}$. Untuk diterima atau tidaknya, perlu dicari pembanding terhadap nilai probabilitas ini. Selanjutnya perlu membangkitkan bilangan random antara 0 sampai 1. Apabila nilai random yang dibangkitkan ini lebih kecil dari nilai $e^{-\Delta f/kT}$, maka terima titik xi+1, Sebaliknya jika nilai random lebih besar maka tolak xi+1. Pada tahap ini kita telah melakukan satu iterasi dari algoritma SA. apabila titik xi+1 tidak diterima, maka dilanjutkan dengan proses pembangkitan nilai baru xi+1 secara random dalam area yang berdekatan dengan titik sekarang xi dalam batas – batas tertentu, lalu mengevaluasi nilai fungsi tujuan fi+1, dan memutuskan untuk menerima xi+1 sebagi titik baru, berdasarkan kriteria metropolis P[Ei+1] = $e^{-\Delta f/kT}$ Untuk mensimulasikan pencapaian equilibrium thermal pada setiap temperatur tertentu T. apabila jumlah titik xi+1 yang diuji pada sembarang temperature T melebihi nilai k (banyaknya iterasi), temperature T dikurangi dengan proporsi tertentu yaitu c (0 < c < 1) dan seluruh proses diulang lagi. Prosedur ini diasumsikan akan mengalami konvergensi ketika temperature T yang dicapai cukup kecil atau jika perubahan nilai fungsi tujuan (Δf)sudah sangat kecil.

Pilihan nilai awal temperature T, jumlah iterasi k sebelum mengurangi temperatur, dan faktor pengurangan temperatur c adalah parameter – parameter penting dalam keberhasilan pelaksanaan algoritma SA. Apabila nilai awal temperatur terlalu tinggi, maka akan menyebabakan lebih banyak pengurangan temperature untuk konvergen. Sebaliknya jika nilai awal ini terlalu kecil proses pencairan ini mungkin kurang sempurna sehingga ada titik-titik potensial terlewati. Faktor pereduksi temperatur c memainkan peran yang sama pentingnya. Apabila terlalu besar (0.8 atau 0.9) akan menyebabkan banyaknya langkah komputasi. Sebaliknya, jika terlalu kecil nilai c (misal 0.1 atau 0.2) bisa menyebabkan terlalu cepatnya penurunan temperatur sehingga memunkinkan banyak titik-titik potensial akan terlewati.

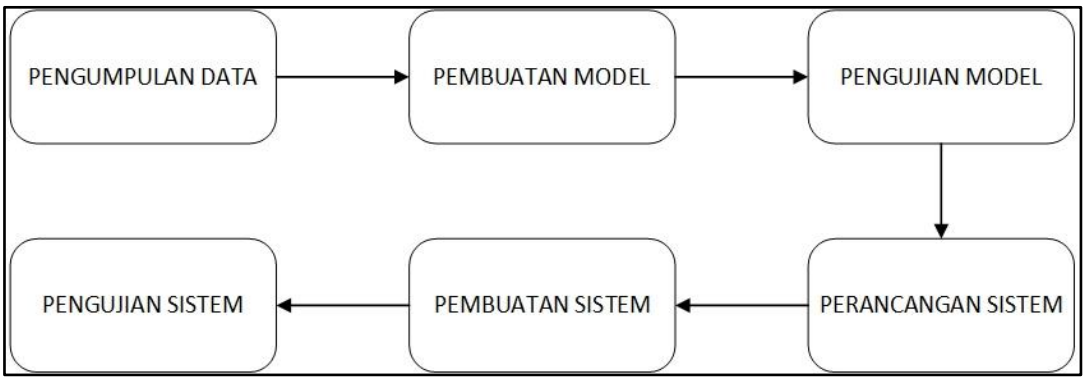
Begitu juga apabila jumlah dari iterasi k terlalu besar maka akan membantu mencapai keseimbangan termal pada setiap temperatur tetapi jumlah komputasi juga akan bertambah. Jika jumlah k sedikit, maka bisa menyebabkan konvergensi yang terlalu cepat atau menuju kesolusi tetapi tidak mengdapatkan hasil yang optimal (local optimum). Meskipun begitu sayangnya tidak ada nilai T, k, dan c yang unik yang akan sesuai untuk semua problem. Tetapi

ada beberapa cara agar mendapatkan nilai yang cukup baik. Misalnya untuk T bisa dipilih dari nilai rata-rata dari fungsi tujuan yang dihitung pada sejumlah solusi awal yang dipilih secara random. Jumlah iterasi k bisa dipilih antara 50 dan 100 berdasarkan pada akurasi solusi yang diinginkan. Faktor pereduksi temperatur c bisa dipilih antara 0.4 dan 0.6 untuk strategi pengurangan temperatur yang masuk akal (cooling schedule) (Firdaus et al., 2015). Dalam SA, perlu diperhatikan adalah adanya langkah khusus untuk keluar dari local optimum. Langkah tersebut berupa penerimaan suatu titik xi+1 dengan peluang walaupun nilai fungsi pada titik ini tidak lebih baik dari titik sebelumnya xi. Proses ini dilakukan dengan harapan pada langkah selanjutnya akan dicapai suatu titik dengan nilai fungsi yang lebih baik lagi. Jika langkah khusus ini tidak ada, maka SA akan seperti teknik optimasi konvensional biasannya yang sering terjebak pada local optimum. Hal-hal yang bisa diperhatikan dalam SA antara lain, solusi akhir tidak dipengaruhi solusi awal yang dimasukkan. Tetapi jika titik awal yang dimasukkan tidak cukup baik maka diperlukan waktu lebih lama untuk mencapai solusi akhirnya kemudian konvergensi tidak dipengaruhi konveksitas dari ruang pencarian yang (*fleasible space*) (Firdaus et al., 2015).

BAB III
METODOLOGI PENELITIAN

3.1. Metode Penelitian

Metode penelitian yang digunakan penulis di dalam penelitian ini adalah penelitian kuantitatif. Sedangkan jenis penelitian yang akan dilakukan penulis dalam penelitian ini adalah penelitian implementatif, penelitian ini memuat perancangan dan pengembangan. Perancangan yang dilakukan adalah membuat sistem penjadwalan produksi yang dibuat menggunakan bahasa pemrograman php yang akan ditampilkan dalam bentuk aplikasi berbasis web yang dapat dibuka menggunakan browser, pada penelitian ini akan menggunakana algoritma genetika dan *simulated annealing* untuk menyelesaikan permasalahan penjadwalan *flowshop.* Pemilihan algoritma genetika dalam menyelesaikan permasalahan penjadwalan *flowshop* karena algoritma tersebut memiliki kemampuan pencarian global yang baik dan dapat memberikan hasil solusi yang optimal/mendekati optimal dalam waktu yang cepat. Dengan ditambahkannya metode *simulated annealing* setelah algoritma genetika bertujuan agar kovergensi premature pada algoritma genetika dapat teratasi sehingga dapat menemukan solusi-solusi yang lebih baik.



Gambar 3.1. Tahapan Penelitian

3.2. Pengumpulan Data

Data yang akan digunakan dalam penelitian ini adalah data dengan jenis data primer, data primer merupakan sumber data yang langsung memberikan data kepada pengumpul data. Sumber data primer didapatkan melalui kegiatan wawancara dengan subjek penelitian dan dengan observasi atau pengamatan langsung di lapangan. Data yang diambil merupakan data dari waktu pemrosesan tiap pekerjaan pada tiap mesin di industri pembuatan pakaian Hafki Project. Hafki Project terletak di Boyolali, Jawa Tengah. Pengumpulan data pada penelitian ini dilakukan dengan cara menghitung waktu pemrosesan tiap pekerjaan pada tiap mesin dengan bertanya langsung kepada pemilik Hafki Project. Data yang diperoleh berupa data waktu proses tiap pekerjaan pada tiap mesin dalam satuan waktu menit, dalam prosesnya terdapat 4 jenis proses yaitu proses potong, jahit, sablon/bordir, dan packing dan tiap proses menggunakan mesin yang berbeda, data tersebut disajikan dalam bentuk tabel 3.1.

Nilai dari waktu proses tiap mesin pada tabel 3.1. sudah dilakukan proses perhitungan yang melibatkan beberapa faktor yaitu jumlah pesanan dan tipe pesanan berikut data jenis pakaian (satuan) dan waktu prosesnya pada tiap mesin.

1. Jenis Pakaian :
    a. Kaos
    b. Polo

c. Jaket
   - Hoodie
   - Parasit
   - Zip Hoodie

d. Kemeja
   - Panjang
   - Pendek

e. Celana Olahraga

2. Proses Pembuatan

a. Proses Potong

Tabel 3.1. Proses Potong

No	Jenis	Sub Jenis	Waktu (menit)
1.	Kaos	-	3
2.	Polo	-	3
3.	Jaket	Hoodie	5
		Parasit	8
		Zip Hoodie	8
4.	Kemeja	Panjang	10
		Pendek	8
5.	Celana Olahraga	-	5

b. Proses Jahit

Tabel 3.2. Proses Jahit

No	Jenis	Sub Jenis	Waktu (menit)
1.	Kaos	-	15
2.	Polo	-	25
3.	Jaket	Hoodie	30
		Parasit	45
		Zip Hoodie	45
4.	Kemeja	Panjang	60
		Pendek	50
5.	Celana Olahraga	-	30

c. Proses Sablon / Bordir

Pada proses ini semua jenis pakaian sama

Tabel 3.3. Proses Sablon / Bordir

No	Jenis	Sub Jenis	Waktu (menit)	Tambah Warna (1- 6)
1.	Sablon	1 Sisi	5	Tiap tambah 1 warna waktu ditambah 3 dikali dengan banyak sisi yang ditambah
		2 Sisi	10	Tiap tambah 1 warna waktu ditambah 3 dikali dengan banyak sisi yang ditambah

Tabel 3.3. Proses Sablon / Bordir

		3 Sisi	15	Tiap tambah 1 warna waktu ditambah 3 dikali dengan banyak sisi yang ditambah
		4 Sisi	20	Tiap tambah 1 warna waktu ditambah 3 dikali dengan banyak sisi yang ditambah
2.	Bordir	Kecil (a6)	2.5 (satu sisi)	-
		Sedang (a5)	5 (satu sisi)	-
		Besar (a4)	10 (satu sisi)	-

d. Proses Packing

Pada proses packing semua jenis pakaian memiliki waktu yang sama yaitu 3 menit tiap pakaian.

Tabel 3.4. Data Pekerjaan dan Waktu Proses Tiap Mesin

No	Mesin / Job	Mesin 1 (potong)	Mesin 2 (jahit)	Mesin 3 (sablon/bordir)	Mesin 4 (packing)
1.	Job 1	45	225	480	45
2.	Job 2	60	300	820	60
3.	Job 3	153	765	1122	153
4.	Job 4	36	300	150	36
5.	Job 5	63	315	336	63
6.	Job 6	125	750	250	75
7.	Job 7	18	90	246	18
8.	Job 8	66	330	770	66
9.	Job 9	36	180	672	36
10.	Job 10	300	1500	1000	300
11.	Job 11	234	1170	1716	234
12.	Job 12	30	180	0	18
13.	Job 13	20	120	35	6
14.	Job 14	78	650	416	78
15.	Job 15	270	1350	1440	270

3.3. Analisis Kebutuhan

Analisis kebutuhan dapat didefinisikan sebuah proses yang dilakukan untuk mencari dan munculkan perbedaan antara tujuan ideal yang ada dan ekspektasi tujuan yang kita harapkan Briggs et al. (1991). Definisi lain dari analisis kebutuhan adalah sebuah proses pengumpulan informasi mengenai ketidakseimbangan yang ada dan menentukan prioritas untuk dipecahkan Sanjaya (2008).Dalam subbab ini menjelaskan tentang apa aja yang dibutuhkan dan dilakukan oleh sistem yang dibuat. Analisis kebutuhan ini dilakukan untuk mengumpulkan data dan digunakan untuk pengambilan keputusan proses apa saja yang akan terlibat dan menentukan tujuan dari sistem yang dibuat. Terdapat dua jenis kebutuhan, yaitu kebutuhan fungsional dan kebutuhan non fungsional.

3.3.1. Kebutuhan Fungsional

Kebutuhan fungsional adalah informasi mengenai apa saja yang harus ada atau yang akan dilakukan sistem tersebut. Beberapa kebutuhan fungsional yang ada di dalam sistem prediksi ini antara lain adalah:

a. Sistem dapat menerima input data yang ada.
b. Sistem dapat melakukan proses penjadwalan berdasarkan data yang diinput oleh user.
c. Sistem dapat menampilkan jadwal yang dihasilkan.

3.3.2. Kebutuhan Nonfungsional

Kebutuhan non fungsional adalah kebutuhan yang melibatkan perilaku sistem. Dalam subbab ini akan menjelaskan spefisikasi perangkat yang digunakan untuk membuat sistem, seperti spefisikasi perangkat keras (hardware) ataupun perangkat lunak (software) yang digunakan.

a. Analisis perangkat keras (hardware) Perangkat keras atau hardware yang digunakan untuk membuat sistem prediksi. Spesifikasi hardware dapat dilihat pada tabel dibawah ini

Table 3.5. Spesifikasi Perangkat Keras

No	Perangkat Keras	Keterangan
1.	Processor	Ryzen3 3200U
2.	RAM	8GB
3.	Storage	1TB
4.	Graphic Card	AMD Radeon RX Vega 3 iGPU
5.	Perangkat Input dan Output	Keyboard, Mouse, Monitor
6.	Koneksi internet	Wifi

b. Analisis kebutuhan perangkat lunak (software) Perangkat lunak atau software yang digunakan untuk membuat sistem prediksi ini.

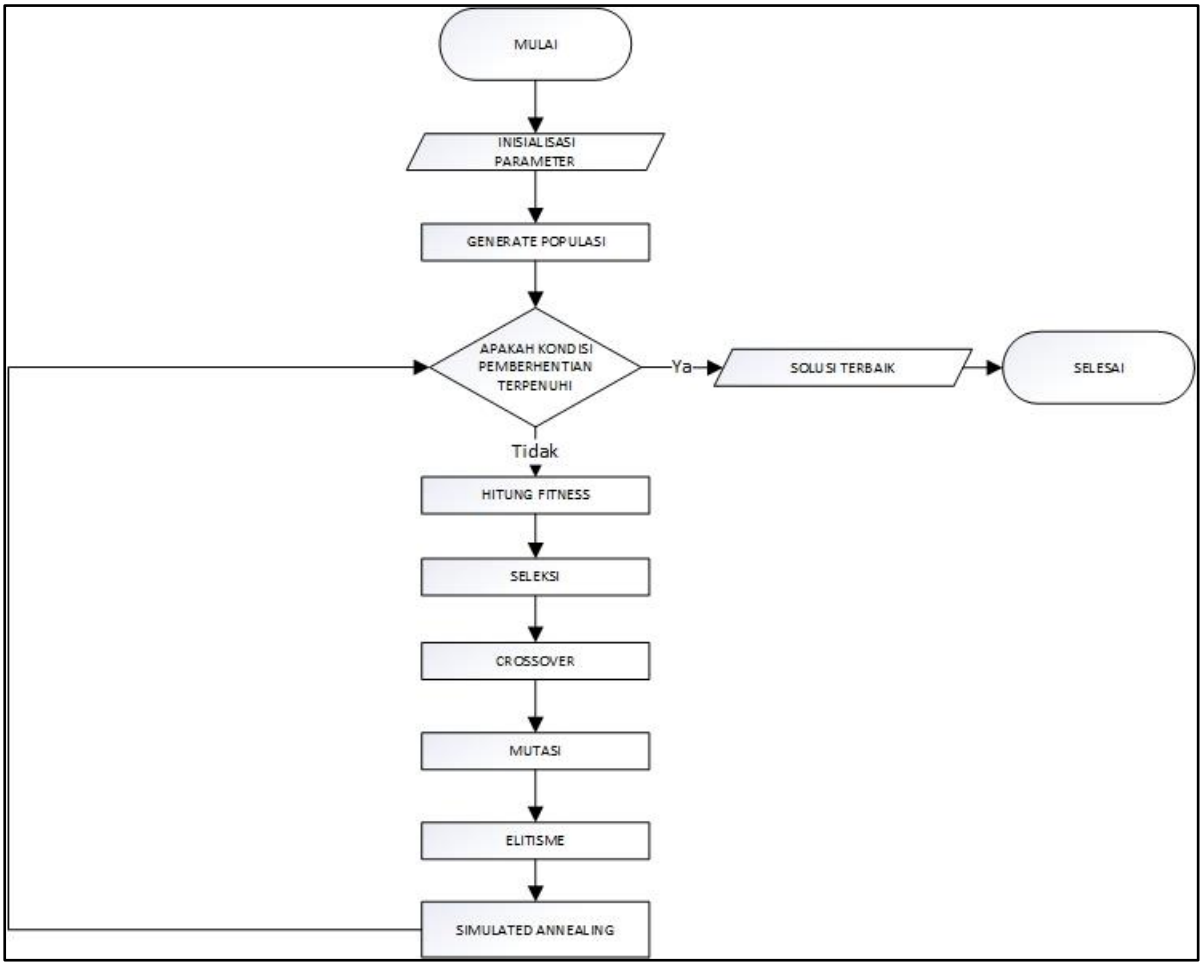
Table 3.6. Spesifikasi Kebutuhan Perangkat Lunak

No	Perangkat Lunak	Keterangan
1.	Operation System	Windows 10 64bit
2.	PHP versi 7.4.8	Bahasa Pemrograman
3.	Visual Studio Code	Text Editor
4.	Google Chrome	*Web Browser*
5.	Microsoft Visio	*Software* untuk membuat diagram
6.	XAMPP	*Server* lokal
7.	Adobe XD	*Software* untuk mendesain UI

3.4. Pembuatan Model

3.4.1. *Hybrid* Algoritma Genetika dan *Simulated Annealing*

Flowchart dari *Hybrid* Algoritma Genetika dan *Simulated Annealing* terlihat pada Gambar 3.11.



Gambar 3.2. *Flowchart Hybrid* Algoritma Genetika dan *Simultaed Annealing*

Berikut detail proses dari Algoritma Genetika dalam model yang dibuat dengan data yang dipakai adalah data di Tabel 3.4 :

a. Inisialisasi Parameter

Terdapat parameter dasar dalam algoritma genetika yang perlu ditentukan terlebih dahulu yaitu probabilitas kawin silang (crossover), probabilitas mutasi, dan jumlah iterasi.

b. Membangkitkan Populasi

Populasi awal dibangkitkan secara random dengan jumlah kromosom 5 kromosom, dimana setiap kromosom berisikan gen dengan jumlah sesuai banyaknya pekerjaan yang ada pada penjadwalan, pada contoh ini menggunakan 15 pekerjaan, sehingga banyaknya gen pada kromosom adalah 15.

Kromosom 1 : 2-11-14-3-15-12-8-6-7-9-5-4-13-10-1
Kromosom 2 : 3-14-9-7-10-11-4-8-1-15-5-13-2-6-12
Kromosom 3 : 5-4-3-8-6-10-1-7-13-15-14-9-12-11-2
Kromosom 4 : 12-2-15-14-5-3-7-11-1-6-8-13-9-10-4
Kromosom 5 : 8-10-11-6-5-2-1-15-4-7-12-9-3-14-13

c. Menghitung nilai objektif (*makespan*)

Setelah populasi awal sudah dibangkitkan lalu dilakukan perhitungan nilai objektif dengan cara menghitung nilai *makespan* tiap masing-masing kromosom, hasil dari nilai objektif ini akan dijadikan nilai *fitness* tiap kromosom. Perhitungan makespan tiap kromosom terlihat pada table dibawah ini.

Tabel 3.7. Perhitungan *Makespan* Kromosom 1 Populasi Awal

No	Job	Mesin 1		Mesin 2		Mesin 3		Mesin 4	
		S	E	S	E	S	E	S	E
1.	Job 2	0	45	45	270	270	750	750	795
2.	Job 11	45	279	279	1449	1449	3165	3165	3399
3.	Job 14	279	357	1449	1799	3165	2215	3399	3477
4.	Job 3	357	510	1799	2564	2564	3686	3686	3839
5.	Job 15	510	780	2564	3914	3914	5354	5354	5624
6.	Job 12	780	810	3914	4094	5354	5354	5624	5642
7.	Job 8	810	876	4094	4424	5354	6124	6124	6190
8.	Job 6	876	1001	4424	5174	6124	6374	6374	6449
9.	Job 7	1001	1019	5174	5264	6374	6620	6620	6638
10.	Job 9	1019	1055	5264	5444	6620	7292	7292	7328
11.	Job 5	1055	1118	5444	5759	7279	7615	7615	7678
12.	Job 4	1118	1154	5759	6059	7615	7765	7765	7801
13.	Job 13	1154	1174	6059	6179	7765	7800	7801	7807
14.	Job 10	1174	1474	6179	7679	7800	8800	8800	9100
15.	Job 1	1474	1519	7679	7904	8800	9280	9280	9325

Setelah dilakukan perhitungan pada kromosom 1, terlihat total waktu yang yang dihasilkan yaitu 9325 nilai total waktu tersebut yang dijadikan nilai objektif / *fitness* dari kromosom 1.

Tabel 3.8. Perhitungan *Makespan* Kromosom 2 Populasi Awal

No	Job	Mesin 1		Mesin 2		Mesin 3		Mesin 4	
		S	E	S	E	S	E	S	E
1.	Job 3	0	153	153	918	918	2040	2040	2193
2.	Job 14	153	231	918	1568	2040	2456	2456	2534
3.	Job 9	231	267	1568	1748	2456	3128	3128	3164
4.	Job 7	267	285	1748	1838	3128	3374	3374	3392
5.	Job 10	285	585	1838	3338	3374	4374	4374	4674
6.	Job 11	585	819	3338	4508	4508	6224	6224	6458
7.	Job 4	819	855	4508	4808	6224	6374	6458	6494
8.	Job 8	855	921	4808	5138	6374	7141	7141	7207
9.	Job 1	921	966	5138	5363	7141	7621	7621	7666
10.	Job 15	966	1236	5363	6713	7621	9061	9061	9331
11.	Job 5	1236	1299	6713	7028	9061	9397	9397	9460
12.	Job 13	1299	1319	7028	7148	9397	9432	9460	9466
13.	Job 2	1319	1379	7148	7448	9432	10252	10252	10312
14.	Job 6	1379	1504	7448	8198	10252	10502	10502	10577
15.	Job 12	1504	1534	8198	8378	10502	10502	10577	10595

Setelah dilakukan perhitungan pada kromosom 2, terlihat total waktu yang yang dihasilkan yaitu 10595 nilai total waktu tersebut yang dijadikan nilai objektif / *fitness* dari kromosom 2.

Tabel 3.9. Perhitungan *Makespan* Kromosom 3 Populasi Awal

No	Job	Mesin 1		Mesin 2		Mesin 3		Mesin 4	
		S	E	S	E	S	E	S	E
1.	Job 5	0	63	63	378	378	714	714	777
2.	Job 4	63	99	378	678	714	864	864	900
3.	Job 3	99	252	678	1443	1443	2565	2565	2718
4.	Job 8	252	318	1443	1773	2526	3296	3296	3362
5.	Job 6	318	443	1773	2523	3296	3546	3546	3621
6.	Job 10	443	743	2523	4023	4023	5023	5023	5323
7.	Job 1	743	788	4023	4248	5023	5503	5503	5548
8.	Job 7	788	806	4248	4338	5503	5749	5749	5767
9.	Job 13	806	826	4338	4458	5749	5784	5784	5790
10.	Job 15	826	1096	4458	5808	5808	7248	7248	7528
11.	Job 14	1096	1174	5808	6458	7248	7664	7664	7742
12.	Job 9	1174	1210	6458	6638	7664	8336	8336	8372
13.	Job 12	1210	1240	6638	6818	8336	8336	8372	8390
14.	Job 11	1240	1474	6818	7988	8336	10052	10052	10286
15.	Job 2	1474	1534	7988	8288	10052	10872	10872	10932

Setelah dilakukan perhitungan pada kromosom 3, terlihat total waktu yang yang dihasilkan yaitu 10932 nilai total waktu tersebut yang dijadikan nilai objektif / *fitness* dari kromosom 3.

Tabel 3.10. Perhitungan *Makespan* Kromosom 4 Populasi Awal

No	Job	Mesin 1		Mesin 2		Mesin 3		Mesin 4	
		S	E	S	E	S	E	S	E
1.	Job 12	0	30	30	210	210	210	210	228
2.	Job 2	30	90	210	510	510	1330	1330	1390
3.	Job 15	90	360	510	1860	1860	3300	3300	3570
4.	Job 14	360	438	1860	2510	3300	3716	3716	3794
5.	Job 5	438	501	2510	2825	3716	4052	4052	4115
6.	Job 3	501	654	2825	3590	4052	5174	5174	5327
7.	Job 7	654	672	3590	3680	5174	5420	5420	5438
8.	Job 11	672	906	3680	4850	5420	7136	7136	7370
9.	Job 1	906	951	4850	5075	7136	7616	7616	7661
10.	Job 6	951	1076	5075	5825	7616	7866	7866	7941
11.	Job 8	1076	1142	5825	6155	7866	8636	8636	8702
12.	Job 13	1142	1162	6155	6275	8636	8671	8702	8708
13.	Job 9	1162	1198	6275	6455	8671	9343	9343	9379
14.	Job 10	1198	1498	6455	7955	9343	10343	10343	10643
15.	Job 4	1498	1534	7955	8255	10343	10493	10643	10679

Setelah dilakukan perhitungan pada kromosom 4, terlihat total waktu yang yang dihasilkan yaitu 10679 nilai total waktu tersebut yang dijadikan nilai objektif / *fitness* dari kromosom 4.

Tabel 3.11. Perhitungan *Makespan* Kromosom 5 Populasi Awal

No	Job	Mesin 1		Mesin 2		Mesin 3		Mesin 4	
		S	E	S	E	S	E	S	E
1.	Job 8	0	66	66	396	396	1166	1166	1232
2.	Job 10	66	366	396	1896	1896	2896	2896	3196
3.	Job 11	366	600	1896	3066	3066	4782	4782	5016
4.	Job 6	600	725	3066	3816	4782	5032	5032	5107
5.	Job 5	725	788	3816	4131	5032	5368	5386	5431
6.	Job 2	788	848	4131	4431	5368	6188	6188	6248
7.	Job 1	848	893	4431	4656	6188	6668	6668	6713
8.	Job 15	893	1163	4656	6006	6668	8108	8108	8378
9.	Job 4	1163	1199	6006	6306	8108	8258	8378	8414
10.	Job 7	1199	1217	6306	6396	8258	8504	8504	8522
11.	Job 12	1217	1247	6396	6576	8504	8504	8522	8540
12.	Job 9	1247	1283	6576	6756	8504	9176	9176	9212
13.	Job 3	1283	1436	6756	7521	9176	10298	10298	10451
14.	Job 14	1436	1514	7521	8171	10298	10714	10724	10802
15.	Job 13	1514	1534	8171	8291	10714	10749	10802	10808

Setelah dilakukan perhitungan pada kromosom 5, terlihat total waktu yang yang dihasilkan yaitu 10808 nilai total waktu tersebut yang dijadikan nilai objektif / *fitness* dari kromosom 5.

Keterangan :

S = *Start* (mulai)

E = *End* (selesai)

Tabel 3.12. Hasil Perhitungan Nilai Objektif

No	Kromosom	Susunan Kromosom	Nilai Objektif (*Makespan*)
1.	1	2-11-14-3-15-12-8-6-7-9-5-4-13-10-1	9325
2.	2	3-14-9-7-10-11-4-8-1-15-5-13-2-6-12	10595
3.	3	5-4-3-8-6-10-1-7-13-15-14-9-12-11-2	10932
4.	4	12-2-15-14-5-3-7-11-1-6-8-13-9-10-4	10679
5.	5	8-10-11-6-5-2-1-15-4-7-12-9-3-14-13	10808

d. Seleksi

Sebelum melakukan proses seleksi dengan *roulette wheel* nilai objektif (*fitness*) tiap kromosom akan dilakukan proses invers yaitu dengan menggunakan nilai 1 dibagi dengan nilai objektif pada kromosom, kemudian nilai *fitness* yang dipakai pada tahap seleksi ini adalah hasil dari nilai invers tersebut hasil dari invers diterlihat seperti Tabel 3.13.

Tabel 3.13. Hasil Perhitungan *Invers*

No	Kromosom	Nilai *Makespan*	Nilai *Fitness*
1.	1	9325	0.00010723
2.	2	10595	0.00009438
3.	3	10932	0.00009147
4.	4	10679	0.00009364
5.	5	10808	0.00009252

Kemudian menghitung total *fitness* dari semua kromosom lalu dihitung probabilitas masing-masing individu, hasilnya terlihat pada Tabel 3.14.

Tabel 3.14. Perhitungan Probabilitas Kromosom

No	Kromosom	*Fitness*	Probabilitas	Range probabilitas
1.	1	0.00010723	0.00010723 / 0.00047924 × 100 = 22.375	0 - 22.375
2.	2	0.00009438	0.00009438 / 0.00047924 × 100 = 19.693	22.375 - 42.068
3.	3	0.00009147	0.00009147 / 0.00047924 × 100 = 19.086	42.068 - 61.607
4.	4	0.00009364	0.00009364 / 0.00047924 × 100 = 19.539	61.607 – 81.146
5.	5	0.00009252	0.00009252 / 0.00047924 × 100 = 19.305	81.146 – 100

Setelah diketahui probabilitas tiap kromosom lalu dilakukan pembangkitan bilangan secara acak antara 1-100 dari hasil bilangan acak tersebut akan menjadi acuan kromosom mana yang terpilih pada tahap seleksi, proses tersebut dilakukan sesuai dengan banyaknya kromosom, hasil pembangkitan bilangan acak adalah 15, 63, 26, 1, dan 33. Maka kromosom-kromosom pada populasi berubah menjadi :

Kromosom 1 = Kromosom 1
Kromosom 2 = Kromosom 4
Kromosom 3 = Kromosom 2
Kromosom 4 = Kromosom 1
Kromosom 5 = Kromosom 2

Tabel 3.15. Populasi Setelah Seleksi

No	Kromosom	Susunan Kromosom	Nilai Objektif (*Makespan*)
1.	1	2-11-14-3-15-12-8-6-7-9-5-4-13-10-1	9325
2.	2	12-2-15-14-5-3-7-11-1-6-8-13-9-10-4	10679
3.	3	3-14-9-7-10-11-4-8-1-15-5-13-2-6-12	10595
4.	4	2-11-14-3-15-12-8-6-7-9-5-4-13-10-1	9325
5.	5	3-14-9-7-10-11-4-8-1-15-5-13-2-6-12	10595

e. *Crossover*

Pada proses *crossover* langkah pertama yang dilakukan yaitu menentukan kromosom yang akan dijadikan induk yang nantinya akan dilakukan proes crossover, proses pemilihan induk dilakukan secara acak, hasil dari pemilihan pasangan induk yaitu kromosom 2 dengan kromosom 4, kromosom 3 dengan kromosom 1 dan kromosom 5 dengan kromosom 2. Setelah itu dilakukan pembangkitan bilangan acak antara 0-1 pada tiap pasangan kromosom lalu dibandingkan dengan Pc (*crossover rate*) jika bilangan acak < Pc maka *crossover* akan dilakukan apabila tidak maka nilai dari induk langsung diturunkan ke keturunan tanpa proses *crossover*. Pada kasus ini yang terjadi *crossover* merupakan kromosom 2 dan kromsom 4, berikut proses *crossover* dengan metode PMX.

Kromosom 2	12	2	15	14	5	3	7	11	1	6	8	13	9	10	4
Kromosom 4	2	11	14	3	15	12	8	6	7	9	5	4	13	10	1

Gambar 3.3. kromsom 2 dan kromosom 4

1. Tentukan dua titik secara random untuk membagi parent Area yang terletak antara dua titik pada kromosom orang tua (parents) disebut area pemetaan (*mapping*), misal angka acak 1-15 yang dibangkitkan adalah 3 dan 7 maka hasil *mapping* terlihat pada Gambar 3.4.

Kromosom 2	12	2	15	14	5	3	7	11	1	6	8	13	9	10	4
Kromosom 4	2	11	14	3	15	12	8	6	7	9	5	4	13	10	1

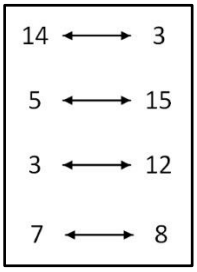
Gambar 3.4. Menentukan titik *crossover*

2. Kemudian tahap selanjutnya yaitu menukar gen-gen yang terdapat di area *mapping.*

12	2	15	3	15	12	8	11	1	6	8	13	9	10	4
2	11	14	14	5	3	7	6	7	9	5	4	13	10	1

Gambar 3.5. Hasil Menukar Gen

3. Tentukan relasi pemetaan pada area *mapping* antara dua *parent.* Dan berikut hasilnya:



Gambar 3.6. Relasi Pemetaan

4. Isi offspring dengan gen yang dicopy secara langsung dari parent secara berurutan dari kiri ke kanan. Jika gen tersebut sudah terdapat pada area mapping yang dicopykan sebelumnya, lakukan pengisian gen dengan melakukan pemetaan.

Child 1	14	2	5	3	15	12	8	11	1	6	7	13	9	10	4
Child 2	2	11	12	14	5	3	7	6	8	9	15	4	13	10	1

Gambar 3.7. Kromosom Hasil *Crossover*

Tabel 3.16. Populasi Setelah Proses Crossover

No	Kromosom	Susunan Kromosom	Nilai Objektif (*Makespan*)
1.	1	2-11-14-3-15-12-8-6-7-9-5-4-13-10-1	9325
2.	2	14-2-5-3-15-12-8-11-1-6-7-13-9-10-4	10403
3.	3	3-14-9-7-10-11-4-8-1-15-5-13-2-6-12	10595
4.	4	2-11-12-14-5-3-7-6-8-9-15-4-13-10-1	10127
5.	5	3-14-9-7-10-11-4-8-1-15-5-13-2-6-12	10595

f. Mutasi

Proses mutasi dimulai dengan membangkitkan angka acak 0-1 sejumlah kromosom yang ada. Jika angka acak tersebut lebih kecil dari parameter mutasi (Pm) maka akan terjadi mutasi, dan jika angka acak tersebut lebih besar dari parameter mutasi (Pm) maka tidak terjadi mutasi, dari proses tersebut akan menghasilkan kromosom mana yang mengalami mutasi dan yang tidak, misal pada permasalahan ini menggunakan Pm 0.05 maka hasilnya terlihat pada Tabel 3.17.

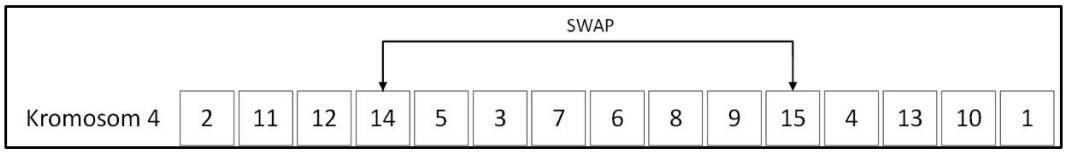
Tabel 3.17. Penentuan Mutasi

No	Kromosom	Angka Acak	Pm	Hasil
1.	1	0.415	0.05	Tidak Terjadi Mutasi
2.	2	0.516	0.05	Tidak Terjadi Mutasi
3.	3	0.398	0.05	Tidak Terjadi Mutasi
4.	4	0.018	0.05	Terjadi Mutasi
5.	5	0.428	0.05	Tidak Terjadi Mutasi

Bila suatu kromosom terjadi mutasi Langkah selanjutnya yaitu membangkitkan 2 angka acak 1-15 untuk menentukan gen mana yang akan ditukar, pada kasus ini kromosom ke 4 pada Tabel 3.17. yang akan dilakukan proses mutasi.

Gambar 3.8. Kromosom 4

Kemudian bangitkan 2 bilangan acak, dan menghasilkan bilangan 4 dan 11 maka proses selanjutnya menukar 2 gen pada urutan 4 dan 11 seperti pada Gambar 3.9. kemudian hasil dari mutasi terlihat pada Gambar 3.10.



Gambar 3.9. Proses Mutasi

Gambar 3.10. Hasil Mutasi Pada Kromosom 4

Tabel 3.18. Populasi Setelah Proses Mutasi

No	Kromosom	Susunan Kromosom	Nilai Objektif (*Makespan*)
1.	1	2-11-14-3-15-12-8-6-7-9-5-4-13-10-1	9325
2.	2	14-2-5-3-15-12-8-11-1-6-7-13-9-10-4	10403
3.	3	3-14-9-7-10-11-4-8-1-15-5-13-2-6-12	10595
4.	4	2-11-12-15-5-3-7-6-8-9-14-4-13-10-1	10097
5.	5	3-14-9-7-10-11-4-8-1-15-5-13-2-6-12	10595

g. *Elitisme*

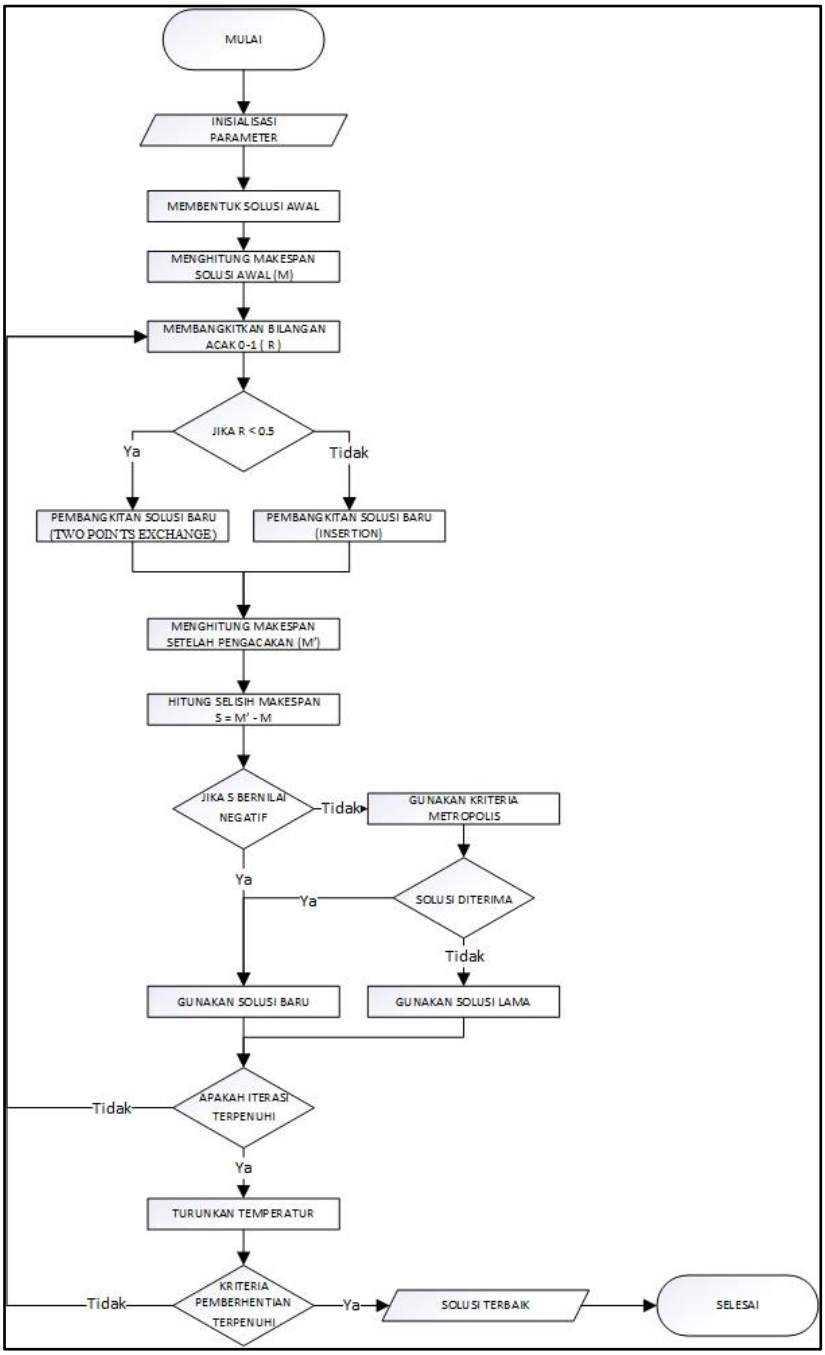
Pada proses *elitism* kromsom yang memiliki *fitness* terbaik akan disimpan kemudian kromosom terbaik tersebut akan dimunculkan lagi diiterasi berikutnya , sehingga akan menjaga agar kromosom dengan fitness tertinggi tersebut tidak akan hilang selama proses evolusi.

h. Kriteria pemberhentian

Proses pencarian akan berhenti ketika iterasi yang ditentukan sudah tercapai.

3.4.3. Proses *Simulated Annealing*

Flowchart proses *Simulated Annealing* dalam *Hybrid* Algoritma Genetika dan *Simulated Annealing* terlihat pada Gambar 3.12.



Gambar 3.12. Flowchart Simulated Annealing

Komponen-komponen dalam proses *Simulated Annealing*:

a. Inisialisasi Parameter

Terdapat beberapa parameter dalam *simulated annealing* yang perlu ditentukan terlebih dahulu yaitu parameter awal (T), jumlah iterasi pengacakan (k),

dan faktor pengurangan temperature (c), pada contoh kasus ini akan menggunakan temperature awal 100, jumlah iterasi 5 dan faktor pengurangan suhu 0.5.

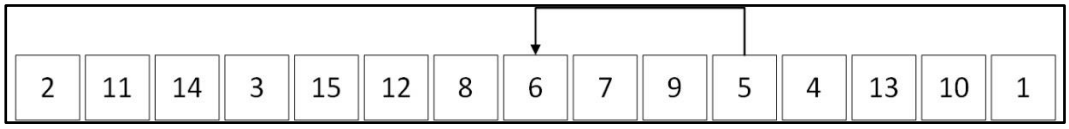
b. Membentuk solusi awal

Pada pembentukan solusi awal solusi yang dipakai adalah solusi terbaik yang dihasilkan oleh proses algoritma genetika, hasil terbaik dari proses algoritma genetika sebelumnya yaitu pada kromosom 1 dengan urutan pekerjaan 2-11-14-3-15-12-8-6-7-9-5-4-13-10-1 dengan nilai *makespan* 9325.

c. Membangkitkan solusi baru

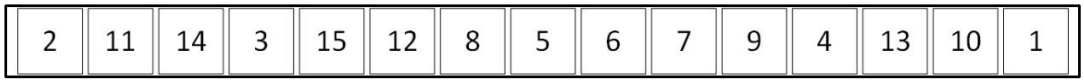
Dalam proses pembangkitan solusi baru, pertama bangkitkan bilangan acak 0-1 kemudian bilangan tersebut akan menjadi acuan untuk menentukan teknik apa yang dipakai, dalam kasus ini bilangan acak yang terbentuk yaitu 0.79 karena bilangan acak > 0.5 maka dilakukan teknik *Insertion*.

1. Bangkitkan 2 bilangan acak 1-15 untuk menentukan posisi yang akan dilakukan *Insertion* , hasil dari pembangkitan yaitu 8 dan 11.
2. Pindahkan gen ke 11 lalu masukkan ke sebelum gen ke 8.



Gambar 3.13. Pemindahan Gen

3. Kemudian geser semua gen yang terdapat diantara gen 8 dan 11.



Gambar 3.14. Hasil *Insertion*

d. Menghitung nilai objektif

Setelah didapatkan solusi baru kemudian dilakukan perhitungan nilai objektif, penghitungan nilai objektif pada *Simulated Annealing* sama dengan tahap perhitungan nilai objektif di Algoritma Genetika.

Tabel 3.19. Perhitungan Nilai Objektif Setelah Insertion

No	Job	Mesin 1		Mesin 2		Mesin 3		Mesin 4	
		S	E	S	E	S	E	S	E
1.	Job 2	0	45	45	270	270	750	750	795
2.	Job 11	45	279	279	1449	1449	3165	3165	3399
3.	Job 14	279	357	1449	1799	3165	2215	3399	3477
4.	Job 3	357	510	1799	2564	2564	3686	3686	3839
5.	Job 15	510	780	2564	3914	3914	5354	5354	5624
6.	Job 12	780	810	3914	4094	5354	5354	5624	5642
7.	Job 8	810	876	4094	4424	5354	6124	6124	6190
8.	Job 5	876	939	4424	4739	6124	6460	6460	6523
9.	Job 6	939	1064	4739	5489	6460	6710	6710	6785
10.	Job 7	1064	1082	5489	5579	6710	6956	6956	6974
11.	Job 9	1082	1118	5579	5759	6956	7628	7628	7664
12.	Job 4	1118	1154	5759	6059	7628	7778	7778	7814
13.	Job 13	1154	1174	6059	6179	7778	7813	7814	7820
14.	Job 10	1174	1474	6179	7679	7813	8813	8813	9113
15.	Job 1	1474	1519	7679	7904	8813	9293	9293	9338

e. Mengevaluasi solusi baru

Pada tahap evalusi solusi baru, solusi baru yang dihasilkan akan dibandingkan dengan solusi lama, seperti pada persamaan (2.5) apabila hasil dari Δf negatif maka terima solusi baru, tetapi apabila hasilnya positif maka perlu digunakan kriteria metropolis untuk menentukan diterima tidaknya solusi tersebut dengan persamaan (2.6). Pada contoh kasus ini maka Δf = 9338 – 9325 dan hasilnya bernilai positif, maka perlu digunakan kriteria metropolis. Pertama membangkitkan bilangan acak 0-1 lalu bandingan dengan $e^{-\Delta f/kT}$, dimana k disini bernilai 1 maka perhitungannya akan menjadi $e^{-13/100}$ maka hasilnya adalah 0.878 dan bilangan random yang dihasilkan adalah 0.45 ,karena bilangan acak lebih kecil maka terima solusi tersebut.

f. Menurunkan temperatur

Setelah iterasi telah terpenuhi maka akan dilakukan penurunan temperatur pada contoh kasus ini temperatur saat ini adalah 100 maka ketika dilakukan penurunan temperatur dengan persamaan (2.6) menjadi T_{baru} = 100 x 0.5 maka temperatur akan menjadi bernilai 50.

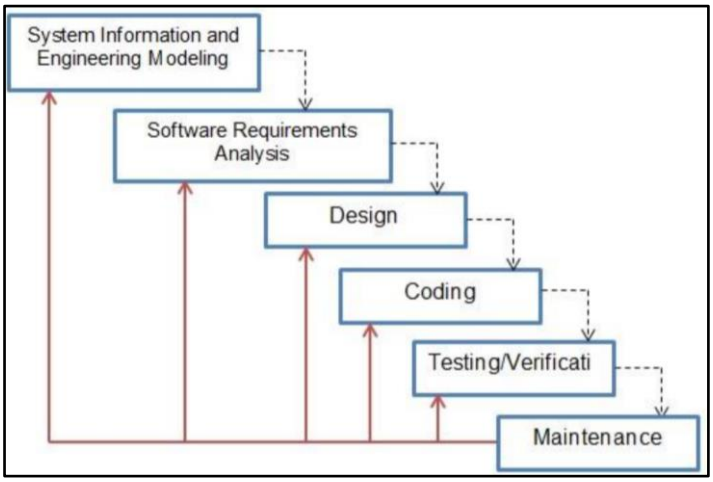
g. Kriteria Pemberhentian

Kriteria pemberhentian yaitu ketika temperatur sudah sampai dititik kriteria pemberhentian suhu, pada kasus ini titik permberhentian suhu yaitu 0.0000001, maka ketika proses pencarian dimulai suhu akan selalu berkurang ketika proses penutunan temperatur sehingga ketika suhu sudah mancapai titik atau kurang dari pemberhentian suhu maka proses akan berhenti.

3.5. Proses Desain

Di dalam subbab proses desain ini berisi tahap-tahapan yang akan menjelaskan proses pembuatan perangkat lunak penjadwalan produksi *flowshop* dengan metode *hybrid* algoritma genetika dan *simulated annealing* yang sudah dijelaskan pada subbab sebelumnya. Di dalam proses desain akan dibagi menjadi beberapa subbab lagi yang akan membahas tentang perancangan sistem, perancangan proses, dan desain perancangan antarmuka perangkat lunak yang akan dibangun. Tujuan dari proses desain adalah untuk memberikan gambaran tentang perangkat lunak yang akan dibangun, mulai dari proses kerja perangkat lunak tersebut, hingga tampilan antarmuka yang akan dibuat.

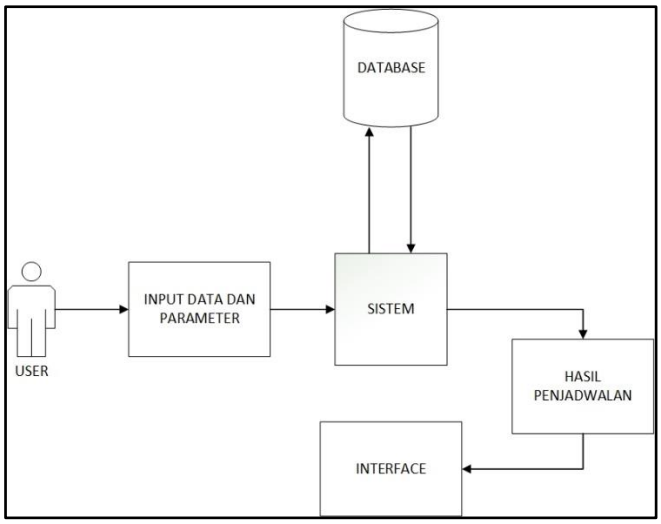
3.5.1. Perancangan Sistem



Gambar 3.15. Model Pengembangan Waterfall

Pengembangan perangkat lunak pada penelitian ini akan menggunakan model pengembangan waterfall. Tahapan model pengembangan waterfall dapat dilihat pada Gambar 3.15. di atas. Di dalam model pengembangan waterfall dibagi ke dalam beberapa tahapan diantaranya adalah.

1. System Information and Engginering Modelling Dalam tahap ini dilakukan pengumpulan kebutuhan sistem. Dilakukan dengan mencatat kebutuhan fungsional dan kebutuhan non fungsional.
2. Software Requirements Analysis Setelah mendapatkan kebutuhan sistem, pada tahap ini dilakukan analisis terhadap data-data kebutuhan yang sudah didapatkan pada tahap sebelumnya.
3. Design Pada tahap ini dilakukan proses desain meliputi desain dfd sistem hingga desain interface sistem.
4. Coding Setelah menyelesaikan tahapan desain dan mendapatkan gambaran desain sistem yang akan dibuat, pada tahap ini dilakukan proses pengkodean atau implementasi dari desain menjadi perangkat lunak.
5. Testing Pada tahap ini dilakukan pengujian terhadap perangkat lunak yang sudah dibuat.
6. Maintenance Tahapan terakhir adalah proses maintenance atau pemeliharaan perangkat lunak tersebut. Apabila ditemukan error maka dapat dilakukan perbaikan dan dilakukan proses testing kembali, dan memungkikan untuk melakukan pengembanga terhadap sistem tersebut.

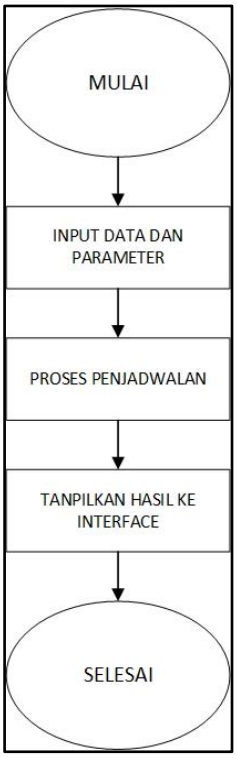


Gambar 3.16. Arsitektur Perangkat Lunak

Gambaran umum arsitektur sistem yang akan dibuat adalah seperti Gambar 3.16. di atas. Dimulai dari user yang melakukan input data dan parameter. Data tersbut kemudian akan dimasukkan ke database yang selanjutnya akan dilakukan proses penjadwalan oleh sistem. Setelah mendapatkan jadwal terbaik hasil dari penjadwalan tersebut akan disimpan didatabase dan ditampilkan diinterface perangkat lunak.

3.5.2. Perancangan Proses

Dalam perancangan proses ini, akan dijelaskan proses-proses yang terjadi di dalam perangkat lunak yang akan dibuat. Flowchart proses sistem terdapat pada Gambar 3.17 di bawah ini.

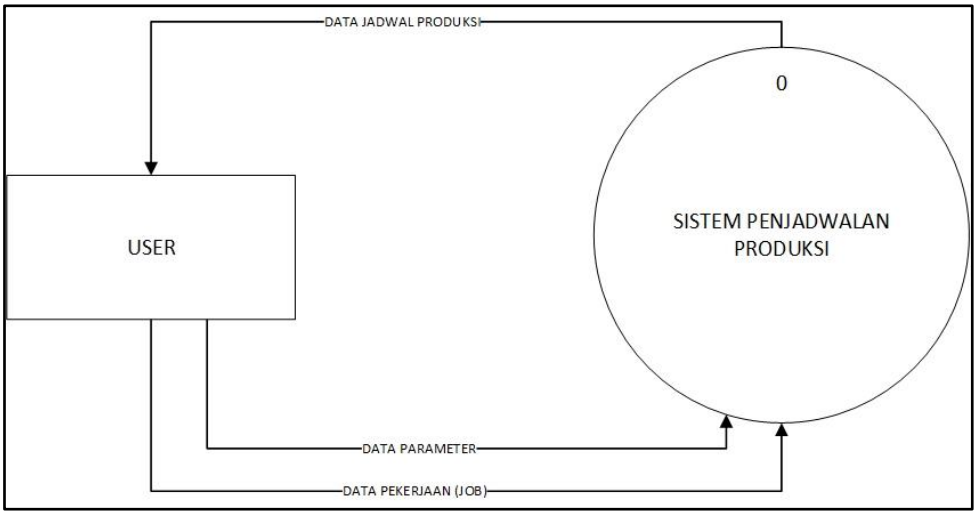


Gambar 3.17. *Flowchart* Sistem

Sesuai dengan flowchart di atas, proses yang terjadi pada sistem adalah proses penjadwalan produksi dengan algoritma *Hybrid* algoritma genetika dan simulated annealing.

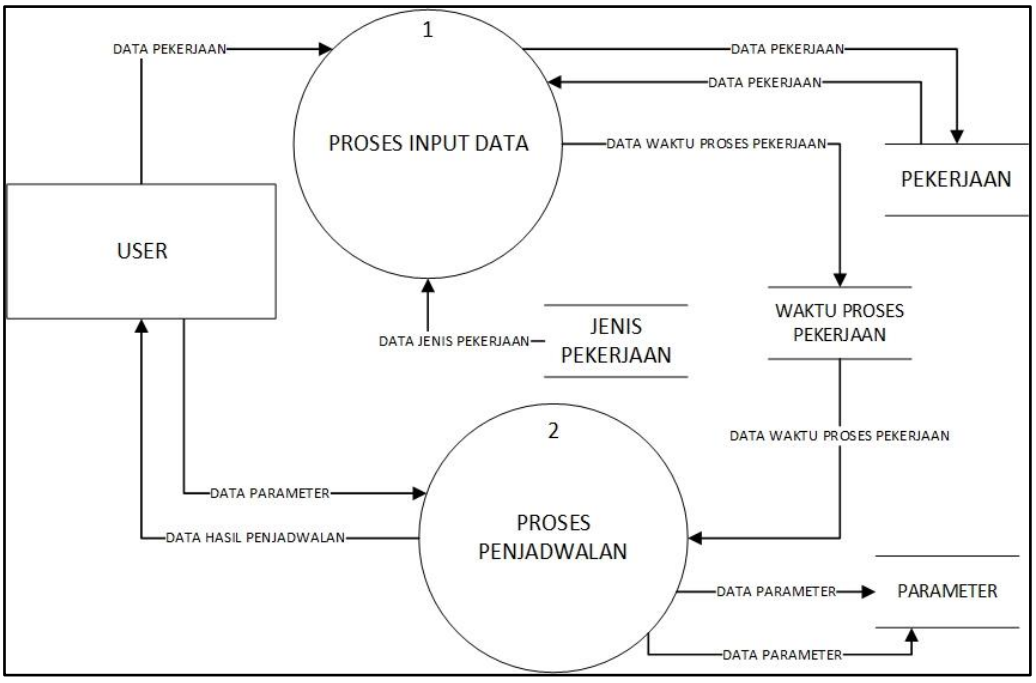
Dimulai dari user yang melakukan input data dan parameter, kemudian inputan data dan parameter tersebut akan diproses untuk mendapatkan jadwal yang terbaik, pada prosesnya sistem akan melakukan proses penjadwalan sesuai dengan parameter yang telah diinput oleh user. Kemudian setelah sistem melakukan proses penjadwalan, nantinya hasil penjadwalan akan ditampilkan ke interface perangkat lunak.

Berikut pada Gambar 3.18 di bawah ini adalah DFD dari sistem yang akan dibuat. Berisi gambaran proses dan data apa saja yang akan berlangsung dalam sistem tersebut.



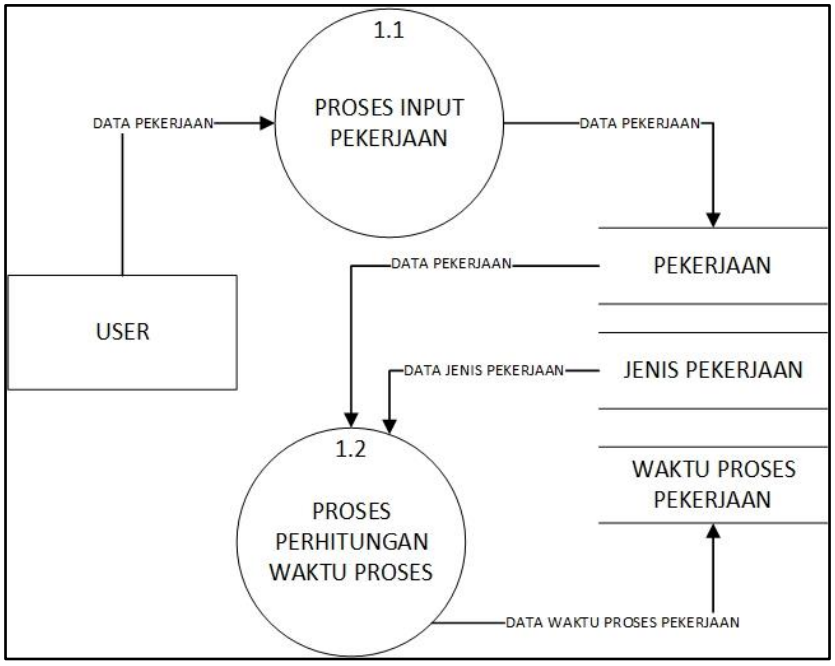
Gambar 3.18. DFD Level 0

Pada Gambar 3.18 di atas terlihat bahwa sistem yang akan dibuat adalah sistem penjadwalan yang sudah siap digunakan, model yang digunakan untuk penjadwalan sudah dibuat sesuai dengan kondisi dalam proses poduksi. Jadi, sistem yang dibuat hanya digunakan untuk penjadwalan. Entitas yang ada di dalam dfd tersebut hanya satu yaitu user. User dapat melakukan input berupa data pekerjaan dan parameter yang akan diproses oleh sistem dan dilakukan proses penjadwalan oleh model. Kemudian hasil penjadwalan yang dihasilkan oleh model akan ditampilkan kepada user.

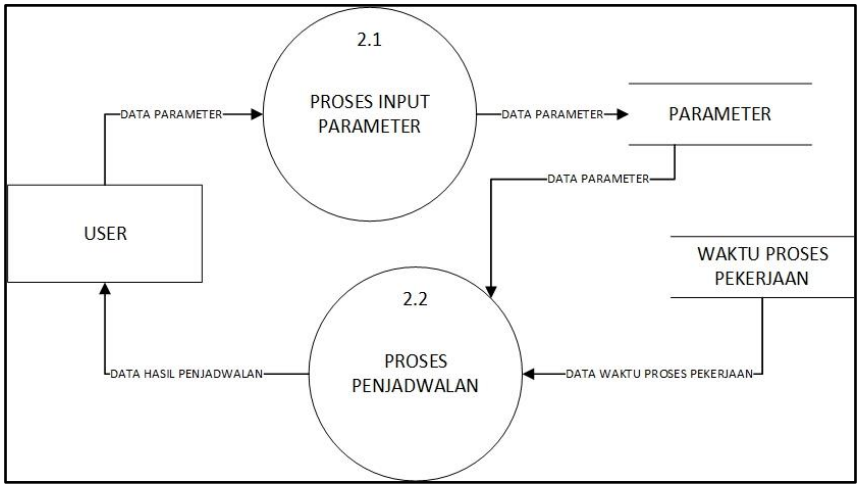


Gambar 3.19. DFD Level 1

Di dalam dfd level 1 yang ditampilkan pada gambar 3.19 di atas, proses utama pada dfd tersebut dipecah menjadi dua proses utama yaitu proses input data dan proses penjadwalan, dimana pada proses input data user menginputkan data pekerjaan yang akan dijadwalkan kemudian data dimasukkan ke database pekerjaan yang nantinya akan diambil untuk proses penjadwalan. Hasil dari proses penjadwalan tersebut nantinya akan ditampilkan ke user.



Gambar 3.20. DFD Level 2 proses 1



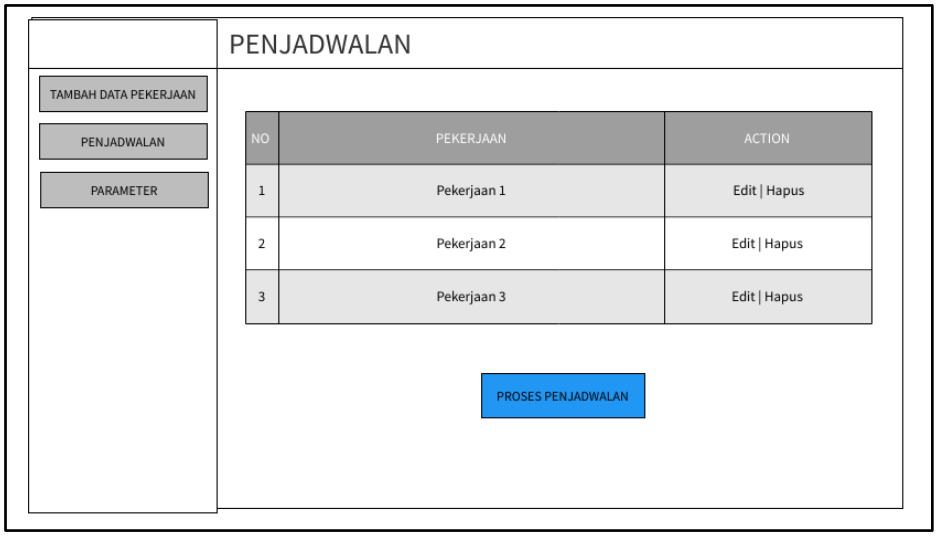
Gambar 3.21. DFD Level 2 proses 2

Dari Gambar 3.20 terlihat bahwa pada proses input data dari user data langsung dimasukkan ke database pekerjaan, setelah itu terlihat pada Gambar 3.21 saat proses penjadwalan data pertama user akan menginputkan parameter setelah itu data dari pekerjaan dari user yang telah diinputkan tadi akan diambil dan diproses sesuai dengan parameter yang diinputkan setelah proses penjadwalan selesai hasil dari penjadawalan akan ditampilkan ke user.

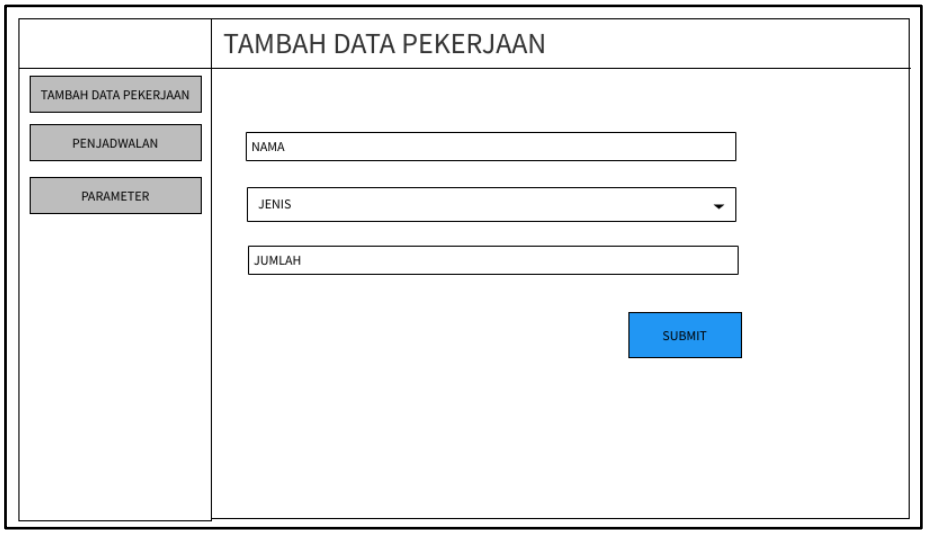
3.5.3. Perancangan Antarmuka

Tampilan antarmuka perangkat lunak atau disebut user interface adalah sebuah media yang digunakan untuk komunikasi antara pengguna dan sistem. Tampilan antarmuka atau user interface berisi fungsi-fungsi yang ada di dalam sistem yang dibuat dengan tampilam grafis

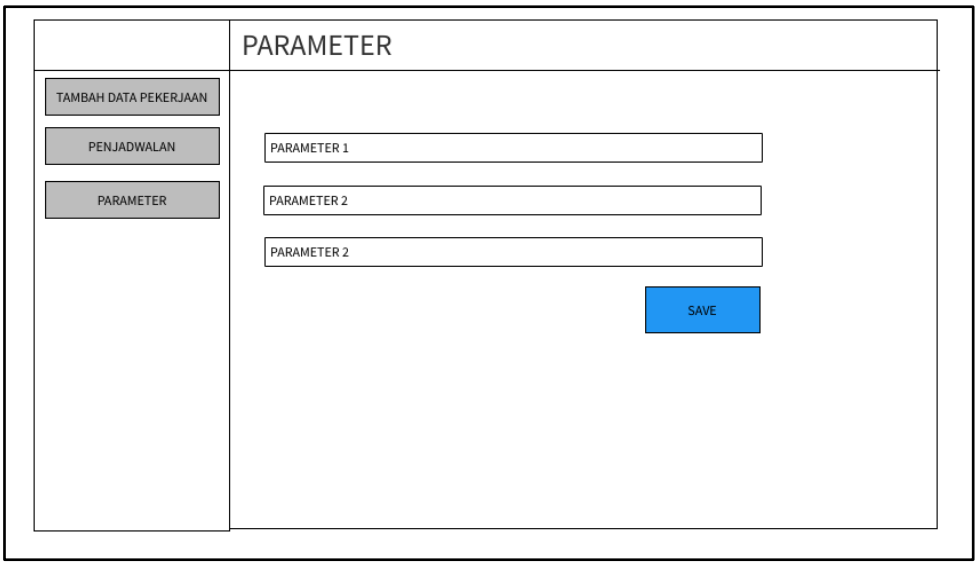
yang memudahkan pengguna untuk menggunakan sistem tersebut. Dalam sistem penjadwalan produksi yang akan dibuat ini, nantinya akan menggunakan beberapa tampilan yang berisi form inputan untuk memasukkan data dan parameter, kemudian terdapat tampilan untuk menampilkan hasil dari penjadwalan ,hasil dari penjadwalan akan ditampilkan dalam bentuk *gantt chart* sehingga akan mempermudah user dalam memahaminya. Rancangan user interface yang akan dibuat adalah seperti dibawah ini.



Gambar 3.22. User interface menu penjadwalan



Gambar 3.23. User interface menu tambah data



Gambar 3.24. User interface menu parameter

3.5.4. Jadwal Penelitian

Tabel 3.20. Jadwal Penelitian

No	Kegiatan	Bulan I				Bulan II				Bulan III				Bulan IV				Bulan V			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	**Pengumpulan Data**																				
	Pengambilan Data	■	■																		
	Analisis Data		■	■																	
2	**Perancangan Sistem**																				
	Planning				■	■															
	Analysis					■	■														
	Design						■	■	■												
	Implementation									■	■	■	■	■							
3	**Pengujian Sistem**																				
	Pengujian dan Evaluasi Sistem														■	■	■				
4	**Pembuatan Laporan**																				
	Pembuatan Laporan																	■	■	■	■

Daftar Pustaka

Ancău, M. (2012). On solving flowshop scheduling problems. *Proceedings of the Romanian Academy Series A - Mathematics Physics Technical Sciences Information Science*, *13*(1), 71–79.

Andresen, M., Bräsel, H., Mörig, M., Tusch, J., Werner, F., & Willenius, P. (2008). Simulated annealing and genetic algorithms for minimizing mean flow time in an open shop. *Mathematical and Computer Modelling*, *48*(7–8), 1279–1293. https://doi.org/10.1016/j.mcm.2008.01.002

Azmi, N., Jamaran, I., Arkeman, Y., & Mangunwidjaja, D. (n.d.). Penjadwalan Pesanan Menggunakan Algoritma Genetika untuk Tipe Produksi Hybrid and Flexible Flowshop pada Industri Kemasan Karton. *Jurnal Teknik Industri*, 176–188.

Baker, K. R., & Trietsch, D. (2013). *Principles of sequencing and scheduling*. John Wiley & Sons.

Briggs, L. J., Gustafson, K. L., & Tillman, M. (1991). *Instructional Design: Principles and Applications*. Educational Technology Publications. https://books.google.co.id/books?id=aOcWFqPw4JQC

Dai, M., Tang, D., Giret, A., Salido, M. A., & Li, W. D. (2013). Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. *Robotics and Computer-Integrated Manufacturing*, *29*(5), 418–429. https://doi.org/10.1016/j.rcim.2013.04.001

Dai, M., Tang, D., Zheng, K., & Cai, Q. (2013). An improved genetic-simulated annealing algorithm based on a hormone modulation mechanism for a flexible flow-shop scheduling problem. *Advances in Mechanical Engineering*, *2013*. https://doi.org/10.1155/2013/124903

Etiler, O., Toklu, B., Atak, M., & Wilson, J. (2004). A genetic algorithm for flow shop scheduling problems. *Journal of the Operational Research Society*, *55*(8), 830–835. https://doi.org/10.1057/palgrave.jors.2601766

Firdaus, M., Masudin, I., & Utama, D. M. (2015). Penjadwalan flowshop dengan menggunakan simulated annealing. *Spektrum Industri*, *13*(1), 27–40.

Gao, J., Chen, R., & Deng, W. (2013). An efficient tabu search algorithm for the distributed

permutation flowshop scheduling problem. *International Journal of Production Research*, *51*(3), 641–651. https://doi.org/10.1080/00207543.2011.644819

Ginting, R. (2009). *Penjadwalan Mesin* (1st ed.).

Govindan, K., Balasundaram, R., Baskar, N., & Asokan, P. (2017). A hybrid approach for minimizing makespan in permutation flowshop scheduling. *Journal of Systems Science and Systems Engineering*, *26*(1), 50–76. https://doi.org/10.1007/s11518-016-5297-1

Grabowski, J., & Wodecki, M. (2004). A very fast tabu search algorithm for the permutation flow shop problem with makespan criterion. *Computers and Operations Research*, *31*(11), 1891–1909. https://doi.org/10.1016/S0305-0548(03)00145-X

Ismail, Z., & Irhamah. (2008). Adaptive permutation-based genetic algorithm for solving VRP with stochastic demands. *Journal of Applied Sciences*, *8*(18), 3228–3234. https://doi.org/10.3923/jas.2008.3228.3234

Li, X., & Gao, L. (2016). An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem. *International Journal of Production Economics*, *174*, 93–110. https://doi.org/10.1016/j.ijpe.2016.01.016

Lin, S. W., & Ying, K. C. (2009). Applying a hybrid simulated annealing and tabu search approach to non-permutation flowshop scheduling problems. *International Journal of Production Research*, *47*(5), 1411–1424. https://doi.org/10.1080/00207540701484939

Low, C. (2005). Simulated annealing heuristic for flow shop scheduling problems with unrelated parallel machines. *Computers and Operations Research*, *32*(8), 2013–2025. https://doi.org/10.1016/j.cor.2004.01.003

Minmei, H., Ronggui, L., & Jijun, Y. (2006). Heuristic-tabu-genetic algorithm based method for flowshop scheduling to minimize flowtime. *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, *2*, 7220–7224. https://doi.org/10.1109/WCICA.2006.1714487

Osaba, E., Diaz, F., & Onieva, E. (2013). A novel meta-heuristic based on soccer concepts to solve routing problems. *GECCO 2013 - Proceedings of the 2013 Genetic and Evolutionary Computation Conference Companion*, 1743–1744. https://doi.org/10.1145/2464576.2480776

Pamungkas, A. (2002). Perbandingan Kinerja Algoritma Genetika Dan Simulated Annealing Untuk Masalah Multiple Objective Pada Penjadwalan Flowshop. *Jurnal Teknik Industri*, *4*(1), 26–35. https://doi.org/10.9744/jti.4.1.pp.26-35

Raghavan, S. S. (2015). Na pravilu zasnovan heuristički pristup smanjenju ukupnog protoka vremena kod programiranja radova u permutacijskoj protočnoj radionici. *Tehnicki Vjesnik*, *22*(1), 25–32. https://doi.org/10.17559/TV-20130704132725

Rahman, H. F., Sarker, R., & Essam, D. (2015). A genetic algorithm for permutation flow shop scheduling under make to stock production system. *Computers and Industrial Engineering*, *90*, 12–24. https://doi.org/10.1016/j.cie.2015.08.006

Reeves, C. R. (1995). A genetic algorithm for flowshop sequencing. *Computers and Operations Research*, *22*(1), 5–13. https://doi.org/10.1016/0305-0548(93)E0014-K

Sanjaya, W. (2008). Perencanaan & desain sistem pembelajaran. *Jakarta: Kencana Prenadamedia Group*.

Santosa, B., & Willy, P. (2011). Metoda Metaheuristik konsep dan implementasi. *Guna Widya, Surabaya*.

Sayoti, F., & Essaid Ri, M. (2016). Golden Ball Algorithm for solving Flow Shop Scheduling Problem. *International Journal of Interactive Multimedia and Artificial Intelligence*, *4*(1), 15. https://doi.org/10.9781/ijimai.2016.413

Schaffer, J. D., Caruana, R. A., Eshelman, L. J., & Das, R. (1989). A study of control parameters affecting online performance of genetic algorithms for function optimization. *Proceedings of the Third International Conference on Genetic Algorithms, November 2014*, 51–60.

Setiya Widodo, D., Budi Santoso, P., & Siswanto, E. (2014). Pendekatan Algoritma Cross Entropy-Genetic Algorithm Untuk Menurunkan Makespan Pada Penjadwalan Flow Shop. *Journal of Engineering and Management Industial System*, *2*(1), 41–49. https://doi.org/10.21776/ub.jemis.2014.002.01.6

Sukkerd, W., & Wuttipornpun, T. (2016). Hybrid genetic algorithm and tabu search for finite capacity material requirement planning system in flexible flow shop with assembly operations. *Computers and Industrial Engineering*, *97*, 157–169. https://doi.org/10.1016/j.cie.2016.05.006

Taillard, E. (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research*, *64*(2), 278–285. https://doi.org/10.1016/0377-2217(93)90182-M

Wang, S., & Liu, M. (2013). A genetic algorithm for two-stage no-wait hybrid flow shop scheduling problem. *Computers and Operations Research*, *40*(4), 1064–1075. https://doi.org/10.1016/j.cor.2012.10.015

Wei, H., Li, S., Jiang, H., Hu, J., & Hu, J. (2018). Hybrid genetic simulated annealing algorithm for improved flow shop scheduling with makespan criterion. *Applied Sciences (Switzerland)*, *8*(12). https://doi.org/10.3390/app8122621

Yin, H. L. (2013). Genetic algorithm nested with simulated annealing for big job shop scheduling problems. *Proceedings - 9th International Conference on Computational Intelligence and Security, CIS 2013*, *3*, 50–54. https://doi.org/10.1109/CIS.2013.18

Yu, C., Semeraro, Q., & Matta, A. (2018). A genetic algorithm for the hybrid flow shop scheduling with unrelated machines and machine eligibility. *Computers and Operations Research*, *100*, 211–229. https://doi.org/10.1016/j.cor.2018.07.025