BAB II

LANDASAN TEORI

2.1. Penjadwalan

Penjadwalan merupakan proses untuk menyusun suatu jadwal atau urusan proses yang diperlukan dalam sebuah persoalan. Persoalan penjadwalan biasanya berhubungan dengan penjadwalan kelas dalam sekolah atau perkuliahan dan juga dalam lingkup yang tidak jauh berbeda seperti penjadwalan mata kuliah atau pelajaran. Penjadwalan ujian, atau bisa juga penjadwalan karyawan, baik dalam suatu perusahaan ataupun dalam rumah sakit (Dian Ariani, 2011).

Menurut Imam Sodikin (2012), penjadwalan dengan menggunakan sistem komputasi dibagi ke dalam 3 metode, yaitu:

a. Metode optimum yang efisien

   Metode ini menghasilkan jadwal optimum dalam waktu yang relative singkat. Algoritma yang dikembangkan biasanya untuk permasalahan yang tidak besar.

b. Metode optimal numeratif

   Metode ini menghasilkan jadwal optimum berdasarkan formulasi matematis, diikuti oleh *Branch and Bround*, *Mixed Integer Linear Programming*, dan *Dynamic Programming*.

c. Metode heuristik

   Metode heuristic melakukan pendekatan suatu solusi optimal. Dasar dari pengembangan motede heuristik dikategorikan menjadi 3, yaitu:

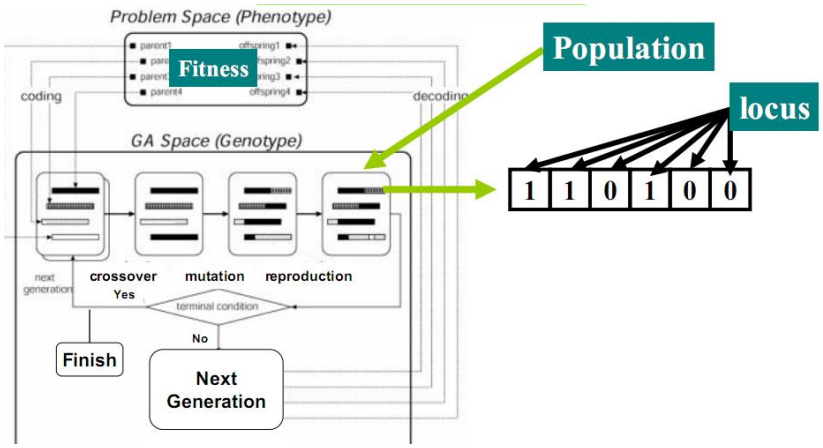
1. Penjadwalan dilakukan setiap mesin selesai melakukan proses atau setiap pekerjaan datang mengantri. Contoh pendekatan ini adalah *priority rule*.
2. Pendefinisian struktur *neightbourhood* dan solusi diperoleh berdasarkan struktur tersebut. Contoh pendekatan ini adalah *tabu search*, *simulated annealing*, dan *genetic algorithm*.
3. Penjadwalan dilakukan pada setiap mesin. Contoh pendekatan ini adalah *shifting bottleneck procedure*.

2.2. Algoritma Genetika

Algoritma Genetika adalah algoritma pencarian berdasarkan mekanisme proses seleksi alam (*biological evolution*). Konsep yang paling mendasar adalah yang kuat akan cenderung mudah beradaptasi dan bertahan sedangkan yang lemah akan cenderung mati. Artinya, optimasi didasarkan pada evolusi dengan konsep "*Survival of the fittest*". (John Holland, 1975)

Algoritma Genetika adalah algoritma pencarian yang didasarkan atas mekanisme dari seleksi alam yang lebih di kenal dengan proses evolusi. Dalam proses evolusi, individu secara terus menerus mengalami perubahan gen untuk menyesuaikan dengan lingkungan hidupnya. Hanya individu-individu yang kuat yang mampu bertahan. Proses seleksi alamiah ini melibatkan perubahan gen yang terjadi pada individu melalui proses perkembangbiakan. Dalam algoritma genetika, proses perkembangbiakan ini menjadi proses dasar yang menjadi perhatian utama, dengan dasar berpikir: "Bagaimana mendapatkan keturunan yang lebih baik" (Ahmad Basuki, 2003).

Menurut John Holland (1975), Algoritma Genetika dapat dijabarkan dengan struktur dasar sebagai berikut:



Gambar 1.1 Struktur Dasar Algoritma Genetika

Algoritma genetika yang digunakan di sini berfungsi untuk mendefinisikan permasalahan ke dalam suatu kumpulan solusi, kemudian melalui proses-proses yang ada dalam algoritma genetika, dilakukan pencarian solusi yang paling optimal yang didasarkan atas nilai fitness. Semakin besar nilai fitness dalam proses evolusi, maka semakin banyak jumlah bentrokan yang terjadi dari solusi yang diberikan. Hal ini dapat dikatakan, bahwa dalam proses algoritma genetika untuk masalah penjadwalan adalah dicari nilai fitness sama dengan nol.

Berikut beberapa istilah dalam Algoritma Genetika:

1. *Genotype* (Gen), merupakan sebuah nilai yang menyatakan satuan dasar yang membentuk suatu arti tertentu dalam satu kesatuan gen yang dinamakan kromosom. Dalam algoritma genetika, gen ini bisa berupa nilai biner, float, integer maupun karakter.
2. *Allele*, merupakan nilai dari gen.

3. Individu atau kromosom, merupakan gabungan gen-gen yang membentuk nilai tertentu.
4. Populasi, merupakan sekumpulan individu yang akan di proses bersama dalam satu siklus proses evolusi.
5. Generasi, merupakan satu siklus evolusi atau satu iterasi di dalam algoritma genetika.

Ada beberapa komponen dalam algoritma genetika, yaitu :

1. Pengkodean

   Teknik pengkodean adalah bagaimana mengkodekan gen dari kromosom, gen merupakan bagian dari kromosom. Satu gen akan mewakili satu variabel. Agar dapat diproses melalui algoritma genetika, maka alternatif solusi tersebut harus dikodekan terlebih dahulu kedalam bentuk kromosom. Masing-masing kromosom berisi sejumlah gen yang mengodekan informasi yang disimpan didalam individu atau kromosom.

   Berdasarkan jenis simbol yang digunakan sebagai nilai suatu gen, metode pengkodean dapat diklasifikasikan sebagai berikut (Arifudin, 2011):

   a. Pengkodean biner merupakan cara pengkodean yang paling umum digunakan, karena pengkodean ini merupakan yang pertama kali digunakan dalam algoritma genetika oleh Holland (Cordon et al, 2001). Pengkodean biner dinyatakan dalam kromosom biner.
   b. Pengkodean bilangan bulat. Pengkodean ini baik digunakan untuk masalah optimasi kombinatorial. Dengan pengkodean bilangan

bulat, ukuran kromosom menjadi lebih sederhana dan tidak terlalu panjang.

Inisialisasi populasi awal merupakan suatu metode untuk menghasilkan kromosom-kromosom awal. Jumlah individu pada populasi awal merupakan masukan dari pengguna. Setelah jumlah individu pada populasi awal ditentukan, dilakukan inisialisasi terhadap kromosom yang terdapat pada populasi tersebut. Inisialisasi dilakukan secara acak, namun tetap memperhatikan domain solusi dan kendala permasalahan yang ada.

2. Fungsi Evaluasi (Fungsi *Fitness*)

Fungsi evaluasi dalam algoritma genetika merupakan sebuah fungsi yang memberikan penilaian kepada kromosom (fitness value) untuk dijadikan suatu acuan dalam mencapai nilai optimal pada algoritma genetika. Nilai fitness ini kemudian menjadi nilai bobot suatu kromosom. Ada dua hal yang harus dilakukan dalam melakukan evaluasi kromosom, yaitu: evaluasi fungsi objektif (fungsi tujuan) dan konversi fungsi objektif ke dalam fungsi fitness.

Didalam evolusi alam, individu yang bernilai fitness tinggi yang akan bertahan hidup. Sedangkan individu yang bernilai fitness rendah akan mati. Pada masalah optimasi, jika solusi yang akan dicari adalah memaksimumkan fungsi h (dikenal sebagai masalah maksimasi) sehingga nilai fitness yang digunakan adalah nilai dari fungsi h tersebut, yakni f = h (dimana f adalah nilai fitness). Tetapi jika masalahnya adalah meminimalkan fungsi h (masalah minimasi), maka fungsi h tidak bisa

digunakan secara langsung. Hal ini disebabkan adanya aturan bahwa individu yang memiliki nilai fitness tinggi lebih mampu bertahan hidup pada generasi berikutnya. Oleh karena itu nilai fitness yang bisa digunakan adalah $f = 1/h$, yang artinya semakin kecil nilai h, semakin besar nilai f. Tetapi hal ini akan menjadi masalah jika h bisa bernilai 0, yang mengakibatkan f bisa bernilai tak hingga. Untuk mengatasinya, h perlu ditambah sebuah bilangan yang dianggap kecil [0-1] sehingga nilai fitnessnya menjadi:

$f = 1/h + a$.....(Persamaan 2.1)

dengan a adalah bilangan yang kecil dan bervariasi [0-1] sesuai dengan masalah yang akan diselesaikan (Suyanto,2005). Oleh karena itu fungsi fitness menjadi masalah atau penentu utama keberhasilan algoritma genetika. Didalam penelitian ini batasan atau constraint dalam penyusunan jadwal proyek yang dijadikan fungsi objektifnya yaitu meminimumkan pelanggaran terhadap constraint yang telah ditentukan. Jadi persamaan (2.1) cocok dipakai pada kasus dalam penelitian ini.

3. Seleksi

Seleksi merupakan proses pemilihan orang tua untuk reproduksi (biasanya didasarkan pada nilai fitness). Seleksi bertujuan untuk memberikan kesempatan reproduksi yang paling besar bagi anggota populasi yang paling baik. Ada beberapa metode yang bisa digunakan dalam tahap seleksi, diantaranya:

a. Range Base Fitness

Populasi diurutkan berdasarkan nilai objektifnya. Nilai fitness dari tiap-tiap individu hanya tergantung pada posisi individu tersebut dalam urutan dan tidak dipengaruhi oleh nilai objektifnya.

b. Roulette Wheel Selection (Seleksi Roda Roulette)

Individu-individu dipetakan dalam suatu segmen garis secara berurutan sedemikian hingga tiap-tiap segmen individu memiliki urutan yang sama dengan ukuran fitness. Sebuah bilangan acak dibangkitkan dan individu yang memiliki segmen dalam kawasan bilangan random tersebut akan terseleksi. Cara penyelesaian metode ini meniru permainan roda roulette, dimana prosedur seleksi dimulai dengan memutar roda roulette sebanyak n, setiap waktu dipilih satu kromosom sebagai induk untuk menghasilkan kromosom baru. Metode roulette-wheel selection sangat mudah diimplementasikan dalam pemrograman. Pertama, dibuat interval nilai kumulatif (dalam interval [0,1]) dari nilai fitness masing-masing kromosom dibagi total nilai fitness dari semua kromosom. Sebuah kromosom akan terpilih jika bilangan random yang dibangkitkan berada dalam interval akumulatifnya.

4. Pindah Silang (*Crossover*)

Pindah silang atau *crossover* adalah sebuah proses yang membentuk kromosom baru dari dua kromosom induk dengan menggabungkan bagian informasi dari masing-masing kromosom. *Crossover* menghasilkan kromosom baru yang disebut kromosom anak

(*offspring*). Crossover bertujuan untuk menambah keanekaragaman string dalam satu populasi dengan penyilangan antar string yang diperoleh dari reproduksi sebelumnya (Arifudin, 2011).

Proses *crossover* dilakukan dengan cara memilih dua induk dengan kualitas yang baik, setelah itu dilakukan proses ektraksi kromosom setiap induk. Titik potong ditentukan secara acak, kemudian dilakukan pertukaran bit-bit kromosom disebelah kanan titik kromosom sehingga terbentuk keturunan yaitu anak A dan B. Kromosom anak sebagian besar masih mewarisi kromosom induk tetapi sebagian lagi sudah terjadi pertukaran materi genetik antar kromosom.

Pindah silang bisa juga berakibat buruk jika ukuran populasinya sangat kecil. Dalam suatu populasi yang sangat kecil, suatu kromosom dengan gen-gen yang mengarah ke solusi akan sangat cepat menyebar ke kromosom-kromosom lainnya. Untuk mengatasi masalah ini digunakan suatu aturan bahwa pindah silang hanya bisa dilakukan dengan suatu probabilitas tertentu (*probabilitas crossover*). Artinya pindah silang bisa dilakukan hanya jika suatu bilangan random [0,1] yang dibangkitkan kurang dari probabilitas crossover (Pc) yang ditentukan. *Probabilitas crossover* (Pc) bertujuan untuk mengendalikan operator *crossover*. Jika n adalah banyaknya string pada populasi, maka sebanyak (Pc) x n string akan mengalami *crossove*r. Semakin besar nilai (Pc), semakin cepat pula string baru muncul dalam populasi. Dan juga jika (Pc) terlalu besar, string yang merupakan kandidat solusi terbaik mungkin dapat hilang lebih cepat pada generasi berikutnya.

Proses crossover memiliki nilai kemungkinan yang besar dalam satu siklus algoritma genetika karena tujuan utamanya adalah membentuk keragaman individu, semakin tinggi probabilitas crossover maka semakin cepat keragaman terbentuk. Crossover dapat dilakukan dengan beberapa cara yang berbeda, diantaranya:

a. Penyilangan Satu Titik (*One-point Crossover*)

Metode crossover ini yang sering digunakan pada algoritma genetika. Pada penyilangan satu titik, posisi penyilangan k (k= 1,2,3,...n) dengan n=panjang kromosom yang diseleksi secara acak. Pada titik tersebut dilakukan pertukaran antar kromosom induk untuk menghasilkan anak.

b. Penyilangan Banyak Titik (*Multi-point Crossover*)

Untuk kromosom yang sangat panjang, misalkan 1000 gen, mungkin saja diperlukan beberapa titik potong. Penyilangan dilakukan sebanyak m (m=1, 2, 3,...n) dengan posisi penyilangan k (k= 1,2,3,...n) yang ditentukan secara acak. Pada titik tersebut dilakukan pertukaran antar kromosom induk untuk menghasilkan anak.

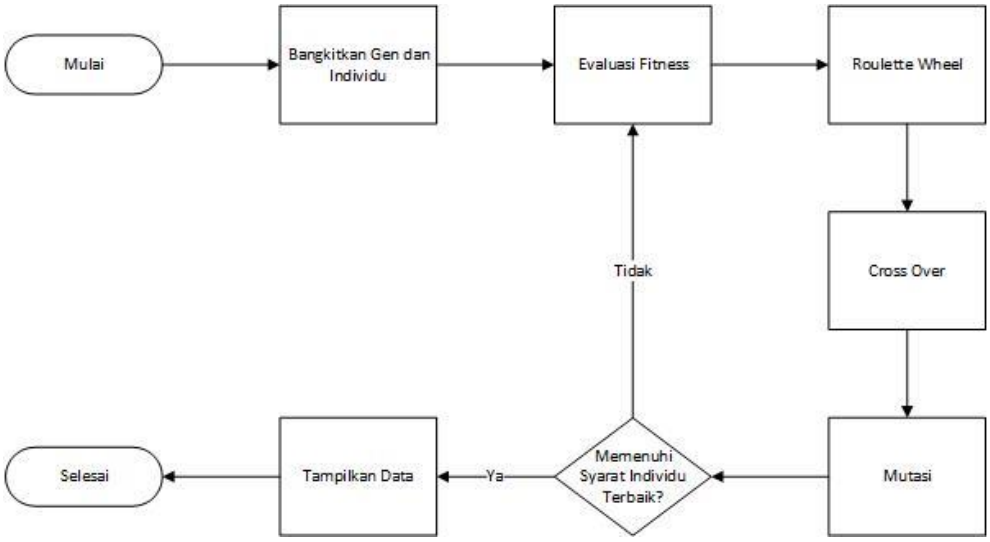
5. Mutasi

Mutasi adalah operator algoritma genetika yang bertujuan untuk membentuk individu-individu yang baik atau memiliki kualitas diatas rata-rata. Selain itu mutasi dipergunakan untuk mengembalikan kerusakan materi genetic akibat proses *crossover*.

Pada mutasi terdapat satu parameter yang sangat penting, yaitu probabilitas mutasi (Pm) yang bertujuan untuk mengendalikan operator mutasi. Probabilitas mutasi didefinisikan sebagai persentasi dari jumlah total gen dalam populasi yang akan mengalami mutasi. Disetiap generasi diperkirakan terjadi mutasi sebanyak (Pm) x n sring. Pada seleksi alam murni, mutasi jarang sekali muncul sehingga probabilitas mutasi yang digunakan umumnya kecil, lebih kecil dari probabilitas crossover. Pm biasanya diset antara [0-1], misalnya 0.1 (Suyanto,2005). Misalkan *offspring* yang terbentuk adalah 100 dengan jumlah gen setiap kromosom adalah 4 dan peluang mutasi adalah 0.10, maka diharapkan terdapat 40 kromosom dari 400 gen yang ada pada populasi tersebut akan mengalami mutasi.

Mutasi yang digunakan dalam penelitian ini adalah mutasi dengan pengkodean bulat. Mutasi gen ini dilakukan dengan cara pemilihan nilai secara acak. Suatu gen yang terpilih untuk dimutasi nilainya diganti dengan nilai baru yang dibangkitkan secara acak dalam interval nilai-nilai gen yang diizinkan. Misalnya, jika nilai-nilai gen berada dalam interval [0.9], maka gen baru yang dibangkitkan secara acak juga berada dalam interval [0.9]. Nilai gen baru yang dihasilkan bisa saja dibatasi dengan aturan berbeda dengan nilai lama.

Secara umum, Algoritma Genetika dapat dijabarkan sebagai alur dan bagan pada gambar berikut:



Gambar 2.2 Bagan Perulangan Algoritma Genetika

Struktur umum dari suatu Algoritma Genetika dapat didefiniskan dengan langkah-langkah sebagai berikut:

1. Membangkitkan populasi awal secara random.
2. Membentuk generasi baru dengan menggunakan tiga operasi antara lain seleksi, *crossover*, mutasi yang dilakukan secara berulang-ulang sehingga diperoleh kromosom yang cukup untuk membentuk generasi baru sebagai representasi dari solusi baru.
3. Evolusi solusi yang akan mengevaluasi setiap popolasi dengan menghitung nilai *fitness* setiap kromosom hingga kriteria berhenti terpenuhi.

Bila kriteria berhenti belum terpenuhi maka akan di bentuk lagi generasi baru dengan mengulangi langkah regenerasi. Beberapa kriteria benhenti yang sering digunakan antara lain:

1. Berhenti pada generasi tertentu.
2. Berhenti setelah dalam beberapa generasi berturut-turut didapakan nilai *fitness* tertinggi atau terendah (tergantung persoalan) tidak berubah.
3. Berhenti dalam generasi berikuutnya tidak diperoleh nilai *fitness* yang lebih tinggi atau rendah.

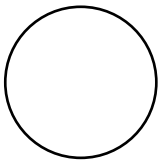
2.3. Data Flow Dokumen

Data Flow Dokumen (DFD) yang di dalam bahasa Indonesia disebut Diagram Arus Dra (DAD) memperlihatkan gambaran tentang masukan – proses – keluaran dari suatu sistem atau perangkat lunak, yaitu obyek – obyek data mengalir ke dalam perangkat lunak, kemudian ditransformasi oleh elemen – elemen pemrosesan, dan obyek – obyek data hasilnya akan mengalir keluar dari sistem atau perangkat lunak (Roger S. Pressman, 2012).

Data Flow Dokumen (DFD) pada dasarnya digambarkan dalam bentuk hirarki, yang pertama sering disebut sebagai DFD level 0 yang menggambarkan sistem secara keseluruhan sedangkan DFD – DFD berikutnya merupakan penghalusam dari DFD sebelumnya.

Tabel 2.1 Simbol Komponen DFD

Nama Komponen	Bentuk Komponen	Keterangan
Entitas		Sumber atau tujuan dari aliran data dari atau ke sistem.
Aliran Data		Menggambarkan aliran data dari sutu proses ke proses lainnya.

Nama Komponen	Bentuk Komponen	Keterangan
Proses		Fungsi yang mentransformaskan data secara umum
Berkas atau tempat penyimpanan		Komponen berfungsi untuk menyimpan data atau file

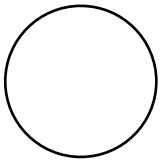
2.4. Context Diagram

Context Diagram adalah bagian dari *Data Flow Dokumen* (DFD) yang berfungsi memetakan model lingkungan, yang dipresentasikan dengan lingkaran tunggal yang mewakili keseluruhan sistem.

Context Diagram menyoroti sejumlah karakteristik penting sistem yaitu diantaranya:

1. Kelompok pemakai, organisasi atau sistem lain dimana sistem melakukan komunikasi (sebagai terminator).
2. Data masuk, yaitu data yang diterima sistem dari lingkungan dan harus di proses dengan cara tertentu.
3. Data keluar, yaitu data yang dihasilkan sistem dan diberikan ke dunia luar.
4. Penyimpanan data (*storage*), yaitu digunakan secara bersama antara sistem dengan terminator. Data ini dapat di buat oleh sistem dan digunakan oleh lingkungan atau sebaliknya.
5. Batasan, antara sistem dan lingkungan.

Tabel 2.2 Simbol komponen *Context Diagram*

Nama Komponen	Bentuk Komponen	Keterangan
Entitas		Sumber atau tujuan dari aliran data dari atau ke sistem.
Aliran Data		Menggambarkan aliran data dari sutu proses ke proses lainnya.
Proses		Fungsi yang mentransformaskan data secara umum

2.5. Diagram HIPO

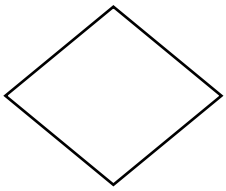
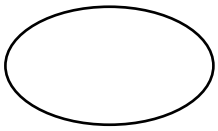
HIPO (Hirarchy Plus Input Process Output) yaitu alat bantu yang digunakan untuk membuat spesifikasi program yang merupakan struktur yang berisi diagram dimana di dalam program ini berisi input yang diproses dan menghasilkan output (Praptingsih, 2012).

2.6. ERD

ERD (*Entity Relationship Diagram*) adalah diagram yang memperlihatkan entitas-entitas yang terlibat dalam suatu sistem serta hubungan-hubungan atau relasi antar entitas tersebut. Model *Entity-Relationship* yang berisi komponen-komponen himpunan entitas dan relasi yang masing-masing dilengkapi dengan *atribut-atribut* yang merepresentasikan seluruh fakta yang ditinjau, dapat digambarkan dengan

lebih sistematis dengan menggunakan diagram *Entity-Relationship* (Fathansyah, 2012).

Tabel 2.3 Simbol Komponen ERD

Nama Komponen	Bentuk Komponen	Keterangan
Entitas		Suatu objek yang dapat diidentifikasi dalam lingkungan pemakai.
Relasi		Menunjukkan adanya hubungan di antara sejumlah entitas yang ada.
Atribut		Mendeskripsikan karakter entitas (atribut yang berfungsi sebagai key di beri garis bawah).
Garis		Penghubung antar relasi dengan entitas, relasi dan entitas dengan atribut.

2.7. PHP

PHP (*Hypertext Preprocessor*) merupakan bahasa pemrograman web yang menggunakan prinsip *server side* paling terkenal di dunia (Priyanto Hidayatullah, 2015).

PHP (atau resminya PHP: Hypertext Proprosessor) adalah skrip bersifat *server-side* yang ditambahkan ke dalam HTML. (Yeni Kustiyaningsih, 2011)

2.8. MySQL

MySQL (*My Structure Query Language*) adalah sebuah perangkat lunak sistem manajemen basis data SQL DBMS (*Database Management System*) dari sekian banyak DBMS, seperti Oracle, MySql, PostagreSql, dan lain-lain. MySql merupakan DBMS yang multi thread, multi user yang bersifat gratis dibawah lisensi *General Public License* (GNU).

Database adalah Strukturpenyimpanan data. Untuk menambah, mengakses dan memperoses data yang disimpandalam sebuah database komputer, diperlukan sistem manajemen database seperti MYSQL Server. (Yeni Kustiyaningsih, 2011)