BAB 2
LANDASAN TEORI

2.1 Optimasi

Optimasi atau optimalisasi adalah proses, cara dan perbuatan untuk mengoptimalkan (menjadikan paling baik, paling tinggi, dan sebagainya) [4]. Sedangkan dalam [5], "*Optimization is the process finding the best solution to some problem where "best" accords to certain criteria*".

Jadi, optimasi atau optimalisasi adalah sebuah proses, cara dan perbuatan (aktivitas/kegiatan) untuk mencari solusi terbaik dalam beberapa masalah yang sesuai dengan kriteria tertentu. Salah satu contoh kasus optimasi adalah penjadwalan.

2.2 Penjadwalan

Penjadwalan adalah pengaturan waktu dari suatu kegiatan. Penjadwalan mencakup kegiatan mengalokasikan fasilitas, peralatan ataupun tenaga kerja bagi suatu kegiatan dan menentukan urutan pelaksanaan kegiatan. Dalam hirarki pengambilan keputusan, penjadwalan merupakan langkah terakhir sebelum dimulainya kegiatan [6].

Kesimpulan yang dapat diambil adalah penjadwalan memiliki fungsi-fungsi mengalokasikan sumber-sumber yang ada guna mencapai tujuan yang telah ditetapkan serta melakukan pengendalian dan koreksi terhadap penyimpangan-penyimpangan yang muncul sehingga kegiatan dapat diselesaikan tepat waktu sesuai dengan rencana yang telah disiapkan. Contoh penjadwalan diantara lainnya adalah penjadwalan kuliah.

2.3 Penjadwalan Kuliah

Penjadwalan kuliah dapat diartikan sebagai penjadwalan sejumlah mata kuliah dimana untuk setiap mata kuliah tersebut diberikan sejumlah ruangan dan periode waktu penyelenggaraan perkuliahan serta kendala atau batasan yang lebih sering disebut dengan *constraint* yang harus dipenuhi [7]. *Constraint* ini dapat

dikategorikan sebagai *hard constraint* dan *soft constraint*, dimana hard constraint dalam optimasi penjadwalan merupakan batas-batas yang harus diterapkan dan terpenuhi oleh sebuah jadwal.

Sebuah solusi hanya dapat dikatakan sahih dan valid apabila dalam solusi tersebut sama sekali tidak ada *hard constraint* yang terlanggar. Berbeda dengan *hard constraint*, *soft constraint* merupakan batasan yang tidak selalu dapat terpenuhi namun jadwal yang dihasilkan harus semaksimal mungkin berusaha memenuhi *soft constraint* [8]. Untuk menghasilkan jadwal kuliah yang optimal maka diperlukan adanya algoritma dalam optimasi penjadwalan.

2.4 Algoritma

Algoritma adalah sekumpulan aturan-aturan berhingga yang memberikan sederetan operasi untuk menyelesaikan suatu jenis masalah yang khusus [9]. Sedangkan dalam [10] algoritma adalah urutan langkah-langkah logis penyelesaian masalah yang disusun secara sistematis.

Jadi algoritma adalah sekumpulan aturan-aturan berhingga yang dilaksanakan dalam urutan langkah-langkah logis untuk menyelesaikan suatu jenis masalah yang khusus dalam susunan sistematis. Dalam penelitian ini algoritma yang digunakan termasuk dalam *swarm inteligence*.

2.5 Swarm Intelligence

Swarm Intelligence adalah salah satu teknik kecerdasan buatan yang berbasis pada perilaku kolektif sistem terdesentralisasi yang dapat mengatur dirinya sendiri. Pengimplementasian *swarm intelligence* biasanya berupa sebuah populasi yang beranggotakan agen-agen sederhana yang berinteraksi secara lokal dengan sesama anggota populasi dan juga berinteraksi secara global dengan lingkungan.

Walaupun biasanya tidak memiliki struktur kendali terpusat yang memberi komando bagaimana masing-masing agen harus bertindak, interaksi secara lokal antar anggota populasi cenderung menuju pada suatu pembentukan perilaku global [11]. Dalam hal ini swarm inteligence-nya terinspirasi dari perilaku kawanan lebah madu dalam mencari makan.

2.6 Perilaku Kawanan Lebah Madu

Model sederhana dari perilaku kawanan lebah madu dalam pencarian makanan mengarah pada munculnya kecerdasan secara kolektif dari kawanan lebah madu yang terdiri dari tiga komponen penting : sumber makanan, lebah pekerja, lebah yang mengganggur dan dua model perilaku : pengerahan ke sumber makanan dan ditinggalkannya sumber makanan.

Nilai dari sumber makanan bergantung pada banyak faktor seperti kedekatannya dengan sarang, kekayaan atau konsentrasi energinya dan kemudahan dalam mengambil energinya. Keuntungan dari sebuah sumber makanan bisa direpresentasikan dengan sebuah angka. Lebah pekerja terasosiasi dengan sumber makanan tertentu yang sedang mereka eksplorasi. Mereka juga membawa informasi dari sumber makanan tersebut, jarak dan arahnya dari sarang, keuntungan dari sumber makanan tersebut dan membagikan informasinya dengan probabilitas tertentu.

Lebah yang menganggur terus menerus mencari sumber makanan lain untuk dieksplorasi. Ada 2 tipe lebah yang mengganggur : lebah pencari, mencari di lingkungan sekitar sarang untuk sumber makanan baru dan lebah penonton yang menunggu di sarang dan memilih sebuah sumber makanan dari informasi yang dibagikan lebah pekerja. Jumlah rata-rata lebah pencari dengan kondisi normal adalah 5-10% [12]. Model perilaku ini lah yang menginspirasi algoritma artificial bee colony.

2.7 Algoritma Artificial Bee Colony

Algoritma *artificial bee colony* dikemukakan oleh Karaboga dan Basturk pada tahun 2007 dan Karaboga pada tahun 2005. Ide dasar dari algoritma ini adalah meniru perilaku mencari makan (seperti eksplorasi, eksploitasi, pengerahan, dan meninggalkan) dari lebah madu. Dalam algoritma ini diasumsikan bahwa setiap posisi sumber makanan adalah solusi yang mungkin dan jumlah nektar dari sumber

makanan adalah kualitas dari solusi tersebut. Langkah-langkah utama dari algoritma ini menurut [13] adalah :

1. inisialisasi : populasi awal dapat didefinisikan sebagai P (G = 0) dari SN solusi (posisi sumber makanan). Dimana SN menunjukkan ukuran dari lebah pekerja atau lebah pencari. Selain itu, setiap solusi x_{ij}(i = 1, 2,..., SN; j = 1,2,…,D) adalah vektor D-dimensional. Disini, D adalah jumlah dari parameter optimasi.
2. Tempatkan lebah pekerja pada sumber makanan di memori dan memperbaharui sumber makanan yang layak. Untuk menghasilkan posisi kandidat sumber makanan dari yang lama (x_{ij}) di memori, memori lebah pekerja diperbaharui dengan persamaan berikut ini.

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}), \quad j \in \{1, 2, \ldots, D\}, k \in \{1, 2, \ldots, SN\} \wedge k \neq i,$$

Persamaan 2.1

Dimana v_{ij} adalah nilai dimensi sumber makanan layak baru yang dimodifikasi dari nilai sumber makanan sebelumnya (x_{ij}) berdasarkan perbandingan dengan nilai sumber makanan tetangga yang terpilih secara acak dan $\emptyset_{ij}$ adalah angka acak diantara |-1, 1| untuk menyesuaikan produksi dari sumber makanan tetangga disekeliling x_{ij} dan merepresentasikan perbandingan dua posisi sumber makanan secara visual.

3. Tempatkan lebah pencari pada sumber makanan di memori. Lebah pencari memilih nilai probabilitas dari sumber makanan tersebut (p_i) dengan persamaan berikut ini.

$$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n},$$

Persamaan 2.2

Dimana fit_i adalah nilai fitness dari solusi i yang sebanding dengan jumlah nektar dari sumber makanan di posisi tersebut; SN menunjukkan ukuran

dari lebah pekerja atau lebah pencari. Jadi, semakin tinggi nilai fit_i semakin besar probabilitas x_{ij} untuk terpilih.

4. Perbaharui sumber makanan yang layak dengan lebah pencari menggunakan persamaan di langkah 2. Jika nilai fit_i tidak meningkat setelah sejumlah percobaan yang banyaknya sudah ditentukan sebelumnya, biasanya dinamakan *limit.* Sekarang, lebah pekerja atau lebah penonton berubah menjadi lebah pencari.
5. Kirim lebah pencari untuk mencari sumber makanan baru. Proses lebah pencari menjelajahi sumber makanan baru dapat didefinisikan dengan persamaan berikut ini.

$$x_i^j = x_{\min}^j + rand[0,1]\left(x_{\max}^j - x_{\min}^j\right),$$

Persamaan 2.3

Dimana x_{min} dan x_{max} adalah batas bawah dan batas atas masing-masing dari lingkup pencarian di tiap dimensi. Disini nilai dari setiap komponen dalam semua vektor x_i harus diantara rentang | x_{min},x_{max} | untuk mengurangi kemungkinan lebah pencari meninggalkan ruang pencarian.

6. Simpan sumber makanan terbaik yang ditemukan sejauh ini.
7. Periksa apakah kriteria untuk berhenti terpenuhi atau tidak contohnya jumlah iterasi maksimal atau MCN (*Maximum Cycle Number*), jika terpenuhi, maka prosesnya sudah selesai. Jika tidak ulangi algoritmanya kembali.

Dalam penelitian ini, algoritma artificial bee colony diimplementasikan menggunakan pemrograman berbasis objek.

2.8 Pemrograman Berorientasi Objek

Pemrograman berorientasi objek adalah menuliskan *statement* dalam bahasa pemrograman untuk mendefinisikan apa yang setiap objek lakukan, termasuk pesan yang dikirim antar objek [14]. Sedangkan menurut [15] pemrograman berorientasi

objek berarti sebuah teknik pemrograman yang pengembangannya menggunakan terminologi objek, dimana setiap objek memiliki atribut beserta fungsi yang saling berinteraksi satu sama lain seperti halnya objek.

Jadi, pemrograman berorientasi objek adalah teknik pemrograman yang mendefinisikan objek beserta atributnya, apa yang bisa dilakukan dalam bentuk fungsi dan interaksi antar objek menggunakan pesan. Untuk menggunakan pemrograman berorientasi objek analisis yang digunakan harus dengan *unified modelling language.*

2.9 Unified Modelling Language

Unified Modelling Language adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek. Pemodelan (*modelling*) sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajari dan dipahami [16].

Sementara di dalam [17] *unified modelling language* adalah suatu alat bantu yang sangat handal dalam bidang pengembangan yang berorientasi objek. Hal ini disebabkan karena *unified model language* adalah adalah sebuah bahasa pemodelan visual yang memungkinkan pengembang sistem untuk membuat cetak biru atas visi mereka dalam bentuk baku.

Dari beberapa pengertian yang dikemukan diatas, dapat disimpulkan bahwa *unified modelling language* adalah sebuah bahasa pemodelan yang berorientasi objek dan menjadi standar dalam visualisasi, merancang dan mendokumentasi sistem perangkat lunak untuk penyederhanaan permasalahan-permasalahan yang kompleks. Dalam [18] dijelaskan diagram-diagram yang digunakan dalam pemodelan *unified modelling language* antara lain :

1. Use Case Diagram

   Use case diagram secara grafis menggambarkan interaksi antara sistem, sistem eksternal dan pengguna. Dengan kata lain *use case diagram* secara

grafis mendeskripsikan siapa yang akan menggunakan sistem dan dalam cara apa pengguna (*user*) mengharapkan interaksi dengan sistem itu. *Use case* secara naratif menggambarkan sekuensi langkah-langkah dari setiap interaksi.

2. Class Diagram

   Class diagram menggambarkan struktur objek sistem. Diagram ini menunjukkan *class-class* objek yang menyusun sistem dan juga hubungan antara *class* objek tersebut.

3. Sequence Diagram

   Sequence diagram secara grafis menggambarkan bagaimana objek berinteraksi satu sama lain melalui pesan pada sekuensi sebuah *use case* atau operasi.

4. Activity Diagram

   Activity diagram secara grafis menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun *use case*. *Activity diagram* juga dapat digunakan untuk memodelkan aksi yang akan dilakukan saat sebuah operasi dieksekusi, dan memodelkan hasil dari aksi tersebut.

Selain melakukan analisis dan pemodelan menggunakan unified modelling language, aplikasi atau sistem tentu membutuhkan basis data untuk mengelola data yang dibutuhkan maupun untuk menyimpan informasi yang dihasilkan.

2.10 Basis Data

Basis data merupakan kumpulan data (arsip) yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan [19]. Sementara menurut [20] basis data adalah koleksi atau kumpulan data yang mekanis, terbagi, terdefinisi secara formal dan juga dikontrol terpusat pada suatu organisasi.

Jadi, basis data adalah kumpulan data mekanis yang saling berhubungan untuk terbagi, terdefinisi secara formal tanpa pengulangan yang tidak perlu untuk

memenuhi berbagai kebutuhan dan dikontrol terpusat pada suatu organisasi. Untuk mengelola basis data diperlukan adanya *database management system*.

2.11 Database Management System (DBMS)

DBMS adalah pengelolaan basis data secara fisik tidak dilakukan oleh pemakai secara langsung, tetapi ditangani oleh sebuah perangkat lunak atau sistem yang khusus atau spesifik. Sistem ini yang akan menentukan bagaimana data diorganisasi,disimpan, diubah dan diambil kembali.

Disamping itu sistem ini juga menerapkan mekanisme pengaman data, pemakaian data secara bersama, pemaksaan keakuratan atau konsistensi data dan sebagainya. Perangkat lunak yang termasuk DBMS misalnya *dBase II+, dBase IV, FoxBase, RBase, MS-Access, Borland-Paradox* atau *Borland-Interbase, MS-SQL Server, CA-Open Ingres, Oracle,Informix dan Sybase*" [14]. Salah satu contoh *DBMS* adalah *MySQL.*

2.12 MySQL

MySQL adalah salah satu *database management system* (DBMS) dari sekian banyak DBMS seperti *Oracle, MS SQL, Postgre SQL* dan lainnya. *MySQL* berfungsi untuk mengolah basis data menggunakan bahasa *SQL*. *MySQL* bersifat *open source* sehingga kita bisa menggunakannya secara gratis [21]. Sementara menurut [22] MySQL merupakan perangkat lunak yang digunakan untuk membangun database yang sering digunakan di lingkungan linux.

MySQL merupakan software open source yang berarti free untuk digunakan. Selain di lingkungan linux, MySQL juga tersedia di lingkungan windows. Jadi, *MySQL* adalah salah satu perangkat lunak DBMS open source yang bisa digunakan untuk mengolah basis data di lingkungan linux maupun windows. Untuk bahasa pemrograman yang digunakan adalah PHP.

2.13 PHP

Menurut [23] “PHP (PHP Hypertext Prosesor) adalah akronim dari Hypertext Preprocessor, yaitu suatu bahasa pemograman berbasiskan kode-kode (script) yang di gunakan untuk mengolah suatu data dan mengirimkannya kembali ke web browser menjadi kode HTML. Kode PHP mempunyai ciri-ciri khusus, yaitu:

1. Hanya dapat dijalankan menggunakan web server misalnya: Apache.
2. Kode PHP dapat diletakan dan dijalankan di web server.
3. Kode PHP dapat digunakan untuk mengakses data base, seperti: MY SQL, PostgreSQL, Oracle, dan lain-lain.
4. Merupakan software yang bersifat open source.
5. Gratis untuk didownload dan digunakan.
6. Memiliki sistem multiplatform, artinya dapat dijalankan menggunakan sistem operasi apapun, seperti Linux, Unix, Windows, dan lain-lain”. Pengujian yang dilakukan pada sistem yang dibangun menggunakan pengujian *black box*.

2.14 Pengujian Black Box

Menurut [24] “Pengujian *black box* merupakan metode perancangan data uji yang didasarkan pada spesifikasi perangkat lunak. Data uji dibangkitkan, dieksekusi pada perangkat lunak dan kemudian keluaran dari perangkat lunak diuji apakah telah sesuai dengan yang diharapkan.”

Berbeda dengan pengujian *white box*, pengujian *black box* tidak membutuhkan pengetahuan mengenai alur, struktur atau implementasi dari *software under test* (SUT). Karena itu pengujian coba *black box* memungkinkan pengembang perangkat lunak untuk membuat himpunan kondisi input yang akan melatih seluruh syarat – syarat fungsional suatu program.