BAB II

LANDASAN TEORI

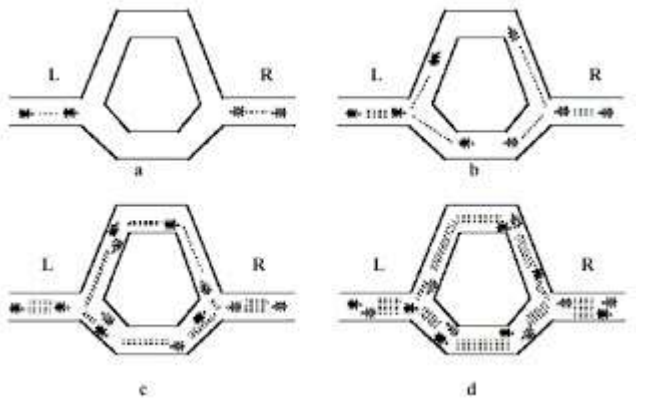
2.1. SISTEM INFORMASI GEOGRAFIS

SIG sebagai sistem informasi yang digunakan untuk memasukkan, menyimpan, memanggil kembali, mengolah, menganalisa dan menghasilkan data bereferensi geografis atau data geospatial, untuk mendukung pengambilan keputusan dalam perencanaan dan pengelolaan penggunaan lahan, sumber daya alam, lingkungan, transportasi, fasilitas kota, dan pelayanan umum lainnya. (Elly, 2009).

2.2. ALGORITMA ANT COLONY OPTIMIZATION

Algoritma Semut diadopsi dari perilaku koloni semut yang dikenal sebagai sistem semut (Dorigo, 1996). Secara alamiah koloni semut mampu menemukan rute terpendek dalam perjalanan dari sarang ke tempat-tempat sumber makanan. Koloni semut dapat menemukan rute terpendek antara sarang dan sumber makanan berdasarkan jejak kaki pada lintasan yang telah dilalui. Semakin banyak semut yang melalui suatu lintasan, maka akan semakin jelas bekas jejak kakinya. Hal ini akan menyebabkan lintasan yang dilalui semut dalam jumlah sedikit, semakin lama akan semakin berkurang kepadatan semut yang melewatinya, atau bahkan akan tidak dilewati sama sekali. Sebaliknya lintasan yang dilalui semut dalam jumlah banyak, semakin lama akan semakin bertambah kepadatan semut yang melewatinya, atau bahkan semua semut akan melalui lintasan tersebut.(Karaboga, 2012)

Gambar berikut menujukkan perjalanan semut dalam menemukan jalur terpendek dari sarang ke sumber makanan.



Gambar 2.1. Perjalanan semut menemukan sumber makanan.

Dari Gambar 2.1 di atas menunjukkan ada dua kelompok semut yang akan melakukan perjalanan. Satu kelompok bernama L yaitu kepompok yang berangkat dari arah kiri yang merupakan sarang semut dan kelompok lain yang bernama kelompok R yang berangkat dari kanan yang merupakan sumber makanan. Kedua kelompok semut dari titik berangkat sedang dalam posisi pengambilan keputusan jalan sebelah mana yang akan diambil. Kelompok semut L membagi dua kelompok lagi. Sebagian melalui jalan atas dan sebagian melalui jalan bawah. Hal ini juga berlaku pada kelompok semut R. Gambar b dan gambar c menunjukkan bahwa kelompok semut berjalan pada kecepatan yang sama dengan meninggalkan feromon atau jejak kaki di jalan yang telah dilalui.

Feromon yang ditinggalkan oleh kumpulan semut yang melalui jalan atas telah mengalami banyak penguapan karena semut yang melalui jalan atas berjumlah lebih sedikit dari pada jalan yang di bawah. Hal ini

dikarenakan jarak yang ditempuh lebih panjang daripada jalan bawah. Sedangkan feromon yang berada di jalan bawah, penguapannya cenderung lebih lama. Karena semut yang melalui jalan bawah lebih banyak daripada semut yang melalui jalan atas. Gambar d menunjukkan bahwa semut-semut yang lain pada akhirnya memutuskan untuk melewati jalan bawah karena feromon yang ditinggalkan masih banyak. Sedangkan feromon pada jalan atas sudah banyak menguap sehingga semut-semut tidak memilih jalan atas tersebut. Semakin banyak semut yang melalui jalan bawah maka semakin banyak semut yang mengikutinya.

Demikian juga dengan jalan atas, semakin sedikit semut yang melalui jalan atas, maka feromon yang ditinggalkan semakin berkurang bahkan hilang. Dari sinilah kemudian terpilihlah jalur terpendek antara sarang dan sumber makanan. Dalam algoritma semut, diperlukan beberapa variabel dan langkah-langkah untuk menentukan jalur terpendek,(Karaboga,2012) yaitu:

Langkah 1:

Inisialisasi harga parameter-parameter algoritma. Parameter-parameter yang di inisialisasikan adalah :

1. Intensitas jejak semut antar kota dan perubahannya (τij)
2. Banyak kota (n) termasuk koordinat (x,y) atau jarak antar kota (dij)
3. Kota berangkat dan kota tujuan
4. Tetapan siklus-semut (Q)
5. Tetapan pengendali intensitas jejak semut (α), nilai $\alpha \geq 0$
6. Tetapan pengendali visibilitas (β), nilai $\beta \geq 0$
7. Visibilitas antar kota = 1/dij (ηij)

8. Banyak semut (m)
9. Tetapan penguapan jejak semut (ρ) , nilai ρ harus > 0 dan < 1 untuk mencegah jejak pheromone yang tak terhingga.
10. Jumlah siklus maksimum (NCmax) bersifat tetap selama algoritma dijalankan, sedangkan τij akan selalu diperbaharui harganya pada setiap siklus algoritma mulai dari siklus pertama (NC=1) sampai tercapai jumlah siklus maksimum (NC=NCmax) atau sampai terjadi konvergensi.

Inisialisasi kota pertama setiap semut. Setelah inisialisasi τij dilakukan, kemudian m semut ditempatkan pada kota pertama tertentu secara acak.

Langkah 2 :

Pengisian kota pertama ke dalam tabu list. Hasil inisialisasi kota pertama setiap semut dalam langkah 1 harus diisikan sebagai elemen pertama tabu list. Hasil dari langkah ini adalah terisinya elemen pertama tabu list setiap semut dengan indeks kota tertentu, yang berarti bahwa setiap tabuk(1) bisa berisi indeks kota antara 1 sampai n sebagaimana hasil inisialisasi pada langkah1.

Langkah 3 :

Penyusunan rute kunjungan setiap semut ke setiap kota. Koloni semut yang sudah terdistribusi ke sejumlah atau setiap kota, akan mulai melakukan perjalanan dari kota pertama masing-masing sebagai kota asal dan salah satu kota-kota lainnya sebagai kota tujuan. Kemudian dari kota kedua masing-masing, koloni semut akan melanjutkan perjalanan dengan memilih salah satu dari kota-kota yang tidak terdapat pada tabuk sebagai

kota tujuan selanjutnya. Perjalanan koloni semut berlangsung terus menerus sampai semua kota satu persatu dikunjungi atau telah menempati tabuk. Jika s menyatakan indeks urutan kunjungan, kota asal dinyatakan sebagai tabuk(s) dan kota-kota lainnya dinyatakan sebagai {N-tabuk}, maka untuk menentukan kota tujuan digunakan persamaan probabilitas kota untuk dikunjungi sebagai berikut :

$P_{ij}^{k} = \frac{[\tau ij]^{\alpha}.[\eta ij]^{\beta}}{\sum[\tau ik]^{\alpha}.[\eta ik]^{\beta}}$ untuk $j \epsilon \{N - tabu_k\}$.....(1)

dan

$P_{ij}^{k} = 0$, *untuk j lainnya*.....(2)

dengan i sebagai indeks kota asal dan j sebagai indeks kota tujuan. Langkah 4 :

a. Perhitungan panjang rute setiap semut.

Perhitungan panjang rute tertutup (length closed tour) atau Lk setiap semut dilakukan setelah satu siklus diselesaikan oleh semua semut. Perhitungan ini dilakukan berdasarkan tabuk masing-masing dengan persamaan berikut :

$L_k = d_{tabu_k(n),tabu_k(1)} + \sum_{s=1}^{n-1} d_{tabu_k}(s), tabu_k(s+1)$.....(3)

dengan dij adalah jarak antara kota i ke kota j yang dihitung berdasarkan persamaan :

$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$.....(4)

b. Pencarian rute terpendek.

Setelah Lk setiap semut dihitung, akan didapat harga minimal panjang rute tertutup setiap siklus atau LminNC dan harga minimal panjang rute tertutup secara keseluruhan adalah atau Lmin.

c. Perhitungan perubahan harga intensitas jejak kaki semut antar kota.

Koloni semut akan meninggalkan jejak-jejak kaki pada lintasan antar kota yang dilaluinya. Adanya penguapan dan perbedaan jumlah semut yang lewat, menyebabkan kemungkinan terjadinya perubahan harga intensitas jejak kaki semut antar kota. Persamaan perubahan ini adalah :

$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$.....(5)

$\Delta\tau_{ij}$adalah perubahan harga intensitas jejak kaki semut antar kota setiap semut yang dihitung berdasarkan persamaan :

$\Delta\tau_{ij}^{k} = \frac{Q}{L_k}$, untuk (i,j) ϵ kota asal dan kota tujuan dalam $tabu_k$ ….....(6)

$\Delta\tau_{ij}^{k} = 0$, untuk (i,j) lainnya. .....(7)

Langkah 5 :

a. Perhitungan harga intensitas jejak kaki semut antar kota untuk siklus selanjutnya. Harga intensitas jejak kaki semut antar kota pada semua lintasan antar kota ada kemungkinan berubah karena adanya penguapan dan perbedaan jumlah semut yang melewati. Untuk siklus selanjutnya, semut yang akan melewati lintasan tersebut harga intensitasnya telah berubah. Harga intensitas jejak kaki semut antar kota untuk siklus selanjutnya dihitung dengan persamaan :

$\tau_{ij} = \rho.\tau_{ij} + \Delta\tau_{ij}$ .....(8)

b. Atur ulang harga perubahan intensitas jejak kaki semut antar kota. Untuk siklus selanjutnya perubahan harga intensitas jejak semut antar kota perlu diatur kembali agar memiliki nilai sama dengan nol.

Langkah 6 :

Pengosongan tabu list, dan ulangi langkah 2 jika diperlukan. Tabu list perlu dikosongkan untuk diisi lagi dengan urutan kota yang baru pada siklus selanjutnya, jika jumlah siklus maksimum belum tercapai atau belum terjadi konvergensi. Algoritma diulang lagi dari langkah 2 dengan harga parameter intensitas jejak kaki semut antar kota yang sudah diperbaharui

2.3. GOOGLE MAPS API

Google Maps adalah layanan gratis yang diberikan oleh Google dan sangat popular. *Google Maps* adalah suatu peta dunia yang dapat kita gunakan untuk melihat suatu daerah. Dengan kata lain, *Google Maps* merupakan suatu peta yang dapat dilihat dengan menggunakan suatu *browser*. Kita dapat menambahkan fitur *Google Maps* dalam web yang telah kita buat atau pada blog kita yang berbayar maupun gratis sekalipun dengan *Google Maps API*. *Google Maps API* adalah suatu *library* yang berbentuk *JavaScript*. (Shodiq,2008)

Cara membuat *Google Maps* untuk ditampilkan pada suatu web atau blog sangat mudah hanya dengan membutuhkan pengetahuan mengenai HTML serta *JavaScript*, serta koneksi Internet yang sangat stabil. Dengan menggunakan *Google Maps* API, kita dapat menghemat waktu dan biaya untuk membangun aplikasi peta digital yang handal, sehingga kita dapat fokus hanya pada data-data yang akan ditampilkan. Dengan kata lain, kita

hanya membuat suatu data sedangkan peta yang akan ditampilkan adalah milik Google sehingga kita tidak dipusingkan dengan mambuat peta suatu lokasi, bahkan dunia.(Wiliam, 2011)

Dalam pembutan program *Google Map API* menggunakan urutan sebagai berikut:

1. Memasukkan Maps API JavaScript ke dalam HTML kita.
2. Membuat element div dengan nama map_canvas untuk menampilkan peta.
3. Membuat beberapa objek literal untuk menyimpan property-properti pada peta.
4. Menuliskan fungsi JavaScript untuk membuat objek peta.
5. Meng-inisiasi peta dalam tag body HTML dengan event onload.

Pada *Google Maps API* terdapat 4 jenis pilihan model peta yang disediakan oleh Google, diantaranya adalah:

1. **Roadmap**, Bentuk ini yang penulis pilih, untuk menampilkan peta biasa 2 dimensi
2. **Satellite**, untuk menampilkan foto satelit
3. **Terrain**,untuk menunjukkan relief fisik permukaan bumi dan menunjukkan seberapa tingginya suatu lokasi, contohnya akan menunjukkan gunung dan sungai
4. **Hybrid**, akan menunjukkan foto satelit yang diatasnya tergambar pula apa yang tampil pada **Roadmap** (jalan dan nama kota)

2.4. PHP

Menurut Octavian (2010) "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 bases, seperti: *MY SQL, PostgreSQL, Oracle*, dan lain-lain.
4. Merupakan *software* yang bersifat *open source*.
5. Gratis untuk di*download* dan digunakan.
6. Memiliki sistem *multiplatform*, artinya dapat dijalankan menggunakan sistem operasi apapun, seperti *Linux, Unix, Windows*, dan lain-lain.

Dengan menggunakan *PHP*, selain memberikan keuntungan seperti pada beberapa point diatas, juga didukung oleh banyak komunitas. Hal ini yang membuat *PHP* terus berkembang. Selain itu, dapat belajar lebih banyak lagi tentang tips dan trik penggunaannya dari berbagai komunitas, lembaga pendidikan, ataupun melalui media internet.(Octavian,2010)

2.5. UML

Dengan menggunakan UML diharapkan penanganan arus data dapat lebih jelas dan terstruktur dengan baik sesuai user yang digunakan. Urutan data UML sebagai berikut

1. Use Case Diagram

Use case diagram digunakan untuk menggambarkan interaksi antara pengguna sistem (*actor*) atau pengguna data yaitu bagian penjualan dan pimpinan dengan kasus (*use case*) data yang diolah seperti pendataan persediaan, proses prediksi penjualan dan penerapan yang disesuaikan dengan langkah – langkah (*scenario*) yang telah ditentukan. Use Case adalah suatu pola atau gambaran yang menunjukan kelakukan atau kebiasaan sistem. Use Case Diagram dibuat untuk memvisualisasikan/ menggambarkan hubungan antara Actor dan Use Case. Use Case diagram mempresentasikan kegunaan atau fungsi-fungsi sistem dari perspektif pengguna.

2. Class Diagram

Class diagram adalah sebuah class yang menggambarkan struktur dan penjelasan class, paket, dan objek serta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

Class juga memiliki 3 area pokok (utama) yaitu : nama, atribut, dan operasi. Nama berfungsi untuk member identitas pada sebuah kelas, atribut fungsinya adalah untuk member karakteristik pada data yang dimiliki suatu objek di dalam kelas, sedangkan operasi fungsinya adalah memberikan sebuah fungsi ke sebuah objek.

3. Activity Diagram

Activity diagram menggambarkan aktifitas - aktifitas, objek, *state*, transisi *state* dan *event*. Activity diagram adalah kegiatan

diagram alur kerja yang menggambarkan perilaku sistem untuk aktivitas.

4. Sequense Diagram

Sequense Diagram menjelaskan secara detail tentang urutan proses yang dilakukan dalam system untuk mencapai tujuan dari *use case*. *Sequense diagram* tersusun dari elemen obyek, interaction dan message. *Interaction* menghubungkan 2 obyek dengan pesannya. Diagram ini menjelaskan aspek dinamis dari system yang sedang dibangun.

5. Componen Diagram

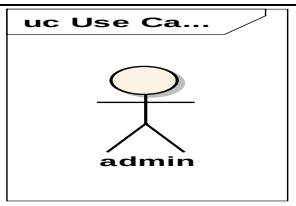
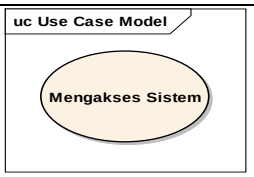
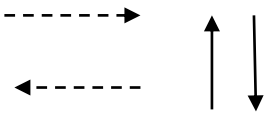
Component Diagram adalah diagram yang menggambarkan sistem pada perangkat lunak yang dipecah menjadi bentuk struktur komponen-komponen yang saling ketergantungan satu dengan komponen lainnya. *Component* diagram menggambarkan struktur fisik dari kode, pemetaan pandangan logis dari kelas proyek untuk kode aktual di mana logika ini dilaksanakan.

6. Deployment Diagram

Deployment Diagram adalah diagram yang menyajikan model perangkat keras yang digunakan dalam penerapan sistemnya dan eksekusi lingkungannya sistem yang digunakan. *Deployment* diagram memberikan gambaran dari arsitektur fisik perangkat lunak, perangkat keras, dan artefak dari sistem. *Deployment* diagram dapat dianggap sebagai ujung spektrum dari kasus penggunaan, menggambarkan bentuk fisik dari sistem yang bertentangan dengan

gambar konseptual dari pengguna dan perangkat berinteraksi dengan sistem.(Rosa,2011)

Tabel 2.1 Simbol-simbol Usecase

No	Simbol	Keterangan
1.		Manusia atau sistem lain yang berinteraksi dengan system
2.		Usecase
3.		Relasi