BAB II

LANDASAN TEORI

Guna mempelajari lebih lanjut dan memudahkan pemahaman dalam penyusunan laporan skripsi, peneliti mengadakan studi kepustakaan mengenai arti dan istilah dalam penelitian laporan skripsi dan tinjauan pustaka dari berbagai jurnal ilmiah, sehingga memudahkan penulis memecahkan suatu masalah yang terdapat dalam suatu penelitian skripsi.

2.1. Sistem

Sistem berasal dari bahasa latin (*systema*) dan bahasa yunani (*sustema*) yang berarti adalah suatu kesatuan yang terdiri dari komponen atau elemen yang saling berhubungan bersama-sama untuk memudahkan aliran informasi, materi atau energi untuk mencapai sutau tujuan. Istilah ini sering dipergunakan untuk menggambarkan suatu set entitas yang berinteraksi, dimana suatu model matematika seringkali bisa dibuat. Sistem juga merupakan kesatuan bagian-bagian yang saling berhubungan yang berada dalam suatu wilayah serta memiliki item-item penggerak, contoh umum misalnya negara. Negara merupakan suatu kumpulan dari beberapa elemen kesatuan antara lain seperti provinsi yang saling berhubungan sehingga membentuk suatu negara dimana yang berperan sebagai penggeraknya yaitu rakyat yang berada di negara tersebut.

Menurut Yakub (2012:1), “Sistem adalah suatu jaringan kerja dari prosedur - prosedur yang berhubungan, terkumpul bersama-sama untuk melakukan suatu kegiatan atau tujuan tertentu”.

2.2. Graph

Graph merupakan kumpulan *vertex* yang dihubungkan satu sama lain melalui sisi atau busur (*edges).* Suatu graph terdiri dari dua himpuan yaitu himpunan *verteks* dan himpunan *edges. Vertek* adalah suatu elemen dari graph yang dapat disajikan berupa titik, lingkaran kecil atau *node.* Sedangkan *edge* adalah suatu elemen dari graph yang disajikan berupa garis.

Edge dapat menunnjukkan hubungan (relasi) sembarang seperti rute penerbangan, jalan raya, sambungan telepon, ikatan kimia, dan lain-lain. *Graph* dinotasikan ddengan G(V,E). pada umumnya graph digunakan untuk memodelkan suatu masalah sehingga menjadi lebih mudah, yaitu dengan cara mempresentasikan objek-objek tersebut. Menurut arah dan bobotnya, *graph* dibagi menjadi empat bagian, yaitu:

1. Graph berarah dan berbobot yaitu graph yang setiap sisinya diberikan orientasi arah dan bobot berupa bilangan bukan negatif.
2. Graph tidak berarah dan berbobot yaitu graph yang setiap sisinya tidak mempunyai orientasi arah tetapi mempuyai bobot.
3. Graph berarah dan tidak berbobot yaitu graph yang setiap sisinya diberikan orientasi arah tetapi tidak berbobot.

4. Graph tidak berarah dan tidak berbobot yaitu graph yang setiap sisinya tidak mempunyai orientasi arah dan tidak berbobot.

2.3. Algoritma Semut

Algoritma ACO telah di perkenalkan oleh Macro Dorigo di awal tahun 1990an. Macro terinspirasi oleh perlakuan semut dalam pencarian makanan. Bagaimana semut dapat menemukan jalur terpendek antara sumber makanan dengan sarang semut. Saat mencari makanan, semut awalnya menjelajahi daerah sekitarnya dengan cara acak (Afrianita & Siska, 2011).

Secara alamiah semut mampu menemukan rute terpendek dalam perjalanan dari sarang ke tempat sumber makanan. Koloni semut dapat menemukan rute berdasarkan jejak kaki pada lintasan yang telah di lalui. Semakin banyak semut yang melalui suatu lintasan, maka akan semakin jelas jejak kakinya. Proses peninggalan jejak kaki yang dilalui semut di namakan *(pheromone).* Jika lintasan yang dilalui semut dalam jumlah sedikit maka semakin lama akan semakin berkurang kepadatan semut. Sebaliknya jika lintasan yang dilalui semut dalam jumlah banyak maka semakin lama semakin bertambah kepadatan semut yang melewatinya atau bahkan semua semut akan melalui lintasan tersebut (Alhanjouri & Alfaran, 2012).

Ant colony optimization (ACO) didasarkan pada perilaku kerjasama dari koloni semut yang dapat menemukan lintasan terpendek dari sarangnya menuju ke sumber makanan.

Proses dalam ACO bisa dijelaskan sebagai berikut. Misalkan ada n semut dalam satu koloni. Semut-semut itu akan memulai suatu gerakan dari sarang mereka ke tujuan akhir melalui beberapa simpul dan berskhir pada simpul tujuan diakhir setiap siklus atau iterasi. Kalau semua semut sudah menyelesaikan lintasannya, jumlah pheromone pada lintasan terbaik secara global akan diperbaharui. Lintasan global terbaik artinya terbaik diantara semua semut. Pada awal proses yaitu pada iterasi 1, semua ruas pada simpul awal akan diberi jumlah pheromone yang sama.

Sehingga pada iterasi 1, semua semut akan mulai dari simpul awal dan berakhir pada simpul tujuan dengan memilih simpul-simpul antara secara random. Proses optimal berakhir jika jumlah iterasi maksimum sudah tercapai atau tidak ada solusi lagi yang lebih baik yang bisa didapat dalam beberapa iterasi yang berurutan.

Seekor semut *k* pada simpul *i* akan memilih simpul *j* yang dituju pada layer berikutnya dengan probabilitas dengan *i* sebagai indeks asal dan *j* sebagai indeks tujuan. Rumus 1 sebagai berikut :

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\Sigma_{l \in N_i^k}[\tau_{il}(t)]^{\alpha}[\eta_{il}]^{\beta}}, if\ j \in N_i^k \\ 0, \qquad\qquad selainnya \end{cases} \qquad \text{…….. (1)}$$

Perhitungan panjang jalur tertutup (*length closed tour*) atau L_k setiap semut dilakukan setelah satu siklus diselesaikan oleh semua semut.

Perhitungan dilakukan berdasarkan *tabu*$_k$ masing-masing dengan persamaan pada rumus 2 berikut :

$$L_k = d_{tabu_k(n),tabu_k(1)} + \sum_{s=1}^{n-1} d_{tabu_k(s),tabu_k(s+1)} \quad \ldots\ldots\ldots \quad (2)$$

Dengan d_{ij} adalah jarak antara kota i ke kota j yang dihitung berdasarkan persamaan rumus 3 sebagai berikut :

$$\text{nilai } d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad \ldots\ldots\ldots\ldots (3)$$

α adalah suatu parameter yang mengedalikan bobot feromon, β adalah parameter pengendali jarak dan $\mathbf{N}_j^k$ adalah himpunan *node-node* yang belum dikunjungi oleh semut *k.*

Setelah L_k setiap semut dihitung, akan diperoleh harga minimal panjang jalur tertutup setiap siklus atau L_{minNC} dan harga minimal panjang jalur tertutup secara keseluruhan adalah L_{min}. Koloni semut akan meninggalkan jejak-jejak kaki pada lintasan antar kota yang dilaluinya. Adanya penguapan dan perbedaan jumlah semut yang lewat, menyebabkan terjadinya perubahan harga intensitas jejak kaki semut antar kota. Persamaan perubahannya pada rumus 4 sebagai berikut :

$$\Delta\tau ij = \sum_{k=1}^{m} \Delta\tau_{ij}^k \quad \ldots\ldots\ldots\ldots\ldots\ldots (4)$$

Dengan $\Delta\tau_{ij}^k$ adalah perubahan harga intensitas jejak kaki semut antar kota setiap semut yang dihitung berdasarkan persamaan rumus 5 sebagai berikut :

$$\Delta\tau_{ij}^{k} = \frac{Q}{L_k} \qquad \text{.....(5)}$$

untuk (i,j) ϵ kota asal dan kota tujuan dalam *tabu*$_k$

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

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: τ_{ij}= ρ. τ_{ij}+ $\Delta\tau ij$.

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

2.4. Java

Java merupakan bahasa pemrograman yang dapat membuat seluruh bentuk aplikasi, dekstop, web, mobile dan lainnya, sebagaimna dibuat dengan menggunakan bahasa pemrograman konvensional yang lain. Bahasa pemrograman java ini berorientasi objek *Object Oriented Programming (OOP),* dan dapat dijalankan pada berbagai platform sistem operasi. Perkembangan java tidak hanya terfokus pada satu sistem operasi , tetapi dikembangkan untuk berbagai sistem operasi dan bersifat open source. Dengan slogannya"*Write once, run anywhere".*

Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam *p-code (bytecode)*

dan dapat dijalankan pada *Mesin Virtual Java* (JVM). Java merupakan bahasa pemrograman yang bersifat umum atau *non-spesifik (general purpose).*

2.5. SQLite

SQLite merupakan sebuah *open source* database yang telah cukup lama, cukup stabil, dan sangat terkenal pada perangkat kecil, termasuk Android. SQLite cocok digunakan sebagai database *android*, maupun berbagai program yang membutuhkan kemudahan serta tidak membutuhkan resource besar. *SQLite* menggunakan konsep *Zero Configuration* dan *embedded* database, yaitu sebuah konsep dimana *developer* tidak perlu menginstall ataupun melakukan setup database kedalam sebuah *server* database sehingga tidak membutuhkan *resourse* yang besar dan proses pembuatan database pun menjadi instan. Selain itu dengan menggunakan konsep *embedded* maka database pada *SQLite* digunakan sebagai *library* dan di *compile* bersamaan dengan program. *SQLite* saat ini banyak digunakan di dalam aplikasi dan program termasuk dalam beberapa *high profile project*. *SQLite* juga merupakan mesin database SQL *embedded* yang berbeda dengan kebanyakan database SQL lainnya seperti yang dijelaskan sebelumnya *SQLite* tidak memiliki proses server yang terpisah sehingga *SQLite* membaca dan menulis secara langsung ke *disk*.

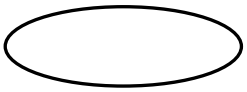
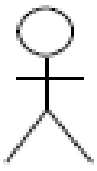
2.6. UML

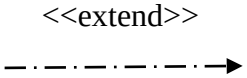
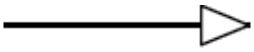
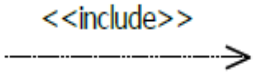
Unified Model Language (UML) merupakan sebuah bahasa untuk menguraikan spesifikasi yang sudah distandarisasikan untuk tujuan permodelan suatu objek. Berikut ini beberapa diagram yang digunakan dalam UML.

1. *Use Case Diagram*

   Use Case Diagram merupakan permodelan untuk kelakuan (*Behaviour*) sistem informasi yangn akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, use *case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informsi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut (Rosa & Shalahuddin, 2013). Simbol *use case diagram* dapat dilihat pada tabel 2.1 sebagai berikut :

Tabel 2.1 Simbol *Use Case Diagram*

No	Nama	Simbol	Keterangan
1	Use Case		Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*.
2	*Actor*		Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari

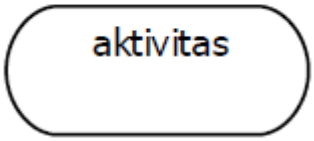
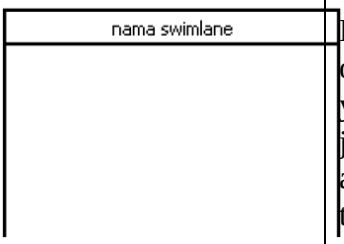
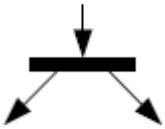
			aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
3	*Assosiation*		Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
4	*Extend*		Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu.
5	*Generalation*		Hubungan generalisasi dan spesialisasi (umum - khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
6	*Include*		Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

2. *Activity Diagram*

Activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis (Shalahuddin, 2011). Simbol *activity diagram* dapat dilihat pada tabel 2.2 sebagai berikut :

Tabel 2.2. Simbol *Activity Diagram*

No	Nama	Simbol	Keterangan
1	Status awal		Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah

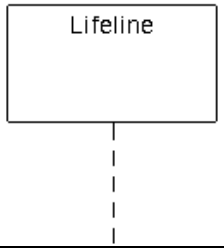
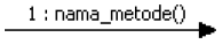
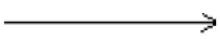
			status awal.
2	Aktivitas		Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
3	Percabanga atau *decision*		Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
4	Status akhir		Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
5	*Swimlane*		Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.
6	*Fork*		*Fork*, digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel.
7	*Join*		*Join*, digunakan utk menunjukkan kegiatan yang digabungkan.

3. *Squence Diagram*

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup *objek* dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar *sequence diagram* maka harus diketahui objek-objek yang terlibat dalam sebuah *use*

case beserta metode-metode yang dimiliki kelas yang di instansiasi menjadi objek itu (Rosa & Shalahuddin, 2013). Simbol *sequence diagram* dapat dilihat pada tabel 2.3 sebagai berikut :

Tabel 2.3 Simbol *Squence Diagram*

No	Nama	Simbol	Keterangan
1	*Actor*		Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
2	*Lifeline*		Merepresentasikan partisipasi dari objek dalam melakukan interaksi.
3	Waktu aktif		Menyatakan objek dalam keadaan aktif dan berinteraksi pesan.
4	Pesan tipe *call*		Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri.
5	Pesan tipe *message*		Menyatakan bahwa suatu objek mengirimkan data/masukan/ informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
No	**Nama**	**Simbol**	**Keterangan**
6	Pesan tipe *return message*		Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan

			suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.

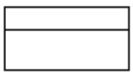
4. *Class Diagram*

Class Diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi (Rosa & Shalahuddin, 2013).

a. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.

b. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas. Simbol *class diagram* dapat dilihat pada tabel 2.4 sebagai berikut.

Tabel 2.4. Simbol *Class Diagram*

No	Nama	Simbol	Keterangan
1	*Generalization*		Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2	*Nary Association*		Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3	*Class*		Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4	*Collaboration*		Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
5	*Realization*		Operasi yang benar-benar dilakukan oleh suatu objek.

No	Nama	Simbol	Keterangan
6	*Dependency*		Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7	*Association*		Apa yang menghubungkan antara objek satu dengan objek lainnya