BAB 2
LANDASAN TEORI

2.1 Optimasi Rute Perjalanan menggunakan Angkot

Masalah perjalanan dengan menentukan rute perjalanan menggunakan angkot adalah sebuah permasalahan optimasi untuk mendapatkan rute terpendek yang harus dilalui oleh seorang penumpang. Penumpang tersebut harus menentukan rute perjalanan angkot yang mana yang akan dia gunakan untuk menuju ke suatu lokasi. Penyelesaian masalah ini dapat diselesaikan dengan menggunakan berbagai algoritma optimasi seperti *Ant Swarm Optimization* (ASO), *Cat Swarm Optimization* (CSO) dan Genetika.

2.2 Algoritma

Algoritma dapat diartikan sebagai metode langkah demi langkah dari pemecahan suatu masalah. Seperti yang dijelaskan oleh Johnsonbaugh [6] langkah – langkah dari suatu algoritma harus dinyatakan dengan jelas sehingga dapat ditulis dalam Bahasa pemograman dan dijalan di komputer. Namun, menurut Suyanto [7] algoritma khusus yang membahas optimasi dapat didefinisikan sebagai metode numerik untuk menemukan nilai x sehingga menghasilkan $f(x)$ yang bernilai terbesar maupun terkecil dengan beberapa batasan pada nilai x.

Algoritma optimasi sendiri terbagi dalam beberapa klasifikasi, yaitu

1. Berdasarkan metode operasinya.
2. Berdasarkan akurasi dan kecepatannya.
3. Berdasarkan analogi yang digunakan.

2.3 Multi-Agent System

Konsep agent ini diperkenalkan oleh seorang peneliti bernama Carl Hewit dengan Concurrent Actor Modelnya pada 1977. Dalam modelnya tersebut hewit mengemukakan teori tentang suatu objek yang dia sebut sebagai Actor. Actor tersebut dapat mempunyai karakteristik menguasai dirinya sendiri, interaktif dan dapat merespon pesan yang datang dari objek lain yang sejenis. Dari penelitian yang berhubungan dengan hal tersebut kemudian lahirlah cabang ilmu besar yang merupakan turunan dari Artificial Intelligence (AI) yaitu Distributed Artificial Intelligence (DAI), yang diantaranya membawahi Multi Agent System (MAS).

Agent memiliki karakteristik yang dapat menbedakan antara agent satu dengan yang lain, dalam Romi SW[8] secara garis besar karakteristik yang dimiliki oleh agent antaralain :

1. Autonomy

   Agent dengan karakteristik ini memiliki sifat yang mandiri, dan tidak dipengaruhi secara langsung oleh user, agent lain, atau lingkungan.

2. Intelligence, Reassoning, dan Learning

   Suatu agent harus memiliki Intelligence atau kecerdasan agar dapat dinamakan agent, dalam konsep ini kemampuan suatu agent yang harus dimiliki antaralain : internal knowledge base, kemampuan reasoning, dan kemampuan learning.

3. Mobility

   Agent mampu bergerak atau mobile untuk melakukan tugas.

4. Delegation

   Agent dibuat dengan tujuan menjalankan tugas yang diperintahkan oleh user, karakteristik ini merupakan yang utama dalam suatu program yang disebut agent.

5. Reactivity

   Kemampuan agent yang lain adalah reaktiv atau kemmpuan untuk cepat beradaptasi terhadap perubahan informasi yang ada pada lingkungan.

6. Proactiove and Goal Oriented

   Agent tidak hanya dituntut bisa beradaptasi pada perubahan lingkungan, tetapi juga mengambil inisiatif langkah penyelesaian apa yang harus diambil.

7. Communication and Coordination Capability

   Agent harus mempunya kemampuan berkomunikasi dengan user dan juga agent lain.

2.4 Cat Swarm Optimization

Cat Swarm Optimization atau yang sering disingkat menjadi CSO, adalah sebuah algoritma yang didasari pada perilaku kucing dalam memburu mangsanya. Algoritma ini diusulkan oleh Shu-Chan Chu dan Pei-Wei Tsai pada tahun 2006 [2]. Pada penggunaan algoritma ini sekumpulan kucing dan perilakunya dapat digunakan untuk menyelesaikan permasalahan optimasi. Ada 2 bagian dalam algoritma ini, yaitu *seeking mode* dan *tracing mode.*

2.4.1 Seeking Mode

Seeking mode adalah pemodelan situasi dimana kucing sedang beristirahat dan mengamati daerah sekitar dan mencari posisi berikutnya untuk mendekati mangsanya. Terdapat empat faktor penting dalam *seeking mode*, yaitu *Seeking Memory Pool* (SMP), *Seeking Range of the selected Dimension* (SRD), *Count of Dimension to Change* (CDC) dan *Self-Positioning Considering* (SPC).

SMP digunakan untuk mendefinisikan ukuran memori pencarian, dengan kata lain adalah posisi yang telah dicoba oleh kucing. Sedangkan SRD adalah sebuah rentang selisih antara nilai baru dengan yang lama, ini berlaku apabila suatu dimensi diputuskan berpindah. CDC memperlihatkan berapa besar dimensi yang akan berubah. Lalu yang terakhir adalah SPC, merupakan variabel Boolean, untuk menentukan apakah suatu titik dapat digunakan sebagai kandidat untuk

bergerak berdasarkan posisi yang pernah kucing singgahi. Tahapan-tahapan *seeking mode* adalah sebagai berikut

1. Buat *j* tiruan posisi kucing saat ini, dimana *j* = SMP. Jika nilai SPC benar maka *j* = (SMP-1), kemudian pertahankan posisi saat ini sebagai salah satu kandidat.
2. Untuk setiap tiruan, berdasarkan dengan CDC, tambahkan atau kurangkan SRD persen dari nilai saat ini secara acak dan gantikan nilai sebelumnya.
3. Hitung nilai *fitness* untuk semua kandidat.
4. Jika semua nilai *fitness* tidak benar-benar sama, maka hitunglah probabilitas terpilih masing-masing titik kandidat dengan menggunakan persamaan (2.1). Jika tidak, tetapkan semua probabilitas pemilihan dari setiap titik kandidat menjadi 1.
5. Secara acak memilih titik untuk berpindah dari titik kandidat, dan mengganti posisi kucing dengan menggunakan persamaan (2.1).

$$P(i) = \frac{(|FS(i) - FS(b)|)}{(FS(max) - FS(min))}$$

(2.1)

Dimana 0<i<j

jika tujuan dari fungsi fitness adalah untuk menemukan solusi minimum, FS(b) = FS(max), jika tidak FS(b) = FS(min).

2.4.2 Tracing Mode / Movement

Tracing mode adalah pemodelan situasi dimana kucing bergerak mendekati mangsanya. Sekali kucing masuk ke mode pelacakan, ia bergerak sesuai dengan kecepatannya sendiri untuk setiap dimensi. Aksi mode pelacakan dapat digambarkan sebagai berikut

1. Perbarui kecepatan untuk setiap dimensi v(k,d) sesuai dengan persamaan (2.2).
2. Periksa apakah kecepatan berada di kisaran kecepatan maksimum. seandainya kecepatan baru melebihi batas, maka kecepatan baru sama dengan batas.

3. Perbarui posisi kucing k sesuai dengan persamaan (2.3).

$$v(k,d) = v(k,d) + r1\,.\,c1\,.\,(x(best,d)\ -\ x(k,d))$$
(2.2)

Dimana x(best, d) adalah posisi kucing, yang memiliki nilai kebugaran terbaik, x(k, d) adalah posisi kucing (k), c adalah konstanta dan r adalah nilai acak dalam kisaran [0,1].

$$x(k,d) = x(k,d) + v(k,d)$$
(2.3)

Sebagaimana telah disebutkan, CSO memiliki dua sub mode, yaitu *Seeking mode* dan *Tracing mode*. Untuk menggabungkan dua mode ini ke dalam algoritma, perlu didefinisikan rasio campuran (MR) yang menentukan penggabungan *Seeking mode* dengan *Tracing mode*.

Dengan mengamati perilaku kucing, dapat diketahui bahwa kucing menghabiskan sebagian besar waktunya beristirahat. Selama beristirahat, kucing mengubah posisinya perlahan dan berhati-hati, terkadang bahkan tetap pada tempatnya. Perilaku inilah yang disebut *Seeking mode.*

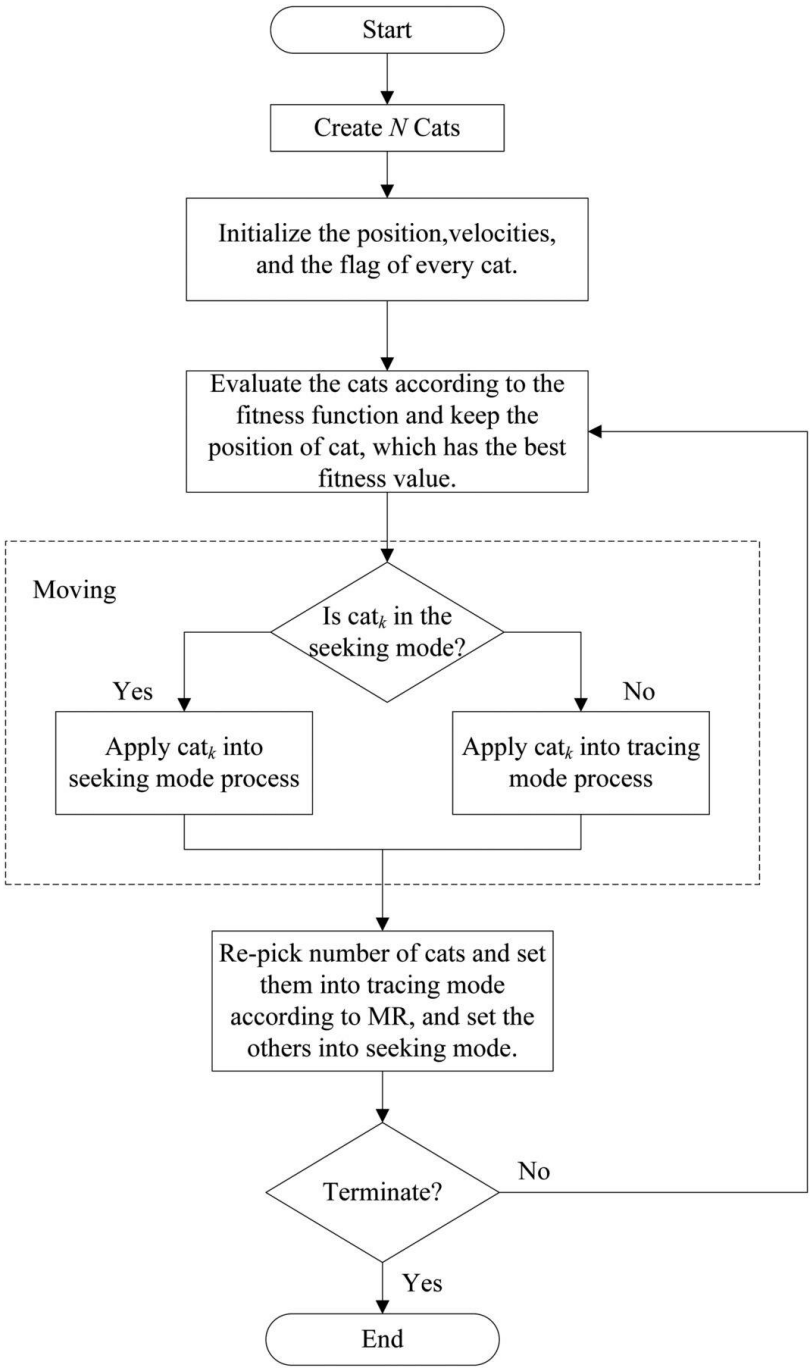
Perilaku saat kucing mengejar target diaplikasikan dalam *Tracing mode.* Karena itu MR harus bernilai kecil untuk memastikan bahwa kucing menghabiskan sebagian waktunya untuk beristirahat seperti di kehidupan nyata.

Proses CSO dapat digambarkan dalam 6 langkah sebagai berikut

1. Bangkitkan m kucing dalam proses.
2. Sebarkan kucing secara acak dalam ruang solusi berdimensi n dan secara acak juga pilih nilai dalam rentang kecepatan maksimum untuk menjadi kecepatan kucing. Kemudian pilih sejumlah kucing secara sembarang dan masukkan dalam *tracing mode* sesuai MR, sisanya dimasukkan ke dalam *seeking mode.*

3. Hitung nilai *fitness* masing-masing kucing dengan memasukkan nilai posisi kucing ke dalam fungsi *fitness*, yang menunjukkan kriteria tujuan dan simpan kucing terbaik ke dalam memori. Perlu diingat bahwa yang perlu disimpan adalah posisi kucing terbaik x(best,d) karena kucing terbaik sejauh ini mewakili solusi terbaik.
4. Pindahkan kucing sesuai benderanya, jika kucing(k) berada dalam *seeking mode*, perlakukan kucing ke proses *seeking mode.* Jika tidak, proses dalam *tracing mode.*
5. Pilih lagi sejumlah kucing dan masukkan dalam *tracing mode* sesuai MR, sisanya masukkan dalam *seeking mode*.

Perhatikan kondisi iterasinya, jika telah terpenuhi hentikan program. Jika tidak ulangi langkah 3-5.



Gambar 2. 1 flowchart cat swarm optimization

2.5 Particle Swarm Optimization

Particle Swarm Optimization (PSO) merupakan salahs atu algoritma metaheuristic yang didasarkan pada perilaku sekawanan burung atau ikan. Algoritma PSO pertama kali dikembangkan oleh James Kennedy dan Russel

Eberhartpada tahun 1995 yang meniru perilaku social sekawanan burung atau ikan. Dalam algoritma PSO burung disebut partikel. James Kennedy dan Russel Eberhart terinspirasi oleh sekawanan burung atau ikan karena partikel ini berperilaku dengan cara menggunakan kecerdasan dan dipengaruhi perilaku kelompok.

Particle Swarm Optimization (PSO) mensimulasikan sekelompok burung atau ikan yang sedang berburu makanannya. Sekelompok partikel tersebut akan bergerak secara acak mencari makanan di sebuah area tertentu. Tetapi hanya ada satu makanan pada area tersebut, dan tujuan kelompok partikel ini untuk menemukan dimana tempat makanan itu. Kelompok partikel ini tidak tahu tepat lokasi makanan berada, hanya saja sekelompok partikel ini dapat mengetahui seberapa jauh makanan tersebut. Strategi terbaik untuk mengetahui dimana lokasi makanan adalah dengan cara mengikuti partikel yang lokasinya paling dekat dengan makanan tersebut. Setiap partikel mewakili solusi dari algoritma PSO. Semua partikel memiliki nilai *fitness* yang dievaluasi oleh fungsi tertentu yang akan dioptimalkan, partikel juga memiliki kecepatan *(velocity)* untuk berpindah tempat. Sehingga algoritma PSO yang diterangkan oleh Santoso[9] adalah sebagai berikut

1. Inisialisasi jumlah partikel, nilai c1 (learning rates eksplorasi lokal) dan c2 (learning rates ekplorasi global), jumlah iterasi, kecepatan awal partikel v(k,d).
2. Evaluasi jarak total dari masing-masing rute yang dihasilkan dari setiap partikel
3. Menentukan posisi terbaik partikel dengan jarak total terkecil dan menetapkan partikel ini sebagai G(best) untuk setiap partikel menggunakan nilai awalnya sebagai P(best).
4. Ulangi langkah berikut sampai kriteria pemberhentian terpenuhi
    a. Gunakan P(best) dan G(best) untuk memperbaharui kecepatan setiap parikel dengan menggunakan rumus

$$v(k,d) = v(k,d) + r1 \,.\, c1 \,.\, (P(best) - x(k,d)) + , r2 \,.\, c2\ (G(best) - x(k,d))$$

(2.4)

dimana G(best) adalah nilai partikel terbaik, P(best) merupakan posisi terbaik dari setiap partikel, c1 dan c2 adalah konstanta bernilai positif, r1, r2 adalah bilangan random yang bernilai 0 sampai 1 dan x(k,d) adalah posisi partikel saat ini.

b. Perbaharui posisi partikel dengan menggunakan rumus

$$x(k,d) = x(k,d) = v(k,d)$$

(2.5)

c. Evaluasi nilai fitness dari setiap posisi baru partikel dengan cara membandingkan x(k,d) dengan G(best) dan juga P(best), apabila x(k,d) lebih baik dari G(best) dan P(best) maka gantikan nilai G(best) dan P(best) dengan nilai x(k,d)
d. Cek kriteria pemberhentian, apabila sudah dipenuhi berhenti, jika tidak kembali ke langkah a.

2.6 Basis Data

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat. Sistem Informasi tidak dapat dipisahkan dengan kebutuhan akan basis data apapun bentuknya, entah berupa *file* teks ataupun *Database Management System* (DBMS)[10].

2.7 DBMS

DBMS (*Database Management System*) atau da;lam Bahasa Indonesia sering disebut sebagai Sistem Manajemen Basis Data adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola, dan menampilkan data[10]. Susatu

sistem aplikasi disebut DMBS jika memenuhi persyaratan minimal sebagai berikut

1. Menyediakan fasilitas untuk mengelola akses data.
2. Mampu menangani integritas data.
3. Mampu menangani akses data yang dilakukan.
4. Mampu menangani backup data.

2.8 SQL

SQL (*Structured Query Language*) adalah Bahasa yang digunakan untuk mengelola data pada RDBMS. SQL awalnya dikembangkan berdasarkan teori aljabar relasional dan kalkulus[10].

2.9 PHP

PHP atau kependekan dari *Hypertext Preprocessor* adalah salah satu Bahasa pemograman *open source* yang sangat cocok atau dikhususkan untuk pengembangan *web* dan dapat ditanamkan pada sebuah skrip HTML. Bahasa PHP dapat dikatakan menggambarkan beberapa bahasa pemograman seperti C, *Java*, dan *Perl* serta mudah untuk dipelajari.

PHP merupakan bahasa *scripting server – side*, dimana pemrosesan datanya dilakukan pada sisi *server*. Sederhananya, *server*lah yang akan menerjemahkan skrip program, baru kemudian hasilnya akan dikirimkan kepada *client* yang melakukan permintaan[11].