Implementasi *Firefly Algorithm-Tabu Search* Untuk Penyelesaian *Traveling Salesman Problem*

Riyan Naufal Hay's
Program Studi Teknik Informatika Fakultas Teknologi Informasi
Universitas Serang Raya
Jalan Raya Serang-Cilegon KM. 5 Taman Drangong Serang, Banten 42116 Indonesia
riyan.unsera@gmail.com

***Abstrak—Traveling Salesman Problem* (TSP) adalah masalah optimasi kombinatorial klasik dan memiliki peran dalam perencanaan, penjadwalan, dan pencarian pada bidang rekayasa dan pengetahuan. TSP juga merupakan objek yang baik untuk menguji kinerja metode optimasi, beberapa metode seperti *Cooperative Genetic Ant System* (CGAS), *Parallelized Genetic Ant Colony System* (PGAS) *Particle Swarm Optimization and Ant Colony Optimization Algorithms* (PSO–ACO), dan *Ant Colony Hyper-Heuristics* (ACO HH) telah dikembangkan untuk memecahkan TSP. Sehingga, pada penelitian ini diimplementasikan kombinasi metode baru untuk meningkatkan akurasi penyelesaian TSP. *Firefly Algorithm* (FA) merupakan salah satu algoritma yang dapat digunakan untuk memecahkan masalah optimasi kombinatorial. FA merupakan algoritma yang berpotensi kuat dalam memecahkan kasus optimasi dibanding algoritma yang ada termasuk *Particle Swarm Optimization*. Namun, FA memiliki kekurangan dalam memecahkan masalah optimasi dengan skala besar. *Tabu Search* (TS) merupakan metode optimasi yang terbukti efektif untuk memecahkan masalah optimasi dengan skala besar. Pada penelitian ini, TS akan diterapkan pada FA (FATS) untuk memecahkan kasus TSP. Hasil FATS akan dibandingkan terhadap penelitian sebelumnya yaitu ACOHH. Perbandingan hasil menunjukan peningkatan akurasi sebesar 0.89% pada dataset Oliver30, 0.14% dataset Eil51, 3.81% dataset Eil76 dan 1.27% dataset KroA100.**

Kata kunci—Firefly Algorithm, Tabu Search, Traveling Salesman Problem

I. PENDAHULUAN

Pada studi kasus penelitian, masalah optimasi menjadi salah satu permasalahan yang sering digunakan dalam pengujian kinerja suatu metode. Beberapa kasus optimasi yang sering digunakan pada penelitian yaitu *Graph Coloring*, *Vehicle Routing* dan *Traveling Salesman Problem*. Namun, pada penelitian ini dipilih masalah optimasi kombinatorial klasik yang masih sering digunakan pada dua dekade terakhir ini, yaitu *Traveling Salesman Problem (TSP).*

TSP merupakan masalah menemukan rute terpendek pada sebuah pencarian, dimana masing-masing kota harus dikunjungi tepat satu kali mulai dari sebuah kota dan kembali lagi pada kota tersebut. Hal ini menjadikan TSP berada dikelas *non-deterministic polynomial-time hard problem*. TSP juga terlibat dalam banyak aplikasi kehidupan nyata [8]. TSP mungkin terlihat sebagai masalah yang sederhana, namun masalah ini merupakan masalah yang penting dan membutuhkan penyelesaian yang konvensional. Hal ini dapat dibuktikan bahwa jika seseorang mencoba menyelesaikan TSP, ia akan membutuhkan waktu lebih lama dibandingkan umur ia hidup di Bumi [21].

Pada Penelitian ini, diajukan metode yang terinspirasi dari perilaku segerombolan kunang-kunang. Metode ini disebut *Firefly Algorithm* (FA), FA sangat efektif dan dapat mengungguli algoritma konvensional lainnya, seperti algoritma genetika, untuk memecahkan banyak masalah optimasi (Nadhir 2013). Namun, pada dasarnya FA memiliki kekurangan dalam penyelesaian optimasi dengan skala besar [6]. Sehingga, dibutuhkan kombinasi metode agar hasil yang diharapkan akan lebih akurat. Tabu Search (TS) merupakan metode optimasi yang terbukti efektif dan sering digunakan untuk memecahkan masalah optimasi dengan skala besar [13][25]. Sehingga, kombinasi FA dan TS (FATS) diharapkan dapat diimplementasikan dalam menyelesaikan TSP, terlebih dapat meningkatkan hasil akurasi penyelesaiannya.

II. TINJAUAN PUSTAKA

A. *Traveling Salesman Problem*

Permasalahan matematika tentang *Traveling Salesman Problem* (TSP) dikemukakan pada tahun 1800 oleh matematikawan Irlandia William Rowan Hamilton dan matematikawan Inggris Thomas Penyngton (Hamilton, 1995). Gambar dibawah ini adalah foto dari permainan Icosian Hamilton yang membutuhkan pemain untuk menyelesaikan perjalanan dari 20 titik menggunakan jalur-jalur tertentu. Diskusi mengenai awal studi dari Hamilton dan Kirkman dapat ditemukan di Graph Theory 1736-19363 oleh N. L. Biggs, E. K. LLoyd, dan R. J. Wilson, Clarendon Press, Oxford, 1976.

Makalah dikirim: 26 April 2017; Revisi: 22 Juni 2017; Diterima: 22 Juni 2017; Publish : 30 Juni 2017

Gambar 1. Permainan *Icosian Hamilton*

Bentuk umum dari TSP pertama dipelajari oleh para matematikawan mulai tahun 1930. Diawali oleh Karl Menger di Vienna dan Harvard. Setelah itu permasalahan TSP dipublikasikan oleh Hassler Whitney dan Merrill Flood di Princeton. Penelitian secara detail dari hubungan antara Menger dan Whitney, dan perkembangan TSP sebagai sebuah topik studi dapat ditemukan di makalah Alexander Schrijver's "On the history of combinatorial optimization (till 1960).

Permasalahan TSP adalah permasalahan dimana seorang salesman harus mengunjungi semua kota dimana tiap kota hanya dikunjungi sekali, dan dia harus mulai dari dan kembali ke kota asal [7]. Tujuannya adalah menentukan rute dengan jarak total atau biaya yang paling minimum. TSP merupakan salah satu masalah yang paling intensif dalam mempelajari masalah optimasi, dan digunakan sebagai patokan bagi banyak metode optimasi dalam jumlah besar dengan cara yang tepat, dan metode yang mudah untuk diketahui, sehingga beberapa kasus dengan puluhan ribu kota dapat diselesaikan dengan baik [14]. TSP adalah salah satu contoh permasalahan kombinatorial dengan kemungkinan penyelesaian yang sangat banyak.

B. *Firefly Algorithm*

Pada bidang *artificial intelligence* atau kecerdasan buatan ada istilah *swarm intelligence* yang diartikan sebagai desain algoritma atau alat *problem solving* terdistribusi yang terinspirasi oleh perilaku sosial kolektif koloni serangga dan koloni binatang [41]. FA merupakan salah satu algoritma *swarm intelligence* yang berkembang sangat cepat pada hampir semua bidang permasalahan optimasi dan perancangan [6].

Algoritma ini dikembangkan oleh Dr Xin-She Yang di Universitas Cambridge pada tahun 2007. Formulasi umum dari algoritma ini disajikan bersama-sama dengan pemodelan matematika analisis untuk memecahkan masalah dengan tujuan ekuivalen fungsi.

Dr. Xin-She Yang merumuskan 3 persamaan yang dibuat dalam hal perilaku pergerakan kunang- kunang yaitu:

1. Intensitas Cahaya

   Ada dua hal yang berkaitan dan sangat penting dalam firefly algorithm yaitu itensitas cahaya dan fungsi keatraktifan. Dalam hal ini banyak dari kita berasumsi bahwa keatraktifan dipengaruhi oleh tingkat itensitas cahaya. Untuk kasus yang paling sederhana contohnya masalah optimasi maksimum, tingkat itensitas cahaya pada sebuah kunang-kunang x dapat dilihat sebagai,

$$I(x) = f(x) \quad (1)$$

   Dengan nilai I merupakan tingkat itensitas cahaya pada x kunang-kunang yang sebanding terhadap solusi fungsi tujuan permasalahan yang akan dicari f(x). Keatraktifan β yang bernilai relatif, karena itensitas cahaya yang harus dilihat dan dinilai oleh kunang-kunang lain. Dengan demikian, hasil penilaian akan berbeda tergantung dari jarak antara kunang-kunang yang satu dengan yang lainnya rij. Selain itu, itensitas cahaya akan menurun dilihat dari sumbernya dikarenakan terserap oleh media contohnya udara γ [40]. Fungsi keatraktifan ialah sebagai berikut [40]:

$$\beta(r) = \beta_0 * e^{(-\gamma r^m)}, \quad (m \geq 1) \quad (2)$$

2. Jarak

   Jarak antara dua kunang-kunang setiap *i* dan *j*, pada posisi *xi* dan *xj*, masing-masing, dapat didefinisikan sebagai jarak *Cartesian* atau *Euclidean* sebagai berikut:

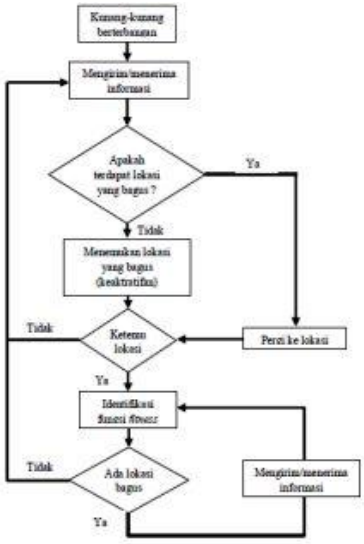
$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad (3)$$

3. Pergerakan Kunang-kunang

   Pergerakan *i* kunang-kunang yang tertarik oleh, lebih terangnya kunang-kunang *j* diberikan oleh persamaan berikut:

$$x_i = x_i + \beta_0 * \exp\left(-\gamma r_{ij}^2\right) * (x_j - x_i) + a * \left(\text{rand} - \frac{1}{2}\right) \quad (4)$$

Langkah-langkah penerapan FA dapat dilihat pada *flowchart* di bawah ini.



Gambar 2. *Flowchart Firefly Algorithm*

C. Tabu Search

Tabu Search (TS) merupakan single-solution based metaheuristik yang diperkenalkan oleh Fred Glover pada tahun 1986. TS sangat popular ditahun 90an, dan sampai sekarang masih menjadi salah satu *single-solution based metaheuristik* yang banyak dipakai untuk menyelesaikan permasalahan optimisasi kombinatorial [13]. TS adalah teknik *local search* yang memilih langkah berikutnya (*neighbor solution*) berdasarkan *constraint dan penalty* (Glover, 1989). Setiap *constraint* yang ada akan didefenisikan dengan sebuah nilai pinalti yang akan dikenakan apabila *constraint* itu terlanggar. Metode ini menggunakan *Tabu List* untuk menyimpan sekumpulan solusi yang baru saja dievaluasi.

Selama proses optimasi, pada setiap iterasi, solusi yang akan dievaluasi akan dicocokkan terlebih dahulu dengan isi *Tabu List* untuk melihat apakah solusi tersebut sudah ada pada *Tabu List*. Apabila solusi tersebut sudah ada pada *Tabu List*, maka solusi tersebut tidak akan dievaluasi lagi pada iterasi berikutnya. Sejumlah pengembangan yang baru dari TS memungkinkan sebuah solution diterima kembali menjadi current solution apabila nilai pinalty keseluruhan jauh lebih kecil daripada solusi sebelumnya menurut batasan-batasan tertentu.

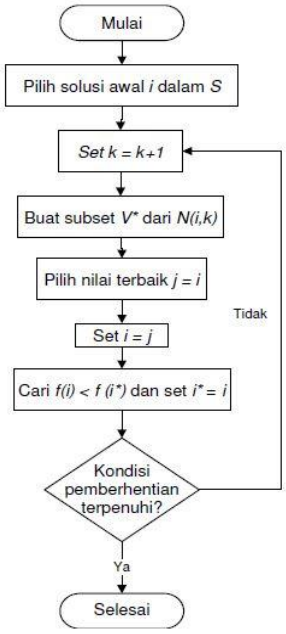
Beberapa elemen utama pada TS adalah sebagai berikut:

1. Representasi solusi: setiap solusi yang mungkin pada suatu permasalahan optimasi harus direpresentasikan.
2. Fungsi *cost*: setiap fungsi *cost* akan memetakan setiap solusi yang mungkin ke nilai *cost*-nya
3. *Neightbourhood* (tetangga): suatu fungsi yang memetakan setiap solusi yang mungkin kesolusi yang lain.
4. *Tabu list* (memori jangka pendek): yaitu memori untuk menyimpan jumlah solusi yang terbatas yang memungkinkan terjadinya perulangan.
5. *Aspiration criteria*: yaitu elemen untuk mencegah proses pencarian mengalami stagnasi (terhambat) karena adanya proses pengujian yang disertai *tabu move*.
6. *Long term memory* (momori jangka panjang): yaitu elemen untuk menyimpan atribut solusi yang akan digunakan dalam *intensification* (untuk memprioritaskan pada atribut dari satu set solusi) dan diversification (untuk memperkecil atribut solusi ketika dipilih untuk memperluas pencarian solusi).

Tahap Algoritma TS secara umum sebagai berikut:

1. Pilih solusi i yang mungkin dalam *S. Set i* = i* dan *k = 0*
2. Tetapkan *k*=*k*+1 dan hasilkan himpunan bagian *V** dari solusi dalam hipunan Solusi *N* (*i*, k).
3. Pilih Solusi terbaik j dalam himpunan bagian *V**. Tetapkan *i = j*.
4. Jika $f(i) \leq f(i^*)$ maka tetapkan *i*= i*.
5. Jika kondisi berhenti terpenuhi maka pencarian berhenti. Jika tidak, Lakukan langkah 2.

Dimana secara bagan dapat dilihat seperti pada gambar 3 berikut ini.



Gambar 3. *Flowchart Tabu Search*

III. PENELITIAN TERKAIT

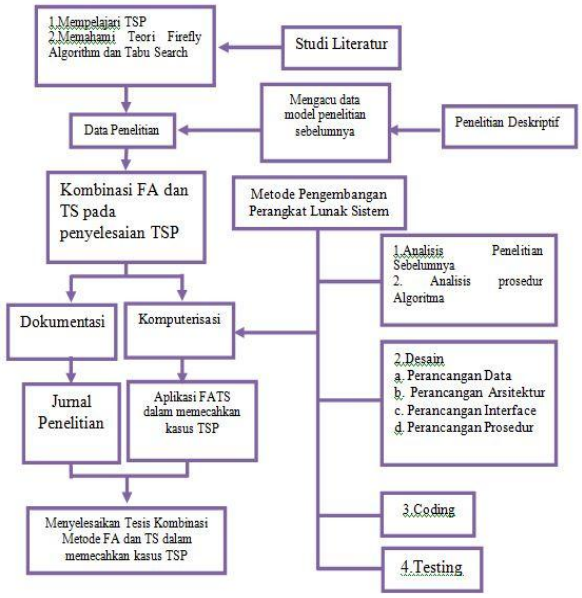
Pada studi kasus TSP, beberapa penelitian baru-baru ini sudah banyak menggunakan metode optimasi untuk membandingkan tingkat kekurangan dan kelebihan algoritma tersebut. Beberapa contoh penelitian seperti, *Cooperative Genetic Ant System* (CGAS) [11] dan *Ant Colony Hyper-heuristics* (ACO HH) [8]. CGAS merupakan metode yang menggabungkan antara Algortima Genetika dan *Ant System*. CGAS merupakan metode yang dapat memecahkan masalah TSP dengan skala besar yaitu dataset (KroA200) [11].

Pada penelitian selanjutnya, TSP diangkat menjadi sebuah objek dalam menguji tingkat akurasi metode yaitu ACO HH. Metode ini merupakan perkembangan dari Algoritma *Ant Colony*, dimana ACO HH dirancang khusus untuk menaikan hasil yang optimal. Algoritma ACO HH hanya mengganti set beberapa nilai heuristik dan nilai fungsi evaluasi tergantung pada masalah yang akan dipecahkannya [8]. Namun pada penelitian ini, fokus pada perbandingan hasil penelitian terbaru yaitu menggunakan metode ACO HH [8].

IV. PERANCANGAN

A. Perancangan Penelitian

Rancangan Penelitian merupakan tahap dimana peneliti merancang sistematika penyelesaian studi kasus terhadap metode yang digunakan. Penelitian kombinasi algoritma FA dan TS dalam menyelesaikan kasus TSP ini selanjutnya dapat dilihat pada gambar 4.

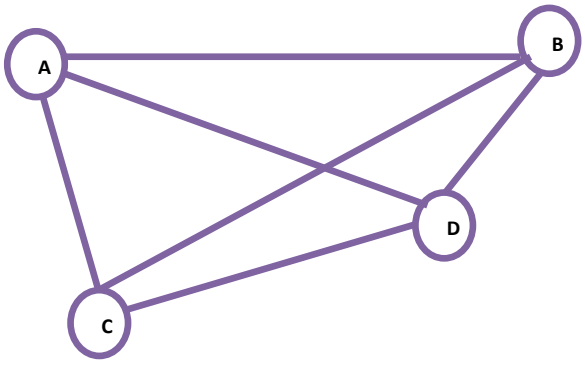


Gambar 4. Rancangan Penelitian

Sebagai langkah awal penelitian, memahami permasalahan TSP yaitu mencari rute terpendek dengan cara mengunjungi seluruh kota tepat satu kali dan kembali pada kota awal. Kemudian memahami peran pada kasus TSP yang sering digunakan dalam hal perancangan, penjadwalan dan pencarian. Sehingga kasus ini menjadikan TSP kedalam kelas masalah kombinasi yang sulit dipecahkan. Maka diperlukan suatu metode konvensional untuk memecahkan kasus TSP. Hal tersebut membuat keterkaitannya dengan metode yang harus dipelajari.

B. Contoh Penyelesaian TSP

Traveling Salesman Problem (TSP) merupakan masalah pencarian rute terpendek dengan kunjungan tepat satu kali dan kembali ke kota awal. TSP memiliki koordinat titik dalam mengasumsikan letak tiap kota. Dapat dilihat sebagai contoh pada gambar di bawah ini.



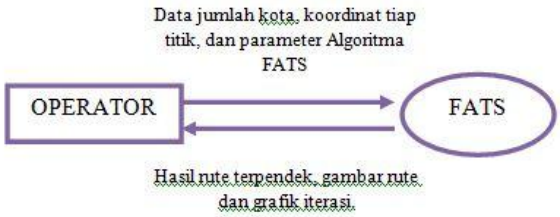
Gambar 5. Contoh TSP dengan 4 Jumlah Kota

Contoh di atas merupakan TSP dengan 4 jumlah kota, dimana tiap kota dapat saling terhubung satu sama lain. Diasumsikan kota A sebagai kota awal dimana pengunjung harus pergi tepat satu kali dan kembali ke kota awal. Pemecahan kasus ini dapat dipecahkan dengan mengambil semua kemungkinan jalur yang dipilih.

- Rute I
  A – B – C – D – A, yaitu 9 + 10 + 7 + 8 = 34 km.
  A – D – C – B – A, yaitu 8 + 7 + 10 + 9 = 34 km.
- Rute II
  A – B – D – C – A, yaitu 9 + 3 + 7 + 5 = 24 km.
  A – C – D – B – A, yaitu 5 + 7 + 3 + 9 = 24 km.
- Rute III
  A – C – B – D – A, yaitu 5 + 10 + 3 + 8 = 26 km
  A – D – B – C – A, yaitu 8 + 3 + 10 + 5 = 26 km.

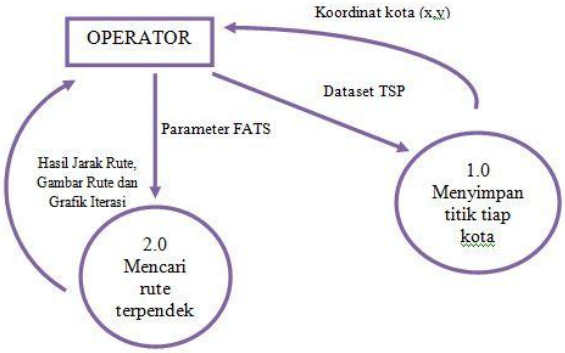
C. Pemodelan Kebutuhan

1. *Context Diagram* (DFD Level 0)
   Pada tahap ini operator memasukan data koordinat tiap kota kemudian menentukan parameter jumlah iterasi dan jumlah populasi Firefly Algorithm. Hasil aplikasi adalah hasil rute terpendek, gambar rute dan grafik iterasi yang dihasilkan pada tiap dataset kasus TSP.



Gambar 6. *Context Diagram* FATS

2. Data Flow Diagram (DFD) Level 1



Gambar 7. DFD Level 1 FATS

Pada gambar 7 di atas, terdapat dua proses utama pada aplikasi menigkatkan akurasi penyelesaian TSP. Proses pertama yaitu menyimpan titik tiap kota dimana koordinat yang digunakan merupakan masukan operator dari dataset TSP. Pada proses yang kedua yaitu pencarian rute terpendek dengan masukan parameter FATS. Hasil dari proses kedua ini merupakan keluaran berupa jarak terpendek kunjungan tiap kota satu kali dan diasumsikan satuan jarak pada rute terpendek ini adalah km. Selain itu, gambar rute terpendek dan grafik iterasi dengan jarak yang optimal akan menjadi keluaran hasil dari pencarian aplikasi ini.

D. Penyelesaian TSP Menggunakan FATS

1. Data Parameter FATS

Data parameter FATS dalam pemecahan TSP dapat dilihat pada tabel 1 di bawah ini.

Tabel 1. Data Parameter FATS

No	Parameter FATS	Lambang	Nilai
1	Jumlah populasi firefly	*nFirefly*	bilangan positif
2	Jumlah iterasi	*Maxit*	bilangan positif
3	Tingkat itensitas cahaya awal	*Intens*	fungsi awal
4	Media penyerapan (udara)	*gamma*	0.98
5	Variabel nilai random pergerakan	α	0.2
6	Tabu Length	TL	Round 0.5
7	Fungsi pergerakan	*Mov*	Rand

Pada tabel di atas memaparkan parameter FATS beserta lambangnya yang digunakan untuk meningkatkan akurasi penyelesaian TSP. Jumlah *firefly* dan iterasi dapat dimasukan dengan nilai berapapun, namun diharuskan bernilai positif kecuali nol. Sedangkan tingkat itensitas cahaya awal, media penyerapan (udara), variabel nilai *random* pergerakan, fungsi pergerakan dan tabu length bernilai konstan dan telah ditetapkan pada FA dan TS.

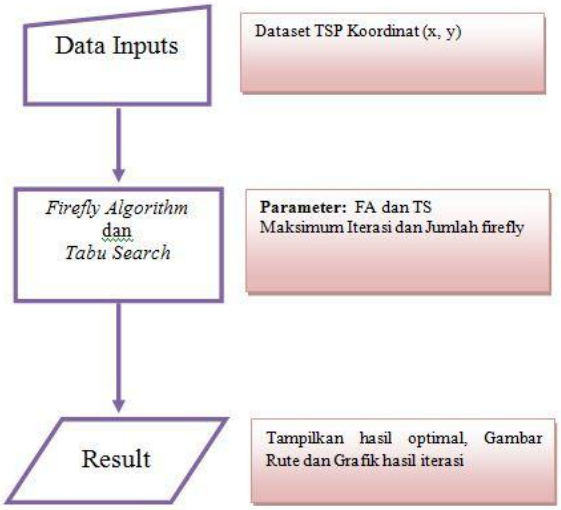
2. Algoritma FATS Menyelesaikan TSP

Algoritma secara umum FATS dalam memecahkan kasus TSP.

a. Inisialisai jumlah populasi FA dan jumlah iterasi.
b. Inisialisasi parameter FA dan TS.
c. Tentukan fungsi *fitness* sebagai fungsi tujuan.
d. Lakukan *update* pada fungsi *fitness* FA dan TS. Pada metode FA, fungsi yang digunakan berupa fungsi intensitas, jarak dan pergerakan. Dimana, kunang-kunang *i* diibaratkan kota awal dan kunang-kunang *j* merupakan kota tujuan. Kunang-kunang dengan tingkat intensitas cahaya yang tinggi akan dipilih. Kemudian update fungsi jarak dan pergerakan. Pada metode TS, cara yang digunakan adalah mengecek satu persatu kota tujuan, dimana kota dengan jalur terdekat akan disimpan pada tabulist. Jika mendapatkan rute yang lebih optimal, maka TS akan melakukan *swap*, yaitu pertukaran data pada *tabulist* sehingga mendapatkan hasil yang optimal.
e. Selanjutnya dilakukan perbandingan fungsi *fitness* FA dan TS yang disimpan dalam *array* matriks.
f. Bandingkan hasil FA dan TS, ambil rute terpendek.
g. Kembali pada tahap ketiga, lakukan sampai batas iterasi atau tercapainya nilai hasil yang optimum.

3. *Flowchart* FATS Menyelesaikan TSP

Flowchart peran FATS dalam memecahkan kasus TSP dapat dilihat pada gambar 8.



Gambar 8. Flowchart TSP Menyelesaikan TSP

V. HASIL PENELITIAN

Pada hasil penelitian, dilakukan 10 kali simulasi dari tiap dataset pada aplikasi penyelesaian TSP dengan jumlah populasi *firefly* dan jumlah iterasi yang berbeda-beda. Dari 10 kali simulasi tersebut akan memperoleh rata-rata hasil yang akan dibandingkan terhadap penelitian sebelumnya.

A. Hasil FATS pada dataset TSP *Oliver30*

Tabel 2. Hasil Rata-Rata FATS Pada Dataset *Oliver30*

No.	Hasil Iterasi		Hasil FATS (km)
	Maxit	nFirefly	
1	100	10	423.7406
2	200	20	423.9117
3	200	50	423.7406
4	300	30	424.6918
5	350	35	424.573
6	400	40	423.9117
7	500	50	423.7406
8	750	75	425.2667
9	1000	100	423.9117
10	2000	200	423.7406
Rata – rata			424.12

Pada tabel di atas dijelaskan, hasil terbaik FATS dalam memecahkan TSP pada dataset Oliver30 yaitu **423.7406**. Dimana hasil terbaik tersebut diperoleh dengan memasukan jumlah iterasi dan populasi *firefly* sejumlah 100-10, 200-50, 500-50 serta 2000-200.

B. Hasil FATS pada dataset TSP *Eil51*

Pada dataset *Eil51*, hasil terbaik yaitu didapatkan dengan jumlah iterasi dan populasi *firefly* yang dimasukan adalah 1000 dan 25. Hasil terbaik pada dataset Eil51 adalah **428.8718**. Hasil keseluruhan 10 simulasi tersebut dapt dilihat pada tabel 3 di bawah ini.

Tabel 3. Hasil Rata-Rata FATS Pada Dataset *Eil51*

No.	Hasil Iterasi		Hasil FATS (km)
	Maxit	nFirefly	
1	200	20	428.9816
2	250	25	428.9816
3	300	30	429.847
4	350	35	429.4841
5	400	40	428.9816
6	500	50	428.9816
7	600	60	428.9816
8	700	70	428.9816
9	1000	25	428.8718
10	2000	50	428.9816
Rata – rata			429.11

C. Hasil FATS pada dataset TSP *Eil76*

Tabel 4. Hasil Rata-Rata FATS Pada Dataset *Eil76*

No.	Hasil Iterasi		Hasil FATS (km)
	Maxit	nFirefly	
1	300	30	551.3177
2	500	50	548.4013
3	600	60	544.3691
4	700	70	552,1403
5	750	75	547.3207
6	800	80	552.3381
7	900	90	547.9541
8	1000	25	547.5294
9	1500	50	547.8551
10	2000	100	544.456
Rata – rata			548.37

Pada dataset Eil76 hasil terbaik FATS diperoleh dengan memasukan jumlah iterasi sebesar 600 dan 60 jumlah populasi *firefly*. Hasil yang diperoleh FATS dalam meningkatkan pemecahan TSP pada dataset Eil76 adalah **544.3691**.

D. Hasil FATS pada dataset TSP *KroA100*

Hasil yang didapatkan FATS dalam memecahkan TSP dataset KroA100 adalah **21307.422**. Pada dataset ini, hasil terbaik diperoleh dengan memasukan jumlah iterasi 800 dan populasi *firefly* 80. Untuk lebih jelasnya, dapat dilihat pada table 5.

Tabel 5. Hasil Rata-Rata FATS Pada Dataset *KroA100*

No.	Hasil Iterasi		Hasil FATS (km)
	Maxit	nFirefly	
1	600	60	21327.8476
2	750	75	21357.1631
3	800	80	21307.4225
4	900	90	21381.3018
5	1000	25	21585.2109
6	1500	50	21559.7298
7	1750	75	21681.2876
8	2000	100	21381.8346
9	2500	150	21433.1677
10	3000	200	21337.1589
Rata – rata			21435.21

E. Hasil Perbandingan Pada Penelitian Sebelumnya

Berdasarkan ruang lingkup penelitian ini, bahwa hasil penelitian akan dibandingkan terhadap penelitian sebelumnya, yaitu *Ant Colony Hyper-Heuristics for Travelling Salesman Problem* (ACO-HH), (Aziz, 2015).

Tabel 6. Hasil Perbandingan Terhadap Penelitian Sebelumnya

Dataset	Hasil ACO-HH	Hasil FATS	Hasil Terbaik Diketahui	Peningkatan Akurasi
Oliver30	427.90	424.12	424	0.89%
Eil51	429.70	429.11	426	0.14%
Eil76	568.89	548.37	538	3.81%
KroA100	21705.70	21435.21	21282	1.27%

Berdasarkan tabel di atas FATS memperoleh peningkatan akurasi pemecahan TSP tiap dataset. FATS memperoleh peningkatan tertinggi pada dataset **Eil76 yaitu sebesar 3.81%**. Sedangkan hasil terendah dalam peningkatan akurasi FATS dalam memecahkan TSP yaitu pada dataset **Eil51 sebesar 0.14%**. Hasil ini diharapkan dapat digunakan dan menjadi inspirasi pada studi kasus optimasi lainnya.

VI. KESIMPULAN DAN SARAN

Penelitian ini telah berhasil mengimplementasikan *Firefly Algorithm* dan *Tabu Search* (FATS) dalam penyelesaian *Traveling Salesman Problem* (TSP). Hasil penelitian ini memiliki hasil rata-rata yang akan dibandingkan terhadap penelitian sebelumnya. Penelitian *Ant Colony Hyper-heuristics for Travelling Salesman Problem* [8] **pada dataset Oliver30 memperoleh peningkatan akurasi sebesar 0.89%, dataset Eil51 sebesar 0.14%, Eil76 sebesar 3.81% dan KroA100 sebesar 1.27%**.

Saran penulis adalah penelitian dengan kasus TSP diharapkan dapat mengambil parameter waktu penyelesaian menjadi hasil perbandingan terhadap penelitian sebelumnya dan diharapkan dapat mencari kombinasi algoritma lainnya yang lebih cepat serta akurat. Sehingga hasil yang diperoleh menjadi lebih optimal dari penelitian sebelumnya.

VII. REFERENSI

[1] Ade Trisnawati, I. B. (2011). Implementasi Tabu Search untuk Penjadwalan Kelas. Seminar Nasional Teknologi Informasi (pp. 1-15). Jakarta: Universitas Tarumanegara.

[2] Adil Hashmi, N. G. (2013). Firefly Algorithm for Unconstrained Optimization. Journal of Computer Engineering , 1-4.

[3] Ahmed Ahmed El-Sawy, E. M.-A. (2013). A Novel Hybrid Ant Colony Optimization and Firefly Algorithm for Solving. Journal of Natural Sciences and Mathematics , 1-22.

[4] Ahmed Ahmed El-Sawy, E. M.-A. (2013). Hybridizing Ant colony Optimization With Firefly Algorithm For Unconstrained Optimization Problems. The Online Journal on Computer Science and Information Technology , 1-9.

[5] Alam, M. N. (2016, March 8). Particle Swarm Optimization: Algorithm and its Codes in MATLAB. pp. 1-11.

[6] Ali, N. (2014). A Review Of Firefly Algorithm. ARPN Journal of Engineering and Applied Sciences , 1-5.
[7] Aulia Rahma Amin, M. I. (2005). Traveling Salesman Problem. Bandung: Departemen Teknik Informatika, Institut Teknologi Bandung.
[8] Aziz, Z. A. (2015). Ant Colony Hyper-heuristics for Travelling Salesman Problem. IEEE International Symposium on Robotics and Intelligent Sensors , 1-5.
[9] Benayad, Z. (2014). A Novel Firefly Algorithm Based Ant Colony Optimization. International Journal of Computer Science and Applications , 1-19.
[10] Buyukozkan, K. (2015). Lexicographic bottleneck mixed-model assembly line balancing. Expert Systems With Applications , 1-16.
[11] Dong, G. (2012). Solving the traveling salesman problem using cooperative genetic ant systems. Expert Systems with Applications , 1-6.
[12] Dr.T.Govindaraj1, V. (2014). Firefly Algorithm for Optimal Power Flow Considering Control Variables. International Journal Of Innovative Research In Electrical, Electronics, Instrumentation And Control Engineering, 1-6.
[13] Eduardo Rodriguez-Tello, H.-M. G.-T. (2014). Tabu Search for The Cyclic Bandwidth Problem. Computers & Operations Research , 1-16.
[14] Elloum, W. (2014). A comparative study of the improvement of performance using a PSOmodified by ACO applied to TSP. Applied Soft Computing , 1-8.
[15] Glove,F, (2012),"Candidate List strategies and Tabusearch", CAAI Research report, University Of Colorado, Boulder.
[16] Isra Natheer Alkallak, R. Z. (2008). Tabu Search Method for Solving the Traveling salesman Problem. Raf. J. of Comp. & Math's. , Vol. 5, No. 2 , 1-13.
[17] Jayaswal, S. (2015). A Comparative Study of Tabu Search and Simulated Annealing for Traveling Salesman Problem. University of Waterloo: Department of Management Sciences.
[18] Katagiri, H. (2012). A hybrid algorithm based on tabu search and ant colony optimization for k-minimum spanning tree problems. Expert Systems with Applications , 1-6.
[19] Kratika Chandra, S. S. (2014). Firefly Algorithm to Solve Two Dimensional Bin Packing Problem. International Journal of Computer Science and Information Technologies , 1-6.
[20] Krishna H. Hingrajiya, R. K. (2012). An Ant Colony Optimization Algorithm for Solving Travelling Salesman Problem. International Journal of Scientific and Research Publications, Volume 2 , 1-6.
[21] Kumbharana, S. N. (2013). Solving Travelling Salesman Problem using Firefly Algorithm. International Journal for Research in Science & Advanced Technologies , 1-5.
[22] Latifa DEKHICI, K. B. (2012). Firefly Algorithm for Economic Power Dispatching With Pollutants Emission. Informatica Economică vol. 16, no 2 , 1-13.
[23] M. K. A. Ariyaratne, T. G. (2014). A Comparative Study on Nature Inspired Algorithms with Firefly Algorithm. International Journal of Engineering and Technology , 1-7.
[24] Mardlijah, A. J. (2013). A New Combination Method Of Firefly Algorithm And T2fsmc For Mobile Inverted Pendulum Robot. Journal of Theoretical and Applied Information Technology , 1-8.
[25] Pedro, O. (2013). A Tabu Search Approach for the Prize. Electronic Notes in Discrete Mathematics , 1-8.
[26] Puspitorini, S. (2011). Penyelesaian Masalah Traveling Salesman. Media Informatika, Vol. 6, No. 1 , 1-17.
[27] Rabindra Kumar Sahu, S. P. (2015). A hybrid firefly algorithm and pattern search technique for automatic. Electrical Power and Energy Systems , 1-15.
[28] Retrieved from MP-TESTDATA - The TSPLIB Symmetric Traveling Salesman Problem Instances: http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/index.html
[29] Retrieved from ScienceDirect: http://www.sciencedirect.com/
[30] Retrieved from SCI-HUB: http://sci-hub.io/
[31] Salari, M. (2015). Combining ant colony optimization algorithm and dynamic. Computers & Industrial Engineering , 1-8.
[32] Sh. M. Farahani, A. A. (2011). A Gaussian Firefly Algorithm. International Journal of Machine Learning and Computing , 1-6.
[33] Singh, S. A. (2013). The Firefly Optimization Algorithm: Convergence Analysis and Parameter Selection. International Journal of Computer Applications , 1-5.
[34] Su, S. (2014). Comparisons of Firefly Algorithm with Chaotic Maps. Computer Modelling & New Technologies , 1-7.
[35] Teams, Y. (2016). Retrieved from Yarpiz: http://yarpiz.com/259/ypea112-firefly-algorithm
[36] The MathWorks, Inc. (2016). Retrieved March 2016, from Mathworks:http://www.mathworks.com/help/optim/ug/traveling-salesman-problem.html
[37] Vittorio Maniezzo, L. M. (2013). Ant Colony Optimizatio. The Future & Emerging Technologies.
[38] Wang, X. (2011). Hybrid Differential Evolution Algorithm for Traveling. Advanced in Control Engineeringand Information Science , 1-5.
[39] Xin-SheYang. (2010). Engineering Optimization. New Jersey: John Wiley & Sons, Inc.
[40] Yang, X.-S. (2010). Firefly Algorithm, Stochastic Test Functions and Design. Department of Engineering, University of Cambridge , 1-12.
[41] Yang, X.-S. (2009). Firefly Algorithms for Multimodal Optimization. pp. 1-10.
[42] Yang, X.-S. (2013). Swarm Intelligence and Bio-Inspired Computation. London: Elsevier.