Implementasi Teknik *Iterative Improvement Heuristic for Flight Path Optimization* dalam Penyelesaian TSP

Adi Purnama
Program Studi Teknik Informatika
Institut Teknologi Bandung
Bandung, Indonesia
13514006@std.stei.itb.ac.id

***Abstrak*— Persoalan pedagang keliling (*Travelling Salesman Problem* - TSP) meruapakan salah satu persoalan optimasi yang aktif dikaji dalam bidang *Computer Science.* Permasalahan ini digolongkan ke kelas persoalan *NP-Hard*. Sampai saat ini, belum ada algoritma baik (algoritma dengan kompleksitas polinomial) untuk menyelesaikan permasalahan ini. Salah satu pendekatan penyelesaian TSP adalah menggunakan teknik heuristik. Pada penelitian ini, penulis mengimplementasikan teknik *Iterative Improvement Heuristic for Flight Path Optimization.* Teknik heuristik ini dikembangkan berdasarkan pengamatan pergerakan lebah dalam menyusun rute optimal saat mengunjungi bunga dan kembali ke sarang.**

Kata Kunci*— *Travelling Salesman Problem* , *Heuristik* , *Iterative Improvement Heuristic

I. PENDAHULUAN

Persoalan pedagang keliling (*Travelling Salesman Problem - TSP)* merupakan salah satu masalah optimasi yang aktif dikaji dalam bidang Matematika dan *Computer Science*. TSP merupakan pengembangan dari permasalahan sirkuit hamilton. Secara singkat, TSP didefinisikan sebagai berikut. Diberikan n buah kota serta diketahui jarak antara setiap kota satu sama lain. Temukan tur terpendek yang dimulai dari sebuah kota , melalui seluruh kota lainnya tepat sekali, dan kembali ke kota asal keberangkatannya.[2]

Meskipun terlihat sederhana, persoalan ini dianggap cukup sulit. Permasalahan ini digolongkan ke kelas persoalan *Nondeterministic Polynomial – Hard (*NP-Hard). Oleh sebab itu, persoalan TSP dengan instans persoalan yang besar membutuhkan waktu komputasi yang besar.

TSP memiliki banyak aplikasi di dunia nyata, diantaranya di bidang logistik, genetika, manufaktur, telekomunikasi , dan neurosains.[3] Namun, aplikasi di dunia nyata biasanya melibatkan instans permasalahan yang besar (banyaknya jumlah "kota" yang harus dikunjungi). Dalam kasus ini, pencarian solusi optimal tidak mungkin untuk dilakukan karena waktu yang dibutuhkan untuk mencari solusi yang lama. Oleh karena itu, digunakanlah pendekatan lain yaitu teknik heuristik.

Penerapan teknik algoritma aproksimasi (heuristik) dalam penyelesaian TSP mungkin tidak menghasilkan solusi yang optimal , namun cara ini diharapkan dapat menyusun solusi TSP dengan waktu yang lebih cepat dan kualitas optimasi yang tidak terlalu buruk. Sifat khusus yang dimiliki oleh heuristik pada TSP adalah kecepatan untuk menemukan solusi dan tingkat kedekatan solusi yang dihasilkan oleh heuristik jika dibandingkan dengan solusi optimal.[4]

II. TRAVELLING SALESMAN PROBLEM

Seperti yang telah dijelaskan sebelumnya, Persoalan pedagang keliling adalah persoalan mencari sirkuit hamilton dengan total bobot sisi paling kecil. Secara matematis, persoalan pedagang keliling didefinisikan sebagai berikut. [5]

A. Fungsi Objektif

Fungsi objektif adalah meminimalisasi bobot simpul

$$F = \min \sum_{i}^{n} \sum_{j}^{n} d_{ij} X_{ij} \qquad (1)$$

Dimana

F : Fungsi objektif
dij : Bobot sisi yang menghubungkan simpul i ke j
Xij : Variabel boolean yang bernilai 1 jika sisi ij dipilih dalam solusi , bernilai nol jika sisi ij tidak dipilih dalam solusi

B. Batasan (*Constraint)*

Setiap sisi hanya dipilih sekali.

$$X_{ij} + X_{ji} \leq 1 \quad \text{(Untuk semua i \& j)} \qquad (2)$$

TSP simetrik , direpresentasikan sebagai graf tak berarah.

$$d_{ij} = d_{ji} \qquad (3)$$

Batasan non-negatif.

$$X_{ij} \geq 0 \quad (4)$$

Seluruh simpul harus dikunjungi.

$$\sum_{i}^{n}\sum_{j}^{n} X_{ij} = 1 \ (Untuk\ semua\ i\ \&\ j) \quad (5)$$

Terdapat beberapa cara untuk menyelesaikan TSP. Namun, sampai sekarang , belum ada algoritma yang cukup baik untuk menyelesaikan TSP. Jack Edmonds mendefinisikanAlgoritma yang baik sebagai berikut.[3] Untuk suatu besaran instans persoalan n, algoritma harus berjalan dalam waktu K n^c , dengan K dan c sebagai konstanta. Konstanta K dan c harus tetap konstan untuk n yang besar. (Kompleksitas O(n^c)).

Cara paling mudah untuk menyelesaikan TSP adalah dengan mengenumerasi seluruh kemungkinan rute yang mungkin, lalu mencari rute dengan jalur terpendek (Algoritma *Brute Force).* Algoritma ini memiliki kompleksitas O(n!). Algoritma ini cukup buruk untuk menyelesaikan persoalan dengan instans yang besar.

Sejauh ini , AlgoritmaHeld-Karp merupakan algoritma terbaik yang menghasilkan solusi umum TSP .[3] Algoritma ini merupakan algoritma program dinamis (*dynamic programming algorithm).* Algoritma ini memiliki kompleksitas O($n^2 2^n$) .

III. HEURISTIK PADA TSP

Sampai saat ini, belum ditemukan algoritma dengan kompleksitas polinomial untuk menyelesaikan TSP. Oleh karena itu, salah satu alternatif solusi untuk menyelesaikan permasalahan TSP di dunia nyata dengan instans yang besar adalah menggunakan heuristik. Heuristik dapat menghasilkan aproksimasi terhadap solusi optimal dengan kompleksitas ruang dan waktu yang dapat diterima.[6]

Biasanya, teknik heuristik digunakan untuk memecahkan persoalan yang tergolong sulit. Suatu persoalan dikelompokan berdasarkan "tingkat kesulitannya" dengan pembagian sebagai berikut [6]

- Kelas P berisi semua permasalahan yang bisa diselesaikan oleh mesin turing deterministik dalam waktu polinomial. Mesin turing deterministik merupakan suatu abstraksi yang menggambarkan algoritma.
- Kelas NP berisi semua permasalahan yang bisa diselesaikan oleh mesin turing non-deterministik dalam waktu polinomial. Karena mesin turing non-deterministik tidak ada, ini berarti ada algortima eksponensial yang dapat menyelesaikan permasalahan ini.
- Kelas *NP-Complete* merupakan subklas dari kelas NP. Berisi permasalahan yang dapat dikonversi menjadi permasalahan kelas NP dalam waktu polinomial.
- Kelas *NP-Hard* merupakan kelas permasalahan NP-Complete atau permasalahan yang lebih sulit lagi. Permasalahan di dalam kelas ini memiliki sifat yang sama dengan kelas NP-Complete , namun tidak harus ada di kelas NP.

A. Jenis Jenis Heuristik pada TSP

Permasalahan yang tergolong pada kelas *NP-Complete* dan *NP-Hard* biasanya diaproksimasi dengan teknik heuristik. Untuk permasalahan TSP , terdapat beberapa golongan teknik heuristik yang dikenal , diantaranya[4] :

1. Heuristik Konstruksi Tur (*Tour Construction)*

   Heuristik golongan ini hanya mengkonstruksi sebuah tur , tanpa melakukan perbaikan tur yang sudah dikonstruksi (untuk meningkatkan efisiensi). Beberapa teknik heuristik yang termasuk di dalam golongan ini adalah :

   a. *Nearest Neighbor Heuristic*
   b. *Greedy Heuristic*
   c. *Insertion Heuristic : Nearest Insertion*
   d. *Insertion Heuristic : Convex Hull*
   e. *Double Minimum Spanning Tree*
   f. *Christofides Algorithm*

2. Heuristik Perbaikan Tur (*Tour Improvement)*

   Heuristik golongan ini menggunakan teknik heuristik konstruksi tur untuk membuat sebuah tur awal. Lalu, tur ini diperbaiki untuk menurunkan total bobot sisi. Beberapa teknik heuristik yang termasuk di dalam golongan ini adalah :

   a. 2-opt , 3 –opt
   b. k-opt
   c. Lin-Kernighan
   d. Tabu Search
   e. Genetic Algorithms

Jadi, terdapat dua aspek heuristik yang harus diperhatikan dalam penggunakan teknik heuristik dalam penyelesaian masalah TSP, yaitu pembangkitan tur awal dan perbaikan tur (*improvement).*

B. Pengukuran Kualitas Heuristik

Terdapat beberapa cara untuk mengukur kualitas heuristik TSP . Sifat – sifat penting dari sebuah heuristik TSP adalah sebagai berikut :

1. Kecepatannya dalam menemukan solusi , sifat ini dapat diukur menggunakan kompleksitas waktu algoritma.
2. Tingkat kedekatannya dengan solusi optimal. Sifat ini dapat diukur menggunakan batas bawah Held-Karp (*Held-Karp lower bound*). Batas bawah Held-Karp merupakan standar *de-facto* untuk menentukan tingkat kedekatan solusi yang dihasilkan oleh heuristik dengan solusi optimal.
3. Rasio kasus terburuk (*Worst case ratio)* . Rasio kasus terburuk adalah rasio solusi yang dihasilkan oleh heuristik dengan solusi optimal. Sebagai contoh, jika rasio kasus terburuk suatu heuristik adalah 2 , maka pada kasus terburuk, panjang lintasan yang dihasilkan oleh heuristik tersebut dua kali lipat dari panjang lintasan optimal.

Berikut adalah kualitas dari beberapa teknik heuristik [4]

a. *Nearest Neighbor Heuristic*

Kompleksitas waktu : $O(n^2)$

Batas bawah held karp: 25%

b. *Greedy Heuristic*

Kompleksitas waktu : $O(n^2 \log_2(n))$

Batas bawah held karp: 20%

c. *Insertion Heuristic : Nearest Insertion*

Kompleksitas waktu : $O(n^2)$

d. *Insertion Heuristic : Convex Hull*

Kompleksitas waktu : $O(n^2 \log_2(n))$

e. *Double Minimum Spanning Tree*

Kompleksitas waktu : $O(n^2 \log_2(n))$

Rasio kasus terburuk : 2

f. *Christofides Algorithm*

Kompleksitas waktu : $O(n^3)$

Rasio kasus terburuk : 3/2

Untuk membandingkan performa heuristik secara tepat, perlu dilakukan eksperimen terhadap masing-masing heuristik dengan *test-case* yang sama. Berikut adalah hasil eksperimen terhadap beberapa jenis heuristik dengan *test-case* 10.000 instans.

TABLE I. PERBANDINGAN BEBERAPA JENIS HEURISTIK PADA PERSOALAN TSP

No	Kode Heuristik	Nama Heuristik	Waktu	Selisih dari Solusi HK
1	Spacefill	Spacefilling Curve	0,02	34,56
2	Strip	Strip Heuristic	0,01	30,75
3	Karp-20	Karp Partitioning Heuristic	0,85	29,34
4	NI	Nearest Insertion	1,71	26,50
5	NN	Nearest Neighbor	0,28	24,79
6	CHCI	Convex Hull Cheapest Insertion	0,83	20,73
7	Greedy	Greedy Heuristic	0,20	16,42
8	FI	Farthest Insertion	2,59	13,35
9	Savings	Clarke-Wright “Savings” Heuristic	0,24	12,03
10	CCA	CCA Heuristic	1129	11,73
11	AppChristo	Approximate Christofides	0,44	11,05
12	Christo-S	Christofides Algorithm	1,04	9,81
13	GENI-10	Gendreau-Hertz-Laporte GENI Heuristic	823,00	5,89
14	2-opt-JM	2-opt	1,41	4,70
15	3-opt-JM	3-opt	1,50	2,88
16	LK-JM	Lin-Kernighan Heuristic	2,06	2,00
17	Tabu-SC-SC	Tabu Search Heuristic	18830,00	1,48
18	MLLK-1N	Multilevel Lin-Kernighan	12,75	1,18
19	CLK-ABCC-N	Chained Lin-Kernighan	63,91	0,90
20	Helsgaun-.1N	Helsgaun Heuristic	1840,00	0,69

Sumber :

David S. Johnson , Lyle A. McGeoch , “The Traveling Salesman Problem and its Variations : Ch1. Experimental Analysis of Heuristics For The STSP” , Kluwer Academic Publishers, 2002

IV. *ITERATIVE IMPROVEMENT HEURISTIC FOR FLIGHT PATH OPTIMIZATION*

Dalam makalah ini, penulis akan membahas sebuah teknik heuristik TSP yang bernama *“Iterative Improvement Heuristic for Flight Path Optimization”*. Teknik heuristik ini adalah pengembangan dari teknik heuristik *tour improvement* : k-opt berdasarkan penelitian terhadap pergerakan lebah saat menyusun rute efektif untuk mengunjungi bunga dan kembali ke sarang.[1]

Teknik heuristik ini terinspirasi dari perilaku hewan saat melakukan perjalanan ke lokasi-lokasi yang penting (seperti migrasi, mencari makanan, dan lain-lain). Berdasarkan beberapa penelitian , hewan cenderung membentuk rute

tertentu saat melakukan perjalanan. Pengamatan ini sudah dilakukan pada beberapa spesies serangga dan burung. Beberapa penelitian terhadap perilaku hewan ini juga sudah menginspirasi pengembangan algoritma & teknik heuristik.

Sebagai contoh, berdasarkan penelitian , lebah diketahui menggunakan tarian untuk mengirimkan sinyal arah dan jarak dari sarang lebah ke sumber makanan kepada lebah lainnya. [8] Berdasarkan pengetahuan ini, dikembangkanlah Algoritma *Artificial Bee Colony* [9] untuk menyelesaikan persoalan optimisasi numerik. Lalu, algoritma ini juga digunakan untuk menyelesaikan persoalan *Travelling Salesman Problem* [10]

Berdasarkan penelitian lain , lebah cenderung membentuk rute untuk mengunjungi beberapa tempat (buah-buahan / nektar) yang meminimalkan jarak tempuh keseluruhan.[1] Pembentukan rute paling efisien untuk mengunjungi setiap bunga tepat sekali dan kembali ke sarang merupakan persoalan pedagang keliling (TSP). Penelitian ini berbeda dengan penelitian yang telah dibahas sebelumnya, karena dalam penelitian ini lebah membuat rute secara individual , tidak dalam sebuah koloni lebah.

Penelitian ini dilakukan dengan cara melacak pergerakan lebah menggunakan radar. Beberapa buah bunga buatan diletakkan pada ruang terbuka , lalu lebah dilepas ke daerah tersebut. Pergerakan jalur terbang lebah akan dicatat menggunakan kamera dan radar.

Berdasarkan data yang dikumpulkan, diketahui bahwa lebah secara individual berusaha untuk mencari bunga , menentukan posisi bunga tersebut , dan membentuk rute optimal. Diketahui bahwa lebah tidak perlu mengenumerasi seluruh kemungkinan rute untuk menemukan rute terbaik.[1]

Data yang telah dikumpulkan dicocokkan dengan beberapa teknik heuristik pada TSP. Beberapa teknik heuristik yang diuji adalah sebagai berikut :

1. *Nearest Neighbor Heuristic*

   Teknik heuristik ini didefinisikan sebagai berikut, pada setiap tahap , kunjungi simpul terdekat yang belum dikunjungi sebelumnya. Definisi heuristik ini secaraformal adalah sebagai berikut.

   Pada tahap awal kita ambil simpul awal c_0 *. Secara induktif , andaikan* $i < N-1$ *dan* $c_0, c_1, \ldots, c_i$ *adalah rute yang dihasilkan sejauh ini. Kita pilih* c_{i+1} *, yaitu simpul yang jaraknya paling dekat dari* c_i *dan belum terpilih sebelumnya. Jika* $i = N - 1$ *,* kita tambahkan sisi $\{c_{N-1}, c_o\}$ *untuk menyelesaikan tur itu. [7]*

   Kompleksitas heuristik ini adalah $O(n^2)$, namun pada kasus tertentu dapat dioptimasi lagi menjadi O(n log n) menggunakan *K-d tree* [7] Namun, heuristik ini belum dapat menjelaskan data yang dikumpulkan [1]

2. *Discovery Order Heuristic*

   Pada teknik heuristik ini, lebah diasumsikan menyusun rute tur berdasarkan urutan penemuan bunga. Namun, heuristik ini juga gagal menjelaskan data hasil eksperimen.

3. *Random "k-opt" Iterative Improvement Heuristic*

   Teknik heuristik ini tergolong dalam *Tour Improvement Heuristic.* Teknik ini tidak menghasilkan sebuah tur , namun metmperbaiki (*improve)* kualitas tur secara iteratif. Pada setiap iterasi, dilakukan optimasi. Jika total bobot sisi pada suatu tahap iterasi lebih kecil daripada total bobot sebelum dilakukan tahap iterasi , maka total bobot sisi yang baru dipertahankan. Berikut adalah skema umum Random k-opt Iterative Improvement

   a. Buat solusi tur awal dari heuristik tertentu

   b. Pilih k buah sisi secara acak, atur ulang susunan sisi tersebut. Jika tur hasil pengaturan ulang silsi memiliki total bobot lebih kecil daripada sebelumnya, keadaan tersebut dipertahankan. Jika sebaliknya, keadaan tur dikembalikan ke posisi awal.

   c. Iterasi langkah (b) hingga ditemukan solusi optimal

   Asumsi yang dibuat dalam memodelkan rute pergerakan lebah dengan *Random "k-opt" Iterative Improvement Heuristic* adalah sebagai berikut [1] :

   a. Lebah mencoba untuk memperbaiki kualitas tur dengan cara mengacak k buah susunan urutan bunga yang dikunjungi.

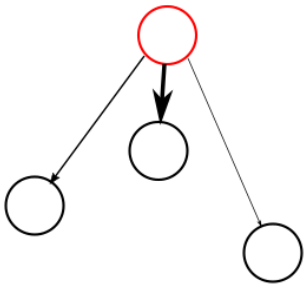
   b. Jika perubahan rute menghasilkan rute yang lebih optimal dari rute sebelumnya. Perubahan tersebut dipertahankan

   Berdasarkan uji coba, teknik heuristik ini tidak dapat menghasilkan rute yang stabil karena bersifat optimisasi acak (*random optimization).*

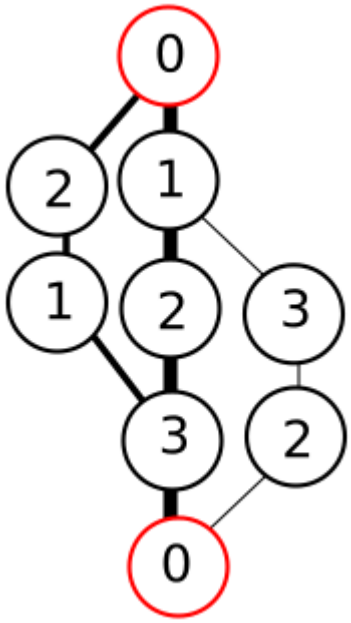
Karena beberapa heuristik sebelumnya belum mampu menjelaskan data eksperimen yang didapat, dikembangkanlah heuristik baru yaitu *Iterative Improvement Heuristic for Flight Path Optimization* [1]. Heuristik ini dibuat berdasarkan analisis terhadap pergerakan lebah saat menyusun rute optimal. Diasumsikan bahwa :

1. Lebah mampu membedakan setiap bunga secara unik.
2. Bobot pada sisi (selanjutnya sisi disebut dengan vektor transisi) mengandung nilai probabilitas suatu sisi itu dipilih oleh lebah dalam tur.
3. Nilai probabilitas ini diinisalisasi berdasarkan jarak tempuh dari simpul awal ke simpul tujuan. Semakin dekat jarak antar simpul, semakin besar probabilitas sisi tersebut dikunjungi oleh lebah
4. Pada setiap tahap iterasi, lebah menghitung total jarak antar bunga yang telah ditempuh saat ini dan membandingkannya dengan total jarak antar bunga

terdahulu. Jika total jarak bunga saat ini lebih pendek , maka nilai probabilitas seluruh vektor transisi yang terlibat dalam penyusunan rute tersebut akan dilipatgandakan dengan faktor tertentu.



Gambar 1. Ilustrasi probabilitas vektor transisi pada awal tahap. Besarnya probabilitas vektor transisi diilustrasikan sebagai tebal garis pada sisi. Perhatikan bahwa sisi dengan jarak antar simpul yang terdekat memiliki probabilitas lebih besar untuk dikunjungi jika dibandingkan dengan sisi lainnya yang lebih jauh.



Gambar 1. Ilustrasi rute solusi setelah beberapa kali iterasi . Besarnya probabilitaas vektor transisi direpresentasikan sebagai ketebalan garis. Mesikpun ada kemungkinan untuk memilih rute lain , probabilitasnya kecil karena rute lain memiliki bobot yang lebih tinggi.

V. IMPLEMENTASI *ITERATIVE IMPROVEMENT HEURISTIC FOR FLIGHT PATH OPTIMIZATION*

Teknik heuristik ini telah diimplementasikan ke dalam bahasa Java. Berikut adalah pseudocode dari heuristik

1. Inisialisasi nilai probabilitas dari tiap vektor transisi. Besarnya nilai probabilitas mula-mula dalam implementasi ini adalah 1/bobot sisi. Lalu nilai probabilitas ini dinormalisasikan agar total probabilitas sisi pada suatu simpul sama dengan satu.
2. Proses pembentukan rute. Memilih sebuah simpul yang belum terpilih berdasarkan probabilitas vektor transisi yang sudah diinisialisasi sebelumnya.
3. Ulangi langkah (2) hingga seluruh simpul terrpilih.
4. Hitung total bobot pada rute.
5. Proses perbaikan rute. Ulangi proses pembentukan rute baru. Lalu bandingkan total bobot rute terbaru dengan total bobot rute sebelumnya. Jika rute baru lebih kecil daripada total bobot rute sebelumnya , seluruh nilai probabilitas vektor transisi yang menyusun rute tersebut dikalikan dengan dua. Setelah itu, lakukan normalisasi ulang terhadap nilai probabilitas
6. Lakukan iterasi proses perbaikan rute

Berikut adalah hasil pengujian terhadap beberapa *test case* yang penulis siapkan.

A. *Test Case #1 : 4 kota*

TABLE II. ANALISIS SOLUSI HEURISTIK DENGAN 10 ITERASI (TESTCASE 4 KOTA)

Iterasi	Percobaan											
	1	2	3	4	5	6	7	8	9	10	11	12
1	41	32	45	41	45	32	32	32	32	32	41	45
2	32	32	45	41	45	32	32	32	32	32	32	32
3	32	32	41	32	41	32	32	32	32	32	32	32
4	32	32	41	32	41	32	32	32	32	32	32	32
5	32	32	41	32	41	32	32	32	32	32	32	32
6	32	32	41	32	41	32	32	32	32	32	32	32
7	32	32	41	32	41	32	32	32	32	32	32	32
8	32	32	41	32	41	32	32	32	32	32	32	32
9	32	32	41	32	41	32	32	32	32	32	32	32
10	32	32	41	32	41	32	32	32	32	32	32	32

TABLE III. ANALISIS SOLUSI HEURISTIK DENGAN 100 ITERASI (*TESTCASE* 4 KOTA)

Iterasi	Percobaan											
	1	2	3	4	5	6	7	8	9	10	11	12
10	32	32	32	32	41	32	32	32	32	32	32	32
20	32	32	32	32	41	32	32	32	32	32	32	32
30	32	32	32	32	41	32	32	32	32	32	32	32
40	32	32	32	32	41	32	32	32	32	32	32	32
50	32	32	32	32	41	32	32	32	32	32	32	32
60	32	32	32	32	41	32	32	32	32	32	32	32
70	32	32	32	32	41	32	32	32	32	32	32	32
80	32	32	32	32	41	32	32	32	32	32	32	32
90	32	32	32	32	41	32	32	32	32	32	32	32
100	32	32	32	32	41	32	32	32	32	32	32	32

TABLE IV. ANALISIS SOLUSI HEURISTIK DENGAN 1000 ITERASI (*TESTCASE* 4 KOTA)

Iterasi	Percobaan											
	1	2	3	4	5	6	7	8	9	10	11	12
100	32	41	32	32	32	32	32	32	32	41	32	32
200	32	41	32	32	32	32	32	32	32	41	32	32
300	32	41	32	32	32	32	32	32	32	41	32	32
400	32	41	32	32	32	32	32	32	32	41	32	32
500	32	41	32	32	32	32	32	32	32	41	32	32
600	32	41	32	32	32	32	32	32	32	41	32	32
700	32	41	32	32	32	32	32	32	32	41	32	32
800	32	41	32	32	32	32	32	32	32	41	32	32
900	32	41	32	32	32	32	32	32	32	41	32	32
1000	32	41	32	32	32	32	32	32	32	41	32	32

Berdasarkan data hasil pengujian , heuristik ini cukup baik untuk menyelesaikan persoalan TSP dengan instans yang kecil. Sel tabel yang berwarna hitam menunjukkan solusi TSP yang belum optimal. Terlihat pula bahwa pada tahap-tahap iterasi tertentu , heuristik ini berhasil meningkatkan kualitas rute.

Secara umum, semakin banyak iterasi yang dilakukan , semakin dekat solusi yang dihasilkan ke solusi optimal. Namun, karena teknik heuristik ini bergantung pada peubah acak , pernyataan ini tidak sepenuhnya benar. Sebagai contoh, berdasarkan percobaan di atas , penggunaan heuristik dengan 1000 iterasi menghasilkan solusi yang lebih buruk daripada heuristik 100 iterasi. Kita tidak bisa menjamin dengan pasti bahwa peningkatan jumlah iterasi akan meningkatkan kualitas rute.

B. Test Case #1 : 10 kota

TABLE V. ANALISIS SOLUSI HEURISTIK DENGAN 10 ITERASI (*TESTCASE* 4 KOTA)

Iterasi	Percobaan											
	1	2	3	4	5	6	7	8	9	10	11	12
1	298	314	322	338	338	298	314	346	298	330	354	258
2	298	298	322	306	290	298	282	330	274	290	298	258
3	298	298	298	282	290	298	274	314	258	290	298	258
4	298	298	298	282	290	290	274	314	258	274	298	258
5	298	298	298	282	290	290	274	314	258	274	298	258
6	298	298	298	282	290	290	274	298	258	274	298	258
7	274	298	298	282	290	290	274	298	258	274	298	258
8	274	298	298	282	290	290	274	298	258	274	282	258
9	274	298	298	282	282	290	274	298	258	274	282	258
10	274	290	298	282	282	290	274	298	258	274	282	258

TABLE VI. ANALISIS SOLUSI HEURISTIK DENGAN 100 ITERASI (*TESTCASE* 10 KOTA)

Iterasi	Percobaan											
	1	2	3	4	5	6	7	8	9	10	11	12
10	250	290	274	250	290	274	282	290	282	274	298	290
20	250	290	274	250	274	274	266	290	282	258	298	290
30	250	290	274	250	274	274	266	290	282	258	298	290
40	250	290	274	250	274	266	266	290	282	258	290	290
50	250	290	274	250	274	266	266	290	282	258	290	290
60	250	290	274	250	274	266	266	290	282	258	290	290
70	250	290	274	250	274	266	266	290	282	258	290	290
80	250	274	274	250	266	266	258	290	282	258	290	290
90	250	274	274	250	266	266	258	290	282	258	282	282
100	250	274	274	250	266	266	258	290	282	258	274	282

TABLE VII. ANALISIS SOLUSI HEURISTIK DENGAN 1000 ITERASI (*TESTCASE* 10 KOTA)

Iterasi	Percobaan											
	1	2	3	4	5	6	7	8	9	10	11	12
100	274	266	250	274	282	282	266	250	282	282	266	298
200	274	258	250	274	282	274	258	250	282	282	266	290
300	274	258	250	274	282	274	258	250	282	282	266	290
400	274	258	250	274	282	274	258	250	282	282	266	290
500	274	258	250	274	282	274	258	250	282	282	266	290
600	266	258	250	274	282	274	258	250	282	282	266	290
700	266	258	250	274	282	274	258	250	282	282	266	290
800	266	258	250	274	282	274	258	250	282	282	266	290
900	266	250	250	274	282	274	258	250	282	282	266	290
1000	266	250	250	274	282	274	258	250	282	282	266	290

Berdasarkan hasil percobaan di atas, dapat disimpulkan bahwa teknik heuristik ini tidak cukup baik untuk menyelesaikan permasalahan TSP dengan instans yang besar. Hal ini disebabkan karena heuristik ini dibuat berdasarkan data eksperimen TSP dengan instans yang kecil (Dalam penelitian yang dilakukan Lihoreau,dkk jumlah bunga yang dikunjungi lebah hanya 5 buah [1]). Oleh karena itu, teknik heuristik ini cukup baik untuk memodelkan pergerakan lebah saat eksperimen tersebut dilakukan (lebah mengunjungi 5 buat bunga) , tetapi tidak cukup baik untuk memodelkan pergerakan lebah dengan instans TSP lebih besar.

Dapat disimpulkan pula bahwa semakin banyak jumlah iterasi yang dilakukan oleh heuristik , secara umum semakin baik solusi yang dihasilkan (meskipun tidak menjamin seutuhnya, seperti yang sudah dijelaskan pada bagian sebelumnya). Terlihat bahwa pada percobaan 10 iterasi , heuristik ini tidak mampu menghasilkan solusi optimal. Solusi optimal baru terlihat pada percobaan 100 iterasi dan 1000 iterasi.

VI. KESIMPULAN

Penulis telah berhasil mengimplementasikan teknik heuristik *Iterative Improvement Heuristic for Flight Path Optimization* dalam bahasa Java. Teknik heuristik ini ditemukan oleh Lihoreau,dkk [1] .Heuristik ini merupakan pengembangan dari *Iterative Improvement Heuristic* berdasarkan analisis pola pergerakan lebah. saat menyusun rute efektif dari sarang ke bunga.

Berdasarkan pengujian terhadap teknik heuristik , heuristik ini cukup baik untuk menyelesaikan persoalan TSP dengan instans yang kecil. Penulis berpendapat bahwa hal ini disebabkan oleh pengembangan heuristik yang berdasarkan pada data eksperimen yang dilakukan oleh Lihoreau,dkk. Pada eksperimen yang dilakukan , jumlah bunga yang dikunjungi oleh lebah hanya lima buah.Untuk menyelesaikan persoalan TSP dengan instans yang lebih besar, dibutuhkan banyak iterasi untuk mencapai solusi optimal.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Allah SWT , karena atas rahmat-Nya, penulis dapat menyelesaikan makalah ini. Selain itu, penulis juga mengucapkan terima kasih kepada Bapak Rinaldi Munir dan Ibu Nur Ulfa Maulidevi selaku dosen pengampu kuliah IF221 Strategi Algoritma yang telah memberikan ilmu yang bermanfaat kepada penulis

REFERENSI

[1] Mathieu Lihoreau, Nigel E. Raine , et al , “Radar Tracking and Motion Sensitive Cameras on Flowers Reveal the Development of Pollinator Multi-Destination Routes over Large Spatial Scales” , PLOS Biology , vol 10 , Issue 9 , September 2012

[2] Dr.Ir.Rinaldi Munir,M.T. , Diktat Kuliah IF2111 : Strategi Algoritma , Bandung : Program Studi Teknik Informatika, Insitut Teknologi Bandung , 2009 , pp.157

[3] Applegate DL, Bixby RE, Chavatal V , Cook WJ . The traveling salesman problem : a computational study. Princeton, NJ: Princeton University Press, pp 1

[4] Christian Nilsson , “Heuristics for the Traveling Salesman Problem” , Linkoping University

[5] Fereidouni , “Solving Traveling Salesman Problem By Using A Fuzzy Multi-Objective Linear Programming”, African Journal of Mathematics and Computer Science Research. ,Vol: 4(11),

[6] Natallia Kokash , “An Introduction to Heuristic Algorithms” , Department of Informatics and Telecommunications , University of Trento

[7] David S. Johnson , Lyle A. McGeoch , “The Traveling Salesman Problem and its Variations : Ch1. Experimental Analysis of Heuristics For The STSP” , Kluwer Academic Publishers, 2002

[8] Riley JR, Greggers U, Smith AD, Reynolds DR, Menzel R , “The fight paths of honeybees recruited by the waggle dance”, Nature 435 , May 2005

[9] D.Karaboga, “An Idea Based On Honey Bee Swarm For Numerical Optimization” , Erciyes University , 2005

[10] Faisal Amri, Erna Budhiati Nababan, Mohamad Fadly Syahputra , “*Artificial Bee Colony Algorithm* untuk menyelesaikan *Travelling Salesman Problem”* , Universitas Sumatera Utara :Fakultas Ilmu Komputer dan Teknologi Informasi , Jurnal Dunia Teknologi Informasi Vol.1 (2012) pp.8-13

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran , atau terjemahan dari makalah orang lain, dan bukan plagiasi

Bandung, 9 Mei 2016

Adi Purnama (13514006)