PERBANDINGAN ALGORITMA KRUSKAL DENGAN ALGORITMA GENETIKA DALAM PENYELESAIAN MASALAH *MINIMUM SPANNING TREE* (MST)

Rusdi Efendi[1], Diyah Puspitaningrum[2], Riska[3],

Program Studi Teknik Informatika, Fakultas Teknik,Universitas Bengkulu.
Jl. WR. Supratman Kandang Limun Bengkulu 38371A INDONESIA
(telp: 0736-341022; fax: 0736-341022)

[1]r_efendi@yahoo.com,[2]diyahpuspitaningrum@yahoo.com, [3]Riska_1611@yahoo.co.id,

ABSTRACT

This research aims to develop a problem solving system of Minimum Spanning Tree using Kruskal Algorithm and Genetic Algorithm. The problem of Minimum Spanning Tree is how to calculate minimum distance in a complete graph where each nodes are connected and the selected edge should not make sircuit. This application system is built using Visual Basic 6.0 programming and MySQL database. The result of the whole process of this Minimum Spanning Tree aplication system is minimum distance which is calculated using Kruskal Algorithm and Gentic Algorithm. The displayed result are both text and visualization graph which show the minimum tree of a graph. Kruskal Algorithm generally shows better result than Genetic Algorithm with minimum distance resulted and running time as the parameters. For 5-25 nodes, Kruskal Algorithm generate minimum distance up to 50% and running time process 50 times faster than Genetic Algorithm.

Keyword: Minimum Spanning Tree, MST, Kruskal, Genetic

PENDAHULUAN

Salah satu penggunaan teknologi komputer yeng berkembang saat ini adalah untuk pengimplementasian teori graf sebagai alat bantu pengambilan keputusan, pencarian rute terpendek dan kecepatan waktu tempuh atau perhitungan biaya minimum untuk menghasilkan keuntungan terbesar.

Teori Graf adalah salah satu cabang ilmu matematika yang sudah ada sejak lama dan memiliki segi terapan di banyak bidang ilmu pengetahuan dan teknologi informasi. Dalam Teori Graf [1], 2006), graf linier didefinisikan sebagai graf G(*V,E)* yang terdiri dari himpunan obyek V = {v_1, v_2, ...} yang disebut *verteks* (simpul) dan himpunan garis E = {e_1, e_2, ...} yang disebut *edge* Menurut [1] menyatakan graf *G* secara matematis sebagai pasangan himpunan (*V,E*) dimana :

V = himpunan tidak kosong dari simpul – simpul

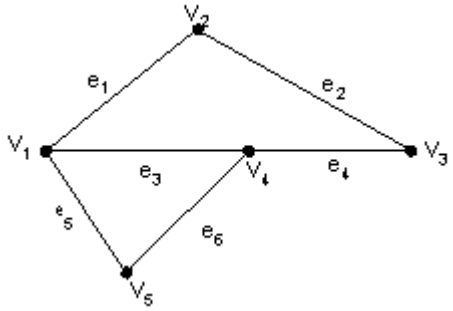
v_i : $\{v_1, v_2, \ldots, v_n\}$ .....(1)

E = himpunan busur yang menghubungkan sepasang simpul

v_j : $\{e_1, e_2, \ldots, e_n\}$ ……(2)

Busur $e_i = (v_i, v_j)$ adalah pasangan simpul dengan $v_i, v_j \subset V$ .....(3)

dan nilai *i,j* = 1,2,3,…n



Gambar 1 Graf G (V,E)

Sebagai contoh, graf G dalam Gambar 1 mempunyai titik-titik v1; v2; v3; v4; v5, sedangkan sisi-sisinya dinyatakan oleh e1 = {v1; v2}, e2 = {v2; v3}, e3 = {v1; v4}, e4 = {v4; v3}, e5 = {v1;v5}, e6 = {v5; v4}.

Graf memiliki banyak konsep, salah satu diantaranya adalah konsep pohon *(tree)*. Pohon (*tree*) didefinisikan sebagai graf terhubung yang tidak mengandung sirkuit. Karena merupakan graf terhubung, maka pohon selalu terdapat jalur *(path)* yang menghubungkan setiap dua simpul dalam pohon [2]. Pemilihan konsep pohon sebagai salah satu konsep terapan graf disebabkan karena konsep pohon merupakan konsep yang sangat dekat dengan kehidupan nyata. Beberapa contoh topik yang dikembangkan dan diselesaikan dalam teori graf bentuk pohon adalah seperti kasus Pohon Keputusan (*Decision Tree*), *Travelling Salesman Problem* (TSP), Penjadwalan dan juga *Minimum Spanning Tree* (MST).

Minimum spanning tree (MST) merupakan sebuah permasalahan dalam suatu graf yang mana banyak aplikasinya baik secara langsung maupun tidak langsung telah dipelajari. Masalah *Minimum spanning tree* hampir serupa dengan masalah rute terpendek (*shortest route*), kecuali bahwa tujuannya adalah untuk menghubungkan seluruh simpul dalam jaringan sehingga total panjang cabang tersebut diminimisasi.

Permasalahan *Minimum spanning tree* sederhana mungkin bisa diselesaikan dengan melakukan perhitungan manual. Namun untuk kasus *spanning tree* yang besar dan kompleks, perhitungan manual akan sulit dilakukan karena akan memakan waktu yang lama. Oleh sebab itu dibutuhkan satu program aplikasi komputer yang dapat melakukan perhitungan nilai minumum spanning tree dengan cepat dan akurat.

Algoritma *Greedy* merupakan metode paling populer untuk memecahkan persoalan optimasi. Algoritma *greedy* adalah algoritma yang memecahkan masalah langkah perlangkah dimana pada setiap langkah dibuat pilihan optimum (*local optimum*) dengan harapan bahwa langkah berikutnya mengarah ke solusi optimum global (*global optimum*). Algoritma *Greedy* yang umum digunakan untuk permasalahan *minimum spanning tree* adalah Algoritma Kruskal.

Algoritma Kruskal pertama kali muncul pada tahun 1956 dalam sebuah tulisan yang ditulis oleh Joseph Kruskal. Algoritma kruskal adalah sebuah algoritma dalam teori graf yang mencari sebuah *minimum spanning tree* untuk sebuah graf berbobot yang terhubung.

Pada Algoritma Kruskal, sisi (edge) dari graf diurut terlebih dahulu berdasarkan bobotnya dari kecil ke besar. Sisi yang dimasukkan ke dalam himpunan T adalah sisi graph G yang sedemikian sehingga T adalah Tree (pohon). Sisi dari Graph G ditambahkan ke T jika ia tidak membentuk cycle.

Namun penelitian terkait [3] tersebut hanya berpusat pada satu algoritma itu saja tanpa membandingkannya dengan algoritma lain. Untuk itu pada penelitian ini peneliti mencoba melakukan suatu komparasi (perbandingan) dengan algoritma lain yaitu Algoritma Genetika.

Algoritma Genetika adalah algoritma pencarian heuristik yang didasarkan atas mekanisme evolusi biologis. Keberagaman pada evolusi biologis adalah variasi dari kromosom antar individu organisme. Variasi kromosom ini akan mempengaruhi laju reproduksi dan tingkat kemampuan organisme untuk tetap hidup [4]. Algoritma Genetika menggunakan prinsip pencarian solusi neighborhood berdasarkan mekanisme seleksi alam (*natural selection*) dan genetika alam (*natural genetics*) yang dapat digunakan untuk memecahkan masalah optimasi seperti permasalahan *Minimum Spanning Tree*.

Pada Algoritma Genetika, penyelesaian permasalahan *Minimum Spanning Tree* hampir sama dengan penyelesaian pada masalah optimasi lainnya. Perbedaannya hanyalah pada proses pengkodean kromosom (*encoding),* perhitungan bobot (*decoding*) dan rekombinasi atau persilangan (*crossover*). Langkah-langkah Algoritma Genetika pada *Minimum Spanning Tree* adalah sebagai berikut [5]:

1. Pengkodean Kromosom (*Encoding*)

Pada *Minimum Spanning Tree* (MST), kromosom merupakan representasi dari pohon rentangan (*Spanning Tree*). Pengkodean kromosom pada MST menggunakan teknik *decoder* kromosom bernama *Prufer Number.*

2. Perhitungan Bobot (*Decoding*)

Proses *decoding* berguna untuk mengkodekan nilai-nilai gen-gen pembentuk individu atau mengubah setiap kromosom menjadi sebuah pohon, sehingga jumlah bobot dari setiap pohon dapat dihitung.

3. Rekombinasi atau Persilangan (*Crossover*)

Crossover (penyilangan) dilakukan atas 2 kromosom untuk menghasilkan kromosom anak (*offspring*). Metode *crossover* yang paling sering digunakan pada algoritma genetika adalah metode *crossover* satu titik (*one-point crossover*).

Berdasarkan hal diatas, pada penelitian ini peneliti akan membuat suatu perangkat lunak yang dapat melakukan perbandingan terhadap Algoritma Kruskal dan Aloritma Genetika dari sudut pandang hasil jarak minimum dan waktu komputasi dalam permasalahan *Minimum Spanning Tree*.

METODA PENELITIAN

Pada penelitian ini akan membandingkan dan menganalisa performansi dari dua algoritma yaitu Algoritma Kruskal dan Algoritma Genetika dalam menyelesaikan masalah minimum spanning tree berdasarkan sudut pandang hasil jarak minimum dan dan waktu komputasi.

Pada pengujian hasil perhitungan algoritma yang di hasilkan pada penelitian, diuji dengan metode Uji Perancangan percobaan. Perancangan percobaan adalah suatu uji atau sederetan uji baik menggunakan statistika deskripsi maupun statistik inferensi yang bertujuan untuk mengubah peubah input menjadi suatu output yang merupakan respon dari percobaan tersebut atau Perancangan percobaan adalah prosedur untuk menempatkan perlakuan ke dalam satuan-satuan percobaan dengan tujuan utama mendapatkan data yang memenuhi persyaratan ilmiah [6]. Tujuan dasar suatu rancangan percobaan adalah untuk membandingkan efek-efek dari berbagai tingkatan (kondisi) suatu percobaan.

Secara umum Tujuan penelitian ini adalah :

1. Membangun satu aplikasi M*inimum Spanning Tree* menggunakan Algoritma Kruskal dan Algoritma Genetika.
2. Membandingkan hasil nilai minimum dan waktu proses dari Algoritma Kruskal dengan Algoritma Genetika dalam penyelesaian permasalahan *Minimum Spanning Tree*
3. Menyimpulkan algoritma manakah yang lebih baik diantara Algoritma Kruskal dan Algoritma Genetika dalam menentukan nilai minimum dari *Spanning Tree*.

Dalam peneltian ini dirumuskan permasalah yang akan dibahasa sebagai berikut:

1. Bagaimana membangun satu aplikasi M*inimum Spanning Tree* menggunakan Algoritma Kruskal dan Algoritma Genetika.
2. Bagaimana hasil perhitungan Algoritma Kruskal dan Algoritma Genetika pada permasalahan *Minimum Spanning Tree*
3. Algoritma manakah yang lebih efektif diantara Algoritma Kruskal dan Algoritma Genetika dalam menentukan M*inimum Spanning Tree*.

Untuk lebih memokuskan pengerjaan penelitian ditetapkan pembatasan-pembatasan sebagai berikut:

1. Konsep Graf yang disajikan adalah graf lengkap dimana semua titik
1. tiap graf terhubung (*Connected Graph*), graf tidak berarah
2. (*Undirected Graph*) dan graph berbobot (*Weighted Graph*)
3. Nilai koordinat yang di inputkan adalah bernilai bulat positif
4. Jumlah simpul yang di inputkan lebih besar atau sama dengan 3 (simpul ≥ 3)

HASIL DAN PEMBAHASAN

Sistem yang dirancang pada penelitian ini adalah aplikasi perbandingan algoritma kruskal dan algoritma genetika. Jenis Penelitian yang digunakan dalam penelitian ini adalah penelitian terapan. Teknik pengumpulan data yang digunakan pada penelitian ini adalah Metode Studi Pustaka

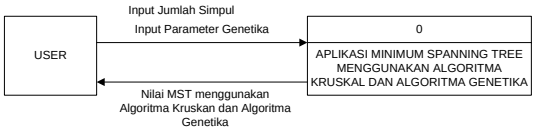
Metode pengembangan sistem yang digunakan dalam penelitian ini adalah metode Sekuensial Linier atau biasa

disebut model Waterfall versi Zhao [7]. Metode pengujian yang dilakukan dibedakan pada dua teknik pengujian, yaitu pengujian perangkat lunak dan pengujian hasil. Pada pengujian perangkat lunak menggunakan teknik pendekatan *black box testing* (ujicoba *blackbox*). Sedangkan untuk hasil perhitungan yang dihasilkan, digunakan teknik Rancangan Percobaan.

Dalam perancangan sistem, pada penelitian ini meggunakan Perancangan *Data Flow Diagram* (DFD).

a. Diagram Konteks

Diagram Konteks menggambarkan sistem secara umum atau keseluruhan, sehingga diagram konteks dari sistem yang akan dibangun dapat dilihat pada Gambar 3 berikut ini.

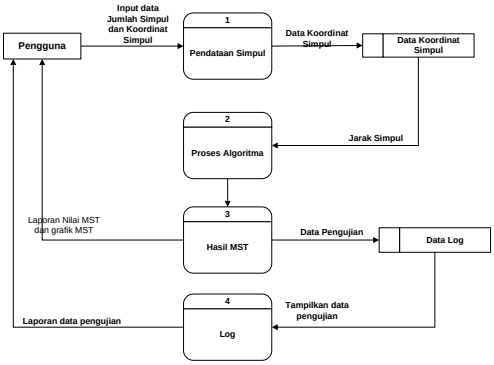


Gambar 3. Diagram Konteks

Pada Gambar 3 terlihat bahwa hanya terdapat *user* yang dapat menggunakan sistem. *User* memberikan sejumlah data yang dibutuhkan oleh sistem seperti koordinat simpul dan parameter-parameter genetika. Sedangkan hasil yang diberikan oleh sistem kepada *user* yaitu nilai MST menggunakan Algoritma Kruskal dan Algoritma Genetika.

b. *Data Flow Diagram* Level 0

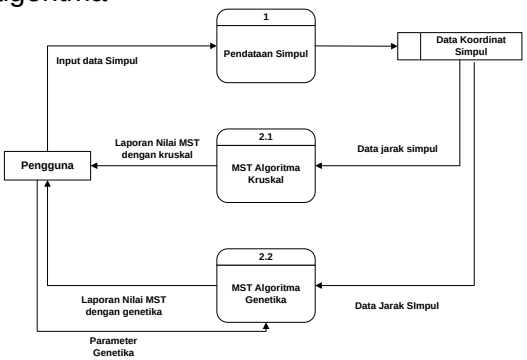
Diagram konteks kemudian dapat dikembangkan secara lebih terinci menjadi DFD Level-0. Proses-proses pada Level-0 merupakan dekomposisi dari proses utama Sistem Aplikasi *Minimun Spanning Tree* Menggunakan Algoritma Kruskal Dan Algoritma Genetika. DFD Level-0 dapat dilihat pada Gambar 4.



Gambar 4. DFD Level-0

c. *Data Flow Diagram* Level 1

Data flow diagram level- 1 merupakan perincian dari DFD level- 0 proses 2 dan menjelaskan aliran dalam tahapan proses algoritma



Gambar 5. DFD Level- 1

Dari Gambar 5, pada DFD level- 1 terdapat 2 tahapan proses yaitu *Minimum Spanning Tree menggunakan* Algoritma Kruskal dan *Minimum Spanning Tree* menggunakan Algoritma Genetika. Data dimulai setelah pengguna meng-*input*-kan koordinat simpul dan disimpan pada data koordinat simpul. Data pada koordinat simpul kemudian dihitung nilai jarak antar simpul dan diproses menggunakan Algoritma Kruskal dan Algoritma Genetika untuk menghasilkan nilai MST.

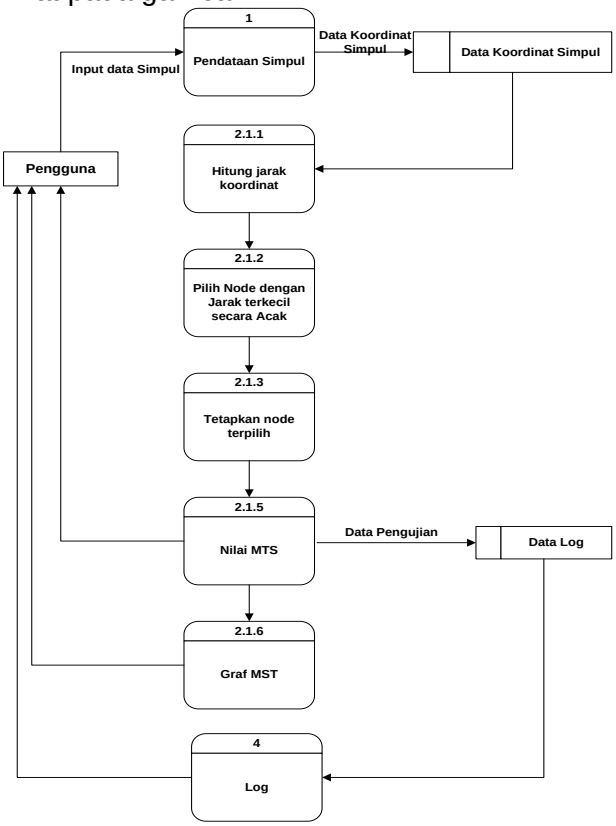
d. *Data Flow Diagram* Level 2 Proses 1

Data flow Diagram Level- 2 proses 1 merupakan perincian dari proses MST menggunakan Algoritma Kruskal. DFD dari proses MST Algoritma Kruskal dilihat pada gambar 6.

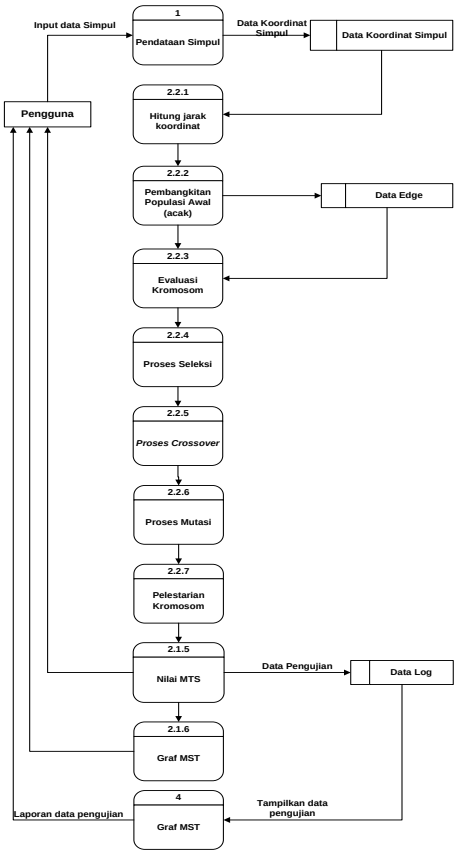
e. *Data Flow Diagram* Level 2 Proses 2

Data flow Diagram Level- 2 proses 2 merupakan perincian dari proses MST

menggunakan Algoritma Genetika. DFD dari proses MST Algoritma Genetika dilihat pada gambar 7



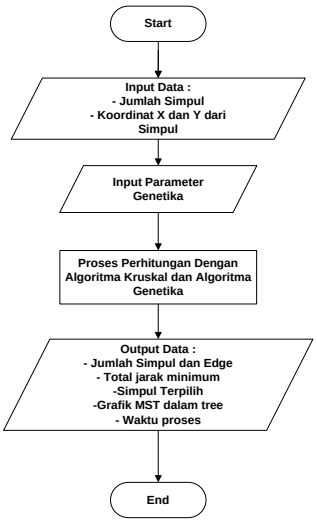
Gambar 6. DFD Level 2 Proses 1



Gambar 7. DFD Level- 2 Proses 2, Minimum SpanningTree dengan Algoritma Genetika

Perancangan *Flowchart* (Diagram Alir)

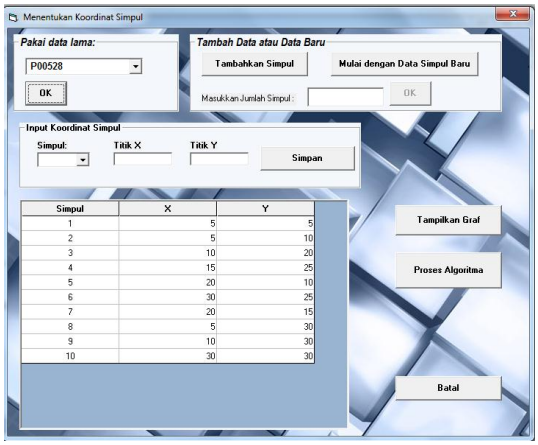
Diagram alir sistem digunakan untuk menggambarkan keseluruhan langkah kerja sistem aplikasi yang akan dibuat. Sistem dimulai dengan pengguna diharuskan mengisi data yang dibutuhkan berupa jumlah simpul dan koordinat simpul Kemudian sistem akan memproses data tersebut dengan Algoritma Kruskal, Sedangkan untuk Algoritma Genetika, pengguna diharuskan memasukkan parameter genetika. Setelah proses selesai, maka pengguna akan menadapatkan hasil nilai/ bobot minimum dari *Spanning Tree*, simpul dan *edge* terpilih, waktu proses dan gambaran *Minimum spanning tree* dalam bentuk grafik. *Flowchart* sistem secara umum dalam penyelesaian *Minimum Spanning Tree* dapat dilihat pada Gambar 8



Gambar 8. Flowchart Sistem Aplikasi Secara Umum

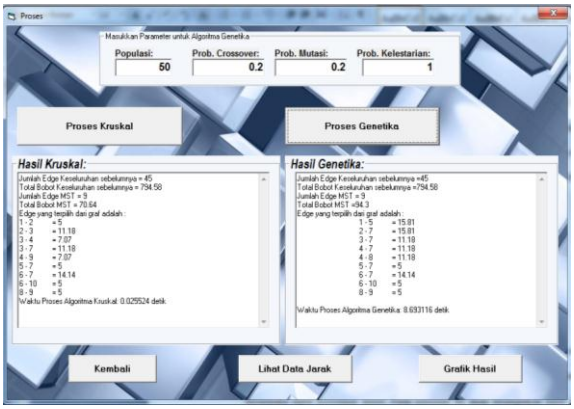
A. Implementasi Sistem

1. Implementasi Antar muka Sistem Implementasi dari sistem yang dibangun dapat dilihat pada gambar dibawah ini.



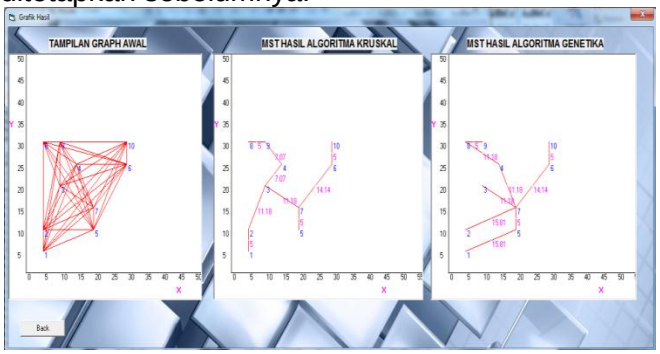
Gambar 9. Tampilan Halaman *Input* Koordinat

Pada Gambar 9 *User* diminta untuk memasukkan data koordinat simpul titik X dan titik Y dari simpul yang nanti nya akan diproses untuk mencari nilai MST nya. Data koordinat yang dimasukkan bisa berupa data koordinat lama yang telah dimasukkan sebelumnya atau menggunakan data yang benar-benar baru. Jika data koordinat simpul telah ditetapkan, maka selanjutnya dapat dihitung nilai MST nya.



Gambar 10. Tampilan Halaman Proses

Gambar 10 adalah halaman proses algoritma, dimana data koordinat simpul yang dimasukkan sebelumnya dapat dihitung nilai MST nya dengan menggunakan Algoritma Kruskal dan Algoritma Genetika. Untuk Algoritma Genetika, terlebih dahulu harus dimasukkan parameter-parameter genetika berdasarkan aturan yang telah ditetapkan sebelumnya.



Gambar 11. Tampilan Halaman Grafik Hasil

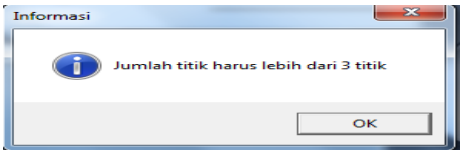
Dari gambar 11 dapat dilihat graf awal dari koordinat simpul yang dimasukkan dihalaman *input* koordinat dan nilai MST yang dihasilkan dalam bentuk grafik *tree.*

B. Pengujian Sistem

1. Pengujian *Blackbox*

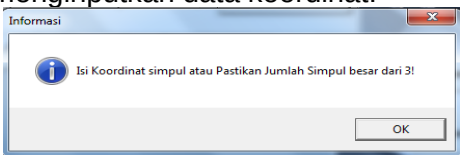
Pengujian yang dilakukan adalah pengujian tidak normal, yaitu pengujian yang dilakukan dengan memberikan masukan dengan spesifikasi yang tidak diijinkan sehingga sistem akan memberikan reaksi lain. Reaksi sistem berupa berupa peringatan *(alert)* atau penanganan kesalahan *(error handling)*.

- Peringatan jika jumlah simpul yang di isikan kecil dari 3.



Gambar 12. Contoh Peringatan Jika data simpul yang diisikan kecil dari 3

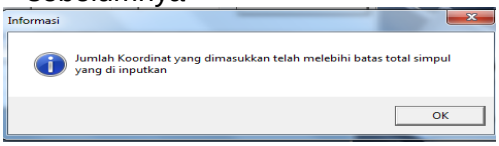
- Peringatan jika pengguna tidak memasukkan data jumlah simpul terlebih dahulu namun akan menginputkan data koordinat.



Gambar 13.Contoh Peringatan Jika data simpul belum dimasukkan

- Peringatan jika data koordinat yang diinputkan lebih besar dari jumlah

simpul yang telah ditentukan sebelumnya



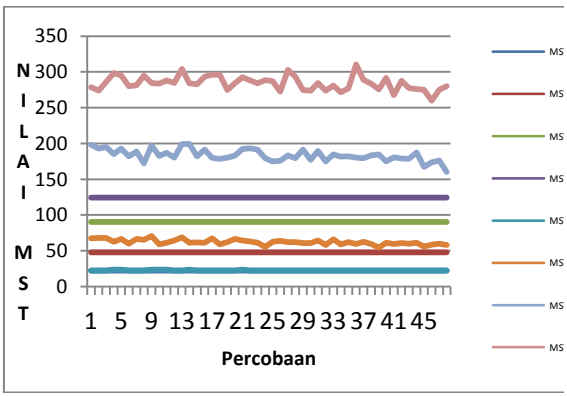
Gambar 14. Contoh Peringatan Jika jumlah data simpul yang akan dimasukkan telah melebihi dari yang ditentukan sebelumnya

2. Pengujian *Sampel Testing.*

Pada pengujian ini dilakukan dengan menggunakan jumlah data simpul yang berbeda, yaitu 5 simpul, 10 simpul, 20 simpul, dan 25 simpul dan parameter pululasi yaitu 2 x Jumlah simpul sampai 10 x jumlah simpul (tidak melebihi dari n^{n-2}) dan probabilitas yang digunakan adalah probalilitas *crossover* (*crossover* < 0.4), mutasi (mutasi < 0), serta seluruh populasi yang di-*generate* sebelumnya diikutkan kembali sebaagai generasi pada iterasi selanjutnya (kelestarian = 1). Dari pengujian yang telah dilakukan, Rata-rata nilai MST yang dihasilkan oleh Algoritma Kruskal dan Algoritma Genetika adalah sebagai berikut :

Tabel 1. Rata-rata Hasil MST Kruskal dan Genetika

Simpul	MST Kruskal	MST Genetika
5	22.27	22.46
10	47.81	62.14
20	90.37	183.36
25	124.39	284.06



Gambar 15. Grafik hasil pengujian MST dengan Algoritma Kruskal dan Algoritma genetika

Dari tabel 1 dan grafik 15, dapat dilihat hasil pengujian MST Algoritma Kruskal dan Algoritma Genetika. Algoritma Kruskal memberikan hasil nilai minimum yang tetap pada tiap percobaan yang dilakukan, hal ini dikarenakan kelemahan Algoritma Kruskal yang terjebak pada solusi lokal optimal, yaitu nilai MST yang dihasilkan akan selalu sama pada tiap percobaan. Sedangkan algoritma Genetika lebih bergerak, hal ini disebabkan pada Algoritma Genetika, hasil MST yang dihasilkan juga dipengaruhi oleh parameter-parameter Algoritma Genetika yang diberikan. Walaupun kelemahan Algoritma Kruskal adalah terjebak pada solusi lokal optimal, namun nilai optimum yang dihasilkan oleh Algoritma Kruskal secara umum lebih baik dari nilai Minimum yang dihasilkan Algoritma Genetika.

Untuk rata-rata waktu proses MST Algoritma Kruskal dan Algoritma Genetika adalah pada tabel 2 :

Tabel 2. Rata-rata waktu proses MST Kruskal dan Genetika

Simpul	Waktu proses MST Kruskal	Waktu Proses MST Genetika
5	0.01373844	2.36437556
10	0.0172689	9.74124621
20	0.03916567	72.2387071
25	0.05063688	103.460496

Pada waktu proses MST dengan Algoritma Kruskal dan Algoritma Genetika, Waktu yang dibutuhkan Algoritma Kruskal juga lebih cepat dari Algoritma Genetika. Hal ini dikarenakan Algoritma Kruskal yang memiliki prinsip pencarian *Greedy* yaitu dengan acak langsung mencari nilai minimum dari graf, sedangkan Algoritma Genetika membutuhkan iterasi-iterasi yang panjang, sehingga membutuhkan waktu yang lebih lama dari Algoritma Kruskal.

C. Analisis Kinerja Sistem

Analisis sistem menggunakan teknik rancangan percobaan, dimana hasil-hasil

pengujian sebelumnya di analisis dengan bantuan aplikasi SPSS 22.

1. Analisis Hasil MST Algoritma Kruskal dan Algoritma Genetika.
   a. Analisis Algoritma Kruskal

Tabel 3. Tabel ANOVA Algoritma Kruskal

ANOVA					
MST					
	Sum of Squares	df	Mean Square	F	Sig.
Between Groups	294619.277	3	98206.426	8.412E+31	.000
Within Groups	.000	188	.000		
Total	294619.277	191			

Analisa:

Dalam Rancangan Acak Lengkap tingkat signifikansi α = 0.05, dan hipotesis yang diberikan untuk analisa efek perlakuan adalah adalah :

H_0 = Perubahan nilai variabel (Simpul) pada percobaan, tidak mempengaruhi nilai MST yang dihasilkan

H_1 = Perubahan nilai variabel (Simpul) pada percobaan, mempengaruhi nilai MST yang dihasilkan

Dasar pengambilan keputusan yaitu tolak Ho jika α > sig.

Berdasarkan data diataas maka disimpulkan oleh karena α = 0.05 > sig.= 0.000 maka H_0 ditolak dan H_1 diterima.

Dengan kata lain nilai MST pada Algoritma Kruskal sangat dipengaruhi oleh banyak nya simpul yang ada.

b. Analisis Algoritma Genetika

Tabel 4. Tabel ANOVA Algoritma Genetika

Tests of Between-Subjects Effects						
Dependent Variable: MST						
Source		Type II Sum of Squares	df	Mean Square	F	Sig.
Intercept	Hypothesis	185383.507	1	185383.507	6792.351	.000
	Error	81.879	3	27.293[a]		
populasi	Hypothesis	196.007	3	65.336	7.858	.007
	Error	74.831	9	8.315[b]		
crossover	Hypothesis	28.782	2	14.391	1.983	.218
	Error	43.551	6	7.259[c]		
mutasi	Hypothesis	81.879	3	27.293	4.428	.211
	Error	10.951	1.777	6.164[d]		
populasi * crossover	Hypothesis	21.448	6	3.575	.380	.882
	Error	169.373	18	9.410[e]		
populasi * mutasi	Hypothesis	74.831	9	8.315	.884	.557
	Error	169.373	18	9.410[e]		
crossover * mutasi	Hypothesis	43.551	6	7.259	.771	.602
	Error	169.373	18	9.410[e]		
populasi * crossover * mutasi	Hypothesis	169.373	18	9.410	.	.
	Error	.000	0	.[f]		

Analisa :

Dari analisis Rancangan Percobaan, nilai tingkat signifikansi ditetapkan, α = 0.05. dan variabel pengubah adalah Jumlah Populasi, *Crossover,* dan Mutasi.

Hipotesis :

H_0 = Perubahan variabel Populasi, *Crossover,* dan Mutasi tidak berpengaruh pada nilai MST yang dihasilkan

H_1 = Perubahan variabel Populasi, *Crossover,* dan Mutasi berpengaruh pada nilai MST yang dihasilkan

Dasar pengambilan keputusan adalah : Tolak Ho jika a > sig

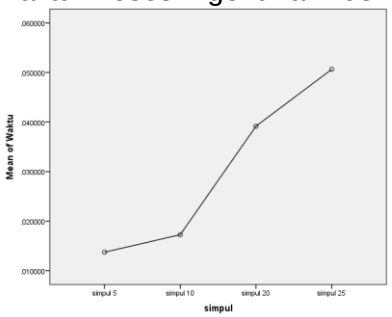
Keputusan :

- Pengaruh Populasi :
  Karena α =0.05 < sig.=0.07, maka H_0 ditolak dan H_1 diterima, atau dengan kata lain bahwa perubahan populasi berpengaruh pada hasil MST genetika yang dihasilkan.
- Pengaruh *Crossover* :
  Karena α =0.05 < sig.=0.218, maka H_0 ditolak dan H_1 diterima, atau dengan kata lain bahwa perubahan

nilai *crossover* berpengaruh pada hasil MST genetika yang dihasilkan.

- Pengaruh Mutasi :
  Karena α =0.05 < sig.=0.211, maka H_0 ditolak dan H_1 diterima, atau dengan kata lain bahwa perubahan nilai mutasi berpengaruh pada hasil MST genetika yang dihasilkan.

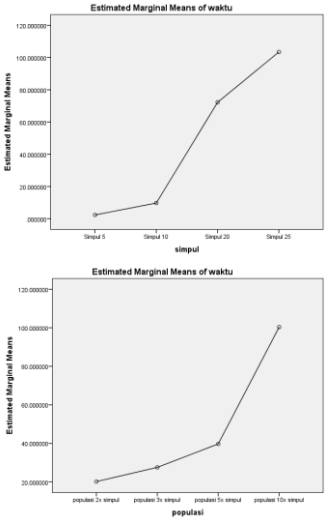
2. Perbandingan Waktu Proses Algoritma Kruskal dan Algoritma Genetika
   a. Waktu Proses Algoritma Kruskal



Gambar 16. Grafik perbandingan jumlah simpul dan perubahan Waktu

Dari Gambar 16 diatas disimpulkan, jumlah waktu proses yang dibutuhkan oleh Algoritma Kruskal dalam menghitung nilai MST sangat berpengaruh pada jumlah simpul. Semakin besar jumlah simpul, maka akan semakin lama waktu yang dibutuhkan oleh Algoritma Kruskal memproses nilai MST.

   b. Waktu Proses Algoritma Genetika



Gambar 17. Grafik perubahan waktu Algoritma Genetika

Dari plot waktu Gambar 17 diatas, disimpulkan jumlah waktu proses yang dibutuhkan oleh Algoritma Genetika dalam menghitung nilai MST sangat berpengaruh pada jumlah simpul dan populasi. Semakin besar jumlah simpul dan populasi, maka akan semakin lama waktu yang dibutuhkan oleh Algoritma genetika memproses nilai MST.

KESIMPULAN

Berdasarkan hasil penelitian, analisis dan pembahasan yang telah dilakukan, maka dapat disimpulkan :

1. Penelitian ini menghasilkan aplikasi penyelesaian masalah *Minimum Spanning Tree* (MST) menggunakan Algoritma Kruskal dan Algoritma Genetika.
2. Dari hasil pengujian untuk simpul < 25, didapatkan beberapa kesimpulan yaitu :
   a. Algoritma Kruskal menghasilkan nilai jarak minimum untuk penyelesaian permasalahan *Minimum Spanning Tree* lebih baik dari Algoritma genetika, dimana nilai MST yang dihasilkan yaitu untuk simpul = 5, nilai MST yang dihasilkan kruskal adalah 22.27, sedangkan MST genetika = 22.46. untuk simpul = 10, nilai MST kruskal = 47.81, sedangkan MST genetika = 62.14, untuk simpul = 20, nilai MST kruskal = 90.37, sedangkan MST genetika = 183.36 untuk simpul = 25, nilai MST kruskal = 124.39, sedangkan MST genetika = 284.06
   b. Waktu proses yang dibutuhkan oleh Algoritma Kruskal dalam menyelesaikan permasalahan *Minimum Spanning Tree* (MST) jauh lebih cepat dibandingkan Algoritma Genetika, dimana rata-rata waktu proses algoritma Kruskal adalah 0.030 detik, dan Algoritma Genetika adalah 46.95 detik.
3. Dari tabel ANOVA (*Analisys of Varians)* didapatkan :
   a. Nilai MST Algoritma Kruskal hanya dipengaruhi oleh jumlah simpul, semakin banyak jumlah simpul yang di-*input*-kan, maka akan semakin besar nilai MST yang dihasilkan

b. Pada Algoritma Genetika, Jumlah Populasi, Probabilitas *Crossover* dan Probabilitas Mutasi sangat mempengaruhi nilai MST
c. Algoritma Genetika akan menghasilkan nilai minimum yang cukup baik pada saat parameter peluang *crossover* kecil, yaitu 0.1 – 0.3 dari populasi di *crossover*, dan mutasi = 0 serta kelestarian = 1 , dimana seluruh populasi awal yg telah di *generate* dijadikan individu untuk iterasi selanjutnya.
d. Waktu proses Algoritma Kruskal dipengaruhi hanya oleh jumlah simpul, sedangkan pada Algoritma Genetika, waktu proses dipengaruhi oleh jumlah simpul dan jumlah populasi yang di-*input*-kan.

SARAN

Berdasarkan hasil penelitian, pengujian dan pembahasan yang telah dipaparkan, maka penulis menyarankan untuk mengembangkan penelitian dimasa yang akan datang sebagai berikut :

1. Permasalahan graf berkembang menjadi jenis graf lain, misalnya direct graf, graf non-simetris,dsb
2. Penelitian dapat dikembangkan dan diperluas untuk pencarian jarak minimum dengan menerapkannya langsung pada studi kasus dengan jarak yang sebenarnya
3. Penelitian dapat dikembangkan dengan membandingkan Algoritma Kruskal dan/atau Algoritma Genetika dengan algoritma-algoritma lainnya, sehingga didapatkan alggoritma yang paling baik dalam penyelesaian permasalahan nilai minimum pada suatu graf

Referensi

[1] R. Munir, "Diktat Kuliah IF2153 Matematika Diskrit Edisi Keempat," Departemen Teknik Informatika, Institut Teknologi Bandung, Bandung, 2006.

[2] Ayu, *Pohon (Tree),* p. http://ayu_ws.staff.gunadarma.ac.id/Downloads/files/33383/05+Pohon+(Tree).pdf, 2012.

[3] R. Efendi, "Penerapan Algoritma Semut Untuk Pemecahan Masalah Spanning Tree Pada Kasus Pemasangan Karingan Kabel Telepon.," Universitas Islam Indonesia, Yogyakarta, 2003.

[4] S. Kusumadewi, Artificial Intelligence Teknik dan Aplikasinya, Yogyakarta: Graha Ilmu, 2003.

[5] I. Lubis, Studi Perbandingan Algoritma Prim Algoritma Kruskal Dan Algoritma Sollin Dalam Menentukan Pohon Merentang Maksimum, Medan: Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Sumatra Utara, 2011.

[6] A. Raupong, "Bahan Ajar Mata Kuliah Perancangan Percobaan. Program Studi Statistika," Jurusan Matematika Fakultas Matematika & Ilmu pengetahuan Alam Universitas hasanuddin, 2011.

[7] I. Sommerville, Software Engineering Edisi 6 Jilid 1, Jakarta: Erlangga, 2003.