PERBANDINGAN PENERAPAN ALGORITMA MODIFIKASI SOLLIN DAN ALGORITMA MODIFIKASI PRIM UNTUK MENYELESAIKAN *MINIMUM ROUTING-COST SPANNING TREE (MRCST) PROBLEM*

(Tesis)

Oleh

JANI SUPARMAN
2227031010

PROGRAM STUDI MAGISTER MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG

2024

ABSTRACT

COMPARISON OF THE APPLICATION OF MODIFIED SOLLIN ALGORITHM AND MODIFIED PRIM ALGORITHM TO SOLVE THE MINIMUM ROUTING-COST SPANNING TREE (MRCST) PROBLEM

By

JANI SUPARMAN

The Prim and Sollin algorithms are popular algorithms for solving the Minimum Spanning Tree (MST) problem. In this study, we compare the application of the Modified Sollin Algorithm and Modified Prim Algorithm to solve the Minimum Routing-Cost Spanning Tree (MRCST) problem. The MRCST problem aims to find the lowest routing cost of a spanning tree in a weighted connected graph. Although not as popular as other problems like the travelling salesman, MRCST is highly complex and has been the focus of research by several scholars. The use of Python programming language facilitates and improves efficiency in the implementation of the modified Prim and Sollin algorithms to solve the MRCST problem. The objective of this study is to gain a better understanding of the performance of the Modified Sollin Algorithm and Modified Prim Algorithm in solving the MRCST problem, as well as to evaluate the influence of parameters on the results obtained by both algorithms. The implementation results of the source code in this study show that, on average, the performance of the Modified Sollin Algorithm is better for solving the MRCST problem, while the Modified Prim Algorithm outperforms the Modified Sollin Algorithm in several orders, namely 10, 20, 30, and 80.

Keywords: MRCST, Prim Algorithm, Sollin Algorithm, Modified Sollin Algorithm, Modified Prim Algorithm.

ABSTRAK

PERBANDINGAN PENERAPAN ALGORITMA MODIFIKASI SOLLIN DAN ALGORITMA MODIFIKASI PRIM UNTUK MENYELESAIKAN *MINIMUM ROUTING-COST SPANNING TREE (MRCST) PROBLEM*

Oleh

JANI SUPARMAN

Algoritma Prim dan Sollin merupakan algoritma yang populer untuk menyelesaikan masalah *Minimum Spanning Tree* (MST). Pada penelitian ini, akan dibandingkan penerapan Algoritma Modifikasi Sollin dan Algoritma Modifikasi Prim dalam menyelesaikan masalah *Minimum Routing-Cost Spanning Tree* (MRCST). Masalah MRCST bertujuan untuk menemukan *routing cost* terendah dari *spanning tree* dalam sebuah graf terhubung berbobot. Meskipun tidak sepopuler masalah lain seperti *Travelling Salesman Problem* (TSP), MRCST memiliki kompleksitas yang tinggi dan telah menjadi fokus penelitian beberapa peneliti. Penggunaan program Python untuk memudahkan dan meningkatkan efisiensi waktu implementasi algoritma modifikasi Prim dan Sollin untuk menyelesaikan masalah MRCST. Tujuan penelitian ini adalah untuk memperoleh pemahaman yang lebih baik tentang kinerja Algoritma Modifikasi Sollin dan Algoritma Modifikasi Prim dalam menyelesaikan masalah MRCST, serta untuk mengevaluasi pengaruh parameter-parameter pada kedua algoritma tersebut terhadap hasil yang diperoleh. Hasil implementasi *source code* pada penelitian ini diperlihatkan bahwa secara rata-rata kinerja Algoritma Modifikasi Sollin lebih baik untuk menyelesaikan masalah MRCST, serta Algoritma Modifikasi Prim memiliki hasil yang lebih baik dibanding Algoritma Modifikasi Sollin pada beberapa orde yaitu 10,20,30, dan 80.

Kata kunci: MRCST, Algoritma Prim, Algoritma Sollin, Algoritma Modifikasi Sollin, Algoritma Modifikasi Prim.

PERBANDINGAN PENERAPAN ALGORITMA MODIFIKASI SOLLIN DAN ALGORITMA MODIFIKASI PRIM UNTUK MENYELESAIKAN *MINIMUM ROUTING-COST SPANNING TREE (MRCST) PROBLEM*

Oleh

JANI SUPARMAN

(Tesis)

Sebagai Salah Satu Syarat untuk Memperoleh Gelar
MAGISTER MATEMATIKA

Pada

Jurusan Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Lampung

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2024

Judul	:	**PERBANDINGAN PENERAPAN ALGORITMA MODIFIKASI SOLLIN DAN ALGORITMA MODIFIKASI PRIM UTUK MENYELESAIKAN *MINIMUM ROUTING-COST SPANNING TREE (MRCST) PROBLEM***
Nama Mahasiswa	:	*Jani Suparman*
NPM	:	**2227031010**
Jurusan	:	**Matematika**
Fakultas	:	**Matematika dan Ilmu Pengetahuan Alam**

Bandar Lampung, 10 Juni 2024

Menyetujui,

1. Komisi Pembimbing

Prof. Dra. Wamiliana, M.A., Ph.D.
NIP.19631108 198902 2 001

Dr. Muslim Ansori, S.Si., M.Si.
NIP. 19720227 199802 1 001

2. Ketua Program Studi Megister Matematika

Prof. Dr. Asmiati, S.Si., M.Si.
NIP. 19760411 20001 2 2001

MENGESAHKAN

1. Tim Penguji

   Ketua : **Prof. Dra. Wamiliana, M.A., Ph.D.** .....

   Sekretaris : **Dr. Muslim Ansori, S.Si., M.Si.** .....

   Penguji
   Bukan Pembimbing : **1. Dr. Notiragayu, S.Si., M.Si.** .....

   2. Prof. Dr. Asmiati, S.Si., M.Si. .....

2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

   Dr. Eng. Heri Satria, S.Si., M.Si.
   NIP. 197110012005011002

3. Direktur Program Pascasarjana

   Prof. Dr. Ir. Muhadi, M.Si.
   NIP. 19640326 199802 1 001

Tanggal Lulus Ujian Tesis: **11 Juni 2024**

PERNYATAAN TESIS MAHASISWA

Yang bertanda tangan di bawah ini:

Nama	: **Jani Suparman**
Nomor Pokok Mahasiswa	: **2227031010**
Program Studi	: **Magister Matematika**
Jurusan	: **Matematika**

Dengan ini menyatakan bahwa tesis saya berjudul, **"PERBANDINGAN PENERAPAN ALGORITMA MODIFIKASI SOLLIN DAN ALGORITMA MODIFIKASI PRIM UTUK MENYELESAIKAN *MINIMUM ROUTING-COST SPANNING TREE (MRCST) PROBLEM*"** adalah hasil pekerjaan saya sendiri. Semua tulisan yang tertuang dalam tesis ini mengikuti kaidah karya penulisan ilmiah Universitas Lampung. Apabila kemudian hari terbukti bahwa tesis ini merupakan hasil salinan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi sesuai dengan ketentuan akademik yang berlaku.

Bandar Lampung, 11 Juni 2024

Yang Menyatakan,

Jani Suparman

NPM. 2227031010

RIWAYAT HIDUP

Penulis bernama Jani Suparman, lahir di Pringsewu pada tanggal 29 Desember 1999, dan merupakan anak ketiga dari lima bersaudara dari pasangan Bapak Suparman dan Ibu Fajar Suami.

Penulis menenempuh pendidikan Sekolah Dasar di SD Negeri 2 Yogyakarta Gadingrejo Pringsewu pada tahun 2006 sampai dengan 2012. Kemudian melanjutkan pendidikan Sekolah Menengah Pertama di SMP Negeri 3 Gadingrejo Pringsewu pada tahun 2012 sampai dengan 2015. Selanjutnya, penulis menempuh pendidikan Sekolah Menengah Atas di SMA Negeri 1 Gadingrejo Pringsewu pada tahun 2015 sampai dengan 2018.

Pada tahun 2018 penulis melanjutkan pendidikan Strata Satu (S1) Program Studi S1 Matematika dan Strata Dua (S2) Program Studi Magister Matematika tahun 2022 Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam (FMIPA) Universitas Lampung (Unila).

KATA INSPIRASI

“Pengetahuan tentang segala hal adalah sebuah kemungkinan”

(Leonardo da Vinci)

“Setelah Anda menghilangkan hal yang tidak mungkin, apa pun yang tersisa, tidak peduli betapa tidak mungkinnya, haruslah kebenaran.”

(Sir Arthur Conan Doyle)

"Belajar dari matematika, lakukanlah dulu apapun yang kau bisa lakukan karena sesulit apapun masalahnya selalu ada banyak cara yang mungkin untuk menyelesaikannya dan dibalik semua kemungkinan itu setidaknya selalu ada satu kebenaran."

(penulis)

PERSEMBAHAN

Puji dan syukur tiada hentinya kepada Allah Subhanahu Wata'ala atas segala nikmat dan hidayah-Nya untuk kita semua. Penulis persembahkan sebuah karya sederhana ini untuk:

Mamak Fajar Suami dan Alm. Bapak Suparman, terimakasih atas kasih sayang, pengorbanan, doa, dan motivasi yang telah diberikan di setiap langkah penulis. Karena atas doa dan ridho kalian, Allah memudahkan setiap perjalanan hidup ini.

Mbak Ayu, Mas Bayu, Adek Emay, dan Adek Firda terimakasih untuk selalu membantu, dan menemani penulis dalam keadaan suka maupun duka.

Sahabat dan teman-teman ku, terimakasih atas kebersamaan, keceriaan, canda dan tawa serta doa dan semangat yang telah diberikan kepadaku.

Almamater Universitas Lampung.

SANWACANA

Puji dan syukur penulis panjatkan kehadirat Allah SWT karena berkat limpahan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan tesis ini yang berjudul "*Perbandingan Penerapan Algoritma Modifikasi Sollin dan Algoritma Modifikasi Prim untuk Menyelesaikan Minimum Routing Cost Spanning Tree (MRCST) Problem* " disusun sebagai salah satu syarat memperoleh gelar Magister Matematika di Universitas Lampung.

Dalam proses penyusunan Tesis ini, banyak pihak yang telah membantu memberikan bimbingan, dukungan, motivasi, serta saran sehingga skripsi ini dapat diselesaikan. Untuk itu penulis mengucapkan terima kasih kepada:

1. Ibu Prof. Dra. Wamiliana, M.A., Ph.D. selaku Pembimbing I dan Pembimbing Akademik yang selalu bersedia memberikan arahan, bimbingan, saran serta dukungan kepada penulis sehingga dapat menyelesaikan tesis ini.
2. Bapak Dr. Muslim Ansori, S.Si., M.Si. selaku Pembimbing II yang telah memberikan arahan, bimbingan dan dukungan kepada penulis.
3. Ibu Dr. Notiragayu, S.Si., M.Si. selaku Penguji I yang telah bersedia memberikan kritik dan saran serta evaluasi kepada penulis sehingga dapat lebih baik lagi.

4. Ibu Prof. Dr. Asmiati, S.Si., M.Si. selaku Penguji II dan Ketua Prodi Magister yang telah bersedia memberikan kritik, saran, dan evaluasi kepada penulis sehingga dapat lebih baik lagi serta dukungan kepada penulis sehingga dapat menyelesaikan tesis ini.
5. Bapak Dr. Aang Nuryaman, S.Si., M.Si. selaku Ketua Jurusan Matematika.
6. Bapak Dr. Eng. Heri Satria, S.Si., M.Si. selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.
7. Bapak Prof. Dr. Ir. Muhardi, M.Si. selaku Direktur Program Pascasarjana Universitas Lampung.
8. Seluruh dosen, staff dan karyawan Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.
9. Orang tua tercinta serta kakak dan adek yang selalu memotivasi dan mendukung penulis dalam memberikan yang terbaik, selalu mendoakan untuk kesuksesan penulis.
10. Teman-teman mahasiswa Magister Matematika angkatan 2022 dan pihak yang membantu dalam pengerjaan tesis ini yang tidak dapat disebutkan satu-persatu.

Penulis menyadari bahwa dalam penyusunan tesis ini masih memiliki banyak kekurangan, sehingga kritik dan saran yang membangun diharapkan untuk penyempurnaan tesis ini. Penulis berharap semoga tesis ini dapat memberikan manfaat bagi pembaca.

Bandar Lampung, 11 Juni 2024

Penulis

Jani Suparman

DAFTAR ISI

Halaman

DAFTAR TABEL

Tabel Halaman

DAFTAR GAMBAR

I. PENDAHULUAN

1.1. Latar Belakang

Graf merupakan kajian dalam matematika yang dapat digunakan untuk mempresentasikan hubungan antara objek-objek diskrit. Salah satu kasus dari graf yaitu mencari *spanning tree*. Misalkan $G = (V, E)$ merupakan graf terhubung yang tak berarah dan mengandung sebuah sirkuit, maka G dapat diubah menjadi *tree* $T = (V_1, E_1)$ dengan dilakukannya penghapusan pada sirkuit tersebut, sehingga sirkuit yang terdapat pada G hilang dan G dapat menjadi sebuah *tree* T, dimana *tree* T ini dinamakan *spanning tree* (Ramadhan, 2017).

Kajian pada *spanning tree* terbagi menjadi dua, yaitu pohon merentang maksimum (*maximum spanning tree*) dan pohon merentang minimum (*minimum spanning tree*/MST). Pohon merentang maksimum (*maximum spanning tree*), yaitu teknik penentuan *spanning tree* dari G berbobot maksimal, sedangkan *minimum spanning tree* yaitu teknik mencari jalan terdekat dalam jaringan sampai diperoleh jarak minimum. Persoalan MST merupakan variasi dari persoalan rute terpendek, Salah satu harus ditentukan sisi yang memiliki bobot minimal yang akan menyatukan titik-titik yang terdapat pada sebuah jaringan, sehingga pada hasil akhir dapat diperoleh total panjang sisi yang minimum dan mengetahui keoptimalan suatu jaringan graf tersebut. Salah satu kasus yang dapat diselesaikan dengan menggunakan MST yaitu rute terpendek (Ismail dan Setiady, 2014).

MST umumnya banyak diaplikasikan dalam masalah desain jaringan di mana parameter graf lain seperti jarak, derajat, diameter, aliran, koneksi, dan lain-lain harus dipenuhi. Penelitian tentang MST telah dilakukan secara ekstensif. Pada tahun 1956 Kruskal memperkenalkan algoritma Kruskal, dan pada tahun berikutnya yaitu tahun 1957 Prim memperkenalkan algoritma Prim untuk menyelesaikan MST. Kedua algoritma tersebut telah digunakan secara luas dan sangat populer. Namun, algoritma pertama yang mengatasi MST diusulkan oleh Boruvka pada tahun 1926 ketika ia menyelesaikan masalah pembangunan jaringan listrik Moravia di Republik Ceko.

Diberikan sebuah graf terhubung berbobot, *routing cost* dari *spanning tree* didefinisikan sebagai jumlah panjang total jalur dari semua pasangan titik dalam *spanning tree* dari graf tersebut. MRCST bertujuan untuk menemukan *routing cost* terendah dari *spanning tree*. MRCST juga dikenal sebagai *shortest average distance spanning tree* dan dalam graf tak berbobot disebut Indeks Minimum Wiener.

Meskipun tidak sepopuler *travelling salesman problem* di mana banyak peneliti telah terlibat dalam topik tersebut, beberapa peneliti telah menyelidiki masalah MRCST, termasuk Wu (2002); Wolf dan Merz (2010); Chen dkk. (2013); Tan dan Due (2013); Lin dkk. (2006), Sattari dan Didehvar (2013); dan Sari dkk (2022). Karena MRCST merupakan masalah NP-*hard*, maka di dalam perkembangannya algoritma heuristik lebih banyak diselidiki untuk menyelesaikan masalah MRCST, misalnya, Singh (2008) dan Tan (2012a). Singh dan Sundar (2011) menyelidiki algoritma koloni lebah, dan Hieu dkk. (2011) mengusulkan algoritma koloni semut. Algoritma genetika untuk menyelesaikan masalah MRCST telah diselidiki oleh Tan (2012b); Julstrom (2001) dan Julstrom (2005). Julstrom (2005) juga mengkodekan pohon menggunakan kode Blob dan menunjukkan bahwa merepresentasikan pohon menggunakan kode Blob dalam algoritma genetika lebih baik daripada mengkodekan pohon sebagai himpunan sisi yang diusulkan oleh Raidl dan

Julstrom (2003). Sattari dan Didehvar (2015) mengusulkan GRASP dengan algoritma *path-relinking* metaheuristik untuk menyelesaikan MRCST.

Berdasarkan dari uraian yang telah dipaparkan tentang masalah MRCST, maka dari itu diperlukan suatu algoritma yang lebih baik dalam menyelesaikan masalah MRCST pada suatu graf terhubung dan berbobot. Penulis tertarik untuk membandingkan penerapan Algoritma Modifikasi Sollin dan Algoritma Modifikasi Prim dan algoritma mana yang mempunyai hasil lebih baik, maka penulis memutuskan dan mengajukan penelitian ini dengan judul "Perbandingan Penerapan Algoritma Modifikasi Sollin dan Algoritma Modifikasi Prim untuk menyelesaikan *Minimum Routing-Cost Spanning Tree (MRCST) Problem*".

1.2. Rumusan Masalah

Bagaimana pengaruh parameter-parameter pada algoritma modifikasi Sollin dan algoritma modifikasi Prim terhadap hasil yang diperoleh dalam menyelesaikan permasalahan MRCST?

1.3. Tujuan Penelitian

Membandingkan kinerja Algoritma Modifikasi Sollin dan Algoritma Modifikasi Prim untuk menyelesaikan *Minimum Routing-Cost Spanning Tree (MRCST) Problem.*

1.4. Manfaat Penelitian

1. Memberikan kontribusi dalam pengembangan yang lebih efektif dan efisien dalam desain jaringan.
2. Meningkatkan efisiensi dalam industri dengan mengoptimalkan perjalanan rute yang mempertimbangkan seperti jarak.

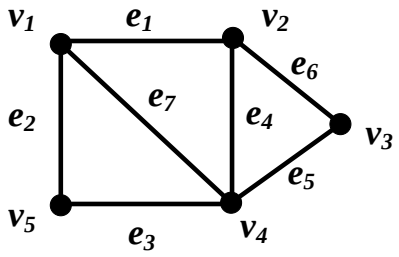
3. Menawarkan solusi alternatif dalam menyelesaikan masalah optimasi seperti MRCST, yang dapat diterapkan pada berbagai bidang lainnya, seperti logistik dan perencanaan rute.
4. Memberikan pengetahuan dan wawasan baru dalam penggunaan algoritma Sollin dan algoritma Prim dalam menyelesaikan masalah optimasi.

II. TINJAUAN PUSTAKA

Pada bab ini akan diberikan beberapa definisi dan teorema yang berhubungan dengan penelitian yang akan dilakukan.

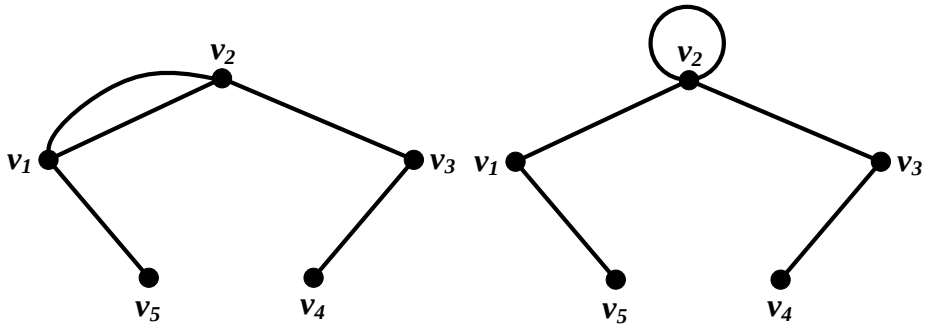
2.1. Konsep Graf

Graf $G = (V, E)$ didefinisikan sebagai pasangan terurut suatu himpunan $(V(G), E(G))$ dengan $V(G) = \{v_1, v_2, \dots, v_n\}$ merupakan himpunan titik, $V(G) \neq \emptyset$, dan $E(G) = \{e_1, e_2, \dots, e_n\}$ merupakan himpunan sisi dari pasangan tak terurut titik titik di $V(G)$.



Gambar 2.1.1. Graf dengan 5 titik dan 7 sisi

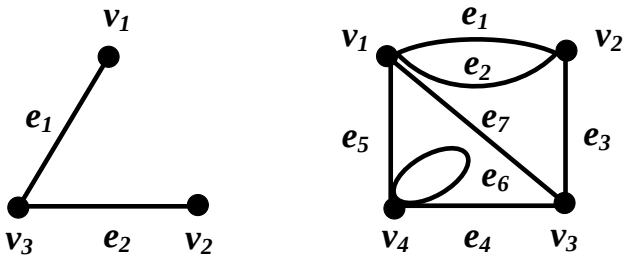
Suatu sisi yang titik awal dan titik akhirnya sama disebut *loop*, sedangkan sisi paralel adalah dua sisi atau lebih yang menghubungkan titik- titik yang sama. Graf sederhana adalah graf yang tidak memuat *loop* atau sisi paralel, sedangkan jika memuat *loop* atau sisi paralel, maka disebut graf tak sederhana.



(a) (b)

Gambar 2.1.2. (a) Graf dengan sisi paralel (b) Graf dengan *loop*

Pada Gambar 2.1.3. dapat dilihat bahwa pada Gambar 2.1.3. (a) merupakan contoh graf sederhana dengan tiga titik dan dua sisi, sedangkan pada Gambar 2.1.3. (b) merupakan graf tidak sederhana dengan *loop* e_6 dan sisi paralel e_1 dan e_2. Misalkan v_j merupakan titik ujung sisi e_j pada suatu graf G, v_j dan e_j dikatakan *incidence* (menempel) satu sama lain. Dua sisi tak paralel dikatakan *adjacent* (bertetangga) jika keduanya menempel pada suatu titik yang sama. Dua titik dikatakan *adjacent* (bertetangga) jika terdapat sisi yang menghubungkan keduanya.



(a) (b)

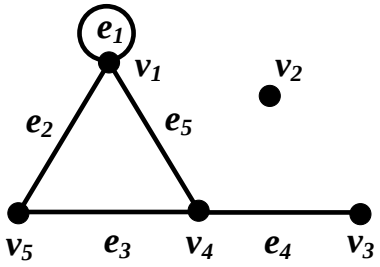
Gambar 2.1.3 (.a) Graf Sederhana (b) Graf tidak sederhana

Misalkan pada Gambar 2.1.3. (a) sisi e_1 menempel pada titik v_1 dan titik v_3, dan sisi e_2 menempel pada titik v_2 dan v_3. Titik v_1 bertetangga dengan titik v_3, titik v_2 bertetangga dengan v_3, serta titik v_3 bertetangga dengan v_1 dan v_2.

Walk adalah barisan berhingga dari suatu titik dan sisi yang dimulai dan diakhiri dengan titik, sedemikian sehingga setiap sisi menempel pada titik sebelum dan sesudahnya. *Walk* yang berawal dan berakhir pada titik yang sama disebut *closed walk*. *Walk* yang melewati titik yang berbeda-beda disebut sebagai *path* (lintasan). *Path* yang berawal dan berakhir pada titik yang sama disebut *cycle*. Suatu graf G disebut graf terhubung (*connected graph*) jika terdapat sekurang-kurangnya ada satu *path* yang menghubungkan setiap pasangan titik di G, jika tidak maka G disebut graf tak terhubung.

Pada suatu graf terhubung G, distance atau jarak $d(v_i, v_j)$ antara dua titik v_i dan v_j adalah panjang dari *path* terpendek. Eksentrisitas $e(v)$ dari sebuah titik v pada graf G adalah jarak dari v ke titik terjauh dari $v \in G$ dinyatakan sebagai: $e(v) = \max_{v_1 \in G} d(v, v_1)$. Diameter suatu graf G adalah eksentrisitas maksimum dari titik-titiknya atau dapat dituliskan dalam persamaan sebagai: $Diam(G) = max\{ e(u) | u \in V\}$. Jari-jari (radius) dari suatu graf G yaitu eksentrisitas minimum dari titik-titiknya, dapat dituliskan sebagai: $rad(G) = min\{ e(u) | u \in V \}$. Pusat (Center) suatu graf terdiri dari titik-titik yang eksentrisitasnya sama dengan jari-jari graf, dapat dituliskan sebagai: $Center\ (G) = \{u \in V \mid e(u) = rad(G)\}$

Derajat (*degree*) dari suatu titik v pada graf G dinotasikan $\deg(v)$, adalah banyaknya sisi yang menempel pada titik v dengan *loop* dihitung dua. Untuk contoh dapat dilihat pada Gambar 2.4. bahwa $d(v_1) = 4, d(v_2) = 0,$ $d(v_3) = 1.$



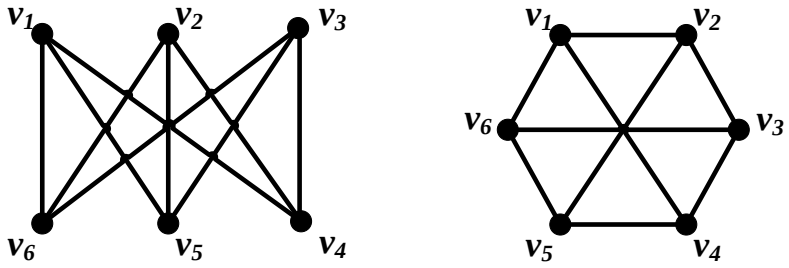
Gambar 2.1.4. Graf dengan 1 titik terasing dan 1 titik *pendant*

Titik terasing merupakan titik yang memiliki derajat nol, sedangkan titik *pendant* (daun) adalah titik yang memiliki derajat satu.

Dua graf dikatakan ekuivalen (dan disebut isomorfik) jika keduanya memiliki ciri-ciri yang sama pada istilah dalam teori graf. Dua graf G dan G' dikatakan isomorfik jika ada korespondensi 1-1 antara titik-titik pada kedua graf tersebut dan antara sisi-sisi keduanya sehingga jika sisi e bersisian dengan titik u dan v pada G maka sisi e' pada G' juga bersisian dengan titik u' dan v'. Dua graf isomorfik harus memiliki:

1. banyak titik yang sama;
2. banyak sisi yang sama;
3. titik-titik yang berkorespondensi mempunyai derajat yang sama.

Perlu diperhatikan bahwa dua graf yang mempunyai sifat 1 sampai dengan 3, belum tentu kedua graf tersebut isomorfik (Deo, 1989).



Gambar 2.1.5. Contoh graf yang saling isomorfik

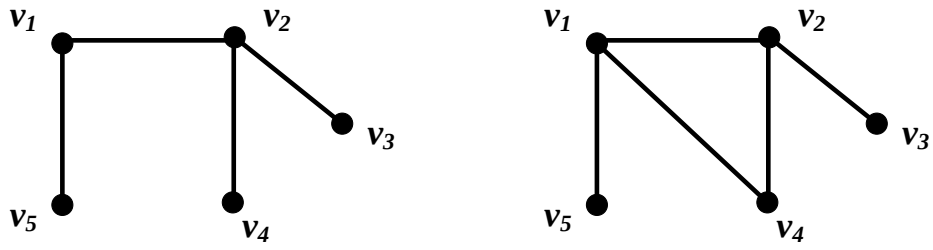
Graf H dikatakan subgraf dari G jika setiap titik dan sisi di H juga merupakan titik dan sisi di G, dengan kata lain $V(H) \subseteq V(G)$ dan $E(H) \subseteq E(G)$ (Wamiliana, 2022).

Sejumlah masalah yang berhubungan dengan graf yang ditemukan manusia dalam kehidupan nyata menimbulkan penemuan konsep-konsep pemecahan masalah graf. Konsep *tree* pernah diterapkan dalam perhitungan molekul kimia oleh Arthur Cayley (matematikawan Inggris) pada tahun 1870-an.

Karya yang lebih baru membuktikan bahwa *tree* telah digunakan di banyak bidang, mulai linguistik sampai komputer (Wilson dan Watkins, 1990).

Tree adalah graf terhubung yang tidak mengandung sirkuit. Menurut definisi tersebut, ada dua sifat penting pada *tree* yaitu terhubung dan tidak mengandung sirkuit (Chartrand dan Lesniak, 1986)

Di dalam suatu *tree,* titik berderajat 1 dinamakan daun (*leaf*) ata titik terminal (*terminal node*), sedangkan titik yang berderajat lebih dari 1 dinamakan titik cabang (*branch node)* atau titik internal *(internal node).*



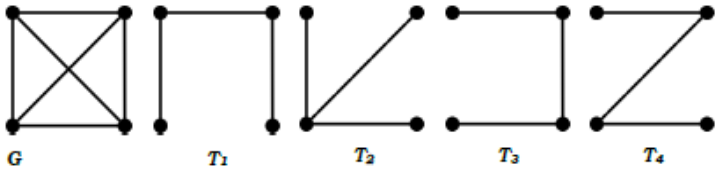
(a) (b)

Gambar 2.1.6. (a) *Tree*, (b) Bukan *Tree*

Diameter dari suatu *tree* T didefinisikan sebagai eksentrisitas di T. (Deo, 1989)

Suatu graf G merupakan graf *acyclic* apabila dalam graf tersebut tidak terdapat sirkuit. *Forest* adalah graf *acyclic* dimana setiap komponen terhubungnya merupakan *tree* (Roman, 1989).

Misalkan G merupakan suatu graf terhubung yang bukan *tree*, berarti G mengandung suatu sirkuit. G dapat diubah menjadi *tree* $T = (V_1, E_1)$ dengan cara menghilangkan semua sirkuit pada graf tersebut, sehingga pada graf tersebut tidak mengandung satu sirkuit pun, maka graf tersebut dapat dikatakan *spanning tree* (Munir, 2014).



Gambar 2.1.7. Graf lengkap *G* dan empat buah *spanning tree*-nya, *T1*, *T2*, *T3*, *T4*.

Suatu sub-graf yang mempunyai bobot minimum dari sub-graf lainnya pada graf dinamakan sebagai MST (Ramadhan, 2017). Permasalahan yang terjadi pada MST hampir menyerupai permasalahan untuk mencari lintasan terpendek, dimana perbedaannya terlihat pada rute yang ditempuh serta jumlah titik. Pencarian MST, setiap titik wajib saling terhubung dan menghasilkan jarak terdekat maksimal dari graf tersebut dan tidak boleh membentuk sirkuit. Sedangkan untuk pencarian lintasan terpendek, setiap titik yang ada tidak wajib terhubung satu sama lain (Ismail dan Setiady, 2014).

Diberikan $G = (V, E, w)$ merupakan graf terhubung tak berarah dengan bobot sisi non-negatif (Costs); dengan V merupakan himpunan n titik, E merupakan himpunan m sisi, w(e) merupakan bobot sisi e, $e \in E$. Misalkan T merupakan *spanning tree* dari G, maka *routing cost* dari suatu pasangan titik (u, v) dalam T , dinotasikan dengan $dT(u, v)$ yang merupakan jumlah dari *cost* sisi pada jalur yang menghubungkan titik u dan titik v dalam T. *Routing cost* dari T dinotasikan dengan $C(T)$ yang didefinisikan sebagai total *routing-cost* dari semua pasangan titik dalam T, dapat dituliskan dalam persamaan:

$$C(T) = \sum_{u,v \in V} d_T(u, v) \qquad (1)$$

Menghitung RCST dari lintasan yang memiliki n titik dalam persamaan (1) memakan waktu $O(n^2)$. Namun dengan *routing load*, dapat dihitung biaya RCST dengan *linear time.* Diberikan *T spanning tree* dengan himpunan sisi $E(T)$. Jika menghapus sebuah sisi e dari T maka T kemudian terbagi menjadi

dua *subtree* T_1 dan T_2, yang memiliki himpunan titik masing-masing $V(T_1), V(T_2)$. *Routing load* dari sisi e didefinisikan sebagai $l(T,e) = 2 \times |V(T_1)| \times |V(T_2)|$. Jadi *routing cost* dari T dapat dirumuskan sebagai berikut:

$$C(T) = \sum_{u,v \in V} l(T,e) \times w(e) \qquad (2)$$

Masalah MRCST terbukti termasuk kelas NP-*hard*. Bobot sisi dan topologi lintasan tree adalah dua faktor yang memengaruhi *spanning tree routing cost*. (Wu dan Chao,2004).

2.2. Algoritma *Sollin*

Algoritma *Sollin* dapat dikatakan gabungan dari Algoritma Prim dan juga Algoritma Kruskal. Pada setiap iterasi, algoritma Solin menggunakan konsep tetangga terdekat untuk menghasilkan jarak minimum antara dua titik. Jadi, pada setiap titik dilakukan pencarian titik dengan jarak terkecil dari titik tersebut (Anggraeni, 2015)

Algoritma Sollin, juga dikenal sebagai algoritma Borůvka, pertama kali dipublikasikan oleh Otakar Borůvka pada tahun 1926. Konsep awal dari algoritma *Sollin* ini adalah menggabungkan setiap sisi dengan bobot terendah dari setiap titik di G. Pada tahap awal dari algoritma *Sollin,* untuk tiap titik pilih sisi dengan bobot terendah yang menempel dengan titik tersebut. Dalam algoritma *Sollin* ada kemungkinan terbentuknya suatu *forest.*

Adapun langkah-langkah yang dilakukan untuk menggunakan algoritma *Sollin,* yaitu sebagai berikut:

Inisiasi: Graf G dengan n titik dan m sisi.

Langkah-langkah:

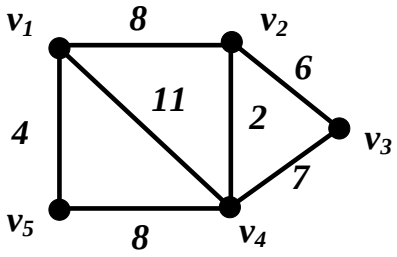
1. Daftarkan semua titik yang ada di graf G.
2. Untuk setiap titik: pilih sisi dengan bobot yang terkecil yang menempel dengan titik tersebut. Jika terjadi ada dua sisi dengan bobot terkecil yang sama, pilih satu. (Pada proses ini mungkin saja terbentuk *forest* yang merupakan gabungan dari k komponen dari Graf G).

3. Tandai sisi-tersebut dengan ketentuan bahwa sisi-sisi yang terhubung diberi tanda yang sama. Banyak tanda menunjukan banyaknya k komponen yang terbentuk.
4. Tentukan sisi terkecil yang berada diluar masing-masing komponen.
5. Hubungkan sisi terkecil yang tersebut, yang menghubungkan dua komponen. Untuk dua komponen yang baru terhubung, ganti tanda salah satu komponen menjadi sama dengan komponen lainnya (sehingga jumlah komponen menjadi $k - 1$).
6. Ulangi langkah 4 sehingga $k = 1$.
7. Hapus sisi-sisi yang tidak terpakai.
8. Selesai, MST didapat.

(Wamiliana, 2022)

Contoh:

Diberikan suatu graf G sebagai berikut:

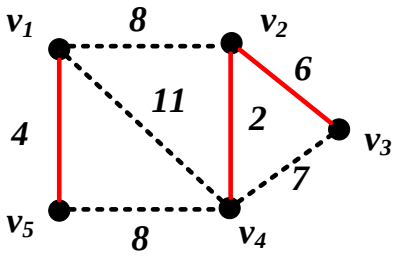


Gambar 2.2.1. Graf G untuk contoh penentuan MST.

Algoritma Sollin untuk MST dari graf G tersebut sebagai berikut:

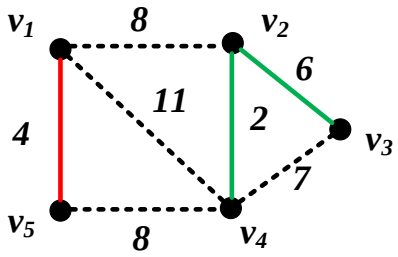
1. $V = \{v_1, v_2, v_3, v_4, v_5\}$

2. Pilih sisi dengan bobot yang terkecil yang menempel dengan masing-masing titik. Sisi yang dipilih diberi warna merah seperti pada Gambar 2.2.2.



Gambar 2.2.2. Graf G pada langkah 2.

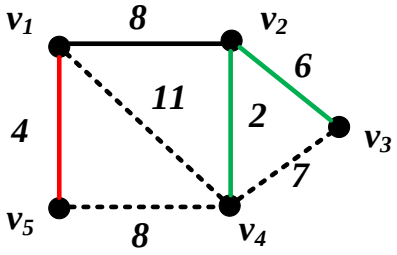
3. Beri tanda masing-masing komponen dengan tanda yang berbeda seperti pada Gambar 2.2.3.



Gambar 2.2.3. Graf G pada langkah 3

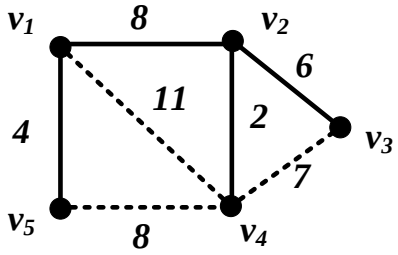
4. Tentukan sisi terkecil yang berada diluar masing-masing komponen.

5. Hubungkan sisi terkecil yang tersebut.



Gambar 2.2.4. Graf G pada langkah 5

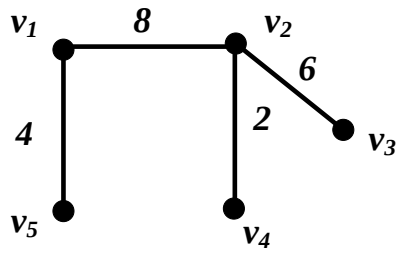
Ganti tanda sama dengan komponen lainnya seperti Gambar 2.12.



Gambar 2.2.5. Graf G diberi warna yang sama

6. Karena $k = 1$ lanjut langkah selanjutnya

7. Hapus sisi-sisi yang tidak terpakai. Selesai, MST didapat seperti pada Gambar 2.2.6.



Gambar 2.2.6. MST dari Graf G

2.3. Algoritma Prim

Masalah *MST* dapat dipecahkan dengan bantuan suatu *tree* yang ditemukan oleh Prim pada tahun 1957. Algoritma ini biasa disebut dengan Algoritma Prim (Bondy dan Murty, 1976). Algoritma Prim adalah suatu Algoritma di dalam teori graf yang bertujuan menentukan suatu *MST* dari suatu graf terhubung yang berbobot. Metode ini digunakan untuk menemukan suatu subset dari sisi yang membentuk suatu *tree* yang melibatkan tiap-tiap titik, dimana total bobot dari semua sisi di dalam *tree* adalah minimum (Munir, 2014.

Pada Algoritma Prim, titik awal ditentukan secara acak. Untuk menentukan langkah selanjutnya, ditentukan dari titik awal yang dipilih sehingga memungkinkan terjadinya *cycle*, dan tidak memungkinkan terjadinya *forest*. Misalkan T adalah himpunan sisi di G dan V adalah himpunan titik di G. Berikut adalah Algoritma Prim (Wamiliana, 2022):

Inisiasi:$T = \emptyset, \mathrm{V} = \emptyset$.

Langkah – langkah Algoritma Prim adalah sebagai berikut:

1. Tentukan salah satu titik sebagai ”root”.
2. Masukkan root ke V.
3. Tentukan sisi minimum yang terhubung dengan root, pilih dan masukkan ke T. Titik ujung sisi tersebut masukkan ke V.
4. Pilih sisi minimum yang terhubung dengan titik – titik di V dan cek apakah membentuk sirkuit jika ditambahkan ke T. Jika ya, maka buang sisi tersebut

dan pilih lagi sisi minimum berikutnya. Jika tidak, masukkan sisi nya di T dan titik nya di V.

5. Cek Jika ya, STOP. Jika tidak, ulangi langkah 4.

Tabel 2.3.1. Contoh Prosedur Algoritma Prim untuk Graf pada Gambar 8.

Iter	Sisi yang Dipertim -bangkan	Sisi yang dipilih	Membentuk sikuit?		V	T	$\|T\| = n - 1$?		Ket
			Ya	Tidak			Ya	Tidak	
1	$e_{v_1,v_2} = 8$ $e_{v_1,v_4} = 11$ $e_{v_1,v_5} = 4$	$e_{v_1,v_5} = 4$		√	$\{v_1, v_5\}$	$\{e_{v_1,v_5}\}$		√	-
2	$e_{v_1,v_2} = 8$ $e_{v_1,v_4} = 11$ $e_{v_5,v_4} = 8$	$e_{v_1,v_2} = 8$		√	$\{v_1, v_2, v_5\}$	$\{e_{v_1,v_5}, e_{v_1,v_2}\}$		√	-
3	$e_{v_1,v_4} = 11$ $e_{v_5,v_4} = 8$ $e_{v_2,v_3} = 6$ $e_{v_2,v_4} = 2$	$e_{v_2,v_4} = 2$		√	$\{v_1, v_2, v_4, v_5\}$	$\{e_{v_1,v_5}, e_{v_1,v_2}, e_{v_2,v_4}\}$		√	-
4	$e_{v_1,v_4} = 11$ $e_{v_5,v_4} = 8$ $e_{v_2,v_3} = 6$ $e_{v_4,v_3} = 7$	$e_{v_2,v_3} = 6$		√	$\{v_1, v_2, v_3, v_4, v_5\}$	$\{e_{v_1,v_5}, e_{v_1,v_2}, e_{v_2,v_4}, e_{v_2,v_3}\}$	√		-

III. METODE PENELITIAN

3.1. Jenis dan Sumber Data

Data yang digunakan penulis dalam penelitian ini ialah data sekunder. Penelitian ini menggunakan data yang dibangkitkan secara acak menggunakan distribusi uniform dan nilainya berupa bilangan bulat (integer). Data tersebut jika representasi dalam graf merupakan data graf lengkap yang memiliki orde (titik) 10, 20, 30 sampai 100 dengan setiap orde memiliki 30 masalah berbeda. Data tersebut diambil dari penelitian yang dilakukan Wamiliana dkk. (2015).

3.2. Waktu dan Tempat Penelitian

Penelitian ini dilakukan pada tahun ajaran 2023/2024 di Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.

3.3. Tahapan Penelitian

Adapun langkah – langkah yang digunakan dalam penelitian ini sebagai berikut :

1. Identifikasi masalah

   Pada penelitian ini diawali dengan melakukan kajian pustaka tentang algoritma yang tepat dalam menyelesaikan masalah MRCST dengan menggunakan Algoritma Sollin dan Algoritma Prim.

2. Pencarian MST dan MRCST

   a. Pencarian menggunakan Algoritma Sollin

      Setelah membuat graf berbobot, dilakukan perhitungan dan analisis data menggunakan Algoritma Modifikasi Sollin untuk menentukan MST.

   b. Pencarian menggunakan Algoritma Prim

      Setelah membuat graf berbobot, dilakukan perhitungan dan analisis data menggunakan Algoritma Prim untuk menentukan MST.

3. Setelah diperoleh MST pada langkah sebelumnya maka lakukan pencarian MST dan MRCST dengan algoritmma yang telah dimodifikasi
   a. Pencarian menggunakan Algoritm Modifikasi Sollin
      Setelah membuat graf berbobot, dilakukan perhitungan dan analisis data menggunakan Algoritma Modifikasi Sollin untuk menentukan MST.
   b. Pencarian menggunakan Algoritma Modifikasi Prim
      Setelah membuat graf berbobot, dilakukan perhitungan dan analisis data menggunakan Algoritma Prim untuk menentukan MST.
4. Membandingkan hasil MRCST
   Dilakukan perbandingan hasil MRCST yang telah didapatkan dari kedua algoritma modifikasi tersebut dan memperhatikan banyaknya iterasi dari masing-masing algoritma untuk menentukan algoritma mana yang lebih efektif.
5. Penarikan kesimpulan
   Diambil kesimpulan setelah dilakukan analisis terhadap hasil yang diperoleh.

Adapun algoritma modifikasi Prim yang telah dibuat dibagi menjadi beberapa tahapan yaitu:

1. Selesaikan MST dengan algoritma Prim. Maka periksa apakah $Diam\ (G) \leq d$. Jika $Diam\ (G) \leq d$ maka selesai, jika tidak maka lanjut langkah 2.
2. Setelah diperoleh algoritma prim dan tidak memenuhi syarat maka pilih titik yang berderajat tinggi dengan jumlah bobot minimum ke semua titik sebagai titik primer (titik primer tidak boleh lebih dari 1 jika ada lebih dari 1 kandidat pilih salah satu).
3. Tentukan titik sekunder yaitu titik tetangga titik primer (yang bukan titik primer misal ada lebih dari 1 titik primer).
4. Setelah ditentukan titik primer dan sekunder. Periksa, jika semua titik merupakan titik sekunder maka selesai, jika tidak maka lakukan langkah berikutnya yaitu menentukan titik bermasalah (titik selain titik primer dan sekunder).
5. Periksa jarak titik bermasalah ke titik primer.

6. Kemudian pilih satu titik dengan bobot terkecil dan bandingkan dengan panjang lintasan sebelumnya ke titik primer (pada MST Prim).
   a. Jika panjang lintasan baru ke titik primer lebih kecil maka pindahkan.
      1. Periksa turunan titik bemasalah yang akan dipindahkan ke titik terpilih.
      2. Kemudian periksa diameter graf tersebut. Jika $Diam\,(G) \leq d$ maka selesai, jika tidak maka lanjut langkah 7.
   b. Jika tidak memenuhi poin a di atas maka pilih titik dengan bobot kecil berikutnya. Jika tetap tidak ada sampai semua data telah diperiksa maka lanjut langkah 7.

   Catatan untuk titik yang tidak terpilih atau gagal dipindahkan maka biarkan.
7. Buat titik masalah baru yaitu dengan memilih titik yang tidak terikat ke titik sekunder sebagai titik bermasalah baru.
8. Kemudian periksa panjang lintasan titik bermasalah ke titik primer melalui titik sekunder.
9. Kemudian pilih satu titik dengan bobot terkecil dan bandingkan dengan panjang lintasan sebelumnya ke titik primer melalui titik sekunder (pada MST Prim).
   a. Jika panjang lintasan baru ke titik primer melalui titik sekunder terpilih lebih kecil maka pindahkan.
      1. Periksa turunan titik bemasalah yang akan dipindahkan ke titik terpilih.
      2. Kemudian periksa diameter graf tersebut. Jika $Diam\,(G) \leq d$ maka selesai, jika tidak maka lanjut langkah 10.
   b. Jika tidak memenuhi poin a di atas maka pilih titik dengan bobot kecil berikutnya. Jika tetap tidak ada sampai semua data telah diperiksa maka lanjut langkah 10

   Catatan untuk titik yang tidak terpilih atau gagal dipindahkan maka biarkan.
10. Pilih titik masalah sebelumnya yang mempunyai selisih panjang lintasan terkecil ke titik primer melalui titik sekunder terpilih jika dibandingkan dengan panjang lintasan sebelumnya pada MST (usahakan juga titik sekunder yang terpilih tersebut dapat membuat panjang diameter graf berkurang).

Adapun algoritma modifikasi Sollin yang telah dibuat dibagi menjadi beberapa tahapan yaitu:

1. Selesaikan MST dengan algoritma Sollin. Maka periksa $Diam\ (G) \leq d$. Jika $Diam\ (G) \leq d$ maka selesai, jika tidak maka lanjut langkah 2.
2. Setelah diperoleh algoritma prim dan tidak memenuhi syarat maka pilih titik yang berderajat tinggi sebagai titik primer (titik primer boleh lebih dari 1).
3. Tentukan titik sekunder yaitu titik tetangga titik primer (yang bukan titik primer misal ada lebih dari 1 titik primer).
4. Setelah ditentukan titik primer dan sekunder. Periksa, jika semua titik merupakan titik sekunder maka selesai, jika tidak maka lakukan langkah berikutnya yaitu menentukan titik bermasalah (titik selain titik primer dan sekunder).
5. Periksa jarak titik bermasalah ke titik primer.
6. Kemudian pilih satu titik dengan bobot terkecil dan bandingkan dengan panjang lintasan sebelumnya ke titik primer (pada MST Sollin).
   a. Jika panjang lintasan baru ke titik primer lebih kecil maka pindahkan.
      1. Periksa turunan titik bemasalah yang akan dipindahkan ke titik terpilih.
      2. Kemudian periksa diameter graf tersebut. Jika $Diam\ (G) \leq d$ maka selesai, jika tidak maka lanjut langkah 7.
   b. Jika tidak memenuhi poin a di atas maka pilih titik dengan bobot kecil berikutnya. Jika tetap tidak ada sampai semua data telah diperiksa maka lanjut langkah 7.

   Catatan untuk titik yang tidak terpilih atau gagal dipindahkan maka biarkan.
7. Buat titik masalah baru yaitu dengan memilih titik yang tidak terikat ke titik sekunder sebagai titik bermasalah baru.
8. Kemudian periksa panjang lintasan titik bermasalah ke titik primer melalui titik sekunder.
9. Kemudian pilih satu titik dengan bobot terkecil dan bandingkan dengan panjang lintasan sebelumnya ke titik primer melalui titik sekunder (pada MST Sollin).
   a. Jika panjang lintasan baru ke titik primer melalui titik sekunder terpilih lebih kecil maka pindahkan.

1. Periksa turunan titik bemasalah yang akan dipindahkan ke titik terpilih.
2. Kemudian periksa diameter graf tersebut. Jika $Diam\ (G) \leq d$ maka selesai, jika tidak maka lanjut langkah 10.

b. Jika tidak memenuhi poin a di atas maka pilih titik dengan bobot kecil berikutnya. Jika tetap tidak ada sampai semua data telah diperiksa maka lanjut langkah 10

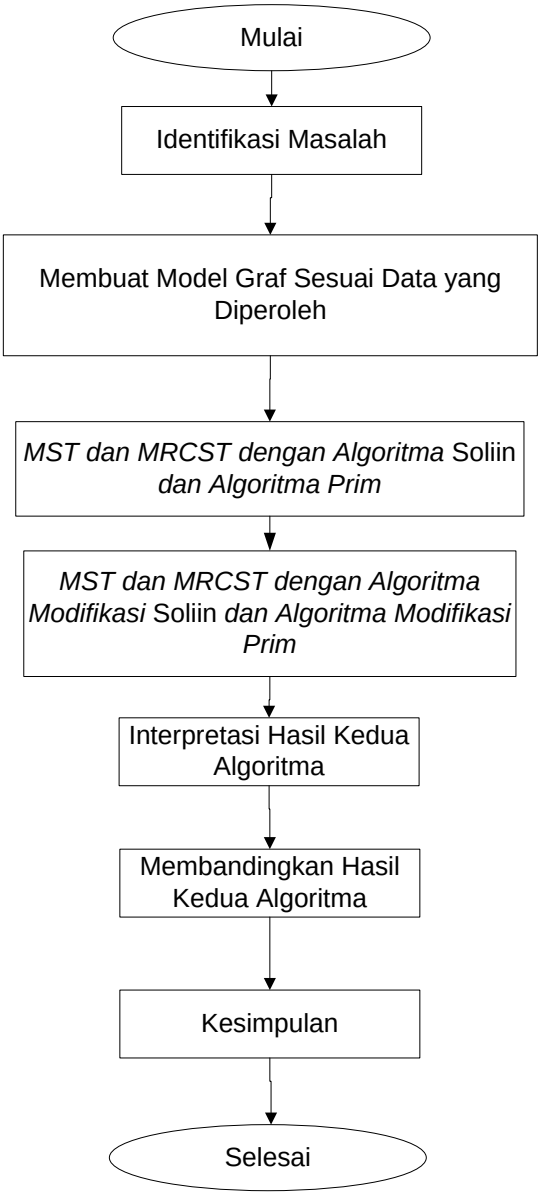
Catatan untuk titik yang tidak terpilih atau gagal dipindahkan maka biarkan.

10. Pilih titik masalah sebelumnya yang mempunyai selisih panjang lintasan terkecil ke titik primer melalui titik sekunder terpilih jika dibandingkan dengan panjang lintasan sebelumnya pada MST (usahakan juga titik sekunder yang terpilih tersebut dapat membuat panjang diameter graf berkurang).

Catatan:

1. Untuk nilai d atau batasan diameter yang digunakan yaitu $Diam(G) \leq \frac{n}{2}$ dan $Diam(G) \leq \frac{3n}{5}$ untuk $n \leq 30$; dan $Diam(G) \leq 15$ dan $Diam(G) \leq 18$ untuk $n > 30$
2. Pada kedua algoritma modifikasi tersebut saat melakukan perhitungan diameter, bobot pada sisi diabaikan.
3. Titik turunan yaitu titik tetangga titik bermasalah dan titik bermasalah lain (jika ada). Apakah mengikuti atau tidak. Dalam menentukan titik turunan mengikuti atau tidak, dikatakan mengikuti jika titik tersebut mempunyai panjang lintasan baru ke titik primer lebih kecil dari yang lama.

3.4. Alur Penelitian



Gambar 3.4.1. Alur Penelitian

V. SIMPULAN DAN SARAN

5.1. Simpulan

Berdasarkan hasil observasi dari algoritma modifikasi Prim dan algoritma modifikasi Sollin, maka diperoleh kesimpulan nilai MRCST untuk algoritma modifkasi Solin terbaik pada saat *Diam (G)* graf dibatasi kurang dari $\frac{n}{2}$ untuk $n \leq 30,$ dan *Diam (G)*$\leq$ 15 untuk $n \geq 40$ sedangkan algoritma modifkasi Prim terbaik pada saat *Diam (G)* graf dibatasi kurang dari $\frac{3n}{5}$ untuk $n \leq 30,$ dan *Diam (G)* ≤ 18 untuk $n \geq 40$. Dengan hasil MRCST algoritma modifikasi Prim lebih baik dari pada algoritma Sollin pada orde 10, 20, 30, dan 80.

Hasil penyelesaian MRCST pada pada *Diam (G)* $\leq \frac{n}{2}$ algoritma modifikasi Sollin 27 data membesar, 50 data mengecil, dan 223 data sama dibanding algoritma modifikasi Prim sedangkan pada *Diam (G)* $\leq \frac{3n}{5}$ ada algoritma modifikasi Sollin 21 data membesar, 25 data mengecil dan 254 data sama dibanding algoritma modifikasi Prim.

5.2. Saran

Penelitian tentang MRCST ini tentunya masih cukup terbuka dan masih banyak algoritma-algoritma yang belum diteliti untuk menyelesaikan kasus MRCST yang lebih baik.

DAFTAR PUSTAKA

Anggraeni, W. 2015. Aplikasi Algoritma *Sollin* Dalam Pencarian Pohon Perentang Minimum Provinsi Jawa Tengah. *Jurnal Faktor Exacta*, **8**(4), pp. 381 – 391.

Bondy, J.A, & Murty, U.S.R. 1976. Graph Theory With Applications. London: MacMillan Press.

Chartrand, G. & Lesniak, L. 1986. Graph and Digraph 2 nd Edition. California: Wadsworth. Inc.

Chen, K., Hsieh, Y. E., & Lu, P. J. 2013. Parallel Approximation Algorithms for Minimum Routing Cost Spanning Tree. *International Journal of Computational Science and Engineering.* **8**(4). pp. 336–348

Deo, N. 1989. *Graph Theory with Application to Engineering and Computer Science*. Prentice-Hall of India Private Limited, New Delhi.

Hieu, N. M., Quoc, P., & Nghia, N. D. 2011. An Approach of Ant Algorithm for Solving Minimum Routing Cost Span- ning Tree Problem. *In Proceedings of the 2nd Symposium on Information and Communication Technology.* Pp.5–10

Ismail, T. & Setiady, T. 2014. Media Pembelajaran Strategi Algoritma Pada Pokok Bahasan Pohon Merentang Minimum Dan Pencarian Lintasan Terpendek. *Jurnal Sarjana Teknik Informatika*. **2**(2) pp. 1423 – 1430.

Julstrom, B. A. 2001. The Blob Code: A Better String Cod- ing of Spanning Trees for Evolutionary Search. *In Genetic and Evolutionary Computation Conference Workshop Program*. Morgan Kaufmann San Francisco. Pp. 256–261

Julstrom, B. A. 2005. The Blob Code is Competitive with Edge-Sets in Genetic Algorithms for the Minimum Routing Cost Spanning Tree Problem. *In Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation. Pp. 585-590.*

Lin, C. M., Te Tsai, Y., & Tang, C. Y. (2006). Balancing Minimum Spanning Trees and Multiple-Source Minimum Routing Cost Spanning Trees on Metric Graphs. *Information Processing Letters*. **99**(2). pp. 64–67

Munir, R. 2014. *Matematika Diskrit*. Bandung: Informatika.

Raidl, G. R. & Julstrom, B. A. 2003. Edge Sets: An Effective Evolutionary Coding of Spanning Trees. *IEEE Transactions on Evolutionary Computation*. **7**(3). Pp. 225–239

Ramadhan, A. F. 2017. Aplikasi Algoritma Prim Dalam Penentuan Pohon Merentang Minimum Untuk Jaringan Pipa Pdam Kota Tangerang. *Jurnal Ilmiah*. **2(**1). pp. 30 – 38.

Roman, S. 1989. An Introduction To Discrete Mathematics. Second Edition. New York: McGraw-Hill, Inc.

Sari, R. P., Wamiliana, A. Junaidi, and W. Susanty. 2022. The Diameter and Maximum Link of the Minimum Rout- ing Cost Spanning Tree Problem. *Science and Technology Indonesia*, **7**(4). Pp. 481–485

Sattari, S. & F. Didehvar. 2013. Variable Neighborhood Search Approach for the Minimum Routing Cost Spanning Tree Problem. *International Journal Operations Research*. **10**(4). pp. 153–160

Sattari, S. and F. Didehvar. 2015. A Metaheuristic Algorithm for the Minimum Routing Cost Spanning Tree Problem. *Iranian Journal of Operations Research*. **6**(1). pp. 65–78

Singh, A. 2008. A New Heuristic for the Minimum Routing Cost Spanning Tree Problem. *IEEE International Conference on Information Technology*. Pp. 9–13

Singh, A. & S. Sundar. 2011. An Artificial Bee Colony Algorithm for the Minimum Routing Cost Spanning Tree Problem. *Soft Computing*. **15**. Pp 2489–2499

Tan, Q. P. 2012a. A Heuristic Approach for Solving Minimum Routing Cost Spanning Tree Problem. *International Journal of Machine Learning and Computing*. **2**(4). pp. 406

Tan, Q. P. 2012b. A Genetic Approach for Solving Minimum Routing Cost Spanning Tree Problem. *International Journal of Machine Learning and Computing.* **2**(4). pp.410

Tan, Q. P. & N. N. Due. 2013. An Experimental Study of Minimum Routing Cost Spanning Tree Algorithms. *IEEE International Conference on Soft Computing and Pattern Recognition (SoCPaR).* pp.158–165

Wamiliana, Elfaki, F.A.M., Usman, M. & Azram, M. 2015. Some greedy based algorithms for multi periods degree constrained minimum spanning tree problem. *ARPN Journal of Engineering and Applied Science*. **10**(21): 10147-10152.

Wamiliana. 2022. *Minimum Spanning Tree dan Desain Jaringan*. Bandarlampung: Pustaka Media.

Wilson, R. J & Watkins, J. J. 1990. Graph An Introductory Approach a First Course In Discrete Mathematics. Canada: John Whiley and Sons, Inc.

Wolf, S. and P. Merz. 2010. Efficient Cycle Search for the Minimum Routing Cost Spanning Tree Problem. *Springer European Conference on Evolutionary Computation in Combinatorial Optimization*. pp. 276–287

Wu, B. Y. 2002. A Polynomial Time Approximation Scheme for the Two-Source Minimum Routing Cost Spanning Trees. *Journal of Algorithms*. **44**(2). pp.359–378

Wu, Bang Ye., & Chao, Kun-Mao. 2004. Spanning Trees and Optimization Problems. *Chapman&Hall/CRC*. pp. 13–139.