STRING (Satuan Tulisan Riset dan Inovasi Teknologi) p-ISSN: 2527 - 9661
Vol. 4 No. 3 April 2020 e-ISSN: 2549 - 2837

PENDEKATAN MATRIKS KETETANGGAAN BERBOBOT UNTUK SOLUSI *MINIMUM SPANNING TREE (MST)*

Tri Yani Akhirina[1], Thomas Afrizal[2]
Informatika, Universitas Indraprasta PGRI[1,2]
triyani.akhirina@unindra.ac.id

Submitted February 21, 2020; Revised March 22, 2020; Accepted March 25, 2020

Abstrak

Minimum Spanning Tree (MST) atau sering juga disebut *Minimum Weighting Spanning Tree (MWST)* adalah sebuah algoritma pencarian jalur atau sisi *(edge)* yang menghubungkan semua simpul (*vertex*) didalam graf saling terhubung dan tidak membentuk sirkuit dengan bobot minimum. Algoritma klasik yang digunakan untuk menyelesaikan masalah MST adalah algoritma prim dan kruskal. Permasalahan pohon merentang minimum merupakan permasalahan yang berkaitan dengan optimalisasi dalam menemukan sisi *(edge)* berbobot minimum yang dapat menghubungkan semua simpul *(Vertex)*. (MST) ini banyak digunakan dalam keilmuan komputer seperti menentukan akses point, membangun jaringan network dan banyak lagi. Tujuan penelitian ini adalah menemukan solusi baru yang dapat memberikan alternative solusi *MST* selain algoritma klasik seperti prim dan kruskal. Penelitian ini bersifat eksperimental dimana pendekatan yang dilakukan adalah menggunakan matrik ketetanggaan berbobot, dari bobot yang ada diambil sisi yang paling minimum di tiap pasangan matriks untuk menghasilkan *minimum spanning tree*. Solusi baru ini dapat menjadi alternatif dalam menyelesaikan permasalahan *minimum spanning tree*.

Kata Kunci : Minimum Spanning Tree, graf, Matriks Ketetanggan berbobot, Algoritma Prim, algoritma Kruskal

Abstract

Minimum Spanning Tree (MST) or often called Minimum Weighting Spanning Tree (MWST) is a path or edge search algorithm that connects all vertices in a connected graph and does not form a circuit with a minimum weight. The classic algorithm used to solve MST problems is the Prim's and Kruskal's algorithms. The problem of minimum spanning tree is a problem related to optimization in finding the minimum weighted edge that can connect all vertices). MST is widely used in computer science such as to determine access points, build networks and many more. The purpose of this reserarch is to find new solutions that can provide alternative MST solutions besides classical algorithms such as Prim and Kruskal. This experimental research uses a weighted adjacency matrix approach, with weight taken from the minimum side in each pair of matrix to produce a minimum spanning tree. This new solution can be an alternative in solving the minimum spanning tree problems.

Keywords : *Minimum Spanning Tree, graph, Weighted Adjacency Matrix, Prim's Algorithm, Kruskal's algorithm*

1. PENDAHULUAN

Minimum Spanning Tree (MST) merupakan salah satu cabang matematika terapan untuk menyelesaikan permasalahan optimasi diskrit [1]. Algoritma klasik yang sudah sangat sering digunakan adalah algoritma kruskal dan algoritma prim [2],[3],[4] untuk mencari jalur yang menghubungkan seluruh simpul dengan bobot terkecil.

Spanning tree adalah sebuah *tree* yang terbentuk dari turunan sebuah graf terhubung *(acyclic)* dimana semua simpulnya saling terhubung [5],[6], seluruh simpulnya terhubung dalam jaringan *tree* [4], dan tidak membentuk sirkuit (*cyclic)*[1]. *Minimum spanning tree* adalah

sebuah spanning tree berbobot minimum[7].

Beberapa penelitian telah dilakukan untuk mencari algoritma baru dalam menyelesaikan masalah MST ini. Pendekatan algoritma *divide and qoncuer* dilakukan dengan mencari sisi ketetanggan terdekat dengan jarak/bobot minimum sebagai dasar mengkelompokkan[8]. Pendekatan algoritma greedy juga telah dilakukan untuk mendapatkan bobot optimal MST[9]. Selain itu pendekatan *Genetic Algorithm* juga dilakukan untuk melihat hasil data huristik kedalam metahuristik dalam mencari solusi optimal [10]. Pendekatan dengan memodifikasi terhadap kedua metode klasik yaitu kruskal dan prim juga telah dilakukan [3].

Pendekatan yang berkaitan dengan menggunakan matriks ketanggaan juga telah dilakukan. Penggunaan matriks ketetanggan dilakukan untuk mencari biaya terkecil terkait jarak dan bobot kemudian mencari tetangga terdekat dari simpul yang bersisian dengan bobot terkecil tersebut hingga terbentuk MST [5]. Pada penelitian lain matriks ketetanggan juga dilakukan untuk pencarian MST, yang membedakan dari penelitian yang sebelumnya adalah pemilihan simpul awal, pada penelitian ini simpul awal di mulai dari 0 dan dilanjutkan berdasarkan bobot matriks terkecil untuk membangun jaringan MSTnya [4]. Pendekatan yang dilakukan oleh keduanya dapat menemukan solusi untuk permasalahan MST. Pada penelitian ini, mencoba melakukan pendekatan yang sama menggunakan matrik ketetanggaan berbobot dengan mengurutkan bobot minimum setiap baris martriks $A_{[i,j]}$, dari atas sampai dengan bawah, hasil bobot minimum yang di peroleh digunakan untuk membangun jaringan MST.

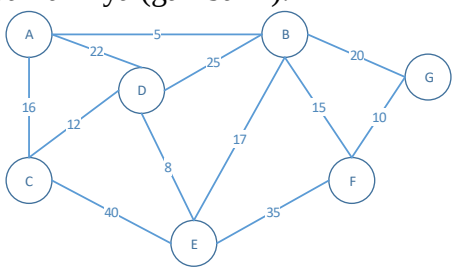
Tujuan dari penelitian ini adalah untuk mencari solusi baru dalam menyelesaikan permasalahan minimum spanning tree berdasarkan pendekatan matriks ketetanggaan berbobot. Diharapkan penelitian ini memberikan kontribusi terhadap keilmuan, dimana algoritma yang diajukan dapat menjadi alternatif dalam menyelesaikan masalah-masalah minimum spanning tree.

2. METODE PENELITIAN

Metode yang digunakan dalam penelitian adalah menggunakan metode ekperimental, dimana diusulkan suatu algoritma baru untuk mencari MST dengan pendekatan matriks ketetanggan berbobot. Algoritma tersebut dibandingkan dengan hasil pencarian MST menggunakan dua algoritma klasik yaitu kruskal dan prim.
Jenis metode eksperimental yang digunakan adalah metode ekperimental dengan kelompok pembanding untuk mendapatkan hasil yang sesuai. Pada penelitian ini, pembandingan yang dilakukan adalah membandingakan hasil pencarian MST algoritma yang di usulkan dengan dua algoritma klasik sebagai kontorlnya.
Tahapan yang dilakukan adalah melakukan studi literatur terkait dengan MST, kemudian melakukan beberapa percobaan berbagai jenis graf untuk diselesaikan dengan algoritma baru, hasil pengujian dilakukan perbandingan dengan hasil pencarian MST dengan algoritma klasik, sampai menghasilkan MST yang optimum.

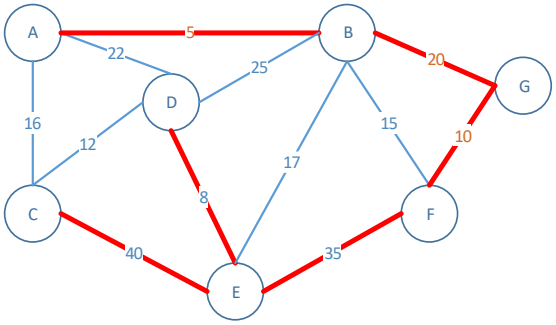
Definisi

Graf Berbobot : graf G yang setiap simpulnya memiliki bobot yang menghubungkan antara satu simpul ke simpul lainnya (gambar 1).



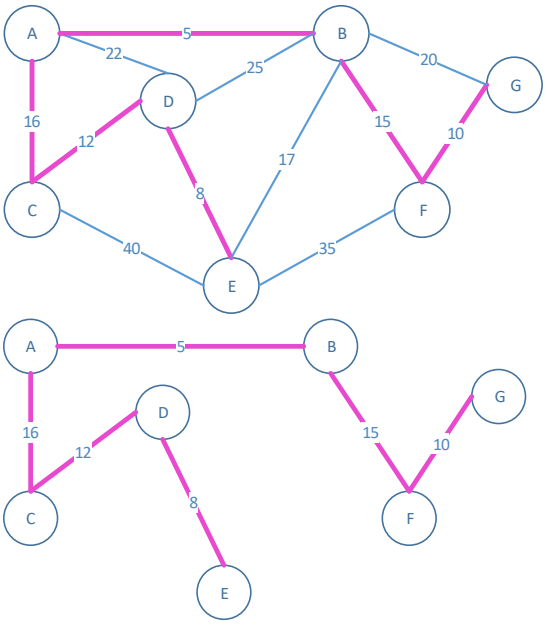
Gambar 1. Graf Berbobot G_1

Spanning tree : sebuah tree T yang merupakan sub graf G dimana seluruh simpulnya terhubung tanpa mengandung sirkuit (gambar 2).



Gambar 2. Spanning Tree T sub graf G_1

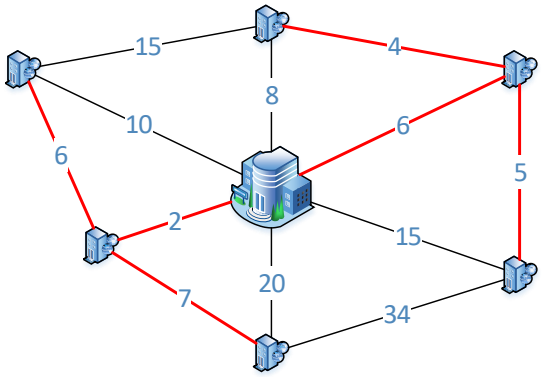
Minimum Spanning Tree (MST) : sebuah tree T yang merupakan sub graf G dimana seluruh simpulnya terhubung tanpa mengandung sirkuit dengan jumlah bobot minimum (gambar 3).



Gambar 3. MST T sub graf G

Permasalah terkait

MST merupakan salah satu algoritma yang digunakan untuk menyelesaikan permasalahan optimasi diskrit, mencari jarak terdekat yang dapat menghubungkan semua simpul, dalam penerapan ilmu komputer misalnya seperti menentukan akses poin, membangun jaringan yang optimal, dan lain sebagainya. Seperti contoh pada gambar 4.



Gambar 4. Contoh Membangun Jaringan antar Gedung

Pada gambar 4 adalah contoh dimana yang diinginkan oleh pihak manajemen semua cabang saling terhubung dalam suatu jaringan dimana pusat transmisinya berada di kantor pusat. Bila dibangun secera keseluruhan membutuhkan biaya yang sangat besar, untuk mengoptimalkan jaringan tersebut maka dibuatlah MST untuk membangun jaringannya agar menghemat biaya.

Algoritma Yang diusulkan

Diberikan, G = (V, E) graf berbobot tak berarah dengan n simpul, dimana V adalah himpunan simpul dan E adalah himpunan sisi dan W adalah himpunan bobot yang terkait dari sisi yang menghubungkan antar simpul dalam grafik. Dimana sisi yang berdekatan

w_{ij} = bobot yang terkait dengan sisi e_{ij}.

Matriks M bobot dari grafik G dibangun sebagai berikut:

If ada sisi antara simpul v_i
ke v_j di G lalu Set,
$M_{[i, j]} = W_{ij}$
else
Set, $M_{[i, j]} = 0$

Input : Buat matriks Berbobot $M= M_{[i,j]\ nxn}$ dimana :

n = jumlah simpul pada Graf (S);
i = baris matriks dari simpul v_i;

j = kolom matriks dari simpul v_j;
$w_{[ij]}$= bobot sisi e_{ij}
e_{ij}= jumlah sisi minimum terpilih;

Output : Optimum *Minimum Spanning Tree* T dari G

Langkah 1 : Start

Langkah 2 : Cari semua bobot minimum tiap baris $W_{[ij]}$ pada matriks $M_{[i,j]} \neq 0$.

Langkah 3 : Masukkan hasil pencarian bobot minimum kedalam Graf T. Update e_{ij}=Count ($e_{[ij]min}$), Periksa e_{ij} = (n-1) dan tidak mengandung sirkuit. Bila mengandung sirkuit tolak $W_{[ij]}$ bobot terbesar $W_{[ij]}$ yang terhubung pada simpul V_i dan v_j yang membentuk sirkuit.

Langkah 4 : Update bobot $W_{[ij]\ min}$ pada matriks terpilih dengan 0.

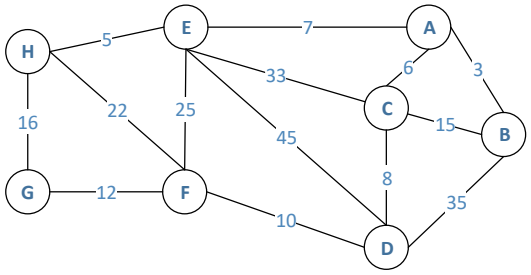
Langkah 5 : Bila belum terhubung $e_{ij=}$ (n-1) cari sisi $W_{[ij]}$ paling minimum sejumlah e_{ij} – n. Masukkan kedalam T, periksa apakah ada sirkuit, bila tidak maka langkah 7, bila belum ulangi langkah 5 dan 6.

Langkah 6 : Bila simpul sudah terhubung hitung bobot MST = Σ min (w_{ij}) maka end.

3. HASIL DAN PEMBAHASAN

Contoh kasus:

Diberikan sebuah graf sebagai berikut:
V: {A,B,C,D,E,F,G,H}
E : {(A,B),(A,C),(A,E),(B,C),(B,D),(C,D),(C,E),(D,E),(D,F),(E,F),(E,H),(F,G)(F,H),(G,H)}
Dengan Bobot W pada gambar 5.



Gambar 5. Graf Berbobot G

Ditanya : Tentukan MST dari graf diatas (gambar 5)

Penyelesaian MST dengan Pendekatan Kruskal

Berikut adalah tahapan penyelesaian MST dengan algoritma Kruskal

Step 1 : Urutkan bobot masing-masing sisi secara menaik, dari terkecil sampai dengan terbesar (tabel 1).

Tabel 1. Urutan bobot sisi

Sisi	Bobot
(A,B)	3
(E,H)	5
(A,C)	6
(A,E)	7
(C,D)	8
(D,F)	10
(F,G)	12
(C,B)	15
(H,G)	16
(F,H)	22
(E,F)	25
(E,C)	33
(B,D)	35
(D,E)	45

Step 2 : Masukkan seluruh simpul kedalam T.

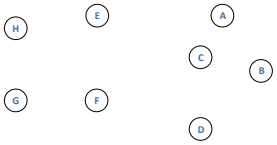
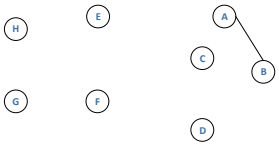
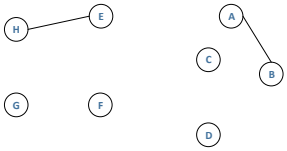
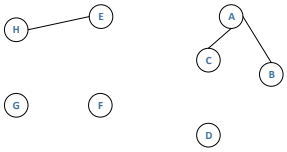
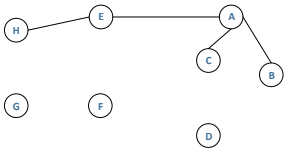
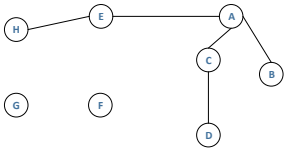
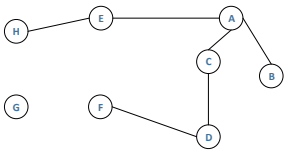
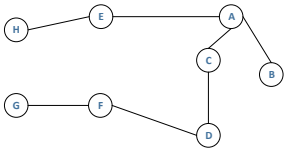
Step 3 : Masukkan masing-masing simpul dari yang paling minimum bobotnya kedalam T.

Step 4 : Cek apakah terbentuk sirkuit, jika iya maka tolak. Jika tidak masukkan sisi berikutnya.

Step 5 : Lakukan berulang sampai dengan seluruh simpul terhubung.

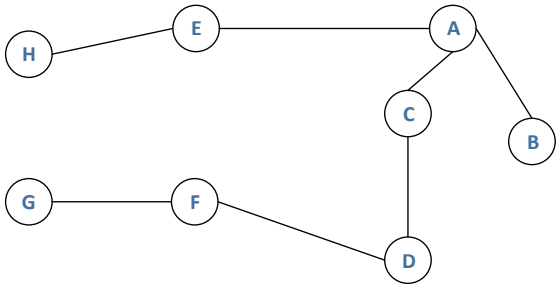
Penerapan algoritma kruskal dapat dilihat pada tabel 2 dibawah ini:

Tabel 2. Penerapan Algoritma Kruskal

No	Sisi = Bobot	Aksi	Tree (T)
0	-	-	
1	(A,B) = 3	T	
2	(E,H)=5	T	
3	(A,C)=6	T	
4	(A,E)=7	T	
5	(C,D)=8	T	
6	(D,F)=10	T	
7	(F,G)=12	T	
8	(C,B)=15	F	Tidak dieksekusi karena simpul sudah saling terhubung
9	(H,G)=16	F	
10	(F,H)=22	F	
11	(E,F)=25	F	
12	(E,C)=33	F	
13	(B,D)=35	F	
14	(D,E)=45	F	

Step 6 : Menghitung total bobot MST

Total MST = 3+5+6+7+8+10+12 = **51**
Hasil MST yang terbentuk dengan menggunakan algoritma Kruskal dapat dilihat pada gambar dibawah ini.



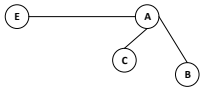
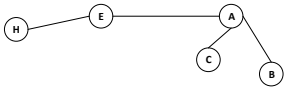
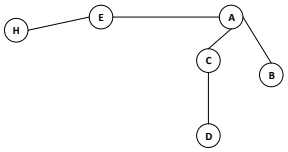
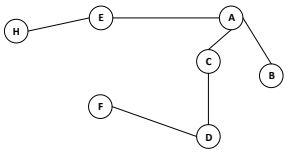
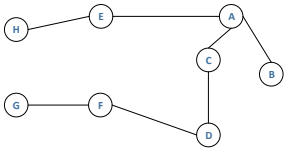
Gambar 6. MST dengan Algoritma Kruskal

Penyelesaian MST dengan Algoritma Prim

Berikut ini adalah langkah-langkah menggunakan algoritma prim:
Step 1 : Pilih bobot paling minimum dari seluruh sisi terhubung dengan simpul.
Step 2 : Masukkan sisi kedalam T.
Step 3 : Mencari sisi yang bersisian dengan simpul dalam T yang berbobot minimum.
Step 4 : Masukan di dalam T.
Step 5 : Cek apakah terbentuk sirkuit, jika ya maka tolak jika tidak maka ulangi step 2 s.d 5 sampai dengan seluruh simpul terhubung,
Step 6 : Hitung total bobot MST
Step 7 : end.
Untuk Ilustrasi mencari MST dengan algoritma prim dapat dilihat pada tabel 3 berikut:

p-ISSN: 2527 - 9661
e-ISSN: 2549 - 2837

Tabel 3. Penerapan algoritma Prim

No	Sisi = Bobot	Aksi	Tree (T)
1	(A,B) = 3	(A,C)	
2	**(A,C)=6** (A,E)=7 (B,C)=15 (B,D)=35	(A,C)	
3	**(A,E)=7** (B,C)=15 (B,D)=35 (C,D)=8 (C,E)=33	(A,E)	
4	(B,C)=15 (B,D)=35 (C,D)=8 (C,E)=33 (D,E)=45 (E,F)=25 **(E,H)=5**	(E,H)	
5	(B,C)=15 (B,D)=35 **(C,D)=8** (C,E)=33 (D,E)=45 (E,F)=25 (H,G)=16	(C,D)	
6	(B,C)=15 (B,D)=35 (C,E)=33 (D,E)=45 (E,F)=25 (H,G)=16 **(D,F)=10**	(D,F)	
7	(B,C)=15 (B,D)=35 (C,E)=33 (D,E)=45 (E,F)=25 (H,G)=16 **(F,G)=12** (F,H)=22	(F,G)	

Total bobot MST = 51 dan Tree nya sama dengan hasil algoritma kruskal (gambar 6).

Penyelesaian MST dengan Pendekatan Matriks Ketetanggaan Berbobot.

Langkah-langkahnya :

Step 1: Bentuk matrik ketetanggan berbobot (tabel 4) dari graf G (gambar 5)

Tabel 4. Matriks Ketetanggaan Berbobot Graf G

	A	B	C	D	E	F	G	H
A	0	3	6	0	7	0	0	0
B	3	0	15	35	0	0	0	0
C	6	15	0	8	33	0	0	0
D	0	35	8	0	45	10	0	0
E	7	0	0	0	0	25	0	5
F	0	0	0	0	25	0	12	22
G	0	0	0	0	0	12	0	16
H	0	0	0	0	5	0	16	0

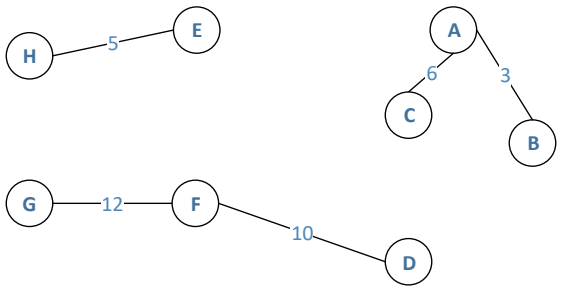
Step 2 : Cari bobot minimum masing masing baris matriks M_{ij} (tabel 5)

Tabel 5. Bobot Minimum per Baris Matriks $M_{[ij]}$

	A	B	C	D	E	F	G	H	W_{ij}
A	0	**3**	**6**	0	7	0	0	0	**3**
B	**3**	0	15	35	0	0	0	0	**3**
C	**6**	15	0	8	33	0	0	0	**6**
D	0	35	**8**	0	45	10	0	0	**8**
E	7	0	0	0	0	25	0	**5**	**5**
F	0	0	0	**10**	25	0	12	22	**10**
G	0	0	0	0	0	**12**	0	16	**12**
H	0	0	0	0	**5**	0	16	0	**5**

Diperoleh $e_{[i,j]min}$ = *{(A,B)=3,(A,C)= 6, (C,D)= 8, (E,H)=5, (F,D)=10,(F,G)=12}*

Step 3 : masukkan $e_{[i,j]\ min}$ kedalam T (gambar 6). Update e_{ij}=count($e_{min[i,j]}$)



Gambar 7. Minimum Spanning Tree Iterasi ke 1

Step 4 : Update $W_{[ij]}$ minimum terpilih dengan nilai 0 (tabel 6).

Tabel 6. $W_{[ij]}$ Minimum Terpilih Bobot Diganti dengan 0

	A	B	C	D	E	F	G	H
A	0	**0**	**0**	0	7	0	0	0
B	**0**	0	15	35	0	0	0	0
C	**0**	15	0	8	33	0	0	0
D	0	35	**0**	0	45	10	0	0
E	7	0	0	0	0	25	0	**0**
F	0	0	0	0	25	0	**0**	22
G	0	0	0	0	0	**0**	0	16
H	0	0	0	0	**0**	0	16	0

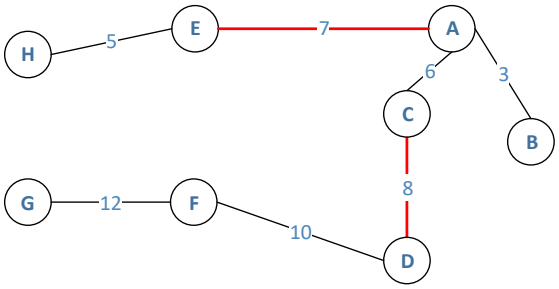
Step 5 : Memeriksa, apakah e_{ij} = (n-1) sudah terpenuhi (semua simpul sudah terhubung) jika sudah maka end, jika belum maka masuk langkah selanjutnya. Dalam kasus ini belum terhubung e_{ij}=n-1 dalam hal ini e_{ij}= 8-1 ➔ 5 ≠ 7 maka lakukan langkah berikutnya.

Step 6 : Mencari bobot paling minimum $W_{[ij]}$ dari seluruh matriks sejumlah n-e_{ij} ➔ 5-6 = 2 simpan dalam $e_{[ij]min}$

Tabel 7. Mencari Bobot Minimum untuk Simpul Belum Terhubung

	A	B	C	D	E	F	G	H
A	0	0	0	0	**7**	0	0	0
B	0	0	15	35	0	0	0	0
C	0	15	0	**8**	33	0	0	0
D	0	35	0	0	45	10	0	0
E	7	0	0	0	0	25	0	0
F	0	0	0	0	25	0	0	22
G	0	0	0	0	0	0	0	16
H	0	0	0	0	0	0	16	0

Himpunan $e_{[ij]min}$ = {(A,C)=7, (D,C) = 8} tabel 4. Masukkan kedalam T, cek apakah membentuk sirkuit, jika tidak maka end.



Gambar 7. MST Terpilih dengan e_{ij} = n-1

Step 7: update e_{ij}, periksa e_{ij} = (n-1). Dalam kasus diatas sudah terpenuhi, maka hitung total MST = 3+6+10+12+5+8+7 = **51.**

Step 8 : end

Maka kita mendapatkan Total MST pada contoh Graf G (Gambar 5), dengan jalur (*path*) yang terbentuk pada himpunan sisi MST = {(A,B),(A,C),(C,D),(A,E),(E,H),(D,F),(F,G)}.

4. SIMPULAN

Penyelesaian masalah *Minimum Spanning Tree* menggunakan pendekatan matrik ketetanggan berbobot menghasilkan MST yang sama dengan penyelesaian mengggunakan algoritma Kruskal dan Prim. Berdasarkan hal tersebut, maka algoritma yang diusulkan dalam penelitian ini dapat digunakan sebagai alternatif penyelesaian permasalahan *Minimum Spanning Tree*.

DAFTAR PUSTAKA

[1] P. Jayawant and K. Glavin, "Minimum spanning trees," *Involv. a J. Math.*, vol. 2, no. 4, pp. 439–450, 2009.

[2] A. Rahmawati and Mulyono, "Minimum Spanning Tree Pada Jaringan Pendistribusian," *UNNES J. Math.*, vol. 4, no. 2, 2015.

[3] L. Caccetta, "The Modified CW1 Algorithm For The Degree Restricted Minimum Spanning Tree Problem," vol. 1, no. 2, 2013.

[4] M. Mismar, "A New Quick Algorithm for Finding The Minimal Spanning Tree A New Quick Algorithm for Finding the Minimal Spanning Tree," no. March, 2018.

[5] D. Vijayalakshmir and R. Kalaivani, "Minimum Cost Spanning Tree using Matrix Algorithm," *Int. J. Sci. Res. Publ.*, vol. 4, no. 9, pp. 1–5, 2014.

[6] A. Y. Khedr and H. M. Bahig, "Debugging tool to learn algorithms: A case study minimal spanning tree," *Int. J. Emerg. Technol. Learn.*, vol. 12, no. 4, pp. 90–100, 2017.

[7] T. Yamada, S. Kataoka, and K. Watanabe, "Listing all the minimum spanning trees in an undirected graph," *Int. J. Comput. Math.*, vol. 87, no. 14, pp. 3175–3185, 2010.

[8] X. Wang, X. Wang, and D. M. Wilkes, "A divide-and-conquer approach for minimum spanning tree-based clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 7, pp. 945–958, 2009.

[9] P. Biswas, M. Goel, H. Negi, and M. Datta, "An Efficient Greedy Minimum Spanning Tree Algorithm Based on Vertex Associative Cycle Detection Method," *Procedia Comput. Sci.*, vol. 92, pp. 513–519, 2016.

[10] C. Contreras-Bolton, C. Rey, S. Ramos-Cossio, C. Rodríguez, F. Gatica, and V. Parada, "Automatically produced algorithms for the generalized minimum spanning tree problem," *Sci. Program.*, vol. 2016, 2016.