Pemodelan dan Penentuan Rute Evakuasi Menggunakan Graf dan Algoritma Dijkstra Pada Bangunan

Naufal Zhafran Latif and 13517095[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13517095@std.stei.itb.ac.id

***Abstract*— Rute evakuasi merupakan salah satu bagian penting dari *Disaster Management Plan*. Rute evakuasi yang efisien dapat memperkecil dampak buruk dari suatu bencana alam. Salah satu cara untuk memodelkan rute evakuasi menggunakan graf. Rute evakuasi akan ditentukan menggunakan Algoritma Dijkstra yang diterapkan pada graf.**

***Keywords*—Disaster Management Plan,Graf,Algoritma Dijkstras.**

I. PENDAHULUAN

Bencana alam adalah kejadian alam yang memiliki dampak besar bagi lingkungan sekitar dan populasi manuasia. Bencana alam tidak dapat kita hindari di bumi ini. Waktu dan tempat kejadian bencana alam sangat acak dan sulit untuk diprediksi. Bencana alam memiliki dampak merusak lingkungan sekitar. Bencana alam bisa berupa banjir, gempa bumi, tanah longsor hingga tsunami. Bencana alam biasanya terjadi secara alami, akan tetapi beberapa bencana alam terjadi karena ulah manusia seperti banjir dan tanah longsor.

Dampak nya yang sangat besar terhadap populasi manusia membuat bencana alam harus diperhatikan lebih. Perlu adanya sebuah sistem untuk mengurangi dampak bencana tersebut. Salah satu bagian dari sistem tersebut adalah rute evakuasi. Rute evakuasi ini menjadi penting karena ketika bencana alam terjadi, manusia pasti perlu untuk pergi ketempat yang lebih aman dengan menggunakan jalur yang cepat dan efisien. Jalur tersebut harus dirancang dengan baik sehingga mempercepat proses evakuasi pada saat bencana berlangsung. Apalagi pada sebuah bangunan yang mobilitas untuk melakukan evakuasi sangat terbatas. Perlu direncanakan dengan baik jalur untuk

Graf bisa menjadi salah satu cara untuk memodelkan jalur evakuasi sebuah bangunan. Dengan mengetahui hubungan antara ruang dan waktu tempuh untuk berpindah ruang, graf dapat dibentuk untuk merepresentasikan denah bangunan tersebut. Dengan menggunakan algoritma *Dijkstra*, jalur tercepat yang dilalui dari sebuah simpul ke simpul yang lain dapat dicari. Sehingga algoritma ini cocok untuk digunakan dalam menentukan rute evakuasi yang tercepat.

II. GRAF

Graf adalah sebuah struktur data yang terdiri dari himpunan simpul dan sisi yang menghubungkan antar simpul. Graf ada yang berhingga ada juga yang tak hingga. Graf yang umum digunakan dan dapat digunakan dalam permasalahan ini adalah graf berhingga.

$$G = (V, E)$$

V = Simpul
E = Sisi

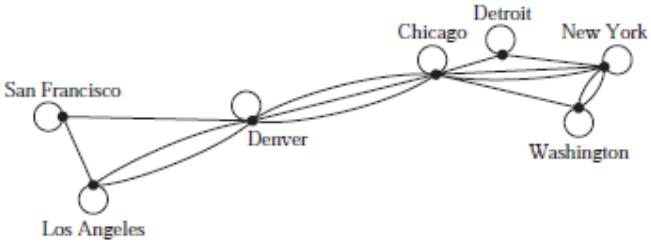
Permasalahan diberbagai bidang dapat di modelkan dan diselesaikan menggunakan graf. Graf dapat merepresentasikan dan memodelkan banyak permasalahan yang ada di dunia ini. Permasalahan yang dapat dimodelkan dengan graf seperti peta. Peta memiliki kota sebagai simpulnya. Hubungan antar kota atau jalanan antar kota direpresentasikan dengan sisi. Selain itu masih banyak lagi masalah yang dapat dimodelkan seperti hubungan antar manusia, jalur pada jaringan komputer, susunan organisasi dan lain-lain.

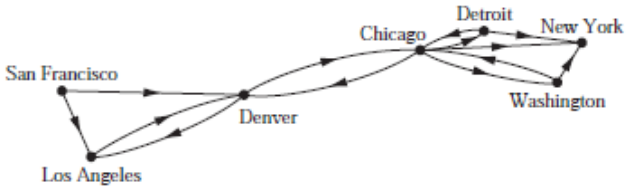
A. Jenis-Jenis Graf

Graf terdiri dari berbagai jenis. Jenis-jenis ini menyesuaikan dari penggunaan graf tersebut. Setiap jenis graf memiliki fungsinya masing-masing. Jenis-jenis graf tersebut dapat di gabung, tidak berpatokan hanya pada satu jenis. Penggabungan ini pun didasari oleh masalah yang ingin dimodelkan.

1. Berdasarkan Orientasi Arah

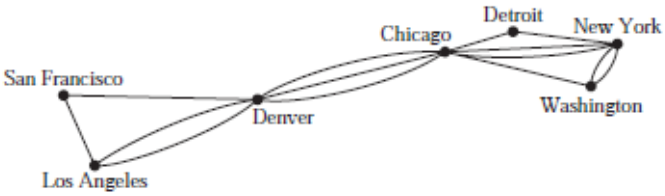
Graf berdasarkan orientasinya dibagi menjadi 2 jenis, graf berarah dan graf tidak berarah. Graf berarah memiliki orientasi arah pada sisinya. Graf tidak berarah tidak memiliki orientasi arah pada sisinya. Graf berarah biasanya digunakan untuk merepresentasikan suatu hubungan yang hanya searah pada mayoritas simpulnya. Sedangkan untuk graf tidak berarah digunakan untuk merepresentasikan hubungan antar simpul yang dua arah pada mayoritas simpulnya.

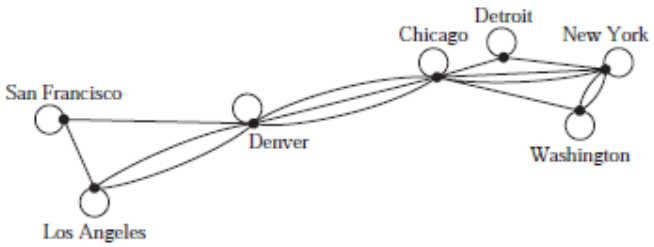




2. Berdasarkan Kompleksitas Graf

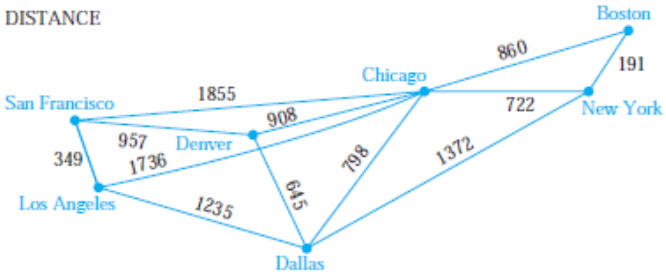
Graf berdasarkan kompleksitasnya dibagi menjadi 2 jenis, graf sederhana dan graf tak-sederhanan.Pada graf sederhana tidak memiliki sisi ganda ataupun cincin sedangkan pada graf tak-sederhana memiliki sisi ganda atau cincin. Adanya cincin membuat sebuah simpul memiliki jalur untuk kembali ke simpul itu sendiri. Salah satu penggunaan cincin adalah pada automata yang dapat tidak merubah state nya ketika sebuah input masuk. Sisi banyak antar dua buah simpul dapat merepresentasikan ada banyak jalan yang dapat dilalui untuk berpindah dari satu simpul ke simpul yang lain.





3. Graf Berbobot

Graf berbobot adalah graf dengan sisi yang memiliki bobot. Bobot tersebut dapat representasikan jarak antar simpulnya seperti pada sebuah peta. Selain itu waktu tempuh juga dapat direpresentasikan dengan graf berbobot.



B. Representasi Graf

Representasi graf dalam hal ini bagaimana cara mengimplementasi sebuah graf pada komputer dan bahasa pemrograman. Implementasi yang paling mudah atau paling sederhana adalah menggunakan sebuah matriks untuk memodelkan hubungan antar simpulnya. Jenis-jenis matriks yang biasa digunakan untuk merepresentasikan matriks yaitu :

1. Matriks Ketetanggaan

Representasi graf menggunakan sebuah matriks dengan sisi dan kolom adalah semua simpul. Misalkan A = [a_{ij}] , jika a_{ij} adalah 1 maka simpul i dan j bertetangga sedangkan jika 0 maka simpul i dan j tidak bertetangga. Untuk graf berbobot nilai dari a_{ij} akan sesuai dengan bobot dari sisi yang menghubungkan simpul i dan j.

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

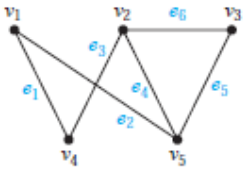
Matriks ketetanggaan akan berdasaran urutan simpul yang telah dibuat oleh pembuat. Misalkan simpul dinamankan dengan huruf a, b dan c maka urutan simpul mulai dari a kemudian b dan c. Ukuran dari matriks ketetanggaan adaah n x n. Dengan n adalah jumlah simpul yang ada pada graf. Jadi matriks yang dihasilkan dari matriks ketetanggaan adalah matriks persegi.

Matriks ketetanggaan adalah matriks sederhana yang simetris dengan $a_{ij} = a_{ji}$. Matriks ketetanggaan juga dapat merepresentasikan graf dengan cincin dan graf dengan sisi banyak. Cincin pada simpul dapat di representasi dengan memberi angka 1 pada diagonal matriks tersebut. Untuk graf dengan sisi banyak pada 2 simpulnya, dapat direpresentasikan dengan menulis jumlah sisi pada simpul tersebut. Tetapi hal ini hanya berlaku pada graf tidak berbobot.

2. Matriks Berisisian

Matriks bersisian merepresentasikan hubungan sisi dengan simpul. Misalkan A = [a_{ij}], jika a_{ij} bernilai 1 maka sisi i akan bersisian dengan simpul j dan jika bernilai 0 maka sisi i dan

simpul j tidak saling bersisian. Dengan baris merepresentasikan simpul dan kolom merepresentasikan sisi maka ukuran dari matriks bersisian adalah jumlah simpul x jumlah sisi. Ini mengakibatkan matriks yang dihasilkan tidak terbatas dengan matriks persegi. Untuk merepresentasikan berat pada sisinya, nilai dari 1 pada matriks diganti dengan berat dari sisi tersebut.



	e_1	e_2	e_3	e_4	e_5	e_6
v_1	1	1	0	0	0	0
v_2	0	0	1	1	0	1
v_3	0	0	0	0	1	1
v_4	1	0	1	0	0	0
v_5	0	1	0	1	1	0

Graf dengan sisi banyak dan cincin dapat direpresentasikan juga dengan matriks ini. Untuk merepresentasikan sisi banyak dapat dengan mengganti angka 1 dengan jumlah sisi yang menghubungkan dua buah simpul. Untuk cincin dapat direpresentasikan dengan menggunakan hanya satu kolom pada suatu sisinya.

III. Algoritma Dijkstra

Algoritma Dijkstra, diperkenalkan pada tahun 1959 dan diberi nama berdasarkan penciptanya Edsger Dijkstra. Algoritma ini mencari jalan tercepat dari suatu simpul menuju simpul yang lain. Algoritma ini memiliki keunggulan yaitu dapat diterapkan pada graf berbobot.

Algoritma ini dapat diterapkan pada graf berarah maupun tidak berarah. Akan tetapi algoritma ini tidak bisa dipakai pada graf dengan bobot negatif. Algoritma ini akan menghitung jarak terdekat dari sebuah simpul sumber menuju simpul-simpul yang lain pada suatu graf.

```
Pseudocode :

Function Dijkstra(Graph,source):

Distance[source] = 0
For each vertex v in Graph :

  If v <> source
    Distance[v] = infinity
  Add v to Q

While Q is not empty :
  V = vertex in Q with min Distance[v]

  For each neighbor u of v:
    Alt = Distance[v] + length(v,u)
    If alt < Distance[u]:
      Distance[u] = alt

return Distance[]

end function
```

Pada algoritma ini, akan ada dua masukkan. Pertama adalah graf yang digunakan kedua simpul sumber yang ingin dicari jarak terdekatnya. Lalu diinisiasi sebuah array yang akan menampung jarak-jarak dari sebuah simpul menuju simpul sumber. Kemudian dilihat simpul sekitar simpul sumber. Jarak akan dihitung dari simpul dengan sisi berbobot paling kecil. Setelah semua simpul sekitar simpul sumber sudah diperiksa maka akan berlanjut ke simpul sekitar simpul dengan jarak terkecil.

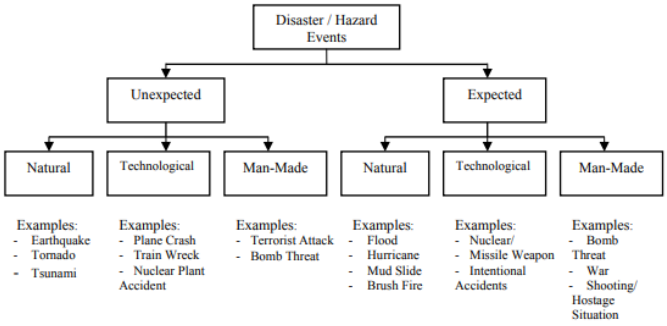
Simpul sekitar simpul ini akan diperiksa jaraknya seperti pada saat simpul sumber. Jika ada sebuah simpul memiliki jarak lebih kecil dari jarak yang sudah dicatat, maka jarak yang sudah dicatat akan diubah menjadi jarak yang terpendek. Hal ini akan terus-menerus berlangsung sampai semua simpul diperiksa.

Saat semua simpul sudah diperiksa maka fungsi tersebut akan me miliki keluaran data jarak-jarak terdekat semua simpul menuju simpul sumber.

IV. Emergency Management

The Federal Emergency Management Agency atau FEMA membuat sebuah taksonomi bencana dan kejadian berbahaya. Taksonomi tersebut membuat bencana terdiri dari 3 tipe, buatan manusia, alami dan teknologi[2]. Bencana pertama yaitu bencana buatan manusia berarti bencana yang sengaja dilakukan atau dibuat oleh manusia. Bencana kedua yaitu bencana alami berarti bencana yang terjadi karena faktor alam.Bencana ini terjadi sendirinya tidak dibuat atau disengaja oleh manusia. Bencana ketiga yaitu bencana teknologi. Bencana ini adalah bencana yang dikarenakan sebuah teknologi. Bisa berupa kecelakaan dalam penggunaan teknologi atau penggunaan yang salah pada teknologi tersebut.

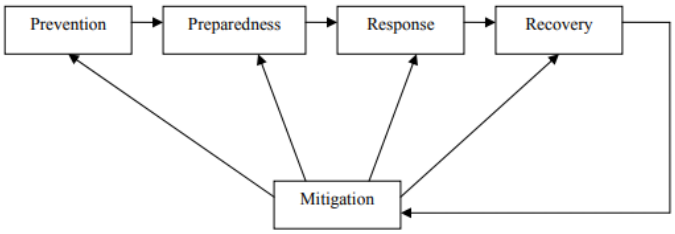
Selain tiga tipe tersebut, bencana juga dikategorikan menjadi dua, yaitu terduga dan tidak terduga. Bencana tidak terduga adalah bencana yang membuat tindakan darurat harus dilakukan dalam waktu yang singkat dan dan cepat. Dengan kata lain seseorang yang mengalami bencana tersebut hanya miliki waktu yang singkat untuk mempersiapkan diri dan mengevakuasi diri sendiri saat bencan berlangsung.



A. Emergency Management Phase

Sudah sejak lama cara terbaik dalam mitigasi dampak negatif suatu bencana adalah kemampuan untuk membuat perencanaan

yang efektif dan cepat dalam merespons suatu bencana. Perencanaan dan respons dari suatu bencana diklasifikasikan menjadi 5 fase[2].



Fase pertama adalah prevention atau pencegahan. Dalam fase ini penyebab dari suatu bencana akan di awasi, analisis dan deteksi. Fase selanjutnya adalah *Preparedness* atau persiapan. Persiapan yang dilakukan seperti pemasangan *early warning system*, pemasangan makanan dan obat-obatan darurat serta pelatihan dalam keadaan darurat. Fase berikutnya adalah *Response* atau respons. Respons yang baik dapat meminimalisir dampak buruk dari suatu bencana yang terjadi. Fase keempat adalah *Recovery* atau pemulihan. Dalam fase ini diusahakan suatu daerah yang terkena bencana dipulihkan kembali ke keadaan sebelum bencana terjadi. Terakhir adalah *Mitigation* atau fase mitigasi. Fase ini adalah fase setelah pemulihan dimana tujuannya adalah mereduksi atau mengurangi dampak buruk pada bencana alam yang serupa.

B. Perencanaan Rute Evakuasi

Perencanaa rute evakuasi merupakan salah satu bagian penting dari penanganan suatu bencana. Rute evakuasi harus dibuat dengan baik dan efisien sehingga evakuasi akan berlangsung cepat. Evakuasi merupakan proses yang kompleks. Setelah bencana terjadi, para korban harus melakukan evakuasi dengan cepat ke tempat yang aman. Dengan adanya rute evakuasi yang baik maka proses evakuasi ini akan berjalan cepat.

Dalam perencanaannya, algoritma sering digunakan untuk menentukan jalur evakuasi. Algoritma digunakan agar rute evakuasi bisa disimulasikan pada komputer. Simulasi berguna untuk mengetahui seberapa efisien dari sebuah jalur evakuasi yang sudah dibuat atau seberapa efisien sebuah rancangan gedung dalam keadaan darurat untuk melakukan evakuasi

Simulasi merupakan media yang sangat baik untuk memperlihatkan dan memproyeksikan kondisi sesungguhnya saat keadaan darurat terjadi. Dengan simulasi ini, untuk menilai sebuah gedung tidak perlu menunggu keadaan darurat terjadi dan mengurangi biaya yang diperlukan untuk melakukan penilaian.

V. Pemodelan Rute Evakuasi Sebuah Bangunan Menggunakan Graf Berbobot

Dalam pemodelan rute evakuasi, graf yang digunakan adalah graf berbobot. Graf tipe tersebut digunakan karena dapat merepresentasikan waktu dari setiap jalur yang dilewati. Waktu yang diperlukan dalam proses evakuasi pun dapat dihitung dan diprediksi.

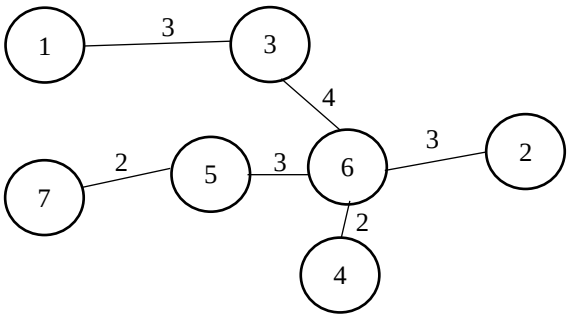
Dalam model rute evakuasi menggunakan graf, setiap simpul akan mewakili sebuah ruangan. Ruangan ini seperti kamar,koridor,tangga dan ruangan-ruangan lain dari sebuah bangunan. Selain itu akan dibuat simpul tambahan yang merepresentasikan daerah aman atau tujuan akhir dari evakuasi. Tujuan akhir ini bisa berupa lapangan yang aman ataupun pintu darurat yang keluar dari sebuah bangunan. Misalkan kita akan membuat rute evakuasi sebuah rumah sederhana. Rumah tersebut akan terdiri dari 7 ruangan yang diwakili oleh 7 buah simpul.

Simpul	Ruangan
1	Kamar 1
2	Kamar 2
3	Tangga
4	Kamar Mandi
5	Ruang Tamu
6	Selasar Tengah
7	Halaman Depan(Daerah Aman)

Setiap ruangan yang direpresentasikan sebagai simpul akan memiliki sisi. Sisi tersebut akan merepresentasikan proses perpindahan dari suatu ruangan ke ruangan yang lain. Dalam graf ini sisi akan memiliki bobot. Bobot tersebut akan merepresentasikan waktu yang diperlukan untuk berpindah dari suatu ruangan ke ruangan yang lain. Diasumsikan waktu yang akan dipakai sebagai contoh tidak menggukan satuan yang sudah ada. Waktu yang ditempuh ini hanya perkiraan waktu berpindah dari posisi tengah suatu ruangan ke ruangan yang lain.

Simpul	Waktu(Satuan Waktu)
1 - 3	3
3 – 6	4
6 – 2	3
6 – 5	3
6 – 4	2
5 – 7	2

Dengan menggunakan pendekatan ini, kita dapat memetakan sebuah gedung menggunakan graf dengan setiap simpul adalah ruangan dan sisi dengan bobot waktu yang ditemput untuk berpindah ruangan.



Graf yang terbentuk telah merepresentasikan jalur-jalur yang ada pada sebuah gedung. Dalam contoh ini adalah sebuah rumah sederhana. Graf yang telah digambar akan diubah representasinya kedalam matriks agar bisa dioleh dalam komputer. Matriks yang digunakan adalah matriks ketetanggaan. Matriks ini dipilih karena dapat menyimpan data bobot dari sisi atau dalam konteks ini waktu tempuh dari suatu ruangan ke ruangan lain. Dari graf diatas dihasil sebuah matriks ketetanggaan seperti berikut.

$$\begin{bmatrix} 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 3 & 0 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 2 \\ 0 & 3 & 4 & 2 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 \end{bmatrix}$$

VI. Penentuan Rute Tercepat Menggunakan Algoritma Dijkstra

Denah atau jalur-jalur dari sebuah gedung atau rumah yang telah direpresentasikan menggunakan graf sudah dapat digunakan dalam implementasi algoritma dijkstra dalam menentukan rute atau jalur evakuasi. Dalam hal ini permasalahan hanya kan terbatas pada rute tercepat yang dapat digunakan dalam evakuasi. Rute ini berdasarkan waktu yang diperlukan untuk berpindah dari satu ruangan ke ruangan yang lain atau dari satu simpul ke simpul yang lain.

Dalam implementasi penentuan rute tercepat, Algoritma Dijkstra akan dipakai saat mencari jalur tercepat dari suatu ruangan menuju daerah aman. Jadi program akan mencari jalur tercepat dari semua ruangan yang ada pada gedung atau rumah tersebut dengan memperhitungkan waktu temputnya.

```
Pseudocode :

Function RuteEvakuasi(Graph,Ruangan[],Aman[]):

Jalur[] = 0

For each ruang R in Ruangan :

  For each daerah D in Aman :

    Jarak[] = Dijkstra mencari jalur terdekat
    kedaerah D

  Jalur[R] = Nilai Jarak[] paling minimal


return Jalur[]

end function
```

Untuk menentukan jalur evakuasi, setiap ruangan akan dimasukkan ke dalam array ruangan dan daerah aman akan dimasukkan kedalam array aman. Untuk jalur evakuasi akan disimpan dalam array jalur. Setiap ruangan akan di cek jalur tercepat menuju setiap daerah aman. Jalur ke daerah aman tercepat pada setiap ruangan yang akan dimasukkan kedalam array jalur. Sehingga program akan menampilkan hanya masing-masing satu jalur tercepat pada tiap ruangan.

Dalam bentuk aslinya, algoritma Dijkstra hanya akan menghitung jarak terdekat dari suatu sumber menuju semua simpul yang ada. Pada hal ini algoritma tersebut harus dimodifikasi sehingga cocok untuk digunakan pada fungsi diatas. Algoritma Dijkstra diubah hanya akan menghitung jarak terdekat menuju suatu simpul dari sumber. Setelah mengetahui jarak terdekat tersebut akan dilakukan *Backtracking* dari simpul tujuan menuju sumber untuk mengetahui jalur tercepat yang dilalui untuk menuju simpul tujuan. Untuk melalukan *Backtracking* harus ada informasi tambahan yaitu simpul sebelumnya yang dilalui pada setiap simpul yang dilalui sehingga *Backtracking* dapat dilakukan.

Ruangan	Jalur	Waktu(Satuan Waktu)
1	1-3-6-5-7	12
2	2-6-5-7	8
3	3-6-5-7	9
4	4-6-5-7	7
5	5-7	2
6	6-5-7	5

Jika contoh rumah sederhana digunakan pada program diatas, maka akan menghasilkan keluaran seperti pada tabel diatas. Dari hasil ini diketahui bahwa waktu terlama untuk evakuasi adalah 12 satuan waktu yaitu dari ruangan 1. Jalur yang dilalui untuk evakuasi yaitu melalui ruangan 3, 6 dan 5. Ruangan ini menjadi ruangan paling jauh dari daerah aman.

Hasil program ini bisa menjadi acuan dalam penentuan jalur evakuasi sebuah gedung. Karena pada contoh ini rumah yang digunakan sangat sederhana sehingga pilihan jalur evakuasi memang tidak banyak bahkan hanya satu. Akan tetapi jika diterapkan pada gedung yang lebih kompleks, hasil dari program tersebut akan lebih terlihat dan penentuan pilihan dari jalur evakuasi akan lebih jelas.

Selain untuk menentukan jalur evakuasi, hasil dari program tersebut dapat mensimulasikan secara kasar waktu yang diperlukan untuk melakukan evakuasi. Dengan adanya informasi waktu yang ditempuh sudah cukup untuk mengetahui seberapa cepat sebenarnya evakuasi seseorang dari sebuah ruangan menuju daerah aman dapat dilakukan.

VII. ACKNOWLEDGMENT

Pada kesempatan ini penulis ingin bersyukur dan berterima kasih kepada Allah SWT. karena atas berkah dan izinnya penulis diberikan kesempatan untuk mendapatkan dan menyelesaikan tugas makalah ini. Kemudian penulis ingin berterima kasih kepada seluruh dosen dari mata kuliah Matematika Diskrit IF2120 – terutama kepada Dra. Harlili, M.Sc – karena telah membimbing penulis dalam menguasai ilmu matematika diskrit selama satu semester terakhir. Kemudian penulis mengucapkan terima kasih atas semua pihak yang telah membantu pengerjaan makalah ini, baik secara langsung maupun tidak langsung.

REFERENCES

[1] Rosen, Kenneth H. , *Discrete Mathematics and Its Application*. New York : McGraw-Hill,2012,pp. 641 – 727

[2] Kittirattanapaiboon, Suebpong. "Emergency Evacuation Route Planning Considering Human Behavior during Short- and No-Notice Emergency Situations." *University of Central Florida* , 2009.

[3] Abiy, Thaddeus et el. "Dijkstra's Shortest Path Algorithm". *Brilliant Math & Science Wiki*. Retrieved 03:44, December 10, 2018, from https://brilliant.org/wiki/dijkstras-short-path-finder/

[4] Shekhar, Shashi, et al. "Experiences with Evacuation Route Planning Algorithms." *International Journal of Geographical Information Science*, vol. 26, no. 12, 2012, pp. 2253–2265., doi:10.1080/13658816.2012.719624.

[5] Mishra, Gopinath, et al. "Improved Algorithms for the Evacuation Route Planning Problem." *Journal of Combinatorial Optimization*, vol. 36, no. 1, 2016, pp. 280–306., doi:10.1007/s10878-016-0082-0.

PERNYATAAN

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

Bandung, 12 Desember 2018

Naufal Zhafran Latif dan 13517095