Analisis Algoritma Dijktra dan Algoritma Bellman-Ford Sebagai Penentuan Jalur Terpendek Menuju Lokasi Kebakaran (Studi Kasus: Kecamatan Praya Kota)

Saeful Hamdi[1], Prihandoko[2]

[1,]Program Studi Teknik Informatika-Universitas Amikom Yogyakarta
[2,]Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Gunadarma, Jakarta
Jalan Ring Road, Condongcatur, Depok, Condongcatur, Kec. Depok, Kabuapten Sleman, Yogyakarta
[1] saefulhamdi215@gmail.com, [2] pri@staff.gunadarma.ac.id,

Terima Naskah : 5 Maret 2018
Terima Revisi : 10 Maret 2018

ABSTRAK

Algoritma *Dijkstra* dan Algoritma *Bellman-ford* merupakan salah satu alogirtma yang digunakan untuk menentukan jalur terpendek. Cara kerja Algoritma *Dijkstra* dan Algoritma *Bellman-Ford* memakai strategi *greedy*, di mana pada setiap langkah dipilih sisi dengan bobot terkecil yang menghubungkan sebuah simpul yang sudah terpilih dengan simpul lain yang belum terpilih. Algoritma *Dijkstra* dan Algoritma *Bellman-Ford* membutuhkan parameter tempat asal, dan tempat tujuan. Hasil akhir dari algoritma ini adalah jarak terpendek dari tempat asal ke tempat tujuan beserta rutenya. Untuk pembuatan jalur terpendek melibatkan beberapa faktor, antara lain kecepatan, tingkat kemacetan, waktu tempuh perjalanan, lebar jalan, dan posisi pipa hidran. Penentuan jalur terpendek dengan menggunakan kedua algoritma tersebut sebagai refrensi rute yang digunakan Dinas Pemadam Kebakaran untuk mencapai titik lokasi. penentuan jalur terpendek dilakukan pada daerah kota praya yang mempunya luas wilayah 31,12 km2. Merupakan wilayah yang tidak terlalu luas namun mempuyai tata ruang dan aministrasi yang kompleks.
Kata kunci : jalur terpendek *Dijkstra* dan *bellman-ford*

ABSTRACT

Dijkstra's Algritma and Bellman-ford Algorithm is one of the algorithms used to determine the shortest path. The workings of the Dijkstra Algorithm and Bellman-Ford Algorithm use the greedy strategy, in which each step is selected with the smallest weights linking a selected node with another undelected node. Dijkstra's Algorithm and Bellman-Ford Algorithm require parameters of origin, and place of destination. The end result of this algorithm is the shortest distance from the place of origin to the destination along with the route. For the shortest path construction involves several factors, including speed, congestion level, travel time, road width, and hydrant pipeline position. Determination of the shortest path by using the two algorithms as a reference route used by Fire Department to reach the point of location. the determination of the shortest path is done in the praya city area which has a total area of 31.12 km2. Is a region that is not too broad but mempuyai spatial and aministrasi complex.
Keywords : *Dijkstra's shortest path and bellman-ford*

PENDAHULUAN

Peristiwa kebakaran merupakan bencana yang tidak bisa diprediksi sebelumnya. Peristiwa ini antara lain disebabkan oleh hubungan arus pendek listrik, puntung rokok yang dibuang di sembarang tempat, ledakan tabung gas LPG, dan lain-lain. Peristiwa kebakaran dapat menyebabkan kerugian yang cukup tinggi bagi korbannya. Sehingga dalam penanganannya harus tepat dan cepat. Ketika kebakaran terjadi, pihak yang pertamakali dihubungi tentulah Dinas Pemadam Kebakaran. Untuk itu mobil pemadam kebakaran harus memilih jalur terpendek dari kantor pemadam kebakaran ke lokasi terjadinya kebakaran.

Kota Praya mempunyai luas wilayah 31,12 km2. walaupun wilayahnya tidak terlalu luas, namun memiliki tata ruang dan administrasi yang lengkap. sektor pertahanan dalam hal ini pemadam kebakaran yang didukung dengan fasilitas serta sarana transportasi yang tersedia di kawasan namun untuk menuju ke tempat kebakaran tersebut, ada

beberapa rute yang bisa ditempuh. petugas pemadam kebakaran pastinya menginginkan jalur yang paling efisien untuk menuju tempat kebakaran sehingga dapat menghemat waktu dan biaya. sehingga dapat menjalankan tugas dengan baik dan lebih sigap dalam memadamkan api.

Penentuan jalur terpendek dibutuhkan langkah-langkah atau metode untuk dapat menentukan jalur yang tepat dan lebih cepat untuk mencapai titik lokasi. Algirtma yang digunakan untuk menentukan jalur terpendek ialah Algoritma Dijkstra dan Algoritma bellmanf-ford. Cara kerja Algoritma *Dijkstra* dan Algoritma *Bellman-Ford* memakai strategi *greedy*, di mana pada setiap langkah dipilih sisi dengan bobot terkecil yang menghubungkan sebuah simpul yang sudah terpilih dengan simpul lain yang belum terpilih. Algoritma *Dijkstra* dan Algoritma *Bellman-Ford* membutuhkan parameter tempat asal, dan tempat tujuan. Hasil akhir dari algoritma ini adalah jarak terpendek dari tempat asal ke tempat tujuan beserta rutenya. Untuk pembuatan jalur terpendek melibatkan beberapa faktor, antara lain kecepatan, tingkat kemacetan, waktu tempuh perjalanan, lebar jalan, dan posisi pipa hidran.

Tinjauan Pustaka

Rancang Bangun Sistem Informasi Goegerafis Untuk Membantu Pencarian Jalur Terpendek Menuju ATM Bank BRI Dengan Metode *Tabu Search Algorithm* (TS), Umumulhadi Dkk, (2016). Pada penelitiannya menjelaskan bagaimana mempermudah rute dalam pencarian mesin ATM Bank BRI. Dengan menggunakan metode *Tabu Search Algorithm*, dapat mencari cost terendah dari beberapa iterasi yang ada. Kesimpulan dari penelitian yang dilakukan dapat menentukan jalur terpendek dari iterasi yang ada, dalam hal ini penelitian dan pengujian dilakukan dia Macini Sawah 1 dan Jalan Borong Raya.

Pencarian Rute Terpendek Menggunakan Algoritma *Dijkstra* Pada Sistem Informasi Geogerafis Pemetaan Stasiun Pengisian Bahan Bakar Umum. Bery Juanda Dkk, (2016). Pada penelitiannya membahas tentang hasil dan implementasi cara kerja dan fungsi pada sistem informasi geogerafis pemetaan SPBU di Kota Padang. Tujuan dari penelitiannya adalah utuk membuat atau menghasilkan sistem pemetaan lokasi SPBU di kota padang menggunakan Algoritma *Dijkstra.*

Optimasi Jalur Tercepat Dengan Menggunakan Modifikasi Algoritma *Bellman-Ford* (Studi Kasus Lintasan antara Kecamatan Kota Malang), Agustian Aji P, Dkk, (2015). Pada penelitiannya mengusulkan modifikasi dari Algoritma *Bellman-Ford*, sebagai salah satu algortima dalam mentukan optimasi jalur tercepat antar kecamatan kota malang. Dengan menggunakan Algortima *Bellman-Ford* perjalan yang akan ditempuh lebih cepat dengan menghemat bahan bakar.

Penentuan Jarak Terpendek dan Jarak Terpendek Alterrnatif Menggunakan Algoritma *Dijkstra* Serta Estimasi Waktu Tempuh, Asti Ratnasari Dkk, (2013). Penelitiannya membahas tentang penggunaan Algoritma *Dijkstra* sebagai pencarian jalur terpendek pada suatu graf. Hasil yang didapatkan dari implementasi sistem yang dibuat adalah mampu menemukan jarak terpendek dan jarak terpendek alternatif ketika terjadi hambatan (pembelokiraan jalan) pada jalur terpendek utama dan juga dapat mengetahui estimasi waktu tempuhnya.

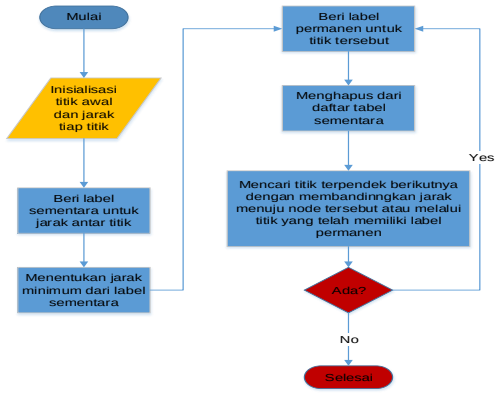
Sistem Informasi Geogerafis Pariwisata Berbasis Web Dan Pencarian Jalur Terpendek Dengan Algoritma *Dijkstra*, Antonio Gusmao Dkk, (2013). Tujuan dari penelitiannya untuk membantu kementerian pariwisata timur leste dalam mengembangkan industri pariwisata untuk memperoleh informasi yang mudah diakses dari berbagai tempat menggunakan internet. Peneletian ini menampilkan peta digital pada *Web* dengan *Google Map API.*

Algoritma

Secara informal algortima adalah prosedur komputasi yang didefinisikan dengan baik yang mengambil beberapa nilai, atau kumpulan nilai, sebagai masukan dan menghasilkan beberapa nilai, atau kumpulan nilai sebagai keluaran. Algoritma dengan demikian merupakan urutan langkah-langkah komputasi yang mengubah *input* menjadi *output*.(Tomas H. Cormen, 2009).

Algoritma *Dijkstra*

Algoritma *Dijkstra* ditemukan oleh *Edsger Wybe Dijkstra* pada tahun 1959. Algoritma ini merupakan algoritma yang dapat memecahkan masalah pencarian jalur terpendek dari suatu graf pada setiap simpul yang bernilai positif atau tidak negative. Dalam pencarian jalur terpendek Algortima *Dijkstra* bekerja dengan mencari bobot yang paling minimal dari suatu graf berbobot, jarak terpendek akan didapatkan dari dua atau lebih titik dari suatu graf dan nilai total yang didapat adalah yang bernilai palling kecil. *Dijkstra* merupakan salah satu varian bentuk algortima populer dalam pemecahan persoalan terkait masalah optimasi pencarian lintasan terpendek sebuah lintasan yang mempunyai panjang minimum dari verteks a ke z dalam graph berbobot, bobot tersebut adalah bilangan positif jadi tidak dilalui oleh node negatif. Namun jika terjadi demikian, maka penyelesaian yang diberikan adalah infiniti (Tak Hingga). Pada Algortima *Dijkstra*, *node* digunakan karen Algoritma *Dijkstra* menggunakan graf berarah untuk penentuan rute lintasan terpendek.



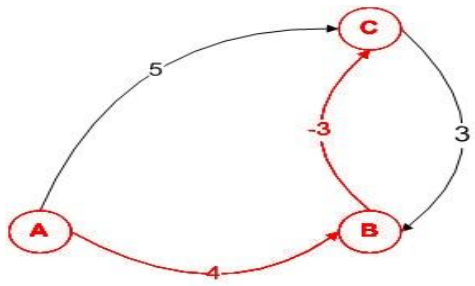
Gambar 1. *Flowchart Algortiam Dijkstra*

Algoritma *Bellman-Ford*

Algoritma *Bellman-Ford* dikembangkan oleh Richard Bellman and Lester Ford, Jr. Algoritma ini sangat mirip dengan Algortima *Dijkstra* namun algortima ini mampu mengani bobot negatif pada pencarian jalur terpendek pada sebauh graf berbobot. Algoritma *Bellman-Ford* merupakan pengembangan dari Algoritma *Dijkstra*, Algortima *Bellman-Ford* akan benar jika dan hanya jika graf tidak terdapat *cycle* dengan bobot negatif yang dicapai dari sumber.

Ciri-ciri Algoritma *Bellman-ford:*

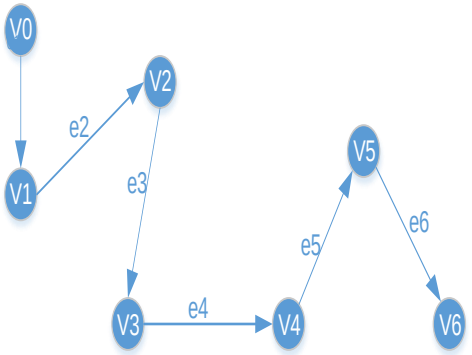
1. Bekerja walaupun terdapat edge dengan bobot negatif.
2. Harus *directed edge* (jika tidak graf akan memilik *cycle* dengan bobot negatif).
3. Iterasi i menemukan seluruh *shortest path* dengan menggunakan i *edge*.
4. Dapat mendeteksi *cycle* dengan bobot negatif jika ada.



Gambar 2. Contoh Algoritma *Bellman-Ford*

Graf

Teori graf merupakan pokok bahasan yang sudah tua usiannya namun memiliki banyak terapan dalam kehidupan sehari-hari saat ini. Graf digunakan untuk mempresentasikan objek-objek diskrit dan hubungan antar objek-objek tersebut. Banyak persoalan di dunia nyata yang merupakan gambaran visual dari graf. Contoh salah satu repsentasi visual graf adalah peta. Banyak hal yang dapat digali dari repsentasi tersebut, diantaranya menentukan jalur terpendek dari satu tempat ke tempat yang lain, menggambar dua kota yang bertetangga dengan warna yang berbeda pada peta, menentukan tata letak jalur transportasi, pengaturan jaringan telekomunikasi atau jaringan internet dan masih banyak lagi. Selain peta, masih banyak hal lain dalam dunia nyata yang merupakan repsentasi visual dari graf.



Gambar 3. contoh Graft

***Global Positioning System* (GPS)**

Global Positioning System (GPS) merupakan suatu sistem koordinat yang dapat digunakan untuk menentukan koordinat suatu lokasi berdasarkan posisi bujur, lintang, serta ketinggiannya. Pada awalnya teknologi ini dikembangkan oleh Departemen Pertahanan Amerika Serikat. Namun pada saat ini teknologi ini dapat digunakan oleh masyarakat luas. Untuk mengetahui lokasi dari penerima terlebih dahulu perlu diketahui jarak antara satelit dengan penerima. Jarak tersebut dapat dapat diketahui dengan mengetahui waktu untuk melakukan transmisi sinyal yang dipancarkan oleh satelit hingga diterima oleh penerima. Jarak tersebetu dapat dihitung menggunakan rumus sebagai berikut:

$$S = c.\Delta t$$

Dimana :

S = Jarak yang dihitung
c = merupakan cepat rambat sinyal udara
Δt =selang waktu yang dibutuhkan gelombang dari satelit ke perangkat GPS.

METODE

Pada penelitian ini akan mengkaji empat kriteria yang diperlukan untuk mengukur unjuk kerja antara *Dijkstra* dan *Bellman-Ford*. Menurut Russel (2011), ada empat kriteria dalam mengevaluasi ferforma dari sebuah algoritma, yaitu :

1. *Completeness*
   Pada kriteria ini menilai bagaimana sebuah algortima mampu menjamin ketersediaan solusi ketika solusi tersebut memang ada. Sehingga, pada penelitian ini harus

menjamin menemukan jalur terdekat dari titik start.

2. *Optimality*

Kriteria ini menjukan kemampuan algortima dalam menemukan solusi optimal. Dalam hal ini kemampuan algortima dalam menemukan jalur terdekat dengan rute yang terpendek.

3. *Time Complexity*

Kompleksitas waktu menunjukkan seberapa lama sebuah algortima dalam menemukan solusi. Solusi penelitian ini adalah, rute terpendek dalam pencarian jalur terdekat dari titik start sampai lokasi tujuan.

4. *Space Complexity*

Kriteria ini menunjukkan seberapa besar memory yang akan dibutuhkan oleh algoritma pencarian dalam menemukan rute terpendek pada pencarian jalur terpendek lokasi kebakaran.

Metode Pengumpulan Data

Metode pengumpulan data adalah cara atau teknik yang dapat digunakan oleh peneliti untuk mengumpulkan data. Dalam penelitian ini mentode pengumpalan data yang dilakukan adalah sebagai berikut:

1. Studi Kasus

Studi pustaka dilakukan dengan mencari informasi dari buku, jurnal, artikel, situs internet yang berhubungan dengan penelitian guna mendukung penelitian ini.

2. Wawancara

Wawancara atau *interview* adalah cara pengumpulan data yang digunakan untuk memperoleh informasi lansung dari sumbernya. Wawancara digunanakan bila kita ingin mengetahui hal-hal dari responden, yang jumlahnya sedikit, secara lebih mendalam.

3. Observasi

Observasi atau pengamatan adalah suatu teknik atau cara untuk mengumpulkan data dengan jalan mengamati kegiatan yang sedang berlansung. Pengamatan dapat dilakukan dengan partisipasi ataupun nonpartisipasi. Dalam pengamatan partisipatori (*participatory observation*) pengamat ikut serta dalam kegiatan yang sedang berlansung. Sementara pengamatan nonpartisipatori (*nonparticipatory observation*) pengamat tidak ikut serta dalam kegiatan. Pengamat hanya berperan mengamati kegiatan

4. Dokumentasi

Dokumentasi ditujukan untuk memperoleh data lansung dari tempat penelitian, meliputi buku, peraturan, laporan kegiatan, foto, film dokumenter, dan data yang relevan dengan penelitian. Dokumen merupakan catatan peristiwa yang sudah berlalu. Dokumen bisa berbentuk tulisan, gambar, atau karya monumental seseorang.

Metode Analisis Data

Metode analisis data dilakukan dengan tujuan untuk memperoleh gambaran mengenai analisis dan perencanaan desain ptototype penentuan jalur terpendek menggunakan algortima dijkstra dan bellman-ford pada pemadam kebakaran menuju lokasi kejadian kebakaran berdasarkan titik lokasi hydrat terdekat dari lokasi kejadian kebakaran.

1. Analisis (*Analyze*)

Tahapan ini bertujuan untuk mengidentifikasi sistem informasi apa yang akan dibagun, sasaran-sasaran yang akan dicapai, jangka waktu pelaksanaan, serta mempertimbangkan *cost* yang akan dihabiskan dan yang tersedia. Dalam hal pecarian jalur terpendek perencanaan sangat dibutuhkan karena rentan kaitan dengan jalur-jalur yang akan dilalui dalam pencarian rute, baik dari segi jalan, tanjakan, kepadatan jalan, dan akses jalan.

2. Desain (*Design*)

Analisis sistem digunakan untuk menjawab pertanyaan *What*?. Sedangkan Desain sistem digunakan untuk menjawab pertanyaan *how?*. Desain berkonsentrasi pada bagaimana sistem dibagun untuk memenuhi fase analisis atau perencanaan.

Manfaat desain sistem adalah memberikan gambaran rancang bangun (*blue print*) yang lengkap, sebagai penuntun (*guideline*) bagi programer dalam membuat aplikasi.

HASIL DAN PEMBAHASAN

Penelitian ini dimulai dengan mengkaji kerteria untuk mengukur kinerja dari masing-masing agloritma untuk untuk mendukung keputusan. Ada empat jenis kerteria untuk mengevaluasi performa dari algoritma tersebut untuk mendukung dalam menentukan pengambilan keputusan untuk menentukan jalur terpendek.

Kelengkapan (*Completeness*)

Pada kerteria ini mengukur performa algoritma dari sisi kelengkapan dalam menyediakan solusi. Kelengkapan dari ketersediaan jalur yang dimulai dari titik awal menuju simpul yang berhubungan. Jaminan kelengkapan ketersedian solusi dari algoritma mampu menentukan jalur terpendek dari setiap simpul pada titik koordinat yang telah ditentukan pada sistem *Global Positioning System* (GPS). Kelengkapan termasuk dari aspek-aspek yang mendukung untuk mencapai dari titik simpul akhir dari graf yang digambarkan.

Optimalitas (*Optimality*)

Tahapan ini tahap pengkajian kemampuan dari algoritma untuk menemukan solusi jalur terpendek yang optimal. Kerteria ini lanjutan dari kerteria

kelengkapan dimana pada pada kerteria tersebut menjamin ketersedian solusi jalur terpendek, sedangkan pada kerteria optimalias menemukan solusi jalur terpendek dari ketersediaan solusi jalur terpendek. Solusi yang dihasilkan ialah membandingkan hasil dari ketersediaan dari masing-masing ketersediaan solusi, untuk menemukan jalur yang menuju titik akhir dari simpual yang telah digambarkan dalam bentuk graf.

kompleksitas waktu (*Time Complexity*)

Evaluasi dari komplesitas waktu menunjukan waktu yang dibutuhkan oleh algoritma dalam menemukan solusi jalur terpendek. Penentuan titik simpul jalur terpendek dedefisikan berdasark titik koordinat yang ditentukan dari *Global Positioning System* (GPS) dan masing titik digambarkan dalam bentu graf. Titik simpulan awal ditentukan dan menghubungkan dengan titik simpul yang dikenali dan yang belum dikenali, menuju titik simpul tujuan dari lintasan yang dihubungkan. Kemudian akan ditentukan waktu yang dibutuhkan dari titik simpul menuju titik simpul yang lain sampi titik tujuan, sehingga ditemukan solusi jalur terpendek.

Kompleksitas ruang (*Space Complexity*)

Besar beban memori akan mempengaruhi kinerja dari sebuah algoritma untuk mencari solusi jalur terpendek. Penggunaan memori akan ditentukan dengan variable yang didefinisikan. Variable akan menggunakan ruang pada memori yang akan mempengaruhi kinerja dari algoritma.

Algoritma *Dijkstra*

a. *Pesudocode Algortima Dijkstra*

```
{ Algoritma Dijkstra }
procedure Dijkstra
        (
        input m: matriks,   a: integer {simpul awal}
        )
{mencari lintasan terpendek dari simpul awal a ke semua simpul lainya   Masukan : matriks ketetanggaan (m) dari graf berbobot G dan simpul awal a   Keluaran: lintasan terpendek dari a ke semua simpul lainnya}
KAMUS
s1,s2,...,sn        : interger {larik interger}
d1,d2,...,dn        : interger {larik interger}
i                   : interger

ALGORITMA
{Langkah 0 (inisialisasi) : }
for i ← 1 to n do
        si ← 0
        di ← mai
endfor

 {Langkah 1: }
        sa ← 1
{karena simpul a adalah simpul asal lintasan terpendek, jadi terpilih dalam lintasan terpendek}
        da ← infinity
{tidak ada lintasan terpendek dari simpul a ke a}
{Langkah 2,3,...,n1 :}
for i ← 2 to n-1 do
{Cari j sedemikian sehingga sj = 0  dan dj = min (d1,d2,...,dn)}
        Sj ← 1
{simpul j sudah terpilih ke dalam lintasan terpendek}
{perbarui di, untuk i = 1,2,3,...,n  dengan : di (baru) = min{di(lama), dj + mji}
endfor
```

b. Kompleksitas Waktu *Algoritma Dijkstra*

Untuk himpunan simpul serta sisi dengan jumlah simpul sebesar V dan jumlah sisi sebesar E, dapat ditentukan kompleksitas waktu Algortima Dijkstra dengan notasi *Big-O*, yaitu $O(|V|^2 + |E|) = O(|V|^2)$, jika algoritma menyimpan simpul dan sisi dalam bentuk list berkait ataupun array.

Algortima dapat lebih efesien dengan menyimpan graf dalam bentuk adjacency list dan menggunakan binary heap sebagi priority queue. Didapatkan dalam notasi Big-0, yaitu $O((|E| + |V|)\log |V|)$.

Algoritma *Bellman-Ford*

Algortima Bellman-Ford menghitung jarak terpendrek (dari satu sumber) pada sebuah graf berbobot. Berikut merupakan *pseudocode Algoritma Bellman-Ford.*

a. Psudocode Algortima Bellman-Ford

```
record titik {
        list sisi2
        real jarak
        titik sebelum
}
Record sisi {
        titik dari
        titik ke
        real bobot
}

Function BellmanFord(list semuatitik, list semuasisi, titik dari)
// Argumennya ialah graf, dengan bentuk daftar titik
// and sisi. Algoritma ini mengubah titik-titik dalam
// semuatitik sehingga atribut jarak dan sebelum
// menyimpan jarak terpendek.

// Persiapan
for each titik v in semuatitik:
        if v is dari then v.jarak = 0
        else v.jarak := tak-hingga
        v.sebelum := null
```

```
// Perulangan relaksasi sisi
for i from 1 to size(semuatitik):
for each sisi uv in semuasisi:
u := uv.dari
v := uv.ke          // uv adalah sisi dari u ke v
if v.jarak > u.jarak + uv.bobot
v.jarak := u.jarak + uv.bobot
v.sebelum := u

// Cari sirkuit berbobot(jarak) negative
for each sisi uv in semuasisi:
u := uv.dari
v := uv.ke
if v.jarak > u.jarak + uv.bobot
    error "Graph mengandung siklus berbobot
total negatif"
```

b. Kompleksitas Waktu *Algoritma Bellman-Ford*

Secara kompleksitas *waktu Algoritma Bellman-For*d memiliki satu kemungkinan, yaitu dengan notasi *Big-O* diberikan O(|V.E|). V adalah jumlah *vertex* atau simpul dalam graf berbobot, lalu E adalah *edges* yang merupakan jumlah sisi dalam graf. Oleh sebab itu jumlah sisi ataupun simpul yang sangat besar akan menyebabkan algoritma ini akan berjalan lebih lama dibandingkan dengan Algortima Dijkstra. Berikut penjabaran perhitungan kompleksitas wajtu Algoritma Bellman-Ford:

1. Tahap inisialisasi mempunyai kompleksitas O(V).
2. Tahap kedua yaitu melakukan pencarian jalur atau jalan terpendek terhadap suatu simpul s mempunyai kompleksitas O(V.E)
3. Tahap ketiga yaitu pengecekan ada atau tidak sisi negative pada jalur, menpunyai kompleksitas O(E).

Dari semua kompleksitas tersebut dapat ditentukan kompleksitas waktu algoritma ini adalah O(V.E).

Perbedaan Algorima Dijkstra dan Algortima Bellman-Ford

Dari penjelasan Algoritma Dijkstra dan Algortima Bellman-Ford tersebut dapat ditemukan beberapa perbedaan

Algoritma Bellman-Ford dapat melakukan perhitungan jarak antar simpul dengan tepat walaupun ada yang memiliki nilai negatif. Tidak seperti algoritma Dijkstra yang tidak dapat melakukan perhitungan apabila ada sisi yang memuat nilai negatif, algoritma Bellman-Ford dapat menyelesaikannya dengan tepat. Hal yang dilakukan algoritma Bellman-Ford saat menemukan sisi yang memiliki bobot negatif dalam jalur terpendek nya , maka ia akan menegembalikan sebuah pesan *error* jika kita melihat contoh pada pseudocode diatas.

Sudah seharusnya sebuah algoritm dapat menangani segala kemungkinan nilai dalam perhitungan. Sehingga tidaka adanya kerancuan dalam memberikan hasil perhitungan. Sisi negatif tidak ada dalam kehidupan nyata sehingga perlu dikeluarkan dari perhitungan jalur terpendek, namun joika ditemukan sisi negatif, algoritma harus mampu menanganinya.

Pada algoritma Dijkstra, kompleksitas waktu yang dimiliki algoritma ini lebih kecil dibandingkan algoritma Bellman-Ford. Namun dalam kasus-kasus terburuk, kedua algoritma ini dapat memiliki kompleksitas waktu yang sama. Untuk algoritma Bellman-Ford kompleksitas waktu akan selalu O(E.V) untuk segala kemungkinan. Kompleksitas waktu algoritma menjadi sorotan penting ketika waktu program untuk menghitung menjadi perhatian utama. Program yang baik akan mempunyai kompleksitas waktu yang kecil sehingga dapat dengan cepat melakukan perhitungan.

SIMPULAN

1. Algoritma dijkstra *Dijkstra* dan *Bellman-Ford*. Merupakan salah satu algoritma yang digunakan untuk menentukan jalur terpendek. Kedua algoritma ini menggunakan strategi greedy yang cara kerjanya di mana pada setiap langkah dipilih sisi dengan bobot terkecil yang menghubungkan sebuah simpul yang sudah terpilih dengan simpul lain yang belum terpilih.
2. Untuk mengukur kinerja dari setiap algoritma menggunakan empat kerteria yaitu kelengkapan (*Completeness*), optimalias (*optomality*), kompleksitas waktu (*time complexity* dan kompleksitas ruang (*space complexity*). Keteria tersebut untuk menganalisis ferforma dari algoritma untuk menhasilkan solusi yang terbai.
3. Algortima *Dijkstra* dan Algortima *Bellman-Ford* merupakan dua algoritma yang digunakan untuk mencari rute terpendek. Namun tidak seperti Algortima *Dijkstra*, Algoritma *Bellman-Ford* dapat digunakan pada graf yang mengandung simpul negative selama graf tersebut tidak mengandung kalang negative yang dapat dicapai dari titik awal. Algoritma Dijkstra lebih menguntungkan dari sisi run time, namun untuk permasalahan khusus yang mengandung simpul negative Algortima Bellman-Ford lebih menguntungkan.

DAFTAR PUSTAKA

[1]. Thomas H. Cormen, 2009, " Introduction to Algorithm " penertbit MIT, London

[2]. Irwansyah, Edy,2013, " Sistem Informasi Geogerafis: Pinsip-Prinsip Dasar dan

Pengembangan Aplikasi" Penerbit DigiBooks, Yogyakarta
[3]. Lexy Moleong, 2004, " Metodologi Penelitian Kualitatif (Edisi Revisi)" Penerbit Remaja Rosdakarya, Bandung
[4]. Dr. Sudaryono, 2014, " Metodologi Riset di Bidang IT (Panduan Praktis Teori dan Contoh Kasus)", Penerbit Andi Offset, Bandung
[5]. Milles Matthew dan Michael Huberman, 1992, " Analisis Data Kualitatif", Penerbit UI Press, Jakarta
[6]. Abdussakir, 2009, " Teori Graf Topik Dasar Untuk Tugas Akhir dan Skripsi", Penerbit UIN Malang Press, Malang.
[7]. Ummulhadi, 2016, " Rancang Bangun Sistem Informasi Geogerafis Untuk Membantu Pencarian Jalur Terpendek Menuju ATM Bank BRI Dengan Metode Tabu Reseacrh Alghorithm", Jurnal Ilmiah Ilkom Volume 8 Nomor 3.
[8]. Junanda, Berry, 2016, " Pencarian Rute Terpendek Menggunakan Algoritma Dijkstra Pada Sistem Informasi Geogerafis Pemetaan Stasiun Pengisian Bahan Bakar Umum", Jurnal Vokasional Teknik Elektronika dan Informatika, Padang.
[9]. Aji, Agustian P, 2015, " Optimasi Jalur tercepat Dengan Menggunakan Modifikasi Algoritma Bellman-Ford (Studi Kasus Lintas Antar Kecamatan Kota Malang)", Jurnal EECCIS Volume 9 Nomor 2.
[10]. Nasytha Nur Farah, 2014, " Aplikasi Pgrouting Untuk Penentuan Jalur Optimum Pada Pembuatan Rute Pemadam Kebakaran (Studi kasus Kota Semarang), Jurnal Geodesi Undip, Semarang
[11]. Yudi Retanto, 2009, " Algortima Dijkstra dan Algortima Bellman-Ford Dalam Pencarian Jalur Terpendek, Makalah IF20191, Bandung
[12]. Hodia, Maelani, and Khairul Imtihan, 2017, " Perancangan Sisitem Informasi Praktek Klinik Kebidanan (PKK) pada Prodi DIII Kebidanan STIKES Qamarul Huda". IJNS-Indonesian Journal on Networking and Security 6.3.
[13]. Khairul Imtihan, 2015, " Perancangan Strategi Sistem Informasi Pendidikan pada Sekolah Tinggi Manajemen Informatika dan Komputer (STMIK) Lombok". Bianglala Informatika.