PENERAPAN ALGORITMA DIJKSTRA DALAM PENCARIAN SOLUSI MAXIMUM FLOW PROBLEM

Fakhri –NIM : 13506102

Jurusan Teknik Informatika, Sekolah Teknik Elektro dan Informatika ITB
Jalan Ganeca no. 10 Bandung
e-mail: fackhry@gmail.com

ABSTRAK

Makalah ini membahas tentang penerapan algoritma djikstra dalam Maksimum Flow Problem. Maximum Flow Problem merupakan salah satu masalah yang mucul dalam teori graf, atau secara spesifik yaitu Flow Network.
Dalam teori graf, Flow Network merupakan salah satu jenis graf yang banyak diimplementasikan dalam kehidupan sehari-hari, misalnya saja dalam pemodelan system lalu lintas atau pipa air. Masalah-masalah yang sering muncul dalam flow network ini antara lain multi-commodity flow problem, minimum cost problem, circulation problem, dan maximum flow problem.
Algoritma Dijkstra merupakan algoritma paling populer dalam ilmu komputer. Algoritma ini sering digunakan dalam pencarian rute terpendek dalam graf. Konsep dasar algoritma ini mirip dengan algoritma Greedy. Dalam makalah ini, algoritma dijkstra akan coba digunakan untuk menyelesaikan Maximum Flow Problem.

Kata kunci: *Flow-Network, Directed Graph Maximum Flow, sink, source*, algoritma dijkstra.

1. PENDAHULUAN

Teori graf merupakan salah satu cabang matematika yang paling banyak aplikasinya dalam kehidupan sehari-hari. Salah satu bentuk dari graf adalah *flow network,* yaitu graf berarah yang tiap sisinya mempunyai kapasitas tertentu.

Flow network ini memiliki banyak aplikasi dalam kehidupan sehari-hari. *Flow network* sering digunakan untuk memodelkan sistem lalulintas, sebuah system yang sering menjadi masalah utama dalam kehidupan, terutama di kota besar, serta sistem pipa air.

Salah satu masalah yang sering muncul dalam *flow network* adalah *maximum flow problem.* Sebenarnya terdapat banyak metode yang dapat digunakan untuk menyelesaikan masalah ini. Akan tetapi, dalam makalah ini penulis akan mencoba menyelesaikan masalah ini dengan algoritma dijkstra, sebuah algoritma yang bias dibilang jarang digunakan untuk menyelesaikan *Maximum Flow Problem.*

2. ALGORITMA DIJKSTRA

Pada tahun 1959 sebuah tulisan sepanjang tiga halaman yang berjudul *A Note on Two Problems in Connexion with Graphs* diterbitkan pada jurnal *Numerische Mathematik.* Pada tulisan ini, Edsger W. Dijkstra – seorang ilmuwan computer berumur dua puluh sembilan tahun - mengusulkan algoritma-algoritma untuk solusi dari dua masalah teoritis graf dasar: the *minimum weight* Algoritma Dijkstra untuk masalah jalan terpendek adalah satu dari algoritma-algoritma paling ternama pada ilmu komputer dan sebuah algoritma paling popular pada oparasi pencarian(OR). Implementasi algoritma dijkstra pada ilmu computer antara lain adalah pada link-state routing protocol, OSPF dan IS-IS.

Pada literatur tersebut, algoritma ini sering digambarkan sebagai sebuah algoritma yang tamak. Contohnya, buku *Algorithmics* (Brassard and Bratley [1988, pp. 87-92]) mengulas ini pada bab tersebut dengan judul *Greedy Algorithms. Encyclopedia of Operations Research and Management Science* (Gass and Harris [1996, pp. 166-167]) menggambarkan algoritma ini sebagai sebuah "*node labelling greedy algorithm* " dan sebuah algoritma yang tamak digambarkan sebagai "*a heuristic algorithm that at every step selects the best choice available at that step without regard to future consequences* " (Gass and Harris [1996, p. 264]).

2.1 Definisi Algoritma Dijkstra

Pada dasarnya, algoritma ini merupakan salah satu bentuk algoritma *greedy*. Algoritma ini termasuk algoritma pencarian graf yang digunakan untuk menyelesaikan masalah lintasan terpendek dengan satu sumber pada sebuah graf yang tidak memiliki cost sisi negatif, dan menghasilkan sebuah pohon lintasan terpendek. Algoritma ini sering digunakan pada *routing*

Algoritma dijkstra mencari lintasan terpendek dalam sejumlah langkah. Algoritma ini menggunakan strategi *greedy* sebagai berikut :

Untuk setiap simpul sumber(*source*) dalam graf, algortima ini akan mencari jalur dengna cost minimum antara simpul tersebut dengan simpul lainnya. Algoritma ini juga dapat digunakan untuk mencari total biaya(*cost*) dari lintasan terpendek yang dibentuk dari sebuah simpul ke sebuah simpul tujuan. Sebagai contoh, bila simpul pada graf merepresentasikan kota dan bobot sisi merepresentasikan jarak antara 2 kota yang mengapitnya, maka algoritma dijkstra dapat digunakan untuk mencari rute terpendek antara sebuah kota dengan kota lainnya.

2.2 Skema Umum Algoritma Dijkstra

Berikut adalah skema umum dari algoritma dijkstra pada pencarian *shortest path* :

1. Buatlah 3 buah list, yaitu list jarak(list 1), list simpul-simpul sebelumnya(list 2), dan list simpul yang sudah dikunjungi(list 3), serta sebuah variable yang menampung simpul saat ini(*current vertex*).
2. Isi semua nilai dalam list jarak dengan tak hingga, kecuali simpul awal yang diisi dengan 0.
3. Isi semua nilai dalam list 2 dengan *false*
4. Isi semua nilai dalam list 3 dengan *null*
5. *Current Vertex* diisi dengan simpul awal(*start*).
6. Tandai *current vertex* sebagai simpul yang telah dikunjungi.
7. *Update* list 1 dan 2 berdasarkan simpul-simpul yang dapat langsung dicapai dari *current vertex*
8. *Update current vertex* dengan simpul yang paling dekat dengan simpul awal.
9. Ulangi langkah 6 sampai semua simpul sudah dikunjungi.

3. MAXIMUM FLOW PROBLEM

3.1 Network Flow

Dalam teori graf, sebuah *flow network* adalah sebuah graf berarah yang tiap sisinya memiliki kapasitas/bobot dan pada tiap sisi tersebut terdapat arus(*flow*) yang mengalir antara 2 simpul yang mengapit sisi tersebut. Jumlah arus yang mengalir pada tiap sisi harus lebih kecil atau sama dengan kapasitas sisi tersebut.

Pada aplikasinya, sebuah graf berarah sering disebut dengan *network.* Setiap arus(*flow*) yang ada dalam *network,* harus memenuhi sebuah batasan yaitu arus yang masuk pada suatu simpul harus sama dengan arus yang keluar pada simpul tersebut, kecuali pada *source,* yang arus keluarnya lebih besar dari arus masuk, dan *sink,* yang arus masuknya lebih besar dari arus keluar.

Sebuah *network* biasanya digunakan untuk memodelkan sistem lalulintas, saluran pipa, sirkuit elektrik, dsb.

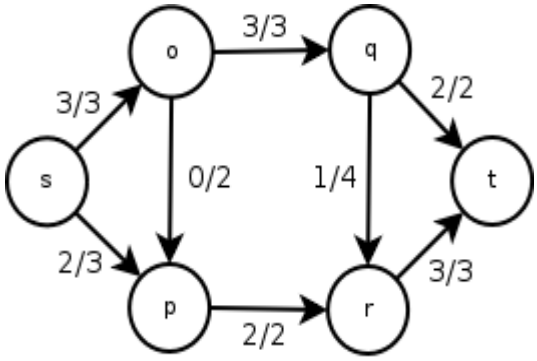
3.2 Maximum Flow Problem

Sebenarnya apa yang dimaksud dengan *Maximum Flow Problem* (permasalahan *max-flow*)? Secara sederhana, *Maximum Flow Problem* dapat dideskripsikan sebagai masalah pencarian untuk mencari arus maksimum yang dapat mengalir pada sebuah *network* yang hanya memiliki sebuah *source* dan sebuah *sink* .

Aplikasi dari *Maximum Flow Problem* ini adalah sebagai berikut : “Terdapat pipa-pipa yang berhubungan, dengan kapasitas / daya tampung yang berbeda – beda. Pipa – pipa ini terhubung dengan sebuah keran, berapa volume maksimum air yang dapat dialirkan dari penampungan air sampai dengan keran dirumah kita ” atau “Sebuah perusahaan memiliki sebuah pabrik di kota x dimana barang yang selesai di produksi harus di kirim ke kota y. Kita memiliki data jalan satu arah yang menghubungkan setiap kota, dan jumlah maksimum truk yang dapat melewati jalan tersebut” Tentukan jumlah maksimum truk yang dapat dikirimkan sekali kirim. Dengan pengamatan pertama, kita dapat menyimpulkan bahwa truk yang masuk ke kota y pasti sama dengan jumlah truk yang keluar dari x.

Merujuk pada teori graf, pada *Maximum Flow Problem* kita diberikan sebuah *network – graph* berbobot dan berarah. Dan disetiap *sisi*nya terdapat kapasitas *c* yang diasosiasikan dengannya, *simpul* awal kita sebut sebagai *source*, dan simpul akhir *sink*. Kita disuruh mencari nilai *f* yang memenuhi persyaratan $f \leq c$ untuk setiap sisi selain *source* dan *sink,* dan jumlah nilai *f* yang masuk kedalam suatu sisi pasti sama dengan jumlah nilai yang meningalkannya. Kemudian kita akan mencari nilai maksimum *f* yang memenuhi persyaratan diatas.

Gambar dibawah ini menunjukkan solusi optimal untuk salah satu permasalahan diatas, setiap *sisi* dilabeli dengan sebuah nilai f/c yang diasosiasikan dengannya :



Gambar 1. Maximum flow pada sebuah flow network

Pada gambar di atas, kita diminta untuk mencari arus maksimal yang dapat mengalir dari simpul s(*source*) ke simpul t(*sink*) melalui beberapa sisi yang masing masing memilik kapasitas tertentu. Dengan melihat gambar di atas maka dapat disimpulkan bahwa arus maksimal yang dapat mengalir pada *network* di atas adalah 5 satuan.

Banyak algoritma yang dapat digunakan untuk menyelesaikan masalah maksimum flow ini. Algoritma yang paling sering digunakan untuk menyelesaikan masalah ini adalah algoritma Ford-Fulkerson, algoritma Edmonds Karp dan algoritma Dinitz blocking flow. Sementara algoritma dijkstra sendiri kurang sering dipakai dalam menyelesaikan masalah ini.

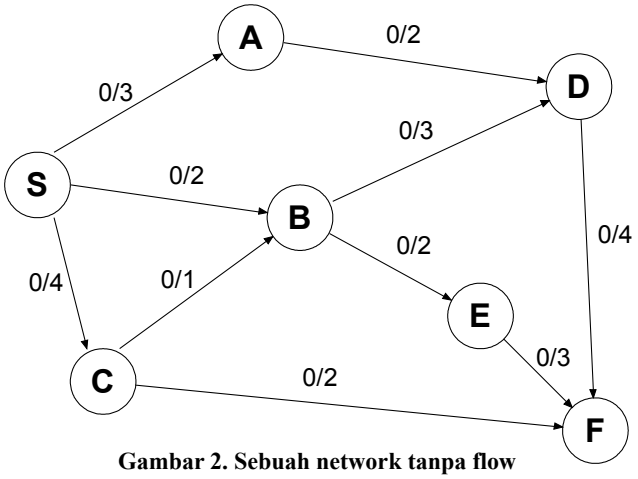
4. PENYELESAIAN MAXIMUM FLOW PROBLEM DENGAN ALGORITMA DIJKSTRA

4.1 Langkah Penyelesaian

Untuk menyelesaikan *Maximum Flow Problem* dengan algoritma Dijkstra, maka kita harus melakukan sedikit modifikasi dari algoritma ini. Secara garis besar, skema dari algoritma yang akan dijelaskan adalah sebagai berikut:

1. Cari sebuah lintasan yang belum dipilih yang menghubungkan simpul awal dengan simpul tujuan.
2. Pada lintasan yang dipilih, carilah sebuah sisi dengan kapasitas sisa minimum. Kapasitas sisa minimum didapat dari kapasitas sisi tersebut dikurangi arus yang sudah mengalir pada sisi itu(*c -f*). Bila kapasitas minimum sisa sama dengan 0, langsung ke langkah 4.
3. Alirkan arus sejumlah kapasitas minimum sisa pada lintasan yang dipilih.
4. Kembali ke langkah 1 sampai semua lintasan diperiksa.

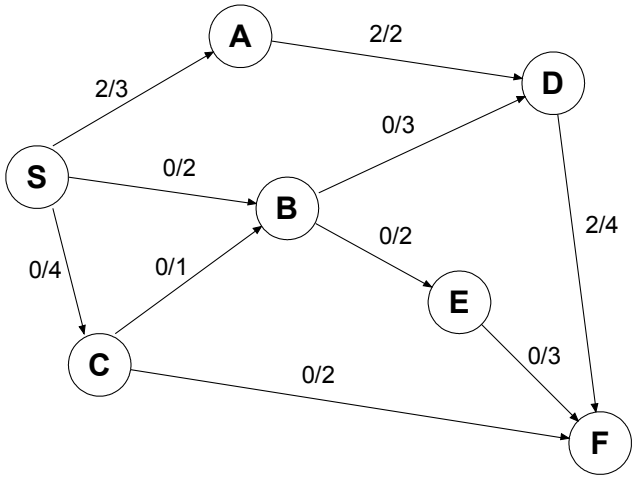
Contoh implementasinya adalah sebagai berikut :



Gambar 2. Sebuah network tanpa flow

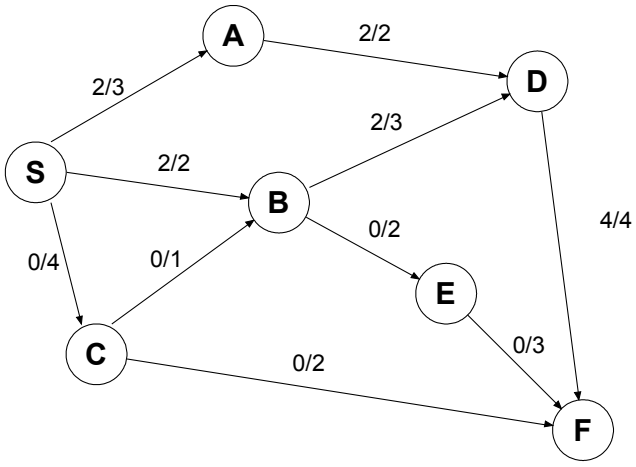
Dalam *network* gambar 2 bisa kita lihat terdapat 5 lintasan yang menghubungkan simpul awal (S) dan simpul tujuan (F).

Langkah 1 : Pilih lintasan S-A-D-F. Pada lintasan ini nilai kapasitas minimum sisanya adalah 2. Alirkan arus sejumlah 2 satuan.



Gambar 3. *Network* hasil Langkah 1

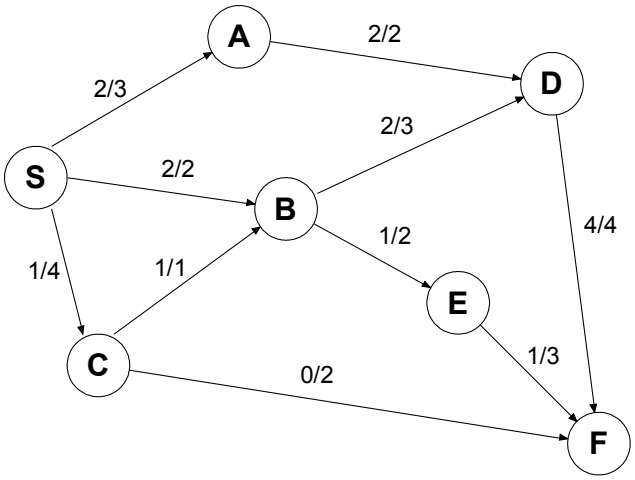
Langkah 2 : Pilih lintasan S-B-D-F. nilai kapasitas sisa minimum pada lintasan ini adalah 2. Alirkan arus sebanyak 2 satuan.



Gambar 4.*Network* hasil Langkah 2

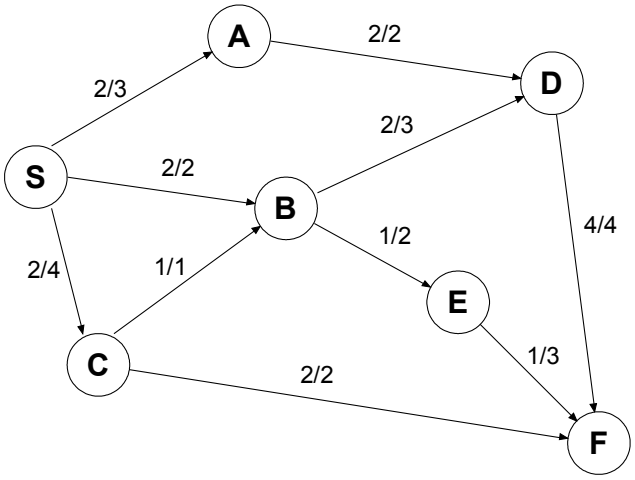
Langkah 3 : pilih lintasan S-B-E-F. lintasan ini sudah tidak dapat dialiri arus lagi karena nilai kapasitasa sisa minimumnya 0.

Langkah 4 : pilih lintasan S-C-B-E-F. Nilai kapasitas sisa minimum pada lintasan ini adalah 1. Alirkan arus sebanyak 1 satuan.



Gambar 5. *Network* hasil Langkah 4

Langkah 5 : pilih lintasan terakhir yaitu S-C-F. Nilai kapasitas sisa minimum pada lintasan ini adalah 2. Alirkan arus sebanyak 2 satuan.



Gambar 6. *Network* hasil Langkah 5

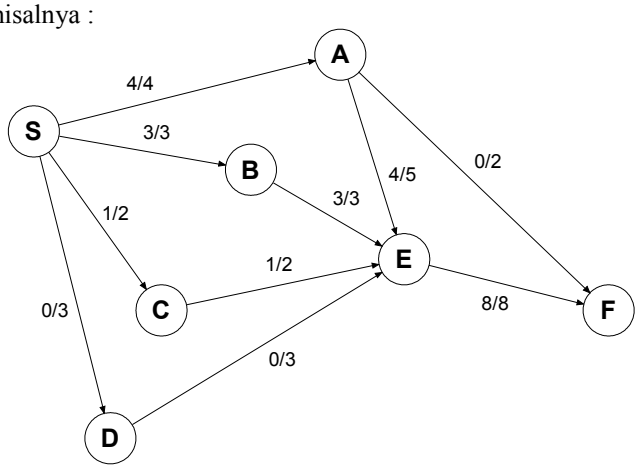
Gambar 6 merupakan solusi dari masalah *maximum flow* pada *network* pada gambar 2. Dari gambar 6 ini dapat dilihat bahwa jumlah arus maksimum yang dapat mengalir dari *network* tersebut adalah 7, yaitu jumlah arus yang meninggalkan simpul awal atau jumlah arus yang masuk ke simpul tujuan.

4.2 Analisis

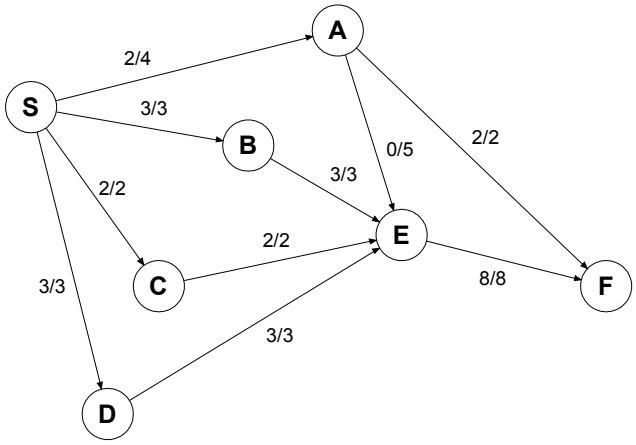
Algoritma dijkstra yang digunakan di atas, atau terkadang disebut Priority First Search(PFS), mampu memberikan solusi optimal untuk kebanyakan kasus, dalam hal ini *Maximum Flow Problem.* Akan tetapi, sama halnya seperti algoritma *greedy* pada umumnya, terdapat beberapa kasus dimana algoritma dijkstra tidak memberikan solusi maksimum.

Hal ini biasanya merupakan akibat dari pemilihan urutan lintasan yang kurang tepat. Karena itu, pemilihan urutan lintasan akan sangat menentukan solusi dari permasalahan ini.

Sama seperti algoritma *greedy,* kita dapat menentukan fungsi prioritas untuk menentukan urutan lintasan yang akan diproses. Biasanya, prioritas utama dalam *maximum flow problem* ini adalah lintasan yang memiliki kapasitas tertinggi. Akan tetapi, dengan fungsi prioritas ini –pun masih ada kasus yang tidak memberikan hasil optimal, misalnya :



Gambar 7.Solusi Maximum Flow Problem dengan algoritma dijkstra



Gambar 8.Solusi Optimal Maximum Flow Problem dari network pada gambar 7

Solusi yang terdapat pada gambar 8 merupakan solusi *Maximum Flow Problem* dengan menggunakan algoritma Ford – Fulkerson, algoritma yang paling sering digunakan untuk menyelesaikan masalah maksimum flow. Algoritma ini ternyata memang lebih baik daripada algoritma dijkstra dalam mencari solusi optimal pada *maximum flow*

problem karena algoritma ini memang dirancang untuk menyelesaikan masalah ini, tidak seperti algoritma dijkstra yang dirancang untuk menyelesaikan masalah *shortest path* secara global.

IV. KESIMPULAN

Untuk kebanyakan masalah, algoritma dijkstra secara umum (tapi tidak selalu) berhasil untuk mencari solusi optimal dalam *maximum flow problem*. Akan tetapi, pada beberapa masalah, algoritma ini gagal memberikan solusi optimal. Hal ini terutama dikarenakan sulit melakukan penentuan urutan lintasan yang akan diperiksa, sama seperti algoritma greedy.

REFERENSI

[1] Munir, Rinaldi. 2006. *Diktat Kuliah IF2251 Strategi Algoritmik*. Penerbit ITB

[2] Agung Santoso, Kiswara. *Metode Simpleks dan Algoritma Dijkstra guna menyelesaikan masalah optimasi.* www.unej.ac.id/fakultas/mipa/majalah_mat/2000/Metode%20Simpleks-**Kiss**.pdf Tanggal Akses : 13 Mei 2008 pukul 21.00

[3] Wikipedia *http://en.wikipedia.org/wiki/Net_flow*. Tanggal akses : 13 Mei 2008 pukul 21.00

[4] Wikipedia *http://en.wikipedia.org/wiki/Maximum_flow_problem*. Tanggal akses : 13 Desember 2008 pukul 21.00

[5] Wikipedia *http://en.wikipedia.org/wiki/Dijkstra_algorithm* Tanggal akses : 14 Mei 2008 pukul 11.00

[6] Maximum Flow Problem www.me.utexas.edu/~jensen/methods/net.pdf/net**max**f.pdf Tanggal akses : 15 Mei 2008 pukul 04.00

[7] Network Flow Algorithm www.leda-tutorial.org/en/unofficial/ch05s03s04.html - 27k Tanggal akses : 15 Mei 2008 pukul 05.00