MENENTUKAN *FLOW* MAKSIMUM PADA JARINGAN BIPARTISI

SKRIPSI

OLEH
NIA CAHYANI
NIM. 11610027

JURUSAN MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2015

MENENTUKAN *FLOW* MAKSIMUM PADA JARINGAN BIPARTISI

SKRIPSI

Diajukan Kepada
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
untuk Memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Sains (S.Si)

Oleh
Nia Cahyani
NIM. 11610027

JURUSAN MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2015

MENENTUKAN *FLOW* MAKSIMUM PADA JARINGAN BIPARTISI

SKRIPSI

Oleh
Nia Cahyani
NIM. 11610027

Telah Diperiksa dan Disetujui untuk Diuji
Tanggal 2015

Pembimbing I,	Pembimbing II,
H. Wahyu H. Irawan, M.Pd	Abdul Aziz, M.Si
NIP. 19710420 200003 1 003	NIP. 19760318 200604 1 002

Mengetahui,
Ketua Jurusan Matematika

Dr. Abdussakir, M.Pd
NIP. 19751006 200312 1 001

MENENTUKAN *FLOW* MAKSIMUM PADA JARINGAN BIPARTISI

SKRIPSI

Oleh
Nia Cahyani
NIM. 11610027

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Sains (S.Si)
Tanggal 26 Juni 2015

Penguji Utama : Drs. H. Turmudi, M.Si .....

Ketua Penguji : Evawati Alisah, M.Pd .....

Sekretaris Penguji : H. Wahyu H. Irawan, M.Pd .....

Anggota Penguji : Abdul Aziz, M.Si .....

Mengetahui,
Ketua Jurusan Matematika

Dr. Abdussakir, M.Pd
NIP. 19751006 200312 1 001

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama : Nia Cahyani

NIM : 11610027

Jurusan : Matematika

Fakultas : Sains dan Teknologi

Judul Skripsi : Menentukan *Flow* Maksimum pada Jaringan Bipartisi

menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambilan data, tulisan atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka. Apabila di kemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 12 Mei 2015
Yang membuat pernyataan,

Nia Cahyani
NIM. 11610027

MOTO

"If you can dream you can do it"

(Walt Disney)

PERSEMBAHAN

Skripsi ini penulis persembahkan untuk:

Kedua orang tua tercinta Ayah Takim dan Bunda Mudayati, kedua saudara penulis terkasih Hendra Ardiansyah dan Erina Cahyani, serta seluruh keluarga besar penulis.

KATA PENGANTAR

Segala puji bagi Alah Swt. atas rahmat, taufik, dan hidayah-Nya sehingga penulis mampu menyelesaikan skripsi yang berjudul "*Menentukan Flow Maksimum pada Jaringan Bipartisi*" ini dengan baik. Shalawat serta salam semoga senantiasa tercurahkan kepada junjungan Nabi Muhammad Saw., yang telah membimbing manusia dari jalan kegelapan menuju jalan yang terang benderang yaitu agama Islam.

Dalam penulisan skripsi ini, penulis banyak mendapat saran, bimbingan, arahan, doa, dan bantuan dari berbagai pihak. Oleh karena itu, penulis sampaikan ucapan terima kasih yang sebesar-besarnya serta penghargaan yang setinggi-tingginya kepada:

1. Prof. Dr. H. Mudjia Rahardjo, M.Si, selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Dr. drh. Bayyinatul Muchtaromah, M.Si, selaku dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Abdussakir, M.Pd, selaku ketua Jurusan Matematika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. H. Wahyu H. Irawan, M.Pd, selaku dosen pembimbing I yang telah banyak memberikan arahan, nasihat, dan motivasi kepada penulis.
5. Abdul Aziz, M.Si, selaku dosen pembimbing II yang telah memberikan saran dan bantuan dalam penulisan skripsi ini.

6. Seluruh dosen Universitas Islam Negeri Maulana Malik Ibrahim Malang khususnya para dosen matematika yang telah memberikan banyak pengalaman dan ilmu kepada penulis.
7. Ayah Takim dan Bunda Mudayati tercinta yang telah mencurahkan kasih sayang, doa, bimbingan, dan motivasi hingga terselesaikannya skripsi ini.
8. Saudara-saudara tersayang yang telah memberikan semangat kepada penulis.
9. Seluruh teman-teman di Jurusan Matematika angkatan 2011, terutama Fahrun Nisa', Anis Mukibatul Badi', Nur Evita Adiningsih, "Keluarga Cikibum", dan "Kos Sunan Ampel 09" sebagai teman yang selalu dapat diandalkan saat penulis mengalami kesulitan dan berjuang bersama-sama untuk meraih mimpi.
10. Keluarga besar Unit Kegiatan Mahasiswa Taekwondo Universitas Islam Negeri Maulana Malik Ibrahim Malang.
11. Semua pihak yang turut membantu selesainya skripsi ini.

Penulis berharap semoga skripsi ini dapat bermanfaat dan menambah wawasan khususnya bagi penulis dan bagi pembaca pada umumnya.

Malang, Mei 2015

Penulis

DAFTAR ISI

DAFTAR GAMBAR

ABSTRAK

Cahyani, Nia. 2015. **Menentukan *Flow* Maksimum pada Jaringan Bipartisi**. Tugas akhir/skripsi. Jurusan Matematika Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) H. Wahyu H. Irawan, M.Pd. (II) Abdul Aziz, M.Si.

Kata Kunci: jaringan, *flow*, jaringan bipartisi, algoritma Ford-Fulkerson

Jaringan merupakan graf berarah terhubung lemah yang mempunyai nilai kapasitas. *Flow* pada jaringan adalah fungsi yang memetakan setiap busur dalam jaringan ke suatu bilangan real non-negatif yang memenuhi:

(i) $0 \leq f(i,j) \leq c(i,j), \forall (i,j) \in \Gamma(N)$ (disebut kapasitas batas)

(ii) $\sum_{(i,j)\epsilon O(s)} f(i,j) = \sum_{(i,j)\epsilon I(t)} f(i,j)$ (disebut nilai *flow* f)

(iii) $\sum_{(i,j)\epsilon O(x)} f(i,j) = \sum_{(i,j)\epsilon I(x)} f(i,j) \forall\ x \in V(N) - \{s,t\}$ (disebut "konservasi *flow*")

Nilai *flow* maksimum pada suatu jaringan dicari dengan menggunakan algoritma Ford-Fulkerson.

Tujuan dari penelitian ini adalah membandingkan nilai *flow* pada jaringan bipartisi yang diperoleh dengan menggunakan algoritma Ford-Fulkerson dengan nilai *flow* yang diperoleh dengan mencari nilai *flow* parsial dari jaringan tersebut dengan tujuan membuat teorema baru.

Untuk menerapkan algoritma Ford-Fulkerson untuk mencari nilai *flow* maksimum pada jaringan bipartisi ditambahkan satu titik sumber yang terhubung ke semua titik sumber yang ada pada jaringan itu, dan satu titik tujuan yang juga terhubung ke semua titik tujuan yang ada pada jaringan tersebut baru menjalankan algoritma Ford-Fulkerson. Sedangkan untuk mencari nilai *flow* dengan membagi jaringan dapat dilakukan dengan membagi jaringan bipartisi menjadi beberapa jaringan berdasarkan banyak titik sumbernya dan mencari nilai *flow* dari masing-masing bagian. Nilai *flow* maksimum yang diperoleh dengan menerapkan algoritma Ford-Fulkerson dibandingkan dengan nilai *flow* maksimum yang diperoleh dengan membagi jaringan tersebut.

Hasil penelitian ini adalah: Nilai *flow* maksimum jaringan bipartisi sama dengan total nilai *flow* maksimum parsialnya.

Bagi penelitian selanjutnya diharapkan dapat menemukan bermacam-macam teorema tentang nilai *flow* pada jaringan bipartisi dengan cara yang lebih mudah.

ABSTRACT

Cahyani, Nia. 2015. **Determine Maximum Flow in Bipartite Networks**. Thesis. Department of Mathematics, Faculty of Science and Technology, State Islamic University of Maulana Malik Ibrahim Malang. Advisors: (I) H. Wahyu H. Irawan, M.Pd. (II) Abdul Aziz, M.Si.

Keyword: network, flow, bipartite network, Ford-Fulkerson algorithm

Network is a weakly connected digraph which has capacity. Flow in a network is a non-negative function defined on the edges that satisfies the following conditions:

(i) $0 \leq f(i,j) \leq c(i,j), \forall (i,j) \in \Gamma(N)$ (it is called bound capacity)

(ii) $\sum_{(i,j)\epsilon O(s)} f(i,j) = \sum_{(i,j)\epsilon I(t)} f(i,j)$ (it is called value of flow f)

(iii) $\sum_{(i,j)\in O(x)} f(i,j) = \sum_{(i,j)\in I(x)} f(i,j) \forall\ x \in V(N) - \{s,t\}$ (it is called "flow conservation")

The maximum flow value of a network is determined using Ford-Fulkerson algorithm.

The purpose of this research is to compare flow value found using Ford-Fulkerson algorithm and flow value found using partial flows to make new theorem of bipartite network flow.

To apply Ford-Fulkerson algorithm for finding maximum flow in bipartite networks, a super-sink and a super-source need to be added to that network. On the other side, to find maximum flow in bipartite networks by splitting the networks become some pieces depend on the number of sources then finding the maximum flow value of each piece. The maximum flow value determined using Ford-Fulkerson algorithm is compared to maximum flow value determined using splitting method.

The result of this research is: Maximum flow of bipartite network is equal to the total of its partial flows

For the next research the writer suggests to find some other theorems about finding maximum flow of bipartite network using easier ways.

ملخص

جهياني، نيا. ٢٠١٥. **تحديد تدفق أقصى من الشبكة الثنائية**. بحث جامعى. شعبة الرياضيات. كلية العلوم و التكنولوجيا. الجامعة الإسلامية الحكومية مولانا مالك إبرهيم مالانج. المشرف: (١) الحج وحيوهنكي إروان الماجستير (٢) عبد العزيز الماجستير

الكلمة الرئيسية: شبكة،تدفق، الشبكةالثنائية، الخوارزمية Ford-Fulkerson

الشبكة هي الرسم البياني الموجهة مرتبطة ضعيفا التي لديها القدرة. تدفق في شبكة هي وظيفة غير السلبية تعرفها على حواف ترضي هذه الشروط:

١. $0 \leq f(i,j) \leq c(i,j), \forall (i,j) \in \Gamma(N)$ (يسمى الحد من قددرة)

٢. $\sum_{(i,j)\epsilon O(s)} f(i,j) = \sum_{(i,j)\epsilon I(t)} f(i,j)$ (يسمى بقيمة التدفق)

٣. $\sum_{(i,j)\epsilon O(x)} f(i,j) = \sum_{(i,j)\epsilon I(x)} f(i,j) \forall\ x \in V(N) - \{s,t\}$ (يسمى الحفظ على تدفق)

قيمة التدفق القصوى من شبكة يتم البحث باستخدام خوارزمية Ford-Fulkerson

والغرض من هذا البحث هو مقارنة قيمة التدفق باستخدام خوارزمية Ford-Fulkerson و قيمة التدفق باستخدام التدفقات الجزئية لجعل نظرية جديدة لتدفق

تطبيق خوارزمية Ford-Fulkerson لإيجاد أقصى تدفق في شبكات ثنائية، وبالوعة السوبر ومصدر سوبر لابد أن يضاف إلى تلك الشبكة. على الجانب الآخر، لإيجاد أقصى تدفق في شبكات ثنائية عن طريق تقسيم الشبكات تصبح بعض القطع تعتمد على عدد من المصادر ثم إيجاد قيمة التدفق القصوى لكل قطعة. تتم مقارنة قيمة التدفق القصوى حصلت باستخدام خوارزمية Ford-Fulkerson إلى أقصى قيمة التدفق حصلت باستخدام طريقة تقسيم.

الشبكة الثنائية.و نتيجة لهذا البحث هي: تدفق الأقصى للشبكة الثنائية يساوي مجموع تدفقاتها جزئية.

للبحث التالي تئمل الكاتبة أن تجد بعض النظريات الأخرى حول إيجاد التدفق الأقصى للشبكة الثنائية بالستخدام طرق أسهل.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Teori graf adalah satu dari beberapa cabang ilmu matematika yang sebenarnya telah ada sejak lebih dari dua ratus tahun yang lalu. Jurnal pertama yang membahas teori graf muncul pada tahun 1736 oleh matematikawan terkemuka dari Swiss yang bernama Euler. Dari perspektif matematika, awalnya teori graf "kurang" signifikan karena pada umumnya digunakan untuk memecahkan teka-teki (*puzzle*), tetapi pada akhirnya mengalami perkembangan yang pesat sekali pada beberapa puluh tahun terakhir ini. Salah satu penyebab berkembangnya teori graf dengan sangat pesat yaitu aplikasinya yang sangat luas dalam kehidupan sehari-hari maupun di dalam berbagai bidang ilmu, misalnya ilmu komputer, teknik, sains, bisnis, dan sosial. Di dalam teori graf juga dibahas tentang *flow* yang penerapannya banyak digunakan khususnya dalam industri, misalnya menentukan penjadwalan, nilai maksimum lintasan terdekat, dan lain-lain (Budayasa, 2007).

Sejauh ini telah banyak buku, jurnal, atau karya tulis lain yang membahas tentang *flow* dan algoritma *flow* maksimum sampai dengan aplikasi *flow* pada jaringan. Di dalam beberapa karya tulis tersebut dijelaskan algoritma *flow* maksimum dari suatu titik sumber ke suatu titik tujuan serta waktu yang diperlukan untuk menemukan nilai *flow* maksimum menggunakan suatu algoritma. Seperti yang telah diteliti oleh Farizal (2013) dalam skripsinya menjelaskan pencarian aliran (*flow*) maksimum suatu jaringan dengan satu sumber

dan satu tujuan dengan menggunakan algoritma Ford-Fulkerson. Dalam karya tulis lain oleh Azar, dkk. (2013), dijelaskan waktu yang diperlukan untuk mencari nilai *flow* maksimum dari suatu jaringan dengan menggunakan berbagai algoritma, akan tetapi tidak dijelaskan bagaimana proses pencarian nilai *flow* maksimum ataupun algoritma itu sendiri.

Budayasa (2007) menjelaskan bahwa untuk suatu jaringan selalu ada suatu *flow* yang maksimum. Kemudian cara membangun suatu *flow* maksimum pada suatu jaringan yaitu menggunakan algoritma Ford-Fulkerson. Selain algoritma Ford-Fulkerson ada juga algoritma MPM, algoritma Edmons Karp, teorema *max flow–min cut* dan algoritma Dinic. Di dalam skripsinya, Thesa Farizal (2013) menjelaskan bahwa hal paling rapi dari algoritma Ford-Fulkerson yaitu selalu memberikan hasil yang benar dalam penyelesaian submasalah dalam pencarian lintasan peningkatan.

Di dalam al-Quran telah dianjurkan Allah untuk bekerjasama dalam kebaikan yang tertulis dalam surat al-Maidah/5:2 berikut.

وَتَعَاوَنُواْ عَلَى ٱلۡبِرِّ وَٱلتَّقۡوَىٰۖ وَلَا تَعَاوَنُواْ عَلَى ٱلۡإِثۡمِ وَٱلۡعُدۡوَٰنِۚ وَٱتَّقُواْ ٱللَّهَۖ إِنَّ ٱللَّهَ شَدِيدُ ٱلۡعِقَابِ ٢

"Dan tolong-menolonglah kamu dalam (mengerjakan) kebajikan dan takwa, dan jangan tolong-menolong dalam berbuat dosa dan pelanggaran. dan bertakwalah kamu kepada Allah, Sesungguhnya Allah amat berat siksa-Nya" (QS. al-Maidah/5:2).

Dari latar belakang di atas, belum pernah dibahas mengenai bagaimana menentukan *flow* maksimum pada jaringan bipartisi dengan cara membagi jaringan bipartisi menjadi beberapa bagian. Penulis merasa penasaran mengenai nilai *flow* pada jaringan bipartisi. Oleh karena itu penulis mengambil judul "Menentukan *Flow* Maksimum pada Jaringan Bipartisi" yang akan menganalisis nilai *flow* pada jaringan bipartisi.

1.2 Rumusan Masalah

Dari latar belakang dapat dirumuskan masalah yang diteliti yaitu bagaimana menentukan *flow* maksimum pada jaringan bipartisi.

1.3 Tujuan Penelitian

Sesuai dengan rumusan masalah maka penelitian ini bertujuan untuk menjelaskan bagaimana menentukan nilai maksimum pada jaringan bipartisi.

1.4 Manfaat Penelitian

Berdasarkan tujuan penelitian maka manfaat penelitian ini dikelompokkan berdasarkan kepentingan beberapa pihak, yaitu:

a. Bagi penulis, sebagai tambahan pengetahuan dan wawasan penelitian teori graf tentang masalah *flow* pada jaringan bipartisi.
b. Bagi mahasiswa, sebagai tambahan pengetahuan tentang *flow* pada jaringan bipartisi.
c. Bagi lembaga, sebagai tambahan literatur yang dapat dijadikan kajian penelitian matematika khususnya tentang teori graf.

1.5 Batasan Masalah

Penulis membatasi pembahasan pada konstruksi algoritma *flow* maksimum pada jaringan bipartisi.

1.6 Metode Penelitian

Jenis penelitian yang digunakan dalam penelitian ini yaitu studi literatur dengan mempelajari berbagai literatur dan mengkaitkannya. Dari studi literatur tersebut diharapkan dapat ditemukan teori baru dari teori-teori lama. Sedangkan pendekatan penelitian ini yaitu pendekatan kualitatif dengan langkah-langkah yang dilakukan sebagai berikut:

1. Menyajikan jaringan (N).
2. Menyelidiki apakah N merupakan jaringan bipartisi, jika N jaringan bipartisi maka lanjut ke langkah 3, namun jika bukan jaringan bipartisi, kembali ke langkah 1. Jika jaringan N bipartisi, maka himpunan titik pada jaringan N yaitu $V(N)$ dapat dibagi menjadi V_1 dan V_2 dengan $|V_1| = m$ dan $|V_2| = n$ sehingga setiap sisi di jaringan N berpangkal di V_1 dan berujung di V_2.
3. Membagi jaringan bipartisi N menjadi m bagian berdasarkan titik sumber, atau menjadi n bagian berdasarkan titik tujuan. Berdasarkan titik sumber dapat dibuat *flow* sebanyak m yaitu $f_1, f_2, f_3, \dots, f_m$. Sedangkan berdasarkan titik tujuan dapat dibuat *flow* sebanyak n yaitu $f_1, f_2, f_3, \dots, f_n$.
4. Membuat *flow* baru dengan satu titik sumber dan satu titik tujuan.
5. Membandingkan *flow* yang diperoleh dari langkah 3 dan 4 dengan tujuan membuat suatu lemma.

1.7 Sistematika Penulisan

Penulisan tugas akhir ini menggunakan sistematika penulisan yang terdiri dari empat bab, dan masing-masing bab dibagi dalam subbab dengan sistematika penulisan sebagai berikut:

Bab I Pendahuluan, meliputi latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, metode penelitian, dan sistematika penulisan.

Bab II Kajian Pustaka, berisi tentang teori yang berhubungan dengan penelitian ini meliputi definisi graf, definisi graf berarah, keterhubungan, jaringan, *flow* maksimum pada jaringan, algoritma *flow* maksimum, titik pemutus, graf bipartisi, jaringan bipartisi, dan kajian konsep *flow* dalam Islam.

Bab III Pembahasan, berisi penjelasan penulis tentang memaksimumkan *flow* pada jaringan bipartisi.

Bab IV Penutup, berisi tentang kesimpulan dari pembahasan serta saran-saran untuk penelitian selanjutnya.

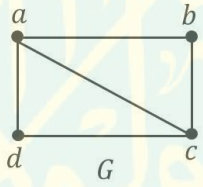
BAB II

KAJIAN PUSTAKA

2.1 Definisi Graf

Pasangan terurut himpunan (V, E) disebut graf jika V merupakan himpunan tak kosong dan berhingga dari unsur-unsur yang dinamakan titik dan E merupakan himpunan (mungkin kosong) pasangan tak terurut titik-titik berbeda yang ada di V yang dinamakan sisi, sehingga tidak terdapat gelung (*loop*) (Chartrand & Lesniak, 1996).

Berikut ini contoh dari graf G.



Gambar 2.1 Graf G

Graf G dalam Gambar 2.1 dapat dinyatakan dengan $G = (V(G), E(G))$ dengan $V(G) = \{a, b, c, d\}$ dan $E(G) = \{ab, ad, ac, bc, cd\}$.

2.1.1 Derajat Titik

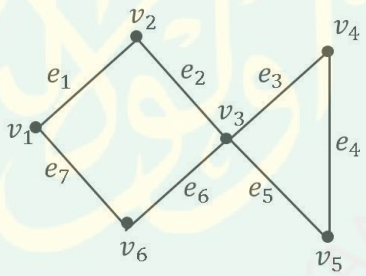
Derajat titik suatu graf adalah banyaknya sisi yang terkait langsung dengan titik tersebut (Budayasa, 2007). Pada Gambar 2.1 derajat titik a adalah 3 karena ada 3 sisi yang terhubung langsung dengan titik a. Derajat titik a adalah 3 ditulis $d(a) = 3$.

2.1.2 Konsep Jalan, Jejak, dan Lintasan

Misalkan G suatu graf. Suatu jalan (*walk*) di G adalah barisan berhingga tak kosong $W = (v_0, e_1, v_1, e_2, \dots, e_k, v_k)$ yang suku-sukunya bergantian titik dan sisi, sedemikian hingga v_{i-1} dan v_i adalah titik-titik akhir sisi e_i untuk $1 \leq i \leq k$. Dikatakan W adalah jalan dari titik v_0 ke v_k atau jalan-(v_o, v_k). Titik v_0 disebut titik awal jalan W dan titik v_k disebut titik akhir jalan W (Budayasa, 2007).

Titik di G boleh muncul lebih dari sekali dalam jalan W, begitu juga dengan sisi di G, boleh muncul lebih dari sekali di jalan W. Jika semua sisi dalam jalan W berbeda, maka W disebut jejak (*trail*). Jika semua titik dan sisi dalam jalan W berbeda, maka W disebut lintasan (*path*) (Budayasa, 2007).

Contoh jalan, jejak, dan lintasan seperti berikut.



Gambar 2.2 Jalan *(Walk)* v_1-v_5 dan Lintasan *(Path)* v_1-v_5

Dari Gambar 2.2 didapatkan suatu jalan dari v_1 sampai v_5 urutannya adalah $(v_1, e_7, v_6, e_6, v_3, e_3, v_4, e_4, v_5)$. Sedangkan suatu lintasan dari v_1 sampai v_5 urutannya adalah $(v_1, e_1, v_2, e_2, v_3, e_5, v_5)$.

2.2 Definisi Graf Berarah

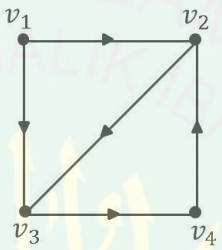
Suatu graf berarah D yaitu pasangan terurut dari dua himpunan $V(D)$ dan $\Gamma(D)$, dengan $V(D)$ himpunan berhingga dan tak kosong yang unsur-unsurnya

disebut titik dan $\Gamma(D)$ himpunan berhingga (mungkin kosong) yang unsur-unsurnya disebut busur dengan setiap busur adalah pasangan terurut dari dua titik di $V(D)$ (Budayasa, 2007).

Contoh graf berarah D yaitu

$$D = (\{v_1, v_2, v_3, v_4\}, \{(v_1, v_2), (v_1, v_3), (v_3, v_4), (v_4, v_2), (v_2, v_3)\})$$

dan dapat diilustrasikan seperti gambar berikut.



Gambar 2.3 Graf Berarah D

2.2.1 Derajat Titik pada Graf Berarah

Misalkan D suatu graf berarah dan $v \in V(D)$. Derajat keluar titik v, dilambangkan $od(v)$, adalah banyaknya busur pada graf berarah D yang keluar dari titik v. Sedangkan derajat masuk titik v dilambangkan $id(v)$, adalah banyaknya busur D yang menuju ke titik v (Ruohonen, 2013). Sebagai contoh, dari graf berarah D pada Gambar 2.3, diperoleh $od(v_1) = 2;\ id(v_1) = 0;$ $od(v_2) = 1; id(v_2) = 2; od(v_3) = 1; id(v_3) = 2; od(v_4) = 1; id(v_4) = 1.$

Setiap busur dari suatu graf berarah dihitung tepat satu kali dalam menghitung jumlah derajat keluar semua titik. Begitu juga setiap busur dihitung tepat satu kali dalam menghitung jumlah derajat masuk semua titik, sehingga diperoleh teorema berikut (Budayasa, 2007).

Teorema 2.1:

Jika $D = (V(D), \Gamma(D))$ graf berarah, maka

$$\sum_{v \in V(D)} id(v) = |\Gamma(D)| = \sum_{v \in V(D)} od(v)$$

Bukti:

Setiap menghitung derajat keluar suatu titik, maka masing-masing busur dihitung tepat satu kali karena setiap busur terkait langsung dari tepat satu titik. Demikian juga setiap menghitung derajat masuk suatu titik, maka masing-masing busur dihitung tepat satu kali karena setiap busur terkait langsung ke tepat satu titik (Abdussakir, dkk, 2009).

2.2.2 Konsep Jalan, Jejak, dan Lintasan Berarah

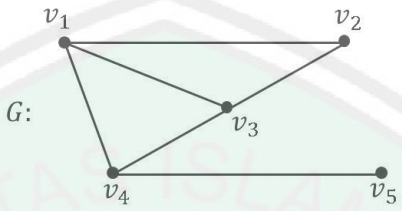
Misalkan D graf graf berarah. Suatu jalan (*walk*) di D adalah barisan berhingga tak kosong $W = (v_0, e_1, v_1, e_2, \dots, e_k, v_k)$ yang suku-sukunya bergantian titik dan busur, sedemikian hingga v_{i-1} dan v_i adalah titik pangkal dan titik ujung busur e_i untuk $1 \leq i \leq k$. Dikatakan W adalah jalan dari titik v_0 ke v_k atau jalan-(v_o, v_k) (Ruohonen, 2013). Jika semua busur dalam jalan W berbeda, maka W disebut jejak (*trail*). Jika semua titik dan busur dalam jalan W berbeda, maka W disebut lintasan (*path*) (Budayasa, 2007).

2.3 Keterhubungan

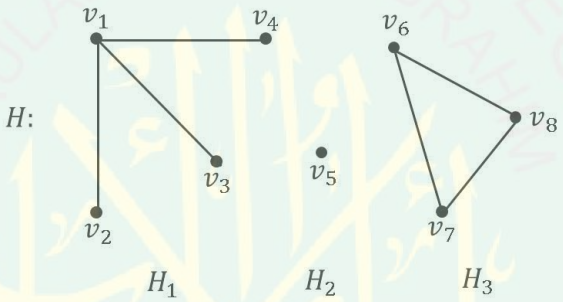
Graf dikatakan terhubung jika untuk setiap pasangan $\{x, y\}$ titik berbeda terdapat lintasan dari x ke y. Sebagai catatan, bahwa graf terhubung dengan *order*

(banyak titik) paling sedikit 2 tidak dapat memiliki titik terasing. Subgraf terhubung maksimal adalah komponen suatu graf (Bollobas, 1998).

Contoh graf terhubung dan tak terhubung seperti berikut.



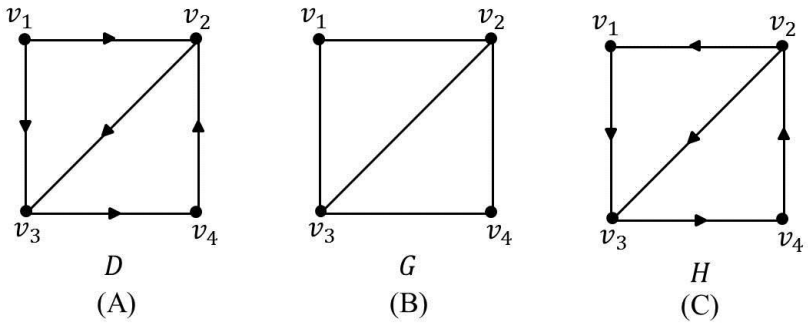
Gambar 2.4 Graf Terhubung G



Gambar 2.5 Graf Tak Terhubung H dengan Komponen H_1, H_2, dan H_3

Ada dua macam keterhubungan pada graf berarah, yaitu terhubung kuat dan terhubung lemah. Graf berarah dikatakan terhubung lemah jika graf dasarnya terhubung, dan akan dikatakan terhubung kuat jika untuk setiap dua titik v_i dan v_j dalam graf berarah terdapat lintasan berarah dari v_i ke v_j (Budayasa, 2007).

Berikut adalah contoh keterhubungan pada graf berarah.



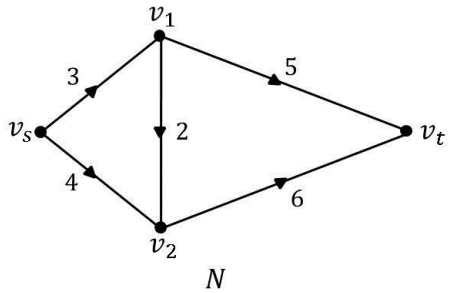
Gambar 2.6 (A) D Graf Berarah Terhubung Lemah; (B) G adalah Graf Dasar dari Graf Berarah D dan H; (C) H Graf Berarah Terhubung Kuat

Graf berarah D pada Gambar 2.6 (A) adalah graf terhubung lemah karena graf dasarnya, yaitu graf G pada Gambar 2.6 (B) terhubung. Karena tidak ada lintasan berarah dari v_4 ke v_1 pada graf berarah D, maka graf D terhubung lemah. Sedangkan graf berarah H pada Gambar 2.6 (C) adalah graf berarah terhubung kuat (Budayasa, 2007).

2.4 Jaringan (*Network*)

Jaringan (*network*) $N = (V(N), \Gamma(N))$ yaitu graf berarah sederhana terhubung lemah yang setiap unsurnya dikaitkan dengan bilangan real non negatif. Kemudian, bilangan real non negatif yang dikaitkan dengan busur (v_i, v_j) atau disingkat (i, j) pada jaringan N dinamakan kapasitas busur (v_i, v_j) dan dinotasikan $c(v_i, v_j)$ boleh disingkat $c(i, j)$. Suatu titik s pada jaringan N dinamakan titik sumber jika $id(s) = 0$, artinya tidak ada busur yang mengarah ke titik s. Sedangkan titik t di jaringan N dinamakan titik tujuan jika $od(t) = 0$ yang artinya tidak ada busur yang keluar dari t. Titik-titik selain s dan t pada jaringan N disebut titik antara (Budayasa, 2007).

Contoh jaringan N dengan titik sumber v_s dan titik tujuan v_t pada gambar berikut.



Gambar 2.7 Jaringan N dengan Titik Sumber v_s dan Titik Tujuan v_t

Asumsikan X dan Y dua himpunan bagian $V(N)$ pada jaringan N. Himpunan semua busur N yang berawal X dan berujung di Y dilambangkan $B(X,Y)$. Total kapasitas semua busur di $B(X,Y)$ dilambangkan $c(X,Y)$ didefinisikan dengan

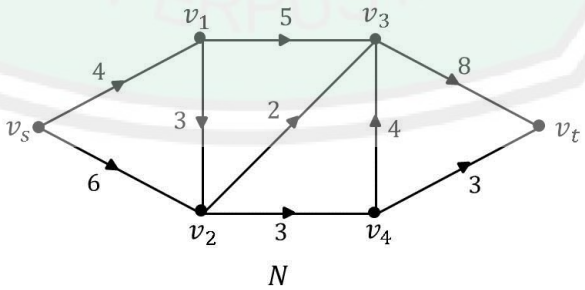
$$c(X,Y) = \sum_{a \epsilon B(X,Y)} c(a).$$

2.4.1 Pemutus pada Jaringan

Misal N jaringan dengan titik sumber s dan titik tujuan t. Misal X adalah himpunan bagian tak kosong dari $V(N)$ dan $X_1 = V(N) - X$. Jika $s \in X$ dan $t \in X_1$, maka himpunan busur $B(X, X_1)$ dinamakan pemutus-(s,t) dari jaringan N jika dengan penghapusan semua busur $B(X, X_1)$ memutus semua lintasan berarah dari s ke t pada jaringan N (Budayasa, 2007).

2.4.2 Konsep *Flow* pada Jaringan

Misalkan N jaringan dengan titik sumber s dan titik tujuan t. Apabila v merupakan titik di N, maka semua busur N yang meninggalkan v disimbolkan $O(v)$ dan himpunan semua busur yang mengarah ke v disimbolkan $I(v)$.



N

Gambar 2.8 Jaringan N dengan Titik Sumber v_s dan Titik Tujuan v_t

Flow pada jaringan N dari titik s ke t yaitu suatu fungsi yang memetakan setiap busur (i,j) pada N ke bilangan real non negatif yang memenuhi syarat-syarat sebagai berikut:

(i) $0 \leq f(i,j) \leq c(i,j), \forall (i,j) \in \Gamma(N)$ (disebut kapasitas batas)

(ii) $\sum_{(i,j)\epsilon O(s)} f(i,j) = \sum_{(i,j)\epsilon I(t)} f(i,j)$ (disebut nilai *flow* f)

(iii) $\sum_{(i,j)\in O(x)} f(i,j) = \sum_{(i,j)\in I(x)} f(i,j), \forall\, x \in V(N) - \{s,t\}$ (disebut konservasi *flow*)

Syarat (i) menyatakan nilai *flow* pada setiap busur tidak pernah melebihi kapasitas busur tersebut. Syarat (ii) menyatakan bahwa total nilai *flow* keluar dari titik sumber sama dengan total nilai *flow* yang masuk ke titik tujuan. Syarat (iii) menyatakan untuk setiap titik antara pada N berlaku total *flow* yang menuju titik antara tersebut sama dengan nilai total *flow* yang meninggalkan titik tersebut (Budayasa, 2007).

Teorema 2.2:

Misalkan N jaringan dengan titik sumber s dan titik tujuan t. Jika f adalah *flow* dari s ke t pada N dengan nilai $f_{s,t}$ dan $B(X, X_1)$ suatu pemutus-(s,t) pada N, maka $f_{s,t} = f(X, X_1) - f(X_1, X) \leq c(X, X_1)$.

Bukti:

Dari definisi *flow*, untuk sumber s diperoleh

$$f(\{s\}, V) - f(V, \{s\}) = f_{s,t}$$

dan untuk setiap titik $x \in V - \{s,t\}$, diperoleh

$$f(\{x\}, V) = \sum_{(i,j)\in O(x)} f(i,j) = \sum_{(i,j)\in I(x)} f(i,j) = f(V, \{x\})$$

atau

$$f(\{x\},V) - f(V,\{x\}) = 0.$$

Sehingga untuk suatu pemutus-(s,t) pada $B(X,X_1)$ diperoleh

$$f(X,V) - f(V,X) = \sum_{x\in X} [f(\{x\},V) - f(V,\{x\})]$$

$$= [f(\{s\},V) - f(V,\{s\})] + 0$$

$$= f_{s,t} \quad (1)$$

Sementara itu

$$f(X,V) - f(V,X) = f(X,X\cup X_1) - f(X\cup X_1,X)$$

$$= f(X,X) + f(X,X_1) - \{f(X,X) + f(X_1,X)\}$$

$$= f(X,X_1) - f(X_1,X) \quad (2)$$

Karena nilai *flow* pada setiap busur non negatif, maka $f(X_1,X) \geq 0$, sehingga

$$f(X,X_1) - f(X_1,X) \leq f(X,X_1) \quad (3)$$

Selanjutnya, karena nilai *flow* pada setiap busur N tidak melebihi kapasitas busur, maka

$$f(X,X_1) \leq c(X,X_1) \quad (4)$$

Dari $(1),(2),(3)$, dan (4) disimpulkan

$$f_{s,t} = f(X,X_1) - f(X_1,X) \leq f(X,X_1) \leq c(X,X_1).$$

Dengan demikian bukti teorema lengkap (Budayasa, 2007).

2.5 *Flow* Maksimum pada Jaringan

Teorema 2.2 menjamin bahwa nilai sebarang *flow* pada jaringan N dari titik sumber s ke titik tujuan t, tidak akan melebihi kapasitas sebarang pemutus-(s,t). Jadi sebesar-besarnya nilai *flow* tidak akan melebihi sekecil-kecilnya nilai kapasitas pemutus-(s,t). Sehingga jika terdapat *flow* f pada N yang nilainya sama

dengan kapasitas pemutus-(s,t), maka *flow* f adalah *flow* maksimum dan pemutus-(s,t) tersebut adalah pemutus minimum. Jadi *flow* f bernilai $f_{s,t}$ dari titik sumber s ke tujuan t pada jaringan N dikatakan *flow* maksimum jika

$$f_{s,t} = \min\{c(X, X_1) | B(X, X_1) \text{ suatu pemutus} - (s,t) \text{ pada jaringan } N\}$$

(Budayasa, 2007).

Misalkan f adalah *flow* dari titik sumber s ke titik tujuan t pada jaringan N, dan G adalah graf dasar N. Misalkan $P = (v_1, v_2, \dots, v_i, v_{i+1}, \dots, v_n)$ lintasan pada G. Jika (v_i, v_{i+1}) suatu busur pada N, maka busur tersebut dinamakan busur maju. Jika (v_{i+1}, v_i) suatu busur pada N, maka busur tersebut dinamakan busur mundur. Inkremen busur a pada N yang berkorespondensi dengan sisi P pada G, dilambangkan dengan $i(a)$ yang didefinisikan sebagai $i(a) = c(a) - f(a)$, jika a busur maju, dan $i(a) = f(a)$ jika a busur balik. Sedangkan inkremen lintasan P dilambangkan $i(P)$ didefinisikan sebagai $i(P) = \min(i(a))$ (Budayasa, 2007).

2.6 Algoritma *Flow* Maksimum pada Jaringan

Untuk jaringan selalu ada *flow* yang maksimum. Kemudian untuk membangun *flow* maksimum pada jaringan digunakan algoritma Ford-Fulkerson. Ide utama dari algoritma ini sebagai berikut:

1. Dimulai dengan mengkonstruksi *flow* f sebarang pada jaringan N dengan titik sumber s dan titik tujuan t. Hal ini selalu dapat dilakukan, misalnya dimulai dengan *flow* nol yaitu *flow* yang mengaitkan setiap busur N dengan bilangan nol.
2. Kemudian, berpijak pada *flow* f tersebut, dicari lintasan P dari titik s ke titik t pada graf dasar G dari N yang inkremennya positif. Jika tidak ada lintasan

yang demikian, maka proses dihentikan dan *flow* f adalah *flow* maksimum di N. Jika ditemukan lintasan P yang demikian, lanjut ke langkah 3.

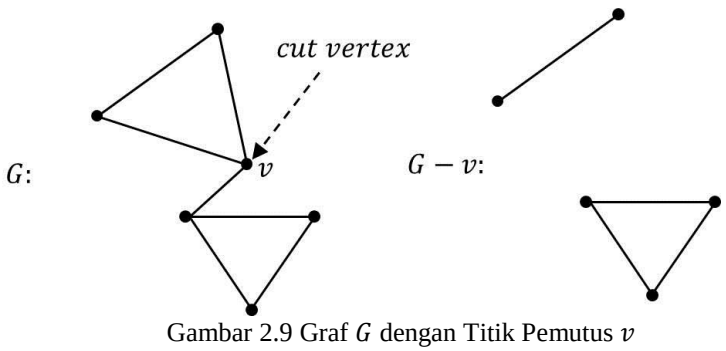
3. Konstruksi *flow* baru, katakan f_1, dari *flow* f berdasarkan lintasan P tersebut. Dalam hal ini nilai *flow* f_1 lebih besar dari pada nilai *flow* f. Ganti *flow* f dengan *flow* f_1, kemudian kembali ke langkah 2.

Walaupun prosedur untuk mengkonstruksi *flow* baru yang nilainya lebih besar dari nilai *flow* yang sama, namun prosedur untuk mendapatkan lintasan peningkatan P tidak tersirat dalam lemma maupun bukti teorema sebelumnya. Untuk mendapatkan lintasan P yang demikian yang akan digunakan teknik pelabelan titik, yang pada prinsipnya melabeli titik-titik N dengan teknik tertentu dimulai titik s, kemudian dilanjutkan melabeli titik yang lain. Jika dengan teknik tersebut dapat melabeli titik t, maka dengan prosedur balik lintasan P ditemukan. Tetapi sebaliknya, jika tidak dapat melabeli titik t, maka tidak ada lintasan P seperti itu pada N (Budayasa, 2007).

2.7 Titik Pemutus

Titik v pada graf G adalah titik pemutus (*cut vertex*) jika graf $G - v$ mempunyai banyak komponen yang melebihi komponen G (Ruohonen, 2013).

Contoh titik pemutus v pada graf seperti berikut.



Gambar 2.9 Graf G dengan Titik Pemutus v

Graf dikatakan *separable* atau dapat dipisah, jika ada setidaknya satu titik pemutus dalam graf tersebut. Sebaliknya, graf tersebut dikatakan *non separable*.

Teorema 2.3:

Titik v adalah titik pemutus pada graf terhubung G jika dan hanya jika ada dua titik u dan w sehingga $u \neq v, v \neq w$ dan $u \neq w$ tetapi v pada setiap lintasan $u - w$.

Bukti:

Diketahui v merupakan titik pemutus di G. Maka, $G - v$ tak terhubung dan terdapat paling sedikit dua komponen yaitu $G_1 = (V_1, E_1)$ dan $G_2 = (V_2, E_2)$. Dipilih $u \in V_1$ dan $w \in V_2$. Lintasan $u - w$ di G karena G terhubung. Andaikan v tidak pada lintasan $u - w$ maka lintasan tersebut juga di $G - v$. Jika lintasan $u - w$ pada $G - v$, artinya u dan w berada dalam satu komponen di $G - v$ yang artinya $G - v$ terhubung. Ini kontradiksi dengan pernyataan awal bahwa v titik pemutus yang menyebabkan $G - v$ tak terhubung. Sehingga pengandaian salah dan v berada pada setiap lintasan $u - w$.

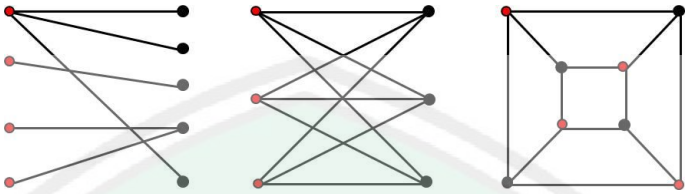
Sebaliknya, jika diketahui $u \neq w \neq v$ di $V(G)$ dan v berada pada setiap lintasan $u - w$. Maka, penghapusan v memotong semua lintasan $u - w$ dan menjadikan graf G tak terhubung. Sehingga v adalah titik pemutus pada G.

Dengan demikian bukti Teorema 2.3 telah lengkap.

2.8 Graf Bipartisi

Graf G disebut k-partisi, $k > 1$, jika $V(G)$ dapat dipartisi menjadi k himpunan bagian $V_1, V_2, \ldots, V_k$ (disebut himpunan partisi) sehingga setiap anggota

$E(G)$ menghubungkan titik di V_i ke titik di V_j, $i \neq j$. Untuk $k = 2$ disebut bipartisi. Contoh graf bipartisi seperti berikut.

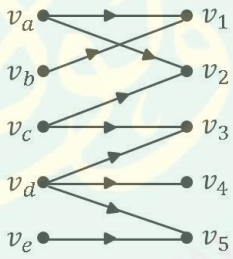


Gambar 2.10 Contoh Graf Bipartisi

2.9 Jaringan Bipartisi

Jaringan $G = (V, E)$ dikatakan bipartisi jika himpunan titik V dapat dipartisi menjadi dua himpunan bagian V_1 dan V_2 sehingga semua sisi dari G mempunyai ujung di V_1 dan ujung lainnya di V_2 (Ahuja, dkk,1994).

Contoh jaringan bipartisi seperti berikut.



Gambar 2.11 Contoh Jaringan Bipartisi

2.10 Kajian Konsep *Flow* dalam Islam

Nilai *flow* tidak pernah melebihi kapasitas busur, dalam al-Quran telah dijelaskan bahwa ujian dari Allah tidak pernah melebihi kemampuan manusia.

لَا يُكَلِّفُ ٱللَّهُ نَفْسًا إِلَّا وُسْعَهَا ۚ لَهَا مَا كَسَبَتْ وَعَلَيْهَا مَا ٱكْتَسَبَتْ ۗ رَبَّنَا لَا تُؤَاخِذْنَآ إِن نَّسِينَآ أَوْ أَخْطَأْنَا ۚ رَبَّنَا وَلَا تَحْمِلْ عَلَيْنَآ إِصْرًا كَمَا حَمَلْتَهُۥ عَلَى ٱلَّذِينَ مِن قَبْلِنَا ۚ رَبَّنَا وَلَا تُحَمِّلْنَا مَا لَا طَاقَةَ لَنَا بِهِۦ ۖ وَٱعْفُ عَنَّا وَٱغْفِرْ لَنَا وَٱرْحَمْنَآ ۚ أَنتَ مَوْلَىٰنَا فَٱنصُرْنَا عَلَى ٱلْقَوْمِ ٱلْكَٰفِرِينَ ﴿٢٨٦﴾

"Allah tidak membebani seseorang melainkan sesuai dengan kesanggupannya. Ia mendapat pahala (dari kebajikan) yang diusahakannya dan ia mendapat siksa (dari kejahatan) yang dikerjakannya. (Mereka berdoa): "Ya Tuhan kami, janganlah Engkau hukum kami jika kami lupa atau kami tersalah. Ya Tuhan kami, janganlah Engkau bebankan kepada kami beban yang berat sebagaimana Engkau bebankan kepada orang-orang sebelum kami. Ya Tuhan kami, janganlah Engkau pikulkan kepada kami apa yang tak sanggup kami memikulnya. Beri ma'aflah kami, ampunilah kami dan rahmatilah kami. Engkaulah penolong kami, Maka tolonglah kami terhadap kaum yang kafir" (QS. al-Baqarah/2:286).

Sedangkan maksud dari ayat tersebut menurut Tafsir Jalalain yaitu (Allah tidaklah membebani seseorang melainkan sesuai dengan kemampuannya), artinya sekadar kesanggupannya. (Ia mendapat dari apa yang diusahakannya) berupa kebaikan artinya pahalanya (dan ia beroleh pula dari hasil kejahatannya), yakni dosanya. Maka seseorang itu tidaklah menerima hukuman dari apa yang tidak dilakukannya, hanya baru menjadi angan-angan dan lamunan mereka. Mereka bermohon, ("Wahai Tuhan kami! Janganlah kami dihukum) dengan siksa (jika kami lupa atau tersalah), artinya meninggalkan kebenaran tanpa sengaja, sebagaimana dihukumnya orang-orang sebelum kami. Sebenarnya hal ini telah dicabut Allah terhadap umat ini, sebagaimana yang telah dijelaskan oleh hadits. Permintaan ini merupakan pengakuan terhadap nikmat Allah. (Wahai Tuhan kami! Janganlah Engkau bebankan kepada kami beban yang berat) yang tidak mungkin dapat kami pikul (sebagaimana Engkau bebankan kepada orang-orang yang sebelum kami), yaitu Bani Israel berupa bunuh diri dalam bertaubat, mengeluarkan seperempat harta dalam zakat dan mengorek tempat yang terkena najis. (Wahai Tuhan kami! Janganlah Engkau pikulkan kepada kami apa yang tidak sanggup) atau tidak kuat (kami memikulnya) berupa tugas-tugas dan cobaan-cobaan. (Beri maaflah kami) atau hapuslah sekalian dosa kami (ampunilah kami dan beri rahmatlah kami) dalam rahmat itu terdapat kelanjutan atau

tambahan keampunan, (Engkaulah pembela kami), artinya pemimpin dan pengatur urusan kami (maka tolonglah kami terhadap orang-orang yang kafir."), yakni dengan menegakkan *hujah* dan memberikan kemenangan dalam peraturan dan pertempuran dengan mereka, karena ciri-ciri seorang *maula* atau pembela adalah menolong anak buahnya terhadap musuh-musuh mereka. Dalam hadits tercantum bahwa tatkala ayat ini turun dan dibaca oleh Nabi Saw., maka setiap kalimat diberikan jawaban oleh Allah Swt., "telah Aku penuhi!" (As-Syuyuti & Muhammad, 2010).

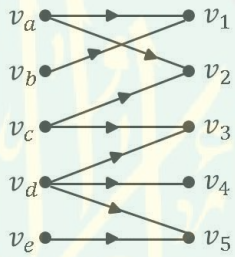
BAB III

PEMBAHASAN

3.1 Jaringan Bipartisi

Jaringan $G = (V, E)$ dikatakan bipartisi jika himpunan titik V dapat dipartisi menjadi dua himpunan bagian V_1 dan V_2 sehingga semua sisi dari G mempunyai ujung di V_1 dan ujung lainnya di V_2 (Ahuja, dkk, 1994).

Contoh jaringan bipartisi seperti berikut.



Gambar 3.1 Jaringan Bipartisi

Dari definisi jaringan bipartisi tersebut, jelas tidak menutup kemungkinan bahwa jaringan bipartisi mempunyai beberapa titik sumber dan beberapa titik tujuan. Untuk mencari nilai *flow* dari jaringan yang memiliki beberapa titik sumber dan beberapa titik tujuan, pada pembahasan ini dibuat algoritma untuk mencari nilai *flow* maksimum jaringan dengan beberapa titik sumber dan beberapa titik tujuan dengan memodifikasi algoritma Ford-Fulkerson.

3.2 Konstruksi Algoritma *Flow* Maksimum pada Jaringan dengan Beberapa Titik Sumber dan Beberapa Titik Tujuan

Pada uraian berikut dijelaskan tentang algoritma *flow* yang nantinya digunakan untuk mencari *flow* maksimum pada suatu jaringan yang memiliki beberapa titik sumber dan beberapa titik tujuan. Hal ini dilakukan karena kasus

jaringan bipartisi dapat memiliki beberapa sumber dan beberapa tujuan. Berdasarkan algoritma *flow* maksimum yang telah diuraikan pada Subbab 2.6, maka penulis menjelaskan algoritma *flow* yang baru. Misalkan jaringan N merupakan suatu jaringan yang memiliki beberapa titik sumber dan beberapa titik tujuan maka algoritma *flow* maksimum pada suatu jaringan dengan beberapa titik sumber dan beberapa titik tujuan adalah sebagai berikut:

Input : Jaringan $N = (V, G)$ dengan beberapa titik sumber (s_i) dan beberapa titik tujuan $t(t_j)$.

Langkah 1 : Dibentuk jaringan baru dari N yang dimisalkan N^*, dengan menambahkan satu titik sumber s dan satu titik tujuan t sedemikian hingga s merupakan satu-satunya titik sumber di jaringan N^* dan t merupakan satu-satunya titik tujuan di N^*.

Langkah 2 : Dibuat kapasitas busur (s, s_i) dan (t_i, t) dengan nilai kapasitas infinit atau dilambangkan ∞, dan nilai *flow* dapat dimulai dari 0.

Langkah selanjutnya yaitu memaksimumkan *flow* $s - t$ dengan menggunakan algoritma *flow* maksimum yaitu:

Langkah 3 : Dilakukan rutin pelabelan.

Langkah 4 : Digunakan prosedur balik.

Langkah 5 : Dilakukan rutin peningkatan.

Langkah 6 : Apabila titik-titik di N yang terlabeli telah teramati semua, dan titik t tidak terlabeli maka iterasi dihentikan.

Output : Diperoleh nilai *flow* $f_{s,t}$ dengan $f_{s,t}$ adalah *flow* maksimum dari s ke t di jaringan N.

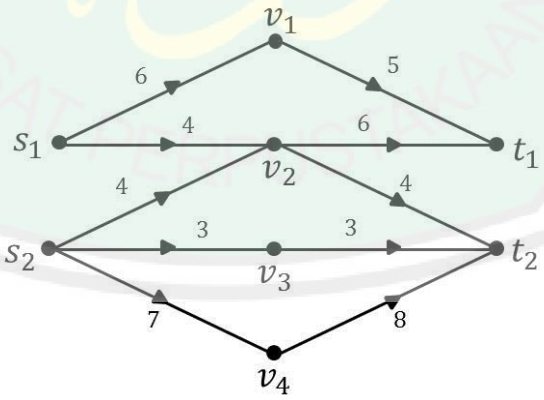
3.2.1 *Flow* Maksimum dengan Beberapa Titik Sumber dan Beberapa Titik Tujuan

Penulis memberikan contoh data untuk permasalahan *flow* maksimum dengan dua titik sumber dan dua titik tujuan ditunjukkan pada tabel berikut.

Tabel 3.1 Data Permasalahan *Flow* Maksimum dengan Titik Sumber, Titik Antara, dan Titik Tujuan

Sumber	Antara			Tujuan		
	Titik Antara	Kapasitas	*Flow*	Titik Tujuan	Kapasitas	*Flow*
s_1	v_1	6	5	t_1	5	5
s_1	v_2	4	4	t_1	6	4
s_2	v_2	4	2	t_1	6	2
s_2	v_2	4	2	t_2	4	2
s_2	v_3	3	3	t_2	3	3
s_2	v_4	7	7	t_2	8	7

Selanjutnya dibentuk suatu jaringan yang akan ditentukan *flow* maksimum dari titik-titik sumber s_1 dan s_2 ke titik-titik tujuan t_1 dan t_2 berdasarkan Tabel 3.1 seperti berikut.

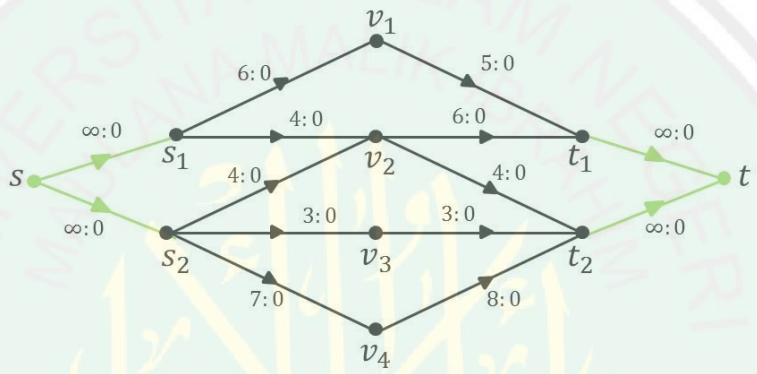


Gambar 3.2 Jaringan *N* dengan Dua Titik Sumber dan Dua Titik Tujuan

Untuk menentukan *flow* maksimum jaringan tersebut digunakan algoritma *flow* maksimum yang baru dengan langkah-langkah sebagai berikut:

Langkah 1 : Dibentuk jaringan baru N^* dari N dengan menambahkan satu titik sumber s dan satu titik tujuan t sedemikian hingga s merupakan satu-satunya titik sumber N^* dan t satu-satunya titik tujuan N^*.

Langkah 2 : Dibuat kapasitas busur $(s, s_1), (s, s_2), (t_1, t)$, dan (t_2, t) dengan nilai kapasitas infinit atau ∞, dan nilai *flow* awal nol. Ditunjukkan seperti gambar berikut:



Gambar 3.3 Jaringan *N* Dengan Nilai Kapasitas Infinit dan Nilai *Flow* Awal Nol

Langkah 3 : Dilakukan rutin pelabelan.

1) Labeli $s = (s, +, \infty)$, titik s telah terlabeli tetapi belum teramati.
2) Dari titik s, diberi label titik s_1 dan s_2 masing-masing $s_1 = (s, +, \infty)$ dan $s_2 = (s, +, \infty)$. Sekarang titik s telah terlabeli dan teramati dengan label $s = (s, \oplus, \infty)$.
3) Dari titik s_1, diberi label untuk titik v_1 dan v_2 masing-masing $v_1 = (s_1, +, 6)$ dan $v_2 = (s_1, +, 4)$. Sekarang titik s_1 telah terlabeli dan teramati dengan label $s_1 = (s, \oplus, \infty)$.

4) Dari v_1, diberi label untuk titik t_1 dengan label $t_1 = (v_1, +, 5)$. Sekarang titik v_1 telah terlabeli dan teramati dengan label $v_1 = (s_1, \oplus, 6)$.

5) Dari titik t_1, diberi label untuk titik t dengan label $t = (t_1, +, \infty)$. Sekarang titik t_1 telah terlabeli dan teramati dengan label $t_1 = (v_1, \oplus, 5)$.

Langkah 4 : Digunakan prosedur balik.

Titik t dilabeli dari titik t_1, titik t_1 dilabeli dari v_1, titik v_1 dilabeli dari titik s_1, dan titik s_1 dilabeli dari titik s. Diperoleh lintasan peningkatan $P = (s, s_1, v_1, t_1, t)$ dan $i(P) = \min(\infty, \infty, 6, 5) = 5$.

Langkah 5 : Dilakukan rutin peningkatan.

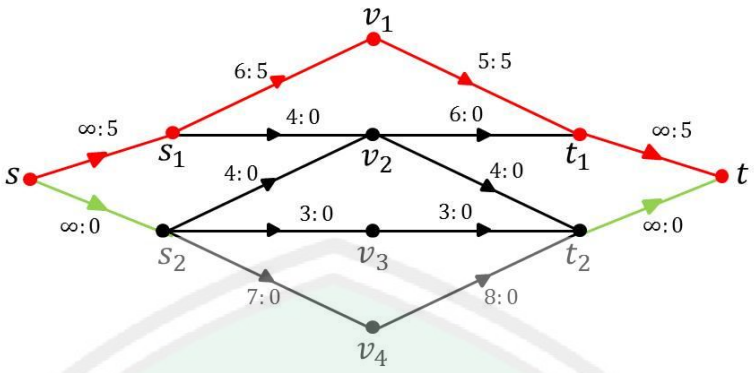
Titik t berlabel $(t_1, +, 5)$ maka nilai *flow* pada busur (t_1, t) ditambah 5.

Titik t_1 berlabel $(v_1, +, 5)$ maka nilai *flow* pada busur (v_1, t_1) ditambah 5.

Titik v_1 berlabel $(s_1, +, 6)$ maka nilai *flow* pada busur (s_1, v_1) ditambah 5.

Titik s_1 berlabel $(s, +, \infty)$ maka nilai *flow* pada busur (s, s_1) ditambah 5.

Ditunjukkan pada gambar berikut:



Gambar 3.4 Jaringan N dengan Nilai *Flow* Baru $f_1 = 5$

Setelah dilakukan penghapusan terhadap semua label, diperoleh nilai *flow* baru $f_1 = 5$.

Selanjutnya mengulang algoritma *flow* maksimum untuk mengkonstruksi *flow* maksimum dari titik s ke titik t di N^* dengan kembali ke langkah 2.

Langkah 2 : *Flow* f_1 dengan nilai 5.

Langkah 3 : Rutin pelabelan.

1) Labeli $s = (s, +, \infty)$, titik s telah terlabeli tetapi belum teramati.
2) Dari s, diberi label titik s_1 dan s_2 masing-masing dengan $s_1 = (s, +, \infty)$ dan $s_2 = (s, +, \infty)$. Sekarang titik s telah terlabeli dan teramati dengan label $s = (s, \oplus, \infty)$.
3) Dari s_1, diberi label untuk titik v_1, v_2 masing-masing $v_1 = (s_1, +, 1)$, dan $v_2 = (s_1, +, 4)$. Titik s_1 telah terlabeli dan teramati dengan label $s_1 = (s, \oplus, \infty)$.
4) Dari v_2, diberi label untuk titik t_1 dan t_2 masing-masing $t_1 = (v_2, +, 6)$ dan $t_2 = (v_2, +, 4)$. Titik v_2 telah terlabeli dan teramati dengan label $v_2 = (s_1, \oplus, 4)$.

5) Dari titik t_1, diberi label untuk titik t dengan label $t = (t_1, +, \infty)$. Selanjutnya t_1 telah terlabeli dan teramati dengan label $t_1 = (v_2, \oplus, 6)$.

Langkah 4 : Digunakan prosedur balik.

Titik t dilabeli dari titik t_1, titik t_1 dilabeli dari v_2, titik v_2 dilabeli dari titik s_1, dan titik s_1 dilabeli dari titik s. Dengan demikian diperoleh lintasan peningkatan $P = (s, s_1, v_2, t_1, t)$ dan $i(P) = \min(\infty, \infty, 4, 6) = 4$.

Langkah 5 : Dilakukan rutin peningkatan.

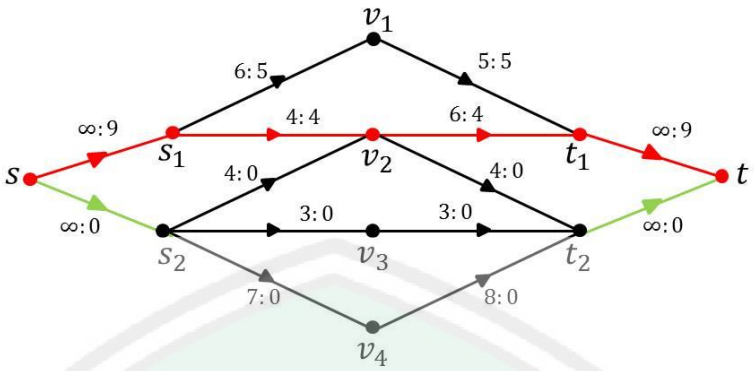
Titik t berlabel $(t_1, +, \infty)$ maka nilai *flow* pada busur (t_1, t) ditambah 4.

Titik t_1 berlabel $(v_2, +, 6)$ maka nilai *flow* pada busur (v_2, t_1) ditambah 4.

Titik v_2 berlabel $(s_1, +, 4)$ maka nilai *flow* pada busur (s_1, v_2) ditambah 4.

Titik s_1 berlabel $(s, +, \infty)$ maka nilai *flow* pada busur (s, s_1) ditambah 4.

Nilai *flow* pada busur lainnya adalah tetap. Setelah dilakukan penghapusan terhadap semua label diperoleh nilai *flow* baru yaitu $f_2 = f_1 + 4 = 5 + 4 = 9$.



Gambar 3.5 Jaringan N dengan Nilai *Flow* Baru $f_2 = 9$

Selanjutnya mengulang algoritma *flow* maksimum untuk mengkonstruksi *flow* maksimum dari titik s ke titik t di N^* dengan kembali ke langkah 2.

Langkah 2 : *Flow* f_2 dengan nilai 9.

Langkah 3 : Rutin pelabelan.

1) Labeli $s = (s, +, \infty)$, titik s telah terlabeli tetapi belum teramati.
2) Dari s, diberi label titik s_1 dan s_2 masing-masing $s_1 = (s, +, \infty)$ dan $s_2 = (s, +, \infty)$. Titik s telah terlabeli dan teramati dengan label $s = (s, \oplus, \infty)$.
3) Dari s_1, diberi label untuk titik v_1 dan v_2 dengan $v_1 = (s_1, +, 1)$ dan $v_2 = (s_1, +, 0)$. Selanjutnya titik s_1 telah terlabeli dan teramati dengan label $s_1 = (s, \oplus, \infty)$.
4) Dari v_2, diberi label titik t_2 dengan $t_2 = (v_2, +, 4)$ dan titik t_1 dengan $t_1 = (v_2, +, 2)$. Titik v_2 telah terlabeli dan teramati dengan label $v_2 = (s_1, \oplus, 0)$.
5) Dari t_2, diberi label untuk titik t dengan label $t = (t_2, +, \infty)$. Sekarang titik t_2 telah terlabeli dan teramati dengan label $t_2 = (v_2, \oplus, 4)$.

Langkah 4 : Digunakan prosedur balik.

Titik t dilabeli dari titik t_2, titik t_2 dilabeli dari v_2, titik v_2 dilabeli dari titik s_1, dan titik s_1 dilabeli dari titik s. Dengan demikian diperoleh lintasan peningkatan $P = (s, s_1, v_2, t_2, t)$ dan $i(P) = \min(\infty, \infty, 0, 4) = 0$.

Langkah 5 : Rutin peningkatan.

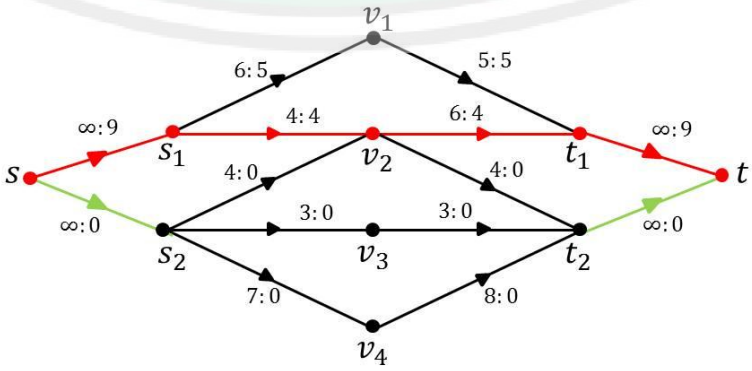
Titik t berlabel $(t_2, +, \infty)$ maka nilai *flow* pada busur (t_2, t) ditambah 0.

Titik t_2 berlabel $(v_2, +, 4)$ maka nilai *flow* pada busur (v_2, t_2) ditambah 0.

Titik v_2 berlabel $(s_1, +, 0)$ maka nilai *flow* pada busur (s_1, v_2) ditambah 0.

Titik s_1 berlabel $(s, +, \infty)$ maka nilai *flow* pada busur (s, s_1) ditambah 0.

Sedangkan nilai *flow* pada busur lainnya adalah tetap. Setelah dilakukan penghapusan terhadap semua label diperoleh nilai *flow* baru yaitu $f_3 = f_2 + 0 = 9 + 0 = 9$.



Gambar 3.6 Jaringan N dengan Nilai *Flow* Baru $f_3 = 9$

Selanjutnya mengulang algoritma *flow* maksimum untuk mengkonstruksi *flow* maksimum dari titik s ke titik t di N^* dengan kembali ke langkah 2.

Langkah 2 : *Flow* f_3 dengan nilai 9.

Langkah 3 : Rutin pelabelan.

1) Labeli $s = (s, +, \infty)$, titik s telah terlabeli tetapi belum teramati.
2) Dari s, diberi label titik s_1 dan s_2 masing-masing dengan $s_1 = (s, +, \infty)$ dan $s_2 = (s, +, \infty)$. Titik s telah terlabeli dan teramati dengan label $s = (s, \oplus, \infty)$.
3) Dari s_2, diberi label titik v_2, v_3, v_4 masing-masing $v_2 = (s_2, +, 4), v_3 = (s_2, +, 3)$, dan $v_4 = (s_2, +, 7)$. Selanjutnya titik s_2 telah terlabeli dan teramati dengan label $s_2 = (s, \oplus, \infty)$.
4) Dari v_2, diberi label untuk titik $t_1$dan t_2 dengan $t_1 = (v_2, +, 2)$ dan $t_2 = (v_2, +, 4)$. Titik v_3 telah terlabeli dan teramati dengan label $v_2 = (s_1, \oplus, 4)$.
5) Dari t_1, diberi label untuk titik t dengan label $t = (t_1, +, \infty)$. Selanjutnya titik t_2 telah terlabeli dan teramati dengan label $t_1 = (v_2, \oplus, 2)$.

Langkah 4 : Digunakan prosedur balik.

Titik t dilabeli dari t_1, titik t_1 dilabeli dari v_2, titik v_2 dilabeli dari titik s_2, dan titik s_2 dilabeli dari titik s. Dengan demikian diperoleh lintasan peningkatan P dengan $P = (s, s_2, v_2, t_1, t)$ dan $i(P) = \min(\infty, \infty, 4, 2) = 2$.

Langkah 5 : Rutin peningkatan.

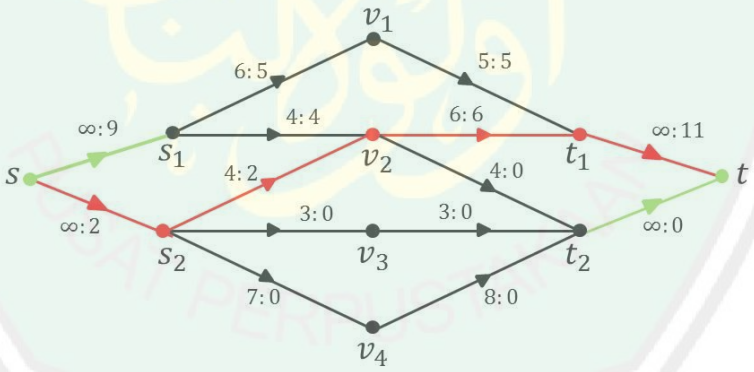
Titik t berlabel $(t_1, +, \infty)$ maka nilai *flow* pada busur (t_1, t) ditambah 2.

Titik t_1 berlabel $(v_2, +, 2)$ maka nilai *flow* pada busur (v_2, t_1) ditambah 2.

Titik v_2 berlabel $(s_2, +, 4)$ maka nilai *flow* pada busur (s_2, v_2) ditambah 2.

Titik s_2 berlabel $(s, +, \infty)$ maka nilai *flow* pada busur (s, s_2) ditambah 2.

Nilai *flow* pada busur lainnya adalah tetap. Setelah dilakukan penghapusan terhadap semua label diperoleh nilai *flow* baru yaitu $f_4 = f_3 + 3 = 9 + 2 = 11$.



Gambar 3.7 Jaringan N dengan Nilai *Flow* Baru $f_4 = 11$

Selanjutnya mengulang algoritma *flow* maksimum untuk mengkonstruksi *flow* maksimum dari titik s ke titik t di N^* dengan kembali ke langkah 2.

Langkah 2 : *Flow* f_4 dengan nilai 11.

Langkah 3 : Rutin pelabelan.

1) Labeli $s = (s, +, \infty)$, titik s telah terlabeli tetapi belum teramati.

2) Dari s, diberi label titik s_1 dan s_2 masing-masing dengan $s_1 = (s, +, \infty)$ dan $s_2 = (s, +, \infty)$. Titik s telah terlabeli dan teramati dengan label $s = (s, \oplus, \infty)$.

3) Dari s_2, diberi label titik v_2, v_3, v_4 masing-masing dengan label $v_2 = (s_2, +, 2), v_3 = (s_2, +, 3)$, dan $v_4 = (s_2, +, 7)$. Titik s_2 telah terlabeli dan teramati dengan label $s_2 = (s, \oplus, \infty)$.

4) Dari v_2, diberi label untuk titik t_1 dan t_2 dengan label $t_1 = (v_2, +, 0)$ dan $t_2 = (v_2, +, 4)$. Titik v_2 telah terlabeli dan teramati dengan label $v_2 = (s_2, \oplus, 2)$.

5) Dari t_2, diberi label untuk titik t dengan label $t = (t_2, +, \infty)$. Selanjutnya titik t_2 telah terlabeli dan teramati dengan label $t_2 = (v_2, \oplus, 4)$.

Langkah 4 : Digunakan prosedur balik.

Titik t dilabeli dari t_2, titik t_2 dilabeli dari v_2, titik v_2 dilabeli dari titik s_2, dan titik s_2 dilabeli dari titik s. Dengan demikian diperoleh lintasan peningkatan P dengan $P = (s, s_2, v_2, t_2, t)$ dan $i(P) = \min(\infty, \infty, 2, 4) = 2$.

Langkah 5 : Rutin peningkatan.

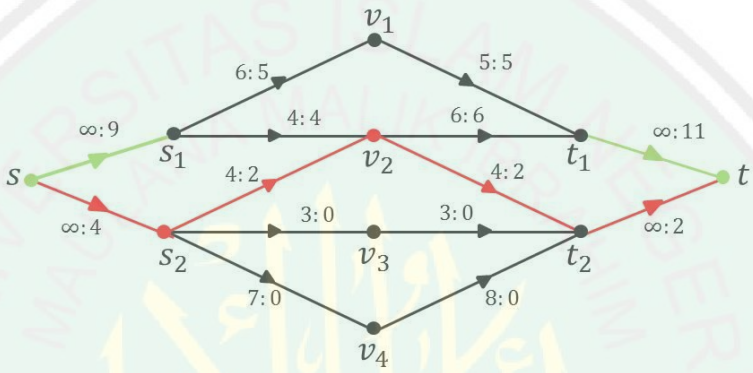
Titik t berlabel $(t_2, +, \infty)$ maka nilai *flow* pada busur (t_2, t) ditambah 2.

Titik t_2 berlabel $(v_2, +, 4)$ maka nilai *flow* pada busur (v_2, t_2) ditambah 2.

Titik v_2 berlabel $(s_2, +, 2)$ maka nilai *flow* pada busur (s_2, v_2) ditambah 2.

Titik s_2 berlabel $(s,+,\infty)$ maka nilai *flow* pada busur (s,s_2) ditambah 2.

Nilai *flow* pada busur lainnya adalah tetap. Setelah dilakukan penghapusan terhadap semua label diperoleh nilai *flow* baru yaitu $f_5 = f_4 + 1 = 11 + 2 = 13$.



Gambar 3.8 Jaringan N dengan Nilai *Flow* Baru $f_5 = 13$

Selanjutnya mengulang algoritma *flow* maksimum untuk mengkonstruksi *flow* maksimum dari titik s ke titik t di N^* dengan kembali ke langkah 2.

Langkah 2 : *Flow* f_5 dengan nilai 13.

Langkah 3 : Rutin pelabelan

1) Labeli $s = (s,+,\infty)$, titik s telah terlabeli tetapi belum teramati.
2) Dari s, diberi label titik s_1 dan s_2 masing-masing dengan labels$_1 = (s,+,\infty)$ dan $s_2 = (s,+,\infty)$. Titik s telah terlabeli dan teramati dengan label $s = (s,\oplus,\infty)$.
3) Dari s_2, diberi label titik v_2, v_3 dan v_4 masing-masing dengan label $v_2 = (s_2,+,0), v_3 = (s_2,+,3)$, dan $v_4 = (s_2,+,7)$.

Selanjutnya titik s_2 telah terlabeli dan teramati dengan label $s_2 = (s, \oplus, \infty)$.

4) Dari v_3, diberi label untuk titik t_2 dengan label $t_2 = (v_3, +, 3)$. Titik v_3 telah terlabeli dan teramati dengan label $v_3 = (s_2, \oplus, 3)$.

5) Dari t_2, diberi label untuk titik t dengan label $t = (t_2, +, \infty)$. Selanjutnya titik t_2 telah terlabeli dan teramati dengan label $t_2 = (v_3, \oplus, 3)$.

Langkah 4 : Digunakan prosedur balik.

Titik t dilabeli dari t_2, titik t_2 dilabel dari v_3, titik v_3 dilabel dari titik s_2, dan titik s_2 dilabel dari titik s. Diperoleh lintasan peningkatan $P = (s, s_2, v_3, t_2, t)$ dan $i(P) = \min(\infty, 3, 3, \infty) = 3$.

Langkah 5 : Rutin peningkatan.

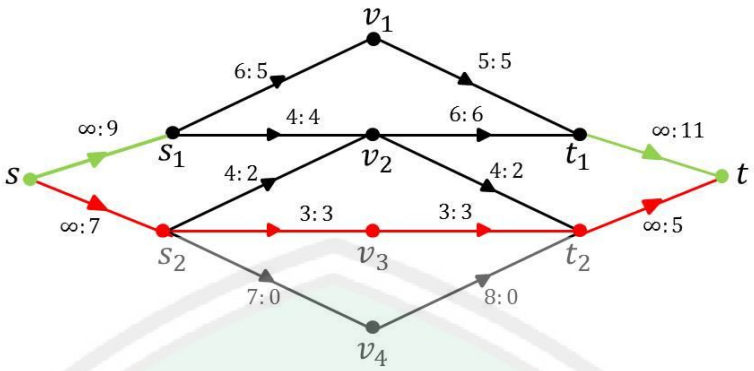
Titik t berlabel $(t_2, +, \infty)$ maka nilai *flow* pada busur (t_2, t) ditambah 3.

Titik t_2 berlabel $(v_3, +, 3)$ maka nilai *flow* pada busur (v_3, t) ditambah 3.

Titik v_3 berlabel $(s_2, +, 3)$ maka nilai *flow* pada busur (s_2, v_3) ditambah 3.

Titik s_2 berlabel $(s, +, \infty)$ maka nilai *flow* pada busur (s, s_2) ditambah 3.

Sedangkan nilai *flow* pada busur lainnya adalah tetap. Setelah dilakukan penghapusan terhadap semua label diperoleh nilai *flow* baru yaitu $f_6 = f_5 + 3 = 13 + 3 = 16$.



Gambar 3.9 Jaringan N dengan Nilai *Flow* Baru $f_6 = 16$

Selanjutnya mengulang algoritma *flow* maksimum untuk mengkonstruksi *flow* maksimum dari titik s ke titik t di N^* dengan kembali ke langkah 2.

Langkah 2 : *Flow* f_6 dengan nilai 16.

Langkah 3 : Dilakukan rutin pelabelan.

1) Labeli $s = (s, +, \infty)$, titik s telah terlabeli tetapi belum teramati.
2) Dari s, diberi label titik s_1 dan s_2 masing-masing dengan label $s_1 = (s, +, \infty)$ dan $s_2 = (s, +, \infty)$. Titik s telah terlabeli dan teramati dengan label $s = (s, \oplus, \infty)$.
3) Dari s_2, diberi label untuk titik v_2, v_3, dan v_4 masing-masing dengan label $v_2 = 0, v_3 = (s_2, +, 0)$ dan $v_4 = (s_2, +, 7)$. Selanjutnya titik s_2 telah terlabeli dan teramati dengan label $s_2 = (s, \oplus, \infty)$.
4) Dari v_4, diberi label untuk titik t_2 dengan label $t_2 = (v_4, +, 8)$. Sekarang titik v_4 telah terlabeli dan teramati dengan label $v_4 = (s_2, \oplus, 7)$.

5) Dari t_2, diberi label untuk titik t dengan label $t = (t_2, +, \infty)$. Selanjutnya titik t_2 telah terlabeli dan teramati dengan label $t_2 = (v_4, \oplus, 8)$.

Langkah 4 : Digunakan prosedur balik.

Titik t telah terlabeli dari t_2, titik t_2 telah terlabeli dari v_4, titik v_4 telah terlabeli dari titik s_2, dan titik s_2 telah terlabeli dari titik s. Dengan demikian diperoleh lintasan peningkatan $P = (s, s_2, v_4, t_2, t)$ dan $i(P) = \min(\infty, \infty, 7, 8) = 7$.

Langkah 5 : Dilakukan rutin peningkatan.

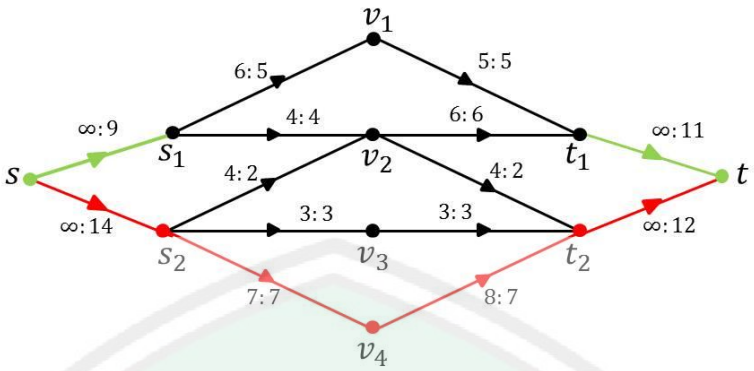
Titik t berlabel $(t_2, +, \infty)$ maka nilai *flow* pada busur (t_2, t) ditambah 7.

Titik t_2 berlabel $(v_4, +, 8)$ maka nilai *flow* pada busur (v_4, t_2) ditambah 7.

Titik v_4 berlabel $(s_2, +, 7)$ maka nilai *flow* pada busur (s_2, v_4) ditambah 7.

Titik s_2 berlabel $(s, +, \infty)$ maka nilai *flow* pada busur (s, s_2) ditambah 7.

Nilai *flow* pada busur lainnya adalah tetap. Setelah dilakukan penghapusan terhadap semua label diperoleh nilai *flow* baru yaitu $f_7 = f_6 + 1 = 16 + 7 = 23$.



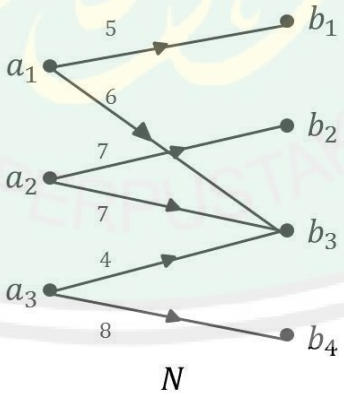
Gambar 3.10 Jaringan N dengan Nilai *Flow* Baru $f_7 = 23$

Karena semua titik di N^* yang terlabeli telah teramati dan titik t tidak dilabeli, maka proses dihentikan.

Jadi disimpulkan bahwa *flow* f_7 adalah *flow* maksimum di N dengan nilai 23.

3.3 Konstruksi Algoritma *Flow* Maksimum pada Jaringan Bipartisi

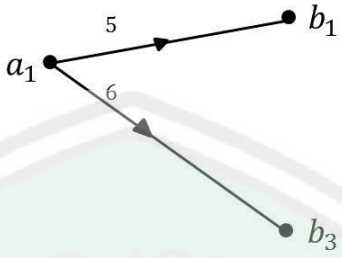
1. Menyajikan jaringan bipartisi $N = (V, \vec{E})$.



Gambar 3.11 Penerapan Algoritma *Flow* Maksimum Jaringan Bipartisi N

Karena N bipartisi maka terdapat himpunan saling lepas $V_1 \subseteq V$ dan $V_2 \subseteq V$ sehingga $V = V_1 + V_2$ dan setiap $\vec{e} \in \vec{E}$ menghubungkan titik di V_1 ke titik di V_2. Misalkan $|V_1| = m$ dan $|V_2| = n$.

2. Membagi jaringan N menjadi m bagian dengan membuat jaringan-jaringan yang berawal di $x \in V_1$ dan berujung di $y \in V_2$.

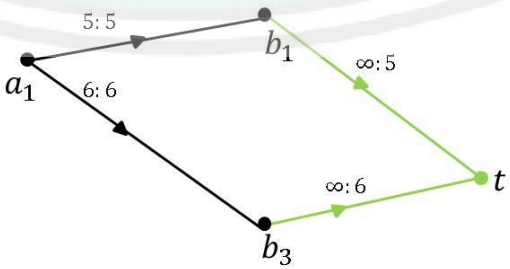


Gambar 3.12 Bagian Pertama Jaringan *N*

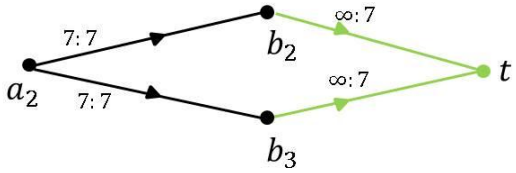
Gambar 3.13 Bagian Kedua Jaringan *N*

Gambar 3.14 Bagian Ketiga Jaringan *N*

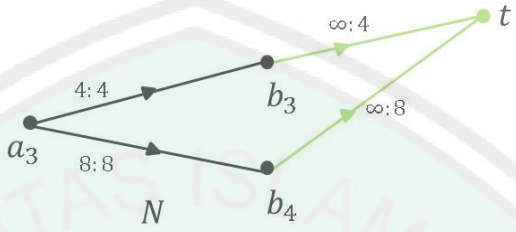
3. Menghitung nilai *flow* maksimum f_i untuk setiap jaringan yang berawal di $x_i \in V_1$untuk $1 \leq i \leq m$.



Gambar 3.15 *Flow* Maksimum Bagian Pertama Jaringan N, $f_1 = 11$



Gambar 3.16 *Flow* Maksimum Bagian Kedua Jaringan N, $f_2 = 14$

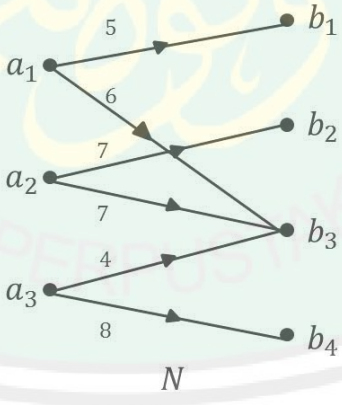


Gambar 3.17 *Flow* Maksimum Bagian Ketiga Jaringan N, $f_3 = 12$

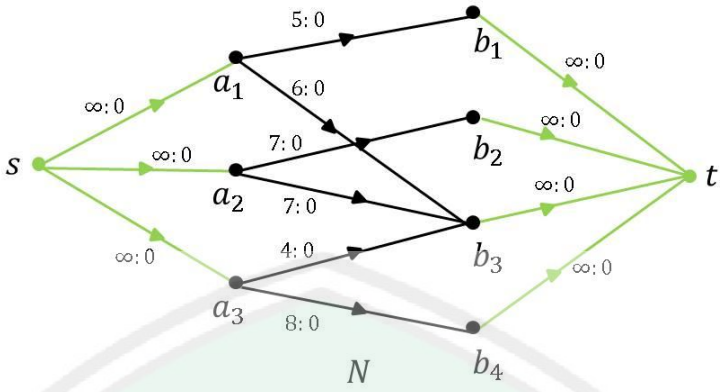
4. *Flow* maksimum jaringan bipartisi $N = \sum_{i=1}^{m} f_i$.

$$f = f_1 + f_2 + f_3 = 37.$$

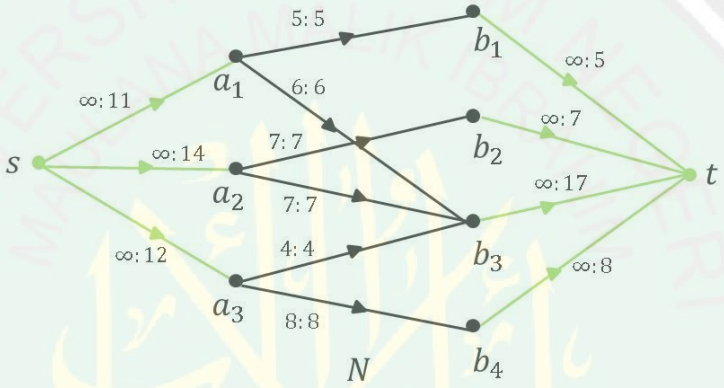
Sekarang dicari *flow* maksimum jaringan N menggunakan algoritma Ford-Fulkerson yang telah dimodifikasi pada Subbab 3.2.



Gambar 3.18 Jaringan Bipartisi N



Gambar 3.19 Jaringan N Setelah Ditambah Sumber Baru dan Tujuan Baru



Gambar 3.20 *Flow* Maksimum Jaringan Bipartisi N Menggunakan Algoritma Ford-Fulkerson adalah 37

Hasil pencarian *flow* maksimum pada jaringan bipartisi dengan algoritma Ford-Fulkerson dan dengan membagi jaringan N menjadi beberapa bagian berdasarkan titik sumbernya memberikan hasil yang sama. Oleh sebab itu, dibentuk lemma berikut serta pembuktiannya.

Lemma:

Nilai *flow* maksimum pada jaringan bipartisi adalah hasil penjumlahan *flow* maksimum dari masing-masing *flow* maksimum yang berasal dari setiap sumber pada jaringan bipartisi.

Bukti:

Misal $N = (V, E)$ adalah jaringan bipartisi dengan $V_1 + V_2 = V$ yang setiap busur di E menghubungkan titik di V_1 ke titik di V_2. Dengan demikian

jaringan tersebut merupakan jaringan dengan beberapa titik sumber yaitu titik-titik anggota V_1, dan mempunyai beberapa titik tujuan yaitu titik-titik yang ada di V_2. Misalkan $|V_1| = m$ dan $|V_2| = n$, maka setiap busur di E menghubungkan setiap titik x di V_1 ke titik y di V_2. Misalkan dibelah jaringan N menjadi m bagian dengan masing-masing bagian adalah jaringan yang bersumber di x_i dan berakhir di V_2 maka dapat dicari nilai *flow* maksimum dari x_i ke V_2. Misalkan f_i adalah nilai *flow* maksimum dari x_i ke V_2.

Setelah diperoleh nilai f_i *flow* maksimum dari x_i ke V_2, dicari nilai *flow* maksimum jaringan N dengan algoritma Ford-Fulkerson yang dimodifikasi untuk mencari *flow* maksimum jaringan dengan beberapa titik sumber dan beberapa titik tujuan seperti dibahas pada Subbab 3.2. Setelah dibuat titik bantu s yang menjadi sumber besar yang terhubung ke setiap x_i dan titik t yang menjadi tujuan besar yang dihubungkan dari V_2 maka nilai *flow* maksimum jaringan bipartisi N adalah nilai maksimum *flow* dari s ke t. Diberi nilai kapasitas ∞ untuk setiap busur dari s ke x_i dan dari V_2 ke t dengan masing-masing nilai awal *flow* adalah nol. Kemudian dilakukan iterasi untuk mencari lintasan peningkatan di jaringan tersebut.

Pada iterasi pertama dicari lintasan peningkatan yang melalui x_1, dan didapat $f = f_1$. Setelah itu dicari lintasan peningkatan yang melalui x_2 dan didapat $f = f_1 + f_2$. Iterasi dilanjutkan sampai tidak ditemukan lagi lintasan peningkatan, yaitu pada saat semua titik x_i telah diselidiki.

Pada iterasi ke-1 diperoleh $f = f_1$. Pada iterasi ke-2 diperoleh $f = f_1 + f_2$. Pada iterasi ke-3 diperoleh $f = f_1 + f_2 + f_3$. Jika iterasi dilakukan sampai m

iterasi, artinya sampai semua $x_i \in V_1$ diselidiki maka nilai *flow* maksimum jaringan bipartisi N, yaitu $f = \sum_{i=1}^{m} f_i$.

3.4 Konsep *Flow* Maksimum pada Jaringan Bipartisi dalam Al-Quran

Dalam kajian *flow* maksimum pada jaringan bipartisi dijelaskan bahwa nilai *flow* maksimum yang diperoleh dengan algoritma Ford-Fulkerson sama dengan nilai *flow* yang diperoleh dengan membagi jaringan bipartisi berdasarkan titik sumbernya dan mencari nilai *flow* maksimum dari masing-masing titik sumber kemudian menjumlahkannya $(f = f_1 + f_2 + \cdots + f_n)$ yang masing-masing nilai f_i dengan $1 \leq i \leq n$ tidak pernah melebihi kapasitas busur.

Diibaratkan f adalah total kebaikan yang dilakukan oleh manusia dan $f_1, f_2, \ldots, f_n$ berbagai macam kebaikan yang dilakukan manusia karena amal kebaikan itu bermacam-macam jenisnya seperti firman Allah dalam surat an-Nuur/24:56 berikut:

وَأَقِيمُواْ ٱلصَّلَوٰةَ وَءَاتُواْ ٱلزَّكَوٰةَ وَأَطِيعُواْ ٱلرَّسُولَ لَعَلَّكُمۡ تُرۡحَمُونَ ﴿٥٦﴾

"Dan Dirikanlah sembahyang, tunaikanlah zakat, dan taatlah kepada rasul, supaya kamu diberi rahmat" (QS. an-Nuur/24:56).

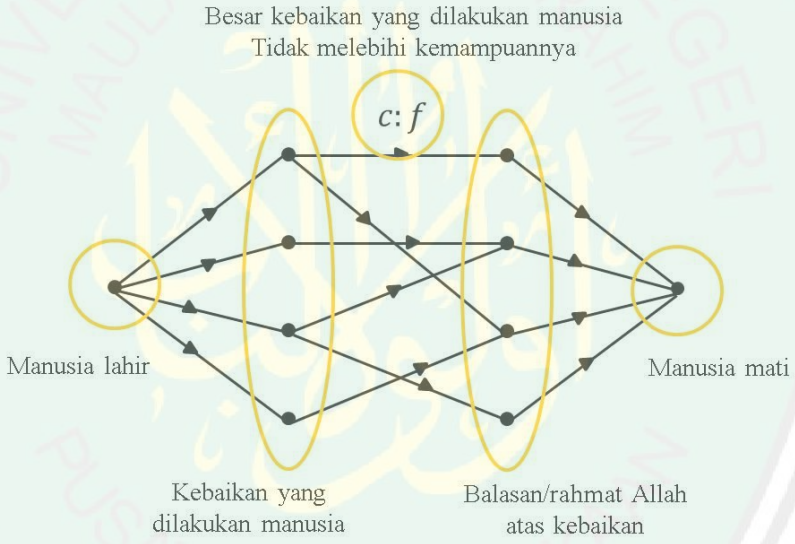
Dari ayat tersebut Allah memerintahkan untuk berbuat kebaikan-kebaikan yaitu melakukan shalat, menunaikan zakat, dan taat kepada rasul. Dalam pelaksanaannya, kebaikan-kebaikan tersebut juga diatur agar tidak melebihi kemampuan manusia itu sendiri. Sebagai contoh shalat, diwajibkan dilakukan dengan berdiri jika mampu, dibolehkan shalat dengan duduk atau berbaring jika dalam keadaan sakit. Demikian dengan zakat, diwajibkan membayar zakat sebesar $\pm 2{,}7$ kg bahan makanan pokok untuk zakat fitrah dan zakat mal sesuai ketentuan yang diatur berdasarkan kekayaan yang dimiliki. Hal ini menjelaskan nilai *flow*

tidak melebihi kapasitas. Dan nilai kebaikan yang dilakukan manusia juga akan selalu bertambah dengan melakukan kebaikan demi kebaikan.

وَإِنَّ ٱلدِّينَ لَوَٰقِعٌ ۝٦

"Dan Sesungguhnya (hari) pembalasan pasti terjadi"(QS. adz-Dzaariyat/51:6).

Nilai dari kebaikan-kebaikan yang dilakukan manusia selama hidup akan dihitung setelah dia mati pada hari perhitungan dan kemudian dibalas oleh Allah sesuai dengan perbuatannya pada hari pembalasan. Secara figural perjalanan hidup manusia dapat digambarkan seperti jaringan bipartisi berikut.



Gambar 3.21 Ilustrasi Kehidupan Manusia dalam Bentuk Jaringan Bipartisi

BAB IV

PENUTUP

4.1 Kesimpulan

Flow maksimum pada jaringan bipartisi ditentukan dengan menggunakan algoritma sebagai berikut:

a. Menyajikan jaringan bipartisi $N = (V, \vec{E})$.

   Karena N bipartisi maka terdapat himpunan saling lepas $V_1 \subseteq V$ dengan $|V_1| = m$ dan $V_2 \subseteq V$ dengan $|V_2| = n$ sehingga $V = V_1 + V_2$ dan setiap $\vec{e} \in \vec{E}$ menghubungkan titik di V_1 ke titik di V_2.

b. Membagi jaringan N menjadi m bagian dengan membuat jaringan-jaringan yang berawal di $x \in V_1$ dan berujung di $y \in V_2$.

c. Menghitung nilai *flow* maksimum f_i untuk setiap jaringan kecil yang berawal di $x_i \in V_1$untuk $1 \leq i \leq m$.

d. *Flow* maksimum jaringan bipartisi $N = \sum_{i=1}^{m} f_i$.

Dengan demikian, diperoleh lemma bahwa nilai *flow* maksimum pada jaringan bipartisi adalah hasil penjumlahan *flow* maksimum dari masing-masing *flow* maksimum yang berasal dari setiap sumber pada jaringan bipartisi.

4.2 Saran

Dalam penelitian ini penulis hanya membatasi pembatasan pada penentuan algoritma *flow* maksimum pada jaringan bipartisi saja. Oleh karena itu, penulis mengharapkan pada pembaca untuk mengembangkan dan mengkaji teori graf secara lebih menyeluruh.

DAFTAR PUSTAKA

Abdussakir, Azizah, N. N., Nofandika, F. F. 2009. *Teori Graf*. Malang: UIN-Malang Press.

Ahuja, R. K., Orlin, J. B., Stein, C., & Tarjan, R. E. 1994. Improved Algorithms For Bipartite Network Flow. *SIAM J. COMPUT*, 906-933.

As-Syuyuti, J., & Muhammad, J. 2010. *Tafsir Jalalain*. Tasikmalaya: Anonim.

Azar, Y., Madry, A., Moscibroda, T., Panigrahi, D., & Srinivasan, A. 2013. *Maximum Bipartite Flow in Networks with Adaptive Channel Width*, (Online): 1-19, (http://research.microsoft.com/pubs/147615/icalp2009.pdf), diakses 24 Januari 2015.

Bollobas, B. 1998. *Modern Graph Theory*. New York: Springer.

Budayasa, I. K. 2007. *Teori Graf dan Aplikasinya*. Surabaya: Unesa University Press.

Chartrand, G., & Lesniak, L. 1996. *Graphs and Digraphs*. London: Chapman & Hall.

Farizal, T. 2013. *Pencarian Aliran Maksimum dengan Algoritma Ford-Fulkerson*. Skripsi tidak dipublikasikan. Semarang: Universitas Negeri Semarang.

Ruohonen, K. 2013. *Graph Theory*. (Online), (http://math.tut.fi/~ruohonen/GT_English.pdf), diakses 24 Januari 2015.

RIWAYAT HIDUP

Nia Cahyani, lahir di Kabupaten Malang pada 8 Februari 1993, biasa dipanggil Nia, tinggal di Jl. Jambu 66 RT: 01 RW: 01 Desa Kedungpedaringan Kecamatan Kepanjen Kabupaten Malang. Anak sulung dari Bapak Takim dan Ibu Mudayati.

Pendidikan dasarnya ditempuh di SDN 01 Kedungpedaringan dan lulus pada tahun 2005, setelah itu dia melanjutkan ke SMP Negeri 4 Kepanjen dan lulus tahun 2008. Kemudian dia melanjutkan pendidikan ke SMA Negeri 1 Kepanjen dan lulus pada tahun 2011. Setelah lulus dari SMA dia menempuh kuliah di Universitas Islam Negeri Maulana Malik Ibrahim Malang mengambil jurusan Matematika.

Selama menempuh pendidikan dasar sampai tingkat SMA, dia selalu meraih prestasi. Prestasi yang pernah penulis raih di antaranya selalu meraih rangking 3 besar selama duduk di bangku sekolah dasar, Juara II Olimpiade Matematika Tingkat SMP se-Kabupaten Malang tahun 2007, Juara III Olimpiade Matematika Tingkat SMA se-Kabupaten Malang tahun 2009, Juara II Olimpiade Matematika Tingkat SMA se-Kabupaten Malang tahun 2010, Juara Harapan I Olimpiade Bahasa Jerman Tingkat SMA se-Jawa Timur tahun 2010, dan Juara I Olimpiade Olahraga Siswa Nasional Tingkat SMA se-Kabupaten Malang untuk cabang olahraga Karate tahun 2010.

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No. 50 Dinoyo Malang Telp./Fax.(0341)558933

BUKTI KONSULTASI SKRIPSI

Nama : Nia Cahyani
NIM : 11610027
Fakultas/Jurusan : Sains dan Teknologi/Matematika
Judul Skripsi : Menentukan *Flow* Maksimum pada Jaringan Bipartisi
Pembimbing I : H. Wahyu H. Irawan, M.Pd
Pembimbing II : Abdul Aziz, M.Si

No	Tanggal	Hal	Tanda Tangan
1.	26 Januari 2015	Konsultasi Bab I dan II	1.
2.	26 Januari 2015	Konsultasi Kajian Agama Bab I dan II	2.
3.	26 Maret 2015	Revisi Kajian Agama	3.
4.	31 Maret 2015	ACC Kajian Agama Bab I dan II	4.
5.	9 Maret 2015	Revisi Bab I dan II	5.
6.	9 Maret 2015	ACC Bab I dan II	6.
7.	11 Mei 2015	Konsultasi Agama Bab III	7.
8.	12 Mei 2015	Konsultasi Bab III dan Bab IV	8.
9.	12 Mei 2015	Revisi Agama Bab III	9.
10.	13 Mei 2015	ACC Kajian Agama	10.
11.	13 Mei 2015	ACC Keseluruhan	11.

Malang, 15 Mei 2015
Mengetahui,
Ketua Jurusan Matematika

Dr. Abdussakir, M.Pd
NIP. 19751006 200312 1 001