PERBANDINGAN ALGORITMA FORD-FULKERSON, EDMONDS-KARP DAN *PUSH-RELABEL* DALAM MENENTUKAN ALIRAN MAKSIMUM

SKRIPSI

NURFADILAH MEDI

H011191011

PROGRAM STUDI MATEMATIKA

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2023

PERBANDINGAN ALGORITMA FORD-FULKERSON, EDMONDS-KARP DAN PUSH RELABEL DALAM MENENTUKAN ALIRAN MAKSIMUM

SKRIPSI

Di ajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Sains pada Program Studi Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

NURFADILAH MEDI

H011191011

PROGRAM STUDI MATEMATIKA

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2023

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama	: Nurfadilah Medi
Nim	: H011191011
Program Studi	: Matematika
Jenjang	: S1

Menyatakan dengan ini bahwa karya tulisan saya yang berjudul

Perbandingan Algoritma Ford-Fulkerson, Edmonds-Karp Dan *Push-Relabel* Dalam Menentukan Aliran Maksimum

Adalah karya tulisan saya sendiri dan bukan merupakan pengambilan alihan tulisan orang lain bahwa tulisan skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri.

Apabila di kemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Makassar, 31 Mei 2023

Yang menyatakan,

Nurfadilah Medi

Nim.H011191011

LEMBAR PENGESAHAN

PERBANDINGAN ALGORITMA FORD-FULKERSON, EDMONDS-KARP DAN *PUSH-RELABEL* DALAM MENENTUKAN ALIRAN MAKSIMUM

Disusun dan diajukan oleh

NURFADILAH MEDI

H011191011

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

Pada tanggal, 31 Mei 2023

Dan dinyatakan telah memenuhi syarat kelulusan.

Menyetujui,

Pembimbing Utama,

Dr. Agustinus Ribal, S.Si. M.Sc.
NIP.19750816 199903 1 001

Pembimbing Pertama,

Dr. Khaeruddin, M.Sc.
NIP.19650914 199103 1 003

Ketua Program Studi,

Prof. Dr. Nurdin, S.Si., M.Si.
NIP.19700807 200003 1 002

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN UNIVERSITAS HASANUDDIN DEPARTEMEN MATEMATIKA FAKULTAS MIPA

KATA PENGANTAR

Puji syukur penulis panjatkan atas kehadirat Allah SWT atas segala rahmat dan hidayah-nya sehingga penulis dapat menyelesaikan skripsi ini. Shalawat serta salam senantiasa tercurahkan kepada junjungan Nabi Muhammad SAW, sebagai Nabi yang telah menjadi suri tauladan bagi seluruh umatnya sehingga penyusunan skripsi ini dapat terselesaikan dengan judul "**Perbandingan Algoritma Ford-Fulkerson, Edmonds-Karp dan Push Relabel dalam Menentukan Aliran Maksimum"**, Sebagai salah satu syarat untuk mendapatkan gelar **Sarjana Sains (S.Si)** pada Program Studi Matematika Departemen Matematika Fakultas Matematika dan Ilmu Pengetahun Alam Universitas Hasanuddin.

Penulis menyadari bahwa dalam penyelesaian skripsi ini tidak dapat terselesaikan tanpa adanya bantuan, dukungan, bimbingan, motivasi, serta nasehat dari berbagai pihak. Pada kesempatan ini, izinkan penulis mengucapkan terima kasih dan memberikan penghargaan kepada kedua orang tua penulis, Bapak **Medi** dan Ibu **Suarni** yang telah sabar membesarkan dan mendidik penulis, serta memberikan do'a dan materi, sehingga penulis bisa mencapai titik ini dan mampu menyelesaikan Pendidikan di perguruan tinggi dan mendapat gelar yang insya Allah dapat bermanfaat dikemudian hari. Terima kasih kepada kakak saya **Hardianti Medi**, dan **Suci Rahmadani M**, serta seluruh keluarga yang telah memberi do'a dan dukungan dalam menyelesaikan skripsi ini. Pada kesempatan ini pula, penulis hendak menyampaikan terima kasih kepada:

1. Bapak **Prof. Dr. Ir. Jamaluddin Jompa, M.Sc.** selaku Rektor Universitas Hasanuddin beserta seluruh jajarannya, serta Bapak **Dr. Eng. Amiruddin** selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam beserta jajarannya.
2. Bapak **Prof. Dr. Nurdin, S.Si., M.Si.** selaku Ketua Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin beserta Bapak dan Ibu **Dosen Departemen Matematika** yang telah memberikan banyak ilmu dan pengetahuan kepada penulis selama menjadi mahasiswa di Program Studi Matematika serta Para Staf Departemen Matematika yang telah membantu dan memudahkan penulis

dalam berbagai hal administrasi.

3. Bapak **Dr. Agustinus Ribal, S.Si., M.Sc.** dan Bapak **Dr. Khaeruddin, M.Sc.** selaku Dosen Pembimbing yang dengan sabar, tulus, dan ikhlas meluangkan banyak waktu di tengah kesibukan dan prioritasnya untuk membimbing dan memberi masukan serta motivasi dalam penulisan skripsi ini.
4. Bapak **Prof. Dr. Budi Nurwahyu, MS.** dan Ibu **Nur Rohmah Oktaviani P., S.Si., M.Si**. selaku Tim Penguji yang telah meluangkan waktunya untuk memberikan masukan dan kritikan yang membangun terhadap penyempurnaan penulisan skripsi ini.
5. Sahabat penulis, **Qalbi, Pio, Dara, Azizah, Dilong, Mumu, Nisa dan Nanda** yang telah memberikan dukungan dan bantuan kepada penulis selama masa studi sarjana.
6. Teman-teman **Matematika 2019** yang senantiasa memberikan bantuan dan dukungan moril kepada penulis, serta memberikan momen berharga bagi penulis selama masa studi sarjana.
7. **Kim Minseok**, **Kim Junmyeon**, **Zhang Yixing**, **Byun Baekhyun**, **Kim Jongdae**, **Park Chanyeol**, **Doh Kyungsoo**, **Kim Jongin** dan **Oh Sehun** selaku namjachinggu yang telah menemani dan menghibur penulis dengan karya-nya sehingga penulis tetap happy kiyowo mengerjakan skripsi ini.
8. Teman-teman **EXOL** yang telah memberikan semangat, motivasi dan menghibur penulis selama mengejarkan skripsi ini.
9. Semua pihak yang tidak dapat di sebutkan satu per satu, yang telah memberikan do'a, dukungan, motivasi dan inspirasi bagi penulis dalam mengerjakan skripsi ini.

Akhir kata, penulis berharap semoga segala bentuk kebaikan yang telah diberikan bernilai ibadah dan mendapatkan balasan dari Allah SWT. Semoga skripsi ini membawa manfaat bagi pengembangan ilmu.

Makassar, 31 Mei 2023

Nurfadilah Medi

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMISI

Sebagai sivitas akademik Universitas Hasanuddin, saya yang bertanda tangan di bawah ini:

Nama	: Nurfadilah Medi
Nim	: H011191011
Program Studi	: Matematika
Departemen	: Matematika
Fakultas	: Matematika dan Ilmu Pengetahuan Alam
Jenis Karya	: Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Hasanuddin **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas karya saya yang berjudul:

Perbandingan Algoritma Ford-Fulkerson, Edmonds-Karp Dan *Push-Relabel* Dalam Menentukan Aliran Maksimum

Beserta perangkat yang ada (jika diperlukan). Terkait dengan hal di atas, maka pihak Universitas berhak menyimpan, mengalih-media/format-kan, mengelolah dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya,
Dibuat di Makassar pada, 31 Mei 2023

Yang menyatakan,

Nurfadilah Medi

ABSTRAK

Masalah aliran maksimum adalah permasalahan optimasi dengan tujuan mengetahui nilai aliran di dalam sebuah jaringan aliran *(network flow).* Jaringan aliran merupakan sebuah graf yang memiliki aliran (*flow*) pada setiap sisinya yang memenuhi syarat $0 \leq f(u,v) \leq c(u,v)$. Masalah aliran maksimum dapat diselesaikan menggunakan beberapa algoritma diantaranya algortima Ford-Fulkerson, Edmonds-Karp dan Push-Relabel. Dalam skripsi ini ketiga metode tersebut akan dibandingkan dalam tiga aspek yang berbeda yaitu kompleksitas waktu, kesederhanaan, dan jumlah iterasi yang diperlukan untuk menentukan aliran maksimum baik secara manual maupun dengan menggunakan program python. Dengan adanya program python tersebut, masalah aliran maksimum dalam sebuah jaringan dapat ditentukan dengan cepat. Hasil dari program python telah dibandingkan dengan hasil yang diperoleh secara manual yang menunjukkan bahwa program python tidak salah. Ketiga algortima memberikan hasil yang sama dalam mencari aliran maksimum akan tetapi Ford-Fulkerson lebih sederhana dalam pencarian lintasan dibandingkan dengan Edmonds-Karp meskipun jumlah iterasi yang dibutuhkan Edmonds-Karp lebih sedikit. Berbeda dengan algortima Ford-Fulkerson dan Edmonds-Karp, pada algoritma Push-Relabel tidak menggunakan lintasan yang mana aliran tidak langsung dikirimkan dari titik sumber ke titik tujuan tetapi di alirkan ke titik yang bertetangga sehingga algoritma ini lebih mudah di terapkan untuk *network* yang besar walaupun membutuhkan lebih banyak iterasi.

Kata kunci: Aliran maksimum, Jaringan aliran, Ford-Fulkerson, Edmonds-Karp, Push-Relabel, Python.

ABSTRACT

The maximum flow problem is an optimization problem aimed at determining the maximum flow value in a network flow. A network flow is a graph with flows on each of its sides that satisfy the condition $0 \leq f(u,v) \leq c(u,v)$. The maximum flow problem can be solved using several algorithms, including the Ford-Fulkerson, Edmonds-Karp, and Push-Relabel algorithms. In this thesis, these three methods will be compared in three different aspects: time complexity, simplicity, and the number of iterations required to determine the maximum flow, both manually and using a Python program. With the Python program, the maximum flow problem in a network can be determined quickly. The results from the Python program have been compared to manually obtained results, showing that the Python program is correct. All three algorithms yield the same results in finding the maximum flow, but Ford-Fulkerson is simpler in finding paths compared to Edmonds-Karp, even though Edmonds-Karp requires fewer iterations. Unlike the Ford-Fulkerson and Edmonds-Karp algorithms, the Push-Relabel algorithm does not use paths, where the flow is not directly sent from the source to the destination, but rather is pushed to neighboring points. Therefore, this algorithm is easier to apply to large networks, although it requires more iterations.

Keywords: Maximum flow, Network flow, Ford-Fulkerson, Edmonds-Karp, Push-Relabel, Python.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1.1 Latar Belakang

Teori graf adalah pokok bahasan yang mempunyai banyak terapan sampai saat ini. Graf dapat digunakan untuk merepresentasikan sebuah masalah agar menjadi sederhana dan mudah untuk dipahami. Salah satu terapan teori graf adalah *network* (jaringan). Jaringan yang sering ditemui dalam kehidupan sehari-hari adalah jaringan pendistribusian barang, jaringan aliran air, jaringan aliran listrik, jaringan telepon, dan jaringan komputer. Pengembangan metode dilakukan untuk memudahkan dalam menyelesaikan permasalahan jaringan seperti pencarian rute terpendek, penyusunan jadwal untuk mendapatkan waktu optimal dan mencari lintasan dalam pendistribusian barang yang maksimal dengan biaya yang lebih kecil. Permasalahan-permasalahan tersebut biasa disebut dengan *network flow*. Aplikasi dari *network flow* diantaranya adalah aliran maksimum (*maximum flow*), *shortest path problem*, dan *minimum cost flow problem* (Ahuja, Magnanti, & Orlin, 1993).

Salah satu permasalahan *network flow* yang kita temui adalah mencari aliran maksimum (*maximum flow*). Aliran maksimum yaitu permasalahan optimasi dengan tujuan mendapatkan hasil maksimal dari jumlah aliran pada sebuah jaringan (*network*) di mana jaringan tersebut mempunyai satu titik sumber (*source*) dan satu titik tujuan (*sink*). Pada sebuah jaringan terdapat kapasitas yang akan membatasi jumlah aliran yang akan melewatinya, yang dapat menyebabkan kebocoran apabila alirannya melebihi kapasitas yang tersedia. Oleh sebab itu aliran maksimal sebuah jaringan perlu ditentukan terlebih dahulu sebelum jaringan tersebut dialiri (Dash, Rahman & Akter, 2019).

Banyak algoritma yang digunakan untuk memecahkan masalah aliran maksimum. Ford dan Fulkerson (1956) memberi algoritma pertama, yang menggunakan *augmenting path*. Edmonds dan Karp (1972) memperkenalkan algoritma Edmonds-Karp dimana algoritma ini menggunakan algoritma *breath first search* dalam pencarian *augmenting path*. Goldberg dan Tarjan (1988) memperkenalkan algoritma *push-relabel* yang mempertahankan fungsi ketinggian

dari aliran awal dan mengubahnya menjadi aliran maksimum menggunakan operasi *push* dan label ulang (*relabel*).

Penelitian terkait algoritma dalam pencarian aliran maksimum telah banyak dilakukan. Kyi dan Naing (2018) menggunakan algoritma Ford-Fulkerson untuk mencari aliran maksimum pada saluran air di Myanmar. Dash, Rahman, dan Akter (2019) dalam penelitiannya membahas tentang algoritma Edmonds-Karp untuk mencari aliran maksimum pada saluran pipa yang memasok gas di Bangladesh. Bhandari dan Khadka (2021) membahas tentang algoritma *push-relabel* untuk aliran maksimum pada perencaanaan jalur evakuasi.

Penggunaan algoritma dalam penerapannya untuk mencari aliran maksimum, diperlukan algoritma yang paling efektif. Oleh karena itu perlu diketahui keunggulan dan kekurangan dari algoritma-algoritma tersebut. Hal inilah yang menjadi pertimbangan peneliti untuk melakukan penelitian dengan judul "**Perbandingan Algoritma Ford-Fulkerson, Edmonds-Karp dan *Push-Relabel* dalam Menentukan Aliran Maksimum**".

1.2 Rumusan Masalah

Rumusan masalah yang akan dibahas pada penelitian ini adalah sebagai berikut:

1. Bagaimana menyelesaikan *maximum flow problem* (masalah aliran maksimum) dengan menggunakan algoritma Ford-Fulkerson, algoritma Edmonds-Karp, dan algoritma *Push-Relabel*?
2. Bagaimana menentukan algoritma paling efektif dari hasil perbandingan algoritma Ford-Fulkerson, algoritma Edmonds-Karp, dan algoritma *Push-Relabel* dalam memecahkan masalah aliran maksimum?

1.3 Batasan Masalah Penelitian

Adapun Batasan masalah pada penelitian ini adalah *network* yang digunakan diperoleh dari jurnal dan menggunakan 4 *network* yaitu *network* medium (10 dan 13 titik) dan *network* yang relatif besar (24 dan 46 titik), yang akan di selesaikan dengan algoritma Ford-Fulkerson, algoritma Edmonds-Karp, dan algoritma *Push-Relabel* dengan pengerjaan manual dan menggunakan program

python dan kemudian akan di lakukan perbandingan jumlah iterasi yang dibutuhkan tiap-tiap algoritma serta kompleksitas masing-masing algoritma dalam menyelesaikan masalah aliran maksimum berdasarkan referensi.

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah diatas maka tujuan penelitian ini yaitu:

1. Mendapatkan penyelesaian *maximum flow problem* (masalah aliran maksimum) dengan menggunakan algoritma Ford-Fulkerson, algoritma Edmonds-Karp, dan algoritma *Push-Relabel.*
2. Membandingkan algoritma Ford-Fulkerson, algoritma Edmonds-Karp, dan algoritma *Push-Relabel* untuk menentukan algoritma yang paling efektif dalam memecahkan masalah aliran maksimum.

1.5 Manfaat Penelitian

Manfaat penelitian ini adalah menambah wawasan mengenai algoritma Ford-Fulkerson, algoritma Edmonds-Karp, dan algoritma *Push-Relabel* serta mengetahui perbedaan algoritma Ford-Fulkerson, algoritma Edmonds-Karp, dan algoritma *Push-Relabel* sehingga dapat diketahui keunggulan dan kekurangan masing-masing algoritma.

BAB II

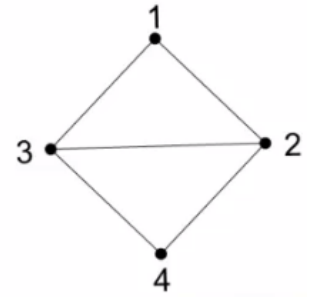
TINJAUAN PUSTAKA

2.1 Terminologi Dasar Graf

Teori graf adalah pokok bahasan yang mempunyai banyak terapan sampai saat ini. Pemakaian teori graf telah banyak dirasakan dalam berbagai ilmu, antara lain: optimisasi jaringan, ekonomi, psikologi, genetika, riset operasi, dan lain-lain. Graf dapat digunakan untuk merepresentasikan sebuah masalah agar menjadi sederhana dan mudah untuk dipahami.

Pada tahun 1736 graf pertama kali digunakan pada permasalahan jembatan Konigsberg, di mana sungai yang mengalir mengelilingi pulau Kneiphof yang bercabang menjadi dua anak sungai dan terdapat tujuh jembatan yang menghubungkan daratan. Leonard Euler adalah orang yang pertama kali menemukan jawaban dari masalah ini dengan menggunakan graf, di mana daratan dinyatakan sebagai titik dan jembatan dinyatakan sebagai sisi dari graf tersebut. Penelitian yang dilakukan Euler inilah yang mendasari munculnya teori graf (Munir, 2010).

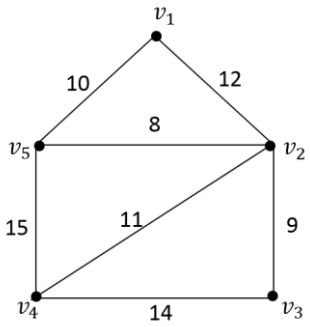
Definisi 2.1 (Munir, 2010) Graf G didefinisikan sebagai pasangan himpunan (V, E), yang dituliskan dengan notasi $G = (V, E)$, yang dalam hal ini V adalah himpunan tidak-kosong dari simpul-simpul (vertex atau node) dan E adalah himpunan sisi (*edges* atau *arcs*) yang menghubungkan sepasang simpul. Pada graf G terdapat orde yang diberi symbol $|V|$ yaitu jumlah titik dari graf G dan *size* (ukuran) yang diberi simbol $|E|$ yaitu jumlah sisi pada draf G. Simpul-simpul pada graf biasanya diberi simbol angka ataupun huruf sedangkan sisinya adalah pasangan yang menghubungkan kedua simpul. Gambar 2.1 berikut adalah contoh dari graf G.



Gambar 2.1 Graf G

Definisi 2.2 (Munir, 2010) Graf sederhana adalah graf yang tidak mempunyai sisi gelang (loop) maupun sisi ganda. Contoh graf sederhana seperti pada Gambar 2.1.

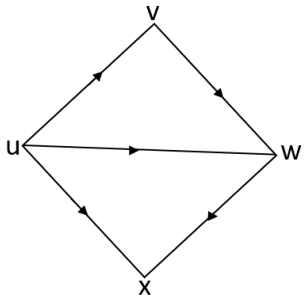
Definisi 2.3 (Bondy dan Murty, 1976) Graf yang diberi bilangan real (bobot) pada setiap sisinya dinamakan graf berbobot.



Gambar 2.2 Graf berbobot

Pada Gambar 2.2 bobot dari sisi-sisinya dinotasikan dengan $wt(v_i v_j)$. Sehingga dapat dilihat bobot masing-masing sisinya adalah $wt(v_1v_2) = 12$, $wt(v_1v_5) = 10$, $wt(v_2v_3) = 9$, $wt(v_3v_4) = 14$, $wt(v_2v_4) = 11$, $wt(v_2v_5) = 8$ dan $wt(v_4v_5) = 15$.

Definisi 2.4 (Munir, 2010) graf berarah adalah graf yang setiap sisinya mempunyai orientasi arah. Pada graf berarah $(u, v) \neq (v, u)$, untuk sisi (u, v) titik u adalah titik asal dan titik v adalah titik tujuan.

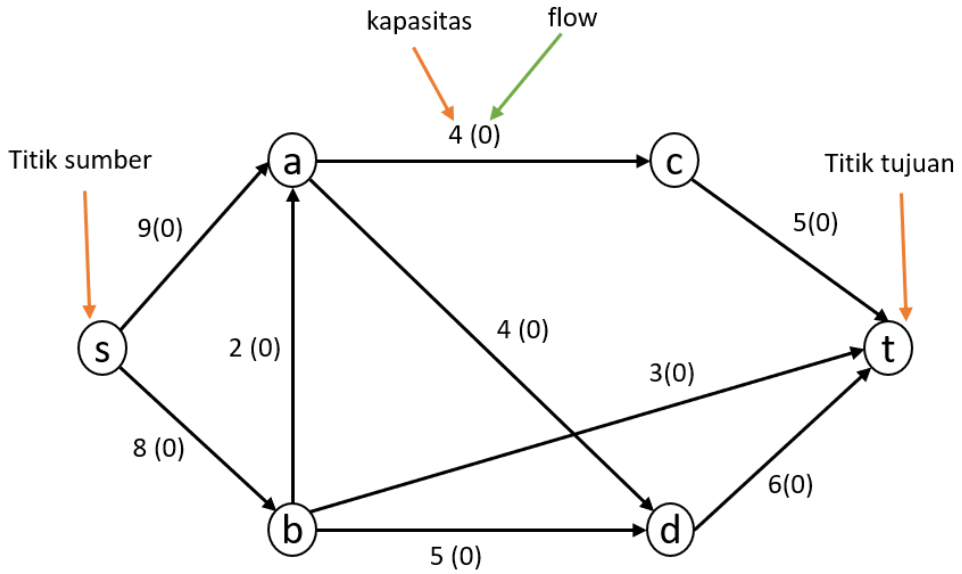


Gambar 2.3 Graf berarah

2.2 *Maximum Flow Problem*

Aliran maksimum adalah sebuah model aliran yang digunakan untuk mengetahui nilai maksimum di dalam sebuah jaringan. Misalkan $c(u, v)$ adalah kapasitas pada sisi berarah (u, v) (Kyi dan Naing, 2018).

Definisi 2.5 Salah satu kajian dalam teori graf adalah jaringan (*network*). Jaringan ialah graf yang memiliki arah dan juga bobot dengan dua titik (simpul) khusus yaitu titik sumber (*source*) dan titik tujuan (*sink*) di mana setiap sisi $e = (u, v)$ mempunyai nilai kapasitas $c(u, v)$ non-negatif. Jaringan aliran (*network flow*) adalah graf yang memiliki aliran (*flow*) $f(u, v)$pada setiap sisinya, dimana memenuhi syarat $0 \leq f(u, v) \leq c(u, v)$ (Dash, Rahman & Akter, 2019)



Gambar 2.4 *Network* (Jaringan)

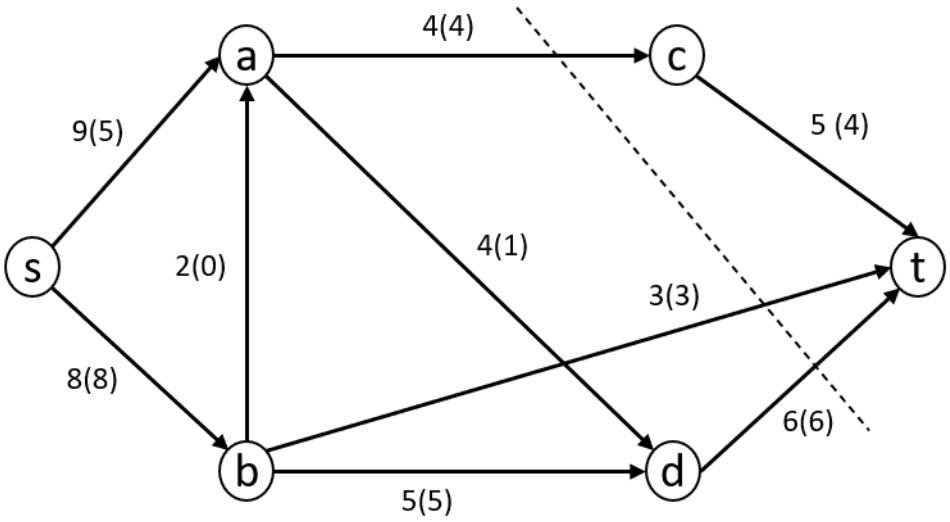
Definisi 2.6 Titik sumber (*source*) adalah titik di mana tidak ada aliran yang yang masuk (tidak ada panah kedalam titik sumber) dan titik tujuan (*sink*) adalah titik dimana tidak ada aliran yang keluar (tidak ada panah keluar dari titik tujuan) (Alsalami dan Rushdi, 2021).

Definisi 2.7 Jaringan sisa (*residual network)* yaitu jaringan yang sisi-sisinya mempunyai kapasitas bernilai positif dan dapat menerima aliran ($flow$). Misalkan $N = (V, E)$ adalah sebuah aliran jaringan (*network flow*) dengan sumber s dan tujuan t, dimana $f(u, v)$ adalah aliran (*flow*) dari N dan titik $u, v \epsilon V$. Jumlah aliran tambahan yang dapat didorong dari u ke v sebelum melebihi kapasitas $c(u, v)$ adalah kapasitas sisa dari (u, v) yang diperoleh dari $c_f(u, v) = c(u, v) - f(u, v)$ (Dash, Rahman & Akter, 2019).

Definisi 2.8 Jalur penambah (*augmenting path*) yaitu jalur dari sumber ke tujuan pada jaringan sisa *network* $N = (V, E)$. Jalur penambah akan digunakan untuk mengubah aliran pada sisi tertentu di $N = (V, E)$ sehingga aliran (*flow*) akan meningkat dari titik sumber ke titik tujuan (Dash, Rahman & Akter, 2019).

Definisi 2.9 *Cut* (pemotongan), pemotongan pada jaringan adalah sebuah partisi titik sedemikian sehingga $X \cup \bar{X} = V,\ X \cap \bar{X} = \emptyset,$ dimana $s \epsilon X$ dan $t \epsilon \bar{X}$. Kapasitas cut (s,t) adalah total semua kapasitas busur (sisi) dari s ke t yang di notasikan $C(X,\bar{X})$. Cut (s,t) adalah *minimum cut*, jika kapasitasnya adalah minimum diantara semua *cut* $s-t$ (Chand dan Dhamala, 2022).

Theorema 2.1 Untuk setiap jaringan (*network*), nilai aliran maksimum (*maximum flow*) dari titik sumber ke titik tujuan adalah sama dengan kapasitas *minimal cut* untuk semua cut sepanjang *s* dan *t.*



Gambar 2.5 *Max flow min cut***.**

Dari gambar tersebut diperoleh minimum cut jaringanya yaitu:

$X = \{s,a,b,d\}, \bar{X} = \{c,t\}, (X,\bar{X}) = \{(a,c),(b,t),(d,t)\}$ dan

$C(X,\bar{X}) = C(a,c) + C(b,t) + C(d,t) = 4 + 3 + 6 = 13$

Jadi minimum cut jaringan tersebut adalah 13.

2.3 Algoritma Ford-Fulkerson

Algoritma Ford-Fulkerson adalah algoritma yang dibuat oleh Ford, L. R. Jr. dan Fulkerson, D.R. pada tahun 1956. Algoritma ini digunakan untuk menghitung aliran maksimum dalam *network flow* (aliran jaringan), di mana pada *network* tersebut terdapat satu titik sumber dan satu titik tujuan. Ide dari algoritma

ini adalah selama terdapat jalur dari sumber ke tujuan, maka aliran dapat dikirim di sepanjang jalur tersebut. Jalur dengan kapasitas yang tersedia disebut jalur penambah (*augmenting path*) (Kyi dan Naing, 2018).

Algortima Ford-Fulkerson terdiri dari dua langkah yaitu langkah pertama dimulai dengan mencari aliran (*flow*) dengan kapasitas yang tersedia yang akan menjadi jalur augmentasi (jalur penambah) yang akan dilabeli, kemudian langkah selanjutnya dengan menyesuaikan alirannya sampai tidak ada lagi jalur penambah dalam aliran jaringan tersebut (Dumlao, Banal, & Livara, 2021).

Proses kerja algoritma *Ford-Fulkerson*

Langkah 1: Mulai dengan inisialisasi *flow* $f = 0$

Langkah 2: Cari jalur penambah P. Jika tidak terdapat jalur maka algortima berhenti, *flow* maksimal. Jika terdapat jalur penambah lanjutkan ke Langkah 3. Pada Langkah kedua dilakukan proses pelabelan.

Langkah 3: Ganti aliran lama f dengan aliran baru $\hat{f}$ dengan menggunakan persamaan dibawah.

$$\hat{f}(x,y) = \begin{cases} f(x,y) + \varepsilon & ;\ jika\ (x,y)\ menuju\ ke\ titik\ P \\ f(x,y) - \varepsilon & ;\ jika\ (x,y) keluar\ dari\ titik\ P \\ f(x,y) & ;\ lainnya \end{cases}$$

Kembali ke langkah 2.

Pada langkah 2 dilakukan proses pelabelan. Pada tahap ini dimulai dengan memberi label titik sumber s. Terdapat dua jenis label yaitu label (z^+, δ) atau (z^-, δ), dimana z adalah titik dan δ adalah bilangan riil positif. Jika titik u dapat diberi label (z^+, δ) atau (z^-, δ), artinya bahwa dapat dicari jalur $(s,u) - P$, dengan syarat berikut

i. $c(x,y) - f(x,y) \geq \delta$, untuk setiap (x,y) menuju ke titik P
ii. $f(y,x) \geq \delta$, untuk setiap (x,y) keluar dari titik P

Selain itu jika u diberi label(z^+, δ) maka (z,u) adalah sisi dari P dan jika diberi label (z^-, δ) maka (u,z) sisi dari P.

Proses pelabelan (untuk mencari jalur penambah)

Langkah 1 : Label titik sumber dengan $s, (-\varepsilon(s) = \infty)$

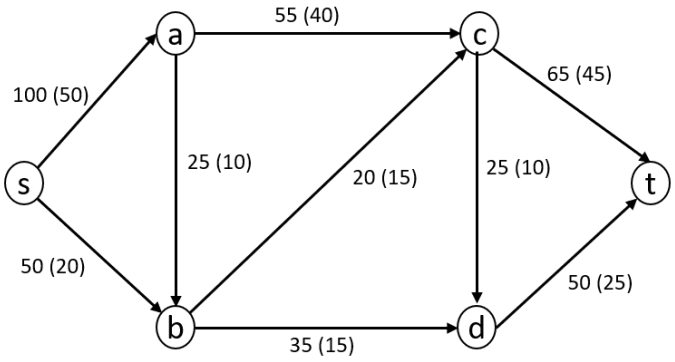
Langkah 2 : Pilih sembarang titik *x* yang berlabel dan belum dipindai. Misalkan diberi label $(z^+, \varepsilon(x))$ atau $(z^-, \varepsilon(x))$. Pindai *x* dan tetapkan label sesuai aturan berikut:

i. Jika (x, y) adalah sisi dengan $f(x, y) < c(x, y)$ dan y tidak terlabel, beri y dengan label $(x^+, (\varepsilon(y))$, dimana $\varepsilon(y) = \min\{\varepsilon(x), c(x, y) - f(x, y)\}$.

ii. Jika (x, y) adalah sisi dengan $f(x, y) > 0$ dan y tidak terlabel, beri y dengan label $(x^-, (\varepsilon(y))$, dimana $\varepsilon(y) = \min\{\varepsilon(x), f(y, x)\}$.

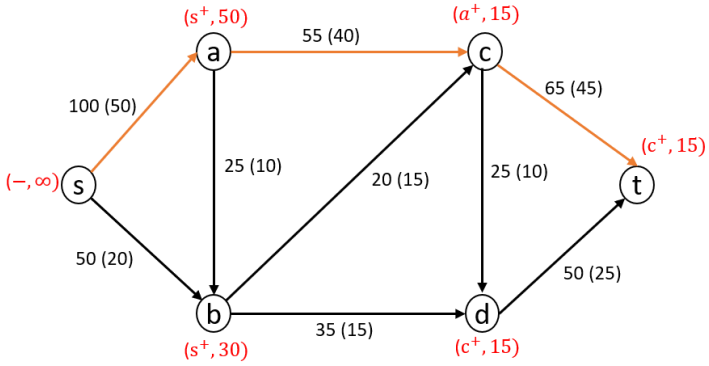
Langkah 3: Ulangi langkah 2 sampai ke titik tujuan atau hingga tidak terdapat lagi titik yang dapat diberi label. Dalam kasus ini tidak terdapat lagi jalur penambah.

Sebagai contoh diberikan network berikut (Kyi dan Naing, 2018).



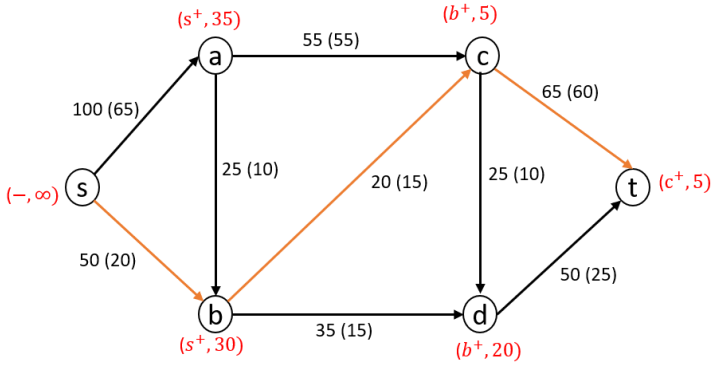
Gambar 2.6 *Network* dengan 6 titik

Langkah pertama: karena network telah memiliki flow maka di lakukan proses pelabelan pada jalur penambah



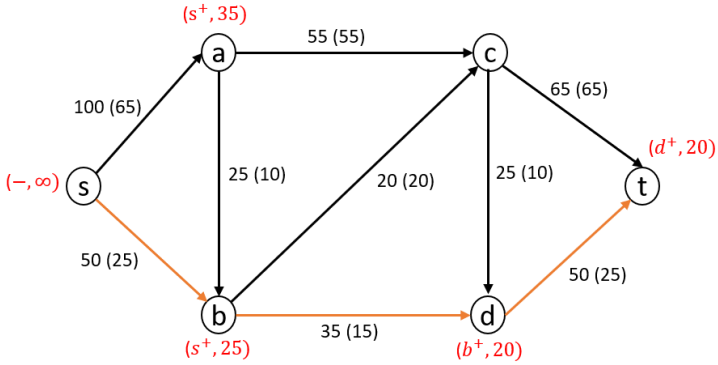
Gambar 2.7 Pelabelan jalur penambah $s - a - c - t$.

Langkah kedua: ganti aliran lama dengan aliran baru kemuadian dilakukan pelabelan pada jalur penambah.



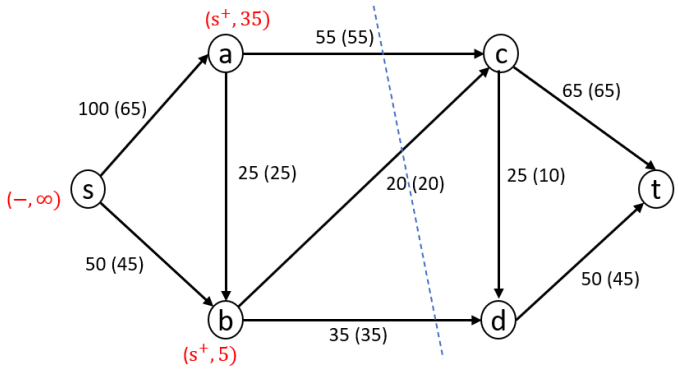
Gambar 2.8 Pelabelan jalur penambah $s - b - c - t$

Langkah ketiga: ganti aliran lama dengan aliran baru kemuadian dilakukan pelabelan pada jalur penambah.



Gambar 2.9 Pelabelan jalur penambah $s - b - d - t$.

Langkah keempat : ganti aliran lama dengan aliran baru. Pada tahap ini aliran maksimum sehingga tidak terdapat jalur penambah.



Gambar 2.10 *Network* hasil.

Setelah flow maksimum tidak terdapat lagi jalur penambah maka selanjutnya adalah menghitung maksimum flow. Dari gambar tersebut diperoleh minimum cut jaringanya yaitu:

$$X = \{s, a, b, \}, \bar{X} = \{c, d, t\}, (X, \bar{X}) = \{(a, c), (b, c), (b, d)\} \text{ dan}$$

$$C(X, \bar{X}) = C(a, c) + C(b, c) + C(b, d) = 55 + 20 + 35 = 110$$

Jadi minimum cut = maksimum flow jaringan tersebut adalah 110.

2.4 Algoritma Edmonds-Karp

Algoritma Edmonds-Karp adalah algoritma yang diperkenalkan oleh Jack Edmond dan dan Richard Karp pada tahun 1972. Algoritma ini adalah pengembangan dari algoritma Ford-Fulkerson dimana dalam algoritma ini menggunakan *Breath First Search* dalam pencarian jalur augmentasi untuk menemukan lintasan terpendek (Dumlao, Banal, & Livara, 2021).

Algoritma BFS atau biasa juga disebut sebagai algoritma pencarian melebar yaitu algoritma yang melakukan pencarian secara melebar mulai dari titik awal kemudian dilanjutkan dengan mengunjungi titik-titik yang bertetangga dengannya dan belum dikunjungi, seterusnya sampai seluruh titik berhasil dikunjungi.

Adapun langkah-langkah algoritma Edmonds-Karp yaitu:

$$[G = (V, E), titik\ 1(= s), \dots, n(= t), sisi\ (i, j), c_{ij}\ ]$$

Algoritma ini menghitung aliran maksimum dalam jaringan G dengan sumber s dan tujuan t, dengan kapasitas $c_{ij} > 0$ dari sisi (i, j).

Input: $n, s = 1, t = n, sisi\ (i, j)\ dari\ G, c_{ij}$

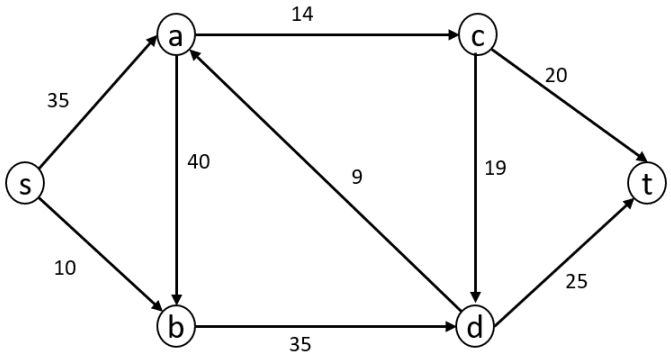
Output : *Maximum flow* f pada G

a. Inisialisasi flow sebesar 0
b. Identifikasi jalur penambah dengan menggunakan algoritma BFS
c. Setelah ditemukan jalur penambah, lakukan proses pelabelan pada jalur penambah seperti algoritma Ford-Fulkerson.
d. Setelah jalur dilabeli, ganti aliran dengan aliran baru.
e. Ulangi langkah 2 sampai 4 hingga tidak terdapat jalur penambah, kemudian hitung aliran maximum.

Adapun Langkah-langkah algoritma BFS yaitu:

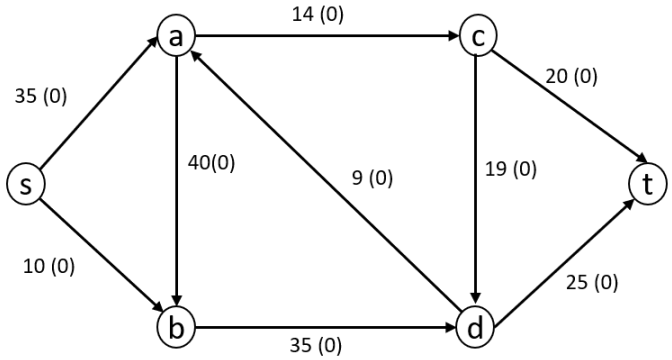
a. Masukkan simpul awal kedalam antrian.
b. Ambil simpul dari awal antrian, lalu cek apakah simpul merupakan solusi
c. Jika simpul merupakan solusi, pencarian selesai.
d. Jika simpul bukan solusi, masukkan seluruh simpul yang bertetangga dengan simpul tersebut kedalam antrian.
e. Jika antrian kosong dan setiap simpul sudah dicek, pencarian selesai dan solusi tidak ditemukan.
f. Ulangi pencarian dari langkah 2.

Berikut adalah contoh dari algoritma Edmonds-Karp dengan *network* yang memiliki 6 titik (Dash, Rahman, dan Akter 2019).



Gambar 2.11 *Network* hasil.

Langkah 1: inisialisasi flow sebesar 0



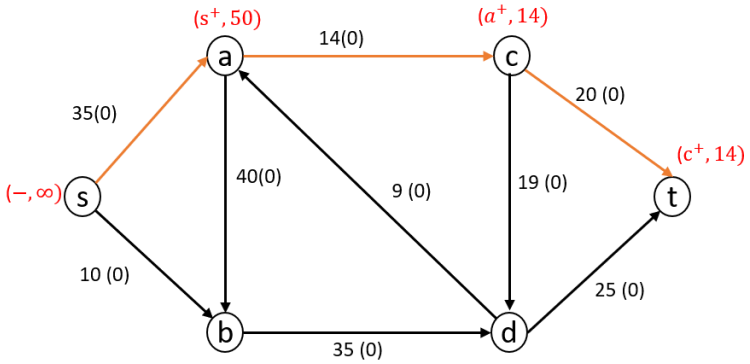
Gambar 2.12 *Network* yang diberi flow sebesar 0.

Langkah 2: cari jalur penambah dengan algoritma BFS

Tabel 2. 1 jalur penambah dengan algoritma BFS iterasi 1

Antrian	Simpul bertetangga yang dapat dikunjungi	Parent map (path)	Simpul yang telah dikunjungi
s	a b	$a-s$ $b-s$	$s\ a\ b$
a b	c	$c-a$	$s\ a\ b\ c\ d$
	d	$d-b$	
c	t	$t-c$	$s\ a\ b\ c\ d\ t$
d	t	$t-d$	
Jalur penambah s − a − c − *t*			

Langkah 3: lakukan proses pelabelan pada jalur penambah



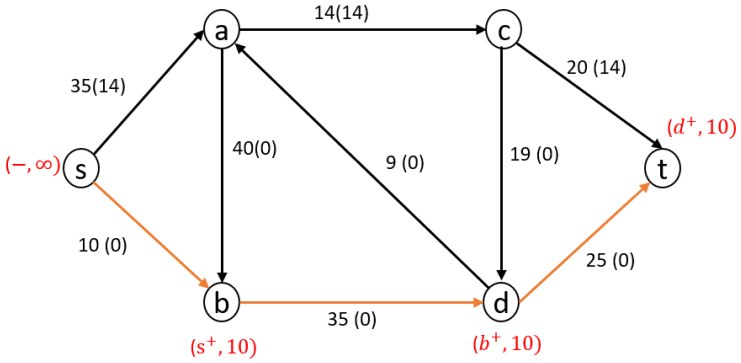
Gambar 2.13 Pelabelan jalur penambah $s - a - c - t$.

Langkah 2: cari jalur penambah dengan algoritma BFS

Tabel 2.2 jalur penambah dengan algoritma BFS iterasi 2

Antrian	Simpul bertetangga yang dapat dikunjungi	Parent map (path)	Simpul yang telah dikunjungi
s	a b	$a - s$ $b - s$	$s\ a\ b$
b	d	$d - b$	$s\ a\ b\ d$
d	t	$t - d$	$s\ a\ b\ d\ t$
Jalur penambah s – b – d – *t*			

Langkah 3: lakukan proses pelabelan pada jalur penambah



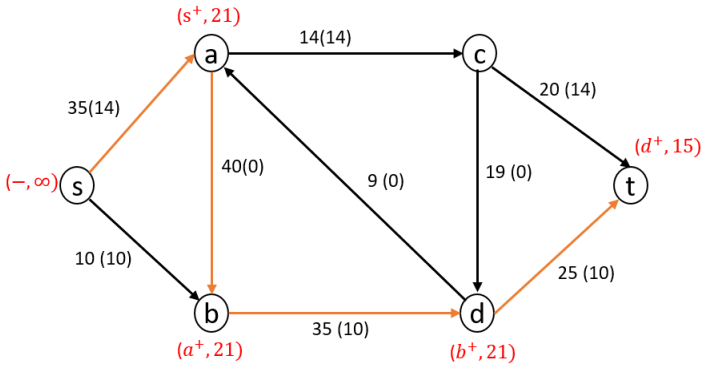
Gambar 2.14 Pelabelan jalur penambah $s - b - d - t$

Langkah 4: cari jalur penambah dengan algoritma BFS

Tabel 2.3 jalur penambah dengan algoritma BFS iterasi 3

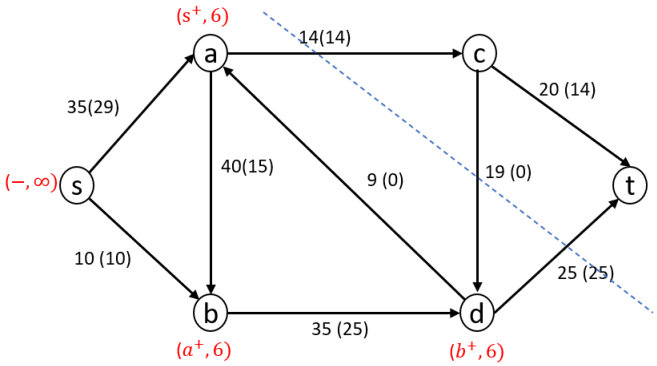
Antrian	Simpul bertetangga yang dapat dikunjungi	Parent map (path)	Simpul yang telah dikunjungi
s	a	$a - s$	$s\ a$
a	b	$b - a$	$s\ a\ b$
b	d	$d - b$	$s\ a\ b\ d$
d	t	$t - d$	$s\ a\ b\ d\ t$
Jalur penambah s – a – b – d – *t*			

Langkah 5: lakukan proses pelabelan pada jalur penambah



Gambar 2.15 Pelabelan jalur penambah $s - a - b - d - t$

Langkah 6: flow maksimum, tidak terdapat lagi jalur penambah



Gambar 2.16 *Network* hasil

Setelah *flow* maksimum tidak terdapat lagi jalur penambah maka selanjutnya adalah menghitung maksimum *flow*. Dari gambar tersebut diperoleh minimum cut jaringanya yaitu:

$$X = \{s, a, b, d\}, \bar{X} = \{c, t\}, (X, \bar{X}) = \{(a, c), (d, t)\} \text{ dan}$$

$$C(X, \bar{X}) = C(a, c) + C(d, t) = 14 + 25 = 39$$

Jadi minimum *cut* = maksimum *flow* jaringan tersebut adalah 39.

2.5 Algoritma Push-Relabel

Algoritma *push-relabel* adalah algoritma yang dikembangkan oleh Goldberg dan Tarjan pada tahun 1988 untuk menyelesaikan masalah aliran maksimum (Talaie, Mousavi ,& Sayadi, 2021).

Definisi 2.10 *Excess* (kelebihan) Jumlah dalam kendala berlebih disebut kelebihan pada titik v, yang dituliskan sebagai ε. Kelebihan pada titik sumber didefinisikan $\varepsilon = \infty$ yang menunjukkan bahwa titik sumber dapat membuat aliran. Titik dikatakan aktif ketika titik tersebut *excess* (kelebihan), $0 < \varepsilon$ (Firet, 2022).

Definisi 2.11 *Height Function* (fungsi ketingian) untuk aliran pada jaringan N adalah fungsi $h: \rightarrow R_{\geq 0}$, yang memenuhi:

a. $h(s) = |V|$, untuk titik sumber

b. $h(t) = 0$, untuk titik tujuan

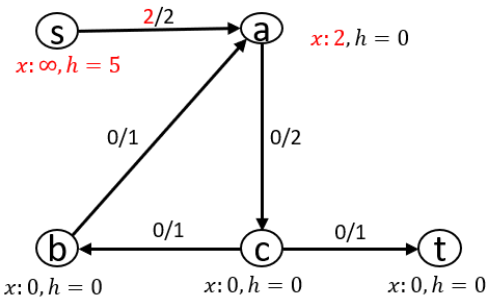
c. $h(u) \leq h(v) + 1$ jika $f(u,v) < c(u,v)$, untuk lainnya

Ketinggian titik u adalah $h(u)$ (Firet, 2022).

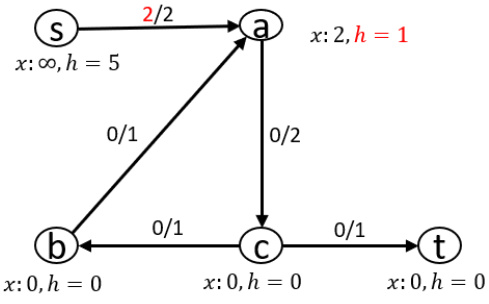
Definisi 2.12 *Push* (dorongan) Dorongan dari u ke v akan meningkatkan flow $f(u,v)$ sebesar δ dan mengurangi flow $f(v,u)$ sebesar δ, dimana $\delta = \min\{\varepsilon(u), c(u,v) - f(u,v)\}$. Dorongan dari u ke v hanya dapat dilakukan apabila $0 < \delta$ dan $h(v) < h(u)$ (Firet, 2022).

Definisi 2.13 *Relabel* **(**Pelabelan) ulang titik u akan menambah $h(u)$ sebesar 1. Pelabelan ulang hanya dilakukan pada titik yang berlebih (*excess*) (Firet, 2022).

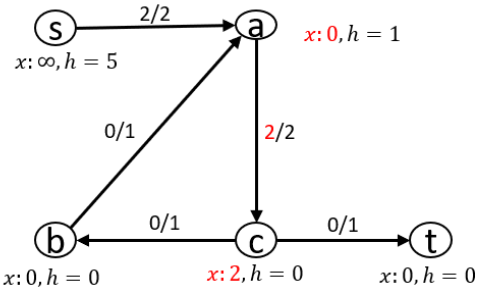
Berikut adalah cara kerja dari algortima *push-relabel* dapat dilihat pada Gambar 2.17 dibawah . Pertama insialisasi $flow$ dan ketinggian pada jaringan (2.17a). Kemudian kita akan melabeli ulang titik a karena tidak ada lagi dorongan dari titik sumber (2.17b). Setelah itu dorong dari titik a ketitik c (2.17c). Selanjutnya karena tidak ada lagi dorongan dari titik a maka titik c diberi label ulang (2.17d). Sekarang dorong dari titik c ke titik t dan juga dorong dari titik c ke titik b (2.17e). Algoritma akan terus mendorong antara titik a, titik b, dan titik c sambal melabel ulang titik-titik tersebut. Sehingga diperoleh $h(a) = 6$ dan akhir dorongan dari titik a ke titik s menyebabkan algoritma berhenti (2.17h).



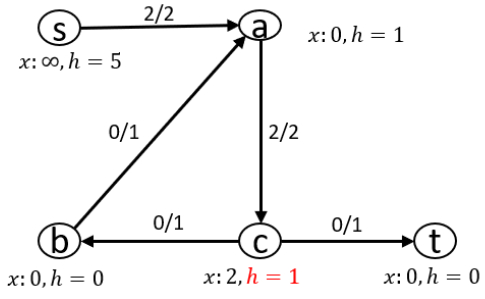
(a) Setelah inisialisasi



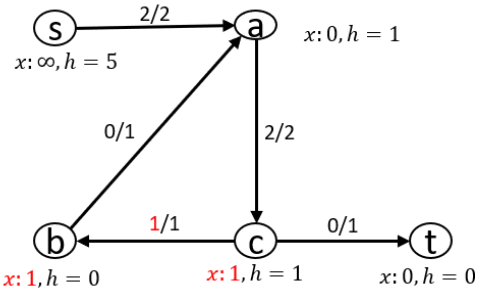
(b) Relabel a



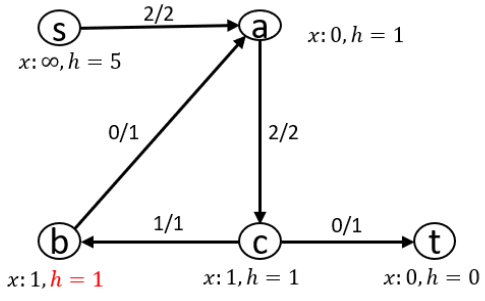
(c) Push dari a ke c



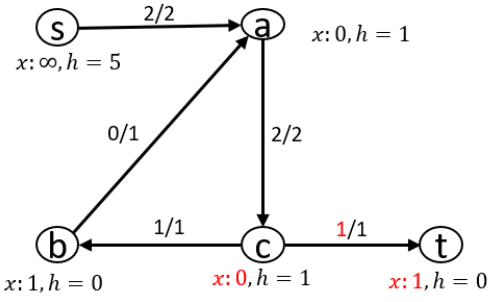
(d) Relabel c



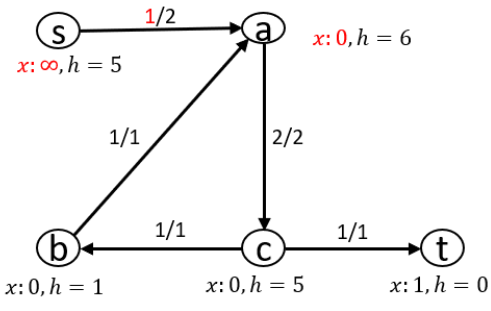
(e) Push dari c ke b



(f) Relabel b



(g) Push dari c ke t



(h)..., Push dari A ke S

Gambar 2.17 Contoh algoritma *Push-Relabel*.

2.6 Kompleksitas Algoritma

Kompleksitas ketiga algoritma untuk menentukan aliran maksimum adalah sebagai berikut (Cormen dkk., 2009):

2.6.1 Kompleksitas Algortima Ford-Fulkerson

Algoritma Ford-Fulkerson memiliki kompleksitas waktu yang tergantung pada cara memilih jalur peningkatan aliran dan cara memperbaharui *network sisa.* Jika menggunakan metode pemilihan jalur peningkatan aliran secara acak, algoritma ini dapat menghasilkan kompleksitas waktu yang tidak pasti, Ford-Fulkerson memiliki kompleksitas waktu terburuk yaitu O(EF), di mana E adalah jumlah tepi dan F adalah nilai aliran maksimum. Namun, algoritma Ford-Fulkerson tidak selalu berhenti dan dapat menghasilkan jawaban yang salah pada *network* tertentu. Namun, jika menggunakan metode pemilihan jalur peningkatan aliran secara optimal seperti dengan algoritma Edmonds-Karp, kompleksitas waktu terbaiknya adalah O(E^2V), di mana E adalah jumlah sisi (edges) dan V adalah jumlah titik (vertices) dalam jaringan.

2.6.2 Kompleksitas Algortima Edmonds-Karp

Algoritma Edmonds-Karp menggunakan strategi pemilihan jalur peningkatan aliran yang mengoptimalkan jarak terpendek (*shortest path*), sehingga kompleksitas waktu terbaiknya adalah O($V\ E^2$), di mana V adalah jumlah titik dan E adalah jumlah sisi dalam graf.

Untuk menunjukkan bahwa algoritma Edmonds-Karp memiliki kompleksitas waktu O($V\ E^2$), kita perlu melihat dua tahap utama dalam algoritma Edmonds-Karp, yaitu:

1. Mencari jalur peningkatan aliran menggunakan algoritma BFS (*Breadth-First Search)* yang mengoptimalkan jarak terpendek.
2. Mengalokasikan aliran dan memperbaharui *network* sisa hingga solusi aliran maksimum ditemukan.

Kompleksitas waktu dari kedua tahap ini adalah sebagai berikut:

1. Pencarian jalur peningkatan aliran:

- Pencarian jalur peningkatan aliran menggunakan algoritma BFS mengoptimalkan jarak terpendek pada *network* sisa dengan mengunjungi setiap simpul dan tepi paling banyak satu kali.
- Kompleksitas waktu algoritma BFS pada *network* dengan jumlah simpul V dan jumlah tepi E adalah O(V + E). Karena algoritma Edmonds-karp menjalankan BFS sebanyak E kali, kompleksitas untuk Langkah ini adalah $O\big(E(E+V)\big) = O(VE)$.

2. Mengalokasikan aliran dan memperbaharui *network* sisa:
    - Setelah jalur peningkatan aliran ditemukan, algoritma akan mengalokasikan aliran pada jalur tersebut dan memperbaharui *network* sisa.
    - Karena setiap jalur peningkatan aliran memiliki paling banyak E tepi, maka setiap iterasi maksimal memiliki kompleksitas waktu O(E).

Oleh karenta itu, dalam Algoritma Edmonds-Karp kompleksitas waktu totalnya adalah $O(VE(E)) = O(VE^2)$.

2.6.3 Kompleksitas Algortima *Push-Relabel*

Algoritma *Push-Relabel* menggunakan teknik pemindahan label dan *Relabel*ing pada simpul-simpul graf untuk mengalokasikan aliran dan menemukan solusi aliran maksimum.

Kompleksitas waktu terbaiknya adalah O(V^3), di mana V adalah jumlah simpul dalam graf.

Untuk menunjukkan bahwa algoritma *Push-Relabel* memiliki kompleksitas waktu $O(V^3)$, kita perlu mempertimbangkan dua tahap utama dalam algoritma *Push-Relabel*, yaitu:

1. Menginisialisasi label dan aliran awal pada simpul-simpul graf.
2. Meningkatkan aliran pada simpul-simpul graf dengan mengalokasikan aliran dari simpul dengan label yang lebih tinggi ke simpul dengan label yang lebih rendah.

Kompleksitas waktu dari kedua tahap ini adalah sebagai berikut:

1. Inisialisasi Label dan Aliran Awal:

- Pada tahap ini, setiap simpul di dalam graf diberi label dan aliran awal yang diinisialisasi dengan kompleksitas waktu O(V).
- Oleh karena itu, kompleksitas waktu tahap ini adalah O(V).

2. Meningkatkan Aliran pada Simpul-Simpul Graf:
   - Tahap ini terdiri dari iterasi sebanyak V^2 kali di mana setiap simpul diproses paling banyak satu kali.
   - Pada setiap iterasi, algoritma akan memeriksa setiap simpul dan mengalokasikan aliran dari simpul dengan label yang lebih tinggi ke simpul dengan label yang lebih rendah.
   - Setiap iterasi memerlukan waktu O(V) untuk memeriksa setiap simpul dan O(V) untuk mengalokasikan aliran ke simpul yang cocok, sehingga kompleksitas waktu total dari tahap ini adalah O(V^3).

Dengan demikian, kompleksitas waktu total dari algoritma *Push-Relabel* adalah O(V^3).