DAFTAR PUSTAKA

Ahuja, R. K. dan Orlin, J. B., (1989). *A Fast and Simple Algorithm for the Maximum Flow Problem*. JSTOR Operation Research, 37, 748-759.

Alhaq, A., Asnawati, R., dan Sutiarso, S. (2014). *Pengaruh Model Pembelajaran Kooperatif Tipe TPS terhadap Kemampuan Komunikasi Matematis Siswa*. Lampung: Universitas Lampung.

Alsalami, O.M, dan Rushdi, A. M. (2021). *A Review of Flow Capasitated Networks: Algorithms, Techniques and Applications.* Asian Journal of Research in Computer Science.

Bulut, M. dan Özcan, E. (2021). *Optimization of electricity transmission by Ford-Fulkerson algorithm*. Elsevier, 28.

Chand, M. B., dan Dharmala, T. N (2002). *A Brief Review on Maximum Flow in Networks with Continuous Time Setting.* Journal of Advanced College of Engineering and Management, 7.

Charland, G. dan Lesniak, L. (1996). *Graphs and Digraphs*. London: Chapman & Hall/CRC.

Dash, P., Rahman, M. M., dan Akter, M.S. (2019). *Developing Algorithm to Obtain the Maximum Flow in a Network Flow Problem.* Jour of Adv in Dynamical and Control System, 11.

Farizal, T., Suyitno, H., dan Darmo. (2014). *Pencarian Aliran Maksimum dengan Algoritma Ford-Fulkerson (Studi Kasus pada Jaringan Listrik Kota Tegal)*. Semarang: Unnes Journal of Mathematics, 3(1), 12-19.

Greenberg, H, J. (1998). *Ford-Fulkerson Max Flow Labeling Algorithm*. Denver: University of Colorado, 1-5.

Gunawan, S. M. dan Santosa, J. (2013). *Analisa Perancangan Gardu Induk Sistem Outdoor 150 kV di Tallasa, Kabupaten Takalar, Sulawesi Selatan*. Surabaya: Jurnal Dimensi Teknik Elektro, 1(1), 37-42.

Hasmawati. (2020). *Pengantar dan Jenis-Jenis Graf*. Makassar: UPT Unhas Press.

Johnsonbaugh, R. (1986). *Discrete Mathematics Revised Edition.* New York: Macmillan Publishing Company.

Khairani, N. dan Sirait, J. (2013). *Membandingkan Kemangkusan Algoritma Dinic dan Algoritma Pelabelan Ford-Fulkerson untuk masalah arus maksimum*.

Medan: Universitas Negeri Medan, 176-190.

Kreyszig, E. (2011). *Advanced Engineering Mathematics.* Columbus : John Wiley & Sons, Inc.

Munir, R. (2003). *Matematika Diskrit.* Bandung : Informatika.

Sa'adah, A., Suyitno, H., dan Dwijanto. (2017). *Optimasi Keuntungan Pakaian dengan Algoritma Titik Interior (Studi Kasus pada PD. Sido Mumbul)*. Semarang: Universitas Negeri Semarang, 6(1), 1-10.

Sujito. (2009). *Sistem Deteksi Hilang Daya pada Gardu Trafo Tiang Jaringan Tegangan Rendah Pelanggan PLN*. Malang: Tekno, 12, 62-75.

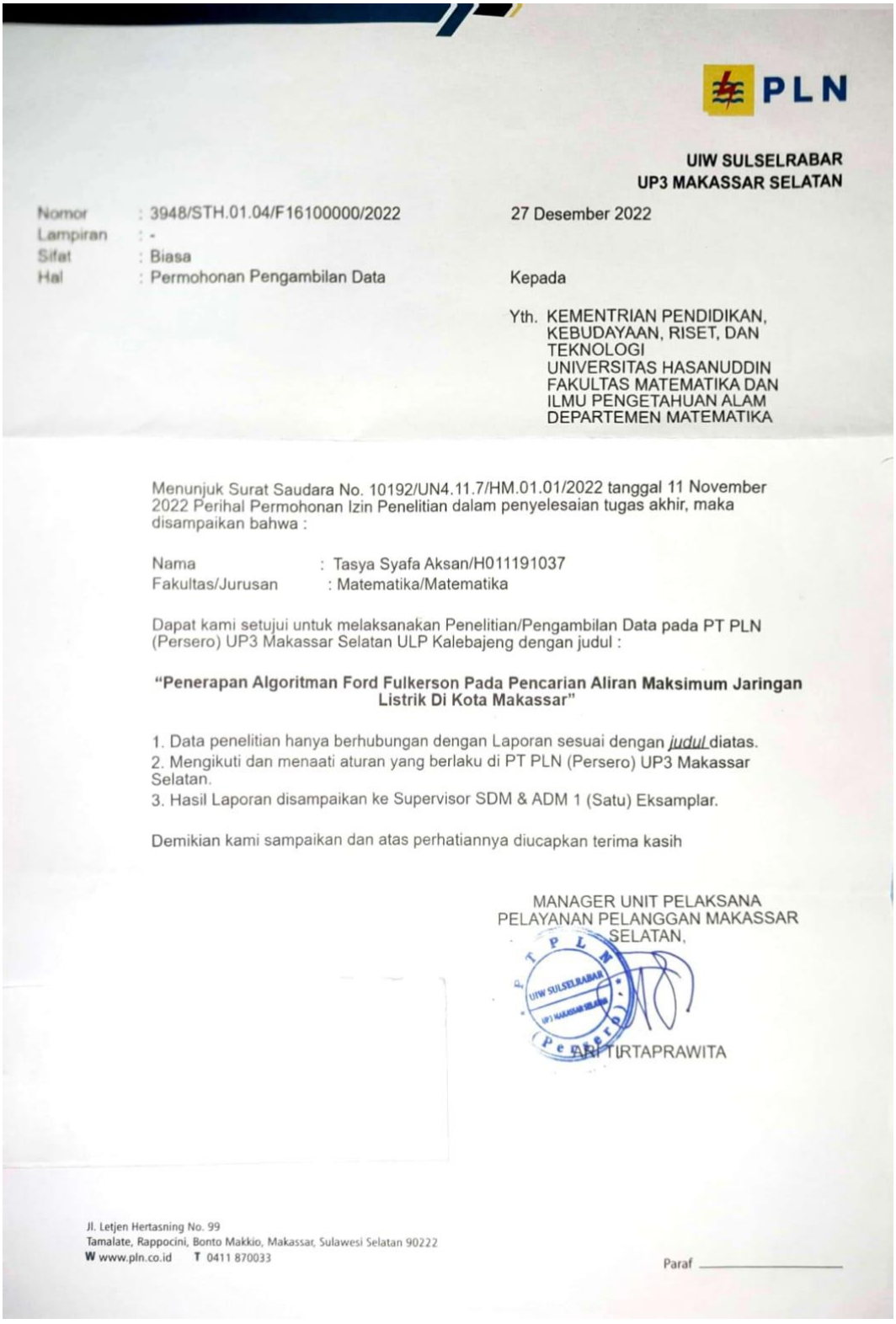
Sumarti, F. (2017). *Pencarian Aliran Maksimum dengan Algoritma Ford-Fulkerson dan Modifikasinya*. Bimaster: Buletin Ilmiah Matematika, Statistika, dan Terapannya, 6(1), 29-36.

Thulasiraman, K. dan Swamy M. N. S. (1992). *Graphs: Theory and Algorithms*. Concordia University Montreal, Canada: John Wiley & Sons.

Viody, Y. (2020). *Server Allocation using Dinic's Algorithm.* Bandung : Institut Teknologi Bandung.

LAMPIRAN

Lampiran 1 Surat Permohonan Pengambilan Data



PLN

UIW SULSELRABAR
UP3 MAKASSAR SELATAN

Nomor : 3948/STH.01.04/F16100000/2022
Lampiran : -
Sifat : Biasa
Hal : Permohonan Pengambilan Data

27 Desember 2022

Kepada

Yth. KEMENTRIAN PENDIDIKAN, KEBUDAYAAN, RISET, DAN TEKNOLOGI
UNIVERSITAS HASANUDDIN
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
DEPARTEMEN MATEMATIKA

Menunjuk Surat Saudara No. 10192/UN4.11.7/HM.01.01/2022 tanggal 11 November 2022 Perihal Permohonan Izin Penelitian dalam penyelesaian tugas akhir, maka disampaikan bahwa :

Nama : Tasya Syafa Aksan/H011191037
Fakultas/Jurusan : Matematika/Matematika

Dapat kami setujui untuk melaksanakan Penelitian/Pengambilan Data pada PT PLN (Persero) UP3 Makassar Selatan ULP Kalebajeng dengan judul :

"Penerapan Algoritman Ford Fulkerson Pada Pencarian Aliran Maksimum Jaringan Listrik Di Kota Makassar"

1. Data penelitian hanya berhubungan dengan Laporan sesuai dengan *judul* diatas.
2. Mengikuti dan menaati aturan yang berlaku di PT PLN (Persero) UP3 Makassar Selatan.
3. Hasil Laporan disampaikan ke Supervisor SDM & ADM 1 (Satu) Eksamplar.

Demikian kami sampaikan dan atas perhatiannya diucapkan terima kasih

MANAGER UNIT PELAKSANA
PELAYANAN PELANGGAN MAKASSAR SELATAN,

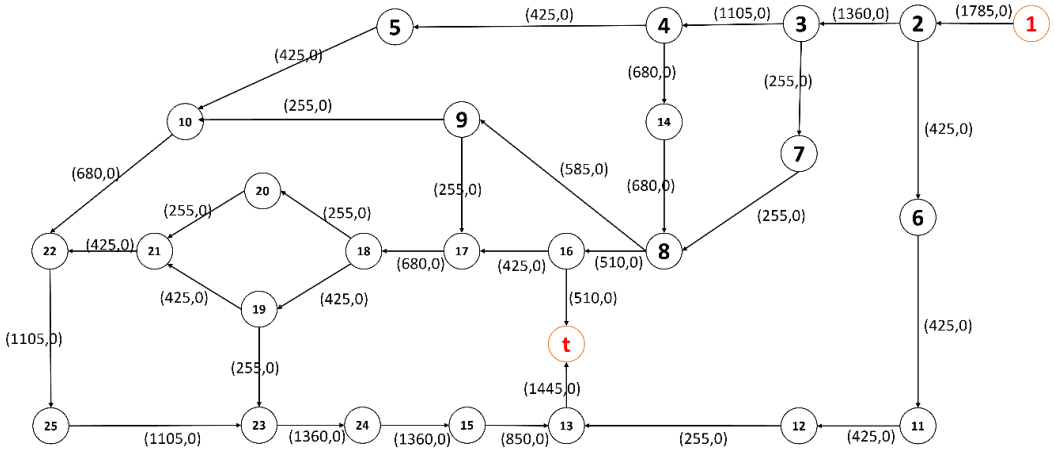
ARI TURTAPRAWITA

Jl. Letjen Hertasning No. 99
Tamalate, Rappocini, Bonto Makkio, Makassar, Sulawesi Selatan 90222
W www.pln.co.id T 0411 870033

Paraf _____

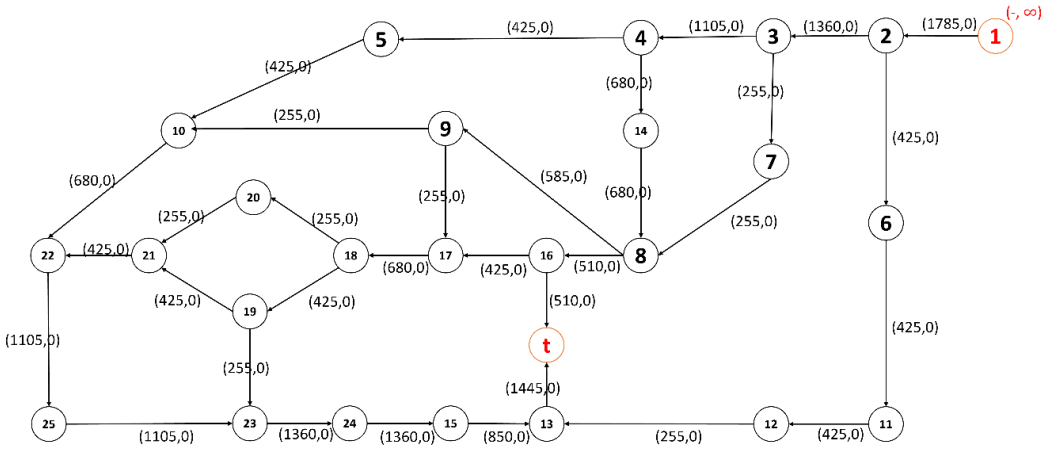
Lampiran 2 Algoritma Ford Fulkerson secara manual dengan tujuan Mall Panakkukang

Diberi aliran 0 pada tiap sisi dengan $f(e) = 0$. Aliran awal tiap sisi bernilai 0 ampere.



Gambar L. 2. 1 Aliran awal tujuan Mall Panakkukang

s dilabeli oleh (-,∞) dan tandai simpul dengan “tidak berlabel”. Simpul yang lain belum dilabeli karena belum ada ditingkatkan aliran.

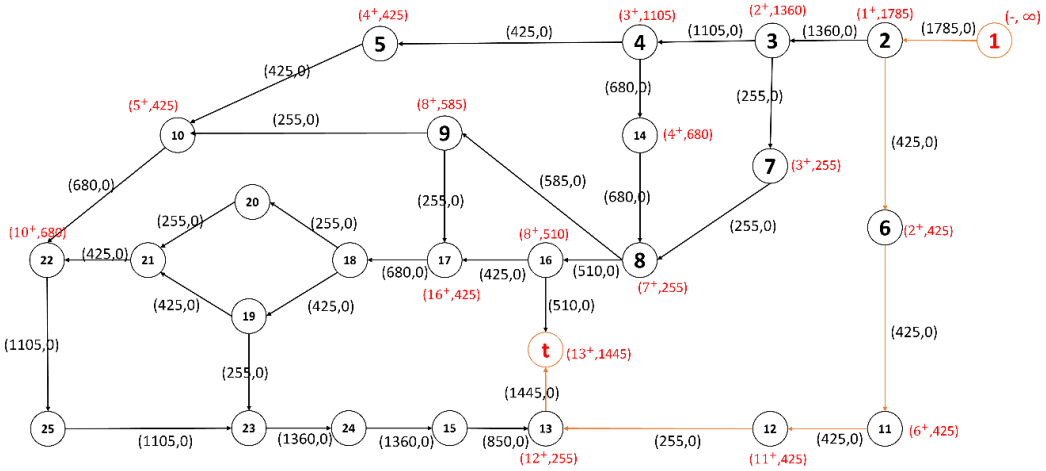


Gambar L. 2. 2 Labeli 1 (sumber)

Selanjutnya pemberian label dan aliran

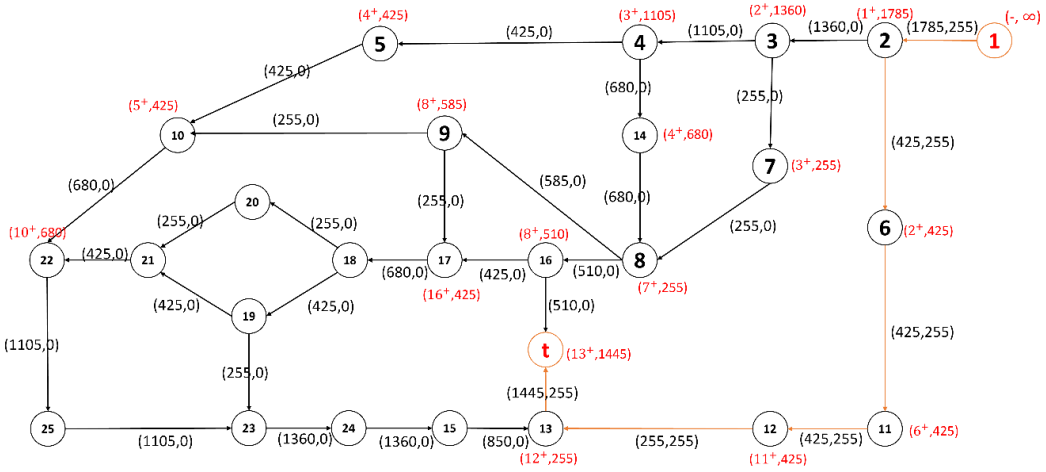
Iterasi 1

- Aliran akan discan dan diberi label pada tiap titiknya. Akan dicari *augmenting path* terpendek dari 1 ke t dan didapat *augmenting path* iterasi 1 adalah 1-2-6-11-12-13-t.



Gambar L. 2. 3 *Scan* dan pelabelan *network* iterasi 1

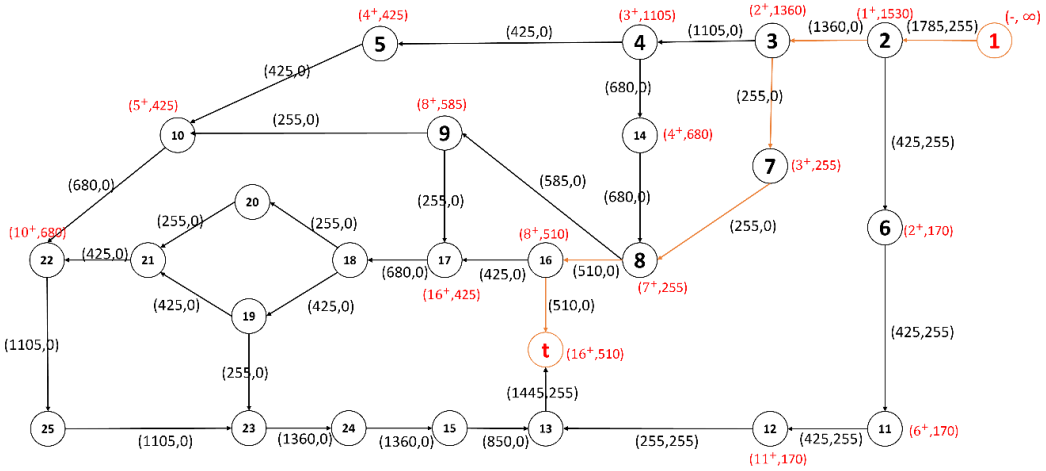
- *Augmenting path* 1-2-6-11-12-13-t dilakukan pembaharuan aliran dan akan dicari nilai maksimum yang dapat mengalir pada tiap sisi di aliran tersebut. Didapat nilai aliran pada iterasi 1 adalah 255 ampere. Pada sisi 12-13 nilainya telah maksimal dan tidak bisa ditingkatkan pada iterasi selanjutnya.



Gambar L. 2. 4 *Network* yang telah diperbaharui pada iterasi 1

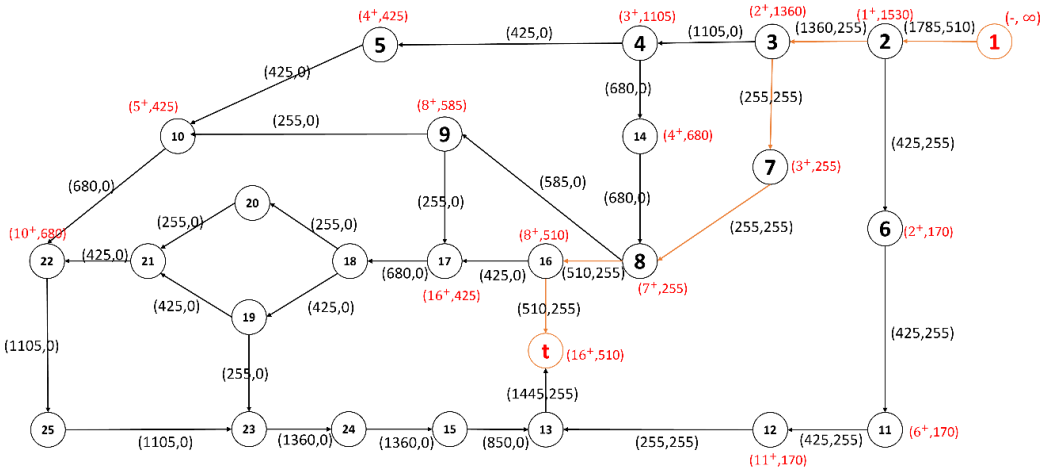
Iterasi 2

- Aliran akan discan dan diberi label pada tiap titiknya. Akan dicari *augmenting path* terpendek dari 1 ke t dan didapat *augmenting path* iterasi 2 adalah 1-2-3-7-8-16-t.



Gambar L. 2. 5 *Scan* dan pelabelan *network* iterasi 2

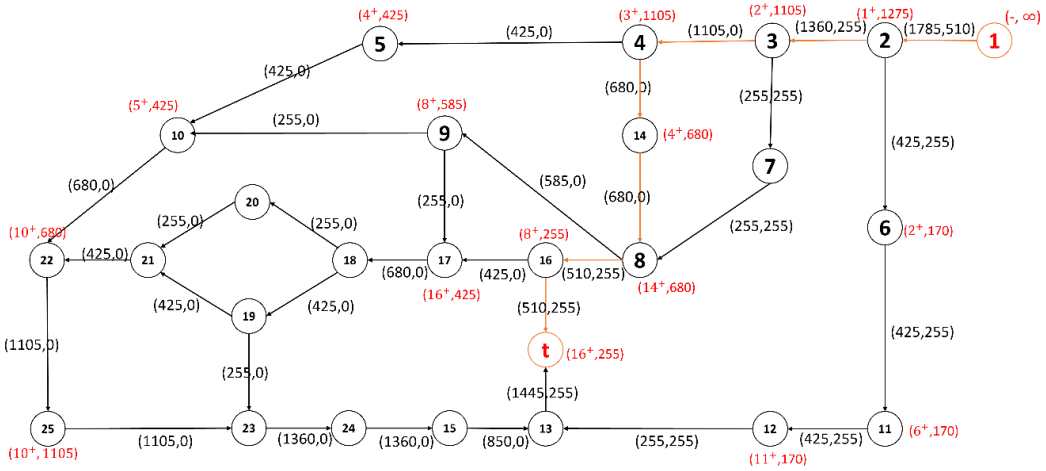
- *Augmenting path* 1-2-3-7-8-16-t dilakukan pembaharuan aliran dan akan dicari nilai maksimum yang dapat mengalir pada tiap sisi di aliran tersebut. Didapat nilai aliran pada iterasi 2 adalah 255 ampere. Pada sisi 3-7 dan 7-8 nilainya telah maksimal dan tidak bisa ditingkatkan pada iterasi selanjutnya.



Gambar L. 2. 6 *Network* yang telah diperbaharui pada iterasi 2

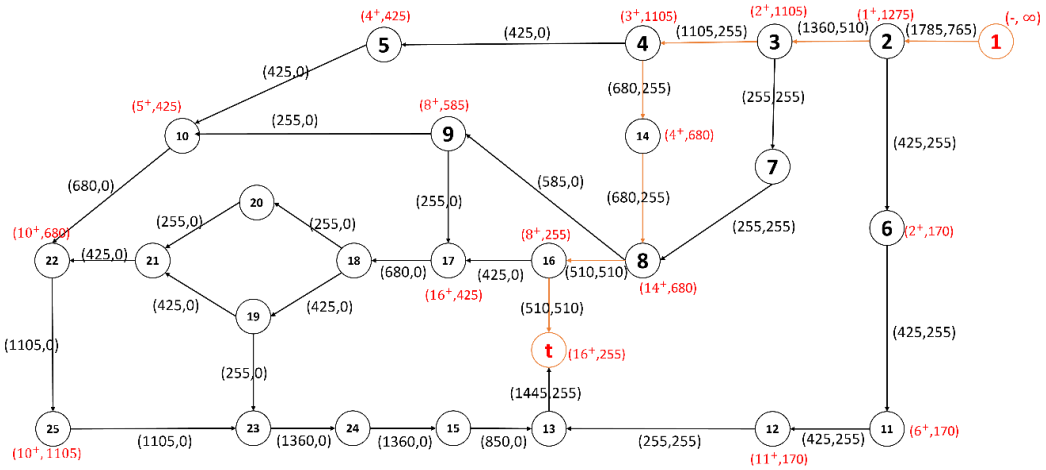
Iterasi 3

- Aliran akan discan dan diberi label pada tiap titiknya. Akan dicari *augmenting path* terpendek dari 1 ke t dan didapat *augmenting path* iterasi 3 adalah 1-2-3-4-14-8-16-t.



Gambar L. 2. 7 *Scan* dan pelabelan *network* iterasi 3

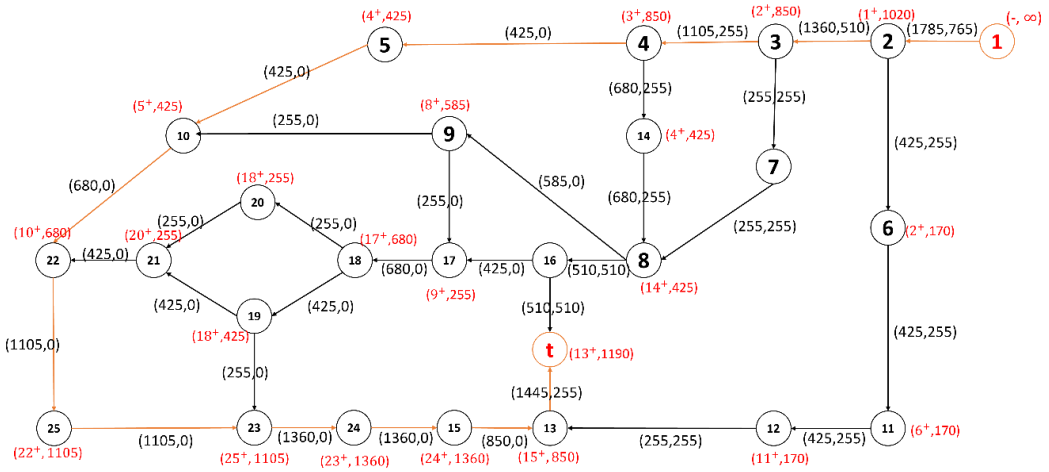
- *Augmenting path* 1-2-3-4-14-8-16-t dilakukan pembaharuan aliran dan akan dicari nilai maksimum yang dapat mengalir pada tiap sisi di aliran tersebut. Didapat nilai aliran pada iterasi 3 adalah 255 ampere. Pada sisi 8-16 dan 16-t nilainya telah maksimal dan tidak bisa ditingkatkan pada iterasi selanjutnya.



Gambar L. 2. 8 *Network* yang telah diperbaharui pada iterasi 3

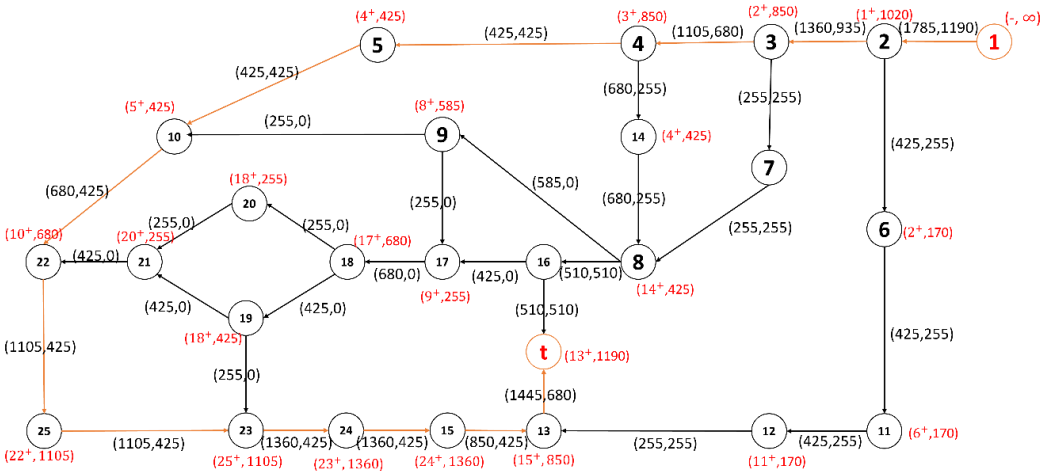
Iterasi 4

- Aliran akan discan dan diberi label pada tiap titiknya. Akan dicari *augmenting path* terpendek dari 1 ke t dan didapat *augmenting path* iterasi 4 adalah 1-2-3-4-5-10-22-25-23-24-15-13-t.



Gambar L. 2. 9 *Scan* dan pelabelan *network* iterasi 4

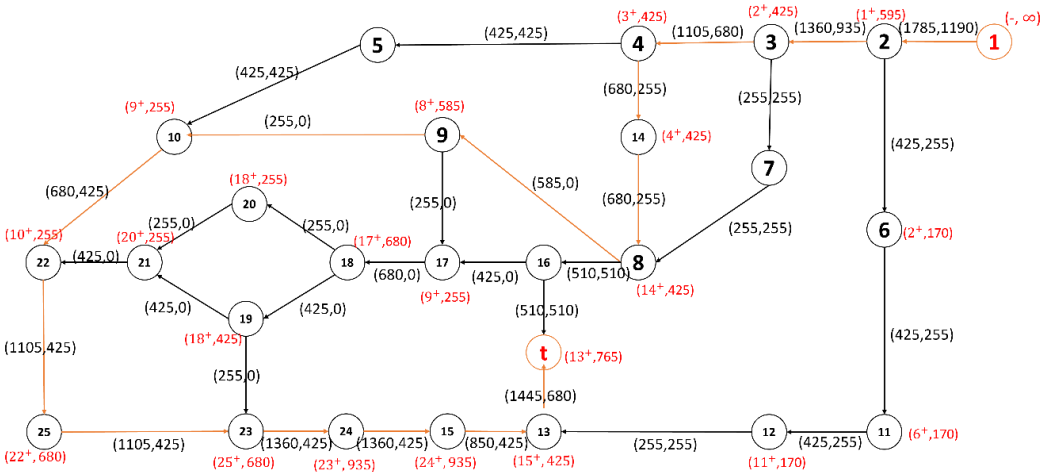
- *Augmenting path* 1-2-3-4-5-10-22-25-23-24-15-13-t dilakukan pembaharuan aliran dan akan dicari nilai maksimum yang dapat mengalir pada tiap sisi di aliran tersebut. Didapat nilai aliran pada iterasi 4 adalah 425 ampere. Pada sisi 4-5 dan 5-10 nilainya telah maksimal dan tidak bisa ditingkatkan pada iterasi selanjutnya.



Gambar L. 2. 10 *Network* yang telah diperbaharui pada iterasi 4

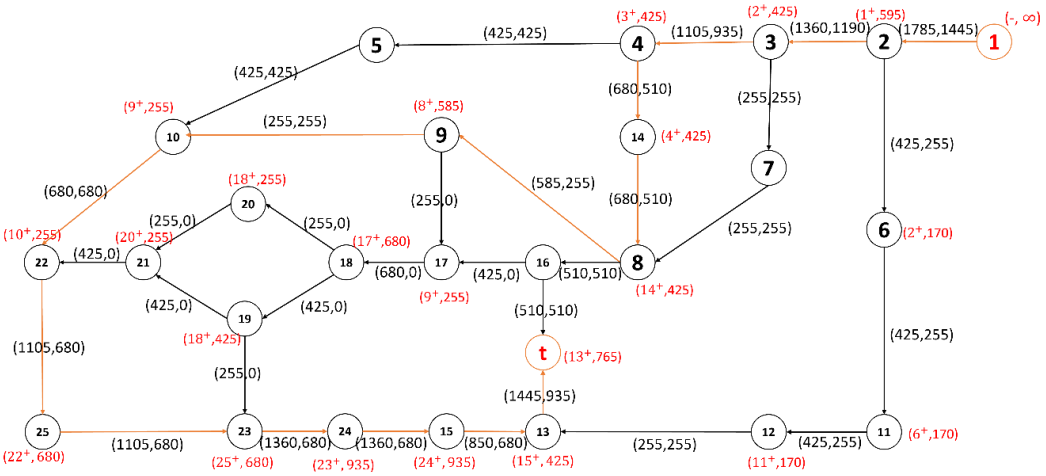
Iterasi 5

- Aliran akan discan dan diberi label pada tiap titiknya. Akan dicari *augmenting path* terpendek dari 1 ke t dan didapat *augmenting path* iterasi 5 adalah 1-2-3-4-14-8-9-10-22-25-23-24-15-13-t.



Gambar L. 2. 11 *Scan* dan pelabelan *network* iterasi 5

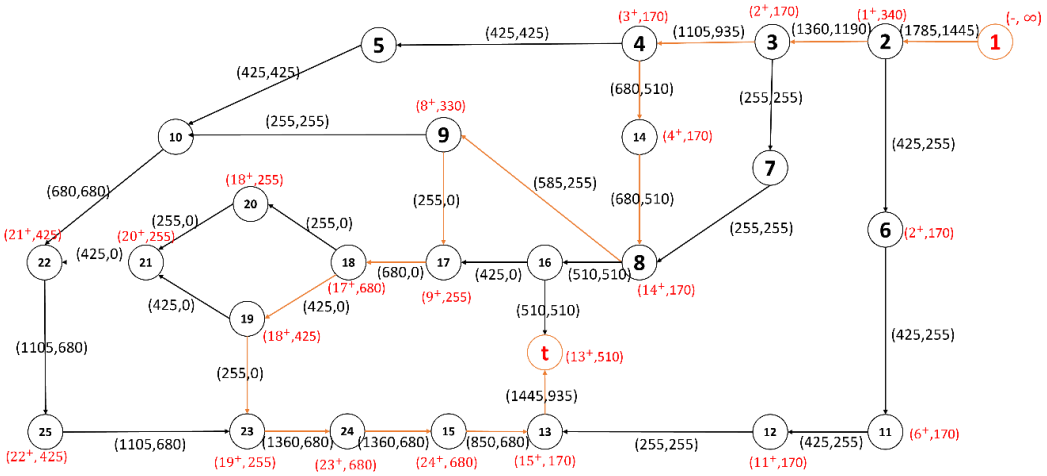
- *Augmenting path* 1-2-3-4-5-10-22-25-23-24-15-13-t dilakukan pembaharuan aliran dan akan dicari nilai maksimum yang dapat mengalir pada tiap sisi di aliran tersebut. Didapat nilai aliran pada iterasi 5 adalah 255 ampere. Pada sisi 9-10 dan 10-22 nilainya telah maksimal dan tidak bisa ditingkatkan pada iterasi selanjutnya.



Gambar L. 2. 12 *Network* yang telah diperbaharui pada iterasi 5

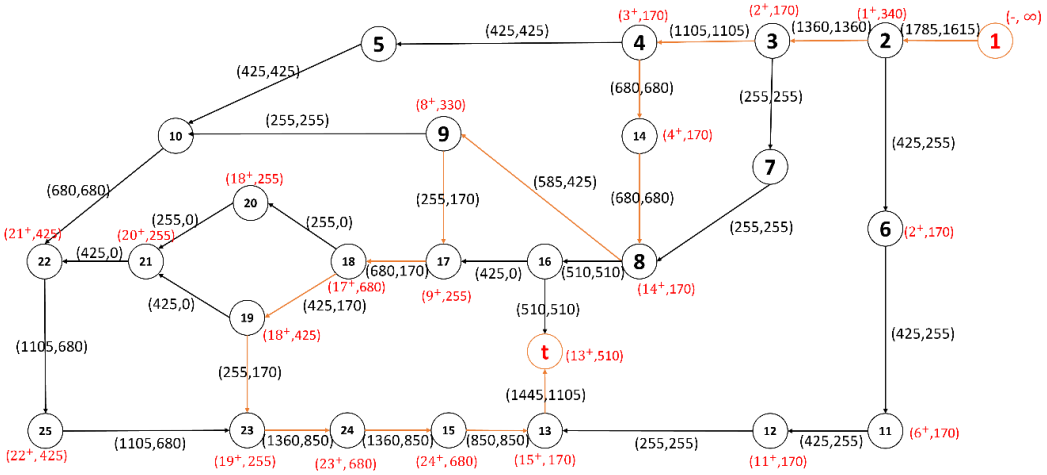
Iterasi 6

- Aliran akan discan dan diberi label pada tiap titiknya. Akan dicari *augmenting path* terpendek dari 1 ke t dan didapat *augmenting path* iterasi 6 adalah 1-2-3-4-14-8-9-17-18-19-23-24-15-13-t.



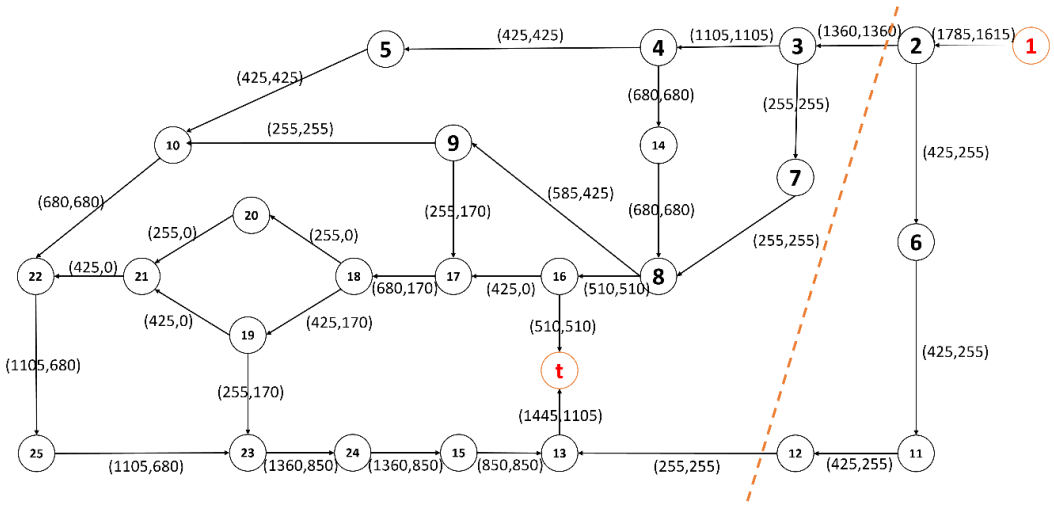
Gambar L. 2. 13 *Scan* dan pelabelan *network* iterasi 6

- *Augmenting path* 1-2-3-4-5-10-22-25-23-24-15-13-t dilakukan pembaharuan aliran dan akan dicari nilai maksimum yang dapat mengalir pada tiap sisi di aliran tersebut. Didapat nilai aliran pada iterasi 6 adalah 170 ampere. Pada sisi 2-3, 3-4, 4-14, dan 14-8 nilainya telah maksimal dan tidak bisa ditingkatkan pada iterasi selanjutnya.



Gambar L. 2. 14 *Network* yang telah diperbaharui pada iterasi 6

Iterasi selesai maka ditentukan *minimum cut* untuk mendapatkan hasil aliran maksimum seperti di bawah ini



Gambar L. 2. 15 Hasil *minimum cut*

Berdasarkan *minimum cut* didapat nilai aliran maksimum dari jaringan listrik area Makassar Selatan dengan tujuan Mall Panakkukang bernilai 1615 ampere.

Lampiran 3 Algoritma Ford Fulkerson pada aplikasi Matlab tujuan Mall Panakkukang

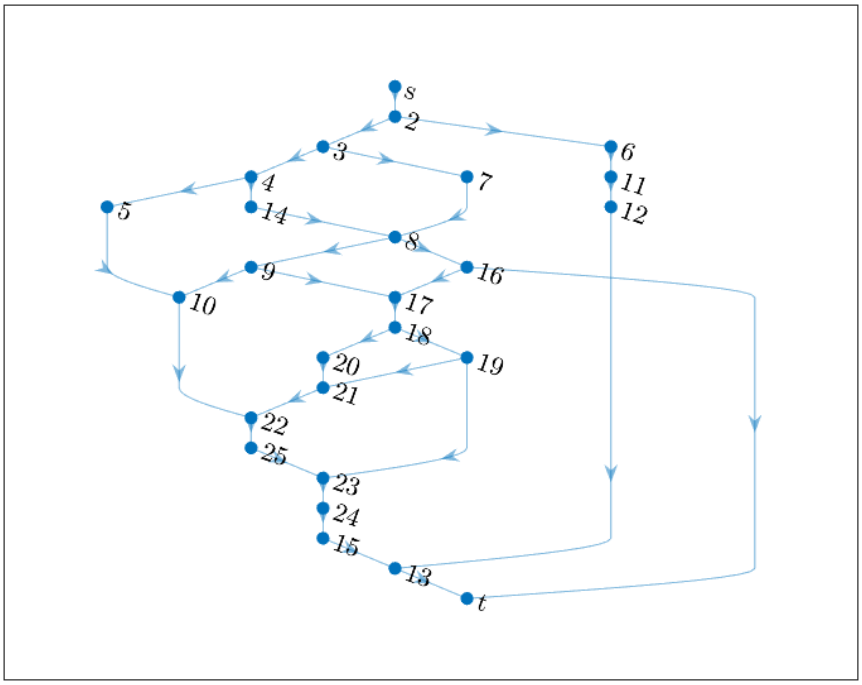
```
function ford_mp
%% Input Network as Adjacency Matrix
    clc; clear; close all;

    atas = [1 2 6 11 12 13 2 3 7 8 16 3 4 14 4 5 8 9 9 10 16 17 18 18
20 19 21 19 22 25 23 24 15];
    bawah = [2 6 11 12 13 26 3 7 8 16 26 4 14 8 5 10 9 10 17 22 17 18
20 19 21 21 22 23 25 23 24 15 13];
    weight = [1785 425 425 425 255 1445 1360 255 255 510 510 1105 680
680 425 425 585 255 255 680 425 680 255 425 255 425 425 255 1105 1105
1360 1360 850];
    tes = digraph(atas,bawah,weight);
    cap = full(adjacency(tes,'weight'));
    s = 1; t = max(bawah); f = 0;
    %disp("Matriks ketetanggan dari network yaitu");
    %disp(cap);
    [kiri kanan] = findedge(tes);
    capacity = tes.Edges.Weight;
    felow = [];
    upper = [];
    for i=1:length(capacity)
        felow = [felow;0];
    end
    for i=1:length(cap)
        upper = [upper;i];
    end
    A = string(capacity);
    B = string(felow);
    str = '('+A+','+B+')';
    tes.Edges.str = str;
    ss = string('s');
    st = string('t');
    titik = [ss 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 st];
```

```
    figure
    z =
plot(tes,'Interpreter','Latex','EdgeLabel',tes.Edges.str,'NodeLabel',t
itik,'edgefontsize',7,'nodefontsize',7);

    len = length(cap);

%% Ford-Fulkerson Algorithm
    while true
        p = findPath(cap);
        Z = fliplr(p);
        Jalurnya = Z;
        Jalurnya(1)=[];
        Jalurnya(end)=[];
        Jalurnya;
        if p(1) == 0, break;end
        flow = max(max(cap));
        for j = 2:length(p)
            flow = min(flow,cap(p(j),p(j-1)));
        end
        for j = 2:length(p)
            a = p(j); b = p(j-1);
            cap(a,b) = cap(a,b) - flow;
            cap(b,a) = cap(b,a) + flow;
        end
        f = f + flow;
        for i=1:length(Z)-1
            G = [Z(i) Z(i+1)];
            for j = 1:length(kiri)
                if [kiri(j) kanan(j)] == G
                    tes.Edges.Weight(j)=tes.Edges.Weight(j)-flow;
                    felow(j) = felow(j)+flow;
                end
            end
        end
        B = string(felow);
        str = '('+A+','+B+')';
```

```
        tes.Edges.str = str;
        text = sprintf('% .0f',Z);
        disp(['Jumlah flow pada augmented path ' num2str(flow)]);
        disp(['Augmented path nya yaitu s  ' num2str(Jalurnya) '
t']);
        %disp("Dengan matriks ketetanggaannya yaitu");
        %disp(full(adjacency(tes,'weighted')));
        disp(' ');
        figure
        z =
plot(tes,'Interpreter','Latex','EdgeLabel',tes.Edges.str,'NodeLabel',t
itik,'edgefontsize',7,'nodefontsize',7);
        highlight(z,Z,'NodeColor','r','EdgeColor','r')
    end
    disp(['Diperoleh max flow nya sebesar ' num2str(f)]);
    %disp("Dengan matriks ketetanggaannya yaitu");
    %disp(full(adjacency(tes,'weighted')))
    figure
    z =
plot(tes,'Interpreter','Latex','EdgeLabel',tes.Edges.str,'NodeLabel',t
itik,'edgefontsize',7,'nodefontsize',7);
%% Find an Augmenting Path
    function F = findPath(A)            % BFS (Breadth-first Search)
        q = zeros(1,len);               % queue
        pred = zeros(1,len);            % predecessor array
        front = 1; back = 2;
        pred(s) = s; q(front) = s;
        while front ~= back
            v = q(front);
            front = front + 1;
            for i = 1:len
                if pred(i) == 0 && A(v,i) > 0
                    q(back) = i;
                    back = back + 1;
                    pred(i) = v;
                end
            end
```

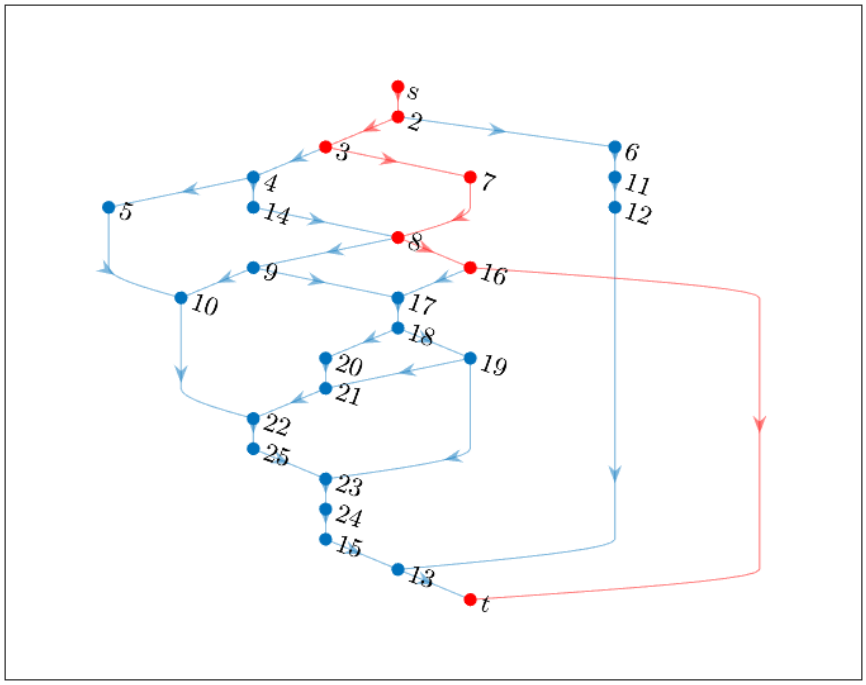
```
        end
        path = zeros(1,len);
        if pred(t) ~= 0
            i = t; c = 1;
            while pred(i) ~= i
                path(c) = i;
                c = c + 1;
                i = pred(i);
            end
            path(c) = s;
            path(c+1:len) = [];
        end
        F = path;
    end
end
```

- Aliran awal tiap sisi bernilai 0 ampere dan belum ada peningkatan aliran.



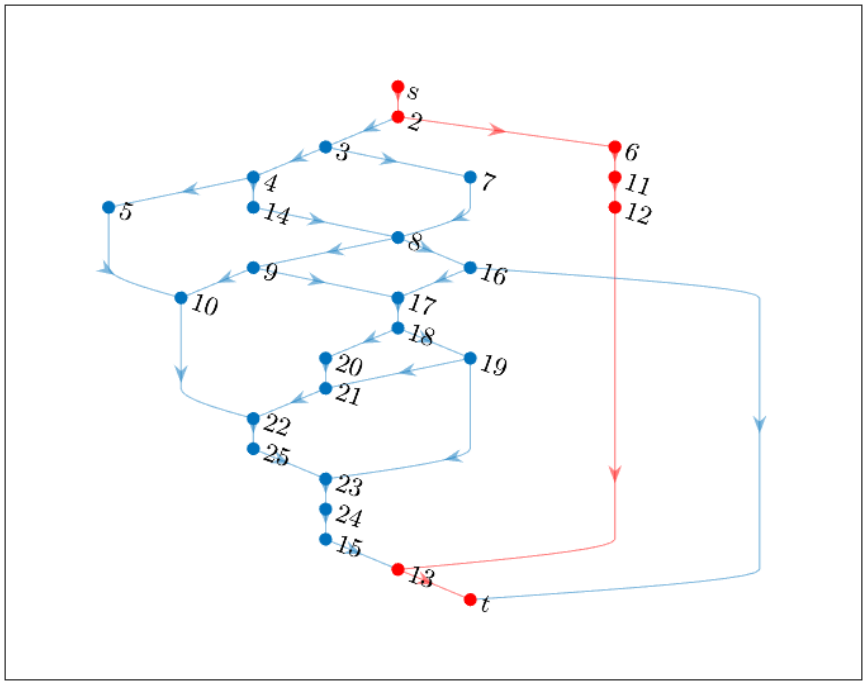
Gambar L. 3. 1 Aliran awal algoritma Ford Fulkerson tujuan Mall Panakkukang

- *Augmenting path* terpendek dari s ke t pada iterasi 1 adalah s-2-3-7-8-16-t.



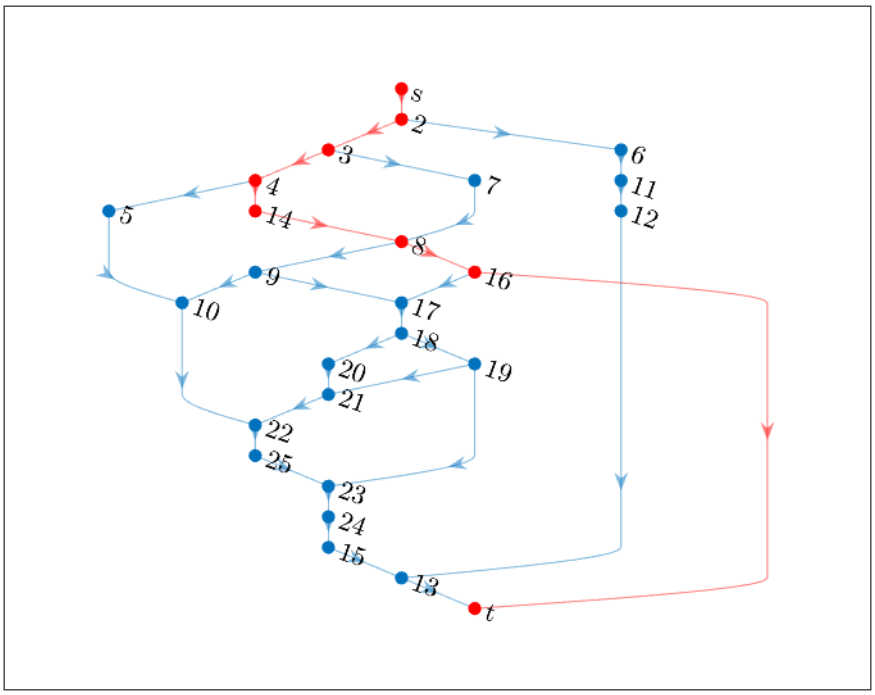
Gambar L. 3. 2 Iterasi 1 algoritma Ford Fulkerson tujuan Mall Panakkukang

- *Augmenting path* terpendek dari s ke t pada iterasi 2 adalah s-2-6-11-12-13-t.



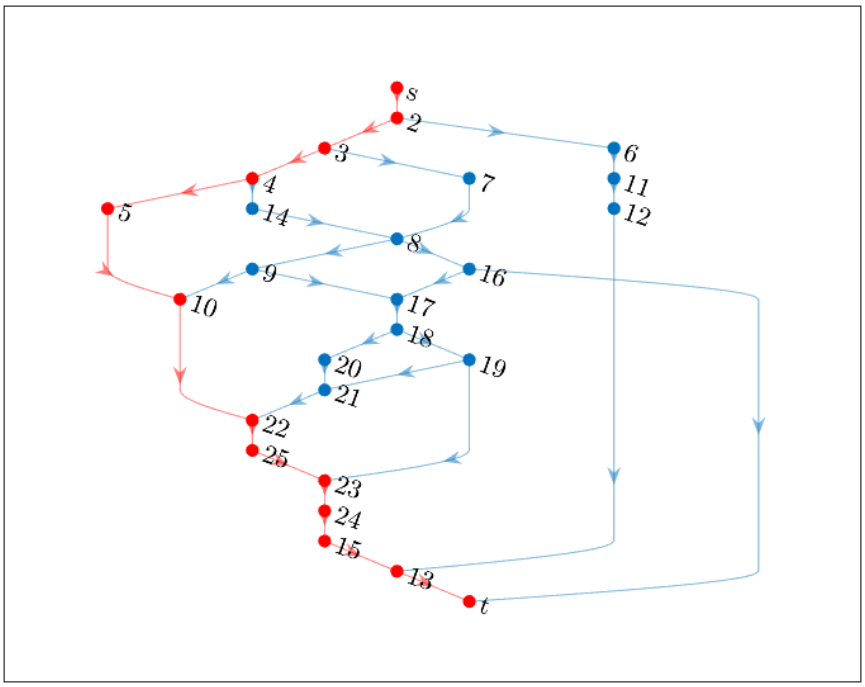
Gambar L. 3. 3 Iterasi 2 algoritma Ford Fulkerson tujuan Mall Panakkukang

- *Augmenting path* terpendek dari s ke t pada iterasi 3 adalah s-2-3-4-14-8-16-t.



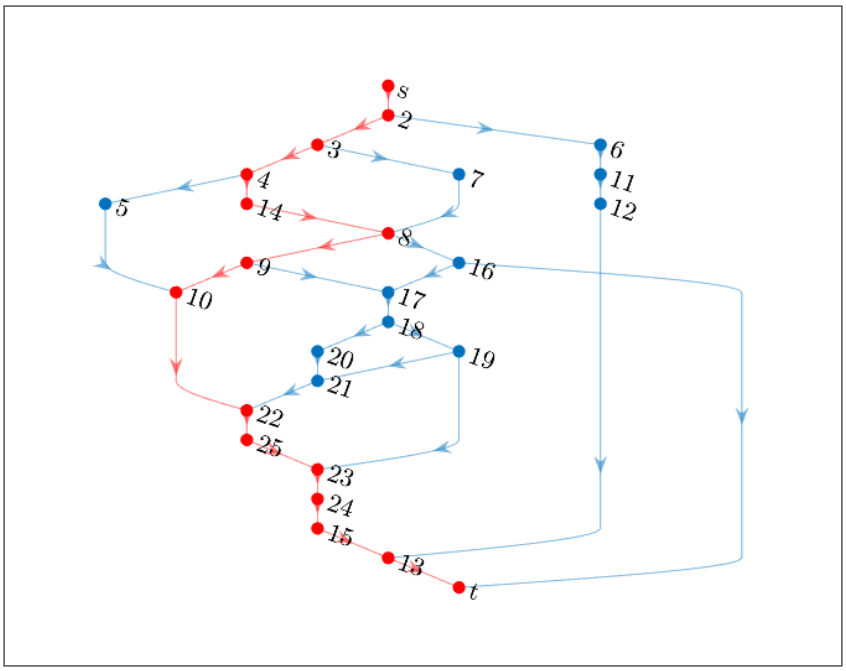
Gambar L. 3. 4 Iterasi 3 algoritma Ford Fulkerson tujuan Mall Panakkukang

- *Augmenting path* terpendek dari s ke t pada iterasi 4 adalah s-2-3-4-5-10-22-25-23-24-15-13-t.



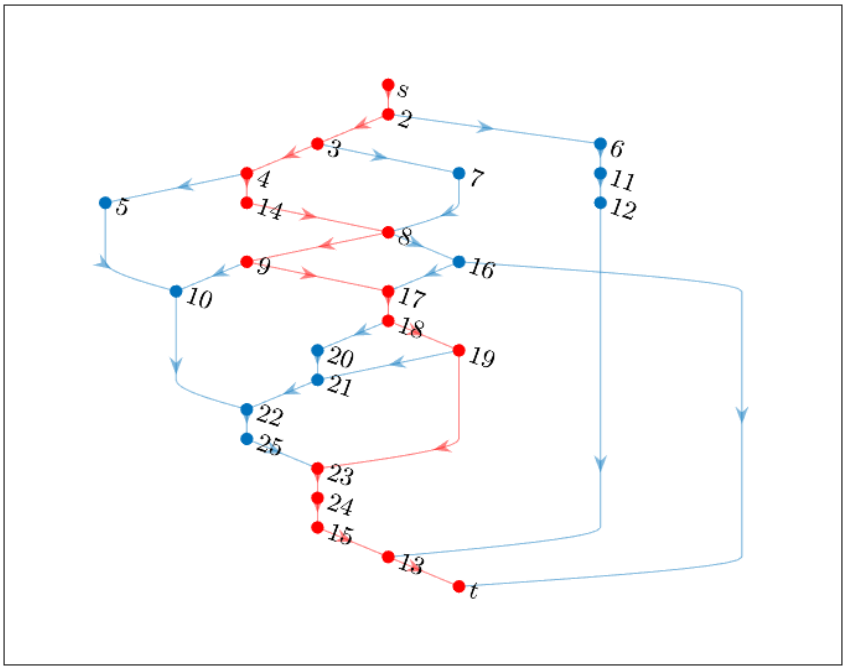
Gambar L. 3. 5 Iterasi 4 algoritma Ford Fulkerson tujuan Mall Panakkukang

- *Augmenting path* terpendek dari s ke t pada iterasi 5 adalah s-2-3-4-14-8-9-10-22-25-23-24-15-13-t.



Gambar L. 3. 6 Iterasi 5 algoritma Ford Fulkerson tujuan Mall Panakkukang

- *Augmenting path* terpendek dari s ke t pada iterasi 6 adalah s-2-3-4-14-8-9-17-18-19-23-24-15-13-t.



Gambar L. 3. 7 Iterasi 6 algoritma Ford Fulkerson tujuan Mall Panakkukang

Lampiran 4 Algoritma Ford Fulkerson pada aplikasi Matlab tujuan Menara Pinisi UNM

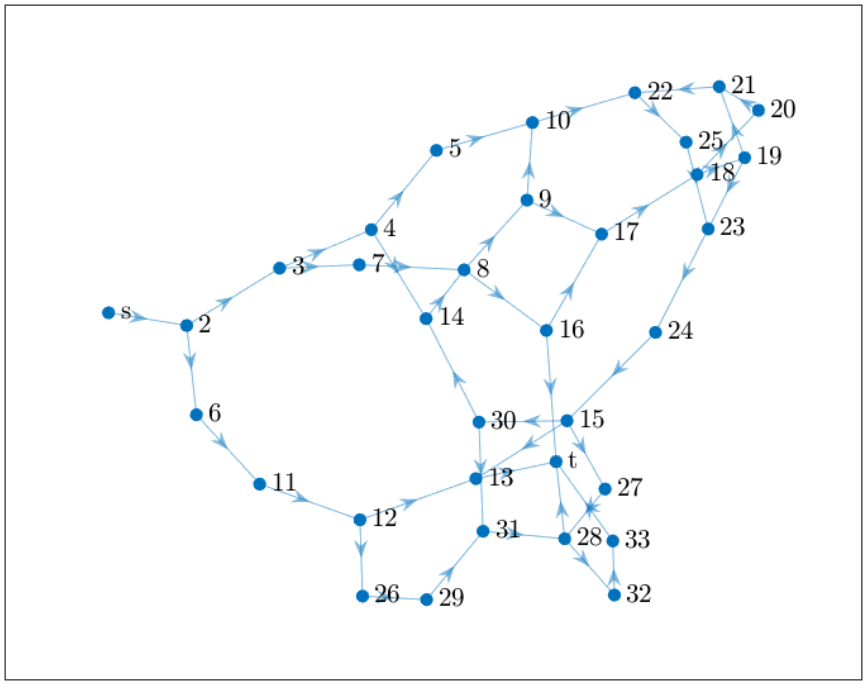
```
function ford_unm
%% Input Network as Adjacency Matrix
    clc; clear; close all;

    atas = [1 2 6 11 12 13 2 3 7 8 16 3 4 14 4 5 8 9 9 10 16 17 18 18
20 19 21 19 22 25 23 24 15 12 26 29 31 28 15 30 30 15 27 28 32 33];
    bawah = [2 6 11 12 13 34 3 7 8 16 34 4 14 8 5 10 9 10 17 22 17 18
20 19 21 21 22 23 25 23 24 15 13 26 29 31 28 34 30 14 31 27 28 32 33
34];
    weight = [1785 425 425 425 255 1445 1360 255 255 510 510 1105 680
680 425 425 585 255 255 680 425 680 255 425 255 425 425 255 1105 1105
1360 1360 850 425 850 850 1105 1360 680 425 255 680 680 425 425 425];
    tes = digraph(atas,bawah,weight);
    cap = full(adjacency(tes,'weight'));
    s = 1; t = max(bawah); f = 0;
    %disp("Matriks ketetanggan dari network yaitu");
    %disp(cap);
    [kiri kanan] = findedge(tes);
    capacity = tes.Edges.Weight;
    felow = [];
    upper = [];
    for i=1:length(capacity)
        felow = [felow;0];
    end
    for i=1:length(cap)
        upper = [upper;i];
    end
    A = string(capacity);
    B = string(felow);
    str = '('+A+','+B+')';
    tes.Edges.str = str;
    ss = string('s');
    st = string('t');
    titik = [ss 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
```

```
23 24 25 26 27 28 29 30 31 32 33 st];
    figure
    z =
plot(tes,'Interpreter','Latex','EdgeLabel',tes.Edges.str,'NodeLabel',t
itik,'edgefontsize',7,'nodefontsize',7);

    len = length(cap);

%% Ford-Fulkerson Algorithm
    while true
        p = findPath(cap);
        Z = fliplr(p);
        Jalurnya = Z;
        Jalurnya(1)=[];
        Jalurnya(end)=[];
        Jalurnya;
        if p(1) == 0, break;end
        flow = max(max(cap));
        for j = 2:length(p)
            flow = min(flow,cap(p(j),p(j-1)));
        end
        for j = 2:length(p)
            a = p(j); b = p(j-1);
            cap(a,b) = cap(a,b) - flow;
            cap(b,a) = cap(b,a) + flow;
        end
        f = f + flow;
        for i=1:length(Z)-1
            G = [Z(i) Z(i+1)];
            for j = 1:length(kiri)
                if [kiri(j) kanan(j)] == G
                    tes.Edges.Weight(j)=tes.Edges.Weight(j)-flow;
                    felow(j) = felow(j)+flow;
                end
            end
        end
        B = string(felow);
```

```
        str = '('+A+','+B+')';
        tes.Edges.str = str;
        text = sprintf('% .0f',Z);
        disp(['Jumlah flow pada augmented path ' num2str(flow)]);
        disp(['Augmented path nya yaitu s  ' num2str(Jalurnya) '
t']);
        %disp("Dengan matriks ketetanggaannya yaitu");
        %disp(full(adjacency(tes,'weighted')));
        disp(' ');
        figure
        z =
plot(tes,'Interpreter','Latex','EdgeLabel',tes.Edges.str,'NodeLabel',t
itik,'edgefontsize',7,'nodefontsize',7);
        highlight(z,Z,'NodeColor','r','EdgeColor','r')
    end
    disp(['Diperoleh max flow nya sebesar ' num2str(f)]);
    %disp("Dengan matriks ketetanggaannya yaitu");
    %disp(full(adjacency(tes,'weighted')))
    figure
    z =
plot(tes,'Interpreter','Latex','EdgeLabel',tes.Edges.str,'NodeLabel',t
itik,'edgefontsize',7,'nodefontsize',7);
%% Find an Augmenting Path
    function F = findPath(A)            % BFS (Breadth-first Search)
        q = zeros(1,len);               % queue
        pred = zeros(1,len);            % predecessor array
        front = 1; back = 2;
        pred(s) = s; q(front) = s;
        while front ~= back
            v = q(front);
            front = front + 1;
            for i = 1:len
                if pred(i) == 0 && A(v,i) > 0
                    q(back) = i;
                    back = back + 1;
                    pred(i) = v;
                end
```

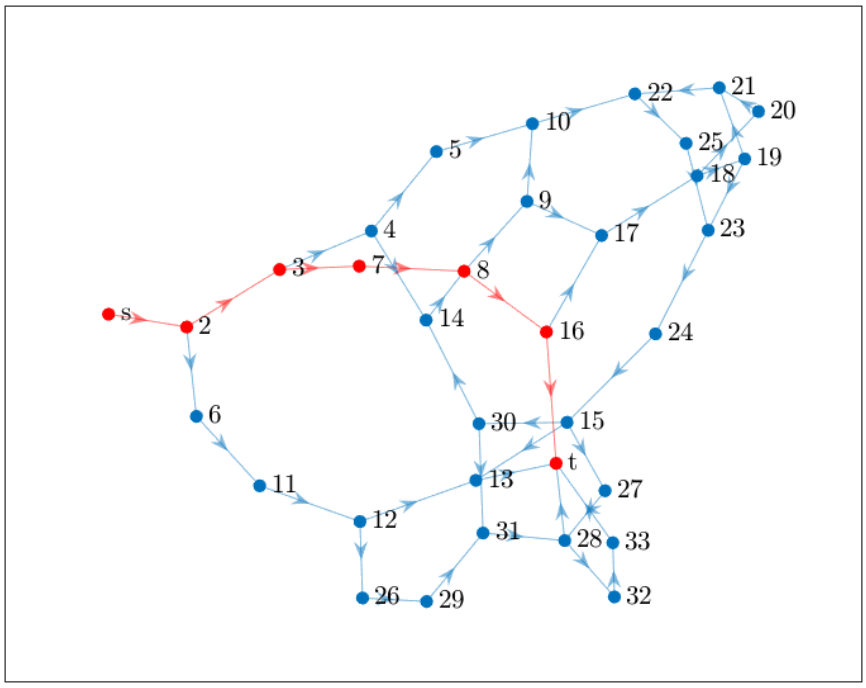
```
            end
        end
        path = zeros(1,len);
        if pred(t) ~= 0
            i = t; c = 1;
            while pred(i) ~= i
                path(c) = i;
                c = c + 1;
                i = pred(i);
            end
            path(c) = s;
            path(c+1:len) = [];
        end
        F = path;
    end
end
```

- Aliran awal tiap sisi bernilai 0 ampere dan belum ada peningkatan aliran.



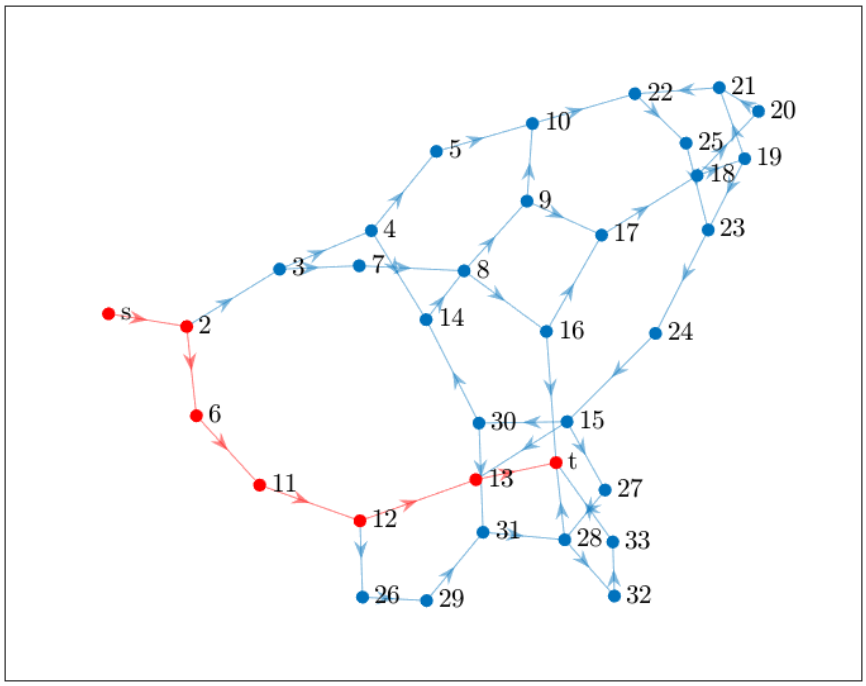
Gambar L. 4. 1 Aliran awal algoritma Ford Fulkerson tujuan Menara Pinisi UNM

- *Augmenting path* terpendek dari s ke t pada iterasi 1 adalah s-2-3-7-8-16-t.



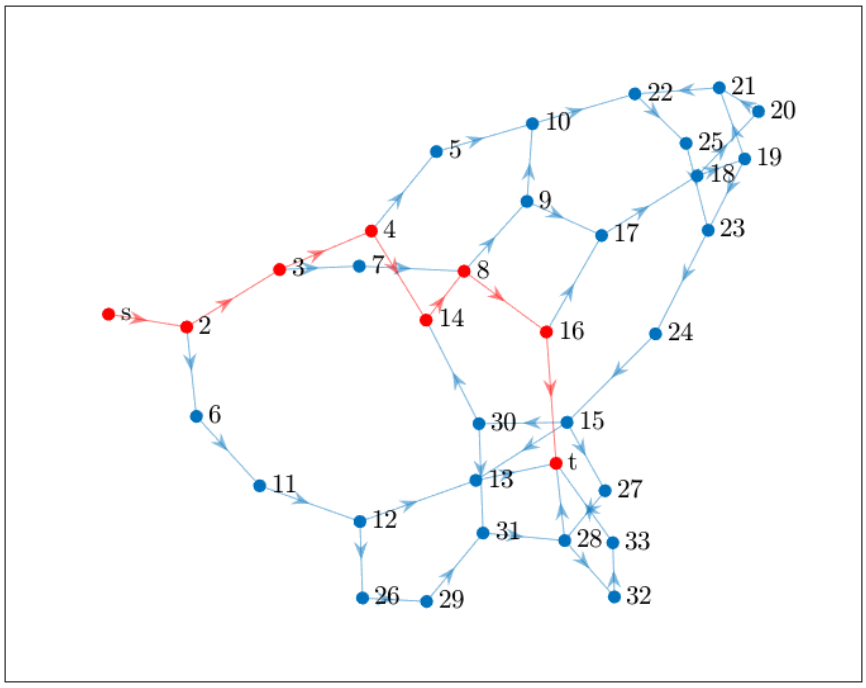
Gambar L. 4. 2 Iterasi 1 Aliran awal algoritma Ford Fulkerson tujuan Menara Pinisi UNM

- *Augmenting path* terpendek dari s ke t pada iterasi 2 adalah s-2-6-11-12-13-t.



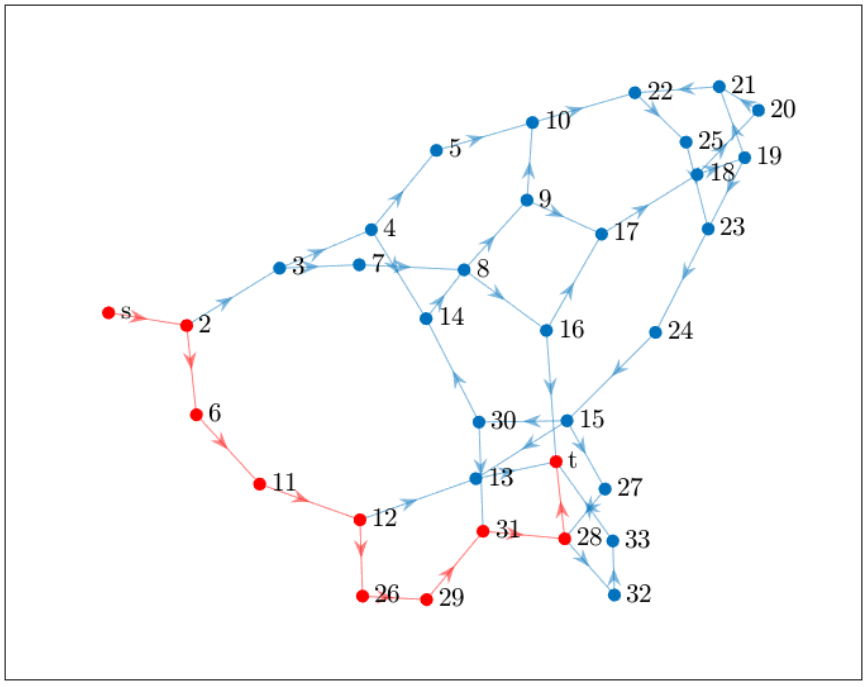
Gambar L. 4. 3 Iterasi 2 algoritma Ford Fulkerson tujuan Menara Pinisi UNM

- *Augmenting path* terpendek dari s ke t pada iterasi 3 adalah s-2-3-4-14-8-16-t.



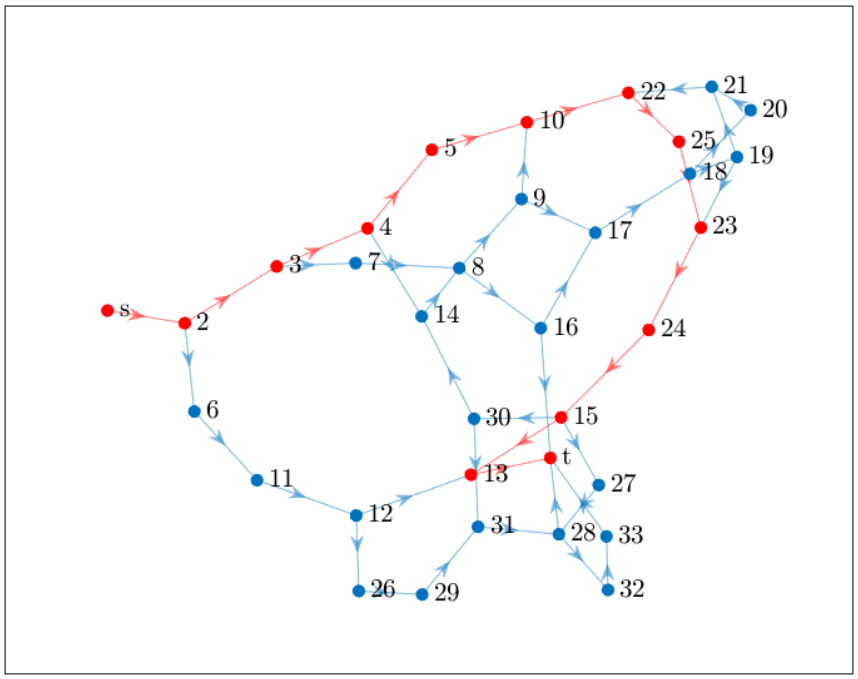
Gambar L. 4. 4 Iterasi 3 algoritma Ford Fulkerson tujuan Menara Pinisi UNM

- *Augmenting path* terpendek dari s ke t pada iterasi 4 adalah s-2-6-11-12-26-29-31-28-t.



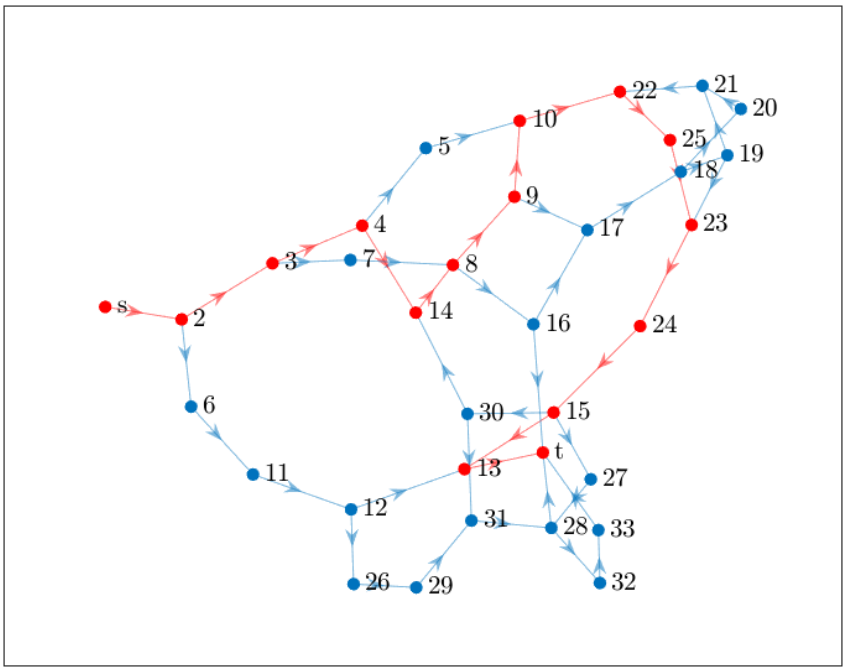
Gambar L. 4. 5 Iterasi 4 algoritma Ford Fulkerson tujuan Menara Pinisi UNM

- *Augmenting path* terpendek dari s ke t pada iterasi 5 adalah s-2-3-4-5-10-22-25-23-24-15-13-t.



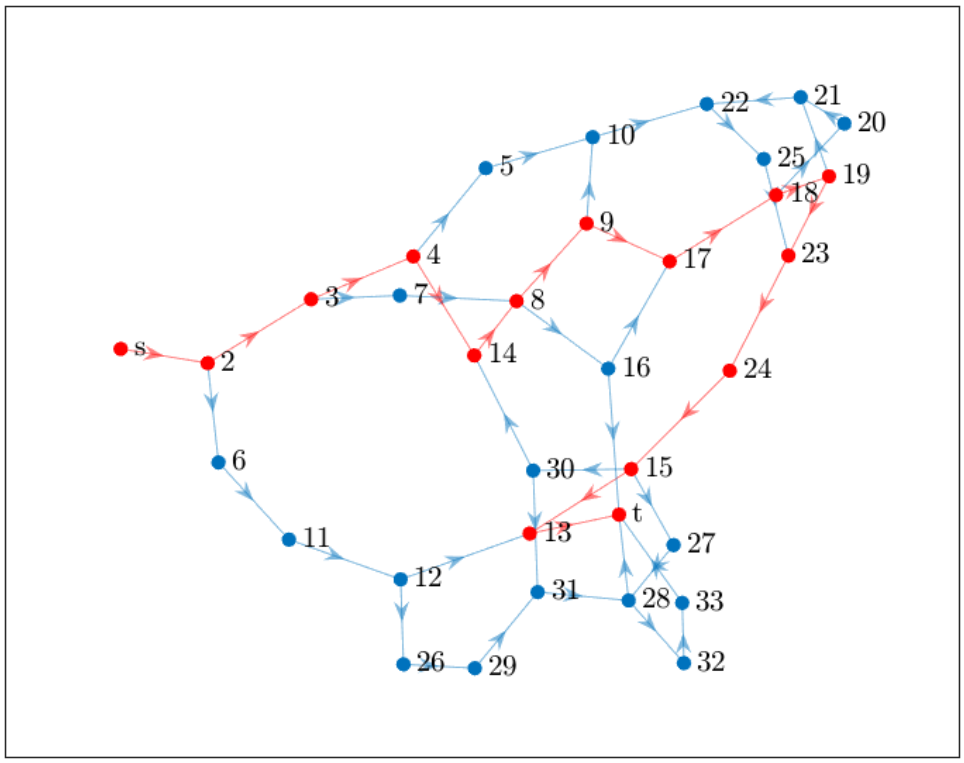
Gambar L. 4. 6 Iterasi 5 algoritma Ford Fulkerson tujuan Menara Pinisi UNM

- *Augmenting path* terpendek dari s ke t pada iterasi 6 adalah s-2-3-4-14-8-9-10-22-25-23-24-15-13-t.



Gambar L. 4. 7 Iterasi 6 algoritma Ford Fulkerson tujuan Menara Pinisi UNM

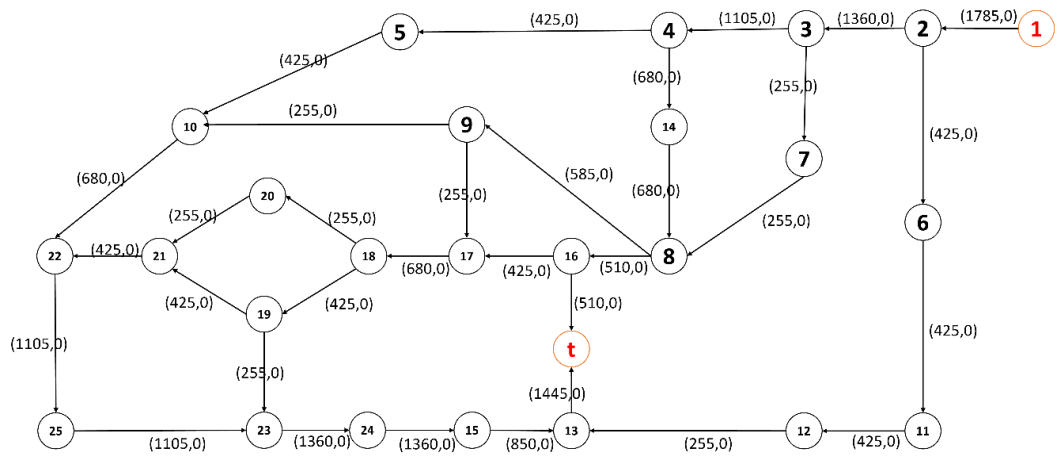
- *Augmenting path* terpendek dari s ke t pada iterasi 7 adalah s-2-3-4-14-8-9-17-18-19-23-24-15-13-t.



Gambar L. 4. 8 Iterasi 7 algoritma Ford Fulkerson tujuan Menara Pinisi UNM

Lampiran 5 Algoritma Dinic dengan cara manual tujuan Mall Panakkukang

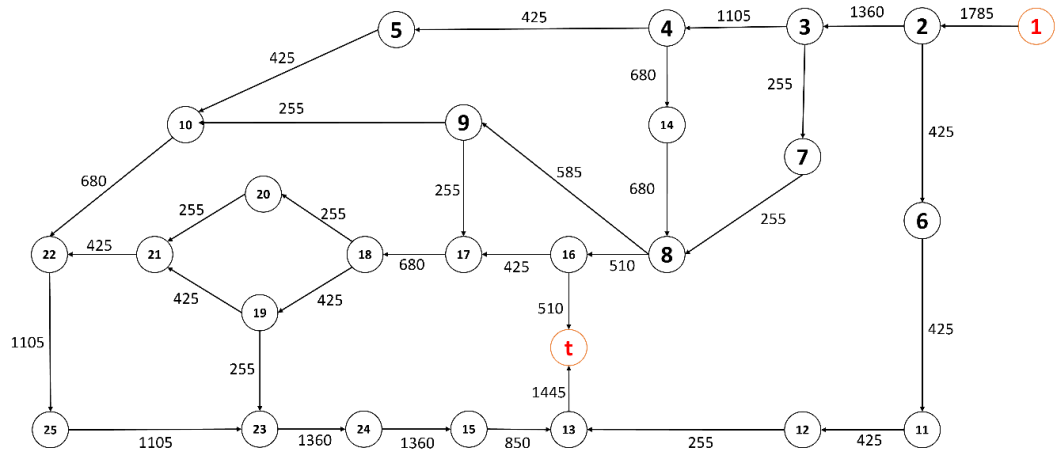
Diberi aliran 0 pada tiap sisi dengan $f(e) = 0$. Aliran awal tiap sisi bernilai 0 ampere.



Gambar L. 5. 1 Aliran awal tujuan Mall Panakkukang

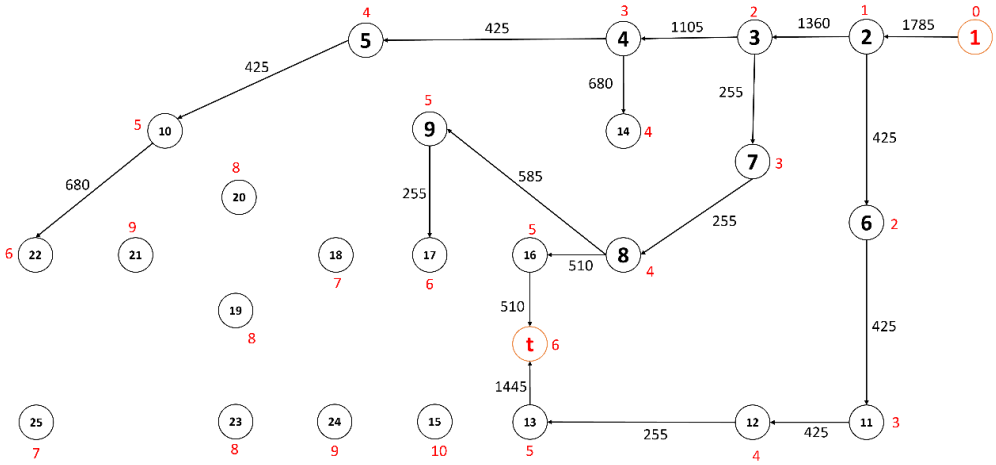
Iterasi 1

- Konstruksi jaringan sisa $N_f = (V, E_f)$. Karena belum ada aliran yang ditingkatkan maka jaringan sisa belum mengalami perubahan.



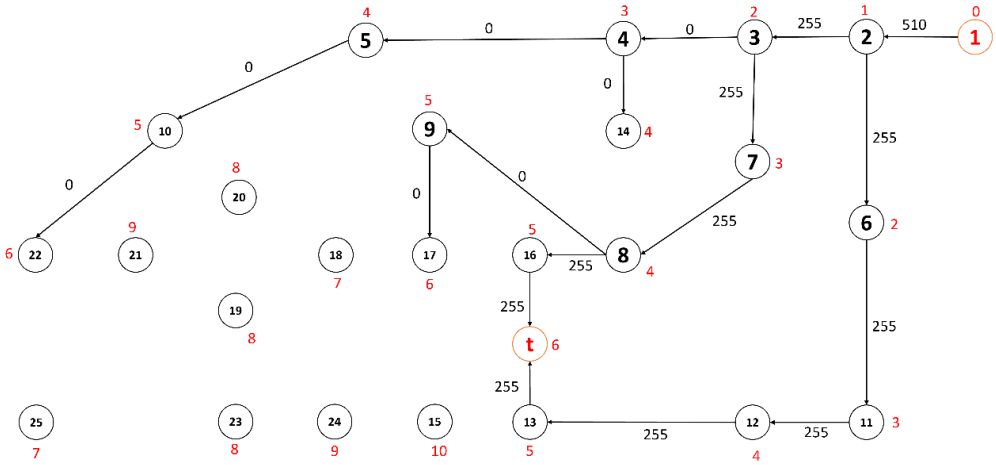
Gambar L. 5. 2 Jaringan sisa iterasi 1

- Konstruksi *layered network* relatif terhadap arus *f*. Didapatkan level atau tingkatan titik terendah adalah 6 maka *augmenting path* iterasi 1 adalah 1-2-6-11-12-13-t dan 1-2-3-7-8-16-t.



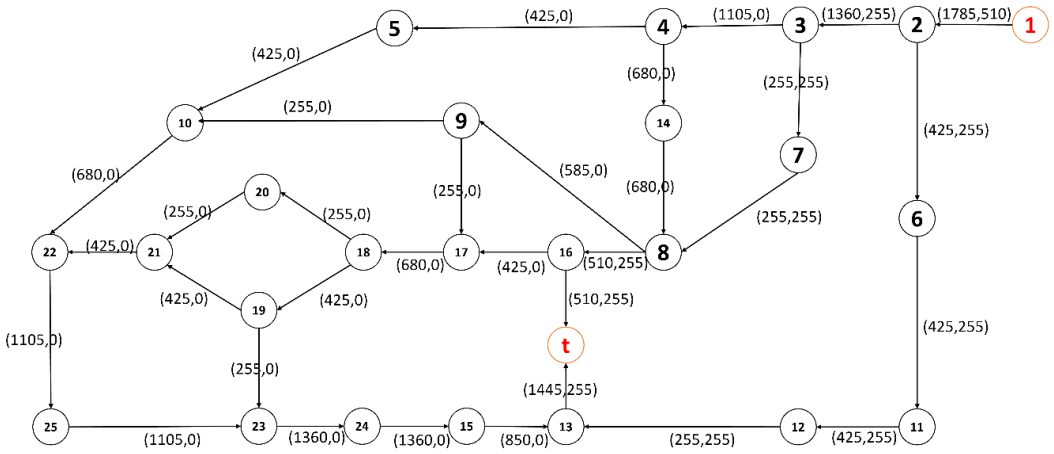
Gambar L. 5. 3 *Layered network* iterasi 1

- Konstruksi *blocking flow g* di *layered network* jalur 1-2-6-11-12-13-t dan 1-2-3-7-8-16-t masing-masing aliran maksimum yang didapat adalah 255 ampere.



Gambar L. 5. 4 *Blocking flow* iterasi 1

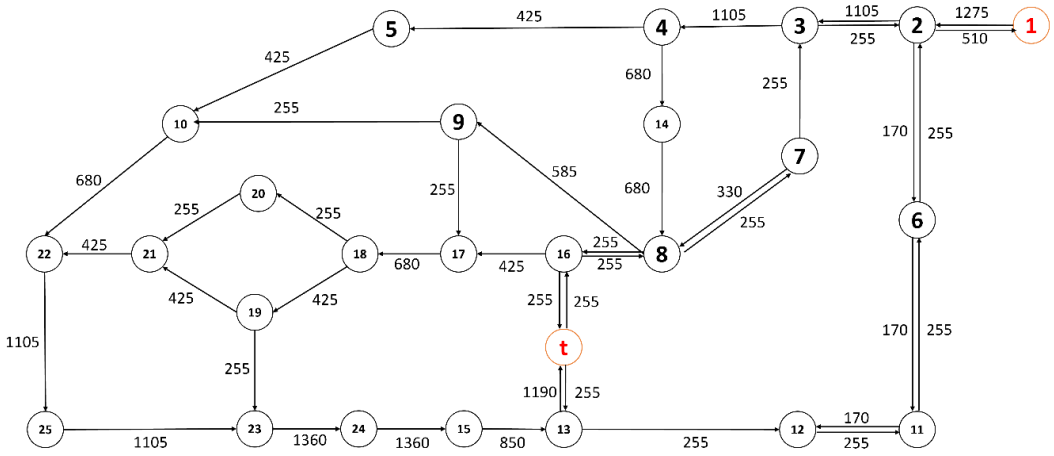
- Dikonstruksi arus baru *f'*. Telah dilakukan peningkatan aliran pada iterasi 1 maka didapat arus baru pada iterasi 1.



Gambar L. 5. 5 Arus baru iterasi 1

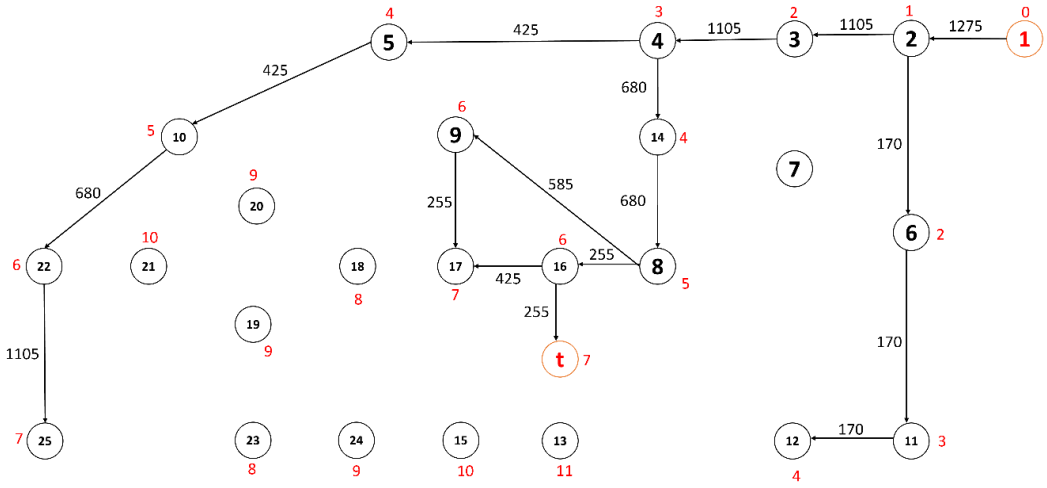
Iterasi 2

- Konstruksi jaringan sisa $N_f = (V, E_f)$. Karena telah dilakukan peningkatan pada iterasi 1 maka sisa aliran yang dapat ditingkatkan pada iterasi 2 seperti Gambar di L. 4. 6.



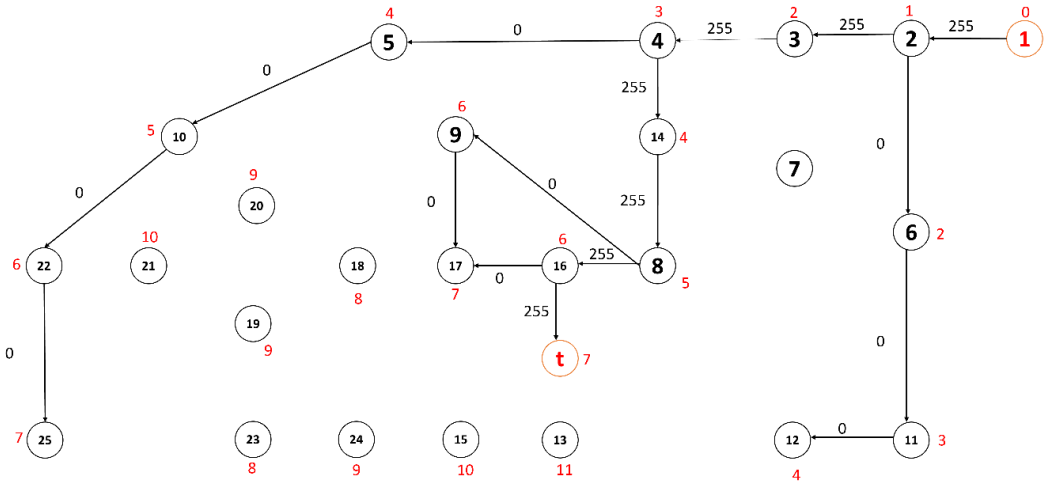
Gambar L. 5. 6 Jaringan sisa iterasi 2

- Konstruksi *layered network* relatif terhadap arus *f*. Didapatkan level atau tingkatan titik terendah adalah 7 maka *augmenting path* iterasi 2 adalah 1-2-3-4-14-8-16-t.



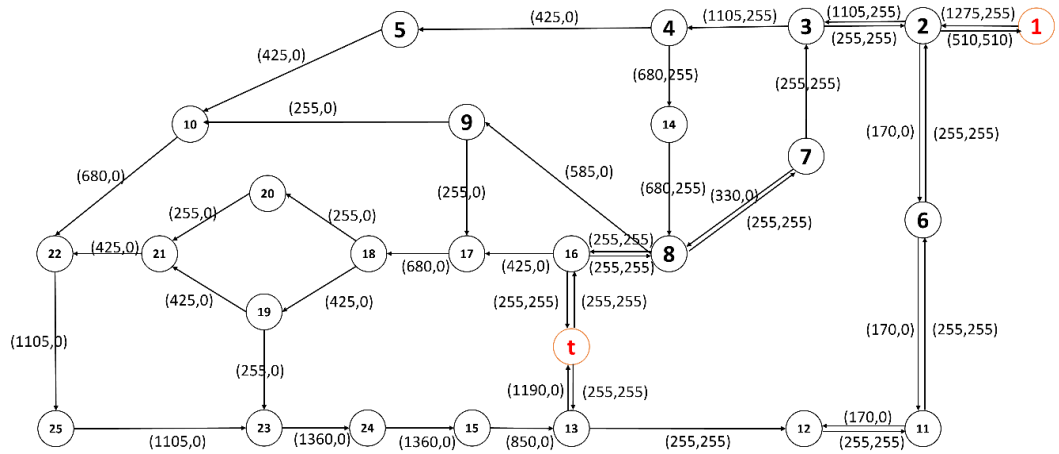
Gambar L. 5. 7 *Layered network* iterasi 2

- Konstruksi *blocking flow g* di *layered network* jalur 1-2-3-4-14-8-16-t dengan aliran maksimum yang didapat adalah 255 ampere.



Gambar L. 5. 8 *Blocking flow* iterasi 2

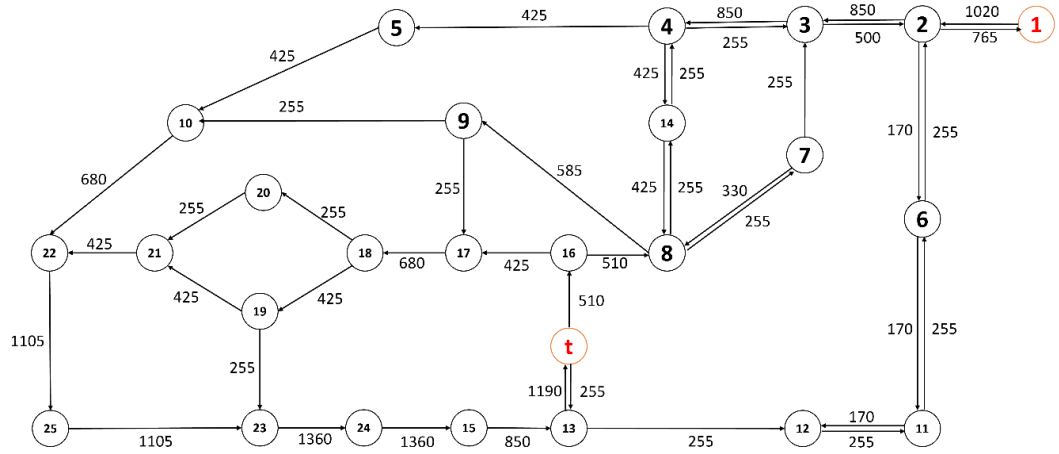
- Dikonstruksi arus baru *f'*. Telah dilakukan peningkatan aliran pada iterasi 2 maka didapat arus baru pada iterasi 2.



Gambar L. 5. 9 Arus baru iterasi 2

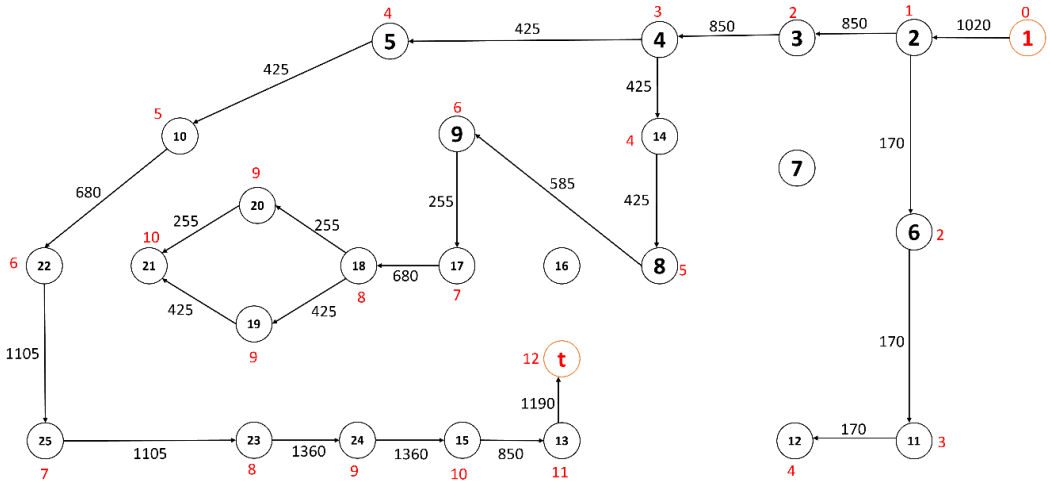
Iterasi 3

- Konstruksi jaringan sisa $N_f = (V, E_f)$. Karena telah dilakukan peningkatan pada iterasi 2 maka sisa aliran yang dapat ditingkatkan pada iterasi 3 seperti Gambar di L. 4. 10.



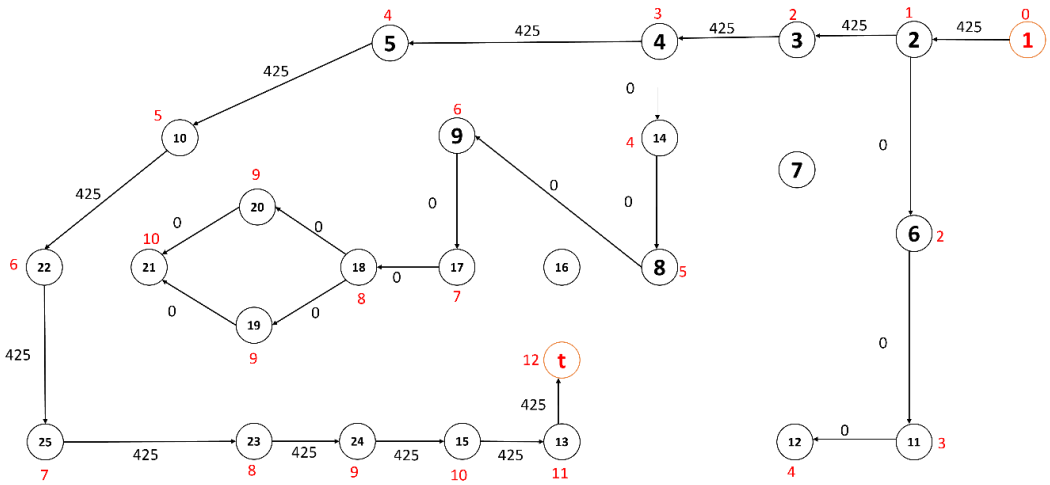
Gambar L. 5. 10 Jaringan sisa iterasi 3

- Konstruksi *layered network* relatif terhadap arus *f*. Didapatkan level atau tingkatan titik terendah adalah 12 maka *augmenting path* iterasi 3 adalah 1-2-3-4-5-10-22-25-23-24-15-13-t.



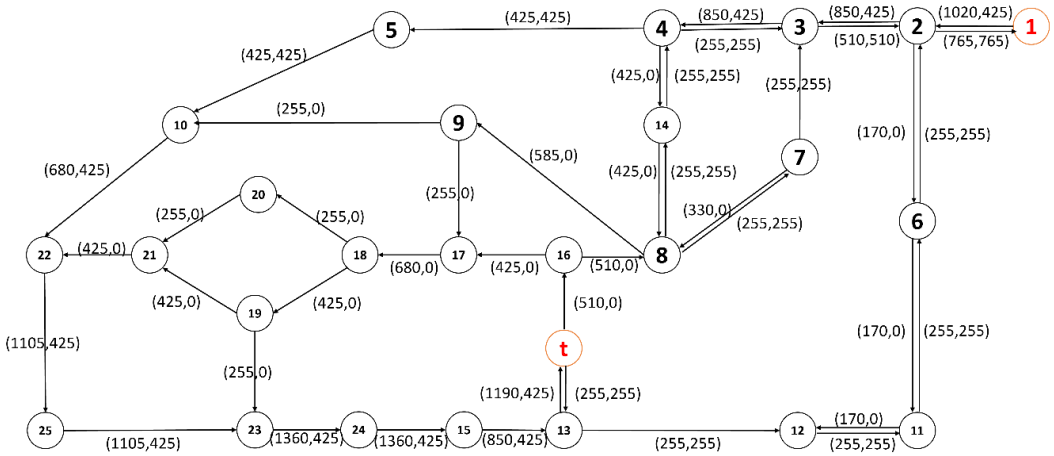
Gambar L. 5. 11 *Layered network* iterasi 3

- Konstruksi *blocking flow g* di *layered network* jalur 1-2-3-4-5-10-22-25-23-24-15-13-t dengan aliran maksimum yang didapat adalah 425 ampere.



Gambar L. 5. 12 *Blocking flow* iterasi 3

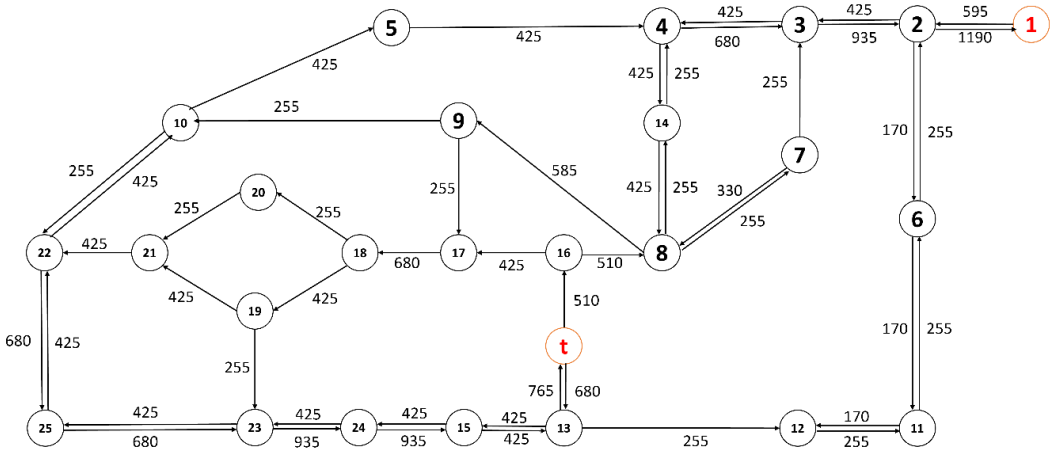
- Dikonstruksi arus baru *f'*. Telah dilakukan peningkatan aliran pada iterasi 3 maka didapat arus baru pada iterasi 3.



Gambar L. 5. 13 Arus baru iterasi 3

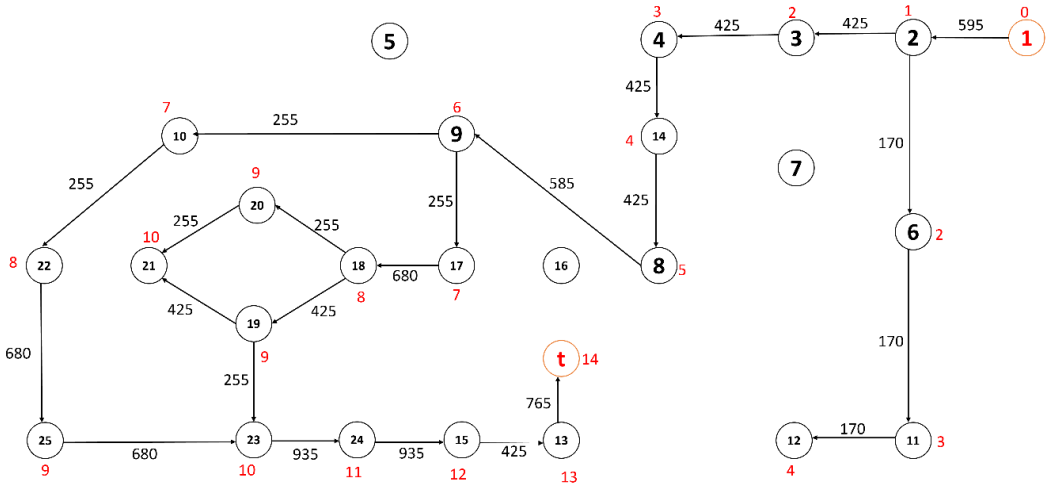
Iterasi 4

- Konstruksi jaringan sisa $N_f = (V, E_f)$. Karena telah dilakukan peningkatan pada iterasi 3 maka sisa aliran yang dapat ditingkatkan pada iterasi 4 seperti Gambar di L. 4. 14.



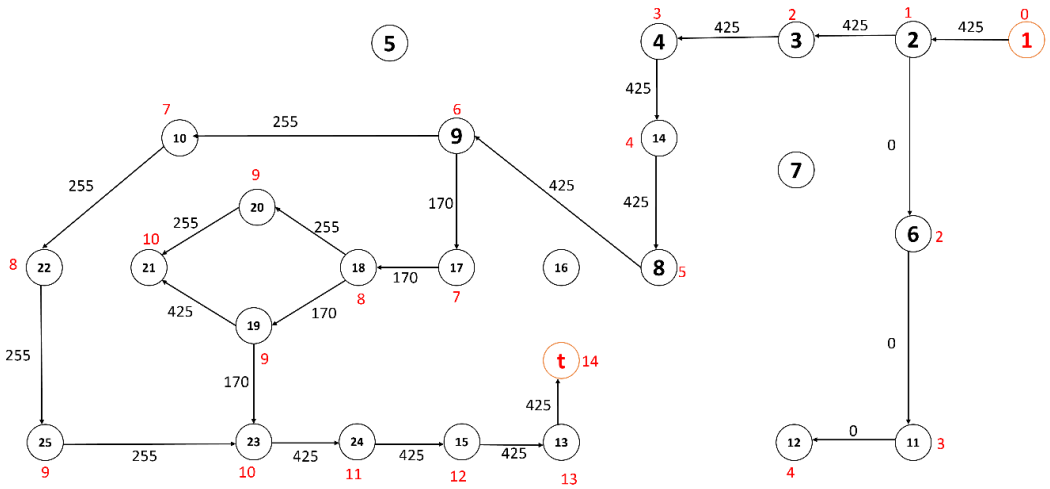
Gambar L. 5. 14 Jaringan sisa iterasi 4

- Konstruksi *layered network* relatif terhadap arus *f*. Didapatkan level atau tingkatan titik terendah adalah 14 maka *augmenting path* iterasi 4 adalah 1-2-3-4-14-8-9-10-22-25-23-24-15-13-t dan 1-2-3-4-14-8-9-17-18-19-23-24-15-13-t



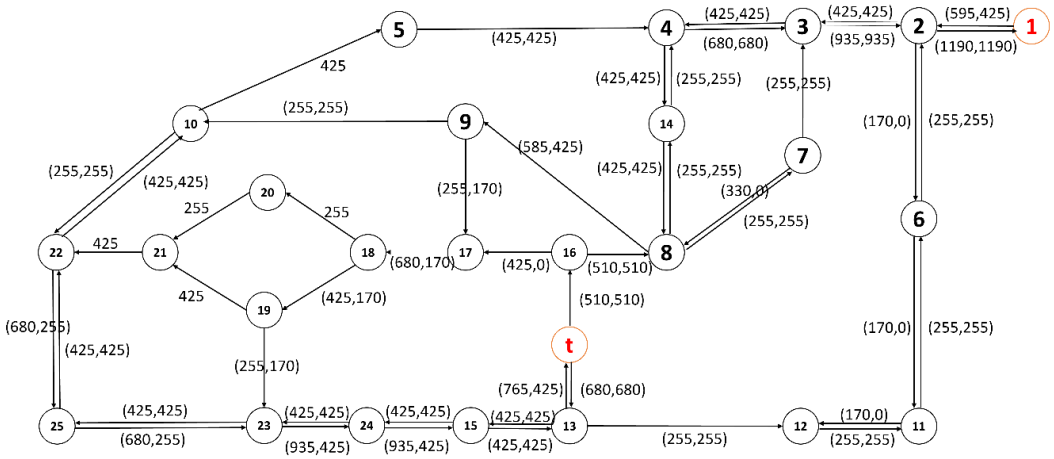
Gambar L. 5. 15 *Layered network* iterasi 4

- Konstruksi *blocking flow g* di *layered network* jalur 1-2-3-4-14-8-9-10-22-25-23-24-15-13-t dengan nilai aliran sebesar 250 ampere dan 1-2-3-4-14-8-9-17-18-19-23-24-15-13-t sebesar 170 ampere.



Gambar L. 5. 16 *Blocking flow* iterasi 4

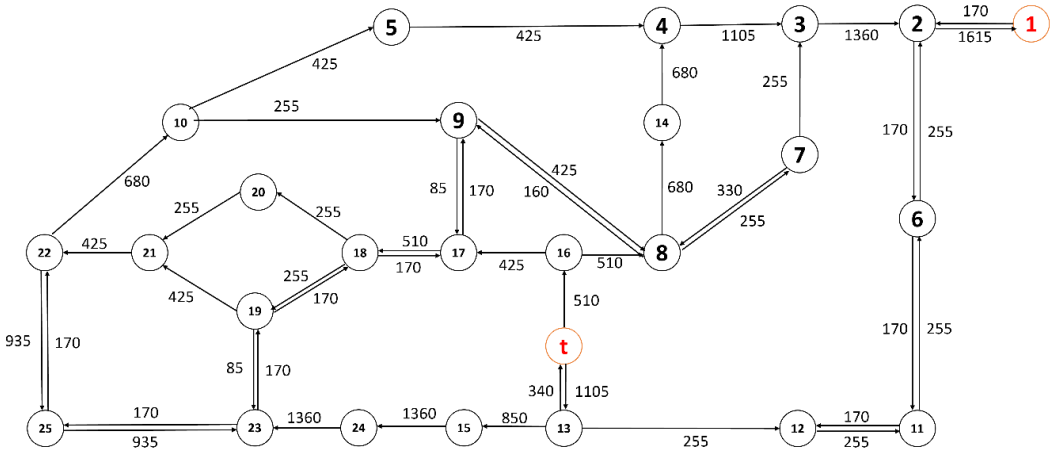
- Dikonstruksi arus baru *f'*. Telah dilakukan peningkatan aliran pada iterasi 4 maka didapat arus baru pada iterasi 4.



Gambar L. 5. 17 Arus baru iterasi 4

Iterasi 5

- Konstruksi jaringan sisa $N_f = (V, E_f)$. Karena telah dilakukan peningkatan pada iterasi 4 maka sisa aliran yang dapat ditingkatkan pada iterasi 5 seperti Gambar di L. 4. 18



Gambar L. 5. 18 Jaringan sisa iterasi 5

Lampiran 6 Algoritma Dinic pada aplikasi Matlab tujuan Mall Panakkukang

```
function dinic_mp
%% Input Network as Adjacency Matrix
    clc; clear; close all;

    atas = [1 2 6 11 12 13 2 3 7 8 16 3 4 14 4 5 8 9 9 10 16 17 18
18 20 19 21 19 22 25 23 24 15];
    bawah = [2 6 11 12 13 26 3 7 8 16 26 4 14 8 5 10 9 10 17 22 17
18 20 19 21 21 22 23 25 23 24 15 13];
    weight = [1785 425 425 425 255 1445 1360 255 255 510 510 1105
680 680 425 425 585 255 255 680 425 680 255 425 255 425 425 255
1105 1105 1360 1360 850];
    tes = digraph(atas,bawah,weight);
    city=full(adjacency(tes,'weight'));
    s = 1; t = max(bawah); f = 0; hasilflow=0;
    [kiri kanan] = findedge(tes);
    capacitytes = tes.Edges.Weight;
    felowtes = [];
    for i=1:length(capacitytes)
        felowtes = [felowtes;0];
    end
    [kirito] = [kiri];
    [kananto] = [kanan];
    selow = felowtes;
    B = string(felowtes);
    ss = string('s');
    st = string('t');
    titik = [ss 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
21 22 23 24 25 st];
    jumlahiter = 0;

%% Dinic Algorithm
while  true
    jumlahiter = jumlahiter+1;
    A = string(capacitytes);
    str = '('+A+','+B+')';
```

```
        tes.Edges.str = str;
        figure
        z =
    plot(tes,'Layout','layered','Interpreter','Latex','EdgeLabel',tes.
    Edges.str,'NodeLabel',titik,'edgefontsize',7,'nodefontsize',7);
        title('Residual Network','color','blue','fontangle','italic');
        [kiri kanan] = findedge(tes);
        felowtes = [];
        for i=1:length(capacitytes)
            felowtes = [felowtes;0];
        end
        sisi = table2cell(tes.Edges(:,1));
        teslama = tes;
        tesbaru = tes;
        layer = [repmat({'0'},length(atas),1)];
        V = [s];
        W = [];
        Iter = 1;
        nomor = 0;
        for i=1:length(atas)
            for j=Iter:length(V)
                T = successors(tes,V(j));
                for k=1:length(V)
                    T(T==V(k))=[];
                end
                for m=1:length(T)
                    nomor=nomor+1;
                    layer(nomor)={[V(j) T(m)]};
                end
                for l=1:length(W)
                        T(T==W(l))=[];
                end
                W=[W;T];
            end
            Iter=Iter+1;
            V=[V;W];
            if V(V==t), break;end
```

```
            W=[];
        end
        if V~=t, break; end
        angka = 0;
        for i=1:length(sisi)
                for j=1:length(layer)
                    if isequal(layer{j},sisi{i})
                        teslama=rmedge(teslama,i-angka);
                        angka = angka+1;
                        continue
                    end
                end
        end
        teslama = table2cell(teslama.Edges(:,1));
        angka = 0;
        for i=1:length(sisi)
                for j=1:length(teslama)
                    if isequal(teslama{j},sisi{i})
                        tesbaru=rmedge(tesbaru,i-angka);
                        angka = angka+1;
                        continue
                    end
                end
        end
        [left right] = findedge(tesbaru);
        capacitytesbaru = tesbaru.Edges.Weight;
        felowtesbaru = [];
        for i=1:length(capacitytesbaru)
            felowtesbaru = [felowtesbaru;0];
        end
        cap = full(adjacency(tesbaru,'weight'));
        len = length(cap);
        figure
        z =
    plot(tesbaru,'Layout','layered','Interpreter','Latex','EdgeLabel',
    tesbaru.Edges.Weight,'NodeLabel',titik,'edgefontsize',7,'nodefonts
    ize',7);
```

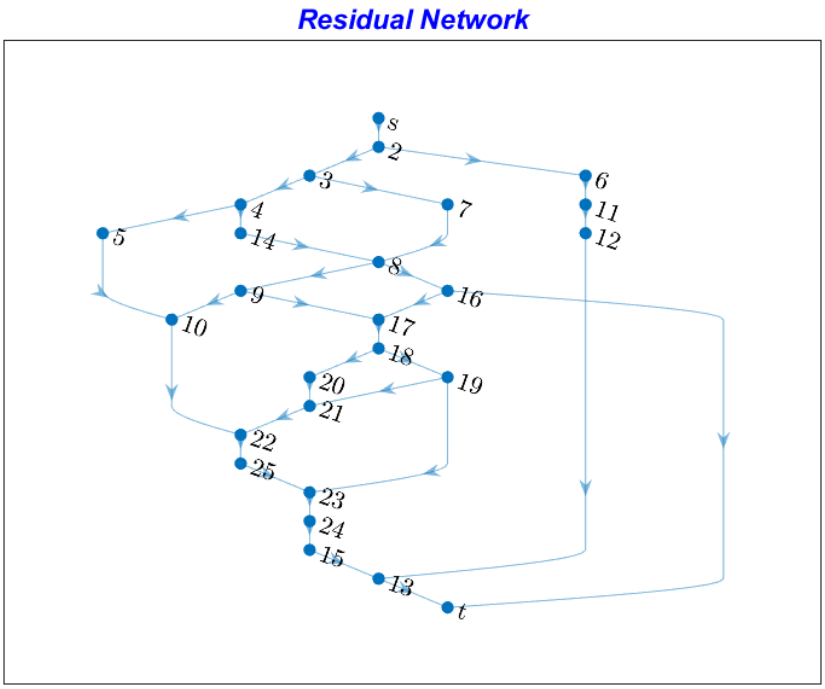
```
        title('Layeres Network','color','blue','fontangle','italic');
        while true
            p = findPath(cap);
            Z = fliplr(p);
            if p(1) == 0, break;end
            flow = max(max(cap));
            for j = 2:length(p)
                flow = min(flow,cap(p(j),p(j-1)));
            end
            for j = 2:length(p)
                a = p(j); b = p(j-1);
                cap(a,b) = cap(a,b) - flow;
                city(a,b) = city(a,b) - flow;
                cap(b,a) = cap(b,a) + flow;
                city(b,a) = city(b,a)+flow;
            end
            f = f + flow;
            hasilflow = hasilflow+flow;
            for i=1:length(Z)-1
                G = [Z(i) Z(i+1)];
                for j = 1:length(left)
                    if [left(j) right(j)] == G
                        felowtesbaru(j) = felowtesbaru(j)+flow;
                    end
                end
            end
            for i=1:length(Z)-1
                G = [Z(i) Z(i+1)];
                for j = 1:length(kiri)
                    if [kiri(j) kanan(j)] == G
                        felowtes(j) = felowtes(j)+flow;
                    end
                end
            end
        end
        tes = digraph(city);
        [kiribaru kananbaru] = findedge(tes);
```

```
        capacitytes = tes.Edges.Weight;
        felow = [];
        for i=1:length(capacitytes)
            felow = [felow;0];
        end
        for i=1:length(kirito)
            for j=1:length(kiribaru)
                if [kirito(i) kananto(i)]==[kiribaru(j) kananbaru(j)]
                    felow(j)=selow(i);
                end
            end
        end
        selow=felow;
        [kirito] = [kiribaru];
        [kananto] = [kananbaru];
        felow = [];
        for i=1:length(capacitytes)
            felow = [felow;0];
        end
        for i=1:length(kiri)
            for j=1:length(kirito)
                if [kiri(i) kanan(i)] == [kananto(j) kirito(j)]
                    selow(j) = selow(j) + felowtes(i);
                end
            end
        end
        for i=1:length(kirito)
            for j=1:length(kiribaru)
                if [kirito(i) kananto(i)] == [kiribaru(j)
    kananbaru(j)]
                    felow(j) = selow(i);
                end
            end
        end
        disp(' ');
        B = string(felow);
        capacitytes = tes.Edges.Weight;
```

```
    fprintf('Diperoleh jumlah flow pada iterasi ke-%g  sebesar
%g\n',jumlahiter,hasilflow);
    disp(['dengan total flow nya sebesar ' num2str(f)]);
    figure
    z =
plot(tesbaru,'Layout','layered','Interpreter','Latex','EdgeLabel',
felowtesbaru,'NodeLabel',titik,'edgefontsize',7,'nodefontsize',7);
    title('Blocking Flow','color','blue','fontangle','italic');
    hasilflow=0;
end
disp(' ');
disp('Karena sudah tidak terdapat lagi augmenting path maka
iterasi terhenti');
disp(['dengan max flow nya sebesar ' num2str(f)]);
%% Find an Augmenting Path
    function F = findPath(A)            % BFS (Breadth-first
Search)
        q = zeros(1,len);               % queue
        pred = zeros(1,len);            % predecessor array
        front = 1; back = 2;
        pred(s) = s; q(front) = s;
        while front ~= back
            v = q(front);
            front = front + 1;
            for i = 1:len
                if pred(i) == 0 && A(v,i) > 0
                    q(back) = i;
                    back = back + 1;
                    pred(i) = v;
                end
            end
        end
        path = zeros(1,len);
        if pred(t) ~= 0
            i = t; c = 1;
            while pred(i) ~= i
                path(c) = i;
```

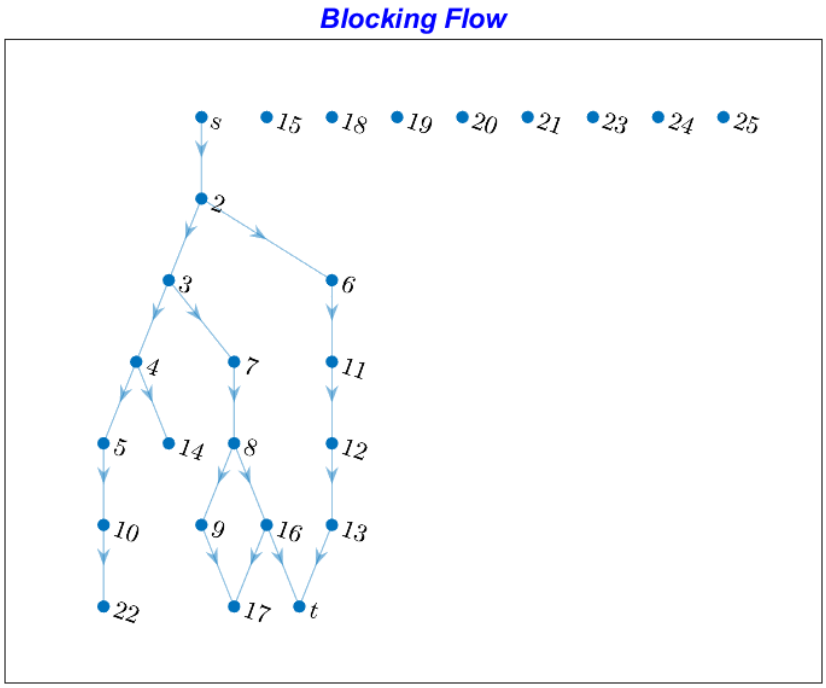
```
                    c = c + 1;
                    i = pred(i);
                end
                path(c) = s;
                path(c+1:len) = [];
            end
            F = path;
        end
end
```

- Aliran awal yang tiap sisinya bernilai 0 ampere karena belum terjadi peningkatan.



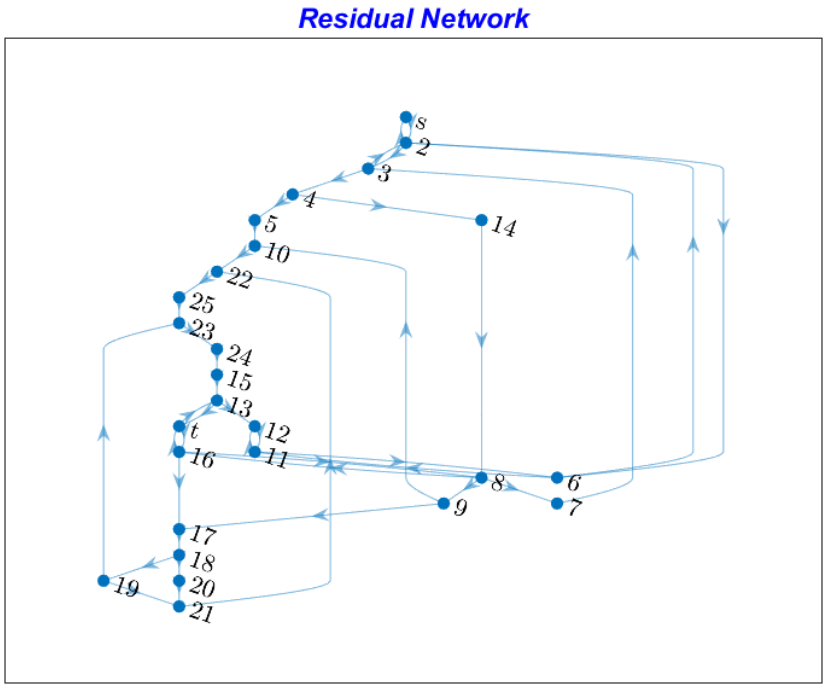
Gambar L. 6. 1 *Residual network* jaringan awal algoritma Dinic tujuan Mall Panakkukang

- Konstruksi *blocking flow g* di *layered network* jalur s-2-6-11-12-13-t dan s-2-3-7-8-16-t.



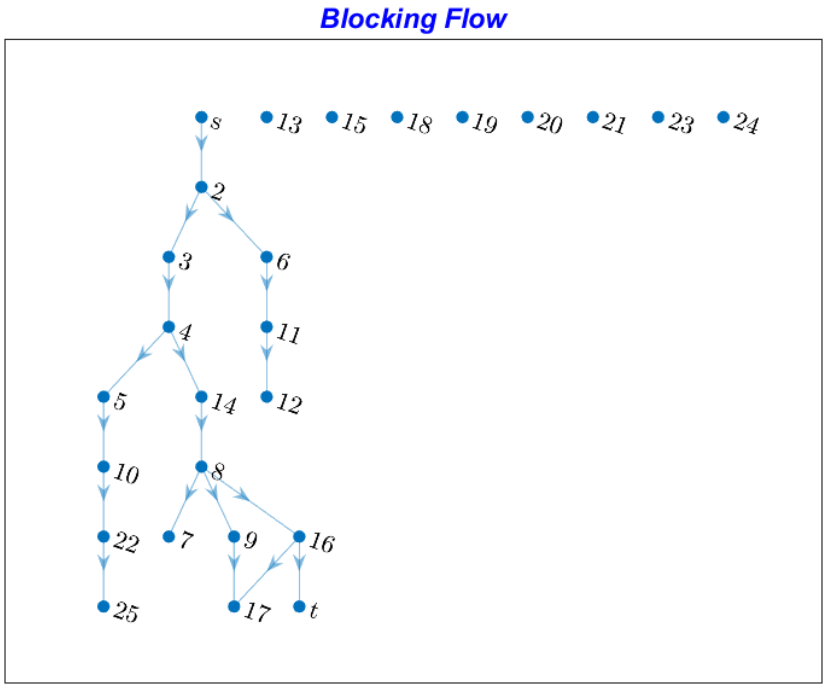
Gambar L. 6. 2 *Blocking flow* iterasi 1 algoritma Dinic tujuan Mall Panakkukang

- Didapat arus baru dan jaringan sisa untuk digunakan pada iterasi 2.



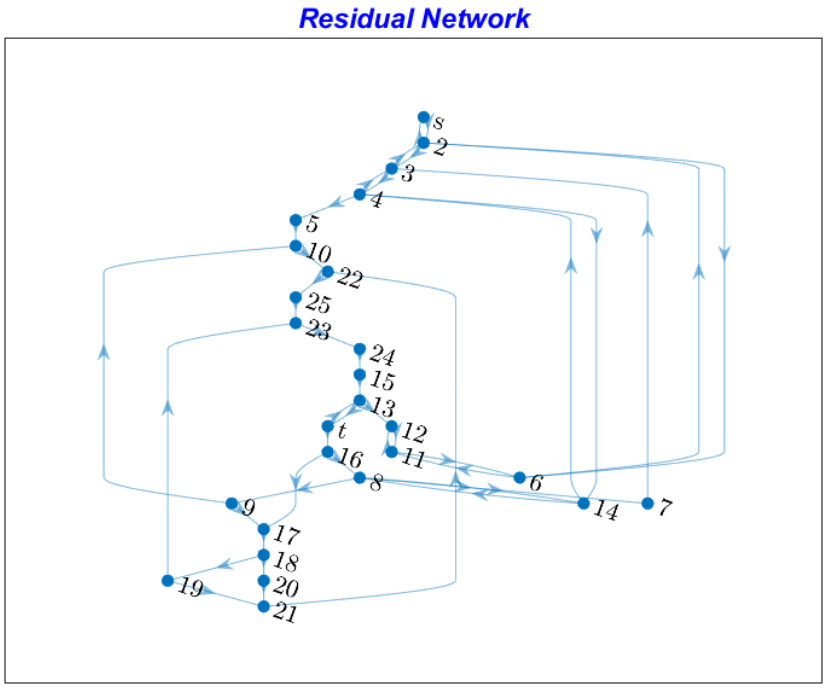
Gambar L. 6. 3 *Residual Network* iterasi 1 algoritma Dinic tujuan Mall Panakkukang

- Konstruksi *blocking flow g* di *layered network* jalur s-2-3-4-14-8-16-t.



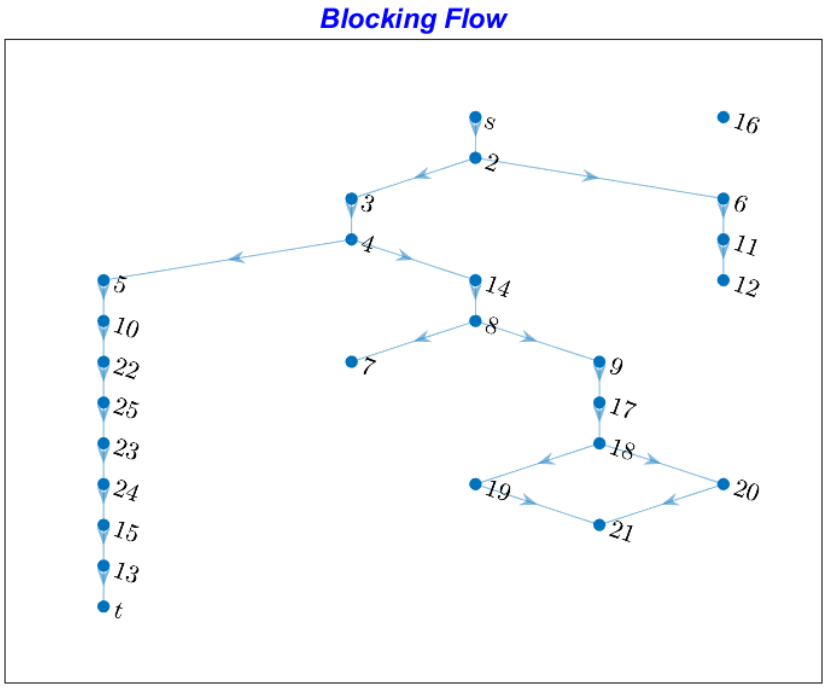
Gambar L. 6. 4 *Blocking flow* iterasi 2 algoritma Dinic tujuan Mall Panakkukang

- Didapat arus baru dan jaringan sisa untuk digunakan pada iterasi 3.



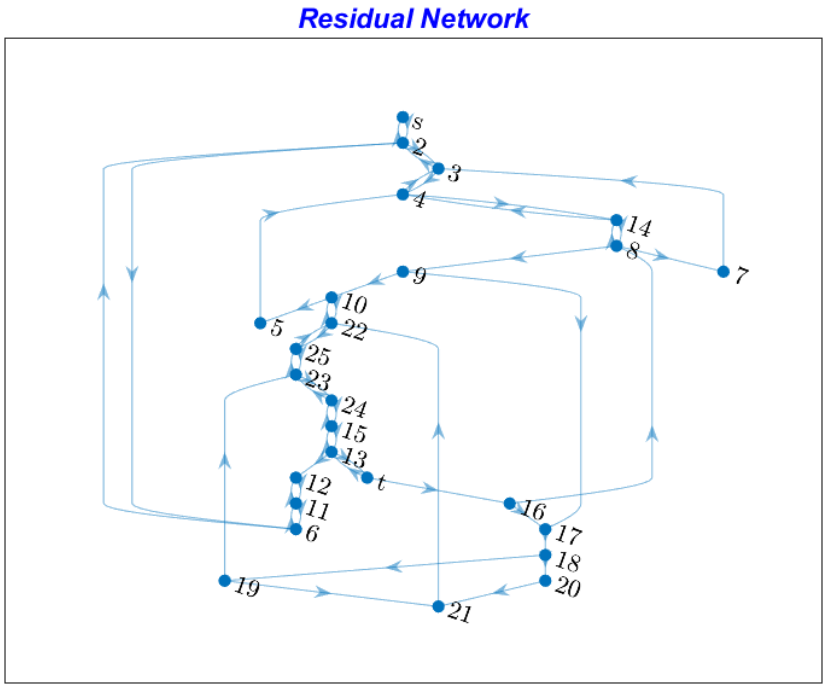
Gambar L. 6. 5 *Residual Network* iterasi 2 algoritma Dinic tujuan Mall Panakkukang

- Konstruksi *blocking flow g* di *layered network* jalur s-2-3-4-5-10-22-25-23-24-15-13-t.



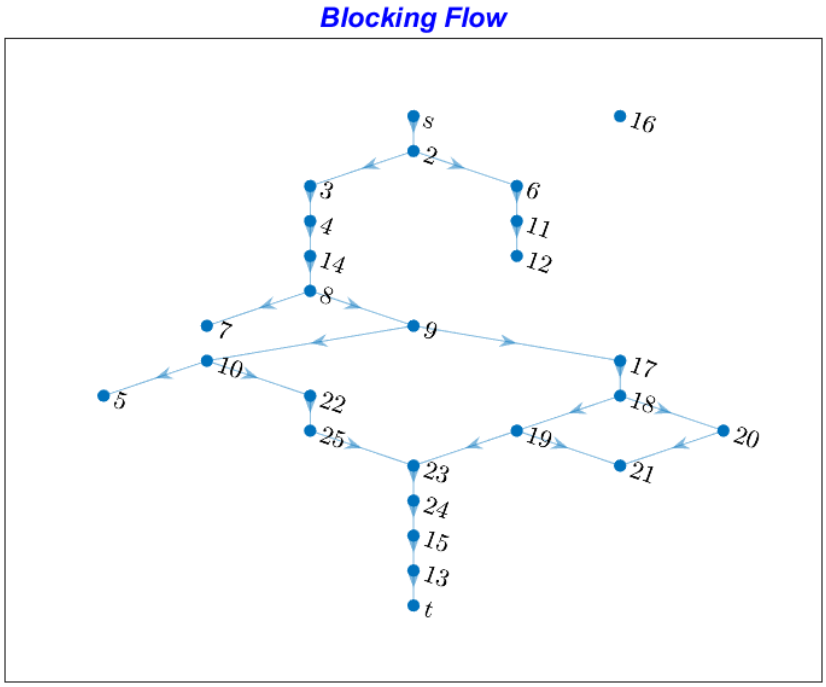
Gambar L. 6. 6 *Blocking flow* iterasi 3 algoritma Dinic tujuan Mall Panakkukang

- Didapat arus baru dan jaringan sisa untuk digunakan pada iterasi 4.



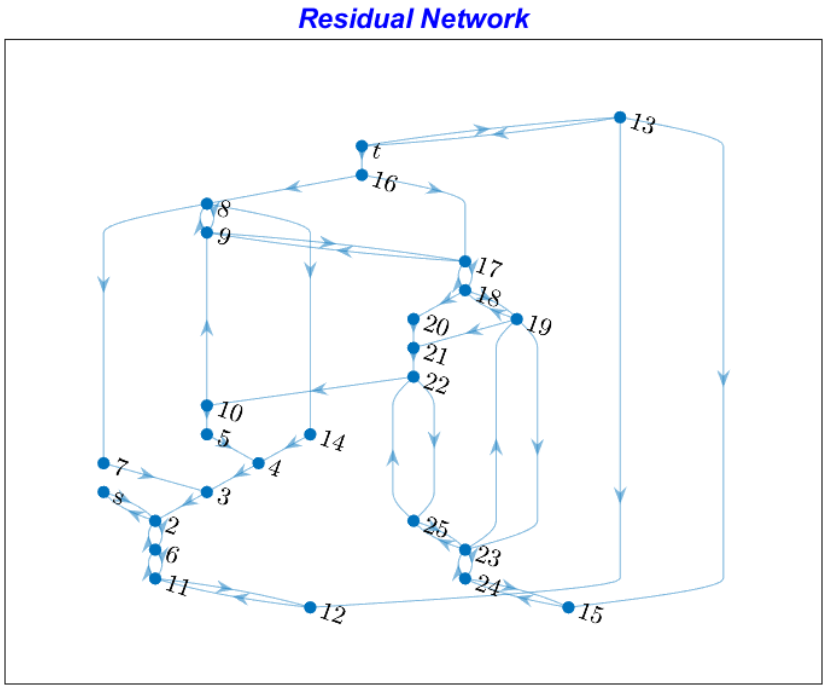
Gambar L. 6. 7 *Residual Network* iterasi 3 algoritma Dinic tujuan Mall Panakkukang

- Konstruksi *blocking flow g* di *layered network* jalur s-2-3-4-14-8-9-10-22-25-23-24-15-13-t dan 1-2-3-4-14-8-9-17-18-19-23-24-15-13-t



Gambar L. 6. 8 *Blocking flow* iterasi 4 algoritma Dinic tujuan Mall Panakkukang

- Didapat arus baru dan jaringan sisa untuk digunakan pada iterasi 5.



Gambar L. 6. 9 *Residual Network* iterasi 4 algoritma Dinic tujuan Mall Panakkukang

Lampiran 7 Algoritma Dinic pada aplikasi Matlab tujuan Menara Pinisi UNM

```
function dinic_unm
%% Input Network as Adjacency Matrix
    clc; clear; close all;

    atas = [1 2 6 11 12 13 2 3 7 8 16 3 4 14 4 5 8 9 9 10 16 17 18 18
20 19 21 19 22 25 23 24 15 12 26 29 31 28 15 30 30 15 27 28 32 33];
    bawah = [2 6 11 12 13 34 3 7 8 16 34 4 14 8 5 10 9 10 17 22 17 18
20 19 21 21 22 23 25 23 24 15 13 26 29 31 28 34 30 14 31 27 28 32 33
34];
    weight = [1785 425 425 425 255 1445 1360 255 255 510 510 1105 680
680 425 425 585 255 255 680 425 680 255 425 255 425 425 255 1105 1105
1360 1360 850 425 850 850 1105 1360 680 425 255 680 680 425 425 425];
    tes = digraph(atas,bawah,weight);
    city=full(adjacency(tes,'weight'));
    s = 1; t = max(bawah); f = 0; hasilflow=0;
    [kiri kanan] = findedge(tes);
    capacitytes = tes.Edges.Weight;
    felowtes = [];
    for i=1:length(capacitytes)
        felowtes = [felowtes;0];
    end
    [kirito] = [kiri];
    [kananto] = [kanan];
    selow = felowtes;
    B = string(felowtes);
    ss = string('s');
    st = string('t');
    titik = [ss 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31 32 33 st];
    jumlahiter = 0;

%% Dinic Algorithm
while  true
    jumlahiter = jumlahiter+1;
    A = string(capacitytes);
```

```
    str = '('+A+','+B+')';
    tes.Edges.str = str;
    figure
    z =
plot(tes,'Layout','layered','Interpreter','Latex','EdgeLabel',tes.Edge
s.str,'NodeLabel',titik,'edgefontsize',7,'nodefontsize',7);
    title('Residual Network','color','blue','fontangle','italic');
    [kiri kanan] = findedge(tes);
    felowtes = [];
    for i=1:length(capacitytes)
        felowtes = [felowtes;0];
    end
    sisi = table2cell(tes.Edges(:,1));
    teslama = tes;
    tesbaru = tes;
    layer = [repmat({'0'},length(atas),1)];
    V = [s];
    W = [];
    Iter = 1;
    nomor = 0;
    for i=1:length(atas)
        for j=Iter:length(V)
            T = successors(tes,V(j));
            for k=1:length(V)
                T(T==V(k))=[];
            end
            for m=1:length(T)
                nomor=nomor+1;
                layer(nomor)={[V(j) T(m)]};
            end
            for l=1:length(W)
                    T(T==W(l))=[];
            end
            W=[W;T];
        end
        Iter=Iter+1;
        V=[V;W];
```

```
        if V(V==t), break;end
        W=[];
    end
    if V~=t, break; end
    angka = 0;
    for i=1:length(sisi)
            for j=1:length(layer)
                if isequal(layer{j},sisi{i})
                    teslama=rmedge(teslama,i-angka);
                    angka = angka+1;
                    continue
                end
            end
    end
    teslama = table2cell(teslama.Edges(:,1));
    angka = 0;
    for i=1:length(sisi)
            for j=1:length(teslama)
                if isequal(teslama{j},sisi{i})
                    tesbaru=rmedge(tesbaru,i-angka);
                    angka = angka+1;
                    continue
                end
            end
    end
    [left right] = findedge(tesbaru);
    capacitytesbaru = tesbaru.Edges.Weight;
    felowtesbaru = [];
    for i=1:length(capacitytesbaru)
        felowtesbaru = [felowtesbaru;0];
    end
    cap = full(adjacency(tesbaru,'weight'));
    len = length(cap);
    figure
    z =
plot(tesbaru,'Layout','layered','Interpreter','Latex','EdgeLabel',tesb
aru.Edges.Weight,'NodeLabel',titik,'edgefontsize',7,'nodefontsize',7);
```

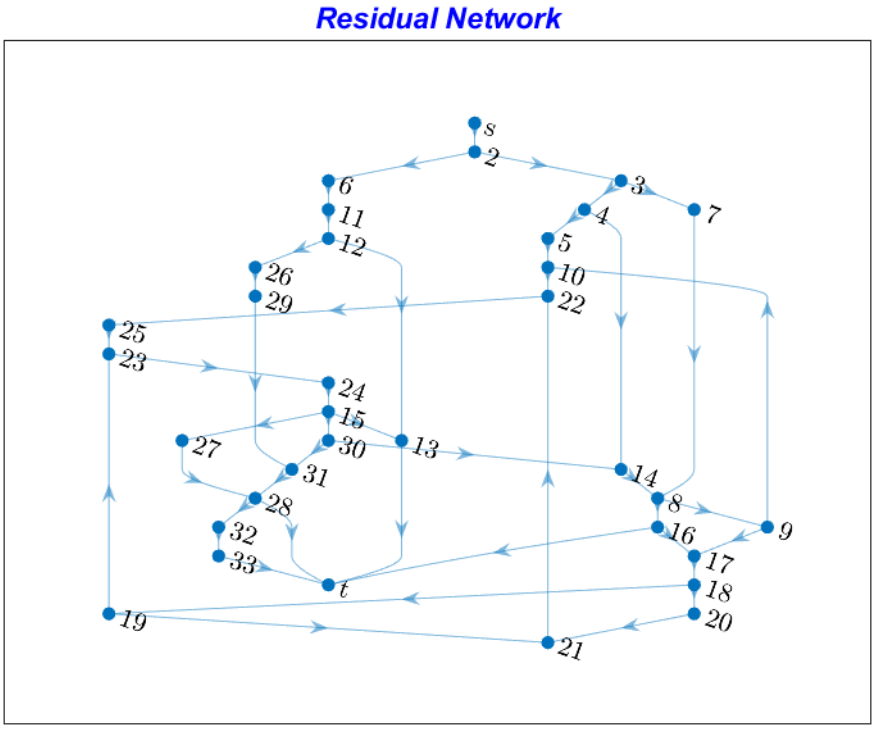
```
    title('Layeres Network','color','blue','fontangle','italic');
    while true
        p = findPath(cap);
        Z = fliplr(p);
        if p(1) == 0, break;end
        flow = max(max(cap));
        for j = 2:length(p)
            flow = min(flow,cap(p(j),p(j-1)));
        end
        for j = 2:length(p)
            a = p(j); b = p(j-1);
            cap(a,b) = cap(a,b) - flow;
            city(a,b) = city(a,b) - flow;
            cap(b,a) = cap(b,a) + flow;
            city(b,a) = city(b,a)+flow;
        end
        f = f + flow;
        hasilflow = hasilflow+flow;
        for i=1:length(Z)-1
            G = [Z(i) Z(i+1)];
            for j = 1:length(left)
                if [left(j) right(j)] == G
                    felowtesbaru(j) = felowtesbaru(j)+flow;
                end
            end
        end
        for i=1:length(Z)-1
            G = [Z(i) Z(i+1)];
            for j = 1:length(kiri)
                if [kiri(j) kanan(j)] == G
                    felowtes(j) = felowtes(j)+flow;
                end
            end
        end
    end
    tes = digraph(city);
    [kiribaru kananbaru] = findedge(tes);
```

```
    capacitytes = tes.Edges.Weight;
    felow = [];
    for i=1:length(capacitytes)
        felow = [felow;0];
    end
    for i=1:length(kirito)
        for j=1:length(kiribaru)
            if [kirito(i) kananto(i)]==[kiribaru(j) kananbaru(j)]
                felow(j)=selow(i);
            end
        end
    end
    selow=felow;
    [kirito] = [kiribaru];
    [kananto] = [kananbaru];
    felow = [];
    for i=1:length(capacitytes)
        felow = [felow;0];
    end
    for i=1:length(kiri)
        for j=1:length(kirito)
            if [kiri(i) kanan(i)] == [kananto(j) kirito(j)]
                selow(j) = selow(j) + felowtes(i);
            end
        end
    end
    for i=1:length(kirito)
        for j=1:length(kiribaru)
            if [kirito(i) kananto(i)] == [kiribaru(j) kananbaru(j)]
                felow(j) = selow(i);
            end
        end
    end
    disp(' ');
    B = string(felow);
    capacitytes = tes.Edges.Weight;
    fprintf('Diperoleh jumlah flow pada iterasi ke-%g  sebesar
```

```
%g\n',jumlahiter,hasilflow);
    disp(['dengan total flow nya sebesar ' num2str(f)]);
    figure
    z =
plot(tesbaru,'Layout','layered','Interpreter','Latex','EdgeLabel',felo
wtesbaru,'NodeLabel',titik,'edgefontsize',7,'nodefontsize',7);
    title('Blocking Flow','color','blue','fontangle','italic');
    hasilflow=0;
end
disp(' ');
disp('Karena sudah tidak terdapat lagi augmenting path maka iterasi
terhenti');
disp(['dengan max flow nya sebesar ' num2str(f)]);
%% Find an Augmenting Path
    function F = findPath(A)            % BFS (Breadth-first Search)
        q = zeros(1,len);               % queue
        pred = zeros(1,len);            % predecessor array
        front = 1; back = 2;
        pred(s) = s; q(front) = s;
        while front ~= back
            v = q(front);
            front = front + 1;
            for i = 1:len
                if pred(i) == 0 && A(v,i) > 0
                    q(back) = i;
                    back = back + 1;
                    pred(i) = v;
                end
            end
        end
        path = zeros(1,len);
        if pred(t) ~= 0
            i = t; c = 1;
            while pred(i) ~= i
                path(c) = i;
                c = c + 1;
                i = pred(i);
```

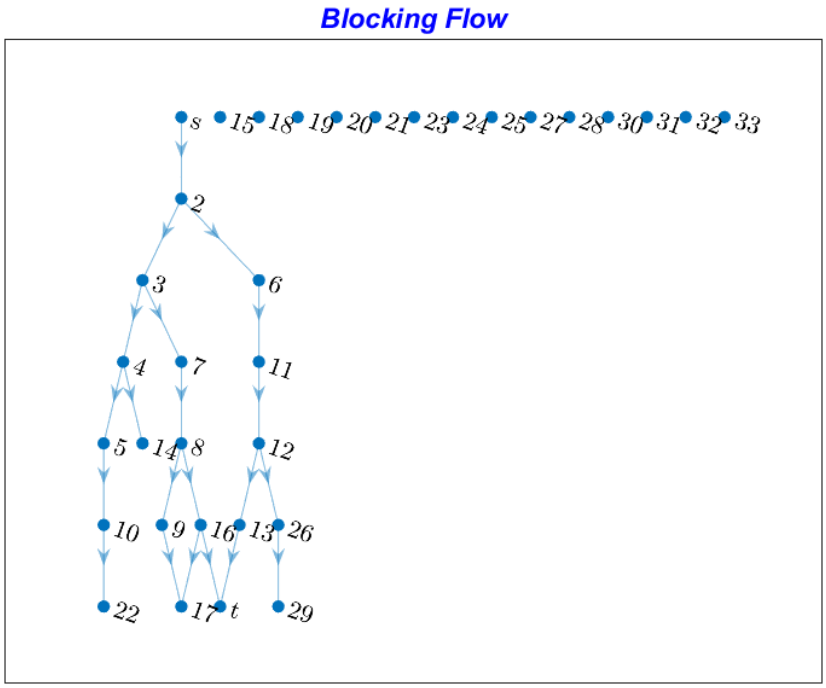
```
            end
            path(c) = s;
            path(c+1:len) = [];
        end
        F = path;
    end
end
```

- Aliran awal yang tiap sisinya bernilai 0 ampere karena belum terjadi peningkatan.



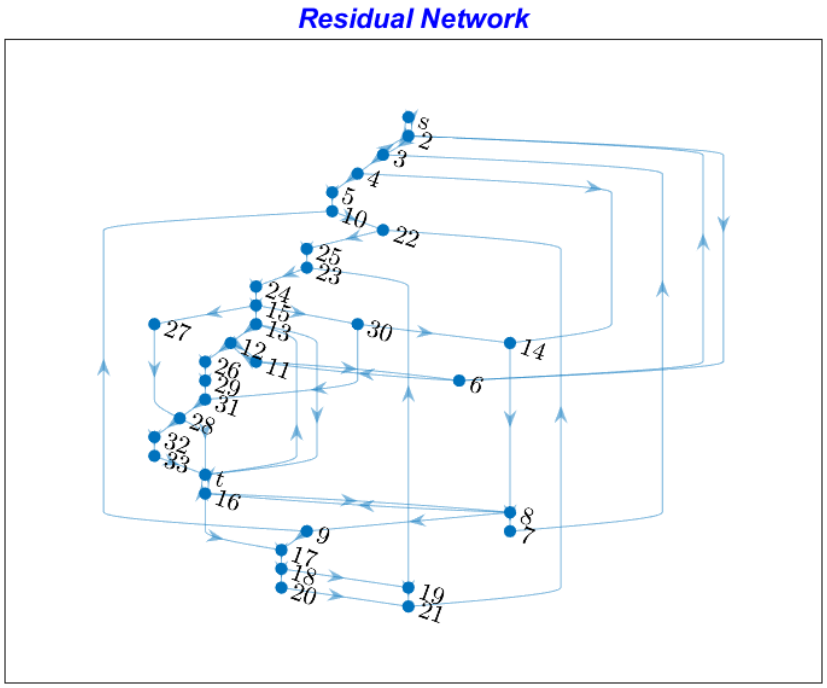
Gambar L. 7. 1 *Residual network* jaringan awal algoritma Dinic tujuan Menara Pinisi UNM

- Konstruksi *blocking flow g* di *layered network* jalur s-2-3-7-8-16-t dan s-2-6-11-12-13-t.



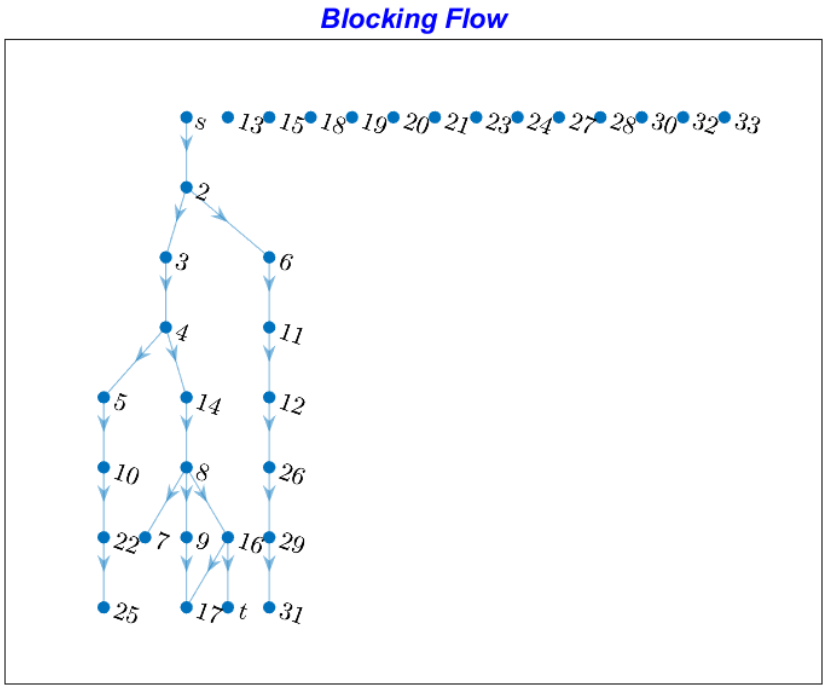
Gambar L. 7. 2 *Blocking flow* iterasi 1 algoritma Dinic tujuan Menara Pinisi UNM

- Didapat arus baru dan jaringan sisa untuk digunakan pada iterasi 2.



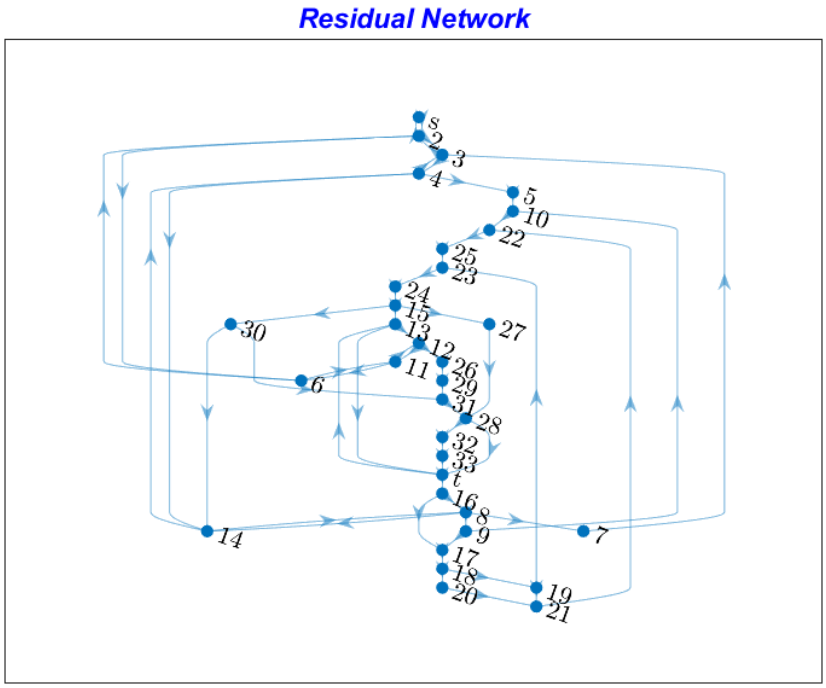
Gambar L. 7. 3 *Residual Network* iterasi 1 algoritma Dinic tujuan Menara Pinisi UNM

- Konstruksi *blocking flow g* di *layered network* jalur s-2-3-4-14-8-16-t.



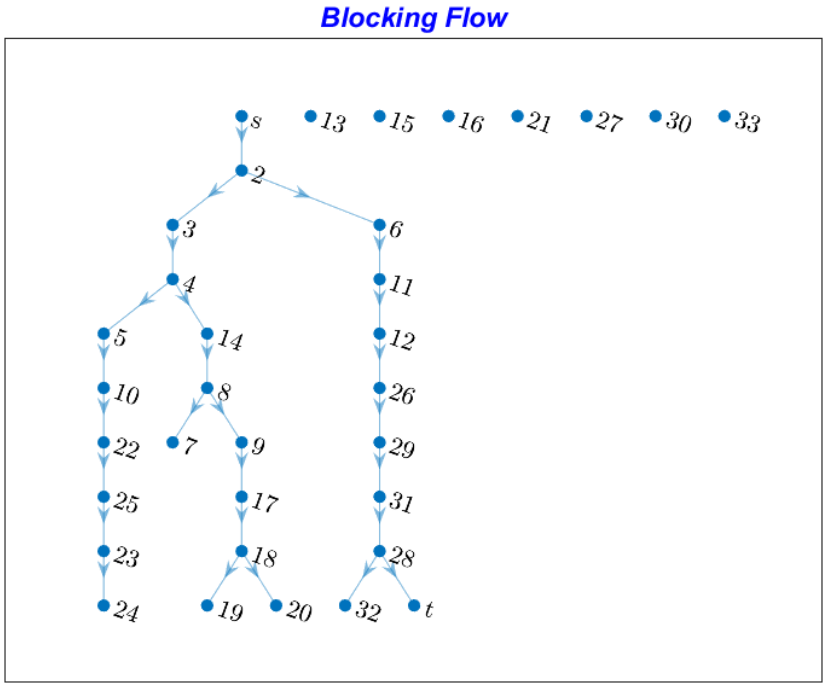
Gambar L. 7. 4 *Blocking flow* iterasi 2 algoritma Dinic tujuan Menara Pinisi UNM

- Didapat arus baru dan jaringan sisa untuk digunakan pada iterasi 3.



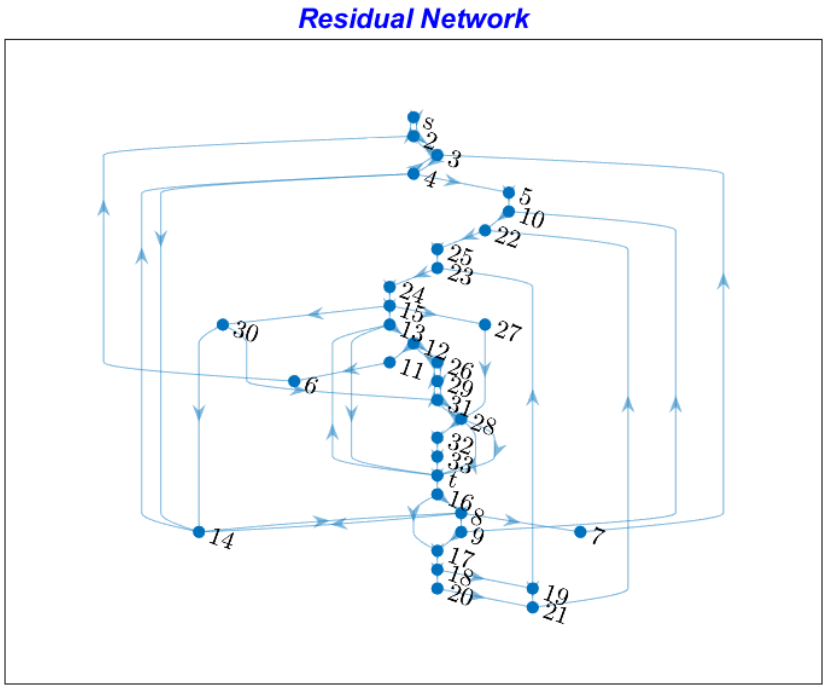
Gambar L. 7. 5 *Residual Network* iterasi 2 algoritma Dinic tujuan Menara Pinisi UNM

- Konstruksi *blocking flow g* di *layered network* jalur s-2-6-11-12-26-29-31-28-t.



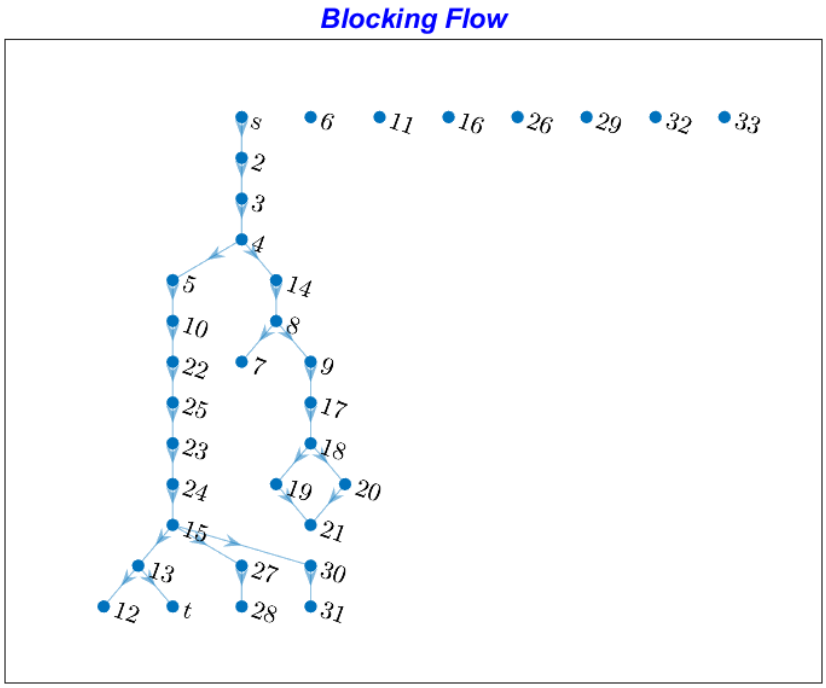
Gambar L. 7. 6 *Blocking flow* iterasi 3 algoritma Dinic tujuan Menara Pinisi UNM

- Didapat arus baru dan jaringan sisa untuk digunakan pada iterasi 4.



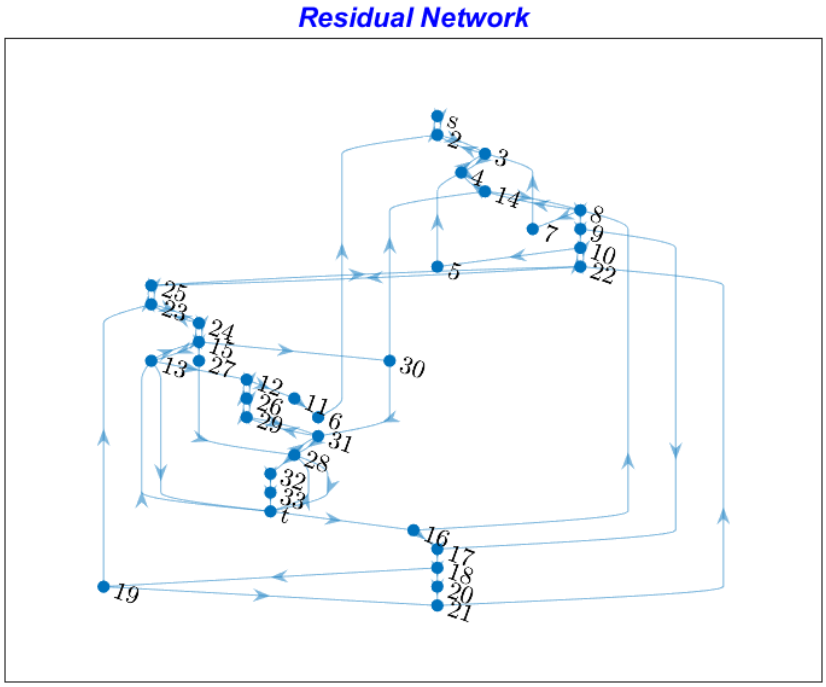
Gambar L. 7. 7 *Residual Network* iterasi 3 algoritma Dinic tujuan Menara Pinisi UNM

- Konstruksi *blocking flow g* di *layered network* jalur s-2-3-4-5-10-22-25-23-24-15-13-t.



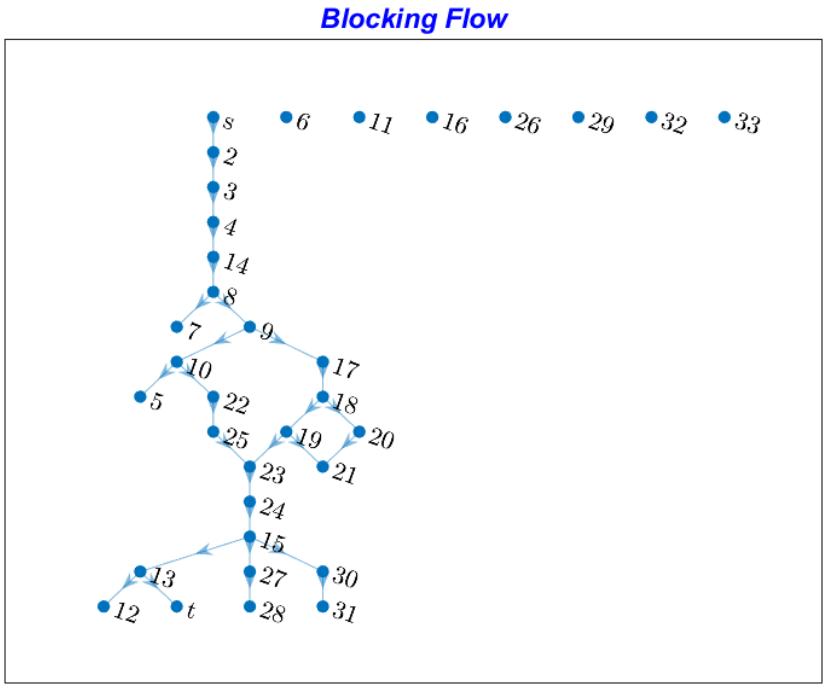
Gambar L. 7. 8 *Blocking flow* iterasi 4 algoritma Dinic tujuan Menara Pinisi UNM

- Didapat arus baru dan jaringan sisa untuk digunakan pada iterasi 5.



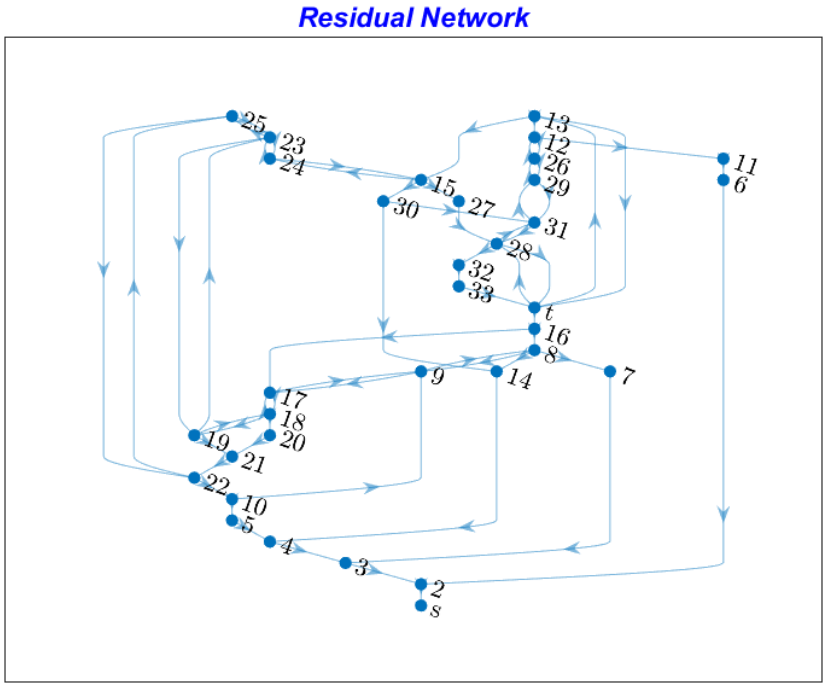
Gambar L. 7. 9 *Residual Network* iterasi 4 algoritma Dinic tujuan Menara Pinisi UNM

- Konstruksi *blocking flow g* di *layered network* jalur s-2-3-4-14-8-9-10-22-25-23-24-15-13-t.



Gambar L. 7. 10 *Blocking flow* iterasi 5 algoritma Dinic tujuan Menara Pinisi UNM

- Didapat arus baru dan jaringan sisa untuk digunakan pada iterasi 6.



Gambar L. 7. 11 *Residual Network* iterasi 5 algoritma Dinic tujuan Menara Pinisi UNM