Komponen Terhubung dan Jalur Terpendek Algoritma Graf Paralel

Yosef Sukianto – Nim 13506035
Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika,
Institut Teknologi Bandung, Jalan Ganesha 10, Bandung
Email : if16035@students.if.itb.ac.id

Abstract

Suatu graf berarah merupakan kumpulan berhingga simpul dan ruas yang menghubungkan simpul simpul dan mempunyai arah tujuan serta hubungan satu arah. Representasi matriks digunakan untuk penyimpanan dan manipulasi graf. Graf G dengan himpunan simpul $V = \{v_1,v_2,...,v_n\}$, dinyatakan sebagai matriks ajasensi A yang simetris order n x n dengan masukan $a_{ij} = 1$ untuk vi dan vj terhubung dan $a_{ij} = 0$ untuk lainnya, $1 \le i,j \le n$. Graf dikatakan graf berbobot, jika setiap ruas memiliki nilai yang real dan dapat berarah atau tidak berarah. Unsur wij matriks bobot W menyatakan bobot ruas (v_i,v_j). Jika v_i dan v_j tidak terhubung, maka $w_{ij} = 0$. Suatu jalur dari simpul awal vi ke simpul tujuan vj dalam graf G = (V,E), adalah barisan ruas-ruas (v_i,v_k), (v_k,v_l), ..., (v_m,v_j) dari E, dimana tidak ada simpul yang muncul lebih dari sekali.

Matriks terhubung graf G dengan n-simpul adalah matriks C, n x n dengan unsur yang didefinisikan c_{jk} , j,k = 1, 2, ..., n. Melalui matriks ajasensi A, dapat dihitung matriks terhubung C. Digunakan pendekatan perkalian matriks Boolean B yang analog dengan perkalian matriks. Matriks B menyatakan semua jalur dalam graf G dengan panjang $b_{jk} = 1$ jika terdapat jalur dengan panjang 0 atau 1 dari v_j ke v_k dan $b_{jk} = 0$ untuk lainnya. Dengan cara yang sama dihitung B^2 B^4 sampai B^n, sehingga diperoleh bahwa $B^n \approx B^{n-1}$ yang menyatakan jalur dengan panjang k.

Dalam makalah ini akan dibahas pencarian komponen terhubung dan jalur terpendek dengan prosedur CUBE-CONNECTIVITY yaitu prosedur perkalian matriks dengan model kubus-terhubung yang mengambil matriks ajasensi A sebagai masukan dan matriks terhubung C sebagai output dengan $N = n^3$ prosesor.

***Kata Kunci** : Graf, Matriks Ajasensi, Jalur terpendek**,** Cube Shortest Path*

1. PENDAHULUAN

Graf mempunyai peranan penting dalam membantu pemodelan untuk menyelesaikan berbagai persoalan optimasi seperti sistem penjadwalan, perjalanan, transportasi dan aliran jaringan. Algoritma graf paralel merupakan suatu metode penyelesaian bagi permasalahan yang dapat dinyatakan dengan graf dan dirancang untuk komputer paralel yang dapat menurunkan biaya, mengefisienkan ruang memori dan mempercepat perhitungan. Dalam mencari penyelesaian untuk berbagai persoalan optimasi tersebut di atas, adalah penting untuk mengembangkan algoritma yang efisien. Pembahasan difokuskan pada suatu persoalan yang dapat dinyatakan dalam graf dengan algoritma graf paralel dalam prosedur PASCAL.

Suatu graf berarah terdiri dari kumpulan berhingga simpul dan ruas yang menghubungkan simpul-simpul dan mempunyai arah tujuan serta hubungan satu arah. Representasi matriks digunakan untuk penyimpanan dan manipulasi graf. Suatu graf *G* dengan sekumpulan simpul $V = \{v_1,v_2,...,v_n\}$ maka graf *G* secara tunggal dapat dinyatakan sebagai suatu matriks ajasensi *A* order *n* x *n* yang simetris dengan masukan a_{ij}, $1 \le i,j \le n$, yang didefinisikan sebagai berikut:

$$a_{ij} = \begin{cases} 0 & \\ 1 & v_i \text{ dan } v_j \text{ terhubung lainnya} \end{cases}$$

Jika setiap ruas dari graf berasosiasi dengan bilangan real atau bobot, maka graf dikatakan mempunyai bobot. Pengertian ruas berbobot bervariasi dari satu aplikasi ke aplikasi lainnya, mungkin menyatakan jarak, biaya, waktu, kemungkinan dan lainnya. Matrix bobot *W* digunakan untuk menyatakan graf berbobot. Unsur w_{ij} menyatakan bobot dari ruas (v_i,v_j). Jika v_i dan v_j tidak terhubung dengan suatu ruas, maka w_{ij} dapat sama dengan nol, atau mempunyai suatu nilai sesuai dengan aplikasinya

Suatu jalur dari simpul awal *vi* ke simpul tujuan v_j dalam graf *G = (V,E)*, adalah barisan ruas-ruas (v_i,v_j), (v_k,v_l), ..., (v_m,v_j) dari *E*, dimana tidak ada simpul yang muncul lebih dari sekali. Subgraf $G' = (V',E')$ dari graf $G = (V,E)$ adalah graf sehingga $V' \subseteq V'$ dan $E' \subseteq E'$, berarti subgraf tersebut dengan simpul dan ruasnya berada dalam graf *G*.

2. TINJAUAN PUSTAKA

2.1. Matriks Terhubung

Misalkan $G = (V, E)$ adalah sebuah graf sederhana dengan $|V| = n$. Anggap simpul pada G disusun dengan urutan v_1 , v_2…, v_n Matriks kedekatan (*Adjacency matrix*) dari graf G, A_g, yang berkaitan dengan simpul-simpul, adalah sebuah matriks boolean $n \times n$ dengan elemen ke (i, j) berharga 1 jika v_i dan v_j bertetangga, dan selainnya itu berharga 0.

$A_{ij} =$ 1 { , } h h
0 { , } h

Matriks Terhubung dari suatu graf G dengan n-simpul adalah suatu matriks C dengan order n x n dengan
Melalui matriks ajasensi A dari graf G, dapat dihitung matriks terhubung C. Dalam hal ini digunakan pendekatan perkalian matriks *Boolean* berikut yang analog dengan perkalian matriks pada umumnya

(i) Hasil perkalian matriks semua unsurnya mempunyai nilai biner dengan nilai masukan 0 atau 1;
(ii) Operasi Boolean "dan", mengikuti operasi perkalian pada umumnya yaitu 0 "dan" 0 = 0, 0 "dan" 1 = 0, 1 "dan" 0 = 0 dan 1 "dan" 1 = 1;
(iii) Operasi Boolean "atau", mengikuti operasi penjumlahan umumnya yaitu 0 "atau" 0 = 0, 0 "atau" 1 = 1, 1 "atau" 0 = 1 dan 1 "atau" 1 = 1.

Jika X, Y dan Z matriks Boolean order n x n dimana Z adalah perkalian Boolean dari X dan Y, maka

$z_{ij} = (x_{i1}$ dan $y_{1j})$ atau $(x_{i2}$ dan $y_{2j})$ atau ... atau $(x_{in}$ dan $y_{nj})$ untuk $i,j = 1, 2, ..,n$

Tahap pertama dalam perhitungan matriks terhubung C adalah memperoleh matriks B order n x n dari matrik ajsensi A sebagai berikut; $b_{jk} = a_{jk}$ untuk $(j \neq k)$ dan $b_{jj} = 1$ untuk $j,k = 1...n$. Matriks B menyatakan semua jalur dalam graf G dengan panjang berikut:

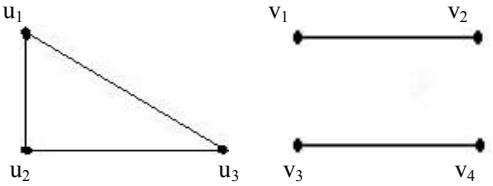
$b_{jk} =$ 0
1

terdapat jalur dengan panjang 0 atau 1 dari v_j ke v_k lainnya

Dengan cara yang sama dihitung B^2 (perkalian Boolean B dengan dirinya), $B^4,...,B^n$, sehingga diperoleh $B^n \approx B^{n-1}$ yang menyatakan jalur dengan panjang k. Akan diselidiki bahwa terdapat jalur dari v_i ke v_j. Untuk menerapkan algoritma dalam pemrosesan paralel, digunakan prosedur perkalian matriks dengan model kubus-terhubung (*CUBE-CONNECTED*) untuk membentuk perkalian matriks Boolean. Hasil dari algoritma diberikan dalam prosedur komponen terhubung CUBE_COMPONENT_CONECTIVITY yaitu prosedur mengambil matriks ajasensi sebagai masukan dan mengembalikan matriks C sebagai output. Hal ini dijalankan pada komputer SIMD kubus-terhubung dengan $N = n^3$ prosesor $P_1,P_2,...,P_N$. Prosesor dapat diatur dalam suatu pola n x n x n jajaran. Dalam jajaran ini, P_r berada dalam posisi (i,j,k) dimana $r = in2+jn+k$ dan $1 \leq i,j,k \leq n$. Berarti terdapat tiga register $A(i,j,k)$, $B(i,j,k)$ dan $C(i,j,k)$. Diinisialisasi prosesor dalam posisi $(1,j,k)$, $1 \leq j,k \leq n$ yang terdiri dari matriks ajasensi $A(1,j,k) = a_{jk}$. Pada akhir komputasi, prosesor-prosesor ini mengandung matriks terhubung, berarti bahwa $C(1,j,k) = C_{jk}$, $1 \leq j,k \leq n$.

2.2. Komponen Terhubung

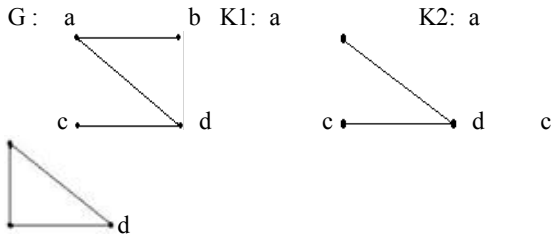
Graf G dikatakan *terhubung* (*connected*) jika setiap dua titik u,v di G, terdapat lintasan yang menghubungkan kedua titik tersebut. Graf G dikatakan *graf tak terhubung* (*disconnected*) jika ada dua titik di G yang tidak mempunyai lintasan.



Gambar 2.1(a) Gambar 2.1(b)

Gambar 2.1(a) adalah graf terhubung dan Gambar 2.1(b) adalah graf tak terhubung.

Graf K dikatakan *subgraf* dari graf G jika semua titik di K dan semua sisi di K adalah titik dan sisi di G. sebagai contoh pada Gambar 2.2, $K1$ adalah subgraf dari G tetapi $K2$ bukan subgraf G karena ada sisi ac di $(K2)$ yang bukan sisi di $E(G)$.



Gambar 2.2 Graf dan Subgrafnya

Kompenen dari graf G adalah subgraf terhubung maksimum dari G. Jadi graf terhubung mempunyai paling banyak satu komponen sedangkan graf tak terhubung paling sedikit mempunyai dua komponen. Contoh, pada Gambar 2.1(a) menunjukkan graf terhubung dan Gambar 2.1(b) menunjukkan graf tak terhubung dengan dua komponen

Suatu graf G dengan n-simpul yang tidak berarah dinyatakan oleh matriks ajasensi. Dapat dicari penyelesaian dari persoalan komponen-terhubung dengan pertama menghitung matriks terhubung C dari G. Menggunakan matriks-terhubung C, dapat dibentuk

perkalian matriks berulang D dengan order n x n dengan masukan yang didefinisikan dengan

$d_{jk} = \begin{matrix} 0 \\ 1 \end{matrix}$ *jika* $c_{jk} = 1$ *lainnya*

untuk $1 \leq j,k \leq n$. Dapat juga dinyatakan bahwa baris j dari D mengandung nama-nama simpul yang berhubungan dalam suatu jalur sampai dengan simpul v_j. Akhirnya, graf dapat tersusun kedalam komponen-terhubung dengan menunjuk tiap simpul v_j ke suatu komponen l dimana $d_{jl} \neq 0$.

Penerapan paralel untuk pendekatan komponen-terhubung digunakan prosedur kubus terhubung matriks C. Algoritma untuk komponenterhubung dinyatakan dalam prosedur CUBE COMPONENT CONECTIVITY. Dibuat prosedur dalam PASCAL yang diproses pada komputer SIMD kubus-terhubung dengan $N=n^3$ prosesor, yaitu masing-masing register A, B dan C. Prosesor-prosesor disusun dalam suatu pola n x n x n jajaran. Nilai $A(1,j,k) = a_{jk}$, untuk $1 \leq j, k \leq n$ menunjukkan prosesor dalam posisi $(1,j,k)$ yang mengandung matriks ajasensi graf G. Bilamana prosedur selesai diproses, $C(1,j,1)$ mengandung sejumlah komponen untuk simpul v_j, dimana $j = 1, 2, ..., n$.

2.3. Jalur terpendek

Dalam kasus mencari semua pasangan jalur terpendek, untuk graf berarah dan berbobot $G = (V, E)$, bobot dari ruas (v_i,v_j) menyatakan panjang ruas tersebut. Untuk setiap pasang simpul vi dan v_j dalam kumpulan simpul V, dimungkinkan menemukan jalur terpendek dari v_i ke v_j sepanjang ruas dalam graf. Panjang jalur adalah jumlah panjang ruas yang dibentuknya. Semua pasangan jalur terpendek dinyatakan sebagai berikut:

(i) Suatu n-simpul graf G diberikan oleh matriks bobot W dengan order n x n;

(ii) Bentuk suatu matriks D berukuran n x n demikian sehingga d_{ij} adalah panjang jalur terpendek dari v_i ke v_j dalam G untuk semua i dan j. Diasumsikan bahwa W bernilai masukan yang positif.

Misal d menyatakan panjang jalur terpendek dari v_i ke v_j, yang mana terdapat k-1 simpul yang dilaluinya. Nilai $d = w_{ij}$, berarti w_{ij} bobot pada ruas v_i ke v_j. Jika tidak ada ruas dari v_i ke vj, dimana i dan j berbeda, $d = \infty$ sedangkan $d = 0$ untuk $i = j$. Diasumsikan bahwa G tidak mempunyai siklus dengan kunjungan beberapa simpul lebih dari satu kali dalam jalur terpendek dari v_i ke v_j (kecuali jika didefinisikan suatu jalur yang mengijinkan untuk suatu simpul yang muncul lebih dari sekali pada suatu jalur). Berlaku bahwa $d_{ij} = d$. Dalam menghitung d untuk $k > 1$, digunakan perhitungan $d = min\{ d + d \}$ berarti bahwa, d adalah sama dengan nilai terkecil dari $\{ d + d \}$, untuk semua nilai l. Matriks D dapat dibangkitkan dari D^1 dengan menghitung D^2, D^4, ..., D^{n-1}, kemudian mengambil $D = D^{n-1}$. Dalam memperoleh matriks D^k, digunakan perkalian matriks yang mana operasi baku matriks perkalian yaitu **x** dan + diganti dengan + dan *minimum*. Kemudian jika prosedur matriks perkalian digunakan, hal ini dapat dimodifikasi untuk membangkitkan D^{n-1} dari D^1.

Algoritma ini diterapkan dalam proses paralel menggunakan suatu prosedur perkalian matriks dengan bentuk perkalian (+,minimum). Dapat digunakan prosedur perkalian matriks dengan kubus-terhubung. Hasil algoritma diberikan dalam prosedur *CUBE SHORTEST PATHS* berikut ini. Prosedur ini diproses pada komputer SIMD kubus-terhubung dengan $N = n^3$ prosesor, masing-masing dengan tiga alamat A, B dan C. Prosesor dapat diatur dalam suatu pola jajaran n x n x n. Didefinisikan A(1,j,k) = w_{jk} untuk $1 \leq j, k \leq n$, berarti prosesor dalam posisi (1,j,k) mengandung matriks bobot dari G. Jika v_j, tidak terhubung untuk v_k atau jika $j = k$, maka $w_{jk} = 0$. Bilamana prosedur berakhir, $C(1,j,k)$ terdiri dari panjang jalur terpendek dari v_j ke v_k untuk $1 \leq j,k \leq n$.

2.4. Algoritma Komponen Terhubung, Penulisan Hasil dan Jalur Terpendek

```
Procedure
          CUBE_COMPONENT_CONECTIVITY;
Var       A,B,C : array[1..10, 1..10, 1..10] of integer;
          TOT,v : array[1..10, 1..10] of integer;
          i,j,k,n,m,l : Integer;
          FVar : Text;
          FName : String;
begin
{unsur diagonal matriks adjasensi didefinisikan
dengan nilai 1}
    WriteLn(FVar,'Matriks terhubung awal : ');
    WriteLn(FVar,' B^1 ');
    for j := 1 to n do
          begin
              A[1,j,j] := 1;
              end;
    for j := 1 to n do
          for k := 1 to n do
          begin
              B[1,j,k] := A[1,j,k];
              WriteLn(FVar,B[1,j,k]);
          end;
{Matriks terhubung atau jalur diperoleh melalui
perkalian matriks Boolean berulang}
    for l := 1 to m do
    begin
{Perkalian matriks dengan kubus terhubung}
    for i := 1 to n do
```

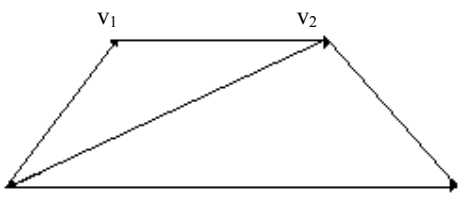
```
            for j := 1 to n do
                for k := 1 to n do
                begin
                    A[i,j,k] := A[1,j,k];
                    B[i,j,k] := B[1,j,k];
                end;
                for i := 1 to n do
                    for j := 1 to n do
                        for k := 1 to n do
                        begin
                            A[i,j,k] := A[i,j,i];
                            B[i,j,k] := B[i,i,k];
                        end;
    for  i := 1 to n do
            for j := 1 to n do
                for k := 1 to n do
                begin
                    C[i,j,k] := A[i,j,k]*B[i,j,k];
                end;
    for j := 1 to n do
            for k := 1 to n do
            begin
                TOT[j,k] := 0;
                for i := 1 to n do
                begin
                    TOT[j,k] := TOT[j,k] + C[i,j,k];
                    If  TOT[j,k] >= 1 then
                    begin
                        TOT[j,k] := 1;
                        C[1,j,k] := TOT[j,k];
                        v[j,k] := 1;
                    end;
                end;
            end;
            for j := 1 to n do
                for k:= 1 to n do
                begin
                    A[1,j,k] := TOT[j,k];
                    B[1,j,k] := TOT[j,k];
                end;
            end;
end;
{Menyatakan nomer komponen tiap simpul}
            for j := 1 to n do
                for k := 1 to n do
                begin
                    C[1,j,1] := l;
                end;
                end;

Procedure WRITEMATRIX;
var
            A,B,C : array[1..10, 1..10, 1..10] of integer;
            TOT,v : array[1..10, 1..10] of integer;
            i,j,k,n,m,l : Integer;
            FVar : Text;
            FName : String;
begin
    Writeln('Matriks Hasil :');
    WriteLn(FVar,'B^',m);
            for j := 1 to n do
                for k := 1 to n do
                begin
                    WriteLn(FVar,TOT[j,k]);
                end;
    for j := 1 to n do
    for k :=1 to n do
    begin
        WriteLn(FVar,'Komponen ke ',j,': ','v',k);
    end;
    Close(FVar);
end;

Procedure CUBE_SHORTEST_PATHS;
Var A,B,C : array[1..10, 1..10, 1..10] of integer;
      d : array[1..10, 1..10] of integer;
      i,j,k,n,m,l,tempmin,min,Max : integer;
      FVar : Text;
      FName : String;]
begin
{Bentuk matrix D1, simpan dalam register A & B}
{Unsur diagonal = 0 & bilangan terbesar diberikan
untuk 2 simpul yang tidak terhubung}
            Max := 1000;
            for j := 1 to n do
                for k := 1 to n do
                begin
                    if (j <> k) and (A[1,j,k] = 0) then
                    begin
                        A[1,j,k] := Max; {large number}
                    end;
                end;
                for j := 1 to n do
                    for k := 1 to n do
                    begin
                        B[1,j,k] := A[1,j,k];
                    end;
{ Bentuk matriks D^2, D^4, ..., D^n-1, melalui perkalian
matriks berulang}
                for l := 2 to m do
                begin
                    WriteLn(FVar,'D^',l);
{Perkalian matriks kubus-terhubung}
                    for i := 1 to n do
                        for j := 1 to n do
                            for k := 1 to n do
                            begin
                                A[i,j,k] := A[1,j,k];
                                B[i,j,k] := B[1,j,k];
                            end;
                    for i := 1 to n do
                        for j := 1 to n do
                            for k := 1 to n do
                            begin
                                A[i,j,k] := A[i,j,i];
                                B[i,j,k] := B[i,i,k];
                            end;
                    for j := 1 to n do
                        for k := 1 to n do
                        begin
```

```
                    for i := 1 to n do
                    begin
                    end;
                end;
            for j := 1 to n do
                for k := 1 to n do
                begin
                  for i := 1 to n do
                    begin
                      C[i,j,k] := A[i,j,k]+B[i,j,k];
                    end;
                end;
            for j := 1 to n do
                for k := 1 to n do
                begin
                      min := C[1,j,k];
                      for i := 2 to n do
                      begin
                          if min > C[i,j,k] then
                          begin
                              tempmin := min;
                              min := C[i,j,k];
                          end;
                      end;
                      d[j,k] := min;
                      if j = k then
                      begin
                            d[j,k] := 0;
                       end;
                end;
                for j := 1 to n do
                      for k:= 1 to n do
                      begin
                          A[1,j,k] := d[j,k];
                          B[1,j,k] := d[j,k];
                          if d[j,k] = Max then
                          begin
                                d[j,k] := 0;
                          end;
                          WriteLn(FVar,d[j,k]);
                      end;
        end;
end;
```

3. Hasil dan Pembahasan

3.1. Analisis Komponen Terhubung

Dimisalkan suatu persoalan dinyatakan dalam graf berarah pada gambar-3.1 berikut ini:



Gambar 3.1 Graf brarah G

Graf di atas dinyatakan dalam matriks adjasensi:

	v1	v2	v3	v4
v1	0	1	0	0
v2	0	0	1	1
v3	0	0	0	0
v4	1	0	1	0

Pada tahap 1 dan 2 dalam prosedur CUBE COMPONENT CONECTIVITY, unsur diagonal matriks adjasensi didefinisikan dengan nilai 1 dan diperoleh matriks awal:

$$B = \begin{matrix} 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{matrix}$$

Selanjutnya iterasi pertama dalam komponen terhubung atau jalur diperoleh melalui perkalian matriks Boolean berulang dengan kubus terhubung yang menghasilkan:

$$B^2 = \begin{matrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{matrix}$$

Sementara iterasi kedua telah konvergen yang menghasilkan bahwa $B^3 = B^2$.

$$B^3 = \begin{matrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{matrix}$$

Hasil Program :

```
Order Matriks : 4x4
derajat adjacency 3
Matriks Data :
        0 1 0 0
        0 0 1 1
        0 0 0 0
        1 0 1 0

Matriks terhubung awal :
B^1
        1 1 0 0
        0 1 1 1
        0 0 1 0
        1 0 1 1

Matriks Hasil :
B^2
        1 1 1 1
        1 1 1 1
```

0 0 1 0
1 1 1 1

B^3

1 1 1 1
1 1 1 1
0 0 1 0
1 1 1 1

Komponen ke 1: v1 v2 v3 v4
Komponen ke 2: v1 v2 v3 v4
Komponen ke 3: v3
Komponen ke 4: v1 v2 v3 v4

Waktu untuk melaksanakan prosedur komponen-terhubung adalah $\boldsymbol{O}(\log n)$. Tahap ini mengalami iterasi dengan waktu *log n*. Berarti waktu pelaksanaan keseluruhan dari prosedur matriks-terhubung adalah $t(n) = \boldsymbol{O}(\log^2 n)$. Karena jumlah prosesor $p(n) = n^3$, maka perkiraan biaya merupakan fungsi dari n yaitu $c(n) = \boldsymbol{O}(n^3 \log^2 n)$. Dari hasil program di atas diperoleh pada komponen ke 1 yaitu v1 terhubung dengan simpul v2, v3 dan v4. Demikian pula komponen ke 2 dan ke 4. Komponen ke 3 yaitu v3 tidak memiliki arah ke komponen lainnya.

3.2. Analisis Jalur Terpendek

Dimisalkan suatu persoalan dinyatakan dalam graf dengan simpul dan ruas yang diketahui bobotnya. Matriks Adjasensi dinyatakan berikut ini. Ruas yang tidak mempunyai bobot diberi tanda ∞ (pada program diberi nilai terbesar) dan untuk simpul terhadap dirinya sendiri ditandai dengan nol.

Matriks Adjasensi D :

$$D = \begin{matrix} 0 & 4 & 1 & \infty & 7 & \infty & \infty \\ \infty & 0 & 8 & \infty & \infty & \infty & \infty \\ \infty & \infty & 0 & 2 & 6 & \infty & \infty \\ \infty & 5 & \infty & 0 & \infty & \infty & 1 \\ \infty & \infty & \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 2 & 1 & 0 & \infty \\ \infty & 3 & \infty & \infty & \infty & 1 & 0 \end{matrix}$$

Hasil Program :
Matrix Order 7x7
derajat adjacency 5
Matrik Input :

0 4 1 0 7 0 0
0 0 8 0 0 0 0
0 0 0 2 6 0 0
0 5 0 0 0 0 1
0 0 0 0 0 0 0
0 0 0 2 1 0 0
0 3 0 0 0 1 0

D^2

0 4 1 3 7 0 0
0 0 8 10 14 0 0
0 7 0 2 6 0 3
0 4 13 0 0 2 1
0 0 0 0 0 0 0
0 7 0 2 1 0 3
0 3 11 3 2 1 0

D^3

0 4 1 3 7 5 4
0 0 8 10 14 12 11
0 6 0 2 5 4 3
0 4 12 0 3 2 1
0 0 0 0 0 0 0
0 6 14 2 1 0 3
0 3 11 3 2 1 0

D^4

0 4 1 3 6 5 4
0 0 8 10 13 12 11
0 6 0 2 5 4 3
0 4 12 0 3 2 1
0 0 0 0 0 0 0
0 6 14 2 1 0 3
0 3 11 3 2 1 0

D^5

0 4 1 3 6 5 4
0 0 8 10 13 12 11
0 6 0 2 5 4 3
0 4 12 0 3 2 1
0 0 0 0 0 0 0
0 6 14 2 1 0 3
0 3 11 3 2 1 0

Pada pembentukan matriks D, data disimpan dalam register A dan B. Matriks D^2, D^4, ..., D^{n-1} diproses melalui perkalian matriks secara berulang, yang dilaksanakan dalam waktu konstan O(log n). Terdapat log(n-1) iterasi untuk memproses perkalian matriks kubus yang mengambil waktu O(log n). Diperoleh waktu keseluruhan $t(n) = O(\log^2 n)$. Pada perkalian kubus-terhubung digunakan prosesor $p(n) = n^3$. Sehingga perkiraan biaya bernilai $c(n) = O(n^3 \log^2 n)$. Pada hasil program di atas matriks $D^5 = D^4$ sehingga iterasi telah konvergen dan tercapai nilai optimal. Diperoleh semua pasangan jalur-jalur terpendek. Misalkan jalur terpendek untuk v1→v2 = 4; v1→v3 = 1; v1→ v4 = 3; v1→v5 = 6; v1→ v7 = 4 dan seterusnya.

4. KESIMPULAN

Algoritma graf paralel merupakan metode penyelesaian bagi permasalahan yang dapat dinyatakan dengan graf dan dirancang untuk komputer paralel. Dalam menyelesaikan suatu masalah yang berhubungan dengan optimasi sangat penting untuk memikirkan bagaimana menciptakan suatu algoritma

yang efisien. Pencarian komponen terhubung dan jalur terpendek dapat dinyatakan dalam graf dengan algoritma graf paralel. Pembentukan matriks ajasensi dan matriks terhubung, dapat membantu pencarian komponen-komponen terhubung dan semua pasangan jalur-jalur terpendek. Algoritma ini diterapkan dalam proses paralel menggunakan prosedur perkalian matriks dengan model kubus terhubung yang disimulasikan dalam satu personal komputer asumsi memori yang dipakai adalah *virtual memory* dengan beberapa *address*.

DAFTAR REFERENSI

[1] http://redaree.itb.ac.id/~suksmono/lectures/el2009/ppt/8.%20Graf,%20Diagram%20Pohon%20dan%20Aplikasinya.pdf. Tanggal akses : 28 Desember 2007 pukul 13.30

[2] http://www.unej.ac.id/fakultas/mipa/skripsi/toriqul.pdf. Tanggal akses : 28 Desember 2007 pukul 14.00

[3] Suryadi, (1996), *Teori Graf Dasar*, Penerbit Gunadarma, Jakarta.

[4] Quinn, Michael J. (1994*), Parallel Computing, Theory and Practice*, Mc Graw-Hill Inc, Second Editions.