Aplikasi Algoritma DFS untuk Deteksi Celah Jaringan Komputer

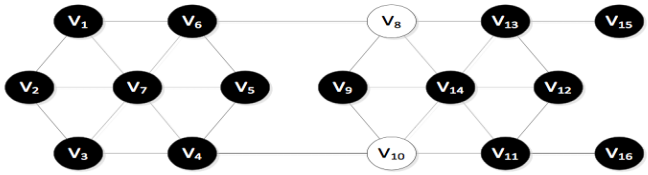
Akmal Fadlurohman 13515074
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132
13515074@std.stei.itb.ac.id, fadlurohmanakmal@rocketmail.com

***Abstrak*—Infrastruktur jaringan komputer sangat rentan terhadap serangan pada celah-celah (*vulnerabilities*) jaringan komputer yang seringkali tidak disadari keberadaanya. Celah dalam sebuah jaringan dapat berupa simpul artikulasi atau sebuah upagraf dari graf jaringan tersebut yang berbentuk siklik. Kerusakan pada sebuah komputer, yang direpresentasikan sebagai sebuah simpul dalam graf, dapat menimbulkan dampak yang besar pada sistem secara keseluruhan. Pada makalah ini, penulis membahas aplikasi algoritma *Depth First Search* (DFS) untuk menemukan simpul artikulasi dalam jaringan komputer dan mendeteksi keberadaan upagraf dari graf jaringan komputer yang berbentuk siklik.**

***Kata Kunci*—Celah Jaringan, Simpul Artikulasi, Siklus Hamilton, Jaringan Komputer**

I. PENDAHULUUAN

Jumlah celah pada sebuah jaringan komputer dapat diketahui dari tingkat penurunan kinerja jaringan tersebut saat terjadi gangguan yang tidak diharapkan seperti bencana alam, kerusakan komponen dasar, atau serangan siber (*Cyber Attack*) dari peretas tidak bertanggung jawab. Bencana alam seperti gempa bumi dapat merusak jalur aliran listrik penting sehingga menimbulkan kelumpuhan jaringan. Dari sudut pandang seorang peretas, Ia hanya perlu mengeksplotasi kelemahan sebuah jaringan dan kemudian menyasar simpul yang menjadi celah dari jaringan tersebut sehingga dapat mengakses simpul-simpul lain yang bertetangga dengan simpul tersebut. Sebagai contoh, Serangan yang ditujukan ke sebuah *Internet Service Provider (*ISP) seperti Telkom dapat mengakibatkan banyak website perusahaan dan layanan-layanan online *down* dan tidak dapat diakses. Untuk mengatasi kejadian-kejadian yang tidak diharapkan seperti yang telah disebutkan sebelumnya, diperlukan pemeliharaan jaringan komputer dan eksplorasi terhadap celah-celah dalam jaringan dengan mengidentifikasi simpul-simpul artikulasi pada jaringan dan struktur bagian dari jaringan yang membentuk lintasan siklik.



Gambar 1 Penghapusan simpul V8 dan V10 dapat melumpuhkan seluruh jaringan (Sumber : https://www.cise.ufl.edu/~mythai/files/ToN-CND.pdf Akses : 16 Mei 2017)

Contoh pada gambar 1 mengilustrasikan sebuah struktur jaringan internet dimana simpul v1,v2,...,v7 adalah simpul yang merepresentasikan sebuah *Internet Service* Provider (ISP) dan simpul sisanya adalah simpul yang mewakili simpul transmisi dan pengguna jasa internet. Serangan pada simpul v8 dan v10 sudah cukup untuk melumpuhkan seluruh jaringan sehingga simpul pengguna tidak dapat mengakses internet. Pada strategi lain, serangan pada simpul v7 atau simpul v14 tidak menimbulkan dampak merugikan pada jaringan karena semua pengguna masih dapat mengakses internet. Contoh ini menggambarkan bahwa untuk melumpuhkan jaringan secara keseluruhan, dibutuhkan kontrol keseimbangan antara komponen yang terputus dari jaringan dan lumpuhnya jaringan induk. Salah satu cara yang efektif dan mungkin dilakukan adalah dengan menghitung metriks *total pairwise connectivity*, i.e., jumlah pasangan simpul yang terhubung [1] dalam jaringan. Kembali ke contoh gambar 1, hancurnya simpul v8 dan v10, yang sebelumnya telah kita ketahui dapat melumpuhkan seluruh jaringan, mengindikasikan bahwa kedua simpul tersebut memang menurunkan metriks *total pairwise connectivity* ke jumlah terbesarnya yang dimungkinkan (65%). Hasil dari penurunan ini membuat keseluruhan jaringan komputer mengalami malfungsi atau lumpuh.

Pengukuran *total pairwise connectivity* dapat digunakan secara efektif di berbagai aplikasi jaringan secara praktis. Pada kebanyakan jaringan berukuran besar, Penghapusan simpul kritis dalam konteks metriks *total pairwise connectivity* tidak hanya dapat menurunkan tingkat performansi dari jaringan tetapi juga memutuskan hubungan jaringan tersebut dengan dunia luar. Aplikasi lain dari pengukuran metriks ini dapat ditemukan pada pelumpuhan jaringan komunikasi teroris. Dengan mengukur metriks *total pairwise connectivity,* pemutusan jaringan kumunikasi antar teroris dapat dilakukan dengan lebih efektif.

Suatu simpul dapat dikategorikan sebagai simpul kritis jika ia termasuk kedalam himpunan R dimana diberikan sebuah graf $G=(V,E)$, $|V|=n$ dan $|E|=m$ terdapat himpunan simpul R, $R \subseteq V$, $|R| \leq k$, yang jika dihapus dari graf menghasilkan graf residu $G \setminus R$ dengan metriks *Total Pairwise Connectivity* minimum.

Jaringan komputer yang menerapkan topologi *ring* juga rentan akan celah. Topologi *ring* di representasikan dengan sebuah graf berbentuk siklik atau siklus. Pada topologi *ring*, jika satu komputer dalam jaringan rusak, maka keseluruhan jaringan komputer tidak dapat berfungsi. Jika satu komputer berhasil

dikuasai oleh peretas, maka peretas tersebut dapat dengan mudahnya mengakses komputer lain dalam jaringan yang sama. Hal ini dikarenakan setiap simpul dalam topologi *ring* berperan dalam meneruskan pesan ke simpul yang bertetanggaan dengan simpul tersebut sehingga dari hanya satu komputer, peretas dapat meneruskan data ke komputer lain dengan mudah.

Suatu jaringan komputer yang besar secara tidak disadari dapat mengandung bagian yang menyerupai topologi *ring*. Dalam representasi graf, bagian dari jaringan yang berbentuk topologi *ring* dapat disamakan dengan siklus hamilton.

II. DASAR TEORI

A. *Algoritma DFS*

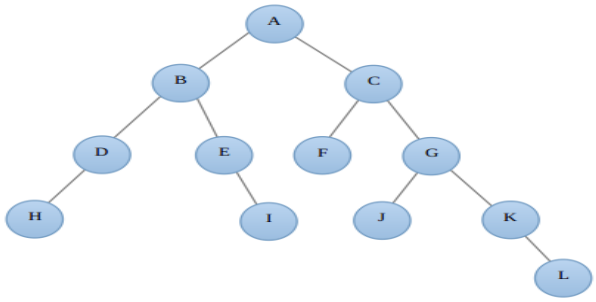
DFS (Depth-First-Search) adalah salah satu algoritma penelusuran struktur graf / pohon berdasarkan kedalaman. Simpul ditelusuri dari *root* kemudian ke salah satu simpul anaknya, dan terus melalui simpul anak pertama dari simpul anak pertama level sebelumnya hingga mencapai level terdalam.

Setelah sampai di level terdalam, penelusuran akan kembali ke satu level sebelumnya untuk menelusuri simpul anak kedua pada pohon biner (simpul sebelah kanan) lalu kembali ke langkah sebelumnya dengan menelusuri simpul anak pertama lagi sampai level terdalam dan seterusnya.

```
procedure DFS(input v:integer)
{Mengunjungi seluruh simpul graf dengan algoritma pencarian DFS

Masukan: v adalah simpul awal kunjungan
Keluaran: semua simpulyang dikunjungi ditulis ke layar
}
Deklarasi
   w : integer

Algoritma:
   write(v)
   dikunjungi[v]←true
   for w←1 to n do
      if A[v,w]=1 then {simpul v dan simpul w bertetangga }
         if not dikunjungi[w] then
            DFS(w)
         endif
      endif
   endfor
```



Gambar 2 Graf Penelusuran DFS
(Sumber:https://saungkode.wordpress.com/2014/04/16/penelusuran-pohon-biner-berdasarkan-kedalaman-dengan-algoritma-dfs-stack-dan-secara-melebar-level-order-dengan-algoritma-bfs-queue-dan-implementasinya-dalam-bahasa-c/ Akses : 16 Mei 2017)

Hasil Penelusuran : A-B-D-H-E-I-C-F-G-J-K-L

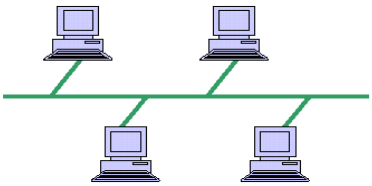
Sisi-sisi dalam pohon ruang status hasil dari penelusuran DFS dapat diklasifikasikan menjadi 4 jenis [2]. Jenis sebuah sisi (u,v) , sisi yang menghubungkan simpul u ke simpul v, bergantung kepada apakah pada saat penelusuran dilakukan simpul v sudah pernah dikunjungi sebelumnya. Jenis sisi (u,v) adalah :

1. *Tree Edge*
   Jika simpul v dikunjungi untuk pertama kalinya saat melakukan traversal terhadap sisi (u,v), maka sisi(u,v) termasuk jenis *Tree Edge*.
2. *Back Edge*
   Jika simpul v sudah pernah dikunjungi sebelumnya dan simpul v merupakan simpul orang tua dari simpul u, maka sisi (u,v) termasuk jenis *Back Edge*.
3. *Forward Edge*
   Jika simpul v sudah pernah dikunjungi sebelumnya dan simpul v merupkan simpul anak dari simpul u, maka simpul (u,v) termasuk jenis *Forward Edge*.
4. *Cross Edge*
   Jika simpul v sudah pernah dikunjungi sebelumnya dan simpul v bukan simpul orang tua maupun simpul anak dari simpul u, maka sisi (u,v) termasuk jenis *Cross Edge*.

B. *Topologi Jaringan Komputer*

Topologi jaringan adalah suatu cara untuk menghubungkan satu komputer dengan komputer lain sehinggga membentuk suatu jaringan [3]. Topologi jaringan juga dapat didefinisikan sebagai gambaran secara fisik dari pola hubungan antar komponen jaringan yang meliputi server, workstation, hub, dan pengkabelan. Topologi jaringan direpresentasikan dengan sebuah graf. Jenis topologi jaringan:

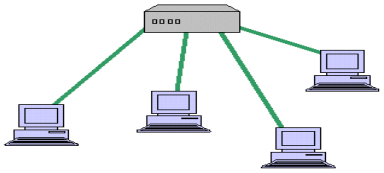
- Topologi BUS
  Topologi bus adalah jenis topologi jaringan komputer dimana setiap komputer terhubung secara linear dengan komputer lain tanpa menggunakan perangkat keras aktif seperti *hub* atau *switch*. Topologi ini menggunakan BUS sebagai kenektor antar komputer dalam satu jaringan dan menggunakan *terminator* pada tiap ujung jaringan untuk mencegah terjadinya kolisi data pada jaringan.



Gambar 3 Topologi BUS (Sumber : https://www.lifewire.com/computer-network-topology-illustrated-4064043 Akses : 17 Mei 2017)

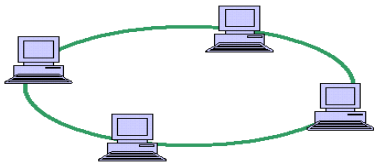
- Topologi Star
  Topologi star adalah jenis topologi jaringan dimana satu komputer dapat mengirimkan data ke semua komputer pada jaringan tersebut sehingga setiap komputer penerima data memiliki kecepatan transfer

dan mendapatkan data yang sama. Tiap komputer pada topologi star dihubungkan dengan *hub* atau *switch*. Topologi star juga memungkinkan setiap komputer memiliki kabelnya sendiri sehingga apabila terjadi kegagalan jaringan pada satu komputer, keseluruhan jaringan tidak akan terganggu. Topologi ini adalah topologi yang paling banyak digunakan saat ini.



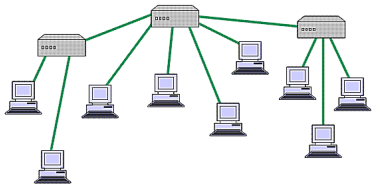
Gambar 4 Topologi Star (Sumber : https://www.lifewire.com/computer-network-topology-illustrated-4064043 Akses : 17 Mei 2017)

- Topologi Ring
  Topologi ring adalah jenis topologi jaringan dimana satu komputer terhunbung dengan dua komputer lain membentuk sebuah siklus. Setiap terjadi pengiriman data, data tersebut harus melalui beberapa komputer terlebih dahulu sebelum mencapai komputer tujuan. Topologi ini merupakan jenis topologi paling sederhana karena tidak membutuhkan alat tambahan seperti *hub*, *switch*, atau *terminator*. Jika terjadi kerusakan atau kegagalan pada satu simpul komputer, keseluruhan jaringan akan terganggu.



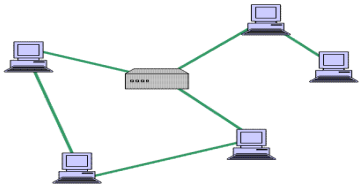
Gambar 5 Topologi Ring (Sumber : https://www.lifewire.com/computer-network-topology-illustrated-4064043 Akses : 17 Mei 2017)

- Topologi Tree
  Topologi tree adalah jenis topologi jaringan dimana satu komputer terhubung dengan komputer lain membentuk struktur pohon. Topologi ini menggunakan sistem hierarki pada proses transfer data. Dengan menggunakan topologi trree, dapat dimungkinkan untuk menghubungkan jaringan antar komputer dengan hierarki atau tingkatan berbeda.



Gambar 6 Topologi Tree (Sumber : https://www.lifewire.com/computer-network-topology-illustrated-4064043 Akses : 17 Mei 2017)

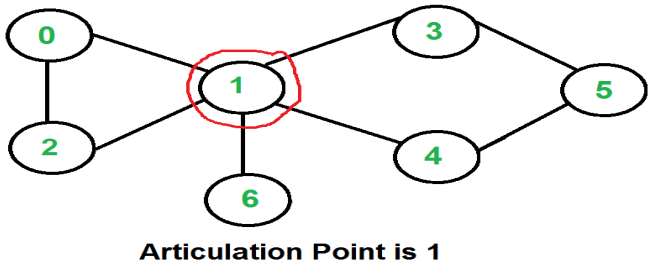
- Topologi Mesh
  Topologi mesh adalah jenis topologi jaringan dimana setiap komputer terhubung dengan setiap komputer lain membentuk struktur graf lengkap. Data pada satu komputer dalam jaringan dapat diakses dengan berbagai cara dari komputer mana saja. Topologi mesh tidak cocok diterapkan pada jaringan dengan jumlah komputer banyak.



Gambar 7 Topologi Mesh (Sumber : https://www.lifewire.com/computer-network-topology-illustrated-4064043 Akses : 17 Mei 2017)

C. *Simpul Artikulasi (Cut Vertex)*

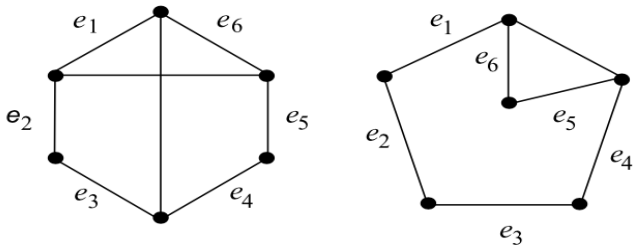
Sebuah simpul dalam graf tak berarah dikatakan sebagai simpul artikulasi jika penghapusan simpul tersebut dari graf menyebabkan graf tersebut terputus (*disconnect*) atau tak terhubung. Simpul artikulasi merepresentasikan celah dalam jaringan komputer. Jika terjadi kerusakan pada simpul artikulasi, jaringan akan terbagi menjadi 2 atau lebih komponen tak terhubung. Analisis terhadap simpul artikulasi seringkali dipakai untuk membangun jaringan komputer yang handal [9].



Gambar 8 Contoh Simpul Artikulasi pada Graf Tak Berarah (Sumber : http://www.geeksforgeeks.org/articulation-points-or-cut-vertices-in-a-graph/ Akses : 17 Mei 2017)

D. *Siklus Hamilton*

Sebuah siklus yang melalui semua simpul dalam sebuah graf disebut dengan siklus hamilton. Sebuah graf yang mengandung siklus hamilton disebut dengan graf hamilton. Jika simpul terakhir dari siklus hamilton dihapus, terbentuklah sebuah lintasan hamilton. Lintasan hamilton adalah lintasan yang melalui semua simpul dalam graf tepat sekali dan simpul ujung dari lintasan tidak sama dengan simpul awal. Sebuah graf non-Hamilton dapat memiliki lintasan hamilton tetapi tidak memiliki siklus hamilton.



Gambar 9 Contoh Graf Hamilton (Sumber : http://compalg.inf.elte.hu/~tony/Oktatas/TDK/FINAL/Chap%203.PDF Akses : 18 Mei 2017)

1. Theorema Dirac

   Jika sebuah graf memiliki n simpul dan setiap simpul dalam graf memiliki derajat minimal n/2, maka graf tersebut mengandung sebuah siklus hamilton [4].

2. Theorema Ore

   Jika sebuah graf sederhana dengan n simpul memiliki jumlah derajat dari 2 simpul sembarang yang tidak bertetanggaan lebih besar dari n, graf tersebut mengandung siklus hamilton [5].

III. Algoritma Pencarian Celah dan Optimasi Jaringan Komputer

Sebuah jaringan komputer yang handal adalah jaringan yang tidak memiliki celah. Celah dalam jaringan, yang direpresentasikan dengan graf, dapat berupa sebuah simpul artikulasi atau bagian dari graf (upagraf) yang berbentuk siklik.

Sebuah simpul dapat dikatakan sebagai simpul artikulasi jika penghapusan simpul tersebut dari graf menyebabkan graf tidak terhubung. Simpul artikulasi merepresentasikan celah dalam sebuah jaringan komputer. Jika sebuah komputer server yang merupakan simpul artikulasi dalam jaringan rusak atau berhasil dikuasai oleh peretas, maka keseluruhan jaringan tersebut akan lumpuh ibarat graf yang terpecah menjadi dua bagian terpisah atau dikuasai oleh peretas sehingga peretas tersebut dapat mengakses semua *resource* dalam jaringan komputer tersebut.

Upagraf berbentuk siklik dalam graf jaringan komputer meyerupai topologi jaringan jenis *ring*. Topologi *ring* memiliki kelemahan dimana jika satu simpul rusak atau dikuasai oleh peretas, secara otomatis semua bagian dari jaringan akan lumpuh atau jatuh ke tangan peretas. Hal ini dapat terjadi karena setiap komputer dalam topologi *ring* berperan dalam menyalurkan data dari satu komputer ke komputer lainnnya sehingga kerusakan atau penguasaan satu komputer dalam jaringan dapat berarti bahwa data tidak dapat disalurkan ke komputer lainnya atau peretas dapat menguasai semua *resource* yang ada di komputer dalam jaringan.

A. *Algoritma Pencarian Simpul Artikulasi*

Langkah-langkah pencarian simpul artikulasi dalam sebuah graf adalah:

1. Lakukan penelusuran terhadap graf dengan algoritma DFS dan berikan nomor kepada setiap simpul yang dibangkitkan pada pohon ruang status sesuai urutan pengunjungan.

2. Simpul akar pada pohon ruang status termasuk ke dalam simpul artikulasi jika simpul tersebut memiliki lebih dari satu anak

3. Simpul daun tidak termasuk ke dalam simpul artikulasi

4. Hitung fungsi Low(v) dimana v adalah sebuah simpul pada pohon ruang status hasil penelusuran DFS dan Low(v) menghasilkan nomor terkecil simpul yang dapat dicapai dari simpul v menggunakan 0 atau lebih *Tree Edge* dan maksimal 1 *Back Edge*.

5. Simpul v merupakan simpul artikulasi jika dan hanya jika simpul v bukan simpul akar maupun simpul daun, serta mempunyai simpul anak w, dimana Low(w) >= Nomor dari simpul v

```
time = 0
function DFS(input adj[][], disc[], low[], visited[], parent[], AP[], V:integer)
{Mencari simpul artikulasi dari sebuah graf dnegan algoritma DFS
Masukan: Graf yang diwakili sebuah matriks ketetanggaan adj
Keluaran: Larik AP berisi semua simpul artikulasi}
        visited[V]<-true
        disc[vertex]<-low[V]<-time+1
        child<-0
        for i<-0 to V
                if adj[V][i] = true
                        if visited[i] = false
                                child<-child + 1
                                parent[i]<-V
                                DFS(adj, disc, low, visited, parent, AP, i, n, time+1)
                                low[V]<-minimum(low[V], low[i])
                                if parent[V] = nil and child > 1
                                        AP[V]<-true
                                if parent[V] <> nil and low[i] >= disc[V]
                                        AP[V]<-true
                        else if parent[V] <> i
                                low[vertex]<-minimum(low[vertex], disc[i])
```

Algoritma pencarian simpul artikluasi memiliki kompleksitas waktu O(V+E).

B. *Algoritma Pendeteksian Siklus Hamilton*

Langkah-langkah untuk mendeteksi apakah graf mengandung siklus hamilton adalah:

1. Lakukan penelusuran terhadapap graf menggunakan algortima DFS.

2. Klasifikasikan setiap simpul pada pohon ruang status yang terbentuk.

3. Jika terdapat *back edge* pada pohon ruang status hasil penelusuran, maka graf mengandung siklus hamilton.

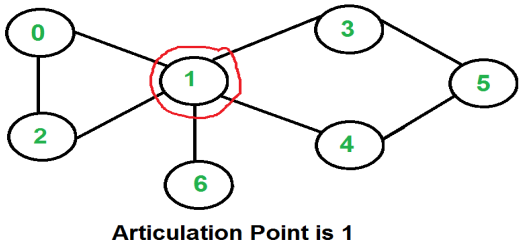
Algoritma pendeteksian siklus hamilton memiliki kompleksitas waktu O(V+E).

IV. Analisis Kasus

Kasus-kasus yang dianalisis adalah kasus jaringan komputer dengan topologi yang biasa ditemukan sehari-hari. Jaringan-jaringan yang memiliki topologi tidak umum cenderung rentan memiliki celah yang tidak disadari keberadaannya.

A. Simpul Artikulasi

1. Kasus Uji 1

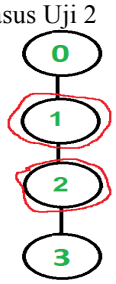


Gambar 10 Kasus Uji Simpul Artikulasi 1 (Sumber : http://www.geeksforgeeks.org/articulation-points-or-cut-vertices-in-a-graph/ Akses : 18 Mei 2017)

```
amas-Macbook-Pro:Tugas Makalah akmalfadlurohman$
Simpul Artikulasi ada pada simpul
1
```

Gambar 11 Hasil Uji Simpul Artikulasi 1 1 (Sumber : Dokumen Pribadi)

2. Kasus Uji 2

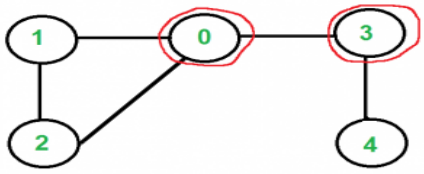


Gambar 12 Kasus Uji Simpul Artikulasi 2 (Sumber : http://www.geeksforgeeks.org/articulation-points-or-cut-vertices-in-a-graph/ Akses : 18 Mei 2017)

```
amas-Macbook-Pro:Tugas Makalah akmalfadlurohman$ ./AP
Simpul Artikulasi ada pada simpul
1 2
```

Gambar 13 Hasil Uji Simpul Artikulasi 2 1 (Sumber : Dokumen Pribadi)

3. Kasus Uji 3



Gambar 14 Kasus Uji Simpul Artikulasi 3 (Sumber : http://www.geeksforgeeks.org/articulation-points-or-cut-vertices-in-a-graph/ Akses : 18 Mei 2017)

```
amas-Macbook-Pro:Tugas Makalah akmalfadlurohman$ ./AP
Simpul Artikulasi ada pada simpul
0 3
```

Gambar 15 Hasil Uji Simpul Artikulasi 3 1 (Sumber : Dokumen Pribadi)

Source Code Pengujian Simpul Artikulasi

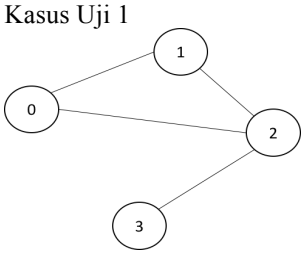
```
int main()
{
    Graph g1(5);
    g1.addEdge(1, 0);
    g1.addEdge(0, 2);
    g1.addEdge(2, 1);
    g1.addEdge(0, 3);
    g1.addEdge(3, 4);
    cout << "Simpul Artikulasi ada pada simpul \n";
    g1.AP();
    cout << endl ;

    Graph g2(4);
    g2.addEdge(0, 1);
    g2.addEdge(1, 2);
    g2.addEdge(2, 3);
    cout << "Simpul Artikulasi ada pada simpul \n";
    g2.AP();
    cout << endl ;

    Graph g3(7);
    g3.addEdge(0, 1);
    g3.addEdge(1, 2);
    g3.addEdge(2, 0);
    g3.addEdge(1, 3);
    g3.addEdge(1, 4);
    g3.addEdge(1, 6);
    g3.addEdge(3, 5);
    g3.addEdge(4, 5);
    cout << "Simpul Artikulasi ada pada simpul \n";
    g3.AP();
    cout << endl ;
    return 0;
}
```

B. Sirkuit Hamilton

1. Kasus Uji 1

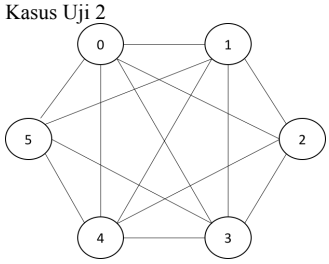


Gambar 16 Kasus Uji Siklus Hamilton 1 (Sumber : Dokumen Pribadi)

```
amas-Macbook-Pro:Tugas Makalah akmalfadlurohman$ ./Cycle
Graf mengandung siklus hamilton
```

Gambar 17 Hasil Uji Siklus Hamilton 1 1 (Sumber : Dokumen Pribadi)

2. Kasus Uji 2



Gambar 18 Kasus Uji Siklus Hamilton 2 1 (Sumber : Dokumen Pribadi)

```
amas-Macbook-Pro:Tugas Makalah akmalfadlurohman$ ./Cycle
Graf mengandung siklus hamilton
```

Gambar 19 Hasil Uji Siklus Hamilton 2 (Sumber : Dokumen Pribadi)

Source Code Pengujian Siklus Hamilton

```
int main()
{
    Graph g1(4);
    g1.addEdge(0, 1);
    g1.addEdge(0, 2);
    g1.addEdge(1, 2);
    g1.addEdge(2, 3);
    g1.isCyclic()? cout << "Graf mengandung
siklus hamilton\n":
    cout << "Graf tidak mengandung siklus
hamilton\n";

    Graph g2(6);
    g2.addEdge(0, 1);
    g2.addEdge(0, 2);
    g2.addEdge(0, 3);
    g2.addEdge(0, 4);
    g2.addEdge(0, 5);
    g2.addEdge(1, 2);
    g2.addEdge(1, 3);
    g2.addEdge(1, 4);
    g2.addEdge(1, 5);
    g2.addEdge(2, 3);
    g2.addEdge(2, 4);
    g2.addEdge(3, 4);
    g2.addEdge(3, 5);
    g2.addEdge(4, 5);
    g2.isCyclic()? cout << "Graf mengandung
siklus hamilton\n":
    cout << "Graf tidak mengandung siklus
hamilton\n";

    return 0;
}
```

V. KESIMPULAN

Sebuah jaringan komputer yang direpresentasikan dengan sebuah graf dapat memiliki celah berupa simpul artikulasi atau upagraf berbentuk siklus. Pencarian simpul artikulasi dapat dilakukan melalui aplikasi algoritma DFS dengan kompleksitas waktu O(V+E). Pendeteksian siklus hamilton dalam graf juga dapat dilakukan melalui aplikasi algoritma DFS dengan komleksitas waktu O(V+E).

Simpul artikulasi pada jaringan komputer yang telah berhasil diidentifikasi sebaiknya ditingkatkan keamanan dan perlindungannya. Hal ini dapat dilakukan dengan menggunakan *firewall*, pemeriksaan rutin terhadap virus komputer, dan penggantian kata sandi akses komputer secara berkala.

Subbagian dari jaringan komputer berupa siklus hamilton sebaiknya diuraikan atau disederhanakan agar tidak membentuk siklus. Hal ini dapat dilakukan dengan penggunaan alat bantuan seperti *hub* atau switch sehingga membentuk topologi *star*.

References

[1] T. Dinh, Y. Xuan,M. Thai,P. Pardalos,and T. Znati. On new approaches of assessing network vulnerability: Hardness and approximation. *Networking, IEEE/ACM Transactions on*, 20(2):609-619, april 2012.

[2] https://courses.csail.mit.edu/6.006/fall11/rec/rec14.pdf

[3] http://www.tutorialcarakomputer.com/2013/12/pengertian-dan-jenis-jenis-topologi-jaringan.html

[4] https://en.wikipedia.org/wiki/Hamiltonian_path#Bondy.E2.80.93Chv.C3.A1tal_theorem

[5] https://en.wikipedia.org/wiki/Ore%27s_theorem

[6] http://www.geeksforgeeks.org/articulation-points-or-cut-vertices-in-a-graph/

[7] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4963334/.

[8] http://www.eecs.wsu.edu/~holder/courses/CptS223/spr08/slides/graphapps.pdf

[9] https://courses.cs.washington.edu/courses/cse421/04su/slides/artic.pdf

[10] https://www.cise.ufl.edu/~mythai/files/ToN-CND.pdf

[11] http://www.geeksforgeeks.org/detect-cycle-undirected-graph/

[12] http://compalg.inf.elte.hu/~tony/Oktatas/TDK/FINAL/Chap%203.PDF.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 18 Mei 2017

Akmal Fadlurohman 13515074