APLIKASI TEORI GRAF DALAM JARINGAN (NETWORKING)

Nama : Glen Chrisitan
NIM : 13505098
Sekolah Teknik Elektro dan Informatika
Program Studi Teknik Informatika
Institut Teknologi Bandung
Jalan Ganesha 10, Bandung

Abstrak

Makalah ini akan membahas mengenai aplikasi graf dalam jaringan *(networking)*. Seperti yang kita tahu graf adalah salah satu materi dari matematika diskrit yang sangat berguna dalam kehidupan sehari-hari. Mungkin anda pernah mendengar pertanyaan "manakah jalur yang paling cepat untuk mencapai kota A?" atau "Apakah bisa, seseorang melalui semua jembatan tepat sekali dan kembali lagi ke posisi semula?". Ini adalah masalah-masalah dalam dunia nyata yang bisa diselesaikan dengan graf.

Tujuan dari pembuatan makalah ini adalah: anda akan mampu untuk memodelkan jaringan sebagai graf berarah, anda akan mengetahui bahwa jaringan telepon dan jaringan internet sangat berbeda, dan bagaimana *network topology* dapat mempengaruhi algoritma routing.

Pada makalah ini, secara spesifik akan dibahas penggunaan teori graf dalam networking. Sebenarnya, mengapa graf sangat dibutuhkan dalam networking? Jawabannya adalah karena kecepatan sangat dibutuhkan dalam komunikasi. GRAF memberi kita solusi untuk mencapai suatu optimasi kecepatan sehingga memenuhi kebutuhan semua orang. Dan juga kebutuhan akan jaringan yang *robust* dan *reliably* sehingga membuat user nyaman.

Jaringan yang akan dibahas dalam makalah ini meliputi dua bagian meliputi *telephone network* dan *internet network*. Kedua jaringan ini sangat berbeda. Dalam makalah ini pembuat berasumsi bahwa pembaca mempunyai pengetahuan dasar tentang graf definisi sisi, verteks dan mengetahui bagaimana graf dapat direpresentasikan.

1. Pendahuluan

Komunikasi adalah hal terpenting yang diperlukan oleh manusia. Seperti yang kita tahu, manusia adalah makhluk sosial. Manusia tidak bisa hidup di dunia ini tanpa manusia lain. Cara berhubungan dengan orang lain itu kita namakan komunikasi. Makalah ini lebih menekankan kepada komunikasi yang memakai media telepon dan internet. Dimulai dari definisi yang berkaitan dengan *Directed Graphs* seperti:

1. The *out-degree* of a vertex u is the number of edges (u, v) for all $v \in V(G)$.
2. The *in-degree* of a vertex u is the number of edges (v, u) for all $v \in V(G)$.
3. Recall that in undirected graphs, a *walk* from vertex v to a vertex u is an alternating finite sequence of vertices and edges $v_0 e_0 v_1 e_1 \ldots v_{k-1} e_{k-1} v_k$ such that $\forall i \in [1, k-1]$ either $e_i = (v_{i-1}, v_i) \in E$ or

$$e_i = (v_i, v_i - 1) \in E$$

.

4. In directed graphs, a *directed walk* from vertex v to a vertex u is an alternating finite sequence of vertices and edges $v_0 e_0 v_1 e_1 \dots v_{k-1} e_{k-1} v_k$ such that $\forall i \in [1, k-1], e_i = (v_{i-1}, v_i) \in E$

.

5. A *directed path* is a directed walk which does not repeat any vertex.
6. A *directed cycle* is a directed walk which does not repeat any vertex except for the first and the last.

A *directed path* adalah perjalanann yang tidak melewati sebuah verteks sebanyak dua kali.
A *directed cycle* adalah perjalanan berarah yang tidak melewati verteks dua kali kecuali verteks pertama dan terakhir.

2. Graf Berarah

Banyak aplikasi dari teori graf yang membutuhkan definisi dari graf yang sisinya berarah yang memperbolehkan adanya sisi dari u ke v tetapi tidak harus selalu sisi dari v to u.

Ini adalah definisi formal dari graf:

A *directed graph* (often called a *digraph* for brevity's sake) G is a pair (V, E) where V is a set of elements called vertices and E is a subset of the set of all ordered pairs (u, v), where u and v are vertices. (An element of E is called an edge of G).

Example

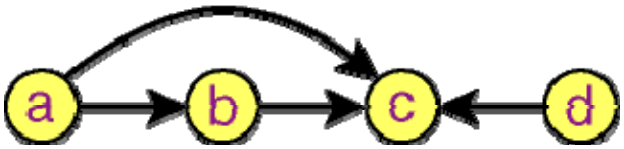

Figure 2:

A directed graph with $V = \{a, b, c, d\}$ and $E = \{(a, b), (a, c), (b, c), (d, c)\}$ can be drawn as shown here.

3. Bobot Graf

Langkah selanjutnya adalah menarik keverteksan bahwa pengertian dari graf adalah untuk mempertimbangkan graf berbobot, yang tiap sisinya diasosiasikan dengan bilangan, disebut berat (harga atau panjang) dari sisi. Bobot dari sisi bisa negative dan positif.

Keverteksan umum ini mendorong kita untuk mengubah definisi kita tentang panjang perjalanan, yang biasa kita sebut bobot perjalanan, yang bertujuan membedakan dari jumlah dari sisi dalam perjalanan.

Ini adalah definisi formal dari bobot perjalanan:

Given a weighted directed graph $G(V, E)$ with a weight function $w : E \to R$, the weight $w(P)$ of a walk

$$P = (v_0, e_0, v_1, e_1, \ldots, v_{k-1}, e_{k-1}, v_k)$$

Equation 1

is defined as the sum

$$w(P) = \sum_{i=0}^{k-1} w(e_i).$$

Equation 2

The distance $\delta(u, v)$ from u to v is defined by

$$\delta(u, v) = \infty$$

Equation 3

if no directed walk from u to v exists $w(P) P$ is a directed walk from u to v

4. Algoritma Graf

Algoritma dalam graf sangat dibutuhkan dalam mencari jarak terpendek (shortest path). Variasi dari problem tersebut adalah sebagai berikut:

Single Pair

Input

A directed weighted graph $G(V, E)$ and vertices $u, v \in V$

Output

A shortest (u, v)-path and its weight.

Single Source

Input

A directed weighted graph $G(V, E)$ and vertex $u \in V$.

Output

A shortest (u, v)-path and its weight for all $v \in V$.

All Pairs

Input

A directed weighted graph $G(V, E)$.

Output

For every pair (u, v) of vertices $\in V$ a shortest (u, v)-path and its weight.

4.1.Algoritma Dijkstra

Algoritma Djikstra dapat menyelesaikan mencari jarak terpendek dalam *single-source problem* pada graf berbobot (berarah atau tidak berarah), jika bobot dari sisi adalah positif.

Algoritma Djikstra mempertahankan sejumlah verteks –verteks terpendek yang sudah dijumlahkan, bersama *lementary set of vertices* yang bobot sisinya belum ditetapkan. Algoritma ini secara berulang memilih verteks. Verteks dengan bobot minimal diantara verteks yang lain yang bobotnya belum berakhir. Algoritma ini meng-update bobot yang diperkirakan untuk semua verteks yang bertetangga (update ini umumnya disebut "relaxation" dari sisi diantara

verteks ini). Kemudian verteks ini dijumlahkan dengan verteks-verteks sebelumnya yang sudah dikalkulasi yang memiliki bobot terpendek.

Algoritma ini berlanjut sampai semua verteks terakhir yang memiliki lintasan terpedek sudah dikalkulasi (until the set of vertices whose shortest-path weights have not been determined is empty).

Algoritma Djikstra adalah:

1. S is the set of vertices v for which the distance from s to v has been computed.
2. Q is the set of vertices v for which the distance from s to v has **not** yet been computed.
3. $\delta(s,v)$ is the current estimate of the weight from vertex s to vertex v.
4. $\pi(v)$ is a predecessor of $v \in V(G)$. In other words, the vertex through which the
5. shortest distance from s to v was established.

$$\delta(s,s)=0; \forall v \neq s, \delta(s,v)=\infty; Q=V; S=s; \textbf{while}\ (Q \neq 0)\{\text{ Find } u \in Q \text{ with the smallest } \delta(s,u); S=S \cup u; Q=Q-u; \text{for every } v \in \mathrm{adj}[u] \text{ if } (\delta(s,v)>\delta(s,u)+w(u,v))\{\delta(s,v)=\delta(s,u)+w(u,v); \pi(v)=u;\}\}$$

4.2.Algoritma Bellman-Ford

Algoritma Bellman-Ford dapat menyelesaikan problem single-path. Algoritma ini memperbolehkan sisi berbobot negative, tapi tidak memperbolehkan graf berarah bersiklus yang mempunyai bobot negative.

Algoritma Bellman-Ford menghasilkan nilai false, yang menjawab bahwa tidak ada solusi yang memenuhi, yaitu jika menemukan *cycle of negative weight yang* bisa dicapai dari *source.* Sebaliknya algoritma Bellman-Ford mengembalikan true jika sudah menemukan semua perjalanan terpendek dari *source.*

Algoritma Bellman-Ford adalah:

1. $\delta(s,v)$ is the current estimate of the weight from vertex s to vertex v.
2. $\pi(v)$ is a predecessor of $v \in V(G)$. In other words, the vertex through which the shortest distance from s to v was established.

$$\text{for each vertex } v \in V(G)\{\delta(s,v)=\infty \pi(v)=\mathrm{nil};\} \delta(s,s)=0; \text{for} (k=1; k<n; k++) \text{ for each edge}$$

$(u,v) \in E(G)_{\text{if}}(\delta(s,v) > \delta(s,u) + w(u,v))\{$
$\delta(s,v) = \delta(s,u) + w(u,v); \pi(v) = u;\}_{\text{for each edge}}$
$(u,v) \in E(G)_{\text{if}}(\delta(s,v) > \delta(s,u) + w(u,v))_{\text{return }}$ **false**;
return **true**;

5. Routing

5.1. Overview

Jaringan elektronik dikembangkan untuk menciptakan informasi dan mendistribusikannya. Jaringan yang paling penting sekarang ini adalah jaringan telepon dan jaringan internet.

Jaringan telepon yang saling berhubungan dengan telepon lain lebih banyak dari satu trilyn telepon di seluruh dunia. Di Negara Amerika sendiri, perusahaan telepon yang paling besar, yaitu AT dan T, membawahi kira-kira 200 milyar hubungan telepon setiap harinya. Selain telepon suara, ada juga video, facsimile, dan telemetry data.

Setiap harinya, *World Wide Web* bertumbuh kira-kira semilyar halaman elektronik, ditambahkan ke semilyar sebelumnya yang sudah digunakan. Banyak informasi yang mengejutkan ini dengan bebas disimpan oleh lebih dari setrilyun koneksi yang disebut hyperlinks

Proses dalam menemukan jalan dari suatu source menuju semua tujuan atau destinasi di dalam jaringan adalah disebut routing. Routing dicapai dengan cara mencari *routing protocols* yang konsisten satu sama lain dari routing *tables* dari setiap *router*, atau menukarkan *controller*.

5.2. Designing A Routing Protocol

Protocol routing mengkomunikasikan *global topological*, informasi untuk setiap router, mengijinkannya untuk mengambil keputusan lokal.

Karena perubahan yang sangat sering dan sekali-kali kegagalan dari elemen jaringan, *routing protocol* secara tidak bersamaan meng-update routing pada tiap router atau menukarkan controller. Updates selesai setelah melalui pertukaran secara periodik routing table dan informasi antar router. Ini meyakinkan bahwa mereka mempunyai pengamatan yang konsisten terhadap *network's topology*.

Syarat-syarat utama dari *routing protocol* adalah:

1. **Menjamin bahwa daftar dari router yang berbeda adalah konsisten.** Daftar routing harus konsisten supay rute dapat ditemukan via konkatenasi dari keputusan local (local decision).
2. **Memperkecil ukuran dari daftar router.** Ukuran dari daftar router mempengaruhi harga dan efisiensi dari router. Diharapkan ukuran dari daftar routing akan tumbuh secara perlahan daripada ukuran dari jaringan.
3. **Memperkecil kemunculan control messages.** Routing protocols membutuhkan pemakaian *control messages*. Control messages yang sekilas adalah atas dari *routing operations*, dan harus diminimalisasi (tetapi fungsi dari control message tidak berkurang).
4. **Kekuatan(Robustness).** Kesalahan atau *misrouting messages*, tidak mencapai tujuannya (memasuki *black hole*), atau mengakibatkan *loops* dan *oscillations*, harus dilakukan seminimal mungkin.

5.3. Routing in Telephone Networks

Jaringan telepon nasional terdiri dari struktur yang mempunyai hierarki tiga level:

1. Pelanggan atau Subscribers (telepon, modem, dan mesin faks)
2. Kantor pusat atau Central Offices
3. Kantor yang berjarak jauh atau Long Distance Offices (toll switching offices)

Pertukaran *local carrier* (contohnya Bell Atlantic), memfasilitasi telepon local dan me-manage sejumlah kantor pusat yang saling berhubungan satu sama lain dengan *one-hop links*.

6. Telephone Network

6.1. Overview

The Telephone Network

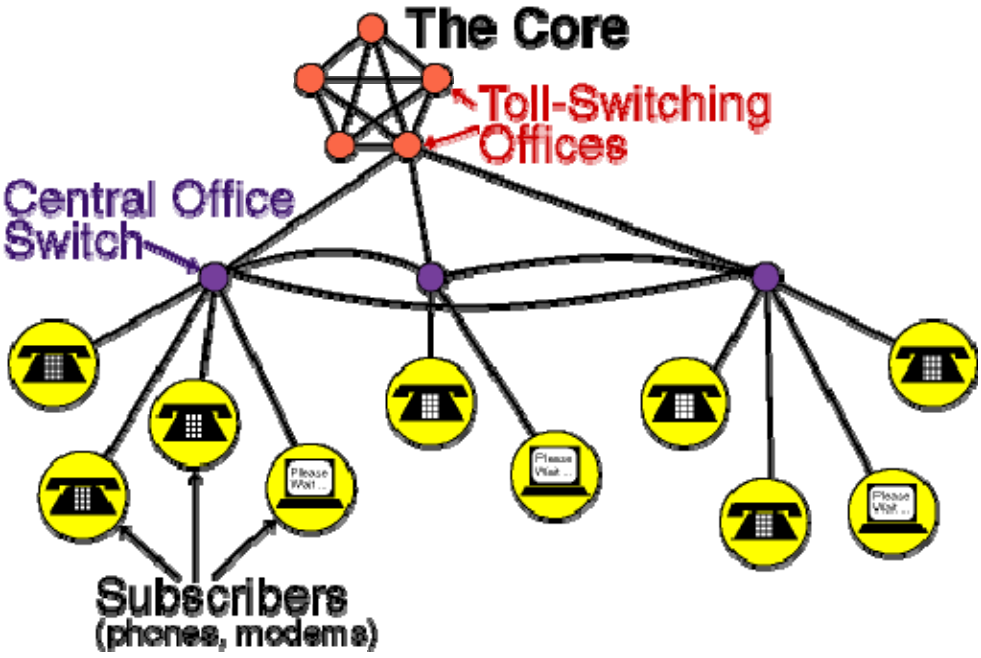

Figure 10:

Subscribers are served by central offices, which are in turn linked at the core at toll-switching offices.

Algoritma Routing yang digunakan:

1. Apabila suatu sumber telepon dan tujuannya dalam batas-batas central office, maka telepon akan langsung dihubungkan.
2. Apabila hubungan telepon berada diantara central offices dan dareah lokal, maka gunakan *one-hop path* diantara central offices.
3. Selain itu, kirim hubungan telepon ke appropriate core (atau cores).

Hanya satu keputusan besar pada seleksi dari jarak antara *two toll switches* untuk melakukan panggilan jarak jauh (long-distance call) adalah

memakai one-hop path dan two-hop path. (Jarak yang lebih panjang dari two hops biasanya tidak dipertimbangkan lagi.

6.2. Routing Strategies

Problem dari sebuah routing muncul pada saat lalu lintas telepon menjadi penuh.

Protocol yang dapat meng-handel jaringan yang memuat terlalu banyak routing:

1. Mengumpulkan informasi tentang lalu lintas telepon tersebut pada semua penghubung, menghitung jalur yang paling kosong, dan menentukan perjalanan melalui jalur tersebut.
2. Berdasarkan kepada strategy routing yang sebelumya dikumpulkan, yaitu statistik tentang pola dari semua penghubung dan periode waktunya (Statistik ini seharusnya diupdate secara periodik).

Strategy #2 adalah strategi yang diterima secara universal pada jaringan telepon yang nyata. Ini menyebabkan kumpulan dari hubungan telepon sangat dapat diprediksi, dan pengubung telepon dan pernukarnya sangat dapat diandalkan. Oleh karena itu, perkiraan beban dapat sebelumnya dihitung untuk setiap pasangan dari pertukaran untuk setiap interval tiap harinya, mengijinkan rute untuk meneteapkannya terlebih dahulu.

6.3. Costs

Penyederhanaan dari protocol hubungan telepon mengakibatkan:

1. Jikalau pertukaran dari pusat rusak, protocol routing tidak tahu bagaimana untuk menemukan jalur algernatif. Lebih jauh, untuk membuat keseluruhan system dapat diandalkan, setiap pertukaran harus dapat diandalkan.
2. Dibutuhkan setiap pasangan pertukaran terhubung oleh one-hop logical trunk yang dibuat untuk jaringan yang sangat mahal untuk dibuat dan untuk dipertahankan.

Sebuah alternatif, akan dikembangkan sebuah jaringan dimana algoritma routing yang sangat rumit dapat memperkecil kebutuhan konektivitas dengan cara memperbolehkan pengeluaran hubungan langsung.

6.4. DNHR

Telephone Networks: Dynamic Non-Hierarchical Routing (DNHR)

Kemajuan dari routing telepon adalah pengembangan dari *dynamic non-hierarchical routing (DNHR)*. DNHR menggunakan prediksi statistic dari kumpulan lalu lintas telepon dan fakta yaitu switches dan links biasanya bisa digunakan untuk memilih two-hop paths pada saat jalur yang paling pendek, shortest-one-hop-path terhalang.

DNHR membagi hari menjadi beberapa period (biasanya 10 atau lebih), dan untuk setiap period, tiap toll switch diberi tugas yaitu:

1. A **primary** one-hop path untuk toll switch yang lain dan
2. An ordered set untuk alternative, yaitu two-hop paths.

The Routing Algorithm

1. Panggilan yang masuk diteruskan ke jalur utama.
2. Jika sumber sudah digunakan dan juga tidak bisa didelegasikan ke panggilan tujuan (situasi ini dinamakan *spilling* atau *overflow*), akan dicoba alternative two-hop path sesuai dengan susunan yang telah dihitung sebelumnya pada periode ini (istilahnya adalah *crank-back*).
3. Jika jalur alternatifnya sibk, maka panggilan ditolak

> ***Perhatian!***
>
> Performansi dari DNHR mungkin terjadi bila lalu lintas berubah dengan tiba-tiba, jadi daftar dari jalur alternatif mungkin berubah. Perubahan secara tiba-tiba lalu lintas, mungkin menambah beban dari sejumlah panggilan, pada saat meninggalkan panggilan lain *underutilized*.

7. The Internet

7.1. Overview

Kita sudah melihat bahwa jaringan telepon diketahui untuk prediksi dari lalu lintas dan core yang kecil dari jaringan tersebut.

Keadaan sekeliling atau lingkungan dari Internet sangat berbeda:

1. Links dan routers tidak dapat dipercaya.
2. Jalur alternatif sulit didapat.
3. Pola dari lalu lintas di internet berubah dengan tidak terprediksi dalam hitungan menit.

Routing strategies pada lingkungan yang yang memiliki karakteristik di atas, harus memiliki banyak toleransi pada perubahan pada jaringan daripada strategi untuk jaringan telepon. Dua dasar dari routing algorithms pada jaringan pertukaran paket (Internet dan jaringan ATM) adalah:

1. Distance-Vector (berdasarkan pada Bellman-Ford Algorithm).
2. Link-State (berdasarkan pada Djikstra's Algorithm).

7.2. Distance Vector Routing

Distance-vector routing mempunyai asumsi sebagai berikut:

> Setiap router tahu identitas dari setiap router lain di dalam jaringan.

Pada saat mencoba untuk memaksakan asumsi ini, setiap router akan mempertahankan *distance vector*, yang terdiri dari list of tuples yaitu:

```
< destination, cost >
```

…untuk semua node pada jaringan. Disini, cost adalah **perkiraan** dari jalur terpendek dari sebuah router ke **tujuannya**; perkiraan ini tidak pernah lebih kecil daripada panjarng jalur yang nyata.

Distance-vector algorithm adalah implementasi yang didistribusikan dari Bellman-Ford Algorithm:

1. Setiap router secara periodik mengirim copy dari distance vector dirinya ke tetangganya.
2. Pada saat router menerima distance-vector dari tetangga, dia memutuskan apabila jarak untuk mencapai suatu destinasi akan menurun jika paket ke

destinasi tersebut akan dikirim melalui tetangganya tersebut.
3. Jika menurun, router tersebut meng-update distance-vector dirinya sendiri.

7.3. Link State Routing

Pada distance-vector routing, informasi tentang topology dari sebuah jaringan didistribusikan diantara routers, yang akan mencapai efisiensi tetapi juga mengakibatkan masalah.

Pendekatan secara alternative, digunakan dalam *link-state routing*, adalah untuk memberi tiap router informasi yang menyeluruh tentang graf dari sebuah jaringan. Setelah itu, setiap router secara independen menghitung jalur yang optimal ke semua destinasi atau tujuan.

Jadi, pertanyaan yang harus dijawab untuk mencapai pendekatan ini adalah:

1. Bagaimana pengetahuan tentang topology sebuah jaringan dapat didistribusikan ke semua router dalam jaringan tersebut?
2. Setelah topology dari sebuah jaringan sudah disebarkan, bagaimana jalur yang terpendek dapat dihitung?

7.3.1.Topology Dissemination

Pada link-state algorithm, secara periodic mendistribusikan *link-state packets* (*LSP*s), yang dikembangkan pada setiap router yang berpartisipasi di sebuah jaringan. Isi dari sebuah kontent yang terdiri dari LSP termasuk:

1. ID Router.
2. ID dari satu tetangga dari sebuah router.
3. Panjang lintasan dari link ke tetangga tersebut.

(Oleh karena itu setiap router akan mengirim sejumlah LSP sesuai dengan jumlah tetangga yang router tersebut punyai.)

Pada saat sebuah router **menerima** LSP, router akan mencoba untuk menyimpannya pada *LSP database* miliknya. Jika informasi tersebut berisi data yang sudah ada pada databasenya, tidak ada aksi yang akan dilakukan. Jika tidak, router menyimpan data tersebut dalam databasenya dan meneruskan copy dari LSP yang baru dia terima ke setiap router tetangga (kecuali kepada router yang telah mengirim paketnya)

Pada saat sebuah router dalam sebuah jaringan mempunyai LSP database yang konsisten, tiap router secara individual menghiutng jalur yang optimal ke semua node jaringan. Algoritma yang biasa digunakan untuk melakukan hal ini adalah algoritma Djikstra.

7.4. What's Next

Dalam dunia nyata, *intra-domain Internet routing* memakai salah satu link-state (untuk larger domains) atau distance vector (untuk smaller domains) untuk pendekatan. Inter-domain routing termasuk persoalan seperti *policy routing* (eg: larangan di perjalanan yang bisa dipakain oleh beberapa *subsets of nodes*, yang dibayar untuk digunakan, dan kualitas dari pelayanan yang mereka dapat, dll.), yang membuat tidak sah asumsi yang berkaitan dengan sistem metrik pada distance-vector dan link-state routing. Ini adalah problem yang sulit ditangani oleh peneliti sekarang ini.

Topology design adalah bidang lain dari jaringan yang sebenarnya berhubungan dengan teori graf secara umum. Seperti masalah sebagai berikut: spesifikasi dari konektivitas, harga jaringan, stabilitas routing dan flexibility, reliability, dan availability. Topologi suatu jaringan sangat bergantung pada parameter yang diberikan yang dimodelkan dan dianalisa pada jangka waktu dari teori graf.

DAFTAR PUSTAKA

RensselaerPolytechnic Institute. *(1999). Graph Theory Networking*

http://java.math.rpi.edu/devmodules/graph_networking

Tanggal akses: 23 Desember 2006 pukul 22:30.