Kompleksitas Algoritma Prim Untuk Pencarian Minimum Spanning Tree Pada Kasus Peer to Peer File Sharing

Prafajar Rizkyanno Multazam - 23514097
Program MagisterInformatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
23514097@std.stei.itb.ac.id

***Abstract*—Peer to Peer file sharing adalah salah satu cara distribusi data pada jaringan internet yang dapat mengurangi beban kerja server. Ide dasarnya adalah setiap PC (Personal Computer) dapat menjadi klien dan server dalam satu waktu. Implementasi dari *peer to peer file sharing* yang berupa program adalah program torrent, program ini melakukan pencarian file yang terdapat di PC yang terkoneksi dengan jaringan. Kita dapat anggap PC yang ada di jaringan representasinya adalah berupa sebuah *graph*. Pada paper ini dibahas tentang kompleksitas algoritma prim untuk pencarian *minimum spanning tree* pada *graph* untuk mengetahui pengaruh jumlah PC dengan lama waktu *file* dapat didistribusikan melalui jaringan internet.**

***Index Terms*—Peer to Peer, torrent, *graph*, *minimum spanning tree*.**

I. PENDAHULUAN

Peer to peer file sharing adalah sebuah cara untuk melakukan distribusi data pada jaringan internet. PC (Personal Computer) dalam satu waktu dapat menjadi klien sekaligus juga menjadi server. Salah satu keunggulan dari teknologi ini adalah mengurangi kinerja server pusat yang menjadi *data center*.

Implementasi yang berupa perangkat lunak dari ide *peer to peer file sharing* adalah torrent. Torrent adalah perangkat lunak yang menyimpan metadata sebuah file. Metadata file ini berisi file dan folder yang didistribusikan. Apabila kita ingin mengunduh sebuah file, maka perangkat lunak ini akan mencari PC mana yang mempunyai file tersebut melalui jaringan internet. Sedangkan, apabila kita ingin berbagi sebuah file (katakanlah menjadi server), maka perangkat lunak ini akan melakukan konfigurasi agar file dapat diunduh oleh pengguna torrent yang melakukan *request* untuk mengunduh sebuah file.

Kita dapat anggap representasi PC dan jaringan internet adalah berupa sebuah *graph*. Misalkan kita ingin mengunduh sebuah file, maka perangkat lunak torrent akan mencari PC mana yang mempunyai file tersebut dan membuat jalur koneksi yang sependek mungkin dengan PC tersebut agar proses unduh cepat. Salah satu metode yang dibahas pada makalah ini untuk kasus pencarian pada *graph* adalah *minimum spanning tree*.

Minimum spanning tree adalah metode yang menghasilkan *sub-graph* yang mencapai seluruh *vertex* dengan jalur seefektif mungkin dengan biaya serendah mungkin. Pada makalah ini dibahas kompleksitas algoritma prim untuk pencarian *minimum spanning tree*, dengan contoh kasus *peer to peer file sharing* untuk melihat pengaruh jumlah PC dan koneksi antar PC pada kecepatan distribusi data baik untuk mengunduh maupun utnuk berbagi file.

II. LANDASAN TEORI

A. Peer to Peer File Sharing

Peer to Peer file sharing adalah sebuah metode untuk melakukan distribusi dan berbagi-pakai konten media digital. Pengguna dapat mengakses file multimedia seperti buku, musik, film, maupun game menggunakan perangkat lunak *Peer to Peer*. Perangkat lunak ini mencari komputer lain yang terkoneksi dengan jaringan dan *Peer to Peer Network* untuk mendapatkan lokasi komputer dari konten yang diinginkan, setelah didapat lokasi kemudian perangkat lunak akan membuat jalur koneksi untuk transfer data.

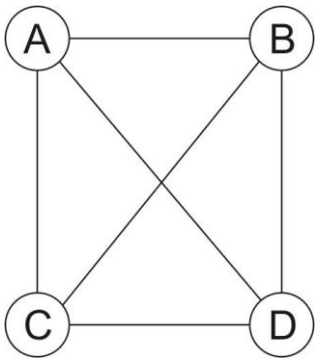
Teknologi *Peer to Peer file sharing* berkembang sejalan dengan meluas dan makin banyaknya kebutuhan akan fasilistas distribusi data. Faktor lain yang mendukung berkembangnya teknologi ini adalah meningkatnya *bandwith* internet, meluasnya digitalisasi dari media fisik, juga meningkatnya kemampuan dan performansi komputer personal. Pengguna dapat mendistribusikan satu atau lebih file juga dapat mengunduh satu atau lebih file melalui jaringan internet.

Kelebihan dari teknologi *peer to peer file sharing* adalah mengurangi beban kerja *data center*. Karena dengan menggunakan teknologi ini sebuah komputer dalam satu waktu dapat menjadi klien dan sekaligus server, sehingga jalur distribusi data tidak terpusat, melainkan tersebar.

Tentu saja ada kekurangan dari teknologi ini, yaitu pembajakan semakin marak, karena begitu cepat dan luasnya distribusi data sehingga dapat dikatakan hampir tidak mungkin untuk mengontrol dan memonitor seluruh aktifitas distribusi data.

B. Graph

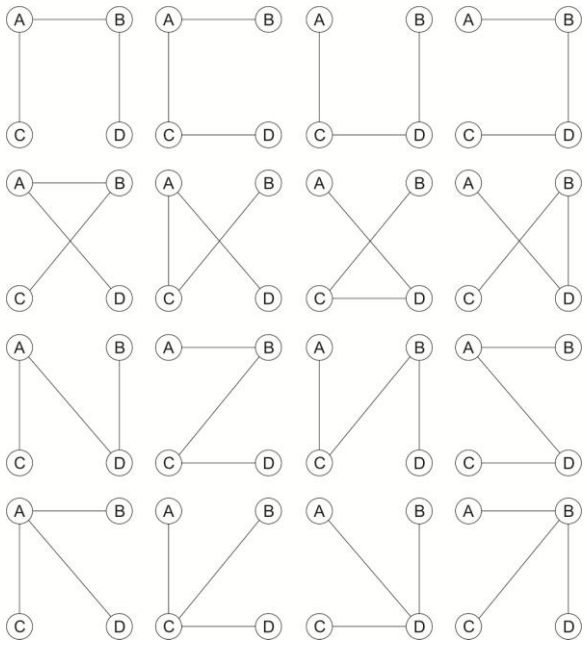
Graph adalah kumpulan dari *vertex* dan *edge* yang sedemikian hingga saling terhubung. *Vertex* adalah simpul yang dalam kasus *peer to peer file sharing* representasinya adalah berupa PC. *Edge* adalah sisi yang menghubungkan *vertex* satu dengan *vertex* lain. Pada makalah ini digunakan *graph* yang memenuhi spesifikasi *undirected weighted graph* untuk representasi PC dan koneksi antar PC dalam suatu jaringan komputer.



Gambar 2.1 *Undirected weighted graph* dengan 4 buah *verte*x.

Undirected weighted graph adalah *graph* tidak berarah yang pada setiap *edge* nya mempunyai *cost*. Maksud dari tidak berarah adalah misalkan *vertex* A bertetangga dengan *vertex* B, *cost* dari *vertex* A menuju *vertex* B adalah sama dengan *cost vertex* B menuju *vertex* A. Pada *directed weighted graph*, mungkin saja *cost* dari *vertex* A menuju *vertex* B tidak sama dengan *cost vertex* B menuju *vertex* A.

C. Minimum Spanning Tree



Gambar 2.1 *Spanning tree* yang dapat dibentuk dari graph dengan 4 buah *vertex* yang saling terhubung.

Minimum spanning tree adalah *sub-graph* dari sebuah *graph* yang mempunyai jumlah total *cost edge* paling sedikit. Sebuah *graph* dapat memiliki beberapa minimum *spanning tree*.

Berikut adalah syarat yang harus dipenuhi untuk mencari *minimum spanning tree* :

- *Graph* yang digunakan harus terhubung. Untuk mencari *minimum spanning tree* maka *graph* yang ada *vertex* nya sedemikian hingga terhubung agar dapat dibentuk *minimum spanning tree*.
- Harus mempunyai *cost* pada *edge* nya.
- *Minimum spanning tree* harus *acyclic*. *Acyclic* adalah tidak boleh ada sirkuit pada *tree* yang dibentuk.

D. Algoritma Prim

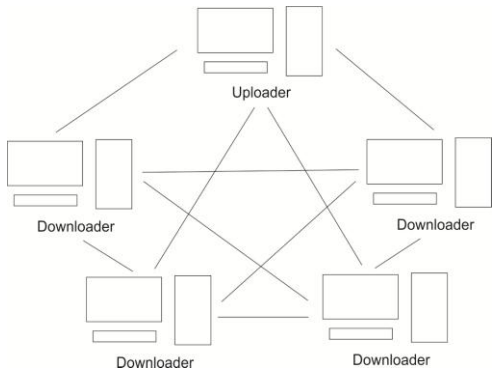
```
procedure Prim(Vertices, Edges) is
    V := {} //array list dari seluruh vertex
    E := {} //array list dari seluruh edge
    V.RemoveAt(0) //Anggaplah vertex awal
    adalah vertex pertama
    MinSpanTree := 0 //variabel nilai minimum
    spanning tree

    while V is not empty loop
        Edge optimalEdge := null
        for all Edge e in E loop
            if optimalEdge == null ||
            e.CompareTo(optimalEdge) < 0
            then
                optimalEdge := e
                MinSPanTree += e
            end if
        end loop
        V.Remove(Vertex yang mengandung
        optimalEdge)
        E.Remove(optimalEdge)
    end loop
end Prim
```

Algoritma Prim adalah algoritma *greedy* yang mencari *minimum spanning tree* pada *graph* yang mempunyai spesifikasi *undirected weighted graph*. Pada setiap perulangannya dicari subset dari kumpulan *edge* yang membentuk *tree acyclic* sampai seluruh *vertex* terkoneksi. Tentu saja pada setiap pencarian *edge*, dicari *edge* yang mempunyai *cost* paling kecil. Algoritma lain untuk persoalan ini adalah algoritma Kruskal dan algoritma Borúvska untuk mencari *minimum spanning forest* pada *graph* yang mempunyai kemungkinan *vertex* nya terputus atau tidak terhubung.

Algoritma Prim ditemukan pada tahun 1930 oleh matematikawan Czech yang bernama Vojtéch Jarník dan kemudian oleh ilmuwan komputer Robert C. Prim pada tahun 1957, dan kemudian ditemukan kembali oleh Edsger Dijkstra pada tahun 1959.

III. PEMBAHASAN

A. Ilustrasi Persoalan



Gambar 3.1 Iustrasi persoalan pada sebuah jaringan komputer.

Seseorang ingin berbagi sebuah file yang dalam kasus ini adalah menjadi server. Maka perangkat lunak torrent akan membuat sebuah file descriptor yang menandai bahwa file tersebut dapat diunduh oleh pengguna PC lain. Dari sisi server maka PC pengguna yang menjadi server akan menjadi vertex tujuan pada graph.

Apabila seseorang ingin mengunduh sebuah file, maka perangkat lunak torrent akan mencari jalur koneksi menuju PC yang menjadi server tersebut yang mana PC tersebut mempunyai file yang ingin diunduh. Disinilah implementasi algoritma prim untuk mencari minimum spanning tree (dalam kasus ini adalah jalur koneksi).

B. Implementasi Algoritma Prim

Untuk mencari minimum *spanning tree* dari sebuah *graph* pada makalah ini digunakan algoritma prim. Dengan asumsi server sudah mengetahui seluruh PC yang terkoneksi dengan jaringan dan setiap PC melakukan *request* untuk mengunduh file sekaligus memberikan status siap untuk diunduh file oleh PC lain. Dengan mengabaikan kualitas koneksi jaringan, maka kita dapat anggap setiap PC terkoneksi tanpa ada masalah di jaringan.

Dengan menggunakan algoritma prim, dicari jalur koneksi antar PC sehingga terbentuk jalur distribusi data yang sependek mungkin dan secepat mungkin. *Cost* pada setiap *edge* dapat kita anggap lama data untuk didistribusikan di jaringan, baik untuk klien yang mengunduh atau sebagai server yang mendistribusikan file.

C. Kompleksitas Algoritma Prim

Kompleksitas algoritma dihitung untuk mengetahui waktu yang dibutuhkan untuk mendapatkan minimum spanning tree yang digunakan untuk membuat jalur koneksi antar PC pengguna sehingga dapat terjadi distribusi data baik untuk mengunduh *file* maupun untuk berbagi *file*.

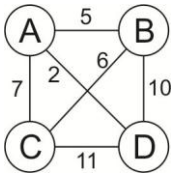
Pada kasus ini representasi graph yang digunakan adalah berupa matriks ketetanggaan. Graph yang mempunyai spesifikasi seperti ini mempunyai kompleksitas $O(n^2)$.

D. Simulasi

Untuk melakukan simulasi sebelumnya diberikan batasan persoalan seperti berikut :

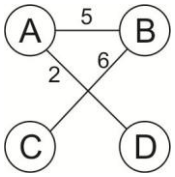
- *Graph* yang digunakan adalah *graph* yang memenuhi spesifikasi *undirected weighted graph*.
- Hanya menghitung lama waktu untuk menemukan *minimum spanning tree* pada sebuah *graph*.
- Seluruh *vertex* sedemikian hingga terhubung satu dengan lain.
- Anggaplah *cost* pada tiap *edge* merupakan representasi dari *bandwith* dan kualitas jaringan internet.
- *Graph* yang digunakan pada simulasi mempunyai *cost edge* yang berbeda-beda.
- Diberikan simulasi untuk *graph* dengan *vertex* berjumlah 4, 5, dan 6.

Simulasi dengan jumlah vertex sebanyah 4 buah, diilustrasikan sebagai berikut :



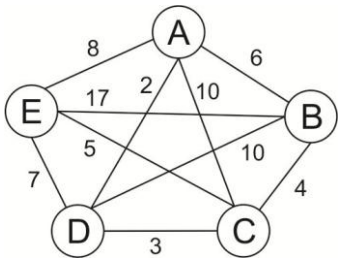
Gambar 3.2 *Graph* dengan 4 buah *vertex* yang saling terhubung.

Minimum spanning tree yang terbentuk, untuk acuan jalur koneksi :



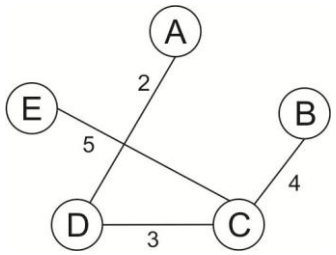
Gambar 3.3 Minimum spanning tree yang terbentuk.

Simulasi dengan jumlah vertex sebanyah 5 buah, diilustrasikan sebagai berikut :



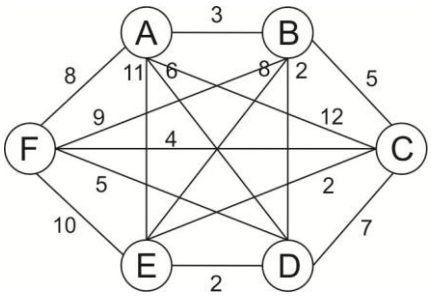
Gambar 3.4 *Graph* dengan 5 buah *vertex* yang saling terhubung.

Minimum spanning tree yang terbentuk, untuk acuan jalur koneksi :



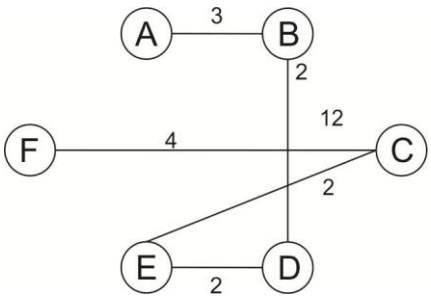
Gambar 3.5 Minimum spanning tree yang terbentuk.

Simulasi dengan jumlah vertex sebanyak 6 buah, diilustrasikan sebagai berikut :



Gambar 3.6 *Graph* dengan 6 buah *vertex* yang saling terhubung.

Minimum spanning tree yang terbentuk, untuk acuan jalur koneksi :



Gambar 3.7 Minimum spanning tree yang terbentuk.

Berikut adalah tabel hasil simulasi pencarian *minimum spanning tree* :

Tabel 3.1 Hasil Simulasi n vertex untuk mengetahui lama waktu pencarian minimum spanning tree.

Jumlah vertex	Waktu Mendapatkan MST
4	0,0014558 detik
5	0,0015275 detik
6	0,0015687 detik

IV. ANALISIS

Pada kasus graph yang mempunyai spesifikasi *undirected weighted graph* dan seluruh vertex terhubung, algoritma prim bekerja dengan cepat dalam menemukan *minimum spanning tree* yang menghubungkan seluruh *vertex*. Memang pada simulasi hanya dijalankan sampai *vertex* berjumlah 7. Mungkin saja hasilnya berbeda apabila spesifikasi *graph* yang digunakan berbeda.

Minimum spanning tree yang dibentuk oleh algoritma prim, pada kenyatannya adalah untuk mencari PC mana saja yang mempunyai *file* tertentu baik untuk diunduh atau untuk berbagi pakai. Dari kumpulan PC tersebut kita dapat alamat IP sehingga perangkat lunak *peer to peer file sharing* membuat jalur koneksi untuk pertukaran data antar PC.

Algoritma Prim bekerja secara efisien jika sebelumnya kita membuat *array list* dari *edge* dan *array list* dari *vertex*. Untuk setiap perulangan kita cari *edge* dengan *cost* paling kecil, kemudian setelah didapat, kita hapus edge dan *vertex* dari *array list*. Begitu seterunya sampai didapat *minimum spanning tree*. Cara ini secara otomatis akan menghindari terbentuknya sirkuit, sehingga setelah perulangan selesai didapat *minimum spanning tree* yang *acyclic*.

Tentu saja pada kenyataannya tidak selalu setiap PC terkoneksi langsung dengan PC lain. Mungkin saja Sebuah PC terhubung ke jaringan melalui PC lain, anggaplah ada sebuah jaringan lokal yang terkoneksi dengan jaringan internet. Untuk mencapai pada PC di jaringan lokal, haruslah melalui PC server lokal yang mengkoneksikan PC di jaringan lokal dengan jaringan internet.

Pada protokol *peer to peer file sharing* tentu saja sub-rutin nya tidak hanya menjalankan sebuah algoritma pencarian, banyak dari algoritma maupun kombinasi dari algoritma yang digunakan untuk mencari dan membuat koneksi antar PC sehingga memungkinkan terjadinya pertukaran data.

V. KESIMPULAN

Algoritma prim adalah algoritma yang mangkus untuk mencari minimum spanning tree untuk *protokol peer to peer file sharing*. Pada kasus dunia nyata dimana tidak selalu setiap PC terkoneksi dengan PC lain, perlu adanya algoritma khusus untuk mencari minimum spanning tree sebagai sarana pembuatan jalur distribusi data.

Seiring dengan meningkatnya *bandwith* internet juga semakin rumitnya jalur koneksi maka teknologi *peer to peer file sharing* akan semakin berkembang. Juga dengan semakin mudah dan murahnya untuk membangun sebuah perangkat PC yang mempunyai spesifikasi tinggi, maka akan semakin banyak server pribadi tersebar. Dari teknologi *peer to peer file sharing* banyak sekali yang dapat diimprovisasi seperti masalah keamanan data juga QoS (Quality of Service).

REFERENSI

[1] Steinmetz. Ralf, Wehrle. Klaus, "Peer-to-Peer Systems and Applications", Springer, ch. 289.

[2] http://www.bittorrent.com/, Diakses tanggal 12/6/2014 11:52.

[3] https://www.ics.uci.edu/~eppstein/161/960206.html, Diakses tanggal 12/6/2014 11:53.

[4] http://mathworld.wolfram.com/MinimumSpanningTree.html, Diakses tanggal 12/6/2014 11:53.

[5] https://www.ics.uci.edu/~eppstein/161/960206.html, Diakses tanggal 12/7/2014 2:28.

[6] https://code.google.com/p/graph-theory-algorithms-book/, Diakses tanggal 12/7/2014 2:04.

[7] https://www.princeton.edu/~achaney/tmve/wiki100k/docs/Greedy_algorithm.html, Diakses tanggal 12/7/2014 3:14.

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, 10 Desember 2014

Prafajar Rizkyanno Multazam

23514097