Penerapan Algoritma Branch and Bound Pada Jaringan Basis Data Tersebar

Dimas Aditiya Nurahman-13508093
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
if18093@students.if.itb.ac.id

Abstraksi—Dewasa ini penggunaan jaringan merupakan hal yang umum digunakan dalam berbagai keperluan transfer data. Data yang dimaksud dapat berupa berbagai macam tipe, mulai dari grafik, suara, video, dan lain-lain. Salah satu yang akan dibahas dalam makalah ini adalah mengenai *sharing* data berupa basis data tersebar (*distributed database*) yang dilakukan melalui jaringan. Pada proses *sharing databse* ini, dilakukan berbagai macam fungsi contohnya dalam melakukan *retrieve* data pada jaringan lain atau *server*. Proses *retrieve* tersebut akan mencari data yang kita inginkan pada *server* tertentu. Pencarian data dalam jaringan tersebut membutuhkan metode khusus agar data proses transfer data dapat berjalan secara optimal mengingat data yang akan di-*retrieve* berada pada *server* atau sumber yang berbeda-beda dalam kasus seperti basis data tersebar ini. Pada makalah ini akan diulas megenai salah satu cara atau metode yang dapat digunakan untuk mencari *sumber* data pada jaringan dalam kasus basis data tersebar. Adapun metode yang akan diulas adalah metode algoritma *branch and bound.* Dengan menggunakan algoritma *branch and bound* diharapkan proses *retrieve* basis data tersebar dapat optimal dan jalur pencarian yang digunakan memiliki *cost* yang seminimal mungkin. Kasus penyelesaian yang digunakan akan sangat berhubungan dengan kasus TSP (*Travelling Salesman Problem*).

Kata Kunci— *distributed database, server, retrieve, branch and bound.*

I. PENDAHULUAN

Basis data tersebar (*distributed database*) sering digunakan dalam kasus dimana basis data yang diperlukan tidak terletak dalam satu komputer, melainkan pada beberapa komputer yang saling berhubungan melalui jaringan. Sistem terdistribusi ini memungkinkan pengguna yang berbeda atau komputer untuk berbagi informasi. Sebuah basis data terdistribusi memungkinkan data tertentu operasi tertentu yang akan didistribusikan ke satu atau lebih mesin yang berbeda. Ini berarti bahwa keputusan untuk melakukan suatu operasi terjadi pada satu mesin dan kemudian satu atau lebih mesin lainnya berakhir melakukan operasi. Koleksi data (misalnya dalam database) dapat didistribusikan di beberapa lokasi fisik. Sebuah basis data terdistribusi bisa berada pada server jaringan di internet , pada perusahaan intranet atau extranet , atau di perusahaan lain jaringan . Replikasi dan distribusi database meningkatkan kinerja database di *worksites* pengguna akhir.

Operasi atau fungsi-fungsi yang diperlukan pada basis data terdistribusi meliputi pencarian, update, dan menghapus data. operasi komputasi terdistribusi bisa jauh lebih bervariasi. Termasuk melakukan perhitungan, penjadwalan tugas, transfer data, dan lain-lain. Proses pencarian termasuk proses *retrieve* di dalamnya. *Retrieve* data pada banyak komputer memerlukan suatu metode agar proses tersebut berjalan dengan optimal. Agar berjalan optimal, diperlukan algoritma yang membuat *cost* yang diperlukan menjadi seminimal mungkin. Algoritma *branch and bound* merupakan salah satu cara yang dapat digunakan untuk menemukan jalan minimal yang diperlukan dalam melakukan proses *retrieve* basis data.

Menggunakan algoritma *branch and bound*, *cost* yang didapat dari kecepatan transfer data dan jarak *server* ke *client* akan dimasukkan ke dalam struktur internal berbentuk seperti halnya matriks. Melalui matriks tersebut kemudian dihitung nilai dimana jalur yang akan dilalui pada proses pencarian atau *retrieve* data berjalan dengan optimal.

Proses pencarian *cost* oleh algoritma *branch and bound*, dilakukan secara iteratif hingga solusi yang diinginkan telah ditemukan. Adapun solusi yang dimaksud adalah *node* sebagai *server* penyedia data yang sedang dalam proses *retrieve.* Setelah *node* penyedia data yang diinginkan ditemukan, maka proses transfer data dapat dilakukan. Untuk lebih jelasnya mengenai teknis dari algoritma *branch and bound* dan sistem basis data terdistribusi akan dijelaskan pada sub bab-sub bab berikutnya.

II. BASIS DATA TERSEBAR

Sebelum menghubungkan penggunaan algoritma *branch and bound* pada kasus basis data tersebar (*distributed database*) akan sedikit diulas mengenai pengertian dari basis data tersebar itu sendiri.

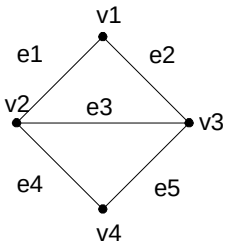
Yang dimaksud dengan basis data tersebar adalah basis data yang berada di bawah kendali pusat sistem manajemen database (DBMS) dimana lokasi tempat penyimpan data tidak hanya terletak pada suatu komputer maupun *server* . Data yang dimaksud dapat disimpan dalam beberapa komputer yang terletak di lokasi fisik yang sama, atau mungkin tersebar melalui jaringan komputer yang saling berhubungan. Proses menghubungkan rute pada jaringan inilah yang akan kemudian dibahas pada makalah ini.

Untuk memastikan bahwa basis data tersebar ini saling terkoneksi antar komputer, ada dua proses di dalamnya yaitu replikasi dan duplikasi. Replikasi melibatkan menggunakan software khusus yang mencari perubahan dalam database distributif. Setelah perubahan yang telah diidentifikasi, proses replikasi database membuat semua terlihat sama. Proses replikasi bisa memakan waktu sangat kompleks dan tergantung pada ukuran dan jumlah database distributif. Proses ini juga dapat memerlukan banyak waktu dan sumber daya komputer. Duplikasi di sisi lain tidak begitu rumit. Pada dasarnya mengidentifikasi satu database sebagai master dan kemudian duplikat database tersebut. Proses duplikasi biasanya dilakukan pada waktu yang ditetapkan pada waktu-waktu tertentu. Hal ini untuk memastikan bahwa setiap lokasi terdistribusi memiliki data yang sama. Dalam proses duplikasi, perubahan master database hanya diperbolehkan. Hal ini untuk memastikan bahwa data lokal tidak akan ditimpa. Kedua proses dapat menyimpan data saat ini di semua lokasi distributif.

Seperti yang telah dijelaskan sebelumnya, walaupun secara fisik disimpan di tempat-tempat yang berbeda, secara lojik basis data tersebut adalah satu basis data. Data yang tersebar dapat berupa data yang direplikasi dan difragmentasi. Kelompok data yang difragmentasi maksudnya adalah kelompok data yang dipotong-potong (terpisah-pisah) yang terletak pada tempat yang berbeda-beda. *Query* yang dilakukan pada basis data tersebut, akan menampilkan suatu sistem basis data yang lengkap. Contohnya yaitu proses pencarian atau *retrieve* yang mencari kumpulan data tersebut secara lengkap dari tempat yang berbeda-beda dan menampilkan basis data yang diinginkan secara lengkap sebagai satu kesatuan dari struktur lojik. Hal tersebut berarti pengguna harus dapat berinteraksi dengan sistem seolah-olah itu adalah satu sistem logis.

Sistem basis data tersebar ini dapat digambarkan sebagai sebuah graf (*graph*). Graf adalah sebuah struktur data lojik yang terdiri atas sekumpulan simpul (*vertices/nodes*) yang satu sama lain dihubungkan oleh sekumpulan sisi/bujur (*arcs/edges*). Graf $G = (V, E)$, yang dalam hal ini:

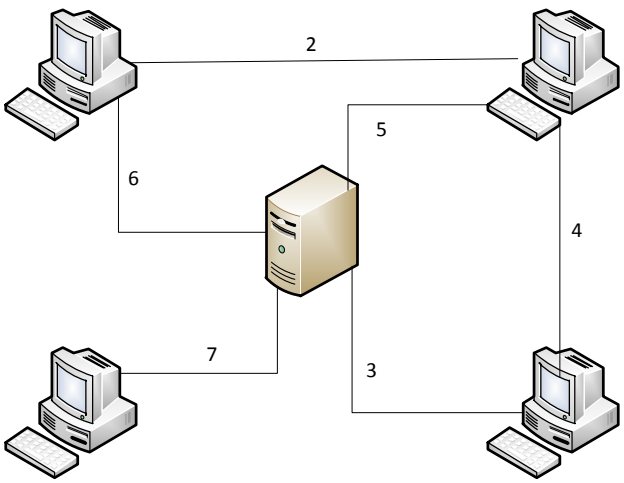
- V = himpunan tidak-kosong dari simpul-simpul (*vertices*)
  $= \{ v_1 , v_2 , ... , v_n \}$
- E = himpunan sisi (*edges*) yang menghubungkan sepasang simpul
  $= \{e_1 , e_2 , ... , e_n \}$



Gambar 1. Contoh graf

Graf pada gambar 1 di atas memiliki 4 buah simpul dengan 5 buah sisi. Tiap sisi graf bisa memiliki bobot (weight) tertentu. Graf yang sisinya memiliki bobot disebut sebagai graf berbobot (*weighted graph*). Setiap sisi dalam graf di atas tidak diketahui mana simpul awal dan simpul akhirnya, sehingga graf ini termasuk dalam graf tidak berarah (*undirected graph*).

Jika basidata tersebar direpresentasikan secara lojik sebagai graf, maka simpul merepresentasikan lokasi/*site* unit penyimpanan/pemrosesan data, sedangkan sisi merepresentasikan keberadaan h ubungan/koneksi antara dua buah lokasi. Pada setiap lokasi, dimungkinkan ada sebuah *server* sebagai unit penyimpanan dan pemrosesan data dan ada sebuah *client*, yaitu aplikasi yang melakukan operasi terhadap data-data yang disimpan. Suatu lokasi bisa saja terdiri atas sebuah server dan sebuah client, tapi bisa juga hanya memiliki server saja atau client saja. Jaringan yang menghubungkan satu lokasi dengan lokasi lain memiliki laju data transmisi (bandwidth) yang menyatakan jumlah maksimum data yang bisa dilewatkan dalam jaringan (baik untuk proses pengambilan/download data maupun pengiriman/upload data.



Gambar 2. Contoh jaringan basis data tersebar

Berdasarkan ilustrasi yang telah digambarkan di atas, graf jaringan basis data tersebar memiliki bobot tertentu yang merepresentasikan *cost* yang diperlukan dalam pencaria data. *Cost* inilah yang kemudian akan dijadikan perhitungan jalur padaproses pencarian data. Jalur optimal bergantung pada jumlah *cost* total yang dilalui.

III. Branch and Bound

A. Definisi

Branch and Bound(B & B) merupakan algoritma untuk mencari solusi optimal dari berbagai optimasi masalah, terutama dalam diskrit dan optimasi kombinatorial. Algoritma Branch and Bound (B&B) juga merupakan metode pencarian di dalam ruang solusi secara sistematis. *Branch and bound* juga merupakan pengembangan dari algoritma BFS hanya saja pembangkitan sub-pohon jalur menuju solusi dibatasi oleh suatu nilai atau batas (*bound)*.

Untuk mempercepat pencarian ke simpul solusi, maka setiap simpul diberi sebuah nilai ongkos (*cost*). Simpul berikutnya yang akan diekspansi tidak lagi berdasarkan urutan pembangkitannya (sebagaimana pada BFS murni),tetapi simpul yang memiliki ongkos yang paling kecil (least cost search) pada kasus minimasi. Hal ini dapat membuat penggunaan *branch and bound* dapat lebih efektif dan mangkus.

B. Algoritma

Seperti yang telah dijelaskan sebelumnya, algoritma *branch and bound* menggunakan batas nilai dalam proses pembangkitan sub-pohon, hal inilah yang membedakan algoritma *branch and bound* dengan algoritma BFS. Untuk menentukan nilai pembatas tersebut dibutuhkan suatu metode. Adapun fungsi heuristik untuk menentukan nilai batas bata sub-pohon pembangkit yang secara umum digunakan adalah

$$c(i) = f(i) + g(i)$$

c(i) = ongkos untuk simpul i
f(i) = ongkos mencapai simpul I dari akar
g(i) = ongkos mencapai simpul tujuan dari simpul i

Berdasarkan nilai huristik tersebut, nilai c(i) akan didapat. Nilai c(i) yang paling minimum merupakan nilai sub-pohon yang akan dibangkitkan/diekspansi. Adapun urutan atau langkah-langkah dalam penggunaan algoritma *branch and bound* adalah

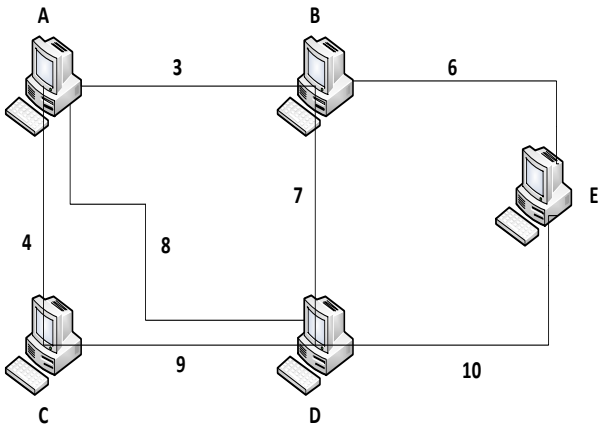
- Masukkan simpul akar ke dalam antrian q, bila simpul tersebut merupakan simpul solusi, maka solusi ditemukan.
- Jika antrian q kosong, maka tidak terdapat solusi.
- Jika antrian q tidak kosong, pilih dari antrian q simpul i yang memiliki c(i) minimum. Jika terdapat lebih dari satu simpul I, maka lakukan pemilihan secara acak.
- Jika simpul I merupakan dimpul solusi, maka solusi ditemukan, sebaliknya jika simpul i bukan merupakan simpul solusi, maka bangkitkan semua anak-anaknya, jika tidak memiliki anak maka kembali ke langkah 2.
- Untuk setiap anak j dari simpul i , hitung c(j) dan masukkan semua anak-anaknya ke dalam antrian q.
- Kembali ke langkah 2.

IV. Branch and Bound Pada Basis Data Tersebar

A. Eksperimen

Penggunaan algotritma *branch and bound* di sini terletak pada pencarian rute terpendek dengan *cost minimum* pada proses *retrieve* data. Algoritma *branch and bound* akan menentukan nilai batas/*bund* pada setiap jalur kemungkinan

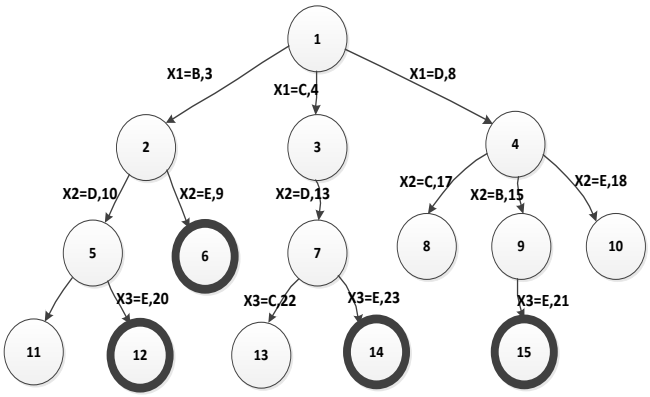
Berikut akan dijabarkan mengenai contoh permasalahan pencarian rute pada basis data terdistribusi.



Gambar 3. Ilustrasi permasalahan

Pada gambar 3 dapat dilihat gambaran jaringan basis data tersebar beserta bobot atau *cost* dari setiap jalur yang dapat dilalui.

Misalkan, *user* pada *node* A ingin melakukan *retrieve* data pada *node* E, maka kemungkinan-kemingkinan jalur yang ada dapat dilihat pada gambar 4.



Gambar 4. Ilustrasi jalur

Berbeda dengan skema BFS, metode *branch and bound* tidak membangkitkan semua pohon kemungkinan jalur, akan tetapi memberi nilai batas (*bound*) jalur mana yang akan dipilih.

Langkah selanjutnya adalah melakukan konversi graf jaringan tersebut ke dalam bentuk matriks.

$$\begin{bmatrix} \infty & 3 & 4 & 8 & \infty \\ 3 & \infty & \infty & 7 & 6 \\ 4 & \infty & \infty & 9 & \infty \\ 8 & 7 & 9 & \infty & 10 \\ \infty & 6 & \infty & 10 & \infty \end{bmatrix}$$

Lanjutkan dengan penyederhanaan matriks yaitu mengurangkan setiap baris dan kolom pada matriks hingga terdapat nilai 0 pada setiap baris dan kolom. Hasilnya adalah sebagai berikut.

$$\begin{bmatrix} \infty & 0 & 0 & 1 & \infty \\ 0 & \infty & \infty & 0 & 0 \\ 0 & \infty & \infty & 1 & \infty \\ 1 & 0 & 1 & \infty & 0 \\ \infty & 0 & \infty & 0 & \infty \end{bmatrix}$$

Total jumlah semua pengurang = 31 yang merupakan nilai dari c *root.* Setelah mengetahui c *root*, langkah selanjutnya adalah menghitung nilai fungsi batas dengan rumusan sebagai berikut.

$$c(s) = c(r) + A(i, j) + r$$

c(S) = bobot perjalanan minimum yang melalui simpul s

c(r) = bobot perjalanan minimum yang melalui simpul s, dimana r merupakan *parent* dari s

A(i,j) = ongkos graf sisi i, j

r = jumlah semua pengurang pada matriks tereduksi

Simpul 2; Jalur:A,B

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 & 0 \\ 0 & \infty & \infty & 1 & \infty \\ 1 & \infty & 0 & \infty & 0 \\ \infty & \infty & \infty & 0 & \infty \end{bmatrix}$$

Didapat nilai r = 1.

$$c(3) = c(1) + A(A, B) + r$$
$$c(3) = 31 + 3 + 1 = 35$$

Simpul 3; Jalur: A, C

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & 0 & 0 \\ \infty & \infty & \infty & 0 & \infty \\ 1 & 0 & \infty & \infty & 0 \\ \infty & 0 & \infty & 0 & \infty \end{bmatrix}$$

Didapat nilai r = 1.

$$c(4) = c(1) + A(A, C) + r$$
$$c(4) = 31 + 4 + 1 = 36$$

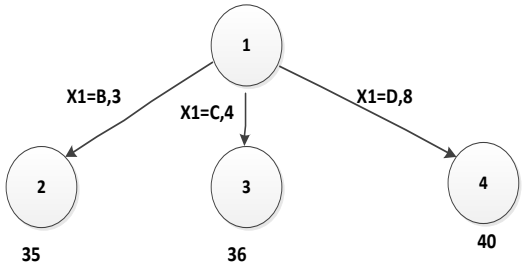
Simpul 4; Jalur :A,D

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & \infty & 0 \\ 0 & \infty & \infty & \infty & \infty \\ \infty & 0 & 0 & \infty & 0 \\ \infty & 0 & \infty & \infty & \infty \end{bmatrix}$$

Didapat nilai r = 1.

$$c(2) = c(1) + A(A, D) + r$$
$$c(2) = 31 + 8 + 1 = 40$$

Berdasarkan nilai rumusan di atas batas terkecil merupakan simpul yang akan diekspansi yaitu simpul 2. Pohon simpul sementara yang didapat adalah sebagai berikut.



Gambar 4. Pohon pertama

Langkah selanjutnya adalah membangkitkan anak-anak dari simpul 2 dan menghitung nilai sesuai rumusan di atas.

Simpul 5; Jalur: A, B, D

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & 0 \\ 0 & \infty & \infty & \infty & \infty \\ \infty & \infty & 0 & \infty & 0 \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

Didapat nilai r = 0.

$$c(6) = c(3) + A(B, D) + r$$

$$c(6) = 35 + 7 + 0 = 42$$

Simpul 6; Jalur: A, B, E

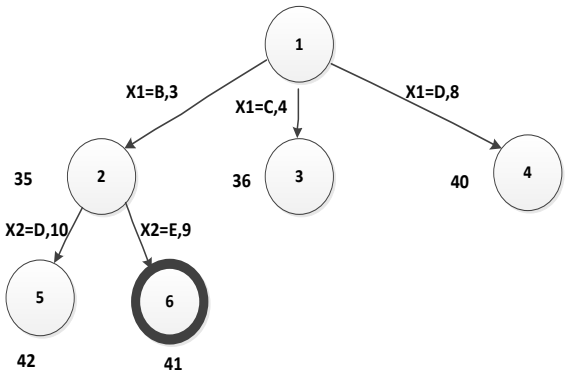
$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & 1 & \infty \\ 1 & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty \end{bmatrix}$$

Didapat nilai r = 0.

$$c(5) = c(3) + A(B, E) + r$$

$$c(5) = 35 + 6 + 0 = 41$$

Berdasarkan perhitungan di atas didapat pohon hasil sebagai berikut.



Gambar 5. Pohon kedua

B. Hasil

Berdasarkan eksperimen di atas dapat dilihat bah wa jalur terpendek atau optimal dari *node* A ke *node* E adalah dengan melewati jalur A-B-E yang memiliki nilai *cost* sebesar 9.

Pembuktian hasil optimal tidaknya algoritma *branch and* bound ini dapat dilihat pada proses pembangkitan seluruh kemungkinan jalur yang ada pada sub-bab eksperimen. Pada pohon tersebut, jalur dari A menuju ke E yang paling membutuhkan bobot atau *cost* yang paling kecil adalah jalur A-B-E.

Hal ini mnunjukkan apabila *user* mengkases dari *node* A dan kemudian ingin mendapatkan data yang ada pada *node* E jalur yang paling optimal adalah A-B-E.

V. KESIMPULAN

Algoritma *branch and bound* memiliki berbagai macam kegunaan, salah satu diantaranya adalah dalam kasus sistem jaringan basis data tersebar. Dewasa ini, penggunaan istem jaringan basis data sudah secara umum digunakan dan data yang diolah pun semakin berkembang dengan jumlah yang sangat banyak. Oleh karena itu, dibutuhkan suatu metode untuk dapat mengelola permasalah jumlah data yang cukup banyak agar pengolahan data dapat berjalan maksimal. Kasus sistem jaringan basis data tersebar membutuhkan suatu algoritma untuk melakukan *retrieve* data yang tersebar pada banyak komputer pada suatu jaringan, algoritma ini digunakan untuk mendeteksi jalur mana yang harus diambil agar proses pengolahan data dapat seefektif mungkin. Berdasarkan conoth kasus eksperimen yang dijabarkan pada makalah ini menggambarkan mengenai algoritma *branc and bound* yang dapat menemukan jalur yang paling membutuhkan bobot atau *cost* paling minimum.

REFERENSI

1. Munir, Rinaldi. (2009). *Strategi Algoritma*. Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung.
2. http://en.wikipedia.org/wiki/Branch_and_bound. Diakses tanggal 29 November 2010.
3. http://en.wikipedia.org/wiki/Distributed_database. Diakses tanggal 29 November 2010.
4. http://www4.informatik.uni-erlangen.de/~geier/corba-faq/why-distrib-computing.html. Diakses tanggal 29 November 2010.

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, 29 April 2010
ttd

Dimas Aditiya Nurahman-13508093