Algoritma *Branch and Bound* dalam *Social Network Friendship Problem*

Agus Gunawan – 13515143
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13515143@std.stei.itb.ac.id

***Abstract*—Seiring dengan majunya perkembangan teknologi makin banyak *Social Network* yang dikembangkan dan salah satu yang sudah terkenal adalah *Facebook*. Perubahan ini membuat makin banyak pertemanan yang dimiliki oleh seorang *user* yang ada pada sebuah *Social Network.* Terdapat sebuah permasalahan dimana seseorang ingin mengenal orang lain tetapi mereka terpisahkan oleh satu atau lebih teman yang mereka harus jadikan teman terlebih dahulu sebelum dapat berkenalan dengan orang yang diinginkan. Permasalahan ini dapat diselesaikan dengan algoritma *Breadth First Search* yang menghasilkan banyak teman minimum yang harus ditambahkan agar dapat mengenal orang yang diinginkan, namun memiliki kelemahan karena tidak menghitung kedekatan yang dimiliki dalam pertemanan tersebut karena faktanya teman – teman yang ada di *Social Network* seperti *Facebook* kebanyakan bukan benar – benar teman yang memang dikenal. Dalam makalah ini akan dijelaskan pendekatan yang lebih baik untuk menyelesaikan permasalahan ini dengan menggunakan algoritma *Branch and Bound* dimana nilai bobot yang digunakan berupa nilai kedekatan.**

Keywords—Branch and Bound; Breadth-First Search; Facebook; Kedekatan; Social Network Friendship

I. PENDAHULUAN

Di zaman sekarang ini *Social Media* merupakan hal yang sudah biasa digunakan oleh banyak orang. Baik digunakan untuk berhubungan sosial, bermain bersama ataupun membagikan info – info menarik. *Social Media* merupakan sebuah media online dimana para penggunanya dapat dengan mudah berpartisipasi, berbagi dan menciptakan isi meliputi Blog, jejaring sosial (*Social Network*), wiki, forum dan dunia virtual.

Salah satu contoh dari sosial media adalah jejaring sosial yang sampai saat ini sangat banyak digunakan oleh anak muda. Jejaring sosial (*Social Network*) adalah sebuah struktur sosial dimana struktur ini terdiri dari simpul – simpul yang merupakan individu atau organisasi yang saling berhubungan karena kesamaan sosialitas, mulai dari yang sudah mengenal sehari – hari sampai dengan keluarga. Tiap simpul – simpul di dalam jejaring sosial berhubungan satu dengan yang lainnya dan tiap – tiap simpul tersebut biasanya merupakan sebuah profil dari seseorang atau organisasi.

Facebook merupakan salah satu contoh dari jejaring sosial yang terkenal dan *facebook* merupakan salah satu jejaring sosial yang dapat di ekstraksi data pertemanannya sehingga dapat dijadikan sebuah graf yang merepresentasikan data pertemanan tiap *user* yang ada di *Facebook*.

Pertemanan antar *user* di *Facebook* tersebut tentunya dapat kita manfaatkan untuk mencari teman baru dengan cara meminta bantuan kepada teman yang kita miliki untuk mengenalkan diri kita kepada individu yang kita ingin ajak berteman tersebut. Tetapi mungkin saja kita tidak dapat langsung mengenal individu yang ingin kita kenal tersebut hanya dengan melewati satu teman saja sehingga kita harus melewati beberapa teman ataupun menambahkan beberapa pertemanan agar dapat menjadi teman dari seseorang yang ingin kita kenal tersebut. Tetapi kita tidak tahu berapa minimal teman yang harus ditambahkan agar dapat menjadi teman dari orang yang kita ingin kenal tersebut.

Terdapat sebuah teori yang dinamakan dengan "Six Degrees of Separation" dimana semua orang yang hidup di dunia ini dapat bertemu dengan hanya maksimal enam urutan rantai pertemanan (*a friend of a friend)* tetapi lama kelamaan teori tersebut dipatahkan dikarenakan rantai pertemanan yang dibutuhkan bahkan menjadi lebih sedikit lagi menurut beberapa riset yang dilakukan oleh *Facebook*. Tetapi, *Facebook* menggunakan data tersebut dengan asumsi ideal dimana seseorang dengan status pertemanan dengan orang lain yang memang benar–benar teman padahal kebanyakan teman di *Facebook* tersebut tidak saling mengenal dan jika diajak untuk membicarakan sesuatu juga tidak akan menjawab dengan baik sehingga diperlukan pendekatan baru dimana terdapat suatu nilai yang menunjukan kedekatan antara satu *user* dengan *user* lainnya.

Dengan latar belakang permasalahan inilah penulis ingin menyelesaikan permasalahan minimum rantai pertemanan yang dibutuhkan untuk dapat menemukan seseorang yang penulis selesaikan nantinya dengan pendekatan algoritma *Branch and Bound* dan akan dibandingkan dengan pendekatan algoritma *Breadth First Search*.

II. DASAR TEORI

2.1. *Facebook Friendship Problem*

Facebook merupakan salah satu jejaring sosial yang sangat terkenal. *Facebook* memiliki sebuah fitur pertemanan dimana tiap *user* dapat menambahkan teman yang *user* tersebut miliki dengan cara menambah seseorang sebagai

temannya. Status pertemanan inilah yang nantinya dapat diekstraksi dengan menggunakan *crawler*, hanya jika *privacy settings* dari sebuah profil mengijinkan hal tersebut. Hasil dari ekstraksi data ini nantinya akan direpresentasikan sebagai sebuah graf dengan simpul dan sisi dimana sebuah simpul merepresentasikan profil dari *user* dan sebuah sisi dari dua buah simpul menandakan bahwa kedua *user* tersebut berteman di *Facebook.*

Facebook Friendship Problem ini merupakan sebuah permasalahan *Social Network Friendship* yang penulis angkat agar dapat diselesaikan dengan pendekatan algoritma Graf seperti BFS (*Breadth-First Search)* yang nantinya penulis akan menyelesaikan permasalahan ini dengan menggunakan algoritma *Branch and Bound* untuk dibandingkan dengan hasil dari algoritma BFS. Permasalahan ini dapat dimodelkan sebagai sebuah permasalahan pencarian banyak rantai pertemanan atau rantai pertambahan pertemanan minimum yang dibutuhkan agar seseorang dapat berkenalan dengan orang lain yang dipisahkan dengan beberapa orang tersebut (*a friend of a friend)*.

2.2. Graf

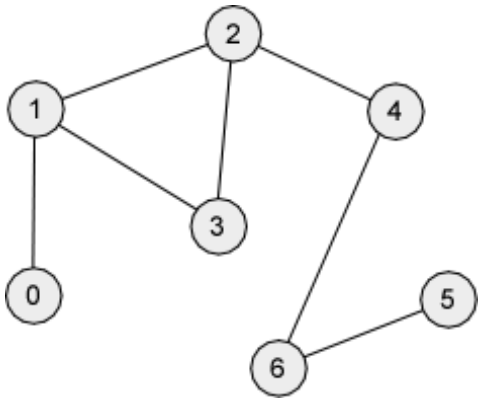
Graf merupakan sebuah struktur data yang terdiri dari simpul dan sisi dimana setiap sisi dari sebuah graf menghubungkan dua buah simpul tetapi sebuah sisi juga dapat menghubungkan sebuah simpul dengan dirinya sendiri. Graf didefinisikan sebagai himpunan pasangan simpul dan sisi (V, E) di mana :

- V merupakan himpunan yang tidak boleh kosong dari simpul – simpul yang ada pada suatu graf {v1, v2, v3, …, .vn}
- E merupakan himpunan yang boleh kosong dari sisi – sisi yang menghubungkan dua buah simpul {e1, e2, e3, …, en}

Terdapat beberapa jenis graf yang akan digunakan dalam tulisan ini seperti :

i. Graf Tak-Berarah (*Undirected Graph*)

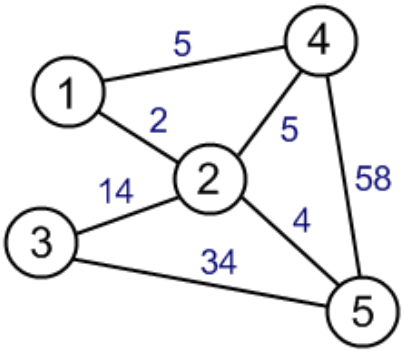
Graf tak-berarah adalah graf yang sisinya tidak memiliki orientasi arah sehingga urutan pasangan simpul yang dihubungkan oleh sebuah sisi tidak diperhatikan. (v1, v2) = (v2, v1) adalah dua buah sisi yang sama pada graf tidak berarah.



Gambar 1. Contoh dari Graf Tak-Berarah
Sumber : stackoverflow.com

ii. Graf Berbobot (*Weighted Graph*)

Graf berbobot adalah graf yang setiap sisinya memiliki sebuah bobot (harga). Bobot pada setiap sisi merepresentasikan hal – hal yang berbeda tergantung pada permasalahan yang dimodelkan menggunakan graf. Bobot dapat merepresentasikan jarak antar dua buah kota, ongkos produksi, nilai kedekatan antar *user* satu dengan *user* lainnya pada sebuah jejaring sosial.



Gambar 2. Contoh dari Graf Berbobot Tak-Berarah
Sumber : web.cecs.pdx.edu

2.3. Algoritma BFS

Algoritma BFS merupakan algoritma traversal graf yang merupakan algoritma yang melakukan sebuah traversal graf dengan cara mengunjungi simpul secara sistematik yang melakukan pencarian secara melebar dan merupakan algoritma pencarian tanpa informasi. Algoritma BFS direpresentasikan dengan langkah – langkah sebagai berikut :

i. Traversal mulai dari simpul x.
ii. Kunjungi semua simpul yang bertetangga dengan simpul x terlebih dahulu.
iii. Kunjungi semua simpul yang belum dikunjungi dan bertetangga dengan simpul – simpul yang sebelumnya dikunjungi.
iv. Lakukan langkah (ii) seterusnya sampai semua simpul telah dikunjungi.

Biasanya algoritma ini menggunakan struktur data antrian untuk mempermudah implementasi algoritma ini karena algoritma ini mengunjungi setiap simpul secara melebar dan struktur data antrian cocok untuk melakukan pencarian simpul secara melebar tersebut. Selain itu, digunakan juga dua struktur data lainnya seperti matriks ketetanggaan dimana nilai setiap elemen matriksnya menandakan status ketetanggaan antara dua simpul dan tabel boolean yang menyimpan simpul mana saja yang sudah ataupun belum dikunjungi.

2.4. Algoritma Branch and Bound

Algoritma Branch and Bound adalah sebuah algoritma yang digunakan untuk persoalan optimasi yang biasanya persoalan ini merupakan persoalan dimana variabel yang ingin dioptimasi berupa sebuah bilangan. Algoritma ini biasanya digunakan untuk meminimalkan ataupun memaksimalkan

suatu fungsi objektif dimana fungsi objektif tersebut tidak boleh melanggar suatu batasan (*constraint*).

Branch and Bound mirip dengan *Breadth First Search* namun aturan ekspansi simpulnya tidak sama dengan *Breadth First Search* dimana BFS melakukan ekspansi simpul berdasarkan urutan pembangkitannya. Sedangkan, algoritma *Branch and Bound* melakukan ekspansi simpul berdasarkan *Least Cost Search*.

Secara umum algoritma *Branch and Bound* memberi sebuah simpul dengan sebuah nilai *cost* yang merupakan sebuah nilai taksiran termurah ataupun termahal tergantung dengan persoalan optimasi yang dilakukan ke simpul status tujuan melalui sebuah simpul yang dipilih tersebut. Simpul berikutnya yang akan diekspan adalah simpul dengan *cost* terkecil ataupun terbesar tergantung dari persoalan optimasi yang dilakukan.

Algoritma *Branch and Bound* juga menerapkan sebuah fungsi pembatas yang memangkas sebuah simpul ketika simpul tersebut sudah dianggap tidak lagi mengarah ke sebuah solusi.

Algoritma *Branch and Bound* biasanya diimplementasikan dengan menggunakan sebuah struktur data *priority queue* yang digunakan untuk mengatur urutan ekspansi sebuah simpul dikarenakan aturan yang digunakan algoritma ini adalah *Least Cost Search*.

III. PEMBAHASAN

Penyelesaian permasalahan ini dapat menggunakan dua cara yaitu *Breadth-First Search* (BFS) dan *Branch and Bound*. Kedua cara tersebut akan penulis jelaskan tetapi fokus penulis adalah untuk membandingkan algoritma *Branch and Bound* yang penulis rumuskan dengan pendekatan *Breadth First Search.*

A. Hasil Ekstraksi Data dari *Facebook*

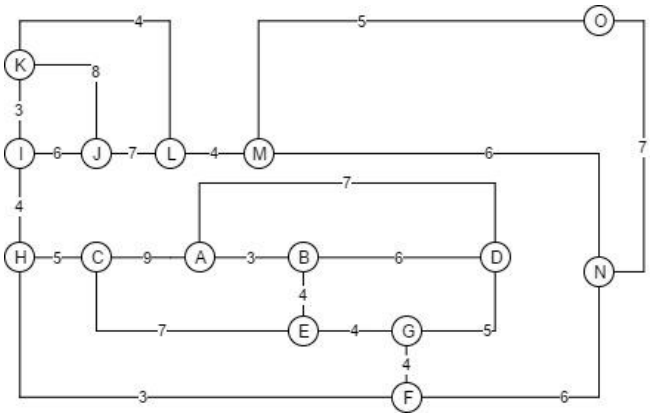
Ekstraksi Data dari *Facebook* menggunakan sebuah *crawler* yang dapat mengambil seluruh data pertemanan dari seorang *user* hanya jika *user* tersebut memiliki pengaturan *privacy* yang mengijinkan seseorang yang tidak dikenal melihat status pertemanan yang *user* tersebut miliki. Cara kerja *crawler* adalah mengambil seluruh data pertemanan dari seorang *user* yang kemudian tiap pertemanan tersebut akan digunakan *crawler* ini untuk menentukan nilai kedekatan dari seseorang dengan seorang temannya tersebut dengan menggunakan pendekatan berapa banyak kedua *user* tersebut bertukar "like", berapa banyak kedua *user* tersebut saling mengomentari di sebuah status, dan berapa banyak kedua *user* tersebut saling membagikan statusnya dengan satu sama lain. Ketiga parameter tersebut nantinya akan menentukan nilai kedekatan antar *user* yang berteman. Penukaran pesan antara *user* tidak dijadikan parameter kedekatan karena tidak mungkin *crawler* ini dapat mengambil data pertukaran pesan karena hal tersebut sudah di luar batas *privacy*. Nantinya nilai kedekatan yang diberikan antar dua *user* yang saling berteman tersebut memiliki skala antara (1 – 9) dimana makin kecil angka yang diberikan makin dekat kedekatan yang dimiliki antar *user* tersebut. Perhitungan yang dilakukan untuk menentukan skala kedekatan tersebut tidak penulis bahas tetapi secara umum perhitungan tersebut dihitung berdasarkan jumlah ketiga parameter tersebut dibagi dengan jumlah total dari ketiga parameter tersebut pada semua *user* yang diambil kemudian diurutkan dan ditentukan skalanya berdasarkan sebuah distribusi. Contoh hasil ekstraksi yang didapatkan dengan simplifikasi yang penulis lakukan adalah sebagai berikut :

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
A	x	3	9	7	x	x	x	x	x	x	x	x	x	x	x
B	3	x	x	6	4	x	x	x	x	x	x	x	x	x	x
C	9	x	x	x	7	x	x	5	x	x	x	x	x	x	x
D	7	6	x	x	x	x	5	x	x	x	x	x	x	x	x
E	x	4	7	x	x	x	4	x	x	x	x	x	x	x	x
F	x	x	x	x	x	x	4	3	x	x	x	x	x	6	x
G	x	x	x	5	4	4	x	x	x	x	x	x	x	x	x
H	x	x	5	x	x	3	x	x	4	x	x	x	x	x	x
I	x	x	x	x	x	x	x	4	x	6	3	x	x	x	x
J	x	x	x	x	x	x	x	x	6	x	8	7	x	x	x
K	x	x	x	x	x	x	x	3	8	x	4	x	x	x	x
L	x	x	x	x	x	x	x	x	x	7	4	x	4	x	x
M	x	x	x	x	x	x	x	x	x	x	x	4	x	6	5
N	x	x	x	x	x	6	x	x	x	x	x	x	6	x	7
O	x	x	x	x	x	x	x	x	x	x	x	5	7	x	

Gambar 3. Hasil Ekstraksi Data dari *Facebook*
Sumber : penulis

Hasil dari ekstraksi data menghasilkan sebuah matriks ketetanggaan dimana nilai yang ada pada setiap elemen matriks tersebut merupakan nilai kedekatan antar dua user berbeda dan ketika nilai tersebut merupakan karakter 'x' berarti kedua user tersebut tidak berteman.

Graf yang dihasilkan dari ekstraksi data merupakan graf tidak berarah yang memiliki bobot di tiap sisinya jika kedua simpul tersebut memang bertetangga. Gambar berikut adalah graf yang dihasilkan dari data matriks ketetanggaan di atas :



Gambar 4. Graf Hasil Ekstraksi
Sumber : penulis

Perlu diketahui untuk *crawler* penulis tidak membuatnya dan penulis membuat hasil ekstraksi data tersebut berdasarkan melihat profil *Facebook* dari tiap user lainnya dan penulis lakukan simplifikasi sehingga dihasilkan data seperti gambar tersebut.

Permasalahan yang diberikan dengan data matriks ketetanggaan hasil ekstraksi tersebut adalah bagaimana cara *user* A dapat berteman dengan *user* O. Permasalahan ini akan

diselesaikan dengan menggunakan pendekatan BFS yang tidak melihat data kedekatan sama sekali dan algoritma *Branch and Bound* yang menggunakan data kedekatan *user* tersebut dalam perhitungan nilai fungsi objektifnya.

B. Penyelesaian dengan Menggunakan Algoritma BFS

Penyelesaian dengan Menggunakan Algoritma BFS (*Breadth First Search)* untuk permasalahan yang telah dijabarkan adalah sebagai berikut :

1. Kunjungi simpul A.
2. Kunjungi semua simpul yang bertetangga dengan A yaitu B, C dan D.
3. Kunjungi simpul yang belum dikunjungi dan bertetangga dengan simpul – simpul yang telah dikunjungi sebelumnya, lakukan langkah ini secara terus menerus sampai bertemu dengan simpul tujuan yaitu O.

Pada setiap iterasi diperlukan juga nilai cost yang dicatat dari seluruh jalan yang nantinya akan dipilih. Hal ini digunakan untuk membandingkan dengan hasil algoritma *Branch and Bound* yang nantinya akan dijelaskan oleh penulis. Penyelesaiaan permasalahan tersebut dengan menggunakan algoritma BFS yang telah dijabarkan dapat digambarkan sebagai berikut dimana hanya dijelaskan beberapa saja karena jika dijelaskan seluruhnya akan menghabiskan halaman yang banyak :

Iterasi Ke -	**Simpul Ekspansi**	**Simpul yang Dibangkitkan**
1	A	Ba, Ca, Da
2	Ba	Ca, Da, Dab, Eab
3	Ca	Da, Dab, Eab, Eac, Hac
4	Da	Dab, Eab, Eac, Hac, Bad, Gad
5	Dab	Eab, Eac, Hac, Bad, Gad, Aabd, Babd, Gabd
6	Eab	Eac, Hac, Bad, Gad, Aabd, Babd, Gabd, Babe, Cabe, Gabe
…	Oachfno (solusi)	…

Tabel 1. Cuplikan langkah eksekusi BFS
Sumber : penulis

Hasil program yang dihasilkan dengan menggunakan algoritma BFS pada data uji dari data ekstraksi yang telah dijabarkan sebelumnya adalah sebagai berikut :

```
A -> C -> H -> F -> N -> O
Bobotnya :
30
Jumlah Simpul yang Dibangkitkan :
85
Waktu yang Dibutuhkan:
0.000845 s
```

Gambar 5. Hasil Program BFS
Sumber : penulis

Cost yang dibutuhkan dalam penyelesaian permasalahan tersebut menggunakan algoritma BFS (*Breadth-First Search*) adalah 30. Rantai pertemanan yang harus ditambahkan menurut algoritma BFS ini adalah meminta C untuk mengenalkan diri kita kepada H. Meminta H untuk mengenalkan diri kita kepada F. Begitu seterusnya sampai kita dapat meminta N untuk mengenalkan diri kita dengan O. Tetapi dalam algoritma BFS ini bobot (*cost*) tidak dilihat sama sekali sehingga menurut penulis hasilnya sedikit kurang baik dikarenakan mungkin saja ketika kita meminta N untuk mengenalkan diri kita kepada *user* O, user O tidak ingin dikenalkan karena user O bahkan tidak terlalu mengenal *user* N ataupun saat kita ingin berkenalan dengan F ternyata F tidak memiliki kedekatan dengan N sehingga saat kita dikenalkan dengan N, N tidak ingin menerima permintaan pertemanan kita. Sehingga algoritma ini hanya cocok jika pendekatan yang dilakukan diberikan asumsi dimana jika kedua *user* saling berteman itu artinya kedua *user* memang saling mengenal. Tetapi permasalahannya banyak sekali teman di *Facebook* yang tidak kita kenal tetapi kita memiliki status pertemanan dengan *user* yang kita tidak kenal tersebut.

Hasil yang sedikit kurang baik tersebutlah yang membuat penulis merumuskan pendekatan lainnya dengan menggunakan algoritma *Branch and Bound* yang penulis jelaskan pada bagian selanjutnya.

C. Penyelesaian dengan Menggunakan Algoritma *Branch and Bound*

Pennyelesaian dengan menggunakan agloritma *Branch and Bound* pada permasalahan yang telah dijelaskan adalah sebagai berikut :

1. Masukkan simpul akar ke antrian prioritas (*priority queue*) Q dimana makin kecil nilai costnya makin tinggi prioritasnya. Stop jika simpul akar adalah simpul solusi.
2. Stop jika Q kosong karena ini menandakan tidak ada solusi.
3. Jika Q tidak kosong pilih elemen pertama yaitu simpul i dari *priority queue* Q untuk diekspansi (elemen yang memiliki nilai *cost* paling kecil).
4. Jika simpul i adalah simpul solusi berarti solusi sudah ditemukan, hapus semua simpul yang memiliki nilai *cost* lebih dari nilai cost simpul i tersebut. Bangkitkan seluruh anak – anak dari simpul i jika simpul i bukan merupakan simpul solusi. Kembali ke langkah 2 jika simpul i tidak memiliki anak.
5. Untuk setiap anak j dari simpul i hitung cost dari j tersebut kemudian masukan ke dalam *priority queue* Q.
6. Kembali ke langkah 2

Nilai *cost* yang digunakan dalam perhitungan tiap simpulnya adalah

$$C(S) = C(R) + A(R,S)$$

Dimana S merupakan simpul anak yang ingin dibangkitkan dan R merupakan simpul parent dari anak tersebut sedangkan nilai A(R,S) adalah nilai matriks ketetanggaan dari simpul akar ke simpul anaknya.

Nilai *cost* yang digunakan dalam penyelesaian masalah yang telah dijabarkan menggunakan algoritma *Branch and Bound* adalah total nilai kedekatan yang didapatkan dalam sebuah urutan pertemanan dimana makin kecil nilainya maka menandakan urutan / rantai pertemanan tersebut makin erat sehingga makin mudah untuk meminta bantuan antar teman yang satu dengan yang lainnya untuk dapat mengenalkan diri kita kepada orang lain yang tidak kita kenal sama sekali namun merupakan teman dari teman dalam urutan tersebut.

Ilustrasi penyelesaian masalah dengan menggunakan algoritma *Branch and Bound* dapat digambarkan sebagai berikut dimana ilustrasi berikut hanya berupa cuplikan :

Iterasi Ke -	**Simpul Ekspansi**	**Simpul yang Dibangkitkan**
1	A	Ba(3), Da(7), Ca(9)
2	Ba	Da(7), Dab(7) Ca(9), Eab(9)
3	Da	Dab(7), Ca(9), Eab(9), Gad(12), Bad(13)
4	Dab	Ca(9), Eab(9), Gad(12), Gabd(12), Bad(13), Babd(13), Aabd(14)
5	Ca	Eab(9), Gad(12), Gabd(12), Bad(13), Babd(13), Aabd(14), Hac(14), Eac(16)
6	Eab	Gad(12), Gabd(12), Bad(13), Babd(13), Aabd(14), Hac(14), Gabe(14), Babe(15), Eac(16), Cabe(16)
…	Oabegfno (28) (solusi)	…

Tabel 2. Cuplikan langkah eksekusi *Branch and Bound*
Sumber : penulis

Hasil program dengan menggunakan algoritma Branch and Bound adalah sebagai berikut :

```
A -> B -> E -> G -> F -> N -> O
Bobotnya :
28
Jumlah Simpul yang Dibangkitkan :
101
Waktu yang Dibutuhkan:
0.001107 s
```

Gambar 6. Hasil Program *Branch and Bound*
Sumber : penulis

Hasil penyelesaian masalah tersebut menggunakan algoritma *Branch and Bound* menghasilkan nilai *cost* (bobot) sebesar 28. Urutan pertemanan yang harus ditambahkan adalah meminta B untuk mengenalkan diri kita kepada E, meminta E untuk mengenalkan diri kita kepada G, meminta G untuk mengenalkan diri kita kepada F, begitu seterusnya sampai kita meminta N untuk mengenalkan diri kita kepada O.

Nilai bobot yang dihasilkan menggunakan algoritma *Branch and Bound* ini lebih kecil dibandingkan dengan algoritma *Breadth-First Search* sehingga menandakan rantai urutan pertemanan yang dihasilkan lebih erat. Hasil penyelesaian masalah dengan menggunakan algoritma *Branch and Bound* ini akan dibandingkan dengan algoritma *Breadth-First Search* pada bagian selanjutnya yaitu analisis.

D. Analisis

Hasil penyelesaian permasalahan ini menggunakan algoritma *Breadth-First Search* menghasilkan rantai urutan pertemanan yaitu A -> C -> H -> F -> N -> O dengan bobot (*cost*) sebesar 30. Nilai tersebut artinya rata – rata kedekatan antar orang satu dengan orang berikutnya memiliki nilai kedekatan 30 / 5 = 6 dimana nilai ini berskala antara 1 sampai dengan 9 dan makin kecil nilai kedekatan tersebut menandakan hubungan antar orang yang satu dengan yang lainnya makin erat.

Hasil penyelesaian permasalahan ini menggunakan algoritma *Branch and Bound* menghasilkan rantai urutan pertemanan A -> B -> E -> G -> F -> N -> O dengan bobot (*cost*) sebesar 28. Nilai tersebut artinya rata – rata kedekatan antar orang satu dengan orang berikutnya pada rantai tersebut adalah 28 / 6 = 4.67 dimana nilai kedekatan ini lebih baik dibandingkan dengan algoritma BFS yang hanya memiliki nilai kedekatan rata – rata sebesar 6.

Hasil penyelesaian masalah dengan pendekatan *Branch and Bound* menurut penulis merupakan hasil yang paling cocok dalam permasalahan ini dan lebih baik dibandingkan dengan menggunakan pendekatan algoritma *Breadth-First Search* dikarenakan hasil *Branch and Bound* sudah menggunakan nilai kedekatan antar orang satu dengan orang lainnya yang berteman sedangkan untuk pendekatan algoritma *Breadth First Search* hasil nilai kedekatan yang dihasilkan lebih buruk dan tidak menggunakan data nilai kedekatan sama sekali. Pada kenyataannya kebanyakan orang yang berada di *Social Network* seperti *Facebook* tersebut tidak saling mengenal sama sekali, tetapi mereka yang tidak saling mengenal tersebut dapat berteman di *Facebook* sehingga hal inilah membuat beberapa penelitian yang dilakukan *Facebook* yang kebanyakan menggunakan algoritma *Breadth-First Search* agak bias karena tidak memperhitungkan nilai kedekatan tersebut. Sehingga pendekatan dengan menggunakan algoritma *Branch and Bound* ini seharusnya dapat digunakan agar dapat menghasilkan hasil yang benar–benar baik dan dapat menyimpulkan bahwa antar dua orang di dunia ini dipisahkan oleh berapa urutan rantai pertemanan minimal. Tetapi pendekatan dengan algoritma *Breadth First Search* tidak sepenuhnya buruk karena algoritma BFS yang digunakan tersebut dilakukan dengan melakukan pendekatan ideal dimana tiap orang satu dengan lainnya benar–benar berteman di sebuah *Social Network.*

Penyelesaian masalah ini menggunakan algoritma BFS dapat digunakan sebagai *benchmark* dalam penentuan algoritma – algoritma yang baik selanjutnya seperti *Branch and Bound* yang nilai *cost*nya menggunakan sebuah nilai kedekatan. Namun, pendekatan yang dilakukan penulis menggunakan nilai kedekatan juga dapat sedikit bias dikarenakan nilai kedekatan yang digunakan penulis hanya dalam skala satu sampai dengan sembilan seharusnya dapat diperbesar lagi agar hasil yang didapatkan menjadi lebih baik

lagi sehingga dapat menyelesaikan masalah ini dengan lebih baik lagi. Selain itu, nilai kedekatan yang dihasilkan seharusnya tidak hanya tergantung pada tiga parameter yang sudah dijelaskan saja karena mungkin ada beberapa parameter lainnya yang lebih signifikan lagi. Penulis menggunakan tiga parameter sebelumnya dikarenakan hanya itu saja yang dapat penulis lihat dan kuantifikasi agar dapat dibuat pemodelan dari kedekatan antar *user* tersebut.

IV. KESIMPULAN

Penyelesaian permasalahan ini dengan menggunakan algoritma *Branch and Bound* dapat dilakukan dengan baik dan bahkan lebih baik dibandingkan dengan *benchmark* yang digunakan yaitu dengan menggunakan algoritma *Breadth-First Search*. Namun, penyelesaian masalah dengan menggunakan algoritma *Branch and Bound* tersebut memiliki beberapa kelemahan di bagian heuristiknya karena parameter yang digunakan mungkin kurang signifikan atau skala yang digunakan mungkin kurang besar. Jika kedua hal tersebut dapat diperbaiki lagi maka penyelesaian masalah ini dengan menggunakan algoritma *Branch and Bound* dapat menjadi lebih baik lagi.

Meskipun dengan kelemahan seperti itu algoritma ini menurut penulis sangat cocok pada kondisi permasalahan yang sebenarnya terjadi pada *Social Network* yang menurut penulis juga dapat memberikan hasil yang benar–benar baik. Sehingga algoritma ini mungkin dapat dijadikan sebagai *benchmark* untuk dapat menentukan algoritma selanjutnya dengan nilai heuristik yang lebih baik dibandingkan dengan yang penulis rumuskan.

UCAPAN TERIMA KASIH

Penulis ucapkan terima kasih kepada Tuhan Yang Maha Esa karena berkat rahmatnya penulis dapat menyelesaikan makalah ini dengan baik. Terima kasih juga kepada Bu Nur Ulfa Maulidevi sebagai dosen yang sudah membimbing penulis dengan baik dan Pak Rinaldi Munir dan juga Bu Masayu Leylia Khodra yang telah membagikan ilmunya dengan baik, karena tanpa ilmu yang telah dibagikan penulis tidak mungkin dapat menulis makalah ini dengan baik. Terima kasih juga kepada seluruh pihak yang telah membantu dalam penyelesaian makalah ini.

DAFTAR PUSTAKA

[1] https://www.quora.com/What-are-some-real-life-examples-of-Breadth-and-Depth-First-Search diakses pada 16 Mei 2017 pukul 15.03

[2] https://core.ac.uk/download/pdf/107698.pdf diakses pada 16 Mei 2017 pukul 15.55

[3] https://research.fb.com/three-and-a-half-degrees-of-separation/ diakses pada 16 Mei 2017 pukul 16.43

[4] https://wibawaadiputra.wordpress.com/2013/01/27/media-sosial-jejaring-sosial-social-media-social-network/ diakses pada 16 Mei 2017 pukul 17.05

[5] Munir, Rinaldi. Matematika Diskrit. Bandung: Informatika Bandung, 2005.

[6] Slide kuliah Strategi Algoritma 2016 - 2017 dari http://informatika.stei.itb.ac.id/~rinaldi.munir/

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, 17 Mei 2017

Agus Gunawan - 13515143