Mencari Rute Pembelian Barang Dalam Acara Uang Kaget Dengan Algoritma Branch and Bound

Hafid Abi Daniswara - 13519028
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail : 13519028@std.stei.itb.ac.id

***Abstract*— Acara uang kaget merupakan salah satu acara realitas yang cukup populer di Indonesia. Konsep dari acara ini adalah memberikan hadiah uang secara random kepada seseorang terpilih, kemudian orang terpilih wajib untuk membelanjakan seluruh uang tersebut dalam waktu yang telah ditentukan. Namun pernahkah anda membayangkan bagaimana harus memanfaatkan waktu yang cukup terbatas untuk berkeliling membelanjakan uang dari hadiah acara tersebut. Pada makalah ini akan dibahas melakukan optimasi rute belanja untuk mendapatkan keuntungan yang maksimal dari hadiah uang kaget tanpa melebihi waktu yang telah ditentukan dengan menggunakan algoritma *branch and bound*.**

Keywords*— *branch and bound, knapsack, uang kaget, optimasi, pencarian rute.

I. Pendahuluan

Uang kaget merupakan acara realitas yang ditayangkan di salah satu stasiun televisi swasta di Indonesia. Acara ini diinisiasi oleh Helmy Yahya, yang kebetulan juga tampil sebagai pembawa acara dan menyamar sebagai Mr. Easy Money. Acara ini seiring waktu telah berganti-ganti nama menjadi Uang Kaget Merubah Nasib lalu menjadi Duit Kaget, serta pembawa acara dari acara ini juga telah berganti beberapa kali.

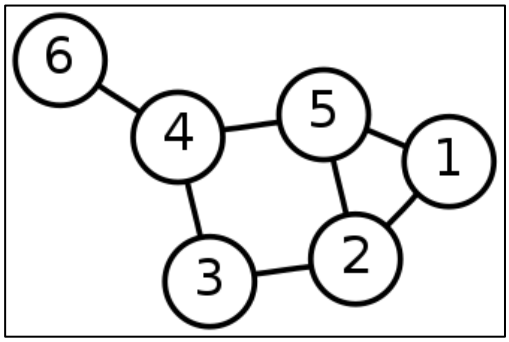
Gambar 1. Acara Uang Kaget
Sumber :
https://miro.medium.com/max/1600/0*MNPp6gh5PuyDNDvr.jpg diakses pada 6 Mei 2021 pukul 20.45

Walau telah berganti nama dan pembawa acara beberapa kali, secara garis besar, konsep yang dibawakan oleh acara ini tetaplah sama. Konsep uang kaget adalah mencari seseorang target dengan berkeliling kota (target biasanya berasal dari orang-orang yang tidak mampu). Setelah menemui target, pembawa acara memberikan uang sebesar 10-15 Juta Rupiah dan memberikan waktu kepada target untuk membelanjakan seluruh uang tersebut dalam waktu 30-33 menit, serta target akan didampingi oleh polisi dan time keeper.

Permasalahan dalam topik ini adalah bagaimana menghabiskan uang yang diberikan seoptimal mungkin dalam rentang waktu yang telah ditentukan, dalam persoalan ini akan digunakan knapsack untuk mencari rute dalam membeli barang dan memastikan waktu yang digunakan akan kurang atau sama dengan waktu yang telah ditentukan, selain itu akan digunakan algoritma *branch and bound* untuk meningkatkan efisiensi dengan cara mematikan simpul yang tidak mengarah ke hasil.

II. Landasan Teori

A. *Graf*



Gambar 2. Graf
sumber :
https://upload.wikimedia.org/wikipedia/commons/thumb/5/5b/6n-graf.svg/333px-6n-graf.svg.png diakses pada 5 Mei 2021 pukul 19.00

Graf merupakan representasi citra atau diagram yang menggambarkan objek diskrit dan hubungan antar objek tersebut. Graf terdiri atas simpul dan sisi. Secara definisi matematis, graf (G) adalah

G = (V,E), dimana:

V = himpunan tidak kosong dari simpul-simpul

= {v1,v2,v3,...,vn}

E = himpunan sisi yang menghubungkan sepasang simpul

= {e1,e2,e3,...,en}.

B. *Permasalahan Knapsack*

Knapsack merupakan permasalahan yang cukup familiar dalam dunia ilmu komputer. Masalah ini merupakan suatu kejadian dimana ada sejumlah barang dengan berat tertentu dan nilai tertentu hendak dimasukkan ke dalam sebuah tas. Dimana sebuah tas memiliki batas berat yang dapat ditampung. Persoalan knapsack merupakan persoalan optimasi, dimana tujuannya adalah mencari nilai maksimal dari barang yang dimasukkan ke dalam tas, selama tidak melebihi batas berat dari tas tersebut.

Gambar 3. Knapsack problem
sumber:
https://www.codesdope.com/staticroot/images/algorithm/knap1.png
diakses pada 7 Mei 2021 pukul 22.30.

Permasalahan 1/0 knapsack merupakan permasalahan knapsack dimana suatu objek dapat dimasukkan ke dalam knapsack (1) atau tidak sama sekali (0) (tidak bisa sebagian). Secara matematis permasalahan 1/0 knapsack dapat ditulis sebagai berikut:

$$Maksimasi\ F = \sum_{i=0}^{n} P_i X_i$$

Serta memiliki kendala (*constraint*)

$$\sum_{i=0}^{n} W_i X_i \leq K$$

Dimana, X_i adalah 0 atau 1 , dan i adalah 1,2,3,…,n.

C. *Branch and Bound*

Algoritma *branch and bound* merupakan jenis algoritma yang digunakan untuk mengatasi persoalan optimasi. Persoalan optimasi sendiri adalah persoalan dimana kita meminimalkan atau memaksimalkan fungsi objektif yang tidak melanggar batasan dari persoalan tersebut. Algoritma *branch and bound* merupakan penggabungan antara BFS (*Breadth First Search*) dan least cost search, dimana pada algoritma BFS murni, untuk melakukan ekspansi simpul berikutnya dilakukan berdasarkan urutan pembangkitanya atau dengan menggunakan metode FIFO (*First In First Out*). Pada *branch and bound* setiap simpul akan diberikan nilai biaya sebelum dibangkitkan, sehingga ekspansi simpul dilakukan berdasarkan biaya yang paling kecil atau besar (sesuai dengan optimasi yang diharapkan).

Pada algoritma *branch and bound* memiliki dua prinsip, yaitu:

1. *Branch and bound* melakukan percabangan untuk memecah masalah besar menjadi masalah yang lebih kecil lagi secara rekursif dan melakukan perhitungan nilai terbaik setiap memecahkan masalah. Proses ini dinamakan dengan *branching*.
2. Kemudian algoritma ini akan melakukan pencatatan terhadap nilai minimal / maksimal sebagai bound dari setiap branching, sehingga jawaban yang berada diluar *range* dari *bound* akan dihapus dan tidak akan diekspansi, berikut juga bound yang tidak akan mengarah ke hasil optimal juga akan dihapus dan tidak diekspansi.

D. *Fungsi Pembatas*

Fungsi pembatas merupakan salah satu fungsi yang berguna dalam algoritma *branch and bound* untuk memangkas simpul yang sekiranya tidak mengarah ke solusi atau tidak mengarah ke solusi yang optimum. Kriteria fungsi pembatas adalah untuk memangkas:

1. Nilai simpul tidak lebih baik dari nilai terbaik saat ini.
2. Simpul sudah tidak merepresentasikan solusi yang layak, karena ada solusi yang dilanggar.
3. Solusi pada simpul tersebut hanya terdiri dari satu titik atau tidak ada pilihan lainya. Kemudian kita bandingkan nilai fungsi objektif dengan solusi terbaik saat ini, lalu ambil yang terbaik.

E. *Acara Uang Kaget*

Uang kaget merupakan acara realitas yang ditayangkan di televisi swasta di Indonesia dengan konsep ada pembawa acara atau biasa disebut dengan Mr Money yang mencari target seseorang (biasanya orang kurang mampu). Setelah menemukan target, Mr Money memberikan sejumlah uang kepada target untuk dibelanjakan dalam waktu 30-33 menit dengan didampingi oleh polisi dan time-keeper. Apabila waktu tersebut sudah habis dan masih ada sisa uang, maka target harus mengembalikan sisa uang kepada Mr Money.

III. Impelementasi dan Pembahasan

Algoritma *branch and bound* akan diterapkan pada kasus mencari rute untuk mendapat keuntungan optimum dari acara Uang Kaget dengan cara mempersiapkan sampel data berupa 5 titik yang dapat dikunjungi ditambah satu titik asal yang telah dilengkapi dengan keuntungan dari masing-masing titik serta waktu yang dibutuhkan untuk mengunjungi antar titik. pada persoalan ini akan dimodelkan sebagai *knapsack problem* dengan waktu antar titik sebagai berat (W_i) dan keuntungan dari tiap titik sebagai keuntungan knapsack (P_i).

A. Persiapan dan Sampel Data

Mengingat keterbatasan waktu, ruang, dan pengetahuan untuk melakukan sampling secara riil, maka penulis akan menggunakan sampel data seperti dibawah, dan dengan asumsi setiap mengunjungi toko i membeli barang senilai n secara utuh atau tidak sama sekali (prinsip 1/0 knapsack) dan mengunjungi toko i hanya sekali saja.

Untuk sampel diberikan waktu 30 menit untuk membelanjakan uang sebesar Rp15.000.000 dengan daftar toko serta waktu yang dibutuhkan untuk mengunjungi toko tersebut seperti pada graf dan tabel dibawah ini.

Untuk memperpekecil lingkup permasalahan serta menghasilkan hasil yang optimal, maka total keuntungan dari seluruh simpul tidak akan lebih dari Rp15.000.000 sehingga tidak akan terjadi kasus dimana uang yang dibelanjakan lebih dari Rp15.000.000, karena dalam makalah ini lebih diutamakan untuk mencari rute yang menghasilkan keuntungan optimum dengan memanfaatkan waktu yang ada.

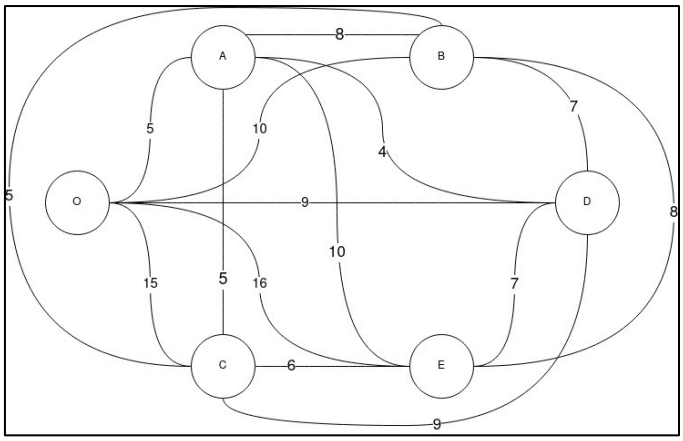
Simpul	Keuntungan (dalam ribuan rupiah)
O (titik start)	0
A	4500
B	3000
C	3500
D	1500
E	2500

Tabel 1. Daftar simpul dan keuntungan dari simpul tersebut.

Sedangkan untuk tabel yang menyatakan jarak dari Simpul i ke i+1 adalah seperti berikut (dalam satuan menit).

	O	A	B	C	D	E
O	-	5	10	15	9	16
A	5	-	8	5	4	10
B	10	8	-	5	7	8
C	15	5	5	-	9	6
D	9	4	7	9	-	7
E	16	10	8	6	7	-

Tabel 2. Jarak dari suatu simpul ke simpul tetangga (dalam menit).



Gambar 4. Representasi citra graf dari tabel 2

B. Pemodelan Knapsack

Pada persoalan ini knapsack akan dimodelkan dengan pohon ruang status, dimana cost dari setiap simpul pada pohon menyatakan batas atas dari solusi minimum. Simpul akar merupakan simpul O dimana merupakan awal perjalanan. Kemudian opsi ekspansi dari setiap simpul i adalah simpul yang belum dikunjungi oleh simpul i dan memiliki ketetanggan dengan simpul i. Dari opsi ekspansi, maka yang akan diekspansi merupakan simpul hidup dengan cost paling besar dan tidak melanggar fungsi pembatas.

Untuk menghitung cost atau batas atas dari simpul i dilakukan dengan cara menghitung total keuntungan yang telah dicapai (F) ditambah dengan perkalian sisa kapasitas knapsack (K - W) dengan rasio keuntungan per bobot objek yang tersisa berikutnya (Pi / Wi-1,i).

$$C(i) = F + (K - W)\frac{P_i}{W_{i-1,i}}$$

Untuk tabel item knapsack secara individu adalah sebagai berikut, dengan $E_{i-1,i-1}$ merupakan edge yang menghubungkan antar dua simpul, Pi merupakan keuntungan yang akan diperoleh (dalam ribu rupiah), dan Wi-1,i merupakan waktu (dalam menit) yang diperlukan untuk menuju simpul i dari simpul sebelum i

E_{i-1} - E_i	P_i	$W_{i-1,i}$	$\frac{P_i}{W_{i-1,i}}$
O – A	4500	5	900
O – B	3000	10	300
O – C	3500	15	233,3
O – D	1500	9	166,7
O – E	2500	16	156,3
A – B	3000	8	375
A – C	3500	5	700
A – D	1500	4	375
A – E	2500	10	250
B – C	3500	5	700

B – D	1500	7	214,3
B – E	2500	8	312,5
C – D	1500	9	166,7
C – E	2500	6	416,7
D – E	2500	7	357,1

Tabel 3. Daftar sisi dari graf beserta beratnya, keuntunganya, dan rasio keuntungan / berat.

C. *Penentuan Fungsi Pembatas*

Pengecekan waktu apakah melebihi 30 menit atau tidak, bila iya maka matikan simpul. Pengecekan waktu dapat dilakukan sebagai berikut:

$t_0 + t_i <= T$

Dimana:

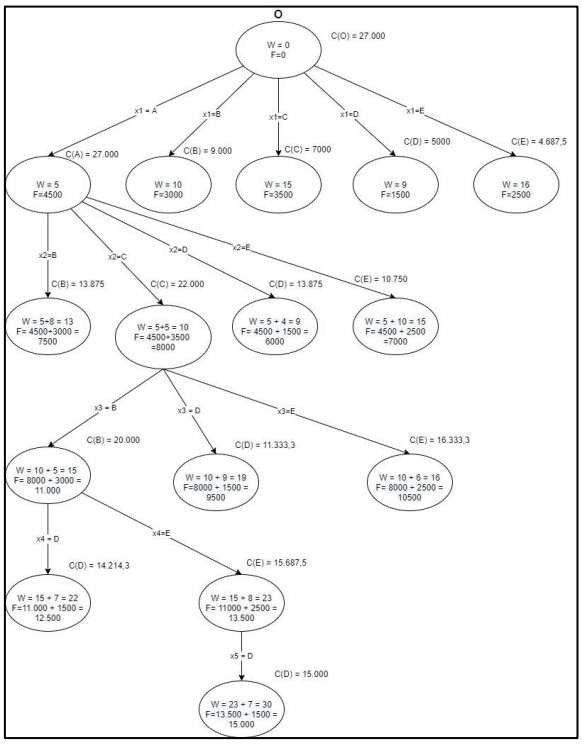
t_0 : waktu yang telah terpakai hingga saat ini

t_i : waktu yang dibutuhkan untuk menjangkau simpul terkait

T : waktu maksimal yang dibolehkan, dimana dalam hal ini adalah 30 menit

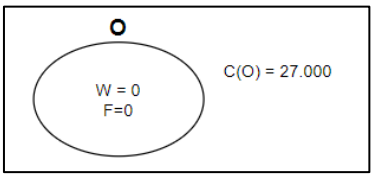
D. *Pohon Ruang Status*

Dengan menggunakan algoritma *branch and bound*, didapat Pohon solusi seperti berikut:



Gambar 5. Pohon solusi dari permasalahan mencari rute pembelian barang dengan menggunakan knapsack dan algoritma *branch and bound.*

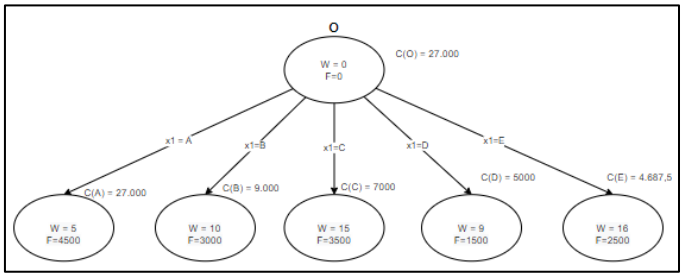
Apabila di-rincikan, maka seperti ini penjelasanya.



Gambar 6. Simpul awal dari knapsack.

Pada simpul awal didapat cost sebesar 27.000. yang berasal dari

C (O) = 0 + (30-0) * (900) = 27.000, dimana 900 didapatkan dari rasio antara keuntungan simpul yang bertetangga dengan O dibagi dengan waktu tempuh / berat sisi yang bersisian dengan O, dan didapat 900 yang berasal dari sisi O – A yang mana simpul A memiliki keuntungan 4500 dan sisi O – A memiliki berat / waktu tempuh 5 menit, sehingga menghasilkan 900.



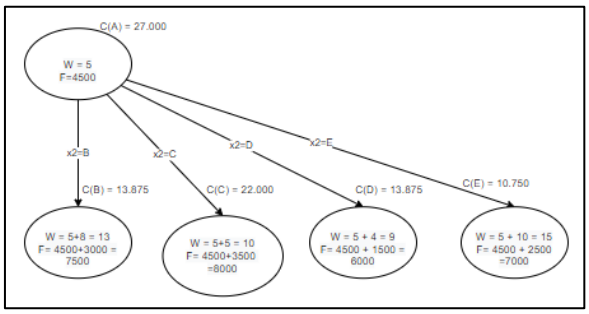
Gambar 7. Simpul ekspansi dari simpul awal.

Kemudian simpul awal (O) diekspansi, dengan cost masing-masing simpul ekspansi (*i*) seperti berikut:

i	$\mathbf{P}_i$	$\mathbf{W}_{O-i}$	$\mathbf{P}_i/\mathbf{W}_{O-i}$	**F**	**W**	**C(*i*)**
A	4.500	5	900	4.500	5	27.000
B	3.000	10	300	3.000	10	9.000
C	3.500	15	233,33	3.500	15	7.000
D	1.500	9	266,67	1.500	9	5.000
E	2.500	16	156,25	2.500	16	4.687,5

Tabel 4. Daftar cost masing – masing simpul ekspansi dari O

Sehingga kita pilih simpul A untuk diekspansi karena tidak melanggar fungsi pembatas dan memiliki nilai *cost* paling besar.



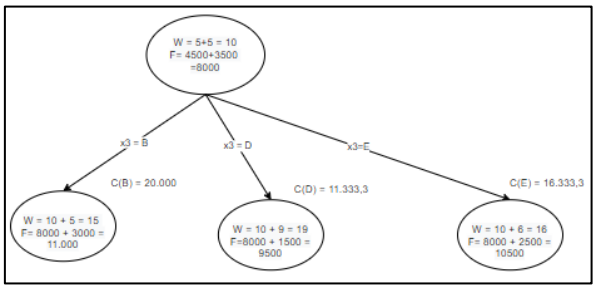
Gambar 8. Simpul ekspansi dari simpul A.

Kemudian simpul A diekspansi, dengan cost masing-masing simpul ekspansi (*i*) seperti berikut:

i	P_i	W_{A-i}	P_i/W_{A-i}	F	W	C(*i*)
B	3.000	8	375	7.500	13	13.875
C	3.500	5	700	8.000	10	22.000
D	1.500	4	375	6.000	9	13.875
E	2.500	10	250	7.000	15	10.750

Tabel 5. Daftar cost masing – masing simpul ekspansi dari A.

Sehingga kita pilih simpul C untuk diekspansi karena tidak melanggar fungsi pembatas dan memiliki nilai *cost* paling besar.



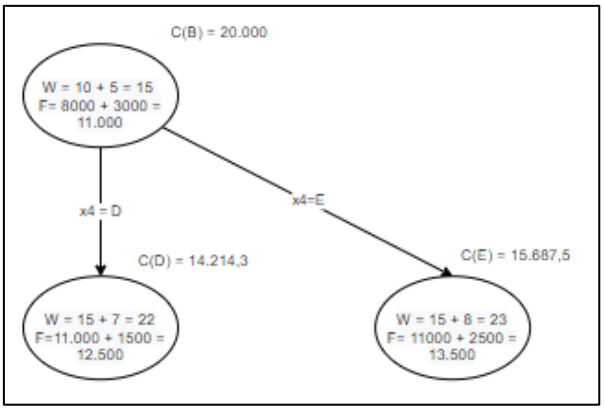
Gambar 9. Simpul ekspansi dari simpul C.

Kemudian simpul C diekspansi, dengan cost masing-masing simpul ekspansi (*i*) seperti berikut:

i	P_i	W_{C-i}	P_i/W_{C-i}	F	W	C(*i*)
B	3.000	5	600	11.000	15	20.000
D	1.500	9	116,67	9.500	19	11.333
E	2.500	6	416,67	10.500	16	16.333

Tabel 6. Daftar cost masing – masing simpul ekspansi dari C.

Sehingga kita pilih simpul B untuk diekspansi karena tidak melanggar fungsi pembatas dan memiliki nilai *cost* paling besar.



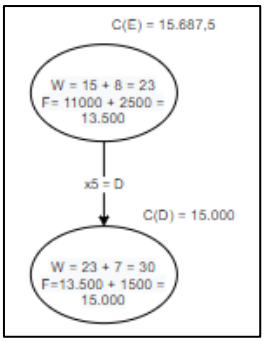
Gambar 10. Simpul ekspansi dari simpul B.

Kemudian simpul B diekspansi, dengan cost masing-masing simpul ekspansi (*i*) seperti berikut

i	P_i	W_{O-i}	P_i/W_{O-i}	F	W	C(*i*)
D	1.500	7	214,29	12.500	22	14.214,3
E	2.500	8	312,5	13.500	23	15.687,5

Tabel 7. Daftar cost masing – masing simpul ekspansi dari B

Sehingga kita pilih simpul E untuk diekspansi karena tidak melanggar fungsi pembatas dan memiliki nilai *cost* paling besar.



Gambar 11. Simpul ekspansi dari simpul D.

Kemudian simpul E diekspansi, dengan cost masing-masing simpul ekspansi (*i*) seperti berikut:

i	P_i	W_{O-i}	P_i/W_{O-i}	F	W	C(*i*)
E	2.500	7	214,29	15.000	30	15.000

Tabel 8. Daftar cost masing – masing simpul ekspansi dari E

Setelah mencari semua simpul, maka didapatkan hasil rute yang harus ditempuh, yaitu O-A-C-B-E-D dengan total waktu tempuh 30 menit dan keuntungan Rp15.000.000,00.

E. Pengujian dengan program

Kami juga telah melakukan implementasi dan pengujian untuk kasus ini dalam program python, untuk hasil dari eksperimen kami dalam bentuk program python adalah sebagai berikut:

```
Current Node = 0
List of candidate to expand
Node name = A
Cost of A = 27000.0
Violate constraint = False

Node name = B
Cost of B = 9000.0
Violate constraint = False

Node name = C
Cost of C = 7000.0
Violate constraint = False

Node name = D
Cost of D = 5000.0
Violate constraint = False

Node name = E
Cost of E = 4687.5
Violate constraint = False

The choosen next candidates is = A
=========================
```

Gambar 12. Hasil pengujian program bagian 1

```
Current Node = A
List of candidate to expand
Node name = B
Cost of B = 13875.0
Violate constraint = False

Node name = C
Cost of C = 22000.0
Violate constraint = False

Node name = D
Cost of D = 13875.0
Violate constraint = False

Node name = E
Cost of E = 10750.0
Violate constraint = False

The choosen next candidates is = C
```

Gambar 13. Hasil pengujian program bagian 2

```
Current Node = C
List of candidate to expand
Node name = B
Cost of B = 20000.0
Violate constraint = False

Node name = D
Cost of D = 11333.333333333334
Violate constraint = False

Node name = E
Cost of E = 16333.333333333332
Violate constraint = False

The choosen next candidates is = B
```

Gambar 14. Hasil pengujian program bagian 3

```
Current Node = E
List of candidate to expand
Node name = D
Cost of D = 15000.0
Violate constraint = False

The choosen next candidates is = D
========================
Rute ditempuh
0
A
C
B
E
D
time elapsed = 30 minutes
profit gained = 15000 Ribu Rupiah

Process finished with exit code 0
```

Gambar 15. Hasil pengujian program bagian 4

```
Current Node = B
List of candidate to expand
Node name = D
Cost of D = 14214.285714285714
Violate constraint = False

Node name = E
Cost of E = 15687.5
Violate constraint = False

The choosen next candidates is = E
```

Gambar 16. Hasil pengujian program bagian 5

IV. Kesimpulan dan Saran

Algoritma *branch and bound* dapat diimplementasikan dalam menyelesaikan pencarian rute dalam acara Uang Kaget. Dalam algoritma *branch and bound* juga dapat membantu mengarahkan ke hasil yang optimal dengan lebih sedikit ruang karena mematikan simpul yang tidak sesuai dan melakukan ekspansi dari simpul yang memiliki cost optimal, sehingga mampu menghemat waktu yang diperlukan untuk membuat keputusan. Untuk program ini akan berjalan hingga tidak ada simpul yang dapat dikunjungi lagi. Namun dalam penggunaan *branch and bound* tidak bisa menjamin memberikan hasil yang optimal apabila terjadi kasus dimana *profit* yang diberikan telah melampaui batas profit maksimal yang telah ditentukan dalam Uang Kaget, sehingga dalam makalah ini dibuat sedemikian rupa untuk jumlah semua simpul tidak melebihi ketentuan profit maksimal yang ditentukan. Diperlukan pengembangan lebih lanjut dengan model algoritma lain untuk bisa melakukan pengecekan terhadap fungsi pembatas waktu dan fungsi pembatas keuntungan maksimal.

Video Link at Youtube

https://youtu.be/V8EBv1_x-bw.

Acknowledgment

Pertama-tama penulis ingin berterima kasih dan mengucap syukur Alhamdulillah kepada Allah SWT, atas rahmat-Nya saya bisa menyelesaikan makalah ini. Penulis juga ingin mengucapkan terima kasih kepada Bapak Ir. Rila Mandala, M.Eng., Ph.D. selaku dosen Strategi Algoritma K-01 yang telah memberikan ilmu-ilmu yang bermanfaat yang membantu dalam menyelesaikan makalah ini, serta tidak lupa juga penulis ingin mengucapkan terimakasih kepada semua penulis dan pencipta referensi yang digunakan penulis sebagai acuan dari makalah ini.

Terakhir penulis juga ingin mengucapkan terimakasih kepada keluarga dan teman-teman penulis yang telah memberi dukungan dan semangat dalam mengerjakan tugas makalah ini.

Tanpa bantuan dan dukungan dari pihak-pihak diatas, penulis ragu mampu menyelesaikan makalah ini dengan baik.

REFERENCES

[1] Voloch, Nadav. (2017). Optimal paths of knapsack-set vertices on a weight-independent graph. WSEAS Transactions on Computers. 16. 163-17.

[2] R., Iwan F., and Djoko Soetarno. "Menentukan Lintasan Terpendek (Shortest Path) dengan 0/1 Knapsack Problem dan Pendekatan Algoritma Dynamic Programming." Creative Communication and Innovative Technology Journal, vol. 4, no. 3, 2011, pp. 293-315.

[3] R. Munir. 2016. Matematika Diskrit. Bandung: Departemen Teknik Informatika.

[4] https://www.baeldung.com/cs/branch-and-bound. Diakses pada 7 Mei 2021 pukul 05.00.

[5] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Branch-and-Bound-2021-Bagian1.pdf. Diakses pada 6 Mei 2021 pukul 21.30.

[6] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Branchand-Bound-2021-Bagian4.pdf. Diakses pada 6 Mei 2021 pukul 22.00.

[7] https://www.splashlearn.com/math-vocabulary/geometry/graph. Diakses pada 8 Mei 2021 pukul 20.10

[8] https://www.educative.io/edpresso/what-is-the-knapsack-problem. Diakses pada 5 Mei 2021 pukul 21.40

[9] https://ajaib.co.id/dapat-uang-kaget-harus-digunakan-untuk-apa-ya. Diakses pada 6 Mei 2021 pukul 23.00

[10] https://medium.com/@fikrira/bicara-layar-kaca-3-uang-kaget-2017-989d21af850d. Diakses pada 6 Mei 2021 pukul 22.40

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, 26 April 2021

Hafid Abi Daniswara -13519028