Penggunaan Algoritma Greedy untuk Meminimumkan Aliran Kas pada Graf Utang Piutang

Prama Legawa Halqavi / 13515132
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13515132@std.stei.itb.ac.id

Abstrak*—Algoritma Greedy banyak diterapkan untuk menyelesaikan persoalan-persoalan di kehidupan nyata khususnya menyelesaikan persoalan mencari maksimum minimum. Permasalahan dalam bentuk Graf saat ini memang banyak dipakai dalam beberapa hal antara lain dalam mencari jarak lintasan terpendek, mencari lintasan tercepat melalui beberapa pertibangan, mencari tur graf *travelling salesman problem* dengan lintasan terpendek dan masih banyak persoalan lainnya. Persoalan-persoalan tersebut dapat diselesaikan dengan strategi algoritma yang berbeda-beda satu sama lain. Salah satu persoalan yang representasinya dalam bentuk graf dan menggunakan strategi algoritma greedy adalah persoalan meminimumkan aliran kas atau *cashflow* utang piutang antar orang dengan representasi graf. Persoalan ini mengenai bagaimana membuat aliran kas yang rumit antarorang yang utang piutang agar menjadi lebih simpel aliran kas utang piutangnya sehingga akan lebih hemat ongkos dan biaya. Penyelesaian persoalan ini cukup berguna apabila kita utang terhadap si A namun B utang terhadap kita yang juga utang terhadap A. Dengan algoritma ini kita dapat membuat aliran kasnya menjadi: kita tidak punya utang lagi kepada si A lalu B utangnya terhadap si A bertambah sejumlah utang kita kepada si A. Itu adalah salah satu contoh sederhana bagaimana penyelesaian masalahnya.

***Keywords*—Graf, Greedy, Algoritma, Persoalan.**

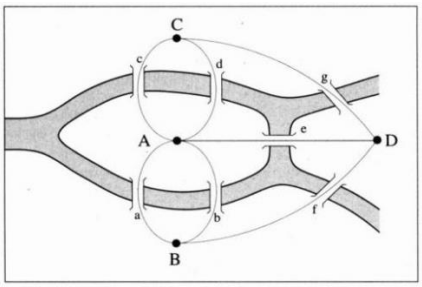
I. PENDAHULUAN

Dalam hidup ini pasti seseorang pernah mengalami naik turun dalam urusan keuangan atau perekonomian pribadinya. Salah satu langkah solusi yang diambil orang yang mendapat masalah seperti itu adalah dengan cara berutang kepada orang lain sehingga untuk sementara urusan keuangannya dapat terpenuhi sementara dan dia harus membayar kepada orang yang dipinjam. Permasalahan utang sangat banyak bahkan bisa memutuskan silaturahim antarmanusia dan mennimbulkan persengketaan. Di zaman dahulu ketika belum ada teknologi transfer bank, seseorang yang berutang harus membayar langsung secara *cash* kepada orang yang dipinjam. Hal ini tentu memerlukan ongkos dan biaya jika seseorang ingin membayar utang karena harus bertemu langsung dengan orang yang bersangkutan. Jika sekelompok orang saling hutang maka banyak orang harus mengeluarkan ongkos lebih untuk membayarnya. Masalah ini dapat diperkecil dengan meminimumkan aliran kas sehingga total ongkos dan biayanya lebih sedikit setelah aliran kasnya diminimumkan. Makalah ini akan membahas bagaimana meminimumkan aliran kas graf utang piutang dengan strategi algoritma greedy.

II. TEORI DASAR

2.1 Definisi Graf

Graf adalah hubungan yang menghubungkan objek-objek diskrit antara satu dengan yang lainnya[1]. Pada tahun 1836, Leonhard Euler membuktikan bahwa perjalanan di kota Konigsberg dengan syarat melalui setiap jembatan tepat satu kali, tidak dapat dilakukan. Dalam pembuktiannya, Euler menyederhanakan gambaran jembatan Konigsberg itu menjadi suatu diagram:

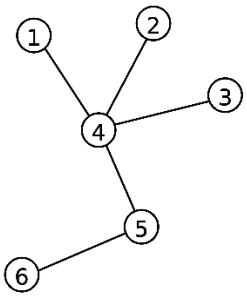


Gambar 2.1: Graf yang menggambarkan kota Konigsberg.

Sejak saat itulah penggunaan diagram semacam itu mulai populer dan teorinya dipakai sampai saat ini yang kita sebut sekarang sebagai graf. Sebagai contoh Graf G=(V,E) dalam hal ini:

V=himpunan yang tidak kosong dari simpul (*vertices*)
={v1,v2,...,vn}

E=himpunan dari sisi yang menghubungkan antarsimpul (*edges*)
={e1,e2,...,en}



Gambar 2.2: Graf G1

Pada Gambar 2.2 Graf G1 adalah Graf dengan
V={1,2,3,4,5,6}
E={(1,4),(2,4),(3,4),(4,5),(5,6)}

2.2 Jenis Graf

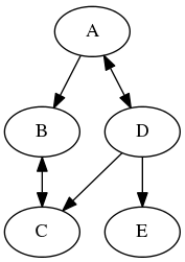
Graf dapat digolongkan menjadi 2 berdasarkan representasi arahnya yaitu:

1. Graf tak Berarah

Graf tak Berarah adalah graf yang sisinya tidak mempunyai orientasi arah[1]. Graf jenis ini dapat diaplikasikan untuk merepresentasikan rangkaian elektrik, rantai makanan pada suatu ekosistem, penggambaran ikatan molekul-molekul kimia. Contoh Graf tak berarah seperti yang ada pada Gambar 2.2.

2. Graf Berarah

Graf Berarah adalah graf yang sisinya mempunyai orientasi arah[1]. Arahnya merepresentasikan hubungan yang ada antara 2 simpul. Graf jenis ini cukup banyak aplikasinya di dalam kehidupan nyata contohnya Persoalan Pedagang Keliling (*Travelling Salesman Problem*) yang setiap sisinya akan diberikan bobot untuk menentukan rute dengan bobot paling minimum yang dapat ditempuh. Contoh Graf Berarah seperti yang ada pada Gambar 2.3 di bawah ini.

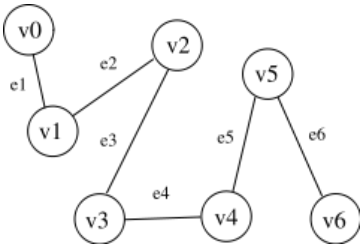


Gambar 2.3: Contoh Graf Berarah.

Pada Gambar 2.3 graf A menunjukan hubungan ke B namun B tidak menunjukkan hubungan ke A karena arah sisinya dari A ke B. Sebuah graf berarah dikatakan Graf Terhubung Kuat (*Strongly Connected graph*) apabila setiap simpul pada graf tersebut mempunyai sisi yang masuk yang masuk dan sisi yang keluar. Graf berarah dapat digunakan untuk merepresentasikan hubungan utang piutang antarmanusia.

2.3 Graf Ketetanggaan

Dua buah simpul dikatakan bertetangga apabila keduanya terhubung dengan sebuah sisi.

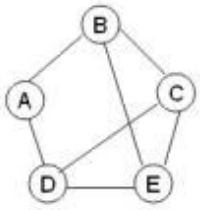


Gambar 2.4: Contoh Graf dengan hubungan ketetanggan.

Tinjau Graf pada gambar di atas. Simpul v1 bertetangga dengan simpul v0 dan v2 karena simpul v1 mempunyai sisi yang menghubungkan antara simpul v1 dengan simpul v0 dan simpul v1 dengan simpul v2. Begitu juga simpul v3 bertetangga dengan simpul v2 dan v4. Simpul v5 tidak bertetangga dengan simpul v0, v1, v2 dan v3 karena tidak ada sisi yang menghubungkan antara simpul v5 dengan simpu-simpul tersebut.

Sebuah Graf juga dapat dinyatakan dalam matriks yang disebut matriks ketetanggaan. Matriks ketetanggaan merepresentasikan hubungan antarsimpul.

A = [a.i.j],
a.i.j =
- 1, jika simpul i dan j bertetangga
- 0, jika simpul i dan j tidak bertetangga



	A 0	B 1	C 2	D 3	E 4
A 0	0	1	0	1	0
B 1	1	0	1	0	1
C 2	0	1	0	1	1
D 3	1	0	1	0	1
E 4	0	1	1	1	0

Gambar 2.5: Graf dengan Representasi Matriks.

Graf tidak berarah yang direpresentasikan dalam matriks bertetanggaan akan memiliki elemen a.i.j=a.j.i sebagai contoh pada gambar di atas simpul B dengan E bertetangga, pada matriks elemen 4.1 = 1.4 = 1 hal ini menyatakan bahwa simpul B bertetangga dengan simpul E, begitu juga dengan simpul E bertetangga dengan simpul B.

Graf dalam representasi matriks bertetanggaan ini dapat digunakan dalam merepresentasikan hubungan antarmanusia yang berutang.

2.4 Algoritma

Algoritma adalah susunan langkah-langkah dalam penyelesaian masalah dalam bentuk kata-kata atau kalimat yang tersusun secara logis dan matematis. Susunan langkahnya jelas dan pasti yang bila diikuti akan mentransformasikan data input menjadi output yang berupa informasi.

Algoritma merupakan fondasi untuk menyelesaikan permasalahan logika secara terstruktur, efektif dan efisien salah satunya penggunaannya yaitu dalam pembuatan program komputer yang akan menyelesaikan suatu masalah. Menurut Donald E. Knuth, suatu algoritma harus memiliki awal dan akhir, harus berhenti setelah mengerjakan serangkaian tugas atau denga kata lain suatu algoritma harus memiliki langkah yang terbatas. Setiap langkah dalam algoritma harus didefinisikan dengan tepat, sehingga tidak memiliki arti ganda dan tidak membingungkan atau ambigu. Setiap algoritma harus memiliki input (kondisi awal) dan output (kondisi akhir). Algoritma harus efektif dan bila diikuti dengan benar langkah-

langkahnya maka akan menyelesaika suatu permasalahan.

Sebelum menyelesaikan masalah dengan program kita harus menganalisis masalah dengan mengidentifikasi informasi data-data masukan dan keluaran pemecahan masalah. Setelah itu masalah yang sudah teridentifikasi akan dituangkan ke dalam algoritma. Kemudian algoritma yang sudah tertuang akan dikonversi kedalam kode program (bahasa pemrograman) yang dapat dimengerti oleh komputer. Program yang sudah ada kemudian diuji apakah pemecahan masalah menggunakan program tersebut sudah berhasil sesuai dengan yang diharapkan atau masih terjadi kesalahana-kesalahan.

2.5 Strategi Algoritma

Sebelum mengenal pengertian strategi algoritma alangkah baiknya kita mengetahui arti per katanya terlebih dahulu. Strategi adalah suatu siasat untuk mencapai sasaran yang diinginkan, sedangkan Algoritma adalah urutan langkah-langkah yang digunakan dalam memecahkan suatu masalah. Jadi dapat disimpulkan bahwa strategi algoritma adalahrencana yang dibuat secara tahapan demi tahapan untuk mencapai saasaran yang diinginkan. Secara umum strategi algoritma dibagi menjadi:

1. Strategi Solusi Langsung (*Direct Solution Strategies*)
   - Brute Force
   - Greedy
2. Strategi Berbasis Pencarian pada ruang status (*State – Space Base Strategies*)
   - BFS (Breadth First Search)
   - DFS(Deep First Search)
   - Backtracking
   - Branch and Bound
3. Strategi Solusi Atas Bawah (*Top – Down Solution Strategies*)
   - Divide and Conquer
   - Transform and Conquer
   - Increase and Conquer
4. Strategi solusi Bawah – Atas (*Bottom – Up Solution Strategies*)
   - Dynnamic Programming
5. Strategi Implementasi Graf dan Pohon
   - Dijkstra
   - Floyd Warshall
   - Prim
   - Kruskal

Pada permasalahan meminimumkan graf utang piutang kita akan menggunakan algoritma dengan strategi solusi langsung (*Direct Solution Strategies*) lebih tepatnya algoritma greedy.

2.6 Algoritma Greedy

Algoritma greedy adalah metode yang paling populer dalam memecahkan persoalan optimasi. Apa itu persoalan optimasi? Persoalan optimasi adalah persoalan yang membutuhkan solusi yang optimal baik itu solusi maksimum atau solusi minimum. Salah satu contohnya yaitu mencari jalan dari titik A ke B yang merupakan jalan terpendek. Walaupun algoritma greedy yang paling populer dalam menyelesaikan persoalan optimum namun persoalan ini masih dapat diselesaikan dengan strategi lain. Nama greedy sendiri diambil dari bahasa inggris yang berarti rakus, tamak atau serakah. Prinsip algoritma greedy adalah "take what you can get now!".

Beberapa contoh persoalan yang dapat diselesaikan dengan algoritma greedy antara lain memilih beberapa jenis investasi, memilih jalur tersingkat, memilih jurusan di perguruan tinggi, bermain kartu remi. Algoritma greedy membentuk solusi langkah per langkah (*step by step*). Pada setiap langkat terdapat banyak pilihan yang perlu dieksplorasi. Oleh karena itu, pada setiap langkah harus dibuat keputusan terbaik dalam menentukan pilihan. Jika keputusan sudah diambil pada langkah sebelumnya maka keputusan itu tidak dapat diubah lagi pada langkah-langkah selanjutnya.

Pada setiap langkahnya kita mengambil solusi optimum lokal diantara beberapa pilihan yang ada pada langkah itu dan berharap solusi itu akan berujung pada solusi global. Pada setiap langkahnya algoritma greedy melakukan:

- Mengambil pilihan terbaik pada saat itu tanpa memikirkan konsekuensi apa yang akan terjadi kedepannya. Ini sesuai dengan prinsip algoritma greedy yaitu "take what you can get now!"
- Berharap bahwa solusi yang diambil akan menghasilkan solusi optimum global.

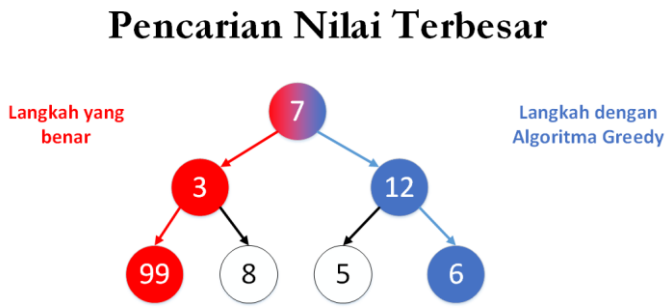
Algoritma greedy memiliki elemen-elemen yang menyusun:

1. Himpunan kandidiat, C
   Langkah-langkah yang menghasilkan alternatif solusi
2. Himpunan solusi, S
   Langkah-langkah himpunan kandidat yang memiliki nilai maksimum/minimum pada himpunan kandidat
3. Fungsi seleksi (*selection function*)
   Fungsi yang menyeleksi suatu alternatif solusi pada himpunan kandidat sehingga menjadi bagian dari himpunan solusi
4. Fungsi kelayakan (*feasible*)
   Fungsi yang mengecek apakah suatu langkah adalah layak yang berarti solusi yang tidak melanggar batasan-batasan tertentu
5. Fungsi obyektif
   Fungsi yang memaksimumkan/meminimumkan nilai solusi

Algoritma greedy akan melakukan pencarian sebuah himpunan bagaian, S dari himpunan kandidat, C; dalam hal ini S harus memenuhi beberapa kritesia yang ditentukan yaitu menyatakan solusi dan S dioptimisasi oleh fungsi obyektif.

Untuk beberapa kasus, algoritma greedy tidak menghasilkan solusi optimum karena suatu optimum global belum tentu merupakan solusi optimum tetapi hanya solusi sub-optimum atau pseudo-optimum dengan alasan algoritma greedy tidak beroperasi secara menyeluruh terhadap alternatif solusi yang ada sehingga bisa saja solusi optimum yang sebenarnya terdapat di solusi alternatif tersebut, terdapat beberapa fungsi seleksi yang berbeda sehingga kita harus memilih fungsi yang tepat jika

kita ingin algoritma menghasilkan solusi optimal.



Gambar 2.6: Contoh Graf dengan hubungan ketetanggan.

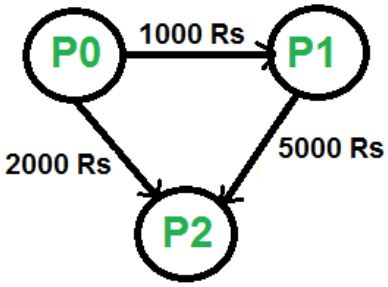
Gambar diatas adalah salah satu contoh pemilihan solusi dengan algoritma greedy yang hanya menghasilkan solusi yang pseudo-optimum. Pada percabangan 7 memang 12 lebih besar daripada 3, namun pemilihan simpul 12 berujung pada daun 6 yang masih lebih kecil dari pada daun yang akan dihasilkan oleh simpul 3 yaitu 99.

Jika kita memang tidak mutlak memerlukan solusi maka algoritma greedy adalah algoritma yang tepat untuk aproksimasi solusi optimal dibandingkan menggunakan algoritma yang lebih rumit.

III. PEMBAHASAN

A. Meminimumkan Aliran Kas Diantara Beberapa Orang yang Saling Berutang

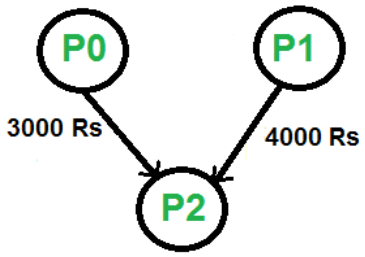
Diketahui beberapa orang yang yang diantaranya harus membayar utang dan ada yang hanya menerima bayaran seperti berikut:



Gambar 3.1: Graf Utang Piutang [4]

Gambar 3.1 adalah graf yang merepresentasikan utang yang harus dibayar tiap orang. P0 harus membayar utang 1000 kepada P1 dan 2000 kepada P2. P1 harus membayar 5000 kepada P2 dan dia akan mendapat bayaran dari P0 sebanyak 1000. Sedangkan P2 akan mendapat bayaran dari P0 2000 dan P1 5000.

Persoalan tersebut jika dioptimumkan akan menghasilkan graf seperti gambar di bawah



Gambar 3.2: Graf Solusi Minimum Utang Piutang [4]

Solusi optimumnya adalah P0 harus membayar 3000 kepada P2 dan P1 harus membayar 4000 kepada P2 tetapi P0 tidak harus membayar kepada P1. Dengan begitu P0 tidak perlu mengeluarkan ongkos untuk bertemu dan memabayar utang kepada P1.

Ide algoritma greedynya disini pertama untuk setiap orang kita menentukan selisih antara total uang yang dipinjamkan dengan total uang yang dipinjam orang tersebut, kita sebut saja nilai ini sebagai net. Jika net semua orang sudah diperoleh maka selanjutnya kita menentukan siapa yang akan membayar siapa secara *step by step* yang akan berujung pada diperolehnya graf minimum aliran kas.

Untuk menentukan siapa yang akan membayar siapa, kita akan menentukan dua orang yang mempunyai nilai maksimum dan minimum dari *array of net* yang sudah kita hitung tadi. Jika sesorang lebih banyak berutang daripada dipiutang maka nilai net akan bernilai negatif, sebaliknya jika seseorang lebih banyak dipiutang daripada berutang maka nilai netnya akan bernilai positif. Oleh karena itu orang yang memiliki nilai maksimum arraynya adalah orang yang lebih banyak dipiutang daripada berutang dan yang memiliki nilai minimum adalah orang yang lebih banyak berutang. Selanjutnya nilai maksimum array net kita sebut MaxCredit dan minimum array kita sebut MaxDebt.

Setelah kita dapat nilai maksimum dan minimumnya, kita tentukan dari kedua nilai ini yang mana yang lebih kecil nilainya setelah kedua nilai ini dimutlakkan (setelah dimutlakkan kedua nilai akan positif). Pemutlakan ini tidak akan berdampak pada nilai array net, pemutlakan ini hanya untuk menentukan yang mana yang lebih kecil. Sebut saja nilai minimum dari kedua nilai ini adalah x dan x adalah bilangan positif. Kemudian nilai MaxDebt akan ditambah dengan nilai x, nilai MaxCredit akan dikurangi dengan nilai x. Nilai array net juga akan berubah. Kita sudah mendapat salah satu solusi yaitu orang dengan MaxDebt membayar sejumlah x kepada orang dengan MaxCredit.

Untuk mencari solusi selanjutnya kita perlu merekursifkan langkah dari mencari nilai maksimum dan minimum. Basisnya adalah jika MaxCredit dan MaxDebt bernilai nol maka rekursi akan berhenti. Adapun algoritma di atas jika ditulis langkah per langkahnya sebagai berikut:

1. Hitung nilai net untuk setiap orang lalu simpan nilainya di dalam *array of net* dimana nilai net adalah selisih dari total uang yang dipinjamkan ke orang lain dengan total uang yang dipinjam dari orang lain.
2. Cari nilai MaxDebt dan MaxCredit dari *array of net.*

Simpan juga indeks net nya sebut saja IdxMaxDebt dan IdxMaxCredit.

3. Jika kedua nilai tersebut bernilai nol, maka solusi sudah ketemu dan keluar dari rekursi, jika belum maka lanjut ke langkah selanjutnya.
4. Mutlakkan kedua nilai tersebut dan cari nilai minimum dari kedua nilai yang sudah dimutlakkan tadi. Sebut saja x adalah nilai minimum dari kedua nilai tadi.
5. Perbaharui nilai net dengan mengurangkan nilai net[IdxMaxCredit] dengan x dan menjumlahkan nilai net[IdxMaxDebt] dengan x.
6. Salah satu solusi sudah didapat yaitu orang IdxMaxDebt membayar sejumlah x kepada orang IdxMaxCredit.
7. Lakukan rekursi dengan kembali ke langkah 2.

Algoritma ini menghasilkan kompleksitas waktu sebesar $O(n^2)$ dimana n adalah jumlah orang yang ada di dalam graf.

B. Penyelesaian Persoalan Meminimumkan Graf Utang Piutang

Berikut adalah beberapa penyelesaikan persoalan meminimumkan graf utang piutang.

- Diketahui gambar graf seperti pada Gambar 3.1

Kita akan mengubah graf menjadi matriks yang merepresentasikan graf ketetanggaan untuk mempermudah komputasi program seperti dibawah ini:

0	1000	2000
0	0	5000
0	0	0

1. Kita akan menentukan terlebih dahulu net untuk setiap orang:

Orang ke 0: net = -1000 – 2000 = -3000
Orang ke 1: net = 1000 – 5000 = -4000
Orang ke 2: net = 2000 + 5000 = 7000

Orang ke-	0	1	2
net	-3000	-4000	7000

2. Selanjutnya kita mencari nilai MaxDebt dan MaxCredit pada *array of net* dengan melakukan pencarian nilai maksimum dan minimum.

Didapat MaxDebt = -4000 dan MaxCredit = 7000
Dimana IdxMaxDebt = 1 dan IdxMaxCredit = 2

3. Kedua nilai tidak bernilai nol maka masih terus lanjut ke langkah selanjutnya.

4. Kedua nilai dimutlakkan terlebih dahulu kemudian dipilih yang minimum (x) antara kedua nilai MaxDebt dan MaxCredit. Maka didapat nilai x = 4000

5. Nilai net akan diperbaharui dengan mengurangkan nilai net[IdxMaxCredit] dengan x dan menjumlahkan nilai net[IdxMaxDebt] dengan x.

net[1] = net[1] + 4000
net[2] = net[2] - 4000
Sehingga arraynya menjadi seperti berikut

Orang ke-	0	1	2
net	-3000	0	3000

6. Didapat solusi Orang 1 harus membayar 4000 ke Orang 2. Solusinya dimasukkan ke dalam matriks solusi.

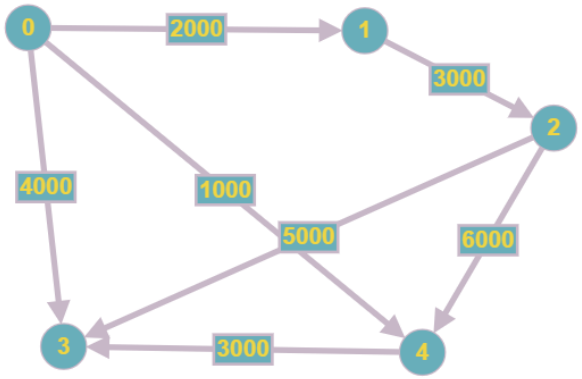
7. Lakukan rekursi dengan kembali ke langkah 2 sampai semua nilai pada elemen net bernilai nol.

Dengan begitu matriks solusi yang didapat adalah:

0	0	3000
0	0	4000
0	0	0

Gambar graf solusinya dapat dilihat pada Gambar 3.2. Kita akan mencoba contoh kasus lain yang sedikit lebih kompleks.

- Diketahui gambar graf seperti pada gambar dibawah



Gambar 3.3: Graf Utang Piutang

Kita akan mengubah graf tersebut menjadi matriks yang merepresentasikan graf ketetanggaan untuk mempermudah komputasi program seperti dibawah ini:

0	2000	0	4000	1000
0	0	3000	0	0
0	0	0	5000	6000
0	0	0	0	0
0	0	0	3000	0

1. Kita akan menentukan terlebih dahulu net untuk setiap orang:

Orang ke 0: net = -1000 – 2000 – 4000 = -7000
Orang ke 1: net = 2000 – 3000 = -1000
Orang ke 2: net = 3000 – 5000 – 6000 = -8000

Orang ke 3: net = 4000 + 5000 + 3000 = 12000
Orang ke 4: net = 6000 + 1000 – 3000 = 4000

Orang ke-	0	1	2	3	4
net	-7000	-1000	-8000	12000	4000

2. Selanjutnya kita mencari nilai MaxDebt dan MaxCredit pada *array of net* dengan melakukan pencarian nilai maksimum dan minimum.
Didapat MaxDebt = -8000 dan MaxCredit = 12000
Dimana IdxMaxDebt = 2 dan IdxMaxCredit = 3

3. Karena kedua nilai tidak nol maka lanjut ke langkah selanjutnya.

4. Kedua nilai dimutlakkan terlebih dahulu kemudian dipilih yang minimum (x) antara kedua nilai MaxDebt dan MaxCredit. Maka didapat nilai x = 8000

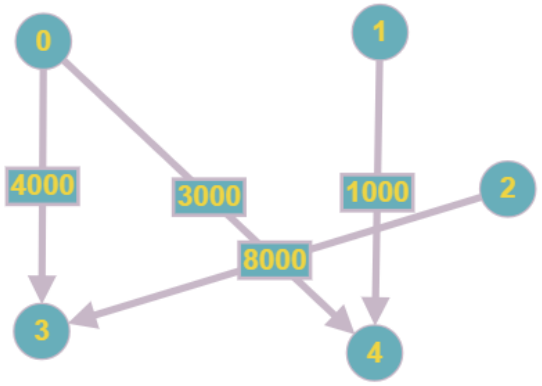
5. Nilai net akan diperbaharui dengan mengurangkan nilai net[IdxMaxCredit] dengan x dan menjumlahkan nilai net[IdxMaxDebt] dengan x.
net[2] = net[2] + 8000
net[3] = net[3] - 8000

6. Didapat solusi Orang 2 harus membayar 8000 ke Orang 3. Solusinya dimasukkan ke dalam matriks solusi.

7. Lakukan rekursi dengan kembali ke langkah 2 sampai semua nilai pada elemen net bernilai nol.
Dengan begitu matriks solusi yang didapat adalah:

0	0	0	4000	3000
0	0	0	0	1000
0	0	0	8000	0
0	0	0	0	0
0	0	0	0	0

Jika direpresentasikan dalam gambar graf, graf solusinya akan seperti:



Gambar 3.4: Graf Utang Piutang

C. Hasil Eksperimen

Berikut adalah hasil eksperimen berupa pengujian eksekusi program:

1. Pengujian kasus pertama yaitu graf pada Gambar 3.1. Diberikan input file eksternal

```
3
0 1000 2000
0 0    5000
0 0    0
```

Pada baris pertama menunjukkan jumlah orang yang ada pada graf. Di bawahnya merupakan representasi graf dalam matriks ketetanggaan. Input tersebut akan menghasilkan output seperti pada gambar dibawah ini

```
Jumlah orang : 3
Matriks graf aliran kas awal :
0 1000 2000
0 0 5000
0 0 0
Orang 0 harus membayar 3000 kepada orang 2
Orang 1 harus membayar 4000 kepada orang 2
Matriks graf aliran kas akhir :
0 0 3000
0 0 4000
0 0 0
```

Gambar 3.5: Hasil Eksekusi Kasus 1

Program berhasil mengevaluasi persoalan kasus 1 sesuai dengan yang diharapkan.

2. Pengujian kasus pertama yaitu graf pada Gambar 3.3. Diberikan input file eksternal

```
5
0 2000 0    4000 1000
0 0    3000 0    0
0 0    0    5000 6000
0 0    0    0    0
0 0    0    3000 0
```

Format input sama seperti kasus 1. Input tersebut akan menghasilkan output seperti pada gambar di bawah ini

```
Jumlah orang : 5
Matriks graf aliran kas awal :
0 2000 0 4000 1000
0 0 3000 0 0
0 0 0 5000 6000
0 0 0 0 0
0 0 0 3000 0
Orang 0 harus membayar 4000 kepada orang 3
Orang 0 harus membayar 3000 kepada orang 4
Orang 1 harus membayar 1000 kepada orang 4
Orang 2 harus membayar 8000 kepada orang 3
Matriks graf aliran kas akhir :
0 0 0 4000 3000
0 0 0 0 1000
0 0 0 8000 0
0 0 0 0 0
0 0 0 0 0
```

Gambar 3.6: Hasil Eksekusi Kasus 2

Program berhasil mengevaluasi persoalan kasus 2 sesuai dengan yang diharapkan.

IV. KESIMPULAN

Algoritma Greedy yang merupakan materi mata kuliah Strategi Algoritma penerapannya sangat banyak dalam kehidupan sehari-hari. Beberapa penerapannya mulai dari pemilihan jalur tercepat dalam suatu graf, *knapsack problem* sampai penyelesaian persoalan meminimumkan aliran kas pada graf utang piutang. Begitu juga Teori Graf yang merupakan materi dari mata kuliah Matematika Diskrit penerapannya sangat banyak dalam kehidupan sehari – hari.

Dengan algoritma greedy kita dapat mereduksi biaya dan ongkos diantara orang-orang yang saling berutang apabila harus bertemu langsung dengan orangnya dengan meminimumkan aliran kasnya. Algoritma ini juga dapat dipakai apabila ada kasus dimana sekumpulan orang makan di restoran lalu ketika hendak membayar, seseorang menalangi dahulu pembayarannya, akibatnya kumpulan orang tadi berutang pada orang yang membayar tersebut dan terjadilah persoalan aliran kas utang piutang.

V. UCAPAN TERIMA KASIH

Terima kasih kepada Allah swt. yang senantiasa memberikan rahmat serta karunianya sehingga penulis dapat menyelesaikan makalah ini tepat waktu. Terima kasih juga kepada Dr. Masayu Leylia Khodra, S.T., M.T., Dr. Nur Ulfa Maulidevi, S.T., dan Dr. Ir. Rinaldi Munir, M.T. selaku dosen mata kuliah Strategi Algoritma yang telah mengajarkan ilmu mata kuliah Strategi Algoritma ini. Terima kasih kepada orang tua penulis dan semua pihak yang telah membantu penulis dalam penyelesaian makalah ini baik dalam bentuk materil maupun moril. Tentunya dalam proses pembuatan makalah ini masih terdapat kesalahan yang tidak disengaja. Oleh karena itu, penulis sangat terbuka dalam menerima kritik, saran, serta komentar dari berbagai pihak. Semoga dengan adanya makalah ini dapat bermanfaat bagi banyak orang dan dapat digunakan sebagaimana mestinya.

REFERENSI

[1] Munir, Rinaldi. *Matematiaka Diskrit* (Edisi Kedua). Bandung: Informatika Bandung, 2003

[2] Halqavi, Prama Legawa. *Implementasi Pewarnaan Graf dan Kombinatorial pada Permainan Sudoku*. Bandung: Prodi Teknik Informatika ITB, 2016

[3] Munir, Rinaldi. *Strategi Algoritmik.* Bandung: Informatika Bandung, 2006

[4] http://www.geeksforgeeks.org/minimize-cash-flow-among-given-set-friends-borrowed-money/ tanggal akses : 15 Mei 2017 Pukul 01:00 WIB

[5] http://staff.unipdu.ac.id/nufan/2013/09/24/pengantar-algoritma-dan-pemrograman/ Tanggal akses : 16 Mei 2017 Pukul 10:00 WIB

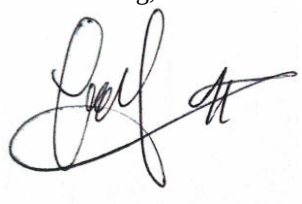
[6] https://anonimnon40.wordpress.com/2014/11/08/algoritma-dan-pemrograman-2/ Tanggal akses : 16 Mei 2017 Pukul 10:00 WIB

[7] https://www.it-jurnal.com/pengertian-algoritma-greedy/ Tanggal akses : 16 Mei 2017 Pukul 10:00 WIB

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, 18 Mei 2017



Prama Legawa Halqavi
13515132