Penerapan *Greedy* pada "Jalan Jalan Di Bandung Yuk! V1.71"

Wiko Putrawan (13509066)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13509066@std.stei.itb.ac.id

***Abstrak*—Bandung, atau yang sering disebut dengan kota Kembang, adalah ibukota provinsi Jawa Barat. Bandung lahir pada tanggal 25 September 1810. Kota yang mempunyai julukan Parijs van Java ini mempunyai luas 167,67 km^2. Dengan luas ini, masyarakat Bandung memerlukan sarana infrastruktur dan akomodasi yang memadai. Salah satu kebijakan pemerintah adalah menentukan trayek angkutan kota. Sampai tahun 2011, Bandung memiliki 67 trayek angkutan kota yang meliputi hampir seluruh wilayah utama kota. Banyaknya trayek angkutan umum yang tersedia pastinya akan memudahkan masyarakat untuk menuju lokasi yang diinginkan. Namun, banyaknya trayek yang tersedia juga menyebabkan masyarakat bingung untuk memilih jalur angkutan umum yang efisien. Salah satu cara untuk menyelesaikan masalah ini adalah dengan menggunakan algoritma *greedy*. Algortima *greedy* membentuk solusi langkah per langkah yang mana pada setiap langkah, dibuat keputusan yang terbaik dalam menentukan pilihan. *Greedy* yang digunakan dalam kasus ini adalah *greedy by* jarak dan *greedy by* jumlah angkutan kota. Diharapkan sistem ini dapat membantu masyarakat dalam mengambil keputusan untuk memilih trayek angkutan kota dari tempat asalnya hingga sampai ke tempat tujuan.**

***Kata Kunci*—Bandung, angkutan kota, *greedy*, himpunan kandidat, himpunan solusi, fungsi seleksi, fungsi kelayakan, fungsi obyektif.**

I. PENDAHULUAN

Jalan Jalan di Bandung Yuk! V1.71 adalah sebuah perangkat lunak yang dibangun oleh D.A. Nugroho bersama team dari ELCEE pada tahun 2006. Perangkat lunak ini bertujuan untuk membantu *user* untuk menetukan angkutan kota apa saja yang harus dipilih jika ingin pergi dari suatu titik awal menuju titik akhir. Kota yang menjadi objek perangkat lunak ini adalah kota Bandung.

Kota Bandung merupakan kota metropolitan terbesar di Jawa Barat sekaligus menjadi ibukota provinsi tersebut. Kota ini terletak 140 km sebelah tenggara Jakarta, dan merupakan kota terbesar ketiga di Indonesia setelah Jakarta dan Surabaya menurut jumlah penduduk. Di kota bersejarah ini, berdiri sebuah perguruan tinggi teknik pertama di Indonesia (*Technische Hoogeschool*, sekarang ITB).

Kota Bandung dikelilingi oleh pegunungan, sehingga bentuk morfologi wilayahnya bagaikan sebuah mangkok raksasa. Sampai pada tahun 2004, kondisi transportasi jalan di kota Bandung masih buruk dengan tingginya tingkat kemacetan serta ruas jalan yang tidak memadai, termasuk masalah parkir dan tingginya polusi udara. Permasalahan ini muncul karena beberapa faktor diantaranya pengelolaan transportasi oleh pemerintah setempat yang tidak maksimal seperti rendahnya koordinasi antara instansi yang terkait, ketidakjelasan wewenang setiap instansi, dan kurangnya sumber daya manusia, serta ditambah tidak lengkapnya peraturan pendukung.

Untuk transportasi di dalam kota, masyarakat Bandung biasanya menggunakan angkutan kota atau yang lebih akrab disebut angkot. Selain itu, bus kota dan taksi juga menjadi alat transportasi di kota ini. Sedangkan sebagai terminal bus antarkota dan provinsi di kota ini adalah terminal Leuwipanjang untuk rute barat dan terminal Cicaheum untuk rute timur.

Angkutan kota adalah sebuah moda transportasi perkotaan yang merujuk kepada kendaraan umum dengan rute yang sudah ditentukan. Tidak seperti bus yang mempunyai halte sebagai tempat perhentian yang sudah ditentukan, angkutan kota dapat berhenti untuk menaikkan atau menurunkan penumpang di mana saja.

II. DASAR TEORI

Algoritma *greedy* merupakan metode yang paling populer untuk memecahkan persoalan optimasi. Persoalan optimasi (*optimization problems*) adalah persoalan mencari solusi optimum. Hanya ada dua macam persoalan optimasi yaitu:

1. Maksimasi (*maximization*).
2. Minimasi (*minimization*).

Greedy dalam bahasa Indonesia berarti rakus, tamak, atau loba. Prinsip *greedy* adalah "*take what you can get now!*". Algoritma *greedy* membentuk solusi langkah per langkah (*step by step*). Pada setiap langkah, terdapat banyak pilihan yang perlu dieksplorasi. Oleh karena itu, pada setiap langkah harus dibuat keputusan yang terbaik dalam menentukan pilihan. Pada setiap langkah, kita membuat pilihan optimum lokal (*local optimum*) dengan harapan bahwa langkah sisanya mengarah ke solusi

optimum global (*global optimum*).

Algoritma *greedy* adalah algoritma yang memecahkan masalah langkah per langkah. Pada setiap langkah:

1. mengambil pilihan yang terbaik yang dapat diperoleh pada saat itu tanpa memperhatikan konsekuensi ke depan (prinsip "*take what you can get now!*")
2. berharap bahwa dengan memilih optimum lokal pada setiap langkah akan berakhir dengan optimum global

Elemen-elemen algoritma *greedy* dijelaskan sebagai berikut:

1. Himpunan kandidat, C.
2. Himpunan solusi, S.
3. Fungsi seleksi (*selection function*).
4. Fungsi kelayakan (*feasible*).
5. Fungsi obyektif.

Dengan kata lain, algortima *greedy* melibatkan pencarian sebuah himpunan bagian, S, dari himpunan kandidat, C, yang dalam hal ini, S harus memenuhi beberapa kriteria yang ditentukan, yaitu menyatakan suatu solusi dan S dioptimisasi oleh fungsi obyektif.

Skema umum algortima greedy dapat dilihat sebagai berikut:

```
function greedy(input C: himpunan_kandidat)→ himpunan_kandidat
{ Mengembalikan solusi dari persoalan optimasi dengan algoritma greedy
  Masukan: himpunan kandidat C
  Keluaran: himpunan solusi yang bertipe himpunan_kandidat
}
Deklarasi
    x : kandidat
    S : himpunan_kandidat

Algoritma:
    S ← {}   { inisialisasi S dengan kosong }
    while (not SOLUSI(S)) and (C ≠ {} ) do
      x ← SELEKSI(C)          { pilih sebuah kandidat dari C}
      C ← C - {x}             { elemen himpunan kandidat berkurang satu }
      if LAYAK(S ∪ {x}) then
          S ← S ∪ {x}
      endif
    endwhile
    {SOLUSI(S) or C = {} }

    if SOLUSI(S) then
       return S
    else
       write('tidak ada solusi')
    endif
```

Pada akhir setiap lelaran, solusi yang terbentuk adalah optimum lokal. Pada akhir kalang while-do diperoleh optimum global. Harus diperhatikan bahwa optimum global belum tentu merupakan solusi optimum (terbaik), tetapi *sub-optimum* atau *pseudo-optimum*. Alasannya karena:

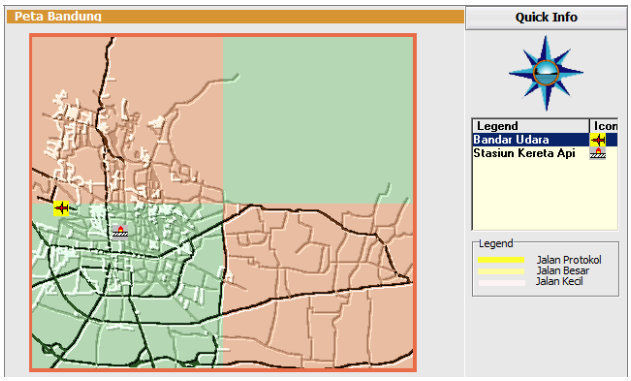
1. algortima *greedy* tidak beroperasi secara menyeluruh terhadap semua alternatif solusi yang ada (sebagaimana pada metode *exhaustive search*)
2. terdapat beberapa fungsi SELEKSI yang berbeda, sehingga kita harus memilih fungsi yang tepat jika kita ingin algoritma menghasilkan solusi optimal

Jadi, pada sebagian masalah algoritma *greedy* tidak selalu berhasil memberikan solusi yang optimal. Jika jawaban terbaik mutlak tidak diperlukan, maka algoritma *greedy* sering berguna untuk menghasilkan solusi hampiran (*approximation*) daripada menggunakan algortima yang lebih rumit untuk menghasilkan solusi yang eksak. Bila algortima *greedy* optimum, maka keoptimalannya itu dapat dibuktikan secara matematis.

III. ISI

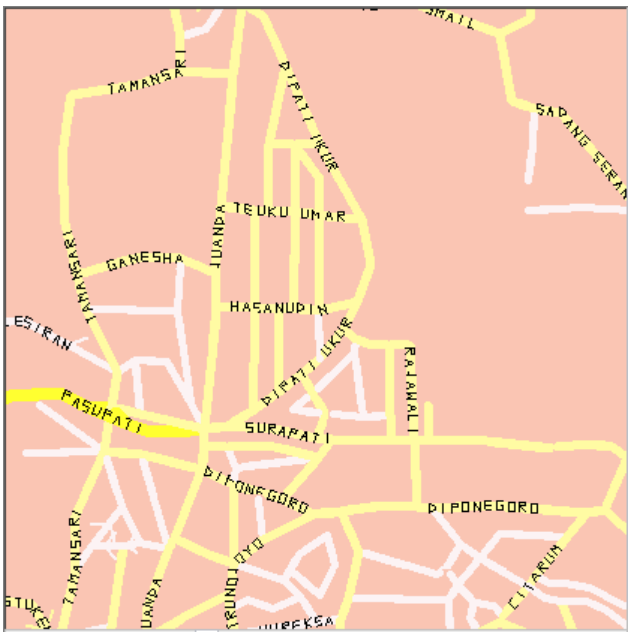
Berikut ini akan dijelaskan kontribusi algoritma *greedy* dalam penentuan trayek angkutan kota di Bandung. Penjelasan ini akan dibantu oleh perangkat lunak "Jalan Jalan di Bandung Yuk V1.71" yang dibangun oleh D.A. Nugroho pada tahun 2006.

Awalnya, peta kota Bandung "dibersihkan" sehingga hanya tersisa ruas jalan dan nama jalannya. Peta tersebut dapat ditambahi legenda seperti bandar udara, statiun kereta api, jalan protokol, jalan besar, dan jalan kecil.



Gambar 1 Peta Bandung

Peta di atas merupakan graf yang bagian jalannya dapat dipilih. Untuk menentukan angkutan kota yang bisa dipilih, pengguna terlebih dahulu memilih titik *start* pada peta yang telah diperbesar kemudian memilih titik *end* pada jalan yang dituju. Misalkan kita memilih lokasi di sekitar kampus ITB, maka akan didapatkan peta yang telah diperbesar sebagai berikut.



Gambar 2 Daerah di sekitar kampus ITB

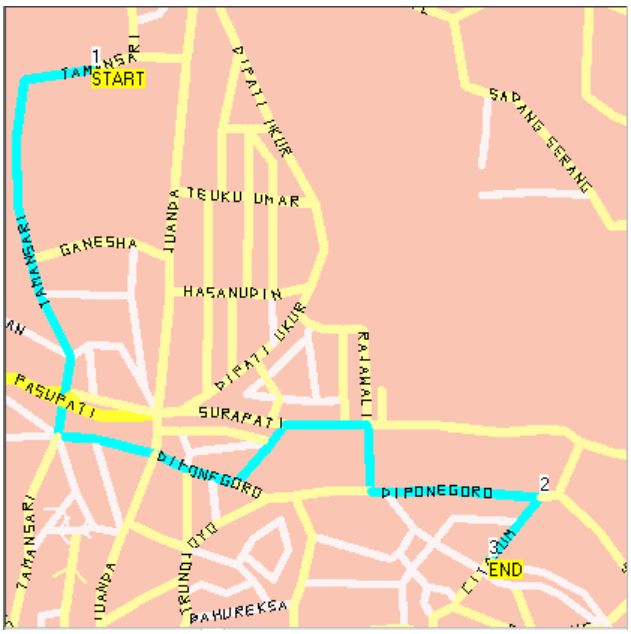
Misalkan pengguna ingin pergi dari jalan Tamansari yang berada di sudut kiri atas peta menuju jalan Citarum yang berada di sudut kanan bawah peta. Berdasarkan pilihan tersebut, sistem membentuk pohon berbobot yang mempunyai simpul berupa tiap persimpangan yang menghubungkan titik *start* dengan titik *end*. Tiap simpul dihubungkan oleh sisi yang mempunyai bobot berupa trayek angkutan kota yang melewati persimpangan yang dimaksud.

Analisis pertama berdasarkan *greedy by* jarak terdekat, yaitu memilih jalur angkutan kota yang menempuh jarak minimal untuk sampai ke titik *end*. Dari titik *start* tersebut, dicari semua jalur angkutan kota yang melewati titik itu. Dari jalur angkutan kota tersebut dipilih satu jalur angkutan kota yang melewati titik *end* atau yang melewati persimpangan yang pailng dekat dengan titik end.

Menurut basis data jalur angkutan kota yang ada pada sistem tersebut, jalur angkutan kota yang melewati persimpangan paling dekat dengan titik *end* adalah jalur Ledeng – Cicaheum.

Gambar 3 Jalur pertama yang dipilih untuk kasus pertama

Pada titik 2, algoritma *greedy* yang telah dijalankan diulang kembali berdasarkan titik start yang baru yaitu titik 2 dan akan menuju titik *end*. Berdasarkan basis data jalur angkutan kota yang ada pada sistem tersebut, jalur angkutan kota yang melewati titik end dari titik 2 adalah jalur Sadang Serang – St.Hall, yang ditujukkan oleh garis biru yang menghubungkan titik 2 dan titik 3.

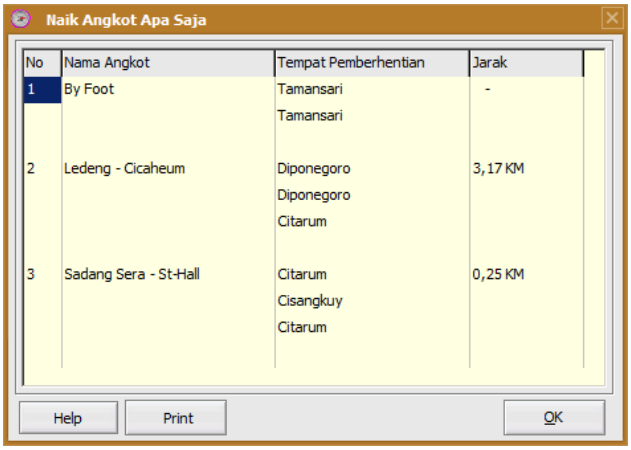


Gambar 4 Jalur angkutan kota yang dipilih untuk kasus pertama

Jadi dapat disimpulkan jika kita ingin pergi dari titik start ke titik end menggunakan angkutan kota, maka angkutan kota yang kita pilih adalah jurusan Ledeng – Cicaheum dilanjutkan dengan jurusan Sadang Serang – St. Hall. Dari contoh kasus ini, elemen-elemen *greedy* dapat diklasifikasikan sebagai berikut:

- Himpunan kandidat: himpunan semua jalur angkutan kota yang ada di Bandung.
- Himpunan solusi: jalur angkutan kota dari titik *start* menuju titik *end*.
- Fungsi seleksi: pilihlah jalur angkutan kota yang melewati titik *end* atau yang melewati persimpangan yang paling dekat dengan titik *end*.
- Fungsi layak: memeriksa apakah jalur angkutan kota yang dipilih melewati titik *end* atau tidak.
- Fungsi obyektif: jarak yang ditempuh oleh jalur angkutan kota yang dipilih minimum.

Hasil pencarian angkutan kota ditampilkan dalam antarmuka sebagai berikut:



Gambar 5 Hasil pencarian jalur angkutan kota untuk kasus pertama

Dari antarmuka di atas terlihat bahwa sistem menampilkan prosedur untuk mencapai tempat tujuan mulai dari nama angkutan kota yang digunakan, tempat pemberhentian, dan jarak yang ditempuh.

Selain greedy by jarak tempuh, untuk masalah ini bisa juga diaplikasikan menggunakan *greedy by* jumlah angkutan kota. Di mana sistem akan menampilkan sejumlah minimal jalur angkutan kota yang bisa dipilih untuk mencapai titik tujuan.

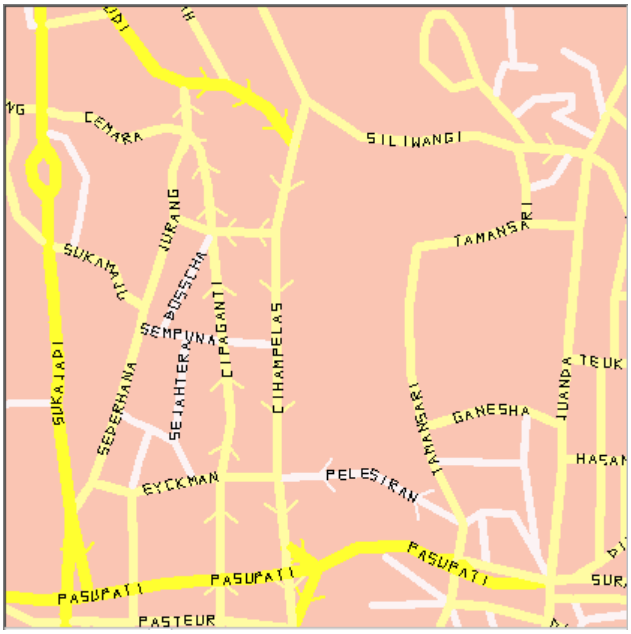
Pertama-tama sistem akan melihat apakah ada jalur angkutan kota yang melewati titik start dan titik end secara bersamaan. Jika tidak ada, sistem akan memilih jalur angkutan kota yang melewati persimpangan yang terdekat dengan titik tujuan. Setelah itu, sistem akan mengiterasi semua persimpangan yang telah dilalui oleh jalur angkutan kota yang dipilih dari persimpangan yang terdekat dengan titik *end* sampai persimpangan yang paling jauh. Dari tiap persimpangan itu dicari lagi apakah ada jalur angkutan kota yang melewati titik tersebut dan titik *end* secara bersamaan. Algortima di atas diulangi hingga telah ditemukan jalur angkutan umum yang melewati titik *end*.

Dari contoh kasus ini, elemen-elemen *greedy* dapat diklasifikasikan sebagai berikut:

- Himpunan kandidat: himpunan semua jalur angkutan kota yang ada di Bandung.
- Himpunan solusi: jalur angkutan kota dari titik *start* menuju titik *end*.
- Fungsi seleksi: pilihlah jalur angkutan kota yang melewati titik *end* atau yang melewati persimpangan yang paling dekat dengan titik *end*.
- Fungsi layak: memeriksa apakah jalur angkutan kota yang dipilih melewati titik *end* atau tidak.
- Fungsi obyektif: jumlah angkutan kota yang digunakan dari titik *start* menuju titik *end* minimum.

Terlihat bahwa algortima *greedy by* jumlah angkot memiliki tingkat komputasi yang lebih rumit dari *greedy by* jarak terdekat karena algortima ini mengiterasi semua kemungkinan jalur angkutan umum di setiap persimpangan yang dilalui dengan tujuan agar jumlah jalur angkutan umum yang digunakan minimum. Karena rumitnya algoritma ini, *programmer* tidak mengimplementasikannya ke dalam sistem.

Misalkan kita memilih lokasi di sekitar kampus ITB dan mengarah ke jalan Cihampelas, maka akan didapatkan peta yang telah diperbesar sebagai berikut.

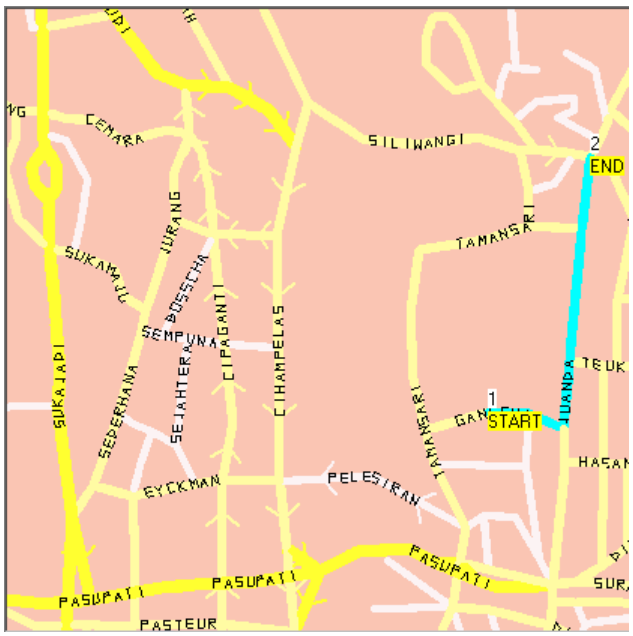


Gambar 6 Peta daerah pencarian yang baru

Misalkan pengguna ingin pergi dari jalan Ganesha atau gerbang depan ITB menuju gedung Cihampelas Walk yang berada di jalan Cihampelas. Berdasarkan pilihan tersebut, sistem membentuk pohon berbobot yang mempunyai simpul berupa tiap persimpangan yang menghubungkan titik *start* dengan titik *end*. Tiap simpul dihubungkan oleh sisi yang mempunyai bobot berupa trayek angkutan kota yang melewati persimpangan yang dimaksud.

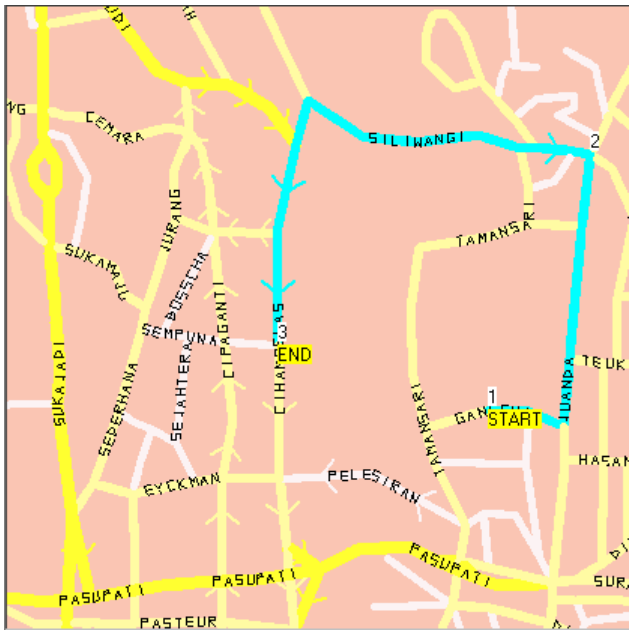
Analisis yang digunakan berdasarkan *greedy by* jarak terdekat, yaitu memilih jalur angkutan kota yang menempuh jarak minimal untuk sampai ke titik *end*. Dari titik *start* tersebut, dicari semua jalur angkutan kota yang melewati titik itu. Dari jalur angkutan kota tersebut dipilih satu jalur angkutan kota yang melewati titik *end* atau yang melewati persimpangan yang pailng dekat dengan titik end.

Menurut basis data jalur angkutan kota yang ada pada sistem tersebut, jalur angkutan kota yang melewati persimpangan paling dekat dengan titik *end* adalah jalur Caringin – Sadang Serang.



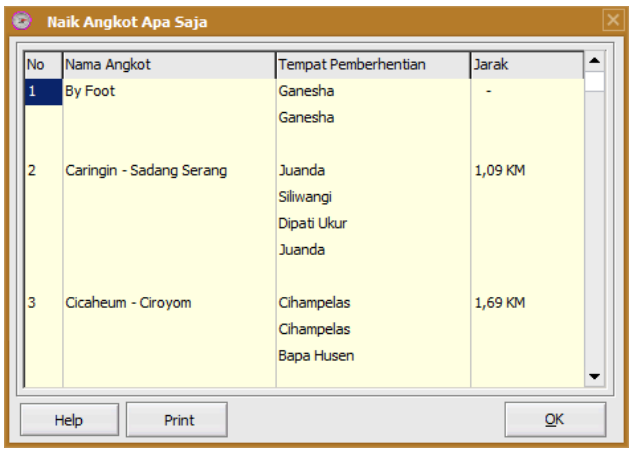
Gambar 7 Jalan pertama yang dipilih untuk kasus ke dua

Pada titik 2, algoritma *greedy* yang telah dijalankan diulang kembali berdasarkan titik start yang baru yaitu titik 2 dan akan menuju titik *end*. Berdasarkan basis data jalur angkutan kota yang ada pada sistem tersebut, jalur angkutan kota yang melewati titik end dari titik 2 adalah jalur Cicaheum – Ciroyom, yang ditujukkan oleh garis biru yang menghubungkan titik 2 dan titik 3.



Gambar 8 Jalur angkutan kota yang dipilih untuk kasus ke dua

Jadi dapat disimpulkan jika kita ingin pergi dari titik start ke titik end menggunakan angkutan kota, maka angkutan kota yang kita pilih adalah jurusan Caringin – Sadang Serang dilanjutkan dengan jurusan Cicaheum – Ciroyom. Hasil pencarian angkutan kota ditampilkan dalam antarmuka sebagai berikut:



Gambar 9 Hasil pencarian jalur angkutan kota untuk kasus kedua

Dari antarmuka di atas terlihat bahwa sistem menampilkan prosedur untuk mencapai tempat tujuan mulai dari nama angkutan kota yang digunakan, tempat pemberhentian, dan jarak yang ditempuh.

Walaupun *greedy by* jumlah angkutan kota tidak diimplementasikan dalam sistem ini, bukan berarti *greedy* ini tidak lebih optimal dari dari *greedy by* jarak.

IV. KESIMPULAN

- Algoritma *greedy* merupakan metode yang paling populer untuk memecahkan persoalan optimasi.
- Algortima *greedy* dapat digunakan untuk membantu memecahkan masalah pengambilan keputusan untuk mengambil trayek angkutan kota dari tempat asal menuju tempat yang ingin kita tuju.
- Salah satu *greedy* yang bisa digunakan untuk masalah ini adalah *greedy by* jarak dan *greedy by* jumlah angkutan kota.
- Algoritma *greedy* belum menjawab persoalan ini secara optimal.

V. TERIMA KASIH

Penulis mengucapkan terima kasih kepada Ir. Rinaldi Munir, M.T., dosen pembimbing IF3051 Strategi Algoritma Program Studi Teknik Informatika Institut Teknologi Bandung dan pihak-pihak yang turut membantu dan mendukung penulis dalam menyelesaikan makalah ini.

REFERENCES

http://id.wikipedia.org/wiki/Kota_Bandung
diakses tanggal 7 Desember 2011 pukul 16.03 WIB
http://www.bandung.go.id/?fa=pemerintah.detail&id=326
diakses tanggal 8 Desember 2011 pukul 16.05 WIB
Munir, Rinaldi. 2003. *Diktat Kuliah IF3051 Strategi Algoritma*. Bandung: Program Studi Teknik Informatika.

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, 9 Desember 2011

Wiko Putrawan (13509066)