# Aplikasi Teori Graf untuk Pencarian Rute Angkutan Kota Terdekat untuk Tempat-tempat di Bandung

**Luqman Abdul Mushawwir – NIM: 13507029**
Jurusan Teknik Informatika ITB, Jalan Ganeca 10, Bandung, email: if17029@students.if.itb.ac.id

***Abstract*** *– Makalah ini membahas masalah pendatang yang seringkali sulit menemukan angkutan kota untuk berjalan-jalan ataupun mengunjungi suatu tempat di Bandung. Untuk mengatasi masalah ini, digunakan graf dan metoda pathfinding untuk mencari jalan terdekat atau termurah dari duatu tempat ke tempat lain.*

***Kata Kunci****: Graf, angkutan kota, Bandung, pathfinding, turis, jalan.*

## 1. PENDAHULUAN

Angkutan kota (angkot) merupakan angkutan umum yang paling banyak ada di kota Bandung. Dalam satu ruas jalan saja, bisa ada berbagai angkot dengan berbagai jurusan. Untuk mencapai suatu tujuan saja, seseorang dapat memilih banyak angkot.

Setiap angkot memiliki tarif yang berbeda-beda, juga jalur tempuh yang berbeda-beda. Bagi orang yang sudah lama di Bandung, dalam menentukan angkot yang akan dinaiki tentu saja mudah. Namun, bagi orang-orang pendatang dan turis yang berkunjung, tentu saja sangat sulit untuk menemukan angkot yang langsung menuju tujuan mereka dengan cepat dan murah.

Sebenarnya banyak cara untuk menemukan rute yang cepat dan murah dengan angkot dari suatu tempat ke tempat lain. Namun, dengan graf berbobot, akan menjadi lebih mudah dalam pengimplementasian ke program, sehingga akan lebih mudah dalam pemakaian, terutama bagi orang-orang yang tidak tahu Bandung.

## 2. TEORI GRAF

Graf dapat didefinisikan sebagai himpunan tidak kosong antara pasangan simpul-simpul dan sisi-sisi yang menghubungkan sepasang simpul. Himpunan simpul tidak boleh kosong, sedangkan himpunan sisi boleh kosong. Jadi suatu titik juga bisa disebut suatu graf. Graf yang hanya terdiri dari satu buah simpul tanpa sebuah sisi pun disebut graf trivial.

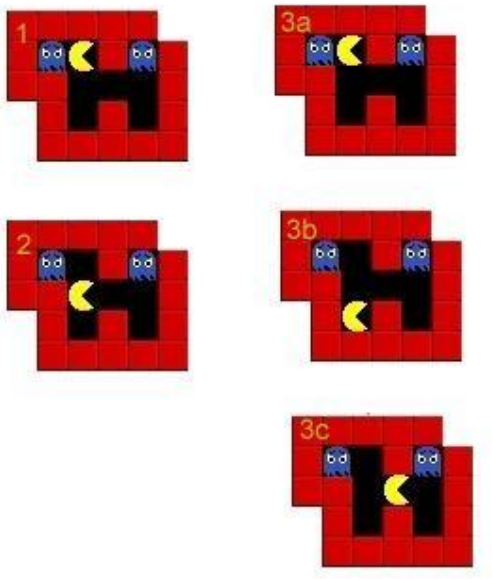
Pada permasalahan makalah ini, graf yang dipakai adalah graf berarah. Graf berarah adalah graf yang setiap sisinya diberikan orientasi arah. Sisi-sisi yang berarah ini biasa disebut busur. Pada graf berarah, sisi (u,v) tidak sama dengan (v,u). Untuk busur (u,v), simpul u merupakan simpul asal dan v merupakan simpul terminal. Dalam makalah ini, graf berarah dipakai dalam menunjukkan arah jalur dari suatu jalan.

## 3. TEORI *PATHFINDING*

### 3.1 Dasar Algoritma A*

Pada dasarnya, *pathfinding* dilakukan untuk mencari jalan paling dekat yang dapat ditempuh dari suatu tempat ke tempat lain. Banyak cara yang dapat dilakukan untuk *pathfinding* ini. Algoritma yang akan dipakai untuk *pathfinding* pada masalah kali ini adalah algoritma A* (*A-star algorithm*).
Algoritma A* didasari oleh sebuah persamaan sebuah benda yang sedang berjalan dari suatu tempat ke tempat lain:
$F(s) = G(s) + H(s)$ ... (1)
Dengan F(s) adalah jarak dari tempat awal ke tempat akhir, G(s) adalah jarak yang sudah ditempuh oleh benda itu dari tempat awal, dan H(s) adalah jarak dari tempat benda itu berada ke tempat akhir.

Contoh mudahnya adalah seperti berikut:




Gambar 1: Contoh pathfinding.

Pada gambar 1, dapat dilihat Pacman yang mencari jalan terdekat dari kotak di pojok kiri atas ke pojok kanan bawah, dengan berbagai alternatif jalan tentu saja.

Sekarang, *pathfinding* yang dilakukan Pacman berdasarkan pada persamaan (1). Pada gambar 1,
F(s) = 4
G(s) = 0
H(s) = 4
Setelah bergerak ke gambar 2,
F(s) = 4
G(s) = 1
H(s) = 3
Setelah itu, Pacman akan mempunyai tiga alternatif jalan, yaitu 3a, 3b, dan 3c.

| Pada 3a, | Pada 3b, | Pada 3c, |
|---|---|---|
| F(s) = 6 | F(s) = 6 | F(s) = 4 |
| G(s) = 2 | G(s) = 2 | G(s) = 2 |
| H(s) = 4 | H(s) = 4 | H(s) = 2 |

Konsep *pathfinding* adalah mencari jalan terpendek ke tujuan, yaitu H(s) yang paling pendek. Jika dilihat dari data di atas, maka sudah jelas Pacman akan memilih ke jalan 3c.

Algoritman A* sendiri dilakukan dengan mencoba satu per satu jalan yang tersedia, dan membandingkan jarak yang terdekat ke tempat tujuan.

### 3.2 Penerapan Algoritma A*

Penerapan Algoritma A* dalam pemrograman sendiri adalah dengan *priority queue*, dengan variabel OPEN, yaitu jalan yang belum pernah dilewati, dan CLOSED, yaitu jalan yang sudah pernah dilewati. [3]

```
OPEN = priority queue containing
START
CLOSED = empty set
while lowest rank in OPEN is not the
GOAL:

  current = remove lowest rank item
     from OPEN

  add current to CLOSED

  for neighbors of current:

    cost = g(current) +
     movementcost(current, neighbor)

    if neighbor in OPEN and cost
     less than g(neighbor):

      remove neighbor from OPEN,
       because new path is better

    if neighbor in CLOSED and cost
     less than g(neighbor):

      remove neighbor from CLOSED

    if neighbor not in OPEN and
     neighbor not in CLOSED:

      set g(neighbor) to cost

      add neighbor to OPEN

      set priority queue rank to
       g(neighbor) + h(neighbor)

      set neighbor's parent to
       current

reconstruct reverse path from goal
to start
by following parent pointers
```

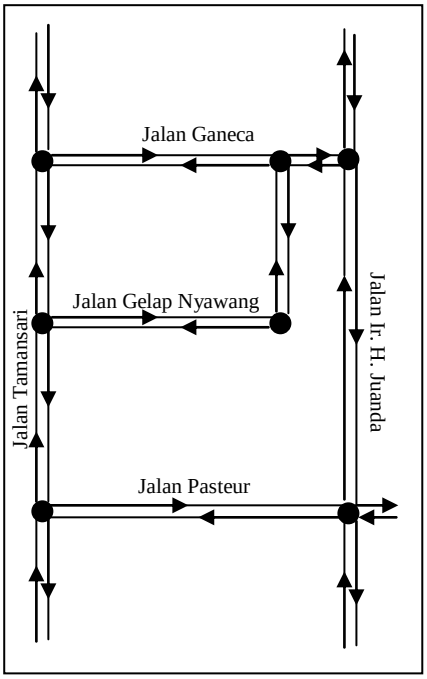
Intinya, algoritma A* berjalan dengan melihat semua kemungkinan jalan yang ada dari START sampai GOAL, dan melihat mana jalan yang paling cepat.

## 4. KOTA BANDUNG DAN RUTE JALAN

### 4.1 Graf Kota Bandung

Kota Bandung dapat digambarkan sebagai graf. Pada graf Kota Bandung tersebut, setiap persimpangan jalan digambarkan sebagai simpul, dan jalan-jalan yang menghubungkan simpul-simpul tersebut digambarkan sebagai sisi. Tentu saja jalan-jalan yang dimaksud adalah jalan besar yang dilewati oleh angkot saja, karena yang dibahas di sini adalah rute angkot.

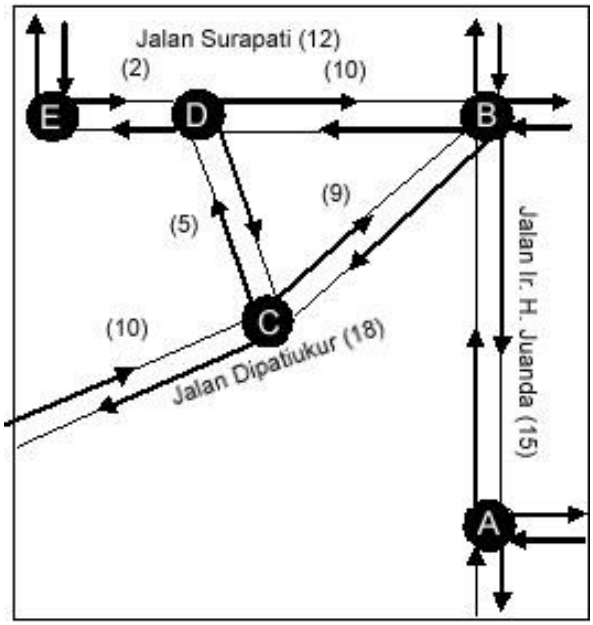
Dengan cara penggambaran seperti yang telah digambarkan di atas, kita dapat menggambarkan seluruh rute angkot yang ada di Kota Bandung. Begitu pula dengan semua tempat-tempat kunjungan dan wisata yang ada di Bandung, semua yang berada di jalan-jalan yang dilewati angkot. Adapun graf untuk menggambarkan Kota Bandung itu sendiri adalah graf berarah, yaitu menunjukkan arah jalan di sana. Contoh graf dari jalan di sekitar ITB:




Gambar 2: Graf jalan sekitar ITB

Dengan cara penggambaran seperti di atas, kita bisa mendapatkan graf Kota Bandung.

Dalam masalah pencarian rute angkot terdekat, tentu saja dibutuhkan data-data tentang jarak dari suatu simpul ke simpul lain. Oleh karena itu, graf Kota Bandung yang sebelumnya merupakan graf berarah, diubah menjadi graf berbobot yang bobotnya merupakan jarak dari satu simpul ke simpul lain. Contoh graf jalan sekitar jalan Suci:




Gambar 3: Graf jalan dengan bobot jarak

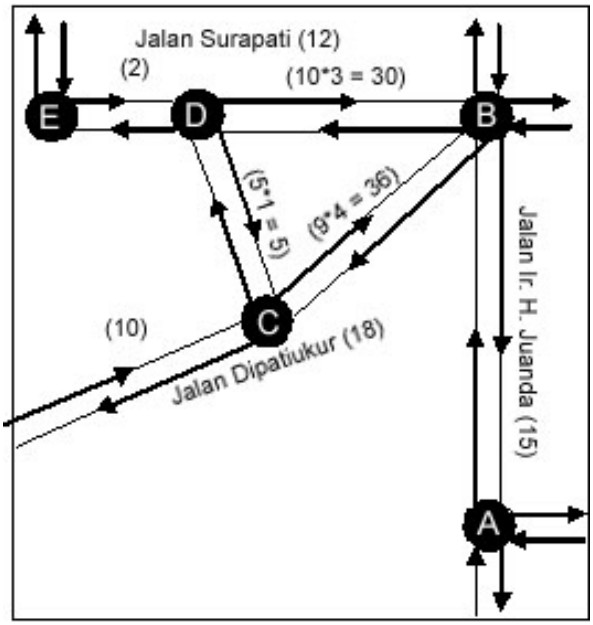
Dengan jarak pada gambar sudah dikali skala. Dengan adanya data-data pada peta, tentu kita sudah akan tahu mana jalan yang akan ditempuh, jika ingin menempuh jarak terpendek. Menurut algoritma A*, jika seseorang ingin berjalan dari A ke C dengan jarak terpendek, pasti orang itu akan berjalan dari A ke B, lalu dari B ke C, bukannya dari A ke B, lalu ke D dan baru ke C.

**4.2 Pengaruh Kepadatan Lalu Lintas dalam Graf**

Ternyata, dalam lalu lintas jalan yang sebenarnya, tidak semudah itu, apalagi berkenaan dengan angkutan umum. Terkadang, dalam jalan yang pendek pun dapat terdapat kemacetan yang sangat luar biasa sehingga tidak dapat cepat mencapai tujuan.

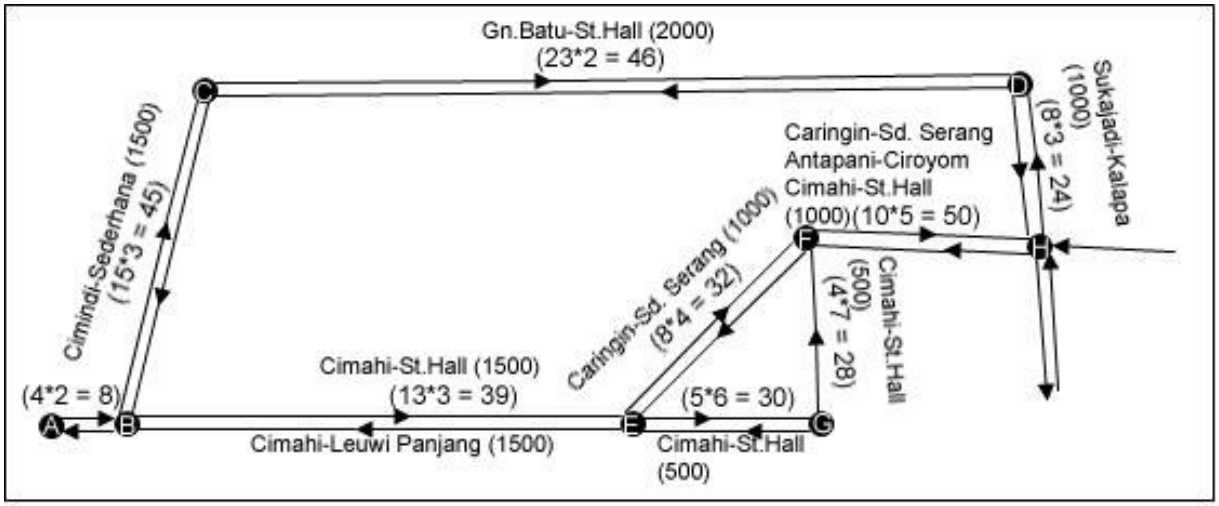
Dalam pemodelan graf, faktor-faktor tersebut dimasukkan dalam pengali yang akan berpengaruh pada jarak tempuh dalam bobot simpul. Dari sana, diketahui bahwa bobot simpul bukanlah jarak seperti sedia kala, namun merupakan "waktu tempuh".

Bobot kepadatan lalu lintas didefinisikan antara 1-10. Skala 1 adalah keadaan jalan yang sangat lancar, sedangkan skala 10 adalah kemacetan lalu lintas. Skala ini nantinya akan dikalikan dengan bobot jarak pada setiap sisi pada graf yang sudah dibuat. Contoh graf yang sudah mempunyai bobot "waktu tempuh":




Gambar 4: Graf jalan dengan bobot jarak dan skala kemacetan

Jika seseorang ingin berjalan dari B ke C, dengan algoritma A*, pasti lebih memilih jalan dari B-D-C daripada jalan B-C langsung, karena B-D-C memiliki




Gambar 5: Graf yang sudah dilengkapi tarif angkutan kota

bobot 35, sedangkan B-C memiliki bobot 36.

### 4.3 Pengaruh Tarif Angkot dalam Graf

Mungkin saja nanti dalam aplikasinya, orang mencari jalan yang hanya mengeluarkan uang paling sedikit, walaupun jauh. Pada graf, bobot pada ruas jalan diganti dengan bobot tarif angkot, dengan catatan setiap kita berganti angkot, harus mengeluarkan biaya tambahan.

Adapun untuk mengetahui yang mana yang optimal, semua aspek (jarak, kemacetan, dan ongkos) harus dikalikan agar terlihat mana jalan yang paling optimal, baik dari segi ongkos maupun dari segi waktu tempuh.

## 5. PENERAPAN *PATHFINDING* DALAM CONTOH KASUS

Seorang turis yang menginap di daerah Cimahi, ingin menuju ke Istana Plaza dengan menggunakan angkutan kota. Didapat data angkutan kota yang mengarah ke sana dapat dilihat pada gambar 5.

Titik awal (Penginapan) adalah A, sedangkan titik akhir (Istana Plaza) adalah H. Jalan baru bercabang di B. Sekarang ditinjau hanya dari jaraknya, diperoleh data sebagai berikut:

Jalur A-B-C-D-H:
Jarak tempuh = 4 + 15 + 23 + 8 = 50

Jalur A-B-E-F-H:
Jarak tempuh = 4 + 13 + 8 + 10 = 35

Jalur A-B-E-G-F-H:
Jarak tempuh = 4 + 13 + 5 + 4 + 10 = 36

Jika sang turis ingin berjalan kaki dari A menuju H, maka jarak terdekat adalah 35. Namun, jika dilihat jarak aktual dari A ke H yang mencapai 6 km, pasti sang turis tidak mau berjalan kaki ke sana.

Sang turis pun memutuskan untuk menaiki angkot dengan angaran ongkos yang tak terbatas. Ia hanya ingin mencapai tempat tujuannya dengan cepat, tidak mempedulikan ongkos yang harus ia keluarkan. Maka hitungannya menjadi jarak dikali kemacetan.

Jalur A-B-C-D-H:
Waktu tempuh = 8 + 45 + 46 + 24 = 123

Jalur A-B-E-F-H:
Waktu tempuh = 8 + 39 + 32 + 50 = 129

Jalur A-B-E-F-G-H:
Waktu tempuh = 8 + 39 + 30 + 28 + 50 = 155

Jika sang turis menempuh jalan menaiki angkot tanpa memperhatikan ongkos yang ia punya, maka lebih baik ia memilih jalan A-B-C-D-H.

Jika seorang turis lainnya ingin menuju ke tempat yang sama dari titik B, tetapi mempunyai ongkos yang pas-pasan, dan ingin mengambil rute yang paling murah. Maka hitungannya adalah dengan tarif angkot.

Jalur B-C-D-H:
Ongkos angkot: (1500 + 500) + (2000 + 500) + (1000 + 500) = 6000

Jalur B-E-F-H:
Ongkos angkot:
Tanpa berganti angkot di F = (1500 + 500) + (1000 + 500) + 1000 = 4500
Dengan berganti angkot di F = (1500 + 500) + (1000 + 500) + (1000 + 500) = 5000

Jalur B-E-F-G-H:
Ongkos angkot:
Tanpa berganti angkot di E dan F = (1500 + 500) + 500 + 500 + 1000 = 4000
Berganti angkot di E saja = (1500 + 500) + (500 + 500) + 500 + 1000 = 4500
Berganti angkot di F saja = (1500 + 500) + 500 + 500 + (1000 + 500) = 4500
Berganti angkot di E dan F = (1500 + 500) + (500 + 500) + 500 + (1000 + 500) = 5000

Dari data di atas, didapat ongkos yang paling murah adalah dengan jalur B-E-F-G-H tanpa mengganti angkot, artinya hanya menaiki angkot Cimahi-St.Hall dari B sampai H.

Ternyata sang turis ingin mendapatkan hasil yang optimal. Tidak terlalu mahal, juga tidak terlalu lama. Dengan keinginannya yang seperti itu, dihitunglah hasil perkalian dari semuanya (setelah diambil yang terbaik dari masing-masing alternatif).

Jalur A-B-C-D-H:
Akumulasi = 123 * 6000 = 738000

Jalur A-B-E-F-H:
Akumulasi = 129 * 4500 = 580500

Jalur B-E-F-H:
Akumulasi = 155 * 4000 = 620000

Dari sana, sang turis mendapatkan jalan terbaik yaitu dengan jalur A-B-E-F-H. Dan dari pengalaman penulis, memang jalan yang paling nyaman untuk dilewati adalah jalur A-B-E-F-H.

## 6. KESIMPULAN

Dalam mencapai suatu tujuan dari tujuan yang lain, dapat dengan melewat berbagai jalan. Namun, pasti ada alternatif terbaik dari jalan-jalan tersebut. Dari sana, dapat digunakan metode *pathfinding* untuk mencari jalan terbaik.

## DAFTAR REFERENSI

[1] Munir, Rinaldi. Buku Teks Ilmu Komputer Matematika Diskrit. Informatika Bandung, 2005.
[2] Russell, S. J.; Norvig, P. (2003). Artificial Intelligence: A Modern Approach
[3]http://theory.stanford.edu/~amitp/GameProgramming/ImplementationNotes.html. Waktu akses: 6 Januari 2009, 00.32
[4] http://en.wikipedia.org/wiki/A*_search_algorithm. Waktu akses: 6 Januari 2009, 01.25