**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**INSTITUT TEKNLOGI SEPULUH NOPEMBER**

**USULAN TUGAS AKHIR**

# IDENTITAS PENGUSUL

**NAMA : Olyn Riptianingdyah**

**NRP : 5110100117**

**DOSEN WALI : Bilqis Amaliah, S.Kom, M.Kom**

**DOSEN PEMBIMBING : 1. Bilqis Amaliah, S.Kom, M.Kom**

**2. Dr. Chastine Fatichah, S.Kom, M.Kom**

# JUDUL TUGAS AKHIR

“Implementasi metode *Node Combination* untuk menemukan rute terpendek untuk Studi Kasus Antar kota di pulau Jawa”

# LATAR BELAKANG

Jalan darat memiliki peranan penting untuk mewujudkan perkembangan kehidupan bangsa. Oleh karena itu jalan darat sangat dibutuhkan bagi masyarakat di dalam melakukan aktifitasnya sehari-hari. Perlu adanya jaringan transportasi untuk menjangkau daerah potensial dan daerah terpencil agar dapat membantu mengembangkan daerahnya. Maka dari itu para ahli tiada henti-hentinya mengembangkan beberapa metode yang lebih baik untuk membantu menemukan rute terpendek agar pengendara mencapai tempat tujuan dengan waktu yang lebih cepat dan efisien.

Seiring dengan waktu yang berjalan dan juga perkembangan ilmu pengetahuan dan teknologi, saat ini banyak sekali algoritma-algoritma yang digunakan untuk memecahkan permasalahan lintasan terpendek. Salah satu dari algoritma yang sangat terkenal dan sangat sering digunakan dalam menemukan rute terpendek adalah algoritma *Dijkstra*. Algoritma *Dijkstra* [1] merupakan salah satu bentuk algoritma *greedy* yang dapat dipahami sebagai berikut:

Untuk setiap *node* sumber(*source*) dalam graf, algortima ini akan mencari jalur dengan *cost minimum*, antara *node* tersebut dengan *node* lainnya. Algoritma ini juga dapat digunakan untuk mencari total biaya(*cost*) dari lintasan terpendek yang dibentuk dari sebuah *node* ke sebuah *node* tujuan. Penulis mengusulkan metode alternatif lain yang lebih sederhana dan mudah untuk dipahami yaitu *Node Combination* untuk mengimplementasikan algoritma *Dijkstra*.

Metode *Node Combination* menemukan rute terpendek dengan tiga langkah yang sederhana yaitu: menemukan tetangga terdekat pada *node* awal, menggabungkan *node* itu dengan *node* awal, dan memodifikasi bobot pada tepi yang terhubung dengan tetangga (*adjacent*)terdekat [2]. Metode *Node Combination* lebih mudah dipahami dan lebih sesuai untuk pemrograman karena tidak perlu menetapkan set dengan jarak *node*-nya. Percobaan evaluasi di berbagai jaringan mengatakan bahwa metode *Node Combination* sama-sama efisien dengan algoritma *Dijkstra*. Seluruh pemrosesan metode *Node Combination* dapat diimplementasikan dengan vektor maka penulis juga menunjukkan bagaimana menemukan rute terpendek pada matriks berbobot.

Hasil yang diharapkan pada pengerjaan tugas akhir ini adalah pembuktian metode *Node Combination* yang lebih sederhana dan mudah dipahami untuk menyelesaikan permasalahan menemukan rute atau lintasan terpendek antar kota di pulau Jawa.

# RUMUSAN MASALAH

Rumusan masalah yang diangkat dalam menyelesaikan tugas akhir ini adalah:

* 1. Bagaimana menemukan rute terpendek dari sebuah *node* ke *node* yang lain dengan metode *Node Combination*?
  2. Bagaimana menemukan *node* mana saja yang dilalui untuk jarak terpendek?
  3. Bagaimana menemukan jalur terpendek antar kota di pulau Jawa dengan menggunakan metode *Node Combination*?

# BATASAN MASALAH

Asumsi dan ruang lingkup permasalahan yang dikerjakan dalam tugas akhir ini adalah:

* 1. Algoritma yang dibandingkan dengan metode *Node Combination* adalah algoritma *Dijkstra*.
  2. Jumlah *node* maksimal adalah dua ratus.

# TUJUAN PEMBUATAN TUGAS AKHIR

Tujuan pembuatan tugas akhir ini adalah mengimplementasikan metode *Node Combination* pada permasalahan menemukan rute terpendek dan menemukan *node* mana saja yang dilalui untuk jarak terpendek.

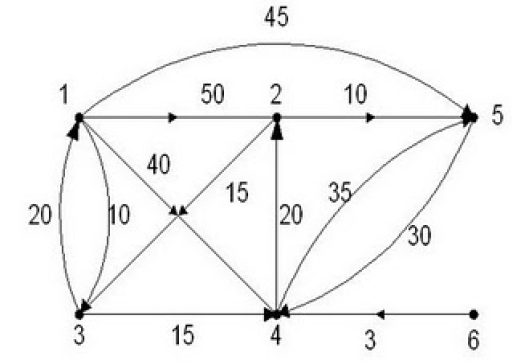
# MANFAAT TUGAS AKHIR

Tugas Akhir ini dikerjakan dengan harapan dapat memberikan manfaat dalam bidang Informatika dan juga memberikan hasil dari pembuktian metode *Node Combination* agar terus dapat dipakai pada penyelesaian untuk menemukan rute terpendek.

# TINJAUAN PUSTAKA

* 1. **Permasalahan menemukan lintasan atau rute terpendek (*Shortest Path Problem*)**

Persoalan menemukan rute terpendek merupakan pencarian rute atau lintasan dengan biaya jalan yang minimum dari satu atau lebih sumber ke satu atau lebih tujuan melewati jaringan yang terhubung. Ini merupakan isu penting karna banyaknya berbagai macam aplikasi dalam transportasi. Dalam beberapa aplikasi, ini juga sangat berguna untuk mengetahui kedua atau ketiga rute terpendek antara dua *node*. Misalnya, untuk meningkatkan ketentuan keefektifan suatu informasi perjalanan, perlu adanya beberapa jalur alternatif rasional untuk pemakai jalanan yang berkendara di jaringan jalan nyata [3]. Persoalan rute terpendek biasanya direpresentasikan dalam bentuk graf yang berbobot (*weighted graph*) seperti pada Gambar 1. Bobot pada sisi graf dapat menyatakan jarak antar kota, waktu pengiriman pesan, ongkos pembangunan, dan sebagainya [4].



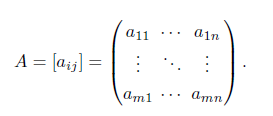
Gambar 1. Graf berarah berbobot [5]

* 1. **Algoritma Dijkstra**

Sampai saat ini, sudah banyak algoritma mencari lintasan terpendek yang pernah ditulis orang. Algoritma lintasan terpendek yang paling terkenal adalah algoritma *Dijkstra*(sesuai dengan nama penemunya, Edsger Wybe *Dijkstra*). Dari naskah aslitnya, algoritma *Dijkstra* diterapkan untuk mencari lintasan terpednek pada graf berarah. Namun, algoritma ini tetap benar untuk graf tak-berarah. Edsger Wybe Dijkstra adalah seorang ilmuwan komputer dan ahli matematika Jerman yang telah memberikan banyak kontribusi pada ilmu komputer. Di antara banyak algoritma yang telah ia temukan, algoritma lintasan terpendek yang dijelaskan di sini merupakan yang terbaik yang pernah diketahui. [4]

* 1. **Matriks**

Matriks merupakan kumpulan elemen yang disusun menurut baris dan kolom sehingga berbentuk empat persegi panjang, yang panjang dan lebarnya ditunjukkan oleh banyaknya kolom dan baris, matriks dapat dikelompokkan sebagai suatu himpunan berdasarkan suatu karakter tertentu dan jenis entri yang tertentu pula. Secara umum, notasi menggambarkan matriks berukuran yang entrinya berada di dalam lapangan(*field*) seperti pada Gambar 2. merupakan ruang vektor berdimensi dengan operasi biner jumlahan dan perkalian dengan scalar, sehingga lebih lanjut dapat diselidiki sifat apa saja yang berlaku pada himpunan matriks tersebut berdasarkan karakter, lapangan yang entri matriksnya berasal serta operasi biner yang berlaku pada matriks tersebut [6].



**Gambar 2. Himpunan matriks berukuran m × n [6]**

* 1. **Metode Node Combination**

Metode *Node Combination* merupakan salah satu metode untuk menemukan rute atau lintasan terpendek dengan cara menggabungkan *node-node* mendjadi *node* awal baru dan memperbaharui bobot *edge* yang terhubung dengan tetangga dekatnya. Jika dibandingkan dengan algoritma Dijkstra metode ini mengkombinasikan *node-node* daripada menetapkan pelabelan pada set-nya. *Node-node* yang tergabung menyerupai sebuah set *node* yang sudah diselesaikan dimana jaraknya sudah ditentukan pada penggunaan algoritma Dijkstra. Tetangga yang berdekatan dengan *node* yang sudah dikombinasikan menyerupai sebuah set *node* potensial dari daerah yang dipilih paling dekat. Sementara itu, bobot *edge* dapat diperbaharui untuk menyimpan label jarak dari *node* awal, daripada menetapkan vektor jarak yang membuat prosedur ini lebih dapat dipahami [2].

# RINGKASAN ISI TUGAS AKHIR

Topik Tugas Akhir yang akan dikerjakan berkaitan dengan implentasi *Node Combination* untuk menemukan rute atau lintasan terpendek.

Kebenaran dari metode ini dapat dibuktikan dengan beberapa teorema sebagai berikut:

***Teorema 1.*** Metode *Node Combination* menyelesaikan permasalahan untuk pencarian rute terpendek dari *node* tertentu ke semua node yang lain (*Single-Source Shortest Path*) dengan urutan dari yang terkecil ke terbesar untuk )

***Definisi:*** dan menjadi bobot dari edge sebelum dan sesudah penggabungan dari node karena dan disini hanya perlu membuktikkan untuk setelah penggabungan *node*. Untuk mendapatkan min untuk

***Teorema 2.*** Diberikan graf G=(V,E,W) dengan nilai bobot pada *edge* yang bukan negatif dan *node* awal metode *Node Combination* menghitung untuk setiap .

***Definisi:*** Perlu diketahui bahwa bobot pada *edge* tidak meningkat selama memproses metode. S merupakan sebuah set node yang sudah diselesaikan dan menjadi bobot terakhir pada edge apabila .

(1). Apabila |S| = 0 atau |S| = 1, sangat jelas bahwa dan adalah rute terpendek dari terhadap tetangga terdekatnya . Ini memiliki arti bahwa .

(2). Misalnya merupakan node pertama yang gagal ditemukan Node Combination untuk mencari rute terpendek, sebagai contoh . (Asumsi |S| = K jika menggabungkan )

(a) Diketahui bahwa rute terpendek bukanlah karena , aturan yang benar dari metode *Node Combination*.

(b) merupakan node terakhir dari di dalam rute terpendek dari s ke f. Maka didapat untuk |S| = k. Untuk |S| apabila mengkombinasikan , min = d(, yang bertentangan dengan hipotesis ini.

Maka bisa diasumsikan rute terpendeknya merupakan , dimana adalah *node* pertama yang tidak digabung. Apabila digabung, maka Kemudian dipilih untuk digabungkan, maka , karena bobot *edge*-nya tidak meningkat, maka . Persamaan kontradiktif dengan hipotesa sehingga setiap node perlu digabungkan dan menghasilkan persamaan: [2].

# METODOLOGI

Metodologi yang digunakan penulis dalam pembuatan serta penyusunan tugas akhir ini adalah sebagai berikut:

## Penyusunan proposal tugas akhir

Penulisan proposal ini merupakan tahap awal dalam pengerjaan Tugas Akhir Pada proposal ini penulis mengajukan gagasan mengenai penerapan metode *Node Combination* pada permasalahan untuk menemukan rute terpendek.

## Studi literatur

Pada tahap ini dilakukan pencarian informasi mengenai topik yang diangkat, studi literatur untuk mengumpulan data dan desain aplikasi yang akan dibuat. Informasi yang didapat yaitu meliputi paper referensi, buku bahasa pemrograman *C*, dan dokumentasi internet.

## Implementasi

Pada tahap ini dilakukan pengkajian lebih lanjut terhadap literatur agar dapat memahami konsep pemahaman pengetahuan lebih detail mengenai metode *Node Combination* serta menemukan solusi yang tepat dengan membandingkan langkah-langkah dan kelebihan metode ini dengan algoritma *Dijkstra*.

## Implementasi perangkat lunak

## Implementasi merupakan tahap untuk membangun metode tersebut. Untuk membangun metode yang telah dirancang sebelumnya, diimplementasikan dengan menggunakan bahasa C.

## Pengujian dan evaluasi

Pada tahap ini dilakukan pengkajian dan analisa keluaran yang berasal dari hasil implementasi yang telah dibuat.

## Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat. Sistematika penulisan buku tugas akhir secara garis besar antara lain:

1. Pendahuluan
   1. Latar Belakang
   2. Rumusan Masalah
   3. Batasan Tugas Akhir
   4. Tujuan
   5. Metodologi
   6. Sistematika Penulisan
2. Tinjauan Pustaka
3. Desain dan Implementasi
4. Pengujian dan Evaluasi
5. Kesimpulan dan Saran
6. Daftar Pustaka
7. **JADWAL KEGIATAN**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tahapan | 2013-2014 | | | | | | | | | | | | | | | | | | | | |
| September | | | | Oktober | | | | November | | | | Desember | | | | | Januari | | | |
| Penyusunan Proposal |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Studi Literatur |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Implementasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Pengujian dan evaluasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Penyusunan buku |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |
| --- |
|  |

# Daftar Pustaka

|  |  |  |
| --- | --- | --- |
| [1] | I. G. P. Wahyu, "ALGORITMA DIJKSTRA," 2013. [Online]. Available: http://share.pdfonline.com/bd4c1da8a3b74704917878a211e9c499/ALGORITMA%20DIJKSTRA.htm. [Accessed Thrusday October 2013]. | |
| [2] | X. Lu and M. Camitz, "Finding the shortest path by node combination," *Applied Mathematics and Computation,* 2011. | |
| [3] | Y. Lim and H. Kim, "A Shortest Path Algorithm For Real Road Network," *Journal of the Eastern Asia Society for Transportation Studies,* vol. VI, 2005. | |
| [4] | A. Ekaputra, "Aplikasi Graf pada Persoalan Lintasan Terpendek dengan Algoritma Dijkstra". | |
| [5] | B. H. Suryanaga, "Beberapa Aplikasi Graf," *Jurnal Ilmu Komputer dan Teknologi Informasi,* vol. III, 2003. | |
| [6] | N. R. Dewi, N. Eliyati and O. H. Marbun, "Kajian Struktur Aljabar Grup Pada Himpunan Matriks yang Invertibel," *Jurnal Penelitian Sains,* vol. XIV. | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
|  |  |
|  |  | |