SKRIPSI

PENCARIAN SEMUA PASANGAN JALUR TERPENDEK DENGAN ALGORITMA DIJKSTRA YANG DIMODIFIKASI

Stanley Hanes

NPM: 2014730036

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2018

UNDERGRADUATE THESIS

FINDING ALL PAIRS SHORTEST PATH USING MODIFIED DIJKSTRA ALGORITHM

Stanley Hanes

NPM: 2014730036

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2018

LEMBAR PENGESAHAN

PENCARIAN SEMUA PASANGAN JALUR TERPENDEK DENGAN ALGORITMA DIJKSTRA YANG DIMODIFIKASI

Stanley Hanes

NPM: 2014730036

Bandung, 18 Mei 2018

Menyetujui,

Pembimbing

Luciana Abednego, M.T.

Ketua Tim Penguji	Anggota Tim Penguji
Natalia, M.Si.	**Mariskha Tri Adithia, P.D.Eng**

Mengetahui,

Ketua Program Studi

Mariskha Tri Adithia, P.D.Eng

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

PENCARIAN SEMUA PASANGAN JALUR TERPENDEK DENGAN ALGORITMA DIJKSTRA YANG DIMODIFIKASI

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal 18 Mei 2018

Stanley Hanes
NPM: 2014730036

ABSTRAK

Menemukan jalur terpendek merupakan masalah mendasar dalam teori graf. Pencarian jalur terpendek dapat diterapkan dalam banyak bidang seperti ilmu komputer, teknik transportasi, maupun analisis jaringan. Salah satu contoh aplikasi dalam bidang transportasi adalah pencarian jalur terpendek dari sebuah kota ke kota lainnya. Walaupun digunakan dalam banyak bidang, tetapi pencarian jalur terpendek pada jaringan kompleks berskala besar masih membutuhkan waktu pengerjaan yang lama. Salah satu algoritma yang dapat digunakan untuk menyelesaikan masalah pencarian jalur terpendek adalah Algoritma Dijkstra.

Akan tetapi, Algoritma Dijkstra kurang cocok jika diterapkan dalam jaringan kompleks berskala besar karena jaringan kompleks berskala besar memiliki sisi yang banyak. Jika menggunakan Algoritma Dijkstra biasa, maka semua sisi tersebut akan dihitung berulang-ulang. Algoritma Dijkstra dapat dikembangkan dengan sedikit modifikasi agar sesuai dengan karakteristik jaringan kompleks berskala besar, sehingga setiap sisi pada jaringan kompleks berskala besar tidak dihitung berulang-ulang. Modifikasi yang dilakukan adalah memanfaatkan informasi yang sudah didapatkan pada langkah sebelumnya, sehingga dapat mempercepat langkah akhir perhitungan. Selain itu, Algoritma Dijkstra modifikasi tersebut digunakan pada tiga algoritma lainnya yang berfungsi untuk memanggil Algoritma Dijkstra modifikasi yaitu *Basic Algorithm*, *Optimized Algorithm*, dan *Adaptive Algorithm*. Pada ketiga algoritma tersebut, memiliki cara yang berbeda dalam pengurutan simpul yang akan dihitung.

Perangkat lunak yang dibangun dapat melakukan pencarian semua pasangan jalur terpendek dengan menggunakan Algoritma Dijkstra biasa maupun Algoritma Dijkstra modifikasi. Pengguna hanya perlu memasukkan graf yang akan digunakan dan memilih pilihan algoritma yang tersedia. Setelah itu, pengguna akan mendapatkan jalur terpendek dari semua pasangan simpul dan waktu pengerjaan dari algoritma tersebut.

Berdasarkan pengujian yang telah dilakukan, waktu pengerjaan dari Algoritma Dijkstra modifikasi pada *Basic Algorithm* lebih baik dibandingkan Algoritma Dijkstra biasa. Waktu pengerjaan *Adaptive Algorithm* pada graf yang memiliki simpul yang sedikit lebih baik dibandingkan Algoritma Dijkstra biasa, tetapi pada graf dengan simpul yang banyak *Adaptive Algorithm* lebih buruk. Hasil pengujian pada *Optimized Algorithm* menunjukkan waktu pengerjaan yang lebih lama dibandingkan Algoritma Dijkstra biasa. Pada *Basic Algorithm*, *Optimized Algorithm*, dan *Adaptive Algorithm*, terdapat parameter yang dapat mempengaruhi performa dari masing-masing algoritma.

Kata-kata kunci: Semua Pasangan Jalur Terpendek, Dijkstra, Jaringan Kompleks Berskala Besar, Kompleksitas Waktu

ABSTRACT

Finding shortest path is a fundamental problem in graph theory. Finding shortest path can be applied in many areas like computer science, transportation, and network analysis. Example of application in transportation is finding shortest path from one city to another city. Although applied in many areas, but finding shortest path still time-consuming for large-scale complex network. One of the algorithm that can be used to solve shortest path problem is Dijkstra Algorithm.

However, Dijkstra Algorithm is still time-consuming when applied on large-scale complex network with tremendous volume of data. If using Dijkstra Algorithm, all edges will be calculated repeatedly. Dijkstra Algorithm can be improved by simple modification with consideration of characteristics of large-scale complex network, so each edge will not be calculate repeatedly. Modifications is to utilize the previously calculated results to accelerate the latter calculation. In addition, modified Dijkstra Algorithm is used on three other algorithms that serve to call modified Dijkstra Algorithm which is Basic Algorithm, Optimized Algorithm, and Adaptive Algorithm. In the three algorithm, it has a different way of sorting the node to be called with modified Dijkstra Algorithm.

The software that has been built can solve all pairs shortest path problem with Dijkstra Algorithm as well as modified Dijkstra Algorithm. Users need to input graph and select algorithm. After that, users will get the all pairs shortest path and running time of the selected algorithm.

Based on tests that have been done, running time of the Basic Algorithm is better than Dijkstra Algorithm. Running time of the Adaptive Algorithm with few vertex is better than Dijkstra Algorithm, but with many vertex is worse. Test results on Optimized Algorithm show that running time of the Optimized Algorithm is longer than Dijkstra Algorithm. In the Basic Algorithm, Optimized Algorithm, and Adaptive Algorithm, there are parameters that can affect the performance of each algorithm.

Keywords: All Pairs Shortest Path Problem, Dijkstra, Large Scale Complex Network, Time Complexity

Dipersembahkan untuk diri sendiri dan orang tua

KATA PENGANTAR

Puji syukur penulis ucapkan kepada Tuhan Yang Maha Esa atas berkat dan karunia-Nya, sehingga penulisan skripsi yang berjudul "Pencarian Semua Pasangan Jalur Terpendek dengan Algoritma Dijkstra yang Dimodifikasi" dapat diselesaikan. Penulisan skripsi ini dibuat untuk memenuhi salah satu syarat dalam memperoleh gelar sarjana pada Program Studi Teknik Informatika Fakultas Teknologi Informasi dan Sains Universitas Katolik Parahyangan.

Penulis menyadari keberhasilan proses perkuliahan dan penulisan skripsi ini tidak lepas dari berbagai bantuan dan dorongan dari berbagai pihak. Atas semua bantuan yang sudah diberikan, penulis ingin mengucapkan terima kasih dan memberikan rasa hormat kepada:

- Orang tua dan keluarga penulis yang selalu mendoakan dan menyemangati penulis dari awal perkuliahan sampai penulisan skripsi ini dapat diselesaikan.
- Bapak Husnul Hakim selaku dosen pembimbing yang telah memberikan banyak masukan dan referensi yang mendukung penulisan skripsi ini.
- Segenap dosen Jurusan Teknik Informatika Universitas Katolik Parahyangan yang telah memberikan ilmu kepada penulis.
- Segenap karyawan Universitas Katolik Parahyangan.
- Sahabat dekat yang selalu mengisi hari-hari penulis Davidson Hans Hanly, Dedy Gunawan, Dessy Handayani, Djong, Freddy Liandy, Ines Nuary, Lucky Kumala, Kelsen, Kenny, Momok, Sthephanie Merin, Valencia Suryaatmaja, Veronica Anjelia, Wiwi.
- Teman main penulis Agus, Andy, Daniel, Fedrian, Ivan, Kelvin, Kevin, Kresna, Sapta, Stillmen, Andre, Elia, Ferdi, Hendri, Agina, Carissa, Clara, Devi, Jovanka, Marchella, Nancy.
- Semua teman sejurusan yang tidak dapat disebutkan satu per satu atas semua dukungan kepada penulis.
- Albert yang telah menyediakan tempat tinggal selama revisi di Bandung.

Penulis ingin memohon maaf atas kekurangan dan kesalahan yang terdapat pada penulisan skripsi ini. Penulis menyadari bahwa skripsi ini masih memiliki banyak kekurangan dan jauh dari sempurna. Akhir kata, semoga skripsi ini dapat bermanfaat dan menambah wawasan bagi pembaca.

Bandung, Mei 2018

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1

PENDAHULUAN

Bab ini berisi tentang latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi penelitian, dan sistematika pembahasan dari penelitian ini.

1.1 Latar Belakang

Menemukan jalur terpendek merupakan masalah mendasar dalam teori graf. Permasalahan yang akan diselesaikan pada pencarian rute terpendek adalah untuk menemukan jalur antara dua simpul dalam sebuah graf, sehingga bobot sisi yang dilewati dapat diminimalkan. Salah satu contoh permasalahan pencarian rute terpendek di dunia nyata adalah pencarian rute terpendek dari suatu kota ke kota lainnya. Ketika ingin berkunjung dari Kota A ke Kota B, terdapat beberapa jalur alternatif. Dengan menyelesaikan masalah tersebut, akan didapatkan jarak terpendek dari semua jalur yang dapat dilalui dari Kota A ke Kota B.

Masalah pencarian jalur terpendek disebut juga sebagai *Single Source Shortest Path (SSSP) problem*. SSSP merupakan masalah untuk menemukan jalur terpendek dari satu simpul ke semua simpul lainnya. Permasalahan pencarian rute terpendek dari suatu kota ke kota lainnya termasuk dalam masalah SSSP. Kota awal merupakan simpul sumber, sedangkan kota tujuan merupakan simpul akhir yang dikunjungi [1].

All Pairs Shortest Path (APSP) problem merupakan masalah untuk menemukan jalur terpendek dari semua pasangan simpul dalam sebuah graf. APSP biasanya diterapkan dalam jaringan kompleks dalam skala yang besar. Pencarian jalur terpendek ini juga dapat diaplikasikan dalam banyak bidang, termasuk ilmu komputer, riset operasi, teknik transportasi, *network routing*, dan analisis jaringan. Karena digunakan dalam banyak bidang, pencarian jalur terpendek tersebut dipelajari secara ekstensif.

Untuk menyelesaikan masalah APSP tersebut, dapat digunakan Algoritma Dijkstra. Pada Algoritma Dijkstra, dibutuhkan sebuah matriks bobot yang merupakan bobot pada setiap sisi pada graf. Jika Algoritma Dijkstra diterapkan dalam jaringan kompleks dalam skala besar, algoritma tersebut masih membutuhkan waktu yang cukup lama. Sehingga, Algoritma Dijkstra tersebut dimodifikasi agar dapat disesuaikan dengan karakteristik dari jaringan kompleks dalam skala besar. Salah satu contoh Algoritma Dijkstra yang dimodifikasi oleh Wei Peng yaitu dengan menggunakan informasi yang sudah didapat dari langkah sebelumnya untuk mempercepat langkah akhirnya. Dengan menggunakan algoritma ini, kompleksitas waktu dari algoritma menjadi $O(n^{2.4})$ [2].

Algoritma Dijkstra yang dimodifikasi tersebut kemudian digunakan pada ketiga algoritma lainnya, yaitu *Basic Algorithm*, *Optimized Algorithm*, dan *Adaptive Algorithm*. Pada *Basic Algorithm*, digunakan Algoritma Dijkstra yang sudah dimodifikasi pada semua simpul yang ada pada graf. Pada *Optimized Algorithm*, urutan perhitungan simpul diurutkan berdasarkan banyaknya derajat pada simpul tersebut. Banyaknya simpul yang diurutkan sesuai dengan rasio yang ditentukan oleh pengguna (0-1) dikalikan dengan banyaknya simpul yang ada pada graf. Pada *Adaptive Algorithm*, semua simpul diurutkan berdasarkan banyaknya derajat pada simpul tersebut, kemudian ketika simpul tersebut baru pertama kali dihitung, akan ditambahkan derajat simpul tersebut dalam kondisi tertentu.

Dengan adanya modifikasi pada Algoritma Dijkstra tersebut, diharapkan dapat menyelesaikan beberapa permasalahan pada APSP dalam skala yang besar dengan waktu yang lebih singkat. Sehingga dapat meminimalisir waktu pengerjaan masalah APSP.

Pada skripsi ini, akan dibuat sebuah perangkat lunak yang dapat membandingkan hasil dan waktu dari algortima-algoritma Dijkstra yang sudah dimodifikasi. Dengan menggunakan perangkat lunak tersebut, pengguna juga dapat mencari semua pasangan jalur terpendek dengan memilih algoritma yang akan digunakan dan menampilkan waktu pengerjaan algoritma yang dipilih.

1.2 Rumusan Masalah

Pada skripsi ini, terdapat beberapa masalah yang perlu dirumuskan. Masalah tersebut yaitu:

- Bagaimana menyelesaikan masalah *All Pairs Shortest Path* dengan menggunakan Algoritma Dijkstra?
- Bagaimana menyelesaikan masalah *All Pairs Shortest Path* dengan menggunakan Algoritma Dijkstra yang dimodifikasi?
- Bagaimana membuat perangkat lunak untuk memecahkan masalah *All Pairs Shortest Path* dengan menggunakan algoritma Dijkstra yang sudah dimodifikasi?
- Bagaimana perbandingan waktu pengerjaan masing-masing algoritma yang menggunakan Algoritma Dijkstra modifikasi?

1.3 Tujuan

Berdasarkan dari rumusan masalah tersebut, tujuan dari skripsi ini yaitu:

- Mempelajari cara kerja dari Algoritma Dijkstra yang secara spesifik digunakan untuk menyelesaikan masalah *All Pairs Shortest Path*
- Mempelajari cara kerja dari Algoritma Dijkstra modifikasi yang secara spesifik digunakan untuk menyelesaikan masalah *All Pairs Shortest Path*
- Membuat perangkat lunak yang dapat digunakan untuk menyelesaikan masalah *All Pairs Shortest Path*
- Membandingkan waktu dari Algoritma Dijkstra yang sudah dimodifikasi menggunakan perangkat lunak yang dibangun

1.4 Batasan Masalah

Terdapat beberapa batasan masalah pada perangkat lunak yang dibangun, yaitu:

- Bobot setiap sisi pada matriks bobot tidak boleh bernilai negatif karena dapat menyebabkan hasil perhitungan yang kurang tepat. Misalkan terdapat sebuah simpul A dengan bobot sisi maksimum dan bobot sisi negatif. Simpul A akan menjadi simpul terakhir yang dikunjungi karena harus melalui simpul dengan bobot maksimum, sedangkan terdapat sisi dengan bobot negatif yang dapat mengurangi jarak terpendek. Walaupun simpul B yang bersisian dengan sisi negatif bertetangga dengan simpul lain, simpul B tidak dapat dihitung ulang karena sudah pernah dikunjungi.

1.5 Metodologi

Metodologi penelitian yang digunakan pada skripsi ini terbagi dalam beberapa langkah, yaitu:

- Melakukan studi literatur mengenai karakteristik dan sifat graf
- Melakukan studi literatur mengenai algoritma Dijkstra dan Dijkstra modifikasi
- Melakukan analisis terhadap penyelesaian masalah *All Pairs Shortest Path* dengan menggunakan algoritma Dijkstra
- Melakukan analisis kebutuhan dari perangkat lunak yang akan dibangun
- Melakukan perancangan perangkat lunak
- Mengimplementasi keseluruhan algoritma dan struktur data yang dirancang
- Melakukan pengujian terhadap hasil yang dikeluarkan perangkat lunak
- Menulis dokumen skripsi

1.6 Sistematika Pembahasan

Sistematika pembahasan dibagi dalam enam bab sebagai berikut:

- Bab 1 Pendahuluan
  Bab 1 membahas tentang latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi dan sistematika pembahasan yang digunakan untuk menyusun laporan ini.
- Bab 2 Landasan Teori
  Bab 2 berisi landasan teori yang digunakan dalam pembangunan perangkat lunak yaitu teori graf, *Single Source Shortest Path* dan *All Pairs Shortest Path*, Algoritma Dijkstra dan Algoritma Dijkstra modifikasi.
- Bab 3 Analisis
  Bab 3 berisi tentang analisis kebutuhan perangkat lunak. Analisis dimodelkan dalam bentuk *use case*, diagram kelas, dan diagram aktivitas. Selain itu, dibahas juga mengenai analisis masalah Algoritma Dijkstra dan contoh penyelesaian Algoritma Dijkstra.
- Bab 4 Perancangan
  Bab 4 berisi tentang perancangan perangkat lunak, meliputi masukan dan keluaran perangkat lunak, antarmuka perangkat lunak, rincian kelas, dan diagram sekuens.
- Bab 5 Implementasi dan Pengujian
  Bab 5 berisi tentang implementasi dari rancangan perangkat lunak. Terdapat juga pengujian yang dilakukan beserta hasil pengujiannya.
- Bab 6 Kesimpulan dan Saran
  Bab 6 berisi tentang kesimpulan dan saran dari penelitian ini.