Implementasi Algoritma Dijkstra Untuk Menentukan Jarak Terdekat Pondok Pesantren Berbasis Situs

Yogi Abubakar Ridwan[1], Asri Adi Sunarto[2], Lelah[3]
[1,2,3] Program Studi Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Muhammadiyah Sukabumi
Jl. R. Syamsudin, S.H. No. 50, Cikole, Kec. Cikole, Kota Sukabumi, Jawa Barat 43113
[1] yogiab88@gmail.com

ABSTRAK

Pondok pesantren merupakan lembaga pendidikan Islam tradisional khas Indonesia yang mampu eksis hingga saat ini. Pondok pesantren terbukti merupakan lembaga pendidikan yang unik, mandiri, dan syarat dengan kultur lokal sejak 500 tahun yang lalu. Model pendidikan barat yang diadopsi pemerintah sebagai pendidikan formal di Indonesia telah memberikan tantangan dan warna terhadap perkembangan pondok pesantren. Pesantren pun menyikapi tantangan tersebut dengan cara yang berbeda-beda, di sisi lain pesantren dituntut untuk mampu mempertahankan idealismenya sekaligus memenuhi tuntutan pragmatis dari masyarakat. Tujuan penelitian ini adalah membangun sebuah sistem informasi pemetaan Pondok Pesantren yang tersebar di Kota Maupun Kabupaten Sukabuni dengan menggunakan metode algoritma *Dijktra* dalam menghitung jarak terdekat Pondok Pesantren. Hasil penelitian ini bisa menjadi sebuah media informasi bagi masyarakat tentang Informasi Pondok Pesantren, Lokasi Pondok Pesantren, dan jarak terdekat Pondok Pesantren.

Kata Kunci : Algoritma Dijktra, Pencarian Jarak Terpendek, Pondok Pesantren

1. Pendahuluan

Pondok Pesantren merupakan lembaga pendidikan Islam tradisional khas Indonesia yang mampu eksis hingga saat ini. Pondok pesantren terbukti merupakan lembaga pendidikan yang unik, mandiri, dan syarat dengan kultur lokal sejak 500 tahun yang lalu. Model pendidikan barat yang diadopsi pemerintah sebagai pendidikan formal di Indonesia telah memberikan tantangan dan warna terhadap perkembangan pondok pesantren. Keberadaan pondok pesantren yang tersebar di seluruh wilayah Kota dan Kabupaten Sukabumi yang berjumlah 756 pondok pesantren salafi dan pondok pesatren modern. yang mengharuskan adanya suatu sistem yang memudahkan dalam penyediaan informasi mengenai pondok pesantren, lokasi pondok pesantren, daya tampung santri, fasilitas yang ada dalam Pondok Pesantren, yang berguna bagi seluruh masyarakat dan khususnya warga Kota dan Kabupaten Sukabumi. [1].

Menurut peraturan menteri agama republik indonesia no 18 tahun 2014 tentang satuan pendidikan muadalah pada pondok pesantren pasal 1 yaitu satuan pendidikan muadalah pada pondok pesantren yang selanjutnya disebut satuan pendidikan muadalah adalah satuan pendidikan keagamaan Islam yang diselenggarakan oleh dan berada di lingkungan pesantren dengan mengembangkan kurikulum sesuai kekhasan pesantren dengan basis kitab kuning atau dirasah islamiyah dengan pola pendidikan muallimin secara berjenjang dan terstruktur yang dapat disetarakan dengan jenjang pendidikan dasar dan menengah di lingkungan Kementerian Agama dan menurut pasal 2 yaitu Pendidikan keagamaan Islam adalah pendidikan yang mempersiapkan peserta didik untuk dapat menjalankan peranan yang menuntut penguasaan pengetahuan tentang ajaran agama Islam dan/atau menjadi ahli ilmu agama Islam dan mengamalkan ajaran agama Islam. [5].

Namun dikarenakan keterbatasan informasi, maka akses terhadap pondok pesantren tersebut sulit untuk didapat, dan menghambat dalam pengelolaan informasi pondok pesantren. Oleh karena itu dibutuhkan sebuah sistem yang dapat memudahkan dalam penyediaan informasi dan penentuan rute dan jarak terpendek pondok pesantren yang dapat akses dimana saja, agar dapat membantu dalam mencari informasi yang efektif dan mengoptimakan geografis dalam menentuka jarak terdekat menuju pondok pesantren.

Pencarian rute terpendek termasuk dalam salah satu persoalan dalam teori graf yang berarti meminimalisasi bobot suatu lintasan dalam graf [2]. Algoritma Dijkstra merupakan salah satu bentuk algoritma populer dalam pemecahan persoalan yang terkait dengan masalah optimasi dan penghitungan jarak terdekat [2].

Berdasarkan permasalahan di atas, maka penulis bermaksud melakukan penelitian dengan judul *"Implementasi Algoritma Dijkstra untuk menentukan jarak terpendek pondok pesantren berbasis situs"*. Melalui pembuatan sistem ini sebagai media publikasi untuk mengenalkan data lokasi dan informasi. Sehingga nantinya selain pengguna dapat melihat posisi pondok pesantren yang ada, pengguna juga dapat mengetahui rute dan jarak terdekat mana yang di lewati untuk mencapaipondok pesantren di Kota maupun Kabupaten Sukabumi dengan cara yang lebih mudah dan lebih cepat.

2. Metode Penelitian

Metode penelitian yang dilakukan oleh penulis pada penelitian kali ini adalah dengan menggunakan algoritma Dijkstra sebagai pencarian rute dan jarak terpendek. Pencarian rute dan jarak terpendek merupakan pencarian lintasan minimum yang diperlukan untuk mencapai suatu tempat dari tempat tertentu. Lintasan minimum yang dicari menggunakan *graph*.

Graph merupakan titik di dalam bidang dua dimensi yang dihubungkan dengan sekumpulan garis (*edge*) dan dibentuk dari kumpulan titik yang dihubungkan dengan garis - garis. Algoritma Dijkstra merupakan salah satu Algoritma yang berfokus pada pencarian jarak terpendek [2].

Masukan algoritma dijkstra dalam pemecahan persoalan yang terkait dengan masalah optimasi dan penghitungan jarak terdekat, sesuai dengan arti greedy yang secara harafiah berarti tamak atau rakus, namun tidak dalam konteks negatif , algoritma greedy ini hanya memikirkan solusi terbaik yang akan diambil pada setiap langkah tanpa memikirkan konsekuensi ke depan. Prinsipnya, ambillah apa yang bisa didapatkan saat ini. Akan tetapi bobot dari graf tersebut harus bernilai bilangan positif (bobot >=0). Input algoritma ini adalah sebuah graf berarah yang berbobot (weighted directed graph) G dan sebuah sumber vertices s dalam G dan V adalah himpunan semua vertices dalam graph G [3].

Algoritma Dijkstra untuk mencari rute dan jarak terpendek adalah sebagai berikut:

G = (V, E).

Keteranga sebagai berikut :

G : Graph
V : Vetices (Titik)
E : Edge (Jarak)

Ada empat langkah yang perlu dilakukan Algoritma Dijkstra sebelum melakukan pada pencarian jarak terpendek :

S : menentukan kumpulan vertices pada graph dimana lokasi awal dan lokasi akhir ditentukan.

V-S : adalah kumpulan dari vertices pada graph dimana shortest path dari satu start ke vertices beluin diketahui.
D : array berisi perkiraan jarak terpendek dari start kesetiap vertices.
T : nilai total dari jarak yang ditempuh.

Cara kerja algoritma Dijkstra adalah :

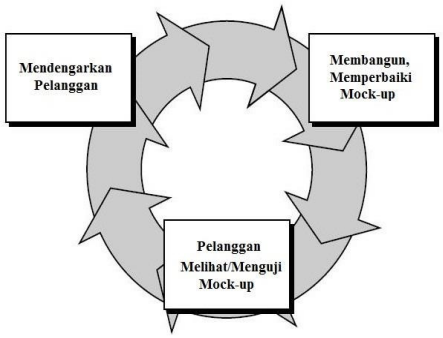
1. Isi S sebagai start (Lokasi awal).
2. Jika lokasi ± lokasi maka isi V-S dengan lokasi yang terhubung dengan lokasi awal.
3. Isi D dengan urutan lokasi yang terhubung dengan lokasi awal yang berdasarkan jarak terpendek dari lokasi awal.
4. Isi T dengan jarak yang ditempuh dan selalu bertambah bila ada jarak yang barn.
5. Apabila lokasi awal = lokasi akhir berari proses pencarian akan selesai, tetapi apabila lokasi awal ^ lokasi akhir lanjutkan langkah ke 2.

```
while($Q->size() > 0) {
list($distance, $u) = $Q->remove();

if (isset($visited[$u])) {
        continue;
$visited[$u] = True;

if (!isset($nodes[$u])) {
//print "WARNING: '$u' is not found in the node list\n";
        }else{

foreach($nodes[$u] as $edge) {

$alt = $dist[$u] + $edge->weight;
$end = $edge->end;
if (!isset($dist[$end]) || $alt < $dist[$end]) {
    $previous[$end] = $u;
    $dist[$end] = $alt;
    $Q->add(array($dist[$end], $end));
                        }
                }
        }
return array($dist, $previous);
}
```

Gambar 1. pseudocode dari algoritma Dijkstra.

Sumber data dan informasi tersebut penulis dapatkan melalui catatan hasil observasi, wawancara, pengamatan secara menyeluruh atau dengan melalui kajian pustaka.

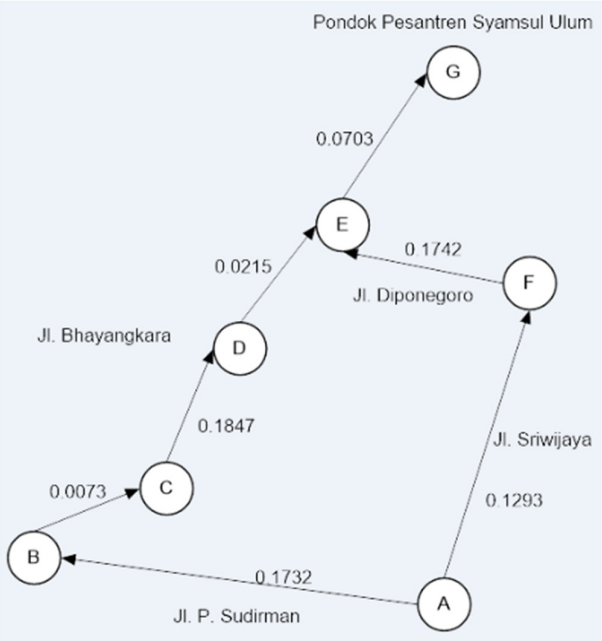
Konsep pengembangan sistem pada penelitian ini menggunakan model prototipe. Model pengembangan ini digunakan karena model ini sangat fleksibel, karena setiap fasenya dapat berulang hingga sesuai dengan kebutuhan [4].



Gambar 2. Tahapan Model Prototipe

3. Hasil dan Pembahasan

Pada bagian ini dibahas studi kasus terhadap implementasi algoritma Dijkstra pada situs *web* yang dibangun. Adapun sebagai contoh diambil graf berbobot yang menghubungan titik A dengan titik G (Pondok Pesantren Syamsul Ulum). Ilustrasi graf dapat dilihat pada gambar 3 di bawah ini :



Gambar 3 Ilustrasi Graf

Langkah-langkah untuk menentukan jarak terpendek dari A, B, C, D, E, F ke pondok pesantren dengan menggunakan algoritma Dijkstra adalah sebagai berikut :

1. Pada awalnya status dari node yang belum terpilih diinisialisasikan dengan ‘0’ dan yang sudah terpilih diinisialisasi dengan ‘1’ dimulai dari node A.
2. Tentukan bobot dari node yang langsung berhubungan dengan node sumber yaitu node A, seperti: dari node A ke node B = 0.1732 M, dan untuk node C, D,E,F diinisialisasi dengan ‘-‘ karena tidak ada lintasan (*arc*) yang menghubungkan secara langsung dengan node A.
3. *Titik* (node sumber) dari node A, B adalah A, karena jarak dihitung dari node A, sehingga node A disebut sebagai *Titik* (node sumber), sedangkan untuk node C, D.,E,F diinisialisasi dengan ‘-‘ dikarenakan tidak ada lintasan (*arc*) yang langsung menghubungkan dari node A, sehingga jaraknya tidak ada.
4. *Titik* (node sumber) dari node A, B adalah A, karena jarak dihitung dari node A, sehingga node A disebut sebagai *Titik* (node sumber), sedangkan untuk node C, D.E,F, diinisialisasi dengan ‘-‘ dikarenakan tidak ada lintasan (*arc*) yang langsung menghubungkan dari node A, sehingga jaraknya tidak ada.

Pengujian Jarak 1

Tabel Hasil Iterasi Ke-1

Node	A	B	C	D	E	F
Status	1	1	0	0	0	0
Bobot	-	1.1732	-	-	-	-
Titik	A	A	-	-	-	-

Dari Tabel pilih node yang memiliki bobot dan status nya masih '0', yaitu node B. Untuk itu status node B menjadi '1' dan *Titik*-nya masih tetap A, dan node yang lain *Titik*-nya masih sama. Jika node B sudah terpilih, maka selanjutnya diperoleh node C dengan bobot 0.007. *Titik* C adalah A.

Tabel Hasil Iterasi Ke-2

Node	A	B	C	D	E	F
Status	1	1	1	0	0	0
Bobot	-	1.1732	0.0073	-	-	-
Titik	A	A	B	-	-	-

Dari Tabel di didapatkan bahwa node C memiliki bobot lebih besar dari A karena hanya C yang berhubungan langsung dengan B, sehingga statusnya akan berubah menjadi '1' dan *Titik*-nya adalah B.

Tabel Hasil Iterasi Ke-3

Node	A	B	C	D	E	F
Status	1	1	1	1	0	0
Bobot	-	1.1732	0.0073	0.1847	-	-
Titik	A	A	B	C	-	-

Dari Tabel di didapatkan bahwa node D memiliki bobot lebih besar 0.1847 dari A karena hanya D yang berhubungan langsung dengan C, B sehingga statusnya akan berubah menjadi '1' dan *Titik*-nya adalah C.

Tabel Hasil Iterasi Ke-4

Node	A	B	C	D	E	F
Status	1	1	1	1	1	0
Bobot	-	1.1732	0.0073	0.1847	0.215	-
Titik	A	A	B	C	D	-

Dari Tabel di didapatkan bahwa node E memiliki bobot lebih besar 0.215 dari A karena hanya E yang berhubungan langsung dengan D, C , B , sehingga statusnya akan berubah menjadi '1' dan *Titik*-nya adalah D.

Tabel Hasil Iterasi Ke-5

Node	A	B	C	D	E	G
Status	1	1	1	1	1	1
Bobot	-	1.1732	0.0073	0.1847	0.215	0.0703
Titik	A	A	B	C	D	E

Dari Tabel di didapatkan bahwa node G memiliki bobot lebih besar 0.0703 dari A karena hanya G yang berhubungan langsung dengan E, D, C , B , sehingga statusnya akan berubah menjadi '1' dan *Titik*-nya adalah E.

Jika node F sudah terhitung semuanya, maka program akan berhenti karena semua node sudah terpilih. Sehingga akan menghasilkan Jarak terpendek dari node A ke setiap node yang ada. Jarak terpendek yang diperoleh adalah A > B>C > D > E > G dengan total bobot 0.4570.

Pengujian Jarak 2

Tabel Hasil Iterasi Ke-1

Node	A	F	E	G		
Status	1	1	-	-		
Bobot	-	1.1293	-	-		
Titik	A	A	-	-		

Dari Tabel di didapatkan bahwa node F memiliki bobot 1.1293 dari A karena hanya E yang berhubungan langsung dengan A , sehingga statusnya akan berubah menjadi '1' dan *Titik*-nya adalah A.

Tabel Hasil Iterasi Ke-2

Node	A	F	E	G		
Status	1	1	1	-		
Bobot	-	1.1293	0.1742	-		
Titik	A	A	F	-		

Dari Tabel di didapatkan bahwa node F memiliki bobot lebih besar 0.1742 dari A karena hanya E yang berhubungan langsung dengan F, sehingga statusnya akan berubah menjadi '1' dan *Titik*-nya adalah F.

Tabel Hasil Iterasi Ke-3

Node	A	F	E	G		
Status	1	1	1	1		
Bobot	-	1.1293	0.1742	0.0703		
Titik	A	A	F	E		

Dari Tabel di didapatkan bahwa node G memiliki bobot lebih besar 0.0703 dari A karena hanya E yang berhubungan langsung dengan E, D, C, B, sehingga statusnya akan berubah menjadi '1' dan *Titik*-nya adalah G.

Jika node sudah terhitung semuanya, maka program akan berhenti karena semua node sudah terpilih. Sehingga akan menghasilkan Jarak terpendek dari node A ke setiap node yang ada. Jarak terpendek yang diperoleh adalah A > F > E > G dengan total bobot 0.3738.

Maka didapatkan jarak terpendek dari hasil penghitungan kedua pengujian tersebut jarak terpendek denga node A > F > E > G dengan total bobot 0.3738 atau 0.37 km.

4. Kesimpulan

Berdasarkan uraian dan pembahasan yang telah dilakukan oleh penulis, maka dapat ditarik kesimpulan sebagai berikut:

1. Algoritma Dijkstra mampu diimplementasikan dalam sistem pencarian rute yang optimal dan dapat memperkirakan jarak yang ditempuh.
2. Implementasi algoritma dijkstra pada situs pondok pesantren ini dapat membantu masyarakat mendapatkan informasi rute terpendek dan jarak yang dapat dilalui

Daftar Pustaka

[1] Kementrian Agama Provinsi Jawa barat. (2016). *Peta Pondok Pesantren jawa barat.* Provinsi Jawa Barat: Kementrian Agama Provinsi Jawa Barat

[2] Siang, J. J. (2009). *Matematika Diskrit Dan Aplikasinya Pada Ilmu Komputer.* Yogyakarta: Andi.
[3] Nawagusti, V. A. (2018). Penerapan Algoritma Floyd Warshall dalam Aplikasi Penentuan Rute Terpendek Mencari Lokasi BTS (Base Tower Station) pada PT.GCI Palembang. *Jurnal Nasional Teknologi dan Sistem Informasi*, 81-89.
[4] Sukamto, R. A., & Shalahuddin, M. (2015). *Rekayasa Perangkat Lunak.* Bandung: Informatika.
[5]PERATURAN MENTERI AGAMA REPUBLIK INDONESIA NOMOR 18 TAHUN 2014 TENTANG SATUAN PENDIDIKAN MUADALAH PADA PONDOK PESANTREN