UJM 4 (2) (2015)
UNNES Journal of Mathematics
http://journal.unnes.ac.id/sju/index.php/ujm

SIMULASI ALGORITMA DIJKSTRA DALAM MENANGANI MASALAH LINTASAN TERPENDEK PADA GRAF MENGGUNAKAN VISUAL BASIC

Hanif Ilmi Mardlootillah✉, Amin Suyitno, Florentina Yuni Arini.

Jurusan Matematika, FMIPA, Universitas Negeri Semarang, Indonesia
Gedung D7 lantai 1 Kampus Sekaran, Gunungpati, Semarang, 50229

Info Artikel

Sejarah Artikel:
Diterima Januari 2014
Disetujui Februari 2014
Dipublikasikan November 2015

Keywords :
Lintasan Terpendek
Algoritma Dijkstra
Visual Basic

Abstrak

Penulisan ini bertujuan untuk memberi gambaran tentang cara membangun simulasi algoritma Dijkstra dalam mencari lintasan terpendek pada suatu graf menggunakan bahasa pemrograman Visual Basic dan membuktikan bahwa penghitungan simulasi algoritma Dijkstra yang dibuat mempunyai hasil solusi yang sama dengan penghitungan manual dalam mencari lintasan terpendek pada graf. Algoritma Dijkstra merupakan algoritma untuk mencari lintasan terpendek yang diterapkan pada graf berarah dan berbobot, yang jarak antar titiknya adalah bobot dari tiap busur pada graf tersebut. Permasalahan yang diangkat adalah cara membangun simulasi algoritma Dijkstra dalam mencari lintasan terpendek pada suatu graf menggunakan bahasa pemrograman Visual Basic dan kecocokan hasil pencarian lintasan terpendek antara penghitungan cara manual dengan menggunakan penghitungan simulasi. Simulasi algoritma Dijkstra dalam menangani masalah lintasan terpendek pada suatu graf dibangun menggunakan bahasa pemrograman Visual Basic. Simulasi yang dibangun kemudian diuji dengan bentuk graf dari hasil representasi. Dari graf yang direpresentasikan, setelah diuji coba menggunakan simulasi ternyata mempunyai solusi hasil lintasan dan jarak yang sama dengan penghitungan manual. Dengan demikian, simulasi algoritma Dijkstra dalam menangani masalah lintasan terpendek pada suatu graf menggunakan Visual Basic selesai direalisasikan dan dapat diimplementasikan pada permasalahan sehari-hari yang dapat direpresentasikan dalam bentuk graf dan dicari lintasan terpendeknya.

Abstract

This paper aims to give information about method a develop simulation algorithm in finding the shortest path Dijkstra on a graph using Visual Basic programming language and prove that Dijkstra's algorithm simulation calculations made have the same results with the solutions manual calculation in finding the shortest path in a graph. Dijkstra's algorithm is an algorithm of finding the shortest path applied to directed, weighted graphs, the distance among nodes is weight of each arc in the graph. Issues raised are how to build the simulation of Dijkstra's algorithm of finding it on a graph using Visual Basic programming language and matches result of the shortest path between manual counting and simulation calculation. Simulation of Dijkstra's algorithm in dealing with it this problems on a graph is built using Visual Basic programming language. Simulations are constructed and then tested with the shape of graph representation result. From the graph represented, after it is tested using simulation result, the solution evidently has the same trajectory and distance with manual counting. Thus, Dijkstra's algorithm simulation in dealing with it this problem on a graph using Visual Basic is already realized and it can be implemented in real life problems represented in a graph form and finded the shortest path.

✉ Alamat korespondensi:
E-mail: hanif.ilmim@gmail.com

ISSN 2252-6943

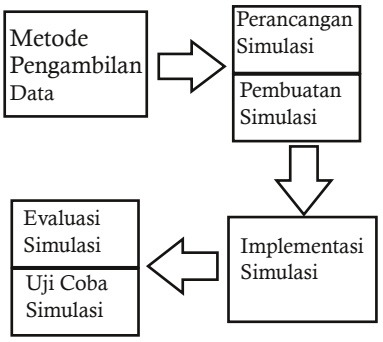
Pendahuluan

Menurut Budayasa (2007), graf adalah himpunan berhingga tak kosong $V(G)$ dari obyek-obyek yang disebut titik, dan himpunan berhingga (mungkin kosong) $E(G)$ yang elemen-elemennya disebut sisi, sedemikian hingga setiap elemen e dalam $E(G)$ merupakan pasangan tak berurutan dari titik-titik di $V(G)$. Teori graf merupakan salah satu cabang matematika yang mendapat perhatian saat ini. Model yang ada pada teori graf berguna untuk aplikasi yang luas dalam kehidupan sehari-hari di antaranya pewarnaan graf, penjodohan (*matching*), dan pencarian lintasan terpendek. Dibutuhkan algoritma untuk menyelesaikan masalah-masalah yang telah direpresentasikan dalam bentuk graf, seperti pencarian lintasan terpendek mempunyai beberapa algoritma untuk penyelesaiannya. Algoritma yang akan dibahas pada pencarian lintasan terpendek adalah algoritma Dijkstra. Algoritma Dijkstra sering disebut dan dikenal baik sebagai algoritma untuk memecahkan lintasan terpendek pada sebuah graf G(V,E) dengan bobot garis tidak negatif (Barbeheen, 1998).

Menurut Munir (2010), persoalan mencari lintasan terpendek di dalam graf merupakan salah satu persoalan optimasi. Dalam mencari lintasan terpendek, semakin banyak titik dan garis pada graf akan semakin rumit dan membutuhkan waktu lama ketika melakukan pencarian secara manual. Oleh karena itu, diperlukan adanya program pendukung dalam melakukan pencarian lintasan terpendek pada graf untuk mempercepat pencarian. Program yang dirancang adalah berupa simulasi pencarian lintasan terpendek pada graf. Simulasi tersebut dirancang bertujuan untuk memberi kemudahan dan menghindari kesalahan penghitungan bagi orang awam yang ingin mengimplementasikan pencarian lintasan terpendek dalam kehidupan sehari-hari. Permasalahan yang dibahas adalah cara membangun simulasi algoritma Dijkstra dalam mencari lintasan terpendek pada suatu graf menggunakan bahasa pemrograman Visual Basic dan kecocokan hasil antara penghitungan manual dengan simulasi pencarian lintasan terpendek pada graf.

Metode Penelitian

Metode pembuatan simulasi menggunakan bahasa pemrograman Visual Basic dengan algoritma Dijkstra dapat digambarkan dengan langkah-langkah penelitian seperti Gambar 1.



Gambar 1 Langkah-langkah pembuatan simulasi

Lintasan Terpendek (*Shortest Path*)

Menurut Brandes (2001), definisi sebuah lintasan (*path*) dari $s \in V$ sampai $t \in V$ adalah sebuah rangkaian titik-titik dan garis, dimulai dengan s dan berakhir di t dengan masing-masing garis menghubungkan titik-titik tersebut. Panjang sebuah lintasan adalah jumlah dari bobot pada garis-garis tersebut. Graf yang digunakan dalam pencarian lintasan terpendek adalah graf berbobot (*weighted graph*), graf berbobot adalah graf yang setiap sisinya diberi sebuah harga (Nugraha, 2011).

Algoritma Dijkstra

Menurut Wibowo & Wicaksono (2012), algoritma adalah sebuah prosedur komputasi yang mentransformasikan sejumlah input menjadi sejumlah output. Sebuah algoritma dikatakan "benar (*correct*)" jika untuk setiap inputnya menghasilkan output yang benar pula.

Berikut algoritma Dijkstra untuk mencari lintasan terpendek.

Input : Graf bobot G dengan $s,t \in V(G)$.

Step 1 : Label titik dengan $\lambda(s) = 0$ dan untuk setiap titik v di G selain s, label titik v dengan $\lambda(v) = \infty$. Dalam praktek ∞ diganti dengan bilangan yang sangat besar. Tulis $T = V(G)$.

Step 2 : Misalkan $u \in T$ dengan $\lambda(u)$ minimum.

Step 3 : Jika u = t, STOP, dan beri pesan: "Panjang lintasan terpendek dari s ke t adalah $\lambda(t)$".

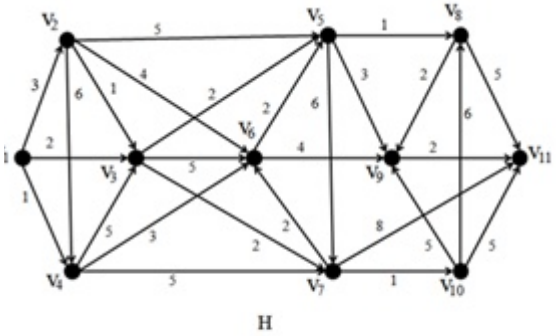
Step 4 : Untuk setiap garis e = uv, $v \in T$, ganti label v dengan $\lambda(v) = \text{minimum}\{\lambda(v), \lambda(u) + w(e)\}$.

Step 5 : Tulis $T = T - \{u\}$, dan kembali ke step 2.

Proses Pencarian Lintasan Terpendek Menggunakan Algoritma Dijkstra

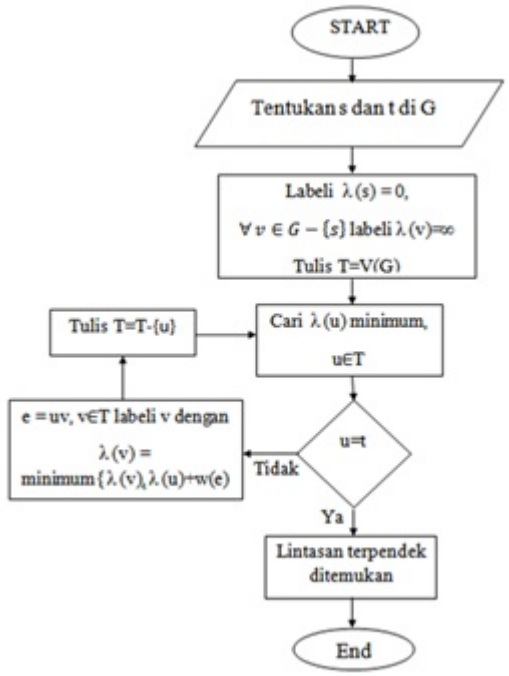
Berikut ini contoh penerapan algoritma Dijkstra dalam mencari lintasan terpendek pada suatu graf.

Input : Graf bobot G dengan s,t ∈ V(G)



Gambar 2 Representasi graf H

Gambar 2 menunjukan representasi graf yang akan dicari lintasan terpendek dari titik v_1 sebagai titik awal (s) ke titik v_{11} sebagai titik tujuan (t). Pencarian secara manual lintasan terpendek menggunakan algoritma Dijkstra pada graf H dijelaskan menggunakan *flowchart* seperti Gambar 3.



Gambar 3 *Flowchart* algoritma Dijkstra

Gambar 3 merupakan alur pengerjaan pencarian lintasan terpendek menggunakan algoritma Dijkstra jika dikerjakan secara manual. Hasil lintasan terpendek diperoleh dengan cara metode telusur balik yaitu dari v_{11} ke v_1.

Hasil dan Pembahasan

Flowchart pencarian lintasan terpendek menggunakan algoritma Dijkstra yang telah dirancang membuat logika berfikir pembuatan program tersusun dengan baik. *Flowchart* program yang dirancang kemudian diterjemahkan dalam bentuk *source code* dengan menggunakan bahasa pemrograman Visual Basic. *Source code* algoritma Dijkstra menggunakan bahasa pemrograman Visual Basic seperti tampak pada Gambar 4.

```
Private Sub cmdAnalisisDijkstra_Click()
    prepareFSP
    awl = getIndexOfTabName(Awal)
    ahr = getIndexOfTabName(Akhir)
    If (awl = -1) Or (ahr = -1) Then
        MsgBox "silahkan cek kembali"
        Exit Sub
    End If
    flxS.Row = 1
    flxDist.Row = 1
    flxLintasan.Row = 1
    Dim MAX As Integer
    MAX = flxgraf.Cols
    Dim alur As Integer
    Dim jrk As Integer
    Dim i As Integer
    Dim min As Integer
    Dim pencarian As Boolean
    pencarian = True
    alur = awl
    jrk = 0
    flxS.TextMatrix(1, alur) = "True"
        flxS.Row = 1
        flxS.Col = alur
        flxS.CellForeColor = vbRed
        flxS.CellFontBold = True
    flxDist.TextMatrix(1, alur) = 0
    Do While pencarian
        If (min = INF) Then
            pencarian = False
        End If
        flxS.TextMatrix(1, alur) = "True"
            flxS.Row = 1
            flxS.Col = alur
            flxS.CellForeColor = vbRed
            flxS.CellFontBold = True
For i = 1 To MAX - 1
If((myVl(flxgraf.TextMatrix(alur,i))<>0)And_
(myVl(flxDist.TextMatrix(1, i)) > myV
(flxgraf.TextMatrix(alur, i)) + jrk)) Then
flxDist.TextMatrix(1, i) = myVl(flxgraf.
TextMatrix(alur, i) + jrk)
flxLintasan.TextMatrix(1, i) = alur
End If
      Next i
min = INF
 For i = 1 To MAX - 1
  If((myVl(flxDist.TextMatrix(1, i))<min)And
  (flxS.TextMatrix(1, i) = "False")) Then
  min = myVl(flxDist.TextMatrix(1, i))
  alur = i
  jrk =myVl(flxDist.TextMatrix(1, i))
    End If
      Next i
    Loop
```

Input

Step 1

Step 2

Step 4

Step 5

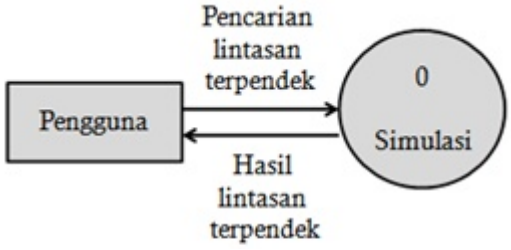
```
iRES_SIZE = 0
makeAllLines_Black
lblhasil.Caption = "Lintasan:"
alur = ahr
Do While alur <> awl
 If(flxLintasan.TextMatrix(1,alur)="0")Then
  lblhasil.Caption="Tidak Ada Lintasan"&
  flxgraf.TextMatrix(0, awl)&" ke"&
  flxgraf.TextMatrix(0, ahr)& "!"
  lbljarak.Caption = ""
  Exit Sub
  End If
  lblhasil.Caption = lblhasil.Caption &
  flxgraf.TextMatrix(0, alur)
  addTO_RESULT (alur)
  lblhasil.Caption=lblhasil.Caption &"<-"
  alur = myVl(flxLintasan.TextMatrix(1, alur))
  Loop
  lblhasil.Caption = lblhasil.Caption &
  flxgraf.TextMatrix(0, awl)
  addTO_RESULT (awl)
  lbljarak.Caption=flxDist.TextMatrix(1,ahr)
markLINES
End Sub
```

Step 3

Gambar 4. *Source code* algoritma Dijkstra

Desain Program

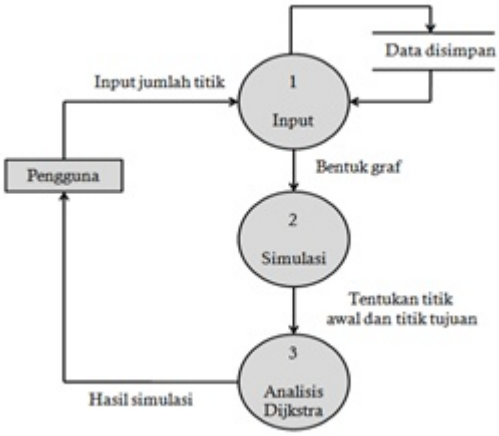
Pada tahap selanjutnya dilakukan pembuatan desain perangkat lunak yang akan dibuat. Desain dapat didefinisikan sebagai proses penerapan berbagai macam-macam teknik dan prinsip dengan tujuan untuk mendefinisikan proses atau sistem secara rinci sehingga mudah dalam penerapannya. Rincian desain yang digunakan dimulai dengan membuat *context diagram*.



Gambar 5 *Data Flow Diagram* (DFD) Level 0

Gambar 5 merupakan *context diagram* atau yang disebut dengan *Data Flow Diagram* (DFD) Level 0 merupakan alat yang digunakan untuk mendokumentasikan proses dalam sistem. Tujuannya adalah memberikan pandangan proses dalam sistem secara umum. Pengguna adalah orang yang dapat menjalankan simulasi untuk meminta pemecahan masalah lintasan terpendek dari kasus yang diinginkan.

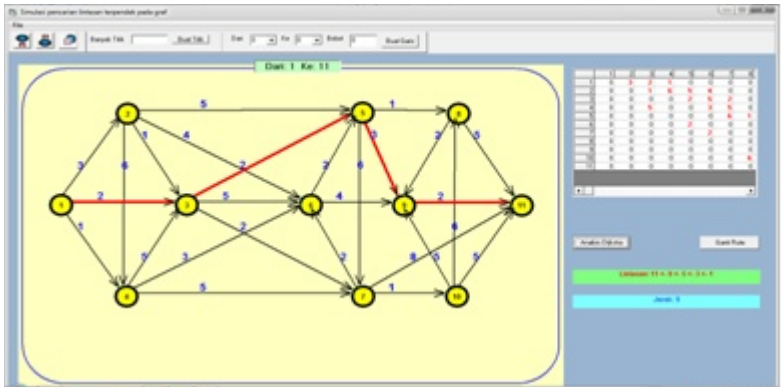
Tahap lanjutan adalah membuat DFD level 1 untuk lebih memperjelas jalannya program. Gambar DFD level 1 menjelaskan proses penggunaan simulasi dimulai dari penginputan data sampai diperoleh hasil dari kasus yang dinginkan seperti tampak pada Gambar 6.



Gambar 6 DFD level 1

Tahap Implementasi

Tahapan implementasi sistem mencakup pengkodean dan pengujian program. Desain yang telah dirancang dimulai dari *flowchart* sampai DFD level 1 kemudian diterjemahkan ke dalam kata-kata bahasa pemrograman dengan menggunakan bahasa pemrograman Microsoft Visual Basic 6.0. Adapun tampilan hasil implementasi program simulasi algoritma Dijkstra dalam menangani masalah lintasan terpendek pada graf berdasarkan desain yang telah dirancang akan tampak seperti pada Gambar 7.



Gambar 7 Tampilan simulasi pencarian lintasan terpendek

Gambar 7 merupakan tampilan simulasi hasil pencarian lintasan terpendek dari titik v_1 sebagai titik awal dan v_{11} sebagai titik tujuan. Alur lintasan terpendek yang dihasilkan oleh simulasi ditandai dengan garis tebal warna merah. Perepresentasian graf ke dalam simulasi didukung oleh beberapa tool seperti tampak pada Gambar 8.

Gambar 8 *Toolbar* untuk membuat representasi graf

Toolbar dalam Gambar 8 terdapat beberapa *tool* yang memudahkan *user* dalam merepresentasikan suatu data kedalam bentuk graf. Adapun macam-macam *tool* yang disediakan adalah sebagai berikut.

1. *Upper tool* ( )

Tool ini berfungsi untuk memberi identitas titik tepat berada diatas titik yang dipilih.

2. *Lower tool* ( )

Tool ini berfungsi untuk memberi identitas titik tepat berada dibawah titik yang dipilih.

3. *Eraser tool* ( )

Tool ini berfungsi untuk menghapus titik yang diseleksi.

4. *Vertex tool*

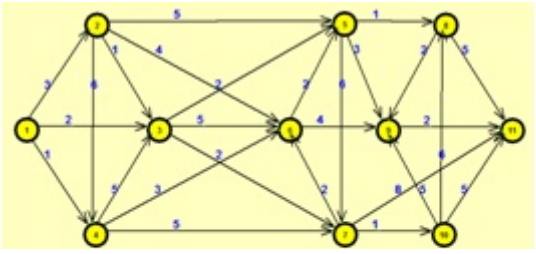
Gambar 9 *Tool* pembuat titik

Tool pada Gambar 9 merupakan langkah pertama yang harus dipilih ketika sebuah data akan direpresentasikan ke dalam bentuk graf. Karena *tool* ini berfungsi membuat sejumlah titik yang diperlukan dalam representasi sebuah graf di *work area*.

5. *Arc tool*

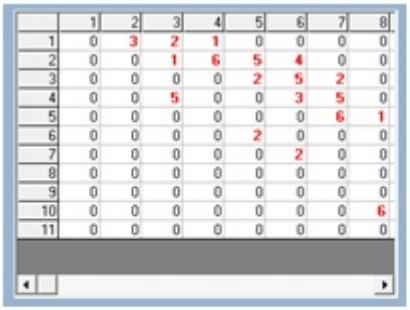
Gambar 10 *Tool* pembuat busur

Tool pada Gambar 10 berfungsi untuk membuat busur dan bobot pada titik yang telah dipilih. Berikut contoh hasil representasi graf menggunakan *toolbar* yang disediakan kedalam simulasi.



Gambar 11 Hasil representasi graf

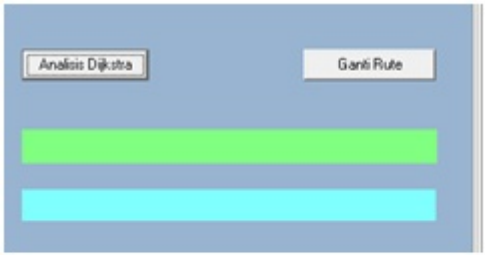
Gambar 11 merupakan contoh hasil representasi graf menggunakan *toolbar* yang disediakan oleh simulasi. Bobot garis pada graf yang direpresentasikan akan termuat ke dalam *form* data matriks. *Form* data matriks pada simulasi tampak seperti Gambar 12.



	1	2	3	4	5	6	7	8
1	0	3	2	1	0	0	0	0
2	0	0	1	6	5	4	0	0
3	0	0	0	0	2	5	2	0
4	0	0	5	0	0	3	5	0
5	0	0	0	0	0	0	6	1
6	0	0	0	0	2	0	0	0
7	0	0	0	0	0	2	0	0
8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	6
11	0	0	0	0	0	0	0	0

Gambar 12 *Form* data matriks

Gambar 12 merupakan tampilan *form* data matriks. Semua bobot pada graf yang direpresentasikan akan termuat ke dalam *form* data matriks. Setelah data direpresentasikan dalam bentuk graf dan semua bobot sudah masuk kedalam *form* data matriks kemudian dilakukan pencarian lintasan terpendek. *Tool* yang mendukung proses pencarian lintasan terpendek tampak seperti pada Gambar 13.



Gambar 13 *Form* pencarian lintasan terpendek

Tool analisis Dijkstra pada Gambar 13 *form* pencarian lintasan terpendek berfungsi untuk mencari lintasan terpendek pada graf yang telah direpresentasikan. *Tool* ganti rute disini berfungsi untuk mengembalikan representasi graf keadaan semula sebelum dilakukan proses pencarian.

SIMPULAN

Dari pembahasan dan analisis yang telah dilakukan, simulasi algoritma Dijkstra untuk menangani masalah pencarian lintasan terpendek pada suatu graf yang dibangun menggunakan bahasa pemrograman Visual Basic telah selesai dirancang dan direalisasikan. Simulasi tersebut mampu menemukan lintasan terpendek dan jarak minimum dari titik awal ke titik tujuan pada graf yang direpresentasikan ke dalam program simulasi.

Simulasi pencarian lintasan terpendek berdasarkan hasil pengujian program terbukti mempunyai hasil solusi yang sama antara perhitungan manual dengan menggunakan program simulasi dalam mencari lintasan terpendek pada suatu graf.

DAFTAR PUSTAKA

Barbehen, M. 1998. A Note on the Complexity of Dijkstra's Algorithm for Graph with Weighted Vertices. *IEEE Transactions on Computers*, Vol. 47, No.2 (IEEECS: 106062). Tersedia di http://www. researchgate.net/publication/3043930_A_note_on_the_complexity_of_Dijkstra%27s_algorithm_for_graphs_withweighted_vertices [diakses 7-6-2013].

Brandes, U. 2001. A Faster Algorithm for Betweenness Centrality. Journal of Mathema tical Sociology. *Department of Computer dan Informatika Science*, 25(3): 163-177. Tersedia di http://www.tandfonline.com/doi/abs/10.1080/0022250X.2001.9990249#preview [di akses 7-6-2013].

Budayasa, K. 2007. *Teori Graph dan Aplikasinya*. Surabaya: Unesa University Press.

Munir, R. 2010. *Matematika Diskrit*. Bandung: Informatika Bandung.

Nugraha, W. D. 2011. Aplikasi Algoritma Prim untuk Menentukan Minimum Spanning Tree Spanning Tree Suatu Graf Berbobot dengan Menggunakan Pemrograman Berorientasi Objek. *Jurnal Ilmiah Foristek*, Vol. 1, No.2. Tersedia di http://jurnal.untad.ac.id/jurnal/index.php/FORISTEK/article/view/702 [diakses 17-8-2013].

Wibowo, G. A, & A. P. Wicaksono. 2012. Rancang Bangun Aplikasi untuk Menentukan Jalur Terpendek Rumah Sakit di Purbalingga dengan MetodeAlgoritma Dijkstra. *JUITA*, Vol 11:21, No. 1(ISSN: 2086-9398). Tersedia di http://jurnal.ump.ac.id/index.php/juita/article/view/454 [diakses 9-6-2013].