UTILIZATION OF THE PRIM ALGORITHM TO DETERMINE THE NEAREST PATH CAR TRANSPORTATION PROBLEMS OF GOODS CARRIER BOX

Paryati[1], *Krit* Salahddine[2]

[1]UPN "Veteran" Yogyakarta, Country Indonesia, Email: yaya_upn_cute@yahoo.com
[2]Ibn Zohr University, Agadir, Country Marocco, Email: salahddine.krit@gmail.com

Abstract
Prim's algorithm has been proven to be able to be used in determining the closest path, minimum spanning tree in the problem of transporting box cars carrying goods. Based on the results of theoretical analysis tests that have been carried out from several experimental data, it can be obtained results showing that, the results of Prim's algorithm complexity are as follows O(n2). By defining n here is the number of vertices contained in the graph. So it can be concluded that, empirically, the data from the average travel time of this prim algorithm shows a graph that has a quadratic nature. As for the large number of edges contained in a graph, it has no effect on the complexity of finding the nearest path, the problem of transporting box cars carrying goods is a minimum spanning tree using the Prims algorithm.
Keyword: Prim Algorithm, Transportation Problem.

I. Introduction
Prim's algorithm is one of the algorithms for finding the minimum spanning tree. This problem has many applications including to determine transportation routes between cities in an area. For example, suppose that the vertices in graph G represent cities, assume the weight of each edge e= (a,b) is the transportation cost from city a to city b, then the minimum spanning tree of graph g corresponds to the transportation distance with the lowest cost connecting transportation routes throughout the city.

II. Transportation Problems
The transportation problem discusses the delivery of commodity goods from several sources (cities) to a number of destinations (cities). Each source and destination has a certain amount of supply and demand for commodity goods. The purpose of solving this problem is to allocate inventory from each source to meet the needs of each goal in such a way as to optimize the specified criteria. The objective function is to find the shortest overall transportation distance so as to minimize transportation costs from each source to each destination. In addition, it can maximize the total profit from the allocation of the transportation distance.

III. Minimum Spanning Tree
Given the graph G=(V,E) an undirected connected graph where V is the set of vertices and E is the set of edges. Each edge has a non-negative weight. The problem is how to find T, a subset of the set of edges in G so that all the vertices are connected and only use the edges that exist in T, and the sum of the edge weights on T is as small as possible. Since G is connected, there is at least one solution. If G has an edge weight of 0, then there are several solutions that have the same total weight using different edges. In one case the weight of an edge can be associated as a cost to each edge.

Take G'=(V,T') the subgraph formed by the vertices on G and the edges on T, and assume there are n vertices in N. A connected graph with n vertices has at least n-1 edges, so this is the minimum

number of edges on T. A graph with n vertices and more than n-1 edges has at least one cycle. If G' is connected and T has more than n-1 edges, at least one edge can be removed without breaking G', i.e. selected edges that are part of the cycle. This will result in the total weight on T being reduced or equal (if the edge is omitted with a weight of 0) by decreasing the number of edges on T. Therefore, the set T with n or more edges cannot be optimal. In the end, T must have n-1 edges and since G' is connected it must be a tree. The graph G' is called the minimum spanning tree for graph G.

IV. Prim Algorithm

Prim found an algorithm to determine the minimum spanning tree on a weighted graph (with positive integer weights), and connected in addition to Kruskal's algorithm.

The algorithm is as follows:

¬ Step 0. Given a connected and weighted graph with n vertices, G =(V,E). All vertices are labeled "unchosen"; arrange T:= V, a graph with n vertices and no edges; select any vertex and label it "chosen".

¬ Step 1. If there is a vertex labeled "unchosen", do step 2.; Stop.

¬ Step 2. Suppose (u,v) is the lightest edge between any vertex labeled "chosen" u and any vertex labeled "unchosen" v, label v with "chosen"; and arrange T : = T {(u,v)}

As an illustration, suppose you have a graph as shown in Figure1 below:

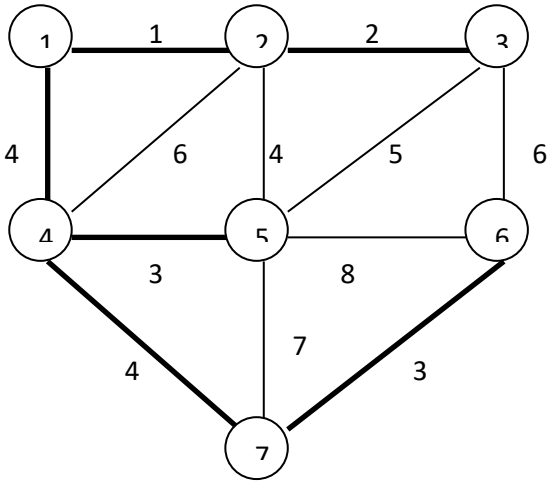


Figure 1. A minimum spanning tree graph with thick edges.

To illustrate Prim's algorithm, see Figure 1. For example, by selecting vertex 1 as the initial vertex. Based on the algorithm above, these steps can be written in table 1 below:

Table.1 Steps of a graph minimum spanning tree has thick edges.

Langkah	**e=(u,v)**	**{"chosen"}**
Inisialisasi	-	{1}
1	(1,2)	{1,2}
2	(2,3)	{1,2,3}

3	(1,4)	{1,2,3,4}
4	(4,5)	{1,2,3,4,5}
5	(4,7)	{1,2,3,4,5,7}
6	(7,6)	{1,2,3,4,5,6,7}

When the algorithm stops because there are no vertices labeled unchosen, then T contains the selected edges (1,2), (2,3), (1,4), (4,5), (4,7), and (7,6). The choice of vertex will affect the algorithm, but in general vertex 1 is chosen as the starting point.
Some theorems of Prim's algorithm which are based on the properties of the tree are:
1. At each execution of step 2., the edges in T form a tree among the set of vertices labeled "chosen".
2. At the end of Prim's algorithm implementation, the T that will be obtained is the spanning tree of graph G.
3. Suppose T is a sub-tree of graph G and let e be the lightest edge connecting the vertices in T and the vertices outside T. There is a spanning tree T' in G which contains T and e so that if T" is any spanning tree of g containing T, then W(T') W(T").
4. If G(V,E) is a connected and weighted graph, and e = (v,w) is the lightest edge that is tangent to vertex v, then there is a minimum spanning tree T containing e.
5. Prim's algorithm determines a minimum spanning tree in a connected and weighted graph G with n vertices.
6. The complexity of Prim's algorithm is O(n2).

V. Experiment

To see the complexity of Prim's algorithm in determining the minimum spanning tree, an experiment was carried out using a computer 2.30 GHz Processor, 3 MB Intel @ Smart Cache Cache, Bus Speed 5 GT / s, TDP 35 W, the data generated by the program automatically was obtained as follows:

Table 2. Experimental data I Computer Processor 2.30 GHz, Cache 3 MB Intel@ Smart Cache, Bus Speed 5 GT/s, TDP 35 W, Number of Cores 2,Number of Strings 4.

n **(Banyak vertex dalam graf)**	**T(n)** **Rata-rata waktu tempuh (detik)**
10	0.0001
20	0.0001
30	0.0280
40	0.0480
50	0.0480
60	0.0910
70	0.1028
80	0.1551
90	0.2028
100	0.2228

From these data, the following graph can be obtained:

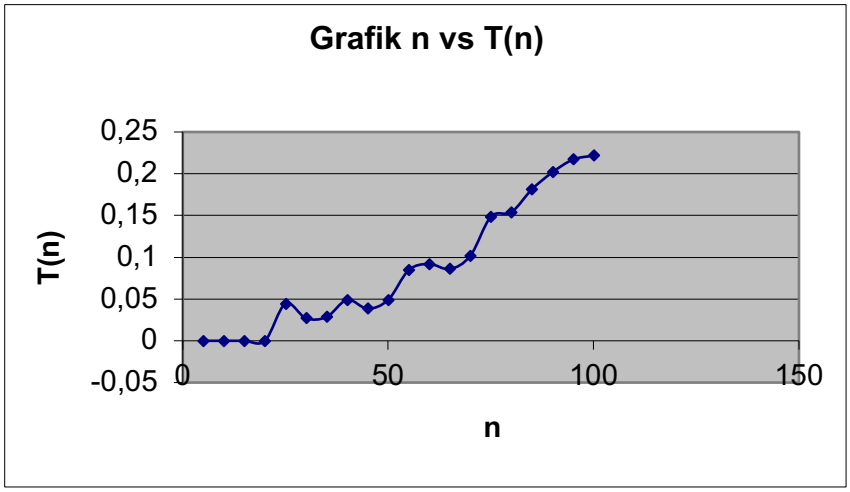


Figure 2. Graph of n vs T(n) in experiment I

In the experiment using a computer type Processor 1.90 GHz, Cache 6 MB Intel @ Smart Cache, Bus Speed 5 GT / s, TDP 35 W, Max Turbo Frequency 2.70 GHz, Number of strands 4, Number of Cores 4 so that the following data is obtained:

Table 3. Experimental data II Computer Processor 1.90 GHz, Cache 6 MB Intel@ Smart Cache, Bus Speed 5 GT/s, TDP 35 W, Max Turbo Frequency 2.70 GHz, Number of strands 4, Number of Cores 4.

n **(Banyak vertex dalam graf)**	**T(n)** **Rata-rata waktu tempuh (detik)**
5	0
10	0
15	0,007
20	0,022
25	0,057
30	0,048
35	0,051
40	0,067
45	0,074
50	0,082
55	0,115
60	0,137
65	0,141
70	0,181
75	0,196
80	0,241
85	0,28
90	0,295
95	0,42
100	0,352

From the data above, the following graph can be obtained:

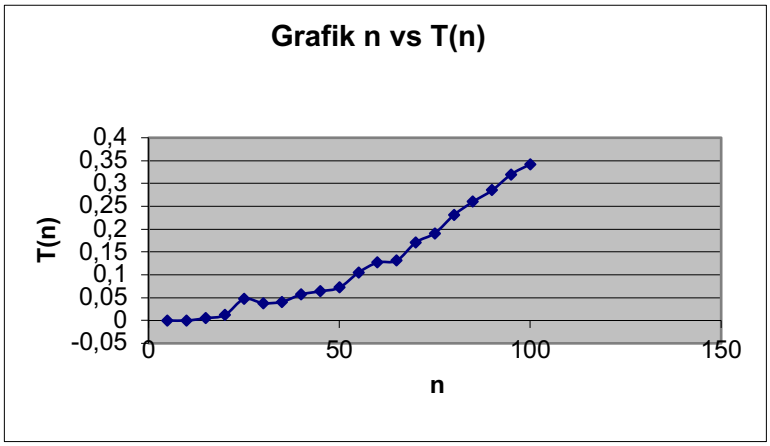


Figure 3. Graph of n vs T(n) in experiment II

VI. Curve Fittings

1. From the experimental data I above, a curve equation that represents the data can be derived. That is by using the curve fitting method. In this case using the polynomial regression method. The data is arranged into the following table:

Table 3. Data for finding the equation of the curve in the I . experiment

No	xi	yi	xi^2	xi^3	xi^4	xiyi	xi^2yi
1	5	0	25	125	625	0	0
2	10	0	100	1000	10000	0	0
3	15	0	225	3375	50625	0	0
4	20	0	400	8000	160000	0	0
5	25	0,0445	625	15625	390625	1,1125	27,8125
6	30	0,027	900	27000	810000	0,81	24,3
7	35	0,0285	1225	42875	1500625	0,9975	34,9125
8	40	0,049	1600	64000	2560000	1,96	78,4
9	45	0,0385	2025	91125	4100625	1,7325	77,9625
10	50	0,049	2500	125000	6250000	2,45	122,5
11	55	0,0846	3025	166375	9150625	4,653	255,915
12	60	0,092	3600	216000	12960000	5,52	331,2
13	65	0,0859	4225	274625	17850625	5,5835	362,9275
14	70	0,1018	4900	343000	24010000	7,126	498,82
15	75	0,1485	5625	421875	31640625	11,1375	835,3125
16	80	0,1541	6400	512000	40960000	12,328	986,24
17	85	0,1814	7225	614125	52200625	15,419	1310,615
18	90	0,2018	8100	729000	65610000	18,162	1634,58
19	95	0,217	9025	857375	81450625	20,615	1958,425
20	100	0,2218	10000	1000000	100000000	22,18	2218
Jumlah	1050	1,7254	71750	5512500	451666250	131,7865	10757,92

By using the polynomial regression method, it can be obtained a curve equation that represents these data, namely: T(n) = 0.0039n2 +0.37n+10.27

2. From the experimental data II above, a curve equation that represents the data can be derived. That is by using the curve fitting method. In this case using the polynomial regression method. The data is arranged into the following table:

Table 4. Data for finding curve equations in experiment II

No	xi	yi	xi^2	xi^3	xi^4	xi*yi	xi^2*yi
1	5	0	25	125	625	0	0
2	10	0	100	1000	10000	0	0
3	15	0,005	225	3375	50625	0,075	1,125
4	20	0,012	400	8000	160000	0,24	4,8
5	25	0,047	625	15625	390625	1,175	29,375
6	30	0,038	900	27000	810000	1,14	34,2
7	35	0,041	1225	42875	1500625	1,435	50,225
8	40	0,057	1600	64000	2560000	2,28	91,2
9	45	0,064	2025	91125	4100625	2,88	129,6
10	50	0,072	2500	125000	6250000	3,6	180
11	55	0,105	3025	166375	9150625	5,775	317,625
12	60	0,127	3600	216000	12960000	7,62	457,2
13	65	0,131	4225	274625	17850625	8,515	553,475
14	70	0,171	4900	343000	24010000	11,97	837,9
15	75	0,191	5625	421875	31640625	14,325	1074,375
16	80	0,231	6400	512000	40960000	18,48	1478,4
17	85	0,26	7225	614125	52200625	22,1	1878,5
18	90	0,285	8100	729000	65610000	25,65	2308,5
19	95	0,32	9025	857375	81450625	30,4	2888
20	100	0,342	10000	1000000	100000000	34,2	3420
Jumlah	1050	2,499	71750	5512500	451666250	191,86	15734,5

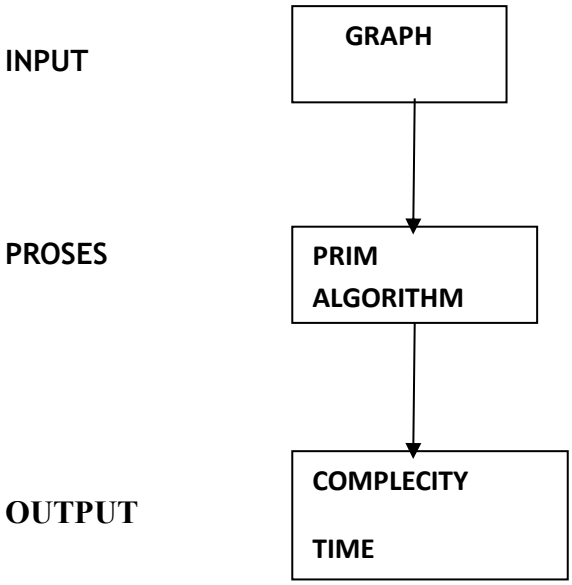
By using the polynomial regression method, it can be obtained a curve equation that represents these data, namely: T(n) = 0.0047n2 +0.47n+12.44

VII. Implementation

The following is the implementation of Prim's algorithm:

1. Block Diagram

Prim's algorithm can be described as follows:



The program is written on a computer with the following specifications:

a. Dell Inspiron AIO 5400/i7-1165G7/23.8-inch FHD computer
b. 8GB DDR4 RAM, 2666MHz
c. LG 19 INCH LED Monitor Screen
d. Windows 10 Operating System Software
e. Borland Delphi Programming Language version 7

The program has been designed to check the connectivity of a graph. An unconnected graph does not have a spanning tree. Checks are carried out at each stage of tracing the spanning tree. The graph is represented as a form of adjacency matrix.

2. Source Code

```
uses dos,crt;
const nPercobaan=2;
type Matriks = array [1..50,1..50] of longint;
    VektorBoolean  = array [1..50] of Boolean;
    VektorInteger  = array [1..50] of Integer;
Procedure BuatGraphConnected(nVertek,nEdge:integer;var Graph:Matriks);
var a,b,i,j:integer;
  Procedure Hubungkan;
  var i:integer;
  begin
    for i:=1 to nEdge do
    begin
      repeat
       a:=random(nVertek)+1;
       repeat
         b:=random(nVertek)+1;
       until b<>a;
```

```
        until graph[a,b]=10000;
        graph[a,b]:=random(200)+1;
        graph[b,a]:=graph[a,b];
      end;
   end;
begin
  {inisialisasi graph, 10000 menyatakan tak terhingga}
   for i:=1 to nVertek do
     for j:=1 to nVertek do

         Graph[i,j]:=10000;
   Hubungkan;
end;
Procedure Prim(var Graph:Matriks;nvertek:integer);
type edge = record
              v1,v2 : integer;
            end;
var B:VektorBoolean;
    i,j,nt,Min,k:integer;
    T:array[1..100] of edge;
    Nearest,Mindist: VektorInteger;
begin
  nt:=0;
  b[1]:=true;
  for i:=2 to nVertek do B[i]:=false;
  for i:=2 to nVertek do
  begin
    Nearest[i]:=1;
    Mindist[i]:=Graph[i,1];
  end;
  for i:=1 to nVertek-1 do
  begin
    min:=10000;
    for j:= 2 to nVertek do
      if (0<=mindist[j]) and (Mindist[j]<min) then
      begin
        min:=Mindist[j];
        k:=j;
      end;
    if min=10000 then
    begin
      writeln('Graph tidak terhubung');
      break;
    end
    else
    begin
```

```
      inc(nt);
      t[nt].v1:=nearest[k];
      t[nt].v2:=k;
      mindist[k]:=-1;
      for j:=2 to nVertek do
       if Graph[j,k]<mindist[j] then
       begin
         mindist[j]:=Graph[j,k];
         nearest[j]:=k;
       end;
    end;
  end;
end;
var i,Percobaan:integer;
    nVertek:integer;
    SumWaktu,rata:real;
    Graph:Matriks;
    hh,mm,ss,sec100:Word;
Begin
  clrscr;
  for i:=1 to 10 do
  begin
      nVertek:=5*i; { Jumlah vertek kelipatan 5}
      SumWaktu:=0;
      for percobaan:=1 to nPercobaan do  {Buat percobaan sebanyak 1 kali}
      begin
        SetTime(0,0,0,0);
        BuatGraphConnected(nVertek,(nVertek*nVertek) div 4,Graph);
        Prim(Graph,nVertek);
        GetTime(hh,mm,ss,sec100);
        SumWaktu:=SumWaktu+hh*3600+60*mm+ss+sec100/100;
      end;
      Rata:=SumWaktu/nPercobaan; {dalam detik}
      Writeln(nVertek:8,SumWaktu:8:2);
  end;
End.
```

VIII. Discussion

Based on the results of theoretical analysis, it has been obtained that the complexity of Prim's algorithm is O(n2). Experiments using computers with two types, namely Computer Processor 2.30 GHz, Cache 3 MB Intel@ Smart Cache, Bus Speed 5 GT / s, TDP 35 W, Number of Cores 2, Number of Strands 4.

In this experiment, data were obtained whose graphs tend to be quadratic. Likewise on computers Processor 1.90 GHz, Cache 6 MB Intel@ Smart Cache, Bus Speed 5 GT/s, TDP 35 W, Max Turbo Frequency 2.70 GHz, Number of strands 4, Number of Cores 4, the graphics tend to be quadratic. The experiment is done repeatedly starting from n = 5 to n = 100 where n is the number of vertices

in the graph. By performing the curve fitting method on the experimental data I (Pentium computer) the equation T(n) = 0.0039n2 +0.37n +10.27 is obtained. And for experimental data II (Processor 1.90 GHz, Cache 6 MB Intel@ Smart Cache, Bus Speed 5 GT/s, TDP 35 W, Max Turbo Frequency 2.70 GHz, Number of strands 4, Number of Cores 4) obtained the equation T(n) = 0.0047n2 +0.47n +12.44.

In principle, the implementation of the algorithm does not affect the complexity of an algorithm. Implementation on two computers with different clocks only affects the speed of graph growth/graph gradient. This can be seen from the gradient of change in

Prim's algorithm has no worst case and best case conditions because each case has the same complexity, which is O(n2).

In Prim's algorithm the number of edges has no effect because this algorithm checks / iterates over all vertices, whether there are edges or not. This happens because on all unchoosen nodes it will still be checked whether the edges are minimal, no matter if the edges are infinite (no connection). So what matters is the number of nodes.

IX. Conclusion

a. Prim's algorithm is proven to be able to be used to determine the minimum spanning tree in determining the closest path, minimum spannning tree in the problem of transporting box cars carrying goods.

b. Based on the results of theoretical analysis, it is found that the complexity of Prim's algorithm is O(n2) where n is the number of vertices in the graph.

c. Empirically, the data from the average travel time of this algorithm shows a quadratic graph.

d. The number of edges in a graph does not affect the complexity of finding the minimum spanning tree with Prims algorithm.

Bibliography

Brassard G., Bratley P., 2018, Fundamentals of algorithmics, Prentice-Hall, New Jersey.

Ding-Zhu Du, Ker-Iko, 2018, Theory of Computational Complexity, John Wiley & Son

Gilles Brassard, Paul Bratley, 2019, Fundamentals of Algorithmics, Prentice-Hall, Inc.

Triatmodjo B., 2019, Numerical Method, Beta Offset, Yogyakarta

Susila I., 2018, Fundamentals of Numerical Methods, Director General of Higher Education Yogyakarta

Steven C. Chapra, Raymond P. Canale, 2019, Numerical Method, Erlangga, Jakarta