Applications of The Shortest Spanning Tree and Path on Graph Theory

KHIN AYE TIN
Department of Mathematics, Technological University (Yamethin), Myanmar

Abstract -- The applications of graph theory have become an exciting research topic in recent years. The applications of the shortest spanning tree and shortest path are applications of graph theory to real life. The applications of these problems are significant in graph theory. These problems are applied to six villages in Chauk Township. The study was done by doing survey: using prepared questionnaire. My native village, Ku Phyu, is one of those six villages and easier to do the study. The key reference data for six villages were got from village tract administrator (Lay Tin Cone Village Tract). The aims of this paper is to develop the life of the people in all small villages such as six villages were expressed in Chauk Township and to help the rural road development for cost effectiveness. The study will develop the life of the people practically by using the shortest path linking among their villages. Some basic definitions and notations of graph theory are mentioned. And then, three algorithms are expressed clearly. Next, using these three algorithms, these problems are mentioned interestingly.

Indexed Terms— six villages, the shortest paths, the shortest spanning tree, weighted graph

I. INTRODUCTION

Applications of graph theory are the sun and the moon of human society. Because, if there were no application on earth, there would be less improvement or civilization on human society. So, the role of applications is very important to real life. There are many applications of graph theory. They are the shortest path problem, the shortest spanning tree problem, a geometry problem, a the optimal assignment problem, the Chinese postman problem, a storage problem, the timetabling problem, the travelling salesman problem, the connector problem and the others. Every application is always useful to the corresponding field of real life. Among them, the following problems are mentioned by using three Algorithms [1], [2], [3], [6].

There are many villages in Chauk Township. Among them, Ku Phyu village (my native village) is one of the developing village in Chauk Township. This small village has born many good citizens such as doctors, engineers, University teachers and the others in different fields for supporting to build a developing country. Because of this, those six villages are the main keys of this paper. . In this paper, graphs, the shortest spanning tree and the shortest paths are investigated with special conditions. This paper is organized with Some Basic Definitions and Notations of graph theory, TURAN'S Theorem, Kruskal's Greedy Algorithm , Dijkstra's Algorithm, Prim's Algorithm, the shortest path problem and the shortest spanning tree problem.

A. Some Basic Definition and Notations

Graphs and digraphs (=directed graphs) have developed into powerful tools in areas, such as electrical and civil engineering, communication networks, operation research, computer science, economics, industrial management and marketing. An essential factor of this growth is the use of computers in large-scale optimization problems that can be modeled by graphs and solved by algorithms provided by graph theory. This approach yields models of general applicability and economic importance.

A graph G consists of two finite sets (sets having finitely many elements), a set V of points, called vertices, and a sets E of connecting lines, called edges, such as each edge connects two vertices, called the end point of the edge. Write G = (V, E) as shown in figure 1[1], [2].

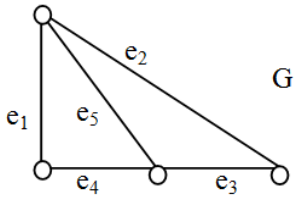


Figure 1. A diagram of graph G

A digraph G = (V, E) is a graph in which each edge e= (i ,j) has a direction from its " initial point " i to its "terminal point" j as shown in figure 2.

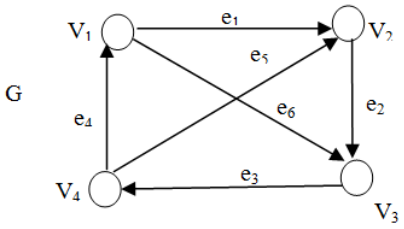


Figure 2. Digraph G

In a graph G = (V, E), can walk from a vertex V_1 along some edges to some other vertex V_k,

(A) make no restrictions, or

(B) require that each edge of G be traversed at most once, or

(C) require that each vertex be visited at most once.

In case (A), call this a walk. Thus a walk from $V_1$to $V_k$is of the form,

$(V_1,V_2), (V_2,V_3), (V_3,V_4),\ldots, (Vk\text{-}_1,V_k)$,

where some of these edges or vertices may be same. In case (B), where each edge may occur at most once, call the walk, a trail Finally, in case (C), where each vertex may occur at most once (and thus each edge automatically occurs at most once), call the trail, a path. Admit that a walk, trail, path may end at the vertex it started from, in which case, call it closed. A closed path is called a cycle. A cycle has at least three edges. Figure 3 illustrates all these concepts[1], [2].

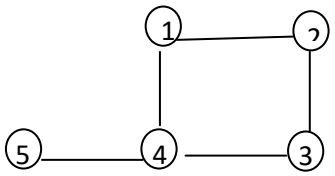


Figure3.walk, trail, path, cycle

1- 2- 3- 2 is a walk(not a trail)

2- 4-1-2-3-4-5 is a trail (not a path)

1-2-3-4-5 is a path (not a cycle)

1-2-3-4-1 is a cycle.

The concept of a shortest path, assume that G = (V,E) is a weighted graph that is, each edge (V_i, V_j) in G has a given weight or length Li j >0. Then a shortest path v1 to V_k (wih fixed V_1 and V_k) is a path such that the sum of the lengths of its edges as follow:

$L_{12}+L_{23}+L_{34}+\ldots+L_{k-1,k}$, is minimum(as small as possible among all paths from V_1 to V_k).

A tree T is a graph that is connected and has no cycle. A spanning tree T in a given connected graph G = (V, E) is a tree containing all the n vertices of G. Such a tree has n-1 edges. A shortest spanning tree T is a connected graph G (whose edges(i, j) have lengths $L_{ij}>0$) is a spanning tree for which $\sum Lij$ (some over all edges of T) is minimum compare to $\sum Lij$ for any other spanning tree in G. [1], [2].

With each edge e of G , let there be associated a real number w(e), called its weight .Then G, together with these weights on its edges, is called a weighted graph.

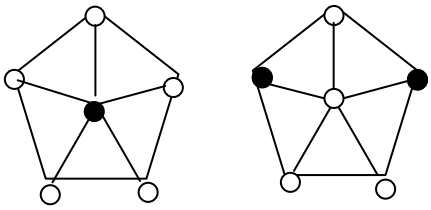
Weighted graphs occur frequently in applications of graph theory. In the friendship graph, for example, weights might indicate intensity of friendship; in the communications graph, they could represent the construction or maintenance costs of the various communication links.

If H is a subgraph of a weighted graph, the weight w(H) of H is the sum of the weights $\sum_e w(e)$ on itsedges. Many optimization problems amount to finding, in a weighted graph, a subgraph of a certain type with minimum (or maximum) weight. One such is the shortest path problem: given a railway network connecting various towns, determine a shortest route two specified towns in the network.

In a weighted graph, a path of minimum weight connecting two specified vertices u_o and v_o; the weights represent distances by rail between directly-linked towns, and are therefore non negative.

A subset S of v is called an independent set of G if no two vertices of S are adjacent in G. An independent

set is maximum if G has no independent set S' with $|S'|>|S|$ as shown in figure 4.



(a) (b)

Figure 4. (a) An independent set

(b)A maximum independent set

Recall that a subset K of V such that every edge of G at least one end in K is called a covering of G.The two examples ofindependent set given in figure 4 are both complements of covering[3], [4].

A set S ⊆ V is an independent set of G if and only if V\ S is a covering of G.

The number of vertices in a maximum independent set of G is called the independent number of G and is denoted by $\propto(G)$; similarly, the number of vertices in a minimum covering of G is called the covering number of G and is denoted by β(G)[3], [5].

TURAN'S THEOREMS

If G is simple and contains no Km+1, then

$\varepsilon(G) \leq \varepsilon(T_{m,v})$. Moreover, $\varepsilon(G) \leq \varepsilon(T_{m,v})$ only if

$G \cong T_{m,v}$.

If a simple graph G contains no $\boldsymbol{K_{m+1}}$, then G is degree-majorised by some complete m-partite H. Moreover, if G has the same degree sequence as H, *then G*≅ $\boldsymbol{H}$[**3**], [**7**].

Bellman's Minimality Principle or Optimality Principle is a Principle if Pj:1→j is a shortest path from 1 to j in G and (i,j) is the last edge of Pj ,then Pi:1→i is a shortest path 1→i.

A path that contains every vertex of G is called a Hamiltonian path (Hamilton path) of G. A hamiltonian cycle (Hamilton cycle) of G is a cycle that contains every vertex of G. A graph is hamiltonian if it contains a Hamilton cycle.

A graph H is a subgraph of G and then (Written H ⊆G),if V(H) ⊆V(G), E(H) ⊆E(G) and $\boldsymbol{\varphi}_H$ is the restriction of $\boldsymbol{\varphi}_G$ to E(H). When H⊆G but H ≠ G, we write H ⊂ G and call H a proper subgraph of G. If H is a subgraph of G, G is a supergraph of H. A spanning subgraph (or spanning supergraph) of G is a subgraph (or supergraph) H with V(H) = V(G).

II. KRUSKAL'S GREEDY ALGORITHM

The following table1 ISKruskal's greedy algorithm for shortest spanning tree.

Table 1. Kruskal's Greedy Algorithm for Shortest Spanning Tree.

ALGORITHM KRUSKAL [G= (V, E), Lij for all(i,j) in E]
Given a connected graph G= (V, E) with edges (i, j) having length Lij > 0, the algorithm determines a shortest spanning tree T in G.

INPUT: Edges (i, j) of G and their length Lij

OUTPUT: Shortest spanning tree T in G.

1. Oder the edges of G and their length Lij.

2. Find all unlabeled vertices adjacent to a vertex labeled i.
   1. Label the vertices just found with i+1.
   2. Choose them in this order as edges of T, rejecting an edge only if it forms a cycle with edges already chosen.
      If n-1 edges have been chosen, then OUTPUT T (=the set of edges chosen). Stop.

End KRUSKAL

A. The Shortest Spanning Tree Problem

The Shortest Spanning Tree Problem is a problem of finding a shortest spanning tree for some example of railway lines connecting a number of cities (the vertices) can be set up in the form of a spanning tree, the "length" of a line (edge) being the construction cost, and one wants to minimize the total construction cost. Similarly for bus lines, where "length" may be the average annual operating cost. Or for steamship line (freight lines), where "length" may be profit and the goal is the maximization of total profit. Or in a network of telephone lines between some cities, a shortest

spanning tree may simply represent a selection of lines that connect all the cities at minimum cost.

As an example, consider the table 2 of motorcycle line distances in miles between six of the villages in Chauk township; Ku Phyu(K), Lay TinCone(L),Ywarma(Y),Wunlo(W),Taungthar(Tt) and Thaykhaehla(Tkh) [**6**].

Table 2. Motorcycle line distances between six villages in Chauk Township

	K	L	Y	W	Tt	Tkh
K	-	2.3	2.6	2.4	4.6	7.5
L	2.3	-	4.9	2	5	6.5
Y	2.6	4.9	-	3.5	4.4	7.4
W	2.4	2	3.5	-	3	5.5
Tt	4.6	5	4.4	3	-	4
Tkh	7.5	6.5	7.4	5.5	4	-

Consider the weighted graph shown in figure 5; it is the same graph as was used in our illustration of Kruskal's Greedy Algorithm [**3**].

Starting with the Vertices K ,Y ,L, W, Tt .Tkh, can be applyed by using Kruskal's Greedy Algorithm as illustrate in figure 6 is the graph of the shortest spanning tree.

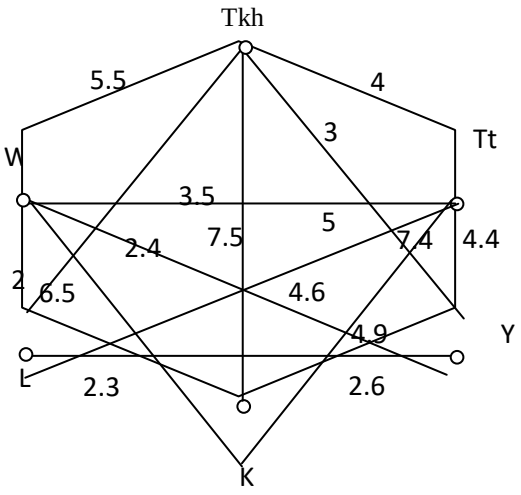


Figure 5. The Weighted Graph

Table 3. Solution of figure 6 by using Kruskal's Greedy Algorithm

Edge	Length	Choice
(L,W)	2	1st
(L,K)	2.3	2nd
(K,W)	2.4	Reject
(K,Y)	2.6	3rd
(W,Tt)	3	4th
(W,Y)	3.5	Reject
(Tt,Tkh)	4	5th
(Y,Tkh)	4.4	Reject
(K,Tt)	4.6	Reject
(L,Tt)	4.9	Reject
(L,Tt)	5	Reject
(W,Tkh)	5.5	Reject
(L,Tkh)	6.5	Reject
(Y,Tkh)	7.4	Reject
(K,Tkh)	7.5	Reject

By using Kryskal's Greedy Algorithm, the table 3 gives the solution of figure 5 as the shortest spanning tree. It can express as shown in figure 6. It is the solution of the shortest spanning tree, $\sum L$ =13.9.

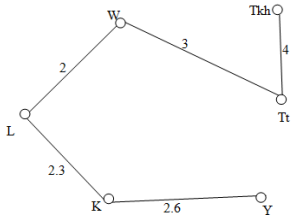


Figure 6. The Graph of Shortest Spanning Tree

III. DIJKSTRA'S ALGORITHM FOR SHORTEST PATH

Dijkstra's Algorithm is shown in Table 4, where a connected graph G is a graph in which for any two vertices v to w in G is a path v→w. The Algorithm is a labeling procedure. At each stage of the computation, each vertex v get a level, either (PL) a permanent Label= length L_v of a shortest path 1→v or (TL) a temporary Label= upper bound $\widetilde{L_v}$ for the length of a shortest path 1→v.

Table 4. Dijkstra's Algorithm for Shortest Path

ALGORITHM DIJKSTRA[G=(V,E),

V={1,2,…,n},Lij for all (i,j) in E]

Give a connected graph G=(V,E) with vertices 1,…,n. and edge (i,j) having lengths Lij>0' this algorithm determines the length of the shortest paths from vertex 1to the vertices 2, …,n.

INPUT: Number of vertices n, edges (I,j) and

lengths Lij

OUTPUT: lengths Lj of shortest paths 1→j, j=2,….n

1. Initial step
   Vertex 1gets $\mathcal{PL}$: L1=0.
   Vertex j(2,…,n) get $\mathcal{TL}$: $\widetilde{L_j}$=L1j(=∞ if there is no edge (1,j) in G).
   $\mathcal{PL}$={1},$\mathcal{TL}$={2,…,n}.
2. Fixing a permanent Label
   Find a k in $\mathcal{TL}$ for which L_k is minimum, set $L_k=\widetilde{L_k}$. Take the smallest k if there are several. Delete k from $\mathcal{TL}$ and include it in $\mathcal{PL}$.
   If $\mathcal{TL}$ =∅(that is, TLis empty)then OUTPUT l2,…,ln. Stop.
   Else continue (that is go to step 3).
3. Updating temporary Labels
   For all j in $\mathcal{TL}$, set $\widetilde{L_j}$ =min($\widetilde{L_j}$, L_k+L_{kj}) (that is ,take the smaller of $\widetilde{L_j}$and

L_k+L_{kj} as your new $\widetilde{L_j}$)

Go to Step 2.

End DIJKSTRA

Denoted by $\mathcal{PL}$ and $\mathcal{TL}$ the sets of the vertices with a permanent Label and a temporary Label, respectively. The Algorithm has an initial step in which vertex 1 get the temporary Labels, and then the Algorithm alternates between step 2 and 3. In step 2, the idea is to pick k "minimally". In step 3, the idea is that the upper bounds will in general improve(decrease) and must be updated accordingly. Namely, the new temporary Label $\widetilde{L_j}$of vertex j will be the old one if there is no improvement or it will be L_k+L_{kj} if there is.

As an example, we consider the weighted graph shown in figure 5 to transform as figure 7 ; it is same graph as was used our construction of the shortest spanning tree as shown in figure 8, here ,assume K=1, L=2, Y=3, W=4, Tt=5, Tkh=6.

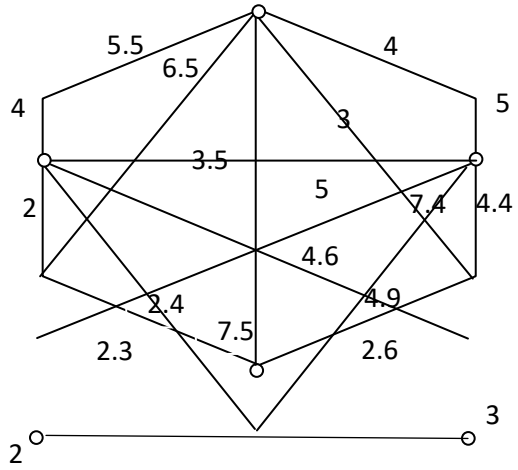


Figure 7. The Weight Graph

1

Solution

1. L_1= 0,$\widetilde{L_4}$=2.3,$\widetilde{L_4}$=2.6, $\widetilde{L_4}$=2.4,$\widetilde{L_5}$=4.6,$\widetilde{L_6}$=7.5
   $\mathcal{PL}$ = {1}$\mathcal{TL}$ = {2, 3, 4, 5, 6}, PL={1}
2. L_2=min{$\widetilde{L_2}$,$\widetilde{L_3}$,$\widetilde{L_4}$,$\widetilde{L_5}$,$\widetilde{L_6}$}=2.3
   K=2, $\mathcal{PL}$ = {1, 2}, $\mathcal{TL}$ = {3, 4, 5, 6},
3. $\widetilde{L_3}$=min{$\widetilde{L_3}$,L_2+L_{23}}
   = min {2.6, 2.3+4.9} = 2.6
   $\widetilde{L_4}$=min {$\widetilde{L_4}$, L_2+L_{24}}
   = min {2.4, 2.3+2} = 2.4
   $\widetilde{L_5}$=min {$\widetilde{L_5}$, L_2+L_{25}}
   = min {4.6, 2.3+5} = 4.6
   $\widetilde{L_6}$=min {$\widetilde{L_6}$, L_2+L_{26}}
   = min {7.5, 2.3+6.5} = 7.5
2. L_4= min{$\widetilde{L_3}$,$\widetilde{L_4}$,$\widetilde{L_5}$,$\widetilde{L_6}$}=2.4, k=4
   $\mathcal{PL}$ = {1, 2, 4},$\mathcal{TL}$ = {3, 5, 6}
1. $\widetilde{L_3}$=min{$\widetilde{L_3}$,L_4+L_{43}}
   = min {2.6, 2.4+3.5} = 2.6
   $\widetilde{L_5}$=min {$\widetilde{L_5}$, L_4+L_{45}}
   = min {4.6, 2.4+3} = 4.6
   $\widetilde{L_6}$=min {$\widetilde{L_6}$, L_4+L_{46}}

= min {7.5, 2.4+5.5} = 7.5

2. L_3=min{$\widetilde{L_3}$,$\widetilde{L_5}$,$\widetilde{L_6}$}=2.6,
K=3, $\mathcal{PL}$ = {1, 2, 3, 4},$\mathcal{TL}$ = {5, 6}

2. $\widetilde{L_5}$=min{$\widetilde{L_5}$,L_3+L_{35}}
= min {4.6, 2.6+4.4} = 4.6

$\widetilde{L_6}$=min {$\widetilde{L_6}$, L_3+L_{36}}
= min {7.5,2.6+7.4}=7.5

2. L_5=min{$\widetilde{L_5}$,$\widetilde{L_6}$ }=4.6, k=5
$\mathcal{PL}$ = {1, 2, 3, 4, 5}, $\mathcal{TL}$ = {6}

3. $\widetilde{L_6}$=min{$\widetilde{L_6}$,L_5+L_{56}}
= min {7.5, 4.6+4} = 7.5

2. L_6=7.5, k =6
$\mathcal{PL}$ = {1, 2, 3, 4, 5, 6} , $\mathcal{TL}$ =∅

The resulting shortest path of lengths are; L_2=2.3, L_3=2.6, L_4=2.4, L_5=4.6, L_6=7.5.

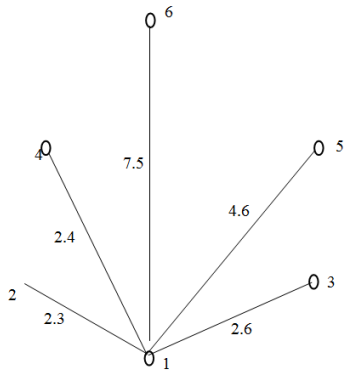


Figure 8. The Shortest Spanning Tree

IV. SHORTEST SPANNING TREES: PRIM'S ALGORITHM

Prim's Algorithm shows in table 6 is another popular Algorithm for the shortest spanning Trees problem. This algorithm avoids ordering edges and gives a tree T at each stage. In Prim's Algorithm, starting from any single vertex , call 1,"grow" the tree T by adding edges to it, one at a time, according to some rule until T finally becomes a Spanning Trees, which is shortest.

Table 5. Prim's Algorithm for Shortest Spanning Trees

ALGORITHM PRIM'S[G=(V,E),

V={1,2,…,n},L_{ij} for all (i,j) in E]

Give a connected graph G=(V,E) with vertices 1,…,n. and edge (i,j) having lengths L_{ij}>0' this algorithm determines a shortest spanning tree T in G and its length L(T).

INPUT: n, edges (i,j) of G and their lengths Lij

OUTPUT: Edges set S of a shortest spanning tree T in G and its length L(T).[Initially, all vertices are unlabeled].

1. Initial step
   Set i(k)=1, U={1}, S=∅.
   Label vertex k (=2,…,n) with λ_k=l_{ik}[=∞ if G has no edge (1,k)].
2. Addition of an edge to the tree T
   Let λ_j be the smallest $\lambda_k$for vertex k not in U. Include vertex j in U and edge (i(j),j) in S.

   If U=V then compute

   L(T)=$\sum l_{ij}$ (sum over all edges in S)

   OUTPUT S. L(T). Stop.

   [S is the edge of a shortest spanning tree T in G.]

   Else continue (that is go to step 3).

3. Label updating
   For every k not in U, if l_{jk}<λ_k, then set λ_k= l_{jk} and i(k)=j.
   Go to Step 2.

End PRIM

Denoted by U the set of vertices of the growing tree T and by S the set of its edges. Thus, initially U= {1} and S= ∅; at the end, U=V, the vertex set of the given graph G=(V,E), whose edges (i,j) have Lij>0. Thus, at the beginning (Step1) the labels

λ_2,…,λ_n of the vertices 2,…,n are the lengths of the edges connecting them to vertex 1(or ∞ if there is no

such edge in G). And pick(Step 2) the shortest of these as the first edge of the growing tree T and include its other end j in U (choosing the smallest j if there are several, to make the process unique). Updating Labels in Step 3(at this stage and at any later stage) concerns each vertex k not yet in U. Vertex k has label $\lambda_k=L_{i(k),k}$ from before. If $L_{ij}<\lambda_k$, this means that k is closer to the new number j just included in U then k is to its old "closest neighbor" i(k) in U. Then, update the label k, replacing $\lambda_k= L_{i(k),k}$by $\lambda_k=l_{ij}$ and setting i(k)=j.

If, however, $l_{jk}\geq \lambda_k$(the old level of k), was not touched. Thus, the label λ_k always identifies the closest neighbor of k in U, and this is updated in Step 3 as U and the tree T grow. From the final labels, can backtrack the final tree, and from their numeric values, compute the total length of this tree.

As an example, we consider the weighted graph shown in figure 9; it is same graph as was used our construction of the shortest spanning tree as shown in figure 10., here ,assume K=1, L=2, Y=3, W=4, Tt=5, Tkh=6.

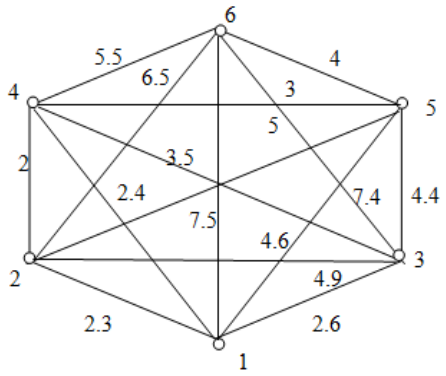


Figure 9.The Weighted Graph

1. i(k)=1,U={1}, S=∅, initial labels see Table.6.

2.$\lambda_2 = L_{12} = 2.3$, is the smallest,
U= {1,2},S={(1,2)}

3 .Update labels as shown in Table 6, column (I)

2.$\lambda_4 = L_{24} = 2$ is the smallest,
U={1,2,4},S={(1,2),(2,4)}

3. Update labels as shown in Table 6, column (II)

2. $\lambda_3 = L_{13} = 2.6$ is the smallest,
U={1,2,3,4},S={(1,2),(2,4),(1,3)}.

3 .Update labels as shown in Table 6, column (III)

2.$\lambda_5 = L_{45} = 3$ is the smallest,
U={1,2,3,4,5},S={(1,2),(2,4),(1,3),(4,5)}

3. Update labels as shown in Table 6, column (IV)

2.$\lambda_6 = L_{56} = 4$ is the smallest,
U=V,S={(1,2),(2,4),(1,3),(4,5), (5,6)}

L (T)=$\sum l_{ij}$ =13.9

Table 6. Labeling of vertices in above example

Vertices	Initial Vertice	Relabeling				
		I	II	III	IV	V
2	L_{12}=2.3	-				
3	L_{13}=2.6	L_{13}=2.6	L_{13}=2.6	-		
4	L_{12}=2.4	L_{24}=2	-			
5	L_{12}=4.6	L_{15}=4.6	L_{45}=3	L_{45}=3	-	
6	L_{12}=7.5	L_{26}=6.5	L_{46}=5.5	L_{56}=5.5	L_{56}=4	-

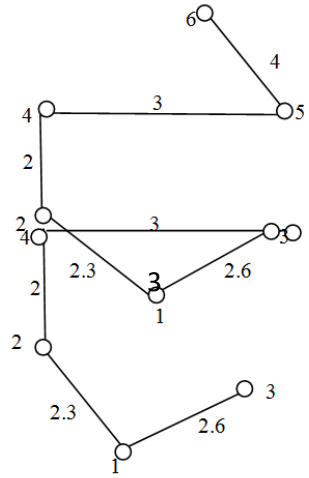


Figure 10. Shortest Spanning Tree

V. CONCLUSION

The problems of finding necessary and sufficient conditions for graphs to be the shortest path and the shortest spanning tree are both interesting and difficult.

There are lots of useful applications for advantage of the people. Among them, two problems that are presented are thought to be every realistic exposures in our real life experience. So, they will serve the human being. Because these problems are applications, it can be applied everywhere for everyone in useful ways. For example, for road building, travelling and paddlers, they are very useful. Thus, it is significant for the places leading to field of development. That is why, these problems are popular. There are different types of shortest path problems in graph theory. Some of which are shortest path between two specific nodes, shortest path from one node (vertex) to all others and shortest path between all nodes (vertices). In Summary, the Floyd, Danzig and double-sweep algorithms can be modified to handle these additional problems.

REFERENCES

[1] ERWIN KREYSZIG, Advance Engineering Mathematics,9th Edition,EM-42008.(Engineering Mathematics VIII). Fourth Year B.E (Second Semester)

[2] W ERWIN KREYSZIG, Advance Engineering Mathematics,10th Edition,EM-42008.(Engineering Mathematics VIII). Fourth Year B.E (Second Semester)

[3] J.A. Bondy and U.S.R. Murty " Graphs Theory With Applications,‘’Department of Combinatorics and Optimization. University of Waterloo, Ontario, Canada. NORTH-HOLLAND, New York. Amsterdam. Oxford.

[4] Erdos, P. and Szekeres, G.(1935).A combinatorial problem in geometry. Compositio Math., 2, 463-70

[5] Erdos, P.(1970). On the graph theory of Turan(Hungarian),Mat. Lapok, 21, 249-51.

[6] U Aung Myo Win, Village tract administrator (2013-2018), Ku Phyu Village , Lay Tin Cone Village Tract, Chauk Township.