

An Efficient Algorithm to Find Next-to-Shortest Path on Trapezoid Graphs

Sambhu Charan Barman

*Department of Applied Mathematics with Oceanology and Computer Programming,
Vidyasagar University, Midnapore – 721 102, India
E-mail: sm5971@rediffmail.com*

Sukumar Mondal

*Department of Mathematics, Y.S. Palpara Mahavidyalaya, Palpara,
Purba Medinipur – 721 458, India*

Madhumangal Pal

*Department of Applied Mathematics with Oceanology and Computer Programming,
Vidyasagar University, Midnapore – 721 102, India
E-mail: madhumangal@lycos.com*

Abstract

Next-to-shortest path between two vertices is a path whose length is strictly greater than the length of the shortest path between the given pair of vertices. The next-to-shortest path problem is an important problem in graph theory and it has many applications in real life problems. This problem is a variation of K -shortest paths problem that finds applications in various fields. In this paper we present an efficient algorithm to find next-to-shortest path between any pair of vertices u, v on trapezoid graphs with n vertices which runs in $O(n^2)$ time.

AMS subject classification:

Keywords: Design of algorithms, analysis of algorithms, shortest paths, next-to-shortest path, trapezoid graphs.

1. Introduction

A trapezoid i is defined by four corner points $[a_i, b_i, c_i, d_i]$ such that a_i and $b_i (> a_i)$ are on the top line and c_i and $d_i (> c_i)$ are on the bottom line of the trapezoid diagram. A trapezoid diagram consist of two horizontal parallel lines, named as top line and bottom line. Each line contains n intervals. Left end point and right end point of an interval i are a_i and b_i on the top line and c_i and d_i on the bottom line. Let $T = \{1, 2, \dots, n\}$, be the n trapezoids where trapezoid i is represented in the trapezoid diagram by four corner points $[a_i, b_i, c_i, d_i]$, a_i, c_i are the left end points and b_i, d_i are the right end points. Let $G = (V, E)$ be an undirected graph with n vertices and m edges and let $V = \{1, 2, \dots, n\}$. G is said to be a *trapezoid graph* if it can be represented by a trapezoid diagram such that each trapezoid corresponds to a vertex in V and $(i, j) \in E$ if and only if trapezoids i and j intersect in the trapezoid diagram [3]. Two trapezoids i and $j (> i)$ intersect if and only if either $(a_j - b_i) < 0$ or $(c_j - d_i) < 0$ or both. We assume that the graph $G = (V, E)$ is connected. Without any loss of generality we assume the following:

- a trapezoid contains four different corner points and that no two trapezoids share a common end point,
- trapezoids in the trapezoid diagram and vertices in the trapezoid graph are one and same thing,
- the trapezoids in the trapezoid diagram T are indexed by increasing right end points on the top channel i.e., $1 < 2 < 3 < \dots < n$ if and only if $b_1 < b_2 < \dots < b_n$.

Figure 2 and Figure 1 represent a trapezoid diagram and corresponding trapezoid graph respectively. The class of trapezoid graph includes two well known classes of intersection graphs: the permutation graphs and the interval graphs [5]. The permutation graphs are obtained in the case where $a_i = b_i$ and $c_i = d_i$ for all i and the interval graphs are obtained in the case where $a_i = c_i$ and $b_i = d_i$ for all i . Trapezoid graphs can be recognized in $O(n^2)$ time [10]. The trapezoid graphs were first studied in [2, 3]. These graphs are superclass of interval graphs ,permutation graphs and subclass of cocomparability graphs [9].

1.1. Definitions

Let $G = (V, E)$ be a graph with vertex set V and edge set E , where n be the number of vertices in V and m be the number of edges in E . The distance between two vertices

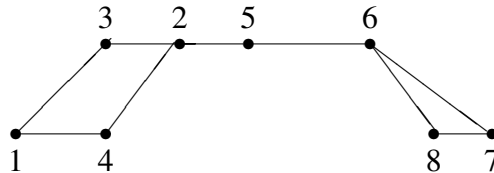


Figure 1: A trapezoid graph G .

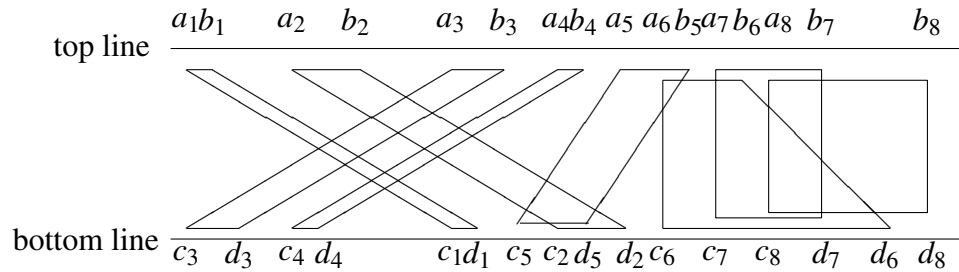


Figure 2: A trapezoid diagram T of a trapezoid graph 1.

u and v in G is denoted by $d(u, v)$ and it is the minimum number of edges required to traversed from u to v .

Next-to-shortest path between two vertices is a path whose length is strictly greater than the length of the shortest path between the given pair of vertices. Next-to-shortest distance between two vertices u and v is the length of the next-to-shortest path between u and v and it is denoted by $d_n(u, v)$.

1.2. Applications

The next-to-shortest path problem is an important problem in graph theory and it has many applications in real life problems. This problem is a variation of K -shortest paths problem that finds applications in operations research, telecommunications, VLSI design and in optimizing compilers for embedded systems [7].

1.3. Survey of the Related Works

Problem of finding next-to-shortest path between two vertices have received much less attention due to the fact that in directed graphs, when we allow edges of length zero, the problem has been shown to be NP-hard [8]. Finding next-to-shortest paths in undirected graphs with strictly positive edge length in time $O(n^3m)$ is studied [6]. Also Lalgudi et al. [8] have designed an algorithm for computing strictly second shortest paths in directed graphs in $O(n^2m)$ time. Recently mandal et al. [11] have designed an optimal algorithm to solve next-to-shortest path problem on circular-arc graphs, in $O(n)$ time where n be the number of vertices.

1.4. Main Result

Here we consider the problem of finding next-to-shortest path in undirected, simple and connected trapezoid graphs. We have designed an efficient algorithm to find next-to-shortest path between any two vertices on trapezoid graphs, in $O(n^2)$ time where n be the number of vertices of the trapezoid graph.

1.5. Organization of the Paper

In the next section, we have discussed about BFS tree of trapezoid graphs. In Section 3, we present the algorithm of marking all alternative shortest paths between two vertices u

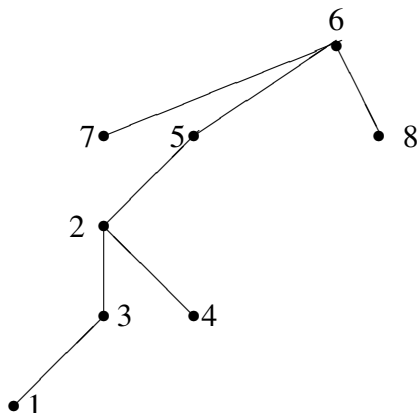


Figure 3: A BFS tree $T^*(6)$ of the graph G of Figure 1.

and v of trapezoid graphs. Some notations are also presented in this section. In Section 4, we prove the important results relating to next-to-shortest path on trapezoid graphs. The algorithm is presented to find next-to-shortest path between two vertices in Section 5. The time complexity is also calculated in this section.

2. BFS Tree on Trapezoid Graph

It is well known that BFS is an important graph traversal technique and also BFS constructs a BFS tree. In BFS, started with a vertex v , we first scan all edges incident on v and then move to an adjacent vertex w . At w we then scan all edges incident on w and move to an adjacent vertex of w . This process is continued till all edges in the graph are scanned [4].

BFS tree can be constructed on general graphs in $O(n + m)$ time where n and m represent respectively the number of vertices and number of edges [15]. Recently, Mondal et al. [12] have designed an **Algorithm TBFS** to construct a BFS tree $T^*(i)$ with root as $i \in V$ on trapezoid graph $G = (V, E)$ in $O(n)$ time where n is the number of vertices. The BFS tree $T^*(6)$ with root as 6 of trapezoid graph of Figure 1 is shown in Figure 3.

We define the level of a vertex v as a distance of v from the root i of the tree $T^*(i)$ and denoted by $level(v)$, $v \in V$ and take the level of root i as 0. The level of each vertex on BFS tree $T^*(i)$, $i \in V$ can be assigned by the BFS algorithm of Chen and Das [1].

2.1. Computation of Main Path on BFS Tree $T^*(i)$

In BFS tree $T^*(i)$, i is the the root of $T^*(i)$. Let y be another specified vertex whose distance from i is k . Now, $y \rightarrow z_1 \rightarrow z_2 \rightarrow \dots \rightarrow z_{k-1} \rightarrow i$, with z_1 as parent of y , z_{j+1} as parent of z_j for all $j = 1, 2, 3, \dots, k - 2$ and i as parent of z_{k-1} , is the shortest path between i and y on BFS tree $T^*(i)$ and this path is called *main path* between i and y .

Also we assume that the vertex whose distance from i is j along the main path, is denoted by u_j .

3. Marking of all Alternative Shortest Paths between two Vertices on Trapezoid Graph

We mark all alternative shortest paths between two vertices u and v , i.e., we construct a subgraph $M^*(u, v)$ of the graph G , which contains only all alternative shortest paths between u and v by the following algorithm.

Algorithm MAST

Input: The corner points $[a_i, b_i, c_i, d_i]$ of the trapezoid i for all $i = 1, 2, \dots, n$.

Output: All marked alternative shortest paths between u and v , which is a sub graph of $G = (V, E)$ and denoted by $M^*(u, v)$.

Step 1: Compute open neighbourhood, $N(u')$, for all $u' \in V$.

Step 2: Construct a BFS tree $T^*(u)$ of the graph G with root as u .

Step 3: Mark the main path between u and v on BFS tree as $u_0 \rightarrow u_1 \rightarrow u_2 \rightarrow \dots \rightarrow u_k$ where $u = u_0$, $u_i = \text{parent}(u_{i+1})$ for $i = 0, 1, 2, \dots, k-1$ and $v = u_k$, i.e., $k = d(u, v)$ and k be a positive integer.

Step 4: Mark all unmarked vertices at level $k-1$ which are adjacent to v on BFS tree and connect them (if they are not connected on tree) with v by edges and mark the edges.

Step 5: Mark all unmarked vertices at level $k-2$ which are adjacent to the marked vertices of level $k-1$ and add the edges(if they are not connected on tree) between the marked vertices of level $k-1$ and the marked vertices of level $k-2$ and also mark the edges(if they are not marked) between the marked vertices of level $k-1$ and the marked vertices of level $k-2$.

Step 6: This process will be continued until we mark all edges between u and the marked vertices of level 1.

Step 7: Delete all unmarked vertices from BFS tree.

end MAST.

Algorithm MAST gives the sub graph $M^*(u, v)$ of G . A subgraph $M^*(6, 1)$ of the graph of Figure 1 is shown in Figure 4. Now we calculate the time complexity of the **Algorithm MAST**. For this purpose, we define the set X_i as follows: X_i : the set of marked vertices at level i on $M^*(u, v)$ with $X_0 = \{u\}$ and $|X_i| = l_{k-i}$ where $i (\geq 0)$ is integer and $d(u, v) = k$ (positive integer).

Theorem 3.1. The time complexity of marking all alternative shortest paths between two vertices on BFS tree is $O(n^2)$.

Proof. Step 1 can be computed in $O(n^2)$ time. In Step 2, BFS tree can be constructed in $O(n)$ time. In Step 3, time complexity of marking the main path between u and v is $O(n)$ time [12]. Step 4 can be completed in $O(l_1)$ time. Time complexity of Step 5

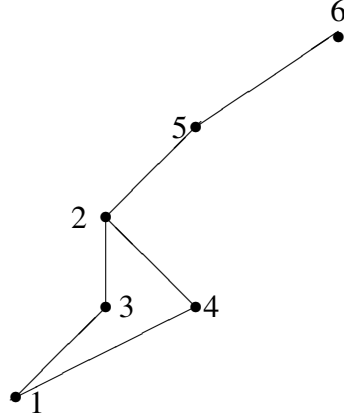


Figure 4: Sub graph $M^*(6, 1)$ of the graph G of Figure 1.

is $O(l_1 l_2)$ time. Time complexity of Step 6 is $O(l_2 l_3 + l_3 l_4 + \dots + l_{k-2} l_{k-1} + l_{k-1})$ time. Also, Step 7 can be completed in $O(n)$ time. Hence the total time complexity of **Algorithm MAST** is

$$\begin{aligned}
& O(n^2) + O(n) + O(n) + O(l_1) + O(l_1 l_2) + O(l_2 l_3 \\
& \quad + l_3 l_4 + \dots + l_{k-2} l_{k-1} + l_{k-1}) + O(n) \\
& = O(n^2) + O(l_1 l_2 + l_2 l_3 + l_3 l_4 + \dots + l_{k-2} l_{k-1}) \\
& = O(n^2) + O((1/2)(l_1 + l_2 + \dots + l_{k-1})^2 - (1/2)(l_1^2 + l_2^2 + \dots + l_{k-1}^2) \\
& \quad - (l_1 l_3 + l_1 l_4 + \dots + l_1 l_{k-1} + l_2 l_4 + l_2 l_5 \dots + l_2 l_{k-1} + l_3 l_5 \\
& \quad + l_3 l_6 \dots + l_3 l_{k-1} + \dots + l_{k-3} l_{k-1})) \\
& \leq O(n^2) + O((1/2)(l_1 + l_2 + \dots + l_{k-1})^2) \\
& \leq O(n^2) + O((1/2)n^2) [\text{as } l_1 + l_2 + \dots + l_{k-1} < n] \\
& \leq O(n^2)
\end{aligned}$$

Therefore, the over all time complexity of the **Algorithm MAST** is $O(n^2)$. ■

3.1. Some Notations

In this subsection, we present some notations which are following.

- k : the length of shortest path between any two vertices u, v .
- $d_n(u, v)$: next-to-shortest distance between two vertices u and v ,
- u_i : u_i is the vertex on the main path at level i of BFS tree $T^*(u)$ with $u_0 = u$.
- X_i : the set of marked vertices at level i on $M^*(u, v)$ with $X_0 = \{u\}$ and $|X_i| = l_{k-i}$, where $i (\geq 0)$ is integer.
- X : $X = \cup_{i=0}^{\text{int}(k/2)} X_{2i}$

- Y : $Y = \begin{cases} \cup_{i=0}^{int(k/2)} X_{2i+1}, & k \text{ is odd,} \\ \cup_{i=0}^{(k/2)-1} X_{2i+1}, & k \text{ is even.} \end{cases}$
 M_i : Open neighbourhood set of u_i excluding adjacent vertices on the main path i.e., $M_i = N(u_i) - \{u_{i+1}\} - \{u_{i-1}\}$.
 $level(v)$: the distance of vertex v from the root u of $T^*(u)$, i.e., $d(u, v) = level(v)$.

Before going to our proposed algorithm we prove the following important results relating to next-to-shortest path on trapezoid graphs.

4. Important Results Relating to Next-to-Shortest Path on Trapezoid Graphs

In this section we present some important results relating to next-to-shortest path on trapezoid graphs as follows.

Lemma 4.1. There exist no cycle of length greater than four without a chord in trapezoid graphs.

Proof. To prove this lemma we consider a trapezoid graph $G = (V, E)$ whose trapezoid diagram is shown in the Figure 6.

From the Figure 5, it is clear that G contains a cycle of length four without any chord. Now, we construct a graph $G' = (V', E')$ shown in the Figure 7 by inserting a vertex 5 between any two vertices ,say, between the vertices 2 and 3 of the graph G . Therefore $(2, 3) \notin E'$, $(3, 5) \in E'$ and $(2, 5) \in E'$. Now, $(2, 3) \notin E'$ implies $b_2 < a_3$ in top line and $d_2 < c_3$ in bottom line. $(3,5) \in E'$ implies $a_5 < b_3$ in top line or $c_5 < d_3$ in bottom line or both. $(2,5) \in E'$ implies $a_5 < b_2$ in top line or $c_5 < d_2$ in bottom line or both. Therefore we have,

- (i) $a_5 < b_2 < a_3 < b_3$ in top line, or
- (ii) $c_5 < d_2 < c_3 < d_3$ in bottom line.

Again from graph G we have, (iii) $a_1 < b_1 < a_2 < b_2 < a_3 < b_3 < a_4 < b_4$ in top line and (iv) $c_3 < d_3 < c_4 < d_4 < c_1 < d_1 < c_2 < d_2$ in bottom line.

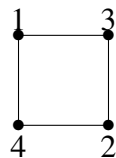


Figure 5: A trapezoid Graph with a cycle of length four.

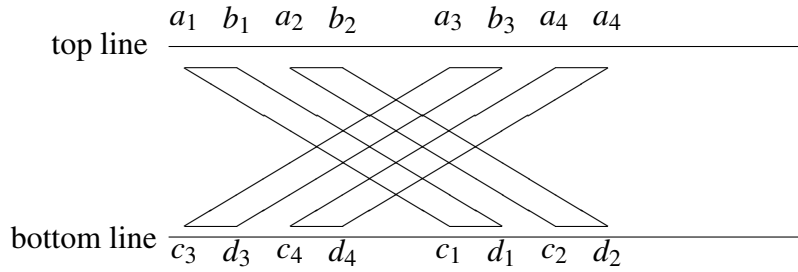


Figure 6: A trapezoid diagram of a trapezoid graph of Figure 5

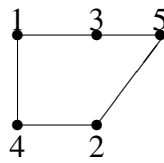


Figure 7: A graph G' which is not a trapezoid graph.

From (iii) we have, (v) $b_2 < a_3 < b_3 < a_4 < b_4$ and From (iv) we have, (vi) $c_3 < d_3 < c_4 < d_4 < c_1 < d_1$.

Now, combining (i) and (v) we have $a_5 < b_2 < a_3 < b_3 < a_4 < b_4$ i.e., $a_5 < b_4$. This implies $(4, 5) \in E'$. Again combining (ii) and (vi) we have, $c_5 < d_2 < c_3 < d_3 < c_4 < d_4 < c_1 < d_1$ i.e., $c_5 < d_1$ and $c_5 < d_4$. This implies $(1, 5) \in E'$ and $(4, 5) \in E'$. If (i) arise in top line and (ii) arise in bottom line, then also we prove that G' has at least one edge among $(4, 5)$ and $(1, 5)$. Hence there exists no cycle of length five without any chord in trapezoid graph. Similarly if we consider a cycle of length six or more then by similar way we can easily prove that there must exists at least one chord. Hence we can say that there exists no cycle of length greater than four without any chord in trapezoid graphs. ■

Now we shall discuss the existence conditions of the next-to-shortest path in trapezoid graphs.

Lemma 4.2. If $N(u') \cap N(v') \neq \phi$ for at least one $u' \in X$ and at least one $v' \in Y$ then $d_n(u, v) = d(u, v) + 1$ where u and v are two vertices of trapezoid graph.

Proof. Let $N(u') \cap N(v') \neq \phi$ for at least one $u' \in X$ and at least one $v' \in Y$. This implies that there exists at least one vertex $y \in N(u') \cap N(v')$ such that $u' \rightarrow y \rightarrow v'$ be the next-to-shortest path between u' and v' . So if we insert y between u' and v' in any shortest path between u and v through u' and v' then $d_n(u, v) = d(u, v) + 1$ and corresponding next-to-shortest path between u and v will be $u \hookrightarrow u' \rightarrow y \rightarrow v' \hookrightarrow v$ where $u \hookrightarrow u'$ and $v' \hookrightarrow v$ be shortest path between u and u' and v' and v respectively on $M^*(u, v)$. ■

Lemma 4.3. If $N(u') \cap N(v') = \phi$ for all $u' \in X$ and for all $v' \in Y$ and $(x, y) \in E$ where $x \in M_i$ and $y \in M_{i+1}$ for any value of $i, i = 0, 1, 2, \dots, k - 1$ then $d_n(u, v) = d(u, v) + 2$, where u and v are two vertices in trapezoid graph.

Proof. Let $N(u') \cap N(v') = \phi$ for all $u' \in X$ and for all $v' \in Y$. Also assume that $u_0 \rightarrow u_1 \rightarrow u_2 \rightarrow \dots \rightarrow u_k$ be the main path between two vertices u and v of length k , i.e., $d(u, v) = k$. Now, if $(x, y) \in E$ where $x \in M_i$ and $y \in M_{i+1}$ for at least one value of $i (i = 0, 1, 2, \dots, k - 1)$ say $i = l$, then $u_l \rightarrow x \rightarrow y \rightarrow u_{l+1}$ be the next-to-shortest path between u_l and u_{l+1} . So if we insert $x \rightarrow y$ between u_i and u_{i+1} for any value of $i (i = 0, 1, 2, \dots, k - 1)$ in main path between u and v , then $d_n(u, v) = d(u, v) + 2$ and corresponding next-to-shortest path will be $u_0 \rightarrow u_1 \rightarrow u_2 \rightarrow \dots \rightarrow u_l \rightarrow x \rightarrow y \rightarrow u_{l+1} \rightarrow u_{l+2} \rightarrow \dots \rightarrow u_k$. ■

Lemma 4.4. There is no next-to-shortest path of length greater than the length of shortest path by three or more between any two vertices in trapezoid graphs.

Proof. By Lemma 4.1 we know that there is no cycle of length greater than four without chord in trapezoid graphs. So we can say that there is no possibility of existing any next-to-shortest path of length greater than the length of shortest path between any two vertices by three or more. ■

Lemma 4.5. The next-to-shortest path does not exist between two vertices u and v on trapezoid graph if $N(u') \cap N(v') = \phi$ for all $u' \in X$ and for all $v' \in Y$ and $(x, y) \notin E$ where $x \in N(u_i) - \{u_{i+1}\}$ and $y \in N(u_{i+1}) - \{u_i\}$ for all $i (i = 0, 1, 2, \dots, k - 1)$.

Proof. If $N(u') \cap N(v') = \phi$ for all $u' \in X$ and for all $v' \in Y$ then there is no cycle of length three whose one edge is situated on the shortest path between two vertices u and v . So we can say that there is no next-to-shortest path of length greater than the length of shortest path between u and v by one.

Again if $(x, y) \notin E$ where $x \in N(u_i) - \{u_{i+1}\}$ and $y \in N(u_{i+1}) - \{u_i\}$ for all $i (i = 0, 1, 2, \dots, k - 1)$ then there is no cycle of length four connecting with the main path between u and v . So we can say that there is no next-to-shortest path of length greater than the length of shortest path between u and v by two.

Besides, in Lemma 4.4 we prove that there is no next-to-shortest path of length greater than the length of shortest path by three or more between any two vertices in trapezoid graphs. Therefore if $N(u') \cap N(v') = \phi$ for all $u' \in X$ and for all $v' \in Y$ and $(x, y) \notin E$ where $x \in N(u_i) - \{u_{i+1}\}$ and $y \in N(u_{i+1}) - \{u_i\}$ for all $i (i = 0, 1, 2, \dots, k - 1)$ then next-to-shortest path does not exist between u and v . ■

5. The Algorithm

To find the next-to-shortest path between two specified vertices u and v we construct a BFS tree with root as u and mark all alternative shortest path between u and v . Then

compute the sets X_i ($i = 0, 1, 2, \dots, k$), X and Y . The main algorithm to find the next-to-shortest path is following.

Algorithm TNSP

Input: A trapezoid graph G with the corner points $[a_i, b_i, c_i, d_i]$ of the trapezoid i for all $i = 1, 2, \dots, n$.

Output: Next-to-shortest path between u and v .

Step 1. Construct a BFS tree $T^*(u)$ with root as u

and let $u_0 \rightarrow u_1 \rightarrow u_2 \rightarrow \dots \rightarrow u_k$ be the main path between u and v on BFS tree with $u_0 = u$ and $u_k = v$.

Step 2. Mark all alternative shortest path between u and v , i.e., construct $M^*(u, v)$.

Step 3. Compute sets $X_i, i = 0, 1, 2, \dots, k$ and

$$X = \bigcup_{i=0}^{\text{int}(k/2)} X_{2i} \text{ and}$$

$$Y = \begin{cases} \bigcup_{i=0}^{\text{int}(k/2)} X_{2i+1}, & k \text{ is odd,} \\ \bigcup_{i=0}^{(k/2)-1} X_{2i+1}, & k \text{ is even} \end{cases}$$

Step 4. Compute the open neighbourhood $N(u')$ for all $u' \in X \cup Y$

Step 5. If $N(u') \cap N(v') \neq \phi$ for at least one $u' \in X$ and at least one $v' \in Y$ then $d_n(u, v) = d(u, v) + 1$.

Compute next-to-shortest path using Lemma 4.2.

Stop.

Step 6. If $N(u') \cap N(v') = \phi$ for all $u' \in X$ and $v' \in Y$ then

compute set M_i for $i = 0, 1, \dots, k - 1$.

Step 6.1. If $(x, y) \in E$ for $x \in M_i$ and $y \in M_{i+1}$ for any i ($i = 0, 1, \dots, k - 1$) then

$$d_n(u, v) = d(u, v) + 2.$$

Compute next-to-shortest path using Lemma 4.3.

Stop.

Else $d_n(u, v) = d(u, v)$

end TNSP.

Theorem 5.1. The time complexity to find the next-to-shortest path between any two vertices u and v in trapezoid graph is $O(n^2)$, where n is the number of vertices of trapezoid graph G .

Proof. Step 1, a BFS tree can be constructed in $O(n)$ time. The time complexity of marking all alternative shortest paths between two vertices u and v in Step 2, is $O(n^2)$ time. Step 3 can be computed in $O(n)$ time. Also, Step 4 can be computed in $O(n^2)$ time. In Step 5, the time complexity of checking $N(u') \cap N(v') \neq \phi$ for at least one $u' \in X$ and at least one $v' \in Y$ is $O(n^2)$ time. In Step 6, sets $M_i, i = 0, 1, 2, \dots, k - 1$ can be computed in $O(n)$ time. The time complexity of checking $(x, y) \in E$ for $x \in M_i$ and $y \in M_{i+1}$ for any of i ($i = 0, 1, \dots, k - 1$) is $O(n^2)$ time. Hence, over all time complexity of the **Algorithm TNSP** is $O(n^2)$. ■

References

- [1] C.C.Y. Chen and S.K. Das, Breadth-first traversal of trees and integer sorting in parallel. *Information Processing Letters*, **41**, pp. 39–49, 1992.
- [2] D.G. Corneil and P.A. Kamula, Extension of permutation and interval graphs, In: *Proc. 18th Southeast Conf. on combinatorics, Graph theory and Computing, Congr. Number*, pp. 267–276, 1987.
- [3] I. Dagan, M.C. Golumbic, and R.Y. Pinter, Trapezoid graphs and their coloring, *Discrete Applied Mathematics*, **21**, pp. 35–46, 1988.
- [4] N. Deo, Graph theory with applications to engineering and computer science (*Prentice Hall of India Private Limited, New Delhi*), 1990.
- [5] M.C. Golumbic, *Algorithmic graph theory and perfect graphs*, (Academic Press, New York, 1980).
- [6] I. Krasikov and S.D. Noble, Finding next-to-shortest paths in a graph, *Information Processing Letters*, **92**, pp. 117–119, 2004.
- [7] K.N. Lalgudi, M.C. Papaefthymiou, and M. Potkonjak, Optimizing systems for effective block-processing: The Kdelay problem, *Tech. Rept. YALE/ DCS/ RR-1102, Dept. of Computer Science*, (Yale University, 1996).
- [8] K.N. Lalgudi and M.C. Papaefthymiou, Computing strictly-second shortest paths, *Information Processing Letters*, **63**, pp. 177–181, 1997.
- [9] Y.D. Liang, Domination in trapezoid graphs, *Information Processing Letters*, **52**, pp. 309–315, 1994.
- [10] T. Ma and J.P. Spindral, An $O(n^2)$ algorithm for 2-chain problem on certain classes of perfect graphs In: *Proc. 2nd ACM-SIAM Symp. on Discrete Algorithms*, 1991.
- [11] S. Mandal and M. Pal, An optimal algorithm to solve next-to-shortest path problem on circular-arc graphs, *Journal of Physical Sciences*, **10**, pp. 201–217, 2006.
- [12] S. Mondal, M. Pal, and T.K. Pal, An optimal algorithm for solving all-pairs shortest paths on trapezoid graphs, *Intern. J. Comput. Engg. Sci.*, **3**(2), pp. 103–116, 2002.
- [13] J.H. Muller and J. Spinrad, Incremental modular decomposition, *J. ACM*, **36**, pp. 1–19, 1989.
- [14] A. Pnueli, A. Lempel and S. Even, Transitive orientation of graphs and identification of permutation graphs, *Canad. J. Math.*, **23**, pp. 160–175, 1971.
- [15] R.E. Tarjan, Depth first search and linear graph algorithm, *SIAM J. Comput.*, **2**, pp. 146–160, 1972.