International Journal of Pure and Applied Mathematics
Volume 75 No. 2 2012, 141-148
ISSN: 1311-8080 (printed version)
url: http://www.ijpam.eu

ijpam.eu

# SHORTEST PATH ALGORITHM FOR SOME GRAPHS BEFORE AND AFTER FOLDING

Habiba El-Zohny[1], Hend El-Morsy[2] §
[1,2]Mathematics Department
Faculty of Science
Al-Azahar University
Cairo, EGYPT

**Abstract:** In this paper we will compute shortest path algorithm for some graphs before and after folding and we will show that the algorithm change for once and the algorithm will not be change in bipartite graph whatever we start at parent vertex or child. Finally if the geometric shape change then the algorithm change.

**AMS Subject Classification:** 05C85, 68R10
**Key Words:** graph algorithm, folding

## 1. Introduction and Background

The concept of a graph is relatively recent since it only formally appeared during the 20th century. Today it has become essential in many fields, in particular in applied and fundamental computer science, in optimization, and in algorithmic complexity. The study of graphs and their applications therefore provides an opportunity to deal with very diverse questions with numerous applications. It can be used, for example, to develop task scheduling methods from optimal paths in graphs, or properties of communication networks in relation to the connectedness of graphs.Historically, graphs were considered long before the

Received: April 27, 2011



§Correspondence author

theory was developed, with famous problems such as the Konigsberg bridge problem[3].Graph algorithms are one of the oldest classes of algorithms and they have been studied for almost 300 years (in 1736), see [8].

## 2. Definitions

**Definition 1.** (Shortest-Path Algorithm) An algorithm that is designed essentially to find a path of minimum length between two specified vertices of connected weighted graph.

**Definition 2.** (Bipartite Graph) A graph $G$ is bipartite if the set of its vertices can be divided into two disjoint subsets such that each edge has an end vertex in each subset. We denote a bipartite graph by $G = (X, Y, E)$, where $X$ and $Y$ are the two subsets of vertices (and so $X$ $(Y)$ is the set of all vertices) and $E$ is the set of edges, see [3].

**Definition 3.** (Complete Bipartite Graph) $G = (V_1 + V_2, E)$ is a bipartite graph such that for any two vertices, $v_1 \in V_1$ and $v_2 \in V_2$, $v_1, v_2$ is an edge in $G$. The complete bipartite graph with partitions of size $|V_1| = m$ and $|V_2| = n$, is denoted $K_{m,n}$, see [1].

**Definition 4.** (Cycle Graph) A cycle graph $C_n$, sometimes simply known as an $n$-cycle is a graph on n nodes containing a single cycle through all nodes. Alternatively, a cycle can be defined as a closed path, see [6].

**Definition 5.** (Graph Map) Let $G_1$ and $G_2$ be graphs and $f : G_1 \to G_2$, be a continuous function. Then $f$ is called a graph map, if:

(i) for each vertex $v \in V(G_1)$, $f(v)$ is a vertex in $V(G_2)$.

(ii) for each edge $e \in E(G_1)$, $\dim(f(e)) \leq \dim(e)$, see [2].

**Definition 6.** (Graph Folding) A graph map $f : G_1 \to G_2$, is called a graph folding if and only if f maps vertices to vertices and edges to edges, i.e., (i) for each vertex v ? V (G1), f(v) is a vertex in V (G2), (ii) for each e ? ?E(G1), f(e) is an edge in E(G2) [ 2 ].

## 3. Main Results. Shortest Path Algorithm. Depth-First-Search (DFS) Algorithm (Breadth-First-Search (BFS) algorithm)

*Step 1:* Insert vertex $v_1$ at the rear of the (initially empty) queue $Q$ and initialize $T$ as the tree made up of this one vertex $v_1$ (the root of the final version of $T$). Visit $v_1$.

*Step 2:* While the queue $Q$ is not empty, delete the vertex $v$ from the front of $Q$:

Now, examine the vertices $v_i$ (for $2 \leq i \leq n$) that are adjacent to $v$ – in the specified order. If $v_i$ has not been visited, perform the following:

(1) Insert $v_i$ at the rear of $Q$;
(2) Attach the edge $\{v, v_i\}$ to $T$; and

(3) Visit vertex $v_i$. If we examine all of the vertices previously in the queue $Q$ and obtain no new edges, then the tree $T$ (generated to this point) is the (rooted ordered) spanning tree for the given order, see [5].

**Lemma.** *Shortest path algorithm for bipartite graph changed before and after folding.*

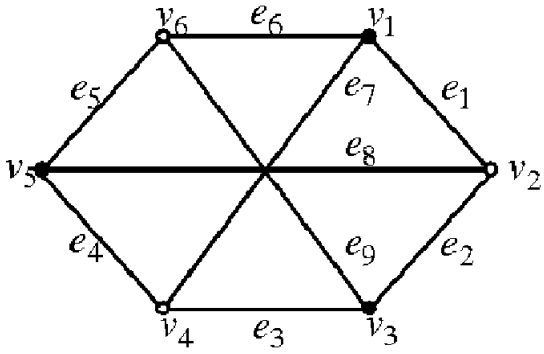


Figure 1

**Example 1.** Let $G$ be complete bipartite graph (see Figure 1), with vertex set $A = \{v_1, v_3, v_5\}$, $B = \{v_2, v_4, v_6\}$, respectively.

Then the shortest path algorithm (see [7]) for this graph before folding, if we start at vertex $v_1$ is:

| | Known | Path | Length |
|---|---|---|---|
| $v_1$ | – | $V_1$ | 0 |
| $v_2$ | – | $V_1$ | 1 |
| $v_3$ | – | $V_2$ | 2 |
| $v_4$ | – | $V_1$ | 1 |
| $v_5$ | – | $V_6$ | 2 |
| $v_6$ | – | $V_1$ | 1 |

We can define graph map $f : G \to G$ by:

$$f\{v_2, v_4\} = \{v_6, v_6\}, \quad f\{e_1, e_2, e_3, e_4, e_7, e_8\} = \{e_6, e_9, e_9, e_5, e_6, e_5\}.$$

This graph map is graph folding such that $f(G) = G'$ is complete bipartite graph shown in Figure 2.

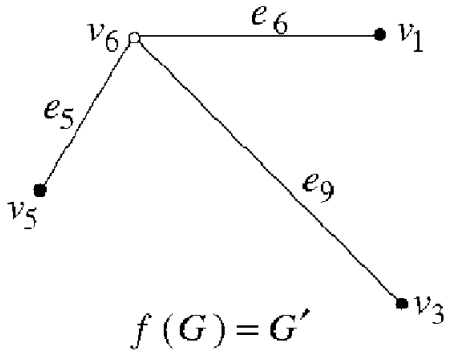


Figure 2

If we compute shortest path algorithm for this graph we have:

| | Known | Path | Length |
|---|---|---|---|
| $v_1$ | – | $V_1$ | 0 |
| $v_3$ | – | $V_6$ | 2 |
| $v_5$ | – | $V_6$ | 2 |
| $v_6$ | – | $V_1$ | 1 |

This differ from the algorithm of the graph before folding.
If we define graph map $g : G' \to G'$ by:

$$g\{v_1, v_5\} = \{v_3, v_3\}, \quad g\{e_I\} = e_9, \quad I = 5, 6,$$

see Figure 3.

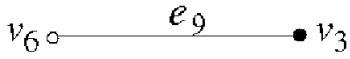


Figure 3

Then the algorithm will be:

| | Known | Path | Length |
|---|---|---|---|
| $v_3$ | – | $V_3$ | 0 |
| $v_6$ | – | $V_3$ | 1 |

If we change the graph map and compute the algorithm for the resulting graph we will obtain the same result.

**Theorem 1.** *Shortest path algorithms computed from (child) vertex to other vertices are the same in all different folding maps (because these folding are homomorphic to each other).*

*Similarly, shortest path algorithm computed from (parent) vertex to other vertices are the same in all different folding maps (because these foldings are homeomorphic to each other).*

**Example 2.** If we compute shortest path algorithm for folding graph shown in Figure 2 starting at vertex $v_1$ (child) we have:

| | Known | Path | Length |
|---|---|---|---|
| $v_1$ | – | $V_1$ | 0 |
| $v_3$ | – | $V_6$ | 2 |
| $v_5$ | – | $V_6$ | 2 |
| $v_6$ | – | $V_1$ | 1 |

But if we starting at v6 (parent) we have:

| | Known | Path | Length |
|---|---|---|---|
| $v_1$ | – | $V_6$ | 1 |
| $v_3$ | – | $V_6$ | 1 |
| $v_5$ | – | $V_6$ | 1 |
| $v_6$ | – | $V_6$ | 0 |

We can define graph map $f : G \to G$ by:

$$F : \{v_2, v_6\} = \{v_4, v_4\}, \quad G : \{e_2, e_8, e_1, e_5, e_6, e_9\} = \{e_3, e_4, e_7, e_4, e_7, e_3\},$$

see Figure 4.

If we compute shortest path algorithm starting at vertex $v_3$ (child) we have:

But if we start at vertex $v_4$ (parent) we have:

If we change graph map, we obtain the same result.

**Lemma.** *There is difference in the shortest path algorithm when we start from parent or chilled.*

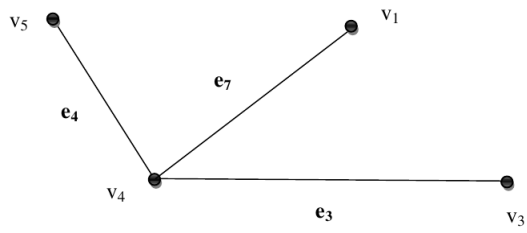


Figure 4

| | Known | Path | Length |
|---|---|---|---|
| $v_1$ | – | $V_1$ | 0 |
| $v_3$ | – | $V_4$ | 2 |
| $v_4$ | – | $V_1$ | 1 |
| $v_5$ | – | $V_4$ | 2 |

| | Known | Path | Length |
|---|---|---|---|
| $v_1$ | – | $V_4$ | 1 |
| $v_3$ | – | $V_4$ | 1 |
| $v_4$ | – | $V_4$ | 0 |
| $v_5$ | – | $V_4$ | 1 |

**Example 3.** Shortest path algorithm for cycle graph before folding:

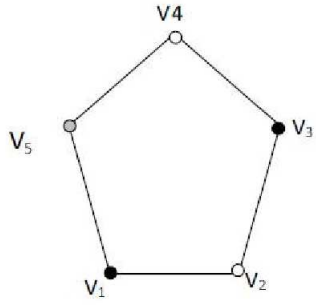


Figure 5

Let $C_n$ is cycle graph with 5 vertices, see Figure 5. We can compute shortest path algorithm for this graph starting at vertex $v_1$ as follows:

If we define a graph folding map $F : Cn \to Cn$ as follows:

$$F(V_1, V_2, V_3, V_4, V_5) = (V_1, V_2, V_1, V_2, V_5),$$

| | Known | Path | Length |
|---|---|---|---|
| $v_1$ | – | $V_1$ | 0 |
| $v_2$ | – | $V_1$ | 1 |
| $v_3$ | – | $V_2$ | 2 |
| $v_4$ | – | $V_5$ | 2 |
| $v_5$ | – | $V_1$ | 1 |

see Figure 6.

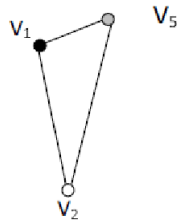


Figure 6

Then shortest path algorithm for folding graph starting at vertex $V_1$ is:

| | Known | Path | Length |
|---|---|---|---|
| $v_1$ | – | $V_1$ | 0 |
| $v_2$ | – | $V_1$ | 1 |
| $v_5$ | – | $V_1$ | 1 |

**Lemma.** *If the geometric shape change then the algorithm change.*

## References

[1] J.A. Bondy, U.S.R. Murty, *Graph Theory with Applications*, North-Holland (1976), ISBN: 0-444-19451-7.

[2] E.M. El-Kholy, A. El-Esawy, Graph folding of some special graphs, *Journal of Mathematics And Statistics*, **I**, No. 1 (2005), 66-70.

[3] Jean-Claude Fournier, *Graph Theory and applications with Exercises and Problems*, ISTE Ltd (2009).

[4] M.R. Zeen M.R. El-Deen, Relation between chromatic number and graph folding, *International Journal of Computational and Applied Mathematics*, **5**, No. 2 (2010), 189-198.

[5] P.Grimadi Ralph, *Discrete and Combinatorial Mathematics: An Applied Introduction*, Fifth Edition, Pearson Education, Inc. (2004).

[6] S. Pemmaraju, S. Skiena, Cycles, stars, and wheels, In: *Computational Discrete Mathematics Combinatiorics and Graph Theory in Mathematica*, Cambridge, England (2003), 248-249.

[7] http://courseweb.sp.cs.cmu.edu/c̃s200/lecture22/lecture22.html

[8] http://www.softpanorama.org/Algorithms/graph_algorithms.shtml #History