Algorithms for Generating Star $S_n$ and Path $P_n$ of Graphs using BFS

Dr. H. B. Walikar$^2$, Ravikumar H. Roog$^1$, Shreedevi V. Shindhe$^3$, Ishwar B$^4$

1$^2$Prof. Dept of Computer Science, Karnatak University, Dharwad,
2$^3$Research Scholars, Dept of Computer Science, Karnatak University, Dharwad

Abstract: In this paper we deal with BFS algorithm by modifying it with some conditions and proper labeling of vertices which results $BFS(G) = Star S_n$ and $BFS(G) = Path P_n$ on applying it to some small basic class of graphs. The BFS algorithm has to modify accordingly. Some graphs will result in $Star S_n$ and $Path P_n$ by direct application of BFS whereas some need modifications in the algorithm. The BFS algorithm starts with a root vertex called start vertex. The resulted output tree structure will be in the form of $Star S_n$ Structure or in $Path P_n$ Structure.

Keywords: BFS, Graph, Star, Path.

1. INTRODUCTION

1.1 Graph:
A graph is a finite collection of objects called vertices together with a set of unordered pairs of distinct vertices of $G$ called edges. The vertex set and the edge set of $G$ are denoted by $V(G)$ and $E(G)$ respectively. A graph $G$ with vertex set $V(G)$ and edge set $E(G)$ is denoted by $G = (V(G), E(G))$. [1]

1.2 Star Graph:
The Star graph $S_n$ of order $n$, sometimes simply known as an "$n$-star", is a tree on $n$-nodes with one node having vertex degree $n - 1$ and the other $1$ having vertex degree $2$. Fig.1 [2]

1.3 Path Graph:
The Path $P_n$ is a tree with two nodes of vertex degree $1$, and the other $n-2$ nodes of vertex degree $2$. A path graph is a graph that can be drawn so that all of its vertices and edges lie on a single straight line. Fig.2 [2]

1.4 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. Fig.3 [2]

1.5 Complete Graph:
A complete graph is a graph in which each pair of graph vertices is connected by an edge. The complete graph with $n$ graph vertices is denoted $K_n$. Fig.4 [2]

1.6 Wheel Graph:
A wheel $W_n$ of order $n$, sometimes simply called an $n$-wheel, is a graph that contains a cycle of order $n$ and for which every graph vertex in the cycle is connected to one other graph vertex. Fig.5 [2]

1.7 Windmill Graph:
The windmill graph $D_n^k$ is the graph obtained by taking copies of the complete graph $K_k$ with a vertex in common. Fig.6 [2]

1.8 Diamond Graph:
The diamond graph is the simple graph on 4 nodes and 5 edges illustrated Fig.7. [2]

1.9 Paw Graph:
The paw graph is the $S$-pan graph, which is also isomorphic to the $(2,1)$-tadpole graph. Fig.8 [2]

1.10 Gem Graph:
The gem graph is the fan graph $F_f$ illustrated Fig.9 [2]

1.11 Dart Graph:
The dart graph is the $S$-vertex graph illustrated Fig.10 [2]

1.12 Tetrahedral Graph:
The tetrahedral graph is the Platonic graph that is the unique polyhedral graph on four nodes which is also the complete graph $K_4$ and therefore also the wheel graph $W_4$. Fig.11 [2]

1.13 Concatenation:
The act of linking together as in a series or chain.

1.14 Concatenation Graph $G_k$:
A Cycle and Path class of graph linked together as in a series $G_k$ as shown Fig.12.

1.15 Breadth First Search Algorithm (BFS):
Breadth first search is another useful tool in many graph algorithms. The BFS visits systematically the vertices of graph or digraph, beginning at some vertex $v$ of $G$ (also called a root vertex). The root vertex is the first active vertex. At any stage during the search, all the vertices adjacent from the current active vertex are scanned for vertices that have not yet been visited that are “broad” search performed for unvisited vertices. Each time a vertex is visited for the first time, it is labeled (according to some rule that depends on the goal to be achieved) and added back to the queue. Note that, in this search a queue is used rather than a stack. The current active vertex is the one at the front of the queue. As soon as its neighbors have been visited, it is deleted from the queue. If the queue is empty and some vertices of the graph or digraph have not yet been visited, we select any unvisited vertex, assign it a label and add it to the queue. When all the vertices of the graph have been visited, the search is complete. [3]

Applications:
1. Finds a tree.
2. Tests cross edge connectivity.
3. Finds paths.
4. Finds cyclicity.

1.16 Queue:
A **queue** is a list of elements which supports the following operations:

- **enqueue**: Adds an element to the end of the list.
- **dequeue**: Removes an element from the front of the list.

Elements are extracted in first-in first-out (FIFO) order, i.e., elements are picked in the order in which they were inserted. [4]

**1.17 Algorithm BFS (v):** [5]
// Implements a breathe-first search traversal of a given graph
// Input: Graph $G = (V, E)$
// Output: Graph $\tilde{G}$ with its vertices marked with consecutive integers
// in the order they have been visited by the BFS traversal
mark each vertex in $\tilde{V}$ with $\emptyset$ as a mark of being “unvisited”

```plaintext
count ← $\emptyset$
for each vertex $v$ in $\tilde{V}$ do
    if $v$ is marked with $\emptyset$
        bfs (v)
    count ← count + 1;
    mark $v$ with count
    initialize a queue with $\emptyset$.
while the queue is not empty do
    for each vertex $w$ in $\tilde{V}$ adjacent to the front vertex do
        if $w$ is marked with $\emptyset$
            count ← count + 1;
            mark $w$ with count
            add $w$ to the queue
            remove the front vertex from the queue.
```

Fig. 1: Star $\mathcal{S}_5$

Fig. 2: Path $\mathcal{P}_4$

Fig. 3: Cycle

Fig. 4: Complete

Fig. 5: Wheel $\mathcal{W}_n$

Fig. 6: Windmill

Fig. 7: Diamond

Fig. 8: Paw

Fig. 9: Gem

Fig. 10: Dart

Fig. 11: Tetrahedral

Fig. 12: Concatenation Graph $\mathcal{C}_n$
2. PROPOSED WORK:
Our aim is to find the small class of graphs which can result in BFS tree which is Star graph called BFS Star $S_n$ and BFS tree which is path called BFS Path $P_n$ after application of BFS. i.e. $\text{BFS}(G) = \text{Star } S_n$ and $\text{BFS}(G) = \text{Path } P_n$.

Some graphs will result in Star $S_n$ and Path $P_n$ by direct application of BFS where some need modifications in the algorithm. The BFS algorithm starts with a root vertex called start vertex. The resulted output tree will be in the form of Star $S_n$ or in Path $P_n$.

In this paper we are dealing with some small basic graphs and their behavior on applying BFS. Some of the graph classes results in $\text{BFS}(G) = \text{Star } S_n$ and $\text{BFS}(G) = \text{Path } P_n$ after applying some conditions and by specifying start vertex and modifying the BFS algorithm accordingly. We want to generate a Star and Path by applying BFS on the given graph. We first show some class of graphs which results in $\text{BFS}(G) = \text{Star}$ and then will show some class of graphs which results in $\text{BFS}(G) = \text{Path } P_n$ by finding the start vertex by modifying BFS algorithm. To elaborate the concept let us consider the following example.

Paw Graph which results both $\text{BFS}(G) = \text{Star } S_n$ and $\text{BFS}(G) = \text{Path } P_n$.

![Fig. 13](image-url)

$\text{BFS}(G) = \text{Star } S_n$:

For the given graph, in Fig.13 if we apply BFS with vertex $V_2$ as start vertex, then we get $\text{BFS}(G)$ as in Fig. 14.

![Fig. 14](image-url)

Therefore the structure of $\text{BFS}(G)$ is in the form of Star $S_n$.

3. GRAPH CLASSES THAT GIVES BFS STAR $S_n$, BY APPLYING BFS WITHOUT ANY CONDITIONS:

3.1. Star graph:
When $G$ is star, then applying BFS by taking any vertex as a start vertex it gives $\text{BFS Star } S_n$. Here each vertex is a start vertex.

3.2. Complete graph:
In complete graph also each vertex is start vertex. When $G$ is complete, then applying BFS with taking any vertex as start vertex it gives $\text{BFS Star } S_n$.

3.3. Tetrahedral Graph:
When $G$ is Tetrahedral, then applying BFS by taking any vertex as start vertex it gives $\text{BFS Star } S_n$. Here also each vertex is start vertex.

4. GRAPH CLASSES THAT GIVE BFS STAR ($S_N$) BY SPECIFYING A START VERTEX:
Algorithm BFS Star $S_n$ by BFS ($G$)

// I/P: Graph $G = (V,E)$
// O/P: Star Graph $S_n$ in the order they have been visited by BFS traversal.
Step 1: Find the degree of vertices and find the max deg $\Delta(G)$ of a Graph.
Step 2: Let the start vertex ‘$S$’ be the vertex with max deg $\Delta(G)$.
Step 3: Apply BFS on given graph with ‘$S$’ as start vertex.
Step 4: We get the output as a BFS Star $S_n$. 

**Fig. 15** $\text{BFS}(G)$ with start vertex
In this section we will study about how the start vertex ‘S’ of given graph is obtained so that vertices are visited in a specific order from start vertex. For some class of graphs we can obtain \( BFST(G) = BFSSmS_n \) by specifying start vertex and applying BFS algorithm accordingly. In the same way we show certain class of graph which results \( BFST(G) = BFSSmS_n \). Consider below example.

Consider Dart graph as shown in fig.16:

The degree of each vertex is:
- \( V_1 \) = 4
- \( V_2 \) = 1
- \( V_3 \) = 2
- \( V_4 \) = 2
- \( V_5 \) = 2

In this graph we have \( \Delta(G) = 4 \) and \( d(G) = 1 \) and degree 2 also.

Therefore \( V_1 \) is a vertex with max degree \( \Delta(G) = 4 \). Let \( V_1 \) be a start vertex.

Applying \( BFST \) to fig.16, let \( V_1 \) be start vertex:

Therefore Dart graph \( BFST(G) = BFSTmS_n \) in fig.17.

Therefore the class of graphs which results \( BFST(G) = BFSTmS_n \) are Wheel graph, Windmill graph and Gem graph.

5. **Graph Classes that gives BFS Path** \( P_n \),
   **By Applying BFS without Any Conditions**
   5.1 **Cycle**:

![Flow Chart of Algorithm BFS Star S_n by BFS (G)](image-url)
When $C$ is cycle $C_n$ then by applying the BFS simply gives a $P_n$. Of course the cycle results in $P_n$ just by removing the edge also. But here we are studying with respect to BFS. In cycle every vertex is a start vertex.

5.2 Path:
By applying BFS on any path graph $P_n$, we get $P_n$ it as a result. Here every vertex is a start vertex.

6. Graph classes that give BFS Path ($P_n$) by specifying a start vertex:

Algorithm BFS Path $P_n$ by BFS (G)

// I/P: Graph $G$ = ($V$, $E$);
// O/P: Path Graph $P_n$ in the order they have been visited by BFS traversal.

Step 1: Find the diameter of cycle $C_n$ which is $\frac{n}{2}$.

Step 2: Find a vertex '$v$' which is at a distance $\frac{\text{diam } C_n}{2}$ from the concatenated vertex of $C_n$.

Step 3: Select '$v$' as a start vertex.

Step 4: Apply BFS using '$v$' as start vertex.

Step 5: We get output $P_n$.

In this section we will study about how the start vertex '$v$' of given graph is obtained so that vertices are visited in a specific order from start vertex. For concatenated graph $C_n$, class of graphs we can obtain $\text{BFS}(G) = \text{BFSPath}(G)$ by specifying start vertex and applying BFS algorithm accordingly. Here if cycle is even we get one start vertex, if odd we get two start vertex we can take any one of them as a start vertex.

Consider a graph $G$ as shown in Fig.20

According to our modified algorithm applying on above graph, in which first step we find the diameter of $C_6$,

\[ \text{Diam } C_6 = \frac{6}{2} = 3 \]

Therefore $\text{Diam } C_6 = 3$

Here concatenated vertex of $C_6$, $V_6$, of given graph is $V_6$. In second step we find vertex $v$ from vertex $V_6$ at distance $\frac{3}{2}$, i.e. $\text{dist } = \text{diam } C_6$. Therefore here the vertex is $V_3$. In third step here we take $V_3$ as a start vertex any one of them. In step four we apply BFS to start vertex, consider $V_3$ as a start vertex.

Applying $BFS$ to Fig.20, let be start vertex:

**Fig.21**

Therefore $\text{BFS}(G) = \text{BFSPath}(G)$.

7. Conclusion:
By using a BFS, here we are finding a Star $S_n$ consisting all the vertices of the given graph and the Path $P_n$ consisting all the vertices of the given graph. Only few classes of graphs have been tested here. We can consider
some more class of graphs for further analysis like House
graph, Bull Graph.

REFERENCES:


AUTHOR

Ravikumar H. Roogi received the B.Sc (Electronics). and M.Sc(Computer Science). degrees from Karnataka University Dharwad in 2006 and 2008, respectively. Currently pursuing PhD (Computer Science) under guidance of Dr. H. B. Walikar from Karnataka University Dharwad.