Introduction to Graph with Breadth First Search(BFS) and Depth First Search(DFS) Traversal Implemented in JAVA

Licence CPOL
First Posted 3 Jan 2009
Views 92,337
Bookmarked 19 times

By bijulsoni | 3 Jan 2009

Java Windows Java SE Dev Beginner

This article provides a brief introduction about graph data structure with BFS and DFS traversal algorithm.

- Download source code - 5.53 KB

Introduction

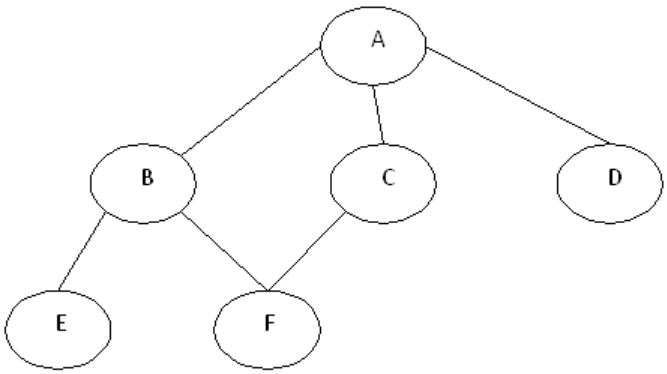
The objective of this article is to provide a basic introduction about graphs and the commonly used algorithms used for traversing the graph, BFS and DFS. Breadth First Search (BFS) and Depth First Search (DFS) are the two popular algorithms asked in most of the programming interviews. I was not able to find a simple, precise explanation for beginners on this topic. So, I decided to write an article for graph. This article will help any beginner to get some basic understanding about what graphs are, how they are represented, graph traversals using BFS and DFS.

What is a Graph?

Graphs are one of the most interesting data structures in computer science. Graphs and the trees are somewhat similar by their structure. In fact, tree is derived from the graph data structure. However there are two important differences between trees and graphs.

1. Unlike trees, in graphs, a node can have many parents.
2. The link between the nodes may have values or weights.

Graphs are good in modeling real world problems like representing cities which are connected by roads and finding the paths between cities, modeling air traffic controller system, etc. These kinds of problems are hard to represent using simple tree structures. The following example shows a very simple graph:



In the above graph, A,B,C,D,E,F are called nodes and the connecting lines between these nodes are called edges. The edges can be directed edges which are shown by arrows; they can also be weighted edges in which some numbers are assigned to them. Hence, a graph can be a directed/undirected and weighted/un-weighted graph. In this article, we will discuss undirected and un-weighted graphs.

Graph Representation in Programming Language

Every graph has two components, Nodes and Edges. Let's see how these two components are implemented in a programming language like JAVA.

1. Nodes

Nodes are implemented by class, structures or as Link-List nodes. As an example in JAVA, we will represent node for the above graph as follows:

Collapse

```
//
Class Node
{
   Char data;
   Public Node(char c)
   {
      this.data=c;
   }
}
//
```

2. Edges

Edges represent the connection between nodes. There are two ways to represent edges.

Adjacency Matrix

It is a two dimensional array with Boolean flags. As an example, we can represent the edges for the above graph using the following adjacency matrix.

	A	B	C	D	E	F
A	0	1	1	1	0	0
B	1	0	0	0	1	1
C	1	0	0	0	0	1
D	1	0	0	0	0	0
E	0	1	0	0	0	0
F	0	1	1	0	0	0

In the given graph, A is connected with B, C and D nodes, so adjacency matrix will have 1s in the 'A' row for the 'B', 'C' and 'D' column.

The advantages of representing the edges using adjacency matrix are:

1. Simplicity in implementation as you need a 2-dimensional array
2. Creating edges/removing edges is also easy as you need to update the Booleans

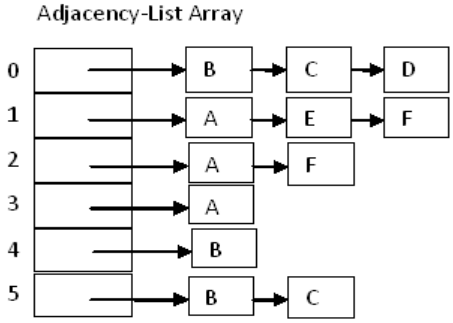
The drawbacks of using the adjacency matrix are:

1. Increased memory as you need to declare N*N matrix where N is the total number of nodes.
2. Redundancy of information, i.e. to represent an edge between A to B and B to A, it requires to set two Boolean flag in an adjacency matrix.

In JAVA, we can represent the adjacency matrix as a 2 dimensional array of integers/Booleans.

Adjacency List

It is an array of linked list nodes. In other words, it is like a list whose elements are a linked list. For the given graph example, the edges will be represented by the below adjacency list:



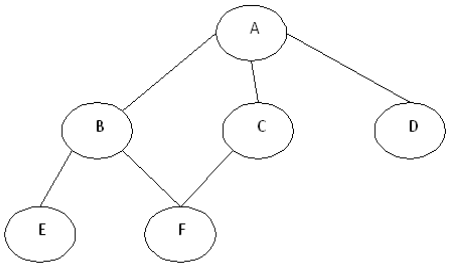
Graph Traversal

The breadth first search (BFS) and the depth first search (DFS) are the two algorithms used for traversing and searching a node in a graph. They can also be used to find out whether a node is

reachable from a given node or not.

Depth First Search (DFS)

The aim of DFS algorithm is to traverse the graph in such a way that it tries to go far from the root node. Stack is used in the implementation of the depth first search. Let's see how depth first search works with respect to the following graph:



As stated before, in DFS, nodes are visited by going through the depth of the tree from the starting node. If we do the depth first traversal of the above graph and print the visited node, it will be "A B E F C D". DFS visits the root node and then its children nodes until it reaches the end node, i.e. E and F nodes, then moves up to the parent nodes.

Algorithmic Steps

1. **Step 1**: Push the root node in the Stack.
2. **Step 2**: Loop until stack is empty.
3. **Step 3**: Peek the node of the stack.
4. **Step 4**: If the node has unvisited child nodes, get the unvisited child node, mark it as traversed and push it on stack.
5. **Step 5**: If the node does not have any unvisited child nodes, pop the node from the stack.

   Based upon the above steps, the following Java code shows the implementation of the DFS algorithm:

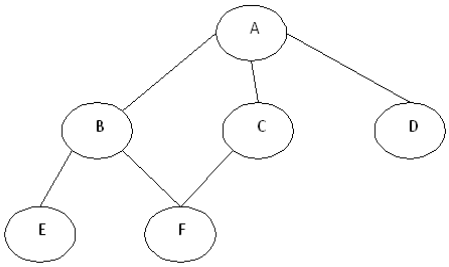
Collapse

```
//
public void dfs()
{
        //DFS uses Stack data structure
        Stack s=new Stack();
        s.push(this.rootNode);
        rootNode.visited=true;
        printNode(rootNode);
        while(!s.isEmpty())
        {
                Node n=(Node)s.peek();
                Node child=getUnvisitedChildNode(n);
                if(child!=null)
                {
                        child.visited=true;
                        printNode(child);
                        s.push(child);
                }
                else
                {
                        s.pop();
                }
        }
        //Clear visited property of nodes
        clearNodes();
}

//
```

Breadth First Search (BFS)

This is a very different approach for traversing the graph nodes. The aim of BFS algorithm is to traverse the graph as close as possible to the root node. Queue is used in the implementation of the breadth first search. Let's see how BFS traversal works with respect to the following graph:



If we do the breadth first traversal of the above graph and print the visited node as the output, it will print the following output. "A B C D E F". The BFS visits the nodes level by level, so it will start with level 0 which is the root node, and then it moves to the next levels which are B, C and D, then the last levels which are E and F.

Algorithmic Steps

1. **Step 1**: Push the root node in the Queue.
2. **Step 2**: Loop until the queue is empty.
3. **Step 3**: Remove the node from the Queue.
4. **Step 4**: If the removed node has unvisited child nodes, mark them as visited and insert the unvisited children in the queue.

   Based upon the above steps, the following Java code shows the implementation of the BFS algorithm:

Collapse

```
//
public void bfs()
{
        //BFS uses Queue data structure
        Queue q=new LinkedList();
        q.add(this.rootNode);
        printNode(this.rootNode);
        rootNode.visited=true;
        while(!q.isEmpty())
        {
                Node n=(Node)q.remove();
                Node child=null;
                while((child=getUnvisitedChildNode(n))!=null)
                {
                        child.visited=true;
                        printNode(child);
                        q.add(child);
                }
        }
        //Clear visited property of nodes
        clearNodes();
}
//
```

The full implementation of this is given in the attached source code.

About the Code

The source code for this article is a JAVA project that you can import in eclipse IDE or run from the command prompt. You need to run the *Main.java* file to see the traversal output.

Main.java is a Java Console application which creates a simple undirected graph and then invokes the DFS and BFS traversal of the graph.

History

- 29th December, 2008: Initial version

License

About the Author

bijulsoni

Software Developer
Microsoft
United States

Member

Article Top

Comments and Discussions

Msgs 1 to 6 of 6 (Total in Forum: 6) (Refresh) First Prev Next

Thankz!	ghon_pritz	15:51 10 Mar '10
THANKS FOR THE CODE	RafayNaeem	7:09 27 Feb '10
Thanks	doancia	9:11 30 Aug '09
important request	midotetos	4:17 22 May '09
Nice	Vishu Gurav	17:08 12 May '09
Thank You!	clashingrocks	20:26 25 Feb '09

Last Visit: 19:00 31 Dec '99 Last Update: 12:12 9 Mar '11 1

General News Question Answer Joke Rant Admin

Use Ctrl+Left/Right to switch messages, Ctrl+Up/Down to switch threads, Ctrl+PgUp/PgDown to switch pages.