

## Fundamental Algorithms 10

### K-Exercise 1: Modified Depth/Breadth-First Traversal

Consider the following, modified traversal algorithm for graphs and trees:

```
ModTraversal(V:Node) {  
    // assume Mark[V.key]=1 at entry  
    // init (local!) list for "active" nodes  
    Queue active = {};  
    // visit all (non-visited) nodes connected to V  
    forall (V,W) in V.edges do {  
        if Mark[W.key] = 0 then [  
            // visit node W and mark as visited:  
            Visit(W);  
            Mark[W.key] := 1;  
            // append node W to "active" nodes  
            append(active,W);  
        ];  
    };  
    // perform traversal from all "active" nodes connected to V  
    forall W in active do {  
        ModTraversal(W);  
    };  
}
```

- a) Consider the graph given in Figure 1: in what order are the nodes "visited" by the modified traversal? (Number the nodes in the graph accordingly.) The traversal shall be called by

```
Mark[S.key] := 1;  
ModTraversal(S);
```

(S being the start node for the traversal).

- b) In the same graph, mark the edges that are part of the spanning tree computed by Mod-Traversal.

c) Now assume that the second forall-loop is changed into a parallel loop:

```
// perform traversal from all (non-visited) nodes connected to V  
forall W in active do in parallel {  
  ModTraversal(W);  
};
```

Discuss whether there can be concurrent read or write access to the elements of the array Mark. Discriminate between the two cases that the traversed graph is a tree and that it is not a tree.

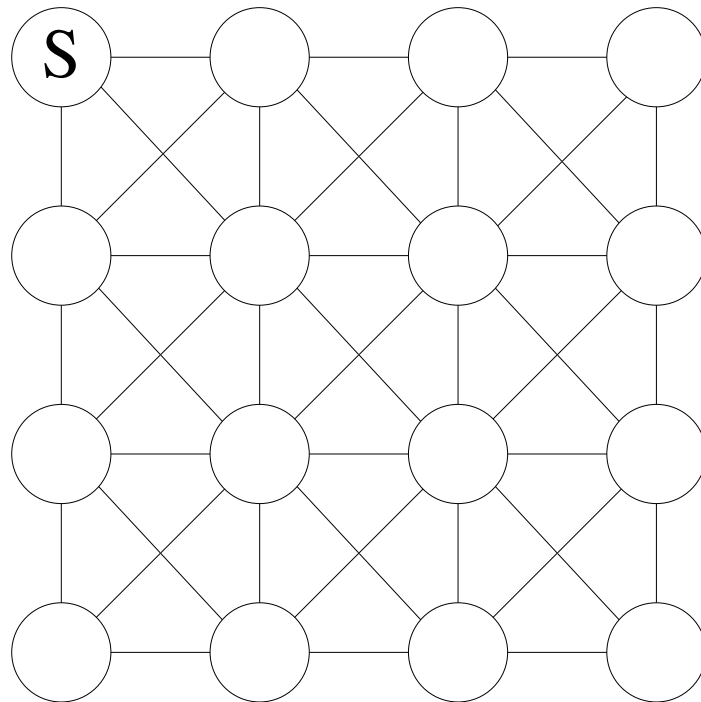


Figure 1: Graph for Exercise 1a) and 1b). It is not specified, in which order edges outgoing from a node  $V$  are stored in the list  $V.edges$  – you may assume any order you like.