Technische Universität München
Institut für Informatik
Theoretical Computer Science

WS 2017/18

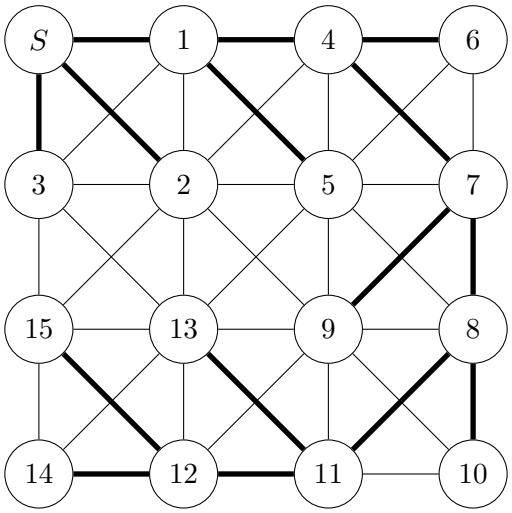
Fundamental Algorithms 10 - Solution Examples

Exercise 1 (Modified Graph Traversal)

Consider the modified traversal algorithm for graphs and trees ModTrav.

Algorithm 1: ModTrav

```
Input: V: Node
       visit: Visit function
act ← []; // Local queue of active nodes
// Iterate over all successors
foreach W ∈ V.succ do
    if mark[W.key] = 0 then
        visit(W); // Visit unmarked node W
        mark[W.key ← 1; // Mark W as visited
        act ← act ∘ W; // Add W to the local active node queue
    end
end
foreach W ∈ act do ModTrav(W);
```



1. Given the graph above, in which order are nodes visited (i.e. in which order `visit` is called on them) by this algorithm? Number the nodes accordingly. The successors of each node (i.e. V.succ) are stored clockwise.

2. In the same graph, mark the edges that are part of the spanning tree computed by the algorithm.

3. Now assume that the second for-loop is changed into a parallel loop. Discuss whether there can be concurrent read or write access to the elements of the array *mark*. Think about what happens if the graph is a tree.

Solution:

1. Due to the recursive call of the function ModTrav, the traversal is similar to a depth-first traversal. However, the approach to first mark the nodes adjacent to the current node and append them to a list of active nodes is similar to breadth-first traversal. Together, the traversal is a mixture between DFT and BFT. First, all nodes adjacent to the current node are visited, but the traversal then proceeds in depth-first manner.

3. Distinguishing the two cases:

   Arbitrary graph: Concurrent access is possible. Consider nodes 2 and 3 in the example graph. The recursion starting from these nodes concurrently accesses the nodes 13 and 15, for example.

   Tree: No concurrent accesses can happen – subtrees are traversed in parallel, but, as subtrees are disjoint, the accesses are exclusive.