# Distributed Systems Part II

## Solution to Exercise Sheet 9

## 1 Clock Synchronization: Spanning Tree

The grid is composed of cells and nodes. The nodes are shown as black dots in Figure 1 and the cells are the areas between four (neighboring) nodes. Now we look at an arbitrary cell in the grid. For any tree we can draw on this grid, there has to be a way to walk out of the grid without crossing the edges of the tree as shown in Fiugre 1 because there are no loops in a tree. This holds for every cell in the grid, especially for the cell in the middle (or adjacent to the center node) of the grid. Let us assume we leave the grid between two nodes (B and C). These two nodes are neighbors on the grid. If $m$ is even, then the distance between node A and node B or C respectively is at least $\frac{m}{2}$. If $m$ is odd, either of the two routes can be $\frac{m-1}{2}$ but the other is at least $\frac{m+1}{2}$.

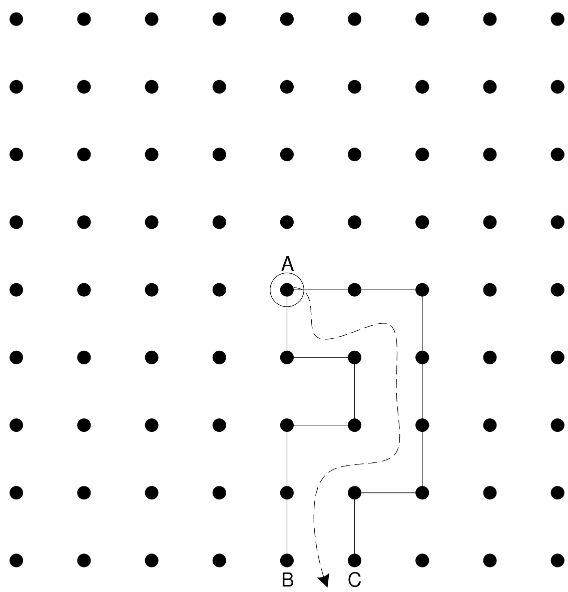


Figure 1: Node A is the center node of the spanning tree (only partially shown). The dashed line is a path through grid cells from the center node A to the outside of the grid.

## 2 Network Updates

**a)** $v_3$ can not change before $v_2$, but $v_2$ needs to wait for $v_1$, requiring three steps in total.

**b)** Let $E'$ be the set of rules that no longer need to be updated and $V'$ the set of nodes with the property that there exists a path to the destination using only rules from $E'$, with

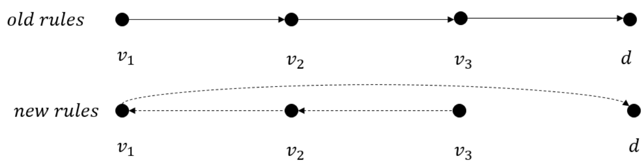


Figure 2: Graph with three rules.

$d \in V'$. Any new rule with the property that it points to a node from $V'$ can not induce a cycle, since all paths from all nodes from $V'$ end at $d$ and $d$ has no outgoing edge. If there are still rules to be updated, then such a rule will always exist, since the set of new rules induces a directed tree with $d$ as its root and all edges in this tree are oriented towards $d$, meaning at least one new rule will point to a node from $V'$.

Note: As seen in **c)**, all rules that can be updated might have this property.

**c)** For a graph with $n$ nodes we use the same concept as in the first item, but with $n$ instead of three vertices $v_i$. Again, $v_i$ can not change before $v_{i-1}$ for $2 \leq i \leq n$, requiring $n$ steps in total.

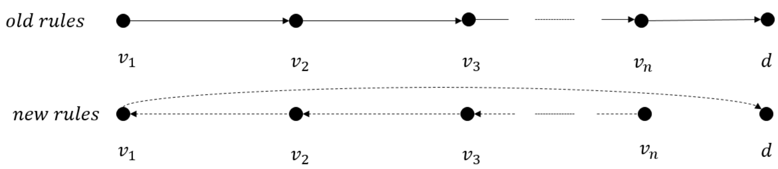


Figure 3: Graph with $n$ rules.

**d)** Obviously, $v_3$ can always change in the first step, without any consequences for the other nodes – see **b)**. But what about $v_2$ and $v_1$? If $v_1$ changes in the first step, then updating $v_2$ would induce a cycle, and vice versa. Therefore two possible ways to migrate the network would be:

- Migrate $v_3$ and $v_1$ in the first step. Migrate $v_2$ in the second step.
- Migrate $v_3$ and $v_2$ in the first step. Migrate $v_1$ in the second step.