Labs on Algorithms for Computer Networks (EDA387)

Please submit your typed solutions in pairs via canvas. In order to submit the assignment, please join one of the lab groups with your lab partner.

1 Value discovery in complete graphs (due: September 17, 2020)

Consider n processors, such that each processor p_i is associated with two registers r_i and s_i (both of constant size that is independent of n). Processor p_i cannot read the value in s_i, however, any other processor can! Processor p_i has both read and write access rights to r_i and any other processor has only read-only access rights to r_i. The value in s_i is unknown to p_i. The other processors can help p_i to discover this unknown value. For example, suppose that $n = 2$. Processor $p_{(i+1) \mod 2}$, can write the value of s_i to $r_{(i+1) \mod 2}$ and then p_i can discover s_i's value by reading $r_{(i+1) \mod 2}$'s value. Please solve the problem, which is to let p_i discover the secret s_i, for the case in which $n > 2$ is any finite known value. Is there a solution when there are no processor identifiers? Prove your claims.

2 Self-stabilizing vertex coloring (due: September 24, 2020)

Let $p_0, \ldots, p_{n-1}$ be n processors on a directed ring that Dijkstra considered in his self-stabilizing token circulation algorithm (in which processors are semi-uniform, i.e., there is one processor, p_1, that is different than other processors but processors have no unique identifiers). Recall that in Dijkstra's directed ring, each processor p_i can only read from $p_{i-1 \bmod n}$'s shared variables and each processor can use shared variables of constant size. Moreover, a vertex coloring is an assignment of colors to vertices (here processors), such that every two vertices that are connected with an edge have different colors.

Design a deterministic self-stabilizing algorithm for vertex coloring in Dijkstra's directed ring and provide its pseudo code. Define a safe configuration with respect to the set of legal executions with respect to your proposed algorithm. Demonstrate the closure and convergence properties. Can you bound the stabilization period? Does your algorithm use an optimal number of colors? What is the number of states (in the finite

state machine that represents each processor) that your algorithm needs? Prove your claims.

3 Self-stabilizing maximum matching on a directed ring with a distinguished node (due: October 1, 2020)

Let $p_0, \ldots, p_{n-1}$ be n processors on a directed ring that Dijkstra considered in his self-stabilizing token circulation algorithm (in which processors are semi-uniform, i.e., there is one distinguished processor, p_0, that runs a different program than all other processors but processors have no unique identifiers). Recall that in Dijkstra's directed ring, each processor p_i can only read from p_{i-1} mod n's shared variables and each processor can use shared variables of constant size. Design a deterministic self-stabilizing matching algorithm that always provides an optimal solution. Please give a clearly written pseudo-code, define the set of legal executions, provide a correctness proof with all the arguments needed to convince the reader that the algorithm is correct and self-stabilizing. Show that your algorithm is always optimal after convergence. What is the number of states (in the finite state machine that represents each processor) that your algorithm needs? Prove your claims.

4 Self-stabilizing algorithm for finding the centers with the tree topology (due: October 8, 2020)

Let $G := (V, E)$ be a graph, where $V = \{p_1, \ldots, p_n\}$ is the set of vertices and $E \subseteq V \times V$. This question considers only tree graphs. A tree is a graph with maximal amount of edges that does not contain a cycle. Let $d(i, j)$ denotes the distance, i.e., the length of a shortest path in G, between vertices p_i and p_j. Let $e(i) = \max\{d(i, j) : p_j \in V\}$ denote the eccentricity of a vertex p_i, which is the distance between p_i and the farthest vertex from p_i in G. Let $center(G) = \{p_i \in V : e(i) \leq e(j), \forall p_j \in V\}$ denote the set of centers of G, the set of vertices in V with minimum eccentricity. Jordan [1] has proved that for trees, there are only two possibilities: (centered trees) exactly one center node, or (bicentered trees) exactly two center nodes that are adjacent.

1. Algorithm 1 presents a template for a self-stabilizing algorithm that finds the center(s) of a tree. The template is missing an expression in line 12. Please complete this expression.

2. Please define the set of legal executions for Algorithm 1 (with the expression that you have completed for line 12).

3. Please demonstrate the convergence for Algorithm 1 (with the expression that you have completed for line 12).

Algorithm 1: Self-stabilizing algorithm for center-finding; code for p_i

1 **Constants:**
2 $N(i) = \{p_j : (p_i, p_j) \in E\}$ p_i's neighbors;
3 **Variables:**
4 h_i stores p_i's h-value.
5 **Macros:**
6 $N_h(i) = \{h_j : (p_i, p_j) \in E\}$ p_i's neighbors h-values (multiset);
7 $N_h^-(i) = N_h(i) \setminus \{\max(N_h(i))\}$ contains all elements of $N_h(i)$ without one maximum h-value. E.g., $N_h(i) = \{2, 3, 3\} \Rightarrow N_h^-(i) = \{2, 3\}$;
8 $leaf(i) = (|N(i)| = 1)$;
9 **do forever**
10 **begin**
11 **if** $(leaf(i) = true) \wedge (h_i \neq 0)$ **then** $h_i \leftarrow 0$;
12 **if** $(leaf(i) \neq true) \wedge (h_i \neq 1 + max(N_h^-(i)))$ **then** $h_i \leftarrow 1 +$ _____;

4. Given Algorithm 1 (with the added expression in line 12, cf. item 1), how can node pi know whether it is part of the center set? Does your answer change in case where there is more than one node in the set $center(G)$?

5. Please modify the code of Algorithm 1, if needed, so that every node knows the entire set of centers in the graph. Moreover, please state your assumptions clearly.

References

[1] Camille Jordan. Sur les assemblages de lignes. *J. Reine Angew. Math*, 70(185):81, 1869.