Principles of Distributed Computing
Exercise 4

1 Deterministic Maximal Independent Set

In the lecture, we discussed a slow but simple deterministic maximal independent set (MIS) algorithm (Algorithm 33) in which the decisions of the nodes are based on their identifiers. The time complexity of this algorithm is $O(n)$.

We might hope that if the nodes with the largest degrees, i.e., the largest number of neighbors, decide to enter the MIS, the set of undecided nodes reduces the most. In the following algorithm we try to exploit the knowledge of the node degrees:

Assume that each node knows its degree and also the degrees of all its neighbors. If a node has a larger degree than all its undecided neighbors, it joins the MIS and informs its neighbors. Once a node v learns that (at least) one of its neighbors joined the MIS, v decides not to join the MIS.

Naturally, the algorithm does not make any progress if two or more neighboring nodes share the largest degree. As this is a difficult problem, we will assume in the following that this situation does not occur, i.e., if a node v has the largest degree, then no neighboring node has the same degree as v.[1]

a) Draw a graph that illustrates that this algorithm has a large time complexity for trees! Give a (non-trivial) lower bound on the (worst-case) time complexity for trees consisting of n nodes!

Hint: A lower bound of $\omega(\log n)$ is sufficient to show that this algorithm is worse than the Fast MIS algorithm for trees, you do not need to find a lower bound of $\Omega(n)$.

Hint: A runtime of $\omega(f(n))$ means that the runtime is *"asymptotically **strictly** larger than f(n)"*.

b) Construct a graph that shows that the time complexity of this algorithm is even worse for arbitrary graphs than for trees! What is the time complexity?

c*) We will now modify the algorithm: The degree of a node v in any given round is only the number of *undecided* neighbors. Prove (tight) lower and upper bounds for this modified algorithm on arbitrary graphs!

[1] The motivation for this constraint is that if we prove that the time complexity is large even if there is no conflict in each step, then being able to break ties clearly does not help.

2 (Local) Reductions

Many problems can be seen as—more or less obvious—variants of others and therefore can be solved by clever use of the same algorithms. In this exercise you may use the algorithms derived in the lecture as subroutines.

a) The *edge coloring* problem is defined as follows: Given a graph $G = (V, E)$, assign each edge $e \in E$ a color $c(e)$ such that no two adjacent edges (i.e., edges sharing a node) have the same color! As usual, the problem should be solved not only fast, but also using a small number of different colors. Give a $2\Delta - 1$ edge coloring algorithm running in time $O(\log n)$ w.h.p.!

Hint: Δ denotes the largest degree of all nodes in G, i.e., $\Delta = \max_{v \in V} d(v)$.

b) Given a graph $G = (V, E)$, a *dominating set* is a subset $D \subseteq V$ such that each node either is in D or has a neighbor in D. The *minimum dominating set* problem is to find a dominating set of minimum cardinality. Give a 3/2-approximation algorithm for this problem on rings which takes $O(\log^* n)$ time!

c) A family of graphs of *bounded independence* is a set of graphs where, for each node, the largest independent set in the one-hop neighborhood (i.e., the direct neighbors) has a size that is bounded by a constant C. Give a C-approximation algorithm to the minimum dominating set problem on graphs of bounded independence running in time $O(\log n)$ w.h.p.!