ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

FS 2009 Prof. R. Wattenhofer / C. Lenzen / T. Locher

Principles of Distributed Computing
Exercise 4: Sample Solution

1 Deterministic Maximal Independent Set

a) Consider the graph consisting of a connected chain of k nodes $v_1, \ldots, v_k$. We add $i-1$ additional edges leading to $i-1$ additional nodes at each node v_i for all $i \in \{1, \ldots, k-1\}$ and k additional nodes and edges to node v_k. The degree $\delta(v_1)$ of v_1 is 1 and for all other nodes $v_i \in \{2, \ldots, k\}$ we have that $\delta(v_i) = i+1$. All additional nodes have degree 1.

In the first round, all nodes except v_k have a neighbor with a larger degree, thus only v_k joins the MIS. Afterwards, v_{k-1} can decide, then v_{k-2} and so on. Thus, after k time all nodes $v_1, \ldots, v_k$ and also the additional nodes have decided to join or not to join the MIS.

The number of nodes in this graph is

$$n = k + \sum_{i=1}^{k}(i-1) + 1 = 1 + \sum_{i=1}^{k} i = 1 + \frac{k(k+1)}{2} < \frac{(k+1)^2}{2}.$$

The time complexity is thus $k \geq \sqrt{2n} - 1 \in \Omega(\sqrt{n})$.

b) Instead of a chain, we now consider a ring of k nodes $v_1, \ldots, v_k$. We use $k-1$ additional nodes $u_1, \ldots, u_{k-1}$ to increase the degrees of the nodes v_i: There is an edge $\{v_i, u_j\}$ from all nodes v_i to all nodes u_j, where $j \in \{1, k-i\}$. It is easy to see that the degree $\delta(v_i)$ of node v_i is $k+2-i$, and that $\delta(u_j) = k-j$.

In the first round, only v_1 joins the MIS. This means that all nodes $u_1, \ldots, u_{k-1}$ and also v_2 and v_k can no longer join the MIS. Thus, in the second round, all these nodes broadcast to all their neighbors that they will not join the MIS. In the third round, only v_3 decides to join the MIS because all other undecided nodes have an undecided neighbor with a larger degree. Subsequently, only v_4 decides (not to join the MIS) in round 4. Repeating this argument, we get that the last node v_{k-1} makes its decision not before round $k-1$. Since $n = k + (k-1) < 2k$, the time complexity is thus $k > \frac{n}{2} \in \Omega(n)$.

2 (Local) Reductions

a) We use Algorithm 19 on the *line graph* of G, i.e., the graph $L(G) = (E, F)$ having the edges of G as nodes, and two nodes (edges of G) are connected exactly if they are adjacent to each other in G (formally: $F = \{\{e, f\} \in E^2 \mid e \neq f \wedge e \cap f \neq \emptyset\}$). Thus, a (node) coloring of $L(G)$ is an edge coloring of G. Since $L(G)$ has $m \in O(n^2)$ nodes and maximum degree $2(\Delta - 1)$ (an edge may be adjacent to $\Delta - 1$ others at each of its nodes), the algorithm will need $O(\log n^2) = O(\log n)$ time and $2(\Delta - 1) + 1 = 2\Delta - 1$ colors.

The line graph can be simulated locally, where nodes of the line graph (i.e., edges of G) are simulated by one of their incident nodes. The nodes simulating adjacent edges are connected by them and therefore at most 2 hops away from each other. Thus, two rounds and (at most) two messages are required to simulate one round of communication and one message on the line graph, respectively. Hence, the time complexity is doubled, but still in $O(\log n)$.

If we do not have edge orientations or identifiers, the decision which of the nodes plays the part of the edge can, e.g., be made w.h.p. in a single round by exchanging random bit strings of size $O(\log n)$ between neighbors.

b) First, we 3-color the ring by means of Algorithm 7 (or its uniform variant from the first exercise sheet). Next, all nodes with color 0 join the dominating set and inform their neighbors. Then, all nodes with color 1 having no neighbor of color 0 join the set and inform their neighbors. Finally, still uncovered nodes with color 2 join the dominating set.

Obviously, the resulting set is a dominating set and the algorithm has a time complexity of $O(\log^* n)$. However, the constructed set is also a (maximal) independent set, as no two neighbors join. An independent set in a ring can not have more than $n/2$ nodes, while a dominating set must contain at least $n/3$ nodes (each node covers itself and its two neighbors). In other words, the computed set is at most a factor of $3/2$ larger than any dominating set and hence also than a minimum dominating set.

c) Again we use Algorithm 19 to compute a maximal independent set I within $O(\log n)$ time. It is also a dominating set (because a node without a neighbor in I could be added) and we claim that it is at most c times larger than a minimum dominating set M.

To prove this, consider a node $v \in I$. Since M is a dominating set, there must be at least one node in $(N(v) \cup \{v\}) \cap M$, i.e., a node from the optimal solution is in v's neighborhood. For each $v \in I$, we count such a node. Because the graph is growth-bounded, no node $m \in M$ is counted more than c times, because there cannot be more than that many independent neighbors of m. Therefore, $|I| \leq c|M|$.