Albert-Ludwigs-Universität
Institut für Informatik
Prof. Dr. F. Kuhn
Mohamad Ahmadi, Hamid Ghodselahi June 15, 2016

Network Algorithms, Summer Term 2016
Problem Set 6 – Sample Solution

Exercise 1: Deterministic Maximal Independent Set Construction

1. All the nodes in V_0 keep the color from the $(\Delta+1)$-coloring of $G[V_0]$. Clearly, this does not create any conflicts in G. For the nodes in V_1, we iterate through all the $\Delta+1$ colors. When considering color x, all nodes that have color x in the $(\Delta+1)$-coloring of $G[V_1]$, select the minimum possible available color (note that these nodes always form an independent set of G). Given on $(\Delta+1)$-coloring of $G[V_0]$ and $G[V_1]$, this allows to compute a $(\Delta+1)$-coloring of G in $\Delta+1$ rounds.

2. For some bit string β of length k (for $k \geq 0$), let V_β be the set of nodes with an identifier whose bit representation starts with β. Assume that all identifiers can be represented by using at most $\ell = O(\log n)$ bits. For some given β of length k, we would like to color the graph $G[V_\beta]$ induced by the nodes in V_β using $\Delta+1$ colors. We prove that this is possible in $(\ell-k)(\Delta+1)$ rounds by induction on $(\ell-k)$. For $k=\ell$, this is clear because for every bit string of length $k=\ell$, there is at most one node in V_β and thus $G[V_\beta]$ can directly be colored with a single color. For the induction step, consider a bit string β of length k and let β_0 and β_1 be the two possible $(k+1)$-bit extensions of β. By induction, we can color $G[V_{\beta_0}]$ and $G[V_{\beta_1}]$ with $\Delta+1$ colors in $(\ell-k-1)(\Delta+1)$ rounds (we can color both of these graphs in parallel!). Using the ideas from the previous exercise, we can then color $G[V_\beta]$ in $\Delta+1$ more rounds. Altogether, we get an algorithm that colors every graph with $\Delta+1$ colors in $\ell(\Delta+1) = O(\Delta \log n)$ rounds.

3. Assume that the network graph G has maximum degree Δ. We can color G with $\Delta+1$ colors in $O(\Delta \log n)$ rounds. Based on this coloring, we can compute an MIS in $O(\Delta)$ additional rounds as follows. Initially, the MIS is empty. We iterate through the $\Delta+1$ colors. When taking care of color x, all nodes with color x that do not yet have a neighbor in the MIS, join the MIS. Overall, we get an MIS algorithm with time complexity $O(\Delta \log n)$.

4. The distributed greedy algorithm adds a node u to the MIS if and only if u has the largest identifier among all non-decided neighboring nodes. We change this rule and always give priority to undecided nodes with larger degrees (ties are broken by IDs). In each phase, the largest ID node, among the nodes with the largest remaining degree ($\geq k$), joins the MIS. Hence, as long as there is a node with degree more than k, at least $k+1$ nodes are removed from the graph in every phase. We therefore need at most $O(n/k)$ rounds until no node of degree larger than k remains.

5. Recall that we assumed that the nodes know n. We fix some k and let the modified greedy algorithm run until there are no nodes of degree more than k remaining. By the observations of Question 4), this requires $O(n/k)$ rounds. For the remaining graph with maximum degree k, we run the algorithm of Question 3). The running time of that algorithm is $O(k \log n)$. Choosing $k = \sqrt{n/\log n}$ we get a deterministic MIS algorithm that computes an MIS in time $O(\sqrt{n \log n})$.

Exercise 2: (Local) Reductions

1. First, we 3-color the ring by means of Algorithm Six-2-Three (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 cannot 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.

2. Again we use one of the fast MIS algorithms to compute a maximal independent set I within $O(\log n)$ time. Observe that I is also a dominating set, due to the maximality of I; if there was a node that was not dominated, i.e., it is not in I and has no neighbor in I, this node would also be independent of I, which contradicts the assumption that I was a *maximal* independent set. Thus, we only need to show that I 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 of bounded independence, no node $m \in M$ is counted more than C times, because there cannot be more than that many independent neighbors of m. Therefore, $|I| \leq C|M|$.