Principles of Distributed Computing
Sample Solution to Exercise 6

1 Maximal vs. Maximum Matching

a) The statement $|M| \leq |M^*|$ follows from maximality of M^*. To see that $|M| \geq \frac{|M^*|}{2}$, we make the following argument based on counting in two ways. Every edge in M^* receives a coin. We redistribute this coins to the edges in M in such a way that every edge receives at most 2 coins. Then $|M^*| \leq 2|M|$.

Consider an edge $e \in M^*$. If $e \in M$, then e keeps the coin. If $e = \{u, v\} \notin M$, then either u or v (or both) have an incident edge in M. Otherwise, M would not be maximal, as $\{u, v\}$ could be added to M. Take one of these edges incident to u or v and give the coin to this edge.

Now, let us argue that every edge in M receives at most 2 coins. Indeed, an edge $\{u, v\} \in M$ can receive a coin from an edge $e^* \in M^*$ only if $u \in e^*$ or if $v \in e^*$. Since a (maximum) matching can at most have one edge incident to u and one edge incident to v, there can be at most two such edges gifting their coin to e.

b) We start with $G = (V, E)$ and iteratively add an (arbitrary) edge from E to a matching M, remove this edge and conflicting (i.e., neighboring) edges from the graph, and repeat until there are no edges left. This constructs a maximal matching. We argue that per edge that is added to M, at most $2\Delta - 1$ edges disappear from the graph. From this, the statement follows.

Adding an edge $e = \{u, v\}$ to M leads to the removal of all edges incident to u and v. These are $\delta(u) + \delta(v) - 1 \leq 2\Delta - 1$ many. The minus 1 comes from the fact that the edge $\{u, v\}$ is counted twice when adding the degrees of u and v.

Remark: Note that this also directly follows from arguing about the size of a MIS in the line graph. It is a well-known fact that a MIS has size at least n/Δ. Since the line graph has $|E|$ nodes and maximum degree $2\Delta - 1$, the claim follows.

2 Maximal Matching in Bipartite Graphs

Let us call the two colors *black* and *white*. In round $2 \cdot i - 1$ for $i = 1, \ldots, \Delta$, every still active black node proposes to an arbitrary incident (still active) edge. The active white nodes accept an arbitrary proposal of one of their incident edges (if any), becomes inactive, and informs its neighbors about the chosen edge as well as about it becoming inactive in round $2 \cdot i$. The chosen edge is added to the matching. The black node whose proposal was accepted as well as all edges incident to the white node become inactive.

Since a black node only proposes to one node per round and becomes inactive after its proposal has been accepted, it cannot have more than one incident edge in the matching. Since a white node only accepts (at most) one proposal and then becomes inactive, it also cannot have more than one incident edge in the matching. The found edge set thus indeed is a valid matching.
A black node has no incident edge in the matching only if all its proposals were not accepted, due to the corresponding white node accepting another proposal (having another incident edge in the matching). A white node remains without incident edge in the matching only if in no round a black nodes sends it a proposal. This only happens if the black nodes all get a proposal accepted before, so if all neighboring black nodes have an incident edge in the matching. The found edge set thus indeed is a maximal matching.

3 Maximal Matching using Forest Decomposition

Remark: This algorithm is due to Panconesi and Rizzi in 2001.

a) Instead of starting with the ID of a node, we can start with its color. The remaining algorithm stays the same, except that we initially have $\log(q)$ instead of $\log(n)$ bits.

b) In round 1, the nodes collect the ID of their neighbors and orient all edges from the smaller-ID node to the larger-ID node. Then for all its outgoing edges, it numbers them from 1 to (at most) Δ. This way, every edge receives exactly one number from 1 to Δ. Let F_i be the graph induced by edges with number i. It is a collection of directed trees: every node has at most one outgoing edge in F_i and there cannot be any cycles since edges are directed from smaller-ID to larger-ID.

c) For each $i = 1, \ldots, \Delta$ and each $c = 1, 2, 3$, each remaining node that has color c in forest F_i, it adds one of its remaining ingoing edges in F_i to the matching. The endpoints of the edge as well as all incident edges are "removed" from the graph.

Since nodes active at the same time must have distance at least 2 (as we are looking at a proper coloring of a forest) and they are adding an ingoing edge to the matching, the chosen set of edges indeed is a matching. It is maximal since a node remains unmatched only if all its (incoming and outgoing) edges lead to already matched nodes.

d) We first compute the forest decomposition from b) in $O(1)$ time, then, in parallel, for each of the Δ forests, compute a 3-coloring in $O(\log^* q)$ from a), then use that 3-coloring to compute a maximal matching as described in c) in $O(\Delta)$ time. Overall, this results in a maximal matching algorithm that runs in $O(\Delta + \log^* q)$ time in graphs with maximum degree Δ and a vertex coloring with q colors.

4 Rounding in Non-Bipartite Graphs

Suppose the 2-decomposition consists of a collection of $\Delta/2$ many triangles. Then rounding will lead to two of the three edges being rounded down to 0 and the third being rounded up by a factor 2. The overall matching size is multiplied by $\frac{2}{3}$, so decreases by a constant factor.
Remark: Repeating this over $\log \Delta$ rounding steps, losing a constant factor c in every iteration, leads to a remaining matching size that is a factor $c^{\log \Delta} = \Theta(\Delta)$ smaller. So even if we would start with a good (or even maximum) matching, over all the rounding steps, we would end up with a matching that is a factor $\Theta(\Delta)$ worse.