Table 1: Execution of Cole-Vishkin 6 color algorithm

<table>
<thead>
<tr>
<th>Step</th>
<th>r</th>
<th>u</th>
<th>v</th>
<th>w</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>104</td>
<td>110</td>
<td>51</td>
<td>170</td>
<td>35</td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>1101000</td>
<td>1101110</td>
<td>110011</td>
<td>100011</td>
<td>1111</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>11</td>
<td>01</td>
<td>100</td>
<td>1000</td>
<td>101</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>01</td>
<td>01</td>
<td>00</td>
<td>00</td>
<td>101</td>
</tr>
</tbody>
</table>

(2.a) \[ Pr[C_u] = \frac{1}{|P(u)|} \sum_{c \in P(u)} Pr(W_{c,N(u)}) \]

\[
Pr[C_u] = \frac{1}{6} \left[ \left( \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 \cdot 1 \right) + \left( \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 \right) + \left( \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 \cdot 1 \right) + \right. \\
\left. \left( \frac{1}{2} \cdot \frac{1}{2} \cdot 1 \cdot 1 \cdot 1 \cdot 1 \right) + (1 \cdot 1 \cdot 1 \cdot 1 \cdot 1) + (1 \cdot 1 \cdot 1 \cdot 1 \cdot 1) \right] 
\]

\[ Pr[C_u] = \frac{1}{6} \left( \frac{1}{16} + \frac{1}{4} + \frac{1}{2} + 1 + 1 \right) \]

\[ Pr[C_u] = \frac{53}{96} \]

(2.b) \( c' = 1, c'' = \{3, 4, 5, 6\} \)
(2.c) Replacing 1 with 6 in the palette of \( v \)

\[
Pr[C_u] = \frac{1}{|P(u)|} \sum_{c \in P(u)} Pr(W_{c,N(u)})
\]

\[
Pr[C_u] = \frac{1}{6} \left[ \left( \frac{1}{2} \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \right) + \left( 1 \cdot \frac{1}{2} \cdot 1 \cdot \frac{1}{2} \cdot 1 \cdot 1 \right) + \left( 1 \cdot 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot 1 \cdot 1 \right) + \left( \frac{1}{2} \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \right) + \left( 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \right) + \left( \frac{1}{2} \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \right) \right]
\]

\[
Pr[C_u] = \frac{1}{6} \left( \frac{1}{8} + \frac{1}{4} + \frac{1}{2} + \frac{1}{2} + 1 \right)
\]

\[
Pr[C_u] = \frac{23}{48} \approx \frac{46}{96} < \frac{53}{96}
\]

The probability goes down as expected.

(3) In the first phase, we run the Cole-Vishkin algorithm to obtain a \( 2^{2\Delta} \)-coloring in \( O(\log^* n) \) rounds. This algorithm runs in the \( \text{CONGEST} \) model. Now we start the second phase of the algorithm. For each color \( c \in \{1, 2, \ldots, 2^{2\Delta}\} \) considered in this order, we process all vertices of color \( c \) in parallel. If a vertex \( v \) of color \( c \) has a neighbor of color \( c' \in \{1, 2, \ldots, c-1\} \) in the MIS, then \( v \) chooses not to join the MIS; otherwise \( v \) joins the MIS. Thus all vertices of color \( c \) are processed in \( O(1) \) rounds and therefore, Phase 2 runs in \( O(2^{2\Delta}) \) rounds. Since it is given that \( \Delta \leq 10 \), \( 2^{2\Delta} \) is bounded above by a constant (though a somewhat large constant). Therefore, Phase 2 runs in \( O(1) \) rounds, also in the \( \text{CONGEST} \) model and the entire algorithm runs in \( O(\log^* n) \) rounds in the \( \text{CONGEST} \).

**Remark:** I did not prove the correctness of the algorithm here, but hopefully it is easy to see.

(4) Node \( v \) initializes its color \( c(v) \) to \( \bot \) and a boolean flag \( done(v) \) to False. To initialize its palette \( P(v) \), node \( v \) executes the following 2-round algorithm. The purpose of this algorithm is simply to count the number of nodes in the 2-neighborhood of each node \( v \).

1. \( v \) sends \( ID_v \) to all neighbors
2. \( v \) receives IDs from neighbors; let \( N_ID(v) \) denote the set of IDs received
3. \( v \) sends \( N_ID(v) \) to all neighbors. (In this step messages can be quite large.)
4. \( v \) receives sets of IDs from neighbors.
5. \( v \) for each neighbor \( u \) do
   7. \( N_ID(v) \leftarrow N_ID(v) \cup N_ID(u) \)
   8. \( P(v) \leftarrow \{1, 2, \ldots, |N_ID(v)|\} \)

After \( P(v) \) has been initialized, node \( v \) repeatedly executes the following 4-round algorithm.

// Pick a tentative color, if not already permanently colored
1. if not \( done(v) \) then
2. \( c(v) \leftarrow \) a color picked uniformly at random from palette \( P(v) \)
3. \( v \) sends \( c(v) \) to all neighbors

// Even permanently colored nodes should continue to pass on received colors to neighbors
4. \( v \) receives colors from neighbors; let \( N_C(v) \) denote the set of colors received
5. \( v \) sends \( N_C(v) \) to all neighbors. (In this step messages can be quite large.)

// Determine if my color collides with the color of any node in my 2-nbd
6. if not \( done(v) \) then
7. \( v \) receives sets of colors from neighbors
The deterministic algorithm for 3-coloring an unoriented tree in $O(n \log n)$ rounds proceeds as follows.

1. Implement the non-distributed algorithm described in the problem in the CONGEST model, but with one change. The given algorithm says that if $e$ has both end points in $S$, orient it arbitrarily. The change we make is to leave such edges unoriented for now.

**Running time:** Each iteration of the while-loop takes $O(1)$ rounds in the CONGEST model and therefore, using the proof in 5(a) we see that this algorithm runs in $O(\log n)$ rounds.

2. Use the Cole-Vishkin algorithm to produce, in parallel for all $i$, $1 \leq i \leq t$, a 3-coloring of $G[S_i]$. 

**Running time:** $O(\log^* n)$ rounds.

**Note:** This is not a 3-coloring of the entire graph because two adjacent nodes, one in $S_i$ and the other in $S_j$, $j \neq i$, can have the same color.

3. Consider the sets $S_1, S_2, \ldots, S_{t+1}$ (in this order, one after the other). For each set $S_t$ and for each $j = 1, 2, 3$, let $S_{t,j}$ denote the subset of $S_t$ of nodes colored $j$. For each set $S_t$ orient every edge $e$ with both endpoints in $S_t$ from the endpoint with larger color to the endpoint with smaller color. In other words, edges with both endpoints in $S_t$ will be oriented from $S_{t,3}$ to $S_{t,2}$ and $S_{t,1}$ and from $S_{t,2}$ to $S_{t,1}$. For each $j = 1, 2, 3$ (considered in this order), process all nodes in $S_{t,j}$ in parallel, as follows. Each node $v$ in $S_{t,j}$ examines the at most two out-neighbors it has and assigns itself a color from $\{1, 2, 3\}$ distinct from the colors assigned to the out-neighbors.

**Running time:** $O(t) = O(\log n)$ rounds.
This algorithm runs in $O(\log n)$ rounds. We will now show that it is correct, i.e., it produces a proper 3-coloring of $G$. The proof is by induction and the inductive hypothesis is the following:

After set $S_t$ has been processed in Step 2 above, we have constructed a proper 3-coloring of the subgraph of $G$ induced by sets $S_t \cup S_{t+1} \cup \cdots \cup S_t$.

Showing this for $i = 1$ gives us a proper 3-coloring of $G$.

The inductive hypothesis is trivially true for $i = t + 1$. Now suppose that we have processed set $S_t$ and have a proper 3-coloring of the graph induced by $S_t \cup S_{t+1} \cup \cdots \cup S_t$. Now the algorithm processes the set $S_{t-1}$ in three sub-steps: first $S_{t-1,1}$ is processed, then $S_{t-1,2}$ is processed, and then $S_{t-1,3}$ is processed. After set $S_{t-1,1}$ is processed, we are guaranteed that the subgraph induced by $S_{t-1,1} \cup S_t \cup S_{t+1} \cup \cdots \cup S_t$ is properly 3-colored. This is because no two nodes in $S_{t-1,1}$ are adjacent and therefore they can choose colors independently. Furthermore, each node $v \in S_{t-1,1}$ has no out-neighbors in $S_1 \cup S_2 \cup \ldots \cup S_{t-1}$ and at most two neighbors in $S_1 \cup S_{t+1} \cup \cdots \cup S_t$ and therefore $v$ can choose a “permanent” color from $\{1, 2, 3\}$ distinct from its neighbors in $S_1 \cup S_{t+1} \cup \cdots \cup S_t$. Similarly, after set $S_{t-1,2}$ is processed, we are guaranteed that the subgraph induced by $S_{t-1,2} \cup S_{t-1,1} \cup S_t \cup S_{t+1} \cup \cdots \cup S_t$ has a proper 3-coloring. Note that when a node $v \in S_{t-1,2}$ is processed, any neighbor(s) it has in $S_{t-1,1}$ have already received a “permanent” color and $v$ will take this into account when assigning itself a “permanent” color from $\{1, 2, 3\}$. The same argument holds for $S_{t-1,3}$ and as a result the inductive hypothesis will hold after set $S_{t-1}$ has been processed.

(6) We start the algorithm with nodes exchanging their $r$-values. This takes $O(1)$ rounds in the CONGEST model. If two neighboring nodes have the same $r$-values, then the algorithm aborts without producing a coloring. Otherwise, we start the greedy $(\Delta + 1)$-coloring algorithm.

We first show that the probability that two neighboring nodes will have the same $r$-values is small. Let $u$ and $v$ be two nodes in the network. Then,

$$Pr\left(r(u) = r(v)\right) = \frac{1}{2c' \cdot |c \log_2 n|}.$$ 

By choosing $c'$ to be a large enough constant (e.g., $c' = 3$), we get that $Pr(r(u) = r(v)) < \frac{1}{n^3}$. In an $n$-node cycle, there are $n$ pairs of neighboring nodes and using the union bound on these $n$ pairs, we get

$$Pr\left(\text{There exist neighbors } u \text{ and } v: r(u) = r(v)\right) < \frac{1}{n^2}.$$ 

(Make sure you understand this calculation.) We say that there is a collision if two neighbors have the same $r$-values. Thus, $Pr(\text{no collision}) > 1 - 1/n^2$.

We now condition the rest of the analysis on the event that there is no collision. For two neighbors $v_1$ and $v_2$,

$$Pr(r(v_1) > r(v_2) \mid \text{no collision}) = \frac{1}{2}.$$ 

This follows from the fact that by symmetry $r(v_1) > r(v_2)$ and $r(v_1) < r(v_2)$ are equally likely. Now consider a path $(v_1, v_2, v_3)$ in the cycle. Then,

$$Pr\left(r(v_1) > r(v_2) > r(v_3) \mid \text{no collision}\right) = Pr\left(r(v_1) > r(v_2) \mid \text{no collision}\right) \times Pr\left(r(v_2) > r(v_3) \mid r(v_1) > r(v_2) \text{ and no collision}\right) = \frac{1}{2} \cdot \frac{1}{3}.$$ 

Continuing in this manner, we see that for a path $(v_1, v_2, \ldots, v_t)$ in the cycle

$$Pr\left(r(v_1) > r(v_2) > \cdots > r(v_t) \mid \text{no collision}\right) = \frac{1}{t!} \leq \frac{1}{2^{t-1}}.$$
Now let \( t = 3\lceil \log_2 n \rceil + 2 \) and for this value of \( t \) we see that

\[
Pr\left( r(v_1) > r(v_2) > \cdots > r(v_t) \mid \text{no collision} \right) < \frac{1}{2n^2}.
\]

We call a path \( P = (v_1, v_2, \ldots, v_t) \) in the cycle **decreasing** if \( r(v_1) > r(v_2) > \cdots > r(v_t) \). There are \( 2n \) length-\( t \) paths in the cycle and taking a union bound over these we see that

\[
Pr\left( \text{There exists a length-} t = 3\lceil \log_2 n \rceil + 2 \text{ decreasing path} \mid \text{no collision} \right) < \frac{1}{n^2}.
\]

Thus conditioned on the “no collisions” event, with probability more than \( 1 - 1/n^2 \), the greedy \((\Delta + 1)\)-coloring (which is a 3-coloring since \( \Delta = 2 \)) algorithm will run in at most

\[
3\lceil \log_2 n \rceil + 2 = \Theta(\log n)
\]

rounds. In other words,

\[
Pr\left( \text{Greedy algorithm produces a 3-coloring in } \Theta(\log n) \text{ rounds} \mid \text{no collision} \right) > 1 - \frac{1}{n^2}.
\]

Finally, using the fact that \( Pr(A \text{ and } B) = Pr(A\mid B) \cdot Pr(B) \), we see that the probability that there is no collision \textit{and} the greedy algorithm produces a 3-coloring in at most \( t = 3\lceil \log_2 n \rceil + 2 \) rounds is more than

\[
(1 - 1/n^2) \cdot (1 - 1/n^2) > 1 - 1/n.
\]