Coloring Unstructured Wireless Multi-Hop Networks

Johannes Schneider
Roger Wattenhofer
Overview

• Motivation
• Model
• Illustration: A simple algorithm
• Related work and contribution
• Algorithm
• Brief analysis
Motivation

• Time division multiple access (TDMA) important way of media access control for wireless networks. It can
  – Reduce energy consumption of nodes
  – Increase throughput
  – Increase reliability of communication

• Coloring foundation for TDMA
Model and definitions

- **(Distance d) maximal independent set (MIS)**
  - For a node v: itself or a node within distance <d is in MIS
  - Nodes u,v in MIS have distance ≥ d

- **Unit Disk Graph (UDG)**
  - Geometrical graph
    - Edge between nodes u,v if dist(u,v) < 1
  - Bounded-independence
    - Maximum size of an independent set in the neighborhood of a node is at most 5

- **(Distance d) coloring**
  - For a node v all nodes within distance d have a distinct color from v
Model

• Collisions/Interference possible, but not detectable
  – Node cannot distinguish a collision from no transmission

• Asynchronous wake-up and no failures
  – Node wakes up at unknown time and operates without errors
  – Links(edges) do not fail

• (Little) topology information
  – Need the number of nodes $n$
  – Faster if maximum degree $\Delta$ known

• Node has unique ID

• Synchronized rounds
A simple algorithm for (distance 1) coloring

Every node picks and transmits a color randomly
Related work and contribution

- Lower bound $\Omega(\Delta)$ time for $\Delta+1$ coloring

- Message passing model
  - no interference
  - Node can transmit distinct messages to neighbors

<table>
<thead>
<tr>
<th>Paper</th>
<th>Time</th>
<th>Colors</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moscriboda, ‘05</td>
<td>$O(\Delta \log n)$</td>
<td>$O(\Delta)$</td>
<td>(Only) 1</td>
</tr>
<tr>
<td>This work</td>
<td>$O(\Delta + \log \Delta \log n)$</td>
<td>$\Delta+1$</td>
<td>Up to a constant with $O(\Delta)$ colors</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Paper</th>
<th>Time</th>
<th>Colors</th>
<th>Graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schneider, ’08</td>
<td>$O(\log^* n)$</td>
<td>$\Delta+1$</td>
<td>UDG</td>
</tr>
<tr>
<td>Luby86</td>
<td>$O(\log n)$</td>
<td>$\Delta+1$</td>
<td>general</td>
</tr>
</tbody>
</table>
Common simple solution strategy

• Every node picks and transmits a color randomly
• Problem
  – All neighbors must know transmitted color but transmitter doesn‘t know if neighbors received anything
  => A node must retransmit color several times to be sure that all neighbors actually received its chosen color
Main idea – Leaders coordinate

• Elect leaders
• Leaders coordinate and synchronize
  – Leader and its neighbors iterate 3 synchronized steps
    i. Neighbors randomly request to choose a color, leader listens
    ii. Feedback by leader (if received request), neighbors listen
    iii. If no collision, requestor chooses an available color and transmits it
Algorithm

• Upon wake-up: Wait and listen for some time
• Iterate two steps
  1. Compute leaders
  2. Leader coordinate and synchronize
Step 1: Compute leaders

1. MIS
   - Use [Moscriboda, `05]
   - Works for asynchronous wake-up

2. Leaders = Distance 6 MIS on MIS
   - Use [Schneider, `08]
   - But it is a message passing algorithm for synchronous wake-up!
   - Can be converted using broadcasts and local synchronizers
Step 2: Leader coordinates and synchronizes

a) Leader broadcasts “DoNot Transmit” up to 3 hops

b) Leader initiates estimation of number of uncolored neighbors
   - Neighbors of leader transmit with probability $\frac{1}{2}$ for $\log n$ slots, then with prob. $\frac{1}{4}$ for $\log n$ slots, then with $\frac{1}{8}$ for $\log n$ slots…
   - Number of neighbors $\approx \frac{1}{\text{probability for which received most messages}}$

c) A leader and its neighbors iterate 3 (synchronized) steps
   i. (Some) neighbors transmit request to choose a color
   ii. Leader grants request (if it receives one)
   iii. The neighbor transmits its chosen color
Step 2: Leader coordinates coloring of neighbors

- A leader and its neighbors iterate 3 (synchronized) steps
  a) Neighbors of leader transmit a request with some *probability*
  b) Leader grants the request (if it receives one)
  c) The neighbor transmits its chosen color

- Probability to transmit request
  - Initially, $1 / \text{Number of (uncolored) neighbors of leader}$
  - A node doubles probability, if it did not receive “many” messages during the last “couple” slots
    - i.e. received less than $c_1 \log n$ out of $c_2 \log n$ last slots
Arbitrary wake-up

- If wake-up, how long do I have to listen in order not to disturb color assignment by leader?
  - Assignment might take $O(\Delta)$ time, if don’t know $\Delta$ must wait $O(n)$
  - => not acceptable
  - **Solution**: Leader interrupts color assignment every “couple” of rounds and broadcasts again

- If wake-up, which colors are taken?
  - **Solution**: If node detects color conflict, it can place a veto in a newly introduced veto phase
Time complexity overview

• Iterate 2 steps
  1. Compute leaders
     a) MIS $S$: $O(\log \Delta \log n)$
     b) Leaders = (Distance 6) MIS on MIS $S$: $O(\log \Delta \log n)$
  2. Leader coordinates coloring of its neighbors
     a) Get number of neighbors: $O(\log \Delta \log n)$
     b) A leader and its neighbors iterate 3 steps: $O(\Delta + \log \Delta \log n)$

• How many iterations (after wake-up of last node)?
  – In an iteration either a node gets colored or there is a node within distance 6, that colors all its neighbors.
  => Leaders from different iterations are independent
  – Since have bounded independence, only constant many nodes independent nodes within distance 6

• Total: $O(\Delta + \log \Delta \log n)$
Thanks for your attention