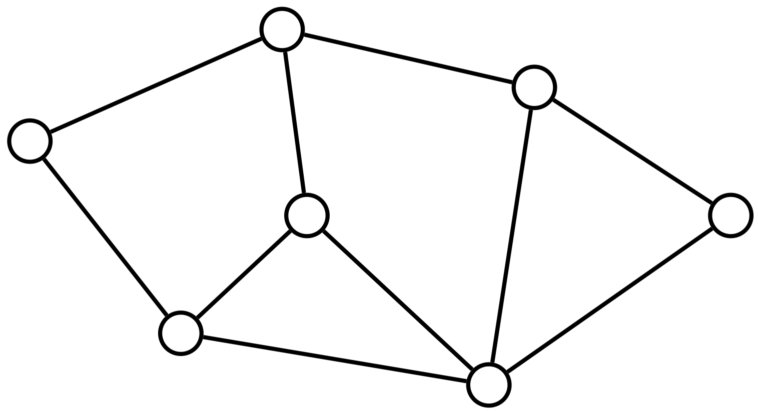
Distributed Algorithms

Inge Li Gørtz

General Model

- Network with n computers (nodes) connected via communication channels (edges).



- Messages. Nodes can exchange messages with neighbors.
- Communication rounds. All nodes perform the same algorithm synchronously in parallel:
  - Receive messages
  - Process
  - Send

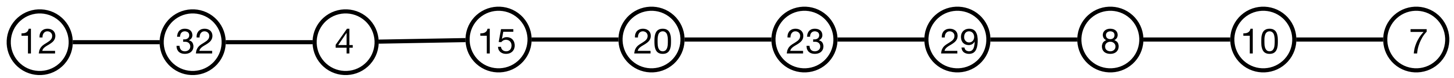
Path colouring

- Path coloring. No neighbouring nodes have the same color.

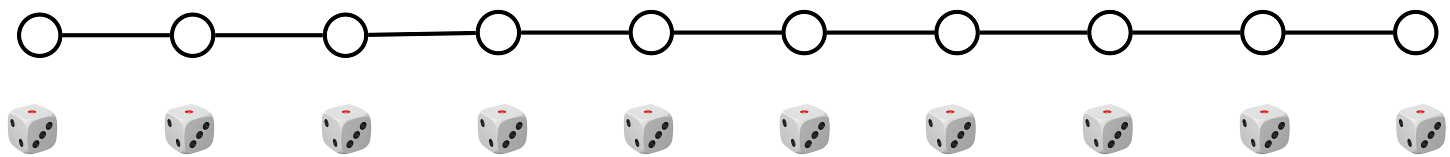
Path colouring

- Path coloring. No neighbouring nodes have the same color.

- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Impossible without unique identifiers or randomness:
  - Each node has a unique name/identifier,



  - or each node has a source of random bits.



Path colouring

- Path coloring. No neighbouring nodes have the same color.

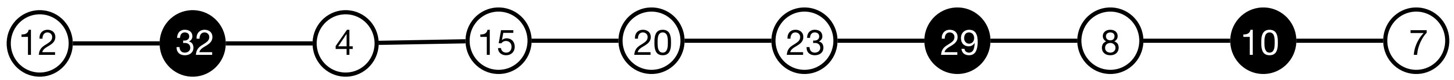
- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Assume we have unique identifiers.

- P3C algorithm.
  - c = id.
  - Repeat forever:
    - Send message c to all neighbors.
    - Receive messages M from neighbors.
    - If $c \neq \{1,2,3\}$ and $c >$ all messages received in this round:
      - $c \leftarrow \min(\{1,2,3\} \backslash M\})$

Path colouring

- Path coloring. No neighbouring nodes have the same color.

- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Assume we have unique identifiers.



- P3C algorithm.
  - c = id.
  - Repeat forever:
    - Send message c to all neighbors.
    - Receive messages M from neighbors.
    - If $c \neq \{1,2,3\}$ and $c >$ all messages received in this round:
      - $c \leftarrow \min(\{1,2,3\} \backslash M\})$

Path colouring

- Path coloring. No neighbouring nodes have the same color.

- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Assume we have unique identifiers.

- P3C algorithm.
  - c = id.
  - Repeat forever:
    - Send message c to all neighbors.
    - Receive messages M from neighbors.
    - If $c \neq \{1,2,3\}$ and $c >$ all messages received in this round:
      - $c \leftarrow \min(\{1,2,3\} \backslash M\})$

Path colouring

- Path coloring. No neighbouring nodes have the same color.

- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Assume we have unique identifiers.

- P3C algorithm.
  - c = id.
  - Repeat forever:
    - Send message c to all neighbors.
    - Receive messages M from neighbors.
    - If $c \neq \{1,2,3\}$ and $c >$ all messages received in this round:
      - $c \leftarrow \min(\{1,2,3\} \backslash M\})$

Path colouring

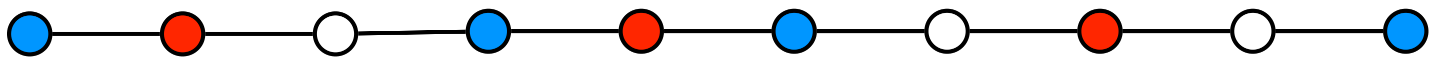
- Path coloring. No neighbouring nodes have the same color.

- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Assume we have unique identifiers.

- P3C algorithm.
  - c = id.
  - Repeat forever:
    - Send message c to all neighbors.
    - Receive messages M from neighbors.
    - If $c \neq \{1,2,3\}$ and $c >$ all messages received in this round:
      - $c \leftarrow \min(\{1,2,3\} \backslash M\})$

Path colouring

- Path coloring. No neighbouring nodes have the same color.

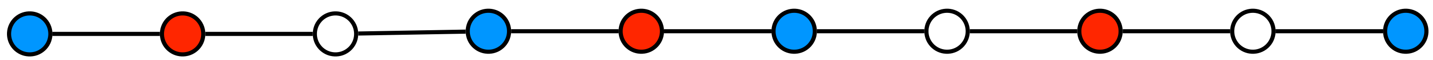


- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Assume we have unique identifiers.

- P3C algorithm.
    - c = id.
    - Repeat forever:
        - Send message c to all neighbors.
        - Receive messages M from neighbors.
        - If $c \neq \{1,2,3\}$ and $c >$ all messages received in this round:
            - $c \leftarrow \min(\{1,2,3\} \backslash M\})$

Path colouring

- Path coloring. No neighbouring nodes have the same color.



- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Assume we have unique identifiers.

- P3C algorithm.
  - c = id.
  - Repeat forever:
    - Send message c to all neighbors.
    - Receive messages M from neighbors.
    - If $c \neq \{1,2,3\}$ and $c >$ all messages received in this round:
      - $c \leftarrow \min(\{1,2,3\} \backslash M\})$

Path colouring

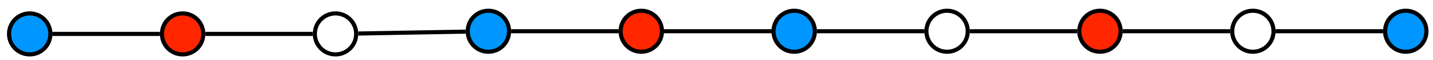
- Path coloring. No neighbouring nodes have the same color.

- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Assume we have unique identifiers.

- P3C algorithm.
  - c = id.
  - Repeat forever:
    - Send message c to all neighbors.
    - Receive messages M from neighbors.
    - If $c \neq \{1,2,3\}$ and $c >$ all messages received in this round:
      - $c \leftarrow \min(\{1,2,3\} \backslash M\})$

Path colouring

- Path coloring. No neighbouring nodes have the same color.

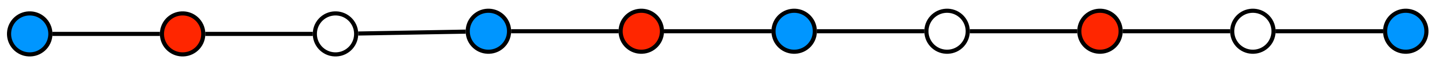


- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Assume we have unique identifiers.

- P3C algorithm.
  - c = id.
  - Repeat forever:
    - Send message c to all neighbors.
    - Receive messages M from neighbors.
    - If $c \neq \{1,2,3\}$ and $c >$ all messages received in this round:
      - $c \leftarrow \min(\{1,2,3\} \backslash M\})$

Path colouring

- Path coloring. No neighbouring nodes have the same color.



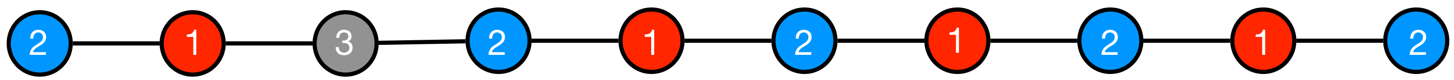
- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Assume we have unique identifiers.

- P3C algorithm.
  - c = id.
  - Repeat forever:
    - Send message c to all neighbors.
    - Receive messages M from neighbors.
    - If $c \neq \{1,2,3\}$ and $c >$ all messages received in this round:
      - $c \leftarrow \min(\{1,2,3\} \backslash M\})$

Path colouring

- Path coloring. No neighbouring nodes have the same color.

- 3-coloring. Color path with 3 colors $\{1,2,3\}$.
- Assume we have unique identifiers.



- P3C algorithm.
  - c = id.
  - Repeat forever:
    - Send message c to all neighbors.
    - Receive messages M from neighbors.
    - If $c \neq \{1,2,3\}$ and $c >$ all messages received in this round:
      - $c \leftarrow \min(\{1,2,3\} \backslash M\})$

Faster deterministic path coloring

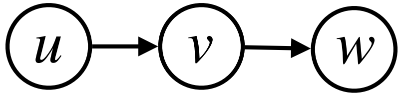
- Assume we have unique identifiers and path is directed.

$$12 \to 32 \to 4 \to 15 \to 20 \to 23 \to 29 \to 8 \to 10 \to 7$$

- Algorithm runs in rounds.
  - In each round reduce the number of colors from 2^x to $2x$.
  - Maintain that it is a proper coloring.
- Round for each node u with color $c(u)$:
  - Send color to predecessor.
  - Know current color $c_o(u) = c(u)$ and color of successor $c_1(u)$. Consider their bit representations.
  - Compute:
    - $i(u)$: the ***index*** of the first bit where $c_0(u)$ and $c_1(u)$ differ.
    - $b(u)$: the value of bit $i(u)$ in $c_0(u)$.
    - Set $c(u) = 2 \cdot i(u) + b(u)$

Correctness

- If we had a proper coloring then it is still a proper coloring:

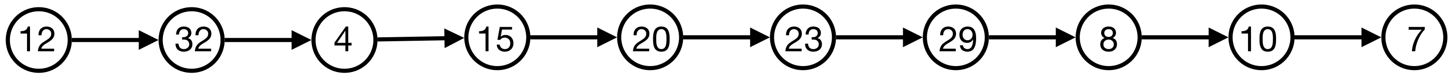


  - Show $c(u) \neq c(v)$. Know $c_0(u) \neq c_1(u)$.
  - 2 cases:
    - $i(u) = i(v) = i$: Then $b(u) \neq b(v) \Rightarrow c(u) \neq c(v)$.
    - $i(u) \neq i(v)$: no matter how we choose $b(u) \in \{0,1\}$ and $b(v) \in \{0,1\}$ then $c(u) = 2 \cdot i(u) + b(u) \neq 2 \cdot i(v) + b(v) = c(v)$.

- Reduction in number of colors:
  - Need x bits to represent the 2^x different colors.
  - Number of different colors is $2x$: we have $0 \leq c(u) \leq 2x - 1$.

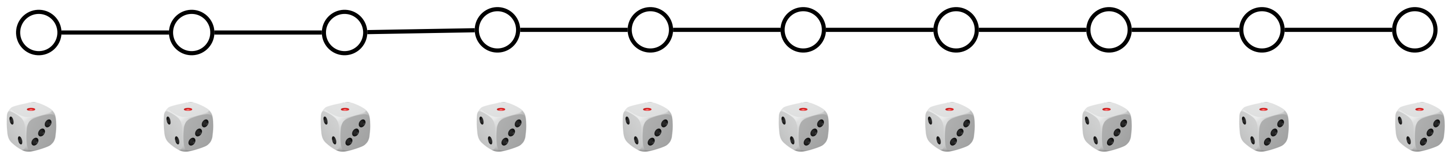
Faster deterministic path coloring

- Assume we have unique identifiers and path is directed.



- Algorithm runs in rounds.
  - Initially, color = id.
  - Continue until at most 6 colors: $O(\log^* n)$ rounds
    - In each round reduce the number of colors from 2^x to $2x$.
  - Use the PC3 algorithm to reduce the number of colors from 6 to 3. 3 rounds

Randomized path coloring



- Each node u has a flag $s(u)$ that indicates it has stopped.
- In each round:
  - Each node u that is not stopped picks a color $c(u) \in \{1,2,3\}$ uniformly at random.
  - Send new color $c(u)$ to neighbors.
  - If new color different from the neighbors colors set $s(u) = 1$.

- Consider node u:
  - Probability that u gets a new color in a round?
  - Expected number of rounds before u has a color?
  - Probability that u does not have a color after k rounds?

Randomized path coloring

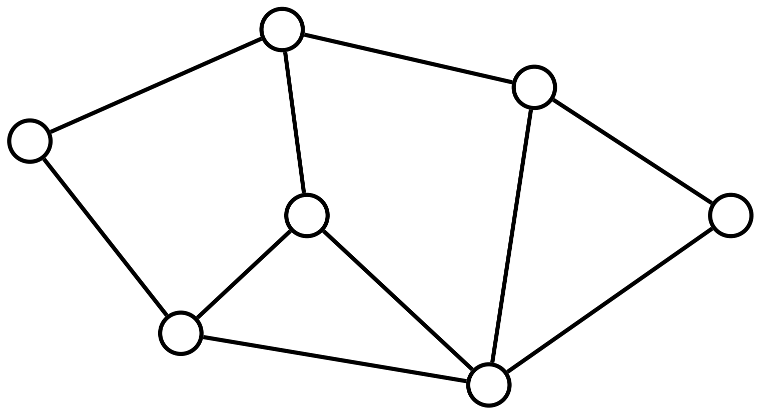
- How many rounds do we need to get that the probability that u does not have a color yet is at most $\dfrac{1}{n^{C+1}}$ for some constant C ?

- Probability that there is a node that did not stopped after this many rounds:

- With high probability all nodes have stopped.

Congest Model

- Network with n computers (nodes) connected via communication channels (edges).



- Identifiers. Nodes has a unique identifier id: $V \rightarrow \{1,2,\ldots,n^c\}$ for some constant c.
- Messages. Nodes can exchange messages with neighbors.
- Communication rounds. All nodes perform the same algorithm synchronously in parallel:
  - Receive messages
  - Process
  - Send
- Message size. In each round over each edge send message of size O(log n) bits.