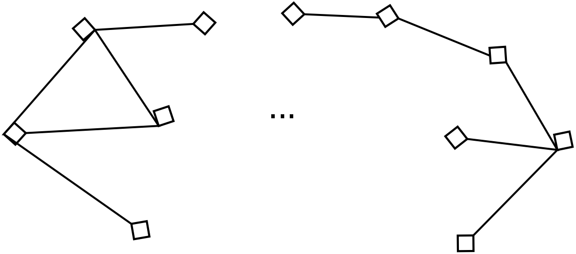
Distributed Algorithms 1

Philip Bille   Inge Li Gørtz   Eva Rotenberg

Distributed Networks



all our favourite questions:

- Is vertex A connected with vertex B?
- shortest route from A to B?
- colouring?
- (minimum/maximum cut, bipartiteness, etc.)

Distributed: Colouring a path

Reminder, (k-) colouring:

Each vertex is given a colour $c : V \to \{1, \ldots, k\}$

such that each edge uv is bichromatic, i.e. $c(u) \neq c(v)$.

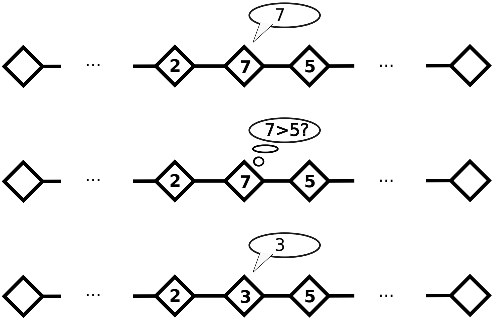
RAM-model colouring of a path? Trivial.

1 — 2 — 1 — 2 — 1 —2 — *cdots*

Distributed colouring of a path? Less trivial.

Distributed path colouring algorithm: P3C

Assume every vertex has a unique identifier of $O(\log n)$ bits.



Round 1:

Send ID to neighbours

All other rounds:

Read neighbours' ID

if my ID is largest and > 3,

then choose ID $\in \{1, 2, 3\}$.

Send ID to neighbours.

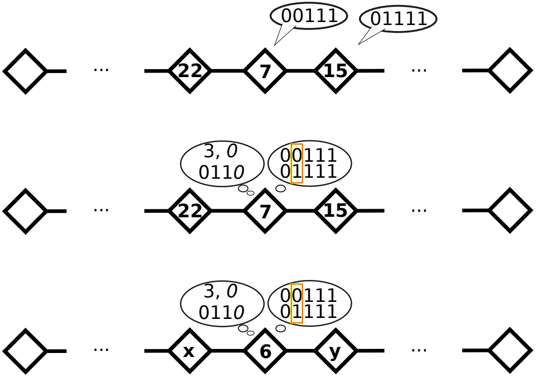
Neighbours cannot both be largest $\Rightarrow c(v_i) \neq c(v_{i+1})$

All IDs are different $\Rightarrow$ some v_i is max $\Rightarrow$ at least that v_i will change colour $\Rightarrow$ every round, at least one fewer vertex has ID> 3 $\Rightarrow$ terminates.

Question: How many rounds before this terminates? worst-case $n + 1$.

Distributed path colouring algorithm: Faster colouring

Assume all IDs are in $\{1, 2, \dots 31\}$.



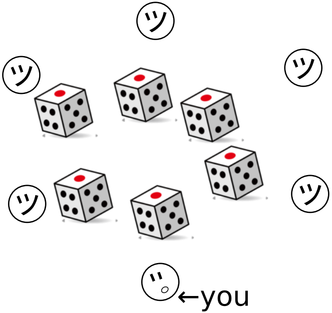
Index: 3

Value: *0*.

Correctness: Consider v_j, v_{j+1}, v_{j+2}. If v_j and v_{j+1} differ in the same digit as v_{j+1} and v_{j+2}, then the *bit value* is different. Otherwise, the bit index is different. Analysis: Exercise.

(Exercises)

Randomised colouring



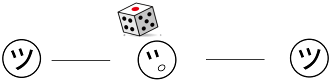
Imagine sitting with 5 friends, and everyone rolls one dice.

Probability you roll something different from everyone else?

Their dice values are at most 5 different values.

So at least $1/6$ chance you roll something unique.

Randomised 3-colouring a path



Even easier: only two friends.

Roll "3-sided dice" until unique among neighbours, stop once unique.

1st round: Probability $\geq 1/3$ of rolling something unique and stopping.

2nd round: Another $\geq 1/3$ chance of stopping,

so $\leq 2/3 \cdot 2/3$ chance of <u>not</u> stopping in rounds 1 or 2.

3rd round: $\leq 2/3 \cdot 2/3 \cdot 2/3$ risk of <u>not</u> stopping in rounds 1, 2, or 3.

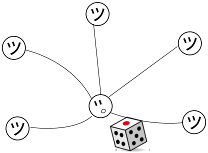
...

k'th round: $\leq (2/3)^k$ risk of <u>not</u> stopping in rounds 1,2,..., k.

So, if $k = (C+1)\log_{3/2} n$, chance of **not stopped** is $\leq 1/n^{C+1}$.

But that was for one vertex. If there are n vertices, the risk of <u>any one</u> not stopped becomes $\leq n \cdot 1/n^{C+1} = 1/n^C$

Randomised Δ+1-colouring

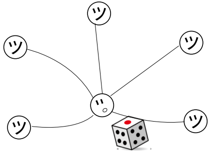


Degree Δ, with a $\Delta + 1$-sided dice, we risk a $1/(\Delta + 1)$ chance of luck.

The argument from before with considering $(\Delta/(\Delta + 1))^k$ is no longer so attractive for large values of Δ.

Solution: somehow take turns.

Randomised Δ+1-colouring



Degree: Δ.

Algorithm: Every node who has not halted is active with probability $\frac{1}{2}$.

Not active: color=blank. Active: random free colour.

When unique: stick with that colour and halt.

Consider an active vertex with k non-halting neighbours, and $k+1$ free colours, conflict with neighbour x is: $< 1/k$ if x is active, 0 if x is inactive. x is active with probability $\frac{1}{2}$. So conflict probability $\frac{1}{2} \cdot 1/k$. Since we had k non-halting neighbours, total conflict probability $\leq k \cdot \frac{1}{2} \cdot 1/k = \frac{1}{2}$. Total probability: $\geq \frac{1}{2} \cdot \frac{1}{2}$ to be active and unique.

So, after κ rounds, $\leq (3/4)^{\kappa}$ risk of vertex v not halting.

As before: $\kappa = (C+1)\log_{4/3} n \Rightarrow 1/n^C$ total risk of not halting.