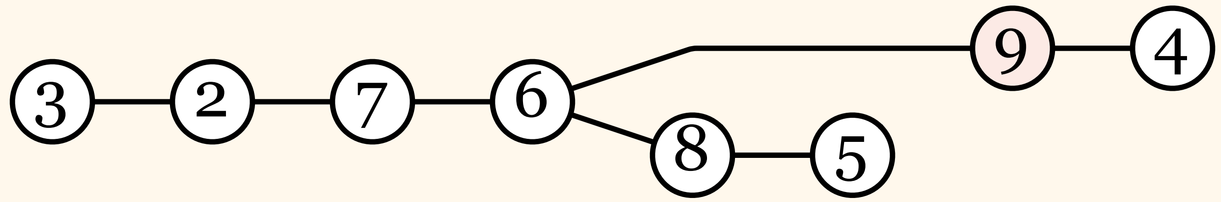
Unique Identifiers

DDA Course

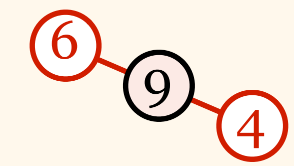
week 5

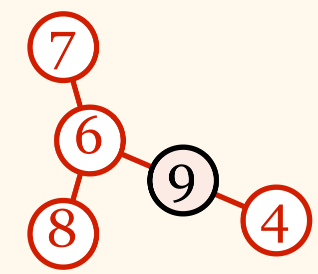
Unique Identifiers

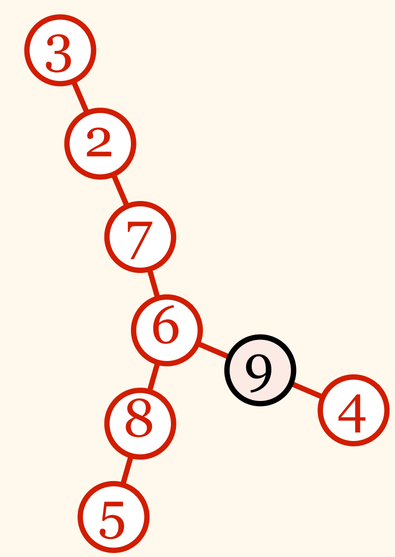
- Networks with *globally unique identifiers*
  - IPv4 address, IPv6 address, MAC address, IMEI number, …
- "Everything" can be discovered
  - in a connected graph G, all nodes can discover full information about G in time $O(\mathrm{diam}(G))$



round 1:	{2,3}	{2,3} {2,7}	{2,7} {6,7}	{6,7} {6,8} {6,9}	{5,8} {6,8}	{5,8}	{4,9} {6,9}	{4,9}
round 2:	{2,3} {2,7}	{2,3} {2,7} {6,7}	{2,3} {2,7} {6,7} {6,8} {6,9}	{2,7} {4,9} {5,8} {6,7} {6,8} {6,9}	{5,8} {6,7} {6,8} {6,9}	{5,8} {6,8}	{4,9} {6,7} {6,8} {6,9}	{4,9} {6,9}
...								
round 5:	{2,3} {2,7} {4,9} {5,8} {6,7} {6,8} {6,9}	{2,3} {2,7} {4,9} {5,8} {6,7} {6,8} {6,9}	{2,3} {2,7} {4,9} {5,8} {6,7} {6,8} {6,9}	{2,3} {2,7} {4,9} {5,8} {6,7} {6,8} {6,9}	{2,3} {2,7} {4,9} {5,8} {6,7} {6,8} {6,9}	{2,3} {2,7} {4,9} {5,8} {6,7} {6,8} {6,9}	{2,3} {2,7} {4,9} {5,8} {6,7} {6,8} {6,9}	{2,3} {2,7} {4,9} {5,8} {6,7} {6,8} {6,9}







Unique Identifiers

- “Everything” can be discovered
  - in a connected graph G, all nodes can discover full information about G in time $O(\mathrm{diam}(G))$
- “Everything” can be solved
  - once all nodes know G, solving a graph problem is just a local state transition
- Key question: what can be solved *fast*?

Graph Colouring

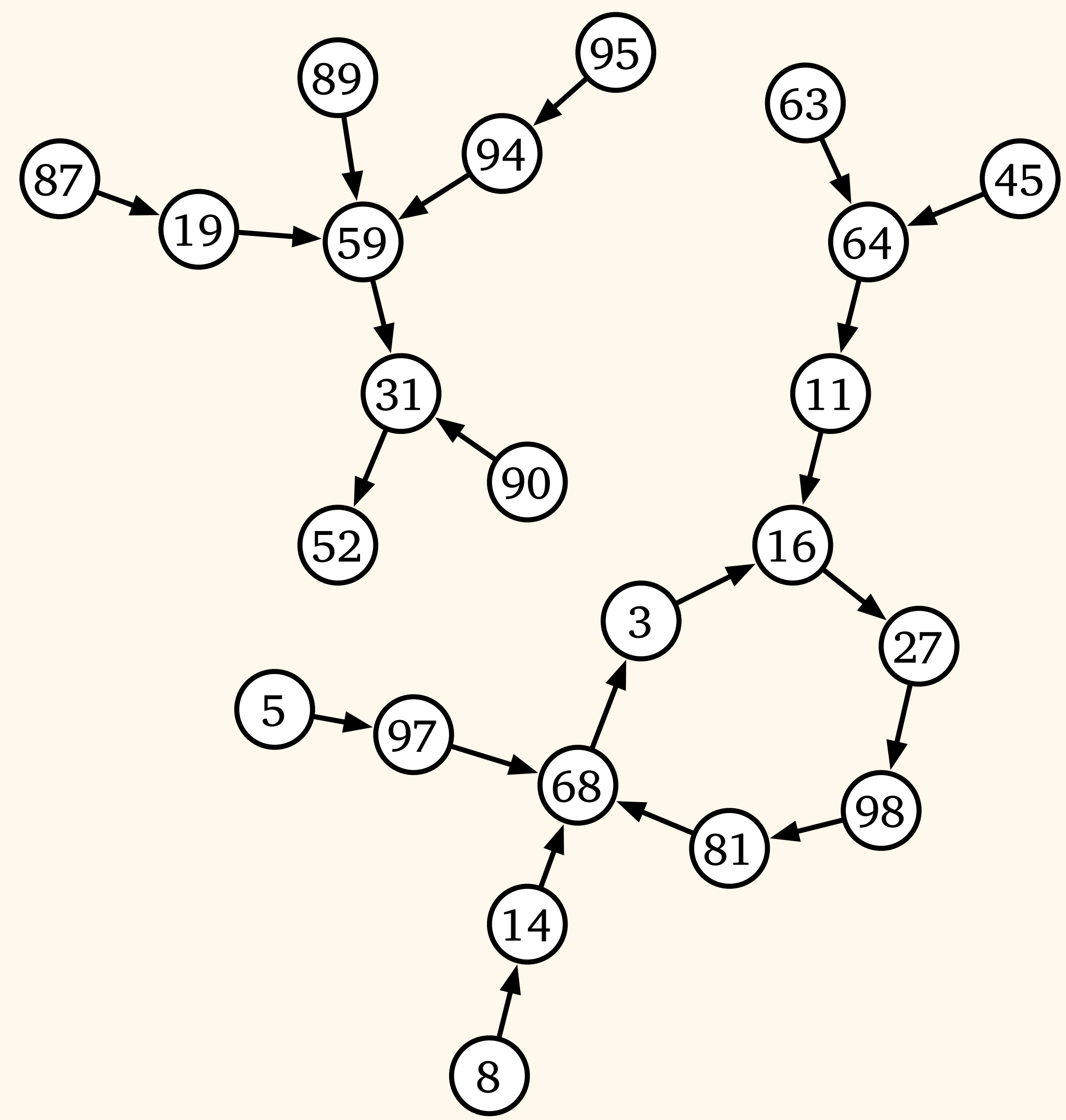
- Given unique identifiers,
  can we find a graph colouring fast?
  - unique identifiers from $\{1, 2, \ldots, x\}$ can be interpreted as a graph colouring with x colours
  - problem: huge number of colours
  - we only need to solve a *colour reduction* problem:
    given an x-colouring, find a y-colouring
    for a small $y < x$

Greedy Graph Colouring

- All nodes of colour x pick the smallest free colour in their neighbourhood
  - there is always a free colour in the set $\{1, 2, \ldots, \Delta + 1\}$
  - reduces the number of colours from x to $x - 1$, assuming that $x > \Delta + 1$
- Very slow…

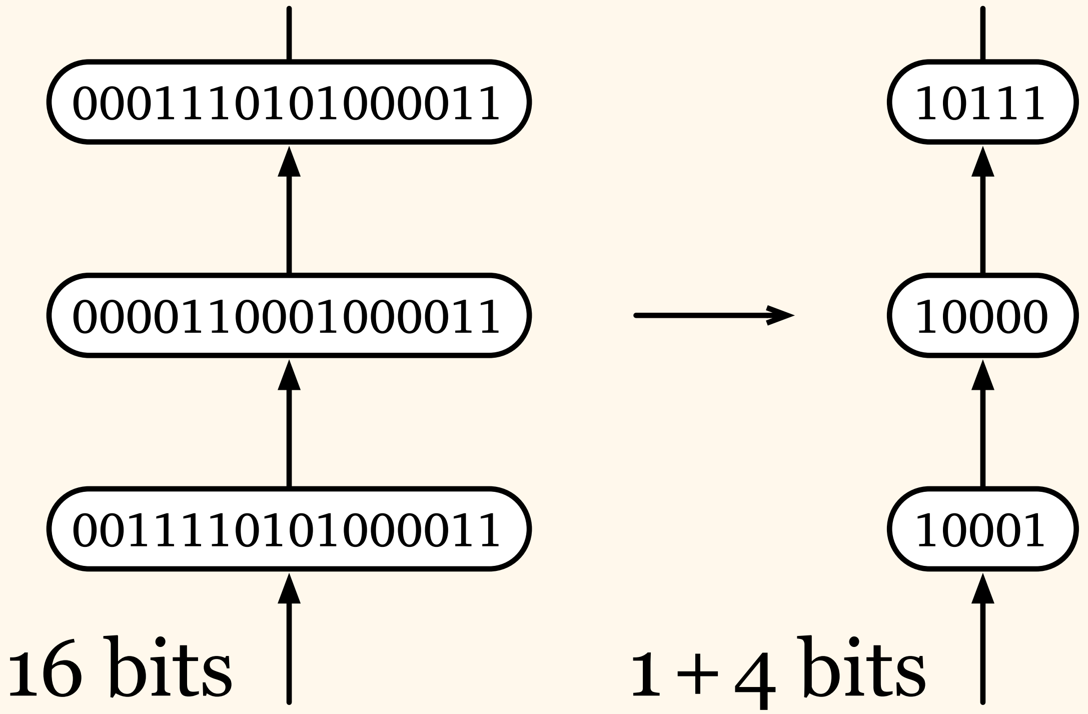
Fast Graph Colouring

- Let's first study a special case…
- *Directed pseudoforest*
  - edges oriented
  - outdegree ≤ 1



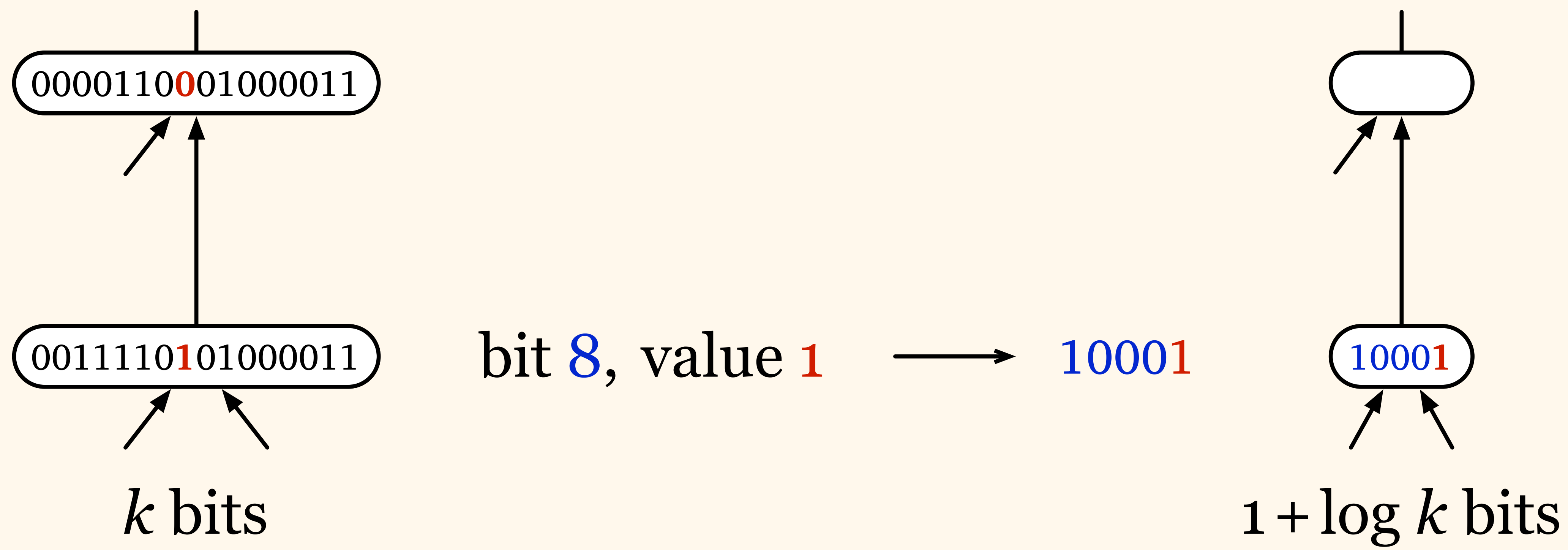
Fast Graph Colouring

- Idea: colour = *binary string*
- Reduce colours:
  - k bits $\to$ $1 + \log_2 k$ bits
  - 2^k colours $\to$ $2k$ colours



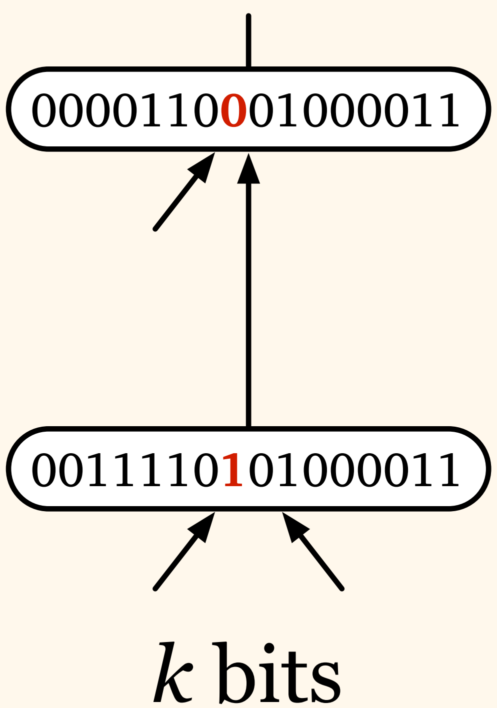
Fast Graph Colouring

- Compare bit string with the successor, find the *first bit that differs*



Fast Graph Colouring

- Correct, no matter what the successor does



…

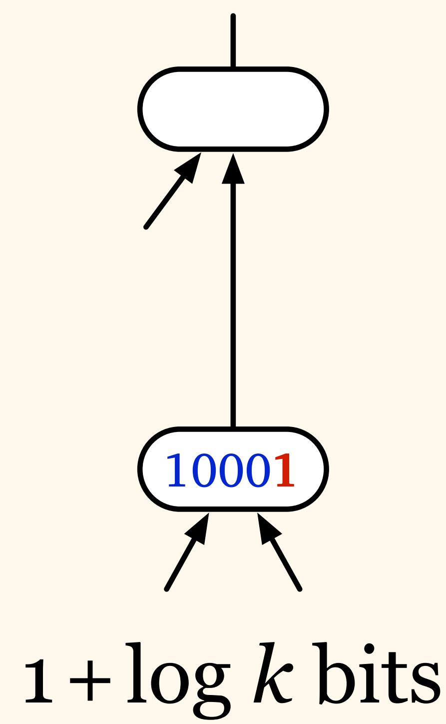
bit 6, value 1 $\longrightarrow$ 01101

bit 7, value 0 $\longrightarrow$ 01110

bit 8, value 0 $\longrightarrow$ 10000

…

bit 8, value 1 $\longrightarrow$ 10001

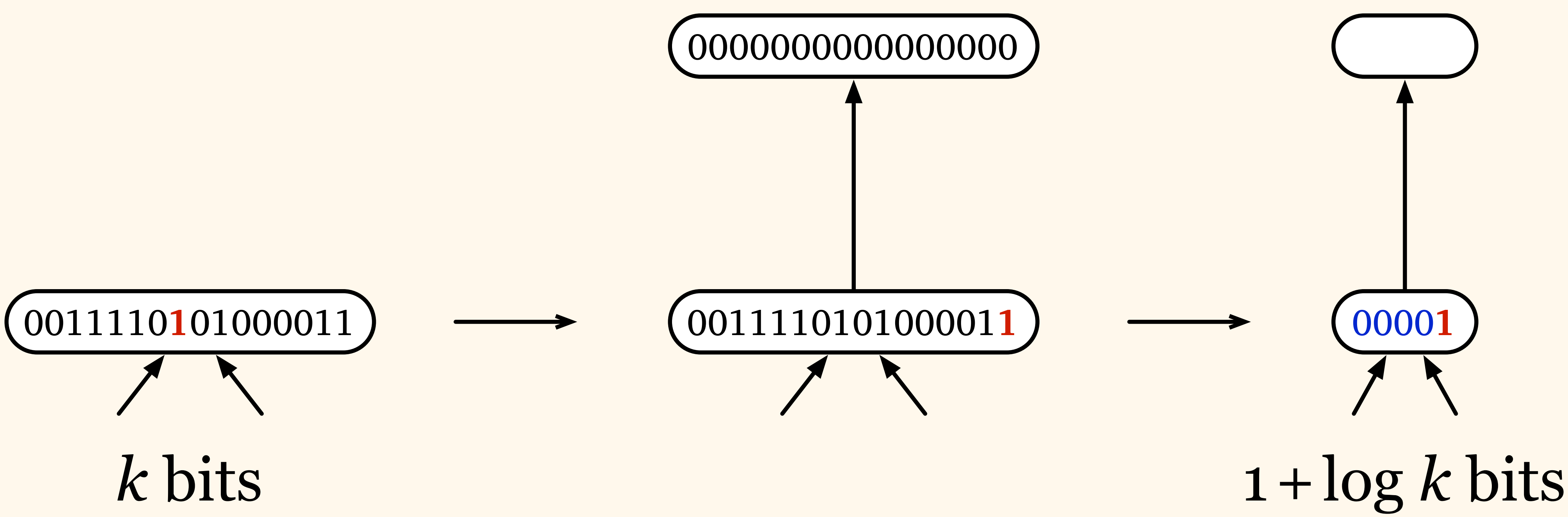


Fast Graph Colouring

- Correct, no matter what the successor does
- For each directed edge (u, v):
  - the new colour of node u is different from the new colour of its successor v
- Proper graph colouring

Fast Graph Colouring

- No successor?
  Pretend that there is one...

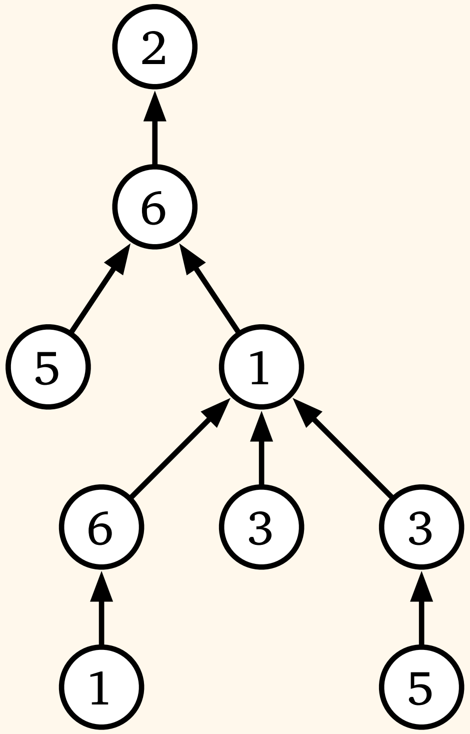


Fast Graph Colouring

- Very fast colour reduction:
  - 2^{128} colours → $2 \cdot 128 = 2^8$ colours
  - 2^8 colours → $2 \cdot 8 = 2^4$ colours
  - 2^4 colours → $2 \cdot 4 = 2^3$ colours
  - 2^3 colours → $2 \cdot 3 = 6$ colours
- But now we are stuck – how to get below 6?

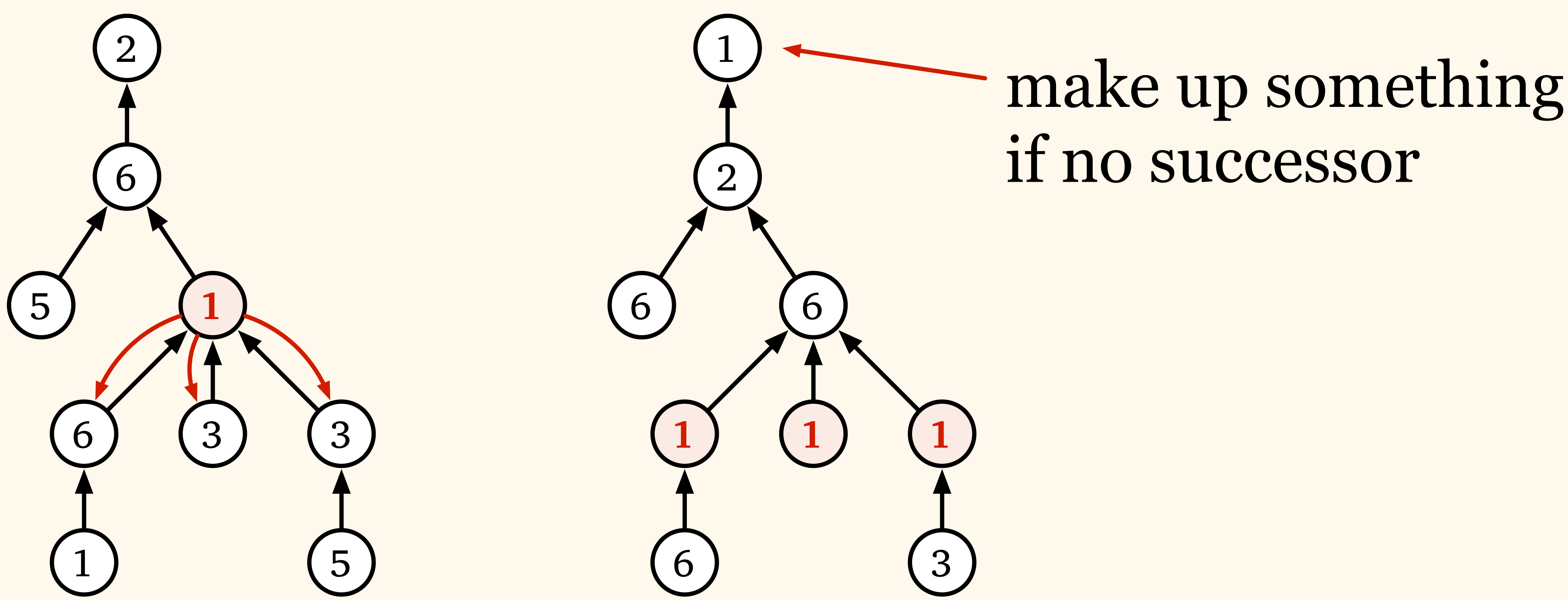
Fast Graph Colouring

- Directed pseudotree with 6 colours: how to reduce the number of colours?



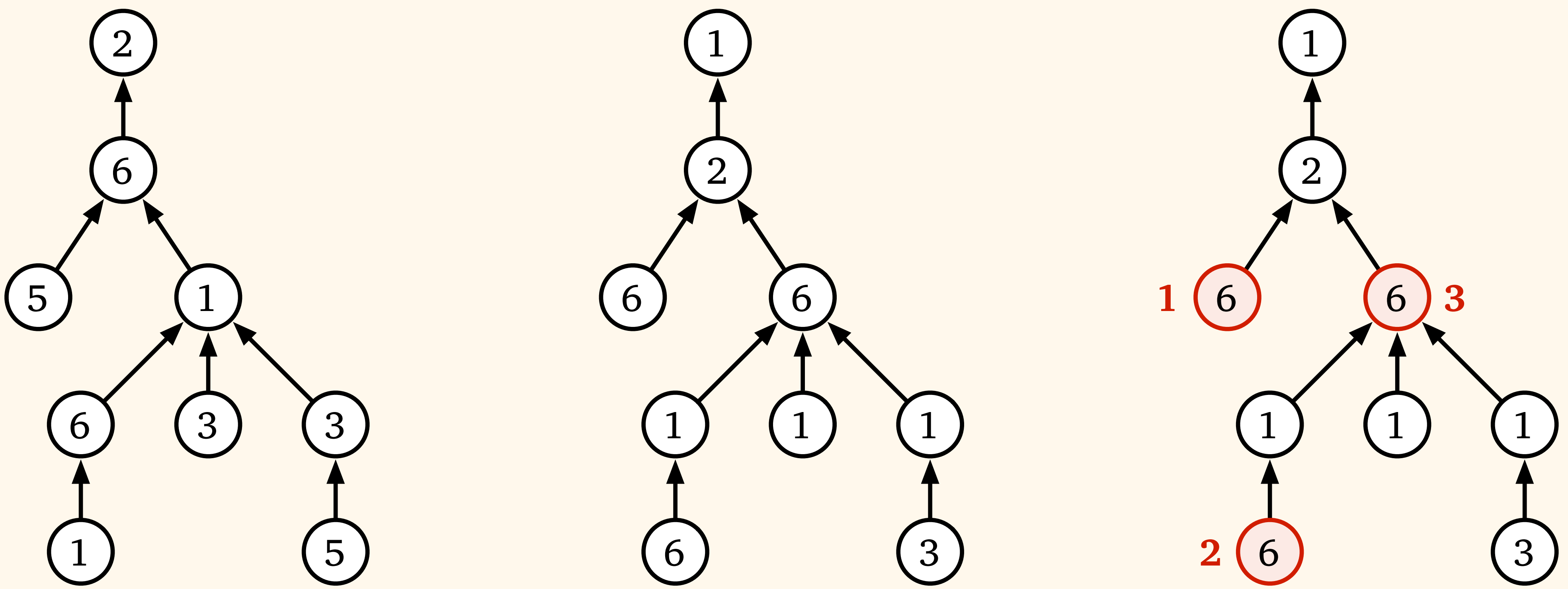
Fast Graph Colouring

- Shift colours "down":
  all predecessors have the same colour



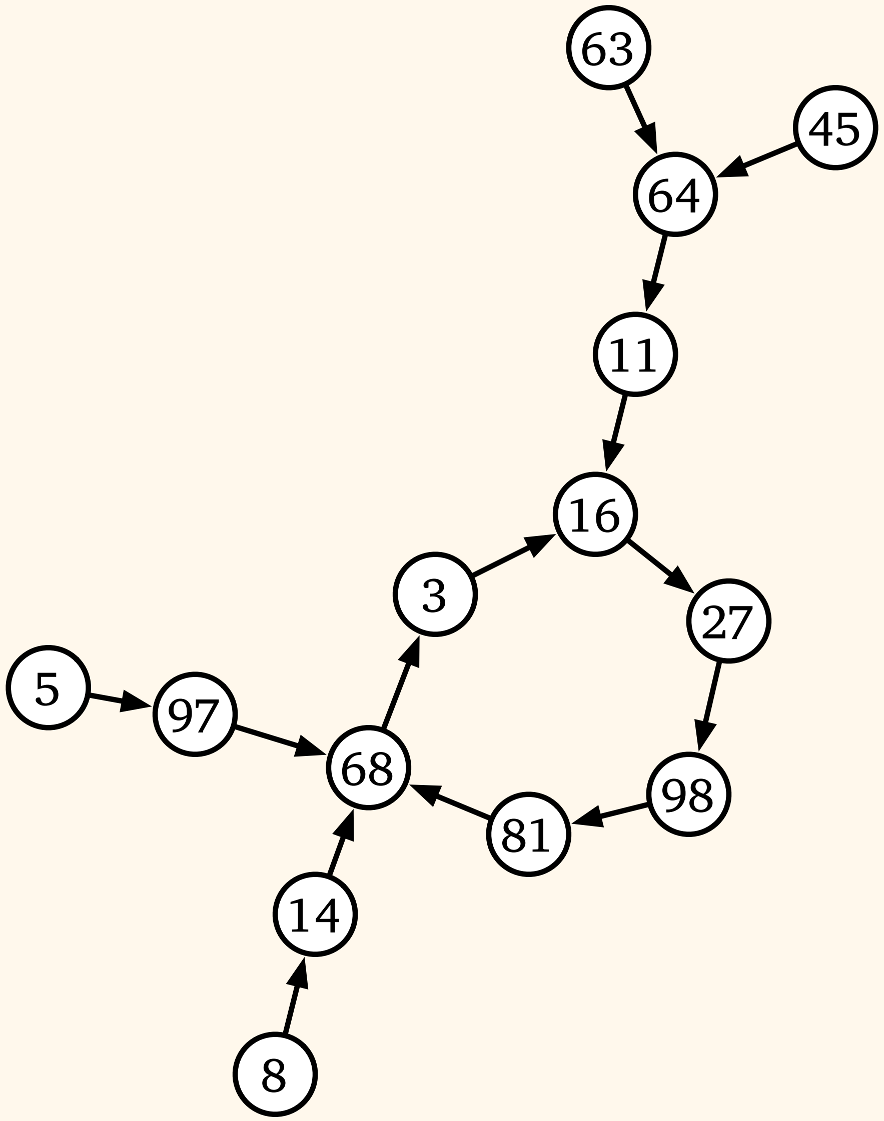
Fast Graph Colouring

- Now greedy works very well:
  there is always a free colour in set {1, 2, 3}



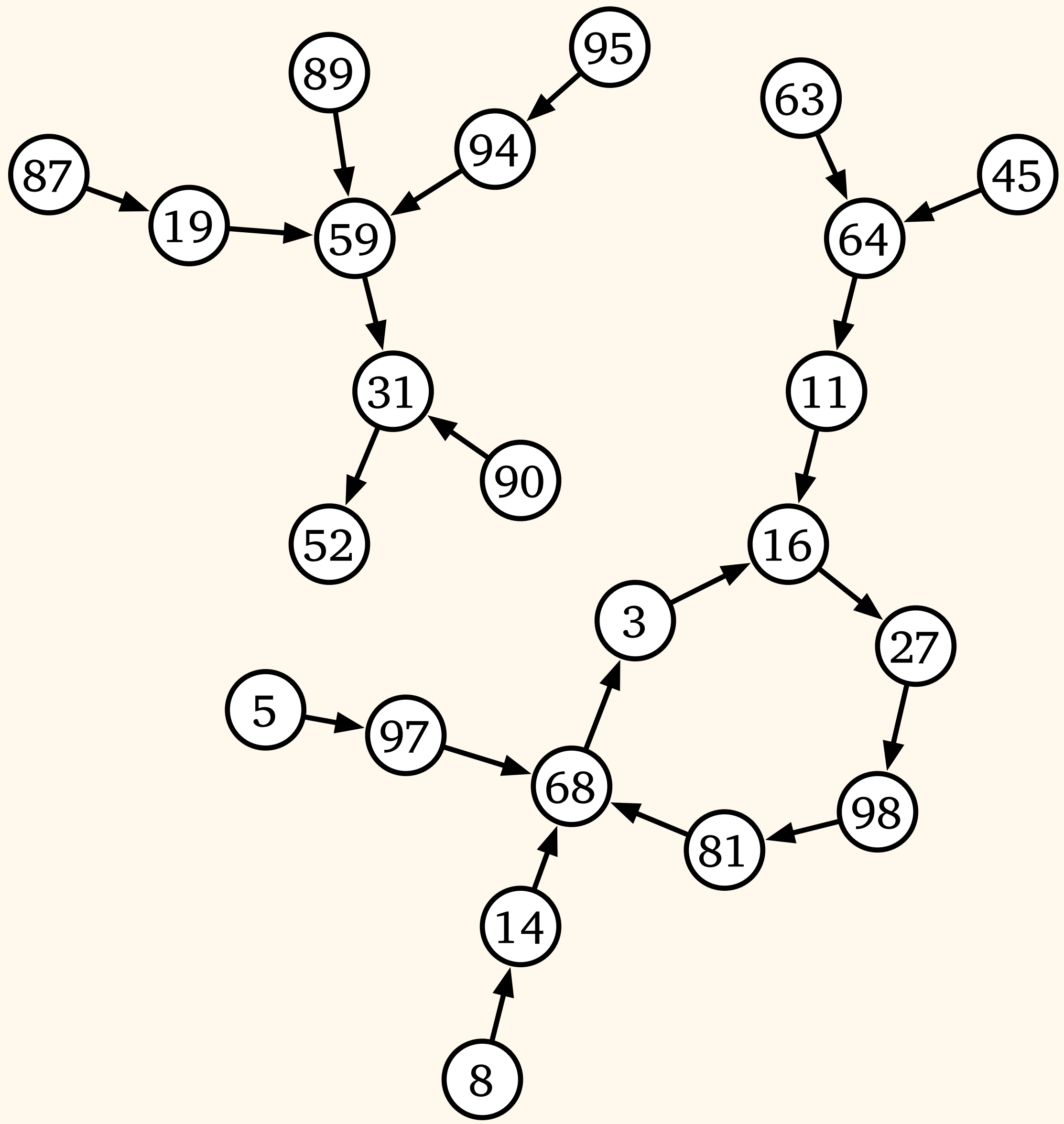
Fast Graph Colouring

- Colour reduction in directed pseudotrees
    - bit comparisons: very quickly from x to 6 colours
    - $2^{128} \rightarrow 2^8 \rightarrow 16 \rightarrow 8 \rightarrow 6$
    - shift + greedy: slowly from 6 to 3 colours
    - $6 \rightarrow 5 \rightarrow 4 \rightarrow 3$

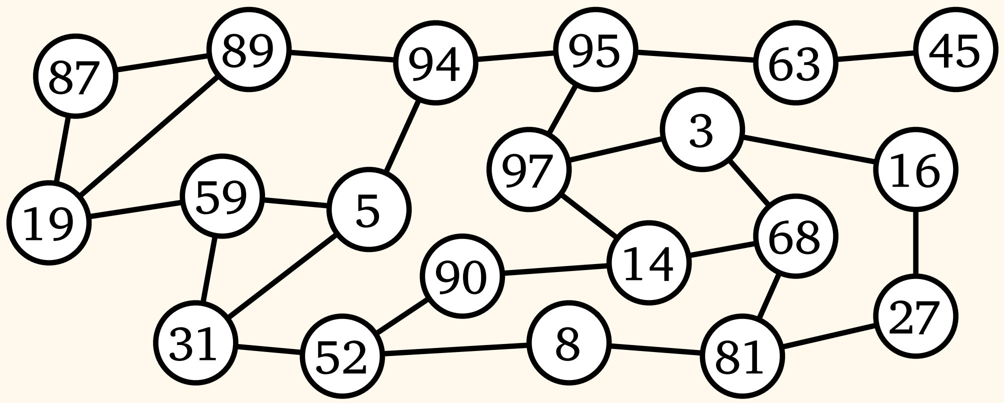


Fast Graph Colouring

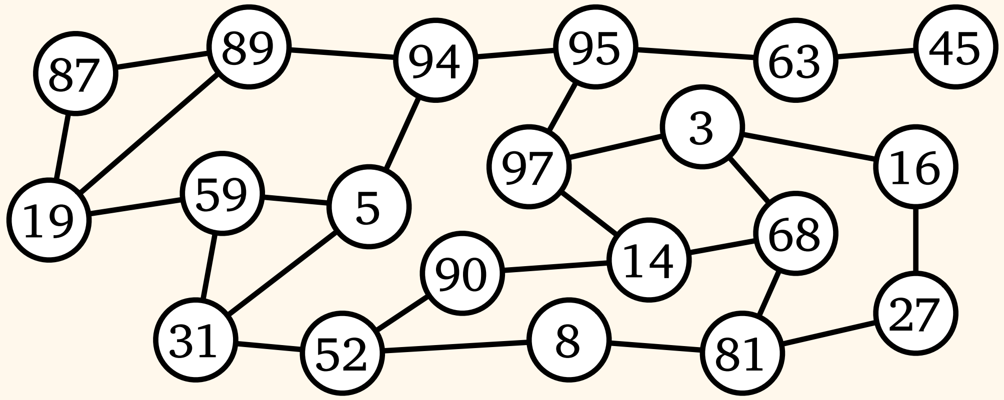
- So far:
    - colour reduction in *directed pseudoforests*
- Next:
    - colour reduction in general graphs of maximum degree Δ



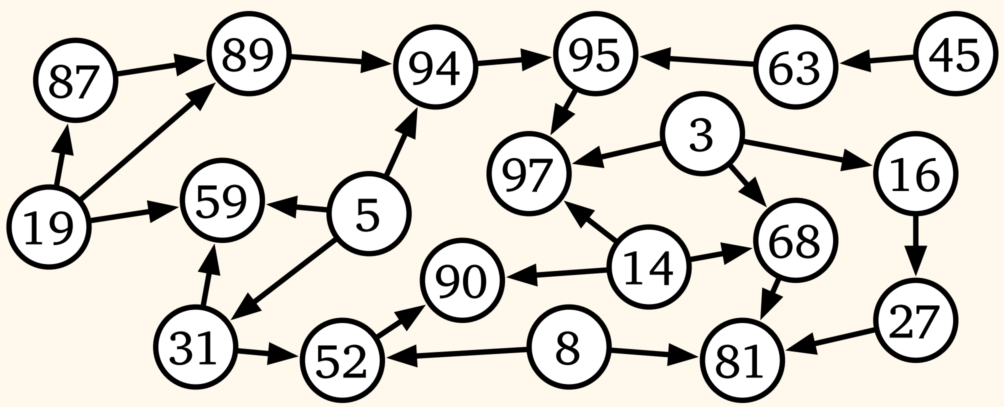
Input:



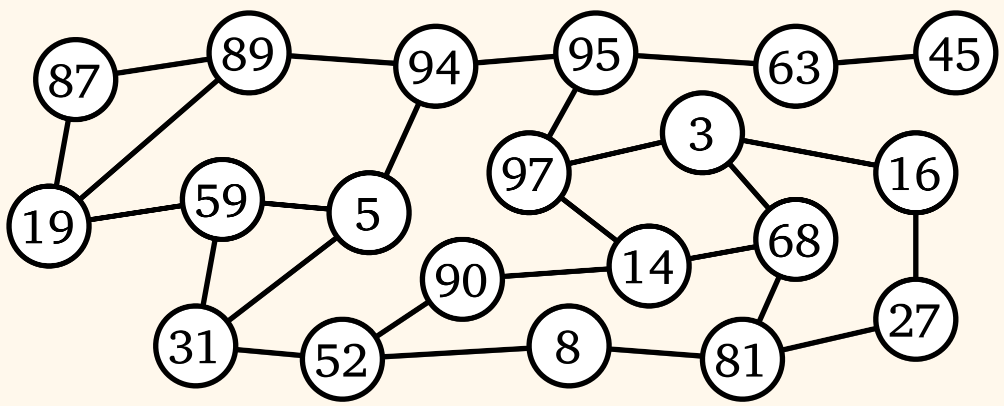
Input:



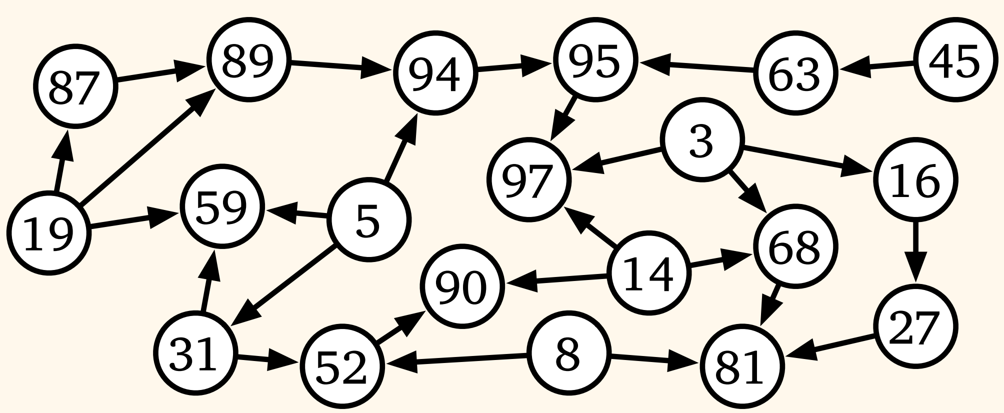
Colours → orientation:



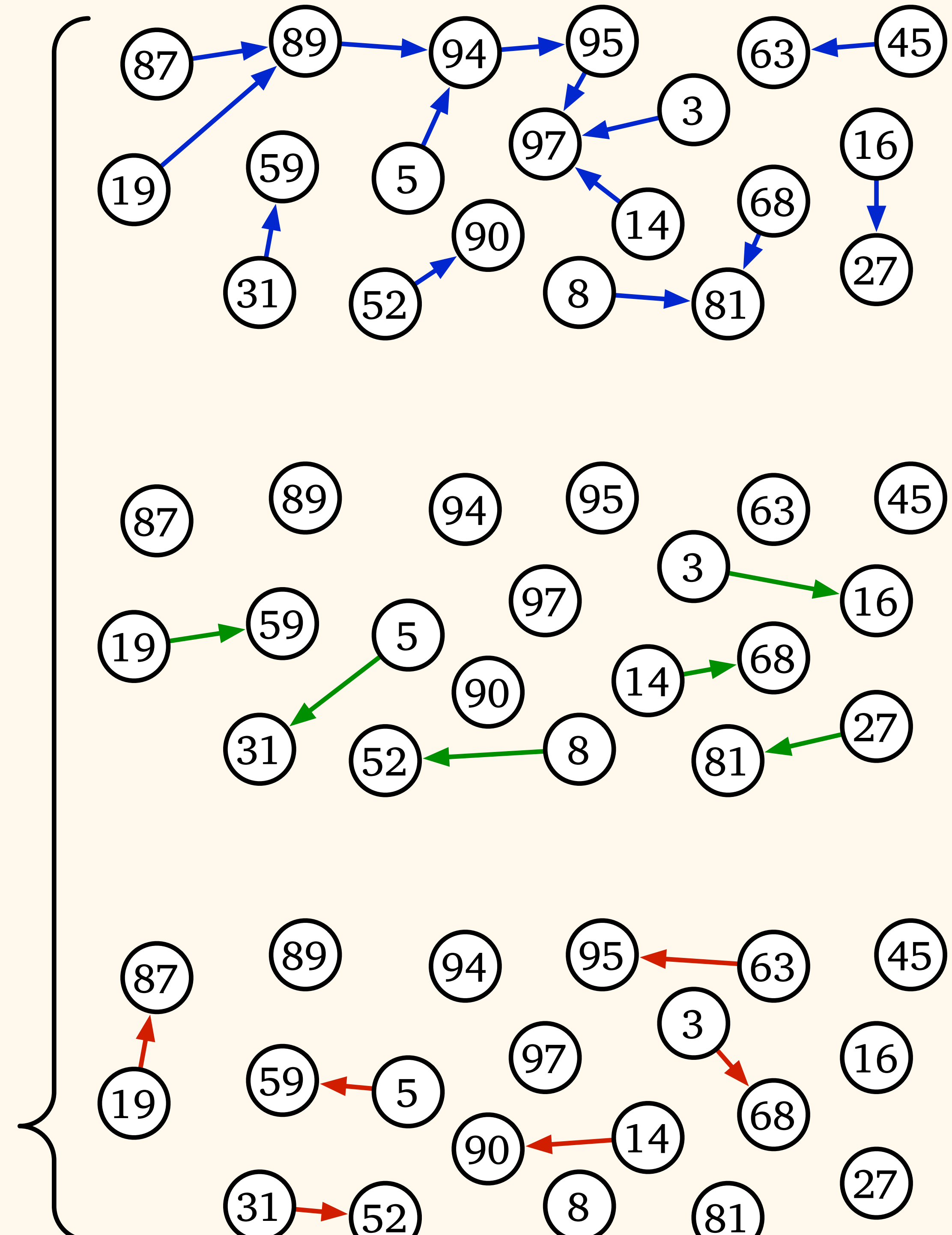
Input:



Colours → orientation:



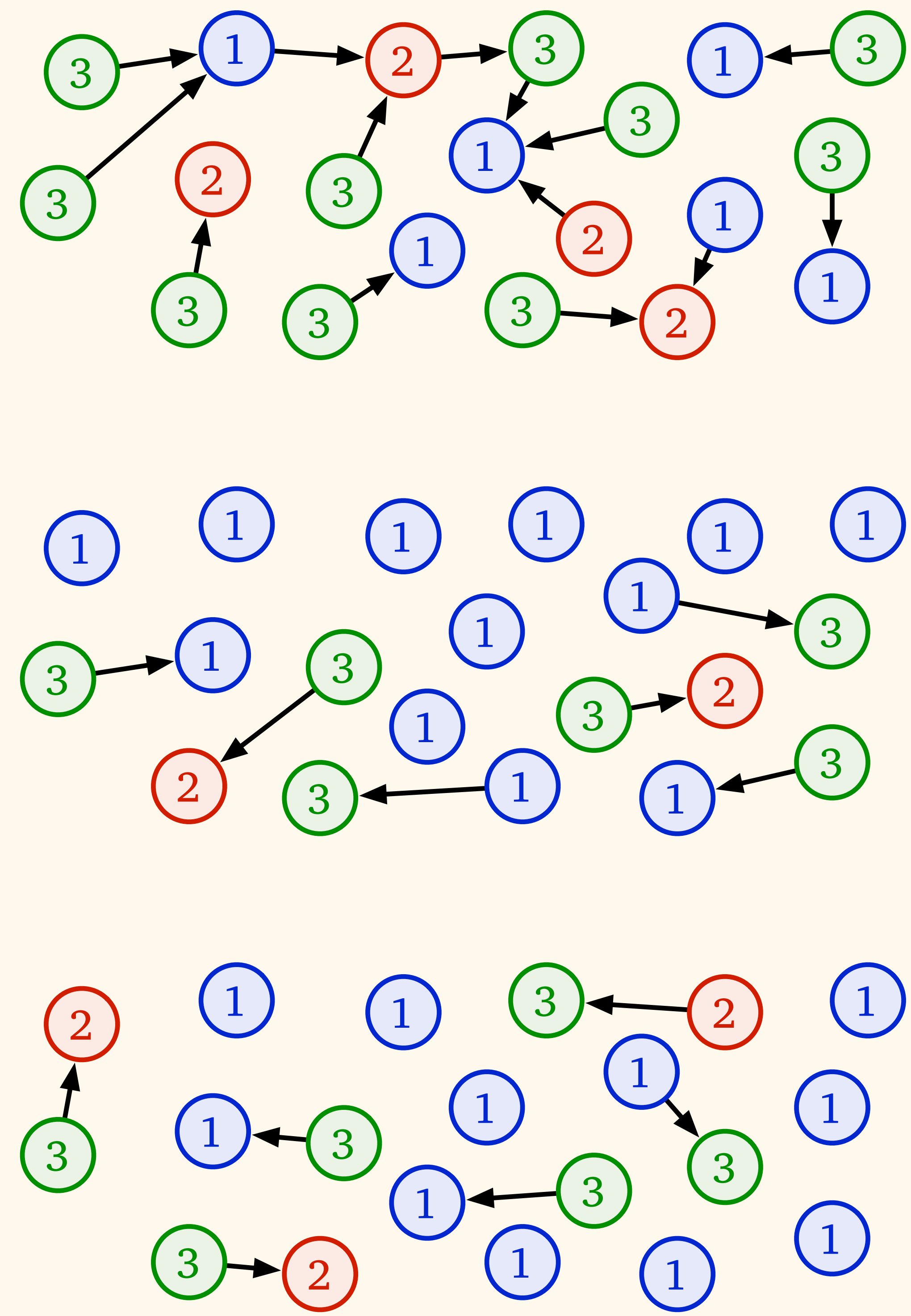
Port numbers → partition
in Δ directed pseudoforests

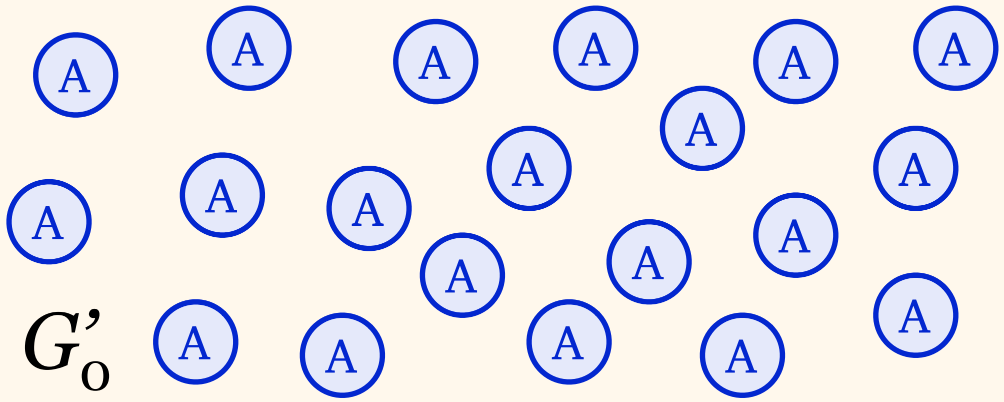


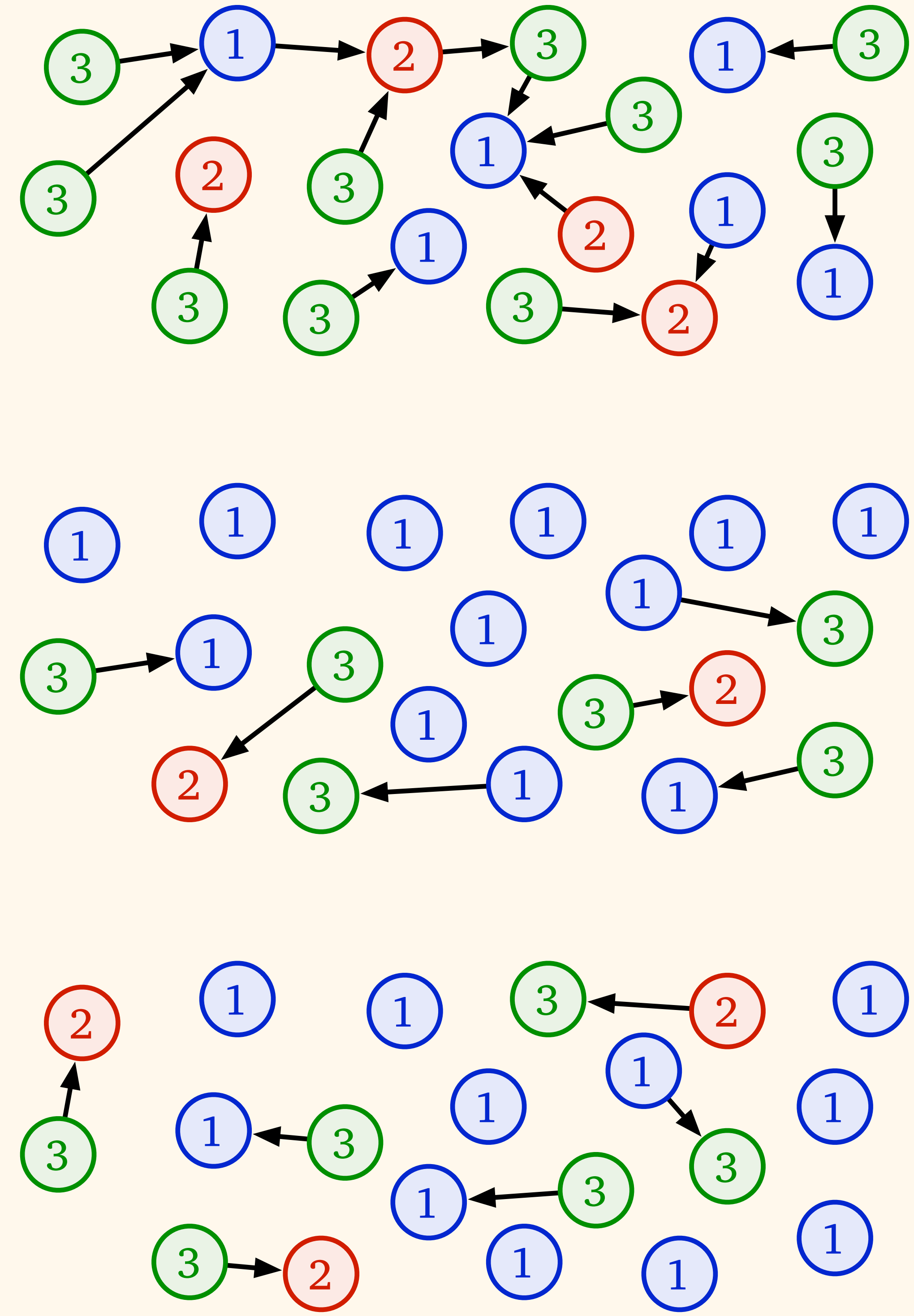
Find a 3-colouring
for each pseudoforest

Computed in parallel,
simulate Δ instances of
the algorithm

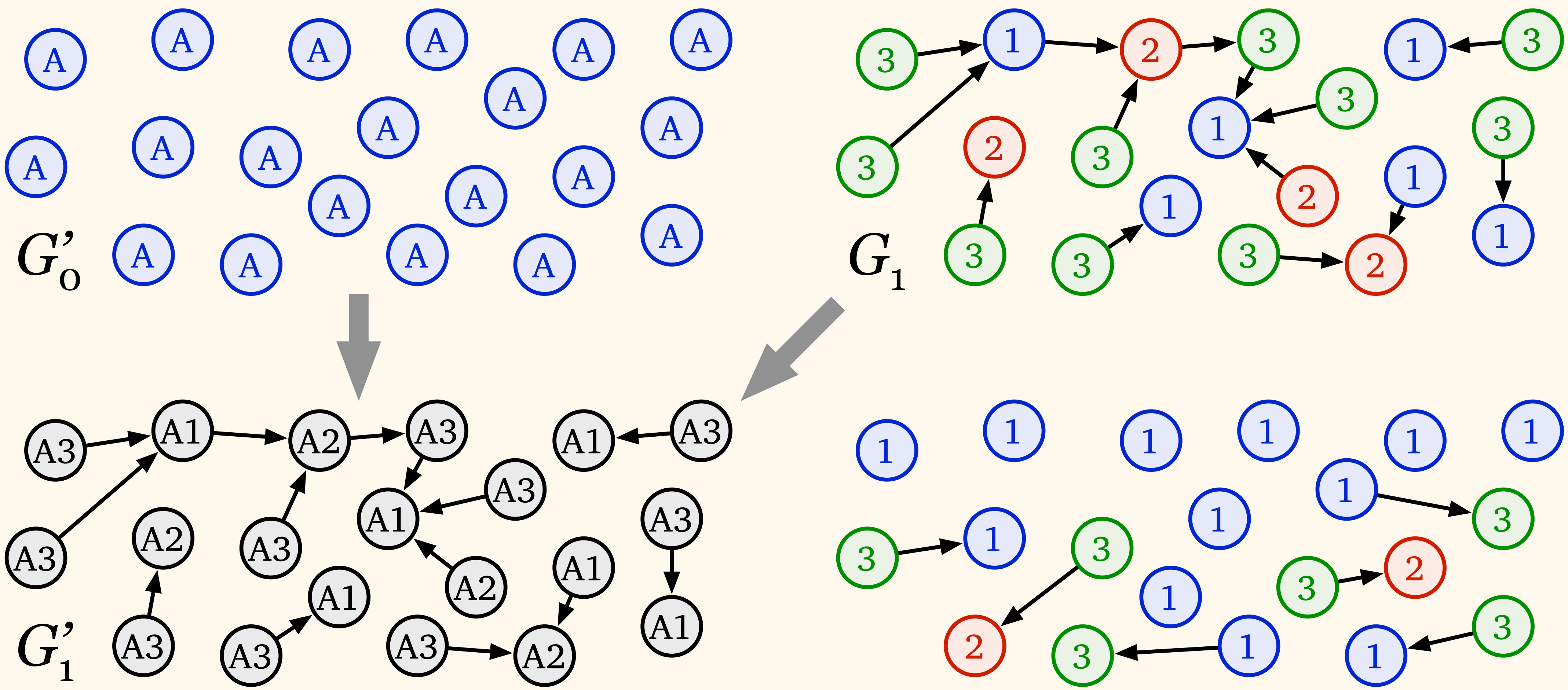
Each node has Δ colours,
one for each forest



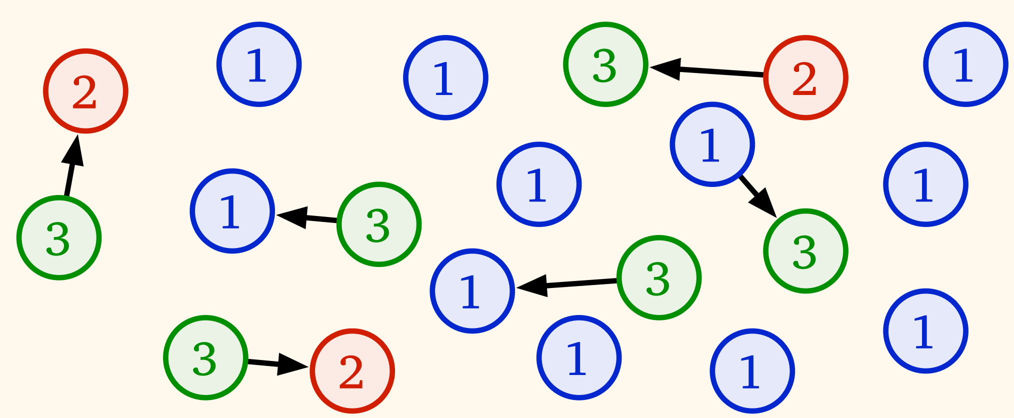




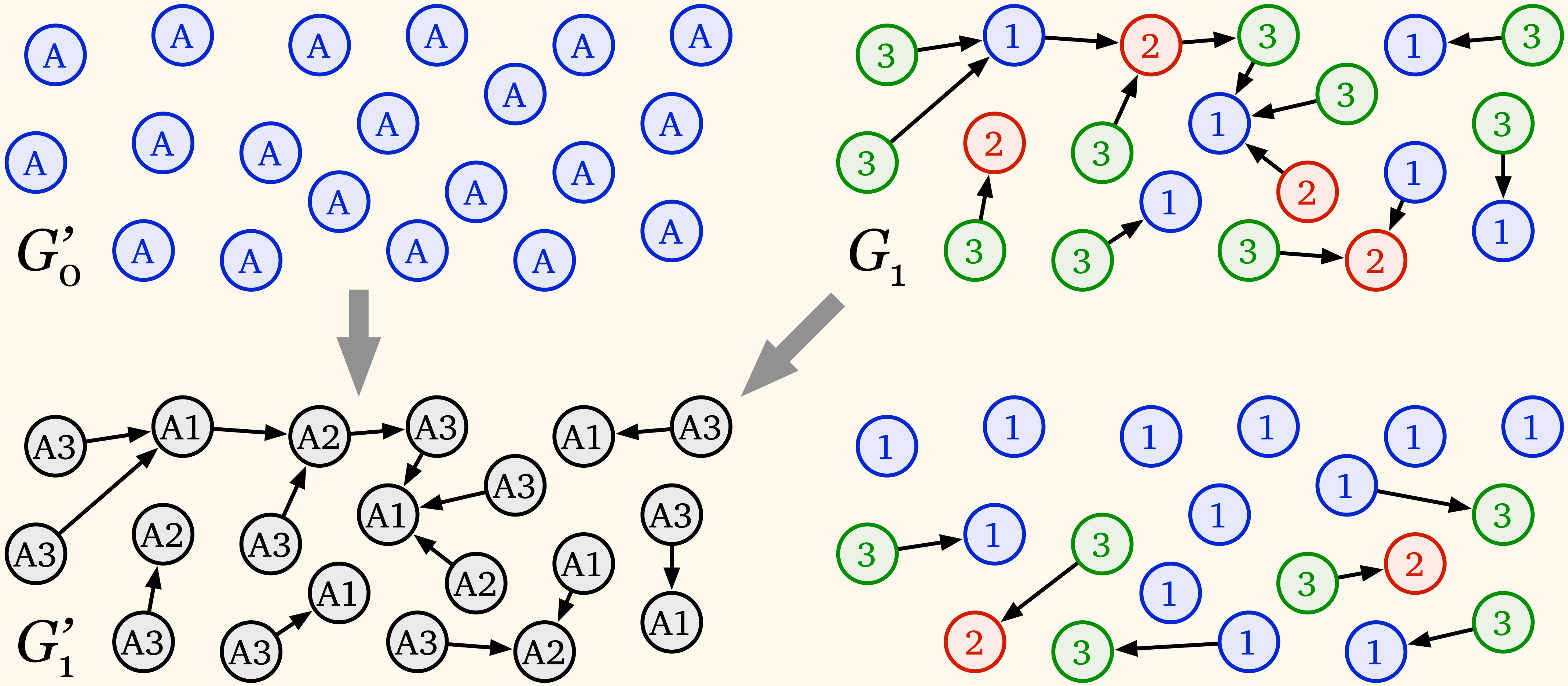
G'_0: $(\Delta+1)$-coloured
– trivial, no edges



union of edges,
combination of colours



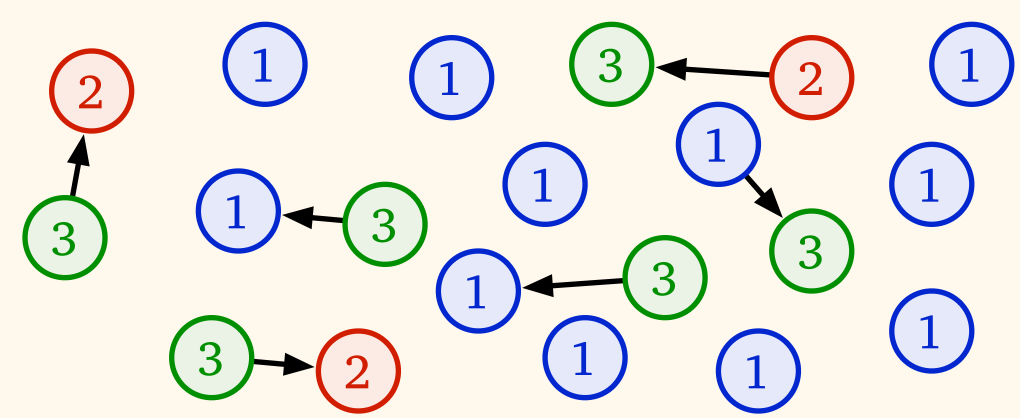
$a + b \to (a, b)$

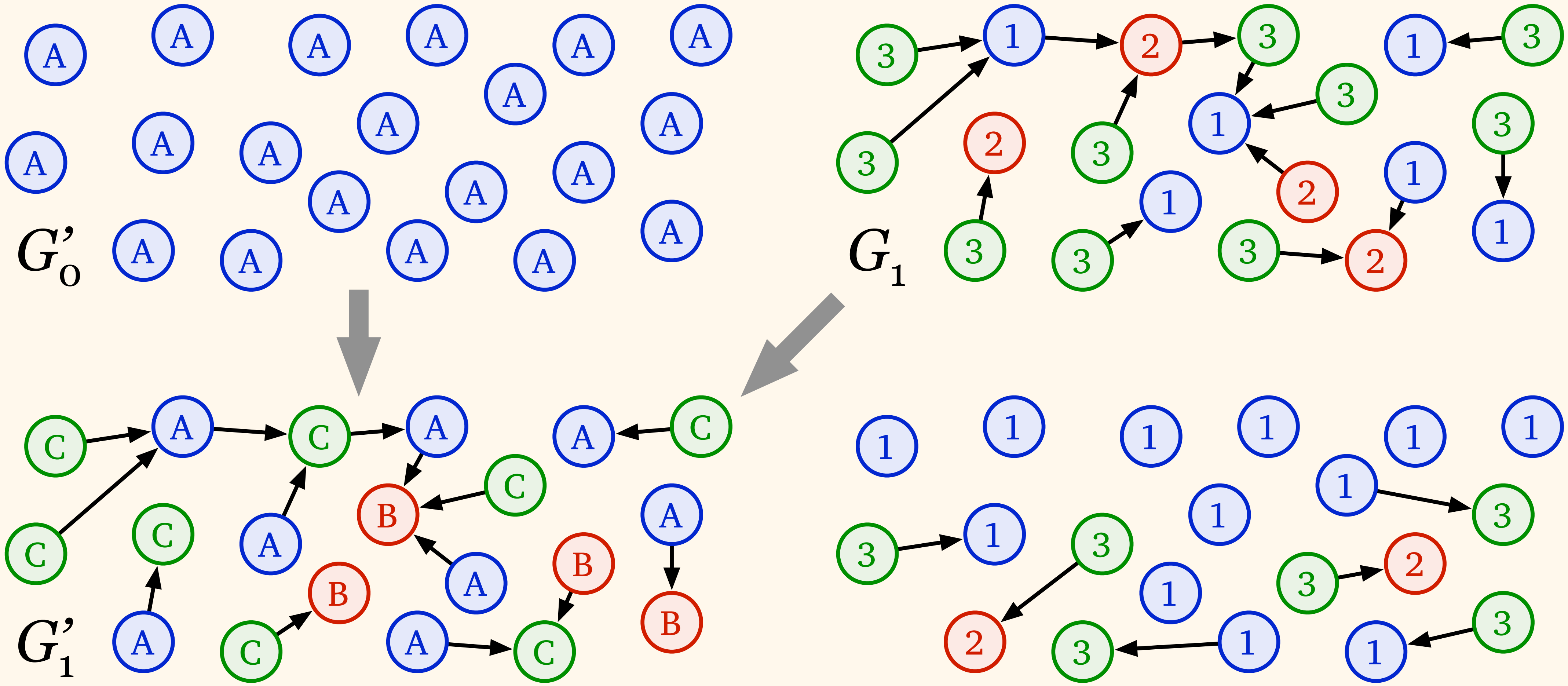


G'_0: $(\Delta+1)$-coloured

G_1: 3-coloured

G'_1: $3(\Delta+1)$-coloured



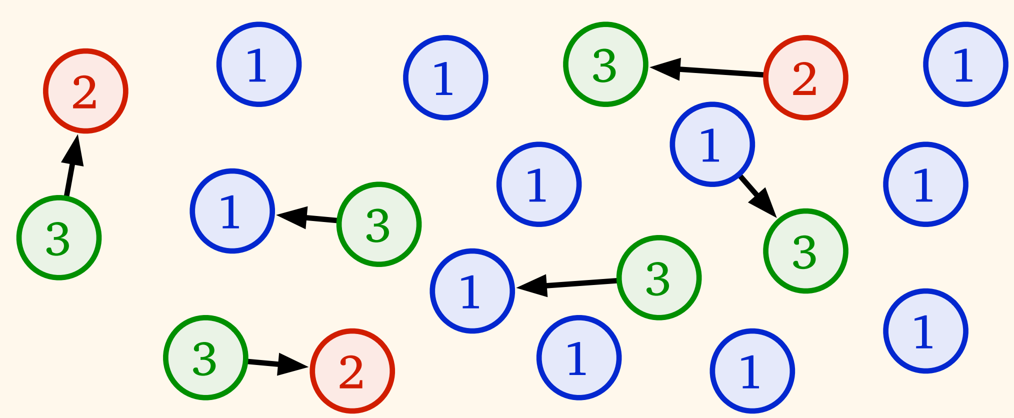


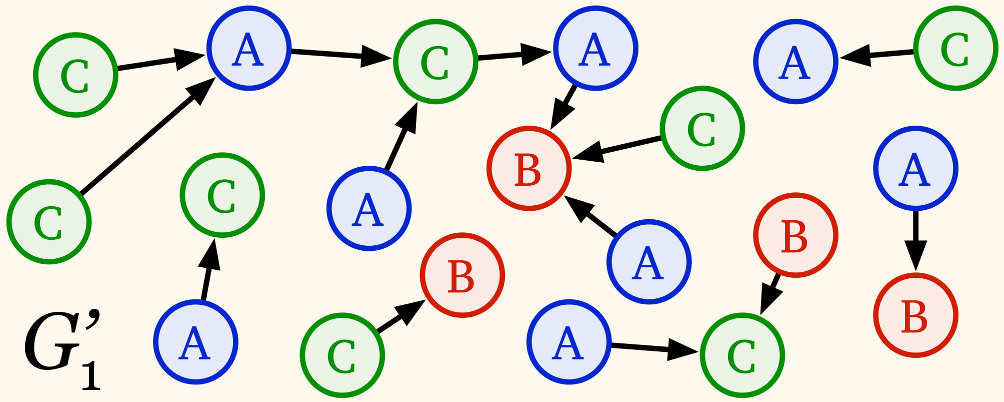
G'_0: $(\Delta+1)$-coloured

G_1: 3-coloured

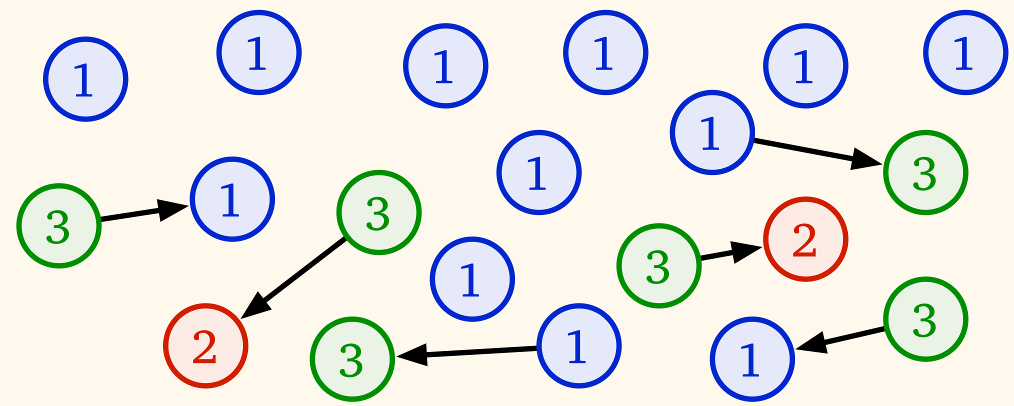
G'_1: $3(\Delta+1)$-coloured,

reduce to $\Delta+1$ greedily

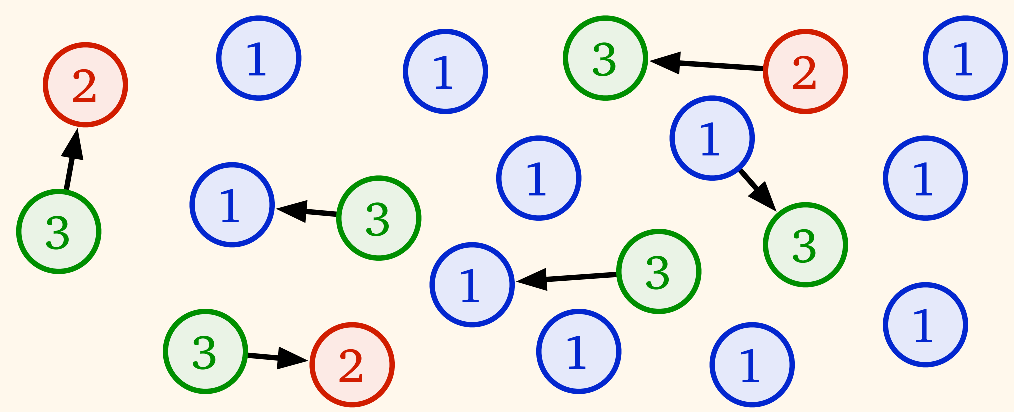


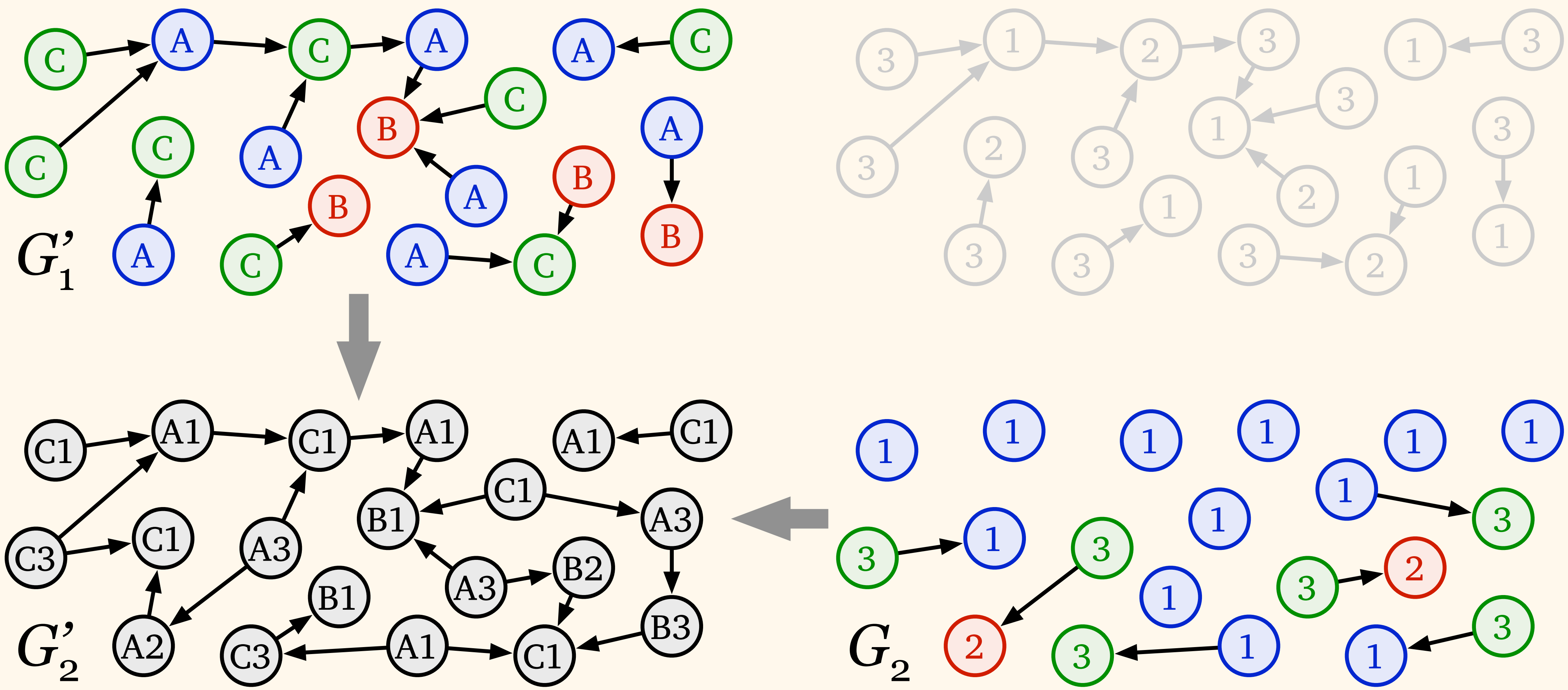


G'_1



G'_1: (Δ+1)-coloured

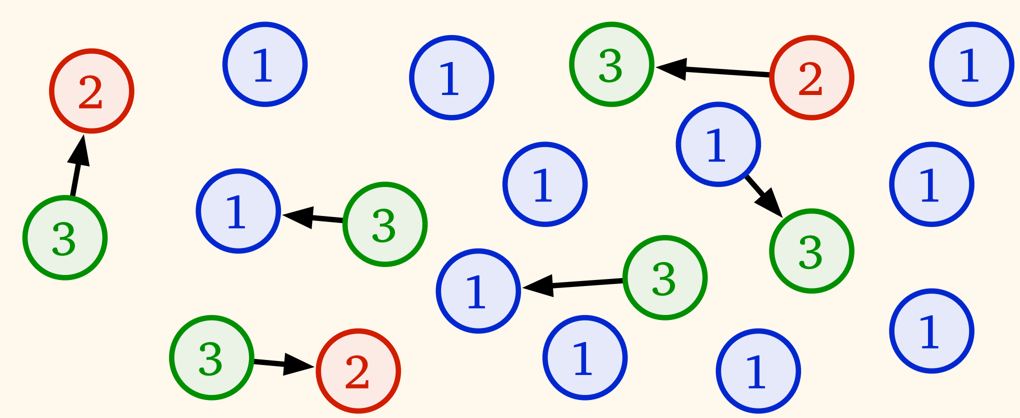


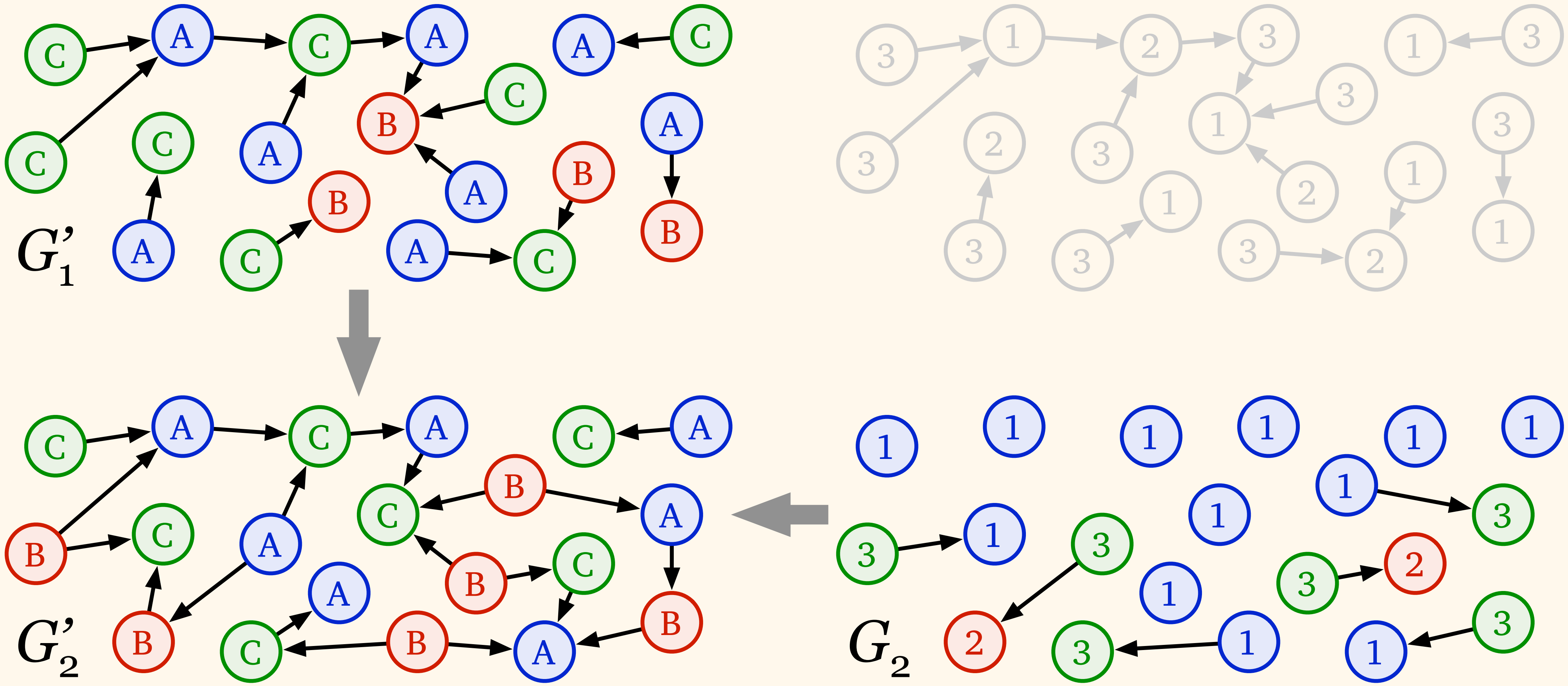


G'_1: $(\Delta+1)$-coloured

G_2: 3-coloured

G'_2: $3(\Delta+1)$-coloured



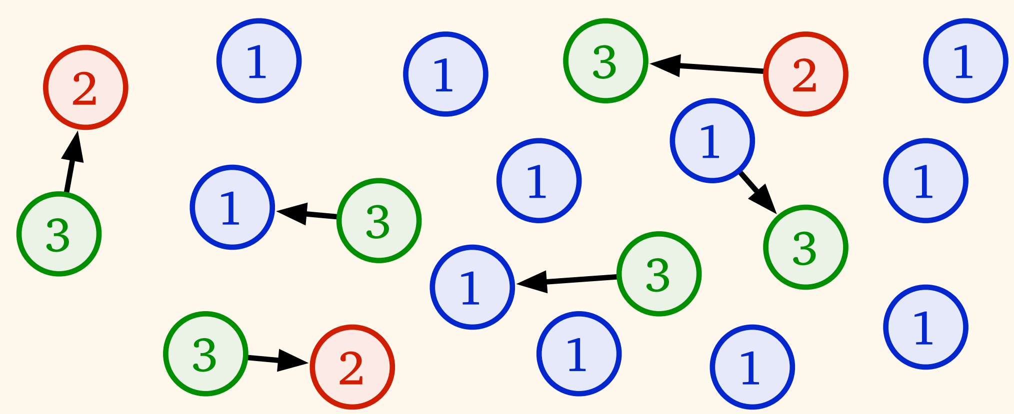


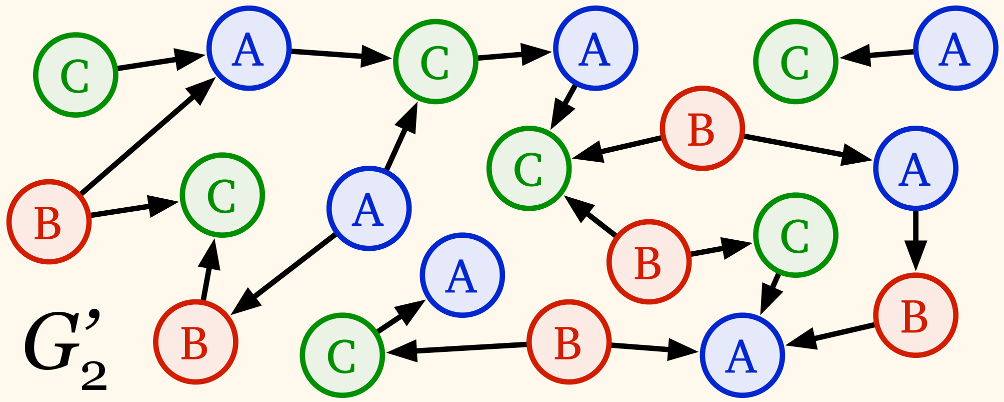
G'_1: $(\Delta+1)$-coloured

G_2: 3-coloured

G'_2: $3(\Delta+1)$-coloured,

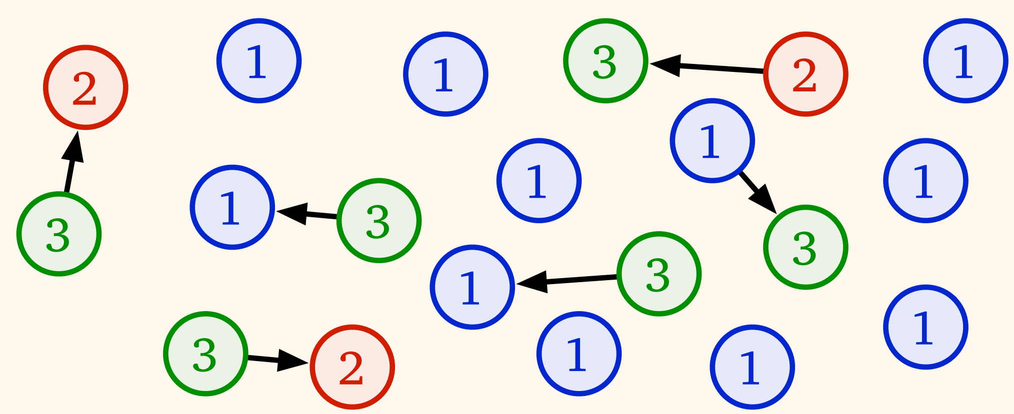
reduce to $\Delta+1$ greedily

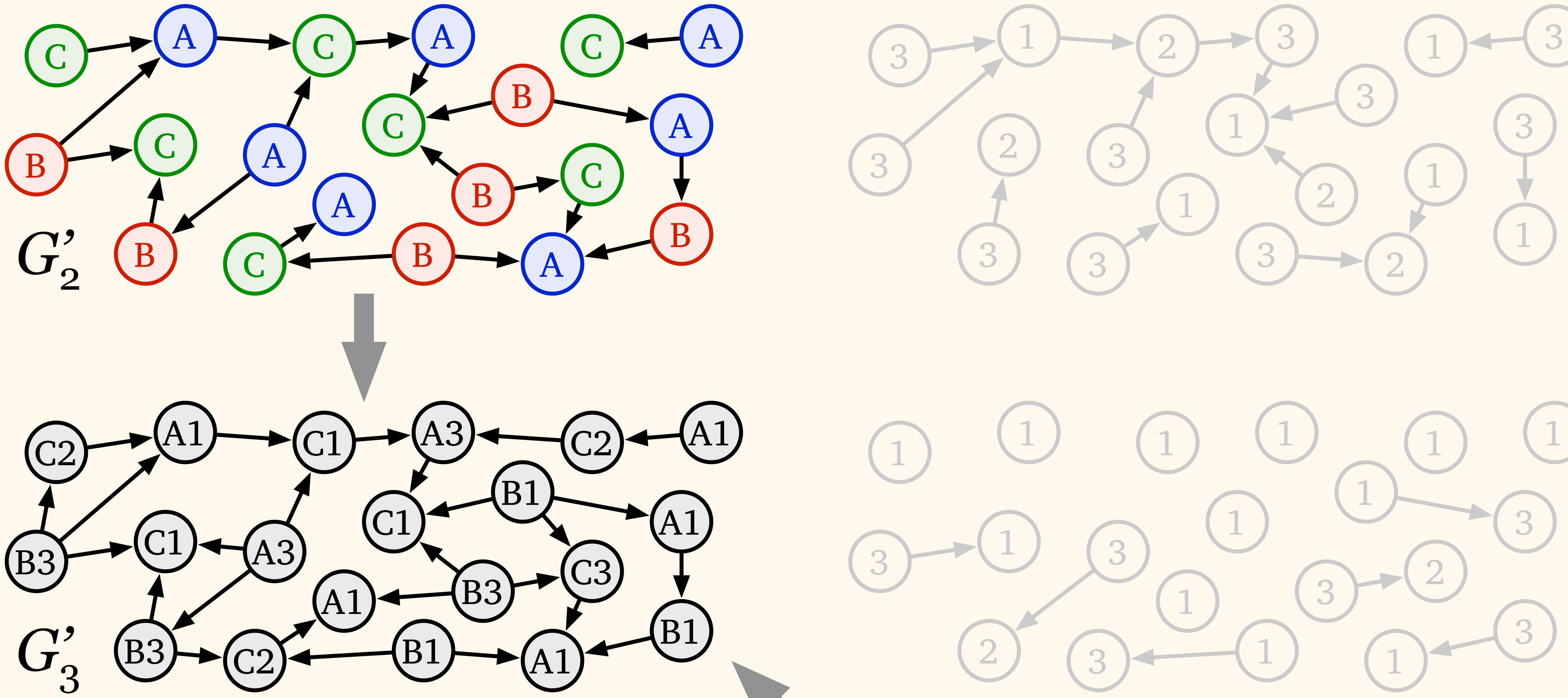




G_2'

G_2': $(\Delta+1)$-coloured

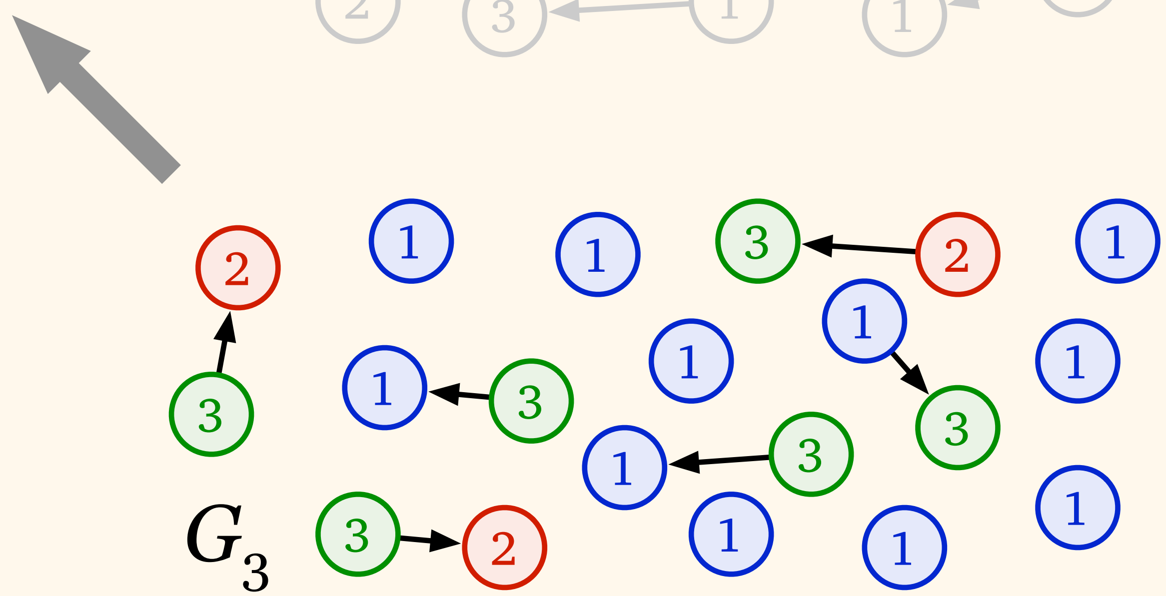


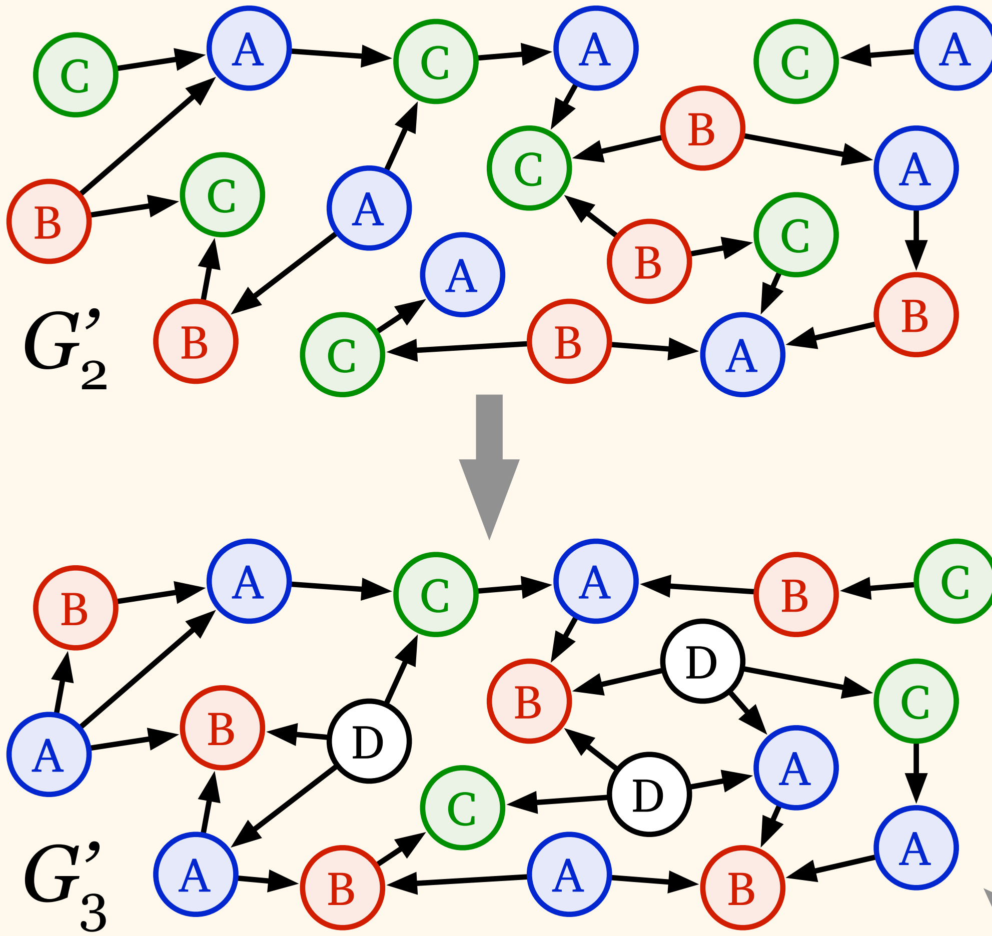


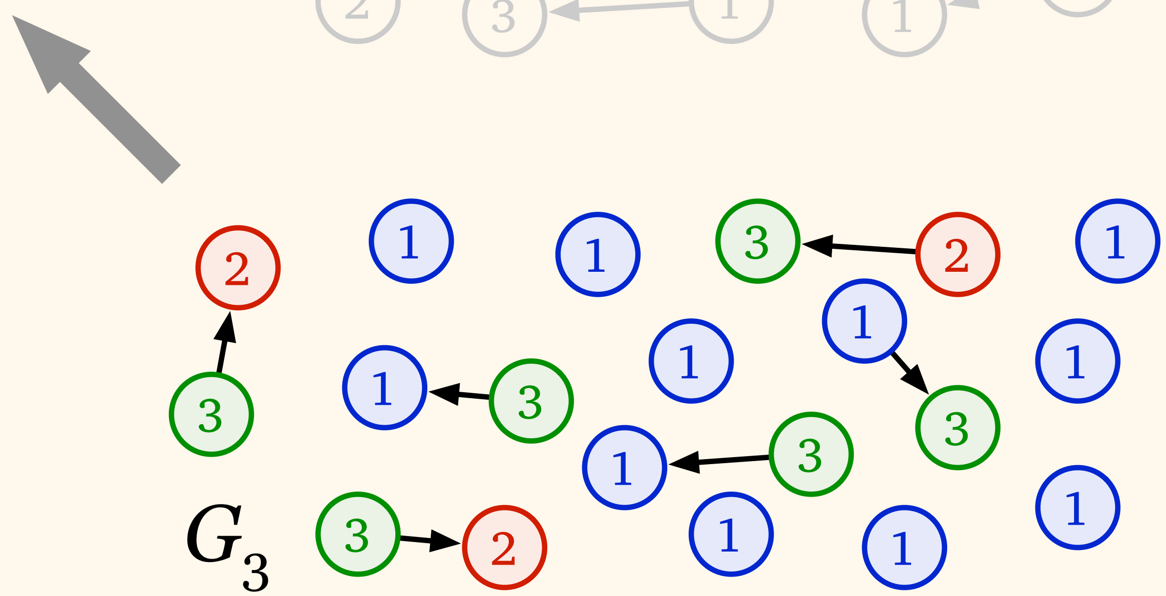
G'_2: $(\Delta+1)$-coloured

G_3: 3-coloured

G'_3: $3(\Delta+1)$-coloured





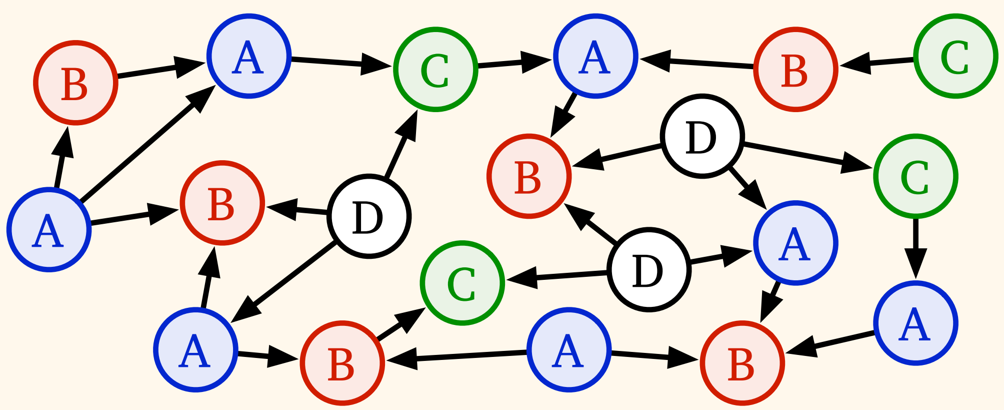


G'_2: $(\Delta+1)$-coloured

G_3: 3-coloured

G'_3: $3(\Delta+1)$-coloured,

reduce to $\Delta+1$ greedily



$(\Delta+1)$-colouring of the original graph

Fast Graph Colouring

- Colour reduction from x to $\Delta+1$
  - orientation: 1 round
  - partition: 0 rounds
  - 3-colouring: $O(\log^* x)$ rounds — see Exercise 5.4
  - Δ phases:
    - merge & reduce $3(\Delta+1) \rightarrow \Delta+1$: $2(\Delta+1)$ rounds
  - **total**: $O(\Delta^2 + \log^* x)$ rounds

Fast Graph Colouring

- Colour reduction from x to $\Delta+1$
  - $O(\Delta^2 + \log^* x)$ rounds
- Plenty of applications — see exercises
- Similar techniques can be used to solve other problems

Fast Graph Colouring

- Colour reduction from x to $\Delta+1$
  - $O(\Delta^2 + \log^* x)$ rounds
- Fast, but running time depends on x
- Next week:
  - dependence on x is necessary
  - even if $\Delta = 2$, we cannot reduce the number of colours from x to 3 in constant time, independently of x