New Lower Bounds for Distributed Graph Algorithms

Jukka Suomela
Helsinki Institute for Information Technology HIIT
Department of Computer Science, University of Helsinki

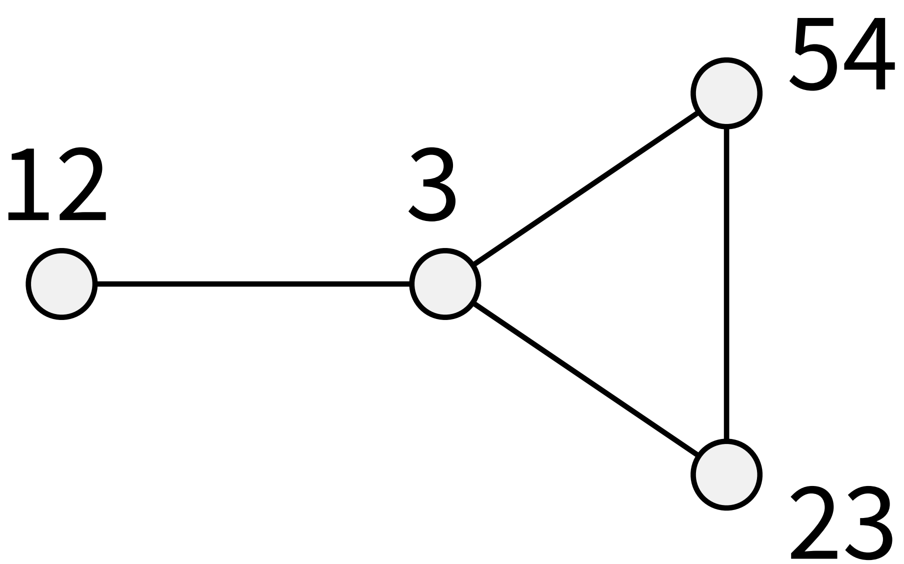
Estonian CS Theory Days
Saka manor, 27 October 2013

Distributed Graph Algorithm

– mapping from
local neighbourhoods
to local outputs

Distributed Graph Algorithm

- **Input:** simple undirected **graph *G***
  - nodes labelled with unique $O(\log n)$-bit identifiers



Distributed Graph Algorithm

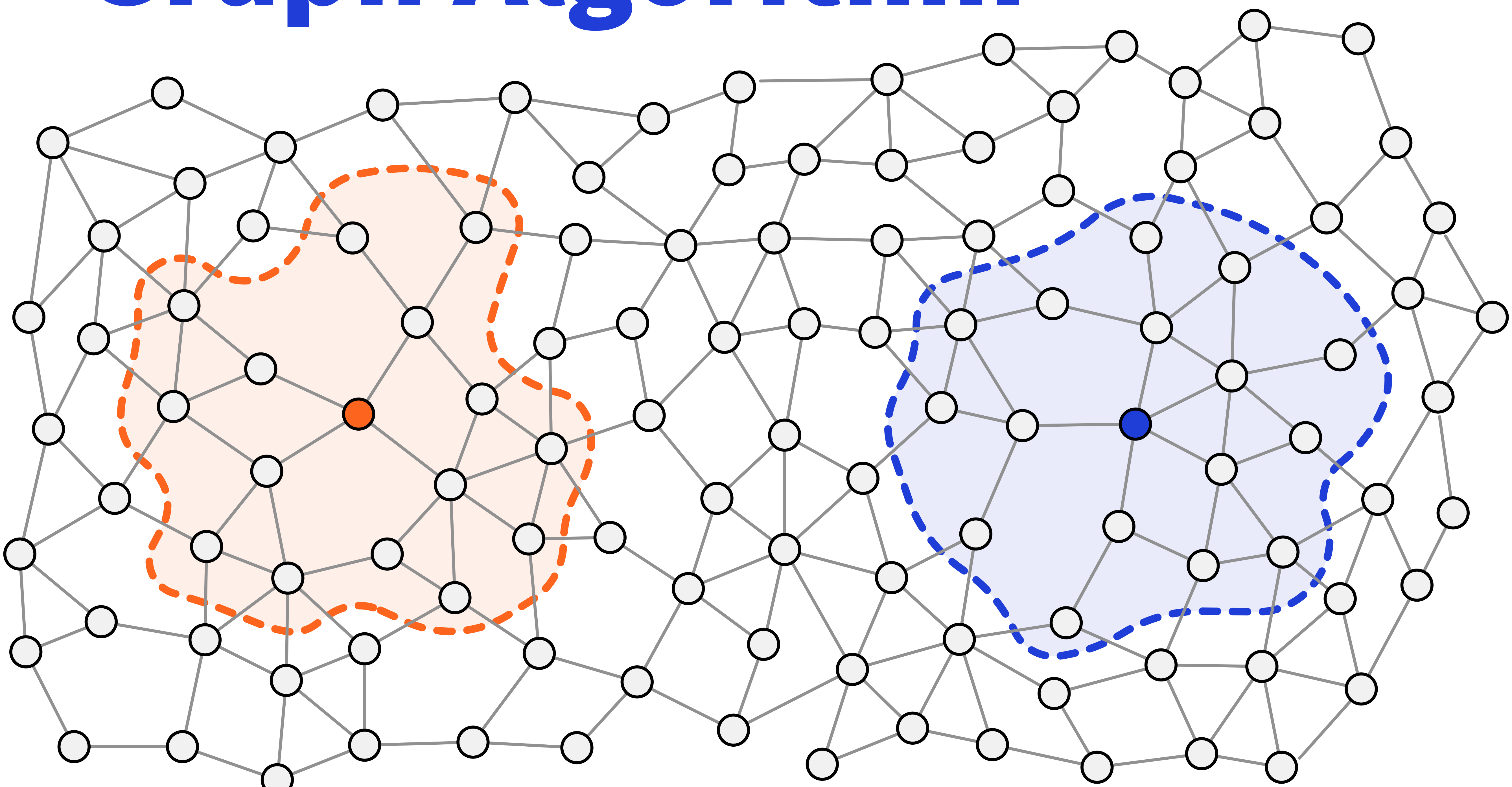
- **Input:** simple undirected graph G
- **Output:** $A(G, v)$ = **local output** of node v
  - graph colouring: $A(G, v)$ = colour of node v
  - vertex cover: $A(G, v) = 1$ if v is in the cover

Distributed Graph Algorithm

- **Input:** simple undirected graph G
- **Output:** $A(G, v)$ = local output of node v
- **Running time is *t*:** local output $A(G, v)$ only depends on **radius-t neighbourhood** of v
  - shortest-path distance at most t

Distributed Graph Algorithm

time $t = 2$



Distributed Graph Algorithm

- **Mapping from local neighbourhoods to local outputs**
  - each node acts based on its radius-t neighbourhood
  - local outputs must form a globally consistent solution

Distributed Graph Algorithm

- **If *G* is a computer network:**
  - time = **number of communication rounds**
  - in *t* communication steps all nodes can learn everything up to distance *t*
- **Fast distributed algorithm = few communication rounds**

Distributed Graph Algorithm

- **Everything trivial in time diam(G)**
  - all nodes see whole G, can compute any function of G
- **What can be solved much faster?**

Our Focus:
Local Algorithms

- **Distributed graph algorithms with running time $O(1)$**
  - extreme limits of distributed algorithms
  - ideal algorithms for large-scale networks
  - fast and fault-tolerant
- **Do these even exist?**

Our Focus: **Local Algorithms**

- **Distributed graph algorithms with running time $O(1)$**
- **We focus on bounded-degree graphs:**
  - n nodes, maximum degree $\Delta = O(1)$
  - running time $t = f(\Delta)$, independently of n

Our Focus: **Local Algorithms**

- **Surprise: many graph problems can be approximated with local algorithms**
  - 2-approximation of minimum vertex cover
  - many linear programming relaxations
  - many problems on bipartite graphs …
- **Today’s main topic: lower bounds**

Lower Bounds

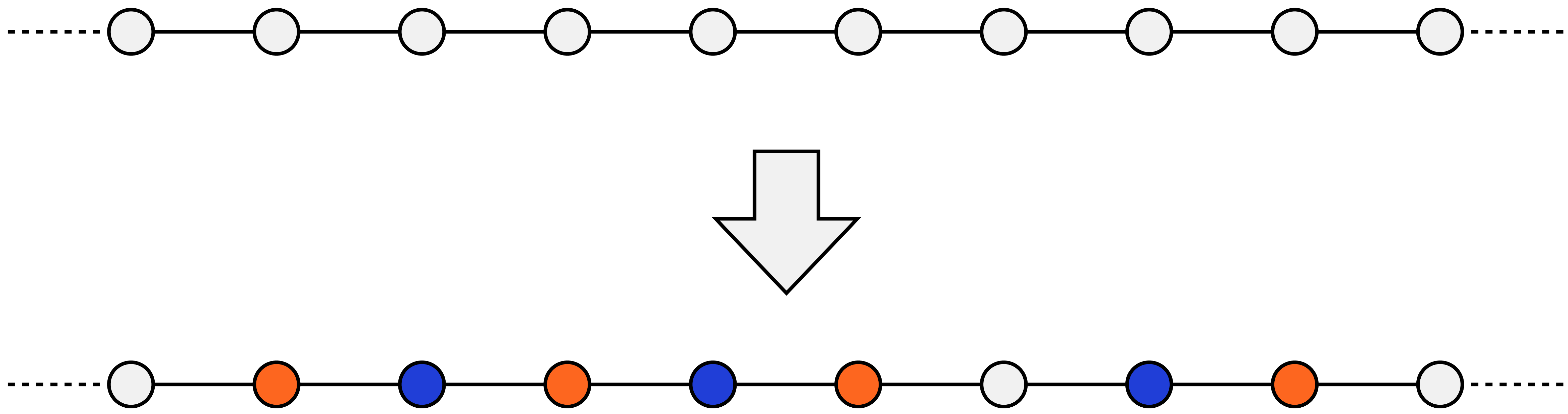
– what cannot
be solved locally?

What Cannot Be Solved Locally?

- **Negative results, even if $\Delta = 2$:**
  - graph colouring
  - maximal matching
  - maximal independent set …
- **Key issue: symmetry breaking**

What Cannot Be Solved Locally?

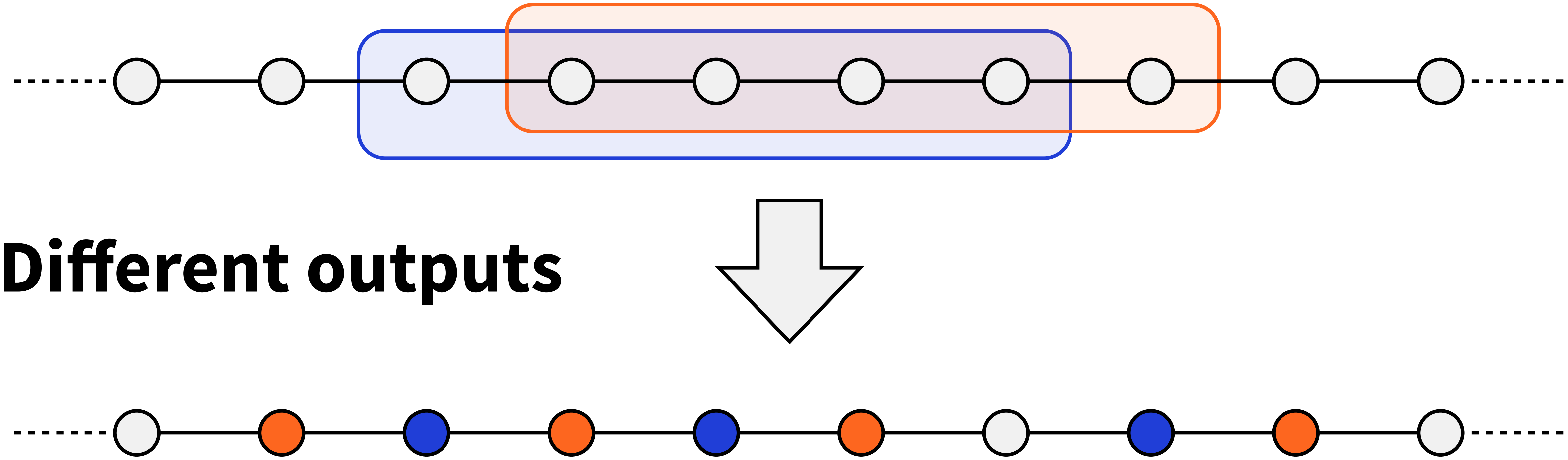
Example: graph colouring (G = long path)



What Cannot Be Solved Locally?

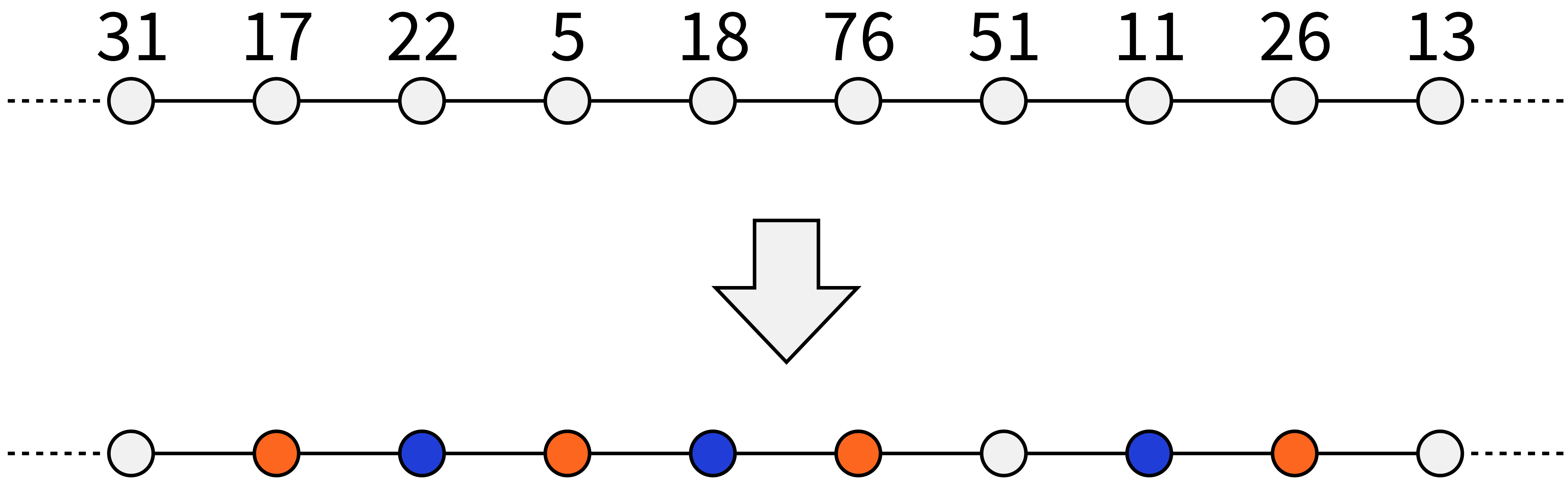
Identical local neighbourhoods

Different outputs



What Cannot Be Solved Locally?

Identical local neighbourhoods — really?

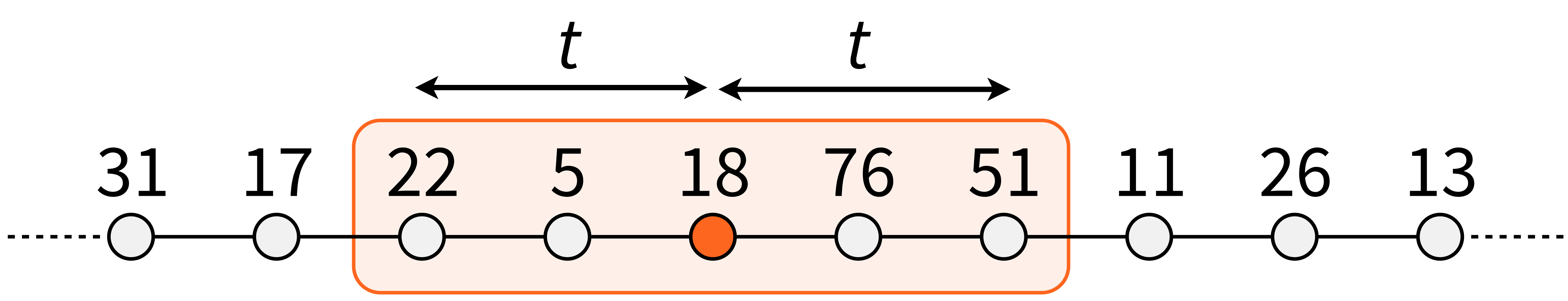


What Cannot Be Solved Locally?

- **Could we somehow use node identifiers to find a graph colouring?**
- **No — would require time $\Omega(\log^* n)$**
  - proof: apply **Ramsey's theorem**…

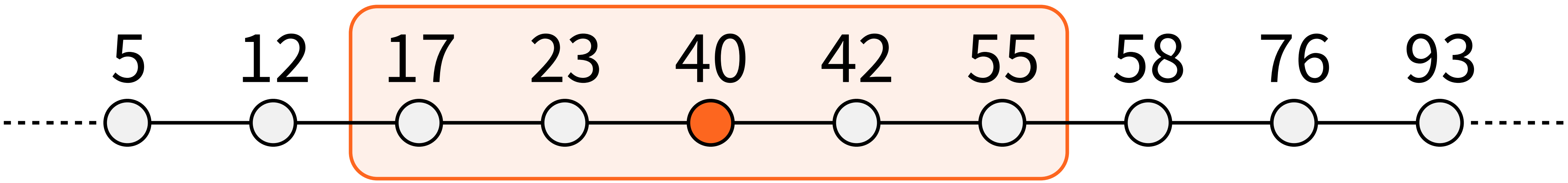
Node Identifiers Do Not Help

- ***G* = path:**
  - local output = f(sequence of k identifiers)
  - $k = 2t + 1$



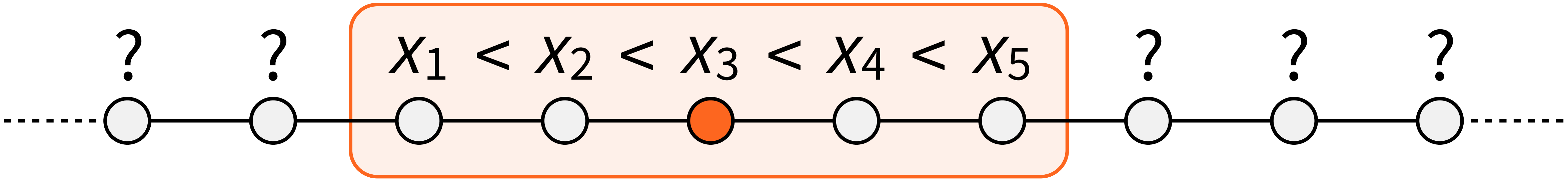
Node Identifiers Do Not Help

- G = **path, identifiers in increasing order:**
  - local output = ***f*(set of *k* identifiers)**
  - $k = 2t + 1$



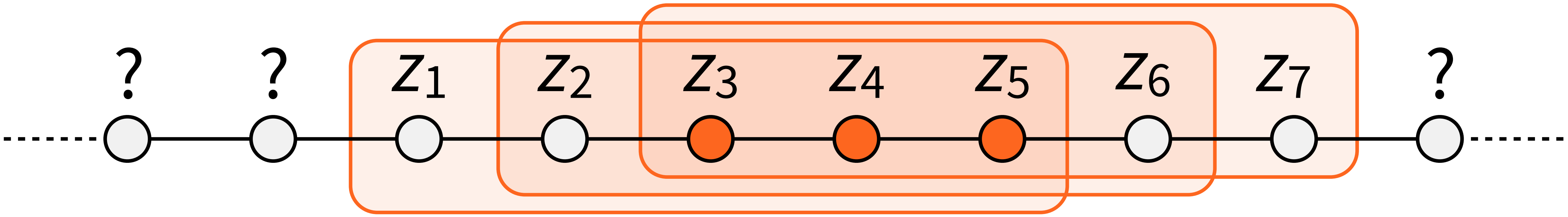
Node Identifiers Do Not Help

- **Universe U = set of all possible identifiers**
- **Colour of k-subset $X = \{x_1, x_2, \ldots, x_k\}$ of U:** output of algorithm A in this neighbourhood



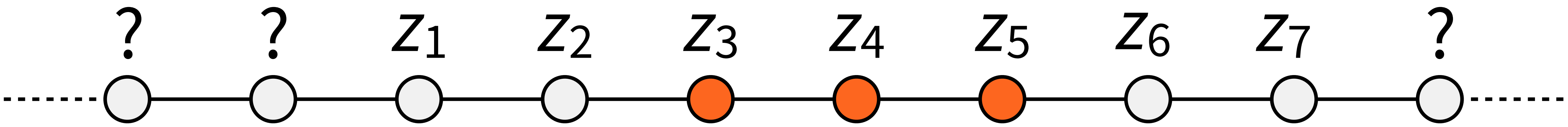
Node Identifiers Do Not Help

- **Ramsey:** there is a **monochromatic** subset $Z = \{z_1, z_2, \ldots, z_m\}$ of universe U
  - all k-subsets of Z have the same colour



Node Identifiers Do Not Help

- **We can construct a path s.t. many adjacent nodes have same local outputs**
  - graph colouring, maximal independent set, maximal matching: **not possible**



Node Identifiers Do Not Help

- **Without unique identifiers:**
  lower bounds often easy to prove
- **With unique identifiers:**
  lower bounds much more difficult
- **Ramsey-like arguments nontrivial for $\Delta > 2$**

Node Identifiers Do Not Help

- **New general result** (Göös et al., 2012):
  - node identifiers do not help with **any local approximation algorithm**
- **Assumption:** “simple” graph problems
  - vertex cover, edge cover, dominating set, matching, independent set, …

Everything Known?

- **Tight upper and lower bounds:**
  - vertex covers & edge covers
  - dominating sets & edge dominating sets
  - independent sets & matchings
  - packing & covering LPs
  - max-min & min-max LPs …

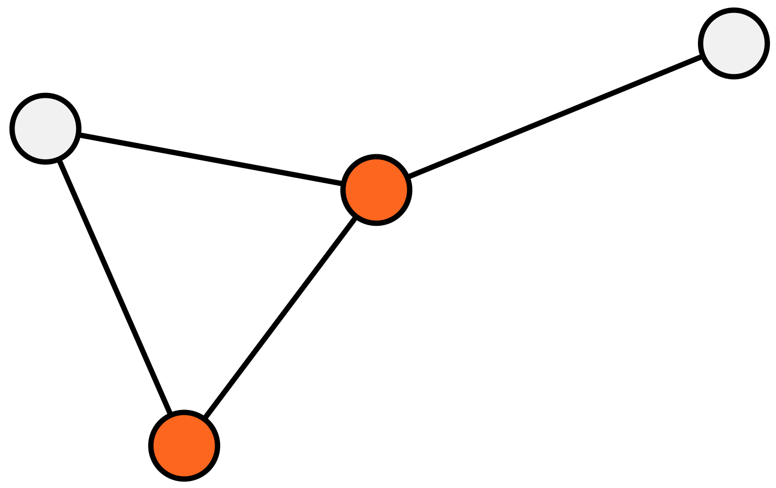
Everything Known?

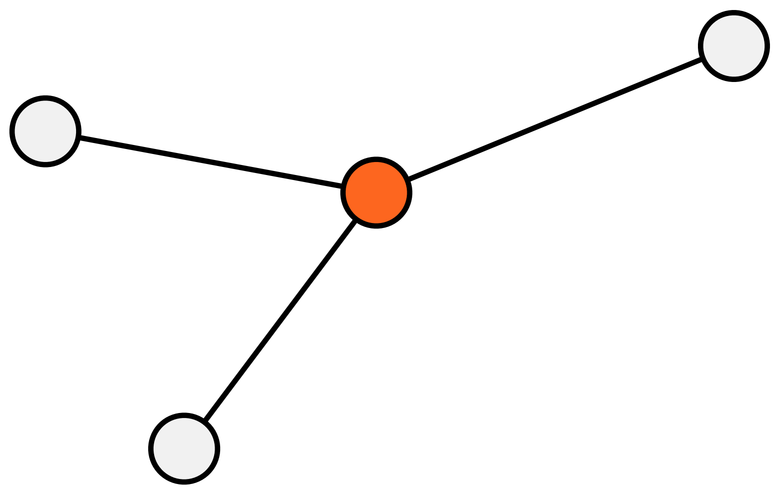
- **Well understood:**
  - what can we do in time that only depends on Δ?
- **Not so well understood:**
  - precisely **how** does t depend on Δ?
  - example: what can we do in time $t = o(\Delta)$?

Time vs. Maximum Degree

– running example: vertex covers and matching

Example:
Vertex Cover

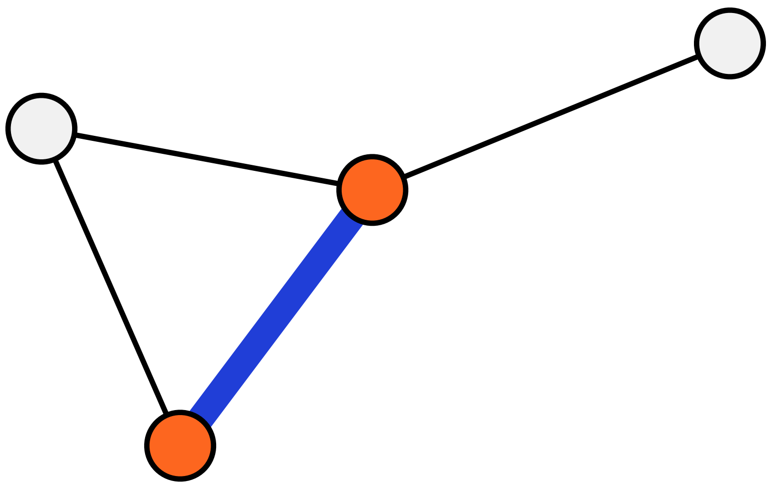




Subset C of nodes that "covers" all edges

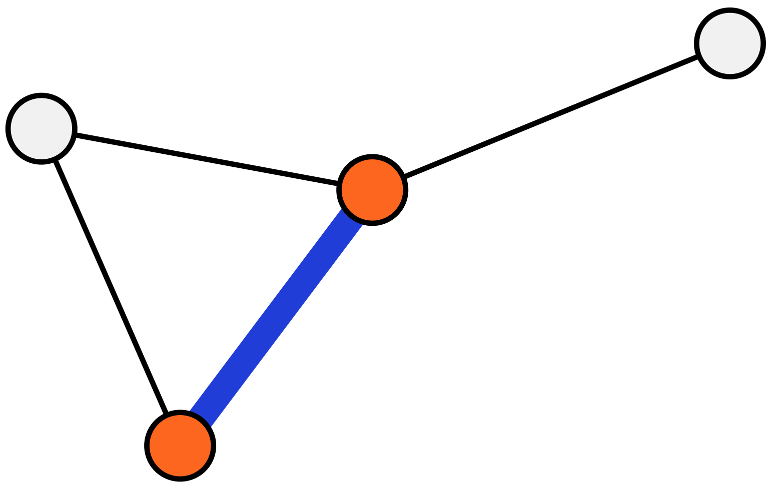
each edge has at least one endpoint in C

Example:
Vertex Cover



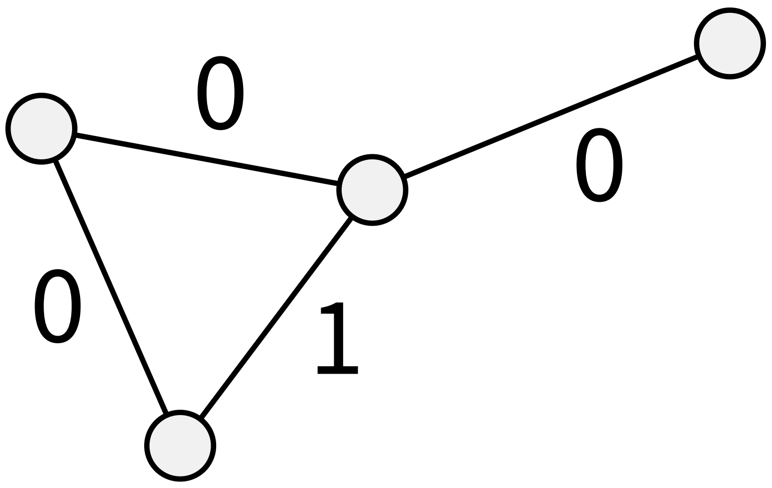
- **2-approximation of minimum vertex cover with centralised algorithms:**
  - find any **maximal matching *M***
  - output **all endpoints** of all edges in *M*
- **Not possible with local algorithms**
  - cannot find a maximal matching

Example:
Vertex Cover



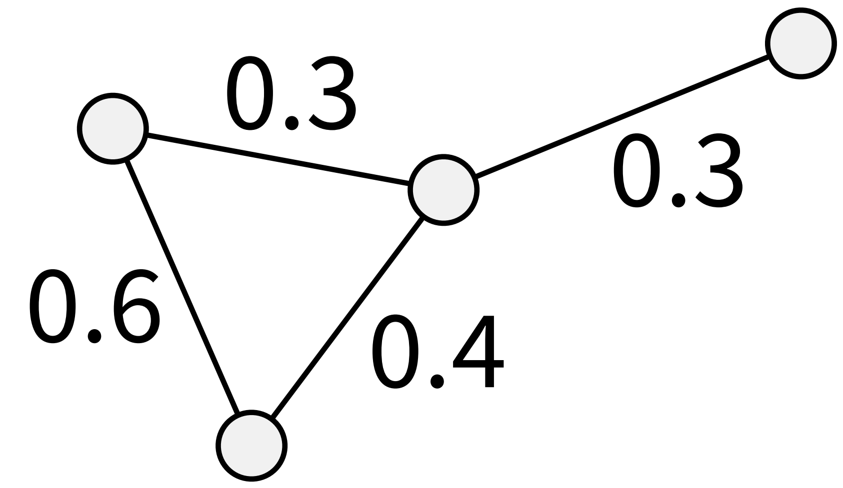
- **Maximal matching:**
  - requires symmetry breaking
- **Maximal fractional matching:**
  - no need to break symmetry
  - still helps with vertex cover approximations!

Matching



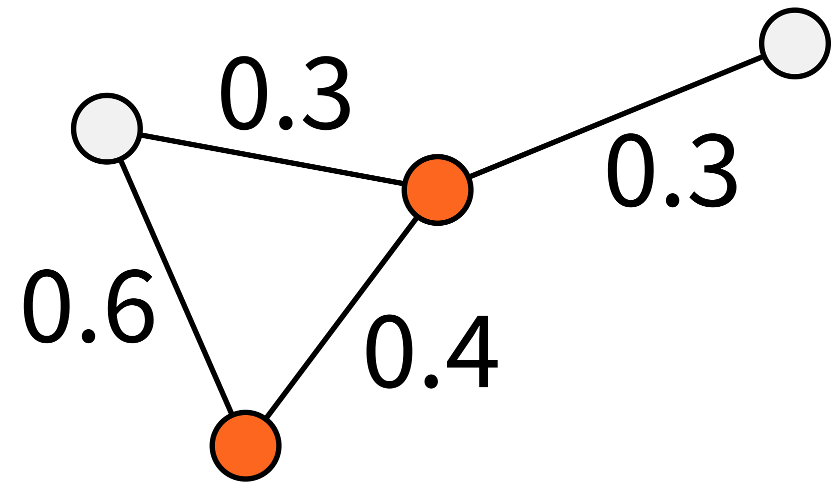
- **Edges labelled with integers {0, 1}**
- **Sum of incident edges at most 1**
- **Maximal matching:**
  cannot increase the value of any label

Fractional Matching



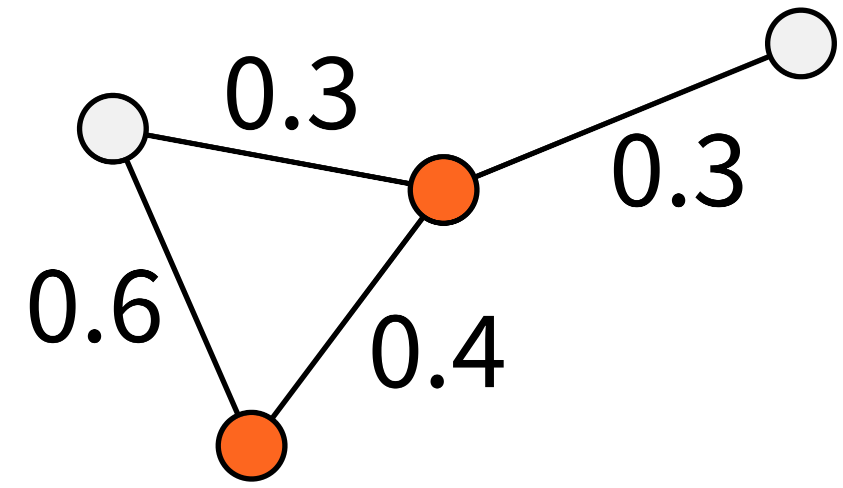
- **Edges labelled with real numbers [0, 1]**
- **Sum of incident edges at most 1**
- **Maximal fractional matching:**
  cannot increase the value of any label

Fractional Matching



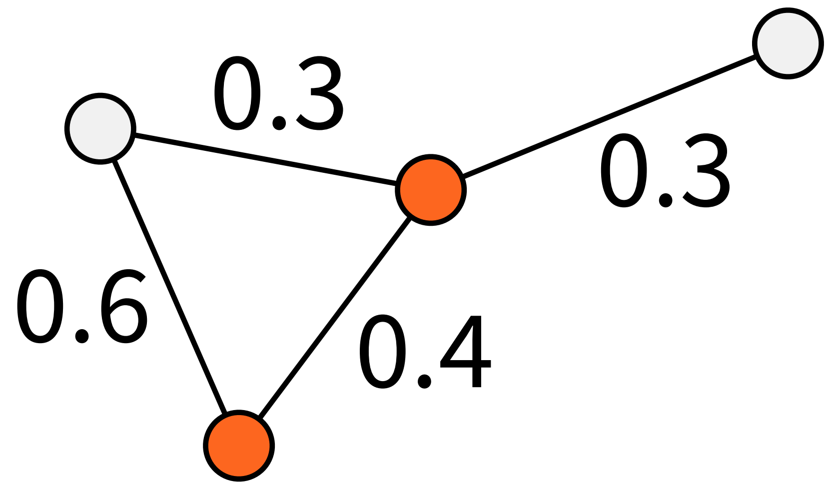
- **Saturated node:**
  - sum of incident edges = 1
- **2-approximation of minimum vertex cover:**
  - find any maximal fractional matching
  - take **all saturated nodes**

Fractional Matching



- **Maximal fractional matching in time $O(\Delta)$** (Åstrand & S., 2010)
  - does **not** require symmetry breaking
  - d-regular graph: label all edges with $1/d$
- **Nontrivial part:** graphs that are not regular…

Fractional Matching



- **Maximal fractional matching in time $O(\Delta)$**
  - 2-approximation of minimum vertex cover in time $O(\Delta)$
- **Can we do it faster?**

Maximal Fractional Matching

– new lower bound

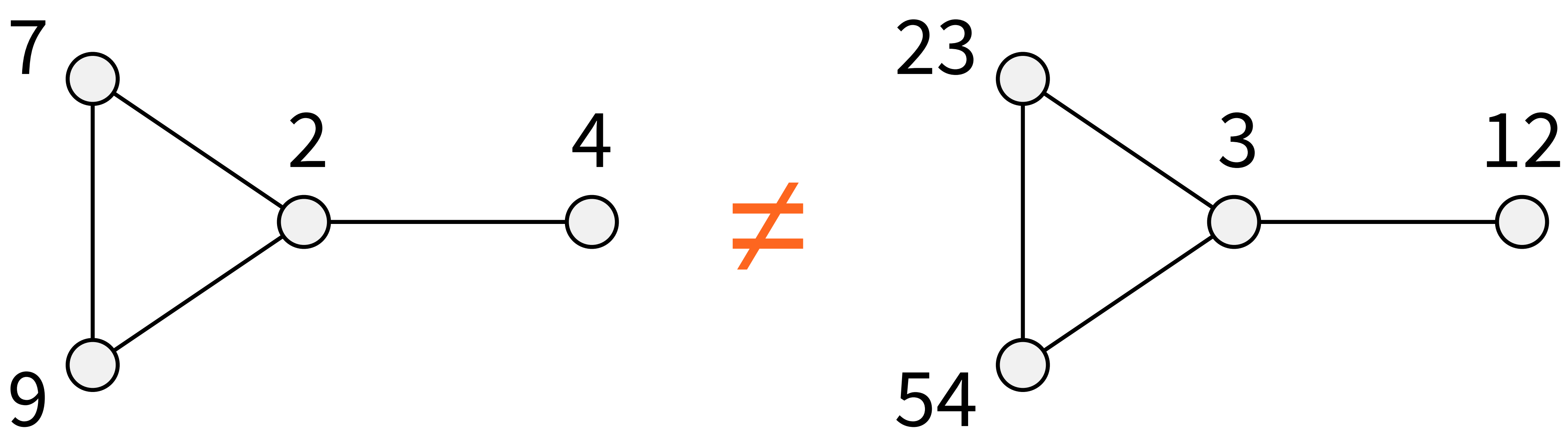
Maximal Fractional Matching

- **Cannot be solved in time $o(\Delta)$**
  (Göös et al., 2013)

- **Key ingredient of the proof:**
  analyse many **different models** of distributed computing

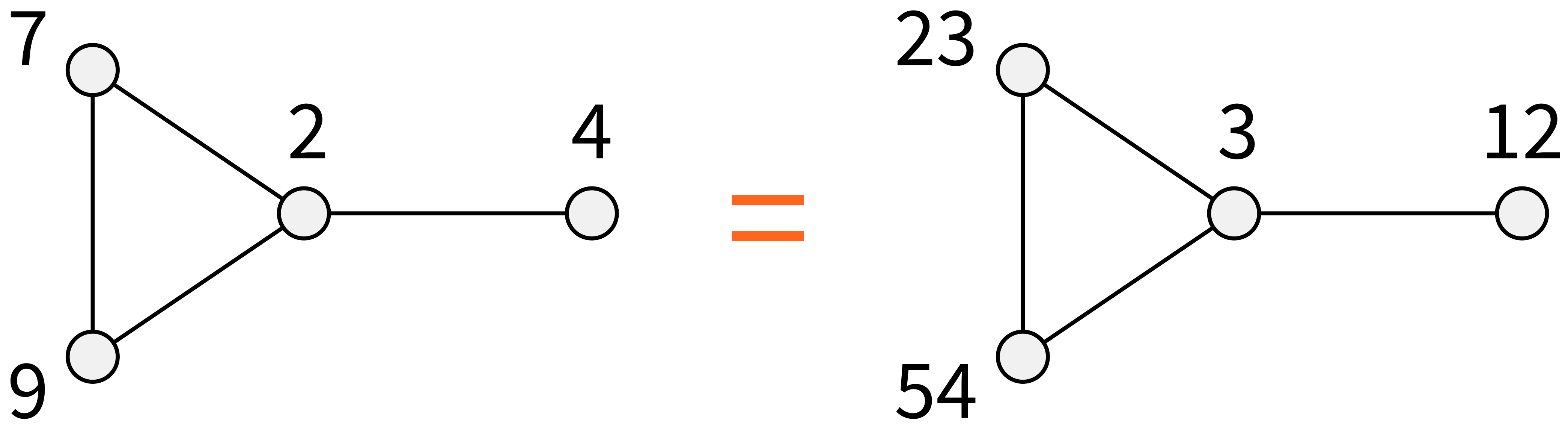
ID:
Unique Identifiers

Nodes have unique identifiers,
output may depend on them



OI:
Order Invariant

Output does not change if we change identifiers but keep their relative order

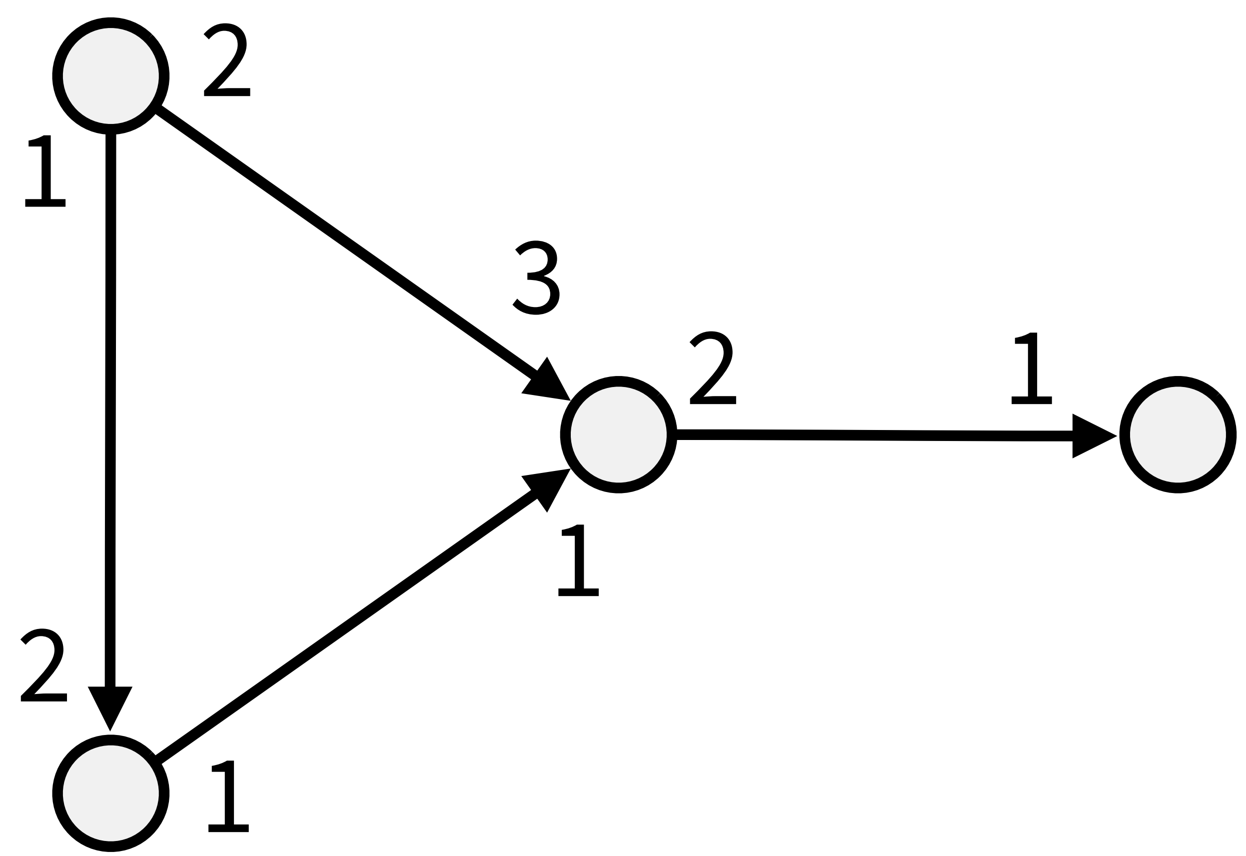


PO:
Ports & Orientation

No identifiers

Node *v* labels incident edges with 1, …, deg(*v*)

Edges oriented

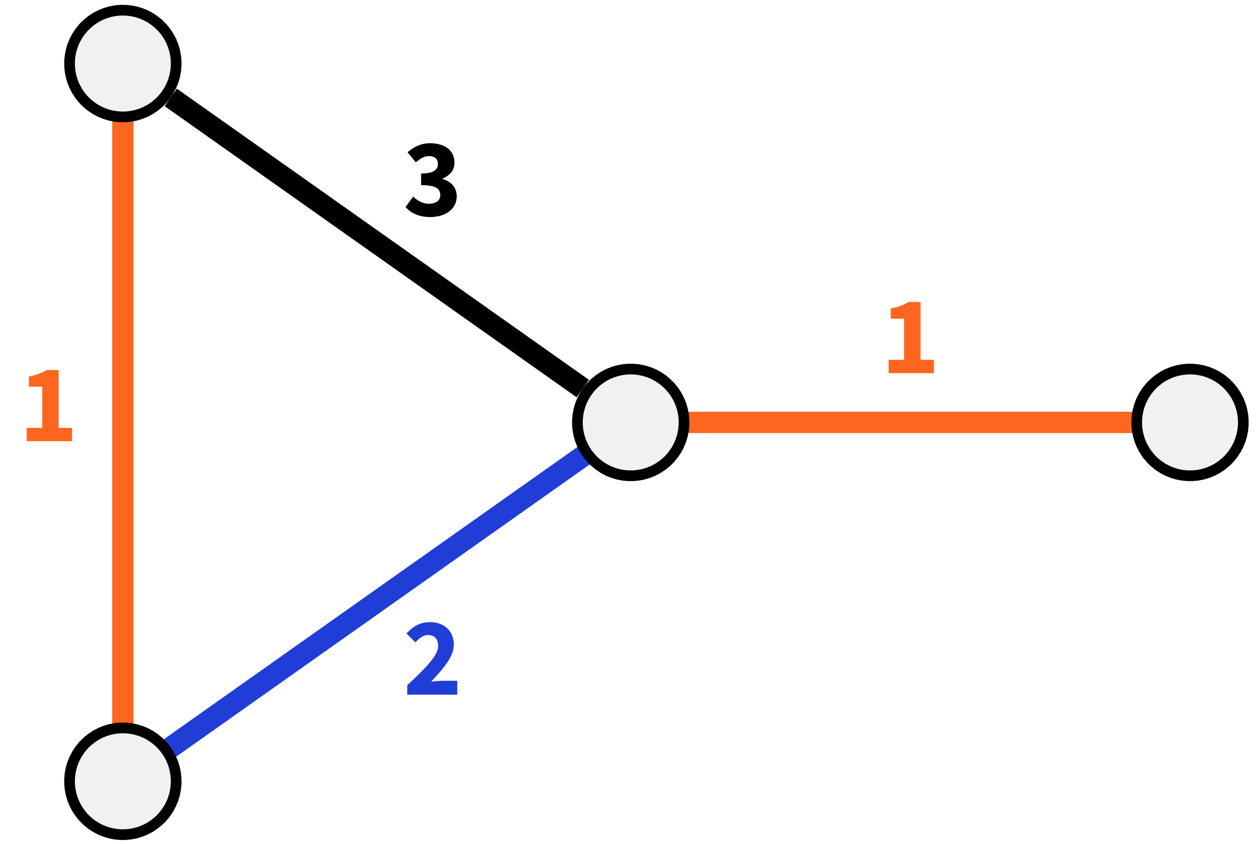


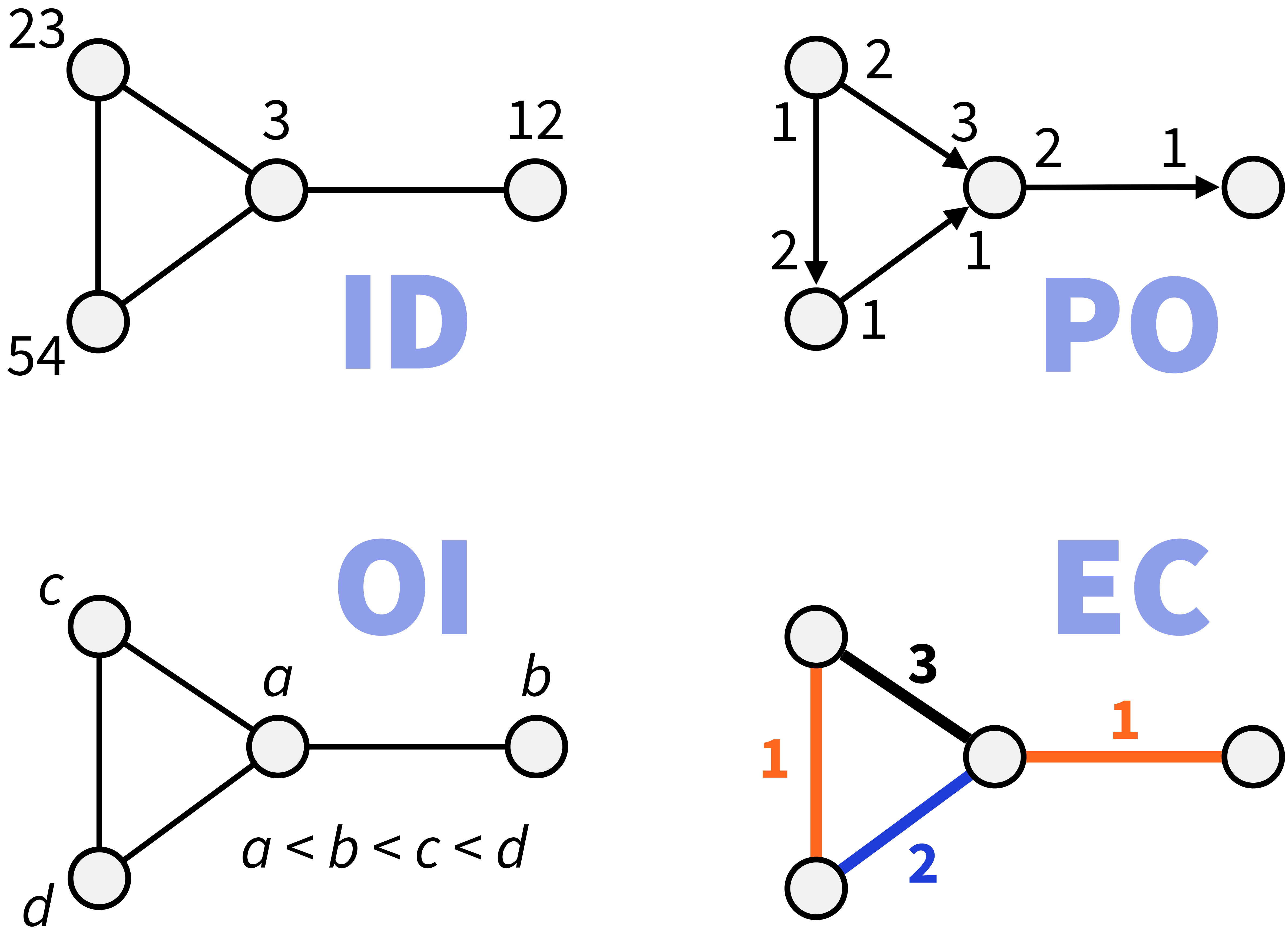
EC:
Edge Colouring

No identifiers

No orientations

Edges coloured
with $O(\Delta)$ colours





23
3
12
54
ID
2
1
3
2
1
2
1
1
PO
OI
c
a
b
d
$a < b < c < d$
EC
3
1
1
2

Oversimplified Proof Overview

- **EC model is very limited**
  - maximal fractional matching requires $\Omega(\Delta)$ time in EC
- **Simulation argument: EC → PO → OI → ID**
  - maximal fractional matching requires $\Omega(\Delta)$ time also in PO, OI, and ID

Bipartite Vertex Cover

– König’s theorem
  is not local

König Duality

- **Bipartite graphs**
- **C^* = minimum vertex cover**
- **M^* = maximum matching**
- **König: $|C^*| = |M^*|$**

Distributed Approximation

Finding (1+ε)-approximation
for constant $\varepsilon > 0$, constant Δ

	Matching	Vertex Cover
Integral	?	?
Fractional	?	?

Distributed Approximation

Kuhn et al. (2004)

Finding (1+ε)-approximation
for constant $\varepsilon > 0$, constant Δ

	Matching	Vertex Cover
Integral	?	?
Fractional	$O(1)$	$O(1)$

Distributed Approximation

Åstrand et al. (2010)

Finding (1+ε)-approximation
for constant ε > 0, constant Δ

	Matching	Vertex Cover
Integral	$O(1)$	?
Fractional	$O(1)$	$O(1)$

Distributed Approximation

Finding (1+ε)-approximation
for constant $\varepsilon > 0$, constant Δ

	Matching	Vertex Cover
Integral	$O(1)$	?
Fractional	$O(1)$	$O(1)$

Distributed Approximation

Göös & S. (2012)

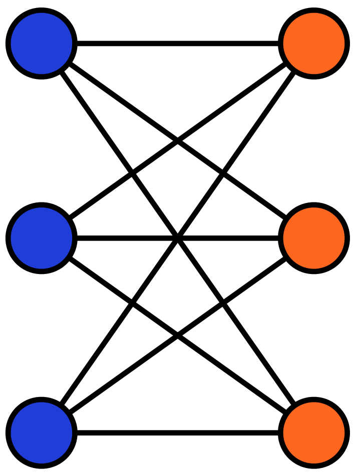
Finding (1+ε)-approximation
for constant $\varepsilon > 0$, constant Δ

	Matching	Vertex Cover
Integral	$O(1)$	$\Omega(\log n)$
Fractional	$O(1)$	$O(1)$

Bipartite Vertex Cover

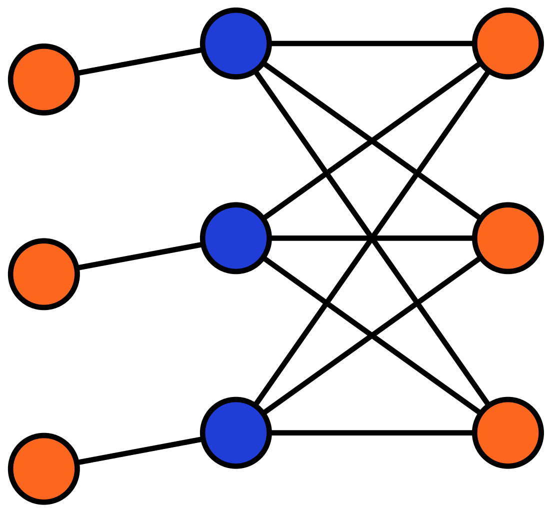
- **No $o(\log n)$-time distributed algorithm for 1.01-approximation of minimum vertex cover**
  - even if we study bipartite graphs of maximum degree 3
  - key ingredient: **expander graphs**

Bipartite Vertex Cover



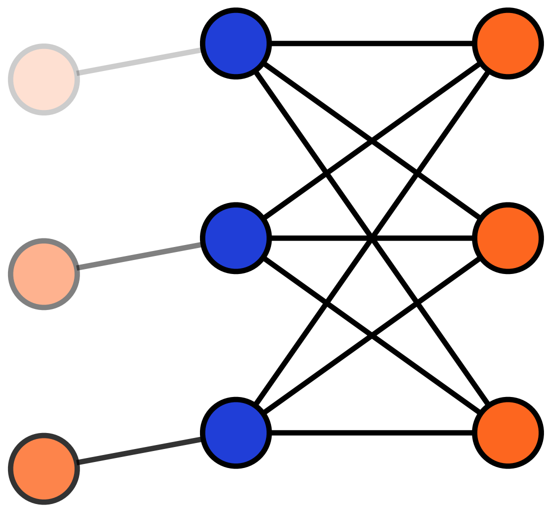
- **Assume we are given an algorithm *A***
- **See what it does in a regular bipartite graph *G***
  - algorithm picks **all orange** or **all blue** nodes
  - w.l.o.g., assume that it picks **all orange** nodes

Bipartite Vertex Cover



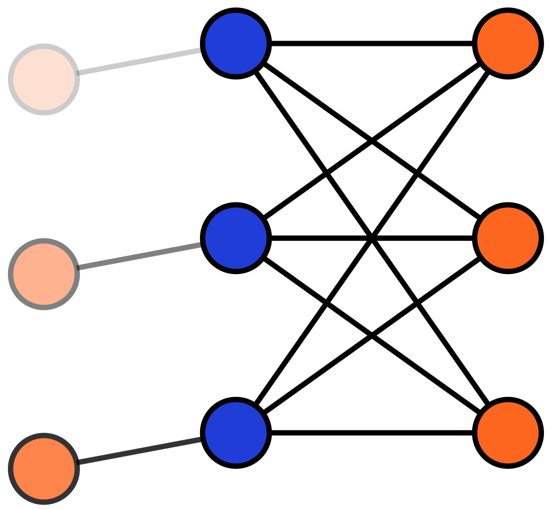
- **Assume we are given an algorithm *A***
- **See what it does in a regular bipartite graph *G***
  - algorithm picks **all orange** nodes
- **Add some gadgets to construct graph *G'***
  - algorithm must pick **all blue** nodes

Bipartite Vertex Cover



- **Construct a sequence of graphs G_0, G_1, … G_n**
- **Start with the regular graph $G_0 = G$**
  - algorithm picks **all orange** nodes in G_0
- **Add gadgets one by one so that $G_n = G'$**
  - algorithm must pick **all blue** nodes in G_n

Bipartite Vertex Cover



- **Construct a sequence of graphs G_0, G_1, … G_n**
  - algorithm picks **all orange** nodes in G_0
  - algorithm must pick **all blue** nodes in G_n
  - only small change from G_i to G_{i+1} if algorithm runs in time $o(\log n)$
  - for some G_k we have **half orange** + **half blue**

Bipartite Vertex Cover

- **For some G_k algorithm outputs**
  half orange + half blue
  - G_k is an expander
  - large **orange**–**blue** boundary
  - many edges redundantly covered
  - poor approximation ratio

Toolbox for
Lower Bound Proofs

- **Highly symmetric graphs:**
  local neighbourhoods "look identical"
- **Ramsey-like arguments:**
  unique identifiers do not help, either
- **Expanders:**
  cannot "hide boundaries"

Toolbox for Lower Bound Proofs

- **Highly symmetric graphs:**
  local neighbourhoods “look identical”
- **Ramsey-like arguments:**
  unique identifiers do not help, either
- **Expanders:**
  cannot “hide boundaries”

Thanks!