Local certification for graph problems

Laurent Feuilloley

LIP6 · Sorbonne Université

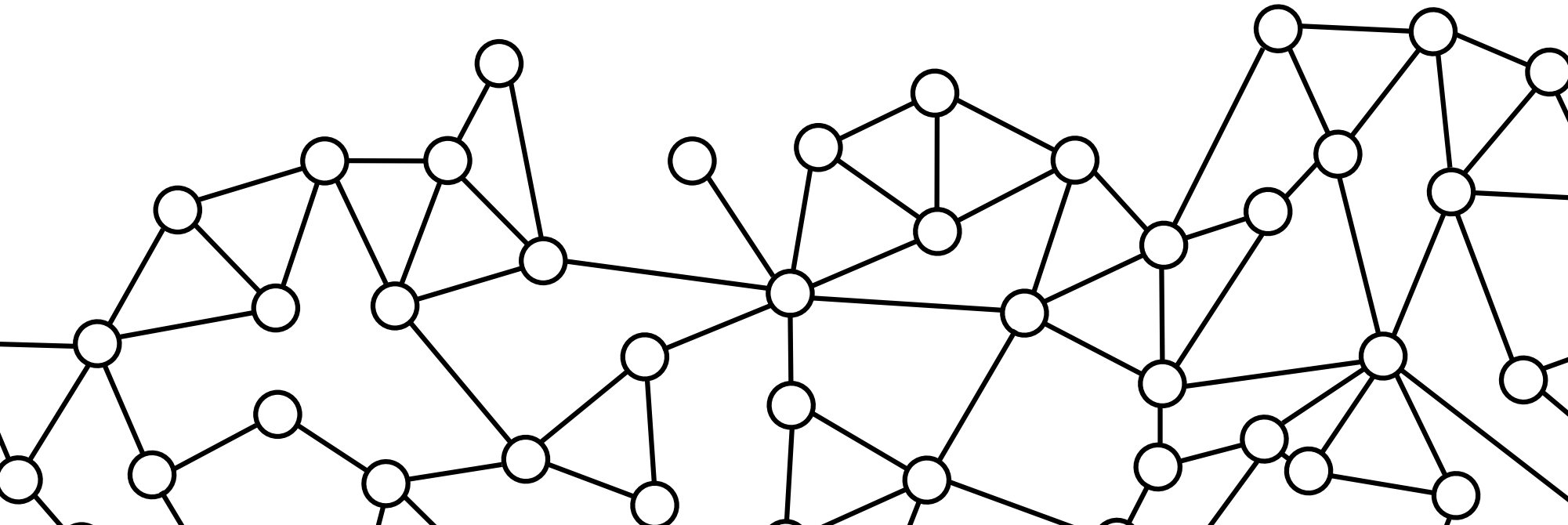
TALGO Seminar · November 2018

Outline

- A graph-oriented model of distributed computing.
- Verifying solutions via certification.
- Typical examples, with upper and lower bounds.
- Recent research directions

Distributed network computing

- No asynchrony, dead-locks, crashes, philosophers.
- Graphs and graph problems
- Locality flavour : similarity property testing, online algorithm, greedy algorithms etc.

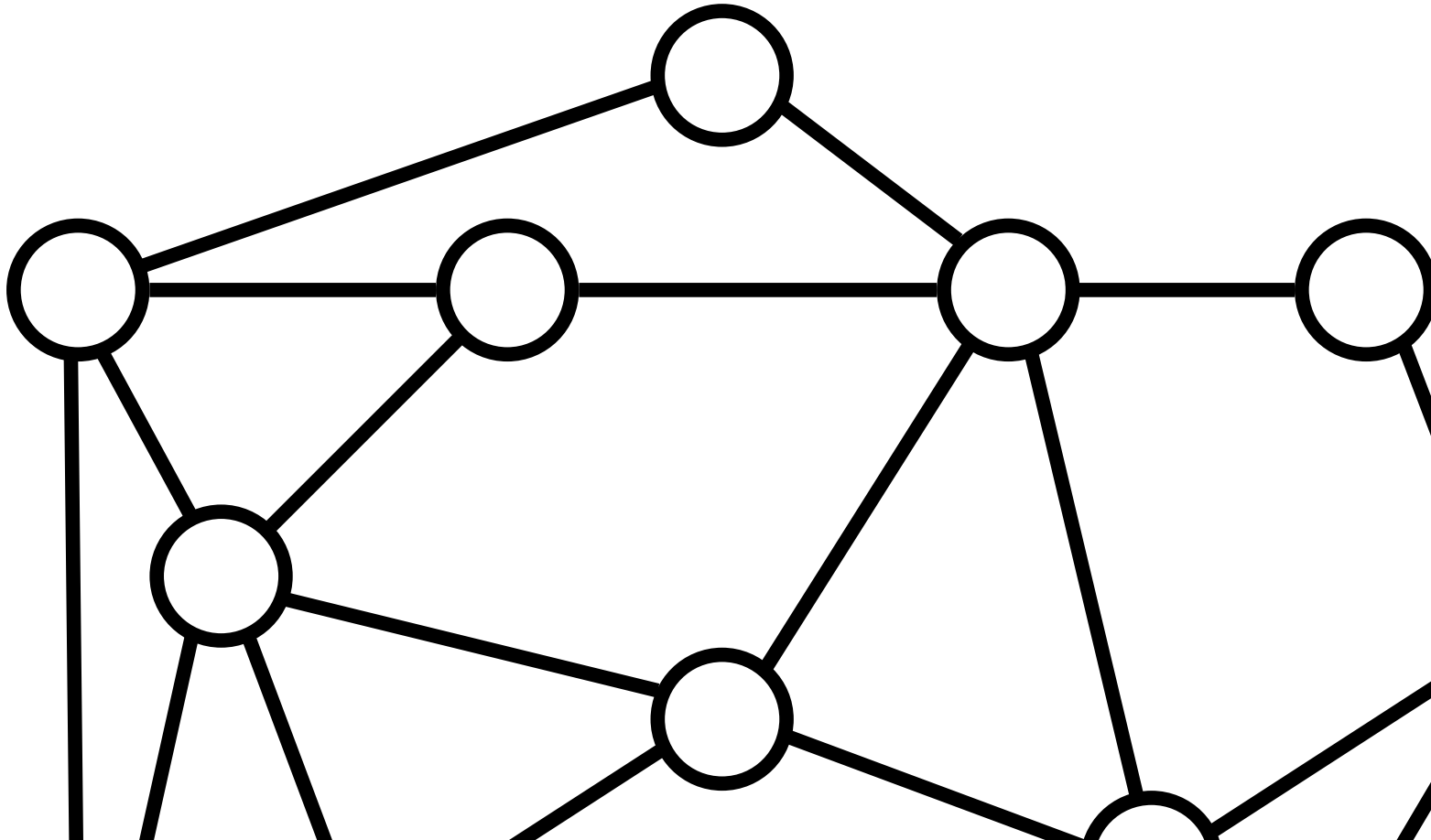


LOCAL model

Nodes $\rightarrow$ machines

Edges $\rightarrow$ Communication channels

Round : sending, receiving, internal computation.

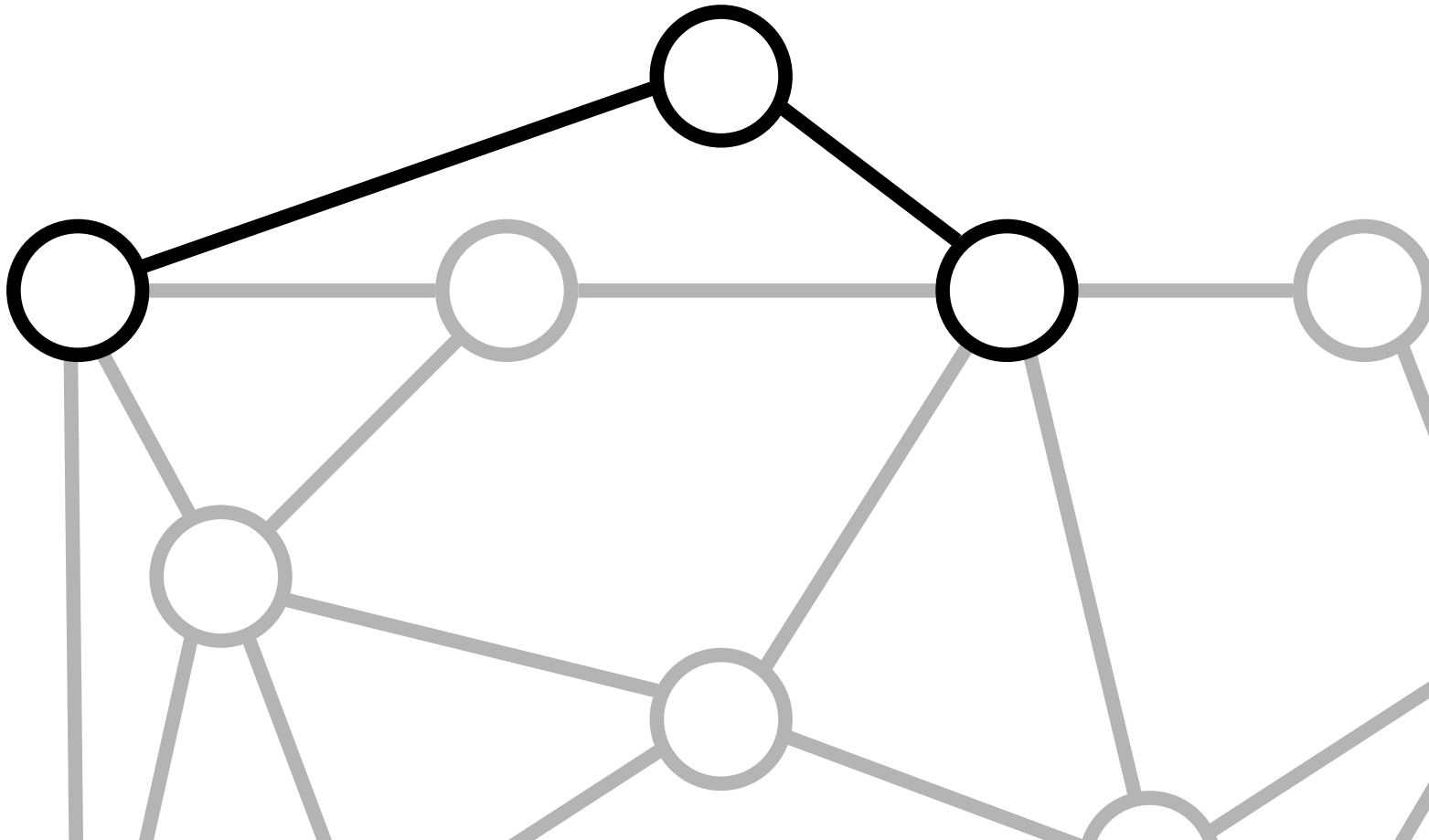


LOCAL model

Nodes $\rightarrow$ machines

Edges $\rightarrow$ Communication channels

Round : sending, receiving, internal computation.

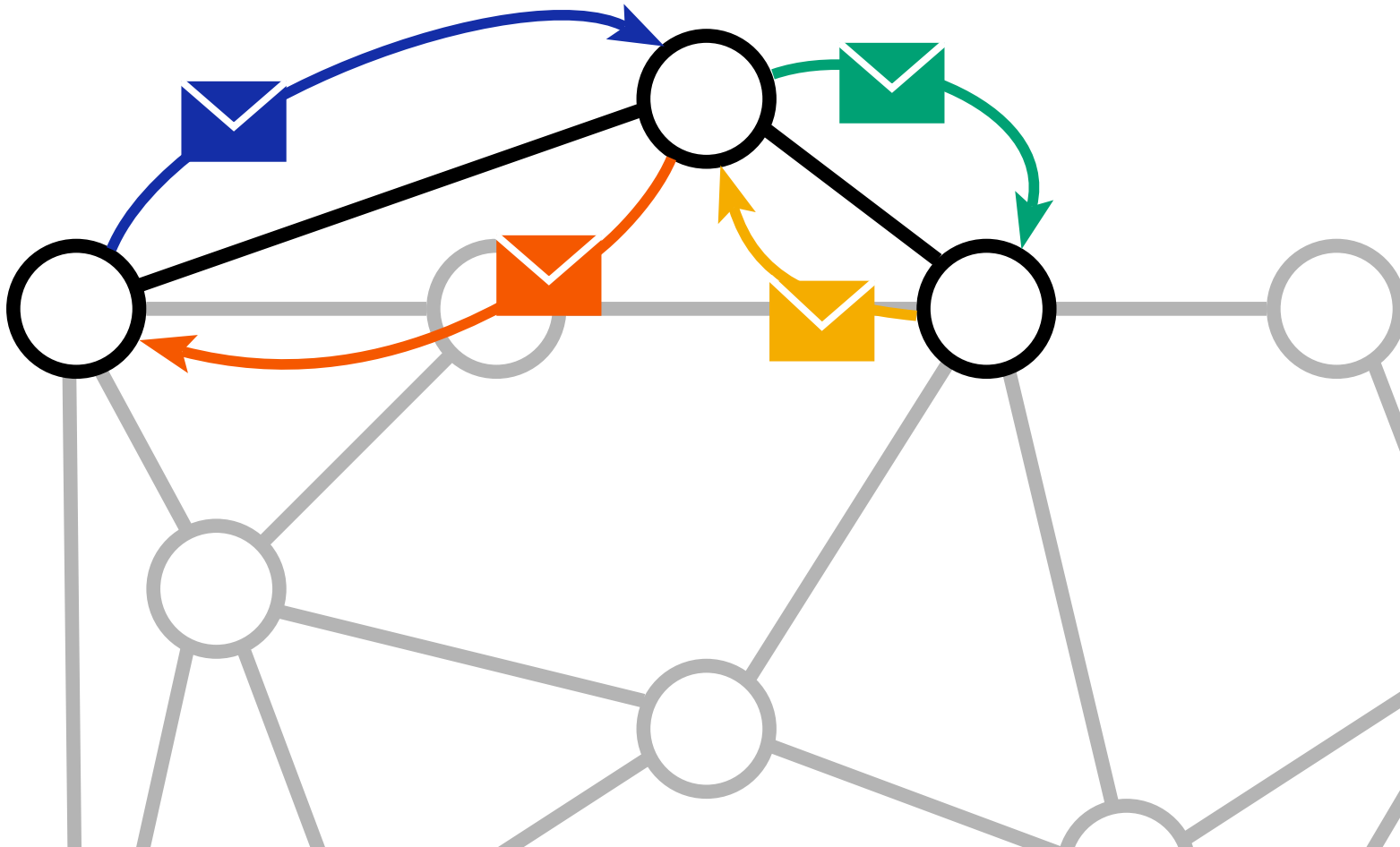


LOCAL model

Nodes $\rightarrow$ machines

Edges $\rightarrow$ Communication channels

Round : sending, receiving, internal computation.

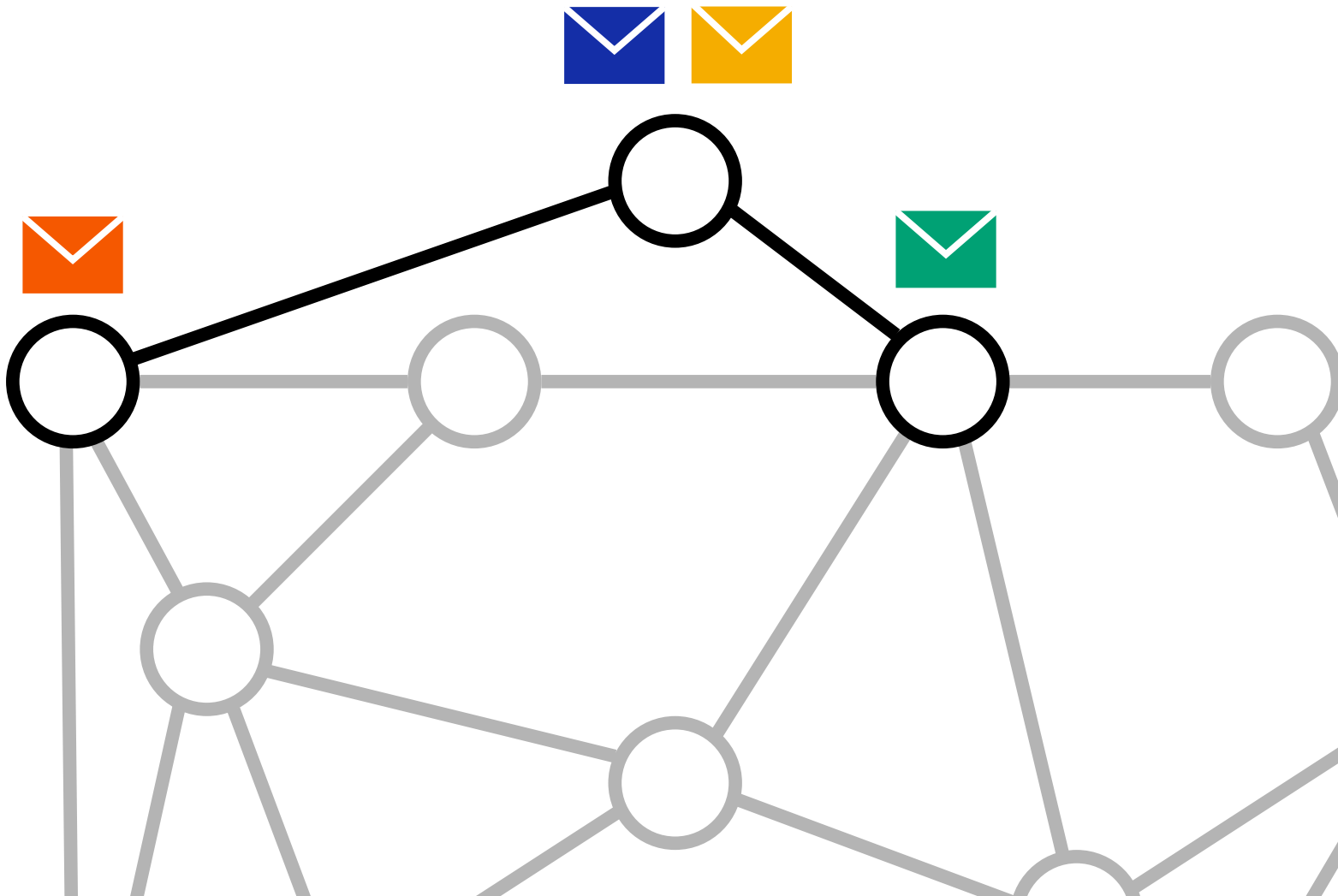


LOCAL model

Nodes $\rightarrow$ machines

Edges $\rightarrow$ Communication channels

Round : sending, receiving, internal computation.

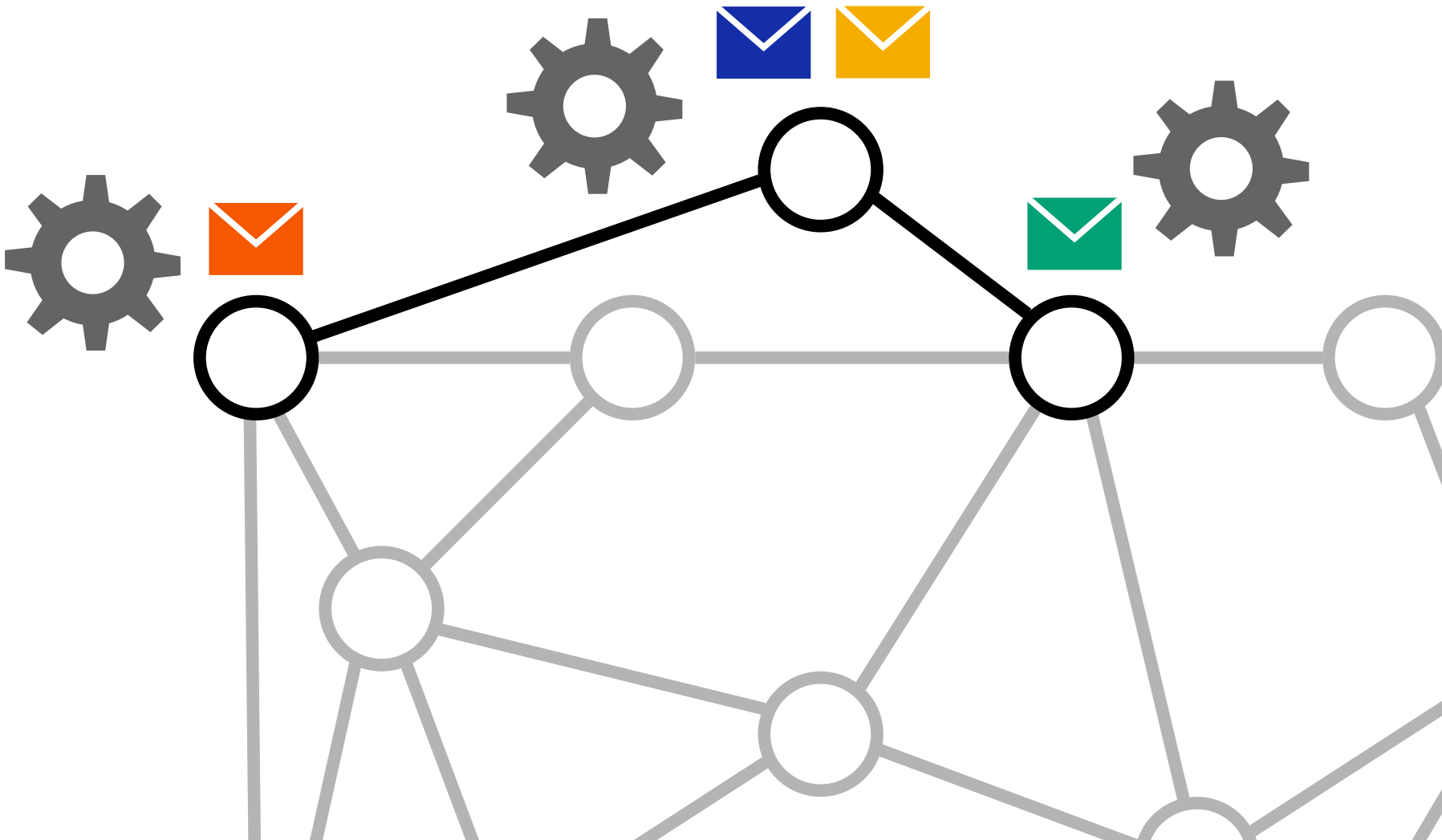


LOCAL model

Nodes $\rightarrow$ machines

Edges $\rightarrow$ Communication channels

Round : sending, receiving, internal computation.

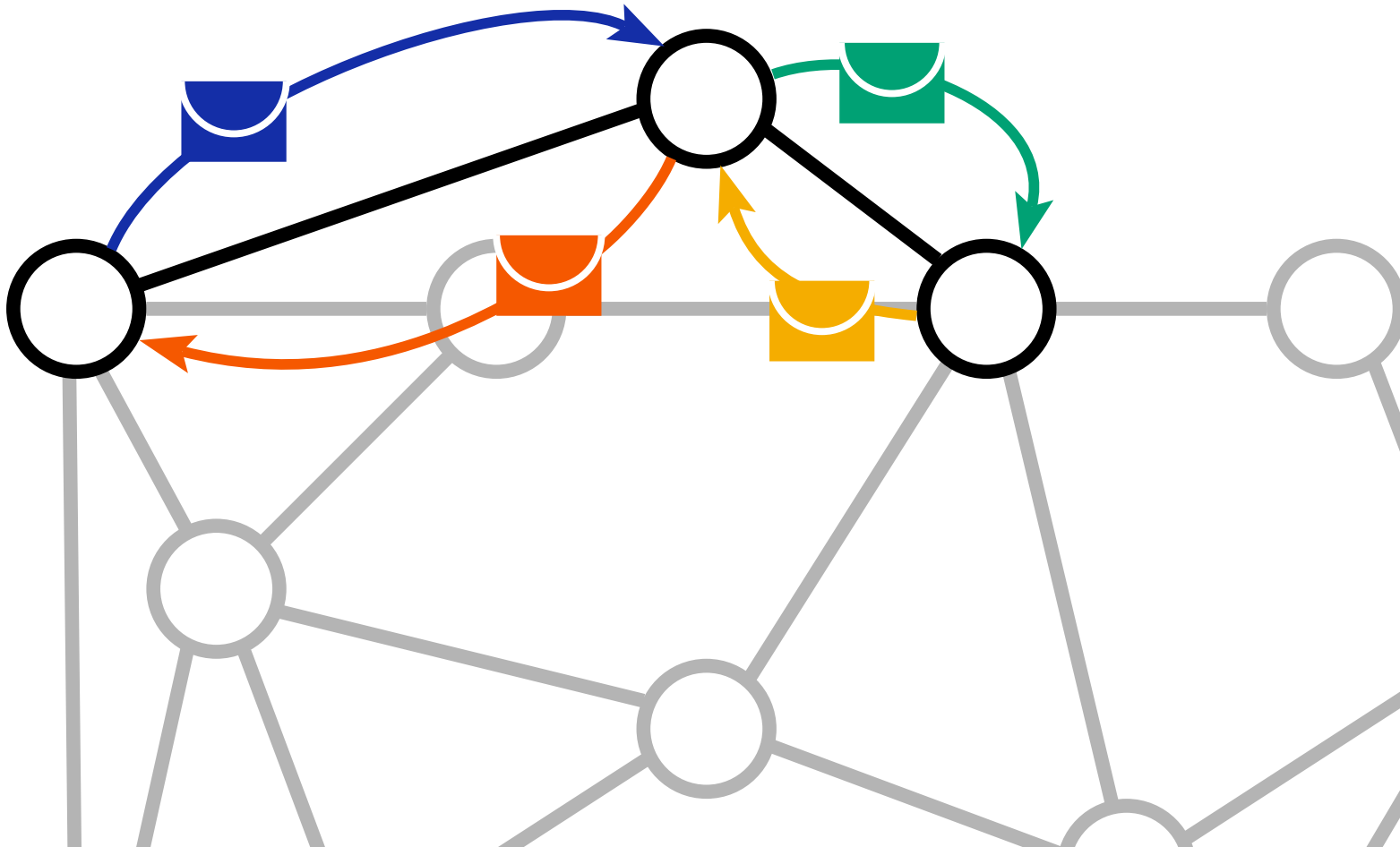


LOCAL model

Nodes $\rightarrow$ machines

Edges $\rightarrow$ Communication channels

Round : sending, receiving, internal computation.



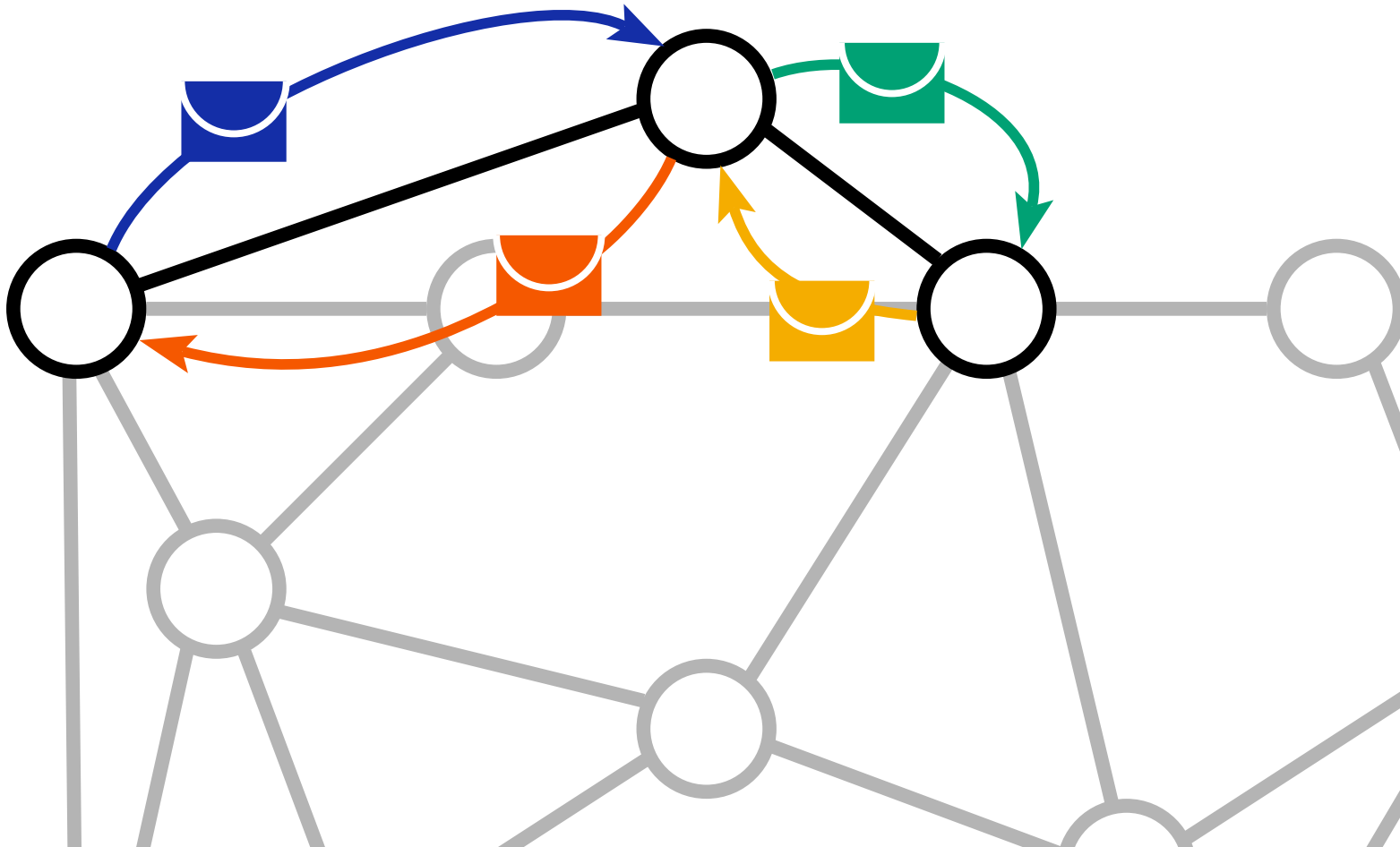
LOCAL model

Nodes $\rightarrow$ machines

Edges $\rightarrow$ Communication channels

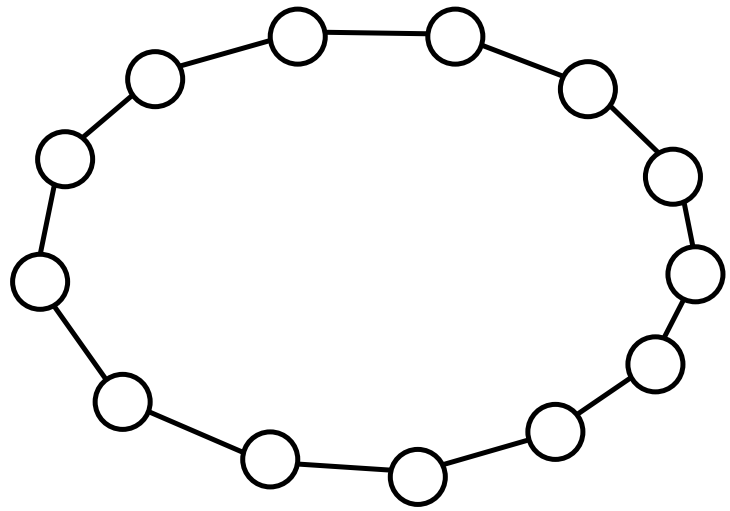
Round : sending, receiving, internal computation.

Complexity : # rounds, # messages, message size



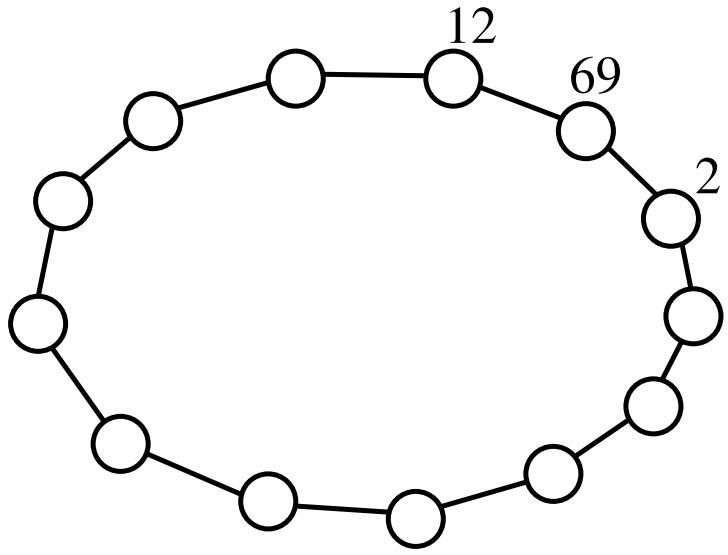
Spanning tree

Task : Select edges that form a spanning tree.



Spanning tree

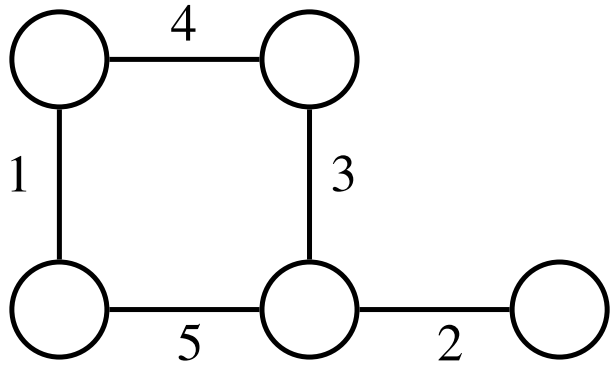
Task : Select edges that form a spanning tree.

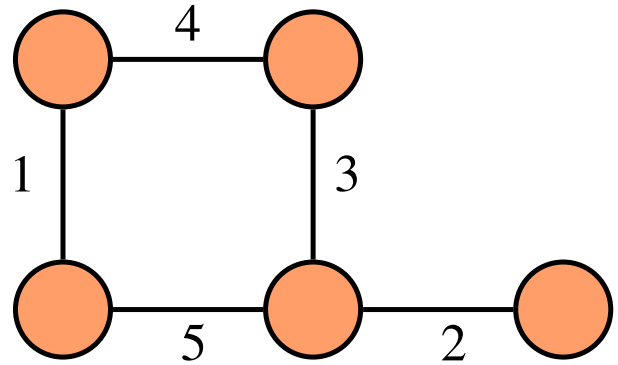


To break symmetry : unique identifiers.
Algorithm sketch : every node grows a BFS, the one with the smallest ID survives.

Minimum spanning tree

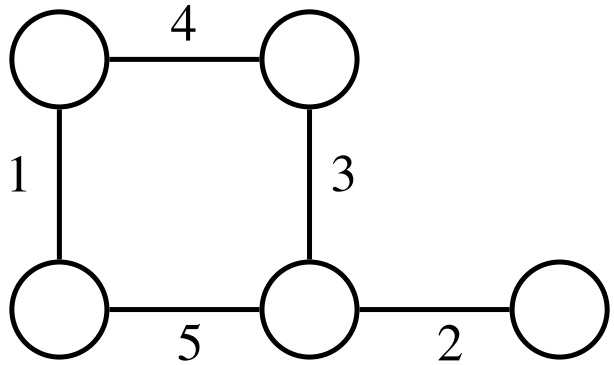
Task : Given the weights, select edges that form a minimum weight spanning tree.
First take : Prim, Kruskal.
The good one : Boruvka.

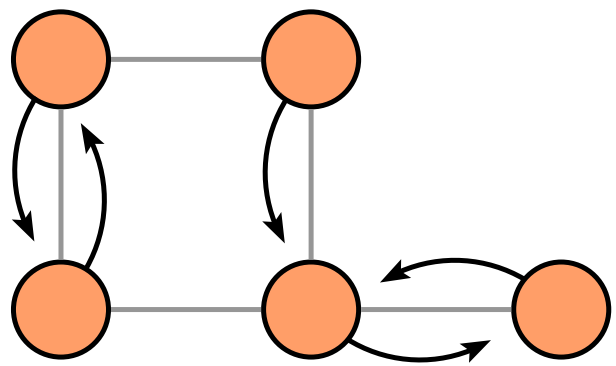




Minimum spanning tree

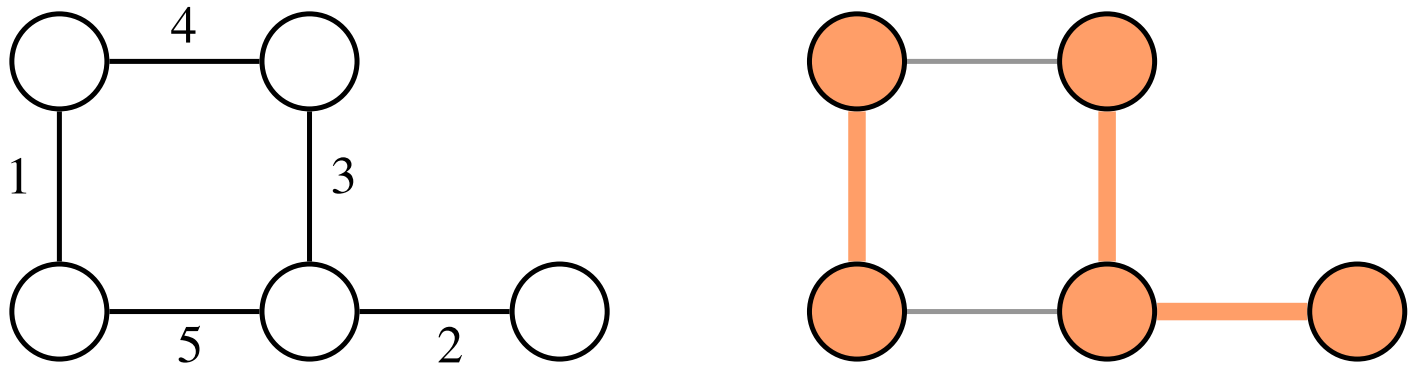
Task : Given the weights, select edges that form a minimum weight spanning tree.
First take : Prim, Kruskal.
The good one : Boruvka.





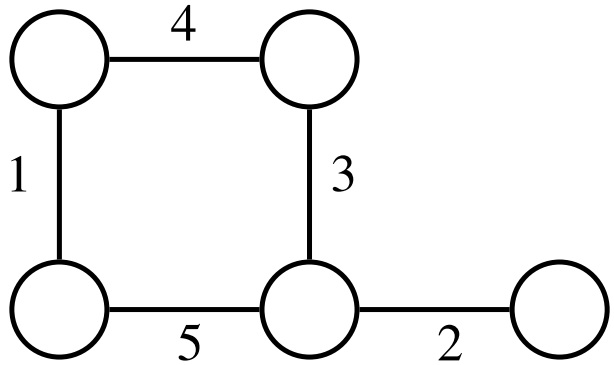
Minimum spanning tree

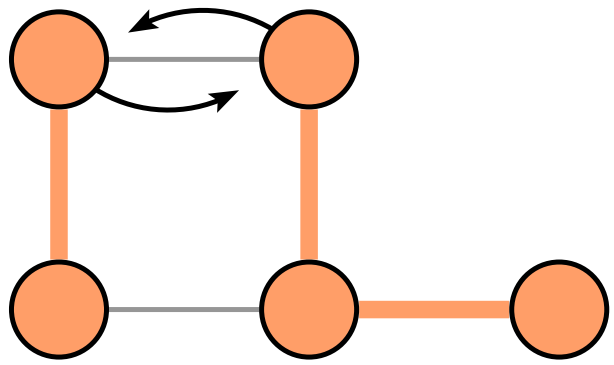
Task : Given the weights, select edges that form a minimum weight spanning tree.
First take : Prim, Kruskal.
The good one : Boruvka.



Minimum spanning tree

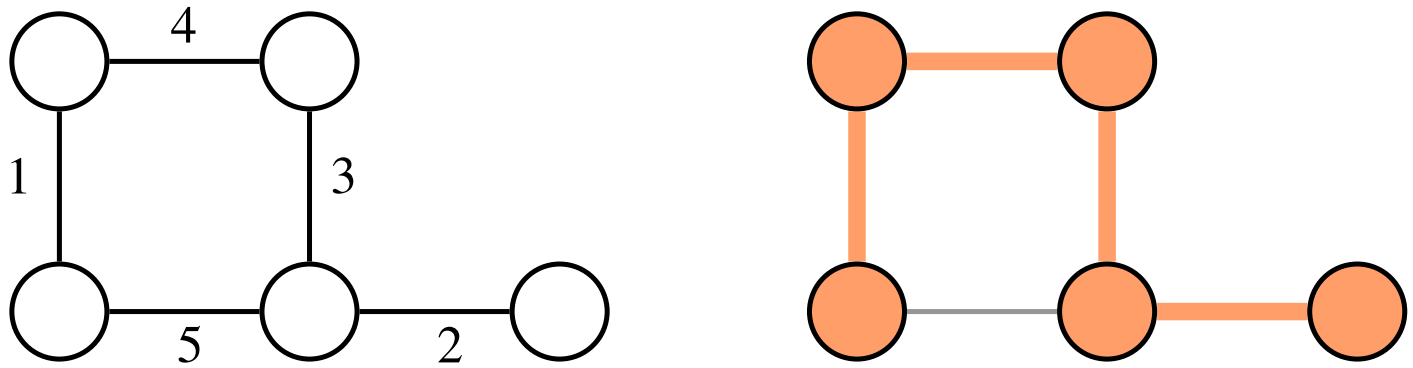
Task : Given the weights, select edges that form a minimum weight spanning tree.
First take : Prim, Kruskal.
The good one : Boruvka.





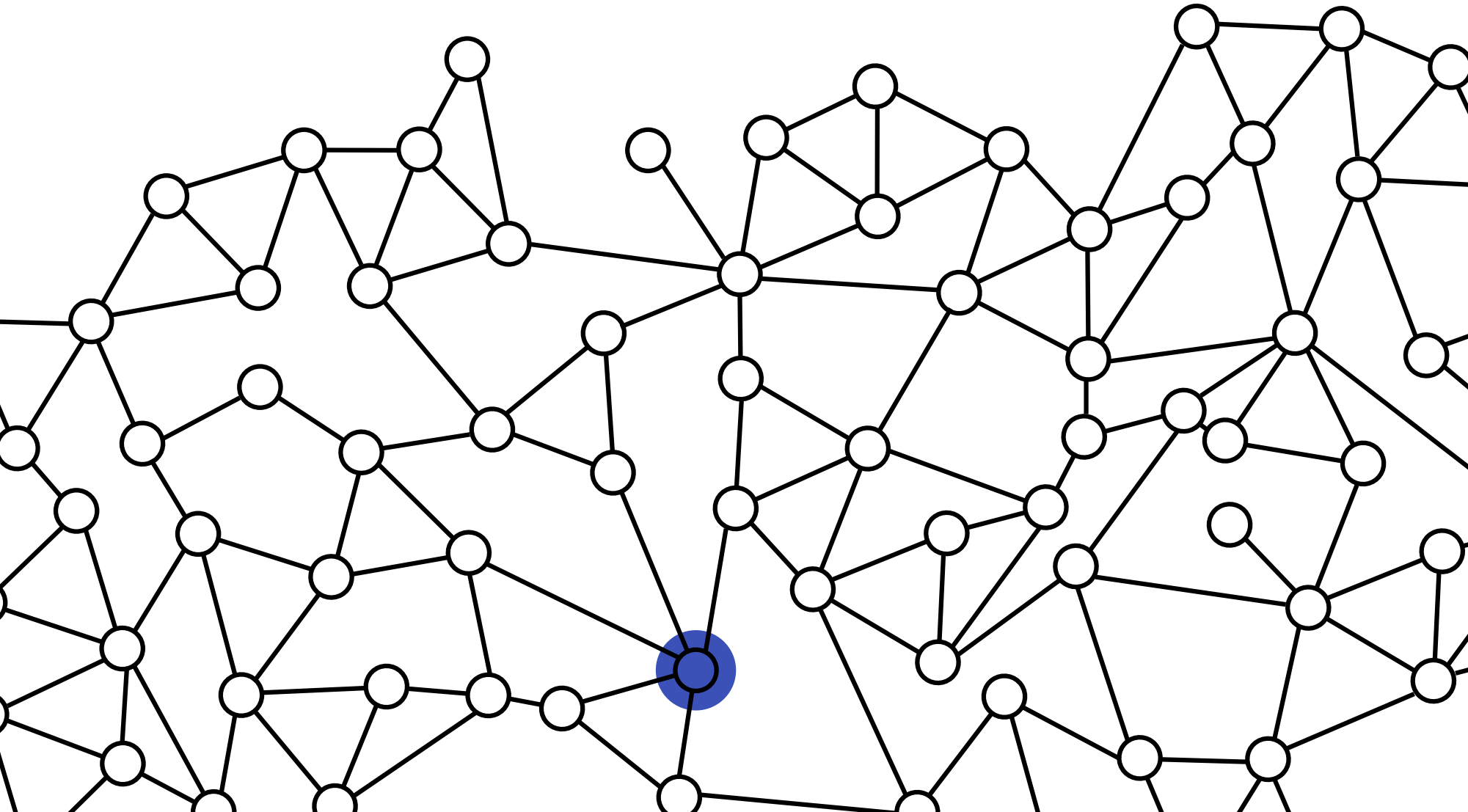
Minimum spanning tree

Task : Given the weights, select edges that form a minimum weight spanning tree.
First take : Prim, Kruskal.
The good one : Boruvka.



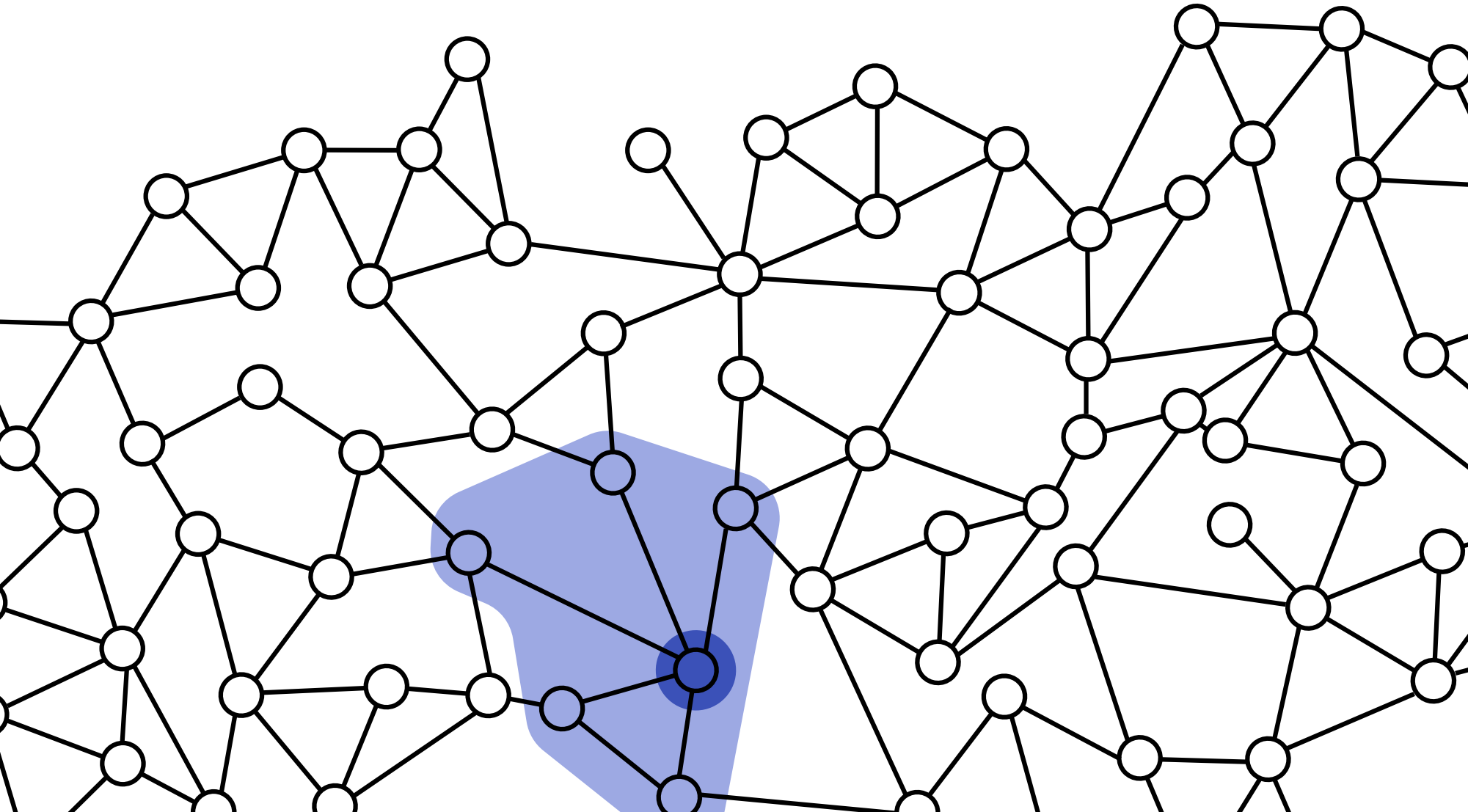
Colouring

Task : Colouring a bounded-degree graphs
Solved in $O(\log^* n)$ rounds $\rightarrow$ a really local problem.



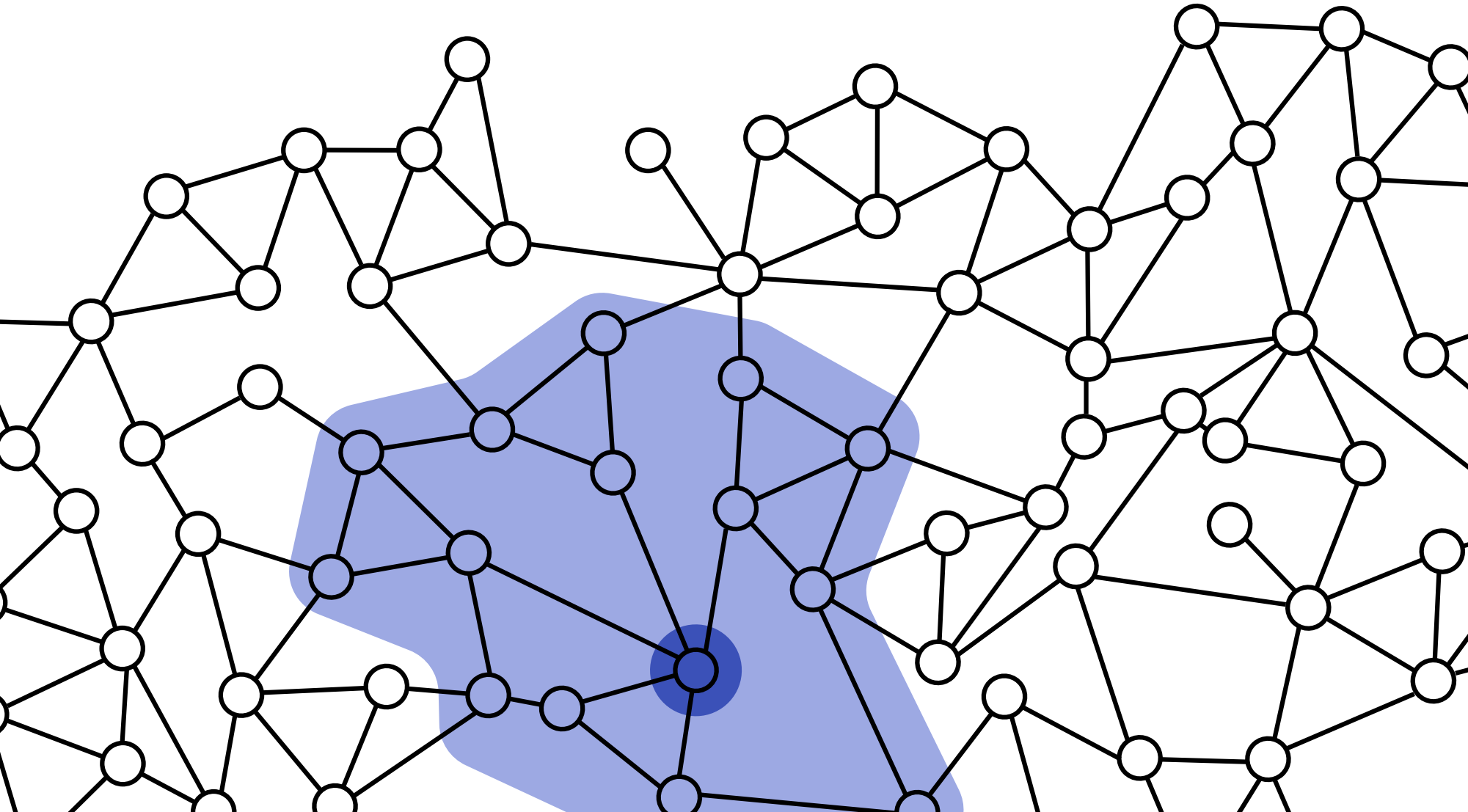
Colouring

Task : Colouring a bounded-degree graphs
Solved in $O(\log^* n)$ rounds $\rightarrow$ a really local problem.



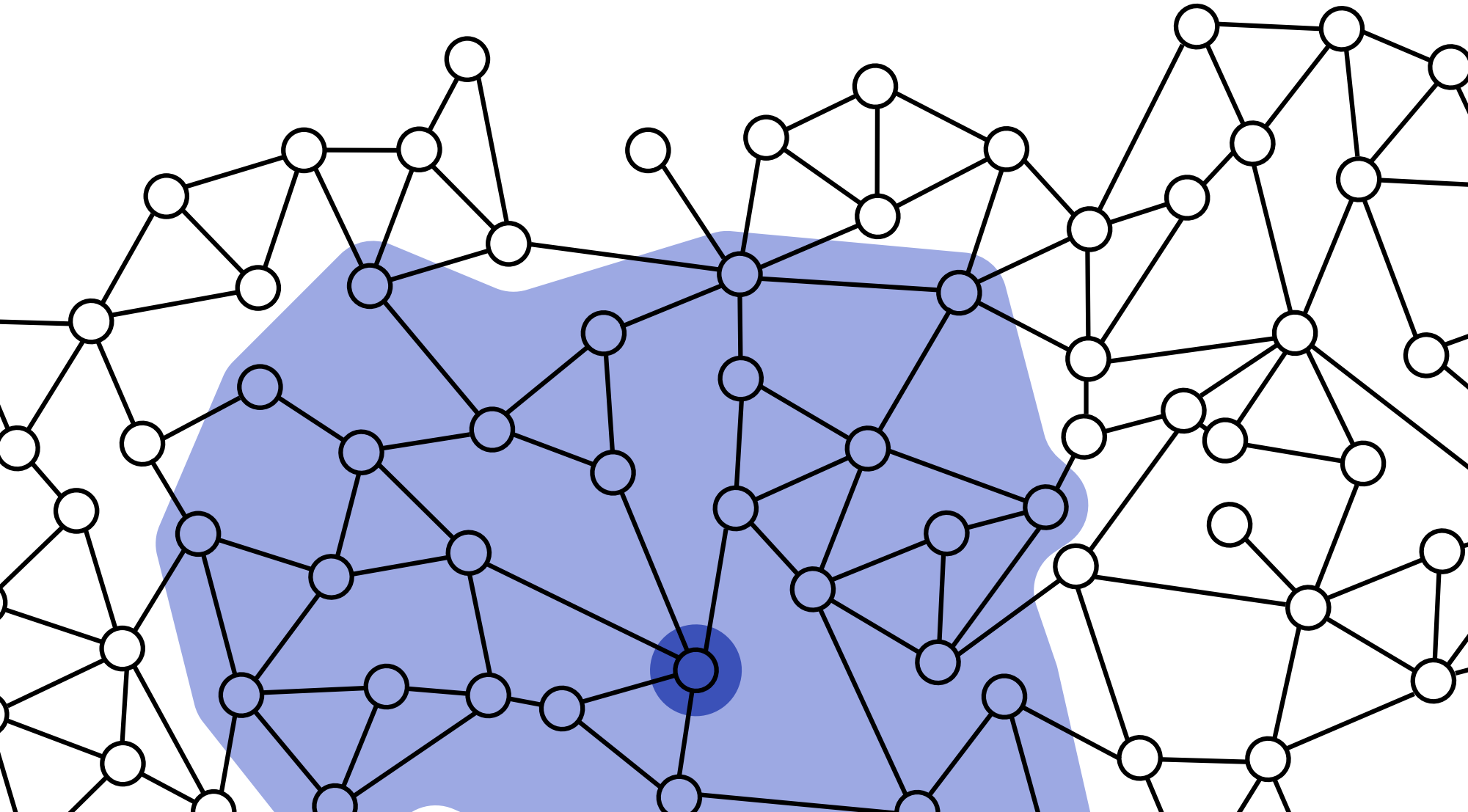
Colouring

Task : Colouring a bounded-degree graphs
Solved in $O(\log^* n)$ rounds $\rightarrow$ a really local problem.



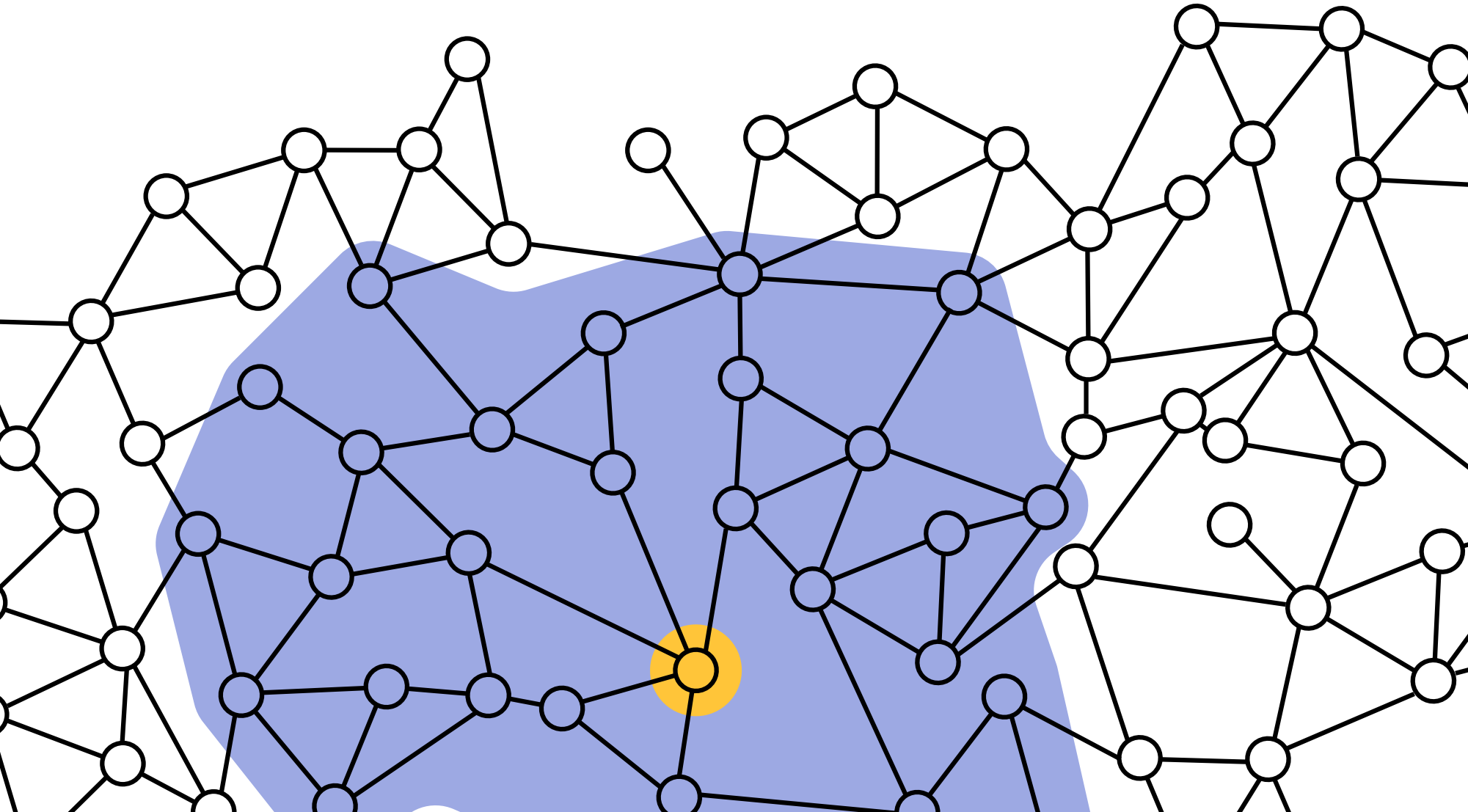
Colouring

Task : Colouring a bounded-degree graphs
Solved in $O(\log^* n)$ rounds $\rightarrow$ a really local problem.



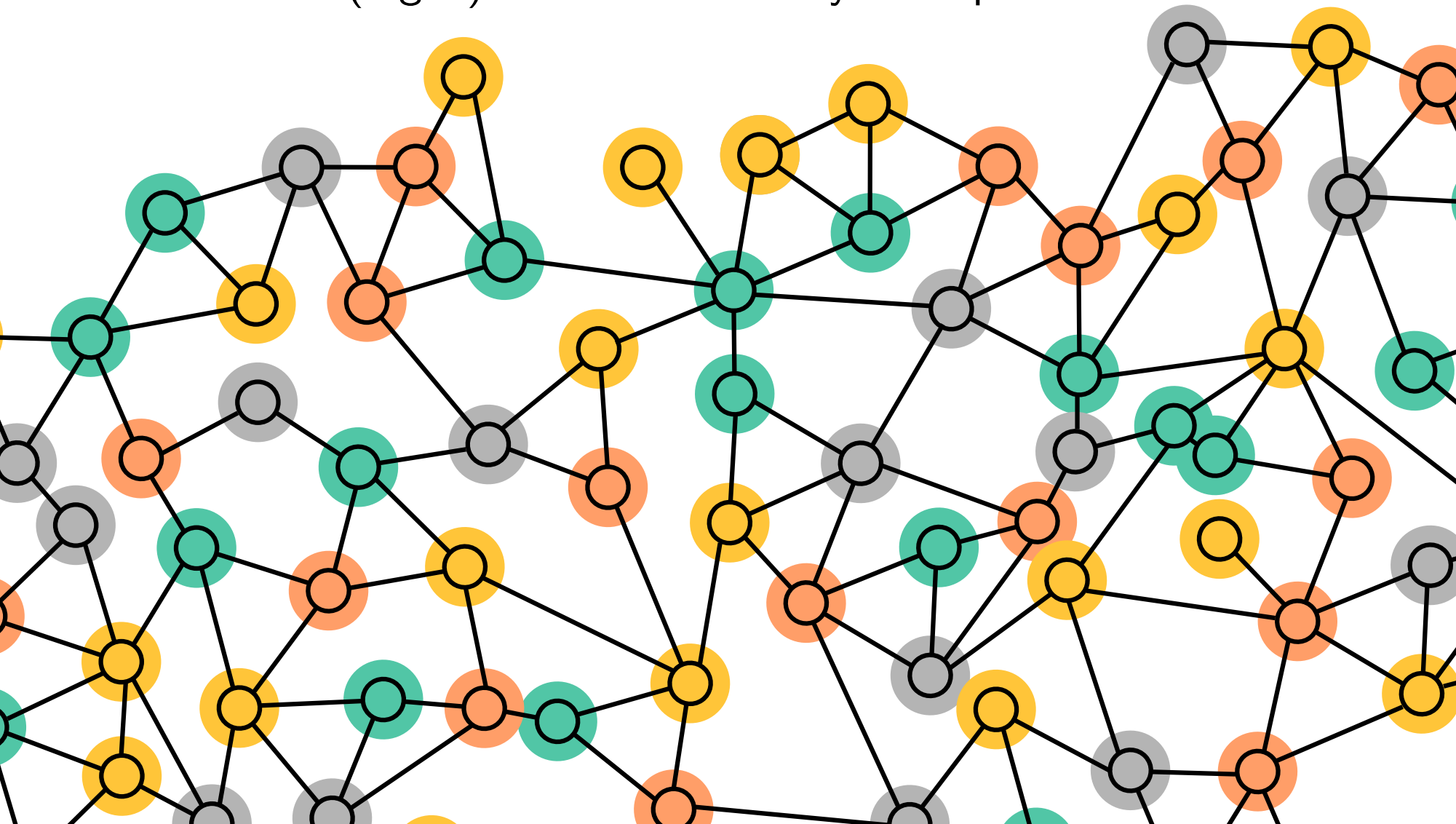
Colouring

Task : Colouring a bounded-degree graphs
Solved in $O(\log^* n)$ rounds $\rightarrow$ a really local problem.



Colouring

Task : Colouring a bounded-degree graphs
Solved in $O(\log^* n)$ rounds $\rightarrow$ a really local problem.



Checking solutions

Scenario 1 : given a black box solution, verify it.

Scenario 2 : Self-stabilizing algorithm : all the memory maybe corrupted.
Silence property : after stabilization, machines states do not change.

Checking solutions

Scenario 1 : given a black box solution, verify it.

Scenario 2 : Self-stabilizing algorithm : all the memory maybe corrupted.
Silence property : after stabilization, machines states do not change.

Certification

We could recompute and compare.

$\rightarrow$ No, little resources, **constant time**.

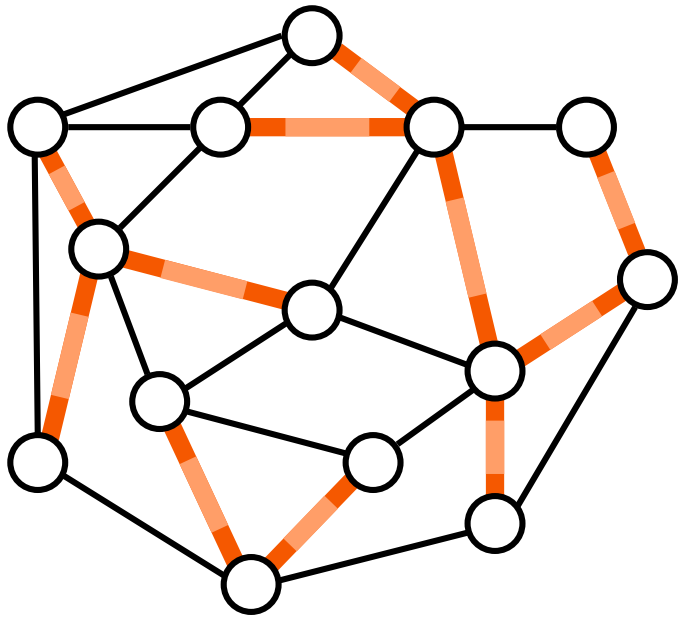
In our problem list :

- checking a colouring : ok !
- checking a tree : nope
- checking an MST : nope

Need help : extra information, called *certificate*.

Goal : minimize certificate size
under the constraint of local verification.

Certification



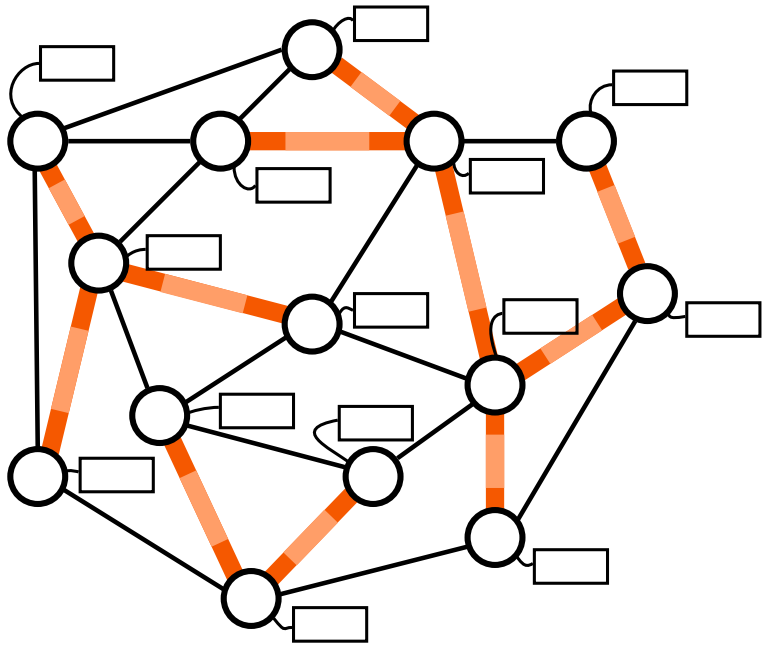
(NP-flavour + veto) rule :

For all instance :

- If correct :
  $\exists c$ s.t. every node accepts.
- If not correct :
  $\forall c$, there exists a node that rejects.

Key words : proofs, certificates, locally checkable proofs, proof-labeling schemes.

Certification



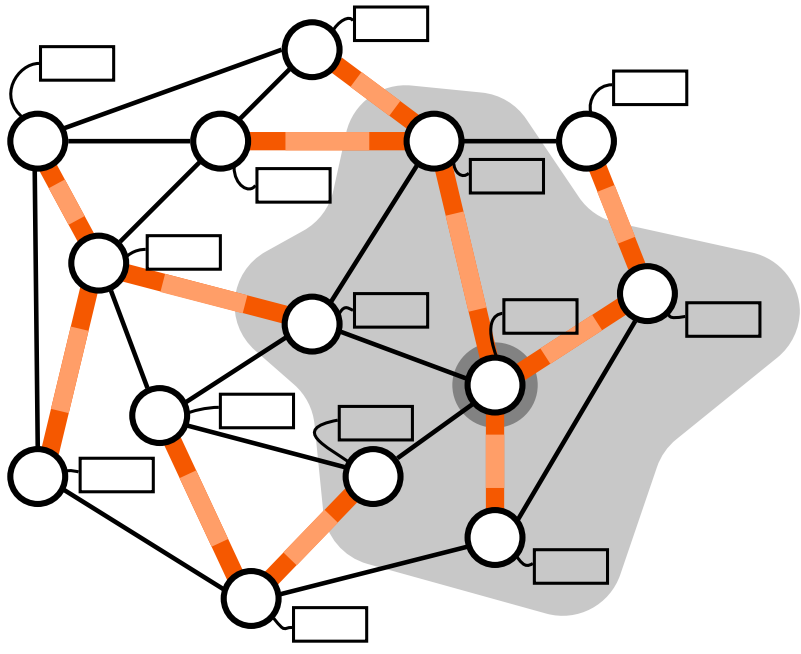
(NP-flavour + veto) rule :

For all instance :

- If correct :
  $\exists c$ s.t. every node accepts.
- If not correct :
  $\forall c$, there exists a node that rejects.

Key words : proofs, certificates, locally checkable proofs, proof-labeling schemes.

Certification



(NP-flavour + veto) rule :

For all instance :

- If correct :
  $\exists c$ s.t. every node accepts.
- If not correct :
  $\forall c$, there exists a node that rejects.

Key words : proofs, certificates, locally checkable proofs, proof-labeling schemes.

Colouring I

Inputs : Colours on the nodes.
Property to check : the graph is properly coloured.

Solution : proofs of "size 0",
every node checks its neighbours'colours.

Every node accepts $\Leftrightarrow$ no edge is monochromatic.

Colouring II

Input : selected edges
Property to check : the subgraph of selected edges is 3-colourable

Solution : Give the colours as certificates. Size : $O(1)$.

Graph is 3-colourable
$\Leftrightarrow$ there exists a proper 3-colouring.
$\Leftrightarrow$ there exist colours, with no monochromatic edge.

Minimum spanning tree

Input : selected edges and weights
Property to check : the selected edges form an MST.

General trick : If you have an algorithm, you can write the whole transcript (all messages).

The instance is good
$\Leftrightarrow$ there exists an algorithm that builds it
$\Leftrightarrow$ there exists a correct transcript

$\rightarrow$ But we want the smallest space possible.

Spanning tree

Input : selected edges
Property to check : the selected edges form a spanning tree.

Transcript :
1. Send 32 to 5 and 18, Receive 5 from 5, 18 from 18.$\rightarrow$ 5, 5.
2. Send 5 to 18. Receive 4 from 18. $\rightarrow$ 4, 18.
3. Send 4 to 5. Receive nothing. $\rightarrow$ 4, 18.
4. Send nothing. Receive 2 from 18. $\rightarrow$ 2, 18.

...

.

Spanning tree

Input : selected edges
Property to check : the selected edges form a spanning tree.

Transcript :
1. Send 32 to 5 and 18, Receive 5 from 5, 18 from 18.$\rightarrow$ 5, 5.
2. Send 5 to 18. Receive 4 from 18. $\rightarrow$ 4, 18.
3. Send 4 to 5. Receive nothing. $\rightarrow$ 4, 18.
4. Send nothing. Receive 2 from 18. $\rightarrow$ 2, 18.

...

Spanning tree

Input : selected edges
Property to check : the selected edges form a spanning tree.

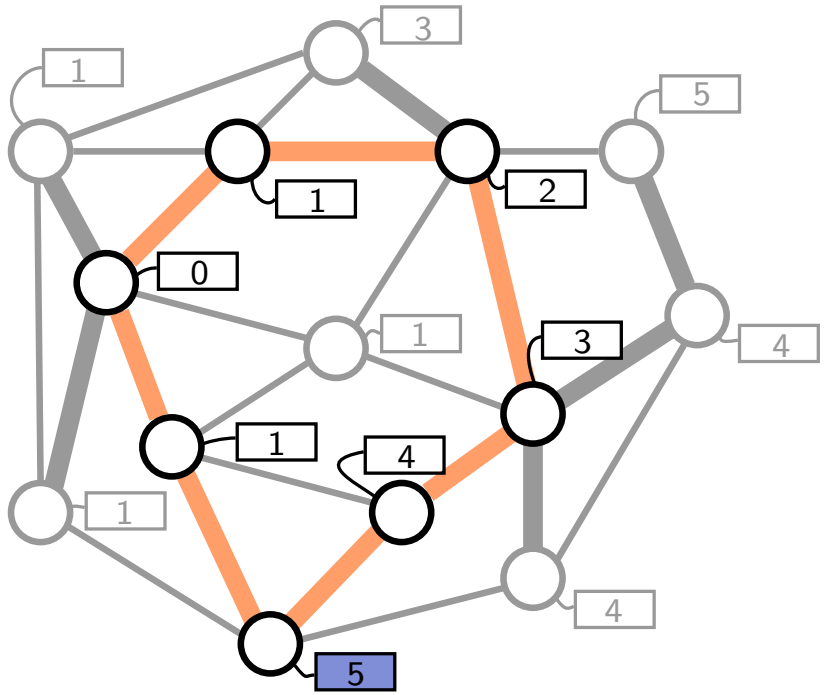
Transcript :
1. Send 32 to 5 and 18, Receive 5 from 5, 18 from 18.$\rightarrow$ 5, 5.
2. Send 5 to 18. Receive 4 from 18. $\rightarrow$ 4, 18.
3. Send 4 to 5. Receive nothing. $\rightarrow$ 4, 18.
4. Send nothing. Receive 2 from 18. $\rightarrow$ 2, 18.

...

Solution : (parent, distance, ID of the root).

Spanning tree

Theorem : (parent, distance, ID of the root) is a correct scheme, it has size $O(\log n)$ and it's optimal.



Minimum spanning tree

Input : selected edges and weights
Property to check : the selected edges form an MST.

Theorem The only information needed is, for each merge :

- the endpoint of the merging edge
- its weight
- a spanning tree that points to the edge.

This scheme has size $O(\log^2 n)$ and it's optimal.

Topic 1 : more bounds

Research topic 1 : Bounds for more problems.

Recent works :

- *Approximate proof-labeling schemes*,
  Censor-Hillel, Paz, Perry. 2017.
  $\rightarrow$ Diameter, matching, new technique from communication complexity.

Favourite open problem :
Simple proof of the $\Omega(\log^2 n)$ bound for MST.

Topic 2 : complexity theory

Research topic 2 : Complexity theory

Historical note : Started around 2010 for good reasons.
Another branch for building LCL problems.

Recent works :

- *What can be verified locally ?* and
  A hierarchy of local decision
  Balliu, Olivetti, Fraigniaud, Feuilloley, Hirvonen. 2016-17.
  $\rightarrow$ Analogues of polynomial hierarchy.
- *Interactive Distributed Proofs*.
  Kol, Oshman, Saxena. 2018.
  $\rightarrow$ Analogues of IP, AM etc.

Favourite open problem : Separating polynomial hierarchy.

Topic 3 : bridging gaps

Research topic 3 : understanding similarity and differences between the different local-style models.

Recent works :

- *Error-sensitive proof-labeling schemes*
  Feuilloley, Fraigniaud, 2017.
  $\rightarrow$ property testing flavour. See also dist. property testing.
- *Certification of compact low-stretch routing schemes*
  Balliu, Fraigniaud. 2017
- Work in progress about graph classes.
- Links to communication complexity

Favourite open problem :
Relating decision and construction (?)

Topic 4 : challenge model

Research topic 4 : challenge the model, understand it better.

Recent works :

- *PLS : broadcast, unicast and in between*
  Patt-Shamir, Perry. 2017.
  $\rightarrow$ Focus on messages.
- *Space-time tradeoffs for distributed verification*,
  Redundancy in distributed proofs
  Local verification of global proofs
  Ostrovsky, Perry, Rosenbaum, Paz, Fraigniaud, Feuilloley, Hirvonen. 2017-18.
  $\rightarrow$ Change access to proof, redundancy
- also, error-sensitive PLS.

Favourite open problem : bipartite with global proofs.

Topic 5 : back to self-stab.

Research topic 5 : go back to the difficult model, self-stabilization.

Recent works :

- *Locally-Iterative Distributed ($\Delta+1$)-Coloring below Szegedy-Vishwanathan Barrier, and Applications to Self-Stabilization and to Restricted-Bandwidth*. Barenboim, Elkin, Goldenberg 2018.
- ...

Favourite open problem : what's going on ? ?