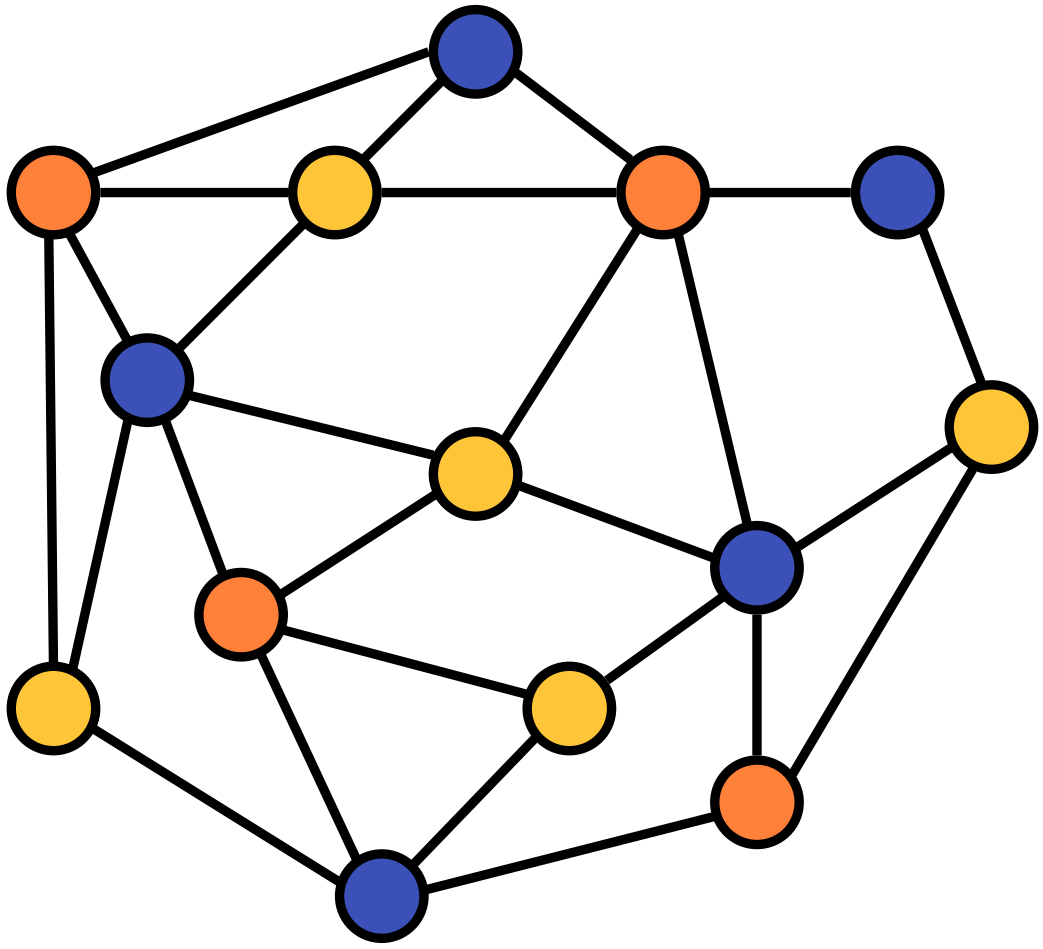
Error-sensitive proof-labeling schemes

Laurent Feuilloley
joint work with Pierre Fraigniaud
Université Paris Diderot

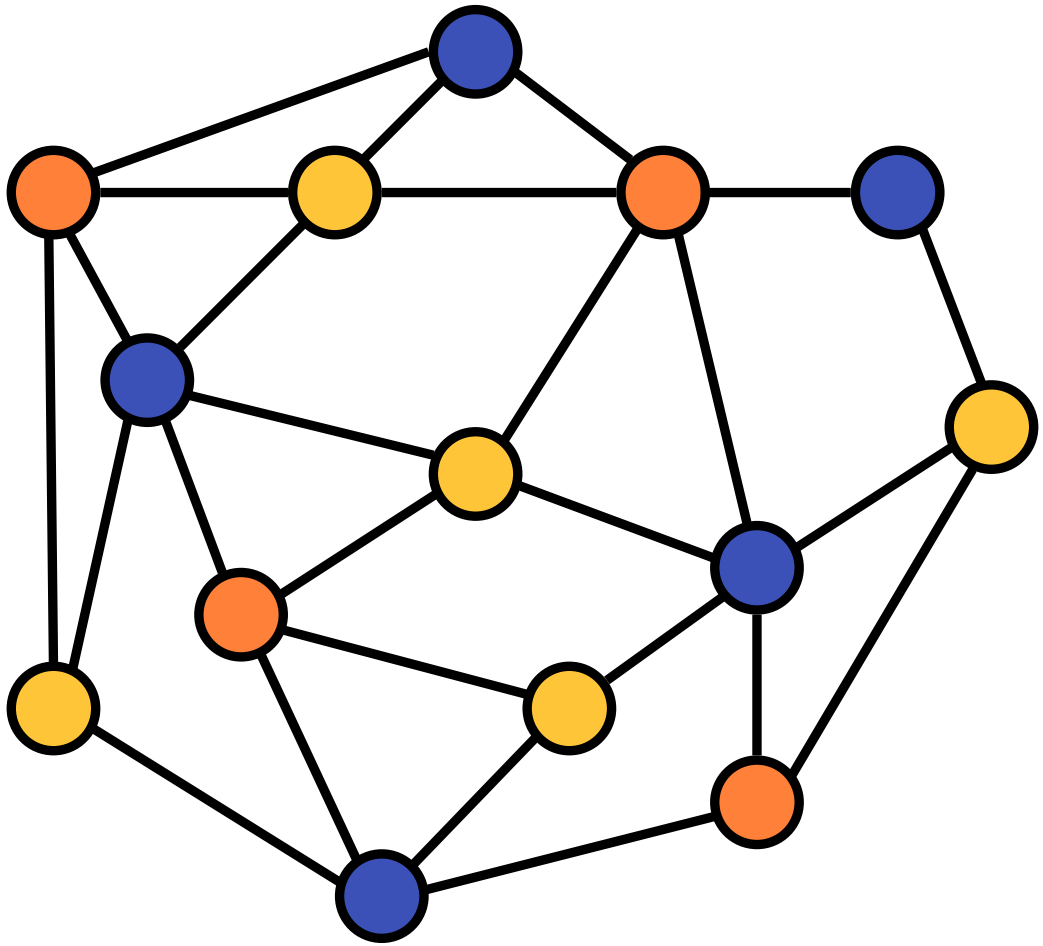
HALG · June 2018 · Amsterdam

originally presented at DISC 2017

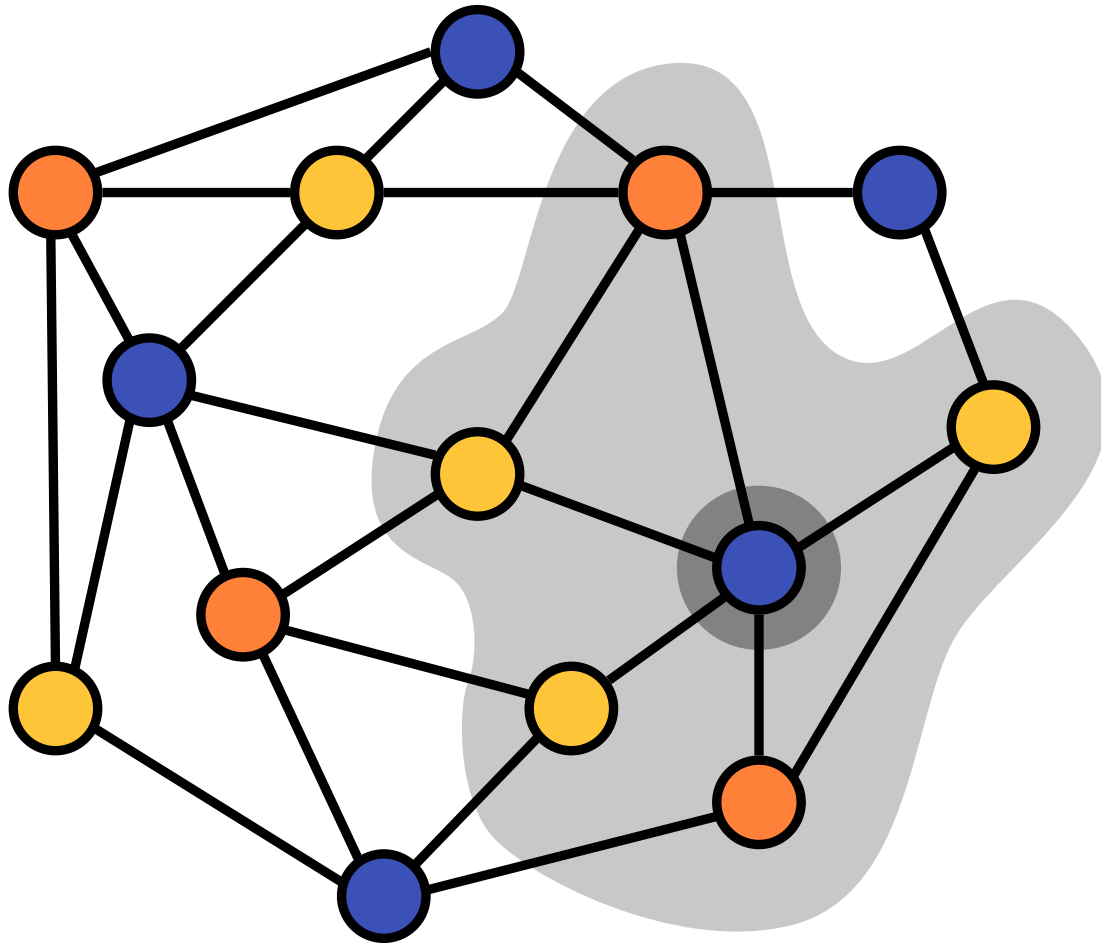
Is it properly colored ?



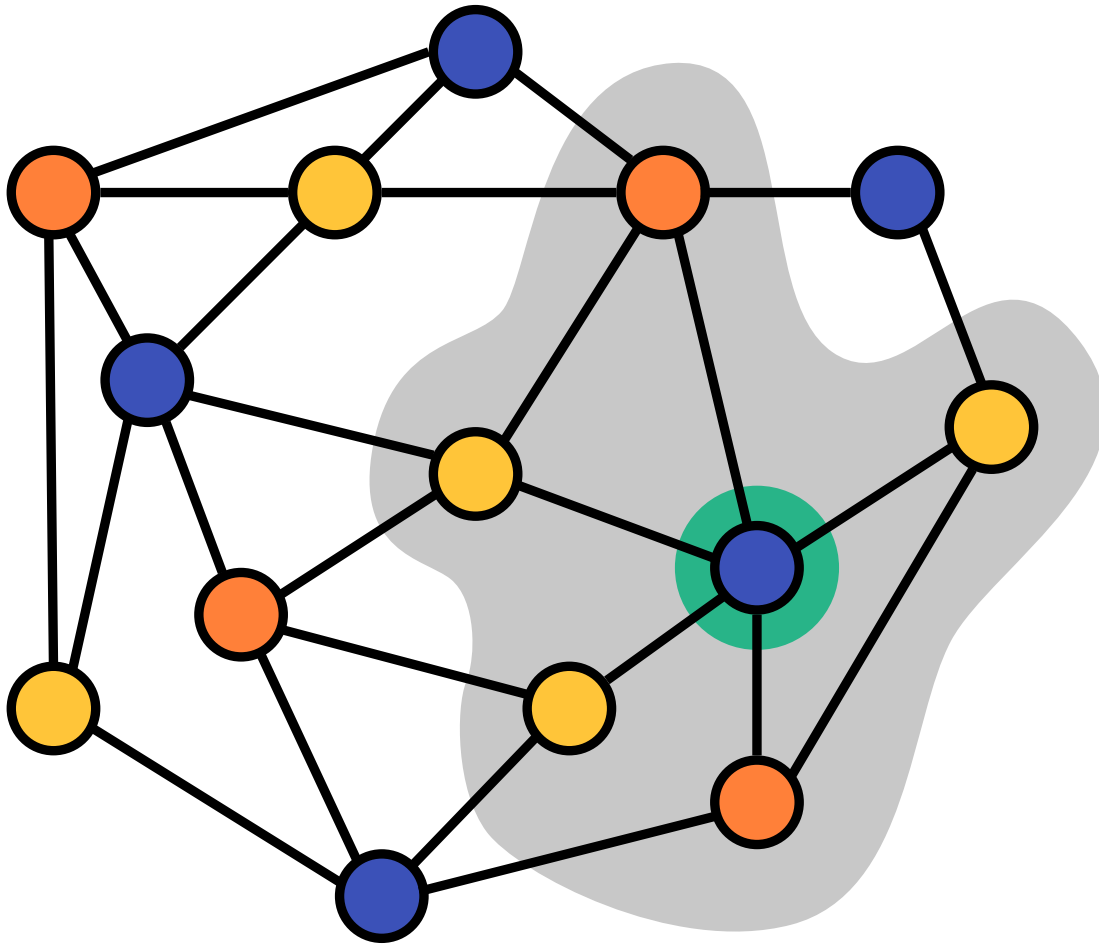
Let the nodes decide...



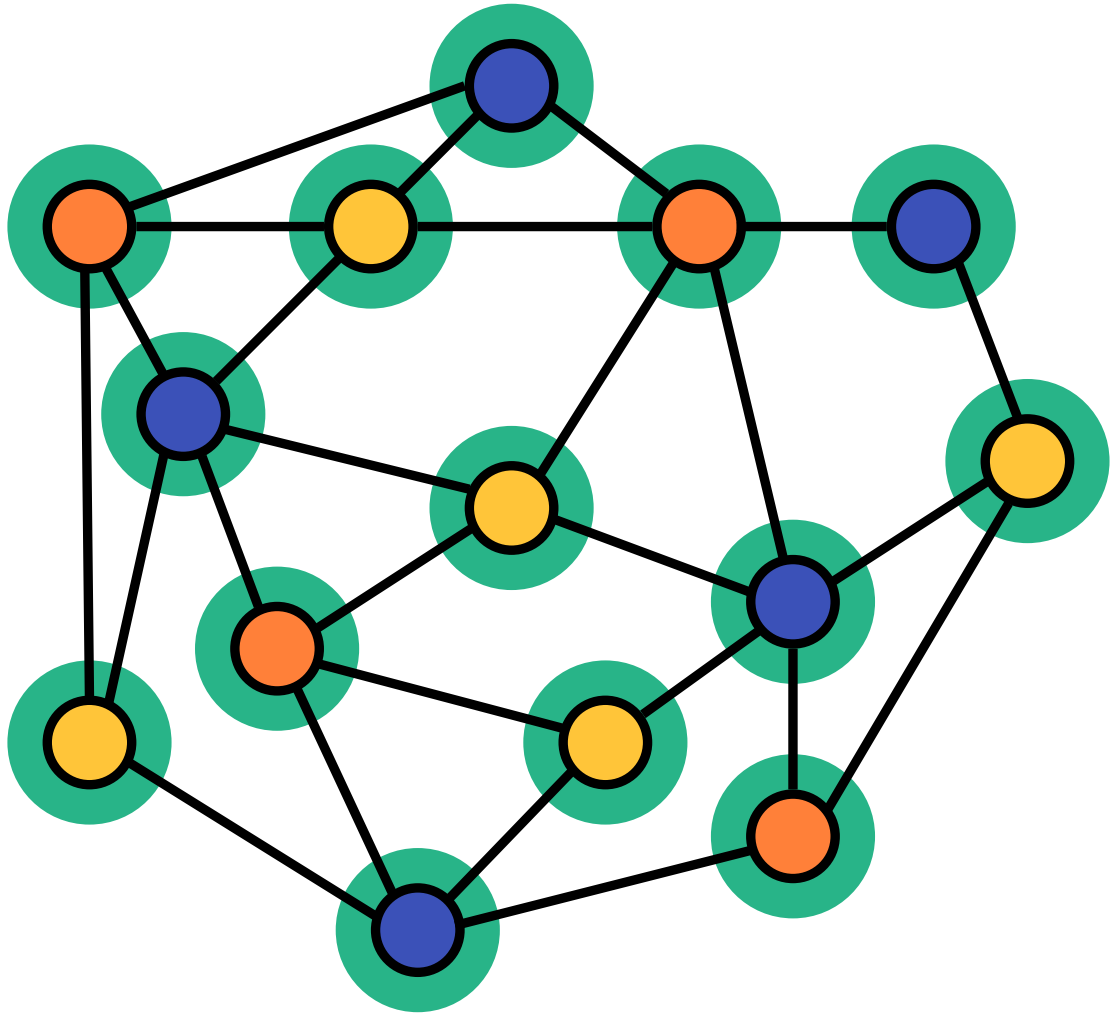
(1) Take a look



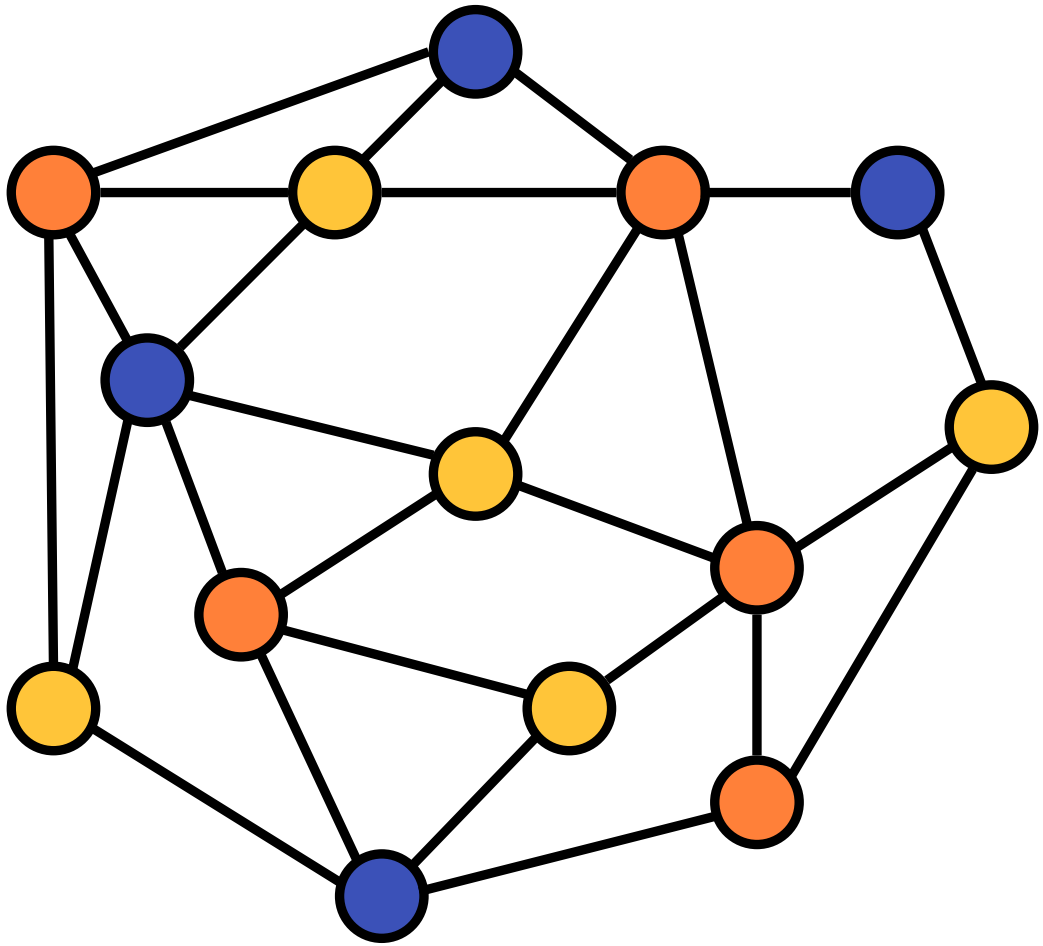
(2) Output a decision



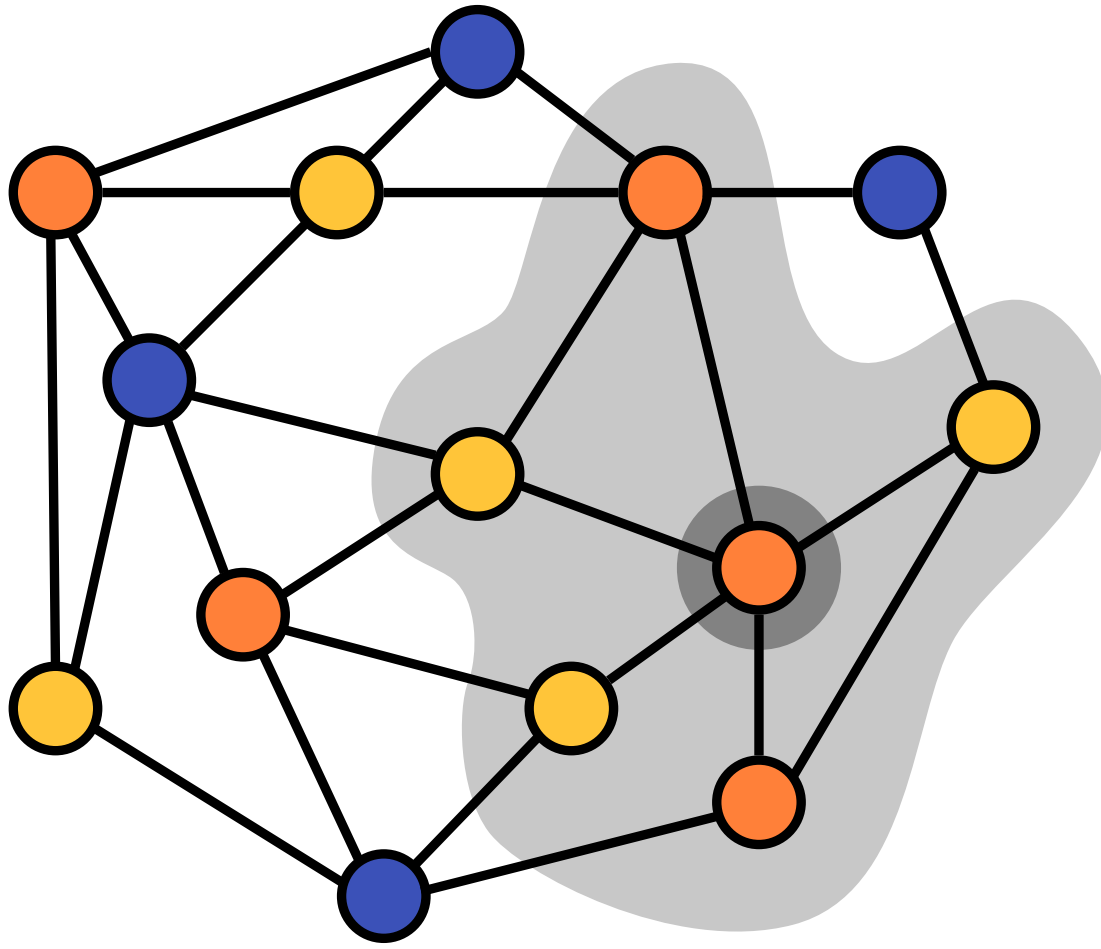
Uniform accept



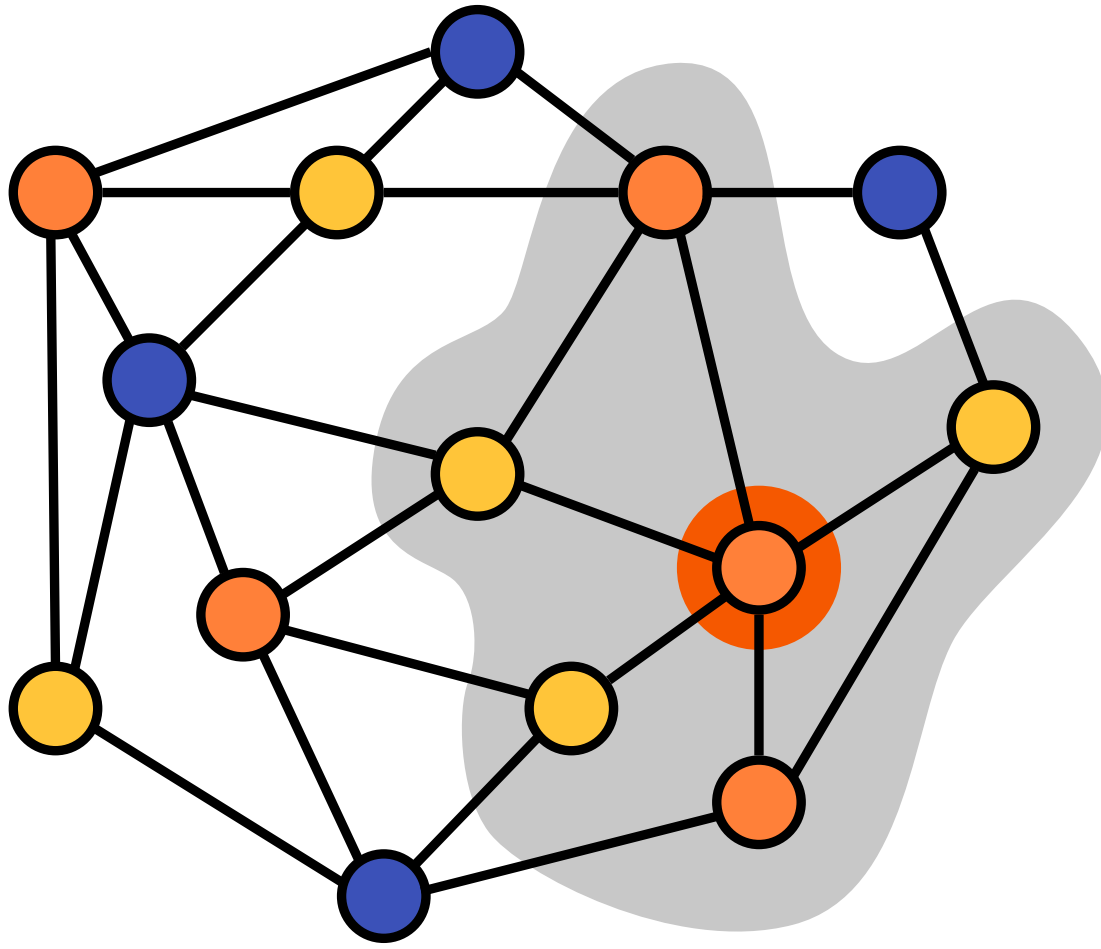
On a bad instance



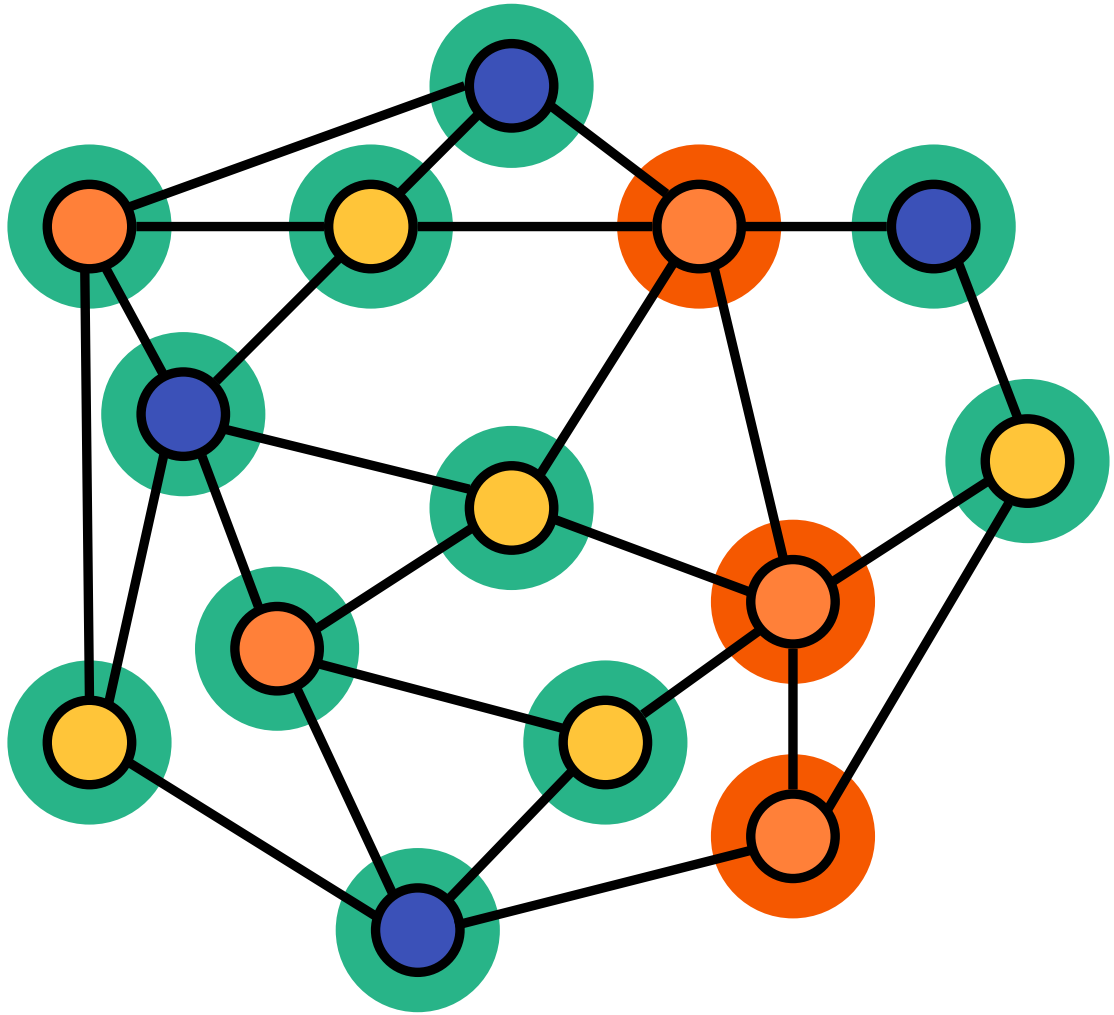
(1) Take a look



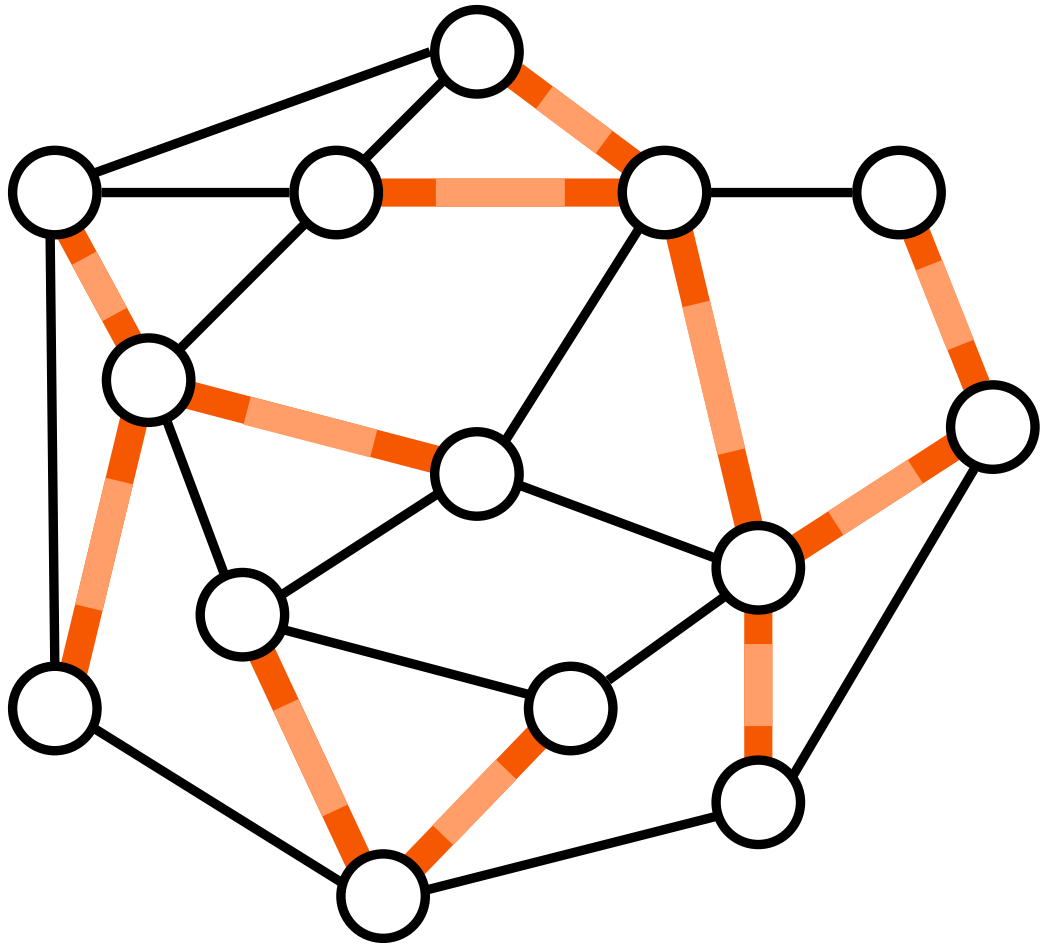
(2) Output a decision



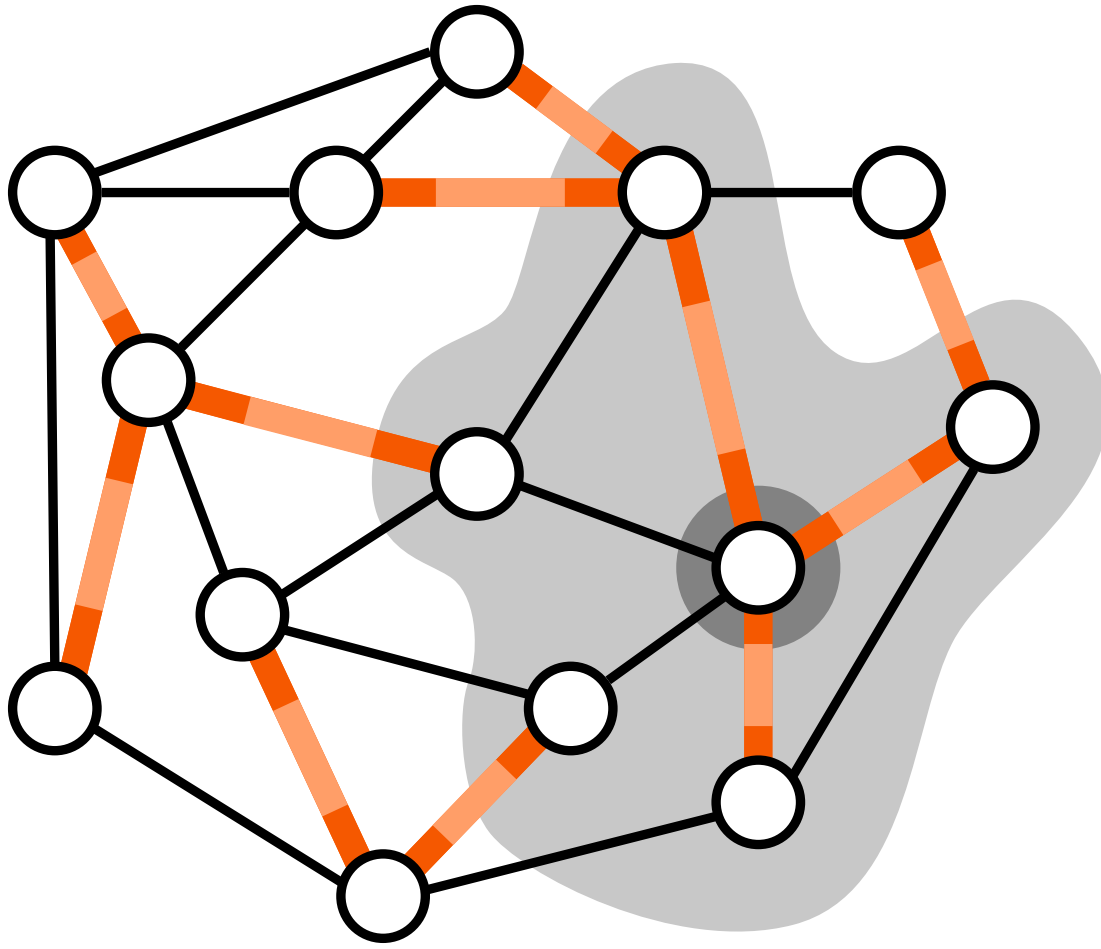
Reject by veto



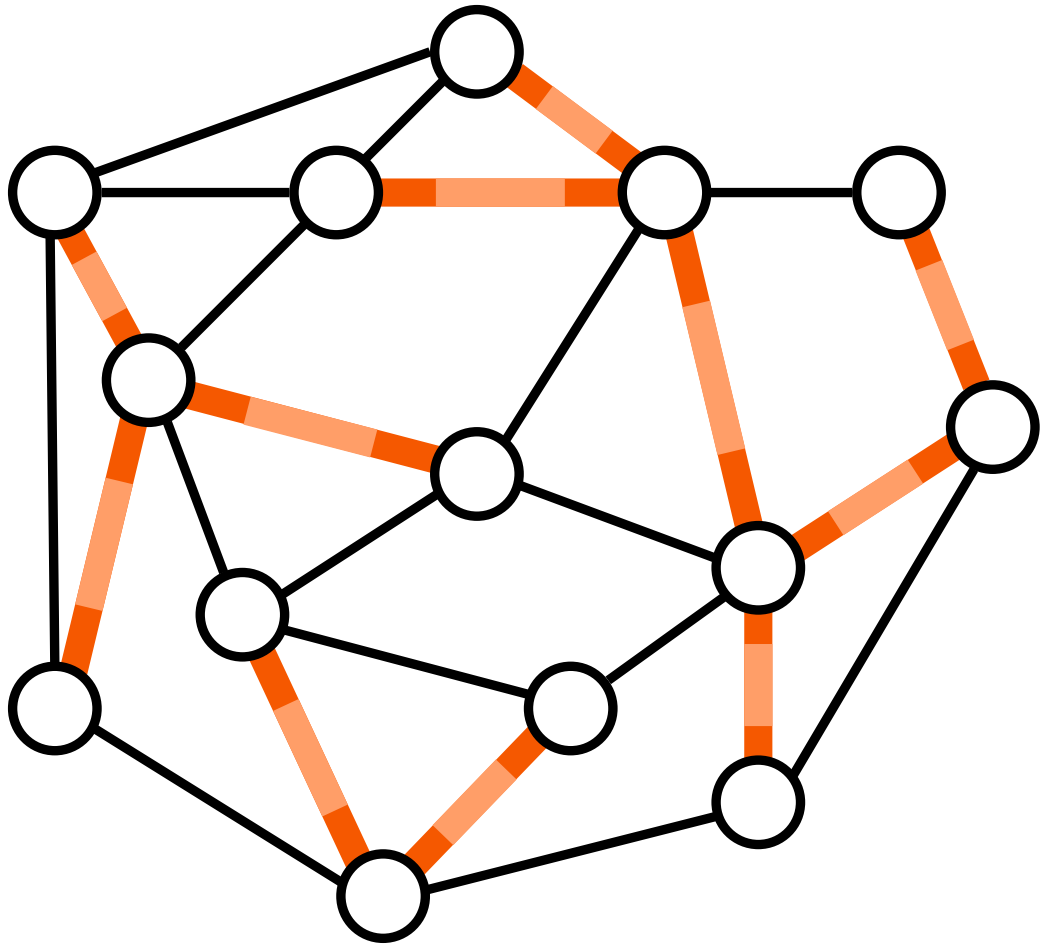
And spanning forest ?



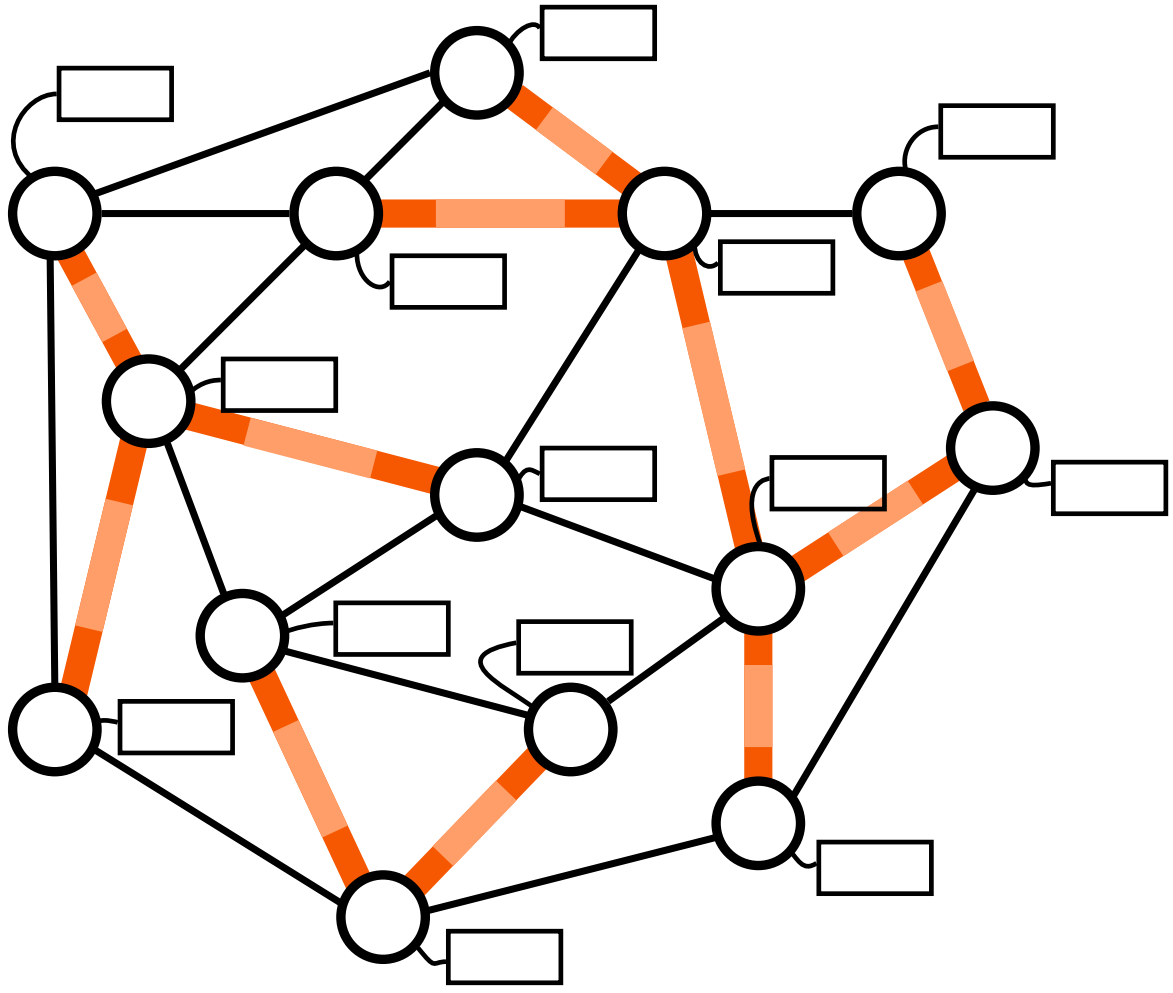
Not so easy...



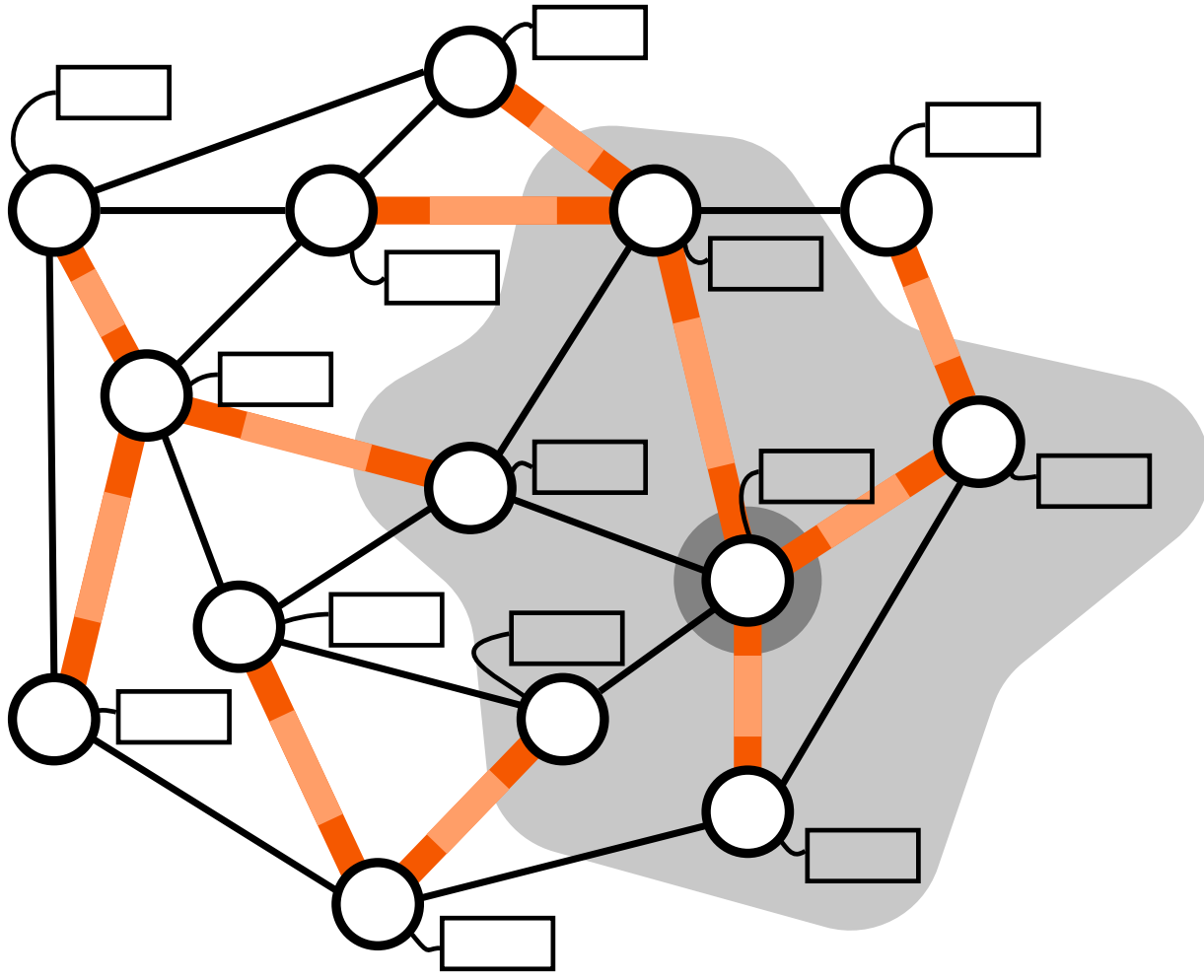
Let's ask a 'friend'



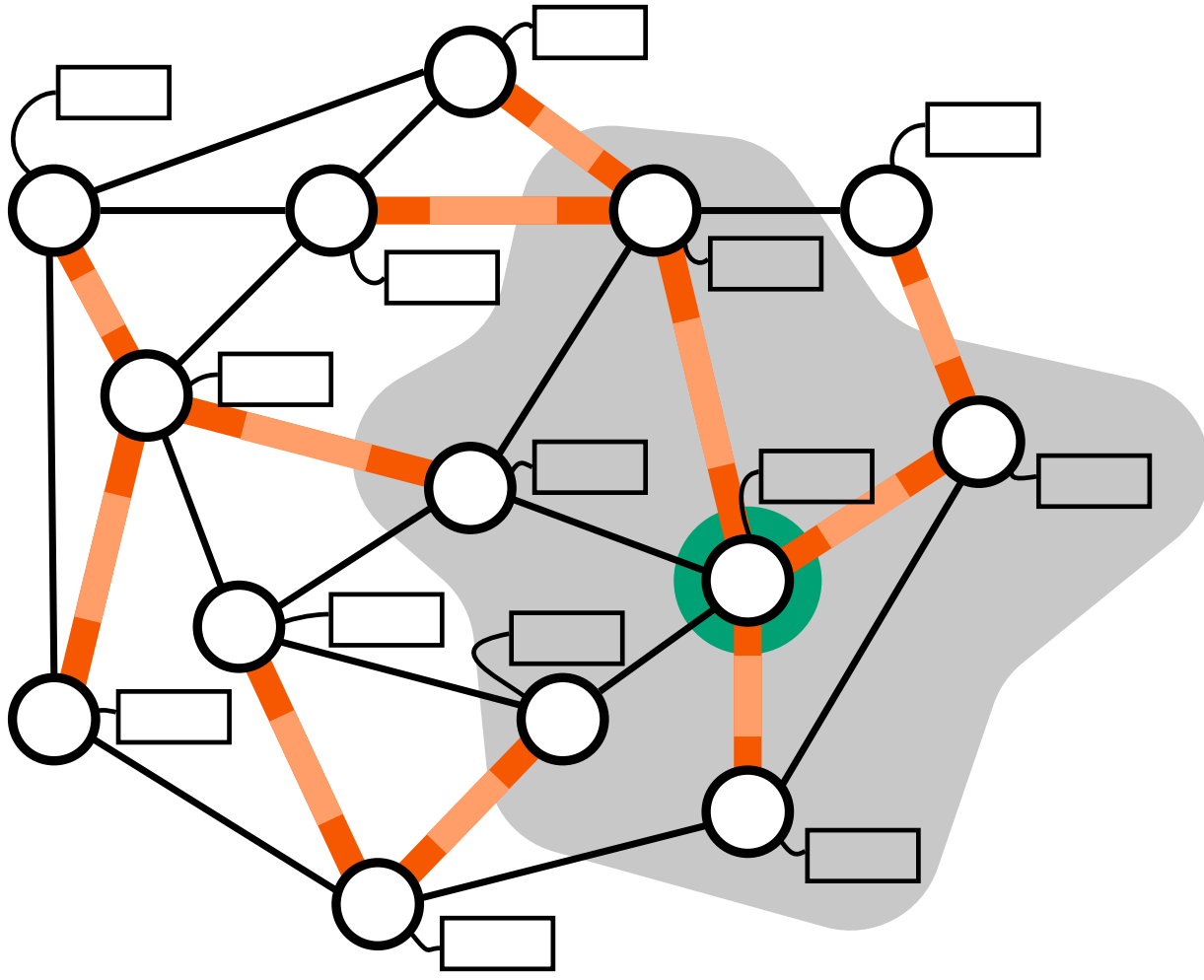
Let's ask a 'friend'



Let's ask a 'friend'



Let's ask a 'friend'



(A slide with text)

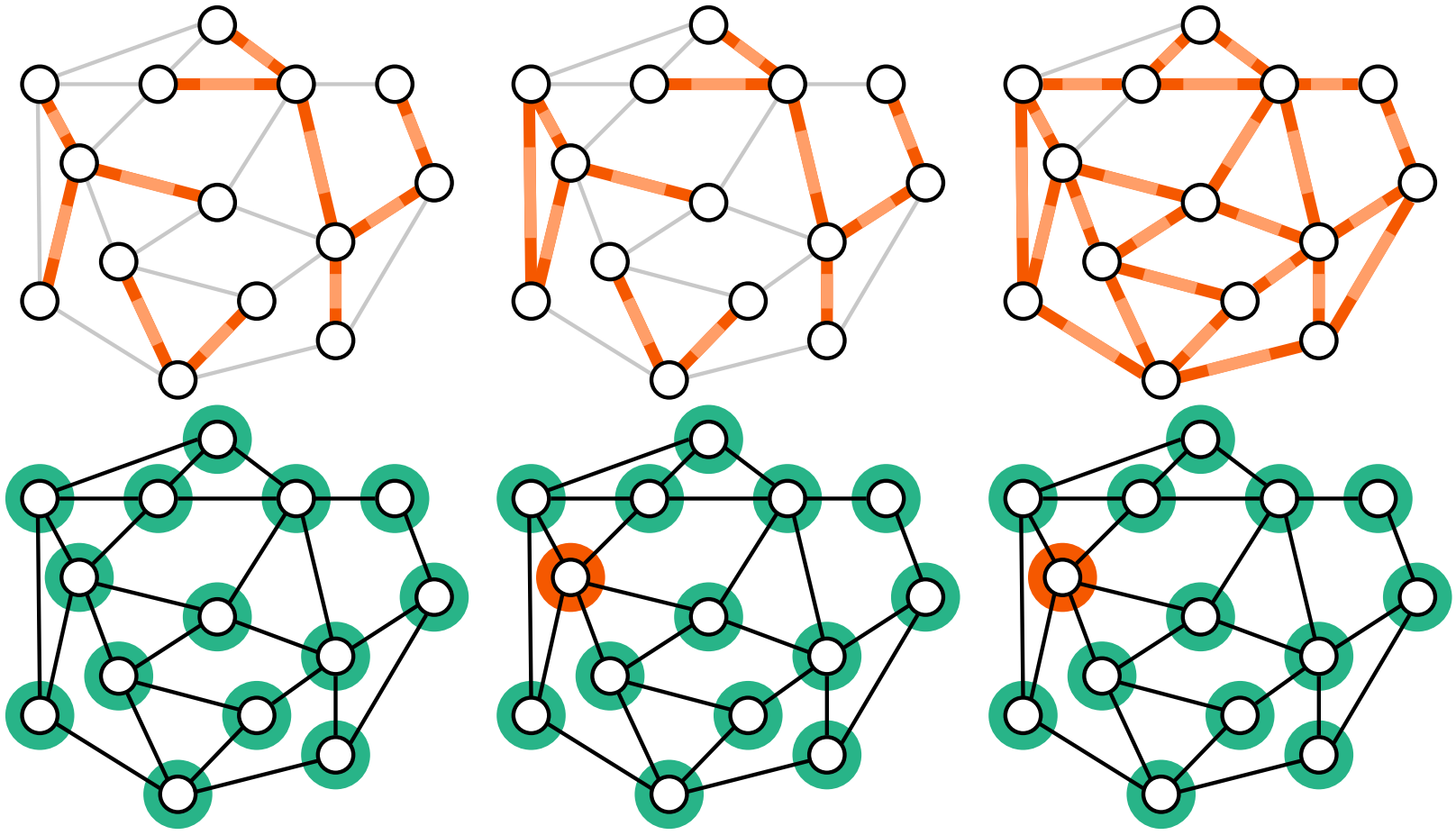
For every labeled graph :

- If it is good :

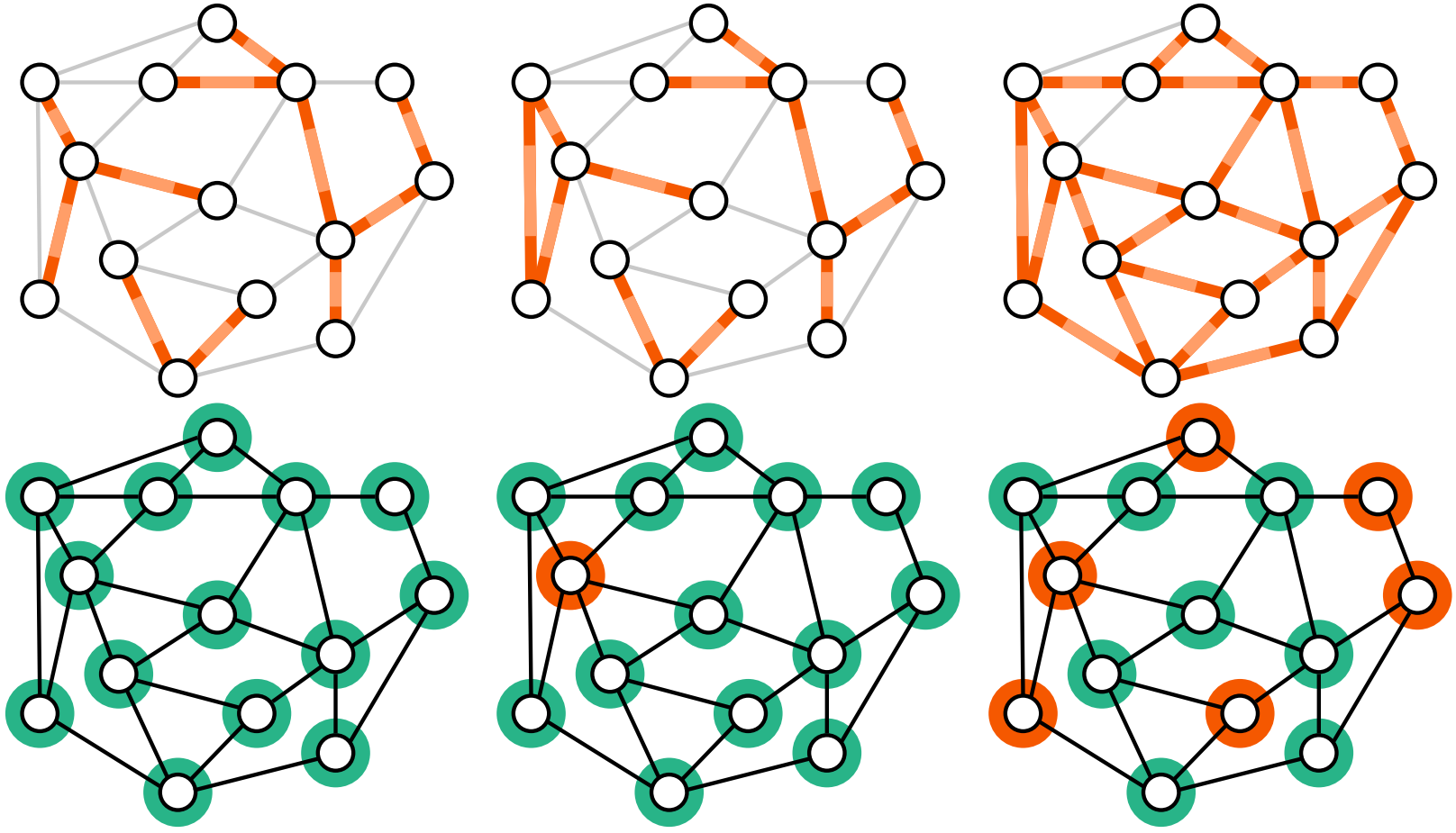
  $\exists$ certificate assignment s.t. every node accepts
- If it is bad :

  $\forall$ certificate assignment, at least one node rejects.

Three instances

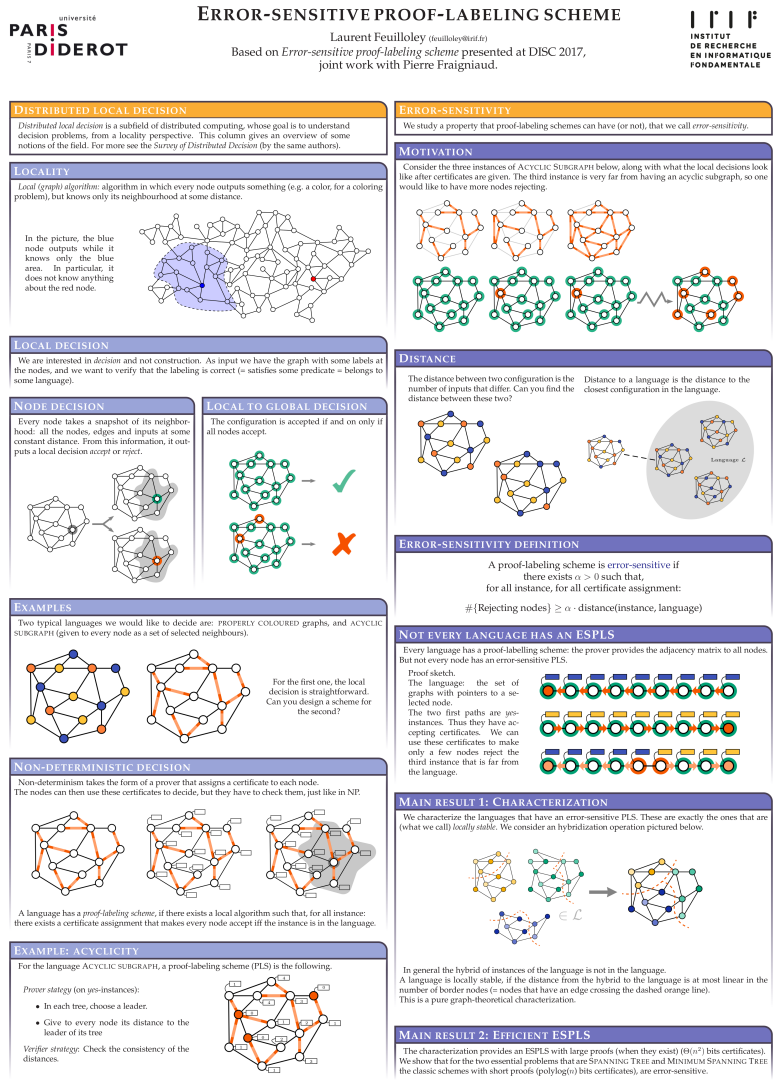


How (in)sensitive ?



See you at the poster session !

The poster looks like this →



ERROR-SENSITIVE PROOF-LABELING SCHEME

Laurent Feuilloley (feuilloley@irif.fr)
Based on *Error-sensitive proof-labeling scheme* presented at DISC 2017, joint work with Pierre Fraigniaud.

Université Paris Diderot

INSTITUT DE RECHERCHE EN INFORMATIQUE FONDAMENTALE

DISTRIBUTED LOCAL DECISION

Distributed local decision is a subfield of distributed computing, whose goal is to understand decision problems, from a locality perspective. This column gives an overview of some notions of the field. For more see the *Survey of Distributed Decision* (by the same authors).

LOCALITY

Local (graph) algorithm: algorithm in which every node outputs something (e.g. a color, for a coloring problem), but knows only its neighbourhood at some distance.

In the picture, the blue node outputs while it knows only the blue area. In particular, it does not know anything about the red node.

LOCAL DECISION

We are interested in *decision* and not construction. As input we have the graph with some labels at the nodes, and we want to verify that the labeling is correct (= satisfies some predicate = belongs to some language).

NODE DECISION

Every node takes a snapshot of its neighborhood: all the nodes, edges and inputs at some constant distance. From this information, it outputs a local decision *accept* or *reject*.

LOCAL TO GLOBAL DECISION

The configuration is accepted if and on only if all nodes accept.

EXAMPLES

Two typical languages we would like to decide are: PROPERLY COLOURED graphs, and ACYCLIC SUBGRAPH (given to every node as a set of selected neighbours).

For the first one, the local decision is straightforward. Can you design a scheme for the second?

NON-DETERMINISTIC DECISION

Non-determinism takes the form of a prover that assigns a certificate to each node.
The nodes can then use these certificates to decide, but they have to check them, just like in NP.

A language has a *proof-labeling scheme*, if there exists a local algorithm such that, for all instance: there exists a certificate assignment that makes every node accept iff the instance is in the language.

EXAMPLE: ACYCLICITY

For the language ACYCLIC SUBGRAPH, a proof-labeling scheme (PLS) is the following.

Prover strategy (on *yes*-instances):

- In each tree, choose a leader.
- Give to every node its distance to the leader of its tree

Verifier strategy: Check the consistency of the distances.

ERROR-SENSITIVITY

We study a property that proof-labeling schemes can have (or not), that we call *error-sensitivity*.

MOTIVATION

Consider the three instances of ACYCLIC SUBGRAPH below, along with what the local decisions look like after certificates are given. The third instance is very far from having an acyclic subgraph, so one would like to have more nodes rejecting.

DISTANCE

The distance between two configuration is the number of inputs that differ. Can you find the distance between these two?

Distance to a language is the distance to the closest configuration in the language.

ERROR-SENSITIVITY DEFINITION

A proof-labeling scheme is error-sensitive if
there exists $\alpha > 0$ such that,
for all instance, for all certificate assignment:

$$\#\{\text{Rejecting nodes}\} \geq \alpha \cdot \text{distance(instance, language)}$$

NOT EVERY LANGUAGE HAS AN ESPLS

Every language has a proof-labelling scheme: the prover provides the adjacency matrix to all nodes. But not every node has an error-sensitive PLS.

Proof sketch.
The language: the set of graphs with pointers to a selected node.
The two first paths are *yes*-instances. Thus they have accepting certificates. We can use these certificates to make only a few nodes reject the third instance that is far from the language.

MAIN RESULT 1: CHARACTERIZATION

We characterize the languages that have an error-sensitive PLS. These are exactly the ones that are (what we call) *locally stable*. We consider an hybridization operation pictured below.

In general the hybrid of instances of the language is not in the language.
A language is locally stable, if the distance from the hybrid to the language is at most linear in the number of border nodes (= nodes that have an edge crossing the dashed orange line).
This is a pure graph-theoretical characterization.

MAIN RESULT 2: EFFICIENT ESPLS

The characterization provides an ESPLS with large proofs (when they exist) ($\Theta(n^2)$ bits certificates). We show that for the two essential problems that are SPANNING TREE and MINIMUM SPANNING TREE the classic schemes with short proofs (polylog(n) bits certificates), are error-sensitive.