Error-sensitive proof-labeling schemes

Laurent Feuilloley
joint work with Pierre Fraigniaud
Université Paris Diderot

October 2017
Locality
Decision problems
Decision problems

Context:
- Communication graph $G$
- Node inputs, $x : v \mapsto x(v)$

A language is a set of configurations $(G, x)$
s.t. $\forall G, \exists x, (G, x) \in \mathcal{L}$
Properly-colored graphs

\[ \mathcal{L} = \{(G, x) \text{ s.t. } x \text{ is a proper coloring of } G\} \]
Spanning forest

\[ \mathcal{L} = \{(G, x) \text{ s.t. } x \text{ describes a spanning forest of } G\} \]
Spanning forest

\[ \mathcal{L} = \{(G, x) \text{ s.t. } x \text{ describes a spanning forest of } G\} \]
Decision mechanism

Every node:

- gathers its 1-neighbourhood
- outputs a local decision accept or reject.
Decision mechanism

Every node:
• gathers its 1-neighbourhood
• outputs a local decision: accept or reject.
Every node:
• gathers its 1-neighbourhood
• outputs a local decision accept or reject.
(G, x) is accepted if all nodes accept.
Decision mechanism

\((G, x)\) is rejected if at least one node rejects.
Properly-colored graphs
Properly-colored graphs
Properly-colored graphs
Spanning forest
Spanning forest
Proof-labeling schemes

Distributed
non-determinism
Proof-labeling schemes
Proof-labeling schemes
Proof-labeling schemes
Proof-labeling schemes

Given a proof-labeling scheme for $\mathcal{L}$:

For all $(G, x)$:

- If $(G, x) \in \mathcal{L}$:
  \[ \exists c \text{ s.t. } (G, x, c) \text{ is accepted}. \]

- If $(G, x) \notin \mathcal{L}$:
  \[ \forall c, (G, x, c) \text{ is rejected}. \]
Proof-labeling schemes
Proof-labeling schemes
Proof-labeling schemes
Proof-labeling schemes
Error-sensitivity of proof-labeling schemes
One node to reject
More nodes to reject
Distance

\[ d((G, x_1), (G, x_2)) = \#\{ v : x_1(v) \neq x_2(v) \} \]
Distance

\[ d((G, x_1), (G, x_2)) = \#\{v : x_1(v) \neq x_2(v)\} \]
Distance

\[ d\left((G, x), \mathcal{L}\right) = \min_{(G', x') \in \mathcal{L}} d\left((G, x), (G', x')\right) \]
Error-sensitivity

in words

A PLS is *error-sensitive* if the number of rejecting nodes grows linearly with the distance.
Error-sensitivity

with a formula

A PLS is error-sensitive if there exists $\alpha > 0$ s.t., for all $(G, x)$, for all certificate:

$$\#\{\text{Rejecting nodes}\} \geq \alpha \cdot d((G, x), L)$$
Examples
# Acyclicity problems

<table>
<thead>
<tr>
<th>Adjacency lists</th>
<th>Pointers</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Spanning forest" /></td>
<td><img src="image2" alt="Spanning forest with pointers" /></td>
</tr>
<tr>
<td><img src="image3" alt="Spanning tree" /></td>
<td><img src="image4" alt="Spanning tree with pointers" /></td>
</tr>
</tbody>
</table>
Spanning forest with pointers
Spanning forest with pointers
Spanning forest with pointers
Spanning forest with pointers
Spanning forest with pointers
Spanning forest with pointers
Spanning forest with pointers
Spanning forest with pointers has an error-sensitive PLS.
Spanning tree with pointers
Spanning tree with pointers
Spanning tree with pointers
Spanning tree with pointers
Spanning tree with pointers
Spanning tree with pointers
Spanning tree with pointers has no error-sensitive PLS (for any certificate size).
Structural characterization
A language $\mathcal{L}$ admits an error-sensitive PLS if and only if $\mathcal{L}$ is locally stable.
Local stability

Hybridization

$\in \mathcal{L}$
Local stability

Boundary nodes
Local stability

\[ \mathcal{L} \text{ is locally stable if:} \]

\[ \exists \beta, \forall G, \forall \text{ hybridization,} \]

\[ d(\text{hybrid, } \mathcal{L}) \leq \beta \cdot \# \{ \text{Boundary nodes} \} \]
Spanning tree with pointers is not locally stable.
But with adjacency lists...

**Thm**: With *adjacency lists*, spanning tree and minimum spanning tree, are locally stable.

⇒ they have error-sensitive PLS.
Compact schemes
Compact PLS

**Theorem** (Korman et al.):
- ST has a $O(\log n)$-PLS;
- MST has a $O(\log^2 n)$-PLS.
Compact PLS

New Theorem:

- ST has a $O(\log n)$-ESPLS;
- MST has a $O(\log^2 n)$-ESPLS.
Open problem

Does error-sensitivity always come for free (when achievable)?