Space-Time Tradeoffs for Distributed Verification

Mor Baruch[1] Rafail Ostrovsky[2] **Will Rosenbaum**[2]

[1]Tel Aviv University

[2]UCLA

July 28, 2016

Distributed Verification

Problem

Given a distributed network $G = (V, E)$ and states $\varphi(v) \in S$ for each $v \in V$, determine if the **graph configuration** (G, φ) satisfies a boolean predicate P.

Distributed Verification

Problem

Given a distributed network $G = (V, E)$ and states $\varphi(v) \in S$ for each $v \in V$, determine if the **graph configuration** (G, φ) satisfies a boolean predicate P.

Examples of distributed verification problem include:

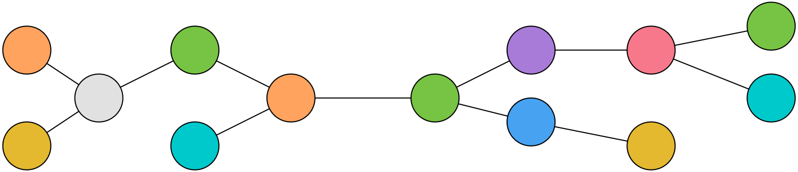
- acyclicity checking: $S = \varnothing$ and P indicates that G is cycle-free.
- proper coloring: $\varphi : V \to S$ is coloring of the vertices of G and P indicates that the coloring is proper (adjacent vertices have different colors).
- spanning tree: φ defines a set of incident edges for each vertex, and P indicates if the subgraph defined by the edges is a spanning tree.
- isomorphism: P indicates that (G, φ) is isomorphic to some fixed graph configuration (H, ψ).

Previous Models

- Many distributed verification problems require $\Omega(\mathrm{diam}(G))$ to solve...
- ...but can be solved much faster when vertices are given additional **labels** (or **certificates**, or **proofs**):
  - proof labeling schemes (PLS)
  - nondeterministic local decision (NLD)
  - locally checkable proofs (LCP)
- In all of these models, verification occurs in $O(1)$ time.

Acyclicity PLS Example

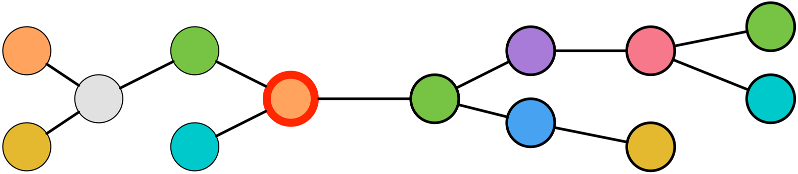
Suppose we want to verify that (G, φ) is cycle-free...



Acyclicity PLS Example

Suppose we want to verify that (G, φ) is cycle-free...

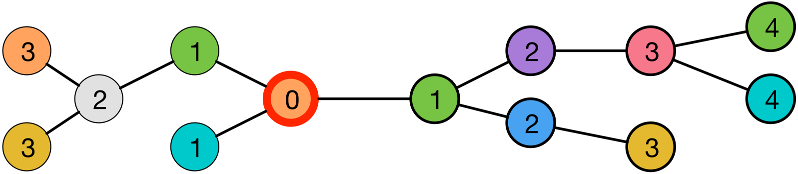
- An oracle picks a vertex from G to be the root, and gives each vertex v a **label** $\ell(v)$ consisting of its distance from the root.



Acyclicity PLS Example

Suppose we want to verify that (G, φ) is cycle-free...

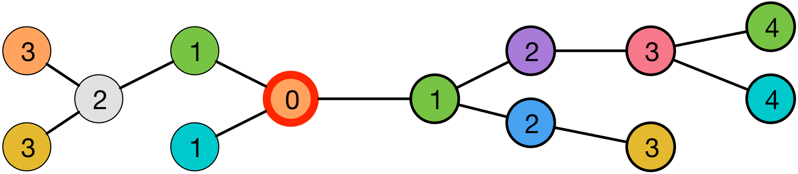
- An oracle picks a vertex from G to be the root, and gives each vertex v a **label** $\ell(v)$ consisting of its distance from the root.
- Each vertex v sends $\ell(v)$ to its neighbors in a single communication round.



Acyclicity PLS Example

Suppose we want to verify that (G, φ) is cycle-free...

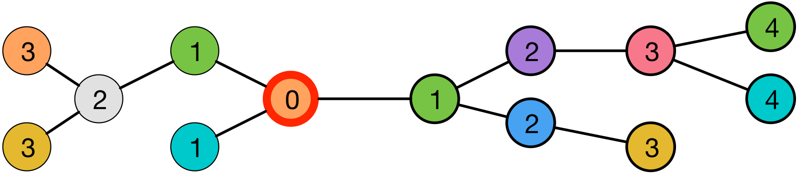
- An oracle picks a vertex from G to be the root, and gives each vertex v a **label** $\ell(v)$ consisting of its distance from the root.
- Each vertex v sends $\ell(v)$ to its neighbors in a single communication round.
- A vertex v **accepts** the labeling if either
  1. $\ell(v) = 0$ and v receives all 1's (i.e., v is the root), or
  2. there is a unique neighbor u with $\ell(u) = \ell(v) - 1$ while all other neighbors w satisfy $\ell(w) = \ell(v) + 1$.



Acyclicity PLS Example

Suppose we want to verify that (G, φ) is cycle-free...

- An oracle picks a vertex from G to be the root, and gives each vertex v a **label** $\ell(v)$ consisting of its distance from the root.
- Each vertex v sends $\ell(v)$ to its neighbors in a single communication round.
- A vertex v **accepts** the labeling if either
  1. $\ell(v) = 0$ and v receives all 1's (i.e., v is the root), or
  2. there is a unique neighbor u with $\ell(u) = \ell(v) - 1$ while all other neighbors w satisfy $\ell(w) = \ell(v) + 1$.
- Otherwise v **rejects** the label.



Acyclicity PLS Example

The PLS for acyclicity satisfies the following two properties:

completeness If G is cycle free, then there exists a labeling ℓ such that all vertices accept.

soundness If G contains a cycle, then for every labeling ℓ, there exists a vertex which rejects the labeling.

The PLS **complexity** of a problem is the minimum **size** of labels needed to solve the problem.

Theorem (Korman, Kutten, Peleg, PODC 2010)

The PLS complexity of verifying acyclicity is $\Theta(\log \operatorname{diam}(G))$.

t-PLS

Question

Can PLS be made more efficient (label size) if we allow longer vertification time? What are the tradeoffs between space and time for PLS verification?

t-PLS

Question

Can PLS be made more efficient (label size) if we allow longer vertification time? What are the tradeoffs between space and time for PLS verification?

Definition

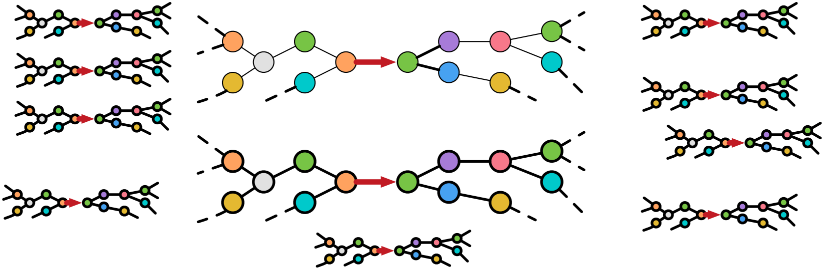
A t-**PLS** for a predicate P consists of a **prover** and a **verifier**.

- The prover is an oracle that assigns labels to the vertices of a network configuration.
- The verifier is a t-round distributed algorithm which verifies the labeling produced by the prover.
- The scheme must be both **complete** and **sound**.

Lower Bounds

Main technique: **edge crossings** (generalizes Baruch, Fraigniaud, Patt-Shamir, PODC '15)

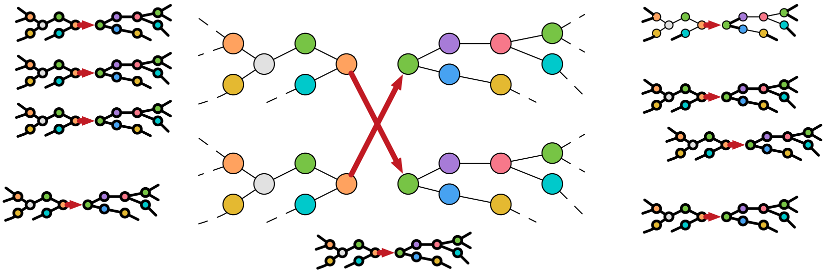
- Suppose
  1. (G, φ) satisfies P,
  2. (G, φ) has many edges $e_1, e_2, \ldots, e_k$ with disjoint, isomorphic, t-neighborhoods, and



Lower Bounds

Main technique: **edge crossings** (generalizes Baruch, Fraigniaud, Patt-Shamir, PODC '15)

- Suppose
  1. (G, φ) satisfies P,
  2. (G, φ) has many edges $e_1, e_2, \ldots, e_k$ with disjoint, isomorphic, t-neighborhoods, and
  3. "crossing" any pair of edges e_i, e_j results in a graph which does not satisfy P.



Lower Bounds

Main technique: **edge crossings** (generalizes Baruch, Fraigniaud, Patt-Shamir, PODC '15)

- Suppose
  1. (G, ϕ) satisfies P,
  2. (G, ϕ) has many edges $e_1, e_2, \ldots, e_k$ with disjoint, isomorphic, t-neighborhoods, and
  3. "crossing" any pair of edges e_i, e_j results in a graph which does not satisfy P.
- Then the labels for any t-PLS for P cannot be too small, or else soundness fails:
  - verifier cannot distinguish original configuration from crossed configuration.

Theorem

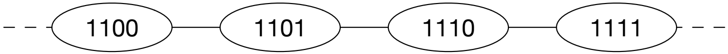
Any t-PLS for acyclicity requires labels of size $\Omega\left(\frac{\log \operatorname{diam}(G)}{t}\right)$.

Upper Bounds

Main technique: **label sharing**

- Start with a 1-PLS.

Example for acyclicity:

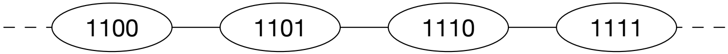


Upper Bounds

Main technique: **label sharing**

- Start with a 1-PLS.
- Observe correlations between nearby labels.

Example for acyclicity:

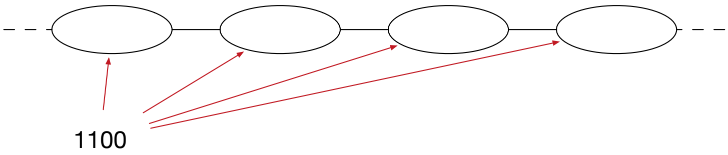


Upper Bounds

Main technique: **label sharing**

- Start with a 1-PLS.
- Observe correlations between nearby labels.
- Break labels into smaller **shares** and distribute to nearby vertices, while eliminating redundant information.

Example for acyclicity:

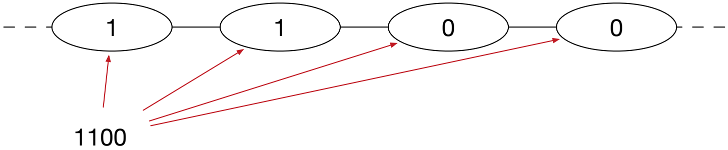


Upper Bounds

Main technique: **label sharing**

- Start with a 1-PLS.
- Observe correlations between nearby labels.
- Break labels into smaller **shares** and distribute to nearby vertices, while eliminating redundant information.

Example for acyclicity:



Theorem

There is a t-PLS for acyclicity which uses labels of size $O\left(\frac{\log \operatorname{diam}(G)}{t}\right)$.

Other results

1. Label sharing improves complexity of **isomorphism** problem by $(1/t)$-factor (**universal scheme**).
2. $\log^* n$-space implementation of acyclicity checker.

Questions

Question

Are there problems for which t verification time...

- improves label size by only $(1/o(t))$-factor?
- does not improve label size at all?

Questions

Question

Are there problems for which t verification time...

- improves label size by only $(1/o(t))$-factor?
- does not improve label size at all?

THANK YOU!!!