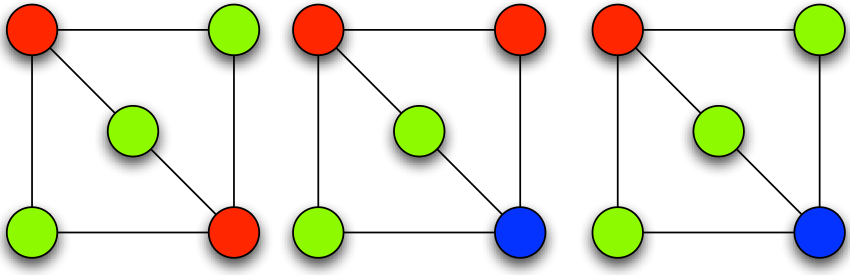
Local Distributed Decision

Pierre Fraigniaud LIAFA Amos Korman LIAFA David Peleg Weizmann Institute of Science

L.E.A. STRUCO Workshop, Pont-à-Mousson, Nov. 12-15, 2013

Outline

Decide coloring



Computational model

$\mathcal{LOCAL}$ model

In each round during the execution of a distributed algorithm, every processor:

1. sends messages to its neighbors,
2. receives messages from its neighbors, and
3. computes, i.e., performs individual computations.

Input

An input configuration is a pair (G, x) where G is a connected graph, and every node $v \in V(G)$ is assigned as its local input a binary string $\mathrm{x}(v) \in \{0, 1\}^*$.

Output

The output of node v performing Algorithm $\mathcal{A}$ running in G with input x and identity assignment Id:

$$\mathrm{out}_{\mathcal{A}}(G, \mathrm{x}, \mathrm{Id}, v)$$

Languages

A distributed language is a decidable collection of configurations.

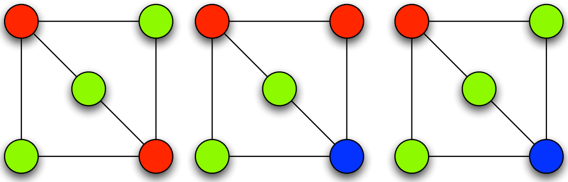
- `Coloring` $= \{(G, \mathrm{x}) \text{ s.t. } \forall v \in V(G), \forall w \in N(v), \mathrm{x}(v) \neq \mathrm{x}(w)\}$.
- `At-Most-One-Selected` $= \{(G, \mathrm{x}) \text{ s.t. } \| \mathrm{x} \|_1 \leq 1\}$.
- `Consensus` $= \{(G, (\mathrm{x}_1, \mathrm{x}_2)) \text{ s.t. } \exists u \in V(G), \forall v \in V(G), \mathrm{x}_2(v) = \mathrm{x}_1(u)\}$.
- `MIS` $= \{(G, \mathrm{x}) \text{ s.t. } S = \{v \in V(G) \mid \mathrm{x}(v) = 1\} \text{ is a MIS}\}$.

Decision

Let $\mathcal{L}$ be a distributed language.

Algorithm $\mathcal{A}$ *decides* $\mathcal{L}$ $\iff$ for every configuration (G, x):

- If $(G, \mathrm{x}) \in \mathcal{L}$, then for every identity assignment Id, $\mathrm{out}_{\mathcal{A}}(G, \mathrm{x}, \mathrm{Id}, v) =$ "yes" for every node $v \in V(G)$;
- If $(G, \mathrm{x}) \notin \mathcal{L}$, then for every identity assignment Id, $\mathrm{out}_{\mathcal{A}}(G, \mathrm{x}, \mathrm{Id}, v) =$"no" for at least one node $v \in V(G)$.



Local decision

Definition
LD(t) is the class of all distributed languages that can be decided by a distributed algorithm that runs in at most t communication rounds.

$$\mathsf{LD} = \cup_{t \geq 0} \mathsf{LD}(t)$$

- `Coloring` $\in$ LD and `MIS` $\in$ LD.
- `AMOS`, `Consensus`, and `SpanningTree` are not in LD.

Outline

Related work

What can be computed locally?

Define LCL as LD($O(1)$) involving

- solely graphs of constant maximum degree
- inputs taken from a set of constant size

Theorem (Naor and Stockmeyer [STOC '93])

If there exists a randomized algorithm that constructs a solution for a problem in LCL in $O(1)$ rounds, then there is also a deterministic algorithm constructing a solution for that problem in $O(1)$ rounds.

Proof uses Ramsey theory.

Not clearly extendable to languages in LD($O(1)$) $\setminus$ LCL.

$(\Delta + 1)$-coloring

Arbitrary graphs

- can be randomly computed in expected #rounds $O(\log n)$
  (Alon, Babai, Itai [J. Alg. 1986]) (Luby [SIAM J. Comput. 1986])
- best known deterministic algorithm performs in $2^{O(\sqrt{\log n})}$ rounds (Panconesi, Srinivasan [J. Algorithms, 1996])

Bounded degree graphs

- Randomization does not help for 3-coloring the ring
  (Naor [SIAM Disc. Maths 1991])
- can be randomly computed in expected #rounds $O(\log \Delta + \sqrt{\log n})$ (Schneider, Wattenhofer [PODC 2010])
- best known deterministic algorithm performs in $O(\Delta + \log^* n)$ rounds
  (Barenboim, Elkin [STOC 2009]) (Kuhn [SPAA 2009])

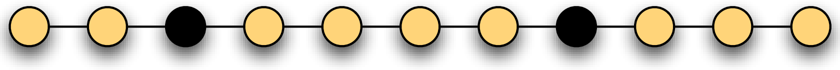
2-sided error Monte Carlo algorithms

Focus on distributed algorithms that use randomization but whose running time are deterministic.

(p, q)-decider

- If $(G, \mathrm{x}) \in \mathcal{L}$ then, for every identity assignment Id, $\mathtt{Pr}[\mathrm{out}_{\mathcal{A}}(G, \mathrm{x}, \mathrm{Id}, v) =$ "yes" for every node $v \in V(G)] \geq p$
- If $(G, \mathrm{x}) \notin \mathcal{L}$ then, for every identity assignment Id, $\mathtt{Pr}[\mathrm{out}_{\mathcal{A}}(G, \mathrm{x}, \mathrm{Id}, v) =$ "no" for at least one node $v \in V(G)] \geq q$

Example: AMOS



Randomized algorithm

- every unmarked node says “yes” with probability 1;
- every marked node says “yes” with probability p.

Remarks:

- Runs in zero time;
- If the configuration has at most one marked node then correct with probability at least p.
- If there are at least $k \geq 2$ marked nodes, correct with probability at least $1 - p^k \geq 1 - p^2$
- Thus there exists a (p, q)-decider for $q + p^2 \leq 1$.

Bounded-probability error local decision

Definition
BPLD(t, p, q) is the class of all distributed languages that have a randomized distributed (p, q)-decider running in time at most t.

Remark
For p and q such that $p^2 + q \leq 1$, there exists a language $\mathcal{L} \in \text{BPLD}(0, p, q)$, such that $\mathcal{L} \notin \text{LD}(t)$, for any $t = o(n)$.

A sharp threshold for hereditary languages

Hereditary languages

A language $\mathcal{L}$ is *hereditary* if it is closed by node deletion.

- `Coloring` and `AMOS` are hereditary languages.
- Every language $\{(G, \epsilon) \mid G \in \mathcal{G}\}$ where $\mathcal{G}$ is hereditary is... hereditary. (Examples of hereditary graph families are planar graphs, interval graphs, forests, chordal graphs, cographs, perfect graphs, etc.)

Theorem (F., Korman, Peleg [FOCS 2011])

Let $\mathcal{L}$ be an hereditary language and let t be a function of triples (G, x, Id). If $\mathcal{L} \in BPLD(t, p, q)$ for constants $p, q \in (0, 1]$ such that $p^2 + q > 1$, then $\mathcal{L} \in LD(O(t))$.

Outline

Distributed certification

One motivation

Settings in which one must perform local verifications repeatedly.

- Self-stabilizing algorithms
- Construction algorithms that may fail
- Property testing

Definition

An algorithm $\mathcal{A}$ verifies $\mathcal{L}$ if and only if for every configuration (G, x), the following hold:

- If $(G, \mathrm{x}) \in \mathcal{L}$, then there exists a certificate y such that, for every id-assignment Id, $\mathrm{out}_{\mathcal{A}}(G, (\mathrm{x}, \mathrm{y}), \mathrm{Id}, v) =$"yes" for all $v \in V(G)$;
- If $(G, \mathrm{x}) \notin \mathcal{L}$, then for every certificate y, and for every id-assignment Id, $\mathrm{out}_{\mathcal{A}}(G, (\mathrm{x}, \mathrm{y}), \mathrm{Id}, v) =$"no" for at least one node $v \in V(G)$.

Non-determinism helps

Definition

$\mathsf{NLD}(t)$ is the class of all distributed languages that can be verified in at most t communication rounds.

$$\mathsf{NLD} = \cup_{t\geq 0}\mathsf{NLD}(t)$$

Example

$\texttt{Tree} = \{(G,\epsilon) \mid G \text{ is a tree}\} \in \mathsf{NLD}(1)$.

Certificate given at node v is $\mathrm{y}(v) = \mathrm{dist}_G(v,\hat{v})$, where $\hat{v} \in V(G)$ is an arbitrary fixed node.

Verification procedure verifies the following:

- $\mathrm{y}(v)$ is a non-negative integer,
- if $\mathrm{y}(v) = 0$, then $\mathrm{y}(w) = 1$ for every neighbor w of v, and
- if $\mathrm{y}(v) > 0$, then there exists a neighbor w of v such that $\mathrm{y}(w) = \mathrm{y}(v) - 1$, and, for all other neighbors w' of v, we have $\mathrm{y}(w') = \mathrm{y}(v) + 1$.

NLD-complete problem

Reduction

$\mathcal{L}_1$ is locally reducible to $\mathcal{L}_2$, denoted by $\mathcal{L}_1 \preceq \mathcal{L}_2$, if there exists a constant time local algorithm $\mathcal{A}$ such that, for every configuration (G, x) and every id-assignment Id, $\mathcal{A}$ produces $\mathrm{out}(v) \in \{0,1\}^*$ as output at every node $v \in V(G)$ so that

$$(G, \mathrm{x}) \in \mathcal{L}_1 \iff (G, \mathrm{out}) \in \mathcal{L}_2 .$$

The language `Containment`

$x(v) = (\mathcal{E}(v), \mathcal{S}(v))$ where:

- $\mathcal{E}(v)$ is an element
- $\mathcal{S}(v)$ is a finite collection of sets

$\{(G, (\mathcal{E}, \mathcal{S})) \mid \exists v \in V,\ \exists S \in \mathcal{S}(v) \text{ s.t. } S \supseteq \{\mathcal{E}(u) \mid u \in V\}\}.$

Theorem

`Containment` *is NLD-complete.*

Proof

Reduction

For every node v, set $\mathcal{E}(v)$ as the ball of radius t around v where t is the "running time" of a non-deterministic algorithm for $\mathcal{L}$.

Let $\mathrm{width}(v) = 2^{|\mathsf{Id}(v)|+|\mathsf{x}(v)|}$. Every node v

- constructs all possible input configurations (G', x') on graphs with at most $\mathrm{width}(v)$ nodes, and,
- for each configuration (G', x'), constructs one set S equal to the collection of every t-ball around every node of G'.

At least one node v gets the actual configuration (G, x).

Hence the equivalence......

NLD membership

Cf. BPNLD

Combining non-determinism with randomization

$$\text{BPNLD}(t) = \cup_{p^2+q\leq 1}\text{BPNLD}(t,p,q)$$

$$\text{BPNLD} = \cup_{t\geq 0}\text{BPNLD}(t,p,q)$$

Theorem
BPNLD contains all languages.

Proof
The certificate is a map of the graph, i.e., an isomorphic copy H of G, with nodes labeled from 1 to n.
Each node v is also given its label $\ell(v)$ in H.
The proof that nodes can probabilistically check $H \sim G$ relies on two facts:

- To be "cheated", a wrong map must be a lift of G.
- One can check whether H is a lift of G by having node(s) labeled 1 acting as in `AMOS`.

□

The “most difficult” decision problem

The problem `Cover`

$\{(G, (\mathcal{E}, \mathcal{S})) \mid \exists v \in V,\ \exists S \in \mathcal{S}(v) \text{ s.t. } S = \{\mathcal{E}(u) \mid u \in V\}\}.$

Theorem

`Cover` *is BPNLD-complete.*

Outline

The oracle GraphSize

Numerous examples in the literature for which the knowledge of the size of the network is required to efficiently compute solutions.

$\texttt{GraphSize} = \{(G, k) \text{ s.t. } |V(G)| = k\}$.

Theorem

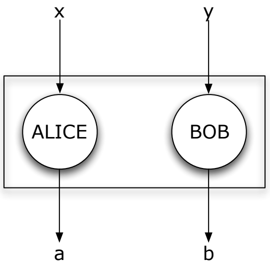
For every language $\mathcal{L}$, we have $\mathcal{L} \in NLD^{\texttt{GraphSize}}$.

Proof

As for BPNLD, the certificate is the map of G.
Nodes cannot be "cheated" whenever they know how many they are.

Outline

Decide whether $x = y$

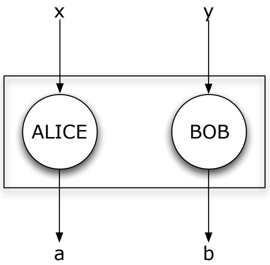


$$a \wedge b = \overline{x \oplus y}$$

Deterministically: impossible !

Randomly (private coin): probability of success $\frac{1}{2}$

CHSH Game (Clauser, Horne, Shimony and Holt [1969])

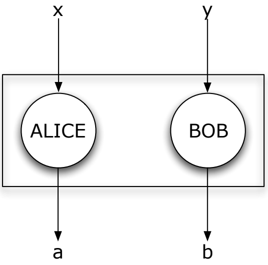


$$a \oplus b = x \wedge y$$

Deterministically: impossible !

Randomly (private coin): probability of success $\frac{1}{2}$

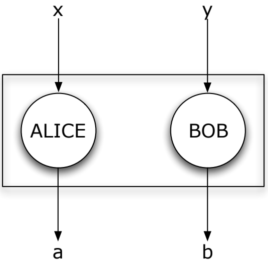
Shared randomness



$$a \oplus b = x \wedge y$$

Deterministically: impossible !
Randomly: probability of success $\frac{1}{2}$
Shared randomness: probability of success $\frac{3}{4}$

Shared randomness



$$a \oplus b = x \wedge y$$

Deterministically: impossible !
Randomly: probability of success $\frac{1}{2}$
Shared randomness: probability of success $\frac{3}{4}$

$$\begin{cases} a(0) \oplus b(0) = 0 \\ a(1) \oplus b(0) = 0 \\ a(0) \oplus b(1) = 0 \\ a(1) \oplus b(1) = 1 \end{cases}$$

What does it mean to be "local"?

Hidden variable $\lambda \in \Lambda$:

$$\Pr(ab \mid xy) = \sum_{\lambda} \Pr(a \mid x\lambda) \cdot \Pr(b \mid y\lambda) \cdot \Pr(\lambda)$$

$\Longrightarrow$ Bell's Inequalities

Physics experiments shows that Bell's inequalities can be violated!

Quantum effect

$$a \oplus b = x \wedge y$$

Deterministically: impossible !

Randomly: probability of success $\frac{1}{2}$

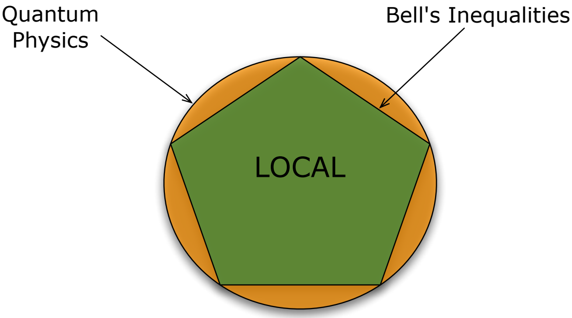
Shared randomness: probability of success $\frac{3}{4}$

Intricated particles (quantum bits):

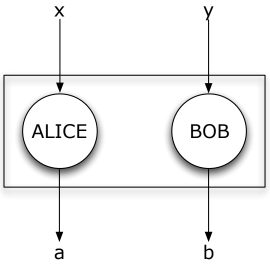
probability of success (Tsilerson [1980]):

$$\cos^2\left(\frac{\pi}{8}\right) \simeq 0.85 > \frac{3}{4}$$

Global picture



PR-box (Popescu, Rohrlic [1994])



$$\Pr(ab \mid xy) = \begin{cases} \frac{1}{2} & \text{if } a \oplus b = x \wedge y \\ 0 & \text{otherwise} \end{cases}$$

Deterministically: impossible !

Randomly: probability of success $\frac{1}{2}$

Shared randomness: probability of success $\frac{3}{4}$

Intricated particles (quantum bits): prob of success $\cos^2(\frac{\pi}{8})$

PR Box: probability of success 1

The PR box respects causality: it is non-signaling

$$\Pr(ab \mid xy) = \begin{cases} \frac{1}{2} & \text{if } a \oplus b = x \wedge y \\ 0 & \text{otherwise} \end{cases}$$

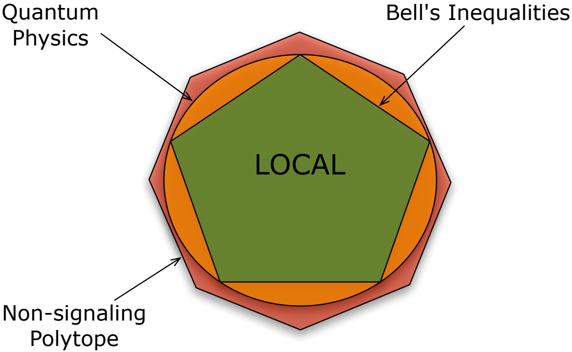
$$\Pr(a \mid xy) = \Pr(a, b = 0 \mid xy) + \Pr(a, b = 1 \mid xy) = \frac{1}{2}$$

and

$$\Pr(a \mid x\bar{y}) = \Pr(a, b = 0 \mid x\bar{y}) + \Pr(a, b = 1 \mid x\bar{y}) = \frac{1}{2}$$

$$\Rightarrow \boxed{\Pr(a \mid xy) = \Pr(a \mid x) \text{ and } \Pr(b \mid xy) = \Pr(b \mid y)}$$

Global picture (enhanced)



Importance of the xor-operator

A game between Alice and Bob is defined by a pair (δ, f) of boolean functions.

The objective of Alice and Bob playing game (δ, f) is, for every inputs x and y, to output values a and b satisfying

$$\delta(a, b) = f(x, y)$$

in *absence of any communication* between the two players.

Theorem (Arfaoui, F. [SIROCCO 2012])
Let (δ, f) be a 2-player game that is not equivalent to any XOR-game. Let p be the largest success probability for (δ, f) over all local boxes. Then every box solving (δ, f) with probabilistic guarantee $> p$ is signaling.

Outline

Further works

- Connection to classical computational complexity theory (time and space).
- Complexity/computability issues: Deciding $\mathcal{L} \in$ LD? $\mathcal{L} \in$ NLD?
- Other interpretation functions (cf. Arfaoui, F., Pelc [SSS 2013])
- Connection with logics (FO, EMSO, ...)

Further works

- Connection to classical computational complexity theory (time and space).
- Complexity/computability issues: Deciding $\mathcal{L} \in$ LD? $\mathcal{L} \in$ NLD?
- Other interpretation functions (cf. Arfaoui, F., Pelc [SSS 2013])
- Connection with logics (FO, EMSO, ...)

Thank You!