Allowing each node to communicate only once in a distributed system

F. Becker[1] A. Kosowski[2] M. Matamala[3] N. Nisse[4]
I. Rapaport[3] K. Suchan[5] I. Todinca[1]

[1] LIFO, Univ. Orléans, France
[2] Inria, LIAFA, Paris, France
[3] DIM, Universidad de Chile, Santiago, Chile
[4] COATI, Inria, I3S, CNRS, UNS, Sophia Antipolis, France
[5] Universidad Adolfo Ibáñez, Santiago, Chile

ANR DISPLEXITY

Bordeaux, Sept. 3rd 2015

Adding a Referee to an Interconnection Network: What Can(not) Be Computed in One Round. IPDPS 2011
Allowing each node to communicate only once in a distributed system: shared whiteboard models SPAA 2012

Goal: Monitoring properties in large-scale distributed networks

Property testing

A *central entity*:

1. queries some nodes (typically $o(n)$)
2. extracts some local information from each query (typically $O(\log n)$ bits)
   ex: "What is your i^{th} neighbor?"
3. decides some (global) property about the graph or about node labeling
   (Is it planar? connected? what is its diameter? Is the coloring proper?...)

this talk is not about Property testing...

Distributed decision

Each node

1. gathers some (local) information by exchanging messages with its neighborhood
2. decides some (global) property about the graph or about node labeling
   Node's Output $\in \{0,1\}$ If YES instance: all nodes must answer 1
   otherwise, at least one node must answer 0

this talk is not about Distributed decision either... but ...

Models of distributed computing

Goal: Monitoring properties in large-scale distributed networks

Property testing

A *central entity*:

1. queries some nodes (typically $o(n)$)
2. extracts some local information from each query (typically $O(\log n)$ bits)
   ex: "What is your i^{th} neighbor?"
3. decides some (global) property about the graph or about node labeling
   (Is it planar? connected? what is its diameter? Is the coloring proper?...)

this talk is not about Property testing...

Distributed decision

Each node

1. gathers some (local) information by exchanging messages with its neighborhood
2. decides some (global) property about the graph or about node labeling
   Node's Output $\in \{0, 1\}$ — If YES instance: all nodes must answer 1, otherwise, at least one node must answer 0

this talk is not about Distributed decision either... but ...

Models of distributed computing

Goal: Monitoring properties in large-scale distributed networks

Property testing

A *central entity*:

1. queries some nodes (typically $o(n)$)
2. extracts some local information from each query (typically $O(\log n)$ bits)
   ex: "What is your i^{th} neighbor?"
3. decides some (global) property about the graph or about node labeling
   (Is it planar? connected? what is its diameter? Is the coloring proper?...)

this talk is not about Property testing...

Distributed decision

Each node

1. gathers some (local) information by exchanging messages with its neighborhood
2. decides some (global) property about the graph or about node labeling
   Node's Output $\in \{0,1\}$
   If YES instance: all nodes must answer 1
   otherwise, at least one node must answer 0

this talk is not about Distributed decision either... but ...

Models of distributed computing

Goal: Monitoring properties in large-scale distributed networks

Property testing

A *central entity*:

1. queries some nodes (typically $o(n)$)
2. extracts some local information from each query (typically $O(\log n)$ bits)
   ex: "What is your i^{th} neighbor?"
3. decides some (global) property about the graph or about node labeling
   (Is it planar? connected? what is its diameter? Is the coloring proper?...)

this talk is not about Property testing...

Distributed decision

Each node

1. gathers some (local) information by exchanging messages with its neighborhood
2. decides some (global) property about the graph or about node labeling
   Node's Output $\in \{0, 1\}$ If YES instance: all nodes must answer 1
   otherwise, at least one node must answer 0

this talk is not about Distributed decision either... but ...

Models of distributed computing

Goal: Monitoring properties in large-scale distributed networks

Property testing

A *central entity*:

1. queries some nodes (typically $o(n)$)
2. extracts some local information from each query (typically $O(\log n)$ bits)
   ex: "What is your i^{th} neighbor?"
3. decides some (global) property about the graph or about node labeling
   (Is it planar? connected? what is its diameter? Is the coloring proper?...)

this talk is not about Property testing...

Distributed decision

Each node

1. gathers some (local) information by exchanging messages with its neighborhood
2. decides some (global) property about the graph or about node labeling
   Node's Output $\in \{0, 1\}$ — If YES instance: all nodes must answer 1
   otherwise, at least one node must answer 0

this talk is not about Distributed decision either... but ...

LOCAL model [Peleg 2000]

$G = (V, E)$ be a n-node graph, with a labeling on nodes
message-passing system

- nodes have distinct IDs
- t rounds (possibly a function of n)
- synchronous
- no faults
- at each round, nodes exchange messages with their neighbors
- messages have arbitrary size
- nodes have arbitrary computational power

If $t \geq diameter(G)$, any (decidable) property can be decided.

LOCAL model [Peleg 2000]

$G = (V, E)$ be a n-node graph, with a labeling on nodes
message-passing system

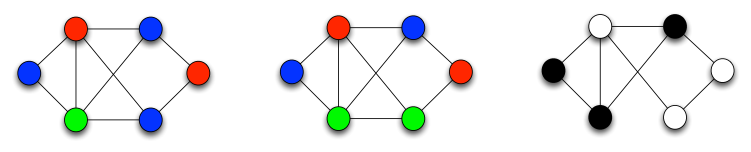
- nodes have distinct IDs
- t rounds (possibly a function of n)
- synchronous
- no faults
- at each round, nodes exchange messages with their neighbors
- messages have arbitrary size
- nodes have arbitrary computational power

If $t \geq diameter(G)$, any (decidable) property can be decided.

LOCAL model in $O(1)$ rounds

Some trivial examples in 1 round

Deciding if a coloring is proper, Deciding a MIS,...



Proper coloring: $c : V \to \mathbb{N}$ s.t. any two adjacent vertices have $\neq$ colors
Maximal Independent Set (MIS): $S \subseteq V$ s.t. no adjacent vertices in S.

Seminal(?) work Cole,Vishkin 86 + Linial 92

Computing a 3-coloring in rings requires $\Theta(\log^* n)$ rounds

What cannot be computed Kuhn,Moscibroda,Wattenhoffer 04

Minimum Vertex Cover (or Max Matching)
in k rounds: $\Omega(n^{c/k^2}/k)$-approx.

LOCAL model in $O(1)$ rounds

Some trivial examples in 1 round

Deciding if a coloring is proper, Deciding a MIS,...

Seminal(?) work — Cole,Vishkin 86 + Linial 92

Computing a 3-coloring in rings requires $\Theta(\log^* n)$ rounds

What cannot be computed — Kuhn,Moscibroda,Wattenhoffer 04

Minimum Vertex Cover (or Max Matching)

in k rounds: $\Omega(n^{c/k^2}/k)$-approx.

$O(1)$-approx requires $\Omega(\sqrt{\log n / \log\log n})$ rounds

Minimum Dominating Set (or Max Indep. Set)

in k rounds: $\Omega(\Delta^{1/k}/k)$-approx.

$O(1)$-approx requires $\Omega(\log \Delta / \log\log \Delta)$ rounds

LOCAL model in $O(1)$ rounds

Some trivial examples in 1 round

Deciding if a coloring is proper, Deciding a MIS,...

Seminal(?) work — Cole,Vishkin 86 + Linial 92

Computing a 3-coloring in rings requires $\Theta(\log^* n)$ rounds

What cannot be computed — Kuhn,Moscibroda,Wattenhoffer 04

Minimum Vertex Cover (or Max Matching)

in k rounds: $\Omega(n^{c/k^2}/k)$-approx.

$O(1)$-approx requires $\Omega(\sqrt{\log n/\log\log n})$ rounds

Minimum Dominating Set (or Max Indep. Set)

in k rounds: $\Omega(\Delta^{1/k}/k)$-approx.

$O(1)$-approx requires $\Omega(\log \Delta/\log\log\Delta)$ rounds

LOCAL model in $O(1)$ rounds

Locally Checkable Labeling (LCL) — Naor, Stockmeyer 95

in graph with bounded degree

- non-decidable in general
- randomization does not help
- does not depend on ID, but on local ordering

Decision Problem — Fraigniaud,Korman,Peleg 2013

- randomization helps
- non-determinism helps
- randomization + non-determinism = *All*

LOCAL model in $O(1)$ rounds

Locally Checkable Labeling (LCL) — Naor, Stockmeyer 95

in graph with bounded degree

- non-decidable in general
- randomization does not help
- does not depend on ID, but on local ordering

Decision Problem — Fraigniaud,Korman,Peleg 2013

- randomization helps
- non-determinism helps
- randomization + non-determinism = *All*

CONGEST model

$G = (V, E)$ be a n-node graph, with a labeling on nodes
message-passing system

- nodes have distinct IDs
- t rounds (possibly a function of n)
- synchronous
- no faults
- at each round, nodes exchange messages with their neighbors
- messages have size $O(\log n)$
- nodes have arbitrary computational power

If $t \geq diameter(G)$, any (decidable) property can be decided.

CONGEST model

$G = (V, E)$ be a n-node graph, with a labeling on nodes
message-passing system

- nodes have distinct IDs
- t rounds (possibly a function of n)
- synchronous
- no faults
- at each round, nodes exchange messages with their neighbors
- messages have size $O(\log n)$
- nodes have arbitrary computational power

~~If $t \geq diameter(G)$, any (decidable) property can be decided.~~

CONGEST model: What cannot be computed...

... in diameter number of rounds

B upper bound on the size of messages
D: diameter of graph

Frischknecht,Holzer,Wattenhofer, 2012

Any distributed randomized ϵ-error algorithm requires:

Diameter
- $\Omega(n/B)$ rounds to decide if $D \leq 4$ (even for $D \leq 5$)
- $\Omega(\sqrt{n}/B + D)$ rounds to c-approx of diameter, $c < 3/2$

Girth $\Omega(\sqrt{n}/B + D)$ rounds to c-approx of girth, $c < 2$

proofs rely on Two-Party Communication Complexity:

Alice: $a \in \{0,1\}^k$ and Bob: $b \in \{0,1\}^k$
need to compute $f : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}$
what number of bits (as a function of k) do they need to exchange?

Disjointness problem: any algorithm using public randomness requires $\Omega(k)$

$f(a,b) = 0$ iff $\exists i \leq k$ such that $a[i] = b[i] = 1$

idea: $V(G) = A \cup B$: A represents a, B represents b, $\frac{Required\ \ info}{|cut(A,B)| * |messages|} \leq \#Rounds$

CONGEST model: What cannot be computed...

... in diameter number of rounds

B upper bound on the size of messages
D: diameter of graph

Frischknecht, Holzer, Wattenhofer, 2012

Any distributed randomized ϵ-error algorithm requires:

Diameter
- $\Omega(n/B)$ rounds to decide if $D \leq 4$ (even for $D \leq 5$)
- $\Omega(\sqrt{n}/B + D)$ rounds to c-approx of diameter, $c < 3/2$

Girth $\Omega(\sqrt{n}/B + D)$ rounds to c-approx of girth, $c < 2$

proofs rely on Two-Party Communication Complexity:

Alice: $a \in \{0,1\}^k$ and Bob: $b \in \{0,1\}^k$
need to compute $f : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}$
what number of bits (as a function of k) do they need to exchange?

Disjointness problem: any algorithm using public randomness requires $\Omega(k)$

$f(a,b) = 0$ iff $\exists i \leq k$ such that $a[i] = b[i] = 1$

idea: $V(G) = A \cup B$: A represents a, B represents b, $\frac{Required\ info}{|cut(A,B)| * |messages|} \leq \#Rounds$

CONGEST model: What cannot be computed...

... in diameter number of rounds

B upper bound on the size of messages
D: diameter of graph

Frischknecht, Holzer, Wattenhofer, 2012

Any distributed randomized ϵ-error algorithm requires:

Diameter
- $\Omega(n/B)$ rounds to decide if $D \leq 4$ (even for $D \leq 5$)
- $\Omega(\sqrt{n}/B + D)$ rounds to c-approx of diameter, $c < 3/2$

Girth $\Omega(\sqrt{n}/B + D)$ rounds to c-approx of girth, $c < 2$

proofs rely on Two-Party Communication Complexity:

Alice: $a \in \{0,1\}^k$ and Bob: $b \in \{0,1\}^k$
need to compute $f : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}$
what number of bits (as a function of k) do they need to exchange?

Disjointness problem: any algorithm using public randomness requires $\Omega(k)$

$f(a,b) = 0$ iff $\exists i \leq k$ such that $a[i] = b[i] = 1$

idea: $V(G) = A \cup B$: A represents a, B represents b, $\frac{Required\ \ info}{|cut(A,B)| * |messages|} \leq \#Rounds$

Distributed testing

So far

- property testing: **central entity** gather structured local info from some nodes
- distributed decision: all **nodes** gather information and answer 0 or 1
  - LOCAL: (unbounded size) message passing
  - CONGEST: $O(\log n)$ bits message passing
  - # of rounds: $O(1)$ or function of n

Distributed testing

So far

- property testing: **central entity** gather structured local info from some nodes
- distributed decision: all **nodes** gather information and answer 0 or 1
  - LOCAL: (unbounded size) message passing
  - CONGEST: $O(\log n)$ bits message passing
  - # of rounds: $O(1)$ or function of n
- distributed testing:
  all **nodes** gather information and send some structured info ("message") to a **central entity**

Distributed testing

So far

- property testing: **central entity** gather structured local info from some nodes
- distributed decision: all **nodes** gather information and answer 0 or 1
  - LOCAL: (unbounded size) message passing
  - $\mathrm{CONGEST}$: $O(\log n)$ bits message passing
  - $\#$ of rounds: $O(1)$ or function of n
- distributed testing:
  all **nodes** gather information and send some structured info ("message") to a **central entity**

in distributed testing setting, LOCAL model and $O(1)$ rounds:
testing Cycle-Freeness requires messages of size $\Omega(\log d)$ per node with degree d
[Arfaoui,Fraigniaud,Ilcinkas,Mathieu, 2014]

Distributed testing

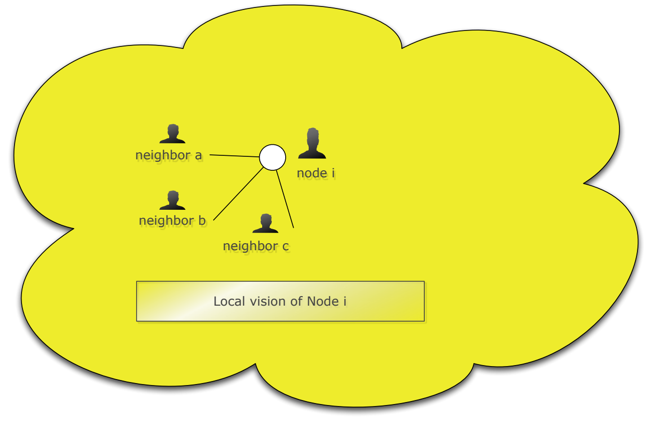
So far

- property testing: **central entity** gather structured local info from some nodes
- distributed decision: all **nodes** gather information and answer 0 or 1
  - LOCAL: (unbounded size) message passing
  - CONGEST: $O(\log n)$ bits message passing
  - # of rounds: $O(1)$ or function of n
- distributed testing:
  all **nodes** gather information and send some structured info ("message") to a **central entity**

in distributed testing setting, LOCAL model and $O(1)$ rounds:
testing Cycle-Freeness requires messages of size $\Omega(\log d)$ per node with degree d
[Arfaoui,Fraigniaud,Ilcinkas,Mathieu, 2014]

- our model: all **nodes** send some "message" to a **central entity**

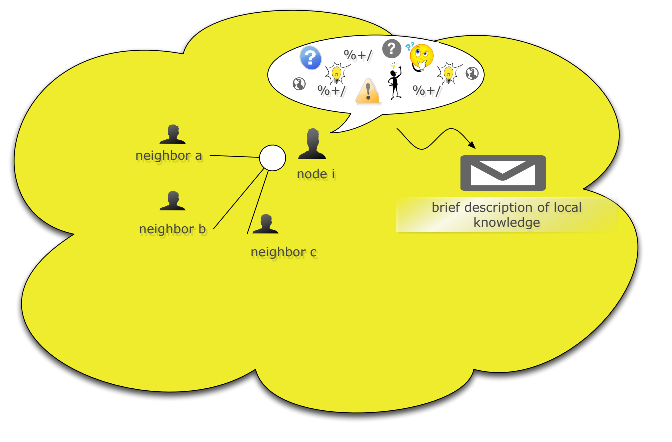
Our model



Graph with n nodes. Nodes have distinct identifiers in $\{1, \cdots, n\}$.

A node knows: its ID and the IDs of its neighbors

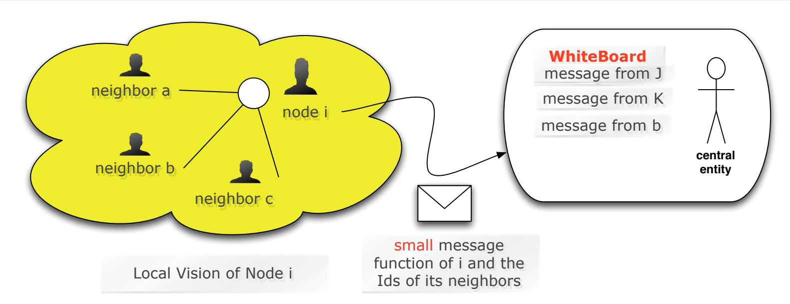
Our model



A node has arbitrary computation power.

Goal: encode its local knowledge in a small message (typically $O(\log n)$ bits).

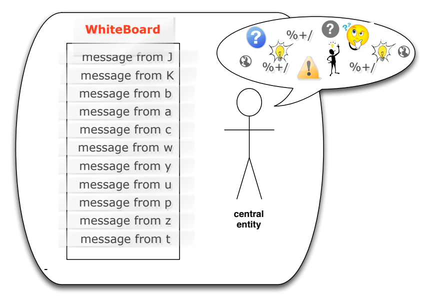
Our model



Each node sends its (unique) message to a central entity

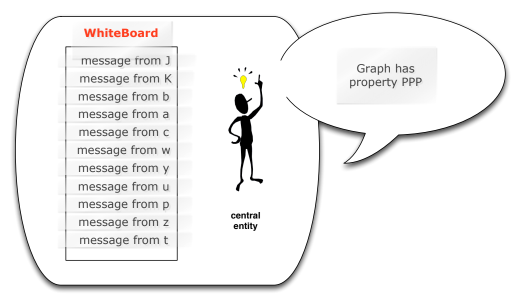
Remark: If $|message| = n$ bits, then node gives its whole neighboorhood

Our model



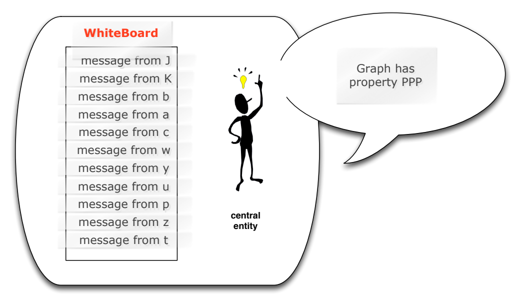
The referee has arbitrary computation power and use the n messages to...

Our model



... answer a question about the graph (typically: "does G has some property?")

Our model



… answer a question about the graph (typically: "does G has some property?")

Related with *Number-in-hand model* with simultaneous messages $f(\mathbf{x_1}, \cdots, \mathbf{x_k})$ [Babai,Kimmel,Lokam 2004]

To sum up: Model of distributed computing

Principle

Does G belongs to $\mathcal{P}$?

- each node encodes its local knowledge

$$message : \text{ID of } v \times \text{IDs of } N(v) \to message(v), \text{ and}$$

$$|message(v)| = O(\log n) \text{ bits}$$

- the referee decodes the n messages to answer

$$answer : (message(v_i))_{i \leq n} \to \{0,1\}^*$$

Hypothesis

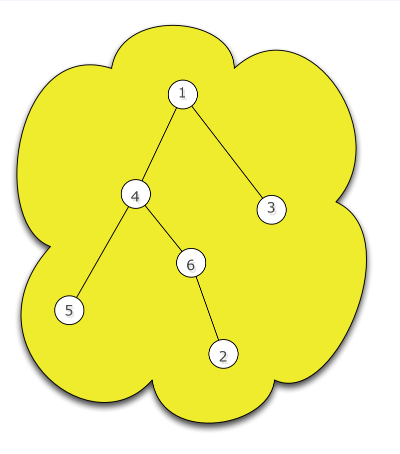
- arbitrary computational power: *message* and *answer* are arbitrary functions
- IDs are distinct in $\{1, \cdots, n\}$

Remark: if bounded maximum degree: each node may send its full adjacency list.

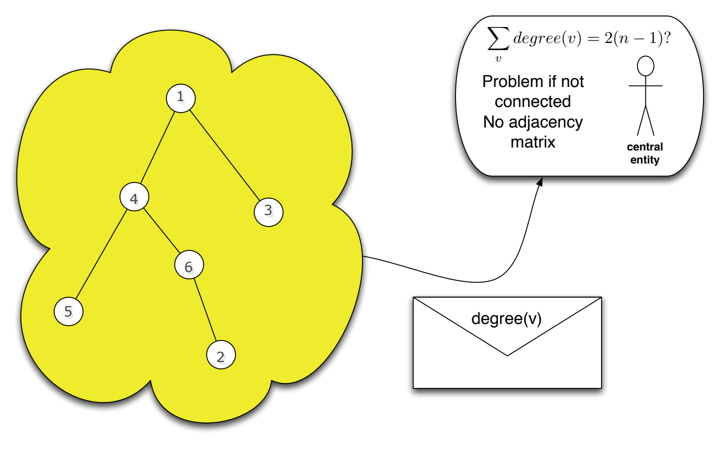
Problem

in total: $O(n \log n)$ bits of local information
What kind of question can be answered?

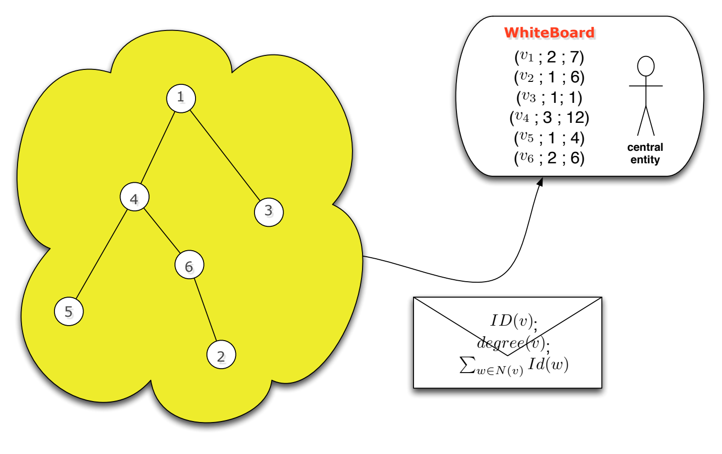
Example: Does G is a tree? If yes, compute its adjacency matrix.



Example: Does G is a tree? If yes, compute its adjacency matrix.

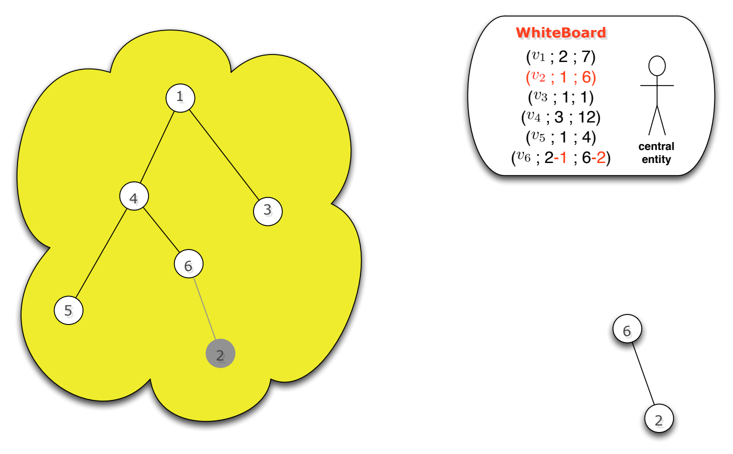


Example: Does G is a tree? If yes, compute its adjacency matrix.



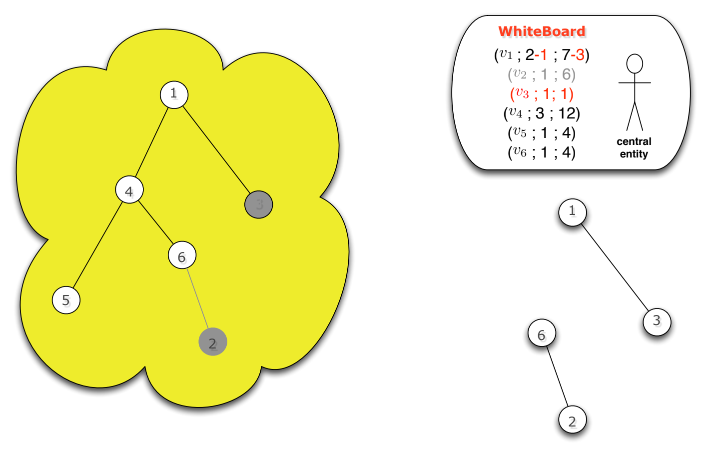
Each node sends its ID, its degree and the sum of the IDs of its neighbors

Example: Does G is a tree? If yes, compute its adjacency matrix.



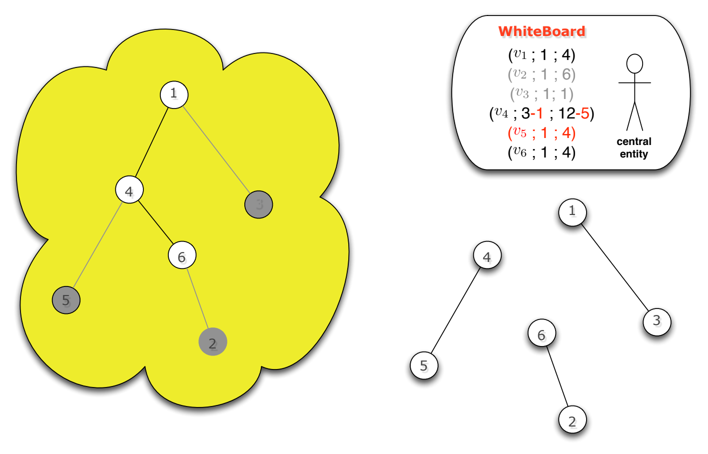
The referee iteratively "prunes" the one-degree nodes in the whiteboard
In parallel, he re-builds the tree.

Example: Does G is a tree? If yes, compute its adjacency matrix.



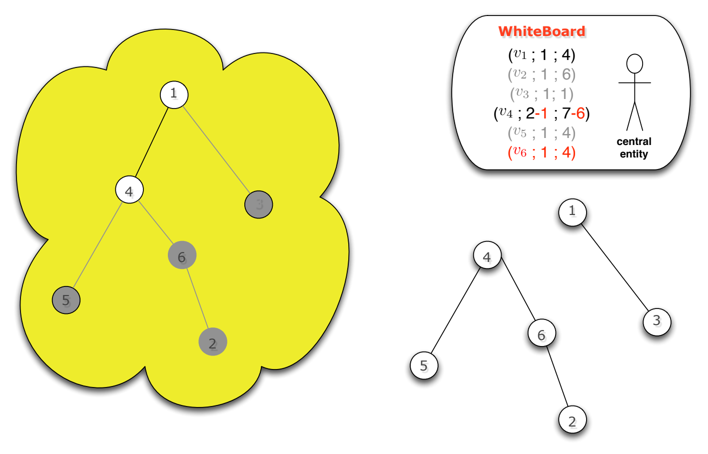
The referee iteratively "prunes" the one-degree nodes in the whiteboard
In parallel, he re-builds the tree.

Example: Does G is a tree? If yes, compute its adjacency matrix.



The referee iteratively "prunes" the one-degree nodes in the whiteboard
In parallel, he re-builds the tree.

Example: Does G is a tree? If yes, compute its adjacency matrix.



The referee iteratively "prunes" the one-degree nodes in the whiteboard
In parallel, he re-builds the tree.

What Can(not) Be Computed in One Round

G degeneracy k: $\exists v \in V(G)$ with degree $\leq k$ and $G \setminus v$ degeneracy $\leq k$.

Possible: BUILD graphs with degeneracy $\leq k$, using messages of size $O(k \cdot \log n)$

Decide if a graph has bounded degeneracy (include planar graphs, bounded genus graphs, bounded treewidth graphs...). If yes, build their adjacency matrix.

proof: generalization of the "pruning process" of trees.

Each node v sends

- its ID
- its degree
- $\sum_{w \in N(v)} ID(w)$
  $\sum_{w \in N(v)} ID^2(w)$
  ...
  $\sum_{w \in N(v)} ID^k(w)$

The referee can compute neighborhood of nodes with degree $\leq k$
(unique integral solution)

What Can(not) Be Computed in One Round

Not possible — using messages of size $O(\log n)$

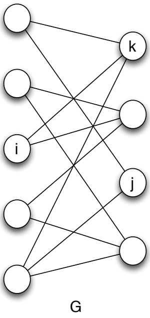
Decide if the graph contains a triangle, a (induced or not) square.
Decide if the graph has diameter at most 3

Recently: characterization of induced subgraphs whose containment can be decided [Kari,Matamala,Rapaport,Salo 2015]

proof: Kind of reduction.
If possible $\Rightarrow$ Possible to build adjacency matrix of bipartite graphs.
$2^{\Omega(n^2)}$ such graphs $\Rightarrow$ impossible to distinguish all of them with $O(n \log n)$ bits.
$\Rightarrow$ contradiction

What Can(not) Be Computed in One Round

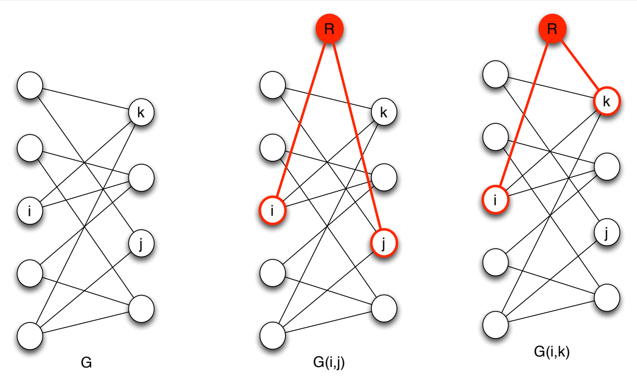
Not possible to decide if G contains a triangle using messages of size $O(\log n)$



Assume $\mathcal{A}$ solves TRIANGLE: use $\mathcal{A}$ to BUILD bipartite graphs

What Can(not) Be Computed in One Round

Not possible to decide if G contains a triangle using messages of size $O(\log n)$

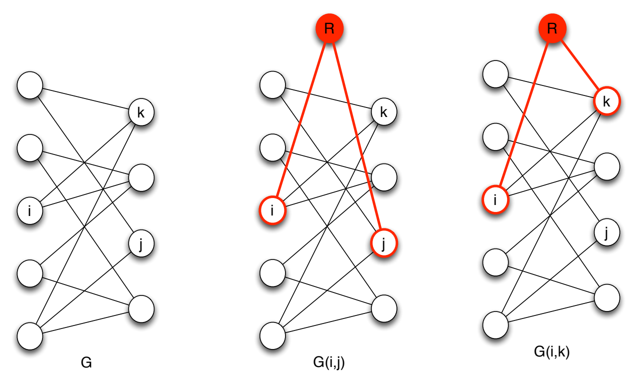


Assume $\mathcal{A}$ solves TRIANGLE: use $\mathcal{A}$ to BUILD bipartite graphs

$$\{i,j\} \in E(G) \Leftrightarrow G(i,j) \text{ contains a triangle}$$

What Can(not) Be Computed in One Round

Not possible to decide if G contains a triangle using messages of size $O(\log n)$

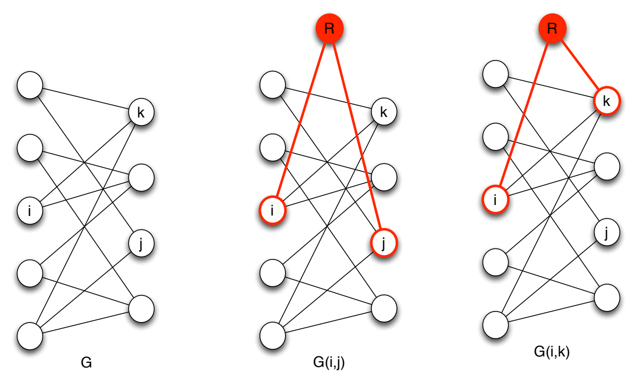


Each node j sends 2 messages:

- $\mathcal{A}_1(j)$ encodes $N_G(j)$ (same as $N_{G(i,k)}(v)$ for any $i, k \in V \setminus \{j\}$)
- $\mathcal{A}_2(j)$ encodes $N_{G(j,i)}(j) = N_G(j) \cup \{R\}$ (the same for any $w \in V \setminus \{j\}$)

What Can(not) Be Computed in One Round

Not possible to decide if G contains a triangle using messages of size $O(\log n)$



Combining the $2n$ messages, the referee simulates $\mathcal{A}$ in each $G(i,j)$, $i,j \in V$
$\Rightarrow$ able to decide if $\{i,j\} \in E(G)$ for any $i,j \in V$

What Can(not) Be Computed in One Round

Randomized version: private/public coins

Not possible [Becker, Montealegre, Rapaport, Todinca, 2014]

For any $\epsilon < 1/2$, any public coins randomized protocol for TRIANGLE (resp., Diameter≤ 3) with ϵ two-sided errors requires messages of size $\Omega(n)$ bits.

Proof: Reduction to INDEX function:

Alice: m-bits vector x, Bob: integer $q \leq m$ INDEX$(x, q) = x_q$, the q^{th} bit of x.

Requires $\Omega(m)$ bits in randomized protocol [Kermer, Nisan, Ron, 1999]

Encoding of x of size $m = n^2$ as a bipartite graph with n vertices.

protocol for TRIANGLE with $g(n)$ bits/message

$\Rightarrow$ protocol for INDEX using $O(g(n) \cdot n)$ bits.

Hierarchy of randomized protocol [Becker, Montealegre, Rapaport, Todinca, 2014]

Public coins $\prec$ Private coins $\prec$ Deterministic

What Can(not) Be Computed in One Round

Randomized version: private/public coins

Not possible [Becker, Montealegre, Rapaport, Todinca, 2014]

For any $\epsilon < 1/2$, any public coins randomized protocol for TRIANGLE (resp., Diameter≤ 3) with ϵ two-sided errors requires messages of size $\Omega(n)$ bits.

Proof: Reduction to INDEX function:
Alice: m-bits vector x, Bob: integer $q \leq m$ INDEX$(x, q) = x_q$, the q^{th} bit of x.
Requires $\Omega(m)$ bits in randomized protocol
[Kermer, Nisan, Ron, 1999]

Encoding of x of size $m = n^2$ as a bipartite graph with n vertices.
protocol for TRIANGLE with $g(n)$ bits/message
$\Rightarrow$ protocol for INDEX using $O(g(n) \cdot n)$ bits.

Hierarchy of randomized protocol [Becker, Montealegre, Rapaport, Todinca, 2014]

Public coins $\prec$ Private coins $\prec$ Deterministic

What Can(not) Be Computed in One Round

Randomized version: private/public coins

Not possible [Becker, Montealegre, Rapaport, Todinca, 2014]

For any $\epsilon < 1/2$, any public coins randomized protocol for TRIANGLE (resp., Diameter≤ 3) with ϵ two-sided errors requires messages of size $\Omega(n)$ bits.

Proof: Reduction to INDEX function:
Alice: m-bits vector x, Bob: integer $q \leq m$ INDEX$(x, q) = x_q$, the q^{th} bit of x.
Requires $\Omega(m)$ bits in randomized protocol
[Kermer, Nisan, Ron, 1999]

Encoding of x of size $m = n^2$ as a bipartite graph with n vertices.
protocol for TRIANGLE with $g(n)$ bits/message
$\Rightarrow$ protocol for INDEX using $O(g(n) \cdot n)$ bits.

Hierarchy of randomized protocol [Becker, Montealegre, Rapaport, Todinca, 2014]

Public coins $\prec$ Private coins $\prec$ Deterministic

Open Problem: Connectivity

Possible — Randomized, using small messages

CONNECTIVITY can be decided by a randomized protocol using public coins and messages of size $O(\log^2 n)$ bits [Ahn,Guha,McGregor, 2012]

What about a deterministic protocol using messages of $o(n)$ bits??

Even in simpler case:

G, $n-1$-regular with $2n$ nodes — G connected or 2 disjoint cliques.

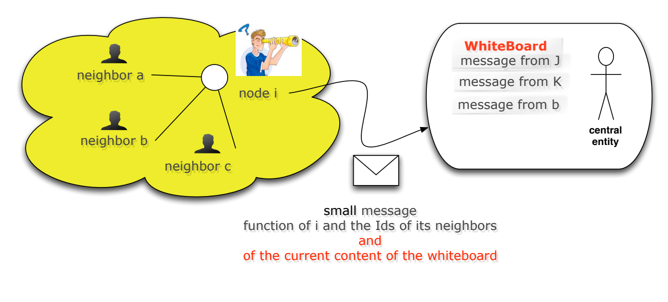
Generalization

Until now:

All nodes write simultaneously on the whiteboard
Don't take advantage of what is written by other nodes.

Now:

Nodes can also read the whiteboard.
Can use previous messages to compute their own message



4 Models

SimAsync model above

All nodes write simultaneously on the whiteboard

SimSync

Nodes write sequentially.
Worst ordering: order chosen by an adversary

Async

Nodes rise hand to speak.
If several nodes rise hand, all write simultaneously.

Sync

Nodes rise hand to speak.
If several nodes rise hand, they write sequentially in worst odering.

Hierarchy of models

$\textsc{SimAsync}(\log n) \prec \textsc{SimSync}(\log n)$

Maximal Independent Set (MIS) containing v_1.

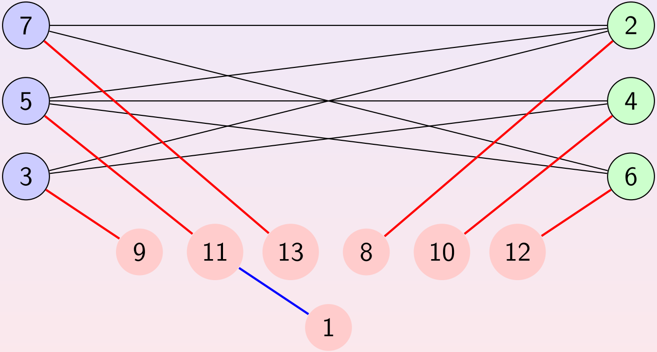
- $MIS \notin \textsc{SimAsync}(\log n)$ (Simultaneous)
- $MIS \in \textsc{SimSync}(\log n)$ (Sequencial, adversarial ordering)

trivial algo.

Hierarchy of models

$\textsc{SimSync}(\log n) \prec \textsc{Async}(\log n)$

BFS-tree in Bipartite graphs $\notin \textsc{SimSync}(\log n)$

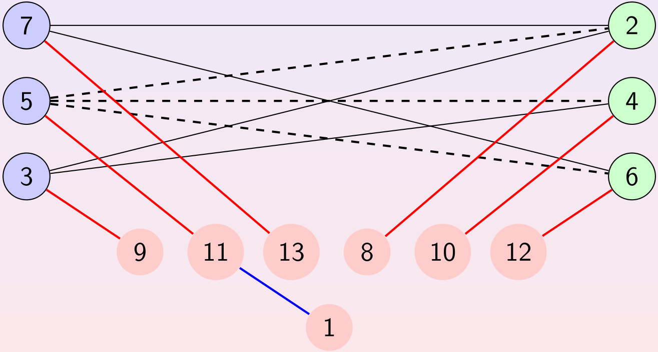


$G = (V, E)$ Bipartite graph $V = \{v_2, \cdots, v_n\}$ (blue and green nodes)
Matching between V and $W = \{v_{n+1}, \cdots, v_{2n-2}\}$ (red nodes)
G_i: Blue edge from 1 to $n + i \in W$ (ex: $n = 6$ and $i = 5$: G_5)

Hierarchy of models

$\textsc{SimSync}(\log n) \prec \textsc{Async}(\log n)$

BFS-tree in Bipartite graphs $\notin \textsc{SimSync}(\log n)$



$N_G(v_i) =$"level 3" of BFS-tree in G_i

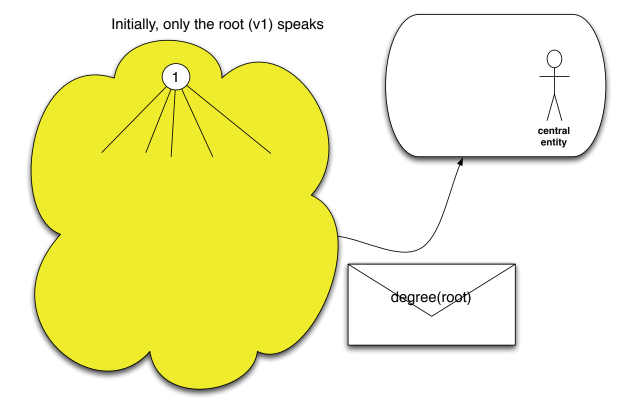
$\textsc{SimSync}$: Adversarial ordering: first nodes of V

Solving BFS $\Rightarrow$ BUILD Bipartite, impossible if $O(\log n)$ bits/message

Hierarchy of models

$\text{SimSync}(\log n) \prec \text{Async}(\log n)$

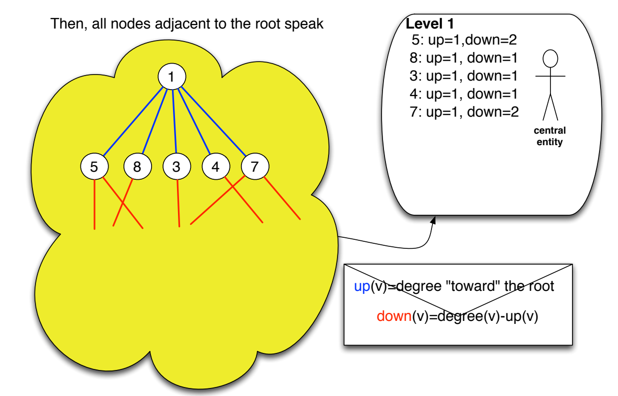
BFS-tree in Bipartite graphs $\in \text{Async}(\log n)$



Hierarchy of models

$\textsc{SimSync}(\log n) \prec \textsc{Async}(\log n)$

BFS-tree in Bipartite graphs $\in \textsc{Async}(\log n)$

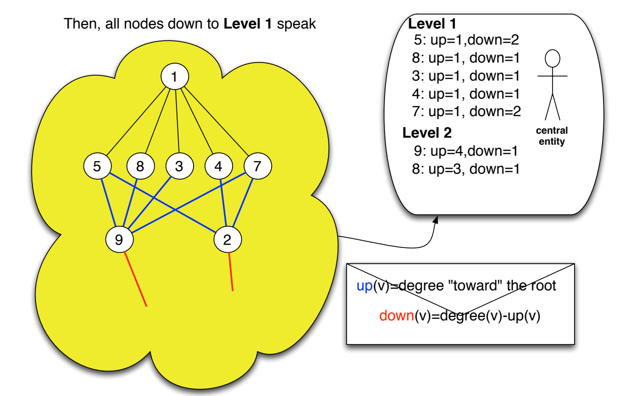


Key point: bipartiteness allows that nodes know when all nodes of previous level have sent their message.

Hierarchy of models

SimSync(log n) $\prec$ Async(log n)

BFS-tree in Bipartite graphs $\in$ Async(log n)

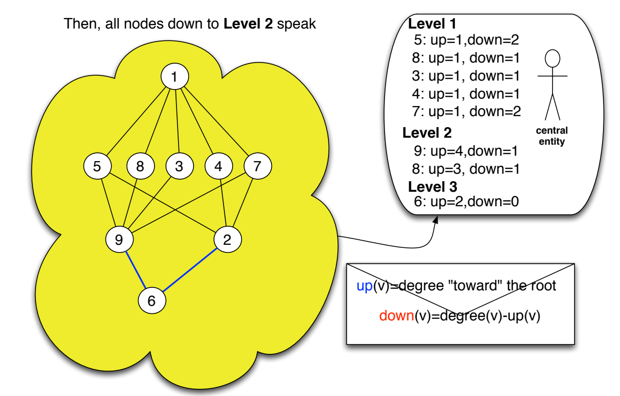


Key point: bipartiteness allows that nodes know when all nodes of previous level have sent their message.

Hierarchy of models

SimSync(log n) $\prec$ Async(log n)

BFS-tree in Bipartite graphs $\in$ Async(log n)

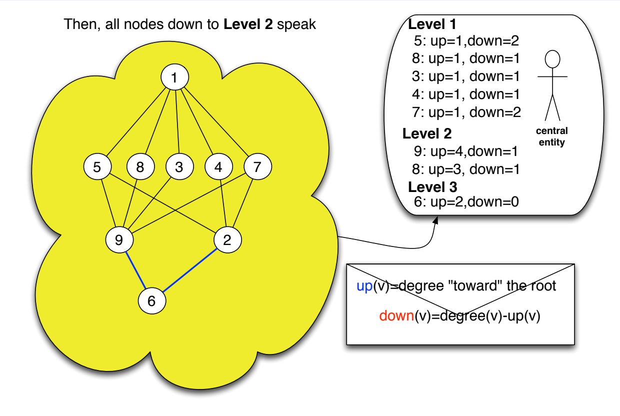


Key point: bipartiteness allows that nodes know when all nodes of previous level have sent their message.

Hierarchy of models

$\textsc{SimSync}(\log n) \prec \textsc{Async}(\log n)$

BFS-tree in Bipartite graphs $\in \textsc{Async}(\log n)$



Similar protocol works in model $\textsc{Sync}$ in general graphs.

Results

$\textsc{SimAsync}(\log n) \prec \textsc{SimSync}(\log n) \prec \textsc{Async}(\log n) \preceq \textsc{Sync}(\log n)$

message: $O(\log n)$ bits	SimAsync	SimSync	Async	Sync
Build k-degenerate	yes	yes	yes	yes
rooted MIS	no	yes	yes	yes
Square	no	no	?	?
Bipartite-BFS	no	no	yes	yes
BFS	?	?	?	yes
Connectivity	?	?	?	yes

Orthogonal criteria

Let $f(n) = o(n)$ and $g(n) = o(f(n))$.

There exist problems solvable in SimAsync(f(n)) and not in Sync(g(n)).

Further works

Probabilistic algorithms?

What if graph partially known? (Distributed testing)

Connectivity?

What is a realistic (useful) model?

...