MPRI Course 2.18.1
Distributed algorithms on networks

Pierre Fraigniaud

https://www.irif.fr/~pierref

MPRI, Université de Paris, 2020-2021
Algorithms vs. programs

Mechanical procedures for solving a given problem

algorithm

program
Distributed Algorithm

A collection of autonomous computing entities collaborating for solving a task in absence of any coordinator
Parallel vs. Distributed

Parallel computing

Performances
> petaFLOPS ($10^{15}$ op./s)

Distributed computing

Coping with uncertainty
temporal and spatial
Sequential vs. Distributed

Typical model for distributed computing

Communication Medium
Communication Medium

Processing elements (PE)

Shared memory

Network

Memory
Limitations Faced by Distributed Computing: Undecidability + Uncertainty

Sources of uncertainties:

- Spatial: communication network
- Temporal: clock drifts (asynchrony, load, etc.)
- Failures (transient, crash, malicious, etc.)
- Selfish behavior (game theory)
- ...

Several Turing machines are weaker than one!
2.18.x Courses

• 2.18.1 Distributed algorithms on networks
  ✓ Spatial issues: locality

• 2.18.2 Distributed algorithms on shared memory
  ✓ Temporal issues: asynchrony and failures
Symmetry Breaking

- Leader election
- Consensus
- Coloring
- Graph problems
- Etc.

Applications:
- Frequency assignments
- Distributed data-bases consistency
2.18.1 Course

1. Fundaments of computing in networks
   - Pierre Fraigniaud (Université de Paris and CNRS)

2. Weak models for computing
   - Mikaël Rabie (Université de Paris)

3. Information spreading in networks
   - Benjamin Doerr (Polytechnic)
Fundaments of computing in networks

Notes de cours

Algorithmique parallèle et distribuée

Ecole Centrale

Algorithmique distribuée pour les réseaux

Masters Programme de Recherche en Informatique
(Univ. Paris-Diderot, ENS, ENS Cachan, Ecole Polytechnique, Univ. Pierre et Marie Curie)

Pierre Fraigniaud
CNRS et Université Paris Diderot
pierre.fraigniaud@irif.fr

Version du March 26, 2018

Abstract

Le contenu de ces notes de cours inclut le matériel présenté dans des cours d’algorithmique distribuée de l’Ecole Centrale de Paris, et du MPRI. Ces notes présentent une introduction à différents aspects du calcul distribué et du calcul parallèle. Sont en particulier traités les problèmes algorithmiques posés par la distance entre les processeurs d’un réseau, et ceux posés par l’asynchronisme entre ces processeurs. L’algorithmique distribuée a cependant pour objectif de résoudre ces problèmes liés à l’espace et au temps. L’algorithmique parallèle se focalise plutôt sur des problèmes de performances, dont un particulier le facteur d’accélération que l’on peut espérer tirer de l’exécution d’un calcul sur plusieurs processeurs, comparé à l’exécution de ce même calcul sur un unique processeur.

Ce document est encore en brouillon. Il est à diffusion restreinte, et n’est mis en ligne que pour fournir une aide aux étudiants des cours concernés. Il n’a pas pour objet remplacer les explications du cours, et certains arguments développés en cours peuvent ne pas apparaître dans ce document.
Local Computing
LOCAL Model

• Each process is located at a node of a network modeled as an $n$-node graph ($n =$ #processes)

• Each process has a unique ID in $\{1, \ldots, n\}$

• Computation proceeds in synchronous rounds during which every process:

  1. Sends a message to each neighbor

  2. Receives a message from each neighbor

  3. Performs individual computation (same algorithm for all nodes)
Lemma If a problem $P$ can be solved in $t$ rounds in the LOCAL model by an algorithm $A$, then there is a $t$-round algorithm $B$ solving $P$ in which every node proceeds in two phases:

Phase 1. Gather all data in the $t$-ball around it
Phase 2. Compute the solution
Graph problems

- Vertex coloring
- Independent set
(Δ+1)-coloring

Δ = maximum node degree of the graph

(Δ+1)-coloring = assign colors to nodes such that every pair of adjacent nodes are assigned different colors.

**Lemma** Every graph is (Δ+1)-colorable

**Theorem** (Brooks, 1941)
Every graph $G$ is $\Delta$-colorable, unless $G$ is a complete graph, or an odd cycle.
3-coloring the n-node cycle $C_n$

Is there an algorithm performing in $o(n)$ rounds?
Deterministic Algorithms
Round complexity of 3-coloring $C_n$

**Theorem** (Cole and Vishkin, 1986) There exists an algorithm for 3-coloring $C_n$ performing in $O(\log^* n)$ rounds.

Iterated logarithms:

- $\log^{(0)} x = \log x$  \quad $\log^{(k+1)} x = \log \log^{(k)} x$
- $\log^* x = \text{smallest } k \text{ such that } \log^{(k)} x < 1$
- $\log^* 10^{100} = 5$

**Theorem** (Linial, 1992) Any 3-coloring algorithm for $C_n$ performs in $\Omega(\log^* n)$ rounds.

Dijkstra Prize 2013
Cole-Vishkin Algorithm

Initial color = ID
Express colors in binary

Assume: $n$ is known, and consistent sense of direction

$p \neq p'$ $\Rightarrow$ proper coloring
$p = p' \Rightarrow b \neq b' \Rightarrow$ proper coloring

$p = p' = 1011$
Number of iterations

- $k$-bit colors $\Rightarrow$ new colors on $\lceil \log_2 k \rceil + 1$ bits
- $\log^* n + O(1)$ rounds to reach colors on 3 bits
- 8 colors down to 3 colors in 5 rounds
- Total number of rounds $= \log^* n + O(1)$
Speeding up the Algorithm

• Every node can simulate 2 rounds in just 1 round
• left round + right round ➞ implemented in 1 round
• Total number of rounds = $\frac{1}{2} \log^* n + O(1)$
Linial’s Lower Bound

every node $x$ decides as a function $\mathcal{A}$ applied to $B_t(x)$ where $B_t(x) = (g_t, g_{t-1}, \ldots, g_1, x, d_1, \ldots, d_{t-1}, d_t)$
Configuration Graph $G_{t,n}$

vertices = \{ (g_t, \ldots, g_1, x, d_1, \ldots, d_t) \in \{1, \ldots, n\}^{2t+1} \}

edges = \{ (g_t, \ldots, g_1, x, d_1, \ldots, d_t) \to (g_{t-1}, \ldots, g_1, x, d_1, \ldots, d_t, d_{t+1}) \}

1. t-round 3-coloring algorithm for $C_n \Rightarrow \chi(G_{t,n}) \leq 3$

2. $t < \frac{1}{2} \log^* n - O(1) \Rightarrow \chi(G_{t,n}) > 3$
Step 1

Lemma \( t \)-round \( c \)-coloring algo for \( C_n \Rightarrow \chi(G_{t,n}) \leq c \)

Proof  \( \mathcal{A} \Rightarrow \) vertex \((g_t, \ldots, g_1, x, d_1, \ldots, d_t)\) colored

\[ \mathcal{A}(g_t, \ldots, g_1, x, d_1, \ldots, d_t) \]

Coloring is proper as

\((g_t, \ldots, g_1, x, d_1, \ldots, d_t)\) and \((g_{t-1}, \ldots, g_1, x, d_1, \ldots, d_{t+1})\)

can appear as view of \( x \) and \( d_1 \) in some instances of ID assignment to the nodes of the ring.
Corollary (Linial, 1992) For $n$ even, 2-coloring $C_n$ requires $\Omega(n)$ rounds.

**Proof** Assume $t$ rounds, with $t \leq n/2 - 2 \Rightarrow 2t + 1 \leq n - 3$.

1. $(x_1, x_2, \ldots, x_{2t+1})$
2. $(x_2, \ldots, x_{2t+1}, y)$
3. $(x_3, \ldots, x_{2t+1}, y, z)$
4. $(x_4, \ldots, x_{2t+1}, y, z, x_1)$
5. $(x_5, \ldots, x_{2t+1}, y, z, x_1, x_2)$
   
   \vdots

2t+1. $(x_{2t+1}, y, z, x_1, \ldots, x_{2t-2})$
2t+2. $(y, z, x_1, \ldots, x_{2t-2}, x_{2t})$
2t+3. $(z, x_1, \ldots, x_{2t-1}, x_{2t})$

\[ \chi(G_{t,n}) > 2 \]
Step 2

**Lemma** \( t < \frac{1}{2} \log^* n - O(1) \implies \chi(G_{t,n}) > 3 \)

Proof is technical (uses line graphs)\(^1\)

But worth reading!

\(^1\)Other proofs use Ramsey theory.
A simpler proof of Linial’s lower bound

Proof (Laurinharju & Suomela, 2014)

\[ \mathcal{A} \text{ is a } k\text{-ary } c\text{-coloring function if} \]

1. \( \mathcal{A}(x_1,x_2,...,x_k) \in \{1,2,...,c\} \) for all \( 1 \leq x_1 < x_2 < ... < x_k \leq n \)
2. \( \mathcal{A}(x_1,x_2,...,x_k) \neq \mathcal{A}(x_2,x_3,...,x_{k+1}) \) for all \( x_k < x_{k+1} \leq n \)

Claim 0. \( t \)-tound algorithm \( \mathcal{A} \) for 3-coloring \( C_n \)

\[ \Rightarrow \mathcal{A} \text{ is } (2t+1)\text{-ary } 3\text{-coloring function} \]

Claim 1. If \( \mathcal{A} \) is a 1-ary \( c \)-coloring function then \( c \geq n \).
Claim 2. If $\mathcal{A}$ is a $k$-ary $c$-coloring function, then there is a $(k-1)$-ary $2^c$-colouring function $\mathcal{B}$.

$$\mathcal{B}(x_1,x_2,\ldots,x_{k-1}) = \{ \mathcal{A}(x_1,x_2,\ldots,x_{k-1},x_k) : x_k > x_{k-1} \}$$

For contradiction, let $1 \leq x_1 < x_2 < \ldots < x_k \leq n$ with

$$\mathcal{B}(x_1,x_2,\ldots,x_{k-1}) = \mathcal{B}(x_2,\ldots,x_{k-1},x_k)$$

Let $d = \mathcal{A}(x_1,x_2,\ldots,x_{k-1},x_k)$.

$$\iff d \in \mathcal{B}(x_1,x_2,\ldots,x_{k-1}) \iff d \in \mathcal{B}(x_2,\ldots,x_{k-1},x_k)$$

$$\iff \exists x_{k+1} > x_k : d = \mathcal{A}(x_2,\ldots,x_k,x_{k+1}) \iff \mathcal{A} \text{ is not proper.} \quad \blacksquare$$
**Theorem** Any 3-coloring algorithm for $C_n$ performs in $\Omega(\log^* n)$ rounds.

**Proof** Let $A$ be a $t$-tound algorithm for 3-coloring $C_n$

$\Rightarrow A$ is $(2t+1)$-ary 3-coloring function (by Claim 0)

$\Rightarrow A$ is $(2t)$-ary $2^3$-coloring function (by Claim 2)

$\Rightarrow A$ is $(2t-1)$-ary $2^{(2^3)}$-coloring function

$\Rightarrow A$ is $(2t-2)$-ary $2^{(3^3)}$-coloring function

$\vdots$

$\Rightarrow A$ is $(1)$-ary $2^{(2t^3)}$-coloring function

$\Rightarrow 2^{(2t^3)} \geq n$ (by Claim 1)

$\Rightarrow t \geq \frac{1}{2} \log^* n - 1.$
(Δ+1)-coloring arbitrary graphs

• Best lower bound (Linial, 1992)
  \[ \Omega(\log^* n) \] rounds

• Best upper bound (Rozon & Ghaffari, 2019)
  \[ O(\text{polylog } n) \] rounds
Complexity as $f(n) + g(\Delta)$

**Theorem** (Linial, 1992)
There is a $(\Delta+1)$-coloring algorithm performing in $O(\log^* n) + \tilde{O}(\Delta^2)$ rounds.

**Theorem** (F., Heinrich, Kosowski, 2016)
There is a $(\Delta+1)$-coloring algorithm performing in $O(\log^* n) + \tilde{O}(\sqrt{\Delta})$ rounds.
(Δ+1)-coloring in O(log* n) + Œ(Δ²) rounds.

**Theorem** (Linial, 1992) Œ(Δ²)-coloring in O(log* n) rounds

**Lemma** For all k > Δ ≥ 2, there exists J = {S₁,...,Sₖ} where

\[ S_i ⊆ \{1,…, 5 \lceil Δ² \log k \rceil \} \text{ for } i=1,…,k \]

such that, for every Δ+1 sets Sᵢ₀, Sᵢ₁,..., Sᵢ₆ in J, we have

\[ Sᵢ₀ ⊈ \bigcup_{j=1,…,Δ} Sᵢⱼ. \]

**Algorithm:**

Init: k = n and color(u) = ID(u)

Each round: color range [1,k] reduced to [1,5 \lceil Δ² \log k \rceil]

color(u) = c ⇒ u has set S_c

New color: smallest \( x ∈ S_c \setminus \bigcup_{i=1,…,Δ} S_{\text{color}(v_i)}. \)
Locally Checkable Labeling

Let $\mathcal{G}_\Delta$ be the set of all (connected) graphs with maximum degree $\Delta$.

**Definition** (Naor and Stockmeyer, 1995) An LCL in $\mathcal{G}_\Delta$ is specified by a finite set of labels, and a finite set of labeled balls with maximum degree $\Delta$, called **good balls**.

**Examples:**

- $k$-coloring, $k$-edge-coloring
- maximal independent set (MIS)
- maximal matching
- Etc.

Focus is on LCL tasks solvable sequentially by a greedy algorithm selecting nodes in arbitrary order, like, e.g., $k$-coloring for $k \geq \Delta+1$. 
Maximal Independent Set

• \((\Delta+1)\)-coloring \(\Rightarrow\) MIS in \(\Delta\) rounds by maximizing \(\{1\}\)

• MIS \(\Rightarrow\) \((\Delta+1)\)-coloring by simulation
Claim 1. At most one node of each clique in the MIS

Claim 2. At least one node of each clique in the MIS

Color = index of node in the MIS
Line Graphs

**Definition** The line graph of a graph $G$ is the graph $L(G)$ such that

- $V(L(G)) = E(G)$
- $\{e, e'\} \in E(L(G)) \iff e$ and $e'$ are incident in $G$
Four classical problems

MIS

MIS on line graph

Maximal Matching

-Vertex Coloring

-coloring on line graph

-Edge Coloring
# Round Complexity

<table>
<thead>
<tr>
<th></th>
<th>MIS</th>
<th>(Δ+1)-coloring</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Deterministic</strong></td>
<td>$2^{\sqrt{\log(n)}}$</td>
<td>$2^{\sqrt{\log(n)}}$</td>
</tr>
<tr>
<td><strong>Randomized</strong></td>
<td>$2^{\sqrt{\log\log(n)}}+O(\log \Delta)$</td>
<td>$2^{\sqrt{\log\log(n)}}$</td>
</tr>
<tr>
<td></td>
<td>[Ghaffari (2016)]</td>
<td>[Chang, Li, Pettie (2018)]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Maximal Matching</th>
<th>(2Δ-1)-edge-coloring</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Deterministic</strong></td>
<td>$O(\log^3 n)$</td>
<td>$O(\log^6 n)$</td>
</tr>
<tr>
<td></td>
<td>[Fischer (2017)]</td>
<td>[Ghaffari, Fisher, Kuhn (2017)]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[Ghaffari, Harris, Kuhn (2018)]</td>
</tr>
<tr>
<td><strong>Randomized</strong></td>
<td>$O(\log^3 \log n)+O(\log \Delta)$</td>
<td>$O(\log^6 \log n)$</td>
</tr>
<tr>
<td></td>
<td>[Barenboim, Elkin, Pettie, Schneider (2012)]</td>
<td>[Elkin, Pettie, Su (2015)]</td>
</tr>
</tbody>
</table>
### Lower Bounds

<table>
<thead>
<tr>
<th>Deterministic</th>
<th>MIS and Maximal Matching</th>
<th>$(\Delta+1)$-coloring and $(2\Delta-1)$-edge-coloring</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Omega\left(\min\left{ \log \Delta / \log \log \Delta, \frac{\sqrt{\log n}}{\log \log n} \right}\right)$</td>
<td>$\Omega(\log^*n)$</td>
<td></td>
</tr>
</tbody>
</table>

Kuhn, Moscibroda, Wattenhofer (2004)  
Linial (1987)  
Naor (1990)
Randomized Algorithms
Randomized algorithm for \((\Delta+1)\)-coloring

**Algorithm** (Barenboim and Elkin, 2013) for node \(u\)

\[
\text{while uncolored do}
\]

\[
\mathcal{C} = \{\text{colors previously adopted by neighbors}\}
\]

pick \(\ell(u)\) at random in \(\{0, 1, \ldots, \Delta+1\} - \mathcal{C}\)

- 0 is picked w/ probability \(\frac{1}{2}\)
- \(\ell(u) \in \{1, \ldots, \Delta+1\} - \mathcal{C}\) is picked w/ proba \(1/(2(\Delta+1-|\mathcal{C}|))\)

\[
\text{if } \ell(u) \neq 0 \text{ and } \ell(u) \not\in \{\text{colors picked by neighbors}\}
\]

then adopt \(\ell(u)\) as my color

else remain uncolored

inform neighbors of status

1 round

1 round
**Definition** A sequence \((\mathcal{E}_n)_{n \geq 1}\) of events holds with high probability (whp) whenever \(\Pr[\mathcal{E}_n] = 1 - O(1/n^c)\) for some constant \(c > 0\).

**Theorem** (Barenboim and Elkin, 2013) The \((\Delta+1)\)-coloring algorithm takes, w.h.p., \(O(\log n)\) rounds.

Recall:
- \(\Pr[A\mid B] = \Pr[A \land B] / \Pr[B]\) or \(\Pr[A \land B] = \Pr[A\mid B] \cdot \Pr[B]\)
- \(\Pr[A] = \Pr[A\mid B] \cdot \Pr[B] + \Pr[A\mid \neg B] \cdot \Pr[\neg B]\)
- Union bound: \(\Pr[A \lor B] \leq \Pr[A] + \Pr[B]\)
- \(\Pr[\exists s \in S : s \models P] = \Pr[(s_1 \models P) \lor (s_2 \models P) \lor \ldots \lor (s_m \models P)]\)
Claim  For every node $u$, at any round, $\Pr[u\ \text{terminates}] \geq \frac{1}{4}$

\[
\Pr[u\ \text{termine}] = \Pr[\ell(u) \neq 0 \text{ et aucun } v \in N(u) \text{ satisfait } \ell(v) = \ell(u)] \\
= \Pr[\forall v \in N(u), \ell(v) \neq \ell(u) \mid \ell(u) \neq 0] \cdot \Pr[\ell(u) \neq 0] \\
= \frac{1}{2} \cdot \Pr[\forall v \in N(u), \ell(v) \neq \ell(u) \mid \ell(u) \neq 0]
\]

\[
\Pr[\ell(v) = \ell(u) \mid \ell(u) \neq 0] = \Pr[\ell(v) = \ell(u) \mid \ell(u) \neq 0 \land \ell(v) = 0] \cdot \Pr[\ell(v) = 0] \\
+ \Pr[\ell(v) = \ell(u) \mid \ell(u) \neq 0 \land \ell(v) \neq 0] \cdot \Pr[\ell(v) \neq 0] \\
= \Pr[\ell(v) = \ell(u) \mid \ell(u) \neq 0 \land \ell(v) \neq 0] \cdot \Pr[\ell(v) \neq 0] \\
\leq \frac{1}{2} \cdot \Pr[\ell(v) = \ell(u) \mid \ell(u) \neq 0 \land \ell(v) \neq 0] \\
= \frac{1}{2} \cdot \frac{1}{\Delta + 1 - |C(u)|}.
\]

\[
\Pr[\exists v \in N(u) : \ell(v) = \ell(u) \mid \ell(u) \neq 0] \leq (\Delta - |C(u)|) \frac{1}{2(\Delta + 1 - |C(u)|)} < \frac{1}{2}
\]
O(log n) rounds w.h.p.

Pr[u does not terminate in k ln(n) rounds]
\[ \leq \left(\frac{3}{4}\right)^k \ln(n) = n^{-k \ln\left(\frac{4}{3}\right)} \]

Pr[\exists u that does not terminate in k ln(n) rounds] \leq n^{1-k \ln\left(\frac{4}{3}\right)}

Let c > 1, by choosing k = (1+c)/ln(4/3), we get:

Pr[all nodes terminates after (1+c)/ln(4/3) ln(n) rounds]
\[ \geq 1 - 1/n^c. \]
Randomized algorithm for MIS

Algorithm (Luby, 1986)

\[
\text{mis}(u) \in \{-1, 0, 1\} = \{\text{undecided, not in MIS, in MIS}\}
\]

At any given round:

\[
H = G\left[\{u : \text{mis}(u) = -1\}\right]
\]

Trick: enforcing an order between nodes:

\[
v \succ u \iff \deg_H(v) > \deg_H(u) \quad \text{or} \quad (\deg_H(v) = \deg_H(u) \text{ and } \text{ID}(v) > \text{ID}(u))
\]
Luby’s algorithm

One phase of the algorithm for node $u$ with $\text{mis}(u) = -1$

\begin{align*}
\text{if } \deg_{H}(u) = 0 \text{ then } \text{mis}(u) &\leftarrow 1 \\
\text{else } \text{join}(u) &\leftarrow \text{true with proba } 1/(2 \deg_{H}(u)), \text{ false otherwise} \\
\text{exchange } \text{join} \text{ with every } v \in \mathcal{N}(u) \\
\text{free}(u) &\leftarrow \nexists \ v \in \mathcal{N}(u) \text{ such that } v \succ u \text{ and } \text{join}(v) = \text{true} \\
\text{if } (\text{join}(u) = \text{true and } \text{free}(u) = \text{true}) \text{ then } \text{mis}(u) &\leftarrow 1 \\
\text{exchange } \text{mis} \text{ with every } v \in \mathcal{N}(u) \\
\text{if } (\text{mis}(u) = -1 \text{ and } \exists v \in \mathcal{N}(u) \text{ mis}(v) = 1) \text{ then } \text{mis}(u) &\leftarrow 0 \\
\text{exchange } \text{mis} \text{ with every } v \in \mathcal{N}(u)
\end{align*}
Luby’s algorithm terminates in $O(\log n)$ rounds, w.h.p.

Structure of the proof:
1. $\Pr[\text{mis}(u) = 1] \geq 1/(4 \deg_H(u))$
2. For a set $\mathcal{N}$ of nodes,
   $$u \in \mathcal{N} \Rightarrow \Pr[u \text{ terminates}] \geq 1/36$$
3. For a large set $\mathcal{E}$ of edges,
   $$e \in \mathcal{E} \Rightarrow \Pr[e \text{ removed from } H] \geq 1/36$$
4. Use concentration result (Chernoff bound) to get w.h.p.
Step 1

\[
\Pr[\text{mis}(u) \neq 1 \mid \text{join}(u)] = \Pr[\exists v \in N(u) : v \succ u \land \text{join}(v) \mid \text{join}(u)]
\]
\[
= \Pr[\exists v \in N(u) : v \succ u \land \text{join}(v)]
\]
\[
\leq \sum_{v \in N(u) : v \succ u} \Pr[\text{join}(v)]
\]
\[
= \sum_{v \in N(u) : v \succ u} \frac{1}{2 \deg(v)}
\]
\[
\leq \sum_{v \in N(u) : v \succ u} \frac{1}{2 \deg(u)}
\]
\[
\leq \frac{\deg(u)}{2 \deg(u)}
\]
\[
\leq \frac{1}{2}
\]

\[
\Pr[\text{mis}(u) = 1] = \Pr[\text{mis}(u) = 1 \mid \text{join}(u)] \cdot \Pr[\text{join}(u)]
\]

\[
\Pr[\text{mis}(u) = 1] \geq \frac{1}{2} \cdot \frac{1}{2 \deg(u)} = \frac{1}{4 \deg(u)}.
\]
Step 2

A node $u$ is large if $\sum_{v \in N(u)} \frac{1}{2 \deg(v)} \geq \frac{1}{6}$

Claim: $u$ large $\Rightarrow$ $\Pr[u \text{ terminates}] \geq 1/36$

- True if $\exists v \in N(u) : \deg_H(v) \leq 2$
- $\forall v \in N(u)$, if $\deg_H(v) \geq 3$ then $\frac{1}{2 \deg(v)} \leq \frac{1}{6}$

$$\implies \exists S \subseteq N(u) : \frac{1}{6} \leq \sum_{v \in S} \frac{1}{2 \deg(v)} \leq \frac{1}{3}$$

$$\Pr[E_1 \lor E_2 \lor \cdots \lor E_r] = \sum_i \Pr[E_i] - \sum_{i \neq j} \Pr[E_i \land E_j] + \sum_{i \neq j \neq k} \Pr[E_i \land E_j \land E_k] - \cdots + (-1)^{r+1} \Pr[E_1 \land \cdots \land E_r].$$
\[
\Pr[\text{mis}(u) \neq -1] \geq \Pr[\exists v \in S : \text{mis}(v) = 1] \\
\geq \sum_{v \in S} \Pr[\text{mis}(v) = 1] - \sum_{v,w \in S, v \neq w} \Pr[\text{mis}(v) = 1 \land \text{mis}(w) = 1].
\]

\[
\implies \Pr[\text{mis}(u) \neq -1] \geq \sum_{v \in S} \Pr[\text{mis}(v) = 1] - \sum_{v,w \in S, v \neq w} \Pr[\text{join}(v) \land \text{join}(w)] \\
\geq \sum_{v \in S} \Pr[\text{mis}(v) = 1] - \sum_{v \in S, w \in S} \sum_{v \in S} \frac{1}{2 \deg(v)} \cdot \frac{1}{2 \deg(w)} \\
\geq \left( \sum_{v \in S} \frac{1}{2 \deg(v)} \right) \left( \frac{1}{2} - \sum_{w \in S} \frac{1}{2 \deg(w)} \right) \\
\geq \frac{1}{6} \left( \frac{1}{2} - \frac{1}{3} \right) = \frac{1}{36}.
\]
Step 3

An edge \( e = \{u, v\} \) is large if \( u \) or \( v \) is large.

For \( e = \{u, v\} \) with \( u < v \), orient the edge \( u \to v \).

**Claim** For every small node \( u \), \( \text{deg}^+(u) \geq 2 \text{deg}^-(u) \)

\[
\begin{array}{c}
\text{Out-degree} \\
\text{In-degree}
\end{array}
\]

Indeed: \( \text{deg}^+(u) < 2 \text{deg}^-(u) \implies \text{deg}(u) < 3 \text{deg}^-(u) \)

\( S = \{v \in N(u) : \text{deg}(v) \leq \text{deg}(u)\} \)

\( |S| \geq \text{deg}^-(u) \implies |S| \geq |N(u)|/3 \)

\[
\sum_{v \in N(u)} \frac{1}{2 \text{deg}(v)} \geq \sum_{v \in S} \frac{1}{2 \text{deg}(v)} \geq \sum_{v \in S} \frac{1}{2 \text{deg}(u)} \geq \frac{\text{deg}(u)}{3} \cdot 2 \frac{1}{\text{deg}(u)} = \frac{1}{6}
\]

\[\blacksquare\]
Let \( m = |E(H)| \) We have:
\[
\sum_{u \text{ petit}} \deg^-(u) \leq \frac{1}{2} \sum_{u \text{ petit}} \deg^+(u) \leq \frac{m}{2}
\]
\[
\Rightarrow \sum_{u \text{ grand}} \deg^-(u) \geq \frac{m}{2} \Rightarrow \text{at least } m/2 \text{ large edges}
\]

\( X_e \) = Bernouilli variable equal to 1 if \( e \) is removed from \( H \)

For \( e \) large, \( \Pr[X_e=1] \geq 1/36 \Rightarrow \mathbb{E}X_e \geq 1/36 \\
X = \sum_{e \text{ large}} X_e \Rightarrow \mathbb{E}X = \sum_{e \text{ large}} \mathbb{E}X_e \geq m/72 \\

Let \( p = \Pr[X \leq \frac{1}{2} \mathbb{E}X] \)
\[
\mathbb{E}X = \sum_{x=0}^{m} x \Pr[X = x] = \sum_{x=0}^{\frac{1}{2} \mathbb{E}X} x \Pr[X = x] + \sum_{x=\frac{1}{2} \mathbb{E}X+1}^{m} x \Pr[X = x] \leq \frac{1}{2} p \mathbb{E}X + (1 - p)m
\]
\[
\Rightarrow p \leq \frac{m - \mathbb{E}X}{m - \frac{1}{2} \mathbb{E}X} \leq \frac{m - \frac{1}{2} \mathbb{E}X}{m} \leq 1 - \frac{1}{144}.
\]

Let \( \mathcal{C} = \text{ « at least } m/144 \text{ edges are removed from } H \) » 

\( \Pr[\mathcal{C}] \geq 1/144 \)
Step 4

Let $Y_1, Y_2, \ldots, Y_k$ be Bernoulli variables w/ parameter $q = 1/144$
Let $Y = Y_1 + Y_2 + \ldots + Y_k$

Remark: Let $\alpha = 144/143$. If $Y \geq \log_\alpha |E(G)|$ then termination.

Chernoff Inequality: $\forall \delta \in ]0, 1[, \Pr[Y \leq (1 - \delta)\mathbb{E}Y] \leq e^{-\frac{1}{2} \delta^2 \mathbb{E}Y}$.

We have $\mathbb{E}Y = kq$, so, with $\delta = \frac{1}{2}$, we get $\Pr[Y \leq \frac{kq}{2}] \leq e^{-\frac{kq}{8}}$

For $k = c \log_\alpha n$, we get $\Pr[Y \leq \frac{c q \log_\alpha n}{2}] \leq e^{-\frac{c q \log_\alpha n}{8}}$

Let $c = 4/q \implies \frac{1}{2} c q \log_\alpha n \geq \log_\alpha |E(G)|$ and $c q \geq 8 \ln(\alpha)$.

$\implies e^{-\frac{c q \log_\alpha n}{8}} = \frac{1}{n^{\frac{c q}{8 \ln \alpha}}} \leq \frac{1}{n}$.

$\implies \Pr[Y \leq \log_\alpha m] \leq \frac{1}{n}$.

Thus Luby’s algorithm terminates in $O(\log n)$ rounds w.h.p.
Deterministic ↔ Randomized
Network Decomposition

**Definition** A \((d, c)\)-decomposition of an \(n\)-node graph \(G = (V, E)\) is a partition of \(V\) into clusters such that each cluster has diameter at most \(d\) and the cluster graph is properly colored with colors \(1, \ldots, c\).

**Theorem** [Linial and Saks (1993)]
Every graph has a \((O(\log n), O(\log n))\)-decomposition, and such a decomposition can be computed by a randomized algorithm in \(O(\log^2 n)\) rounds in the LOCAL model.

**Theorem** [Panconesi and Srinivasan (1992)]
A \((2^{O(\sqrt{\log n})}, 2^{O(\sqrt{\log n})})\)-decomposition can be computed \textit{deterministically} in \(2^{O(\sqrt{\log n})}\) rounds in the LOCAL model.
Lemma Given a \((d,c)\)-decomposition, \((\Delta+1)\)-coloring and MIS can be solved in \(O(cd)\) rounds in the LOCAL model.

Proof

Proceed in \(c\) phases, each of \(O(d)\) rounds.
**Theorem** [V. Rozhon and M. Ghaffari (2019)]
A $(O(\log n), O(\log n))$-decomposition can be computed deterministically in $O(\log^{O(1)} n)$ rounds in the LOCAL model.

**Corollary** $(\Delta + 1)$-coloring and MIS can be deterministically solved in $O(\log^{O(1)} n)$ rounds in the LOCAL model.
SLOCAL Model
M. Ghaffari, F Kuhn, Y. Maus (2017)

- Sequential variant of the LOCAL model:
  - nodes are considered sequentially, one by one
  - the current node computes its output based solely on the states of the nodes in the ball of some fixed radius around it

- LOCAL\( (t) \) = \{problems solvable in \( t \) rounds\}

- SLOCAL\( (t) \) = \{problems solvable with balls of radius \( t \)\}

- P-LOCAL = LOCAL\( (\log^{O(1)}n) \)

- P-SLOCAL = SLOCAL\( (\log^{O(1)}n) \)
Completeness Results

In the LOCAL model, a problem $Q$ is $t$-reducible to another problem $P$ if

- $t$-round algorithm for $P$ \Rightarrow $t$-round algorithm for $Q$.

$P$ is P-SLOCAL-complete if $P \in P$-SLOCAL, and any $Q \in P$-SLOCAL is $O(\log^{O(1)}n)$-reducible to $P$.

**Theorem** [M. Ghaffari, F Kuhn, Y. Maus (2017)]
Computing a $(O(\log^{O(1)}n), O(\log^{O(1)}n))$-decomposition is P-SLOCAL-complete.

**Corollary** P-LOCAL $=$ P-SLOCAL.
Derandomization

For Locally Checkable Labeling (LCL) problems:

**Theorem** [M. Naor and L. Stockmeyer (1992)]
\[
\text{LOCAL}(O(1)) = \text{RLOCAL}(O(1))
\]

**Theorem** [L. Feuilloley and P. F. (2015)]
\[
\text{LOCAL}(O(1)) = \text{RLOCAL}(O(1)) \text{ also for } \text{randomly}
\]
locally checkable problems.

**Theorem** [V. Rozhon and M. Ghaffari (2019)]
\[
\text{P-LOCAL} = \text{P-RLOCAL}.
\]
Randomized Algorithms using Shattering

Pick $\bullet$ or $\bigcirc$ u.a.r.

W.h.p., max length monochromatic interval $\leq O(\log n)$

3-coloring or MIS: $\text{Rand}(n) = O(\log^*(\log n))$
Graph Shattering

1. Shatter the graph using randomization
2. Complete each piece deterministically

parts that are fixed after 1.

parts that remain to be fixed by 2.

\[ \text{Rand}(n) \approx \text{Det}(O(\log^{O(1)} n)) \]
Deterministic lower bounds

For any LCL problem in the LOCAL model, its randomized complexity on instances of size $n$ is at least its deterministic complexity on instances of size $\sqrt{\log n}$.

Conclusion: one needs to design better deterministic algorithms for improving the performances of randomized algorithms!
Concluding remarks
## Round Complexity

<table>
<thead>
<tr>
<th></th>
<th>MIS</th>
<th>(Δ+1)-coloring</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Deterministic</strong></td>
<td>$O(\log^{O(1)} n)$</td>
<td>$O(\log^{O(1)} n)$</td>
</tr>
<tr>
<td><strong>Randomized</strong></td>
<td>$O(\log^{O(1)} \log n) + O(\log \Delta)$</td>
<td>$O(\log^{O(1)} \log n)$</td>
</tr>
<tr>
<td><strong>Maximal Matching</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Deterministic</strong></td>
<td>$O(\log^3 n)$</td>
<td>$O(\log^6 n)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ghaffari, Harris, Kuhn (2018)</td>
</tr>
<tr>
<td><strong>Randomized</strong></td>
<td>$O(\log^3 \log n) + O(\log \Delta)$</td>
<td>$O(\log^6 \log n)$</td>
</tr>
</tbody>
</table>
## Lower Bounds

<table>
<thead>
<tr>
<th>Deterministic</th>
<th>MIS and Maximal Matching</th>
<th>$(\Delta+1)$-coloring and $(2\Delta-1)$-edge-coloring</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\Omega(\log n + \Delta)$</td>
<td>$\Omega(\log^* n)$</td>
</tr>
</tbody>
</table>

Balliu et al. (2019)

Linial (1987)

Naor (1990)
Open problems

- Improve the degrees $c$ of the polylog (i.e. $O(\log^c n)$)
- Close the gaps between lower and upper bounds
- Is $(\Delta+1)$-coloring solvable in $O(\log^* n)$ rounds?
- Closing the gap between asynchronous fault-tolerant computing on shared memory and network computing:
  - Existing work on fault-tolerance in synchronous systems
  - Existing work on synchronizers in fault-free systems
  - What about both?
Impact of congestion
CONGEST Model

- Each process is located at a node of a network modeled as an \( n \)-node graph (\( n = \# \text{processes} \))
- Each process has a unique ID in \( \{1, \ldots, n\} \)
- Computation proceeds in synchronous rounds during which every process:
  1. **Sends** a message to each neighbor
  2. **Receives** a message from each neighbor
  3. **Performs** individual computation (same algorithm for all nodes)

Typically, \( B = O(\log n) \)
Non-local problems
Minimum Spanning Tree (MST)

Input of node $u$: ID($u$), $w(e)$ for every $e \in E(u)$

Output of node $u$: list of edges $e \in E(u)$ belonging to MST
Facts about MST

Let $G = (V,E)$ be a connected weighted graph

- Without loss of generality, all weights can be assumed distinct $\iff$ for every $e = \{u,v\}$ with $\text{ID}(u) > \text{ID}(v)$, replace $w(e)$ by $(w(e), \text{ID}(u), \text{ID}(v))$.

- For every cut $(S,V\setminus S)$ in $G$, the edge of \textit{minimum} weight in the cut belongs to the MST.

- For every cycle $C$ in $G$, the edge of \textit{maximum} weight in $C$ does not belong to the MST.
MST is a non-local problem

\[ I_1 = (1, 3) \quad I_2 = (3, 2) \quad I_3 = (1, 2) \]

**Remark** MST requires at least \(D\) rounds in the cycle.

Algorithms with round-complexity \(O(f(n)+D)\) in \(n\)-node graphs of diameter \(D\).

**Objective:** minimizing \(f(n)\)
Borůvska’s algorithm (1926) distributed version

Collection of subtrees called « fragments »

A phase = fragments are merged
Merges use the edge of minimum weight going out of each fragment
Fragments & Merging

\[ e = (ID(u), ID(v), w(e)) \]

\[ N(t) = \# \text{fragments after } t \text{ phases} \]

\[ N(0) = n \]

\[ N(t+1) \leq N(t)/2 \quad \Rightarrow \text{at most } \lceil \log_2 n \rceil \text{ phases} \]
Round complexity

complexity of a phase $= O(\max_F \text{diam}(F))$

$diam(F) \leq n-1$

**Theorem** The distributed version of Borůvska’s algorithm can be implemented in $O(n \log n)$ rounds in the CONGEST model.

The bound is tight:
Another MST algorithm

Based on a Breadth-First Search (BFS) tree

**Lemma** BFS construction requires $O(D)$ rounds in the CONGEST model

Algorithm of node u

```
$id_{min} \leftarrow ID(u)

repeat
    send $id_{min}$ to neighbors, and receive IDs from neighbors
    if $\exists id \in \{IDs\ sent\ by\ neighbors\} : id < id_{min}$ then
        $id_{min} \leftarrow id$
        parent($u$) $\leftarrow ID(v)$ where $v$ is the neighbor which sent $id$
```
Matroid Algorithm (1)

Algorithm for a node $u$

$K \leftarrow E(u)$  \hspace{1cm} edges incident to node $u$

wait until having received an edge from each child

repeat

Know

$K \leftarrow K \cup \{\text{received edges}\}$

Up

$U \leftarrow \{\text{edges previously sent to parent}(u)\}$

Remove

$R \leftarrow \{e \in K \setminus U : U \cup \{e\} \text{ contains a cycle}\}$

Candidate

$C \leftarrow K \setminus (U \cup R)$

if $C \neq \emptyset$ then

send $e \in C$ with minimum weight to parent

receive edges from children

else terminate
Proof of correctness

**Theorem** The Matroid algorithm performs in $O(n + D)$ rounds in the CONGEST model, and enables the root of the tree to construct a MST.

**Lemma 0** Let $A$ and $B$ be acyclic subsets of edges. If $|A| > |B|$ then there exists $e \in A \setminus B$ such that $B \cup \{e\}$ is acyclic.

**Proof** B is a forest $\{T_1, \ldots, T_k\}$. Let $n_i = |V(T_i)|$. We have $|E(T_i)| = n_i - 1$.

For every $i$, there are at most $n_i - 1$ edges of $A$ connecting nodes in $T_i$.

$\Rightarrow$ There is an edge in $A$ whose extremities do not belong to a same tree $T_i$.\[\square\]
A node $u$ is said **active** at phase $t$ if it has not terminated at phase $t - 1$.

Let $h(u) = \text{height of } u = \text{length of longest path to a leaf of the subtree } T_u \text{ rooted at } u$.

**Lemma 1** For every active child $v$ of a node $u$, the set $C$ of candidates for $u$ at time $t$ contains at least one edge sent by $v$ to $u$ before time $t$. $\implies$ **no premature termination**

**Proof** Induction on $h(u)$. Lemma holds for $h(u)=0$.

Assume lemma hold for all nodes at height $\leq k$.

Let $u$ with $h(u)=k+1$, and $v$ active child of $u$. Note $h(v) \leq k$.

Let $E_u$ and $E_v$ be edges sent by $u$ to $p(u)$, and by $v$ to $u=p(v)$ before phase $t$.

Since $h(v)<h(u)$ we have $|E_v| > |E_u|$.

By Lemma 0, $\exists \ e \in E_v \setminus E_u$ such that $E_u \cup \{e\}$ is acyclic $\implies e \in C$. $\blacksquare$
Lemma 2

(a) If $u$ sends $e$ to $p(u)$ at phase $t$ then
1. all edges received by $u$ at phase $t-1$ from its active children were of weight $\geq w(e)$, and
2. all edges to be received by $u$ at phases $\geq t$ will be of weight $\geq w(e)$.

(b) The weights of the edges sent by $u$ to its parent are

Proof True for height $0$. Assume holds for height $k$.

(a.1) Let $u$ with $h(u) = k+1$.
Let $e'$ be edge sent by child $v$ at phase $t-1$.
Let $e'' \in C$ whose existence follows from Lemma 1.
By induction, property (b) implies $w(e'') \leq w(e')$.
By the choice of the edge in $C$, we have $w(e) \leq w(e'')$.
$\Rightarrow w(e') \geq w(e)$.

(a.2) follows from (a.1) and by induction from (b).

(b) follows from (a.2) by the choice of the edge in $C$.  

$\Rightarrow$ it is legitimate to remove edges creating cycles with previously sent edges.
Complexity

In n-node graphs, any set of n edges includes a cycle

- every node sends \( \leq n-1 \) edges
- #rounds \( \leq D + n - 1 \)
Broadcasting the MST from the root to all nodes

Pipelining the edges of $T = \{e_1, e_2, \ldots, e_{n-1}\}$ down the BFS tree

$\Rightarrow \#\text{rounds} \leq D + n - 1$
Wrap Up

- **Borůvska**: $O(n \log n)$ rounds — this is because fragments can have arbitrarily large diameter

- **Matroid**: $O(D+n)$ rounds — this is because too many edges are gathered at a single node.

- **Combining Borůvska and Matroid**:
  - control the diameter of the fragment, and stops when fragments have too large diameter
  - carry on with matroid for computing the (few) edges connecting the fragments already computed by Borůvska
• $D \subseteq V$ is a dominating set if every $u \not\in D$ has a neighbor in $D$.

Remarks:

• Every maximal independent set is a dominating set.
• Every tree has a dominating set of size $\leq n/2$

Objective: Distributed computing of a dominating set of size $\leq n/2$ in consistently oriented trees.
MIS in Rooted Trees

Every node has pointer to its parent

- Perform Cole and Vishkin algorithm with parent
- When colors are on 3 bits, every node pushes down its color
- Performs 5 rounds to get all colors in \( \{1,2,3\} \).

Complexity: \( O(\log^* n) \) rounds
Computing small dominating sets in rooted trees

- $X_d = \{\text{nodes at distance } d \text{ from a leaf}\}$
- $Y = V(T) \setminus (X_0 \cup X_1 \cup X_2)$
- Let $J$ be MIS in $Y$ (comput. in $O(\log^*n)$ rounds)
- Let $D = J \cup X_1$

- $D$ is a dominating set
  - $|X_1| \leq |X_0| \Rightarrow |X_1| \leq \frac{1}{2} |X_0 \cup X_1|$
  - $|J| \leq |(Y \cup X_2) \setminus J| \Rightarrow |J| \leq \frac{1}{2} |Y \cup X_2|$

$\Rightarrow |D| \leq \frac{n}{2}$
Bounding the diameter of fragments
Fast MST algorithm

Two stages:
1. Few phases of Borůvska
2. Completed by Matroid

\[
N(t) \leq \frac{N(t-1)}{2} \implies N(t) \leq \frac{n}{2^t}
\]
\[
diam(t) \leq 3 \cdot diam(t-1) + 2 \implies diam(t) \leq 3^t - 1
\]

Phase \( t \) costs \( O(diam(t) \log^* n) \) rounds
\( \tau \) phases Borůvska costs \( \tilde{O}(3^\tau) \) rounds
Matroid completes in \( O(D + N(\tau)) \) rounds

\[3^\tau = \frac{n}{2^\tau} \implies \text{#rounds} = \tilde{O}(D + n^{0.6131})\]

**Theorem** MST construction can be achieved in \( \tilde{O}(D + \sqrt{n}) \) rounds in the CONGEST model.
Local problems
**C₄-detection**

H is a subgraph of G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$

G is $H$-free if G does not contain H as a subgraph.
Distributed decision

A distributed algorithm $A$ *decides* $\phi$ if and only if:

- $G \models \phi \implies$ all nodes output *accept*
- $G \not\models \phi \implies$ at least one node output *reject*

**Theorem** Deciding $C_4$-freeness can be done in $O(\sqrt{n})$ rounds.
Algorithm 3 $C_4$-detection executed by node $u$.

1: send ID($u$) to all neighbors, and receive ID($v$) from every neighbor $v$
2: send $\deg(u)$ to all neighbors, and receive $\deg(v)$ from every neighbor $v$
3: $S(u) \leftarrow \{$IDs of the $\min\{\sqrt{2n}, \deg(u)\}$ neighbors with largest degrees$\}$
4: send $S(u)$ to all neighbors, and receive $S(v)$ from every neighbor $v$
5: if $\sum_{v \in N(u)} \deg(v) \geq 2n + 1$ then
6: output reject
7: else
8: if $\exists v_1, v_2 \in N(u), \exists w \in S(v_1) \cap S(v_2): w \neq u$ and $v_1 \neq v_2$ then
9: output reject
10: else
11: output accept
12: end if
13: end if

Case 1: there exists a ‘large’ node $w$ in $C$
Case 2: all nodes of $C$ are ‘small’
Lower bound techniques
Réduction du calcul d'une borne inférieure dans le modèle congest au calcul d'une borne inférieure pour un problème de complexité des communications.

La valeur $D$ dépendait du poids d’une arête $e$ à distance $D = \text{diam}(G)$ de nous permettant de conclure à une borne inférieure $\Omega(D)$, ce qui n’est pas forcément d’une grande pertinence pour des graphes de petit diamètre. En revanche, si nous pouvons établir que les nœuds de la partie $A$ ont besoin de « beaucoup » de données provenant de $B$, alors nous pouvons potentiellement exprimer une borne inférieure dépendante de $n$, même pour de petits diamètres. En effet, si $A$ a besoin de $x$ bits provenant de $B$, et que la coupe minimale séparant $A$ et $B$ est de $k$ arêtes, alors le transfert de ces $x$ bits de $A$ vers $B$ nécessitera au moins $x/k \log n$ étapes, car, à chaque étape, au plus $k \log n$ bits peuvent transiter sur $k$ liens dans le modèle congest.

L’argument ci-dessus n’est toutefois pas forcément trivial à utiliser. En effet, outre qu’il convient d’identifier correctement les ensembles $A$ et $B$, l’« nombre de bits de $B$ nécessaires à $A$ » pour prendre une décision n’est pas formellement définie ici. En fait, ce questionnement fait précisément l’objet d’une théorie appelée la complexité de la communication (« communication complexity » en anglais). Dans sa version classique, la complexité des communications fait intervenir deux joueurs, Alice et Bob, devant calculer une fonction $f$ de leurs entrées. Plus précisément, Alice et Bob disposent chacun d’une donnée respective $a$ et $b$, et la question est : combien de bits doivent échanger Alice et Bob pour calculer $f(a, b)$ ? Par exemple $f$ peut être l’indicatrice de $a = b$. Le point crucial est que, pour certaines fonctions $f$, Alice n’a pas nécessairement besoin de tous les bits nécessaires à encoder $b$ pour calculer $f(a, b)$. Cela est particulièrement vrai dans un contexte probabiliste. Ainsi, la technique générale évoquée ci-dessus et conduisant à la borne inférieure $x/k \log n$ ne doit pas définir $x$ comme le nombre de bits pour encoder les données de $B$, mais $x$ comme la complexité de la communication permettant à $A$ de prendre la décision convenable. Ainsi, la partie $A$ du graphe devra correspondre à un joueur, par exemple Alice, et la partie $B$ devra correspondre à un second joueur, par exemple Bob.

Nous allons appliquer cette technique à la mise en évidence d’une borne inférieure pour la construction d’un MST.

3.3.2 Borne inférieure pour la construction d’un MST

Pour établir une borne inférieure sur le temps minimum nécessaire à la construction d’un MST dans le modèle congest, nous allons considérer une famille d’instances, indexées par un entier.
Communication complexity

\[ f : \{0,1\}^N \times \{0,1\}^N \rightarrow \{0,1\} \]

Alice & Bob must compute \( f(a,b) \)

How many bits need to be exchanged between them?
Equality

• Alice gets $a \in \{0,1\}^N$, and Bob gets $a \in \{0,1\}^N$

$$f(a,b) = 1 \iff a = b$$

**Theorem** $\text{CC}(\text{EQ}) = \Omega(N)$. 
Set-disjointness

- Ground set $S$ of size $N$
- Alice gets $A \subseteq S$, and Bob gets $B \subseteq S$

$$f(A,B) = 1 \iff A \cap B = \emptyset$$

**Theorem** $\text{CC}(\text{DISJ}) = \Omega(N)$, even using randomization (i.e., even if Alice and Bob have access to sources of random bits).
\begin{center}
\textbf{Application 1}
\end{center}

**\(\Omega(\sqrt{n})\) lower bound for MST**

\[ n = \Theta(k^4) \]

\[ \text{weight } w \]

\[ \text{weight } 0 \quad \text{weight } \infty \]

\[ k \]

\[ k^2 \]

\[ \text{weight } w_i \]

\[ w_i = 2 \quad w_i' = 2x_i + 1 \text{ with } x_i \in \{0, 1\} \]

**Lemma** Transmitting \(k^2\) bits from \(c_k\) to \(c_1\) takes \(\Omega(k^2)\) rounds

**Proof** (simplified: no recombination)

- \(\exists i, x_i \text{ uses } \leq k/2 \text{ of highway} \Rightarrow \Omega(k \cdot k/2)\) rounds
- \(\forall i, x_i \text{ uses } > k/2 \text{ of highway} \Rightarrow \Omega((k^2 \cdot k/2)/(k \log n))\) rounds
Application 2

Deciding $C_4$-freeness

**Theorem** (Drucker, Kuhn & Oshman, 2014)
Deciding $C_4$-freeness required sending $\Omega(\sqrt{n}/\log n)$ bits between some neighbors

Reduction from Set-Disjointness.

**Lemma** There are $C_4$-free graphs $G_n$ with $n$ nodes and $m=\Omega(n^{3/2})$ edges.
Reduction

Let $A$ and $B$ as in set-disjointness with $N = m = \Omega(n^{3/2})$

Alice's copy of $G_n$

Bob's copy of $G_n$

- Alice keeps $e \in E(G_n)$ iff $e \in A$
- Bob keeps $e \in E(G_n)$ iff $e \in B$

Algo in $R$ rounds exchanges $R \cdot n \log n$ bits

$\implies R \geq \Omega(n^{3/2})/(n \log n)$

$= \Omega(\sqrt{n}/\log n)$
Open problem

deciding $\bigtriangleup$-freeness

$C_3$-free graph

communication complexity fails
Distributed Property Testing

- **Property testing**: checking correctness of large data structure, by performing small (sub-linear) amount of queries.

- Graph queries (with nodes labeled from 1 to n):
  - what is degree of node $x$?
  - what is the $i^{th}$ neighbor of node $x$?

- Two relaxations (farness and randomized decision):
  - $G$ is $\epsilon$-far from satisfying $\phi$ if removing/adding up to $\epsilon m$ edges to/from $G$ results in a graph which does not satisfy $\phi$.
  - algorithm $A$ tests $\phi$ if and only if:
    - $G \models \phi \Rightarrow \Pr[\text{all nodes output accept}] \geq \frac{2}{3}$
    - $G$ is $\epsilon$-far from $\not\models \phi \Rightarrow \Pr[\text{at least one node outputs reject}] \geq \frac{2}{3}$
Testing T-freeness

**Theorem** For every tree $T$, there exists a 1-sided error randomized algorithm performing in $O(1)$ rounds in the CONGEST model, which correctly detects if the given input network contains $T$ as a subgraph, with probability at least $\frac{2}{3}$.

Color coding:

Let $T$ be a $k$-node tree. Label the node from $k$ down to 1, using BFS from arbitrary root.
Algorithm 1 Randomized tree-detection, for a given tree $T$. Algorithm executed by node $u$.

1: send ID($u$) to all neighbors, and receive ID($v$) from every neighbor $v$
2: let $k = |V(T)|$, and pick color($u$) $\in [k]$ uniformly at random
3: send color($u$) to all neighbors, and receive color($v$) from every neighbor $v$
4: for every $c \in [1,k]$, let $N_c(u) = \{v \in N(u) \mid \text{color}(v) = c\}$
5: active($u$) $\leftarrow$ false
6: for $c = 1$ to $k$ do
7: send active($u$) to all neighbors, and receive active($v$) from every neighbor $v$
8: compute $A(u) = \{v \in N(u) \mid \text{active}(v) = true\}$
9: if color($u$) = $c$ and ($\forall c' \in \text{child}(c)$, $N_{c'}(u) \cap A(u) \neq \emptyset$) then
10: active($u$) $\leftarrow$ true
11: end if
12: end for
13: if color($u$) = $k$ and active($u$) = true then
14: output reject
15: else
16: output accept
17: end if

Remark: does not use $\varepsilon$-farness.

$$\Pr[\text{detecting } T] \geq (1/k)^k$$

Perform $O(k^k)$ repetitions of Algorithm 1 to get
$$\text{prob}[\text{detecting } T] \geq \frac{2}{3}$$
Testing $C_3$-freeness

Algorithm of node $u$

Exchange IDs with neighbors
for every neighbor $v$ do
    pick a received ID u.a.r.
    send that ID to $v$
if $u$ receives ID($w$) from $v \in N(u)$ with $w \in N(u)$ and $v \neq w$ then output reject
else output accept

Lemma 1 For any triangle $\Delta$, $\Pr[\Delta \text{ is detected}] \geq 1/n$
Analysis

**Theorem** Let $\epsilon \in ]0, 1[$. If $G$ is $\epsilon$-far from being $C_3$-free, then the algorithm detects a cycle with prob $\geq 1 - (1/e)^{\epsilon/3}$

**Lemma 2** If $G$ is $\epsilon$-far from being $C_3$-free, then $G$ contains at least $\epsilon m/3$ edge-disjoint triangles.

**Proof** Let $S=\{e_1, e_2, \ldots, e_k\}$ be min #edges to remove for making $G$ triangle-free ($k \geq \epsilon m$).

Repeat removing $e$ from $S$, as well as all edges of a triangle $\Delta e$ containing $e$ at least $k/3$ steps.

All triangles $\Delta e$ are edge-disjoint.
Analysis (continued)

Proof (of theorem)

- \( \Pr[\text{no } \Delta \text{ detected}] \leq (1-1/n)^{\varepsilon m/3} \leq (1-1/n)^{\varepsilon n/3} \)

- \((1-1/n)^n \leq 1/e\)

- \(\Pr[\text{no } \Delta \text{ detected}] \leq (1/e)^{\varepsilon/3}\)

Repeat k times with k such that \((1/e)^{\varepsilon k/3} \leq 1/3\)

That is \(k \geq 3 \ln(3) / \varepsilon \Rightarrow \#\text{rounds} = O(1/\varepsilon)\).
Open problem

Is there a distributed tester for $K_5$-freeness running in $O(1)$ rounds in the CONGEST model?
Distributed Verification
Cycle-Freeness

Non locally decidable!
Certifying Cycle-Freeness

Algorithm of node $u$

exchange counters with neighbors if $\exists! \ v \in N(u) : \text{cpt}(v) = \text{cpt}(u) - 1$ and $\forall w \in N(u) \setminus \{v\}, \text{cpt}(w) = \text{cpt}(u) + 1$
then accept else reject

if $G$ is acyclic, then there is an assignment of the counter resulting in all nodes accept.

if $G$ is has a cycle, then for every assignment of the counters, at least one node rejects.
Proof-Labeling Scheme

A distributed algorithm $A$ verifies $\phi$ if and only if:

- $G \models \phi \Rightarrow \exists c : V(G) \rightarrow \{0,1\}^* :$ all nodes accept $(G,c)$
- $G \not\models \phi \Rightarrow \forall c : V(G) \rightarrow \{0,1\}^*$ at least one node rejects $(G,c)$

The bit-string $c(u)$ is called the certificate for $u$ (cf. class NP)

Objective: Algorithms in $O(1)$ rounds (ideally, just 1 round in LOCAL)

Examples:
- Cycle-freeness: $c(u) = \text{dist}_G(u,r)$
- Spanning tree: $c(u) = (\text{dist}_G(u,r),\text{ID}(r))$

Measure of complexity: $\max_{u \in V(G)} |c(u)|$
Application: Fault-Tolerance

Example: Self-stabilization

- Construction algorithm
- Solution
- Certificate of correctness

System state:
- Initial state
- Legal configurations
- Illegal configurations

Time:
- Current state
- Time
- Fault
Universal PLS

**Theorem** For any (decidable) graph property \( \phi \), there exists a PLS for \( \phi \), with certificates of size \( O(n^2) \) bits in \( n \)-node graphs.

**Proof** \( c(u) = (M, x) \) where
- \( M = \) adjacency matrix of \( G \)
- \( x = \) table[1..n] with \( x(i) = \) ID(node with index \( i \))

Verification algorithm:
1. check local consistency of \( M \) using \( x \)
2. if no inconsistencies, check whether \( M \) satisfies \( \phi \)

exercise

\( G \) satisfies \( \iff \) both tests are passed
Lower bound

**Theorem** There exists a graph property for which any PLS has certificates of size $\Omega(n^2)$ bits.

**Proof** Graph automorphism = bijection $f: V(G) \rightarrow V(G)$ such that $\{u,v\} \in E(G) \iff \{f(u),f(v)\} \in E(G)$

**Fact** For $n$ large enough, there are $\geq 2^{\varepsilon n^2}$ graphs with no non-trivial automorphism.

If certificates on $< \varepsilon n^2/3$ bits, then $\exists i \neq j$ such that the three nodes have same certificates on $G_i - G_i$ and $G_i - G_i$. 

![Diagram of graphs showing nodes and edges with node overlap to illustrate the proof.](image-url)
Local hierarchy

• Equivalent of, e.g., polynomial hierarchy in complexity theory

• \{locally decidable properties\} = \Sigma_0 = \Pi_0

• \{locally verifiable properties (with PLS)\} = \Sigma_1

Deciding graph property \(\phi\) is in \(\Sigma_1\) if and only if:
• \(G \models \phi \Rightarrow \exists c\text{ all nodes accept } (G,c)\)
• \(G \not\models \phi \Rightarrow \forall c\text{ at least one node rejects } (G,c)\)

Deciding graph property \(\phi\) is in \(\Pi_1\) if and only if:
• \(G \models \phi \Rightarrow \forall c\text{ all nodes accept } (G,c)\)
• \(G \not\models \phi \Rightarrow \exists c\text{ at least one node rejects } (G,c)\)
The hierarchy \((\Sigma_k, \Pi_k)_{k \geq 0}\)

Deciding graph property \(\phi\) is in \(\Sigma_2\) if and only if:
- \(G \models \phi \Rightarrow \exists c_1 \forall c_2\text{ all nodes accept } (G,c_1,c_2)\)
- \(G \not\models \phi \Rightarrow \forall c_1 \exists c_2 \text{ at least one node rejects } (G,c_1,c_2)\)

Deciding graph property \(\phi\) is in \(\Pi_2\) if and only if:
- \(G \models \phi \Rightarrow \forall c_1 \exists c_2\text{ all nodes accept } (G,c_1,c_2)\)
- \(G \not\models \phi \Rightarrow \exists c_1 \forall c_2 \text{ at least one node rejects } (G,c_1,c_2)\)

Deciding graph property \(\phi\) is in \(\Sigma_k\) if and only if:
- \(G \models \phi \Rightarrow \exists c_1 \forall c_2 \exists c_3 \ldots Q c_k \text{ all nodes accept } (G,c_1,\ldots,c_k)\)
- \(G \not\models \phi \Rightarrow \forall c_1 \exists c_2 \forall c_3 \ldots \neg Q c_k \text{ at least one node rejects } (G,c_1,\ldots,c_k)\)

Deciding graph property \(\phi\) is in \(\Pi_k\) if and only if:
- \(G \models \phi \Rightarrow \forall c_1 \exists c_2 \forall c_3 \ldots Q c_k \text{ all nodes accept } (G,c_1,\ldots,c_k)\)
- \(G \not\models \phi \Rightarrow \exists c_1 \forall c_2 \exists c_3 \ldots \neg Q c_k \text{ at least one node rejects } (G,c_1,\ldots,c_k)\)
Example: Minimum Dominating Set

Decision problem MinDS:
- input = dominating set \( \mathcal{D} \) (i.e., \( \mathcal{D}(u) \in \{0,1\} \))
- output = accept if \( |\mathcal{D}| = \min_{\text{dom } \mathcal{D}} |\mathcal{D}| \)

**Theorem** \( \text{MinDS} \in \Pi_2 \)

**Proof**
\( c_1 \) encodes a dominating set, i.e., \( c_1(u) \in \{0,1\} \)
\( c_2 \) encodes:
- a spanning tree \( T_{\text{err}} \) pointing to node \( u \) with error in \( c_1 \) if any
- a spanning tree \( T_0 \) for counting \( |\mathcal{D}| \) (w/ same root)
- a spanning tree \( T_1 \) for counting \( |c_1| \) (w/ same root)

**Algorithm:**
- If root \( u \) sees \( |c_1| < |\mathcal{D}| \) with no error, it rejects, otherwise it accepts
- If any node detects inconsistencies in \( T_0, T_1 \) or \( T_{\text{err}} \) it rejects, otherwise it accepts.
Randomized Protocols

[FKP, 2013]

• At most one selected (AMOS)

• Decision algorithm (2-sided):
  - let \( p = \frac{\sqrt{5} - 1}{2} = 0.61\ldots \)
  - If not selected then accept
  - If selected then accept w/ prob \( p \), and reject w/ prob \( 1-p \)

• Issue with boosting! — But OK for 1-sided error
Distributed Interactive Protocols

• Arthur-Merlin Phase
  (no communication, only interactions)
• Verification Phase
  (only communications)

• Merlin has infinite computation power
• Arthur is randomized

• $k = \#\text{interactions}$
• $d\text{AM}[k]$ or $d\text{MA}[k]$
• $d\text{AM} = d\text{AM}[2]$
• $d\text{AMA} = d\text{MA}[3]$
Example: AMOS

- In BPLD with success probability \( (\sqrt{5}-1)/2 = 0.61 \ldots \)

- In \( \Sigma_1 \text{LD}(O(\log n)) \) — Not in \( \Sigma_1 \text{LD}(o(\log n)) \)

- Not in \( d\text{MA}(o(\log n)) \) for success prob > 4/5

- In \( d\text{AM}(r) \) with \( r \) random bits, and success prob \( 1-1/2^r \)
  - Arthur independently picks a \( r \)-bit index \( x_u \) at each node \( u \), u.a.r.
  - Merlin answer \( \bot \) if no nodes selected, or the index \( x_v \) of the selected node \( v \)
Sequential setting

- For every $k \geq 2$, $AM[k] = AM$

- $MA \subseteq AM$ because $MA \subseteq MAM = AM[3] = AM$

- $MA \subseteq \Sigma_2P \cap \Pi_2P$

- $AM \subseteq \Pi_2P$

- $AM[poly(n)] = IP = PSPACE$
Known results (1/2)

[KOS 2018, NPY 2018]

- $\text{Sym} \in d\text{AM}(n \log n)$
- $\text{Sym} \in d\text{MAM}(\log n)$
- Any $d\text{AM}$ protocol for $\text{Sym}$ requires $\Omega(\log \log n)$-bit certificates
- $\neg \text{Sym} \notin d\text{AMAM}(\log n)$
- Other results on graph non-isomorphism
Known results (2/2)

Diameter (unweighted graphs):

- diam 2 vs. 3 requires $\Omega(n)$ rounds in CONGEST
- diam 3 vs. 4 requires certificates on $\Omega(n)$ bits for $\Sigma_1\text{LD}$
- $\tilde{O}(n)$ bits suffices for $\Sigma_1\text{LD}$, even for weighted graphs
- diam 5 vs. 6 requires certificates on $\Omega(n)$ bits for dMA
  [FMORT, 2019]
Parameters

- Number of interactions between and
- Size of
- Size of
- Number of random
- Shared vs distributed
Limitations and Tradeoffs

[CFP, 2019]

• **Theorem 1** There exists a graph property that
  - admits a proof-labeling scheme ($\Sigma_1LD$) with certificates and messages on $O(n)$ bits,
  - but cannot be solved by an Arthur-Merlin (dAM) protocol with certificates on $o(n)$ bits, for any fixed number $k$ of interactions between Arthur and Merlin.

• **Theorem 2** For every $c$, there exists a Merlin-Arthur (dMA) protocol for *triangle-freeness*, using $O(\log n)$ bits of shared randomness, with $\tilde{O}(n/c)$-bit certificates and $\tilde{O}(c)$-bit messages between nodes.
Proof of Theorem 2

Every node solves set-disjointness with each of its neighbors

Use a protocol by Aaronson-Wigderson (2009), recently revisited by Abboud, Rubinstein & Williams (2017)

Assume IDs in $\{1, \ldots, n\} = \{1, \ldots, n/c\} \times \{1, \ldots, c\} = [n/c] \times [c]

Let $q = \Theta(nc)$ prime (possibly with a log factor).

Node $u$ represents $N(u)$ as $c$ functions $F_{u,t} : [n/c] \to \{0,1\}$ s.t.

$F_{u,t}(i) = 1 \iff (i,t) \in N(u)$

Interpolation by $c$ polynomials $P_{u,t} : \mathbb{F}_q \to \mathbb{F}_q$ of degree $n/c-1$.

$N(u) \cap N(v) = \emptyset \iff P_{u,t}(i) P_{v,t}(i) = 0$ for every $i \in [n/c]$ and $t \in [c]$
Let $P_{u,v,t} = P_{u,t}P_{v,t}$ for every $v \in N(u)$ and $t \in [c]$

Let $P_u = \Sigma_{t \in [c]} \Sigma_{v \in N(u)} P_{u,v,t}$ of degree $\leq 2(n/c-1)$

Rmk: $u$ is not part of a triangle $\iff P_u(i) = 0$ for every $i \in [n/c]$

Merlin assigns polynomial $Q_u$ to node $u$ using $O(n/c \log q)$ bits. Arthur at node $u$ checks that:

(1) $Q_u(i) = 0$ for every $i \in [n/c]$

(2) $Q_u = P_u$

For (2), node $u$ picks $i^*$ u.a.r. in $\mathbb{F}_q$ and sends $\{ P_{u,t}(i^*), t \in [c] \}$ to all its neighbors, consuming bandwidth $O(c \log q)$ bits.

Node $u$ then computes $P_u(i^*) = \Sigma_{t \in [c]} \Sigma_{v \in N(u)} P_{u,t}(i^*)P_{v,t}(i^*)$

Node $u$ accepts if (1) and $Q_u(i^*) = P_u(i^*)$, and rejects otherwise.

The probability that two non-equal polynomials on $\mathbb{F}_q$ of degree at most $2(n/c-1)$ are equal at a random point $i^*$ is at most $2(n/c-1)/q < 1/3$ as $q = \Theta(nc)$. $\blacksquare$
Congested Clique
Definition

CONGEST model, but on a clique!

Unicast variant

Broadcast variant

potentially different messages

same messages
Graph Problems in the Congested Clique
Lower bound in the Broadcast Congested Clique

**Theorem** (Drucker, Kuhn & Oshman, 2014)
Deciding $C_4$-freeness required sending $\Omega(\sqrt{n})$ bits between some neighbors in the Broadcast Congested Clique.
Lower bound in the Unicast Congested Clique

To date, no lower bounds for this model are known...

**Theorem** (*informal* - Drucker, Kuhn & Oshman, 2014))
The unicast congested clique can « *simulate* » « *powerful* » classes of bounded-depth circuits.

It follows that even slightly super-constant lower bounds for the unicast congested clique would give new lower bounds in circuit complexity.
Algebraic Topology
(a.k.a. Combinatorial Topology)
Ad Hoc Approach
Binary Consensus in the LOCAL model

• For every node $u$, input $= x(u) \in \{0,1\} \rightarrow u$ proposes $x(u)$

• For every node $u$, output $= y(u) \in \{0,1\} \rightarrow u$ decides $y(u)$

• Two conditions:
  - **Validity**: The decided value must have been proposed by at least one node
  - **Agreement**: All nodes must agree on their outputs, i.e., they must be identical
Lower bound

- eccentricity: $\text{ecc}_G(u) = \max_v \text{dist}_G(u,v)$
- diameter($G$) = $\max_u \text{ecc}_G(u)$
- radius($G$) = $\min_u \text{ecc}_G(u)$

**Theorem** For every network $G$, consensus in $G$ requires at least $\text{radius}(G)$ rounds.
Proof

Assume \#rounds < radius(G)

Input configuration: \( x(u_1)x(u_2)\ldots x(u_n) \)

An edge between two input configurations indicates that these configurations are indistinguishable from at least one node.
k-set agreement

- $m \geq k \geq 1$

- For every node $u$, input $= x(u) \in \{0, 1, \ldots, m\}$,

- For every node $u$, output $= y(u) \in \{0, 1, \ldots, m\}$

- Two conditions:
  - **Validity**: Any decided value must have been proposed by at least one node
  - **Agreement**: At most $k$ values must be decided
A dominating set in $G=(V,E)$ is a set $D \subseteq V$ such that every node not in $D$ has a neighbor in $D$.

**Definition** $G$ has dominating number $d$ if the min size of a dominating set in $G$ has cardinality $= d$.

**Theorem** $k$-set agreement in $G$ requires at least $r$ rounds where $r$ is the minimum integer such that $G^r$ has dominating number $\leq k$. 

Lower bound
Proof for $m=3$ and $k=2$

Input configuration: $v_1v_2...v_n$ with $v_i \in \{0,1,2\}$

For every $i,j$, there exists process $q$ that is not dominated by $p_i$ nor $p_j$.

These triangles can be glued together.
Assume existence of an algorithm.

- Color each node by the discarded color

- Remark: Impossible

The coloring of the border nodes is forced
Sperner’s Lemma

**Lemma** Every Sperner coloring of a subdivision of an $n$-dimensional simplex contains a cell colored with a complete set of colors.
Proof (for n=2)

\[ V(G) = \{ \circ \} \quad E(G) = \{ \circ - \circ \} \]

- By induction on n: \( \text{deg}(u) \) is odd
- \( \sum_{v \in V(G)} \text{deg}(v) = 2 |E(G)| \)
- Triangles with 1 or 2 colors induce nodes with even degrees (0 or 2)
  \[ \downarrow \]
  Odd number of 3-colored triangles
General Approach
Basic definitions

• A simplicial complex is a collection $\mathcal{K}$ of subsets of a set $V$, closed under containment:
  \[( A \in \mathcal{K} \text{ and } B \subseteq A ) \Rightarrow B \in \mathcal{K}\]

• An element $u \in V$ is a vertex
• An set $S \in \mathcal{K}$ is a simplex
• $\dim(S) = |S| - 1$
• $\dim(\mathcal{K}) = \max_S \dim(S)$
Complex and system configuration

At any given time of the computation, all possible states of a distributed system can be modeled as a single simplicial complex:

- Vertex = \((p, s)\) where \(p\) = name, and \(s\) = individual state
- \(S = \{(p_1, s_1), \ldots, (p_k, s_k)\}\) with \(k \leq n\) forms a simplex if
  1. \(p_i \neq p_j\) for every \(i \neq j\)
  2. There is a configuration of the system in which \(p_1, p_2, \ldots, p_k\) are respectively in state \(s_1, s_2, \ldots, s_k\).
Task

**Definition** A task is described by \((I,O,\Delta)\) where 
\(I\) is the input complex, \(O\) is the output complex, and 
\(\Delta: I \rightarrow 2^O\) is a carrier map:

\[
\{(p_1,y_1),\ldots,(p_k,y_k)\} \in \Delta(\{(p_1,x_1),\ldots,(p_k,x_k)\})
\]

specifies that \(\{(p_1,y_1),\ldots,(p_k,y_k)\}\) is a legal output for 
the input \(\{(p_1,x_1),\ldots,(p_k,x_k)\}\)

Example: Consensus
Protocol complex

**Definition** $\mathcal{P}(t)$ denotes the protocol complex after $t$ rounds, which describes all possible configurations of the system after $t$ rounds: $\{(p_1,s_1),\ldots,(p_k,s_k)\}$ is a simplex of $\mathcal{P}(t)$ if $p_1,p_2,\ldots,p_k$ can respectively be in states $s_1,s_2,\ldots,s_k$ after $t$ rounds.

**Remark:** The protocol complex is a *deformation* of the input complex, and conceptually understanding distributed computing can be done through the study of this deformation.
Example

Restricted variant of LOCAL model in which every process
- has a name (no IDs)
- knows the structure of the network G
- knows which node each process occupies in G

Input complex

Protocol complex for $C_3$
Impact of structure

Input complex

Protocol complex for $C_3$

Protocol complex for $S_3$
Task solvability

Input complex \rightarrow \text{Protocol complex} \rightarrow \text{Output complex}

\begin{align*}
\text{Input complex} & \quad \Xi \quad \text{execution map} \quad \Xi \\
\text{Output complex} & \quad \delta \quad \text{decision map} \\
\end{align*}

\text{Task specification}

\begin{align*}
\text{Protocol complex} & \quad \Delta \\
\end{align*}

\text{carrier map}

\text{ Depends on the model}

\text{ Depends on the algorithm}
Decision map

• In a \( t \)-round algorithm, every process \( p \) must decide an output based on its states after \( t \) round.

• This induces a name-preserving mapping

\[
\delta : \mathcal{P}^t \to O
\]

\[
\delta(p, s) = (p, v)
\]

where \( v \) is the output value of \( p \) in state \( s \).
Simplical map

Let $\mathcal{K}$ and $\mathcal{L}$ be two simplical complexes.

**Definition** Vertex-map $f : V(\mathcal{K}) \rightarrow V(\mathcal{L})$ is a simplicial map if $f$ maps every simplex of $\mathcal{K}$ to a simplex of $\mathcal{L}$.

In « colored complexes » (e.g., by names), one is interested in color-preserving mappings.
Decision map is simplicial

Process \( \bullet \) does not know in which of the four configurations the system is after \( t \) rounds.

Its output must agree with all configurations compatible with its state.
Characterization

**Theorem** (Herlihy and Shavit, 1998) **Gödel Prize 2004**

Task \((I, O, \Delta)\) is solvable iff there exists a simplicial map \(\delta : P^t \rightarrow O\) which agrees with \(\Delta\), that is, for every simplex \(S\) of \(I\),

\[\delta(\Xi(S)) \in \Delta(S).\]
Application to consensus

**Theorem** In the restricted variant of the LOCAL model, consensus is solvable in one round if and only if $G$ has a universal vertex.

**Lemma 0** If $G$ has a universal vertex then consensus is solvable.

**Lemma 1** If consensus is solvable in one round then $P^1$ is disconnected.

**Lemma 2** If $P^1$ is disconnected then $G$ has a universal vertex.
Proof of Lemma 1

Assume $P^1$ is connected

Protocol complex

Output complex

$\delta$ cannot be simplicial

Consensus is not solvable
Proof of Lemma 2
(skbetch)

Show that if $G$ has no universal vertex then $\mathcal{P}_1$ is connected
Research Directions
Research directions in the framework of network computing

- **Graph problems**: designing new algorithms, improving existing lower bounds, studying new problems, etc. (variety of models: LOCAL, CONGEST, congested clique, mobile agents, etc.).

- **Algebraic topology** applied to network computing (e.g., to LOCAL model):
  - Better conceptual understanding
  - Deriving better lower bounds
Conclusion
Conclusion

**Distributed algorithms**
- Coping with uncertainty

**Tools**
- Algorithms
- Graph theory
- Combinatorics
- Algebraic topology
- Communication complexity
Partial exam
Dec 3, 9h-12h