Presentation and Analysis of Some Distributed Randomised Algorithms

Habilitation à diriger des recherches

Akka Zemmari

LaBRI - Université Bordeaux 1

October 19, 2009
Research Field

- Design and Analysis of (Randomised) Distributed Algorithms
  - Graph Decomposition
  - Computing by Mobile Agents
  - Computing in Sensor Networks
  - Election Problem
  - Bit Complexity
    - Maximal Independent Set (MIS) Problem
    - Colouring Problem
  - Handshake and Matching Problems
Research Field

- Design and Analysis of (Randomised) Distributed Algorithms
  - Graph Decomposition
  - Computing by Mobile Agents
  - Computing in Sensor Networks
  - Election Problem
  - Bit Complexity
    - Maximal Independent Set (MIS) Problem
    - Colouring Problem
  - Handshake and Matching Problems
Outline

1. Election Problem
2. Bit Complexity of the MIS Problem
3. Colouring Problem
4. Conclusion and Perspectives
Introduction
Introduction
Introduction
Introduction
Introduction
Election in Trees
Election in Trees
Election in Trees
Election in Trees

[Diagram of trees with nodes and edges, illustrating the concept of election in trees]
Election in Trees
Election in Trees
Election in Trees

- A tree $T$ is given;
- while $T$ is not reduced to a leaf do remove a leaf from $T$;
- The elected vertex is the surviving one.

Goal of study
- To determine the chances of being elected for each vertex.
- To guide the “ballot” toward a given probability distribution by imposing “locally computable” control structures.
Election in Trees

- A tree $T$ is given;
- while $T$ is not reduced to a leaf do remove a leaf from $T$;
- The elected vertex is the surviving one.

Goal of study

- To determine the chances of being elected for each vertex.
- To guide the “ballot” toward a given probability distribution by imposing “locally computable” control structures.
A tree $T$ is given;
while $T$ is not reduced to a leaf do remove a leaf from $T$

The *elected vertex* is the surviving one.

**Goal of study**

- To determine the chances of being elected for each vertex.
- To guide the “ballot” toward a given probability distribution by imposing “locally computable” control structures.
A tree $T$ is given;
while $T$ is not reduced to a leaf do remove a leaf from $T$
The elected vertex is the surviving one.

Goal of study
- To determine the chances of being elected for each vertex.
- To guide the “ballot” toward a given probability distribution by imposing “locally computable” control structures.
Election in Trees

- A tree $T$ is given;
- while $T$ is not reduced to a leaf do remove a leaf from $T$;
- The *elected vertex* is the surviving one.

**Goal of study**

- To determine the chances of being elected for each vertex.
- To guide the “ballot” toward a given probability distribution by imposing “locally computable” control structures.
Election in Trees

- A tree $T$ is given;
- while $T$ is not reduced to a leaf do remove a leaf from $T$
- The elected vertex is the surviving one.

**Goal of study**

- To determine the chances of being elected for each vertex.
- To guide the “ballot” toward a given probability distribution by imposing “locally computable” control structures.
Combinatorial Approach [Y. Métivier and N. Saheb in CAAP’94]

All leaf-removal sequences have the same probability:

\[ p_x(T) = \frac{C_x^T}{C_T^\emptyset}. \]

\( C_x^T \): the number of leaf-removal sequences yielding to the election of \( x \)

\( C_T^\emptyset \): the total number of leaf-removal sequences.
Combinatorial Approach [Y. Métivier and N. Saheb in CAAP’94]

All leaf-removal sequences have the same probability:

\[ p_x(T) = \frac{C_x^T}{C_T^\emptyset}. \]

\( C_x^T \): the number of leaf-removal sequences yielding to the election of \( x \)
\( C_T^\emptyset \): the total number of leaf-removal sequences.
Election in Trees

Combinatorial Approach

Theorem.
In a tree, the median vertex (ices) has (have) the highest probability of being elected.
Election in Trees

Locally Uniform Approach

At each step all leaves have the same probability of being removed.
Election in Trees

Locally Uniform Approach

At each step all leaves have the same probability of being removed.

\[
\begin{align*}
\frac{1}{12} & \quad \frac{11}{36} & \quad \frac{5}{12} \\
\quad & \quad \quad & \\
\frac{7}{72} \quad \frac{7}{72} & \\
\end{align*}
\]
Election in Trees

Locally Uniform Approach

At each step all leaves have the same probability of being removed.

\[ q_x (\{x\}) = 1. \]

\[ q_x (T) = \frac{1}{|F(T)|} \sum_{f \in F(T), f \neq x} q_x (T - \{f\}). \]
Election in Trees

Locally Guided Election

Each leaf $v$ is active:
- it has a lifetime $L(v)$ which is exponentially distributed

\[
\Pr (L(v) \leq t) = 1 - e^{-\lambda(v)t}, \quad \forall t \in \mathbb{R}^+.
\]

\[\iff\]

probability of its death in the interval $[t, t + h)$ is $\lambda(v)h + o(h)$, as $h \to 0$.

$\lambda(v)$: death rate of $v$. It must be determined at the instant where $v$ has become a leaf by the local information received from the removed leaves on its side.
Question:

- How to choose the $\lambda$-assignment to guide the “ballot” towards a given probability distribution over the set of vertices?
- In particular, is it possible to guide it towards a uniform election?
Question:

- How to choose the \( \lambda \)-assignment to guide the “ballot” towards a given probability distribution over the set of vertices?

- In particular, is it possible to guide it towards a uniform election?
Election in Trees

An application: Uniform Model

- Initially all vertices $v$ have the same weight $\omega(v) = 1$. (Anonymity)
- Once a leaf has been vanished, its father recuperates its weight.
- The death rate for a leaf $v$ is its weight: $\lambda(v) = \omega(v)$. 
Theorem.

In the last model of election, all vertices of the tree have the same probability $\frac{1}{n}$ of being elected.
Election in Trees

Election Duration

Let $T$ a tree of size $n$, and $D_n$ the uniform election duration on $T$.

**Lemma.**

The density function of the random variable $D_n$ is

$$f(t) = n(n - 1)e^{-2t} \left(1 - e^{-t}\right)^{n-2}.$$ 

**Theorem.**

The expected value of $D_n$ is

$$\sum_{i=2}^{n} \frac{1}{i} = H_n - 1,$$

where $H_n$ is the $n$th harmonic number.
Election in Trees

Election Duration

Let $T$ a tree of size $n$, and $D_n$ the uniform election duration on $T$.

Lemma.
The density function of the random variable $D_n$ is

$$f(t) = n(n - 1)e^{-2t}(1 - e^{-t})^{n-2}.$$ 

Theorem.
The expected value of $D_n$ is $\sum_{i=2}^{n} \frac{1}{i} = H_n - 1$, where $H_n$ is the $n$th harmonic number.
Election in Trees

Generalisation

- To other classes of graphs
- Each vertex $v$ has a weight $\omega(v) \neq 1$. 
Election in Trees

Other classes of graphs

- $k$-trees
- Polyominoids
To more general classes?

- How to distinguish vertices which can start generating a life duration?
- When a vertex vanishes, which vertex will collect the transmitted weight?
Election in Trees

Non uniform weights

- $\omega(v)$ is an integer
- $\omega(v) \in \mathbb{R}_+^*$.

$\Rightarrow$ The probability that a vertex $v$ is elected is proportional to its initial weight.
Election in Trees

Non uniform weights

- $ω(v)$ is an integer
- $ω(v) ∈ \mathbb{R}^*_+$. 

$⇒$ The probability that a vertex $v$ is elected is proportional to its initial weight.
Outline

1. Election Problem

2. Bit Complexity of the MIS Problem

3. Colouring Problem

4. Conclusion and Perspectives
The MIS Problem

Definition.

Let $G = (V, E)$ be a graph. An independent set is a subset $U$ of $V$ s.t.,

$$\forall u, v \in U, \{u, v\} \notin E,$$

$U$ is maximal if $\forall v \in V \setminus U, U \cup \{v\}$ is not an independent set.
The MIS Problem

Definition.

Let $G = (V, E)$ be a graph. An independent set is a subset $U$ of $V$ s.t.,

$$\forall u, v \in U, \ {u, v} \notin E,$$

$U$ is maximal if $\forall v \in V \setminus U$, $U \cup \{v\}$ is not an independent set.
The graph is supposed anonymous.

We propose a *distributed* and *randomised* algorithm.

We study:

- its average complexity,
- its bit complexity,
A General Schema
A General Schema
A General Schema
A General Schema
A General Schema
## Related Works and Comparisons

<table>
<thead>
<tr>
<th>Knowledge</th>
<th>Time (on average)</th>
<th>Message size (number of bits)</th>
<th>Bit complexity (per channel)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Luby (Lynch)</td>
<td>$O(\log n)$</td>
<td>$\log n$</td>
<td>$O(\log^2 n)$</td>
</tr>
<tr>
<td>Luby (Peleg)</td>
<td>$O(\log^2 n)$</td>
<td>$\log n$</td>
<td>$O(\log^3 n)$</td>
</tr>
<tr>
<td>Luby (Wattenhofer)</td>
<td>$O(\log n)$</td>
<td>$\log n$</td>
<td>$O(\log^2 n)$</td>
</tr>
<tr>
<td>Algorithm C</td>
<td>$O(\log n)$</td>
<td>1</td>
<td>$O(\log n)$</td>
</tr>
</tbody>
</table>
## Related Works and Comparisons

<table>
<thead>
<tr>
<th></th>
<th>Knowledge</th>
<th>Time (on average)</th>
<th>Message size (number of bits)</th>
<th>Bit complexity (per channel)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Luby (Lynch)</td>
<td>Size of the graph</td>
<td>$O(\log n)$</td>
<td>$\log n$</td>
<td>$O(\log^2 n)$</td>
</tr>
<tr>
<td>Luby (Peleg)</td>
<td>Maximum degree in 2-Neighbourhood</td>
<td>$O(\log^2 n)$</td>
<td>$\log n$</td>
<td>$O(\log^3 n)$</td>
</tr>
<tr>
<td>Luby (Wattenhofer)</td>
<td>Maximum of neighbours degrees</td>
<td>$O(\log n)$</td>
<td>$\log n$</td>
<td>$O(\log^2 n)$</td>
</tr>
<tr>
<td>Algorithm C</td>
<td>no knowledge</td>
<td>$O(\log n)$</td>
<td>1</td>
<td>$O(\log n)$</td>
</tr>
</tbody>
</table>
## Related Works and Comparisons

<table>
<thead>
<tr>
<th>Knowledge</th>
<th>Time (on average)</th>
<th>Message size (number of bits)</th>
<th>Bit complexity (per channel)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size of the graph</td>
<td>$O(\log n)$</td>
<td>log $n$</td>
<td>$O(\log^2 n)$</td>
</tr>
<tr>
<td>Maximum degree in 2-Neighbourhood</td>
<td>$O(\log^2 n)$</td>
<td>log $n$</td>
<td>$O(\log^3 n)$</td>
</tr>
<tr>
<td>Maximum of neighbours degrees</td>
<td>$O(\log n)$</td>
<td>log $n$</td>
<td>$O(\log^2 n)$</td>
</tr>
</tbody>
</table>
## Related Works and Comparisons

<table>
<thead>
<tr>
<th></th>
<th>Knowledge</th>
<th>Time (on average)</th>
<th>Message size (number of bits)</th>
<th>Bit complexity (per channel)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Luby (Lynch)</td>
<td>Size of the graph</td>
<td>$O(\log n)$</td>
<td>$\log n$</td>
<td>$O(\log^2 n)$</td>
</tr>
<tr>
<td>Luby (Peleg)</td>
<td>Maximum degree in 2-Neighbourhood</td>
<td>$O(\log^2 n)$</td>
<td>$\log n$</td>
<td>$O(\log^3 n)$</td>
</tr>
<tr>
<td>Luby (Wattenhofer)</td>
<td>Maximum of neighbours degrees</td>
<td>$O(\log n)$</td>
<td>$\log n$</td>
<td>$O(\log^2 n)$</td>
</tr>
<tr>
<td>Algorithm $C$</td>
<td>no knowledge</td>
<td>$O(\log n)$</td>
<td>1</td>
<td>$O(\log n)$</td>
</tr>
</tbody>
</table>
Algorithm $\mathcal{A}$: A Real Based Algorithm

A phase of the algorithm:

- each vertex $u$, still in the graph, generates a random variable $x(u)$;
- $u$ sends $x(u)$ to all its neighbours still in the graph;
- $u$ receives $x(v)$ from each neighbour $v$, still in the graph;

$u$ is included in the independent set if its $x$ is a local minimum, i.e., $x(u) < x(v)$ for each neighbour $v$ of $u$
Lemma.

In any phase, the expected number of edges removed from the remaining graph $G$ is at least half the number of edges in $G$.

Corollary.

There are constants $k_1$ and $K_1$ such that for any graph $G = (V, E)$ of $n$ vertices the number of phases to remove all edges from $G$ is:

1. less than $k_1 \log n$ on average,
2. less than $K_1 \log n$ with probability $1 - o(n^{-1})$. 
Analysis of Algorithm $\mathcal{A}$

Lemma.
In any phase, the expected number of edges removed from the remaining graph $G$ is at least half the number of edges in $G$.

Corollary.
There are constants $k_1$ and $K_1$ such that for any graph $G = (V, E)$ of $n$ vertices the number of phases to remove all edges from $G$ is:

1. less than $k_1 \log n$ on average,
2. less than $K_1 \log n$ with probability $1 - o(n^{-1})$. 
Algorithm $B$: A Bit Based Algorithm

**Main idea:** simulates exchanges of real numbers by exchanges of bits which define real numbers, most significant first.

*One phase of exchange of real numbers is replaced by a phase* composed of a sequence of rounds.
Algorithm $B$: A Bit Based Algorithm

In each round:

- $u$ generates uniformly one bit $b(u) \in \{0, 1\}$;
- $u$ sends $b(u)$ to all its active neighbours;
- $u$ receives $b(v)$ from each active neighbour $v$;
- $u$ makes a decision:
  - $u$ is $IN - MIS$
  - $u$ is $NOT - IN - MIS$
  - $u$ is $INELIGIBLE$: when a vertex $u$ sends this message it means that until the end of the current phase $u$ cannot be in the MIS.
Theorem.

Algorithm $B$ constructs an MIS for any arbitrary graph of size $n \geq 1$ in $O(\log^2 n)$ exchanges of bits on average and with high probability.
Algorithm $C$: An Optimal Bit Complexity Algorithm

**Main Idea:** desynchronisation of phases between edges in algorithm $B$

If, in a round, $u$ breaks the symmetry with $v_1$ and does not break the symmetry with $v_2$ then $u$ considers that a phase with $v_1$ is completed and starts, in anticipation, a new phase with $v_1$ and it continues the previous phase with $v_2$. 
Algorithm $C$: An Optimal Bit Complexity Algorithm

**Main Idea:** desynchronisation of phases between edges in algorithm $B$

If, in a round, $u$ breaks the symmetry with $v_1$ and does not break the symmetry with $v_2$ then $u$ considers that a phase with $v_1$ is completed and starts, in anticipation, a new phase with $v_1$ and it continues the previous phase with $v_2$. 
Theorem.
The randomised distributed MIS Algorithm $C$ for arbitrary graphs of size $n$ halts in time $O(\log n)$ with probability $1 - o(n^{-1})$, each message containing 1 bit.

Corollary.
The bit complexity per channel of Algorithm $C$ is $O(\log n)$. 
Analysis of Algorithm $C$

Theorem.
The randomised distributed MIS Algorithm $C$ for arbitrary graphs of size $n$ halts in time $O(\log n)$ with probability $1 - o(n^{-1})$, each message containing 1 bit.

Corollary.
The bit complexity per channel of Algorithm $C$ is $O(\log n)$.
Outline

1. Election Problem
2. Bit Complexity of the MIS Problem
3. Colouring Problem
4. Conclusion and Perspectives
The Problem

Definition.

Let $G = (V, E)$ be a simple connected undirected graph. A proper vertex colouring for $G$ is an assignment of a colour $c(v)$ to each vertex $v$, such that any two adjacent vertices have a different colour, i.e., $c(v) \neq c(u)$ for every $\{u, v\} \in E$. 
The Problem

Definition.

Let $G = (V, E)$ be a simple connected undirected graph. A proper vertex colouring for $G$ is an assignment of a colour $c(v)$ to each vertex $v$, such that any two adjacent vertices have a different colour, i.e., $c(v) \neq c(u)$ for every $\{u, v\} \in E$. 

![Diagram of a graph with vertex colouring examples](image-url)
Related Works

Johansson’s Algorithm
Related Works

Johansson’s Algorithm
Johansson’s Algorithm
Related Works

Johansson’s Algorithm

![Diagrams showing Johansson's Algorithm with sets {0,1,2,3} and {0,1,2} at different steps.](image-url)

- For {0,1,2,3} and {0,1,2}, the algorithm progresses through various steps, demonstrating the coloring and selection process.

---

Akka Zemmari  
Habilitation à diriger des recherches


Johansson’s Algorithm

Related Works
Related Works

Johansson’s Algorithm
Related Works

Johansson’s Algorithm
Theorem.
Johansson’s Algorithm runs in $O(\log n)$ rounds on average and w.h.p.

Remark.
Messages are of size $O(\log n)$, and hence, its average bit complexity is $O(\log^2 n)$. 
Theorem.
Johansson’s Algorithm runs in $O(\log n)$ rounds on average and w.h.p.

Remark.
Messages are of size $O(\log n)$, and hence, its average bit complexity is $O(\log^2 n)$. 
Theorem. [Kothapalli et al.]

If only one bit can be sent along each edge in a round, then every distributed vertex colouring algorithm (in which every node has the same initial state and initially only knows its own edges) needs at least $\Omega(\log n)$ rounds w.h.p.
Algorithm *Fast\_Colour*
Algorithm *Fast\_Colour*
Algorithm *Fast_Colour*
Algorithm *Fast Colour*
Algorithm *Fast_Colour*

![Graphs](image-url)
Algorithm *Fast Colour*

![Image of graph structures with binary codes]

- **Algorithm Fast Colour**
  - **Election Problem**
  - **Bit Complexity of the MIS Problem**
  - **Colouring Problem**
  - **Conclusion and Perspectives**
Algorithm \textit{Fast\_Colour}
Analysis of the Algorithm

Expected Time Complexity

Lemma.
In any phase of the algorithm, the expected number of edges removed from the residual graph $G$ is half the number of its edges.

Corollary.
There are constants $k_1$ and $K_1$ such that for any graph $G$ of $n$ vertices, the number of phases to remove all edges from $G$ is:
- less than $k_1 \log n$ on average,
- less than $K_1 \log n$ w.h.p.
Expected Time Complexity

Lemma.
In any phase of the algorithm, the expected number of edges removed from the residual graph $G$ is half the number of its edges.

Corollary.
There are constants $k_1$ and $K_1$ such that for any graph $G$ of $n$ vertices, the number of phases to remove all edges from $G$ is:

- less than $k_1 \log n$ on average,
- less than $K_1 \log n$ w.h.p.
Analysis of the Algorithm

Theorem.

Algorithm *Fast_Colour* computes a colouring for any arbitrary graph of size $n$ in time $O(\log n)$ w.h.p., each message containing 1 bit.
Analysis of the Algorithm

Local Complexity

Proposition

Let $G = (V, E)$ be a connected graph and $v \in V$ with $d(v) = d \to \infty$. Let $L_d$ denote the number of bits generated by $v$ before it obtains its final colour. Then

$$\mathbb{E}(L_d) = \log_2 d + \frac{1}{2} + \frac{\gamma}{\log 2} + Q(\log_2 d) + O(d^{-2}),$$

where $Q(u) = -\frac{1}{\log 2} \sum_{k \in \mathbb{Z}\setminus\{0\}} \Gamma\left(\frac{2ik\pi}{\log 2}\right) e^{-2ik\pi u}$ is a Fourier series with period 1 and with an amplitude which does not exceed $10^{-6}$. 
Analysis of the Algorithm

Local Complexity

Lemma.
If $d \to \infty$, then we have

$$\text{Var} (L_d) = \left( \frac{1}{\log 2} - 1 \right) \log_2 d + \frac{1}{12} + \frac{1}{6} \frac{\pi^2}{(\log 2)^2} - P(\log_2 d) + O\left(d^{-2}\right),$$

where $P(u) = Q(u)^2 + \left(2u + \frac{2\gamma}{\log 2} - \frac{2}{\log 2}\right) Q(u)$.

Proposition
The ratio $R_d$ between $L_d$ and $\log_2 d$ tends in probability to 1 as $d$ tends to $\infty$. 
Analysis of the Algorithm

Local Complexity

Lemma.

If $d \to \infty$, then we have

$$\text{Var} (L_d) = \left( \frac{1}{\log 2} - 1 \right) \log_2 d + \frac{1}{12} + \frac{\pi^2}{6 (\log 2)^2} - P (\log_2 d) + O (d^{-2}),$$

where $P(u) = Q(u)^2 + \left( 2u + \frac{2\gamma}{\log 2} - \frac{2}{\log 2} \right) Q(u)$. 

Proposition

The ratio $R_d$ between $L_d$ and $\log_2 d$ tends in probability to 1 as $d$ tends to $\infty$. 

Outline

1. Election Problem
2. Bit Complexity of the MIS Problem
3. Colouring Problem
4. Conclusion and Perspectives
Election Problem
- is there any “totally fair” algorithm for any class of graph?
- can we design a one-pass distributed algorithm which implements the $p$-distribution?

Bit Complexity
- of the 2-MIS problem?
- of the distance 2 colouring problem?

Colouring Problem
- the main challenge: reduce the number of colours
Conclusion and Perspectives

- Election Problem
  - is there any “totally fair” algorithm for any class of graph?
  - can we design a one-pass distributed algorithm which implements the $p$-distribution?

- Bit Complexity
  - of the 2-MIS problem?
  - of the distance 2 colouring problem?

- Colouring Problem
  - the main challenge: reduce the number of colours
Conclusion and Perspectives

- **Election Problem**
  - is there any “totally fair” algorithm for any class of graph?
  - can we design a one-pass distributed algorithm which implements the $p$-distribution?

- **Bit Complexity**
  - of the 2-MIS problem?
  - of the distance 2 colouring problem?

- **Colouring Problem**
  - the main challenge: reduce the number of colours
Election Problem


Related Publications

- **MIS Problem**

- **Colouring Problem**
Thank You