About Randomised Distributed Graph Colouring and Graph Partition Algorithms

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari

GT AlgoDist
June 21, 2009

* to appear in Information and Computation
Outline

**Problems**
- Distributed Algorithms for the Colouring Problem.
- Distributed Algorithms for the Graph Partition Problem.
- Bit Complexity.

**Hypothesis**
- Anonymous networks
- No global knowledge

⇒ Probabilistic Solutions.
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 $\{u, v\} \in E$. 

---

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari

Colouring and Partition
Johansson’s Algorithm
Johansson’s Algorithm

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari
Johansson’s Algorithm
Johansson’s Algorithm
Johansson’s Algorithm

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari  
Colouring and Partition
Johansson’s Algorithm
Johansson’s Algorithm

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari

Colouring and Partition
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 $O(\log^2 n)$. 

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari
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 \textit{Fast\_Colour}
Algorithm \textit{Fast\_Colour}
Algorithm *Fast\_Colour*

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari

Colouring and Partition
Algorithm \textit{Fast\_Colour}

\begin{figure}
\centering
\begin{subfigure}{0.3\textwidth}
\centering
\begin{tikzpicture}
\node (A) at (0,0) [circle, fill=black] {};
\node (B) at (1,0) [circle, fill=black] {};
\node (C) at (1,1) [circle, fill=black] {};
\node (D) at (0,-1) [circle, fill=black] {};
\node (E) at (-1,1) [circle, fill=black] {};
\node (F) at (-1,-1) [circle, fill=black] {};
\draw (A) -- (B) -- (C) -- (D) -- (E) -- (F) -- (A);
\end{tikzpicture}
\end{subfigure}
\begin{subfigure}{0.3\textwidth}
\centering
\begin{tikzpicture}
\node (A) at (0,0) [circle, fill=black] {};
\node (B) at (1,0) [circle, fill=black] {};
\node (C) at (1,1) [circle, fill=black] {};
\node (D) at (0,-1) [circle, fill=black] {};
\node (E) at (-1,1) [circle, fill=black] {};
\node (F) at (-1,-1) [circle, fill=black] {};
\draw (A) -- (B) -- (C) -- (D) -- (E) -- (F) -- (A);
\end{tikzpicture}
\end{subfigure}
\begin{subfigure}{0.3\textwidth}
\centering
\begin{tikzpicture}
\node (A) at (0,0) [circle, fill=black] {};
\node (B) at (1,0) [circle, fill=black] {};
\node (C) at (1,1) [circle, fill=black] {};
\node (D) at (0,-1) [circle, fill=black] {};
\node (E) at (-1,1) [circle, fill=black] {};
\node (F) at (-1,-1) [circle, fill=black] {};
\draw (A) -- (B) -- (C) -- (D) -- (E) -- (F) -- (A);
\end{tikzpicture}
\end{subfigure}
\end{figure}
Algorithm *Fast-Colour*

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari

Colouring and Partition
Algorithm \textit{Fast\_Colour}

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari

Colouring and Partition
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.
### 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.
Reducing the Number of Colors - Algorithm $\mathcal{R}$
Reducing the Number of Colors - Algorithm $\mathcal{R}$
Reducing the Number of Colors - Algorithm $\mathcal{R}$

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari

Colouring and Partition
Reducing the Number of Colors - Algorithm $\mathcal{R}$
Reducing the Number of Colors - Algorithm $\mathcal{R}$

Coloring and Partition
Reducing the Number of Colors - Algorithm $\mathcal{R}$
Theorem

For any graph $G = (V, E)$ having a maximum degree $\Delta$, Algorithm $\mathcal{R}$ achieves a $(\Delta + 1)$-colouring of $G$ in at most $e\Delta + 2 \log n + \log^* n$ rounds w.h.p.
Theorem. (Time Complexity of Stage 1.)

Let $G = (V, E)$ be a ring of size $n \geq 3$. Let $T_1$ denote the number of rounds necessary to colour all vertices by Algorithm \textit{Fast\_Colour}.

- The expected value of $T_1$ is asymptotically equal to $\log_2 n - \frac{1}{2} + \frac{\gamma}{\log 2} + Q (\log_2 n) + O \left( n^{-2} \right)$, where $Q \left( u \right) = -\frac{1}{\log 2} \sum_{k \in \mathbb{Z} \setminus \{0\}} \Gamma \left( \frac{2ik\pi}{\log 2} \right) e^{-2ik\pi u}$ is a Fourier series with period 1 and with an amplitude which does not exceed $10^{-6}$.
- It is less than $2\log_2 n$ w.h.p.
Lemma

Let $G = (V, E)$ be a ring graph of size $n \geq 3$ and $T_1$ be the r.v. defined above. Then:

- $P(T_1 = 0) = 0$,
- If $n$ is odd, then: $P(T_1 = 1) = 0$, and $P(T_1 = 2) = \frac{3^n - 3}{4^n}$.
- If $n$ is even, then: $P(T_1 = 1) = \frac{1}{2^{n-1}}$, and $P(T_1 = 2) = \frac{3^n + 3}{4^n} - \frac{1}{2^{n-1}}$.

and, for any $k > 2$:

$P(T_1 = k) = (1 - \frac{1}{2^k})^n - (1 - \frac{1}{2^{k-1}})^n + (-1)^{n+1} \left( \left( \frac{1}{2^{n-1}} \right)^{k-1} - \left( \frac{1}{2^n} \right)^{k-1} - \left( \frac{1}{2^{n-1}} \right)^k + \left( \frac{1}{2^n} \right)^k \right)$.
Theorem

Stages 1. and 2. compute a 3–colouring for any ring graph of size $n$ in $3 \log n$ rounds on average and w.h.p.
Computation of a Graph Partition - Stage 1.
Computing a Graph Partition - Stage 1.
Computation of a Graph Partition - Stage 1.
Computation of a Graph Partition - Stage 1.

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari

Colouring and Partition
Computation of a Graph Partition - Stage 1.
Computation of a Graph Partition - Stage 2.
Computation of a Graph Partition - Stage 2.
Computation of a Graph Partition - Stage 2.

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari

Colouring and Partition
Computation of a Graph Partition - Stage 2.

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari

Colouring and Partition
Computation of a Graph Partition - Stage 2.

Y. Métivier, J.M. Robson, N. Saheb, A. Zemmari

Colouring and Partition
Theorem

Let $G = (V, E)$ be a graph of size $n \geq 1$. Algorithm SF constructs a spanning forest of $G$ in time $O(\log n)$ on average and w.h.p., the size of each message is equal to 1.
Radius of Trees

For any graph of size $n$ and maximum degree $\Delta$, trees of the spanning forest computed by Algorithm $SF$ have a radius upper bounded by $e\Delta + 2\log n$ with probability $1 - o\left(n^{-1}\right)$. 
The Number of Trees.

\[ E(X_n) = \sum_{v \in V} \frac{1}{d(v) + 1}. \]
Particular case: The Ring Graph

Observation: The set of numbers generated by Stage 1 can be interpreted as a permutation.

With $\psi_n(x)$ the generating function of the number $X_n$ of local minima in a ring of size $n \geq 3$, we derive:

$$n \psi_{n+1}(x) = nx \psi_n(x) - 2x(x - 1) \psi'_n(x),$$

with the initial value $\psi_1(x) = x$.

$\Rightarrow$ The expected number of local minima is equal to $\frac{n}{3}$ for $n \geq 3$ and its variance is $\frac{2n}{45}$ for $n \geq 4$. 
Particular case: The Ring Graph

\[ \Pr (X_n = m) = \]

\[ \frac{2^{n+2}}{n!} \sum_{i=1}^{m} (-1)^{m-i} 2^{-2i} \left( \frac{n+1}{m-i} \right) \sum_{j=1}^{i} \frac{j^{n+1}}{(i-j)!(i+j)!(2i-1)!} \]
Perspectives

Open Question:
Are the two algorithms optimal (from the point of view of the bit complexity) even for graphs with unbounded degrees?