Complexity analysis of a decentralised graph colouring algorithm

K. R. Duffy[(1)], N. O'Connell[(2)] and A. Sapozhnikov[(3)]

August 2006; revised December 2007

(1) Hamilton Institute, National University of Ireland, Maynooth, Ireland.
(2) Mathematics Institute, University of Warwick, Coventry, CV4 7AL, UK.
(3) Centrum voor Wiskunde en Informatica, Kruislaan 413, 1098SJ Amsterdam, Holland.

Abstract

Colouring a graph with its chromatic number of colours is known to be NP-hard. Identifying an algorithm in which descisions are made locally with no information about the graph's global structure is particuarly challenging. In this article we analyse the complexity of a decentralised colouring algorithm that has recently been proposed for channel selection in wireless computer networks.

Keywords: computational complexity; graph algorithms; randomized algorithms.

1 Introduction

[1] Consider an undirected connected graph G with vertices $V = \{1, \ldots, N\}$, where $N \geq 2$. For $i \neq j \in V$ let $i \leftrightarrow j$ denote the existance of an edge joining i and j. A proper vertex colouring of a graph G using C colours is a map $f : V \mapsto \{1, \ldots, C\}$ such that $f(i) \neq f(j)$ if $i \leftrightarrow j$. The smallest number C for which such an f exists is called the chromatic number of the graph G and denoted χ or $\chi(G)$.

It is well known that properly colouring a graph with χ colours is NP-hard for $\chi \geq 3$ [4]. Restricting to certain classes of random graphs, polynomial time algorithms exist that properly colour them with high probability as the number of vertices diverges, e.g [5, 10, 2, 1]. As in many practical applications graph structure is not well represented by a random graph, algorithms have been proposed for colouring arbitary graphs. For example, Feder and Motwani [3] propose a randomized algorithm with exponential running time and linear space. They also provide a robust defence of the importance of investigating exponential running time algorithms.

[1]NOTICE: this is the author's version of a work that was accepted for publication in Information Processing Letters. Changes resulting from the publishing process, such as peer review, editing, corrections, structural formatting, and other quality control mechanisms may not be reflected in this document. Changes may have been made to this work since it was submitted for publication. A definitive version was subsequently published in INFORMATION PROCESSING LETTERS, doi:10.1016j.opl.2008.01.002.

All of the algorithms mentioned above, and brute-force searching, are centralised techniques, which are appropriate for many traditional applications including register allocation in compilers. That is, if we assume that there is intelligence at each vertex, then in order for this intelligence to run these algorithms it is necessary that at least one vertex is in possession of complete knowledge of the graph's structure. To use a colouring that has been found, this information must be shared with the rest of the vertices and they must accept the colour they are given. For practical purposes this means message passing must gather graph data for a controller that has been accepted to make the colouring decision. Once a colouring was found, the controller must then message pass the solution back to the other vertices. Complexity of distributed graph colouring algorithms is investigated in [8, 6].

Clearly it is significantly more difficult to colour in a decentralised fashion where no element in the graph has knowledge of the graph's structure and only limited information regarding its neighbours. Yet in certain applications centralised solutions are inapplicable and a decentralised algorithm must be used. For example, this is the case of channel allocation in wireless computer networks that use the IEEE 802.11 standard, which are prevasive. Here the adjacency relation in the graph is determined by interference. Dependent on the physical layer, there are a given number of non-overlapping radio frequencies (colours in the interference graph): 802.11b/g has 3 and 802.11a has 12. In a wireless network employing IEEE 802.11, a decentralised algorithm is necessary as: (1) the distance to which interference extends is significantly greater than the distance at which it is possible to decode messages, so that stations may interfere without being able to communicate; (2) the elements of the network may be owned by distinct entities (companies, individuals, etc.) that are unwilling to allow them to communicate, even if they are within range or are connected through a wired back-haul. A consequence of this is that no part of the protocol allows one access point to dictate the frequency selection of another access point.

Leith and Clifford [7, 9] have recently proposed a fully decentralised stochastic colouring algorithm, inspired by frequency allocation within the constraints of wireless networks employing the IEEE 802.11 standard. They call it the Communication-Free Learning (CFL) algorithm and have shown emperically that it has desireable properties. They also outline a proof that the algorithm converges almost surely to a proper colouring, if one exists. Inspection of their approach gives a upper bound on the running time to achive a high probability of convergence that is of order $\exp(N^2\delta_1)$, for some $\delta_1 > 0$. In this article we further analyse the complexity of the CFL algorithm. In particular, Corollary 5 proves that an upper bound on the running time to achive a high probability of convergence that is $\exp(N\delta_2)$, for a given $\delta_2 > 0$.

2 The algorithm

The CFL algorithm is parameterized by $\beta \in (0, 1)$ and works as follows. Time $t \in \{0, 1, \ldots\}$ is discrete and at each instant every node $n \in 1, \ldots, N$ in the graph selects its colour $c_n(t)$ according to its own probability distribution $p_n(t)$ on the range of available colours $\{1, \ldots, C\}$. At $t = 0$, each node's colour distribution is initialized to be the uniform distribution: $p_n(0) = (1/C, \ldots, 1/C)$ for all $n \in \{1, \ldots, N\}$. The distributions $p_n(t)$ then evolve for $t \geq 1$ according to the following rule

$$p_n(t+1) = \begin{cases} \delta_{c_n(t)} & \text{if } c_n(t) \neq c_i(t) \text{ for all neighbors } i \leftrightarrow n \\ (1-\beta)p_n(t) + \dfrac{\beta}{C-1}\bar{\delta}_{c_n(t)} & \text{if } c_n(t) = c_i(t) \text{ for any neighbor } i \leftrightarrow n \end{cases} \tag{1}$$

where δ_i denotes the distribution with unit mass on $i \in \{1, \ldots, C\}$ and $\bar{\delta}_i$ denotes the vector in $\mathbb{R}^C$ which has component 0 at i and 1 at all other positions. Let $X(t) = (p_1(t), \ldots, p_N(t)) \in [0,1]^{NC}$. By construction the sequence $\{X(t), t \in \{0, 1, \ldots\}\}$ forms a Markov chain.

Note that the algorithm only assumes that if you select a colour that one of your neighbours has chosen, you are aware of it. The effect of the update rule in equation (1) is to cause a node to stick with a colour if none of its neighbors have chosen the same colour. If one of its neighbors chooses the same colour, it moves probability off the previously selected colour and distributes it to all other colours. If β is small, there is a stickiness, with nodes unlikely to leave a colour once it has proved to be a local solution. If β is large, less resistance is present to global changes, as the effects of collisions (neighbors choosing the same colour) propagate with greater likelihood.

3 Main results

We will say two vertices $i \leftrightarrow j$ collide if they both select the same colour. We define the absorbed state $A \subset [0,1]^{CN}$ to be the collection of nodes' probability distributions corresponding to acceptable colourings:

$$A = \{(\delta_{c(1)}, \ldots, \delta_{c(N)}) : \text{where } c(i) \in \{1, \ldots, C\} \text{ for all } i \in \{1, \ldots, N\} \text{ and } c(i) \neq c(j) \text{ if } i \leftrightarrow j\}.$$

Let $|A|$ denote the cardinality of this set. We say that the CFL algorithm has converged, or is in an absorbing state, at time t if $X(t) \in A$. Once the Markov chain $\{X(t)\}$ enters a state in A, the probability it leaves its current state is zero. Let τ denote the stopping time $\tau = \inf\{t : X(t) \in A\}$. Theorem 3 below will show by construction that there exists $\gamma \in (0,1)$ such that, for any t and any valid state $\vec{x} \in [0,1]^{NC}$, $P(\tau \leq t + N | X(t) = \vec{x}) \geq \gamma^N$.

We first prove the following lemma, whose motivation can be found by reading the statement of its corollary, before moving on to the main theorem, Theorem 3.

Lemma 1. *Define the collection of states*

$$\begin{aligned} U \quad = \quad & \{(p_1, \ldots, p_N) : \textit{for all } i \in \{1, \ldots, N\} \textit{ either } p_i = \delta_j \textit{ for some } j \in \{1, \ldots, C\} \\ & \textit{or } (p_i)_k \geq \beta(1-\beta)/(C-1) \textit{ for all } k \in \{1, \ldots, C\}\}. \end{aligned}$$

If $C \geq \chi$, for any valid state $\vec{x} \in [0,1]^{NC}$ at time t,

$$P(X(t+1) \in U \mid X(t) = \vec{x}) \geq \left(\frac{\beta}{C-1}\right)^N. \tag{2}$$

Proof. If there does not exist $i \leftrightarrow j$ such that $c_i(t) = c_j(t)$, then $\vec{x} \in A \subset U$ and $X(t+1) = \vec{x} \in U$. If there exists one or more pairs $i \leftrightarrow j$ such that $c_i(t) = c_j(t)$, then define the following set of vertices for each t' and each $c \in \{1, \ldots, C\}$:

$$V(t', c) = \{i \in \{1, \ldots, N\} : c_i(t') = c_j(t') = c, \text{ some } j \leftrightarrow i\}.$$

Thus the set of all colliding vertices at time t' is

$$\bigcup_{c=1}^{C} V(t', c).$$

Note that every $v \in V(t', c)$ has $(p_v(t'))_k \geq \beta/(C-1)$ for all $k \neq c$. Consider the probability of the following event: for each $c \in \{1, \ldots, C\}$, every vertex in $V(t, c)$ selects the same $c' \neq c$. The probability this happens is lower bounded by $(\beta/(C-1))^N$ and then every

$$v \in \bigcup_{c=1}^{C} V(t+1, c)$$

has $(p_v(t+1))_k \geq (1-\beta)\beta/(C-1)$ for every $k \in \{1, \ldots, C\}$. Thus the bound (2) follows. □

An immediate corollary to the proceeding Lemma is the following:

Corollary 2. *For any initial state at time t, the probability that at time $t+2$ that all colliding vertices are doing so* on the same colour *is lower bounded by*

$$(1-\beta)^N \left(\frac{\beta}{C-1}\right)^{2N}. \tag{3}$$

We are now in a position to prove the lower bound.

Theorem 3. *If $C \geq \chi$, for any valid state $\vec{x} \in [0,1]^{NC}$ at time t,*

$$P(\tau \leq t+N \mid X(t) = \vec{x}) \geq |A| \left((1-\beta)^2 \left(\frac{\beta}{C-1}\right)^{2C+1}\right)^N =: |A|\gamma^N.$$

Proof. By corollary 2, uniformly over the initial states, a lower bound on the probability that in 2 time steps all colliding vertices are colliding on the same colour is given by (3). Thus we initially assume that all colliding vertices at time t are colliding on the same colour.

We shall lower bound the probability the algorithm follows a particular sequence of events that leads to a correctly coloured graph. Starting with the vertices that originally collided, we shall cycle them through their neighbours' colours, growing the set until it encompasses the whole graph. Once a vertex is sufficiently deep within the set, its colour can be fixed on one corresponding to a proper colouring. We shall demonstrate this can occur in $N-2$ steps and lower bound its probability.

For a set V of vertices we define its outer-boundary, ∂V, by

$$\partial V = \{i \in G \setminus V : \exists j \in V \text{ such that } i \leftrightarrow j\}.$$

We define the nested sets $\{G_s : s \geq t\}$ by the following procedure. Let G_t be the set of colliding vertices at time t. If $\partial G_t \neq \emptyset$, let $c = \min\{c_v(t) : v \in \partial G_t\}$ and define

$$G_{t+1} = G_t \cup \{v \in \partial G_t : c_v(t) = c\}.$$

If $\partial G_t \setminus G_{t+1} \neq \emptyset$, let $c = \min\{c_v(t) : v \in \partial G_t \setminus G_{t+1}\}$ and define

$$G_{t+2} = G_{t+1} \cup \{v \in \partial G_t \setminus G_{t+1} : c_v(t) = c\}.$$

We repeat this procedure until the first k such that $\partial G_t \setminus G_{t+k} = \emptyset$, which, as there are C colours, takes no more than $C-1$ steps. This is the first time at which the boundary ∂G_t is absorbed by

some G_{t+k}. If $G_{t+k} = G$, the procedure stops and we set $G_{t+k+s} = G$ for all $s \geq 0$. Otherwise, let $c = \min\{c_v(t) : v \in \partial G_{t+1} \setminus G_{t+k}\}$ and define

$$G_{t+k+1} = G_{t+k} \cup \{v \in \partial G_{t+1} \setminus G_{t+k} : c_v(t) = c\}$$

repeating the procedure as before to build the entire sequence $\{G_s, s \geq t\}$. Note that: (i) $\{G_s, s \geq t\}$ is a only a function of the graph G and configuration of colours at time t; and (ii) as we start with at least two nodes colliding, the graph G is included in a G_{t+k} in at most $k = N - 2$ steps, so that $G_{t+N-2} = G$.

For $s \geq t$, if $G_s = G$ we define the set $\widetilde{G}_s$ to be G. Otherwise $\widetilde{G}_s$ is defined to be the maximal subset of vertices of G_s such that the procedure described above for constructing $\{G_s, s \geq t\}$ when applied to graph $G \setminus \widetilde{G}_s$ with colliding vertices $G_s \setminus \widetilde{G}_s$ and all other vertices coloured according to the colouring of $G \setminus G_s$ at time t produces nested sets $\{\widehat{G}_k\}_{k \geq s}$ such that, for any $k \geq s$,

$$\widehat{G}_k = G_k \setminus \widetilde{G}_s.$$

Finally, we define the sequence $\{\mathrm{int}(\widetilde{\mathcal{G}}_s), s \geq t\}$ by: $\mathrm{int}(\widetilde{\mathcal{G}}_s) \subset \widetilde{\mathcal{G}}_s$ is the interior of $\widetilde{\mathcal{G}}_s$ if all neighbours of vertices of elements of $\mathrm{int}(\widetilde{\mathcal{G}}_s)$ are in $\widetilde{\mathcal{G}}_s$. Note that if $v \in \widetilde{\mathcal{G}}_s$, then $v \in \mathrm{int}(\widetilde{\mathcal{G}}_{s+C-1})$.

The idea is that the sets $\{G_s, s \geq t\}$ "grow" to encompass the whole graph, leaving a correctly coloured graph in their wake. The sets $\{\widetilde{G}_s, s \geq t\}$ are within the interior of this growth and the elements of $\{\mathrm{int}(\widetilde{\mathcal{G}}_s), s \geq t\}$ are deep within the interior.

Each vertex v experiences no more than $2C - 1$ collisions after t until it is in $\widetilde{\mathcal{G}}_s$, for some s. The probability that it selects the correct sequence of colours in that period is bounded below by

$$\left(\frac{\beta}{C-1}\right)^{2C-1}.$$

It then selects its "correct" final colour and stays on it. Should its final colour be the same as the colour that it last collided on, the probability this happens is lower bounded by $\beta(1-\beta)/(C-1)$. Should its final colour not be the same as the colour that it last collided on, this happens with probability lower bounded by $\beta/(C-1)$, but it may experience one more collision before becoming part of the interior. Should this happen, the likelihood of staying on the same colour after the collision is $(1-\beta)$. Thus, regardless of what happens, the likelihood that the vertex selects its correct colour and stays on it is lower bounded by $\beta(1-\beta)/(C-1)$.

As there N vertices and we start with at least two colliding, in $N-2$ steps the probability we have a correct colouring is lower bounded by

$$\left(\frac{\beta}{C-1}\right)^{2CN}(1-\beta)^N.$$

Including the lower bound given in equation (3), that from any configuration we can get to the stage where all colliding vertices are doing so on the same colour in 2 steps, gives the result □

Corollary 4. *The probability that the graph is not properly coloured by time m is bounded above by*

$$(1 - |A|\gamma^N)^{\lceil m/N \rceil}.$$

Corollary 5. *With high probability the algorithm colours in exponential time. That is, for any given graph and $\epsilon \in (0,1)$ the number of steps m for which the graph must be run to ensure that a colouring is obtained with likelihood $1-\epsilon$ is of order less than*

$$N \exp(N \log(\gamma^{-1})) \log(\epsilon^{-1}),$$

where N is the number of vertices of the graph.

We can find the tightest bound by identifying the β that maximizes $\gamma(\beta)$: $\beta^* = (2C+1)/(2C+2)$ and

$$\log(\gamma^{-1}(\beta^*)) = 2\log(2C+2) + (2C+1)\log\left(\frac{(C-1)(2C+2)}{2C+1}\right).$$

This does not suggest the best β for the algorithm, merely for our bound. Note that $\gamma(\beta^*)$ grows as $C\log(C)$, in comparison to bounds of $\log(C)$ for centralized algorithms such as in [3].

Acknowledgments: K. D. is supported by Science Foundation Ireland grants 03-IN3-I396 and RFP-07-ENEF530. N. O'C. and A. S. were with University College Cork, Ireland, supported by grant SFI-04-RP1-I512 when this work was performed.

References

[1] Noga Alon and Nabil Kahale. A spectral technique for coloring random 3-colorable graphs. *SIAM J. Comput.*, 26(6):1733–1748, 1997.

[2] M. E. Dyer and A. M. Frieze. The solution of some random NP-hard problems in polynomial expected time. *J. Algorithms*, 10(4):451–489, 1989.

[3] Tomás Feder and Rajeev Motwani. Worst-case time bounds for coloring and satisfiability problems. *J. Algorithms*, 45(2):192–201, 2002.

[4] Michael R. Garey and David S. Johnson. *Computers and intractability.* W. H. Freeman and Co., San Francisco, Calif., 1979.

[5] L. Kučera. Expected behavior of graph coloring algorithms. In *Fundamentals of computation theory (Proc. Internat. Conf., Poznań-Kórnik, 1977)*, pages 447–451. Lecture Notes in Comput. Sci., Vol. 56. Springer, Berlin, 1977.

[6] F. Kuhn and R. Wattenhofer. On the complexity of distributed graph coloring. In *Proc. of the 25th ACM PODC*, 2006.

[7] D. J. Leith and P. Clifford. Convergence of distributed learning algorithms for optimal wireless channel allocation. In *IEEE CDC*, pages 2980–2985, 2006.

[8] N. Linial. Locality in distributed graph algorithms. *SIAM J. Comput.*, 21(1):193–201, 1992.

[9] D. Malone, P. Clifford, D. Reid, and D.Leith. Experimental implementation of optimal WLAN channel selection without communication. In *IEEE DySPAN*, pages 316–319, 2007.

[10] Jonathan S. Turner. Almost all k-colorable graphs are easy to color. *J. Algorithms*, 9(1):63–82, 1988.