Self-organizing networks: Optimizing channel allocation with local knowledge only

Matthew Wiltshire[*] Keith Briggs[†] Antal A. Járai[‡]

Last modified 2014-09-25 13:50

1 Abstract

In a network of wireless base stations, there are a certain number of frequency channels on which each base station can run. There needs to be a mechanism in place to decide which of these channels each base station will run on. One way of achieving this is for each base station to have an algorithm which allows it to choose its channel itself. Ideally base stations which are close enough together for there to be interference between them should choose different channels in order to reduce this interference. In this project we represent this situation as a graph coloring problem, and we consider several different algorithms for choosing channels, assessing the optimality of each one.

2 Introduction

The problem of reducing interference in a wireless network is crucial to ensure the best possible user experience. Since there are only a limited number of channels available, it may not be possible to avoid all interference, but we wish to find the best way of assigning channels so that the interference is minimal. One method of channel assignment is the so-called self-organizing network, where each base station chooses its channel itself. This has the advantage that the network will run itself without needing someone watching the whole network and choosing the channels manually. These sorts of networks use distributed algorithms, a topic discussed further in Barbosa (1996) and Tel (2000).

In this report we will present four different algorithms for channel assignment along with an analysis of each one, assessing their suitability for use in wireless networks. We will also outline the graph coloring model and the methods used to generate these results.

[*]Department of Mathematical Sciences, University of Bath, UK. mdw27@bath.ac.uk

[†]BT Technology, Services and Operations (TSO), Adastral Park, Martlesham Heath, UK. keith.briggs@bt.com

[‡]Department of Mathematical Sciences, University of Bath, UK. A.Jarai@bath.ac.uk

3 The graph coloring model

Consider a network of wireless base stations. We will represent this network as a graph $G=(V,E)$, a set of nodes V where certain pairs of nodes are connected by edges $e \in E$. The nodes represent the base stations, and if two base stations are close enough together to cause interference, then their nodes are connected by an edge. Each node has a color which represents the frequency channel on which it is currently running. If two nodes connected by an edge have the same color, then there will be interference between them.

We have considered two different types of coloring model in this project, the first of which is the tight coloring model. In this model, for each node $v \in V$, the set of colors available to that node is $\{1, \ldots, \deg(v)+1\}$. In the second type of model, the non-tight coloring model, a global maximum color k is specified for the whole network, so that each node has the colors $\{1, \ldots, k\}$ available to choose from. This type of model is more akin to real-world wireless networks where there are a specific number of frequency channels from which all base stations can choose. Furthermore, we have also considered adding a 0 color in some cases, which represents a base station being turned off.

In this project we have imposed certain constraints on the model. Firstly, the network must be self-organizing, so that each node chooses its own color. When choosing its color, the only knowledge that a node has is the colors of its neighbours; none of the nodes have any global knowledge about the rest of the network. Moreover, nodes do not share any information about their neighbours' colors with each other. The system is updated as follows: at each step, a node is chosen uniformly at random. This node then looks at its neighbours' current colors and updates its color according to the specified algorithm, whilst the other nodes retain their current colors. This updating process continues indefinitely; the process does not stop once a certain coloring has been reached.

4 Methodology

We represented the model as a Markov chain, in which the set of states of the chain is the set of possible colorings of the graphs. A coloring c of the graph is an n-tuple where n is the number of nodes in the graph, in which the i^{th} element of c is the color of node i. To generate our results, we used the stationary distribution of the chain. This had the advantage of giving us exact solutions, but it limited us to looking at smaller problems since it required greater computing power than if we had looked at simulations of the chain instead. Hence the networks that we considered would most likely represent a tight cluster inside a larger network, such as a group of terraced houses within a larger settlement.

In order to compute the stationary distribution, we first needed to generate all of the states of the Markov chain, for which we had two methods. In the first method, we started with an empty coloring and built up the coloring node-by-node. As soon as a partial coloring became

invalid, we discarded all colorings beginning with those values. This method always generates all valid colorings and is efficient since it avoids having to generate every possible coloring and then check their validity. In the second method, an initial state was chosen and we recursively followed the transition rules of the Markov chain to find every coloring that one can move to. This method however only generates all valid colorings if the Markov chain is connected, which is not always the case in our situation.

After generating all the states, the stationary distribution was computed as follows. Let $P_{i,j}=\mathbb{P}$(move from state i to state j). We want to solve:

$$\begin{aligned} & z^T P = z^T \\ \Leftrightarrow\ & z^T(P-I) = 0^T \\ \Leftrightarrow\ & (P^T - I)z = 0 \end{aligned}$$

To make the solution unique and normalized, we replaced the first row of the matrix $P^T - I$ by a row of 1's and the first element of the zero vector by a 1. We then passed it to a sparse matrix solver which outputted the stationary distribution z.

We used two types of random graph to generate our results. The simplest type is the Erdős-Rényi random graph, in which n nodes are created and each pair of nodes is connected by an edge with some specified probability p. The second type that we used are geometric random graphs. Here n nodes are placed at random inside a unit square, and circles of some specified radius r are created centred on each node. If two nodes' circles overlap, then they are connected by an edge. These graphs are most like the real-world networks we are considering since the nodes have a physical location and are connected if they are close enough to each other.

Our aim in this work was to try to find the optimal algorithm for the nodes to use to update their color. In order for an algorithm to be considered optimal, the Markov chain needs to spend as much time as possible in proper colorings, i.e., colorings in which each node is colored differently to all of its neighbours. Proper colorings represent the system being in a state with no interference, which is desirable. However we also wish to use as few colors as possible. This is because in a real-world network there will be a limit on the number of channels available. To assess this, we compared the average number of colors used with the chromatic number χ of the graph, which is the smallest number of colors that can be used to obtain a proper coloring.

5 Results

We now present four different algorithms for coloring the nodes, analyzing each of them in turn.

5.1 Variant (a)

At each time step:

- Choose a node v uniformly at random to update
- Create a set $\{1, \ldots, \deg(v)+1\}$ and remove any colors that a neighbour of v is using
- With probability α, choose the smallest color from the above set of valid colors
- Otherwise, choose a color uniformly from the set of valid colors

This variant is a tight coloring variant. Bullimore and Briggs (2011) make use of the tight coloring theorem, conjectured by Keith Briggs and proven by Colin McDiarmid, which is repeated below for reference:

Theorem: Call a proper graph coloring tight if each node i is colored from the set $\{0, 1, \ldots, \delta(i)\}$, where $\delta(i)$ is its degree. Every graph has at least one tight χ-coloring, where χ is the chromatic number.

This theorem tells us that the Markov chain will always be in a proper coloring under this variant. Thus we analyze this variant by looking at the number of colors used. Figure 1 shows this for 250 realizations of a 10-node Erdős-Rényi graph, whilst Figure 2 instead uses 100 realizations of a 10-node geometric random graph.

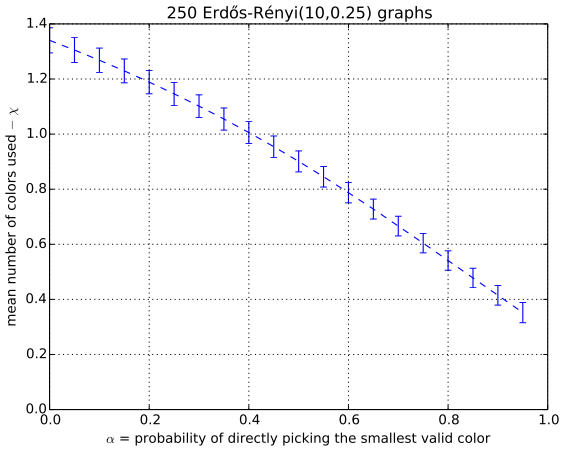
In both cases the number of colors used gets closer to the chromatic number as the parameter α increases, suggesting that going directly to the smallest valid color is beneficial. Since there are never any color clashes in the network, we can conclude that this variant with a large value of α could potentially be a good algorithm to use.

5.2 Variant (b)

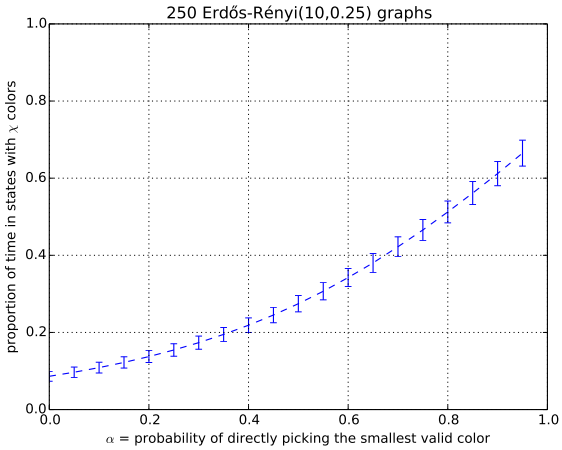
At each time step:

- Choose a node v uniformly at random to update
- Create a set $\{1, \ldots, \deg(v)+1\}$ and remove any colors that a neighbour of v is using
- With probability α, choose the largest color from the above set of valid colors
- Otherwise, choose a color uniformly from the set of valid colors

This is the same algorithm as in Variant (a), except that the largest color is chosen with probability α instead of the smallest. Again this is a tight coloring variant so we only need to consider the number of colors used. Figure 3 shows this for 250 realizations of a 10-node Erdős-Rényi graph, and Figure 4 uses 100 realizations of a 10-node geometric random graph.

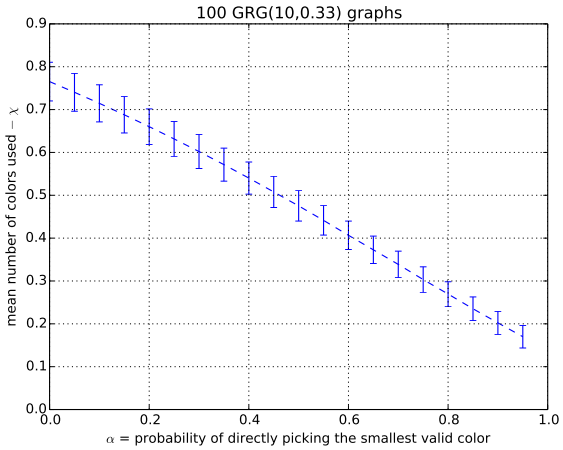


(a) Mean number of colors used - χ

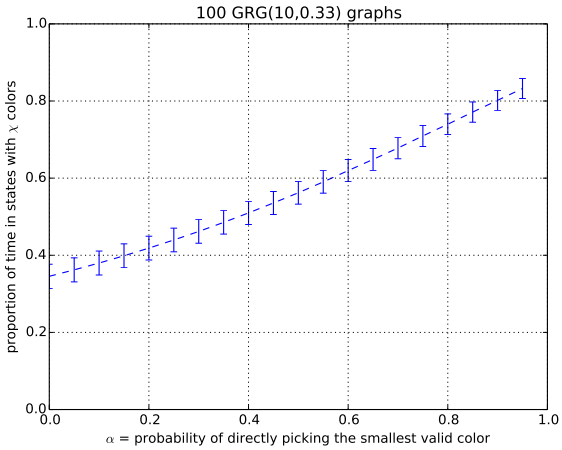


(b) Proportion of time in states with χ colors

Figure 1: Mean and standard error for 250 Erdős-Rényi(10,0.25) graphs under Variant (a)

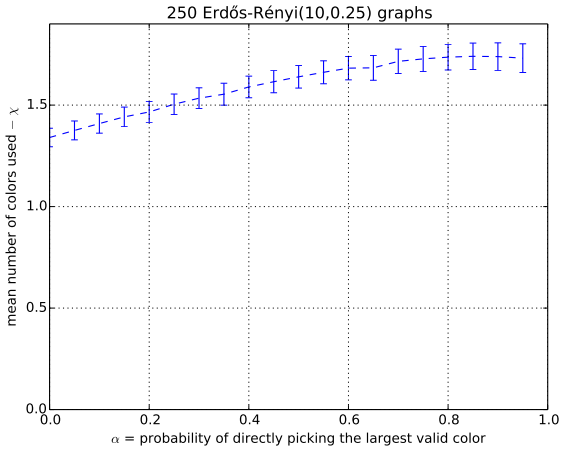


(a) Mean number of colors used - χ

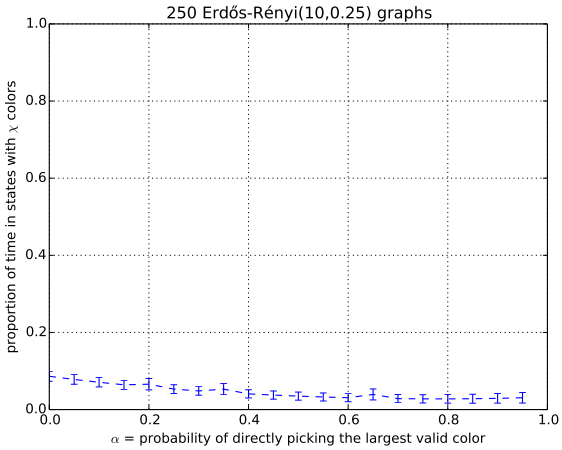


(b) Proportion of time in states with χ colors

Figure 2: Mean and standard error for 100 GRG(10,0.33) graphs under Variant (a)

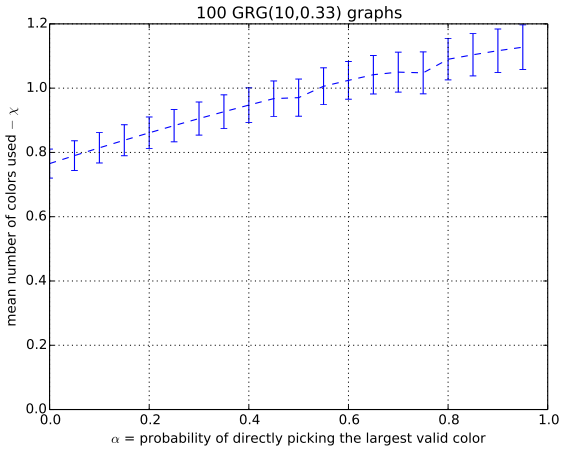


(a) Mean number of colors used - χ

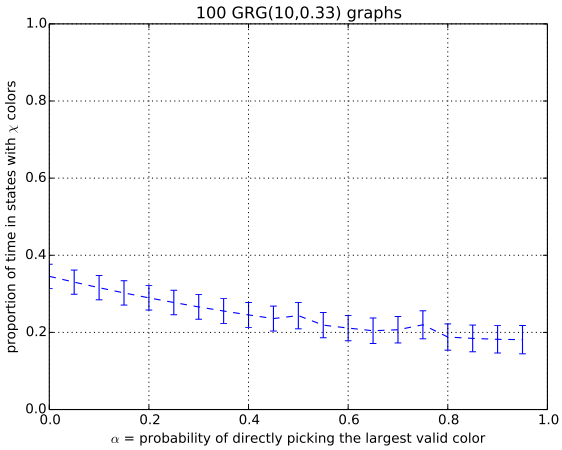


(b) Proportion of time in states with χ colors

Figure 3: Mean and standard error for 250 Erdős-Rényi(10,0.25) graphs under Variant (b)

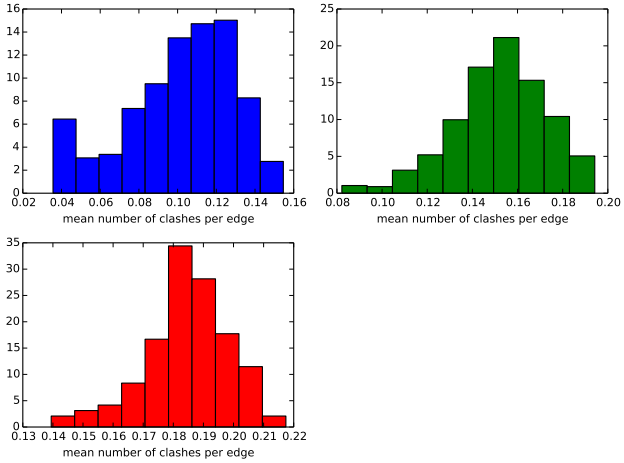


(a) Mean number of colors used - χ

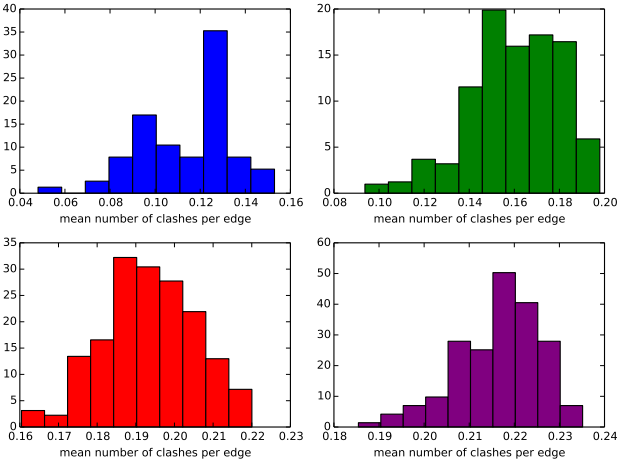


(b) Proportion of time in states with χ colors

Figure 4: Mean and standard error for 100 GRG(10,0.33) graphs under Variant (b)



(a) 1000 Erdős-Rényi(8,0.57) graphs.
From top left to bottom left, χ=3, 4, 5



(b) 1000 GRG(8,0.57) graphs.
From top left to bottom right, χ=3, 4, 5, 6

Figure 5: Mean number of clashes per edge under Variant (c) with maximum color 3

It is apparent that the results get worse as α increases, since the mean number of colors moves further away from the chromatic number and the chain spends less time with χ colors. Thus directly choosing the largest valid color with any probability does not do any better than simply choosing a valid color uniformly at random, and so this variant would not be the best one to use regardless of the value of α chosen.

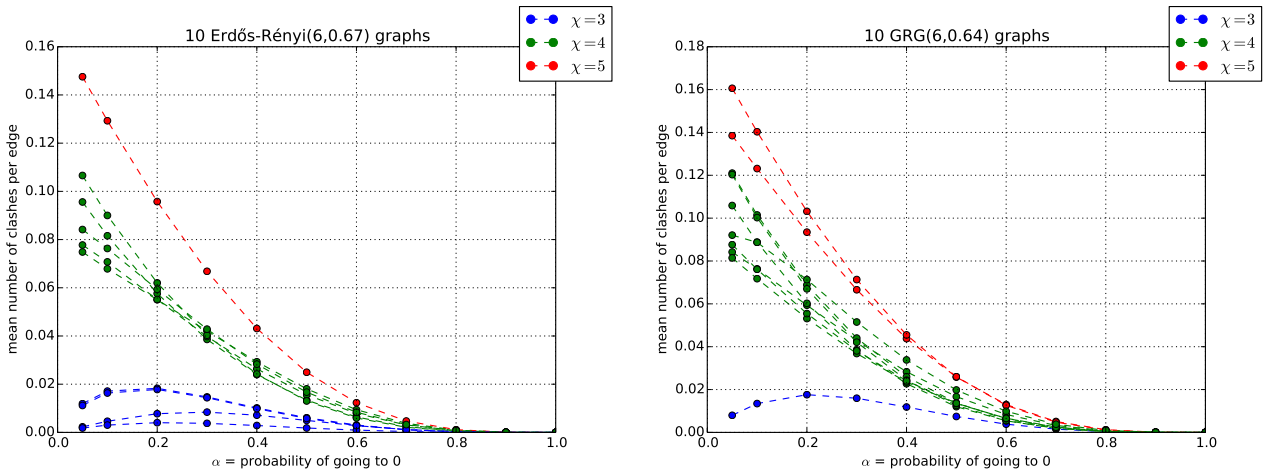
5.3 Variant (c)

We now fix a maximum color k for the whole graph. At each time step:

- Choose a node v uniformly at random to update
- Look at the colors one higher and one lower than its current color
- Choose whichever of the two is least used by its neighbours (or choose uniformly from the two if they are equally used)
- If already at the maximum color, then instead of considering one higher it considers its current color. Similarly if at the minimum color, it considers its current color instead of one lower

This is not a tight coloring variant, so there is the possibility of clashes occurring. The maximum color k can be fixed to represent the number of frequency channels available. Hence we analyze this variant by considering the number of clashes in the network, as shown in Figure 5 for 1000 8-node Erdős-Rényi graphs and 1000 8-node geometric random graphs.

As is to be expected, in graphs with a higher chromatic number, each edge causes a clash more often. However even when the chromatic number is equal to the maximum color, each



(a) 10 Erdős-Rényi(6,0.67) graphs

(b) 10 GRG(6,0.64) graphs

Figure 6: Mean number of clashes per edge under Variant (d) with maximum color 3

edge still causes a clash roughly one tenth of the time. Hence it may be possible to find a better variant than this, since when χ=3 there are proper colorings available but the chain is spending some time away from them, meaning there is interference in the network.

5.4 Variant (d)

Again a maximum color k is fixed for the whole graph. At each time step:

- Choose a node v uniformly at random to update
- With probability α, choose color 0
- Otherwise, choose the color least used by its neighbours from the range $\{1, \ldots, k\}$ (if there is more than one such color, then choose uniformly from them)

This variant has the possibility of clashes, but it also introduces the color 0 which represents the base station being turned off. We looked at how the number of clashes in the network is affected by the parameter α, and this is shown for 10 6-node Erdős-Renyi graphs and 10 6-node geometric random graphs in Figure 6.

The plots show that by increasing the probability of choosing color 0, the number of clashes in the network is reduced (except when χ=3 and α is small). However, in practice it would not be feasible to require the base stations to have a large probability of switching off. Hence there is a playoff between reducing interference and allowing a large proportion of the base stations to be turned on.

6 Future ideas

In this project we have considered many different algorithms for channel assignment in a wireless network. Although some of these algorithms show promising results, it is not clear whether they can be improved on or not. We wish to find an algorithm which reduces the amount of interference and at the same time uses as few colors as possible, and based on our results it does not appear that we have found the optimal algorithm yet.

There are a large number of different ways that colors can be assigned to the nodes of a graph under our constraints, and we have only really scratched the surface here with the algorithms we have considered. Future work in this area could look at a much wider range of algorithms in order to find the optimal one to use. One particular algorithm we would like to look into further is given in Section 7 below.

Another factor which could be investigated is the mixing time of the Markov chains used (Levin, Peres, and Wilmer 2009). Although we have generated our results using the stationary distribution in each case, we have not taken into account how long it takes for each chain to reach this distribution. It would be useful to consider this when analyzing each variant, and to be able to optimize both the quality of the stationary distribution and the time taken to reach it.

7 A proposed new variant

1. First run the Markov chain with transition probabilities proportional to $\exp[-\beta \sum_{v,w:v\sim w} b(\eta(v)-\eta(w))]$, where $b(0)>b(1)>\cdots>b(15)$ are constants calibrated to account for how much we care about the frequency overlap between $\eta(v)$ and $\eta(w)$, the colors of v and w.

   Run this for a fixed time, e.g., 10 times the mixing time, then stop.

2. Select a node v at random, and check if $\eta(v)$ can be changed so that $\sum_{w:w\sim v} b(\eta(w)-\eta(v))$ is reduced. If so, change $\eta(v)$ to that value, otherwise leave it unchanged.

   Run this for a sufficient time so that a local minimum has been found, or has been found with high probability.

3. Keep that coloring for some specified time, e.g., until the graph changes, when we might repeat it, perhaps on a local neighbourhood of where the change has occurred.

This two-step strategy appears to have certain desirable properties. It can be considered 'fair' in the sense that no user is favoured. The quality of the coloring only improves in step 2, so the result from step 1 can be used as a rigorous upper bound on performance even if step 2 is difficult to analyze. Moreover, if a local minimum is found in step 2, then no user can improve

their connection by switching channel. In the future we would like to analyze how this model performs to assess if it is indeed a good algorithm to use.

8 Mixing time proofs

In addition to the work presented above, we also looked at the mixing time of a different proposed coloring algorithm. In this section we present proofs of a lower bound and an upper bound on the mixing time of this variant. The dynamics of the variant are as follows:

- Choose a node v uniformly at random to update
- Set v to be white with probability p, and non-white with probability $1-p$
- If v is set to be non-white, select its color c with probability proportional to the weight $\exp(-\beta N(v,c))$, where $N(v,c)$ is the number of neighbours of v with color c

8.1 Lower bound on the mixing time

Levin, Peres, and Wilmer (2009) proved that for Glauber dynamics on the set of proper colorings of the empty graph, a lower bound for the mixing time is $\frac{1}{2}n\log n - c(q)n$ for a constant $c(q)$ (solution to Exercise 7.3, pp. 335-336). I will adapt this proof to show that for the above variant, a lower bound for the mixing time is $c(p)n\log n$ for a constant $c(p)$.

Let $\{v_1,\ldots,v_n\}$ be the node set of the graph, and let(X_t) be the Markov chain started at initial coloring $\mathbf{0}$, where every node is colored white.
Let

$$N=\sum_{i=1}^{n}\mathbb{I}_{\{x(v_i)=0\}}=\text{number of white nodes in coloring } x$$

First consider N under the measure $P^t(\mathbf{0},\cdot)$:

Let $X_i(t)=\mathbb{I}_{\{x_t(v_i)=0\}}$. Then $X_i(t)=0$ if and only if v_i has been updated at least once in the first t time steps and at the latest update it is colored non-white.

This occurs with probability $\left[1-\left(1-\frac{1}{n}\right)^t\right](1-p)$.

So

$$\begin{aligned}\mathbb{E}_{\mathbf{0}}(X_i(t))&=1-\left[1-\left(1-\frac{1}{n}\right)^t\right](1-p)\\&=1-(1-p)+(1-p)\left(1-\frac{1}{n}\right)^t\\&=p+(1-p)\left(1-\frac{1}{n}\right)^t\end{aligned}$$

Hence $\mathbb{E}_{\mathbf{0}}(N(t))=np+n(1-p)\left(1-\frac{1}{n}\right)^t$.

We next prove that $\mathrm{Var}_{\mathbf{0}}(N_t)\leqslant\frac{n}{4}$ by adapting the solution to Exercise 7.1 (p. 334).

Let $Y_i=p-X_i$, so that $X_i=1\Leftrightarrow Y_i=p-1$ and $X_i=0\Leftrightarrow Y_i=p$.

Then the conditional expectation of Y_i given that v_i has been chosen in the first t steps is $(p-1)p+p(1-p)=0$.

So

$$\begin{aligned}\mathbb{E}(Y_i)&=\mathbb{P}(v_i \text{ not chosen in first } t \text{ steps})\mathbb{E}(Y_i|v_i \text{ not chosen in first } t \text{ steps})\\&=\left(1-\frac{1}{n}\right)^t(p-1)\end{aligned}$$

Similarly

$$\mathbb{E}(Y_iY_j)=\left(1-\frac{2}{n}\right)^t(p-1)^2$$

So

$$\begin{aligned}\mathrm{Cov}(Y_i,Y_j)&=\mathbb{E}(Y_iY_j)-\mathbb{E}(Y_i)\mathbb{E}(Y_i)\\&=\left(1-\frac{2}{n}\right)^t(p-1)^2-\left(1-\frac{1}{n}\right)^{2t}(p-1)^2\\&=(p-1)^2\left(\left(1-\frac{2}{n}\right)^t-\left(1-\frac{1}{n}\right)^{2t}\right)\\&\leqslant 0\end{aligned}$$

Hence $\mathrm{Cov}(X_i,X_j)\leqslant 0$ since $\mathrm{Cov}(X_i,X_j)=\mathrm{Cov}(Y_i,Y_j)$, as $Y_i=p-X_i$.

Since $X_i(t)$ are indicators, $\mathrm{Var}(X_i(t))\leqslant\frac{1}{2}*\frac{1}{2}=\frac{1}{4}$.

So

$$\begin{aligned}\mathrm{Var}_{\mathbf{0}}(N_t)&=\sum_{i=1}^n\mathrm{Var}(X_i(t))+\sum_{i\neq j}\mathrm{Cov}(X_i(t),X_j(t))\\&\leqslant\sum_{i=1}^n\frac{1}{4}\\&=\frac{n}{4}\end{aligned}$$

Now consider N under the measure π:

Since the 0 colors are independent percolation on V, each node is colored white with probability p independently of all the other nodes.

So under π, $N \sim Bin(n,p)$.

Hence $\mathbb{E}_\pi(N)=np$ and $Var_\pi(N)=np(1-p)\leqslant \frac{1}{2}*\frac{1}{2}=\frac{1}{4}$.

So $\mathbb{E}_{\mathbf{0}}(N(t))-\mathbb{E}_\pi(N)=n(1-p)\left(1-\frac{1}{n}\right)^t$.

Let $\sigma^2:=\max\{\mathrm{Var}_{\mathbf{0}}(N_t),\mathrm{Var}_\pi(N)\}\leqslant\frac{n}{4}$.

Then

$$\begin{aligned}|\mathbb{E}_{\mathbf{0}}(N(t))-\mathbb{E}_\pi(N)| &\geqslant \sigma 2\sqrt{n}(1-p)\left(1-\frac{1}{n}\right)^t \\ &\geqslant \sigma 2(1-p)\exp\left\{-\frac{t}{n}\left(1+\frac{1}{n}\right)+\frac{\log n}{2}\right\}\end{aligned}$$

Let $r(t)=2(1-p)\exp\left\{-\frac{t}{n}\left(1+\frac{1}{n}\right)+\frac{\log n}{2}\right\}$.

Evaluating at $t_n:=\left[\frac{1}{2}n\log n-\left(c(p)-\frac{1}{2}\right)n\right]\left[1-\frac{1}{n+1}\right]$ gives $r(t_n)=2(1-p)e^{c(p)-\frac{1}{2}}$.

So $r^2(t_n)=4(1-p)^2e^{2c(p)-1}$.

Choosing $c(p)=\frac{1}{2}\left[1+\log\left(\frac{32}{12(1-p)^2}\right)\right]$ gives $r^2(t_n)=\frac{32}{3}$.

Thus for $t\leqslant t_n$, $r^2(t)\geqslant\frac{32}{3}$.

Hence by Remark 7.11, $t_{mix}\geqslant t_n\geqslant\frac{1}{2}n\log n-c(p)n$. ■

8.2 Upper bound on the mixing time

Levin, Peres, and Wilmer (2009) proved an upper bound for the mixing time of the Metropolis chain on proper colorings (Theorem 5.7, pp. 70-73). I will adapt this proof to find an upper bound for the mixing time of the variant outlined at the beginning of this section.

We create a grand coupling by doing the following at each step:

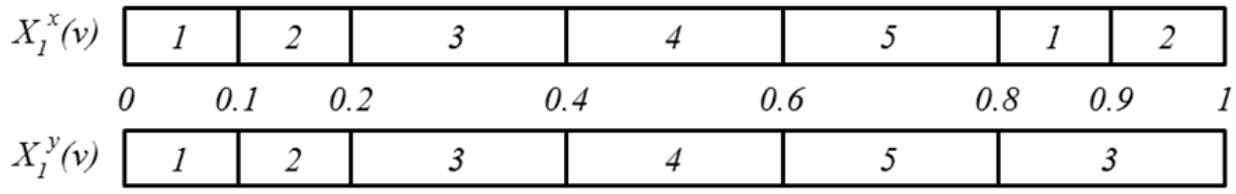
- Pick a node v at random.
- Flip a coin with probability p of heads.
- If heads, color the node white in each chain
- If tails, then for each chain split the interval $[0,1]$ up as follows:
  - Starting at 0, create segments of length $\min_{x\in\Omega}\{\mathbb{P}(X_1^x(v)=i)\}$ for $i\in\{1,\ldots,k\}$, where $\mathbb{P}(X_1^x(v)=i)$ is the probability of coloring node v with color i (proportional to $e^{-\beta N(v,i)}$)

– Then fill in the rest of the interval so that for each i the proportion of the interval designated to color i is equal to $\mathbb{P}(X_1^x(v)=i)$

Then generate a random number in $[0,1]$ and color the node with the color whose part of the interval the random number lies in

Note: This gives us a common source of randomness for all of the chains as required for a grand coupling, namely the node chosen, the result of the coin toss and the random number generated.

By splitting up the intervals in this way, at least the first k segments will be the same length in every chain. For example, suppose $X_1^x(v)$ has probabilities $(0.2, 0.2, 0.2, 0.2, 0.2)$ and $X_1^y(v)$ has probabilities $(0.1, 0.1, 0.4, 0.2, 0.2)$. Then the intervals would be split like this:



Let

$$\rho(x,y)=\sum_{v\in V}\mathbb{I}_{\{x(v)\neq y(v)\}}=\text{number of nodes where } x \text{ and } y \text{ disagree}$$

Suppose $\rho(x,y)=1$, so that x and y only disagree at some node v_0.

Let $\rho(X_1^x, X_1^y)$ be the distance after updating x and y in one step of the grand coupling.

The distance goes to zero if and only if v_0 is chosen (if the coin is heads then v_0 will be white in both cases; if it is tails then the split intervals for x and y are the same so v_0 will go to the same non-white color in both cases).

So $\mathbb{P}(\rho(X_1^x, X_1^y)=0)=\frac{1}{n}$.

Let w be a neighbour of v_0. Suppose w is chosen.

If the coin is heads, then w will be colored white in both x and y, so $\rho(X_1^x, X_1^y)=1$.

If the coin is tails, then w may not be the same in x and y since $x(v_0)\neq y(v_0)$.

So

$$\begin{aligned}\mathbb{P}(\rho(X_1^x, X_1^y)=2)=&\mathbb{P}(\text{a neighbour of } v_0 \text{ is chosen, coin toss is tails}\\ &\text{and colors picked disagree})\\ \leqslant&\frac{\Delta}{n}(1-p)\mathbb{P}(\text{colors picked disagree})\end{aligned}$$

Suppose the coin toss is tails. Then

$$\begin{aligned}\mathbb{P}(\text{colors picked agree}) &= \text{proportion of } [0,1] \text{ where split intervals agree} \\ &\geqslant \sum_{i=1}^{k} \min_{x\in\Omega}\{\mathbb{P}(X_1^x(w)=i)\}\end{aligned}$$

since at least the first k segments of the split intervals agree.

Now

$$\begin{aligned}\mathbb{P}(X_1(w)=i) &= \frac{e^{-\beta N(w,i)}}{\sum_{j=1}^{k} e^{-\beta N(w,j)}} \\ &= \frac{1}{\sum_{j=1}^{k} e^{-\beta(N(w,j)-N(w,i))}} \\ &\geqslant \frac{1}{ke^{\beta\Delta}} \qquad \text{since } N(w,j)-N(w,i)\geqslant -\Delta\end{aligned}$$

So

$$\mathbb{P}(\text{colors picked agree}) \geqslant \sum_{i=1}^{k} \frac{1}{ke^{\beta\Delta}} \;=\; \frac{1}{e^{\beta\Delta}}$$

and so

$$\mathbb{P}(\text{colors picked disagree}) \leqslant 1-\frac{1}{e^{\beta\Delta}}$$

Hence

$$\mathbb{P}(\rho(X_1^x, X_1^y)=2) \leqslant \frac{\Delta}{n}(1-p)\left(1-\frac{1}{e^{\beta\Delta}}\right)$$

So

$$\mathbb{E}(\rho(X_1^x, X_1^y)-1) \leqslant \frac{\Delta}{n}(1-p)\left(1-\frac{1}{e^{\beta\Delta}}\right)-\frac{1}{n}$$

Thus

$$\begin{aligned}\mathbb{E}(\rho(X_1^x, X_1^y)) &\leqslant 1+\frac{\Delta}{n}(1-p)\left(1-\frac{1}{e^{\beta\Delta}}\right)-\frac{1}{n} \\ &= 1-\frac{1}{n}\left[1-\Delta(1-p)\left(1-\frac{1}{e^{\beta\Delta}}\right)\right] \\ &< 1 \quad \text{for } \beta \text{ sufficiently small}\end{aligned}$$

$\left(\text{specifically, for } \beta < -\frac{1}{\Delta}\log\left(1-\frac{1}{\Delta(1-p)}\right)\right)$

Define $c(\Delta, p, \beta) := 1-\Delta(1-p)\left(1-\frac{1}{e^{\beta\Delta}}\right)$

Then

$$\mathbb{E}(\rho(X_1^x, X_1^y)) \leqslant 1-\frac{c(\Delta,p,\beta)}{n} < 1 \quad \text{for } \beta \text{ sufficiently small} \tag{1}$$

The rest of the proof is identical to the proof of Theorem 5.7, since the form of the inequality (1) matches exactly the form of the inequality (5.17) from the proof of Theorem 5.7.

Therefore it follows that $t_{mix}(\varepsilon) \leqslant c(\Delta, p, \beta)^{-1} n \left[\log n + \log \frac{1}{\varepsilon}\right]$, giving us an upper bound for the mixing time when β is sufficiently small. ■

References

Barbosa, Valmir C. (1996). *An Introduction to Distributed Algorithms*. Cambridge, Massachusetts: The MIT Press.

Bullimore, Sharon R. and Keith M. Briggs (2011). "Incorporating tight coloring into distributed heuristics for wireless channel assignment". In: *Journal of Discrete Algorithms* 00, pp. 1–15.

Levin, David A., Yuval Peres, and Elizabeth L. Wilmer (2009). *Markov Chains and Mixing Times*. Providence, Rhode Island: American Mathematical Society.

Tel, Gerard (2000). *Introduction to Distributed Algorithms*. Cambridge: Cambridge University Press.