Acta Informatica manuscript No.
(will be inserted by the editor)

Online multi-coloring on the path revisited

Marie G. Christ · Lene M. Favrholdt · Kim S. Larsen

Received: date / Accepted: date

Abstract Multi-coloring on the path is a model for frequency assignment in linear cellular networks. Two models have been studied in previous papers: calls may either have finite or infinite duration. For hexagonal networks, a variation of the models where limited frequency reassignment is allowed has also been studied.

We add the concept of frequency reassignment to the models of linear cellular networks and close these problems by giving matching upper and lower bounds in all cases. We prove that no randomized algorithm can have a better competitive ratio than the best deterministic algorithms. In addition, we give an exact characterization of the natural greedy algorithms for these problems.

All of the above results are with regard to competitive analysis. Taking steps towards a more fine-grained analysis, we consider the case of finite calls and no frequency reassignment and prove that, even though randomization cannot bring the competitive ratio down to one, it seems that the greedy algorithm is expected optimal on uniform random request sequences. We prove this for small paths and indicate it experimentally for larger graphs.

Keywords Online algorithms; graph coloring; competitive analysis.

1 Introduction

An online algorithm is an approximation algorithm that solves a problem without knowing the complete input to the problem from the beginning. The input (often called the request

Marie G. Christ
Department of Mathematics and Computer Science, University of Southern Denmark,
Campusvej 55, DK-5230 Odense M, Denmark
E-mail: christm@imada.sdu.dk

Lene M. Favrholdt
Department of Mathematics and Computer Science, University of Southern Denmark,
Campusvej 55, DK-5230 Odense M, Denmark
E-mail: lenem@imada.sdu.dk

Kim S. Larsen
Department of Mathematics and Computer Science, University of Southern Denmark,
Campusvej 55, DK-5230 Odense M, Denmark
E-mail: kslarsen@imada.sdu.dk

sequence) is given to the algorithm step by step, and at each step, the online algorithm must make a decision regarding the part of the input (the request) presented to the algorithm in that step. Most often, this decision is irrevocable, but sometimes relaxations of this requirement are of interest. We follow the tradition from the area of approximation algorithms and measure the performance of an online algorithm using competitive analysis [1,6,11,10], comparing the algorithm's performance to that of an optimal offline algorithm.

In this paper, we work with *cost minimization*. Let $\mathrm{ALG}(I)$ denote the cost of an online algorithm ALG on an input sequence I, and let $\mathrm{OPT}(I)$ denote the cost of an optimal offline algorithm on that input sequence. The algorithm ALG is called *c-competitive* if and only if there is a constant b (independent of I) such that for any input sequence, it holds that $\mathrm{ALG}(I) \leq c\,\mathrm{OPT}(I) + b$. The infimum over all c for which ALG is c-competitive is called the (asymptotic) *competitive ratio* of ALG. Establishing the result above using $b = 0$ in the inequality is referred to as the *absolute* competitive ratio. Absolute competitive ratio results are often less interesting when considering lower bounds since a few short request sequences can be the sole cause of a large value. When considering upper bounds, results obtained using the absolute competitive ratio may be correspondingly stronger.

1.1 The Problem

The motivation for this work comes from frequency allocation in *cellular networks*. Such networks consist of base transceiver stations (BTSs), each covering a cell. Each call within a cell is assigned a frequency for communicating with the BTS. BTSs in neighboring cells may interfere and, hence, cannot use the same frequencies. In all of our results, we are simply working with the underlying graph where BTSs are interpreted as nodes in the graphs and the possible interference as edges. Frequencies are modelled by colors and as bandwidth is limited, one wants to keep the number of frequencies used low.

The degree of detail and the type of scenarios that are modelled vary. Calls may be finite or infinite. In the case of finite calls, we say that a call is *cancelled* when it ends. Sometimes a limited degree of frequency reallocation is allowed at a given location or neighboring locations.

With regard to graph structure, researchers have focused on cellular networks in the form of paths (also referred to as *linear* cellular networks) and *hexagonal* grids, since these graph structures model different types of connection properties in cellular networks. The models are also known as the *highway* and *city* models, respectively. In this paper, we only consider the highway model (*paths*).

The *graph coloring* problem is well known [9]: the given graph must be colored in such a way that no two neighboring nodes receive the same color. We consider *multi-coloring*, where each node must be given a number of colors. The requirement is that the same color is only in use once per node and the set of colors used at any two neighboring nodes must be disjoint. A coloring fulfilling these requirements is called *legal*. We model colors using numbers starting from one, and our *objective* is to minimize the maximal color used anywhere in the graph.

In the *online* version of the problem, the graph is known from the beginning, but we do not know in advance how many colors each node must be given. Thus, a request is simply a node in the graph, and the online algorithm must assign an (additional) color to that node.

In the case where requests may be cancelled, the time of the cancellation (if any) is unknown to the online online algorithm until the time when the cancellation takes place. This case is called the *general* case, since it contains the case where cancellations do not

occur as a special case. The objective is still to minimize the maximal color used at any node after any request.

Usually when working with online algorithms, decisions are irrevocable, i.e., once a color is assigned to a node, this decision is final. If the algorithm is given some limited power to change the colors, we refer to this as *recoloring*. An algorithm is *d-recoloring* if, in the process of treating a request or cancellation, it may recolor up to a distance d away from the node of the request/cancellation.

We now discuss the results most closely related to ours. The terminology used in different papers has been inspired by terms from computer science, graph theory, and communication engineering. We have decided on a uniform terminology in the rest of this paper, using graphs and graph coloring concepts.

1.2 Related Work

Previous work relates to the path or to hexagonal graphs.

The path: For the path, there are no previous results on multi-coloring with recoloring allowed.

We first state the known results for the case without cancellations. Chan et al. [2] analyzed the greedy algorithm, GREEDY, which always uses the smallest color resulting in a legal coloring. They showed that the absolute competitive ratio of GREEDY is $\frac{3}{2}$ and that this is optimal. For the asymptotic competitive ratio, they showed a lower bound of $\frac{4}{3}$ and proposed an algorithm, HYBRID, with a competitive ratio of $(5-\sqrt{5})/2 \approx 1.382$. Chrobak and Sgall [4] closed the gap, designing an optimal algorithm, 4-BUCKET. These asymptotic results are obtained as the number of colors used approaches infinity; not using the number of requests, as is usually the case.

For the general case, Chan et al. [2] showed that the (absolute as well as asymptotic) competitive ratio of GREEDY is 2. They defined an algorithm, BORROW, and proved it optimal with regard to the absolute competitive ratio, getting a ratio of $\frac{5}{3}$. The upper bound of $\frac{5}{3}$ carries over to the asymptotic competitive ratio for which Chrobak and Sgall [4] proved a lower bound of $\frac{11}{7}$.

Hexagonal graphs: We give a brief account of the results on hexagonal graphs. These are all for the general case and are mainly concerned with a distributed setting where the algorithms are called *d-local* if they are allowed to view and change the coloring of nodes up to a distance of d away from the current request/cancellation.

Without recoloring, no online algorithm is better than 2-competitive [8].

With recoloring allowed, the results are significantly better: In [13], Witkowski and Zerovnik introduced a $\frac{33}{24}$-competitive 1-local algorithm. Janssen et al. developed a $\frac{4}{3}$-competitive algorithm for the case of 4-locality [8], strengthened to 2-locality by Sparl and Zerovnik [12]. Janssen et al. [8] also showed general lower bounds for hexagonal graphs: No d-local algorithm can have a competitive ratio better than $1+\frac{1}{4(d+1)}$. For 0-recoloring, the lower bound can be improved to $\frac{9}{7}$, independent of the view distance.

1.3 Our Results

We consider all cases for the path with recoloring allowed, providing matching upper and lower bounds. Our results are valid for the absolute as well as the asymptotic competitive ratio. When making coloring or recoloring decisions, the algorithms considered inspect only immediate neighbors.

For the general case, we devise a 1-competitive 1-recoloring algorithm, closing the problem for all d-recoloring algorithms, $d \geq 1$. For 0-recoloring algorithms, we argue that known general bounds carry over, and that Chrobak and Sgall's $\frac{4}{3}$-competitive 4-BUCKET algorithm is optimal. Having closed this problem, we also consider the most natural adaption of GREEDY to 0-recoloring, and show that its competitive ratio is exactly two. For the case without cancellations, these results carry over.

As is most often done, for the lower bound results above, we are implicitly assuming deterministic algorithms. It is natural to consider whether randomized algorithms can beat the deterministic lower bounds, as is the case for many other problems. We prove that randomization does not help in any of these cases. This completes the competitive analysis of this problem.

Taking steps towards a more fine-grained analysis, we also indicate that the fact that randomization does not help may be due to select non-uniform worst-case sequences, since we can show that the simple algorithm, GREEDY, is asymptotically expected 1-competitive (expected optimal, that is) on a path consisting of just four nodes with respect to a uniform distribution of requests to the four nodes. Thus, even though randomization can be proven not to help, the deterministically non-optimal greedy algorithm is expected optimal on such a uniform distribution. For larger graphs, we provide experimental results indicating the optimality of GREEDY under this distribution.

2 Competitive Analysis

We first establish results for the general case, since the upper bounds established here immediately carry over to the case without cancellations.

2.1 The General Case

In this setting, requests may be cancelled again, freeing some previously used colors for possible reuse. Since it is requests that are cancelled, it is the color used when this request was issued that is becoming available for possible reuse, as opposed to an arbitrary color in use at that node. Thus, we will sometimes talk about the request being colored rather than the node receiving an additional color.

2.1.1 Recoloring distance 1—a 1-competitive algorithm

First, we consider 1-recoloring algorithms. The possibility to recolor at neighboring nodes when a new request arrives allows us to design a 1-competitive online algorithm. Since the upper bound carries over to larger recoloring distances, this closes the problem for d-recoloring, $d \geq 1$.

The nodes are divided into two sets called *upper* and *lower*, such that every second node belongs to *upper* and the remaining nodes belong to *lower*. The following invariant is maintained: After each request,

– each node in *lower* uses consecutive colors starting with the color 1.
– each node in *upper* uses consecutive colors ending with a color no larger than the optimal number of colors for the sequence of requests seen so far.

Informally, when a color request r arrives at a node v in *upper*, the color interval of v is extended below, if possible. Otherwise, it is extended above. When a color request r arrives at a node v in *lower*, the color interval is extended above, i.e., the lowest color c not already in use at v is used. For each neighbor of v, if c is already used for some request s at that neighbor, s is recolored using the lowest possible color. When a request r at a node v is cancelled, the request at v with the highest color is recolored using the color c of r (if c is not the highest color used at v). To make it precise, we give the details in Algorithm 1. This also serves as an introduction to the notation we use in the proofs of theorems to follows.

Consider a path of ℓ nodes, numbered from 1 through ℓ. Let $f_t(v)$ denote the set of colors assigned to node v after the first t requests, starting with request 1. We introduce two additional nodes, 0 and $\ell+1$, such that each of the ℓ nodes has two neighbors. The extra nodes do not receive any requests and therefore no colors. Also, for notational convenience, we define $f_0(v) = \emptyset$ for all v. To smoothly handle initially empty sets of colors in the algorithm, we define that if $f_t(v)$ is the empty set, then $\min f_t(v) = \max f_t(v) = 0$.

The new 1-recoloring algorithm, GREEDYOPT, is given in Algorithm 1.

Algorithm 1 The 1-recoloring algorithm, GREEDYOPT, treating the tth request.

```
1:  Assume that the tth request, r, is to node v
2:  min ← min f_{t−1}(v)
3:  max ← max f_{t−1}(v)
4:  if r is a color request then
5:      if v ∈ upper then
6:          if f_{t−1}(v) = ∅ then
7:              give r color max(f_{t−1}(v−1) ∪ f_{t−1}(v+1)) + 1
8:          else if min > 1 ∧ min−1 ∉ f_{t−1}(v−1) ∪ f_{t−1}(v+1) then
9:              give r color min−1
10:         else
11:             give r color max+1
12:     else
13:         /* v ∈ lower */
14:         if there exists a request s at node v−1 with color max+1 then
15:             recolor s, giving it color max f_{t−1}(v−1) + 1
16:         if there exists a request s at node v+1 with color max+1 then
17:             recolor s, giving it color max f_{t−1}(v+1) + 1
18:         give r color max+1
19: else
20:     /* r is a cancellation */
21:     if the color of r is different from max f_{t−1}(v) then
22:         recolor the request that has color max f_{t−1}(v), giving it the color of r
23:     remove request r
```

Theorem 1 *Algorithm* GREEDYOPT *is correct and 1-competitive.*

Proof It is easily checked that after any number t of requests,

- each node $v \in$ *lower* is colored with a set $\{1,2,\ldots,c\}$ of consecutive colors, where $c < \min(f_t(v-1) \cup f_t(v+1))$ if $f_t(v-1) \cup f_t(v+1) \neq \emptyset$
- each node $v \in$ *upper* is colored with a set $\{d, d+1,\ldots,e\}$ of consecutive colors, where $d > \max(f_t(v-1) \cup f_t(v+1))$ if $f_t(v-1) \cup f_t(v+1) \neq \emptyset$.

This immediately implies that the coloring is legal. Furthermore, whenever a new color c is used at a node v, we have that $f_t(v-1) \cup f_t(v) = \{1,2,\ldots,c\}$ or $f_t(v) \cup f_t(v+1) = \{1,2,\ldots,c\}$. Hence, the coloring is optimal. □

2.1.2 Recoloring distance 0—optimality and the performance of GREEDY

A lower bound of $\frac{4}{3}$ for multi-coloring on the path when there are no cancellations or recolorings was given in [2]. For the sequence used, 0-recoloring gives no extra possibilities, and, hence, this lower bound carries over to 0-recoloring algorithms. For the upper bound, the proof by Chrobak and Sgall [4], carries over to the general case (even though the algorithm does not use recoloring). Their algorithm 4-BUCKET is $\frac{4}{3}$-competitive and therefore optimal, so this subproblem is completely settled.

Because of its simplicity, it is also interesting to consider the most natural greedy algorithm for this problem: At a request for a new color, GREEDYADAPT colors using the smallest color resulting in a legal coloring, just like GREEDY. At a cancellation, GREEDYADAPT recolors the request with maximal color using the color available due to the cancellation. Effectively, this can be thought of as always cancelling the request with maximal color.

For the general case with no recoloring allowed, Chan et al. [2] showed that the competitive ratio of the greedy algorithm is 2. The lower bound was proven by preventing reusage of colors by forcing pairs of nodes to lose all colors they have in common [3]. When recoloring is allowed, this does not necessarily happen. However, we can still show that the competitive ratio of GREEDYADAPT is exactly 2.

Theorem 2 *For* 0*-recoloring, the competitive ratio of* GREEDYADAPT *is* 2.

Proof For the upper bound, assume the nodes on the path are numbered consecutively and that the maximal competitive ratio is reached when giving the $(t+1)$*st* request and that this is to node v. Due to greediness of GREEDYADAPT, the new maximal color cannot be larger than $|f_t(v)|+1+|f_t(v-1)|+|f_t(v+1)|$. OPT uses a color which is at least $|f_t(v)|+1+\max\{|f_t(v-1)|,|f_t(v+1)|\}$. Thus, the ratio is bounded by

$$\frac{|f_t(v)|+1+|f_t(v-1)|+|f_t(v+1)|}{|f_t(v)|+1+\max\{|f_t(v-1)|,|f_t(v+1)|\}}$$

$$\leq \frac{|f_t(v)|+1+2\max\{|f_t(v-1)|,|f_t(v+1)|\}}{|f_t(v)|+1+\max\{|f_t(v-1)|,|f_t(v+1)|\}} < 2$$

However, also note that the ratio could be arbitrarily close to 2.

Now we turn to the lower bound. We number the nodes $1,2,\ldots,\ell$ and decide on an optimal number of colors exponential in ℓ, defined by $n = 2^{\lfloor \frac{\ell+2}{4} \rfloor}$. More precisely, we give a sequence of requests and cancellations where OPT at no point uses more than n colors. This can be ensured by arranging that no two neighboring nodes ever have more than n uncancelled requests together. Since OPT knows the entire sequence in advance, and therefore

Algorithm 2 The construction of the worst case sequence for GREEDYADAPT.

```
1:  /* Initialization */
2:  Request (2, n/2)
3:  Request (1, n/2)
4:  Cancel (2, n/2)
5:  Request (3, n/2)
6:  for i = 1 to 2 log_2 n − 2 do
7:      /* Phase i */
8:      s ← n / 2^⌈i/2⌉+1        /* halve s in every second iteration */
9:      Request (i + 1, s)
10:     k ← 3
11:     while there are still requests remaining from the previous phase do
12:         Cancel (i + m, s), where m ∈ {0, 2} is chosen such that the highest colors
            remaining from the previous phase are found at node i + m
13:         Request (i + k, s)
14:         k ← k + 2  mod 4
```

also n, it can maintain a coloring where every second node uses consecutive colors in increasing order starting from one, and the other nodes use consecutive colors starting from n in decreasing order. At any cancellation, it can recolor to maintain this invariant.

The construction of the sequence is done in an initialization step followed by a number of phases. We show that during the last phase, GREEDYADAPT uses $2n-2$ colors, obtaining a ratio of $2-\frac{2}{n} = 2-\frac{1}{2^{\lfloor(\ell-2)/4\rfloor}}$. Hence, for arbitrarily long paths, we get arbitrarily close to a ratio of 2.

The construction of the sequence is described in Algorithm 2. We let *Request* (v,m) denote a sequence of m requests to each of the nodes $v+4i$, $i = 0,1,\ldots,\lfloor\frac{\ell-v}{4}\rfloor$. Similarly, *Cancel* (v,m) denotes a sequence of m cancellations at each of the nodes $v+4i$, $i = 0,1,\ldots,\lfloor\frac{\ell-v}{4}\rfloor$.

The proof is self-contained, but it may help to refer to Figures 3 and 4 at the end of the paper, where we illustrate Initialization and Phases 1–4 for $\ell = 15$. Node numbers are shown horizontally, and color numbers are shown vertically. We use white and black to indicate the phases in which requests were made. Black is used for requests from an odd numbered phase.

Note that, after Phase i, at least $\ell-1-2i$ nodes still have uncancelled requests ($\ell-2i$ nodes, if ℓ is odd). Thus, after the last phase, there will be requests to at least $\ell-1-2\cdot(2\log_2 n-2) = \ell+3-4\cdot\lfloor\frac{\ell+2}{4}\rfloor \geq 1$ nodes.

During initialization, GREEDYADAPT uses n colors. In each of Phases $2i-1$ and $2i$, $i \geq 1$, it introduces $\frac{n}{2^{i+1}}$ new colors compared to the previous phase. Hence, Phase $2i$ contains $\frac{n}{2^i}$ more colors than Phase $2i-2$. It follows that, in the last phase, GREEDYADAPT uses $n+\frac{n}{2}+\frac{n}{4}+\ldots+2 = 2n-2$ colors. □

2.1.3 Randomized algorithms

We note that for d-recoloring algorithms, if $d \geq 1$, then randomization cannot contribute anything further, since we have an optimal algorithm. For $d = 0$, Theorem 3 of the next section gives us a randomized lower bound of $\frac{4}{3}$, matching the deterministic upper bound from this section.

2.2 Multi-Coloring without Cancellations

The optimal algorithm from the previous section is also optimal for the more restricted case without cancellations, except for 0-recoloring. For the case of 0-recoloring algorithms, we observe that the lower bound of $\frac{4}{3}$ from [2] carries over, since the lower bound sequence does not contain deletions. Hence, the 4-BUCKET algorithm from [2] is also optimal for the case without cancellations.

Thus, the only remaining question is whether randomization might beat the deterministic lower bound of $\frac{4}{3}$ for 0-recoloring algorithms. As usual, the hope comes from the fact that different deterministic strategies do not necessarily have the same worst-case sequences. Thus, in addition to randomization possibly giving better results in practice, it may be possible to prove a better competitive ratio (against an oblivious adversary [1]). However, we can establish that the search for such a randomized algorithm is in vain. To settle this, we use Yao's principle [14, 1] and establish that no randomized algorithm can be better than the best deterministic algorithm:

Theorem 3 *For a path with at least eight nodes, no randomized online* 0*-recoloring algorithm can have a competitive ratio smaller than* $\frac{4}{3}$ *against an oblivious adversary.*

Proof Let the eight nodes be labelled $v_1, v_2, \ldots, v_8$. Let $\left\{\sigma_j^n\right\}$ denote the set of all possible request sequences of length n. We define a probability distribution $y^n(j)$ on this set. We will allow multiple requests to be given in each step; with a more cumbersome formulation, these could just as well be given one at a time. At Step 1, $\frac{n}{4}$ requests arrive at node v_3 and $\frac{n}{4}$ requests arrive at node v_6. At Step 2, with probability $\frac{1}{2}$, at each of the nodes v_4 and v_5, $\frac{n}{4}$ requests arrive (Case 1). With probability $\frac{1}{2}$, at each of the nodes v_1 and v_8, $\frac{n}{4}$ requests arrive (Case 2). All other request sequences are assigned zero probability.

In Case 1, an optimal algorithm would assign different colors to the requests from Step 1, so that the colors used at nodes v_3 and v_6, respectively, can be reused for the upcoming requests at nodes v_5 and v_4, respectively, resulting in $2 \cdot \frac{n}{4} = \frac{n}{2}$ used colors. In Case 2, no two neighboring nodes receive requests. Thus, the optimal algorithm would assign v_3 and v_6, as well as the subsequent $\frac{n}{4}$ requests to each of v_1 and v_8 the same colors (from 1 through $\frac{n}{4}$), so that the overall number of used colors would be $\frac{n}{4}$. The expected cost of the optimal algorithm would therefore be $\mathrm{E}[\mathrm{OPT}] = \frac{1}{2} \cdot \frac{n}{2} + \frac{1}{2} \cdot \frac{n}{4} = \frac{3}{8}n$.

Let $\mathscr{A}$ denote the set of all deterministic algorithms for the problem. We derive a lower bound on the performance of any algorithm in this set. After the coloring in Step 1, let x denote the fraction of the colors at v_6 that were not used at v_3. Thus, up to this point, a total of $(1+x)\frac{n}{4}$ colors have been used.

In Case 1, the $(1-x)\frac{n}{4}$ colors used at both v_3 and v_6 cannot be used at v_4 and v_5. Thus, $2(1-x)\frac{n}{4}$ new colors have to be assigned, resulting in an overall result of $(1+x)\frac{n}{4} + 2(1-x)\frac{n}{4} = (3-x)\frac{n}{4}$ colors. In Case 2, at Step 2, the lowest total number of used colors is $(1+x)\frac{n}{4}$, achievable by using the same colors as in Step 1. Therefore the expected value of any online algorithm ALG on the given probability distribution of $y^n(j)$ cannot be smaller than $\frac{1}{2}(3-x)\frac{n}{4} + \frac{1}{2}(1+x)\frac{n}{4} = \frac{n}{2}$.

A prerequisite for using Yao's principle is that the cost for OPT goes towards infinity for increasing length of the request sequences, and this is clearly fulfilled here. Then the lower bound of any randomized online algorithm is at least as large as the limit for n going towards infinity of the quotient of the infimum of the expected value of all deterministic algorithms to the expected value of the optimal algorithm:

$$\liminf_{n\to\infty} \frac{\inf_{\mathrm{ALG}\in\mathscr{A}} \mathrm{E}_{y^n(j)}\left[\mathrm{ALG}\left(\sigma_j^n\right)\right]}{\mathrm{E}_{y^n(j)}\left[\mathrm{OPT}\left(\sigma_j^n\right)\right]} \geq \liminf_{n\to\infty} \frac{\frac{n}{2}}{\frac{3}{8}n} = \frac{4}{3}.$$

Since all requests given to a node are given consecutively, having 0-recoloring available gives no extra possibilities. □

3 Uniformly Random Request Sequences

Having completely settled all issues regarding competitive analysis of the problem, we initiate a more fine-grained analysis of the probabilistic issues by investigating uniformly random request sequences in the model without cancellation.

3.1 The Smallest Non-Trivial Graph

The following result shows that on paths of four nodes where requests are given to nodes uniformly at random, GREEDY is asymptotically expected 1-competitive (expected optimal).

If two requests for different nodes are given the same color, we refer to these as a *pair*. A request which has been given a color different from any color given to any other request is referred to as a *singleton*. If we have a singleton r and a new request r' (at a different node) is given the same color as the singleton, then r and r' become a pair and r is no longer a singleton. Clearly, a new request either becomes a singleton or becomes part of a pair and in the process eliminates a singleton.

On a four-node path, we denote the nodes by v_1, v_2, v_3, and v_4. We refer to v_2 and v_3 as *inner* nodes and the two others as *outer* nodes.

We start with two simple observation:

Lemma 1 *When using* GREEDY, *there cannot be singletons on nodes that are a distance of more than one apart.*

Proof Assume to the contrary that this happened. Then consider the first singleton to be colored c at a node at a distance of more than one away from another node with singletons, the latest of which was given color c'. Clearly, because of greediness, we cannot have that $c > c'$, so $c < c'$. However, then the color c' was not given in a greedy fashion, since c must have been available. We can conclude that singletons must reside on at most two consecutive nodes. □

Lemma 2 *On a four-node path, unless a configuration has singletons on both inner nodes and nowhere else,* GREEDY *has used the optimal number of colors.*

Proof We consider all cases, as limited by Lemma 1. Assume that all singletons reside on an outer node and on its neighbor, and assume without loss of generality that these two nodes are v_1 and v_2. Since all pairs must have one of their two requests on one of these nodes, all colors used on v_3 and v_4 are also used on v_1 and v_2. Thus, since the total number of requests to v_1 and v_2 is a lower bound on the number of colors needed, GREEDY's coloring is optimal. This argument includes all situations where all singletons reside on one node. □

A sequence of independent stochastic variables $\{X_i\}_{i\geq 1}$, where $\text{Prob}[X_i = 1] = \text{Prob}[X_i = -1] = \frac{1}{2}$, is called a *simple random walk* [5]. It is well known that if we define $T_n = \sum_{i=1}^{n} X_i$, then $\lim_{n\to\infty} \frac{E[|T_n|]}{\sqrt{n}} = \sqrt{\frac{2}{\pi}}$ [7]. We use this fact in the form of the following equivalent statement: $E[|T_n|] \leq \sqrt{\frac{2}{\pi}}\sqrt{n} + \varepsilon_n$, where we use ε_n as notation for some function such that $\varepsilon_n \in o(\sqrt{n})$. Due to the absolute value, this is equivalent to considering *one-sided* random walks (also called a random walk with a barrier or with reflection), where one then does not need the absolute value. A one-sided random walk is defined exactly as above, except that when $T_i = 0$, then $\text{Prob}[X_{i+1} = 1] = 1$. We only use the following:

Observation 1 $E[T_n] \leq \sqrt{\frac{2}{\pi}}\sqrt{n} + \varepsilon_n$.

Let S_i^j be the stochastic variables denoting the number of singletons on node v_j after the ith request. Define $S_0 = 0$ and for $i \geq 1$,

$$S_i = \begin{cases} 0, & \text{if } \sum_{j=1}^{4} S_i^j = 0 \\ \sum_{j=1}^{4} S_i^j, & \text{if } S_i^1 > 0 \text{ or } S_i^4 > 0 \\ |S_i^2 - S_i^3|, & \text{otherwise} \end{cases}$$

Lemma 3 *The expected value of S_n is bounded from above by $\sqrt{\frac{2}{\pi}}\sqrt{n} + \varepsilon_n$.*

Proof For $i \geq 1$, we define the stochastic variable $X_i = S_i - S_{i-1}$. We will prove that $\{X_i\}$ is a one-sided simple random walk from which the result will follow.

We consider the cases in the definition of S_{i-1}:

If $\sum_{j=1}^{4} S_{i-1}^j = 0$, clearly, $X_i = 1$ with probability one.

If $S_{i-1}^1 > 0$ or $S_{i-1}^4 > 0$, we assume without loss of generality that $S_{i-1}^1 > 0$. Then, according to Lemma 1, $S_{i-1}^3 = S_{i-1}^4 = 0$. By a request to v_1 or v_2, S_i (like S_{i-1}) would be calculated by the second case in the definition, and clearly, $X_i = 1$. A request to v_3 would eliminate a singleton on v_1, and S_i would be calculated by the first, second, or third case, depending on S_{i-1}^1 and S_{i-1}^2. In each case, $X_i = -1$. By a request to v_4, S_i would be calculated by the second case, if $S_{i-1}^1 \geq 2$, by the first case, if $S_{i-1}^1 = 1$ and $S_{i-1}^2 = 0$, and by the second or third case otherwise. In each case, $X_i = -1$.

In the remaining case, we have that $S_{i-1}^1 = S_{i-1}^4 = 0$ and that $S_{i-1}^2 > 0$ or $S_{i-1}^3 > 0$. Assume without loss of generality that $S_{i-1}^2 \geq S_{i-1}^3$. If $S_{i-1}^2 = S_{i-1}^3$, then $S_{i-1} = 0$ and $X_i = 1$ with probability one (S_i, like S_{i-1}, would be calculated by the third case). Otherwise, we have $X_i = 1$, if v_1 or v_2 is requested, and $X_i = -1$, if v_3 or v_4 is requested. (If $S_{i-1}^3 > 0$, then S_i is calculated by the third case in the definition. If $S_{i-1}^3 = 0$, then S_i is calculated by the second case, if v_1 is requested, and by the third case otherwise.)

Thus, $\{X_i\}$ is a one-sided simple random walk and the result follows from Observation 1. □

Corollary 1 *After m requests, given uniformly at random to nodes in a four-node path graph, if exactly one inner node and no outer nodes has singletons, the expected number of singletons is at most $\sqrt{\frac{2}{\pi}}\sqrt{m} + \varepsilon_m$.*

Proof This follows from Lemma 3 and the observation that, by definition of S_i, in the case of the corollary formulation, S_i simply counts the number of singletons on the inner node. □

We can now prove that under a uniform distribution of requests, meaning that requests will be for each of the four nodes with probability $\frac{1}{4}$ each, GREEDY is asymptotically expected 1-competitive.

Theorem 4 *For a path of four nodes,* GREEDY *is asymptotically expected optimal under a uniform distribution of requests.*

Proof By Lemma 2, GREEDY is optimal unless it enters a configuration with singletons on both of and only on the inner nodes. If the configuration changes away from that, GREEDY is optimal again. Thus, as a worst-case consideration, we analyze the asymptotic behavior for this case, i.e., in the rest of this section, we assume that we are in a configuration with singletons on both of the inner nodes and only on these nodes.

Let X_i be the stochastic variable such that

$$X_i = \begin{cases} 1, & \text{if the } i\text{th request becomes a new singleton} \\ -1, & \text{if the } i\text{th request eliminates a singleton} \end{cases}$$

By the case assumption, if the ith request is to one of the two inner nodes, it becomes a new singleton, and otherwise, it forms a pair with an already existing singleton. Clearly, $\text{Prob}[X_i = 1] = \text{Prob}[X_i = -1] = \frac{1}{2}$.

By Lemma 2, the only way to enter this state with singletons on both and exclusively on the inner nodes is by first having singletons exclusively on one of the inner nodes, and after that get a request to the other inner node. Let Y denote the number of singletons just before this step, i.e., the number of singletons at the last time when all the singletons were exclusively on one inner node. Let Z denote the number of pairs present at that time. Thus, $m = Y + 2Z$ is the number of requests up to this point.

We consider the situation after p requests in this state and let I refer to the entire input sequence with $n = m + p$ requests. Let $S_p = \sum_{i=1}^{p} X_i$.

GREEDY ends with $Y + S_p$ singletons and thus makes $\frac{p-S_p}{2}$ pairs, so it uses $Z + Y + \frac{p+S_p}{2}$ colors. The sequence $\{X_i\}$ is a one-sided simple random walk, so by Observation 1, $E[S_p] \leq \sqrt{\frac{2}{\pi}}\sqrt{p} + \varepsilon_p$. By Corollary 1, $E[Y] \leq \sqrt{\frac{2}{\pi}}\sqrt{m} + \varepsilon_m$. Additionally, $Z + Y = \frac{m-Y}{2} + Y = \frac{m+Y}{2}$. By linearity of expectation,

$$E\left[\frac{m+Y}{2} + \frac{p+S_p}{2}\right] \leq \frac{m + \sqrt{\frac{2}{\pi}}\sqrt{m} + \varepsilon_m}{2} + \frac{p + \sqrt{\frac{2}{\pi}}\sqrt{p} + \varepsilon_p}{2}.$$

Any algorithm, also OPT, must use at least $\max\left\{Z + \frac{Y+p}{2}, Z + Y\right\}$ colors. These expressions coincide for $Y = p$. We proceed by treating the two cases of $Y < p$ and $p \leq Y$ separately.

First, consider the case where $Y < p$. Then $Z + \frac{Y+p}{2}$ is the maximum. Since, $Z + \frac{Y+p}{2} = \frac{m-Y}{2} + \frac{Y+p}{2} = \frac{m+p}{2}$, we get

$$\frac{E[\text{GREEDY}(I)]}{E[\text{OPT}(I)]} \leq \frac{\frac{m+\sqrt{\frac{2}{\pi}}\sqrt{m}+\varepsilon_m}{2} + \frac{p+\sqrt{\frac{2}{\pi}}\sqrt{p}+\varepsilon_p}{2}}{\frac{m+p}{2}} = 1 + \varepsilon'_m + \varepsilon'_p,$$

where $\varepsilon'_m = \varepsilon_m + \sqrt{\frac{2}{\pi}}\sqrt{m}$ and $\varepsilon'_p = \varepsilon_p + \sqrt{\frac{2}{\pi}}\sqrt{p}$.

Next, consider the case where $p \leq Y$. Then $Z+Y$ is the maximum. Since $Z+Y = \frac{m-Y}{2}+Y = \frac{m+Y}{2}$, with the same definition of ε'_m and ε'_p, we obtain that

$$\begin{aligned}\frac{E[\textsc{Greedy}(I)]}{E[\mathrm{OPT}(I)]} &\leq \frac{\frac{m+\sqrt{\frac{2}{\pi}}\sqrt{m}+\varepsilon_m}{2}+\frac{p+\sqrt{\frac{2}{\pi}}\sqrt{p}+\varepsilon_p}{2}}{\frac{m+Y}{2}} \\ &\leq \frac{\frac{m+\sqrt{\frac{2}{\pi}}\sqrt{m}+\varepsilon_m}{2}+\frac{p+\sqrt{\frac{2}{\pi}}\sqrt{p}+\varepsilon_p}{2}}{\frac{m+p}{2}} \\ &= 1+\varepsilon_m+\varepsilon_p.\end{aligned}$$

Any constant upper bound on the terms dependent on m and p in $\varepsilon'_m+\varepsilon'_p$ can be absorbed in the additive constant in the definition of the competitive ratio. For m and p approaching infinity, the additive term approaches zero, so the competitive ratio approaches one. □

To extend the proof above to general paths, an obvious strategy would be to transfer the result for a four-node graph to a result for a four-node component on a longer path. Then convert the expected result to a low probability of being far away from the expected value. The probabilities of these finitely many groups of four nodes can be summed up, and the result converted back to an expected value for the larger graph. However, it is not obvious how to establish the first partial result, since the proof strongly relies on properties coming from knowing that there is no interference from other neighboring nodes of v_1 and v_4. And, in fact, we are not convinced such a result holds. We will return to a discussion of this in the concluding remarks.

3.2 Experimental Results on Larger Graphs

We have proved that GREEDY approaches optimal performance for an increasing number of requests on a path of length four. Empirical evidence suggests that GREEDY is approaching a performance ratio of one compared to OPT for longer paths as well.

We have run a simulation using a uniform distribution on paths with a varying number of nodes, the result of which can be seen in Figure 1. The x-axis displays the density, i.e., the average number of requests per node. As measuring points, we have used densities that are ten times and fifteen times powers of two, i.e., of the form $10 \cdot 2^i$ and $15 \cdot 2^i$, up to $10,000$. We found that the variance was quite high, so for each measuring point, we have taken the average of 100 runs.

Variance is, not surprisingly, greatest for small densities. However, after some initial fluctuation for small densities, all curves reach a maximum after which the value decreases for increasing densities. Notice also that the curves for graphs with more nodes consistently appear above curves with fewer nodes, but then seem to have smaller (more negative, that is) slope. Thus, it appears plausible that these curves could all be approaching one.

4 Summary and Future Work

Regarding competitive analysis, we have settled all issues for multi-coloring on the path regardless of whether or not cancellations may occur, whether or not randomization is applied, and to what extend recoloring is permitted. In addition, we have characterized the behavior of the natural recoloring adaption of the greedy algorithm.

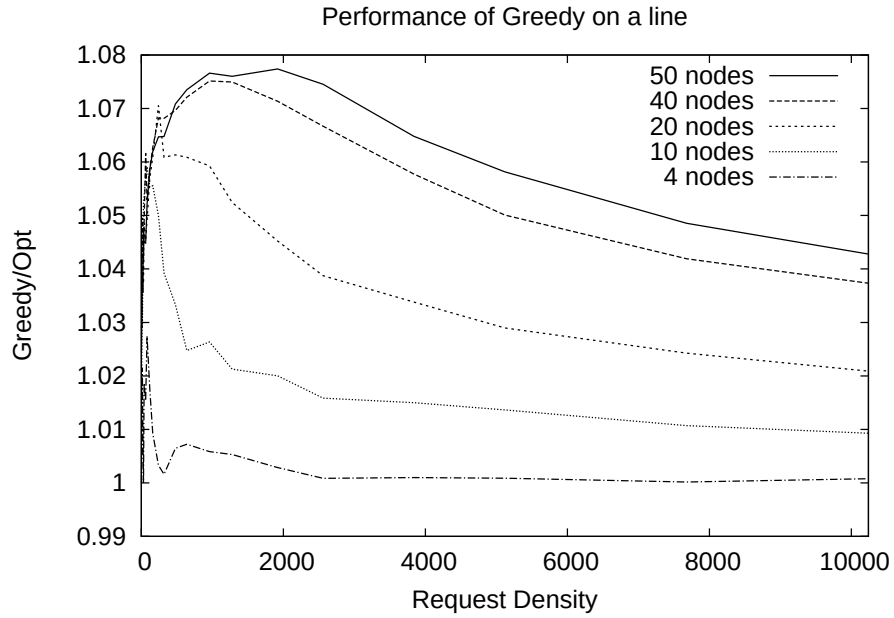


Fig. 1 Simulation of GREEDY on paths of different lengths.

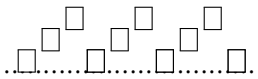


Fig. 2 A "bad" configuration: $3/2$ times as many colors as necessary are in use, and a request to any of the inner nodes will increase the number of colors.

In extension of that, we have initiated the analysis of probabilistic issues. There are many directions this work could take. We have started considering uniformly distributed request sequences with some analytical and some experimental work. This could be continued, generalizing our results, considering other distributions, or proceeding in the direction of hexagonal grids.

It is not a trivial matter to extend the results from the four-node case. In fact, it is possible that the greedy algorithm is not expected optimal for longer paths, depending on how this is defined. We believe that configurations similar to the one in Figure 2, though occurring with low probability, are the cause of the peaks seen in the experimental results for not too dense request sequences. Notice also that for such configurations, the probability of progressing towards a larger competitive ratio is temporarily more than a half, which means that we do not see how to apply random walk techniques to obtain results. For high request density, we are convinced that the small probability of entering this type of configuration is outweighed by the higher probability of getting a large number of requests to two neighboring nodes, which will force any algorithm, including OPT, to use a lot of colors, therefore giving near optimal results for the online algorithm. We leave the full theoretical analysis of probabilistic behavior on larger graphs as an interesting open problem.

Acknowledgment

This work was supported in part by the Danish Council for Independent Research.

References

1. Borodin, A., El-Yaniv, R.: Online Computation and Competitive Analysis. Cambridge University Press (1998)
2. Chan, J., Chin, F., Ye, D., Zhang, Y., Zhu, H.: Frequency allocation problems for linear cellular networks. In: 17th International Symposium on Algorithms and Computation, *LNCS*, vol. 4288, pp. 61–70. Springer (2006)
3. Chan, J., Chin, F., Ye, D., Zhang, Y., Zhu, H.: Frequency allocation problems for linear cellular networks, full version. Personal communication (2011)
4. Chrobak, M., Sgall, J.: Three results on frequency assignment in linear cellular networks. In: 5th International Conference on Algorithmic Aspects in Information and Management, *LNCS*, vol. 5564, pp. 129–139. Springer (2009)
5. Durrett, R.: Probability: Theory and Examples. Dixbury Press (1991)
6. Graham, R.L.: Bounds for certain multiprocessing anomalies. Bell Systems Technical Journal **45**, 1563–1581 (1966)
7. Hoffmann-Jørgensen, J.: Probability with a View towards Statistics, *Chapman & Hall Probability Series*, vol. I. Chapman & Hall (1994)
8. Janssen, J., Krizanc, D., Narayanan, L., Shende, S.: Distributed online frequency assignment in cellular networks. J. Algorithms **36**(2), 119–151 (2000)
9. Jensen, T., Toft, B.: Graph Coloring Problems. John Wiley & Sons (1995)
10. Karlin, A., Manasse, M., Rudolph, L., Sleator, D.: Competitive snoopy caching. Algorithmica **3**, 79–119 (1988)
11. Sleator, D., Tarjan, R.: Amortized efficiency of list update and paging rules. Communications of the ACM **28**(2), 202–208 (1985)
12. Sparl, P., Zerovnik, J.: 2-local 4/3-competitive algorithm for multicoloring hexagonal graphs. Journal of Algorithms **55**(1), 29–41 (2005)
13. Witkowski, R., Zerovnik, J.: 1-local 33/24-competitive algorithm for multicoloring hexagonal graphs. In: 8th International Workshop on Algorithms and Models for the Web Graph, *LNCS*, vol. 6732, pp. 74–84 (2011)
14. Yao, A.C.: Probabilistic computations: Toward a unified measure of complexity. In: 18th FOCS, pp. 222–227 (1977)

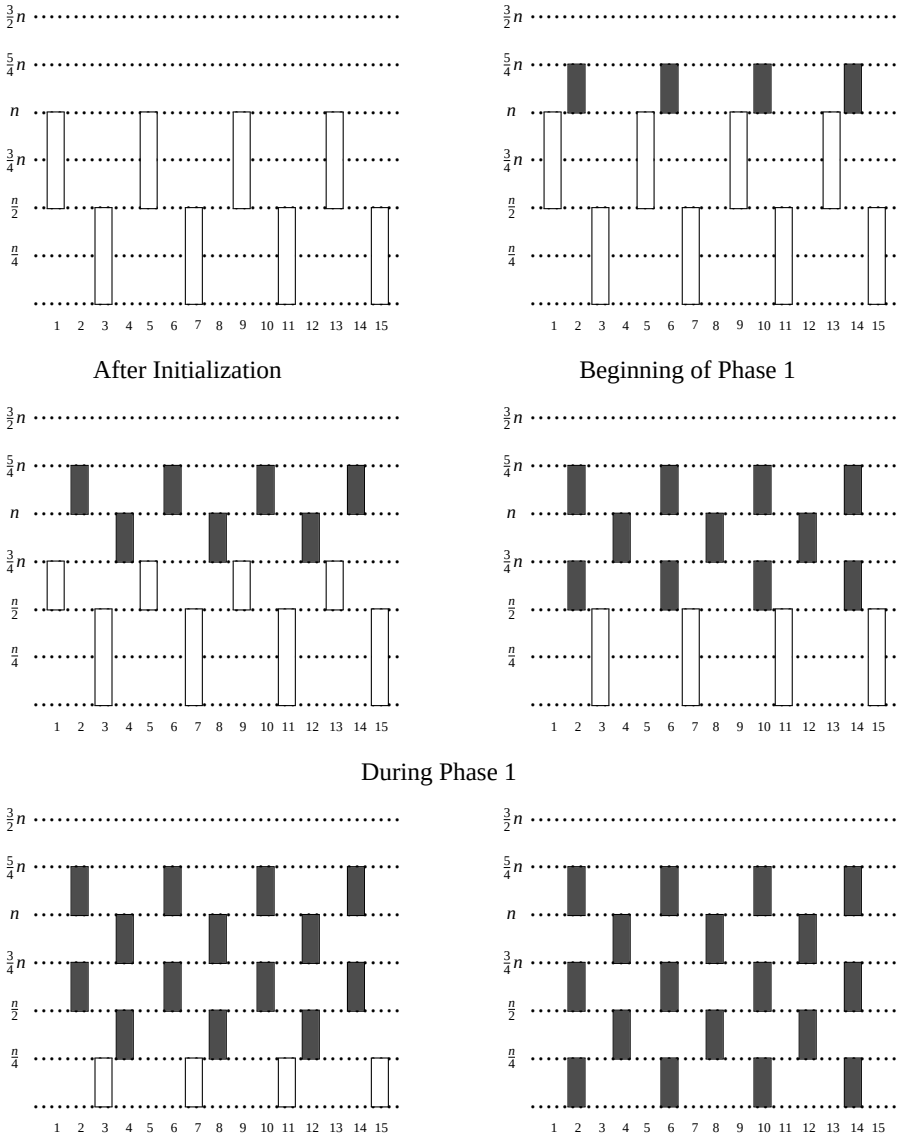
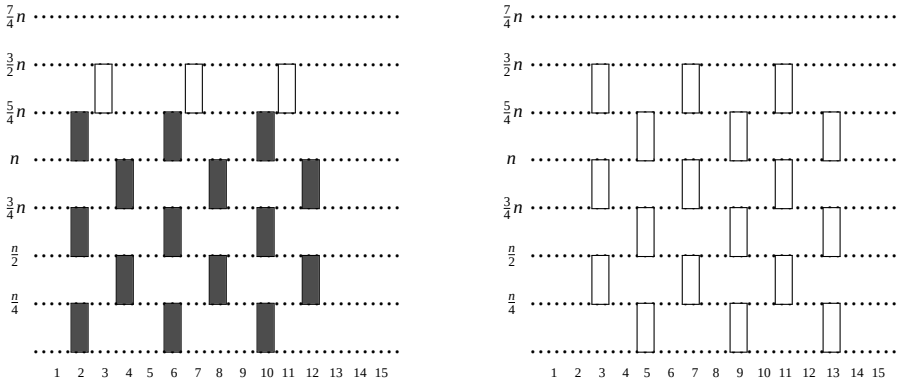
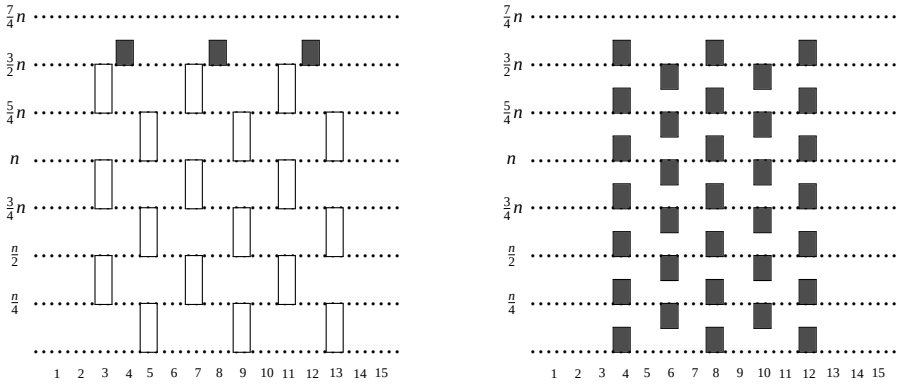


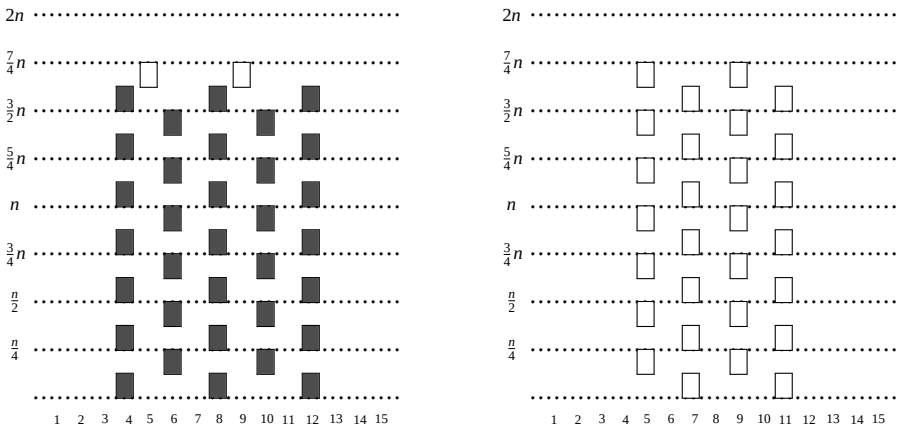
Fig. 3 Illustrating Initialization and Phase 1 for $\ell = 15$



Beginning and End of Phase 2



Beginning and End of Phase 3



Beginning and End of Phase 4

Fig. 4 Illustrating Phases 2, 3, and 4 for $\ell = 15$