Greedy online colouring with buffering

Wojciech Kordecki
Department of Computer Science
Faculty of Technical and Economic Science
The Witelon State University of Applied Sciences in Legnica
e-mail: wojciech.kordecki@pwsz-legnica.eu

Anna Łyczkowska-Hańćkowiak
Faculty of Informatics and Electronic Economy
Poznań University of Economics
e-mail: anna.lyczkowska-hanckowiak@ae.poznan.pl

January 5, 2016

Abstract

We consider the problem of online graph colouring. Whenever a node is requested, a colour must be assigned to the node, and this colour must be different from the colours of any of its neighbours. According to the greedy algorithm the node is coloured by the colour with the smallest possible k.

The goal is to use as few colours as possible. We propose an algorithm, where the node is coloured not immediately, but only after the collection of next requests stored in the buffer of size j. In other words, the first node in the buffer is coloured definitively taking into account all possible colourisations of the remaining nodes in the buffer. If there are r possible corrected colourings, then the one with the probability $1/r$ is chosen. The first coloured node is removed from the buffer to enable the entrance of the next request. A number of colours in a two examples of graphs: crown graphs and Kneser graphs have been analysed.

Keywords: online colouring, greedy algorithm.

2010 Mathematics Subject Classification: 05C15, 05C85.

1 Introduction

An online colouring of a graph G is the one assigned to G by colouring its vertices in some order

$$v_1, v_2 \ldots, v_n \,.$$

The colour of v_i is assigned by only looking at the subgraph of G induced by the set $\{v_1, v_2, \ldots, v_i\}$, and the assigned colour of v_i is never changed. Greedy colouring is a colouring of the vertices of a graph formed by a greedy algorithm that considers the vertices of the graph in sequence and assigns to each vertex its first available colour k. Another name used for the such an algorithm is *First Fit* one. Of course, greedy colourings do not generally use the minimum number of colours possible.

The unpublished review paper by Miller [14] contains introductory information the comprehensive form and presents the main problems considered in this paper. In [14], Miller introduces the problem formally and he defines a performance metric to evaluate the success of an online colouring algorithm. He points out that online (greedy, first fit) algorithm performs very well in the cases where the input graph belongs to a certain class of graphs.

Nevertheless, in many cases such an algorithm works very badly. The best known example of such a graph is a crown graph (see [14]). The algorithm with buffering presented in Section 2 essentially improves the effectiveness of colouring in the worst case, even for the buffer of a very small size.

Let $\mathcal{A}$ be an online algorithm used for colouring the graph G. Denote

- $\chi(G)$ – chromatic number of G,
- $\chi_{\mathcal{A}}(G)$ – the maximum number of used colours for each possible ordering of the vertices (the worst-case).

The performance ratio of an online graph colouring algorithm $\mathcal{A}$ for a class of graphs $\mathcal{C}$ is defined as

$$\rho(G) = \max_{G \in \mathcal{C}} \left\{ \frac{\chi_{\mathcal{A}}(G)}{\chi(G)} \right\}. \tag{1}$$

Follow[14] , we present two theorems by Halldórsson and Szegedy: [8], [9] and [7]

Theorem 1. *The performance ratio of any deterministic online colouring algorithm is at least*

$$\frac{2n}{\ln^2 n}.$$

Theorem 2. *The expected performance ratio of any randomised online colouring algorithm is at least*

$$\frac{n}{16\ln^2 n}.$$

It is known that for any bipartite graph on n vertices and any deterministic algorithm at least

$$1.13747 \cdot \log_2 n - 0.49887$$

colours are needed: Bianchi et al. [3].

Lovász, Saks, Trotter [13] prove (see also Kierstead and Trotter [12], Bianchi et al. [3]):

Theorem 3. *For any bipartite graph on n vertices there exists an online algorithm using at most $2\log_2^* n\,(o(1)+1)$ colours.*

Binary iterated logarithm $\log_2^*$ is the number of times the logarithm function must be iteratively applied before the result is less or equal to 1, i.e. $\log_2^* n = k$ where k is the smallest number for which k times iterated logarithm of n is at most 1:

$$\underbrace{\log_2 \ldots \log_2 n}_{k \text{ times}} \leq 1.$$

It is essential to know that arriving vertices are from the bipartite graph.

Colouring online has many real applications. See for example Bartal et al. [2] and Zang et al. [15]. The recent articles, e.g. Bianchi et al. [3] and Christ et al. [5] present an another look at the problem of optimising a number of colours using the so called "bit advice". In [5] one can also find an interesting application of this method to the cellular networks.

The next step to the reality is allowing that the vertices can be not only coloured but they can also be discoloured (see Borowiecki and Sidorowicz [4]). Dynamic graph colourings can be naturally applied in system modeling, e.g. for scheduling threads of parallel programs, time sharing in wireless networks, session scheduling in high-speed LANs, channel assignment in WDM optical networks as well as traffic scheduling.

In this paper we focus our attention on the case when we now know not only the present arriving vertex but we also know in advance the vertices which will arrive in the next several moments. In Section 2 we present an algorithm and analyse two particular known classes of graphs: crown graphs and Kneser graphs. In Section 4 we present some numerical results obtained by computer simulations.

2 Colouring with buffering

The problem of online colouring with buffering is known as *lookahead* and has been considered in the case of d-inductive graphs in Irani [10] (the review in Miller [14]), also Halldórsson [8] for hypergraphs.

The *d-inductive graph* is a graph with a numbered sequence of vertices in such a way that every vertex is joined with the vertex with a maximal number by at most d edges. Irani showed that the greedy algorithm uses $O(d \log n)$ colours on G if G belongs to the class of d-inductive graphs [10]. Thus the performance ratio of the greedy algorithm on chordal and planar graphs is bounded above by $O(\log n)$.

Algorithm 4. (online colouring with buffering).
Let $B \subset V$ be the buffer of size b.

1. Fix a maximal size of the buffer B as $b \geq 1$.
2. Let $V_c = \emptyset$ be the set of already coloured vertices and set $B = \emptyset$.
3. Colour the first vertex by the colour 1 and move it to V_c.
4. Fill the buffer by subsequent vertices as a queue FIFO until the buffer is full.
5. If the buffer is full, colour the vertices in the queue properly (including V_c) using colours of the minimal values.
6. Among all possible colourings of the buffer choose only such ones whose subsequent colours from the biggest one to the smallest one are minimal.
7. If such possible colourings are r, choose one with the probability $1/r$.

8. Colourings of all the vertices in the buffer except the first vertex is temporary. At the moment when the next vertex arrives, move the first one to V_c and repeat the procedure of colouring.

3 Analysis of special cases

3.1 Crown graph

Definition. A crown graph $\mathcal{C}_n = (V, E)$ on $2n$, vertices is an undirected graph with two sets of vertices, $V = V_1 \cup V_2$ with an edge from $v_{1,i}$ to $v_{2,j}$ whenever $i \neq j$. The crown graph can be viewed as a complete bipartite graph from which the edges of a perfect matching have been removed.

$$V_k = \{v_{k,1}, v_{k,2}, \ldots, v_{k,n}\}$$

and

$$(u, w) \in E \iff u = v_{1,i}, w = v_{2,j}, i \neq j.$$

Crown graphs can be used to show that greedy colouring algorithms behave badly in the worst case: if the vertices of a crown graph are presented to the algorithm in the order u_0, v_0, u_1, v_1, etc., then a greedy colouring uses n colours, whereas the optimal number of colours is two. This construction is attributed to Johnson [11]; crown graphs are sometimes called Johnsons graphs with notation J_n. Fürer [6] uses crown graphs as part of a construction showing hardness of approximation of colouring problems.

Let us denote

$$V_1 = \{v_{1,1}, v_{1,2}, \ldots, v_{1,n}\}$$
$$V_2 = \{v_{2,1}, v_{2,2}, \ldots, v_{2,n}\}$$

Let $\mathcal{C}_n$ have an linear order if

$$V = (v_{1,1}, v_{1,2}, \ldots, v_{1,n}, v_{2,1}, v_{2,2}, \ldots, v_{2,n}).$$

and $\mathcal{C}_n$ have an alternate order if

$$V = (v_{1,1}, v_{2,1}, v_{1,2}, v_{2,2}, \ldots, v_{1,n}, v_{2,n}).$$

The following theorems show, how the size of the buffer affects the performance ratio.

Theorem 5. *If $\mathcal{C}_n$, $n \geq 2$ has the alternate order, then for Algorithm 4 with $b = 2$ we have*

$$\mathbf{E}C_n = 3 - \frac{1}{2^n}\,. \tag{2}$$

Proof. At the every level a number of used colours is increased by 1, if the right and left vertex have the same colour. Such a situation can occur with the probability 1/2 under condition that at every lower level the left and right vertices obtained the same colours. If at the left and right vertices at the lower level have the different colours then the number of colours does not increase. Therefore

$$\Pr\left(C_n = k\right) = \begin{cases} \frac{1}{2^{k-1}} & \text{dla } 1 < k < n, \\ \frac{1}{2^{k-2}} & \text{dla } k = n. \end{cases} \tag{3}$$

Hence

$$\mathbf{E}C = \sum_{k=2}^{n-1} \frac{k}{2^{k-1}} + \frac{n}{2^{n-2}} = 3 - \frac{1}{2^n}\,,$$

which proved the formula (2). □

Theorem 6. *If $\mathcal{C}_n$, $n \geq 2$ has the alternate order, then for Algorithm 4 with $b = 2$ and $m < n$ we have*

$$\Pr\left(C_n \geq m\right) = \frac{1}{2^{m-2}}\,. \tag{4}$$

Proof. Formula (4) follows immediately from (3) in the proof of Theorem 5

$$\Pr\left(C_n \geq m\right) = \sum_{k=m}^{n-1} \frac{1}{2^{k-1}} + \frac{1}{2^{n-2}}$$

Since

$$\sum_{k=m}^{n-1} \frac{1}{2^{k-1}} = \frac{1}{2^{m-2}} \left(1 - \frac{1}{2^{n-m}}\right),$$

$$\sum_{k=m}^{n-1} \frac{1}{2^{k-1}} + \frac{1}{2^{n-2}} = \frac{1}{2^{m-2}}\,.$$

□

Property 7. *Buffer of the size $b = 3$ does not decrease the number of colours relatively to the buffer of the size $b = 2$ for $\mathcal{C}_n$ with an alternate order.*

Proof. The fact that we know two next vertices in a crown graph with an alternate order does not give any additional information, because the last arrived vertex and the next third are always not joined. □

Property 8. *If $\mathcal{C}_n$, $n \geq 2$, has the alternate order then for Algorithm 4 with the buffer $b = 4$ the number of used colours is always equal to 2.*

Proof. It easy to observe that the four subsequent vertices in the crown graph with the alternate order give full information that the vertices in the buffer form a bipartite graph. □

Note that if $b = 2$ it may occur that Algorithm 4 give the worse colourisation than in the case $b = 1$.

Example 9. In Figure 1 the labels of vertices have the form $\mathsf{n}{:}\mathsf{L}$, where n is the number of a subsequent arriving vertex and the letter L denotes the colour used by Algorithm 4. Vertices are coloured by colours A, B, C, D.

In Figure 1 the difference in the colouring process with $b = 2$ in comparison with the case $b = 1$ is such that the second and the third vertex have to obtain different colours. If the colour B was chosen (with probability $1/2$) for the second vertex then the third vertex has to obtain the colour A. A a result the Algorithm 4 has to give colours C and D for the last four vertices.

3.2 Kneser graphs

The vertices of $\mathcal{K}_{n,k}$ are all the k-element subsets of $\{1, 2, \ldots, n\}$, and an edge joins vertices S and T if and only if $S \cap T = \emptyset$. Such graphs were introduced by J. Kneser in 1955 – see [1], Section 38, p. 251. Kneser conjectured that $\chi(\mathcal{K}_{n,k}) = n - 2k + 2$ for $n \geq 2$. This conjecture is proved by Lovász and with subsequent simpler proofs by Bàràny and Matoušek – see [1]. The class of Kneser graphs contains many familiar classes of graphs.

- If $k > n/2$, then $\mathcal{K}_{n,k}$ is the empty graph.
- If $k = 1$, then $\mathcal{K}_{n,k} = K_n$, the complete graph on n vertices.
- $\mathcal{K}_{5,2}$ is the Petersen graph.

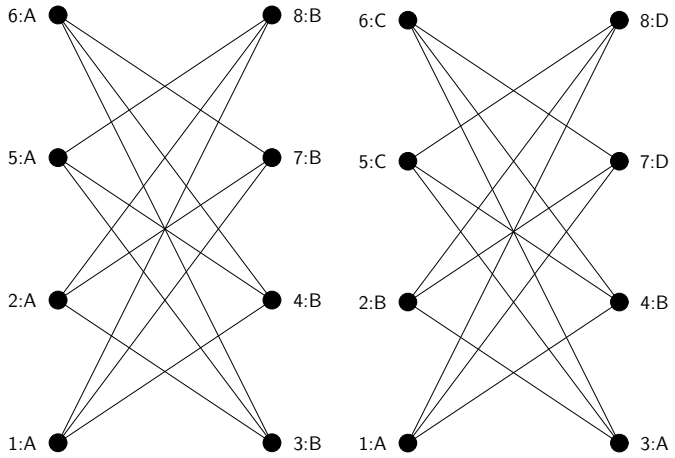

Figure 1: Colouring online for the crown graphs $\mathcal{C}_4$ with buffering: $b = 1$ (left) and $b = 2$ (right)

Miller in [14] looks at the class of Kneser graphs as an interesting one and still unconsidered.

At first let us consider the simplest example, i.e. Petersen graph, using the greedy algorithm with the buffer of size $b = 1$ and $b = 2$. Let us assume that vertices of Petersen graph are numbered in the lexicographic order:

$$v_1 = \{1, 2\}, v_2 = \{1, 3\}, \ldots, v_{10} = \{4, 5\}. \tag{5}$$

In the following example we point out that a buffering can both decrease and increase the necessary number of colours. However, as we point out through simulations in Section 4, the average number colours used by our algorithm with the buffer of size $b = 2$ is a bit smaller than the average number colours given with the buffer of size $b = 1$, i.e. without buffering. Therefore we can formulate the following problem.

Problem 10. *Determine the smallest b that*

$$\frac{\mathbf{E}C_{n,k}^{(2)}}{n - 2k + 2} - \frac{\mathbf{E}C_{n,k}^{(b)}}{n - 2k + 2} > \delta \tag{6}$$

for some fixed δ.

Example 11. In Figures 2 and 3 the labels of vertices have the form v-n:L, where v is the number of vertex in Petersen graph as in Equation (5), n is

the number of a subsequent arriving vertex and a letter L denotes the colour used by this algorithm. Let us colour the vertices by colours A, B, C, D.

Assume that the order of arriving vertices is $8, 1, 5, 7, 6, 2, 10, 4, 3, 9$. In Figure 2 the difference in the colouring process with $b = 2$ in comparison with the case $b = 1$ is such that if the present vertex is 6, we also know that the next vertex is 2. Since these vertices have to obtain different colours, we have two possibilities according to point 5 of Algorithm 4:

1. colour $(5) = A$, colour $(2) = C$ (as in the case $b = 1$),
2. colour $(5) = B$, colour $(2) = A$.

According to point 6 we choose the second possibility. Therefore, using the buffer of size 2, we can paint Petersen graph using three instead four colours.

Using the buffer of the size two at least, we do not always obtain a better result. Let as assume that the order of arriving vertices is $9, 7, 5, 8, 1, 6, 3, 2, 4, 10$. In the case colouring with the buffer of size $b = 1$ gives three colours. In the case $b = 2$, if we colour the fourth vertex v_8 and we know that the next vertex is v_1, then we have to colour these vertices by B and C. If we decide (with probability $1/2$) that colour $(v_8) = C$, then finally we must use four colours to paint Petersen graph instead of three colours. Such a case is presented on Figure 3.

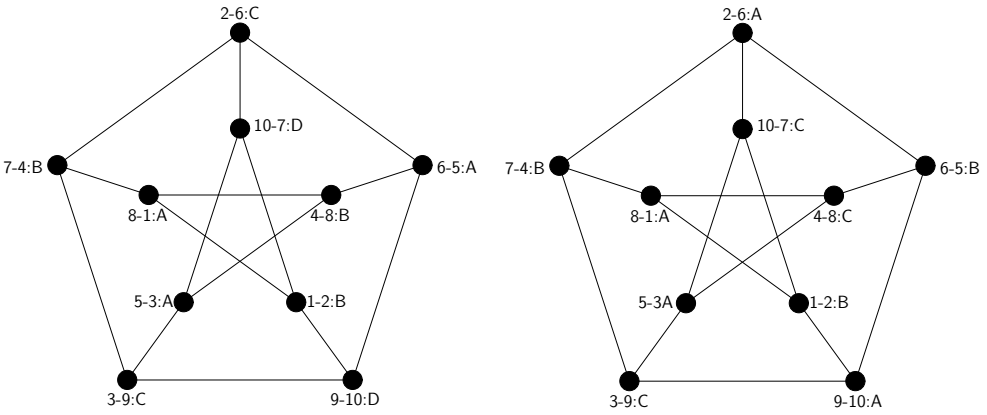

Figure 2: Colouring online for Petersen graphs with buffering: $b = 1$ (left) and $b = 2$ (right) and ordering $8, 1, 5, 7, 6, 2, 10, 4, 3, 9$

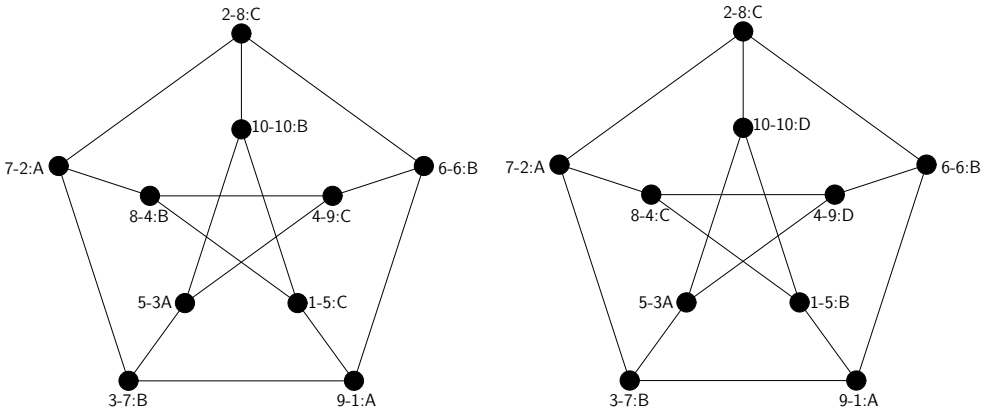

Figure 3: Colouring online for Petersen graphs with buffering: $b = 1$ (left) and $b = 2$ (right) and ordering $9, 7, 5, 8, 1, 6, 3, 2, 4, 10$

4 Simulations

Simulations for crown graphs and Kneser graphs used 20 000 repetitions[1]. Taking into account of theorems 5–8 we carried out the simulations for crown graphs only for $b \leq 2$. In the both considered cases the maximal number of vertices is equal to 200.

Table 1: The average number of colours: result of simulations for random order of $\mathcal{C}_n$

n	$b = 2$	$b = 1$
4	2.24	2.32
6	2.15	2.20
10	2.08	2.12
20	2.04	2.05
50	2.02	2.02
100	2.01	2.01

For Kneser graphs, simulations were carried out for $5 \leq n \leq 10$, $k \geq 2$ and $b \leq 2$. The result of simulations for $b = 1$ and $b = 2$ is given in Table 2.

[1]The computer program was written in Pascal using Lazarus environment and the standard random number generator.

Simulations for $b = 2$ give smaller but almost the same results as in the case when $b = 1$.

Table 2: The average number of colours: result of simulations for random order of $\mathcal{K}_{n,k}$

	$k=2$		$k=3$		$k=4$	
n \ b	1	2	1	2	1	2
5	3.13	3.10				
6	4.28	4.23				
7	5.44	5.37	3.93	3.918		
8	6.58	6.51	5.69	5.65		
9	7.70	7.64	7.35	7.30	4.89	4.88
10	8.81	8.74	8.93	8.88	7.52	7.49

References

[1] M. Aigner and G. M. Ziegler. *Proofs from THE BOOK*. Springer, Berlin Heidelberg, 2010.

[2] Y. Bartal, A. Fiat, and S. Leonardi. Lower bounds for on-line graph problems with application to on-line circuit and optical routing. *SIAM J. Comput.*, 36:354–393, 2006.

[3] M. P. Bianchi, H.-J. Böckenhauer, J. Hromkovič, and L. Keller. Online coloring of bipartite graphs with and without advice. *Algorithmica*, 70:92–111, 2014.

[4] P. Borowiecki and E. Sidorowicz. Dynamic coloring of graphs. *Fund. Inform.*, 114:105–128, 2012.

[5] M. G. Christ, L. M. Favrholdt, and K. S. Larsen. Online multi-coloring with advice. In O. S. E. Bampis, editor, *Lecture Notes in Computer Science*, volume 8952, pages 83–94. Springer, 2015.

[6] M. Fürer. Improved hardness results for approximating the chromatic number. In *Proc. 36th IEEE Symp. Foundations of Computer Science (FOCS '95)*, pages 414–421, 1995.

[7] M. M. Halldórsson. Online coloring known graphs. *Electron. J. Combin.*, 7:1–9, 2000.

[8] M. M. Halldórsson. Online coloring of hypergraphs, 2010. `www.ru.is/faculty/mmh/papers/onhyper-final.pdf`.

[9] M. M. Halldórsson and M. Szegedy. Lower bounds for on-line graph coloring. *Theoretical Computer Science*, 130:163–174, 1994.

[10] S. Irani. Coloring inductive graphs on-line. *Algoritmica*, 11:53–72, 1994.

[11] D. S. Johnson. Worst-case behavior of graph coloring algorithms. In *Proc. 5th Southeastern Conf. on Combinatorics, Graph Theory, and Computing, Utilitas Mathematicae*, pages 513–527, Winnipeg, 1974.

[12] H. A. Kierstead and W. T. Trotter. On-line graph coloring. *DIMACS*, 7:85–92, 1992.

[13] L. Lovász, M. Saks, and W. T. Trotter. An on-line graph coloring algorithm with sublinear performance ratio. *Discrete Math.*, 75:319–325, 1989.

[14] A. Miller. Online graph colouring, 2004. `http://www.cumc.math.ca/2005/papers/miller.pdf`.

[15] H. Zang, J. P. Jue, and B. Mukherjee. A review of routing and wavelength assignment approaches for wavelength-routed optical wdm networks. *SPIE Optical Networks Magazine*, 1:47–60, 2000.