Discrete Mathematics and Theoretical Computer Science DMTCS vol. **15:3**, 2013, 127–138

1-local 33/24-competitive Algorithm for Multicoloring Hexagonal Graphs

Rafał Witkowski[1†] and Janez Žerovnik[2‡]

[1] *Adam Mickiewicz University, Faculty of Mathematics and Computer Science, Poznań, Poland*
[2] *University of Ljubljana, Faculty of mechanical engineering, Ljubljana, Slovenia*

received 11th May 2012, revised 21st May 2013, accepted 17th Nov. 2013.

In the frequency allocation problem, we are given a cellular telephone network whose geographical coverage area is divided into cells, where phone calls are serviced by assigned frequencies, so that none of the pairs of calls emanating from the same or neighboring cells is assigned the same frequency. The problem is to use the frequencies efficiently, i.e. minimize the span of frequencies used. The frequency allocation problem can be regarded as a multicoloring problem on a weighted hexagonal graph, where each vertex knows its position in the graph. We present a 1-local 33/24-competitive distributed algorithm for multicoloring a hexagonal graph, thereby improving the previous 1-local 7/5-competitive algorithm.

Keywords: approximation algorithm, multicoloring, triangular grid, channel assignment

1 Introduction

A vertex weighted graph is a triple $G(E, V, d)$, where V is the set of vertices, E is the set of edges and $d : V \to \mathbb{N}$ is a weight function assigning (non-negative) integer demands to vertices of G. A *proper multicoloring* of G is a mapping f from $V(G)$ to subsets of integers such that $|f(v)| = d(v)$ for any vertex $v \in V(G)$ and $f(v) \cap f(u) = \emptyset$ for any pair of adjacent vertices u and v in the weighted graph G with vertex weights $d(v)$. The minimum number of colors needed for a proper multicoloring of G, $\chi_m(G)$, is called the *multichromatic number*. Another invariant of interest in this context is the *(weighted) clique number*, $\omega_m(G)$, see formal definition in Section 2. It is well known that $\chi_m(G) \geq \omega_m(G)$ and that it is NP-complete problem to decide whether $\chi_m(G) = \omega_m(G)$ [6]. A lot of work has been done to find approximation algorithms that imply upper bounds for the multichromatic number of hexagonal graphs.

Earlier studies of multicoloring problem on hexagonal graphs were motivated by a fundamental problem concerning cellular networks where sets of frequencies (colors) are assigned to transmitters (vertices) in order to avoid unacceptable interferences [2, 5, 6, 7, 8]. The number of frequencies that are demanded at a transmitter may vary between transmitters. In a usual cellular model, the transmitters are the centers

[†]Email: `rmiw@amu.edu.pl`. Supported by grant N206 1842 33 for years 2011-2014.
[‡]Email: `janez.zerovnik@fs.uni-lj.si`. Supported in part by ARRS, the research agency of Slovenia.

of hexagonal cells and the corresponding adjacency graph is an induced subgraph of the infinite triangular lattice. An integer $d(v)$ is assigned to each vertex of the triangular lattice and will be called the *demand* (or *weight*) of the vertex v. The vertex weighted graph induced by the subset of the triangular lattice of vertices of positive demand is called a (vertex weighted) *hexagonal graph*.

An assumption that naturally arises in hexagonal graphs which model cellular networks is that each vertex is aware of its location. In distributed graph algorithms, the property of ”locality” plays special roles. In this paper, so called k-local algorithms for multicoloring hexagonal graphs are studied. An algorithm is *k-local* if the computation at any vertex v uses the information about the demands of vertices at distance at most k from v only. In other words, every vertex knows its position and can use the information within its k-neighborhood, i.e. in this case it knows the demands of vertices at graph distance at most k.

A framework for studying distributed online assignment in cellular networks was developed in [5]. A distinction between *online* and *offline* algorithms was introduced and the definition of *p-competitive algorithm* was given as well. In the offline version of the problem the demands are fixed and known in advance while in the online version the demands may change over time, motivated by the fact that the number of calls within the cell in the network changes. Here we will only consider the offline version as we develop an algorithm that multicolors a hexagonal graph with known demands at vertices. Online version is only mentioned in the corollary that follows directly using result of [5]. Finally, an algorithm is p-competitive if it uses at most p times as many colors (frequencies) overall as the optimal offline algorithm would. In the same paper [5], a 3/2-competitive 1-local, 17/12-competitive 2-local and 4/3-competitive 4-local algorithms were outlined. Later, a 4/3-competitive 2-local algorithm was developed in [11]. The best ratio for 1-local case was first improved to 13/9 [1], and later to 17/12 [14] and to 7/5 [15]. In this paper, we develop a new 1-local algorithm which uses no more than $\frac{33}{24}\omega_m(G)+O(1)$ colors, implying the existence of a 33/24-competitive algorithm.

It may be worth mentioning that the approximation bound for multicoloring algorithms on hexagonal graphs $\chi_m(G) \leq (4/3)\omega_m(G) + O(1)$ [6, 8, 11] is still the best known, both for distributed and non-distributed models of computation. Considering that, one can naturally take 4/3 as (maybe too ambitious) goal ratio for 1-local algorithms. With this assumption, our improvement from 7/5 to 33/24 can be calculated by the following formula

$$\frac{\frac{7}{5}-\frac{33}{24}}{\frac{7}{5}-\frac{4}{3}} = \frac{3}{8} = 37.5\%$$

which is a considerable. Evaluating the previous improvements from 3/2 to 13/9 to 17/12 to 7/5 by the same formula gives 33.3%, 25% and 20%, respectively. From this point of view this is the biggest improvement.

Our algorithm substantially differs from the algorithms in [1] and [11] which are composed of two stages. In these algorithms, a triangle-free hexagonal graph with weighted clique number no larger than $\lceil \omega_m(G)/3 \rceil$ is constructed from G at the first stage, while at the second stage an algorithm for multicoloring a triangle-free hexagonal graph is used (see [1, 4, 12, 16]). Our improvement is based on the idea of borrowing some colors used in the first stage and using them for the demands of the second stage (see [15]). This in particular implies that the second stage of our algorithm cannot be applied as a stand-alone algorithm for multicoloring arbitrary triangle-free hexagonal graphs.

The main result of this paper is the following:

Theorem 1.1 *There is a 1-local distributed approximation algorithm for multicoloring hexagonal graphs which uses at most $\frac{33}{24}\omega_m(G) + O(1)$ colors. Time complexity of the algorithm at each vertex is constant.*

In [5] it was proved that a k-local c-approximate offline algorithm can be easily converted to a k-local c-competitive online algorithm, so we have:

Corollary 1.1 *There is a 1-local 33/24-competitive online algorithm for multicoloring hexagonal graphs.*

The paper is organized as follows: in the next section, we formally define some basic terminology. In Section 3, we present an overview of the algorithm, while in Section 4 we provide a proof of Theorem 1.1.

2 Basic definition and useful facts

Following the notation from [6], the vertices of the triangular lattice T can be described as follows: the position of each vertex is an integer linear combination $x\vec{p}+y\vec{q}$ of two vectors $\vec{p} = (1, 0)$ and $\vec{q} = (\frac{1}{2}, \frac{\sqrt{3}}{2})$. Thus vertices of the triangular lattice may be identified with pairs (x, y) of integers. Given the vertex v, we will refer to its coordinates as $x(v)$ and $y(v)$. Two vertices are adjacent when the Euclidean distance between them is one. Therefore each vertex (x, y) has six neighbors: $(x-1, y), (x-1, y+1), (x, y+1), (x+1, y), (x+1, y-1), (x, y-1)$. For simplicity we refer to the neighbors as: *left*, *up-left*, *up-right*, *right*, *down-right* and *down-left*.

Assume that we are given a weight function $d : V \to \{0, 1, 2, \ldots\}$ on vertices of triangular lattice. We define a *weighted hexagonal graph* $G = (V, E, d)$ as an induced subgraph by vertices of positive demand on the triangular lattice, (see Figure 1). Sometimes we want to consider (unweighted) hexagonal graphs $G = (V, E)$ that can be defined as induced by subsets of vertices of the triangular lattice. In both cases, we can assume that every vertex of hexagonal graph G knows its coordinates (x, y) in the triangular lattice.

Let us recall that a *proper multicoloring* of $G = (V, E, d)$ is a mapping f from $V(G)$ to subsets of integers such that $|f(v)| = d(v)$ for any vertex $v \in V(G)$ and $f(v) \cap f(u) = \emptyset$ for any pair of adjacent vertices u and v in the weighted graph G. The minimum number of colors needed for a proper multicoloring of G, $\chi_m(G)$, is called the *multichromatic number*. The *(weighted) clique number*, $\omega_m(G)$, is the maximal clique weight on G, where the weight of a clique is the sum of demands on its vertices.

Notice that in any weighted hexagonal graph G, a subgraph of the triangular lattice T induced by vertices with positive demands $d(v)$, the only cliques are triangles, edges and isolated vertices. Recall that by definition all vertices of T which are not in G must have demand $d(v) = 0$. Therefore, the weighted clique number of G can be computed as follows:

$$\omega_m(G) = \max\{d(u) + d(v) + d(t) : \{u, v, t\} \in \tau(T)\},$$

where $\tau(T)$ is the set of all triangles of T i.e., the weighted clique number is the maximum weight over weights of all triangles, edges and weights of isolated vertices.

There exists an obvious 3-coloring of the infinite triangular lattice, which gives partition of the vertex set of any hexagonal graph into three independent sets. Let us denote a color of any vertex v in this 3-coloring by $bc(v)$ and call it a *base color* (for simplicity we will use *red*, *green* and *blue* as the base colors and their arrangement is given in Figure 1), i.e. $bc(v) \in \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}$.

An induced subgraph of the triangular lattice without a 3-clique is called a *triangle-free hexagonal graph*. A *corner* in a triangle-free hexagonal graph is a vertex which has at least two neighbors and none

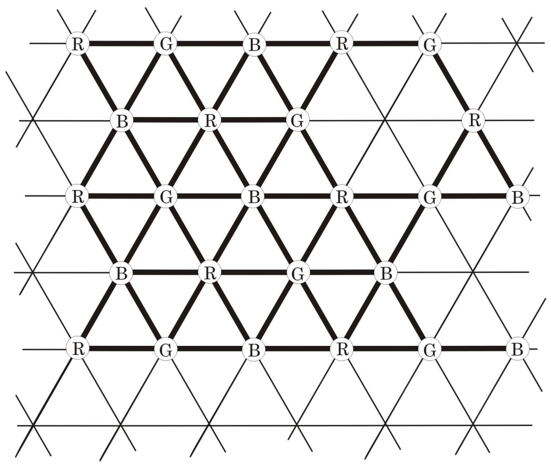

Fig. 1: An example of a hexagonal graph (with base coloring)

of which are at angle π. A vertex is a *right corner* if it has an up-right or a down-right neighbor, otherwise it is a *left corner* (see Figure 2). A vertex which is not a corner is called a *non-corner*.

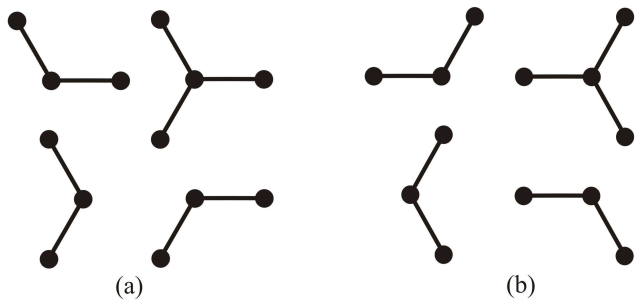

Fig. 2: All possibilities for: (a) - left corners, (b) - right corners

Lemma 2.1 *[13] Consider a 3-coloring* $(\mathrm{R}, \mathrm{G}, \mathrm{B})$ *of the triangular lattice. Every odd cycle of the triangle-free hexagonal graph* G *contains at least one non-corner vertex of every color.*

As the elegant proof of Sudeep and Vishwanathan [13] is very short, we recall it for completeness and for future reference.

Proof: Assume without loss of generality that there exists an odd cycle in the graph which does not have a non-corner vertex colored red. Notice that in the 3-coloring of the triangular lattice, a corner has all its neighbors colored by the same color (they are at the angle $2\pi/3$). Hence, if all neighbors of a red colored corner are blue, we can recolor this corner by green color and vice-versa. That gives a valid 2-coloring of an odd cycle, a contradiction. □

Notice that removing all non-corners of one color from a hexagonal graph G results in a bipartite graph. For any weighted bipartite graph H, $\chi_m(H) = \omega_m(H)$ (see [8]), and it can be optimally multicolored by the following 1-local procedure.

Procedure 2.1 *Let $H = (V, E, d)$ be a weighted bipartite graph and let a bipartition $V = V' \cup V''$ be given. We get an optimal multicoloring of H if we assign to each vertex $v \in V'$ a set of colors $\{1, 2, \ldots, d(v)\}$, and with each vertex $v \in V''$ we associate a set of colors $\{m(v)+1, m(v)+2, \ldots, m(v)+d(v)\}$, where $m(v) = \max\{d(u) : uv \in E\}$.*

Proof: The procedure is 1-local, because each vertex v uses only its weight function $d(v)$ or calculates value $m(v)$ which is taken from its neighbors. From definition of $m(v)$ we can clearly see that no conflict occurs in this multicoloring. Since in a bipartite graph the only cliques are edges and isolated vertices, the largest number of color used is

$$\max\{\max\{d(v) : v \in V'\}, \max\{d(v) + m(v) : v \in V''\}\} =$$

$$= \max\{\max\{d(v) : v \in V'\}, \max\{d(v) + \max\{d(u) : uv \in E\} : v \in V''\}\} =$$

$$= \max\{\max\{d(v) : v \in V'\}, \max\{d(v) + d(u) : uv \in E\}\} = \omega_m(G).$$

□

For each vertex $v \in V(G)$, define *base function* κ as

$$\kappa(v) = \max\{a(v, u, t) : \{v, u, t\} \in \tau(T)\},$$

where

$$a(u, v, t) = \left\lceil \frac{d(u) + d(v) + d(t)}{3} \right\rceil,$$

is the average weight of the triangle $\{u, v, t\} \in \tau(T)$. We can naturally define $a(u, v, t)$ for any triple of vertices that forms a triangle in the lattice, including the cases when $d(.) = 0$ for some of the vertices.

Clearly, $\kappa(v)$ can be computed using 1-local information only, and the following fact holds.

Fact 2.1 *For each $v \in V(G)$,*

$$\kappa(v) \leq \left\lceil \frac{\omega_m(G)}{3} \right\rceil$$

We call vertex v *heavy* if $d(v) > \kappa(v)$, otherwise we call it *light*. If $d(v) > 2\kappa(v)$ we say that the vertex v is *very heavy*.

To color vertices of G, we use colors from an appropriate *palette*. For a given color c, its palette is defined as a set of pairs $\{(c, i)\}_{i \in \mathbb{N}}$. A palette is called a *base color palette* if $c \in \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}$ is one of the base colors, and it is called *additional color palette* if $c \notin \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}$. In algorithm we will use 5 additional color palletes of different size (without explicitly naming the colors).

An important notion used in the algorithm will be the parity of the coordinates. Let $p(x) = x \bmod 2$ denote the parity function, which we will use for coordinates.

In our 1-local model of computation we assume that each vertex knows its coordinates as well as its own demand (weight) and demands of all its neighbors. In the next section, we will demonstrate how each vertex can color itself properly in constant time, using this information only.

3 Algorithm

Our algorithm consists of three main phases. In the first phase (Step 1 and 2 below) vertices take $\kappa(v)$ colors from its base color palette, so use no more than $\omega_m(G)$ colors. After this phase, all light vertices in G are fully colored, i.e. every light vertex $v \in V(G)$ already received all needed $d(v)$ colors. The vertices that are heavy but not very heavy induce a triangle-free hexagonal graph with weighted clique number not exceeding $\lceil \omega_m(G)/3 \rceil$. Very heavy vertices in G are isolated in the remaining graph and therefore they can easily be fully colored. However, they have to be considered separately (Step 2).

In the second phase we construct two types of bipartite graphs. First (Step 3 below) we construct three bipartite graphs by removing noncorners of each color type and use Procedure 2.1 for optimally satisfying 1/8 of demands in the remaining graph. Next (Step 4 below), we divide the vertices in the triangular lattice into two sets of well separated parallel lines. Finally (Step 5 and 6 below), by using this partition, we construct two bipartite graphs and use Procedure 2.1 for optimally satisfying 3/8 of demands of all vertices except some corners which have to be treated in a separate way – by using free colors from the base color palettes.

More precisely, our algorithm is the following:

Algorithm

Input: Weighted hexagonal graph $G = (V, E, d)$, where all vertices know their position in the graph.

Output: A proper multicoloring of G, using at most $33/24 \cdot \omega_m(G) + O(1)$ colors.

Step 0 For each vertex $v \in V$ compute its base color $bc(v)$

$$bc(v) = \begin{cases} \text{R} & \text{if} \quad (x(v) + 2y(v)) \bmod 3 = 0 \\ \text{G} & \text{if} \quad (x(v) + 2y(v)) \bmod 3 = 1 \\ \text{B} & \text{if} \quad (x(v) + 2y(v)) \bmod 3 = 2 \end{cases},$$

and its base function value

$$\kappa(v) = \max\left\{ \left\lceil \frac{d(u) + d(v) + d(t)}{3} \right\rceil : \{v, u, t\} \in \tau(T) \right\}.$$

Step 1 For each vertex $v \in V$ assign to v the first $\min\{\kappa(v), d(v)\}$ colors from its base color palette. Construct a new weighted triangle-free hexagonal graph $G_1 = (V_1, E_1, d_1)$ where $d_1(v) = \max\{d(v) - \kappa(v), 0\}$, $V_1 \subseteq V$ is the set of vertices with $d_1(v) > 0$ (heavy vertices in G) and $E_1 \subseteq E$ is the set of all edges in G with both endpoints from V_1 (G_1 is induced by V_1).

Step 2 For each vertex $v \in V_1$ with $d_1(v) > \kappa(v)$ (very heavy vertices in G) assign the first unused $\kappa(v)$ colors of the base color palettes of its neighbors in T. Construct a new graph $G_2 = (V_2, E_2, d_2)$ where $d_2(v)$ is the difference between $d_1(v)$ and the number of colors assigned in this step, $V_2 \subseteq V_1$ is the set of vertices with $d_2(v) > 0$ and $E_2 \subseteq E_1$ is the set of all edges in G_1 with both endpoints from V_2 (G_2 is induced by V_2).

Step 3 For the set of red noncorners do the following:

- Construct the bipartite graph $G_{n\mathrm{R}} = (V_{n\mathrm{R}}, E_{n\mathrm{R}}, d_{n\mathrm{R}})$ where $d_{n\mathrm{R}}(v) = d_2(v)/8$, $V_{n\mathrm{R}} \subseteq V_2$ is the set of vertices from G_2 without red noncorners, and $E_{n\mathrm{R}} \subseteq E_2$ is the set of all edges in $G_{n\mathrm{R}}$ with both endpoints in $V_{n\mathrm{R}}$ ($E_{n\mathrm{R}}$ is induced by $V_{n\mathrm{R}}$)
- By the proof of Lemma 2.1 ($V'_{n\mathrm{R}}$, $V''_{n\mathrm{R}}$) is a bipartition of $G_{n\mathrm{R}}$ where:
  - $V'_{n\mathrm{R}}$ is the set of blue vertices and red corners with all neighbors green,
  - $V''_{n\mathrm{R}}$ is the set of green vertices and red corners with all neighbors blue.
- Use the bipartition $(V'_{n\mathrm{R}}, V''_{n\mathrm{R}})$ and apply Procedure 2.1 for the weighted graph $G_{n\mathrm{R}}$ by using colors from new additional color palette.

Do analogous for graphs $G_{n\mathrm{B}}$ (G_2 without blue noncorners) and $G_{n\mathrm{G}}$ (G_2 without green noncorners).

Step 4 Divide vertices of the infinite triangular lattice into two sets (two sets of horizontal lines):

$$\begin{aligned} V_e &= \{v \in \tau(T) : p(x(v)) = p(y(v))\} \\ V_o &= \{v \in \tau(T) : p(x(v)) \neq p(y(v))\} \end{aligned}$$

Step 5 Using the sets V_e and V_o do as follows:

5a
- Construct a bipartite graph $\tilde{G} = (\tilde{V}, \tilde{E}, \tilde{d})$ where $\tilde{d}(v) = \frac{3d_2(v)}{8}$, $\tilde{V} \subseteq V_2$ consists of vertices from V_e and noncorners from V_o, and $\tilde{E} \subseteq E_2$ is the set of all edges in G_2 with both endpoints in $\tilde{V}$ ($\tilde{E}$ is induced by $\tilde{V}$).
- Construct the bipartition ($\tilde{V}'$,$\tilde{V}''$) of $\tilde{G}$, where
  - $*$ $\tilde{V}'$ is the set of vertices with both coordinates even ($\{v \in \tilde{V} : p(x(v)) = p(y(v)) = 0\}$), noncorners with neighbors with both coordinates odd, and noncorners u with both neighbors in V_o and $p(x(u)) = 0$.
  - $*$ $\tilde{V}''$ is the set of vertices with both coordinates odd ($\{v \in \tilde{V} : p(x(v)) = p(y(v)) = 1\}$), noncorners with neighbors with both coordinates even, and noncorners u with both neighbors in V_o and $p(x(u)) = 1$.
- Use the bipartition $(\tilde{V}', \tilde{V}'')$ and apply Procedure 2.1 for weighted graph $\tilde{G}$ by using colors from new additional color palette.

5b For each corner $v \in V_o$ assign $\frac{3d_2(v)}{8}$ colors from the free base color palettes. Left corners take unused odd colors and right corners take unused even colors.

Step 6 Switch the sets V_e and V_o and do the same as in Step 5.

4 Correctness proof

At the very beginning of the algorithm there is a 1-local communication, when each vertex finds out about the demands of all its neighbors. Later, no communication is needed. Recall that each vertex knows its position (x, y) on the triangular lattice T. Note that whenever we mention "very heavy/heavy/light vertex", we refer to the property of this vertex in the graph G, i.e. there is no reclassification in graphs $G_i, i \in 1, 2, 3$.

Regarding Step 0, there is nothing to prove.

In Step 1 each heavy vertex v in G is assigned $\kappa(v)$ colors from its base color palette, while each light vertex u is assigned $d(u)$ colors from its base color palette. Hence the remaining weight of each vertex $v \in G_1$ is

$$d_1(v) = d(v) - \kappa(v).$$

Note that $V(G_1)$ consists only of heavy vertices in G, therefore

Lemma 4.1 *G_1 is a triangle-free hexagonal graph.*

Proof: Assume that there exists a triangle $\{v, u, t\} \in G_1$, which means that $d_1(v), d_1(u), d_1(t) > 0$. Then we have:

$$\begin{aligned} d(v) + d(u) + d(t) &= d_1(v) + \kappa(v) + d_1(u) + \kappa(u) + d_1(t) + \kappa(t) \geq \\ &\geq d_1(v) + d_1(u) + d_1(t) + 3a(u, v, t) \geq \\ &\geq d_1(v) + d_1(u) + d_1(t) + d(v) + d(u) + d(t) \\ &> d(v) + d(u) + d(t) \end{aligned}$$

a contradiction. Therefore, the graph G_1 does not contain a 3-clique, so it is a triangle-free hexagonal graph. □

In Step 2, only vertices with $d_1(v) > \kappa(v)$ (very heavy vertices in G) are colored. Each very heavy vertex in G has enough unused colors in its neighborhood to be finally multicolored. If vertex v is very heavy in G then it is isolated in G_1 (all its neighbors are light in G). Otherwise, for some $\{v, u, t\} \in \tau(T)$ we would have

$$d(v) + d(u) > 2\kappa(v) + \kappa(u) \geq 3a(v, u, t) \geq d(v) + d(u),$$

a contradiction. Without loss of generality we may assume that $bc(v)$ is green. Denote by

$$D_{\mathrm{R}}(v) = \min\{\kappa(v) - d(u) : uv \in E(T), bc(u) = \mathrm{R}\},$$

$$D_{\mathrm{B}}(v) = \min\{\kappa(v) - d(u) : uv \in E(T), bc(u) = \mathrm{B}\}.$$

Obviously, $D_{\mathrm{R}}(v), D_{\mathrm{B}}(v) > 0$ for very heavy vertices v in G. Since in Step 1 each light vertex t uses exactly $d(t)$ colors from its base color palette, we have at least $D_{\mathrm{R}}(v)$ free colors from the green base color palette and at least $D_{\mathrm{B}}(v)$ free colors from the blue base color palette. If vertex v could assign those colors to itself, we would have G_2 with $\omega_m(G_2) \leq \lceil \omega_m(G)/3 \rceil$. Formally it can be proved that

Lemma 4.2 *In G_1 for every edge $vu \in E_1$ we have:*

$$d_1(v) + d_1(u) \leq \kappa(v), \quad d_1(u) + d_1(v) \leq \kappa(u),$$

Proof: Assume that v and u are heavy vertices in G and $d_1(v) + d_1(u) > \kappa(v)$. Then for some $\{v, u, t\} \in \tau(T)$ we have:

$$d(v) + d(u) = d_1(v) + \kappa(v) + d_1(u) + \kappa(u) > 2\kappa(v) + \kappa(u) \geq 3a(u, v, t) \geq d(u) + d(v),$$

again a contradiction. □

From these facts we can observe that

Fact 4.1

$$\omega_m(G_2) \leq \left\lceil \frac{\omega_m(G)}{3} \right\rceil .$$

Proof: Recall that in a hexagonal graph the only cliques are triangles, edges and isolated vertices. Since G_1 is a triangle-free hexagonal graph, G_2 also does not contain any triangle, so we have only edges and isolated vertices to check.

For each edge $vu \in E_2$, using Lemma 4.2 and Fact 2.1, we have:

$$d_2(v) + d_2(u) \leq d_1(v) + d_1(u) \leq \kappa(v) \leq \lceil \omega_m(G)/3 \rceil .$$

For each isolated vertex $v \in G_2$ we should have $d_2(v) \leq \lceil \omega_m(G)/3 \rceil$. Recall that v is very heavy, so $d_2(v) = d(v) - 2\kappa(v)$ because the vertex has received $\kappa(v)$ colors both in Step 1 and in Step 2. We claim that $d_2(v) \leq \kappa(v)$. Indeed, if $d_2(v) > \kappa(v)$, then $d(v) = d_2(v) + 2\kappa(v) > 3\kappa(v)$ contradicting the definition of $\kappa(v)$. Hence, $d_2(v) \leq \kappa(v) \leq \lceil \omega_m(G)/3 \rceil$ as needed. □

In Step 3 each vertex v has to decide whether it is a corner in G_2 or not. Only heavy neighbors of v can still exist in G_2. Unfortunately, in 1-local model v does not know which of its neighbors are heavy (and still exist in G_2) and which were light in G. Vertex v knows only where its neighbors u with $d(u) \leq a(v, u, t_1)$ and $d(u) \leq a(v, u, t_2)$, are located. We call these vertices *slight neighbors* of v. They must be light in G and as such they are fully colored in Step 1. Therefore, v knows where it cannot have neighbors in G_2 and presumes that all its neighbors which are not slight, still exist in G_2. Based on this knowledge, it can decide whether it is a corner or not. In each triangle in $\tau(T)$ containing v at least one neighbor of v is slight, so v has at least three such neighbors. If vertex v has more than four slight neighbors, then it is a non-corner. If vertex v has four slight neighbors, then the remaining two are not slight. In this case if an angle between those two is π, then v is a non-corner, otherwise it is a corner – a right corner if its down-left, up-left and right neighbors are slight, and a left corner if its down-right, up-right and left neighbors are slight. If vertex v has three slight neighbors, then it is a corner and the distinction between left and right is determined in the same way as above.

According to the proof of Lemma 2.1 it is obvious that $V'_{n\mathrm{R}}$ and $V''_{n\mathrm{R}}$ form a bipartition of $G_{n\mathrm{R}}$.

Note that, under 1-locality assumption, vertices cannot calculate the value of d' of their neighbors, which is needed in Procedure 2.1 to calculate the value $m(v) = \max\{\lceil d'(u)/8 \rceil : uv \in E'\}$. However, we can replace $d'(u)$ by $d'_v(u)$, which is the number of expected demands on vertex u in vertex v after Step 2, and take $m'(v) = \max\{\lceil d'_v(u)/8 \rceil : uv \in E'\}$. More precisely,

$$d'_v(u) = d(u) - \max\{a(u, v, t) : \{u, v, t\} \in \tau(T)\}$$

Note that $d'_v(u) \geq d'(u)$ for any $vu \in E'$. However,

Lemma 4.3 *For every $vu \in E'$ we have*

$$d'(v) + d'_v(u) \leq \kappa(v).$$

Proof: Assume that this inequality does not hold, hence $d'(v) + d'_v(u) > \kappa(v)$. Denote by

$$b(u, v) = \max\{a(u, v, t) : \{u, v, t\} \in \tau(T)\}.$$

Then for some $\{t, v, u\} \in \tau(T)$ we have:

$$d(v) + d(u) = d'(v) + \kappa(v) + d'_v(u) + b(u, v) > 2\kappa(v) + b(u, v) \geq$$
$$\geq 3a(u, v, t) \geq d(u) + d(v),$$

a contradiction. □

Hence, if we use d'_v instead of d' at each vertex from the second set of our bipartition, we formally work with a new graph $\tilde{G}'$ with a new $\omega_m(\tilde{G}')$. Because of Fact 2.1 and Lemma 4.3 we have the inequality

$$\omega_m(\tilde{G}') \leq \left\lceil \frac{\omega_m(G)}{3} \right\rceil$$

analogous to the inequality from Fact 4.1. Thus, Procedure 2.1 works and in one substep uses at most $\lceil 3\omega_m(G)/24 \rceil + 1$ colors from some new additional color palettes.

Regarding Step 4, there is nothing to prove.

In Step 5a it is easy to check that $\tilde{V}'$ and $\tilde{V}''$ gives the bipartition of $\tilde{G}$. The problem with 1-locality assumption in Procedure 2.1 can be solved as in Step 3.

In Step 5b we take the corners and use again the base color palettes. If vertex $v \in G_2$ is a corner, it means that it has three slight neighbors of the same base color. Without loss of generality, assume that $bc(v) = \mathrm{R}$ and its slight neighbors' base color is blue. Recall the function D_{B} from Step 2 – we have $D_{\mathrm{B}}(v)$ free colors from blue base color palette. We claim that

Lemma 4.4 *If v is a corner in G_2 with three slight neighbors colored blue, then $d_2(v) \leq D_{\mathrm{B}}(v)$.*

Proof: Let v be a red corner in G_2. Without loss of generality assume that t is the green vertex which is not slight neighbor of v, and u is the blue vertex which is a slight neighbor of v so that $\{u, v, t\} \in \tau(T)$ is a triangle. Then we have

$$\kappa(v) + d_2(v) + a(u, v, t) + d(u) \leq d(v) + d(t) + d(u) \leq 3a(u, v, t) \leq a(u, v, t) + 2\kappa(v)$$

and the left inequality occurs because $d_2(v) = d(v) - \kappa(v)$ and from definition of slight neighbors $d(t) \geq a(u, v, t)$. Since v is a corner, each slight neighbor of v has to belong to some triangle in $\tau(T)$ with a non slight neighbor. Hence we can repeat this argument for all slight neighbors of v. As

$$d_2(v) \leq \kappa(v) - d(u)$$

holds for any slight neighbor of v, it is true for the minimum, i.e. $d_2(v) \leq D_{\mathrm{B}}(v)$. □

Therefore, vertex v has as much as $d_2(v)$ free colors from the blue base color palette at its disposal. Since left corners take unused odd colors and right corners take unused even colors, no conflict occurs, because no two left (right) corners can be connected.

During Step 3 each noncorner participates in exactly two from the three rounds and receives $\lceil 2d_2(v)/8 \rceil$ colors while each corner participates in all three rounds and receives $\lceil 3d_2(v)/8 \rceil$ colors. During Steps 5 and 6 each vertex receives $\lceil 3d_2(v)/8 \rceil$ colors twice, so in both steps it receives $\lceil 6d_2(v)/8 \rceil$ colors. Combining all steps each noncorner receives $\lceil 8d_2(v)/8 \rceil$, hence as many as it needs, and each corner receives $\lceil 9d_2(v)/8 \rceil$, so even more than it needs. Therefore, at the end, all of the demands are satisfied.

Ratio

We claim that during the first phase (Steps 1 and 2) our algorithm uses at most $\omega_m(G)+2$ colors. To see this, notice that in Step 1 each vertex v uses at most $\kappa(v)$ colors from its base color palette and, by Fact 2.1 and the fact that there are three base colors, we know that no more than $3\lceil \omega_m(G)/3\rceil \leq \omega_m(G)+2$ colors are used. Note also that in Step 5b we use only those colors from base color palettes which were not used in Step 1, so altogether no more than $\omega_m(G)+2$ colors from base color palettes are used in total in the first and second phase.
To count the number of colors used in the second phase (Steps 3-6) notice that we can divide the demands of each vertex in G_2 into eight equal parts. In Step 3 we introduce three new palettes that contain $\lceil \omega_m(G_2)/8\rceil$ colors each. In Step 5 we introduce next palette that contain $\lceil 3\omega_m(G_2)/8\rceil$ colors, and repeat this operation in Step 6.

Let $A(G)$ denote the number of colors used by our algorithm for the graph G. Thus, since $\omega_m(G_2) \leq \lceil \omega_m(G)/3\rceil \leq \omega_m(G)/3+1$, the total number of colors used by our algorithm is at most

$$A(G) \leq \omega_m(G)+2+3\left(\frac{\omega_m(G_2)}{8}+1\right)+\left(\frac{3\omega_m(G_2)}{8}+1\right)+\left(\frac{3\omega_m(G_2)}{8}+1\right)=$$

$$=\omega_m(G)+2+\frac{9\omega_m(G_2)}{8}+3<\omega_m(G)+\frac{9\omega_m(G)}{24}+2+5=\frac{33}{24}\omega_m(G)+7.$$

The performance ratio for our strategy is $33/24$, hence we arrived at the statement of Theorem 1.1.

5 Conclusion

We have given a 1-local 33/24-approximation algorithm for multicoloring hexagonal graphs. This implies a 33/24-competitive solution for the online frequency allocation problem, which involves servicing calls in each cell in a cellular network. The distributed algorithm is practical in the sense that the frequency allocation for each base station is done locally, based on the information about itself and its neighbors only, and the time complexity is constant.

According to goal ratio 4/3, the improvement of the result is the largest since paper [5] published in the year 2000. Present authors strongly believe that by use of this kind of method the competitive ratio achieved cannot be improved. Namely, the conjecture of McDiarmid and Reed [6] implies that 9/8 is the best possible competitive ratio on triangular-free hexagonal graphs. (The best proven competitive ratio for triangle free graphs is still 7/6 [3, 9, 10].) An essentially new idea has to be found to improve the ratio further.

References

[1] Chin, F.Y.L., Zhang, Y., Zhu H. *A 1-local 13/9-competitive Algorithm for Multicoloring Hexagonal Graphs*, Algorithmica, vol. 54, pp 557-567 (2009)

[2] Hale, W.K. *Frequency assignment: theory and applications*, Proceedings of the IEEE, vol 68(12), pp 1497-1514 (1980)

[3] Havet, F. *Channel assignment and multicoloring of the induced subgraphs of the triangular lattice*, Discrete Mathematics, vol. 233, pp 219-231 (2001)

[4] Havet, F., Žerovnik, J. *Finding a five bicolouring of a triangle-free subgraph of the triangular lattice*, Discrete Mathematics vol. 244, pp 103-108 (2002)

[5] Janssen, J., Krizanc, D., Narayanan, L., Shende, S. *Distributed Online Frequency Assignment in Cellular Networks*, Journal of Algorithms, vol. 36(2), pp 119-151 (2000)

[6] McDiarmid, C., Reed, B. *Channel assignment and weighted coloring*, Networks, vol. 36(2), pp. 114-117 (2000)

[7] Narayanan, L. *Channel assignment and graph multicoloring*, Handbook of wireless networks and mobile computing, pp 71-94, Wiley, New York, (2002)

[8] Narayanan, L., Shende, S. *Static frequency assignment in cellular networks*, Algorithmica, vol. 29(3), pp 396-409 (2001)

[9] Sau, I., Šparl, P., Žerovnik, J. *Simpler multicoloring of triangle-free hexagonal graphs*, Discrete mathematics, vol. 312(1), pp. 181-187 (2012).

[10] Šparl, P., Witkowski, R., Žerovnik, J. *1-local 7/5-competitive algorithm for multicoloring hexagonal graphs*, Algorithmica, vol. 64(4), pp. 564-583 (2012).

[11] Šparl, P., Žerovnik, J. *2-local 4/3-competitive Algorithm for Multicoloring Hexagonal Graphs*, Journal of Algorithms, vol. 55(1), pp 29-41 (2005)

[12] Šparl, P., Žerovnik, J. *2-local 5/4-competitive algorithm for multicoloring triangle-free hexagonal graphs*, Information Processing Letters, vol. 90(5), pp 239-246 (2004)

[13] Sudeep, K.S., Vishwanathan, S. *A technique for multicoloring triangle-free hexagonal graphs*, Discrete Mathematics, vol. 300, pp. 256-259 (2005)

[14] Witkowski, R., *A 1-local 17/12-competitive Algorithm for Multicoloring Hexagonal Graphs*, Lecture Notes of Computer Science, vol. 5699/2009, pp 346-356 (2009)

[15] Witkowski, R., Žerovnik, J. *1-local 7/5-competitive Algorithm for Multicoloring Hexagonal Graphs*, Electronic Notes in Discrete Mathematics, vol 36, pp 375-382 (2010)

[16] Žerovnik, J. *A distributed 6/5-competitive algorithm for multicoloring triangle-free hexagonal graphs*, International Journal of Pure and Applied Mathematics, vol. 23(2), pp 141-156 (2005)