1-Local 7/5-Competitive Algorithm for Multicoloring Hexagonal Graphs

Petra Šparl · Rafał Witkowski · Janez Žerovnik

Received: 3 March 2010 / Accepted: 7 August 2011 / Published online: 7 September 2011
© The Author(s) 2011. This article is published with open access at Springerlink.com

Abstract 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 frequencies assigned to them, 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 every vertex knows its position in the graph. We present a 1-local 7/5-competitive distributed algorithm for multicoloring a hexagonal graph, thereby improving the previous 1-local 17/12-competitive algorithm.

R. Witkowski was supported by grant N206 017 32/2452 for years 2007–2010.

P. Šparl and J. Žerovnik were supported in part by ARRS, the research agency of Slovenia.

P. Šparl

Faculty of Organizational Sciences, University of Maribor, Kidričeva 55a, 4000 Kranj, Slovenia

e-mail: petra.sparl@fov.uni-mb.si

P. Šparl · J. Žerovnik

IMFM, Ljubljana, Slovenia

J. Žerovnik

e-mail: janez.zerovnik@imfm.si

R. Witkowski (✉)

Faculty of Mathematics and Computer Science, Adam Mickiewicz University, ul. Umultowska 87,
61-614 Poznań, Poland

e-mail: rmiw@amu.edu.pl

J. Žerovnik

Faculty of Mechanical Engineering, University of Ljubljana, Aškerčeva 6, 1000 Ljubljana, Slovenia

e-mail: janez.zerovnik@fs.uni-lj.si

Keywords Multicoloring · Hexagonal · Hexagonal graph · Frequency assignment · Cellular network · 1-Local · Competitive

1 Introduction

A fundamental problem concerning cellular networks is to assign sets of frequencies (colors) to transmitters (vertices) in order to avoid unacceptable interferences [2, 6]. The number of frequencies demanded at a transmitter may vary between transmitters. In a usual cellular model, the transmitters are the centers 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* 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*. The phenomenon of hexagonal graphs and its multicoloring arises naturally in studies of cellular networks.

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 . 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*, $\omega(G)$, see formal definition in Sect. 2. It is well known that $\chi_m(G) \geq \omega(G)$ and that it is NP-complete problem to decide whether $\chi_m(G) = \omega(G)$ [5]. There was a lot of work done on finding approximation algorithms that imply upper bounds for the multichromatic number of hexagonal graphs.

An assumption that naturally arises in hexagonal graphs which model cellular networks is that each vertex is aware of its location. 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 only the information about the demands of vertices at distance at most k from v .

A framework for studying distributed online assignment in cellular networks was developed in [4]. In [4] a distinction between *online* and *offline* algorithms was introduced and the definition of p -competitive algorithm was given. 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 consider only 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 [4]. 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 [4], 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 [10]. The best ratio for 1-local case was first improved to $13/9$ [1], and later to $17/12$ [11]. In this paper we develop a new 1-local algorithm which uses no more than $\frac{7}{5}\omega(G) + O(1)$ colors, implying the existence of a $7/5$ -competitive algorithm.

It may be worth mentioning that the approximation bound for multicoloring algorithms on hexagonal graphs $\chi_m(G) \leq (4/3)\omega(G) + O(1)$ [5, 7, 10] is still the best known, both for distributed and not distributed models of computation.

Our algorithm substantially differs from the algorithms in [1] and [10] which are composed of two stages. At the first stage, a triangle-free hexagonal graph with weighted clique number no larger than $\lceil \omega(G)/3 \rceil$ is constructed from G , while at the second stage an algorithm for multicoloring a triangle-free hexagonal graph is used (see [1, 3, 9, 12]). Our improvement is based on an idea to borrow some colors used in the first stage and to use them for the demands of the second stage. This in particular, implies that the second stage of our algorithm cannot be applied as a stand-alone algorithm for the multicoloring arbitrary triangle-free hexagonal graphs.

The main result of this paper is

Theorem 1.1 *There is a 1-local distributed approximation algorithm for multicoloring hexagonal graphs which uses at most $\frac{7}{5}\omega(G) + O(1)$ colors. Time complexity of the algorithm at each vertex is constant.*

In [4] 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 $7/5$ -competitive online algorithm for multicoloring hexagonal graphs.*

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

2 Basic Definition and Useful Facts

A vertex weighted graph is given by a triple $G(E, V, d)$, where V is the set of vertices, E is the set of edges and $d : V \rightarrow \mathbb{N}$ is a weight function assigning (nonnegative) integer demands to vertices of G .

Following the notation from [5], 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. 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 $d : V \rightarrow \{0, 1, 2, \dots\}$ 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 Fig. 1). Sometimes we want to consider (unweighted) hexagonal graphs $G = (V, E)$ that can be defined as subgraph induced by subsets of vertices of the triangular lattice. In both

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

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*, $\omega(G)$, is the maximal clique weight on G , where the weight of a clique is the sum of demands on its vertices. As cliques in hexagonal graphs have at most three vertices, 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 Fig. 1), i.e. $bc(v) \in \{R, G, B\}$.

An induced subgraph of the triangular lattice without 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 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 Fig. 2). A vertex which is not a corner is called a *non-corner*.

Definition 2.1 For a given graph $G = (V, E)$, a coloring $f : V \rightarrow \{1, \dots, k\}$ is *k-good* if for every odd cycle $C \subseteq G$ and for every i , $1 \leq i \leq k$, there is a vertex $v \in V(C)$ such that $f(v) = i$. A graph G is *k-good* if such coloring exists.

Lemma 2.1 [8] Consider a 3-coloring (R, G, 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.

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

Fig. 3 An example of a hexagonal graph with (extra) 4-coloring

As the elegant proof of Sudeep and Vishwanathan [8] 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. $\square$

In addition to the basic 3-coloring we will also use the following obvious proper 4-coloring of the infinite triangular lattice which gives partition of the vertex set of any hexagonal graph into four independent sets. Let us denote a color of any vertex v in this 4-coloring by $ec(v)$ and call it *extra color* (for simplicity we will use cyan, magenta, yellow and black as extra colors and their arrangement is given in Fig. 3), i.e. $ec(v) \in \{C, M, Y, K\}$.

Note that this coloring is defined in such a way that each line is properly colored by exactly two of the four extra colors. Moreover, we can prove the following fact:

Lemma 2.2 *In a triangle-free hexagonal graph, corners of each odd cycle meet at least three of the four extra colors.*

Proof Assume that a cycle in a triangle-free hexagonal graph meets corners of only one color. Observe that the distance on a line between corners of the same extra color is even. Hence, the length of cycle is even.

Assume now that a cycle in a triangle-free hexagonal graph meets corners of only two colors. The distance on a straight line between two corners of the same color is always even, and the distance between two corners of a different color is always odd. Since we have a cycle, each time when we have a straight path from a corner of the first color to a corner of the second color, we must have a straight path from corner of the second color to the first one as well. Hence, the length of the cycle is even.

Therefore, all odd cycles meet at least three colors in this 4-coloring. $\square$

Notice that if a graph G is k -good then after removing vertices colored by any of those k colors, the resulting graph is bipartite. For any weighted bipartite graph H , $\chi_m(H) = \omega(H)$ (see [7]), 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 $\{1, 2, \dots, d(v)\}$, and with each vertex $v \in V''$ we associate a set of colors $\{m(v) + 1, m(v) + 2, \dots, m(v) + d(v)\}$, where $m(v) = \max\{d(u) : \{u, v\} \in E\}$.*

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

$$\begin{aligned} & \max\{\max\{d(v) : v \in V'\}, \max\{d(v) + m(v) : v \in V''\}\} \\ &= \max\{\max\{d(v) : v \in V'\}, \max\{d(v) + \max\{d(u) : \{u, v\} \in E\} : v \in V''\}\} \\ &= \max\{\max\{d(v) : v \in V'\}, \max\{d(v) + d(u) : \{u, v\} \in E\}\} = \omega(G). \end{aligned} \quad \square$$

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:

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

where $\tau(T)$ is the set of all triangles of T .

For each vertex $v \in G$, 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 an average weight of the triangle $\{u, v, t\} \in \tau(T)$.

Clearly, the following fact holds.

Fact 2.1 For each $v \in G$,

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

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 $c \in \{R, G, B\}$ is one of the base colors, and it is called *additional color palette* if $c \in \{C, M, Y, K\}$ is one of the extra colors.

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 only this information.

3 Algorithm

Our algorithm consists of three main phases. In the first phase (Steps 1 and 2 below) vertices take $\kappa(v)$ colors from its base color palette, so use no more than $\omega(G)$ 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 $\lceil \omega(G)/3 \rceil$. Very heavy vertices in G are isolated in the remaining graph and therefore are easily and fully colored. However, they have to be treated separately (Step 2). In the second phase (Steps 3 and 4) we color black corners by assigning free colors from its neighbors. The remaining graph is a triangle-free hexagonal graph without black corners, so that only cyan, magenta and yellow colored corners are left. This graph is 6-good, and we can construct this 6-good coloring. Recall that it holds for a 6-good coloring that the graph is bipartite after removing vertices of any of these six colors. Roughly speaking, if we apply six times Procedure 2.1 and color such graphs optimally with weight function equal at each vertex to $1/5$ of its demands, then the remaining graph would be fully colored.

More precisely, our algorithm consists of the following steps:

Algorithm

Input: Weighted hexagonal graph $G = (V, E, d)$, where every vertex knows its position in the graph G .

Output: A proper multicoloring of G , using at most $7/5 \cdot \omega(G)$ colors.

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

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

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 $\min\{\kappa(v), d(v)\}$ 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 each vertex $v = (x, y) \in V$ compute its extra color $ec(v)$

$$ec(v) = \begin{cases} C, & \text{if } x \bmod 2 + 2 * (y \bmod 2) = 0, \\ M, & \text{if } x \bmod 2 + 2 * (y \bmod 2) = 1, \\ Y, & \text{if } x \bmod 2 + 2 * (y \bmod 2) = 2, \\ K, & \text{if } x \bmod 2 + 2 * (y \bmod 2) = 3. \end{cases}$$

Step 4 For each corner vertex $v \in V_2$ with $ec(v) = K$ assign free colors from the first $\kappa(v)$ of the base color palettes of its neighbors in T . Construct a new graph $G_3 = (V_3, E_3, d_3)$ where $d_3(v) = d_2(v)$, $V_3 \subseteq V_2$ is the set of vertices with no corners of $ec(v) = K$ and $E_3 \subseteq E_2$ is the set of all edges in G_2 with both endpoints from V_3 (G_3 is induced by V_3).

Step 5 Determine a 6-good coloring of G_3 —put each vertex $v \in V_3$ in exactly one of the six sets defined by:

- I* : red non-corners in G_3 ,
- II* : green non-corners in G_3 ,
- III* : blue non-corners in G_3 ,
- IV* : cyan corners in G_3 ,
- V* : magenta corners in G_3 ,
- VI* : yellow corners in G_3 .

Step 6 For each set $S \in \{I, II, III, IV, V, VI\}$ do as follows: remove from G_3 all vertices from S , find a bipartition of the remaining graph and apply Procedure 2.1 to satisfy $\lceil d_3(v)/5 \rceil$ demands in $G_3 \setminus S$ by colors from the corresponding additional color palette.

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. From now on, no further communication will be 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 graph G , there is no reclassification in graphs $G_i, i \in 1, 2, 3$.

In 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 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:

$$\begin{aligned} d(v) + d(u) + d(t) &> d_1(v) + \kappa(v) + d_1(u) + \kappa(u) + d_2(t) + \kappa(t) \\ &\geq d_1(v) + d_1(u) + d_1(t) + 3a(u, v, t) \\ &\geq d_1(v) + d_1(u) + d_1(t) + d(v) + d(u) + d(t) \end{aligned}$$

a contradiction. Therefore, the graph G_1 does not contain a 3-clique, i.e. it is a triangle-free hexagonal graph. $\square$

In Step 2 only vertices with $d_1(v) > \kappa(v)$ (very heavy vertices in G) are colored. We are going to prove that 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 $bc(v) = R$. Denote by

$$D_G(v) = \min\{\kappa(v) - d(u) : \{u, v\} \in T, bc(u) = G\},$$

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

Obviously, $D_G(v), D_B(v) > 0$ 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 $D_G(v)$ free colors from the green base color palette and at least $D_B(v)$ free colors from the blue base color palette. If vertex v could assign those colors to itself, we would have G_2 with $\omega(G_2) \leq \lceil \omega(G)/3 \rceil$. We formally prove this in the following lemma:

Lemma 4.2 *In G_1 for every edge $\{v, u\} \in E_1$ holds:*

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

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:

$$\begin{aligned} d(v) + d(u) &= d_1(v) + \kappa(v) + d_1(u) + \kappa(u) \\ &> 2\kappa(v) + \kappa(u) \geq 3a(u, v, t) \geq d(u) + d(v), \end{aligned}$$

a contradiction. □

Fact 4.1

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

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 $\{v, u\} \in E_2$ from Lemma 4.2 and Fact 2.1 we have:

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

For each isolated vertex $v \in G_2$ we should have $d_2(v) \leq \lceil \omega(G)/3 \rceil$. Indeed, if $d_2(v) > \kappa(v)$, then it holds by Fact 2.1. If $d_2(v) > \kappa(v)$, then $d_1(v) > \kappa(v)$, so v has to borrow colors from its neighbors' base color palettes in Step 2. Then, for $bc(v) = R$,

$$\begin{aligned} d_2(v) &= d_1(v) - D_G(v) - D_B(v) \leq d(v) - \kappa(v) - \kappa(v) + d(u) - \kappa(v) + d(t) \\ &\leq 3a(v, u, t) - 3\kappa(v) \leq 0 \end{aligned}$$

for some $\{v, u, t\} \in \tau(T)$. Hence, $d_2(v) \leq \lceil \omega(G)/3 \rceil$, and so $\omega(G_2) \leq \lceil \omega(G)/3 \rceil$. □

In Step 3 there is nothing to prove.

In Step 4 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 his 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 that 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 are 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.

We are going to prove that each black corner in G_2 has enough unused colors in its neighborhood in G to be finally multicolored, and that no conflict occurs following this coloring. In Step 4 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 with the same base color. Without loss of generality, assume that $bc(v) = R$ and its slight neighbors' base color is blue. Recall function D_B from Step 2—we have $D_B(v)$ free colors from blue base color palette. We claim that

Lemma 4.3 *If v is a corner in G_2 with three slight neighbors colored blue, then*

$$d_2(v) \leq D_B(v).$$

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

$$\begin{aligned} \kappa(v) + d_2(v) + a(u, v, t) + d(u) &\stackrel{\star}{\leq} d(v) + d(t) + d(u) \leq 3a(u, v, t) \\ &\leq a(u, v, t) + 2\kappa(v), \\ d(u) + d_2(v) &\leq \kappa(v), \\ d_2(v) &\leq \kappa(v) - d(u), \end{aligned}$$

and inequality $\star$ 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. $d_2(v) \leq D_B(v)$. $\square$

Therefore, vertex v has as much as $d_2(v)$ free colors from the blue base color palette at his disposal. After that, due to fact that extra colors provide a proper 4-coloring of G_2 no conflict occurs as all black corners in G_2 are well separated.

For correctness of Step 5 we need to prove the following fact:

Fig. 4 Graph G_{IV} with bipartition

Fact 4.2 Graph G_3 is 6-good.

Proof A proper 6-coloring of this graph is given in description of Step 5. It is clear that all sets $\{I, II, III, IV, V, VI\}$ are independent and each vertex $v \in G_3$ is in exactly one of those sets. Each odd cycle meets all of those six sets, which follows from Lemmas 2.1 and 2.2. $\square$

In Step 6 we have 6 substeps, in each we remove from G_3 vertices from one of the sets $\{I, II, III, IV, V, VI\}$. In the remaining graphs we need to determine their bipartition in 1-local model. For sets $\{I, II, III\}$ the procedure is the same—for simplicity, consider only graph $G_I = (V_I, E_I)$ where V_I is obtained from V_3 by removing vertices from set I , i.e. all red non-corners. Following the proof of Lemma 2.1 we can make the following bipartition:

- first set: blue vertices and left red corners, i.e. red corners with all green neighbors
- second set: green vertices and right red corners, i.e. red corners with all blue neighbors.

It is clear that this is a bipartition of G_I and that it can be computed in the 1-local model.

For sets $\{IV, V, VI\}$ we also have an easy procedure—for simplicity, consider only graph $G_{IV} = (V_{IV}, E_{IV})$ where V_{IV} is obtained from V_3 by removing vertices from set IV , i.e. all cyan corners. Recall that black corners were removed in Step 4 and hence there may exist only magenta or yellow corners in G_{IV} . We can make the following bipartition:

- first set: magenta vertices and non-corners with neighbors colored yellow or black
- second set: yellow vertices and non-corners with neighbors colored magenta or cyan.

In the first and second set magenta and yellow vertices are well separated. All other vertices are non-corners after removing black corners in Step 4 and cyan corners in

Fig. 5 Input weighted hexagonal graph G with 3-coloring

Fig. 6 Graph G with base coloring and $\kappa(v)$ in each vertex

this substep. Each non-corner has both of its neighbors of the same extra color—if it is yellow, then the vertex has to be in the first set, while if it is magenta, then vertex has to be in the second set. The remaining non-corners with neighbors of extra color black or cyan form a line in G_{IV} that never meets any corner, since corners in these two colors do not exist in graph G_{IV} . We can divide them into two sets of our bipartition in arbitrary way (see Fig. 4).

Next, we can apply Procedure 2.1 with bipartitions defined above and weight function on each vertex v equal to $\lceil d_3(v)/5 \rceil$, assigning colors from one of six additional color palettes. The problem is that, under 1-locality assumption, vertices cannot calculate value of d_3 of their neighbors, which is needed in Procedure 2.1 to calculate value $m(v) = \max\{\lceil d_3(u)/5 \rceil : \{u, v\} \in E_3\}$. However, we can replace $d_3(u)$ by $d_3^v(u)$, which is the number of expected demands on vertex u in vertex v after Step 4, and take $m'(v) = \max\{\lceil d_3^v(u)/5 \rceil : \{u, v\} \in E_3\}$. More precisely,

$$d_3^v(u) = d(u) - \max\{a(u, v, t) : \{u, v, t\} \in \tau(T)\}.$$

Fig. 7 Step 1—coloring all light vertices

Fig. 8 Graph G_1 with base coloring and $d_1(v)$ in each vertex

Note that $d_3^v(u) \geq d_3(u)$ for any $\{u, v\} \in E_3$. However,

Lemma 4.4 For every $\{v, u\} \in E_3$ we have

$$d_3(v) + d_3^v(u) \leq \kappa(v).$$

Proof Assume that this inequality does not hold, hence $d_3(v) + d_3^v(u) > \kappa(v)$. 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:

$$\begin{aligned} d(v) + d(u) &= d_3(v) + \kappa(v) + d_3^v(u) + b(u, v) > 2\kappa(v) + b(u, v) \\ &\geq 3a(u, v, t) \geq d(u) + d(v), \end{aligned}$$

a contradiction. □

Fig. 9 Step 2 **Fig. 10** Graph G_2 with 3-coloring and $d_2(v)$ in each vertex

Hence, if we use d_3^v instead of d_3 at each vertex from the second set of our bipartition, we formally work with a new graph $\tilde{G}_3$ with a new $\omega(\tilde{G}_3)$. Because of Fact 2.1 and Lemma 4.4 we have the inequality

$$\omega(\tilde{G}_3) \leq \left\lceil \frac{\omega(G)}{3} \right\rceil$$

analogous to the inequality from Fact 4.1. Thus, Procedure 2.1 works and in one substep uses at most $\lceil \omega(G)/15 \rceil + 1$ colors from one of the additional color palettes.

During Step 6 each vertex v participates in exactly five from six rounds (in each round one set is removed from G_3) and $\lceil d_3(v)/5 \rceil$ colors are assigned in each. Therefore, at the end, all of the demands are satisfied.

Fig. 11 Step 3—graph G_2 with extra coloring

Fig. 12 Step 4

4.1 Ratio

We claim that during the first phase (Steps 1 and 2) our algorithm uses at most $\omega(G) + 2$ 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 $3\lceil\omega(G)/3\rceil \leq \omega(G) + 2$ colors are used. Note also that in Steps 2 and 4 we use only those colors from base color palettes which were not used in Step 1, so altogether no more than $\omega(G) + 2$ colors are used in total in the first and second phase.

To count the number of colors used in the third phase (Step 6) notice that we divide the demands of each vertex in G_3 into five equal parts. Each vertex v participates in five from six rounds and assigns $\lceil d_3(v)/5 \rceil$ colors in each round. Since in each round of Step 6 we use $\omega(G_3)/5 + 1$ colors from additional color palette, therefore we use only $6(\omega(G_3)/5 + 1)$ colors in total.

Fig. 13 Step 5—graph G_3 with 6-good coloring

Fig. 14 Step 6a—in graph G_3 we remove vertices from set I (the same with II and III)

Let $A(G)$ denote the number of colors used by our algorithm for the graph G . Thus, since $\omega(G_3) \leq \omega(G_2) \leq \lceil \omega(G)/3 \rceil \leq \omega(G)/3 + 1$, the total number of colors used by our algorithm is at most

$$A(G) \leq \omega(G) + 2 + 6 \left(\frac{\omega(G_3)}{5} + 1 \right) \leq \omega(G) + 2 + \frac{6\omega(G)}{15} + \frac{6}{5} + 6 < \frac{7}{5}\omega(G) + 10.$$

The performance ratio for our strategy is $7/5$, hence we arrived at the statement of Theorem 1.1.

4.2 Working Example

Let take a weighted hexagonal graph as shown on Fig. 5.

In Step 0 we determine base coloring using the coordinates, as shown on Fig. 6.

Fig. 15 Step 6a—going back to the base colors

Fig. 16 Step 6a—bipartition

In Step 1 we fully color all light vertices (Fig. 7) and the remaining graph G_1 is triangle-free (see Fig. 8).

In Step 2 we fully color all very heavy vertices (Fig. 9) and the remaining graph G_2 is still triangle-free (see Fig. 10).

In Step 3 we determine extra coloring using the coordinates, as shown on Fig. 11.

In Step 4 we fully color all black vertices (Fig. 12).

In Step 5 we determine 6-good coloring of the remaining graph G_3 (see Fig. 13).

In Step 6 there are two cases. In the first one we remove from G_3 one kind of non-corners and after finding a bipartition we use Procedure 2.1 (see Figs. 14, 15 and 16).

Fig. 19 Step 6b—bipartition

Acknowledgement The authors wish to thank to an anonymous referee for careful reading of the manuscript and remarks that helped to considerably improve the presentation.

Open Access This article is distributed under the terms of the Creative Commons Attribution Noncommercial License which permits any noncommercial use, distribution, and reproduction in any medium, provided the original author(s) and source are credited.

References

1. Chin, F.Y.L., Zhang, Y., Zhu, H.: A 1-local 13/9-competitive algorithm for multicoloring hexagonal graphs. *Algorithmica* **54**, 557–567 (2009)
2. Hale, W.K.: Frequency assignment: theory and applications. *Proc. IEEE* **68**(12), 1497–1514 (1980)
3. Havet, F., Žerovnik, J.: Finding a five bicolouring of a triangle-free subgraph of the triangular lattice. *Discrete Math.* **244**, 103–108 (2002)
4. Janssen, J., Krizanc, D., Narayanan, L., Shende, S.: Distributed online frequency assignment in cellular network. *J. Algorithms* **36**(2), 119–151 (2000)
5. McDiarmid, C., Reed, B.: Channel assignment and weighted coloring. *Networks* **36**(2), 114–117 (2000)
6. Narayanan, L.: Channel assignment and graph multicoloring. In: *Handbook of Wireless Networks and Mobile Computing*, pp. 71–94. Wiley, New York (2002)
7. Narayanan, L., Shende, S.M.: Static frequency assignment in cellular networks. *Algorithmica* **29**(3), 396–409 (2001)
8. Sudeep, K.S., Vishwanathan, S.: A technique for multicoloring triangle-free hexagonal graphs. *Discrete Math.* **300**, 256–259 (2005)
9. Šparl, P., Žerovnik, J.: 2-local 5/4-competitive algorithm for multicoloring triangle-free hexagonal graphs. *Inf. Process. Lett.* **90**(5), 239–246 (2004)
10. Šparl, P., Žerovnik, J.: 2-local 4/3-competitive algorithm for multicoloring hexagonal graphs. *J. Algorithms* **55**(1), 29–41 (2005)
11. Witkowski, R.: A 1-local 17/12-competitive algorithm for multicoloring hexagonal graphs. In: *Lecture Notes of Computer Science*, vol. 5699/2009, pp. 346–356 (2009)
12. Žerovnik, J.: A distributed 6/5-competitive algorithm for multicoloring triangle-free hexagonal graphs. *Int. J. Pure Appl. Math.* **23**(2), 141–156 (2005)