1-local 7/5-competitive Algorithm for Multicoloring Hexagonal Graphs

Petra Šparl
University of Maribor,
FCE, Smetanova 17,
SI-2000 Maribor, Slovenia
petra.sparl@uni-mb.si

Rafał Witkowski*
Adam Mickiewicz University,
Faculty of Mathematics and Computer Science,
ul. Umultowska 87
Poznań, Poland
rmiw@amu.edu.pl

Janez Žerovnik†
University of Ljubljana,
Faculty of mechanical engineering,
Aškerčeva 6,
SI-1000 Ljubljana, Slovenia
and
Institute of Mathematics, Physics and Mechanics,
Ljubljana, Slovenia
janez.zerovnik@fs.uni-lj.si, janez.zerovnik@imfm.si

February 26, 2010

Abstract

Hexagonal graphs are graphs induced on subsets of vertices of triangular lattice. They arise naturally in studies of cellular networks. 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.

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]. The number of

*This work was supported by grant N206 017 32/2452 for years 2007-2010
†Supported in part by ARRS, the research agency of Slovenia.

frequencies demanded at a transmitter may vary between transmitters. In a usual cellular model, transmitters are centers of hexagonal cells and the corresponding adjacency graph is a 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 on the subset of the triangular lattice of vertices of positive demand is called a (vertex weighted) *hexagonal graph*. Hexagonal graphs arise naturally in studies of cellular networks. A *proper multicoloring* of G is a mapping f from $V(G)$ to subsets of integers such that $|f(v)| \geq d(v)$ 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 graph G. The minimal cardinality of 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*, $\omega(G)$, defined as follows: the weight of a clique of G is the sum of demands on its vertices and $\omega(G)$ is the maximal clique weight on G. Clearly, $\chi_m(G) \geq \omega(G)$. It was shown in [4] that it is NP-complete problem to decide whether $\chi_m(G) = \omega(G)$.

A framework for studying distributed online assignment in cellular networks was developed in [3]. 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. In [3] distinction between *online* and *offline* algorithms was introduced and the definition of *p-competitive algorithm* was given. In the same paper, 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 [6]. The best ratio for 1-local case was first improved to 13/9 [1], and later to 17/12 [9]. 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 $\chi_m(G) \leq (4/3)\omega(G) + O(1)$ [4, 5, 6] is still the best known, both for distributed and not distributed models of computation. In view of this one can naturally take 4/3 as (maybe too ambitious) goal ratio for 1-local algorithms. With this assumption, the improvement [1] from 3/2 to 13/9 decreases the difference to the goal ratio 4/3 for one third (because (3/2-13/9) / (3/2-4/3) = 1/3). Later improvements from 13/9 to 17/12 [9], and from 17/12 to 7/5 (this paper) are both closing of the remaining gap for 1/4 of the difference.

Our algorithm substantially differs from the algorithms in [1] and [6] 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], [7], [10]). Our improvement is based on the idea to borrow some colors used in the first stage for demands of the second stage. This in particular implies that the second stage of our algorithm cannot be applied for 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 hexag-*

onal graphs which uses at most $\frac{7}{5}\omega(G) + O(1)$ colors. Time complexity of the algorithm at each vertex is constant.

In [3] 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 Section 3 we present an overview of the algorithm, while in Section 4 we will prove 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 $d : V \to \mathbb{N}$ is a weight function assigning (nonnegative) integer demands to vertices of G.

Following the notation from [4], 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 $d : V \to \{0, 1, 2, \ldots\}$ on vertices of triangular lattice. We define a *weighted hexagonal graph* $G = (V, E, d)$ as an induced subgraph on vertices of positive demand on the triangular lattice (see Figure 1). Sometimes we want to work with (unweighted) hexagonal graphs $G = (V, E)$ that can be defined as induced on subsets of vertices of the triangular lattice.

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 base colors and their arrangement is given in Figure 1), i.e. $bc(v) \in \{R, G, B\}$.

We call a *triangle-free hexagonal graph* an induced subgraph of the triangular lattice without 3-clique. We define a *corner* in a triangle-free hexagonal graph as a vertex which has at least two neighbors and none of them are at angle π. A vertex is a *right corner* if it has an up-right or a down-right neighbor, and otherwise it is a *left corner* (see Figure 2). A vertex which is not a corner is called a *non-corner*.

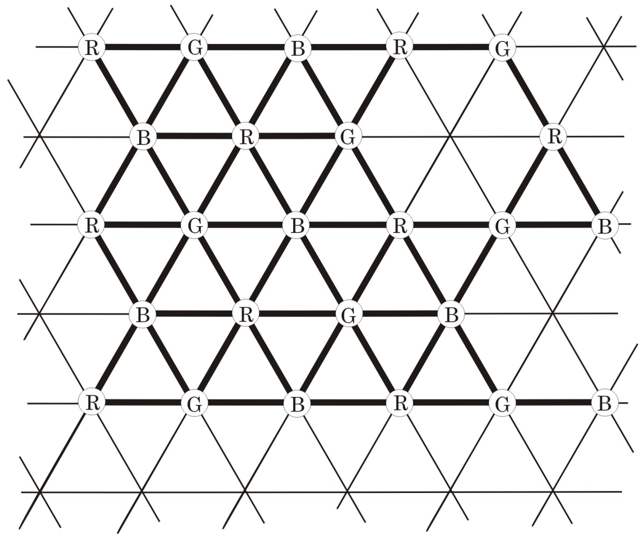

Figure 1: An example of a hexagonal graph (with base coloring)

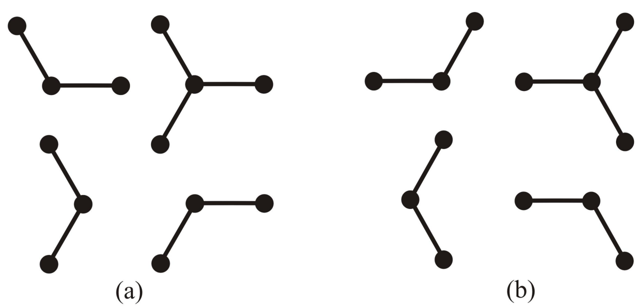

Figure 2: All possibilities for: (a) - left corners, (b) - right corners

Definition 2.1 *In graph $G = (V, E)$, we call a coloring $f : V \to \{1, \ldots, k\}$ k-*good *if for every odd cycle in G and for every i, $1 \leq i \leq k$, there is a vertex $v \in V$ in the cycle such that $f(v) = i$. A graph 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.*

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 $2\pi/3$

since the graph is triangle-free). 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. ■

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 Figure 3), i.e. $ec(v) \in \{C, M, Y, K\}$.

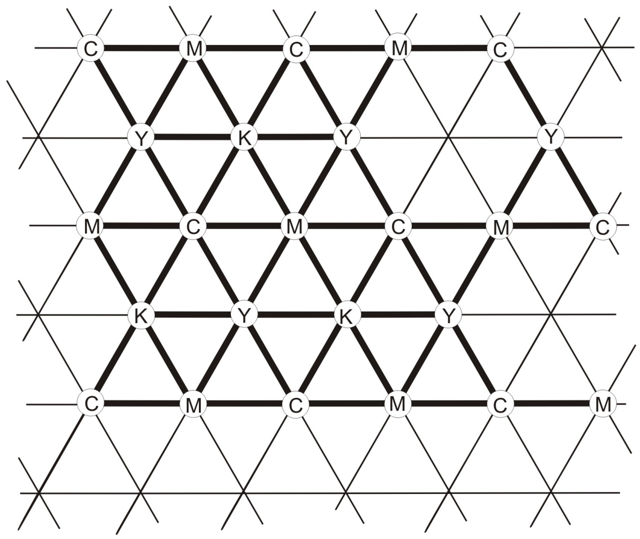

Figure 3: An example of a hexagonal graph with (extra) 4-coloring

Note that this coloring is defined in the 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 triangle-free hexagonal graph meets corners of only one color. Note that the distance on a line from one corner to another corner of the same extra color is even. Hence, the length of cycle is even.
Assume now that cycle in triangle-free hexagonal graph meets corners of only two colors. Distance on a straight line between two corners of the same color is always even, and distance between two corners of different color is always odd. Each time when we have a straight path from a corner of the first color to a corner of the second color, we have to have a straight path from corner of the second color to the first one as well, since we have

a cycle. Hence, length of the cycle is even.
Therefore all odd cycles meet at least three colors in this 4-coloring. ■

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 [5]), 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 to each vertex $v \in V'$ we assign a set of colors $\{1, 2, \ldots, d(v)\}$, while with each vertex $v \in V''$ we associate a set of colors $\{m(v)+1, m(v)+2, \ldots, m(v)+d(v)\}$, where $m(v) = \max\{d(u) : \{u, v\} \in E\}$.*

Proof: Notice that this procedure is 1-local, since each vertex v uses only its weight function $d(v)$ or calculate value $m(v)$ which is well known from its neighbors. From definition of $m(v)$ it is easy to see, that no conflict occurs in this multicoloring. The biggest number of color used is

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

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

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

since in bipartite graph the only cliques are edges and isolated 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. Note also that we assume that all vertices of T which are not in G have to 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 $\{u, v, t\} \in \tau(T)$.

It is easy to observe that 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 call vertex *very heavy*.

To color vertices of G we use colors from 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 *base color palette* if $c \in \{R,G,B\}$, while it is called *additional color palette* otherwise.

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 it neighbors. In the next section we will show how using only this information, each vertex has to color itself properly in constant time and hence in a distributed way.

3 Algorithm and its correctness

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 $\omega(G)$ colors. After this phase all light vertices are fully colored while the remaining vertices form a triangle-free hexagonal graph with weighted clique number not exceeding $\lceil \omega(G)/3 \rceil$. Very heavy vertices are isolated in the remaining graph and are therefore easily colored, but 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. After this phase we construct 6-good coloring of the remaining graph. Recall that in 6-good coloring, a graph is bipartite after removing vertices of any of these six colors. Roughly speaking, if we use six times Procedure 2.1 and color such graphs optimally with weight function equal in each vertex to 1/5 of its demands, then we would fully color the remaining graph and use no more than 6/5 colors needed in the new graph.

More precisely, our algorithm consists of the following steps:

Algorithm

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

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

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) and $E_1 \subseteq E$ is the set of all edges in G with both endpoints from V_1 (E_1 is induced by V_1).

Step 2 For each vertex $v \in V_1$ with $d_1(v) > \kappa(v)$ (very heavy vertices) assign free colors from the first $\kappa(v)$ of base color palettes of its neighbors in T. Construct a new graph $G_2 = (V_2, E_2, d_2)$ where d_2 is the difference between $d_1(v)$ and the number of assigned colors 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 (E_2 is induced by V_2).

Step 3 For each vertex $v = (x, y) \in V$ compute its extra color $ec(v)$

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

Step 4 For each corner vertex $v \in V_2$ with $ec(v) = K$ assign free colors from the first $\kappa(v)$ of 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 (E_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:

$$\begin{array}{lcl} I & : & \text{red non-corners in } G_3 \\ II & : & \text{green non-corners in } G_3 \\ III & : & \text{blue non-corners in } G_3 \\ IV & : & \text{cyan corners in } G_3 \\ V & : & \text{magenta corners in } G_3 \\ VI & : & \text{yellow corners in } G_3 \end{array}$$

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 $G_3 \backslash S$ by colors from 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 more communication will be needed. Recall that each vertex knows its position (x, y) on the triangular lattice T.

In Step 0 there is nothing to prove.

In Step 1 each heavy vertex v 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, therefore

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

Proof: For any $\{v,u,t\} \in \tau(G)$, since $3\min\{\kappa(v),\kappa(u),\kappa(t)\} \geq d(v)+d(u)+d(t)$ and $\min\{\kappa(v),\kappa(u),\kappa(t)\} \geq \min\{d(v),d(u),d(t)\}$, at most two of $d_1(v), d_1(u), d_1(t)$ are strictly positive and at least one of the vertices u, v and t has all its required colors totally assigned in Step 1. Therefore, the graph G_1 does not contain 3-clique, i.e. it is a triangle-free hexagonal graph. ■

In Step 2 only vertices with $d_1(v) > \kappa(v)$ (very heavy vertices) are colored. 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, so that vertex v can assign those colors to itself. Then, we would have G_2 with $\omega(G_2) \leq \lceil \omega(G)/3 \rceil$. To prove it, we will need 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:

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

again a contradiction. ■

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 $d_2(v) \leq \kappa(v)$, 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 $bc(v) = R$,

$$d_2(v) = d_1(v) - D_G(v) - D_B(v) \leq d(v) - \kappa(v) - \kappa(v) + d(u) - \kappa(v) + d(t) \leq$$

$$\leq 3a(v, u, t) - 3\kappa(v) \leq 0$$

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 are light. Vertex v knows only where its neighbors with $d(u) \leq \max\{a(v, u, t) : \{v, u, t\} \in \tau(T)\}$ are located. We call those vertices *slight neighbors* of v. They must be light and, so, they are fully colored in Step 1. Thus, 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 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 distinction between left and right is determined in the same way as above.

In Step 4 we take 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

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

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

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

and $\star$ occurs cause $d_2(v) = d(v) - \kappa(v)$ 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 $\tau(T)$ in which there exists 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. $d_2(v) \leq D_B(v)$. ■

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

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

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. ■

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 its 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, ie. all red non-corners. From proof of Lemma 2.1 we can make the following bipartition:

- first set: blue vertices and left red corners, ie. red corners with all green neighbors
- second set: green vertices and right red corners, ie. 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, ie. 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 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. Remaining non-corners with neighbors with extra color black or cyan form a line in G_{IV} which 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 Figure 4).

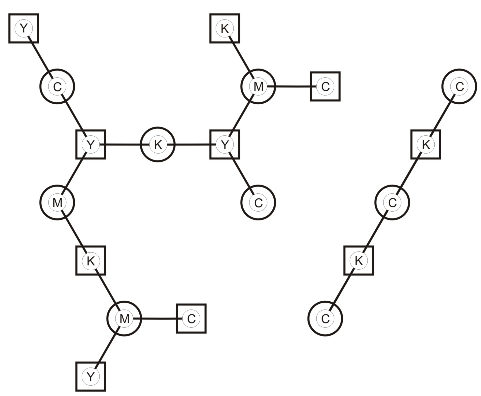

Figure 4: Graph G_{IV} with bipartition

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,

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

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:

$$d(v) + d(u) = d_3(v) + \kappa(v) + d_3^v(u) + b(u,v) > 2\kappa(v) + b(u,v) \geq$$

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

a contradiction. ■

Hence, if we use d_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 demands are satisfied.

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 had not been used in Step 1, so overall 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.

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

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

So, the performance ratio for our strategy is 7/5 and we arrived at the statement of Theorem 1.1.

5 Conclusion

We have given a 1-local 7/5-approximation algorithm for multicoloring hexagonal graphs. This implies a 7/5-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 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.

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

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

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

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

[7] 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)

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

[9] 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)

[10] 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)