Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Discrete Mathematics 36 (2010) 375–382
Electronic Notes in DISCRETE MATHEMATICS
www.elsevier.com/locate/endm

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

Rafał Witkowski [1,2]

Faculty of Mathematics and Computer Science
Adam Mickiewicz University
Poznań, Poland

Janez Žerovnik [3,4]

IMFM and Faculty of mechanical engineering
University of Ljubljana
Ljubljana, Slovenia

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.

Keywords: Frequency assignment, hexagonal graph, multicoloring.

[1] rmiw@amu.edu.pl
[2] This work was supported by grant N206 017 32/2452 for years 2007-2010
[3] janez.zerovnik@fs.uni-lj.si, janez.zerovnik@imfm.si
[4] Supported in part by ARRS, the research agency of Slovenia.

doi:10.1016/j.endm.2010.05.048

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 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 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 paper [3] it was also introduced distinction between *online* and *offline* algorithms and definition of *p-competitive algorithm* as well. In [3] a 3/2-competitive 1-local, 17/12-competitive 2-local and 4/3-competitive 4-local algorithms are 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 [8]. 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 $\chi_m(G) \leq (4/3)\omega(G) + O(1)$ [6,4,5] is still the best known, both for distributed and not distributed model 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 improvements from 3/2 to 13/9, from 13/9 to 17/12, and from 17/12 to 7/5 are closing respectively a 1/3, 1/4, and 1/4 of the remaining gap.

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* $\left\lceil \frac{7}{5}\omega(G) \right\rceil + O(1)$ *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 1.2 *There is a 1-local 7/5-competitive 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. The proofs of some lemmas, algorithm correctness and Theorem 1.1 are omitted here and can be found in full version paper.

2 Basic definition and useful facts

A vertex weighted graph is given by a triple $G(E, V, d)$, where V 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. 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).

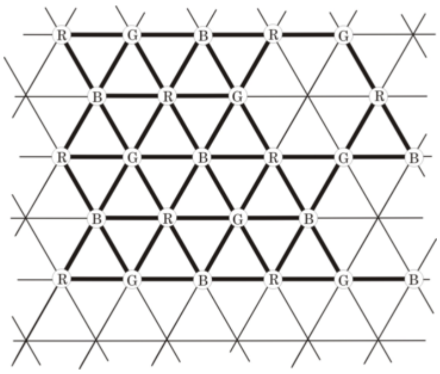

Fig. 1. An example of a hexagonal graph (with base coloring)

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*. A vertex which is not a corner is called a *non-corner*.

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 *[7] Consider a 3-coloring of the triangular lattice (R,G,B). Every odd cycle of the triangle-free hexagonal graph G contains at least one non-corner vertex of every color.*

In addition to the basic 3-coloring we will also use the following obvious 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 a *extra color* (for simplicity we will use *cyan*, *magenta*, *yellow* and *black* as extra colors and their arrangement is given in Figure 2), i.e. $ec(v) \in \{C, M, Y, K\}$.

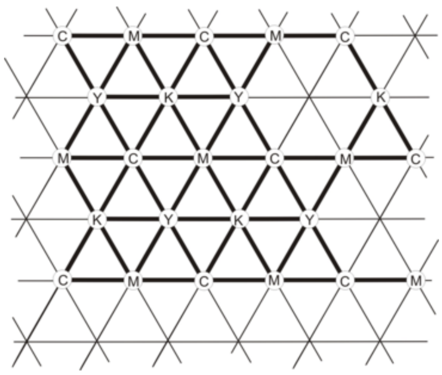

Fig. 2. An example of a hexagonal graph with (extra) 4-coloring

Note that this coloring is defined in the way that each line is properly colored by exactly two of the four extra colors. Moreover:

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

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.

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

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 the average weight of the triangle $\{u, v, t\} \in \tau(T)$.

It is easy to observe that for each $v \in G$, $\kappa(v) \le \lceil \omega(G)/3 \rceil$.

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 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 *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 its neighbors. With this knowledge, each vertex has to multicolor itself properly in constant time in a distributed way.

3 Algorithm and its correctness

Our algorithm consists of three main phases. In the first phase each vertex takes $\kappa(v)$ colors from its base color palette, so this step of the algorithm uses 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 $\lceil \omega(G)/3 \rceil$ (after technical step of removing very heavy vertices). In the second phase 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 Algorithm 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:

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

- R if $(x + 2y) \bmod 3 = 0$
- G if $(x + 2y) \bmod 3 = 1$
- B if $(x + 2y) \bmod 3 = 2$

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 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 induced by V_2.

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

- C if $x \bmod 2 + 2 * (y \bmod 2) = 0$
- M if $x \bmod 2 + 2 * (y \bmod 2) = 1$
- Y if $x \bmod 2 + 2 * (y \bmod 2) = 2$
- K if $x \bmod 2 + 2 * (y \bmod 2) = 3$

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 $ec(v) \neq K$ and $E_3 \subseteq E_2$ 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 Algorithm 1 to satisfy $\lceil d_3(v)/5 \rceil$ demands in $G_3 \backslash S$ by colors from corresponding additional color palette.

For sets $\{I, II, III\}$ the procedure of finding bipartition is the same — if we consider 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, 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.

For sets $\{IV, V, VI\}$ the procedure of finding bipartition is the same — if we consider 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, 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.

To use Algorithm 1 in 1-local model we use function d_3^v instead of d_3 in each vertex from the second set of our bipartition which is defined as

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

and means the number of expected demands on vertex u in vertex v after Step 4. It can be shown that $d_3^v(u) \geq d_3(u)$ and $d_3^v(u) \leq \kappa(v) - d_3(v)$ for all $\{v, u\} \in E_3$.

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.

4 Ratio

During the first phase (Steps 1 and 2) our algorithm uses at most $\omega(G) + 2$ colors. Note 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. Since in each round of Step 6 we use $\omega(G_3)/5 + 1$ colors from additional color palette, we use only $6(\omega(G_3)/5 + 1)$ colors in total. Since $\omega(G_3) \le \omega(G_2) \le \lceil \omega(G)/3 \rceil \le \omega(G)/3 + 1$, the number of colors used by our algorithm for the graph G is at most

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

So, the performance ratio for our strategy is 7/5.

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

[8] 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)