*Computational Logic and Applications, CLA '05* DMTCS proc. **AF**, 2006, 61–68

# *On-line coloring of $I_s$-free graphs and co-planar graphs*

Iwona Cieślik, Marcin Kozik and Piotr Micek [†]

*Algorithmics Research Group,*
*Faculty of Mathematics and Computer Science,*
*Jagiellonian University*
*Gronostajowa 3, 30-387 Kraków, Poland*
*E-mail: icieslik@ii.uj.edu.pl*
*E-mail: kozik@ii.uj.edu.pl*
*E-mail: Piotr.Micek@tcs.ii.uj.edu.pl*

---

An on-line vertex coloring algorithm receives vertices of a graph in some externally determined order. Each new vertex is presented together with a set of the edges connecting it to the previously presented vertices. As a vertex is presented, the algorithm assigns it a color which cannot be changed afterwards. The on-line coloring problem was addressed for many different classes of graphs defined in terms of forbidden structures. We analyze the class of $I_s$-free graphs, i.e., graphs in which the maximal size of an independent set is at most $s-1$. An old Szemerédi's result implies that for each on-line algorithm $A$ there exists an on-line presentation of an $I_s$-free graph $G$ forcing $A$ to use at least $\frac{s}{2}\chi(G)$ colors. We prove that any greedy algorithm uses at most $\frac{s}{2}\chi(G)$ colors for any on-line presentation of any $I_s$-free graph $G$. Since the class of co-planar graphs is a subclass of $I_5$-free graphs all greedy algorithms use at most $\frac{5}{2}\chi(G)$ colors for co-planar $G$'s. We prove that, even in a smaller class, this is an almost tight bound.

**Keywords:** on-line algorithm, graph coloring, planar graph

---

A *graph* is a pair $G=(V,E)$ of sets such that $E \subseteq [V]^2$, i.e., the elements of $E$ are 2-element subsets of $V$. The elements of $V$ are *vertices* of the graph $G$, the elements of $E$ are its *edges*. For $\{x,y\} \in E$ vertices $x$, $y$ are said to be *adjacent*. A set $S \subseteq V$ is a *clique* if each two points in $S$ are adjacent. Dually, $S \subseteq V$ is an *independent set* if $S$ has no two adjacent points. We consider only finite graphs, i.e., $V$ is always finite. A *complement* of a graph $G=(V,E)$ is a graph $\overline{G}=(V,[V]^2 \setminus E)$, i.e., a graph containing precisely the edges missing in $G$. For other graph terminology we refer the reader, e.g., to [3].

A function $c: V \to \mathbb{N}$ is a *coloring* of the graph $G=(V,E)$ if for all $n \in \mathbb{N}$, $c^{-1}(n)$ is an independent set. The value $c(v)$ is a *color* of the vertex $v$. A function $cc: V \to \mathbb{N}$ is a *clique covering* of the graph $G=(V,E)$ if for all $n \in \mathbb{N}$, $cc^{-1}(n)$ is a clique. In such a case a number $cc(v)$ is a *clique number* of a vertex $v$. Note that a function $c: V \to \mathbb{N}$ is a coloring of graph $G=(V,E)$ if and only if it is a clique

---

[†]The author is a scholar of the project which is co-financed from the European Social Fund and national budget in the frame of The Integrated Regional Operational Programme.

covering of $\overline{G}$. The *chromatic number* of a graph $G$, i.e., the minimal number of colors needed to color $G$, is denoted by $\chi(G)$. The minimal number of cliques needed to cover $G$ is denoted by $v(G)$.

An on-line coloring of a graph may be viewed as a two person game. In each step of the game the first player, called *Spoiler*, presents one vertex of a graph together with edges connecting it to the previously presented vertices. A graph $G$ together with the order of presentation of its vertices is called an *on-line presentation* of $G$ (or just an *on-line graph*) and is denoted by $G^{<}$. As each vertex is presented, the second player, called *Algorithm*, assigns a color (a natural number) to it. This color cannot be changed afterwards. By an *on-line algorithm* we mean a strategy for Algorithm. Its goal is to find a graph coloring with the smallest possible number of colors. The goal of Spoiler is to force Algorithm to use as many colors as possible. A color assigned by the on-line algorithm $A$ to a vertex $v$ is denoted by $A(v)$. The set of colors used by $A$ to color vertices from $X \subseteq V$ is denoted by $A(X)$. The set of vertices colored by the algorithm $A$ with color $n$ is denoted by $A^{-1}(n)$. Finally, $\chi_A(G^{<})$ denotes the number of colors $A$ uses to color $G^{<}$. Analogously, we define an on-line clique covering game and an on-line clique covering algorithm $A$. Then $A(v)$ is a clique number assigned by $A$ to the vertex $v$. Similarly, we put $A(X)$, $A^{-1}(v)$ and $v_A(G^{<})$. Note that the algorithm that colors a graph finds a clique covering for presented in the same order complement of this graph and vice versa, the algorithm that covers a graph finds a coloring for presented in the same order complement of this graph. The most familiar on-line coloring (clique covering) algorithms are so called *greedy* algorithms, which try to assign – to the current vertex – a color (a clique number) already in use always when it is possible.

The performance of an on-line algorithm is measured by a function which compares its results to the optimal (off-line) results. An on-line coloring algorithm $A$ is *competitive with a function* $f$ [(i)] on a class of graphs $\Gamma$ if for all but finitely many $G \in \Gamma$ and for all its on-line presentations $G^{<}$, $\chi_A(G^{<}) \leqslant f(\chi(G))$. The class $\Gamma$ is $f$*–competitive* if there exists an on-line, competitive algorithm on $\Gamma$ with the function $f$. A class of graphs $\Gamma$ is *at least* $f$*–competitive* if for all $n$ there is a strategy for Spoiler building a graph with at least $n$ vertices and forcing Algorithm to use $f(\chi)$ colors where $\chi$ is the chromatic number of a given graph. Finally, $\Gamma$ is *exactly* $f$*–competitive* if it is at least $f$–competitive and $f$–competitive. Analogously, we may discuss the competitiveness of on-line clique covering algorithms.

The on-line graph coloring problem has been widely studied for various classes of graphs. The class of trees is not competitive (see [4]). This implies that classes containing all trees are not competitive, e.g., bipartite graphs, perfect graphs, etc. There are many examples of competitive classes: split graphs are exactly $\chi$-competitive (i.e., there is an on-line algorithm using optimal number $\chi(G)$ colors for all on-line split graphs $G^{<}$), complements of bipartite graphs are exactly $\frac{3}{2}\chi$–competitive and complements of chordal graphs are exactly $(2\chi - 1)$–competitive (all these results are in [4; 2]), interval graphs are exactly $(3\chi - 2)$–competitive [6].

The power of on-line coloring algorithms depends, to some extend, on the absence of certain induced subgraphs. A graph $G$ is called $H$*-free* if it does not contain an induced subgraph isomorphic to $H$. The following notation is used: $P_s$ - path with $s$ vertices, $I_s$ - independent set with $s$ vertices, $K_s$ - clique with $s$ vertices and $K_{s,t}$ - complete bipartite graph with parts of $s$ and $t$ vertices. Some classes defined in terms of forbidden substructures had been already studied. For example, it is known [4] that $P_4$-free graphs are exactly $\chi$-competitive and that $P_6$-free graphs are not competitive at all. There is a huge gap between known lower and upper bounds for on-line coloring of $P_5$-free graphs. Kierstead, Penrice and Trotter [8] have shown $(\frac{4^{\chi}-1}{3})$–competitiveness while the only known lowerbound is $\binom{\chi+1}{2}$ [2]. Even

---

(i) For example, we will simply use notations: $\chi$-competitive or $\binom{\chi}{2}$–competitive if $f(x) = x$ or $f(x) = \binom{x}{2}$, respectively.

narrowing this competitiveness gap for the substantially smaller class of $2K_2$-free graphs seems to be hard. Gyarfas and Lehel [4] reduced here the upper bound to $(2^{\chi} - 1)$, while [2] exhibits a quadratic lowerbound. Since trees are both $K_s$-free ($s \geqslant 3$) as well as $K_{s,t}$-free ($s, t \geqslant 2$), none of the classes of $K_s$-free graphs or $K_{s,t}$-free graphs is competitive. Cieślik [1] has shown that $K_{1,t}$-free graphs are exactly $((t-1)\chi - t + 2)$–competitive. Some general result on on-line coloring of $T$-free graphs, where $T$ is a tree, is established in [7].

This paper is devoted to determine the exact competitiveness of $I_s$-free graphs and the competitiveness of co-planar graphs. To warm-up observe that the class of $I_s$-free graphs is $(s-1)\chi$–competitive, by considering the on-line algorithm coloring each point with a new color. We claim that it uses at most $(s-1) \cdot \chi(G)$ colors for each presentation of an $I_s$-free graph $G = (V, E)$. Indeed, it uses exactly $|V|$ colors while $\chi(G) \geqslant \frac{|V|}{s-1}$ as at most $s-1$ points may have the same off-line color. We improve this trivial upper bound by showing that this class is exactly $\frac{s}{2}\chi$–competitive. First, the upper bound will be established for each on-line, greedy algorithm to be $\frac{s}{2}\chi$. We are indebted to revisor for reminding us an old Szemerédi's result which easily implies that the competitive function for $I_s$-free graphs is at least $\frac{s}{2}\chi$. We will shortly discuss it. In the second part we investigate co-planar graphs. A graph $G$ is *planar* if it can be drawn in the plane with its edges intersecting at their vertices only, in other words, without edge crossing. In particular, all trees are planar. As the class of trees is not competitive (see [4]) the class of planar graphs is also not competitive. The complements of planar graphs are called co-planar graphs. Since planar graphs are $K_5$-free, co-planar graphs are $I_5$-free. Therefore, from Theorem 1, greedy algorithms perform no worse than $\frac{5}{2}\chi$. We show that for any algorithm $A$ there exist a graph $G$ of arbitrarily large chromatic number for which $\chi_A(G^<) \geq \frac{5}{2}\chi(G) - 11.5$. We start with the analysis of the behavior of the greedy algorithm.

**Theorem 1** *Each on-line, greedy, coloring algorithm uses at most $\frac{s}{2}\chi(G)$ colors on every presentation of an $I_s$-free graph $G$. Thus, the class of $I_s$-free graphs is $\frac{s}{2}\chi$–competitive.*

**Proof:** Let $A$ be an on-line, greedy, coloring algorithm $A$ and $G^<$ be an on-line presentation of an $I_s$-free graph $G = (V, E)$. Consider an optimal coloring of $G$ and let $C_1, \ldots, C_{\chi(G)}$ be a partition of $V$ into monocolored, independent sets. Analogously, let $A_1, \ldots, A_{\chi_A(G^<)}$ be a partition determined by a coloring of $G^<$ produced by $A$. We say that an independent set $A_i$ ($1 \leqslant i \leqslant \chi_A(G^<)$) is *constrained* if there are at least two independent sets among $C_1, \ldots, C_{\chi(G)}$ containing at least one vertex from $A_i$, i.e., $A_i \cap C_k \neq \emptyset \neq A_i \cap C_l$ for some $k \neq l$. Inspecting the result of $A$ on $G^<$, i.e. $A_1, \ldots, A_{\chi_A(G^<)}$, we define

- $c$ = number of constrained $A_i$'s,
- $u$ = number of unconstrained $A_i$'s,
- $k_c$ = number of vertices in the join of all constrained $A_i$'s,
- $k_u$ = number of vertices in the join of all unconstrained $A_i$'s.

Obviously, $u \leqslant k_u$ and $c + u = \chi_A(G^<)$. Since each constrained $A_i$ has at least two points, we have $c \leqslant \frac{k_c}{2}$. The fact that $G$ is $I_s$-free yields that $G$ has at most $(s-1) \cdot \chi(G)$ vertices and therefore, $k_c + k_u \leqslant (s-1) \cdot \chi(G)$. Finally, since $A$ is greedy each $C_j$ contains at most one unconstrained $A_i$.

Thus, $u \leqslant \chi(G)$. After these remarks we are ready to bound the number of colors used by $A$ as follows

$$\begin{aligned}\chi_A(G^<) &= c+u \leqslant \frac{k_c}{2}+u \leqslant \frac{(s-1)\cdot\chi(G)-k_u}{2}+u \\ &\leqslant \frac{(s-1)\cdot\chi(G)}{2}+\frac{u}{2} \leqslant \frac{s\cdot\chi(G)}{2}.\end{aligned}$$

□

Before a general case, a lowerbound for on-line, greedy algorithms is provided.

**Proposition 2** *For each $s \geqslant 2$ there exists an on-line, $I_s$-free graph $G^<$ forcing all greedy on-line coloring algorithms to use $\frac{s}{2}\chi(G)$ colors.*

**Proof:** Consider an on-line bipartite graph $G^< = (V \cup U, E, <)$, where $V = \{v_1, \ldots, v_{s-1}\}$, $U = \{u_1, \ldots, u_{s-1}\}$ and $v_1 < u_1 < v_2 < u_2 < \ldots < v_{s-1} < u_{s-1}$. Each point $v_i$ is adjacent to all vertices in $U \setminus \{u_i\}$. Similarly, each point $u_i$ is adjacent to all vertices in $V \setminus \{v_i\}$. Moreover, $v_{s-1}$ is adjacent to $u_{s-1}$. Obviously, $G$ is $I_s$-free and $\chi(G) = 2$. It is also easy to check that each on-line, greedy algorithm $A$ is forced to use $s$ colors on $G^<$, as $A(v_i) = i = A(u_i)$ except $A(u_{s-1}) = s$. □

The fact that $I_s$-free graphs are at least $\frac{s}{2}$–competitive follows from Szemerédi's theorem concerned on an on-line antichain partitioning of partially ordered sets. A *partially ordered set* (an order) is a pair $P = (X, R)$ where $X$ is a set, and $R$ is a reflexive, antisymmetric, and transitive binary relation on $X$. A set $C \subseteq X$ is a *chain* in $P$ if $xRy$ for all $x, y \in X$. Dually, a set $A \subseteq X$ is an *antichain* in $P$ if $\neg xRy$ for all distinct $x, y \in X$. The *width* of $P$ is the maximal size of an antichain in $P$ and the *height* of $P$ is the maximal size of a chain in $P$. We say that a graph $G = (V, E)$ is a *comparability graph* of $P = (X, R)$ if $\{x, y\} \in E$ iff $xRy$ or $yRx$ for all $x, y \in V$. We may consider on-line orders and therefore on-line algorithms for (anti)chain partitioning of orders. For the terminology of orders as well as for an overview of an on-line partitioning problems we refer the reader to [10].

**Theorem 3 (Szemerédi [9] )** *For every $k \geqslant 1$ there is a strategy $S_k$ for presenting points of on-line order of height $k$ and width $k$ such that every on-line antichain partitioning algorithm uses at least $\binom{k+1}{2}$ antichains.*

Note that Theorem 3 may be easily extended to:

*For each $k, x \geqslant 1$ there is a strategy $S_{k,x}$ for presenting points of on-line order of height $k \cdot x$ and width $k$ such that every on-line antichain partitioning algorithm uses at least $\binom{k+1}{2} \cdot x$ antichains.*

Indeed, it suffices to repeat a strategy $S_k$ $x$ times in such a way that all vertices of $i$-th presented order are below all vertices of $j$-th one for all $1 \leqslant i < j \leqslant x$.

Observe also that each on-line coloring algorithm $A$ induces an on-line antichain partitioning algorithm $A'$ by coloring the comparability graph of the order presented as an input. If $P^<$ is an on-line order and $G^<$ is an on-line presentation of his comparability graph (an order of appearance of the the graph $G^<$ is naturally inherited from $P^<$) then the number of antichains used by $A'$ on $P^<$ is exactly the number of colors used by $A$ on $G^<$. On the other hand, height of the order $P$ is the size of the maximum clique in the comparability graph $G$. Therefore, since comparability graphs are perfect, height of $P$ is equal to $\chi(G)$. Moreover, width of $P$ is exactly equal to the number of the maximal independent set in $G$. Having these observations we immediately get.

**Proposition 4** *For each $s \geqslant 2$ and for each chromatic number $\chi \geqslant 1$ there is a strategy for Spoiler presenting on-line graph such that for each on-line coloring algorithm $A$ it produces an on-line $I_s$-free graph $G^<$ with $\chi(G) \geqslant \chi$ forcing $A$ to use at least $\frac{s}{2}\chi(G)$ colors.*

Proposition 4 implies that the class of $I_s$-free graphs is at least $\frac{s}{2}\chi$–competitive.

Now, we are going to investigate the competitiveness of on-line coloring for co-planar graphs. To show that no on-line algorithm can color a co-planar graph $G$ of arbitrarily large chromatic number (and thus of arbitrary large size) using less then $\frac{5}{2}\chi(G) - 11.5$ we produce a strategy for presenting appropriate on-line graphs. As we have already mentioned a graph is *planar* if it can be drawn in the plane without edge crossing. Such a drawing of a planar graph is called a plane drawing or a *plane graph*. A plane graph divides the plane into several regions called faces. In other words, a *face* in a plane graph is a subset of the plane bounded by a cycle without diagonal paths. For an overview of planar graphs we refer the reader to [3]. Description of co-planar graphs as complements of planar graphs may be cumbersome, so we are working with the on-line clique covering problem for planar graphs instead.

We present a strategy for Spoiler which, for any given on-line clique covering Algorithm $A$, produces a connected, planar, on-line graph $G^<$ of arbitrarily big clique covering number (and thus of arbitrary large size) and such that $v_A(G^<) \geq \frac{5}{2}(v(G) - 5) + 1$. The construction of $G^<$ is presented in *stages*. By an abuse of notation we denote by $G^<$, or simply $G$, a graph that is being presented by the Spoiler. For simplicity we denote $\left|A^{-1}(A(v))\right|$ by $\omega_A(v)$, note that at any step of construction $\omega_A(v) \leq 2$ for every vertex $v$. At the end of each stage the set of vertices of $G^<$ is a disjoin union of $\bigcup \mathcal{V}$, two sets of vertices $L_1, L_2$ (called *loose ends*) and a one element set $\{r\}$ ($r$ is called a *root* of $G$) such that:

- $\omega_A(r) = 1$ and
- for any $W \in \mathcal{V}$ the set $W$ is a clique in $G^<$ and $\sum_{v\in W} \frac{1}{\omega_A(v)} \geq \frac{5}{2}$,
- one of the following is true for each loose end:
  - the loose end is empty, or
  - the loose end consists of one element $v$, and $\omega_A(v) = 2$, or
  - the loose end consist of two vertices $u, v$ such that there is no edge in $G^<$ between $u$ and $v$, both $u$ and $v$ are adjacent to the same face of the graph $G^<$, $\omega_A(u) = 2$ and $\omega_A(v) = 2$.

Note that the sets $\{r\}, L_1, L_2, \mathcal{V}$ change (the use of this slightly imprecise notation allows us to simplify the presentation without leading to confusion).

A *configuration* of a graph $G^<$ is a triple consisting of a root and two loose ends and denoted by $[r, L_1, L_2]$. The following notation simplifies the construction: $\langle$vertices : edges$\rangle$ is a pair consisting of a set of vertices and a set of edges to be presented by Spoiler (in any order).

Each member of $\mathcal{V}$ is a clique thus, at the end of each stage, $v(G) \leq |\mathcal{V}| + 5$ (if a graph presented so far is $G^<$). At the same time

$$|A(V)| = \sum_v \frac{1}{\omega_A(v)} \geq \sum_{W\in\mathcal{V}} \sum_{v\in W} \frac{1}{\omega_A(v)} + \frac{1}{\omega_A(r)} \geq \frac{5}{2}|\mathcal{V}| + 1,$$

so certainly $v_A(G^<) \geq \frac{5}{2}(v(G) - 5) + 1$. Intuitively, the set $\mathcal{V}$ contains cliques such that the algorithm $A$ uses approximately $\frac{5}{2}$ “on-line cliques” to cover it.

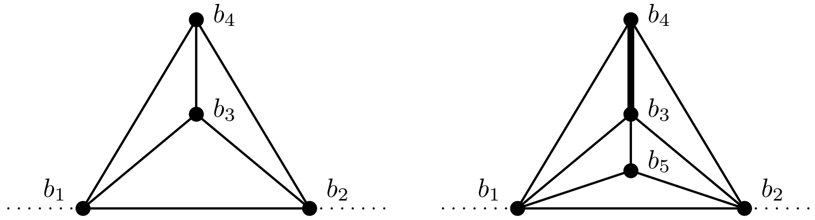


**Fig. 1:** Play schema

The following "pattern" appears very often in the strategy: Suppose that a graph contains two vertices $b_1, b_2$ connected by an edge, and that $A(b_1) \neq A(b_2)$. Moreover, there exist unique vertices $d_1, d_2$ such that $A(b_1) = A(d_1)$, $A(b_2) = A(d_2)$ and $b_1 \neq d_1$, $b_2 \neq d_2$. Under such conditions the following strategy (compare Figure 1) produces a loose end instead of these two vertices. Present two vertices $b_3, b_4$ and edges $\{b_1, b_3\}, \{b_1, b_4\}, \{b_2, b_3\}, \{b_2, b_4\}$. There are two possibilities: either $A(b_3) = A(b_4)$ or $A(b_3) \neq A(b_4)$. In either case $A(b_3)$ and $A(b_4)$ are numbers not used previously. If $A(b_3) \neq A(b_4)$ set $\mathcal{V} := \mathcal{V} \cup \{\{b_1, b_3, b_4\}\}$ (or $\mathcal{V} := \mathcal{V} \cup \{\{b_2, b_3, b_4\}\}$) and the single element set $\{b_2\}$ (or $\{b_1\}$) is a new loose end. If, on the other hand, $A(b_3) = A(b_4)$ present $b_5$ and edges $\{b_5, b_1\}, \{b_5, b_2\}, \{b_5, b_3\}$. The number $A(b_5)$ is a new number; set $\mathcal{V} := \mathcal{V} \cup \{\{b_1, b_2, b_3, b_5\}\}$ and $\{b_4\}$ is a new loose end.

An algorithmic description of one stage of the strategy for Spoiler follows. We assume that a graph presented so far is denoted by $G = (V, E)$ the configuration of the graph is $[r, L_1, L_2]$ and the set of "dealt with" cliques is $\mathcal{V}$.

**1:** **If** $L_1 \cup L_2 = \emptyset$ **then** present $\langle a : \{a, r\}\rangle$

**1.1:** **if** $A(a) = A(r)$ **then** present $\langle b : \{b, a\}\rangle$. Note that $A(b) \notin A(V)$, the new configuration is $[b, \{r\}, \{a\}]$ and this stage is **DONE**

**1.2:** **else** present $\langle b : \{b, r\}, \{b, a\}\rangle$

**1.2.1:** **if** $A(b) = A(a)$ (or dually $A(b) = A(r)$) **then** the new configuration is $[r, \{a\}, \{b\}]$ (or dually $[a, \{r\}, \{b\}]$) **DONE**

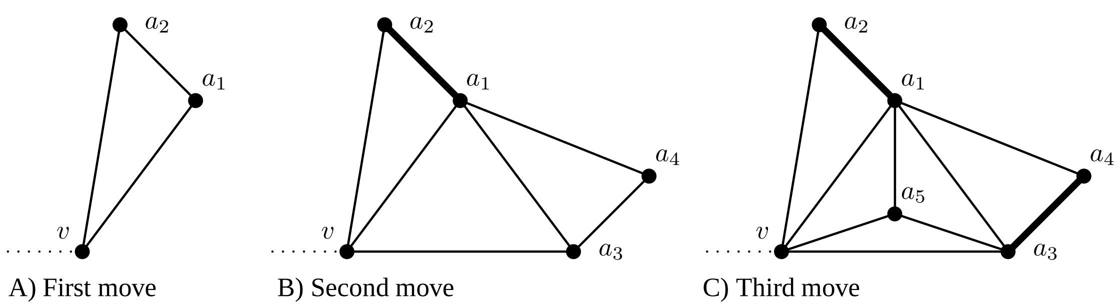


**Fig. 2:** Loose end type one

**1.2.2: else** $A(a) \neq A(b) \neq A(r)$, set $\mathcal{V} := \mathcal{V} \cup \{\{a, b, r\}\}$ and present $\langle c : \{c, r\}\rangle$. If $A(c) \neq A(r)$ the new configuration is $[c, \emptyset, \emptyset]$, if $A(c) = A(r)$ we present $\langle d : \{d, c\}\rangle$ with $A(d) \neq A(c)$ and the new configuration is $[d, \{c\}, \emptyset]$ **DONE**

**2: if** $|L_1| = 1$ or $|L_2| = 1$ **then** without loss of generality we may assume that $L_1 = \{v\}$ for some vertex $v$. In this case present $\langle a_1, a_2 : \{v, a_1\}, \{v, a_2\}, \{a_1, a_2\}\rangle$ (compare Figure 2). If $A(a_1) \neq A(a_2)$ then $\mathcal{V} := \mathcal{V} \cup \{\{v, a_1, a_2\}\}$ and the new configuration is $[r, \emptyset, L_2]$. If, on the other hand, $A(a_1) = A(a_2)$ present two more vertices $\langle a_3, a_4 : \{v, a_3\}, \{a_1, a_3\}, \{a_3, a_4\}, \{a_1, a_4\}\rangle$.

**2.1: if** $A(a_3) \neq A(a_4)$ **then** set $\mathcal{V} := \mathcal{V} \cup \{\{a_1, a_3, a_4\}\}$ and follow the schema presented in the pattern above with vertices $v, a_2$ in place of $b_1, b_2$. Depending on the choice of the algorithm put $\mathcal{V} := \mathcal{V} \cup \{\{v, b_3, b_4\}\}$ and the new configuration $[r, \{a_2\}, L_2]$ or $\mathcal{V} := \mathcal{V} \cup \{\{v, a_2, b_3, b_5\}\}$ and the new configuration $[r, \{b_4\}, L_2]$ **DONE**

**2.2: else** present $\langle a_5 : \{a_5, v\}, \{a_5, a_1\}, \{a_5, a_3\}\rangle$. Then $A(a_5) \notin A(V \cup \{a_1, a_2, a_3, a_4\})$. Set $\mathcal{V} := \mathcal{V} \cup \{\{v, a_1, a_3, a_5\}\}$ and the new configuration to be $[r, \{a_2, a_4\}, L_2]$ **DONE**

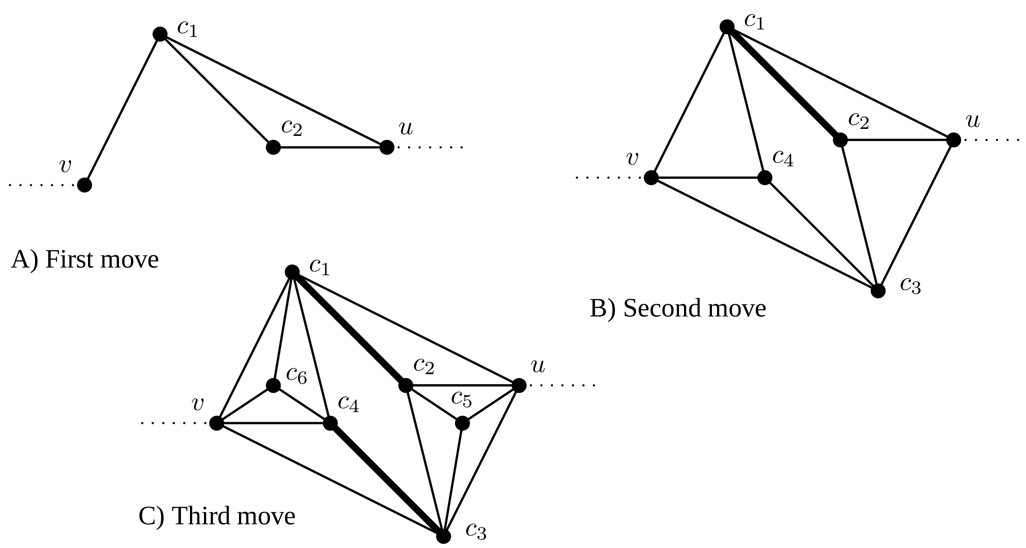


**Fig. 3:** Loose end type two

**3: if** $|L_1| = 2$ or $|L_2| = 2$ **then** without loss of generality we may assume that $L_1 = \{u, v\}$ for some vertices $u, v$. In this case present $\langle c_1, c_2 : \{v, c_1\}, \{u, c_1\}, \{u, c_2\}\rangle$ (compare Figure 3).

**3.1: if** $A(c_1) \neq A(c_2)$ **then** set $\mathcal{V} := \mathcal{V} \cup \{\{u, c_1, c_2\}\}$ and the new configuration to be $[r, \{v\}, L_2]$ **DONE**

**3.2: else** present $\langle c_3, c_4 : \{c_1, c_4\}, \{v, c_4\}, \{c_3, c_4\}, \{v, c_3\}, \{c_2, c_3\}, \{u, c_3\}\rangle$.

**3.2.1:** **if** $A(c_3) \neq A(c_4)$ **then** set $\mathcal{V} := \mathcal{V} \cup \{\{v, c_3, c_4\}\}$ and follow the pattern presented in the example with vertices $u, c_2$ taken to be $b_1, b_2$ (introduce new vertices in a face adjacent to $c_1$).

**3.2.1.1:** **if** $A(b_3) = A(b_4)$ **then** put $\mathcal{V} := \mathcal{V} \cup \{\{u, c_2, b_3, b_5\}\}$ and the new configuration to be $[r, \{c_1, b_4\}, L_2]$ **DONE**

**3.2.1.2:** **else** put $\mathcal{V} := \mathcal{V} \cup \{\{c_2, b_3, b_4\}\}$ and once more follow the pattern presented in the example. Take vertices $u, c_1$ to be $b_1, b_2$. Depending on the choice of the algorithm put $\mathcal{V} := \mathcal{V} \cup \{\{u, b_3, b_4\}\}$ and the new configuration $[r, \{c_1\}, L_2]$ or $\mathcal{V} := \mathcal{V} \cup \{\{u, c_1, b_3, b_5\}\}$ and the new configuration $[r, \{b_4\}, L_2]$ **DONE**

**3.2.2:** **else** present $\langle c_5, c_6 : \{c_5, u\}, \{c_5, c_2\}, \{c_5, c_3\}, \{c_6, v\}, \{c_6, c_1\}, \{c_6, c_4\}\rangle$. Certainly $A(c_5), A(c_6) \notin A(V \cup \{c_1, c_2, c_3, c_4\})$. Put $\mathcal{V} := \mathcal{V} \cup \{\{v, c_1, c_4, c_6\}, \{u, c_2, c_3, c_5\}\}$, new configuration is $[r, \emptyset, L_2]$ **DONE**

The strategy starts with presenting a single vertex $r$, the starting configuration is $[r, \emptyset, \emptyset]$ and $\mathcal{V} = \emptyset$.

This strategy is a way of constructing, for any given clique covering algorithm $A$, on-line presentation of planar graphs of arbitrarily large clique covering number such that $A$ uses approximately $\frac{5}{2}$-times too many cliques to cover such a graph. It shows that for co-planar graphs of sufficiently large chromatic number no coloring algorithm can perform essentially better than a greedy algorithm. More precisely

**Theorem 5** *For each on-line coloring algorithm $A$ and each chromatic number $\chi$ there exists an on-line presentation $G^<$ of a co-planar graph $G$ such that $\chi(G) \geq \chi$ and $\chi_A(G^<) \geq \frac{5}{2}\chi(G) - 11.5$.*

The construction immediately implies that a class of co-planar graphs is at least $(\frac{5}{2}\chi - 11.5)$–competitive.

# References


[1] I.Cieślik: On-line coloring and cliques covering for $K_{s,t}$-free graphs, Acta Informatica 42 (2005), 1-22.

[2] I.Cieślik: On-line graph coloring, PhD Thesis, Jagiellonian University, Kraków (2006).

[3] R.Diestel: Graph Theory, Springer, New York (1997).

[4] A.Gyárfás, J.Lehel: On-line and first fit colorings of graphs. Journal of Graph Theory 12 (1988), 217-227.

[5] A.Gyárfás, Z.Király, J.Lehel: Effective on-line coloring of $P_5$-free graphs. Combinatorica 11 (1991), 181-184.

[6] H.A.Kierstead, W.T.Trotter: An extremal problem in recursive combinatorics, Congressus Numerantium 33 (1981), 143-153

[7] H.A.Kierstead, S.G.Penrice, W.T.Trotter: On-line coloring and recursive graph theory. SIAM Journal of Discrete Mathematics 7 (1994), 72-89.

[8] H.A.Kierstead, S.G.Penrice, W.T.Trotter: On-line and first-fit coloring of graphs which do not induce $P_5$. SIAM Journal of Discrete Mathematics 8 (1995), 485-498.

[9] H.A.Kierstead: Recursive ordered sets. Contemp. Math. 57 (1986), 75-102.

[10] W.T.Trotter: Partially ordered sets. Handbook of combinatorics (1995), Vol. 1, 2, 433–480.