ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Faculté des Sciences de Base **(FSB)**
Institut de Mathématiques **(IMA)**
Chaire de Recherche Opérationnelle
Station 8
CH-1015 LAUSANNE

Tel +41.21.693.2566
Fax +41.21.693.5840

Online Coloring of Comparability Graphs: some results

M. Demange & B. Leroy-Beaulieu

ORWP 07/01
April 2007

Online Coloring of Comparability Graphs: Some Results

Marc Demange[1], Benjamin Leroy-Beaulieu[2]

[1] Département SID, ESSEC, Cergy Pontoise, France
[2] IMA-ROSE, Ecole Polytechnique Fédérale de Lausanne, Switzerland

April 4, 2007

Abstract. We study online partitioning of posets from a graph theoretical point of view, which is coloring and cocoloring in comparability graphs. For the coloring problem, we analyse the First-Fit algorithm and show a ratio of $O(\sqrt{n})$; furthermore, we devise an algorithm with a competitivity ratio of $\frac{\chi+1}{2}$. For the cocoloring problem, we point out a tight bound of $\frac{n}{4}+\frac{1}{2}$ and we give better bounds in some more restricted cocoloring problems.

1 Introduction

Graph coloring is one of the most studied topics in operations research. It consists in assigning a color to every vertex of a graph in such a way that two adjacent vertices have different colors. Graph coloring with a minimum number of colors is known to be NP-hard in general graphs.

Coloring problems have many applications in areas like logistics, scheduling, telecommunications and robotics. Such industrial applications make it natural to look at online versions of the problem, for which decisions must be taken while the full data are not known yet. In an *online problem*, the instance is presented step by step. Whenever a new part is presented, the solution dealing with this part is irrevocably decided by an *online algorithm*. In this context, we refer to the usual version of the problem, where the instance is fully known in advance, as the "offline" version.

Online graph coloring has been widely studied in general graphs [11, 13, 16]. The results obtained so far point out that only very loose bounds can be achieved. It is thus natural to look at particular classes of graphs for which coloring is known to be easy (polynomial) in the offline case. Some classes have already been studied in online cases [1, 3, 9, 17]. In this paper, we consider two classes of graphs, namely comparability graphs (graph representation of partially ordered sets)[3] and permutation graphs (graphs of inversions in permutations).

Furthermore, we also study a generalization of the coloring problem, called *cocoloring*. Both online coloring and cocoloring in these two classes of graphs have applications in industrial contexts [6, 19, 20].

[3] All notions will be formally defined in section 1.1

This work can be seen as online partitioning posets into antichains (or into chains and antichains). Several online problems have been studied on posets (See for instance [4]). In particular, online partitioning posets into chains has been widely studied [7, 12, 13]. From a graph theoretical point of view, it consists in online coloring cocomparability graphs. To our knowledge, online coloring of comparability graphs has been essentially studied in the case of permutation graphs [18, 21].

1.1 Definitions and notations

A *graph* (directed graph, also called digraph) G consists of a finite set V and an irreflexive binary relation on V. V is called the set of *vertices*. The binary relation is represented by a collection E of ordered pairs, called *arcs* or *directed edges*. We then denote $G = (V, E)$. If $(v, w) \in E$, then v and w are said to be *adjacent*.

Let $G = (V, E)$ be a graph. The graph $G^{-1} = (V, E^{-1})$ is said to be the *reversal* of G, where $E^{-1} = \{(v, w)|(w, v) \in E\}$. Given $V' \subseteq V$, the *subgraph induced by V' in G* is $G[V'] = (V', E')$, where $E' = \{(v, w) \in E|v \neq w \text{ and } \{v, w\} \subseteq V'\}$.

If the relation is symmetric ($E = E^{-1}$), then the graph is called *undirected*. In this case, arcs are usually called *edges* and are represented by an unordered pair of vertices. If the relation is asymmetric ($E \cap E^{-1} = \emptyset$) then the graph is called *oriented*.

Given an undirected graph $G = (V, E)$, a *partial subgraph* $H = (V, F)$ with $F \subseteq E$ is called an orientation of G if H is oriented ($F \cap F^{-1} = \emptyset$) and if $F \cup F^{-1} = E$. Moreover, if the relation associated to F is transitive, then H is called a *transitive orientation* of G. The graph $\overline{G} = (V, \overline{E})$ is called the *complement* of G, where $\overline{E} = \{(v, w) \in V \times V|v \neq w \text{ and } (v, w) \notin E\}$.

A *stable set*[4] of a given graph $G = (V, E)$ is a set $S \subseteq V$ such that $\forall (v, w) \in S^2,\ (v, w) \notin (E \cup E^{-1})$. A *clique*[4] is a set $K \subseteq V$ such that $\forall (v, w) \in K^2,\ (v, w) \in (E \cup E^{-1})$.

A *proper coloring*[4] of a graph, sometimes simply called a *coloring*, consists in a partition of V into stable sets. Each stable set is associated to a color. The stable sets in a coloring are often called *color classes* or simply *colors*. A k-coloring is a coloring that uses at most k stable sets. The *chromatic number* $\chi(G)$ of a graph G is the smallest number k such that G admits a k-coloring.

The *clique-size*[4] of a graph $G = (V, E)$ is the size of a largest clique $K \subseteq V$ and is denoted $\omega(G)$. For convenience, if there is no ambiguity, we will note χ and ω instead of $\chi(G)$ and $\omega(G)$.

Clearly, for every graph G, $\chi(G) \geq \omega(G)$. G is a *perfect graph*[4] if, for all $V' \subseteq V$, $\chi(G[V']) = \omega(G[V'])$ [2].

Cocoloring a graph $G = (V, E)$ is partitioning V into cliques and stable sets. The *cochromatic number* of a graph G, denoted by $z(G)$, is the smallest number of such sets needed to cover all vertices. A *k-cochromatic* graph is a graph G

[4] This notion is usually known for undirected graphs. We give this definition for convenience to use it in directed graphs.

such that $z(G) = k$ and a *k-cocolorable graph is a graph G such that* $z(G) \leq 2$. Cocoloring was introduced in [15].

A *comparability graph*[4] is an undirected graph which admits a transitive orientation. Every transitive orientation of a comparability graph can be seen as the graph representation of some partially ordered set (poset), where the vertices represent the elements of the set and there is an arc from a to b if $a < b$ and no arc between a and b if these elements are not comparable. Clearly, in comparability graphs, a clique of order n corresponds to a *path*[5] of length n. It corresponds in the related poset to a chain. Moreover, a stable set in a comparability graph corresponds to an antichain in the related poset. Consequently, coloring (resp. partitioning into cliques) a comparability graph is equivalent to partitioning a poset into antichains (resp. chains). Cocoloring a comparability graph is equivalent to partitioning a poset into chains and antichains.

A *permutation graph* is an undirected graph for which there exists a permutation such that every vertex of the graph corresponds to an element in the permutation and two vertices are adjacent if and only if the corresponding elements appear in reverse order in the permutation. Note that permutation graphs are comparability graphs. More precisely, a permutation graph is a comparability graph, the complement of which is also a comparability graph (cocomparability graph) [8]. Consequently, every hardness result stated for permutation graphs also holds for comparability graphs; conversely, every competitive analysis on comparability graphs also holds for permutation graphs. Coloring a permutation graph is equivalent to partitioning a permutation into increasing subsequences. Cocoloring a permutation graph is equivalent to partitioning a permutation into monotone subsequences; in some works, cocoloring a permutation is called monotone partitioning [21].

Both permutation graphs and comparability graphs are perfect graphs. Coloring such graphs offline can be done optimally in polynomial time [8]. On the other hand, cocoloring permutation graphs (and consequently comparability graphs) is NP-hard [22].

In this work, we consider online versions of the problems of coloring and cocoloring comparability graphs. An online problem can be seen as a two player game involving an adversary and an algorithm. The adversary presents the instance and the algorithm gives the solution. The online problem is generally characterized by the underlying offline problem and two sets of rules that have to be respected by the adversary and the algorithm, respectively. In our online model, the graph is presented together with a transitive orientation; consequently, this work can be seen as online partitioning of posets into antichains (or into chains and antichains). The vertices are presented one by one, together with the arcs connecting them to the vertices already presented. In some cases, we will specify further rules to define a particular case of the problem. At each step, one has to irrevocably assign a color to the new vertex.

[5] In graph theory, arcs of a chain are not necessarily all oriented in the same direction. To avoid ambiguity with chains in posets, we will use the graph terminology "path" instead of "directed chain".

Online algorithms are traditionally evaluated according to their *performance ratio*. Let A be an online graph-coloring algorithm. Then $\chi_A(G)$ denotes the maximum number of colors A uses to color G over all online presentations of G respecting the given rules. An online algorithm is said to guarantee a performance ratio of $\rho(G)$ (or to be $\rho(G)$-competitive) if, for every graph G, $\chi_A(G) \leq \rho(G)\chi(G)$. An online algorithm is called *exact* (or solves the problem exactly) if it computes the optimal offline solution for any online instance (it has a competitivity ratio of 1). It is called *optimal* if its competitivity ratio can not be improved by any other online algorithm. Some authors [14] also use an alternative way to characterize the performance of an online algorithm: a class of graphs Γ is said to be χ-bounded if the performance ratio for Γ only depends on χ. It means that there exists a function f such that for all $G \in \Gamma$, $\chi_A(G) \leq f(\chi(G))$.

Throughout this paper, we will use a very common representation for permutation graphs, called the *lattice representation*. The permutation is represented in a two-dimensional plane: the y-axis represents the values of the elements of the permutation and the x-axis represents the position of these elements. A point (x, y) on the plan means that an element of value y is at position x in the permutation. Usually, x-axis and y-axis are discrete axes, with values going from 1 to n, where n is the size of the permutation. In this paper, we mostly consider the relaxation where the axes are continuous, called the *continuous latticial model*. This means that, whenever a new point is presented, we are given its position relatively to points previously presented along each axis but not its absolute position in the permutation. It is equivalent to present a permutation graph G together with a transitive orientation and also a transitive orientation of $\overline{G}$ or to present a sequence of numbers one has to decompose into monotone sequences. The related permutation can be computed when all elements are known.

In the last section, we mention the *discrete latticial model*, where the latticial representation is drawn in $\{1, \ldots, n\}^2$, n being the order of the graph. Given the lattice representation, for each point v, we define four regions of the plane, which are inspired from the four cardinal points: $NW(v)$, which is its upper-left corner, $NE(v)$, which is the upper-right corner, $SW(v)$, which is the lower-left corner and $SE(v)$, which is the lower-right corner. It is immediate to see that (v, w) is an arc if and only if $w \in SE(v) \Leftrightarrow v \in NW(w)$. We will also use this terminology to specify directions on the plane.

2 Coloring

2.1 Preliminaries

A very common algorithm for coloring graphs is the greedy algorithm *First-Fit*, denoted by FF. It considers the vertices one after the other and attributes to each one the smallest possible color. It is very popular for its simplicity and since, for some classes of graphs, it is easy to find an ordering which makes First-Fit exact. When dealing with First-Fit, one often refers to the graph $P_4 = (\{a; b; c; d\}, \{(a, b); (b, c); (c, d)\})$. It is the only minimal configuration for

which First-Fit may find a non-optimal coloring. More precisely, if the adversary presents the vertices in the order (a, d, b, c), then, First-Fit uses three colors while two are sufficient. On the other hand, the class of graphs without induced P_4, called *cographs*, is known to be characterized by the fact that First-Fit will find an optimal coloring no matter the order in which it takes the vertices [5]. It is simple to see that P_4 is a permutation graph. Thus, permutation graphs do not have the nice property that any order is suitable for First-Fit. However, for some classes of perfect graphs, including comparability graphs (and consequently permutation graphs) and interval graphs (intersection graph of a set of intervals), such a suitable ordering is easy to compute offline [8].

In an online framework, First-Fit is also a very natural algorithm, but its behavior depends on the order of presentation of the vertices. While it is exact for some classes of graphs, like cographs [5], independently of the order of presentation of the vertices, it can be very bad for other classes of graphs and in particular for permutation graphs [18]. In this last paper it is indeed shown that First-Fit is not χ-bounded since for any integers $\chi > 0$ and k, there exists a permutation graph G such that $\chi(G) = \chi$ and $\chi_{\mathrm{FF}}(G) \geq \frac{1}{2}((\chi^2+\chi)+k(\chi^2-\chi))$.

Remark 1. First-Fit can trivially be adapted to partitioning a graph into cliques (we denote by FF_k the adapted algorithm) and also to cocoloring (FF_z).

In this section, we define a way to present a permutation graph in a given direction and characterize the directions making First-Fit optimal. Then, in section 2.2, we propose an analysis of First-Fit for bipartite permutation graphs.

Given the lattice representation of a permutation graph G, it is well known that it can be colored optimally if the vertices are presented from west to east, since First-Fit colors G exactly using this order [8]. This order of presentation corresponds to presenting the permutation from left to right. Considering this, one may wonder whether it is easy to color online a permutation graph if the vertices are presented in some other direction. It can be presented, for instance, from west to east, from south-west to north-east and so on. More precisely, given a fixed direction $\overrightarrow{u} \in \mathbb{R}^2$, the graph is said to be presented in the direction $\overrightarrow{u}$ if

$$\overrightarrow{OA} \cdot \overrightarrow{u} < \overrightarrow{OB} \cdot \overrightarrow{u} \Rightarrow A \text{ is presented before } B$$

If $\overrightarrow{OA} \cdot \overrightarrow{u} = \overrightarrow{OB} \cdot \overrightarrow{u}$, then A may be presented before or after B.

Proposition 1. *If a permutation graph G is presented from north-west to south-east or from south-east to north-west ($\overrightarrow{u} = (x, y)$ such that $x \cdot y \leq 0$ and $(x, y) \neq (0, 0)$) on a latticial model, then First-Fit colors G optimally.*

Proof. This proof is inspired from Chvátal's proof [5]. Let us suppose $x \geq 0$, $y \leq 0$ and $(x, y) \neq (0, 0)$. Then, for all vertices M and M' such that $M' \in NW(M))$ and $M \neq M'$, M' is presented before M.

Let us suppose that the color k is attributed to M_k. Then, there exists a point M_{k-1} with color $(k-1)$ such that M_{k-1} is presented before M_k and the related vertices in G are linked. Thus, $M_{k-1} \in NW(M_k)$.

By the same argument, we show that there exist M_i, $1 \leq i \leq k-1$, where M_i is of color i and $M_i \in NW(M_{i+1})$.

Then $\{M_i\}, i \in \{1, \ldots, k\}$ constitute a clique of order k, which concludes the proof. ■

Proposition 2. *If a permutation graph G is presented from south-west to north-east or from north-east to south-west ($x \cdot y > 0$) on a latticial model, then no algorithm can guarantee an optimal coloring for any arbitrary comparability graph G, even if G is a P_4.*

Proof. Let us suppose $x = 1$ and $y = 1$. The proof is similar for other cases. Let us present $M_1 = (1, 3)$ and $M_2 = (4, 1)$. M_1 is colored with color 1 and M_2 with color 2. $M_3 = (5, 4)$ is presented. We consider three cases.

1. If M_3 is colored with color 3 then three colors are used already.
2. If M_3 is colored with color 1 then $M_4 = (3, 8)$. It must be colored with 3.
3. If M_3 is colored with color 2 then $M_4 = (9, 2)$. It will also be colored with 3.

In all cases, the graph presented is a bipartite P_4 and is colored with at least 3 colors. ■

Corollary 1. *(proposition 1) If the graph is presented from west to east, from east to west, from south to north or from north to south, then First-Fit solves it exactly.*

Let us conclude the preliminaries with some remarks about symmetries. Consider two permutation graphs $G = (V, E)$ and $G' = (V', E')$, where the latticial representations of G and G' are symmetric to each other with respect to the x-axis (or y-axis). For any vertex v of G, let v' be its symmetric vertex in G'. Clearly, $(u, v) \in E \Leftrightarrow (u', v') \notin E'$. Thus, G' is the complement of G.

Consider now a permutation graph $G'' = (V'', E'')$ obtained from G by symmetry respectively to the axis given by the vector $\overrightarrow{u} = (1, 1)$. Clearly, G'' is isomorphic to G.

Thus, we can deduce proposition 3.

Proposition 3. *Any algorithm for coloring permutation graphs presented in direction $\overrightarrow{u} = (x, y)$ is equivalent to an algorithm for partitioning permutation graphs presented in direction $\overrightarrow{u}' = (-x, y)$ (or $(x, -y)$) into cliques. It is also equivalent to an algorithm for coloring permutation graphs presented in direction $\overrightarrow{u}'' = (-x, -y)$.*

We then deduce from propositions 1, 2 and 3:

Corollary 2. *A permutation graph presented from south-west to north-east or from north-east to south-west ($\overrightarrow{u} = (x, y)$ with $xy \geq 0$ and $(x, y) \neq (0, 0)$) can be partitioned into cliques exactly by FF_k while no online algorithm can partition exactly permutation graphs presented from south-east to north-west or from north-west to south-east.*

2.2 Competitive analysis of First-Fit

In this section, we focus on the competitivity ratio of First-Fit. We first note that it achieves a competitivity ratio of $\frac{n}{4} + \frac{1}{2}$ for general graphs and that this bound is tight even for bipartite graphs. We then focus on bipartite permutation graphs and show that First-Fit is $O(\sqrt{n})$-competitive for this class.

First-Fit is known to guarantee a ratio close to $\frac{n}{4}$ for general graphs. Indeed, Miller [17] shows that a ratio lower than $\frac{n}{4}$ can not be achieved even for bipartite graphs and the $\frac{n}{4}$-competitiveness is mentioned by Lovász et al. [16]. Since we did not find a proof for this claim in the literature, we give it here, and take this opportunity to slightly precise this result.

Proposition 4. (See [16, 17]) *First-Fit guarantees a competitivity ratio of $\frac{n}{4}+\frac{1}{2}$ for every graph of order n and this bound is tight even for bipartite graphs.*

Proof. Let us first note that a deep reading of Miller's proof [17] shows that the performance ratio of First-Fit can be as bad as $\frac{n}{4} + \frac{1}{2}$ for bipartite graphs.

We prove now that the claimed competitivity ratio is guaranteed. Let us consider an online instance consisting in a graph G of order n presented vertex by vertex in an arbitrary order.

Note first that if $\chi(G) = 1$, then First-Fit is exact. So we can assume that $\chi(G) \geq 2$.

Let k be the number of colors containing only one vertex. First-Fit is conceived in such a way that the related vertices constitute a clique of order k and consequently $\chi(G) \geq k$.

On the other hand, the number of colors used by First-Fit is at most $k + \frac{n-k}{2} \leq \frac{n}{2} + \frac{k}{2}$. Consequently, $\rho(G) \leq \frac{n}{2\chi(G)} + \frac{k}{2\chi(G)} \leq \frac{n}{4} + \frac{1}{2}$, where $\rho(G)$ is the competitivity ratio of First-Fit. ■

Since the negative result already holds for bipartite graphs, we focus in our next analysis on bipartite permutation graphs.

Theorem 1. *The competitivity ratio of First-Fit for the online coloring of a bipartite permutation graph is $O(\sqrt{n})$ and this bound is tight, even if we impose a direction of presentation $\overrightarrow{u} = (x, y)$, $xy > 0$.*

Proof. We start by proving that $O(\sqrt{n})$ is an upper bound for the competitivity ratio of First-Fit using lemma 1. We then prove that the given bound is tight using an adversary called FFAD.

Let us first analyze the competitivity ratio of First-Fit for bipartite permutation graphs. Consider a bipartite permutation, that is a permutation which can be partitioned into two increasing subsequences. Let S_1 and S_2 be the two colors obtained by applying First-Fit in the direction $\overrightarrow{u} = (1, 0)$ on this permutation. Recall that in this case, S_1 and S_2 represent an optimal coloring of this graph. Thus, for every arc (x, y) of the associated permutation graph, we have $x \in S_1$ and $y \in S_2$.

An increasing subsequence $(x_i)_i$ is *alternating* with respect to S_1 and S_2 if $[(x_{2i})_i \in S_1$ and $(x_{2i+1})_i \in S_2]$ or $[(x_{2i})_i \in S_2$ and $(x_{2i+1})_i \in S_1]$.

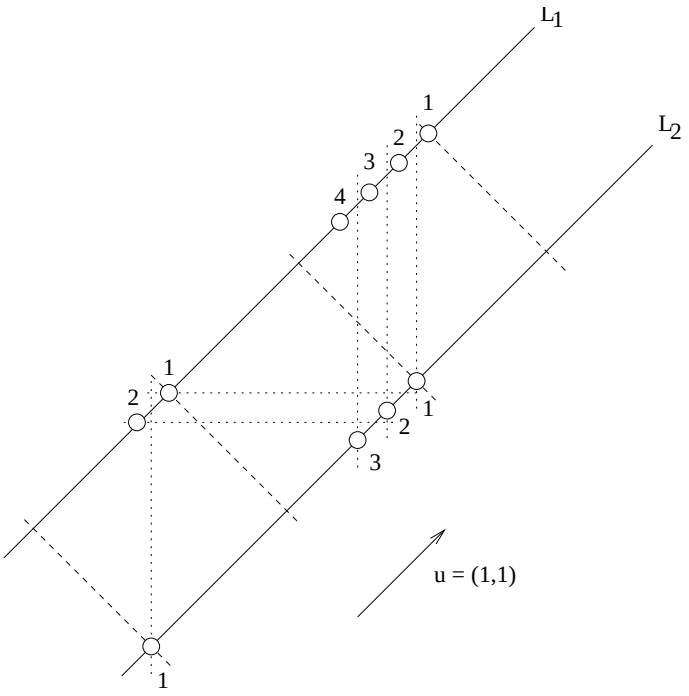

Fig. 1. This figure illustrates the principle of FFAD, presented on page 9. The vertices are presented from South-West to North-East ($u = (1, 1)$). The numbers close to them represent their colors attributed by First-Fit. The dotted lines are level-lines representing the x or y coordinates of the vertex they cross. The only purpose of these lines on this figure is helping to see whether a given vertex is to the left or the to right of some other vertex. The dashed lines show that each group of vertices with k =constant is presented after the group of vertices with $k-1$ in the direction of u.

Lemma 1. *Suppose that First-Fit, applied online to the permutation, colors with the color $k \geq 3$ an alternating increasing subsequence of size P. Then First-Fit must have colored with color $(k-1)$ an alternating increasing subsequence of size P and with color $(k-2)$ an alternating increasing subsequence of size $P+1$.*

Proof. Without loss of generality, we can suppose $k = 3$. Suppose $P = 2p$ (The case $P = 2p + 1$ is similar) and let $x_1, \ldots, x_{2p}$ be the increasing alternating subsequence of size P colored with color 3. Without loss of generality, we can suppose $x_{2i+1} \in S_1$, $i = 0, \ldots, p-1$ and $x_{2i} \in S_2$, $i = 1, \ldots, p$ (in the other case, the proof is identical).

Let $SE(x)$ be the south-eastern part of the plan associated to x, that is the set of successors of the vertex x in the permutation graph associated to the permutation, and $NW(x)$ be the north-western part of the plan associated to x, that is the set of predecessors of the vertex x in the permutation graph associated

Algorithm 1 FFAD

Draw two parallel imaginary lines on a plane, call them L_1 and L_2, such that L_1 is above L_2 and their direction vector has two positive components. These lines represent the colors of the vertices in an optimal offline coloring.
$k := 1$
$j := 1$
Present a vertex $v_{(k,j)}$ on L_2. It will be colored with color 1.
for $k := 2..K$ **do**
  $j := k$
  if $v_{(k-1,j-1)}$ is on L_2 **then**
    Present $v_{(k,j)}$ on L_1 such that $v^x_{(k,j)} = v^x_{(k-1,j-1)} - \varepsilon$
  else /*$v_{(k-1,j-1)}$ *is on* L_1 */
    Present $v_{(k,j)}$ on L_2 such that $v^y_{(k,j)} = v^y_{(k-1,j-1)} - \varepsilon$
  end if
  for $j := (k-1)..2$ **do**
    if $v_{(k-1,j-1)}$ is on L_2 **then**
      Present $v_{(k,j)}$ on L_1 such that $v^x_{(k-1,j)} < v^x_{(k,j)} < v^x_{(k-1,j-1)}$
    else /*$v_{(k-1,j-1)}$ *is on* L_1 */
      Present $v_{(k,j)}$ on L_2 such that $v^y_{(k-1,j)} < v^y_{(k,j)} < v^y_{(k-1,j-1)}$
    end if
  end for
  $j := 1$
  if $v_{(k-1,j)}$ is on L_2 **then**
    Present $v_{(k,j)}$ on L_1 such that $v^x_{(k,j)} = v^x_{(k-1,j)} + \varepsilon$
  else /*$v_{(k-1,j)}$ *is on* L_1 */
    Present $v_{(k,j)}$ on L_2 such that $v^y_{(k,j)} = v^y_{(k-1,j)} + \varepsilon$
  end if
end for

to the permutation. We have that

$$\forall i \;\; \exists y_{2i+1} \in SE(x_{2i+1}),\; y_{2i+1} \text{ colored 2},\; y_{2i+1} \in S_2$$

$$\forall i \;\; \exists y_{2i} \in NW(x_{2i}),\; y_{2i} \text{ colored 2},\; y_{2i} \in S_1$$

Thus,

$$y_{2i+1} <^{(a)} x_{2i+1} <^{(b)} x_{2i+2} <^{(c)} y_{2i+2} <^{(d)} y_{2i+3}$$

(a) Because $y_{2i+1} \in SE(x_{2i+1})$.
(b) Because (x_i) is increasing.
(c) Because $y_{2i+2} \in NW(x_{2i+2})$.
(d) Because y_{2i+3} is on the right of x_{2i+3}, thus to the right of y_{2i+2}; and (y_i) is increasing, since all its elements have the same color.

Thus, the $(y_j)_j$ are all different and form an increasing alternating subsequence colored with color 2, $y_{2i} \in S_1$ and $y_{2i+1} \in S_2$.

Similarly, the existence of the subsequence (x_i) proves the existence of an increasing alternating subsequence (z_k) of size P and of color 1 with $z_{2i} \in S_1$ and $z_{2i+1} \in S_2$.

Similarly again, the existence of the subsequence (y_j) proves the existence of an increasing alternating subsequence (z'_k) of size P and of color 1 with $z'_{2i} \in S_2$ and $z'_{2i+1} \in S_1$.

In addition, $z_1 \neq z'_1$ because $z_1 \in S_2$ and $z'_1 \in S_1$.

If $z_1 < z'_1$, the subsequence $z_1, z'_1, \dots z'_{2p}$ is increasing, alternating and of size $P+1$. If $z'_1 < z_1$, then the subsequence $z'_1, z_1, \dots z_{2p}$ is increasing, alternating and of size $P+1$.

This ends the proof of lemma 1 for the case $P = 2p+1$. The proof is similar if $P = 2p$. □

To conclude the competitive analysis, we distinguish two cases:

1. If First-Fit colors the permutation with two colors, the competitivity ratio is one.
2. If First-Fit uses $k \geq 3$ colors, then by an iterative application of lemma 1, the number n of vertices is such that $n \geq \lceil \frac{k}{2} \rceil \left(\lceil \frac{k}{2} \rceil + 1 \right) \geq (\frac{k}{2})^2$.

Thus $k \leq 2\sqrt{n}$. This inequality still holds for $k = 2$. This proves that the competitivity ratio of $\sqrt{n}$ is guaranteed.

In order to conclude the proof of theorem 1, let us use the adversary FFAD, illustrated in figure 1 and presented on page 9, to prove that the bound is tight. In this algorithm v^x and v^y represent the x-coordinate (resp. the y-coordinate) of a vertex v, while the index (k, j) is a numbering of the vertices. This proof is given here for the direction $\overrightarrow{u} = (1, 1)$; by a simple rotation, one can easily see that it holds for any $\overrightarrow{u} = (x, y)$, $x > 0$, $y > 0$.

For any k and any j, the vertex $v_{(k,j)}$ is colored with color j because it is the smallest color that is admissible in this region. Thus, at each step k, one new color is used. Let c_k be the total number of colors used when the step k is reached. We have $c_k = k$.

Also, at each step k we add k vertices, the number n_k of vertices used is thus $n_k = n_{k-1} + k$, which induces that $n_k = \frac{k(k+1)}{2}$. So, if c_k is the number of colors used at step k, we have $c_k = O(\sqrt{n_k})$. This is true in particular at the last step: let the total number of colors used be C and the total number of vertices be n, we have $C = O(\sqrt{n})$ and this concludes the proof of theorem 1. ■

By remark 1 and proposition 3, we get:

Corollary 3. *The competitivity ratio of FF_k for online partitioning a co-bipartite (partitionable into two cliques) permutation graph is $O(\sqrt{n})$ and this bound is tight, even if we impose a direction of presentation $\overrightarrow{u} = (x, y)$, $xy < 0$.*

Remark 2. For the bipartite case, the result of Nikolopoulos and Papadopoulos [18] states that online coloring does not admit a constant competitive ratio. Our result improves it to $O(\sqrt{n})$.

2.3 Competitive ratio of online coloring of comparability graphs

Given a comparability graph, it is well known that an optimal k-coloring $S_1, \ldots, S_k$ can be found such that, for every arc (v, w), $v \in S_i$ and $w \in S_j$ with $i < j$ [8]. We call such a coloring an *increasing coloring*. This notion will be useful to understand the algorithm used in the proof of the following theorem.

Theorem 2. *There exists an online algorithm for coloring comparability graphs (presented with a transitive orientation) guaranteeing a competitivity ratio of* $\frac{\chi+1}{2}$ *and this bound is tight, even for permutation graphs presented in a continuous latticial model.*

Proof. Let us consider the algorithm OCC that computes a proper coloring for a comparability graph presented vertex by vertex together with a transitive orientation.

Colors used by OCC are named by ascending sequences of consecutive integers. If $C(v)$ is the color of a particular vertex v, we will call $C_r(v)$ the right-most integer of $C(v)$ and $C_l(v)$ the left-most integer of $C(v)$. Sometimes, when there is no ambiguity on the vertex, we might just note C, C_l and C_r. Note that C is a color, while C_l and C_r are integers. The principle of the algorithm is the following: at each online step, the color assigned to the vertex presented is the sequence of integers corresponding to the possible colors the vertex can be assigned in every optimal increasing coloring in the already presented graph. The current coloring is also renamed in such a way that the sequence used to color every vertex always contains the possible colors in an optimal increasing coloring of the current graph.

Algorithm 2 Online Comparability Graph Coloring (OCC)

Input: A comparability graph G delivered online, vertex by vertex.
Output: A $\frac{\chi(\chi+1)}{2}$-coloring of G, where χ is the chromatic number of G. χ is unknown before the end of the algorithm.

1: **while** G is not completely presented **do**
2: Let G' be the subgraph of G induced by the currently presented vertices. Define $k := \chi(G')$ and accept a new vertex v^*.
3: In the graph defined by $G' \cup \{v^*\}$, consider a longest path K containing v^*. Let $l = |K|$.
4: If $l > k$ rename all the currently attributed colors as indicated here: for each vertex v, rename its color by concatenating the name $C(v)$ of its color with $C_r(v) + 1$ on the right. Increment k by 1.
5: Let p be the rank of v^* in K. Attribute to v^* the color $p \ldots (k - l + p)$.
6: **end while**

Note that, at each step, OCC has to compute a longest path containing v^* in a comparability graph. This can be done in polynomial time by computing a maximum clique in the neighborhood of v^*. The complexity of steps 2 and 3 is

$O(|V'| + |E'|)$ each [8]. Thus, the whole complexity of OCC at each online step is also $O(|V'| + |E'|)$.

The proof of theorem 2 is based on three lemmas.

Lemma 2. *If any vertex v in the comparability graph is part of two different paths, both of maximum length, then v will hold the same rank in both paths.*

Proof. Assume v has not the same rank in both paths. Let K_1 be the path where v has the smallest rank and K_2 be the other path. We can make a new path consisting of all elements of K_2 preceding v, v and all elements of K_1 with a rank larger than v. This path will be longer than both K_1 and K_2, which is in contradiction. □

Lemma 3. *For every vertex v with color C, there exists a path with length $\chi(G') + C_l - C_r$, where v is exactly at rank C_l.*

Proof. This is obviously true when v is first assigned a color. Afterwards, every time $\chi(G')$ increases by one, C_r also increases by one by step 4 of the algorithm. So the sequence which was used in step 3 will always verify this property. □

Lemma 4. *OCC computes a proper coloring.*

Proof. Let v^* be the last introduced vertex. Let v be a vertex with the same color as v^*; we have to show that v and v^* are not in a same path.

Else, let us first suppose that v is before v^* on some path K. By lemma 3, v and v^* both belong to a path (K_v, K_{v^*} respectively) of length $\chi(G') + C_l(v^*) - C_r(v^*)$ at position $C_l(v^*)$. Moreover, K_{v^*} is a path containing v^* of maximum length in the already presented instance.

The path consisting of the elements of K_v preceding v, v, v^* and the elements of K_{v^*} with a rank larger than v^* would be longer than K_{v^*}, which is a contradiction.

A similar argument holds if v is after v^* on K. □

To conclude the competitivity analysis of OCC, we have to note that the number of colors produced by OCC is at most $\frac{\chi(\chi+1)}{2}$. Indeed, the colors used by the algorithm are all subintervals of $[1, \ldots, \chi]$. There are exactly $\chi(\chi + 1)/2$ such subintervals.

Next, we show that the bound is tight, even for permutation graphs. We prove this by building an adversary called OCCAD, illustrated in figure 2.

OCCAD presents a graph G in a continuous latticial model, such that $\chi(G) = K$ and OCC will use all subintervals of $[1..K]$ as colors. This ends the proof of theorem 2. ■

Corollary 4. *(Theorem 2, by proposition 3) There exists an online algorithm for partitioning cocomparability graphs (presented with a transitive orientation of a complementary graph) into cliques guaranteeing a competitivity ratio of $\frac{k+1}{2}$ (where k is the offline optimum) and this bound is tight, even for permutation graphs. In particular, it gives an algorithm for partitioning permutations into decreasing sequences.*

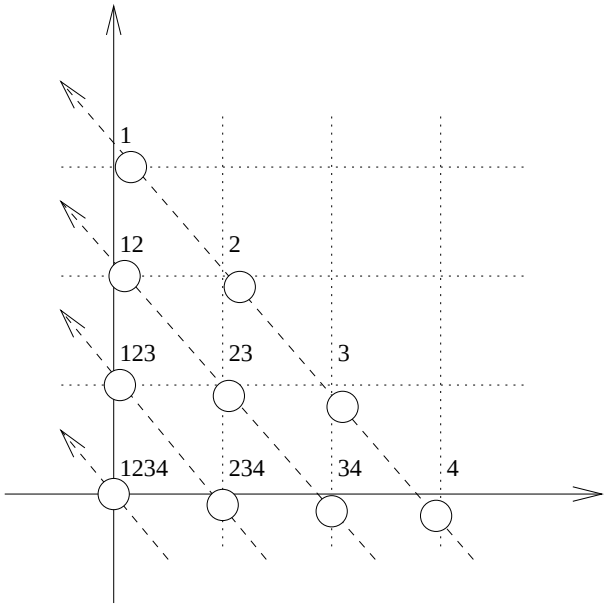

Fig. 2. Illustration of the algorithm OCCAD. Vertices are inserted in the order south-west to north-east, and on each line in the order given by the arrow.

Algorithm 3 OCCAD

1: **for** $k := 1..K$ **do**
2: **for** $x := k..0$ **do**
3: $y := k - x$
4: Put a vertex at coordinate $(x + \varepsilon_y, y - \varepsilon_x)$, where

$$\varepsilon_y := \begin{cases} 0 & \text{if } y = 0, \\ \varepsilon_{y-1} < \varepsilon_y < \varepsilon_{y-1} + \frac{1}{K} & \text{if } y \neq 0. \end{cases}$$

$$\varepsilon_x := \begin{cases} 0 & \text{if } x = 0, \\ \varepsilon_{x-1} < \varepsilon_x < \varepsilon_{x-1} + \frac{1}{K} & \text{if } x \neq 0. \end{cases}$$

5: **end for**
6: **end for**

Remark 3. Note that the class of cocomparability graphs is also known to be χ-bounded with an exponential binding function [12, 13]. Moreover, polynomial binding functions for cocomparability graphs is given in [7] with some restrictions on the order of presentation of the vertices. For the case of permutation graphs, which are comparability *and* cocomparability graphs, we achieve a polynomial binding function for a general quite online model.

Remark 4. Theorem 2 tells us that the class of comparability graphs is χ-bounded by the binding function $\frac{\chi(\chi+1)}{2}$. For comparability graphs with a bounded chromatic number, it leads to a constant competitivity ratio. In particular, OCC guarantees 3 colors for bipartite graphs, which is optimal by proposition 2, and 6 colors for 3-colorable comparability graphs. It is worth noting that the class of bipartite graphs (and thus also the class of comparability graphs) is known to be not χ-bounded [10] if the transitive orientation is not given. This hypothesis on the online model allows us to reduce the best known bound [17] for online coloring of bipartite graphs from $2\log_2 n$ to 3.

Remark 5.

Proposition 5. *No online algorithm guaranteeing 1 color on stable sets and 3 colors on bipartite permutation graphs (given with a transitive orientation) can guarantee less than 6 colors for 3-colorable permutation graphs.*

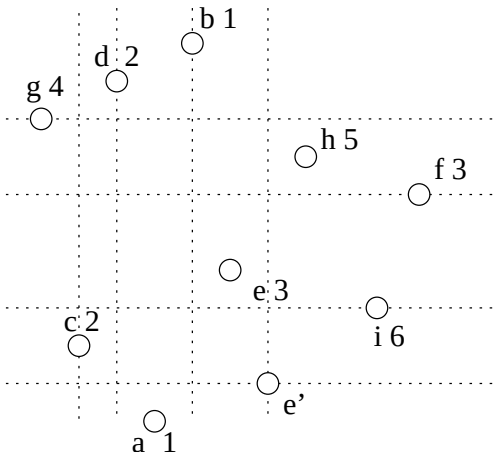

Fig. 3. Graph delivered to force any admissible algorithm to use at least $3(\chi - 1)$ colors on a χ-colorable graph G. The letters near the vertices represent their order of appearance. The numbers represent their colors. The proof of proposition 5 explains why vertex d cannot be colored with color 3. The dotted lines are a help to visualize the placement of vertices relatively to each-other.

Proof. If $\chi(G) = 3$, we devise an adversary that presents the vertices as shown on figure 3. The letters near the vertices represent their order of delivery and the numbers represent the colors that any admissible algorithm, in the sense of proposition 5, would have to attribute to these vertices. Vertex e' is not actually delivered and is only used for the proof. We explain for each vertex why it can get only one color below.

- vertex a is the first vertex. It must be colored with 1.
- When vertex b is delivered, G is still 1-colorable. So, in order to respect the 1-bound, we must color it with color 1.
- vertex c is adjacent to vertex a. We must use a new color, call it 2.
- vertex d could not be colored with color 3, because if it was, an adversary could deliver the vertex e', which would then have to be colored with 4, thus not respecting the 3-bound on bipartite graphs.
- vertex f must be colored with color 3.
- vertices g, h and i can be colored only with one color each.

3 Cocoloring

In the offline case, cocoloring is known to be more difficult than coloring in comparability graphs (NP-hard) [22]. A natural question is whether it is also more difficult in the online case. Indeed, in the next section, we will point out that online cocoloring is as difficult in permutation graphs than in general graphs. For this reason, we will restrict us to permutation graphs and we will consider some relaxations of the online model allowing interesting results. We restrict ourselves to 2-cochromatic graphs. We first consider restrictions on the way the adversary may present the graph: we adopt the discrete latticial model and the permutation graph is presented from west to east. The second relaxation gives more freedom to the algorithm: we allow it a bounded delay before deciding the color of the presented vertices.

Of course, competitive analysis devised in the most general model remain valid for the relaxations.

3.1 A dramatic bound in the continuous latticial model

Di Stefano et al. [21] have shown that two natural greedy algorithms can guarantee a performance ratio of $\frac{n}{4} + \frac{1}{2}$ for the problem of online cocoloring a permutation graph. One can very simply note that this result also holds for general graphs. In what follows, we point out that no algorithm can guarantee a better ratio, even for split permutation graphs. We recall that a graph is called split if its vertices can be covered with one clique and one stable set.

Proposition 6.

i. FF_z guarantees a performance ratio of $\frac{n}{4} + \frac{1}{2}$ for online cocoloring general graphs.

ii. No algorithm for online cocoloring graphs can guarantee a performance ratio better than $\frac{n}{4} + \frac{1}{2}$ even for split permutation graphs presented from west to east (in the continuous latticial model).

Proof.

i. The proof is very similar to the proof given in [21]. It is straightforward to verify that FF_z cocolors exactly graphs with cochromatic number 1. So, we can assume that the cochromatic number is at least 2. Then, it is very simple to see that FF_z leaves at most one vertex alone in its color-class, since any two vertices form either a clique or a stable set. Thus, the ratio $\frac{n}{4} + \frac{1}{2}$ immediately follows ($\frac{n}{4}$ if n is even).

ii. We devise an adversary which presents a split permutation graph and forces $\lceil \frac{n}{2} \rceil$ colors.

The adversary delivers the latticial representation of the permutation from west to east. While the algorithm makes color-classes of size one, the elements can be presented with any value. As soon as the algorithm makes a color-class of size two (increasing or decreasing), all the upcoming elements must be presented strictly within the interval of the two elements of this color-class, thus making it impossible to put any upcoming element in this color-class. Thus, no color-class has a cardinality larger than 2, and the number of color-classes used is bigger or equal to $\lceil \frac{n}{2} \rceil$. ■

Remark 6. Note that proposition 6 holds even if the graph representing the permutation is a threshold graph (which means that the highest element of the stable set is lower than the lowest element of the clique).

3.2 Split permutation graphs in a discrete latticial model

Let us now consider the discrete latticial model, where the vertices have coordinates in $\{1, 2, \ldots, n\}$, n being the order of the graph. Moreover, the vertices are presented along the direction from west to east. In other words, the corresponding permutation, seen as a sequence of $\{1, \ldots, n\}$, is revealed from left to right.

Di Stefano et al. [21] prove that one can force $\frac{\log_2 n}{2}$ colors on a 3-cocolorable permutation graph, thus achieving a lower bound of $\frac{\log_2 n}{6}$. Let us note that a slight modification of their proof allows to force $\frac{\log_2 n}{2}$ colors on a 2-cocolorable permutation graph, thus achieving a better lower bound of $\frac{\log_2 n}{4}$. To make this paper self-contained, we give a sketch of proof that is the "discrete counterpart" of the proof of proposition 6.

We consider an adversary called SPLAD that presents split permutation graph of order $n = 2^p$ from west to east.

It is straightforward to see that at most $\log_2 n = p$ vertices are presented before step 12 is executed. Moreover, these $\log_2 n$ first vertices are colored with at least $\frac{\log_2 n}{2}$ color-classes. To complete the proof, we just have to note that it is always possible for the adversary to correctly execute step 12 by filling the

Algorithm 4 SPLAD

1: Let R be the range of integer values that the vertices can take.
2: Let $l = \min(R)$ and $h = \max(R)$.
3: **repeat**
4: Let $m = \text{round } \left(\frac{l+h}{2}\right)$.
5: Introduce a vertex with value m.
6: **if** vertex m is colored with a clique-color **then**
7: $h := m - 1$
8: **else** */*vertex m is colored with a stable-color or with a new color*/*
9: $l := m + 1$
10: **end if**
11: **until** ($l = m$ or $h = m$)
12: Fill the positions that have no vertex in such a way that the complete graph is a split graph.

remaining positions with a stable set above m and a clique below m. Since the complete graph is 2-cochromatic, we have the competitive ratio of $\frac{\log_2 n}{4}$.

Next, we present an algorithm, called Closest-Fit, which guarantees a competitive ratio of $2\log_2(n)$ if the permutation is presented from west to east.

Theorem 3. *Closest-Fit is an online algorithm for cocoloring split permutation graphs in the discrete latticial model, where the graph is presented from west to east. It guarantees at most* $3+2\log_2 n$ *color-classes, which leads to a competitivity ratio of* $\log_2(n) + \frac{3}{2}$.

Proof. We denote by v^x and v^y the coordinates of a vertex in the latticial representation of a graph.

Lemma 5. *If $S_m > C_m$, then no vertex will be presented between S_m and C_m ($C_m \leq v^y \leq S_m$). Moreover, all vertices that will be presented in the future above S_m (respectively below C_m) constitute a stable set (respectively a clique) and will be colored as such by Closest-Fit.*

Proof. Let s_m (respectively c_m) be the vertex at height S_m (respectively C_m). Suppose $S_m > C_m$. Thus, $\exists v_s \in NE(s_m)$ and $\exists v_c \in SE(c_m)$. Since G is split, one of the vertices v_s and s_m is stable and one of the vertices v_c and c_m is clique. Thus, one vertex between S_m and C_m would lead to a contradiction.

$\forall v \in NE(s_m)$ not yet presented, $v \in NE(c_m) \cap NE(v_c)$. Thus, v must be stable in all possible split decomposition of G. A similar reasoning shows that $\forall v \in SE(c_m)$ must be clique in all possible split decompositions of G. □

Since S_m increases and C_m decreases along the execution of the algorithm, as soon as step 13 is executed once, the rest of the graph is cocolored with at most 2 new colors. Let us now calculate the number of colors used before step 13 is executed.

Remark 7. As long as $S_m < C_m$, $SE(s_m) \cap NE(c_m)$ contains exactly 1 vertex.

Algorithm 5 Closest-Fit

Input: A permutation over n elements delivered online in the order of the permutation.
Output: A partition of this permutation into at most $\log_2 n$ cliques and stable sets.

1: Introduce a dummy vertex at coordinate $(-1,-1)$ and color it with color s_1.
2: Introduce a dummy vertex at coordinate $(-1,h)$, where h is the height of the interval, and color it with color c_2
3: **for** each new vertex v with value m **do**
4: $S_m := \max_v \{v^y | \exists v' : v' \in NE(v)\}$
5: $C_m := \min_v \{v^y | \exists v' : v' \in SE(v)\}$
6: **if** $S_m < C_m$ **then**
7: **if** v is closer to C_m than to S_m **then**
8: Color v with the first available color c_i.
9: **else**
10: Color v with the first available color s_i.
11: **end if**
12: **else** /*$S_m > C_m$, so the point where the stable set crosses the clique has passed*/
13: **if** $v^y > S_m$ **then**
14: Color v with the first available color s_i.
15: **else** /*$v^y < C_m$*/
16: Color v with the first available color c_i.
17: **end if**
18: **end if**
19: **end for**

Lemma 6. *If the color s_i is used, we note Δ_i^s the difference $(C_{m_i} - S_{m_i})$ at the time when s_i was used for the first time. We define Δ_j^c similarly for the color c_j. Then, if s_i, $i \geq 2$ is used, $\Delta_i^s \leq \frac{1}{2}\Delta_{i-1}^s$ and if c_j, $j \geq 2$ is used, $\Delta_j^c \leq \frac{1}{2}\Delta_{j-1}^c$.*

Proof. We will prove lemma 6 for Δ_i^s. The proof for Δ_j^c is similar.

At the time when a first vertex v_i gets the color s_i, there exist v_{i-1} colored with s_{i-1}, with $v_{i-1} \in NW(v_i)$. Let $S'_{m_{i-1}}$ and $C'_{m_{i-1}}$ be the values of S_m and C_m when v_{i-1} was presented. $\Delta_{i-1}^s \geq |C'_{m_{i-1}} - S'_{m_{i-1}}|$. Since v_{i-1} was colored s_{i-1}, $|v_{i-1}^y - S'_{m_{i-1}}| \leq |C'_{m_{i-1}} - v_{i-1}^y|$. Now, $v_i^y < v_{i-1}^y$ and $v_i^x > v_{i-1}^x$.

Let c_i and s_i be the vertices such that $S_{m_i} = s_i^y$ and $C_{m_i} = c_i^y$. According to remark 7, $c_{m_i} \leq v_{i-1}^y$. Thus, $\Delta_i^s \leq |v_{i-1}^y - S'_{m_{i-1}}| \leq \frac{1}{2}|S'_{m_{i-1}} - C'_{m_{i-1}}| \leq \frac{1}{2}\Delta_{i-1}^s$. □

Suppose now that the algorithm uses t colors before the execution of step 13, and that these colors appeared in the order $k_1, k_2, \ldots, k_t$. $\forall i$, $k_i \in S$ (Stable colors) or $k_i \in C$ (Clique colors). We note Δ_i the difference $C_{m_1} - S_{m_i}$ at the time when k_i was used for the first time. $\Delta_1 \geq \Delta_2 \geq \ldots \Delta_t$.

Since, out of 3 colors, two at least are of the same type, lemma 6 tells us that $\Delta_{2k+1} \leq \frac{1}{2}\Delta_{2k-1}$. Now, $\Delta_1 \leq n$. Thus, $\Delta_{2k+1} \leq \frac{1}{2^k}n$ and $\Delta_{2k+2} \leq \frac{1}{2^k}n$. So $\Delta_t \leq \frac{1}{2^{(t/2)-1}}n$. Since $\Delta_t \geq 1$, $2^{(t/2)-1} \leq n$ and $\frac{t}{2} - 1 \leq \log_2 n$. Thus, $t \leq 2\log_2 n + 1$.

Altogether, Closest-Fit has used at most $3 + 2\log_2 n$ colors. ■

Corollary 5. *It is easy to transform Closest-Fit into an online algorithm for cocoloring 2-cochromatic graphs with at most* $7 + 2\log_2 n$ *color-classes, leading to a competitivity ratio of* $\frac{7}{2} + \log_2 n$.

Proof. The modified algorithm begins as FF until it needs a third color, then it switches to FF_k until it needs 5 colors and finally it turns to Closest-Fit by using the 5^{th} color as a first color of Closest-Fit. Since the graph is revealed along the direction $(1, 0)$, FF and FF_k are exact. If the graph is bipartite it will be colored exactly; if it is cobipartite, it will be colored with 4 color-classes and if it is a split graph, it will be colored with at most $7 + 2\log_2 n$ color-classes. ■

3.3 Delayed cocoloring

Next, we look at two relaxations of the cocoloring problem where the algorithm is allowed a bounded delay before deciding the color-class of the presented vertices. We consider that one vertex is delivered at each time unit.

We start with a case where the graph presented is a split permutation graph, presented from west to east. The algorithm is allowed to wait for at most one time unit before deciding the color-class of each vertex presented.

Theorem 4. *A split permutation graph G delivered online from west to east on a latticial model can be cocolored exactly if the algorithm is allowed a delay of one time unit before deciding the color-class of each vertex presented.*

Algorithm 6 1-Late-Coloring

Proof. **Input:** A permutation graph G presented online from west to east on a continuous latticial model.
Output: An exact cocoloring of G.

```
 1: for each new vertex v do
 2:    if v can be colored only with s, respectively c then
 3:       Color v with s, respectively with c /*The next test is for the case of a split-
          ordered graph and we have reached the end of the permutation*/
 4:    else if v is the last vertex of the permutation then
 5:       Color v with s or with c
 6:    else
 7:       Wait for the next vertex v' to be delivered
 8:       if v' ∈ NE(v) then
 9:          Color v with color s
10:       else /*v' ∈ SE(v)*/
11:          Color v with color c
12:       end if
13:    end if
14: end for
```

The algorithm 1-Late-Coloring cocolors a split permutation graph G delivered online from west to east on a latticial model exactly. We prove this by induction on v.

Suppose the already attributed cocolors correspond to an admissible split-representation of G. v has been presented, but not yet cocolored, and v' is being presented.

If v is above the lowest vertex cocolored with c or below the highest vertex cocolored with s, then we are in the case of line 2, and the cocoloring of v is still admissible.

Suppose v is above the highest vertex cocolored with s and below the lowest vertex cocolored with c. Suppose v' is above v (the proof is similar in the opposite case). If there exist a representation for which cocolor(v)=c, necessarily, cocolor(v') will be s (because v' is higher than a vertex cocolored with c). But in this case, one could change the cocolor of v into s and still have an admissible cocoloring. Thus, coloring v with s is never wrong.

Initialization of the induction: the same reasoning holds for the first two vertices. We can consider that there is a vertex with color s at $-\infty$ and a vertex with color c at $+\infty$. ■

Given this pleasant result, we look now at a less relaxed model, where the presented graph G may have a cochromatic number $z(G)$ higher than two.

Theorem 5. *Even if an algorithm is allowed to wait for $n-4$ time units before coloring a vertex, it is not possible to cocolor exactly a 3-cocolorable graph online.*

Proof. The proof uses figure 4. The vertices without label represent a stable set of size $n-5$. The vertices are presented from west to east. a is colored 1. Once the vertices of the stable set are presented, b must be colored. If a and b are colored with the same color, the adversary will present vertices c, d and e. Else, the adversary will present f, g and h. In any case, the algorithm will have to use at least four color-classes to cocolor the graph, while three would have been sufficient. ■

4 Conclusion

This work gives a tight analysis of First-Fit for the problem of online coloring comparability and permutation graphs and shows that its performance ratio is $O(\sqrt{n})$. It also presents an algorithm which dramatically improves the performance ratio of this problem to $\frac{\chi+1}{2}$, and gives a tight analysis of it. Furthermore, some variations of coloring problems are analyzed on the same type of graphs: The exact performance ratio $(\frac{n}{4}+\frac{1}{2})$ of the problem of online cocoloring comparability graphs is given and online cocoloring split-graphs is analyzed.

Some questions remain open, such as the exact performance ratio of the problem of online coloring comparability graphs, as well as the performance ratio of online coloring these graphs while respecting bounds on the size of the colors.

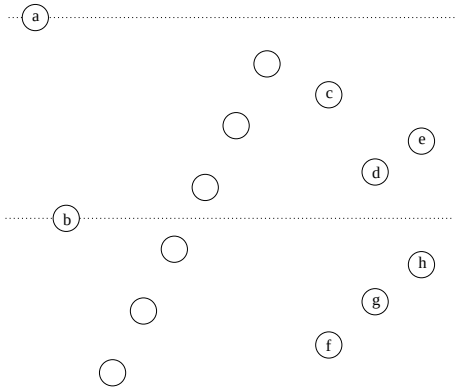

Fig. 4. This figure proves Theorem 5. The vertices without label represent a stable set of size $n - 5$. The vertices are presented from west to east. a is colored 1. Once the vertices of the stable set are presented, b must be colored. If a and b are colored with the same color-class, the adversary will present vertices c, d and e. Else, the adversary will present f, g and h.

5 Acknowledgment

The authors would like to thank Jack Edmonds and Dominique de Werra for fruitful comments on this work, as well as Bernard Monjardet for constructive criticisms on its presentation.

This work was partially financed by the ESSEC Business School and the ADONET grant MRTN-CT-2003-504438.

References

1. Eric Bach, Joan Boyar, Leah Epstein, Lene M. Favrholdt, Tao Jiang, Kim S. Larsen, Guo-Hui Lin, and Rob van Stee. Tight bounds on the competitive ratio on accommodating sequences for the seat reservation problem. *J. Scheduling*, 6(2):131–147, 2003.
2. Claude Berge. *Graphs and Hypergraphs*. North Holland, Amsterdam, 1976.
3. Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 2nd edition, 2005.
4. Vincent Bouchitté and Jean-Xavier Rampon. On-line algorithms for orders. *Theor. Comput. Sci.*, 175(2):225–238, 1997.
5. Vašek Chvátal. Perfectly ordered graphs. In *Topics on perfect graphs*, volume 88 of *North-Holland Math. Stud.*, pages 63–65. North-Holland, Amsterdam, 1984.
6. Marc Demange, Tinaz Ekim, and Dominique de Werra. Variations of split-coloring in permutation graphs. Technical report, École Polytechnique Fédérale de Lausanne, 2006.
7. Stefan Felsner. On-line chain partitions of orders. *Theor. Comput. Sci.*, 175(2):283–292, 1997.
8. Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Annals of discrete mathematics. Elsevier, 2nd edition, 2004.
9. András Gyárfás, Zoltán Király, and Jenö Lehel. On-line 3-chromatic graphs I. triangle-free graphs. *SIAM J. Discrete Math.*, 12(3):385–411, 1999.
10. András Gyárfás and Jenö Lehel. On-line and first-fit colorings of graphs. *Journal of Graph Theory*, 12(2):217–227, 1988.
11. Magnús M. Halldórsson and Mario Szegedy. Lower bounds for on-line graph coloring. *Theor. Comput. Sci.*, 130(1):163–174, 1994.
12. Henry A. Kierstead. An effective version of dilworth's theorem. *Trans. Amer. Math. Soc.*, 268(1):63–77, 1981.
13. Henry A. Kierstead, Stephen G. Penrice, and William T. Trotter. On-line coloring and recursive graph theory. *SIAM J. Discret. Math.*, 7(1):72–89, 1994.
14. Henry A. Kierstead, Stephen G. Penrice, and William T. Trotter. On-line and first-fit coloring of graphs that do not induce p_5. *SIAM J. Discret. Math.*, 8(4):485–498, 1995.
15. Linda Lesniak and H. Joseph Straight. The cochromatic number of a graph. *Ars Combinatoria*, 3:39–46, 1977.
16. László Lovász, Michael E. Saks, and William T. Trotter. An online graph coloring algorithm with sublinear performance ratio. *Discrete Math*, 75:319–325, 1989.
17. Avery Miller. Online graph colouring. Canadian Undergraduate Mathematics Conference `http://www.cumc.math.ca/2005/papers/miller.pdf`, 2004.
18. Stavros D. Nikolopoulos and Charis Papadopoulos. On the performance of the first-fit coloring algorithm on permutation graphs. *Inf. Process. Lett.*, 75(6):265–273, 2000.
19. Frits Spieksma and Linda Moonen. Partitioning a permutation graph: algorithms and an application. In *Operations Research 2004, International Conference, Tilburg University (The Netherlands)*, pages 150–150, September 2004.
20. Frits Spieksma and Linda Moonen. Partitioning a weighted partial order. Technical report, K.U.Leuven, 2005. DTEW Research Report 0538, 15 pp.
21. Gabriele Di Stefano, Stefan Krause, Marco E. Lübbecke, and Uwe T. Zimmermann. On minimum -modal partitions of permutations. In *LATIN*, pages 374–385, 2006.
22. Klaus Wagner. Monotonic coverings of finite sets. *Elektronische Informationsverarbeitung und Kybernetik*, 20(12):633–639, 1984.