Intervalizing colored graphs is NP-complete for caterpillars with hair length 2 *

C. Alvarez† J. Diaz† M. Serna†

March 20, 1998

Abstract

The problem of Intervalizing Colored Graphs has received a lot of attention due to their use as a model for DNA physical mapping with ambiguous data. If k is the number of colors, the problem is known to be NP-Complete for general graphs for $k \geq 4$ and has polynomial time algorithms for $k = 2$ and $k = 3$. In this paper we prove that the problem is NP-Complete for caterpillars with hairs of length at most 2. In the positive side we give polynomial time algorithms for the problem in the cases, caterpillars with hairs of length at most 1 and any number of colors and caterpillars with hairs of length at most 2 and a constant number of colors. It is the first time a problem has been shown NP-complete for the particular case of caterpillars of hairs of length at most 2.

1 Introduction

Interval graphs have become very fashionable because of their use as a model for cloning of DNA (see Section 5.2 in [SM97]). Recall that in an *interval graph* one can assign to each vertex in the graph an interval of the real line, in such a way that two vertices are adjacent if and only if their intervals have nonempty intersection [Gol80].

Given a graph with colors assigned to its vertices the problem of *Intervalizing Colored Graphs* (ICG) consists in deciding if there is a properly colored

*This research was partially supported by the ESPRIT Long Term Research Project no. 20244 – ALCOM-IT and CICYT project TIC97-1475-CE.

†Departament de Llenguatges i Sistemes Informatics, Universitat Politècnica Catalunya, Edifici C6 Campus Nord, Jordi Girona Salgado 1-3, 08034-Barcelona {alvarez,diaz,mjserna}@lsi.upc.es

supergraph which is an interval graph. The ICG problem is a special case of the *Interval Sandwich Problem* [GS93]. The ICG problem has received a lot of attention recently, as a simplified model for reconstructing the ordering physical DNA mapping problems [FHW93]. In this last work they prove that the general ICG problem is NP-Complete (see also [GKS94]). If the problem is given parameterized on the number of colors, hardness results were obtained in [BFH95]. Later it was shown that the problem is NP-Complete for the case of 4 or more colors, while if we only have 2 and 3 colors the problem is solvable in linear and quadratic time respectively [BdF95]. A different approach to study the problem is to fix the number of colors and look at the complexity of the problem considering the degree of vertices. For fixed number of colors k and degree bounded by a constant d there is a $O(n^{k-1})$ algorithm for the ICG [KS96].

In this paper we address the complexity of the ICG problem for caterpillar. Recall that a *caterpillar* is a particular kind of tree consisting of a simple chain (called the *backbone*) with an arbitrary number of chains (*hairs*) attached by coalescing an endpoint of the hair with a vertex in the backbone. A caterpillar has hairs of *length* at most l if all the simple chains attached to the body have length at most l. Up to now, the bandwidth minimization problem was the example of problem known to be NP-complete for caterpillars with hairs of length at most 3 [Mon86]. The problem can be solved in $O(n \log n)$ time for caterpillars of length at most 2 [APSZ81].

We will show that Intervalizing colored graphs is NP-complete for caterpillars with hairs of length at most 2 and that the problem can be solved in NC for caterpillars with hairs of length at most 1 (independently of the number of colors). We also give a polynomial time algorithm for caterpillars with hairs of length at most 2 and constant number of colors. Notice that our caterpillars do not fall in the class of graphs considered by Bodlaender and de Fluiter [BdF95]. In this last work, the authors obtain hardness results for the case of 4 colors, they construct graphs with pathwidth 2, but not trees. The caterpillars with hairs of length at most 2 are trees with pathwidth 2.

To obtain the result, we exploit the relationship between the ICG problem and a layout problem, the *Colored Vertex Separation*. Both problems were shown identical in [Din96]. The non-colored version of the vertex separation problem is an NP-Complete problem with several applications in Computer Science (VLSI design, compiler construction, etc.) and it has been well studied [Len81, EST79, KP86, Kin92].

In section 2 we introduce the basic definitions and the formal statement for the problems. We show that the ICG problem for caterpillars with

hairs of length at most 1 is in the class NC. In section 3 we show that the ICG problem is NP-Complete for caterpillars with hairs of length at most 2. Finally in section 4 we give a polynomial time algorithm in the case of caterpillars with hairs of length at most 2 and a constant number of colors. Notice that it is open to find a NC algorithm for this last case.

2 Definitions and basic results

A *k-coloring* of a graph $G = (V, E)$ is a mapping $\kappa : V \longrightarrow \{1, \ldots, k\}$. For any vertex subset $V' \subseteq V$, let $\kappa(V') = \{\kappa(v) \mid v \in V'\}$. A *proper coloring* of G is a coloring such that no two adjacent vertices have the same color. A *k-colored* graph is a graph together with a k-coloring.

Given a graph $G = (V, E)$ with $|V| = n$ a *layout* φ of a G is a one to one mapping $\varphi : V \longrightarrow \{1, \ldots, n\}$. For any layout φ of G, we can define the sets $V_i = \{v \mid \varphi(v) \leqslant i \text{ and } \exists u\ \varphi(u) > i\, (u, v) \in E\}$ one for each $1 \leqslant i \leqslant n$. Recall that the *vertex separation* of a graph G with respect to a given layout φ is defined as $\mathrm{vs}(G, \varphi) = \max_{1 \leqslant i \leqslant n}\{|V_i|\}$. The *vertex separation* of a graph G, denoted as $\mathrm{vs}(G)$ is the minimum $\mathrm{vs}(G, \varphi)$ over all layouts of G.

For a given $k \in \mathbb{N}$, a *colored layout* of a k-colored graph $G = (V, E)$ is a layout φ such that for all $u \in V$ with $\varphi(u) > 1$, $\kappa(u) \notin \kappa(V_{\varphi(u)-1})$.

Let us consider the following problem: Given a k-colored graph $G = (V, E, \kappa)$, the **colored vertex separation problem** (**CVS**) is to decide if there is a colored layout φ of G, such that $\mathrm{vs}(G, \varphi) < k$.

It is known that for a given parameter $k \in \mathbb{N}$, the CVS problem is identical to the ICG problem [Din96]. Therefore all the P and NP-Completeness results described above for the ICG also apply to the CVS problem.

From the definition of colored layout, given a k-colored graph $G = (V, E, \kappa)$, if there is a colored layout of G $\varphi : V \longrightarrow \{1, \ldots, n\}$, then for all $u, v, x \in V$, such that $\varphi(u) < \varphi(v) < \varphi(x)$ and $(u, x) \in E$, then $\kappa(u) \neq \kappa(v)$ and $\kappa(x) \neq \kappa(u)$, In this case we say that the color of u is *blocked* until x appears and κ is a proper coloring. From now on, the term *colored-graph* will refer to a colored graph in which the coloring is proper.

Given a layout φ of a graph $G = (V, E)$ such that $\mathrm{vs}(G, \varphi) < k$, we can use a greedy procedure to assign colors to V. Following the order given by φ as $|V_i| < k$, at any point the number of blocked colors is at most $k - 1$ and we can always assign a free color the next vertex. Therefore,

Lemma 1. *Given a graph G, if φ is a layout of G and $\mathrm{vs}(G, \varphi) < k$ there is a k-coloring of G for which φ is a colored layout.*

To complete the relationship between vertex separation and coloring, it is straightforward to show that,

Lemma 2. *Given a k-colored graph $G = (V, E, \kappa)$, if there is a colored layout of G then* $\mathrm{vs}(G) < k$.

These lemmas show that the *colored layout* problem of finding a k-coloring for a graph for which there is a colored layout, is NP-complete. Furthermore it can be solved in polynomial time for trees, using the algorithm described in [EST79] to find a layout with optimal vertex separation. Notice that these results allow us to simplify the definition of colored vertex separation in the sense that we just need to decide about the existence of a colored layout.

The following technical result will be used in section 4 and follows from the definitions,

Lemma 3. *Given a k-colored graph $G = (V, E, \kappa)$. If there is a colored layout of G then there is a colored layout for any of its subgraphs.*

A caterpillar with hairs of length at most one is formed by a chain, each node in the chain can be connected only to degree one nodes. Therefore given a k-coloring of the caterpillar, we can find a colored layout provided the coloring is proper (see figure 1, colors are written inside the circles representing the nodes). Just lay in order the chain, and put all the remaining neighbors of a node right after it. Using pointer jumping and prefix sums we get,

Lemma 4. *The Colored Vertex Separation problem for caterpillars with hairs of length at most one belongs to the class NC.*

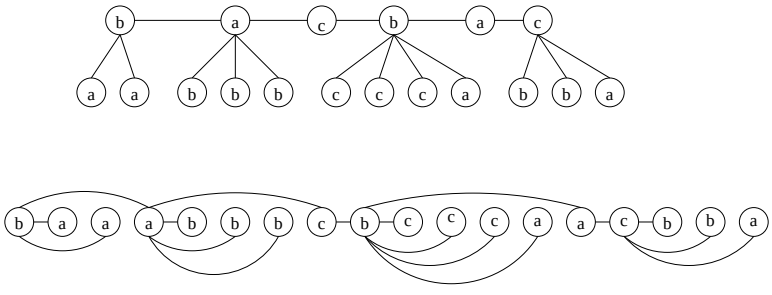

Figure 1: A colored layout for a caterpillar with hairs of length 1.

The above solution can be used to produce a solution of the ICG problem with the same complexity bounds.

3 The reduction

We will show that the COLORED VERTEX SEPARATION problem is NP-complete for caterpillars with hairs of length two. The proof follows the schema given in [Mon86]. We give a reduction from the *Multiprocessor Scheduling problem*. That is, given a set of n tasks, having duration t_i for $1 \leqslant i \leqslant n$, a deadline D and a set of m processors determine whether the tasks can be assigned to processors so that all processors finish their work before the deadline D. Recall that the problem is NP-complete in the strong sense, so we can assume that the duration of each task is polynomially bounded.

Our reduction will produce a colored caterpillar such that it has a colored layout if and only if the original Multiprocessor Scheduling instance has solution.

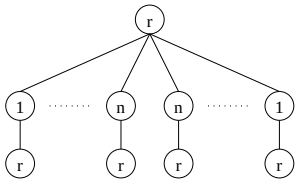

Figure 2: A barrier $B(\Delta, r)$, with $\Delta = \{1, \ldots, n\}$.

We first consider an auxiliary gadget, the colored caterpillar given in figure 2. The graph is formed by a star, whose center will be in the backbone of the caterpillar. We call such graph a *barrier* with colors Δ and *center* colored r with $r \notin \Delta$. We will denote such graph by $B(\Delta, r)$. The main properties of a barrier are the following

Lemma 5. *In any colored layout φ of a barrier $B(\Delta, r)$ the children must be partitioned into two sets, one in each side of the center in such a way that in each set there is not repeated color.*

Proof. Let us suppose that there are two children a, b with $\kappa(a) = \kappa(b)$ in the same side of the center c. If they are in the left side then we have that $\varphi(a) < \varphi(b) < \varphi(c)$. In this case φ does not satisfy the condition of a colored layout since $\kappa(a) = \kappa(b)$ and there is an edge $(a, c) \in E$.

On the other hand, if $\varphi(c) < \varphi(a) < \varphi(b)$ let us consider the sons a' and b' of a and b. As a' and b' have the same color than c, then $\varphi(b) < \varphi(a'), \varphi(b')$. But as $\kappa(a) = \kappa(b)$ then neither in the case that $\varphi(b) < \varphi(a') < \varphi(b')$ nor in the case $\varphi(a) < \varphi(b') < \varphi(a')$, φ is a colored layout. □

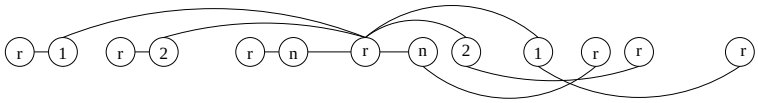

Figure 3: A colored layout for the barrier $B(\Delta, r)$.

So, since any two children of the center of the barrier with the same color must lay in different sides of the center, we get that modulo some permutation the only colored layout is given in figure 3.

Lemma 6. *Given a barrier $B(\Delta, r)$ and an edge (u, v) such that $\kappa\{u, v\} \subseteq \Delta \cup \{r\}$. In any colored layout of the barrier and the edge the two endpoints of the edge must lay both on the same side (before or after) of the barrier's center.*

Proof. A colored layout for the barrier must partition the children into two sets such that in each set there is no repeated color (see figure 3). Let us suppose that u and v are such that $\varphi(u) < \varphi(c) < \varphi(v)$. By lemma 5, we have that all the colors Δ are blocked before the center. Hence in both cases $\kappa(u) \neq r$, or $\kappa(u) = r$, φ is not a colored layout. □

The second gadget is the colored caterpillar T given in figure 4, the backbone follows the vertical line, and the dashed lines represents the connection to other gadgets to be described later. Some of the nodes are labeled, this labels will be used to refer to them. We will refer to such caterpillar as *the turning point* with center c.

Our construction of the caterpillar is based in the fact that in any colored layout of the turning point, both end nodes lay in the same side of the center.

Theorem 1. *In any colored layout of the turning point T, the two end points, a_1 and b_4 lay on the same side of c.*

In order to prove this theorem, we consider three technical lemmas to obtain colored layouts for critical subgraphs of the turning point. Let us start with the first one.

Lemma 7. *In any colored layout of the turning point T the nodes x_1 and b_1 can not lay in the same side of the center c.*

Proof. Let us suppose that x_1 and b_1 lay in the same side of c.
Left side: If x_1 and b_1 lay in the left side then φ is not a colored layout, since $\kappa(x_1) = \kappa(b_1)$ and both are connected to c.

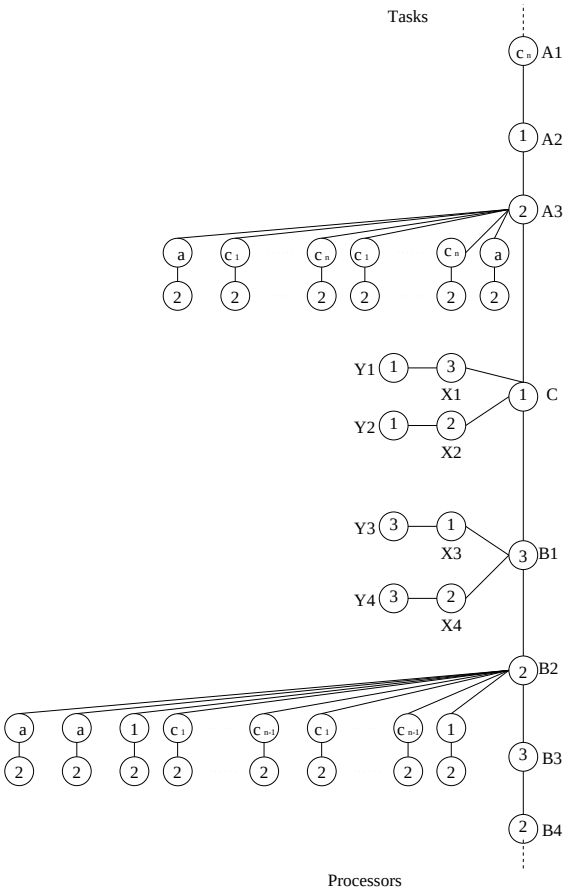

Figure 4: The turning point T.

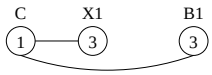

Figure 5: $\varphi(c) < \varphi(x_1) < \varphi(b_1)$

Right side: Now x_1 and b_1 lay in the right side of the center. We have two possibilities depending on the position of b_1 respecting x_1:
(i) $\varphi(c) < \varphi(x_1) < \varphi(b_1)$, as in figure 5. Since $(x_1, y_1) \in E$ and $\kappa(y_1) = \kappa(c)$ then $\varphi(y_1) > \varphi(b_1)$. But, $\varphi(x_1) < \varphi(b_1) < \varphi(y_1)$ is not possible in a colored layout because $(x_1, y_1) \in E$ and $\kappa(x_1) = \kappa(b_1)$.

Figure 6: $\varphi(c) < \varphi(b_1) < \varphi(x_1) < \varphi(y_1)$

(ii) Let $\varphi(c) < \varphi(b_1) < \varphi(x_1)$, as in figure 6. It is easy to see that $\varphi(x_1) < \varphi(y_1)$ since $\kappa(c) = \kappa(y_1)$. Moreover, as $\kappa(x_3) = 1$ and it is connected to b_1, $\varphi(x_1) < \varphi(x_3)$ but b_1 blocks the color of x_1. □

Lemma 8. *In any colored layout of the turning point T the nodes x_2 and a_3 can not lay in the same side of c.*

Proof. Let us suppose that x_2 and a_3 lay in the same side of the center. As in the previous lemma we break the proof into cases:
Left side: Let $\varphi(x_2), \varphi(a_3) < \varphi(c)$. Since $\kappa(x_2) = \kappa(a_3)$ and both are connected to c, then φ is not a colored layout.

Figure 7: $\varphi(c) < \varphi(a_3) < \varphi(x_2)$

Right side: Let us consider the other case when x_2 and a_3 are in the right side of c. Depending on the position of x_2 respecting to a_3 we have two cases.
(i) $\varphi(c) < \varphi(a_3) < \varphi(x_2)$ (see figure 7). The conflictive node is a_2 connected with a_3. Since $\kappa(a_2) = \kappa(c)$, and c blocks its color until x_2, then $\varphi(a_3) < \varphi(x_2) < \varphi(a_2)$. But as $\kappa(a_3) = \kappa(x_2)$, then φ is not a colored layout.

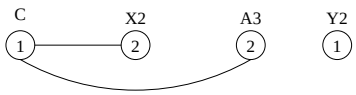

Figure 8: $\varphi(c) < \varphi(x_2) < \varphi(a_3)$.

(ii) $\varphi(c) < \varphi(x_2) < \varphi(a_3)$ (see figure 8). As $\kappa(y_2) = \kappa(c)$, y_2 can not be placed to the left of c, the only possibility is $\varphi(a_3) < \varphi(y_2)$ but x_2 blocks a_3's color. □

Lemma 9. *In any colored layout of the turning point T the nodes x_2 and b_1 can not lay in the same side of the center c.*

Proof. Let us suppose that x_2 and b_1 lay in the same side of the center.
Left side: To see x_2 and b_1 cannot lay in the left side of c, let us consider first the case that $\varphi(x_2) < \varphi(b_1) < \varphi(c)$. Since x_2 is blocking the color of b_2

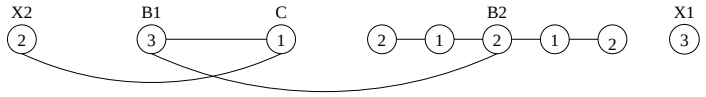

Figure 9: $\varphi(x_2) < \varphi(b_1) < \varphi(c) < \varphi(b_2)$.

which is connected to b_1, we get $\varphi(c) < \varphi(b_2)$ (see figure 9). As b_1 has the color of x_1, and it is connected to c and b_2, we get $\varphi(b_2) < \varphi(x_1)$. But b_2 is the center of a barrier which contains the color of c and by lemma 5 we know it cannot be the case that $\varphi(b_2) < \varphi(x_1)$.

The other possibility is that $\varphi(b_1) < \varphi(x_2) < \varphi(c)$. Since b_1 is blocking the color of x_1, and x_1 is connected to c, then $\varphi(c) < \varphi(x_1)$. On the other hand, there is an edge (x_1, y_1) and $\kappa(y_1) = \kappa(c)$, then $\varphi(x_1) < \varphi(y_1)$ (see figure 10). Let us analyze where to place x_4. Depending on the position of

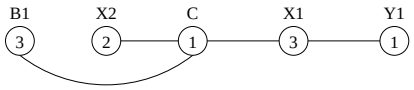

Figure 10: $\varphi(b_1) < \varphi(x_2) < \varphi(c) < \varphi(x_1) < \varphi(y_1)$.

b_1 we have two cases.
(i) $\varphi(x_4) < \varphi(b_1)$: The conflictive node is b_2 connected to b_1. Since b_2 is the center of a barrier that contains $\kappa(x_4)$, we need $\varphi(b_1) < \varphi(b_2)$. Now b_2 can take different positions (see figure 10) in such layout:

- $\varphi(b_1) < \varphi(b_2) < \varphi(x_2)$, as b_2 is connected with b_3, $\kappa(b_3) = \kappa(b_1)$ and $\kappa(x_2) = \kappa(b_2)$, we do not get a colored layout.
- $\varphi(x_2) < \varphi(b_2) < \varphi(c)$, this is impossible because $\kappa(x_2) = \kappa(b_2)$.
- $\varphi(c) < \varphi(b_2) < \varphi(x_1)$ can not happen as b_2 is the center of a barrier that contains $\kappa(c)$.
- $\varphi(x_1) < \varphi(b_2)$ again impossible as $\kappa(x_1) = \kappa(b_1)$.

(ii) $\varphi(b_1) < \varphi(x_4)$: In this situation we have two consider the following possibilities depending on the position of x_4 (see figure 10).

- $\varphi(b_1) < \varphi(x_4) < \varphi(x_2)$, since x_4 is connected with y_4, $\kappa(x_4) = \kappa(x_2)$ and $\kappa(y_4) = \kappa(b_1)$, we do not get a colored layout.
- $\varphi(x_2) < \varphi(x_4) < \varphi(c)$, this is impossible because $\kappa(x_2) = \kappa(x_4)$.
- $\varphi(c) < \varphi(x_4) < \varphi(x_1)$, we have to consider b_2, connected to b_1, with color 2. The argument used for x_4 in the previous case, can be used when $\varphi(b_1) < \varphi(b_2)$ the only position in which x_4 can be placed is in between c and x_1. This is not possible since b_2 is the center of a barrier that contains the color of c and there is an edge (c, x_1). Therefore the only remaining possibility is $\varphi(b_2) < \varphi(b_1)$, we have to place x_3 and y_3, they cannot lay to the left of b_1 because b_2 holds a barrier with $\kappa(x_3)$. As $\kappa(x_3) = \kappa(c)$ and $\kappa(y_3) = \kappa(b_1)$, x_3 can not be in between b_1 and c. But $\kappa(x_1) = \kappa(b_1)$ so, $\varphi(x_3) < \varphi(x_1)$, and this is impossible.
- $\varphi(x_1) < \varphi(x_4)$ again impossible as $\kappa(x_1) = \kappa(b_1)$.

Hence, we can conclude that b_1 and x_2 cannot lay in the left side of c. In the remaining part of the proof we show that the symmetric case gives no colored layout.
Right side: We have two possibilities depending on the position of x_2 respecting to b_1.
(i) $\varphi(c) < \varphi(b_1) < \varphi(x_2)$. Since b_1 lays at the right of c, by lemma 7 we have that x_1 has to lay at the left of c. In the same way, by lemma 8, a_3 has to lay at the left of c (see figure 11). Since $\kappa(b_2) = \kappa(a_3)$ we have that $\varphi(c) < \varphi(b_2)$. Furthermore b_2 is the center of a barrier which contains the color of c, then by lemma 6 (c, x_2) cannot cross the barrier. So we get that $\varphi(x_2) < \varphi(b_2)$. When we try to lay b_3 as b_3 is connected to b_2 and $\kappa(b_1) = \kappa(b_3)$, then $\varphi(b_2) < \varphi(b_3)$. The edge (b_1, x_3) forces $\varphi(x_3) < \varphi(b_3)$. Moreover $\kappa(x_3) = \kappa(c)$ and the edge (c, x_2) forces $\varphi(x_2) < \varphi(x_3)$. As

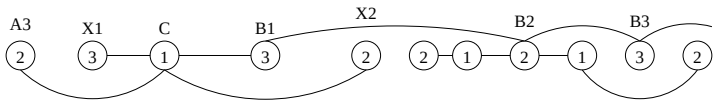

Figure 11: $\varphi(a_3) < \varphi(x_1) < \varphi(c) < \varphi(b_1) < \varphi(x_2)$.

the barrier with center b_2 contains color 1, x_3 is connected with y_3 and $\kappa(y_3) = \kappa(b_1)$, the nodes x_3 and y_3 can not be placed.

(ii) $\varphi(c) < \varphi(x_2) < \varphi(b_1)$, in this case there is an edge (x_2, y_2) and $\kappa(y_2) = \kappa(c)$ so $\varphi(b_1) < \varphi(y_2)$. Now, as $\kappa(b_2) = \kappa(x_2)$ and b_2 connected to b_1 we need $< \varphi(y_2) < \varphi(b_2)$. But b_2 is connected to b_3 and $\kappa(b_3) = \kappa(b_1)$, therefore $\varphi(b_2) < \varphi(b_3)$. Furthermore, the edge (b_1, x_4) forces $\varphi(b_1) < \varphi(x_4) < \varphi(b_2)$ because $\kappa(x_4) = \kappa(x_2) = \kappa(b_2)$. When we try to lay y_4, there is and edge (x_4, y_4) and $\kappa(y_4) = \kappa(b_1)$ so we need $\varphi(b_2) < \varphi(y_4)$ and this is again not possible. □

Proof of theorem 1. By lemmas 7, 8 and 9 we have that the only possible colored layout of the turning point T, when a_1 and b_4 lay to the right of c, is the one given in figure 12 modulo some permutation of the nodes in the barriers. Similarly it can be obtained a layout for the other case. □

The graph R_m, associated to the m processors, is the caterpillar given in figure 13. As before the backbone corresponds to the vertical line. The graph is formed by a series of barriers, with two big barriers, one at the beginning and the other at the end, joined by a backbone with two alternating colors (the second barrier forms part of the turning point). The length of the backbone between two successive barriers is $2(D + 1)$.

Lemma 10. *In any colored layout of the graph R_m the backbone must be layed out as it is, in between the centers of the first and the last barriers.*

Proof. It follows from lemma 6 and the fact that a line with alternating colors can be only laid out as it is except possible the first end point that may go after the second. In this case the last situation is impossible due to the continuation of the line after the barrier's center. □

Finally we use the colored caterpillar T_i of figure 14 to represent task i. We will join the graphs T_i and T_{i+1} with a node colored c_i, we select a new different color for each conexion. The last task is joined to the turning point through the node colored c_n that is an end point of the turning point.

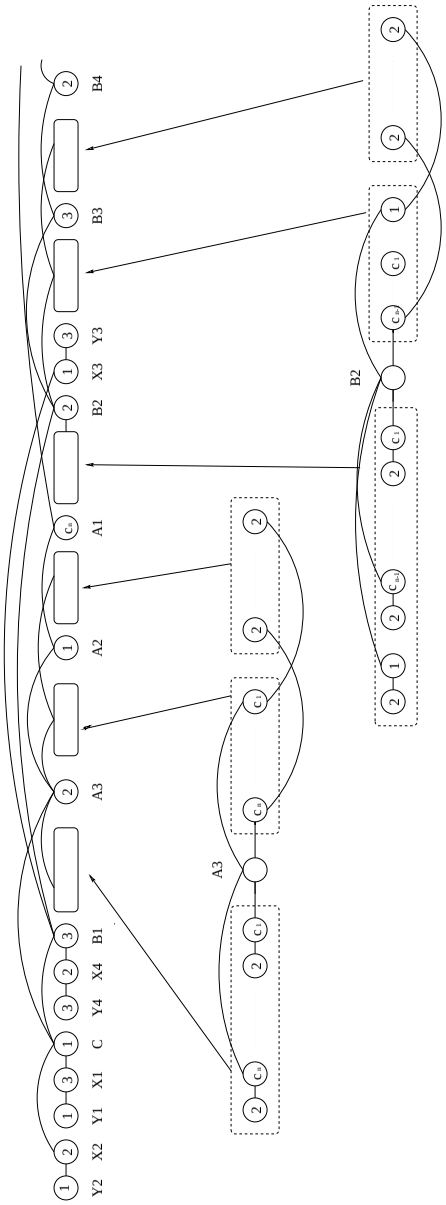

Figure 12: A layout for the turning point.

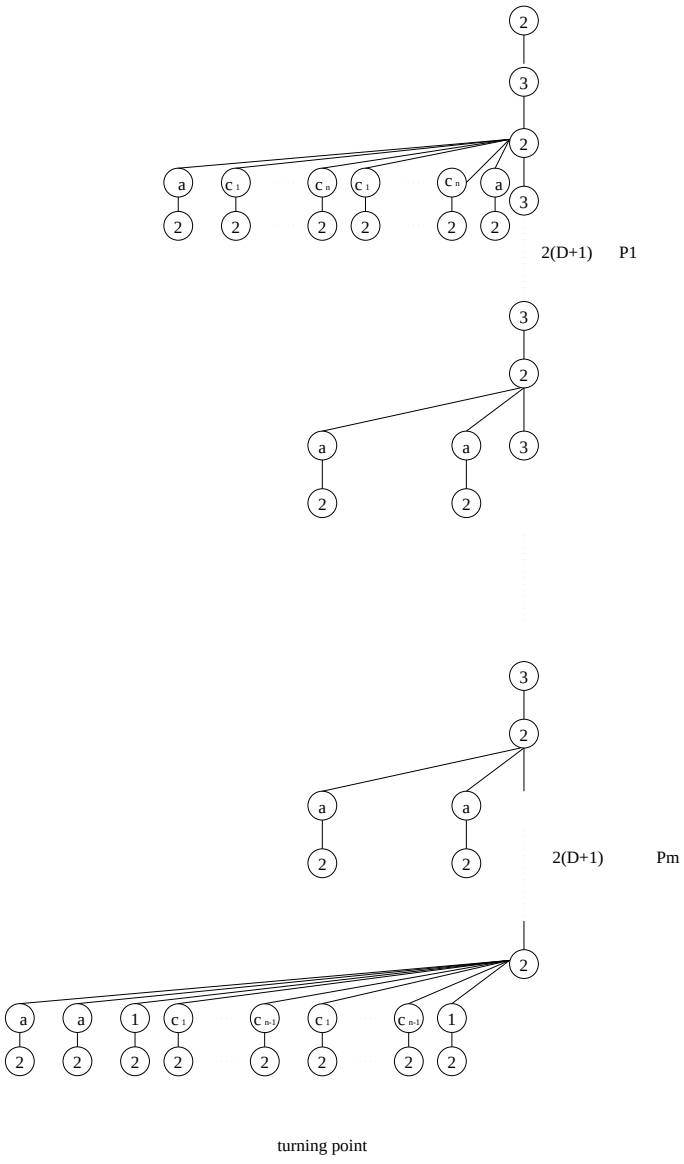

Figure 13: The graph R_m associated to the processors.

Figure 14: The graph T_i associated to task i.

For sake of simplicity let us denote by L_d a two colored line of length $2d$ such that the end points are colored with color 2 and the chain alternates colors between 3 and 2. This is just a part of the backbone of the caterpillar for the processors.

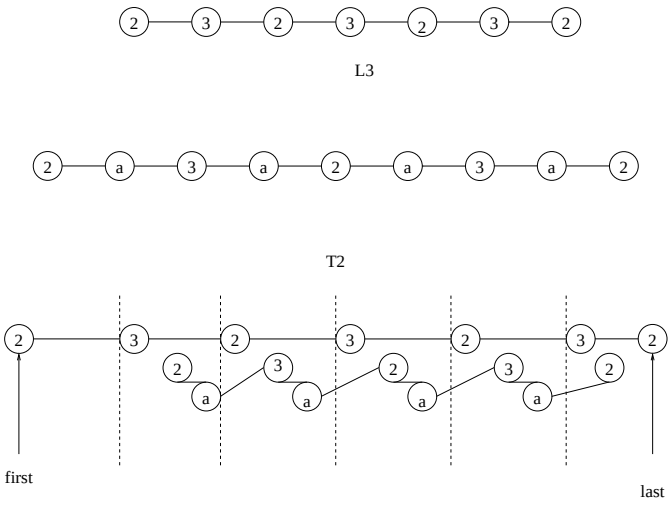

Figure 15: A joint layout of L_3 in order and T_2.

Lemma 11. *Consider the two graphs L_d and T_i. In any colored layout in which the first and the last node are the end points of L_d, T_i can be placed together with L_d if and only if $t_i < d$.*

Proof. If the two end points of L_d are first and last the chain L_d must be laid out in order. Notice that the nodes colored 2 in T_i can only be placed after a node colored 3 of L_d, furthermore, before the next node colored 2 of L_d appears, it must be placed the neighbor colored a. A similar situation happens with nodes colored 3 in T_i that can only go after a color 2 node in L_d and before next node of L_d must appear the corresponding neighbor

colored a (see figure 15). Finally notice that it is impossible to place more than one node colored 2 or 3 inside an edge of L_d and the lemma follows. □

Lemma 12. *Consider the graphs L_d, T_i, T_j . In any colored layout in which the first and the last node are the end points of L_d, T_i and T_j can be placed together with L_d if $t_i + t_j < d$.*

Proof. As the backbone of T_i and T_j holds the same alternate colors, the only possible way to lay both graphs is to put first one and after the other. The last part of T_i may overlap with the first of T_j, so the length needed is $2(t_i + t_j) + 1$. □

Given an instance $I = (t_1, \ldots, t_n, m)$ of the Multiprocessor Scheduling problem let $G(I)$ be the caterpillar obtained from the different gadgets (see figure 16). Then we have

Theorem 2. *The instance $I = (t_1, \ldots, t_n, m)$ of multiprocessor Scheduling has a solution if and only if the colored caterpillar $G(I)$ has a colored layout.*

Proof. From the previous lemmas the only possible colored layout for the caterpillar must maintain as it is the ground line corresponding to the processors, between the center of the two big barriers, the turning point must be *turned* at its center an the two endpoints on the same side (by theorem 1). The piece corresponding to the tasks must be placed between the center of the two barriers of the processors part. Now from lemma 6 the part corresponding to just one task, must go between the center of two barriers. And from lemma 12 the total duration of the tasks placed in the line corresponding to a processor must be at most D. Finally notice that the additional nodes with different colors allow us to place different tasks in any relative position provided we block the color c_i with the task T_i or T_{i+1} that is placed first. So each scheduling meeting the deadline requirement gives a colored layout and every colored layout gives a correct scheduling. □

Notice that in our reduction we construct a caterpillar with $p(n)$ vertices, where p is a polynomial, the number of colors used is $n + 4$, and n is the number of tasks in the original instance.

4 Caterpillars with hairs of length at most 2 and constant number of colors

In this section, we show that there is a polynomial time algorithm for the CVS problem in the case of caterpillars with hairs of length at most 2.

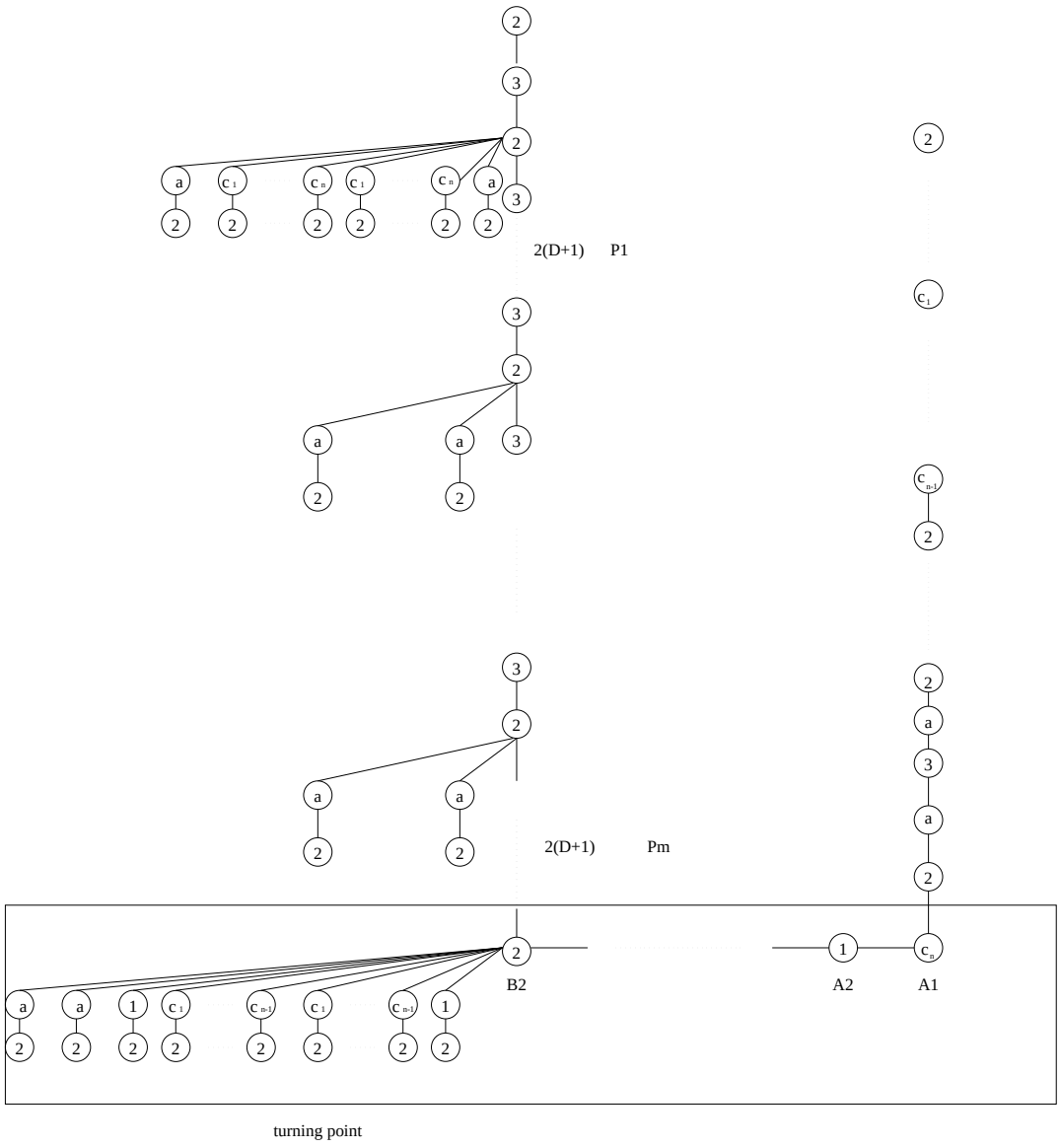

Figure 16: A sketch of the caterpillar $G(I)$.

Theorem 3. *There is a polynomial time algorithm to solve the Colored Vertex Separation problem for caterpillars with hairs of length at most two when k is a constant.*

In order to prove the above theorem we begin by analyzing some particular graphs.

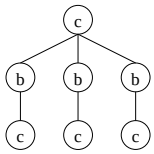

Figure 17: A simple caterpillar with no colored layout.

Lemma 13. *The graph given in figure 17 has no colored layout.*

Proof. The graph is a barrier with an additional hair, the only way to lay the barrier is as usual. From that is impossible to place the third hair. □

Notice that from lemmas 3 and 13 in case such subgraph exists there is no colored layout for the caterpillar. The next result concerns the number of repeated hairs with length 2 and will allow us to reduce the number of such hairs.

Lemma 14. *Given a caterpillar, with a backbone node v that has more than three identical hairs of length two that do not contain the color of v. The caterpillar has a colored layout if and only if the caterpillar in which we keep only three such identical hairs has a colored layout.*

Proof. The only if part follows by lemma 3. Next we show how to place the additional hairs when there is a colored layout for the caterpillar. Let us call x_1, x_2 and x_3 the three children of v and y_1, y_2 and y_3 the other node in their hair. Assume that v has color a, all the nodes x have the same color b, and nodes y have the same color c, for a, b, c differents. Notice that two children of v, cannot be placed both before v, they have the same color and one will block the other's color. Without loss of generality assume that x_1 and x_2 are placed after v, and that $\varphi(x_1) < \varphi(x_2)$. The edge (x_1, y_1) forces $\varphi(y_1) < \varphi(x_2)$, therefore $\varphi(y_1) < \varphi(y_2)$ and $\varphi(x_1) < \varphi(y_2)$.

Let us consider two possibilities; $\varphi(y_1) < \varphi(x_1)$ and $\varphi(x_1) < \varphi(y_1)$ (see figure 18). In the first case the color c must be free at the moment we place

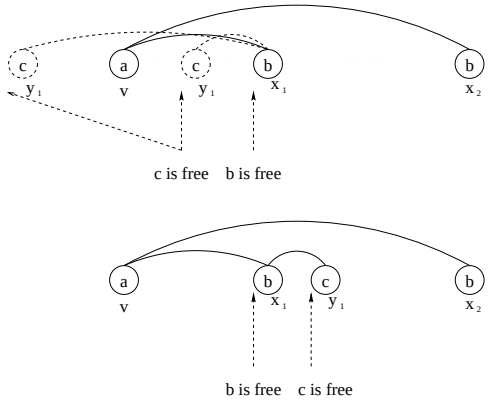

Figure 18: $\varphi(v) < \varphi(x_1) < \varphi(x_2)$.

y_1, and it remains blocked until x_1 is placed. After placing x_1 both b and c must be free. The second case is similar reversing the colors b and c. So, in the two cases, just after the placement of the last node in the hair x_1, y_1 the colors b and c are free while color a is still blocked. We can place another hair with the same sequence of colors just in this position. The existence of x_2 insures that color a, a color that the new hair may block, is already blocked. □

For hairs of length one, we only need to keep two identical hairs.

Lemma 15. *A caterpillar with a backbone node v, that has more than two identical hairs of length one has a colored layout if and only if the caterpillar in which we keep only two such identical hairs has a colored layout.*

Proof. Again both hairs cannot be laid before v. So, when it appears the child after v, its color must be free and we can place as many identical nodes connected to v as we wish. □

Now we can prove the main result.

Proof of theorem 3. Let G be a colored caterpillar with hairs of length at most two. We first check in polynomial time, the existence of the subgraph given in figure 17, if this subgraph appears the caterpillar has no colored layout.

Otherwise, construct the caterpillar G' obtained by removing for each node in the backbone, all the fourth, fifth, ..., occurrences of the same hair

of length two, and all the third, fourth, ... , occurrences of the same hair of length one. The number of different colored hairs of length 2 is k^2 and the number of different hairs of length 1 is k, so G' has degree bounded by a constant $3k^2 + 2k + 2$. Then use the algorithm given in [KS96] to decide in polynomial time whether G' has a colored layout (see figure 19). By lemmas 14 and 15 if G' has a colored layout, so does the initial caterpillar G. □

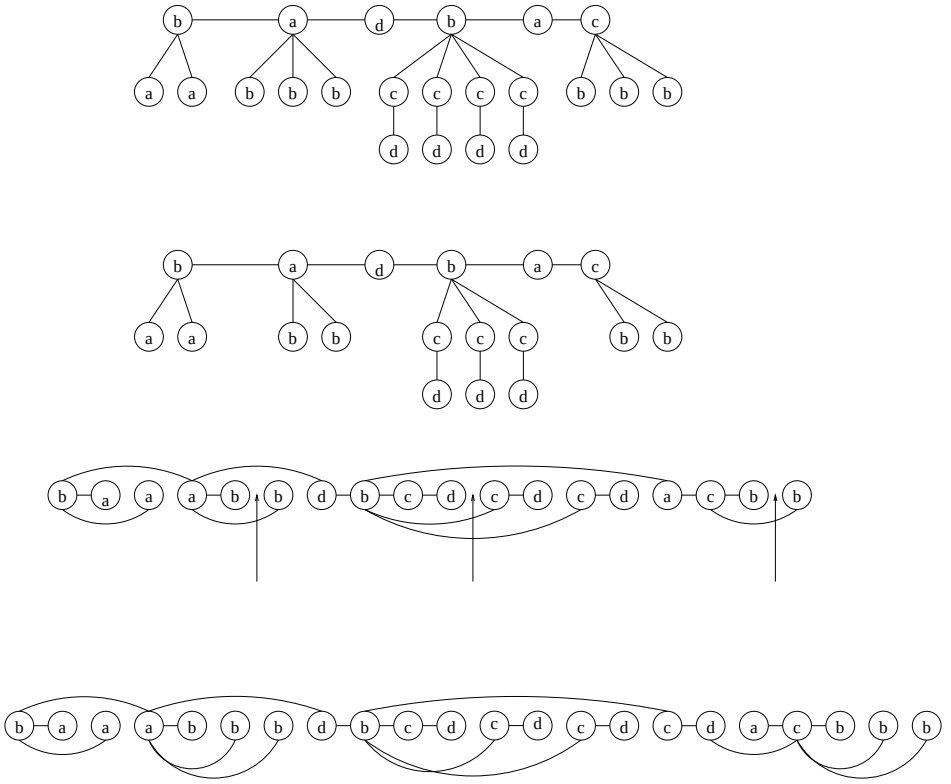

Figure 19: Algorithm for caterpillar with hairs of length 2 and 4 colors.

References

[APSZ81] S.F. Assman, G.W. Peck, M.M. Syslo, and J. Zak. The bandwidth of caterpillars with hair of lengths 1 and 2. *SIAM Journal on Algebraic and Discrete Methods*, 2:387–393, 1981.

[BdF95] H. Bodlaender and B. de Fluiter. Intervalizing k-colored graphs. In Z. Fulop and F. Gecseg, editors, *International Colloquium on Automata Languanges and Programming*, volume 944, pages 87–98. Springer-Verlag, Lecture Notes in Computer Science, 1995.

[BFH95] H. Bodlaender, M.R. Fellows, and M.T. Hallet. Beyond NP-completeness for problems of bounded width: hardness for the W-hierarchy. In *ACM Symposium on Theory of Computing*, pages 449–458, 1995.

[Din96] M.J. Dinneen. VLSI Layouts and DNA physical mappings. Technical report, Los Alamos National Laboratory, 1996.

[EST79] J. Ellis, I.H. Sudborough, and J. Turner. The vertex separation and search number of a graph. *Information and Computation*, (113):50–79, 1979.

[FHW93] M.R. Fellows, M.T. Halett, and T. Wareham. DNA physical mapping: Three ways difficult. In T. Lengauer, editor, *1st European Symposium on Algorithms*, volume 726 of *Lecture Notes in Computer Science*, pages 157–168. Springer-Verlag, Berlin, 1993.

[GKS94] M.C. Golumbic, H. Kaplan, and R. Shamir. On the complexity of DNA physical mapping. *Advances in Applied Mathematics*, 15:203–215, 1994.

[Gol80] M.C. Golumbic. *Algorithmic graph theory and perfect graphs*. Academic Press, New York, 1980.

[GS93] M.C. Golumbic and R. Shamir. Complexity and algorithms for reasoning about time: A graph theoretical approach. *Journal of the ACM*, 40:1108–1113, 1993.

[Kin92] N.G. Kinnersley. The vertex separation number of a graph equals its path-width. *Information Processing Letters*, 42:345–350, 1992.

[KP86] L.M. Kirousis and C.H. Papadimitriou. Searching and pebbling. *Theoretical Computer Science*, 47:205–216, 1986.

[KS96] H. Kaplan and R. Shamir. Physical maps and interval sandwich problems: Bounded degrees helps. Technical report, Shackler Faculty of Exact Sciences, Tel Aviv, 1996.

[Len81] T. Lengauer. Black-white pebbles and graph separation. *Acta Informatica*, 16:465–475, 1981.

[Mon86] B. Monien. The bandwidth minimization problem for caterpillars with hair length 3 is NP-complete. *SIAM Journal on Algebraic and Discrete Methods*, 7(4):505–512, 1986.

[SM97] J.C. Setubal and J. Meidanis. *Introduction to Computational Molecular Biology.* PWS Publishing Company, Boston, 1997.