Complexity results for the optimum cost chromatic partition problem

Klaus Jansen[1]

Fachbereich IV - Mathematik, Universität Trier, 54 286 Trier, Germany, email: jansen@dm3.uni-trier.de

Abstract. In this paper, we study the optimum cost chromatic partition (OCCP) problem for several graph classes. The OCCP problem is the problem of coloring the vertices of a graph such that adjacent vertices get different colors and that the total coloring costs are minimum.
First, we prove that the OCCP problem for cographs and graphs with constant treewidth k can be solved in $O(|V|+|E|)$ and $O(|V|\cdot(\log|V|)^{k+1})$ time, respectively. Next, we study an ILP formulation of the OCCP problem given by Sen et al. [27]. We show that the corresponding polyhedron contains only integral 0/1 extrema if and only if the graph G is a diamond - free chordal graph. Furthermore, we prove that the OCCP problem is NP-complete for bipartite graphs. On the other hand, the OCCP problem for complements of bipartite and triangle free graphs can be solved using a matching algorithm. Finally, we show that the precoloring extension and the OCCP problem are NP-complete for permutation graphs.

1 Introduction

In this paper, we study the optimum cost chromatic partition (OCCP) problem for several graph classes. The OCCP problem can be described as follows: Given a graph $G=(V,E)$ with n vertices and a sequence of coloring costs $(k_1,\ldots,k_n)$, find a feasible coloring $f(v)$ for each vertex $v\in V$ such that the total coloring costs $\sum_{v\in V} k_{f(v)}$ are minimum. A coloring $f: V\to\{1,\ldots,n\}$ is feasible if adjacent vertices have different colors. Alternatively, the OCCP problem can be formulated as follows: Given a graph $G=(V,E)$ with n vertices and a sequence of coloring costs $(k_1,\ldots,k_n)$, find a partition into independent sets $U_1,\ldots,U_s$ such that $\sum_{c=1}^{s} k_c\cdot|U_c|$ is minimum. We may assume that $k_c\le k_d$ whenever $c<d$.

The OCCP problem introduced by Supowit [28] corresponds to a VLSI layout problem. A net is a group of two terminals (e.g. points on a circle or points on two opposite parallel lines) that need to be electrically connected. A routing segment is a line that connects two terminals of a net. Given a netlist and an associated cost value for each layer, the objective is to partition the net list and place them on different layers in such a way that routing segments do not cross on the same layer and that the total costs are minimum. Therefore, this layout problem is equivalent to the OCCP problem (restricted e.g. to circle graphs or to permutation graphs).

Another application is given by Kroon et al. [24]. The OCCP problem for interval graphs is equivalent to the Fixed Interval Scheduling Problem (FISP) with machine dependent processing costs. In this scheduling problem each job $j\in J$ must be executed during a given time interval (s_j,f_j). We assume that a sufficient number of machines is available and that each job must be executed by one of the machines. If job j is executed by machine c, then the associated processing costs are k_c. The objective is to find a feasible schedule for all jobs with minimum total processing costs. Other variants of the FISP have been considered in the literature (e.g. by Arkin and Silverberg [1], Doneti and Emmons [11], Fischetti, Martello and Toth [12–14] and Kolen and Kroon [22, 23]).

A generalization of the OCCP problem contains a graph $G=(V,E)$ and a $(n\times m)$ - cost-matrix $(k_{i,c})$ with unrelated costs $k_{i,c}$ to execute job i on machine c. The general optimum cost chromatic partition problem (GOCCP) can be formulated as follows: Find a partition of the graph G into independent sets $U_1,\ldots,U_s$ such that $\sum_{c=1}^{s}\sum_{i\in U_c} k_{i,c}$ is minimum.

It is not difficult to see that the OCCP problem is NP-complete for arbitrary graphs. Sen et al. [27] proved that the OCCP problem for circle graphs is NP-complete. Moreover, they considered an integer linear program formulation of the OCCP problem. The polytope corresponding to the constraints of this ILP contains only integral (0/1) vertices, if the cartesian product $G \times K_{|V|}$ is a perfect graph. Sen et al. [27] proved that $G \times K_{|V|}$ is perfect if G is a tree and that there exists a perfect graph G such that $G \times K_{|V|}$ is not perfect.

Kroon et al. [24] studied the OCCP problem for interval graphs and trees. They showed that the problem restricted to trees can be solved in linear time and that the problem restricted to interval graphs is NP-complete even if there are only four different values for the coloring costs. For interval graphs G, they proved that the zero-one matrix corresponding to the constraints of the ILP is perfect, if and only if $G \times K_{|V|}$ does not contain an odd cycle of size 7 or more as induced subgraph.

If there are only two different values for the coloring costs, then the OCCP problem is equivalent to the maximum q-colorable subgraph problem. Suppose that the first q costs are equal and that the last $n-q$ costs are equal ($k_1 = \ldots = k_q < k_{q+1} = \ldots = k_n$). Then, we get an optimum solution if the maximum q-colorable subgraph is colored with the colors $1, \ldots, q$ and if the other vertices are colored with the remaining colors. The maximum q-colorable subgraph problem has been studied extensively by Frank [15], Gavril [17], Yannakakis and Gavril [30], Jansen et al. [20] and Chang et al. [8]. Using these studies, the OCCP problem with two different values for the coloring costs is NP-complete even for split graphs, undirected path graphs and their complements and k-trees with unbounded k.

In this paper, we study the complexity of the OCCP problem for several graph classes. In Section 2 we show that the OCCP problem for cographs can be solved in linear time and that the GOCCP problem for cographs is NP-complete. Next, in Section 3 we propose an algorithm for graphs with constant treewidth k that runs in $O(|V|(\log|V|)^{k+1})$ time. Furthermore, we prove that an optimum solution of the OCCP problem restricted to graphs with constant treewidth needs at most $O(\log|V|)$ different colors. Moreover, we give an example that there exists a tree and a sequence of coloring costs such that each optimum solution needs $\Omega(\log|V|)$ different colors. The GOCCP problem can be solved in $O(m^{k+1}|V|)$ time for graphs with constant treewidth k.

In Section 4, we study the integer program formulation given by Sen et al. [27]. We show that the cartesian product $G \times K_m$ for $m \geq 3$ is perfect, if and only if G is a diamond - free chordal graph (a generalization of forests). Next, we show that these graphs have a special tree decomposition. Furthermore, we prove that the polyhedron corresponding to the constraints of the ILP contains only integral extrema, if and only if G is a diamond - free chordal graph. For diamond - free chordal graphs, the GOCCP problem can be solved in polynomial time using minimum weighted matchings in bipartite graphs.

In Section 5, we prove that the OCCP problem for bipartite graphs is NP-complete. For the reduction we use the 1 - precoloring extension problem (for short: 1-PrExt) that is NP-complete for bipartite graphs [7]. The problem 1-PrExt is formulated as follows: Given an integer $m \geq 3$, an undirected graph $G = (V, E)$ with $|V| \geq m$ and m vertices $v_1, \ldots, v_m \in V$, decide whether there exists a m-coloring of G such that each vertex v_i receives color i, for $1 \leq i \leq m$.

In Section 6, we prove that the OCCP problem for complements of bipartite graphs or triangle free graphs can be solved in polynomial time. Furthermore, the list coloring problem and the GOCCP problem are NP-complete even for an union of two complete graphs. The list coloring problem is defined as follows: Given a graph $G = (V, E)$ with color sets $S_v \subset \{1, \ldots, m\}$, decide whether there exists a graph coloring $f : V \to \{1, \ldots, m\}$ with $f(v) \in S_v$. Finally, in Section 7, we show that the

OCCP and the 1 - precoloring extension problem are NP-complete for permutation graphs. The first result solves an open question in [27] and the second an open question in [18].

2 Cographs

In this section, we show that the OCCP problem for cographs can be solved in linear time $O(|V|+|E|)$. Cographs can be generated by disjoint union and product operations on graphs starting with single vertex graphs and can be represented about these operations. For graphs $G_i = (V_i, E_i)$ with $V_1 \cap V_2 = \emptyset$ the union of G_1 and G_2, $\cup(G_1, G_2)$ is given by $(V_1 \cup V_2, E_1 \cup E_2)$. The product of G_1 and G_2, denoted by $\times(G_1, G_2)$ is obtained by first taking the union of G_1 and G_2, and then adding all edges $\{v_1, v_2\}$ with $v_i \in V_i$. The product of three or more graphs $G_1, \ldots, G_r$ is obtained similary: take the disjoint union and add all edges between vertices in different graphs G_i.

Definition 1. *The class of cographs is the smallest set of graphs, fulfilling the following rules:*

1. *Every graph $G = (V, E)$ with one vertex and no edges ($|V| = 1$ and $|E| = 0$) is a cograph.*
2. *If $G_1 = (V_1, E_1)$ is a cograph, $G_2 = (V_2, E_2)$ is a cograph, and V_1 and V_2 are disjoint sets, then $\cup(G_1, G_2)$ is a cograph.*
3. *If $G_1 = (V_1, E_1)$ is a cograph, $G_2 = (V_2, E_2)$ is a cograph, and V_1 and V_2 are disjoint sets, then $\times(G_1, G_2)$ is a cograph.*

Alternatively, a graph is a cograph, if it does not contain a path with four vertices P_4 as an induced subgraph. Many NP-complete problems are polynomial time solvable on cographs; but there are some expections, e.g. achromatic number [3], list coloring [20] and partition into three independent sets of size at most ℓ [6] are NP-complete for cographs.

To each cograph G one can associate a corresponding rooted tree T, called the *cotree* of G, in the following way. Each non-leaf node in the tree is labeled with either "$\cup$" (union-nodes) or "$\times$" (product-nodes) and each leaf is labeled with a vertex of G. Each non-leaf node has two or more children. Let T_x be the subtree of T rooted at node x and let V_x be the set of vertices corresponding to the leaves in T_x. Then, each node x of the cotree corresponds to the cograph $G_x = (V_x, E_x)$. An union-node (product-node) corresponds to the disjoint union (product) of the cographs, associated with the children of the node. Finally, the cograph that is associated with the root of the cotree is just G, the cograph represented by this cotree. We assume that the union and product nodes in the cotree alternate on each path from a leaf to the root. In [10], it is shown that one can decide in $O(|V| + |E|)$ time, whether a graph is a cograph, and build a corresponding cotree.

In the following, we study an algorithm A that computes a sequence $(U_1, \ldots, U_m)$ of maximum independent sets in a given graph G.

(1) set $I = V$, $i = 0$
(2) while $I \neq \emptyset$ do begin
(2.1) let $G[I]$ be the subgraph of G induced by the set I,
(2.2) compute a maximum independent set U in $G[I]$,
(2.3) set $i = i + 1$, $U_i = U$, $I = I \setminus U$
(3) end

A maximum independent set can be computed in linear time $O(|V|)$ for cographs using the cotree representation. The length of the sequence m for a cograph G is equal to the chromatic number $\chi(G)$, since $\chi(G[V \setminus U]) = \chi(G) - 1$ for each maximum independent set U in G. Therefore, algorithm

A runs in $O(|V|^2)$ time on cographs. First, we show that algorithm A is optimal for cographs and later we improve the algorithm to get the time complexity $O(|V|+|E|)$. We notice that the equality $\chi(G[V \setminus U]) = \chi(G) - 1$ holds also for each maximal independent set U in a cograph G. Therefore, we may assume that the length of the optimum list of independent sets for the OCCP problem restricted to cographs is equal to $\chi(G)$.

Lemma 1. *Let $(U_1, \ldots, U_{\chi(G)})$ be the list of independent sets generated by algorithm A for a cograph G. If $(A_1, \ldots, A_{\chi(G)})$ is an optimum list of independent sets for the OCCP problem, then*

$$\sum_{i=1}^{\chi(G)} k_i|U_i| = \sum_{i=1}^{\chi(G)} k_i|A_i|.$$

Proof. We prove this assertion by induction on the height $h(T)$ of the corresponding cotree T. For $h(T) = 1$, the corresponding cograph is a single vertex graph and, therefore, the assertion is clear.

We consider now the root r of a cotree T with $h(T) > 1$. Let $1, \ldots, \ell$ be the children of r. W.l.o.g. we assume that A_i is a maximal independent set in $G[V \setminus (A_1 \cup \ldots \cup A_{i-1})]$ for $1 \leq i \leq \chi(G)$.

Suppose that r is labeled with $\cup$. Then, each independent set A_i (and also U_i) is a disjoint union of sets $A_i^{(1)}, \ldots, A_i^{(\ell)}$ (or $U_i^{(1)}, \ldots, U_i^{(\ell)}$) where each set $A_i^{(k)}$ ($U_i^{(k)}$) is a maximal independent set in G_k. By induction, the sequences $(A_1^{(k)}, \ldots, A_{\chi(G)}^{(k)})$ and $(U_1^{(k)}, \ldots, U_{\chi(G)}^{(k)})$ have the same costs with respect to G_k for $1 \leq k \leq \ell$. Therefore, $\sum_{j=1}^{\chi(G)} k_j|A_j^{(k)}| = \sum_{j=1}^{\chi(G)} k_j|U_j^{(k)}|$. This implies that

$$\sum_{j=1}^{\chi(G)} k_j|A_j| = \sum_{j=1}^{\chi(G)} k_j(|A_j^{(1)}| + \ldots + |A_j^{(\ell)}|) =$$
$$\sum_{j=1}^{\chi(G)} k_j(|U_j^{(1)}| + \ldots |U_j^{(\ell)}|) = \sum_{j=1}^{\chi(G)} k_j|U_j|.$$

Suppose now that r is labeled with $\times$. In this case, each independent set A_i (and U_i) is an independent set in one cograph G_{t_i} where $t_i \in \{1, \ldots, \ell\}$ is a child of r. For a child i of r, let $j_{i,1} < \ldots < j_{i,h_i}$ be the sequence of indices with $A_{j_{i,k}} \subset V_i$ for $1 \leq k \leq h_i$. Then, the list $(A_{j_{i,1}}, \ldots, A_{j_{i,h_i}})$ is an optimum solution of the OCCP problem for the cograph G_i corresponding to child i with cost vector $(k_{j_{i,1}}, \ldots, k_{j_{i,h_i}})$; otherwise $(A_1, \ldots, A_{\chi(G)})$ cannot be an optimum solution. Since $\chi(G_i) \leq h_i$ and $\sum_{i=1}^{\ell} \chi(G_i) = \chi(G) = \sum_{i=1}^{\ell} h_i$, we have $h_i = \chi(G_i)$.

By induction, the solution $(\bar{U}_1^{(i)}, \ldots, \bar{U}_{\chi(G_i)}^{(i)})$ generated by algorithm A for G_i with cost vector $(k_{j_{i,1}}, \ldots, k_{j_{i,\chi(G_i)}})$ has the costs

$$\sum_{p=1}^{\chi(G_i)} k_{j_{i,p}}|\bar{U}_p^{(i)}| = \sum_{p=1}^{\chi(G_i)} k_{j_{i,p}}|A_{j_{i,p}}|.$$

Since the algorithm A chooses maximum independent sets independent on the cost values, we may assume that the set $\{U_k | 1 \leq k \leq \chi(G), U_k \subset V_i\}$ is equal to $\{\bar{U}_k^{(i)} | 1 \leq k \leq \chi(G_i)\}$. The list $(\bar{U}_1, \ldots, \bar{U}_{\chi(G)})$ with

$$\bar{U}_p = \bar{U}_k^{(i)} \text{ for } p = j_{i,k} \text{ and } 1 \leq i \leq \ell, 1 \leq k \leq \chi(G_i),$$

has the same costs as the list $(A_1, \ldots, A_{\chi(G)})$. Next, we sort the list such that the generated list $(U'_1, \ldots, U'_{\chi(G)})$ satisfies the property $|U'_1| \geq \ldots \geq |U'_{\chi(G)}|$. This sorting can be done by simple exchange steps: If $|\bar{U}_i| < |\bar{U}_{i'}|$ and $i < i'$ then replace $\bar{U}_i$ by $\bar{U}_{i'}$ and $\bar{U}_{i'}$ by $\bar{U}_i$. Using the property that $k_i \leq k_{i'}$ for $i < i'$, we obtain $k_i|\bar{U}_i| + k_{i'}|\bar{U}_{i'}| \geq k_i|\bar{U}_{i'}| + k_{i'}|\bar{U}_i|$. This implies that an exchange step does not increase the costs. Therefore, the list $(U'_1, \ldots, U'_{\chi(G)})$ is equivalent to the solution $(U_1, \ldots, U_{\chi(G)})$ with exception of the order of the sets with the same cardinality. Since $(U'_1, \ldots, U'_{\chi(G)})$ has the same costs as the list $(A_1, \ldots, A_{\chi(G)})$, the Lemma is proved. □

In the following, we show how the sequence $(U_1, \ldots, U_{\chi(G)})$ generated by algorithm A can be computed in linear time $O(|V| + |E|)$. First, we give an algorithm that computes the vector of the cardinalities $(|U_1|, \ldots, |U_{\chi(G)}|)$. These vectors are computed recursively for the nodes of the cotree. For a node x we denote by $sizes(x) = (s_1^{(x)}, \ldots, s_{\chi(G_x)}^{(x)})$ the vector of the sorted cardinalities of independent sets generated by algorithm A for G_x. For a leaf x in the cotree, we get $sizes(x) = (1)$.

For an union node x with children $t_1, \ldots, t_\ell$,

$$s_i^{(x)} = \sum_{k|1 \leq k \leq \ell, i \leq \chi(G_{t_k})} s_i^{(t_k)}$$

for $1 \leq i \leq \chi(G_x)$. If the vectors $sizes(t_1), \ldots, sizes(t_\ell)$ are given, then the vector $sizes(x)$ can be computed in time bounded by $O(\sum_{k=1}^{\ell} \chi(G_{t_k})) \leq O(|V_x|)$.

For a product node x with children $t_1, \ldots, t_\ell$, the vector $sizes(x)$ can be computed as follows:

(1) set $S_1 = sizes(t_1)$,
(2) for $i = 1$ to $\ell - 1$ do
(2.1) sort the concatenated vector $S_{i+1} = S_i \cdot sizes(t_{i+1})$,
(3) set $sizes(x) = S_\ell$.

The sorting step (2.1) can be done in time $O(|S_{i+1}|) = O(|S_i| + |sizes(t_{i+1})|) = O(\sum_{j=1}^{i+1} |sizes(t_j)|)$, because the vectors $sizes(t_{i+1})$ and S_i are sorted before. Since $|sizes(t_j)|$ is equal to the chromatic number $\chi(G_{t_j})$, the sum $\sum_{j=1}^{i+1} |sizes(t_j)|$ is bounded by the number of edges between vertices in $G_{t_{i+1}}$ and vertices in $\cup_{j=1}^{i} G_{t_j}$ plus one. Therefore, the sequence $sizes(x)$ can be computed in time proportional to the number of edges generated in the cotree at node x. Summing up, we get for all product nodes (and also for the union nodes that are children of product nodes), $O(|E|)$ time. The time for the root r (if labeled with $\cup$) is at most $O(|V|)$. Therefore, we have proved the assertion.

Lemma 2. *The sequence $sizes(r)$ for a cotree T with root r can be computed in time $O(|V| + |E|)$.*

If we implement each independent set as a linked list with pointers to the first and last element of the list, then the disjoint union of two sets can be computed in constant time. Then, we can compute the list of the independent sets parallel to the computation of the vectors and in the same time $O(|V| + |E|)$.

Theorem 1. *The OCCP problem for cographs can be solved in $O(|V| + |E|)$ time.*

For the next theorem, we use a reduction form the list coloring problem for cographs that is NP - complete [19]. The list coloring problem can be described as follows:

List coloring
Given: A graph $G = (V, E)$ with color sets $S_v \subset \{1, \ldots, m\}$ for each vertex $v \in V$,
Question: Does there exist a graph coloring $f : V \to \{1, \ldots, m\}$ with $f(v) \in S_v$?

Theorem 2. *The GOCCP problem for cographs is NP-complete.*

Proof. Let I be an instance of the list coloring problem containing a cograph $G = (V, E)$ with color sets $S_v \subset \{1, \ldots, m\}$. An instance I' of the GOCCP problem is given by the same graph G and the cost matrix $(k_{i,c})$ with $k_{i,c} = 0$ for $c \in S_i$ and $k_{i,c} = 1$ otherwise. Then, it is easy to prove: I is a yes - instance of the list coloring problem, if and only if the minimum total costs of coloring all vertices in I' is zero. □

3 Graphs with bounded treewidth

The notion of the treewidth of a graph was introduced by Robertson and Seymour [26], and is equivalent to several other interesting graph theoretic notions, for instance the notion of partial k-trees (see e.g., [2, 5]).

Definition 2. *A tree-decomposition of a graph $G = (V, E)$ is a pair $(\{X_i \mid i \in I\}, T = (I, F))$, where $\{X_i \mid i \in I\}$ is a collection of subsets of V, and $T = (I, F)$ is a tree, such that the following conditions hold:*

1. $\bigcup_{i \in I} X_i = V$.
2. *For all edges $(v, w) \in E$, there exists a node $i \in I$, with $v, w \in X_i$.*
3. *For every vertex $v \in V$, the subgraph of T, induced by the nodes $\{i \in I \mid v \in X_i\}$ is connected.*

The treewidth of a tree-decomposition $(\{X_i \mid i \in I\}, T = (I, F))$ is $\max_{i \in I} |X_i| - 1$. The treewidth of a graph is the minimum treewidth over all possible tree-decompositions of the graph.

Note that a tree-decomposition of G with treewidth $\leq k$ can be found, if it exists, in $O(n)$ time [4]. Since each chordal graph has at most $|V|$ maximal cliques, we may assume that the decomposition tree has at most $|V|$ nodes.

First, we consider an optimum solution $(U_1, \ldots, U_m)$ of the OCCP problem with cost vector $(k_1, \ldots, k_n)$. Since $k_1 \leq \ldots \leq k_n$, we may assume that U_i is a maximal independent set in $G[V \setminus (U_1 \cup \ldots \cup U_{i-1})]$; otherwise we can make U_i maximal without increasing costs. Using a similar argument, we can generalize a result of Kroon et al. [24] for trees. Let $d(v)$ be the degree of vertex $v \in V$.

Lemma 3. *For a graph $G = (V, E)$ there exists a coloring f of minimum costs such that $f(v) \leq d(v) + 1$ for all vertices $v \in V$.*

Proof. Let f be a coloring of minimum costs, and suppose that there is a vertex v with $f(v) = c > d(v) + 1$. Then, there exists a color c' with $c' \leq d(v) + 1$ such that $c' \neq f(v')$ for all neighbors v' of v. We define the alternative coloring f' of G:

$$f'(w) = \begin{cases} c' & \text{if } w = v, \\ f(w) & \text{if } w \neq v. \end{cases}$$

Clearly, f' is a feasible coloring of G. Furthermore, $k_{c'} \leq k_c$ since $c' < c$. Thus, f' is also a coloring of minimum costs. By repeating this argument, we obtain a feasible coloring of minimum costs satisfying the condition of the lemma. □

Next, we prove that the length of the cost vector can be bounded by $O(\log |V|)$ for graphs with constant treewidth.

Lemma 4. *For a graph $G = (V, E)$ with constant treewidth, there exists a coloring f with minimum costs such that the color $f(v) \leq O(\log |V|)$ for each vertex $v \in V$.*

Proof. For a graph with constant treewidth k, there exists a tree decomposition T with at most $|V|$ nodes. The number of edges in G is bounded by $k|V|$, since every graph with treewidth $\leq k$ is a subgraph of a k-tree and a k-tree has maximal $k|V|$ edges [21]. If we choose one maximal independent set U, then the degree $d(v)$ of each remaining vertex v is decreased by at least one (otherwise U is not maximal).

Next, we prove the following assertion: If we choose $2k\ell$ maximal independent sets $U_1, \ldots, U_{2k\ell}$, then the number of edges in the remaining graph $G[V \setminus (U_1 \cup \ldots \cup U_{2k\ell})]$ is at most $\frac{k|V|}{2^\ell}$.

Let $\bar{n}$ be the number of remaining vertices. We prove the assertion by induction on ℓ.

$\ell = 1$ **and** $\bar{n} \leq \frac{|V|}{2}$**:** In this case, there is a decomposition tree for the induced graph with at most $\frac{|V|}{2}$ nodes and width at most k. This implies that the number of edges is at most $\frac{k|V|}{2}$.

$\ell = 1$ **and** $\bar{n} > \frac{|V|}{2}$**:** In this case, the degree of each remaining vertex is reduced by at least $2k$. This implies that at least $k\bar{n}$ edges are deleted. Therefore, the number of edges is at most

$$k|V| - k\bar{n} \leq k|V| - \frac{k|V|}{2} = \frac{k|V|}{2}.$$

$\ell \to \ell + 1$ **and** $\bar{n} \leq \frac{|V|}{2^{\ell+1}}$**:** Again, there is tree decomposition T of width k for the induced graph with $\bar{n}$ vertices. Therefore, the number of remaining edges is at most $\frac{k|V|}{2^{\ell+1}}$.

$\ell \to \ell + 1$ **and** $\bar{n} > \frac{|V|}{2^{\ell+1}}$**:** By induction, after the choice of $2k\ell$ maximal independent sets, the number of remaining edges is at most $\frac{k|V|}{2^\ell}$. Using the next $2k$ maximal independent sets, the degree of each remaining vertex is reduced by at least $2k$. This implies that at least $k\bar{n}$ edges are deleted in $G[V \setminus (U_1 \cup \ldots \cup U_{2k\ell})]$. Therefore, the number of edges after choosing $2k(\ell+1)$ maximal independent sets is at most

$$\frac{k|V|}{2^\ell} - k\bar{n} \leq \frac{k|V|}{2^\ell} - \frac{k|V|}{2^{\ell+1}} = \frac{k|V|}{2^{\ell+1}}.$$

The assertion above implies that the number of edges is at most $2k-1$ after $2k\lfloor \log |V| \rfloor$ steps. After at most $2k-1$ further steps, the number of remaining vertices is zero. Since k is constant, an optimum solution $(U_1, \ldots, U_m)$ consists of at most $O(\log |V|)$ maximal independent sets. □

It is not difficult to make small modifications to a tree-decomposition, without increasing its treewidth, such that one can see T as a rooted tree, with root $r \in I$, and the following conditions hold:

1. T is a binary tree.
2. If a node $i \in I$ has two children j_1 and j_2, then $X_i = X_{j_1} = X_{j_2}$.
3. If a node $i \in I$ has one child j, then either $X_j \subset X_i$ and $|X_i - X_j| = 1$, or $X_i \subset X_j$ and $|X_j - X_i| = 1$.

We will assume in the remainder that a tree-decomposition of G of this type is given, with treewidth at most k, for some constant k.

For every node $i \in I$, let Y_i be the set of all vertices in a set X_j with $j = i$ or j is a descendant of i in the rooted tree T. Our algorithm is based upon computing for every node $i \in I$ a table $minc_i$. Let m be the number of allowed colors. For each coloring $f : X_i \to \{1, \ldots, m\}$, there is an entry in the table $minc_i$, fulfilling

$$minc_i(f) = \min_{\bar{f}: Y_i \to \{1,\ldots,m\}, \bar{f}(x) = f(x) \forall x \in X_i} \sum_{j=1}^{m} k_j |\{y | y \in Y_i, \bar{f}(y) = j\}|.$$

In other words, for each coloring f of X_i, $minc_i(f)$ denotes the minimum costs over all colorings $\bar{f}$ of Y_i where $\bar{f}$ and f have the same colors for each vertex $x \in X_i$. The tables are computed in a bottom-up manner: start with computing the tables for the leaves, then always compute the table for an internal node later than the tables of its child or children are computed. The following lemma, which is easy to proof, shows how the tables can be computed efficiently:

Lemma 5. (1) *Let i be a leaf in T. Then for all mappings $f : X_i \to \{1, \ldots, m\}$,*

$$minc_i(f) = \begin{cases} \sum_{\ell=1}^{m} k_\ell |\{x \in X_i | f(x) = \ell\}| & \text{if } f \text{ is a feasible coloring,} \\ \infty & \text{otherwise.} \end{cases}$$

(2) *Let i be a node with one child j in T. Suppose $X_i \subseteq X_j$ and $X_j \setminus X_i = \{v\}$. Then for all $f : X_i \to \{1, \ldots, m\}$, $minc_i(f) = \min_{\ell \in C_v} minc_j(f_\ell)$ where $C_v = \{1 \leq \ell \leq m | \nexists v' \in X_j, \{v, v'\} \in E, f(v') = \ell\}$ (the feasible colors for vertex v) and $f_\ell(v) = \ell$ and $f_\ell(v') = f(v')$ for each $v' \in X_i$.*

(3) *Let i be a node with one child j in T. Suppose $X_j \cup \{v\} = X_i$, $v \notin X_j$. For each mapping $f : X_i \to \{1, \ldots, m\}$,*

$$minc_i(f) = \begin{cases} minc_j(f|_{X_j}) + k_{f(v)} & \text{if } f(v') \neq f(v) \text{ for each } v' \in X_j, \{v, v'\} \in E \\ \infty & \text{otherwise.} \end{cases}$$

(4) *Let i be a node with two children j_1, j_2 in T, with $X_i = X_{j_1} = X_{j_2}$. For all mappings $f : X_i \to \{1, \ldots, m\}$,*

$$minc_i(f) = minc_{j_1}(f) + minc_{j_2}(f) - \sum_{\ell=1}^{m} k_\ell |\{x | x \in X_i, f(x) = \ell\}|.$$

Proof. It is easy to see that these conditions hold from the definition. The subtraction in case (4) is needed because the costs for X_i are otherwise counted twice. □

The time complexity of the algorithm is mainly determined by the number of the nodes in the decomposition tree. For each node $i \in I$, at most $O(m^{k+1}k^2)$ operations are executed to compute the table $minc_i$. In total, the algorithm runs in $O(m^{k+1}k^2|V|)$ time if m different colors are allowed. The costs of an optimum coloring are $\min_{f:X_r \to \{1,\ldots,m\}} minc_r(f)$. We note that it is possible to modify the algorithm, such that it also yields a coloring with the minimum costs.

Theorem 3. *The problem OCCP for graphs $G = (V, E)$ with constant treewidth k can be solved in time $O((\log|V|)^{k+1}|V|)$. If the maximum degree $\Delta(G)$ is bounded by a constant, then the problem OCCP can be solved in linear time $O(|V|)$.*

Proof. First, compute a tree decomposition of G with constant width k in linear $O(|V|)$ time [4]. Then, using Lemma 4 we set $m = \bar{c}\lfloor \log|V| \rfloor$ with constant $\bar{c}$. Next, we use the algorithm above that runs in $O(m^{k+1}|V|) = O((\log|V|)^{k+1}|V|)$ time.

Suppose that $\Delta(G)$ is bounded by a constant. Using Lemma 3, there exists a coloring f of minimum costs such that $f(v) \leq d(v) + 1 \leq \Delta(G) + 1$ for all $v \in V$. In this case, we define m as the constant $\Delta + 1$ and obtain an algorithm that runs in $O(|V|)$ time. □

We note that the definition of the table $minc_i$ and the algorithm can be modified such that we obtain the following result:

Theorem 4. *The GOCCP problem for graphs $G = (V, E)$ of constant treewidth k and $(n \times m)$ cost matrix can be solved in $O(m^{k+1}|V|)$ time.*

Using the general approach we get the time complexity $O(m^2|V|)$ for the GOCCP problem restricted to trees. We note that we can improve the time complexity for trees to $O(m|V|)$.

Next, we show that an optimum solution for a tree and also for a graph with constant treewidth can have a logarithmic number of colors.

Theorem 5. *There exists a tree $G = (V, E)$ and a cost vector $(k_1, \ldots, k_{n+1})$ such that the optimum solution for the OCCP problem is a partition of G into $n + 1 = \Omega(\log|V|)$ independent sets.*

Proof. Starting with a single vertex graph $G^0 = (\{v\}, \emptyset)$, we construct recursively a tree $G^{n+1} = (V^{n+1}, E^{n+1})$ (see also G^3 in Figure 1) as follows:

$$V^{n+1} = V^n \cup \{(v, 1), (v, 2) | v \in V^n\}$$
$$E^{n+1} = E^n \cup \{\{v, (v, 1)\}, \{v, (v, 2)\} | v \in V^n\}$$

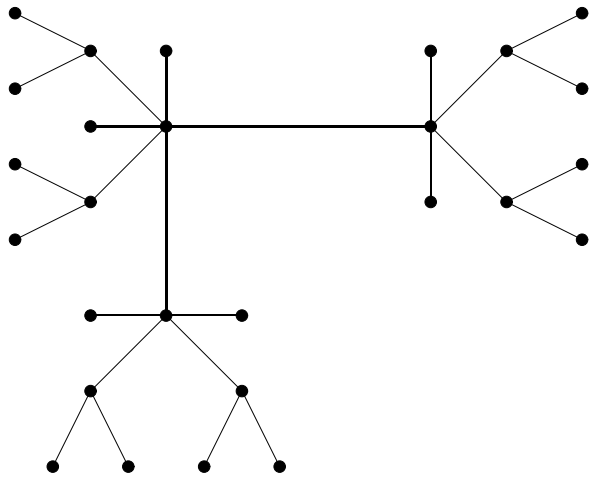

Fig. 1. The constructed graph G^3.

Since each vertex $v \in V^n$ gets two new neighbours, the number of vertices in G^n is equal to 3^n. Furthermore, for $n \geq 1$ the cardinality $\alpha(G^n)$ of a maximum independent set is $2 \cdot 3^{n-1}$. A maximum independent set of this size is the set $\{(v,1),(v,2)|v \in V^{n-1}\}$. Suppose that an independent set U of G^n has a vertices in V^{n-1} with $a > 0$. Then, the cardinality of U can be bounded by $a + 2(3^{n-1} - a)$. Since

$$max_{0<a\leq 3^{n-1}} [a + 2(3^{n-1} - a)] = 2 \cdot 3^{n-1} - 1,$$

a maximum independent set in G^n cannot have vertices in V^{n-1}. This implies that the maximum independent set is uniquely determined in G^n. If we choose iteratively maximum independent sets in G^n, then we obtain a partition into independent sets $U_1, \ldots, U_{n+1}$ such that $U_i = \{(v,1),(v,2)|v \in V^{n-i}\}$ for $1 \leq i \leq n$ and $U_{n+1} = \{v\}$. This implies that $|U_i| = 2 \cdot 3^{n-i}$ for $1 \leq i \leq n$ and $|U_{n+1}| = 1$.

Next, we define a cost vector and prove that this partition is the optimum solution of the OCCP problem for G^n. This shows us that the optimum solution consists of $n+1 = \log_3 |V^n|+1 = \Omega(\log |V^n|)$ independent sets. As cost vector we use $(k_1, \ldots, k_{n+1})$ with $k_1 = 1$ and

$$k_{i+1} = \sum_{j=0}^{i} (\frac{1}{3^n})^j$$

for $1 \leq i \leq n$. We prove that $(U_1, \ldots, U_{n+1})$ is the uniquely determined optimum solution of the OCCP problem. After choosing i independent sets, we have per induction a tree with 3^{n-i} vertices. We show that we must take $2 \cdot 3^{n-i-1}$ independent vertices; otherwise we loose the optimality. Since there exists only one independent set U_{i+1} with $2 \cdot 3^{n-i-1}$ vertices, we must take U_{i+1} and get the tree G^{n-i-1} after $i+1$ steps.

We show that we must take $2 \cdot 3^{n-i-1}$ vertices. If we take $U_{i+1}, \ldots, U_{n+1}$ in G^{n-i}, then we obtain the costs

$$\sum_{k=i+1}^{n} 2 \cdot 3^{n-k} [\sum_{j=0}^{k-1} (\frac{1}{3^n})^j] + 1 \cdot \sum_{j=0}^{n} (\frac{1}{3^n})^j = 3^{n-i} \cdot [\sum_{j=0}^{i} (\frac{1}{3^n})^j] + \sum_{j=0}^{n-i-1} [3^j \cdot (\frac{1}{3^n})^{n-j}].$$

If we choose at most $b \leq 2 \cdot 3^{n-i-1} - 1$ independent vertices in G^{n-i}, then the cost value is at least

$$b[\sum_{j=0}^{i} (\frac{1}{3^n})^j] + (3^{n-i} - b)[\sum_{j=0}^{i+1} (\frac{1}{3^n})^j] = 3^{n-i}[\sum_{j=0}^{i+1} (\frac{1}{3^n})^j] - b(\frac{1}{3^n})^{i+1} \geq 3^{n-i}[\sum_{j=0}^{i} (\frac{1}{3^n})^j] + (1 + 3^{n-i-1})(\frac{1}{3^n})^{i+1}.$$

Comparing both cost values, the first partition is better if and only if

$$\sum_{j=i+2}^{n} [3^{n-j} \cdot (\frac{1}{3^n})^j] < (\frac{1}{3^n})^{i+1}.$$

We bound the left side of the inequality by

$$\sum_{j=i+2}^{n} [3^{n-j} \cdot (\frac{1}{3^n})^j] \leq \frac{3^n}{3^{i+2}} (\frac{1}{3^n})^{i+1} [\sum_{j=0}^{\infty} (\frac{1}{3^n})^j - 1].$$

Using the geometric sum, the term at the right side is at most

$$\frac{3^n}{9} (\frac{1}{3^n})^{i+1} \frac{1}{3^n - 1}.$$

Using the fact that $\frac{3^n}{9} < 3^n - 1$ for each $n \geq 1$, we see that $\sum_{j=i+2}^{n} [3^{n-j} \cdot (\frac{1}{3^n})^j]$ is less than $(\frac{1}{3^n})^{i+1}$. Therefore, a choice of $b < 2 \cdot 3^{n-i-1}$ independent vertices after step i cannot create an optimum solution. Therefore, we have to choose exactly $2 \cdot 3^{n-i-1}$ independent vertices of U_{i+1} and obtain the graph G^{n-i-1} after $i+1$ steps. □

4 Perfect matrices

In this section, we study the integer linear program formulation of the OCCP problem proposed in [24, 27]. Let I be an instance of the OCCP problem containing a graph $G = (V, E)$ with n vertices and a sequence of coloring costs $(k_1, \ldots, k_m)$. The objective function and the constraints of the problem can be described as follows:

$$\begin{array}{lll} min \sum_{i=1}^{n} \sum_{c=1}^{m} k_c \, x_{i,c} & & (0) \\ \sum_{c=1}^{m} x_{i,c} = 1 & \text{for } 1 \leq i \leq n & (1) \\ \sum_{i \in C} x_{i,c} \leq 1 & \text{for each clique C in G}, 1 \leq c \leq m & (2) \\ x_{i,c} \in \{0, 1\} & \text{for } 1 \leq i \leq n, 1 \leq c \leq m & (3) \end{array}$$

The total coloring costs are minimized by the objective function (0). The constraints (1) specify that each vertex is colored exactly once and the constraints (2) guarantee that vertices that are connected by an edge are colored differently. Notice that the clique constraints could be replaced by the corresponding edge constraints. Since the number of cliques for arbitrary graphs may be extremely (exponentially) large, it may be better to use the edge constraints. In this paper, we study the structure of the ILP with clique constraints.

We note that the GOCCP problem with cost matrix $(k_{i,c})$ can be solved with the same approach using the objective function

$$min \sum_{i=1}^{n} \sum_{c=1}^{m} k_{i,c} \, x_{i,c}.$$

The coefficient matrix corresponding to the restrictions $(1) - (2)$ is called M. This matrix M is a zero - one matrix. A zero - one matrix M is called *perfect* if the polyhedron $P(M) = \{x | Mx \leq 1, x \geq 0\}$ has only integral extremal points. It follows that the OCCP problem can be solved by applying a linear programming algorithm, if the matrix M is perfect. We define $P_I(M)$ as the convex hull of the set $\{x | Mx \leq 1, x \in \{0, 1\}^{mn}\}$. The submatrix of M that contains the first n rows of M is denoted by M_1, the submatrix that contains the other rows is denoted by M_2. Furthermore, we define $P^*(M) = \{x | M_1 x = 1, M_2 x \leq 1, x \geq 0\}$ and $P_I^*(M)$ as the convex hull of $\{x | M_1 x = 1, M_2 x \leq 1, x \in \{0, 1\}^{mn}\}$.

The first goal of this section is to find necessary and sufficient conditions for the matrix M to be perfect in terms of the associated intersection graph $G(M)$. The second and main goal of this section is a classification of the graphs such that the polyhedron $P^*(M)$ contains only integral extrema. The intersection graph $G(M)$ of a zero - one matrix M is a graph with one vertex for each column of M. Two vertices v and v' are adjacent in $G(M)$ if and only if $M_{rv} = M_{rv'} = 1$ for at least one row r of M. The vertex set of $G(M)$ for our matrix M is equal to $\{(j,c)|1 \leq j \leq n, 1 \leq c \leq m\}$. Two different vertices (j,c) and (j',c') are adjacent if $j = j'$ and $c \neq c'$ or if $c = c'$ and $\{j,j'\}$ is an edge in the original graph G of the OCCP problem. Notice that the subgraph $G_c(M)$ induced by the vertex set $\{(j,c)|1 \leq j \leq n\}$ is equivalent to the original graph G, for each $1 \leq c \leq m$.

The *cartesian product* $G_1 \times G_2 = (V_1 \times V_2, E)$ of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is defined by the edgeset

$$E = \{\{(u_1,u_2),(v_1,v_2)\}|[u_1 = v_1 \wedge \{u_2,v_2\} \in E_2] \vee [u_2 = v_2 \wedge \{u_1,v_1\} \in E_1]\}.$$

A graph $G = (V,E)$ is perfect, if and only if for each subset $V' \subset V$ the chromatic number $\chi(G[V'])$ of the subgraph $G[V']$ induced by V' is equal to the cardinality $\omega(G[V'])$ of a maximum clique in $G[V']$. It follows that the associated intersection graph $G(M)$ is the cartesian product $G \times K_m$ of the original graph G and a complete graph K_m with m vertices. Since the zero - one matrix M is a clique matrix of its associated intersection graph $G(M)$, the matrix M is perfect if and only if the graph $G(M)$ is perfect (Chvatal [9]).

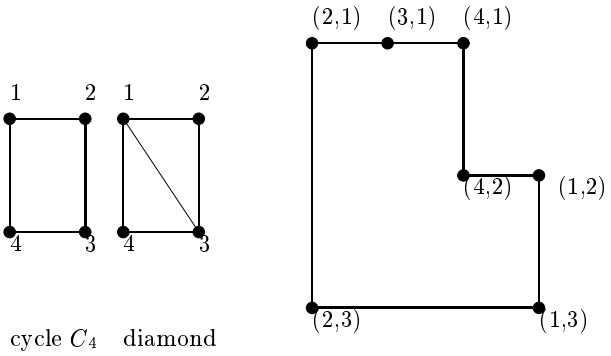

Fig. 2. A cycle C_4 and a diamond generate a C_7 in $G \times K_3$.

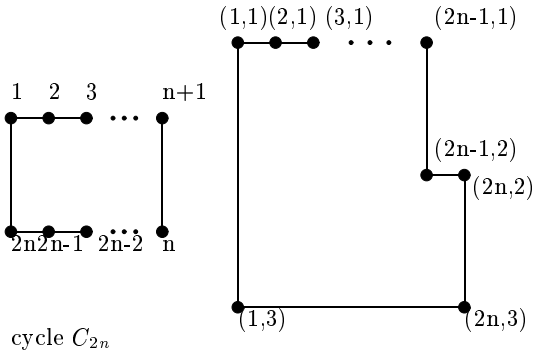

Fig. 3. An even cycle C_{2n} generates an odd cycle C_{2n+3} in $G \times K_3$.

First, we note that the cycle C_4 and the diamond (see Figure 2) generates a cycle C_7 in $G \times K_3$. Furthermore, each cycle C_k ($k \geq 4$) generates a cycle of length $k+3$ in $G \times K_3$ (see Figure 3). This implies that each cycle with even length creates an odd cycle in $G \times K_3$. We may assume that G is a diamond - free chordal graph; otherwise $G \times K_m$ cannot be perfect. In the following we characterize the graphs G such that $G \times K_m$ remains perfect for each m.

Lemma 6. *If G is a diamond - free chordal graph, then for each maximal clique C in G it holds:*

(1) *each vertex $x \in V \setminus C$ is adjacent to at most one vertex $x_c \in C$,*
(2) *for vertices $x, y \in V \setminus C$ with $\{x, x_c\}, \{y, y_c\} \in E$ and $x_c, y_c \in C$, $x_c \neq y_c$ there exists no path in $G[V \setminus C]$ that connects x with y,*
(3) *for each vertex $c \in C$ the set of neighbours $\Gamma(c) \cap (V \setminus C)$ is a disjoint set of cliques $C_1, \ldots, C_k$ such that*
  (3.1) *$C_1, \ldots, C_k$ are separated by C [there are no paths in $G[V \setminus C]$ that connect vertices in C_i with vertices in C_j for $i \neq j$],*
  (3.2) *$C_i \cup \{c\}$ is a maximal clique in G and if $|C_i| \geq 2$ then C_i is a maximal clique in $G[V \setminus C]$.*

Proof. Let C be a maximal clique in $G = (V, E)$ and let x, y be two vertices in $V \setminus C$.

(1): Suppose that x is adjacent to two vertices $y_1, y_2 \in C$, $y_1 \neq y_2$. Then, using the maximality of C there exists a vertex $z \in C$ with $\{x, z\} \notin E$. In this case, the subgraph induced by $\{x, z, y_1, y_2\}$ is a diamond and we get a contradiction.

(2): Suppose that $x, y \in V \setminus C$ are connected to different vertices $x_c, y_c \in C$. Suppose that $\{x, y\} \in E$. Using fact (1), we have $\{x, y_c\} \notin E$ and $\{y, x_c\} \notin E$. In this case, the subgraph induced by $\{x, y, x_c, y_c\}$ is a C_4. Suppose now that $\{x, y\} \notin E$ but connected by a path in $G[V \setminus C]$. In this case, we choose a shortest path $x, a_1, \ldots, a_m, y$ in $G[V \setminus C]$ that connects x with y. Using this choice, we have $\{x, a_i\} \notin E$ for $i \geq 2$, $\{y, a_i\} \notin E$ for $i \leq m-1$ and $\{a_i, a_{i+j}\} \notin E$ for $1 \leq i < i+j \leq m$ and $j \geq 2$ (otherwise there exists a shorter path). If $\{a_1, y_c\} \in E$ or $\{a_m, x_c\} \in E$, then the subgraph induced by $\{x, a_1, x_c, y_c\}$ or the subgraph induced by $\{y, a_m, x_c, y_c\}$ is a cycle C_4, respectively. If $\{a_1, x_c\} \in E$, then we define $x' = a_1$ (otherwise $x' = x$). If $\{a_m, y_c\} \in E$, then we define $y' = a_m$ (otherwise $y' = y$). Then, we use this argument iteratively and obtain either directly a cycle C_4 or one of the following cases:

Case A: $\{x', y'\} \in E$: In this case, the subgraph induced by $\{x_c, y_c, x', y'\}$ is a C_4.

Case B: there exists one vertex z between x' and y'. Since G contains no odd cycle C_5, there must be either an edge between z and x_c or between z and y_c. In both cases, we can apply the argument once again and obtain case A.

Case C: there are two vertices z_1, z_2 between x', y'. Since we cannot apply the argument above, the vertices z_1 and z_2 are not adjacent to x_c or y_c. In this case, the subgraph induced by $\{x_c, y_c, x', y', z_1, z_2\}$ creates a cycle C_6 in G.

Case D: there are more than two vertices between x' and y' (see Figure 4). Let $x', b_1, \ldots, b_{m'}, y'$ be the remaining path in $G[V \setminus C]$ with $m' \geq 3$. Since we cannot the argument above, b_1 and $b_{m'}$ are not connected with x_c or y_c.

Case D.1: all vertices b_i, $2 \leq i \leq m'-1$ are not connected with x_c or y_c. In this case, the subgraph induced by $\{x_c, y_c, x', y', b_1, \ldots, b_{m'}\}$ is a cycle with even length $m'+4$.

Case D.2: there exists at least one vertex b_i, $2 \leq i \leq m'-1$ such that $\{b_i, x_c\} \in E$ or $\{b_i, y_c\} \in E$. Note that exactly one of these edges exists. We choose the vertex b_i with the smallest index that

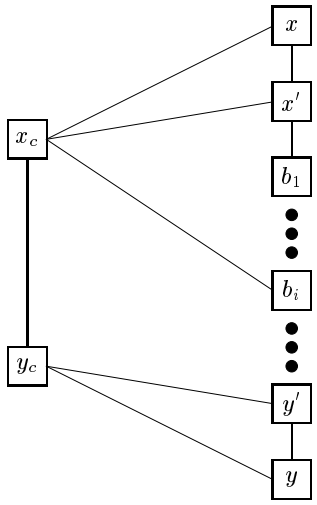

Fig. 4. The vertex b_i is the smallest vertex with an edge to x_c or y_c.

satisfies this property. If $\{x_c, b_i\} \in E$, then the subgraph induced by $\{x_c, x', b_1, \ldots, b_i\}$ (see also Figure 4) is an even cycle. If $\{y_c, b_i\} \in E$, then the subgraph $G[\{x_c, x', b_1, \ldots, b_i, y_c\}]$ is also a cycle of even length.

In all cases $A - D.2$, we have found a cycle of even length in G. This proves property (2).

(3): To prove the third condition, first we consider two vertices $x, y \in V \setminus C$ that are connected to the same neighbour $c \in C$. Suppose that $\{x, y\} \notin E$ but there exists a path in $G[V \setminus C]$ that connects x with y. We choose a shortest path $x, a_1, \ldots, a_m, y$ in $G[V \setminus C]$ that connects x with y.

Case A: each a_i is connected with $c \in C$. In this case, the subgraph $G[\{c, x, a_1, a_2\}]$ (if $m \geq 2$) or $G[\{c, x, a_1, y\}]$ (if $m = 1$) is a diamond.

Case B: there exists a vertex a_i with $\{a_i, c\} \notin E$. In this case, there exist vertices $x' \in \{x, a_1, \ldots, a_{i-1}\}$ and $y' \in \{a_{i+1}, \ldots, a_m, y\}$ that are connected with c. We choose the last vertex x' and the first vertex y' with this property. Then, the subgraph $G[\{c, x', \ldots, a_i, \ldots, y'\}]$ is a cycle of even length in G.

In both cases, we have found a contradiction. Therefore, vertices x and y with $\{x, y\} \notin E$ are separated by the clique C and lie in different connected components of $G[V \setminus C]$.

Let us consider a connected component $Z \subset \Gamma(c) \cap [V \setminus C]$. We show that Z is a clique of G. Let $u, v \in Z$ and $u \neq v$. Both vertices u and v are adjacent to c or there exists a path in $G[V \setminus C]$ that connects u and v. Then, using the fact above u and v must be connected by an edge in G; otherwise G is not a diamond free chordal graph. This implies that the set of neighbours $\Gamma(c) \cap [V \setminus C]$ forms a disjoint union of cliques $C_1, \ldots, C_k$ that satisfies condition (3.1).

Tp prove condition (3.2), first we consider the clique $C' = C_i \cup \{c\}$. If C' is not maximal, there exists a vertex $x \in C$, $x \neq c$ or a vertex $x \in V \setminus (C \cup C_i)$ that is adjacent to all vertices in C'. The first case cannot occur using fact (1). In the second case, x must be in $\Gamma(c) \cap [V \setminus C]$. Since x is connected to all vertices in C_i, it holds that $x \in C_i$. Therefore, the second case cannot occur and $C_i \cup \{c\}$ is a maximal clique. Next, we consider a clique C_i with at least two vertices c_1 and c_2 in C_i. If C_i is not maximal with respect to $G[V \setminus C]$, there exists a vertex $x \in V \setminus (C \cup C_i)$ that is adjacent to all vertices in C_i. In this case, $G[\{c, c_1, c_2, x\}]$ is a diamond. □

Next, we define the graph class *tree - chordal* and show a classification of the perfect graphs $G \times K_m$ for $m \geq 3$.

Definition 3. *A graph $G = (V, E)$ is tree-chordal if and only if there exists a directed tree $T = (U, W)$, an edge labelling $L : W \to V$ and sets $X_u \subset V$ for each $u \in U$ such that*

(1) *each vertex $v \in V$ is in exactly one set X_u,*
(2) *$L((u, v)) \in X_u$ for each edge $(u, v) \in W$,*
(3) *the edgeset E is given by the union of*
  (3.1) *the clique X_r where r is the root of T and*
  (3.2) *the cliques $X_v \cup \{L((u, v))\}$ for each edge $(u, v) \in W$.*

A graph $G = (V, E)$ is forest-chordal if G is the disjoint union of tree - chordal graphs.

We notice that each forest-chordal graph is chordal. In the following we prove our main result.

Theorem 6. *The following statements are equivalent:*

(1) *G is a diamond - free chordal graph.*
(2) *G is a forest chordal graph.*
(3) *$G \times K_m$ (for $m \geq 3$) is a perfect graph.*

We have proved already the implication (3) $\Rightarrow$ (1). If we apply Lemma 6 to each connected component of $G[V \setminus C]$, we get the implication (1) $\Rightarrow$ (2).

Lemma 7. *If G is a diamond free chordal graph, then G is a forest - chordal graph.*

Proof. Consider a connected component Z of G and a maximal clique C in Z. Using Lemma 6, each vertex $x \in Z \setminus C$ is adjacent to at most one vertex $x_c \in C$. For each $c \in C$, the set of neighbours $\Gamma(c) \cap [Z \setminus C]$ forms a disjoint union of cliques $C_1^{(c)}, \ldots, C_{i_c}^{(c)}$. Then, we construct a tree with root r and children $\{(c, j) | c \in C, 1 \leq j \leq i_c\}$ and define $X_r = C$, $X_{(c,j)} = C_j^{(c)}$ and $L((r, (c, j)) = c \in X_r$. Using Lemma 6, two different cliques $C_j^{(c)}$ and $C_{j'}^{(c')}$ are disjoint and separated by C.

Next, we consider a clique $C_j^{(c)}$ and compute recursively a directed tree with root (c, j). To do this, we analyse the set of neighbours $\Gamma(c') \cap [V \setminus (C \cup C_j^{(c)})]$ for each vertex $c' \in C_j^{(c)}$. Since $C_j^{(c)} \cup \{c\}$ is a maximal clique in G, we can apply Lemma 6 again and obtain the properties (1) $-$ (3.2) also for the clique $C_j^{(c)}$. By induction, we get a tree-chordal graph for each connected component Z of G. $\square$

The last implication (2) $\Rightarrow$ (3) for our main theorem is proved by the following Lemma.

Lemma 8. *If $G = (V, E)$ is a forest-chordal graph, then $G \times K_m$ is a perfect graph for each m.*

Proof. We prove this result only for $m \geq |V|$. For $m \leq |V|$, $G \times K_m$ is an induced subgraph of $G \times K_{|V|}$ and, therefore, also perfect. Clearly in $G \times K_m$ with $m \geq |V|$, $\omega(G \times K_m) = \chi(G \times K_m) = m$. Suppose that $H = (V_H, E_H)$ is an induced subgraph of $G \times K_m$ with $\omega(H) = k$. We may assume that G is connected; otherwise we compute a k-coloring parallel for each connected component of G. Let $T = (U, W)$ be the corresponding directed tree with edge labelling $L : W \to V$ and subsets X_u for $u \in U$.Recursively, we determine a k-coloring of H as follows.

First, we consider the set $X_r \subset V$ for the root r of T. Let $H' = (V_{H'}, E_{H'})$ be the subgraph of H induced by $\{(x, i) | x \in X_r, 1 \leq i \leq m\} \cap V_H$. We can describe H' by a bipartite graph $B = (V_B, E_B)$ with

$$V_B = X_r \cup \{1, \ldots, m\},$$
$$E_B = \{\{x, i\} | x \in X_r, 1 \leq i \leq m, (x, i) \in V_{H'}\}.$$

A vertex coloring of H' in $G \times K_m$ corresponds to an edge coloring of the bipartite graph B. The minimum number of colors for an edge coloring of a bipartite graph B can be computed in polynomial

time and is equal to the maximum degree of a vertex v in V_B. The edges incident to v correspond to a clique in H' and, therefore, the chromatic number $\chi(H') \leq k$.

Next, we consider a set X_v for a node $v \in U$ with parent node $p(v) = u$. Suppose that the set $\{(x,i)|x \in X_u, 1 \leq i \leq m\} \cap V_H$ has been colored in a previous step with at most k colors. We define H' as the subgraph of H induced by $\{(x,i)|x \in X_v \cup \{L((u,v))\}, 1 \leq i \leq m\} \cap V_H$ and show that H' can be colored also with at most k colors. Since $L((u,v)) \in X_u$, the vertices $(L((u,v)),i) \in V_H$ are colored previously. Again, we define a bipartite graph $B = (V_B, E_B)$ with

$$\begin{aligned} V_B &= X_v \cup \{L((u,v))\} \cup \{1,\ldots,m\}, \\ E_B &= \{\{x,i\}|(x,i) \in V_{H'}, x \in X_v \cup \{L((u,v))\}, 1 \leq i \leq m\}. \end{aligned}$$

In this case, the edges $\{L((u,v)),i\} \in E_B$ are colored in a previous step with at most k colors. Since the corresponding vertices $(L((u,v)),i)$ form a clique in H, an extension of the precoloring is computable in polynomial time. Since $X_v \cup \{L((u,v))\}$ is a clique in G, we can bound the minimum number of colors in an edge coloring of B or a vertex coloring of H' again by k.

By induction on the height of T, it follows that $\omega(H) = \chi(H)$ and, therefore, that $G \times K_m$ is perfect. □

Now we are ready for our main classification.

Theorem 7. *Let I be an instance of the OCCP problem containing a graph G and a sequence of coloring costs $(k_1,\ldots,k_{|V|})$. Then, we have the following equivalence: G is a diamond - free chordal graph if and only if the polyhedron $P^*(M)$ contains only integral extrema.*

Proof. Let n be the number of vertices. First, suppose that G is diamond - free chordal. Then, the cartesian product $G \times K_n$ is perfect. Using the theorem of Chvatal [9], the matrix M corresponding to the restrictions $(1)-(2)$ is a perfect zero-one matrix. This implies that the polyhedron $P(M) = \{x|Mx \leq 1, x \geq 0\}$ has only integral extrema. Since each extrema in $P^*(M)$ is also an extrema in $P(M)$, the first direction of the proof is shown.

Conversely, suppose that G is not a diamond - free chordal graph. Then, we study two cases.

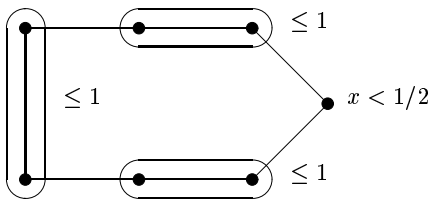

Fig. 5. A C_7 with objective value $3 + x < 3 + 1/2$

Case 1: G is perfect. Then, there exists a diamond or a cycle of even length as induced subgraph in G. This subgraph generates a cycle of odd length in $G \times K_3$. Let $C = \{(i_1,c_1),\ldots,(i_{2\ell+1},c_{2\ell+1})\}$ be such a cycle in $G \times K_3$ (see Figures 2 and 3). The set of variables in the ILP corresponding to the vertices in C is $\bar{X} = \{x_{i_1,c_1},\ldots,x_{i_{2\ell+1},c_{2\ell+1}}\}$. Let M_C be the submatrix of M containing the columns corresponding to the variables in $\bar{X}$. This matrix M_C is not perfect, and the corresponding polyhedron $P(M_C)$ contains a non-integral extremum with components $\bar{x}_{i_j,c_j} = 1/2$. To see that consider the cost values $k_{i_j,c_j} = -1$. An optimum integral solution with respect to $P_I(M_C)$ has the objective value $-\ell$,

and an optimum fractional solution $\bar{x} = (\bar{x}_{i_1,c_1}, \ldots, \bar{x}_{i_{2\ell+1},c_{i_{2\ell+1}}})$ with respect to $P(M_C)$ has the value $-(\ell + 1/2)$. Suppose that one variable x is smaller than $1/2$. Then, (see also Figure 5 for $\ell = 3$) the objective value of the solution must be greater than $-(\ell + 1/2)$.

For each $1 \leq i \leq n$, there are at most two vertices in C with first index equal to i (otherwise we have a triangle). If we have two vertices $(i, c), (i, c') \in C$ with $c \neq c'$, then $\bar{x}_{i,c} + \bar{x}_{i,c'} = 1$. We define

$$A_1 = \{i | 1 \leq i \leq n, (i, c_i) \in C \text{ for exactly one } c_i \in \{1, \ldots, m\}\}$$
$$A_2 = \{i | 1 \leq i \leq n, (i, c) \notin C \text{ for all } c \in \{1, \ldots, m\}\}.$$

Let us analyse the odd cycles in Figures 2 and 3. We may assume that A_1 is a path in G with $c_i = 1$ for each $i \in A_1$. We define the cost matrix $(k_{i,c})$ with

$$k_{i,c} = \begin{cases} -1 & \text{if } (i, c) \in C \\ 0 & \text{otherwise.} \end{cases}$$

Then, we get a feasible solution $x = (\bar{x}_{i,c})$ with respect to $P^*(M)$ as follows: We define $\bar{x}_{i,c} = \frac{1}{2}$ if $(i, c) \in C$ or if $i \in A_1$ and $c = 4$. The cardinality of A_2 is at most $n-4$. Therefore, we can distribute the value 1 to $\bar{x}_{i,5}, \ldots, \bar{x}_{i,n}$ for each $i \in A_2$ such that $\sum_{j=5}^{n} \bar{x}_{i,j} = 1$ and $\bar{x}_{i,5}, \ldots, \bar{x}_{i,n} \leq \frac{1}{n-4}$ for each $i \in A_2$. Let $x^{(int)} \in P_I^*(M)$ be an optimum solution with respect to the cost matrix $(k_{i,c})$. The objective value of $x^{(int)}$ is equal to $-\ell$, and the objective value of x is equal to $-(\ell + 1/2)$. This implies that $P^*(M)$ contains non-integral extrema.

Case 2: G is not perfect. Again, using the theorem of Chvatal [9], the clique matrix M_G corresponding to G is not perfect. Using similar arguments, we can show that $P^*(M)$ contains non-integral extrema. □

In the next theorem, we show that the GOCCP problem can be solved using minimum weighted matchings in bipartite graphs.

Theorem 8. *The GOCCP problem for forest chordal graphs $G = (V, E)$ with cost matrix $(k_{i,c})$ can be solved in polynomial time using weighted matchings in bipartite graphs.*

Proof. The GOCCP problem can be solved independently for each connected component Z of G. We may assume that G is connected. Let $T = (U, W)$ be the tree corresponding to G with edge labelling $L : W \to V$ and sets X_u for $u \in U$. For each vertex x and color c, $K(x, c)$ denotes the costs of coloring vertex x with color c and the subtrees rooted at the children of x as cheap as possible. The idea is to compute the values $K(x, c)$ bottom up in the tree. For a leaf $u \in U$ and vertex $x \in X_u$, $K(x, c) = k_{x,c}$.

Let $u \in U$ be an internal node and $x \in X_u$. Notice that each child v of u in T corresponds to a clique X_v in the neighbour set $\Gamma(L((u, v)))$. Additionally, we compute for a maximal clique X_v at a node v in T a value $\bar{K}(X_v, c)$ that gives the minimum costs of coloring the subtree rooted at node v such that X_v is colored with colors different from c. To do this, we construct a complete bipartite graph $B = (V_B, E_B)$ with $V_B = X_v \cup \{1, \ldots, m\}$, $E_B = \{\{x, j\} | x \in X_v, 1 \leq j \leq m\}$ and edge weights

$$w(x, j) = \begin{cases} K(x, j) \text{ for } j \neq c \\ \infty \qquad \text{otherwise.} \end{cases}$$

The cost value $\bar{K}(X_v, c)$ is given as the weight of a minimum weighted matching in B that can be computed in $O((|X_v| + m)^3)$ time. Let $i_1, \ldots, i_\ell$ be the children of u with $L((u, i_j)) = x$. Then, we have a set of disjoint cliques $X_{i_1}, \ldots, X_{i_\ell}$ where x is adjacent to each vertex in these cliques. Then, we get

$$K(x, c) = k_{x,c} + \sum_{j=1}^{\ell} \bar{K}(X_{i_j}, c).$$

Let r be the root of T. Finally, the optimum value is given by a minimum weighted matching in the bipartite graph $B = (V_r, E_r)$ with $V_r = X_r \cup \{1, \ldots, m\}$, $E_r = \{\{x, j\} | x \in X_r, 1 \leq j \leq m\}$ and edge weights $w(x, j) = K(x, j)$ for $x \in X_r$, $1 \leq j \leq m$. □

5 Bipartite graphs

In this section we prove that the OCCP problem is NP-complete for bipartite graphs. We use the 1 - precoloring extension problem that is NP-complete for bipartite graphs and $m = 3$ proved by Bodlaender, Jansen and Woeginger [7].

1-PrExt for bipartite graphs
Given: A bipartite graph $G = (V, E)$ with vertex set $V = A \cup B$ and edge set $E \subset \{\{v, w\} | v \in A, w \in B\}$ and three specified vertices $a_1, a_2, a_3 \in A$,
Question: Does there exist a 3-coloring of G with $f(a_1) = 1$, $f(a_2) = 2$ and $f(a_3) = 3$?

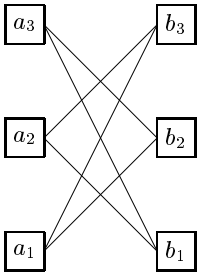

Fig. 6. The control graph.

Theorem 9. *The OCCP problem for bipartite graphs is NP-complete if there are at least four different cost values.*

Proof. The theorem is proved by a reduction from 1-PrExt restricted to bipartite graphs. We may assume that $G = (A \cup B, E)$ contains three further vertices $b_1, b_2, b_3 \in B$ with $\{a_i, b_j\} \in E$ for $1 \leq i \neq j \leq 3$ (see Figure 6). Let n be the number of vertices in G. Clearly, the number n is greater than 6.

Let I be an instance of 1-PrExt containing the bipartite graph $G = (A \cup B, E)$ with $a_1, a_2, a_3 \in A$ and $b_1, b_2, b_3 \in B$ as described above. An instance I' of the OCCP problem is constructed as follows. First, we define a bipartite graph $G' = (V', E')$ with vertex set

$$\begin{aligned} V' = &\{v_{1,j}, v_{2,j} | 1 \leq j \leq 2000n\} \cup \\ &\{v_{3,j'}, v_{4,j'} | 1 \leq j' \leq 100n\} \cup \{v_5, v_6\} \end{aligned}$$

and edge set

$$\begin{aligned} E' = &\{\{v_{1,j}, v_{3,j'}\}, \{v_{2,j}, v_{4,j'}\} | 1 \leq j \leq 2000n, 1 \leq j' \leq 100n\} \cup \\ &\{\{v_5, v_{3,j'}\}, \{v_6, v_{4,j'}\} | 1 \leq j' \leq 100n\} \cup \\ &\{v_5, v_6\}. \end{aligned}$$

The bipartite graph G' illustrated in Figure 7 contains $4000n + 200n + 2$ vertices. Then, we connect G and G' using the following edges:

$$\begin{aligned}\bar{E} = \{&\{a_1, v_{3,j'}\}, \{b_1, v_{4,j'}\}, \{b_1, v_5\}, \{a_2, v_{2,j}\},\\ &\{\{b_2, v_{1,j}\}, \{b_2, v_5\}, \{a_3, v_{3,j'}\}, \{a_3, v_{2,j}\},\\ &\{b_3, v_{1,j}\}, \{b_3, v_{4,j'}\} \mid 1 \leq j \leq 2000n, 1 \leq j' \leq 100n\}\}.\end{aligned}$$

In total, the bipartite graph $\bar{G}$ for I' is given by

$$\bar{G} = (A \cup B \cup V', E \cup E' \cup \bar{E}).$$

The cost values are $k_1 = 1$, $k_2 = 10$, $k_3 = 100$ and $k_4 = 15000n$. A cheap coloring of $\bar{G}$ has to use only three colors; otherwise the costs would be more than $15000n$.

If we use the 3-coloring f for G' as given in Table 1, the total costs are $(4000n + 1) \cdot 1 + 200n \cdot 10 + 1 \cdot 100 \leq 6000n + 101$.

vertices $v' \in V'$	$v_{1,j}, v_{2,j}, v_6$	$v_{3,j'}, v_{4,j'}$	v_5
coloring f	1	2	3

Table 1. A good coloring for G'

We prove the following statement: I is a yes instance of 1-PrExt if and only if the minimum total costs of coloring all vertices in I' don't exceed $6100n + 101$.

Suppose, that f is a 3-coloring for G with $f(a_1) = f(b_1) = 1$, $f(a_2) = f(b_2) = 2$ and $f(a_3) = f(b_3) = 3$. Then, using the 3-coloring for G' as described above (in Table 1), the total costs for G' are equal to $6000n + 101$. This 3-coloring for G' is compatible with the solution of 1-PrExt for G (consider the connecting edges in $\bar{E}$). Furthermore, the costs of the 3-coloring for G can be bounded simply by $100n$. Therefore, the total costs are at most $6100n + 101$.

Conversely, suppose that the minimum costs of coloring all vertices in I' don't exceed $6100n + 101$. Let f be such a coloring. First, we prove that $f(a_1) = f(b_1) = 1$, $f(a_2) = f(b_2) = 2$ and $f(a_3) = f(b_3) = 3$. The key idea is to show that otherwise the minimum costs exceed $6100n + 101$.

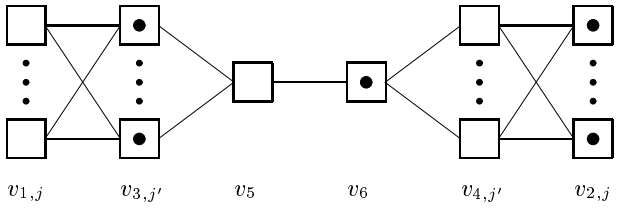

Fig. 7. The constructed graph G' and a feasible 2-coloring of G'.

Case 1: $f(a_3) = 1$. This implies that $f(v_{3,j'}), f(v_{2,j}) \geq 2$. The cheapest coloring (see Figure 7) of the vertices in G' has costs at least

$$\begin{aligned}&(2000n + 100n + 1) \cdot 1 + (2000n + 100n + 1) \cdot 10\\ &= 20000n + 3000n + 100n + 11.\end{aligned}$$

In this case, we obtain additional costs of at least $17000n - 90 \geq 16910n$.

Case 2: $f(a_3) = 2$. This implies that $f(v_{3,j'}), f(v_{2,j}) \in \{1, 3\}$.

Case 2.1: $f(v_{3,j'}) = 3$ for each $1 \leq j' \leq 100n$. The cheapest coloring of the vertices in G' has costs of at least

$$(4000n + 1) \cdot 1 + (100n + 1) \cdot 10 + 100n \cdot 100 = 10000n + 5000n + 11.$$

In this case, we obtain additional costs of at least $8900n - 90 \geq 8810n$.

Case 2.2: $f(v_{3,j'}) = 1$ for at least one $j' \in \{1, \ldots, 100n\}$. This case gives (as in case 1) additional costs of at least $16910n$.

Using the symmetry of the graph G', $f(b_3) \in \{1, 2\}$ implies also additional costs of at least $8810n$. Therefore, we must have $f(a_3) = f(b_3) = 3$.

Case 3: $f(a_2) = 1$. This implies that $f(v_{2,j}) \geq 2$ for each $1 \leq j \leq 2000n$. Again, as in case 1 we obtain additional costs of at least $16910n$.

Case 4: $f(a_2) = 3$. This case is not possible, since $f(b_3) = 3$ and $\{a_2, b_3\} \in E$.

Case 5: $f(b_2) = 1$. This implies that $f(v_{1,j}) \geq 2$ for each $1 \leq j \leq 2000n$ and that $f(v_5) \geq 2$. In this case, the cheapest coloring of the vertices in G' has costs of at least $23100n + 11$. Again, this gives additional costs of at least $16910n$.

Case 6: $f(b_2) = 3$. This case is not possible, since $f(a_3) = 3$ and $\{a_3, b_2\} \in E$.

This case analysis shows that a_2 and b_2 are colored with color 2. Using the control graph (see Figure 6), we obtain that $f(a_1) = f(b_1) = 1$. Using this 3-coloring of the control graph and the edges connecting the graph G with G', the 3-coloring f satisfies the following properties:

(1) $f(v_{3,j'}) \neq 1$, $f(v_{4,j'}) \neq 1$ and $f(v_5) \neq 1$.
(2) $f(v_{2,j}) \neq 2$, $f(v_{1,j}) \neq 2$ and $f(v_5) \neq 2$.
(3) $f(v_{3,j'}) \neq 3$, $f(v_{2,j}) \neq 3$, $f(v_{1,j}) \neq 3$ and $f(v_{4,j'}) \neq 3$.

These properties imply:

(1) $f(v_{1,j}) = f(v_{2,j}) = 1$ for each $1 \leq j \leq 2000n$,
(2) $f(v_{3,j'}) = f(v_{4,j'}) = 2$ for each $1 \leq j' \leq 100n$ and
(3) $f(v_5) = 3$ and $f(v_6) = 1$.

The costs of the coloring f for G' are equal to $6000n + 101$. For the graph G, it remains costs of at most $100n$. Since we have only three colors in G (note that $k_4 = 15000n$) and $f(a_1) = f(b_1) = 1$, $f(a_2) = f(b_2) = 2$ and $f(a_3) = f(b_3) = 3$, the coloring f restricted to G is a solution of the 1-PrExt problem. □

6 Co-bipartite and co-triangle free graphs

In this section we consider complements of bipartite graphs (short: co-bipartite graphs) and complements of triangle free graphs (short: co-triangle free graphs). A graph is a *triangle free graph* if it contains no clique of size three. First, we show that the OCCP problem restricted to these graph classes can be solved in polynomial time. On the other hand, we prove that the list coloring and the GOCCP problem are NP-complete even for an union of two complete graphs and, therefore, also for co-bipartite or co-triangle free graphs.

Since the largest independent sets in a co-triangle free graph $G = (V, E)$ have two vertices, the OCCP problem restricted to co-triangle free graphs can be solved by a matching algorithm. A matching in a graph is an edge set $E' \subset E$ with $|\{v | v \in V \wedge v \in e \text{ for some } e \in E'\}| = 2 \cdot |E'|$. Compute a

matching of maximum cardinality in the complement graph $\bar{G} = (V, \bar{E})$ of G and let D' be the number of edges in this matching. Then, the optimum solution consists of D' independent sets of size 2 and $|V| - 2D'$ independent sets of size 1. The cost value of the optimum solution is given by

$$\sum_{i=1}^{D'} 2k_i + \sum_{i=D'+1}^{|V|-D'} k_i.$$

Since the cardinality matching problem can be solved in $O(|V|^{0.5}|\bar{E}|)$ time (see Micali and Vazirani [25]), we have proven the following theorem.

Theorem 10. *For a co-bipartite or co-triangle free graph $G = (V, E)$, an optimum solution for the OCCP problem can be computed in polynomial time $O(|V|^{0.5}|\bar{E}|)$.*

To prove that GOCCP is NP-complete for an union of two complete graphs, first we show the NP-completeness of list coloring for this graph class. Next, we use the same reduction as in Theorem 5 to show that GOCCP is NP-complete for an union of two complete graphs. For the first reduction, we use a restricted version of 3-SAT (that is also NP-complete [16]):

Restricted 3-SAT
Given: A set of unnegated variables $X = \{x_1, \ldots, x_n\}$ and negated variables $\bar{X} = \{\bar{x}_1, \ldots, \bar{x}_n\}$, a collection of clauses $c_1, \ldots, c_m$ such that
- each clause c_i contains two or three different literals $y_{i,k} \in X \cup \bar{X}$ and
- there are at most three clauses that contain x_j or $\bar{x}_j$ for each $1 \leq j \leq n$.

Question: Does there exist a truth assignment for the variables such that each clause is satisfied?

Theorem 11. *The list coloring problem restricted to an union of two complete graphs is NP-complete.*

Proof. Given an instance I of the restricted 3-SAT problem, we construct an instance I' of the list coloring problem as follows. For each variable x_j we use six vertices $a_j^{(1)}$, $a_j^{(2)}$, $a_j^{(3)}$ and $b_j^{(1)}$, $b_j^{(2)}$, $b_j^{(3)}$ with the color sets:

vertices v	$a_j^{(k)}$	$b_j^{(k)}$
color sets S_v	$\{x_j^{(k)}, \bar{x}_j^{(k)}\}$	$\{\bar{x}_j^{(k)}, x_j^{(k \bmod 3+1)}\}$

Table 2. Color sets for the six vertices

We define $A = \{a_j^{(k)} | 1 \leq j \leq n, 1 \leq k \leq 3\}$ and $B = \{b_j^{(k)} | 1 \leq j \leq n, 1 \leq k \leq 3\}$. The set $A \cup B$ forms the first clique in our graph G. Then, there are only two possibilities to color these six vertices correctly:

(1) $f(a_j^{(k)}) = x_j^{(k)}$, $f(b_j^{(k)}) = \bar{x}_j^{(k)}$ for $1 \leq k \leq 3$,
(2) $f(a_j^{(k)}) = \bar{x}_j^{(k)}$, $f(b_j^{(k)}) = x_j^{(k \bmod 3+1)}$ for $1 \leq k \leq 3$.

For each clause c_i we use one vertex c_i. Let $C = \{c_i | 1 \leq i \leq m\}$. We get an union of two complete graphs with vertex set $V = A \cup B \cup C$ using the second clique $A \cup C$. A clause c_i contains two or three literals $y_{i,k}$. We assign each literal $y_{i,k}$ a number $v_{i,k} \in \{1, 2, 3\}$ (the first, second or third occurrence of the corresponding variable $x_j, \bar{x}_j$) with the property:

$$y_{i,k}, y_{i',k'} \in \{x_j, \bar{x}_j\} \wedge i \neq i' \Longrightarrow v_{i,k} \neq v_{i',k'}.$$

Since there are at most three clauses with x_j or $\bar{x}_j$, such an assignment exists and can computed in polynomial time. If c_i has two literals, then the color set for c_i is $S_{c_i} = \{y_{i,1}^{(v_{i,1})}, y_{i,2}^{(v_{i,2})}\}$. If c_i contains three literals, we use as color set $S_{c_i} = \{y_{i,1}^{(v_{i,1})}, y_{i,2}^{(v_{i,2})}, y_{i,3}^{(v_{i,3})}\}$.

We claim that G has a feasible vertex coloring $f : V \to \{x_j^{(k)}, \bar{x}_j^{(k)} | 1 \leq j \leq n, 1 \leq k \leq 3\}$ with $f(v) \in S_v$ for all $v \in V$ if and only if there is a truth assignment that satisfies all m clauses.

Suppose, we have a truth assignment that satisfies all clauses. Then, we construct a feasible coloring in the following way:

$$f(a_j^{(k)}) = \begin{cases} x_j^{(k)} \text{ if } x_j \text{ is } false, \\ \bar{x}_j^{(k)} \text{ otherwise.} \end{cases}$$

$$f(b_j^{(k)}) = \begin{cases} \bar{x}_j^{(k)} & \text{if } x_j \text{ is } false, \\ x_j^{(k \bmod 3 + 1)} & \text{otherwise.} \end{cases}$$

$$f(c_i) = \begin{cases} y_{i,1}^{(v_{i,1})} \text{ if } y_{i,1} \text{ is } true, \\ y_{i,2}^{(v_{i,2})} \text{ if } y_{i,1} \text{ is } false \text{ and } y_{i,2} \text{ is } true, \\ y_{i,3}^{(v_{i,3})} \text{ otherwise.} \end{cases}$$

Conversely, suppose that there is a feasible coloring for G with $f(v) \in S_v$ for each $v \in V$. We define a truth assignment as follows:

$$x_j = \begin{cases} false \text{ if } f(a_j^{(1)}) = x_j^{(1)}, \\ true \quad \text{otherwise.} \end{cases}$$

We notice that either $f(a_j^{(k)}) = x_j^{(k)}$ for $1 \leq k \leq 3$ or $f(a_j^{(k)}) = \bar{x}_j^{(k)}$ for $1 \leq k \leq 3$. Let $f(c_i) = y_{i,\ell}^{(v_{i,\ell})}$ and $y_{i,\ell} \in \{x_j, \bar{x}_j\}$ with $\ell \in \{1, 2, 3\}$.

Case 1: If $x_j = false$, then $f(a_j^{(k)}) = x_j^{(k)}$ for $1 \leq k \leq 3$. Since $\{a_j^{(k)}, c_i\} \in E$, we have $f(c_i) \notin \{x_j^{(1)}, x_j^{(2)}, x_j^{(3)}\}$. This implies that $f(c_i) = \bar{x}_j^{(v_{i,\ell})}$ and that the clause c_i is satisfied.

Case 2: If $x_j = true$, then $f(a_j^{(k)}) = \bar{x}_j^{(k)}$ for $1 \leq k \leq 3$. Again, using $\{a_j^{(k)}, c_i\} \in E$ we get $f(c_i) \notin \{\bar{x}_j^{(1)}, \bar{x}_j^{(2)}, \bar{x}_j^{(3)}\}$. This implies that $f(c_i) = x_j^{(v_{i,\ell})}$ and that the clause c_i is satisfied. □

Using the reduction in Theorem 5, we obtain:

Corollary 1. *The GOCCP problem restricted to an union of two complete graphs is NP-complete.*

Since an union of two complete graphs is a co-bipartite graph, we get:

Corollary 2. *The list coloring and the GOCCP problem restricted to co-bipartite or co-triangle free graphs are NP-complete.*

7 Permutation graphs

In this section, we show that the OCCP and the precoloring extension problem are NP-complete for permutation graphs. The first result solves an open question in [27] and the second an open question in [18]. The proof uses a technique of Wagner [29] who proved that the cochromatic number problem is NP-complete for permutation graphs.

Let $\Pi = (i_1, \ldots, i_n)$ be a permutation of $\{1, \ldots, n\}$. We denote by $\Pi^{-1}(i)$ the position of i in Π. A graph $G = (V, E)$ with $V = \{1, \ldots, n\}$ is a permutation graph if there is a permutation Π such that $\{i, j\} \in E$ if and only if $i > j$ and $\Pi^{-1}(i) < \Pi^{-1}(j)$. In our proof, we use the point model, where the

permutation $\Pi = (i_1, \ldots, i_n)$ is interpreted as a set of points $\{p_j = (j, \Pi^{-1}(j)) | 1 \leq j \leq n\}$. An independent set $U = \{a_1, \ldots, a_u\}$ corresponds to the point set $S_U = \{(a_1, \Pi^{-1}(a_1)), \ldots, (a_u, \Pi^{-1}(a_u))\}$ in increasing order ($a_1 < \ldots < a_u$ and $\Pi^{-1}(a_1) < \ldots < \Pi^{-1}(a_u)$). Conversely, a clique C corresponds to a point set S_C in decreasing order.

For our NP-completeness result, we use a reduction from 3-SAT.

3-SAT

Given: A set of unnegated variables $X = \{x_1, \ldots, x_m\}$ and negated variables $\bar{X} = \{\bar{x}_1, \ldots, \bar{x}_m\}$, a collection of clauses $c_1, \ldots, c_n$ over $X \cup \bar{X}$ such that each clause c_i contains three different literals $y_{i,1}, y_{i,2}, y_{i,3} \in X \cup \bar{X}$.

Question: Does there exist a truth assignment for the variables such that each clause is satisfied?

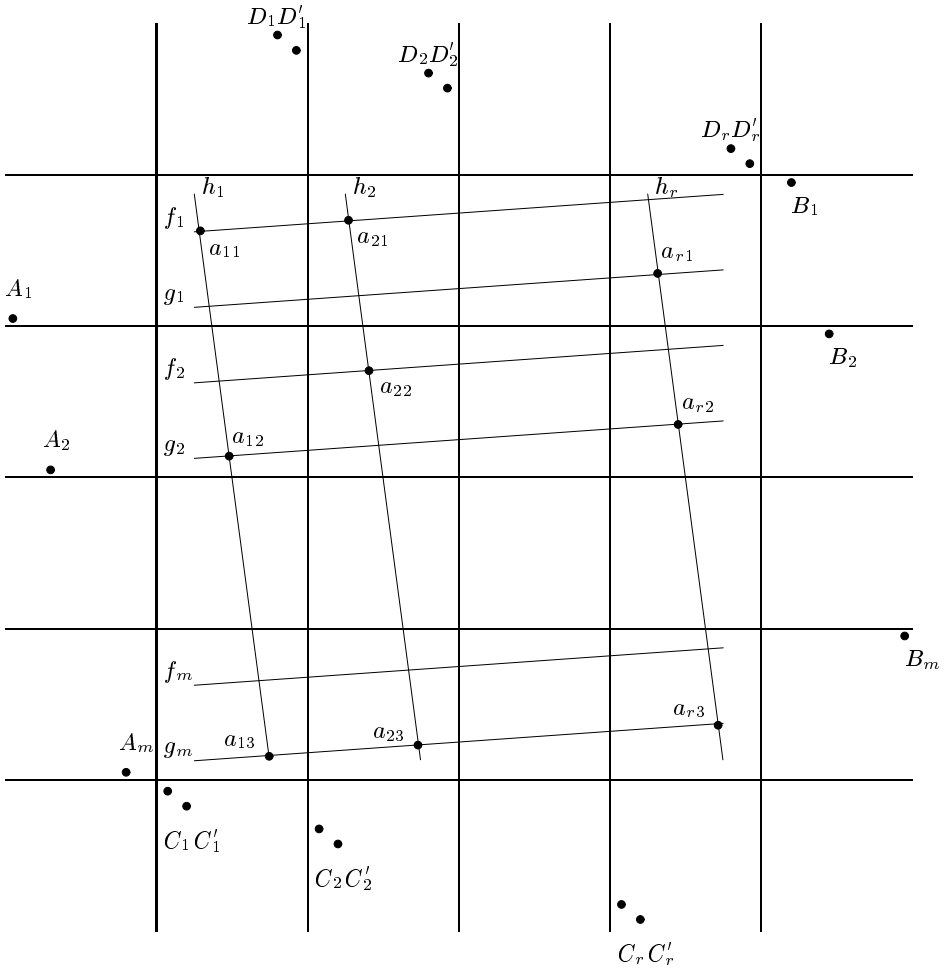

Fig. 8. The point set P constructed for the 3-SAT instance $(\bar{x}_1 \vee x_2 \vee x_m) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_m) \wedge \ldots \wedge (x_1 \vee x_2 \vee x_m)$

Theorem 12. 1-*PrExt is NP-complete for permutation graphs.*

Proof. The theorem is proved by a reduction from 3-SAT. Let I_1 be an instance of 3-SAT containing a collection of n clauses $c_1, \ldots, c_n$ over a set of unnegated and negated variables $\{x_1, \bar{x}_1, \ldots, x_m, \bar{x}_m\}$. In the first step, we construct a new instance I_1' with $r = (m+1)n$ clauses such that $c'_{in+j} = c_j$ for

$0 \leq i \leq m$ and $1 \leq j \leq n$. The literals of clause c'_i are denoted by $y'_{i,1}, y'_{i,2}$ and $y'_{i,3}$. Clearly, I_1 is a yes-instance if and only if I'_1 is a yes-instance of 3-SAT. An instance I_2 of the precoloring problem is constructed as follows. The permutation graph G is given by the corresponding point set

$$P = \bigcup_{i=1}^{m} (\{A_i, B_i\}) \cup \bigcup_{i=1}^{r} (\{C_i, C'_i, D_i, D'_i\} \cup \{a_{i1}, a_{i2}, a_{i3}\})$$

where

- the points $A_i, B_i, C_i, C'_i, D_i, D'_i$ are placed as given in Figure 8.
- a_{ik} is situated in the crossing of the decreasing line marked by h_i and the increasing line marked by f_ℓ if $y'_{i,k} = \bar{x}_\ell$ or g_ℓ if $y'_{i,k} = x_\ell$.

Furthermore, the precolored vertices are $B_1, \ldots, B_m$ and $C_1, C'_1, \ldots, C_r, C'_r$ and the number of used colors is equal to $m + 2r$.

As illustrated in Figure 8, the point set $\{a_{ik} | 1 \leq i \leq r, 1 \leq k \leq 3\}$ can be divided into r vertical stripes and m horizontal stripes. Since the instance I'_1 consists of $m + 1$ copies of instance I_1, we can subdivide the $r = (m + 1)n$ vertical stripes into $m + 1$ sections of n consecutive stripes.

We prove the following **claim**: I'_1 is a yes-instance of 3-SAT if and only if there is a $m+2r$-coloring f of G such that $f(B_i) = i$ for $1 \leq i \leq m$, $f(C_j) = m + 2j - 1$ and $f(C'_j) = m + 2j$ for $1 \leq j \leq r$.

Suppose that α is an assignment to the truth variables which makes the clauses $c'_1, \ldots, c'_r$ true. First, we define $f(A_i) = i$ for $1 \leq i \leq m$, $f(D_j) = m + 2j - 1$ and $f(D'_j) = m + 2j$ for $1 \leq j \leq r$. Let $i \in \{1, \ldots, m\}$. If $\alpha(x_i) = false$, then all points situated on f_i are colored with color i. If $\alpha(x_i) = true$, then all points situated on g_i are colored with color i. Since α is a solution of the 3-SAT instance, at least one of the points a_{j1}, a_{j2}, a_{j3} is colored with one of the first m colors for each clause c'_j with $1 \leq j \leq r$. Then, the remaining uncolored points in the vertical stripe j (at most two for each clause c'_j) can be colored with the colors $m + 2j - 1$ and $m + 2j$. This gives an extension of the $m + 2r$ - coloring to the entire graph G.

Conversely, suppose that there exists a $m+2r$ - coloring f of G such that $f(B_i) = i$ for $1 \leq i \leq m$ and $f(C_j) = m+2j-1$, $f(C'_j) = m+2j$ for $1 \leq j \leq r$. First, we consider the vertex set $\{A_1, \ldots, A_m\}$. The vertex A_1 can be colored only with color 1, since $\{A_1, B_i\} \in E$ for $i \geq 2$ and $\{A_1, C_j\}, \{A_1, C'_j\} \in E$ for $1 \leq j \leq r$. Then, vertex A_2 can be colored only with color 2, since $\{A_2, B_i\} \in E$ for $i \geq 3$, $\{A_1, A_2\} \in E$ and $\{A_2, C_j\}, \{A_2, C'_j\} \in E$ for $1 \leq j \leq r$. Using an inductive argument, we observe that $f(A_i)$ must be equal to i.

Next, we consider the vertex set $\{D_1, D'_1, \ldots, D_r, D'_r\}$. We notice (see Figure 8) that the set $\{D_1, D'_1, \ldots, D_r, D'_r, B_1, \ldots, B_m\}$ forms a clique in the permutation graph (since the corresponding points occur in decreasing order). Furthermore, the sets $\{D_1, D'_1, \ldots, D_i, D'_i, C_{i+1}, C'_{i+1}, \ldots, C_r, C'_r\}$ form also cliques in the permutation graph for each $1 \leq i < r$. Therefore, the vertices D_j, D'_j can be colored only with the colors $m + 2j - 1$ and $m + 2j$.

Let U_i be the set of vertices in G that are colored with color i. Then, for $1 \leq i \leq m$ the independent set U_i can cover only points in the horizontal stripe i (using $f(A_i) = f(B_i) = i$). We may assume that a maximal number of points in the horizontal stripe i are colored with color i, for each $1 \leq i \leq m$. Since the point set colored with color i must be increasing order, we can not take first some points on line f_i and then some points on line g_i. The only possibility is to take first some points on line g_i and then some points on line f_i.

The sets U_{m+2j-1} and U_{m+2j} can cover only points in the vertical stripe j for $1 \leq j \leq r$ (using $\{f(C_j), f(C'_j)\} = \{f(D_j), f(D'_j)\} = \{m + 2j - 1, m + 2j\}$). Furthermore, the independent sets U_{m+2j-1} and U_{m+2j} can cover only two of the points a_{j1}, a_{j2}, a_{j3}. Since we have $m + 1$ sections and

only m independent sets $U_1, \ldots, U_m$, there exists at least one section i_0 in which no independent set U_i $(1 \leq i \leq m)$ jumps from the lower line g_i to the upper line f_i. The section i_0 consists of the vertical stripes $(i_0 - 1)n + 1, \ldots, i_0 n$. We define the following truth assignment α for the variables $x_1, \ldots, x_m$:

$$\alpha(x_j) = \begin{cases} true & \text{if } U_j \text{ covers the points on line } g_j \text{ in section } i_0 \\ false & \text{if } U_j \text{ covers the points on line } f_j \text{ in section } i_0 \end{cases}$$

Since the independent sets $U_{m+2j'-1}$ and $U_{m+2j'}$ cover at most two of the points $a_{j'1}$, $a_{j'2}$ and $a_{j'3}$ for $1 \leq j' \leq r$, the sets $U_1, \ldots, U_m$ have to cover at least one of these points for each j with $(i_0 - 1)n + 1 \leq j \leq i_0 n$. Let a_{j,s_j} be one point covered by the sets $U_1, \ldots, U_m$, for $(i_0 - 1)n + 1 \leq j \leq i_0 n$. If a_{j,s_j} lies on line f_ℓ, then we have $y'_{j,s_j} = \bar{x}_\ell$ and $\alpha(x_\ell) = false$. If a_{j,s_j} lies on line g_ℓ, then we have $y'_{j,s_j} = x_\ell$ and $\alpha(x_\ell) = true$. In both cases, we get $\alpha(y'_{j,s_j}) = true$ and, therefore, α satisfies all clauses in I'_1. □

Theorem 13. *The OCCP problem for permutation graphs is NP-complete if there are at least three different cost values.*

Proof. Again, the theorem is proved by a reduction from 3-SAT. Let I_1 be an instance of 3-SAT containing a collection of n clauses $c_1, \ldots, c_n$ over a set of unnegated and negated variables $\{x_1, \bar{x}_1, \ldots, x_m, \bar{x}_m\}$. We construct an instance I_2 for the OCCP problem as follows: We take the same point set P as above and replace the points A_i and B_i each by $10r$ points situated on an increasing short line. Furthermore, there are m colors with costs 1, $2r$ colors with costs $10m$ and all other colors have costs $100rm$.

Then, we can prove (similar as above) the **claim**: I_1 is a yes-instance of 3-SAT if and only if the minimum total costs of coloring the permutation graph in instance I_2 do not exceed

$$(20rm + 3r) + 6r \cdot 10m \leq 83rm.$$

□

References

[1] E.M. Arkin and E.L. Silverberg. Scheduling jobs with fixed start and finish times. *Discrete Applied Mathematics* 18 (1987), 1–8.

[2] S. Arnborg, Efficient algorithms for combinatorial problems on graphs with bounded decomposability — A survey, *BIT* 25 (1985), 2 – 23.

[3] H.L. Bodlaender, Achromatic number is NP-complete for cographs and interval graphs, *Information Processing Letters*, 31 (1989), 135–138.

[4] H.L. Bodlaender, A linear time algorithm for finding tree-decompositions of small treewidth, 25.th *ACM Symposium on the Theory of Computing* (1993), 226 – 234.

[5] H.L. Bodlaender, A tourist guide through treewidth, *Acta Cybernetica*, 11 (1993), 1–23.

[6] H.L. Bodlaender and K. Jansen, Restrictions of graph partition problems, *Theoretical computer science*, 148 (1995), 93-109.

[7] H.L. Bodlaender, K. Jansen and G. Woeginger, Scheduling with incompatible jobs, *Graph Theoretic Concepts in Computer Science*, LNCS, 657 (1992), 37-49.

[8] M.S. Chang, Y.H. Chen, G.J. Chang and J.H. Yan, Algorithmic aspects of the generalized clique-transversal problem on chordal graphs, *Discrete Applied Mathematics* 66 (1996), 189–203.

[9] V. Chvatal, On certain polytopes associated with graphs, *Journal Combinatorial Theory* B-18 (1975), 138–154.

[10] D.G. Corneil, Y. Perl, and L.K. Stewart, A linear recognition algorithm for cographs, *SIAM Journal Computing* 4 (1985), 926–934.

[11] V.R. Dondeti and H. Emmons, Job scheduling with processors of two types, *Operations Research* 40 (1992) S76–S85.

[12] M. Fischetti, S. Martello and P. Toth, The Fixed Job Schedule Problem with spread time constraints, *Operations Research* 6 (1987), 849–858.

[13] M. Fischetti, S. Martello and P. Toth, The Fixed Job Schedule Problem with working time constraints, *Operations Research* 3 (1989), 395–403.

[14] M. Fischetti, S. Martello and P. Toth, Approximation algorithms for Fixed Job Schedule Problems, *Operations Research* 40 (1992), S96–S108.

[15] A. Frank, On chain and antichain families of a partially ordered set, *Journal Combinatorial Theory* (B) 29 (1980), 176–184.

[16] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completness, Freeman, San Francisco, 1979.

[17] F. Gavril, Algorithms for maximum k-colorings and k-coverings of transitive graphs, *Networks* 17 (1987), 465–470.

[18] M. Hujter and Z. Tuza: Precoloring extension. III. Classes of perfect graphs, Combinatorics, Probability and Computing 5 (1996), 35 – 56.

[19] K. Jansen and P. Scheffler, Some coloring results for tree like graphs, *Workshop on Graph Theoretic Concepts in Computer Science*, LNCS, 657 (1992), 50–59.

[20] K. Jansen, P. Scheffler and G.J. Woeginger, Maximum covering with D cliques, *Fundamentals of Computation Theory*, LNCS, 710 (1993) 319-328.

[21] T. Kloks, Treewidth, *Ph. D. Thesis*, Utrecht University, The Netherlands (1991).

[22] A.W.J. Kolen and L.G. Kroon, Licence Class Design: complexity and algorithms, *European Journal of Operational Research* 63 (1992), 432–444.

[23] A.W.J. Kolen, and L.G. Kroon, Analysis of shift class design problems, *European Journal of Operational Research* 79 (1994), 417–430.

[24] L.G. Kroon, A. Sen, H. Deng and A. Roy, The optimal cost chromatic partition problem for trees and interval graphs, to appear in: *Workshop on Graph Theoretical Concepts in Computer Science* (1996).

[25] S. Micali and V.V. Vazirani, An $O(|V|^{0.5}|E|)$ algorithm for finding maximum matchings in general graphs, **21**.st *IEEE Symposium Foundations of Computer Science*, (1980), 17–27.

[26] N. Robertson and P.D. Seymour, Graph minors. II. Algorithmic aspects of tree-width, *Journal on Algorithms* 7 (1986), 309–322.

[27] A. Sen, H. Deng and S. Guha, On a graph partition problem with an application to VLSI layout, *Information Processing Letters* 43 (1992), 87–94.

[28] K.J. Supowit, Finding a maximum planar subset of a set of nets in a channel, *IEEE Transactions on Computer Aided Design*, CAD 6, 1 (1987) 93–94.

[29] K. Wagner: Monotonic coverings of finite sets, *Journal of Information Processing and Cybernetics - EIK* 20 (1984), 633 – 639.

[30] M. Yannakakis and F. Gavril, The maximum k-colorable subgraph problem for chordal graphs, *Information Processing Letters* 24 (1987), 133–137.