Coloring a graph using split decomposition

Michaël Rao

Université de Metz, Laboratoire d'Informatique Théorique et Appliquée
57045 Metz Cedex 01, France
rao@sciences.univ-metz.fr

Abstract. We show how to use split decomposition to compute the weighted clique number and the chromatic number of a graph and we apply these results to some classes of graphs. In particular we present an $O(n^2m)$ algorithm to compute the chromatic number for all those graphs having a split decomposition in which every prime graph is an induced subgraph of either a C_k or a $\overline{C_k}$ for some $k \geq 3$.

Keywords: algorithms, graph algorithms, NP-complete problems, split decomposition.

1 Introduction

Decompositions play an important role in graph theory. The central role of decompositions in the recent proof of one of the major open conjectures in Graph Theory, the Strong Perfect Graph Conjecture of C. Berge, is an exciting example [7]. Furthermore various decompositions of graphs such as decomposition by clique separators, tree decomposition and clique decomposition are often used to design efficient graph algorithms. There are even beautiful general results stating that a variety of NP-complete graph problems can be solved in linear time for graphs of bounded treewidth and bounded clique-width, respectively [1, 9].

The typical approach to design efficient algorithms using graph decompositions works as follows. The algorithm recursively decomposes the graph into smaller graphs, until the obtained graphs cannot be decomposed further. Such graphs are called prime. Then the algorithm solves the problem on the prime graphs, and combines the solutions recursively to find eventually the solution for the original graph. In order to obtain an efficient algorithm by this approach, the input graphs have to be restricted to a graph class nicely decomposable with respect to the decomposition.

Several decompositions have been studied in this direction. Tarjan has given some NP-complete problems which can be solved using decomposition by clique separators [24]. Modular decomposition for discrete structures is known and studied for a long time. A nice survey on this topic has been written by Möhring and Radermacher [21]. Recently, a lot of work has been done studying modular decomposition on graphs. This includes linear time modular decomposition algorithms [10, 20], the study of modular decomposition for certain graphs classes [6], and efficient algorithms for some NP-complete graph problems using modular decomposition [4, 3, 5]. Many NP-complete problems can be solved by polynomial time (or even linear time) algorithms using tree decomposition or clique decomposition, if the treewidth or the clique-width of the graph is bounded (and the graph is given with a clique decomposition in the case of the clique-width) [1, 9, 13, 18].

In this paper, we consider the split decomposition (also called join decomposition) which can be seen as an extension of the modular decomposition. There are only few papers presenting algorithms for NP-complete problems using split decomposition. Cunningham has given an algorithm for the independent set problem [11]. Cicerone and Di Stefano showed how to apply this algorithm to parity graphs, which are exactly those graphs for which all prime graphs with respect to the split decomposition are bipartite or complete [8]. They also presented an algorithm unfortunately uncorrect for the clique problem, and applications to parity graphs [8]. Split decomposition is also used for circle graph recognition [14, 23], and parity graph recognition [8, 12].

This paper is organized as follows: Section 2 gives several preliminaries and Section 3 introduces the split decomposition. In Section 4 we discuss known and new results in the approach for obtaining efficients algorithms for NP-complete problems using the split decomposition. In Section 5 we present an algorithm for the coloring problem using split decomposition. Finally, in Section 6 we present some polynomial time coloring algorithms for nicely decomposable graph classes based on the algorithm of Section 5.

2 Preliminaries

Let $G = (V, E)$ be an undirected, simple and finite graph. We denote by $N_G(v)$ =$\{u \in V : \{u, v\} \in E\}$ the neighborhood of v in G and by $N_G[v] = N_G(v) \cup \{v\}$ its closed neighborhood. We shall write $N(v)$ and $N[v]$ if there is no ambiguity. Let $V' \subseteq V$. We denote by $G[V'] = (V', \{\{u, v\} \in E : u, v \in V'\})$ the subgraph of G induced by V'. We denote by $G - V' = G[V \setminus V']$ the subgraph induced by $V \setminus V'$ and, if $v \in V$, we write $G - v$ instead of $G - \{v\}$. A *module* is a set $M \subseteq V$ such that for all $v \in V \setminus M$ either $N(v) \cap M = \emptyset$ or $M \subseteq N(v)$.

Let $w : V \to \mathbb{N}$ be a weight function. The weight of a subset $V' \subseteq V$ is $w(V') = \sum_{v \in V'} w(v)$. A *stable set* of G is a subset S of V such that for all $u, v \in S$, $\{u, v\} \notin E$. The *weighted stability number*, denoted by $\alpha_w(G)$, of a weighted graph (G, w) is the maximum weight of a stable set of G. A *clique* of G is a subset C of V such that for all $u, v \in C$, $\{u, v\} \in E$. The *weighted clique number*, denoted by $\omega_w(G)$, of a weighted graph (G, w) is the maximum weight of a clique of G.

The *chromatic number* of a graph $G = (V, E)$, denoted by $\chi(G)$, is the smallest integer k such that there is a function $f : V \to \{1, 2, \dots, k\}$ with for all $u, v \in V$, $\{u, v\} \in E$ implies $f(u) \neq f(v)$. Clearly a *coloring* C of G can be seen as a partition of V into stable sets.

A *multiset* may contain multiple instances of the same element. We denote it by $\langle e_1, \dots e_k \rangle$. A *weighted coloring* of the weighted graph (G, w) (or a w-*weighted coloring* of G) is a multiset C of stable sets of G such that for all $v \in V$, $|\langle S \in C : v \in S \rangle| \geq w(v)$. The *weighted chromatic number* of a weighted graph (G, w), denoted by $\chi_w(G)$ is the minimum cardinality of a w-weighted coloring of G. It is well known that $\chi_w(G) \geq \omega_w(G)$.

Throughout the paper we use the following notation in order to simplify the presentation. Let $w : V \to \mathbb{N}$ be a function and let $a \in \mathbb{N}$. We denote $w|_{v\to a}$ the function of domain $V \cup \{v\}$ such that $w|_{v\to a}(v) = a$ and $w|_{v\to a}(u) = w(u)$ for all $u \in V \setminus \{v\}$.

3 Reviewing split decomposition

A *split* of a graph $G = (V, E)$ is a partition of V into two sets V_1 and V_2 such that $|V_1| \geq 2$, $|V_2| \geq 2$, and every vertex in V_1 with a neighbor in V_2 has the same neighborhood in V_2. Following this definition, we can define the *simple decomposition* of $G = (V, E)$ by the split V_1,V_2. G is decomposed into G_1 and G_2, where, for $i \in \{1, 2\}$, G_i is the subgraph of G induced by V_i with an additional vertex v, called a *marker*, such that the neighborhood of v in G_i is the set of those vertices in V_i which are adjacent in G to a vertex outside of V_i. A graph is *prime* if it does not have a split, and we say that G is decomposable into G_1 and G_2 if there is a split V_1,V_2 such that G is decomposable into G_1 and G_2 with the split V_1,V_2.

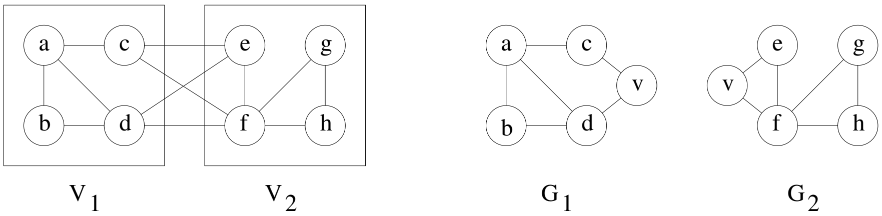

Fig. 1. A graph with a split V_1, V_2, and the two graphs G_1 and G_2 obtained by the simple decomposition.

Additionally, we define a related composition. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two graphs, such that $V_1 \cap V_2 = \{v\}$. Then $G_1 * G_2$ is the graph with vertex set $(V_1 \cup V_2) \setminus \{v\}$, and edge set $\{\{x, y\} \in E_1 : x \neq v \text{ and } y \neq v\} \cup \{\{x, y\} \in E_2 : x \neq v \text{ and } y \neq v\} \cup \{\{x, y\} : x \in N_{G_1}(v) \text{ and } y \in N_{G_2}(v)\}$. Obviously, if G is decomposable into G_1 and G_2, then $G = G_1 * G_2$. We write $G_1 * \ldots * G_k$ instead of $((G_1 * G_2) \cdots) * G_k$.

The *split decomposition* of a graph is the recursive decomposition of the graph using simple decomposition until none of the obtained graphs can be decomposed further. The *split decomposition tree* of the graph G is the tree T in which each node h corresponds to a prime graph denoted by G^*_h obtained by the split decomposition. Furthermore two nodes h and h' of T are adjacent iff the corresponding graphs G^*_h and $G^*_{h'}$ have a common marker (see figure 2).

Remark 1. The split decomposition of a graph is not necessarily unique. Cunningham [11] showed that every graph has a unique decomposition by splits into prime graphs, stars and complete graphs, with a minimum number of non decomposed graphs. This decomposition, which we call a Cunningham decomposition, is not necessarily a

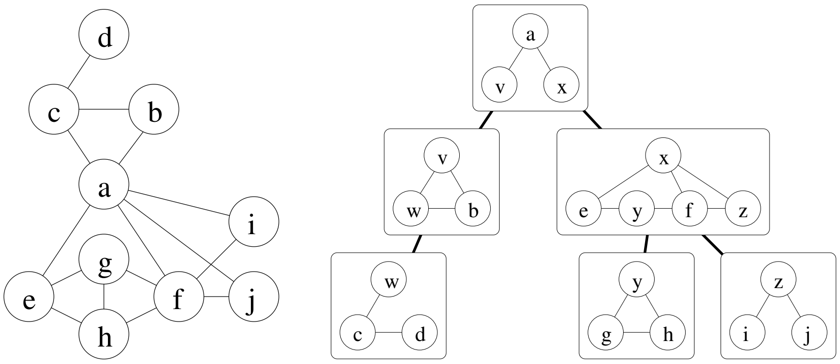

Fig. 2. A graph and its split decomposition tree. The markers are v, w, x, y and z.

split decomposition, since stars and complete graphs can be decomposable. Notice that it is easy to obtain a split decomposition from a Cunningham decomposition. Furthermore a split in a graph is either a split in the Cunningham decomposition or a split in a star or a complete graph in the Cunningham decomposition [11]. Thus each split decomposition can be obtained from the Cunningham decomposition, and the set of all prime graphs is the same for every split decomposition, up to isomorphism.

A simple induction shows that a split decomposition tree of a graph G with n vertices and m edges has at most $n-2$ nodes. The sum of the number of vertices of all prime graphs is at most $3n-4$ since each vertex in a prime graph is either a vertex of G, or a marker. The sum of the number of edges of all prime graphs is at most $m+n-3$, since each simple decomposition adds at most one edge to the overall number of edges.

All known algorithms to compute a split decomposition or a split decomposition tree compute in fact a Cunningham decomposition. The first algorithm was given by Cunningham and has running time $O(n^3)$ [11]. This has been improved to $O(nm)$ in [14], and to $O(n^2)$ in [19]. Finally Dahlhaus has given a linear time algorithm in [12].

4 Split decomposition, graph classes and the clique problem

Some graphs classes are nicely decomposable by split decompositions. Distance hereditary graphs are completely decomposable by split decomposition (i.e. all primes graphs have at most 3 vertices) [16]. A graph is a circle graph if and only if every prime graph in its split decomposition is a circle graph [14]. A graph is a parity graph if and only if every prime graph in its split decomposition is bipartite or complete [8]. The best known recognition algorithms for circle graphs [23] and parity graphs [8, 12] are based on split decomposition and these forementioned properties.

We already know that a graph is perfect iff every prime graph in the split decomposition is perfect [2]. A simple decomposition cannot destroy a C_k (induced cycle on k vertices) or its complement $\overline{C_k}$ for some $k \geq 5$. Thus if every prime graph in a split decomposition of G is weakly chordal then G is weakly chordal. (Notice that C_4 and $\overline{C_4}$ are decomposable by a simple decomposition.)

Cunningham has given an algorithm to compute the weighted stability number using split decomposition [11]. In [8] the authors apply this algorithm to parity graphs in the split decomposition of G. This algorithm has running time $O(n^{2.5})$. They also present algorithms to compute the clique number and the weighted clique number of a parity graph using split decomposition. Unfortunately theses algorithms are flawed. Figure 3 gives a counter-example for the clique number algorithm (and also a counter-example for the weighted clique number when taking all vertex weights to be 1).

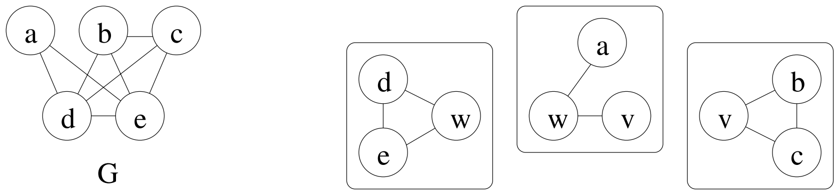

Fig. 3. A parity graph for which the algorithm for the clique number in [8] fails, and its unique split decomposition, which is also its Cunningham decomposition. The markers are v and w. $\omega(G) = 4$ and the output of the algorithm is 3.

Lemma 1 recalls Cunningham's algorithm. Lemma 2 provides a correct algorithm to compute the weighted clique number using split decomposition.

Let $G = (V, E)$ be a graph such that $G = G_1 * G_2$, $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ and $V_1 \cap V_2 = \{v\}$. Let $w : V \to \mathbb{N}$ be a weight function.

Lemma 1 ([11]). *Let $a = \alpha_w(G_2 - N_{G_2}[v])$ and $a' = \alpha_w(G_2 - v)$. Then*

$$\alpha_w(G) = \alpha_{w|_{v \to a' - a}}(G_1) + a.$$

Lemma 2. *Let $a = \omega_w(G_2[N_{G_2}(v)])$. Then*

$$\omega_w(G) = \max(\omega_w(G_2 - v), \omega_{w|_{v \to a}}(G_1)).$$

Proof. Obviously, $\omega_w(G) \geq \omega_w(G_2 - v)$. Furthermore $\omega_w(G) \geq \omega_w(G[(V_1 \setminus \{v\}) \cup N_{G_2}(v)])$. Now $\omega_w(G[(V_1 \setminus \{v\}) \cup N_{G_2}(v)]) = \omega_{w|_{v \to a}}(G_1)$, since $N_{G_2}(v)$ is a module in $G[(V_1 \setminus \{v\}) \cup N_{G_2}(v)]$. Consequently $\omega_w(G) \geq \max(\omega_w(G_2 - v), \omega_{w|_{v \to a}}(G_1))$.

Furthermore $\omega_w(G) = \max(\omega_w(G_2 - v), \omega_{w|_{v \to a}}(G_1))$ since if C is a clique of G then either $C \subseteq V_2 \setminus \{v\}$ and $w(C) \leq \omega_w(G_2 - v)$ or $C \subseteq (V_1 \setminus \{v\}) \cup N_{G_2}(v)$ and $w(C) \leq \omega_{w|_{v \to a}}(G_1)$. $\square$

Lemma 1 and Lemma 2 respectively can be used to calculate $\alpha_w(G)$ and $\omega_w(G)$ respectively of a graph G using a split decomposition tree T of G (see [8]). We know that we can use the split decomposition to compute a optimal coloring of a perfect graph [2]. But in the general case, the problem seems to be more complex than the problems in this section, since if $G = G_1 * G_2$, $\chi(G)$ cannot easily be obtained from solutions of subproblems for G_1 and G_2.

5 Coloring a graph using a split decomposition tree

In this section we present an algorithm to compute the chromatic number of a graph $G = (V, E)$ using a split decomposition tree of G. Typically our algorithm has to compute the weighted chromatic number for a variety of induced subgraphs of G corresponding to nodes of the split decomposition tree.

Let T be a split decomposition tree of the graph G computed using the linear time algorithm of Dahlhaus [12]. Our coloring algorithm runs recursively on the split decomposition tree T for which we choose any vertex to be its root r.

We recall that for a node h of T, G^*_h is the prime graph of the split decomposition corresponding to the node h. For each node $h \neq r$ of T and its parent h' in T, let v_h be the unique marker belonging to G^*_h and $G^*_{h'}$. We define for every leaf h of T, $G_h = G^*_h$. We define recursively for each internal node h of T with children $h_1, h_2, \ldots, h_k$, $G_h = G^*_h * G_{h_1} * \ldots * G_{h_k}$. We call G_h the graph corresponding to the subtree of T rooted at h. Notice that $G_r = G$.

We present our algorithm in Figure 4. Its correctness proof is based on Lemma 3 and 4. The following notation is used throughout the proofs of these Lemmas. Let C be a w-weighted coloring of $G = (V, E)$ and $V' \subseteq V$. We denote by $C_{V'}$ the collection of all stables sets of the coloring C that contain at least one vertex of V', i.e. $C_{V'} = \langle S \in C : S \cap V' \neq \emptyset \rangle$. We denote by $\mathcal{D}(G, w, V')$ the set of all pairs $(a, b) \in \mathbb{N} \times \mathbb{N}$ such that there is a w-weighted coloring C of G with $a + b$ colors and $|C_{V'}| = a$. Obviously, if $(a, b) \in \mathcal{D}(G, w, V')$, then $(a', b) \in \mathcal{D}(G, w, V')$ for all $a' > a$ and $(a, b') \in \mathcal{D}(G, w, V')$ for all $b' > b$. We call a pair $(a, b) \in \mathcal{D}(G, w, V')$ minimal if $(a - 1, b) \notin \mathcal{D}(G, w, V')$ and $(a, b - 1) \notin \mathcal{D}(G, w, V')$. Note that for every minimal pair (a, b) of $\mathcal{D}(G, w, V')$, both $a \leq \chi_w(G)$ and $b \leq \chi_w(G)$.

From now on, for every node $h \neq r$, we call $\mathcal{D}(h) = \mathcal{D}(G_h - v_h, w, N_{G_h}(v_h))$ the $\mathcal{D}$-set of h, where for all $v \in V_h \setminus \{v_h\}$, $w(v) = 1$. Our algorithm uses a dynamic programming and computes in a bottom-up fashion for all nodes $h \neq r$ of T, the $\mathcal{D}$-set of h from the $\mathcal{D}$-sets of all its children. Finally it computes $\chi(G)$ from the $\mathcal{D}$-sets of the children of the root r of T.

The following lemma provides the operation to be executed for each node of the split decomposition tree when computing its $\mathcal{D}$-set and its chromatic number using the $\mathcal{D}$-sets of its children. Let h be a node of T. Let $G^*_h = (V^*_h, E^*_h)$ be the prime graph corresponding to h, and let $G_h = (V_h, E_h)$ be the graph corresponding to the subtree of T rooted at h. Let $h_1, h_2, \ldots, h_k$ be the children of h in T. Then G_{h_i} is the graph corresponding to the subtree in T rooted at h_i and v_{h_i} is the unique marker belonging to G^*_h and G_{h_i}. For simplifying the notations in Lemma 3 and its proof, we note $G_i = G_{h_i} = (V_i, E_i)$, $v_i = v_{h_i}$ and $G^* = G^*_h = (V^*, E^*)$. As discussed above, $G_h = G^* * G_1 * \ldots * G_k$. Let $w : V_h \to \mathbb{N}$ be a weight function.

Lemma 3. *For all $i \in \{1, \ldots k\}$, let $(a_i, b_i) \in \mathcal{D}(G_i - v_i, w, N_{G_i}(v_i))$. Let $\mathfrak{C}$ be the set of all w-weighted colorings C of G_h with $|C_{V_i \setminus \{v_i\}}| = a_i + b_i$ and $|C_{N_{G_i}(v_i)}| = a_i$. Then*

$$\min_{C \in \mathfrak{C}} |C| = \max(\chi_{w^*}(G^*), \max_{i \in \{1,2,\ldots,k\}} (a_i + b_i))$$

where $w^* : V^* \to \mathbb{N}$ such that for all $i \in \{1, 2, \ldots, k\}, w^*(v_i) = a_i$, and for all $v \in V^* \setminus \{v_1, v_2, \ldots, v_k\}$, $w^*(v) = w(v)$.

Function CHROMATICNUMBER(T)
Input: split decomposition tree T of G, Output: chromatic number of G
begin
  for each node h of T, in bottom-up fashion, where k is the number of children of h
      and $h_1, h_2, \ldots, h_k$ are its children
    if $h \neq r$ **then**
      $D_h \leftarrow \emptyset$
      for b **from** 0 **to** n
        $D_h \leftarrow D_h \cup \{(\text{CHROMSPLIT}(h, b, k, D_{h_1}, D_{h_2}, \ldots, D_{h_k}) - b, b)\}$
      endfor
    else return CHROMSPLIT($r, 0, k, D_{h_1}, D_{h_2}, \ldots, D_{h_k}$)
    endif
  endfor
end {ChromaticNumber}

Function CHROMSPLIT($h, b, k, D_{h_1}, D_{h_2}, \ldots, D_{h_k}$)
begin
  if $k = 0$ **then**
    let $w^* : V_h^* \to \mathbb{N}$ such that for all $v \in V_h^* \setminus \{v_h\}$, $w^*(v) = 1$ and, if $h \neq r$, $w(v_h) = b$
    return $\chi_{w^*}(G_h^*)$
  else
    let $m \leftarrow +\infty$
    for c **from** 0 **to** $2n$ **do**
      for i **from** 1 **to** k **do**
        let $a_i \leftarrow \min\{a' : \exists b'$ such that $(a', b') \in D_{h_i}$ and $a' + b' \leq c\}$
      endfor
      let $w^* : V_h^* \to \mathbb{N}$ such that for all $i \in \{1, 2, \ldots, k\}$, $w^*(v_{h_i}) = a_i$,
        for all $v \in V_h^* \setminus \{v_h, v_{h_1}, \ldots v_{h_k}\}$, $w^*(v) = 1$ and, if $h \neq r$, $w(v_h) = b$
      $m \leftarrow \min(m, \max(c, \chi_{w^*}(G_h^*)))$
    endfor
    return m
  endif
end {ChromSplit}

Fig. 4. Algorithm to compute the chromatic number.

Proof. Recall that if C is a weighted coloring of G and $V' \subseteq V$ then $C_{V'}$ is defined to be the multiset of all stable sets belonging to C having a non-empty intersection with $V' \subseteq V$. For all $i \in \{1, 2, \ldots, k\}$, let $A_i = N_{G_i}(v_i)$ and let $B_i = V_i \setminus N_{G_i}[v_i]$.

Firstly, we prove that every coloring $C \in \mathfrak{C}$ contains at least $\max(\chi_{w^*}(G^*), \max_{i \in \{1,2,\ldots,k\}}(a_i + b_i))$ stable sets. For all $i \in \{1, \ldots k\}$, $a_i + b_i = |C_{V_i \setminus \{v_i\}}| \leq |C|$. It remains to show that $\chi_{w^*}(G^*) \leq |C|$. To do so we construct a w^*-weighted coloring C^* of G^* from $C_{V_h \setminus \bigcup_{i=1}^k B_i}$. For each $S \in C_{V_h \setminus \bigcup_{i=1}^k B_i}$, let $S^* \subseteq V^*$ be the set satisfying $S^* \cap (V^* \setminus \{v_1, \ldots v_k\}) = S \cap (V^* \setminus \{v_1, \ldots v_k\})$, and for all $i \in \{1, 2, \ldots, k\}$, $v_i \in S^*$ iff $S \cap A_i \neq \emptyset$. Notice that S^* is a stable set of G^*. Let C^* be the multiset of all sets S^* Clearly $|C^*| \leq |C|$. Furthermore C^* is a w^*-weighted coloring of G^* since for all i, $|C^*_{\{v_i\}}| = |C_{A_i}| = a_i$. Hence $\min_{C \in \mathfrak{C}} |C| \geq \max(\chi_{w^*}(G^*), \max_{i \in \{1,2,\ldots,k\}}(a_i + b_i))$.

It remains to prove equality. To do so we construct a w-weighted coloring C of G_h such that $C \in \mathfrak{C}$ and $|C| = \max(\chi_{w^*}(G^*), \max_{i\in\{1,2,\ldots,k\}}(a_i + b_i))$. Let C^* be a minimum w^*-weighted coloring of G^*, and for all i, let C^i be a coloring of $G_i - v_i$ such that $|C^i| = a_i + b_i$ and $|C^i_{A_i}| = a_i$.

We construct C as follows. In the first stage, we pick a stable set $S^* \in C^*$ and for all $i \in \{1, 2, \ldots, k\}$, if $v_i \in S^*$ we pick a stable set S^i in $C^i_{A_i}$, otherwise we pick S^i in $C^i \setminus C^i_{A_i}$. If there is no remaining stable set, we take $S^i = \emptyset$. We add to C the set $S = S^* \setminus \{v_1, \ldots v_k\} \cup \bigcup_{i=1}^k S^i$. Notice that S is a stable set. We repeat this operation until C^* is empty. Since $w^*(v_i) = a_i = |C^i_{A_i}|$, there is no remaining set in $S^i \in C^i$ such that $S_i \cap A_i \neq \emptyset$ at the end of the first stage.

In the second stage, as long as there is an $i \in \{1, 2, \ldots, k\}$ such that $C^i \neq \emptyset$, we pick for all $i \in \{1, 2, \ldots, k\}$ a stable set $S^i \in C^i$ if $C^i \neq \emptyset$, otherwise we take $S^i = \emptyset$. We add to C the set $S = \bigcup_{i=1}^k S^i$. Notice that S is a stable set. It is not hard to see that C is a w-weighted coloring of G_h, and $C \in \mathfrak{C}$.

At the end of the first stage, C has $\chi_{w^*}(G^*)$ stable sets, and for each $i \in \{1, 2, \ldots, k\}$, C^i has $\max(0, |C^i \setminus C^i_{A_i}| - |C^* \setminus C^*_{v_i}|) = \max(0, b_i - \chi_{w^*}(G^*) + a_i)$ remaining stable sets. Then we add in the second stage $\max(0, -\chi_{w^*}(G^*) + \max_{i\in\{1,2,\ldots,k\}}(a_i + b_i))$ stable sets to C. Thus, at the end $|C| = \max(\chi_{w^*}(G^*), \max_{i\in\{1,2,\ldots,k\}}(a_i + b_i))$. □

Then $\chi_w(G_h)$ is the minimum of $\max(\chi_{w^*}(G^*_h), \max_{i\in\{1,2,\ldots,k\}}(a_i + b_i))$ over all possible choice of the k-tuple $((a_1, b_1), (a_2, b_2), \ldots, (a_k, b_k))$ into $\mathcal{D}(h_1) \times \mathcal{D}(h_2) \times \ldots \times \mathcal{D}(h_k)$. For all $c \in \mathbb{N}$, let t_c be the k-tuple such that each for all $i \in \{1, 2, \ldots k\}$, (a_i, b_i) is a pair of $\{(a, b) \in D(h_i) : a + b \leq c\}$ with the smallest a. Obviously, if t is a k-tuple $((a_1, b_1), (a_2, b_2), \ldots, (a_k, b_k))$ and $c = \max_i(a_i + b_i)$, then the choice t_c at least as good as t. The algorithm computes $\chi_w(G_h)$ by taking the minimum over all k-tuple t_c, $c \in \{0, 1, \ldots, 2n\}$.

The previous lemma has shown how to compute the weighted chromatic number of G_h from the $\mathcal{D}$-set of its children. If h is a node different from r, then we have to compute the $\mathcal{D}$-set of h. The following lemma shows how this can be done.

Lemma 4. *Let $G = (V, E)$ be a graph, $v \in V$ and $w : V \setminus \{v\} \to \mathbb{N}$ a weight function. Let $b \in \mathbb{N}$ and $w' : V \to \mathbb{N}$ such that $w'(v) = w(v)$ for all $v \in V \setminus \{v\}$, and $w'(v) = b$. Then $(\chi_{w'}(G) - b, b) \in \mathcal{D}(G - v, w, N(v))$, and $(\chi_{w'}(G) - b - 1, b) \notin \mathcal{D}(G - v, w, N(v))$.*

Proof. Let C be a minimum w'-weighted coloring of G. Then $C' = \langle S \setminus \{v\} : S \in C\rangle$ is a w-weighted coloring of G, and has at least b stable sets not having a vertex in common with $N(v)$. Thus $(\chi_{w'}(G) - b, b) \in \mathcal{D}(G - v, w, N(v))$.

If there would be a a w-weighted coloring $\widehat{C}$ of $G - v$ such that $|\widehat{C}| < \chi_{w'}(G)$ having at least b stable sets having no vertex in common with $N(v)$, then the multiset C obtained by adding v to all stable sets having no vertex in common with $N(v)$, would be a w'-weighted coloring of G, contradiction. □

Now the minimals pairs of the $\mathcal{D}$-set of h can be computed using $\chi_{w|_{v_h \to b}}(G_h)$, $b = 0, 1, \ldots, n$. Now we are ready to summarize the correctness proof.

Theorem 1. *The algorithm* ChromaticNumber *takes as input a graph G and it split decomposition tree T and computes the chromatic number of G.*

Proof. The function CHROMSPLIT$(h, b, \ldots)$ returns $\chi_w(G_h)$, with $w(v) = 1$ for all v in $V_h \setminus \{v_h\}$ and, if $h \neq r$, $w(v_h) = b$, using Lemma 3 and the remark following the proof of Lemma 3. For all $h \neq r$ the main loop computes D_h which contains all the minimal pairs of the $\mathcal{D}$-set of h, using the Lemma 4. Finally, the algorithm computes the chromatic number of $G_r = G$. □

Remark 2. Our algorithm computes the chromatic number of the input graph. We mention that it is not hard to modify the algorithm such that it computes a minimum coloring if we use as sub-function an algorithm which computes a minimal weighted coloring of a prime graph.

Theorem 2. *If the algorithm for weighted chromatic number for the prime graphs in the input split decomposition tree has running time $f(n, m)$ then the total running time of the algorithm* CHROMATICNUMBER *is $O(n^3 \cdot f(n, m))$.*

Proof. This algorithm executes $O(n^2)$ times the algorithm for the weighted chromatic number for each prime graph in the split decomposition, and the split decomposition tree has $O(n)$ nodes. The remainder can be done in time $O(n^3)$ by pre-calculating $\min\{a' : \exists b'$ such that $(a', b') \in D_h$ and $a' + b' \leq c\}$ for all node h of T and for all $c \in \{0, 1, \ldots, 2n\}$. Consequently the overall running time is $O(n^3 \cdot f(n, m))$. □

Remark 3. This running time can be improved to $O(n^2 \cdot f(n, m))$ for functions f such that the running time to execute the algorithm for weighted chromatic number on each prime graph is $O(f(n, m))$.

6 Polynomial time algorithms

6.1 Some classes of perfect graphs

In order to compute the chromatic number (respectively a minimum coloring) using the split decomposition, we have to compute the weighted chromatic number (respectively a minimum weighted coloring) of the prime graphs. Hoàng has given an $O(nm)$ algorithm for minimum weighted coloring for perfectly orderable graphs if a perfect order is given with the graph, and $O(n^2)$ algorithms for comparability graphs and for chordal graphs [17]. Grötschel, Lovász and Schrijver have given a weakly polynomial algorithm for this problem for perfect graphs (i.e. the algorithm is polynomial on the sum of the vertex weights) [15], which is applicable in our algorithm since the sum of the weights is bounded by n.

Using Hoàng's $O(n^2)$ algorithm for the minimum weighted coloring for the comparability graphs and the algorithm in Section 5, we obtain an $O(n^4)$ algorithm for the weighted coloring on the class of graphs for which every prime graph in the split decomposition is a comparability graph. Raschle and Simon have given $O(m^2)$ algorithms for orienting P_4-comparability graphs and P_4-indifference graphs, which are subclasses of perfectly orderable graphs [22]. So using these algorithms, Hoàng's $O(nm)$ algorithm for the minimum weighted coloring for perfectly orderable graphs, and the algorithm in Section 5, we obtain an $O(n^2m^2)$ algorithm for the weighted coloring on the class of graphs for which every prime graph in the split decomposition is a P_4-comparability graph or a P_4-indifference graph.

6.2 Every prime graph has bounded size

Let $\mathcal{G}_k$, $k \geq 3$ be the class of graph for which every prime graph in a split decomposition has at most k vertices. Notice that the $\mathcal{G}_3$ is the class of all distance hereditary graphs. For graphs of bounded size the weighted stability number and the weighted clique number can be computed in constant time. The weighted chromatic number of graphs of bounded size can be computed in constant time [3].

Corollary 1. *For any fixed $k \geq 3$, there is an $O(n)$ algorithm to compute the weighted stability number and the weighted clique number, respectively, and an $O(n^3)$ algorithm to compute the chromatic number of graph in the class $\mathcal{G}_k$, if a split decomposition tree is given with the graph.*

6.3 Every prime graph is an induced subgraph of C_k or $\overline{C_k}$ for some $k \geq 3$

Let $\mathcal{G}_c$ be the class of all graphs for which every prime graph in the split decomposition is an induced subgraph of C_k or $\overline{C_k}$, $k \geq 3$. For all these graphs, computing the weighted stability number and the weighted clique number is easy and can be done in linear time. Thus using results of Section 4, we obtain

Corollary 2. *There are $O(n+m)$ algorithms to compute the weighted stability number and the weighted clique number, respectively, on the graph class $\mathcal{G}_c$.*

A more challenging problem is to compute the chromatic number for $\mathcal{G}_c$. The following lemma generalizes a theorem of Vanherpe for the graph C_5 [25].

Lemma 5. *Let $G = (V, E)$ with $V = \{v_1, v_2, \ldots, v_n\}$ be a cycle of length $n \geq 3$, and let $w : V \to \mathbb{N}$ be a weight function of G. Then*

$$\chi_w(G) = \max\left(\omega_w(G), \left\lceil \frac{\sum_{i=1}^n w(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil\right).$$

Proof. We already mentioned that $\chi_w(G) \geq \omega_w(G)$. Furthermore each color class has at most $\lfloor \frac{n}{2} \rfloor$ vertices, thus $\chi_w(G) \geq \left\lceil \frac{\sum_{i=1}^n w(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil$. If n is even or $n = 3$, then G is perfect. Thus $\chi_w(G) = \omega_w(G)$ [15].

Now we assume that n is odd and $n \neq 3$. We show the claimed equality by induction on the sum of the weights of G. If there is a $v \in V$ such that $w(v) = 0$, then $\chi_w(G) = \chi_w(G - v) = \omega_w(G)$, since $G - v$ is perfect. Otherwise, let E' be the set of all edges of G for which the sum of the weights of its two endpoints is $\omega_w(G)$.

Case 1: $E' = E$. Then all the weights are equals (since n is odd), so $\left\lceil \frac{\sum_{i=1}^n w(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil > \omega_w(G)$. Let S be a stable set of $\lfloor \frac{n}{2} \rfloor$ vertices, and w' be the weight function defined by $w'(v) = w(v) - 1$ for all $v \in S$ and $w'(v) = w(v)$ for all $v \in V \setminus S$. Then

$$\begin{aligned}\chi_w(G) &\leq 1 + \chi_{w'}(G) = 1 + \max\left(\omega_{w'}(G), \left\lceil \frac{\sum_{i=1}^n w'(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil\right) \\ &\leq \left\lceil \frac{\sum_{i=1}^n w(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil\end{aligned}$$

since $\left\lceil \frac{\sum_{i=1}^{n} w(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil > \omega_w(G) = \omega_{w'}(G)$ and $\left\lceil \frac{\sum_{i=1}^{n} w'(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil = \left\lceil \frac{\sum_{i=1}^{n} w(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil - 1$.

Case 2: $E' \neq E$. Then there is a stable set S of $\lfloor \frac{n}{2} \rfloor$ vertices containing an endpoint for each edge of E'. Let w' be the weight function defined by $w'(v) = w(v) - 1$ for all $v \in S$ and $w'(v) = w(v)$ for all $v \in V \setminus S$. Then

$$\begin{aligned} \chi_w(G) &\leq 1 + \chi_{w'}(G) = 1 + \max\left(\omega_{w'}(G), \left\lceil \frac{\sum_{i=1}^{n} w'(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil\right) \\ &\leq \max\left(\omega_w(G), \left\lceil \frac{\sum_{i=1}^{n} w(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil\right) \end{aligned}$$

since $\left\lceil \frac{\sum_{i=1}^{n} w'(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil = \left\lceil \frac{\sum_{i=1}^{n} w(v_i)}{\lfloor \frac{n}{2} \rfloor} \right\rceil - 1$ and $\omega_{w'}(G) = \omega_w(G) - 1$. $\square$

The *weighted clique partition number* of a weighted graph (G, w) is $\kappa_w(G) = \chi_w(\overline{G})$. The following lemma gives the weighted clique partition number of a graph C_n for all $n \geq 3$. Thus it gives the weighted chromatic number of a graph $\overline{C_n}$ for all $n \geq 3$. The proof is similar to the proof of Lemma 5 and omitted.

Lemma 6. *Let $G = (V, E)$ with $V = \{v_1, v_2, \ldots, v_n\}$ be a cycle of length $n \geq 3$, and let $w : V \to \mathbb{N}$ be a weight function of G. Then*

$$\kappa_w(G) = \max\left(\alpha_w(G), \left\lceil \frac{\sum_{i=1}^{n} w(v_i)}{2} \right\rceil\right).$$

By Lemma 5 and 6 the weighted chromatic number of a graph C_n or $\overline{C_n}$, $n \geq 3$ can be computed in linear time. Using Theorem 2 and it's the remark we obtain

Corollary 3. *There is an $O(n^2 m)$ algorithm to compute the chromatic number for all graphs in the class $\mathcal{G}_c$.*

Acknowledgement

I am grateful to my supervisor D. Kratsch and anonymous referees for helpful comments.

References

1. S. Arnborg, J. Lagergren, D. Seese, Easy problems for tree-decomposable graphs, *Journal of Algorithms* 12 (1991) 308-340
2. R. Bixby, A composition for perfect graphs, *Annals of Discrete Math.* 21 (1984) 221-224
3. H. L. Bodlaender, A. Brandstädt, D. Kratsch, M. Rao, J. Spinrad, Linear time algorithms for some NP-complete problems on (P_5,gem)-free graphs, *Proceedings of FCT'2003*, LNCS 2751, 61-72, Springer-Verlag, Berlin, 2003
4. H.L. Bodlaender, K. Jansen, On the complexity of the maximum cut problem, *Nord. J. Comput.* 7 (2000) 14-31

5. H. L. BODLAENDER, U. ROTICS, Computing the treewidth and the minimum fill-in with the modular decomposition, *Algorithmica* 36 (2003) 375-408
6. A. BRANDSTÄDT, D. KRATSCH, On the structure of (P_5,gem)-free graphs, *Manuscript* 2002 `http://www.informatik.uni-rostock.de/(en)/~ab/ps-files/p5gemdam.ps`
7. M. CHUDNOVSKY, N. ROBERTSON, P.D.SEYMOUR, R.THOMAS, The strong perfect graph theorem, *Manuscript* 2002 `http://www.math.gatech.edu/~thomas/spgc.ps.gz`
8. S. CICERONE, D. DI STEFANO, On the extension of bipartite graphs to parity graphs, *Discrete Applied Math.* 95 (1999) 181-195
9. B. COURCELLE, J. A. MAKOWSKY, U. ROTICS, Linear time solvable optimization problems on graphs of bounded clique-width, *Theory of Computing Systems* 33 (2000) 125-150
10. A. COURNIER, M. HABIB, A new linear algorithm for modular decomposition, *Trees in Algebra and Programming - CAAP '94*, LNCS 787 (1994) 68-84
11. W. CUNNINGHAM, Decomposition of directed graphs, *SIAM Journal on Algebraic and Discrete Methods* 3 (1982) 214-228
12. E. DAHLHAUS, Parallel algorithms for hierarchical clustering and applications to split decomposition and parity graph recognition, *Journal of Algorithms* 36 (2000) 205-240
13. W. ESPELAGE, F. GURSKI, E. WANKE, How to solve NP-hard graph problems on clique-width bounded graphs in polynomial time, *Proceedings of WG 2001*, LNCS 2204, 117-128, Springer-Verlag, Berlin, 2001
14. C. P. GABOR, W. L. HSU, K. J. SUPOWIT, Recognizing circle graphs in polynomial time, *Journal of the ACM* 36 (1989) 435-473
15. M. GRÖTSCHEL, L. LOVÁSZ, A. SCHRIJVER, Polynomial algorithms for perfect graphs, *Annals of Discrete Math.* 21 (1984) 325-356
16. P. HAMMER, F. MAFFRAY, Completely separable graphs, *Discrete Applied Math.* 27 (1990) 85-99
17. C. T. HOÀNG, Efficient algorithms for minimum weighted colouring of some classes of perfect graphs, *Discrete Applied Math.* 55 (1994) 133-143
18. D. KOBLER, U. ROTICS, Edge dominating set and colorings on graphs with fixed clique-width, *Discrete Applied Math.* 126 (2003) 197-221
19. T.-H. MA, J. SPINRAD, An $O(n^2)$ algorithm for undirected split decomposition, *Journal of Algorithms* 16 (1994) 154-160
20. R.M. MCCONNELL, J. SPINRAD, Modular decomposition and transitive orientation, *Discrete Math.* 201 (1999) 189-241
21. R.H MÖHRING, F.J. RADERMACHER, Substitution decomposition for discrete structures and connections with combinatorial optimization, *Annals of Discrete Math.* 19 (1984) 257-356
22. T. RASCHLE, K. SIMON, On the P_4-components of graphs, *Discrete Applied Math.* 100 (2000) 215-235
23. J. SPINRAD, Recognition of circle graphs, *Journal of Algorithms* 16 (1994) 264-282
24. R. E. TARJAN, Decomposition by clique separators, *Discrete Math.* 55 (1985) 221-232
25. J.-M. VANHERPE, Décomposition et algorithmes efficaces sur les graphes, *Ph.D. thesis*, Université de Picardie, LaRIA, 1999