Linear Time Algorithms for some NP-complete problems on (P_5,Gem)-Free Graphs

(Extended Abstract)

Hans Bodlaender[1], Andreas Brandstädt[2], Dieter Kratsch[3], Michaël Rao[3], and Jeremy Spinrad[4]

[1] Institute of Information and Computing Sciences, Utrecht University
P.O. Box 80.089, 3508 TB Utrecht, the Netherlands
hansb@cs.uu.nl

[2] Fachbereich Informatik, Universität Rostock
A.-Einstein-Str. 21, 18051 Rostock, Germany
ab@informatik.uni-rostock.de

[3] Université de Metz, Laboratoire d'Informatique Théorique et Appliquée
57045 Metz Cedex 01, France
(kratsch|rao)@sciences.univ-metz.fr
fax: ++ 00 33 387315309

[4] Department of Electrical Engineering and Computer Science
Vanderbilt University, Nashville TN 37235, U.S.A.
spin@vuse.vanderbilt.edu

Abstract. A graph is (P_5,gem)-free, when it does not contain P_5 (an induced path with five vertices) or a gem (a graph formed by making an universal vertex adjacent to each of the four vertices of the induced path P_4) as an induced subgraph.
Using a characterization of (P_5,gem)-free graphs by their prime graphs with respect to modular decomposition and their modular decomposition trees [6], we obtain linear time algorithms for the following NP-complete problems on (P_5,gem)-free graphs: Minimum Coloring, Maximum Weight Stable Set, Maximum Weight Clique, and Minimum Clique Cover.

Keywords: algorithms, graph algorithms, NP-complete problems, modular decomposition, (P_5,gem)-free graphs.

1 Introduction

Graph 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 so-called Strong Perfect Graph Conjecture of C. Berge, is an exciting example [9]. Furthermore various decompositions of graphs such as decomposition by clique cutsets, tree-decomposition and clique-width 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, 12].

Despite the fact that modular decomposition is a well-known decomposition in graph theory having algorithmic uses that seem to be simple and obvious, there is relatively few research concerning non-trivial uses of modular decomposition such as designing polynomial time algorithms for NP-complete problems on special graph classes. An important exception are the many linear and polynomial time algorithms for cographs [10, 11] i.e. P_4-free graphs which are known to have a cotree representation which allows to solve various NP-complete problems in linear time when restricted to cographs, among them the problems Maximum (Weight) Stable Set, Maximum (Weight) Clique, Minimum Coloring and Minimum Clique Cover [10, 11].

The original motivation to study (P_5,gem)-free graphs, as a natural generalization of cographs, by the authors of [6] had been to construct a faster, possibly linear time algorithm for the Maximum Stable Set problem on (P_5,gem)-free graphs. They established a characterization of the (P_5,gem)-free graphs by their prime induced subgraphs called the Structure Theorem for (P_5,gem)-free graphs. We show in this paper that the Structure Theorem is a powerful tool to design efficient algorithms for NP-complete problems on (P_5,gem)-free graphs. All our algorithms use the modular decomposition tree of the input graph and the structure of the prime (P_5,gem)-free graphs. We are convinced that efficient algorithms for other NP-complete graph problems (e.g. domination problems) on (P_5,gem)-free graphs can also be obtained by this approach.

It is remarkable that there are only few papers establishing efficient algorithms for NP-complete graph problems using modular decomposition and that most of them consider a single problem, namely Maximum (Weight) Stable Set. For work dealing with other problems we refer to [4, 5, 18]. Concerning the limits of modular decomposition it is known, for example, that Achromatic Number, List Coloring, and $\lambda_{2,1}$-Coloring with pre-assigned colors remain NP-complete on cographs [2, 3, 19]. This implies that these three problems are NP-complete on (P_5,gem)-free graphs.[1]

There is also a strong relation between modular decomposition and the clique-width of graphs. For example, if all prime graphs of a graph class have bounded size then this class has bounded clique-width. Problems definable in a certain logic, so-called LinEMSOL($\tau_{1,L}$)-definable problems, such as Maximum (Weight) Stable Set, Maximum (Weight) Clique and Minimum (Weight) Dominating Set, can be solved in linear time on any graph class of bounded clique-width, assuming a k-expression describing the graph is part of the input [12]. Many other NP-complete problems which are

[1] A proof, similarly to the one in [3] shows that $\lambda_{2,1}$-Coloring is NP-complete for graphs with at most one prime induced subgraph, the P_4, and hence for (P_5,gem)-free graphs.

not LinEMSOL($\tau_{1,L}$)-definable can be solved in polynomial time on graph classes of bounded clique-width [15, 20].

Brandstädt et al. have shown that (P_5,gem)-free graphs have clique-width at most five [7]. However this does not yet imply linear time algorithms for LinEMSOL($\tau_{1,L}$)-definable problems on (P_5,gem)-free graphs, since their approach does not provide a linear time algorithm to compute a suitable k-expression.

We present a linear time algorithm to solve the NP-complete Minimum Coloring problem on (P_5,gem)-free graphs using modular decomposition in Section 5. The NP-complete problems Maximum Weight Stable Set, Maximum Weight Clique and Minimum Clique Cover can also be solved by linear time algorithms using modular decomposition for (P_5,gem)-free graphs. Due to space constraints, these algorithms are not shown in this extended abstract.

2 Preliminaries

We assume the reader to be familiar with standard graph theoretic notations. In this paper, $G = (V, E)$ is a finite undirected graph, and $|V| = n$ and $|E| = m$. $N(v) := \{u : u \in V, u \neq v, uv \in E\}$ denotes the *open neighborhood* of v and $N[v] := N(v) \cup \{v\}$ the *closed neighborhood* of v. The *complement graph* of G is denoted $\overline{G} = (V, \overline{E})$. For $U \subseteq V$ let $G[U]$ denote the subgraph of G induced by U. A graph is *co-connected* if its complement $\overline{G}$ is connected. If for $U \subset V$, a vertex not in U is adjacent to exactly k vertices in U then it is called a *k-vertex* for U.

A function $f : V \to \mathbb{N}$ is a *(proper) coloring* of the graph $G = (V, E)$, if $\{u, v\} \in E$ implies $f(u) \neq f(v)$. The *chromatic number* of G, denoted $\chi(G)$, is the smallest k such that the graph G has a k-coloring $f : V \to \{1, 2, \ldots, k\}$.

Let $G = (V, E)$ be a graph with vertex weight function $w : V \to \mathbb{N}$. The weight of a vertex set $U \subseteq V$ is defined to be $w(U) := \sum_{u \in U} w(u)$. We let $\alpha_w(G)$ denote the maximum weight of a stable set of G and $\omega_w(G)$ denote the maximum weight of a clique of G. A *weighted k-coloring* of (G, w) assigns to each vertex v of G $w(v)$ different colors, i.e. integers of $\{1, 2, \ldots, k\}$, such that $\{x, y\} \in E$ implies that no color assigned to x is equal to a color assigned to y. $\chi_w(G)$ denotes the smallest k such that the graph G with weight function w has a weighted k-coloring. Note that each weighted k-coloring of (G, w) corresponds to a multiset $S_1, S_2, \ldots, S_k$ of stable sets of G where S_i, $i \in \{1, 2, \ldots, k\}$, is the set of all vertices of G to which color i is assigned.

3 Modular decomposition

Modular decomposition is a fundamental decomposition technique that can be applied to graphs, partially ordered sets, hypergraphs and other structures. It has been described and used under different names and it has been rediscovered various times. Gallai introduced and studied modular decomposition in his seminal 1967 paper [17] where it is used to decompose comparability graphs.

A vertex set $M \subseteq V$ is a *module* in G if for all vertices $x \in V \setminus M$, x is either adjacent to all vertices in M, or non-adjacent to all vertices in M. The *trivial modules* of G are $\emptyset, V$ and the singletons. A *homogeneous set* in G is a nontrivial module in G. A graph containing no homogeneous set is called *prime*. Note that the smallest prime graph is the P_4. A homogeneous set M is *maximal* if no other homogeneous set properly contains M.

Modular decomposition of graphs is based on the following decomposition theorem.

Theorem 1 ([17]). *Let $G = (V, E)$ be a graph with at least two vertices. Then exactly one of the following conditions holds:*

(i) *G is not connected: it can be decomposed into its connected components;*
(ii) *$\overline{G}$ is not connected: G can be decomposed into the connected components of $\overline{G}$;*
(iii) *G is connected and co-connected. There is some $U \subseteq V$ and a unique partition P of V such that*
  (a) *$|U| > 3$,*
  (b) *$G[U]$ is a maximal prime induced subgraph of G, and*
  (c) *for every class S of the partition P, S is a module of G and $|S \cap U| = 1$.*

Consequently there are three decomposition operations.

0-Operation: If G is disconnected then decompose it into its connected components $G_1, G_2, \ldots, G_r$.
1-Operation: If $\overline{G}$ is disconnected then decompose G into $G_1, G_2, \ldots G_s$, where $\overline{G}_1$, $\overline{G}_2, \ldots \overline{G}_s$ are the connected components of $\overline{G}$.
2-Operation: If $G = (V, E)$ is connected and co-connected then its maximal homogeneous sets are pairwise disjoint and they form the partition P of V. The graph $G[U]$ is obtained from G by contracting every maximal homogeneous set of G to a single vertex; it is called the *characteristic graph* of G and denoted by G^*. (Note that the characteristic graph of a connected and co-connected graph G is prime.)

The decomposition theorem and the above mentioned operations lead to the uniquely determined *modular decomposition tree T* of G. The leaves of the modular decomposition tree are the vertices of G. The interior nodes of T are labeled 0, 1 or 2 according to the operation corresponding to the node. Thus we call them 0-node (parallel node), 1-node (series node) and 2-node (prime node). Any interior node x of T corresponds to the subgraph of G induced by the set of all leaves in the subtree of T rooted at x, denoted by $G(x)$.

0-node. The children of a 0-node x correspond to the components obtained by a 0-operation applied to the disconnected graph $G(x)$.
1-node. The children of a 1-node x correspond to the components obtained by a 1-operation applied to the not co-connected graph $G(x)$.
2-node The children of a 2-node x correspond to the subgraphs induced by the maximal homogeneous sets or single vertices of the connected and co-connected graph $G(x)$. Additionally, the characteristic graph of $G(x)$ is assigned to the 2-node x.

The modular decomposition tree is of basic importance for many algorithmic applications, and in [22, 13, 14], linear time algorithms are given for determining the modular decomposition tree of an input graph.

Often, algorithms exploiting the modular decomposition have the following structure. Let Π be a graph problem to be solved on some graph class $\mathcal{G}$, e.g., Maximum

Stable Set on (P_5,gem)-free graphs. First the algorithm computes the modular decomposition tree T of the input graph G using one of the linear time algorithms. Then in a bottom up fashion the algorithm computes for each node x of T the optimal value for the subgraph $G(x)$ of G induced by the set of all leaves of the subtree of T rooted at x. Thus the computation starts assigning the optimal value to the leaves. Then the algorithm computes the optimal value of an interior node x by using the optimal values of all children of x depending on the type of the node. Finally the optimal value of the root is the optimal value of Π for the input graph G. (Note that various more complicated variants of this method can be useful.)

Thus to specify such a modular decomposition based algorithm we only have to describe how to obtain the value for the leaves, and which formula to evaluate or which subproblem to solve on 0-nodes, 1-nodes and 2-nodes, using the values of all children as input. It is well-known how to do this for 0-nodes and 1-nodes for the NP-complete graph problems Maximum Weight Stable Set, Maximum Weight Clique, Minimum Coloring and Minimum Clique Cover from the corresponding cograph algorithm [10, 11]. On the other hand to find out the algorithmic problem to solve on 2-nodes, called the 2-node subproblem, for solving problem Π using modular decomposition can be quite challenging.

4 The Structure Theorem for (P_5,gem)-Free Graphs

To state the Structure Theorem of (P_5,gem)-free graphs we need to define three classes of (P_5,gem)-free graphs which together contain all prime (P_5,gem)-free graphs.

Definition 1. *A graph $G = (V, E)$ is called* matched cobipartite *if its vertex set V is partionable into two cliques C_1, C_2 with $|C_1| = |C_2|$ or $|C_1| = |C_2| - 1$ such that the edges between C_1 and C_2 form a matching and at most one vertex in C_1 and at most one vertex in C_2 are not covered by the matching.*

Definition 2. *A graph G is called* specific *if it is the complement of a prime induced subgraph of one of the three graphs in Figure 1.*

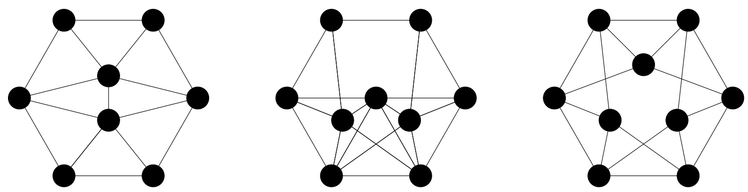

Fig. 1.

To establish a definition of the third class of prime graphs, we do need some more notions. A graph is *chordal* if it contains no induced cycles C_k, $k \geq 4$. See e.g. [8] for properties of chordal graphs. A graph is *cochordal* if its complement graph is chordal. A vertex v is *simplicial* in G if its neighborhood $N(v)$ in G is a clique. A vertex v is *cosimplicial* in G if it is simplicial in $\overline{G}$. It is well-known that every chordal graph has a simplicial vertex and that such a vertex can be found in linear time.

We also need the following kind of substituting a C_5 into a vertex: For a graph G and a vertex v in G, let the result of the extension operation $ext(G, v)$ denote the graph G' resulting from G by replacing v with a C_5 $(v_1, v_2, v_3, v_4, v_5)$ of new vertices such that v_2, v_4 and v_5 have the same neighborhood in G as v, and v_1, v_3 have only their C_5 neighbors, i.e. have degree 2 in G'. For a vertex set $U \subseteq V$ of G, let $ext(G, U)$ denote the result of applying repeatedly the extension operation to all vertices of U. Note that the resulting graph does not depend on the order of replacing U vertices.

Definition 3. *For $k \geq 0$, let $\mathcal{C}_k$ be the class of prime graphs $G' = ext(G, Q)$ resulting from a (not necessarily prime) cochordal gem-free graph G by extending a clique Q of exactly k cosimplicial vertices of G. Thus, $\mathcal{C}_0$ is the class of prime cochordal gem-free graphs.*

Clearly each graph in $\mathcal{C}_k$ contains k C_5's which are vertex-disjoint. It is also known that each graph in $\mathcal{C}_k$ has neither $\overline{C_4}$ nor $\overline{C_6}$ as an induced subgraph [6].

Lemma 1. *Let $G = (V, E)$ be a graph of $\mathcal{C}_k$, $k \geq 1$. Then for every C_5 $C = (v_1, v_2, v_3, v_4, v_5)$ of G, the vertex set V has a partition into $\{v_1, v_2, v_3, v_4, v_5\}$, the stable set A of 0-vertices for C and the set B of 3-vertices for C such that all vertices of B have the same non consecutive neighbors in C, say v_2, v_4, v_5, and $G[B]$ is a cograph.*

Theorem 2 (Structure Theorem [6]). *A connected and co-connected graph G is (P_5,gem)-free if and only if the following conditions hold:*

(1) *The homogeneous sets of G are P_4-free (i.e., induce a cograph);*
(2) *For the characteristic graph G^* of G, one of the following conditions holds:*
  (2.1) *G^* is a matched co-bipartite graph;*
  (2.2) *G^* is a specific graph;*
  (2.3) *there is a $k \geq 0$ such that G^* is in $\mathcal{C}_k$.*

Consequently, the modular decomposition tree T of any connected (P_5,gem)-free G contains at most one 2-node. If G is a cograph then T has no 2-node. If G is not a cograph then the only 2-node of T is its root.

5 An Algorithm for Minimum Coloring on (P_5,gem)-Free Graphs

In this section we present a linear time algorithm for the Minimum Coloring problem on (P_5,gem)-free graphs. That is we are given a (P_5,gem)-free graph G, and want to determine $\chi(G)$.

Minimum Coloring is not LinEMSOL($\tau_{1,L}$) definable. Nevertheless there is a polynomial time algorithm for graphs of bounded clique-width [20]. However this algorithm is only of theoretical interest. For graphs of clique-width at most five (which is the best known upper bound for the clique-width of (P_5,gem)-free graphs [7]), the exponent r of the running time $O(n^r)$ of this algorithm is larger than 2000.

5.1 The subproblems

We use the approach discussed in Section 3. Thus, we start by computing (in linear time) the modular decomposition tree T of G. For each node x of T, we compute $\chi(G(x))$. Suppose $x_1, x_2, \ldots, x_r$ are the children of x. For leaves, 0-nodes, and 1-nodes x, the steps of the linear time algorithm for Minimum Coloring on cographs can be used: If x is a **leaf** of T then $\chi(G(x)) := 1$. If x is a **0-node**, then $\chi(G(x)) := \max\limits_{i=1,\ldots,r} \chi(G(x_i))$. If x is a **1-node**, then $\chi(G(x)) := \sum_{i=1}^{r} \chi(G(x_i))$.

Suppose x is a **2-node** of T. Let $G^* = (V^*, E^*)$ be the characteristic graph assigned to x. We assign to the vertex set V^* of G^* the weight function $w^* : V^* \to \mathbb{N}$ such that $w^*(v_i) := \chi(G(x_i))$. We have that $\chi(G(x)) := \chi_{w^*}(G^*)$.

Thus, the Minimum Coloring problem on (P_5,gem)-free graphs becomes the problem of computing the minimum number of colors for a weighted coloring of (G^*, w^*), where G^* is a prime (P_5,gem)-free graph. The remainder of this section is devoted to this problem. The Structure Theorem tells us that G^* either is a matched co-bipartite graph, a specific graph, or there is a $k \geq 0$ with G^* in $\mathcal{C}_k$. In three subsections, each of these cases will be dealt with. We also use the following notation and lemma.

Let $N = \sum_{v \in V^*} w^*(v)$ be the total weight. Observe that N is at most the number of vertices of the original (P_5,gem)-free graph.

Lemma 2. *Let G be a perfect graph and w be a vertex weight function of G. Then $\chi_w(G) = \omega_w(G)$ and $\kappa_w(G) = \alpha_w(G)$.*

Proof. Let G' be the graph obtained from G by substituting each vertex v of G by a clique of cardinality $w(v)$. As any weighted coloring of (G, w) corresponds to a coloring of G' and vice versa, we have $\chi_w(G) = \chi(G')$. Similarly, $\omega_w(G) = \omega(G')$.

Let G be perfect. Then $\overline{G}$ is perfect by Lovasz's Perfect Graph Theorem [21]. $\overline{G'}$ is obtained from the perfect graph $\overline{G}$ by vertex multiplication, and thus it is perfect [21]. As G' is the complement of a perfect graph ($\overline{G'}$), it is perfect. Since G' is perfect we have $\chi(G') = \omega(G')$ and thus $\chi_w(G) = \omega_w(G)$. Similarly, since $\overline{G'}$ is perfect we obtain $\chi_w(\overline{G}) = \omega_w(\overline{G})$. Hence $\kappa_w(G) = \alpha_w(G)$. $\square$

We now discuss how to solve the weighted coloring problem for each of the three classes of prime (P_5,gem)-free graphs.

5.2 Matched cobipartite graphs

The graph G^* is cobipartite and thus perfect. By Lemma 2 we obtain $\chi_{w^*}(G^*) = \omega_{w^*}(G^*)$. One easily finds in linear time a partition of the vertex set of G^* into two cliques, C_1, and C_2. Now, as each maximal clique of G^* is either C_1, C_2, or an edge of G^*, $\omega_{w^*}(G^*) = \chi_{w^*}(G^*)$ can be computed by a linear time algorithm.

5.3 Specific graphs

Each specific graph G^* is a prime induced subgraph of the complement of one of the three graphs in Figure 1. To solve the weighted coloring problem on specific graphs, we formulate this problem as an integer linear programming problem, and then argue that this ILP can be solved in constant time.

Consider the specific graph G^* with weights w^*. Let $\mathcal{S}$ be the collection of all maximal stable sets of G^*. We build an integer linear programming with for each $S \in \mathcal{S}$ a variable x_S, as follows.

$$\text{minimize} \sum_{S\in\mathcal{S}} x_S \quad \text{such that} \tag{1}$$

$$\sum_{v\in S, S\in\mathcal{S}} x_S \geq w(v) \quad \text{for all } v \in V \tag{2}$$

$$x_S \geq 0 \quad \text{for all } S \in \mathcal{S} \tag{3}$$

$$x_S \text{ integer} \quad \text{for all } S \in \mathcal{S} \tag{4}$$

With $\boldsymbol{x}$ we denote a vector containing for each $S \in \mathcal{S}$ a value x_S.

Let z be the optimal value of this ILP. z equals the minimum number of colors needed for (G^*, w^*). If we have a coloring of (G^*, w^*) with a minimum number of colors, then assign to each color one maximal stable set $S \in \mathcal{S}$, such that this color is given to (a subset of) all vertices in S. Let x_S be the number of colors assigned to S. Clearly, x_S is a non-negative integer. For each $v \in V$, as v has $w(v)$ colors, we have $\sum_{v\in S, S\in\mathcal{S}} x_S \geq w(v)$. $\sum_{S\in\mathcal{S}} x_S$ equals the number of colors. Conversely, suppose we have an optimal solution x_S of the ILP. For each $S \in \mathcal{S}$, we can take a set of x_S unique colors, and use these colors to color the vertices in x_S. As S is stable, this gives a proper coloring, and as $\sum_{v\in S, S\in\mathcal{S}} x_S \geq w(v)$, each vertex has sufficiently many colors available. So, this gives a coloring of (G^*, w^*) with z colors.

The relaxation of the ILP is the linear program, obtained by dropping the integer condition (4):

$$\text{minimize} \sum_{S\in\mathcal{S}} x_S \quad \text{such that} \tag{5}$$

$$\sum_{v\in S, S\in\mathcal{S}} x_S \geq w(v) \quad \text{for all } v \in V \tag{6}$$

$$x_S \geq 0 \quad \text{for all } S \in \mathcal{S} \tag{7}$$

Let $\boldsymbol{x}'$ be an optimal solution of this relaxation, with value $z' = \sum_{S\in\mathcal{S}} x'_S$.

As G^* is a specific graph, the linear program has a constant number of variables (namely, the number of maximal stable sets of G^*) and a constant number of constraints (at most nine, one per vertex of G^*), and hence can be solved in constant time. (E.g., enumerate all corners of the polyhedron spanned by program, and take the optimal one.) Note that we can write the linear program in the form $\max\{cx \mid Ax \leq b\}$, such that each element of A is either 0 or 1. Let Δ be the maximum value of a subdeterminant of this matrix A. It follows that Δ is bounded by a constant. Write $s = |\mathcal{S}|$.

Now we can use a result of Cook, Gerards, Schrijver, and Tardos, see Theorem 17.2 from [23]. This theorem tells us that the ILP has an optimal solution $\boldsymbol{x}''$, such that for each $S \in \mathcal{S}$, $|x'_S - x''_S| \leq s\Delta$.

Thus, the following is an algorithm that finds the optimal solution to the ILP (and hence the number of colors needed for (G^*, w^*)) in constant time. First, find an optimal solution $\boldsymbol{x}'$ of the relaxation. Then, enumerate all integer vectors $\boldsymbol{x}''$ with for all $S \in \mathcal{S}$, $|x'_S - x''_S| \leq s\Delta$. For each such $\boldsymbol{x}''$, check if it fulfils condition (2), and select the solution vector that fulfils the conditions with the minimum value. By Theorem 17.2 from [23], this is an optimal solution of the ILP. This method takes constant time, as s and Δ are bounded by constants, and thus 'only' a constant number of vectors have to be checked, and each is of constant size.[2]

A straightforward implementation of this procedure would not be practical, as more than $(s\Delta)^s$ vectors are checked, with s the number of maximal stable sets in one of the specific graphs. In a practical setting, one could first solve the linear program, and use that value as starting point in a branch and bound procedure.

Remark 1. The method not only works for the specific graphs, but for any constant size graph. This implies that Minimum Coloring can be solved in linear time for graphs whose modular decomposition has a constant upper bound on the size of the characteristic graphs.

Remark 2. In the full version we present an $O(N^3)$ time algorithm to solve the weighted coloring of the specific graphs, that has no large hidden constant in the running time.

5.4 $\bigcup_{k=0}^{\infty} \mathcal{C}_k$

Let $G^* \in \mathcal{C}_k$, for some $k \geq 0$, and w^* the weight function G^*. All C_5's of G^* can be computed by a linear time algorithm that first computes all vertices of degree two.

If $G^* = C_5$ then with the technique applied to specific graphs $\chi_{w^*}(G^*)$ can be computed in constant time. If $G^* \in \mathcal{C}_0$ then it is cochordal and thus perfect. Hence $\chi_{w^*}(G^*) = \omega_{w^*}(G^*)$ by Lemma 2.

Lemma 3. *The Maximum Weight Clique problem and the weighted coloring problem can be solved by a linear time algorithm for cochordal graphs.*

Proof. Frank [16] gave a linear time algorithm to compute the maximum weight of a stable set of a chordal graph G. This implies that there is an $O(n^2)$ algorithm to compute the maximum weight of a clique in a cochordal graph $\overline{G}$ since $\omega_w(\overline{G}) = \alpha_w(G)$. To get a linear time algorithm, we must avoid the complementation; thus, we simulate Frank's algorithm applied to $\overline{G}$. This is Frank's algorithm: First it computes a perfect elimination ordering $v_1, \ldots, v_n$ of the input chordal graph $G = (V, E)$. Then a maximum weight stable set is constructed as follows. Initially, let $c\_w(v_i) = w(v_i)$, for all $1 \leq i \leq n$. For each i from 1 to n, if $c\_w(v_i) > 0$ then colour v_i red, and subtract $c\_w(v_i)$ from $c\_w(v_j)$ for all $v_j \in \{v_i\} \cup (N(v_i) \cap \{v_{i+1}, \ldots, v_n\})$. After all vertices have been processed, set $I = \emptyset$ and, for each i from n down to 1, if v_i is red and not adjacent to any vertex of I then $I = I \cup \{v_i\}$. When all vertices have been processed again, the algorithm terminates and outputs the maximum weight stable set I of (G, w).

We now describe our simulation of this algorithm. First a perfect elimination ordering $v_1, v_2, \ldots, v_n$ of $\overline{G}$ is computed in linear time (see e.g. [22]).

The maximum weight of a clique of G is constructed as follows. Initially, let $W' = 0$ and $s(v_i) = 0$ for all i ($1 \leq i \leq n$). For each i from 1 to n, if $w(v_i) - W' + s(v_i) > 0$

[2] Computer computation shows that $\Delta \leq 3$ for specific graphs.

then colour v_i red, set $W' = w(v_i) + s(v_i)$ and add $w(v_i) - W' + s(v_i)$ to $s(v_j)$ for all $v_j \in (N(v_i) \cap \{v_{i+1}, \ldots, v_n\})$.

After all vertices have been processed, set $K = \emptyset$ and, for each i from n down to 1, if v_i is red and adjacent to all vertices of K then $K = K \cup \{v_i\}$. Finally the algorithm outputs the maximum weight clique K of (G, w).

Clearly our algorithm runs in linear time. Its correctness follows from the fact that when treating the vertex v_i, the difference $W' - s(v_i)$ is precisely the value the original Frank algorithm applied to the complement of G would have subtracted from $c\_w(v_i)$ up to the point when it treats v_i. Thus our algorithm simulates Frank's algorithm on $\overline{G}$, and thus it is correct. □

In the remaining case, we consider a prime graph $G^* \in \mathcal{C}_k$, $k \geq 1$ such that $G^* \neq C_5$.

Lemma 4. *Let $k \geq 1$, $G^* \in \mathcal{C}_k$ and $G^* \neq C_5$. Let $C = (v_1, v_2, v_3, v_4, v_5)$ be a C_5 in G^* and v_1 and v_3 its vertices of degree two. Let w^* be the vertex weight function of G^*. Then there is a minimum weight coloring $\mathcal{S}^*$ of (G^*, w^*) with precisely $\max(w^*(v_2), w^*(v_4) + w^*(v_5))$ stable sets containing at least one of the vertices of $\{v_2, v_4, v_5\}$.*

Proof. By Lemma 1, the set A of 0-vertices for $C = (v_1, v_2, v_3, v_4, v_5)$ is a stable set, $B = V^* \setminus (C \cup A) = N(v_2) \setminus C = N(v_4) \setminus C = N(v_5) \setminus C$, and $G^*[B]$ is a cograph.

Let $\mathcal{S}$ be any minimum weight coloring of (G^*, w^*). Since $N(v_1) \setminus C = N(v_3) \setminus C = \emptyset$ and $N(v_2) \setminus C = N(v_4) \setminus C = N(v_5) \setminus C = B$ we may assume that every stable set of $\mathcal{S}$ contains either none or two vertices of C. Therefore we study weighted colorings of a C_5 $C = (v_1, v_2, v_3, v_4, v_5)$ of G^* with vertex weights w^*, where all stable sets are non edges of C and call them partial weight colorings (abbr. pwc) of C. Clearly any pwc of $C = (v_1, v_2, v_3, v_4, v_5)$ contains at least $w^*(v_2)$ stable sets containing v_2, and it contains at least $w^*(v_4) + w^*(v_5)$ stable sets containing v_4 or v_5.

Let $\mathcal{S}'$ be a weighted coloring of G^* containing the smallest possible number of stable sets S with $S \cap \{v_2, v_4, v_5\} \neq \emptyset$. Let t be the number of stable sets S of $\mathcal{S}'$ satisfying $S \cap \{v_2, v_4, v_5\} \neq \emptyset$ and suppose that, contrary to the statement of the lemma, $t > \max(w^*(v_2), w^*(v_4) + w^*(v_5))$. Let $s(v)$ be the number of stable sets of $\mathcal{S}'$ containing the vertex v. Then $t > w^*(v_4) + w^*(v_5)$ implies $s(v_4) > w^*(v_4)$ or $s(v_5) > w^*(v_5)$. W.l.o.g. we may assume $s(v_4) > w^*(v_4)$. Hence there is a stable set $S' \in \mathcal{S}'$ containing v_4. Consequently either $S' \subseteq \{v_2, v_4\} \cup A$ or $S' \subseteq \{v_1, v_4\} \cup A$. In both cases we replace the stable set S' of $\mathcal{S}'$ by $\{v_1, v_3\} \cup A$. Thus the replacement decrements the number of stable sets containing v_4 and possibly the number of stable sets containing v_2. Thus we obtain a new weighted coloring $\mathcal{S}''$ of G^* with $t - 1$ stable sets S with $S \cap \{v_2, v_4, v_5\} \neq \emptyset$. This contradicts the choice of t. Consequently $t = \max(w^*(v_2), w^*(v_4) + w^*(v_5))$. □

To extend any pwc of a C_5 C to G^* only two parameters are important: the number a of stable sets $\{v_1, v_3\}$ in the pwc of C, and the number b of non edges in the pwc of C different from $\{v_1, v_3\}$. Each of the a stable sets $\{v_1, v_3\}$ in the pwc of C, can be extended to a maximal stable set $\{v_1, v_3\} \cup A'$ of G^*, where A' is some maximal stable set of $G^* - C$. Each of the b non edges S, $S \neq \{v_1, v_3\}$, in the pwc of C has the unique extension to the maximal stable set $S \cup A$ of G^*.

By Lemma 4, for each C_5 of G^* there is a minimum weight coloring of G^* extending a pwc of the C_5 C with $b = \max(w^*(v_2), w^*(v_4) + w^*(v_5))$. Taking such a minimum weight coloring we can clearly remove vertices v_1 and v_3 from stable sets containing both until we obtain the smallest possible value of a in a pwc of C with $b = \max(w^*(v_2), w^*(v_4) + w^*(v_5))$.

Finally given a C_5 C, the smallest possible value of a in a pwc of C with $b = \max(w^*(v_2), w^*(v_4) + w^*(v_5))$ can be computed in constant time. (Details omitted.)

Now we are ready to present our coloring algorithm that computes a minimum weight coloring of (G^*, w^*) for a graph G^* of $\mathcal{C}_k$, $k \geq 1$. It removes at most k times the precomputed C_5 from the current graph until the remaining graph has no C_5 and is therefore a cochordal graph. Then by Lemma 3 there is an algorithm to solve the weighted coloring problem for the cochordal graph in linear time.

In each round, i.e. when removing one C_5 $C = (v_1, v_2, v_3, v_4, v_5)$ from the current graph G' with current weight function w', the algorithm proceeds as follows: First it computes in constant time a pwc of C such that $b = \max(w'(v_2), w'(v_4) + w'(v_5))$ and a as small as possible. Then the algorithm removes all vertices of C and obtains the graph $G'' = G' - C$. Furthermore it removes all vertices of the stable set A of 0-vertices for C in G' with weight at most a and decrements the weight of all other vertices in A by a. Recursively the algorithm solves the minimum weight coloring problem on the obtained graph G'' with weight function w''. Finally the minimum number of stable sets in a weighted coloring of (G', w') is obtained using the formula $\chi_{w'}(G') = a + \max(b, \chi_{w''}(G''))$.

Thus the algorithm removes at most $k \leq n$ times a C_5. Each pwc of a C_5 can be computed in constant time. For the final cochordal graph the minimum weight coloring can be solved in linear time. Hence the overall running time of the algorithm is linear. We have given a linear time algorithm for the weighted coloring problem for $\bigcup_{k \geq 0} \mathcal{C}_k$. We can finally conclude:

Theorem 3. *There is a linear time algorithm to solve the Minimum Coloring problem on (P_5,gem)-free graphs.*

6 Conclusion

We have shown how modular decomposition and the Structure Theorem for (P_5,gem)-free graphs can used to obtain a linear time algorithm to solve the Minimum Coloring problem. In a quite similar way one can construct a linear time algorithm to solve the Minimum Clique Cover problem on (P_5,gem)-free graphs. Modular decomposition can also be used to obtain linear time algorithms for the LinEMSOL($\tau_{1,L}$) definable NP-complete graph problems Maximum Weight Stable Set and Maximum Weight Clique on (P_5,gem)-free graphs. These algorithms are given in the full version of this paper.

Acknowledgement

Thanks are due to Alexander Schrijver for pointing towards Theorem 17.2 from his book [23].

References

1. S. ARNBORG, J. LAGERGREN, D. SEESE, Easy problems for tree-decomposable graphs, *J. Algorithms* 12 (1991), 308-340.
2. H. BODLAENDER, Achromatic number is NP-complete for cographs and interval graphs, *Inform. Process. Lett.* 31 (1989) 135-138
3. H.L. BODLAENDER, H.J. BROERSMA, F.V. FOMIN, A.V. PYATKIN, G.J. WOEGINGER, Radio labeling with pre-assigned frequencies, *Proceedings of the 10th European Symposium on Algorithms (ESA'2002)*, LNCS 2461 (2002) 211-222
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, *Proceedings of the 8th Scandinavian Workshop on Algorithm Theory (SWAT'2002)*, LNCS 1851 (2002) 388-397
6. A. BRANDSTÄDT, D. KRATSCH, On the structure of (P_5,gem)-free graphs, *Manuscript* 2002
7. A. BRANDSTÄDT, H.-O. LE, R. MOSCA, Chordal co-gem-free graphs have bounded clique width, *Manuscript* 2002
8. A. BRANDSTÄDT, V.B. LE, J. SPINRAD, Graph Classes: A Survey, *SIAM Monographs on Discrete Math. Appl., Vol.* 3, SIAM, Philadelphia (1999)
9. M. CHUDNOVSKY, N. ROBERTSON, P.D.SEYMOUR, R.THOMAS, The Strong Perfect Graph Theorem, *Manuscript* 2002
10. D.G. CORNEIL, H. LERCHS, L. STEWART-BURLINGHAM, Complement reducible graphs, *Discrete Applied Math.* 3 (1981) 163-174
11. D.G. CORNEIL, Y. PERL, L.K. STEWART, Cographs: recognition, applications, and algorithms, *Congressus Numer.* 43 (1984) 249-258
12. 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
13. A. COURNIER, M. HABIB, A new linear algorithm for modular decomposition, *Trees in Algebra and Programming - CAAP '94*, LNCS 787 (1994) 68-84
14. E. DAHLHAUS, J. GUSTEDT, R.M. MCCONNELL, Efficient and practical algorithms for sequential modular decomposition, *J. Algorithms* 41 (2001) 360-387
15. W. ESPELAGE, F. GURSKI, E. WANKE, How to solve NP-hard graph problems on clique-width bounded graphs in polynomial time, *Proceedings of the 27th Workshop on Graph-Theoretic Concepts in Computer Science (WG 2001)*, LNCS 2204 (2001) 117-128
16. A. FRANK, Some polynomial algorithms for certain graphs and hypergraphs, *Proceedings of the Fifth British Combinatorial Conference (Univ. Aberdeen, Aberdeen, 1975)* 211-226, *Congressus Numerantium* No. XV, Utilitas Math., Winnipeg, Man. (1976)
17. T. GALLAI, Transitiv orientierbare Graphen, *Acta Mathematica Academiae Scientiarum Hungaricae* 18 (1967) 25-66
18. V. GIAKOUMAKIS, I. RUSU, Weighted parameters in ($P5$, $\overline{P5}$)-free graphs, *Discrete Appl. Math.* 80 (1997) 255-261
19. K. JANSEN, P. SCHEFFLER, Generalized coloring for tree-like graphs, *Discrete Appl. Math.* 75 (1997) 135-155
20. D. KOBLER, U. ROTICS, Edge dominating set and colorings on graphs with fixed clique-width, *Discrete Appl. Math.* 126 (2003) 197-221
21. L. LOVÁSZ, Normal hypergraphs and the perfect graph conjecture, *Discrete Math.* 2 (1972) 253-267
22. R.M. MCCONNELL, J. SPINRAD, Modular decomposition and transitive orientation, *Discrete Math.* 201 (1999) 189-241
23. A. SCHRIJVER, *Theory of Linear and Integer Programming*, John Wiley & Sons, Chichester, 1986.