On forests, stable sets and polyhedras associated with clique partitions

Denis Cornaz *

March 14, 2006

Abstract

Let $G = (V, E)$ be a simple graph on n nodes. We show how to construct a partial subgraph D of the line graph of G satisfying the identity: $\overline{\chi}(G) + \alpha(D) = n$, where $\overline{\chi}(G)$ denotes the minimum number of cliques in a clique partition of G and $\alpha(D)$ denotes the maximum size of a stable set of D. This is based on correspondences between the cliques partitions and the clique-connecting forests of G. We use this to develop a cutting-plane algorithm for the graph coloring problem that is tested on random and DIMACS benchmark graphs.
Keywords: Graph coloring, maximum stable set, polytope.
AMS subject classifications: 05C15, 90C09.

1 Introduction

Let $G = (V, E)$ be a simple undirected graph. A *clique of* G is a set of nodes any two of which are adjacent. A *clique partition of* G is a partition of V into cliques. The minimum number of cliques in a clique partition of G is denoted by $\overline{\chi}(G)$. The *minimum clique partition problem* is to determine $\overline{\chi}(G)$. The *graph coloring problem* is to determine the minimum number, denoted $\chi(G)$, of stable sets in a partition of V into stable sets (a *stable set* is a set of nodes any two of which are nonadjacent). It is well-known that both problems are equivalent since $\overline{\chi}(G) = \chi(\overline{G})$ (where $\overline{G}$ is the complementary graph of G).

*France Telecom R&D, 38-40, rue du Gal Leclerc, 92794 Issy-les-Moulineaux Cedex 9, France. E-mail: denis.cornaz@francetelecom.com. Phone: +33 1 45 29 40 64. Fax: +33 1 45 29 60 69.

The minimum clique partition problem arises in many applications such as scheduling, timetabling, electronic bandwith allocation, sequencing, computer registration allocation and many other areas (though often under the name of graph coloring). Several authors proposed exact algorithms for the problem based on 0-1 linear programming formulation. In [10, 7], the authors use an efficient column generation approach based on a formulation of the problem with an exponential number of variables associated to maximal cliques of the graph. In [3], the authors use a formulation with a polynomial number of variables corresponding to clique assignements to nodes. That formulation has many symmetrical solutions and it requires the use of powerfull families of valid inequalities to be efficient in practice.

Our approach of the minimum clique partition problem is the following: we show that the minimum clique partition problem is equivalent to finding a maximum forest of G which satisfies a property (that we have called "clique-connecting"). Based on this, we obtain a new integer linear programming formulation, denoted (IP_{forest}), of the minimum clique partition problem. But (IP_{forest}) has an exponential number of constraints and many symmetrical solutions. However, the symmetry of (IP_{forest}) can be broken with the help of an arbitrary order on the nodes of G. Given an order, we find two ILP formulations with a one-to-one correspondence between the solution vectors and the clique partitions of the graph. Surprisingly, the first one, denoted (IP_{stab}), is the simple formulation of the maximum stable set problem in an auxiliary edge graph of G. The inequalities of the second one, denoted (IP_{path}), are associated to an exponential number of paths.

The paper is organized as follows. At the end of this section we give the notation. In Section 2 we present (IP_{forest}). We show that its continuous relaxation can be solved in polynomial time. We give also a class of non-trivial facets for the integer polytope associated with (IP_{forest}). In Section 3 we prove the validity of (IP_{stab}) and (IP_{path}). Given an order $\mathcal{O}$ on V, we show how to construct a subgraph $D(G, \mathcal{O})$ of the line graph of G with a one-to-one correspondence between the stable sets of $D(G, \mathcal{O})$ and the clique partitions of G, and we study the structure of $D(G, \mathcal{O})$. We show also that the continuous relaxation of (IP_{path}) can be solved in polynomial time. In Section 4 we present some computational experiments with random and DIMACS benchmark graphs.

Notation. We consider a simple undirected graph $G = (V, E)$. Let $n := |V|$. An edge $e \in E$ is denoted by $e = uv$ where $u, v \in V$ are the endnodes of e. The set of edges *induced* by $U \subseteq V$ is denoted by $E[U] = \{uv \in E : u, v \in U\}$ and $G[U] = (U, E[U])$. The *nodes of* $F \subseteq E$ are the nodes incident to an edge in F and the set of nodes of F is denoted by $V(F)$. A *path of* G *linking* v_o, v_k is a subset of edges $P \subseteq E$ such that $P = v_o v_1, \ldots, v_{k-1} v_k$ where $v_o, v_1, \ldots, v_k$ are distinct nodes. If all the nodes are distinct except $v_o = v_k$, then P is a *cycle*. An induced cycle C with $|C| \geq 4$ is a *hole* (and an *anti-hole* in $\overline{G}$). If $F \subseteq E$ is a *forest* (i.e. F contains no cycle) and if the connected components of the (partial) subgraph (V, F) of G are $(V_1, T_1), \ldots, (V_k, T_k)$, then T_i is called a *tree of* F and T_i is called a *spanning tree of* V_i. (Note that a tree may be empty.) A subset of edges incident to a same node is a *star*. A star of size 3 is a *claw*. As usual, K_n is the complete graph on n nodes and $K_n \setminus e$ is obtained from K_n by removing one edge. The *forest (resp. matching) polytope of* G is the convex hull of the incidence vectors (in $\{0,1\}^E$) of the forests (resp. matchings) of G. (A *matching* is a subset of pairwise disjoint edges.) The *line graph* $L(G)$ *of* G is the graph with node-set E and two nodes are linked by an edge in $L(G)$ whenever the corresponding edges are adjacent in G. The *clique number of* G, denoted $\omega(G)$, is the maximum cardinality of a clique of G. The maximum cardinality of a stable set of G is denoted by $\alpha(G)$. For every vector $x \in \mathbb{R}^E$, we let $x(F) = \sum_{e \in F} x_e$. (So $x(E) = \mathbf{1}^\mathsf{T} x$.) For every node v we let $N(v) = \{w \in V : \exists vw \in E\}$ and $\delta(v) = \{vw : w \in N(v)\}$.

2 The clique-connecting forest polytope

A forest F of G is said to be *clique-connecting* if, for each tree T of F, the nodes of T form a clique of G. (Note that $\emptyset$ is a clique-connecting forest.) Let the *clique-connecting forest polytope* of G, denoted by $CCF(G)$, be the convex hull of the incidence vectors (in $\{0,1\}^E$) of the clique-connecting forests of G.

The following lemma is the key of our approach of the minimum clique partition problem (it will be used in the proof of Theorem 3.4). The proof is based on the well-known fact that $k + |F| = n$ for every forest F of G such that (V, F) has k connected components.

Lemma 2.1 *$\overline{\chi}(G)+|F| = n$, for every graph G and every clique-connecting forest F of G with $|F|$ maximum.*

Proof. Let F be a maximum cardinality clique-connecting forest of G and let $V_1, \ldots, V_{\overline{\chi}(G)}$ be a minimum clique partition of G. Let $(U_1, T_1), \ldots, (U_k, T_k)$ be the connected components of (V, F), so $U_1, \ldots, U_k$ is a clique partition of G. Hence $\overline{\chi}(G) \leq k = n - |F|$. Let T_i be a spanning tree of V_i, so $\bigcup_i T_i$ is a clique-connecting forest of G. Hence $|F| \geq \sum_i |T_i| = n - \overline{\chi}(G)$. □

Corollary 2.2 $\overline{\chi}(G) = n - \max_{x \in CCF(G)} \mathbf{1}^{\mathsf{T}} x$. □

Finding the maximum value of $\mathbf{1}^{\mathsf{T}} x$ with $x \in CCF(G)$ is equivalent to the following 0-1 linear program (IP_{forest}).

$$
\begin{array}{l}
\max \mathbf{1}^{\mathsf{T}} x \\
\text{s.t.} \\
\qquad x_e \quad \in \{0,1\} \qquad\qquad\qquad\qquad\qquad \text{for } e \in E, \\
x(E[U]) \quad \leq |U| - \left\{ \begin{array}{ll} 1 & \text{if } U \text{ is a clique} \\ 2 & \text{otherwise} \end{array} \right. \qquad \text{for nonempty } U \subseteq V.
\end{array}
$$

If we replace the integrity constraints $x_e \in \{0,1\}$ by the trivial constraints $0 \leq x_e \leq 1$ for $e \in E$ in (IP_{forest}), we obtain a linear program (LP_{forest}). The proof of Proposition 2.3 below is an adaptation of the proof of [15, 14] for the separation problem associated to the inequalities $x(E[U]) \leq |U| - 1$. (See also [18], vol. B, pp. 880-881.)

Proposition 2.3 *The linear program (LP_{forest}) can be solved in polynomial time.*

Proof. By [8] we only need to solve the following separation problem in polynomial time: given $x \in \mathbb{Q}_+^E$, decide if x satisfies the constraints of (LP_{forest}), and if not, find an inequality violated by x. Since the number of the trivial constraints is polynomial, we can focus on the remaining inequalities.
Define $y_v := 2 - x(\delta(v))$ for $v \in V$. Then $2(x(E[U]) - |U|) = -x(\delta(U)) - y(U)$. So any nonempty set U_1 with $U_1 \subseteq V$ minimizing $x(\delta(U_1)) + y(U_1)$, maximizes $x(E[U_1]) - |U_1|$. By [16] we can find such a U_1 in polynomial time. If $x(E[U_1]) \leq |U_1| - 1$, then x satisfies $x(E[U]) \leq |U| - 1$ for every nonempty $U \subseteq V$, and otherwise U_1 gives a violated inequality. Let $\mathcal{S}_2$ be the set of

the stable sets of G with cardinality 2. Any set U_2 with $S \subseteq U_2 \subseteq V$ for some $S \in \mathcal{S}_2$ minimizing $x(\delta(U_2)) + y(U_2)$ maximizes $x(E[U_2]) - |U_2|$. Since $|\mathcal{S}_2|$ is polynomial, by [16], we can find such a U_2 in polynomial time. If $x(E[U_2]) \leq |U_2| - 2$, then x satisfies $x(E[U]) \leq |U| - 2$ for every non-clique $U \subseteq V$, and otherwise U_2 gives a violated inequality. □

The rank of a subset $E' \subseteq E$, denoted $r(E')$, is the maximum size of a clique-connecting forest F' of G such that $F' \subseteq E'$.

Example 2.4 *1) Let v be a node of G. Then $r(\delta(v)) = \omega(G[N(v)])$.*
2) Let U be a subset of nodes. Then $r(E[U]) = |U| - \overline{\chi}(G[U])$.

A rank inequality associated with a subset $E' \subseteq E$ is an equality of the form $x(E') \leq r(E')$. Clearly, a rank inequality is satisfied by the incidence vector of every clique-connecting forest of G, and so it is valid for $CCF(G)$.

Remark that, if $Match(G)$ denotes the matching polytope and $Forest(G)$ the forest polytope of G, then:

- $Match(G) \subseteq CCF(G) \subseteq Forest(G)$,
- $Match(G) = CCF(G) \Leftrightarrow G$ is triangle-free,
- $CCF(G) = Forest(G) \Leftrightarrow$ each component of G is a complete graph.

It is easily seen that $CCF(G)$ is full-dimensional and that the nonnegativity constraints are facet-determining for $CCF(G)$. A trivial class of graphs for which $CCF(G)$ is decribed by the inequalities of (LP_{forest}) is the class of graphs each of whose components is a complete graph, because, by Edmond's independent set polytope of a matroid theorem [5], $Forest(G)$ is described by $x_e \geq 0$ for $e \in E$ and $x(E[U]) \leq |U| - 1$ for nonempty $U \subseteq V$. A less trivial class for which $CCF(G)$ is described by the rank inequalities is the class of triangle-free graphs, because, by Edmond's matching polytope theorem [4], $Match(G)$ is described by $x_e \geq 0$ for $e \in E$, $x(\delta(v)) \leq 1$ for $v \in V$, and $x(E[U]) \leq \frac{|U|-1}{2}$ for $U \subseteq V$ with $|U|$ odd.

A clique inequality is a rank inequality associated with a maximal subset with rank 1. It is not hard to see that every clique inequality is associated with a maximal induced star of G. Note that finding a stable set with maximum cardinality in a graph H can be reduced to finding an induced star with maximum cardinality in the graph obtained from H by adding a node adjacent to all the other nodes of H. In consequence, the following

separation problem is NP-hard: given $x \in \mathbb{Q}_+^E$, decide if x satisfies the clique inequalities, and if not, find a clique inequality violated by x.

A subset K of E is said to be a *clique-complete set with clique U* if there exists a clique U of G such that K is an inclusionwise maximal subset containing $E[U]$ with $r(K) = |U| - 1$. A *clique-complete set inequality* is a rank inequality associated with a clique-complete set. Notice that a maximal induced star is a clique-complete set with a clique of size 2, and hence the clique-complete set inequalities generalize the clique inequalities.

The following proposition states that the clique-complete set inequalities are facet-determining for $CCF(G)$.

Proposition 2.5 *Let $G = (V, E)$ be a graph and K be a clique-complete set with clique U. Then the inequality*

$$x(K) \leq |U| - 1 \tag{1}$$

determines a facet of the clique-connecting forest polytope.

Proof. Since $r(K) = |U| - 1$, inequality (1) is valid. To see that (1) determines a facet, let $a^\mathsf{T} x = \beta$ be satisfied by all x in the clique-connecting forest polytope with $x(K) = |U|-1$. So $a(F) = \beta$ for each clique-connecting forest F with $|F \cap K| = |U| - 1$. If $|U| = 2$, there is a clique-connecting forest $\{e\}$ where e is the unique edge in $E[U]$, then $a_e = \beta$. Otherwise let e_1, e_2 be two distinct edges in $E[U]$. Since U is a clique, we can assume that there exists two spanning trees of U, say T_1 and T_2, such that $T_1 \setminus T_2 = e_1$ and $T_2 \setminus T_1 = e_2$. Then $a(T_1) - a(T_2) = a_{e_1} - a_{e_2} = 0$, and hence $a_e = \beta/(|U| - 1)$ for every $e \in E[U]$. Now let $e \in K \setminus E[U]$. Since every edge in K is incident to at least one node in U, we can suppose that $e = uv$ with $u \in U$. Let T be a spanning tree of $U - u$. Since $F = T + e$ is a clique-connecting forest such that $|F \cap K| = |U| - 1$, $a(F) = a(T) + a_e = \beta$. Since $a(T) = (|U|-2) \times \beta/(|U|-1)$, then $a_e = \beta/(|U|-1)$. Also, $a_f = 0$ for each $f \in E \setminus K$. Indeed, since K is maximal, $r(K+f) = r(K)+1$, then there is a clique-connecting forest F such that $F \setminus K = f$ and $|F \cap K| = |U| - 1$, and hence $a(F) = \beta = a(F - f)$. So $a_f = 0$. Concluding, $a^\mathsf{T} x = \beta$ is some multiple of $x(K) = |U| - 1$, and hence (1) determines a facet. □

3 Avoiding permutations

In this section we focus on two classes of clique-connecting forests such that, given a clique partition of G and a class, there is exactly one clique-connecting forest in the class that corresponds to the clique partition. Each class is defined with the help of a (total) order on the nodes, that is a bijection $\mathcal{O} : V \leftrightarrow \{1, \ldots, n\}$. The *order of a node* v is $\mathcal{O}(v)$. We let δ_v^+ (resp. δ_v^-) be the set of the edges $vw \in \delta(v)$ with $\mathcal{O}(w) > \mathcal{O}(v)$ (resp. with $\mathcal{O}(w) < \mathcal{O}(v)$).

3.1 Stellar forests

A forest F is said to be *stellar* if each tree of F is a star. A clique-connecting stellar forest F of G is said to be *canonical (with respect to $\mathcal{O}$)* if, for each star S of F, the node of S with the lowest order is the center of S. The following remark is true.

Remark 3.1 *Given $\mathcal{O} : V \leftrightarrow \{1, \ldots, n\}$. There is a one-to-one correspondence between the canonical clique-connecting stellar forests of G and the clique partitions of G.*

Note that being a canonical clique-connecting stellar forest is an hereditary property with respect to inclusion. A pair of edges $\{e_1, e_2\}$ of G is said to be an *independent pair (w.r.t. $\mathcal{O}$)* if e_1 and e_2 are disjoint, or if e_1 and e_2 are adjacent and the two following conditions hold, say $e_1 = uv$ and $e_2 = vw$:

(i) $\mathcal{O}(v) < \min\{\mathcal{O}(u), \mathcal{O}(w)\}$, and

(ii) uw is an edge of the graph G.

A *dependent pair (w.r.t. $\mathcal{O}$)* is a pair of (adjacent) edges of G that is not independent. We define now the *dependent edge graph* $D(G, \mathcal{O})$ associated to a graph G and an order $\mathcal{O}$ on its nodes. Examples of dependent edge graphs are depicted in Figure 1.

Definition 3.2 *The dependent edge graph of G w.r.t. $\mathcal{O}$, denoted $D(G, \mathcal{O})$, is the graph whose nodes are the edges of G; two nodes of $D(G, \mathcal{O})$ are adjacent if the corresponding edges of G form a dependent pair.*

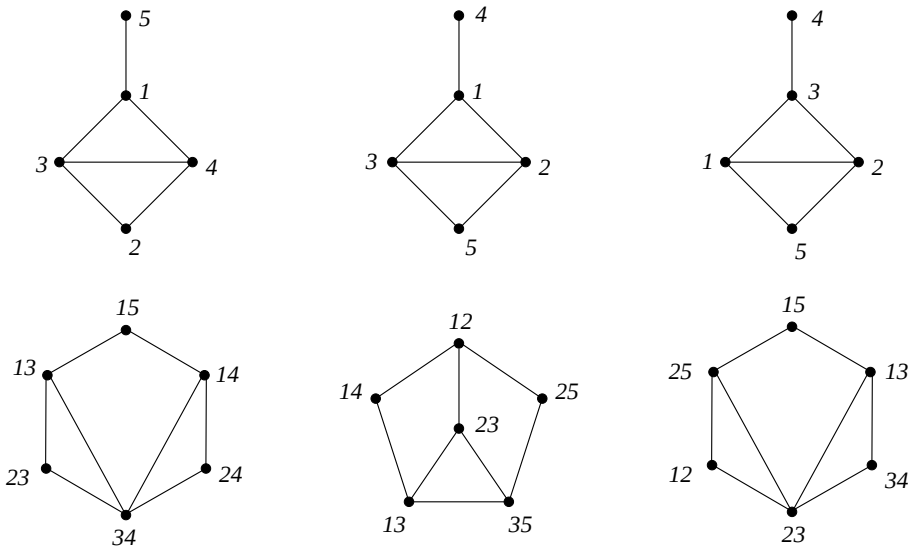

Figure 1: The same graph G with three different orders $\mathcal{O}_1$, $\mathcal{O}_2$ and $\mathcal{O}_3$ on the nodes (first line) and the corresponding dependent edge graphs $D(G, \mathcal{O}_1)$, $D(G, \mathcal{O}_2)$ and $D(G, \mathcal{O}_3)$ (second line)

The following lemma will be used in the proof of Theorem 3.4.

Lemma 3.3 *A set $F \subseteq E$ is a canonical clique-connecting stellar forest of G if and only if F is a stable set in the dependent edge graph $D(G, \mathcal{O})$ of G.*

Proof. Let $F \subseteq E$ be a canonical clique-connecting stellar forest of G. Since any two edges of F form an independent pair, F is a stable set in $D(G, \mathcal{O})$. Now let $F \subseteq E$ be a stable set in $D(G, \mathcal{O})$. Suppose that there exist three edges $e_1 = uv$, $e_2 = vw$ and $e_3 = wx$ of F where u, v, w are distinct nodes of G. (It may be that $u = x$ but, since G is a simple graph, x is distinct form v and w.) Since $\{e_1, e_2\}$ is an independent pair, then, by condition (i), $\mathcal{O}(v) < \min\{\mathcal{O}(u), \mathcal{O}(w)\}$. Similarly, $\mathcal{O}(w) < \min\{\mathcal{O}(v), \mathcal{O}(x)\}$, and hence $\mathcal{O}(v) < \mathcal{O}(w) < \mathcal{O}(v)$, a contradiction. In consequence, the graph (V, F) has no path of length 3 and no cycle of length 3. Hence F is a stellar forest of G. Then, by condition (ii), F is clique-connecting. Finally, F is a canonical clique-connecting stellar forest of G. □

We state now the main result of this paper.

Theorem 3.4 *$\overline{\chi}(G) + \alpha(D(G,\mathcal{O})) = n$, for every graph G and every bijection $\mathcal{O} : V \leftrightarrow \{1,\ldots,n\}$.*

Proof. If F is a maximum stable set of the dependent edge graph $D(G,\mathcal{O})$ of G, then, by Lemma 3.3, F is a maximum canonical clique-connecting stellar forest in the graph G. Hence F is a maximum clique-connecting forest of G and then, by Lemma 2.1, $\overline{\chi}(G) + |F| = n$. □

In the following we make some basic remarks about dependent edge graphs. Proposition 3.5 shows that the number of edges of $D(G,\mathcal{O})$ is the same for every bijection $\mathcal{O}$.

Proposition 3.5 *The number of edges of $D(G,\mathcal{O})$ is equal to the number of edges of the line graph $L(G)$ of G minus the number of triangles in G.*

Proof. For every triangle of G there is exactly one edge of $L(G)$ that is not and edge of $D(G,\mathcal{O})$. Conversely, if ef is an edge of $L(G)$ that is not an edge of $D(G,\mathcal{O})$, then, in G, the nodes of e, f induce one triangle. □

Clearly, $D(G,\mathcal{O}) = L(G)$ if and only if G is triangle-free. The following proposition identifies the dependent edge graphs with less than four nodes.

Proposition 3.6 *The graphs with less than four nodes that are not dependent edge graphs are the claw graph and $K_4 \setminus e$.*

Proof. It is easily checked that, except the claw graph and $K_4 \setminus e$, the graphs with less than four nodes are dependent edge graphs. Suppose that the claw graph is the dependent edge graph of G with $\mathcal{O}$. Since the claw graph is not a line graph, G has a triangle. If G has four nodes, by Theorem 3.4, $\overline{\chi}(G) = 1$; this is impossible since G has four edges. Thus G has an isolated edge, and hence its dependent edge graph has an isolated node; contradiction. Now suppose that $K_4 \setminus e = D(G,\mathcal{O})$. Since there are two cliques $\{e_1, e_2, e_3\}$ and $\{e_2, e_3, e_4\}$ in $K_4 \setminus e$, then $\{e_1, e_2, e_3\}$ and $\{e_2, e_3, e_4\}$ are two stars of G, and hence G is a star graph. Then $D(G,\mathcal{O})$ is isomorphic to K_4; contradiction. □

It is well-known that there is a forbidden induced subgraphs characterization of line graphs, Proposition 3.7 below shows that there is no such charaterization for dependent edge graphs. In consequence, the class of dependent edge graphs is not closed under taking (induced) subgraphs.

Proposition 3.7 *For every graph H there is a graph G and an order $\mathcal{O}$ such that H is an induced subgraph of $D(G, \mathcal{O})$.*

Proof. Let $\overline{H}$ be the complementary graph of H and let G be the graph obtained by adding a new node u to $\overline{H}$ and a new edge uv for every node v of $\overline{H}$. Let $\mathcal{O}$ be an order on the nodes of G such that $\mathcal{O}(u) < \mathcal{O}(v)$ for every $v \neq u$. The subgraph of $D(G, \mathcal{O})$ induced by the nodes corresponding to the edges incident to u is isomorphic to H. □

Now we study the maximum stable set problem in dependent edge graphs. The stable set polytope $Stab(H)$ of a graph H is the convex hull of the incidence vectors of the stable sets of H. In view of Theorem 3.4, the minimum clique partition problem is equivalent to determining the maximum value of $\mathbf{1}^\mathsf{T} x$ with x in the stable set polytope of $D(G, \mathcal{O})$. It can be formulated by the following 0-1 linear program (IP_{stab}):

$$\begin{array}{ll} \max \mathbf{1}^\mathsf{T} x & \\ \text{s.t.} & \\ \qquad x_e & \in \{0,1\} \quad \text{for } e \in E, \\ x_e + x_f & \leq 1 \qquad \text{for every edge } ef \text{ of } D(G, \mathcal{O}). \end{array}$$

From now on, we may write D instead of $D(G, \mathcal{O})$ and L instead of $L(G)$.

Clearly, $Match(G) = Stab(L) \subseteq Stab(D) \subseteq CCF(G)$, and $Stab(D) = CCF(G)$ if and only if G is triangle-free. The rank inequalities of $Stab(D)$ are: $x(E') \leq \alpha(D[E'])$ for $E' \subseteq E$. Those inequalities give a description of $Stab(D)$ when G is triangle-free. A maximum (weighted) stable set in a claw-free graph can be found in polynomial time [11, 17] (but no polyhedral description is known for this class). In consequence, if we have an order $\mathcal{O}$ such that $D(G, \mathcal{O})$ is claw-free, then $\overline{\chi}(G)$ can be determined in polynomial time. But we can note that the class of non-triangle-free graphs whose dependent edge graphs are claw-free for any order is insignificant. Moreover, (as we will see in the next paragraph) if $\omega(G) \geq 5$, then D is not claw-free. A clique inequality of $Stab(D)$ is a rank inequality associated with a maximal clique E' of D. As for $CCF(G)$, the separation problem for the clique inequalities of $Stab(D)$ is NP-hard. By [2], $Stab(D)$ is described by the clique inequalities if (and only if) D is *perfect* (a graph G is said to be perfect if $\omega(H) = \chi(H)$ for each induced subgraph H of G, see [18]). The

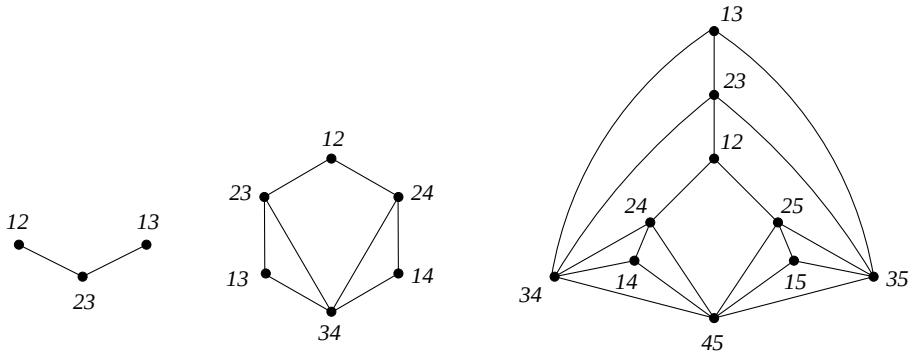

Figure 2: D_3, D_4 and D_5

strong perfect graph theorem states that a graph is perfect if and only if it has no odd hole and no odd anti-hole (see [1, 18]). By [8], a maximum stable set and a minimum clique partition in a perfect graph can be found in polynomial time. But as for line graphs, if G is not perfect, then D is not perfect. (Indeed, the edge-set of an odd hole in G corresponds to the node-set of an odd hole in D, and, if $v_1, v_2, \ldots, v_{2k+1}$ is the node-sequence of an anti-hole in G, then the sequence of edges $v_1v_{k+2}, v_{k+2}v_2, v_2v_{k+3}, \ldots, v_kv_{2k+1}, v_{2k+1}v_{k+1}, v_{k+1}v_1$ of G corresponds to the node-sequence of an odd hole in D.) Notice that the converse is not true since the graph G of Figure 1 is perfect altough $D(G, \mathcal{O}_2)$ is non-perfect. A natural question is now whether perfection can be transmitted from G to D for some order?

The dependent edge graph $D_n = D(K_n, \mathcal{O})$, where K_n is the complete graph on n nodes, is the same for every $\mathcal{O}$ (up to isomorphism). Figure 2 depicts D_3, D_4 and D_5. Note that D_4 is isomorphic to edge dependent graphs $D(G, \mathcal{O}_1)$ and $D(G, \mathcal{O}_3)$ of Figure 1 despite the graph G of Figure 1 is not isomorphic to K_4. Note also that D_5 is not claw-free ($\{12, 23, 24, 25\}$ induces a claw). Surprisingly, D_5 is non-perfect ($\{12, 25, 45, 34, 23\}$ induces an odd cycle); this means that there is no hope of conserving graph perfection.

To close this section, we give a class of valid inequalities that is usefull in practice to solve (IP_{stab}). Given $v \in V$, δ_v^- is a clique of D. Moreover, ef is an edge of D for every $e \in \delta_v^-$ and $f \in \delta_v^+$. Set $D_v^+ := D[\delta_v^+]$. Clearly, $x(\delta_v^+) \leq \alpha(D_v^+)$ is valid for $Stab(D)$; that (rank) inequality can be lifted to obtain the following inequality:

$$x(\delta_v^+) + \alpha(D_v^+) \times x(\delta_v^-) \leq \alpha(D_v^+) \tag{2}$$

which is valid for $Stab(D)$. By [13], the lifted rank inequality (2) is facet for $Stab(D)$ if and only if the rank inequality is facet for $Stab(D_v^+)$. A sufficient condition for when $x(\delta_v^+) \leq \alpha(D_v^+)$ is facet for $Stab(D_v^+)$ can be found in [2].

3.2 Linear forests

A forest F is said to be *linear* if each tree of F is a path. A linear forest F of G is said to be *well-ordered (w.r.t. $\mathcal{O}$)* if, for each path P of F, the path P is well-ordered, that is P has a sequence of nodes $(v_1, v_2, \ldots, v_k)$ such that $\mathcal{O}(v_1) < \mathcal{O}(v_2) < \ldots < \mathcal{O}(v_k)$. The following remark holds.

Remark 3.8 *Given $\mathcal{O} : V \leftrightarrow \{1, \ldots, n\}$. There is a one-to-one correspondence between the clique partitions of G and the clique-connecting well-ordered linear forests of G.*

Note that being a clique-connecting well-ordered linear forest of G is an hereditary property. A pair of adjacent edges of G, say uv and vw, is said to be a *well-ordered pair (w.r.t. $\mathcal{O}$)* if the following condition holds:

$(*)$ $\min\{\mathcal{O}(u), \mathcal{O}(w)\} < \mathcal{O}(v) < \max\{\mathcal{O}(u), \mathcal{O}(w)\}$.

An *untidy pair* is a pair of adjacent edges that is not well-ordered. A path of G is said to be a *bad path (w.r.t. $\mathcal{O}$)* if it is an untidy pair or if it is a well-ordered path $P = v_o v_1, \ldots, v_{k-1} v_k$ with $v_o v_k \notin E$.

Proposition 3.9 *A subset $F \subseteq E$ is a clique-connecting well-ordered linear forest of G if and only if F contains no bad path.*

Proof. Necessity being trivial we show sufficiency. Assume that F has no bad path. Suppose that F has a cycle C. Let v be the node of C with the lowest order and let u, w be the two neighboors of v on the cycle C. Then $\mathcal{O}(v) < \min\{\mathcal{O}(u), \mathcal{O}(w)\}$ which contradicts $(*)$. Thus F has no cycle and hence F is a forest. Suppose that there exist four distinct nodes u, v, w, x such that $uv, vw, vx \in F$. We can assume w.l.o.g. that $\mathcal{O}(u) < \mathcal{O}(w) < \mathcal{O}(x)$. Since $\{uv, vw\}$ is a well-ordered pair, by $(*)$, $\mathcal{O}(u) < \mathcal{O}(v) < \mathcal{O}(w)$. Similarly, $\mathcal{O}(w) < \mathcal{O}(v) < \mathcal{O}(x)$, and hence $\mathcal{O}(v) < \mathcal{O}(w) < \mathcal{O}(v)$; this is impossible. Thus F is a linear forest. Then, F is a well-ordered linear forest. Hence every path $P \subseteq F$ links two adjacent nodes. Finally, F is a clique-connecting well-ordered linear forest of G. □

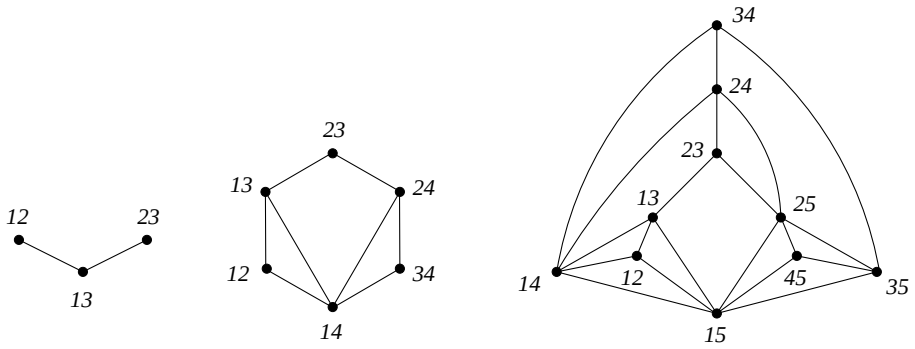

Figure 3: B_3, B_4 and B_5

Let $CCWF(G, \mathcal{O})$ be the convex hull of the incidence vectors of the clique-connecting well-ordered linear forests of G. For convenience, set $CCWF(G) := CCWF(G, \mathcal{O})$. Clearly, $Stab(L) \subseteq CCWF(G) \subseteq CCF(G)$, and we can replace inclusions by equalities when G is triangle-free. It is also clear that the minimum clique partition problem is equivalent to determining $\max \mathbf{1}^{\mathsf{T}} x$ with $x \in CCWF(G)$; it can be formulated by the following 0-1 linear program (IP_{path}):

$$\begin{aligned} &\max \mathbf{1}^{\mathsf{T}} x \\ &\text{s.t.} \\ &\qquad x_e \;\; \in \{0,1\} \qquad \text{for } e \in E, \\ &\qquad x(P) \;\; \leq |P| - 1 \quad \text{for every bad path } P. \end{aligned}$$

If we replace the integrity constraints $x_e \in \{0,1\}$ by the trivial constraints $0 \leq x_e \leq 1$ for $e \in E$ we obtain a linear program (LP_{path}).

Proposition 3.10 *The linear program (LP_{path}) can be solved in polynomial time.*

Proof. As in the proof of Proposition 2.3, we only need to solve the following separation problem in polynomial time: given $x \in \mathbb{Q}_+^E$, decide if $x(P) \leq |P| - 1$ for every bad path P, and if not, find a bad path P such that $x(P) > |P| - 1$.
Since the number of untidy pairs is polynomial, we can focus on the inequalities where P is a well-ordered path. Define $y_e := 1 - x_e$ for $e \in E$.

So $x(P) \leq |P| - 1$ is equivalent to $y(P) \geq 1$. Let D be the directed acyclic graph obtained from G by orienting each edge of G from the node with the lower order to the node with the greater order. In D, set the length of the arc corresponding to the edge e of G to $y(e)$, for every $e \in E$. By finding the shortest directed path of D from u to v for every (u, v) with $uv \notin E$ and $\mathcal{O}(u) < \mathcal{O}(v)$, a bad path P of G with P minimizing $y(P)$ can be found in polynomial time. (A shortest path in a directed acyclic graph can be found in linear time [12].) If $y(P) \geq 1$, then x satisfies the inequalities of (LP_{path}), otherwise P gives a violated inequality. □

Inequalities of (LP_{path}) associated with non-minimal bad paths are redundant, where a bad path is said to be minimal if it contains no bad path. The following is not hard to see: P is a minimal bad path if and only if P is an untidy pair or a well-ordered bad path with all the possible chords (except $v_o v_k$). If G is $K_4 \setminus e$-free, then $CCWF(G)$ is equal to the stable set polytope of a subgraph of L since the length of every minimal bad path is equal to 2. (Necessity is not true, since the graph G of Figure 1 is not $K_4 \setminus e$-free and each minimal bad path w.r.t. $\mathcal{O}_1$ for instance has length 2.)

To close this section remark that, if each component of G is a complete graph, then $CCWF(G)$ is described by $x_e \geq 0$ for $e \in E$, $x(\delta_v^-) \leq 1$ and $x(\delta_v^+) \leq 1$ for $v \in V$. To see this, first remark that every bad path is now an undity pair, and let $I(G, \mathcal{O})$ be the graph with node-set the edge-set of G and edge-set the set of undity pairs. Set $I := I(G, \mathcal{O})$ so $CCWF(G) = Stab(I)$. Graphs B_2, B_3 and B_4 are depicted in Figure 3, where $B_n = I(K_n)$ (up to isomorphism B_n does not dependent on $\mathcal{O}$). Notice now that I is the line graph of a bipartite graph. Indeed, I is the line graph of the bipartite graph $\tilde{G} = (V^-, V^+; \tilde{E})$ where V^- and V^+ are two copies of V and two nodes $u \in V^-$ and $v \in V^+$ are linked by an edge $uv \in \tilde{E}$ if $uv \in E$ and $\mathcal{O}(u) < \mathcal{O}(v)$. Hence I is perfect (see [18]). Since $CCWF(G) = Stab(I)$ and I is perfect, by [2], $CCWF(G)$ is decribed by the nonnegativity and the clique inequalities of $Stab(I)$. The result follows from the fact that the maximal cliques of I are the sets δ_v^- and δ_v^+ for each $v \in V$.

$G_{n,p}$	$\overline{m}$	CPU	Nodes	ω	LP	χ	Heu
$G_{60,1}$	1605	10	0	3	4	4	4
$G_{70,1}$	2176	125	10	3	4	4	5
$G_{80,1}$	2833	3035	397	4	4	4	5
$G_{50,2}$	1000	30	46	4	4	5	5
$G_{60,2}$	1421	38641	16452	4	5	6	6
$G_{60,3}$	1253	790	420	7	7	7	8
$G_{70,3}$	1721	8246	2821	8	8	8	10
$G_{50,4}$	751	161	450	7	7	8	8
$G_{60,4}$	1076	9718	24531	6	8	9	10
$G_{50,5}$	632	8	0	7	9	9	9
$G_{60,5}$	886	1397	7480	8	10	11	11
$G_{70,5}$	1211	67034	148573	8	10	12	13
$G_{70,6}$	1023	116	134	9	12	13	15
$G_{75,6}$	1097	2501	6920	11	14	15	17
$G_{80,7}$	894	322	3407	15	19	20	20
$G_{85,7}$	1036	133	572	13	19	20	21
$G_{90,7}$	1219	3877	16637	14	19	20	22

Table 1: Random graphs

4 Experimentations

In our computational experiments we consider the graph coloring problem. Given a graph G, we have solved the integer linear programs (IP_{stab}), (IP_{path}) and (IP_{forest}) associated to the complementary graph $\overline{G}$ of G.

The code was implemented in C++ using Cplex 9.0 MIP solver. We have performed the experiments on a Bi-Intel(R) Xeon(TM) CPU 2.80GHz (with HyperThreading, RAM=3GiB).

For solving (IP_{path}) and (IP_{forest}), we have implemented a cutting-plane algorithm based on the separation algorithms described in previous sections. In our computational experiments, the quality of the relaxation and the computational running time have always been worse with (IP_{path}) and (IP_{forest}) than with (IP_{stab}). This is why we report the computational experiments with (IP_{stab}) only.

$G_{n,p}$	$\overline{m}$	CPU	Nodes	ω	LP	χ	Heu
$G_{100,8}$	973	16	96	21	27	28	28
$G_{105,8}$	1110	27	104	21	27	28	28
$G_{110,8}$	1221	448	2937	20	28	29	32
$G_{125,8}$	1616	51332	116566	20	30	31	32
$G_{150,9}$	1099	6	70	37	49	49	50
$G_{175,9}$	1560	95	1426	40	56	56	58
$G_{200,9}$	2003	328	2746	40	61	62	62
$G_{225,9}$	2496	624	2380	?	67	68	69
$G_{250,9}$	3113	42254	69763	?	72	73	74

Table 2: Random graphs with high density

Tables 1 and 2 show the results with random graphs $G_{n,p}$. The graph $G_{n,p}$ is formed by generating a graph on n nodes, where each edge occurs independently with probability $p/10$. Table 3 shows the result with DIMACS instances. These particular instances, introduced and described in [10], have been largely studied in the literature and thus constitute a good reference for comparisons.

The number of variables of the 0-1 linear program is equal to the number of edges of $\overline{G}$, which is contain in the column " $\overline{m}$ " in the tables showing the results. The CPU times, reported in seconds, are showed in the column " CPU ". The column " Nodes " contains the number of nodes in the enumeration tree (with the convention that Nodes=0 at the root node). The column " ω " contains the clique number of the graph (sometimes it was not possible to obtain that parameter even after hours of computations with Cplex). The column " LP " contains the rounding value of the relaxation at the root node (after the use of cuts, before the first branching). The column " χ " contains the optimum value. Finally, the column " Heu " contains the best integer value given by Cplex's heuristic. A time limit has been set to 86400 secondes (when that limit has been exeeded there is a " - " in columns CPU and Nodes).

name	n	$\overline{m}$	CPU	Nodes	ω	LP	χ	Heu
Myciel4	23	182	1	558	2	4	5	5
Myciel5	47	845	39970	1370874	2	4	6	6
Myciel6	95	3710	-	-	2	5	7	7
jean	80	2906	1	0	10	10	10	10
huck	74	2400	1	0	11	11	11	11
david	87	3335	11	0	11	11	11	11
anna	138	8960	323	0	11	11	11	11
1-FullIns-3	30	335	1	4	3	3	4	4
1-FullIns-4	93	3685	37822	9876	3	4	5	5
DSJC125-1	125	7014	-	-	4	4	5	7
DSJC125-5	125	3859	-	-	10	13	17	24
DSJC125-9	125	789	3	43	34	43	44	44
DSJC250-9	250	3228	17180	16982	?	71	72	72
DSJC500-9	500	12313	-	-	?	122	?	137
queen6-6	36	340	7	120	6	6	7	7
queen7-7	49	700	1	0	7	7	7	7
queen8-8	64	1288	8459	19406	8	8	9	10
queen9-9	81	2184	-	-	9	9	10	11

Table 3: DIMACS benchmark

5 Conclusion

We have developped a new approach of the minimum clique partition problem, and so also of the minimum graph coloring problem. It appears that these problems are very strongly related to the maximum stable set problem, since the identity $\overline{\chi}(G)+\alpha(L) = n$, which holds for every triangle-free graph G and its line graph L, has been generalized in this paper. The generalized identity is: $\overline{\chi}(G)+\alpha(D) = n$ for every G, where D belongs to a class of partial subgraphs of the line graph of G. Some basic remarks about that class has been done but no characterization is given. Our computational experiments show that computing $\alpha(D)$ using Branch&Cut is an efficient method for determining $\overline{\chi}(G)$ if G is sparse but our results compare badly to [10, 7] for middle density graphs and badly to [10, 3, 7] for dense graphs. Since all the knowledge about the stable set problem has not been used in our

experiments, it is not clear yet whether or not this will lead to an efficient method for general graphs. A remaining question is whether $n - \vartheta'(D)$ is an interesting lower bound for $\overline{\chi}(G)$, where ϑ' is the strengthening of Lovász's theta number (see [6])?

Acknowledgement

The author would like to thank Vincent Jost for pointing out the utility of orders.

References

[1] M. Chudnovsky, N. Robertson, P. Seymour, R. Thomas, The Strong Perfect Graph Theorem, to appear in the Annals of Mathematics.

[2] V. Chvátal, On certain polytopes associated with graphs, *Journal of Comb. Theory, Series B 18* (1975) 138-154.

[3] P. Coll, J. Marenco, I. Méndez Díaz, and P. Zabala, Facets of the graph coloring polytope, *Annals of Operations Research 116(1-4)* (2002) 79-90.

[4] J. Edmonds, Maximum matching and a polyhedron with 0,1-vertices, *Journal of Resarch National Bureau of Standards Section B 69* (1965) pp. 67-72.

[5] J. Edmonds, Matroids and the greedy algorithm, *Math. Program. 1* (1971) pp. 127-136.

[6] N. Gvozdenovic, M. Laurent, Approximating the Chromatic Number of a Graph by Semidefinite Programming, *preprint.*

[7] P. Hansen, M. Labbé, D. Schindl, Set covering and packing formulations of graph coloring: algorithms and first polyhedral results, Cahier du GERAD G-2005-76, Montreal, september 2005.

[8] M. Grötschel, L. Lovàsz and A. Schrijver, The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica 1(2)* (1981) pp. 169-197.

[9] V. Jost, private communication, *Aussois meeting* (2006).

[10] A. Mehrotra and M.A. Trick, A column generation approach for graph coloring, *INFORMS, Journal on Computing, 8(4)* (1996) pp. 344-354.

[11] G.J. Minty, On maximal independent sets of vertices in claw-free graphs, *Journal of Comb. Theory, Series B 23* (1980) pp. 284-304.

[12] J. Morávek, A note upon minimal path problem, *Journal of Mathematical Analysis and Applications 30* (1970) pp 702-717.

[13] M.W. Padberg, On the facial structure of set packing polyhedra, *Math. Program. 5* (1973) 199-215.

[14] M.W. Padberg and L.A. Wolsey, Fractional covers for forests and matchings, *Math. Program. 29* (1984) pp. 1-14.

[15] J.-C. Picard and M. Queyranne, Selected applications of minimum cuts in networks, *INFOR Canadian Journal of Operational Research and Information Processing 20* (1982) pp. 394-422.

[16] J.M. W. Rhys, A selection problem of shared fixed costs and network flows, *Management Science 17* (1970) pp. 200-207.

[17] N. Sbihi, Algorithme de recherche dans un graphe sans étoile, *Disc. Math. 29* (1980) pp. 53-76.

[18] A. Schrijver, *Combinatorial Optimization*, Springer-Verlag Berlin Heidelberg (2003).