Steffen Rebennack
Stable Set Problem: Branch & Cut Algorithms
in *Encyclopedia of Optimization*, Second Edition,
C.A. Floudas and P.M. Pardalos (Eds.),
Springer, pp. 3676–3688, 2008

Stable Set Problem: Branch & Cut Algorithms

Steffen Rebennack
University of Florida
Center of Applied Optimization
Gainesville, FL 32611, USA
e-mail: steffen@ufl.edu

July 2007

Key words: Stable Set, Independent Set, Maximum Clique, Vertex Packing, Branch & Cut, Separation

In this article, we provide an overview on how the maximum weighted stable set problem can be solved exactly with Branch & Cut techniques. In addition, we provide selected references to other exact methods. We start with a brief introduction of the stable set problem and a few basic definitions but assuming that the reader is already familiar with the basic concepts. The main stress of this article lies in the review of polyhedral results for the stable set polytope in Section 2.1 and the discussion of separation procedures, Section 2.2. An efficient Branch & Cut algorithm needs, in addition to strong separation routines, also a good branching strategy. This is discussed in Section 2.3. At the end, some implementation aspects are considered.

1 Introduction

Let $G = (V, E)$ be an undirected graph consisting of a nonempty finite set V, the node set; and a finite set E, the edge set, of unordered pairs of distinct elements of V. A stable set of graph G is defined as a set of nodes S with the property that the nodes of S are pairwise non adjacent; two nodes are called adjacent if there is an edge in E connecting them. In the literature, stable set is also called independent set, vertex packing, co-clique or anticlique. If each node v_i of a graph G is assigned a weight c_i, then the graph is called weighted. In this case, the maximum weighted stable set problem looks for a stable set S which maximizes the sum of the weights corresponding to the nodes in S, $\sum_{v_i \in S} c_i$. In the case when G is not weighted, or all $c_i = 1$, we are interested in a stable set with the maximum number of nodes, which is called maximum cardinality stable set. The size

of a maximum cardinality stable set is called the stability number of graph G and is denoted by $\alpha(G)$. Throughout this article, references to the maximum stable set problem, or just stable set problem, consider the weighted case unless otherwise noted.

According to its definition, the stable set problem has many applications in various fields, [BBPP99]. Especially when "conflicts" between some objects occur, it is an indicator that the stable set problem is applicable. Next to the Traveling Salesman Problem, it is one of the most important combinatorial optimization problems. It is well known that it is $\mathcal{NP}$-*hard* to determine a maximum stable set in an arbitrary graph, [GJ79]. This holds true for the cardinality case. Furthermore, it is also hard to approximate the stable set number: It can be shown that for any fixed $\varepsilon > 0$ there is no polynomial time algorithm for approximating the stability number within a factor of $|V|^{\varepsilon}$, under the assumption that $\mathcal{P} \neq \mathcal{NP}$, [AS92, FMK95].

Let us briefly and informally introduce some polyhedral terminologies. In our case it is sufficient to define a polyhedron as the solution set of a system of linear inequalities. If the solution set is bounded, it is called a polytope. Graphically speaking, a polytope in $\mathbb{R}^n$ is of full-dimension if it contains an n-dimensional sphere completely; in 2-dimensions it is therefore forbidden that the polytope is empty, one point or a line segment. A linear inequality $\beta^\top x \leq b_0$ is valid with respect to a polyhedron P, if P is a subset of $\{x \,|\, \beta^\top x \leq b_0\}$. We call a set $F \subseteq P$ a facet of P if there is a valid inequality $\beta^\top x \leq b_0$ for P such that $F = \{x \in P \,|\, \beta^\top x = b_0\}$, and the inequality is not dominated by any other valid inequality. This inequality is called a facet-defining inequality for P. In the case when v is a point in the polyhedron P and $F = \{v\}$, we call v a vertex of polyhedron P. Now, let P be a polytope and x^* be a given point. The task to decide if this point lies in P or if not to find a valid inequality $\beta^\top x \leq b_0$ for P which is violated by x^*, is called the separation problem for polytope P. The convex hull of points $y_1, ..., y_n \in \mathbb{R}^d$ is the set of points x satisfying $x = \sum_{i=1}^n \lambda_i y_i$ with $\sum_{i=1}^n \lambda_i = 1$ and $\lambda_i \geq 0 \ \forall i$. It is denoted by $conv\{y_1, \ldots, y_n\}$. More precise formulations can be found for instance in [GLS88, Brø83, Zie95].

We introduce now, in addition to the ones above, several graph theoretic definitions and notations needed throughout this article. A node v is incident to an edge e, if $e = uv$. The two nodes incident to an edge are its endnodes. A node is isolated if it has no neighbor in the graph, which means that it is not an endnode of any edge of the graph. The neighborhood of a node v is the collection of all its neighbor ans is abbreviated with $\Gamma(v)$. If a graph has no isolated nodes, it is called connected. A graph is said to be complete if it contains an edge connecting each pair of its nodes. A clique is the node set of a complete subgraph. If a clique has three nodes it is also called a triangle. The cardinality of a graph G is abbreviated by $|G|$ and denotes the number of nodes in the graph. The complement graph $\overline{G}$ of the graph G has the same node set as G and contains an edge between two nodes, iff no edge is contained in G. We call a graph G bipartite if its node set V can be partitioned into two disjoint sets V_1, V_2 with $V = V_1 \cup V_2$ such that neither two nodes of set V_1 nor two nodes of set V_2 are neighbors. We call $H = (W, F)$ a subgraph of G, and write $H \subseteq G$, when $W \subseteq V$ and $F \subseteq E$ is the set of edges of graph G with both endnodes in W. Two graphs $G = (V, E)$ and $H = (W, F)$ are called isomorphic, if there is a bijection $\phi \colon V \to W$ such that

$uv \in E \Leftrightarrow \phi(u)\phi(v) \in F$. A matching is a collection of pairwise disjoint edges. If in a matching M every node of G is incident with exactly one edge M, then it is a perfect matching. More about graph theory can be found in [Die00, Wes00].

We do not describe the Branch & Cut algorithm in general here, as we assume the reader is familiar with its basic ideas. For more information we refer to [GJR84, PR87, Wol98, KW97].

Before we discuss some aspects of a Branch & Cut algorithm to solve the maximum stable set problem, we provide a list of some other exact solution methods. Clearly, this list does not claim to be complete. A more detailed list of exact methods can be found in [BBPP99, But03]. In this context, we want to mention that the stable set problem is equivalent to the maximum clique problem in the complement graph. Hence, each method solving the maximum clique problem can also be used to solve the stable set problem. For polynomial time algorithms for some special classes of graphs, see [MNKF90, PY81, Mos97, BCN87, BY89, Ale03, Ola89, BK97, BDLS05, CT05, LKSN06] and Section 2.1. Algorithms finding all maximum stable sets in a graph are considered in [HR57, BK73, TTT88, LT81, CN85]. In the literature, many variants of Branch & Bound algorithms have been discussed, [Bar00, WH06, PR92, Sew98, Woo97, Öst02, Fah02, MS05, BT90]. Other methods using, for instance, continuous formulations, column generation or constraint programming can be found in [CP90, MS99, BMZ02, PP90, BLM97, WWWH05, BX96, VA99, Rég03, BS99].

Benchmark instances are provided by the second DIMACS Challenge, [dim], from 1992/1993 and by the BHOS library from 2000, [BHO]. Note that some of these stable set instances are still unsolved. A test case generator was introduced by Hasselberg et al., [HPV93].

2 Method

Let us now formulate the maximum stable set problem as a linear program. Therefore, one choice could be to introduce variables x_i for each node $v_i \in V$, which have value one, if node v_i is in a stable set, say S, and otherwise zero. Such a vector is called an incidence vector. Obviously, for each edge, only one endnode can be in a stable set and hence, we get the so called edge-inequalities

$$x_i + x_j \leq 1 \qquad \forall\, ij \in E. \tag{1}$$

It is easy to see, that if vector x has a positive integer domain (or more precisely, binary domain), each vector satisfying inequalities (1) induces a stable set and vice versa. Hence, if c denotes the (positive) weight vector of the nodes, one gets the following integer program

$$\begin{array}{lll} \max\limits_{x} & c^\top x & \\ s.t. & x_i + x_j \leq 1 & \forall\, ij \in E \\ & x \in \{0,1\}^{|V|}, & \end{array} \tag{2}$$

which solves the maximum weighted stable set problem. We recognize, that this formulation has only $|E|$ constraints and $|V|$ variables. So the formulation is quite compact. Unfortunately, the binary constraints on x make it hard to solve this linear program. We will discuss some relaxations of this problem in the next section which is mainly based on [GLS88, GL06, NW88, BP76, Sch03].

2.1 Stable Set Polytope

The stable set polytope of graph $G = (V, E)$ is defined as the convex hull of the incidence vectors of all stable sets in G. It is denoted by

$$P_{STAB}(G) := conv\{\, \chi^S \,|\, S \subseteq V \text{ stable set}\},$$

where χ^S is the incidence vector of set S. From the integer program formulation (2) we see that $P_{STAB}(G)$ is a polyhedron. As it is bounded by the $|V|$-dimensional unit cube, it is indeed a polytope. The definition of a stable set implies that the unit vectors are always stable sets. Trivially, the zero vector is a stable set, the empty set, therefore, the stable set polytope is full-dimensional. This implies that all facets of $P_{STAB}(G)$ are inequalities, and hence, we do not have to consider equalities, [NW88, Reb06].

Let us now discuss some relaxations of the integer program formulation (2) which will also give us relaxations of the stable set polytope. The obvious idea is to relax the binary condition on x, and instead make them continuous which leads to

$$0 \leq x_i \leq 1 \qquad \forall v_i \in V. \tag{3}$$

The integer problem reduces to a linear program which can be solved in polynomial time. This relaxation leads to the so called stable set polytope relaxation

$$P_{RSTAB}(G) := \{x \in \mathbb{R}^n \,|\, x_i + x_j \leq 1,\, 0 \leq x \leq 1 \;\; \forall ij \in E\}. \tag{4}$$

From its construction, we get that $P_{STAB}(G) \subseteq P_{RSTAB}(G)$. For a complete graph with cardinality ≥ 3, the x vector with value $1/2$ in each component is a vertex of $P_{RSTAB}(G)$, but it cannot be contained in $P_{STAB}(G)$ as a maximum cardinality stable set in a complete graph has cardinality one. This example shows that the relaxation above is very weak. Note, the vector whose entries are all $1/2$ is always contained in $P_{RSTAB}(G)$ – independent of the structure of the graph G. The following corollary generalizes this observation. It was first indicated by Balinski [Bal70].

Corollary 2.1. *The vertices of $P_{RSTAB}(G)$ are $(0, \frac{1}{2}, 1)$-valued.*

We saw that for a complete graph G with cardinality ≥ 3, $P_{STAB}(G) \subset P_{RSTAB}(G)$. The next theorem states that this holds except for connected bipartite graphs. In this case the stable set polytope and the stable set polytope relaxation are equal, that is $P_{STAB}(G) = P_{RSTAB}(G)$.

Theorem 2.2. *[GLS88] The non-negativity inequalities, $x_i \geq 0 \;\forall v_i \in V$, together with the edge inequalities (1) are sufficient to describe $P_{STAB}(G)$, iff G is bipartite and has no isolated nodes.*

Theorem 2.2 has the following important implication: It states that the maximum stable set problem for bipartite graphs can be solved in polynomial time by solving the stable set problem over (4). As a consequence, a Branch & Cut algorithm using the stable set polytope relaxation (4), will terminate for bipartite graphs in the root node of the branching tree after solving one linear program. However, we already have indicated that the stable set polytope relaxation is very week and hence not a good choice in a Branch & Cut framework for general graphs. Obviously, it can be checked in linear time whether a graph is bipartite or not. Exact polynomial time algorithms for bipartite graphs can be found in [Law01, GLS88].

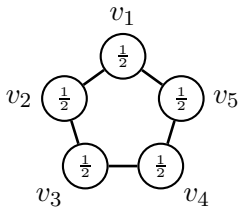


Figure 1: Odd cycle with five nodes

As the restriction to bipartite graphs is very tough, we want to find some ways to strengthen the stable set polytope relaxation. One idea is to add additional valid inequalities to $P_{RSTAB}(G)$. Therefore, let us consider the Figure 1. It shows a graph with the five nodes $v_1, v_2, ..., v_5$. Such a graph is called odd-cycle. In general, whenever a (sub-)graph H has an odd number of nodes, say n, and there are n adjacent edges in the edge set such that each node is incident to exactly two nodes, then we call H an odd cycle. Notice that an odd cycle can have more than n edges. In this case, any additional edge is called chord. For the graph of the Figure 1, the stable set polytope relaxation allows the fractional solution with all entries of 1/2, as illustrated. This solution is optimal, and for the cardinality stable set problem, the objective function value is 5/2, which is greater than any optimal stable set which has cardinality two. Now, summing up all edge inequalities corresponding to the five edges in the graph, one gets

$$2x_1 + 2x_2 + 2x_3 + 2x_4 + 2x_5 \leq 5.$$

In this case each node is incident to exactly two edges, giving the coefficients for the variables, and there are five edge inequalities, providing the right-hand side. This inequality can be divided by two and as all variables are binary, one gets

$$x_1 + x_2 + x_3 + x_4 + x_5 \leq \left\lfloor \frac{5}{2} \right\rfloor.$$

This inequality can be generalized to the so called odd-cycle inequalities

$$\sum_{v_i \in \widetilde{V}} x_i \leq \frac{|C| - 1}{2} \qquad \text{for each odd cycle } C = (\widetilde{V}, \widetilde{E}) \subseteq G. \tag{5}$$

From the construction above, it is obvious that the odd-cycle inequalities are valid for the stable set polytope. If, in addition, the stable set polytope relaxation satisfies all odd-cycle inequalities of G, then it is called a cycle-constraint stable set polytope and is denoted by

$$P_{CSTAB}(G) := \left\{ x \in \mathbb{R}^{|V|} \,|\, x \text{ satisfies (1), (3) and (5)} \right\}.$$

If you consider, again, a complete graph, you see that there is no constant which relates the optimal solution over $P_{CSTAB}(G)$ to an optimal stable set. However, the graphs for which $P_{CSTAB}(G) = P_{STAB}(G)$ are called t-perfect[1]. Two examples for t-perfect graphs are bipartite graphs and almost bipartite[2] graphs. The problem of checking whether a graph is t-perfect or not belongs to $co\text{-}\mathcal{NP}$. The special structure of t-perfect graphs helps to find a maximum stable set. This is stated by the next corollary.

Corollary 2.3. *The maximum stable set problem in a t-perfect graph can be solved in polynomial time.*

We will see in Section 2.2 that the odd-cycle inequalities can be separated in polynomial time. This proves together with the Equivalence of Optimization and Separation the Corollary 2.3. Polynomial time algorithms for the class of t-perfect graphs can be found in [GLS81, GLS88].

We are mainly interested in facets of $P_{STAB}(G)$ since they are not dominated by any valid inequality of $P_{STAB}(G)$. The odd-cycle inequalities can only be facet-defining if their odd cycles are chordless. If there is a chord, one gets a smaller odd cycle and an even cycle. The smaller odd-cycle inequality together with the edge inequalities dominate the odd-cycle inequality which shows that it cannot be facet-defining. A graph which is a chordless cycle is called a hole. If an odd cycle induces an odd hole, the corresponding odd-cycle inequality is called an odd-hole inequality. Consider the following

Corollary 2.4. *[NT74] Let G be an odd hole. Then $\sum_{v_i \in V} x_i \leq \frac{|V|-1}{2}$ is facet-defining for $P_{STAB}(G)$.*

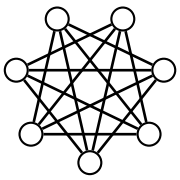


Figure 2: Odd antihole

A counterpart of the odd-cycle inequalities are the antihole inequalities. They are valid for antiholes, which is the complement graph of an odd hole with at least five nodes. From the Figure 2, we recognize that each stable set of an antihole with n nodes can contain at most two nodes as each node is adjacent to exactly $n-2$ nodes. Therefore, the following inequalities hold

$$\sum_{v_i \in \widetilde{V}} x_i \leq 2 \qquad \text{for each antihole } A = (\widetilde{V}, \widetilde{E}) \subseteq G. \tag{6}$$

[1] The "t" stands for "trou", which is the French word for "hole".
[2] A graph G is called almost bipartite if there is a node v such that graph G without v is bipartite.

Note that an antihole with 5 nodes is isomorphic[1] to an odd hole with 5 nodes. The separation problem for the antihole inequalities is not known whether it belongs to P or not.

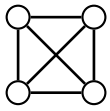


Figure 3: Complete graph with four nodes: clique

Another group of inequalities for the stable set polytope builds the clique inequalities. From the Figure 3 we get

$$\sum_{v_i \in Q} x_i \leq 1 \qquad \text{for each clique } Q. \tag{7}$$

In 1979, Padberg showed the following important

Theorem 2.5. *[Pad73] Let G be a graph with node set V and $Q \subseteq V$. Inequality (7) is valid for $P_{STAB}(G)$. An inequality $\sum_{v_j \in Q} x_j \leq 1$ is a facet of P_{STAB}, iff Q is a maximal clique in G.*

Theorem 2.5 shows that the edge inequalities (1) are only facet-defining for $P_{STAB}(G)$, if they build a maximal clique. Hence, they are dominated by the clique inequalities. We will use this observation later in Section 2.4. Note that for triangles, the clique inequality and the odd-cycle inequality are the same. We define the so called clique-constraint stable set polytope as

$$P_{QSTAB}(G) := \left\{ x \in \mathbb{R}^{|V|} \,|\, x \text{ satisfies (1), (3) and (7)} \right\}.$$

A graph G is called perfect[2] if $P_{QSTAB}(G) = P_{STAB}(G)$. The maximum clique and the stable set problem are very closely related. Therefore, it is not surprising that it is $\mathcal{NP}$-*hard* to separate the clique inequalities in an arbitrary graph. With this fact, it is quite remarkable that the following theorem holds.

Theorem 2.6. *[GLS88] The maximum stable set problem for perfect graphs can be solved in polynomial time.*

We do not go into the details of the proof here, but nevertheless, we give a rough explanation. It is possible to generalize the clique inequalities to a class of so called orthonormal representation constraints which are polynomially separable. The convex set of all vectors satisfying them and the non-negativity inequalities build the so called theta body, [Lov79, YFO06]. In the case for perfect graphs, this theta body is a polytope which equals $P_{STAB}(G)$. This implies Theorem 2.6. However, it is even $\mathcal{NP}$-*hard* to determine an optimal solution over $P_{QSTAB}(G)$, in general. More about perfect graphs can be found, for instance, in [RAR01, CRST04, CCL+05].

[1] Two graphs $G = (V, E)$ and $H = (W, F)$ are called isomorphic, if there is a bijection $\phi \colon V \to W$ such that $uv \in E \Leftrightarrow \phi(u)\phi(v) \in F$.

[2] Originally, in 1961 Berge called a graph perfect if the coloring number is equal to the clique number. This definition is equivalent to the polyhedral one given above.

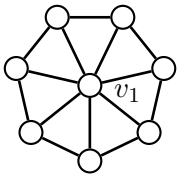
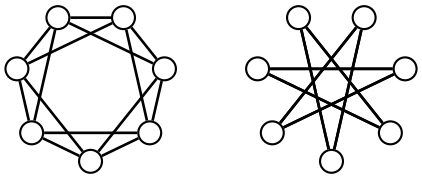


Figure 4: Odd wheel

If we consider the Figure 4, we recognize an odd hole with cardinality seven with the additional node v_1 which is adjacent to all other nodes. Such a graph is called wheel and node v_1 is its hub. We see that if node v_1 is contained in a stable set, no other node of the wheel can be contained in it. Hence, we get the following odd-wheel inequalities

$$\sum_{v_i \in \widetilde{V}} x_i + \tfrac{|C|-1}{2} x_u \leq \tfrac{|C|-1}{2} \qquad \text{for each odd wheel } C = (\widetilde{V}, \widetilde{E}) \subseteq G \text{ with hub } u. \tag{8}$$

From its construction, inequality (8) is valid for $P_{STAB}(G)$. It defines a facet if G is isomorphic to an odd wheel. Recognize that the wheel inequality dominates the odd-hole inequality. Generalizations of the wheel inequalities can be found in [CC97].



(a) (7,3)-web (b) (7,3)-antiweb

Figure 5: Web and antiweb

Another class of inequalities are the web and antiweb inequalities. Let p and q be integers satisfying $p > 2q + 1$ and $q > 1$. A graph G is called a web if G is isomorphic to the graph consisting of the nodes $\{v_1, \ldots, v_p\}$ with an edge $v_i v_j$, iff $|i - j| \equiv r < q$ modulo $(n - 2)$. A web is abbreviated with $W(p, q)$. A graph is called antiweb, denoted by $AW(p, q)$, iff $AW(p, q)$ is isomorphic to $\overline{W}(p, q)$. Examples can be seen in the Figures 5 (a) and 5 (b), respectively. The following inequalities

$$\sum_{v_i \in W(p,q)} x_i \leq q, \tag{9}$$

$$\sum_{v_i \in AW(p,q)} x_i \leq \left\lfloor \tfrac{p}{q} \right\rfloor \tag{10}$$

are called web inequalities and antiweb inequalities, respectively. Both types of inequalities are valid for $P_{STAB}(G)$. The web inequalities (9) define facets if p and q are relatively prime[1] and

[1]Two natural numbers are called relatively prime if their greatest common divisor is 1, or in formula: gcd(p,q)=1.

$G = W(p, q)$; while the antiweb inequalities (10) are facet-defining for $P_{STAB}(AW(p, q))$ if there is no $k \in \mathbb{N}$ with $p = k \cdot q$. More details can be found, for instance, in [Tro75, CdV02].

Now, consider the following class of inequalities for a graph $G = (V, E)$ and $W \subseteq V$

$$x(W) := \sum_{v_i \in W} x_i \leq \alpha(G[W]). \tag{11}$$

They are called rank inequalities. From their construction, inequalities (11) are valid for $P_{STAB}(G)$. The edge, odd-cycle, clique, antihole, web and antiweb inequalities belong to this class. Therefore, these inequalities are not facet-defining for $P_{STAB}(G)$ in general. For instance, an odd-wheel with 5 or more nodes does not lead to a rank inequality. Let us now have a closer look at the separation of the discussed inequalities.

2.2 Separation

In order to separate the odd-cycle inequalities (5) for a graph G and a vector x^*, one has to find an odd cycle for which x^* violates the corresponding inequality, or one has to prove that such cycles do not exist. In other words, we have to find a minimum-weight odd cycle in a graph, with an appropriate weighting function. If this cycle satisfies the corresponding inequality (5), it is proven that all odd-cycle inequalities are satisfied. Otherwise, one has found a maximal violated odd-cycle inequality. Therefore, we first recognize

Proposition 2.7. *A minimum-weight odd cycle in a graph G with edge weights can be computed in $O(|V| \cdot |E| \cdot \log(V))$.*

The idea is to construct an auxiliary bipartite graph H. This node set of H consists of two copies of the node set of the original graph G, and there is an edge between two nodes of H if the corresponding original nodes in G are adjacent. The edge weights are copied with the edges. From the construction of H, a minimum odd-cycle with respect to the edge weighting in G corresponds to a shortest path in H and vice versa. Hence, calculation of a shortest path for each node of the original graph G to its copy, gives a minimum weight odd cycle in G. Computing the shortest paths with Dijkstra's algorithm yields to the running time of Proposition 2.7. Now, define the following edge weighting of graph G depending on x^* as

$$c(v_i v_j) := \frac{1 - x_i^* - x_j^*}{2}. \tag{12}$$

With this weighting, it can be shown that an odd-cycle inequality in G is violated by vector x^*, if and only if

$$\exists C = (\widetilde{V}, \widetilde{E}) \subseteq G \text{ with } C \text{ odd cycle and } \sum_{v_i v_j \in \widetilde{E}} c(v_i v_j) < \frac{1}{2}.$$

This yields directly to the following theorem.

Theorem 2.8. *The separation problem for the class of odd-cycle inequalities can be solved in* $O(|V| \cdot |E| \cdot \log(|V|))$.

One has to mention that x^* has to satisfy all inequalities (3) before the odd-cycle inequalities can be separated with the above procedure. Otherwise, the weights defined in (12) can become negative, and the shortest path cannot be calculated with Dijkstra's algorithm anymore. More details and the description of the algorithms can be found in [CC97, Reb06, GS86]. It is interesting to realize that, in general, there are exponentially many odd cycles in a graph, but on the contrary, the separation of them is polynomial. We want to mention that GRÖTSCHEL and PULLEYBLANK introduced in 1981 another method to separate the odd-cycle inequalities with the use of perfect matchings[1] resulting in a running time of $O(|E|^4)$, [GP81].

Finding a maximum clique is $\mathcal{NP}$-*hard*, while computing an arbitrary maximal[2] clique as well as an arbitrary maximal stable set can be done in linear time. The separation problem for the clique inequalities asks to find a violated clique inequality in a particular graph G with a given linear program solution or to state that all clique inequalities are satisfied. This is equivalent to finding a maximum clique in G with the linear program solution as node-weighting c. Hence, the separation problem for the clique inequalities is $\mathcal{NP}$-*hard*. Computational tests show that the clique inequalities are very important for polyhedral approaches to the stable set problem, cf. [RS01]. As an exact separation cannot be considered, one idea could be to fix the size of the cliques to be separated, as then the problem becomes polynomially solvable. Another observation is that it is enough to consider maximal cliques. The reason is that the resulting clique inequality dominates all clique inequalities corresponding to contained cliques. Practically, it is more successful to separate over larger classes of inequalities containing the clique inequalities. We discuss that later for the case of rank inequalities. However, the best computational results, so far, are achieved with heuristic separation methods.

A separation algorithm for the wheel inequalities is given by CHENG and CUNNINGHAM, [CC95]. The appealing idea is to treat each node of the graph as a possible hub of an odd wheel and define appropriate weights for all nodes which are adjacent to this hub. Then, on the new graph, the odd-cycle inequalities can be separated. Hence, this algorithm results in a total running time of $O(|V|^2 \cdot |E| \cdot \log(|V|))$ or $O(|V|^4)$, dependent on the shortest path algorithm. For practical Branch & Cut algorithms, this complexity is already too high compared to the number of violated inequalities one can expect. Even more sophisticated are the separation routines for the web and antiweb inequalities. They are discussed by CHENG and VRIES, [CdV02]. Although they can also be separated in polynomial time, the complexity of the separation algorithms are again too large for practical use. In addition, these type of inequalities usually occur in graphs with high density[3], e.g. greater than 0.7, which makes its separation doubtful for graphs with low density.

[1] A matching is a collection of pairwise disjoint edges. If in a matching M every node of G is incident with exactly one edge M, then it is a perfect matching.

[2] We distinguish between maximum and maximal. Maximal means that there is no better solution containing the particular one. So maximal can be seen as locally best while maximum is global.

[3] The density of graph $G = (V, E)$ is defined as $\frac{2 \cdot |E|}{|V| \cdot (|V|-1)}$

Next to the special type of inequalities for the stable set polytope discussed in the Section 2.1, one can use general classes of inequalities. In this context, we want to mention two of such a type. The first are the so called mod-k cuts which belong to the CHVÁTAL-GOMORY cuts with rank 1. The appealing idea of the mod-k cuts is to find a multiplier μ, such that a particular inequality system $Ax \leq b$ multiplied with this vector μ can be strengthened by dividing it by a positive integer k. Therefore, let $k > 1$ be integral, and suppose that we have given a system of linear inequalities $Ax \leq b$ with integral coefficients. Let μ be a vector with positive integer entries and appropriate dimension such that

$$\begin{aligned} \mu^\top A &\equiv 0 \pmod{k}, \\ \mu^\top b &\equiv k-1 \pmod{k}. \end{aligned}$$

From this, one can obtain the mod-k inequalities

$$\frac{1}{k}\mu^\top Ax \leq \frac{1}{k}(\mu^\top b - (k-1)).$$

Examples for the case of $k = 2$ are the odd-cycle inequalities or the wheel inequalities. The separation of the mod-k inequalities and more details can be found in [CFL00, Reb06].

The second class of general inequalities that we discuss here in brief are the so called local cuts. The principal idea was introduced by APPLEGATE et al. in 2001 for the Traveling Salesman Problem, [ABCC01]. Suppose we have given the set of all m feasible solutions to the stable set problem, for instance of a subgraph with n nodes. The idea is to check if a given (solution) vector lies in its convex hull or if it lies outside, to compute a facet which separates this point from the convex hull. This can be achieved by solving a linear program with m variables and n constraints. Its optimal objective function value provides the information if the point lies inside the polytope, and the optimal dual variables give the coefficients of the separation inequality, called local cut. Obviously, this method has some weaknesses. One first has to find a 'good' subgraph and then enumerate all stable sets. In addition, the linear program to be solved can be large, as the number of stable sets in a graph can be exponential. Nevertheless, the resulting cuts can be quite strong, especially if all other separation procedures do not bring new cuts. More details and computational results can be found in [Reb06].

At the end of this subsection, we introduce the idea of separating rank inequalities. We do not go into full detail here but instead focus on the discussion of the principle ideas of the beautiful results of MANNINO and SASSANO, [MS96]. The appealing idea is to reduce the size of the graph and to make it denser at the same time. In general, any node of the graph can be selected and its two endnodes will be removed from the graph. In addition, new so called false edges are added to the graph and some other nodes may also be removed. Therefore, this procedure is called edge projection. Now, after a few iterations of this procedure, when the graph is small enough, one can separate any type of rank inequality, for instance, the clique inequalities or the odd-cycle inequalities. If a violated inequality has been found, it must be projected to be valid for the

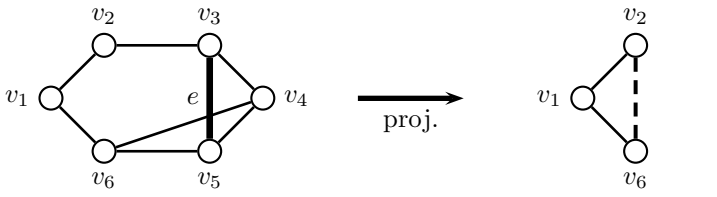
polytope of the original graph. This process is called anti-projection. We have to mention that it is possible that a projected inequality is no longer violated by a solution vector, even if it was in the projected graph. The edge-projection and the anti-projection can be done in linear time which makes this method very fast. Let us now consider a small example indicated in the Figure 6. In part (a) you see a small graph with six nodes. It is an odd hole with the additional node v_4. If we select edge $e = v_3v_5$ to project on, then in this case, nodes v_3, v_5 and, in addition[1], v_4 are removed from the graph; as well as all edges incident with any of these nodes. The false edge[2] v_2v_6 is added, and one gets the triangle shown in part (b) of the Figure 6. We recognize that the resulting graph is smaller and, indeed, more dense. We realize that the triangle inequality

$$x_1 + x_2 + x_6 \leq 1$$

is valid for $P_{STAB}(\overline{G})$ but not for $P_{STAB}(G)$. In most cases, the anti-projection adds the deleted nodes to the inequality and increases the right hand side by value one. In this case we get

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \leq 2,$$

which is a valid inequality for $P_{STAB}(G)$. We recognize that it is a lifted[3] odd-hole inequality, which is facet-defining for $P_{STAB}(G)$. In general, it turns out that facet-defining inequalities for the polytope of the projected graph might not be facet-defining for the polytope of the original graph and vice versa. The method was successfully developed and implemented by Rossi and Smriglio, [RS01]. More details and polyhedral results can be found in [Reb06].



(a) Graph G

(b) Graph $\overline{G}$ after edge-projection of edge v_3v_5

Figure 6: Edge projection

2.3 Branching

The branching strategy in a Branch & Cut algorithm influences the overall performance of the algorithm very much. In general, it is very difficult to find a good strategy. Various techniques have been explored and none of them can always outperform the others. But for special problems, there are different strategies that help to reduce the size of the Branch & Bound tree and speedup the algorithm. We start this section with a motivation for the need of special techniques and present

[1] The reason therefore is that v_4 is the common neighbor of nodes v_3 and v_5.

[2] False edges are added between the set of nodes which are only neighbors of v_3 and not of v_5 and the set of nodes which are only neighbors of v_5 and not of v_3.

[3] The extension of a valid inequality for P to a valid inequality for a higher dimensional polytope $\overline{P} \supseteq P$ is called lifting.

the branching idea of Balas and Yu from 1986, [BY86]. It still remains the most successful strategy in practice.

One standard branching idea for a problem with binary variables is to generate two subproblems. One variable is set to value one in one subproblem and to value zero in the other one. However, this branching strategy leads to a very unbalanced Branch & Bound tree for the maximum stable set problem. This can be seen by the following argument. Setting a variable to value one sets all nodes of its neighborhood to value zero. In contrast, setting a variable to value zero has no consequence for all other variables of the graph. To avoid this drawback, one could think to set in each subproblem of the tree at least one variable to value one. Basically, this is the key property of the branching strategy by Balas and Yu.

Let $G' = (V', E')$ be the subgraph induced by the set of nodes which are not fixed in a current subproblem. In each subproblem, the goal is to find a maximum stable set in the particular subgraph given by the tree, or to prove that $\alpha(G') < LB$, with the lower bound LB. Let $W \subseteq V'$, and assume that we can show that $\alpha(G[W]) \leq LB$. Clearly, if $W = V'$, the subproblem can be fathomed; otherwise, if $\alpha(G') > LB$ any maximum stable set must contain at least one node of set $Z := V' \setminus W = \{v_1, \ldots, v_p\}$. Based on this observation, Balas and Yu showed that every maximum-cardinality stable set with greater weight than the lower bound must be contained in one of the sets

$$V'_i := \{v_i\} \cup V' \setminus (\Gamma(v_i) \cup \{v_{i+1}, \ldots, v_p\}) \quad \text{for } i = 1, \ldots, p.$$

Note, that this strategy is also true for the weighted case with $c \neq 1$. This branching leads to p new subproblems in one branching step. In each subproblem, node v_i is set to value one, and all nodes of $\Gamma(v_i) \cup \{v_{i+1}, \ldots, v_p\}$ are set to value zero.

Let us now discuss some properties of this strategy. The size of W and the ordering of the nodes in Z can affect the total number of subproblems to be solved. Of course, the larger W is, the fewer subproblems will be generated in that state. The size of W is strongly effected by the quality of the computed lower bound. Determining W is quite crucial and can be done for the cardinality stable set problem, for instance, with a clique covering, cf. [BY86, RS01]. Also other methods such as matchings or holes [Sew98] have been considered. In addition, the choice of the branching variable also has a great impact on the number of subproblems being solved. The size of the tree can be reduced by branching on nodes with a high degree, which was empirically shown by Carraghan and Pardalos [CP90]. The reason is the previously mentioned observation that with the branching node its neighborhood is also set. To sort the nodes in each subproblem in ascending order of degree seems to be computationally expensive as the degree of all nodes has to be calculated in each branching step prior to sorting. However, Sewell [Sew98] showed experimentally that for sparse graphs this is still convenient.

2.4 Implementation aspects

In general, when implementing a solver for the stable set problem, the following two things are crucial. First, one has to obtain a good lower bound, which means in the case of maximization, a feasible solution. One should use one of the many suggested heuristics in literature. We refer to the article "*Heuristics for maximum clique and independent set*" (in *Encyclopedia of Optimization*). Second, it is recommended to have a strong preprocessing. This becomes even more important when the graphs result from applications. Many contributions have been made for this purpose. Among them are, for instance, fixing of nodes, fixing of cliques and treating connected components separately, [NT75, SVA00, Reb06].

For the case of a Branch & Cut algorithm in particular, one first needs a formulation of the stable set problem. We discussed some of the relaxations in Section 2.1. For practical efficiency, it is not recommended to start with the optimization over $P_{RSTAB}(G)$. The reasons are that this relaxation is very weak and contains relatively many constraints. A better idea is to start with maximal cliques which contain all edges. Such a covering can be found in linear time. The resulting relaxation is stronger, as each maximal clique is facet-defining and dominates all the edge inequalities contained. Recognize that for bipartite graphs the relaxations is the same for both methods.

If one decides to separate several classes of inequalities within a Branch & Cut framework, one needs an order in which the separation routines are called. It is recommended to first call the polynomial separation routines, then the ones which take higher computational effort. However, practical tests show that the clique inequalities are very important. Therefore, a Branch & Cut solver should focus on fast heuristic separation of the clique inequalities combined with the very powerful tool of edge-projection. This leads to the best computational results so far.

Moreover, it is recommended to focus on facet-defining inequalities. Therefore, each clique should be lifted to a maximal clique before its inequality is added to the formulation. Accordingly, each odd-cycle should be checked not to contain any chords, and if so, the smaller odd-cycle would be added instead. These transformations can be done in linear time.

More details regarding implementation and Branch & Cut modules for the stable set problem can be found, for instance, in [RS01, BCCP96, Reb06, CLI00].

3 Conclusion

Many contributions have been made to solve the stable set problem exactly. One of the exact algorithms is based on the Branch & Cut method. However, the stable set polytope is not yet fully understood, and the known inequalities are either easy to separate with little impact, or they can only be separated with a large amount of computational effort and are very crucial for polyhedral approaches to the stable set problem. Therefore, it is not a surprise that there are some stable set instances with less than 1000 nodes which cannot be solved exactly with current methods.

4 Crossreferences

ϑ-function (Lovász number); Graph Coloring: Exact algorithms and integer programming; Integer programming; Integer programming: Branch and bound methods; Integer programming: Branch and cut algorithms, branch and cut; Integer programming: Cutting plane algorithms; Heuristics for maximum clique and independent set

References

[ABCC01] David Applegate, Robert Bixby, Vašek Chvátal, and William Cook. TSP Cuts Which Do Not Conform to the Template Paradigm. *Computational Combinatorial Optimization*, LNCS 2241:157–222, 2001.

[Ale03] Vladimir E. Alekseev. On easy and hard hereditary classes of graphs with respect to the independent set problem. *Discrete Appl. Math.*, 132(1-3):17–26, 2003.

[AS92] S. Arora and S. Safra. Probabilistic Checking of Proofs; a new Characterization of NP. In *Proceedings 33rd IEEE Symposium on Foundations of Computer Science*, pages 2–13. IEEE Computer Society, Los Angeles, CA, 1992.

[Bal70] M. L. Balinski. On Maximum Matching, Minimum Covering and their Connections. In H.W. Kuhn, editor, *Proceedings of the Princeton symposium on mathematical programming*, pages 303–312. Princeton University Press, Princeton, N.J., 1970.

[Bar00] E. R. Barnes. *A Branch-and-Bound Procedure for the Largest Clique in a Graph.* Approximation and Complexity in Numerical Optimization: Continuous and Discrete Problems. Kluwer Academic Publishers, Boston, 2000.

[BBPP99] I. M. Bomze, M. Budinich, P. M. Pardalos, and M. Pelillo. *The Maximum Clique Problem.* Handbook of Combinatorial Optimization. Kluwer Academic Publishers, Boston, 1999.

[BCCP96] E. Balasa, S. Ceria, G. Cornuejols, and G. Pataki. chapter Polyhedral methods for the maximum clique problem, pages 11–28. American Mathematical Society, 1996. D. S. Johnson and M. A. Trick (eds.), DIMACS Vol. 26.

[BCN87] Egon Balas, Vašek Chvátala, and Jaroslav Nešetřil. On the Maximum Weight Clique Problem. *Mathematics of Operations Research*, 12(3):522–535, August 1987.

[BDLS05] Flavia Bonomo, Guillermo Durán, Min Chih Lin, and Jayme L. Szwarcfiter. On Balanced Graphs. *Mathematical Programming*, 105(2–3):233–250, February 2005.

[BHO] BHOSLIB: Benchmarks with Hidden Optimum Solutions for Graph Problems (Maximum Clique, Maximum Independent Set, Minimum Vertex Cover and Vertex Coloring) – Hiding Exact Solutions in Random Graphs, 2000. http://www.nlsde.buaa.edu.cn/˜kexu/benchmarks/graph-benchmarks.htm.

[BK73] C. Bron and J. Kerbosch. Algorithm 457: Finding all cliques on an undirected graph. *Communications of ACM*, 16:575–577, 1973.

[BK97] Binay K. Bhattacharya and Damon Kaller. An O(m + n log n) Algorithm for the Maximum-Clique Problem in Circular-Arc Graphs. *Journal of Algorithms*, 25(3):336–358, November 1997.

[BLM97] Jean-Marie Bourjolly, Gilbert Laporte, and Hélène Mercure. A combinatorial column generation algorithm for the maximum stable set problem. *Operations Research Letters*, 20(1):21–29, 1997.

[BMZ02] Samuel Burer, Renato D. C. Monteiro, and Yin Zhang. Maximum stable set formulations and heuristics based on continuous optimization. *Mathematical Programming*, 94(1):137–166, 2002.

[BP76] Egon Balas and Manfred W. Padberg. Set Partitioning: A Survey. *SIAM Review*, 18(4):710–761, 1976.

[Brø83] Arne Brønsted. *An introduction to Convex Polytopes*. Graduate Texts in Mathematics. Springer, 1983.

[BS99] I. M. Bomze and V. Stix. Genetic engineering via negative fitness: Evolutionary dynamics for global optimization. *Annals of Operations Research*, 89:297–318, 1999.

[BT90] L. Babel and G. Tinhofer. A branch and bound algorithm for the maximum clique problem. *ZOR-Methods and Models of Operations Research*, 34:207–217, 1990.

[But03] Sergiy Butenko. *Maximum Independent Set and Related Problems, with Applications*. PhD thesis, University of Florida, USA, 2003.

[BX96] E. Balas and J. Xue. Weighted and Unweighted Maximum Clique Algorithms with Upper Bounds from Fractional Coloring. *Algorithmica*, 15(5):397–412, 1996.

[BY86] Egon Balas and Chang Sung Yu. Finding a Maximum Clique in an Arbitrary Graph. *Siam Journal on Computing*, 14(4):1054–1068, 1986.

[BY89] Egon Balas and Chang Sung Yu. On graphs with polynomially solvable maximum-weight clique problem. *Networks*, 19(2):247–253, March 1989.

[CC95] Eddie Cheng and William H. Cunningham. *Separation problems for the stable set polytope*, pages 65–79. The 4th Integer Programming and Combinatorial Optimization Conference Proceedings. 1995. E. Balas and J. Clausen (eds.).

[CC97] Eddie Cheng and William H. Cunningham. Wheel Inequalities for Stable Set Polytopes. *Mathematical Programming*, 77:389–421, 1997.

[CCL+05] Maria Chudnovsky, Gérard Cornuéjols, Xinming Liu, Paul Seymour, and Kristina Vušković. Recognizing Berge Graphs. *Combinatorica*, 25(2):143–186, March 2005.

[CdV02] Eddie Cheng and Sven de Vries. Antiweb-wheel inequalities and their separation problems over the stable set polytopes. *Mathematical Programming*, 92(1):153–175, March 2002.

[CFL00] A. Caprara, M. Fiscetti, and A. N. Letchford. On the Separation of Maximally Violated mod-k Cuts. *Mathematical Programming*, 87(1):37–56, 2000.

[CLI00] Robert D. Carr, Giuseppe Lancia, and Sorin Istrail. Branch-and-Cut Algorithms for Independent Set Problems: Integrality Gap and An Application to Protein Structure Alignment. Technical report, Sandia National Laboratories, Albuquerque, NM (US); Sandia National Laboratories, Livermore, CA (US), September 2000.

[CN85] N. Chiba and T. Nishizeki. Arboricity and subgraph listing algorithms. *Siam Journal on Computing*, 14:210–223, 1985.

[CP90] Randy Carraghan and Panos M. Pardalos. An exact algorithm for the maximum clique problem. *Operations Research Letters*, 9(6):375–382, November 1990.

[CRST04] Maria Chudnovsky, Neil Robertson, Paul Seymour, and Robin Thomas. The Strong Perfect Graph Theorem. Typescript, revised June 4, 2004.

[CT05] O. Cogisa and E. Thierry. Computing maximum stable sets for distance-hereditary graphs. *Discrete Optimization*, 2(3):185–188, June 2005.

[Die00] Reinhard Diestel. *Graph Theory*. Electronic Edition 2000. Springer-Verlag New York 1997, 2000.

[dim] Second DIMACS Challenge, 1992/1993. http://mat.gsia.cmu.edu/challenge.html.

[Fah02] Torsten Fahle. volume 2461/2002 of *Lecture Notes in Computer Science*, chapter Simple and Fast: Improving a Branch-And-Bound Algorithm for Maximum Clique, pages 485–498. 2002.

[FMK95] Katsuku Fujisawa, S. Morito, and Miko Kubo. Experimental Analyses of the Life Span Method for the Maximum Stable Set Problem. *The Institute of Statistical Mathematics Cooperative Research Report*, 75:135–165, 1995.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability, A guide to the Theory of NP-Completeness*. A series of books in the mathematical sciences, Vicot Klee, Editor. W. H. Freeman and Company, 1979.

[GJR84] M. Grötschel, M. Jünger, and G. Reinelt. A Cutting Plane Algorithm for the Linear Ordering Problem. *Operations Research*, 32:1195–1220, 1984.

[GL06] Monia Giandomenico and Adam N. Letchford. Exploring the Relationship Between Max-Cut and Stable Set Relaxations. *Mathematical Programming*, 106(1):159–175, 2006.

[GLS81] Martin Grötschel, László Lovász, and Alexander Schrijver. The Ellipsoid Method and Its Consequences in Combinatorial Optimization. *Combinatorica*, 1:169–197, 1981.

[GLS88] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Algorithms and Combinatorics 2. Springer, 1988.

[GP81] M. Grötschel and W. R. Pulleyblank. Weakly Bipartite Graphs and the Max-cut Problem. *Operations research letters*, 1(1):23–27, October 1981.

[GS86] A. M. H. Gerards and A. Schrijver. Matrices with the Edmonds-Johnson property. *Combinatorica*, 6(4):365–379, 1986.

[HPV93] Jonas Hasselberg, Panos M. Pardalos, and George Vairaktarakis. Test case generators and computational results for the maximum clique problem. *Journal of Global Optimization*, 3(4):463–482, Dezember 1993.

[HR57] F. Harary and I. C. Ross. A procedure for clique detection using the group matrix. *Sociometry*, 20:205–215, 1957.

[KW97] Josef Kallrath and John M. Wilson. *Business Optimization using Mathematical Programming*. Macmillan, 1997.

[Law01] Eugene Lawler. *Combinatorial Optimization: Networks and Matroids*. Dover Publications, 2001.

[LKSN06] Katharina A. Lehmann, Michael Kaufmann, Stephan Steigele, and Kay Nieselt. On the maximal cliques in c-max-tolerance graphs and their application in clustering molecular sequences. *Algorithms for Molecular Biology*, 1:9:1–17, 2006.

[Lov79] László Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information Theory*, 25(1):1–7, 1979.

[LT81] E. Loukakis and C. Tsouros. A depth first serach algorithm to generate the family of maximal independet sets of a graph lexicographically. *Computing*, 27:249–266, 1981.

[MNKF90] Sumio Masuda, Kazuo Nakajima, Toshinobu Kashiwabara, and Toshio Fujisawa. Efficient algorithms for finding maximum cliques of an overlap graph. *Networks*, 20(2):157–171, March 1990.

[Mos97] Raffaele Mosca. Polynomial algorithms for the maximum stable set problem on particular classes of p5-free graphs. *Inf. Process. Lett.*, 61(3):137–143, 1997.

[MS96] C. Mannino and A. Sassano. Edge Projection and the Maximum Cardinality Stable Set Problem. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 26:249–261, 1996.

[MS99] Carlo Mannino and Egidio Stefanutti. An augmentation algorithm for the maximum weighted stable set problem. *Comput. Optim. Appl.*, 14(3):367–381, 1999.

[MS05] C. Mannino and A. Sassano. An exact algorithm for the maximum stable set problem. *Computational Optimization and Applications*, 3(3):243–258, 2005.

[NT74] G. L. Nemhauser and L. E. Trotter, Jr. Properties of Vertex Packing and Independence System Polyhedra. *Mathematical Programming*, 6:48–61, 1974.

[NT75] G. L. Nemhauser and L. E. Trotter, Jr. Vertex Packings: Structural Properties and Algorithms. *Mathematical Programming*, 8:232–248, 1975.

[NW88] G. L. Nemhauser and Laurence A. Wolsey. *Integer and Combinatorial Optimization*. Wiley-Interscience Series in Discrete Mathematics and Optimization. Johnson Wiley & Sons, Inc., 1988.

[Ola89] Stephan Olariu. Weak bipolarizable graphs . *Discrete Mathematics*, 74(1–2):159–171, 1989.

[Öst02] Patric R. J. Östergård. A fast algorithm for the maximum clique problem. *Discrete Appl. Math.*, 120(1-3):197–207, 2002.

[Pad73] Manfred W. Padberg. On the Facial Structure of Set Packing Polyhedra. *Mathematical Programming*, 5:199–215, 1973.

[PP90] P. M. Pardalos and A. T. Phillips. A global optimization approach for solving the maximum clique problem. *International Journal of Computer Mathematics*, 33(3–4):209–216, 1990.

[PR87] M. W. Padberg and G. Rinaldi. Optimization of a 532 City Symmetric Traveling Salesman Problem by Branch and Cut. *Operations Research Letters*, 6:1–7, 1987.

[PR92] Panos M. Pardalos and Gregory P. Rodgers. A branch and bound algorithm for the maximum clique problem. *Computers & Operations Research*, 19(5):363–375, July 1992.

[PY81] Christos H. Papadimitriou and Mihalis Yannakakis. The clique problem for planar graphs. *Information Processing Letters*, 13(4–5):131–133, 1981.

[RAR01] Jorge L. Ramírez-Alfonsín and Bruce A. Reed, editors. *Perfect Graphs*. John Wiley & Sons, 2001.

[Reb06] Steffen Rebennack. Maximum Stable Set Problem: A Branch & Cut Solver. Diplomarbeit, Ruprecht–Karls Universität Heidelberg, Heidelberg, Germany, 2006.

[Rég03] Jean-Charles Régin. chapter Using consraint Programming to Solve the Maximum Clique Problem, pages 634–648. Lecture Notes in Computer Science. Springer, 2003.

[RS01] Fabrizio Rossi and Stefano Smriglio. A Branch-and-Cut Algorithm for the Maximum Cardinality Stable Set Problem. *Operations Research Letters*, 28:63–74, 2001.

[Sch03] Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*, volume 24 of *Algorithms and Combinatorics*. Springer, 2003.

[Sew98] E. C. Sewell. A Branch and Bound Algorithm for the Stability Number of a Sparse Graph. *INFORMS Journal on Computing*, 10(4):438–447, 1998.

[SVA00] Tycho Strijk, Bram Verweij, and Karen Aardal. Algorithms for maximum independent set applied to map labelling. Technical Report UU-CS-2000-22, 2000.

[Tro75] L. E. Trotter. A class of facet producing graphs for vertex packing polyhedra. *Discrete Mathematics*, 12(4):373–388, 1975.

[TTT88] E. Tomita, A. Tanaka, and H. Takahashi. The worst-time complexity for finding all the cliques. Technical report, University of Electro-Communications, Tokyo, Japan, 1988.

[VA99] Bram Verweij and Karen Aardal. volume 1643/1999 of *Lecture Notes in Computer Science*, chapter An Optimisation Algorithm for Maximum Independent Set with Applications in Map Labelling, pages 426–437. August 1999.

[Wes00] Douglas B. West. *Introduction to Graph Theory, Second Edition*. Prentice Hall, 2000.

[WH06] Jeffrey S. Warren and Illya V. Hicks. Combinatorial Branch-and-Bound for the Maximum Weighted Independent Set Problem. working paper, August 7, 2006.

[Wol98] Laurence A. Wolsey. *Integer Programming*. Series in Discrete Mathematics and Optimization. John Wiley & Sons, 1998.

[Woo97] David R. Wood. An algorithm for finding a maximum clique in a graph. *Operations Research Letters*, 21(5):211–217, 1997.

[WWWH05] Deepak Warrier, Wilbert E. Wilhelm, Jeffrey S. Warren, and Illya V. Hicks. A branch-and-price approach for the maximum weight independent set problem. *Netw.*, 46(4):198–209, 2005.

[YFO06] E. Alper Yildirim and Xiaofei Fan-Orzechowski. On Extracting Maximum Stable Sets in Perfect Graphs Using Lovász's Theta Function. *Computational Optimization and Applications*, 33(2–3):229–247, 2006.

[Zie95] Günter M. Ziegler. *Lecture on Polytopes*. Graduate Texts in Mathematics. Springer, 1995.