Approximating Stable Sets Using the ϑ-function and Cutting Planes

Gerald Gruber * Franz Rendl *

Universität Klagenfurt, Institut für Mathematik
Universitätsstraße 65-67, A-9020 Klagenfurt, Austria
{gerald.gruber, franz.rendl}@uni-klu.ac.at

Abstract. We investigate an approximation algorithm for the maximum stable set problem based on the Lovász number $\vartheta(G)$ as an initial upper bound. We strengthen this relaxation by adding two classes of cutting planes, odd circuit and triangle inequalities. We present computational results using this tighter model on several classes of graphs.

1 Linear and Semidefinite Relaxations of the Stable Set Problem

Let $G = (V, E)$ denote an undirected graph with node set V and edges set E, where $|V| = n$ and $|E| = m$. We denote an edge e with endnodes i and j by (ij).
A stable set S in G is by definition a subset S of pairwise nonadjacent nodes. The *maximum stable set problem* is the problem of finding a stable set of maximum cardinality. Its cardinality is denoted by $\alpha(G)$.
An assignment of colors to the vertices of G, one color to each vertex, so that adjacent vertices receive different colors, is called a *coloring of G*. A k-coloring of G is a coloring of G using k colors. The chromatic number $\chi(G)$ of a graph G is the minimum value k for which a k-coloring of G exists. The complement of a graph G is denoted by $\bar{G}$.
It is NP-complete to compute $\alpha(G)$ or $\chi(G)$. The Lovász number $\vartheta(G)$ bounds $\alpha(G)$ from above and $\chi(\bar{G})$ from below, $\alpha(G) \leq \vartheta(G) \leq \chi(\bar{G})$. This number is also called theta function (ϑ-function) of G. $\vartheta(G)$ can be computed in polynomial time, see [2]. In this work we are primarily interested in estimating $\alpha(G)$.
The stable set polytope of G is defined by

$$STAB(G) := \text{conv}\{\ \chi^S \in \mathbb{R}^n | S \subseteq \text{V stable set}\}.$$

where χ^S denotes the incidence vector of S, i.e., $\chi_i^S = \begin{cases} 1 & \text{for } i \in S \\ 0 & \text{otherwise} \end{cases}$. Thus the maximum stable set problem **(STAB)** is given by

$$(\textbf{STAB}) \qquad \max \sum_{i=1}^{n} x_i \quad \text{such that} \quad x \in STAB(G).$$

A complete description of STAB(G) by linear inequalities is unlikely to be tractable, in view of the NP-completeness to compute $\alpha(G)$. The following sets of linear inequalities contain a partial description of STAB(G).

$$\begin{array}{rlll} \text{nonnegativity constraints:} & x_i \geq 0 & \forall i \ \in \ V & (1) \\ \text{edge constraints:} & x_i + x_j \leq 1 & \forall (ij) \ \in \ E & (2) \\ & x_i \leq 1 & \text{for isolated nodes} & \end{array}$$

Obviously (1) and (2) hold for $x \in STAB(G)$. Moreover, the integral solutions x of (1) and (2) are the incidence vectors of stable sets in G. It is well known that (1) and (2) are enough to describe $STAB(G)$ if and only if G is bipartite and has no isolated nodes, see [4].

*Partial financial support from the Austrian Science Foundatation, FWF Project P12660-MAT is gratefully acknowledged.

However, if G contains odd circuits this polytope is no longer enough to describe $STAB(G)$. Take for instance a triangle T and set $x_i = \frac{1}{2}$ for $i \in V(T)$. Clearly x satisfies (1) and (2) but is not in $STAB(G)$. Therefore the set of inequalities (1) and (2) is extended by adding the class of odd circuit constraints (3),

$$\sum_{i \in V(C)} x_i \leq \frac{|V(C)|-1}{2} \quad \text{for } C \text{ odd circuit}, \tag{3}$$

yielding the polytope $CSTAB(G)$:

$$CSTAB(G) := \{ x \in \mathbb{R}^n | x \text{ satisfies } (1),(2) \text{ and } (3)\}.$$

The polytope $CSTAB(G)$ is still only a (linear) relaxation of $STAB(G)$, i.e. $STAB(G) \subseteq CSTAB(G)$. Any linear function can be optimized over $CSTAB(G)$ in polynomial time [3]. The relaxation $CSTAB(G)$ gives $\alpha(G)$ for the class of $t-$perfect graphs, see [3, 4].

We denote by $S(n)$ the space of $n \times n$ symmetric matrices. A nonlinear relaxation of $STAB(G)$ was introduced by Lovász, see [10], and can also be found in [11]. There are many equivalent ways to define this relaxation, we use the following definition, see [11]:

$$TH(G) := \{d \in \mathbb{R}^n | \exists X \in S(n) \text{ such that } d = diag(X), \text{ and } X \text{ satisfies } (4)\}$$

$$X = (x_{ij}) \text{ with } x_{ij} = 0 \text{ for } (ij) \in E(G),\ d = diag(X),\ \bar{X} := \begin{pmatrix} X & d \\ d^T & 1 \end{pmatrix} \succeq 0. \tag{4}$$

The set $TH(G)$ is a relaxation of the stable set polytope, because the characteristic vector χ of a stable set produces the matrix $\chi\chi^T$, which is feasible for (4).

One can optimize over $TH(G)$ in polynomial time, see [2]. In fact, see [11, 4, 9]

$$\vartheta(G) := \max\left\{\sum_i x_i | x \in TH(G)\right\}. \tag{5}$$

Our main goal is to get tighter bounds for $\alpha(G)$, starting from $\vartheta(G)$. Therefore we extend this relaxation by adding further constraints to the set $TH(G)$. As already suggested in [7], one can include the triangle inequalities, $X \in MET$. By definition, $X \in MET$ exactly if

$$\begin{array}{l} 0 \leq X_{ij} \leq X_{ii},\ X_{ii} + X_{jj} - X_{ij} \leq 1, \\ -X_{kk} - X_{ij} + X_{ik} + X_{jk} \leq 0, \\ X_{kk} + X_{ii} + X_{jj} - X_{ij} - X_{ik} - X_{jk} \leq 1, \quad \text{for } 1 \leq i,j,k \leq n. \end{array} \tag{6}$$

A further refinement can be achieved by intersecting $TH(G)$ with $CSTAB(G)$. Finally we can combine $TH(G)$ with the odd circuit constraints and the triangle inequlities. In a slight abuse of notation we write $X \in CSTAB(G), (TH(G))$ for $diag(X) \in CSTAB(G), (TH(G))$, respectively.

The remaining sections are organized as follows. In section 2 we summarize the relaxation models. Implementation details are described in section 3. The computational tests appear in section 4. We conclude with comments in section 5.

2 Strenghtening $\vartheta(G)$ by linear cutting planes

We first rewrite the problem defining $\vartheta(G)$. We write trace(X) to denote the trace of a square matrix X. The inner product in $S(n)$ is defined by $\langle A, B\rangle := \text{trace}\,(AB)$. The symbol $\succeq$ denotes the Löwner partial order, i.e., $A \succeq B$ if $A - B$ is positive semidefinite. $A \succ 0$ means A is positive definite.

We now express the constraints from (4) through $\bar{X} \in S(n+1)$. Let $E_{ij} := e_i e_j^t + e_j e_i^t$ where e_i stands for the ith unit vector in $\mathbb{R}^{n+1}$. Then $\langle E_{ij}, \bar{X}\rangle = 0$ corresponds to $\bar{X}_{ij} = 0$. $diag(X) = d$ is equivalent to $\bar{X}_{ii} = \frac{1}{2}\bar{X}_{i,n+1} + \frac{1}{2}\bar{X}_{n+1,i}$ for $1 \leq i \leq n$. Setting $a_i := e_i e_i^t - \frac{1}{2} e_{n+1} e_{n+1}^t$, and using $\bar{X}_{n+1,n+1} = 1$, this is the same as $\langle a_i a_i^T, \bar{X}\rangle = \frac{1}{4}$. The relaxation (5) can therefore be written as the (primal) semidefinite program **(PSDPϑ)**.

$$\begin{array}{llll} (\mathbf{PSDP}\vartheta) & \vartheta(G) := & \max & \operatorname{trace} \bar{X} - 1 \\ & & \text{s.t.} & \langle E_{ij}, \bar{X}\rangle = 0 \quad \forall (ij) \in E(G) \\ & & & \bar{X}_{n+1,n+1} = 1 \\ & & & \langle a_i a_i^T, \bar{X}\rangle = \frac{1}{4} \quad 1 \le i \le n \\ & & & \bar{X} \in S(n+1),\ \bar{X} \succeq 0. \end{array} \tag{7}$$

The number of constraints in this model is $m + n + 1$.
We extend **(PSDPϑ)** by adding odd circuit constraints and triangle inequalities. This gives rise to the following relaxations, where (8) was already investigated in [7].

$$\max \operatorname{trace} \bar{X} - 1 \quad \text{such that} \quad X \in TH$$

$$\max \operatorname{trace} \bar{X} - 1 \quad \text{such that} \quad X \in TH \cap MET \tag{8}$$

$$\max \operatorname{trace} \bar{X} - 1 \quad \text{such that} \quad X \in TH \cap CSTAB \tag{9}$$

$$\max \operatorname{trace} \bar{X} - 1 \quad \text{such that} \quad X \in TH \cap CSTAB \cap MET. \tag{10}$$

We compare the quality of these relaxations and investigate their computational effort on a series of randomly generated problems. It should be noted that the computational effort to solve these relaxations is heavily influenced by the number of constraints included. Even if a constraint is inactive at the optimum, and hence could have been ignored without weakening the relaxation, it causes the same computational cost as active constraints. The computational effort is ultimately growing at least cubically with the number of constraints (because a linear system has to be solved for the dual variables), therefore it is extremely important to identify important constraints. Therefore we propose an iterative procedure and include only a few of the most violated inequalities and drop constraints as they become inactive. Further details are described in the next sections.

3 Implementation Details

In this section we briefly sketch some details of our implementation. A more precise description can be found in the thesis [5].
The semidefinite programs are solved using primal-dual path following interior point methods. We follow the implementation given by Helmberg et al. [8]. All our codes are implemented in Matlab.
We compute the optimal solution of **(PSDPϑ)** and take $\vartheta(G)$ as initial upper bound on $\alpha(G)$. The main diagonal of the optimal X is used to generate a stable set as follows. A large entry $\bar{x}_{ii}$ indicates that vertex i is likely to be contained in a large stable set. Hence we run through all the vertices of G in the order given by the main diagonal of $\bar{X}$ sorted nonincreasingly. We select the vertices to be contained in a stable set with probability proportional to $\bar{x}_{ii}$. The final stable set is locally improved, and the process is repeated several times. We include the size of a stable set found this way in the tables of the next section.
The upper bound is improved now by identifying and including violated odd circuit constraints and/or triangle inequalities. In general there are a lot more violated inequalities (=cutting planes) than we are willing to add to the current relaxation. Since the computational effort is growing drastically with the number of constraints, we follow an iterative procedure and add only the most violated constraints. This process is iterated several times.
In our experiments we found that computing the relaxation (9) can be done efficiently with computational effort of the same order of magnitude as computing $\vartheta(G)$. On the other hand we found it impractical to optimize over all constraints MET, because there was a typical tailing off effect. The biggest improvement in the bound is obtained in the first few iterations of adding violated inequalities. In subsequent iterations, the bound is hardly reduced, the largest violation of MET is already quite small, but there are still a lot of inequalities violating MET. In this situation it does not seem to be advisable to continue the iterations, because of the excessive computational effort. Further details are contained in the next section.

4 Numerical Results

In this section we report computational experience on different classes of test problems, applied to the relaxations just described. All our computational tests were performed on a DEC Alpha. The experiments were carried out on the following sets of graphs.

1. Random graphs with given density,
2. Sanchis graphs, see [12, 6],
3. Triangulated planar graphs,
4. Graphs with high girth.

Sanchis graphs are constructed in such a way that $\alpha(G)$ is known. Originally they were designed to test maximum clique algorithms. For our tests we take the complement graph and indicate this by suffix _c in the tables. These graphs can be downloaded from: *ftp://dimacs.rutgers.edu/pub/challenge/*. Triangulated planar graphs are perfect, thus $\vartheta(G) = \alpha(G)$, see [4]. To generate graphs with high girth, we start with a randomly generated graph G and randomly replace edges of G by odd paths of prescribed maximum length. The computational results on the four problem sets are reported in Tables 1 to 4. Column n gives the number of vertices, column m the number of edges in the graph G. The column labeled *lbd on* $\alpha(G)$ stands for "lower bound on $\alpha(G)$". This column gives the size of the largest stable set which we have found using our heuristic method scetched in the previous section. The remaining columns contain the upper bounds found by our code. In addition to the value of the upper bound, we also provide the gap in % with respect to the values given in column *lbd on* $\alpha(G)$, separated from the upper bound by a dash. The numbers given in brackets represent the number of active inequality constraints of the final relaxation. Two numbers separated by a slash within one bracket show remaining odd circuit and triangle inequalities separately.
We note first that for the triangulated planar graphs of Table 3 we are able to compute $\alpha(G)$ quite efficiently even for large graphs. This is not surprising since these graphs are perfect. The computation times are reasonably moderate for these graphs.
Looking at the Tables 1 and 4, we note that optimizing over (9) leads to substantial improvements of $\vartheta(G)$. The inclusion of additional inequalities from MET leads to more improvements, as can be seen from the last columns in Tables 1 and 4, however associated with an enormous increase of computational cost (see Table 5). The final solutions do not satisfy all the triangle inequalities $X \in MET$, but their violation has gone down from typically 0.1 to 0.03 (in the test sets of dimension 50 to 300 in Table 1). It is for the reasons stated in section 3 that we only did a limited number of rounds of adding inequalities on the remaining test data. No odd circuit constraints are violated, though. It should also be noted that only a few odd circuit constraints are active in the final solution. Note that we have found already the optimal solution of many of the test problems with this approach. This is denoted by a star in column *lbd on* $\alpha(G)$.
It is also interesting to see that the class of Sanchis graphs seems to be easy to approximate through $\vartheta(G)$. This is in contrast to several heuristic approaches, which have difficulties finding large stable sets in these graphs, see [1].
Table 5 compares computation times from optimizing over (10), (9) and (5) on several randomly generated sparse problems of varying dimension. Each line lists the average CPU time to solve 10 problems of given size and density. The values in the brackets indicate the (average number of) number of active constraints at termination. As mentioned before, optimizing over MET is not done exactly, but the iterations are stopped once the maximum violation of MET has fallen below a threshold (of 50 % of the maximum violation in our case).

5 Concluding Comments

We derive the following conclusions from our computational tests. The process of adding and dropping cutting planes works robust and can significantly improve the quality of the final solution. However, we observed also that the improvement goes at the expense of computation time, which increases dramatically with growing dimension. The results summarized in Table 5 indicate that the inclusion of triangles is fairly expensive, see also [7]. We observed that optimizing over (9) is not much harder than computing $\vartheta(G)$. This justifies our strategy. On the other hand, a combination of triangles and odd circuits produces the widest

n	m	lbd on $\alpha(G)$	$\vartheta(G)$	optimizing over $TH(G) \cap CSTAB(G)$		optimizing over $TH(G) \cap MET \cap CSTAB(G)$	
50	251	14*	15.0016 - 7.15	14.9990 - 7.14	(1)	14.6960 - 4.97	(1/130)
80	320	27*	28.1924 - 4.42	28.1053 - 4.09	(20)	27.5501 - 2.04	(8/315)
100	393	35	37.0451 - 5.84	36.8770 - 5.36	(21)	36.2010 - 3.43	(7/320)
100	434	33	35.1823 - 6.61	35.0532 - 6.22	(21)	34.3682 - 4.15	(3/361)
150	451	59	62.2523 - 5.51	61.4944 - 4.23	(77)	60.6595 - 2.81	(37/694)
150	552	55	58.0287 - 5.51	57.6335 - 4.79	(49)	56.5200 - 2.76	(21/710)
170	383	77*	79.3290 - 3.02	77.6756 - 0.88	(60)	77.5463 - 0.71	(44/318)
170	423	75*	76.7735 - 2.36	75.6992 - 0.93	(56)	75.4973 - 0.66	(40/305)
200	418	93*	94.5638 - 1.68	93.2256 - 0.24	(65)	93.0111 - 0.01	(52/401)
200	563	81	85.5002 - 5.56	84.3313 - 4.11	(99)	83.1449 - 2.65	(39/1059)
300	894	121	129.3327 - 6.89	127.7402 - 5.57	(153)	126.4218 - 4.48	(55/1195)
350	994	146	152.5927 - 4.52	150.3067 - 2.95	(183)	150.1405 - 2.84	(122/213)
400	835	191*	193.4866 - 1.30	191.0000 - 0.00	(132)	191.1058 - 0.06	(72/5)

Table 1: *Numerical Results on randomly generated sparse graphs*

problem name	n	m	*lbd on* $\alpha(G)$	$\alpha(G)$	$\vartheta(G)$	optimizing over $TH(G) \cap MET \cap CSTAB(G)$	
san200_0.9_1.clq_c	200	1990	70	70	70.0000	-	
san200_0.9_2.clq_c	200	1990	60	60	60.0000	-	
san200_0.9_3.clq_c	200	1990	44	44	44.0000	-	
sanr200_0.9.clq_c	200	2037	42	-	49.2735	48.8689	(0/294)

Table 2: *Numerical Results on Sanchis graphs*

n	m	*lbd on* $\alpha(G)$	$\vartheta(G)$	CPU time [sec]
100	294	44	44.0000	2.2698
200	594	87	87.0000	16.5672
300	894	126	126.0000	53.6591
400	1194	174	174.0000	155.8248
500	1494	220	220.0000	335.4370
600	1794	270	270.0000	606.9892

Table 3: *Numerical Results on triangulated planar graphs*

n	m	lbd on $\alpha(G)$	$\vartheta(G)$	optimizing over $TH(G) \cap CSTAB(G)$		optimizing over $TH(G) \cap MET \cap CSTAB(G)$	
118	152	57*	58.6107 - 2.83	57.4000 - 0.70	(18)	57.3252 - 0.57	(14/83)
292	365	141*	144.2935 - 2.34	142.6667 - 1.18	(86)	141.9742 - 0.69	(10/181)
280	394	134*	137.1680 - 2.36	136.6476 - 1.98	(94)	134.4599 - 0.34	(3/215)
282	339	136	139.5288 - 2.59	137.6923 - 1.24	(19)	137.7490 - 1.29	(16/120)
372	534	179*	181.2776 - 1.27	179.0000 - 0.00	(96)	179.0000 - 0.00	(104/370)
448	465	220	223.8245 - 1.74	222.3333 - 1.06	(20)	222.3333 - 1.06	(19/1)
520	947	245*	250.1892 - 2.12	250.1473 - 2.10	(7)	245.9738 - 0.40	(0/354)
544	688	264	269.0532 - 1.91	267.0000 - 1.14	(104)	266.1425 - 0.81	(24/254)

Table 4: *Numerical Results on graphs with long odd circuits*

		CPU time [sec] from optimizing over				
n	*density*[%]	$TH(G)$	$TH(G) \cap CSTAB(G)$		$TH(G) \cap MET \cap CSTAB(G)$	
50	10	0.7125	4.3357	(10)	8.8111	(8/47)
100	10	5.0945	41.4115	(9)	141.8472	(2/180)
150	5	10.0519	150.7397	(42)	354.3603	(22/266)
200	3	16.1386	309.5576	(94)	958.8453	(63/383)
250	2	25.5563	424.7727	(95)	1458.3181	(98/297)

Table 5: *Average Running Times on randomly generated sparse graphs*

progress. We observed also that most of the improvement is obtained in the first few rounds of adding inequalities.

We also carried out numerical experiments where we optimized over (8), as proposed in [7]. In this case the quality of the approximation was much weaker than our current results using (10). The reason for this behaviour is probably that the presence of odd circuit inequalities reduces the number of possibly violated triangle inequalities.

Further work is necessary to examine this relaxation in the context of Branch and Bound methods for the stable set problem.

References

[1] I. M. Bomze, M. Pelillo, and R. Giacomini. Evolutionary approach to the maximum clique problem: Empirical evidence on a larger scale. In I. M. Bomze, T. Csendes, R. Horst, and P. M. Pardalos, editors, *Developments in Global Optimization*, pages 95–108. Kluwer Academic Publishers, Dordrecht, 1997.

[2] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, 1981.

[3] M. Grötschel, L. Lovász, and A. Schrijver. Relaxations of vertex packing. *Journal of Combinatorial Theory*, 40:330–343, 1986.

[4] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, New York, 1988.

[5] G. Gruber. *Semidefinite Programming and Combinatorial Optimization*. PhD thesis, University of Technology, Graz, Austria, 1999. (forthcoming).

[6] Jonas Hasselberg, Panos M. Pardalos, and George Vairaktarakis. Test case generators and computational results for the maximum clique problem. *Global Optimization*, 3:463–482, 1993.

[7] C. Helmberg, S. Poljak, F. Rendl, and H. Wolkowicz. Combining semidefinite and polyhedral relaxations for integer programs. In *Proceedings of the 4th International IPCO Conference*, volume 920 of *Lecture Notes in Computer Science*, pages 124–134. Springer, 1995.

[8] C. Helmberg, F. Rendl, R. J. Vanderbei, and H. Wolkowicz. An interior point method for semidefinite programming. *SIAM Journal on Optimization*, 6(2):342–361, May 1996.

[9] Donald E. Knuth. The sandwich theorem. *The Electronic Journal of Combinatorics*, 1(A1), 1994.

[10] L. Lovász. On the shannon capacity of a graph. *IEEE Trans. Inform. Theory*, 25:1–7, 1979.

[11] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal of Optimization*, 1(2):166–190, May 1991.

[12] L. A. Sanchis. Test case construction for the vertex cover problem. In *Computational Support for Discrete Mathematics*, volume 15 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 1052–1798. American Mathematical Society, March 1994.