Chapter 1

APPROXIMATING MAXIMUM STABLE SET AND MINIMUM GRAPH COLORING PROBLEMS WITH THE POSITIVE SEMIDEFINITE RELAXATION

S. J. Benson
Division of Mathematics and Computer Science
Argonne National Laboratory
Argonne, IL 60439
benson@mcs.anl.gov

Y. Ye
Department of Management Sciences
The University of Iowa
Iowa City, Iowa 52242
yyye@yuan.biz.uiowa.edu

Abstract We compute approximate solutions to the maximum stable set problem and the minimum graph coloring problem using a positive semidefinite relaxation. The positive semidefinite programs are solved using an implementation of the dual scaling algorithm that takes advantage of the sparsity inherent in most graphs and the structure inherent in the problem formulation. From the solution to the relaxation, we apply a randomized algorithm to find approximate maximum stable sets and a modification of a popular heuristic to find graph colorings. We obtained high quality answers for graphs with over 1000 vertices and almost 7000 edges.

Keywords: Stable Set, Independent Set, Maximum Clique, Graph Coloring, Positive Semidefinite Relaxation.

1 INTRODUCTION

Given an undirected graph $G = (V, E)$, a *stable set* of vertices (or *vertex packing* or *independent set*) is a subset of V such that no two vertices are adjacent. The *Maximum Stable Set Problem* (MSS) asks for the stable set with the maximum cardinality. A *clique* of graph G is a subset set of vertices such that every pair of vertices is adjacent. A *vertex cover* is a subset of vertices that are incident to each edge in the graph. Denoting $\bar{G}$ as the graph complement of G, the following statements concerning any $S \subset V$ are known to be equivalent:

1. S is a stable set of G,
2. S is a clique of $\bar{G}$,
3. $V \setminus S$ is vertex cover of G

Accordingly, the problems of finding a maximum stable set of G, a maximum clique in $\bar{G}$, and a minimum vertex cover in G are equivalent.

A *vertex coloring* of a graph is an assignment of colors to the vertices V such that no two adjacent vertices receive the same color. Equivalently, the problem looks to partition the vertices into independent sets. The smallest number of colors needed for this coloring is called the the *chromatic number* of G. A graph is *k-colorable* if it can be colored with k colors or less. Obviously, the cardinality of any clique in G is a lower bound on the chromatic number of G. A *perfect graph* is a graph whose chromatic number equals the cardinality of its largest clique. For this special class of graphs, the MSS problem can be solved to optimality using a polynomial algorithm.

These problems are classical problems in combinatorial optimization and are well known to be NP-complete[18]. The MSS problem can be solved using polynomial time algorithms for special classes of graphs such as perfect graphs and t-perfect graphs, circle graphs and their complements, circular arc graphs and their complements, claw-free graphs, and graphs with long odd cycles[26], but the existence of a polynomial time algorithm for arbitrary graphs seems unlikely.

Various exact solution methods have been developed for these combinatorial optimization problems. An implicit enumeration technique of Carrahan and Pardalos[11], integer programming with branch and bound by Babel and Tinhofer[3][4], Balas, Xue, and Yu[6][7], Mannino and Sassano [26], and Nemhauser[29], integer programming with cutting planes by Balas [5], Nemhauser[30], and Nemhauser and Sigismondi [29], and a tabu search by Friden[16] have all been applied to the maximum stable set problem. There effectiveness, however, has usually been limited to graphs with less than 500 vertices. For the minimum graph coloring problem, implicit enumeration and branch and bound based methods of Kubale[24] [25] have been limited to very small instances, and a column generation approach based upon the stable set formulation by Mehrotra and Trick[27] has been applied to graphs with up to 600 vertices. Of course all of these algorithms have exponential complexity, so for larger graphs, the only option available is heuristic methods [33] [20] [21] [28], which have the cost of regularly suboptimal solutions.

Aside from its theoretical interest, the MSS problem arises in applications in information retrieval, experimental design, signal transmission, and computer vision[7]. Graph coloring arises when using finite differences to approximate sparse Hessian matrices, and well as applications in computer register allocation[10][13][12], timetable scheduling[9][14][39], and electronic bandwidth allocation[17]. In many of these applications, it suffices to find an approximately optimal solution. This fact and the difficulty of finding exact solutions has encouraged considerable effort on finding good approximation algorithms.

2 POSITIVE SEMIDEFINITE RELAXATIONS

The standard form of positive semidefinite program is:

$$\text{(SDP)} \qquad \begin{array}{ll} \text{Minimize} & C \bullet X \\ \text{Subject to} & A_i \bullet X = b_i, \quad i = 1, \ldots, m, \\ & X \in K \end{array}$$

where $K = K_1 \oplus K_2 \oplus \cdots \oplus K_r$ and K_l is the cone of $n_l \times n_l$ symmetric positive semidefinite matrices, $C, A_i \in \Re^{n \times n}$ are symmetric, and $A \bullet C = \text{tr }(A^T C)$.

The dual of (SDP) can be written as:

$$\text{(DSP)} \qquad \begin{array}{ll} \text{Maximize} & b^T y \\ \text{Subject to} & \displaystyle\sum_{i=1}^{m} y_i A_i + S = C, \quad S \in K, \end{array}$$

where $y \in \Re^m$.

The well known Slater condition states that if both (SDP) and (DSP) have a relative interior, then both problems have a solution and their solutions are equal. If (X, y, S) solves the positive semidefinite programs, the complementarity condition that $XS = 0$ also holds.

There are some very strong connections between positive semidefinite programming and combinatorial optimization. The famous Lovasz number, which provides an upper bound to the maximum stable set of a graph and a lower bound to its chromatic number, is the solution to a positive semidefinite program. Many more combinatorial problems have can be relaxed into a positive semidefinite program. As opposed to linear programming relaxations, SDP relaxations offer rounding techniques that are guaranteed to be within a specified fraction of optimality. Goemans and Williamson[19], in a now classic result, applied the solution of a maximum cut positive semidefinite relaxation to a randomized algorithm and proved that the answers it generates have an expectation greater than 0.878 of optimality. More recently, provably good approximation algorithms using a positive semidefinite relaxation have been found for MAX-3-SAT, MAX-4-SAT, MAX k-CUT[35], MAX-3-CSP, scheduling, and minimum bandwidth, graph bisection, bound constrained quadratic programming[40],[31], graph coloring[22] problems.

Much like the vertex cover formulation of Kleinberg and Goemans [23] the SDP relaxation of the MSS problem will assign each vertex an integer value of -1 or $+1$. One of the two sets will be a stable set. Given a graph G with $n-1$ vertices, add an artificial vertex v_n with no edges connecting it to other vertices. Since the artificial vertex is obviously a member of the maximal stable set of the new graph, it will used to identify the stable set and enforce the constraints of the problem. The MSS problem can be stated as:

$$
\text{(MSS)} \qquad
\begin{array}{ll}
\text{Maximize} & \dfrac{1}{2}\left(\displaystyle\sum_{i=1}^{n-1} v_i^2 + v_n v_i\right) \\[2ex]
\text{Subject to} & v \in \{-1,1\}^n, \\[1ex]
 & |v_i + v_j + v_n| = 1 \text{ if } (v_i, v_j) \in \mathrm{E}
\end{array}
$$

Denoting $e_{i,j,n} \in \Re^n$ as the vector with zeros at all indices except i, j, and n, whose elements equal 1, the positive semidefinite relaxation of MSS is

$$
\text{(MSSSDP)} \qquad
\begin{array}{ll}
\text{Maximize} & \begin{pmatrix} .5 & & & .25 \\ & \ddots & & \vdots \\ & & .5 & .25 \\ .25 & \cdots & .25 & 0 \end{pmatrix} \bullet X \\[4ex]
\text{Subject to} & diag(X) = e, \\[1ex]
 & (e_{i,j,n} e_{i,j,n}^T) \bullet X = 1 \qquad \forall (i,j) \in \mathrm{E} \\
 & X \succeq 0
\end{array}
$$

Imposing the additional constraint upon (MSSSDP) that the matrix X have rank one would make it equivalent to (MSS). Relaxing this constraint to include all symmetric positive semidefinite matrices makes the problem linear, and the solution to this problem provides an upper bound to the integer program (MSS). A randomized algorithm uses a solution the relaxed problem, X^*, to identify stable sets. The randomized algorithm goes as follows:

1. Given a solution X^* to (MSSSDP), find a V such that $X^* = V^T V$.
2. Select a unit vector $u \in \Re^n$ from the unit sphere and let $v = \text{sign}(V^T u)$.
3. For each $(i, j) \in$ E, if $|v_i + v_j + v_n| \neq 1$, change the sign of either v_i or v_j.

The stable set will be the set of vertices with the same sign as v_n. For arbitrary graphs, the constraints corresponding to the edges of the graph will be satisfied with a frequency greater than 91% []. The third step of the randomized algorithm ensures that no edge connects vertices in the set by selectively removing vertices from the set. The choice of whether to switch vertices v_i or v_j may be arbitrary, but a better choice may be made by switching the vertex whose value is farthest from v_n: if $|v_i - v_n| > |v_j - v_n|$, change the sign of v_i, otherwise change the sign of v_j. This randomized algorithm can be applied multiple times to calculate multiple stable sets.

Like in the linear programming relaxation of the maximal stable set problem, larger cliques can be incorportated into the formulation. When present and previously identified, these larger cliques provide a much tighter approximation to the convex hull of the combinatorial problem than the formulation that accounts only for edges. Given cliques $\mathcal{C}^1, \ldots, \mathcal{C}^d$, such that $\mathcal{C}^k$ has n_k vertices, stable sets $v \in \{-1, 1\}^n$ must satisfy $|(n_k - 1)v_n + \sum_{v_i \in \mathcal{C}^k} v_i| = 1$ for $k = 1, \ldots, d$. Not only does this positive semidefinite formulation provide a tighter relaxation, but this program also has fewer constraints which allows it to be solved more quickly.

The vertices may also have weights w_i assigned to them. The *weighted maximal stable set* problem seeks the stable set of vertices with the largest cumulative weight. Subject to the same constraints as (MSS), this problem maximizes the function $\frac{1}{2} \sum_{i=1}^{n-1} w_i \left(v_i^2 + v_n v_i\right)$. These problems can also be relaxed into a positive semidefinite program.

For the graph coloring problem, instead of assigning colors or integers to the vertices of the graph, a unit vector $v_i \in \Re^n$ is assigned to the each of the n vertices i in V. To capture the property of coloring, the vectors of adjacent vertices should different in a natural way. Using the definition of [22], the vector k- coloring of G is an assignment of unit vectors $v_i \in \Re^n$ to each vertex i in V such that for any two adjacent vertices i and j, the dot product of the vectors satisfies the inequality $v_i^T v_j \leq -\frac{1}{k-1}$. In other words, the angle between the vectors of adjacent vertices must be sufficiently large. Define the matrix V such that column i is given by v_i and let $X = V^T V$. The matrix X is positive semidefinite and satisfies the inequalities $X_{ij} = X_{ji} \leq -\frac{1}{k-1}$ for each pair of adjacent edges (i, j). Obviously, any matrix is n-colorable, so the graph coloring problem can be posed as:

$$
\text{(COLOR)} \quad \begin{array}{lll} \text{Minimize} & \text{rank}(X) & \\ \text{Subject to} & diag(X) & = e, \\ & X_{ij} & \leq -\frac{1}{n-1} \text{ for } (i,j) \in \text{E} \\ & X \succeq 0 & \end{array}
$$

By ignoring the objective function, this problem becomes a positive semidefinite program. This program looks for a feasible point, which can be used by a heuristic to color the graph.

Let $a_{ij} \in \Re^n$ ve a vector of zeros except indices i and j, whose elements equal one. The positive semidefinite relaxation of the graph k coloring problem can be rewritten as:

$$
\text{(COLORSDP)} \quad \begin{array}{lll} \text{Minimize} & 0 \bullet X & \\ \text{Subject to} & diag(X) & = e, \\ & (a_{ij} a_{ij}^T) \bullet X & \leq 2 - \frac{2}{k-1} \text{ if } (i,j) \in \text{E} \end{array} \tag{1.1}
$$

A solution X^* with rank less than or equal to k, identifies a legal k-coloring. the problem can be solved exactly. More generally, Karger, Motwani, Sudan propose a randomized algorithm that produces a k-semicoloring, an assignment of colors with relatively few adjacent vertices with the same color. We propose a heuristic procedure for to obtain a legal coloring, albeit with more than k colors if necessary.

Coloring Algorithm For $k = 1, \ldots,$

1. Let U^k be the uncolored vertices. If U^k is empty, terminate the algorithm.

2. Sort the vertices of U^k in decreasing order of degree in $G[U^k]$, the graph induced by the uncolored vertices, and let i be the vertex with highest degree.

3. Build a vertex set W^k by examining vertices $j \in U^k$ in the decreasing order of X_{ij}. Add j to W^k if it is not adjacent to any of the vertices in W^k.

4. Assign the vertices in W^k color k.

This algorithm is a modification of the algorithm proposed by [34]. In their algorithm, only step 3 is different. Instead of using the solution to the a positive semidefinite program, they examine the vertices in decreasing order of degree in $G[U^k]$. This algorithm remains one of the simplest and most popular mothods of coloring a graph, although other heuristics have been proposed and can be modified to include information inherent in the positive semidefinite program.

3 POSITIVE SEMIDEFINITE PROGRAMMING ALGORITHMS

There are actually several polynomial algorithms that can solve positive semidefinite programs. One is the primal-scaling algorithm (Nesterov and Nemirovskii [32], Alizadeh [1], Vandenberghe and Boyd [38], and Ye [41]), which is the analogue of the primal potential reduction algorithm for linear programming. This algorithm uses X to generate the iterate direction. Another is the dual-scaling algorithm (Vandenberghe and Boyd [38], Anstreicher and Fampa [2], and Ye [41]), which is the analogue of the dual-scaling algorithm for linear programming. The dual-scaling algorithm uses only S to generate the iterate direction. The third is the primal-dual scaling algorithm which uses both X and S to generate iterate directions, including Alizadeh-Haeberly-Overton, Helmberg-Rendl-Vanderbei-Wolkowicz / Kojima-Shida-Hara / Monteiro, Nesterov-Todd, Gu, and Toh directions, as well as directions called the MTW and Half directions (see Todd [36] and references therein). All these algorithms possess $O(\sqrt{n}\log(1/\epsilon))$ iteration complexity to yield accuracy ϵ.

The features of the positive semidefinite program should determine which algorithm and which implementation of the algorithm is most appropriate. In contrast to applications of SDP in control theory and truss topology design, positive semidefinite programs arising in combinatorial optimization typically have many variables, contain sparse low rank constraint matrices, and require relatively low precision solutions. Although rank one matrices reduce the complexity of algorithms that solve positive semidefinite programs by a factor of n, not all implementations utilize this structure to reduce the complexity. Our implementation of the dual scaling algorithm explicitly accounts for these features[8]. In addition, the dual scaling algorithm has particularly low computational cost per iteration. While primal and primal dual algorithms require matrix multiplications and expensive eigenvalue or LU matrix factorizations, the dual scaling algorithm, uses only the relatively inexpensive Cholesky factorizations. Furthermore, the dual matrix S has a sparsity pattern like that of the graph's adjacency matrix; this sparsity offers the potential for additional savings in computation time and memory requirements, which the dual scaling algorithm can uniquely exploit.

One assumption for the convergence of the dual scaling algorithm is that the feasible primal region has a relative interior.

Theorem 1 *The positive semidefinite relaxation (MSSSDP) and its dual have a relative interior.*

Proof. 1 *Define the vectors $v^1, \ldots, v^{n-1}$ by*

$$v_i^j = \begin{cases} -1 & \text{if } i = j \text{ or } i = n \\ 1 & \text{otherwise} \end{cases}$$

and

$$v_i^n = \begin{cases} -1 & \text{if } 1 \leq i \leq n-1 \\ 1 & \text{if } i = n \end{cases}$$

These vectors satisfy the constraints of (MSS) and the matrices $X^i = v^i(v^i)^T$ satisfy the positive semidefinite relaxation (MSSSDP). Let

$$X = \frac{1}{n}\sum_{i=1}^{n} X^i.$$

This matrix also satisfies the constraints of (MSSSDP) since it is a convex combination of the others. This matrix can be defined explicitly by:

$$X_{i,j} = \begin{cases} 1 & \text{if } i = j \\ \frac{n-4}{n} & \text{if } i \neq j, \quad i,j \neq n \\ \frac{2-n}{n} & \text{if } i \neq j, \quad i = n \text{ or } j = n \end{cases}$$

For $n = 2, 3, 4$, routine linear algebra proves that this matrix is positive definite. For $n > 4$,

$$X = aa^T + \frac{4}{n}I - \left(\frac{4}{n(n-4)} + \frac{4}{n}\right) e_n e_n^T$$

where I is the identity matrix, e_n is the n^{th} unit vector, and

$$a_i = \begin{cases} \sqrt{\frac{n-4}{n}} & 1 \leq i \leq n-1 \\ \frac{1-n}{\sqrt{n}\sqrt{n-4}} & i = n \end{cases}$$

The matrix X is the sum of two rank one matrices and a multiple of the identity matrix. Define an orthonormal matrix Q such that the first column, $Q_{:,1} = e_n$ and the second column $Q_{:,2} = \left[\sqrt{\frac{n-4}{n}} \ \ldots \ \sqrt{\frac{n-4}{n}} \ 0\right]^T$. Let the other rows be any set of unit vectors that are normal to each other and the first two columns.

Apply an orthogonal transformation to X, defining $\hat{X} = Q^T X Q$. Matrices X and $\hat{X}$ have identical eigenvalues, so X is positive definite if and only if $\hat{X}$ is positive definite. Written explicitly,

$$\hat{X}_{i,j} = \begin{cases} 1 & \text{if } i = j = 1 \\ \frac{(n-1)(n-4)}{n} + \frac{4}{n} & \text{if } i = j = 2 \\ \frac{2-n}{\sqrt{n}\sqrt{n-4}}\sqrt{\frac{(n-1)(n-4)}{n}} & \text{if } i + j = 3 \\ \frac{4}{n} & \text{if } i = j > 2 \\ 0 & \text{otherwise} \end{cases}$$

$$\det \hat{X}_{1:2,1:2} = \frac{(n-4)(n-1)}{n} + \frac{4}{n} - \frac{(2-n)^2}{n(n-4)}\frac{(n-1)(n-4)}{n} = \frac{4}{n^2} > 0$$

The upper left submatrices all have positive determinants, so $X \succ 0$.

In the dual of (MSSSDP)

$$S = C - \mathrm{Diag}(y) - \sum_{(i,j)\in \mathrm{E}} (e_{i,j,n} e_{i,j,n}^T)\lambda_{ij}$$

where $y \in \Re^n$ and $\lambda \in \Re^{|E|}$. By setting $\lambda = 0$ and the elements of $y \in \Re^n$ to be negative with sufficiently large magnitude, S will be a diagonally dominant and positive definite matrix.

Because (COLORSDP) also has constraints that bound its diagonal, its dual variables can also be set to create a diagonally dominant positive semidefinite matrix. In fact, the DSDP software uses this technique to find initial dual starting points. Although it is possible for (COLORSDP) to lack a relative interior, the presence of a dual relative interior implies that there is no duality gap between the primal and dual positive semidefinite programs; the objective values of the dual points will converge to zero.

4 COMPUTATIONAL RESULTS

In our computational experiments, we used a variety of previously tested graphs drawn from a large number of sources. For each of these graphs, we formulated the positive semidefinite relaxation of the integer combinatorial problem and solved the relaxation until a relative duality gap of 10^{-3} has been achieved.

For the maximum stable set problems, most of the graphs are taken from the 2nd DIMACS Challenge [15]. These graphs were contributed as test problems for solving the maximum clique problem. For these graphs, we took the complement of these graphs and applied our maximum stable set algorithm. The results are supplied in Table 1.1. A second set of test problems are examples of Mycielski graphs[37]. These graphs are interesting because they contain no cliques of size larger than 2. For these graphs, we expect our relaxation to be very tight. The results are also included in Table 1.1.

Table 1.1 Maximum Stable Set Problems

Graph	$\|V\|$	$\|E\|$	SDP	Optimal	DSDP
hamming10-2	1024	5120	512.1	512	512
hamming6-2	64	192	32.0	32	32
hamming6-4	64	1312	5.35	4	4
hamming8-2	256	1024	128.0	128	128
johnson16-2-4	120	1600	8.0	8	8
johnson8-2-4	28	168	4.0	4	4
san200_0.9_1	200	1990	70.0	70	70
san200_0.9_2	200	1990	60.0	60	60
san200_0.9_3	200	1990	44.1	44	44
sanr200_0.7	200	6032	23.9	18	11
sanr200_0.9	200	2037	49.3	?	34
brock200_1	200	5066	27.5	21	14
myciel3	11	20	5.0	5	5
myciel4	23	71	11.0	11	11
myciel5	47	236	23.0	23	23
myciel6	95	755	47.0	47	47
myciel7	191	2360	95.0	95	95

For each graph we solved the semidefinite relaxation (MSSSDP), and applied the randomized procedure for finding stable sets. Since the time required by the randomization algorithm is very small relative to the time spent on solving the positive semidefinite program, we applied the randomized procedure n times on each problem. The data in Table 1.1 includes the number of vertices ($|V|$) and edges ($|E|$) in each graph, the upper bound provided by the semidefinite relaxation (SDP), the size of the maximum stable set (Optimal), and the size the the largest stable set we found (DSDP).

Of the 17 graphs, we found a maximum stable set exactly 14 times. In 13 of those 14 instances, the positive semidefinite relaxtion was extremely tight. These 13 instances include the five Mycielski graphs, which have no large cliques. In two other instances, the relaxation was also not very tight and DSDP found a large, although not maximum, stable set. In the final instance, sanr200_0.9, a solution to (MSS) is not known. Our relaxation is not particularly close the size of our largest stable set, and we assume our solution is not optimal. This evidence demonstrates the importance of using large cliques when such knowledge is available.

For the graph coloring problem, we used examples collected by Trick and Mehrotra [37]. Some of these graphs (mulsol, zeroin) were taken from problems in register allocation. The other graphs were created from characters in books, cities on maps, chessboards, and a college football schedule. For these problems, we formulated and solved the positive semidefinite relaxation of the n-coloring problem (COLORSDP). From this solution, we applied the graph coloring heuristic to obtain one graph coloring. Table 1.2 shows the minimal number of colors used, the number of colors we used in DSDP, and the number of colors used by the heuristic [34].

Of these test problems, the optimal coloring is known for 34 of them. In 24 of these 34 problems, we correctly identified an optimal coloring of the graph. Although the heuristic also found an optimal coloring in many of these graphs, problem queen5.5, utilized the solution to the positive semidefinite program to find an optimal coloring which the heuristic could not do. In a total of four problems, the coloring obtained using the SDP relaxation was better than the coloring obtained by the heuristic, and in no case did the our method do worse than the heuristic alone.

There were seven graph for which we did not find an optimal coloring. We tried again to find an optimal coloring for these graphs by solving a tighter positive semidefinite relaxation. Instead of using the n-color formulation, we used the k-color formulation where k is the minimal graph coloring. We solved these tighter relaxations and applied our heuristic to these solutions, hoping to identify a better coloring. The results are in Table 1.3. The number of colors used with the tighter formulation is in the last column (DSDP2). In only one of the seven instances did the tighter formulation actually improve the coloring. On the other hand, there was one instance where the tighter formulation

Table 1.2 Graph Coloring Problems

Graph	$\|V\|$	$\|E\|$	Optimal	DSDP	Heuristic
anna	138	493	11	11	11
david	87	406	11	11	11
homer	561	1629	13	13	13
huck	74	301	11	11	11
jean	80	254	10	10	10
games120	120	638	9	9	9
miles250	128	387	8	8	8
miles500	128	1170	20	20	20
miles750	128	2113	31	32	32
miles1000	128	3216	42	42	42
miles1500	128	5198	73	73	73
queen5.5	25	160	5	5	7
queen6.6	36	290	7	9	9
queen7.7	49	476	7	11	11
queen8.8	64	728	9	11	12
queen9.9	81	1055	10	13	13
queen10.10	100	1470	?	14	14
queen11.11	121	1980	11	15	15
queen12.12	144	2596	?	17	17
queen13.13	169	3328	13	18	18
queen14.14	196	4186	?	19	19
myciel3	11	20	4	4	4
myciel4	23	71	5	5	5
myciel5	47	236	6	6	6
myciel6	95	755	7	7	7
myciel7	191	2360	8	8	8
zeroin.i.1	211	4100	49	49	49
zeroin.i.2	211	3541	30	30	30
zeroin.i.3	206	3540	30	30	30
mulsol.i.1	197	3925	49	49	49
mulsol.i.2	188	3885	31	31	31
mulsol.i.3	184	3916	31	31	31
mulsol.i.4	185	3946	31	31	31
mulsol.i.5	186	3973	31	31	31
DSJC125.1	125	736	?	6	7
DSJC125.5	125	3891	?	21	22
DSJC250.1	250	3218	?	11	11
DSJR500.1	500	3555	?	13	13

Table 1.3 Graph Coloring Problems with a Tighter Relaxation

Graph	$\lvert V \rvert$	$\lvert E \rvert$	Optimal	DSDP	DSDP2
miles750	128	2113	31	32	32
queen6.6	36	290	7	9	9
queen7.7	49	476	7	11	10
queen8.8	64	728	9	11	12
queen9.9	81	1055	10	13	13
queen11.11	121	1980	11	15	15
queen13.13	169	3328	13	18	18

actually worsened the coloring of the graph. This k-coloring formulation is only slightly tighter that the n-coloring formulation, which explains why the results were only slightly different. The evidence suggests that the n-coloring relaxation is sufficient.

The time required to solve these problems ranged from less than a second for queen5.5 to over twelve hours to find the maximum stable set of brock200_1. Each iteration of the current positive semidefinite algoritms require the factorization of a dense matrix of size $|V| + |E|$. These factorizations dominate the computation time and limit the size of problems that can be solved using this technique.

Nonetheless, the positive semidefinite relaxation generates high quality solutions. We have applied the theory developed for the maximum cut problem and applied it to the maximum stable set and minimum graph coloring problems. The $|E|$ constaints in (MSS) with three nonzeros will be satisfied at least 91% of the time. In fact, very few vertices needed to change signs in order to satisfy the constraints. When the relaxation was tight, as it was in 13 of the problems, each application of the randomized algorithm changed the sign of at most one vertex. In all graphs, the number of vertices who changed signs averaged less than four. These small numbers imply very little modifications to classic Goemans and Williamson algorithm were needed to create stable sets and answers provides by these algorithms can be expected to be very high. In the minimum graph coloring problem, we have also shown that the positive semidefinite relaxation contains valuable information that can be used in conjuction with other heuristics. Our experiments contribute to the growing mountain of evidence demonstrating the high quality of solutions that can be obtained using the semidefinite relaxation.

References

[1] F. Alizadeh. *Combinatorial optimization with interior point methods and semidefinite matrices.* PhD thesis, University of Minnesota, Minneapolis, MN, 1991.

[2] K. M. Anstreicher and M. Fampa. A long-step path following algorithm for semidefinite programming problems. Working Paper, Department of Management Science, The University of Iowa, Iowa City, IA, 1996.

[3] L. Babel. Finding maximum cliques in arbitrary and in special graphs. *Computing*, 46:321–341, 1991.

[4] L. Babel and G. Tinhofer. A branch and bound algorithm for the maximum clique problem. *J. of Global Optimization*, 4, 1994.

[5] Egon Balas and H. Samuelsson. A node covering algorithm. *Naval Research Logistics Quarterly*, 24(2):213–233, 1977.

[6] Egon Balas and Jue Xue. Minimum weighted coloring of triangulated graphs, with application to maximum weight vertex packing and clique finding in arbitrary graphs. *SIAM Journal on Computing*, 20(2):209–221, April 1991.

[7] Egon Balas and Chang Sung Yu. Finding a maximum clique in an arbitrary graph. *SIAM Journal on Computing*, 15(4):1054–1068, November 1986.

[8] S. Benson, Y. Ye, and X. Zhang. Solving large–scale sparse semidefinite programs for combinatorial optimization. Technical report, Department of Management Science, University of Iowa, Iowa City, IA 52242, USA, September 1997. To appear in *SIAM J. of Optimization.*

[9] C. Berge. *Graphs and Hypergraphs.* North–Holland, Amsterdam, 1973.

[10] P. Briggs, K. Cooper, K. Kennedy, and L. Torczon. Coloring heuristics for register allocation. In *ASCM Conference on Program Language Design and Implementation*, pages 275–284. The Association for Computing Machinery, 1989.

[11] R. Carrahan and P. M. Pardalos. An exact algorithm for the maximum clique problem. *Operations Research Letters*, 9:375–382, 1990.

[12] G.J. Chaitin, M. Auslander, A.K. Chandra, J. Cocke, M.E. Hopkins, and P. Markstein. Register allocation via coloring. *Computer Languages*, 6:47–57, 1981.

[13] Gregory J. Chaitin. Register allocation and spilling via graph coloring. *SIGPLAN Notices (Proceedings of the SIGPLAN '82 Symposium on Compiler Construction, Boston, Mass.)*, 17(6):98–101, June 1982.

[14] D. De Werra. An introduction to timetabling. *European Journal of Operations Research*, 19:151–162, 1985.

[15] DIMACS Center Web Page. The Second DIMACS Implementation Challenge: 1992-1993. ftp://dimacs.rutgers.edu/pub/challenge/graph/benchmarks/clique/.

[16] C. Friden, A. Hertz, and D. de Werra. An exact algorithm based on tabu search for finding a maximum inderpendent set in a graph. *Computers Operations Research*, 17(5):375–382, 1990.

[17] Andreas Gamst. Some lower bounds for a class of frequency assignment problems. *IEEE Transactions of Vehicular Technology*, 35(1):8–14, 1986.

[18] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP–Completeness.* W.H Freeman, San Francisco, CA, 1979.

[19] M. X. Goemans and D. P. Williamson. .878-approximation for MAX CUT and MAX 2SAT. In *Proc. 26$^{\text{th}}$ ACM Symp. Theor. Computing*, pages 422–431, 1994.

[20] David S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.

[21] David S. Johnson. Worst case behavior of graph coloring algorithms. In *Proceedings of 5th Southeastern Conference on Combinatorics, Graph Theory, and Computing*, pages 513–527. Utilitas Mathematica, Winnipeg, Canada, 1974.

[22] David Karger, Rajeev Motwani, and Madhu Sudan. Approximate graph coloring by semidefinite programming. Technical report, MIT, Cambridge, MA 52242, USA, 1994.

[23] Jon Kleinberg and Michel X. Goemans. The lovász theta function and a semidefinite programming relaxation of vertex cover. *SIAM Journal on Discrete Mathematics*, 11(2):196–204, May 1998.

[24] M. Kubale and B. Jackowski. A generalized implicit enumeration algorithm for graph coloring. *Communications of the ACM*, 28:412–418, 1985.

[25] M. Kubale and E. Kusz. Computational experience with implicit enumeration algorithms for graph coloring. In *Proceedings of the WG'83 International Workshop on Graphtheoretic Concepts in Computer Science*, pages 167–176, Linz, 1983. Trauner Verlag.

[26] Carlo Mannino and Antonio Sassano. An exact algorithm for the maximum cardinality stable set problem. *Networks*, page (submitted), 1993. ftp://dimacs.rutgers.edu/pub/challenge/graph/ contributed.

[27] Anuj Mehrotra and Michael A. Trick. A column generation approach for graph coloring. http://mat.gsia.cmu.edu/trick.html, April 1995.

[28] Craig A. Morgenstern and Harry D. Shapiro. Coloration neighborhood structures for general graph coloring. In *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco, Jan, 1990*. Society for Industrial and Applied Mathematics, Philadelpia, 1990.

[29] George L. Nemhauser and G. L. Sigismondi. A strong cutting plane / branch and bound algorithm for node packing. *Journal of the Operational Research Society*, 43(5), 1992.

[30] George. L. Nemhauser and Les. E. Trotter, Jr. Vertex packings: Structural properties and algorithms. *Mathematical Programming*, 8(2):232–248, 1975.

[31] Yu. E. Nesterov. Quality of semidefinite relaxation for nonconvex quadratic optimization. In *CORE Discussion Paper, #9719*, Belgium, March 1997.

[32] Yu. E. Nesterov and A. S. Nemirovskii. *Interior Point Polynomial Methods in Convex Programming: Theory and Algorithms.* SIAM Publications, SIAM, Philadelphia, 1993.

[33] B. Pittel. On the probable behaviour of some algorithms for finding the stability number of a graph. *Mathematical Proceedings of the Cambridge Philosophical Society*, 92:511–526, 1982.

[34] M. J. D. Powell and P. L. Toint. On the estimation of sparse hessian matrices. *SIAM Journal on Numerical Analysis*, 16:1060–1074, 1979.

[35] Proc. 4th IPCO Conference. *Improved approximation algorithms for max k-cut and max bisection*, 1995.

[36] M. J. Todd. On search directions in interior-point methods for semidefinite programming. Technical Report 1205, School of Operations Research and Industrial Engineering, Cornell University, Ithica, NY 14853–3801, October 1997.

[37] M. Trick. Graph coloring instances. http://mat.gsia.cmu.edu/COLOR/instances.html.

[38] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[39] D. C. Wood. A technique for coloring a graph applicable to large scale time-tabling problems. *The Computer Journal*, 3:317–319, 1969.

[40] Y. Ye. Approximating quadratic programming with quadratic constraints. Working Paper, Department of Management Science, The University of Iowa, Iowa City, 1997.

[41] Y. Ye. *Interior Point Algorithms : Theory and Analysis.* Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, New York, 1997.