A New Trust Region Technique for the Maximum Weight Clique Problem *

Stanislav Busygin
busygin@a-teleport.com
WWW: http://www.busygin.dp.ua

Revised: September 14, 2002

Abstract

A new simple generalization of the Motzkin–Straus theorem for the maximum weight clique problem is formulated and directly proved. Within this framework a new trust region heuristic is developed. In contrast to usual trust region methods, it regards not only the global optimum of a quadratic objective over a sphere, but also a set of other stationary points of the program. We formulate and prove a condition when a Motzkin–Straus optimum coincides with such a point. The developed method has complexity $O(n^3)$, where n is the number of graph vertices. It was implemented in a publicly available software package *QUALEX-MS*.

Computational experiments evidence that the algorithm is exact on small graphs and exceptionally efficient on DIMACS benchmark graphs and various random maximum weight clique problem instances.

Keywords: maximum weight clique, continuous approach, Motzkin-Straus theorem, quadratic programming, heuristics, trust region, algorithms, NP-hard.

1 Introduction

Let $G(V, E)$ be a simple undirected graph, $V = \{1, 2, \ldots, n\}$. The *adjacency matrix* of G is a matrix $A_G = (a_{ij})_{n \times n}$, where $a_{ij} = 1$ if $(i, j) \in E$, and $a_{ij} = 0$ if $(i, j) \notin E$. The set of vertices adjacent to a vertex $i \in V$ will be denoted by $N(i) = \{j \in V : \ (i, j) \in E\}$ and called the *neighborhood* of the vertex i. A *clique* Q is a subset of V such that any two vertices of Q are adjacent. The maximum clique problem asks for a clique of the maximum cardinality. This cardinality is called the *clique number* of the graph and denoted by $\omega(G)$.

Next, we associate with each vertex $i \in V$ a positive number w_i called the vertex *weight*. This way, along with the adjacency matrix A_G, we consider the vector of vertex weights $w \in \mathbb{R}^n$. The total weight of a vertex subset $S \subseteq V$ will be denoted by

$$W(S) = \sum_{i \in S} w_i.$$

The maximum weight clique problem asks for a clique Q of the maximum $W(Q)$ value. We denote this value by $\omega(w, G)$.

Both the maximum cardinality and the maximum weight clique problems are NP-hard [2], so it is considered unlikely that an exact polynomial time algorithm for them exists. Approximation

of large cliques is also hard. It was shown in [14] that unless $NP = ZPP$ no polynomial time algorithm can approximate the clique number within a factor of $n^{1-\epsilon}$ for any $\epsilon > 0$. Recently this margin was tightened in [15] to $n/2^{(\log n)^{1-\epsilon}}$. Hence, any heuristic algorithm nicely approximating maximum clique in a short time in practice is a significant achievement. In this paper we present such an algorithm named *QUALEX-MS* (QUick ALmost EXact Motzkin–Straus-based.) A software package implementing the algorithm is available at [19].

The paper is organized as follows. In Section 2 we revise the Motzkin–Straus theorem to use the quadratic programming formulation for the maximum weight clique problem. Section 3 reviews the trust region problem and finding its stationary points. In Section 4 we provide a theoretical result connecting the trust region stationary points with maximum clique finding and formulate the QUALEX-MS method itself. Section 5 describes computational experiments with the algorithm and their results. In the final Section 6 we make some conclusions and outline further research work.

2 The Motzkin–Straus Theorem for Maximum Clique and Its Generalization

In 1965 Motzkin and Straus formulated the maximum clique problem as a certain quadratic programming over a simplex [11].

Theorem 1 (Motzkin–Straus) *The global optimum value of the quadratic program*

$$\max f(x) = \frac{1}{2} x^T A_G x \tag{1}$$

subject to

$$\sum_{i \in V} x_i = 1, \; x \geq 0 \tag{2}$$

is

$$\frac{1}{2}\left(1 - \frac{1}{\omega(G)}\right). \tag{3}$$

See [3] for a recent direct proof.

We formulate a simple generalization of this result for the maximum weight clique problem. In contrast to the generalization established in [6], this one does not require any reformulation of the maximum clique quadratic program to another minimization problem. It maximally preserves the form of the original Motzkin–Straus result.

Let $w_{\min}$ be the smallest vertex weight existing in the graph. We introduce a vector $d \in \mathbb{R}^n$ such that

$$d_i = 1 - \frac{w_{\min}}{w_i}.$$

Theorem 2 *The global optimum value of the quadratic program*

$$\max f(x) = x^T (A_G + diag(d_1, \ldots, d_n))x \tag{4}$$

subject to

$$\sum_{i \in V} x_i = 1, \; x \geq 0 \tag{5}$$

is

$$1 - \frac{w_{\min}}{\omega(w, G)}. \tag{6}$$

At that, there is a global optimum, where the set of nonzero variables designates a maximum weight clique and the value of each of those variables is

$$x_i = \frac{w_i}{\omega(w, G)}. \tag{7}$$

We give a direct proof of this theorem similar in spirit to the proof in [3].

Proof. At first, we show that there is an optimum such that if $x_i > 0$ and $x_j > 0$ then $(i, j) \in E$. Indeed, let $(i, j) \notin E$. We may partition the objective into items dependent on x_i, items dependent on x_j, and the other items:

$$\begin{aligned} f_i(x) &= d_i x_i^2 + 2x_i \sum_{k \in N(i)} x_k, \\ f_j(x) &= d_j x_j^2 + 2x_j \sum_{k \in N(j)} x_k, \\ \overline{f}_{ij}(x) &= f(x) - f_i(x) - f_j(x). \end{aligned}$$

Consider the partial derivatives of the objective on x_i and x_j:

$$\begin{aligned} \frac{\partial f}{\partial x_i} &= \frac{\partial f_i}{\partial x_i} = 2d_i x_i + 2 \sum_{k \in N(i)} x_k, \\ \frac{\partial f}{\partial x_j} &= \frac{\partial f_j}{\partial x_j} = 2d_j x_j + 2 \sum_{k \in N(j)} x_k. \end{aligned}$$

Without loss of generality, let $\partial f_i / \partial x_i \geq \partial f_j / \partial x_j$. Consider a point x^* such that $x_i^* = x_i + x_j$, $x_j^* = 0$, and $x_k^* = x_k$ for all $k \in V$, $i \neq k \neq j$. It is easy to see that x^* obeys the constraints (5). $\overline{f}_{ij}(x^*) = \overline{f}_{ij}(x)$ and $f_j(x^*) = 0$ now, so we evaluate $f_i(x^*)$ and compare it with $f_i(x) + f_j(x)$. In the computations below we take into account that d_i and d_j are always nonnegative.

$$f_i(x^*) = d_i(x_i + x_j)^2 + 2(x_i + x_j) \sum_{k \in N(i)} x_k =$$

$$= f_i(x) + 2d_i x_i x_j + d_i x_j^2 + 2x_j \sum_{k \in N(i)} x_k =$$

$$= f_i(x) + x_j \left(2d_i x_i + 2 \sum_{k \in N(i)} x_k \right) + d_i x_j^2 =$$

$$= f_i(x) + x_j \frac{\partial f_i}{\partial x_i} + d_i x_j^2 \geq f_i(x) + x_j \frac{\partial f_j}{\partial x_j} + d_i x_j^2 \geq$$

$$\geq f_i(x) + x_j \frac{\partial f_j}{\partial x_j} = f_i(x) + 2d_j x_j^2 + 2x_j \sum_{k \in N(j)} x_k \geq$$

$$\geq f_i(x) + d_j x_j^2 + 2x_j \sum_{k \in N(j)} x_k = f_i(x) + f_j(x).$$

Thus, the objective value $f(x^*)$ is not less than $f(x)$. This means that if x is an optimum, all $\geq$ signs above must be equalities and x^* is an optimum as well. So, there is an optimum such that the vertices corresponding to nonzero variables form a clique.

Next, we show that the variable values in the optimum must be proportionate to the vertex weights. Let $Q \subseteq V$ be the clique designated by the nonzero variables. In the subspace $\{x_i\} : i \in Q$ we have the program:

$$\max f(x) = \sum_{i \in Q} d_i x_i^2 + \sum_{i \in Q} \sum_{\substack{j \in Q \\ j \neq i}} x_i x_j \tag{8}$$

subject to

$$\sum_{i \in Q} x_i = 1.$$

The objective may be transformed to

$$\left(\sum_{i \in Q} x_i \right)^2 - \sum_{i \in Q} \frac{w_{\min} x_i^2}{w_i}.$$

The first item equals 1 due to the constraint, so we may consider an equivalent program:

$$\sum_{i \in Q} \frac{x_i^2}{w_i} \to \min$$

The Lagrangian of the program is

$$\sum_{i \in Q} \frac{x_i^2}{w_i} + \lambda \left(\sum_{i \in Q} x_i - 1 \right).$$

It is easy to see it has the only stationary point

$$x_i = \frac{w_i}{W(Q)},\ i \in Q; \quad \lambda = \frac{2}{W(Q)}.$$

and this point is the minimum.

Evaluate the objective $f(x)$. It is

$$1 - \sum_{i \in Q} \frac{w_{\min} x_i^2}{w_i} = 1 - \sum_{i \in Q} \frac{w_{\min} w_i^2}{w_i (W(Q))^2} = 1 - \frac{w_{\min}}{W(Q)}.$$

This value is largest when $W(Q)$ is largest, so the objective has an optimum when Q is a maximum weight clique.

$$\max f(x) = 1 - \frac{w_{\min}}{\omega(w, G)},\ x_i = \frac{w_i}{\omega(w, G)} \text{ if } i \in Q,\ x_i = 0 \text{ otherwise}$$

at that. QED.

For development of our method we will use a rescaled form of the quadratic program (4,5). First of all, for the graph $G(V, E)$ with the vertex weights w define the *weighted adjacency matrix* $A_G^{(w)} = (a_{ij}^{(w)})_{n \times n}$ such that

$$a_{ij}^{(w)} = \begin{cases} w_i - w_{\min}, & \text{if } i = j \\ \sqrt{w_i w_j}, & \text{if } (i, j) \in E \\ 0, & \text{if } i \neq j \text{ and } (i, j) \notin E. \end{cases} \tag{9}$$

Obviously, it is the ordinary adjacency matrix when all vertex weights are ones. Next, we introduce the vector of vertex weight square roots

$$z \in \mathbb{R}^n : \ z_i = \sqrt{w_i}. \tag{10}$$

The rescaled formulation is given in the following corollary of Theorem 2.

Corollary 1 *The global optimum value of the quadratic program*

$$\max f(x) = x^T A_G^{(w)} x \tag{11}$$

subject to

$$z^T x = 1, \ x \geq 0 \tag{12}$$

is

$$1 - \frac{w_{\min}}{\omega(w, G)}.$$

At that, there is a global optimum, where the set of nonzero variables designates a maximum weight clique and the value of each of those variables is

$$x_i = \frac{z_i}{\omega(w, G)}. \tag{13}$$

Proof. Perform the variable scaling $x_i \to \sqrt{w_i} x_i$ in the formulation of Theorem 2. The corollary is obtained immediately. QED.

A useful property of the rescaled formulation is that optima corresponding to all maximum weight cliques are located at the same distance from the origin. Now we designate this fact formally.

Definition 1 *An* indicator *of a clique $Q \subseteq V$ is a vector $x^Q \in \mathbb{R}^n$ such that*

$$x_i^Q = \begin{cases} z_i / W(Q), & \text{if } i \in Q \\ 0, & \text{if } i \in V \setminus Q. \end{cases}$$

Fact 1 *All cliques of the same weight σ have indicators located at the same distance $1/\sqrt{\sigma}$ from the origin.*

Proof. It follows immediately that the indicator of a clique $Q \subseteq V$ with the weight $W(Q) = \sigma$ is a vector of the length

$$\sqrt{\sum_{i \in Q} (z_i / \sigma)^2} = \sqrt{W(Q)} / \sigma = 1 / \sqrt{\sigma}.$$

QED.

We may notice here that heavier cliques have indicators located closer to the origin. The indicators of the maximum weight cliques have the smallest radius, namely, $1/\sqrt{\omega(w, G)}$. The idea of our method is to replace the nonnegativity constraint $x \geq 0$ in (12) by a ball constraint $x^T x \leq r^2$ of a radius $r \approx 1/\sqrt{\omega(w, G)}$ and to regard the stationary points of this new program as vectors significantly correlating with the maximum weight clique indicators. In the next section we outline polynomial time finding of stationary points of a quadratic on a sphere. In our case this technique can be used after the objective is orthogonally projected onto the hyperplane $z^T x = 1$, so this equality may be removed from the constraints. In the subsequent section we give a substantiation of the used constraint replacement proving a particular case when a spheral stationary point is exactly an optimum of the program (11,12) and formulate the algorithm itself.

3 The Trust Region Problem

The trust region problem is optimization of a quadratic function subject to a ball constraint. The term originates from a nonlinear programming application of this problem. Namely, to improve a feasible point, a small ball – *trust region* – around the point is introduced and a quadratic approximation of the objective is optimized in it. Then, if the objective approximation is good enough within this locality, the ball optimum of the quadratic is very close to the optimum of the objective there, and it may be taken as the next improved feasible solution. This technique is very attractive in many cases since the optimization of a quadratic function over a sphere is polynomially solvable in contrast to general nonconvex programming [17]. There is a vast range of other sources describing theoretical and practical results on the trust region problem [5, 7, 8, 10]. Here we outline the complete diagonalization method deriving not only the global optimum at a given sphere radius, but all stationary points corresponding to particular radii we want to consider. That is, the radius value remains non-fixed up to a final step when it appears as a parameter of a univariate equation determining the stationary points. We note that for our application we are interested in hyperbolic objectives only, so interior stationary points never exist.

Thus, consider finding of stationary points for a function

$$f(x) = x^T A x + 2b^T x \tag{14}$$

$$\text{s.t.} \sum_{i=1}^{n} x_i^2 = r^2,$$

where A is a given real symmetric $n \times n$ matrix, $b \in \mathbf{R}^n$ is a given vector, and $x \in \mathbf{R}^n$ is the vector of variables. At first, we diagonalize the quadratic form in (14) performing eigendecomposition of A:

$$A = Q\text{diag}(\lambda_1, \ldots, \lambda_n)Q^T,$$

where Q is the matrix of eigenvectors (stored as columns) and the eigenvalues $\{\lambda_i\}$ has nondecreasing order. In the eigenvector basis, (14) is

$$f(y) = \sum_{i=1}^{n} \lambda_i y_i^2 + 2\sum_{i=1}^{n} c_i y_i, \tag{15}$$

$$\sum_{i=1}^{n} y_i^2 = r^2. \tag{16}$$

At that

$$x = Qy,\ y = Q^T x,\ b = Qc,\ c = Q^T b. \tag{17}$$

The Lagrangian of (15, 16) is

$$L(y, \mu) = \sum_{i=1}^{n} \lambda_i y_i^2 + 2\sum_{i=1}^{n} c_i y_i - \mu \left(\sum_{i=1}^{n} y_i^2 - r^2 \right). \tag{18}$$

μ is the lagrangian multiplier of the spheral constraint here. We take it with negative sign for the sake of convenience. The stationary conditions are

$$\frac{\partial L}{\partial y_i} = 0,\ \frac{\partial L}{\partial \mu} = 0.$$

So,

$$\frac{\partial L}{\partial y_i} = 2(\lambda_i - \mu)y_i + 2c_i = 0,$$

and assuming $\mu \neq \lambda_i$,

$$y_i = \frac{c_i}{\mu - \lambda_i}. \tag{19}$$

Substituting (19) into the spheral constraint (16), we get

$$\sum_{i=1}^{n} \frac{c_i^2}{(\mu - \lambda_i)^2} - r^2 = 0. \tag{20}$$

The left-hand side of (20) is a univariate function consisting of $n + 1$ continuous and convex pieces. As all the numerators are positive, in each piece between two successive eigenvalues of A it may intersect μ-axis twice (determining two stationary points on the sphere), touch it once (determining one stationary point), or be over the axis (no stationary point corresponds to these μ values.) That depends on the chosen radius r: the greater the radius, the more cases of two spheral stationary points within one continuous piece of (20). Two outermost continuous pieces are $(-\infty;\ \lambda_1)$ and $(\lambda_n;\ +\infty)$. In each of them (20) always has one and only one root. The root in the first piece is the global minimum, the root in the second piece is the global maximum.

A degenerative case when $\mu = \lambda_i$ for some i is possible if $c_i = 0$. At that, if λ_i is a multiple eigenvalue of A, all c_j corresponding to $\lambda_j = \lambda_i$ must be equal to zero to cause the degeneration. Then all y_j such that $\mu \neq \lambda_j$ should be computed by (19), and if the sum of their squares is not above r^2, any combination of the rest entries of y obeying (16) gives a stationary point. Formally, we have a cluster of k equal eigenvalues $\lambda_i = \lambda_{i+1} = \ldots = \lambda_{i+k-1}$ and

$$c_i = c_{i+1} = \ldots = c_{i+k-1} = 0 \tag{21}$$

at that. If

$$r_0^2 = \sum_{j=1}^{i-1} y_j^2 + \sum_{j=i+k}^{n} y_j^2 \leq r^2, \tag{22}$$

where the values y_j are computed by (19) with $\mu = \lambda_i$, then any $y_i, y_{i+1}, \ldots, y_{i+k-1}$ such that

$$\sum_{j=i}^{i+k-1} y_j^2 = r^2 - r_0^2$$

provide a stationary point.

So, it is possible then that the number of stationary points is infinite. In our method we will consider in the degenerative case only such points that all but one of the entries $y_i, y_{i+1}, \ldots, y_{i+k-1}$ are zero. There are $2k$ cases:

$$\begin{gathered} y_i = \pm\sqrt{r^2 - r_0^2},\ y_{i+1} = 0,\ \ldots,\ y_{i+k-1} = 0, \\ y_i = 0,\ y_{i+1} = \pm\sqrt{r^2 - r_0^2},\ \ldots,\ y_{i+k-1} = 0, \\ \ldots \\ y_i = 0,\ y_{i+1} = 0,\ \ldots,\ y_{i+k-1} = \pm\sqrt{r^2 - r_0^2}, \end{gathered} \tag{23}$$

so an eigenvalue of multiplicity k gives $2k$ points to consider.

Finally, we note that the total complexity of the procedure above is $O(n^3)$ if we derive $O(n)$ stationary points and it takes not more than $O(n^2)$ time to get one μ value. Indeed, the eigendecomposition may be computed up to any fixed precision in $O(n^3)$ time [16], and each basis conversion in (17) takes quadratic time, so generally we have one $O(n^3)$ computation at the beginning of the procedure, and $O(n)$ computations of $O(n^2)$ complexity each afterwards.

4 The QUALEX-MS Algorithm

Thus, we will work with the program

$$\max f(x) = x^T A_G^{(w)} x \tag{24}$$
$$\text{s.t. } z^T x = 1,\ x^T x \le r^2,$$

where r is a parameter not fixed á priori. We designate now a particular case, when a stationary point of the program (24) is an optimum of the program (11,12). It happens when for each vertex outside a maximum weight clique the weight sum of adjacent vertices in the clique is constant. Namely, the following theorem holds.

Theorem 3 *Let $Q \subseteq V$ be a maximal clique of the graph $G(V,E)$ such that*

$$\forall v \in V \setminus Q:\ W(N(v) \cap Q) = C,$$

where C is some fixed value. Then the indicator x^Q of Q

$$x_i^Q = \begin{cases} z_i/W(Q), & \text{if } i \in Q \\ 0, & \text{if } i \in V \setminus Q. \end{cases}$$

is a stationary point of the program (24) when the parameter $r = 1/\sqrt{W(Q)}$.

Proof. Consider the Lagrangian of the program (24). It is

$$L(x^Q, \mu_1, \mu_2) = (x^Q)^T A_G^{(w)} x^Q + \mu_1 (z^T x^Q - 1) + \mu_2 ((x^Q)^T x^Q - r^2).$$

Its partial derivatives are

$$\frac{\partial L}{\partial x_i^Q} = 2 \sum_{i \in V} a_{ij}^{(w)} x_j^Q + z_i \mu_1 + 2 x_i^Q \mu_2 =$$

$$= 2 z_i \left(z_i x_i^Q + \sum_{j \in N(i)} z_j x_j^Q \right) - 2 w_{\min} x_i^Q + z_i \mu_1 + 2 x_i^Q \mu_2.$$

Let $i \in Q$. Then it gives

$$\frac{\partial L}{\partial x_i^Q} = 2 z_i \sum_{j \in Q} \frac{w_j}{W(Q)} - 2 w_{\min} \frac{z_i}{W(Q)} + z_i \mu_1 + \frac{2 z_i}{W(Q)} \mu_2 =$$

$$= z_i \left(2 - \frac{2 w_{\min}}{W(Q)} + \mu_1 + \frac{2 \mu_2}{W(Q)} \right).$$

Conversely, if $i \in V \setminus Q$,

$$\frac{\partial L}{\partial x_i^Q} = 2 z_i \sum_{j \in N(i) \cap Q} z_j x_j^Q + z_i \mu_1 = z_i \left(\frac{2C}{W(Q)} + \mu_1 \right).$$

We may see that in both cases the final expressions are independent of a particular i value. So, the stationary point criterion system $\partial L / \partial x_i^Q = 0$ is reduced to two equations over two variables μ_1 and μ_2. The second equation directly gives

$$\mu_1 = -\frac{2C}{W(Q)}.$$

Substituting this into the first equation, we obtain

$$\mu_2 = C + w_{\min} - W(Q).$$

So, there are values of the lagrangian multipliers satisfying the stationary point criterion. Therefore, x^Q is a stationary point of the program (24). QED.

We notice that the obtained μ_2 value is negative unless the clique Q can be made heavier by a one-to-one vertex exchange. This means that in the stationary points we are interested in the gradient of the objective is directed outside the constraining sphere. It consists with the fact that we look for a maximum of the objective.

We note a special case of Theorem 3 corresponding to the maximum cardinality clique problem.

Corollary 2 *Let $Q \subseteq V$ be a maximal clique of the graph $G(V, E)$ such that*

$$\forall v \in V \setminus Q: \ |N(v) \cap Q| = C,$$

where C is some fixed value, and all vertex weights w_i equal 1. Then the indicator x^Q of Q

$$x_i^Q = \begin{cases} 1/|Q|, & \text{if } i \in Q \\ 0, & \text{if } i \in V \setminus Q. \end{cases}$$

is a stationary point of the program (24) when the parameter $r = 1/\sqrt{|Q|}$.

Generally, optima of (11,12) cannot be found directly as stationary points of (24). However, we accept the supposition that if the parameter r is close to $1/\sqrt{\omega(w, G)}$, then the stationary points of (24), where the objective gradient is directed outside, provide significant information about maximum weight clique indicators. This may be supported by the fact that the conjunction of three imposed requirements – maximization of a quadratic form whose matrix is nonnegative, positive dot product with the positive vector z, and a rather small norm of the sought vector x – make it probabilistically more profitable to have positive entries in the vector x. So, the occured violation of the nonnegativity constraint should not be dramatic.

As the next step, we show how to reduce the program (24) to a trust region problem projecting orthogonally the objective onto the hyperplane $z^T x = 1$. At first, we move the origin into a new point

$$x^0 = z/W(V). \tag{25}$$

This point is the orthogonal projection of the origin onto the hyperplane $z^T x = 1$. That is, we introduce new variables $\hat{x} = x - z/W(V)$. This way we obtain a new program equivalent to (24)

$$\max g(\hat{x}) = \hat{x}^T A_G^{(w)} \hat{x} + 2(x^0)^T A_G^{(w)} \hat{x} \tag{26}$$

$$\text{s.t. } z^T \hat{x} = 0, \ \hat{x}^T \hat{x} \le \hat{r}^2,$$

where $\hat{r}^2 = r^2 - 1/W(V)$ (here we took into account that $(x^0)^T x^0 = 1/W(V)$.) Now the constraining equality determines a linear subspace. The orthogonal projector onto it is a matrix $P = (p_{ij})_{n \times n}$, where

$$p_{ij} = \begin{cases} 1 - w_i/W(V), & \text{if } i = j \\ -\sqrt{w_i w_j}/W(V), & \text{if } i \ne j. \end{cases}$$

Thus, the program (26) may be reformulated as

$$\max g(\hat{x}) = \hat{x}^T \hat{A} \hat{x} + 2\hat{b}^T \hat{x} \tag{27}$$

$$\text{s.t. } \hat{x}^T\hat{x} \le \hat{r}^2,$$

where $\hat{A} = PA_G^{(w)}P$ and $\hat{b}^T = (x^0)^T A_G^{(w)} P$.

This is a trust region problem – optimization of a quadratic subject to a single ball constraint. Direct matrix manipulations show that $\hat{A}$ and $\hat{b}$ can be computed by the formulas

$$\hat{a}_{ij} = a_{ij}^{(w)} - x_j^0\delta_i^{(w)} - x_i^0\delta_j^{(w)} + x_i^0 x_j^0 D \tag{28}$$

and

$$\hat{b}_i = \frac{\delta_i^{(w)} - x_i^0 D}{W(V)}, \tag{29}$$

where

$$\delta_i^{(w)} = \sqrt{w_i}(w_i - w_{\min} + \sum_{j\in N(i)} w_j) \tag{30}$$

(which are vertex degrees in the unweighted case), and

$$D = \sum_{j\in V} w_j(w_j - w_{\min}) + \sum_{(j,k)\in E} w_j w_k. \tag{31}$$

Thus, if Q is a maximum weight clique obeying Theorem 3 conditions, its indicator may be recognized by the trust region procedure described in the previous section. Generally, we will handle the maximum weight clique problem in the following way allowing us to preserve the total complexity of the method in an $O(n^3)$ time.

Before applying the trust region technique, we find a possibly best clique Q by a fast greedy procedure. To improve it, we will try to search for cliques weighting $W(Q) + w_{\min}$ at least using the stationary points of the program (24). It follows from Fact 1 that we should be interested in those points, where

$$\hat{r}^2 = \frac{1}{W(Q) + w_{\min}} - \frac{1}{W(V)} \tag{32}$$

or less. In our method we consider the stationary points having this $\hat{r}^2$ value, plus those corresponding to μ values minimizing the left hand side of (20) in each continuous section. Since heavier cliques correspond to lesser radii, we have a chance to correct the "shallowness" of the formula (32) considering the minimum possible radii. Besides, to find stationary points at any fixed radius, we need to find those minimizing μ values anyway to determine how many roots does (20) have on each continuous section. If the left hand side minimum on a continuous section is negative, there are two roots and each of them is bracketed between the minimizing point and one of the section bounds. Both univariate minimization and univariate root finding when a root is bracketed may be efficiently performed by Brent's method [1].

Next, each of the obtained stationary points is passed to a greedy heuristic as a new vertex weight vector. For the result we take the clique heaviest among those obtained this way due to the stationary points if it is better than the clique found at the preliminary stage. Otherwise, we preserve the old clique.

The greedy heuristic used in our method to process the stationary points is a generalization of the New-Best-In sequential degree heuristic. It runs in $O(n^2)$ time.

Algorithm 1 (New-Best-In Weighted)

Input: *a graph* $G(V, E)$, *a vector* $x \in \mathbb{R}^n$.

Output: *a maximal clique* Q.

1. *Construct vector* $y \in \mathbb{R}^n$ *such that* $y_i = x_i + \sum_{j\in N(i)} x_j$.

2. *Set* $V_1 := V; \quad k := 1; \quad Q := \oslash$.

3. *Choose a vertex* $v_k \in V_k$ *such that* y_{v_k} *is greatest.*

4. *Set* $Q := Q \cup \{v_k\}$.

5. *Set* $V_{k+1} := V_k \cap N(v_k)$.

6. *For each* $j \in V_{k+1}$, $y_j := y_j - \sum_{\ell \in (V_k \setminus V_{k+1}) \cap N(j)} x_\ell$.

7. *If* $V_{k+1} \neq \oslash$, *then* $k := k + 1$ *and go to 3.*

8. *STOP.*

The usual version of this algorithm is when the input vector x is the vertex weight vector w. Within our trust region technique we submit to this routine the obtained spheral stationary points.

Before anything else we apply a preprocessing able to reduce the input graph in some instances. It is clear that removing of too low connected vertices and preselection of too high connected vertices – when these operations do not result missing of the exact solution – are desirable as the Theorem 3 condition may be violated because of such vertices most. Thus, we iteratively remove vertices, whose weight together with the neighborhood weight is below the clique weight derived by Algorithm 1, and preselect any vertex disconnected only with a set weighting not more than the vertex itself.

Algorithm 2 (NBIW-based Graph Preprocess)

Input: *a graph* $G(V, E)$, *its vertex weight vector* w.

Output: *a reduced graph* $G(V, E)$, *a preselected vertex subset* Q_0, *a clique* Q.

1. *Set* $Q_0 := \oslash$, $B := 0$.

2. *Do:*

2.1. *assign* Q *the result of Algorithm 1 for* $G(V, E)$ *with its vertex weight vector* w;

2.2. *if* $W(Q) \leq B$, *go to 3;*

2.3. *set* $B := W(Q)$;

2.3. *set* $flag := false$;

2.4. *compose set* R *of vertices* $i \in V$ *such that* $w_i + \sum_{j \in N(i)} w_j < B$;

2.5. *if* $R \neq \oslash$, *then* $flag := true$;

2.6. *remove the vertex subset* R *from the graph* $G(V, E)$;

2.7. *compose a clique* P *of vertices* $i \in V$ *such that* $w_i \geq \sum_{j \in V \setminus N(i) \setminus \{i\}} w_j$;

2.8. $B := B - \sum_{j \in P} w_j$;

2.9. $Q_0 := Q_0 \cup P$;

2.10. *compose set* R *of vertices* $i \in V$ *such that* $P \setminus N(i) \neq \oslash$;

2.11. *if* $R \neq \oslash$, *then* $flag := true$ *and go to 2.6;*

While ($flag$ *AND* $V \neq \oslash$)

3. *STOP.*

It is easy to see that one 2.6–2.11 cycle takes not more than an $O(n^2)$ time and is repeated only if at least one vertex is removed from the graph. As well, there are not more than n calls of the Algorithm 1. Hence, the preprocessing complexity is in $O(n^3)$.

The preliminary greedy heuristic we use to derive a first approximation of the maximum weight clique calls Algorithm 1 n times starting from each of the vertices as chosen á priori.

Algorithm 3 (Meta-NBIW Algorithm)

Input: *a graph $G(V, E)$, its vertex weight vector w.*

Output: *a maximal clique $\hat{Q}$.*

1. *Set $\hat{Q} := \oslash$.*

2. *For each $i \in V$:*

2.1. *construct the subgraph $\mathcal{N}_G^i$ induced by $N(i)$;*

2.2. *assign Q the result of Algorithm 1 for $\mathcal{N}_G^i$ with its vertex weight subvector;*

2.3. *$Q := Q \cup \{i\}$;*

2.4. *if Q is better than $\hat{Q}$, then $\hat{Q} := Q$.*

3. *STOP.*

Obviously, the complexity of Algorithm 3 is $n \cdot O(n^2) = O(n^3)$. It does not exceed the trust region procedure complexity, so this process does not increase the total complexity of the method.

Thus, we propose the following method for the maximum weight clique problem.

Algorithm 4 (QUALEX-MS)

Input: *a graph $G(V, E)$, its vertex weight vector w.*

Output: *a maximal clique Q.*

1. *Execute Algorithm 2; store the preselected vertex set Q_0 and the clique Q.*

2. *If $V = \oslash$, then go to 12.*

3. *Execute Algorithm 3 and store the result $\hat{Q}$.*

4. *Compute z by (10), x^0 by (25), $\delta^{(w)}$ by (30), and D by (31).*

5. *Compute $\hat{A}$ by (28) and $\hat{b}$ by (29).*

6. *Perform the eigendecomposition $\hat{A} = R diag(\lambda_1, \ldots, \lambda_n) R^T$.*

7. *Compute the vector $c = R^T \hat{b}$.*

8. *Compute r^2 as $\hat{r}^2$ by (32) for $W(\hat{Q})$.*

9. *For each $\mu > 0$ minimizing left-hand side of (20) in a continuous interval or obeying (20):*

9.1. *compute y by (19);*

9.2. *compute $x = Ry + x^0$;*

9.3. *rescale $x_i := z_i x_i,\ i \in V$;*

9.4. *execute Algorithm 1 with the vector x and rewrite the result in $\hat{Q}$ if it is a better solution.*

End

10. *For each eigenvalue cluster $\lambda_i = \ldots = \lambda_{i+k-1} > 0$ satisfying (21):*

10.1. *compute all $y_j,\ j \in V \setminus \{i, \ldots, i+k-1\}$ by (19);*

10.2. *compute r_0^2 by (22);*

10.3. *if $r_0^2 \leq r^2$, then for each combination of $y_j,\ j \in \{i, \ldots, i+k-1\}$ defined by (23):*

10.3.1. *compute $x = Ry + x^0$;*

10.3.2. *rescale $x_i := z_i x_i,\ i \in V$;*

10.3.3. *execute Algorithm 1 with the vector x and rewrite the result in $\hat{Q}$ if it is a better solution;*

end

End

11. *If $\hat{Q}$ is a better solution than Q, then $Q := \hat{Q}$.*

12. $Q := Q \cup Q_0$.

13. *STOP.*

5 Computational Experiment Results

The goal of the first computational experiment was to find a smallest maximum clique instance, where QUALEX-MS cannot find an exact solution. We used the program `geng` available at [20] to generate all non-isomorphic to each other graphs up to 10 vertices inclusive. QUALEX-MS successfully found exact solutions to all those instances. Though it may not be excepted for sure that with another vertex numbering in one of them the exact solution would be lost, we consider this result to be a strong evidence that counterexamples to the algorithm do not exist up to 11-vertex graphs at least. Unfortunately, there are too many non-isomorphic 11-vertex graphs to continue the experiment the same way, so it has not been completed.

Next, we tested QUALEX-MS on all 80 DIMACS maximum clique instances[1]. At that we compared QUALEX-MS with the algorithm QSH presented in [4] and our previous maximum clique package QUALEX 2.0 (also available at [19].) These two algorithms previously showed considerable advances in handling DIMACS benchmark graphs comparing to the results of other published algorithms. All three programs were run on a Pentium IV 1.4GHz computer under OS Linux RedHat. However, the QUALEX-MS package makes use of a new eigendecomposition routine `DSYEVR` from LAPACK involving Relatively Robust Representations to compute eigenpairs after the matrix is reduced to a tridiagonal form [18]. This explains improvement of the average running time versus the two other programs. As a BLAS implementation, the platform-specific prebuilt of ATLAS library[2] was used.

Exact or best known solutions were found by QUALEX-MS in 57 instances. It is significantly better than 39 exact or best known solutions by QSH and an advance comparing to 51 exact or best known solutions by QUALEX 2.0. For the rest DIMACS graphs QUALEX-MS obtained good approximation solutions. The results are composed in Table 1.

Table 1: DIMACS maximum clique benchmark results

Instance	n	density	$\omega(G)$	QSH		QUALEX 2.0		QUALEX-MS	
				found	t (sec.)	found	t (sec.)	found	t (sec.)
`brock200_1`	200	0.745	21	21	1	21	1	21	1
`brock200_2`	200	0.496	12	12	< 1	12	< 1	12	< 1
`brock200_3`	200	0.605	15	15	< 1	15	1	15	1
`brock200_4`	200	0.658	17	17	1	17	< 1	17	< 1
`brock400_1`	400	0.748	27	27	4	27	4	27	2
`brock400_2`	400	0.749	29	29	4	29	4	29	3
`brock400_3`	400	0.748	31	31	4	31	5	31	2
`brock400_4`	400	0.749	33	33	4	33	4	33	2
`brock800_1`	800	0.649	23	17	37	23	36	23	18
`brock800_2`	800	0.651	24	24	38	24	35	24	18
`brock800_3`	800	0.649	25	25	38	25	37	25	18
`brock800_4`	800	0.650	26	26	37	26	35	26	18
`C125.9`	125	0.898	≥ 34	31	< 1	33	< 1	34	< 1
`C250.9`	250	0.899	≥ 44	42	1	43	1	44	1
`C500.9`	500	0.900	≥ 57	52	8	53	8	55	4
`C1000.9`	1000	0.901	≥ 68	62	103	63	71	64	27
`C2000.5`	2000	0.500	≥ 16	13	1593	16	1547	16	278
`C2000.9`	2000	0.900	≥ 77	67	1545	72	1519	72	215
`C4000.5`	4000	0.500	≥ 18	15	16198	17	15558	17	2345
`c-fat200-1`	200	0.077	12	12	< 1	12	< 1	12	< 1
`c-fat200-2`	200	0.163	24	24	< 1	24	< 1	24	< 1

[1] available at ftp://dimacs.rutgers.edu/pub/challenge/graph/

[2] available at http://www.netlib.org/atlas/archives/

Table 1: DIMACS maximum clique benchmark results

Instance	n	density	$\omega(G)$	QSH		QUALEX 2.0		QUALEX-MS	
				found	t (sec.)	found	t (sec.)	found	t (sec.)
c-fat200-5	200	0.426	58	58	< 1	58	< 1	58	< 1
c-fat500-1	500	0.036	14	14	5	14	4	14	1
c-fat500-2	500	0.073	26	26	5	26	3	26	2
c-fat500-5	500	0.186	64	64	2	64	2	64	2
c-fat500-10	500	0.374	126	126	3	126	3	126	2
DSJC500.5	500	0.500	≥ 13	11	9	13	8	13	5
DSJC1000.5	1000	0.500	≥ 15	13	85	14	74	14	36
gen200_p0.9_44	200	0.900	44	37	1	39	1	42	< 1
gen200_p0.9_55	200	0.900	55	55	< 1	55	< 1	55	1
gen400_p0.9_55	400	0.900	55	48	4	50	4	51	2
gen400_p0.9_65	400	0.900	65	63	4	65	4	65	2
gen400_p0.9_75	400	0.900	75	75	4	75	4	75	2
hamming6-2	64	0.905	32	32	< 1	32	< 1	32	< 1
hamming6-4	64	0.349	4	4	< 1	4	< 1	4	< 1
hamming8-2	256	0.969	128	128	1	128	1	128	< 1
hamming8-4	256	0.639	16	16	1	16	1	16	1
hamming10-2	1024	0.990	512	512	72	512	61	512	38
hamming10-4	1024	0.829	≥ 40	36	70	36	62	36	45
johnson8-2-4	28	0.556	4	4	< 1	4	< 1	4	< 1
johnson8-4-4	70	0.768	14	14	< 1	14	< 1	14	< 1
johnson16-2-4	120	0.765	8	8	< 1	8	< 1	8	< 1
johnson32-2-4	496	0.879	16	16	5	16	5	16	8
keller4	171	0.649	11	11	< 1	11	< 1	11	1
keller5	776	0.751	27	23	22	26	19	26	16
keller6	3361	0.818	≥ 59	48	6095	51	5721	53	1291
MANN_a9	45	0.927	16	16	< 1	16	< 1	16	< 1
MANN_a27	378	0.990	126	125	2	126	2	125	1
MANN_a45	1035	0.996	345	342	70	342	61	342	17
MANN_a81	3321	0.999	≥ 1100	1096	6671	1096	6057	1096	477
p_hat300-1	300	0.244	8	7	1	8	2	8	1
p_hat300-2	300	0.489	25	24	2	24	1	25	1
p_hat300-3	300	0.744	36	33	1	35	2	35	1
p_hat500-1	500	0.253	9	9	9	9	9	9	3
p_hat500-2	500	0.505	36	33	8	36	9	36	4
p_hat500-3	500	0.752	≥ 50	46	8	48	9	48	4
p_hat700-1	700	0.249	11	8	23	11	24	11	10
p_hat700-2	700	0.498	44	42	24	43	26	44	12
p_hat700-3	700	0.748	≥ 62	59	24	61	24	62	11
p_hat1000-1	1000	0.245	≥ 10	9	82	10	76	10	28
p_hat1000-2	1000	0.490	≥ 46	43	85	45	79	45	34
p_hat1000-3	1000	0.744	≥ 68	62	83	65	76	65	32
p_hat1500-1	1500	0.253	12	10	458	12	489	12	95
p_hat1500-2	1500	0.506	≥ 65	62	453	64	507	64	111
p_hat1500-3	1500	0.754	≥ 94	85	465	91	486	91	108
san200_0.7_1	200	0.700	30	30	< 1	30	< 1	30	1
san200_0.7_2	200	0.700	18	18	1	18	< 1	18	< 1
san200_0.9_1	200	0.900	70	70	< 1	70	1	70	< 1
san200_0.9_2	200	0.900	60	60	< 1	60	< 1	60	1
san200_0.9_3	200	0.900	44	35	1	40	< 1	40	< 1
san400_0.5_1	400	0.500	13	9	3	13	4	13	2
san400_0.7_1	400	0.700	40	40	4	40	4	40	3
san400_0.7_2	400	0.700	30	30	4	30	4	30	2
san400_0.7_3	400	0.700	22	16	4	17	4	18	2
san400_0.9_1	400	0.900	100	100	3	100	4	100	2
san1000	1000	0.502	15	10	76	15	69	15	25
sanr200_0.7	200	0.697	18	15	1	17	1	18	1
sanr200_0.9	200	0.898	42	37	< 1	41	< 1	41	< 1
sanr400_0.5	400	0.501	13	11	4	12	4	13	2
sanr400_0.7	400	0.700	≥ 21	18	4	20	5	20	2

The last computational experiment performed with QUALEX-MS was finding maximum weight cliques. Since there are no widely accepted maximum weight clique test suites, we followed the approach accepted in [9] and tested the algorithm against *normal* and *irregular* random graphs with various edge densities. To generate the irregular random graphs Algorithm

4.1 from [9] was used. Vertex weights were evenly distributed random integer numbers from 1 to 10. Due to significantly better speed of QUALEX-MS comparing to the heuristics considered in [9] and availability of a highly optimized exact maximum weight clique solver `cliquer` by P. Östergård and S. Niskanen[3], we were able to perform the tests not only on 100-vertex graphs but also on 200-vertex graphs up to the edge density 0.8. As well, we increased the number of tested graphs in each group from 20 to 50. The running time of QUALEX-MS on all those instances is in 1 second, so it may be considered negligible. However, similar testing on larger graphs is unfortunately difficult because of significant slowing down of the exact solver.

Table 2 presents the results of this computational experiment comparing with results of the algorithm *PBH* suggested in [9]. The measured value is percentage of the found clique weights to the heaviest clique weights averaged through all graphs of a group (*Avg.* R columns). Second result columns represent standard deviations of these values (*St. Dev.* columns.) The obtained figures show that our method strictly outperforms the algorithm PBH and the weight difference between heaviest cliques and those found by QUALEX-MS is rather negligible. This success consists with the fact reported in [12] regarding the ability of the QUALEX 2.0 routine to solve always exactly certain maximum weight clique instances arisen in practice in a short time.

Table 2: Performance of QUALEX-MS vs. PBH on random weighted graphs

n	density	QUALEX-MS				PBH			
		Normal		Irregular		Normal		Irregular	
		Avg. R	St. Dev.	Avg. R	St. Dev.	Avg. R	St. Dev.	Avg. R	St. Dev.
100	0.10	100.00%	±0.00	100.00%	±0.00	97.95%	±0.15	98.44%	±0.13
100	0.20	100.00%	±0.00	99.88%	±0.05	97.73%	±0.16	98.64%	±0.12
100	0.30	99.87%	±0.05	99.89%	±0.04	97.25%	±0.17	98.84%	±0.11
100	0.40	99.48%	±0.18	99.75%	±0.05	95.04%	±0.23	98.53%	±0.12
100	0.50	99.45%	±0.19	99.81%	±0.04	94.61%	±0.24	98.74%	±0.12
100	0.60	99.18%	±0.21	99.93%	±0.02	94.71%	±0.23	99.64%	±0.06
100	0.70	98.02%	±0.32	99.84%	±0.03	96.10%	±0.20	98.94%	±0.11
100	0.80	98.54%	±0.29	99.99%	±0.00	93.13%	±0.26	98.56%	±0.12
100	0.90	98.43%	±0.27	99.99%	±0.00	94.29%	±0.24	99.56%	±0.07
100	0.95	98.72%	±0.20	100.00%	±0.00	96.49%	±0.19	99.75%	±0.05
200	0.10	100.00%	±0.00	99.97%	±0.04				
200	0.20	99.55%	±0.19	99.86%	±0.04				
200	0.30	99.33%	±0.29	99.45%	±0.16				
200	0.40	99.08%	±0.45	99.36%	±0.35				
200	0.50	98.34%	±0.46	99.32%	±0.14				
200	0.60	98.00%	±0.35	99.61%	±0.10				
200	0.70	96.99%	±0.64	99.54%	±0.12				
200	0.80	96.21%	±0.55	99.71%	±0.10				

6 Remarks and Conclusions

We have presented a new fast method for the maximum weight clique approximation. Both theoretically and experimentally we have shown there is a considerable range of instances where the method is exact. Without any overestimation it may be said currently there is no other heuristic approach to the clique problem producing results competitive with the presented ones. Among instance classes solved by the algorithm exactly there are those definitely hard for existing combinatorial algorithms and possibly even intractable for any combinatorial approach. Such an extreme case is Brockington-Culberson graphs from the DIMACS test suite [13] (`brock*`). Apart from the fact that no combinatorial algorithm succeeded on them so far, they were claimed hard for algorithms based on the Motzkin–Straus continuous formulation in [9]. However, the QUALEX-MS algorithm handles these instances with the greatest success.

[3]available at http://www.hut.fi/~ pat/cliquer.html

As the next step of QUALEX-MS development it should be investigated is there a possibility to express Motzkin–Straus optima as a function of a particular subset of the spheral stationary points. It may lead to a generalization of the Theorem 3 expanding the class of maximum weight clique instances where the optimum is directly computable by the presented trust region procedure.

A case theoretically seeming to be the worst for the described technique is when there are multiple eigenvalues causing the trust region problem degeneration. This occurs when the graph has a certain symmetry. It may be supposed that a special submethod dealing with such instances should be developed.

References

[1] R.P. Brent, *Algorithms for Minimization without Derivatives* (Englewood Cliffs, NJ: Prentice-Hall, 1973).

[2] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman & Co., 1979).

[3] J. Abello, S. Butenko, P.M. Pardalos, and M.G.C. Resende, Finding independent sets in a graph using continuous multivariable polynomial formulations, *Journal of Global Optimization* **21**:4 (2001) 111–137.

[4] S. Busygin, S. Butenko, and P.M. Pardalos, A heuristic for the maximum independent set problem based on optimization of a quadratic over a sphere, *Journal of Combinatorial Optimization* **6**:3 (2002) 287-297.

[5] G.E. Forsythe and G.H. Golub, On the stationary values of a second degree polynomial on the unit sphere, *SIAM J. Appl. Math.* **13** (1965) 1050–1068.

[6] L.E. Gibbons, D.W. Hearn, P.M. Pardalos, and M.V. Ramana, Continuous characterizations of the maximum clique problem, *Math. Oper. Res.* **22** (1997) 754–768.

[7] W. Hager, Minimizing a quadratic over a sphere, *SIAM J. Optim.* **12** (2001) 188–208.

[8] S. Lyle and M. Szularz, Local minima of the trust region problem, *Journal of Optimization Theory and Applications* **80** (1994) 117–134.

[9] A. Massaro, M. Pelillo, and I.M. Bomze, A complementary pivoting approach to the maximum weight clique problem, *SIAM J. Optim.* **12**:4 (2001) 928–948.

[10] J.J. Moré and D.S. Sorensen, Computing a trust region step, *SIAM J. Sci. Statist. Comput.* **4** (1983) 553–572.

[11] T.S. Motzkin and E.G. Straus, Maxima for graphs and a new proof of a theorem of Turan, *Canadian Journal of Mathematics* **17**:4 (1965) 533–540.

[12] S. Babu, M. Garofalakis, and R. Rastogi, SPARTAN: A model-based semantic compression system for massive data tables, *ACM SIGMOD* (Santa Barbara, CA, May 2001) 283–295.

[13] M. Brockington and J.C. Culberson, Camouflaging independent sets in quasi-random graphs, in: D. Johnson and M.A. Trick, eds., *Cliques, Coloring and Satisfiability, DIMACS Ser. Discrete Math. Theoret. Comput. Sci.* **26** (AMS, Providence, RI, 1996) 75–88.

[14] J. Håstad, Clique is hard to approximate within $n^{1-\epsilon}$, in: *Proc. 37th Annual IEEE Symposium on the Foundations of Computer Science (FOCS)* (1996) 627–636.

[15] S. Khot, Improved inapproximability results for maxclique, chromatic number and approximate graph coloring, in: *Proc. 42nd Annual IEEE Symposium on the Foundations of Computer Science (FOCS)* (2001) 600–609.

[16] V.Y. Pan and Z.Q. Chen, The complexity of the matrix eigenproblem, in: *The 21st Annual ACM Symposium on Theory of Computing* (Atlanta, Georgia, ACM Press, 1999) 507–516.

[17] Y. Ye, A new complexity result on minimization of a quadratic function with a sphere constraint, in: *Recent Advances in Global Optimization* (Princeton University Press, 1992), 19–31.

[18] I.S. Dhillon, A new $O(n^2)$ algorithm for the symmetric tridiagonal eigenvalue/eigenvector problem, *Computer Science Division Technical Report No. UCB//CSD-97-971* (UC Berkeley, 1997).

[19] S. Busygin, Stas Busygin's NP-Completeness Page,
`http://www.busygin.dp.ua/npc.html`

[20] B.D. McKay, The nauty page,
`http://cs.anu.edu.au/~bdm/nauty/`