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 $O(n^2)$, 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

*Copyright © Stanislav Busygin, 2001, 2002. All rights reserved.
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 \textit{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].

\textbf{Theorem 1 (Motzkin–Straus)} The global optimum value of the quadratic program

\[ \max f(x) = \frac{1}{2} x^T A_G x \]  
subject to

\[ \sum_{i \in V} x_i = 1, \ x \geq 0 \]  
is

\[ \frac{1}{2} \left( 1 - \frac{1}{\omega(G)} \right). \]  

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

\[ d_i = 1 - \frac{w_{\text{min}}}{w_i}. \]

\textbf{Theorem 2} The global optimum value of the quadratic program

\[ \max f(x) = x^T (A_G + \text{diag}(d_1, \ldots, d_n)) x \]  
subject to

\[ \sum_{i \in V} x_i = 1, \ x \geq 0 \]  
is

\[ 1 - \frac{w_{\text{min}}}{\omega(w, G)}. \]
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

$$x_i = \frac{\omega_i}{\omega(u, G)}.$$  \hfill(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:

\[
\begin{align*}
  f_i(x) &= d_i x_i^2 + 2x_i \sum_{k \in N(i)} x_k, \\
  f_j(x) &= d_j x_j^2 + 2x_j \sum_{k \in N(j)} x_k, \\
  f_{ij}(x) &= f(x) - f_i(x) - f_j(x).
\end{align*}
\]

Consider the partial derivatives of the objective on \(x_i\) and \(x_j\):

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

Without loss of generality, let \(\frac{\partial f_i}{\partial x_i} \geq \frac{\partial f_j}{\partial x_j}\). 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). \(f_{ij}(x^*) = f_{ij}(x)\) 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.

\[
\begin{align*}
  f_i(x^*) &= d_i(x_i + x_j)^2 + 2(x_i + x_j) \sum_{k \in N(i)} x_k = \\
  &= f_i(x) + 2d_i x_i x_j + d_i x_j^2 + 2x_j \sum_{k \in N(i)} x_k = \\
  &= f_i(x) + x_j \left( 2d_i x_i + 2 \sum_{k \in N(i)} x_k \right) + d_i x_j^2 = \\
  &\geq f_i(x) + x_j \frac{\partial f_i}{\partial x_i} + d_i x_j^2 \geq f_i(x) + x_j \frac{\partial f_j}{\partial x_j} + d_i x_j^2 \geq \\
  &\geq f_i(x) + x_j \frac{\partial f_j}{\partial x_j} = f_i(x) + 2d_j x_j^2 + 2x_j \sum_{k \in N(j)} x_k \geq \\
  &\geq f_i(x) + d_j x_j^2 + 2x_j \sum_{k \in N(j)} x_k = f_i(x) + f_j(x).
\end{align*}
\]

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:

\[
\max f(x) = \sum_{i \in Q} d_i x_i^2 + \sum_{i \in Q} \sum_{j \not\in Q} x_i x_j
\]

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:

\[
\sum_{i \in Q} \frac{x_i^2}{w_i} \rightarrow \min
\]

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

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

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.

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

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 \textit{weighted adjacency matrix} \( A^{(w)}_G = (a^{(w)}_{ij})_{n \times n} \) such that

\[
a^{(w)}_{ij} = \begin{cases} 
  w_i - w_{\min}, & \text{if } i = j \\
  \sqrt{w_i w_j}, & \text{if } (i, j) \in E \\
  0, & \text{if } i \neq j \text{ and } (i, j) \not\in E.
\end{cases}
\]
Obviously, it is the ordinary adjacency matrix when all vertex weights are ones. Next, we introduce the vector of vertex weight square roots

\[ z \in \mathbb{R}^n : z_i = \sqrt{w_i}. \] (10)

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

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

\[ \max f(x) = x^T A_G^{(w)} x \] (11)

subject to

\[ z^T x = 1, \ x \geq 0 \] (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

\[ x_i = \frac{z_i}{\omega(w, G)}. \] (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

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

**Fact 1** All cliques of the same weight \( \sigma \) 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 spherical 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

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

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

where \( A \) is a given real symmetric \( n \times n \) matrix, \( b \in \mathbb{R}^n \) is a given vector, and \( x \in \mathbb{R}^n \) 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 non-decreasing 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

\[x = Q y, \quad y = Q^T x, \quad b = Q c, \quad c = Q^T b. \tag{17}\]

The Lagrangian of (15, 16) is

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

\( \mu \) is the lagrangian multiplier of the spherical constraint here. We take it with negative sign for the sake of convenience. The stationary conditions are

\[\frac{\partial L}{\partial y_i} = 0, \quad \frac{\partial L}{\partial \mu} = 0. \]
So,
\[ \frac{\partial L}{\partial y_i} = 2(\lambda_i - \mu)y_i + 2\mu = 0, \]
and assuming \( \mu \neq \lambda_i \),
\[ y_i = \frac{c_i}{\mu - \lambda_i}. \tag{19} \]
Substituting (19) into the spherical 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 \( \mu \)-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 \( \mu \) values.) That depends on the chosen radius \( r \): the greater the radius, the more cases of two spherical 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 degenerate case when \( \mu = \lambda_i \) for some \( i \) is possible if \( c_i = 0 \). At that, if \( \lambda_i \) is a multiple eigenvalue of \( A \), all \( c_j \) corresponding to \( \lambda_i = \lambda_i \) 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 degenerate 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 2\( k \) cases:
\[
\begin{align*}
y_i &= \pm \sqrt{r^2 - r_0^2}, & y_{i+1} &= 0, \ldots, y_{i+k-1} &= 0, \\
y_i &= 0, & y_{i+1} &= \pm \sqrt{r^2 - r_0^2}, \ldots, y_{i+k-1} &= 0, \\
& \quad \ldots \\
y_i &= 0, y_{i+1} = 0, \ldots, y_{i+k-1} &= \pm \sqrt{r^2 - r_0^2},
\end{align*} \tag{23}
\]
so an eigenvalue of multiplicity \( k \) gives 2\( k \) 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 \( \mu \) 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 \( O(n^3) \) complexity each afterwards.
4 The QUALEX-MS Algorithm

Thus, we will work with the program

$$\max f(x) = x^T A_G^{(w)} x$$  \hspace{1cm} (24)

$$\text{s.t. } z^T x = 1, \quad x^T x \leq r^2,$$

where $r$ is a parameter not fixed à 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

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

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

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

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

$$\frac{\partial L}{\partial x^Q_i} = 2 \sum_{i \in V} a_{ij}^{(w)} x^Q_j + z_i \mu_1 + 2 x^Q_i \mu_2 =$$

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

Let $i \in Q$. Then it gives

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

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

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

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

We may see that in both cases the final expressions are independent of a particular $i$ value. So, the stationary point criterion system $\partial L/\partial x^Q_i = 0$ is reduced to two equations over two variables $\mu_1$ and $\mu_2$. The second equation directly gives

$$\mu_1 = -\frac{2C}{W(Q)}.$$
Substituting this into the first equation, we obtain

\[ \mu_2 = C + w_{\text{min}} - W(Q). \]

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 \( \mu_2 \) 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 \( Q \)

\[ x^Q_i = \begin{cases} 
 1/|Q|, & \text{if } i \in Q \\
 0, & \text{if } i \in V \setminus Q 
\end{cases} \]

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 occurred 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

\[ x^0 = z/W(V). \]  \hspace{1cm} (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)

\[ \max g(\hat{x}) = \hat{x}^T A_G^{(w)} \hat{x} + 2(x^0)^T A_G^{(w)} \hat{x} \]

\[ \text{s.t. } z^T \hat{x} = 0, \hat{x}^T \hat{x} \leq \hat{r}^2, \]  \hspace{1cm} (26)

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

\[ p_{ij} = \begin{cases} 
 1 - w_i/W(V), & \text{if } i = j \\
 -w_{ij}W_{ij}/W(V), & \text{if } i \neq j
\end{cases} \]

Thus, the program (26) may be reformulated as

\[ \max g(\hat{x}) = \hat{x}^T \hat{A} \hat{x} + 2\hat{b}^T \hat{x} \]  \hspace{1cm} (27)
s.t. \( \tilde{x}^T \tilde{x} \leq \tilde{r}^2, \)

where \( \tilde{A} = PA_G^{(w)}P \) and \( \tilde{b}^T = (x_0)^T A_G^{(w)}P. \)

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

\[
\tilde{a}_{ij} = a_{ij}^{(w)} - x_j^0 \delta_i^{(w)} - x_i^0 \delta_j^{(w)} + x_i^0 x_j^0 D
\]

and

\[
\tilde{b}_i = \delta_i^{(w)} - x_i^0 D, \tag{29}
\]

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

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

or less. In our method we consider the stationary points having this \( \tilde{r}^2 \) value, plus those corresponding to \( \mu \) 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 \( \mu \) 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 \( V_1 := V; \quad k := 1; \quad Q := \emptyset. \)
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 \( j \in V_{k+1}, \ y_j := \sum_{\ell \in (V_1 \setminus V_{k+1}) \cap N_j} x_\ell. \)
7. If \( V_{k+1} \neq \emptyset, \) 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 spherical 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 \( Q_0 := \emptyset, \) \( B := 0. \)
2. Do:
   1. assign \( Q \) the result of Algorithm 1 for \( G(V,E) \) with its vertex weight vector \( w; \)
   2. if \( W(Q) \leq B, \) go to 3;
   3. set \( B := W(Q); \)
   4. set \( \text{flag} := \text{false}; \)
   5. compose set \( R \) of vertices \( i \in V \) such that \( w_i + \sum_{j \in N(i)} w_j < B; \)
   6. if \( R \neq \emptyset, \) then \( \text{flag} := \text{true}; \)
   7. remove the vertex subset \( R \) from the graph \( G(V,E); \)
   8. compose a clique \( P \) of vertices \( i \in V \) such that \( w_i \geq \sum_{j \in V \setminus N(i) \setminus i} w_j; \)
   9. \( B := B - \sum_{j \in P} w_j; \)
   10. \( Q_0 := Q_0 \cup P; \)
   11. if \( R \neq \emptyset, \) then \( \text{flag} := \text{true} \) and go to 2.6;
   12. While (\( \text{flag AND V} \neq \emptyset) \)

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 a 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 $\hat{Q} := \emptyset$.

2. For each $i \in V$:
   1. construct the subgraph $N_G^i$ induced by $N(i)$;
   2. assign $Q$ the result of Algorithm 1 for $N_G^i$ with its vertex weight subvector;
   3. $Q := Q \cup \{i\}$;
   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 $V = \emptyset$, 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 $\hat{A} = R \text{diag}(\lambda_1, \ldots, \lambda_n) R^T$.
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):
   1. compute $y$ by (19);
   2. compute $x = Ry + x^0$;
   3. rescale $x_i := z_i x_i$, $i \in V$;
   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):
   1. compute all $y_j$, $j \in V \setminus \{i, \ldots, i + k - 1\}$ by (19);
   2. compute $r^2$ by (22);
   3. if $r^2 \leq r^2$, then for each combination of $y_j$, $j \in \{i, \ldots, i + k - 1\}$ defined by (23):
     1. compute $x = Ry + x^0$;
     2. rescale $x_i := z_i x_i$, $i \in V$;
     3. execute Algorithm 1 with the vector $x$ and rewrite the result in $\hat{Q}$ if it is a better solution;

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 geamp 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>
<thead>
<tr>
<th>Instance</th>
<th>n</th>
<th>density</th>
<th>$\omega(G)$</th>
<th>QSH</th>
<th>QUALEX 2.0</th>
<th>QUALEX-MS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>found</td>
<td>found</td>
<td>found</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$t$ (sec.)</td>
<td>$t$ (sec.)</td>
<td>$t$ (sec.)</td>
</tr>
<tr>
<td>breck200_1</td>
<td>200</td>
<td>0.745</td>
<td>21</td>
<td>21</td>
<td>1</td>
<td>21</td>
</tr>
<tr>
<td>breck200_2</td>
<td>200</td>
<td>0.496</td>
<td>12</td>
<td>12</td>
<td>&lt;1</td>
<td>12</td>
</tr>
<tr>
<td>breck200_3</td>
<td>200</td>
<td>0.605</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>breck200_4</td>
<td>200</td>
<td>0.588</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>breck200_1</td>
<td>400</td>
<td>0.578</td>
<td>27</td>
<td>27</td>
<td>7</td>
<td>27</td>
</tr>
<tr>
<td>breck200_2</td>
<td>400</td>
<td>0.749</td>
<td>30</td>
<td>30</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>breck200_3</td>
<td>400</td>
<td>0.748</td>
<td>31</td>
<td>31</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>breck200_4</td>
<td>400</td>
<td>0.749</td>
<td>33</td>
<td>33</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>breck200_1</td>
<td>800</td>
<td>0.649</td>
<td>23</td>
<td>23</td>
<td>33</td>
<td>33</td>
</tr>
<tr>
<td>breck200_2</td>
<td>800</td>
<td>0.651</td>
<td>24</td>
<td>24</td>
<td>35</td>
<td>35</td>
</tr>
<tr>
<td>breck200_3</td>
<td>800</td>
<td>0.619</td>
<td>25</td>
<td>25</td>
<td>37</td>
<td>37</td>
</tr>
<tr>
<td>breck200_4</td>
<td>800</td>
<td>0.650</td>
<td>26</td>
<td>26</td>
<td>35</td>
<td>35</td>
</tr>
<tr>
<td>C125.9</td>
<td>125</td>
<td>0.898</td>
<td>$\geq 34$</td>
<td>31</td>
<td>$&lt; 1$</td>
<td>34</td>
</tr>
<tr>
<td>C250.9</td>
<td>250</td>
<td>0.899</td>
<td>$\geq 44$</td>
<td>42</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>C500.9</td>
<td>500</td>
<td>0.900</td>
<td>$\geq 57$</td>
<td>52</td>
<td>8</td>
<td>55</td>
</tr>
<tr>
<td>C1000.9</td>
<td>1000</td>
<td>0.901</td>
<td>$\geq 68$</td>
<td>62</td>
<td>103</td>
<td>63</td>
</tr>
<tr>
<td>C2000.5</td>
<td>2000</td>
<td>0.500</td>
<td>$\geq 16$</td>
<td>13</td>
<td>1593</td>
<td>16</td>
</tr>
<tr>
<td>C5000.9</td>
<td>5000</td>
<td>0.900</td>
<td>$\geq 77$</td>
<td>77</td>
<td>1541</td>
<td>72</td>
</tr>
<tr>
<td>C10000.5</td>
<td>10000</td>
<td>0.500</td>
<td>$\geq 18$</td>
<td>15</td>
<td>16198</td>
<td>17</td>
</tr>
<tr>
<td>r-fat200_1</td>
<td>200</td>
<td>0.077</td>
<td>12</td>
<td>12</td>
<td>$&lt; 1$</td>
<td>12</td>
</tr>
<tr>
<td>r-fat200_2</td>
<td>200</td>
<td>0.163</td>
<td>24</td>
<td>24</td>
<td>$&lt; 1$</td>
<td>24</td>
</tr>
</tbody>
</table>

\(^1\)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

<table>
<thead>
<tr>
<th>Instance</th>
<th>n</th>
<th>density</th>
<th>(\omega(n))</th>
<th>QSH</th>
<th>QUALEX 2.0</th>
<th>QUALEX-MS</th>
</tr>
</thead>
<tbody>
<tr>
<td>c-fat500-5</td>
<td>500</td>
<td>0.426</td>
<td>58</td>
<td>58</td>
<td>58</td>
<td>58</td>
</tr>
<tr>
<td>c-fat500-1</td>
<td>500</td>
<td>0.036</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>c-fat500-2</td>
<td>500</td>
<td>0.073</td>
<td>26</td>
<td>26</td>
<td>26</td>
<td>26</td>
</tr>
<tr>
<td>c-fat500-5</td>
<td>500</td>
<td>0.186</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>c-fat500-10</td>
<td>500</td>
<td>0.374</td>
<td>126</td>
<td>126</td>
<td>126</td>
<td>126</td>
</tr>
<tr>
<td>gsc500.5</td>
<td>500</td>
<td>0.500</td>
<td>\geq 13</td>
<td>11</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>gsc1000.5</td>
<td>1000</td>
<td>0.500</td>
<td>\geq 15</td>
<td>13</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>gen200.p0.9.44</td>
<td>200</td>
<td>0.000</td>
<td>44</td>
<td>37</td>
<td>39</td>
<td>42</td>
</tr>
<tr>
<td>gen200.p0.9.55</td>
<td>200</td>
<td>0.000</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>gen400.p0.9.55</td>
<td>400</td>
<td>0.000</td>
<td>55</td>
<td>48</td>
<td>50</td>
<td>51</td>
</tr>
<tr>
<td>gen400.p0.9.6S</td>
<td>400</td>
<td>0.000</td>
<td>65</td>
<td>63</td>
<td>65</td>
<td>65</td>
</tr>
<tr>
<td>gen400.p0.9.7S</td>
<td>400</td>
<td>0.000</td>
<td>75</td>
<td>75</td>
<td>75</td>
<td>75</td>
</tr>
<tr>
<td>hamming6-2</td>
<td>64</td>
<td>0.005</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>hamming6-4</td>
<td>64</td>
<td>0.349</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>hamming8-2</td>
<td>256</td>
<td>0.069</td>
<td>128</td>
<td>128</td>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td>hamming8-4</td>
<td>256</td>
<td>0.639</td>
<td>16</td>
<td>1</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>hamming10-2</td>
<td>1024</td>
<td>0.090</td>
<td>512</td>
<td>512</td>
<td>512</td>
<td>512</td>
</tr>
<tr>
<td>hamming10-4</td>
<td>1024</td>
<td>0.829</td>
<td>\geq 40</td>
<td>36</td>
<td>36</td>
<td>36</td>
</tr>
<tr>
<td>johnson8-2-4</td>
<td>28</td>
<td>0.556</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>johnson8-4-4</td>
<td>70</td>
<td>0.768</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>johnson16-2-4</td>
<td>120</td>
<td>0.765</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>johnson32-2-4</td>
<td>256</td>
<td>0.579</td>
<td>16</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>keller4</td>
<td>411</td>
<td>0.649</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>keller6</td>
<td>776</td>
<td>0.751</td>
<td>27</td>
<td>23</td>
<td>26</td>
<td>26</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>keller6</td>
<td>3361</td>
<td>0.818</td>
<td>\geq 59</td>
<td>48</td>
<td>51</td>
<td>51</td>
</tr>
</tbody>
</table>

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\(^a\), 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

<table>
<thead>
<tr>
<th>n</th>
<th>density</th>
<th>QUALEX-MS</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Normal</td>
<td></td>
<td>Irregular</td>
<td></td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>0.10</td>
<td>100.00% ±0.00</td>
<td>100.00% ±0.00</td>
<td>97.95% ±0.15</td>
<td>98.44% ±0.13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>0.20</td>
<td>99.98% ±0.05</td>
<td>99.88% ±0.05</td>
<td>97.73% ±0.16</td>
<td>98.64% ±0.12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>0.30</td>
<td>99.49% ±0.19</td>
<td>99.75% ±0.05</td>
<td>97.23% ±0.17</td>
<td>98.81% ±0.11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>0.40</td>
<td>99.85% ±0.18</td>
<td>99.81% ±0.04</td>
<td>95.64% ±0.23</td>
<td>98.53% ±0.12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>0.50</td>
<td>99.18% ±0.21</td>
<td>99.93% ±0.02</td>
<td>94.61% ±0.24</td>
<td>98.74% ±0.12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>0.60</td>
<td>98.02% ±0.32</td>
<td>99.84% ±0.03</td>
<td>94.17% ±0.23</td>
<td>99.64% ±0.06</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>0.70</td>
<td>98.54% ±0.29</td>
<td>99.99% ±0.00</td>
<td>96.10% ±0.20</td>
<td>98.94% ±0.11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>0.80</td>
<td>98.43% ±0.27</td>
<td>99.90% ±0.00</td>
<td>93.13% ±0.26</td>
<td>98.96% ±0.12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>0.95</td>
<td>98.72% ±0.20</td>
<td>100.00% ±0.00</td>
<td>94.29% ±0.24</td>
<td>99.56% ±0.07</td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>0.10</td>
<td>100.00% ±0.00</td>
<td>99.97% ±0.04</td>
<td>96.49% ±0.19</td>
<td>99.75% ±0.05</td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>0.20</td>
<td>99.55% ±0.19</td>
<td>99.86% ±0.04</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>0.30</td>
<td>99.33% ±0.29</td>
<td>99.45% ±0.16</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>0.40</td>
<td>99.08% ±0.45</td>
<td>99.30% ±0.35</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>0.50</td>
<td>98.34% ±0.86</td>
<td>99.32% ±0.14</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>0.60</td>
<td>98.00% ±0.35</td>
<td>99.61% ±0.10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>0.70</td>
<td>96.99% ±0.64</td>
<td>99.54% ±0.12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>0.80</td>
<td>96.21% ±0.55</td>
<td>99.71% ±0.10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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.

\(^a\)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 spherical 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


