Available online at www.sciencedirect.com

SCIENCE DIRECT®

Operations Research Letters 32 (2004) 523–529

Operations Research Letters

www.elsevier.com/locate/dsw

The combinatorics of pivoting for the maximum weight clique

Marco Locatelli[a,*,1], Immanuel M. Bomze[b], Marcello Pelillo[c]

[a]*Dipartimento di Informatica, Università di Torino, Corso Svizzera 185, 10149 Torino, Italy*
[b]*Inst. für Statistik und Decision Support Systems, Universität Wien, Austria*
[c]*Dipartimento di Informatica, Università Ca' Foscari di Venezia, Italy*

Received 20 March 2003; accepted 9 February 2004

Abstract

In this paper we prove the equivalence between a pivoting-based heuristic (PBH) for the maximum weight clique problem and a combinatorial greedy heuristic. It is also proved that PBH always returns a local solution although this is not always guaranteed for Lemke's method, on which PBH is based.
© 2004 Elsevier B.V. All rights reserved.

Keywords: Maximum clique; Linear complementarity; Pivoting methods; Greedy heuristics; Combinatorial optimization

1. Introduction

Given an undirected graph, the maximum clique problem (MCP) consists of finding a subset of pairwise adjacent vertices (i.e., a *clique*) having largest cardinality. The problem is known to be NP-hard for arbitrary graphs and so is the problem of approximating it within a constant factor. A large set of benchmark problems, the DIMACS ones, are available for MCP (see [5]). An important generalization of the MCP arises when positive weights are associated to the vertices of the graph. In this case the problem is known as the maximum weight clique problem (MWCP) and consists of finding a clique in the graph which has largest total weight. More formally, let $G=(V,E,w)$ be an arbitrary undirected and weighted graph, where $V=\{1,\dots,n\}$ is the *vertex set*, E is the *edge set* and $w\in\mathbb{R}^n$ is the *weight* vector, the ith component of which corresponds to the weight assigned to vertex i. It is assumed that $w_i>0$ for all $i\in V$. Given a subset of vertices S, the weight assigned to S will be denoted by

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

As usual, the sum over the empty index set is defined to be zero. Then, problem MWCP is the following

$$\max\{W(S): S \text{ is a clique}\}.$$

The MWCP has important applications in such fields as computer vision, pattern recognition and robotics, where weighted graphs are employed as a convenient way of representing high-level pictorial information. We refer to [1] for a recent review concerning algorithms, applications and complexity issues of this important problem.

There are many reformulations of MCP and MWCP as continuous global optimization problems (see again

* Corresponding author.
E-mail address: locatelli@di.unito.it (M. Locatelli).
[1] Acknowledges support from Progetto FIRB "Ottimizzazione Non Lineare su Larga Scala".

[1] and references therein). In particular, a generalization of an earlier remarkable result by Motzkin and Strauss shows that MWCP can be reformulated as the problem of minimizing a quadratic function over the unit simplex. Based on this reformulation, a new pivoting-based heuristic (PBH) for the MWCP, based on the corresponding linear complementarity problem (LCP), has been proposed in the recent paper [7]. The algorithm is essentially a variant of Lemke's classical algorithm that incorporates an effective look-ahead pivot rule, and it proved to be among the most powerful MWCP heuristics available in the literature (in Section 2.1 both the quadratic programming reformulation and the PBH will be briefly recalled).

In this paper we provide some theoretical results which prove a few conjectures that have been put forward in [7] based on empirical observations. These results allow for an interesting combinatorial interpretation of the algorithm: PBH is essentially equivalent to a greedy combinatorial heuristic.

2. The PBH and its combinatorial interpretation

2.1. The PBH heuristic

Given a weighted graph $G=(V,E,w)$, consider the following standard quadratic program (StQP):

$$\begin{aligned} &\text{minimize} \quad x^{\mathrm{T}} Q_G x \\ &\text{subject to} \quad x \in \Delta, \end{aligned} \tag{1}$$

where the matrix $Q_G = (q_{ij})_{(i,j)\in V\times V}$ is defined as

$$q_{ij} = \begin{cases} \dfrac{1}{2w_i} & \text{if } i=j, \\ 0 & \text{if } \{i,j\}\in E, \\ \dfrac{1}{2w_i}+\dfrac{1}{2w_j} & \text{otherwise} \end{cases} \tag{2}$$

and Δ denotes the standard simplex in the n-dimensional Euclidean space $\mathbb{R}^n$:

$$\Delta = \left\{ x\in\mathbb{R}^n : \sum_{i\in V} x_i = 1 \text{ and } x_i \geqslant 0 \text{ for all } i\in V \right\}.$$

Given a subset of vertices $S\subseteq V$, the *weighted characteristic vector* of S, denoted by $x^{S,w}$, is the vector in Δ whose coordinates are given by

$$x_i^{S,w} = \begin{cases} \dfrac{w_i}{W(S)} & \text{if } i\in S, \\ 0 & \text{otherwise.} \end{cases}$$

The PBH heuristic is based upon the following result (see [2]).

Theorem 1. *Let $G=(V,E,w)$ be an arbitrary graph with positive weight vector $w\in\mathbb{R}^n$. Then the following assertions hold.*

- *A vector $x\in\Delta$ is a local solution to (1), i.e., yields a local minimum, if and only if $x=x^{S,w}$, where S is a maximal clique of G.*
- *A vector $x\in\Delta$ is a global solution to (1), i.e., yields a global minimum, if and only if $x=x^{S,w}$, where S is a maximum weight clique of G.*

Moreover, all local solutions to (1) are strict, and are characteristic vectors of maximal cliques of G.

It is well known that stationary points of quadratic optimization problems with linear constraints, i.e. the points which satisfy the first order necessary conditions for local optimality, can be characterized as the solutions of a LCP, a class of inequality systems for which a rich theory and a large number of algorithms have been developed (see [4]). Hence, once the MWCP is formulated in terms of an StQP, the use of LCP algorithms naturally suggests itself, and this is precisely the main idea proposed in [7].

Specifically, computing the stationary points of (1) can be done by solving the LCP (q_G, M_G), which is the problem of finding a vector x satisfying the system

$$\begin{aligned} &y = q_G + M_G\bar{x} \geqslant 0, \\ &\bar{x} = [x, x_{n+1}, x_{n+2}] \geqslant 0, \quad \bar{x}^{\mathrm{T}} y = 0, \end{aligned} \tag{3}$$

where

$$q_G = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ -1 \\ 1 \end{bmatrix} \quad \text{and} \quad M_G = \begin{bmatrix} Q_G & -e & e \\ e^{\mathrm{T}} & 0 & 0 \\ -e^{\mathrm{T}} & 0 & 0 \end{bmatrix},$$

with Q_G as in (2) and e is the vector with all coordinates equal to 1. With the above definitions, it is well known that if $\bar{x} \in \mathbb{R}^{n+2}$ solves (3)—in which case we say that $z = [x^T, y^T]^T$ is a complementary solution of LCP (q_G, M_G)—then x is a stationary point of (1). Note that Q_G is strictly $\mathbb{R}^n_+$-copositive, i.e.

$$xQ_Gx > 0 \quad \text{for all } x \geqslant 0 \quad \text{with } x \neq 0.$$

Hence so is M_G, and this is sufficient to ensure that LCP (q_G, M_G), or (3), always has a solution (see [4]).

Among the many LCP methods presented in the literature, pivoting procedures are widely used and within this class Lemke's method is certainly the best known, largely for its ability to provide a solution for several matrix classes. Given the generic LCP (q, M), this method considers the augmented problem LCPA (q, d, M) defined by

$$y = q + [M, d] \begin{bmatrix} \bar{x} \\ \theta \end{bmatrix} \geqslant 0, \quad \theta \geqslant 0,$$

$$\bar{x} \geqslant 0, \quad \bar{x}^T y = 0.$$

The vector d is called the *covering vector*, and must satisfy $d_i > 0$ whenever $q_i < 0$. Note that, if $q \geqslant 0$, then ($\theta = 0$ and) $\bar{x} = 0$, $y = q$ promptly yields a solution for LCP (q, M). Else, Lemke's method starts with the solution for LCPA (q, d, M) given by $\theta = \max\{-q_i/d_i : q_i < 0\} \geqslant 0$, $x = 0$ and $y = q + \theta d$. Then it generates new solutions for LCPA (q, d, M) where the value of θ is progressively decreased (or, at least, not increased) until a solution $\theta^*, \bar{x}^*, y^*$ with $\theta^* = 0$ is reached. Then $\bar{x}^*, y^*$ is a solution for LCP (q, M). We refer to [4] for a detailed description of Lemke's algorithm. In [7], $d = e$ was chosen, and our problem does not expose peculiarities that would justify a deviation from this common practice.

Unfortunately, like other pivoting schemes, the convergence of Lemke's algorithm is guaranteed only for non-degenerate problems, and ours is indeed degenerate (here degeneracy is intended in the usual LP sense). In [7], standard degeneracy resolution strategies were tested over a number of benchmark graphs, but the computational results obtained were rather discouraging. The inherent degeneracy of the problem, however, is beneficial as it leaves freedom in choosing the blocking variable (in Lemke's method fixing the blocking variable also fixes the next variable entering the basis), and this property is exploited to develop a variant of Lemke's algorithm: the look-ahead pivot rule which we will now shortly describe, for the readers' convenience.

As customary, we will use a superscript for the problem data and, to simplify notation, subscripts indicating the dependence on graph G will be omitted. Hence, q^v and M^v will identify the situation after v pivots and A^v will indicate the leading principal $n \times n$ submatrix of M^v. Consistently, y^v and x^v will indicate the vectors of basic and non-basic variables, respectively, each made up of a combination of the original x_i and y_i variables. The notation $\langle x_i^v, y_j^v \rangle$ will be used to indicate pivoting transformations. The index set of the basic variables that satisfy the min-ratio test at iteration v will be denoted by Ω^v, i.e.

$$\Omega^v = \arg\min_i \left\{ \frac{-q_i^v}{m_{is}^v} : m_{is}^v < 0 \right\},$$

Algorithm 2.1. A reduced version of Lemke's Scheme I with the look-ahead rule, applied to the MWCP.

```
Input: A graph $G = (V, E, w)$ and $h \in V$.
Let $v \leftarrow 0$. $K \leftarrow \emptyset$. $x_p^0 \leftarrow x_h$.
Infinite loop
    Let $x_p^v$ denote the driving variable.
    $\Omega^v = \{i : m_{ip}^v < 0\}$.
    If $\Omega^v \subseteq \{h\}$ stop: the result is $K$.
    $\Phi^v = \arg\min_i \{|\Omega^v| - |\Omega_i^{v+1}| > 0 : i \in \Omega^v\}$.
    $r = \min \Phi^v$
    If $y_r^v \equiv x_i$ for some $i$, then $K \leftarrow K \setminus \{i\}$
    If $x_p^v \equiv x_i$ for some $i$, then $K \leftarrow K \cup \{i\}$
    Perform $\langle y_r^v \rangle x_p^v$,
    The new driving variable is the variable
    complementary to $y_r^v$
    (recall $y_i$ is the complementary variable to $x_i$
    and vice versa)
    $v \leftarrow v + 1$
```

where s is the index of the driving column, i.e. the column related to the next variable entering the basis. The non-degeneracy assumption basically amounts to having $|\Omega^v| = 1$ for all v, thereby excluding any cycling behavior.

PBH uses the least-index rule, which amounts to blocking the driving variable with a basic one that has minimum index within a certain subset of Ω^v, i.e. $r = \min \Phi^v$ for some $\Phi^v \subseteq \Omega^v$. The set Φ^v is chosen in order to make the number of degenerate variables decrease as slowly as possible, i.e. among the index set

$$\Phi^v = \arg\min_i\{|\Omega^v| - |\Omega_i^{v+1}| > 0 : i \in \Omega^v\} \subseteq \Omega^v,$$

where Ω_i^{v+1} is the index set of those variables that would satisfy the min-ratio test at iteration $v+1$ if the driving variable at iteration v were blocked with y_i^v as $i \in \Omega^v$. The previous condition implies that a pivot step is taken and then reset in a sort of look-ahead fashion, hence we refer to this rule as the *look-ahead* (*pivot*) *rule*. The resulting procedure is specified as Algorithm 2.1. Empirical evidence indicated h, the index of the first driving variable, as a key parameter for the quality of the final result of Algorithm 2.1. Unfortunately no effective means could be identified to restrict the choice of vertices in V that can guarantee a good sub-optimal solution, so one has to consider iterating for most, if not all, vertices of V as outlined in Algorithm 2.2, which is the PBH. A simple criterion avoids considering those nodes $h \in V$ for which Algorithm 2.1 with input node h is certainly not able to produce a better clique with respect to the best known one, because the weight of h and that of its neighborhood is too small. Unfortunately, this criterion is effective only for very sparse graphs. It has been observed that the schema is sensitive to the ordering of nodes. The best figures were obtained by reordering G by

Algorithm 2.2. The PBH for the MWCP.

```
**Input:** A graph $G = (V, E, w)$.
Let $G' = (V', E', w')$ be a permutation of $G$
such that (4) holds.
$K^* \leftarrow \emptyset$.
For $v' = 1, \dots, n : W(\{v'\} \cup N_{v'}) > W(K^*)$ do
        Run Algorithm 2.1 with $G'$ and $v'$ as input.
        Let $K$ be the obtained result.
        If $W(K) > W(K^*)$, then $K^* \leftarrow K$.
The result is the mapping of $K^*$ in $G$.
```

decreasing weight of each node and its neighborhood, i.e.

$$W(\{u'\} \cup N_{u'}) \geqslant W(\{v'\} \cup N_{v'}) \quad \text{for all} \quad u', v' \in V' : u' < v', \tag{4}$$

where

$N_j = \{k \in V : (k, j) \in E\}$

is the set of all neighbors of vertex j.

2.2. A combinatorial multistart greedy heuristic

Given an input $h \in V$, the following greedy heuristic for MWCP returns a maximal clique.

GH(h)

Step 1 : Set $p = h$ and $K = \emptyset$.

Step 2 : If $K \cup \{p\}$ is a maximal clique, then set $K = K \cup \{p\}$ and return K; otherwise go to Step 3.

Step 3 : Let

$$M_0(K \cup \{p\}) = \arg\max_{j \in C_0(K \cup \{p\})} W(C_0(K \cup \{p\}) \cap N_j),$$

where

$$C_0(K \cup \{p\}) = \{j \in V : (j, k) \in E \text{ for all } k \in K \cup \{p\}\} = \bigcap_{k \in K \cup \{p\}} N_k$$

is the set of all vertices in G adjacent to each vertex in the current clique $K \cup \{p\}$. Then, select, according to the least-index rule, a vertex $\ell \in M_0(K \cup \{p\})$.

Step 4 : Set $K = K \cup \{p\}$ and $p = \ell$ and go back to Step 2.

Basically, at each iteration GH adds to the current clique $K \cup \{p\}$ the least-index vertex in set $M_0(K \cup \{p\})$, i.e. within the subset of the vertices connected to each vertex in the current clique (the set $C_0(K \cup \{p\})$) with maximum weight of the intersection of their neighbor set with $C_0(K \cup \{p\})$. The following multistart greedy heuristic (MGH) for MWCP, after reordering the vertices of the graph in such a way that (4) holds, simply runs GH for each $h \in V$.

MGH

Step 1 : Given a weighted graph $G = (V, E, w)$, let $G' = (V', E', w')$ be a permutation of G such that (4) holds. Let $Q = V'$. Set $K^* = \emptyset$, $h = 0$ and max $= 0$.

Step 2 : If $h \leqslant n$, go to Step 3; otherwise, return max and K^*.
Step 3 : Run GH(h) and let K_h be the returned maximal clique. If $|K_h| > \max$, set $\max = |K_h|$ and $K^* = K_h$. Set $h = h + 1$ and go back to Step 2.

We underline here that in the unweighted case, MCP, if the least-index rule for the selection of vertex ℓ in Step 3 of GH is substituted by a random choice, then MGH corresponds to heuristic SM^1 belonging to the class of heuristics SM^i presented in [3].

In the following subsection we will prove that PBH and MGH are equivalent, i.e. they always return the same solution.

2.3. Equivalence between PBH and MGH

In order to prove the equivalence between PBH and MGH we first need to introduce some notation and two lemmas.

Given a set $S \subseteq V$ of vertices, abbreviate by $S_i = S \setminus N_i$ the vertices in S that are not adjacent to i, and denote by

$$\tilde{d}_S(i) = \sum_{j \in S_i} \left(1 + \frac{w_j}{w_i}\right), \quad i \in V \setminus S$$

twice the average of weighted and unweighted co-degree of i w.r.t. S. Note that if $S_i \neq \emptyset$, then $\tilde{d}_S(i) \geqslant 1$. Further, for $i \neq j$ we have $(1/2w_j)\tilde{d}_{\{j\}}(i) = q_{ij}$ as defined in (2).

Let us represent the relevant part of the tableau at iteration v of Algorithm 2.1 in Table 1, where x_p denotes the driving variable. Notice that, without loss of generality, the vertices of the graph can always be renumbered in such a way that the variables which entered the basis up to iteration v are those corresponding to vertices 1 up to $p - 1$; these variables entered the basis in the same order of the vertices (i.e. x_1 first, then x_2, and so on); the driving variable x_p is the one corresponding to vertex p; the relative order of the indices of all other variables is left unchanged. Note that each entry $a^v_{i,j}$ of the tableau corresponds to the following couple of variables:

$$(x_{i-1}, x_j) \quad \text{for } i = 1, \ldots, p \quad \text{and} \quad j = p, \ldots, n,$$

$$(y_i, x_j) \quad \text{for } i = p+1, \ldots, n \quad \text{and} \quad j = p, \ldots, n,$$

$$(x_{i-1}, y_{j+1}) \quad \text{for } i = 1, \ldots, p \quad \text{and} \quad j = 1, \ldots, p-1,$$

$$(y_i, y_{j+1}) \quad \text{for } i = p+1, \ldots, n \quad \text{and} \quad j = 1, \ldots, p-1,$$

where $x_0 \equiv x_{n+1}$.

Given the situation displayed in the tableau of Table 1 we are now able to present the following lemmas, whose technical proofs are given in [6].

Lemma 1. *At iteration v of Algorithm* 2.1, *let x_p be the driving variable and $K = \{i \in V : x_i$ is basic$\} = \{1, \ldots, p-1\}$. Then, the following statements are true*:

1. *For all $i \notin K \cup \{p\}$*
$$a^v_{i,p} = \frac{1}{2w_p}\left[\tilde{d}_{K \cup \{p\}}(i) - 1\right],$$
which can be negative only if $a^v_{i,p} = -\frac{1}{2w_p}$, i.e. when $K \cup \{p\} \setminus N_i = \emptyset$ or, equivalently, vertex i is adjacent to any vertex in $K \cup \{p\}$.
2. *For all $i \notin K \cup \{p\}$ such that $a^v_{i,p} < 0$ (or equivalently, in view of point 1., $a^v_{i,p} = -\frac{1}{2w_p}$), it holds that*
$$a^v_{i,i} = \frac{1}{2w_i}.$$
3. *For all $i \notin K \cup \{p\}$ such that $a^v_{i,p} < 0$, it holds that*
$$a^v_{s,i} = \frac{1}{2w_i}\tilde{d}_{\{i\}}(s) = q_{is} \quad \textit{for all } s \notin K \cup \{i, p\}.$$

Proof. See the proof of Proposition 2.2 in [6]. □

Lemma 2. *At iteration v of Algorithm*, 2.1 *let x_p be the driving variable and $K = \{i \in V : x_i$ is basic$\} = \{1, \ldots, p-1\}$. Then, the following statements are true*:

1. *For any $j \in K$*
$$a^v_{j+1,p} = \frac{w_j}{w_p}.$$

Table 1
Tableau at iteration v of Algorithm 2.1

	q	y_2	$\cdots$	y_p	x_p	$\cdots$	x_n	y_1	x_{n+2}	y_{n+1}
x_{n+1}	1	$a^v_{1,1}$		$a^v_{1,p-1}$	$a^v_{1,p}$		$a^v_{1,n}$			
x_1	0	$a^v_{2,1}$		$a^v_{2,p-1}$	$a^v_{2,p}$		$a^v_{2,n}$			
$\vdots$	$\vdots$									
x_{p-1}	0	$a^v_{p,1}$		$a^v_{p,p-1}$	$a^v_{p,p}$		$a^v_{p,n}$			
y_{p+1}	0	$a^v_{p+1,1}$		$a^v_{p+1,p-1}$	$a^v_{p+1,p}$		$a^v_{p+1,n}$			
$\vdots$	$\vdots$									
y_n	0	$a^v_{n,1}$		$a^v_{n,p-1}$	$a^v_{n,p}$		$a^v_{n,n}$			
θ	1									
y_{n+2}	2									

2, *If* $a^v_{t,p} < 0,\ t \neq 1$, *then*

$$for\ all\ j \in K\ we\ have\ a^v_{j+1,t} = 0.$$

3. *If* $a^v_{t,p} < 0,\ t \neq 1$, *then* $a^v_{1,t} = a^1_{1,t} = -1$.

4. $a^v_{1,p} = \dfrac{1}{2w_p}\left[1 - 2W(K \cup \{p\})\right]$.

Proof. See the proof of Proposition 2.3 in [6]. □

Now we are ready to prove the equivalence between PBH and MGH.

Theorem 2. *PBH and MGH always return the same solution.*

Proof. Notice that both PBH and MGH are multistart algorithms that run a subroutine (Algorithm 2.1 for PBH and GH for MGH) for each $h \in V$. Therefore, in order to prove their equivalence it suffices to prove that, given the same input $h \in V$, Algorithm 2.1 and GH return the same maximal clique.

First we observe that, as conjectured in [7], in Algorithm 2.1 once a x_i variable with $i \in V$ enters the basis, it never exits. This follows from point 1 of Lemma 2 where each entry $a^v_{j+1,p}$ in the column of the driving variable and related to a variable x_j in the basis is equal to $w_j/w_p > 0$ so that x_j cannot exit the basis. This fact allows to establish an equivalence between the set K containing the indices of all variables x_i within the current basis in Algorithm 2.1, and the clique K in GH: in both cases once a vertex/variable enters the clique/basis, i.e. it enters set K, it will never be removed from this set.

Next, we notice that in Algorithm 2.1 the set Ω^v of candidates to become the new driving variable, i.e. with $a^v_{i,p} < 0$, are only the variables related to vertices i which form a clique with the vertices in $K \cup \{p\}$. This follows from point 1 of Lemma 1 from which $a^v_{i,p} < 0$ if and only if $[K \cup \{p\}]_i = \emptyset$ which exactly means that i is connected with every vertex in $K \cup \{p\}$. In other words, $\Omega^v \equiv C_0(K \cup \{p\})$, i.e. for a given set K and vertex p, the set of candidates to enter the basis/clique in Algorithm 2.1 and GH is the same. Note that the above result also shows that in Algorithm 2.1 the set K is always a clique.

But we can also extend this result. Indeed, it is immediate to see that also the equivalence $\Phi^v \equiv M_0(K \cup \{p\})$ holds. Since both Algorithm 2.1 and GH employ the least-index rule and in both cases the vertices are ordered according to (4), when the set K and the vertex p are the same for the two algorithms, they select the same vertex ℓ.

Finally, since we have assumed that the input $h \in V$ is the same for Algorithm 2.1 and GH, an easy inductive proof shows that at each iteration these two algorithms always select the same vertex and they will finally return the same maximal clique. Hence the assertion is proved. □

In [7] it was also conjectured that saddle points detected by Algorithm 2.1 are always maximal cliques, and thus local solutions, even if problem (1) has stationary points which are not local solutions. The above proof shows that the conjecture is true, as stated in the following corollary.

Corollary 1. *For any starting driving variable* x_h, *Algorithm* 2.1 *returns a local solution of problem* (1).

References

[1] I.M. Bomze, M. Budinich, P.M. Pardalos, M. Pelillo, The maximum clique problem, in: D.-Z. Du, P.M. Pardalos, (Eds.), Handbook of Combinatorial Optimization—Suppl, Vol. A, Kluwer Academic Publishers, Dordrecht, 1999, pp. 1–74.

[2] I.M. Bomze, M. Pelillo, V. Stix, Approximating the maximum weight clique using replicator dynamics, IEEE Trans. Neural Networks 11 (2000) 1228–1241.

[3] M. Brockington, J.C. Culberson, Camouflaging independent sets in quasi-random graphs, in: D. Johnson, M.A. Trick (Eds.), Cliques, Coloring and Satisfiability, DIMACS Series, Vol. 26, AMS, Providence, RI, 1996, pp.75–88.

[4] R.W. Cottle, J. Pang, R.E. Stone, The Linear Complementarity Problem, Academic Press, Boston, MA, 1992.

[5] D. Johnson, M.A. Trick, (Eds.), Cliques, Coloring and Satisfiability: Second DIMACS Implementation Challenge, DIMACS Series, Vol. 26, AMS, Providence, RI, 1996.

[6] M.Locatelli, I.M. Bomze, M. Pelillo, Swaps, diversification and the combinatorics of pivoting for the maximum weight clique, Technical Report CS-2002-12, Dipartimento di Informatica, Universitá Ca' Foscari di Venezia, 30172 Venezia Mestre (VE), Italy, also available at the web site http://www.di.unito.it/∼locatell/combpivot6.ps 2002.

[7] A. Massaro, M. Pelillo, I.M. Bomze, A complementary pivoting approach to the maximum weight clique problem, SIAM J. Optim. 12 (2002) 928–948.