Handbook of Combinatorial Optimization
D.-Z. Du and P.M. Pardalos (Eds.) pp. – – –

The Maximum Clique Problem

Immanuel M. Bomze
Institut für Statistik, Operations Research und Computerverfahren
Universität Wien
Universitätsstraße 5, A-1010 Wien, Austria
E-mail: `immanuel.bomze@univie.ac.at`

Marco Budinich
Dipartimento di Fisica
Università di Trieste
Via Valerio 2, I-34127 Trieste, Italy
E-mail: `mbh@trieste.infn.it`

Panos M. Pardalos
Center for Applied Optimization, ISE Department
University of Florida, Gainesville, FL 32611
E-mail: `pardalos@ufl.edu`

Marcello Pelillo
Dipartimento di Informatica
Università Ca' Foscari di Venezia
Via Torino 155, 30172 Venezia Mestre, Italy
E-mail: `pelillo@dsi.unive.it`

Contents

1 Introduction

The maximum clique problem is a classical problem in combinatorial optimization which finds important applications in different domains. In this paper we try to give a survey of results concerning algorithms, complexity, and applications of this problem, and also provide an updated bibliography. Of course, we build upon precursory works with similar goals [39, 232, 266].

1.1 Notations and Definitions

Throughout this paper, $G = (V, E)$ is an arbitrary undirected and weighted graph unless otherwise specified, where $V = \{1, 2, \ldots, n\}$ is the vertex set of G (we use the terms *vertex* and *node* synonymously throughout), and $E \subseteq V \times V$ is the edge set of G. For each vertex $i \in V$, a positive weight w_i is associated with i, collected in the *weight vector* $w \in \mathbb{R}^n$. The symmetric $n \times n$ matrix $A_G = (a_{ij})_{(i,j) \in V \times V}$, where $a_{ij} = 1$ if $(i, j) \in E$ is an edge of G, and $a_{ij} = 0$ if $(i, j) \notin E$, is called the *adjacency matrix* of G. For any node v, let

$$N(v) = \{j \in V : a_{vj} = 1\}$$

denote the *neighborhood* of v in G, i.e. the set of all nodes adjacent to v.

The *complement graph of* $G = (V, E)$ is the graph $\overline{G} = (V, \overline{E})$, where $\overline{E} = \{(i, j) \mid i, j \in V,\ i \neq j \text{ and } (i, j) \notin E\}$. For a subset $S \subseteq V$, we define the weight of S to be $W(S) = \sum_{i \in S} w_i$, and call $G(S) = (S, E \cap S \times S)$ *the subgraph induced by* S.

A graph $G = (V, E)$ is *complete* if all its vertices are pairwise adjacent, i.e. $\forall i, j \in V$ with $i \neq j$, we have $(i, j) \in E$. A *clique* C is a subset of V such that $G(C)$ is complete. The *clique number* of G, denoted by $\omega(G)$ is the size of the maximum clique. The maximum clique problem asks for cliques of maximum cardinality (the cardinality of a set S, i.e., the number of its elements which will be denoted by $|S|$):

$$\omega(G) = \max\{|S| : S \text{ is a clique in } G\}\,.$$

The maximum weight clique problem asks for cliques of maximum weight. Given the weight vector $w \in \mathbb{R}^n$, the *weighted clique number* is the total weight of the maximum weight clique, and will be denoted by $\omega(G, w)$:

$$\omega(G, w) = \max\{W(S) : S \text{ is a clique in } G\}\,.$$

An *independent set* (*stable set, vertex packing*) is a subset of V, whose elements are pairwise nonadjacent. The maximum independent set problem asks for an independent set of maximum cardinality. The size of a maximum independent set is the *stability number* of G (denoted by $\alpha(G)$). The maximum weight independent set problem asks for an independent set of maximum weight. A *vertex cover* is a subset of V, such that every edge $(i, j) \in E$ has at least one endpoint i or j in the subset. The minimum vertex cover problem asks for a vertex cover of minimum cardinality. The minimum weighted vertex cover problem asks for a vertex cover of minimum weight.

It is easy to see that S is a clique of G if and only if S is an independent set of $\overline{G}$, and if and only if $V \setminus S$ is a vertex cover of $\overline{G}$. Any result obtained for one of the above problems has its equivalent forms for the other problems. Hence $\alpha(G) = \omega(\overline{G})$, and, more generally, $\alpha(G,w) = \omega(\overline{G},w)$.

We should distinguish a *maximum* clique (independent set) from a *maximal* clique (independent set). A maximal clique (independent set) is a clique (independent set) that is not a proper subset of any other clique (independent set). A maximum (weight) clique (independent set) is a maximal clique (independent set) that has the maximum cardinality (weight).

In the sequel, it will prove useful to consider Δ, the standard simplex in the n-dimensional Euclidean space $\mathbb{R}^n$:

$$\Delta = \left\{ x \in \mathbb{R}^n \ : \ x_i \geq 0 \text{ for all } i \in V, \ e^T x = 1 \right\}$$

where e^T denotes the transpose of the vector e consisting of unit entries (hence $e^T := (1, 1, \ldots, 1)$ and $e^T x = \sum_{i \in V} x_i$). For a subset $S \subseteq V$ of vertices, we shall denote the face of Δ corresponding to S by

$$\Delta_S = \{ x \in \Delta : x_i = 0 \text{ if } i \notin S \} \ .$$

Further, let us introduce the notion of a *characteristic vector* x^S which is the vector in Δ defined by $x_i^S = 1/|S|$ if $i \in S$ and $x_i^S = 0$ otherwise. Likewise, given a weight vector $w \in \mathbb{R}^n$, denote by $x^{S,w}$ the *weight barycenter* of Δ_S or, synonymously, the *weighted characteristic vector* of S, which is a vector with coordinates $x_i^{S,w} = w_i/W(S)$ if $i \in S$, and $x_i^{S,w} = 0$ otherwise. Of course, we have $x^S = x^{S,e}$.

2 Problem Formulations

The maximum clique problem has many equivalent formulations as an integer programming problem, or as a continuous nonconvex optimization problem. As with many problems of combinatorial optimization, using the appropriate formulation is of crucial importance in solving the problem. In addition, using different formulations, we gain more insight into the problem's complexity and we can prove interesting results.

2.1 Integer Programming Formulations

The simplest formulation of the maximum clique problem is the following *edge formulation*:

$$\max \sum_{i=1}^{n} w_i x_i, \tag{1}$$

$$\text{s.t. } x_i + x_j \leq 1, \ \forall \ (i,j) \in \overline{E},$$

$$x_i \in \{0,1\}, \ i = 1, \ldots, n.$$

A polyhedral result concerning formulation (1) is due to Nemhauser and Trotter [244, 245]. In 1975 they found that if a variable x_i had integer value 1 in an optimal solution to the linear relaxation of (1), then $x_i = 1$ in at least one optimal solution to (1).

Theorem 2.1 (see [245], also [278]) *Let x be an optimum $(0, \frac{1}{2}, 1)$-valued solution to the linear relaxation of* (1)*, and let $P = \{j \mid x_j = 1\}$. Then there exists an optimum solution x^* to* (1) *such that $x_j^* = 1$, $\forall j \in P$.*

This theorem suggests an implicit enumerative algorithm for (1) via solving its linear relaxation problem. However, in most cases, few variables have integer values in an optimal solution to the linear relaxation of (1), and the gap between the optimal values of (1) and its linear relaxation problem is too large, which seriously restrict the use of this approach.

Let $\mathcal{S}$ denote the set of all maximal independent sets of G. An alternative formulation is the following *independent set* formulation.

$$\max \sum_{i=1}^{n} w_i x_i, \tag{2}$$

$$\text{s.t. } \sum_{i \in S} x_i \leq 1, \ \forall S \in \mathcal{S},$$

$$x_i \in \{0,1\}, \ i = 1, \ldots, n.$$

The advantage of formulation (2) over (1) is a smaller gap between the optimal values of (2) and its linear relaxation. However, since the number of constraints in (2) is exponential, solving the linear relaxation of (2) is not an easy problem. In fact, Grötschel et al. [150, 151] have shown that the linear relaxation problem of (2) is NP-hard on general graphs. They have also shown that the same problem is polynomially solvable on *perfect* graphs. Furthermore, they have shown that a graph is perfect if and only if the optimal solution to the linear relaxation of (2) always takes integer values (see also [149] and [152]).

In 1990, Shor [293] considered an interesting formulation of the maximum weight independent set problem. Note that the maximum weight independent set problem can be formulated as

$$\min f(x) = \sum_{i=1}^{n} w_i x_i, \tag{3}$$

$$\text{s.t. } x_i + x_j \leq 1, \ \forall\ (i,j) \in E, \ x \in \{0,1\}^n.$$

The above formulation is equivalent to the following quadratically constrained global optimization problem

$$\min f(x) = \sum_{i=1}^{n} w_i x_i, \tag{4}$$

$$\text{s.t. } x_i x_j = 0, \ \forall\ (i,j) \in E,$$

$${x_i}^2 - x_i = 0, \ i = 1, 2, ..., n.$$

Applying dual quadratic estimates, Shor reported very good computational results using (4).

Next, we formulate the maximum clique, the maximum independent set and the maximum weight independent set problems as quadratic zero-one problems. We use I to denote the $n \times n$ identity matrix. To facilitate our discussion, define a transformation t from $\{0,1\}^n$ to 2^V,

$$t(x) = \{i \in V : x_i = 1\}, \ \forall\ x \in \{0,1\}^n.$$

Denote the inverse of t by t^{-1}. If $x = t^{-1}(S)$ for some $S \in 2^V$ then $x_i = 1$ if $i \in S$ and $x_i = 0$ if $i \notin S$, $i = 1, \ldots, n$, i.e. $x = |S| x^S$ (cf. Section 1).

We can rewrite the maximum problem (1) as a minimization problem (when $x_i = 1$)

$$\min f(x) = -\sum_{i=1}^{n} x_i, \tag{5}$$

$$\text{s.t. } x_i + x_j \leq 1, \ \forall\ (i,j) \in \overline{E}, \ x \in \{0,1\}^n.$$

If x^* solves (5), then the set $C = t(x^*)$ is a maximum clique of G with $|C| = -z = -f(x^*)$.

Another way of stating the constraints for (5) is to make use of the fact that the quadratic expressions $x_i x_j = 0$ for all $(i,j) \in \overline{E}$, since for $x_i, x_j \in \{0,1\}$, we have $x_i + x_j \leq 1$ if and only if $x_i x_j = 0$. The constraints in (5) can be removed by adding two times the quadratic terms to the objective function, which is now

$$f(x) = -\sum_{i=1}^{n} x_i + 2 \sum_{(i,j)\in\overline{E},\, i>j} x_i x_j = x^T (A_{\overline{G}} - I)x.$$

The quadratic terms represent penalties for violations of $x_i x_j = 0$. This leads to the following theorem.

Theorem 2.2 *The maximum clique problem is equivalent to the following global quadratic zero-one problem*

$$\min f(x) = x^T Ax, \tag{6}$$

$$s.t.\ x \in \{0,1\}^n,\ \textit{where } A = A_{\overline{G}} - I.$$

If x^* solves (6), then the set C defined by $C = t(x^*)$ is a maximum clique of G with $|C| = -z = -f(x^*)$.

The off-diagonal elements of the matrix A are the same as the adjacency matrix of $\overline{G}$. Hence, formulations (5) and (6) are advantageous for dense graphs because a sparse data structure can be used (for details, see [264]; cf. also [123]). Following the equivalence of the maximum clique problem with the maximum independent set problem, we have

Theorem 2.3 *The maximum independent set problem is equivalent to the following global quadratic zero-one problem*

$$\min f(x) = x^T Ax \tag{7}$$

$$s.t.\ x \in \{0,1\}^n,\ \textit{where } A = A_G - I.$$

If x^* solves (7), then the set S defined by $S = t(x^*)$ is a maximum independent set of G with $|S| = -z = -f(x^*)$.

Next, we discuss the maximum weight independent set problem. The above theorems for the maximum clique problem and the maximum independent set problem can be regarded as a special case by taking $w = e$.

Theorem 2.4 *The maximum weight independent set problem is equivalent to the following global quadratic zero-one problem*

$$\min f(x) = x^T Ax, \tag{8}$$

$$s.t.\ x \in \{0,1\}^n,$$

where $a_{ii} = -w_i,\ i = 1, ..., n,\ a_{ij} = \frac{1}{2}(w_i + w_j),\ \forall\ (i,j) \in E,$ *and* $a_{ij} = 0,\ \forall\ (i,j) \in \overline{E}$.

Let x^* *solve* (8), *then the set* S *defined by* $S = t(x^*)$ *is a maximum weight independent set of* G *with weight* $W(S) = -z = -f(x^*)$.

In addition, we have the following relationship between the local minima of the quadratic problem and the maximal independent sets of a graph:

Theorem 2.5 x *is a discrete local solution to problem* (8) *if and only if* x *represents a maximal independent set of* G.

2.2 Continuous Formulations

In 1965, Motzkin and Straus [238] established a remarkable connection between the maximum clique problem and a certain standard quadratic programming problem providing an alternative proof of a slightly weaker version of the fundamental Turán theorem [311] (for a discussion see [54]). Let $G = (V, E)$ be an undirected (unweighted) graph, and let Δ denote the standard simplex in the n-dimensional Euclidean space $\mathbb{R}^n$, as well as Δ_S the face of Δ corresponding to a subset $S \subseteq V$ (cf. Section 1). Now, consider the following quadratic function

$$g(x) = x^T A_G x \tag{9}$$

where $A_G = (a_{ij})_{i,j \in V}$ is the adjacency matrix of G, and let x^* be a global maximizer of g on Δ. Motzkin and Straus proved that the clique number of G is related to $g(x^*)$ by the following formula:

$$\omega(G) = \frac{1}{1 - g(x^*)} \geq \frac{1}{1 - g(x)}\ \forall x \in \Delta\ . \tag{10}$$

Additionally, they proved that a subset of vertices S is a maximum clique of G if and only if its characteristic vector x^S (see Section 1) is a global maximizer of g on Δ. In [274, 135], the Motzkin-Straus theorem has been extended by providing a characterization of *maximal* cliques in terms of *local* maximizers of g on Δ. Moreover, in [135] Gibbons et al. characterize the first- and second-order optimality conditions of the Motzkin-Straus program and of a newly introduced parametrized version. A further generalization of the Motzkin-Straus theorem to hypergraphs can be found in [299].

One drawback associated with the original Motzkin-Straus formulation relates to the existence of spurious solutions, i.e., maximizers of g which are not in the form of characteristic vectors. This was observed empirically by Pardalos and Phillips [261] and more recently formalized by Pelillo and Jagota [274]. In principle, spurious solutions represent a problem since, while providing information about the size of the maximum clique, they do not allow us to easily extract its vertices.

The spurious solution problem has recently been solved by Bomze [56], by considering the following regularized version of function g:

$$\hat{g}(x) = x^T A_G x + \frac{1}{2} x^T x \tag{11}$$

which is obtained from (9) by substituting the adjacency matrix A_G of G with

$$\hat{A}_G = A_G + \frac{1}{2} I \tag{12}$$

where I is the identity matrix. The following is the spurious-free counterpart of the original Motzkin-Straus theorem [56].

Theorem 2.6 *Let S be a subset of vertices of a graph G, and let x^S be its characteristic vector. Then the following statements hold:*

(a) S is a maximum clique of G if and only if x^S is a global maximizer of the function $\hat{g}$ over the simplex Δ. In this case, $\omega(G) = 1/2(1-\hat{g}(x^S))$.

(b) S is a maximal clique of G if and only if x^S is a local maximizer of $\hat{g}$ in Δ.

(c) All local (and hence global) maximizers x of $\hat{g}$ over Δ are strict, and of the form $x = x^S$ for some $S \subseteq V$.

Unlike the Motzkin-Straus formulation, the previous result guarantees that *all* maximizers of $\hat{g}$ on Δ are strict, and are characteristic vectors of maximal/maximum cliques in the graph. In an exact sense, therefore, a one-to-one correspondence exists between maximal cliques and local maximizers of $\hat{g}$ in Δ on the one hand, and maximum cliques and global maximizers on the other hand. This solves the spurious solution problem in a definitive manner.

In a recent paper, Gibbons et al. [135] generalized the Motzkin-Straus theorem to the weighted case. They first reformulated the Motzkin-Straus problem as a minimization problem by considering the function

$$f(x) = x^T(I + A_{\overline{G}})x \tag{13}$$

where $A_{\overline{G}}$ is the adjacency matrix of the complement graph $\overline{G}$. It is straightforward to see that if x^* is a global minimizer of f in Δ, then we have:

$$\omega(G) = \frac{1}{f(x^*)} \ .$$

This is simply a different formulation of the Motzkin-Straus theorem. Given a weighted graph $G = (V, E)$ with weight vector w, they then considered the following classes of symmetric $n \times n$ matrices: let

$$\mathcal{M}(G) = \left\{ (b_{ij})_{i,j\in V} : b_{ij} \geq \frac{b_{ii} + b_{jj}}{2} \text{ if } (i,j) \notin E \,, b_{ij} = 0, \text{ otherwise} \right\} ,$$

and put

$$\mathcal{M}(G, w) = \left\{ B = (b_{ij})_{i,j\in V} \in \mathcal{M}(G) : b_{ii} = \frac{1}{w_i} \text{ and } b_{ij} = b_{ji} \text{ for all } i, j \right\} .$$

Now the following (generally indefinite) quadratic program is introduced in [135]:

$$\begin{array}{ll} \text{minimize} & f(x) = x^T B x \\ \text{subject to} & x \in \Delta \end{array} \tag{14}$$

where $B \in \mathcal{M}(G, w)$.

Using a proof technique suggested by Lovász (cf. [219]), the following result is shown, which establishes a correspondence between maximum weight cliques and global minimizers of (14).

Theorem 2.7 *Let G be an arbitrary weighted graph with positive weight vector $w \in I\!R^n$. Then, for any $B \in \mathcal{M}(G, w)$ we have: $\omega(G, w) = \frac{1}{f(x^*)}$ where x^* is a global minimizer of program (14).*

It can be seen that a subset S of vertices of a weighted graph G is a maximum weight clique if and only if its weighted characteristic vector $x^{S,w}$ (cf. Section 1) is a global minimizer of (14). Notice that the matrix $I + A_{\overline{G}}$

belongs to $\mathcal{M}(G, e)$. In other words, the Motzkin-Straus theorem turns out to be a special case of the preceding result.

As in the unweighted case, the existence of spurious solutions entails the lack of one-to-one correspondence between the solutions of the continuous problem and those of the original, discrete one. Bomze et al. [62] have recently characterized these spurious solutions and have introduced and studied a regularized version which avoids this kind of problems, exactly as in the unweighted case (for proofs see also [58]): instead of the Motzkin-Straus class $\mathcal{M}(G, w)$ here a different class $\mathcal{C}(G, w)$ of matrices is considered to be used as input data for problem (14): let

$$\mathcal{C}(G) = \{(c_{ij})_{i,j \in V} : c_{ij} \geq c_{ii} + c_{jj} \text{ if } (i,j) \notin E \,, c_{ij} = 0, \text{ otherwise } \} \;,$$

and consider, given the weights w,

$$\mathcal{C}(G, w) = \left\{ C = (c_{ij})_{i,j \in V} \in \mathcal{C}(G) : c_{ii} = \frac{1}{2w_i} \text{ and } c_{ij} = c_{ji} \text{ for all } i, j \right\} .$$

Theorem 2.8 *Let G be an arbitrary graph with positive weight vector $w \in \mathbb{R}^n$, and consider a matrix $C \in \mathcal{C}(G, w)$ in place of B for problem (14). Then the following assertions hold:*

(a) A vector $x \in \Delta$ is a local solution to problem (14) if and only if $x = x^{S,w}$, where S is a maximal clique.

(b) A vector $x \in \Delta$ is a global solution to problem (14) if and only if $x = x^{S,w}$, where S is a maximum weight clique.

Moreover, all local (and hence global) solutions to (14) are strict.

The class $\mathcal{C}(G, w)$ is isomorphic to the positive orthant in $\binom{n}{2} - |E|$ dimensions. This class is a polyhedral pointed cone with its apex given by the matrix $C(w)$ with entries

$$c_{ij}(w) = \begin{cases} \frac{1}{2w_i} & \text{if } i = j \,, \\ \frac{1}{2w_i} + \frac{1}{2w_j} & \text{if } i \neq j \text{ and } (i,j) \notin E \,, \\ 0 & \text{otherwise.} \end{cases} \qquad (15)$$

Observe that in the unweighted case, $C(e) = ee^T - \hat{A}_G = \hat{A}_{\overline{G}}$, the regularized adjacency matrix of the complement graph $\overline{G}$. This reflects the elementary property that an independent set of G is a clique of $\overline{G}$. So, while the local maximizers of $x^T \hat{A}_G x$ over Δ are exactly the barycenters x^C of maximal cliques C of G, the local minimizers of $x^T \hat{A}_G x$ over Δ are exactly the barycenters x^S of maximal independent sets S of G. Compare with Theorem 2.5 at the end of the preceding section. Note that within the Motzkin-Straus class $\mathcal{M}(G, e)$, there is no matrix with this straightforward interpretation.

3 Computational Complexity

The maximum clique problem is one of the first problems shown to be NP-complete [194], which means that, unless $P = NP$, a fact which is widely believed to be false, exact algorithms are guaranteed to return a solution only in a time which increases exponentially with the number of vertices in the graph [126]. Of course, the maximum independent set and the minimum vertex cover problems are NP-complete too. For a recent complexity result on the class of planar counting problems which includes the minimum vertex cover problem see [182].

Interest therefore has soon shifted towards characterizing the approximation properties of this problem (see [255, 70, 16] for an introduction to approximation complexity in optimization problems). Early works in this area go back to mid-1970's when Garey and Johnson [125] proved that if the maximum clique problem admits a polynomial-time approximation algorithm (i.e., it is approximable within a constant factor), then it has a polynomial-time approximation scheme (namely, it is approximable within *any* arbitrarily small factor). This is concisely expressed by saying that if the maximum clique problem belongs to the class APX, then it belongs to PTAS.

Johnson et al. [190] showed in 1988, based on one of their results given below, the nonexistence of a polynomial-delay algorithm for enumerating maximal cliques in reverse lexicographic order (if $P \neq NP$).

Theorem 3.1 *Given a graph G and a maximal independent set S, it is coNP-complete to prove whether S is the lexicographically last maximal independent set of G.*

In [257, 258], Papadimitriou and Yannakakis introduced the complexity classes $MAX\ NP$ and $MAX\ SNP$. They showed that all problems in $MAX\ NP$ admit a polynomial-time approximation algorithm, and that many natural problems are complete in $MAX\ SNP$, under an approximation preserving reduction called the L-reduction. For example, the vertex cover problem (for constant degree graphs), max cut problem, dominating set problem, and the MAX 3-SAT problem are such complete problems [323]. Recently, this result has been extended to the vertex cover problem for cubic graphs [109].

If the solution to any of these complete problems can be approximated to arbitrary small constant factors, then the optimum solution to any problem in the class can be approximated to arbitrarily small constant factors. The question of whether such approximation schemes can be found for the complete problems in this class was left unresolved.

By using randomized reductions, Berman and Schnitger [45, 46] have shown that even an $O(n^{\varepsilon})$-approximation algorithm for maximum clique (for some small enough ε) would yield a *randomized* polynomial-time approximation scheme for MAX 3-SAT, where n is the number of vertices in the graph (see also Feige et al. [111], where a connection between approximation complexity and interactive proof systems is discussed). In [7], Alon et al. derandomized Berman and Schnitger's reduction by using the so-called *expander graphs* (also used by Arora and Safra in [14]), and obtained the following result: there exists an $\varepsilon > 0$ such that if the maximum clique problem is approximable within $O(n^{\varepsilon})$ then MAX 3-SAT is in PTAS.

Panconesi and Ranjan [253, 254] expanded on the work of Papadimitriou and Yannakakis. They introduced the complexity class $MAX\ \Pi_1$, and proved that all complete problems in this class do not admit a polynomial-time approximation algorithm, unless $P = NP$. They also showed that the maximum clique problem does not belong to $MAX\ NP$ but is in the lowest subclass of $MAX\ \Pi_1$, see also [313].

In 1991, Crescenzi, Fiorini and Silvestri [97] proved that all problems in $MAX\ NP$ are strongly reducible to the maximum clique problem. As a consequence, if the maximum clique is approximable within a constant factor, then all problems in $MAX\ NP$ can be approximated within any arbitrarily small factor. This provided evidence that the maximum clique problem does not have a polynomial-time approximation algorithm. Stronger evidence of this fact was given independently in the same year, when Feige et al. [111] (see also [112]) proved that if there is a polynomial-time algorithm that approximates the maximum clique problem within a factor of $2^{\log^{1-\varepsilon} n}$, then any NP problem can be solved in "quasi-polynomial" time (i.e., in $2^{\log^{O(1)} n}$ time).

A breakthrough in approximation complexity is the result by Arora et al. [13], [14]. It is shown that the maximum number of satisfiable clauses in a 3-SAT formula (MAX 3-SAT) cannot be approximated to arbitrary small constants (unless $P = NP$), thus resolving the open question in [257, 258]. This immediately shows the difficulty of finding good approximate solutions to all the above listed problems. In particular, it is shown that no polynomial-time algorithm can approximate the maximum clique size within a factor of n^{ε} ($\varepsilon > 0$), unless $P = NP$ (by using the results of Feige et al. [111]). The work of Arora et al. stimulated much research, and many investigators have progressively refined the exact approximation ratio for which approximating maximum clique becomes intractable [40, 42, 113, 41, 160].

The best polynomial-time approximation algorithm for the maximum clique problem was developed by Boppana and Halldórsson [66], and achieves an approximation ratio of $n^{1-o(1)}$. In [161], Hastad shows that this is actually the best we can achieve, by proving that, unless $NP = coR$, the maximum clique problem cannot be approximated within a factor of $n^{1-\varepsilon}$, for any $\varepsilon > 0$.

Although these complexity results characterize worst case instances, they nevertheless indicate that the maximum clique problem is indeed a very difficult problem to solve.

Some other results in the literature concerning the approximation of the maximum clique/independent set problem on arbitrary or special graphs can be found in [88], [90], [283], [97], [66] [240], [84], [212], [291], [85], [196].

If we restrict ourselves to graphs with special structure, then in many cases the maximum clique/independent set problem can be solved in polynomial time. For example, Balas et al. [25] introduced several classes of graphs and showed that the maximum weight clique problem can be solved in polynomial time on them. Balas and Yu [32] discuss classes of graphs that have polynomially many maximal cliques. On those graphs, the maximum weight clique problem can also be solved in polynomial time.

A well known class of graphs where the maximum clique problem is polynomially solvable, is the class of perfect graphs [43]. A graph G is called *perfect* if every induced subgraph of G has the property that the size of its maximum clique equals the minimum number of independent sets needed to cover all the vertices (commonly called a *coloring* in the literature). Since the complement graph of a perfect graph is also perfect, the maximum clique problem can be solved in polynomial time on perfect graphs and their complements. The class of perfect graphs contains many well known graphs in the literature [145], among them *bipartite* graphs, *interval* graphs, and *triangulated* graphs [127], [119], [287], [288], [306]. Examples of more recently found perfect graphs are *Meyniel graphs* [233], [78], *quasi parity* graphs [234], *weakly triangulated* graphs [163], [164], *perfectly orderable* graphs [95], and *unimodular* graphs [165].

A class of graphs that is closely related to the perfect graphs is the *t-perfect* graphs. This class of graphs was defined in [92]. Polynomial algorithms for the maximum weight independent set problem on *t-perfect* graphs exist [151]. The class of *t-perfect* graphs contains *bipartite graphs, series-parallel graphs* [106], [92], [67], and *strongly t-perfect graphs* [133]. For a polynomial time algorithm for solving the maximum weight independent set problem on a bipartite graph $G(V_1, V_2)$ see the book by Lawler [209].

Other special classes of graphs where the maximum clique/independent set problem have been studied in the literature can be found in [23], [48], [49], [79], [83], [88], [89], [91], [90], [107], [114], [128], [129], [146], [147], [153], [170], [179], [180], [181], [197], [196], [202], [223], [227], [228], [235], [242], [248], [256], [281], [289], [290], [303], [86], [102], and [324].

We should note here that the weighted or unweighted version of the maximum clique problem, the maximum independent set problem, and the minimum vertex cover problem may, with respect to hardness, not be equivalent on graphs with special structures.

4 Bounds and Estimates

We now present bounds for the clique number $\omega(G)$ and discuss also the complexity of their calculation; most of them are based on properties of the matrix A_G.

A graph is said to be connected if each pair of nodes is joined by a path of edges. Since the undirected graphs we are considering can always be subdivided into connected subgraphs, we consider here only connected graphs. These have adjacency matrices with the property of being irreducible.

Let m be the number of edges of the graph and $\delta = \frac{2m}{n^2}$ the density of 1's in A_G; for connected graphs we have $2\frac{n-1}{n^2} \leq \delta \leq \frac{n-1}{n}$. The request that a graph with a clique number $\omega(G)$ be connected gives the simple bound [12]

$$\omega(G) \leq \frac{3 + \sqrt{9 - 8(n-m)}}{2}. \tag{16}$$

Let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of A_G and $\rho(A_G) = \max_{i=1,\ldots,n} |\lambda_i|$ its spectral radius. A frequently cited upper bound, appeared for the first time in 1967 [316], is

$$\omega(G) \leq \rho(A_G) + 1 \tag{17}$$

the equality holding if and only if the graph is complete. To prove this relation we use (10): let x^S be the characteristic vector of a maximum clique, then $(x^S)^T A_G x^S = 1 - \frac{1}{\omega(G)}$ and also $(x^S)^T x^S = \frac{1}{\omega(G)}$. Bound (17) derives from the general property $\frac{x^T A_G x}{x^T x} \leq \rho(A_G)$. With (17) one can apply to $\omega(G)$ all the bounds valid for $\rho(A_G)$ (for a partial list see [75]).

Since A_G is irreducible, symmetric and with non-negative elements, all its eigenvalues are real and the largest one, the *Perron root* $\lambda_P := \rho(A_G)$, is simple and dominating (see e.g. [176] pp. 507 ff.). Moreover there may be at most one negative eigenvalue $-\lambda_P$ with the same absolute value, and this happens if and only if the graph is bipartite (see [100], Theorems 3.11 and 3.4). Finally, all the components of the only *Perron eigenvector* x_P (such that $A_G x_P = \lambda_P x_P$) are strictly positive, i.e. $x_P > 0$.

Straightforward calculation of Perron root and eigenvector is not the most efficient method for large n: it is faster to exploit the exponentially fast convergence of $\lim_{m\to\infty} \left(\frac{1}{\lambda_P} A_G\right)^m = x_P x_P^T$ valid if A_G is primitive (when this does not happen one takes the primitive matrix $\hat{A}_G = A_G + \frac{1}{2}I$). Successive squaring of A_G allows to perform this calculation, with arbitrary precision, essentially in $O(n^3)$ [176].

Let N_{-1} be the number of eigenvalues of A_G that do not exceed -1 and N_0 the number of zero eigenvalues. Amin and Hakimi [12] proved that

$$\omega(G) \leq N_{-1} + 1 < n - N_0 + 1\,, \tag{18}$$

with equality holding if the graph is complete multipartite; also the latter bound can be calculated in $O(n^3)$.

A geometrical formulation of the maximum clique problem [76] in complex space produces the bound

$$\omega(G) \leq \frac{n + \overline{N}_0}{2} \tag{19}$$

where $\overline{N}_0$ is the number of zero eigenvalues of the adjacency matrix of the complement graph $\overline{G}$. Also in this case the calculation of $\overline{N}_0$ can be done in $O(n^3)$.

While bound (16) can be calculated in $O(n)$, bounds (17), (18) and (19) require more work but are usually much tighter. Experimentally, most of the times (18) is the sharpest bound, but nothing can be said in general because one can devise graphs for which bounds (17) or (19) are respectively the best bound. Consequently the safest strategy is to calculate all bounds and to choose the sharpest.

A different bound derives from the 'Sandwich theorem' on which Knuth published a review [204]. This theorem focuses on the *Lovász number* [218] (cf. also [292]) $\theta(\overline{G})$ and states that this number is sandwiched between the two NP-hard quantities, the clique number $\omega(G)$ and the *chromatic number* $\chi(G)$, which is the minimum number of colors needed to color the vertices of G:

$$\omega(G) \leq \theta(\overline{G}) \leq \chi(G)\,. \tag{20}$$

Since $\theta(\overline{G})$ is computable in polynomial time, e.g. with interior point methods like *semidefinite programming* (see, e.g. [140, 203], and for special sparsity methods related to the maximum clique problem [123]) via the (dual) eigenvalue bound identity [166]

$$\theta(\overline{G}) = \min\{\lambda_{\max}(ee^T + Y) : Y \in \mathcal{Y}(G)\}$$

with

$$\mathcal{Y}(G) = \{Y\,(n \times n) \text{ matrix } : Y^T = Y, Y_{ij} = 0 \text{ if } (i,j) \in E\}\,,$$

the Sandwich theorem shows also how, for perfect graphs (for which $\omega(G) = \chi(G)$ holds) the maximum clique number can be computed in polynomial time.

Bounding $\omega(G)$ from below is harder; a simple bound derives from application of (10) to e, which gives $\frac{e^T}{n}A_G\frac{e}{n} = \frac{2m}{n^2} \leq 1 - \frac{1}{\omega(G)}$ and thus

$$\omega(G) \geq \frac{1}{1-\delta} > 1. \tag{21}$$

Wilf [318] obtained a sharper bound considering Perron eigenvector x_P properly normalized with $s := e^T x^P$, yielding via (10) $\frac{(x^P)^T}{s}A_G\frac{x^P}{s} = \frac{\lambda_P}{s^2} \leq 1 - \frac{1}{\omega(G)}$ and so

$$\omega(G) \geq \frac{\lambda_P}{s^2 - \lambda_P} + 1 \geq \frac{\lambda_P}{n - \lambda_P} + 1 \tag{22}$$

with equality holding if and only if the graph is complete; the rightmost inequality is easily derived from $x_P^T x_P = 1$ and the Cauchy/Schwarz inequality which gives $s^2 = (e^T x_P)^2 \leq (e^T e)(x_P^T x_P) = n$.

If one knows the full set of eigenvalues and eigenvectors of A_G this bound can be improved strictly. In fact with the full set of eigenvectors one can always build $x^* \in \Delta$ such that $g_* := (x^*)^T A_G x^* > \frac{\lambda_P}{s^2}$ (see [75] for details) and (10) immediately gives

$$\omega(G) \geq \frac{1}{1-g_*}. \tag{23}$$

Bound (21) is easy to compute while (22), requiring Perron eigenvector and eigenvalue, is a little harder. To find g_* of (23) one needs the full set of eigenvalues and eigenvectors of A_G and, consequently, this bound is the hardest to calculate but also, provably, the sharpest one. A final, obvious, remark, is that, being $\omega(G)$ integer, any non integer bound can be sharpened applying the appropriate ceiling $\lceil\rceil$ or floor $\lfloor\rfloor$ operators.

There exist fortunate cases in which these bounds almost solve the maximum clique problem, like, for example, the 64-node graph 'hamming6-2' of the DIMACS challenge [189]. Application of (18) and (23) allows to state that for this graph $32 \leq \omega(G) \leq 33$..

Considering just a subset of all possible graphs one can exploit the characteristics of the given subset to obtain sharper bounds. A much studied case is that of randomly generated graphs. For these graphs there exists a well established theory [53] and research proceeds in several directions; for example see [55], [122], [193], [229] and [236] for eigenvalues of random graphs. In the specific field of maximum cliques a well known result, due to Matula, accurately predicts the size of the maximum clique when the number of vertices n is sufficiently large [230]. In particular Matula was able to prove that the probability that

$$\lfloor M(n,\delta)\rfloor \leq \omega(G) \leq \lceil M(n,\delta)\rceil \tag{24}$$

tends to 1 when $n \to \infty$ and where

$$M(n,\delta) = 2\log_{1/\delta} n - 2\log_{1/\delta}\log_{1/\delta} n + 2\log_{1/\delta}\frac{e}{2} + 1\ . \tag{25}$$

δ being the density of the random graphs under consideration. There is also another result [55] about the smallest maximal clique and it shows that its size is almost surely $M(n,\delta)/2$ in the limit of large n.

Another interesting result for these graphs is known as the Jerrum conjecture [187] (see also [10] for recent, related results). It states that in large random graphs of density $\delta = 1/2$ there is no polynomial time algorithm that, with probability greater than 1/2, can find a clique larger than the smallest maximal clique.

5 Exact Algorithms

5.1 Enumerative Algorithms

The first algorithm for enumerating all cliques of an arbitrary graph in the literature is probably due to Harary and Ross [158]. In 1957, they proposed an inductive method that first identified all the cliques of a special graph with no more than three cliques. Then the problem on general graphs is reduced to this special case. Their work was stimulated by the matrix manipulation problem of sociometric data to find a complete identification of cliques.

Early works following that of Harary and Ross can be found in [222, 268, 65, 226, 38, 108]. What Paull and Unger [268], and Marcus [226] proposed were algorithms to minimize the number of rows in a flow table for a sequential switching function. Bonner addresses in [65] the clustering problem in information systems. Bednarek and Taulbee [38] proposed algorithms for generating all maximal chains of a set with a binary relation defined on it. Although these problems come from different fields and apparently deal with different problems, they are solving the same problem of enumerating all cliques of a graph. With the technology at that time, these early algorithms could only be tested on special graphs.

In 1970, Auguston and Minker [15] investigated several graph theoretic clustering techniques used in information systems. In their work, the algorithm of Bierstone and that of Bonner were tested. The method used in both algorithms was called the *vertex sequence method* or *point removal method*. This method produces cliques of G from the cliques of $G \setminus \{v\}$ with $v \in V$. From their computational results, they found the algorithm of Bierstone was more efficient. The original work of Bierstone was not published. The version of Bierstone's algorithm contained in Auguston and Minker [15] had two errors that were corrected by Mulligan and Corneil [239] in 1972.

Then in 1973, two new algorithms using the *backtracking method* were proposed by Akkoyunlu [8], and by Bron and Kerbosch [73]. The advantage of the backtracking method is the elimination of the redundancy in generating the same clique. What was more important for these two algorithms was their polynomial storage requirements. For example, the Bron and Kerbosch algorithm requires at most $\frac{1}{2}n(n+3)$ storage space. Bron and Kerbosch tested their algorithm on graphs of 10 to 50 vertices and densities ranging from 10% to 95%. Here the density was defined as the probability of a pair of vertices being connected. They found their algorithm was much more efficient than Bierstone's algorithm. One very interesting phenomenon from their test was the ratio of CPU time over the number of cliques of the graph, as they put it, "hardly dependent on the size of the graph". Bron and Kerbosch's algorithm is *Algorithm 457* in the ACM collection.

More enumerative algorithms were proposed in the 70's following that of Bron and Kerbosch, among them [250, 249, 231, 191, 192, 210, 310, 132]. The algorithm of Osteen and Tou [250] was an improved version of the point removal method. Osteen's [249] algorithm was designed for a special class of graphs. The algorithm of Meeusen and Cuyvers [231] started with decomposing a graph into subgraphs satisfying *the chain of subsets in G* requirement. Such a decomposition had the property that every clique is contained completely in at least one subgraph. Based on this property, they proposed an algorithm to find all cliques of a graph. The work of Johnston [192] contains a family of algorithms that are variations of Bron and Kerbosch's algorithm. By comparing several algorithms computationally, Johnston [192] concluded that the Bron and Kerbosch algorithm was one of the most efficient algorithms.

Tsukiyama et al. [310] proposed an enumerative algorithm that combined the approaches used in [15, 73], and by Akkoyunlu [8]. The result was an algorithm with time complexity of $O(nm\mu)$ and storage requirement of $O(n+m)$, where n, m, μ are the number of vertices, edges and maximal cliques of a graph. As pointed out in [310], this bound is stronger than the earlier bound of $O(\mu^2)$ from [15]. The algorithm of Gerhards and Lindenberg [132] started with partial cliques related to fixed vertices of G. Then, cliques were generated from these partial cliques. Their computational results suggested their proposed algorithm was as efficient as that in [73] for general graphs, but more efficient on sparse graphs.

In 1980's, other proposed algorithms include those in [215, 214, 87, 309, 190].

Loukakis and Tsouros [215] proposed a depth-first enumerative algorithm that generated all maximal independent sets lexicographically. They compared their algorithm with the algorithms of [73] and [310]. Their computational results on graphs of up to 220 vertices suggested the superior efficiency of their algorithm: which was two to fifteen times faster than that in [73], and three times faster than that in [310]. Two years later, Loukakis [214] claimed an additional improvement of a factor three speed-up compared to [215], tested on graphs of 30 to 220 vertices with densities from 10% (for small graphs) to 90% (for large graphs).

In 1988, Johnson et al. [190] proposed an algorithm that enumerated all maximal independent sets in lexicographic order. The algorithm has an $O(n^3)$ delay between the generation of two subsequent independent sets; cf. the complexity result Theorem 3.1. Chiba and Nishizeki's [87] algorithm lists all cliques with time complexity of $O(a(G)m\mu)$, where $a(G)$ is the *arboricity* of graph G. This is an improvement over the time complexity in [310].

Finally, Tomita et al. [309] proposed a modified Bron and Kerbosch [73] algorithm and claimed its time complexity to be $O(3^{n/3})$. As they pointed out, this was the best one could hope for since the Moon and Moser graphs [237] have $3^{n/3}$ maximal cliques.

5.2 Exact Algorithms for the Unweighted Case

If our goal is to find a maximum clique or just the size of a maximum clique, a lot of work can be saved from the above enumerative algorithms. Because once we find a clique, we only need to enumerate cliques better than the current best clique. Modifying the enumerative algorithms based on this argument results in various implicit enumerative algorithms. This argument can also be used in designing implicit enumerative algorithms.

The most well known and commonly used implicit enumerative method for the maximum clique problem is the *branch and bound* method. Background information on how branch and bound method works can be found in, for example, [28] and [246]. The key issues in a branch and bound algorithm for the maximum clique problem are:

1. How to find a good lower bound, i.e. a clique of large size?
2. How to find a good upper bound on the size of maximum clique?
3. How to branch, i.e., break a problem into smaller subproblems?

Implicit enumerative algorithms for the maximum clique/independent set problem started in the 1970's by Desler and Hakimi [105], Tarjan [304], and Houck [177]. These early works were improved in 1977 by [305] and [178]. Tarjan and Trojanowski proposed in [305] a recursive algorithm for the maximum independent set problem. They show their algorithm has a time complexity of $O(2^{n/3})$. This time bound illustrates that it is possible to solve a NP-complete problem much faster than the simple, enumerative approach. In the same year, Chvátal used a certain type of *recursive proofs* in [93] to show the upper bound on the stability number "has length at least $O(c^n)$", where $c > 1$ is a constant. The work of Houck and Vemuganti [178] exploited the relationship between the maximum independent set and a special class of bipartite graphs. They used this relationship to find an initial solution in their algorithm for the maximum independent set problem.

Most algorithms in the literature for the maximum clique/independent set problem were proposed in the 1980's. For example, in 1982, Loukakis and Tsouros [216] proposed a *tree search* algorithm that finds the size of a maximum independent set. Then in 1984, Ebenegger et al. [110] proposed another algorithm for finding the stability number of a graph. Their approach is based on the relationship between the maximization of a pseudo-Boolean function and the stability number of a graph. Computational tests on graphs with up to 100 vertices were reported in [110].

One of the most important contributions in the 1980's on practical algorithms for the maximum clique problem is due to Balas and Yu [31]. In their algorithm, the implicit enumeration was implemented in a new way. The idea of their approach is as follows. First, find a maximal induced triangulated subgraph D of G. Once D is found, find a maximum clique of D. This clique provides a lower bound and a feasible solution to the maximum clique problem. Then, they used a heuristic coloring procedure to extend D to a larger (maximal) subgraph that had no clique better than the current best clique. The importance of this second step is that it helps to reduce the number of subproblems generated from each node of the search tree, which in turn, reduces the size of the whole search tree. They solved the maximum clique problem on graphs of up to 400 vertices and 30,000 edges. Comparing their algorithm with other such algorithms, they found their algorithm not only generated a smaller search tree, but also required less CPU time.

In 1986, Kikusts [199] proposed a branch and bound algorithm for the maximum independent set problem based on a new recursive relation for the stability number of a graph G. Namely,

$$\alpha(G) = \max\{1 + \alpha(G \setminus [\{v\} \cup N(v)]),\ \alpha'_v\}, \tag{26}$$

where $\alpha'_v = \max\{|I| : \ I \subseteq G \setminus \{v\}$ is independent with $|I \cap N(v)| \geq 2\}$. This relation is different from the recursive relation

$$\alpha(G) = \max\{1 + \alpha(G \setminus [\{v\} \cup N(v)]),\ \alpha(G \setminus \{v\})\}, \tag{27}$$

traditionally used in designing branch and bound algorithms for the maximum independent set problem. Intuitively, relation (26) is stronger than relation (27). However, the trade off is a more complicated situation in (26). Some computational results were provided in [199] without comparison to other algorithms.

Also in 1986, Robson [286] proposed a modified recursive algorithm of Tarjan and Trojanowski [305]. Robson showed through a detailed case analysis that his algorithm had a time complexity of $O(2^{0.276n})$. This is an improvement over the time complexity $O(2^{n/3})$ of [305]. Here we want to mention the complexity proof of a similar recursive algorithm by Wilf [317]. Although Wilf's complexity of $O(1.39^n)$ is not as tight as that of [305], his proof is much simpler. Also in [317], Wilf proved (under certain probabilistic assumptions) that the average number of independent sets in a graph with n vertices is given by:

$$I_n = \sum_{k=0}^{n} \binom{n}{k} 2^{-k(k-1)/2}. \tag{28}$$

Using (28), it can be shown that the average complexity of a backtracking algorithm for the maximum independent set problem is subexponential, because I_n grows at the rate of $O(n^{\log n})$.

In late 1980's, new algorithms were proposed in [307, 130, 131] and in [265] (published in 1992). The algorithm of Tomita et al. [307] uses a greedy coloring algorithm to get an upper bound on the size of the maximum clique. Some computational results can be found in [131] and [307]. Gendreau et al. [130, 131] use an implicit enumerative algorithm. In [131], the branching rule (the selection of the next vertex to branch) is based on the number of triangles a vertex belongs to. The algorithm of Pardalos and Rodgers [265] is based on an unconstrained quadratic zero-one programming formulation of the maximum clique problem. In their work, the merit of two different branching rules, *greedy* and *nongreedy*, are tested.

In the 1990's, more algorithms were proposed, for example in [261, 121, 80, 21, 19, 321, 98].

Pardalos and Phillips [261] formulate the maximum clique problem as an indefinite quadratic global optimization problem with linear constrains. The algorithm of Friden et al. [121] is a branch and bound algorithm for the maximum independent set problem, employing tabu search techniques in finding lower and upper bounds. Carraghan and Pardalos [80] propose an implicit enumerative algorithm which is very efficient for sparse graphs (see also [262]). Their branching rule corresponds to the *nongreedy* rule described in [265]. Using this algorithm, they are able to solve problems on graphs of 500 vertices. Some test instances of graphs with 1000 and 2000 vertices are also examined. Since the algorithm is transparent and the code is publicly available, it can serve as a benchmark for comparing different algorithms.

Babel and Tinhofer propose in [21] a branch and bound algorithm for the maximum clique problem. The main ingredient of their algorithm is the use of a fast and relatively good heuristic for the minimum coloring problem proposed by Brelaz [71]. The coloring heuristic is called the *degree of saturation largest first* (DSATUR). Applying DSATUR to a graph, one can find an upper bound on the size of the maximum clique as well as a maximal clique (thus, a lower bound). Babel and Tinhofer exploit this distinct feature and apply DSATUR at each node of the search tree. They tested their algorithm on graphs of 100 to 400 vertices with varying densities. In [19], Babel further refines and improves the algorithm of [21].

Also in 1991, based on the fact that a fractional coloring solution provides a tighter upper bound than an integer coloring solution for the maximum clique problem, a heuristic for the fractional coloring problem is proposed by Xue in [321] and used in a branch and bound algorithm for the maximum clique problem. Substantial reduction in the search tree size and the improvement in efficiency of the branch and bound algorithm are observed because of the use of this new bounding procedure. Details about the method and how to extended it to the weighted case can also be found in [30].

Della Croce and Tadei reformulate in [98] the maximum clique problem into a multivariate binary knapsack problem (cf. also [252]) and combine a greedy algorithm with branch and bound methods. In [69], Bourjolly et al. propose a column-generation method embedded in a branch and bound algorithm, to obtain competitive lower bounds for the maximum clique problem, and also to obtain valid cuts for the linear relaxation of the minimal vertex cover problem. Bourjolly and coworkers presented in [68] a new approach for minimizing general quadratic 0-1 functions related to the satisfiability of a sequence of Boolean expressions. They developed lower bounding procedures for minimizing such functions and interpreted them in the context of the maximum independent set problem. These bounding rules were then incorporated in a standard branch and bound algorithm which was tested on various DIMACS benchmark graphs (cf. Section 6).

The continuous formulation in Theorem 2.6 allows to convert any finite exact procedure for solving indefinite quadratic programming problems globally, e.g. that in [60], into an exact algorithm. More adapted to our problem are solvers specifically dedicated to global optimization of quadratic forms over a simplex, which problems also recently were introduced under the name *standard quadratic problems* [57, 58]. The algorithms proposed there typically find a local solution for this sort of problems, and then either generate a certificate for global optimality, or produce an improving feasible point from which a new (monotonic) local search can be started. Both this escape step and the optimality certificate rely on the *copositivity* property of symmetric matrices which generalizes semi-definiteness: a $(n \times n)$ matrix Q is said to be copositive, if it generates a quadratic form taking no negative values on the positive orthant, or, equivalently, if

$$x^T Q x \geq 0 \quad \forall x \in \Delta \,.$$

In [57] it is shown that a maximal clique S (corresponding to a local maximizer x^S of $x^T \hat{A}_G x$ over Δ) is a maximum clique if and only if the matrix $Q_S = (2|S|-1)ee^T - 2|S|\hat{A}_G$ is copositive (observe that Q_S only depends on $|S|$). If, however, $x^T Q_S x < 0$ for some $x \in \Delta$, then $x^T \hat{A}_G x > (x^S)^T \hat{A}_G (x^S)$ so that an escape direction is found. Now while checking copositivity is again NP-hard, the full arsenal of finite procedures for this goal can be employed, see [57] and the references therein. In [57] also some experiments are conducted which show that even suboptimal local solutions serve well as heuristics in hard cases (i.e. large and dense graphs) where the (in principle finite) algorithm must be stopped prematurely.

A similar, very recent approach called *Genetic Engineering via Negative Fitness (GENF)* [64] also focuses on copositivity analysis, which is attacked by a block pivoting argument. Despite of the similarity in nomenclature, GENF is *not* a genetic algorithm (cf. Section 6.3.3), but rather a deterministic, finite and exact recursive procedure which produces simultaneously large cliques and large stable sets for the same graph (large stable sets are needed to obtain good pivots for efficient dimensional reduction).

5.3 Exact Algorithms for the Weighted Case

Algorithms for finding a maximum weight independent set of an arbitrary graph started in 1975 by Nemhauser and Trotter [245]. They considered the polyhedron relationships between the edge formulation (1) of the maximum weight independent set problem and its linear relaxation problem. Their main results were given as theorem 2.1 in section 2.1 of the present paper. Based on this result, they proposed an algorithm for the maximum weight independent set problem.

In 1977, Balas and Samuelsson [27] proposed an algorithm that solved the minimum vertex covering problem (a weighted version was also announced in [27]). Their algorithm was based on the relationship between the integer dual feasible solution and an equivalent linear programming for the vertex covering problem. Labeling procedures were designed to generate and improve vertex covers. When these labeling procedures could not continue, branch and bound was used. The computational tests in [245] and [27] were conducted on unweighted graphs of up to 50 vertices.

In 1983, Loukakis and Tsouros [217] proposed an algorithm for the maximum weight independent set problem. It seems that almost nothing else appeared in the literature until late 1980's and early 1990's. Recently published algorithms we are aware of for the maximum weight clique/independent set problem are due to Pardalos and Desai [260], Balas and Xue [29], and Nemhauser and Sigismondi [243].

The algorithm proposed by Pardalos and Desai [260] was based on an unconstrained quadratic 0-1 formulation of the maximum weight independent set problem. Their algorithm (for maximum weight independent set) used the *nongreedy* search strategy described in [265]. With this algorithm they were able to solve problems of up to 500 vertices with different densities. In an unpublished paper [81], Carraghan and Pardalos test a weighted version of their algorithm from [80], and it turns out that this is more efficient than that of Pardalos and Desai in [260].

Balas and Xue [29] extend the algorithm of Balas and Yu [31] to the weighted case. To accomplish this, a minimum weighted coloring of a triangulated graph is needed. Although the minimum weighted coloring problem on triangulated graphs is known to be in class P (see [145]), there was no algorithm in the literature that had reasonable time complexity. In [29] a combinatorial algorithm for this problem with a time complexity of $O(n^2)$ is proposed and used to extended the algorithm of [31] to the weighted case. Computational results (graphs of size up to 2000 vertices are solved on a workstation) show that the size of the search tree is greatly reduced and the CPU time is much smaller than other such algorithms, especially for large, dense graphs.

In [30], Balas and Xue propose a fast heuristic for the weighted fractional coloring problem and used this heuristic as an upper bounding procedure in a branch and bound algorithm for the maximum weight clique problem. Comparing with the method in [29], computational results show the reduction in search tree size and the improved efficiency of the resulting algorithm.

The algorithm of Nemhauser and Sigismondi [243] uses the polyhedron approach. They attack the problem by first solving the linear relaxation of the corresponding integer programming problem. If the optimal solution to the relaxation problem is integer, we are done. Otherwise, sets of valid inequalities are generated and added into the relaxed problem to cut off the current fractional solution. Attention is focused to some classes of facet defining inequalities for the maximum independent set problem. Since not all facets for the clique/independent set polytope are known, there is no guarantee that a fractional solution would always be cut off. When this happened, Nemhauser and Sigismondi switch in [243] to a branch and bound method. From their computational test, they found too many iterations in solving the linear relaxation problems. The largest graphs they tried to solve had up to 120 vertices. Similar approaches, partly only for the unweighted case, can be found in [244, 18, 24].

In an algorithm published in 1994, Babel [20] uses a branch and bound approach as follows: upper and lower bounds for the maximum weight of cliques are found by coloring the weighted graph, where the number of colors represents the total sum of all weights. So to process a graph of order 500 with weights out of $\{1, \ldots, 10\}$ we may have to deal with up to 5000 colors. The branching part of Babel's algorithm divides the bounded search-tree into smaller subproblems, the branching decisions depending on a specific order of all possible remaining nodes.

Similarly to the unweighted case, the continuous formulation of Theorem 2.8 provides exact procedures for the maximum weight clique problem out of any finite (global) indefinite quadratic program solver. Now, as easily can be seen, for any clique S of G, and any matrix $C \in \mathcal{C}(G, w)$, we have $(x^S)^T C(x^S) = 1/[2W(S)]$. If S is a maximal clique, then x^S is a local minimizer of the quadratic form generated by C over Δ, and so S is a maximum weight clique if and only if the matrix $Q_{S,w} = 2W(S)C - ee^T$ is copositive. Observe that $Q_{S,w}$ depends on S and w only via $W(S) = \sum_{i \in S} w_i$. Specializing $C = C(e)$ we see that in the unweighted case $w = e$ we again get $Q_{S,e} = 2|S|(ee^T - \hat{A}_G) - ee^T = Q_S$, and regarding copositivity detection the arguments at the end of the preceding section apply as well. This approach is followed in [62], with some empirical evidence that (local) solutions obtained even in case of premature stopping of the algorithm (which in principle is finite) may again yield satisfying approximations. Also, the GENF approach [64] is applicable here though the maximizer yielding auxiliary pivots have not the same nice interpretation as in the unweighted case.

6 Heuristics

Because of the computational complexity of the maximum clique problem, which as seen in Section 3 is also hard to approximate, much effort has recently been directed towards devising efficient heuristics, for which no formal guarantee of performance may be provided, but which are anyway of interest in practical applications.

In a branch and bound algorithm for the maximum clique problem, its lower bounding procedure usually provides a maximal clique which can be used to approximate the maximum clique of G. Since different branch and bound algorithms tend to have different bounding procedures, they provide different heuristics for the maximum clique problem. On the other hand, there are many heuristics in the literature explicitly designed to find approximation solutions to the maximum clique problem. These heuristics are usually more complicated than the lower bounding procedures from a branch and bound algorithm. In this section we provide a description of these procedures.

Lacking (almost by definition) a general theory of how these algorithms work, their evaluation is essentially based on massive experimentation. In order to facilitate comparisons among different heuristics, a set of benchmark graphs arising from different applications and problems has recently been constructed in conjunction with the 1993 DIMACS challenge on cliques, coloring and satisfiability [189]. These include graphs arising from coding theory [159], artificially generated graphs with known clique size, graphs in which the expected clique number is much smaller than the actual one [72], etc. These data are available at the following WWW address:

`http://dimacs.rutgers.edu/Challenges/index.html`

along with additional useful material.

6.1 Sequential Greedy Heuristics

Many approximation algorithms in the literature for the maximum clique problem are called *sequential greedy heuristics*. These heuristics generate a maximal clique through the repeated addition of a vertex into a partial clique, or the repeated deletion of a vertex from a set that is not a clique. Kopf and Ruhe [205] named these two classes of heuristics the *Best in* and the *Worst out* heuristics. Decisions on which vertex to be *added in* or *moved out* next are based on certain indicators associated with candidate vertices. For example, a possible *Best in* heuristic constructs a maximal clique by repeatedly *adding in* a vertex that has the largest degree among candidate vertices. In this case, the indicator is the degree of a vertex. On the other hand, a possible *Worst out* heuristic can start with the whole vertex set V. It will repeatedly remove a vertex out of V until V becomes a clique.

Kopf and Ruhe [205] further divided the above two classes of heuristics into *New* and *Old* (*Best in* or *Worst out*) heuristics. Namely, if the indicators are updated every time a vertex is added in or moved out, then the heuristic is called a *New* heuristic. Otherwise it is called an *Old* heuristic. We can find in the literature that many heuristics for the maximum clique problem fall in one or the other classes. See for example, the approximation algorithm of Johnson [188], and the approximation algorithm of Tomita et al. [308]. The differences among these heuristics are their choice of indicators and how indicators are updated. A heuristic of this type can run very fast.

6.2 Local Search Heuristics

A common feature of the sequential heuristics just described is that they all find only one maximal clique. Once a maximal clique is found, the search stops. We can view this type of heuristics from a different point of view. Let us define $\mathcal{S}_G$ to be the system consisting of all the maximal cliques of G. What a sequential greedy heuristic does is to find one set in $\mathcal{S}_G$, hoping it is (close to) the optimal set. This suggests us a possible way to improve our approximation solutions, namely, expand the search in $\mathcal{S}_G$. For example, once we find a set $S \in \mathcal{S}_G$, we can search its neighbors to improve S. This leads to the class of the *local search heuristics* [2].

In a local search heuristic, if we search more neighbors of $S \in \mathcal{S}_G$, we increase the chance of finding a better solution. Depending on the neighborhood definition of a set $S \in \mathcal{S}_G$, and how the search is performed, different local search heuristics result. A well known class of local search heuristics in the literature is the k-*interchange* heuristics. They are based on the k-*neighbor* of a feasible solution. In the case of the maximum clique problem, a k-neighbor of $S \in \mathcal{S}_G$ is defined as follows. A set $C \in \mathcal{S}_G$ is a k-neighbor of S if $|C \triangle S| \leq k$, where $k \leq |S|$. A k-interchange heuristic first finds a maximal clique $S \in \mathcal{S}_G$. Then it searches all the k-neighbors of S and outputs the best clique found. As one will expect, the main factors for the complexity of this class of heuristics are the size of the neighborhood and the searches involved. For example, in the k-interchange heuristic, the complexity grows roughly with $O(n^k)$.

The solution quality of a local search heuristic directly depends on the starting set $S \in \mathcal{S}_G$ and the neighborhood of S. To improve the quality of its solution, we need to increase the neighborhood of S (the starting set) to include a "better" set. If we want to look at various sets spread over $\mathcal{S}_G$, we need to have a very large neighborhood. The problem is when the size of the neighborhood increases, the search effort increases so rapidly that one could not afford it.

A class of heuristics designed to search various sets of $\mathcal{S}_G$ is called the *randomized heuristics*. The main ingredient of this class of heuristics is the part that finds a random set in $\mathcal{S}_G$. A possible way to do that is to include some random factors in the generation of a set of $\mathcal{S}_G$. A randomized heuristic runs a heuristic (with random factors included) a number of times to find different sets over $\mathcal{S}_G$. For example, we can randomize a sequential greedy heuristic and let it run N times. The complexity of a randomized heuristic depends on the complexity of the heuristic and the number N.

An elaborated implementation of the randomized heuristic for the maximum independent set problem can be found in Feo et al. [115] where local search is combined with randomized heuristic. Their computational results indicated that their approach was effective in finding large cliques of randomly generated graphs. For example, for randomly generated graphs with 1000 vertices and 50% density, their approach found cliques of size 15 or larger in most cases. Here, 15 is a bound derived from the probabilistic analysis of this class of graphs [53, 55, 122]. A different implementation of a randomized algorithm for the maximum independent set problem can be found in [9].

6.3 Advanced Search Heuristics

Local search algorithms are only capable of finding *local* solutions of an optimization problem. In the past few years, many powerful variations of the basic local search procedure have been developed which try to avoid this problem, many of which are inspired from various phenomena occurring in nature. Examples of such algorithms are simulated annealing, neural networks, genetic algorithms and DNA computing. Because of its importance it is not surprising that these techniques have been applied to the maximum clique problem.

6.3.1 Simulated annealing

In condensed-matter physics, the term "annealing" refers to a physical process to obtain a pure lattice structure, where a solid is first heated up in a heat bath until it melts, and next cooled down slowly until it solidifies into a low-energy state. During the process, the free energy of the system is minimized. Simulated annealing, introduced in 1983 by Kirkpatrick, Gelatt and Vecchi [201], is a randomized neighborhood search algorithm based on the physical annealing process. Here, the solutions of a combinatorial optimization problem correspond to the states of the physical system, and the cost of a solution is equivalent to the energy of the state.

In its original formulation, simulated annealing works essentially as follows. Initially, a tentative solution in the state space is somehow generated. A new neighboring state is then produced from the previous one and, if the value of the cost function f improves, the new state is accepted, otherwise it is accepted with probability $\exp\{\Delta f/\tau\}$, where Δf is the difference of the cost function between the new and the current state, and τ is a parameter usually called the *temperature* in analogy with physical annealing, which is varied carefully during the optimization process. The algorithm proceeds iteratively this way until a stopping condition is met. One of the critical aspects of the algorithm relates to the choice of the proper "cooling schedule," i.e., how to decrease the temperature as the process evolves. While a logarithmically slow cooling schedule (yielding an exponential time algorithm) provably guarantees the exact solution, faster cooling schedules, producing acceptably good results, are in widespread use. Introductory textbooks describing both theoretical and practical issues of the algorithm are [208] and [1].

Aarts and Korst [1], without presenting any experimental result, suggested the use of simulated annealing for solving the independent set problem, using a *penalty function* approach. Here, the solution space is the set of all possible subsets of vertices of the graph G, and the problem is formulated as one of maximizing the cost function $f(V') = |V'| - \lambda|E'|$, where $|E'|$ is the number of edges in $G(V')$, and λ is a weighting factor exceeding 1.

Jerrum [187] conducted a theoretical analysis of the performance of a clique-finding *Metropolis* process, i.e., simulated annealing at fixed temperature, on random graphs. He proved that the expected time for the algorithm to find a clique that is only slightly bigger than that produced by a naive greedy heuristic, grows faster than any polynomial in the number of vertices. This suggests that "true" simulated annealing would be ineffective for the maximum clique problem.

Jerrum's conclusion seems to be contradicted by practical experience. In [173], Homer and Peinado compare the performance of three heuristics, namely the greedy heuristic developed by Johnson [188], a randomized version of Boppana and Halldórsson's subgraph-exclusion algorithm [66], and simulated annealing, over very large graphs. The simulated annealing algorithm was essentially that proposed by Aarts and Korst, with a simple cooling schedule. This penalty function approach was found to work better than the method in which only cliques are considered, as proposed by Jerrum [187]. The algorithms were tested on various random graphs as well as on DIMACS benchmark graphs. The authors ran the algorithms over an SGI workstation for graphs with up to 10,000 vertices, and on a Connection Machine for graphs with up to 70,000 vertices. The overall conclusion was that simulated annealing outperforms the other competing algorithms; it also ranked among the best heuristics for maximum clique presented at the 1993 DIMACS challenge.

6.3.2 Neural networks

Artificial neural networks (often simply referred to as "neural networks") are massively parallel, distributed systems inspired by the anatomy and physiology of the cerebral cortex, which exhibit a number of useful properties such as learning and adaptation, universal approximation, and pattern recognition (see [167], [162] for an introduction).

In the mid-1980's Hopfield and Tank [174] showed that certain feedback continuous neural models are capable of finding approximate solutions to difficult optimization problems such as the traveling salesman problem [174]. This application was motivated by the property that the temporal evolution of these models is governed by a quadratic Lyapunov function (typically called "energy function" because of its analogy with physical systems) which is iteratively minimized as the process evolves. Since then, a variety of combinatorial optimization problems have been tackled within this framework. The customary approach is to formulate the original problem as one of energy minimization, and then to use a proper relaxation network to find minimizers of this function. Almost invariably, the algorithms developed so far incorporate techniques borrowed from statistical mechanics, in particular mean field theory, which allow one to escape from poor local solutions. We mention the articles [213, 277] and the textbook of Takefuji [302] for surveys of this field, and a journal special issue [184] for recent advances. In [1], an excellent introduction to a particular class of neural networks (the Boltzmann machine) for combinatorial optimization is provided.

Early attempts at encoding the maximum clique and related problems in terms of a neural network were already done in the late 1980's by Ballard et al. [34], Godbeer et al. [139], Ramanujam and Sadayappan [284], Aarts and Korst [1], and Shrivastava et al. [294] (see also [295]). However, little or no experimental results were presented, thereby making it difficult to evaluate the merits of these algorithms. In [211], Lin and Lee used the quadratic zero-one formulation from [264] as the basis for their neural network heuristic. On random graphs with up to 300 vertices, they found their algorithm to be faster than the implicit enumerative algorithm in [80], while obtaining slightly worse results in terms of clique size.

Grossman [148] proposed a discrete, deterministic version of the Hopfield model for maximum clique, originally designed for an all-optical implementation. The model has a threshold parameter which determines the character of the stable states of the network. The author suggests an annealing strategy on this parameter, and an adaptive procedure to choose the network's initial state and threshold. On DIMACS graphs the algorithm performs satisfactorily but it does not compare well with more powerful heuristics such as simulated annealing.

Jagota [183] developed several variations of the Hopfield model, both discrete and continuous, to approximate maximum clique. He evaluated the performance of his algorithms over randomly generated graphs as well as on harder graphs obtained by generating cliques of varying size at random and taking their union. Experiments on graphs coming from the Solomonoff-Levin, or "universal" distribution are also presented in [185]. The best results were obtained using a stochastic steepest descent dynamics and a mean-field annealing algorithm, an efficient deterministic approximation of simulated annealing. These algorithms, however, were also the slowest, and this motivated Jagota et al. [186] to improve their running time. The mean-field annealing heuristic was implemented on a 32-processor Connection Machine, and a two-temperature annealing strategy was used. Additionally, a "reinforcement learning" strategy was developed for the stochastic steepest descent heuristic, to automatically adjust its internal parameters as the process evolves. On various benchmark graphs, all their algorithms obtained significantly larger cliques than other simpler heuristics but ran slightly slower. Compared to more sophisticated heuristics, they obtained significantly smaller cliques on average but were considerably faster.

Other attempts at solving the maximum clique problem using Hopfield-style neural networks can be found in [303], [124], [320], [47]. Pelillo [271] takes a completely different approach to the problem, by exploiting the Motzkin-Straus continuous formulation and the dynamical properties of the so-called relaxation labeling networks. His algorithm is the prototype of the replicator-equation based procedures described in Section 6.4.

6.3.3 Genetic algorithms

Genetic algorithms are parallel search procedures inspired from the mechanisms of evolution in natural systems [172, 141]. In contrast to more traditional optimization techniques, they work on a population of points, which in the genetic algorithm terminology, are called chromosomes or individuals. In the simplest and most popular implementation, chromosomes are simply long strings of bits. Each individual has an associated "fitness" value which determines its probability of survival in the next "generation:" the higher the fitness, the higher the probability of survival. The genetic algorithm starts out with an initial population of members generally chosen at random and, in its simplest version, makes use of three basic operators: reproduction, crossover and mutation. Reproduction usually consists of choosing the chromosomes to be copied in the next generation according to a probability proportional to their fitness. After reproduction, the crossover operator is applied between pairs of selected individuals to produce new offsprings. The operator consists of swapping two or more sub-segments of the the strings corresponding to the two chosen individuals. Finally, the mutation operator is applied, which randomly reverses the value of every bit within a chromosome with a fixed probability. The procedure just described is sometimes referred to as the "simple" genetic algorithm [141].

One of the earliest attempts to solve the maximum clique problem using genetic algorithms was done in 1993 by Carter and Park [82]. After showing the weakness of the simple genetic algorithm in finding large cliques, even on small random graphs, they introduced several modifications in an attempt to improve performance. However, despite their efforts they did not get satisfactory results, and their general conclusion was that genetic algorithms need to be heavily customized in order to be competitive with traditional approaches, and that they are computationally very expensive. In a later study [267], genetic algorithms were proven to be less effective than simulated annealing. At almost the same time Bäck and Khuri [22], working on the maximum independent set problem, arrived at the opposite conclusion. By using a straightforward, general-purpose genetic algorithm called GENEsYs and a suitable fitness function which included a graded penalty term to penalize infeasible solutions, they got interesting results over random and regular graphs with up to 200 vertices. These results indicate that the choice of the fitness function is crucial for genetic algorithms to provide satisfactory results.

Murthy et al. [241] also experimented with a genetic algorithm using a novel "partial copy crossover," and a modified mutation operator. However, they presented results over very small (i.e., up to 50 vertices) graphs, thereby making it difficult to properly evaluate the algorithm.

Bui and Eppley [77] obtained encouraging results by using a hybrid strategy which incorporates a local optimization step at each generation of the genetic algorithm, and a vertex-ordering preprocessing phase. They tested the algorithm over some DIMACS graphs getting results comparable to that in [134] (cf. Section 6.4).

Instead of using the standard binary representation for chromosomes, Foster and Soule [118] employed an integer-based encoding scheme. Moreover, they used a time weighting fitness function similar in spirit to those of Carter and Park [82]. The results obtained are interesting, but still not comparable to those obtained using more traditional search heuristics.

Fleurent and Ferland [117] developed a general-purpose system for solving graph coloring, maximum clique, and satisfiability problems. As far as the maximum clique problem is concerned, they conducted several experiments using a hybrid genetic search scheme which incorporates tabu search (described in Section 6.3.4) and other local search techniques as alternative mutation operators. The results presented are encouraging, but running time is quite high.

In [169], Hifi modifies the basic genetic algorithm in several aspects: (a) a particular crossover operator creates two new different children; (b) the mutation operator is replaced by a specific heuristic feasibility transition adapted to the weighted maximum stable set problem. This approach is also easily parallelizable. Experimental results on randomly generated graphs and also some (unweighted) instances from the DIMACS testbed [189] are reported to validate this approach.

Finally, Marchiori [225] has recently developed a simple heuristic-based genetic algorithm which consists of a combination of the simple genetic algorithm and a naive greedy heuristic procedure. Unlike previous approaches, here there is a neat division of labour, the search for a large subgraph and the search for a clique being incorporated into the fitness function and the heuristic procedure, respectively. The algorithm outperforms previous genetic-based clique finding procedures over various DIMACS graphs, both in terms of quality of solutions and speed.

We note that the GENF procedure proposed in [64] does not fall into the category of genetic algorithms, despite similarity in nomenclature. Both approaches share the modeling idea that the objective is interpreted as fitness and the decision variables could be interpreted as characteristics of a population subject to selection. The most important difference is that the (random) mutation prevailing in genetic algorithms has no counterpart in the GENF algorithm, where escaping from inefficient local solutions is done deliberately and deterministically (similar to real-world genetic engineering).

6.3.4 Tabu search

Tabu search, introduced independently by Glover [136], [137] and Hansen and Jaumard [156], is a modified local search algorithm, in which a prohibition-based strategy is employed to avoid cycles in the search trajectories and to explore new regions in the search space. At each step of the algorithm, the next solution visited is always chosen to be the best *legal* neighbor of the current state, even if its cost is worse than the current solution. The set of legal neighbors is restricted by one or more *tabu lists* which prevent the algorithm to go back to recently visited solutions. These lists are used to store historical information on the path followed by the search procedure. Sometimes the tabu restriction is relaxed, and tabu solutions are accepted if they satisfy some *aspiration level* condition. The standard example of a tabu list is one which contains the last k solutions examined, where k may be fixed or variable. Additional lists containing the last modifications performed, i.e., changes occurred when moving from one solution to the next, are also common. These types of lists are referred to as *short-term* memories; other forms of memories are also used to *intensify* the search in a promising region or to *diversify* the search to unexplored areas. Details on the algorithm and its variants can be found in [138] and [168].

In 1989, Friden et al. [120] proposed a heuristic for the maximum independent set problem based on tabu search. The size of the independent set to search for is fixed, and the algorithm tries to minimize the number of edges in the current subset of vertices. They used three tabu lists: one for storing the last visited solutions and the other two to contain the last introduced/deleted vertices. They showed that by using hashing for implementing the first list and choosing a small value for the dimensions of the other two lists, a best neighbor may be found in almost constant time. The tabu-search-based branch and bound algorithm presented by the same authors in [121] has already been mentioned in Section 5.

In [131, 298], three variants of tabu search for maximum clique are presented. Here the search space consists of complete subgraphs whose size has to be maximized. The first two versions are deterministic algorithms in which no sampling of the neighborhood is performed. The main difference between the two algorithms is that the first one uses just one tabu list (of the last solutions visited), while the second one uses an additional list (with an associated aspiration mechanism) containing the last vertices deleted. Also, two diversification strategies were implemented. The third algorithm is probabilistic in nature, and uses the same two tabu lists and aspiration mechanism as the second one. The differences lie in a random sampling of the neighborhood, and also in the possibility of multiple vertex deletion in the current solution. Here no diversification strategy was used. In [131, 298] results on randomly generated graphs were presented and the algorithms were shown to be very effective. More recently, Soriano and Gendreau [297] tested their algorithms over the DIMACS benchmark graphs and the results confirmed the early conclusions.

Recently, Battiti and Protasi [37] extended the tabu search framework by introducing a reactive local search method. They modified a previously introduced reactive scheme by exploiting the particular neighborhood structure of the maximum clique problem. In general, reactive schemes aim at avoiding the manual selection of control parameters by means of an internal feedback loop. Battiti and Protasi's algorithm adopts such a strategy to automatically determine the so-called prohibition parameter k, i.e., the size of the tabu list. Also an explicit memory-influenced restart procedure is activated periodically to introduce diversification. The search space consists of all possible cliques, as in Friden et al.'s approach, and the function to be maximized is the clique size. The worst case computational complexity of this algorithm is $O(\max\{n, m\})$ where n and m are the number of nodes and edges of the graph respectively. They noticed, however, that in practice, the number of operations tends to be proportional to the average degree of the nodes of the complement graph. They tested their algorithm over many DIMACS benchmark graphs obtaining better results than those presented at the DIMACS workshop in competitive time.

6.4 Continuous-based Heuristics

Recently, there has been much interest around the Motzkin-Straus and related continuous formulations of the maximum clique problem (see Section 2). They suggest in fact a fundamentally new way of solving the maximum clique problem, by allowing us to shift from the discrete to the continuous domain in an elegant manner. As recently pointed out [259], continuous formulations of discrete optimization problems turn out to be particularly attractive. They not only allow us to exploit the full arsenal of continuous optimization techniques, thereby leading to the development of new algorithms, but may also reveal unexpected theoretical properties.

In [261], Pardalos and Phillips developed a global optimization approach based on the Motzkin-Straus formulation and implemented an iterative clique retrieval process to find the vertices of the maximum clique. However, due to its high computational cost they were not able to run the algorithm over graphs with more than 75 vertices. More recently, Pelillo [271] used *relaxation labeling algorithms* to approximately determining the size of the maximum clique using the original Motzkin-Straus formulation. These are parallel, distributed algorithms developed and studied in computer vision and pattern recognition, which are also surprisingly related to *replicator equations*, a class of dynamical systems widely studied in evolutionary game theory and related fields [171]. The model operates in the simplex Δ and possesses a quadratic Lyapunov function which drives its dynamical behavior. It is these properties that naturally suggest using them as a local optimization algorithm for the Motzkin-Straus program. The model is especially suited for parallel implementation, and is attractive for its operational simplicity, since no parameters need to be determined. Extensive simulations over random graphs with up to 2000 vertices have demonstrated the effectiveness of the approach and showed that the algorithm outperforms previous neural network heuristics.

In order to avoid time-consuming iterative procedures to extract the vertices of the clique, Gibbons, Hearn and Pardalos [134] have proposed a heuristics which is based on a parametrized formulation of the Motzkin-Straus program. They consider the problem of minimizing the function

$$h(x) = \frac{1}{2}x^T A_{\overline{G}} x + \left(\sum_{i=1}^{n} x_i - 1 \right)^2$$

on the domain:

$$S(k) = \left\{ x \in \mathbb{R}^n \ : \ \sum_{i=1}^{n} x_i^2 \leq \frac{1}{k}, \text{ and } x_i \geq 0, \ i = 1, \ldots, n \right\}$$

where k is a fixed parameter. Let x^* be a global minimizer of h on $S(k)$, and let $V(k) = h(x^*)$. In [134] it is proved that $V(k) = 0$ if and only if there exists an independent set S of $\overline{G}$ with size $|S| \geq k$. Moreover, the vertices of $\overline{G}$ associated with the indices of the positive components of x^* form an independent set of size greater than or equal k.

These properties motivated the following procedure to find a maximum independent set of $\overline{G}$ or, equivalently, a maximum clique of G. Minimize the function h over $S(k)$, for different values of k between predetermined upper and lower bounds. If $V(k) = 0$ and $V(k+1) \neq 0$ for some k, then the maximum clique of G has size k, and its vertices are determined by the positive components of the solution. Since minimizing h on $S(k)$ is a difficult problem, Gibbons and coworkers developed a heuristic based on the observation that by removing the nonnegativity constraints, the problem is that of minimizing a quadratic form over a sphere, a problem which is solvable in polynomial-time. However, in so doing a heuristic procedure is needed to round the approximate solutions of this new problem to approximate solutions of the original one. Moreover, since the problem is solved approximately, we have to find the value of the spherical constraint $1/k$ which yields the largest independent set. A careful choice of k is therefore needed. The resulting algorithm was tested over various DIMACS benchmark graphs [189] and the results obtained confirmed the effectiveness of the approach.

In [61], replicator equations are used in conjunction to the spurious-free formulation given in Theorem 2.6 to find maximal cliques of G. Note that here the nodes comprising the clique are directly given by the positive components of the converged vectors, and no iterative procedure is needed to determine them, as in [261]. The results obtained over a set of DIMACS benchmark graphs [189] were encouraging, especially considering that replicator equations do not incorporate any mechanism to escape from local optimal solutions. This suggests that the basins of attraction of the global solution w.r.t. the quadratic functions g and $\hat{g}$ occurring in (9) and Theorem 2.6 are quite large; for a thorough empirical analysis see also [64]. One may wonder whether a subtle choice of initial conditions and/or a variant of the dynamics may significantly improve the results, but experiments in [63] indicate this is not the case. In [59] the properties of the following function are studied

$$\hat{g}_\alpha(x) = x^T A_G x + \alpha x^T x$$

and a heuristic is proposed which is based on the modification of the parameter α in the course of the optimization process.

Finally, Bomze et al. [62] have recently used replicator equations to find maximal weight cliques in weighted graphs, using Theorem 2.8.

6.5 Miscellaneous

Another type of heuristics that finds a maximal clique of G is called the *subgraph approach* (see [31]). It is based on the fact that a maximum clique C of a subgraph $G' \subseteq G$ is also a clique of G. The subgraph approach first finds a subgraph $G' \subseteq G$ such that the maximum clique of G' can be found in polynomial time. Then it finds a maximum clique of G' and use it as the approximation solution. The advantage of this approach is that in finding the maximum clique $C \subseteq G'$, one has (implicitly) searched many other cliques of G' ($\mathcal{S}_{G'} \subseteq \mathcal{S}_G$). Because of the special structure of G', this implicit search can be done efficiently. In Balas and Yu [31], G' is a maximal induced triangulated subgraph of G. Since many classes of graphs have polynomial algorithms for the maximum clique problem, the same idea also applies there. For example, the class of edge-maximal triangulated subgraphs was chosen in [23], [321], and [322]. Some of the greedy heuristics, randomized heuristics and subgraph approach heuristics are compared in [321] and [322] on randomly generated weighted and unweighted graphs.

Various new heuristics were presented at the 1993 DIMACS challenge devoted to clique, coloring and satisfiability [189]. In particular, Balas and Niehaus [26] proposed an algorithm which is based on the observation that finding the maximum clique in the union of two cliques can be done using bipartite matching techniques. Goldberg and Rivenbrugh [142] used restricted backtracking to provide a tradeoff between the size of the clique and the completeness of the search. Mannino and Sassano [224] proposed an edge projection technique to obtain a new upper bound heuristic for the maximum independent set problem. This procedure was used, in conjunction with Balas and Yu's branching rule [31], to develop an exact branch and bound algorithm which works well especially on sparse graphs.

Abbattista et al. [3] developed a new population-based optimization heuristic inspired by the natural behavior of human or animal scouts in exploring unknown regions, and applied it to maximum clique. The results obtained over a few DIMACS graphs are comparable with those obtained using continuous-based heuristics but are inferior to those obtained by reactive local search.

Recently, DNA computing [5] has also emerged as a potential technique for the maximum clique problem [251], [325]. The major advantage of DNA computing is its high parallelism, but at present the size of graphs this algorithm can handle is limited to a few tens.

Additional heuristics for the maximum clique/independent set and related problems on arbitrary or special class of graphs can be found in [89], [91], [94], [116]. Among others, further (partly randomized) parallel algorithms for the (weighted) maximum clique problem are proposed in [195], [220], [101], [143], [144], [81], [263], [6], and [99].

7 Selected Applications

In many applications, the underlying problem can be formulated as a maximum clique problem while in others a subproblem of the solution procedure consists of finding a maximum clique. This necessitates the development of fast exact and approximate algorithms for the problem.

The proliferation of massive data sets brings with it a series of special computational challenges. Many of these data sets can be modeled as very large multidigraphs with a special set of edge attributes that represent special characteristics of the application at hand. Understanding the structure of the underlying digraph is essential for storage organization and information retrieval. In [4] experiments with data from telecommunications traffic, the corresponding multigraph has 53,767,087 vertices and over 170 million of edges. A giant connected component with 44,989,297 vertices was computed. The maximum clique problem is considered in this giant component. Similar computational challenges with very large graphs appear in several other practical applications.

The application areas considered below are diverse. For example, we will present a class of graphs from which we can prove or disprove Keller's conjecture; a famous problem in geometry, a part of which is still open. Another example arises from coding theory where one wishes to find binary codes as large as possible that can correct a prespecified number of errors. The problem can be solved by solving the maximum clique problem in a corresponding graph. We also indicate how the maximum clique problem occurs in fault diagnosis models. A further important application area of the maximum clique problem discussed here is computer vision and pattern recognition.

Other applications can be found, e.g., in [17], [269], [104], [270], [103], [312], [319], [314], [35], [266], [315], [223].

7.1 Coding Theory: Hamming and Johnson Graphs

In this section we will describe how coding theory problems can be interpreted as maximum clique problems on certain graphs. In Coding Theory, one wishes to find a binary code as large as possible that can correct a certain number of errors for a given size of the binary words (vectors), see [74, 296]. In order to correct errors, the code must consist of binary words among which any two differ in a certain number of positions so that a misspelled word can be detected and corrected. A misspelled word is corrected by replacing it with the word from the code that differs the least from the misspelled one.

The *Hamming distance* between the binary vectors $u = (u_1, u_2, \ldots, u_n)$ and $v = (v_1, v_2, \ldots, v_n)$ is the number of indices i such that $1 \leq i \leq n$ and $u_i \neq v_i$. We denote the Hamming distance by $dist(u, v)$.

It is well known that a binary code consisting of a set of binary vectors any two of which have Hamming distance greater or equal to d can correct $\lfloor \frac{d-1}{2} \rfloor$ errors [221]. Thus, what a coding theorist would like to find is the maximum number of binary vectors of size n with Hamming distance d. We denote this number by $A(n, d)$.

Another problem arising from Coding Theory, closely related to the one mentioned above, is to find a weighted binary code, that is, to find the maximum number of binary vectors of size n that have precisely w 1's and the Hamming distance of any two of these vectors is d. This number is denoted by $A(n, w, d)$. A binary code consisting of vectors of size n, weight w and distance d, can correct $w - \frac{d}{2}$ errors [221].

Now the *Hamming graph* $H(n, d)$, of size n and distance d, is defined as the graph with vertex set the binary vectors of size n, in which two vertices are adjacent if their Hamming distance is *at least* d. Then, $A(n, d)$ is the size of a maximum clique in $H(n, d)$.

The graph $H(n, d)$ has 2^n vertices, $2^{n-1} \sum_{i=d}^{n} \binom{n}{i}$ edges and the degree of each vertex is $\sum_{i=d}^{n} \binom{n}{i}$.

Next we define the Johnson graph, $J(n, w, d)$, with parameters n, w and d, as the graph with vertex set the binary vectors of size n and weight w, where two vertices are adjacent if their Hamming distance is *at least* d. Then, similar to Hamming graph, the size of the weighted code, $A(n, w, d)$, equals the size of the maximum clique in $J(n, w, d)$.

The graph $J(n, w, d)$ has $\binom{n}{w}$ vertices, $\frac{1}{2}\binom{n}{w} \sum_{k=\lceil \frac{d}{2} \rceil}^{w} \binom{w}{k}\binom{n-w}{k}$ edges and the degree of each vertex is $\sum_{k=\lceil \frac{d}{2} \rceil}^{w} \binom{w}{k}\binom{n-w}{k}$.

7.2 Geometry of Tiling: Keller's Conjecture

A family of hypercubes with disjoint interiors whose union is the Euclidean space $\mathbb{R}^n$ is a *tiling*. A lattice tiling is a tiling for which the centers of the cubes form a lattice [300].

In the beginning of the century, Minkowski conjectured that in a lattice tiling of $\mathbb{R}^n$ by translates of a unit hypercube, there exist two cubes that share a $(n-1)$-dimensional face. About fifty years later, Hajós [154] proved Minkowski's conjecture.

At 1930, Keller [198] suggested that Minkowski's conjecture holds even in the absence of the lattice assumption. Ten years later Perron [276] proved the correctness of Keller's conjecture for $n \leq 6$. Since then, many papers have been devoted to prove or disprove this conjecture [301] and recently, Lagarias and Shor [207] proved that Keller's conjecture fails for $n \geq 10$. Thus, it is left to prove whether the conjecture holds for $n = 7, 8, 9$.

We define the Keller Graph Γ_n as a graph with vertex set

$$V_n = \{(d_1, d_2, \ldots, d_n) : d_i \in \{0, 1, 2, 3\},\ i = 1, 2, \ldots, n\}$$

where two vertices $u = (d_1, d_2, \ldots, d_n)$ and $v = (d'_1, d'_2, \ldots, d'_n)$ in V_n are adjacent if and only if

$$\exists\, i,\ 1 \leq i \leq n : d_i - d'_i \equiv 2 \bmod 4 \tag{29}$$

and

$$\exists\, j \neq i,\ 1 \leq j \leq n : d_j \neq d'_j. \tag{30}$$

In [96], Corrádi and Szabó presented a graph theoretic equivalent of Keller's conjecture. It is shown that, there is a counterexample to Keller's conjecture if and only if there exist a $n \in \mathbf{N}^+$ such that Γ_n has a clique of size 2^n.

Γ_n has 4^n vertices, $\frac{1}{2}4^n(4^n - 3^n - n)$ edges and the degree of each node is $4^n - 3^n - n$. Γ_n is very dense and has at least $8^n n!$ different maximum cliques. It can be shown [207] that the maximum clique size of Γ_n is less than or equal to 2^n.

7.3 Problems Arising From Fault Diagnosis

A crucial problem in studying the reliability of large multiprocessor systems, is the problem known as system-level fault diagnosis. The task is to identify all faulty processors (units) in the system. The classical approach to fault diagnosis was originated over thirty years ago by Preparata, Metze and Chien [280], leading to a fault diagnosis model known as the PMC model.

In the PMC model each unit can test some other units and it is assumed that fault-free units always give the correct results while faulty ones are unpredictable and can output any results. Furthermore, it is assumed that the number of faulty units never exceed some upper bound t. Upon completion of all tests, the results are gathered by a monitoring unit which computes the status of all units based on the gathered results.

The assumption that a fault-free unit always detects faulty units may seem a little optimistic. Also, the upper bound assumption may restrict the model to unrealistic situations. Further, the PMC model is accurate only if the upper bound t does not exceed the number of neighbors of any unit. For large systems however, the connectivity might be fairly low, making it quite probable that the number of faulty units exceed the number of neighbors for some units.

Yet another assumption in the PMC model, the existence of a central monitoring unit, makes it less reliable. In order to overcome this problem, distributed fault-tolerance was introduced. The goal of this approach is to find a way to let every fault-free unit to be able to determine the status of every other unit.

The above observations have led to several different models one of which was introduced by Blough [50]. In his model, processors test each other and fault-free units always detect other fault-free processors correctly, while they detect faulty processors with a fixed probability less than 1. No assumption is made about how faulty units behave as testers.

In [44], Berman and Pelc study a realistic approach to fault diagnosis by simultaneously relaxing all the three assumptions from the PMC model described above. Their model is based on a probabilistic model presented by Blough performed in a distributed fashion. Consequently, a processor can never be sure that the information it receives is correct. Berman and Pelc define a system design represented by a class of graphs, G_n. They show that the probability of correct diagnosis of fault processors for such systems, happens with probability at least $1 - n^{-1}$. The algorithm they propose is based on a model where a test by a fault-free unit on a faulty one does not detect a fault with probability q, while they assume that fault-free units never detect faults in each other.

For a given parameter c, a *c-fat ring* is the graph $G = (V, E)$ defined as follows. Let

$$k = \lfloor \frac{|V|}{c \log |V|} \rfloor$$

and let $W_0, \ldots, W_{k-1}$ be a partition of V such that

$$c \log |V| \leq |W_i| \leq 1 + \lceil c \log |V| \rceil \text{ for } i = 0, 1, \ldots, k-1. \qquad (31)$$

For $u \in W_i$ and $v \in W_j$ we have $(u, v) \in E$ if and only if $u \neq v$ and $|i - j| \in \{0, 1, k-1\}$.

A major step in the algorithm proposed in [44] is to find the maximum clique of a c-fat ring. Therefore Hasselberg et al. [159] construct a c-fat ring generator and perform some computations to see how the maximum clique algorithms perform on such graphs.

7.4 Computer Vision and Pattern Recognition

Many fundamental problems in computer vision and pattern recognition can be formulated as the problem of matching *relational structures* [33]. In the computer vision terminology, a relational structure is a triple $S = (U, \mathcal{P}, \mathcal{R})$, where U is a set of units, $\mathcal{P} = \{P_1, \ldots, P_l\}$ is a set of properties, and $\mathcal{R} = \{R_1, \ldots, R_k\}$ is a set of (binary) relations over the units. Generalizations involving higher-order relations are also common, but for the sake of simplicity we shall only consider the binary case here. Note that the notion of a relational structure is essentially equivalent to that of a *pseudograph* employed in graph theory [157]. A relational structure becomes a graph, in the traditional sense, when the relation set $\mathcal{R}$ contains a single relation, and the property set $\mathcal{P}$ is empty. The relation may be symmetric, in which case we obtain an undirected graph.

Consider two relational structures $S' = (U', \mathcal{P}', \mathcal{R}')$ and $S'' = (U'', \mathcal{P}'', \mathcal{R}'')$. A pair of units (u', u''), one from S' and the other from S'', is said to be *good* if all properties that hold for u' hold for u'' as well, and vice versa, that is if

$$P_i'(u') \Leftrightarrow P_i''(u'')$$

for all $i = 1, \ldots, l$, where $P_i' \in \mathcal{P}'$ and $P_i'' \in \mathcal{P}''$. Similarly, two good pairs (u', u'') and (v', v''), with $u' \neq v'$ and $u'' \neq v''$, are said to be *compatible* if

$$R_j'(u', v') \Leftrightarrow R_j''(u'', v'') \text{ and } R_j'(v', u') \Leftrightarrow R_j''(v'', u'')$$

for all $j = 1, \ldots, k$, where $R_j' \in \mathcal{R}'$ and $R_j'' \in \mathcal{R}''$. A *match* between S' and S'' is any relation $\mu \subseteq U' \times U''$ such that all its assignments are good and mutually compatible. A match is *maximal* if it is not included in any other match, and is *maximum* if it has largest cardinality. The relational structure matching problem is just the problem of finding a maximum match between two relational structures. When the relational structures being matched are graphs the problem becomes the maximum common subgraph problem, which is known to be NP-complete [126]. Obviously, for this reason, the relational structure matching problem too is NP-complete.

Ambler et al. [11] (see also [36] and [206]) introduced the notion of *association graph* as an auxiliary structure for matching relational structures. The association graph of two relational structures S' and S'' is the undirected graph $G = (V, E)$ defined as

$$V = \{(u', u'') \in U' \times U'' \ : \ (u', u'') \text{ is good}\}$$

and

$$E = \{((u', u''), (v', v'')) \in V \times V \ : \ (u', u'') \text{ and } (v', v'') \text{ are compatible}\} \ .$$

It is clear that, given the way we have constructed the association graph, the notions of match, maximal match, and maximum match turn out to coincide with those of clique, maximal clique, and maximum clique of the association graph, respectively. The problem of matching two relational structures is therefore equivalent to the maximum clique problem.

In many practical applications, properties as well as relations may be assigned one or more numerical attributes. For example, if units represents segmented regions in an image, the property "circular" can be associated with the diameter of the circle; or, the relation which establishes that two regions are adjacent can be assigned a numerical value specifying the (relative) amount of common boundary. It is straightforward to generalize the association graph idea described above to the case of *attributed* relational structures. To this end, we simply need to specify some similarity measure between attribute vectors; the topology of the association graph is then defined according to the degree of similarity between corresponding property's and relation's attributes. An alternative approach is to construct a *weighted* association graph (see, e.g., [175]). In this case, a pair is declared good using the same criterion as in the unattributed case, and the corresponding vertex in the association graph is assigned a positive weight which represents the (overall) similarity between the attribute vectors. In this case, one is interested in determining a clique having largest total weight and this is the maximum weight clique problem.

In [273] and [272], Pelillo uses the association graph framework and the Motzkin-Straus formulation (see Section 2.2) to develop a new quadratic programming formulation for graph isomorphism and related relational structure matching problems. Replicator equations, mentioned in Section 6.4, are used to solve the program and the experimental results obtained show how on the graph isomorphism problem the replicator dynamical system outperforms more sophisticated heuristics based on mean-field theory.

Computer vision and pattern recognition problems for which the maximum clique formulation has proven to be effective include, for example, stereo correspondence [175], object recognition [51], [52], [155], point pattern matching [247], and motion analysis [282], [279]. The framework has recently been extended to handle the problem of matching hierarchical relational structures, such as trees, and applied to the problem of shape matching [275].

8 Conclusions

Over the past four decades, research on the maximum clique and related problems has yielded many interesting and profound results. However, a great deal remains to be learned about the maximum clique problem.

This paper provides an expository survey on complexity, algorithms and applications of the maximum clique problem. Furthermore, an extensive up-to-date bibliography is included. However, the present activity in work related to the maximum clique problem is so extensive that a survey of this nature is outdated before it is written.

Acknowledgments

The authors would like to thank A. Panconesi for reading an early version of this chapter, and providing useful comments and suggestions, as well as M. Dür for valuable hints.

References

[1] E. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machines*, J. Wiley & Sons, Chichester, UK, 1989.

[2] E. Aarts and J.K. Lenstra (eds.), *Local Search in Combinatorial Optimization*, J. Wiley & Sons, Chichester, UK, 1997.

[3] F. Abbattista, F. Bellifemmine and D. Dalbis, The Scout algorithm applied to the maximum clique problem, in *Advances in Soft Computing—Engineering and Design*, Springer, Berlin, 1998.

[4] J. Abello, P.M. Pardalos and M.G.C. Resende, On maximum clique problems in very large graphs, in *External Memory Algorithms, DIMACS Series, AMS* (J. Abello and J. Vitter, Editors) 1999.

[5] L.M. Adleman, Molecular computation of solutions to combinatorial optimization, *Science*, Vol. 266: 1021–1024, 1994.

[6] F. Alizadeh, A sublinear-time randomized parallel algorithm for the maximum clique problem in perfect graphs, in: A. Aggarwal (ed.), *Discrete Algorithms, Proc. 2nd ACM-SIAM Symposium*, Vol. 2: 188–194, 1991.

[7] N. Alon, U. Feige, A. Wigderson and D. Zuckerman, Derandomized graph products, *Comput. Complex.*, Vol. 5: 60–75, 1995.

[8] E.A. Akkoyunlu, The enumeration of maximal cliques of large graphs, *SIAM J. Comput.*, Vol. 2: 1–6, 1973.

[9] N. Alon, L. Babai and A. Itai, A fast and simple randomized parallel algorithm for the maximal independent set problem, *J. Algorithms*, Vol. 7: 567–583, 1986.

[10] N. Alon, M. Krivelevich and B. Sudakov, Finding a large hidden clique in a random graph, in: *Proc. SODA '98—Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco CA*, January 25–27, 1998.

[11] A.P. Ambler, H.G. Barrow, C.M. Brown, R.M. Burstall and R.J. Popplestone, A versatile computer-controlled assembly system, in *Proc. 3rd Int. J. Conf. Artif. Intell.*: 298–307, 1973.

[12] A.T. Amin and S.L. Hakimi, Upper bounds on the order of a clique of a graph, *SIAM J. Appl. Math.*, Vol. 22: 569–573, 1972.

[13] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy, Proof verification and hardness of approximation problems, *Proc. 33rd Ann. Symp. Found. Computer Sci.*: 14–23, 1992.

[14] S. Arora and S. Safra, Probabilistic checking of proofs: A new characterization of NP, *Proc. 33rd Ann. Symp. Found. Computer Sci.*: 2–13, 1992.

[15] J.G. Auguston and J. Minker, An analysis of some graph theoretical cluster techniques, *J. ACM*, Vol. 17: 571–588, 1970.

[16] G. Ausiello, P. Crescenzi and M. Protasi, Approximation solution of NP optimization problems, *Theor. Comput. Sci.*, Vol. 150: 1–55, 1995.

[17] G. Avondo-Bodeno, *Economic Applications of the Theory of Graphs*, Gordon and Breach Science Publishers, New York, 1962.

[18] S.M. Baas, M.C. Bonvanie and A.J. Verschoor, A relaxation method for the set packing problem using polyhedron characteristic, *Technical Report 699*, University of Twente, Netherlands, 1988.

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

[20] L. Babel, A fast algorithm for the maximum weight clique problem, *Computing*, Vol. 52: 31–38, 1994.

[21] L. Babel and G. Tinhofer, A branch and bound algorithm for the maximum clique problem, *ZOR–Methods and Models of Operations Research*, Vol. 34: 207–217, 1990.

[22] T. Bäck and S. Khuri, An evolutionary heuristic for the maximum independent set problem, *Proc. 1st IEEE Conf. Evolutionary Comput.*: 531–535, 1994.

[23] E. Balas, A fast algorithm for finding an edge-maximal subgraph with a TR-formative coloring, *Discr. Appl. Math.*, Vol. 15: 123–134, 1986.

[24] E. Balas, S. Ceria, G. Corneujols and G. Pataki, Polyhedral methods for the maximum clique problem, in [189]: 11–28, 1996.

[25] E. Balas, V. Chvátal and J. Nešetril, On the maximum weight clique problem, *Math. Oper. Res.*, Vol. 12: 522–535, 1987.

[26] E. Balas and W. Niehaus, Finding large cliques in arbitrary graphs by bipartite matching, in [189]: 29–51, 1996.

[27] E. Balas and H. Samuelsson, A Node Covering Algorithm, *Naval Research Logistics Quarterly*, Vol. 24: 213–233, 1977.

[28] E. Balas and P. Toth, Branch and bound methods, In: E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan and D.B. Shmoys (Eds.), *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, J. Wiley & Sons, Chichester, UK: 361–403, 1985.

[29] E. Balas and J. Xue, Minimum weighted coloring of triangulated graphs, with application to maximum weight vertex packing and clique finding in arbitrary graphs, *SIAM J. Comput.*, Vol. 2: 209–221, 1991. "Addendum," *SIAM J. Comput.*, Vol. 21: 1000, 1992.

[30] E. Balas and J. Xue, Weighted and unweighted maximum clique algorithms with upper bounds from fractional coloring, *Algorithmica* Vol. 15: 397–412, 1996.

[31] E. Balas and C.S. Yu, Finding a maximum clique in an arbitrary graph, *SIAM J. Comput.*, Vol. 14: 1054–1068, 1986.

[32] E. Balas and C.S. Yu, On graphs with polynomially solvable maximum-weight clique problem, *Networks*, Vol. 19: 247–253, 1989.

[33] D.H. Ballard and M. Brown, *Computer Vision*, Prentice-Hall, Englewood Cliffs, N.J., 1982.

[34] D.H. Ballard, P.C. Gardner and M.A. Srinivas, Graph problems and connectionist architectures, *Technical Report TR 167*, Dept. Computer Science, University of Rochester, 1987

[35] F. Barahona, A. Weintraub, and R. Epstein, Habitat dispersion in forest planning and the stable set problem, *Oper. Res.* Vol. 40, Supp. 1: S14–S21, 1992.

[36] H.G. Barrow and R.M. Burstall, Subgraph isomorphism, matching relational structures and maximal cliques, *Inform. Proc. Lett.*, Vol. 4: 83–84, 1976.

[37] R. Battiti and M. Protasi, Reactive local search for the maximum clique problem, *Technical Report TR-95-052*, International Computer Science Institute, Berkeley, CA, 1995.

[38] A.R. Bednarek and O.E. Taulbee, On maximal chains, *Roum. Math. Pres et Appl.*, Vol. 11: 23–25, 1966.

[39] L.W. Beineke, A survey of packings and coverings in graphs, in: G. Chartrand and S. Kapoor (Eds.), *Many Facets of Graph Theory*, Springer: 45–53, Berlin, 1969.

[40] M. Bellare, O. Goldreich, C. Lund and A. Russell, Efficient probabilistically checkable proofs and application to approximation, *Proc. 25th Ann. ACM Symp. Theory of Comput.*: 294–304, 1993.

[41] M. Bellare, O. Goldreich and M. Sudan, Free bits, PCPs and non-approximability—Towards tight results, *SIAM J. Comput.* Vol. 27: 804–915, 1997.

[42] M. Bellare and M. Sudan, Improved nonapproximability results, *Proc. 26th Ann. ACM Symp. Theory of Comput.*: 184–193, 1994.

[43] C. Berge and V. Chvátal (Eds.), Topics on Perfect Graphs, *Ann. Discr. Math.*, Vol. 21, 1984.

[44] P. Berman and A. Pelc, Distributed fault diagnosis for multiprocessor systems, *Proc. of the 20th Annual Int. Symp. on Fault-Tolerant Computing* (Newcastle, UK): 340-346, 1990.

[45] P. Berman and G. Schnitger, On the complexity of approximating the independent set problem, in *Proc.1989 Symposium on Theoret. Aspects of Comp. Sci.*, Springer, Lecture Notes in Computer Sciences Vol. 349: 256–268, Berlin, 1989.

[46] P. Berman and G. Schnitger, On the complexity of approximating the independent set problem, *Inform. and Comput.*, Vol. 96: 77–94, 1992.

[47] A. Bertoni, P. Campadelli and G. Grossi, A discrete neural algorithm for the maximum clique problem: Analysis and circuit implementation, presented at *WAE'97: Int. Workshop on Algorithm Engineering*, Venice, Italy, 1997.

[48] B.K. Bhattacharya, P. Hell and J. Huang, A linear algorithm for maximum weight cliques in proper circular arc graphs, *SIAM J. Discr. Math.* Vol. 9: 274–289, 1996.

[49] B.K. Bhattacharya and D. Kaller, An $O(m+n\log n)$ algorithm for the maximum clique problem in circular-arc graphs, *J. Algorithms* Vol. 25: 33–358, 1997.

[50] D.M. Blough, Fault detection and diagnosis in multiprocessor systems, *Ph.D. Thesis, The Johns Hopkins University*, 1988.

[51] R.C. Bolles and R.A. Cain, Recognizing and locating partially visible objects: The locus-feature-focus method, *Int. J. Robotics Res.*, Vol. 1: 57–82, 1982.

[52] R.C. Bolles and P. Horaud, 3DPO: A three-dimensional part orientation system, *Int. J. Robotics Res.*, Vol. 5: 3–26, 1986.

[53] B. Bollobás, *Random graphs*, Academic Press, New York, NY, 1985.

[54] B. Bollobás, *Modern graph theory*, Springer, New York, NY, 1998.

[55] B. Bollobás and P. Erdös, Cliques in Random Graphs, *Math. Proc. Cambridge Philos. Soc.*, Vol. 80: 419–427, 1976.

[56] I.M. Bomze, Evolution towards the maximum clique, *J. Glob. Optim.*, Vol. 10: 143–164, 1997.

[57] I.M. Bomze, Global escape strategies for maximizing quadratic forms over a simplex, *J. Global Optim.* Vol. 11: 325–338, 1997.

[58] I.M. Bomze, On standard quadratic optimization problems, *J. Glob. Optim.* Vol. 13: 369–387, 1998.

[59] I.M. Bomze, M. Budinich, M. Pelillo and C. Rossi, Annealed replication: A new heuristic for the maximum clique problem, submitted for publication, 1998.

[60] I.M. Bomze and G. Danninger, A finite algorithm for solving general quadratic problems, *J. Global Optim.* Vol. 4: 1–16, 1994.

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

[62] I.M. Bomze, M. Pelillo, and V. Stix, Approximating the Maximum Weight Clique: An Evolutionary Game Theory Approach, manuscript in preparation, 1998.

[63] I.M. Bomze and F. Rendl, Replicator dynamics for the evolution towards the maximum clique: Variants and experiments, in: *High Performance Algorithms and Software in Nonlinear Optimization*, R. De Leone, A. Murlí, P.M. Pardalos and G. Toraldo (eds.), Kluwer: 53–67, Dordrecht, 1998.

[64] I.M. Bomze and V. Stix, Genetical engineering via negative fitness: Evolutionary dynamics for global optimization, to appear in: *Ann. Oper. Res.* Vol. 90, 1999.

[65] R.E. Bonner, On some clustering techniques, *IBM J. Res. Develop.*, Vol. 8: 22–32, 1964.

[66] R. Boppana and M.M. Halldórsson, Approximating maximum independent sets by excluding subgraphs, *BIT*, Vol. 32: 180–196, 1992.

[67] M. Boulala and J.P. Uhry, Polytope des Independants dans un Graphe Serie-Parallele, *Discr. Math.*, Vol. 27: 225–243, 1979.

[68] J.-M. Bourjolly, P. Gill, G. Laporte and H. Mercure, An exact quadratic 0-1 algorithm for the stable set problem, in [189]: 53–73, 1996.

[69] J.-M. Bourjolly, G. Laporte and H. Mercure, A combinatorial column generation algorithm for the maximum stable set problem, *Oper. Res. Lett.* Vol. 20: 21–29, 1997.

[70] D.P. Bovet and P. Crescenzi, *Introduction to the Theory of Complexity*, Prentice-Hall, Englewood Cliffs, N.J., 1994.

[71] D. Brelaz, New methods to color the vertices of a graph, *Commun. ACM*, Vol. 22: 251–256, 1979.

[72] M. Brockington and J.C. Culberson, Camouflaging independent sets in quasi-random graphs, in [189]: 75–88, 1996.

[73] C. Bron and J. Kerbosch, Algorithm 457: Finding all cliques of an undirected graph, *Commun. ACM*, Vol. 16: 575–577, 1973.

[74] A. E. Brouwer, J. B. Shearer, N. J. A. Sloane and W. D. Smith, A new Table of constant weight codes, *IEEE Trans. Inform. Theory*, Vol. 36: 1334–1380, 1990.

[75] M. Budinich, Bounds on the maximum clique of a graph, *submitted* (`http://www.ts.infn.it/~mbh/MC_Bounds.ps.Z`).

[76] M. Budinich and P. Budinich, A Clifford algebra formulation of the maximum clique problem of a graph, preprint in preparation (will be available from `http://www.ts.infn.it/~mbh/PubNN.html`).

[77] T.N. Bui and P.H. Eppley A hybrid genetic algorithm for the maximum clique problem, *Proc. 6th Int. Conf. Genetic Algorithms*: 478–484, 1995.

[78] M. Burlet and J. Fonlupt, Polynomial algorithm to recognize a Meyniel graph, W.R. Pulleyblank (ed.), *Progress in Combinatorial Optimization*, Academic Press, Toronto: 69–99, 1984.

[79] J.E. Burns, The maximum independent set problem for cubic planar graph, *Networks*, Vol. 9: 373–378, 1989

[80] R. Carraghan and P.M. Pardalos, An exact algorithm for the maximum clique problem, *Oper. Res. Lett.*, Vol. 9: 375–382, 1990.

[81] R. Carraghan and P.M. Pardalos, A parallel algorithm for the maximum weight clique problem, *Technical Report CS-90-40*, Dept. of Computer Science, Penn. State Univ., 1990.

[82] B. Carter and K. Park, How good are genetic algorithms at finding large cliques: An experimental study, *Technical Report BU-CS-93-015*, Computer Science Dept., Boston University, 1993.

[83] M.-S. Chang, Y.-H. Chen, G.J. Chang, and J.-H. Yan, Algorithmic aspects of the generalized clique-transversal problem on chordal graphs, *Discr. Appl. Math.*, Vol. 66: 189–203, 1996.

[84] R.C. Chang, On the query complexity of clique size and maximum satisfiability, *J. Comput. Syst. Sci.*, Vol. 53: 298–313, 1996.

[85] R.C. Chang, W.I. Gasarch, C. Lund, On bounded queries and approximation, *SIAM J. Comput.*, Vol. 26: 188–209, 1997.

[86] R.C. Chang and H.S. Lee, Finding a maximum set of independent chords in a circle, *Inform. Proc. Lett.*, Vol. 41: 99–102, 1992.

[87] N. Chiba and T. Nishizeki, Arboricity and subgraph listing algorithms, *SIAM J. Comput.*, Vol. 14: 210–223, 1985.

[88] N. Chiba, T. Nishizeki and N. Saito, An approximation algorithm for the maximum independent set problem on planar graphs, *SIAM. J. Comput.*, Vol. 11: 663–675, 1982.

[89] N. Chiba, T. Nishizeki and N. Saito, An algorithm for finding a large independent set in planar graphs, *Networks*, Vol. 13: 247–252, 1983.

[90] E. Choukhmane and J. Franco, An approximation algorithm for the maximum independent set problem in cubic planar graphs, *Networks*, Vol. 16: 349–356, 1986.

[91] M. Chrobak and J. Naor, An efficient parallel algorithm for computing a large independent set in a planar graph, *Algorithmica*, Vol. 6: 801–815, 1991.

[92] V. Chvátal, On certain polytopes associated with graphs, *J. Combin. Theory B*, Vol. 18: 138–154, 1975.

[93] V. Chvátal, Determining the stability number of a graph, *SIAM J. Comput.*, Vol. 6: 643–662. 1977.

[94] V. Chvátal, A greedy heuristic for the set-covering problem, *Math. Oper. Res.*, Vol. 4: 233–23, 1979.

[95] V. Chvátal, Perfectly orderable graphs, *Ann. Discr. Math.*, Vol. 21: 63–65, 1985.

[96] K. Corradi and S. Szabo, A combinatorial approach for Keller's conjecture, *Periodica Mathematica Hungarica*, Vol. 21: 95–100, 1990.

[97] P. Crescenzi, C. Fiorini and R. Silvestri, A Note on the approximation of the MAX CLIQUE problem, *Inform. Process. Lett.* 40: 1-5, 1991.

[98] F. Della Croce and R. Tadei, A multi-KP modeling for the maximum-clique problem, *Europ. J. Oper. Res.*, Vol. 73: 555–561, 1994.

[99] V.-D. Cung, S. Dowaji, B. Le Cun, T. Mautor and C. Roucairol, Concurrent data structures and load balancing strategies for parallel branch-and-bound/A^* algorithms, in S.N. Bhatt (ed.), *Parallel algorithms, 3rd DIMACS Implementation Challenge, October 17-19, 1994, AMS DIMACS Ser. Discrete Math. Theor. Comput. Sci.* Vol. 30: 141–162, 1997.

[100] D.M. Cvetković, M. Doob and H. Sachs, *Spectra of Graphs*, Academic Press, New York, NY, 1980.

[101] E. Dahlhaus and M. Karpinski, A fast parallel algorithm for computing all maximal cliques in a graph and the related problems, in: Algorithm Theory. Proc. 1st Scand. Workshop, Halmstad/Sweden 1988. Lect. Notes Comput. Sci. Vol. 318: 139–144, 1988.

[102] C. De Simone and A. Sassano, Stability number and bull-and chair-free graphs, *Discr. Appl. Math.*, Vol. 41: 121–129, 1993.

[103] V. Degot and J.M. Hualde, De l'utilisation de la notion de clique en matière de typologie des populations (in French), R.A.I.R.O., Vol. 9: 5–18, 1975.

[104] N. Deo, *Graph Theory with Applications to Engineering and Computer Science*, Prentice-Hall, Englewood Cliffs, NJ, 1974.

[105] J.F. Desler and S.L. Hakimi, On finding a maximum internally stable set of a graph, *Proc. of Fourth Annual Princeton Conference on Information Sciences and Systems*, Vol. 4: 459-462, Princeton, NJ, 1970.

[106] G.A. Dirac, A property of 4-chromatic graphs and some remarks on critical graphs, *J. London Math. Soc.*, Vol. 27: 85–92, 1952.

[107] G.A. Dirac, On rigid circuit graphs, *Abh. Math. Sem.*, Univ. Hamburg, Vol. 25: 71–76, 1961.

[108] P. Doreian, A note on the detection of cliques in valued graphs, *Sociometry*, Vol. 32: 237–242, 1969.

[109] D.-Z. Du, B. Gao and W. Wu, A special case for subset interconnection designs, *Discr. Appl. Math.*, Vol. 78: 51–60, 1997.

[110] C. Ebenegger, P.L. Hammer, and D. de Werra, Pseudo-boolean functions and stability of graphs, *Ann. Discr. Math.*, Vol. 19: 83–98, 1984.

[111] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy, Approximating the maximum clique is almost NP-complete, *Proc. 32nd IEEE Symp. on Foundations of Computer Science*: 2–12, 1991.

[112] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy, Interactive proofs and the hardness of approximating cliques, *J. ACM*, Vol. 43: 268–292, 1996.

[113] U. Feige and J. Kilian, Two prover protocols—Low error at affordable rates, *Proc. 26th Ann. ACM Symp. Theory of Comput.*: 172–183, 1994.

[114] S. Felsner, R. Mueller and L. Wernisch, Trapezoid graphs and generalizations, geometry and algorithms, *Discr. Appl. Math.*, Vol. 74: 13–32, 1997.

[115] T.A. Feo, M.G.C. Resende and S.H. Smith, A greedy randomized adaptive search procedure for maximum independent set, *Oper. Res.*, Vol. 42: 860–878, 1994.

[116] M.L. Fisher and L.A. Wolsey, On the greedy heuristic for continuous covering and packing problems, *SIAM J. Alg. Discr. Meth.*, Vol. 3: 584–591, 1982.

[117] C. Fleurent and J.A. Ferland, Object-oriented implementation of heuristic search methods for graph coloring, maximum clique, and satisfiability, in [189]: 619–652, 1996.

[118] J.A. Foster and T. Soule, Using genetic algorithms to find maximum cliques, *Technical Report LAL 95-12*, Dept. of Computer Science, U. Idaho, 1995.

[119] A. Frank, Some polynomial algorithms for certain graphs and hypergraphs, *Proc. 5th British Combin. Conf.*: 211–226, 1976.

[120] C. Friden, A. Hertz, and D. de Werra, STABULUS: A technique for finding stable sets in large graphs with tabu search, *Computing*, Vol. 42: 35–44, 1989.

[121] C. Friden, A. Hertz and M. de Werra, TABARIS: An exact algorithm based on tabu search for finding a maximum independent set in a graph, *Comput. Oper. Res.*, Vol. 17: 437–445, 1990.

[122] A. Frieze, On the independence number of random graphs, *Discr. Math.*, Vol. 81: 171–175, 1990.

[123] K. Fujisawa, M. Kojima, and K. Nakata, Exploiting sparsity in primal-dual interior-point methods for semidefinite programming, *Math. Programming*, Vol. 79: 235–253, 1997.

[124] N. Funabiki, Y. Takefuji, and K.C. Lee, A neural network model for finding a near-maximum clique, *J. Parallel Distrib. Comput.*, Vol. 14: 340–344, 1992.

[125] M. Garey and D. Johnson, The complexity of near-optimal coloring, *J. ACM*, Vol. 23: 43–49, 1976.

[126] M. Garey and D. Johnson, *Computers and Intractability—A guide to the Theory of NP-Completeness*, Freeman, San Francisco, 1979.

[127] F. Gavril, Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph, *SIAM J. Comput.*, Vol. 1: 180–187, 1972.

[128] F. Gavril, Algorithms for a maximum clique and a maximum independent set of a circle graph, *Networks*, Vol. 3: 261–273, 1973.

[129] F. Gavril, Algorithms on circular arc graphs, *Networks*, Vol. 4: 357–369, 1974.

[130] M. Gendreau, J.C. Picard and L. Zubieta, An efficient implicit enumeration algorithm for the maximum clique problem, A. Kurzhanski et al. (eds), *Lecture Notes in Economics and Mathematical Systems*, Vol. 304: 70–91, Springer, Berlin 1988.

[131] A. Gendreau, L. Salvail, and P. Soriano, Solving the maximum clique problem using a tabu search approach, *Ann. Oper. Res.*, Vol. 41: 385–403, 1993.

[132] L. Gerhards and W. Lindenberg, Clique detection for nondirected graphs: Two new algorithms, *Computing*, Vol. 21: 295–322, 1979.

[133] A.M.H. Gerards and A. Schrijver, Matrices with the Edmonds-Johnson property, *Combinatorica*, Vol. 6: 403–417, 1986.

[134] L.E. Gibbons, D.W. Hearn and P.M. Pardalos, A continuous based heuristic for the maximum clique problem, in [189]: 103–124, 1996.

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

[136] F. Glover, Tabu search—Part I, *ORSA J. Comput.*, Vol. 1: 190–260, 1989.

[137] F. Glover, Tabu search—Part II, *ORSA J. Comput.*, Vol. 2: 4–32, 1990.

[138] F. Glover and M. Laguna, Tabu search, in *Modern Heuristic Techniques for Combinatorial Problems*, C. Reeves (ed.), Blackwell, Oxford, UK: 70–141, 1993.

[139] G.H. Godbeer, J. Lipscomb, and M. Luby, On the computational complexity of finding stable state vectors in connectionist models (Hopfield nets), *Technical Report 208/88*, Dept. of Computer Science, University Toronto, 1988.

[140] M.X. Goemans, Semidefinite programming in combinatorial optimization, *Math. Programming*, Vol. 79: 143–161, 1997.

[141] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA, 1989.

[142] M.K. Goldberg and R.D. Rivenburgh, Constructing cliques using restricted backtracking, in [189]: 89–101, 1996.

[143] M. Goldberg and T. Spencer, A new parallel algorithm for the maximal independent set problem, *SIAM J. Comput.*, Vol. 18: 419-427, 1989.

[144] M. Goldberg and T. Spencer, Constructing a maximal independent set in parallel, *SIAM J. Discr. Math.*, Vol. 2: 322–328, 1989.

[145] M.C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, NY, 1980.

[146] M.C. Golumbic and P.L. Hammer, Stability in circular-arc-graphs, *J. Algorithms*, Vol. 9: 314–320, 1988.

[147] G.R. Grimmett and W.R. Pulleyblank, Random near-regular graphs and the node packing problem, *Oper. Res. Lett.*, Vol. 4: 169–174, 1985.

[148] T. Grossman, Applying the INN model to the max clique problem, in [189]: 125–146, 1996.

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

[150] M. Grötschel, L. Lovász and A. Schrijver, Relaxations of vertex packing, *J. Combin. Theory B*, Vol. 40: 330–343, 1986.

[151] M. Grötschel, L. Lovász and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, 2nd Ed., Springer, Berlin, 1993.

[152] M. Grötschel, L. Lovász and A. Schrijver, Polynomial algorithms for perfect graphs, *Ann. Discr. Math.*, Vol. 21: 325–356, 1989.

[153] U.I. Gupta, D.T. Lee and J.Y.T. Leung, Efficient algorithms for interval graphs and circular-arc graphs, *Networks*, Vol. 12: 459–467, 1982.

[154] G. Hajós, Sur la factorisation des abeliens, *Casopis* Vol. 74: 189–196, 1950.

[155] M.-H. Han and D. Jang, The use of maximum curvature points for the recognition of partially occluded objects, *Pattern Recognition*, Vol. 23: 21–33, 1990.

[156] P. Hansen and B. Jaumard, Algorithms for the maximum satisfiability problem, *Computing*, Vol. 44: 279–303, 1990.

[157] F. Harary, *Graph Theory*, Addison-Wesley, Reading, MA, 1969.

[158] F. Harary and I.C. Ross, A procedure for clique detection using the group matrix, *Sociometry*, Vol. 20: 205–215, 1957.

[159] J. Hasselberg, P.M. Pardalos and G. Vairaktarakis, Test case generators and computational results for the maximum clique problem, *J. Global Optim.*, Vol. 3: 463–482, 1993.

[160] J. Hastad, Testing of the long code and hardness clique, *Proc. 28th Ann. Symp. Theory of Comput.*: 11–19, 1996.

[161] J. Hastad, Clique is hard to approximate within $n^{1-\varepsilon}$, *Proc. 37th Ann. Symp. Found. Comput. Sci.*: 627–636, 1996.

[162] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Macmillan, New York, 1994.

[163] R.B. Hayward, Weakly triangulated graphs, *J. of Combin. Theory B*, Vol. 39: 200–209, 1985.

[164] R. Hayward, C.T. Hoang and F. Maffray, Optimizing weakly triangulated graphs, *Graphs and Combinatorics*, Vol. 5: 339–349, 1989. See also *Erratum*, Vol. 6: 33–35, 1990.

[165] I. Heller, On linear systems with integral valued solutions, *Pacific J. Math.*, Vol. 7: 1351–1364, 1957.

[166] C. Helmberg, F. Rendl, R.J. Vanderbei and H. Wolkowicz, An interior-point method for semidefinite programming, *SIAM J. Optim.*, Vol. 6: 342–361, 1996.

[167] J. Hertz, A. Krogh and R.G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Redwood City, CA, 1991.

[168] A. Hertz, E. Taillard and D. de Werra, Tabu search, in [2]: 121–136.

[169] M. Hifi, A genetic algorithm-based heuristic for solving the weighted maximum independent set and some equivalent problems, *J. Oper. Res. Soc.* Vol. 48: 612–622, 1997.

[170] C.W. Ho and R.C.T. Lee, Efficient parallel algorithms for finding maximal cliques, clique trees, and minimum coloring of chordal graphs, *Inform. Proc. Lett.*, Vol. 28: 301–309, 1988.

[171] J. Hofbauer and K. Sigmund, *Evolutionary Games and Population Dynamics*, Cambridge University Press, Cambridge, UK, 1998.

[172] J.H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI, 1975.

[173] S. Homer and M. Peinado, Experiments with polynomial-time CLIQUE approximation algorithms on very large graphs, in [189]: 147–167, 1996.

[174] J.J. Hopfield and D.W. Tank, "Neural" computation of decisions in optimization problems, *Biol. Cybern.*, Vol. 52: 141–152, 1985.

[175] R. Horaud and T. Skordas, Stereo correspondence through feature grouping and maximal cliques, *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 11: 1168–1180, 1989.

[176] R.A. Horn and C.R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, UK, 1985.

[177] D.J. Houck, On the vertex packing problem, Ph.D. dissertation, The Johns Hopkins University, Baltimore, 1974.

[178] D.J. Houck and R.R. Vemuganti, An algorithm for the vertex packing problem, *Oper. Res.*, Vol. 25: 773–787, 1977.

[179] W.L. Hsu, Maximum weight clique algorithms for circular-arc graphs and circle graphs, *SIAM J. Comput.*, Vol. 14: 224–231, 1985.

[180] W.L. Hsu, The coloring and maximum independent set problems on planar perfect graphs, *J. ACM*, Vol. 35: 535–563, 1988.

[181] W. Hsu, Y. Ikura and G.L. Nemhauser, A polynomial algorithm for maximum weighted vertex packings on graphs without long odd cycles, *Math. Programming*, Vol. 20: 225–232, 1981.

[182] H.B. Hunt III, M.V. Marathe, V. Radhakrishnan and R.E. Stearns, The complexity of planar counting problems, *SIAM J. Comput.*, Vol. 27: 1142–1167, 1998..

[183] A. Jagota, Approximating Maximum Clique with a Hopfield Network, *IEEE Trans. Neural Networks*, Vol. 6: 724–735, 1995.

[184] A. Jagota (ed.), Neural Networks for Combinatorial Optimization, *J. Artif. Neural Networks*, Vol. 2, 1995.

[185] A. Jagota and K.W. Regan, Performance of neural net heuristics for maximum clique on diverse highly compressible graphs, *J. Global Optim.*, Vol. 10: 439–465, 1997.

[186] A. Jagota, L. Sanchis, and R. Ganesan, Approximately solving maximum clique using neural networks and related heuristics, in [189]: 169–204, 1996.

[187] M. Jerrum, Large cliques elude the Metropolis process, *Random Structures and Algorithms*, Vol. 3: 347–359, 1992.

[188] D.S. Johnson, Approximation algorithms for combinatorial problems, *J. Comput. Syst. Sci.*, Vol. 9: 256-278, 1974.

[189] D.S. Johnson and M.A. Trick (eds.), *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge*, DIMACS Vol. 26, American Mathematical Society, 1996 (see also `http://dimacs.rutgers.edu/Volumes/Vol26.html`).

[190] D.S. Johnson, M. Yannakakis and C.H. Papadimitriou, On generating all maximal independent sets, *Inform. Proc. Lett.*, Vol. 27: 119–123, 1988.

[191] L.F. Johnson, Determining cliques of a graph, *Proc. 5th Manitoba Conf. on Numer. Math.*: 429–437, 1975.

[192] H.C. Johnston, Cliques of a graph: Variations on the Bron-Kerbosch algorithm, *Int. J. Comput. Inform. Sci.*, Vol. 5: 209–238, 1976.

[193] N. Karmarkar, K.G. Ramakrishnan and M.G.C. Resende, An interior point approach to the maximum independent set problem in dense random graphs, *Proc. XV Latin American Conference on Informatics, Santiago, Chile*, I: 241–260, 1989.

[194] R.M. Karp, Reducibility among combinatorial problems, in *Complexity of Computer Computations*, R.E. Miller and J.W. Thatcher (eds.), Plenum Press, New York: 85–103, 1972.

[195] R.M. Karp and A. Wigderson, A fast parallel algorithm for the maximal independent set problem, *J. ACM*, Vol. 32: 762–773, 1985.

[196] M. Karpinski and A. Zelikovsky, Approximating dense cases of covering problems, in P.M. Pardalos et al. (eds), *Network Design: Connectivity and Facilities Location, DIMACS Workshop, April 28–30, 1997. AMS-DIMACS Ser. Discr. Math. Theor. Comput. Sci.* Vol. 40: 169–178, 1998.

[197] T. Kashiwabara, S. Masuda, K. Nakajima and T. Fujisawa, Generation of maximum independent sets of a bipartite graph and maximum cliques of a circular-arc-graph, *J. Algorithms*, Vol. 13: 161–174, 1992.

[198] O.H. Keller, Über die lückenlose Erfüllung des Raumes mit Würfeln (in German), *J. Reine Angew. Math.*, Vol. 163: 231–248, 1930.

[199] P. Kikusts, Another algorithm determining the independence number of a graph, *Elektron. Inf. Verarb. Kybern.* ELK 22: 157–166, 1986.

[200] D.S. Kim and D.-Z. Du, On conflict-free channel set assignments for optical cluster-based hypercube networks, In *DIMACS Series Vol. 46*: 109-116, American Mathematical Society (1999).

[201] S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi, Optimization by simulated annealing, *Science*, Vol. 220: 671–680, 1983.

[202] P.N. Klein, Efficient parallel algorithms for chordal graphs, *Proc. 29th Ann. Symp. Found. Computer Sci.*: 150–161, 1988.

[203] J. Kleinberg and M.X. Goemans, The Lovász theta function and a semidefinite programming relaxation of vertex cover, *SIAM J. Discr. Math.* Vol. 11: 196–204, 1997.

[204] D.E. Knuth, The sandwich theorem, *Electr. J. Combin.*, Vol. 1: A1, 1994 (`http://www2.combinatorics.org/Volume_1/volume1.html#A1`).

[205] R. Kopf and G. Ruhe, A computational study of the weighted independent set problem for general graphs, *Found. Control Engin.*, Vol. 12: 167–180, 1987.

[206] D. Kozen, A clique problem equivalent to graph isomorphism, *SIGACT News*: 50–52, Summer 1978.

[207] J.C. Lagarias and P.W. Shor, Keller's cube-tiling conjecture is false in high dimensions, *Bull. Amer. Math. Soc. New Ser.*, Vol. 27: 279–283, 1992.

[208] P.J.M. van Laarhoven and E.H.L. Aarts, *Simulated Annealing: Theory and Applications*, Reidel, Dordrecht, 1987.

[209] E.L. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart and Winston, 1976.

[210] L.J. Leifman, On construction of all maximal complete subgraphs (cliques) of a graph, Technical Report, Dept. of Mathematics., University of Haifa, Israel, 1976.

[211] F. Lin and K. Lee, A parallel computation network for the maximum clique problem, in *Proc. 1st Int. Conf. Fuzzy Theory Tech.*, Baton Rouge, Louisiana, 1992.

[212] R.J. Lipton, On proving that a graph has no large clique: A connection with Ramsey theory, *Inform. Proc. Lett.*, Vol. 58: 39–42, 1996.

[213] C.-K. Looi, Neural network methods in combinatorial optimization, *Comput. Oper. Res.*, Vol. 19: 191–208, 1992.

[214] E. Loukakis, A new backtracking algorithm for generating the family of maximal independent sets of a graph, *Comp. Math. Appl.*, Vol. 9: 583–589, 1983.

[215] E. Loukakis and C. Tsouros, A depth first search algorithm to generate the family of maximal independent sets of a graph lexicographically, *Computing*, Vol. 27: 249–266, 1981.

[216] E. Loukakis and C. Tsouros, Determining the number of internal stability of a graph, *Int. J. Comp. Math.*, Vol. 11: 207–220, 1982.

[217] E. Loukakis and C. Tsouros, An algorithm for the maximum internally stable set in a weighted graph, *Int. J. Comput. Math.*, Vol. 13: 117–129, 1983.

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

[219] L. Lovász and A. Schrijver, Cones of matrices and set-functions and 0-1 optimization, *SIAM J. Optim.*, Vol. 1: 166–190, 1991.

[220] M. Luby, A simple parallel algorithm for the maximum independent set problem, *SIAM J. Comput.*, Vol. 15: 1036–1053, 1986.

[221] J. MacWilliams and N.J.A. Sloane, *The Theory of Error Correcting Codes*, North-Holland, Amsterdam, 1979.

[222] K. Maghout, Sur la determination des nombres de stabilite et du nombre chromatique d'un graphe, *C.R. Acad. Sci.*, Paris, Vol. 248: 2522–2523, 1959.

[223] G.K. Manacher and T.A. Mankus, Finding a maximum clique in a set of proper circular arcs in $O(n)$ with applications, *Int. J. Found. Comput. Sci.*, Vol. 8: 443–467, 1997.

[224] C. Mannino and A. Sassano, Edge projection and the maximum cardinality stable set problem, in [189]: 205–219, 1996.

[225] E. Marchiori, A simple heuristic based genetic algorithm for the maximum clique problem, *Proc. ACM Symp. Appl. Comput.*: 366–373, 1998.

[226] P.M. Marcus, Derivation of maximal compatibles using boolean algebra, *IBM J. Res. Develop.*, Vol. 8: 537–538, 1964.

[227] S. Masuda and K. Nakajima, An optimal algorithm for finding a maximum independent set of a circular-arc graph, *SIAM J. Comput.*, Vol. 17: 41–52, 1988.

[228] S. Masuda, K. Nakajima, T. Kashiwabara and T. Fujisawa, Efficient algorithms for finding maximum cliques of an overlap graph, *Networks*, Vol. 20: 157–171, 1990.

[229] D.W. Matula, On the complete subgraphs of a random graph, *Proc. 2nd Conf. Combin. Theory Appl.*, Chapel Hill, NC: 356–369, 1970.

[230] D.W. Matula, The largest clique size in a random graph, *Technical Report CS 7608*, Department of Computer Science, Southern Methodist University, 1976.

[231] W. Meeusen and L. Cuyvers, Clique detection in directed graphs: A new algorithm, *J. Comput. Appl. Math.*, Vol. 1: 185–193, 1975.

[232] W. Meeusen and L. Cuyvers, An annotated bibliography of clique-detection algorithms and related graph-theoretical problems, *CAM Discussion Paper 7909*, State University Center Antwerp, 1979.

[233] H. Meyniel, On the perfect graph conjecture, *Discr. Math.*, Vol. 16: 339–342, 1976.

[234] H. Meyniel, A new property of imperfect graphs and some consequences, *Europ. J. Combin.*, Vol. 8: 313–316, 1987.

[235] G.J. Minty, On maximal independent sets of vertices in claw-free graphs, *J. Combin. Theory B*, Vol. 28: 284–304, 1980.

[236] B. Mohar and S. Poljak, Eigenvalues in combinatorial optimization, In *Combinatorial and Graph-Theoretic Problems in Linear Algebra* R. Brualdi, S. Friedland and V. Klee, eds., IMA Volumes in Mathematics and Its Applications, Vol. 50, Springer, Berlin, 1993.

[237] J.W. Moon and L. Moser, On cliques in graphs, *Israel Journal of Mathematics*, Vol. 3: 23-28, 1965.

[238] T.S. Motzkin and E.G. Straus, Maxima for graphs and a new proof of a theorem of Turán, *Canad. J. Math.*, Vol. 17: 533–540, 1965.

[239] G.D. Mulligan and D.G. Corneil, Corrections to Bierstone's algorithm for generating cliques, *J. ACM*, Vol. 19: 244–247, 1972.

[240] W.J. Murphy, Computing independent sets in graphs with large girth, *Discr. Appl. Math.*, Vol. 36: 167–170, 1992.

[241] A.S. Murthy, G. Parthasarathy and V.U.K. Sastry, Clique finding—A genetic approach, *Proc. 1st IEEE Conf. Evolutionary Comput.*: 18–21, 1994.

[242] J. Naor, M. Naor and A.A. Schaffer, Fast parallel algorithms for chordal graphs, *Proc. 19th Annual ACM Symp. Theory Comput.*: 355–364, 1987.

[243] G.L. Nemhauser and G. Sigismondi, A strong cutting plane/branch-and bound algorithm for node packing, *J. Opl. Res. Soc.*, Vol. 43: 443–457, 1992.

[244] G.L. Nemhauser and L.E. Trotter,, Properties of vertex packings and independence system polyhedra, *Math. Programming*, Vol. 6: 48–61, 1974.

[245] G.L. Nemhauser and L.E. Trotter, Vertex packings: Structural properties and algorithms, *Math. Programming*, Vol. 8: 232–248, 1975.

[246] G.L. Nemhauser and L.A. Wolsey, *Integer and Combinatorial Optimization*, J. Wiley & Sons, New York, 1988.

[247] H. Ogawa, Labeled point pattern matching by Delaunay triangulation and maximal cliques, *Pattern Recognition*, Vol. 19: 35–40, 1986.

[248] S. Olariu, Weak bipolarizable graphs, *Discr. Math.*, Vol. 74: 159–171, 1989.

[249] R.E. Osteen, Clique detection algorithms based on line addition and line removal, *SIAM J. Appl. Math.*, Vol. 26: 126–135, 1974.

[250] R.E. Osteen and J.T. Tou, A clique-detection algorithm based on neighborhoods in graphs, *Int. J. Comput. Inform. Sci.*, Vol. 2: 257–268, 1973.

[251] Q. Ouyang, P.D. Kaplan, S. Liu and A. Libchaber, DNA solutions of the maximal clique problem, *Science*, Vol. 278: 446–449, 1997.

[252] M.W. Padberg, Covering, packing and knapsack problems, *Ann. Discr. Math.*, Vol. 4: 265–287, 1979.

[253] A. Panconesi and D. Ranjan, Quantifiers and approximation, *Proc. 22nd ACM Ann. Symp. Theory of Comput.*: 446–456, 1990.

[254] A. Panconesi and D. Ranjan, Quantifiers and approximation, *Theor. Comput. Sci.*, Vol. 107: 145–163, 1993.

[255] C. Papadimitriou, *Computational Complexity*, Addison-Wesley, Amsterdam, 1994.

[256] C. Papadimitriou and M. Yannakakis, The clique problem for planar graphs, *Inform. Proc. Lett.*, Vol. 13: 131–133, 1981.

[257] C. Papadimitriou and M. Yannakakis, Optimization, approximation and complexity classes, *Proc. 20th Ann. ACM STOC*: 229–234, 1988.

[258] C. Papadimitriou and M. Yannakakis, Optimization, approximation and complexity classes, *J. Comput. Syst. Sci.*, Vol. 43: 425–440, 1991.

[259] P.M. Pardalos, Continuous approaches to discrete optimization problems, in *Nonlinear Optimization and Applications*, G. Di Pillo and F. Giannessi, eds., Plenum Press, New York: 313–328, 1996.

[260] P.M. Pardalos and N. Desai, An algorithm for finding a maximum weighted independent set in an arbitrary graph, *Int. J. Comput. Math.*, Vol. 38: 163–175, 1991.

[261] P.M. Pardalos and A.T. Phillips, A global optimization approach for solving the maximum clique problem, *Int. J. Comput. Math.*, Vol. 33: 209–216, 1990.

[262] P.M. Pardalos, J. Rappe and M.G.C. Resende, An Exact Parallel Algorithm for the Maximum Clique Problem, in *High Performance Algorithms and Software in Nonlinear Optimization*, R. De Leone, A. Murlí, P.M. Pardalos and G. Toraldo (eds.), Kluwer: 279–300, Dordrecht, 1998.

[263] P. M. Pardalos and G. Rodgers, Parallel branch and bound algorithms for quadratic zero-one programming on a hypercube architecture, *Ann. Oper. Res.*, Vol. 22: 271–292, 1990.

[264] P. M. Pardalos and G. Rodgers, Computational aspects of a branch and bound algorithm for quadratic zero-one programming, *Computing*, Vol. 45: 131–144, 1990.

[265] P.M. Pardalos and G.P. Rodgers, A branch and bound algorithm for the maximum clique problem, *Comput. Oper. Res.*, Vol. 19: 363–375, 1992.

[266] P.M. Pardalos and J. Xue, The maximum clique problem, *J. Global Optim.*, Vol. 4: 301–328, 1994.

[267] K. Park and B. Carter, On the effectiveness of genetic search in combinatorial optimization, *Technical Report BU-CS-94-010*, Computer Science Dept., Boston University, 1994. (a shorter version appears in: *Proc. 10th ACM Symp. Appl. Comput.*, 1995.)

[268] M.C. Paull and S.H. Unger, Minimizing the number of states in incompletely specified sequential switching functions, *IRE Trans. Electr. Comput.*, Vol. EC-8: 356–367, 1959.

[269] E.R. Peay, Jr., An iterative clique detection procedure, Michigan Math. Psychol. Program: MMPP 70-4, 1970.

[270] E.R. Peay, Jr., Hierarchical clique structures, *Sociometry*, Vol. 37: 54–65, 1974.

[271] M. Pelillo, Relaxation labeling networks for the maximum clique problem, *J. Artif. Neural Networks*, Vol. 2: 313–328, 1995.

[272] M. Pelillo, A unifying framework for relational structure matching, in: *Proc. 14th Int. Conf. Pattern Recognition*: 1316–1319, 1998.

[273] M. Pelillo, Replicator equations, maximal cliques, and graph isomorphism, *Neural Computation*, to appear, 1999.

[274] M. Pelillo and A. Jagota, Feasible and infeasible maxima in a quadratic program for maximum clique, *J. Artif. Neural Networks*, Vol. 2: 411–420, 1995.

[275] M. Pelillo, K. Siddiqi and S.W. Zucker, Matching hierarchical structures using association graphs, in H. Burkhardt and B. Neumann (eds.), *Computer Vision—ECCV'98* (Lecture Notes in Computer Science, Vol. 1407), Springer, Berlin: 3–16, 1998.

[276] O. Perron, Über lückenlose Ausfüllung des n-dimensionalen Raumes durch kongruente Würfel (in German), *Math. Z.*, Vol. 46: 1–26, 161–180. Zbl 22, 202, 1940.

[277] C. Peterson and B. Söderberg, Artificial neural networks, in [2]: 173–213, 1997.

[278] J.C. Picard and M. Queyranne, On the integer-valued variables in the linear vertex packing problem, *Math. Programming*, Vol. 12: 97–101, 1977.

[279] F. Pla and J.A. Marchant, Matching feature points in image sequences through a region-based method, *Comput. Vision Image Understanding*, Vol. 66: 271–285, 1997.

[280] F.P. Preparata, G. Metze and R.T. Chien, On the connection assignment problem of diagnosable systems, *IEEE Trans. Electr. Comput.*, Vol. 16: 848–854, 1967.

[281] W.R. Pulleyblank, Minimum node covers and 2-bicritical graphs, *Math. Programming*, Vol. 17: 91–103, 1979.

[282] B. Radig, Image sequence analysis using relational structures, *Pattern Recognition*, Vol. 17: 161–167, 1984.

[283] P. Raghavan, Probabilistic construction of deterministic algorithms: Approximating packing integer programs, *J. Comput. Syst. Sci.*, Vol. 37: 130–143, 1988.

[284] J. Ramanujam and P. Sadayappan, Optimization by neural networks, *Proc. IEEE Int. Conf. Neural Networks*: 325–332, 1988.

[285] M.C.G. Resende, T.A. Feo, and S.H. Smith, Fortran subroutines for approximate solution of maximum independent set problems using GRASP, *ACM Transactions on Mathematical Software*, to appear 1999.

[286] J.M. Robson, Algorithms for maximum independent sets, *J. Algorithms*, Vol. 7: 425–440, 1986.

[287] D.J. Rose, Triangulated graphs and the elimination process, *J. Math. Anal. Appl.*, Vol. 32: 597–609, 1970.

[288] D.J. Rose, R.E. Tarjan and G.S. Leuker, Algorithmic aspects of vertex elimination on graphs, *SIAM J. Comput.*, Vol. 5: 266–283, 1976.

[289] D. Rotem and J. Urrutia, Finding maximum cliques in circle graphs, *Networks*, Vol. 11: 269–278, 1981.

[290] B.E. Sagan, A note on independent sets in trees, *SIAM J. Discr. Math.*, Vol. 1: 105–108, 1988.

[291] L. Sanchis and A. Jagota, Some experimental and theoretical results on test case generators for the maximum clique problem, *INFORMS J. Comput.*, Vol. 8: 87–102, 1996.

[292] C.E. Shannon, Certain results in coding theory for noisy channels, *Inform. and Control*, Vol. 1: 6–25, 1957.

[293] N. Z. Shor, Dual quadratic estimates in polynomial and Boolean programming, in *Computational Methods in Global Optimization*, P. M. Pardalos and J. B. Rosen (eds.), *Ann. Oper. Res.*, Vol. 25: 163–168, 1990.

[294] Y. Shrivastava, S. Dasgupta, and S.M. Reddy, Neural network solutions to a graph theoretic problem, in *Proc. IEEE Int. Symp. Circuits Syst.*: 2528–2531, 1990.

[295] Y. Shrivastava, S. Dasgupta, and S.M. Reddy, Guaranteed convergence in a class of Hopfield networks, *IEEE Trans. Neural Networks*, Vol. 3: 951–961, 1992.

[296] N.J.A. Sloane, Unsolved problems in graph theory arising from the study of codes, *J. Graph Theory Notes of New York*, Vol. 18: 11–20, 1989.

[297] P. Soriano and M. Gendreau, Tabu search algorithms for the maximum clique problem, in [189]: 221–242, 1996.

[298] P. Soriano and M. Gendreau, Diversification strategies in tabu search algorithms for the maximum clique problem, *Ann. Oper. Res.*, Vol. 63: 189–207, 1996.

[299] V.T. Sós and E.G. Straus, Extremal of functions on graphs with applications to graphs and hypergraphs, *J. Combin. Theory B*, Vol. 32: 246–257, 1982

[300] S.K. Stein, Algebraic tiling, *Amer. Math. Monthly*, Vol. 81: 445–462, 1974.

[301] S. Szabó, A reduction of Keller's conjecture, *Periodica Math. Hung.*, Vol. 17: 265–277, 1986.

[302] Y. Takefuji, *Neural Network Parallel Computing*, Kluwer, Dordrecht, 1992.

[303] Y. Takefuji, L. Chen, K. Lee and J. Huffman, Parallel algorithms for finding a near-maximum independent set of a circle graph, *IEEE Trans. Neural Networks*, Vol. 1: 263-267, 1990.

[304] R.E. Tarjan, Finding a maximum clique, *Technical Report 72-123*, Computer Sci. Dept., Cornell University, Ithaca, NY, 1972.

[305] R.E. Tarjan and A.E. Trojanowski, Finding a maximum independent set, *SIAM J. Comput.*, Vol. 6: 537–546, 1977.

[306] R.E. Tarjan and M. Yannakakis, Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs, *SIAM J. Comput.*, Vol. 13: 566–579, 1984.

[307] E. Tomita, Y. Kohata and H. Takahashi, A simple algorithm for finding a maximum clique, *Technical Report UEC-TR-C5*, 1988.

[308] E. Tomita, S. Mitsuma and H. Takahashi, Two algorithms for finding a near-maximum clique, *Technical Report UEC-TR-C1*, 1988.

[309] E. Tomita, A. Tanaka and H. Takahashi, The worst-case time complexity for finding all the cliques, *Technical Report UEC-TR-C5*, 1988.

[310] S. Tsukiyama, M. Ide, H. Aviyoshi and I. Shirakawa, A new algorithm for generating all the maximum independent sets, *SIAM J. Comput.*, Vol. 6: 505–517, 1977.

[311] P. Turán, On an extremal problem in graph theory (in Hungarian), *Mat. és Fiz. Lapok*, Vol. 48: 436–452, 1941.

[312] L.G. Valiant, The complexity of enumeration and reliability problems, *SIAM J. Comput.*, Vol. 8: 410–421, 1979.

[313] O. Verbitsky, On the hardness of approximating some optimization problems that are supposedly easier that MAX CLIQUE, *Comb. Probab. Comput.* Vol. 4: 167–180, 1995.

[314] M. Vingron and P. Argos, Motif recognition and alignment for many sequences by comparison of dot-matrices, *J. Molecular Biol.*, Vol. 218: 33–43, 1991.

[315] M. Vingron and P.A. Pevzner, Multiple sequence comparison and consistency on multipartite graphs. *Adv. Appl. Math* Vol.16: 1–22, 1995.

[316] H.S. Wilf, The eigenvalues of a graph and its chromatic number, *J. London Math. Soc.*, Vol. 42: 330–332, 1967.

[317] H.S. Wilf, *Algorithms and Complexity*, Prentice-Hall, Englewood Cliffs, NJ, 1986.

[318] H.S. Wilf, Spectral bounds for the clique and independence numbers of graphs, *J. Combin. Theory B*, Vol. 40: 113–117, 1986.

[319] S. Wimer, R.Y. Pinter and J. Feldman, Optimal chaining of CMOS transistors in a functional cell, *IEEE Trans. Computer-Aided Design*, Vol. 6: 795–801, 1987.

[320] J. Wu, T. Harada and T. Fukao, New method to the solution of maximum clique problem: Mean-field approximation algorithm and its experimentation. *Proc. IEEE Int. Conf. Syst., Man, Cybern.*: 2248–2253, 1994.

[321] J. Xue, Fast Algorithms for Vertex Packing and Related Problems, *Ph.D Thesis*, GSIA, Carnegie Mellon University, 1991.

[322] J. Xue, Edge-maximal triangulated subgraphs and heuristics for the maximum clique problem, *Networks*, Vol. 24: 109–120, 1994.

[323] M. Yannakakis, On the approximation of maximum satisfiability, *Proc. 3rd Annual ACM-SIAM Symp. Discr. Algorithms*, 1992.

[324] M.-S. Yu, L. Yu and S.-J. Chang, Sequential and parallel algorithms for the maximum-weight independent set problem on permutation graphs, *Inform. Proc. Lett.*, Vol. 46: 7–11, 1993.

[325] B.-T. Zhang and S.-Y. Shin, Code optimization for DNA computing of maximal cliques, in *Advances in Soft Computing—Engineering and Design*, Springer, Berlin, 1998.