# HEURISTICS FOR MAXIMUM CLIQUE AND INDEPENDENT SET

**Introduction**. Throughout this article, $G = (V, E)$ is an arbitrary undirected and weighted graph unless otherwise specified, where $V = \{1, 2, \ldots, n\}$ is the vertex set of $G$ and $E \subseteq V \times V$ is its edge set. For each vertex $i \in V$, a positive weight $w_i$ is associated with $i$, collected in the *weight vector* $w \in \mathbb{R}^n$. For a subset $S \subseteq V$, the weight of $S$ is defined as $W(S) = \sum_{i \in S} w_i$, and $G(S) = (S, E \cap S \times S)$ is the *the subgraph induced by* $S$. The *cardinality* of $S$, i.e., the number of its vertices, will be denoted 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 the cardinality of the maximum clique. The maximum clique problem asks for cliques of maximum cardinality. 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 $\omega(G, w)$.

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

An *independent set* (also called *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.

The *complement graph of* $G = (V, E)$ is the graph $\overline{G} = (V, \overline{E})$, where $\overline{E} = \{(i, j) \mid i, j \in V,\ i \neq j \text{ and } (i, j) \notin E\}$. It is easy to see that $S$ is a clique of $G$ if and only if $S$ is an independent set of $\overline{G}$. Any result or algorithm obtained for one of the two problems has its equivalent forms for the other one. Hence $\alpha(G) = \omega(\overline{G})$, and, more generally, $\alpha(G, w) = \omega(\overline{G}, w)$.

The maximum clique and independent set problems are well-known examples of intractable combinatorial optimization problems [18]. Apart from the theoretical interest around these problems, they also find practical applications in such diverse domains as computer vision, experimental design, information retrieval, fault tolerance, etc. Moreover, many important problems turn out to be easily reducible to them, and these include, for example, the Boolean satisfiability problem, the subgraph isomorphism problem, and the vertex covering problem. The maximum clique problem has also a certain historical value, as it was one of the first problems shown to be $NP$-complete in Karp's now classical paper on computational complexity [64].

Due to their inherent computational complexity, exact algorithms are guaranteed to return a solution only in a time which increases exponentially with the number of vertices in the graph, and this makes them inapplicable even to moderately large problem instances. Moreover, a series of recent theoretical results show that the problems are in fact difficult to solve even in terms of approximation. Strong evidence of this fact came in 1991, when Feige *et al.* [32] proved that if there is a polynomial-time algorithm that approximates the maximum clique within a factor of $2^{\log^{1-\epsilon} n}$, then any $NP$ problem can be solved in "quasi-polynomial" time (i.e., in $2^{\log^{O(1)} n}$ time). The result was further refined by Arora *et al.* [7, 6] one year later. Specifically, they proved that there exists an $\epsilon > 0$ such that no polynomial-time algorithm can approximate the size of the maximum clique within a factor of $n^\epsilon$, unless $P = NP$. More recent developments along these lines can be found in [14, 15, 49].

In light of these negative results, much effort has recently been directed towards devising efficient heuristics for maximum clique and independent set, for which no formal guarantee of performance may be provided, but are anyway of interest in practical application. 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 [63].

In this article we provide an informal survey of recent heuristics for maximum clique and related problems, and up-to-date bibliographic pointers to the relevant literature. A more comprehensive review and bibliography can be found in [18].

**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. Decisions on which vertex to be added in or moved out next are based on certain indicators associated with candidate vertices as, for example, the vertex degree. There is also a distinction between heuristics that update the indicators every time a vertex is added in or moved out, and those that do not. Examples of such heuristics can be found in [62, 89]. The differences among these heuristics are their choice of indicators and how indicators are updated. A heuristic of this type can run very fast.

**Local search heuristics**. Let us define $\mathcal{C}_G$ to be the set of all the maximal cliques of $G$. Basically, a sequential greedy heuristic finds one set in $\mathcal{C}_G$, hoping it is (close to) the optimal set, and stops. A possible way to improve our approximation solutions is to expand the search in $\mathcal{C}_G$. For example, once we find a set $S \in \mathcal{C}_G$, we can search its "neighbors" to improve $S$. This leads to the class of the *local search heuristics* [2]. Depending on the neighborhood structure 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 set $C \in \mathcal{C}_G$ 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 $S \in \mathcal{C}_G$, then it searches all the $k$-neighbors of $S$ and returns the best clique found. Clearly, 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 $O(n^k)$.

A class of heuristics designed to search various sets of $\mathcal{C}_G$ is called the *randomized heuristics*. The main ingredient of this class of heuristics is the part that finds a random set in $\mathcal{C}_G$. A possible way to do that is to include some random factors in the generation of a set of $\mathcal{C}_G$. A randomized heuristic runs a heuristic (with random factors included) a number of times to find different sets over $\mathcal{C}_G$. 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.* [33] 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. A different implementation of a randomized algorithm for the maximum independent set problem can be found in [5].

**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.

*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 [65], 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 $\Delta f$ is the difference of the cost function between the new and the current state, and $\tau$ 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 logarithmic 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 [90] [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 $\lambda$ is a weighting factor exceeding 1.

Jerrum [61] 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 [56], Homer and Peinado compare the performance of three heuristics, namely the greedy heuristic developed by Johnson [62], a randomized version of Boppana and Halldórsson's subgraph-exclusion algorithm [24], 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 [61]. 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 [63].

*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 [50, 52] for an introduction).

In the mid-1980's Hopfield and Tank [57] showed that certain feedback continuous neural models are capable of finding approximate solutions to difficult optimization problems such as the traveling salesman problem [57]. This application was motivated by the property that the temporal evolution of these models is governed by a quadratic Liapunov 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 [68, 82] and the textbook of Takefuji [88] for surveys of this field. 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.* [12], Godbeer *et al.* [44], Ramanujam and Sadayappan [83], Aarts and Korst [1], and Shrivastava *et al.* [84] (see also [85]). However, little or no experimental results were presented, thereby making it difficult to evaluate the merits of these algorithms. In [67], Lin and Lee used the quadratic zero-one formulation from [77] 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 [26], while obtaining slightly worse results in terms of clique size.

Grossman [47] 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 [58] 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 [59]. 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.* [60] 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.

Pelillo [79] takes a completely different approach to the problem, by exploiting a continuous formulation of maximum clique and the dynamical properties of the so-called relaxation labeling networks. His algorithm is described in the next section.

*Genetic algorithms.* Genetic algorithms are parallel search procedures inspired from the mechanisms of evolution in natural systems [55, 45]. 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 [45].

One of the earliest attempts to solve the maximum clique problem using genetic algorithms was done in 1993 by Carter and Park [27]. 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 [78], genetic algorithms were proven to be less effective than simulated annealing. At almost the same time Bäck and Khuri [8], 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.* [73] 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 [25] 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 [39]

Instead of using the standard binary representation for chromosomes, Foster and Soule [36] employed an integer-based encoding scheme. Moreover, they used a time weighting fitness function similar in spirit to those of Carter and Park [27]. The results obtained are interesting, but still not comparable to those obtained using more traditional search heuristics.

Fleurent and Ferland [35] 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 and other local search techniques as alternative mutation operators. The results presented are encouraging, but running time is quite high.

In [53], 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 [63] are reported to validate this approach.

Finally, Marchiori [70] 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.

*Tabu search.* Tabu search, introduced independently by Glover [41], [42] and Hansen and Jaumard [48], 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 [43] and [51].

In 1989, Friden *et al.* [37] 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.

In [38, 86], 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. It differs from it because it performs a random sampling of the neighborhood, and also because it allows for multiple deletion of vertices in the current solution. Here no diversification strategy was used. In [38, 86] results on randomly generated graphs were presented and the algorithms were shown to be very effective. More recently, Soriano and Gendreau [87] tested their algorithms over the DIMACS benchmark graphs and the results confirmed the early conclusions.

Recently, Battiti and Protasi [13] 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 vertices 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 vertices of the complement graph. They tested their algorithm over many DIMACS benchmark graphs obtaining better results then those presented at the DIMACS workshop in competitive time.

**Continuous based heuristics**. In 1965, Motzkin and Straus [72] established a remarkable connection between the maximum clique problem and a certain quadratic programming problem. Let $G = (V, E)$ be an undirected (unweighted) graph and let $\Delta$ denote 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 the letter $e$ is reserved for a vector of appropriate length, consisting of unit entries (hence $e^T x = \sum_{i \in V} x_i$).

Now, consider the following quadratic function, sometimes called the *Lagrangian* of $G$:

$$g(x) = x^T A_G x \qquad (1)$$

where $A_G = (a_{ij})$ is the adjacency matrix of $G$, i.e. the symmetric $n \times n$ matrix where $a_{ij} = 1$ if $(i, j) \in E$ and $a_{ij} = 0$ if $(i, j) \notin E$, and let $x^*$ be a global maximizer of $g$ on $\Delta$. In [72] it is proved that the clique number of $G$ is related to $g(x^*)$ by the following formula:

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

Additionally, it is shown that a subset of vertices $S$ is a maximum clique of $G$ if and only if its *characteristic* vector $x^S$, which is the vector of $\Delta$ defined as $x_i^S = 1/|S|$ if $i \in S$ and $x_i^S = 0$ otherwise, is a global maximizer of $g$ on $\Delta$. In [81, 40], the Motzkin-Straus theorem has been extended by providing a characterization of *maximal* cliques in terms of *local* maximizers of $g$ on $\Delta$.

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 [76, 81]. In principle, spurious solutions represent a problem since, while providing information about the cardinality of the maximum clique, they do not allow us to easily extract its vertices.

In the past few years, there has been much interest around the Motzkin-Straus and related continuous formulations of the maximum clique problem. 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 [75], 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 [76], 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 [79] 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 [54, 80]. The model operates in the simplex $\Delta$ and possesses a quadratic Liapunov 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 algorithm 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 [39] have proposed a heuristics which is based on a parameterized 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:

$$S(k) = \left\{ x \in \mathrm{I\!R}^n : \sum_{i=1}^{n} x_i^2 \le \frac{1}{k},\ x_i \ge 0,\ \forall i \right\}$$

where $k$ is a fixed parameter. Let $x^*$ be a global minimizer of $h$ on $S(k)$, and let $V(k) = h(x^*)$. In [39] it is proved that $V(k) = 0$ if and only if there exists an independent set $S$ of $\overline{G}$ with size $|S| \ge k$. 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 $V(k+1) \ne 0$ 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, they developed a heuristic based on the observation that by removing the non-negativity 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 [63] and the results obtained confirmed the effectiveness of the approach.

The spurious solution problem has recently been solved by Bomze [16]. Consider the following regularized version of function $g$:

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

which is obtained from (1) by substituting the adjacency matrix $A_G$ of $G$ with

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

where $I$ is the identity matrix. Unlike the Motzkin-Straus formulation, it can be proved that *all* maximizers of $\hat{g}$ on $\Delta$ 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 $\Delta$ on the one hand and maximum cliques and global maximizers on the other hand. In [16, 20], replicator equations are used in conjunction to this spurious-free formulation to find maximal cliques of $G$. Note that here the vertices 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 [76]. The results obtained over a set of random as well as DIMACS benchmark graphs 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 $\hat{g}$ are quite large; for a thorough empirical analysis see also [23]. 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 [22] indicate this is not the case.

In [19] the properties of the following function are studied

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

It is shown that when $\alpha$ is positive all the properties enjoyed by the standard regularization approach [16] hold true. Specifically, in this case a one-to-one correspondence between local/global maximizers in the continuous space and local/global solutions in the discrete space exists. For negative $\alpha$'s an interesting picture emerges: as the absolute value of $\alpha$ grows larger, local maximizers corresponding to maximal cliques disappear. In [19], bounds on the parameter $\alpha$ are derived which affect the stability of these solutions. These results have suggested an *annealed replication* heuristic, which consists of starting from a large negative $\alpha$ and then properly reducing it during the optimization process. For each value of $\alpha$ standard replicator equations are run in order to obtain local solutions of the corresponding objective function. The rationale behind this idea is that for values of $\alpha$ with a proper large absolute value only local solutions corresponding to large maximal cliques will survive, together with various spurious maximizers. As the value of $\alpha$ is reduced, spurious solutions disappear and smaller maximal cliques become stable. An annealing schedule is proposed which is based on the assumption that the graphs being considered are random. In this respect, the proposed procedure differs from usual simulated annealing approaches. which mostly use a "black-box" cooling schedule Experiments conducted over several DIMACS benchmark graphs confirm the effectiveness of the proposed approach and the robustness of the annealing strategy. The overall conclusion is that the annealing procedure *does* help to avoid inefficient local solutions, by initially driving the dynamics towards promising regions in state space, and then refining the search as the annealing parameter is increased.

Recently, the Motzkin-Straus theorem has been generalized to the weighted case [40]. Note that the Motzkin-Straus program can be reformulated as a minimization problem by considering the function

$$f(x) = x^T(I + A_{\overline{G}})x \tag{3}$$

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 $\Delta$, then we have: $\omega(G) = 1/f(x^*)$. This is simply a different formulation of the Motzkin-Straus theorem. Given a weighted graph $G = (V, E)$ with weight vector $w$, let $\mathcal{M}(G, w)$ be the class of symmetric $n \times n$ matrices $B = (b_{ij})_{i,j \in V}$ defined as $2b_{ij} \geq b_{ii} + b_{jj}$ if $(i, j) \notin E$ and $b_{ij} = 0$ otherwise, and $b_{ii} = \frac{1}{w_i}$ for all $i \in V$.

Given the following quadratic program, which is in general indefinite,

$$\begin{array}{ll} \text{minimize} & f(x) = x^T B x \\ \text{subject to} & x \in \Delta \end{array} \tag{4}$$

in [40] it is shown that for any $B \in \mathcal{M}(G, w)$ we have:

$$\omega(G, w) = \frac{1}{f(x^*)} \tag{5}$$

where $x^*$ is a global minimizer of program (4). Furthermore, denote by $x^S$ the *weighted characteristic vector* of $S$, which is a vector with coordinates $x_i^S = w_i/W(S)$ if $i \in S$ and $x_i^S = 0$ otherwise. 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 characteristic vector $x^S$ is a global minimizer of (4). 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. In [21] these spurious solutions are characterized and a regularized version which avoids this kind of problems is introduced, exactly as in the unweighted case (see also [17]). Replicator equations are then used to find maximal weight cliques in weighted graphs, using this formulation. Experiments with this approach on both random graphs and DIMACS graphs are reported. The results obtained are compared with those produced by a very efficient maximum weight clique algorithm of the branch-and-bound variety. The algorithm performed remarkably well especially on large and dense graphs, and it was typically an order of magnitude more efficient than its competitor.

Finally, we mention the recent work by Massaro and Pelillo [71], who transformed the Motzkin-Straus program into a linear complementarity problem [31], and then solved it using Lemke's well-known algorithm [66]. The preliminary results obtained over many weighted and unweighted DIMACS graphs show that this approach substantially outperforms all other continuous based heuristics.

**Miscellaneous**. Another type of heuristics that finds a maximal clique of $G$ is called the *subgraph approach* (see [11]). 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 $G'$ ($\mathcal{C}_{G'} \subseteq \mathcal{C}_G$). Because of the special structure of $G'$, this implicit search can be done efficiently. In Balas and Yu [11], $G'$ 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 [9], [91], and [92]. Some of the greedy heuristics, randomized heuristics and subgraph approach heuristics are compared in [91] and [92] on randomly generated weighted and unweighted graphs.

Various new heuristics were presented at the 1993 DIMACS challenge devoted to clique, coloring and satisfiability [63]. In particular, Balas and Niehaus [10] 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 [46] used restricted backtracking to provide a tradeoff between the size of the clique and the completeness of the search. Mannino and Sassano [69] 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 [11], 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 [4] has also emerged as a potential technique for the maximum clique problem [74, 93]. 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 [28, 29, 30, 34].

**Conclusions**. Over the past few years, research on the maximum clique and related problems has yielded many interesting heuristics. This article has provided an expository survey on these algorithms and an up-to-date bibliography. However, the activity in this field is so extensive that a survey of this nature is outdated before it is written.

## References


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

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

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

[4] Adleman, L. M.: 'Molecular computation of solutions to combinatorial optimization', *Science* **266** (1994), 1021–1024.

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

[6] Arora, S., Lund, C., Motwani, R., Sudan, M., and Szegedy, M.: 'Proof verification and the hardness of approximation problems': *Proc. 33rd Ann. Symp. Found. Comput. Sci.*, Pittsburgh, PA, 1992, pp. 14–23.

[7] Arora, S., and Safra, S.: 'Probabilistic checking of proofs: A new characterization of NP': *Proc. 33rd Ann. Symp. Found. Comput. Sci.*, Pittsburgh, PA, 1992, pp. 2–13.

[8] Bäck, T., and Khuri, S.: 'An evolutionary heuristic for the maximum independent set problem': *Proc. 1st IEEE Conf. Evolutionary Comput.*, 1994, pp. 531–535.

[9] BALAS, E.: 'A fast algorithm for finding an edge-maximal subgraph with a TR-formative coloring', *Discr. Appl. Math.* **15** (1986), 123–134.

[10] BALAS, E., AND NIEHAUS, W.: 'Finding large cliques in arbitrary graphs by bipartite matching', In Johnson and Trick [63], pp. 29–51.

[11] BALAS, E., AND YU, C.S.: 'Finding a maximum clique in an arbitrary graph', *SIAM J. Comput.* **14** (1986), 1054–1068.

[12] BALLARD, D. H., GARDNER, P. C., AND SRINIVAS, M. A.: Graph problems and connectionist architectures, Tech. Rep. TR 167, Dept. Computer Science, University of Rochester, 1987.

[13] BATTITI, R., AND PROTASI, M.: Reactive local search for the maximum clique problem, Tech. Rep. TR-95-052, International Computer Science Institute, Berkeley, CA, 1995, to appear in *Algorithmica*.

[14] BELLARE, M., GOLDWASSER, S., LUND, C., AND RUSSELL, A.: 'Efficient probabilistically checkable proofs and application to approximation': *Proc. 25th Ann. ACM Symp. Theory Comput.*, 1993, pp. 294–304.

[15] BELLARE, M., GOLDWASSER, S., AND SUDAN, M.: 'Free bits, PCPs and non-approximability—Towards tight results': *Proc. 36th Ann. Symp. Found. Comput. Sci.*, 1995, pp. 422–431.

[16] BOMZE, I. M.: 'Evolution towards the maximum clique', *J. Glob. Optim.* **10** (1997), 143–164.

[17] BOMZE, I. M.: 'On standard quadratic optimization problems', *J. Glob. Optim.* **13** (1998), 369–387.

[18] BOMZE, I. M., BUDINICH, M., PARDALOS, P. M., AND PELILLO, M.: 'The maximum clique problem', *Handbook of Combinatorial Optimization (Supplement Volume A)*, in D.-Z. DU AND P. M. PARDALOS (eds.). Kluwer Academic Publishers, Boston, MA, 1999.

[19] BOMZE, I. M., BUDINICH, M., PELILLO, M., AND ROSSI, C.: 'Annealed replication: A new heuristic for the maximum clique problem', *Discr. Appl. Math.* (2000), in press.

[20] BOMZE, I. M., PELILLO, M., AND GIACOMINI, R.: 'Evolutionary approach to the maximum clique problem: Empirical evidence on a larger scale', in I. M. BOMZE, T. CSENDES, R. HORST, AND P. M. PARDALOS (eds.): *Developments in Global Optimization*, Kluwer Academic Publishers, 1997, pp. 95–108.

[21] BOMZE, I. M., PELILLO, M., AND STIX, V.: Approximating the maximum weight clique using replicator dynamics, Tech. Rep. CS-99-13, Dipartimento di Informatica, Università Ca' Foscari di Venezia, 1999.

[22] BOMZE, I. M., AND RENDL, F.: 'Replicator dynamics for evolution towards the maximum clique: Variations and experiments', in R. DE LEONE, A. MURLÍ, P. M. PARDALOS, AND G. TORALDO (eds.): *High Performance Algorithms and Software in Nonlinear Optimization*, Kluwer Academic Publishers, 1998, pp. 53–67.

[23] BOMZE, I. M., AND STIX, V.: 'Genetic engineering via negative fitness: Evolutionary dynamics for global optimization', *Ann. Oper. Res.* **90** (1999), in press.

[24] BOPPANA, R., AND HALLDÓRSSON, M. M.: 'Approximating maximum independent sets by excluding subgraphs', *BIT* **32** (1992), 180–196.

[25] BUI, T. N., AND EPPLEY, P. H.: 'A hybrid genetic algorithm for the maximum clique problem': *Proc. 6th Int. Conf. Genetic Algorithms*, 1995, pp. 478–484.

[26] CARRAGHAN, R., AND PARDALOS, P.M.: 'An exact algorithm for the maximum clique problem', *Oper. Res. Lett.* **9** (1990), 375–382.

[27] CARTER, B., AND PARK, K.: How good are genetic algorithms at finding large cliques: An experimental study, Tech. Rep. BU-CS-93-015, Computer Science Dept., Boston University, 1993.

[28] CHIBA, N., NISHIZEKI, T., AND SAITO, N.: 'An algorithm for finding a large independent set in planar graphs', *Networks* **13** (1983), 247–252.

[29] CHROBAK, M., AND NAOR, J.: 'An efficient parallel algorithm for computing a large independent set in a planar graph', *Algorithmica* **6** (1991), 801–815.

[30] CHVÁTAL, V.: 'A greedy heuristic for the set-covering problem', *Math. Oper. Res.* **4** (1979), 233–23.

[31] COTTLE, R. W., PANG, J., AND STONE, R. E.: *The Linear Complementarity Problem*, Academic Press, Boston, MA, 1992.

[32] FEIGE, U., GOLDWASSER, S., LOVÁSZ, L., SAFRA, S., AND SZEGEDY, M.: 'Approximating clique is almost NP-complete': *Proc. 32nd Ann. Symp. Found. Comput. Sci.*, San Juan, Puerto Rico, 1991, pp. 2–12.

[33] FEO, T. A., RESENDE, M. G. C., AND SMITH, S. H.: 'A greedy randomized adaptive search procedure for maximum independent set', *Oper. Res.* **42** (1994), 860–878.

[34] FISHER, M. L., AND WOLSEY, L. A.: 'On the greedy heuristic for continuous covering and packing problems', *SIAM J. Alg. Discr. Meth.* **3** (1982), 584–591.

[35] FLEURENT, C., AND FERLAND, J. A.: 'Object-oriented implementation of heuristic search methods for graph coloring, maximum clique, and satisfiability', In Johnson and Trick [63], pp. 619–652.

[36] FOSTER, J. A., AND SOULE, T.: Using genetic algorithms to find maximum cliques, Tech. Rep. LAL 95-12, Dept. of Computer Science, U. Idaho, 1995.

[37] FRIDEN, C., HERTZ, A., AND WERRA, M. DE: 'STABULUS: A technique for finding stable sets in large graphs with tabu search', *Computing* **42** (1989), 35–44.

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

[39] Gibbons, L. E., Hearn, D. W., and Pardalos, P. M.: 'A continuous based heuristic for the maximum clique problem', In Johnson and Trick [63], pp. 103–124.

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

[41] Glover, F.: 'Tabu search—Part I', *ORSA J. Comput.* **1** (1989), 190–260.

[42] Glover, F.: 'Tabu search—Part II', *ORSA J. Comput.* **2** (1990), 4–32.

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

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

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

[46] Goldberg, M. K., and Rivenburgh, R. D.: 'Constructing cliques using restricted backtracking', In Johnson and Trick [63], pp. 89–101.

[47] Grossman, T.: 'Applying the INN model to the max clique problem', In Johnson and Trick [63], pp. 125–146.

[48] Hansen, P., and Jaumard, B.: 'Algorithms for the maximum satisfiability problem', *Computing* **44** (1990), 279–303.

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

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

[51] Hertz, A., Taillard, E., and Werra, D. de: 'Tabu search', In Aarts and Lenstra [2], pp. 121–136.

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

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

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

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

[56] Homer, S., and Peinado, M.: 'Experiments with polynomial-time CLIQUE approximation algorithms on very large graphs', In Johnson and Trick [63], pp. 147–167.

[57] Hopfield, J. J., and Tank, D. W.: 'Neural computation of decisions in optimization problems', *Biol. Cybern.* **52** (1985), 141–152.

[58] Jagota, A.: 'Approximating maximum clique with a Hopfield neural network', *IEEE Trans. Neural Networks* **6** (1995), 724–735.

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

[60] Jagota, A., Sanchis, L., and Ganesan, R.: 'Approximately solving maximum clique using neural networks and related heuristics', In Johnson and Trick [63], pp. 169–204.

[61] Jerrum, M.: 'Large cliques elude the Metropolis process', *Random Structures and Algorithms* **3** (1992), 347–359.

[62] Johnson, D. S.: 'Approximation algorithms for combinatorial problems', *J. Comput. Syst. Sci.* **9** (1974), 256–278.

[63] Johnson, D. S., and Trick, M. A. (eds.): *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge*, Vol. DIMACS 26, American Mathematical Society, 1996.

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

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

[66] Lemke, C. E.: 'Bimatrix equilibrium points and mathematical programming', *Managem. Sci.* **11** (1965), 681–689.

[67] Lin, F., and Lee, K.: 'A parallel computation network for the maximum clique problem': *Proc. 1st Int. Conf. Fuzzy Theory Tech.*, 1992.

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

[69] Mannino, C., and Sassano, A.: 'Edge projection and the maximum cardinality stable set problem', In Johnson and Trick [63], pp. 205–219.

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

[71] Massaro, A., and Pelillo, M.: A complementary pivoting approach to the maximum clique problem, Manuscript in preparation, 1999.

[72] Motzkin, T. S., and Straus, E. G.: 'Maxima for graphs and a new proof of a theorem of Turán', *Canad. J. Math.* **17** (1965), 533–540.

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

[74] Ouyang, Q., Kaplan, P. D., Liu, S., and Libchaber, A.: 'DNA solutions of the maximal clique problem', *Science* **278** (1997), 1446–449.

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

[76] Pardalos, P. M., and Phillips, A. T.: 'A global optimization approach for solving the maximum clique problem', *Int. J. Comput. Math.* **33** (1990), 209–216.

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

[78] Park, K., and Carter, B.: On the effectiveness of genetic search in combinatorial optimization, Tech. Rep. BU-CS-9-010, Computer Science Dept., Boston University, 1994.

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

[80] Pelillo, M.: 'Replicator dynamics in combinatorial optimization', *Encyclopedia of Optimization*, in P. M. Pardalos and C. A. Floudas (eds.). Kluwer Academic Publishers,, Boston, MA, 1999.

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

[82] Peterson, C., and Söderberg, B.: 'Artificial neural networks', In Aarts and Lenstra [2], pp. 173–213.

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

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

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

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

[87] Soriano, P., and Gendreau, M.: 'Tabu search algorithms for the maximum clique problem', In Johnson and Trick [63], pp. 221–242.

[88] Takefuji, Y.: *Neural Network Parallel Computing*, Kluwer Academic Publishers, Dordrecht, 1992.

[89] Tomita, E., Mitsuma, S., and Takahashi, H.: Two algorithms for finding a near-maximum clique, Tech. Rep. UEC-TR-C1, 1988.

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

[91] Xue, J.: *Fast Algorithms for Vertex Packing and Related Problems*, PhD thesis, GSIA, Carnegie Mellon University, 1991.

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

[93] Zhang, B.-T., and Shin, S.-Y.: 'Code optimization for DNA computing of maximal cliques': *Advances in Soft Computing—Engineering and Design*, Springer-Verlag, 1998.

*Marcello Pelillo*
Università Ca' Foscari di Venezia
Italy
*E-mail address*: `pelillo@dsi.unive.it`

*AMS 1991 Subject Classification*: 90C59, 05C69, 05C85, 68W01.

*Key words and phrases:* Heuristics, algorithms, cliques, independent sets.