

Swap Strategies for Graph Matching *

Paolo Fosser⁺, Roland Glantz⁺, Marco Locatelli[#], and Marcello Pelillo⁺

⁺ Dipartimento di Informatica

Università Ca' Foscari di Venezia

Via Torino 155, 30172 Mestre (VE), Italy

e-mail:{pfosser,glantz,pelillo}@dsi.unive.it

[#] Dipartimento di Informatica

Università di Torino

Corso Svizzera 185, I-10149 Torino, Italy

e-mail:Locatelli@di.unito.it

Abstract

The problem of graph matching is usually approached via explicit search in state-space or via energy minimization. In this paper we deal with a class of heuristics coming from a combination of both approaches. Combinatorially, the basic heuristic of the class can be interpreted as a greedy algorithm to form maximal cliques in an association graph. To avoid one of the main drawbacks of greedy strategies, i.e. that they are easily fooled by poor local optima, we propose a modification which allows for vertex swaps during the formation of a clique. Experiments on random graphs show the effectiveness of the proposed heuristics both in terms of quality of solutions and speed.

1 Introduction

Graph matching is a fundamental problem in computer vision and pattern recognition, and a great deal of effort has been devoted over the past decades to devise efficient and robust algorithms for it (see [7] for an update on recent developments). Basically, two radically distinct approaches have emerged, a distinction which reflects the well-known dichotomy originated in the artificial intelligence field between “symbolic” and “numeric” methods. The first approach views the matching problem as one of explicit search in state-space (see, e.g., [14, 20, 21]). The pioneering work of Ambler *et al.* [1] falls into this class. Their approach is based on the idea that graph matching is equivalent to the problem of finding maximal cliques in the so-called association graph, an auxiliary graph derived

*This work is supported by the Austrian Science Foundation (FWF) under grant P14445-MAT and by MURST under grant MM09308497.

from the structures being matched. This framework is attractive because it casts the matching problem in terms of a pure graph-theoretic problem, for which a solid theory and powerful algorithms have been developed [4].

In the second approach, the relational matching problem is viewed as one of energy minimization. In this case, an energy (or objective) function is sought whose minimizers correspond to the solutions of the original problem, and a dynamical system, usually embedded into a parallel relaxation network, is used to minimize it [9, 10, 22]. Typically, these methods do not solve the problem exactly, but only in approximation terms. Energy minimization algorithms are attractive because they are amenable to parallel hardware implementation and also offer the advantage of biological plausibility.

In [16], we have developed a new framework for graph matching which does unify the two approaches just described, thereby inheriting the attractive features of both. The approach is centered around a remarkable result in graph theory which allows us to map the maximum clique problem onto the problem of extremizing a quadratic form over a linearly constrained domain (i.e., the standard simplex in Euclidean space) [15]. Local gradient-based search methods such as *replicator dynamics* have proven to be remarkably effective when we restrict ourselves to simple versions of the problem, such as tree matching [19, 18] or graph isomorphism [17]. For more difficult versions of the problem a pivoting-based heuristic (PBH), which is based on a linear complementarity formulation [8], has recently proven to be quite effective [13, 12].

In [11] it has been shown that PBH is essentially equivalent to a greedy combinatorial heuristic and that modifying the combinatorial counterpart of PBH, the quality of the results may be improved, at the cost of moderately increasing the computation time. Basically, the modification consists of allowing for vertex swaps during the clique-construction process. In this paper we apply these heuristics to graph matching problems. Experiments on random graphs with various levels of corruption are presented. We found that the purely greedy heuristic performed as well as PBH but was dramatically faster, and the swap strategy always improved the results. Both algorithms were also found to perform better than graduated assignment [9], a state-of-the-art representative of energy minimization methods.

The plan of the paper is the following. In Section 2 we formulate the graph matching problem as a maximum clique problem on a derived association graph. The interpretation of PBH in terms of a greedy combinatorial heuristic is given in Section 3 and Section 4 is devoted to vertex swaps. Experimental results are provided in Section 5.

2 Graph Matching and Maximal Cliques

Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, an *isomorphism* between them is any bijection $\phi : V_1 \rightarrow V_2$ such that $(i, j) \in E_1 \Leftrightarrow (\phi(i), \phi(j)) \in E_2$, for all $i, j \in V_1$. Two graphs are said to be isomorphic if there exists an isomorphism between them. The maximum common subgraph problem consists of finding the

largest isomorphic subgraphs of G_1 and G_2 . A simpler version of this problem is to find a maximal common subgraph, i.e., an isomorphism between subgraphs which is not included in any larger subgraph isomorphism.

The *association graph* derived from $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is the undirected graph $G = (V, E)$ defined as follows:

$$V = V_1 \times V_2$$

and

$$E = \{((i, h), (j, k)) \in V \times V : i \neq j, h \neq k, \text{ and } (i, j) \in E_1 \Leftrightarrow (h, k) \in E_2\} .$$

Given an arbitrary undirected graph $G = (V, E)$, a subset of vertices C is called a *clique* if all its vertices are mutually adjacent, i.e., for all $i, j \in C$, with $i \neq j$, we have $(i, j) \in E$. A clique is said to be *maximal* if it is not contained in any larger clique, and *maximum* if it is the largest clique in the graph. The *clique number*, denoted by $\omega(G)$, is defined as the cardinality of the maximum clique.

The following result establishes an equivalence between the graph matching problem and the maximum clique problem (see, e.g., [2]).

Theorem 2.1 *Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two graphs, and let G be the corresponding association graph. Then, all maximal (maximum) cliques in G are in one-to-one correspondence with maximal (maximum) common subgraph isomorphisms between G_1 and G_2 .*

The problem of finding a maximum clique in a graph is known to be *NP*-hard for arbitrary graphs and so is the problem of approximating it within a constant factor. However, several powerful heuristics have been developed in the last few years. We refer to [4] for a recent review concerning algorithms, applications and complexity issues of this important problem.

3 Combinatorics of Pivoting

Recently a powerful pivoting-based heuristic (PBH) has been introduced for attacking the maximum clique problem, based on the linear complementarity formulation of an equivalent (standard) quadratic program [13]. In [11] it has been shown that the continuous approach PBH for the solution of the maximum clique problem is equivalent to a combinatorial heuristic. Such a heuristic turns out to belong to the class of greedy heuristics denoted in [5] by SM^i ($i = 1, 2, \dots$), which are described in what follows.

Greedy heuristic SM^i

Step 1 Given a graph $G = (V, E)$, let \mathcal{Q} be the set of all cliques of graph G with cardinality i . Set $\mathcal{K} = \emptyset$, $K^* = \emptyset$ and $max = 0$.

Step 2 If $\mathcal{Q} \neq \emptyset$, select a clique $H \in \mathcal{Q}$ and update $\mathcal{Q} = \mathcal{Q} \setminus \{H\}$. Put $K = H$ and go to Step 3; otherwise, return max and K^* .

Step 3 If K is a maximal clique, then update $\mathcal{K} = \mathcal{K} \cup \{K\}$ and go to Step 4; otherwise go to Step 5.

Step 4 If $|K| > \max$, set $\max = |K|$ and $K^* = K$. Go back to Step 2.

Step 5 Let N_j denote the neighborhood of vertex j , i.e.

$$N_j = \{k \in V : (j, k) \in E\}. \quad (1)$$

Select a vertex

$$\ell \in \arg \max_{j \in C_0(K)} |C_0(K) \cap N_j| \quad (2)$$

at random, where

$$C_0(K) = \{j \in V \setminus K : (j, k) \in E \ \forall k \in K\} = \bigcap_{k \in K} N_k \quad (3)$$

is the set of all vertices outside K that are adjacent to each vertex in the current clique K .

Step 6 Set $K = K \cup \{\ell\}$ and go back to Step 3.

We notice that the combinatorial version of PBH is equivalent to SM^1 , except for the minimal difference of the random selection of ℓ in (2)—in PBH ℓ is selected as the node with the lowest index in $C_0(K)$. Thus, we established a new connection between the technique of pivoting and well-known heuristics for finding large cliques. To our knowledge, the SM^i heuristics have never been applied to association graphs.

Although the heuristic SM^1 frequently returns results of good quality, the heuristic SM^2 is often superior to SM^1 . This was shown in [11], where numerous tests were made using DIMACS graphs. On the other hand, as expected, the computation times for SM^2 are much larger than those for SM^1 [11]. In the next section we shall modify the heuristic SM^1 in such a way that the modification is closer to SM^2 than to SM^1 from the point of view of the quality of the results, but is much closer to SM^1 than to SM^2 from the point of view of the computation times.

4 Vertex Swaps

The heuristics SM^i start with a clique of cardinality i and add a new vertex at each iteration until a maximal clique is reached. Some other approaches do the same but do not stop once a maximal clique is reached, allowing for the removal of vertices when this situation occurs (see e.g. the plateau search in [3]). The modification proposed here does not always add a new vertex to the current clique but also allows to swap vertices. If K is the current clique, let $C_0(K)$ be defined as in (3) and let

$$C_1(K) = \{j \in V \setminus K : |N_j \cap K| = |K| - 1\},$$

i.e. $C_1(K)$ is the set of vertices outside K which are connected to all vertices in K but exactly one. If we select a vertex $\ell \in C_0(K)$, then we can add it to K and update K as follows

$$K = K \cup \{\ell\}.$$

This is exactly what heuristics SM^i do with the selection (2). But if we select a vertex $\ell \in C_1(K)$ we can not simply add it to K ; if we still want to have a clique we need to swap vertex ℓ with the unique vertex

$$k_\ell \in K \quad \text{such that} \quad (\ell, k_\ell) \notin E, \quad (4)$$

i.e. the new clique will be

$$K = K \cup \{\ell\} \setminus \{k_\ell\}.$$

Therefore, if we select a vertex in $C_1(K)$ we change the current clique but without increasing the cardinality, i.e. the method is not strictly monotone. In the modification proposed here we decide whether to increase the cardinality, by selecting a vertex in $C_0(K)$, or to swap vertices, by selecting a vertex in $C_1(K)$. Note that $C_0(K) = \emptyset$ if K is a maximal clique. Similarly, some authors call a maximal clique K *strictly maximal*, if $C_1(K) = \emptyset$, i.e., if every swap destroys the clique property.

Now, selection of a vertex in $C_0(K) \cup C_1(K)$ is done according to a rule similar to (2) but with the computation of the maximum extended also to vertices in $C_1(K)$, i.e.

$$\ell \in \arg \max_{j \in C_0(K) \cup C_1(K)} |C_0(K) \cap N_j|. \quad (5)$$

We underline that in some cases we force the heuristic to select the next vertex according to (2), namely:

- during a fixed number (*START_SWAP*) of initial iterations;
- when the number of swaps performed is a fixed number T times the cardinality of the current clique $|K|$;
- when the vertex selected in $C_1(K)$ is the same as the last one removed by the last swap. Without this condition, which can be interpreted as a very limited application of taboo rules, it has been observed that the heuristic is much less efficient (worse results and larger computation times), apparently because it happens that the same two vertices are swapped in and out until the maximum allowed number of swaps is reached and then the heuristic behaves exactly as the standard greedy one. Experimentally, we found that longer lists of forbidden vertices, corresponding to higher order taboo rules, have no effect on the quality of the results, i.e. on the sizes of the cliques.

The heuristic with the proposed modification, denoted with *SM¹_SWAP* is described in what follows.

Heuristic SM^1_SWAP

Step 1 Given a graph $G = (V, E)$, let $W = V$, $K^* = \emptyset$ and $max = 0$. Select a value for the parameters $START_SWAP$ and T .

Step 2 If $W = \emptyset$ return K^* and max . Otherwise select a vertex $h \in W$, set $W = W \setminus \{h\}$ and set $K = \{h\}$. Set $n_swaps = 0$ and $last_swap = \emptyset$.

Step 3 If K is a maximal clique, then set $\mathcal{K} = \mathcal{K} \cup \{K\}$ and go to Step 4; otherwise go to Step 5.

Step 4 If $|K| > max$, set $max = |K|$ and $K^* = K$. Go back to Step 2.

Step 5 If $n_swaps \leq T|K|$ and $|K| \geq START_SWAP$, then go to Step 6, otherwise go to Step 7.

Step 6 Select randomly $\ell \in V$ according to (5). If $\ell = last_swap$, then go to Step 7, otherwise go to Step 8.

Step 7 Select $\ell \in V$ according to (2).

Step 8 If $\ell \in C_1(K)$, then update

$$n_swaps = n_swaps + 1 \quad \text{and} \quad last_swap = k_\ell$$

where k_ℓ is defined in (4), and

$$K = K \cup \{\ell\} \setminus \{k_\ell\}.$$

Otherwise set

$$K = K \cup \{\ell\}.$$

Go back to Step 3.

5 Experimental Results

In this section we test the performance of SM^1 and SM^1_SWAP on graph matching problems, and compare them with a well-known representative of energy minimization methods, i.e., Gold and Rangarajan's graduated assignment algorithm [9] (abbreviated as GR in what follows). All experiments are performed on random graphs. Random structures represent a useful benchmark not only because they are not constrained to any particular application, but also because it is simple to replicate experiments and hence to make comparisons with other algorithms.

The following protocol is standard and was already used in [12]. We generated random 50-node graphs using edge-probabilities (i.e. densities) ranging from 0.1 to 0.9. For each density value, 20 graphs were constructed so that, overall, 180 graphs were used in the experiments. Each graph had its vertices permuted and was then subject to a corruption process which consisted

of randomly deleting a fraction of its nodes. In so doing we obtained a graph isomorphic to a proper subgraph of the original one. Various levels of corruption (i.e., percentage of node deletion) were used, namely 0% (the pure isomorphism case), 10%, 20% and 30%. In other words, the order of the corrupted graphs ranged from 50 to 35. Each algorithm was applied on each pair of graphs thus constructed and the percentage of matched nodes was recorded. All experiments were performed on a 1.4 GHz AMD Athlon processor.

Since PBH is essentially equivalent to SM^1 , the results obtained by the two heuristics were almost identical, as expected. However, the computation time differed dramatically. For example, in case of pure isomorphism (i.e., 0% level of corruption) and a density of 0.5, PBH required more than 4 hours, whereas SM^1 terminated after only 14 seconds. Similar differences were also found at other densities and levels of corruption. We note that in [13] it was shown that PBH compares well with state-of-the-art clique finding algorithms.

Before proceeding with the experimental results, we note that the GR algorithm may yield “inconsistent” results. Formally, let the two graphs be denoted by G_1 and G_2 . Furthermore let U_1 [U_2] denote the set of vertices in G_1 [G_2] that have been matched to a vertex in G_2 [G_1]. Then, the subgraph of G_1 induced by U_1 and the subgraph of G_2 induced by U_2 are not necessarily isomorphic. In other words, the matches found by GR do not necessarily induce a clique in the association graph of G_1 and G_2 (see Section 2). To find a maximum consistent “kernel” of the results from GR, we applied an exact algorithm for the maximum clique problem, i.e. Bron and Kerbosch [6], to the subgraph of the association graph induced by the matches. The following results on GR refer to the output of the Bron and Kerbosch algorithm. The computation times, however, do not include the time needed by the Bron and Kerbosch algorithm.

Figures 1 to 4 show the results obtained at various levels of corruption. Each figure shows two plots. One concerns the quality of the solutions found and the other the CPU time. For each plot, the x-axis represents the (approximate) density of the matched graphs. The y-axis of the top plots represents the portion of the correct matches. Here, ω is the size of the maximum clique of the association graph, i.e., the size of the maximum isomorphism, and $|C|$ is the size of the isomorphism returned by the algorithms, i.e. the size of the maximal clique found (in the case of GR, this corresponds to the size of the maximum consistent kernel, as described before). Finally, y-axis of the bottom plots represents CPU time expressed in seconds. All curves represent average over 20 trials.

Let us first focus on the quality of the results, i.e. on the top part of the figures. As corruption increases, the advantage of SM^1 and SM^1_SWAP over GR becomes more and more evident. Moreover, except for the pure isomorphism case in which SM^1 and SM^1_SWAP both yielded always the optimal result, SM^1_SWAP is always superior to SM^1 .

Looking at the computation times, see the bottom part of the figures, one can easily distinguish the bowl shapes of SM^1 and SM^1_SWAP from the monotonic behavior of GR. The “bowls” are due to the fact that the computation times of SM^1 and SM^1_SWAP depend mainly on the number of edges in the association

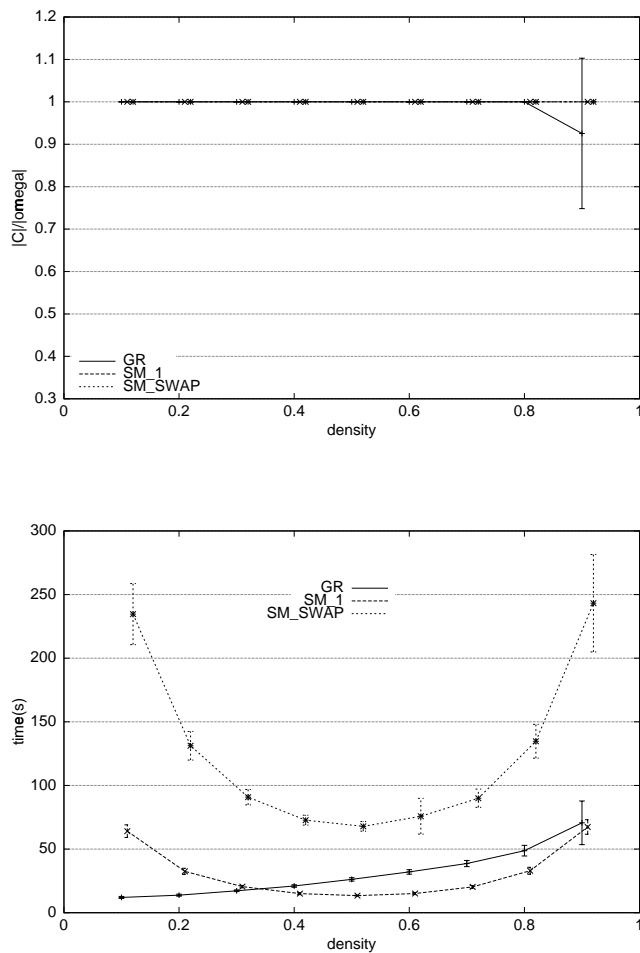


Figure 1: Results of matching 50-node random graphs with varying densities, and 0% corruption. Top: Portion of correct matches. Bottom: Average computational time taken by the algorithms (in seconds).

graph and that the density of the latter becomes high, if G_1 and G_2 are sparse or dense.

6 Conclusions

In this paper, we have proposed a class of combinatorial heuristics for graph matching, based on the well-known idea of finding maximal cliques in the association graph. The basic heuristic of the class, equivalent to a recently in-

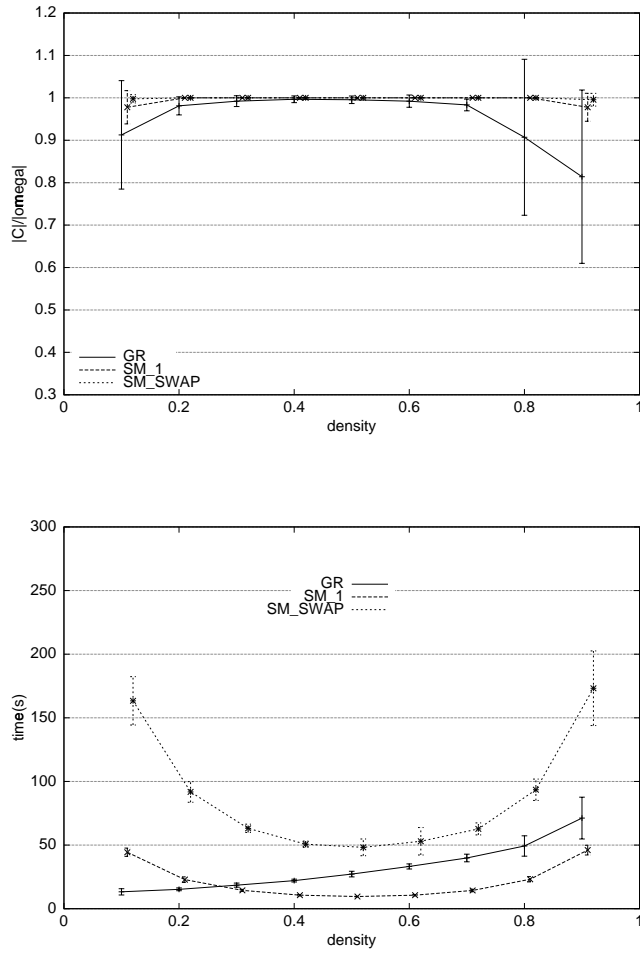


Figure 2: Results of matching 50-node random graphs with varying densities, and 10% corruption. Top: Portion of correct matches. Bottom: Average computational time taken by the algorithms (in seconds).

roduced pivoting-based algorithm, was shown to fall into the class of greedy algorithms. Trying to avoid one of the classical drawbacks of greedy strategies (i.e., the ease of being trapped in poor local optima), we proposed a modification of the basic algorithm allowing for vertex swaps during the clique-construction process. Experiments on random graphs at various levels of corruption have shown the effectiveness of the approaches in terms of quality of solutions and speed. Future work will aim at extending the proposed heuristics to deal with attributed graph matching, along the lines suggested in [16, 19].

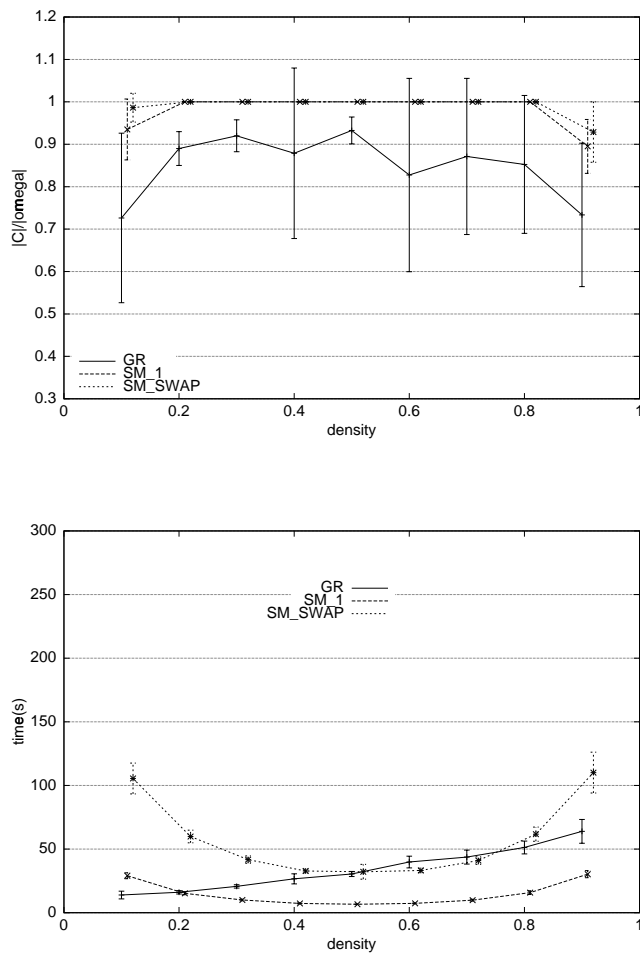


Figure 3: Results of matching 50-node random graphs with varying densities, and 20% corruption. Top: Portion of correct matches. Bottom: Average computational time taken by the algorithms (in seconds).

References

- [1] A. P. Ambler et al. A versatile computer-controlled assembly system. In *Proc. of 3rd Int. J. Conf. Art. Intell.*, pages 298–307, 1973.
- [2] H. G. Barrow and R. M. Burstall. Subgraph isomorphism, matching relational structures and maximal cliques. *Inform. Process. Lett.*, 4(4):83–84, 1976.

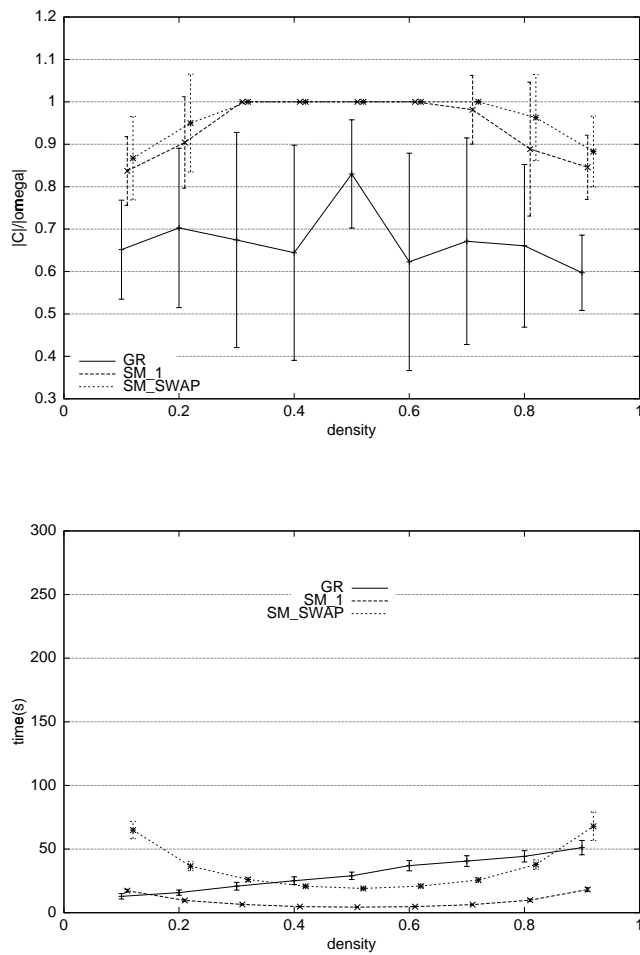


Figure 4: Results of matching 50-node random graphs with varying densities, and 30% corruption. Top: Portion of correct matches. Bottom: Average computational time taken by the algorithms (in seconds).

- [3] R. Battiti and M. Protasi. Reactive local search for the maximum clique problem. *Algorithmica*, 29:610–637, 2001.
- [4] I. M. Bomze, M. Budinich, M. P. Pardalos, and M. Pelillo. The maximum clique problem. In D.-Z. Du and P. M. Pardalos, editors, *Handbook of Combinatorial Optimization (Suppl. Vol. A)*, pages 1–74. Kluwer, Boston, MA, 1999.
- [5] M. Brockington and J. C. Culberson. Camouflaging independent sets in

- quasi-random graphs. In D. Johnson and M. A. Trick, editors, *Cliques, Coloring and Satisfiability*, volume 26 of *DIMACS series in Discrete Mathematics and Theoretical Computer Science*, pages 75–88. Americ. Math. Soc., 1996.
- [6] C. Bron and J. Kerbosch. Algorithm457: Finding all cliques of an undirected graph. *Comm. ACM*, 16:575–577, 1973.
- [7] H. Bunke. Recent developments in graph matching. In A. Sanfeliu, J. Villanueva, M. Vanrell, R. Alquezar, A. Jain, and J. Kittler, editors, *Proc. 15th Int. Conf. Pattern Recognition*, volume 2, pages pp.117–124. IEEE Computer Society, 2000.
- [8] R. W. Cottle, J.-S. Pang, and R. E. Stone. *The Linear Complementarity Problem*. Accademic Press, Boston, MA, 1992.
- [9] S. Gold and A. Rangarajan. A Graduated Assignment Algorithm for Graph Matching. *IEEE Trans. Pattern Anal. and Machine Intell.*, 18(4):377–388, 1996.
- [10] S. Z. Li. Matching: invariant to translations, rotations and scale changes. *Pattern Recognition*, 25(6):583–594, 1992.
- [11] M. Locatelli, I. M. Bomze, and M. Pelillo. Swaps, diversification, and the combinatorics of pivoting for the maximum weight clique. Technical Report CS-2002-12, Dipartimento di Informatica, Università Ca’ Foscari di Venezia, 30172 Venezia Mestre, Italy, 2002.
- [12] A. Massaro and M. Pelillo. Matching graphs by pivoting. *Pattern Recognition Letters*, 24:1099–1106, 2003.
- [13] A. Massaro, M. Pelillo, and I. M. Bomze. A complementary pivoting approach to the maximum weight clique problem. *SIAM Journal on Optimization*, 12(4):928–948, 2002.
- [14] B. Messmer and H. Bunke. A New Algorithm for Error-Tolerant Subgraph Isomorphism Detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(7):493 – 504, 1998.
- [15] T. S. Motzkin and E. G. Straus. Maxima for graphs and a new proof of a theorem of Turán. *Cand. J. Math.*, 17:533–540, 1965.
- [16] M. Pelillo. A unifying framework for relational structure matching. In A. K. Jain, S. Venkatesh, and B. C. Lovell, editors, *Proc. 14th Int. Conf. Pattern Recognition*, pages 1316–1319. IEEE-Computer Society Press, 1998.
- [17] M. Pelillo. Replicator equations, maximal cliques, and graph isomorphism. *Neural Computation*, 11:1933–1955, 1999.

- [18] M. Pelillo. Matching free trees, maximal cliques, and monotone game dynamics. *IEEE Trans. Pattern Anal. and Machine Intell.*, 24(11):1535–1541, 2002.
- [19] M. Pelillo, K. Siddiqi, and S. W. Zucker. Matching Hierarchical Structures Using Association Graphs. *IEEE Trans. Pattern Anal. and Machine Intell.*, 21(11):1105–1120, 1999.
- [20] L. G. Shapiro and R. M. Haralick. Structural descriptions and inexact matching. *IEEE Trans. Pattern Anal. and Machine Intell.*, 3:504–519, 1981.
- [21] W. H. Tsai and K. S. Fu. Subgraph error-correcting isomorphisms for syntactic pattern recognition. *IEEE Trans. Syst. Man Cybern.*, 13:48–62, 1983.
- [22] R. C. Wilson and E. R. Hancock. Structural Matching by Discrete Relaxation. *Trans. Pattern Anal. Machine Intell.*, 19(6):634–648, 1997.