Randomized Algorithms for Scene Recognition by Inexact Graph Matching

Maria C. Boeres [a], Celso C. Ribeiro [b] and Isabelle Bloch [c]

[a] *Universidade Federal do Espírito Santo, Department of Computer Science, R. Fernando Ferrari, Campus de Goiabeiras, Vitoria, ES 29060-970, Brazil.*
boeres@inf.ufes.br

[b] *Department of Computer Science, Catholic University of Rio de Janeiro, R. Marquês de São Vicente 225, Rio de Janeiro, RJ 22453-900, Brazil.*
celso@inf.puc-rio.br

[c] *Ecole Nationale Supérieure des Télécommunications, CNRS URA 820, 46 rue Barrault, 75634 Paris Cedex 13, France.*
Isabelle.Bloch@enst.fr

Abstract

We propose a new method for non-bijective graph matching for model-based pattern recognition. We formulate the search for the best correspondence between a model and an over-segmented image as a new combinatorial optimization problem, defined by the relational attributed graphs representing the model and the image where recognition has to be performed, together with the node and edge similarities between them. We discuss the choice of the objective function and the nature of the constraints of the graph matching problem. A randomized construction algorithm is proposed to build feasible solutions to the problem. A neighborhood structure and a local search procedure for solution improvement are also proposed. Computational results on ten test instances are presented and discussed, illustrating the effectiveness of the combined approach involving randomized construction and local search.

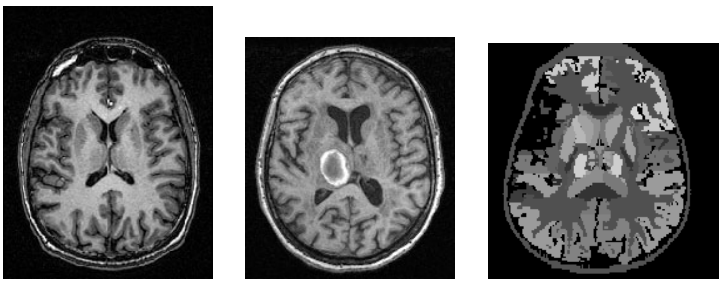
Key words: Model-based scene recognition, graph matching, heuristics, randomized algorithms, local search.

1 Introduction

The recognition and understanding of complex scenes requires not only a detailed description of the objects involved, but also of the spatial relationships

between them. Indeed, the diversity of the forms of the same object in different instantiations of a scene, and also the similarities of different objects in the same scene, make relationships between objects of prime importance in order to disambiguate the recognition of objects with similar appearance. Graph based representations are often used for scene representation in image processing [6,8,12,22,23]. Vertices of the graphs usually represent the objects in the scenes, while their edges represent the relationships between the objects. Relevant information for the recognition is extracted from the scene and represented by relational attributed graphs. In model-based recognition, both the model and the scene are represented by graphs. *Graph matching* is one of the important problems to be solved whenever such representations are used. Many publications deal with the problem of graph or subgraph isomorphism and there is a huge literature on this subject, see e.g. [7,8,11]. However, the assumption of a bijection between the elements in two instantiations of the same scene is too strong for many problems. In this case, scene recognition may be better expressed as an inexact graph matching problem. As a matter of fact, inexact graph matching appears as an important research subject in the field of model-based pattern recognition. Usually, the model has a schematic aspect. Moreover, the construction of the image graph often relies on segmentation techniques that may fail in accurately segmenting the image into meaningful entities. Therefore, no isomorphism can be expected between both graphs and such problems call for inexact (i.e., non-bijective) graph matching.

Our motivation comes from an application in medical imaging, in which the goal consists in recognizing brain structures from 3D magnetic resonance images, previously processed by a segmentation method. This application typically raises the type of problem above mentioned. The model consists of an anatomical atlas. A graph is built from the atlas, in which each node represents exactly one anatomical structure of interest. Edges of this graph represent spatial relationships between the anatomical structures. Inaccuracies constitute one of the main characteristics of the problem. Objects in the image are segmented and all difficulties with object segmentation will be reflected in the representation, such as under-segmentation and/or over-segmentation (we restrict ourselves to the second case), unexpected objects found in the scene (pathologies for instance), expected objects not found and deformations of objects (see [14] for details). Also, the attributes computed for both the image and the model may be imprecise. To illustrate these difficulties, Figure 1 presents slices of three different volumes: (a) a normal brain, (b) a pathological brain with a tumor, and (c) the representation of a brain atlas where each grey level corresponds to a unique connected structure. Middle dark structures (lateral ventricles) are much bigger in (b) than in (a). The white hyper-signal structure (tumor) does not appear in the atlas (c) nor in the normal brain (a). Similar problems occur in other applications, such as aerial or satellite image interpretation using a map, face recognition or character recognition.



(a) (b) (c)

Fig. 1. Examples of magnetic resonance images: (a) axial slice of a normal brain, (b) axial slice of a pathological brain with a tumor, and (c) axial slice of a brain atlas.

This paper focuses on algorithms for the graph matching problem, rather than on specific applications. Several techniques have been applied to graph matching, such as combinatorial optimization algorithms [7,21], relaxation-based methods [14,16,19], expectation and maximization algorithms [9], and estimation of distribution algorithms [1].

In this work, we formulate the scene recognition problem as a combinatorial optimization problem defined by the relational attributed graphs representing the model and the over-segmented image, together with the node and edge similarities between their nodes and edges. Graph morphisms are summarized in the next section. Section 3 describes in detail our formulation of the search for the best correspondence between the two graphs as a non-bijective graph matching problem. We discuss the choice of the objective function and the nature of the constraints of the graph matching problem. A randomized construction algorithm is proposed in Section 4 to build feasible solutions to the problem. Besides the quality of the solutions found, this algorithm may also be used as a robust generator of initial solutions for a GRASP metaheuristic [17] or for population methods such as the genetic algorithm described in [15]. A local search algorithm is proposed in Section 5 to improve the solutions obtained by the construction algorithm. Numerical results obtained with the randomized construction and the local search algorithms are presented and discussed in Section 6. They illustrate the robustness of the construction algorithm and the improvements attained by the local search algorithm in terms of solution quality and object identification. Concluding remarks are presented in the last section.

2 Graph Morphism

Attributed graphs are widely used in pattern recognition problems. The definition of the attributes and the computation of their values are specific to each

application and problem instance. Fuzzy attributed graphs are used for recognition under imprecision [5,14]. For instance, some possible attributes of the relations between brain structures in the medical imaging example depicted in Figure 1 are fuzzy adjacencies [4], fuzzy distances [3], and fuzzy directional relative positions [2]. The construction of a FAG depends on the imperfections of the scene or of the reference model, and on the attributes of the object relations. The common point is that there is always a single vertex for each region of each image. Differences may occur due to the strategy applied for the creation of the edge set, as a result of the chosen attributes or of defects in scene segmentation. Once the graph is built, the next step consists in computing the attributes of vertices and edges. Finally, vertex-similarity and edge-similarity matrices are computed from the values of the attributed graphs, relating each pair of vertices and each pair of edges, one of them from the model and the other from the segmented image.

Two graphs are used to represent the problem: $G_1 = (N_1, E_1)$ represents the model, while $G_2 = (N_2, E_2)$ represents the over-segmented image. In each case, N_i denotes the vertex set and E_i denotes the edge set, $i \in \{1, 2\}$. We assume that $|N_1| \leq |N_2|$, which is the case when the image is over-segmented with respect to the model. Fuzzy morphisms have been introduced in [14] and allow to assign each node of G_1 to multiple nodes of G_2, with different degrees of strength of association.

In this work, each node of G_2 is assigned to a label corresponding to one node of G_1. These assignments are represented by binary variables: $x_{ij} = 1$ if nodes $i \in N_1$ and $j \in N_2$ are associated, $x_{ij} = 0$ otherwise. To ensure that the structure of G_1 appears within G_2, we favor solutions where a correspondence between edges also implies a correspondence between their extremities (edge association condition). Thus, edge associations are derived from vertex associations, according to the following rule. Let (a, b) be any edge of G_1. Then, edge $(a, b) \in E_1$ is associated with all edges $(a', b') \in E_2$ such that (i) $a' \in N_2$ is associated with $a \in N_1$ and $b' \in N_2$ is associated with $b \in N_1$ or (ii) $a' \in N_2$ is associated with $b \in N_1$ and $b' \in N_2$ is associated with $a \in N_1$.

3 Optimal Matching

The definition of graph morphism in the previous section typically allows many correspondences between G_1 and G_2. In this section, we formulate the *Graph Matching Problem* (GMP). Each of its solutions is a correspondence between the vertex and edge sets of the graphs representing the model (G_1) and the over-segmented image (G_2). Figure 2 illustrates two different solutions of a small instance. Each model vertex (resp. model edge) and the associated image vertices (resp. edges) are represented by the same color. The associations

depend on problem constraints and attribute values.

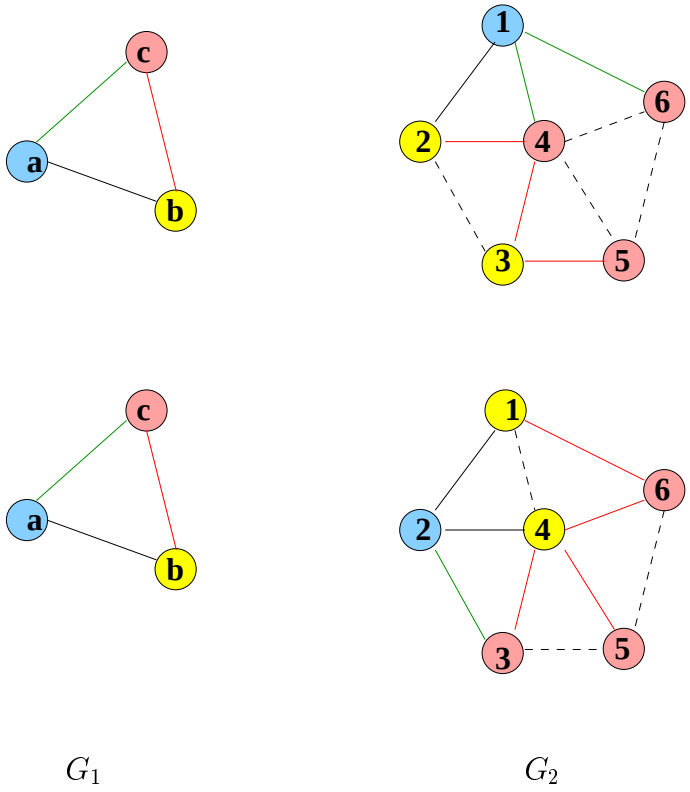


Fig. 2. Two distinct solutions of the same instance.

All correspondences allowed by the morphism formulation discussed in Section 2 share a common characteristic: edges linking image vertices associated with the same model vertex are not in correspondence with any model edge. The dashed lines in Figure 2 represent edges of E_2 which are not associated with edges of E_1. The set

$$A(i) = \{j \in N_2 \mid x_{ij} = 1\} \tag{1}$$

denotes the subset of vertices of N_2 associated with vertex $i \in N_1$.

A good matching is a solution to GMP in which the associations correspond to high similarity values. The value of any solution is expressed by an objective function which is optimized. Due to inaccuracies and bad description of relevant structural aspects of the input data, it is not easy to define an appropriate objective function whose global optimum always coincides with the best correspondence. We have investigated several objective functions to be used in the formulation of GMP, in terms of their properties and capabilities of recognition.

Similarity matrices are constructed from similarity values calculated from graph attributes. The choice of these attributes depends on the images. Let S^v

(resp. S^e) denote an $|N_1| \times |N_2|$ (resp. $|E_1| \times |E_2|$) vertex-similarity (resp. edge-similarity) matrix, where the elements $s^v(i,j)$ (resp. $s^e((i,i'),(j,j'))) \in [0,1]$ represent the similarity between vertices (resp. edges) $i \in N_1$ and $j \in N_2$ (resp. $(i,i') \in E_1$ and $(j,j') \in E_2$).

A small, illustrative example is described below. The model and image graphs are depicted in Figure 3. The vertex and edge similarity matrices are provided in Tables 1 and 2, respectively. Edge similarities are not very relevant in this example, since they are not discriminant between different correspondences. The vertex similarity matrix shows ambiguous information concerning vertex similarities. Therefore, there are several good solutions in terms of the node associations, and not only a distinguishable one.

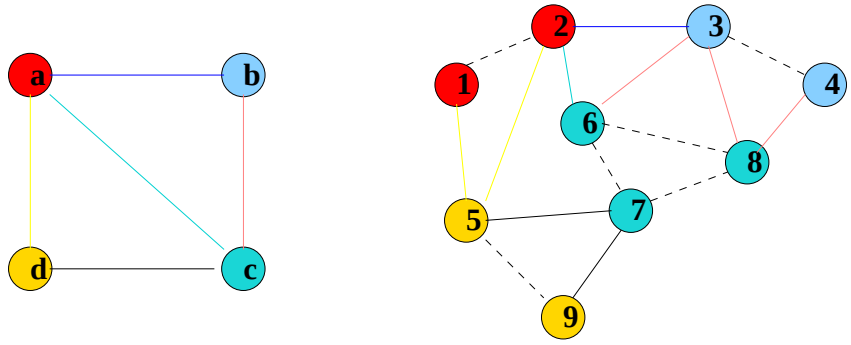


Fig. 3. Example: model and image graphs and one possible correspondence.

Table 1
Node similarity matrix S^v.

	1	2	3	4	5	6	7	8	9
a	0.910	1.	0.750	0.	0.750	0.850	0.200	0.100	0.
b	0.	0.	1.	1.	0.	0.	0.	0.9	0.
c	1.	0.	0.	0.	0.	0.800	1.	1.	0.
d	1.	0.	0.	0.	1.	0.	1.	0.	0.700

To reduce the solution space, we restrain the search only to solutions which satisfy Constraint 1 below, which states that each vertex of N_2 has to be associated with exactly one vertex of N_1. The rationale for this condition is that image segmentation is performed by an appropriate algorithm which preserves the boundaries and, in consequence, avoids situations in which one region of the segmented image is located in the frontier of two adjacent regions of the model, and no undersegmentation occurs.

Constraint (1): For every $j \in N_2$, there exists exactly one node $i \in N_1$ such that $x_{ij} = 1$, i.e. $|A^{-1}(j)| = 1$.

Table 2
Edge similarity matrix S^e.

	(1,2)	(1,5)	(2,3)	(2,5)	(2,6)	(3,4)	(3,6)	(3,8)
(a,b)	0.99	1.	1.	0.99	1.	0.99	0.99	0.99
(a,c)	0.99	1.	1.	0.99	1.	0.99	0.99	0.99
(a,d)	0.51	0.50	0.50	0.51	0.50	0.51	0.51	0.51
(b,c)	0.99	1.	1.	0.99	1.	0.99	0.99	0.99
(c,d)	0.99	1.	1.	0.99	1.	0.99	0.99	0.99

	(4,8)	(5,7)	(5,9)	(6,7)	(6,8)	(7,8)	(7,9)
(a,b)	0.99	1.	0.98	0.99	0.99	0.99	1.
(a,c)	0.99	1.	0.98	0.99	0.99	0.99	1.
(a,d)	0.51	0.50	0.52	0.51	0.51	0.51	0.50
(b,c)	0.99	1.	0.98	0.99	0.99	0.99	1.
(c,d)	0.99	1.	0.98	0.99	0.99	0.99	1.

For each solution x, we define

$$f_1(x) = \frac{\alpha}{|N_1| \cdot |N_2|} f_1^v(x) + \frac{(1-\alpha)}{|E_1| \cdot |E_2|} f_1^e(x),$$

with

$$f_1^v(x) = \sum_{i \in N_1} \sum_{j \in N_2} (1 - |x_{ij} - s^v(i,j)|)$$

and

$$f_1^e(x) = \sum_{(i,i') \in E_1} \sum_{(j,j') \in E_2} (1 - |\max\{x_{ij}x_{i'j'}, x_{ij'}x_{i'j}\} - s^e((i,i'),(j,j'))|),$$

where α is a parameter used to weight each term of f_1 and $x_{ij} = 1$ if nodes $i \in N_1$ and $j \in N_2$ are associated, $x_{ij} = 0$ otherwise. This function consists of two terms which represent respectively the vertex and edge contributions to the measure of the solution quality associated with each correspondence. Vertex and edge associations with high similarity values are privileged by f_1, while those with low similarity values are penalized. This function behaves appropriately when the image features are well described by the graph attributes. The first term of $f_1(x)$ represents the average vertex contribution to the correspondence. Four extreme situations are possible:

- $x_{ij} = 0$ and $s^v(i,j) = 0$: the term $1 - |x_{ij} - s^v(i,j)|$ gives a maximal contribution to f_1 (vertices $i \in N_1$ and $j \in N_2$ with null similarity values

are not put into correspondence);

- $x_{ij} = 0$ and $s^v(i,j) = 1$: the term $1 - |x_{ij} - s^v(i,j)| = 0$ does not contribute to the objective function value if vertices $i \in N_1$ and $j \in N_2$ with high similarity values are not put into correspondence;
- $x_{ij} = 1$ and $s^v(i,j) = 0$: again, the term $1 - |x_{ij} - s^v(i,j)| = 0$ does not contribute to the value of the objective function if vertices $i \in N_1$ and $j \in N_2$ with null similarity values are put into correspondence; and
- $x_{ij} = 1$ and $s^v(i,j) = 1$: the term $1 - |x_{ij} - s^v(i,j)|$ gives a maximal contribution to f_1 (vertices $i \in N_1$ and $j \in N_2$ with high similarity values are put into correspondence).

A similar interpretation stands for the terms involving edge associations and their similarity values. The second term of $f_1(x)$ represents the average edge contribution to the correspondence and acts to enforce the edge association condition introduced in Section 2. For instance, if $s^e((i,i'),(j,j'))$ is high and there are associations between the extremities of edges (i,i') and (j,j'), then $\max\{x_{ij}x_{i'j'}, x_{ij'}x_{i'j}\} = 1$ and the contribution to f_1 is high. On the contrary, if the extremities of edges (i,i') and (j,j') are not associated, then $\max\{x_{ij}x_{i'j'}, x_{ij'}x_{i'j}\} = 0$ and the contribution to f_1 is null.

Figure 4 illustrates another solution to the example. Its objective function value is 0.707, while that of the solution in Figure 3 is 0.699 (for $\alpha = 0.8$). The vertex associations in this new solution are almost the same as those in the previously illustrated solution. However, the edge associations induced by them are different. The number of edges associated to the model in this solution is larger than in the former.

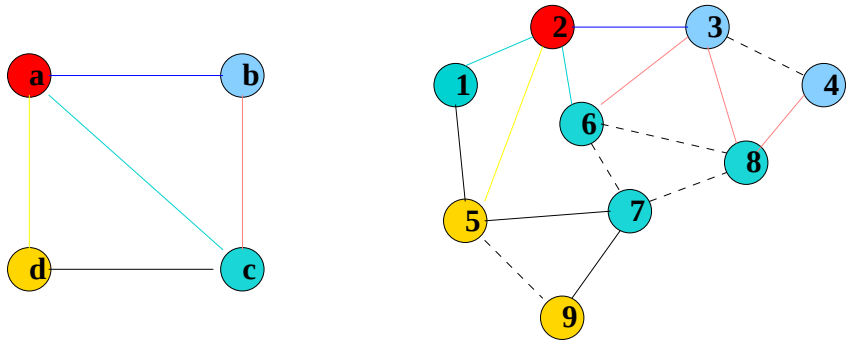


Fig. 4. Another solution with a larger value for the objective function.

The quality of the input data (vertex and edge similarity matrices) is primordial for the identification of the best correspondence. However, as this quality is not always easy to be achieved in real applications, we emphasize some aspects which can be used as additional information to improve the search. For example, in the correspondence shown in Figure 4, vertices 1, 6, 7, and 8 of N_2 associated to the model vertex $c \in N_1$ are not all connected among themselves. This is not a common caracteristic of real situations (since an over-segmentation method can split an object in several smaller pieces, but

it does not change the piece positions). A good strategy to avoid this type of solution is to restrain the search to correspondences for which each set $A(i)$ of vertices induces a connected subgraph in G_2, for every model vertex $i \in N_1$ (connectivity constraint). Regions of the segmented image corresponding to the same region of the model should necessarily be connected. This additional constraint is enforced in the formulation of GMP:

Constraint (2): For every $i \in N_1$, the subgraph induced in $G_2(N_2, E_2)$ by $A(i)$ is connected.

Pairs of vertices with null similarity cannot be associated. Such associations are discarded by the constraint below, which strengthens the penalization of associations between vertices with low similarity values induced by the objective function f_1.

Constraint (3): For every $i \in N_1$ and $j \in N_2$, if $s^v(i, j) = 0$, then $x_{ij} = 0$.

Finally, to ensure that all objects of the model appear in the image graph, one additional constraint is imposed:

Constraint (4): For every $i \in N_1$, there exists at least one node $j \in N_2$ such that $(i, j) \in E'$ (or, alternatively, $|A(i)| \geq 1$)).

4 Randomized Construction Algorithm

The construction algorithm proposed in this section is based on progressively associating a node of N_1 with each node of N_2, until a feasible solution x to GMP is built. The objective function

$$f_1(x) = \frac{\alpha}{|N_1| \cdot |N_2|} f_1^v(x) + \frac{(1-\alpha)}{|E_1| \cdot |E_2|} f_1^e(x),$$

does not have to be evaluated from scratch at each iteration. Its value is computed and initialized for $x = 0$ and updated after the creation of each new node association between a pair of vertices from N_1 and N_2. The following result holds:

$$f_1^v(x) = \sum_{i \in N_1} \sum_{j \in N_2} (1 - |x_{ij} - s^v(i,j)|) =$$
$$= \sum_{(i,j):x_{ij}=0} (1 - |0 - s^v(i,j)|) + \sum_{(i,j):x_{ij}=1} (1 - |1 - s^v(i,j)|) =$$
$$= \sum_{(i,j) \in N_1 \times N_2} (1 - s^v(i,j)) - \sum_{(i,j):x_{ij}=1} (1 - s^v(i,j)) + \sum_{(i,j):x_{ij}=1} s^v(i,j) =$$
$$= \sum_{(i,j) \in N_1 \times N_2} (1 - s^v(i,j)) + \sum_{(i,j):x_{ij}=1} (2 \cdot s^v(i,j) - 1).$$

Therefore, $f_1^v(x)$ is increased by $2 \cdot s^v(a,b) - 1$ whenever a node $a \in N_1$ is associated with a new node $b \in N_2$. Similar considerations are used in the evaluation of $f_1^e(x)$, which is increased by $2 \cdot s^e((a,a'),(b,b')) - 1$ whenever a new pair of edges $(a,a') \in E_1$ and $(b,b') \in E_2$ are associated.

The pseudo-code of the `RandomizedConstructionGMP` randomized algorithm to build solutions to GMP is given in Figure 5. The algorithm takes as parameters the initial seed, the maximum number $MaxTrials$ of attempts to build a feasible solution before stopping, and the maximum number $MaxSolutions$ of solutions built. The number of attempts, the number of solutions built, and the indicator that a feasible solution has been found are initialized in line 1. The optimal value is initialized in line 2. The loop in lines 3–35 performs at most $MaxTrials$ attempts to build at most $MaxSolutions$ solutions. Lines 4–7 prepare and initialize the data for each attempt. The solution $x = 0$, the set $A(i)$ of nodes associated with each node $i \in N_1$, and the node $A^{-1}(j)$ associated with each node $j \in N_2$ are initialized in line 4. The terms f_1^v and f_1^e, corresponding to the node similarity and edge similarity contributions to the objective function value of solution x, are initialized respectively in lines 5 and 6. A temporary copy V_2 of the node set N_2 is created in line 7.

The loop in lines 8–26 performs one attempt to create a feasible solution and stops after the associations to each node in V_2 have been investigated. A node $j \in V_2$ is randomly selected and eliminated from V_2 in line 9. A temporary copy V_1 of the node set N_1 is created in line 10. The loop in lines 11–25 searches for a node in N_1 to be associated with node $j \in V_2$. It stops after all possible associations to nodes in N_1 have been investigated without success or if one possible association was found. A node $i \in V_1$ is randomly selected and eliminated from V_1 in line 12. The algorithm checks in line 13 if node i can be associated with node j, i.e., if their similarity is not null and if the graph induced in G_2 by $A(i) \cup \{j\}$ is connected. If this is the case, the current solution and its objective function value are updated in lines 14–24. The current solution is updated in lines 15–16. The term f_1^v corresponding to the node similarities is updated in line 17. The term f_1^e corresponding to the edge similarities is updated in lines 18–23.

```
procedure RandomizedConstructionGMP(seed,MaxTrials,MaxSolutions)
1.   trials ← 1 and solutions ← 1;
2.   f* ← −∞ and feasible ← .FALSE.;
3.   while trials ≤ MaxTrials and solutions ≤ MaxSolutions do;
4.        x ← 0, A(i) ← ∅ ∀i ∈ N1, and A^{-1}(j) ← ∅ ∀j ∈ N2;
5.        f1^v ← Σ_{(i,j)∈N1×N2} (1 − s^v(i,j));
6.        f1^e ← Σ_{((i,i'),(j,j'))∈E1×E2} (1 − s^e((i,i'),(j,j')));
7.        V2 ← N2;
8.        while V2 ≠ ∅ do;
9.             Randomly select a node j from V2 and update V2 ← V2 − {j};
10.            V1 ← N1;
11.            while V1 ≠ ∅ and A^{-1}(j) = ∅ do;
12.                 Randomly select a node i from V1 and update V1 ← V1 − {i};
13.                 if s^v(i,j) > 0 and
                         the graph induced in G2 by A(i) ∪ {j} is connected
14.                 then do;
15.                      x_ij ← 1;
16.                      A(i) ← A(i) ∪ {j} and A^{-1}(j) ← {i};
17.                      f1^v ← f1^v + 2 · s_v(i,j) − 1;
18.                      forall i' ∈ N1 : (i,i') ∈ E1 do;
19.                           forall j' ∈ N2 : (j,j') ∈ E2 do;
20.                                if x_{i',j'} = 1
21.                                then f1^e ← f1^e + 2 · s^v((i,i'),(j,j')) − 1;
22.                           end_forall;
23.                      end_forall;
24.                 end_if;
25.            end_while;
26.       end_while;
27.       if A(i) ≠ ∅ ∀i ∈ N1 and A^{-1}(j) ≠ ∅ ∀j ∈ N2
28.       then do;
29.            feasible ← .TRUE.;
30.            solutions ← solutions + 1;
31.            Compute f1 ← α/(|N1| · |N2|) · f1^v + (1 − α)/(|E1| · |E2|) · f1^e;
32.            if f1 > f* then update f* ← f1 and x* ← x;
33.       end_if;
34.       trials ← trials + 1;
35.  end_while;
36.  return G* = (N1 ∪ N2, x*);
end RandomizedConstructionGMP.
```

Fig. 5. Pseudo-code of the randomized construction algorithm.

The algorithm checks in line 27 if the solution x built in lines 8–26 is feasible, i.e., if there is at least one node of N_2 associated with every node of N_1 and if there is exactly one node of N_1 associated with every node of N_2. If this is the case, the indicator that a feasible solution was found is reset in line 29 and the number of feasible solutions built is updated in line 30. The value of the

objective function for the new solution G' is computed in line 31. If the new solution is better than the incumbent, then the latter is updated in line 32. The number of attempts to build a feasible solution is updated in line 34 and a new iteration resumes, until the maximum number of attempts is reached. The best solution found x^* and its objective function value f^* are returned in line 36. In case no feasible solution was found, the returned value is $f^* = -\infty$.

We now evaluate the computational complexity of each attempt to build a feasible solution (lines 4–34). The node similarity and edge similarity contributions to the objective function value can be computed in time $O(|N_1| \cdot |N_2| + |E_1| \cdot |E_2|)$. The other initializations take time $O(|N_1| + |N_2|)$. The loop in lines 8–26 is performed exactly $|N_2|$ times. The loop in lines 11–25 is performed at most $|N_1|$ times and has time complexity $O(|N_1| \cdot |N_2|)$. The test in line 27 can be performed in time $O(|N_1| + |N_2|)$. Therefore, the overall complexity of each attempt to build a feasible solution is $O(|N_1|^2 \cdot |N_2|^2)$.

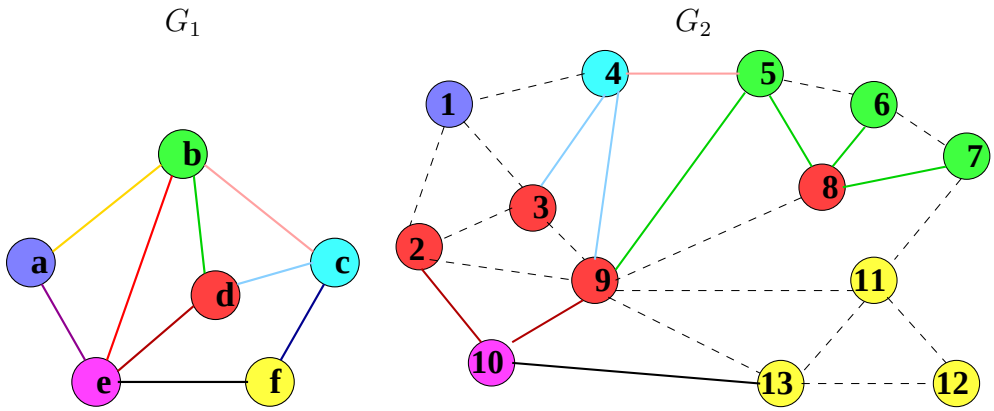
5 Local Search

The solutions generated by a randomized construction algorithm are not necessarily optimal, even with respect to simple neighborhoods. Hence, it is almost always beneficial to apply a local search to attempt to improve each constructed solution. A local search algorithm works in an iterative fashion by successively replacing the current solution by a better solution in the neighborhood of the current solution. It terminates when a local optimum is found, i.e., when no better solution can be found in the neighborhood of the current solution.

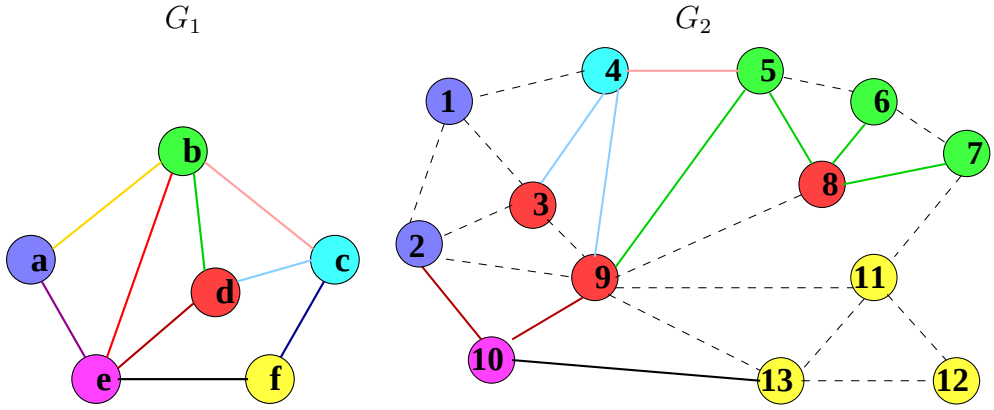
The effectiveness of a local search procedure depends on several aspects, such as the neighborhood structure, the neighborhood search technique, the fast evaluation of the cost function of each neighbor solution, and the starting solution itself. Simple neighborhoods are usually used. The neighborhood search may be implemented using either a *best-improving* or a *first-improving* strategy. In the case of the best-improving strategy, all neighbors are investigated and the current solution is replaced by the best neighbor. In the case of a first-improving strategy, the current solution moves to the first neighbor whose cost function value improves that of the current solution.

We define a neighborhood associated to any solution x of GMP as formed by all feasible solutions that can be obtained by the modification of $A^{-1}(j)$ for some $j \in N_2$. For each vertex $j \in N_2$, the candidate set $C(j)$ is formed by all vertices in N_1 that can replace the node currently associated with N_2, i.e. $C(j) = \{k \in N_1 \mid x' \text{ is a feasible solution, where } x'_{i\ell} = 1 \text{ if } i = k \text{ and } \ell = j, x'_{i\ell} = 0 \text{ if } i = A^{-1}(j) \text{ and } \ell = j, x'_{i\ell} = x_{i\ell} \text{ otherwise}\}$.

The number of solutions in any neighborhood is bounded by $|N_1| \cdot |N_2|$. We illustrate two neighbor solutions to instance GM-3 in Figure 6. The solution depicted in Figure 6-(b) belongs to the neighborhood of that in Figure 6-(a). It may be obtained from the latter by replacing $A^{-1}(2) = d$ by $A^{-1}(2) = a$, since $a \in C(2)$. The neighbor solution improves the value of the objective function f_1 from 0.5877 to 0.5906.



(a) current solution



(b) neighbor solution

Fig. 6. Two neighbor solutions to GM-3.

The pseudo-code of the local search algorithm `LocalSearchGMP` using a first-improving strategy based on the neighborhood structure defined above is given in Figure 7. The algorithm takes as inputs the solution x^* built by the randomized construction algorithm and its objective function value f^*. We denote by $\Gamma_G(j)$ the nodes adjacent to vertex j in a graph G. Initializations are performed in lines 1-2. The loop in lines 3-32 performs the local search and stops at the first local optimum of the objective function f_1 with respect to the neighborhood defined by the sets $C(j)$. The control variable is initialized at each local search iteration in line 4. The loop in lines 5-31 considers each node j of the image graph G_2. The replacement of the node $i = A^{-1}(j)$ currently associated with j (line 6) by each node belonging to the candidate set $C(j)$ is

investigated in the loop in lines 7-30. The increase in the value of the objective function due to the node similarity contributions is computed in line 8, while that due to the edge similarity contributions is computed in lines 9-19. If the total increase in the objective function value computed in line 20 is strictly positive (line 21), then the current solution and the control variables are updated in lines 22-28. The procedure returns in line 33 the local optimum found. Each local search iteration has time complexity $O(|N_1| \cdot |N_2|^2)$.

```
procedure LocalSearchGMP(G*, f*)
1.   improvement ← .TRUE.;
2.   Build sets C(j), ∀j ∈ N2;
3.   while improvement do
4.        improvement ← .FALSE.;
5.        forall j ∈ N2 while .NOT.improvement do
6.             i ← A^{-1}(j);
7.             forall k ∈ C(j) while .NOT.improvement do
8.                  Δ^v ← 2 · s^v(k, j) − 2 · s^v(i, j);
9.                  Δ^e ← 0;
10.                 forall j' ∈ Γ_{G2}(j) do
11.                      forall i' ∈ Γ_{G1}(i) do
12.                           if i' = A^{-1}(j')
13.                           then Δ^e ← Δ^e + 1 − 2 · s^e((i, i'), (j, j')));
14.                      end_forall;
15.                      forall k' ∈ Γ_{G1}(k) do
16.                           if k' = A^{-1}(j')
17.                           then Δ^e ← Δ^e − 1 + 2 · s^e((k, k'), (j, j')));
18.                      end_forall;
19.                 end_forall;
20.                 Δ ← α/(|N1| · |N2|) · Δ^v + (1 − α)/(|E1| · |E2|) · Δ^e;
21.                 if Δ > 0
22.                 then do;
23.                      improvement ← .TRUE.;
24.                      x*_kj ← 1, x*_ij ← 0;
25.                      A(i) ← A(i) − {j};
26.                      A(k) ← A(k) ∪ {j};
27.                      f* ← f* + Δ;
28.                      Update sets C(j), ∀j ∈ N2;
29.                 end_if;
30.            end_forall;
31.       end_forall;
32.  end_while;
33.  return x*;
end LocalSearchGMP.
```

Fig. 7. Pseudo-code of the local search algorithm.

6 Computational Experiments

Algorithms `RandomizedConstructionGMP` and `LocalSearchGMP` were implemented in C using version 2.96 of the `gcc` compiler. We used an implementation in C of the random number generator described in [20]. All computational experiments were performed on a 450 MHz Pentium II computer with 128 Mbytes of RAM memory, running under version 7.1 of the Red Hat Linux operating system.

6.1 Test Problems

Unlikely as for other problems in the literature, unfortunately there are no benchmark instances available for the problem studied in this work. We describe below a set of ten test instances used in the evaluation of the model and the algorithm proposed in Sections 3 to 5.

Instances GM-2, GM-3, and GM-4 were artificially built, with randomly generated node and edge similarity values in the interval [0,1]. The graphs associated with these three instances and the correspondences maximizing f_1 for $\alpha = 0.5$ are illustrated in Figure 8.

Instances GM-5, GM-8, and GM-9 were also randomly generated [1]. Instance GM-8 was also used in the computational experiments reported in [1]. Instances GM-5 and GM-8 have isolated nodes: two in the image graph G_2 (nodes 0 and 5) of GM-5 and three in the model graph G_1 (nodes 5, 15, and 21) of GM-8. Instances GM-5a and GM-8a are derived from them, by the introduction of additional edges to connect the isolated nodes.

Instances GM-6 and GM-7 were provided by Perchant and Bengoetxea [13] and built from real images. Instance GM-6 was built from magnetic resonance images of a human brain, as depicted in Figure 9. Instance GM-7 was created for the computational experiments reported in [15] from the 2D images given in Figure 10. The image (Figure 10-(a)) was over-segmented in 28 regions (Figure 10-(c)) and compared with a model with only 14 well defined regions (Figure 10-(b)). The model graph G_1 has 14 vertices and 27 edges, while the over-segmented image graph G_2 has 28 vertices and 63 edges. They are presented in Figure 11, together with the solution found by the construction algorithm. Grey levels were used in the computation of node similarities, while distances and adjacencies were used for the computation of edge similarities.

We summarize the characteristics of instances GM-2 to GM-9 in Table 3. For each instance, we first give the number of nodes and edges of the model and

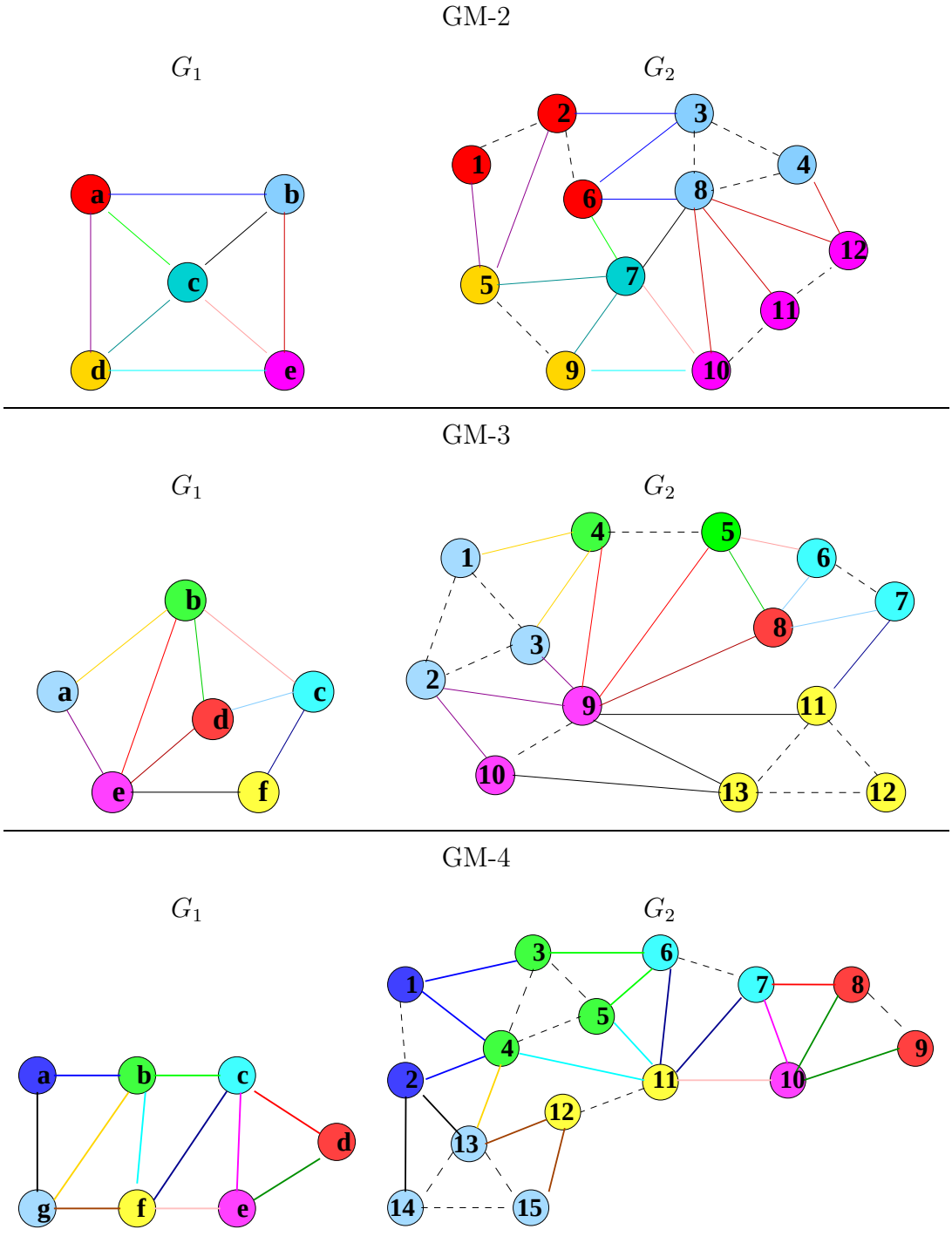


Fig. 8. Instances GM-2, GM-3, and GM-4 and their optimal solutions.

image graphs, together with the maximum value f_1^* for $\alpha = 0.1$, $\alpha = 0.5$, and $\alpha = 0.9$, whenever the latter is known (obtained by complete enumeration of all feasible solutions).

(a) (b) (c)

Fig. 9. Instance GM-6: (a) original image, (b) model, and (c) over-segmented image.

(a) (b) (c)

Fig. 10. Cut of a muscle (instance GM-7): (a) original 2D image, (b) model, and (c) over-segmented image.

6.2 Numerical Results

Table 4 shows the results obtained by algorithm `RandomizedConstructionGMP` on instances GM-2 to GM-9 with $\alpha = 0.1$ and $\alpha = 0.5$. The maximum number of attemps to find a feasible solution was fixed at $MaxTrials = 500$ and the maximum number of feasible solutions built was fixed at $MaxSolutions = 100$. For each instance and for each value of α, we give the number of attempts necessary to find the first feasible solution, the value $f_1^{(1)}$ of the first feasible solution found, the number of attempts necessary to find the best among the first 100 feasible solutions built, the value $f_1^{(100)}$ of the best feasible solution found, and the average computation time per attempt in seconds. Better solutions can be obtained if additional attempts are performed. The computation time per attempt to build a feasible solution is very small, even for the largest instances. The algorithm is very fast and finds the first feasible solution in quite few attempts, except in the cases of the very difficult instances with isolated nodes. However, we notice that even in these hard cases the algorithm managed to find a feasible solution, after 297 (GM-5) and 26 (GM-8) attempts. In all other cases, the construction algorithm found a feasible solution in at most nine iterations (GM-5a).

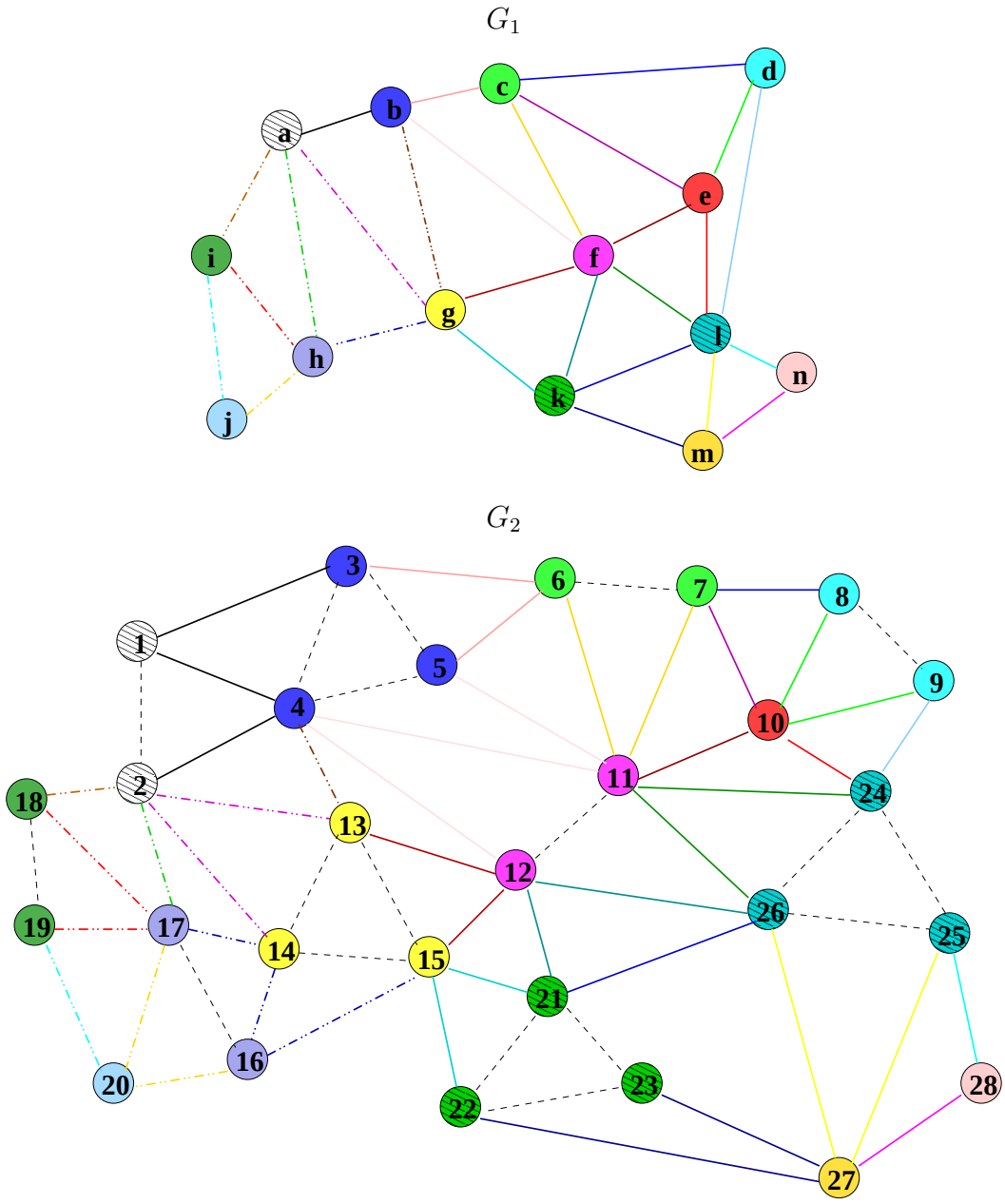


Fig. 11. Instance GM-7 and its optimal solution.

Figure 12 illustrates the solution obtained for instance GM-7 with $\alpha = 0.1$. We notice that 12 out of the 28 vertices of the over-segmented image are correctly identified in this solution (vertices 8, 11, 13, 14, 15, 17, 20, 22, 23, 25, 26, and 27), with respect to the optimal correspondence depicted in Figure 11.

The numerical results obtained by the local search algorithm `LocalSearchGMP` for $\alpha = 0.1$ and $\alpha = 0.5$ are presented in Table 5. For each test instance, we first recall the optimal value f_1^* whenever available and the value $f_1^{(100)}$ of the best solution found by the randomized construction algorithm proposed

Table 3
Characteristics of instances GM-2 to GM-9.

Instance	$\|N_1\|$	$\|E_1\|$	$\|N_2\|$	$\|E_2\|$	f_1^* ($\alpha = 0.1$)	f_1^* ($\alpha = 0.5$)	f_1^* ($\alpha = 0.9$)
GM-2	5	8	12	23	0.2818	0.5098	0.7465
GM-3	6	9	13	25	0.5603	0.6375	0.7147
GM-4	7	11	15	29	–	0.6754	0.7761
GM-5	10	15	30	39	–	–	–
GM-5a	10	15	30	41	–	–	–
GM-6	12	42	95	1434	–	–	–
GM-7	14	27	28	63	–	–	–
GM-8	30	39	100	297	–	–	–
GM-8a	30	42	100	297	–	–	–
GM-9	50	88	250	1681	–	–	–

–: not available

in Section 4. The next three columns report statistics for the local search algorithm: the number of local search iterations until local optimality, the value f_1^{LS} of the best solution found, and the average computation time per iteration in seconds.

We notice that the local search algorithm improved the solution built by the construction algorithm for all test instances. The average improvement with respect to the value of the solution obtained by the construction algorithm was 1.4%.

Figure 13 illustrates the solution obtained by the local search algorithm for instance GM-7 with $\alpha = 0.1$. The comparison with the solution depicted in Figure 11 shows that 13 vertices of the image graph are now correctly associated (vertices 6, 7, 11, 13, 14, 15, 16, 17, 20, 23, 25, 26, and 27), i.e., 13 regions of the over-segmented image are now correctly identified. The value of the objective function increased from 0.1269 to 0.1297.

7 Concluding Remarks

We proposed a 0-1 formulation for the problem of finding the best correspondence between two graphs representing a model and an over-segmented image. Due to inaccuracies of the input data and to the complexity of the problem itself, a precise objective function for this problem is hard to be defined. We described an appropriate objective function to identify the best correspon-

Table 4
Results obtained by the construction algorithm with $MaxTrials = 500$ and $MaxSolutions = 100$ ($\alpha = 0.1$ and $\alpha = 0.5$).

Instance	attempts (1)	$f_1^{(1)}$	attempts (100)	$f_1^{(100)}$	time (s)
GM-2 ($\alpha = 0.1$)	1	0.2646	117	0.2780	$<$
GM-3	1	0.4921	7	0.5213	$<$
GM-4	3	0.5166	206	0.5519	$<$
GM-5	297	0.4908	451	0.4941	$<$
GM-5a	9	0.5106	237	0.5215	$<$
GM-6	1	0.3922	11	0.3927	0.001
GM-7	5	0.1226	198	0.1269	$<$
GM-8	26	0.5020	436	0.5024	0.002
GM-8a	26	0.5339	292	0.5343	0.002
GM-9	1	0.5001	86	0.5002	0.010
GM-2 ($\alpha = 0.5$)	1	0.4986	50	0.5090	$<$
GM-3	1	0.5878	7	0.6020	$<$
GM-4	3	0.6200	206	0.6559	$<$
GM-5	297	0.5038	297	0.5038	$<$
GM-5a	9	0.5044	417	0.5223	$<$
GM-6	1	0.4022	40	0.4045	0.001
GM-7	5	0.3704	34	0.3757	$<$
GM-8	26	0.4999	292	0.5023	0.002
GM-8a	26	0.5176	292	0.5200	0.002
GM-9	1	0.5025	207	0.5031	0.010

$<$: computation time smaller than 10^{-3} second

dences. Additional constraints on the solution space were imposed to make the formulation more strong.

A robust randomized construction algorithm was proposed to build feasible solutions for the graph matching problem. We also proposed a local search algorithm based on a new neighborhood structure to improve the solutions built by the construction algorithm. Computational results were presented to different test problems. Both algorithms are fast and easily found feasible solutions to realistic problems with up to 250 nodes and 1681 edges in the graph representing the over-segmented image. The local search algorithm

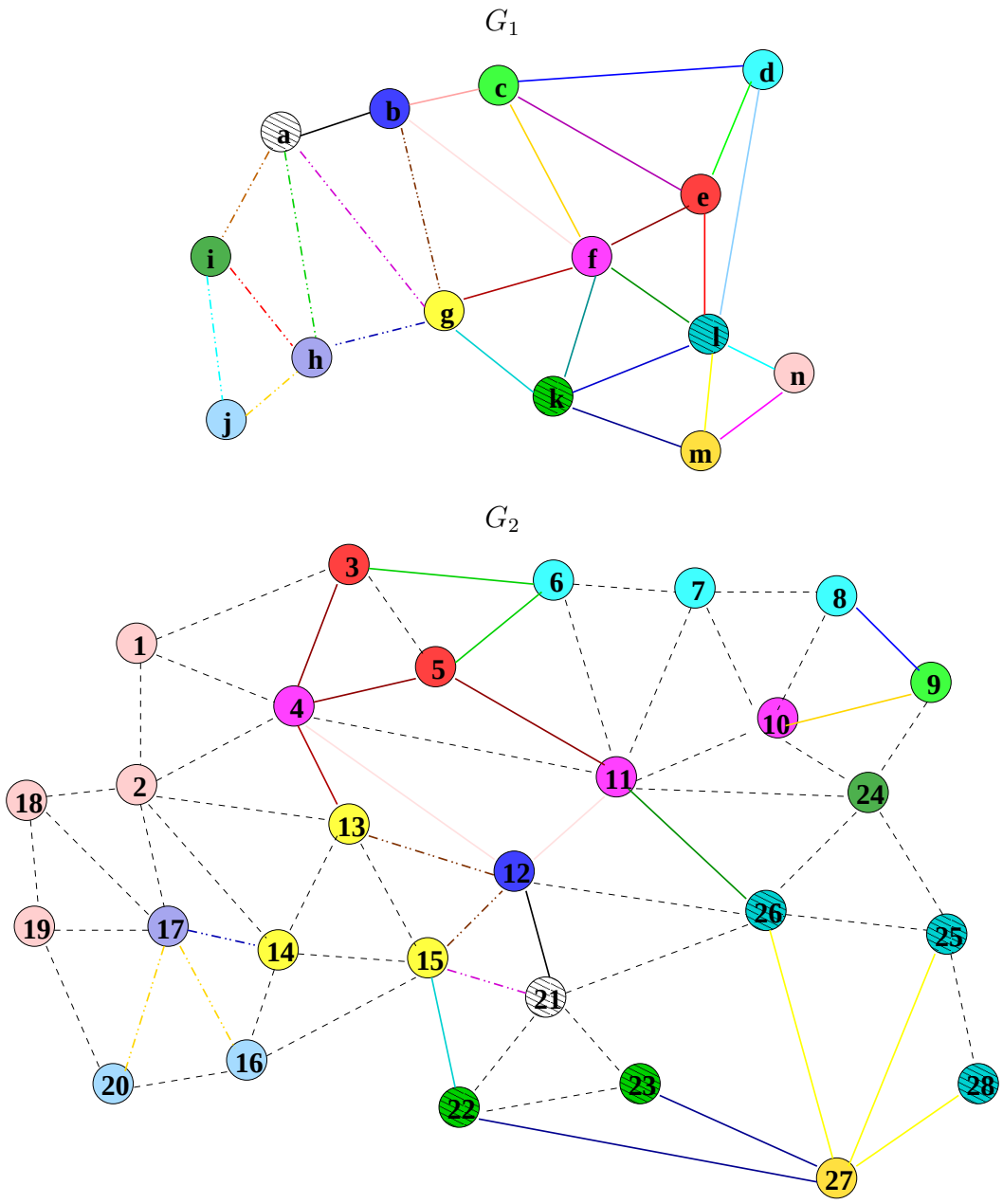


Fig. 12. Solution obtained by the construction algorithm for instance GM-7 with $MaxTrials = 500$ and $MaxSolutions = 100$ ($\alpha = 0.1$).

consistently improved the solutions found by the construction heuristic. Both algorithms can be easily adapted to handle more complex objective function formulations.

Besides the quality of the solutions found, the randomized algorithm may also be used as a robust generator of initial solutions for population methods such as the genetic algorithm described in [15], replacing the low quality randomly generated solutions originally proposed. The construction and lo-

Table 5
Results obtained by the local search algorithm ($\alpha = 0.1$ and $\alpha = 0.5$).

			Local search		
Instance	f_1^*	f_1^{100}	Iteration	f_1^{LS}	time (s)
GM-2 ($\alpha = 0.1$)	0.2818	0.2780	2	0.2818	$<$
GM-3	0.5603	0.5213	3	0.5371	$<$
GM-4	–	0.5519	4	0.5621	$<$
GM-5	–	0.4941	11	0.5023	0.002
GM-5a	–	0.5215	5	0.5238	0.002
GM-6	–	0.3927	235	0.3952	0.019
GM-7	–	0.1269	5	0.1297	$<$
GM-8	–	0.5024	69	0.5047	0.013
GM-8a	–	0.5343	86	0.5362	0.013
GM-9	–	0.5002	430	0.5017	0.146
GM-2 ($\alpha = 0.5$)	0.5098	0.5090	2	0.5098	$<$
GM-3	0.6375	0.6020	3	0.6131	$<$
GM-4	0.6754	0.6559	5	0.6754	$<$
GM-5	–	0.5038	17	0.5212	0.001
GM-5a	–	0.5223	6	0.5284	0.002
GM-6	–	0.4045	204	0.4090	0.017
GM-7	–	0.3757	7	0.3771	$<$
GM-8	–	0.5023	84	0.5086	0.012
GM-8a	–	0.5200	83	0.5266	0.012
GM-9	–	0.5031	399	0.5087	0.123

–: not available

$<$: computation time smaller than 10^{-3} second

cal search algorithms can also be put together into an implementation of the GRASP metaheuristic [17]. Future developments reporting these investigations and more extensive tests and computational experiments on real data will be reported elsewhere.

Acknowledgments: We are grateful to one anonymous referee for several remarks which contributed to improve the revised version of this work.

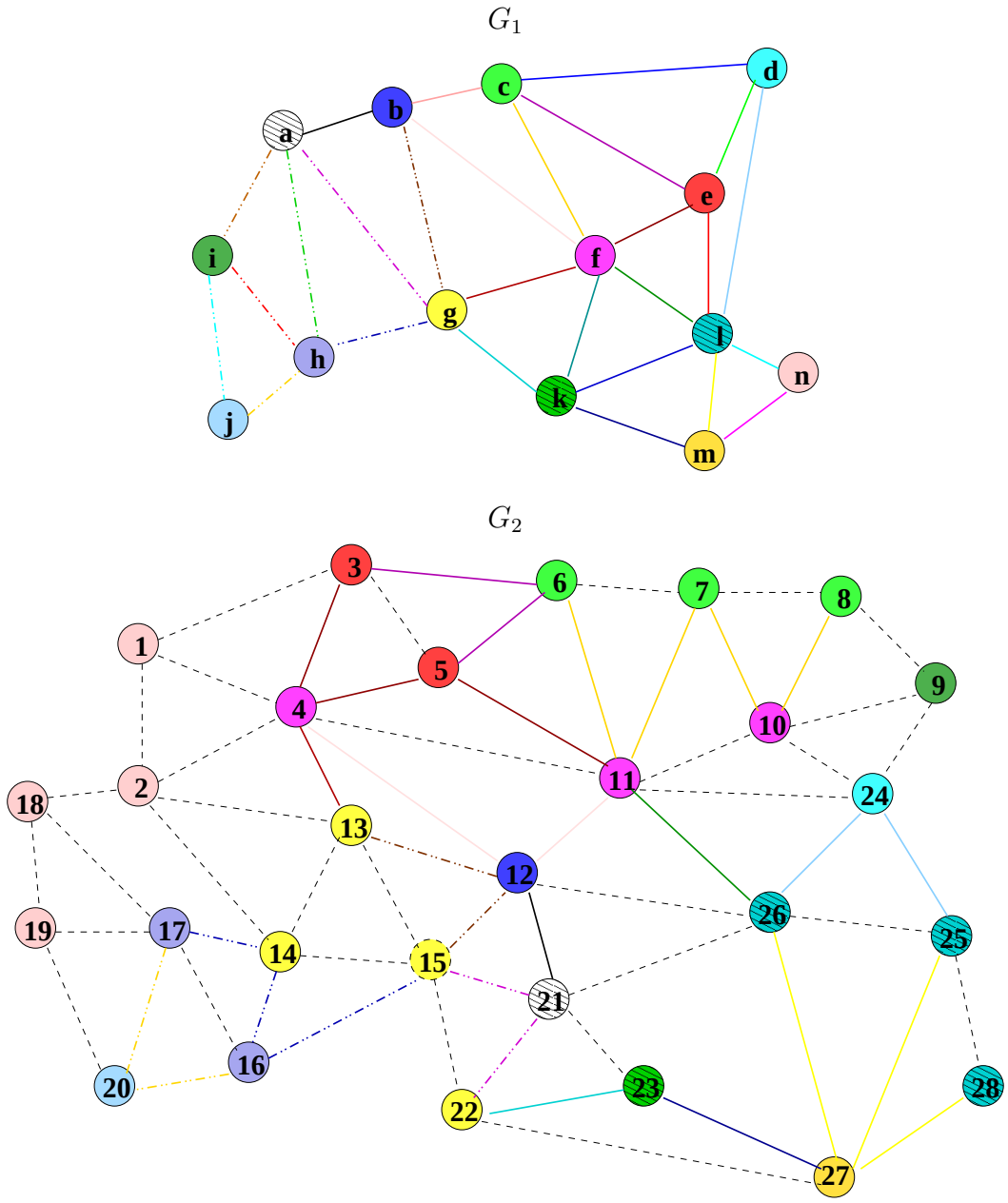


Fig. 13. Solution obtained by the local search algorithm for instance GM-7 ($\alpha = 0.1$).

References

[1] E. Bengoetxea, P. Larranaga, I. Bloch, A. Perchant, and C. Boeres. Inexact graph matching by means of estimation distribution algorithms. *Pattern Recognition*, 35:2867-2880, 2002.

[2] I. Bloch. Fuzzy relative position between objects in image processing: a morphological approach. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 21:657–664, 1999.

[3] I. Bloch. On fuzzy distances and their use in image processing under imprecision. *Pattern Recognition*, 32:1873–1895, 1999.

[4] I. Bloch, H. Maître, and M. Anvari. Fuzzy adjacency between image objects. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 5:615–653, 1997.

[5] K.P. Chan and Y.S. Cheung. Fuzzy-attribute graph with application to chinese character recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 22:402–410, 1992.

[6] A.D.J. Cross and E.R. Hancock. Relational matching with stochastic optimization. In *International Conference on Computer Vision*, pages 365–370, 1995.

[7] A.D.J. Cross, R.C. Wilson, and E.R. Hancock. Inexact graph matching using genetic search. *Pattern Recognition*, 30:953–970, 1997.

[8] Y. El-Sonbaty and M. A. Ismail. A new algorithm for subgraph optimal isomorphism. *Pattern Recognition*, 31:205–218, 1998.

[9] A. W. Finch, R. C. Wilson, and E. R. Hancock. Symbolic Graph matching with the EM algorithm. *Pattern Recognition*, 31:1777–1790, 1998.

[10] H. Maître, I. Bloch, H. Moissinac, and C. Gouinaud. Cooperative use of aerial images and maps for the interpretation of urban scenes. In *Automatic Extraction of Man-Made Objects from Aerial and Space Images Proceedings*, pages 297–306, Ascona, 1995.

[11] B.T. Messmer and H. Bunke. A decision tree approach to graph and subgraph isomorphism detection. *Pattern Recognition*, 32:1979–1998, 1999.

[12] H. Moissinac, H. Maître, and I. Bloch. Markov random fields and graphs for uncertainty management and symbolic data fusion in a urban scene interpretation. *EUROPTO Conference on Image and Signal Processing for Remote Sensing*, 2579:298–309, 1995.

[13] A. Perchant. *Morphisme de graphes d'attributs flous pour la reconnaissance structurelle de scènes*. Doctorate thesis, École Nationale Supérieure des Télécommunications, 2000.

[14] A. Perchant and I. Bloch. A new definition for fuzzy attributed graph homomorphism with application to structural shape recognition in brain imaging. In *16th IEEE Instrumentation and Measurement Technology Conference Proceedings*, pages 402–410, 1999.

[15] A. Perchant, C. Boeres, I. Bloch, M. Roux, and C.C. Ribeiro. Model-based scene recognition using graph fuzzy homomorphism solved by genetic algorithm. In *2nd IAPR-TC-15 Workshop on Graph-based Representations*, pages 61–70, Haindorf, 1999.

[16] H.S. Ranganath and L.J. Chipman. Fuzzy relaxaton approach for inexact scene matching. *Image and Vision Computing*, 10:631–640, 1992.

[17] M.G.C. Resende and C.C. Ribeiro. "Greedy randomized adaptive search procedures". *Handbook of Metaheuristics* (F. Glover and G. Kochenberger, eds.), pages 219-249, Kluwer, 2002.

[18] A. Rosenfeld. *Fuzzy sets and their applications*. Academic Press, 1975.

[19] A. Rosenfeld, R. Hummel, and S. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man and Cybernetics*, 6:420–433, 1976.

[20] L. Schrage. A more portable FORTRAN random number generator. *ACM Transactions on Mathematical Software*, 5:132-138, 1979.

[21] M. Singh and A. C. S. Chaudhury. Matching structural shape descriptions using genetic algorithms. *Pattern Recognition*, 30:1451–1462, 1997.

[22] A.K.C. Wong, M. You, and S.C. Chan. An algorithm for graph optimal monomorphism. *IEEE Transactions on Systems, Man and Cybernetics*, 20:628–636, 1990.

[23] E.K. Wong. Model matching in robot vision by subgraph isomorphism. *Pattern Recognition*, 25:287–303, 1992.