Pyramid and Interior

Jean-Gérard Pailloncy

Laboratoire LVR, IUT de Bourges,
F-18000 Bourges, France
pailloncy@ifrance.com

Abstract. A Pyramid: a hierarchy of Region Adjacency Graphs (RAG) has only one type of edge "to be neighbor with". We want to add some new types of edge. The relation "to be inside" is already present as a special case of the neighborhood relation. We detail the difference between a neighborhood relation and an interior relation. Then we show that some new types of interior relations are compatible with the process of building a pyramid.

1 Introduction

The Region Adjacency Graph (RAG) has been studied for a long time. Some recent works presents different views of this topic, like the pyramid of dual graph of Kropatsch [1], or the discrete map of Braquelaire and Brun [2], etc. In GBR'99 [8], Bunke and *al.* in [4] presents some works about *type-n* graphs, and Deruyver and *al.* in [5] about semantic graphs. We have many discussions about the possible extensions of the RAGs. [4] and [5] present different kind of graphs with many different types of edges.

We will present some topics of our discussion in Section 2. Then, in Section 3 we will detail the structure of the RAG followed by an answer for a limited case in Section 4. And we will end our work by some perspectives.

2 Discussion

An image is a grid of pixel that may be seen as graph. In this graph, the relation between two nodes is the neighborhood of two pixels. We have only one relation "*to be neighbor with*". In fact, with the 4-connexity we have four types of relations "*to be neighbor on the top with*", "*…on the bottom…*", "*…on the left…*", "*…on the right…*", but we use them very rarely. The regularity of the grid gives us a kind of metric on the image graph, especially when we use the four different relations. During the merging process of the graph to extract from the grid graph some semantic information, we use only the relation "*to be neighbor with*". We have introduced the relation "*to be included in*", when after a fusion an object become included in an other one. In most of the works, the relation "*to be included in*" is managed more as a trick than by it-self. Indeed it is only a special case of "*to be neighbor with*" and is represented by a loop edge in [1].

F.J. Ferri et al. (Eds.): SSPR&SPR 2000, LNCS 1876, pp. 437-446, 2000.

2.1 The *type-n* Graph

Bunke and *al.* [4] have proposed, in the field of graph matching, to split the matching into 3 variants named *type-0*, *type-1*, *type-2* with a increasing strictness of the matching.

The *type-0* use five relations: *disjoint*, *adjacent*, *intersection*, *included*, *including*. The *type-1* imposes that two objects have the same relation along each ax. These relations along each ax defined if an object is *on the top, on the bottom, on the left, on the right* of an other object. The *type-2* adds 169 new relation: a set of 13 relations by axes (*before, intersection*, *included*, *equal*, *adjacent*, *end*, *start*, and the opposite relations). We are able to follow this approach to build *type-0*, *type-1* and *type-2* graphs.

A RAG is *type-0*, because it uses three of the five defined relations for a *type-0* matching. *Disjoint* is not represented, *adjacent* is "to be neighbor with", *including* is the special case of "*to be neighbor with*", *included* is the opposite of *including* and the relation *intersection* has no meanings in a RAG (a segmentation is a partition). The grid graph is *type-1*. A *type-2* graph may be defined, however it is sometime hard to define the relations between two complex shapes.

It is possible to define a merging process to build a hierarchy of such graphs. We just need the merging table. Given two objects *A* and *B* merged into *C*, and an object *X*, the merging table give us the relation *CX* from the relations *AB*, *AX*, *BX*.

The relations *type-1*, *type-2* have some metric information. In the grid graph the relation "*to be neighbor with*" is linked to "*to be neighbor, by a distance of 1 pixel, with*". The RAG has only one type of relation and includes no metric information.

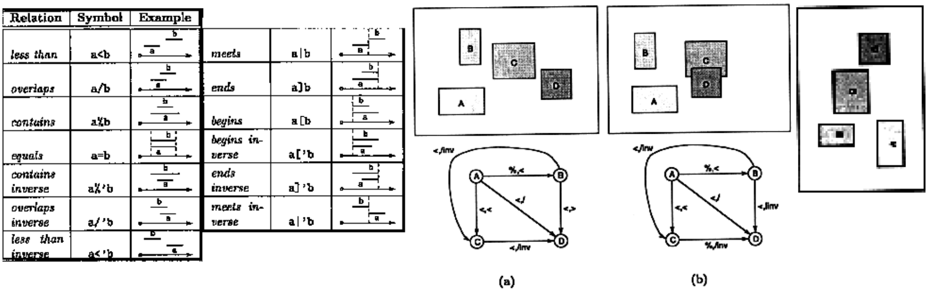

Fig. 1. Orientation and *type-2*. We have on the left the set of possible *type-2* relations as in [4]. On the right, we have two drawings with their associated graphs. An algorithm is detailed in [4] to extract the largest common sub-graph, and so, to track an object in a known environment. The main axe must be known. If we turn the drawing by 90°, the relation between the objects stay the same, but the graph is very different.

For the *type-2*, given two objects there are only nine possible relations (over 13) by axe and seven if the two objects have the same size. There are 81, 63 or 49 possible relations (over 169). We may see these 81 relations as a 9x9 grid around an object in which we locate the other one. We do not have a full metric information, but the richness is far better than in the simple case "to be/not to be neighbor with". The *type-2* relation allows us to define a graph with a richer information than the RAG and the grid graph. However the choice of the axis is not a simple question, we often need some outside information. See Fig. 1.

2.2 The Semantic Graph

An other way to analyze an image is the partial matching between the image and some regions already segmented of an *a priori* model imbedded in a semantic graph. The semantic graphs embed two important information: the structural information and the contextual information. The structural information is defined by the unary constraints of each node. These constraints define, for example, the topology of the region. The contextual information is defined by the binary relations of the edges. They define, for example, the spatial relation of two regions. These graphs are very smart to encode the complex constraint and they allow us to compute the influence of one node overall graph. See Fig. 2.

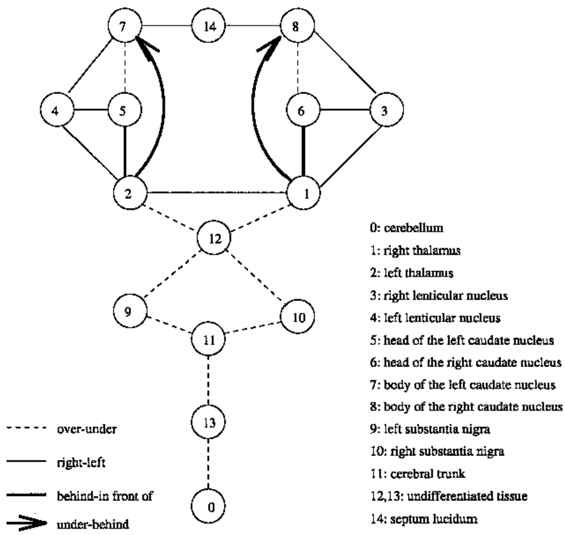

Fig. 2. Semantic Graph. Example of a semantic graph build from a human brain description. We have four types of relations. This graph is reproduced from [9].

2.3 RAG Weakness

After this short presentation of two types of graph that has a richer set of edge's type than the RAG, we will present some weakness of the RAG. See Fig. 3. The RAG encodes o relation about the included regions, but from the human perception point of view there is a big difference.

In [3], we have shown that the theories [1,2,6] are equivalent. We will not detail the addition of new edge for each of these theories. In addition, it has been shown that we can represent the pyramid, as a labeled graph [7]. Therefore, we have a smaller structure to encode the multi-levels segmentation, and an easy merging process. For all these reasons, we want to enhance the RAG by the introduction of new edge type. However, is it possible to extend the types of edge in a RAG-like graph, while preserving the easiness of the building of the hierarchy?

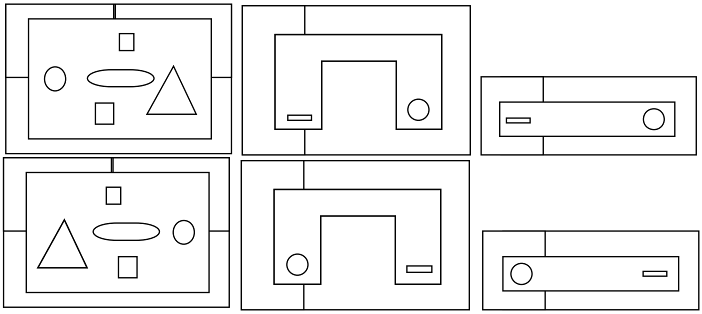

Fig. 3. At left, the 5 objects included in the central region are the same, only their relative positions change. No RAG-like structure can represent this kind of differences. The two other cases are other examples where from a human point of view there is a big difference between the two drawing, but the RAG can not catch it.

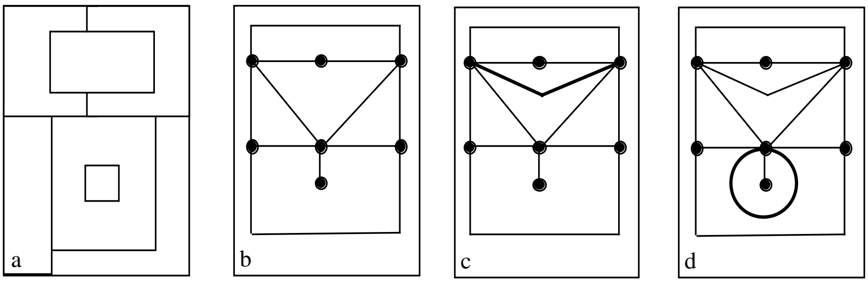

Fig. 4. (a) A partition, the face graph, *i.e.*, the graph of the element's border. (b) A simple region adjacency graph. (c) A RAG with double edges (the thick edge). We must add multiple edges to be able to recover the face graph by duality when the common border of two elements is not connected. (d) A RAG with multiple edges and a loop (the thick edge). We must add a loop to be able to recover the face graph by duality when an element of the partition is included in an other.

3 Region Adjacency Graph

A Region Adjacency Graph is a graph build on a partition of the 2D space where each node is an element of the partition, and each edge is an adjacency relation between two elements. The Fig. 4b shows us a simple Region Adjacency Graph (RAG) obtained from the partition in Fig. 4a.

3.1 Two Types of Edges in RAG

When we merge some nodes, two new situations appear. Two extensions have been developed when two elements have a non-connected common border (Fig. 4b), and when a region is surrounded by an other (Fig. 4c).

In a RAG or in the Frontier-Region Graph [6], there is only one type of edge "*to be neighbor with*". However, when a region is surrounded by another, the relation between the inside and the outside region is more than "*to be neighbor with*": it becomes "*to be included in*". In the Discrete Map, the relation "*to be include in*" is encoded in a separate structure, an inclusion tree defined by a function that return for each element the father, *i.e*. the surrounding one.

We name the edges "*to be neighbor with*": neighborhood edges and the *edges "to be inside of*": interior edges.

3.2 Transition of Types

By merging the nodes, we build a pyramid: a hierarchy of graphs. During a merge of two nodes, the only one transition may appear is a couple of neighborhood edges (Fig. 5a) that becomes an interior edge (Fig 5b).

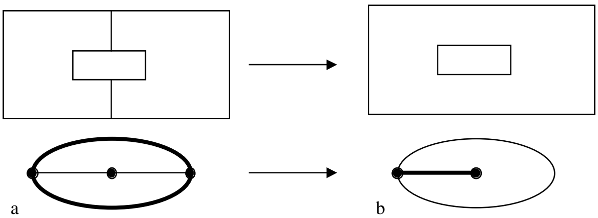

Fig. 5. Merge of a graph that produces an interior edge (the thick one).

We will now detail with few math. The support S of an image is, in a generic way, a finite partition of the 2D Euclidean space with only one infinite element the background.

Let A, B and C be three elements of the partition. We note:

$A \leftrightarrow B$ when A is a neighbor of B (but not included in B). **(1)**

$A \rightarrow B$ when A is included in B. **(2)**

$B \oplus C$ the merge of B and C, *i.e.*, $B \cup C$ when B and C are adjacent. **(3)**

The Fig. 6 presents 4 drawings of the 4 cases that appear during a merge and are example of the equations (4), (5) and (6).

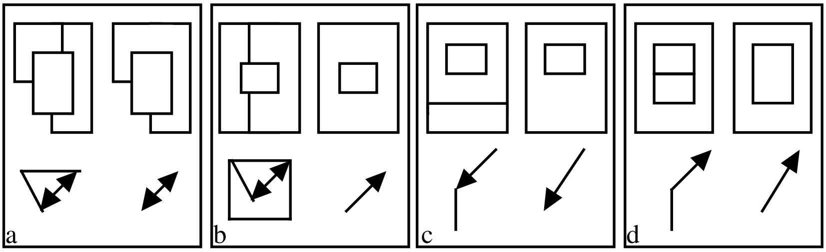

Fig. 6. (a) and (b) shows an example for (4), (c) for (5) and (d) for (6). For each drawing, the top shows a partition example and the bottom the corresponding graph.

We have the following relations (Fig. 6):

$$A \leftrightarrow B \text{ and } B \leftrightarrow C \Rightarrow A \leftrightarrow B \oplus C \text{ or } A \rightarrow B \oplus C. \tag{4}$$

$$A \rightarrow B \text{ and } B \leftrightarrow C \Rightarrow A \rightarrow B \oplus C. \tag{5}$$

$$A \rightarrow B \text{ and } A \leftrightarrow C \Rightarrow A \oplus C \rightarrow B \text{ (and } C \rightarrow B \text{ before the merge)} \tag{6}$$

In (6), if A is included in B, and we merge A and C, C must be included in B. For the topological interior, this is obvious, but we will see some other possibilities.

4 New Interior Edge

In the common way, when we say something is inside another, we mean more things that the interior edge we discuss before. We use only a topological meaning of interior. In Fig. 7, we present three different cases where A is in B.

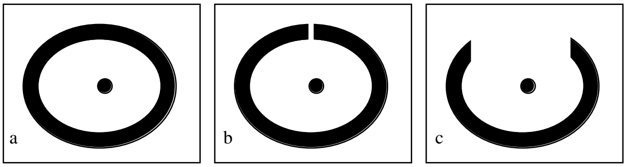

Fig. 7. A is the dot, B is the ellipse. (a) A is inside B by the topological way, (b) by a "mechanical" way and in (c) by an "optical" way.

The topological interior relation has already been studied for years. We will study the third one, the optical interior relation. A is "optically" inside B, if A is topologically inside the smallest convex shape surrounding B. A is "mechanically" inside B, if A can not be moved outside B. We will further extend the result to the "mechanical" relation.

Let A, B be two elements of the partition. We note:

$A \vee B$ if A is "optically" in B, and A is not included in B. **(7)**

Let Cvx(A) be the smallest convex surrounding A. **(8)**

We have: $$A \vee B \Leftrightarrow A \rightarrow Cvx(B) \quad \textbf{(9)}$$

We want only non-ambiguous relation. We will say that A is "optically" in B only if A is not already included topologically in B.

4.1 Merging Process

First, suppose we have a RAG with three types of edges: the interior one and the neighborhood one as define in (1), and a third one the "optical" interior one. We will study if this kind of edge is compatible with (1). We will then study the creation of such edges.

The Fig. 8 shows the new cases that appear during a merge. We have the same relations (4) and we have the new following relation:

$$A \vee B \text{ and } B \leftrightarrow C \Rightarrow A \vee B \oplus C \text{ (Fig. 8a)} \quad \textbf{(10)}$$

Proof. $A \vee B \Rightarrow A \rightarrow Cvx(B)$, $C \subset Cvx(C)$, $A \rightarrow Cvx(B) \oplus Cvx(C) \subset Cvx(B \oplus C)$. ♦

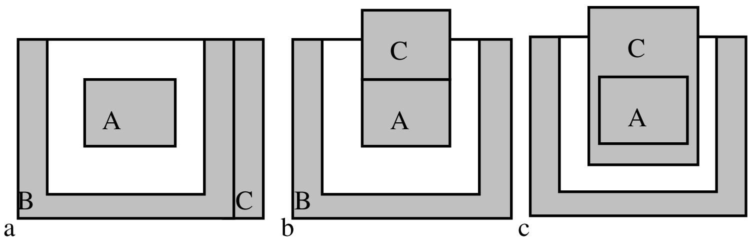

Fig. 8. Merging with "optical" interior edges.

But can not say directly something about $A \oplus C$ and B. See Fig. 8b. In this case, we do not have obviously $C \vee B$ before the merge. However, we have:.

$$A \vee B \text{ and } C \vee B \text{ and } A \leftrightarrow C \Rightarrow A \oplus C \vee B \quad \textbf{(11)}$$

$$A \vee B \text{ and } B \rightarrow C \Rightarrow A \vee B \oplus C \quad \textbf{(12)}$$

Proof. $A \vee B \Rightarrow A \rightarrow Cvx(B) \subset Cvx(C)$, $A \vee C$. ♦

$$A \vee B \text{ and } C \rightarrow B \Rightarrow A \vee B \oplus C \quad \textbf{(13)}$$

$$A \vee B \text{ and } C \rightarrow A \Rightarrow A \oplus C \vee B \quad \textbf{(14)}$$

We have the same problem for $A \vee B$ and $A \rightarrow C$: we do not have $C \vee B$ as for Fig. 8b. See Fig. 8c for an example. However, we have:.

$$A \vee B \text{ and } C \vee B \text{ and } A \rightarrow C \Rightarrow A \oplus C \vee B \quad \textbf{(15)}$$

In an algorithmic point of view, we need to check two new conditions before doing a merge to verify that both merged regions are included in the same one when $A \vee B$ and $A \rightarrow C$ or $A \leftrightarrow C$. If the condition is not verify, we must suppress the edge $A \vee B$ during the merge.

We have now a graph, with three types of edges: neighborhood, topological interior and optical interior. And with a small addition, the merging process stays the same. An important change is that during a merge an edge not member of the merged set may be suppressed. A new side-effect appears. Before, the only edges remove from the pyramid are the edge in the merged set, the set of edges that connected two vertices that would be merged during the merging process.

Even if this change is important in the concept of the pyramid, it has little impact overall structure, because there is no transition between topological interior edge and optical interior one. The merge may appear only between two neighbor vertices, we can not merge an optical interior edge.

The optical interior edge are on an other level, they are create by user interaction, and stay in the structure without truly modify it. Sometime they are suppressed to keep the structure coherent. However, an other way exists.

4.2 New Region

We will now relax the constraint on the type of edge used in a merging process. We want to allow the merge to occur between any two vertices linked by an edge, even if this edge is an optical interior one. This will break some topology stuff. The segmentation in this way will not stay a partition, because some elements may be not connex.

For most of the new case, the merge is possible without any trouble.

$$A \prod B \text{ and } B \vee C \Rightarrow A \prod B \oplus C \quad \textbf{(16)}$$

$$A \vee B \text{ and } B \vee C \Rightarrow A \vee B \oplus C \text{ or } B \vee A \oplus C \text{ or } C \vee A \oplus B \quad \textbf{(17)}$$

$$A \vee B \text{ and } C \vee B \Rightarrow A \vee B \oplus C \text{ or } B \vee A \oplus C \text{ or } C \vee A \oplus B \quad \textbf{(18)}$$

$$A \vee B \text{ and } A \vee C \Rightarrow A \vee B \oplus C \quad \textbf{(19)}$$

We got a problem only when $A \vee B$ and $A \vee C$ and we want to merge A with B or C. The Fig. 9 shows an example of this new situation. We must have $B \vee C$ or $C \vee B$ to merge A without loosing an edge. This case has been already studied in (17). The new situation of Fig. 9 is similar to the Fig. 8c one. The merging process stays simple but in two situations (Fig. 9 and Fig. 8c) one edge must disappear.

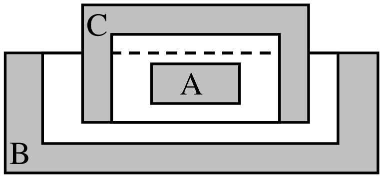

Fig. 9. When $A \vee B$ and $A \vee C$, we can not merge A with B or C, without breaking one edge.

For the mechanical interior edge, the merging process is pretty the same than for the optical interior edge. We do not detail it.

5 Conclusion

By the relaxation of the connexity constraint, we have a new definition of segmentation. The constraint is not fully removed, it is moved from the structure to the merging process. We have shown that some extensions of the edge's types changes a little the merging process. So, during the merging process most of the merge will occur when $A \rightarrow B$ or $A \leftrightarrow B$, but when an optical interior edge become meaningful, we add it to the structure. The merging process may then follow to different may: the classic one (Sec. 4.1) where only adjacent elements may be merged, or the new one (Sec. 4.2) where any edges may be merged, and some non-connected region may appear. The representation of non-connected region may a point of interest. For example, in Fig. 3 left, the ellipse is optically in the set of circle, rectangles and triangles. We have a classic subjective region in Fig. 10.

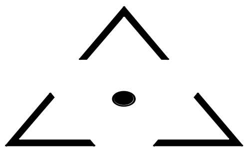

Fig. 10. Subjective region as an example of possible use of optical interior edge.

After a discussion about the interest of adding some new type of edges to a RAG-like structure, we have presented an extension of interior relation. This extension was made with a minimum of changes of the merging process. This point of view has allowed us to consider some new pyramid, *i.e.*, without the connexity constraint, but with the same merging process. The two next steps will be to study the extraction of these new edge types, and in a more theoretical side to study the properties that must respect the edges, in general, to be compatible with this merging process.

References

1. W. G. Kropatsch, *Building Irregulars Pyramids by Dual Graph Contraction*, IEE-Proc. Vis., Image and Signal Proc. Vol. 142 N° 6, 12/95, pp. 366-374.
2. J. P. Braquelaire, L. Brun, Image Segmentation with Topological Maps and Inter-pixel Representation, Journal of Visual Communication and Image Representation, 9, 1998, pp. 62-79.
3. J.-G. Pailloncy, PhD "Contribution de la théorie des Grapohes à l'analyse d'image", INSA de Lyon, Lyon, 01/12/1999
4. K. Shearer, H. Bunke, S. Venkatesh, D. Kieronska, *Efficient Graph Matching for Video Indexing*, Proc. of GbR'97, 1st IAPR Int. Workshop on Graph based Representations, Springer Verlag, Computing Suppl., 12, Lyon, France, 1998, ISBN 3-211-83121-5, pp. 53-62.
5. A. Deruyver, Y. Hodé, Constraint Satisfaction Problem with Bi-level Constraint : Application to Interpretation of Over-segmented Images, Artificial Intell., 93, 1997, pp. 321-335.
6. J.-G. Pailloncy, J-.M .Jolion, *The Frontier-Region Graph : A topologically consistent representation for image analysis, suitable for parallel computers*, Proc. of GbR'97, 1st IAPR Int. Workshop on Graph based Representations, Spring Verlag, Computing Suppl., 12, Lyon, France, 1998, ISBN 3-211-83121-5, pp. 123-134.
7. W. G. Kropatsch, *Equivalent Contraction Kernels and The Domain of Dual Irregular Pyramids*, Technical report PRIP-TR-42, Institute f. Automation 183/2, Dept. for Patt. Recognition and Image Proc., TU Wien, Austria, 17/12/95.
8. J.-M. Jolion, W. G. Kropatsch, *Graph based Representation*, Jean-Michel Jolion and Walter Kropatsch eds., Proc. of GbR'97, 1st IAPR Int. Workshop on Graph based Representations, Spring Verlag, Computing Suppl., 12, Lyon, France, ISBN 3-211-83121-5, 1998.
9. A. Deruyver, Y. Hodé, *Constraint Satisfaction Problem with Bi-level Constraint : Application to Interpretation of Over-segmented Images*, Artificial Intell., 93, 1997, pp. 321-335.