Graph matching for visual object recognition

HORST BUNKE*

Department of Computer Science, University of Bern, Neubrückstrasse 10, CH-3012 Bern, Switzerland

Received 19 June 1999; accepted 16 March 2000

1. INTRODUCTION

Graphs are a powerful and universal tool widely used in information processing. In pattern recognition and machine vision, object similarity is an important issue. Given a database of known objects (models) and an unknown input object, the task is often to retrieve one or several models from the database that are similar to the unknown input. If graphs are used for object representation, this task turns into the problem of determining the similarity of graphs, which is also known as graph matching.

Graph matching has found numerous applications in pattern recognition and machine vision. They include character recognition (Lu et al., 1991; Rocha and Pavlidis, 1994), graphical symbol recognition (Lee et al., 1990; Jiang et al., 1999), shape analysis (Lourens, 1998), three-dimensional object recognition (Wong, 1992), and others. Additional applications from the field of artificial intelligence are case-based reasoning (Poole, 1993), machine learning (Cook and Holder, 1994; Messmer and Bunke, 1996), and planning (Sanders et al., 1997).

In this paper, we first introduce some basic concepts and review recent theoretical results and algorithms. Then a novel concept, the mean of a set of graphs, is introduced and application examples are given in Section 3. Finally, conclusions will be provided in Section 4.

2. BASIC CONCEPTS

We consider labeled graphs with directed edges. Formally, such a graph is a 4-tuple \( g = (V, E, \alpha, \beta) \) where \( V \) is the finite set of vertices, \( E \subseteq V \times V \) is the set of

*E-mail: bunke@iam.unibe.ch
edges, $\alpha : V \rightarrow L_V$ is the node labeling function assigning a label from some node label set $L_V$ to each node, and $\beta : E \rightarrow L_E$ is the edge labeling function assigning a label from some edge label set $L_E$ to each edge. We say that $g' = (V', E', \alpha', \beta')$ is a subgraph of $g$, if $V' \subseteq V$, $E' = E \cap (V' \times V')$, $\alpha'(v) = \alpha(v)$ for all $v \in V'$, and $\beta'(e) = \beta(e)$ for all $e \in E'$. Two graphs are isomorphic if there exists a bijective mapping $f : V \rightarrow V'$ that preserves the structure of the edges and all labels. If $f$ is a graph isomorphism between $g$ and $g'$, and $g'$ is a subgraph of $g''$, then $f$ is called a subgraph isomorphism from $g$ to $g''$. A common subgraph of two graphs, $g$ and $g'$, is a graph $g''$ such that there are subgraph isomorphisms from $g''$ to $g$, and from $g''$ to $g'$. A common subgraph $g''$ of $g$ and $g'$ is called a maximum common subgraph if there is no other common subgraph of $g$ and $g'$ that has more nodes than $g''$.

For an illustration of the concepts introduced in the last paragraph, look at Fig. 1. (In Fig. 1 there are no edge labels, for the purpose of simplicity. The numbers inside the circles are node names, while the letters outside the circles denote node labels.) Graphs $g_1$ and $g_2$ are isomorphic to each other. There is a subgraph isomorphism from $g_1$ (and $g_2$) to $g_3$, but not to $g_4$. Graph $g_1$ (and $g_2$) is a maximum common subgraph of $g_1$ and $g_3$. The maximum common subgraph of $g_3$ and $g_4$ is defined by nodes 7 and 8 in $g_3$ (or, equivalently, by 11 and 12 in $g_4$).

Graphs are suitable for the representation of structured objects. Typically, nodes are used to represent object parts, or individual objects that are part of a complex scene, while edges model relationships between object parts or objects. The relations modeled by the edges can be of various type, for example, spatial, temporal, causal, a.s.o. For certain applications in machine vision, invariance properties, such as invariance under translation, rotation or scaling, are desired. Notice that these invariance properties are automatically guaranteed by a graph representation (see the definitions given above). On the other hand, there exist applications where invariance properties are not useful. An example is character recognition, where rotational invariance implies that we can’t distinguish between ‘6’ and ‘9’. However, invariance properties inherently present in a graph representation can be easily disabled by selecting appropriate node or edge labels. For example, if absolute $x$- and $y$-coordinates of features in the image plane are used as labels, then the representation is no longer invariant under translation, rotation, and scaling.

Figure 1. An example showing four graphs $g_1, \ldots, g_4$ (see text).