Error-Correcting Isomorphisms of Attributed Relational Graphs for Pattern Analysis

WEN-HSIANG TSAI, STUDENT MEMBER, IEEE,
AND KING-SUN FU, FELLOW, IEEE

Abstract—**The pattern deformational model proposed by Tsai and Fu [11] is extended so that numerical attributes and probability or density distributions can be introduced into primitives and relations in a nonhierarchical relational graph. Conventional graph isomorphisms are then generalized to include error-correcting capability for matching deformed patterns represented by such attributed relational graphs. An ordered-search algorithm is proposed for determining error-correcting isomorphisms. Finally, a pattern classification approach using graph isomorphisms is described, which can be considered as a combination of structural and statistical techniques.**

I. Introduction

RELATIONAL GRAPHS are used in syntactic pattern recognition to represent the structural information of given patterns [1]. The nodes in a relational graph denote subpatterns and primitives, and the branches between two nodes represent the relations between subpatterns and primitives. Other terms, such as relational structures [2], webs [3], [26], and labeled graphs [4], are also adopted for such structural representations. As an example, given the scene in Fig. 1(a), a relational graph to represent the structure of the scene is shown in Fig. 1(b). Relational graphs or their simpler versions (without branch labels or even node labels) are found in such applications as scene analysis, chemical structure descriptions, relational database systems, network representations, switching theory, etc.

One way to recognize the structure of a given unknown pattern is to transform this pattern into a structural representation using a relational graph and then to match this graph with those which represent structures of prototype patterns. Such graph matching, or *graph isomorphism*, is necessary and important when a grammatical analysis in terms of parsing is not applicable. This case happens, for example, when training samples are too few to infer pattern grammars or when each pattern itself could be regarded as a prototype of the pattern class. There is a significant amount of literature on graph isomorphisms [2], [5]–[9], but most of these investigations deal with unlabeled graphs except for those by Barrow *et al.* [2] in which several labeled relational graph matching methods are discussed and compared.

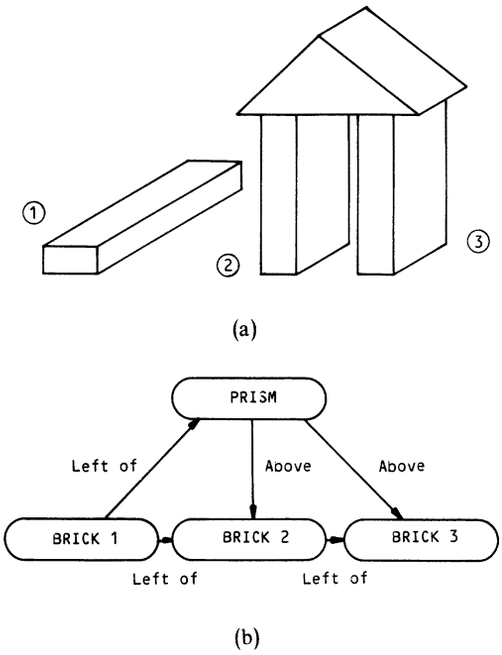


Fig. 1. (a) A scene. (b) A relational graph for (a).

In some practical applications a certain amount of uncertainty may exist in the pattern structures under study. Deformations on patterns due to noise or distortion may cause an input pattern different from all prototype patterns, either in primitive properties or in their relations, so that the assignment of it to any pattern class is rejected by conventional graph isomorphisms, even though the input pattern is deformed very slightly and still can be easily recognized as coming from one of the known classes. Such weakness of conventional graph isomorphisms is due to the following two reasons. 1) The isomorphism procedures lack *error-correcting* capability; only exact matching is allowed. 2) The procedures are *symbolic* in nature [5], [6]; they cannot process continuous numerical attributes which are often associated with node and branch labels to give more precise descriptions of primitives and their relations [2], [10].

It is attempted to solve these two problems in this paper. Definitions of relational graphs are first extended to include

Manuscript received March 21, 1979; revised August 8, 1979.
This work was supported in part by the National Science Foundation under Grant ENG78-16970 and in part by the Office of Naval Research under Contract N00014-79-C-0574.
The authors are with the School of Electrical Engineering, Purdue University, West Lafayette, IN 47907.

0018-9472/79/1200-0757$00.75 © 1979 IEEE

numerical attributes. The pattern deformational model proposed by Tsai and Fu [11], [12] is then extended to include relation deformations. Under such an extended deformational model, error-correcting isomorphisms and matching criteria are defined. Graph isomorphisms are next formulated as state-space search problems [13]. By using the probability distributions or weighted distances introduced on the nodes and branches in the graphs to guide the state-space search, an ordered-search algorithm is proposed for determining error-correcting isomorphisms of two relational graphs. Finally, a hybrid approach to pattern analysis using error-correcting isomorphisms and various decision rules is described which may be regarded as a combination of statistical and structural techniques [14].

II. Relational Graphs with Attributes

In this section basic concepts about primitives and their relations are first formalized with emphasis on the necessity of numerical attributes for more precise descriptions of these terms. According to these concepts a definition for attributed relational graphs is then given, which can be considered as a generalization of conventional ones without numerical attributes.

1) *Primitives:* As pointed out in [1], [11], and [15], two kinds of information are usually needed to give a precise description of a pattern primitive, viz., the *primitive structure* and *its (numerical and/or logical) attributes*. Conventional syntactic approaches usually use symbolic notations only to specify primitive structures, and if numerical information has to be utilized, thresholding usually is resorted. For example, the octal chain code [16] uses the thresholding results of directional attributes of line segments.

Definition 2.1: A pattern primitive a is denoted as a 2-tuple or a pair

$$a = (s, x)$$

where s is a *syntactic symbol* denoting the structure of a, and $x = [x_1, x_2, \cdots, x_m]$ is a *semantic vector* denoting m numerical and/or logical attributes of a. When $m = 0$, or no semantic attributes are used, set $x = \phi$.

2) *Relations:* To denote the relation between a given pair of primitives, phrases such as "above," "left of," "inside," etc., are used, which essentially only describe the syntactic information of the relations. If more accurate descriptions of the relations are required, semantic information such as numerical attributes can again be added. So, similar to the description of a primitive, we can describe a relation between two primitives by two kinds of information: the syntactic structure and the semantic attributes. For example, in Barrow and Popplestone [10] syntactic structures such as "above," "besides," etc., and semantic attributes such as relative sizes and distances, boundary adjacency and convexity, etc., are proposed to describe relations between pairs of picture regions.

Definition 2.2: A relation e between a pair of primitives a_1 and a_2 is denoted as a 2-tuple

$$e = (u, y)$$

where u is a *syntactic symbol* denoting the structure of e, and $y = [y_1, y_2, \cdots, y_n]$ is a *semantic vector* denoting n numerical and/or logical attributes of e. When $n = 0$, or no semantic attributes are used, set $y = \phi$.

The above definition is almost identical to that for a pattern primitive; this is no wonder because we can consider the description of a primitive as a relation between the primitive and itself. We will call either a primitive or a relation a *terminal*. Note that only binary relations are to be treated in this paper although generalization is easy in [2] and [5]. Actually, as pointed out in [1] and [17], given an n-ary relation it is easy to convert it into a set of binary relations.

3) *Attributed Relational Graphs:* With primitives and relations defined, a formal definition for attributed relational graphs is given in the following, which essentially follows that defined in Brayer and Fu [3] but includes semantic attributes for more general applications.

Definition 2.3: An attributed relational graph over $V = V_N \cup V_B$ is a 4-tuple

$$\omega = (N, B, \mu, \varepsilon)$$

where

N	is a finite nonempty set of nodes;
$B \subset N \times N$	is a set of distinct ordered pairs of distinct elements in N called branches;
V_N	is a finite nonempty set of node labels (primitive descriptions) each of which is of the form (s, x) as defined in Definition 2.1;
V_B	is a set of branch labels (relation descriptions) each of which is of the form (u, y) as defined in Definition 2.2;
$\mu: N \to V_N$	is a function called node interpreter;
$\varepsilon: B \to V_B$	is a function called branch interpreter.

III. A Deformational Model for Relational Graphs

In Tsai and Fu [11], [12] a pattern deformational model is proposed to take care of noisy or distorted pattern primitives with attributes, and based on such a model, a Bayes error-correcting parsing and classification system has been constructed for string and tree languages. In this section this deformational model is reviewed from a relational-graphic point of view and is extended to cover relation deformations. The extended model will serve in the next section for constructing an error-correcting isomorphism procedure for attributed relational graphs.

A. The Deformational Model

An *observed pattern* can usually be regarded as deformed from a *pure pattern*. Given a pure pattern there may exist a set of corresponding observed patterns[1] which form a pattern cluster or a subcluster of a pattern class.

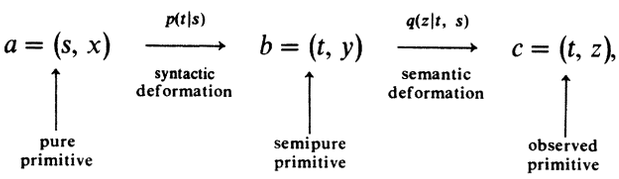
Using relational graphs as pattern representations, let $\omega = (N, B, \mu, \varepsilon)$ over $V_N \cup V_B$ and $\omega' = (N', B', \mu', \varepsilon')$ over $V_{N'} \cup V_{B'}$ be a pure pattern and one of its observed version,

[1] A pure pattern is also regarded as a possible observed pattern.

respectively, which we will call a *pure relational graph* and an *observed relational graph*. Note that $V_N \subset V_{N'}$ and $V_B \subset V_{B'}$. When $B = B'$, $\varepsilon = \varepsilon'$, and $N = N'$, but $\mu \neq \mu'$, i.e., when all relations are kept unchanged, and only primitives of graph nodes are deformed, we say that a *local deformation* is induced on ω and that ω' is *deformed locally* from ω. Since the pattern structures of ω and ω' represented by B, ε, and N are the same, such a deformation is usually called a *structure-preserved deformation* [12]. On the other hand, when the difference between ω and ω' is more than just $\mu \neq \mu'$, we say that a *structural deformation* is induced on ω and that ω' is *deformed structurally* from ω. Though various types of structural deformations can be identified, we consider in this paper only the case where $B = B'$ and $N = N'$, but $\varepsilon \neq \varepsilon'$ and $\mu \neq \mu'$. This case happens when the deformation does not affect the structure of the underlying unlabeled graph $G_\omega = (N, E)$ for ω but only corrupts the information contained in the primitives and relations. It includes the local deformation as a special case and is in nature quite general to include pattern deformations encountered in many practical applications. We will call this kind of deformation as *graph-preserved*.

B. Graph-Preserved Deformation

As proposed in [11] and [12], a primitive deformation can further be decomposed into two steps. Let $a = (s, x)$ be a primitive in the pure pattern ω and $c = (t, z)$ be a corresponding deformed version of a in the observed version ω' of ω. Then the two-step primitive deformation can be viewed in the following way:

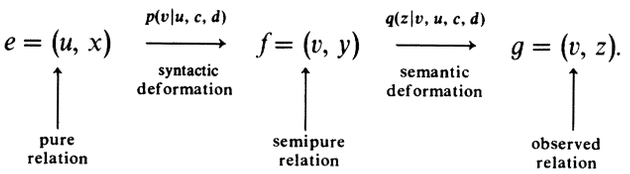


where the first step is a syntactic deformation induced on the syntactic symbol s of primitive a. It transforms a into a so-called *semipure primitive* $b = (t, y)$ with discrete probability $p(t|s)$, where y is a *representative semantic vector* for t. The second step is a semantic deformation induced on the semantic vector y of b. It corrupts b into the observed primitive $c = (t, z)$ with probability (or density) $q(z|t, s)$. The total probability (or density) for a to be deformed into c is

$$r(c|a) = q(z|t, s)p(t|s)$$

which is called the *primitive deformation probability* (*or density*) of c from a. Note that in the previous discussion we have already assumed implicitly that a primitive is deformed independently from any other terminals. This assumption is often made in discussing pattern deformations [11], [19], [20], although it means that no contextual information from primitives is to be utilized in subsequent recognition procedures.

In the case of relation deformations it becomes impractical if we also assume that a relation is deformed independently from any primitive or relation in the same graph. Actually, it is more likely for a relation to be changed according to how its two end primitives are changed and vice versa. So, given a pure relation $e = (u, x)$ between two primitives a, b in ω and its corresponding observed relation $g = (v, z)$ between primitives c, d which are deformed locally from a, b, respectively, we can decompose the relation deformation induced on e into two steps as follows:



The explanation is analogous to the one for primitive deformations except that the probability (or density) functions p and q also depend on the observed primitive c, d on the two end nodes of relation g. Now the probability (or density) for e to be deformed into g becomes

$$r(g|e, c, d) = q(z|v, u, c, d)p(v|u, c, d),$$

which is called the *relation deformation probability* (*or density*) of g from e.

Note that the decomposition of a terminal deformation into two steps is just for discussion convenience; the two steps may not be independent of each other. In the cases where the syntactic and semantic deformation probabilities (or densities) cannot be inferred separately, the total probabilities (or densities), i.e., $r(c|a)$ or $r(g|e, c, d)$, should be inferred directly.

C. Pattern Deformation Probabilities (or Densities)

Recall the following two assumptions made in Section B on the interdependency of terminal deformations.

1) A primitive in ω is deformed independently of any other terminals in ω.
2) A relation in ω is deformed independently of any other terminals in ω except those two on its two end nodes; it is deformed according to the *observed* primitives on these two end nodes.

Of course, more complicated assumptions other than these can also be made if more contextual information is to be utilized. Now let $\omega = (N, B, \mu, \varepsilon)$ over $V_N \cup V_B$ and $\omega' = (N, B, \mu', \varepsilon')$ over $V_{N'} \cup V_{B'}$ be the relational graph of a pure pattern and one of its observed versions, respectively, where

$$N = \{\alpha_i | i = 1, 2, \cdots, n_N\},$$
$$B = \{\gamma_i | \gamma_i = (\alpha_{i_1}, \alpha_{i_2}) \in N \times N, i = 1, 2, \cdots, n_B\},$$
$$V_N = \{a_i | a_i = \mu(\alpha_i) = (s_i, x_i), i = 1, 2, \cdots, n_N\},$$
$$V_B = \{e_i | e_i = \varepsilon(\gamma_i) = (u_i, y_i), i = 1, 2, \cdots, n_B\},$$
$$V_{N'} = \{a'_i | a'_i = \mu'(\gamma_i) = (s'_i, x'_i), i = 1, 2, \cdots, n_N\},$$
$$V_{B'} = \{e'_i | e'_i = \varepsilon'(\gamma_i) = (u'_i, y'_i), i = 1, 2, \cdots, n_B\},$$

and let $a \underset{\text{pd}}{\rightarrow} a'$ denote a primitive deformation of a' from a, and $e \underset{\text{rd}}{\rightarrow} e'$ denote a relation deformation of e' from e. The *pattern deformation probability* (*or density*) *of* ω' *from* ω, i.e.,

the probability (or density) that ω' is deformed from ω, $P(\omega'|\omega)$, can be computed as follows according to the above-mentioned assumptions (note that the two end primitives of relation e_j corresponding to branch $\gamma_j = (\alpha_{j_1}, \alpha_{j_2}) \in B$ are a_{j_1} and a_{j_2} according to the above notations):

$$\begin{aligned} P(\omega'|\omega) &= P\left\{\left(a_i \underset{\text{pd}}{\rightarrow} a_i',\ i = 1, 2, \cdots, n_N\right), \right. \\ &\qquad \left. \left(e_j \underset{\text{rd}}{\rightarrow} e_j',\ j = 1, 2, \cdots, n_B\right)\right\} \\ &= P\left\{a_i \underset{\text{pd}}{\rightarrow} a_i',\ i = 1, 2, \cdots, n_N\right\} \\ &\quad \cdot P\left\{e_j \underset{\text{rd}}{\rightarrow} e_j',\ j = 1, 2, \cdots, n_B \middle| a_i \underset{\text{pd}}{\rightarrow} a_i',\right. \\ &\qquad \left. i = 1, 2, \cdots, n_N\right\} \\ &= \prod_{i=1}^{n_N} P\left\{a_i \underset{\text{pd}}{\rightarrow} a_i'\right\} \\ &\quad \cdot \prod_{j=1}^{n_B} P\left\{e_j \underset{\text{rd}}{\rightarrow} e_j' \middle| a_{j_1} \underset{\text{pd}}{\rightarrow} a_{j_1}', a_{j_2} \underset{\text{pd}}{\rightarrow} a_{j_2}'\right\} \\ &= \prod_{i=1}^{n_N} r(a_i'|a_i) \cdot \prod_{j=1}^{n_B} r(e_j'|e_j, a_{j_1}', a_{j_2}'), \end{aligned}$$

where $r(a_i'|a_I)$ and $r(e_j'|e_j, a_{j_1}', a_{j_2}')$ are primitive and relation deformation probabilities (or densities), respectively, as defined in the last section. Or, in more detail, by decomposing $r(a_i'|a_i)$ and $r(e_j'|e_j, a_{j_1}', a_{j_2}')$ into syntactic and semantic deformation probabilities (or densities), we get

$$\begin{aligned} P(\omega'|\omega) = \prod_{i=1}^{n_N} & q(x_i'|s_i', s_i)p(s_i'|s_i) \\ & \cdot \prod_{j=1}^{n_B} q(y_j'|u_j', u_j, a_{j_1}', a_{j_2}')p(u_j'|u_j, a_{j_1}', a_{j_2}'). \end{aligned}$$

D. Pattern Deformation Distances

In some practical applications, due to an inadequate number of available sample patterns, it might be difficult to infer deformation probabilities or densities for use. Then nonstochastic methods must be adopted for measuring the possibility for a given pattern ω to be deformed into another pattern ω'. In this section we propose two such measurements, called *weighted distance* and *weighted-square-error distance*, for two special cases of our proposed general deformation formalism.

When there is no semantic deformation involved in a pattern deformation, we can assign a positive *deformation weight*, instead of a probability or density value, to specify *how far* an observed terminal (a primitive or a relation) is deformed *syntactically* from its pure version:

$$(s, x) \xrightarrow[\substack{\text{syntactic}\\ \text{deformation}}]{w(t|s)} (t, x') \text{ and } (u, y) \xrightarrow[\substack{\text{syntactic}\\ \text{deformation}}]{w(v|u,c,d)} (v, y')$$

for a primitive deformation induced on $a = (s, x)$ and for a relation deformation induced on $e = (u, y)$ with deformed primitives c, d on its two end nodes, respectively. Then the *weighted distance* for the pattern deformation of ω' from ω is defined to be

$$W(\omega'|\omega) = \sum_{i=1}^{n_N} w(s_i'|s_i) + \sum_{j=1}^{n_B} w(u_j'|u_j, a_{j_1}', a_{j_2}').$$

If no syntactic deformation is involved in a pattern deformation, then given a pure primitive (or relation) $a = (s, x)$ with $x = [x_1, x_2, \cdots, x_{n_a}]$ and an observed primitive (or relation) $b = (s, y)$ with $y = [y_1, y_2, \cdots, y_{n_a}]$, we can define the *weighted square error* of b with respect to a as

$$E(b|a) = \sum_{i=1}^{n_a} w_i(a) \cdot [y_i - x_i]^2,$$

where $w_i(a)$ is the *deformation weight* for the ith attribute of the semantic vector x in a. And the *weighted-square-error distance* for the pattern deformation of ω' is defined to be

$$\begin{aligned} E(\omega'|\omega) = \sum_{i=1}^{n_N} \sum_{l=1}^{n_{a,i}} & w_l(a_i) \cdot [x_{il}' - x_{il}]^2 \\ & + \sum_{j=1}^{n_B} \sum_{k=1}^{n_{e,j}} w_k(e_j) \cdot [y_{jk}' - y_{jk}]^2, \end{aligned}$$

where x_{il}' is the corresponding observed version of the lth attribute x_{il} in the semantic vector $x_i = [x_{i1}, x_{i2}, \cdots, x_{in_{a,i}}]$ of the primitive a_i in ω, and y_{jk}' is the observed attribute of y_{jk} in $y_j = [y_{j1}, y_{j2}, \cdots, y_{jn_{e,j}}]$ of the relation e_j in ω.

The larger these two distances are, the more impossible the corresponding pattern deformation is. Although these two kinds of *pattern deformation distances* are practical for use, they are optimal only under certain special conditions which were discussed in Tsai and Fu [25].

IV. Error-Correcting Isomorphisms of Relational Graphs

Given several pure patterns and an unknown observed pattern ω', all of which are represented by attributed relational graphs after appropriate preprocessing, we would like to assign ω' to the same class as one of the pure patterns according to a certain similarity criterion. One way to accomplish this pattern analysis problem is to use graph-matching procedures or graph isomorphisms. As pointed out in the Introduction most conventional graph isomorphism procedures do not have error-correcting matching capabilities. They also cannot handle semantic attributes which are often included in the description of primitives and relations in a graph. We will propose in this section an error-correcting graph isomorphism procedure for attributed relational graphs.

A. Definitions of Error-Correcting Isomorphisms

Conventional graph isomorphisms are *exact matchings*; a successful matching requires every pair of matched node labels and branch labels to be identical. In this section we define *error-correcting isomorphisms*. Let $\omega = (N, B, \mu, \varepsilon)$

over $V_N \cup V_F$ be a pure relational graph with N, B, V_N, V_E as denoted in Section III-C. Let the set of observed primitives into which a pure primitive $a_i \in V_N$ might be deformed be denoted as

$$D(a_i) = \left\{ a'_{ij} \,|\, a'_{ij} = (s'_{ij}, x'_{ij}),\ a_i \underset{\text{pd}}{\rightarrow} a'_{ij} \right\},$$

and let the set of observed relations into which a pure relation $e_i \in V_B$ (with a'_{i_1}, a'_{i_2} as the two *deformed* primitives on its two end nodes) might be deformed be denoted as

$$D(e_i \,|\, a'_{i_1}, a'_{i_2}) = \left\{ e'_{ij} \,|\, e'_{ij} = (u'_{ij}, y'_{ij}),\ e_i \underset{\text{rd}}{\rightarrow} e'_{ij} \right\}.$$

Note that $D(e_i \,|\, a'_{i_1}, a'_{i_2}) \neq D(e_i \,|\, a'_{i_2}, a'_{i_1})$.

Definition 4.1: Let $\omega' = (N', B', \mu', \varepsilon')$ over $V_{N'} \cup V_{B'}$ be an *observed relational graph* of the *pure relational graph* $\omega = (N, B, \mu, \varepsilon)$, where $N' = \{\alpha'_i \,|\, i = 1, 2, \cdots, n_{N'}\}$ and $B' = \{\gamma'_i \,|\, \gamma'_i = (\alpha'_{i_1}, \alpha'_{i_2}) \in N' \times N',\ i = 1, 2, \cdots, n_{B'}\}$. A function $h: N' \rightarrow N$ is called an *error-correcting isomorphism from* ω' *to* ω, which we denote as $h: \omega' \rightarrow \omega$, if the following conditions are satisfied:

1) for each $\alpha'_i \in N'$, $a'_i \in D(a_j)$, where $a'_i = \mu'(\alpha'_i)$, and $a_j = \mu(h(\alpha'_i))$, i.e., a'_i is one of the observed versions of a_j;
2) for each $\gamma'_i = (\alpha'_{i_1}, \alpha'_{i_2}) \in B'$, $h(\gamma'_i) \in B$, and $e'_i \in D(e_j \,|\, a'_{i_1}, a'_{i_2})$, where
$$h(\gamma'_i) = (h(\alpha'_{i_1}), h(\alpha'_{i_2})),$$
$$e'_i = \varepsilon'(\gamma'_i),$$
$$e_j = \varepsilon(h(\gamma'_i)),$$
$$a'_{i_1} = \mu'(\alpha'_{i_1}),$$
$$a'_{i_2} = \mu'(\alpha'_{i_2}),$$
i.e., e'_i is one of the observed versions of e_j;
3) h is one-to-one, i.e., for any two nodes $\alpha'_1, \alpha'_2 \in N'$, $h(\alpha'_1) = h(\alpha'_2)$ implies $\alpha'_1 = \alpha'_2$.

Definition 4.2: An error-correcting isomorphism (ECI) is called *full* or *graph-preserved* if $N' = N$ and $B' = B$. Otherwise, it is called *partial*. A partial ECI is also called a *subgraph ECI*.

Since only graph-preserved deformations are studied in this paper, we are mainly concerned with graph-preserved error-correcting isomorphisms (GPECI) in the following discussions, leaving subgraph error-correcting isomorphisms for further investigations. As an example to illustrate the definition of a GPECI, let Fig. 2 and Fig. 3 be the observed and the pure graphs $\omega' = (N', B', \mu', \varepsilon')$ and $\omega = (N, B, \mu, \varepsilon)$, respectively, where

$$N' = \{\alpha'_1, \alpha'_2, \alpha'_3, \alpha'_4\}, \qquad N = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\},$$
$$B' = \{\gamma'_1 = (\alpha'_1, \alpha'_3),\ \gamma'_2 = (\alpha'_1, \alpha'_2),\ \gamma'_3 = (\alpha'_1, \alpha'_4),\ \gamma'_4 = (\alpha'_4, \alpha'_3),\ \gamma'_5 = (\alpha'_2, \alpha'_4)\},$$
$$B = \{\gamma_1 = (\alpha_2, \alpha_1),\ \gamma_2 = (\alpha_4, \alpha_2),\ \gamma_3 = (\alpha_4, \alpha_1),\ \gamma_4 = (\alpha_3, \alpha_2),\ \gamma_5 = (\alpha_4, \alpha_3)\},$$

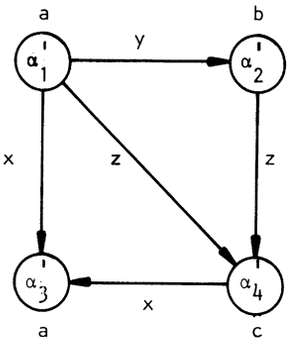


Fig. 2. Observed graph ω'.

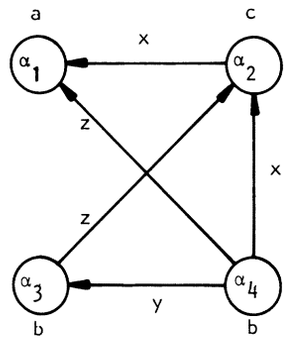


Fig. 3. Pure graph ω.

$$\mu'(\alpha'_1) = a, \quad \mu'(\alpha'_2) = b, \quad \mu'(\alpha'_3) = a, \quad \mu'(\alpha'_4) = c,$$
$$\mu(\alpha_1) = a, \quad \mu(\alpha_2) = c, \quad \mu(\alpha_3) = b, \quad \mu(\alpha_4) = b,$$
$$\varepsilon'(\gamma'_1) = x, \quad \varepsilon'(\gamma'_2) = y, \quad \varepsilon'(\gamma'_3) = z,$$
$$\varepsilon'(\gamma'_4) = x, \quad \varepsilon'(\gamma'_5) = z,$$
$$\varepsilon(\gamma_1) = x, \quad \varepsilon(\gamma_2) = x, \quad \varepsilon(\gamma_3) = z,$$
$$\varepsilon(\gamma_4) = z, \quad \varepsilon(\gamma_5) = y.$$

Obviously, ω' cannot match ω because ω' has two "a" node labels and ω has only one. Now suppose we have the following deformations:

$$D(a) = \{a, c\}, \quad D(b) = \{a, b\}, \quad D\{c\} = \{a, c\},$$
$$D(x \,|\, a, c) = \{x, z\}, \quad D(x \,|\, c, a) = \{x, y\},$$
$$D(y \,|\, a, b) = \{y, z\}, \quad D(y \,|\, b, b) = \{x, y\},$$
$$D(z \,|\, a, a) = \{x, z\}, \quad D(z \,|\, b, c) = \{x, z\}.$$

Then with the following one-to-one function h:

$$h: \begin{cases} \alpha'_1 \rightarrow \alpha_4 \\ \alpha'_2 \rightarrow \alpha_3 \\ \alpha'_3 \rightarrow \alpha_1 \\ \alpha'_4 \rightarrow \alpha_2, \end{cases}$$

we check the first two conditions of Definition 4.1:

1) $\mu'(\alpha'_1) = a \in D(b) = \{a, b\}$ where $b = \mu(\alpha_4) = \mu(h(\alpha'_1))$;
2) $\mu'(\alpha'_2) = b \in D(b) = \{a, b\}$ where $b = \mu(\alpha_3) = \mu(h(\alpha'_2))$;

3) $\mu'(\alpha'_3) = a \in D(a) = \{a, c\}$
where $a = \mu(\alpha_1) = \mu(h(\alpha'_3))$;
4) $\mu'(\alpha'_4) = c \in D(c) = \{a, c\}$
where $c = \mu(\alpha_2) = \mu(h(\alpha'_4))$;

and

1) $h(\gamma'_1) = h(\alpha'_1, \alpha'_3) = (\alpha_4, \alpha_1) = \gamma_3 \in B$
and $\varepsilon'(\gamma'_1) = x \in D(z \mid a, a) = \{x, z\}$
where $z = \varepsilon(\gamma_3) = \varepsilon(h(\gamma'_1))$, $a = \mu'(\alpha'_1)$,
and $a = \mu'(\alpha'_3)$;
2) $h(\gamma'_2) = h(\alpha'_1, \alpha'_2) = (\alpha_4, \alpha_3) = \gamma_5 \in B$
and $\varepsilon'(\gamma'_2) = y \in D(y \mid a, b) = \{y, z\}$
where $y = \varepsilon(\gamma_5) = \varepsilon(h(\gamma'_2))$, $a = \mu'(\alpha'_1)$,
and $b = \mu'(\alpha'_2)$;
3) $h(\gamma'_3) = h(\alpha'_1, \alpha'_4) = (\alpha_4, \alpha_2) = \gamma_2 \in B$
and $\varepsilon'(\gamma'_3) = z \in D(x \mid a, c) = \{x, z\}$
where $x = \varepsilon(\gamma_2) = \varepsilon(h(\gamma'_3))$, $a = \mu'(\alpha'_1)$,
and $c = \mu'(\alpha'_4)$;
4) $h(\gamma'_4) = h(\alpha'_4, \alpha'_3) = (\alpha_2, \alpha_1) = \gamma_1 \in B$
and $\varepsilon'(\gamma'_4) = x \in D(x \mid c, a) = \{x, y\}$
where $x = \varepsilon(\gamma_1) = \varepsilon(h(\gamma'_4))$, $c = \mu'(\alpha'_4)$,
and $a = \mu'(\alpha'_3)$;
5) $h(\gamma'_5) = h(\alpha'_2, \alpha'_4) = (\alpha_3, \alpha_2) = \gamma_4 \in B$
and $\varepsilon'(\gamma'_5) = z \in D(z \mid b, c)$
$= \{x, z\}$ where $z = \varepsilon(\gamma_4) = \varepsilon(h(\gamma'_5))$,
$\mu'(\alpha'_2) = b$, and $\mu'(\alpha'_4) = c$.

Since all three conditions are satisfied and $N = N$, $B' = B$, we conclude that there exists a GPECI from ω' to ω.

Using the notations in the previous definitions and according to the discussion in Section III, it is easy to derive for a GPECI $h: \omega' \to \omega$ the pattern deformation probability (or density) of ω' from ω as

$$P_h(\omega' \mid \omega) = \prod_{i=1}^{n_N} r(a'_i \mid a_j) \prod_{i=1}^{n_B} r(e'_i \mid e_j, a'_{i_1}, a'_{i_2}),$$

which specifies a measure for the goodness of matching by the GPECI and will be called the *likelihood* of the GPECI. Similarly, when only syntactic deformations are involved in the pattern deformation of ω' from ω, we can derive its weighted distance as

$$W_h(\omega' \mid \omega) = \sum_{i=1}^{n_N} w(s'_i \mid s_j) + \sum_{i=1}^{n_B} w(u'_i \mid u_j, a'_{i_1}, a'_{i_2}),$$

which will be called the *distance* between ω and ω' of the GPECI. Or when only semantic deformations are involved, we can derive for the pattern deformation its weighted-square-error distance as

$$E_h(\omega' \mid \omega) = \sum_{i=1}^{n_N} \sum_{l=1}^{n_{a,j}} w_l(a_j) \cdot [x'_{il} - x_{jl}]^2 + \sum_{i=1}^{n_B} \sum_{l=1}^{n_{e,j}} w_l(e_j) \cdot [y'_{il} - y_{jl}]^2,$$

which will be called the *square error* between ω and ω' of the GPECI.

Now, given two relational graphs ω' and ω, since there may exist *several* GPECI's from ω' to ω due to the varieties of terminal mappings between ω' and ω, it is necessary to determine a GPECI h with the *maximum likelihood* $P_h(\omega' \mid \omega)$, or with the *minimum distance* $W_h(\omega' \mid \omega)$, or with the *least square error* $E_h(\omega' \mid \omega)$, as the desired graph matching.

B. Graph Isomorphism as a State-Space Search Problem

It is well known that conventional graph isomorphisms for unlabeled graphs can be solved by tree-search methods [6], [13]. Without information such as path costs to guide such search procedures, graph isomorphism falls into the set of nondeterministic polynomial-time complete (NP-complete) problems [18] and needs exponential time with respect to the node number of an input graph, although various attempts have been tried to reduce the time requirement [6]–[9]. However, in the search of an *error-correcting isomorphism*, blind-search methods can be avoided because all the primitive and relation deformation probabilities (or densities) can be used to guide the search procedure. Actually, taking the negative logarithm of the likelihood $P_h(\omega' \mid \omega)$ of a GPECI h, we get

$$-\ln P_h(\omega' \mid \omega) = -\sum_{i=1}^{n_N} \ln r(a'_i \mid a_j) - \sum_{i=1}^{n_B} \ln r(e'_i \mid e_j, a'_{i_1}, a'_{i_2});$$

and we can consider each term $-\ln r(a'_i \mid a_j)$ or $-\ln r(e'_i \mid e_j, a'_{i_1}, a'_{i_2})$ as the *cost* of matching a node or a branch so that a uniform-cost search, or more generally, an ordered-search procedure [13] can be applied to determine maximum-likelihood (ML) GPECI's.

Before describing an ordered-search method for finding MLGPECI's, we first formulate an isomorphism problem as a state-space problem and explain how a blind tree-search procedure can be used to find an error-correcting subgraph isomorphism from an observed relational graph ω' to a pure one ω, where $\omega' = (N', B', \mu', \varepsilon')$ over $V_{N'} \cup V_{B'}$ and $\omega = (N, B, \mu, \varepsilon)$ over $V_N \cup V_B$ are as those specified in Definitions 4.1 and 4.2. Considering the problem as finding an isomorphism function $h: N' \to N$ such that all primitives and relations in ω match observed versions of corresponding primitives and relations in ω', we have the following state-space formulation [13].

a) *State Descriptions*—A state is described by a collection M of 2-tuples (i, j), each of which denotes a pair of matched nodes $\alpha'_i \in N'$ and $\alpha_j \in N$ found so far. The initial state is $M = \phi$.
b) *Operators*—Let $M_1 = \{i \mid (i, j) \in M\}$ and $M_2 = \{j \mid (i, j) \in M\}$, then an operator performs the following actions to a state M.
   1) Pick up a node $\alpha'_k \in N'$ with $k \notin M_1$, and a node $\alpha_l \in N$ with $l \notin M_2$, and form a 2-tuple (k, l).
   2) Add (k, l) to M if it is *valid* by satisfying the following conditions.
      i) Let $a'_k = \mu'(\alpha'_k)$, $a_l = \mu(\alpha_l)$, then $a'_k \in D(a_l)$, i.e., a'_k is a deformed version of a_l in ω.

ii) For each $(i, j) \in M$, if $\gamma'_m = (\alpha'_k, \alpha'_i)$ (or $\gamma'_m = (\alpha'_i, \alpha'_k)) \in B'$ then $\delta_n = (\alpha_l, \alpha_j)$ (or $\delta_n = (\alpha_j, \alpha_l)) \in B$ and $e'_m \in D(e_n | a'_k, a'_i)$ (or $e'_m \in D(e_n | a'_i, a'_k)$) where $e'_m = \varepsilon'(\gamma'_m)$, $e_n = \varepsilon(\delta_n)$, and $a'_k = \mu'(\alpha'_k)$, $a'_i = \mu'(\alpha'_i)$, i.e., each relation e'_m related to α'_k is a deformed version of the corresponding relation e_n in ω.

3) If (k, l) is not valid then the operator is not applicable. Operators corresponding to other 2-tuples (k', l') must be tried.

c) *The Goal State*—A state M with its corresponding $M_1 = N'$ is a goal state. (Note that for a subgraph isomorphism, $M_2 = N$ is not required.)

Next, define the *cost for adding a 2-tuple* (k, l) *to* M as

$$c(k, l) = -\ln r(a'_k | a_l) + \sum_{\gamma_m' \in R} [-\ln r(e'_m | e_n, a'_k, a'_i)],$$

where $R = \{\gamma'_m | \gamma'_m = (\alpha'_k, \alpha'_i)$ or $(\alpha'_i, \alpha'_k) \in B'$ and $(i, j) \in M\}$, i.e., the set of all branches between α'_k and all nodes α'_i in ω' which have been matched already. The value $c(k, l)$ specifies a negative quantitative measurement of the possibility for α'_k in ω' to match α_l in ω. The smaller the cost is, the more possible it is to match α'_k to α_l. Then the MLGPECI can be found by searching through the *state-space graph* for a solution path with the minimum path cost, which can be achieved by a uniform-cost search algorithm [13]. But since all the deformation probabilities or densities contained in ω provide us with more information than the cost function $c(k, l)$, an ordered-search algorithm becomes feasible for a more efficient search which uses an evaluation function *more informed* than the cost function, although a uniform-cost search is enough to guarantee a MLGPECI solution.

C. An Ordered-Search Algorithm for MLGPECI

During the search for a solution path in the state-space graph, each state description in the state space is called a *node*. The *successors of a node* are obtained by applying all applicable operators to the state description of the node. The process of calculating all the successors of a node is called "expanding a node." Then an ordered-search algorithm uses an evaluation function to order nodes for expansion [13]. For a node N_i we take an evaluation function as

$$\hat{f}(N_i) = \hat{g}_1(N_i) + \hat{g}_2(N_i),$$

where $\hat{g}_1(N_i)$ is the minimum total path cost from the start node S (corresponding to the initial state description) to N_i calculated during the search, and $\hat{g}_2(N_i)$ is a *consistent lower bounded estimate* using any heuristic information available of $g_2(N_i)$ which is the cost of an optimal path from node n to a goal node. Then, as long as $\hat{g}_2(N_i) \leq g_2(N_i)$, a corresponding ordered-search algorithm will expand fewer nodes, compared with a search algorithm using no heuristic information (such as a uniform-cost search algorithm) and will still be guaranteed to find a minimal cost solution path in the state-space graph.

In the case of finding a MLGPECI, each collection of 2-tuples M defined previously denotes a node N_M in the state-space graph. $\hat{g}_1(N_M)$ is just the total cost of adding all matched 2-tuples found so far to M. A heuristic information source for estimating $g_2(N_M)$ lies in the fact that when a 2-tuple (k, l) is to be added to M to specify a mapping h: $\alpha'_k \to \alpha_l$, all the nodes in ω' related to α'_k must match those nodes in ω related to α_l in terms of node numbers, node labels, and associated branch labels. Following this discussion, an evaluation function for the order-search algorithm to be proposed for MLGPECI's is defined as follows. Let $M = \{(i_1, j_1), (i_2, j_2), \cdots, (i_L j_L)\}$ with $(i_L, j_L) = (k, l)$ as the most recently added 2-tuple to M, then set the evaluation function for node N_M as

$$\hat{f}(N_M) = \hat{g}_1(N_M) + \hat{g}_2(N_M), \tag{1}$$

where $\hat{g}_1(N_M) = \sum_{k=1}^{L} c(i_k, j_k)$, and $\hat{g}_2(N_M)$ takes various values according to the following rules to specify a partial quantitative measurement of the possibility for the mapping h: $\alpha'_k \to \alpha_l$ to be accepted.

Estimation Rules for $\hat{g}_2(N_M)$:

1) Let $M_1 = \{i | (i, j) \in M\}$,
$M_2 = \{j | (i, j) \in M\}$,
$M_{31} = \{\alpha'_i | (\alpha'_i, \alpha'_k) \in B', \alpha'_i \notin M_1\}$,
with n_{31} elements,
$M_{32} = \{\alpha'_i | (\alpha'_k, \alpha'_i) \in B', \alpha'_i \notin M_1\}$,
with n_{32} elements,
$M_3 = M_{31} \cup M_{32}$ = all unmatched nodes related to α'_k in B',
$M_{41} = \{\alpha_j | (\alpha_j, \alpha_l) \in B, \alpha_j \notin M_2\}$,
with n_{41} elements,
$M_{42} = \{\alpha_j | (\alpha_l, \alpha_j) \in B, \alpha_j \notin M_2\}$
with n_{42} elements, and
$M_4 = M_{41} \cup M_{42}$ = all unmatched nodes related to α_l in β,
if $n_{31} = n_{41}$ *and* $n_{32} = n_{42}$, then check the next rule, or else set $\hat{g}_2(N_M) = \infty$ which means α'_k cannot match α_l due to unequal numbers of *directed branches* related to α'_k and α_l.

2) For *each* $\alpha'_i \in M_3$, if $\gamma'_m = (\alpha'_i, \alpha'_k)$ (or $\gamma'_m = (\alpha'_k, \alpha'_i)$), try to find *at least* one $\alpha_j \in M_4$ such that $a'_i \in D(a_j)$ and $e'_m \in D(e_n | a'_i, a'_k)$ (or $e'_m \in D(e_n | a'_k, a'_i)$), where $a'_k = \mu'(\alpha'_k)$, $a'_i = \mu'(\alpha'_i)$, $a_j = \mu(\alpha_j)$, $e'_m = \varepsilon'(\gamma'_m)$, and $e_n = \varepsilon(\gamma_n)$ with $\gamma_n = (\alpha_j, \alpha_l)$ (or $\gamma_n = (\alpha_l, \alpha_j)$). If no such α_j exists for some $\alpha'_i \in M_3$ then set $\hat{g}_2(N_M) = \infty$, which means that the mapping $\alpha'_k \to \alpha_l$ is impossible due to unmatched labels of neighboring nodes and branches, or check the next rule.

3) Let $\omega'(\alpha'_k)$ and $\omega(\alpha_l)$ be two reduced relational graphs from ω' and ω whose nodes are $\{\alpha'_k\} \cup M_3$ and $\{\alpha_l\} \cup M_4$, respectively. Try to find a *one-to-one* mapping h': $\omega'(\alpha'_k) \to \omega(\alpha_l)$ constrained with h': $\alpha'_k \to \alpha_l$ such that the following cost is minimized:

$$c'(h') = \sum_{\alpha_i' \in M_3} [-\ln r(a'_i | a_j) - \ln r(e'_m | e_n, a'_i, a'_k)],$$

and let $c'(k, l) = \min_{h'} c'(h')$, where $h': \alpha'_i \to \alpha_j$ and all notations are as defined in Rule 2) above. If such a mapping cannot be found, set $\hat{g}_2(N_M) = \infty$. Otherwise, set $\hat{g}_2(N_M) = c'(k, l)$ which specifies partially the goodness of a solution containing the mapping $h: \alpha'_k \to \alpha_l$.

Rule 3) is essentially a reduced-size error-correcting isomorphism problem constrained with the fixed mapping h': $\alpha'_k \to \alpha_l$. Rule 2) is actually included in Rule 3). However, since Rule 3) becomes computationally impractical when n_{31} or n_{32} is large,[2] we may in such cases have to replace Rule 3) with a simpler one *as long as we can find some lower bound of* $c'(k, l)$ to be set as the value of $\hat{g}_2(N_M)$ and keep Rule 2) for use. We suggest the following simpler rule for Rule 3).

4) For each $\alpha'_i \in M_3$, find an $\alpha_j \in M_4$ such that the cost for the mapping $h'': \alpha'_i \to \alpha_j$,

$$c''(\alpha'_i) = -\ln r(a'_i | a_j) - \ln r(e'_m | e_n, a'_i, a'_k),$$

is minimized. Note that h'' may be many-to-one. Let the total cost be

$$c''(k, l) = \sum_{\alpha_i' \in M_3} \min_{\alpha_j \in M_4} c''(\alpha'_i),$$

then set $\hat{g}(N_M) = c''(k, l)$.

Rule 4) essentially tries to find an error-correcting *homomorphism* h'': $M_3 \to M_4$ (an isomorphism not necessarily one-to-one) which can always be found easily if Rule 2) is already satisfied. Note that $c''(k, l) \le c'(k, l)$ is always true such that $c''(k, l)$ also serves as a lower bound for $g_2(N_M)$. We are ready now to propose an ordered-search algorithm for finding MLGPECI's.

MLGPECI Algorithm

Input: An observed relational graph $\omega' = (N', B', \mu', \varepsilon')$ over $V_{N'} \cup V_{B'}$ and a pure relational graph $\omega = (N, B, \mu, \varepsilon)$ over $V_N \cup V_B$, where $N = N'$, $B = B'$.

Output: A maximum-likelihood graph-preserved error-correcting isomorphism $h: \omega' \to \omega$ with likelihood $P_h(\omega' | \omega)$.

Steps: Let N_M denote a node in the state-space graph with state description M. M_1 is as defined previously.

1) Put the start node N_{M_o} with $M_o = \phi$ on a list called OPEN, and set $f(N_M) = 0$.
2) If OPEN is empty then no MLGPECI exists; set $P(\omega' | \omega) = 0$. Otherwise execute the following steps.
3) Remove from OPEN the node N_M with a smallest $\hat{f}$ value and put it on a list called CLOSED.
4) If the M_1 of this node N_M is equal to N', then a MLGPECI h represented by M is found whose likelihood is given by

$$P_h(\omega' | \omega) = \exp[-\hat{g}_1(N_M)],$$

where $\hat{g}_1(N_M)$ is as defined previously. Otherwise continue.

5) Expand node N_M, using all operators applicable to M as defined in Section B. Compute the value $\hat{f}(N_{M'})$ for each successor $N_{M'}$ of N_M according to (1) and the estimation rules for $\hat{g}_2$ mentioned previously. Put these successors on OPEN.
6) Go to Step 2).

The above algorithm follows that given in [13]. If $\hat{g}_2(N_M)$ is always set zero, it is reduced to a uniform-cost search algorithm. Note that during the computation of $\hat{f}(N_{M'})$, numerical attributes are included in the deformation probabilities or densities and thus have been utilized for guiding the search. Consequently, we have obtained an isomorphism algorithm with an error-correcting capability for attributed relational graphs.

The previous discussions and the ordered-search algorithm for MLGPECI's certainly can be easily modified for determining the minimum-distance GPECI (MDGPECI) and the least-square-error GPECI (LSEGPECI) as follows.

i) For MDGPECI's, replace the negative logarithm of each terminal deformation probability (or density) with its corresponding weight and the likelihood $P_h(\omega' | \omega)$ with the weighted distance $W_h(\omega' | \omega)$ in the search algorithm, and set $W_h(\omega' | \omega) = \hat{g}_1(N_M)$ in Step 4) of the algorithm.

ii) For a LSEGPECI, replace the negative logarithm of each terminal deformation probability (or density) with its corresponding weighted square error and $P_h(\omega' | \omega)$ with the weighted-square-error distance $E_h(\omega' | \omega)$ in the algorithm, and set $E_h(\omega' | \omega) = \hat{g}_1(N_M)$ in Step 4).

D. An Illustrative Example for Determining MDGPECI

The example given in the following is used to illustrate the usage of the proposed ordered-search algorithm for determining minimum-distance GPECI's. We use Figs. 4 and 5 as the observed and pure relational graphs ω' and ω, respectively. In summary, we have the following notations for $\omega' = (N', B', \mu', \varepsilon')$ over $V_{N'} \cup V_{B'}$ and $\omega = (N, B, \mu, \varepsilon)$ over $V_N \cup V_B$:

$$N' = \{\alpha'_i | i = 1, 2, 3, 4, 5\},$$

$$N = \{\alpha_i | i = 1, 2, 3, 4, 5\}$$

$$B' = \{\gamma'_1 = (\alpha'_1, \alpha'_2), \gamma'_2 = (\alpha'_1, \alpha'_5), \gamma'_3 = (\alpha'_2, \alpha'_5),$$
$$\gamma'_4 = (\alpha'_3, \alpha'_1), \gamma'_5 = (\alpha'_2, \alpha'_3),$$
$$\gamma'_6 = (\alpha'_3, \alpha'_4), \gamma'_7 = (\alpha'_3, \alpha'_5),$$
$$\gamma'_8 = (\alpha'_4, \alpha'_1), \gamma'_9 = (\alpha'_4, \alpha'_2)\}$$

$$B = \{\gamma_1 = (\alpha_1, \alpha_2), \gamma_2 = (\alpha_3, \alpha_1), \gamma_3 = (\alpha_1, \alpha_4),$$
$$\gamma_4 = (\alpha_1, \alpha_5), \gamma_5 = (\alpha_2, \alpha_3),$$
$$\gamma_6 = (\alpha_2, \alpha_4), \gamma_7 = (\alpha_3, \alpha_5),$$
$$\gamma_8 = (\alpha_4, \alpha_3), \gamma_9 = (\alpha_4, \alpha_5)\}$$

[2] A case for this happens when the *connectivity* among nodes is high, i.e., when there exists a relation between almost every node pair.

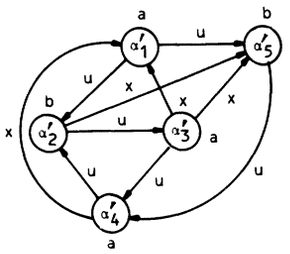


Fig. 4. Observed graph ω'.

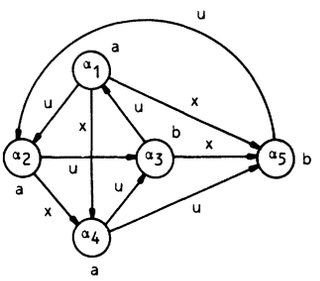


Fig. 5. Pure graph ω.

$$V_{N'} = \{a'_1 = a,\ a'_2 = b,\ a'_3 = a,\ a'_4 = a,\ a'_5 = b\}$$

$$V_N = \{a_1 = a,\ a_2 = a,\ a_3 = b,\ a_4 = a,\ a_5 = b\}$$

$$V_{B'} = \{e'_1 = u,\ e'_2 = u,\ e'_3 = x,\ e'_4 = x,\ e'_5 = u,\ e'_6 = u,\ e'_7 = u,\ e'_8 = x,\ e'_9 = u\}$$

$$V_B = \{e_1 = u,\ e_2 = u,\ e_3 = x,\ e_4 = x,\ e_5 = u,\ e_6 = x,\ e_7 = x,\ e_8 = u,\ e_9 = u\}.$$

Note that $a'_i = \mu'(\alpha'_i)$, $a_i = \mu(\alpha_i)$, $e'_i = \varepsilon'(\gamma'_i)$, and $e_i = \varepsilon(\gamma_i)$. Next we assume the following possible deformations for primitives and relations in ω together with the specified deformation weights:

$$a \xrightarrow[\text{pd}]{w(a|a)=0.1} a, \qquad a \xrightarrow[\text{pd}]{w(b|a)=0.3} b$$

$$b \xrightarrow[\text{pd}]{w(a|b)=0.4} a, \qquad b \xrightarrow[\text{pd}]{w(b|b)=0.1} b$$

$$x \xrightarrow[\text{rd}]{w(u|x)=0.6} u, \qquad x \xrightarrow[\text{rd}]{w(x|x)=0.1} x$$

$$u \xrightarrow[\text{rd}]{w(u|u)=0.1} u, \qquad u \xrightarrow[\text{rd}]{w(x|u)=0.7} x.$$

For simplicity we have assumed that each terminal is deformed identically regardless of its position in the graph and, furthermore, that each relation is deformed independently from how its two end primitives are deformed. Now we want to find out a minimum-distance GPECI from ω' to ω by the ordered-search algorithm. We use Rules 1), 2), and 4) when computing $\hat{g}_2(N_M)$. The resulting state-space graph is shown in Fig. 6 with dark arcs showing the solution path. The numbers beside the arcs are the values $\hat{f}(N_M)$ of the pointed nodes. The distance of the solution GPECI is 1.5. Totally, only five nodes are expanded with 11 nodes generated. As a computational example, the value $\hat{f}(N_M)$ associated with the arc from node 3 to node 4 (i.e., for $M = \{(1,4), (2,3), (3,1)\}$) is computed as

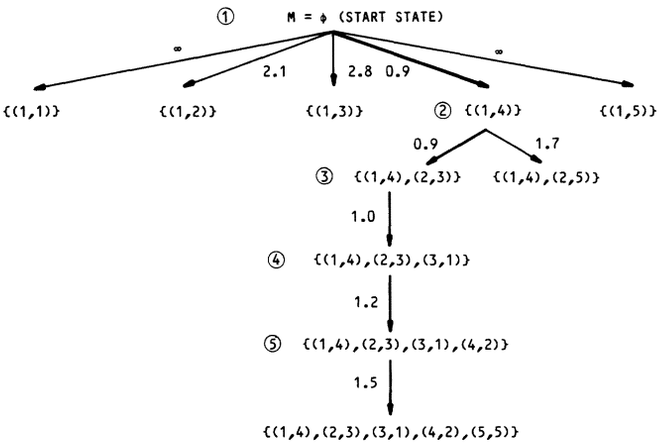


Fig. 6. State-space graph using ordered-search algorithm (circled numbers specify node expansion order).

$$\begin{aligned}\hat{g}_1(N_M) &= c(1,4) + c(2,3) + c(3,1)\\ &= w(a'_1|a_4) + [w(a'_2|a_3) + w(e'_1|e_8)]\\ &\quad + [w(a'_3|a_1) + w(e'_4|e_3) + w(e'_5|e_2)]\\ &= w(a|a) + [w(b|b) + w(u|u)]\\ &\quad [w(a|a) + w(x|x) + w(u|u)]\\ &= 0.1 + (0.1 + 0.1) + (0.1 + 0.1 + 0.1)\\ &= 0.6,\end{aligned}$$

$$\begin{aligned}\hat{g}_2(N_M) &= c''(3,1) && \text{[by Rule 4)]}\\ &= \min_{\alpha_j \in M_4} c''(\alpha'_4)\\ &\quad + \min_{\alpha_j \in M_4} c''(\alpha'_5) && [M_4 = \{\alpha_2, \alpha_5\}]\\ &= [w(a'_4|a_2) + w(e'_6|e_1)] && [\text{for } \alpha'_4 \to \alpha_2]\\ &\quad + [w(a'_5|a_5) + w(e'_7|e_4)] && [\text{for } \alpha'_5 \to \alpha_5]\\ &= [w(a|a) + w(u|u)]\\ &\quad + [w(b|b) + w(x|x)]\\ &= (0.1 + 0.1) + (0.1 + 0.1)\\ &= 0.4,\end{aligned}$$

and

$$\hat{f}(N_M) = \hat{g}_1(N_M) + \hat{g}_2(N_M) = 1.0.$$

For comparison, we also use the uniform-cost search algorithm (with $\hat{g}_2(N_M)$ always set zero) to obtain the same MDGPECI. The resulting state-space graph is shown in Fig. 7. As can be seen, many more nodes are expanded and generated before a solution path is found.

V. Pattern Analysis By Graph Isomorphisms

Given pure patterns $\omega_1, \omega_2, \cdots, \omega_c$ and an observed pattern ω', all represented by relational graphs, we would like to assign ω' to the same class as a pure pattern in the

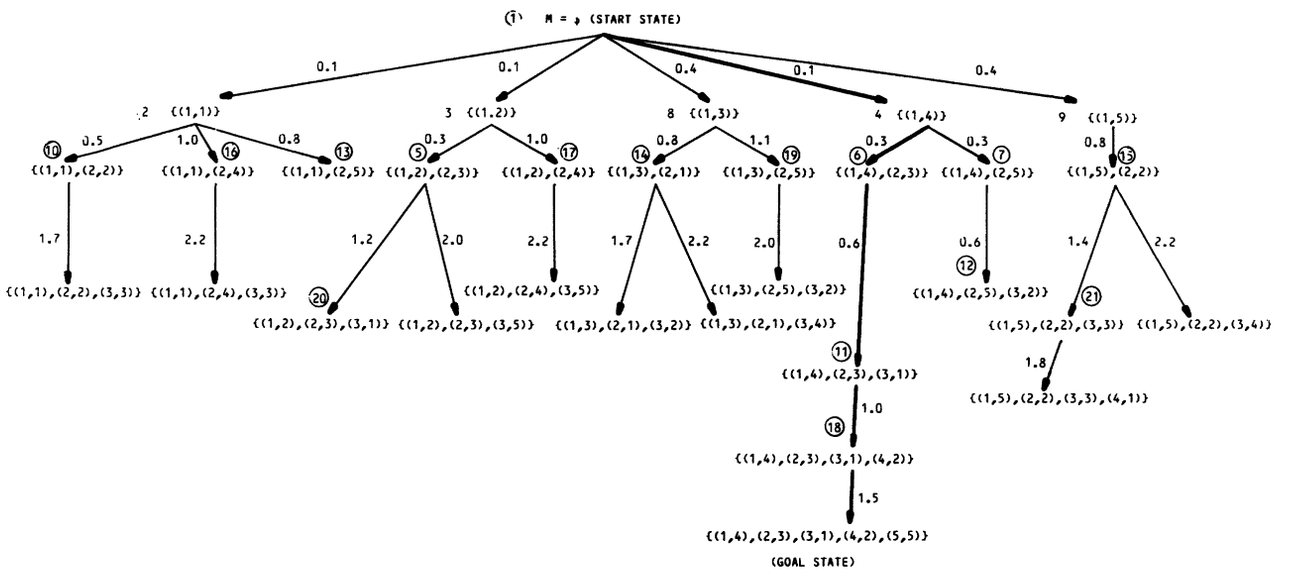


Fig. 7. State-space graph using uniform-cost search algorithm (circled numbers specify node expansion order).

sense of maximum likelihood $P(\omega_l|\omega')$. This can be accomplished by the following rule. Decide

$$\omega' \to \omega_l, \quad \text{if } P(\omega_l|\omega') = \max_{i=1,2,\cdots,c} P(\omega_i|\omega'), \text{ or}$$

$$\text{if } P(\omega'|\omega_l)P(\omega_l) = \max_{i=1,2,\cdots,c} P(\omega'|\omega_i)P(\omega_i),$$

where $P(\omega_i)$ is the *a priori* probability for pure pattern ω_i, and $P(\omega'|\omega_i)$, the so-called *likelihood for* ω_i *with respect to* ω', is exactly the pattern deformation probability (or density) of ω' from ω and can be computed by the proposed MLGPECI algorithm in the last section.

When no terminal deformation probabilities (or densities) are available and only syntactic deformations occur in the pattern deformation, the minimum-distance GPECI algorithm (modified from the MLGPECI algorithm by the methods discussed in Section IV-D) can be used for computing the value $W(\omega'|\omega_l)$, $l = 1, 2, \cdots, c$, and ω' can be classified according to the following rule. Decide

$$\omega' \to \omega_l, \quad \text{if } W(\omega'|\omega_l) = \min_{i=1,2,\cdots,c} W(\omega'|\omega_i).$$

If all terminal deformations are semantic, the least-square-error GPECI algorithm (also modified from the MLGPECI algorithm) can be used for computing the value $E(\omega'|\omega_l)$, $l = 1, 2, \cdots, c$, and ω' can be classified according to the following rule. Decide

$$\omega' \to \omega_l, \quad \text{if } E(\omega'|\omega_l) = \min_{i=1,2,\cdots,c} E(\omega'|\omega_i).$$

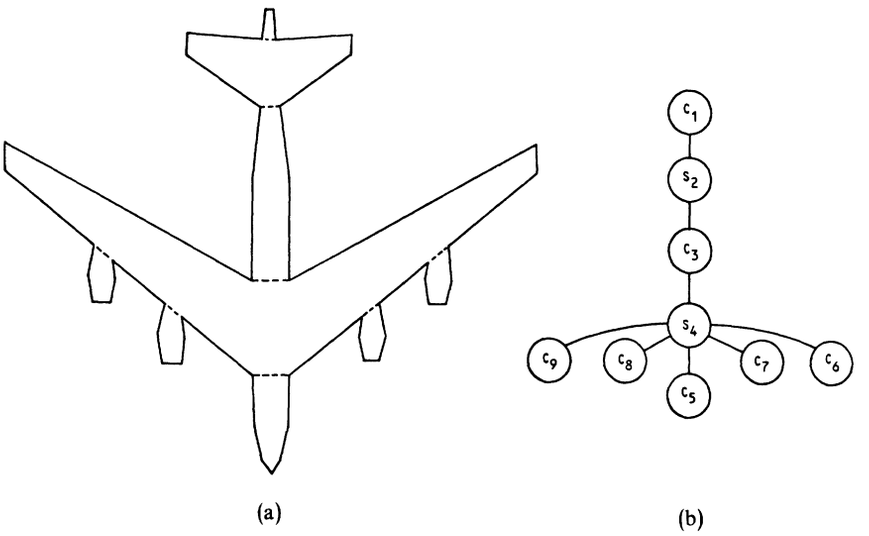
Both of the two nonstochastic classification rules are statistically optimal only when all *a priori* probabilities $P(\omega_i)$ are equal and when those conditions given in [25] are satisfied, although they are useful for practical applications.

A. An Application Example—Shape Analysis

Feng and Pavlidis [21] proposed a procedure for decomposing the polygonal approximation of a given shape into simpler components and described the decomposed components by labeled graphs. Each node primitive of the graph is a convex, T-type, or spiral-type object which can be described by such numerical attributes as width, elongation, etc., [22]. The branches connecting the nodes certainly need some attributes to describe the relative positions and adjacencies of the nodes if detailed descriptions are necessary. The resulting labeled graph is an *attributed relational graph* we have defined previously, and for shape classification using this approach, the proposed error-correcting isomorphisms obviously are necessary and applicable. For example, given two different aircrafts shown in Figs. 8(a) and 9(a), respectively, after they are decomposed (marked by dotted lines), two identical *symbolic* relational graphs are obtained as shown in Figs. 8(b) and 9(b), respectively. This illustrates that conventional relational graphs are insufficient for describing certain patterns. The only way to discriminate these two aircrafts by relational graphs is to use primitive and relation attributes. Now given a distorted unknown aircraft shown in Fig. 10(a), polygonal decomposition again results in a graph (Fig. 10(b)) identical to those shown in Figs. 8(b) and Fig. 9(b). Again, we have to extract attributes for primitives and relations, and the recognition of this distorted aircraft can be accomplished by applying the proposed error-correcting isomorphism algorithm and the classification rule using the weighted least-square-error criterion. For details and more examples refer to Tsai and Fu [25], which also discusses two other possible applications—scene analysis and texture analysis. Recently, a database system for images has been proposed [27] in which an attributed relational graph can be transformed into a corresponding relational database. The proposed error-correcting isomorphism algorithm certainly can be applied when images with specific graph structures and attributes are searched in the database system.

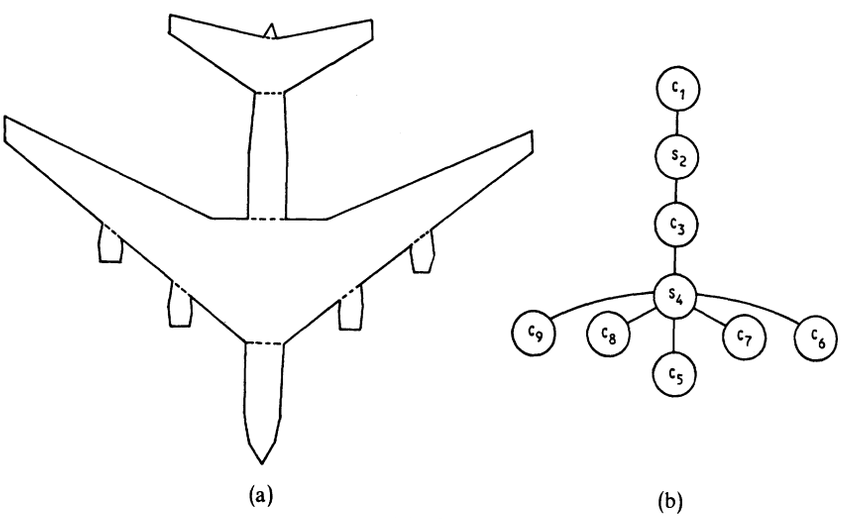
VI. Conclusion

A hybrid approach to pattern analysis as suggested by several investigators [14], [23], [24] has been formulated in



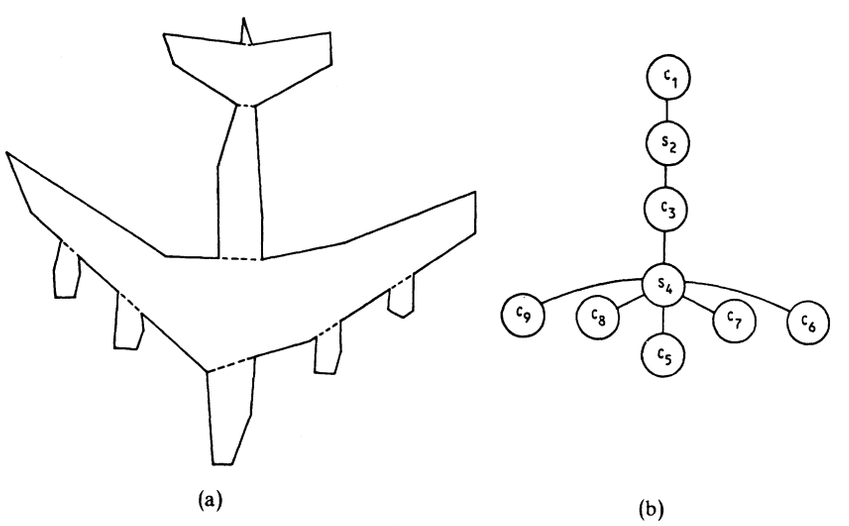
(a) (b)

Fig. 8.



(a) (b)

Fig. 9.



(a) (b)

Fig. 10.

this paper, which consists of both statistical classification methods, such as the use of deformation probabilities and the maximum-likelihood decision rule, and structural techniques, such as representing patterns by relational graphs and finding their isomorphisms. Suggested directions for further investigations include 1) extension of the proposed approach to cover *subgraph isomorphisms* which can take care of observed patterns with deleted or inserted primitives, 2) inference of primitive and relation deformation probabilities (or densities) or weights under the formalism of the proposed deformational model, and 3) application of the proposed approach to more complex real world problems.

REFERENCES

[1] K. S. Fu, *Syntactic Methods in Pattern Recognition.* New York: Academic, 1974.

[2] H. G. Barrow, A. P. Ambler, and R. M. Burstall, "Some techniques for recognizing structures in pictures," in *Frontiers in Pattern Recognition*, Watanabe, Ed. New York: Academic, 1972.

[3] J. M. Brayer and K. S. Fu, "Web grammars and their application to pattern recognition," Purdue Univ., Lafayette, IN, Tech. Rep. TR-EE 75-1, Dec. 1975.

[4] T. Pavlidis, "Grammatical and graph theoretical analysis of pictures," in *Graphic Languages*, Nake and Rosenfeld, Eds. Amsterdam, North-Holland, 1972.

[5] R. M. Haralick and J. S. Kartus, "Arrangements, homomorphisms, and discrete relaxations," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-8, Aug. 1978.

[6] J. R. Ullman, "An algorithm for subgraph isomorphism," *J. Ass. Comput. Mach.*, vol. 23, no. 1, Jan. 1976.

[7] S. H. Unger, "GIT—A heuristic program for testing pairs of directed line graphs for isomorphism," *Commun. Ass. Comput. Mach.*, vol. 7, no. 1, Jan. 1964.

[8] D. G. Corneil and C. C. Gotlieb, "An efficient algorithm for graph isomorphism," *J. Ass. Comput. Mach.*, vol. 17, no. 1, Jan. 1970.

[9] A. T. Berztiss, "A backtrack procedure for isomorphism of directed graphs," *J. Ass. Comput. Mach.*, vol. 20, no. 3, July 1973.

[10] H. G. Barrow and R. J. Popplestone, "Relational descriptions in picture processing," *Machine Intelligence*, no. 6, Meltzer and Michie, Eds. Edinburgh: University Press, 1971.

[11] W. H. Tsai and K. S. Fu, "A pattern deformational model and Bayes error-correcting recognition system," in *Proc. Int. Conf. Cybernetics and Society*, Tokyo, Japan, Nov. 3–7, 1978.

[12] ——, "A pattern deformational model and Bayes error-correcting recognition system," Purdue Univ., Lafayette, IN, Tech. Rep. TR-EE 78-26, May 1978.

[13] N. J. Nilsson, *Problem Solving Methods in Artificial Intelligence.* New York: McGraw-Hill, 1971.

[14] F. W. Blackwell, "Combining mathematical and structural pattern recognition," in *Proc. 2nd Int. Joint Conf. Pattern Recognition*, Copenhagen, Demark, 1974.

[15] K. C. You and K. S. Fu, "Syntactic shape recognition using attributed grammars," *IEEE Trans. Syst., Man. Cybern.*, vol. SMC-9, June 1979.

[16] H. Freeman, "On the encoding of arbitrary geometric configurations," *IEEE Trans. Electron. Comput.*, vol. EC-10, 1961.

[17] A. C. Shaw, "A formal picture description scheme as a basis for picture processing systems," *Inform. Cont.*, vol. 14, pp. 9–52, 1969.

[18] M. R. Garey, D. S. Johnson, and R. E. Tarjau, *SIAM J. Comput.*, vol. 5, 1976.

[19] L. W. Fung and K. S. Fu, "Stochastic syntactic decoding for pattern classification," *IEEE Trans. Comput.*, vol. C-24, no. 6, June 1975.

[20] V. A. Kovalevsky, "Sequential optimization in pattern recognition and pattern description," in *Proc. Int. Fed. Info. Process. Congr.*, Amsterdam, the Netherlands, 1968.

[21] H. Y. Feng and T. Pavlidis, "Decomposition of polygons into simpler components: Feature generation for syntactic pattern recognition," *IEEE Trans. Comput.*, vol. C-24, no. 6, June 1975.

[22] T. Pavlidis, *Structural Pattern Recognition*, New York: Springer-Verlag, 1977.

[23] C. H. Chen, "On statistical and structural feature extraction," Joint Workshop Pattern Recog. and Art. Intellig., Hyannis, MA, June 1976.

[24] L. Kanal and B. Chandrasekaran, "On linguistic, statistical, and mixed models for pattern recognition," in *Frontier of Pattern Recognition*, S. Watanabe, Ed. New York: Academic, 1972.

[25] W. H. Tsai and K. S. Fu, "Error-correcting isomorphisms of attributed relational graphs for pattern classification," *Purdue Univ.*, Lafayette, IN, Tech. Rep. TR-EE 79-3, Jan. 1979.

[26] J. L. Pfaltz, "Web grammrs and picture descriptions," *Comput. Graphics and Image Processing*, vol. 1, no. 2, pp. 193–220, 1972.

[27] N. S. Chang and K. S. Fu, "A relational database, system for images," *Purdue Univ.*, Lafayette, IN, Tech. Rep. TR-EE 79-28, June 1979.