Similarity Assessment for Relational CBR

Eva Armengol and Enric Plaza

IIIA - Artificial Intelligence Research Institute,
CSIC - Spanish Council for Scientific Research,
Campus UAB, 08193 Bellaterra, Catalonia (Spain).
{eva, enric}@iiia.csic.es,

Abstract. Reasoning and learning from cases are based on the concept of similarity often estimated by a distance. This paper presents LAUD, a distance measure that can be used to estimate similarity among relational cases. This measure is adequate for domains where cases are best represented by relations among entities. An experimental evaluation of the accuracy of LAUD is presented for the task of classifying marine sponges.

1 Introduction

Reasoning and learning from cases is based on the concept of similarity. Often similarity is estimated by a distance (a metric) or a pseudo-metric. This approach proceeds by a pairwise similarity comparison of a *problem* with every *precedent case* available in a case base; then one case (or k cases) with greatest (greater) similarity is (are) selected. This process is called the *retrieval* phase in Case-based Reasoning (CBR), and also plays a pivotal role in lazy learning techniques like Instance-based Learning (IBL) and k-nearest neighbor. In classification tasks, the solution class of the *problem* is inferred from the solution class of the precedent case(s) selected.

However, distance-based approaches to case retrieval are mainly used for propositional cases, i.e. cases represented as attribute-value vectors. We are interested in this paper in learning tasks where cases are best represented in a scheme that uses relations among entities. We will call this setting *relational case-based learning*. One option to achieve case-based learning in a relational setting is to adapt the process of pairwise similarity comparison by defining a distance that works upon relational instances. An example of similarity to be applied in relational cases is that used by RIBL ([7]) where the cases are represented as collections of Horn clauses (see related work on section 5).

We are interested in using cases represented in a relational way using *feature terms*[1]. Feature terms are a generalization of first order terms. In this representation entities are typed by *sorts* and relations among entities are represented by *features*. In this paper we introduce LAUD, a new distance measure that we use to estimate the similarity of relational cases represented as *feature terms*.

The structure of this paper is the following. In section 2 we introduce the feature term representation. In section 3 we introduce a new similarity for esti-

mating the similarity of cases represented as feature terms. In section 4 we provide some results of the application of the similarity to identify marine sponges. Finally, we report some related work and the conclusions and future work.

2 Representation of the cases

Feature Terms (also called feature structures or ψ-terms) are a generalization of first order terms. The difference between feature terms and first order terms is the following: a first order term, e.g. $f(x, y, g(x, y))$ can be formally described as a tree and a fixed tree-traversal order. In other words, parameters are identified by position. The intuition behind a feature term is that it can be described as a labelled graph i.e. parameters are identified by name. A formal definition of feature terms is the following:

Given a signature $\Sigma = \langle S, \mathcal{F}, \preceq \rangle$ (where S is a set of sort symbols that includes $\bot$; $\mathcal{F}$ is a set of feature symbols; and $\preceq$ is a decidable partial order on S such that $\bot$ is the least element) and a set ϑ of variables, we define *feature terms* as an expression of the form:

$$\psi ::= X : s[f_1 \doteq \Psi_1 \dots f_n \doteq \Psi_n] \tag{1}$$

where X is a variable in ϑ called the *root* of the feature term, s is a sort in S, the function *root(X)* returns the sort of the root, $f_1 \dots f_n$ are features in $\mathcal{F}$, $n \geq 0$, and each Ψ_i is a set of feature terms and variables. When $n = 0$ we are defining a variable without features. The set of variables occurring in ψ is noted as ϑ_ψ.

Sorts have an informational order relation ($\preceq$) among them, where $\psi \preceq \psi'$ means that ψ has less information than ψ' or equivalently that ψ is more general than ψ'. The minimal element ($\bot$) is called *any* and it represents the minimum information. When a feature has unknown value it is represented as having the value *any*. All other sorts are more specific than *any*. We restrict ourselves to use sort hierarchies with single inheritance (every sort has only one most specific supersort).

Using the $\preceq$ relation, we can introduce the notion of *least upper bound (lub)* commonly used in ILP [9]. The *lub* of two sorts is the most specific sort generalizing both. As we will explain in section 2.1 the notion of *lub* will be used to define the anti-unification of two feature terms. A *path* $\pi(X, f_i)$ is defined as a sequence of features going from the variable X to the feature f_i. The *depth* of a feature f in a feature term ψ with root X is the number of features that compose the path from the root X to f, including f, with no repeated nodes. Given a particular maximum feature depth k, a *leaf feature* of a feature term is a feature f_i such that either 1) the depth of f_i is k or 2) the value of f_i is a term without features. We call $leaves(\psi, k)$ the set of leaf features of a term ψ.

Let us illustrate the concepts introduced above with an example. The feature term of Figure 1 represents the description of a marine sponge. The *root* of this feature term is *s364*, the sorts are written in *italic* (for instance, *sponge, external-features, growing*, ...), some features are external-features, ecological-features, megascleres, separable, aspect, etc. Notice that the features ornamentation and

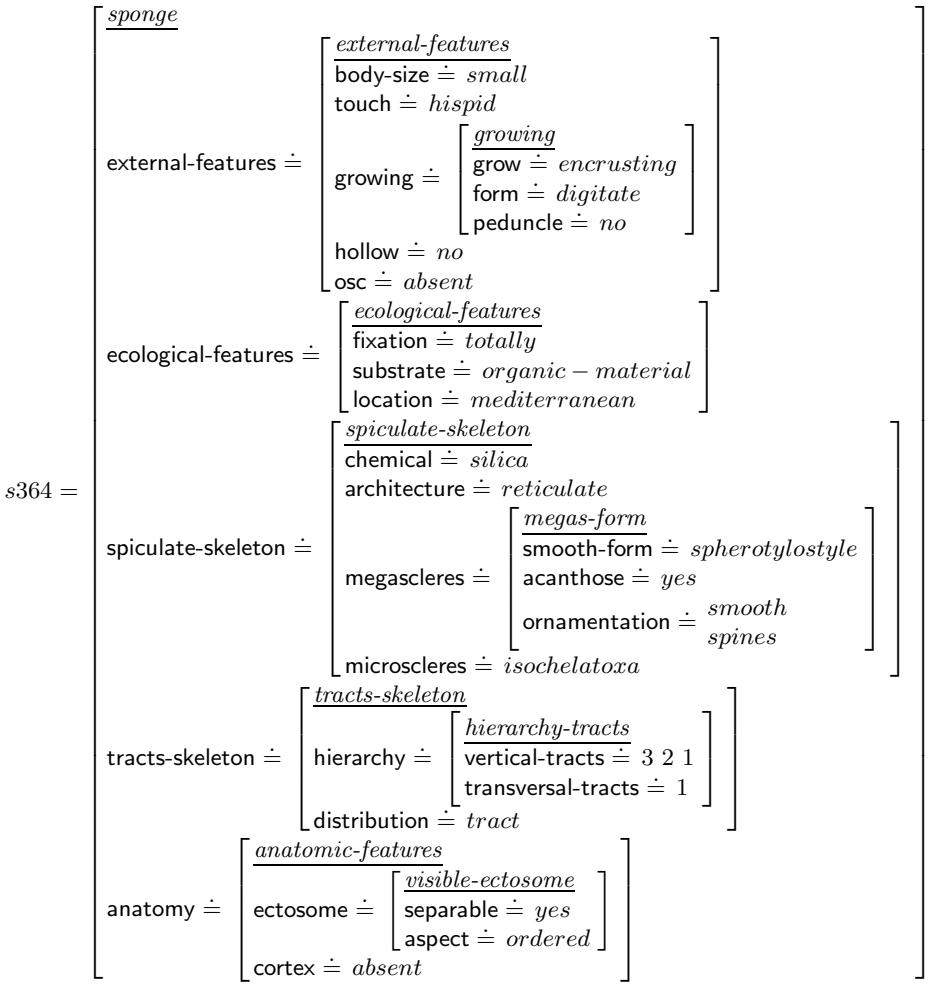

Fig. 1. Representation of a sponge using feature terms.

vertical-tracts are set-valued. The feature leaves of *s364* are the following {body-size, touch, grow, form, peduncle, hollow, osc, fixation, substrate, location, chemical, architecture, smooth-form, acanthose, ornamentation, microscleres, vertical-tracts, transversal-tracts, distribution, separable, aspect, cortex}. An example of path is π(*s364*, acanthose) that represents the path from the root to the leaf feature acanthose, i.e. the sequence of features (spiculate-skeleton, megascleres, acanthose).

As we have explained above, there is an order relation between sorts. Figure 2 shows the sort/subsort hierarchy for the values of the feature megascleres. The most general sort allowed for the values of the feature megascleres is *megas-form*

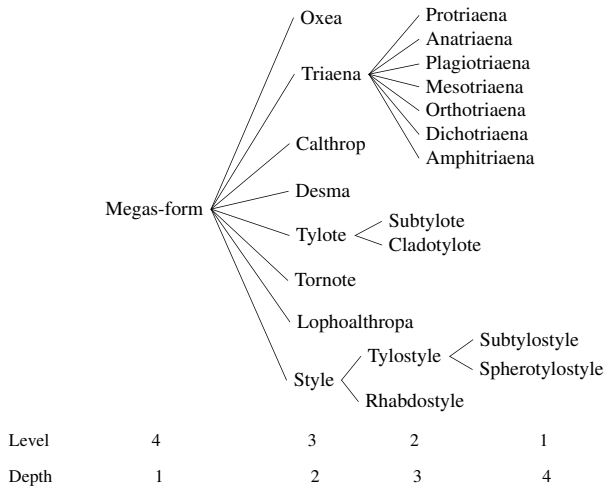

Fig. 2. Part of the sort hierarchy corresponding to the values of the feature *megascleres*. It is a hierarchy of depth 4. The levels of the hierarchy will be used to determine the sort similarity.

and there are several subsorts (e.g. *triaena, style, calthrop,* etc). In turn, some of these subsorts (e.g. *triaena, style, tylote*) have subsorts. Let us suppose that $v_1 = protriaena$, $v_2 = anatriaena$ and $v_3 = tylostyle$. The least upper bound of v_1 and v_2, *lub*(v_1, v_2), is *triaena* that is the most specific supersort of *protriaena* and *anatriaena* whereas the *lub*(v_1, v_3) is *megas-form* since *protriaena* and *tylostyle* only share the fact that both are kinds of megascleres.

There are two important concepts concerning the representation using feature terms that will be used later for assessing similarity. One of them is the *subsumption* relation and the other one is the *anti-unification* operation. Both concepts are explained in the next section.

2.1 Subsumption and Anti-unification of Feature Terms

The semantic interpretation of feature terms brings an ordering relation among feature terms that we call *subsumption*. Intuitively, a feature term ψ subsumes another feature term ψ' ($\psi \sqsubseteq \psi'$) when all information in ψ is also contained in ψ'. More formally, a feature term ψ subsumes other feature term ψ' when the following conditions are satisfied:

1. the sort of $root(\psi')$ is either the same or a subsort of $root(\psi)$,
2. if F_ψ is the set of features of ψ and $F_{\psi'}$ is the set of features of ψ' then $F_\psi \subseteq F_{\psi'}$ and
3. the values of the features in F_ψ and $F_{\psi'}$ satisfy the two conditions above.

Figure 3 shows the feature term *s-encrusting* representing sponges that have an spiculate skeleton and that grow in encrusting form. The sponge *s364* in

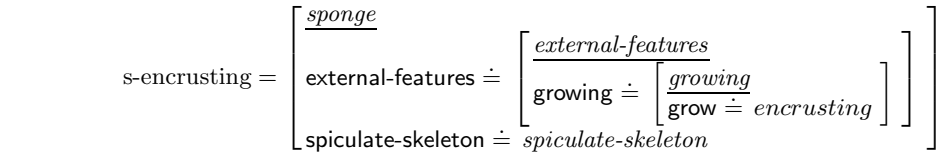

Fig. 3. Description of marine sponges with spiculate skeleton that grow encrusting.

Figure 1 is subsumed by this description (*s-encrusting* $\sqsubseteq$ *s364*) since all the information in *s-encrusting* is also contained in *s364* – although *s364* can have more (or more refined) information.

In [1] can be found a more formal explanation about the feature terms and the subsumption relation.

Feature terms form a partial ordering by means of the subsumption relation. The *anti-unification* is defined over the subsumption lattice as an upper lower bound with respect to the subsumption ($\sqsubseteq$) ordering.

Intuitively, the anti-unification (AU) of two feature terms gives what is common to both (yielding the notion of generalization) and all that is common to both (the most specific generalization). Therefore, the AU of two feature terms F_1 and F_2 produces a feature term D that contains the features that are common to both F_1 and F_2. The values of the features in D have to satisfy the following conditions:

1. If a feature f has the same value v in both examples F_1 and F_2, the value of f in D is also v.
2. In a feature f has value of sort s_1 in F_1 and value of sort s_2 in F_2, the value of f in D is the most specific sort common to s_1 and s_2, i.e. the least upper bound of s_1 and s_2 in the $\preceq$ sort order.
3. otherwise, the examples F_1 and F_2 cannot be anti-unified.

The features of feature terms can be set-valued. Let f_k be a feature that takes the set V_1 as value in F_1 and the set V_2 as value in F_2. Intuitively, the AU of V_1 and V_2 has to produce as result a set W. The cardinality of the set W is $MinCard = min(Card(V_1), Card(V_2))$ and each element in W is the AU of a value of V_1 and a value of V_2 (obtaining the most specific combination).

The elements in W are obtained as follows. First the set $C = \{(x_i, y_j) \mid x_i \in V_1 \text{ and } y_j \in V_2\}$ is obtained. Then the AU of each pair in C is computed. Finally, the set W contains the $MinCard$ most specific compatible combinations of values.

Given the feature terms $w_1 = AU(x, y)$ and $w_2 = AU(x', y')$ we say that w_1 and w_2 are *compatible* when $x \neq x'$ and $y \neq y'$. Otherwise w_1 and w_2 are *incompatible*. Intuitively, two anti-unification feature terms are compatible if they have both been obtained from the AU of different values. This means that the values of the sets to be anti-unified have been used only once.

Table 1. Example of the anti-unification of sets. V_1 = {anatriaena, desma} and V_2 = {anatriaena, protriaena, calthrop}. Columns 1 and 2 form all the possible combinations of the values of V_1 and V_2 . The third column shows the anti-unification of each possible combination of the values of V_1 and V_2.

$v \in V_1$	$u \in V_2$	AU(v, u)
anatriaena	anatriaena	anatriaena
anatriaena	protriaena	triaena
anatriaena	calthrop	megas-form
desma	anatriaena	megas-form
desma	protriaena	megas-form
desma	clathrop	megas-form

For instance, let us suppose that we want to obtain the anti-unification of the sets $V_1 = \{$*anatriaena, desma*$\}$ and $V_2 = \{$*anatriaena, protriaena, calthrop*$\}$. The first step is to build the set C of all the possible combinations of values. In this example $C = \{$ *(anatriaena, anatriaena), (anatriaena, protriaena), (anatriaena, calthrop), (desma, anatriaena), (desma, protriaena), (desma, calthrop)*$\}$. The table 1 shows the anti-unification of each combination using the sort hierarchy in Figure 2. The anti-unification of V_1 and V_2 is a set of cardinality 2 (since $|V_1| = 2$) containing the two most specific compatible values obtained from the anti-unification. In the example, the most specific sort is *anatriaena* obtained from *AU(anatriaena, anatriaena)*. This means that all the combinations using *anatriaena* $\in V_1$ and *anatriaena* $\in V_2$ have to be eliminated because they are incompatibles with *AU(anatriaena, anatriaena)*. In this case all the remaining values have the same anti-unification (i.e. *megas-form*), thus the result is the set $W = AU(V_1, V_2) = \{$*anatriaena, megas-form*$\}$.

3 Similarity between cases

There are three aspects that we need to define in order to perform CBR on relational cases: 1) to define a *case* from a constellation of relations, 2) to assess the similarity of values, and 3) to define a way to assess similarity between cases. These three aspects are explained in this section.

A *case* is a term defined (in feature terms) by two parameters: a *root sort* and a *depth*. That is to say, assuming a "case base" expressed as a collection of feature terms, a case is a feature term whose root node is subsumed by the *root sort* and whose depth is at most *depth*. An example of case specification is *case*$[\mathsf{root\text{-}sort} \doteq \mathit{sponge}, \mathsf{depth} \doteq 4]$ in the marine sponges domain (see §4).

Section 3.1 explains how to estimate the similarity between the values of a feature and section 3.2 how to estimate the similarity among cases.

3.1 Similarity between values

For the purpose of assessing the similarity of values we distinguish between features having numerical values and features having symbolic values. For those

features with numerical values we use the usual measure. For the features having simbolic values we estimate the similarity using a new distance measure called LAUD.

The similarity between cases is estimated taking into account the similarity of the features describing the cases. Let c_1 and c_2 be two cases represented as feature terms. Given a feature f taking value v_1 in c_1 and value v_2 in c_2, the similarity of the values is estimated depending of whether the values are either symbolic or numeric.

When the value of f is numerical of a range $[a, b]$ the similarity of v_1 and v_2 is computed, as usual, by means of the following expression:

$$sim(v_1, v_2) = 1 - \frac{\mid v_1 - v_2 \mid}{b - a} \tag{2}$$

When the value of f is symbolic, the similarity of two feature values v_1 and v_2 is computed using the hierarchy of the sorts S. The idea is that the similarity between two values depends on the level of the hierarchy where their *lub* is situated with respect to the whole hierarchy: the more general *lub*(v_1, v_2) the greater is the distance between v_1 and v_2.

Formally, let $S_f \in S$ be the most general sort that can take the values of a feature f. We consider S_f as the root of a subsort hierarchy, therefore the *depth* of S_f is 1. Given a subsort s of S_f (i.e. $s \preceq S_f$) we define the *level* of s as follows: $level(s) = M - depth(s)$, where M is the maximum depth of the hierarchy of root S_f.

For instance, in the part of the sort hierarchy in Figure 2 the *lub(anatriaena, orthotriaena) = triaena* is at level 3, whereas *lub(anatriaena, desma) = megasform* is at level 4. This means that *orthotriaena* is more similar to *anatriaena* than *desma*.

Thus, the similarity of two symbolic values will be estimated using the following expression:

$$sim(v_1, v_2) = \begin{cases} 1 & \text{if } v_1 = v_2 \\ 1 - \frac{1}{M} level(lub(v_1, v_2)) & \text{otherwise} \end{cases} \tag{3}$$

Proposition 1. *Let x be a feature term of sort s_1 and y a feature term of sort s_2. We consider a sort hierarchy of root S_f and maximum depth M such that $S_f \preceq s_1$ and $S_f \preceq s_2$ (i.e. s_1 and s_2 are subsorts of S_f by the $\preceq$ relation). In that situation the following measure is a distance*

$$\delta(x, y) = \begin{cases} 0 & \textit{if } x = y \\ \frac{1}{M} level(lub(x, y)) & \textit{otherwise} \end{cases} \tag{4}$$

Proof. A measure is a distance if the following three conditions are satisfied:

1. $0 \leq dist(x, x)$
2. $dist(x, y) = dist(y, x)$
3. $dist(x, y) \leq dist(x, z) + dist(z, y)$

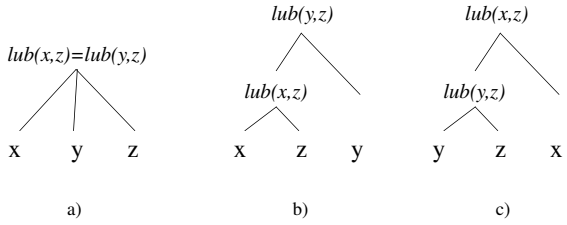

Fig. 4. Different relations that can occur between the sorts of three objects.

We will proof that the measure $\delta(x, y)$ satisfies the three conditions above.

1. $\delta(x, x) = 0$ according to the definition
2. The symmetry of $\delta(x, y)$ is proved using the symmetry of *lub*'s
   $\delta(x, y) = \frac{1}{M} level(lub(x, y)) = \frac{1}{M} level(lub(y, x)) = \delta(y, x)$
3. To proof the triangle inequality we need to distinguish three cases, corresponding to the situations illustrated in Figure 4 (remember that S is restricted to single inheritance). Let z be a feature term of sort s_3.
   - a) $lub(x, z) = lub(z, y)$
     we can assure (Figure 4a) that also $lub(x, y) = lub(x, z)$, therefore
     $\frac{1}{M}$ *level(lub(x, y))* $= \frac{1}{M}$ *level (lub(x, z))*, and thus
     $$\delta(x, y) = \delta(x, z) \leq \delta(x, z) + \delta(z, y)$$
   - b) $lub(x, z) \preceq lub(z, y)$
     we can assure (Figure 4b) that *lub(x, y) = lub(z, y)* therefore
     $\frac{1}{M}$ *level(lub(x, y))* $= \frac{1}{M}$ *level (lub(z, y))*, and thus
     $$\delta(x, y) = \delta(z, y) \leq \delta(x, z) + \delta(z, y)$$
   - c) $lub(x, z) \succeq lub(z, y)$
     we can assure (Figure 4c) that *lub(x, y) = lub (x, z)* therefore
     $\frac{1}{M} level(lub(x, y)) = \frac{1}{M} level\ (lub(x, z))$, and thus
     $$\delta(x, y) = \delta(x, z) \leq \delta(x, z) + \delta(z, y)$$

   Therefore $\delta(x, y)$ satisfies the three properties of distance. □

Let us now consider how to estimate the feature similarity when two cases have a set-valued feature f, i.e. when $c_i.f = V_i$ and $c_j.f = V_j$ for some sets $V_i = \{x_1 \ldots x_n\}$ and $V_j = \{y_1 \ldots y_m\}$. In this situation we compute $n \times m$ similarities between the terms in V_i and the terms in V_j, i.e for all possible pairs $P_{ij} = \{(x_h, y_k) | x_h \in V_i \wedge y_k \in V_j\}$ we compute the similarity score using the previously defined feature similarity functions. Clearly, there will be $min(n, m)$ mappings that we can establish. Moreover, since we are interested in minimizing distance,

we want those pairs of P_{ij} that have lesser distances and, taken together, they have the smaller aggregate distance.

For the case when V_i and V_j are numeric we compute the similarities of all pairs in P_{ij} obtaining the set $\mathcal{S}(P_{ij})$ as follows:

$$\mathcal{S}(P_{ij}) = \{\langle (x_h, y_k), \delta(x_j, y_k)\rangle | x_h \in V_i \wedge y_k \in V_j\}$$

Let us call the set of possible pairs $S_0 = P_{ij}$. We take the pair $p_1 = (x, y) \in S_0$ with maximum similarity in $\mathcal{S}(P_{ij})$. Now, this pair and all the pairs incompatible with it have to be removed from the set S_0, i.e. we build the set $S_1 = S_0 - P_1$ where $P_1 = \{(x', y') \in S_0 | x' = x \vee y' = y\}$. Next, we take the pair p_2 from the remaining pairs in S_1 with maximum similarity in $\mathcal{S}(P_{ij})$. We proceed this way until we find all pairs $P_{min} = \{p_1 \dots p_{min(n,m)}\}$ that together have a maximum value of similarity. Then the feature similarity is the following aggregate:

$$sim(V_1, V_2) = \frac{1}{min(n, m)} \sum_{(x_h, y_k) \in P_{min}} sim(x_h, y_k)$$

For the case when sets V_i and V_j have symbolic values the idea is also the same: finding those pairs whose similarity is higher. As we have seen, for a pair (x_j, y_k) of symbolic values the more specific their $lub(x_j, y_k)$ the higher is their similarity. Therefore we want to find the collection of pairs $\{p_1 \dots p_{min(n,m)}\}$ whose *lub*s are more specific. But this is precisely the definition of anti-unification shown in §2.1. Therefore the anti-unification of the values $V_i = \{x_1 \dots x_n\}$ and $V_j = \{y_1 \dots y_m\}$ provides the pairs that have the highest similarity.

Let $W = \{w_1 \dots w_{min(n,m)}\}$ be the anti-unification of V_i and V_j. Each $w_l \in W$ is the result of the anti-unification of a pair (x_h, y_k) such that $x_h \in V_1$ and $y_k \in V_2$. Let us call P_W the set of those pairs:

$$P_W = \{(x_h, y_k) | x_h \in V_1 \wedge y_k \in V_2 \wedge AU(x_h, y_k) \in W\}$$

Then the aggregate similarity for sets V_1 and V_2 is as follows:

$$sim(V_1, V_2) = \frac{1}{min(n, m)} \sum_{(x_j, y_k) \in P_W} sim(x_j, y_k) \tag{5}$$

3.2 Aggregated similarity

The similarity among cases has to be computed as an aggregation of the similarities of the feature values. In propositional cases global similarity $S(c_1, c_2)$ is usually a mean aggregation function Φ from feature-wise similarities over a fixed collection of attributes $F = (f_1, \dots, f_m)$, i.e.

$$S(c_1, c_2) = \Phi(sim(c_1.f_1, c_2.f_1), \dots, sim(c_1.f_m, c_2.f_m))$$

where $sim(c_1.f_j, c_2.f_j)$ is the feature similarity for the values of f_j. However, in practice cases can be incomplete, so pseudo-similaritys have to be included in

that global similarity assessment, e.g. enforcing $sim(c_1.f_j, c_2.f_j) = 0$ whenever one of the two cases has a missing value in f_j.

In our experiments with relational case-based learning we have focused on exploiting the information present in the object-centered formalisms, namely *sorts* and *features*. Sorts express domain knowledge that clusters domain objects into meaningful classes; in fact, the estimation of the similarity of symbolic values using the sort hierarchy (see equation 3) captures this domain information. Information about features can be exploited because we do not (and can not) assume, as in propositional cases, that a case should have all features $F = (f_1, \ldots, f_m)$.

Using feature terms, the features with unknown value are represented by the value *any* for that features. Because *any* is the minimal element on the $\preceq$ relation, when a case has value *any* in a feature f we can think of this case as not having the feature f. For instance, the sponge *s364* in Figure 1 has two skeletons: one spiculate skeleton and one skeleton with tracts (feature tracts-skeleton) whereas the sponge *s252* in Figure 5 has only one spiculate skeleton and thus the feature tracts-skeleton does not occur in *s252*.

The similarity between two cases c_1 and c_2 is estimated taking into account both the information that they share and the information that they do not share. The *shared features* is the set of features occurring in both cases and it is obtained using the anti-unification operation (section 2.1). The *relevant features*, are those features that occurs at least in one of the two cases (notice that the relevant information also includes the shared features). The global similarity of two cases is computed by estimating the similarity of the shared features and then normalizing this result by the cardinality of the set of relevant features.

A feature term ψ can be viewed as the set $\Pi(\psi, k) = \{\pi(\psi, f_i) \mid f_i \in leaves(\psi, k)\}$, i.e. the set of paths from the root to the leaves with depth k. The anti-unification of two cases $c = AU(c_1, c_2)$ captures the common structure on these two cases. Moreover, since c is a feature term it can also be viewed as a set of paths $\Pi(c, k)$ that collects that which is common to c_1 and c_2. We assess the similarity of two cases using the feature-wise similarity measure defined in section 3.1 on the set of leaves of $c = AU(c_1, c_2)$; let $\mathcal{A}(c_1, c_2)$ denote that set of leaves. Assessing the similarity of the leaves of the paths in $\Pi(c, k)$ we are implicitly assessing the common structure of two cases represented by $\Pi(c, k)$.

Moreover, we want to take into account those features not shared by two cases. For this purpose we define the set of relevant leaves as follows $\mathcal{L}(c_1, c_2) = \{f_j | f_j \in leaves(c_1, k) \vee f_j \in leaves(c_2, k)\}$. The ratio $\mid \mathcal{A}(c_1, c_2) \mid / \mid \mathcal{L}(c_1, c_2) \mid$ estimates how great is the shared structure between two cases. Therefore we will use $\mathcal{L}(c_1, c_2)$ to normalize the aggregation of feature-wise similarities, as follows:

$$S(c_1, c_2) = \frac{\sum_{f_j \in \mathcal{A}(c_1, c_2)} sim(\pi(c_1, f_j), \pi(c_2, f_j))}{\mid \mathcal{L}(c_1, c_2) \mid} \tag{6}$$

where $sim(\pi(c_1, f_j), \pi(c_2, f_j))$ is the feature-wise similarity measure applied to the values of f_j in the cases c_1 and c_2. Notice that the similarity is computed over the values in c_1 and c_2 defined by the paths found in the anti-unification.

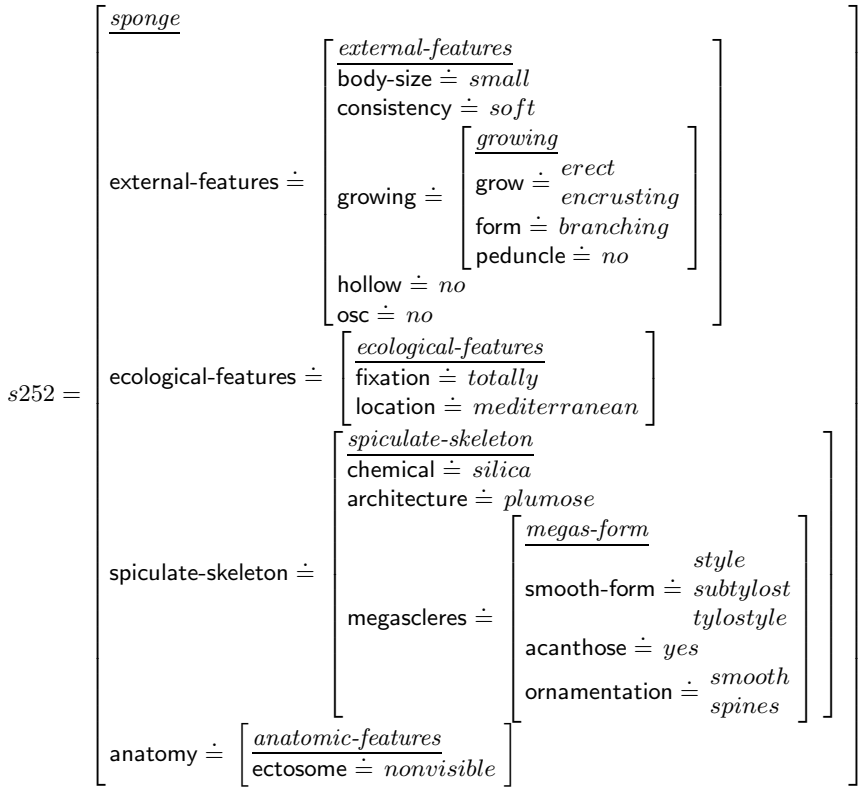

Fig. 5. Representation of a sponge using feature terms.

3.3 Example

Let us suppose that the goal is to estimate the similarity of the sponges *s364* and *s252* (Figures 1 and 5 respectively). The first step is to determine the set of leaf features of each sponge, and then the set $\mathcal{A}$ of leaf features that are common to both sponges. In our examples the set $\mathcal{A}$ is the following:

$\mathcal{A} = \{$(external-features body-size) (external-features growing grow)
(external-features growing form) (external-features growing peduncle)
(external-features hollow) (external-features osc)
(ecological-features fixation) (ecological-features location)
(spiculate-skeleton chemical) (spiculate-skeleton architecture)
(anatomy ectosome) (spiculate-skeleton megascleres smooth-form)
(spiculate-skeleton megascleres acanthose)
(spiculate-skeleton megascleres ornamentation) $\}$

The second step is to evaluate the similarity of the values that the features in $\mathcal{A}$ take in the current sponges. This similarity is the following:

- (external-features body-size). Both sponges have the same symbolic value (*small*) in this feature, therefore the similarity is 1. The following features also have similarity 1: (external-features growing peduncle), (external-features hollow), (external-features osc), (ecological-features fixation), (ecological-features location), and (spiculate-skeleton chemical).
- (external-features growing grow). The sponge *s252* has as value the set $V_1 = \{erect\ encrusting\}$ and the sponge *s364* has as value the set $V_2 = \{encrusting\}$. The similarity for this feature is estimated according to Equation 5. Therefore the first step is to determine the elements of V_1 and V_2 providing the anti-unification of both sets and then to estimate the similarity of the values belonging to this anti-unification.

  The anti-unification is $AU(V_1, V_2) = \{encrusting\}$ that has been obtained from the anti-unification of $encrusting \in V_1$ and $encrusting \in V_2$. Since both values are the same $sim(encrusting, encrusting) = 1$, and since the minimum cardinality is 1, the result following Equation 5 is 1.
- (external-features growing form). The sponge *s252* has as value $v_1 = branching$ and the sponge *s364* has as value $v_2 = digitate$. Both values are symbolic, therefore according to Equation 3 *lub(branching, digitate) = form*. Since *form* is the root of the sort hierarchy for the legal values of the feature form, the similarity on this feature is zero. The features (spiculate-skeleton architecture), and (anatomy ectosome) have also similarity zero.
- (spiculate-skeleton megascleres smooth-form). This feature takes as value the set $V_1 = \{style,\ subtylostyle,\ tylostyle\}$ in the sponge *s252* and the set $V_2 = \{spherotylostyle\}$ in the sponge *s364*. Following the Equation 5 we compute the anti-unification $AU(V_1, V_2) = tylostyle$. Since this value has been obtained from $subtylostyle \in V_1$ and from $spherotylostyle \in V_2$ following Equation 5 we have to compute

$$sim(subtylostyle, spherotylostyle) = 1 - \frac{1}{M} level(lub(subtylostyle, spherotylostyle)) = \frac{2}{4} = 0.5$$

  where *lub(subtylostyle, spherotylostyle) = tylostyle*, *level(tylostyle) = 2*, and $M = 4$ (see Figure 2).

Finally, the similarity of both cases is estimated as the sum of the similarities computed above normalized by the cardinality of the set of features that are in some of the both sponges.

$$S(s252, s364) = \frac{\sum_{f_j \in \mathcal{A}} sim(\pi(s252, f_j), \pi(s364, f_j))}{\mid \mathcal{L}(s252, s364) \mid} = \frac{11}{24} = 0.458$$

order	N	correct	incorrect	%accuracy
astrophorida	95	88	7	92.63
hadromerida	117	108	9	92.30
poecilosclerida	95	86	9	90.53
TOTAL	307	282	25	91.86

Fig. 6. Results of the application of the similarity to classify marine sponges.

4 Experiments

In this section we describe some experiments that use the similarity to identify the order of marine sponges. Marine sponges are relatively little studied and most of the existing species are not yet fully described. Main problems in the identification are due to the morphological plasticity of the species, to the incomplete knowledge of many of their biological and cytological features and to the frequent description of new taxa. Moreover, there is no agreement on the species delimitation since characterization of taxa is unclear.

We used a case base containing 307 marine sponges belonging to three orders of the *demospongiae* class: *astrophorida, hadromerida* or *poecilosclerida*. The sponges are represented using feature terms as in Figures 1 and 5. Each experiment has been performed using the leave-one-out method. Thus, in one experiment we take out one sponge *sp* and then we compute the similarity of *sp* with each one of the remaining 306 sponges. Finally, *sp* is classified as belonging to the same order than the sponge estimated as more similar.

The Figure 6 shows the results of these experiments, detailing the accuracy, and the number of correct and incorrect answers for each order. Thus, there are 95 sponges in the case-base belonging to the order *astrophorida*. For 88 of these sponges the similarity finds that the most similar sponge is an *astrophorida*, i.e. they are correctly classified. Similarly, 108 of the 117 sponges of the order *hadromerida* and 86 of the 95 sponges of order *poecilosclerida* are correctly classified. Summarizing, from the 307 sponges of the case-base, 282 of them are correctly classified with respect the order where they belong. This represents an accuracy of 91.86%.

5 Related Work

Most of work in similarity assessment has been done in propositional cases but there is an active research field focusing in similarity between relational cases [5, 6, 4, 11, 3]. These approaches use the notion of "structural similarity" and use techniques of subtree-isomorphism or subgraph isomorphism to detect this similarity [6].

Two proposals for similarity measure for object-based representations are [4] and [3]. They both distinguish between inter-class and intra-class similarity. This is because they separate similarity among instances of the same class from

similarity among classes for instances of different classes. However, feature terms unify this distinction using the sort hierarchy. Inter-class similarity in [3] requires the assignement of "similarity degrees" to the class hierarchy while LAUD defines a distance over the sort hierarchy. Both LAUD and [4] support set-valued attributes while [3] does not. LAUD defines precisely the idea of "shared structure" using anti-unification, while [3] requires that every instance to have all defined attributes and [4] uses a similarity measure that is expressed as a system of equations to be solved iteratively.

RIBL [7] is a first-order instance-based learner that applies IBL to an instance space composed of first-order descriptions. The similarity measures used in RIBL are applied to function-free cases. A recent improvement of RIBL [8] allows the use of lists and terms in the representation of the cases, therefore new similarity measures have to be defined in order to work with them. In particular, authors propose the use of edit distances for computing the distances between terms and lists. There are several differences between RIBL and LAUD due to the differences between Horn clauses and feature terms. First, RIBL assumes the cases and the problem are described by a fixed set of attributes, while feature terms do not make this assumption. For this reason LAUD uses anti-unification in order to determine the common features of two cases. Second, LAUD use the sort hierarchy to compute the similarity between symbolic values while RIBL just checks if two symbolic values are equal or not.

Related but different approaches are those in [11] and [2] where the cases are represented using feature terms and the similarity is estimated using the notion of "similarity term". [11] proposes to build the similarity term using the anti-unification (i.e. the most specific generalization of two cases) and then use an entropy measure [10] to assess the similarity term that is better. In [2] we consider that the "similarity term" is a feature term containing the more relevant features allowing the classification of a new case. The relevant features of the cases are determined using a heuristic.

6 Conclusions

In this paper we introduced LAUD, a new distance measure to estimate the similarity of relational cases. In particular, LAUD uses cases represented as feature terms. The representation using feature terms allows on one hand to represent partial information, i.e. the features with unknown values do not appear in the representation of the cases. This means that two cases may be described by different features. On the other hand, the values of the features are sorted and there is an informational order relation ($\preceq$) between sorts. Using the $\preceq$ relation we can define the notion of *least upper bound* and then the anti-unification of two feature terms.

The measure we propose estimates the similarity of relational cases as an aggregation of the similarities of the features that are common to both cases. These common features are obtained using the anti-unification concept. The similarity among features is estimated as usual when they take numerical values.

Nevertheless, when the value of a feature is symbolic the similarity is estimated using the part of the sort hierarchy containing the legal values for that feature.

An interesting fact is that our extension of similarity assessment to relational cases is that LAUD is not much more computationally intensive than a distance-based assessment for propositional cases. First, LAUD focuses on the leaf features, which is a manageable number of elements. Second, when a leaf feature has a single value (numerical or symbolic) the computation of the distance is straightforward. Only when a leaf feature is set-valued the computation of similarity is somewhat more expensive: the anti-unification operator has to consider the combinations of pairs of values. However, this computation is not so expensive in practice and the cost is worthwhile the added power of relational cases for those CBR applications that require them.

Acknowledgements This work has been supported by Project IBROW (IST-1999-19005). The authors thank Dr. Marta Domingo and Dr. Toni Bermudez for their assistance in collecting the case base of marine sponges.

References

[1] E. Armengol and E. Plaza. Bottom-up induction of feature terms. *Machine Learning Journal*, 41(1):259–294, 2000.

[2] E. Armengol and E. Plaza. Individual prognosis of diabetes long-term risks: A CBR approach. *Methods of Information in Medicine*, page to appear, 2001.

[3] R. Bergmann and A. Stahl. Similarity measures for object-oriented case representations. In *Proc. European Workshop on Case-Based Reasoning, EWCBR-98*, Lecture Notes in Artificial Intelligence, pages 8–13. Springer Verlag, 1998.

[4] G. Bisson. Why and how to define a similarity measure for object based representation systems, 1995.

[5] K Börner. Structural similarity as a guidance in case-based design. In *Topics in Case-Based Reasoning: EWCBR'94*, pages 197–208, 1994.

[6] H Bunke and B T Messmer. Similarity measures for structured representations. In *Topics in Case-Based Reasoning: EWCBR'94*, pages 106–118, 1994.

[7] W. Emde and D. Wettschereck. Relational instance based learning. In Lorenza Saitta, editor, *Machine Learning - Proceedings 13th International Conference on Machine Learning*, pages 122 – 130. Morgan Kaufmann Publishers, 1996.

[8] T. Horváth, S. Wrobel, and U. Bohnebeck. Relational instance-based learning with lists and terms. *Machine Learning Journal*, 43(1):53–80, 2001.

[9] Stephen Muggleton and Luc De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19–20:629–679, 1994.

[10] E. Plaza, R. López de Mántaras, and E. Armengol. On the importance of similitude: An entropy-based assessment. In I. Smith and B. Saltings, editors, *Advances in Case-based reasoning*, number 1168 in Lecture Notes in Artificial Intelligence, pages 324–338. Springer-Verlag, 1996.

[11] Enric Plaza. Cases as terms: A feature term approach to the structured representation of cases. In M. Veloso and A. Aamodt, editors, *Case-Based Reasoning, ICCBR-95*, number 1010 in Lecture Notes in Artificial Intelligence, pages 265–276. Springer-Verlag, 1995.