Proceedings of the Twenty-Third International Florida Artificial Intelligence Research Society Conference (FLAIRS 2010)

Similarity Measures in Hierarchical Behaviours from a Structural Point of View [*]

Gonzalo Florez-Puga and **Belen Diaz-Agudo** and **Pedro Gonzalez-Calero**
Dep. de Ingenieria del Software e Inteligencia Artificial
Universidad Complutense de Madrid
28040, Madrid, Spain

Abstract

Case-Based Reasoning (CBR) systems dealing with complex object-based case representation structures need to employ complex structured-based similarity measures. However, obtaining such similarity requires to solve problems on graphs are known to be NP-complete. In this paper, we show that, in spite of its theoretical complexity, structured-based similarity is of practical use and can be incorporated into the CBR toolbox. We analyze, in terms of quality and efficiency, three different methods for assessing similarity between graphs, which we apply in the domain of behaviour generation for a soccer simulation environment (SoccerBots). Our implementation of such methods has been incorporated into jCOLIBRI, a general framework for CBR development, and ready to be tested on other applications.

Introduction

Although simple similarity measures through feature vectors is the most common approach to assess similarity in Case-Based Reasoning Systems, more complex case representations requiring more sophisticated similarity mechanisms have been investigated (see (Cunningham 2009) for a taxonomical review).

In particular we are interested in similarity measures for cases represented as graphs. A case representation language based on graphs is more expressive than one based on feature vectors, but, unfortunately, the problem of assessing similarity between two graphs is essentially intractable when using methods taking the graph structure into account. The methods for assessing similarity between two graphs are based on finding a subgraph isomorphism, which is an NP-complete problem (Garey and Johnson 1979), or computing some measure of the graph edit distance between the two graphs which is also NP-complete (Bunke and Messmer 1994).

The question that we try to answer in this work is whether we can find, for a particular application, any meaningful difference in terms of quality or execution time of the results between three methods used for graph-based similarity measures. To estimate the quality of the similarity, we compare with a high level description produced by an expert, in the domain of behaviour generation for a soccer simulation environment (SoccerBots).

The rest of the paper runs as follows. The next section briefly describes the methods for structured-based similarity between graphs. Section 3 describes the experiment and its results, while last Section presents the conclusions of the experiments.

Similarity Measures

We have proposed an approach to the similarity problem in graphs, and more specifically in finite state machines (Flórez-Puga, Díaz-Agudo, and González-Calero 2008) that is based in both the structure of the graph and the labeling in the nodes and edges. The labels associated to the nodes are used to express the functionality of the behaviours contained in them.

The next subsections deal with three different approaches to assess this similarity measure.

Edit Distance Based Similarity

This approach is based on the calculation of the edit distance between two graphs (Bunke and Messmer 1994). The distance is obtained as the sum of the operations needed to transform one graph into the other. The set of edit operations we are using is: adding a node (A), deleting a node (D) and editing the label of a node (E), and adding an edge (A'), deleting an edge (D') and editing an edge(E')).

Each operation has an associated cost (C_A, C_D, C_E, etc.). Using different sets of cost values will lead us to different results. In our approach, we are considering the costs of edit operations, not as constants, but as functions defined over the source and target nodes or edges. This way, we can express the intuitive idea that changing one label for another is cheaper in cost if the labels are more similar.

The edit distance ($dist$) is the minimum cost among all sequences of edit operations that transform the source graph into the target graph. The distance can be converted into a similarity measure by defining a function that uses the distance, like:

$$\mathrm{sim}(G_1, G_2) = [1 + dist(G_1, G_2)]^{-1}$$

We also impose the following restrictions on the possible values of the cost functions, so the results of the distance

[*]Supported by the Spanish Ministry of Science and Education(TIN2009-13692-C03-03)

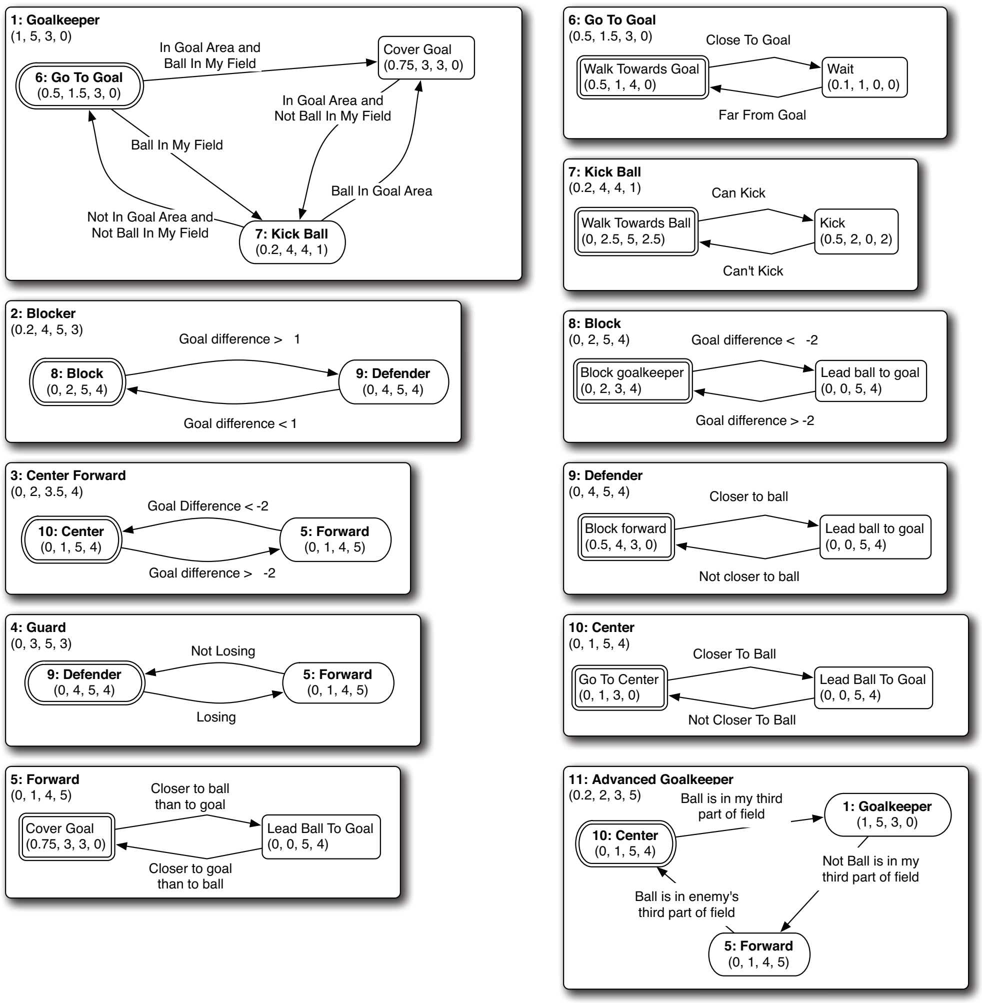

Figure 1: Graphs used in the experiment

function are reasonable:

1. $C_E \leq C_A + C_D$ and $C_{E'} \leq C_{A'} + C_{D'}$
   This means that editing the label of a node is cheaper than an addition and a deletion of the same node with different labels.
2. $C_A = C_D$ and $\mathrm{sim}(X, Y) = \mathrm{sim}(Y, X)$
   These two restrictions give symmetry to our distance measure.

Correspondence Based Similarity

This approach is based in the definition of a correspondence between the nodes of the query and the case graphs. It is based in the similarity measure proposed by (Champin and Solnon 2003).

Each graph G is defined by a triplet $\langle V, r_V, r_E \rangle$ where V is the finite set of nodes, r_V is a relation that associates vertices with labels, and r_E is a relation that associates pairs of vertices (i.e. edges) with labels. r_V and r_E is called the set of features of the graph. A correspondence C between G_1 and G_2 is a subset of $V_1 \times V_2$, that associates, to each vertex of one graph, 0, 1 or more vertices of the other.

Given a correspondence C between G_1 and G_2, the similarity is defined in terms of the intersection of the sets of features (r_V and r_E) of both graphs with respect to C.

We add a value β to each tuple in the intersection. This value represents the similarity between the labels of the nodes or edges:

$$\begin{aligned}
&descr\,(G_1) \cap_C descr\,(G_2) = \\
&\quad \{(v, v', \beta) \mid (v, v') \in C \wedge (v, l) \in r_{V1} \wedge (v', l') \in r_{V2} \wedge \\
&\qquad \beta = \mathrm{sim}\,(l, l')\} \cup \\
&\quad \{\big((v_i, v_j), (v'_i, v'_j), \beta\big) \mid (v_i, v'_i) \in C \wedge \big(v_j, v'_j\big) \in C \wedge \\
&\qquad (v_i, v_j, l) \in r_{E1} \wedge \ \big(v'_i, v'_j, l'\big) \in r_{E2} \wedge \\
&\qquad \beta = \mathrm{sim}\,(l, l')\}
\end{aligned}$$

$$\mathrm{sim}_C\,(G_1, G_2) = \frac{f\,(descr\,(G_1) \cap_C descr\,(G_2)) - g(splits(C))}{F}$$

Where $splits(C)$ is the set of vertices from $V_1 \cup V_2$ which are associated to two or more vertices by the correspondence C.

The similarity degree of two graphs G_1 and G_2 is the maximum similarity of G_1 and G_2 over all the possible correspondences:

$$\mathrm{sim}\left(G_1, G_2\right) = \max_C \left\{\mathrm{sim}_C\left(G_1, G_2\right)\right\}$$

The similarity value β is used by the function f to obtain the final similarity value, and the constant F is an upper bound of f that maintains the result in the interval $[0, 1]$.

To simplify this approach, we can consider only the nodes and edges whose β is greater than a certain threshold.

Weighted Similarity

The third approach is also based in defining the possible correspondences between the graphs being compared, and is based on the one proposed by (Wang and Ishii 1997).

This method doesn't use the intersection, but an algebraic formula to obtain the final similarity measure. As in the previous approach, the similarity degree of two graphs G_1 and G_2 is the maximum similarity of G_1 and G_2 over all the possible correspondences.

The similarity of G_1 and G_2 over the correspondence C is

$$\mathrm{sim}_C\left(G_1, G_2\right) = \frac{F_n + F_e}{M_n + M_e}$$

$$F_n = \sum_{n \in V_1} \frac{W\left(n\right) + W\left(C\left(n\right)\right)}{2} \cdot \mathrm{sim}\left(n, C\left(n\right)\right)$$

$$F_e = \sum_{e \in E_1} \frac{W\left(e\right) + W\left(C\left(e\right)\right)}{2} \cdot \mathrm{sim}\left(e, C\left(e\right)\right)$$

$$M_n + M_e = \max\left\{\sum_{n \in V_1} W\left(n\right), \sum_{n \in V_1} W\left(C\left(n\right)\right)\right\} + \max\left\{\sum_{e \in E_1} W\left(e\right), \sum_{e \in E_1} W\left(C\left(e\right)\right)\right\}$$

where W is the weight of a node or an edge. The weight is a value between 0 and 1 that indicates the importance of a node or an edge in the final similarity result.

Experimental Results

For this experiment we are using Hierarchical Finite State Machines (HFSMs) that represent behaviours for the Soccerbots simulation environment. HFSMs are an extension to traditional Finite State Machines. In a HFSM there are two kind of nodes: atomic nodes, that contain actions that can be executed, and composite nodes, that contain a subordinate HFSM. This hierarchical organization can help to reduce the overall complexity of the state machine, favoring its legibility.

Figure 1 shows the testing set TS of HFSMs used for the experiment. The bold-typed ones are composite behaviours that reference another HFSM. The ones in plain type are atomic behaviours that cannot be further decomposed.

Each behaviour, wether it is atomic or composite, has a set of attributes used to describe them. The attributes we are using for Soccerbots are:

- Goalkeeper: proficiency as goalkeeper. Can take real values in the interval $[0, 1]$.
- Defender: proficiency as defender. Its valid interval is $[0, 5]$.
- Mobility: ability to move around the playing field. Can take real values between $[0, 5]$.
- Attacker: proficiency as attacker. Its valid interval is $[0, 5]$.
- Description: a natural language description of the behaviour. It's value can be any string.

All the behaviours in TS have been classified by an expert using this set of attributes. Their values are shown in Figure 1, under the name of each behaviour. For the sake of clarity we have omitted the attribute names and the textual description.

Experimental Similarity Measures

As we have seen, to completely specify a graph similarity function we need two more similarity functions, one for nodes and another one for edges.

To measure the edge similarity we use a function based on the Levenshtein distance on the edge labels (Levenshtein 1966).

To obtain node similarity we used two different functions:

- Attribute based similarity: the similarity is given by the average of the similarity of each attribute of the behaviour in the node, including the textual description:

$$sim_{ATTR}(n_1, n_2) = sim_{AS}(n_1.attrSet, n_2.attrSet)$$
$$sim_{AS}(attrSet_1, attrSet_2) = \sum_{\substack{attr \in \\ attrSet_1 \cap attrSet_2}} \left(\frac{sim_{attr}(n_1.attr, n_2.attr)}{max\{|attrSet_1|, |attrSet_2|\}}\right)$$
$$sim_{attr}(a_1, a_2) = \frac{|a_1 - a_2|}{L}$$

  where $n.attrSet$ is the set of attributes associated to the behaviour in node n and L is the size of the interval of valid values for each attribute.

- Structural similarity: since the graphs we are comparing correspond to HFSMs, if the node is composite it can contain a subordinate HFSM. If this is the case for both of the nodes, we can use a graph similarity measure to compare the subordinate HFSMs:

$$sim(n_1, n_2) = \begin{cases} \bullet\ n_1 \text{ is composite } \wedge n_2 \text{ is composite:} \\ \quad sim_{STR}(n_1.subgraph, n_2.subgraph) \\ \bullet \text{ otherwise:} \\ \quad sim_{ATTR}(n_1, n_2) \end{cases}$$

  Where $n.subgraph$ is the graph corresponding to the subordinate behaviour of a composite node and sim_{STR} is any of the structure based similarity measures in section : Edit Distance Similarity, Correspondence Based Similarity or Weighted Similarity.

Another variation axis is the use of flattened HFSMs. The process of flattening a HFSM consists in transforming it in another state machine that has the same functionality, and in which all the nodes are atomic. We can use the following algorithm to flatten a HFSM:

```
for each composite node n in HFSM
  set G' to the sub-behaviour
  contained in n
  for each edge e that enters n
    change the destination of e to
    the first node of G'cx
  end for
  for each edge e that leaves n
    for each node n' in G'
      create a copy of e and change
      its origin to n'
    end for
    remove e
  end for
end for
remove n
add G' to HFSM
```

The similarity value is obtained flattening the state machines before applying any of the similarity functions. In particular, for our experiment, we flattened the state machines down to the last but one hierarchy level.

In summary, we are using 10 different similarity measures, that we will name as follows:

1. ATTR: Attribute based similarity.
2. EDA: Edit distance with attribute based similarity for sub-behaviours.
3. CSA: Correspondence based similarity with attribute based similarity for sub-behaviours.
4. WSA: Weighted similarity with attributes based similarity for sub-behaviours.
5. EDS: Edit distance using structural similarity for sub-behaviours.
6. CSS: Correspondence based similarity using structural similarity for sub-behaviours.
7. WSS: Weighted similarity using structural similarity for sub-behaviours.
8. EDF: Edit distance using structural similarity for sub-behaviours. The HFSMs are flattened prior to similarity assessing.
9. CSF: Correspondence based similarity using structural similarity for sub-behaviours. The HFSMs are flattened prior to similarity assessing.
10. WSF: Weighted similarity using structural similarity for sub-behaviours. The HFSMs are flattened prior to similarity assessing.

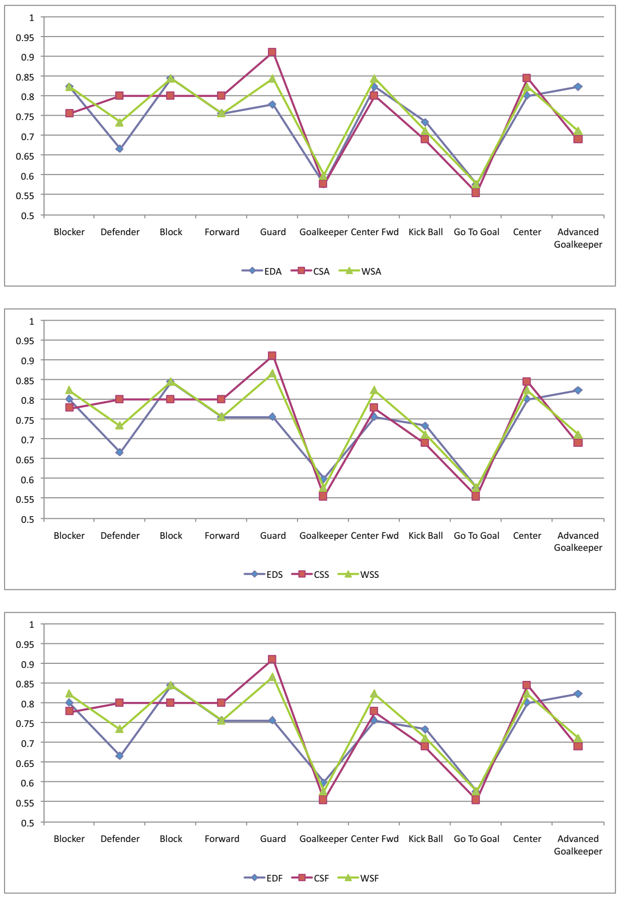

Figure 2: Similarity results

For the correspondence based similarity measures (CSA, CSF and CSS) the functions f and g we used were:

$$f(I) = \sum_{\text{for each node n in I}} (f_N(n)) + \sum_{\text{for each edge e in I}} (f_E(e))$$

$$f_N((v, v', \beta)) = \beta$$

$$f_E(((v_i, v_j), (v'_i, v'_j), \beta)) = \beta$$

$$g(S) = |S|$$

$$F = \max\{|r_{V1}|, |r_{V2}|\} + \max\{|r_{E1}|, |r_{E2}|\}$$

For the weighted measures (WSA, WSF and WSS) we are supposing that the weights for nodes and edges are 1.

Procedure and results

For each similarity measure, sim, and each HFSM, Q, from the test set (TS), our experiment consisted in measuring its similarity to the remaining individuals in the set. Therefore, for each similarity measure and HFSM, we got a list, $L_{(sim,Q)}$, with the rest of HFSMs, sorted by its similarity to Q.

To compare the results of the different measures, we compared each $L_{(sim,Q)}$ with a list obtained applying experts

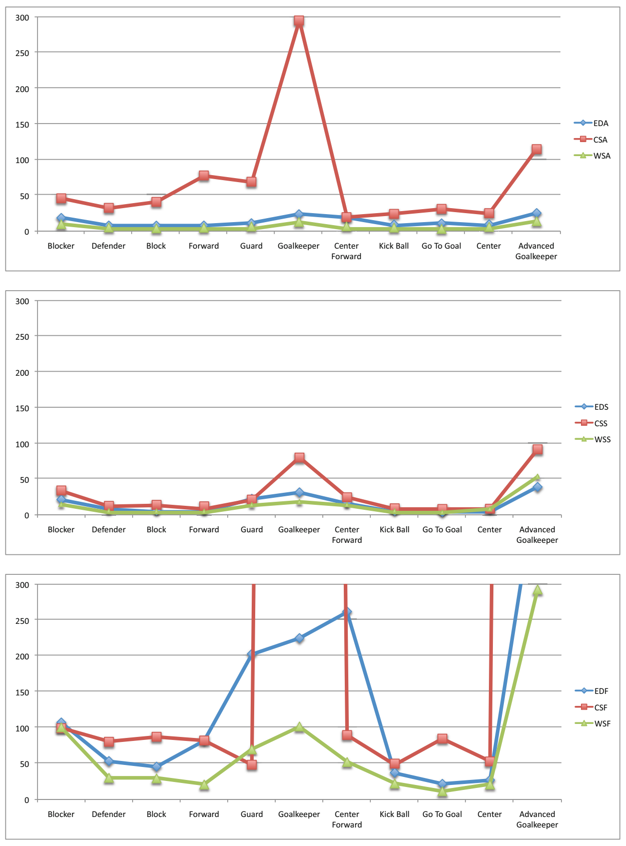

Figure 3: Times measured for the execution of each query

knowledge. In this case, we used the $ATTR$ similarity function with the expert parameters, $L_{(ATTR,Q)}$.

Both lists being compared are permutations of the set $SS = TS \setminus Q$ and, thus, they are equal on size. To compare them, we used the following similarity measure:

$$sim_L(l_1, l_2) = \frac{\sum\limits_{e \in SS} (1 - dist(e, l_1, l_2))}{|SS|}$$

$$dist(e, l_1, l_2) = \frac{|pos_{l_1}(e) - pos_{l_2}(e)|}{|SS|}$$

Where $pos_l(e)$ is the position of element e in the list l.

Figure 2 shows the similarity results obtained for each measure using each graph of TS as query.

Conclusions and Future Work

Regarding quality of the results, we have found no meaningful difference between the three methods. As Figure 2 shows, the three methods provide highly similar results along all the examples in the corpus.

Regarding efficiency, on the other hand, we can conclude that for this particular domain, the WSS method consistently outperform the other two, as can be seen in Figure 3. Nevertheless, further experimentation should be done in order to characterize the properties of the soccerbots domain that are responsible for these results.

All three methods follow the general algorithm shown in Listing 1, differing in the assignment of the initial list of correspondences (in line 2) and the similarity function used to obtain the similarity of two graphs given a correspondence (line 7). Edit distance similarity and weighted similarity use one-to-one correspondences, that is, each node in the first graph is mapped to one node in the second (in the event that one graph has more nodes than the other, the extra nodes will be mapped to $\emptyset$). The correspondence based method uses a many-to-many correspondence in which each node is mapped to zero, one or more nodes of the other graph. In the case of weighted similarity, we also made test using many-to-many correspondences, which gave similar times to the ones obtained for the correspondence based methods.

The reason for this is that the similarity function is evaluated once per correspondence in the correspondence list LC. Given two graphs, G_1 and G_2, in the case of one-to-one correspondences it means that it is evaluated $\max\{|N_{G_1}|, |N_{G_2}|\}!$, being N_{G_i} the set of nodes of graph G_i. On the other hand, for many-to-many correspondences, the similarity function is evaluated $2^{|N_{G_1}|\cdot|N_{G_2}|}$ times.

This also explains the peaks found for the goalkeeper and the advanced goalkeeper behaviours. Both behaviours have 3 nodes each. In the worst case, when they are being compared one against the other, the number of correspondences is 3! for one-to-one correspondences and 2^9 for many-to-many. If we flatten the graphs, this difference is even greater. The flattened Goalkeeper behaviour has 5 nodes and the Advanced Goalkeeper, when flattened, has 9. For the one-to-one correspondences, we have $9! = 362880$ different cases, and for many-to-many correspondences the number grows to $2^{5\cdot 9} = 35.18 \cdot 10^{12}$

For the typical cases we are dealing with, with graphs sizes between two and six nodes, the one-to-one correspondence leads to better results in the number of evaluations of similarity function.

In any case, heuristic methods can be found to prune the search space, so we can deal with bigger graphs. These methods depend on the graph, node and edge similarity functions and demand further research.

Although in this paper we have focused in similarity, our work is also related to reuse as more similar cases are more easily adaptable and more applicable in the query situation. As future work we will consider if there are dependencies between how useful and reusable were the structurally similar cases. An advantage that the approach discussed in the paper is that it already takes into account some amount of the effort needed to transform one case to another.

References

Bunke, H., and Messmer, B. T. 1994. Similarity measures for structured representations. In *EWCBR '93: Selected papers from the First European Workshop on Topics in Case-Based Reasoning*, 106–118. London, UK: Springer-Verlag.

Champin, P. A., and Solnon, C. 2003. Measuring the similarity of labeled graphs. In Ashley, K. D., and Bridge,

```
similarity(Q: Graph, G: Graph)
  generate a list LC of correspondences
      between nodes of Q and G
  set best_sim to $-\infty$
  for each correspondence C in LC
    add_Required_Edges(C, Q, G)
    set current_sim to $\mathrm{sim}_C(Q,G)$
    if best_sim < current_sim then
      set best_sim to current_sim
  end for
end


add_Required_Edges (C: Correspondence,
    Q: Graph, G: Graph)
  for each edge $e$ in Q
    set $n_o$ to origin of e
    set $n_t$ to target of e
    set $n'_o$ to C($n_o$)
    set $n'_t$ to C($n_t$)
    if there is an edge $e'$ from $n'_o$ to $n'_t$
      add the pair $(e,e')$ to C
      mark $e'$ as visited
    else
      add the pair $(e,\emptyset)$ to C
    end if
  end for
  for each edge $e'$ in G
    if $e'$ is not marked
      add the pair $(\emptyset,e')$ to C
end
```

Listing 1: Pseudo-code for the general similarity assessment

D. G., eds., *5th Int. Conf. On Case-Based Reasoning (ICCBR 2003)*, LNAI, 80–95. Springer.

Cunningham, P. 2009. A taxonomy of similarity mechanisms for case-based reasoning. *IEEE Transactions on Knowledge and Data Engineering* 21(11):1532–1543.

Flórez-Puga, G.; Díaz-Agudo, B.; and González-Calero, P. 2008. Experience-based design of behaviours in videogames. In Springer-Verlag., ed., *Proceedings Of The 9th European Conference on Case Based Reasoning*, 180–194.

Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Levenshtein, V. I. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady* 10(8):707–710.

Wang, Y., and Ishii, N. 1997. A method of similarity metrics for structured representations. *Expert Systems with Applications* 12(1):89–100.