# Study and Formalization of a Case-Based Reasoning System with a Description Logic *

**Sylvie Salotti, Véronique Ventos**
LIPN-CNRS URA 1507,
Université Paris-Nord
93430 Villetaneuse France.
Tel: 33 1 49 40 36 09
email: <firstname>.<lastname>@lipn.univ-paris13.fr

## Abstract

In this paper the C-CLASSIC *description logic* is used to design a *Case-Based Reasoning system* with homogeneous, explicit and formal criteria. The case base is organized by means of a taxonomy of *index concepts*. Case retrieval is performed using the *automatic concept classification* of the description logic. *Similarity* between two cases is characterized by the most specific concept which subsumes the two cases (*Least Common Subsumer* or LCS), and *dissimilarity* by a concept representing properties which belong to one case but not to the other. A partial order induced by the *subsumption relation* on these concepts is used to select the most similar cases. Finally, the paper proposes a *supervised learning step* of index concepts using a learning algorithm whose main operation is LCS computation.

**Key-words:** Case-Based Reasoning, Similarity.

## 1 Introduction

In most systems the different tasks of Case-Based Reasoning (CBR) are performed using heterogeneous methods which generally rely on ad hoc, application-dependent criteria. What is proposed here is to use a Description Logic (DL) to design a CBR system with homogeneous, formal, application-independent criteria. The ability to automatically build taxonomies of concepts and to link individuals to concepts is an interesting property for the management of a case base. Besides, DLs are restrictions of First Order Logic in which the subsumption relationship has been largely and formally studied.
C-CLASSIC (see [2]) is a description logic expressive enough to be of practical use in which the computation of subsumption and *least common subsumption* are polynomial and in which it is possible to learn with respect to the Valiant paradigm (i.e. C-CLASSIC is PAC-learnable see [12] and [5]). This paper defines homogeneous, explicit criteria in C-CLASSIC to formalize the following tasks of CBR:

1. **Organization of the case base**: the stored cases are represented as individuals and are indexed by concepts organized in a taxonomy. The links between *index concepts* are automatically computed using *classification*. The links between cases and index concepts are automatically computed using *realization*.
2. **Case retrieval**: the new case is represented as an individual. The index concepts which the case has to be linked to are found by applying the *realization* operation to this individual. Then, the stored cases which share at least one index concept with the new case are retrieved.
3. **Case selection**: each retrieved case is compared with the new case using two criteria, similarity and dissimilarity. Similarity between two cases is characterized by a computed concept corresponding to the most specific concept which subsumes the two

*This research is supported as part of the "Creation and Enrichment of a Knowledge-Base: Application to the Supervision of a Telephone Network" project involving CNRS-cognisciences, CNET (National Center for Telecommunication Studies),INRIA, LAMIH, LIPN.

cases (called Least Common Subsumer (LCS) in DL). Dissimilarity between two cases is characterized by a computed concept representing the set of properties which belongs to one case but not to the other. The particularity of this approach is to represent the similarity and the dissimilarity between two cases explicitly by concepts. These concepts are compared using the subsumption relationship, and thus a partial order is defined which enables the most similar cases to be selected.

4. **Index concept learning**
   A concept learning algorithm in C-CLASSIC is presented in [14]. The main operation of this algorithm is the computation of LCS. Using this algorithm, the operator is invited to learn a new index concept from the new case and the retrieved similar cases.

In [14], C-CLASSIC has been given an intensional semantics in which concepts are denoted by a *normal form* of the set of their properties rather than the set of their instances as is the case in model-theoretic semantics. These *normal forms* are the data structures handled by the algorithms involved in the tasks described upper.
The next section presents the syntax and semantics of C-CLASSIC; section 3 defines the operations which compute similarity (LCS) and dissimilarity (DISS) between concepts. Section 4 describes the procedures used to perform the case retrieval and the selection of most similar cases. Section 5 introduces supervised learning of index concepts. Finally, this approach is compared with related work in section 6.

## 2 C-CLASSIC

C-CLASSIC is defined using a set **R** of primitive roles, a set **P** of primitive concepts, the constants $\top$ and $\bot$, a set **I** of individuals, and the following syntactic rule ($C$ and $D$ are concepts, $P$ is a primitive concept, $R$ is a primitive role, $u$ is a real, $n$ is an integer and $I_i$ are individuals):

$C, D \rightarrow \top \mid \bot \mid P$ (primitive concept) $\mid ONE\text{-}OF$ $\{I_1 \ldots I_n\}$ (concept in extension) $\mid MIN\, u$ (u is a real number) $\mid MAX u \mid C \sqcap D \mid \forall R : C \mid R\ FILLS\ \{I_1 \ldots I_n\}$ $\mid R\ AT\text{-}LEAST\ n \mid R\ AT\text{-}MOST\ n$
Note that cyclic concept definitions are not allowed.

The theoretical study of the inference mechanisms (completeness, soundness and complexity of inference algorithms) requires to formalize the notions used during the inference processes and especially the subsumption relationship which partially orders concepts of C-CLASSIC. To achieve this task, an algebraic approach (similar to [7], for instance, but for a simpler language) is used (see also [13]). In [14], C-CLASSIC has been given an intensional semantics in which concepts are denoted by a *normal form* of the set of their properties (rather than the set of their instances as is the case in model-theoretic semantics). These *normal forms* are not only elements of the semantics but also the data structures handled by algorithms used in the CBR system. The computation of a *normal form* relies on the equational system EQ which highlights the main properties of the connectives of C-CLASSIC (EQ is completely defined in [14]). The computation of *normal forms* is given in [14], their structure is presented below.

**Definition 1** *A normal form corresponding to a concept $C$ of C-CLASSIC is a tuple (c.dom,c.min,c.max,c.$\pi$,c.r) defined as follows:* **c.dom** *is either a set of individuals if the definition of $C$ contains a ONE-OF property, or the special symbol UNIV otherwise;* **c.min** *(resp.* **c.max***) is either a real if $C$ contains a MIN property (resp. MAX), or the special symbol MIN-R (resp. MAX-R) otherwise;* **c.$\pi$** *is the set of primitive concepts belonging to $C$;* **c.r** *is a set of elements defined as follows: $< R, fillers, least, most, d >$ where $R$ is a name role; fillers is either a set of individuals if $C$ contains an $R$ FILLS property in its definition, or $\emptyset$ otherwise; least is an integer representing an $R$ AT-LEAST property; most is either an integer representing an $R$ AT-MOST property, or the special symbol NOLIMIT otherwise; $d$ is a structure if $C$ contains $\forall R : D$ in its definition.*

**Example :** C $\equiv$ *ONE-OF* {a,b,c} $\sqcap$ P1 $\sqcap$ P2 $\sqcap$ P3 $\sqcap$ *MAX* 4 $\sqcap$ *MAX* 8
The *normal form* of C is ({a,b,c}, *MIN-R*, 4, {P1,P2,P3}, $\emptyset$)

## 3 Operations on C-CLASSIC

Here are described *LCS* and *DISS*, two operations used during *case selection*. Note that these operations are polynomial in time and space.

### 3.1 LCS

In the framework of DL, the notion of LCS has been introduced by Borgida, Cohen and Hirsh in [4]. Computing the Least Common Subsumer of two concepts consists in finding the most specific concept which subsumes both of them.
An LCS algorithm which computes the least common subsumer of two concepts of C-CLASSIC using *normal forms* has been given in [14][1]. This algorithm was proved to be correct, tractable, and it has been shown that the LCS is unique.

[1] Cohen and Hirsh gave an LCS algorithm for C-CLASSIC in [5].

### 3.2 Dissimilarity

Now is defined the *dissimilarity operation* (*DISS*) which characterizes differences between two concepts.
DISS(A,B) is a concept whose the set of properties are A's properties which do not subsume a B's property.
Let A and B be two concepts of C-CLASSIC, DISS(A,B) is such as:
DISS(A,B) $\sqcap$ B $\sqsubseteq$ A.

**DISS algorithm**

We compute the dissimilarity between the concepts $A1$ and $A2$.
Let $a1$ and $a2$ be *normal forms* of $A1$ and $A2$ ($a1$ and $a2 \neq$ *bo* with *bo* denoting $\bot$) with:
$a1$ = (a1.dom, a1.min, a1.max, a1.$\pi$, a1.r) and $a2$ = (a2.dom, a2.min, a2.max, a2.$\pi$, a2.r).
Let *diss* be the result of DISS($a1$,$a2$) = ($diss.dom$, $diss.min$, $diss.max$, $diss.\pi$, $diss.r$).

DISS($a1$,$a2$)
**BEGIN**
• **If** $a2.dom \subseteq a1.dom$ **then** $diss.dom \leftarrow UNIV$
**else** $diss.dom \leftarrow a1.dom$
• **If** $a1.min \leq a2.min$ **then** $diss.min \leftarrow MIN\text{-}R$
**else** $diss.min \leftarrow a1.min$
• **If** $a1.max \geq a2.max$ **then** $diss.max \leftarrow MAX\text{-}R$
**else** $diss.max \leftarrow a1.max$
• $diss.\pi \leftarrow a1.\pi$ - $a2.\pi$
• $diss.r \leftarrow \emptyset$
• **For each** $< R, fillers1, least1, most1, c1 >$ in $a1.r$
• **if** there exists $< R, fillers2, least2, most2, c2 >$ belonging to $a2.r$ **then**
**Begin**
• $fillers3 \leftarrow fillers1$ - $fillers2$
• **if** $least1 \leq least2$ **then** $least3 \leftarrow 0$
**else** $least3 \leftarrow least1$
• **if** $most1 \geq most2$ **then** $most3 \leftarrow NOLIMIT$
**else** $most3 \leftarrow most1$
• **If** c1 subsumes c2 *then* c3 $\leftarrow t$
($t$ denotes $\top$)
*else* c3 $\leftarrow$ c1
• $diss.r \leftarrow diss.r \cup < R, fillers3, least3, most3, c3 >$
**End**
**else** $diss.r \leftarrow diss.r \cup < R, fillers1, least1, most1, c1 >$
**End For each**
**END Example:**
$A \equiv Child \sqcap Tennisman \sqcap \forall age : MIN6 \sqcap \forall age : MAX8$
$B \equiv Child \sqcap Footballer \sqcap \forall age : MIN4 \sqcap \forall age : MAX7$
DISS(A,B) $\equiv Tennisman \sqcap \forall age : MIN6$
DISS(B,A) $\equiv Footballer \sqcap \forall age : MAX7$

**Remark:** if LCS(A,B) subsumes LCS(A,C) then DISS(A,C) subsumes DISS(A,B).

## 4 Retrieval and selection of similar cases

### 4.1 Case retrieval

First, *realization* (with the 'Abstraction-Classification' method) is used to find the most specific index concepts the new case belongs to. Let $C_{new}$ be the abstract concept of the new case and let $E_{C_{new}}$ be the set of the most specific index concepts $C_{new}$ is an instance of, obtained by 'Abstraction-Classification'. Then, the stored cases linked to the index concepts of $E_{C_{new}}$ are retrieved. Let $E_{sim}$ be the set of these cases. This first retrieval offers two advantages: it improves the efficiency of the search by decreasing the number of comparisons between the new case and the stored cases and it insures a minimal, relevant similarity between the new case and each retrieved case.

### 4.2 Selection of the most similar cases

The goal of this step is to select from among the cases belonging to $E_{sim}$ the most similar ones to the new case. This is done using two criteria, similarity and dissimilarity, which are described below.

**Similarity criterion**: the LCS of the new case and a stored case has been chosen here to represent the similarity between the two cases.
**Dissimilarity criterion**: the concepts DISS(new-case, stored-case) and DISS(stored-case, new-case) are used to represent the dissimilarities between the two cases.
The case selection procedure using these two criteria is described below.

**Case selection procedure**

It has been decided here to use the similarity criterion as the main criterion and the dissimilarity criterion where the cases have the same similarity.
The case selection procedure is as follows:

1. First, the system computes the LCS between the new case and each stored case belonging to $E_{sim}$. Since the LCSs obtained are concepts, they can be compared using the subsumption relationship and then ordered in a taxonomy. As the most specific LCSs of the taxonomy have more properties than the others, they "maximize" the similarity with the new case. Let $E_{LCS}$ be the set of these most specific LCSs. To each of these LCSs correspond one or more stored cases of $E_{sim}$. If an LCS corresponds to a single case of $E_{sim}$, this case is selected and put in the set of the most similar cases denoted as $E_{sim+}$. Otherwise, if an LCS corresponds to several cases of $E_{sim}$, the system tries to distinguish between them by applying the dissimilarity criterion as follows.

2. LCS and DISS have the following property: if LCS(A,B) = LCS(A,C) then DISS(A,C) = DISS(A,B). As the cases to be distinguished (noted $C_{old}$) have the same LCS($C_{new}$,$C_{old}$), it is useless to consider DISS($C_{new}$,$C_{old}$). So, for the $C_{old}$ with the same LCS, DISS($C_{old}$,$C_{new}$) are computed and are then compared using the subsumption relationship. As the most general DISSs have fewer properties than the others, they "minimize" the dissimilarity. Consequently, the corresponding cases are selected and put into $E_{sim+}$.

**Example :**
Let $C_{new}$ be the abstract concept of a new case with:
$C_{new} \equiv$ Incident $\sqcap$ affected-people *AT-LEAST* 200 $\sqcap$ inefficient-beam *AT-LEAST* 3
Let $E_{sim} = \{C_{old1}, C_{old2}, C_{old3}, C_{old4}\}$ be the set of the abstract concepts of four stored cases:
$C_{old1} \equiv$ Incident $\sqcap$ affected-people *AT-LEAST* 100 $\sqcap$ inefficient-beam *AT-LEAST* 4
$C_{old2} \equiv$ Incident $\sqcap$ affected-people *AT-LEAST* 300 $\sqcap$ Equipment-dependent $\sqcap$ inefficient-beam *AT-LEAST* 1
$C_{old3} \equiv$ Incident $\sqcap$ affected-people *AT-LEAST* 300 $\sqcap$ Equipment-dependent
$C_{old4} \equiv$ Incident $\sqcap$ affected-people *AT-LEAST* 100 $\sqcap$ inefficient-beam *AT-LEAST* 6 $\sqcap$ Equipment-dependent $\sqcap$ Serious-incident

LCS($C_{new}$, $C_{old1}$) = Incident $\sqcap$ affected-people *AT-LEAST* 100 $\sqcap$ inefficient-beam *AT-LEAST* 3
LCS($C_{new}$, $C_{old2}$) = Incident $\sqcap$ affected-people *AT-LEAST* 200 $\sqcap$ inefficient-beam *AT-LEAST* 1
LCS($C_{new}$, $C_{old3}$) = Incident $\sqcap$ affected-people *AT-LEAST* 200
LCS($C_{new}$, $C_{old4}$) = Incident $\sqcap$ affected-people *AT-LEAST* 100 $\sqcap$ inefficient-beam *AT-LEAST* 3
The most specific LCS are the following: LCS($C_{new}$, $C_{old1}$), LCS($C_{new}$, $C_{old2}$) and LCS($C_{new}$, $C_{old4}$).
$C_{old3}$ is not selected in $E_{sim+}$ as LCS($C_{new}$, $C_{old3}$) strictly subsumes LCS($C_{new}$, $C_{old2}$). $E_{sim+} \leftarrow \{C_{old2}\}$.
LCS($C_{new}$, $C_{old1}$) and LCS($C_{new}$ ,$C_{old4}$) are equal. In order to distinguish between $C_{old1}$ and $C_{old4}$, DISS($C_{old1}$, $C_{new}$) and DISS($C_{old4}$, $C_{new}$) are compared.
DISS($C_{old1}$, $C_{new}$) = inefficient-beam *AT-LEAST* 4.
DISS($C_{old4}$, $C_{new}$) = inefficient-beam *AT-LEAST* 6 $\sqcap$ Equipment-dependent $\sqcap$ Serious-incident.
DISS($C_{old1}$, $C_{new}$) strictly subsumes DISS($C_{old4}$, $C_{new}$) as a consequence, $E_{sim+} \leftarrow \{C_{old2}, C_{old1}\}$.

## 5 Supervised learning of index concepts

The framework of the learning step is learning from instances. [14] defines the Elena+[2] algorithm, which enables the system to learn a C-CLASSIC concept definition from positive instances (or examples) and negative instances (or counter-examples). The main operation of this algorithm is LCS computation, and it has been chosen to learn new index concepts, as described briefly below.
Once the most similar cases have been selected, a set of cases containing the new case and the most similar stored cases is available. The operator is then asked to trigger a learning step, the goal of which is to learn a new index concept from this set of cases. If the operator agrees, he is asked to identify the cases from this set which can be clustered with the new case. These cases define the set of positive instances of the concept to be learned (denoted as *POS*) while the other cases define the set of negative instances[3] (denoted as *NEG*). Finally, the Elena+ algorithm is applied to the sets *POS* and *NEG*.

## 6 Related work

Over the last few years, different work on CBR has revolved around the use of DL formalism and inferences. Koelher ([10]) and Kamp ([8], [9]) use DLs to perform the retrieval task in a case base. Besides, Napoli and al. ([11]) use classification in an object-centered representation to define a similar method of indexation. However, none of these approaches defines coherent criteria w.r.t this indexation in order to select the most similar cases. In [10] Koehler describes a method based mainly on the number of properties shared by the new case and the stored cases. In addition, in order to ensure minimal similarity, retrieved cases must have a minimal number of shared properties with the new case. The drawback of this approach is that it relies on numerical and heuristic criteria, thus losing the explicitness and the formal aspect of the DL classification step. Coupey and Salotti ([6]) have designed a more flexible indexation method which uses a DL including defaults and exceptions in concept definitions to define similarity criteria between index concepts. In the CANDIDE system ([1]) Beck assesses the similarity between two cases using a function called INTERSECT which computes a concept representing the properties shared by the two cases and which is similar to the LCS operation. However, in [1] the presentation is only operational and no formal description is given. Note that INTERSECT is applied to all the stored cases which share one property with the new case, whereas the LCS is only applied to a subset of stored cases retrieved using index concepts. Moreover, Beck computes heuristically a numerical similarity value according to the position in the taxonomy of the concept returned by INTERSECT. The research presented

[2] This algorithm is based on the Elena algorithm defined by Brézellec and Soldano in [3].

[3] These negative instances are nearmisses.

here prefers to define a partial order based only on the subsumption relationship in order to preserve the explicitness and the formal aspect of the approach.

## 7 Conclusion

The work described in this paper has defined explicit, formal and homogeneous criteria to model CBR in C-CLASSIC DL. Using these criteria it is possible to retrieve similar stored cases, to identify similarities and dissimilarities between the retrieved cases and the new case, and to update the case base organization using an index concept learning step. Nevertheless, the formalization given here does not model all the steps of CBR, i.e. retrieval, adaptation and learning. Research must therefore continue in order to include an adaptation step (e.g. representing actions to help solve a case in DL) and to improve the learning step (e.g. defining clustering criteria).

## References


[1] Howard W. Beck. A conceptual clustering algorithm for semantic data models. In *International Workshop on Terminological Logics, Document D-91-13*, pages 17–22, SchloßDagstuhl, Germany, 1991.

[2] A. Borgida and P. F. Patel-Schneider. Complete algorithm for subsumption in the CLASSIC description logic. *Artificial Intelligence Research*, 1:278–308, 1994.

[3] P. Brézellec and H. Soldano. Eléna : a bottom-up algorithm. In Morgan-Kaufmann, editor, *10th International Conference on Machine Learning*, pages 9–16, Amherst, 1993.

[4] W. W. Cohen, A. Borgida, and H. Hirsh. Computing least common subsumer in description logics. In *10th National Conference of the American Association for Artificial Intelligence*, pages 754–760, San Jose, California, 1992.

[5] W. W. Cohen and H. Hirsh. Learning the CLASSIC description logic: Theoretical and experimental results. In *International Conference on Knowledge Representation and Reasoning*, pages 121–133. 1994.

[6] P. Coupey and S. Salotti. Using a description logic with default and exception to formalize the retrieval process of CBR. In *International Congress of Information Processing and Management of Uncertainty*, Granada, Spain, 1996.

[7] R. Dionne, E. Mays, and F.J. Oles. The equivalence of model-theoretic and structural subsumption in description logics. In *13th International Joint Conference on Artificial Intelligence*, pages 710–716, Chambéry, France, 1993.

[8] G. Kamp. On the use of CBR in corporate service and support. In *Proceedings of the 2nd European Workshop on Case-Based Reasoning*, pages 175–183, Paris, France, 1994.

[9] G. Kamp. Using description logics for knowledge intensive case-based reasoning. In Springer-Verlag, editor, *Advances in Case-Based Reasoning, Third European Workshop on Case-Based Reasoning, EWCBR'96*, volume 1168 of *Lecture Notes in Artificial Intelligence*, pages 204–218. I. Smith and B. Faltings, 1996.

[10] J. Koehler. An application of terminological logics to case-based reasoning. In *Principles of Knowledge Representation and Reasoning, Proc. of 5th International Conference*, pages 351–362, Bonn, Germany, 1994.

[11] A. Napoli, J. Lieber, and R. Curien. Classification-based problem-solving in case-based reasoning. In Springer-Verlag, editor, *Advances in Case-Based Reasoning, Third European Workshop on Case-Based Reasoning, EWCBR'96*, volume 1168 of *Lecture Notes in Artificial Intelligence*, pages 295–307. I. Smith and B. Faltings, 1996.

[12] L.G. Valiant. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984.

[13] V. Ventos. A deductive study of c-classic$_{\delta\epsilon}$. In *Actes du Workshop international Description Logics 96*, pages 192–196, Boston, USA, 1996.

[14] V. Ventos. *C-CLASSIC$_{\delta\epsilon}$ : une logique de descriptions pour la définition et l'apprentissage de concepts avec défauts et exceptions.* Thèse, Université Paris-Nord, France, 1997.