Commonality Subtraction Operator for the $\mathcal{EL}$ Description Logic

Axel Mascaro[1], Christophe Rey[1]

[1]Université Clermont Auvergne, CNRS, Ecole des Mines de Saint-Etienne, LIMOS, F-63000 Clermont-Ferrand, France.

Abstract

In the context of the $\mathcal{EL}$ description logic, we define and study a new concept difference operator, called commonality subtraction operator (CSO), with respect to an acyclic definitional ontology $\mathcal{T}$, and noted $A \ominus_{\mathcal{T}} B$. CSO aims at removing from a concept description A all common parts with another description B, w.r.t. $\mathcal{T}$, which we call descriptional commonalities. Based on the proposed operator of tree subtraction (TSO), we give an algorithm to compute CSO along with its complexity. CSO fits well with existential restrictions and applies to any couple of concepts (A, B), which makes it different from existing difference operators. We practically justify the definition of CSO by explaining our needs for such an operator in the context of a metrology resources management project.

Keywords

difference, subtraction, EL, descriptional commonalities, TSO, CSO

1. Introduction

The STAM project[1], funded by the European Regional Development Fund (FEDER) of the European Union, aims at developing a multitool platform in the field of metrology. One of its objectives is to provide a kind of facebook for metrology. In that purpose, it is based on a documentation repository in which metrological resources (e.g. pdf documents, images, texts, data files, instruments...) could be easily retrieved. In its current version, metrological resources are identified by characteristics defined in a metrology dictionary and which are retrieved by a keyword-based search. Besides, resources are also tagged by annotations called "families", another kind of keywords. By selecting one or many families, the user can restrict the results of a keyword search to resources annotated by the chosen families.

In [1], we have improved this exact retrieval process by generalizing it into a matchmaking one, which aims at finding the semantically closest resources with respect to the user query, using the $\mathcal{EL}$ description logic (DL). The choice of $\mathcal{EL}$ is linked to the underlying metrology resource management system which is powered by GraphDB, an RDF data management system that allows tractable reasoning in the OWL2 EL profile, based on $\mathcal{EL}$ [2]. First, leveraging the existing dictionary, an $\mathcal{EL}$ ontology is built by associating logical descriptions to metrological keywords and families in order to obtain $\mathcal{EL}$ concept definitions. Then, user queries that describe

DL 2023: 36th International Workshop on Description Logics, September 2–4, 2023, Rhodes, Greece
axel.mascaro@uca.fr (A. Mascaro); christophe.rey@uca.fr (C. Rey)
0000-0003-3581-9449 (C. Rey)
© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)
[1]https://limos.fr/news_project/118

CEUR-WS.org/Vol-3515/paper-18.pdf

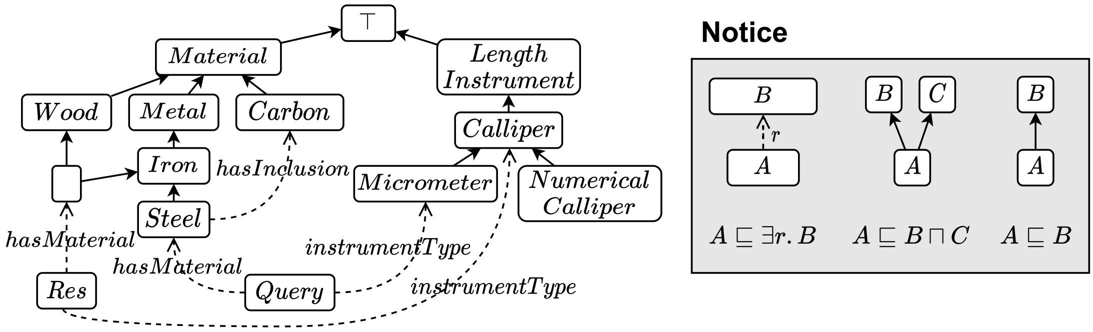

Figure 1: Ontology of the metrology (as an $\mathcal{EL}$ TBox) in example 1.

wanted resources are built as $\mathcal{EL}$ concept descriptions. The semantically closest resources w.r.t. a user query are obtained by pairwisely comparing all resources w.r.t the query, in a semantic way using the ontology. This process produces a ranking of resources w.r.t the query based on the idea that the bigger the shared information between the query and the resources, the better. But instead of directly computing the shared information, we compute what is original in each resource w.r.t. the query. This means best resources are the ones which have the least original parts w.r.t. the query. Moreover, the process also computes what parts of the query are original w.r.t. each resource, which can be used to further refine the ranking if needed. The whole approach is based on a new difference operator for $\mathcal{EL}$, namely the commonality subtraction operator (CSO, noted $\ominus_{\mathcal{T}}$), which is the contribution presented in this paper.

Example 1. *In the metrology context, we may have the following $\mathcal{EL}$ ontology and resource and query descriptions (see figure 1):* $\mathcal{T} = \{Steel \equiv Iron \sqcap \exists hasInclusion.Carbon, Iron \sqsubseteq Metal, Metal \sqsubseteq Material, Wood \sqsubseteq Material, Carbon \sqsubseteq Material, Micrometer \sqsubseteq Calliper, NumericalCalliper \sqsubseteq Calliper, Calliper \sqsubseteq LengthInstrument\}$, $Res = \exists instrumentType.Calliper \sqcap \exists hasMaterial.(Iron \sqcap Wood)$, *and* $Query = \exists instrumentType.Micrometer \sqcap \exists hasMaterial.Steel$. *Intuitively,* $Steel$ *is defined as* $Iron$ *in which is included* $Carbon$, $Iron$ *is a kind of* $Metal$ *which is a kind of* $Material$, *as* $Wood$ *and* $Carbon$. $Micrometer$ *is a kind of* $Calliper$, *as* $NumericalCalliper$, *and* $Calliper$ *is a kind of* $LengthInstrument$. Res *describes a resource that is about a* $Calliper$ *as instrument type, made of* $Iron$ *and* $Wood$. *With* $Query$, *a user is looking for resources about* $Micrometer$ *as instrument type, made of* $Steel$. *We then would like to have: (i)* $Res \ominus_{\mathcal{T}} Query = \exists hasMaterial.Wood$, *which means* $Query$ *shares all aspects of* Res *except the fact that* Res *is a resource about an instrument made of wood, and (ii)* $Query \ominus_{\mathcal{T}} Res = \exists instrumentType.Micrometer \sqcap \exists hasMaterial.\exists hasInclusion.Carbon$, *which means that* Res *shares with* $Query$ *the fact that it describes resources made of* $Iron$, *but does not share other aspects of* $Query$ *(the* $Micrometer$ *instrument type and the* $Carbon$ *inclusion inside the material).*

The CSO ensures two important features: inverse subsumption criterion to define commonalities, and fine-grained difference. The inverse subsumption criterion states that a commonality between the minuend and the subtrahend exists when a part of the minuend subsumes the

Table 1
$\mathcal{EL}$ constructors and axioms. $A \in \mathbf{C}$, $R \in \mathbf{r}$, and C and D are concepts.

Constructors/Axioms	Syntax	Semantics	Remarks
top	$\top$	$\Delta^{\mathcal{I}}$	It is assumed that:
concept name $\in \mathbf{C}$	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$	• conjunctions do not contain $\top$ nor
role $\in \mathbf{r}$	r	$r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$	many times the same conjunct, and
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$	• writing $\bigsqcap_{i=1}^{n} C_i$ means the C_is are
existential restriction	$\exists r.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y \in C^{\mathcal{I}} : (x, y) \in r^{\mathcal{I}}\}$	not conjunctions themselves.
Concept definition	$A \equiv C$	$A^{\mathcal{I}} = C^{\mathcal{I}}$	A appears only once as the lhs of a
Primitive concept definition	$A \sqsubseteq C$	$A^{\mathcal{I}} \subseteq C^{\mathcal{I}}$	definition.

subtrahend. A contrario, in existing operators, the minuend is usually subsumed by (parts of) the subtrahend. This is justifed by our resource retrieval context where the minuend is seen as a query and the subtrahend may answer parts of it: we consider answering part of a query as corresponding to being subsumed by this part of the query. In example 1, considering $Query \ominus_{\mathcal{T}} Res$, $\exists hasMaterial.Iron$ is a commonality between $Query$ and Res since it is a part of $Query$ (once $Steel$ has been replaced by its definition $Iron \sqcap \exists hasInclusion.Carbon$) and it subsumes Res. A contrario, $\exists hasMaterial.Steel$ is not a commonality with Res since it does not subsume Res. The fine-grained difference browses the tree structure implied by existential restrictions in order to precisely remove commonalities between the minuend and the subtrahend without modifying the remaining of the minuend. Going on with the same example, the CSO removes $\exists hasMaterial.Iron$ from $Query$ since it is a commonality with Res, and it keeps $\exists hasMaterial.\exists hasInclusion.Carbon$, and $\exists instrumentType.Micrometer$.

After recalling notions about $\mathcal{EL}$ in section 2, we study CSO for $\mathcal{EL}$ (with an acyclic and definitional TBox) in section 3. In section 4, we relate CSO to other difference operators. At last, we conclude. When not given in the text body, full proofs of properties are given in appendix[2].

2. Recalls about $\mathcal{EL}$

We assume to have two countably infinite sets: $\mathbf{C}$ for concept names and $\mathbf{r}$ for role names. From these, with the help of $\mathcal{EL}$ constructors (see table 1), $\mathcal{EL}$ *concept descriptions* can be built. From now on, unless stated otherwise, the term *concept* refers to the expression "$\mathcal{EL}$ concept description". A concept that is not a concept name nor $\top$ is called a *compound concept*. Given a concept C, we can define its size.

Definition 1 (size [3]). *Given a concept C, its size noted* $\mathsf{size}(C)$ *is defined by induction on its structure: if $C \in \mathbf{C} \cup \top$ then* $\mathsf{size}(C) = 1$*; if $C = C_1 \sqcap C_2$, then* $\mathsf{size}(C) = 1 + \mathsf{size}(C_1) + \mathsf{size}(C_2)$*; and if $C = \exists r.D$ then* $\mathsf{size}(C) = 1 + \mathsf{size}(D)$

Concepts are given a model-theoretic semantics based on *interpretations* which are couples $(\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ of, respectively, a universe of discourse and an *interpretation function*, see the third

[2] https://github.com/Myakko/DL2023-Appendix

column of table 1. Axioms that relate concepts are of the following kinds: *concept definitions* of the form $A \equiv C$ and *primitive concept definitions* of the form $A \sqsubseteq C$. The *size* of an axiom is the sum of the sizes of the left and right hand sides of the axiom. An $\mathcal{EL}$ *TBox*, or just TBox, is a finite set of axioms. The *size* $\mathsf{size}(\mathcal{T})$ of a TBox $\mathcal{T}$ is the sum of the sizes of its axioms. An interpretation $(\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ is a *model* of a TBox $\mathcal{T}$ if, for each axiom in $\mathcal{T}$, the condition given in the third column of table 1 is satisfied. A concept C is *subsumed by* another concept D w.r.t. a TBox $\mathcal{T}$, noted $C \sqsubseteq_{\mathcal{T}} D$ (or $\mathcal{T} \models C \sqsubseteq D$), if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ in every model of $\mathcal{T}$. When $\mathcal{T} = \emptyset$, we can note interchangeably $\sqsubseteq$ or $\sqsubseteq_{\emptyset}$.

A *definitional TBox* contains only concept definitions. A TBox containing primitive concept definitions can be made definitional in linear time w.r.t. $\mathsf{size}(\mathcal{T})$ [3] since (i) each primitive concept definition $A \sqsubseteq C$ can be transformed into the concept definition $A \equiv C \sqcap \overline{A}$, with $\overline{A}$ a new concept name, and (ii) two primitive concept definitions $A \sqsubseteq B$ and $A \sqsubseteq C$ can be grouped into one $A \sqsubseteq B \sqcap C$. In the sequel, TBoxes are supposed to be definitional.

The *signature* of a TBox $\mathcal{T}$, noted $\mathsf{sig}_{\mathcal{T}}$, is the set of all concept names and roles that occur in $\mathcal{T}$. We note $\mathbf{C}_{\mathcal{T}} = \mathbf{C} \cap \mathsf{sig}_{\mathcal{T}}$, $\mathbf{r}_{\mathcal{T}} = \mathbf{r} \cap \mathsf{sig}_{\mathcal{T}}$, and $\mathcal{T}_{\mathcal{EL}}$ the set of all concepts that can be built using elements of $\mathsf{sig}_{\mathcal{T}}$ and $\top$. Concept names appearing as the left-hand side of a definition are called *defined concepts* and they define the set $\mathsf{def}_{\mathcal{T}} \subseteq \mathbf{C}_{\mathcal{T}}$. Defined concepts may only appear once as the left hand side of a concept definition. Other concept names are called *primitive concepts*. They define the set $\mathsf{prim}_{\mathcal{T}} \subseteq \mathbf{C}_{\mathcal{T}}$. The set of concepts built using only primitive concepts of $\mathcal{T}$ and $\top$ is noted $\mathcal{T}_{\mathcal{EL}}^{\mathsf{prim}}$.

Following definition 2.9 of [3], for A, B and B' concept names, we say that A *directly uses* B in $\mathcal{T}$ if there is in $\mathcal{T}$ a primitive concept definition $A \sqsubseteq C$, or a concept definition $A \equiv C$, such that B occurs in C. We say that A *uses* B if A directly uses B, or if there is a concept name B' such that A uses B' and B' directly uses B. A TBox contains a *cycle* when some concept name A uses itself. A TBox is *acyclic* if it contains no cycle. In the sequel, TBoxes are supposed to be acyclic, in addition to being definitional.

The *complete expansion* (a.k.a. unfolding) $\mathcal{T}^*$ of an acyclic definitional TBox $\mathcal{T}$ [4, 5] rewrites every concept definition of $\mathcal{T}$ into an equivalent one with only primitive concepts in its right hand side. Then, for a concept C, $\mathcal{T}^*(C)$ is the complete expansion of C w.r.t. $\mathcal{T}$. This process is EXPTIME in the sizes of $\mathcal{T}$ and C.

Example 2. *We have the following acyclic definitional TBox:* $\mathcal{T} = \{A \equiv B \sqcap C, C \equiv D \sqcap \exists r.B, D \equiv E, F \equiv \exists r.D\}$. *Thus we have* $\mathbf{C}_{\mathcal{T}} = \{A, B, C, D, E, F\}$, $\mathbf{r}_{\mathcal{T}} = \{r\}$, $\mathsf{prim}_{\mathcal{T}} = \{B, E\}$ *and* $\mathsf{def}_{\mathcal{T}} = \{A, C, D, F\}$. *The complete expansion* $\mathcal{T}^*$ *is* $\mathcal{T}^* = \{A \equiv B \sqcap E \sqcap \exists r.B, C \equiv E \sqcap \exists r.B, D \equiv E, F \equiv \exists r.E\}$.

3. The Commonality Subtraction Operator (CSO)

In section 3.1, we define the CSO, first informally, by presenting the notions of characteristic branch and descriptional commonality, and then formally. In section 3.2, we present a new syntactical operator called the tree subtraction operator (TSO) and show how to use it to compute the CSO. We also give the main properties associated to both the TSO and the CSO, namely existence, unicity, and termination, soundness and complexity of the associated algorithms.

3.1. Definition of CSO

The CSO operator $C \ominus_{\mathcal{T}} D$ is intended to remove from the minuend C all concept parts shared with the subtrahend D w.r.t. some TBox $\mathcal{T}$. We call these shared parts $\mathcal{T}$ *descriptional commonalities* (or $\mathcal{T}$ commonalities for short) from C to D. Syntactically, we want commonalities to be removable from the minuend without impacting its other parts. So they have to be atomic in some sense. Since $\mathcal{EL}$ concepts have a tree structure (see [6]), we capture the notion of atomic parts of a concept as its *branches* in its tree structure. We define the notion of branch with the ones of *subdescription* and *width* of a concept. Semantically, being a $\mathcal{T}$ commonality from C to D means being a part of C linked to D: we propose a $\mathcal{T}$ commonality from C to D to be defined as a *characteristic branch* of C w.r.t. $\mathcal{T}$ that subsumes D:

- A characteristic branch of C w.r.t. $\mathcal{T}$ is a primitive branch of C or of a concept equivalent to C (w.r.t. $\mathcal{T}$) such that it cannot be syntactically removed from C without changing its semantics. Fine-grained difference (cf. introduction) is achieved by working at the level of characteristic branches.
- Imposing D being subsumed by a characteristic branch of C expresses the fact that commonalities from C to D are parts of C to which D answers (by being subsumed by them). This is how the inverse subsumption criterion is implemented (cf. introduction).

Then, $C \ominus_{\mathcal{T}} D$ is defined as the minimal concept E that subsumes C such that there are no $\mathcal{T}$ commonalities from E to D (meaning all $\mathcal{T}$ commonalities from C to D have been removed).

We now formalize these notions. First, the width of a concept C is the maximum number of conjuncts composing any conjunction occuring in C.

Definition 2 (width of a concept)**.** *The width of C, noted $\mathsf{wid}(C)$, is defined as follows:*
$\mathsf{wid}(C) = 1$ *if* $C \in \mathbf{C} \cup \{\top\}$
$\mathsf{wid}(C) = Max(n, Max\{\mathsf{wid}(C_i), 1 \leq i \leq n\})$ *if* $C = \bigsqcap_{i=1}^{n} C_i$, $n \geq 2$
$\mathsf{wid}(C) = \mathsf{wid}(D)$ *if* $C = \exists r.D$

A subdescription of a concept C is obtained by removing zero or many conjuncts anywhere in C, provided it remains a syntactically correct concept[3]. The following definition formalizes this idea. It is equivalent to the definition given in [7] (restricted to $\mathcal{EL}$).

Definition 3 (subdescription of a concept)**.** *With $n \geq 2$, the set of subdescriptions of C, noted subd_C, is set to $\{C\}$ if $C \in \mathbf{C} \cup \{\top\}$, or to $\{\exists r.E \mid E \in \mathsf{subd}_D\}$ if $C = \exists r.D$, or to*

$$\bigcup_{\substack{\mathcal{C} \subseteq \{C_i \mid 1 \leq i \leq n\} \\ \text{with } |\mathcal{C}| = m \neq 0}} \left(\bigcup_{\langle S_1, \ldots, S_m \rangle \in \prod_{C_i \in \mathcal{C}} \mathsf{subd}_{C_i}} \left(\bigsqcap_{j=1}^{m} S_j \right) \right) \text{ if } C = \bigsqcap_{i=1}^{n} C_i.$$

We note $\mathsf{subd}^{\mathsf{prim}}_{C,\mathcal{T}} = \mathsf{subd}_C \cap \mathcal{T}^{\mathsf{prim}}_{\mathcal{EL}}$ the set of subdescriptions of C where concept names are primitive concepts or $\top$ only (w.r.t. $\mathcal{T}$).

Example 3. *Let $D = A \sqcap \exists r.(B \sqcap C)$. We have:*
$\mathsf{subd}_D = \{A \sqcap \exists r.(B \sqcap C), \exists r.(B \sqcap C), A \sqcap \exists r.C, A \sqcap \exists r.B, \exists r.B, \exists r.C, A\}$

[3] The notion of subdescription is close to but not the same as the one of *subconcept* defined in [3]. Informally, a subconcept is any conjunction taken in the original concept from which zero or many conjuncts have been removed (keeping at least one).

A branch is essentially a concept having a width of at most 1.

Definition 4 (branch). *Let $\mathcal{T}$ be a TBox and $C \in \mathcal{T}_{\mathcal{EL}}$. The set of branches over $\mathcal{T}$ is noted $\mathsf{br}_{\mathcal{T}}$. The set of primitive branches over $\mathcal{T}$ is noted $\mathsf{br}^{\mathsf{prim}}_{\mathcal{T}}$. The set of branches of C is noted br_C. And the set of primitive branches of C is noted $\mathsf{br}^{\mathsf{prim}}_{C,\mathcal{T}}$. These sets are defined as follows:*
$\mathsf{br}_{\mathcal{T}} = \{S \in \mathcal{T}_{\mathcal{EL}} \mid \mathsf{wid}(S) = 1\}$
$\mathsf{br}^{\mathsf{prim}}_{\mathcal{T}} = \{S \in \mathcal{T}^{\mathsf{prim}}_{\mathcal{EL}} \mid \mathsf{wid}(S) = 1\}$
$\mathsf{br}_C = \{S \in \mathsf{subd}_C \mid \mathsf{wid}(S) = 1\}$
$\mathsf{br}^{\mathsf{prim}}_{C,\mathcal{T}} = \{S \in \mathsf{subd}^{\mathsf{prim}}_{C,\mathcal{T}} \mid \mathsf{wid}(S) = 1\}$

Example 4. *Using TBox $\mathcal{T}$ from example 2, let G be the following concept:*
$G = C \sqcap \exists r_1.(\exists r_2.\exists r_3.\top \sqcap D \sqcap \exists r_4.B)$. *Then we have:*
$\mathsf{br}_G = \{C, \exists r_1.\exists r_2.\exists r_3.\top, \exists r_1.D, \exists r_1.\exists r_4.B\}$ *and* $\mathsf{br}^{\mathsf{prim}}_{G,\mathcal{T}} = \{\exists r_1.\exists r_2.\exists r_3.\top, \exists r_1.\exists r_4.B\}$.

A characteristic branch of C w.r.t. $\mathcal{T}$ is a primitive branch of C or of a concept equivalent to C such that it cannot be syntactically removed from C without changing its semantics.

Definition 5 (characteristic branch). *Let $\mathcal{T}$ be a TBox and $C \in \mathcal{T}_{\mathcal{EL}}$. The set $\mathsf{char}^{\mathcal{T}}_C$ of characteristic branches of C w.r.t. $\mathcal{T}$ is defined as follows:*

$$\mathsf{char}^{\mathcal{T}}_C = \{S \in \mathsf{br}^{\mathsf{prim}}_{\mathcal{T}} \mid \exists C' \in \mathcal{T}_{\mathcal{EL}} \text{ such that } \big(C \equiv_{\mathcal{T}} C' \text{ and } S \in \mathsf{br}_{C'} \text{ and } \forall C'' \in \mathcal{T}_{\mathcal{EL}}\ (\mathsf{br}_{C''} = \mathsf{br}_{C'} \setminus \{S\}) \rightarrow (C'' \not\equiv_{\mathcal{T}} C)\big)\}$$

$\mathcal{T}$ commonalities from C to D are defined as characteristic branches of C that subsume D.

Definition 6 (descriptional commonality). *Let $\mathcal{T}$ be a TBox and $(C, D) \in (\mathcal{T}_{\mathcal{EL}})^2$. The set $\mathsf{dcom}^{\mathcal{T}}_{C,D}$ of $\mathcal{T}$ descriptional commonalities from C to D is defined as follows:*
$\mathsf{dcom}^{\mathcal{T}}_{C,D} = \{S \in \mathsf{char}^{\mathcal{T}}_C \mid D \sqsubseteq_{\mathcal{T}} S\}$

At last, $C \ominus_{\mathcal{T}} D$ is defined as the minimal concept E (w.r.t. $\sqsubseteq_{\mathcal{T}}$) that subsumes C with no $\mathcal{T}$ commonalities from E to D (unicity is shown in proposition 2). When all characteristic branches are commonalities (and thus must all be removed from C), the result is $\top$.

Definition 7 (CSO). *Let $\mathcal{T}$ be a TBox and $(C, D) \in (\mathcal{T}_{\mathcal{EL}})^2$. The binary operator $\ominus_{\mathcal{T}}$, called commonality subtraction operator (CSO) for $\mathcal{T}$, is defined as follows:*

$$C \ominus_{\mathcal{T}} D = \begin{cases} Min_{\sqsubseteq_{\mathcal{T}}}\{E \in \mathcal{T}_{\mathcal{EL}} \mid C \sqsubseteq_{\mathcal{T}} E \text{ and } \mathsf{dcom}^{\mathcal{T}}_{E,D} = \emptyset\} & \text{if } \mathsf{char}^{\mathcal{T}}_C \not\subseteq \mathsf{dcom}^{\mathcal{T}}_{C,D} \\ \top & \text{if } \mathsf{char}^{\mathcal{T}}_C \subseteq \mathsf{dcom}^{\mathcal{T}}_{C,D} \end{cases}$$

Example 5. *Table 2 shows two examples of CSO, with $\mathcal{T}$ from example 2: $Res \ominus_{\mathcal{T}} Query$ and $Query \ominus_{\mathcal{T}} Res$ where $Res = A \sqcap F \sqcap \exists r.\top \sqcap \exists s.\top$ and $Query = B \sqcap \exists r.D \sqcap \exists s.E$. In both cases, the corresponding sets of characteristic branches and descriptional commonalities are given.*

3.2. Computing the CSO using the Tree Subtraction Operator (TSO)

In order to compute $C \ominus_{\mathcal{T}} D$, we propose an approach based on a syntactical difference operator that operates on branches of expansions of C and D. This operator is not the classical set difference of the branch sets since it takes into account subsumption relationships between

Table 2
An example of $Res \ominus_{\mathcal{T}} Query$ and $Query \ominus_{\mathcal{T}} Res$, with $\mathcal{T}$ from example 2.

$\mathcal{T} = \{A \equiv B \sqcap C, C \equiv D \sqcap \exists r.B, D \equiv E, F \equiv \exists r.D\}$	
$Res = A \sqcap F \sqcap \exists r.\top \sqcap \exists s.\top$ $\mathcal{T}^*(Res) = B \sqcap E \sqcap \exists r.B \sqcap \exists r.E \sqcap \exists r.\top \sqcap \exists s.\top$ $\mathsf{char}^{\mathcal{T}}_{Res} = \{B, E, \exists r.B, \exists r.E, \exists s.\top\}$	$Query = B \sqcap \exists r.D \sqcap \exists s.E$ $\mathcal{T}^*(Query) = B \sqcap \exists r.E \sqcap \exists s.E$ $\mathsf{char}^{\mathcal{T}}_{Query} = \{B, \exists r.E, \exists s.E\}$
$\mathsf{dcom}^{\mathcal{T}}_{Res,Query} = \{B, \exists r.E, \exists s.\top\}$ and thus $Res \ominus_{\mathcal{T}} Query = E \sqcap \exists r.B$ $\mathsf{dcom}^{\mathcal{T}}_{Query,Res} = \{B, \exists r.E\}$ and thus $Query \ominus_{\mathcal{T}} Res = \exists s.E$	

branches, e.g. those involving the concept $\top$. Moreover it does not change the original tree structure of the minuend. We call this operator the tree subtraction operator (TSO), noted Δ. Informally, $C \Delta D$, to be read "C deprived of D", is intended to be the minimal concept w.r.t. $\sqsubseteq$ that subsumes C such that all branches in br_C that subsume a branch in br_D have been removed from br_C. That means we remove from br_C branches that are also in br_D, but also branches of br_C that end by $\top$ when there is in br_D a branch beginning with the same existential restrictions. If all branches of C are removed, then the result is set to be $\top$.

Definition 8 (TSO). *Let C and D be two concepts. The binary operator Δ, called tree subtraction operator (TSO), is defined as follows:*

$$C \Delta D = \begin{cases} Min_{\sqsubseteq}\{E \mid C \sqsubseteq E \text{ and } \mathsf{br}_E = \mathsf{br}\} & \text{if } \mathsf{br} \neq \emptyset \\ \top & \text{if } \mathsf{br} = \emptyset \end{cases}$$

with $\mathsf{br} = \mathsf{br}_C \setminus \big(\mathsf{br}_D \cup \{S = \exists r_1.\exists r_2 ... \exists r_n.\top \in \mathsf{br}_C, n \geq 0 \mid$
$\exists S' = \exists r_1 ... \exists r_n.\exists r_{n+1} ... \exists r_{n+m}.P \in \mathsf{br}_D, m \geq 0\}\big)$
and P any concept name or $\top$ (with the exception that P cannot be $\top$ when $m = 0$).

Example 6. *Following table 2, with $R = \mathcal{T}^*(Res)$ and $Q = \mathcal{T}^*(Query)$, we have:*
$\mathsf{br}_R \setminus \big(\mathsf{br}_Q \cup \{S = \exists r_1.\exists r_2 ... \exists r_n.\top \in \mathsf{br}_R, n \geq 0 \mid$
$\exists S' = \exists r_1 ... \exists r_n.\exists r_{n+1} ... \exists r_{n+m}.P \in \mathsf{br}_Q, m \geq 0\}\big) = \{E, \exists r.B\}$.
Thus $Res \ominus_{\mathcal{T}} Query = R \Delta Q = E \sqcap \exists r.B$.

It is not difficult to show that definition 8 ensures $C \Delta D$ keeps the tree structure of C, i.e. is a subdescription of C. We can illustrate this with non equivalent concepts having the same set of branches, like $C_1 = \exists r.A \sqcap \exists r.B$ and $C_2 = \exists r.(A \sqcap B)$ which set of branches is $\mathsf{br}_{C_1} = \mathsf{br}_{C_2} = \{\exists r.A, \exists r.B\}$. Suppose we remove from C_1 (resp. C_2) a concept D that has no commonality with C_1 (resp. C_2), then the result is C_1 and not C_2 (resp. C_2 and not C_1), thus keeping the initial tree structure.

TSO is implemented by algorithm 1. Its principle is to traverse at the same time the tree structures of both C and D, removing from C branches that subsume, w.r.t. the empty TBox, a branch of D. Properties of the TSO and algorithm 1 (unicity, termination, soundness, complexity) are given in proposition 1.

Proposition 1 (Properties of TSO and algorithm 1). *Let C and D be two concepts. We have:*

a. $C \Delta D$ always exists and is unique.

b. Algorithm 1 terminates and produces a unique result.

Algorithm 1 tso(C, D)

Require: C and D two $\mathcal{EL}$ concepts.
Ensure: $C \Delta D$ (cf. def. 8)

if $C = D$ or $C = \top$ **then**
  $Result := \top$
else
  if $C = C_1 \sqcap \ldots \sqcap C_n$ with $n \geq 2$ **then**
    $Result1 :=$ tso$(C_1, D) \sqcap \ldots \sqcap$ tso(C_n, D)
    if There is at least one conjunct $\neq \top$ in $Result1$ **then**
      $Result := Result1$ without any $\top$ conjunct.
    else
      $Result := \top$
    end if
  else if $D = D_1 \sqcap \ldots \sqcap D_m$ with $m \geq 2$ **then**
    $Result :=$ tso$(\ldots($tso$($tso$(C, D_1), D_2), \ldots), D_m)$
  else if $C = \exists r.C'$ and $D = \exists r.D'$ **then**
    $Result1 :=$ tso(C', D')
    if $Result1 = \top$ **then**
      $Result := \top$
    else
      $Result := \exists r.Result1$
    end if
  else
    $Result := C$
  end if
end if
return $Result$

Algorithm 2 cso$(\mathcal{T}, C, D)$

Require:

- $\mathcal{T}$ an acyclic and definitional $\mathcal{EL}$ TBox
- $(C, D) \in (\mathcal{T}_{\mathcal{EL}})^2$

Ensure: $C \ominus_{\mathcal{T}} D$ (cf. def. 7)

return tso$(\mathcal{T}^*(C), \mathcal{T}^*(D))$

c. $C \Delta D =$ tso(C, D) (soundness).
d. Computing tso(C, D) is in PTIME in the sizes of C and D.

Sketch of proof. a. Existence comes from the definition of br which always corresponds to at least one concept, or $\top$ when it is empty. Unicity trivially comes from the minimality w.r.t. $\sqsubseteq$.

b. We show by induction on the sizes of C and D that tso(C, D) always terminates and generates an output which size is strictly less than size(C) + size(D).

c. Soundness is showed in 3 steps: (i) from the characterization of subumption in $\mathcal{EL}$ without

Table 3
Comparison of existing difference operators on an example where $Res_1 \equiv A \sqcap B \sqcap \exists r.(C \sqcap D)$ and $Res_2 \equiv A \sqcap B \sqcap \exists r.C$.

Oper.	$Res_1 - Res_2$	$Res_2 - Res_1$
$\ominus_{Te}$	$\exists r.(C \sqcap D)$	undefined
$\ominus_{Br}$	$\exists r.(C \sqcap D)$	$\top$
$\ominus_{Su}$	$\top$	$\exists r.C$
$\ominus_{He}$	$B \sqcap \exists r.(C \sqcap D)$	Res_2
$\ominus_{Ri}$	Either $\exists r.D$ (full meet case) or one of $\{B \sqcap \exists r.(C \sqcap D), A \sqcap \exists r.(C \sqcap D), A \sqcap B \sqcap \exists r.D\}$ (maxi-choice case)	Res_2 (full meet and maxi-choice cases)
$\ominus_{\mathcal{T}}$	$\exists r.D$	$\top$

any TBox given in [6], we derive a characterization of subsumption in $\mathcal{EL}$ in terms of subdescriptions , (ii) then is derived a characterization of TSO in terms of subdescriptions, and (iii) at last a proof by induction of soundness is given, using the characterization obtained at step (ii).

d. Tractability is showed by finding a worst case (when C and D are conjunctions of concepts names, without any existential restriction) and studying the complexity in this case. □

Now, we can use TSO to compute CSO: $C \ominus_{\mathcal{T}} D$ is obtained by computing $\mathcal{T}^*(C) \Delta \mathcal{T}^*(D)$, cf. algorithm 2. Proposition 2 shows properties associated to CSO, namely existence, uniqueness, and soundness and complexity of algorithm 2 (termination is trivially implied by terminations of the complete expansion and algorithm 1).

Proposition 2. *Let $\mathcal{T}$ be a TBox and $(C, D) \in (\mathcal{T}_{\mathcal{EL}})^2$. There is:*

a. $C \ominus_{\mathcal{T}} D$ exists and is unique (up to $\equiv_{\mathcal{T}}$).

b. $C \ominus_{\mathcal{T}} D = \mathcal{T}^*(C) \Delta \mathcal{T}^*(D) = cso(\mathcal{T}, C, D)$ (soundness).

c. Computing $cso(\mathcal{T}, C, D)$ is in EXPTIME in the sizes of $\mathcal{T}$, C and D and in PTIME in the sizes of $\mathcal{T}^*(C)$ and $\mathcal{T}^*(D)$.

Sketch of proof. a. Existence and unicity of CSO easily come from definition 7.

b. Soundness of cso$(\mathcal{T}, C, D)$ is grounded on (i) the characterization of subsumption in terms of subdescriptions already used in proof of proposition 1, and on (ii) a lemma stating $\text{br}_{C\Delta D}$ is the set of branches of C that do not subsume D.

c. The result easily comes from complexity of the complete expansion and algorithm 1. □

4. Related works

As far as we know, five difference operators have been defined for DLs before CSO, in [8, 7, 9, 10, 11]. In the sequel, we respectively note them $\ominus_{Te}$, $\ominus_{Br}$, $\ominus_{Su}$, $\ominus_{He}$ and $\ominus_{Ri}$ (and $\ominus_{\mathcal{T}}$ for CSO). We do not consider the contraction operator defined in [12], since it appears to be more a matchmaking operator rather than a difference one.

We first begin by illustrating how these difference operators behave. Let's take the following two descriptions: $Res_1 \equiv A \sqcap B \sqcap \exists r.(C \sqcap D)$ and $Res_2 \equiv A \sqcap B \sqcap \exists r.C$. In table 3, we give

Table 4
Comparison of difference operators in DLs.

Reference and informal principle	①	②	③	④	Remarks and complexity
[8] $C \ominus_{Te} D$ finds the maximal concept w.r.t. $\sqsubseteq$ that added to D by conjunction gives a concept equivalent to C.	C	M	M	N	• Defined for any DL $\mathcal{L}$. • Not defined if $C \not\sqsubseteq D$. • Unstudied with a TBox. • Complexity is not given.
[7] $C \ominus_{Br} D$ finds the minimal concepts w.r.t. the subdescription order such that, added to D by conjunction, they give a concept equivalent to $C \sqcap D$.	C	M	T	N	• C is in $\mathcal{ALC}$, D is in $\mathcal{ALE}$. • Unstudied with a TBox. • PTIME in sizes of C and D given an oracle for subsumption.
[9] $C \ominus_{Su} D$ removes the minimal conjuncts of C, w.r.t. a syntactical total order on $\mathcal{EL}$ concepts, that are subsumed by conjuncts of D (one removed conjunct for one conjunct of D).	S	B	–	N	• Defined for $\mathcal{EL}$. • $\mathcal{T}$ is acyclic and definitional • EXPTIME in the sizes of $\mathcal{T}$, C and D and PTIME in the sizes of $\mathcal{T}^*(C)$ and $\mathcal{T}^*(D)$.
[10] $C \ominus_{He} D$ removes conjuncts of C that are subsumed by the smallest conjunct of D, w.r.t. a syntactical total order. Recursive process inside existential restrictions.	S	B	–	Y	• Defined for $\mathcal{EL}$. • $\mathcal{T}$ is acyclic and definitional. • $C \ominus_{He} D = C$ if $C \not\sqsubseteq D$. • Complexity is not given.
[11] Extending the notion of subdescription to contain concepts obtained after replacing a concept name by $\top$, $C \ominus_{Ri} D$ finds the single subdescription (full meet mode) or the many subdescriptions (maxichoice mode) S of C that are minimal w.r.t. $\sqsubseteq$ such that $S \not\sqsubseteq D$.	S	M	–	Y	• Defined for $\mathcal{EL}$. • $\mathcal{T}$ is acyclic and definitional. • $C \ominus_{Ri} D = C$ if $C \not\sqsubseteq D$. • Complexity is not given.
[This paper] $C \ominus_{\mathcal{T}} D$ removes $\mathcal{T}$ descriptional commonalities from C to D, i.e. the characteristic branches of C that subsume D.	S	B	M	Y	• Defined for $\mathcal{EL}$. • $\mathcal{T}$ is acyclic and definitional. • EXPTIME in the sizes of $\mathcal{T}$, C and D and PTIME in the sizes of $\mathcal{T}^*(C)$ and $\mathcal{T}^*(D)$.

the result of $Res_1 \ominus Res_2$ and $Res_2 \ominus Res_1$, for $\ominus$ replaced by each operator. Note that we suppose to have an empty TBox $\mathcal{T}$, for a sake of simplicity. We can see that each of the six operators give different results when considering both $Res_1 \ominus Res_2$ and $Res_2 \ominus Res_1$.

Second, we propose to classify these operators in table 4 according to 4 dimensions (precise definitions of operators are recalled in appendix [4]. ① is their type: S for subtraction-based (something is removed from the minuend) or C for completion-based (something is added to the subtrahend). ② is the definition type of the difference: M for semantical or B for both semantical and syntactical. ③ is the optimization (min or max) criterion type to choose the best result: T for syntactical, M for semantical, or – if non relevant. ④ is the fine grained

[4] https://github.com/Myakko/DL2023-Appendix

difference property i.e. the ability to remove precise subdescriptions of the minuend inside nested existential restrictions without removing unnecessary ones: Y for yes and N for no.

Tables 3 and 4 show that CSO is original w.r.t. existing operators. More precisely:

- $\ominus_{Te}$ and $\ominus_{Br}$ are completion-based operators, they do not achieve fine-grained difference, and $\ominus_{Br}$ does not ensure unicity.
- $\ominus_{Su}$ is a subtraction operation, however it is not a fine-grained difference. Moreover the notion of commonality is based on subsumption and not inverse subsumption.
- $\ominus_{He}$ and $\ominus_{Ri}$ are a fine-grained subtraction operators, as is CSO. However, the differences with CSO are the following ones: (i) for both operators, there can be subtraction only when the minuend is subsumed by the subtrahend, which is a restriction CSO does not have; (ii) in $\ominus_{He}$, the notion of common part is not based on inverse subsumption; and (iii) $\ominus_{Ri}$ is sometimes too much fine-grained, which may lead it not to remove existential restrictions in cases where CSO does, e.g. $\exists r.A \ominus_{Ri} \exists r.A = \exists r.\top$, while $\exists r.A \ominus_{\mathcal{T}} \exists r.A = \top$.

5. Conclusion

We propose a difference operator for $\mathcal{EL}$ w.r.t. an acyclic and definitional TBox, named CSO. It is based on the TSO, a operator to achieve a syntactical tree difference between two concepts. We propose a tractable algorithm to compute TSO and thus CSO (in the sizes of the complete expansion of the inputs w.r.t. the TBox), and show that CSO is an original difference operator w.r.t existing ones. Also, an implementation of these operators has been done in the Ruby programming language for integration into the metrology platform. Performance tests are being achieved.

Even if CSO is intended to be used in a matchmaking process, we plan to study how it instanciates properties that are defined for difference operators in the AGM approach of agent belief revision [13], namely preservation, success, inclusion, vacuity, recovery, failure, fullness and relevance. This would provide a more precise insight on CSO w.r.t. existing operators and help decide its potential interest in the AGM framework. Besides, we would like to extend this work to the case where TBoxes can be general and cyclic, for which we do not know any DL difference operator yet.

Acknowledgements

This work has been fully supported by the European Regional Development Fund[5].

References

[1] A. Mascaro, C. Rey, Recommandation ontologique multicritère pour la métrologie, in: Rencontres des Jeunes Chercheur·ses en Intelligence Artificielle, in PFIA 2020, Angers, 2020.

[5] https://ec.europa.eu/regional_policy/en/funding/erdf/.

[2] D. Calvanese, G. D. Giacomo, J. Carroll, I. Herman, B. Parsia, P. F. Patel-Schneider, A. Ruttenberg, U. Sattler, M. Schneider, Owl 2 web ontology language profiles (second edition), https://www.w3.org/TR/owl2-profiles/, 2012. Accessed: 2023-06-14.

[3] F. Baader, I. Horrocks, C. Lutz, U. Sattler, An Introduction to Description Logic, Cambridge University Press, 2017. URL: http://www.cambridge.org/de/academic/subjects/computer-science/knowledge-management-databases-and-data-mining/introduction-description-logic?format=PB#17zVGeWD2TZUeu6s.97.

[4] B. Nebel, Terminological reasoning is inherently intractable, Artificial Intelligence 43 (1990) 235–249.

[5] The Description Logic Handbook: Theory, Implementation and Applications, 2 ed., Cambridge University Press, 2007. doi:10.1017/CBO9780511711787.

[6] F. Baader, R. Küsters, R. Molitor, Computing least common subsumers in description logics with existential restrictions, in: T. Dean (Ed.), Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 99, Stockholm, Sweden, July 31 - August 6, 1999. 2 Volumes, 1450 pages, Morgan Kaufmann, 1999, pp. 96–103. URL: http://ijcai.org/Proceedings/99-1/Papers/015.pdf.

[7] S. Brandt, R. Küsters, A.-Y. Turhan, Approximation and difference in description logics, in: Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR'2002), Toulouse, France, D. Fensel, F. Giunchiglia et al., Eds. San Francisco, California, USA: Morgan Kaufmann Publishers, 2002.

[8] G. Teege, Making the difference: A subtraction operation for description logics, in: J. Doyle, E. Sandewall, P. Torasso (Eds.), Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning (KR'94). Bonn, Germany, May 24-27, 1994, Morgan Kaufmann, 1994, pp. 540–550.

[9] F. M. Suchanek, C. Menard, B. M., C. C., Can you imagine... a language for combinatorial creativity?, in: First Part of the Proceededings of the 15th International Semantic Web Conference (ISWC'2016), Kobe, Japan, volume 9981 of *Lecture Notes in Computer Science, P. T. Groth, E. Simperlet al., Eds.*, Cham, Switzerland:Springer International Publishing, 2016, pp. 532 – 548.

[10] H. Heinz, How I Lost My OWL: Retracting Knowledge from EL Concepts, Master's thesis, Koblenz-Landau University, 2018.

[11] T. Rienstra, C. Schon, S. Staab, Concept contraction in the description logic el, in: International Conference on Principles of Knowledge Representation and Reasoning (KR-20), 2020, pp. 723–732.

[12] S. Colucci, T. Di Noia, E. Di Sciascio, F. Donini, M. Mongiello, Concept abduction and contraction in description logics, in: Proceedings of the 16th International Workshop on Description Logics (DL'03), volume 81 of *CEUR Workshop Proceedings*, 2003.

[13] C. E. Alchourrón, P. Gärdenfors, D. Makinson, On the logic of theory change: Partial meet contraction and revision functions, The Journal of Symbolic Logic 50 (1985) 510–530.