Partial and Informative Common Subsumers of Concepts Collections in Description Logics

Simona Colucci[1 3], Eugenio Di Sciascio[1], and Francesco Maria Donini[2]

[1] SisInfLab–Politecnico di Bari, Bari, Italy
[2] Università della Tuscia, Viterbo, Italy
[3] D.O.O.M. s.r.l., Matera, Italy

Abstract. Least Common Subsumers in Description Logics have shown their usefulness for discovering commonalities among all concepts of a collection. Several applications are nevertheless focused on searching for properties shared by significant portions of a collection rather than by the collection as a whole. Actually, this is an issue we faced in a real case scenario that provided initial motivation for this study, namely the process of Core Competence extraction in knowledge intensive companies. The paper defines four reasoning services for the identification of meaningful common subsumers describing partial commonalities in a collection. In particular Common Subsumers adding informative content to the Least Common Subsumer are investigated, with reference to $\mathcal{ALN}$ and $\mathcal{ELN}$.

1 Introduction

Least Common Subsumers(LCSs) were originally proposed by Cohen, Borgida and Hirsh [5] as novel reasoning service for the Description Logic underlying Classic [4]. By definition, for a collection of concept descriptions, their LCS represents the most specific concept description subsuming all of the elements of the collection.

The usefulness of the inference task has been shown in several applications, varying from learning from examples ([6],[7] , [9]), to similarity-based Information Retrieval ([12], [13]) and bottom-up construction of knowledge bases ([1], [2]).

Nevertheless, there are some problems in which the computation of LCS does not provide solutions. The LCS in fact intuitively represents properties shared by *all* the elements of a given collection. In several applications, instead, such a sharing is not required to be full: in other words we could be interested in finding a concept description subsuming a portion of the elements in the collection.

Different perspectives on the introduced problem may be taken: if the LCS of the collection is the universal concept, we can determine the concept description subsuming a number m of concept descriptions in the collection, where m is the maximum cardinality of subsets of the collection for which a common subsumer non-equivalent to the universal concept exists. We give the name *Best Common Subsumer* to such a concept description, in analogy with LCS.

Alternatively, we could be interested in determining a concept description subsuming at least k elements in the collection, where k is a threshold value established a priori

on the basis of a decisional process dependent on the application domain. We give such a different concept description the name k-*Common Subsumer* (k-CS).

In particular, the search should revert on those k-CSs adding informative content to LCS: we call *Informative k-Common Subsumer* (IkCS) a k-CS more specific than the LCS of the collection.

We here introduce the k-*CS*, the *IkCS*, the *BCS* and one more specific service (*Best Informative Common Subsumer*), starting by outlining the application problem motivating our research: Core Competence individuation in knowledge-intensive companies. In particular we give a definition of introduced services and some useful results known by LCS theory in Section 3. In Section 4 we propose some computation results for introduced services in different DLs. Final remarks close the paper.

2 Motivation

The proposal of the k-Common Subsumer and all its specifications as reasoning problems in DL originates from a universally shared need in knowledge intensive companies: the recognition of most characterizing knowledge, well known in knowledge management literature by the name *Core Competence*. The term *Core Competence* was introduced by Hamel and Prahalad [10] to identify the knowledge specializing a company, the skills the management should invest on to achieve competitive advantage.

In large companies the process of identifying Core Competence becomes a very complex task, due to difficulties in getting in touch with and analyzing available sources of knowledge. Just as an example, let us think of a multinational consulting company, involved in the decisional process of choosing the business sectors — and the related intellectual capital — to invest on in its long term strategy. By manually performing such a selection process the management may neglect several competencies that, although characterizing many company employees, are difficult to discover because of different reasons (for example, competencies never used in past projects, or competencies not explicitly asserted in personnel profiles).

The depicted scenario suggests that in large, knowledge intensive, companies the process of Core Competence extraction calls for knowledge-based systems exploiting the semantics of available information. For the implementation of services for automatic Core Competence extraction, we model company knowledge through DLs; in particular we assume hereafter for the sake of simplicity that the only source of knowledge lies in company personnel and we therefore focus on employees profile descriptions. The information stored in profiles can be aggregated in order to extract fields of excellence of the company.

We assume that the possession of Core Competence, as it is reasonable, is not necessarily required by the whole company personnel: the management can set the level of coverage necessary to consider a particular skill as part of the Core Competence.

In order to understand the rationale of the proposed assumptions, consider the following simple example. In a small company only four people are employed:

- *Allison*, a Process Planner and Mathematician, endowed with knowledge in Information Systems and Quality Assurance Techniques
- *Eva*, an Asset Manager with knowledge about Psychology

– *Walter*, an Engineer expert in VBScript, Java and C since 3 years and with knowledge about UML, Computer Graphics, Client Server Protocol and Information Systems
– *Michael*, a Managerial Engineer with advanced knowledge of Internet Technology, Operation Optimization, Production Management and Distribution Management and expert in Asset Allocation since 3 years.

In order to model this small knowledge domain we provide the TBox excerpt in Figure 1. The four concept descriptions representing employees profiles are shown in Table 1.

$$
\begin{aligned}
Skill \sqsupseteq \Bigg\{ & Psychology \\
& ProcessPlanning \\
& Mathematics \\
& ProcPerfMonitoring \\
& QualityAssTechs \\
& Internet \\
& OperationsOptimization \\
& DistributionManagement \\
& AssetAllocation \\
& Design \\
& Management \\
& SupplyChain \\
& ProductionManagement \\
& ComputerGraphics \\
& ClientServerProtocol \\
& UML \\
& Programming \sqsupseteq \Bigg\{ \begin{array}{l} Script \sqsupseteq \left\{ \begin{array}{l} Javascript \\ VBscript \end{array} \right. \\ OOP \sqsupseteq \left\{ \begin{array}{l} Java \\ C++ \end{array} \right. \\ Structural \sqsupseteq C \end{array} \\
& InformationSystems \sqsupseteq \left\{ \begin{array}{l} DBMS \\ TCP/IP \\ ERPsystem \end{array} \right.
\end{aligned}
\Bigg.
$$

$$
\begin{aligned}
& Mathematician \equiv \exists hasKnowledge.(Mathematics \sqcap (\geq\ 5\ hasExperienceYears)) \\
& ProcessPlanner \equiv \exists hasKnowledge.(ProcessPlanning \sqcap (\geq\ 5\ hasExperienceYears)) \sqcap \\
& \quad \exists hasKnowledge.ProcPerfMonitoring \\
& Manager \equiv \exists hasKnowledge.(Management \sqcap (\geq\ 7\ hasExperienceYears)) \\
& AssetManager \equiv Manager \sqcap \exists hasKnowledge.(AssetAllocation \sqcap (\geq\ 5\ hasExperienceYears)) \\
& Engineer \equiv \exists hasKnowledge.(Design \sqcap (\geq\ 4\ hasExperienceYears)) \\
& ManagerialEng \equiv Engineer \sqcap \exists hasKnowledge.SupplyChain \sqcap \\
& \quad \exists hasKnowledge.(ProductionManagement \sqcap (\geq\ 5\ hasExperienceYears))
\end{aligned}
$$

Fig. 1. An Excerpt of TBox $\mathcal{T}$ Concept Inclusions and Concept Definitions

We note that the DL sufficient to express such a TBox is $\mathcal{ELN}$. However, the general problem of Skill Management may require other constructs, as well [8].

It is easy to see that the LCS of the collection of employees profiles is the universal concept: if we defined Core Competence only the competence held by all of the employees, we may simply assert that our small company does not have any Core Competence. If we instead give up such a full skill coverage and assume that a competence owned by at least half of the employees might be a Core Competence, we may list "Information Systems", "Engineer" job title and three years of experienced knowledge in "Asset Al-

Employee Name	Concept Description Representing Employee Profile
Allison	$ProcessPlanner \sqcap Mathematician \sqcap$ $\exists hasKnowledge.InformationSystems \sqcap$ $\exists hasKnowledge.QualityAssTechs$
Eva	$AssetManager \sqcap \exists hasKnowledge.Psychology$
Walter	$Engineer \sqcap$ $\exists hasKnowledge.(VBscript \sqcap (\geq\ 3\ hasExperienceYears)) \sqcap$ $\exists hasKnowledge.(Java \sqcap (\geq\ 3\ hasExperienceYears)) \sqcap$ $\exists hasKnowledge.(C \sqcap (\geq\ 3\ hasExperienceYears)) \sqcap$ $\exists hasKnowledge.UML \sqcap \exists hasKnowledge.ComputerGraphics \sqcap$ $\exists hasKnowledge.ClientServerProtocol \sqcap$ $\exists hasKnowledge.InformationSystems$
Michael	$ManagerialEng \sqcap \exists hasKnowledge.Internet \sqcap$ $\exists hasKnowledge.OperationsOptimization \sqcap$ $\exists hasKnowledge.DistributionManagement \sqcap$ $\exists hasKnowledge.(AssetAllocation \sqcap (\geq\ 3\ hasExperienceYears))$

Table 1. The Concept Descriptions representing Employee Profiles

location" in it. Such a result is obviously much more significant than the first one w.r.t. the objective of determining skills to invest on.

3 Definitions and Known Results

Definition 1 (LCS, [7]). *Let $C_1, \ldots, C_n$ be n concept descriptions in a DL $\mathcal{L}$. An LCS of $C_1, \ldots, C_n$, denoted by $LCS(C_1, \ldots, C_n)$, is a concept description E in $\mathcal{L}$ such that the following conditions hold:*

(i) $C_i \sqsubseteq E$ *for* $i = 1, \ldots, n$
(ii) E *is the least $\mathcal{L}$-concept description satisfying (i),i.e., , if E' is an $\mathcal{L}$-concept description satisfying $C_i \sqsubseteq E'$ for all $i = 1, \ldots, n$, then $E \sqsubseteq E'$*

It is well known that, if the DL $\mathcal{L}$ admits conjunction of concepts "$\sqcap$", then the LCS is unique up to concept equivalence (since if both E_1 and E_2 are common subsumers of $C_1, \ldots, C_n$, then so is $E_1 \sqcap E_2$). Moreover, if union of concepts "$\sqcup$" is allowed in $\mathcal{L}$, then for every set of concepts $C_1, \ldots, C_n \in \mathcal{L}$, their LCS is $C_1 \sqcup \cdots \sqcup C_n$. Hence, the study of LCS is limited to DLs not admitting union.

Definition 2 (k-CS). *Let $C_1, \ldots, C_n$ be n concepts in a DL $\mathcal{L}$, and let be $k < n$. A* k-Common Subsumer *(k-CS) of $C_1, \ldots, C_n$ is a concept D such that D is an LCS of k concepts among $C_1, \ldots, C_n$.*

Definition 3 (IkCS). *Let $C_1, \ldots, C_n$ be n concepts in a DL $\mathcal{L}$, and let $k < n$. An* Informative k-Common Subsumer *(IkCS) of $C_1, \ldots, C_n$ is a k-CS E such that E is strictly subsumed by $LCS(C_1, \ldots, C_n)$.*

Definition 4 (BICS). *Let $C_1, \ldots, C_n$ be n concepts in a DL $\mathcal{L}$. A* Best Informative Common Subsumer *(BICS) of $C_1, \ldots, C_n$ is a concept B such that B is an Informative k-CS for $C_1, \ldots, C_n$, and for every $k < j \leq n$ every j-CS is not informative.*

For collections whose LCS is equivalent to the universal concept the following definition makes also sense:

Definition 5 (BCS). *Let $C_1, \ldots, C_n$ be n concepts in a DL $\mathcal{L}$. A* Best Common Subsumer *(BCS) of $C_1, \ldots, C_n$ is a concept S such that S is a k-CS for $C_1, \ldots, C_n$, and for every $k < j \leq n$ every j-CS $\equiv \top$.*

Proposition 1. *If $LCS(C_1, \ldots, C_n) \equiv \top$, every BCS is also a BICS.*

Even though the services defined above may appear quite similar to each other at a first sight, it has to be underlined that they deal with different problems:

- k-CS: can be computed for every collection of elements and finds least common subsumers of k elements among the n belonging to the collection;
- IkCS: describes those k-CSs adding an informative content to the one provided by LCS, *i.e.,* more specific than LCS. Observe that IkCS does not exist when every subset of k concepts has the same LCS as the one of all $C_1, \ldots, C_n$;
- BICS: describes IkCSs subsuming h concepts, such that h is the maximum cardinality of subsets of the collection for which an IkCS exists. A BICS does not exist if and only if $C_i \equiv C_j$ for all $i, j = 1, \ldots, n$;
- BCS: may be computed only for collections admitting only LCS equivalent to the universal concept; it finds k-CSs such that k is the maximum cardinality of subsets of the collection for which an LCS not equivalent to $\top$ exists.

By reverting to the example introduced in Section 2 it is easy to see that the concepts describing knowledge of "Information Systems", three years of experience in "Asset Allocation" and "Engineer" job title represent a k-CS of the analyzed set of personnel profiles if k equals 2. The three concepts can moreover be considered as Informative k-CS because they add information to the LCS. We can also recognize in the three concepts Best (and Best Informative) Common Subsumers: by adding any other personnel profile in the set of subsumed concepts, their common subsumer reverts to the universal concept and stops being Informative w.r.t. the LCS.

4 Computation

In the computation of partial and informative common subsumers of a collection of concept descriptions $C_1, \ldots, C_n$ we assume that all concepts C_i in the collection are consistent; hence $C_i \not\equiv \bot$ for every $C_i \in (C_1, \ldots, C_n)$.

The reasoning services introduced in Section 3 ask for the concepts of the input collection to be written in components according to the following recursive definition:

Definition 6 (Concept Components). *Let C be a concept description in a DL $\mathcal{L}$, with C written in a conjunction $C^1 \sqcap \cdots \sqcap C^m$. The* Concept Components *of C are defined as follows:*

1. *if C^j, with $j = 1 \ldots, m$ is either a concept name, or a negated concept name, or a number restriction, then C^j is a* Concept Component *of C*
2. *if $C^j = \exists R.D$, with $j = 1 \ldots, m$, then $\exists R.\top$ is a* Concept Component *of C*

3. *if* $C^j = \forall R.E$*, with* $j = 1 \ldots, m$ *, then* $\forall R.E^k$ *is a* Concept Component *of* C*, for each* E^k Concept Component *of* E

Observe that we do not propagate universal restriction over existential restriction since existential restriction always simplify to a component of the form $\exists R.\top$.

For the computation of the sets of k-CSs, IkCSs, BICSs and BCSs of a collection of concepts we define in the following a *Subsumers Matrix*, for the representation of the collection itself.

Definition 7 (Subsumers Matrix). *Let* $C_1, \ldots, C_n$ *be a collection of concept descriptions* C_i *in a Description Logic* $\mathcal{L}$ *and let* $D_j \in \{D_1, \ldots, D_m\}$ *be the* Concept Components *deriving from all concepts in the collection. We define the* ***Subsumers Matrix*** $S = (s_{ij})$ *, with* $i = 1 \ldots n$ *and* $j = 1 \ldots m$*, such that* $s_{ij} = 1$ *if the component* D_j *subsumes* C_i*, and* $s_{ij} = 0$ *if the component* D_j *does not subsume* C_i*.*

Definition 8. *Referring to the Subsumers Matrix of* $C_1, \ldots, C_n$*, we define:*

Concept Component Signature (sig_{D_j}) *: set of indices of concepts* $C_1, \ldots, C_n$ *subsumed by* D_j*; observe that* $sig_{D_j} \subseteq \{1, \ldots, n\}$*;*

Concept Component Cardinality (T_{D_j}) *: cardinality of* sig_{D_j}*, that is, how many concepts among* $C_1, \ldots, C_n$ *are subsumed by* D_j*. Such a number is* $\sum_{i=1}^{n} s_{ij}$*;*

Maximum Concept Component Cardinality (M_S) *: maximum among all concept component cardinalities, that is,* $M_S = max\{T_{D_1}, \ldots T_{D_m}\}$*;*

Partial Maximum Concept Component Cardinality (PM_S) *: maximum among the cardinalities of concept components not subsuming all* n *concepts in the collection (*$PM_S = max\{T_{D_j} | T_{D_j} < n\}$*); by definition* $PM_S < n$*;*

Common Signature Class($\bigcap_{sig_{D_j}}$) *: concept formed by the conjunction of all concept components whose signature contains* D_j*:* $\sqcap\{D_h \mid sig_{D_j} \subseteq sig_{D_h}\}$

Definition 7 hints that the determination of Subsumers Matrix includes an oracle to subsumption. As a consequence the following proposition holds:

Proposition 2. *Let* $\mathcal{L}$ *be a DL whose subsumption problem is decidable in polynomial time. Then Subsumers Matrix determination in* $\mathcal{L}$ *is computable in polynomial time too.*

Such a result causes the computation of common subsumers in DLs with different complexities for subsumption to be treated separately in next subsections. Nevertheless some considerations are logic independent and preliminary to the determination of common subsumers in every DL.

Firstly, we define the solution sets for the introduced reasoning services, independently on the DL employed for the representation of concepts in a given collection:

B : set of BCSs of the collection
BI : set of BICSs of the collection
I_k : set of IkCSs of the collection, given $k < n$
L_k : set of k-CSs of the collection, given $k < n$

Proposition 3. *Given a DL* $\mathcal{L}$ *and a collection of concept descriptions in* $\mathcal{L}$*, for each* $k < n$ *the solution sets of the collection are such that* $I_k \subseteq L_k$*. If the collection admits only the universal concept as LCS, then* $B = BI$ *also holds.*

Proposition 4. *The following observations on the Subsumers matrix represent necessary conditions to determine common subsumers of the collection:*

1. *Only concept components D_j for which $T_{D_j} \geq k$ are meaningful for the determination of L_k elements*
2. *Only concept components subsuming k concepts but not all n concepts in the collection are meaningful for the determination of I_k elements: D_j for which $k \leq T_{D_j} < n$*
3. *Only concept components D_j for which $T_{D_j} = PM_S$ are taken into account for the determination of BI elements*
4. *Only concept components D_j for which $T_{D_j} = M_S$, with $M_S < n$ may determine B elements.*

The representation (for $k = 2$) in Figure 2 shows the introduced necessary conditions and should clarify the differences among introduced services. In particular, notice that in case of a collection admitting LCS $\neq \top$ (left-hand side of Figure 2), the best common subsumer of the collection is the LCS itself, so only the computation of BICSs makes sense. On the contrary, for a collection admitting only LCS $\equiv \top$ (right-hand side of Figure 2), BCSs may be computed and are BICSs too.

	D_1	D_2	$D\ldots$	D_m
C_1	x	x		x
C_2		x		x
$C\ldots$				x
C_n	x	x		x
	IkCS, k-CS	*k-CS, IkCS, BICS*		*LCS, k-CS*

	D_1	D_2	$D\ldots$	D_m
C_1	x	x		x
C_2		x		x
$C\ldots$				
C_n	x			x
	k-CS, IkCS	*k-CS, IkCS*		*k-CS, IkCS, BICS, BCS*

Fig. 2. Subsumers Matrices of a collection whose LCS $\neq \top$ (left) and of one whose LCS $\equiv \top$ (right)

It has to be anyway underlined that the criteria exposed before are necessary to determine common subsumers, but are not sufficient to find the most specific ones, as required by Definition 2.

Given that associative property holds for LCS, one could think to compute k-CSs by exploiting such a property and taking, for example, a simple dynamic programming approach. In the following we provide a theorem dealing with the number of possible kCSs showing that such an approach is instead inappropriate.

Theorem 1. *For some sets of n concepts $C_1, \ldots, C_n$ in a DL $\mathcal{L}$, and for some $k < n$, there are exponentially many kCS of $C_1, \ldots, C_n$.*

Proof. Define the set of concepts $\mathbf{A} = \{A_1, ..., A_n\}$, and define $\mathbf{A}_{-i} = \mathbf{A} - \{A_i\}$. For $i = 1, ..., n$, define $C_i = \sqcap \mathbf{A}_{-i}$ (so each C_i is the conjunction of all A's but A_i). Then, it is easy to verify that $LCS\{C_1, \ldots, C_n\} \equiv \top$ (no concept name A_i is common to all $C_1, \ldots, C_n$).

Now let $\mathbf{B}$ be any subset of $\{1, ..., n\}$, and define $\mathbf{A}_{-\mathbf{B}} = \{A_j \mid j \in \{1, \ldots, n\} - \mathbf{B}\}$. We prove by induction on the cardinality of $\mathbf{B}$ that $LCS\{C_i \mid i \in \mathbf{B}\} = \sqcap \mathbf{A}_{-\mathbf{B}}$,

e.g., if $\mathbf{B} = \{2, 4, 5\}$, then $LCS\{C_2, C_4, C_5\} = A_1 \sqcap A_3 \sqcap A_6 \sqcap \cdots \sqcap A_n$. For $|\mathbf{B}| = 1$, let $\mathbf{B} = \{i\}$ for any $i \in \{1, \ldots, n\}$. Then $LCS\{C_i\} = C_i = \sqcap\mathbf{A}_{-\mathbf{B}}$ by definition. Assuming the claim true for $|\mathbf{B}| = k$, let $i \notin \mathbf{B}$; observe that $LCS\{\{C_i\} \cup \{C_j \mid j \in \mathbf{B}\}\} = LCS\{C_i, LCS\{\{C_j \mid j \in \mathbf{B}\}\}\} = LCS\{C_i, \sqcap\mathbf{A}_{-\mathbf{B}}\}$ by inductive hypothesis. Now observe that C_i contains all concept names but A_i, hence $LCS\{C_i, \sqcap\mathbf{A}_{-\mathbf{B}}\} = \sqcap\mathbf{A}_{-\mathbf{B}\cup\{i\}}$, which proves the claim for $|\mathbf{B}| = k + 1$. Hence, for every $\mathbf{B}$ such that $|\mathbf{B}| = k$, $\sqcap\mathbf{A}_{-\mathbf{B}}$ is a kCS of $C_1, \ldots, C_n$.

Finally, observe that there are $\binom{n}{k}$ possible k-cardinality subsets of $\{1, ..., n\}$, hence there are as many concepts $\sqcap\mathbf{A}_{-\mathbf{B}}$ which are kCS of $C_1, \ldots, C_n$, and for $k = \lceil n/2 \rceil$, $\binom{n}{k} \in \Omega(2^n)$.

Observe that in the proof, for $k = n - 1$, there are exactly n kCS $A_1, \ldots, A_n$— in fact, $LCS\{C_2, ..., C_n\} = A_1$, and similarly for every subset of $n - 1$ concepts among $C_1, \ldots, C_n$. By definition, these $(n-1)CS$ are also BCS, with maximum $k = n - 1$, and moreover, in this case such $(n-1)CS$ are also $BICS$.

It may seem that computing a BCS of $C_1, \ldots, C_n$ adds a source of complexity to the problem of computing their LCS. However, once the k concepts whose LCS is a BCS are identified, the computation reduces to finding an LCS. To identify such k concepts, observe that it is sufficient to inspect the subsumers matrix, and find a component D_j such that T_{D_j} is maximum (observe that such a maximum must be less than n, otherwise $LCS\{C_1, \ldots, C_n\} \not\equiv \top$ and the BCS equals the LCS). Once such a component D_j is found, the BCS must be subsumed by D_j, either strictly, or the BCS is D_j itself. Hence the BCS is $LCS\{C_i \mid s_{ij} = 1\}$, and we can conclude with the following theorem.

Theorem 2. *Let m be the sum of the sizes of $C_1, \ldots, C_n$. Then finding a BCS of $C_1, \ldots, C_n$ amounts to the computation of $O(m^2)$ subsumption tests in $\mathcal{L}$, plus the computation of one LCS.*

4.1 Informative Common Subsumers in $\mathcal{ALN}$

Theorem 3. *For a collection of concept descriptions in $\mathcal{ALN}$ the necessary conditions for solution sets determination exposed in Proposition 4 are also sufficient.*

Proof. Let the Common Signature Class $\bigcap_{sig_{D_j}}$ of a concept component D_j satisfy one of the necessary conditions in Proposition 4 and let LCS_{D_j} be the LCS of the k concepts subsumed by D_j. If $\bigcap_{sig_{D_j}}$ is just a common subsumer and not the least one, $LCS_{D_j} \sqsubset \bigcap_{sig_{D_j}}$, hence LCS_{D_j} must be equivalent to $\bigcap_{sig_{D_j}} \sqcap F$, with F a suitable concept. This means that F subsumes each concept C_i subsumed by D_j hence, according to Definition 6, F must be one of the concept components conjoined in $\bigcap_{sig_{D_j}}$ for the computation of B, BI, L_k and I_k elements.

When a TBox is present—even a simple one, made of axioms $A \doteq C$ where A is a name—Nebel proved that subsumption is PSPACE-hard even for the simple DL $\mathcal{FL}_0$ [14][4]. Hence, in the following complexity analysis we decouple the contribution

[4] However, Nebel himself claims that exponentiality raises from the nesting of the definitions (a concept that defines another that defines another etc.) and that for "bushy but not deep"

of the subsumption tests in the subsumers matrix computation from the computation of different introduced common subsumers.

Theorem 4. *Let $C_1, \ldots, C_n$, $\mathcal{T}$ be n concepts and a simple Tbox in $\mathcal{ALN}$, let m be the sum of the sizes of $C_1, \ldots, C_n$, and let $S(s)$ be a monotone function bounding the cost of deciding $C \sqsubseteq_{\mathcal{T}} D$ in $\mathcal{ALN}$, whose argument s is $|C| + |D| + |\mathcal{T}|$. The computation of the solution sets B, BI, L_k, I_k for a collection of concept descriptions in $\mathcal{ALN}$ is then a problem in $O(m^2 + (S(m))^2)$.*

Proof. We propose an algorithm determining the sets BI, L_k, I_k, B of a collection $\{C_1, \ldots, C_n\}$ of concepts in $\mathcal{ALN}$, whose Subsumers Matrix is given as input. Hence the computation of the Subsumers Matrix can be carried over in polynomial time (see Proposition 2).

According to Theorem 3 the determination of the output sets asks then only for the enumeration of Common Signature Classes of the components D_j chosen according to conditions in Proposition 4.

The algorithm for Common Subsumers enumeration in $\mathcal{ALN}$ is shown in the following:

Input : Subsumers Matrix $S = (s_{ij})$ for a collection of concepts $\{C_1, \ldots, C_n\}$ in $\mathcal{ALN}$, integer $k < n$
Output: L_k; I_k; BI; B

foreach $D_j \in \{D_1, \ldots, D_m\}$ **do**
  if $T_{D_j} \geq k$ **then**
    $L_k = L_k \cup \bigcap_{sig_{D_j}}$;
    if $T_{D_j} < n$ **then**
      if $T_{D_j} = M_S$ **then**
        $B = B \cup \bigcap_{sig_{D_j}}$;
        $BI = BI \cup \bigcap_{sig_{D_j}}$;
      else if $(T_{D_j} = PM_S)$ **then** $BI = BI \cup \bigcap_{sig_{D_j}}$;
      else $I_k = I_k \cup \bigcap_{sig_{D_j}}$;
return L_k,B , I_k, BI;

Algorithm 1: An algorithm for Common Subsumers enumeration in $\mathcal{ALN}$

It is straightforward to verify that the algorithm runs in $O(m^2)$. Given that subsumers matrix can be computed in $O(m^2 + (S(m))^2)$, the claim follows.

4.2 Informative Common Subsumers in $\mathcal{ELN}$

Theorem 5. *The computation of the solution sets B, BI, L_k, I_k for a collection of concept descriptions in $\mathcal{ELN}$ may be reduced to the problem of computing the LCS of the subsets of the collection and may then grow exponential in the size of the collection.*

TBoxes exponentiality does not arise. A precise characterization of what "bushy but not deep" means has been given by Di Noia et al. [15].

Proof. For computing L_k it is sufficient to compute for every subset $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$ the concept $LCS(C_{i_1}, \ldots, C_{i_k})$. The same holds for I_k, excluding those $LCS(C_1, \ldots, \mathcal{C}_k)$ which are equivalent to $LCS(C_1, \ldots, \mathcal{C}_n)$.

For the computation of the sets B and BI, instead, we provide an algorithm that used the one proposed by Kusters and Molitor [11] for LCS computation. The algorithm takes as input the collection $C_1, \ldots, C_n$ represented through its Subsumers Matrix.

Consider now the Concept Components of the elements C_i in the collection: the reduction in Step 2 of Definition 6 causes not all the components to be straightly included in the solution sets B and BI. For example, consider the concept description $C_1 = AssetManager \sqcap \exists hasKnowledge.Psychology$: the resulting concept component is $D_1 = \exists hasKnowledge.\top$ (taking also into account the TBox in Figure 1. Even though such component is selected for the determination of the solution sets according to Proposition 4, it just individuates the concepts in the collection to consider for the determination of B and BI. For each component D_j we denote LCS_{D_j} the LCS of the C_i such that $s_{ij} = 1$.

The proposed algorithm is shown in the following:

Input : Subsumers Matrix $S = (s_{ij})$ for a collection of concepts $C_i \in \{C_1, \ldots, C_n\}$ in $\mathcal{ELN}$
Output: BI; B

1 **if** $M_S = n$ **then**
2 $B := \emptyset$;
3 **foreach** D_j *s.t.* $T_{D_j} = PM_S$ **do** $BI := BI \cup LCS_{D_j}$
4 **else**
5 **foreach** D_j *s.t.* $T_{D_j} = M_S$ **do**
6 $B := B \cup LCS_{D_j}$;
7 $BI := B$
8 **return** B , BI;

Algorithm 2: An algorithm for Common Subsumers enumeration in $\mathcal{ELN}$

The algorithm requires the computation of the LCS of l concepts — with $l \leq n$— in lines 3, 6. Similarly to the approach used in [3] we limit the size on the input collection from n to l, and we compute the LCS of the l concepts as shown in [11]. The problem of determining the solution sets of a collection may be then reduced to the computation of the LCS of subsets of the collection itself.

5 Final Remarks

Motivated by a real-world application need, namely finding Core Competence in knowledge intensive companies, novel reasoning services finding commonalities among portions in a collection of concepts in DL have been investigated and defined in $\mathcal{ALN}$ and $\mathcal{ELN}$. It has been shown that partial and informative common subsumers can be computed in polynomial time in $\mathcal{ALN}$; in $\mathcal{ELN}$ the problem of computing partial and informative common subumers may be instead reduced to LCS computation and the complexity is therefore bounded by the one for LCS.

The computation algorithm for $\mathcal{ALN}$ has been implemented in the framework of Impakt, a novel and optimized commercial system for competences and skills management [8], which will be released late this year by D.O.O.M.srl.

Acknowledgment

We acknowledge the three anonymous reviewers for helpful advices. This work has been supported in part by EU-FP6-IST-26896 project and Apulia Region funded projects PE_013 *Innovative models for customer profiling* and PS_092 *DIPIS*.

References

1. F. Baader and R. Küsters. Computing the least common subsumer and the most specific concept in the presence of cyclic $\mathcal{ALN}$-concept descriptions. In *Proc. of KI-98*, volume 1504, pages 129–140, 1998.
2. F. Baader, R. Küsters, and R. Molitor. Computing least common subsumers in description logics with existential restrictions. In *Proc. of IJCAI '99*, pages 96–101, 1999.
3. F. Baader and R. Molitor. Building and structuring description logic knowledge bases using least common subsumers and concept analysis. In *Proc. of ICCS '00*, pages 292–305, London, UK, 2000.
4. A. Borgida, R.J. Brachman, D. L. McGuinness, and L. Alperin Resnick. CLASSIC: A structural data model for objects. In *Proc. of ACM SIGMOD*, pages 59–67, 1989.
5. W. Cohen, A. Borgida, and H. Hirsh. Computing least common subsumers in description logics. In *Proc. of AAAI-92*, pages 754–760. AAAIP, 1992.
6. W. Cohen and H. Hirsh. The learnability of description logics with equality constraints. *Machine Learning*, 17(2-3):169–199, 1994.
7. W. Cohen and H. Hirsh. Learning the CLASSIC description logics: Theorethical and experimental results. In *Proc. of KR'94*, pages 121–133, 1994.
8. S. Colucci, T. Di Noia, E. Di Sciascio, F.M. Donini, and A. Ragone. Semantic-based skill management for automated task assignment and courseware composition. *Journal of Universal Computer Science*, 13(9):1184–1212, 2007.
9. M. Frazier and L. Pitt. CLASSIC learning. In *Proc. of the 7th Annual ACM Conference on Computational Learning Theory*, pages 23–34, 1994.
10. G. Hamel and C. K. Prahalad. The core competence of the corporation. *Harvard Business Review*, May-June:79–91, 1990.
11. Ralf Kusters and Ralf Molitor. Computing least common subsumers in ALEN. In *IJCAI*, pages 219–224, 2001.
12. T. Mantay, R. Moller, and A. Kaplunova. Computing probabilistic least common subsumers in description logics. In *KI - Kunstliche Intelligenz*, pages 89–100, 1999.
13. R. Möller, V. Haarslev, and B. Neumann. Semantics-based information retrieval. In *Proc. of IT&KNOWS-98*, Vienna, Budapest, 1998.
14. Bernhard Nebel. *Reasoning and Revision in Hybrid Representation Systems*, volume 422 of *LNAI*. Springer, 1990.
15. T. Di Noia, E. Di Sciascio, and F.M. Donini. Semantic matchmaking as non-monotonic reasoning: A description logic approach. *Journal of Artificial Intelligence Research*, 29:269–307, 2007.