Syllogistic Knowledge Bases with Description Logic Reasoners

Ersin Çine
Department of Computer Engineering
İzmir Institute of Technology
İzmir, Turkey
ersincine@gmail.com

***Abstract*—Reasoning is a core topic both for natural intelligence and for artificial intelligence. While syllogistic logics (SLs) are often studied by cognitive scientists for understanding human reasoning, description logics (DLs) are usually studied by computer scientists for performing automated reasoning. Although the studies on both of these logics are extensive, their literatures are interestingly isolated from each other. Firstly, we formally define a practical family of SLs with different levels of expressivity, including a logic which has recently been introduced for automated reasoning. Then, we reveal their theoretical properties either by defining direct algorithms for deductive reasoning or by translation rules for them into relevant DLs. These algorithms and rules prove that (i) two of our SLs (namely `PolSyl` and `NegSyl`) are tractable fragments of DLs, and (ii) other two SLs (namely `ComSyl` and `ComSyl`$^+$) are categorical fragments of DL $\mathcal{ALC}$ and DL $\mathcal{ALCO}$ with general TBoxes, respectively. These findings bridge the gap between (ancient) SLs and (modern) DLs. An immediate result is that it is possible to combine powerful features of both logics, for example, intuitional user interface of an SL and efficient reasoning algorithms for a DL. Finally, we propose a framework for knowledge representation in SLs and link it to sound and complete DL reasoners for automated deduction.**

***Keywords*—syllogistic logic, description logic, knowledge representation, automated deduction, automated reasoning, syllogistic reasoning, syllogism**

I. Introduction

Reasoning is the process of drawing conclusions from facts. Types of reasoning usually fall into two categories:

- Inductive reasoning, or induction is the process of risky generalizations of known facts. Scientific theories are developed thanks to inductions. Statistics is the main tool for this kind of reasoning. A strong induction means that if the premises are true, the conclusion is true with a high probability. An example of strong inductive arguments is: *a human DNA consists of 4 types of nucleotides, a worm DNA consists of 4 types of nucleotides, (...), therefore all animal DNAs consist of 4 types of nucleotides.*
- Deductive reasoning, or deduction is making safe conclusions out of known facts. Mathematical proofs are of this type. Logic is the main tool for this kind of reasoning. A valid deduction means that when the premises are true, it is impossible that the conclusion is false. A well-known example of valid deductive arguments is: *all humans are mortal, Socrates is human, therefore Socrates is mortal.*

Automated reasoning is reasoning which is done by computers without any human guidance. At this point, the inflexible nature of computers bring the problem of knowledge representation into the scene: a solution to automated reasoning problem becomes practical only if the knowledge is stored structurally with consistent semantics. However, in knowledge representation, there is a trade-off between expressive power and algorithmic complexity [1]: when a formalism is equipped with new features in order to represent different kinds of knowledge, the algorithms for reasoning becomes more time-consuming. Because of this, there is a great variety of formalisms for knowledge representation. These formalisms include ontological approaches such as description logic and rule-based approaches such as Datalog.

The family of description logics (DLs) [2], [3], [4] are perhaps the most successful example of knowledge representation formalisms. It includes many logics with different expressivities, which vary from the lightweight DLs such as DL-Lite [5] or $\mathcal{EL}^{++}$ [6] to the typical DLs such as $\mathcal{ALC}$ [7] or $\mathcal{ALCO}$ [8] to the more expressive DLs such as $\mathcal{SHOIQ}$ [9] or $\mathcal{SROIQ}$ [10] to other extensions such as fuzzy [11] or temporal [12] DLs. Their theoretical properties are well-studied. These logics have applications in a wide range of domain from semantic web to medical informatics to software engineering among others [13]. DL reasoners include Racer, FaCT++, Pellet, JFact, HermiT, Konclude and many others.

On the other hand, there has been a long history of studies on syllogistic logics (SLs). Aristotle's SL, the first known formal logic study in history, is sometimes considered as the first formal study in artificial intelligence. It was state-of-the-art until the 19th century [14]. In the 19th century, modern logics superseded Aristotle's syllogisms [15] [16].[1] However, modern logics, first-order logic (FOL) and its extensions, suffer from semi-decidability (i.e. in finite time, no algorithm can prove contradictions in FOL). Furthermore, FOL is unnatural to the humans: its syntax bears no resemblance to the syntax of the natural language sentences [17]. Thus, homophonic theories should be preferred [18]. A candidate for these homophonic theories is naturally, the chronologically first logics, SLs. Today, we witness a revival of SLs. Extentions

[1]From the perspective of cognitive science, it has never been useless: its importance in understanding human reasoning makes the syllogism an active research topic.

such as fuzzy quantification [19], [20], [21], indefinite terms [22], [23] and complex terms [24] are added to SLs in order to acquire more expressive SLs. An extended logic for categorical polysyllogisms (CPS) has recently been introduced for automated deduction [24]. We name the presented logic $\mathtt{ComSyl}^{+}$ and along with that, we define three less expressive SLs including the logic with indefinite terms through syntactic restrictions.[2] $\mathtt{ComSyl}^{+}$ is decidable but an upper bound for algorithmic complexity is unknown.

A proof system is sound if everything that the system concludes is true given the premises, and is complete if everything that is true given the premises can be concluded by the system. The vast majority of studies on automated reasoning focuses on sound and complete deduction as opposed to approximate reasoning. In this work, we study sound and complete deduction.

Deduction services are in different forms such as consistency detection, logical implication or query answering. An algorithmic solution to one of them is usually sufficient: it is trivial to develop other services based on one service. In this paper, we focus on consistency detection.

A. Relevant Description Logics

In DLs, a knowledge base, or ontology, typically consists of a terminological part (TBox) for representing relationships between concepts and an assertional part (ABox) for representing membership of an individual to a concept or relationships between two individuals. Types of TBoxes are the empty TBox, acyclic TBox, cyclic TBox and general TBox.

We will only give definitions of the most relevant DLs, $\mathcal{ALC}$ and $\mathcal{ALCO}$ with general TBoxes.

Let $\mathtt{X}$ be a concept name, $\mathtt{r}$ a role, and $\mathtt{C}$ and $\mathtt{D}$ concepts. The production rule in $\mathcal{ALC}$ is:

$$\mathtt{C},\mathtt{D} ::= \bot \mid \top \mid \mathtt{X} \mid \neg\mathtt{C} \mid \mathtt{C}\sqcap\mathtt{D} \mid \mathtt{C}\sqcup\mathtt{D} \mid \exists\mathtt{r}.\mathtt{C} \mid \forall\mathtt{r}.\mathtt{C}$$

Let $\mathtt{x}$ be an individual. $\mathcal{ALCO}$ introduces nominals:

$$\mathtt{C},\mathtt{D} ::= \bot \mid \top \mid \mathtt{X} \mid \neg\mathtt{C} \mid \mathtt{C}\sqcap\mathtt{D} \mid \mathtt{C}\sqcup\mathtt{D} \mid \exists\mathtt{r}.\mathtt{C} \mid \forall\mathtt{r}.\mathtt{C} \mid \{\mathtt{x}\}$$

An ABox contains concept assertions which involve a concept and an individual (e.g. Aristotle is a human: $\mathtt{Human}(\mathtt{Aristotle})$) and role assertions which involve a role and two individuals (e.g. Aristotle is a student of Plato: $\mathtt{studentOf}(\mathtt{Aristotle},\mathtt{Plato})$).

A general TBox contains general concept inclusions between two concepts (e.g. A game player is a human who plays some games: $\mathtt{GamePlayer} \sqsubseteq \mathtt{Human} \sqcap \exists\mathtt{play}.\mathtt{Game}$).

Semantics of $\mathcal{ALC}$ and $\mathcal{ALCO}$ are similar to those of SLs. Thus, we will omit them for saving space.

Reasoning in $\mathcal{ALC}$ and $\mathcal{ALCO}$ with general TBoxes are EXPTIME-complete.

[2]Here, we make small changes: existential imports and the surplus term become syntactic sugars, and the symbol '$\mathtt{S}$' (singula / singular) replaces '$\mathtt{IND}$' (individual). We also omit $\mathtt{S}$s as queries for compatibility with DLs and introduce non-empty domain assumption in order to follow the standard semantics of first-order logic.

II. A Family of Syllogistic Logics

In increasing order of expressive power, our SLs are:

1) $\mathtt{PolSyl}$ (CPS with atomic terms),
2) $\mathtt{NegSyl}$ (CPS with possibly negated atomic terms),
3) $\mathtt{ComSyl}$ (CPS with complex terms),
4) $\mathtt{ComSyl}^{+}$ ($\mathtt{ComSyl}$ with individuals).

A. Syntax

Let $\mathtt{X}$ be an atomic term, $\mathtt{P}$ a proposition, and $\mathtt{C}$ and $\mathtt{D}$ complex terms. The production rules for complex terms and propositions are defined as:

$$\mathtt{C},\mathtt{D} ::= \mathtt{X} \mid \top \mid \bot \mid \neg\mathtt{C} \mid \mathtt{C}\sqcap\mathtt{D} \mid \mathtt{C}\sqcup\mathtt{D}$$

$$\mathtt{P} ::= \mathtt{A}(\mathtt{C},\mathtt{D}) \mid \mathtt{E}(\mathtt{C},\mathtt{D}) \mid \mathtt{I}(\mathtt{C},\mathtt{D}) \mid \mathtt{O}(\mathtt{C},\mathtt{D}) \mid \mathtt{S}(\mathtt{C})$$

For example, $\neg\mathtt{smart}\sqcap\mathtt{cat}$ is a complex term, and $\mathtt{I}(\neg\mathtt{expensive}\sqcap(\mathtt{car}\sqcup\mathtt{motorcycle}),\mathtt{red})$ and $\mathtt{S}(\mathtt{Socrates})$ are propositions.

Note that an atomic term is usually a simple noun (e.g. $\mathtt{cat}$) or adjective (e.g. $\mathtt{smart}$). A negated atomic term (or an indefinite term) is a complement of an atomic term (e.g. $\neg\mathtt{cat}$, or $\neg\mathtt{smart}$). Complex terms inherently include both atomic terms and negated atomic terms among others.

Our SLs are defined with syntactic restrictions applied on this logic. Table I shows definitions of these logics.

TABLE I: Syntactic Restrictions

Logic	**Allowed Terms**[3]			**Allowed Propositions**[4]				
	X	¬X	C	A	E	I	O	S
PolSyl	✓			✓	✓	✓	✓	
NegSyl	✓	✓		✓	✓	✓	✓	
ComSyl	✓	✓	✓	✓	✓	✓	✓	
ComSyl^{+}	✓	✓	✓	✓	✓	✓	✓	✓

B. Semantics

Let $\mathtt{C}$ and $\mathtt{D}$ be complex terms. An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty set of individuals $\Delta^{\mathcal{I}}$ called the domain and an interpretation function $\cdot^{\mathcal{I}}$ that maps C to a set $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ such that

1) $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$,
2) $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$,
3) $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$.

$\bot$ and $C \sqcup D$ can be seen as syntactic sugars which are synonyms to $\neg\top$ and $\neg(\neg C \sqcap \neg D)$, respectively. Thus it is the case that $\bot^{\mathcal{I}} = \emptyset$ and $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$.

Table II shows all possible types of propositions and their corresponding conditions. A proposition holds if and only if its corresponding condition is met.

[3]The symbols $\mathtt{X}$, $\neg\mathtt{X}$ and $\mathtt{C}$ stand for atomic terms, negated atomic terms and complex terms, respectively.

[4]Propositions of type $\mathtt{S}$, or individuality propositions may be allowed only as assertions, not as queries. The other types of propositions, or quantified propositions may be assertions or queries.

TABLE II: Semantics of Propositions

Proposition	Condition	English
$\mathcal{I} \vDash \mathtt{A(C,D)}$	$\mathtt{C}^{\mathcal{I}} \setminus \mathtt{D}^{\mathcal{I}} = \emptyset$	All C are D
$\mathcal{I} \vDash \mathtt{E(C,D)}$	$\mathtt{C}^{\mathcal{I}} \cap \mathtt{D}^{\mathcal{I}} = \emptyset$	No C are D
$\mathcal{I} \vDash \mathtt{I(C,D)}$	$\mathtt{C}^{\mathcal{I}} \cap \mathtt{D}^{\mathcal{I}} \neq \emptyset$	Some C are D
$\mathcal{I} \vDash \mathtt{O(C,D)}$	$\mathtt{C}^{\mathcal{I}} \setminus \mathtt{D}^{\mathcal{I}} \neq \emptyset$	Some C are not D
$\mathcal{I} \vDash \mathtt{S(C)}$	$\lvert\mathtt{C}^{\mathcal{I}}\rvert = 1$	C is an individual

Syntax and semantics of syllogistic logics are intuitive. An example of valid syllogisms $\mathtt{I(Turkish \sqcap engineer, smart \sqcap person)}, \mathtt{A(person, mammal)} \vDash \mathtt{I(mammal, engineer)}$ corresponds to the following argument in English: *Some Turkish engineers are smart people. All people are mammals. Therefore, some mammals are engineers.*

III. Syllogistic Knowledge Bases

A knowledge base, or an ontology, or in this case a syllogism, is a set of propositions. In this section, we define a framework for representing a knowledge base and its parts: terms and propositions. Figure 1 illustrates the high-level components of the framework.

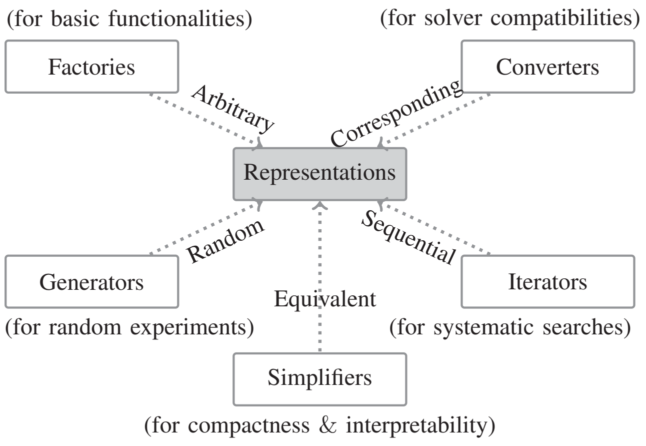


Fig. 1: Factories, generators, iterators and converters create arbitrary, random, sequential and equivalent representations of terms, propositions and syllogisms.

A. Representation of Terms

As explained in [24], region sets of atomic terms are calculating using the following formula:

$$X_i = \{ \sum_{n \in N} 2^{n-1} \mid N \subseteq \{1, 2, ..., t\}, i \in N \}$$

Figure 2 illustrates the region set representation and an alternative representation for terms.

Note that region sets require that t, the number of atomic terms, is pre-determined and assume that it is fixed. An advantage of region sets is that they are always in their simplest form. Along with that, when t is small, visualization of region sets is useful for research and education. On the other

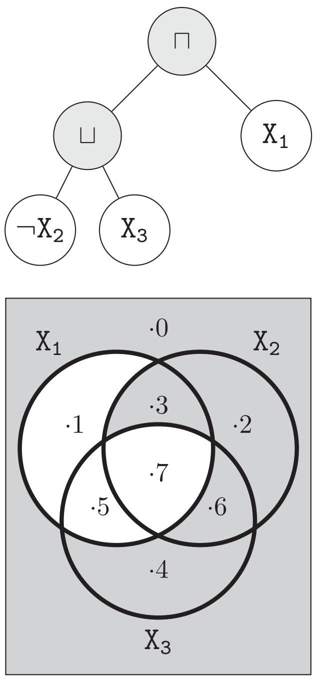


Fig. 2: A term can be expressed as an expression tree or a region set. $(\neg X_2 \sqcup X_3) \sqcap X_1$ corresponds to $\{1, 5, 7\}$ when $t = 3$.

hand, expression trees are more compact and they enable lazy evaluation.

Conversion from expression trees to region sets is trivial: atomic terms are replaced with the corresponding region sets according to the formula and then the expression is evaluated. Conversion from region sets to expression trees can be done via Quine-McCluskey algorithm [26] or Petrick's method [27].

B. Representation of Propositions

We have defined 5 types of propositions. These propositions are called *relational propositions*, or *relations*. In essence, those propositions express three types of *cardinality propositions*, or *cardinalities*: empty ($\nexists$), non-empty ($\exists$), or singleton ($\exists!$) sets. Table III shows correspondence of these proposition types.

TABLE III: Correspondence Between Proposition Types

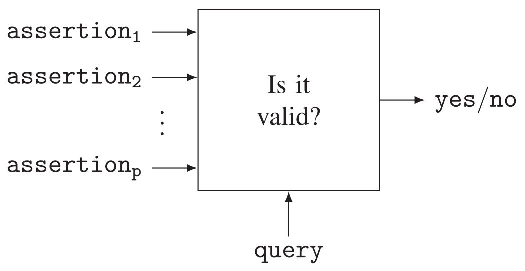
Relation	Cardinality
A(C, D)	$\nexists(\mathtt{C} \sqcap \neg\mathtt{D})$
E(C, D)	$\nexists(\mathtt{C} \sqcap \mathtt{D})$
I(C, D)	$\exists(\mathtt{C} \sqcap \mathtt{D})$
O(C, D)	$\exists(\mathtt{C} \sqcap \neg\mathtt{D})$
S(C)	$\exists!(\mathtt{C})$

Cardinality	Relation
$\nexists(\mathtt{C})$	E(C, C)
$\exists(\mathtt{C})$	I(C, C)
$\exists!(\mathtt{C})$	S(C)

Note that negations of these propositions other than those of type S (and $\exists!$) exist within these logics: their negations can be used in need.

C. Representation of Syllogisms

The natural representation of a syllogism is a logical argument: validity of $((a_1 \wedge a_2 \wedge \cdots \wedge a_p) \Rightarrow q)$ is the question to answer. An alternative representation of a syllogism is a set of propositions: the question is inconsistency of $\{a_1, a_2, \ldots, a_p, \neg q\}$. Figure 3 illustrates these representations.



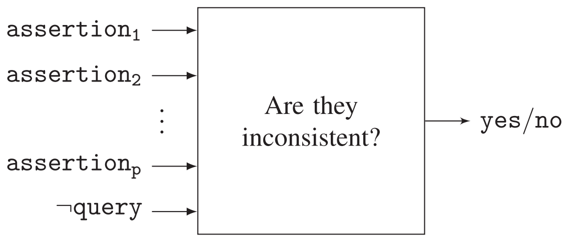


Fig. 3: A syllogism can be represented as validity of an argument or inconsistency of propositions

Conversion between syllogism representations is trivial and done via the following equivalence: $((a_1 \wedge a_2 \wedge \cdots \wedge a_p) \Rightarrow q)$ is valid $\equiv \{a_1, a_2, \ldots, a_p, \neg q\}$ is inconsistent.

IV. Theoretical Properties

Both DLs and SLs are extended by definitions. Thus, their intrarelationships require no proof: $\mathcal{ALC} \subset \mathcal{ALCO}$ and `PolSyl` $\subset$ `NegSyl` $\subset$ `ComSyl` $\subset$ `ComSyl`$^+$.

It is also obvious that all of these logics are monotonic: once a proposition is shown to be true, no further propositions can change that truth.

In this section, we will reveal theoretical properties of SLs and their relationships with the DLs.

A. Expressivity

Let X and Y be atomic terms. All 16 possible relations between X and Y in `NegSyl` constitute 8 groups of semantically equivalent propositions:

1) $\mathtt{A}(\mathtt{X},\mathtt{Y}) \equiv \mathtt{E}(\mathtt{X},\neg\mathtt{Y}) \equiv \nexists(\mathtt{X} \sqcap \neg\mathtt{Y})$
2) $\mathtt{E}(\mathtt{X},\mathtt{Y}) \equiv \mathtt{A}(\mathtt{X},\neg\mathtt{Y}) \equiv \nexists(\mathtt{X} \sqcap \mathtt{Y})$
3) $\mathtt{A}(\neg\mathtt{X},\mathtt{Y}) \equiv \mathtt{E}(\neg\mathtt{X},\neg\mathtt{Y}) \equiv \nexists(\neg\mathtt{X} \sqcap \neg\mathtt{Y})$
4) $\mathtt{E}(\neg\mathtt{X},\mathtt{Y}) \equiv \mathtt{A}(\neg\mathtt{X},\neg\mathtt{Y}) \equiv \nexists(\neg\mathtt{X} \sqcap \mathtt{Y})$
5) $\mathtt{I}(\mathtt{X},\mathtt{Y}) \equiv \mathtt{O}(\mathtt{X},\neg\mathtt{Y}) \equiv \exists(\mathtt{X} \sqcap \mathtt{Y})$
6) $\mathtt{O}(\mathtt{X},\mathtt{Y}) \equiv \mathtt{I}(\mathtt{X},\neg\mathtt{Y}) \equiv \exists(\mathtt{X} \sqcap \neg\mathtt{Y})$
7) $\mathtt{I}(\neg\mathtt{X},\mathtt{Y}) \equiv \mathtt{O}(\neg\mathtt{X},\neg\mathtt{Y}) \equiv \exists(\neg\mathtt{X} \sqcap \mathtt{Y})$
8) $\mathtt{O}(\neg\mathtt{X},\mathtt{Y}) \equiv \mathtt{I}(\neg\mathtt{X},\neg\mathtt{Y}) \equiv \exists(\neg\mathtt{X} \sqcap \neg\mathtt{Y})$

As a result of the correspondence above, using cardinality propositions, a term is an element of $\{\mathtt{X} \sqcap \mathtt{Y},\ \mathtt{X} \sqcap \neg\mathtt{Y},\ \neg\mathtt{X} \sqcap \mathtt{Y},\ \neg\mathtt{X} \sqcap \neg\mathtt{Y}\}$. Special terms X, $\neg$X and $\bot$ are derived when the atomic terms are equal (i.e. $\mathtt{X} = \mathtt{Y}$). We will call these terms "regular" when the atomic terms are unequal (i.e. $\mathtt{X} \neq \mathtt{Y}$). Note that these regular terms represent all 4 regions in the Venn diagram of two atomic terms.

Commutative property of intersection, unequal atomic terms in the term expressions, and 4 regions of 2 atomic terms reveal the true expressivity of `NegSyl`: it allows for asserting emptiness or non-emptiness of $2t^2 + 1$ pairwise unequal sets: The bottom term $\bot$, t different atomic terms, t different negated atomic terms, and $4 \times \frac{t \times (t-1)}{2}$ regular terms.

Expressiveness of `PolSyl` is calculated very similarly. It can express 3 regions out of 4 regions between two atomic terms: $\{\mathtt{X} \sqcap \mathtt{Y},\ \mathtt{X} \sqcap \neg\mathtt{Y},\ \neg\mathtt{X} \sqcap \mathtt{Y}\}$. X and $\bot$ can be derived when $\mathtt{X} \neq \mathtt{Y}$. In total, $\frac{3}{2}t^2 - \frac{1}{2}t + 1$ pairwise unequal sets can be represented.

t atomic terms constitute 2^t regions in a Venn diagram. `ComSyl` and `ComSyl`$^+$ are able to express all subsets of those regions: there are 2^{2^t} pairwise unequal sets to represent.

B. Complexity of `PolSyl` and `NegSyl`

Theorem 1. *Consistency detection in* `NegSyl` *is tractable.*

Proof. Algorithm 1 is a sound and complete algorithm for consistency detection which will always terminate in polynomial time (PTIME, or P in short).

Algorithm. This algorithm consists of three stages: first, it searches for direct inconsistencies in the knowledge base. If both of a proposition and its negation appear in the knowledge base, then there is inconsistency. In the second stage, negations of immediately inferable propositions are searched. In the third stage, consistency with the non-empty domain assumption is checked. The second and the third stages are repeated until there is no change. The function $immediatelyImplies$ checks whether a given proposition is immediately inferable given a knowledge base. The function $implies$ checks the immediate inferences ignoring most of the propositions: it adds p to a copy of $kb.knowns$, then filters the propositions that contain irrelevant atomic terms, and then runs the exhaustive search in [24].

Soundness and completeness. Soundness of the algorithm relies on the soundness of the exhaustive search in [24]. The most important reason of the low time complexity is the proposition elimination in the function $implies$: the algorithm ignores many combinations of propositions. However, the elimination of the propositions does not violate the completeness. Because polysyllogisms can be expressed as a chain of multiple syllogisms: when it is possible to construct a bridge between two terms, it is possible to construct that bridge using a single middle term.

Time complexity. In the algorithm design, we take advantage of the monotonicity. The algorithm is guaranteed to converge in polynomial time: the number of all propositions is $\Theta(t^2)$ and in the worst-case, a single proposition will be inferred in each iteration. Numbers of iterations in the inner loops are also a polynomial. Algorithm 2 is a linear-time algorithm: there are $\Theta(t)$ iterations in each call and the innermost function call takes a constant time as the upper bound for the number of atomic terms is fixed to 3. These facts make the whole algorithm polynomial.

□

Algorithm 1 Consistency detection in `NegSyl`

```
function IsConsistent(kb)                      ▷ kb is knowledge base
  // Checking consistency of current knowledge:
  for all p $\in$ kb.knowns do
    if p.negation() $\in$ kb.knowns then
      kb.fillKnownsClearUnknowns()
      return False
  repeat
    // Checking consistency of inferable knowledge:
    for all p $\in$ kb.unknowns do
      if immediatelyImplies(kb, p) then
        kb.addToKnownsRemoveFromUnknowns(p)
        if p.negation() $\in$ kb.knowns then
          kb.fillKnownsClearUnknowns()
          return False
    // Checking domain non-emptiness:
    for all X $\in$ kb.atomicTerms do
      if $\{\nexists$(X), $\nexists(\neg$X)$\}\subseteq$ kb.knowns then
        kb.fillKnownsClearUnknowns()
        return False
  until convergence
  return True
```

Algorithm 2 Immediate inferences in `NegSyl`

```
function ImmediatelyImplies(kb, p)
  XY = p.relevantAtomicTerms                   ▷ $0 \le |XY| \le 2$
  tautologies = $\{\nexists(\bot), \exists(\top)\}$
  if p $\in$ tautologies then
    return True
  for all Z $\in$ kb.atomicTerms do
    XYZ = XY $\cup$ {Z}                           ▷ $1 \le |XYZ| \le 3$
    if implies(kb, p, XYZ) then
      return True
  return False
```

Corollary 1.1. *Consistency detection in* `PolSyl` *is tractable.*

Proof. As discussed before, `PolSyl` is a fragment of `NegSyl`. As a result, consistency detection in `PolSyl` is tractable (i.e. in PTIME).

□

C. Complexity of `ComSyl` and `ComSyl`$^+$

Theorem 2. `ComSyl` *is a fragment of $\mathcal{ALC}$ with general TBox.*

Proof. Terms of both logics have the same syntax. We will define translation rules for `ComSyl` propositions into $\mathcal{ALC}$ axioms.

Let `C` and `D` be complex terms. Propositions of type `A` and `E` correspond to general concept inclusions:

$$\texttt{A(C,D)} \equiv (C \sqsubseteq D)$$

$$\texttt{E(C,D)} \equiv (C \sqcap D \sqsubseteq \bot)$$

Let rnd_1 and rnd_2 be random individuals such that no further knowledge about them can be found in the ontology. Propositions of type `I` and `O` correspond to assertions on random individuals:

$$\texttt{I(C,D)} \equiv F(rnd_1)\ such\ that\ F \equiv C \sqcap D$$

$$\texttt{O(C,D)} \equiv F(rnd_2)\ such\ that\ F \equiv C \sqcap \neg D$$

□

Corollary 2.1. `ComSyl` *is categorical fragment of $\mathcal{ALC}$ with general TBox.*

Proof. The only syntactic difference between `ComSyl` terms and $\mathcal{ALC}$ concepts is that $\mathcal{ALC}$ allow us to define concepts via non-categorical roles. When the only role is the categorical role *be* (or, *beSubsetOf*) the semantics of $\exists r.C$ and $\forall r.C$ can be represented via propositions of type $\exists$ and $\nexists$, respectively. □

Corollary 2.2. `ComSyl`$^+$ *is categorical fragment of $\mathcal{ALCO}$ with general TBox.*

Proof. The relationship between `ComSyl`$^+$ and $\mathcal{ALCO}$ is very similar to that of `ComSyl` and $\mathcal{ALC}$. The only addition is the propositions of type `S` which correspond to nominals in DLs.

Let rnd be a random individual such that no further knowledge about it can be found in the ontology. Propositions of type `S` corresponds to the following concept equivalence:

$$\texttt{S(C)} \equiv (C \equiv \{rnd\})$$

A concept equivalence can be defined using two general concept inclusions (e.g. $C \equiv \{rnd\}$ can be defined using $C \sqsubseteq \{rnd\}$ and $\{rnd\} \sqsubseteq C$).

□

Corollary 2.3. *Consistency detection in* `ComSyl` *and* `ComSyl`$^+$ *are in EXPTIME.*

Proof. It is already stated that consistency detection in $\mathcal{ALC}$ and $\mathcal{ALCO}$ with respect to general TBoxes is EXPTIME-complete. We proved that `ComSyl` and `ComSyl`$^+$ are fragments of those logics. Thus, they are in EXPTIME. □

In the software framework, the correspondence between SLs and OWL API of DLs is as below:

- Terms correspond to *OWLClassExpression*s.
- Propositions of type `A` and `E` correspond to *OWLSubClassOfAxiom*s and *OWLDisjointClassesAxiom*s, respectively.
- Propositions of type `I` and `O` correspond to *OWLClassAssertionAxiom*s.
- Propositions of type `S` correspond to *OWLEquivalentClassesAxiom*s.

D. Relationships Between SLs and DLs

Figure 4 illustrates the relationssships of SLs and DLs. The complexities of DLs in the figure are exact and with respect to general TBoxes. $(\neg)$ indicates atomic negation.

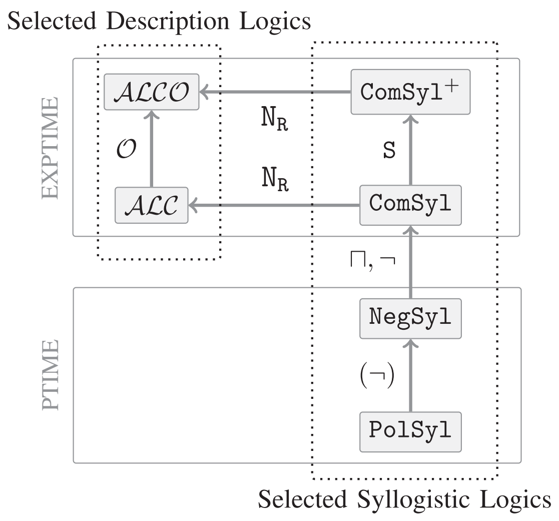


Fig. 4: Arrows show extensions (Expressiveness increases in the arrow directions). Vertical arrows show extensions by definition while horizontal arrows show role generalizations.

V. Conclusion

We have defined a family of SLs: $\mathtt{PolSyl} \subset \mathtt{NegSyl} \subset \mathtt{ComSyl} \subset \mathtt{ComSyl}^+$. We proved that PolSyl and NegSyl are tractable (i.e. they are in PTIME). We also proved that ComSyl and $\mathtt{ComSyl}^+$ are categorical fragments of $\mathcal{ALC}$ and $\mathcal{ALCO}$, respectively, and therefore, they are in EXPTIME. Note that the lower bounds for algorithmic complexities have not yet been proved.

These findings allow us to combine the best of both worlds: syntax of SLs and reasoners for DLs. Both intuitiveness of the natural syntax of SLs and time complexity of DLs in real world ontologies are already proven. This bridge is a chance to utilize the big data of natural language through an ontology learner as well as to develop easy-to-use end-user programs for manual knowledge management.

In this work, we developed a prototypical implementation of the software framework in Java and linked it to a DL reasoner, HermiT. As future work, we will develop the software framework beyond the prototypical implementation and perform experiments.

Furthermore, as future work, it is interesting to investigate if other features such as non-categorical roles or extended quantifiers can be added to ComSyl or $\mathtt{ComSyl}^+$ without sacrificing intuitiveness and algorithmic properties much.

Most complex terms are unnecessarily complex for expressing everyday natural language sentences: a new logic can be defined via restricting terms of ComSyl or $\mathtt{ComSyl}^+$ to including only a couple of operators rather than possibly infinite number of them.

References

[1] R. J. Brachman and H. J. Levesque, "The tractability of subsumption in frame-based description languages," in *AAAI*, vol. 84, 1984, pp. 34–37.

[2] M. Krötzsch, F. Simancik, and I. Horrocks, "A description logic primer," *arXiv preprint arXiv:1201.4089*, 2012.

[3] S. Rudolph, "Foundations of description logics," in *Reasoning Web. Semantic Technologies for the Web of Data*. Springer, 2011, pp. 76–136.

[4] F. Baader, I. Horrocks, C. Lutz, and U. Sattler, *Introduction to Description Logic*. Cambridge University Press, 2017.

[5] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati, "Tractable reasoning and efficient query answering in description logics: The dl-lite family," *Journal of Automated reasoning*, vol. 39, no. 3, pp. 385–429, 2007.

[6] F. Baader, S. Brandt, and C. Lutz, "Pushing the el envelope further," 2008.

[7] F. M. Donini and F. Massacci, "Exptime tableaux for alc," *Artificial Intelligence*, vol. 124, no. 1, pp. 87–138, 2000.

[8] S. Grimm and P. Hitzler, "A preferential tableaux calculus for circumscriptive alco," in *International Conference on Web Reasoning and Rule Systems*. Springer, 2009, pp. 40–54.

[9] Y. Kazakov and B. Motik, "A resolution-based decision procedure for shoiq," in *International Joint Conference on Automated Reasoning*. Springer, 2006, pp. 662–677.

[10] I. Horrocks, O. Kutz, and U. Sattler, "The even more irresistible sroiq." *Kr*, vol. 6, pp. 57–67, 2006.

[11] S. Moral *et al.*, "Fuzzy description logics–a survey," in *Scalable Uncertainty Management: 11th International Conference, SUM 2017, Granada, Spain, October 4-6, 2017, Proceedings*, vol. 10564. Springer, 2017, p. 31.

[12] C. Lutz, F. Wolter, and M. Zakharyaschev, "Temporal description logics: A survey," in *Temporal Representation and Reasoning, 2008. TIME'08. 15th International Symposium on*. IEEE, 2008, pp. 3–14.

[13] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, *The Description Logic Handbook: Theory, Implementation and Applications*, 2nd ed. New York, NY, USA: Cambridge University Press, 2010.

[14] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2009.

[15] G. Frege, *Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*. L. Nebert, 1879.

[16] E. M. Hammer, "Semantics for existential graphs," *Journal of Philosophical Logic*, vol. 27, no. 5, pp. 489–503, 1998.

[17] W. V. Quine, *Philosophy of logic*, 2nd ed. Harvard University Press, 1986.

[18] G. Evans, "Pronouns, quantifiers, and relative clauses (i)," *Canadian journal of philosophy*, vol. 7, no. 3, pp. 467–536, 1977.

[19] D. G. Schwartz, "Qualified syllogisms with fuzzy predicates," *International Journal of Intelligent Systems*, vol. 29, no. 10, pp. 926–945, 2014.

[20] M. Pereira-Fariña, J. C. Vidal, F. Díaz-Hermida, and A. Bugarín, "A fuzzy syllogistic reasoning schema for generalized quantifiers," *Fuzzy Sets and Systems*, vol. 234, pp. 79–96, 2014.

[21] P. Murinová and V. Novák, "Intermediate syllogisms in fuzzy natural logic," *Journal of Fuzzy Set Valued Analysis*, vol. 2016, no. 2, pp. 99–111, 2016.

[22] E. Alvarez and M. Correia, "Syllogistic with indefinite terms," *History and Philosophy of Logic*, vol. 33, no. 4, pp. 297–306, 2012.

[23] E. Alvarez-Fontecilla, "Canonical syllogistic moods in traditional aristotelian logic," *Logica Universalis*, vol. 10, no. 4, pp. 517–531, 2016.

[24] E. Çine and B. İ. Kumova, "An extended syllogistic logic for automated reasoning," in *Computer Science and Engineering (UBMK), 2017 International Conference on*. IEEE, 2017, pp. 759–763.

[25] L. A. Nguyen and J. Golińska-Pilarek, "Exptime tableaux with global caching for the description logic shoq," *arXiv preprint arXiv:1405.7221*, 2014.

[26] E. J. McCluskey, "Minimization of boolean functions," *Bell Labs Technical Journal*, vol. 35, no. 6, pp. 1417–1444, 1956.

[27] S. R. Petrick, "A direct determination of the irredundant forms of a boolean function from the set of prime implicants," *Air Force Cambridge Res. Center Tech. Report*, pp. 56–110, 1956.