KIT REPORT 91

A Correspondence Theory for Terminological Logics: Preliminary Report

Klaus Schild

Technische Universität Berlin
Projekt KIT–BACK, FR 5–12
Franklinstr. 28/29, W–1000 Berlin 10

August 1991

Abstract

We show that the terminological logic $\mathcal{ALC}$ comprising Boolean operations on concepts and value restrictions is a notational variant of the propositional modal logic $\mathbf{K_{(m)}}$. To demonstrate the utility of the correspondence, we give two of its immediate by-products. Namely, we axiomatize $\mathcal{ALC}$ and give a simple proof that subsumption in $\mathcal{ALC}$ is PSPACE-complete, replacing the original six-page one.

Furthermore, we consider an extension of $\mathcal{ALC}$ additionally containing both the identity role and the composition, union, transitive-reflexive closure, range restriction, and inverse of roles. It turns out that this language, called $\mathcal{TSL}$, is a notational variant of the propositional dynamic logic *converse*-PDL. Using this correspondence, we prove that it suffices to consider finite $\mathcal{TSL}$-models, show that $\mathcal{TSL}$-subsumption is decidable, and obtain an axiomatization of $\mathcal{TSL}$.

By discovering that features correspond to deterministic programs in dynamic logic, we show that adding them to $\mathcal{TSL}$ preserves decidability, although violates its finite model property. Additionally, we describe an algorithm for deciding the coherence of inverse-free $\mathcal{TSL}$-concepts with features. Finally, we prove that universal implications can be expressed within $\mathcal{TSL}$.

1 Motivation

We shall establish correspondences between terminological logics and propositional modal and dynamic logics. These correspondences turn out to be highly productive because formerly unrelated fields are brought together. In the area of terminological logics, running systems such as BACK, CLASSIC, KL-ONE, KRYPTON, and LOOM have been developed since the late seventies. Only recently theoretical investigations have been undertaken mainly concerning the computational complexity of terminological logics.[1] In the very contrast to that, elaborated theories for modal and dynamic logics have been developed much earlier.[2] Particularly for modal logic there is—apart from first order logic—the most elaborated theory, and dynamic logic has benefited from these results. By detecting these correspondences, we gain new insights into terminological logics solely by expounding the theorems of modal and dynamic logic as theorems of the corresponding terminological logic. There can also be redundant research if correspondences are overlooked. For instance, Ladner [1977] showed that the propositional modal logic $\mathbf{K_{(m)}}$ is PSPACE-complete, and twelve years later this was reproved by Schmidt-Schauß and Smolka [1991] for its notational variant $\mathcal{ALC}$.

2 Preliminaries

To understand the complexity results to be presented, you should know the complexity classes P, NP, PSPACE, and EXPTIME: P is the class of problems decidable in deterministic polynomial time, NP are those problems decidable in nondeterministic polynomial time, PSPACE are the problems decidable in deterministic polynomial space, and EXPTIME are those problems decidable in deterministic exponential time. Furthermore, you should know that a problem L is *PSPACE-complete* iff both $L \in$ PSPACE and each problem in PSPACE is log space reducible to L; a problem M is *log space reducible* to L iff there is a function $f : M \to L$ computable in space log (and therefore also computable in polynomial time) such that for all x, $x \in M$ iff $f(x) \in L$. Be aware of the complexity hierarchy P $\subseteq$ NP $\subseteq$ PSPACE $\subseteq$ EXPTIME for which it is only *known* that P $\neq$ EXPTIME.

[1] Confer [Nebel, 1990] for a good overview of the systems and the complexity results.

[2] For the history of modal logic confer [Hughes and Cresswell, 1984], and for that of dynamic logic confer [Harel, 1984].

3 Terminological Logics and Modal Logics

We first consider a terminological logic investigated by Schmidt-Schauß and Smolka [1991], named $\mathcal{ALC}$. Like any other terminological logic, $\mathcal{ALC}$ comprises *concepts*, denoting sets, as well as *roles*, which denote binary relations. Contrary to roles, concepts can be compound, viz. by intersection $\sqcap$, union $\sqcup$, complementation $\neg$, and the *value restrictions* $\forall.$ and $\exists.$ taking a role and a concept as their arguments. $\forall R.C$ is to be read as "all objects for which all R's are in C," whereas $\exists R.C$ is to be read as "all objects for which there exists an R in C." So we can express concepts such as 'mothers having only sons' by the $\mathcal{ALC}$-expression

$$women \sqcap \neg men \sqcap (\exists child.men) \sqcap \forall child.(men \sqcap \neg women).$$

Formally, $\boldsymbol{\mathcal{ALC}}$ is given by the following formation rules, where c denotes a concept symbol and r a role symbol:

$$\begin{array}{rcl} C, D & \rightarrow & c \mid \top \mid C \sqcap D \mid \neg C \mid \forall R.C \\ R & \rightarrow & r \end{array}$$

As usually, we specify the formal semantics of $\mathcal{ALC}$ by an *extension function*. Let $\mathcal{D}$ be any set, called the *domain*. An *extension function $\mathcal{E}$ over $\mathcal{D}$* is a function mapping concepts to subsets of $\mathcal{D}$ and roles to subsets of $\mathcal{D} \times \mathcal{D}$ such that

$$\begin{array}{rcl} \mathcal{E}[\top] & = & \mathcal{D} \\ \mathcal{E}[C \sqcap D] & = & \mathcal{E}[C] \cap \mathcal{E}[D] \\ \mathcal{E}[\neg C] & = & \mathcal{D} \setminus \mathcal{E}[C] \\ \mathcal{E}[\forall R.C] & = & \{d \in \mathcal{D} : \forall \langle d, e\rangle \in \mathcal{E}[R].e \in \mathcal{E}[C]\} \end{array}$$

Using extension functions, we can define the semantic notions *subsumption*, *equivalence*, and *coherence*: *D subsumes C*, written $\models \boldsymbol{C \sqsubseteq D}$, iff for each extension function $\mathcal{E}$, $\mathcal{E}[C] \subseteq \mathcal{E}[D]$, whereas C and D are *equivalent*, written $\models \boldsymbol{C = D}$, iff for each extension function $\mathcal{E}$, $\mathcal{E}[C] = \mathcal{E}[D]$. Finally, C is *coherent* iff there is an extension function $\mathcal{E}$ with $\mathcal{E}[C] \neq \emptyset$, called a *model for C*; otherwise C is *incoherent*. The following lemma justifies merely investigating subsumption in $\mathcal{ALC}$:

Lemma 1 *Subsumption, equivalence, and incoherence are log space reducible to each other in any terminological logic comprising Boolean operations on concepts.*

Proof: Since $\models C \sqsubseteq D$ iff $\models C \sqcap D = C$, subsumption is log space reducible to equivalence in presence of $\sqcap$. Equivalence in turn is log space reducible to incoherence in presence of $\sqcap$ as well as $\neg$ because $\models C = D$ iff $(C \sqcap \neg D) \sqcup (D \sqcap \neg C)$ is incoherent. And finally, incoherence clearly is log space reducible to subsumption in presence of $\neg\top$ since C is incoherent iff $\models C \sqsubseteq \neg\top$. $\square$

We take the union of concepts $\sqcup$ and the operation $\exists.$ dual to $\forall.$ with $\mathcal{E}[\exists R.C] = \{d \in \mathcal{D} : \exists \langle d, e\rangle \in \mathcal{E}[R].e \in \mathcal{E}[C]\}$ as abbreviations for linearly length-bounded $\mathcal{ALC}$-expressions:

$$
\begin{aligned}
C \sqcup D &\stackrel{\text{def}}{=} \neg(\neg C \sqcap \neg D)\\
\exists R.C &\stackrel{\text{def}}{=} \neg\forall R.\neg C
\end{aligned}
$$

It is well known that $\mathcal{ALC}$ is a sublanguage of first order logic since atomic concepts correspond to one-place predicates and atomic roles to two-place predicates. The $\mathcal{ALC}$-concept $\neg c_1 \sqcup \forall r.(c_2 \sqcap c_3)$, for instance, can be expressed by the first order formula $\neg c_1(x) \vee \forall y.r(x, y) \Rightarrow c_2(y) \wedge c_3(y)$.

Viewing $\mathcal{ALC}$ from the modal logic perspective, atomic concepts simply can be expounded as atomic *propositional* formulae, and can be interpreted as the set of worlds in which the atomic propositional formula holds. In this case $\forall.$ becomes a modal operator since it is applied to formulae. Thus $\neg c_1 \sqcup \forall r.(c_2 \sqcap c_3)$ can be expressed by the propositional modal formula $\neg c_1 \vee \mathbf{K_r}(c_2 \wedge c_3)$. $\mathbf{K_r}(c_2 \wedge c_3)$ is to be read as "agent r *knows* proposition $c_2 \wedge c_3$," and means that in every world accessible for r, both c_2 and c_3 hold. Actually

- the domain of an extension function can be read as a set of *worlds*.
- atomic concepts can be interpreted as the set of worlds in which they hold, if expounded as atomic formulae.
- atomic roles can be interpreted as *accessibility relations*.

Hence $\forall R.C$ can be expounded as "all worlds in which agent R knows proposition C" instead of "all objects for which all R's are in C."

It is not essential for the following to know propositional modal logic except for establishing the correspondence. To begin with, we inductively fix the syntax of $\mathbf{K_{(m)}}$. The atomic propositions $p_1, p_2, ...$ as well as *true* are $\mathbf{K_{(m)}}$-formulae, and, if α and β are $\mathbf{K_{(m)}}$-formulae, so are $\alpha \wedge \beta$, $\neg\alpha$, $\mathbf{K_1}\alpha$,..., and $\mathbf{K_m}\alpha$. The meaning of a $\mathbf{K_{(m)}}$-formula is given by a *Kripke structure* $M = \langle \mathcal{D}, \pi, \mathcal{P}_1, ..., \mathcal{P}_m\rangle$, where $\mathcal{D}$ is any set of worlds, $\mathcal{P}_1, ..., \mathcal{P}_m \subseteq \mathcal{D} \times \mathcal{D}$ are accessibility relations, and π is a truth assignment mapping atomic propositions to subsets of $\mathcal{D}$. $d \in \pi(p)$ says that p holds in the world d, and $d\mathcal{P}_i e$ denotes that in world d agent i considers world e to be possible.

We call $\mathbf{K_{(m)}}$ *normal* because there are no restrictions on the accessibility relations. The denotation of arbitrary formulae is given by a *valuation* v_M mapping $\mathbf{K_{(m)}}$-formulae to subsets of $\mathcal{D}$. v_M is uniquely defined w.r.t. a Kripke structure $M = \langle \mathcal{D}, \pi, \mathcal{P}_1, ..., \mathcal{P}_m \rangle$ by $v_M(true) = \mathcal{D}$, $v_M(p_i) = \pi(p_i)$, $v_M(\alpha \wedge \beta) = v_M(\alpha) \cap v_M(\beta)$, $v_M(\neg\alpha) = \mathcal{D} \setminus v_M(\alpha)$, and $v_M(\mathbf{K_i}\alpha) = \{w \in \mathcal{D} : \forall \langle w, w' \rangle \in \mathcal{P}_i . w' \in v_M(\alpha)\}$. Now a $\mathbf{K_{(m)}}$-formula α is *satisfiable* iff there is a valuation v_M with $v_M(\alpha) \neq \emptyset$, and α is *valid* iff $\neg\alpha$ is not satisfiable. For a slightly more comprehensive introduction to $\mathbf{K_{(m)}}$ confer [Halpern and Moses, 1985].

To establish the correspondence, consider the function f mapping $\mathcal{ALC}$-concepts to $\mathbf{K_{(m)}}$-formulae with $f(c) = c$, $f(\top) = true$, $f(C \sqcap D) = f(C) \wedge f(D)$, $f(\neg C) = \neg f(C)$, and $f(\forall R.C) = \mathbf{K_R} f(C)$. It could easily be shown by induction on the complexity of $\mathcal{ALC}$-concepts that f is a linearly length-bounded isomorphism such that an $\mathcal{ALC}$-concept C is coherent iff the $\mathbf{K_{(m)}}$-formula $f(C)$ is satisfiable. Hence we can conclude:

Theorem 1 *$\mathcal{ALC}$ is a notational variant of the propositional modal logic $\mathbf{K_{(m)}}$, and satisfiability in $\mathbf{K_{(m)}}$ has the same computational complexity as coherence in $\mathcal{ALC}$.*

By this correspondence, several theoretical results for $\mathbf{K_{(m)}}$ can easily be carried over to $\mathcal{ALC}$. We immediately know, for example, that without loss of generality, it suffices to consider either finite $\mathcal{ALC}$-models of exponential size or a single infinite *canonical* $\mathcal{ALC}$-model (cf. e.g. [Hughes and Cresswell, 1984]). We shall expose two other outcomes of the correspondence, namely an axiomatization of $\mathcal{ALC}$ and the complexity of $\mathcal{ALC}$ somewhat closer.

Proposition 1 (An Axiomatization of $\mathcal{ALC}$)
The axioms forcing $\langle \top, \sqcap, \neg \rangle$ to be a Boolean algebra together with

$$\begin{aligned} \forall R.\top &= \top \\ \forall R.(C \sqcap D) &= (\forall R.C) \sqcap (\forall R.D) \end{aligned}$$

are a sound and complete axiomatization of $\mathcal{ALC}$-equivalence [Lemmon, 1966].

Now we shall prove that deciding subsumption in $\mathcal{ALC}$ is PSPACE-complete. Of course, this is not a new result, but its proof is much simpler than the original six-page one in [Schmidt-Schauß and Smolka, 1991]. The reason is that Ladner [1977] proved the PSPACE-completeness of validity in $\mathbf{K_{(m)}}$ for $m = 1$, and, as Halpern and Moses [1985] noted, this result also holds for each $m \geq 1$. Using Theorem 1 and Lemma 1, we can immediately conclude:

Proposition 2 (Complexity of $\mathcal{ALC}$)
Deciding subsumption in $\mathcal{ALC}$ is PSPACE-complete, even if involving only a single atomic role [Ladner, 1977].

An algorithm for deterministically computing coherence in $\mathcal{ALC}$ in quadratic space is also given in [Ladner, 1977].

4 Terminological Logics and Dynamic Logics

Although concepts can be composed in $\mathcal{ALC}$, we are not able to form compound roles. To overcome this deficiency, we shall introduce an extension of $\mathcal{ALC}$, named $\mathcal{TSL}$. Based on the role *parent*, for instance, we can express in $\mathcal{TSL}$ its transitive-reflexive closure by $parent^*$, its inverse role 'has child' by $parent^{-1}$, and the union of the roles *parent* and $parent^{-1}$, for example, by $parent \sqcup parent^{-1}$. Thus in $\mathcal{TSL}$ we could formalize the concept 'human beings' as follows:

$$\begin{array}{ll} & \forall parent^*.(\exists parent.women \sqcap \exists parent.men) \\ \sqcap & \forall (parent \sqcup parent^{-1})^*.((women \sqcap \neg men) \sqcup \\ & \qquad\qquad\qquad\qquad (men \sqcap \neg women)) \end{array}$$

Beside these operations, $\mathcal{TSL}$ comprises the composition of roles $\circ$ and the operation id with $\mathsf{id}(C)$ denoting the identity relation over C. Using both we can express, for instance, the role 'mother' by $parent \circ \mathsf{id}(women)$.

To be exact, $\boldsymbol{\mathcal{TSL}}$ is given by the same concept-formation rule as $\mathcal{ALC}$ together with the role-formation rule

$$R, S \quad \to \quad r \mid R \circ S \mid R \sqcup S \mid R^* \mid R^{-1} \mid \mathsf{id}(C).$$

To specify the formal semantics of $\mathcal{TSL}$, we additionally require an extension function $\mathcal{E}$ to be a mapping with

$$\begin{array}{rcl} \mathcal{E}[R \circ S] & = & \mathcal{E}[R] \circ \mathcal{E}[S] \\ \mathcal{E}[R \sqcup S] & = & \mathcal{E}[R] \cup \mathcal{E}[S] \\ \mathcal{E}[R^*] & = & \mathcal{E}[R]^* \\ \mathcal{E}[R^{-1}] & = & \{\langle d, e\rangle : \langle e, d\rangle \in \mathcal{E}[R]\} \\ \mathcal{E}[\mathsf{id}(C)] & = & \{\langle d, d\rangle : d \in \mathcal{E}[C]\} \end{array}$$

The identity role self, the *range restriction* $|$ defined by $\mathcal{E}[R|C] = \{\langle d, e\rangle \in \mathcal{E}[R] : e \in \mathcal{E}[C]\}$, and the transitive closure of roles $^+$ all can be defined within

$\mathcal{TSL}$:

$$\begin{aligned} \mathsf{self} &\stackrel{\text{def}}{=} \mathsf{id}(\top) \\ R|C &\stackrel{\text{def}}{=} R \circ \mathsf{id}(C) \\ R^{+} &\stackrel{\text{def}}{=} R \circ R^{*} \end{aligned}$$

However, only the first and the second definition are linearly bounded in their length whereas the third generally is not. Remarkably, we could have introduced the range restriction $|$ and the role self into the logic and then define $\mathsf{id}(C)$ as $\mathsf{self}|C$.

Now it is important to realize that roles not only can be interpreted as accessibility relations but also as nondeterministic programs. In this case $\langle d, e\rangle \in \mathcal{E}[R]$ denotes that there is an execution of the program R starting in state d and ending in state e. To achieve this interpretation

- the domain of an extension function is to be read as a set of program *states*.
- atomic concepts are to be interpreted as the set of states in which they hold, if expounded as atomic formulae.
- atomic roles are to be interpreted as *atomic nondeterministic programs*.

Using this interpretation, compound $\mathcal{TSL}$-expressions can be expounded as follows:

- $\forall R.C$ as "whenever program R terminates, proposition C holds on termination"
- $R_1 \circ R_2$ as "run R_1 and R_2 consecutively"
- $R_1 \sqcup R_2$ as "nondeterministically do R_1 or R_2"
- R^{*} as "repeat program R a nondeterministically chosen number of times ≥ 0"
- R^{-1} as "run R in reverse"
- $\mathsf{id}(C)$ as "proceed without changing the program state iff proposition C holds"

This illustrates that $\mathcal{TSL}$ corresponds to propositional dynamic logic. Indeed, $\mathcal{TSL}$ could easily be shown to be a notational variant of *converse*-PDL as given in [Fischer and Ladner, 1979]. There, $\top$ is written as *true*, $C \sqcap D$ as $C \wedge D$, $\neg C$ as $\sim C$, $\forall R.C$ as $[R]C$, $\exists R.C$ as $\langle R\rangle C$, $R \circ S$ as $R; S$, $R \sqcup S$ as $R \cup S$, R^{-1} as R^{-}, and $\mathsf{id}(C)$ as $C?$.

Theorem 2 *$\mathcal{TSL}$ is a notational variant of converse-PDL, the propositional dynamic logic PDL with the converse-operator. Moreover, satisfiability in converse-PDL has the same computational complexity as coherence in $\mathcal{TSL}$.*

For an excellent introduction to dynamic logic you may confer [Harel, 1984]. There, one can find a survey of interesting theorems of dynamic logic which can now be expounded as theorems of $\mathcal{TSL}$. Hence, for instance, we can narrow down the computational complexity of $\mathcal{TSL}$. Fischer and Ladner [1979] showed that PDL-validity is EXPTIME-hard, even if involving only a single atomic program and its transitive-reflexive closure as programs. Using Theorem 2, we can infer:

Proposition 3 (Lower Compl. Bound of $\mathcal{TSL}$)
Deciding subsumption in $\mathcal{ALC}$ extended with the transitive-reflexive closure of roles is EXPTIME-hard, even if involving only a single atomic role [Fischer and Ladner, 1979].

Concerning the upper complexity bound, Pratt [1979] gave an algorithm for deciding PDL-satisfiability requiring at most exponential time. As Harel [1984, Section 2.5.6] pointed out, the algorithm can easily be extended to deal with $^{-1}$. The reason is that without loss of generality, we can assume $^{-1}$ applied only to atomic roles. To see this, realize that for each $\mathcal{E}$, $\mathcal{E}[(R \circ S)^{-1}] = \mathcal{E}[S^{-1} \circ R^{-1}]$, $\mathcal{E}[(R \sqcup S)^{-1}] = \mathcal{E}[R^{-1} \sqcup S^{-1}]$, and $\mathcal{E}[(R^*)^{-1}] = \mathcal{E}[(R^{-1})^*]$. Using Theorem 2, we know that $\mathcal{TSL}$-coherence is contained in EXPTIME, and therefore $\mathcal{TSL}$-subsumption is in co-EXPTIME. However, co-EXPTIME is the same as EXPTIME. So we have:

Proposition 4 (Upper Compl. Bound of $\mathcal{TSL}$)
Subsumption in $\mathcal{TSL}$ can be decided in exponential time [Pratt, 1979, Harel, 1984, Section 2.2/2.5.6].

As another by-product of the correspondence we gain an axiomatization of $\mathcal{TSL}$ which assumes $^{-1}$ to be applied only to atomic roles.

Proposition 5 (An Axiomatization of $\mathcal{TSL}$)
Let $C \sqsubseteq D$ be defined as $C \sqcap D = C$. The axioms for $\mathcal{ALC}$ together with

$$\begin{aligned}
\forall R \sqcup S.C &= (\forall R.C) \sqcap (\forall S.C) \\
\forall R \circ S.C &= \forall R.\forall S.C \\
\forall \mathsf{id}(C).D &= \neg C \sqcup D \\
\forall R^*.C &= C \sqcap \forall R^+.C \\
C \sqcap \forall R^*.(\neg C \sqcup \forall R.C) &\sqsubseteq \forall R^*.C \\
C &\sqsubseteq \forall r.\exists r^{-1}.C \\
C &\sqsubseteq \forall r^{-1}.\exists r.C
\end{aligned}$$

are a sound and complete axiomatization of $\mathcal{TSL}$-equivalence [Pratt, 1979, Harel, 1984, Section 2.5.6].

By the way, the first three axioms indicate that $\mathcal{TSL}$ without * and $^{-1}$ is reduced to $\mathcal{ALC}$ since $\sqcup$, $\circ$, id can be eliminated linearly.

The correspondence theorem additionally provides us with an elaborated model theory. The main model theorem for *converse*-PDL says that it suffices to consider only finite connected models of exponential size. To carry over this result, we call an extension function $\mathcal{E}$ over $\mathcal{D}$ mapping the atomic roles $r_1, ..., r_m$ *connected* iff for every $d, e \in \mathcal{D}$, $\langle d, e\rangle \in \mathcal{E}[(r_1 \sqcup r_1^{-1} ... \sqcup r_m \sqcup r_m^{-1})^*]$. Furthermore, $\boldsymbol{l(C)}$ denotes the length of the concept C regarded as a string over $\top, \sqcap, \sqcup, \neg, \forall., \circ, ^*, ^{-1}$, and atoms.

Proposition 6 (Finite $\mathcal{TSL}$-Models)
Every coherent $\mathcal{TSL}$-concept C has a connected model over $\mathcal{D}$ with $|\mathcal{D}| \leq 2^{l(C)}$ [Fischer and Ladner, 1979].

In contrast to that, the finite model property does not hold for $\mathcal{TSL}$ augmented with the intersection of roles, called $\boldsymbol{\mathcal{TSLR}}$. The reason is that every model $\mathcal{E}$ for

$$\forall r^*.((\exists r.\top) \sqcap \forall(r^+ \sqcap \mathsf{self}).\neg\top)$$

has an infinite acyclic $\mathcal{E}[r]$-chain.

Proposition 7 (Infinite $\mathcal{TSLR}$-Models)
There is a coherent $\mathcal{TSLR}$-concept which has no finite model [Harel, 1984, Theorem 2.35].

Note that this does not mean that $\mathcal{TSLR}$ is undecidable. Actually, the decidability of $\mathcal{TSLR}$ seems to be unknown. It is known, however, that $\mathcal{TSL}$ extended with the complementation of roles is undecidable [Harel, 1984, Theorem 2.34], and that $\mathcal{TSLR}$ with features is highly undecidable as we shall see in the next section.

4.1 $\mathcal{TSL}$ with Features

In $\mathcal{TSL}$ we are able to force, for instance, that something has at least two parents, namely a female and a male one:

$$(\exists parent|women.\neg men) \sqcap (\exists parent|men.\neg women)$$

Unfortunately, this expression does not stipulate that something has exactly two parents. The reason is that we have expressed 'has mother' as the role $parent|women$. However, 'has mother' rather is a partial function than a relation. If something has a mother, it has exactly one. This suggests to extend $\mathcal{TSL}$ with *features*, denoting atomic partial functions. If *mother* and *father* were features, the above expression indeed would force that each human being has exactly one mother and father.

We define $\mathcal{FSL}$ by the same formation rules as $\mathcal{TSL}$ exept that an $\mathcal{FSL}$-role additionally can be a *feature* symbol. Moreover, we require an extension function $\mathcal{E}$ over $\mathcal{D}$ to be a mapping such that for each feature symbol f, $\mathcal{E}[f]$ is a partial function mapping $\mathcal{D}$ to $\mathcal{D}$.

Note that $f_1 \circ f_2$ and $\mathsf{id}(C)$ denote partial functions, whereas $f_1 \sqcup f_2$, f^*, and f^{-1} generally denote binary relations.

Clearly, features correspond to *atomic deterministic programs* considered in dynamic logic. Thus [Parikh, 1979, §7] can be read to show that any atomic role r is expressible by $f_r \circ (f_{new})^*$, where f_r is the feature uniquely corresponding to r and f_{new} is a new feature. In this manner each non-feature atomic role in a $\mathcal{TSL}$-expression can be eliminated without increasing its length more than linearly.[3] Thus we can assume that the only atomic roles which $\mathcal{FSL}$ comprises are features. So, it is obvious that $\mathcal{FSL}$ is a notational variant of the deterministic version of *converse*-PDL.

Theorem 3 *$\mathcal{FSL}$ is a notational variant of converse-DPDL, the deterministic propositional dynamic logic DPDL with the converse-operator. Moreover, satisfiability in converse-DPDL has—up to linear time—the same computational complexity as coherence in $\mathcal{FSL}$.*

Ben-Ari *et al.* [1982] showed that DPDL-satisfiability is contained in EXPTIME, and Vardi [1985] pointed out that satisfiability in *converse*-DPDL can be decided in double exponential time. Utilizing Theorem 3 and Lemma 1, we can conclude:

Proposition 8 (Complexity of $\mathcal{FSL}$)
Subsumption in $\mathcal{FSL}$ without $^{-1}$ can be decided in exponential time [Ben-Ari et al.*, 1982], and deciding subsumption in $\mathcal{FSL}$ can be done in double exponential time [Vardi, 1985]. Furthermore, both problems are EXPTIME-hard [Parikh, 1979, §7].*

We now present a sound and complete algorithm for deciding coherence of $^{-1}$-free $\mathcal{FSL}$-concepts in deterministic exponential time as described in [Ben-Ari *et al.*, 1982]. To test the coherence of a concept C, the algorithm builds a tree with nodes labeled by sets of $\mathcal{FSL}$-concepts with root $\{C\}$, where double negations occurring in concepts are assumed to be eliminated.

[3] By the way this means that in the presence of * deciding subsumption in feature logics is at least as hard as deciding subsumption in the corresponding terminological logic.

The algorithm uses three functions to build the tree, namely $\wedge$-*succ*, $\vee$-*succ*, and f-*succ*. $\wedge$-*succ* and $\vee$-*succ* are nondeterministic functions to generate necessary and possible consequences of a set of $\mathcal{FSL}$-concepts represented by an and-or-tree.

$$\wedge\text{-}succ(\Gamma) \stackrel{\text{def}}{=} \Gamma \cup \begin{cases} \{C, D\} & \textit{if } C \sqcap D \in \Gamma \\ \{C, D\} & \textit{if } \exists \mathsf{id}(C).D \in \Gamma \\ \{\exists R.\exists S.C\} & \textit{if } \exists R \circ S.C \in \Gamma \\ \{\forall R.\forall S.C\} & \textit{if } \forall R \circ S.C \in \Gamma \\ \{\forall R.C, \forall S.C\} & \textit{if } \forall R \sqcup S.C \in \Gamma \\ \{C, \forall R.\forall R^*.C\} & \textit{if } \forall R^*.C \in \Gamma \end{cases}$$

$\vee\text{-}succ(\Gamma) \stackrel{\text{def}}{=}$

$$\begin{cases} \langle \Gamma \cup \{C\}, \Gamma \cup \{D\} \rangle & \textit{if } C \sqcup D \in \Gamma \\ \langle \Gamma \cup \{\neg C\}, \Gamma \cup \{D\} \rangle & \textit{if } \forall \mathsf{id}(C).D \in \Gamma \\ \langle \Gamma \cup \{\exists R.C\}, \Gamma \cup \{\exists S.C\} \rangle & \textit{if } \exists R \sqcup S.C \in \Gamma \\ \langle \Gamma \cup \{C\}, \Gamma \cup \{\exists R.\exists R^*.C\} \rangle & \textit{if } \exists R^*.C \in \Gamma \end{cases}$$

f-*succ* is a function to generate the f-successors of a node, defined for each feature f by

$$f\text{-}succ(\Gamma) \stackrel{\text{def}}{=} \{C : \exists f.C \in \Gamma\} \cup \{C : \forall f.C \in \Gamma\}.$$

To compute the coherence of an $^{-1}$-free $\mathcal{FSL}$-concept, the algorithm performs the following steps:

1. Replace each occurrence of a non-feature atomic role r in C by $f_r \circ (f_{new})^*$, where f_r is the feature uniquely corresponding to r and f_{new} is a new feature.

2. Take the tree consisting solely of the root $\{C\}$ as the current tree.

3. Apply $\wedge$-*succ* and $\vee$-*succ* to the not yet eliminated leaves until every node thus obtainable already is on the tree.

4. Repeatedly eliminate each leaf Γ violating either (a) $S \subseteq \Gamma$ for every $S = \wedge\text{-}succ(\Gamma)$, (b) $S_1 \subseteq \Gamma$ or $S_2 \subseteq \Gamma$ for every $\langle S_1, S_2 \rangle = \vee\text{-}succ(\Gamma)$, or (c) $C \in \Gamma$ iff $\neg C \notin \Gamma$ for each C. Call the set of nodes of the current tree N.

5. For each feature f, apply f-*succ* to all not yet eliminated leaves where two nodes $f\text{-}succ(\Gamma)$ and $f'\text{-}succ(\Gamma)$ are to be identified if they are equal. Now define for each feature f, $\mathcal{E}[f] \subseteq N \times N$ such that $\langle \Gamma_s, \Gamma_t \rangle \in \mathcal{E}[f]$

iff Γ_t is a node of the tree with root f-$succ(\Gamma_s)$ generated by $\wedge$-$succ$ and $\vee$-$succ$. For each node $\Gamma_s \in N$ check (in polynomial time) whether for every $\exists R.D \in \Gamma_s$ there is a Γ_t such that $\langle \Gamma_s, \Gamma_t \rangle \in \mathcal{E}[R]$ and $D \in \Gamma_t$. Eliminate each node not satisfying this condition and all edges leading from and to that node. Continue with 3. until all leaves do satisfy the condition.

6. Return 'yes' if C is an element of a not eliminated node; otherwise return 'no.'

To test $\exists f^*.c$, for instance, the algorithm first generates the root $\Gamma_0 = \{\exists f^*.c\}$ and its $\vee$-successors $\Gamma_1 = \{\exists f^*.c, c\}$ and $\Gamma_2 = \{\exists f^*.c, \exists f.\exists f^*.c\}$. Then the f-successor of Γ_2, $\Gamma_3 = \{\exists f^*.c\}$, is generated. $\mathcal{E}[f]$ is defined as $\{\langle \Gamma_2, \Gamma_0 \rangle, \langle \Gamma_2, \Gamma_1 \rangle, \langle \Gamma_2, \Gamma_2 \rangle\}$ because Γ_1 as well as Γ_2 are $\vee$-successors of f-$succ(\Gamma_2)$. Now for each $\exists R.C \in \Gamma_2$ there is a node Γ such that $\langle \Gamma_2, \Gamma \rangle \in \mathcal{E}[R]$ and $C \in \Gamma$. Therefore no leaf has to be eliminated. The algorithm halts returning 'yes' because each $\vee$-successor of Γ_3 is already on the tree and $\exists f^*.c \in \Gamma_0$.

Surprisingly, allowing the intersection of roles does not preserve the decidability of $\mathcal{FSL}$.

Proposition 9 (Undecidability of $\mathcal{FSLR}$)
Subsumption in $\mathcal{FSL}$ augmented with role intersection is highly undecidable, even if involving only features as atomic roles [Harel, 1984, Theorem 2.36].

With Theorem 3 we gain new model theoretic insights into $\mathcal{FSL}$. For the ability to force an $\mathcal{FSL}$-model to be infinite, both features and $^{-1}$ are essential. To see this, first realize that each model $\mathcal{E}$ for

$$c \sqcap \forall (f^{-1})^*.\exists f^{-1}.\neg c$$

has an infinite acyclic $\mathcal{E}[f]$-chain if f is a feature. Requiring f to be a feature is necessary to preclude cyclic models such as $\mathcal{E}$ over $\mathcal{D} = \{d, e\}$ with $\mathcal{E}[c] = \{e\}$ and $\mathcal{E}[f] = \{\langle d, e \rangle, \langle d, d \rangle\}$, where $\mathcal{E}[f]$ is not functional.

Proposition 10 (Infinite $\mathcal{FSL}$-Models)
There is a coherent $\mathcal{FSL}$-concept which has no finite model [Vardi, 1985].

In contrast to this result, the finite model property holds for $^{-1}$-free $\mathcal{FSL}$.

Proposition 11 (Finite Models for $^{-1}$-free $\mathcal{FSL}$)
Every coherent $^{-1}$-free $\mathcal{FSL}$-concept C has a connected model over $\mathcal{D}$ with $|\mathcal{D}| \leq 4^n n^2$, where $n = l(C)$ [Ben-Ari et al.*, 1982].*

4.2 Concept Equations and Inequations

The terminological logics investigated seem to be very expressive. Nevertheless, they do not support concept definitions and constraints. We could want to define, for instance, human beings by

$$human \quad = \quad \forall parent.human \tag{1}$$

Of course, we want to know the consequences of such definitions and constraints. Like in dynamic logic we say, for example, that $human \sqsubseteq \forall parent^*.human$ is a *global consequence* of the set $\Gamma = \{(1)\}$ and write:

$$\Gamma \models human \sqsubseteq \forall parent^*.human$$

Formally, we pin this down as follows: Let $\Gamma \cup \{E\}$ be a set of concept equations and inequations. E is a *global consequence* of Γ, written $\boldsymbol{\Gamma} \models \boldsymbol{E}$, iff each extension function $\mathcal{E}$ satisfying all elements of Γ, also satisfies E; where $\mathcal{E}$ *satisfies* $C = D$ iff $\mathcal{E}[C] = \mathcal{E}[D]$, and $\mathcal{E}$ *satisfies* $C \neq D$ iff $\mathcal{E}[C] \neq \mathcal{E}[D]$.

We treat $C \sqsubseteq D$ as an abbreviation for $C \sqcap D = C$, and each element of Γ of this form is called *universal implication.*

Now there is the question whether the ability to define and constrain sets actually increases the expressive power of $\mathcal{TSL}$. It will turn out that for finite Γ, global consequence in $\mathcal{TSL}$ is reducible to subsumption in $\mathcal{TSL}$. To show this, we need two lemmata. The first signifies that only connected models are relevant for global consequence in $\mathcal{TSL}$. This can easily be proved by utilizing the so-called *collapsed model property* of $\mathcal{TSL}$ stating that every $\mathcal{TSL}$-model can be collapsed into a finite connected one [Harel, 1984, Theorem 2.13].

Lemma 2 *Let $\Gamma \cup \{E\}$ be a set of $\mathcal{TSL}$-concept equations and inequations. Then $\Gamma \models E$ iff each connected extension function $\mathcal{E}$ satisfying all elements of Γ, also satisfies E.*

The following lemma states that universally quantified expressions are expressible in connected models—viz. by $\boldsymbol{c(C)}$ defined as $\forall(r_1 \sqcup r_1^{-1} \ldots \sqcup r_m \sqcup r_m^{-1})^*.C$, where $r_1, \ldots, r_m$ are the atomic roles and features of the language in question. Note that c owes its name to the common knowledge operator (cf. [Halpern and Moses, 1985]).

Lemma 3 *Let C be any $\mathcal{TSL}$-concept and $\mathcal{E}$ any connected extension function over $\mathcal{D}$. $\mathcal{E}[C] = \mathcal{D}$ iff $\mathcal{E}[c(C)] = \mathcal{D}$, and moreover $\mathcal{E}[c(C)] \neq \mathcal{D}$ iff $\mathcal{E}[c(C)] = \emptyset$.*

Theorem 4 *For finite sets of concept equations and inequations, global consequence in $\mathcal{TSL}$ is log space reducible to subsumption in $\mathcal{TSL}$.*

Proof: Let $eq(C, D)$ be $c(\neg C \sqcup D) \sqcap c(\neg D \sqcup C)$. According to the last two lemmata, it is obvious that $\{C_1 = D_1\} \cup \Gamma \models C \sqsubseteq D$ iff $\Gamma \models eq(C_1, D_1) \sqcap C \sqsubseteq D$ and that $\{C_1 \neq D_1\} \cup \Gamma \models C \sqsubseteq D$ iff $\Gamma \models \neg eq(C_1, D_1) \sqcap C \sqsubseteq D$. Taking this as induction step, it can be easily shown by induction on $|\Gamma|$ that all elements of Γ can be eliminated linearly. $\square$

However, the proof does not work for infinite Γ since $\mathcal{TSL}$ lacks compactness. This becomes obvious when considering the infinite set $\Gamma = \{c \sqsubseteq \forall r.d, c \sqsubseteq \forall r \circ r.d, ...\}$. Clearly, $c \sqsubseteq \forall r^+.d$ is a global consequence of Γ but not a global consequence of any finite subset of Γ. In fact, global consequence in $\mathcal{TSL}$ is known to be highly undecidable [Harel, 1984, Section 2.4]. However, Theorem 4 also holds for $\mathcal{FSL}$ since Vardi [1985] proved the collapsed model property for $\mathcal{FSL}$.

5 Conclusions

So far we have seen that correspondences between terminological logics and propositional modal and dynamic logics can be used to gain new insights into the nature of terminological logics. However, this work can be extended in two ways. First, we can further exploit the correspondences already established by carefully studying the corresponding theories of modal and dynamic logic. For example, we proved that a syntactically restricted form of global consequence in $\mathcal{TSL}$, known in dynamic logic as *partial completeness assertions*, is NP-complete. Secondly, we can establish further correspondences. *Constants* in terminological logics, for instance, correspond to *names* (atomic formulae denoting single element sets) in dynamic logic. Similarly, temporal expressions can be integrated into terminological logics.

Acknowledgements

This work was supported by the Commission of the European Communities and is part of the ESPRIT Project 5210.

References

[Ben-Ari *et al.*, 1982] Mordechai Ben-Ari, Joseph Y. Halpern, and Amir Pnueli. Deterministic Propositional Dynamic Logic: Finite Models, Complexity, and Completeness. *Journal of Computer and System Science*, 25:402–417, 1982.

[Fischer and Ladner, 1979] Michael J. Fischer and Richard E. Ladner. Propositional Dynamic Logic of Regular Programs. *Journal of Computer and System Science*, 18:194–211, 1979.

[Halpern and Moses, 1985] Joseph Y. Halpern and Yoram Moses. A Guide to the Modal Logics of Knowledge and Belief. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence*, pages 480–490, Los Angeles, Cal., 1985.

[Harel, 1984] David Harel. Dynamic logic. In D. Gabbay and F. Guenther, editors, *Handbook of Philosophical Logic*, volume 2, pages 497–604. Reidel, Dordrecht, Holland, 1984.

[Hughes and Cresswell, 1984] George E. Hughes and M. J. Cresswell. *A Companion to Modal Logic*. Methuen, London, 1984.

[Ladner, 1977] Richard E. Ladner. The Computational Complexity of Provability in Systems of Modal Propositional Logic. *SIAM Journal of Computing*, 6(3):467–480, 1977.

[Lemmon, 1966] E. J. Lemmon. Algebraic Semantics for Modal Logic I. *Journal of Symbolic Logic*, 31(1):46–65, 1966.

[Nebel, 1990] Bernhard Nebel. *Reasoning and Revision in Hybrid Representation Systems*. Lecture Notes in Artificial Intelligence. Springer-Verlag, Berlin, West Germany, 1990.

[Parikh, 1979] Rohit Parikh. Propositional Dynamic Logics of Programms: A Survey. In E. Engeler, editor, *Proceedings of the Workshop on Logic of Programs*, volume 125 of *Lecture Notes in Computer Science*, pages 102–144, Berlin, West Germany, 1979. Springer-Verlag.

[Pratt, 1979] Vaughan R. Pratt. Models of Program Logics. In *Proceedings of the 20th Annual Symposium on Foundations of Computer Science*, pages 115–122, San Juan, Puerto Rico, 1979.

[Schmidt-Schauß and Smolka, 1991] Manfred Schmidt-Schauß and Gert Smolka. Attributive Concept Descriptions with Complements. *Artificial Intelligence*, 48(1):1–26, 1991. A preliminary version of this paper is available as IBM Germany Scientific Center, IWBS, Stuttgart, West Germany, 1989.

[Vardi, 1985] Moshe Y. Vardi. The Taming of Converse: Reasoning about Two-Way Computations. In R. Parikh, editor, *Proceedings of the Workshop on Logic of Programs*, volume 193 of *Lecture Notes in Computer Science*, pages 413–424, Berlin, West Germany, 1985. Springer-Verlag.