Modal and Higher Order Logical Reasoning with SUMO

Adam Pease[1]

[1]Naval Postgraduate School, Monterey, CA, adam.pease@nps.edu

1 Introduction

Inference with modal and higher order logics has a long history of foundational work[7, 6, 8], but the practical illustrations of the mathematics has most often been limited to proofs of correctness or complexity[10], exploring the relationships among various axioms, or relatively small problems where all or most of the axioms are relevant to the particular problem being solved [19]. More recently work has been done in embedding several different modal operators in a single higher order logic framework, expanding the scope of possible inferences[3].

If we wish for work in these expressive logics to have applicability to general purpose question answering, in the context of explainable AI, then much larger and more comprehensive axiom sets are needed, where necessarily few of those axioms are employed for answering any given question. Such an exercise should also elucidate how different modalities can be used in common sense question answering, and how the interactions among these modalities and first order logical expressions characterising our world govern which axioms to include in a single large theory.

For these reasons we have been exploring how to translate the axioms in the Suggested Upper Merged Ontology (SUMO)[13, 15] into several different logics that are implemented with different automated theorem provers.

Several discussions with Richard Crouch over the years, especially with regards to the possibility of embedding modalities in first order, and motivated by representations of the semantics of natural language, have formed the germ of this effort, and we are honored to celebrate Dr. Crouch's achievements with our addition to this volume.

2 Background

The overall goal of the SUMO effort is to capture as much knowledge as possible, in as formal and computable a language as possible. Over the course of the long-term effort since its origin in the year 2000, it was expected that there would be advances in logic and in automated theorem proving, and that the collection of knowledge should not be limited by the current state of the art. At the start of the effort, the semantics of first order logic were well-known and studied. While it was possible at that time to perform automated inference in first order logic, the implementation of efficient ATP with equality, enabled in part by the superposition calculus[2] was still new. It would still be some years before the ATP community reached a standard for a classical higher order language [5]. While it might appear risky to define axioms without a fully specified semantics, it has proven

possible to refine and correct the body of knowledge as better support for more expressive logics has come about[20, 17]. Had we limited the knowledge collected to what could be subjected to efficient inference at the time, much useful knowledge would have been missed.

The more expressive the logic used, the greater opportunity to create general axioms which have more inference power or what has been called a greater inferential closure. While the number of axioms in SUMO beyond first order is small compared to the largest category of ground binary relationships, they are considerably more productive when part of a chain of inference. At the time of this writing there are the following statistics for SUMO:

Total Terms	Total Axioms	Total Rules
16129	227412	6955

Relations: 1713

Ground tuples:	220371
of which are binary:	150609
of which are $arity > 2$:	69848

Rules:	6955
of which are horn:	2311
first-order:	5103
temporal:	789
modal:	257
epistemic:	87
other higher-order:	827

There are a few dozen relations in SUMO that take formulas as arguments[1]:

- KappaFn
- ProbabilityFn
- attitudeForFormula
- believes
- causesProposition
- conditionalProbability
- confersNorm
- confersObligation
- confersRight
- considers
- containsFormula
- decreasesLikelihood
- deprivesNorm
- describes
- desires
- disapproves
- doubts
- entails
- expects
- hasPurpose
- hasPurposeForAgent
- holdsDuring
- holdsObligation
- holdsRight
- increasesLikelihood
- independentProbability
- knows
- modalAttribute
- permits
- prefers
- prohibits
- rateDetail
- treatedPageDefinition
- visitorParameter

[1]The full axiomatization of each can been seen by entering the case-sensistive term in the "KB term" field at `https://sigma.ontologyportal.org:8443/sigma/Browse.jsp?kb=SUMO&lang=EnglishLanguage`

Over the years, we have created translators[25, 16, 4][2] from SUMO's authoring language of SUO-KIF[14] to languages in the TPTP family of languages[23], as languages of increasing expressiveness have become available and supported in Automated Theorem Provers. The first translation effort was to the TPTP First Order Form (FOF) language. The first step was simply to convert the syntax of the two languages, as SUO-KIF uses a prefix notation for logic, built on LISP S-expression syntax and TPTP uses a combination of prefix and infix notation adhering to Prolog syntax. Since initial uppercase identifiers in Prolog syntax are interpreted as variables, we add a lowercase prefix to all constant terms and an uppercase prefix to all variables.

For a very simple axiom in SUMO like

```
(=>
  (and
    (connected ?X ?Y)
    (part ?Y ?Z))
  (connected ?X ?Z))
```

it becomes in TPTP FOF

```
fof(kb1,axiom,(![V__X,V__Y,V__Z]: (
  (s__connected(V__X,V__Y) & s__part(V__Y,V__Z)) =>
  (s__connected(V__X,V__Z))))).
```

SUO-KIF has only the basic logical operators as primitives (`and or not => <=> forall exists equal`). All other constant terms must be axiomatized in SUO-KIF to state their intended semantics explicitly. This causes an issue in that TPTP FOF requires that constant term and relation symbols are disjoint but SUMO refers to relation symbols as arguments in defining their types. Each relation symbol is given a suffix in translation when used as an argument to distinguish it from the same relation symbol, when used as a relation. For example we state that a wheel is a part of a particular car `s__part(s__MyWheel,s__MyCar)` but if we want to say that the `part` relation takes an instances of type `Object` as its first argument, we append a `__m` suffix as in `s__domain(s__part__m,1,s__Object)`.

SUO-KIF has existential and universal quantifiers but for convenience, any variable that is not quantified is assumed to be universally quantified.

All SUMO relations have a type signature. In order to enforce that signature, the translation involves adding type guards to formulas. During translation, all the type restrictions on variables are collected, on the basis of in which relation they appear. If multiple restrictions apply, then the most restrictive type wins out.

The `part` and `connected` relations both take two instance of the class `Object` as their arguments, so our example becomes

```
fof(kb1,axiom,(![V__X,V__Y,V__Z]: (
  (s__instance(V__X,s__Object) &
   s__instance(V__Y,s__Object) &
   s__instance(V__Z,s__Object)) =>
    ((s__connected(V__X,V__Y) & s__part(V__Y,V__Z)) =>
      (s__connected(V__X,V__Z)))))).
```

[2]Implemented as part of the Sigma Knowledge Engineering Environment and found at `https://github.com/ontologyportal/sigmakee/tree/master/src/java/com/articulate/sigma/trans`

An additional issue is that SUMO makes use of relation variables - variables in the relation position. While this might indicate use of a logic beyond first order, it can be given a restricted first order interpretation, which is what we do. The first order interpretation involves treating formulas with a relation variable as an axiom schema, where we create a new axiom for each existing relation SUMO that can be substituted for a relation variable.

```
(=>
  (instance ?REL TransitiveRelation)
  (=>
    (and
      (?REL ?INST1 ?INST2)
      (?REL ?INST2 ?INST3))
    (?REL ?INST1 ?INST3)))
```

becomes

```
fof(kb2,axiom(![V__INST1,V__INST2,V__INST3]:(
  (s__instance(V__INST1,s__Object) &
   s__instance(V__INST2,s__Object) &
   s__instance(V__INST3,s__Object)) =>
    (s__instance(s__part,s__TransitiveRelation) =>
        ((s__part(V__INST1,V__INST2) &
          s__part(V__INST2,V__INST3)) =>
            s__part(V__INST1,v__INST3))))
```

This axiom is also repeated, with the appropriate type guards, for all other TransitiveRelations, such as subclass, larger, sister etc.

An additional language feature in SUO-KIF is that of the VariableArityRelation. We have a similar treatment to relation variables in that axioms with a variable arity relation can be treated as a schema, and expanded automatically. This is a restricted interpretation, but which has been an acceptable limitation in practice, at least in our work so far. Each formula containing a variable arity relation becomes a set of formulas, expanding the arities up to an arbitrary limit of 7 arguments and again employing type guards. If there is more than one row variable then there will be 7^N copies of the axiom, where N is the number of row variables.

```
(=>
  (partition @ROW)
  (and
    (exhaustiveDecomposition @ROW)
    (disjointDecomposition @ROW)))
```

Partitions of classes are exhaustive and disjoint. A partition can have a variable number of arguments.

```
fof(kb3,axiom(![V__ROW1,V__ROW2,V__ROW3]:(
  (s__instance(V__ROW1,s__Class) &
   s__instance(V__ROW2,s__Class) &
```

```
   s__instance(V__ROW3,s__Class)) =>
    (s__partition(V__ROW1,V__ROW2,V__ROW3) =>
      (s__exhaustiveDecomposition(V__ROW1,V__ROW2,V__ROW3)
      (s__disjointDecomposition(V__ROW1,V__ROW2,V__ROW3)))))))

fof(kb4,axiom(![V__ROW1,V__ROW2,V__ROW3]:(
  (s__instance(V__ROW1,s__Class) &
   s__instance(V__ROW2,s__Class) &
   s__instance(V__ROW3,s__Class) &
   s__instance(V__ROW4,s__Class)) =>
    (s__partition(V__ROW1,V__ROW2,V__ROW3,V__ROW4) =>
      (s__exhaustiveDecomposition(V__ROW1,V__ROW2,V__ROW3,V__ROW4)
      (s__disjointDecomposition(V__ROW1,V__ROW2,V__ROW3,V__ROW4)))))))
```

etc.

The next translation effort for SUMO was an initial attempt at classical typed higher order logic, with the TPTP THF language[4]. While this work was a largely "syntactic" translation that didn't handle modalities, it provided a foundation for our present work.

We also implemented a translation[16] to Typed First order form with Arithmetic (TFA)[24]. TFA requires that we state the types of variables in the syntax of the language, rather than just as additional literals that constrain the meaning of a formula, as in SUO-KIF. There are only a few types - numbers, booleans (notated in TFA as type "$o") and everything else (type $i). Numbers are further divided into rationals ($rat), reals ($real) and integers ($int). This causes some complexity in the SUMO/SUO-KIF translator since SUMO has subclasses of integers such as `PositiveInteger`. THF language and its implementations in E and Vampire do not currently allow us to combine numbers and arithmetic functions with THF.

3 Modal Logic Axioms

Modal logic was originally developed[6] to handle the notions of necessity and possibility (called *alethic* operators), expanding first order logic with these two new operators that condition first order logical formula in a way that would conform to natural language intuitions but using formal logic. The same framework was later expanded to addresses other modalities including knowledge, belief, and temporal relationships.

There have come to be a set of modal axioms that researchers have argued apply to some or all modalities. In this section we examine the modal axioms developed for modalities in SUMO prior to undertaking a systematic analysis. We also discuss development of a set of problems to test whether particular axioms should hold for different modalities.

Necessity is normally given the symbol $\Box$ and possibility the symbol $\Diamond$. They are unary operators that hold over a formula. So "John necessarily like sushi." would be $\Box likes(john, sushi)$. The necessity operator states that something must be true in all possible worlds. The possibility operator holds true over only some worlds. Necessity and possibility operate identically to quantifiers ($\forall$ and $\exists$, respectively) over possible worlds. They have the same relationship as our usual first order quantifiers -

$$\Diamond\phi =_{def} \neg\Box\neg\phi \tag{1}$$

just as $\exists\phi =_{def} \neg\forall\neg\phi$. To avoid special symbology, like the TPTP family of modal logics, we use ASCII symbols. A set of modal axioms (termed 'N', 'D', 'K', 'T', 'B', '4', '5' etc.) have been developed and different sets may be assumed for particular work in modal logic

A paradox such as the Gentle Murderer, or Forrester's paradox [22] depends upon certain standard modal axioms from a logic of possibility and necessity also being true in a logic of obligation. In order to provide flexibility for different axiomatizations for different modalities, we use a general modal relation in SUMO termed `modalAttribute`. For each modal attribute related to a formula, we can provide a different set of modal axioms. For example, to avoid the gentle murderer paradox, we might eliminate axiom 'N' for the attribute of `Obligation`.

Due to the fact that there are several modalities, and we claim that the axiomatizations of those modalities are different, we introduce a parameterized modal operator M for our notation in this paper. M has subscripts for each operator, and later we will see that is has additional subscripts for its parameters, such as the agent for which the particular modality holds. Many of the standard modal axioms are already in SUMO, for example, axiom 'D'

```
(=>
  (modalAttribute ?FORMULA Necessity)
  (modalAttribute ?FORMULA Possibility))
```

which we may notate more formally as $M_nP \rightarrow M_pP$, meaning that the modal operator of necessity over a formula implies that the formula is also possible. Axiom 'N', the necessitation rule, is a property of the logical system, that anything true is necessarily true: $(\vDash p) \Rightarrow (\vDash \Box p))$.

We have 'K', the distribution axiom. This with 'N' is the most basic modal logic.

$$(M_n(P \rightarrow Q)) \rightarrow (M_nP \rightarrow M_nQ) \tag{2}$$

'T' - reflexivity

$$(M_nP) \rightarrow P \tag{3}$$

axiom '4'

$$(M_nP) \rightarrow (M_n(M_nP)) \tag{4}$$

axiom 'B'

$$P \rightarrow (M_n(M_pP)) \tag{5}$$

axiom 'D'

$$(M_nP) \rightarrow (M_pP) \tag{6}$$

Note that axiom '5' $\Diamond p \rightarrow \Box\Diamond p$ should follow from 'N' and therefore is not in SUMO.

3.1 Epistemic Operators

To have a comprehensive theory of world knowledge, we need to include operators for the knowledge and beliefs of agents, so this is part of SUMO. Beliefs should intuitively apply on the basic logical operators of conjunction and disjunction. We augment our notation here with a subscript for the agent, so $M_{ba}P_1$ signifies that an agent a believes formula P_1.

$$(M_{ba}P_1 \wedge M_{ba}P_2) \rightarrow (M_{ba}(P_1 \wedge P_2)) \tag{7}$$

$$(M_{ba}P_1 \vee M_{ba}P_2) \rightarrow (M_{ba}(P_1 \vee P_2)) \tag{8}$$

It may seem initially that this is the case for implication as well

$$(M_{ba}P_1 \wedge (P_1 \rightarrow P_2)) \rightarrow (M_{ba}P_2) \tag{9}$$

but is intuitively more questionable on its logical equivalent

$$(M_{ba}P_1 \wedge (\neg P_1 \vee P_2)) \rightarrow (M_{ba}P_2) \tag{10}$$

'K' the distribution axiom is generally considered to hold for epistemic logic.

$$(M_{ka}P_1(P_1 \rightarrow P_2)) \rightarrow (M_{ka}P_1 \rightarrow M_{ka}P_2) \tag{11}$$

as does axiom '4'

$$(M_{ka}P_1 \rightarrow (M_{ka}(M_{ka}P_1)) \tag{12}$$

Take axiom 'T': $\Box P \rightarrow P$ and replace modal box for necessity with 'knows':

$$M_{ka}P \rightarrow P. \tag{13}$$

While P implies necessarily P, P doesn't imply that one knows P (axiom 'N'). Axioms 'T', 'K', and '4' appear valid in an epistemic logic, but not axiom 'N'. If we add 'believes' as the analogue to possibility, axiom 'B' doesn't make sense. If P is true that doesn't mean one knows that one believes P. Axiom 'D' is intuitively appropriate with epistemic operators - if one knows P then one also believes P. Axiom '5' appears valid - if one believes P then one knows that one believes P. Axioms 4 and 5 could be called 'introspection' axioms for the epistemic and doxastic operators.

Axiom 'D' is in SUMO for epistemic and doxastic operators.

$$(M_{ka}P) \rightarrow (M_{ba}P). \tag{14}$$

as is axiom '5'.

$$(M_{ba}P) \rightarrow (M_{ka}(M_{ba}P)). \tag{15}$$

4 Deontic logic

It is typical to map obligation to necessity and permition to possibility. However, 'N' doesn't make sense - if P is true that doesn't mean it's obligatory. 'K' could be true - it's obligatory that if you steal money you steal a small amount of money, therefore if you're obliged to steal money then you're obliged to steal a small amount of money (at least several times over). Or should we say if one is obliged to steal money (and one does) that one has in fact also stolen a small amount of money (several times over). We give M subscripts $_o$ for obligation, $_p$ for permission etc. Let p be the proposition that one steals money. We should further clarify the standard example that p should be the proposition that one steals a large amount of money, since if p stands only for the general proposion of stealing money, maybe it's a tiny or insignificant amount of money. So p stands for the proposition that one steals a large amount of money and let p' be the proposition that one steals a small amount of money. If $M_o p \rightarrow M_o p'$ should at the very least only hold true if $p' \in p$.

Axiom 'T' does not appear to be true of obligation - if you ought to do P that doesn't mean P is true since people don't always do what they're supposed to. Axiom '4' is questionable - if you ought to do P does that mean you ought to ought to do P? Axiom 'B' is problematic - if P is true then ought it to be permissible to P? If someone committed murder that doesn't mean that one ought to be permitted to commit murder. Axiom 'D' however appears to make sense - if you ought to do P then you should be permitted to do P. We have this in two forms in SUMO:

```
(=>
  (confersNorm ?E ?F Obligation)
  (confersNorm ?E ?F Permission))

(=>
  (modalAttribute ?FORMULA Obligation)
  (modalAttribute ?FORMULA Permission))
```

and a related axiom

```
(=>
  (deprivesNorm ?E ?F Permission)
  (deprivesNorm ?E ?F Obligation))
```

we might therefore add

```
(=>
  (modalAttribute (not ?FORMULA) Permission)
  (modalAttribute (not ?FORMULA) Obligation))
```

Axiom '5' seems a little problematic - if you have permission for P does that mean you're obliged to have permission? It may be dangerous to have obligation and permission without an authority - what does it really mean that P is permitted? If there's no authority, there can be no notion of permissibility, just possibility. Permission and obligation are always with respect to an authority. "He was permitted to do X (by authority Y)." Once relativized to an authority '5' makes a bit more sense, but only if we have a hierarchy of authority - if entity X gives you permission to do Y then maybe some parent authority A should have obliged X to give you permission for Y,

but that all starts to sound rather contrived. To support the notion of an authority conferring or depriving norms, SUMO has the modal operators `confersNorm` and `deprivesNorm`.

We need many more examples in a common, computable, multi-modal framework to make sure our intuitions accord with a particular axiomatization, and that the axiomatization is consistent across all modalities, and in conjunction with the other formulas that we need to state about the world. Having a much larger inventory of practical axiomatizations of aspects of the world should help to highlight different scenarios where our modal axioms might be true or false.

5 Temporal Logic

SUMO has Allen's 13 temporal relations between intervals[1], and functions that allow specification of calendar dates and times. The function `WhenFn` allows us to related temporal specifications to `Process`es. All the preceding can be handled in first order. However, we also have the relation `holdsDuring`, which relates a `TimePosition` (points and intervals) to a `Formula`.

One system of temporal model logic is that of [18]. There are four modal operators in Prior's system

- P: "It was the case that..." (P stands for "past")
- F: "It will be the case that..." (F stands for "future")
- G: "It always will be the case that..."
- H: "It always was the case that..."

We take 'P' and 'F' to be "It was/will be the case (at one point) that...". These can be handled in SUMO with the functions `FutureFn` and `PastFn` along with a diectic[12] `Now`. For example, "It was the case that John had a blue car."

```
(exists (?C ?T)
  (holdsDuring ?T
    (and
      (during ?T (PastFn Now))
      (instance ?C Automobile)
      (possesses John ?C)
      (attribute ?C Blue))))
```

'F' is handled by instead using `(FutureFn Now)` and 'G' and 'H' by having the `holdsDuring` over the entire `(PastFn...)` or `(FutureFn...)` rather than just being `during` the period.

6 Translation of Modalities

We employ Kripke's *possible worlds* semantics[9] [3].

[3]The notion of possible worlds was first conceived by Liebnitz and also developed by Kripke's contemporary, Lewis[11]

We relax the requirements (per [21]) having the "normal" axiom 'K'. We use Kripke's notion of an *accessibility relation* between possible worlds. The intuition is that modalities select sets of worlds that are related to other worlds. A benefit of this approach is that we can state some formulas involving modalities in a first order logic, as well as using them in higher order logic.

We employ the standard relational semantics for modal logic. A relational model is a tuple $\mathfrak{M} = \langle W, R, V \rangle$ where: W is a set of possible worlds, R is a binary relation on W (which we call the *accessibility relation* between worlds), and V is a valuation function which assigns a truth value to each pair of an atomic formula and a world, (i.e. $V : W \times F \rightarrow \{0, 1\}$ where F is the set of atomic formulae.

Many modal operators in SUMO use the relation `modalAttribute`, which takes a `Formula` and a modal operator as arguments, as we've seen in some of the discussion above. There are an additional number of modalities that take a parameter of some sort, and a formula. These are `considers`, `sees`, `believes`, `knows`, `holdsDuring`, and `desires`. `holdsDuring` is different than the others in that its parameter is a `TimePosition` rather than an `Agent`.

Each of the modalities constrain the set of worlds in which a formula is true. We add a world parameter to every logical literal. Start with the SUO-KIF formula

```
(authors ThomasEdison HarryPotterSorcersStone)
```

that makes the questionable assertion that Thomas Edison authored a Harry Potter book. But

```
(believes JohnSmith
  (authors ThomasEdison HarryPotterSorcersStone))
```

is a bit more reasonable, that merely some deluded person believes he wrote that book. So in the set of worlds corresponding to what John Smith believes, Edison wrote the book. We translate this into an implication (involving an accessibility relation between worlds, and adding a world argument for every logical literal in the formula) that constrains the worlds in which this is true

```
(accreln believes JohnSmith ?W ?W2) =>
  (authors ThomasEdison HarryPotterSorcersStone ?W2))
```

We can also compose modalities, or nest them. Let's say that John only believes this statement on Tuesdays.

```
(and
  (instance ?T Tuesday)
  (holdsDuring ?T
    (believes JohnSmith
      (authors ThomasEdison HarryPotterSorcerersStone))))
```

This becomes

```
(and
  (instance ?T Tuesday)
  (accreln holdsDuring ?T ?W ?W2) =>
    (accreln believes JohnSmith ?W2 ?W3) =>
      (authors ThomasEdison HarryPotterSorcerersStone ?W3))
```

and in THF (leaving aside for now details like term name prefixing, for clarity)

```
thf(kb5,axiom,
  (![W:w,W2:w,W3:w,X:$o,Y:$o,A:$i]:(
    (((instance @ T @ tuesday) &
     ((accreln @ holdsDuring @ T @ W @ W2) =>
       ((accreln @ believes @ johnSmith @ W2 @ W3) =>
         (authors @ thomasEdison @ harryPotterSorcerersStone @ W3))))).
```

7 Test Problems

We have just a handful of test problems at the moment[4], including one that is designed not to get an answer/refutation. We are using "Vampire 4.8 HO - Sledgehammer schedules (2023-10-19)" and "E 3.0pre008 Shangri-La (57b1a04ffd1869871b2c863334343811aea78b68)" on an 8 core Lenovo laptop. But the problems are all quite small so the execution times are minimal.

Problem	Vampire	Eprover
TQM2.tff	0.026	0.008
TQM3.tff	correct, saturation	correct, saturation
TQM3.thf	0.015	0.033
TQM6g.tff	0.008	0.291
TQM6g.thf	0.094	0.35
TQM7.tff	0.011	0.009
TQM7.thf	0.01	0.01

- Problem TQM2 encodes that "Mary knows that Bill likes Sue and Bob likes Joan. Does Mary know that Bill likes Sue. Does Bill like Sue?"
- Problem TQM3.tff encodes that "Bill believes that the Earth is flat. Does Bill know that the Earth is flat?" TQM3.thf adds "If you circumnavigate the Earth and it's flat you will die. Joe circumnavigates the Earth. Does Bill believe Joe will die?"
- Problem TQM6g encodes that "Bill believes things until they are contradicted and then believes the latest thing. Bill sees Mary wearing a red hat (after he sees her wearing a green hat). What color does Bill believe Mary is wearing (now)?". The THF version of this problem is able to state the first sentence generally, but in TFF it has to be specific to the wearing of colored hats.
- Problem TQM7 is from the QMLTP problems[5] and encodes "if a second class flight to Paris necessarily costs $70.00 and I necessarily fly to Paris second class, what is the necessary cost of my flight?". The THF and TFA versions are similar other than we can have numbers in the TFA version but the THF version uses constant terms.

[4] TQM* problems at https://github.com/ontologyportal/sumo/tree/master/tests/TPTP
[5] http://www.iltp.de/qmltp/problems.html

8 Higher Order Interpretations

One advantage of moving to a THF translation of SUMO is that we would have less preprocessing to do for axioms with type guards, predicate variables and row variables. We can refer to relations directly as objects in the logical language and directly use SUMO's domain relation that specifies argument types. We could simplify this even in TFF, as long as we rename relations (which we do with a suffix '__m') when they appear as arguments. To implement this approach in THF however we would still need numbers, which are not yet part of THF.

```
tff(kb1,axiom,(![V__X:$i,V__Y:$i,V__Z:$i,T__X1:$i,T__Y1:$i,T__Y2:$i]: (
  (s__domain(s__connected__m,1,T__X1) & s__instance(V__X,T__X1) &
  s__domain(s__connected__m,1,T__Y1) & s__instance(V__Y,T__Y1) &
  s__domain(s__connected__m,1,T__Z1) & s__instance(V__Z,T__Z1) &
  s__domain(s__part__m,1,T__Y2) & s__instance(V__Y,T__Y2) &
  s__domain(s__part__m,1,T__Z2) & s__instance(V__Z,T__Z2)) =>
  (s__connected(V__X,V__Y) & s__part(V__Y,V__Z)) =>
  (s__connected(V__X,V__Z))))).
```

In THF we do not need to instantiate predicate variables, which significantly reduces the size of the resulting translation from SUMO, since we don't need potentially hundreds of copies of the same formula, such as the axiom of transitivity, for every transitive relation (again assuming that numbers will eventually be supported in THF).

```
thf(kb1,axiom,(![V__INST1:$i,V__INST2:$i,V__INST3:$i,
                 V__R:$i,V__T1:$i,V__T2:$i,V__T3:$i]: (
  ((s__domain @ R @ 1 @ V__T1) & (s__instance @ V__INST1 @ V__T1) &
   (s__domain @ R @ 2 @ V__T2) & (s__instance @ V__INST2 @ V__T2) &
   (s__domain @ R @ 1 @ V__T1) & (s__instance @ V__INST2 @ V__T1) &
   (s__domain @ R @ 2 @ V__T3) & (s__instance @ V__INST3 @ V__T3)) =>
    ((s__instance @ R @ s__TransitiveRelation) =>
        (((R @ V__INST1 @ V__INST2) &
          (R @ V__INST2 @ V__INST3)) =>
            (R @ V__INST1 @ V__INST3))))))
```

9 Conclusions and Future Work

We have shown that it is possible to take SUMO's modal axioms and embed them in a multi-modal logic framework in TFF and THF. Our next step is to implement an automated translator for SUMO that generates TFF and THF, rather than using our current manual approach. The present work is a means to an end of being able to do practical reasoning with knowledge about beliefs, obligations, temporal qualification etc. Once we have an automated translation in place that embodies the manual translations presented, we will attempt to create more tests to determine if we can develop a practically useful system of modal operators and axioms that is also logically consistent. Since up until this point we have only been able to do manual proofs with SUMO's modal constructs, a fully automated framework should allow us to make more rapid progress, and provide a higher level of assurance that the resulting system is logically valid.

10 Acknowledgements

The approach to Kripke semantics in this translation of SUMO, as well as the new higher order translation of SUMO into THF, are thanks to Chad Brown. Any errors in the text are however due solely to the author.

References

[1] Allen, J. F. Towards a general theory of action and time. In *Artificial Intelligence* (1984).

[2] Bachmair, L., Ganzinger, H., Lynch, C., and Snyder, W. Basic Paramodulation and Superposition. In *Proceedings of the 11th International Conference on Automated Deduction* (1992), vol. 607 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 462–476.

[3] Benzmüller, C., and Paulson, L. C. Quantified multimodal logics in simple type theory. *Logica Universalis 7*, 1 (2013), 7–20.

[4] Benzmüller, C., and Pease, A. Progress in automating higher-order ontology reasoning. In *Workshop on Practical Aspects of Automated Reasoning (PAAR-2010)*, B. Konev, R. Schmidt, and S. Schulz, Eds. CEUR Workshop Proceedings, Edinburgh, UK, 2010.

[5] Benzmüller, C., Rabe, F., and Sutcliffe, G. Thf0 – the core of the tptp language for higher-order logic. vol. 5195, pp. 491–506.

[6] Blackburn, P., de Rijke, M., and Venema, Y. *Modal Logic*. No. 53 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, Cambridge, UK, 2001.

[7] Fitting, M., and Mendelsohn, R. L. *First-Order Modal Logic*. Kluwer, 1998.

[8] Goldblatt, R. Mathematical modal logic: A view of its evolution. *Journal of Applied Logic 1*, 5 (2003), 309–392.

[9] Kripke, S. *Naming and Necessity*. Harvard University Press, 1980.

[10] Kripke, S. A. A completeness theorem in modal logic. *Journal of Symbolic Logic 24*, 1 (1959), 1–14.

[11] Lewis, D. K. *On the Plurality of Worlds*. Wiley-Blackwell, Malden, Mass., 1986.

[12] Lyons, J. Deixis, space and time. In *Semantics*. Cambridge University Press, 1977, pp. 636–724.

[13] Niles, I., and Pease, A. Toward a Standard Upper Ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)* (2001), C. Welty and B. Smith, Eds., pp. 2–9.

[14] Pease, A. SUO-KIF Reference Manual. `https://github.com/ontologyportal/sigmakee/blob/master/suo-kif.pdf`, 2009. retrieved 20 June 2020.

[15] Pease, A. *Ontology: A Practical Guide*. Articulate Software Press, Angwin, CA, 2011.

[16] Pease, A. Converting the Suggested Upper Merged Ontology to Typed First-order Form.

[17] Pease, A., and Schulz, S. Contradiction detection and repair in a large theory. In *The International FLAIRS Conference Proceedings* (2022).

[18] Prior, A. N. *Time and Modality: Being the John Locke Lectures for 1955-6 Delivered in the University of Oxford*. Oxford University Press, Oxford, England, 1957.

[19] Raths, T., and Otten, J. The qmltp problem library for first-order modal logics. In *Automated Reasoning* (Berlin, Heidelberg, 2012), B. Gramlich, D. Miller, and U. Sattler, Eds., Springer Berlin Heidelberg, pp. 454–461.

[20] Schulz, S., Sutcliffe, G., Urban, J., and Pease, A. Detecting inconsistencies in large first-order knowledge bases. In *Proceedings of CADE 26* (2017), Springer, pp. 310–325.

[21] Scott, D. *Advice on Modal Logic*. Springer Netherlands, Dordrecht, 1970, pp. 143–173.

[22] Sinnott-Armstrong, W. A solution to forrester's paradox of gentle murder. *Journal of Philosophy 82*, 3 (1985), 162–168.

[23] Sutcliffe, G. The TPTP world - infrastructure for automated reasoning. In *LPAR (Dakar)* (2010), E. M. Clarke and A. Voronkov, Eds., vol. 6355 of *LNCS*, Springer, pp. 1–12.

[24] Sutcliffe, G., Schulz, S., Claessen, K., and Baumgartner, P. The TPTP typed first-order form with arithmetic. In *LPAR* (2012), N. Bjørner and A. Voronkov, Eds., vol. 7180 of *LNCS*, Springer, pp. 406–419.

[25] Trac, S., Sutcliffe, G., and Pease, A. Integration of the TPTPWorld into SigmaKEE. In *Proceedings of IJCAR '08 Workshop on Practical Aspects of Automated Reasoning (PAAR-2008)* (2008), CEUR Workshop Proceedings.