

Arithmetic and Inference in a Large Theory

Adam Pease, Infosys, Foothill Research Center
adam.pease@infosys.com

<http://www.ontologyportal.org/>

Motivation

“Robot! Pick up my car”

- Or “Pick up my cart”

-

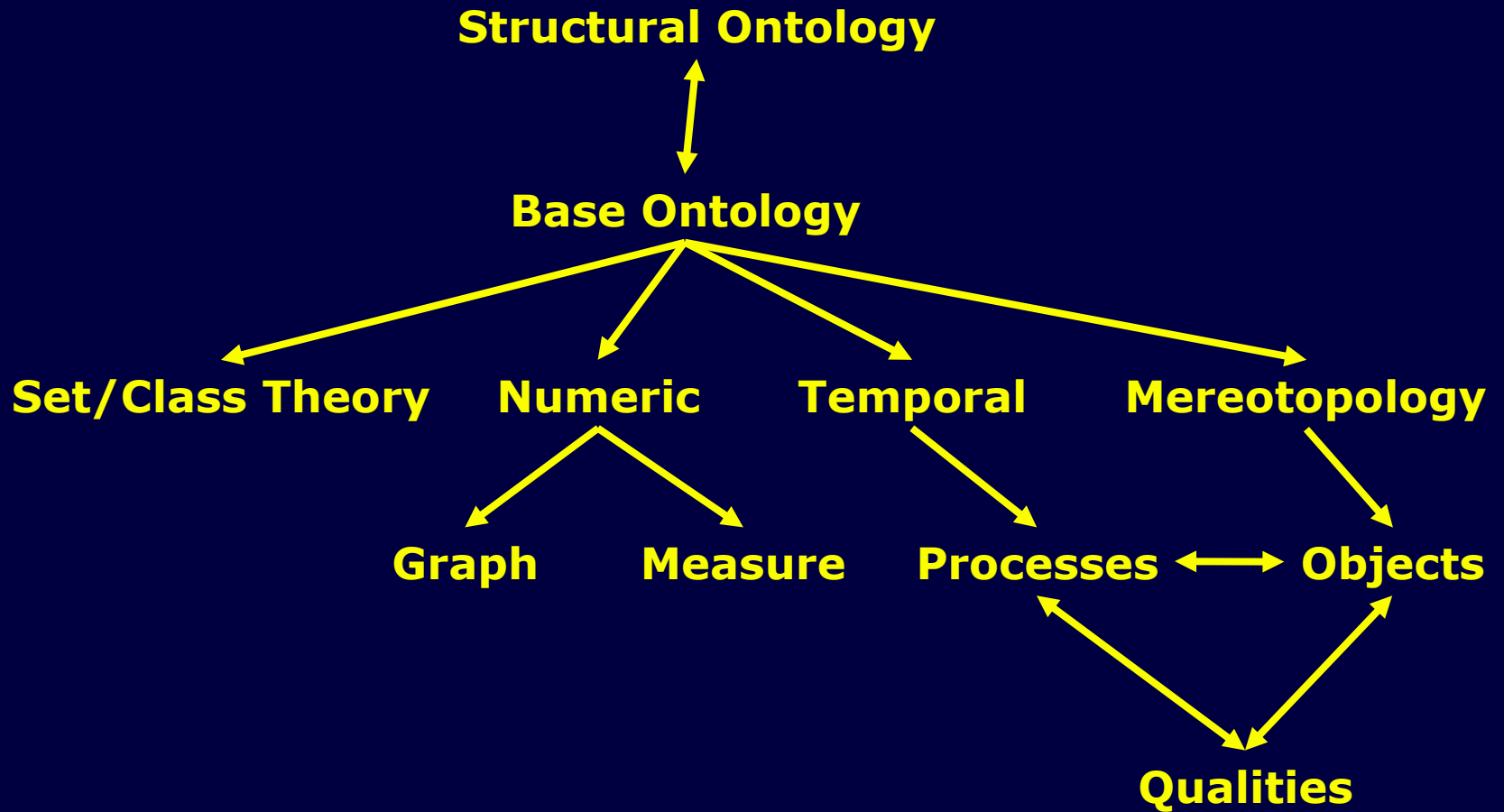
Disambiguate car/cart by knowing robot can carry $< 100\text{kg}$, car weighs 1.5 tons

And explain “why didn’t you bring me my car!”

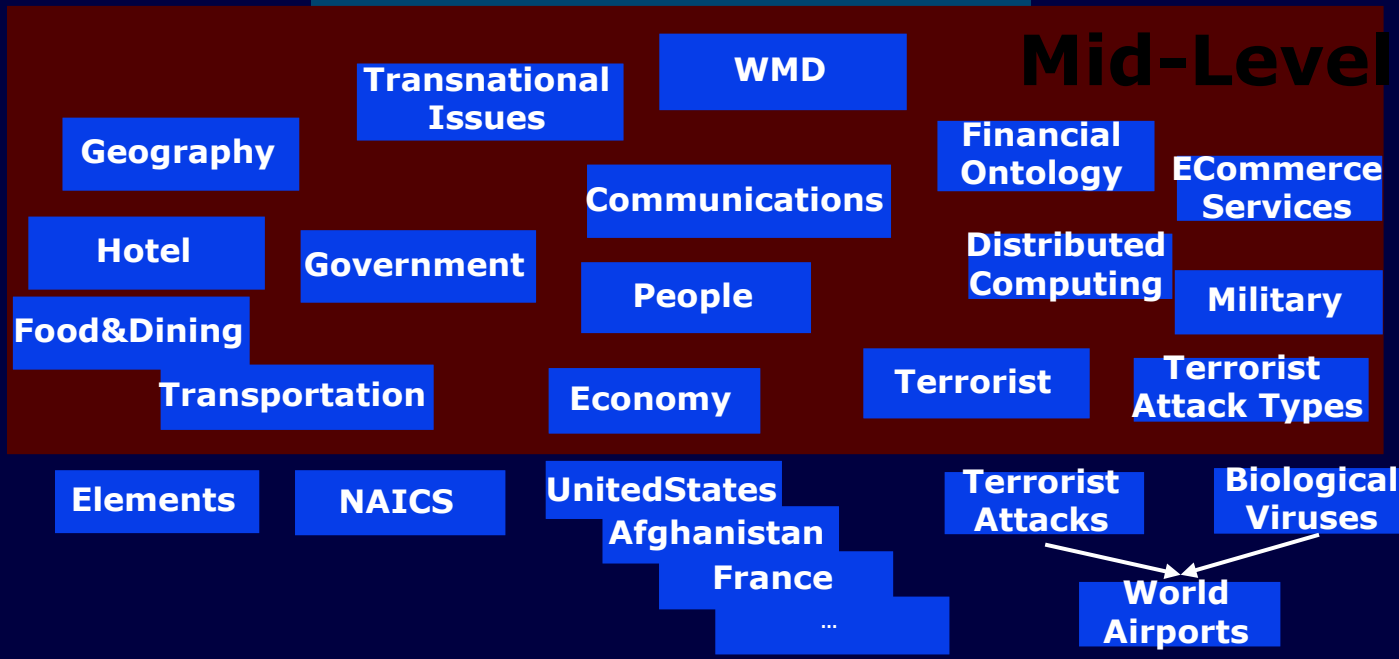
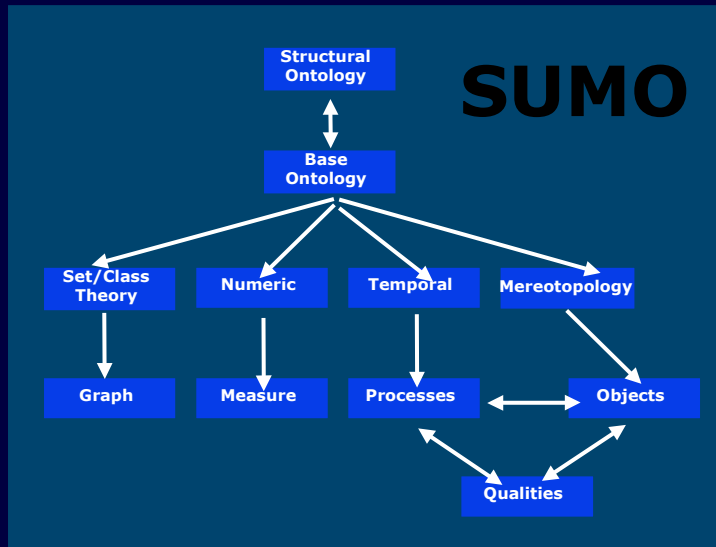
Suggested Upper Merged Ontology

- Associated domain ontologies totalling 20,000 terms and 80,000 axioms in (up to) HOL
 - Linked with factbases including YAGO for millions of facts
 - Most of SUMO is FOL, arithmetic is common, some HOL
- Mapped by hand to all 100k word senses in WordNet lexicon
- Open source toolset Sigma for translating to different logics, debugging, analysis, interface to theorem provers, NLP
- Free – tools and theory are GNU GPL
- Many SUMO-based problems in TPTP
- First release in year 2000

SUMO Structure



SUMO+Domain Ontology



Problem Domains

Software engineering

- Rather than just x : real I'd like to know x : humanAge, greater than 0, less than 122 etc
- Static analysis of data consistency, rather than coverage of execution

Expert Reasoning

- Large machine troubleshooting
- Prove correctness of constraints and diagnosis procedure
- Reuse – and robustness of design with changing scope and requirements

“Casual Semantics”

Thanks Chad!

- Interpretation of SUMO into TPTP, TFF0, THF gives a semantics
- When SUMO started, FOL ATP was still very hard, HOL ATP arguably just “academic”
 - Record what we know about the world, figure out how to do ATP with all of it later

SUMO Characteristics

Deep hierarchy of types

- Up to a dozen or so “levels”
- But sorts in TFF0 are disjoint

All relation arguments have types

- No need for an explicit sort syntax

ATP with SUMO

“Needle in a haystack” - lots of axioms
but few are relevant to any given
conjecture

Hoder's SUMO Inference Engine (SInE)
method had dramatic performance
effect

- Axiom selection - prove over likely
relevant subset

Static Checking

method X in class Y cannot be applied to given types

```
tff(kb_SUMO_351, axiom, (  
  ! [V__ROW1 : $int, V__NUMBER : $int] :  
    (s__GreatestCommonDivisorFn_1__0In1InFn(V__ROW1) =  
      V__NUMBER =>  
    ! [V__ELEMENT:$int] :  
      (s__inList(V__ELEMENT, s__ListFn_1__1InFn(V__ROW1)) =>  
        s__RemainderFn__0In1In2InFn(V__ELEMENT, V__NUMBER) =  
          0))))).
```

Parsing Error on line 3822

The sort \$int of function argument X6 does not match the expected sort \$i

Sorts

```
(domain AdditionFn 1 Quantity)
(domain AdditionFn 2 Quantity)
```

```
(domain FloorFn 1 RealNumber)
(range FloorFn Integer)
```

```
(domain finishes 1 TimeInterval)
(domain finishes 2 TimeInterval)
```

```
(domainSubclass typicalPart 1 Object)
(domainSubclass typicalPart 2 Object)
```

```
(domain PathWeightFn 1 GraphPath)
(range PathWeightFn Quantity)
```

```
tff(floorFn__1InFn_sig,type,
s__FloorFn__1InFn : ( $int ) >
$int ).
```

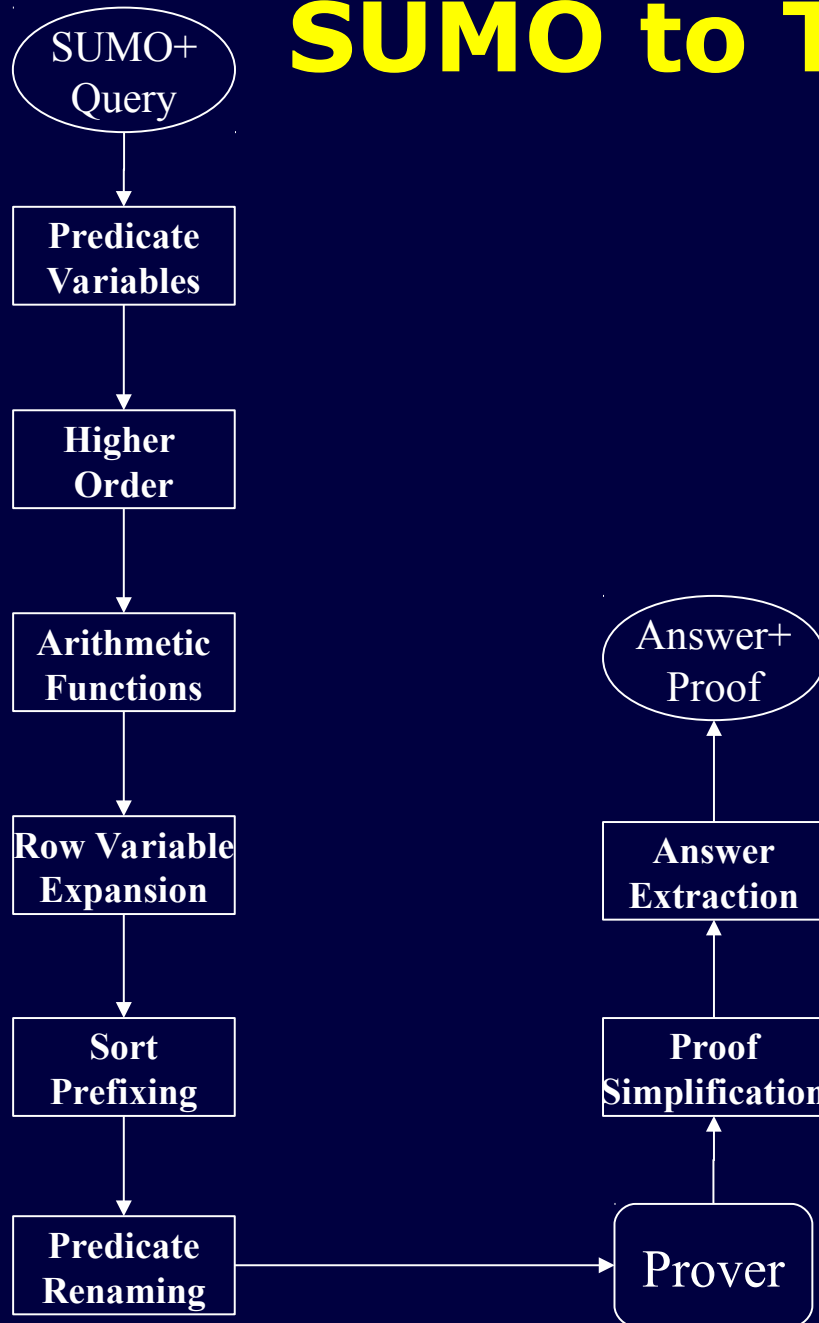
```
tff(pathWeightFn__0ReFn_sig,type,
s__PathWeightFn__0ReFn :
( $i ) > $real ).
```

```
tff(pathWeightFn__0InFn_sig,type,
s__PathWeightFn__0InFn :
( $i ) > $int ).
```

```
tff(pathWeightFn__0RaFn_sig,type,
s__PathWeightFn__0RaFn :
( $i ) > $rat ).
```

SUMO to TPTP

Thanks to Geoff Sutcliffe & Stephan Schulz



- expanding "row variables"
 - similar to Lisp's @REST construct
- instantiating "predicate variables" with all possible values
- expanding the arity of all variable arity relations with different names
- renaming any relations given as arguments to other relations
- Add type guards to axioms to express sorts

Quantity

PhysicalDimension

Number

RealNumber

RationalNumber

Integer

EvenInteger

OddInteger

PrimeNumber

NonnegativeInteger

PositiveInteger

NegativeInteger

IrrationalNumber

NonnegativeRealNumber

PositiveRealNumber

PositiveInteger

NegativeRealNumber

NegativeInteger

BinaryNumber

ImaginaryNumber

ComplexNumber

PhysicalQuantity

ConstantQuantity

TimeMeasure

TimeDuration

TimePosition

TimeInterval

Numerical Hierarchy

Arithmetic Axioms

```
(=>
  (and
    (instance ?PM ParticulateMatter)
    (part ?Particle ?PM)
    (approximateDiameter ?Particle
      (MeasureFn ?Size Micrometer))
    (greaterThan 10 ?Size)
    (greaterThan ?Size 2.5))
  (instance ?PM CoarseParticulateMatter)))
```

```
(=>
  (and
    (instance ?AREA DesertClimateZone)
    (subclass ?MO Month)
    (averageTemperatureForPeriod ?AREA ?MO ?TEMP)
    (greaterThan ?TEMP
      (MeasureFn 18 CelsiusDegree)))
  (instance ?AREA SubtropicalDesertClimateZone))
```

Basic Algorithm

- Collect the types of all variables from relation types
- Constrain types in equality or inequality to lowest type (above \$int)
- "promote" types that are specializations of Integer or RealNumber
 - if a number is an Integer add ".0" to it so it's a real and record its type as a RealNumber
- Rename the RelationExtendedToQuantities with a suffix if its arguments have been constrained to those types (or their subclasses)
- for variable types below Quantity but not below Number in the SUMO hierarchy, create two version of the axiom - one with all the original names of the predicates and one with every SUMO predicate without a TFF0 equivalent having a type specification suffix (as in the previous step) and those that do have a TFF0 equivalent converted to that equivalent
- Translate symbols to TFF0 syntax, including translating all relations that have a corresponding native name (such as \$sum)

Quantity Comparison Axioms

```
(=>
  (and
    (equal ?Q1 (MeasureFn ?I1 ?U))
    (equal ?Q2 (MeasureFn ?I2 ?U))
    (greaterThan ?I1 ?I2))
  (greaterThan ?Q1 ?Q2))
```

More general:

```
(=>
  (and
    (instance ?REL RelationExtendedToQuantities)
    (equal ?Q1 (MeasureFn ?I1 ?U))
    (equal ?Q2 (MeasureFn ?I2 ?U))
    (?REL ?I1 ?I2))
  (?REL ?Q1 ?Q2))
```


Type Promotion

PositiveRealNumber \rightarrow RealNumber + definitional constraint

(=>

```
(instance ?X PositiveRealNumber)
(greaterThan ?X 0))
```

definitional constraint

(=>

```
(measure ?QUAKE
  (MeasureFn ?VALUE RichterMagnitude))
(instance ?VALUE PositiveRealNumber))
```

axiom

(=>

```
(measure ?QUAKE
  (MeasureFn ?VALUE RichterMagnitude))
(greaterThan ?VALUE 0))
```

substitution

```
! [V_QUAKE : $i, V_VALUE : $real] :
(s__measure (V_QUAKE,
  s__MeasureFn (V_VALUE, s__RichterMagnitude)) =>
$greater (V_VALUE, 0.0)
```

TFF0

Consistent?

No guarantee for 80k axioms but 3600
sec on Vampire fails to find
contradiction

Problems Found

```
(=>
  (instance ?NUMBER Quantity)
  (equal 1
    (MultiplicationFn ?NUMBER
      (ReciprocalFn ?NUMBER))))
```

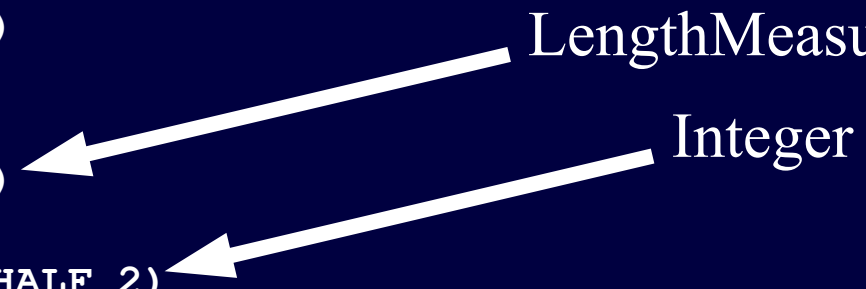
```
(=>
  (and
    (instance ?NUMBER RealNumber)
    (not
      (equal ?NUMBER 0)))
  (equal 1
    (MultiplicationFn ?NUMBER
      (ReciprocalFn ?NUMBER))))
```

Sort Conflict

```
(=>
  (diameter ?CIRCLE ?LENGTH)
  (exists (?HALF)
    (and
      (radius ?CIRCLE ?HALF)
      (equal
        (MultiplicationFn ?HALF 2)
        ?LENGTH))))
```

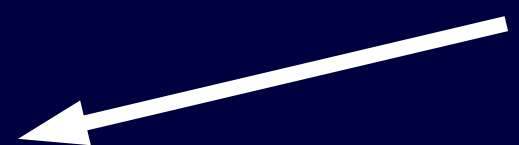
LengthMeasure

Integer



```
(=>
  (diameter ?CIRCLE ?LENGTH)
  (exists (?N ?U)
    (and
      (radius ?CIRCLE (MeasureFn ?N ?U))
      (equal
        (MeasureFn
          (MultiplicationFn ?N 2)
          ?U)
        ?LENGTH))))
```

LengthMeasure



Code

<https://github.com/ontologyportal/sigma>

```
java -Xmx7g -classpath  
  /home/apease/workspace/sigma/build/cl  
  asses:/home/apease/workspace/sigma/  
  build/lib/*:/home/apease/workspace/  
  sigma/lib/*  
  com.articulate.sigma.trans.SUMOKBtoTFAKB
```

Proof

```
tff(f4,axiom,(
s__instance(s__Carry1,s__Carrying) &
s__patient(s__Carry1,s__MyCar) &
s__instrument(s__Carry1,s__Robot1)),
file('/home/apease/.sigmakee/Kbs/Robot-small.tff',assert)).
```

... 45 steps total ...

```
(tff(f95,plain,(
$false),
inference(evaluation,[],[f78])).
% SZS output end Proof for Robot-small
% -----
% Version: Vampire 4.2.2 (commit 6588b35 on 2018-07-19 13:39:17
+0200)
% Termination reason: Refutation
```

Also works in CVC4 – thanks Geoff!

Will be part of CASC-TPTP after CASC 2019

Ontology: A Practical Guide is a comprehensive textbook for Computer Scientists, Database Developers, Linguists and other professionals and students looking to add rigor to their computational systems. This book is suitable as an advanced textbook or self-study guide for the professional as well as a resource for classroom instruction. It covers many of the relevant aspects of formal ontology, including relationships to other approaches to information design, knowledge representation languages, ontology content development and linguistic semantics. It offers a practical approach based on free software including ontology development tools, a representation language, a lexicon, and a large ontology.



About the author

Adam Pease is the CEO of Articulate Software and the developer of the free software described in this book. He is a consultant on ontology development and implementation.

\$25.00 U.S.



ISBN 1-889455-10-5
ISBN-13 978-1-889455-10-5

All designs and
photographs by the
author.

Ontology

Pease

Articulate
Software
Inc.

Ontology



A Practical Guide
Adam Pease

<http://www.ontologyportal.org/Book.html>