Modular Construction of Models

Till Mossakowski[1,2] Oliver Kutz[1] Dominik Lücke[1]

[1]SFB/TR 8 Spatial Cognition
University of Bremen

[2]DFKI GmbH Bremen
Safe & Secure Cognitive Systems

IFIP WG 1.3
July 4, 2010, Etelsen

Outline

1. Motivation
2. The Common Algebraic Specification Language (CASL)
3. DOLCE
4. Model Finders
5. Models along architectural specifications

Motivation

- Foundational ontologies provide the language and semantics for domain ontologies
- they are specified in many cases in **FOL**: like DOLCE and SUMO
- important question: Do they have a model? Are they consistent?
- Model-finders often fail to find a model for them directly
- several inconsistencies have already been found in SUMO [Ian Horrocks, Andrei Voronkov (2006)]
- SUMO-challenge on `http://www.tptp.org` has a first winner of $100
- we can construct a global model from smaller ones using CASL architectural specifications

The Common Algebraic Specification Language (CASL)

CASL

- CASL is a first order language designed by CoFI and approved by IFIP WG 1.3

Example (Basic spec)

spec TEMPORARY_STRICT_PARTIAL_ORDER =
  esorts $s < EDorPDorQ; T$
  pred $Rel : s \times s \times T$
  $\forall\, x, y, z : s; t : T$
  $\bullet\ Rel(x, y, t) \Rightarrow \neg\, Rel(y, x, t)$
  $\bullet\ Rel(x, y, t) \wedge Rel(y, z, t) \Rightarrow Rel(x, z, t)$
end

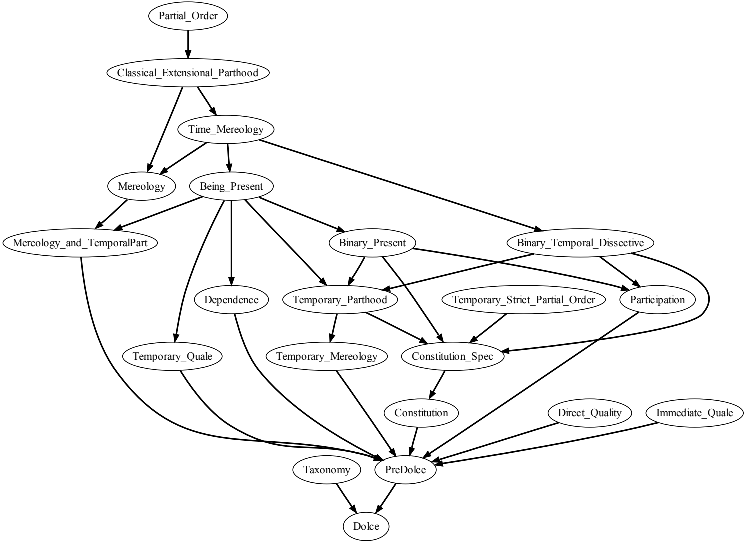
- SP ::= BasicSP | SP then SP | SP and SP | SP with σ | SP hide σ
- tool support is available via HETS (the Heterogeneous Tool Set)

Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE)

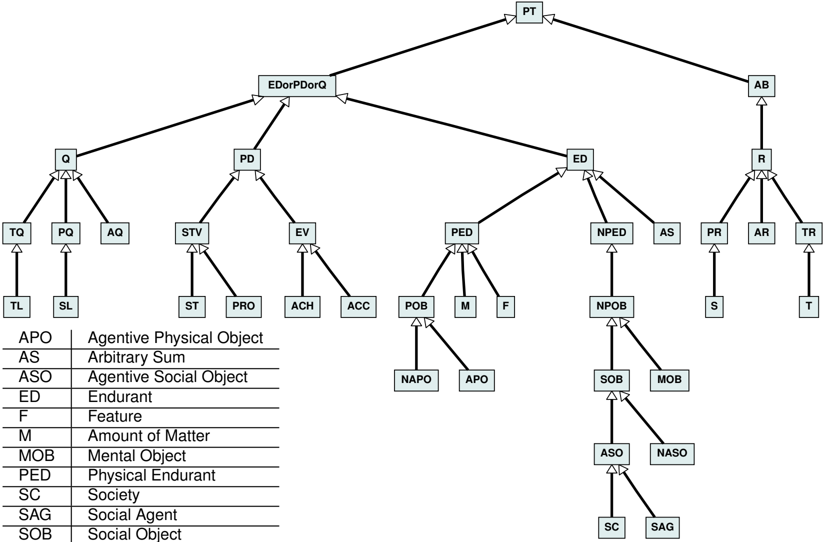
Dolci?

- DOLCE: Descriptive Ontology for Linguistic and Cognitive Engineering
- developed at the LOA in Trento
- contains several hundreds of axioms
- initially formalized in KIF (some variant of first-order logic)
- modularized formalization in CASL is also available and the starting point of our work
- complexity of DOLCE stems from the fact that it combines several (non-trivial) formalised ontological theories into one theory
  - theories of essence and identity
  - parts and wholes (mereology)
  - dependence
  - composition and constitution
  - properties and qualities

DOLCE's Modules



DOLCE's Taxonomy



APO	Agentive Physical Object
AS	Arbitrary Sum
ASO	Agentive Social Object
ED	Endurant
F	Feature
M	Amount of Matter
MOB	Mental Object
PED	Physical Endurant
SC	Society
SAG	Social Agent
SOB	Social Object
POB	Physical Object

Model Finders

Model finders

- We have made experiments with several Model finders on DOLCE
  - Darwin
  - SPASS
  - Isabelle-refute

CEP, a part of DOLCE

Example (Classical extensional parthood ⟨*CEP*⟩)

sort s; **pred** $At : s$; **pred** $AtP : s \times s$; **pred** $Ov : s \times s$
pred $P : s \times s$; **pred** $PP : s \times s$; **pred** $Sum : s \times s \times s$

$\forall\, x : s \bullet P(x, x)$ %(reflexivity)%

$\forall\, x, y : s \bullet P(x, y) \wedge P(y, x) \Rightarrow x = y$ %(antisymmetry)%

$\forall\, x, y, z : s \bullet P(x, y) \wedge P(y, z) \Rightarrow P(x, z)$ %(transitivity)%

$\forall\, x : s; y : s \bullet PP(x, y) \Leftrightarrow P(x, y) \wedge \neg\, P(y, x)$ %(Dd14)%

$\forall\, x : s; y : s$
$\bullet\; Ov(x, y) \Leftrightarrow \exists\, z : s \bullet P(z, x) \wedge P(z, y)$ %(Dd15)%

$\forall\, x : s \bullet At(x) \Leftrightarrow \neg\, \exists\, y : s \bullet PP(y, x)$ %(Dd16)%

$\forall\, x : s; y : s \bullet AtP(x, y) \Leftrightarrow P(x, y) \wedge At(x)$ %(Dd17)%

$\forall\, z : s; x : s; y : s$
$\bullet\; Sum(z, x, y) \Leftrightarrow \forall\, w : s \bullet Ov(w, z) \Leftrightarrow Ov(w, x) \vee Ov(w, y)$

$\forall\, x, y : s \bullet \exists\, z : s \bullet Sum(z, x, y)$ %(Existence of the sum)%

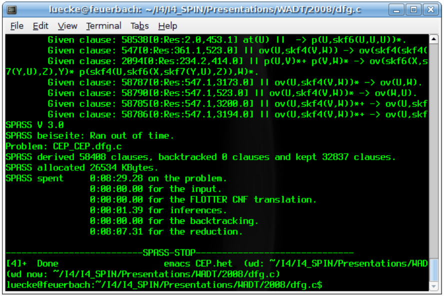
CEP, a part of DOLCE

With a bit of meta-reasoning, we can see that

finite CEP-models = finite powersets without $\emptyset$

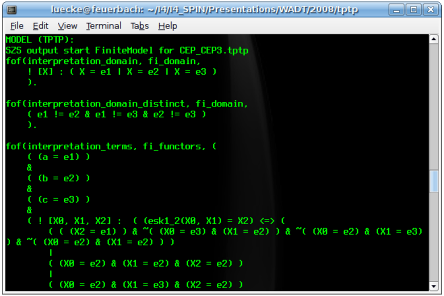
SPASS

- SPASS is a first order theorem prover based on resolution
- can check consistency if for a theory the *Th* the problem is given as $Th \vdash False$
- *Th* is consistent if SPASS reaches saturated set of clauses in such a problem
- could not verify consistency of CEP



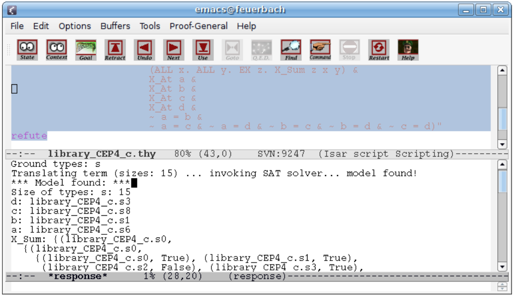
Darwin

- Darwin is a first order theorem prover/model finder based on the model evolution calculus
- it can find a counter-model for $Th \vdash False$ as well as a model for Th in a constructive way
- output in: TPTP, DIG
- scored quite well at the CASC in the last years
- could find a model with 3 atoms for CEP



Isabelle-refute

- part of the Isabelle interactive theorem prover
- uses SAT solver to find finite counter-models for first order specifications, so negation of the actual theory is used
- could find a model with 4 atoms for CEP; with some help: expected size of the model had to be supplied
- drawback: CASL sub-sorting is not supported directly



Comparison

	CEP_1	CEP_2	CEP_3	CEP_4
SPASS	×	×	×	×
Darwin	✓	✓	✓	×
Isabelle-refute	✓	✓	✓	✓

- Isabelle needed help in form of the specification of the actual size of the model for CEP_4
- none of the model-finders was able to find a model for DOLCE within several days/weeks

Models along architectural specifications

Unit-Specifications

Units are:

- named models $U : USP$

Unit specifications *USP* are:

- structured specifications *SP* of single units, for which a model has to be found directly (after flattening)
- specifications $SP_1 \times \cdots \times SP_n \xrightarrow{\tau} SP_{n+1}$ of parameterized units (roughly theory-extensions)

Syntax of architectural specifications in a nutshell

Definition (arch spec)

arch spec ASP $=$

units $U_1 : USP_1$;

...

$U_n : USP_n$

result UT

end

with U_i being the names of unit-models or parameterized unit-functions that map models to models, USP_i being their specifications

Definition (Syntax.)

- Unit Declarations: $U : SP \mid U_F : SP_1 \times \cdots \times SP_n \xrightarrow{\tau} SP_{n+1}$
- Unit Terms: $U \mid T_1 \, \textbf{and} \, T_2 \mid U_F[T_1] \ldots [T_n]$

Semantics of architectural specifications in a nutshell

arch spec ASP $=$
    units $U_1 : USP_1$;
        ...
        $U_n : USP_n$
    result UT
end

$[\![\mathrm{ASP}]\!] =$
$E \mapsto [\![UT]\!]_E$

Unit environments:
$E = (F_1, \ldots, F_n) \in \mathbf{Mod}(USP_1) \in \times \cdots \times \mathbf{Mod}(USP_n)$

Semantics of unit terms:

- $[\![U_i]\!]_E = F_i$
- $[\![T_1 \text{ **and** } T_2]\!]_E = [\![T_1]\!]_E \oplus [\![T_2]\!]_E$ (amalgamation)
- extended static semantics of arch specs guarantees that amalgamability is always ensured (using sharing analysis)

The consistency proof

- write an arch spec for the decomposition of DOLCE
-
-
    -
        -
    -
        -
        -
        -
-
    -
        -
    -
        -

The consistency proof

- write an arch spec for the decomposition of DOLCE
- check its well-formedness using HETS
-
    -
        -
    -
        -
        -
        -
-
    -
        -
    -
        -

The consistency proof

- write an arch spec for the decomposition of DOLCE
- check its well-formedness using HETS
- prove consistency of the architectural specification
    -
        -
    -
        -
        -
        -
-
    -
        -
    -
        -

The consistency proof

- write an arch spec for the decomposition of DOLCE
- check its well-formedness using HETS
- prove consistency of the architectural specification
- prove that DOLCE refines to the architectural spec

The consistency proof

- write an arch spec for the decomposition of DOLCE
- check its well-formedness using HETS
- prove consistency of the architectural specification:
  - prove consistency of non-parameterised unit specs
    - all of them are small ⇒ find models using e.g. Darwin
  -
    -
    -
    -
- prove that DOLCE refines to the architectural spec
  -
    -
  -
    -

The consistency proof

- write an arch spec for the decomposition of DOLCE
- check its well-formedness using HETS
- prove consistency of the architectural specification:
  - prove consistency of non-parameterised unit specs
    - all of them are small $\Rightarrow$ find models using e.g. Darwin
  - prove consistency of parameterised unit specs:
    - show that result spec is conservative over parameter spec:
    - construct a free extension of parameter spec, with recursive definitions (this is known to be conservative)
    - show that this is a refinement of the result spec
- prove that DOLCE refines to the architectural spec
  -
    -
  -
    -

The consistency proof

- write an arch spec for the decomposition of DOLCE
- check its well-formedness using HETS
- prove consistency of the architectural specification:
  - prove consistency of non-parameterised unit specs
    - all of them are small $\Rightarrow$ find models using e.g. Darwin
  - prove consistency of parameterised unit specs:
    - show that result spec is conservative over parameter spec:
    - construct a free extension of parameter spec, with recursive definitions (this is known to be conservative)
    - show that this is a refinement of the result spec
- prove that DOLCE refines to the architectural spec:
  - construct a unit spec for the architectural spec
    - use proof calculus presented by Mihai at WADT 2010
  -
    -

The consistency proof

- write an arch spec for the decomposition of DOLCE
- check its well-formedness using HETS
- prove consistency of the architectural specification:
  - prove consistency of non-parameterised unit specs
    - all of them are small $\Rightarrow$ find models using e.g. Darwin
  - prove consistency of parameterised unit specs:
    - show that result spec is conservative over parameter spec:
    - construct a free extension of parameter spec, with recursive definitions (this is known to be conservative)
    - show that this is a refinement of the result spec
- prove that DOLCE refines to the architectural spec:
  - construct a unit spec for the architectural spec
    - use proof calculus presented by Mihai at WADT 2010
  - prove that DOLCE refines to this unit spec
    - can be proved using structural development graph rules alone

Some data and lessons learned

- arch spec has 38 units
- well-formedness check using HETS not feasible
- after split into four arch specs, well-formedness check using HETS took 35h on i7
  - for chosing the split, unit dependency diagrams needed
  - often, only parts linked by several arrows can be found
    $\Rightarrow$ appropriate restriction of units needed
- unit dependency diagrams also needed in order to understand amalgamability problems
  $\Rightarrow$ display of diagrams of extended static semantics implemented

Lessons learned (cont'd)

- first attempt: arch spec structure follows that of structured spec ⇒ failed (due to DEPENDENCE)
- second attempt followed structured of taxonomy ⇒ successful
- by using a strengthening of DEPENDENCE, we could rely on stronger assumptions for the interpretation of DEPENDENCE for various subconcepts when extending it to a superconcept.
- only subsorted logic allows for the architectural decomposition, single-sorted logic does not (universe has to be fixed at once)

Conclusion

- Standard model finders cannot cope with DOLCE
- Developed a CASL architectural specification for DOLCE, hence we have split the task of constructing a DOLCE model into several independent subtasks
- Use of subsorting has been crucial for obtaining the decomposition

Future Work

- Checking if all extensions in the arch spec are conservative
- Using our approach for other large theories like the SUMO ontology
- Deriving a toolkit for model-finding for large theories
- Adding support for semi-automatic derivation of arch specs from structured specifications to HETS

The End

Thank You