# Ontology Evaluation Workflow in COLORE

Michael Grüninger

Track D: Software Environments for Evaluating Ontologies

February 14, 2013

# COLORE

## Common Logic Ontology Repository

- The objective of the COLORE project is to construct an open repository of first-order ontologies that will serve as a testbed for ontology evaluation and integration techniques, and that can support the design, evaluation, and application of ontologies in first-order logic.
- COLORE is a participant in the Open Ontology Repository Initiative `colore.oor.net`
- Today we will explore ontology evaluation within COLORE – what happens when an ontology is uploaded into the repository?

# Ontology Evaluation

- Consistency and entailment
- Relationships and comparisons to other ontologies
- Verification
- Modularity
- Reusing ontologies

# Consistency

- Each ontology is translated into Prover9 syntax, and we attempt to construct a nontrivial model.
- All models are stored in the `consistency` subdirectories of the repository.
- Example:
  `http://code.google.com/p/colore/source/browse/trunk/verification/complex/combined_time/consistency/`

# Entailments

- What are the logical consequences of the ontology?
- Example:
  `http://code.google.com/p/colore/source/browse/trunk/verification/complex/combined_time/entailment/endpoints_lemmas`
- Competency questions
  - Given a motivating scenario, a set of queries will arise which place demands on an underlying ontology. We can consider these queries to be requirements that are in the form of questions that an ontology must be able to answer.
  - The competency questions are defined formally as an entailment or consistency problem with respect to the axioms in the ontology.

# Relationships to Other Ontologies

## Comparing ontologies

- Do two ontologies make the same commitments?
- What are the differences between two ontologies?

## Sharability

- Which subtheories are common between two ontologies?

# Metalogical Relationships in COLORE

## Do the theories have the same signature?

- Nonconservative extension

## Do the theories have different signatures?

- Conservative extension
- Relative interpretation

# Theories and Extensions

## Definition

Let $T_1, T_2$ be two first-order theories such that $\Sigma(T_1) \subseteq \Sigma(T_2)$.
We say that $T_2$ is an extension of $T_1$ iff for any sentence $\sigma \in \mathcal{L}(T_1)$,

$$\text{if} \quad T_1 \models \sigma \quad \text{then} \quad T_2 \models \sigma.$$

$T_2$ is a conservative extension of $T_1$ iff for any sentence $\sigma \in \mathcal{L}(T_1)$,

$$T_2 \models \sigma \quad \text{iff} \quad T_1 \models \sigma.$$

$T_2$ is a non-conservative extension of $T_1$ iff $T_2$ is an extension of $T_1$ and there exists a sentence $\sigma \in \Sigma(T_1)$ so that

$$T_1 \not\models \sigma \quad \text{and} \quad T_2 \models \sigma.$$

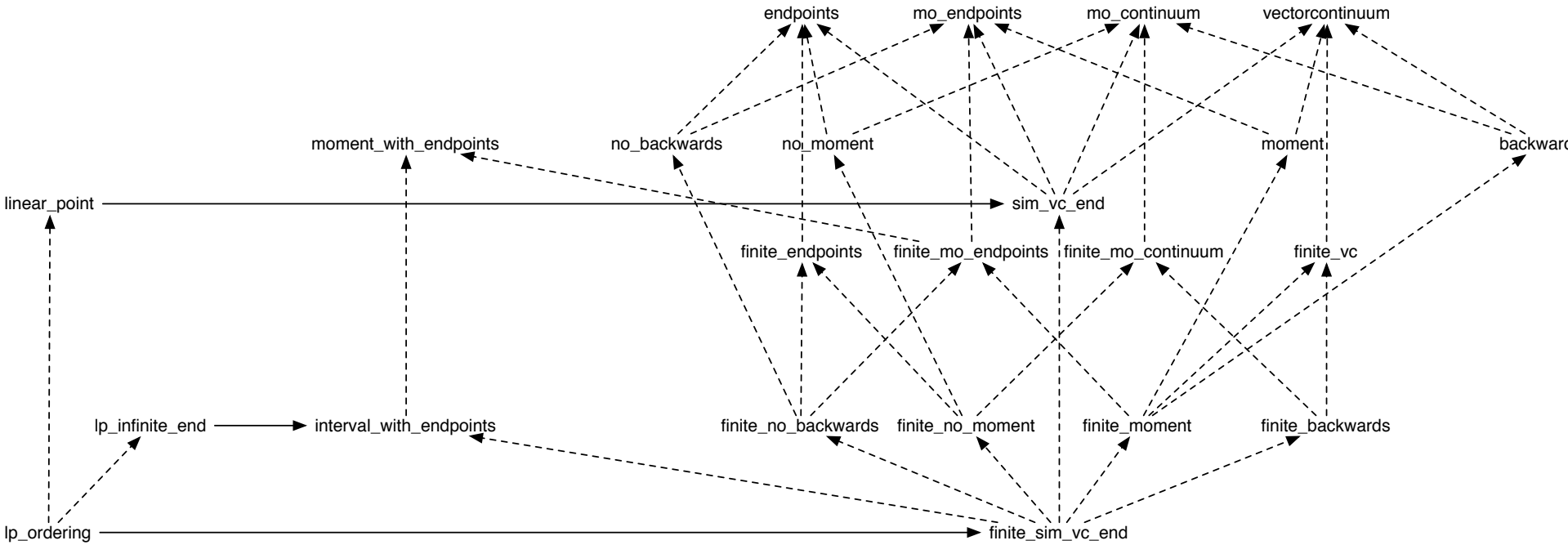
# Hierarchies

## Definition

A hierarchy $\mathbb{H} = \langle \mathcal{H}, \leq \rangle$ is a partially ordered, finite set of theories $\mathcal{H} = T_1, ..., T_n$ such that

1. $\Sigma(T_i) = \Sigma(T_j)$, for all $i, j$;
2. $T_1 \leq T_2$ iff $T_2$ is an extension of $T_1$;
3. $T_1 < T_2$ iff $T_2$ is a non-conservative extension of $T_1$.

# Combined Time Ontologies



# Interpretability

Specify mappings between models by specifying the relationships between modules in the repository.

- Interpretable – mapping between theories that preserves theorems
- Faithfully interpretable – mapping between theories that preserves models
- Definable equivalence – theories that are interpretable into each other.

# Reducibility

**Definition**

A theory $T$ is reducible to a set of theories $T_1, ..., T_n$ iff

1. $T$ faithfully interprets each theory $T_i$,

$$T \cup \Delta_i \models T_i$$

2. $T_1 \cup ... \cup T_n$ faithfully interprets $T$.

$$T_1 \cup ... \cup T_n \cup \Pi \models T$$

Example:
`http://code.google.com/p/colore/source/browse/trunk/verification/complex/combined_time/interpret/endpoints2strict_graphical`

# Amalgamating Models

**Theorem**

*Let $\mathfrak{N}$ be the set of amalgamations of models in*

$$\langle \mathcal{N}_1, ..., \mathcal{N}_n \rangle \in Mod(T_1) \times ... \times Mod(T_n)$$

*for the theories $T_1, ..., T_n$.*
*A theory $T$ is reducible to $T_1, ..., T_n$ iff there is a bijection $\varphi : Mod(T) \rightarrow \mathfrak{N}$, such that $\mathcal{M}$ is definably equivalent to $\varphi(\mathcal{M})$.*

# Ontology Verification

- With verification, we want to characterize the models of an ontology up to isomorphism (or elementary equivalence) and determine whether or not these models are equivalent to the intended models of the ontology.

# Modularity

Can we decompose an ontology into smaller modules?

- Theories in the reduction of an ontology $T$ are definably equivalent to modules of $T$.
- The reduction gives us a modular decomposition of the ontology, and also provides guidance on how to combine smaller ontologies into larger ones.

# Procedures

- The *FindTheory* Procedure finds a set of theories which are interpretable by the theory $T$.
- The *Decomp* Procedure finds the modules of $T$ that are definably equivalent to the theories in the reduction of $T$.
- The *UpdateHierarchy* Procedure inserts a new theory into a hierarchy and guarantees that similarities and differences are also theories in the hierarchy.

# Reusing Ontologies

## Which ontology in the repository do I need?

- We can adapt the Procedures to search for the weakest theories in a hierarchy that can entail a given set of competency questions.
- We can also search for the weakest theories that are satisfied by a set of models that formalize examples of concepts and relations with the user's intended semantics.

# References


- Grüninger, M., Hahmann. T., Hashemi, A., Ong, D., and Ozgovde, A. (2012) Modular first-order ontologies via repositories. *Applied Ontology* 7:169-209.
- Ong, D. and Grüninger, M. (2011), Constructing an Ontology Repository: A Case Study with Theories of Time Intervals, 5th International Workshop on Modular Ontologies, ESSLLI 2011, Ljubljana, Slovenia.
- Grüninger, M., Hahmann, T., Hashemi, A., and Ong, D. Ontology Verification with Repositories, Formal Ontologies and Information Systems 2010.
- Mossakowski, T., Codescu, M., Kutz, O., Lange, C., and Grüninger, M. (2013) Proof Support for Common Logic.

# Summary

- Ontology repositories can support ontology evaluation by exploiting the relationships between different ontologies.
- Techniques include – consistency-checking, entailment, relative interpretation, verification, and modularization.
- Automated theorem provers play a critical role in these techniques.

# Thanks to members of the Semantic Technologies Lab

- Ali Hashemi
- Torsten Hahmann
- Darren Ong
- Atalay Ozgovde
- Megan Katsumi
- Carmen Chui