A Methodology for the Development & Verification of Expressive Ontologies

Ontology Summit 2013 – Track B

Megan Katsumi

Semantic Technologies Laboratory
Department of Mechanical and Industrial Engineering
University of Toronto

January 24, 2013

Introduction

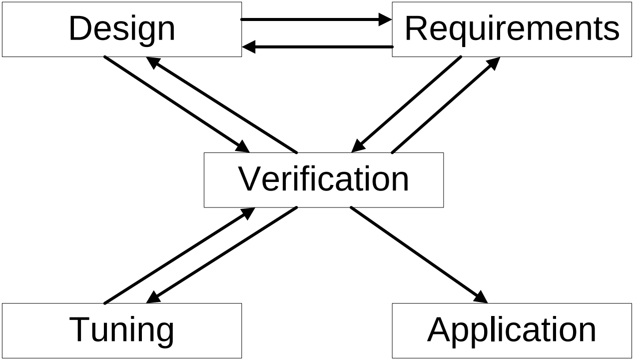
- A methodology for the development and verification of expressive ontologies
- Addresses the following extrinsic aspects of ontology evaluation:
  - Requirements and their verification
  - How evaluation can be used to revise requirements
  - How evaluation can be used to correct an ontology

Motivation

- Existing lifecycle methodologies for ontology development do not adequately address challenges that arise with the development of ontologies in full First-Order Logic (FOL), specifically:
  - Expressiveness of requirements
    - Consistency-checking is not enough!
  - Verification guidance required
    - How do we continue when we are unable to verify a requirement?

Challenges Addressed

We proposed a development methodology for the design and verification of expressive ontologies:



Design

Requirements

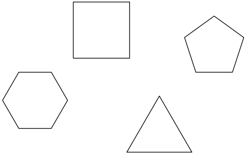
Verification

Tuning

Application

The BoxWorld Ontology

- Used to describe 2-dimensional and 3-dimensional shapes
- Applications in computer vision, manufacturing (e.g. sheet metal)
- Relations:
  - $point(p)$
  - $edge(e)$
  - $surface(s)$
  - $part(p, e)$
  - $meet(e_1, e_2, p)$
- Consider $T_{surface}$, fragment of $T_{BoxWorld}$ describing only 2-dimensional shapes



Prototype Design

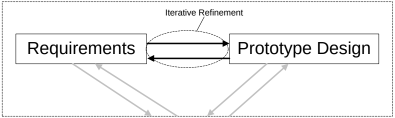


Figure: Developing the ontology *and* our understanding of its requirements

- Prototype Design: draft axioms (from scratch, or via reuse)
- Model Exploration: generate and review resulting models
    - Undirected
    - Directed
- Iterative Refinement: revise prototype and / or informal requirements

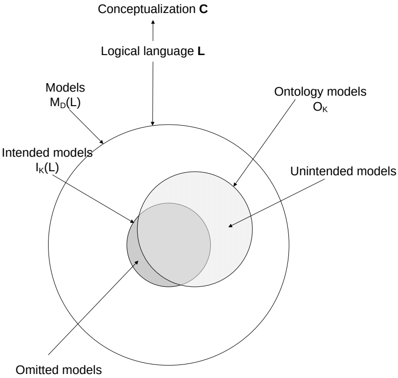
Requirements: Intended Models

From informal requirements to semantic requirements

Figure: Intended and unintended models

Requirements: Intended Models

The relationship between the intended models for an ontology and the actual models of its axioms, reproduced from Guarino (2009)



Requirements: Intended Models

Definition

Semantic requirements specify conditions on the intended models for the ontology, and/or models of the ontology's axioms.

- There are two types of such conditions for semantic correctness:

$$\mathcal{M} \in Mod(T_{onto}) \Rightarrow \mathcal{M} \in \mathfrak{M}^{onto}$$

and

$$\mathcal{M} \in \mathfrak{M}^{onto} \Rightarrow \mathcal{M} \in Mod(T_{onto})$$

Requirements: Semiautomatic Verification

$$\underbrace{\overbrace{T_{onto} \cup \Sigma_{domain}}^{Design} \models \overbrace{\Phi}^{Requirements}}_{Verification}$$

- The requirements are formulated as part of an entailment problem, this allows for the use of an automated theorem prover to evaluate the requirements
- In this way, verification is the process of using the theorem prover to evaluate if the requirements are entailed by the ontology

Requirements: Characterizing the Intended Models

- Challenge: recognize characteristics of the intended models
- For example, in the BoxWorld ontology: $\mathcal{M}$ is a model of $T_{surface}$ iff it is equivalent to a cyclic graph $G = (V, A)$ such that:
    - $V = \{\mathbf{e} \ : \ \langle \mathbf{e} \rangle \in \mathbf{edge}\}$
    - $A = \{(\mathbf{e_1}, \mathbf{e_2}) \ : \ \langle \mathbf{e_1}, \mathbf{e_2}, \mathbf{p} \rangle \in \mathbf{meet}\}$

Requirements: Partial Characterization

- Competency Questions: queries the ontology must be able to entail
- Can be used to specify:
    - Required properties, a partial characterization of the intended models
    - Necessary level of detail
    - Required performance with a theorem prover

Requirements: Competency Questions

- For example, in the BoxWorld ontology, for all 2-dimensional shapes, every edge meets exactly two distinct edges:
  - An edge cannot meet another edge at two distinct points

$$T_{surface} \models (\forall e_1, e_2, e_3, p_1, p_2) meet(e_1, e_2, p_1)$$

$$\wedge meet(e_1, e_3, p_2) \wedge \neg(p_1 = p_2) \supset \neg(e_2 = e_3)$$

  - Every edge meets at most two distinct edges

$$T_{surface} \models (\forall e_1, e_2, e_3, e_4, p_1, p_2, p_3) meet(e_1, e_2, p_1)$$

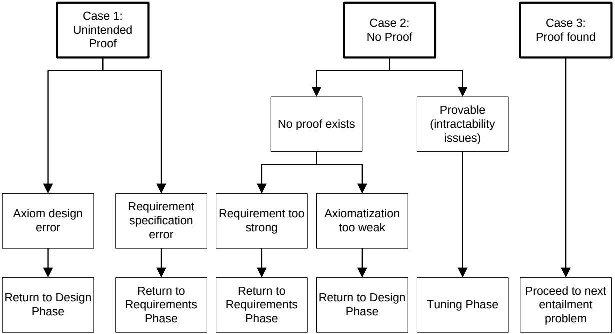
$$\wedge meet(e_1, e_3, p_2) \wedge meet(e_1, e_4, p_3) \supset$$

$$((e_2 = e_3) \wedge (p_1 = p_2)) \vee ((e_2 = e_4) \wedge (p_1 = p_3))$$

$$\vee((e_3 = e_4) \wedge (p_2 = p_3))$$

Verification

Guidance for each possible outcome of verification:



Case 1: Unintended Proof

Case 2: No Proof

Case 3: Proof found

No proof exists

Provable (intractability issues)

Axiom design error

Requirement specification error

Requirement too strong

Axiomatization too weak

Return to Design Phase

Return to Requirements Phase

Return to Requirements Phase

Return to Design Phase

Tuning Phase

Proceed to next entailment problem

Verification Case 1: Unintended Proof

- An unintended proof indicates:
    - Error in the design of the axioms
    - Error in the specification of the requirement

Verification Case 1: Unintended Proof

- For example, we “proved” the competency question presented earlier: Every edge meets at most two distinct edges
- But, it was a proof that: $T_{surface} \models (\neg\exists p) point(p)$
- Examination of the proof showed that the axiom:

$$(\forall e, p_1, p_2, p_3) edge(e) \land point(p_1) \land point(p_2) \land point(p_3) \land part(p_1, e)$$

$$\land part(p_2, e) \land part(p_3, e) \supset (p_1 = p_3) \lor (p_2 = p_3)$$

was incorrect

Verification Case 1: Unintended Proof

- For example, we "proved" the competency question presented earlier: Every edge meets at most two distinct edges
- But, it was a proof that: $T_{surface} \models (\neg \exists p) point(p)$
- Examination of the proof showed that the axiom:

$$(\forall e, p_1, p_2, p_3) edge(e) \land point(p_1) \land point(p_2) \land point(p_3) \land part(p_1, e)$$

$$\land part(p_2, e) \land part(p_3, e) \supset (p_1 = p_3) \lor (p_2 = p_3) \lor (\mathbf{p_1} = \mathbf{p_2})$$

was incorrect

Verification Case 2: No Proof Found

- Failure to find a proof indicates:
    - Case 2A: No proof exists
        - Requirement is too strong
        - Ontology is too weak
    - Case 2B: Provable
        - Requirement is provable, but the theorem prover is having difficulties producing the proof
- Generating counterexamples can identify Case 2A, but sometimes the cause is ambiguous

Verification Case 2: No Proof Found

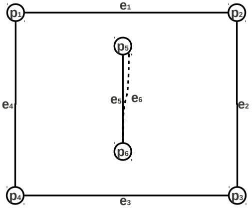
- For example, we could not find a proof of the competency question that an edge cannot meet another edge at two distinct points:

$$(\forall e_1, e_2, e_3, p_1, p_2) meet(e_1, e_2, p_1) \wedge meet(e_1, e_3, p_2)$$
$$\wedge \neg (p_1 = p_2) \supset \neg (e_2 = e_3)$$

- Intuitions and knowledge of development history help determine the cause:
    - Poor theorem prover performance? Or not provable?

Verification Case 2A: Not Provable

- In this instance, no proof existed



- A design decision:
  - Relax the requirement?
  - Strengthen the axioms?

Verification Case 2B: Provable

- Tuning: The addition of lemmas or the use of subsets of the ontology in order to improve theorem prover performance
    - Lemmas - in the traditional mathematical sense; some consequence of a theory (ontology) that can be used to help prove some goal (requirement)
    - Subsets - large ontologies may slow theorem prover performance; reasoning with a subset of the ontology's axioms may improve performance enough to prove a particularly challenging requirement

Summary

- Expressiveness of requirements
  - We proposed a lifecycle to support the development of expressive ontologies, employing automated reasoners for a rigorous specification and semiautomatic verification of semantic requirements.
- Verification guidance
  - We provided pragmatic guidance for the development phases, addressing all possible outcomes of theorem prover verification, including ambiguous timeouts (intractability or semidecidability?).
- This methodology has been effectively used with ontologies for sheet metal manufacturing and PSL (ISO18629).

Future Work

- Incorporate consideration of other ontology development issues such as quality, requirements validation, etc.
- Address the challenge of model exploration for iterative refinement in cases where the ontology has only infinite models
- Include more specific guidance on how to leverage system use cases (when appropriate) to identify semantic requirements

References

- Nicola Guarino, Daniel Oberle, and Steven Staab. *What is an Ontology?*, pages 1–17. Springer-Verlag, Berlin, 2nd edition, 2009.
- Megan Katsumi. A methodology for the development and verification of expressive ontologies. M.Sc. thesis, Department of Mechanical and Industrial Engineering, University of Toronto, 2011.
- Megan Katsumi and Michael Grüninger. Theorem proving in the ontology lifecycle. In *Proceedings of the International Conference on Knowledge Engineering and Ontology Development*, 2010.