



Structure-Based Partitioning of Large Concept Hierarchies

Heiner Stuckenschmidt, Michel Klein
Vrije Universiteit Amsterdam

Outline

- Motivation: The Case for Ontology Partitioning
 - ← Lots of Pictures
- A Partitioning Method
 - Create a dependency graph
 - Determine the strength of dependencies
 - Compute Partitioning
 - Improve Partitioning
- Experiments

Ontologies and the Semantic Web

- Ontologies are the backbone of semantic web applications
 - Content-Based Retrieval
 - Information Integration
 - Web Service Discovery
- More and More Large Ontologies become available
 - General Purpose: Open Directory, Yahoo!, ...
 - Medicine: GALEN, UMLS, FMA, ...

The Case for Partitioning

- Distributed Development and Maintenance
 - Experts can update their portion independently of other parts
- Selective Publication and Use of Terminologies
 - Stable subsets can be published in the development phase
 - Users can chose relevant subset of an ontology

An Abstract View of the Problem

- Despite the standardization of Languages there is no agreement on the way ontologies are represented.
 - All ontologies contain **classes**
 - Most organize them in a **hierarchy**
 - Many define **relations** between classes
 - Some provide formal **definitions** of classes
- We concentrate on partitioning ontologies

Overview of the Process

1. Create Dependency Graph
2. Determine Strength of Dependencies
3. Compute Partitions

Overview of the Process

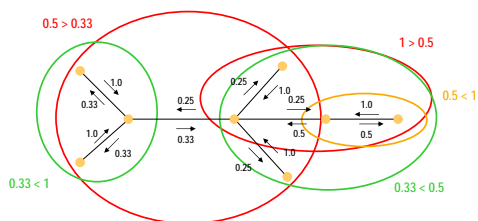
1. Create Dependency Graph
2. Determine Strength of Dependencies
3. Compute Partitions

Computing Islands

- We use maximal **line islands** [Batage] 2000] to compute partitions in the dependency graph.
 - A set of vertices is a line island in network if and only if it induces a connected subgraph

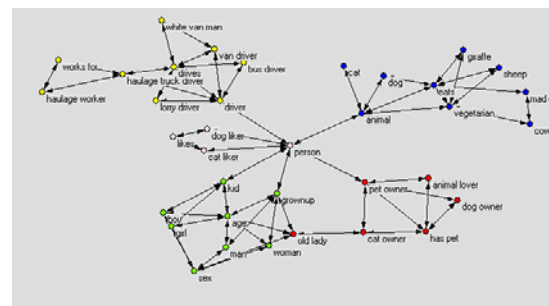
$$\max_{\{(v,v') \in D\} \cap (v \in I \wedge v' \notin I) \vee (v' \in I \wedge v \notin I)} w(v,v') < \min_{(u,u') \in E_T} w(u,u')$$
 related among them than with the neighboring vertices. In particular there is a maximal spanning tree T over nodes in the island such that:

Understanding Islands



$$\max_{\{(v,v') \in D\} \cap (v \in I \wedge v' \notin I) \vee (v' \in I \wedge v \notin I)} w(v,v') < \min_{(u,u') \in E_T} w(u,u')$$

Result for the Example

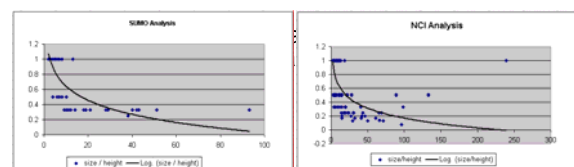


Overview of the Process

1. Create Dependency Graph
2. Determine Strength of Dependencies
3. Compute Partitions

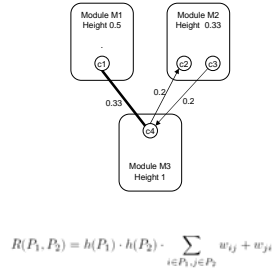
Improving Partitions (work in progress)

- Islands are often very small (2 - 4 nodes) resulting in unwanted partitions of the ontology.



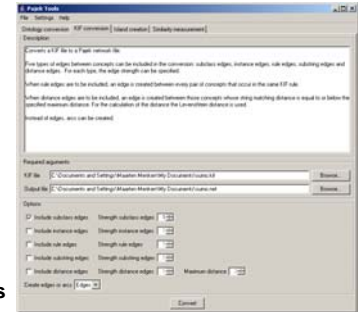
Improving Partitions

- Islands are often very small (2 - 4 nodes) resulting in unwanted partitions of the ontology.
- Observation: small islands almost always have a large height value (1 or 0.5).



Ontology Partitioning Tool

- Features:**
 - OWL and KIF Import
 - Selection of Criteria
 - Computation of line islands



Experiments

- The ACM Computer Science Classification
 - 1000 concepts, fixed hierarchy
- The Standard Upper Model Ontology SUMO
 - 600 concepts, fixed hierarchy
- The NCI cancer ontology
 - Subset with 2400 concepts, fixed hierarchy
- The DICE ontology
 - About 2000 concepts, classified hierarchy

Distributed Representation and Reasoning

Based on Work of:

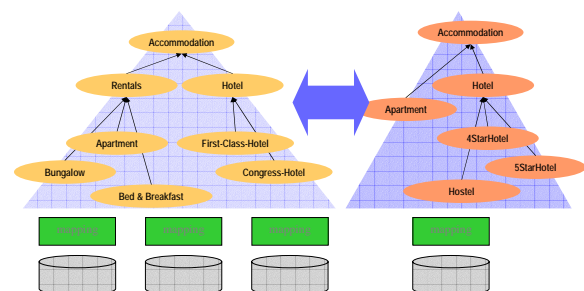
Alex Borgida
Paolo Bouquet
Fausto Giunchiglia
Frank van Harmelen
Luciano Serafini
Heiner Stuckenschmidt
Andrei Tamalin
Holger Wache

Distributed Representation and Reasoning

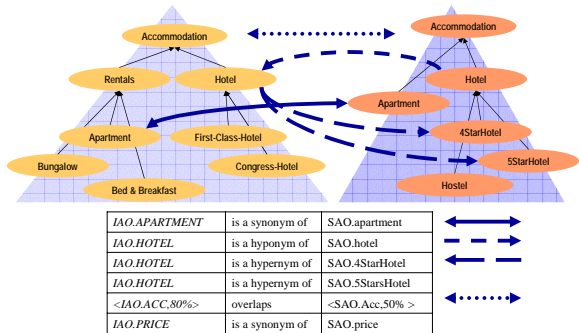
Based on Work of:

Alex Borgida
Paolo Bouquet
Fausto Giunchiglia
Frank van Harmelen
Luciano Serafini
Heiner Stuckenschmidt
Andrei Tamalin
Holger Wache

Ontologies



Ontology Mappings

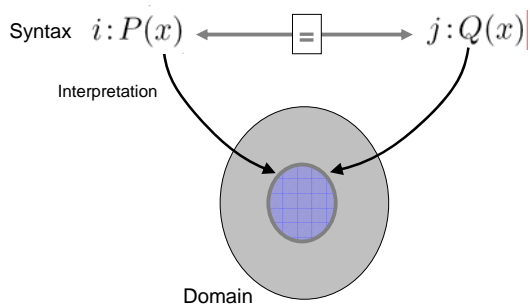


Problem: Semantics of Mappings

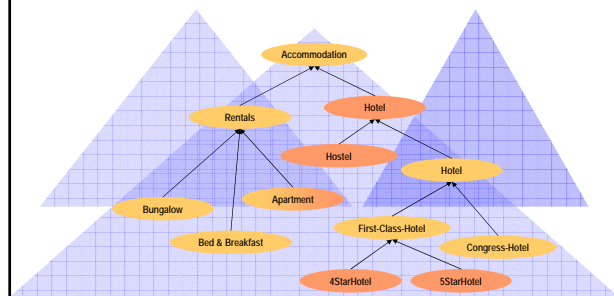
<i>IAO.APARTMENT</i>	is a synonym of	SAO.apartment
<i>IAO.HOTEL</i>	is a hyponym of	SAO.hotel
<i>IAO.HOTEL</i>	is a hypernym of	SAO.4StarHotel
<i>IAO.HOTEL</i>	is a hypernym of	SAO.5StarsHotel
< <i>IAO.ACC,80%</i> >	overlaps	<SAO.Acc,50% >
<i>IAO.PRICE</i>	is a synonym of	SAO.price

- What is the semantic of this mapping?
- Share different approaches the semantic?
- If not where are the differences?

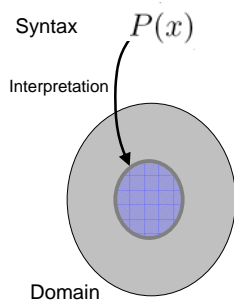
The Logic of Mappings



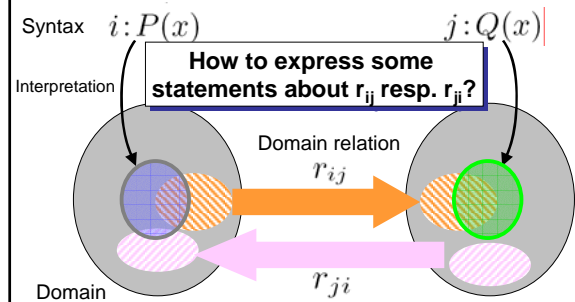
Global Interpretation



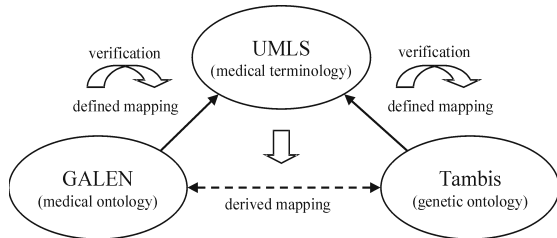
An Alternative Semantics



Local Domain Semantics



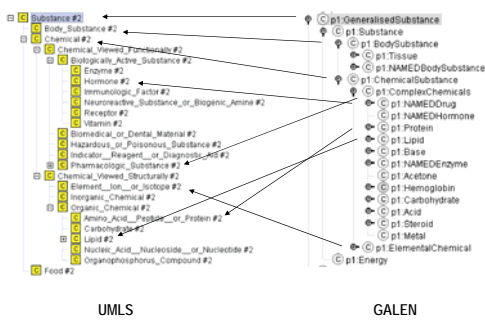
Goal: Reasoning with and about mappings



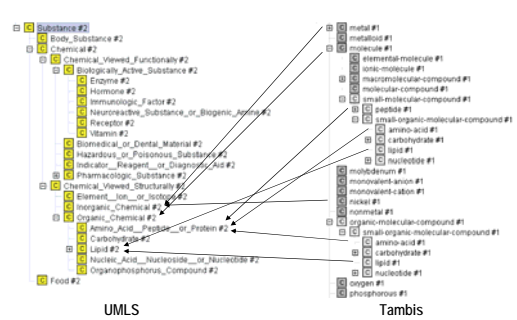
The Case Study:

- Select three topics with sufficient overlap
 - Substances
 - Structures
 - Processes
- Analyse Subhierarchies in the different Terminologies
- Define partial ad-hoc mappings between individual concepts
- Use semantics to verify and complete mappings

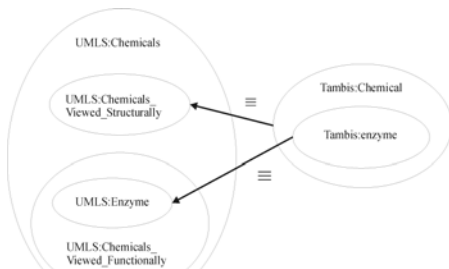
Example: Substances I



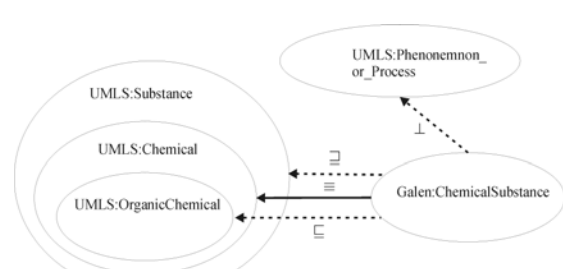
Example: Substances II



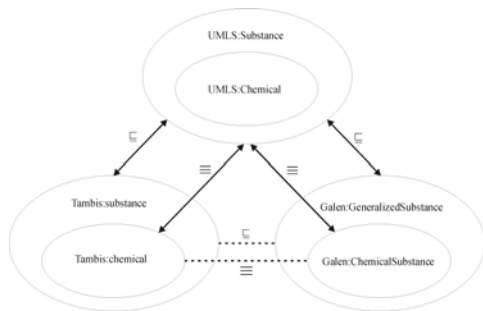
Verification of Mappings



Inferring new Mappings



Reasoning using Mappings



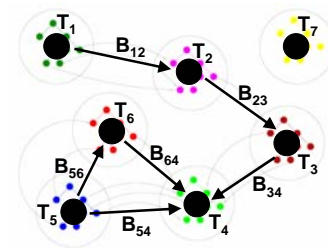
The formal Basis: Distributed Description Logics

DDL syntax

- **DDL** is a family of description logics $\{DL_i\}_{i \in I}$
- A **bridge rule** from i to j is an expression of the form:
 - $i:X \xrightarrow{\sqsubseteq} j:Y$ (into-bridge rule)
 - $i:X \xrightarrow{\sqsupseteq} j:Y$ (onto-bridge rule)
 where X and Y are concepts of DL_i and DL_j .
- A **distributed T-box (DTB)** is a pair $\langle \{T_i\}_{i \in I}, \{B_{ij}\}_{i \neq j \in I} \rangle$ where B_{ij} is a collection of bridge rules from i to j

Bridge graph

A **bridge graph** of a DTB



DDL semantics

A **distributed interpretation (DI)** of a DTB

$$\langle \{I_i\}_{i \in I}, \{r_{ij}\}_{i \neq j \in I} \rangle$$

- I_i is a **local interpretation** of T_i on a **local domain** Δ^{I_i}
 - $T_1, T_2, T_3, T_4, T_5, T_6, T_7$
 - $I_1, I_2, I_3, I_4, I_5, I_6, I_7$
- r_{ij} is a **domain relations** from I to j

$$r_{ij} \subseteq \Delta^{I_i} \times \Delta^{I_j}$$

DDL satisfiability

$$DI = \langle \{I_i\}_{i \in I}, \{r_{ij}\}_{i \neq j \in I} \rangle \text{ satisfies DTB} = \langle \{T_i\}_{i \in I}, \{B_{ij}\}_{i \neq j \in I} \rangle$$

$$DI \models \text{DTB}$$

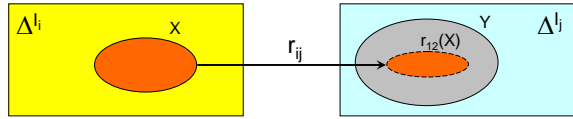
If

- all T_i are satisfied
- all bridge rules B_{ij} are satisfied

Into-bridge rule satisfiability

$$DI \models i:X \xrightarrow{\sqsubseteq} j:Y$$

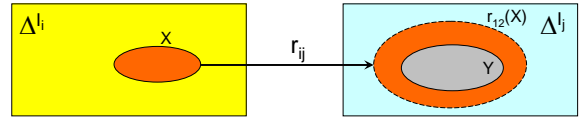
$$r_{ij}(x^i) \subseteq Y^j$$



Onto-bridge rule satisfiability

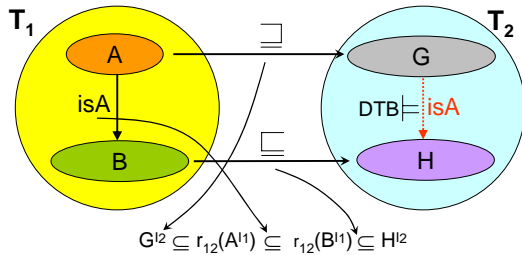
$$DI \models i:X \xrightarrow{\supseteq} j:Y$$

$$r_{ij}(x^i) \supseteq Y^j$$



Subsumption propagation in DDL

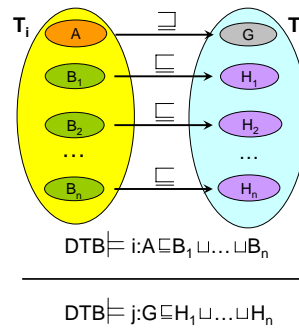
$$DTB = \langle T_1, T_2, B_{12} \rangle$$



Directionality property:
Knowledge propagates **ONLY** along the direction of bridge rules!

Generalized subsumption propagation

$$DTB = \langle \{T_i\}_{i \in I}, \{B_{ij}\}_{i,j \in I} \rangle$$



Soundness and completeness

Let $DTB_{12} = \langle T_1, T_2, B_{12} \rangle$ be a distributed T-box

Bridge operator encapsulates propagated axioms

$$B_{12}(T_1) = \left\{ \left. \begin{array}{l} G \sqsubseteq \bigsqcup_{k=1}^n H_k \\ T_1 \models A \sqsubseteq \bigsqcup_{k=1}^n B_k \\ 1:A \xrightarrow{\supseteq} 2:G \in B_{12} \\ 1:B_k \xrightarrow{\sqsubseteq} 2:H_k \in B_{12} \\ \text{for } 1 \leq k \leq n, n \geq 0 \end{array} \right\}$$

Theorem

$$DTB_{12} \models 2:X \sqsubseteq Y \Leftrightarrow T_2 \cup B_{12}(T_1) \models X \sqsubseteq Y$$

Distributed tableau algorithm

Basic reasoning service of DDL

$$\boxed{\text{DTB} \models i:C \sqsubseteq D} \quad \boxed{\text{DTB} \models i:X}$$

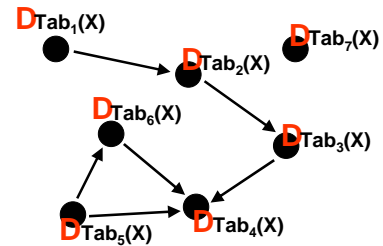
$i:X$ is **satisfiable** with respect to DTB

- if there exist a DI such that $DI \models \text{DTB}$
- and $X^i \neq 0$

Restrictions:

- (1) bridge graph is cycle-free
- (2) bridge rules connect atomic concepts
- (3) no nominals

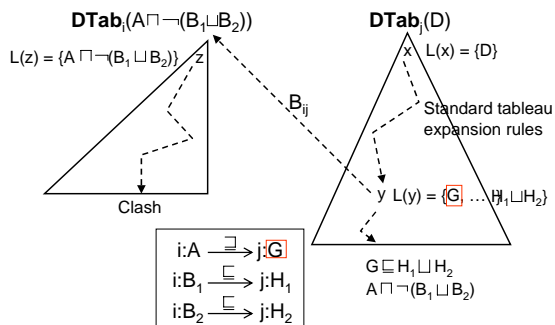
Distributed tableau intuition



$$\mathbf{DTab}_i(X) = \mathbf{Tab}_i(X) + \text{“lazy computation of bridge operator”}$$

Distributed tableau intuition

Is $j:D$ satisfiable wrt DTB?



Algorithm formalization

\mathbf{DTab}_j

- *SHIQ*-tableau expansion rules +
- “bridge” expansion rule:

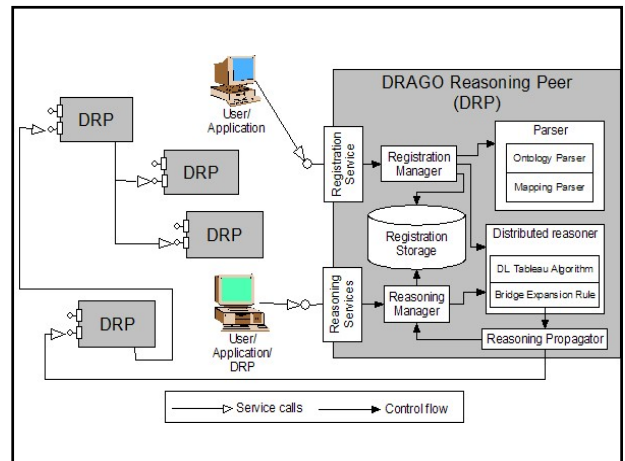
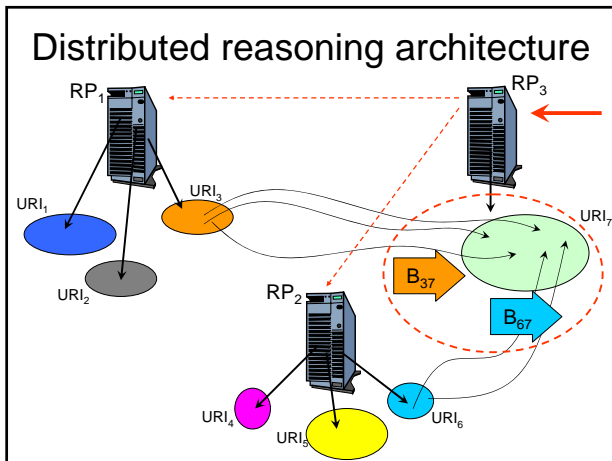
If 1. $G \in L(x)$, $i:A \sqsupset \rightarrow j:G \in B_{ij}$, and
 2. $\mathbf{B} \subseteq \{\mathbf{B} \mid i:B \sqsubseteq \rightarrow j:H \in B_{ij}\}$, and
 $\mathbf{H} \subseteq \{\mathbf{H} \mid i:B \sqsubseteq \rightarrow j:H \in B_{ij}\}$, and
 $\mathbf{DTab}_i(A \sqcap \neg \sqcup \mathbf{B}) = \text{Unsatisfiable}$ for some $\mathbf{H} \notin L(x)$,
 then
 $L(x) \rightarrow L(x) \cup \{\sqcup \mathbf{H}\}$

Algorithm properties

Theorem (Termination) For any acyclic distributed T-box and for any *SHIQ* concept X , $\mathbf{DTab}_i(X)$ terminates.

Theorem (Soundness and completeness)
 $j:X$ is satisfiable in distributed T-box if and only if $\mathbf{DTab}_i(X)$ can generate a complete and clash-free completion tree.

DRAGO reasoning architecture



- ### Implementation
- OWL ontologies
 - C-OWL semantic mappings
 - Distributed Reasoner is an extension to open source OWL Reasoner Pellet