

Relational Data Mining through Propositionalization and Subsequent Propositional Learning for Semantic Virtual Engineering

**Monika Žáková¹, Filip Železný¹, Petr Křemen¹,
Cyril Masia-Tissot² and Nada Lavrač³**

¹ Czech Technical University in Prague, Czech Republic

²Semantic Systems, Derio, Spain

³Jozef Stefan Institute, Ljubjana, Slovenia

1. Motivation
2. Annotation of CAD designs
3. ILP background
4. RDM system
 - Sorted Logic
 - Feature construction
 - Adaptation of rule learning
5. Experiments and results
6. RDM results management
7. Conclusions and future work

- **Engineering is one of the most knowledge-intensive activities**
- **Knowledge in form of CAD designs, documents, simulation models and ERP data bases**
- **No industrial software employing ILP techniques in real-life regular use we are aware of**
- **Goal: Making implicit knowledge contained in CAD designs explicit useful for reuse, training, quality control**



Design Annotation

- **the information available in CAD files and other data sources formalized and integrated by means of semantic annotation based on ontologies**
- **semantic annotation of CAD designs**
 - generated automatically from the commands history available via the API of CAD tools
 - based on a CAD ontology developed in SEVENPRO
 - available in RDF format



Annotation Example - RDFS

```
<sp_cad:Body rdf:about="&sp_cad;Body_22083581184246506">
  <rdfs:label>Redondeo4</rdfs:label>
  <sp_cad:hasFeature>
    <sp_cad:SolidExtrude rdf:about="&sp_cad;SolidExtrude_22083591184246507"/>
  </sp_cad:hasFeature>
  <sp_cad:hasFeature>
    <sp_cad:SolidPocket rdf:about="&sp_cad;SolidPocket_22083621184246509"/>
  </sp_cad:hasFeature>
</sp_cad:Body>
<sp_cad:SolidPocket rdf:about="&sp_cad;SolidPocket_22083621184246509">
  <rdfs:label>Cortar-Extruir4</rdfs:label>
  <sp_cad:hasLimit2>
    <sp_cad:OffsetLimit rdf:about="&sp_cad;OffsetLimit_22083631184246509"/>
  </sp_cad:hasLimit2>
  <sp_cad:hasLimit1>
    <sp_cad:OffsetLimit rdf:about="&sp_cad;OffsetLimit_22083641184246510"/>
  </sp_cad:hasLimit1>
</sp_cad:SolidPocket>
```



ILP Background

- **Inductive logic programming (ILP) aims at learning a theory in a subset of first-order logic from given examples, taking background knowledge into account**
- **Traditional ILP setting cannot exploit explicit taxonomies on concepts and terms**
- **Our aim: exploiting taxonomies in the framework of propositionalization and subsequent learning from the propositionalized representation**

The CAD ontology declares a concept **PrismSolFeature** and its subconcept **SolidExtrude**. It is possible to declare in background knowledge e.g.

```
subclass(prismSolFeature, solidExtrude).  
hasFeature(B, F1):-hasFeature(B,F2),subclassTC(F1,F2).
```

Unfortunately, in such an approach, for the following two exemplary clauses (hypotheses)

```
C = itemFamilyLiner(P):-hasBody(P,B),hasFeature(B, prismSolFeature).  
D = itemFamilyLiner(P):-hasBody(P,B),hasFeature(B, solidExtrude).
```

it does not hold $C \theta \subseteq D$, so clause D is not obtained by applying a specialization refinement operator onto clause C .

- A **sorted variable** is a pair $x:\tau$
 - where x is a variable name
 - τ is a sort symbol, which denotes a subset of the domain called a sort
- A **sort theory** is a finite set of formulas containing function formulas and subsort formulas
 - function formula $\forall x_1, \dots, x_n \tau_1(x_1) \wedge \dots \wedge \tau_n(x_n) \rightarrow \tau(f(x_1, \dots, x_n))$
 - subsort formula $\forall x \tau_1(x) \rightarrow \tau_2(x)$
- It is required that the directed graph corresponding to the sort theory is acyclic and has a single root
- For a sort theory Σ , a Σ -sorted substitution is a mapping from variables to terms such that for every variable $x:\tau$, it holds that $\Sigma \models \forall x \tau(t)$, where t is $(x:\tau)\theta$ and θ is the sorted substitution



RDM Core Overview

Examples

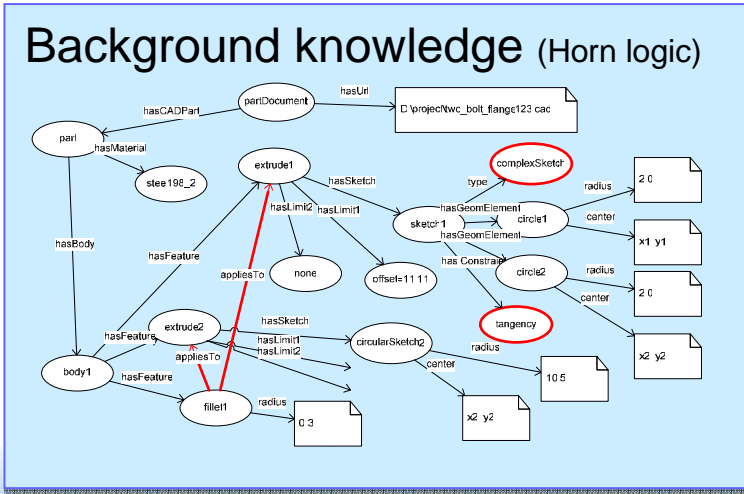
```
eItem(eItemT_BA1341).  
eItem(eItemT_BA1342).  
eItem(eItemT_BA1343).
```

Predicate declarations

```
mode hasBody(+CADPart, -Body).  
mode hasMaterial(+CADPart, -Material).  
mode hasSketch(+CADPart, -Sketch).  
mode hasLength(+Sketch, -float).
```

Sort theory

```
subClassOf(CADPart, CADEntity).  
subClassOf(CADAssembly, CADEntity).  
...  
subPropertyOf(hasCircularSketch, hasSketch).  
subPropertyOf(firstFeature, hasFeature).
```



Feature construction

Propositional rule learning (Weka)

Propositional rule learning (adapted)

Feature subsumption table

Subsumption and exclusion matrix

Downward Δ, Σ -refinement

- extension of sorted refinement proposed by Frisch
- defined using 3 refinement rules:

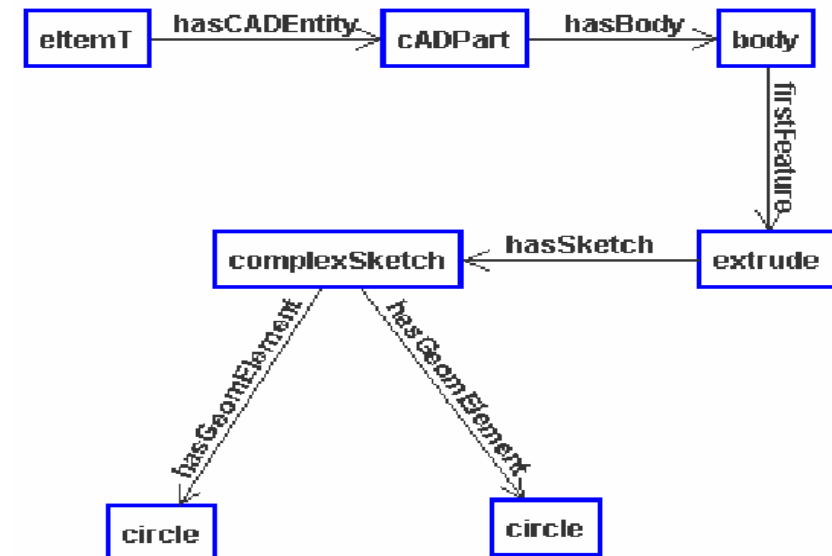
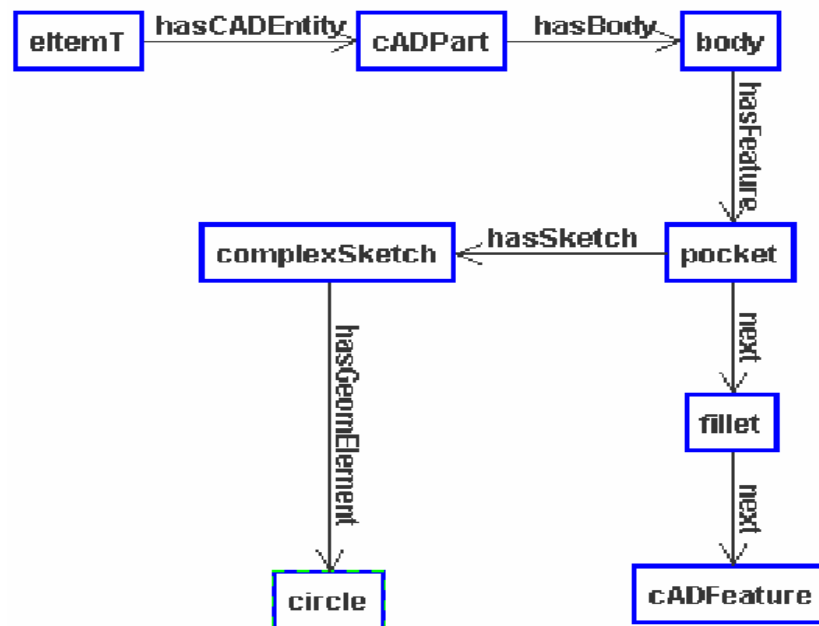
1. adding a literal to the conjunction

2. replacing a sort with $pred_1(x_1:T_1, \dots, x_n:T_n)$ with one of its direct subsorts $pred_1(x_1:T_1', \dots, x_n:T_n)$

3. replacing a literal $pred_1(x_1:T_1, \dots, x_n:T_n)$ with one of its direct subrelations $pred_2(x_1:T_1, \dots, x_n:T_n)$



Examples of Generated Features



- information about feature subsumption hierarchy stored and passed to the propositional learner
- assume that features f_1, \dots, f_n have been generated with corresponding conjunctive bodies b_1, \dots, b_n
- *elementary subsumption matrix* E of n rows and n columns is defined such that $E_{i,j} = 1$ whenever $b_i \times \rho_{\Delta, \Sigma}(b_j)$ and $E_{i,j} = 0$ otherwise
- *exclusion matrix* X of n rows and n columns is defined such that $X_{i,j} = 1$ whenever $i = j$ or $b_i \times \rho_{\Delta, \Sigma}(\rho_{\Delta, \Sigma}(\dots \rho_{\Delta, \Sigma}(b_j) \dots))$ and $X_{i,j} = 0$ otherwise.



Propositional Rule Learning

2 propositional algorithms adapted to accept elementary subsumption and exclusion matrix

1. Top-down deterministic algorithm

2. Stochastic local DNF algorithm



Top-down deterministic algorithm

- **stems from the rule inducer of RSD**
- **based on**
 - a heuristic general-to-specific beam search for the induction of a single rule for a given target class
 - and a cover-set wrapper for the induction of the entire rule set for the class
- **using matrices E , X it can**
 - prevent the combination of a feature and its subsumee within the conjunction
 - specialize a conjunction by replacing a feature with its direct subsumee



Stochastic Local DNF Search Algorithm

- algorithm introduced in Rückert 2003 and later transferred into the propositionalization framework by Paes 2006
- conducts search in the space of DNF formulas i.e. refines entire propositional rule sets
- refinement done by local non-deterministic DNF term changes
- we use matrix X to prevent combination of a feature with its subsumee within a DNF term

experiments performed to assess

1. runtime impact of the extended sorted refinement operator in propositionalization
2. exploitation of the explicit feature-taxonomy in subsequent propositional learning
3. accuracy of classification by standard propositional algorithm using propositional features



Dataset Description

- semantic annotations of command histories of 160 design drawings, generated automatically using CAD Annotator
- annotations of individual examples and the CAD ontology in RDFS format
- classification of examples given by the belongsToFamily relation defined Item ontology
- examples classified into 4 proper classes describing families of designs (57 examples that did not belong to any of the 4 classes were classified as 'other').

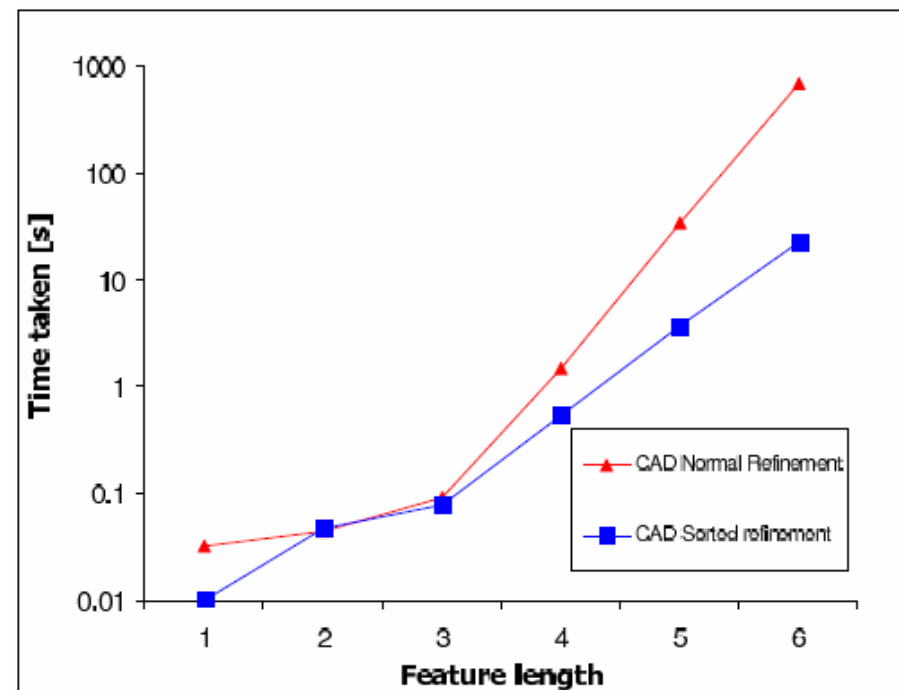
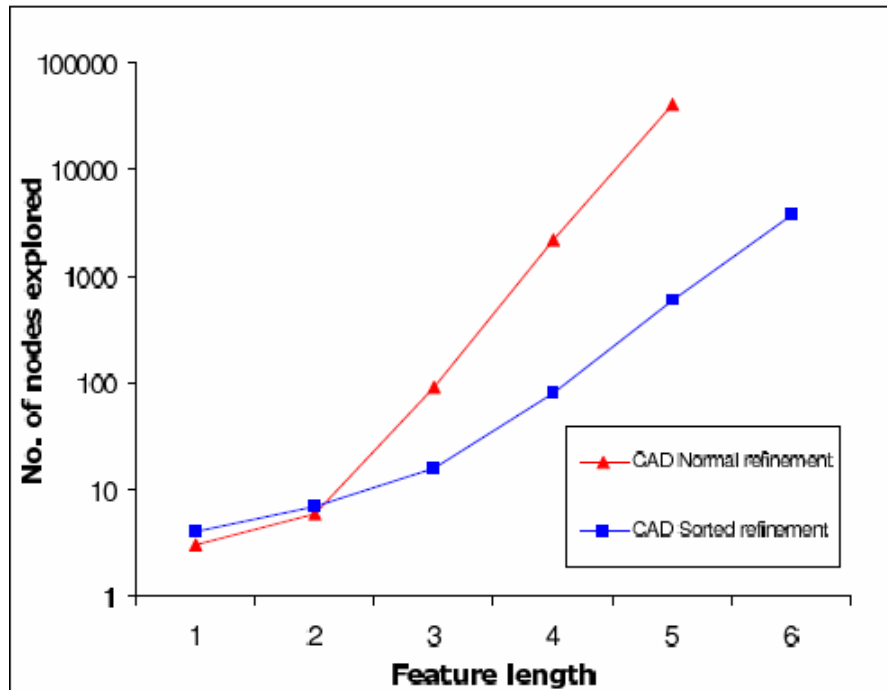


Additional Preprocessing

- additional important information (from consultation with users) : the first feature used and relative order of the features
- properties **next**, **sequenceStart** and **firstFeature** describing the order of CAD features added to the CAD ontology
- relations added to the background knowledge:
 - **subpropertyOf(firstFeature,hasFeature)**,
 - **subpropertyOf(hasFeature,sequenceStart)**.
- special treatment of relations, which are subproperties of **next** and **sequenceStart** implemented



Sorted Refinement vs. Normal Refinement





Propositional Learning Results

Algorithm	CAD data	
	Time taken [s]	Predictive accuracy
Top-down	0.22 ± 0.08	0.66 ± 0.21
Top-down with feat. taxonomy	0.06 ± 0.02	0.66 ± 0.22
SLS	0.63 ± 1.45	0.62 ± 0.18
SLS with feature taxonomy	0.28 ± 0.83	0.61 ± 0.19



Classification Results

- **Classification performed with J48 decision tree induction algorithm implemented in Weka**

Class	Prec.	Recall	F-Measure
itemFamilyTT	0.792	0.826	0.809
itemFamilyLiner	0.879	0.895	0.887
itemFamilyStdPlate	0.571	0.5	0.533
itemFamilySlottedPlate	0.727	0.8	0.762
other	0.883	0.855	0.869



RDM Results Management

- **Framework for storing and management of RDM results required due to their large amount and diversity**
- **RDM ontology is being developed providing**
 - The logical model of the RDM knowledge base
 - An interface between the RDM system and semantic server
- **Ontology designed w.r.t. 2 types of queries**
 - End-user requirements e.g. finding several classification rules with the highest confidence for the given example
 - Supporting RDM feedback and algorithm tuning e.g. metric evaluation for clustering algorithms



Ongoing and Future Work

- **extend the scope of meta-information exploitable by refinement operators beyond taxonomic information**
- **e.g. to deal with meta-knowledge such as “relation R is a function” or “binary relation R is symmetrical,” etc.**
- **exploring the semantic subsumption operator**
- **developing RDM ontology**