

Pelletheart

A prototype of a hybrid rule reasoner for ontologies.

Description

Integrating classic forward chaining rule reasoning implemented by [HeaRT](#) with the [Pellet](#).

Concept by [G. J. Nalepa](#) and [W. T. Adrian](#) (de domo Furmańska), prototype implementation by W. T. Adrian.

Idea

- **Conceptual level:** Integration of *Attribute Logic with Set Values over Finite Domains* (ALSV(FD)) and *Description Logics (DL)* (research papers on integration: [here](#) and [here](#))
- **Implementation level:** Integration of Pellet ontology reasoner and HeaRT rule inference engine (research paper on architecture proposal: [here](#), poster:



Integration Proposal

- Attributes in AL correspond to Concepts in DL
- model of a system stored in HeaRT, rule conditions checked by Pellet, execution of rules by HeaRT
- communication: DIG or command line

Implementation

Top-down overview

There are 2 aspects of the integration of Pellet and HeaRT:

1. Communication channel
 1. command line
 1. sending RDF/XML (dedicated translators from HML/R to RDF/XML)
 2. DIG interface
 1. sending DIG message (dedicated translators HML/R to DIG)
2. Inference scenario
 1. rule precondition checked with *consistency checking* DL task
 2. rule precondition checked with *realisation* DL task

What has been done

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Reasoning with HearT-Pellet:                                     %
% 1. Build TBox: definitions of types and attributes               %
%   a) build additional statements: 'allDifferent' for individuals %
% 2. Call any inference mode you wish (GDI, TDI etc.)           %
% 3. In each state build an ABox representing this state         %
% 4. Whenever you check a rule preconditions:                    %
%   a) build rule axioms (temporary TBox),                      %
%   b) ontology = definitions TBox + rule axioms TBox + state ABox %
%   c) send the ontology to Pellet to check its consistency      %
% 5. Interpret the result, carry on as usual                     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Reasoning with HearT-Pellet - Alternative version:             %
%                                                                 %
% 1. Build TBox: definitions of types and attributes             %
%   a) build additional statements: 'allDifferent' for individuals %
% 2. Call any inference mode you wish                           %
% 3. In each state build an ABox representing this state         %
% 4. Whenever you check a rule preconditions:                    %
%   a) build rule axioms (ABox statements)                      %
%   b) definitions TBox + rule axioms aBox + state ABox         %
%   c) send to Pellet to check realization the rule conditions   %
% 5. Interpret the result, carry on as usual                     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

Following the inference scenario:

1. HearT can be started with additional parameter in the gox predicate:

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% gox(StateIn,T,Mode,ExternalReasoner)                           %
%   ExternalReasoner - the one used to check the rules preconditions %
%                                                                 %
% Augmented version of Heart - additional variable for an ext.reasoner %
%                                                                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

2. Translating parts of the HMR model to RDF/XML

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% DAAL prototype translator.
% -----
%
% Translates attributes into concepts,
% attribute values into instances and
% rule preconditions into T-Box like axioms.
%
% Supported operators: 'in', 'eq'.
% Supported attributes: 'symbolic'.

```

```
%
% Basic predicates:
% owl_xml_gen/0 - translate HMR file into DAAL representation (ontology).
% owl_xml_gen/1 - translate HMR file into DAAL representation (ontology)
%                   and write it to the file given as the argument.
% owl_attr_gen/0 - translate the attribute definitions
% owl_rulp_gen/0 - translate the rules preconditions
% owl_rulp_gen/1 - translate the given rule preconditions
% owl_stat_gen/0 - translate the states statements
% owl_stat_gen/1 - translate the given state statement
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

3. Sending partial ontologies from HeaRT to Pellet

4. Interpreting the Pellet answers by HeaRT

Technically

- `heart-pellet.pl` - Extended version of HeaRT (works with the standard HeaRT distribution): additional parameter in `gox` predicates for the external reasoner to use
- `heart-daal-translator.pl` - predicates

Papers

- G.J. Nalepa, W.T. Furmańska: [Proposal of a New Rule-Based Inference Scheme for the Semantic Web Applications](#), New Challenges in Computational Collective Intelligence. Studies in Computational Intelligence, 2009, Vol. 244/2009, 15-26.
- G.J. Nalepa, W.T. Furmańska: [Pellet-HeaRT - Proposal of an Architecture for Ontology Systems with Rules](#), KI 2010: Advances in Artificial Intelligence. LNCS, Vol. 6359/2010, 143-150.
- G.J. Nalepa, W.T. Furmańska: [Integration Proposal for Description Logic and Attributive Logic - Towards Semantic Web Rules](#), TRANSACTIONS ON COMPUTATIONAL COLLECTIVE INTELLIGENCE II, LNCS, Vol. 6450/2010, 1-23.

Releases

See us back on spring 2011

Comments

Support by [W.T.Adrian](#).

software Semantic_Web ontologies rules

Go back to → [software](#)

From:

<https://www.geist.re/> - **GEIST Research Group**

Permanent link:

<https://www.geist.re/pub:software:pelletheart?rev=1304357039>

Last update: **2011/05/02 17:23**

