Design and Implementation of OO jDREW

Marcel A. Ball
**OO jDREW**

- Reasoning engine that was implemented. Includes:
  - Positional Prolog style logic
  - OO features from RuleML
  - Slots, Order-Sorted Types, oids
  - Implements a subset of the SWRL built-ins

- Based to some extent upon Bruce Spencer’s jDREW reasoning engine ([www.jdrew.org](http://www.jdrew.org))
Keyed Parameters (Slots)

- Allows for a non-positional knowledge representation
- Useful for representing RDF descriptions

In a positional knowledge representation

- Multiple possible interpretations
- Ordering of arguments is important

father(henry, george).
Keyed Parameters (Slots)

- Slotted version is unambiguous
- Order is no longer important

father(parent->Henry; child->George).

- Canonical order is imposed internally to make unification efficient
Keyed Parameters & Rest Variables

**Fact**

```
person(name->Henry; sex->male; ...; age->22).
```

**Queries**

- **Fails**
  
  ```
  person(name->?name; age->22).
  ```

- **Succeeds!**
  
  ```
  person(age->22; name->?name !?rest).
  ```
Order-Sorted Types

- Type sorts encoded in RDF Schema
- Internally represented as a lattice
- Type operations computed using lattice algorithms
Order-Sorted Types Example

Facts:

1. **Vehicle type is “ToyotaCorolla”**
   - Inheritance path: ToyotaCorolla -> Sedan -> Car -> Vehicle -> Thing

2. **Succeeds!**
   - **Vehicle type:** Car
   - **Price:** 800.00
   - **Customer:** Joe
   - **Age:** 22
   - **Sex:** Male
   - **Year:** 2001
   - **Model:** CE
   - **Color:** Blue

3. **Succeeds!**
   - **Vehicle type:** Van
   - **Price:** 650.00
   - **Customer:** Male

4. **Fails**
   - **Vehicle type:** Car
   - **Price:** 600.00
   - **Customer:** Male

Query:

- **Vehicle type is “ToyotaCorolla”**
- Inheritance path: ToyotaCorolla -> Sedan -> Car -> Vehicle -> Thing
Order-Sorted Types

* Basic order-sorted types can be reduced to extra unary predicates called in the body
* More complex representation
* More resolution steps, leads to slower inferences

Time comparison of built-in sorts with unary predicates

<table>
<thead>
<tr>
<th></th>
<th>OO jDREW with built-in order-sorted types</th>
<th>OO jDREW with unary predicates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Time</td>
<td>256 ms</td>
<td>473 ms</td>
</tr>
</tbody>
</table>
Object Identifiers (oids)

- Allows unique identification of facts
- Makes finding possible facts quicker if oid is known and specified in a goal
- Like sending a message to a specified object
- Does not affect the main unification algorithm - only affects the search for matching clauses
Built-in Relations

- Many relations cannot be expressed as a finite set of facts and rules
- This requires having a system for built-in relations
- Two goals of the Built-in system:
  - Easily expandable without detailed knowledge of the system
  - Common built-in relations included within engine
- OO jDREW implements subset of SWRL built-ins
Future Work

- Improved Indexing System
  - Currently only indexed by relation name and oid (to ensure uniqueness)
  - Possible to create an indexing system based upon Discrimination Trees
- Improved Typing System
  - Currently only models taxonomic relationships
Conclusions and Questions

* The developed engine is able to use the combined positional/Object-Oriented RuleML.
* Work in progress on an advanced indexing system that may improve scaleability for large knowledge bases.
* URIs as oids are not currently implemented (difficulties with normalizing URIs).
* Available online at http://www.jdrew.org/oojdrew/