MKR - an English-like DL language

Richard H. McCullough
Email: rhm@cdepot.net

Abstract

MKR is a very-high-level knowledge representation language that is both English-like and UNIX-shell-like. By design, MKR is a general purpose KR language that is easy for humans to read and write. MKR is an extended DL language. Extensions relative to a typical Description Logic language include: context, questions, methods, n-ary relations.

This paper introduces the MKR language, describes some Semantic Web applications, and relates MKR context to other context theories.

1 Introduction

Knowledge Representation is a life-long passion of mine. In 1996, I designed my own KR language, which emphasizes context, definitions, concept formation, easy reading and writing by humans, and easy parsing by machines. In 2002 and 2003, I changed the syntax to improve the uniformity and generality of the MKR language, and modified the MKE program [1] to facilitate interoperability with the emerging W3C standards [2][3] for the Semantic Web.

Concept formation "should" utilize only disjoint species, but MKR permits non-disjoint species. MKE detects such "ambiguous" species as part of its consistency checks.

MKE is implemented in Unicon [4] and KornShell [5]. MKE is available for Linux and Windows, under the open source license: GNU GPL.

2 Hierarchy Relations

Concept hierarchies are created, modified and searched using regular-expression-style relations. The base relations are

isu unit,... isu primitive concept;

iss species,... iss genus concept;

isa unit or species,... isa concept;

and their inverses

isp primitive concept isp unit,...;

isg genus concept isg species,...;

isc concept isc unit or species,...;

Note: "unit" is the term used in Rand [6], which gives good definitions of "unit" and "concept". The Description Logic Handbook [7] uses the term "individual". Multi-level relations are expressed using the regular-expression-style suffixes

* zero or more levels

\+ one or more levels

**n n levels

For example, consider the "Implementation" access functions described by McGuinness and Patel-Schneider [7]

(concept-descendents C) C isg* ?;

(concept-children C) C isg ?;

(concept-ancestors C) C iss* ?;

(concept-parents C) C iss ?;

(concept-instances C) C isp* ?;

(concept-direct-instances C) C isp ?;

Complete hierarchies can be defined using a hierarchy group. For example

```
begin hierarchy ECP;
existent;
/    entity;
//       man;
//       group;
///          enumeration;
///          list;
///          sequence;
///          Set;
///          multiset;
///          LATTICE;
////             hierarchy;
////             lattice;
/    characteristic;
//       differentia;
```

```
//        part;
//        attribute;
//        relation;
//        action;
//        interaction;
/     proposition;
//        context;
//        sentence;
end hierarchy ECP;
```

3 Sentence Types

The basic sentence types of MKR are

statement
question
command
assignment
conditional
iteration

Statements, questions and commands are discussed later. Assignments have the format

```
set variable = value;
unset variable;
```

Conditionals have the format

```
if proposition list
then proposition list
else proposition list
fi;
```

Iterations use a "generator" statement to produce a list of variable values. For example

```
every p isa person {
   $p has email = ?;
};
every topic in knowledge representation,
      description logic,
```

```
        artificial intelligence {
    ? has topic += $topic;
};
```

4 Statement Types

The basic statement types of MKR are

definition
identity
binary relation
n-ary relation
part
attribute
action
interaction

Here are simple examples of each type

```
phonebook is relation with
    format = [person:1, phone:2],
    meaning = {$1 has phone = $2;};
man is person;
man isa entity;
Mary Doe isin phonebook = phonebook_123;
knowledge haspart proposition list;
Tom, Dick, Harry has sex = male;
Sue do walk from her home to the store done;
smoking causes lung cancer;
```

5 Questions

True-false questions are formed by wrapping statements with if and fi. For example

```
if Mary Doe isin phonebook; fi;
```

Form-based questions replace one or more elements of a statement or assignment by a question mark. For example

```
? has phone;
```

Sue do ? done;

? is ?;

apple ? orange;

set charformat = ?;

6 Commands and Methods

Commands have the same format as action statements, but with the subject omitted. For example

do read from http://rhm.cdepot.net/kb/tabrasa.html done;

The output of commands (and other sentences) can be assigned to a variable. For example

people := do ulist od person done;

males := ? has sex = male;

Methods are user-defined commands. Methods are defined by a genus-differentia definition. For example

```
read dmoz isa method with
   automatic = "isu",
   format = {do read dmoz from directory:1 done;},
   meaning = {
      do read from "$1/structure.rdf" done;
      do read from "$1/terms.rdf" done;
      do read from "$1/content.rdf" done;
   };
do read dmoz from "$kedb" done;
```

The automatic = "isu" specification generates the automatic declaration

$1 isu directory;

The meaning of a method can be implemented as an MKR script or a Unicon procedure.

7 n-ary Relations

n-ary relations are defined by a genus-differentia definition, and populated by a relation group. For example

```
phonebook is relation with
   format = [person:1, phone:2],
```

meaning = {$1 has phone = $2;};

begin relation phonebook;

John Doe, 555-1234;

Mary Doe, 555-5678;

end relation phonebook;

Since the automatic attribute defaults to "isa", the meaning is augmented by the automatic declarations

John Doe isa person;

555-1234 isa phone;

Mary Doe isa person;

555-5678 isa phone;

8 Context

The context of a proposition consists of a "view" attribute which names a proposition list (in most cases, an equivalent set of propositions can be determined), and "space", "time" attributes which specify the "where" and "when" of actions. For example

at space = my house,

time = any Saturday afternoon,

view = weekend routine

{

I do walk to the corner

with purpose = get my snailmail

done;

};

The propositions of a context are organized into an entity-characteristic-proposition hierarchy which supports efficient inferences and question answering.

9 Semantic Web Applications

The Semantic Web is a natural application for the MKE/MKR system. I am cuurently working to make MKE/MKR interoperable with systems using the RDF, OWL, and CycL languages. MKE can already read simple RDF and OWL files, and interfaces with the Stanford TAP KB [8] and the ODP directories [9] used by Google [10].

10 MKR context and other context theories

Although there are subtle differences, the MKR view is roughly isomorphic to all of these contexts

OWL Ontology [3]

FCA context [11]

CycL Microtheory [12]

McCarthy context [13]

Devlin situation [14]

Note: MKR has no modal logic; it achieves the same descriptive power using context.

11 References

1. McCullough Knowledge Explorer and the MKR language, http://rhm.cdepot.net/
2. Resource Description Framework, http://www.w3.org/RDF/
3. OWL Web Ontology Language, http://www.w3.org/2001/sw/WebOnt/
4. Clinton Jeffery, Shamim Mohamed, Ray Pereda, Robert Parlett, "Programming with Unicon", draft manuscript, 2/21/2003.
5. Morris I. Bolsky, David G. Korn, "The New KornShell Command and Programming Language", Prentice Hall PTR, 1995.
6. Ayn Rand, "Introduction to Objectivist Epistemology", Expanded Second Edition, Meridian, 1990.
7. Franz Baader, Diego Calvanese, Deborah L. McGuinness, Peter F. Patel-Schneider, editors, "The Description Logic Handbook", Cambridge University Press, 2003.
8. Stanford TAP KB, http://tap.stanford.edu/
9. Open Directory Project, http://dmoz.org/
10. Google, http://www.google.com/
11. Bernhard Ganter, Rudoff Wille, "Formal Concept Analysis: Mathematical Foundations", Springer-Verlag, 1999.
12. OpenCyc, http://www.opencyc.org/

13. John F. Sowa, "Knowledge Representation", Brooks/Cole, 2000.

14. Keith Devlin, "Logic and Information", Cambridge University Press, 1995.