Brief Overview of the Existential Rule Framework

GraphIK Team, LIRMM-Inria, Montpellier

July 2012

Existential rules allow to assert the existence of not-yet-known individuals. The existential rule framework is also known as an extension to Datalog, called *Datalog+/-*. It is particularly relevant to *ontology-based query answering*. In this framework, a knowledge base is composed of facts —or data— and of ontological knowledge expressed by existential rules (including rules with equality atoms) and negative constraints. The considered queries are conjunctive queries; extending them to a union of conjunctive queries is straightforward. In this document, we briefly present the main components of the existential rule framework. Examples are given under the form of logical sentences as well as in the dedicated *dlgp* (for "Datalog Plus") format.

A *term* is a variable or a constant. An *atom* is of the form $p(e_1 \dots e_k)$ where p is a predicate, $k \geq 1$ is the arity of p, and each e_i is a term. An *equality atom* is of the form $e_i = e_j$, where e_i and e_j are terms.

Example 1 *In the examples below, we consider the following predicates, with their arity mentioned after their name: area/1, project/1, researcher/1, isproject/3, hasExpertise/2, isMember/2. Intuitively, the unary predicates can be seen as types of entities (classes, concepts); the ternary predicate* isProject *is supposed to relate a project, the area of this project and the leader of this project; binary predicates* hasExpertise *and* isMember *relate a researcher to an area and a project, respectively.*

A **knowledge base** is composed of facts, existential rules and negative constraints. For each of these constructs, we consider below their simplified logical form (quantifiers and parentheses are omitted, which can be done because there is no ambiguity), their full logical form and their dlgp form.

A **fact** F is given as a conjunction of atoms (simplified form) logically interpreted as its existential closure $\exists \bar{X} F[X]$, where $\bar{X}$ is the set of variables occurring in F.

Example 2 (Fact)
"Researcher a is member of a project in kr area"

- *simplified logical form:*
  $F = researcher(a) \wedge isMember(a, X) \wedge isProject(X, kr, Y)$
- *full logical form:*
  $F = \exists X \exists Y (researcher(a) \wedge isMember(a, X) \wedge isProject(X, kr, Y))$
- *dlgp form:*

```
researcher(a), isMember(a,X), isProject(X, kr, Y).
% unamed fact
[F] researcher(a), isMember(a,X), isProject(X, kr, Y).
% named fact
```

An **existential rule** is a positive rule of the form $B \rightarrow H$ (simplified form), where B and H are conjunctions of atoms; it is interpreted as the formula $\forall \bar{X}(\exists \bar{Y} B[\bar{X}, \bar{Y}] \rightarrow \exists \bar{Z} H[\bar{X}, \bar{Z}])$, or equivalently $\forall \bar{X} \forall \bar{Y}(B[\bar{X}, \bar{Y}] \rightarrow \exists \bar{Z} H[\bar{X}, \bar{Z}])$, where $\bar{X}$ are the variables shared by B and H, $\bar{Y}$ are the variables that occur only in B and $\bar{Z}$ are the variables that occur only in H; $\bar{Z}$ variables are existentially quantified.

Example 3 (Existential Rule)
"Every leader of a project is a member of this project"
"Every researcher expert in an area is member of a project in this area"

- *simplified logical form:*
  $isProject(X, Y, Z) \rightarrow isMember(Z, X)$
  $researcher(X) \wedge hasExpertise(X, Y) \rightarrow isProject(Z, Y, L) \wedge isMember(X, Z)$
- *full logical form:*
  $\forall X \forall Y (isProject(X, Y, Z) \rightarrow isMember(Z, X))$
  $\forall X \forall Y (researcher(X) \wedge hasExpertise(X, Y) \rightarrow \exists Z \exists L (isProject(Z, Y, L) \wedge isMember(X, Z)))$
- *dlgp form:*

```
ismember(Z,X) :- isProject(X,Y,Z).
% plain Datalog rule
isProject(Z,Y,L), isMember(X,Z) :- researcher(X), hasExpertise(X,Y).
% extended Datalog rule:  non-atomic head and
% existential variables in the head
```

Rules may also contain equalities. The *dlgp* format allows for any form of equality anywhere in a rule. However, equalities are often restricted as follows: a distinction is made between *standard existential rules* that do not contain equalities at all and *equality rules* of the form $B \rightarrow e_1 = e_2$, where B does not contain equalities and e_1, e_2 are variables occurring in B, or constants.

Example 4 (Equality Rule)
"Every project has at most one leader"

- *simplified logical form:*
  $isProject(X, Y, Z1) \wedge isProject(X, Y, Z2) \rightarrow Z1 = Z2$
- *dlgp form:*
  `Z1=Z2:- isProject(X,Y,Z1), isProject(X,Y,Z2).`

A **negative constraint** C is a conjunction of atoms (simplified form) interpreted as the negation of its existential closure $\neg(\exists \bar{X} C[\bar{X}])$; equivalently it is a rule of the form $C \rightarrow \bot$, where $\bot$ denotes the absurd symbol (which is always false).

Example 5 (Negative Constraint)
"Classes researcher and project are disjoint"

- *simplified logical form:*
  $\neg(researcher(X) \wedge project(X))$ or
  $researcher(X) \wedge project(X) \rightarrow \bot$
- *full logical form:*
  $\neg\exists X(researcher(X) \wedge project(X))$ or
  $\forall X(researcher(X) \wedge project(X)) \rightarrow \bot)$
- *dlgp form:*
  ```
  !  :- researcher(X), project(X).
  % !  is the ‘‘always false’’ symbol
  ```

A fundamental problem consists in *querying* a knowledge base. We consider here basic queries, so-called conjunctive queries. Conjunctive queries can be encoded in the dlgp format, which allows to associate typical or frequent queries with a knowledge base.

A **conjunctive query** Q is a conjunction of atoms with a distinguished subset of its variables (the answer part of the query); it is interpreted as the logical formula obtained from Q by existentially quantifying non-distinguished variables. When the set of distinguished variables is empty, Q is a *Boolean query* (and has the same logical translation as a fact).

Example 6 (Conjunctive Query)
Q_1 :"Find the members of projects in *kr* area"
Q_2 :"Is there a project in *kr* area ?"

- *simplified logical form: same as a fact, added with a way of distinguishing some variables*

- *full logical form:*
  $Q_1 = \exists Y(isMember(X, Y) \wedge isProject(Y, kr, Z))$
  $Q_2 = \exists X \exists Z isProject(X, kr, Z)$

- *dlgp form:*

```
? (X) :- isMember(X,Y), isProject(Y, kr, Z).
? :- isProject(X,kr,Z).
% Boolean query
```