PSOA Prova: PSOA Translation of Pure Production Rules to the Prova Engine

Lukas Grätz¹  Harold Boley²  Adrian Paschke³

Institut für Philosophie, Universität Leipzig, Germany
lukas[DT]graetz[AT]studserv.uni-leipzig.de

Faculty of Computer Science, University of New Brunswick, Fredericton, Canada
harold[DT]boley[AT]unb.ca

Fraunhofer FOKUS and Freie Universitaet Berlin, Germany
adrian[DT]paschke[AT]fokus.fraunhofer.de

RuleML+RR 2018 – Esch-sur-Alzette, Luxembourg
Session 29B: Rule Challenge
September 20, 2018
Table of Contents

1 Introduction

2 Use Case: Royal Family

3 PSOA RuleML for Reconstruction
   Extension: Runtime KB Consult and Unconsult

4 Demonstration
   Succession by a Derivation Rule
   Succession by a Pure Production Rule

5 Results
Introduction
Use Case: Royal Family

- How did Prince William become a successor?
- How to represent successorship with KR?
Use Case: Royal Family

- How did Prince William become a successor?
  - Succession to the Crown Act

- How to represent successorship with KR?
  - Our challenge
Royal Family

1. 

Diana  Charles 🕯️

2. 

3. 

4. 

5.
Royal Family
Royal Family

1. Diana Charles ♠

2. Diana ⊤ Charles ♠

3. Diana ⊤ Charles ♠
   William

4. 

5. 
Royal Family

1. Diana  Charles 💍

2. Diana ★ Charles 💍

3. Diana ★ Charles 💍 William

4. Diana ★ Charles 💍 William 💍

5.  

Lukas Grätz, Harold Boley, Adrian Paschke
PSOA Prova
RuleML+RR Challenge 2018
Royal Family

1. Diana  Charles  

2. Diana  Charles  

3. Diana  Charles  
   William

4. Diana  Charles  
   William

5. Diana  Charles  
   William
PSOA RuleML for Reconstruction

- **object-relational knowledge**
  - Object#Predicate
  - Object#Predicate( key+>value )
  - Object#Top ( key->value )
  - Object#Predicate( elem_1 elem_2 ... )
  - Object#Top ( -[elem_1 elem_2 ...] )
  - OID#Pred( d1... -[e1... k1+>v1 k2->v2 ...) 

- **derivation rules**

  Forall ?Q1 ?Q2 ... ?Qn (  
    <conclusion> :- And( <cond_1> <cond_2> ...)  
  )
object-relational knowledge

- **Object#Predicate**
  - Predicate( key+>value )
- **Object#Top**
  - ( key->value )
  - Predicate( elem_1 elem_2 ... )
- **Object#Top**
  - ( -[elem_1 elem_2 ...] )
- **OID#Pred**
  - ( d1... -[e1...] k1+>v1 k2->v2 ... )

**derivation rules**

\[
\text{Forall } ?Q_1 ?Q_2 \ldots ?Q_n ( \\
\hspace{1cm}<\text{conclusion}> : - \text{And}( <\text{cond}_1> <\text{cond}_2> \ldots )
)\]
Extension: Runtime KB Consult and Unconsult

New feature in PSOA Prova: consult and unconsult at run-time

> consult RoyalFamily-KB2.psoa

> unconsult RoyalFamily - KB2 . psoa

Translated KB:
prdsloterm ('_1 ', '_marriage ', '_partner ', '_Diana ').
prdsloterm ('_1 ', '_marriage ', '_partner ', '_Charles '). 
memterm ('_1 ', '_marriage ').

Translated Query:
prdsloterm (Q1 ,' _marriage ',' _partner ',' _Diana '),
prdsloterm (Q1 ,' _marriage ',' _partner ',' _Charles '),
memterm (Q1 ,' _marriage ').

Answer (s): No
New feature in PSOA Prova: **consult** and **unconsult** at run-time

```prolog
> consult RoyalFamily-KB2.psoa
```

Translated KB:

```prolog
prdsloterm(''_1',''_marriage',''_partner',''_Diana'').
prdsloterm(''_1',''_marriage',''_partner',''_Charles'').
memterm(''_1',''_marriage'').
```

```prolog
> unconsult RoyalFamily - KB2.psoa
```

Translated Query:

```prolog
prdsloterm(Q1,''_marriage',''_partner',''_Diana''),
prdsloterm(Q1,''_marriage',''_partner',''_Charles''),
memterm(Q1,''_marriage'').
```

Answer(s): No
Extension: Runtime KB Consult and Unconsult

New feature in PSOA Prova: consult and unconsult at run-time

> consult RoyalFamily-KB2.psoa

Translated KB:

.prdsloterm('1','_marriage','_partner','_Diana').  
.prdsloterm('1','_marriage','_partner','_Charles').  
.memterm('1','_marriage').

> unconsult RoyalFamily-KB2.psoa

Translated Query:

.prdsloterm(Q1,'_marriage','_partner','_Diana'),  
.prdsloterm(Q1,'_marriage','_partner','_Charles'),  
.memterm(Q1,'_marriage').

Answer(s):

No
Extension: Runtime KB Consult and Unconsult

New feature in PSOA Prova: consult and unconsult at run-time

> consult RoyalFamily-KB2.psoa

Translated KB:

prdsloterm(''_1'',''_marriage',''_partner',''_Diana'').
prdsloterm(''_1'',''_marriage',''_partner',''_Charles'').
memterm(''_1'',''_marriage'').

> unconsult RoyalFamily-KB2.psoa

> marriage( partner+>Diana partner+>Charles )

Translated Query:

prdsloterm(Q1,''_marriage',''_partner',''_Diana''),
prdsloterm(Q1,''_marriage',''_partner',''_Charles''),
memterm(Q1,''_marriage'').

Answer(s):
No
Succession by a Derivation Rule

\[
\forall ?Ch \ ?P1 \ ?P2 \quad (?Ch\#\text{successor} :- \quad \text{And} (\ ?Ch\#\text{child} (\ \text{parent}-\rightarrow\ ?P1 \ \text{parent}-\rightarrow\ ?P2 ) \quad \text{marriage} (\ \text{partner}+\rightarrow\ ?P1 \ \text{partner}+\rightarrow\ ?P2 ) \quad ?P1\#\text{successor})
\]

\[
\Rightarrow
\]

Translation into Prolog:

\[
\text{memterm} (QCh \ ' \ _\text{successor} ') \quad :- \quad \text{sloterm} (QCh \ ' \ _\text{parent} ',QP1) \quad , \quad \text{sloterm} (QCh \ ' \ _\text{parent} ',QP2) \quad , \quad \text{memterm} (QCh \ ' \ _\text{child}') \quad , \quad \text{prdsloterm} (Q1 \ ' \ _\text{marriage} ' \ ' \ _\text{partner}',QP1) \quad , \quad \text{prdsloterm} (Q1 \ ' \ _\text{marriage} ' \ ' \ _\text{partner}',QP2) \quad , \quad \text{memterm} (Q1 \ ' \ _\text{marriage}') \quad , \quad \text{memterm} (QP1 \ ' \ _\text{successor}).
\]
Succession by a Derivation Rule

\[
\text{For all } \forall \text{Ch } \text{?P1 } \text{?P2 (}
\text{?Ch}\#\text{successor} :-
\quad \text{And ( } \text{?Ch}\#\text{child ( parent->?P1 parent->?P2 )}
\quad \text{marriage ( partner+->?P1 partner+->?P2 )}
\quad \text{?P1}\#\text{successor}
\quad )}
\]

\[\Rightarrow\] Translation into Prolog:

```prolog
memterm(QCh,'_successor')
:- sloterm(QCh,'_parent',QP1),
    sloterm(QCh,'_parent',QP2),
    memterm(QCh,'_child'),
    prdsloterm(Q1,'_marriage','_partner',QP1),
    prdsloterm(Q1,'_marriage','_partner',QP2),
    memterm(Q1,'_marriage'),
    memterm(QP1,'_successor').
```
Succession by a Pure Production Rule

Definition

A pure production rule is an extended derivation rule, where the derived conclusion is asserted persistently to the KB. If the condition holds, the conclusion is derivable; moreover, the conclusion will be asserted at least before the condition becomes unsatisfied.

Notation: \[ \text{<conclusion>} ::= \text{<condition>} \]
Succession by a Pure Production Rule

Definition

A pure production rule is an extended derivation rule, where the derived conclusion is asserted persistently to the KB. If the condition holds, the conclusion is derivable; moreover, the conclusion will be asserted at least before the condition becomes unsatisfied.

Notation: \(<\text{conclusion}> ::= <\text{condition}>\)

\[
\text{Forall } ?Ch \ ?P1 \ ?P2 ( \\
\quad ?Ch\#\text{successor} :::- \\
\quad \quad \text{And}( \ ?Ch\#\text{child}( \ \text{parent} -> ?P1 \ \text{parent} -> ?P2) \\
\quad \quad \quad \text{marriage}( \ \text{partner} + -> ?P1 \ \text{partner} + -> ?P2) \\
\quad \quad ?P1\#\text{successor} \\
\quad )
\]

Lukas Grätz, Harold Boley, Adrian Paschke
PSOA Prova
RuleML+RR Challenge 2018
Demonstration
(screenrecord – just in case)
Evaluation of a Pure Production Rule

Events to start evaluation \( \hat{=} \) assert the conclusion:

1. **Structure operation**
   - after consult
   - before unconsult
   - ...

2. **Behavior invocation**
   - conclusion is derivation (sub)goal
   - ...

3. **Clock** (e.g. polling)

4. **External**

5. ...

transparent for the user!
Results

✓ PSOATransRun fork on Github
✓ PSOATransRun[PSOA2Prova,Prova]
✓ Consult and unconsult
✓ Pure production rules

Wiki
The End