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

Lukas Grätz[1] Harold Boley[2] Adrian Paschke[3]

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

Introduction

Use Case: Royal Family

- How did Prince William become a successor?
- How to represent successorship with KR?

Use Case: Royal Family

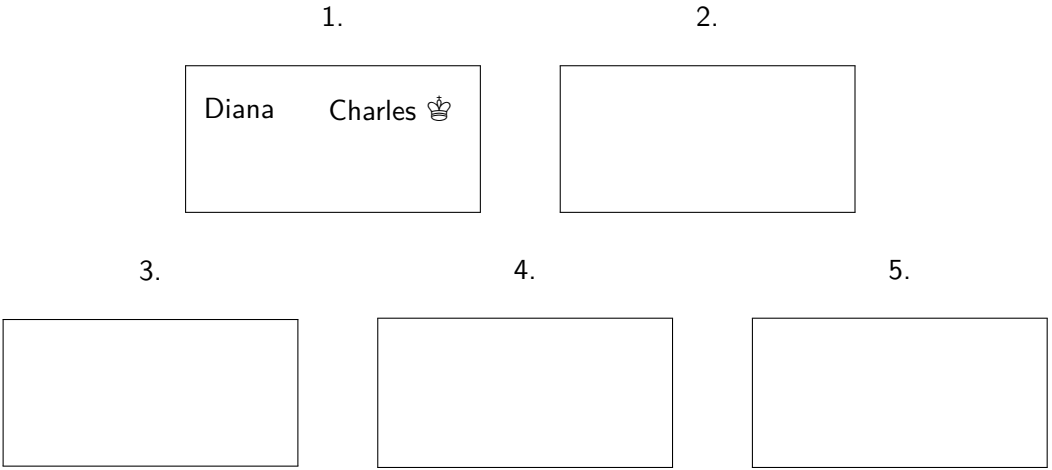
- How did Prince William become a successor?

  ☞ *Succession to the Crown Act*

- How to represent successorship with KR?

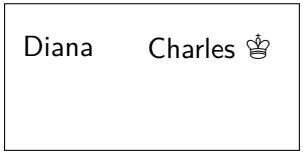
  ☞ Our challenge

Royal Family

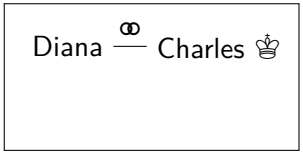


Royal Family

1.



2.



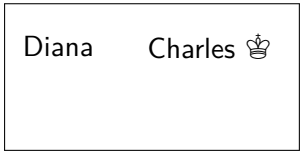
3.

4.

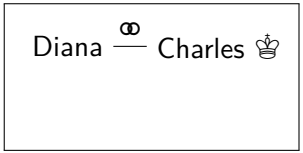
5.

Royal Family

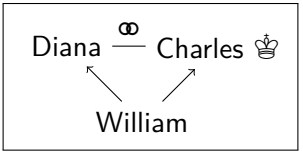
1.



2.



3.

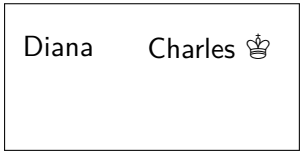


4.

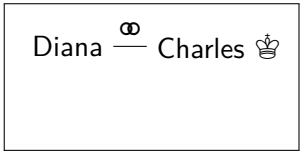
5.

Royal Family

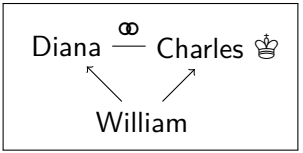
1.



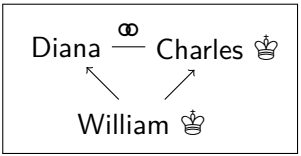
2.



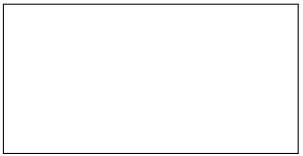
3.



4.

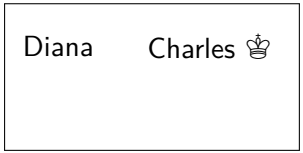


5.

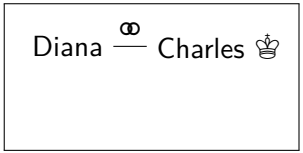


Royal Family

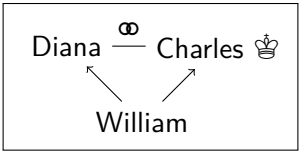
1.



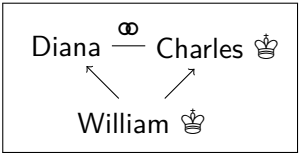
2.



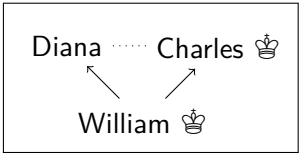
3.



4.



5.



PSOA RuleML for Reconstruction

- object-relational knowledge
  - `Object#Predicate` — membership
  - `Object#Predicate( key+>value )` — dependent slot
  - `Object#Top ( key->value )` — independent slot
  - `Object#Predicate( elem_1 elem_2 ... )` — dependent tuple
  - `Object#Top ( -[elem_1 elem_2 ...] )` — independent tuple
  - `OID#Pred( d1... -[e1...] k1+>v1 k2->v2 ...)` — all together
- derivation rules

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

PSOA RuleML for Reconstruction

- object-relational knowledge
  - `Object#Predicate` — membership
  - `Predicate( key+>value )` — dependent slot
  - `Object#Top ( key->value )` — independent slot
  - `Predicate( elem_1 elem_2 ... )` — dependent tuple
  - `Object#Top ( -[elem_1 elem_2 ...] )` — independent tuple
  - `OID#Pred( d1... -[e1...] k1+>v1 k2->v2 ...)` — all together
- derivation rules

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

Extension: Runtime KB Consult and Unconsult

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

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

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').
```

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
```

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 (
  ?Ch#successor  :-
     And( ?Ch#child( parent->?P1 parent->?P2 )
          marriage( partner+>?P1 partner+>?P2 )
          ?P1#successor
     )
)
```

Succession by a Derivation Rule

```
Forall ?Ch ?P1 ?P2 (
  ?Ch#successor  :-
     And( ?Ch#child( parent->?P1 parent->?P2 )
          marriage( partner+>?P1 partner+>?P2 )
          ?P1#successor
     )
)
```

$\rightsquigarrow$ Translation into 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: `<conclusion> ::- <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: `<conclusion> ::- <condition>`

```
Forall ?Ch ?P1 ?P2 (
  ?Ch#successor ::-
    And( ?Ch#child( parent->?P1 parent->?P2 )
         marriage( partner+>?P1 partner+>?P2 )
         ?P1#successor
    )
)
```

Derivation vs. Pure Production Rule

Demonstration

(screenrecord – just in case)

Evaluation of a Pure Production Rule

Events to start evaluation $\hat{=}$ `assert` the conclusion:

1. **Structure operation**
   1. after `consult`
   2. before `unconsult`
   3. ...
2. **Behavior invocation**
   1. conclusion is derivation (sub)goal
   2. ...
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