Operational Semantics for SPARQL Update

1st Joint International Semantic Technology Conference
Hangzhou, China

Ross Horne Vladimiro Sassone Nicholas Gibbins

Electronics and Computer Science, University of Southampton, United Kingdom

6th December 2011

Some data about the capital of the Song dynasty.

Data:

$$\{dbp{:}Song\_Dynasty \;\; ont{:}capital \;\; dbp{:}Nanjing\}$$

The capital of the Song dynasty changes.

Data:

$$\{dbp{:}Song\_Dynasty \;\; ont{:}capital \;\; dbp{:}Nanjing\}$$

Update:

$$\texttt{DELETE}\,\{dbp{:}Song\_Dynasty \;\; ont{:}capital \;\; dbp{:}Nanjing\}$$
$$\texttt{JOIN}$$
$$\texttt{INSERT}\,\{dbp{:}Song\_Dynasty \;\; ont{:}capital \;\; dbp{:}Hangzhou\}$$

The capital of the Song dynasty becomes Hangzhou.

Data before:

{*dbp:Song_Dynasty ont:capital dbp:Nanjing*}

Update:

`DELETE` {*dbp:Song_Dynasty ont:capital dbp:Nanjing*}
`JOIN`
`INSERT` {*dbp:Song_Dynasty ont:capital dbp:Hangzhou*}

Data after:

{*dbp:Song_Dynasty ont:capital dbp:Hangzhou*}

The capital of the Song dynasty becomes Hangzhou.

Data before:

{*dbp:Song_Dynasty* *ont:capital* *dbp:Nanjing*}

Update:

`DELETE` {*dbp:Song_Dynasty* *ont:capital* *dbp:Nanjing*}
`INSERT` {*dbp:Song_Dynasty* *ont:capital* *dbp:Hangzhou*}

Data after:

{*dbp:Song_Dynasty* *ont:capital* *dbp:Hangzhou*}

Several dynasties located their capital in Nanjing.

Data:

$$\{dbp{:}Kingdom\_of\_Wu \;\; ont{:}capital \;\; dbp{:}Nanjing\},$$
$$\{dbp{:}Kingdom\_of\_Wu \;\; ont{:}ended \;\; 208\},$$
$$\{dbp{:}Chen\_Dynasty \;\; ont{:}capital \;\; dbp{:}Nanjing\},$$
$$\{dbp{:}Chen\_Dynasty \;\; ont{:}ended \;\; 589\},$$
$$\{dbp{:}Song\_Dynasty \;\; ont{:}capital \;\; dbp{:}Nanjing\},$$
$$\{dbp{:}Song\_Dynasty \;\; ont{:}ended \;\; 1279\},$$
$$\{dbp{:}Ming\_Dynasty \;\; ont{:}capital \;\; dbp{:}Nanjing\},$$
$$\{dbp{:}Ming\_Dynasty \;\; ont{:}ended \;\; 1664\}$$

Several dynasties located their capital in Nanjing before 1368.

Data:

{*dbp:Kingdom_of_Wu* *ont:capital* *dbp:Nanjing*},
{*dbp:Kingdom_of_Wu* *ont:ended* 208},
{*dbp:Chen_Dynasty* *ont:capital* *dbp:Nanjing*},
{*dbp:Chen_Dynasty* *ont:ended* 589},
{*dbp:Song_Dynasty* *ont:capital* *dbp:Nanjing*},
{*dbp:Song_Dynasty* *ont:ended* 1279},
{*dbp:Ming_Dynasty* *ont:capital* *dbp:Nanjing*},
{*dbp:Ming_Dynasty* *ont:ended* 1664}

Query:

```
SELECT ?year{
  ASK{:dynasty ont:ended ?year}
  FILTER (?year < 1368)
}
```

Several dynasties located their capital in Nanjing before 1368.

Data before:

{*dbp:Kingdom_of_Wu* *ont:capital* *dbp:Nanjing*},
{*dbp:Kingdom_of_Wu* *ont:ended* 208},
{*dbp:Chen_Dynasty* *ont:capital* *dbp:Nanjing*},
{*dbp:Chen_Dynasty* *ont:ended* 589},
{*dbp:Song_Dynasty* *ont:capital* *dbp:Nanjing*},
{*dbp:Song_Dynasty* *ont:ended* 1279},
{*dbp:Ming_Dynasty* *ont:capital* *dbp:Nanjing*},
{*dbp:Ming_Dynasty* *ont:ended* 1664}

Query:

```
SELECT :dynasty{
  SELECT ?year{
    ASK{:dynasty ont:ended ?year}
    FILTER (?year < 1368)
  }
  DisplayLinkedData(:dynasty)
}
```

Nanjing was founded on the site of Jiankang in 1368.

Data before:

{*dbp:Kingdom_of_Wu* *ont:capital* *dbp:Nanjing*},
{*dbp:Kingdom_of_Wu* *ont:ended* 208},
{*dbp:Chen_Dynasty* *ont:capital* *dbp:Nanjing*},
{*dbp:Chen_Dynasty* *ont:ended* 589},
{*dbp:Song_Dynasty* *ont:capital* *dbp:Nanjing*},
{*dbp:Song_Dynasty* *ont:ended* 1279},
{*dbp:Ming_Dynasty* *ont:capital* *dbp:Nanjing*},
{*dbp:Ming_Dynasty* *ont:ended* 1664}

Update:

```
DO SELECT :dynasty{
   SELECT ?year{
      ASK{:dynasty ont:ended ?year}
      FILTER (?year < 1368)
   }
   DELETE{:dynasty ont:capital dbp:Nanjing}
   INSERT{:dynasty ont:capital dbp:Jiankang}
}
```

Jiankang has been swallowed by modern Nanjing.

Update:

```
DO SELECT :dynasty {
   SELECT ?year {
      ASK {:dynasty ont:ended ?year}
      FILTER (?year < 1368)
   }
   DELETE {:dynasty ont:capital dbp:Nanjing}
   INSERT {:dynasty ont:capital dbp:Jiankang}
}
```

Data after:

{*dbp:Kingdom_of_Wu ont:capital dbp:Jiankang*},
{*dbp:Kingdom_of_Wu ont:ended* 208},
{*dbp:Chen_Dynasty ont:capital dbp:Jiankang*},
{*dbp:Chen_Dynasty ont:ended* 589},
{*dbp:Song_Dynasty ont:capital dbp:Jiankang*},
{*dbp:Song_Dynasty ont:ended* 1279},
{*dbp:Ming_Dynasty ont:capital dbp:Nanjing*},
{*dbp:Ming_Dynasty ont:ended* 1664}

An abstract syntax for data, constraints and updates.

Data ::= {}
| *URI URI Object*
| *Data*, *Data*
| `BNODE` *URI Data*

Object ::= `'literal'`
| *URI*
| *?variable*

Update ::= `DELETE` *Data*
| `INSERT` *Data*
| `ASK` *Data*
| `FILTER` *Constraint*
| *Update* `CHOOSE` *Update*
| *Update* `JOIN` *Update*
| `SELECT` *URI Update*
| `SELECT` *?variable Update*
| `DO` *Update*

Constraint ::= `true`
| `false`
| *Constraint* && *Constraint*
| *Constraint* || *Constraint*
| `!` *Constraint*
| `regex`(*?variable*, *RegEx*)
| ...

The axioms and rules of the operational semantics.

$$(P, Q), R = P, (Q, R) \qquad P, \{\} = P \qquad P, Q = Q, P$$

$$Data, \texttt{DELETE}\ Data \to \{\} \qquad \{\}, \texttt{INSERT}\ Data \to Data \qquad \frac{Constraint = \texttt{true}}{\{\}, \texttt{FILTER}\ Constraint \to \{\}}$$

$$\frac{Data_0, Update_0 \to Data_2 \quad Data_1, Update_1 \to Data_3}{Data_0, Data_1, Update_0\ \texttt{JOIN}\ Update_1 \to Data_2, Data_3}$$

$$\frac{Data_0, Update[\texttt{`literal'}/?variable] \to Data_1}{Data_0, \texttt{SELECT}\ ?variable\ Update \to Data_1} \qquad \frac{Data_0, Update[:b/:a] \to Data_1}{Data_0, \texttt{SELECT}\ :a\ Update \to Data_1}$$

$$\frac{Data_0, Update_0 \to Data_1}{Data_0, Update_0\ \texttt{CHOOSE}\ Update_1 \to Data_1} \qquad \frac{Data_0, Update_1 \to Data_1}{Data_0, Update_0\ \texttt{CHOOSE}\ Update_1 \to Data_1}$$

$$\{\}, \texttt{DO}\ Update \to \{\} \qquad \frac{Data_0, Update \to Data_1}{Data_0, \texttt{DO}\ Update \to Data_1} \qquad \frac{Data_0, \texttt{DO}\ Update\ \texttt{JOIN}\ \texttt{DO}\ Update \to Data_1}{Data_0, \texttt{DO}\ Update \to Data_1}$$

$$\frac{Data_0, Update \to Data_1}{Data_0, Data_2, Update \to Data_1, Data_2} \qquad \frac{Data_0, Data_1, Update \to Data_2, Data_3}{Data_0, \texttt{BNODE}\ :a\ Data_1, Update \to Data_2, \texttt{BNODE}\ :a\ Data_3}$$

$$:a \notin \mathrm{fn}\,(Data_0, Data_2)$$

The capital of the Song dynasty becomes Hangzhou.

The following two axioms hold.

Before: {*dbp:Song_Dynasty ont:capital dbp:Nanjing*}
Update: `DELETE` {*dbp:Song_Dynasty ont:capital dbp:Nanjing*}
After: {}

Before: {}
Update: `INSERT` {*dbp:Song_Dynasty ont:capital dbp:Hangzhou*}
After: {*dbp:Song_Dynasty ont:capital dbp:Hangzhou*}

Hence, by the join rule, the following Update holds.

Before: {*dbp:Song_Dynasty ont:capital dbp:Nanjing*}
Update: `DELETE` {*dbp:Song_Dynasty ont:capital dbp:Nanjing*}
`INSERT` {*dbp:Song_Dynasty ont:capital dbp:Hangzhou*}
After: {*dbp:Song_Dynasty ont:capital dbp:Hangzhou*}

This should be familiar from last night!

What we thought was either pork or lamb, at the baquet, was crocodile.

Data before:

{*my:feast* *ont:dish* *dbp:pork*}

Update:

```
{
  DELETE {my:feast  ont:dish  dbp:pork}
  CHOOSE
  DELETE {my:feast  ont:dish  dbp:lamb}
}
INSERT {my:feast  ont:dish  dbp:crocodile}
```

Data after:

{*my:feast* *ont:dish* *dbp:crocodile*}

What we thought was either pork or lamb, was crocodile.

Data before:

{*my:feast ont:dish dbp:pork*} ,
{*my:feast ont:dish dbp:lamb*}

Update:

```
DO{
  DELETE{my:feast ont:dish dbp:pork}
  CHOOSE
  DELETE{my:feast ont:dish dbp:lamb}
}
INSERT{my:feast ont:dish dbp:crocodile}
```

Data after:

{*my:feast ont:dish dbp:crocodile*}

First, travel from Shanghai to Hangzhou.

Data before:

$$
\begin{aligned}
&\texttt{BNODE}\ :journey_1\ \{\\
&\quad \{my{:}trip\ \ ont{:}route\ \ :journey_1\},\\
&\quad \{:journey_1\ \ ont{:}journey\ \ my{:}Shangahi\_to\_Hangzhou\},\\
&\quad \{:journey_1\ \ ont{:}previous\ \ rdf{:}nil\}\\
&\}
\end{aligned}
$$

Hangzhou to Nanjing is the next leg of the trip.

Data before:

```
BNODE :journey_1 {
  {my:trip  ont:route  :journey_1},
  {:journey_1  ont:journey  my:Shangahi_to_Hangzhou},
  {:journey_1  ont:previous  rdf:nil}
}
```

Update:

```
SELECT :current{
  DELETE{my:trip  ont:route  :current}
  BNODE :next{
    INSERT{:next  ont:route  :current}
    INSERT{:next  ont:previous  :current}
    INSERT{:next  ont:journey  my:Hangzhou_to_Nanjing}
  }
}
```

Nanjing is the next stop after Hangzhou.

Update:

```
SELECT :current {
  DELETE {my:trip  ont:route  :current}
  BNODE :next {
    INSERT {:next  ont:route  :current}
    INSERT {:next  ont:previous  :current}
    INSERT {:next  ont:journey  dbp:Hangzhou_to_Nanjing}
  }
}
```

Data after:

```
BNODE :journey_2 {
  {my:trip  ont:route  :journey_2},
  {:journey_2  ont:journey  my:Hangzhou_to_Nanjing},
  BNODE :journey_1 {
    {:journey_2  ont:previous  :journey_1},
    {:journey_1  ont:journey  my:Shangahi_to_Hangzhou},
    {:journey_1  ont:previous  rdf:nil}
  }
}
```

How long do trains take from Hangzhou to Nanjing?

Definition of *Hangzhou_to_Nanjing* (*:train*, *?duration*):

```
SELECT :stop_1 {
    ASK {:train  ont:stop  :stop_1}
    ASK {:stop_1  ont:station  Hangzhou}
    ASK {:stop_1  ont:depart  ?departure_time}
}
SELECT :stop_2 {
    ASK {:train  ont:stop  :stop_2}
    ASK {:stop_2  ont:station  Nanjing}
    ASK {:stop_2  ont:arrive  ?arrival_time}
}
FILTER (?duration = ?departure_time − ?arrival_time)
FILTER (?duration > 0)
```

Publish a fast train from Hangzhou to Nanjing.

Update:

```
SELECT ?small_duration {
  DO SELECT :train ?duration {
    Hangzhou_to_Nanjing (:train, ?duration)
    FILTER ?small_duration ≤ ?duration
  }
  SELECT :fast_train {
    Hangzhou_to_Nanjing (:fast_train, ?duration)
    SELECT :next_leg {
      ASK {my:trip ont:route :next_leg}
      INSERT {:next_leg ont:train :fast_train}
    }
  }
}
```

Conclusion

1. An increasingly relevant application; as more users contribute data.
2. An executable specification for a candidate W3C recommendation.
3. A concise modular syntax.
4. The first genuine operational semantics for a language for Linked Data.
5. The first rich verified algebra and genuine type system for Linked Data.

Remember that Jiankang is the ancient capital on the site of Nanjing.

Update:

```
DO SELECT :dynasty {
   SELECT ?year {
      ASK {:dynasty ont:ended ?year}
      FILTER (?year < 1368)
   }
   DELETE {:dynasty ont:capital dbp:Nanjing}
   INSERT {:dynasty ont:capital dbp:Jiankang}
}
```

Using that algebra the following update has the same operational behaviour.

Update:

```
DO SELECT :dynasty ?year {
   DELETE {:dynasty ont:capital dbp:Nanjing}
   INSERT {:dynasty ont:capital dbp:Jiankang}
   ASK {:dynasty ont:ended ?year}
   FILTER (?year < 1368)
}
```

The update can be translated into HP Labs original language.

Update:

```
DELETE{?dynasty ont:capital dbp:Nanjing}
INSERT{?dynasty ont:capital dbp:Jiankang}
WHERE{
   {?dynasty ont:ended ?year}
   FILTER (?year < 1368)
}
```

Example of rewriting a process using the algebra

$$\texttt{SELECT}\ a \left\{ \begin{array}{l} \texttt{SELECT}\ z \left\{ \begin{array}{l} \left\{ \begin{array}{l} \texttt{SELECT}\ x, y \left\{ \begin{array}{l} \texttt{ASK}\{a\ \ \textit{foaf:givenName}\ \ x\} \\ \texttt{ASK}\{a\ \ \textit{foaf:familyName}\ \ y\} \\ \texttt{FILTER}\ (z = x + \text{`}\ \ \text{'} + y) \end{array} \right\} \\ \texttt{CHOOSE} \\ \texttt{ASK}\{a\ \ \textit{foaf:name}\ \ z\} \end{array} \right\} \\ \texttt{FILTER}\ (z \in \text{`}\texttt{J.* Armstrong}\text{'}) \end{array} \right\} \\ \left\{ \begin{array}{l} \texttt{ASK}\{a\ \ \textit{rdf:type}\ \ \textit{dbp:Athlete}\} \\ \texttt{CHOOSE} \\ \texttt{ASK}\{a\ \ \textit{rdf:type}\ \ \textit{dbp:Artist}\} \end{array} \right\} \\ \texttt{THEN}\ P \end{array} \right\}$$

The same process as the disjunction of four processes

$$\texttt{SELECT}\ a \left\{ \begin{array}{l} \texttt{SELECT}\ \mathit{given}\ \mathit{family} \left\{ \begin{array}{l} \texttt{ASK}\,\{a\ \ \mathit{rdf{:}type}\ \ \mathit{dbp{:}Artist}\} \\ \texttt{ASK}\,\{a\ \ \mathit{foaf{:}givenName}\ \ \mathit{given}\} \\ \texttt{ASK}\,\{a\ \ \mathit{foaf{:}familyName}\ \ \mathit{family}\} \\ \texttt{FILTER}\ ((\mathit{given} + \text{`}\ \ \text{'} + \mathit{family}) \in \text{`J.* Armstrong'}) \end{array} \right\} \\ \\ \texttt{THEN}\ P \end{array} \right\}$$

$$\texttt{CHOOSE}$$

$$\texttt{SELECT}\ a \left\{ \begin{array}{l} \texttt{SELECT}\ \mathit{full} \left\{ \begin{array}{l} \texttt{ASK}\,\{a\ \ \mathit{rdf{:}type}\ \ \mathit{dbp{:}Artist}\} \\ \texttt{ASK}\,\{a\ \ \mathit{foaf{:}name}\ \ \mathit{full}\} \\ \texttt{FILTER}\ (a \in \text{`J.* Armstrong'}) \end{array} \right\} \\ \\ \texttt{THEN}\ P \end{array} \right\}$$

$$\texttt{CHOOSE}$$

$$\texttt{SELECT}\ a \left\{ \begin{array}{l} \texttt{SELECT}\ \mathit{full} \left\{ \begin{array}{l} \texttt{ASK}\,\{a\ \ \mathit{rdf{:}type}\ \ \mathit{dbp{:}Athlete}\} \\ \texttt{ASK}\,\{a\ \ \mathit{foaf{:}name}\ \ \mathit{full}\} \\ \texttt{FILTER}\ (\mathit{full} \in \text{`J.* Armstrong'}) \end{array} \right\} \\ \\ \texttt{THEN}\ P \end{array} \right\}$$

$$\texttt{CHOOSE}$$

$$\texttt{SELECT}\ a \left\{ \begin{array}{l} \texttt{SELECT}\ \mathit{given}, \mathit{family} \left\{ \begin{array}{l} \texttt{ASK}\,\{a\ \ \mathit{rdf{:}type}\ \ \mathit{dbp{:}Athlete}\} \\ \texttt{ASK}\,\{a\ \ \mathit{foaf{:}givenName}\ \ \mathit{given}\} \\ \texttt{ASK}\,\{a\ \ \mathit{foaf{:}familyName}\ \ \mathit{family}\} \\ \texttt{FILTER}\ ((\mathit{given} + \text{`}\ \ \text{'} + \mathit{family}) \in \text{`.* Armstrong'}) \end{array} \right\} \\ \\ \texttt{THEN}\ P \end{array} \right\}$$