Grailog KS Viz 2.0: Graph-Logic Knowledge Visualization by XML-Based Translation

Leah Bidlake

The 6th Atlantic Workshop on Semantics and Services (AWoSS 2015)

December 9th, 2015

UNB Fredericton

Outline

- Introduction
  - Knowledge Visualization
  - Graph Inscribed Logic (Grailog)
- Grailog KS Viz
  - Architecture
  - Supported Elements
  - Structure
  - Example
- Conclusions & Future Work

Knowledge Visualization

- Expression of knowledge through visualizations in order to communicate and analyze knowledge
- Visualizations increase the rate and quality of (human-to-human and machine-to-human) knowledge transfer
- (Semi-)Formal knowledge as used in Data Modeling, the Semantic Web, etc. can be visualized using (generalized) graphs

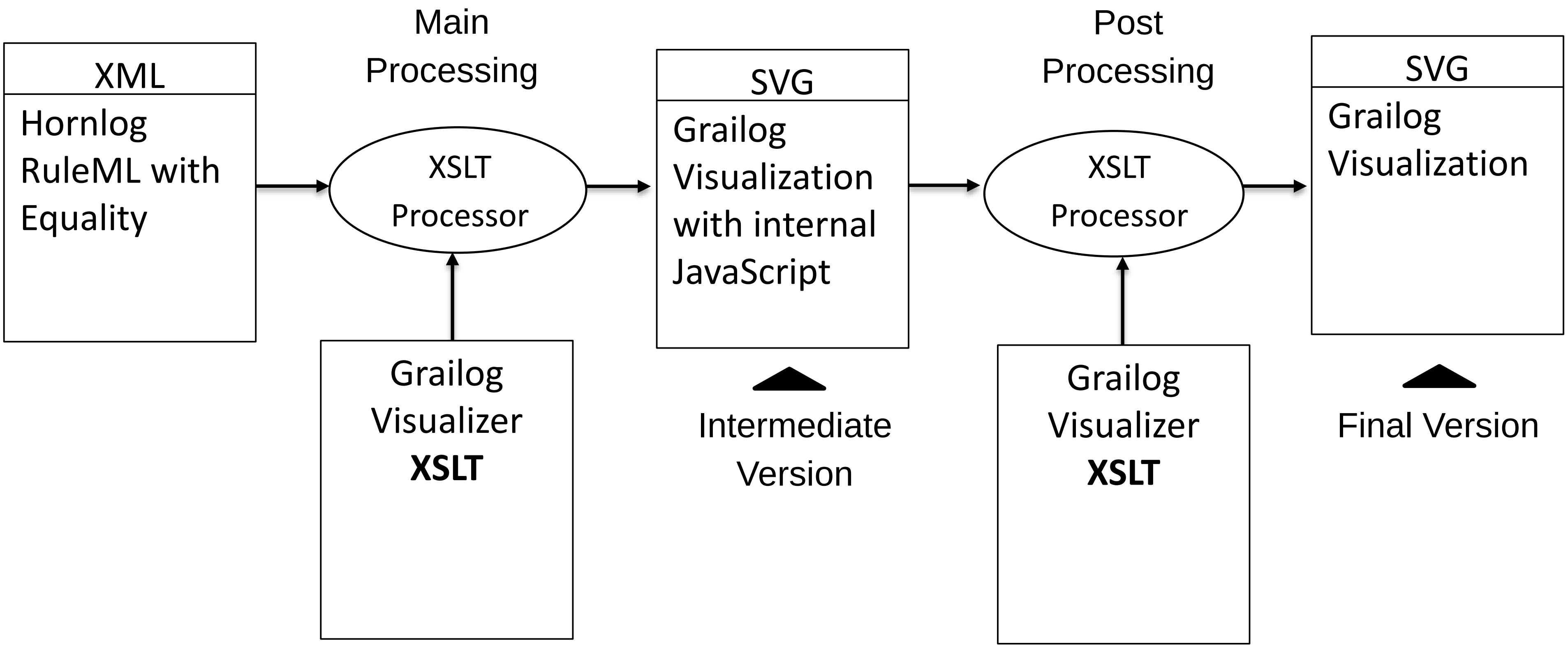
Graph Inscribed Logic (Grailog)

- Grailog used to present languages of the Rule Markup Language (RuleML) system
- Highly expressive generalized graphs for logical knowledge visualization
(in labelnode normal form)
- Contain directed n-ary hyperarcs that begin at a class/relation labelnode,
pass through n-1 intermediate argument labelnodes, and point to the nth argument labelnode

Grailog KS Viz 2.0: Subset of Horn Logic with Equality in SVG (1)

- XML documents containing HornlogEq RuleML are transformed – using an XSLT stylesheet and processor – into a Grailog visualization in SVG format that contains JavaScript
- Post-processing the SVG removes the JavaScript, which is no longer needed after positioning the SVG elements

Grailog KS Viz 2.0: Subset of Horn Logic with Equality in SVG (2)



Main Processing

Post Processing

XML

Hornlog RuleML with Equality

XSLT Processor

Grailog Visualizer **XSLT**

SVG

Grailog Visualization with internal JavaScript

Intermediate Version

XSLT Processor

Grailog Visualizer **XSLT**

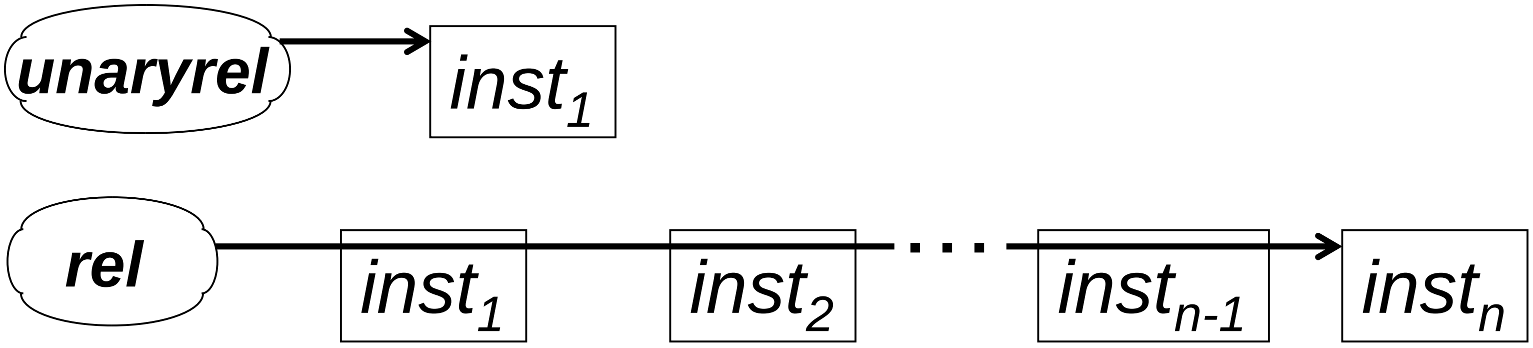
SVG

Grailog Visualization

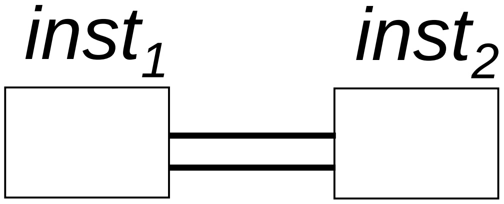
Final Version

Supported Grailog Elements (1)

- Predicates with n-ary relations for n ≥ 1



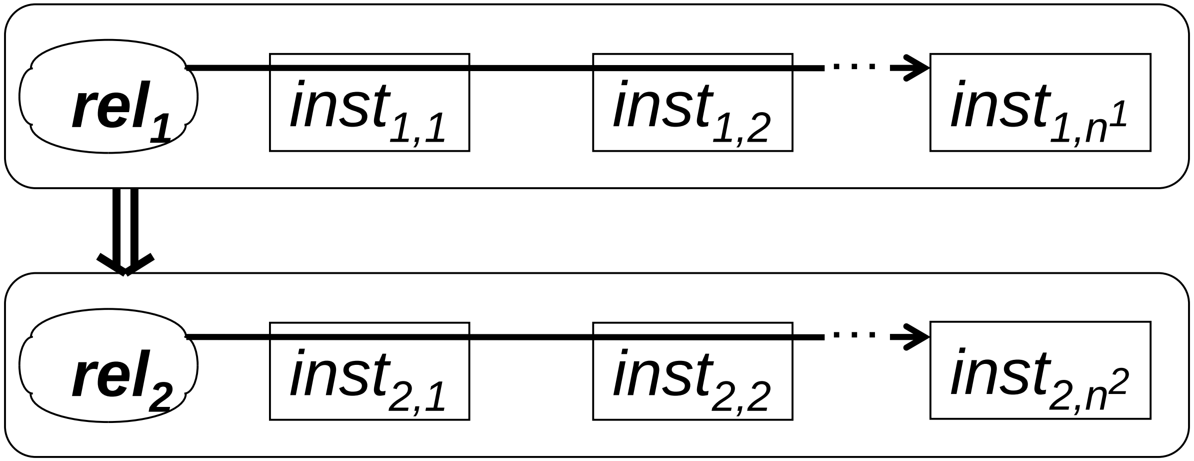
- Equality (Datalog$^+$)



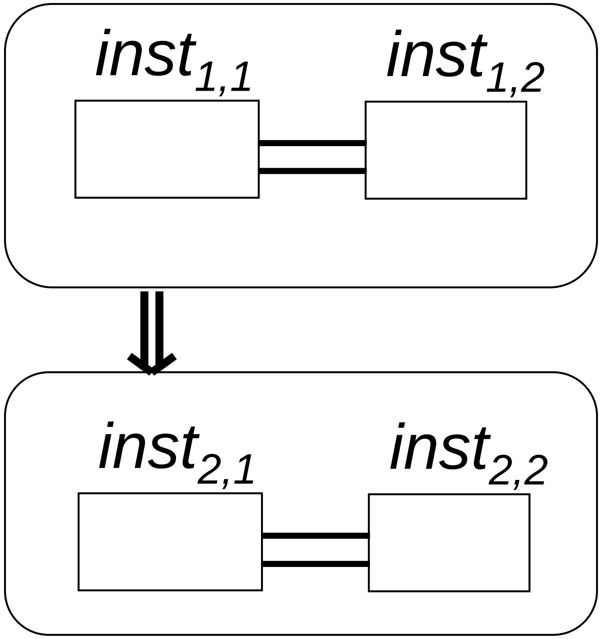
Supported Grailog Elements (2)

Single-premise rules containing:

- n-ary relations (n ≥ 1)



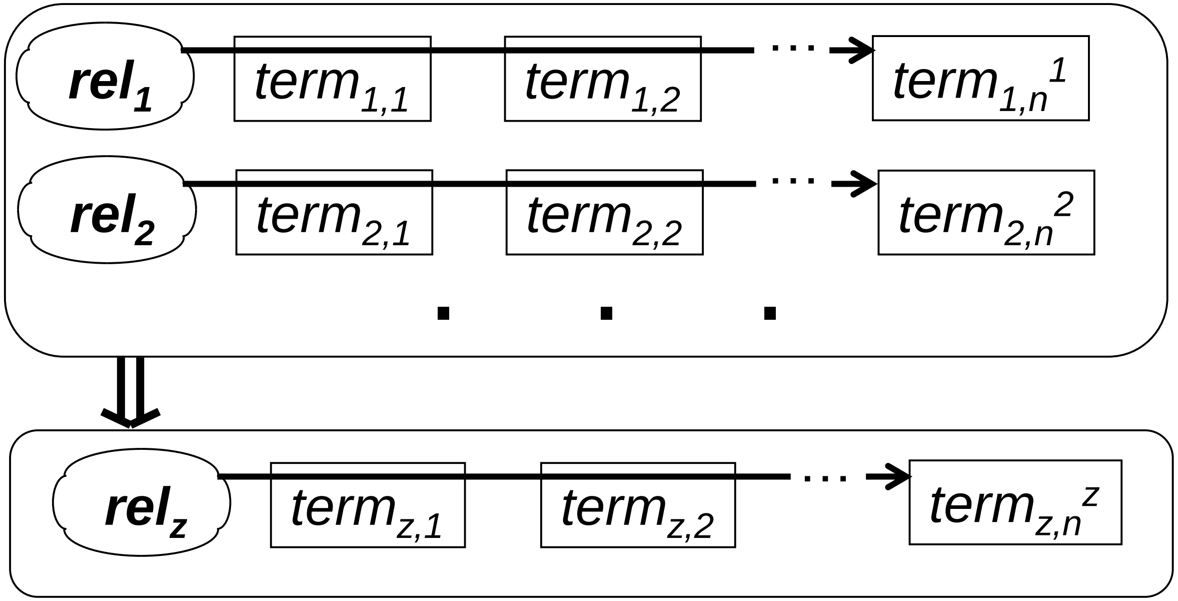
- Equality (Datalog$^+$)



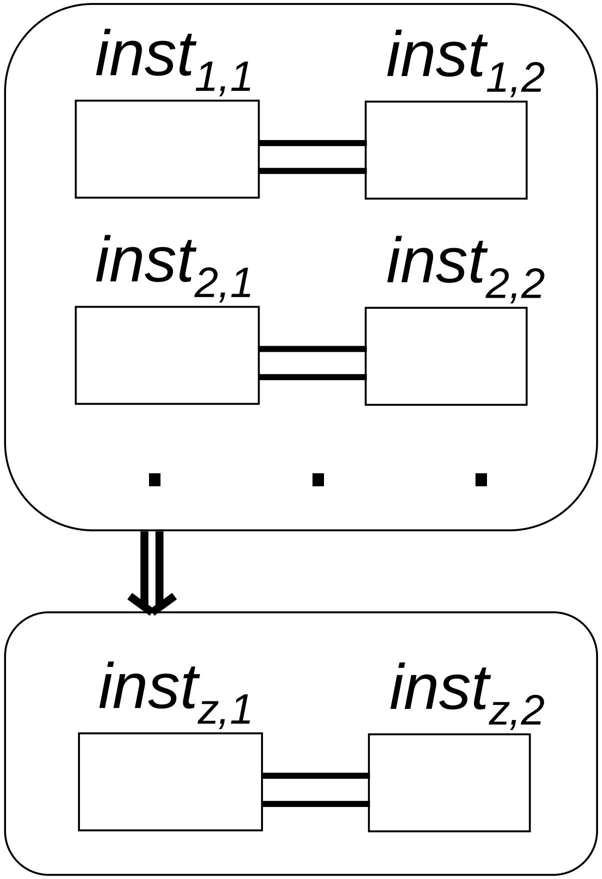
Supported Grailog Elements (3)

Multi-premise rules containing:

- n-ary relations (n ≥ 1)

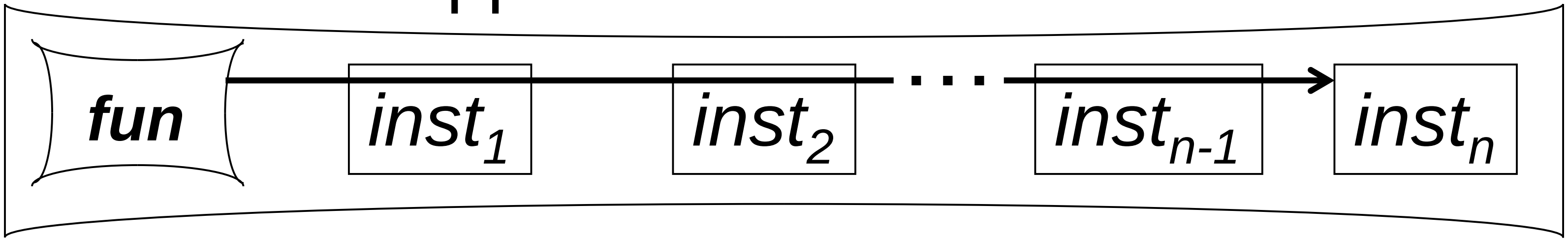


- Equality (Datalog$^+$)

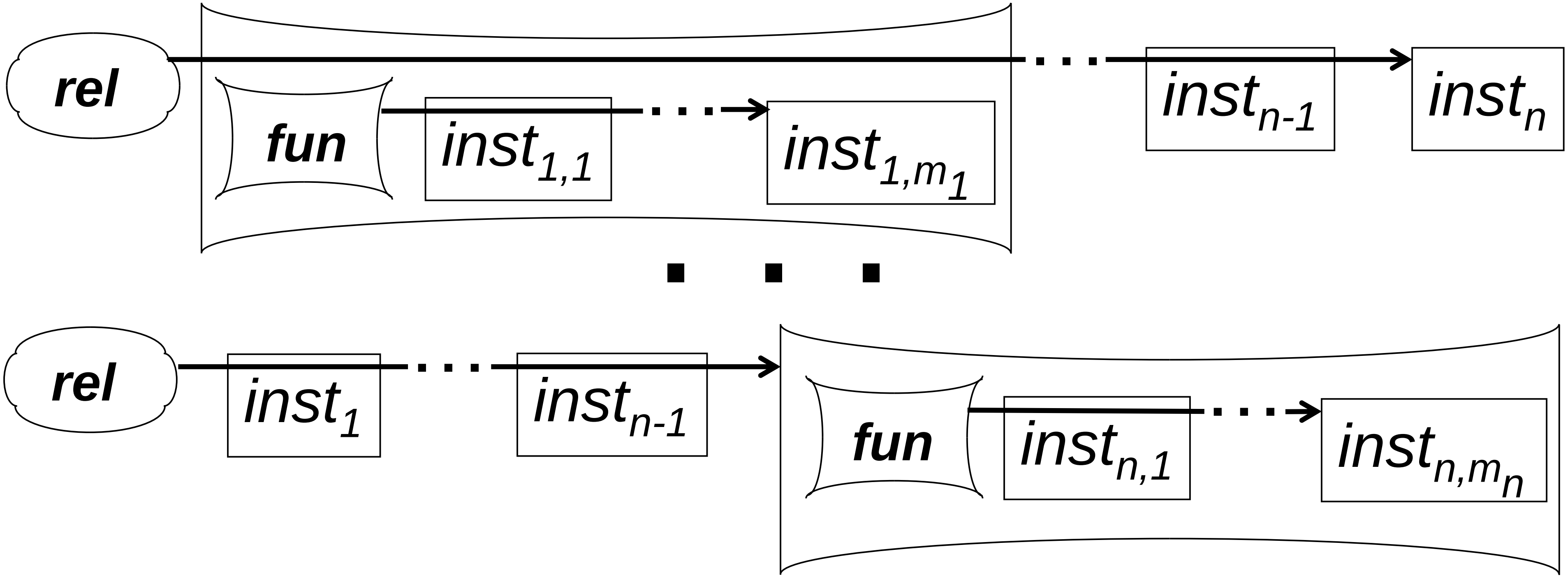


Supported Grailog Elements (4)

- Function Applications

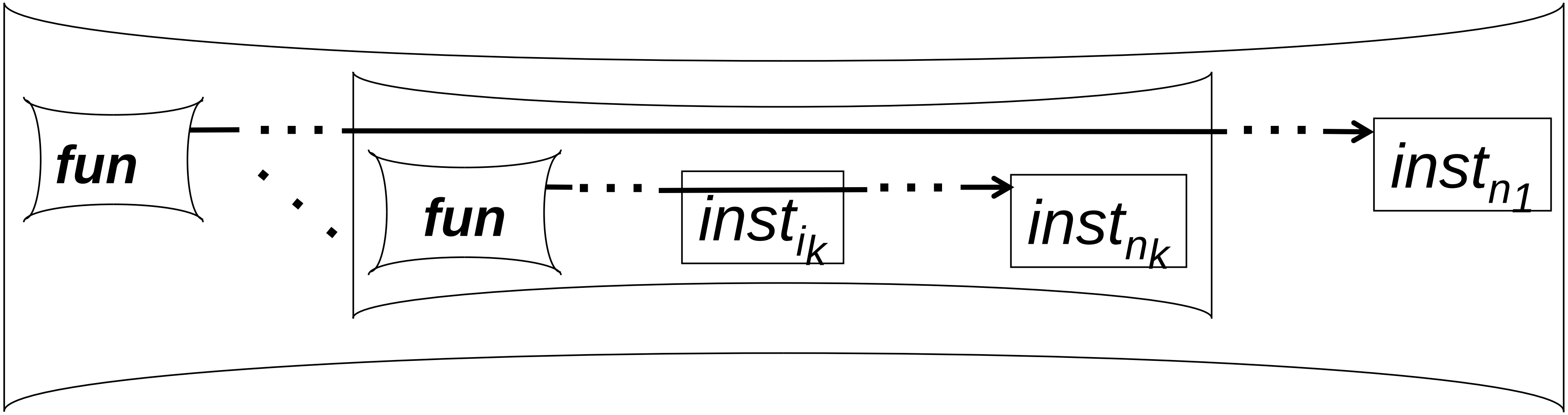


- Predicates with (constructor) function application in any position



Supported Grailog Elements (5)

- Arbitrary Levels of Nested Function Applications



Grailog KS Viz Structure (1)

- Set up SVG file with an initial viewbox to contain the drawings. Dimensions of viewbox are determined using JavaScript
  - Height is determined by a variable that is updated with the last y-coordinate of each new drawing
  - Width is determined by a variable that stores the greatest x-coordinate of all the drawings
- Datalog RuleML/XML files should not contain namespaces in order to be processable

Grailog KS Viz Structure (2)

- SVG
  - Drawings contain text, rectangles, polygons, patterns, straight paths, rounded rectangles, markers, and convex and concave paths using cubic Bézier curve

Grailog KS Viz Structure (3)

- XSLT Template for <Implies>
  - Check for single-premise rule containing child <Atom>: Call <Atom> template
  - Check for single-premise rule containing child <Equal>: Call <Equal> template
  - Check for multi-premise rule: Call <And> template

Grailog KS Viz Structure (4)

- XSLT Template for <And>
  - Determine the number of children of <And>
    - If child is <Atom>, draw n-ary relation (n ≥ 1)
    - If child is <Equal>, draw binary “=“ line
  - Draw <And> contents within surrounding box in the body of a rule

Grailog KS Viz Structure (5)

- XSLT Template for <Atom>
  - If parent:
    - is <Assert>, draw n-ary relation (n ≥ 1)
      in labelnode normal form
    - is <Implies>, draw n-ary relation (n ≥ 1)
      as a single premise or conclusion of the rule
  - If child: is <Expr>, draw nested function application

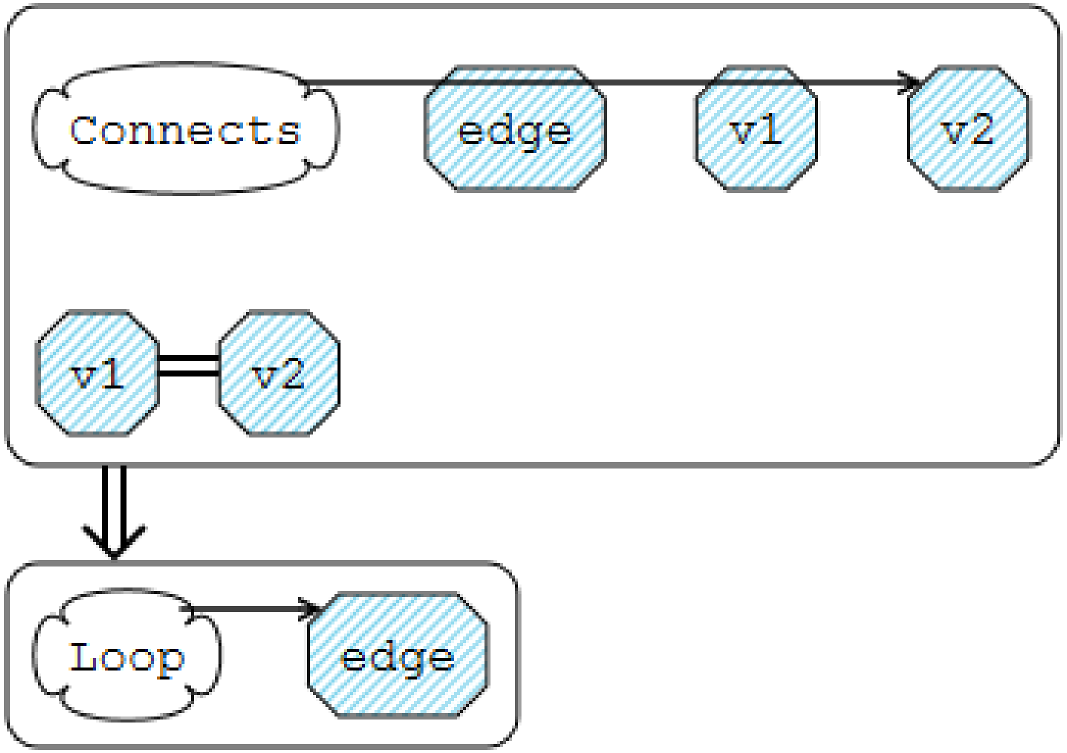
Grailog KS Viz Structure (6)

- Internal JavaScript
  - Calculates the coordinates used in the SVG drawings
  - Updates the variables used to determine the SVG viewbox height and width
- XSLT & JavaScript
  - Create unique variable names used in the SVG drawings by concatenating constants and positions of an element in the XML tree

Datalog$^+$ Example: Two-Premise Rule

- “If an edge connects two vertices and the vertices are the same, then the edge is a loop”

```
<RuleML>
  <Assert>
    <Implies>
      <And>
        <Atom>
          <Rel>Connects</Rel>
          <Var>edge</Var>
          <Var>v1</Var>
          <Var>v2</Var>
        </Atom>
        <Equal>
          <Var>v1</Var>
          <Var>v2</Var>
        </Equal>
      </And>
      <Atom>
        <Rel>Loop</Rel>
        <Var>edge</Var>
      </Atom>
    </Implies>
  </Assert>
</RuleML>
```



Conclusions

- Grailog KS Viz has been extended to the labelnode normal form of Grailog including n-ary (e.g., unary) predicates
- Visualizes Datalog$^+$ with (head) Equality as a binary predicate
- Single- and multi-premise rules may contain any combination of n-ary relations and Equality in the body premise(s) and the head
- Visualizes Hornlog nested function applications, allowing arbitrary levels of nesting
- Tested on usual browsers: Firefox, Chrome, IE
- Instant visualization of 24KB RuleML/XML files
- From Grailog KS Viz 1.0 to Grailog KS Viz 2.0

Future Work

- Automate composition of main and post-processing
- Address any JavaScript security and stability issues in browsers beyond Firefox, Chrome, and IE
- Complete extension for Hornlog's nested function applications with arbitrary levels of nesting and in arbitrary positions simultaneously
- Continue to improve software reusability for future development
- Optional merging of labelnode copies
- Inverse translator, parsing Grailog into RuleML
- Extend for FOL, Higher-Order, Modal, ... RuleML