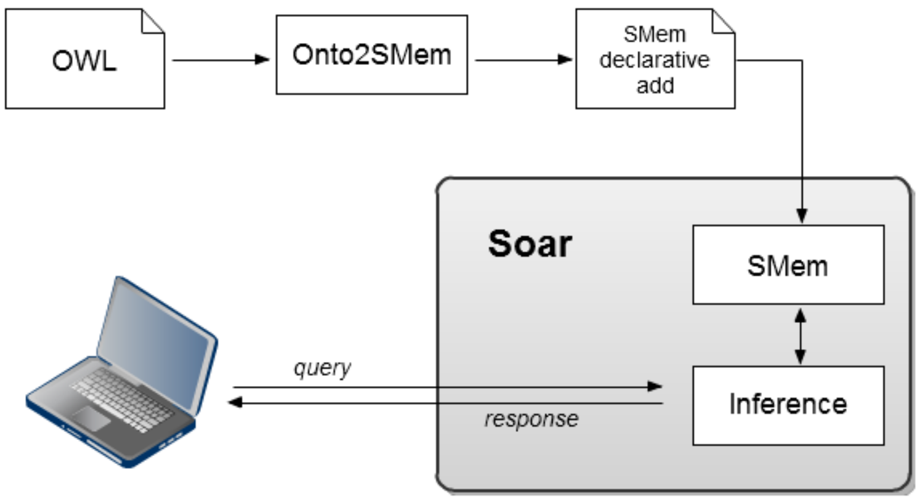
Bryan Smith
May 2010

Representing Ontologies and Reasoning with SMem

Introduction

- Tool (*Onto2SMem*) to generate declarative knowledge base in SMem from ontology
- Sound (if incomplete) inference
  - Proof of concept
  - Baseline implementation



SMem

- Semantic memory (SMem)
  - Store facts about world (declarative)
    - Graph: nodes, augmentations
  - Retrieval and storage
    - Cue- or non-cue –based retrievals
    - Efficient retrievals with activation bias
  - Memory or file (SQLite)

Ontology

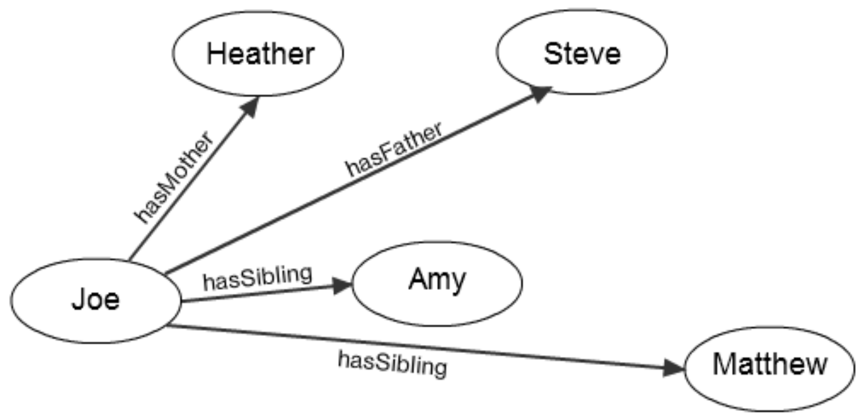
- Formal representation of domain
  - Classes and instances
    - E.g., Steve is an instance of Person
    - Classes have attributes (e.g., name, SSN), restrictions (e.g., Father must have at least one child)
  - Relationships expressed as properties
    - E.g., `isFatherOf(Person,Person)`
    - `isFatherOf(Steve,Matthew)`, so both Steve and Matthew are instances of Person

OWL

- Web Ontology Language
  - Based on descriptive logics
  - Two versions with multiple sublanguages with associated use cases and computational profiles
  - Represented in multiple formats, including XML/RDF
- http://www.w3.org/TR/owl-features/

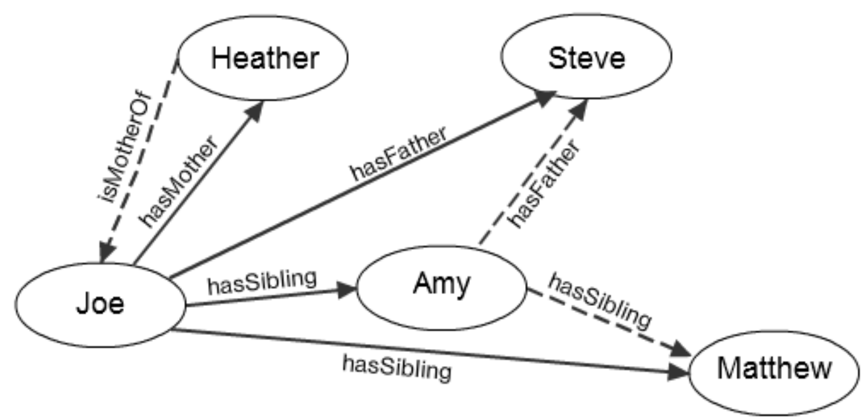
Ontology example

- Simple family example
- Must define
  - Relationships
    - `hasSibling(Person, Person)`
  - instances
    - Joe, Amy
  - instance relationships
    - `hasSibling(Joe,Amy)`



OWL: more than a graph

- Direct assertions easy to query
  - `hasMother(Joe, Heather)`
- Some relationships require inference
  - `isMotherOf(Heather, Joe)`
  - `hasSibling(Amy, Matthew)`
  - `hasFather(Amy,Steve)`



Notes about OWL

- OWL uses *open-world assumption*
  - With OWL, if not verifiably true or false, uncertain
    - Verifiably true if directly asserted or implied
    - Verifiably false if property restrictions imply
- OWL does not use the *unique name assumption*
  - OWL does not assume two names mean two distinct entities
  - Inferred or directly asserted (`sameAs` or `differentFrom`)

OWL features

- OWL 1 and OWL 2 have properties, property chains, property restrictions, quantifiers
  - OWL 1 guide: http://www.w3.org/TR/owl-features/
  - OWL 2 guide: http://www.w3.org/TR/2009/REC-owl2-primer-20091027/
- For our example, interested in:
  - Inverse properties
  - Symmetric properties
  - Transitive properties
  - Property chains
  - These must be preserved when representing ontology in SMem

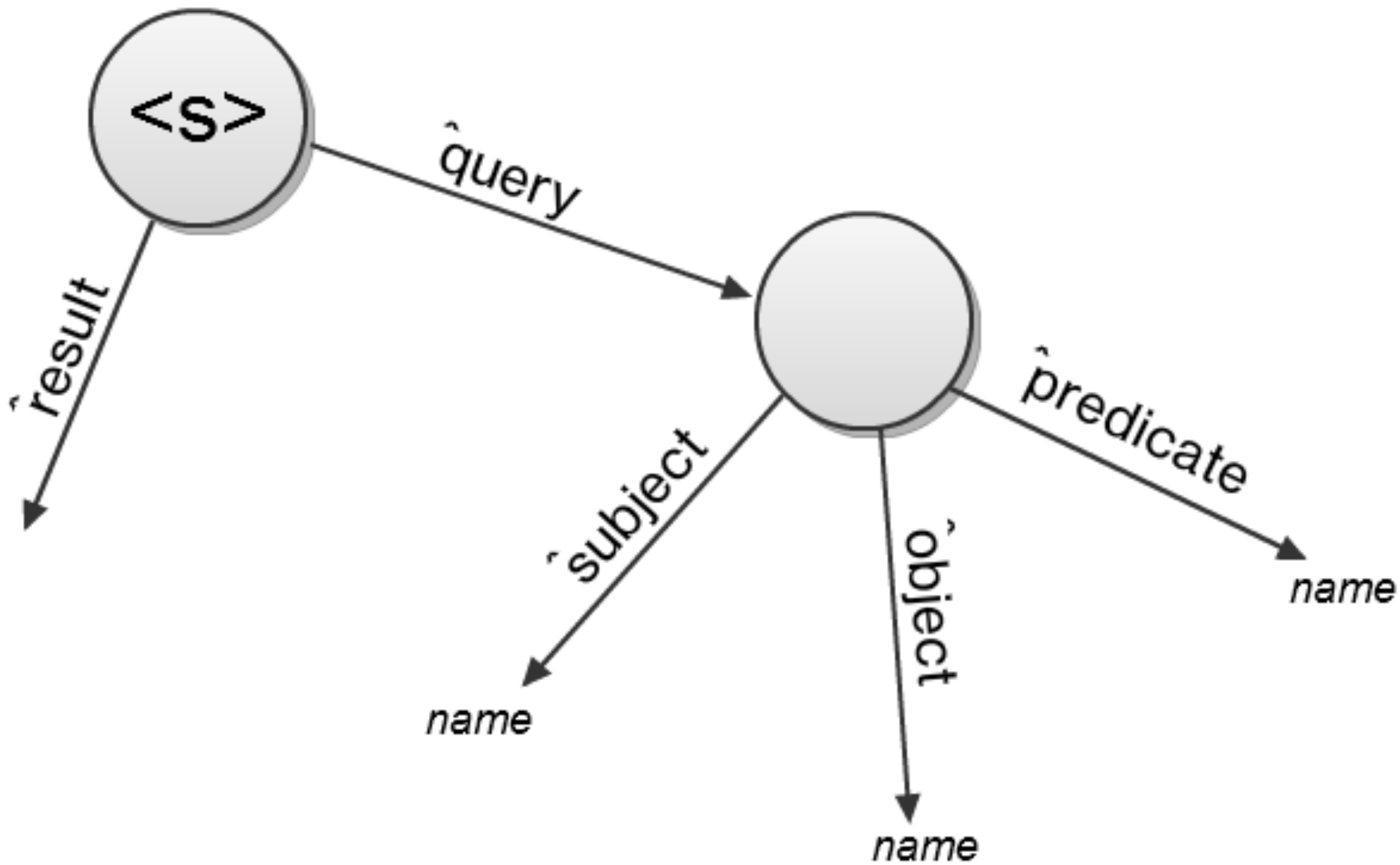
Onto2SMem

- Java utility using Jena framework API
  - Input: OWL file
  - Output: SMem declarative add commands
  - Allows use of existing ontologies in SMem
  - Preserves properties and arbitrary graph structure
  - Adds supporting collections useful for inference
- Onto2SMem
  - http://bryanesmith.com/soar/inference/
- Jena
  - http://jena.sourceforge.net/

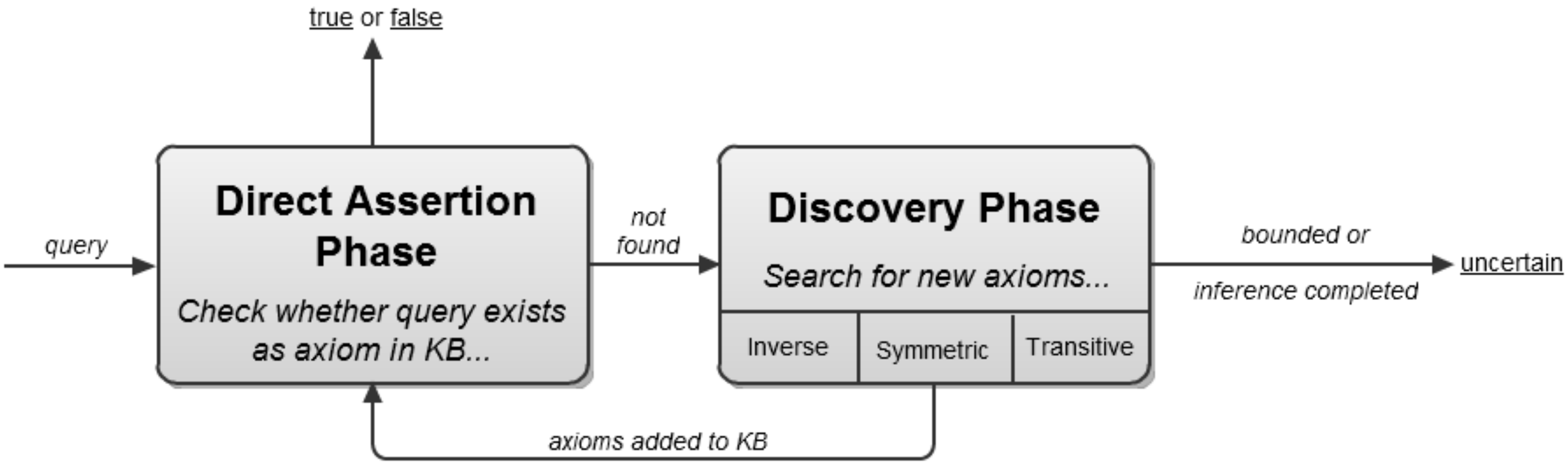
Inference with SMem

- Requirement: sound if incomplete
- Domain independent
  - Works with KB generated by Onto2SMem
- Implemented in Soar agent space
- Useful subset of OWL features
  - Inverse properties
  - Symmetric properties
  - Transitive properties
  - Property chains (not implemented)

Inference with SMem: interface



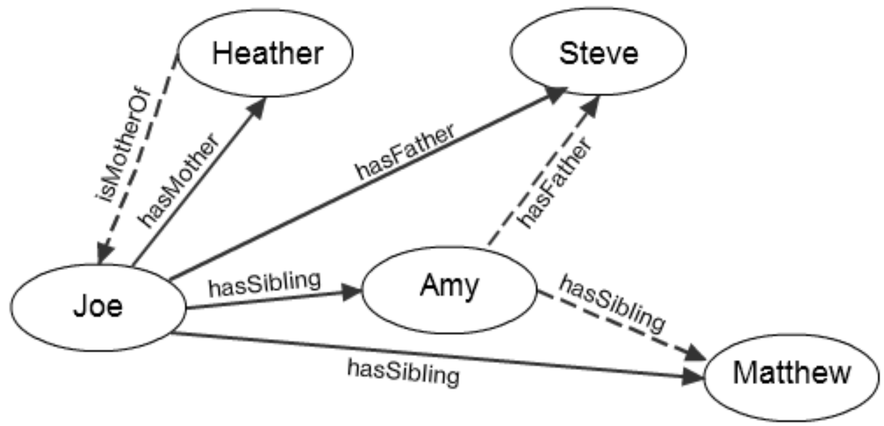
Inference with SMem: how it works



- Forward chaining
  - Iterative discovery of new axioms
- Unbounded
  - Bounded searches could be implemented

Initial demo: family relationships

- Using simple family ontology, inference tool finds missing relationships
- Subsequent runs using tool take require fewer decisions
  - First run: 497 decisions/10 true queries
  - Second run: 345 decisions/ 10 true queries
  - Replacing two true queries with uncertain resulted in approximately 4K queries



Ontology: http://www.bryanesmith.com/ontologies/family-example.owl

Inference hypothesis

- Expecting polynomial or hyper-polynomial decision growth for inference as number of axioms increases
  - Due to transitive property check, which is $O(n^2)$
  - Might be ontology-dependent

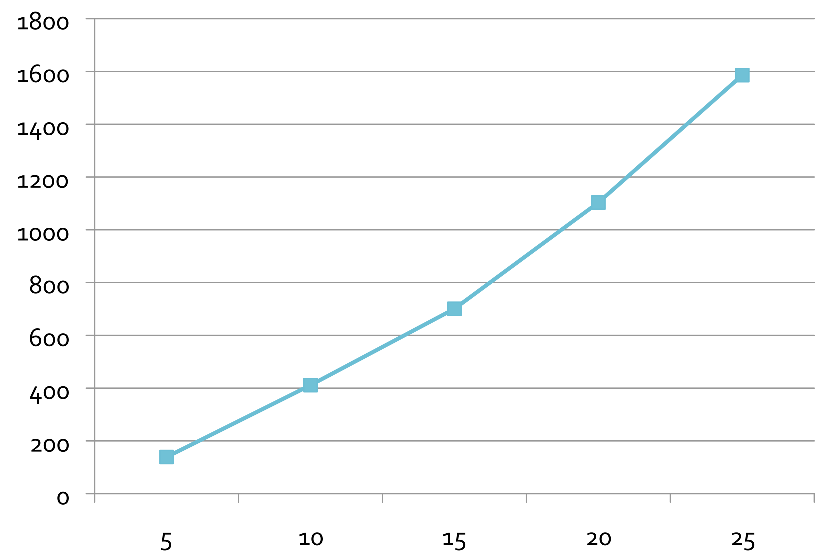
Preliminary testing

- What are the number of decisions required for agent to fulfill tasks?
  - Using 100 queries what is the impact of changing the number of families?
- Run agent twice
  - First run: inference + direct assertions
  - Second run: just direct assertions
    - Has "compiled" KB from first run

Inference and growth

- Number of steps spent for inference appears polynomial with relation to elements in KB
  - Additional data points verified this trend, but…
    - discarded due to bug exposed in larger KB

Decision for inference



Data are averages of three runs per condition.

Thoughts

- (At least) polynomial decision growth, but more testing needed
  - Verify trend with more data
  - Determine whether growth trend is ontology-specific

Recommendations for inference

1. Instead of single general-purpose inference engine, use cases with restrictions to guide implementations
   - Consider OWL sublanguages
   - Carefully crafted ontologies with certain DL properties can be much more efficient
     - Require knowledge of DL and efficient inference implementation
2. Introduce bounded searches
   - Optional parameter to limit total inference cycles
   - Default to unbounded search

Improving inference performance

1. During exploratory phase, terminate early if find result for query
   - Laziest approach most efficient
2. Use reinforcement learning for task ordering in exploratory phase
   - Reward based on number of new axioms found
   - Works with KBs with certain trends in the types of relationships
     - Assuming order impacts total number of inference cycles require

Improving inference performance

3. Perform inference offline and use compiled KB with agent
   - SMem can store KB to disk
   - Still requires some estimation to determine whether feasible

Conclusions

- Can represent ontologies in Soar using the SMem module
  - Preserve the semantics
  - Perform inference
- General-purpose inference is expensive
  - Use cases and restrictions to guide inference module development
  - Bounded searches and forward chaining provide value if agent can defer when uncertain

Nuggets versus coal

NUGGETS

- Preliminary feasibility demonstrated
- Domain and task independent
  - Reusable
- Sound
- Efficient queries after inference complete

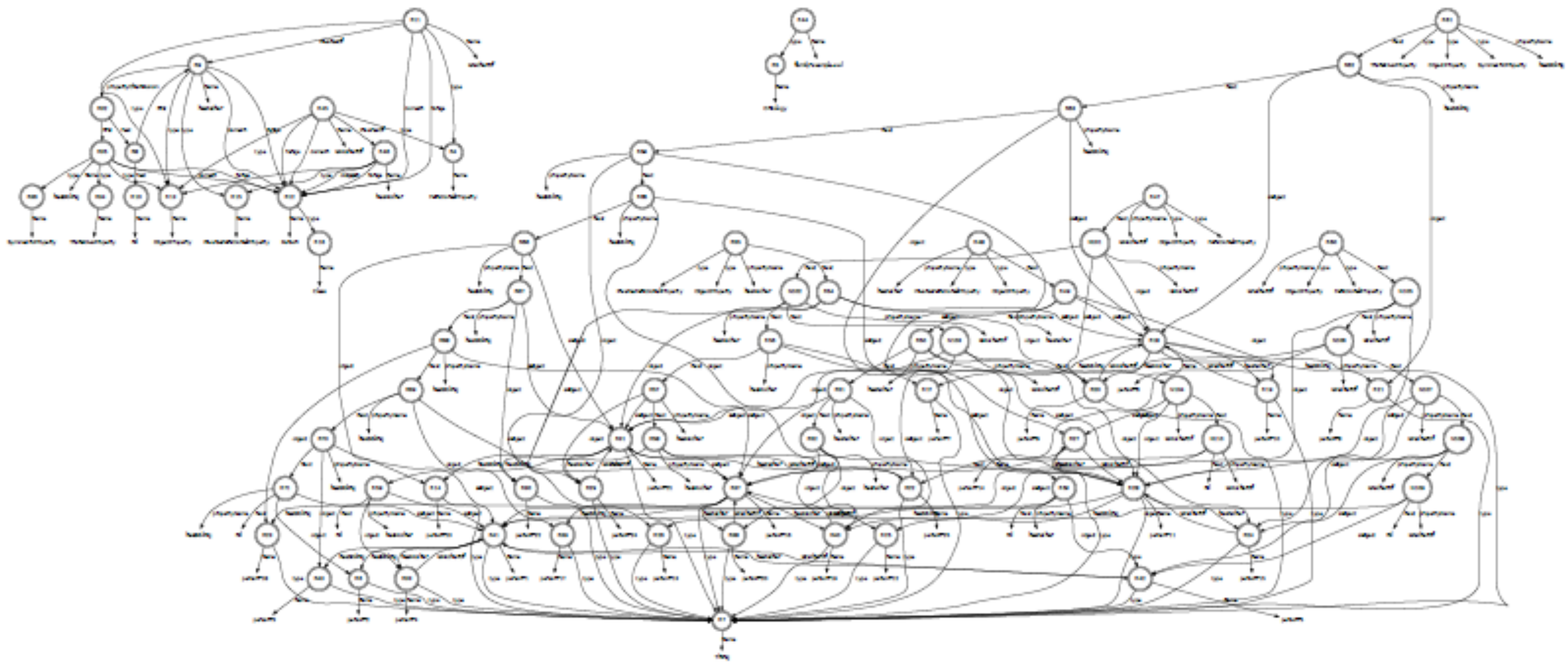
COAL

- Incomplete (subset of features)
- Unbounded with polynomial (or worse) growth for inference tasks
- Preliminary testing with few data points
  - Much more testing needed

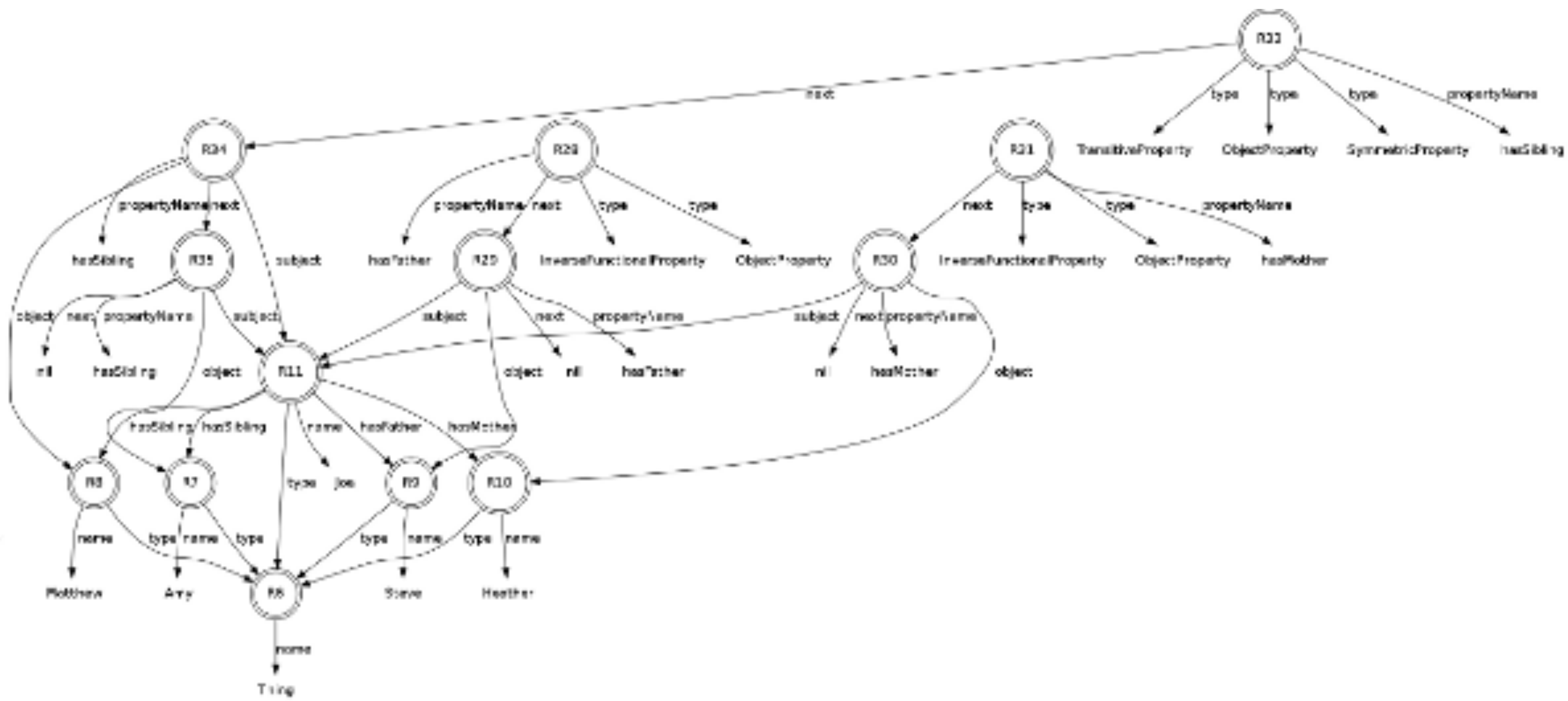
Questions?

- Onto2SMem, sample ontology, Soar agent with inference, these slides, and other tools:
  - http://bryanesmith.com/soar/inference/
- Jena (OWL Java API)
  - http://jena.sourceforge.net/
- Protégé (ontology editor)
  - http://protege.stanford.edu/

Thanks!

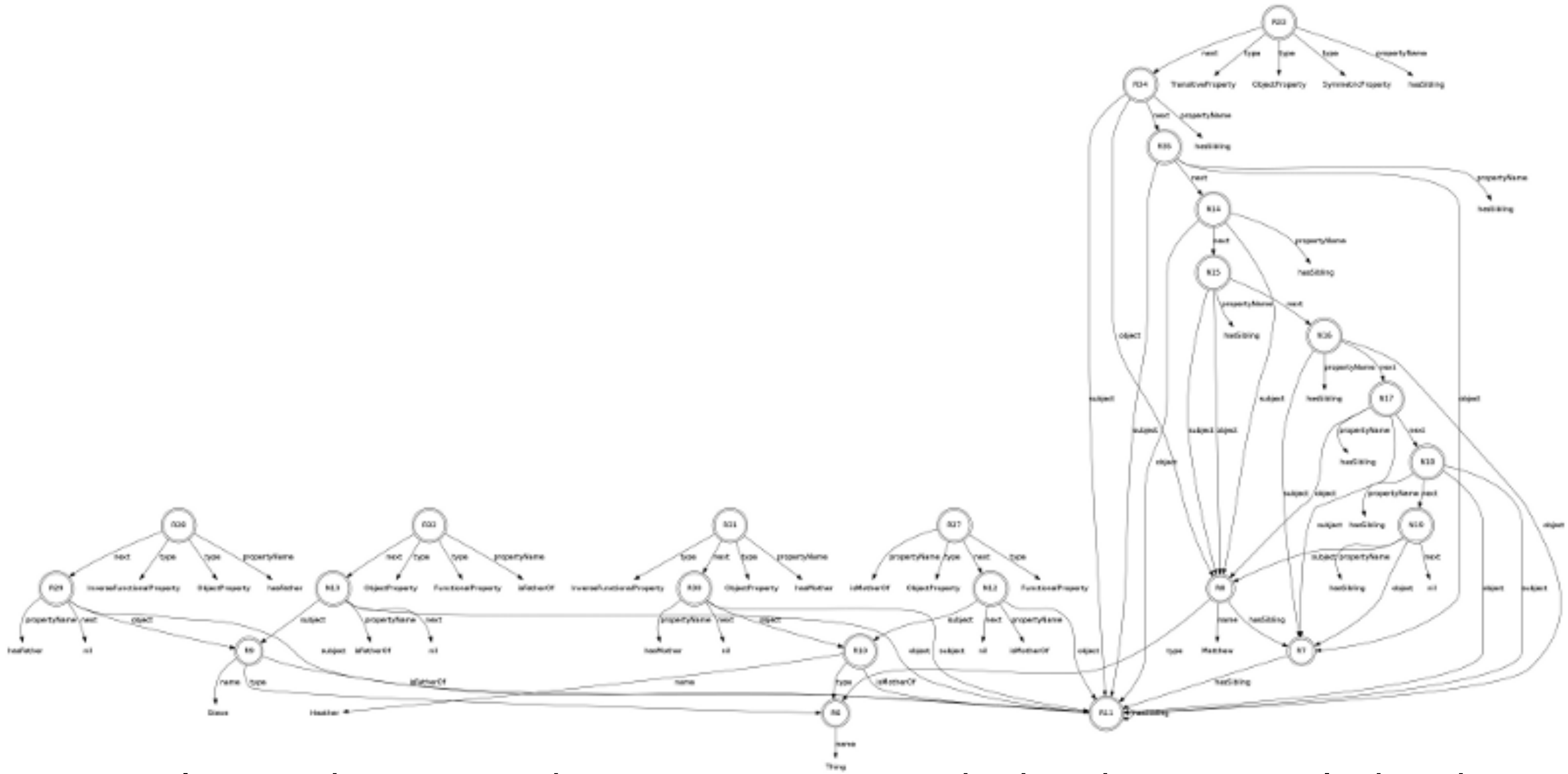


Appendix:
Sample family KB before inference



Note: Class and property descriptions removed. This diagram includes the instances and supporting data structures used for inference.

Appendix: Sample family KB after inference



Note: Class and property descriptions removed. This diagram includes the instances and supporting data structures used for inference.

Appendix: Inference with inverse properties

- Properties can be defined as inverse
  - `isMotherOf(A,B)` and `hasMother(B,A)`
  - `isFatherOf(A,B)` and `hasFather(B,A)`
- Functional relationships (e.g., `isFatherOf`) and inverse functional relations (e.g., `hasFather`) useful restrictions
  - Entity matching

Appendix: OWL and inverse properties

```
<owl:ObjectProperty
   rdf:about="#isMotherOf"> <rdf:type
   rdf:resource="&owl;FunctionalProperty"/>
   <rdfs:range rdf:resource="#Person"/>
   <rdfs:domain rdf:resource="#Person"/>
   <owl:inverseOf rdf:resource="#hasMother"/>
</owl:ObjectProperty>
```

Appendix: Inference with symmetric property

- Property is symmetric if property(A,B) → property(B,A)
  - `hasSibling(Joe,Amy) → hasSibling(Amy,Joe)`

Appendix: OWL and symmetric property

```
<owl:ObjectProperty rdf:about="#hasSibling">
  <rdf:type
  rdf:resource="&owl;SymmetricProperty"/>
  <rdf:type
  rdf:resource="&owl;TransitiveProperty"/>
  <rdfs:domain rdf:resource="#Person"/>
  <rdfs:range rdf:resource="#Person"/>
</owl:ObjectProperty>
```

Appendix: Inference with transitive property

- Property is transitive if property(A,B), property(B,C) → property(A,C)
  - `hasSibling(Joe,Amy), hasSibling(Amy,Matthew) → hasSibling(Joe,Matthew)`

Appendix: OWL and transitive property

```
<owl:ObjectProperty rdf:about="#hasSibling">
  <rdf:type
  rdf:resource="&owl;SymmetricProperty"/>
  <rdf:type
  rdf:resource="&owl;TransitiveProperty"/>
  <rdfs:domain rdf:resource="#Person"/>
  <rdfs:range rdf:resource="#Person"/>
</owl:ObjectProperty>
```

Appendix: Inference with property chain

- $property_1(e_1, e_2), property_2(e_2, e_3), \ldots, property_M(e_{N-1}, e_N) \rightarrow property(e_1, e_N)$
  - `hasFather(a,b), hasSister(b,c) → hasAunt(a,c)`
  - `hasSibling(a,b), hasFather(b,c) → hasFather(a,c)`

Appendix: OWL and property chain

```
<owl:ObjectProperty rdf:about="#hasFather">
   <rdf:type
   rdf:resource="&owl;InverseFunctionalProperty"/>
   <rdfs:domain rdf:resource="#Person"/>
   <rdfs:range rdf:resource="#Person"/>
   <owl:propertyChainAxiom rdf:parseType="Collection">

      <rdf:Description rdf:about="#hasSibling"/>
      <rdf:Description rdf:about="#hasFather"/>
   </owl:propertyChainAxiom>
</owl:ObjectProperty>
```

Appendix: Difference between OWL 1 and 2

- OWL 2 added:
  - Property chains
  - Asymmetric, reflexive, and disjoint properties
  - Qualified cardinality
  - etc.
- More information: http://www.w3.org/TR/2009/WD-owl2-overview-20090327/#New_Features

Appendix:
OWL 1 sublanguages

- OWL Full
  - Unrestricted
  - Not decidable
- OWL DL
  - Disjointness between classes and instances
  - Axioms complete, form “tree-like structure”
  - Others
- OWL Lite
  - Forbidden constructs (e.g., oneOf, unionOf, disjointWith, etc)
  - Basically support “subclasses and property restrictions”
- More information: http://www.w3.org/TR/owl-ref/#Sublanguage-def

Appendix: OWL 2 sublanguages (profiles)

- OWL 2 EL
  - Useful for large number of classes and properties
  - Existential quantification
- OWL 2 QL
  - Designed for conjunctive queries with instances
    - LOGSPACE
  - Highly restricted
- OWL 2 RL
  - Restrictions that permit polynomial-time growth with rule-based reasoners (if-then)
- More information: http://www.w3.org/TR/owl2-profiles/

Appendix: Inference profile

- Loading state (25)
- Direct assertion (9)
- Adding axiom (23)
  - Plus reload state (23+25=48)
- Inverse properties (14+8*n*)
  - *n* = number of inverse properties
- Transitive properties (4 + *n*[8(*n*-*m*) + 14*m*])
  - *n* =number of axioms involving transitive properties
  - *m* = number of axiom pairs with transitive alignment
- Symmetric (2+8*n*)
  - *n* = number of symmetric properties