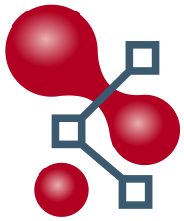
Exploration of SWRL Rule Bases through Visualization, Paraphrasing, and Categorization of Rules

Saeed Hassanpour, Martin J. O'Connor and Amar K. Das

Stanford Center for Biomedical Informatics Research
MSOB X215, 251 Campus Drive, Stanford, California, USA.



{saeedhp, martin.oconnor, amar.das}@stanford.edu

Why Do We Need Methods for Rule Management?

- Increasing use of rules in ontologies
- Increasing size of these rule bases
- Lack of tools for rule management

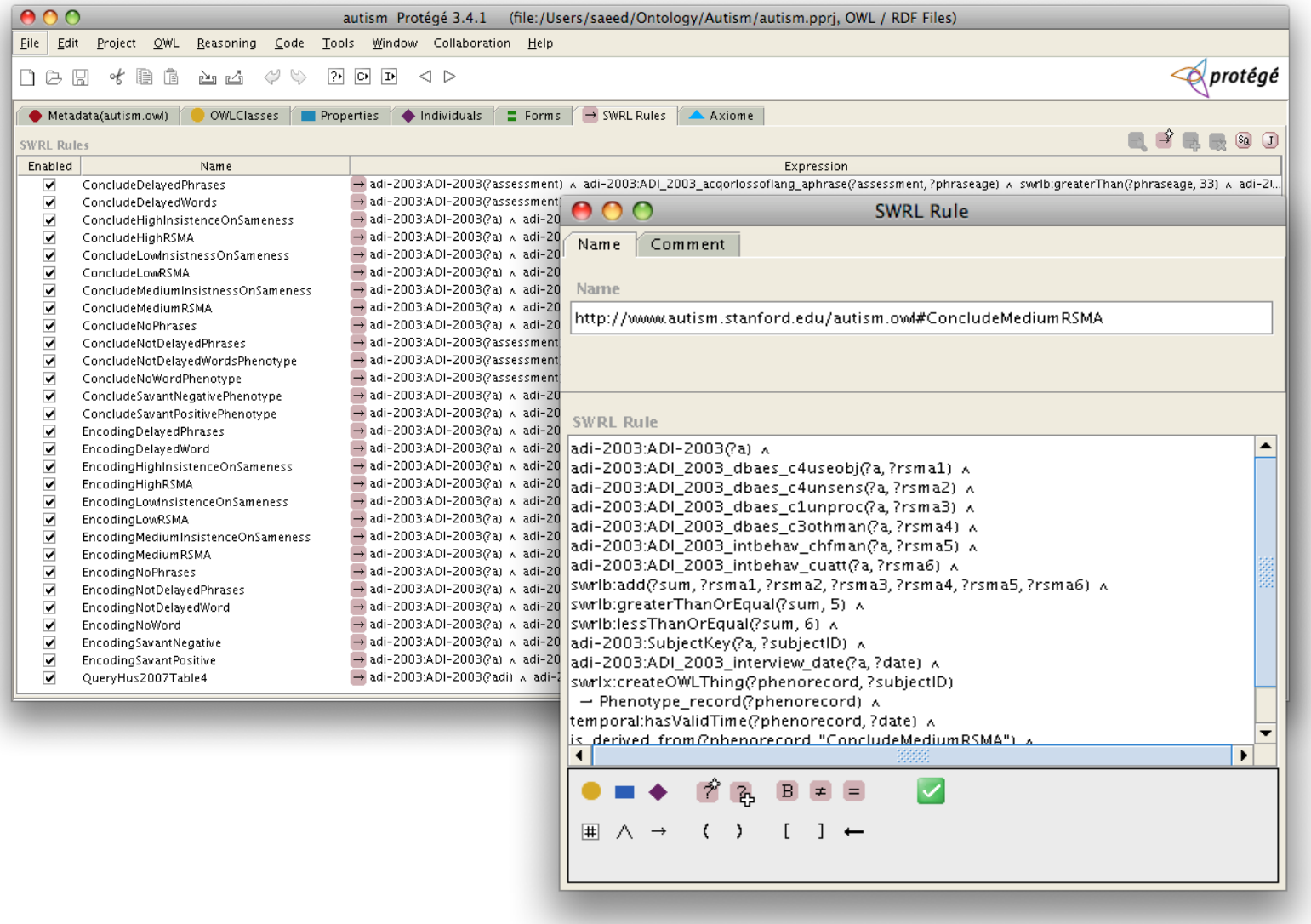
Previous Work

- Business rule management
- Association rule mining
- Rule argumentation
- Expert systems

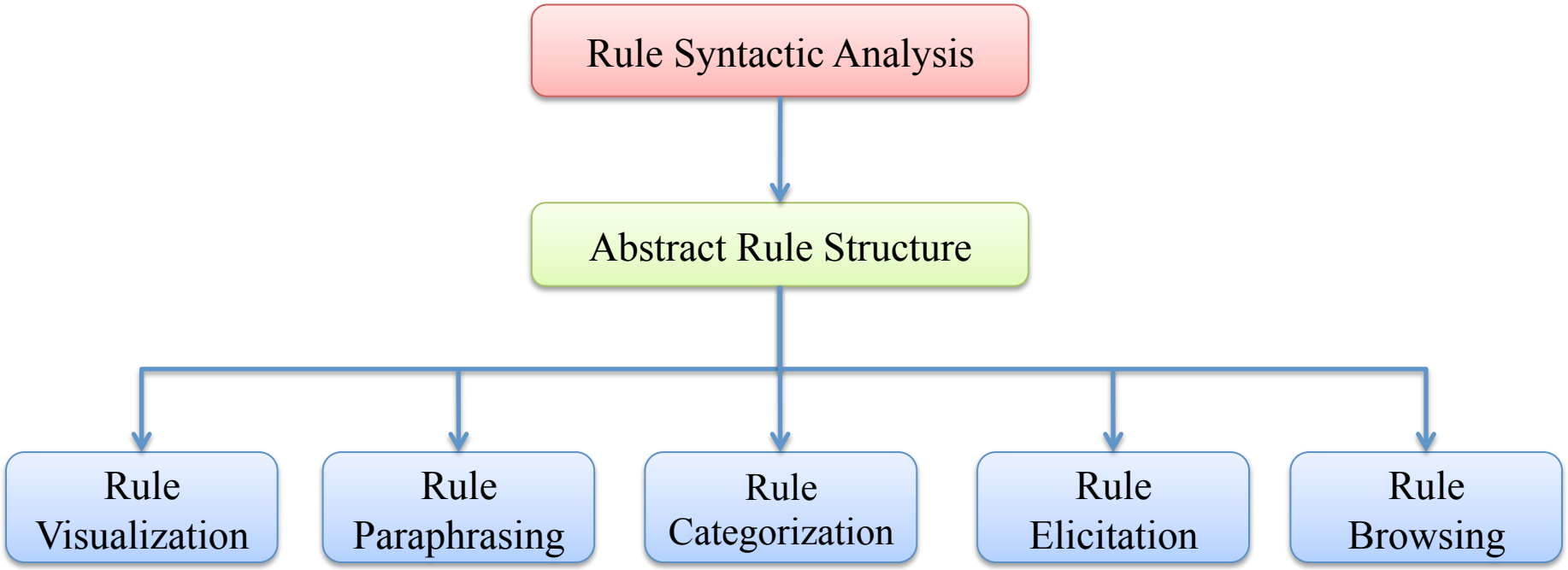
Semantic Web Rule Language (SWRL)

- SWRL is intended to be the rule language of the Semantic Web.
- SWRL is based on a high-level abstract syntax for Horn-like rules.
- All rules are expressed in terms of OWL concepts (classes, properties, individuals).
- SWRL rules are maintained as part of OWL ontologies.

Rule Base Complexity



Rule Management Methodology



Example SWRL Rule

Individuals under the age of 18 as a potential driver of a vehicle with a weight of less than 26,000 lbs are able to drive in California if they possess an out-of-state driver's license and are visiting the state for less than 10 days.

Person(?p) ^ has_Driver_License(?p,?d) ^ issued_in_state_of(?d,?s) ^
swrlb:notEqual(?s,"CA") ^ hasAge(?p,?g) ^
swrlb:lessThan(?g,18) ^ Number_of_Visiting_Days_in_CA(?p,?x) ^
swrlb:lessThan(?x,10) ^ Car(?c) ^
has_weight_in_lbs(?c,?w) ^ swrlb:lessThan(?w,26000)
→ Can_Drive(?p,?c)

SWRL Atom Types

SWRL Atom Type	Example Atom
Class atom	`Person(?p), Car(?c)`

Individual property atom	`has_Driver_License(?p,?d)`
	`issued_in_State_of(?d,?s)`
	`can_Drive(?p,?c)`

Same/Different atom	`sameAs(?x, ?y)` `differentFrom(?x, ?y)`

Datavalued property atom	`has_Age(?p,?g)`
	`number_of_Visiting_Days_in_CA(?p,?x)`
	`has_Weight_in_lbs(?c,?w)`

Built-in atom	`swrlb:notEqual(?s,"CA")` `swrlb:lessThan(?g,18)`
Data range atom	`xsd:double(?x)`

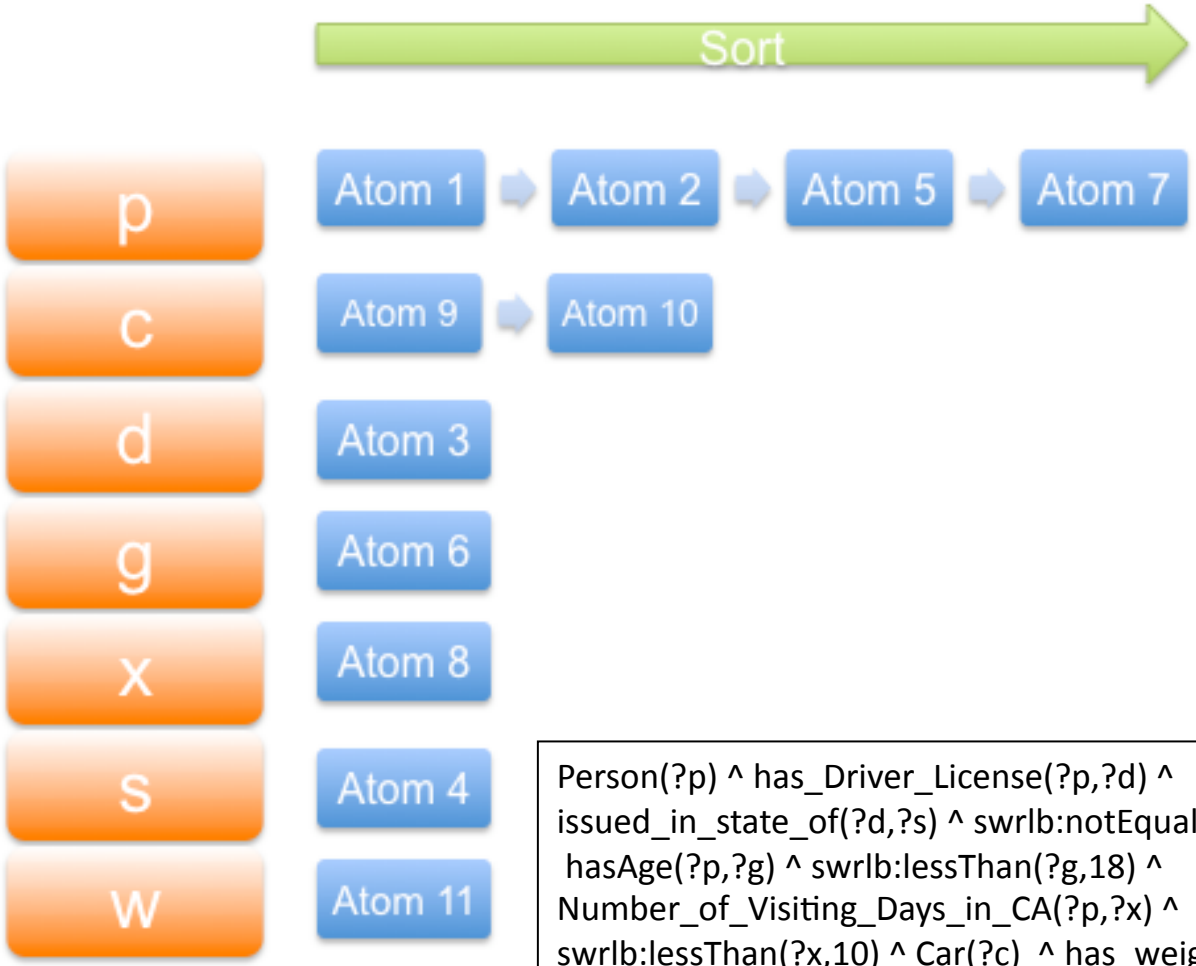
Rule Syntactic Analysis

- Atom prioritization
- Sorting atoms
- Rule atom chaining

Atom Prioritization

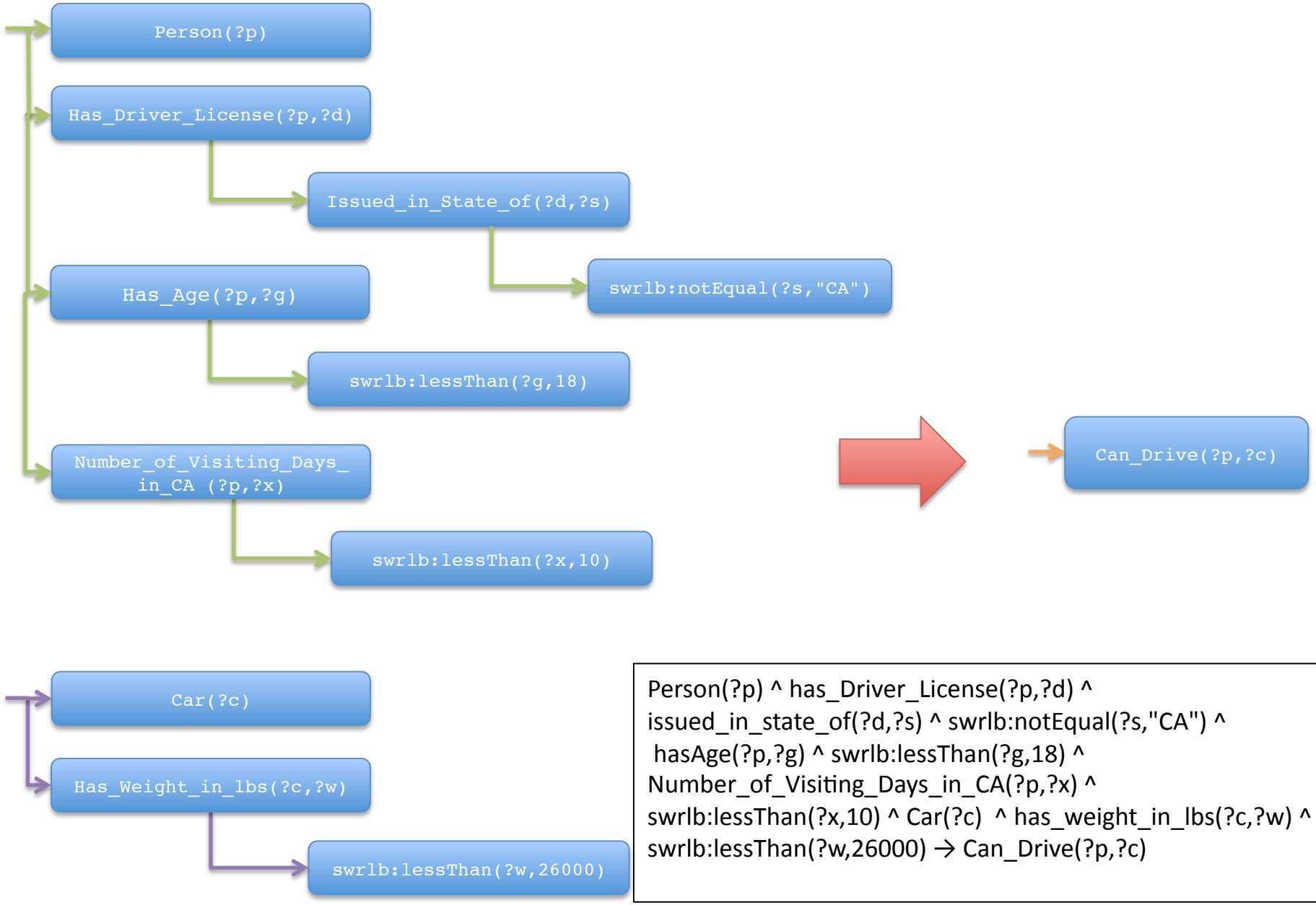
SWRL Atom Type	Priority
Class atom	1
Individual property atom	2
Same/Different atom	3
Data-valued property atom	4
Built-in atom	5
Data range atom	6

Sorting Atoms



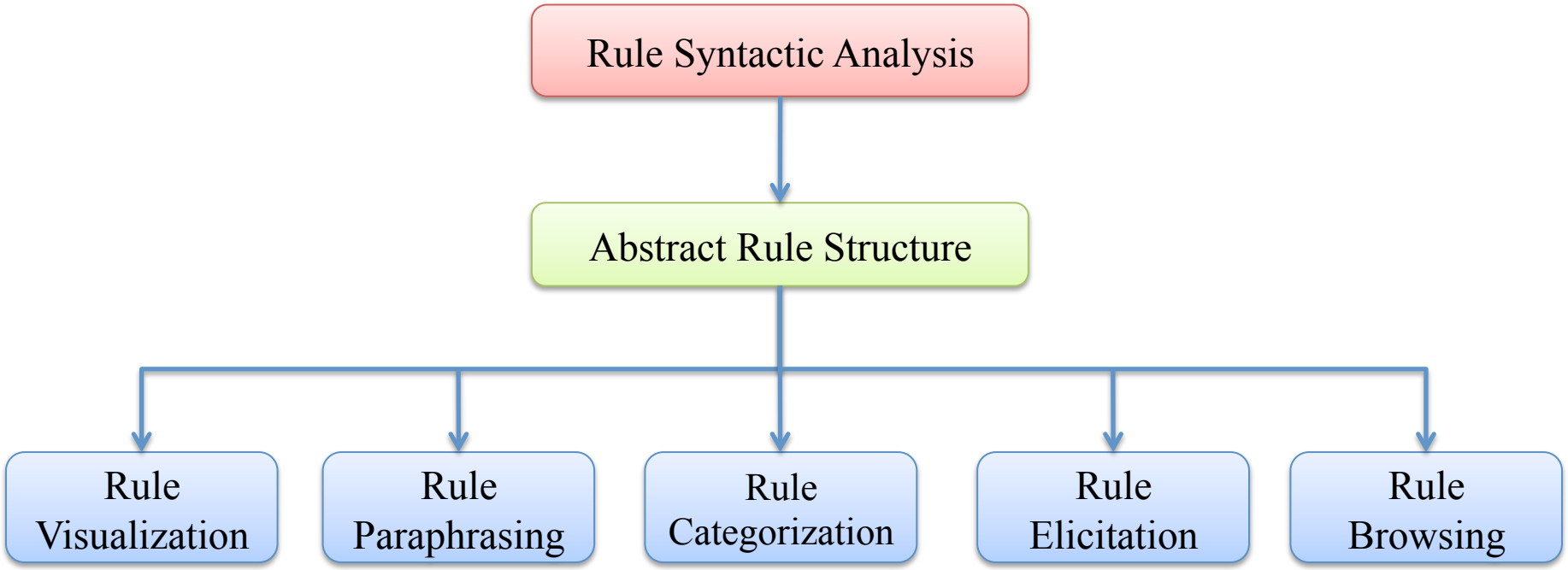
Person(?p) ^ has_Driver_License(?p,?d) ^
issued_in_state_of(?d,?s) ^ swrlb:notEqual(?s,"CA") ^
hasAge(?p,?g) ^ swrlb:lessThan(?g,18) ^
Number_of_Visiting_Days_in_CA(?p,?x) ^
swrlb:lessThan(?x,10) ^ Car(?c) ^ has_weight_in_lbs(?c,?w) ^
swrlb:lessThan(?w,26000) → Can_Drive(?p,?c)

Rule Atom Chaining

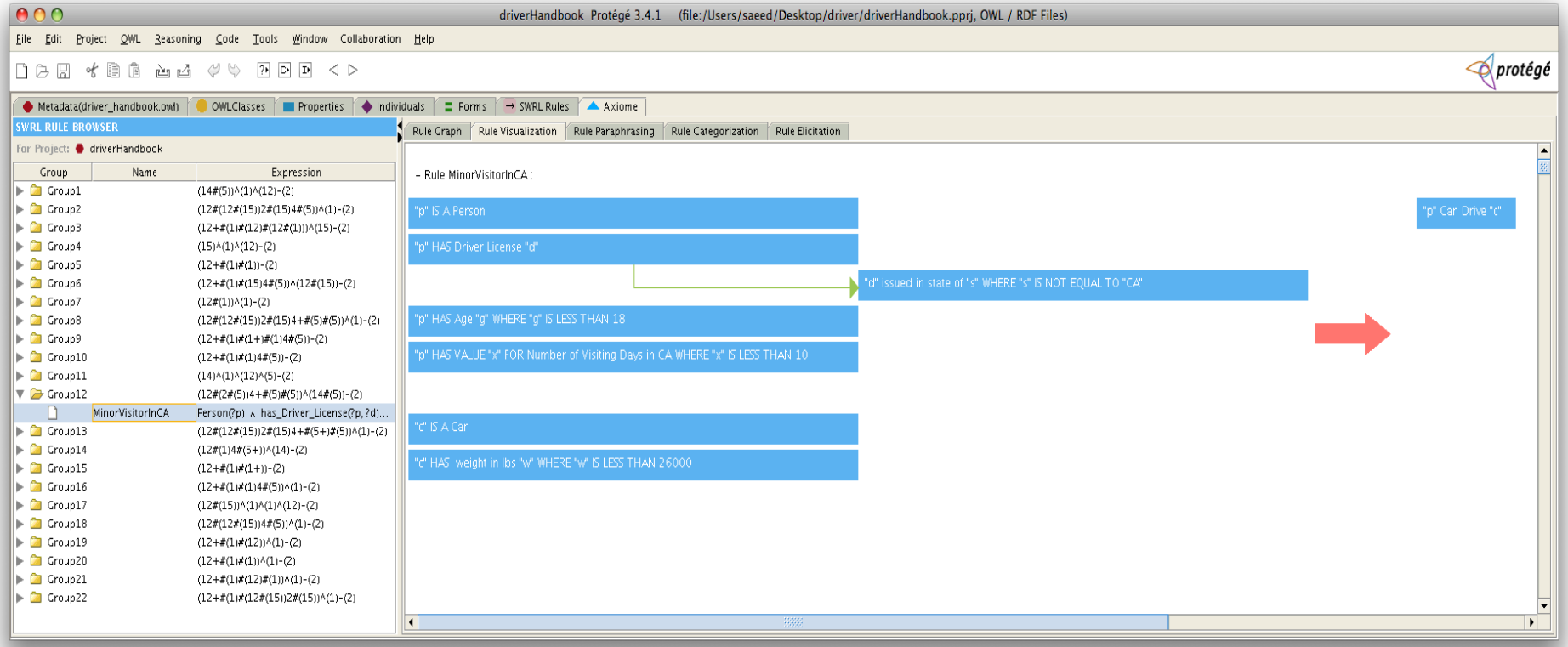


Person(?p) ^ has_Driver_License(?p,?d) ^
issued_in_state_of(?d,?s) ^ swrlb:notEqual(?s,"CA") ^
hasAge(?p,?g) ^ swrlb:lessThan(?g,18) ^
Number_of_Visiting_Days_in_CA(?p,?x) ^
swrlb:lessThan(?x,10) ^ Car(?c) ^ has_weight_in_lbs(?c,?w) ^
swrlb:lessThan(?w,26000) → Can_Drive(?p,?c)

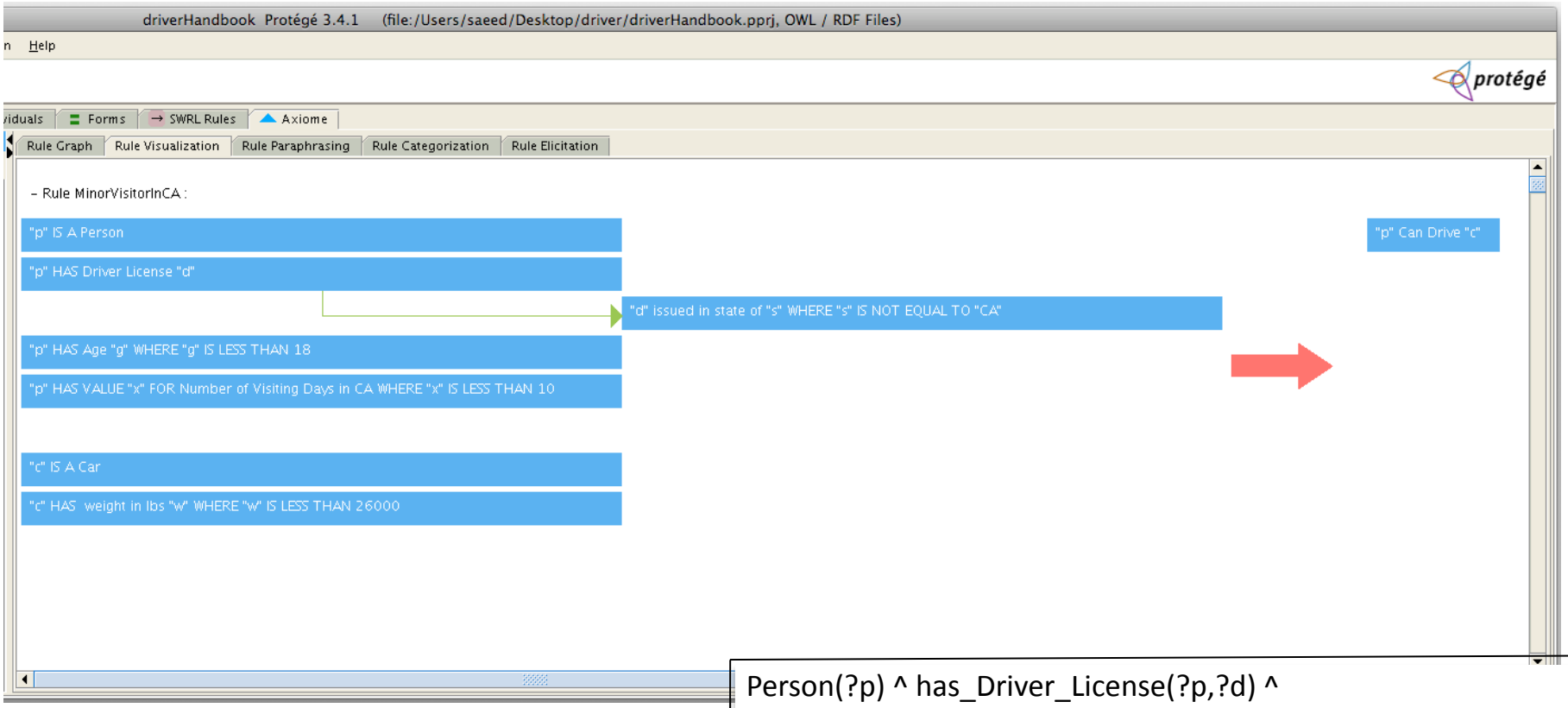
Rule Management Methodology



Axiomé


driverHandbook Protégé 3.4.1 (file:/Users/saeed/Desktop/driver/driverHandbook.pprj, OWL / RDF Files)
File Edit Project OWL Reasoning Code Tools Window Collaboration Help
protégé
Metadata(driver_handbook.owl) OWLClasses Properties Individuals Forms SWRL Rules Axiome
SWRL RULE BROWSER
For Project: driverHandbook
Group Name Expression
Group1 (14#(5))^(1)^(12)-(2)
Group2 (12#(12#(15))2#(15)4#(5))^(1)-(2)
Group3 (12+#(1)#(12)#(12#(1)))^(15)-(2)
Group4 (15)^(1)^(12)-(2)
Group5 (12+#(1)#(1))-(2)
Group6 (12+#(1)#(15)4#(5))^(12#(15))-(2)
Group7 (12#(1))^(1)-(2)
Group8 (12#(12#(15))2#(15)4+#(5)#(5))^(1)-(2)
Group9 (12+#(1)#(1+)#(1)4#(5))-(2)
Group10 (12+#(1)#(1)4#(5))-(2)
Group11 (14)^(1)^(12)^(5)-(2)
Group12 (12#(2#(5))4+#(5)#(5))^(14#(5))-(2)
MinorVisitorInCA Person(?p) ∧ has_Driver_License(?p,?d)...
Group13 (12#(12#(15))2#(15)4+#(5+)#(5))^(1)-(2)
Group14 (12#(1)4#(5+))^(14)-(2)
Group15 (12+#(1)#(1+))-(2)
Group16 (12+#(1)#(1)4#(5))^(1)-(2)
Group17 (12#(15))^(1)^(1)^(12)-(2)
Group18 (12#(12#(15))4#(5))^(1)-(2)
Group19 (12+#(1)#(12))^(1)-(2)
Group20 (12+#(1)#(1))^(1)-(2)
Group21 (12+#(1)#(12)#(1))^(1)-(2)
Group22 (12+#(1)#(12#(15))2#(15))^(1)-(2)
Rule Graph Rule Visualization Rule Paraphrasing Rule Categorization Rule Elicitation
- Rule MinorVisitorInCA :
"p" IS A Person
"p" HAS Driver License "d"
"d" issued in state of "s" WHERE "s" IS NOT EQUAL TO "CA"
"p" HAS Age "g" WHERE "g" IS LESS THAN 18
"p" HAS VALUE "x" FOR Number of Visiting Days in CA WHERE "x" IS LESS THAN 10
"c" IS A Car
"c" HAS weight in lbs "w" WHERE "w" IS LESS THAN 26000
"p" Can Drive "c"

Rule Visualization

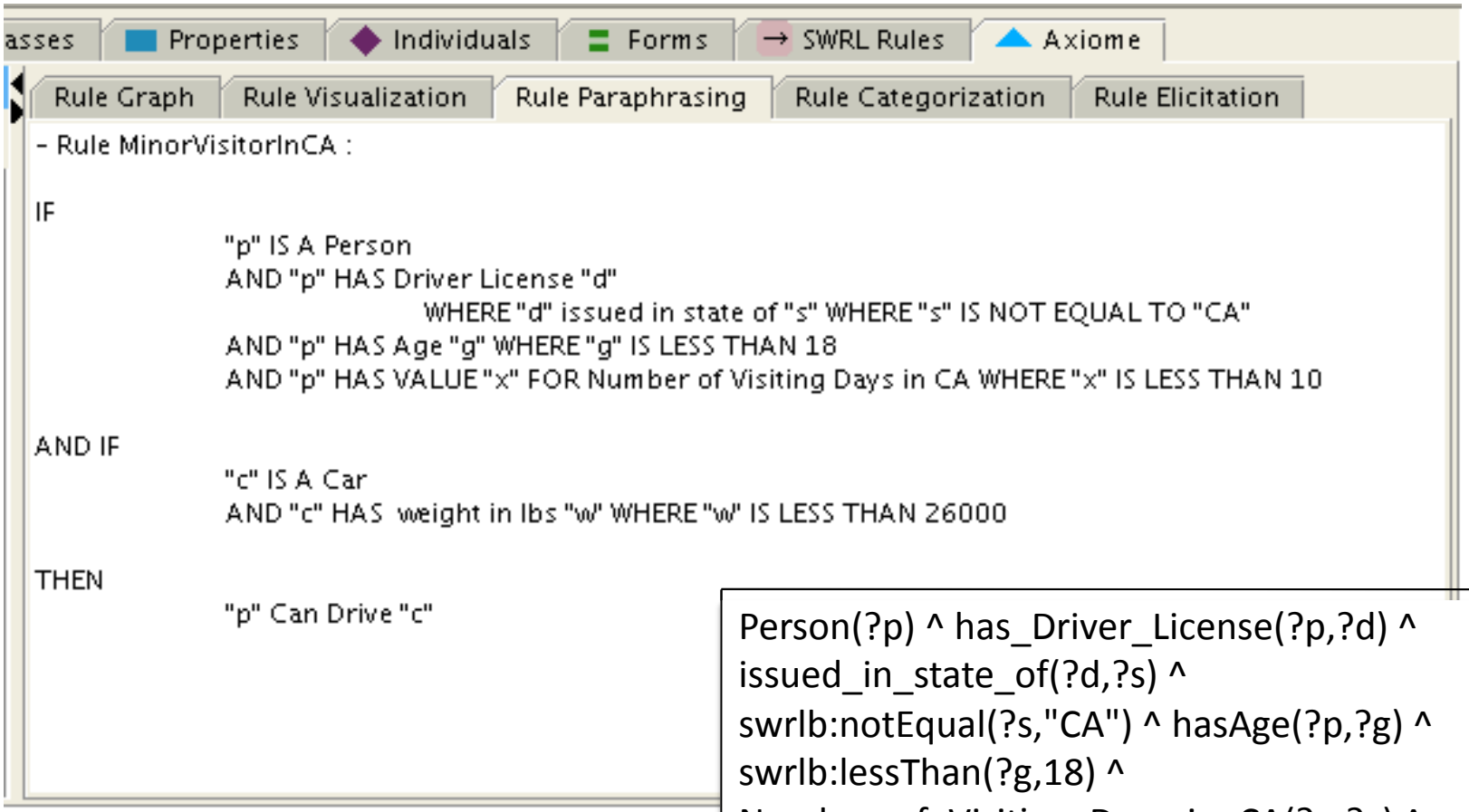


Person(?p) ^ has_Driver_License(?p,?d) ^
issued_in_state_of(?d,?s) ^ swrlb:notEqual(?s,"CA") ^
hasAge(?p,?g) ^ swrlb:lessThan(?g,18) ^
Number_of_Visiting_Days_in_CA(?p,?x) ^
swrlb:lessThan(?x,10) ^ Car(?c) ^ has_weight_in_lbs(?c,?w) ^
swrlb:lessThan(?w,26000) → Can_Drive(?p,?c)

Rule Paraphrasing

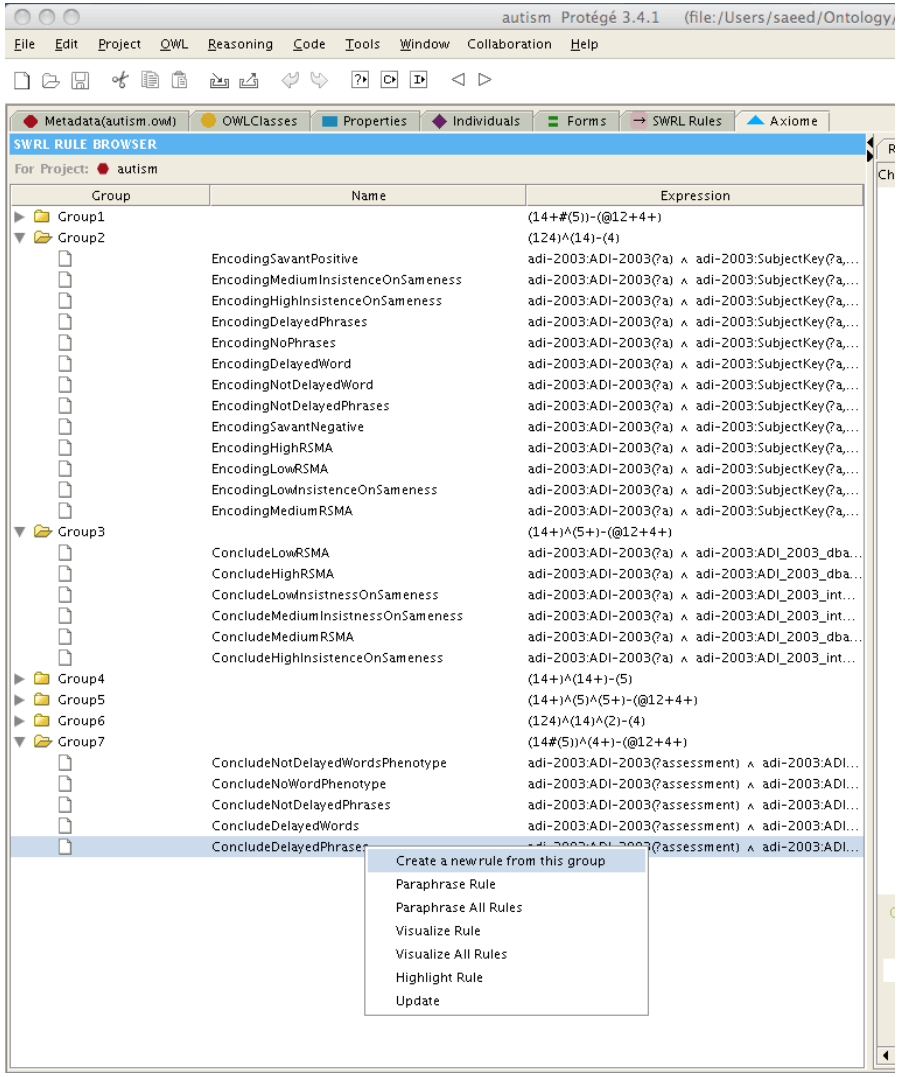
- Use the rule's tree data structure
- Use different heuristics such as:
  - Annotations for built-ins atoms
  - Atoms' name analysis
  - Appropriate articles and conjunctions
  - Indentation

Rule Paraphrasing



Person(?p) ^ has_Driver_License(?p,?d) ^
issued_in_state_of(?d,?s) ^
swrlb:notEqual(?s,"CA") ^ hasAge(?p,?g) ^
swrlb:lessThan(?g,18) ^
Number_of_Visiting_Days_in_CA(?p,?x) ^
swrlb:lessThan(?x,10) ^ Car(?c) ^
has_weight_in_lbs(?c,?w) ^
swrlb:lessThan(?w,26000) → Can_Drive(?p,c)

Rule Base Categorization



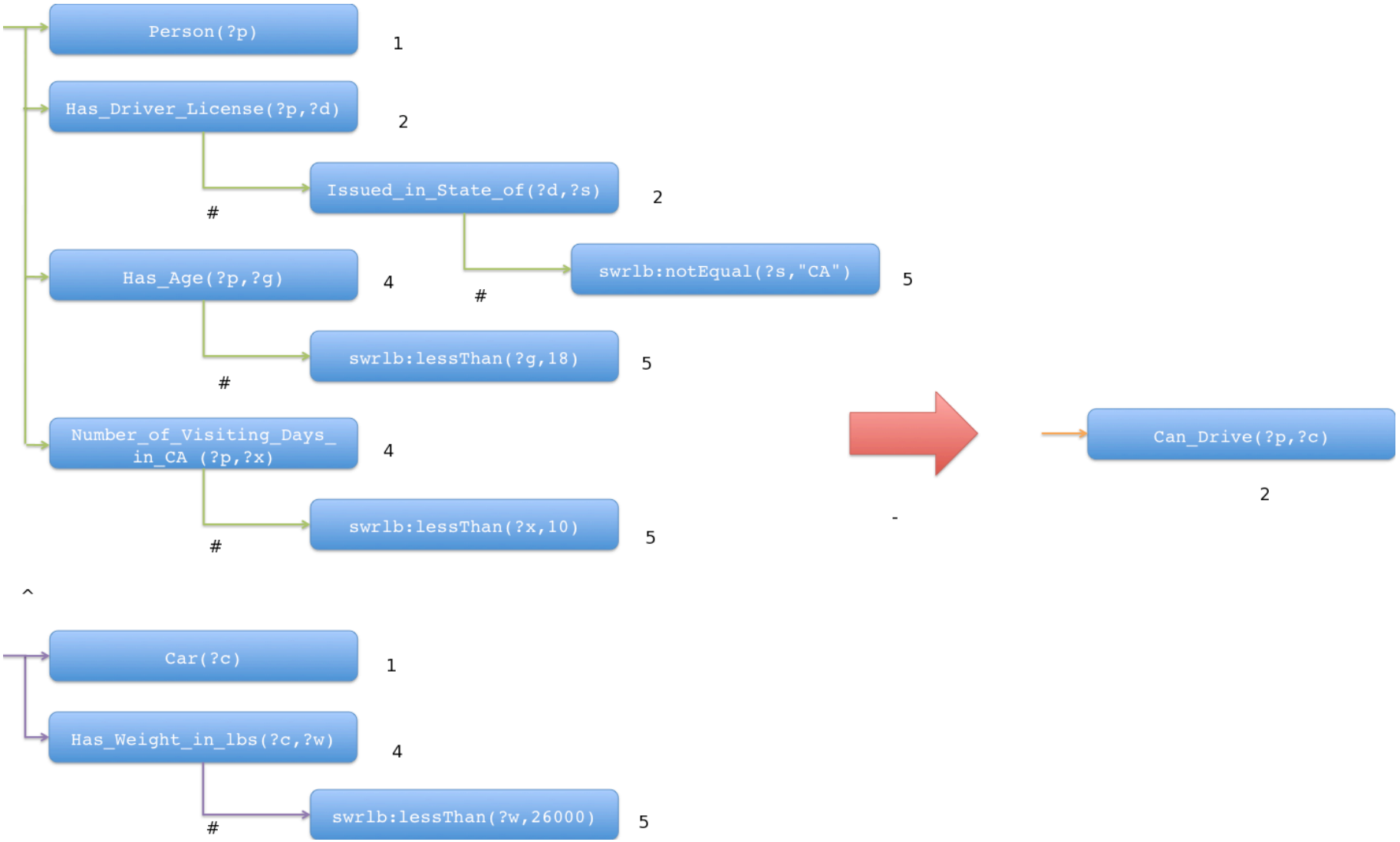
Rule Categorization

$\Sigma = \{1, 2, 3, 4, 5, 6\}$ *SWRL Atom Types*

$Q = \{-, \hat{}, (\ ), \#, +\}$ *Rule Quantifiers*

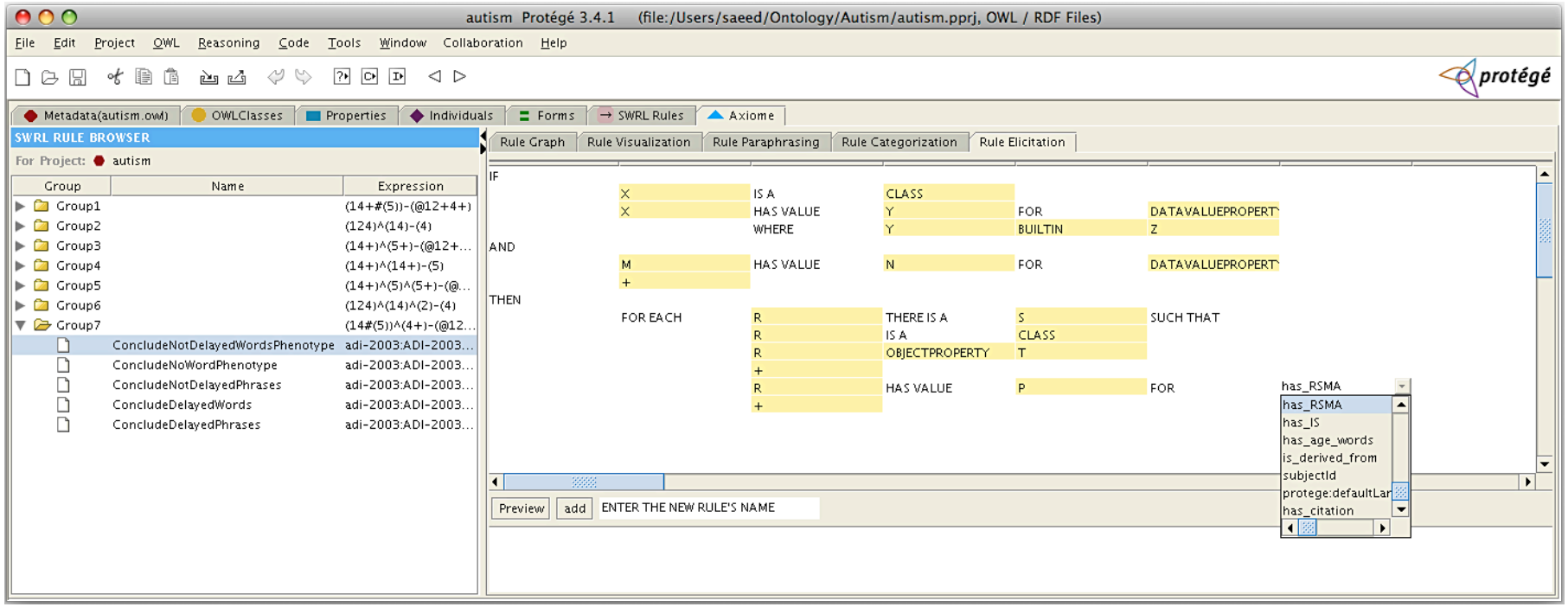
Rule Quantifier	**Role**
-	Body-Head separator
^	Tree separator
( )	Direct descendents of a node
#	Node expansion
+	Repetition

Rule Signature

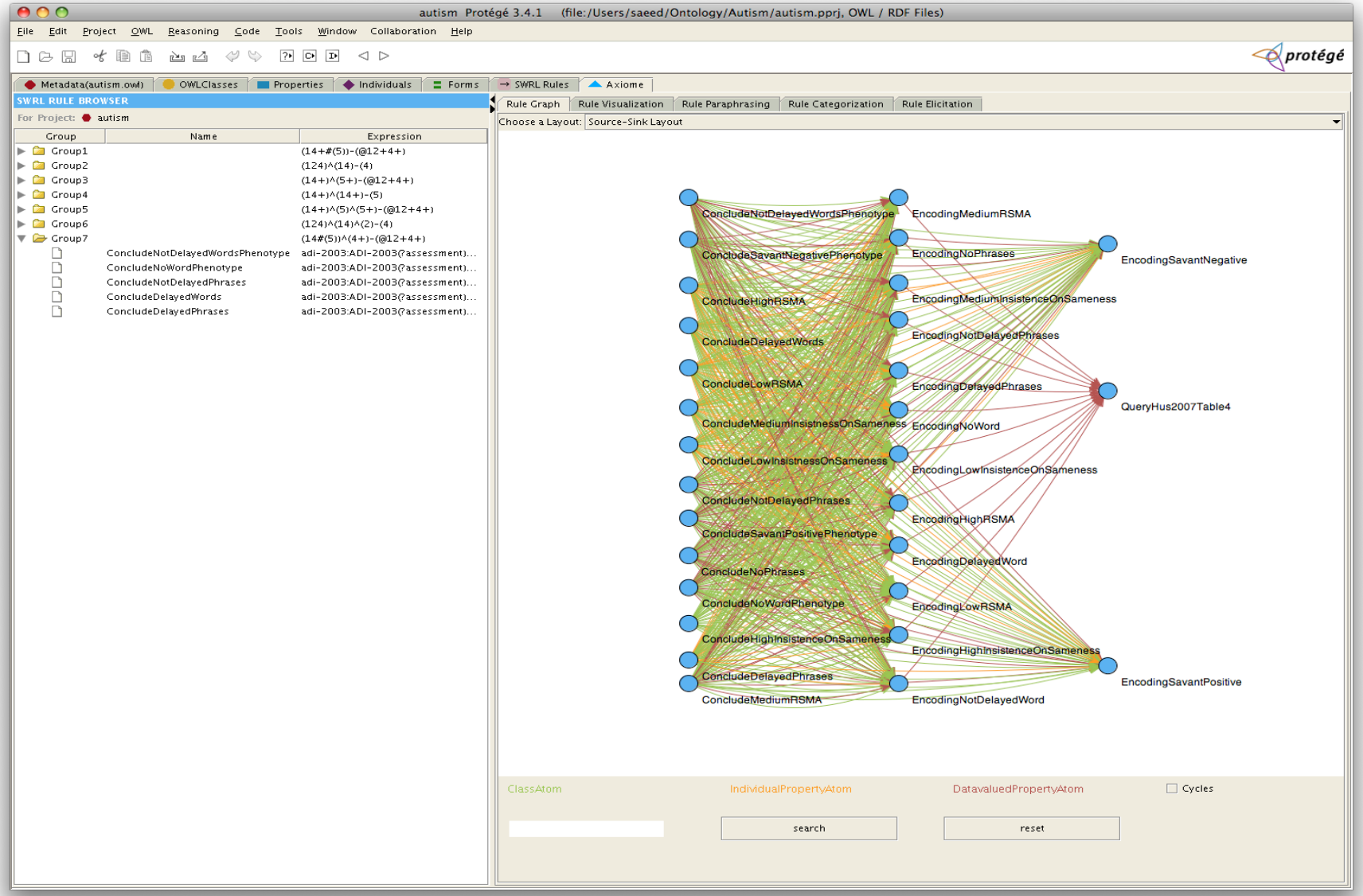


(12#(2#(5))4+#(5)#(5))^(14#(5))-(2)

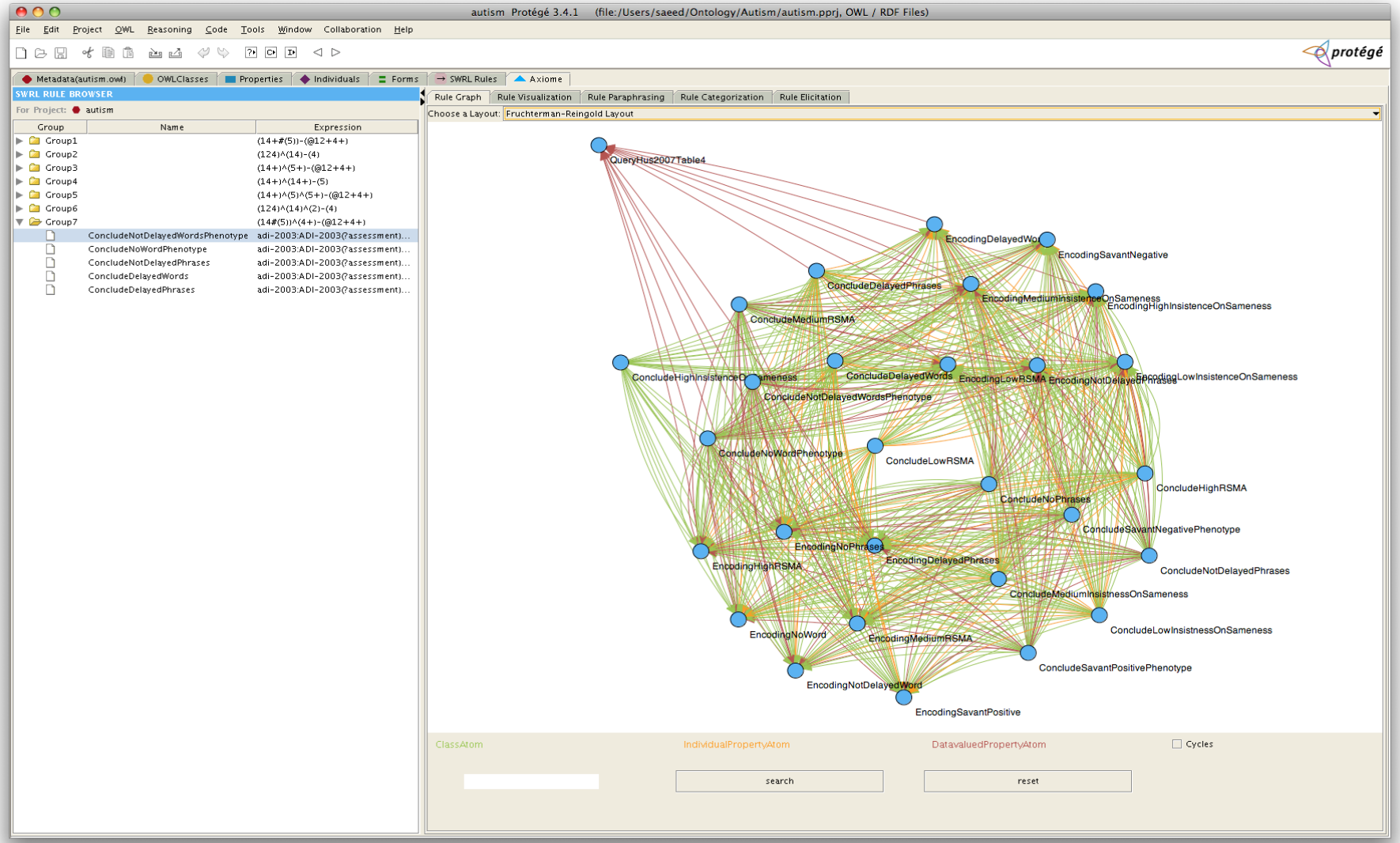
Rule Elicitation



Rule Base Browsing



Rule Base Browsing



Summary

- The increasing size and complexity of rule bases makes tools for rule management a necessity
- Our relatively simple method of rule syntactic analysis enables five powerful rule management techniques
- The Axiomé tool incorporates these techniques and is freely available as a rule management plug in for Protégé-OWL software

Future Work

- Combine semantic analysis with syntactic analysis:
  - Apply vector space modeling to represent each rule as a vector
  - Use ontology hierarchy to define measures of semantic similarity
  - Cluster rules based on k-means clustering

Thank You!

Questions?