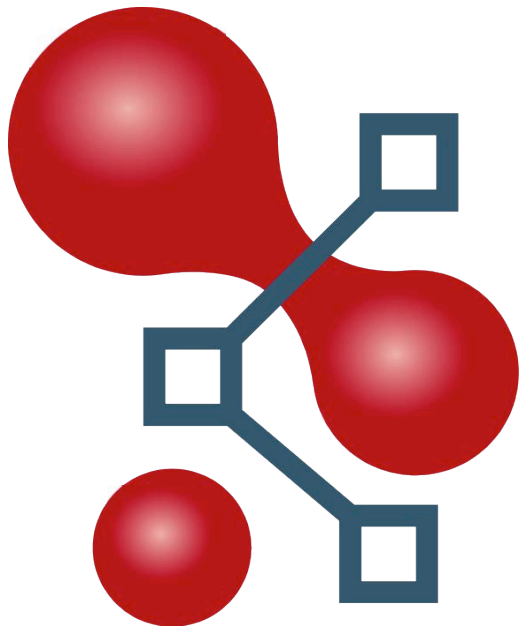


# A Framework for the Automatic Extraction of Rules from Online Text



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

Stanford Center for Biomedical Informatics Research  
Stanford, CA, U.S.A.

RuleML, Barcelona, Spain, July 21<sup>st</sup>, 2011

# Automatic Rule Extraction from Text

The screenshot shows the Hertz website's 'Requirements to Rent' page. The page title is 'Requirements to Rent' and the main heading is 'Requirements to Rent'. The content includes a question: 'What are the general requirements to rent a car with Hertz?' and a list of three requirements:

1. You must meet the renting location's minimum age requirement or qualify with the age differential.
2. You must have a valid driver's license. Licenses including restrictions requiring use of a portable breathalyzer are unacceptable for rental or driving a Hertz vehicle.
3. You must have a valid form of payment. For more information search Keyword "Credit Card".

Below the requirements, there is a section 'Was this answer helpful?' with 'Yes' and 'No' buttons. There is also a section 'Answers others found helpful' with a list of links: 'Minimum age to rent car', 'Reserving using Debit Cards', 'Rental payment by second party', 'Additional Driver Age', and 'suspended driver's license'. At the bottom of the page, there is a footer with various links under categories like 'About Hertz', 'Hertz Assistance', 'Business to Business', 'Travel Agents', and 'Featured Products'.



The screenshot shows the SWRL Rule editor interface. The title is 'SWRL Rule'. There are two tabs: 'Name' and 'Comment'. The 'Name' tab is selected, and the text in the field is 'http://www.owl-ontologies.com/Ontology1293225564.owl#Rule-3'. Below the name field, there is a section for the SWRL Rule itself. The rule text is:

```
Person(?a) ^ minimum_age_requirement(?b) ^ present(?a, ?b) ^ valid_payment(?c) ^ present(?a, ?c) ^ valid_driver_license(?d) ^ present(?a, ?d) ^ hertz(?e) -> qualifiedToRentFrom(?a, ?e)
```

At the bottom of the editor, there is a toolbar with various icons for editing the rule, including a question mark, a plus sign, a minus sign, a checkmark, and various logical symbols like '^', '→', '(', ')', '[', ']', and '←'.

# General Requirements in Extracting Rules from Text

- Ability to recognize domain concepts in text
- Recognize *relationships* between concepts
- Assemble these relationships into chains
- Requires understanding grammatical structure of sentence to detect relationships

# Method Overview

- Method requires:
  - User-selected text containing rule-like information
  - Existing OWL domain ontology
  - Existing SWRL rules
- Using these in combination with NLP techniques, automatically generate SWRL rule(s) from specified text

# What is SWRL?

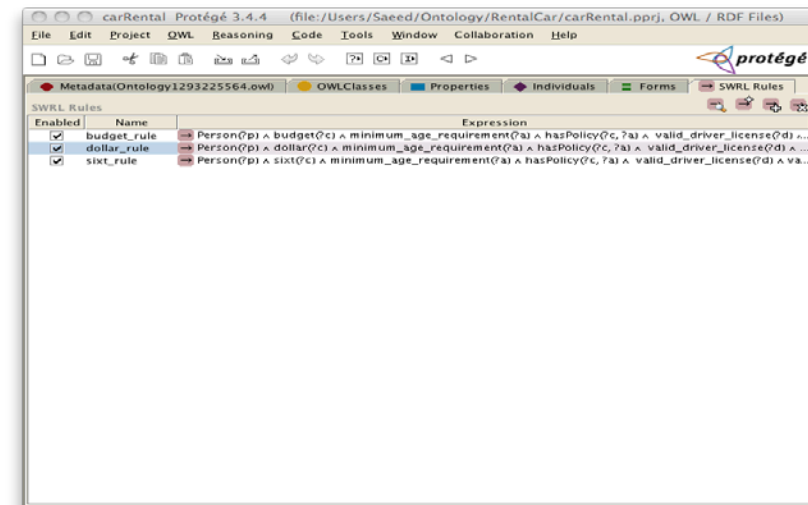
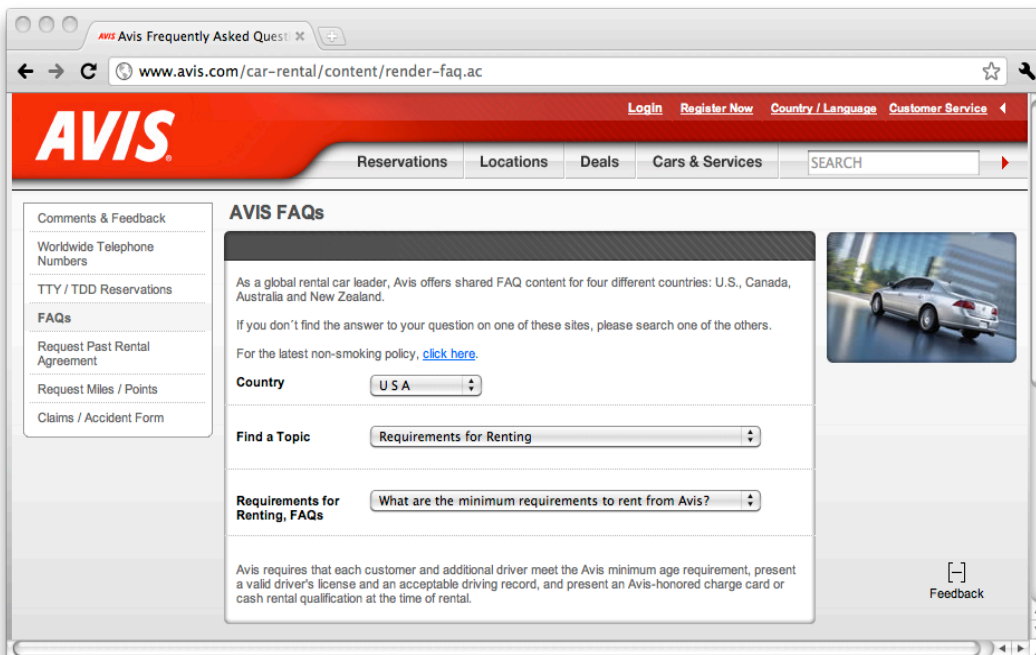
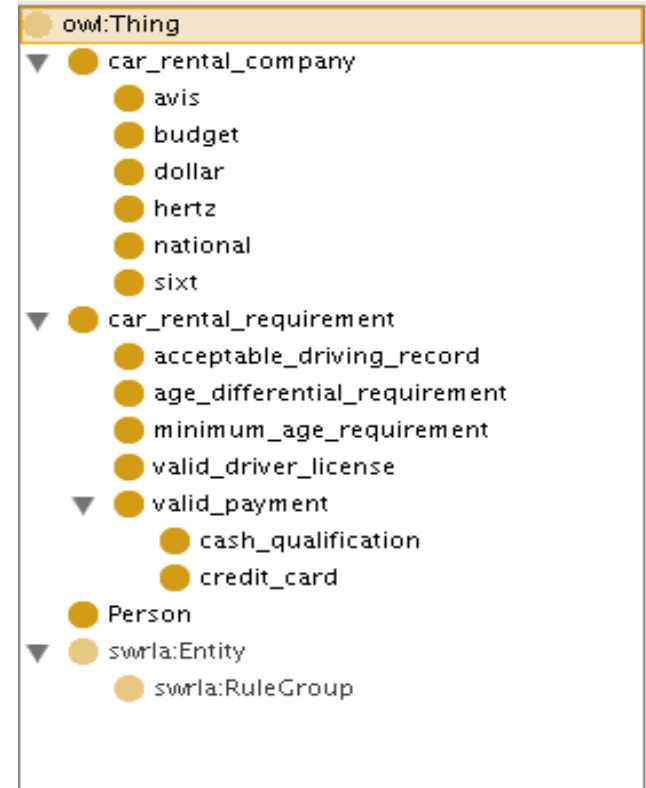
- SWRL is an acronym for Semantic Web Rule Language
- SWRL includes a high-level abstract syntax for Horn-like rules
- SWRL is an OWL-based rule language
- All rules are expressed in terms of OWL concepts (classes, properties, individuals)
  - e.g.,  $\text{Person}(?a) \wedge \text{minimum\_age\_requirement}(?b) \wedge \text{meet}(?a, ?b) \rightarrow \text{EligibleToRent}(?a)$

# Tight Interaction between SWRL and OWL

<b>SWRL Atom Type</b>	<b>Example Atom</b>
Class atom	Person(?x), Car(?y)
Object property atom	hasLicense(?x, ?y) hasLocation(?x, ?y)
SameAs/DifferentFrom atom	sameAs(?x, ?y) differentFrom(?x, ?y)
Data property atom	hasName(?x, "Joe") hasAge(?x, ?g)
Built-in atom	swrlb:notEqual(?state, "CA") swrlb:lessThan(?g, 18)
Data range atom	xsd:double(?x), [3-5](?x)

# Method Inputs

- User-selected text
- OWL domain ontology
- SWRL domain rules



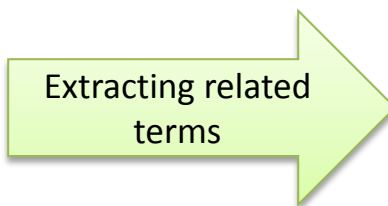
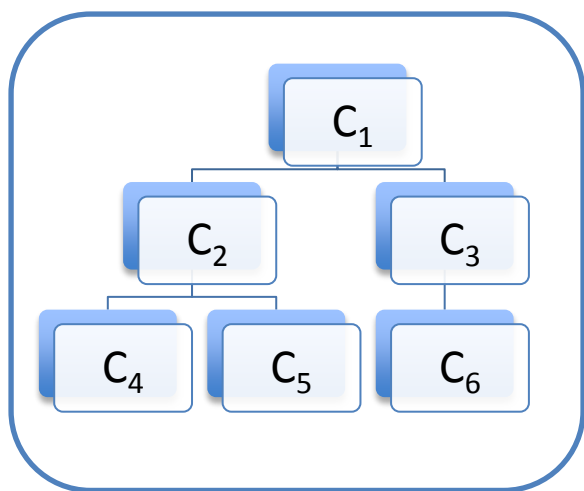
# Task Decomposition

- Expansion of domain ontology terms
- Grammatically analyze text
- Finding (property) relationships between entities in text
- Finding (class) concepts for those entities
- Assemble rule bodies
- Restricting relationships' domain and range
- Assembling rule head



# Step I: Expansion of Domain Ontology Terms

- Using WordNet, we expand each ontology term (classes and properties) to synonym terms and their morphological variations
  - e.g., **Present**: *present, show, demonstrate*



Concept	Related terms
C <sub>1</sub>	t <sub>1</sub> ,t <sub>2</sub> ,t <sub>3</sub>
C <sub>2</sub>	t <sub>4</sub> ,t <sub>5</sub>
C <sub>3</sub>	t <sub>6</sub> ,t <sub>7</sub> ,t <sub>8</sub>
C <sub>4</sub>	t <sub>9</sub> ,t <sub>10</sub>
C <sub>5</sub>	t <sub>11</sub> ,t <sub>12</sub>
C <sub>6</sub>	t <sub>13</sub> ,t <sub>14</sub> ,t <sub>15</sub>

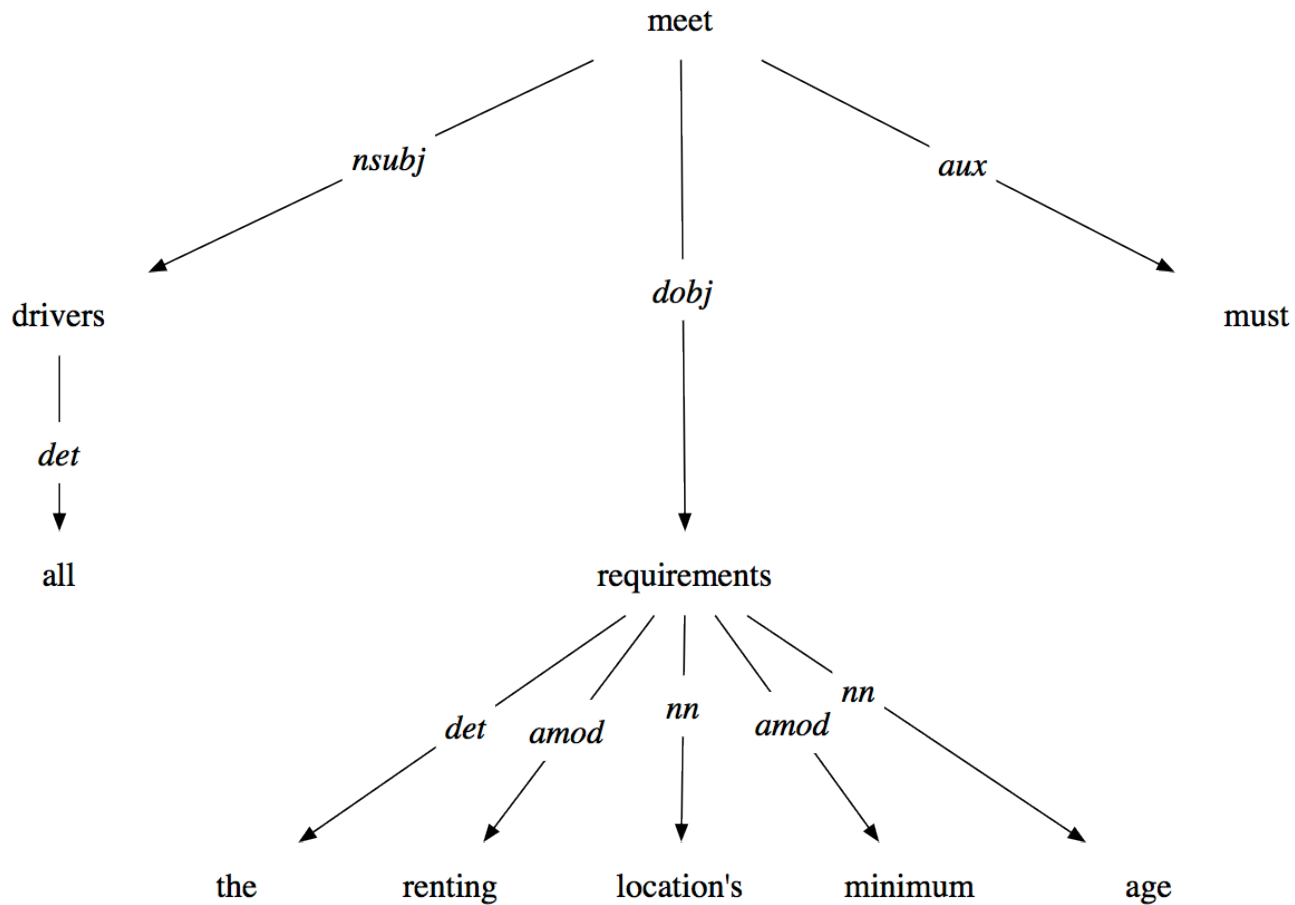
# Step II: Finding Grammatical Relationships in Text

- Using Stanford Parser, find all grammatical relationships in text
- Tree-based statistical parser
- Six main relevant types of relationships

<b>Abbreviation</b>	<b>Explanation</b>
Det	determiner
nsubject	nominal subject
Aux	auxiliary
Amod	adjectival modifier
nn	noun compound modifier
dobj	direct object

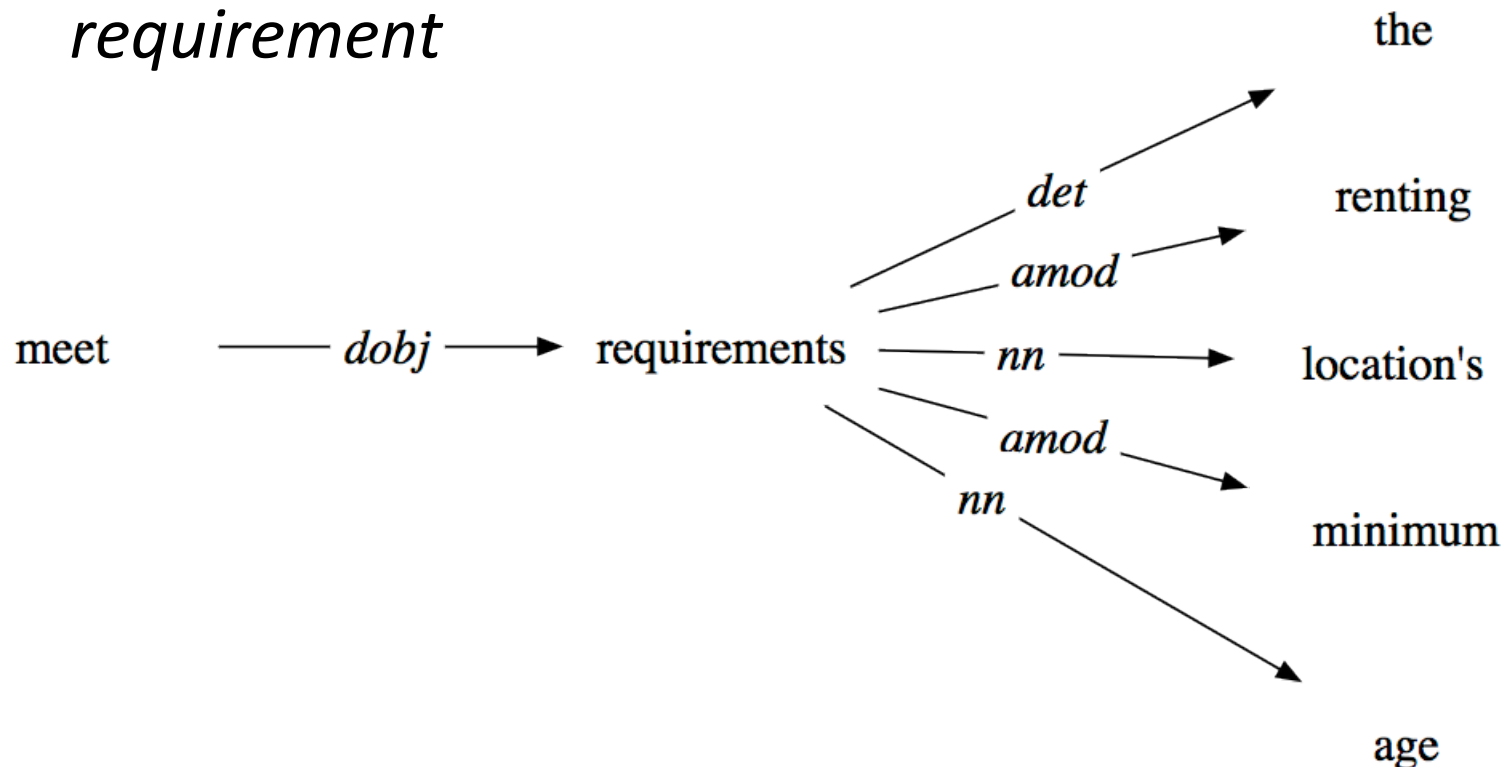
# Step II: Finding Grammatical Relationships in Text

e.g., “All drivers must meet the renting location's minimum age requirements.”



# Step III: Finding Property Concepts in the Text

- Expand dependency graph for each OWL property and reorder terms
  - e.g., meet: *the renting location's minimum age requirement*



# Step IV: Finding Class Concepts in Text

- After we find dependent phrases, we try to identify OWL classes in the domain ontology corresponding to each phrase
- Use Needleman-Wunsch global sequence alignment algorithm
- Uses dynamic-programming approach to find the best sequence alignment based on the scores of all possible alignments for two sequences
  - *e.g., “the renting location’s minimum age requirement”*: minimum\_age\_requirement

# Step V: Assembling Rule Bodies

- Assemble each rule body using the relationships that were identified in the last step
- First assemble chains of object property atoms, where each atom property contains an OWL object property corresponding to a relationship identified in the source text
- Add class atoms
- If no disjunctions are identified by the parser, we generate conjunctions of relevant object property atoms
- Since SWRL does not support disjunctions of atoms, rules containing disjunctions expand to multiple rules

# Step VI: Restricting relationships' domain and range

- Restrict relationships in generated rules to domain and range specified in domain ontology
- For example, in
  - e.g.,  $\text{minimum\_age\_requirement}(?b) \wedge \text{meet}(?a, ?b)$ 
    - ?a could refer to any matched domain concept
- Use ontology to restrict it:
  - $\text{Person}(?a) \wedge \text{minimum\_age\_requirement}(?b) \wedge \text{meet}(?a, ?b)$

# Step VII: Generating Rule Heads

- Use existing rule head for this type of rule
- Align variables in head with variables from corresponding type in body using ontology
- Limitation: currently only supports simple rule heads from existing rule set

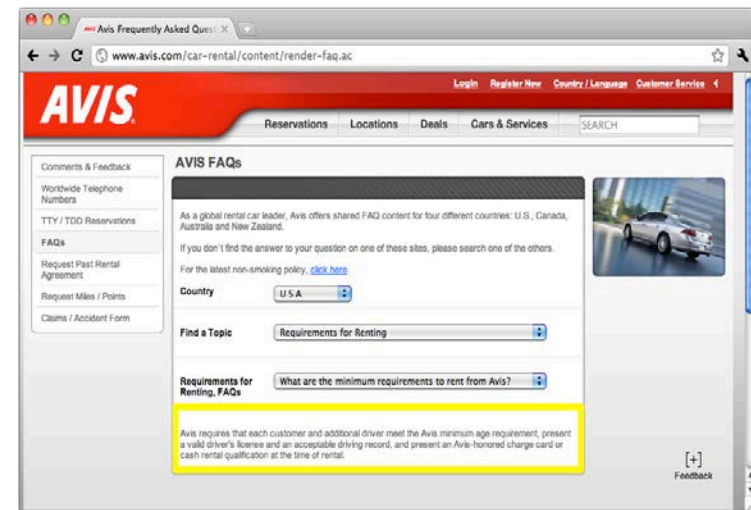


# Evaluation

- Used rental requirements for 17 California car rental companies
  - 6 training
  - 11 testing
- User-selected relevant text
- Applied method

No.	Company	Location(s)	Usage
1	<a href="#">Ace</a>	SAN	Testing
2	<a href="#">Advantage</a>	LAX, SAN, SJC	Training
3	<a href="#">Alamo</a>	LAX, SAN, SFO, SJC	Testing
4	<a href="#">Avis</a>	LAX, SAN, SFO, SJC	Testing
5	<a href="#">Budget</a>	LAX, SAN, SFO, SJC	Training
6	<a href="#">Dollar</a>	LAX, SAN, SFO, SJC	Training
7	<a href="#">Enterprise</a>	LAX, SAN, SFO, SJC	Testing
8	<a href="#">Fox</a>	LAX, SAN, SFO, SJC	Testing
9	<a href="#">Hertz</a>	LAX, SAN, SFO, SJC	Testing
10	<a href="#">Midway</a>	SAN	Testing
11	<a href="#">National</a>	LAX, SAN, SFO, SJC	Testing
12	<a href="#">Pacific</a>	SAN	Testing
13	<a href="#">Payless</a>	LAX, SAN	Testing
14	<a href="#">Renty</a>	SAN	Testing
15	<a href="#">Thrifty</a>	LAX, SAN, SFO, SJC	Training
16	<a href="#">TravCar</a>	SAN	Training
17	<a href="#">West Coast</a>	SAN	Training

# Example Result: Avis



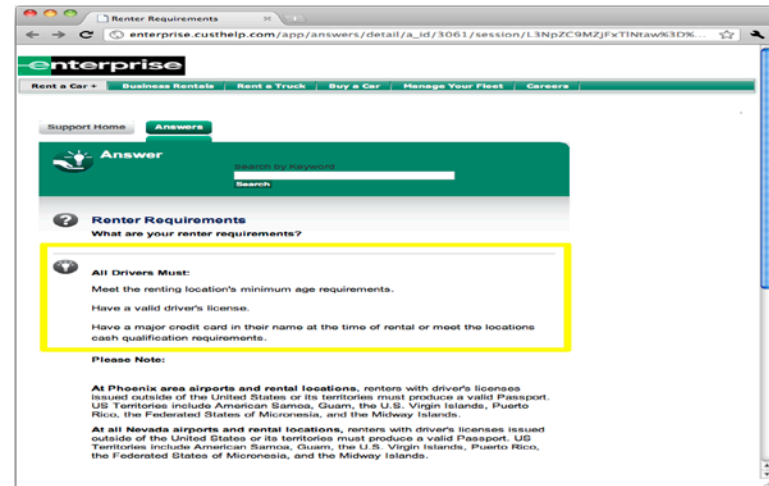
- Input text:

"Avis requires that each customer and additional driver meet the Avis minimum age requirement, present a valid driver's license and an acceptable driving record, and present an Avis-honored charge card or cash rental qualification at the time of rental."

- Output rules:

- $\text{Person}(?a) \wedge \text{credit\_card}(?b) \wedge \text{present}(?a, ?b) \wedge \text{minimum\_age\_requirement}(?c) \wedge \text{present}(?a, ?c) \wedge \text{valid\_driver\_license}(?d) \wedge \text{present}(?a, ?d) \wedge \text{acceptable\_driving\_record}(?e) \wedge \text{present}(?a, ?e) \wedge \text{avis}(?f) \rightarrow \text{qualifiedToRentFrom}(?a, ?f)$
- $\text{Person}(?a) \wedge \text{cash\_qualification}(?b) \wedge \text{present}(?a, ?b) \wedge \text{minimum\_age\_requirement}(?c) \wedge \text{present}(?a, ?c) \wedge \text{valid\_driver\_license}(?d) \wedge \text{present}(?a, ?d) \wedge \text{acceptable\_driving\_record}(?e) \wedge \text{present}(?a, ?e) \wedge \text{avis}(?f) \rightarrow \text{qualifiedToRentFrom}(?a, ?f)$

# Example Result: Enterprise



- Input text:

*“All Drivers Must: Meet the renting location's minimum age requirements. Have a valid driver's license. Have a major credit card in their name at the time of rental or meet the location's cash qualification requirements.”*

- Output rules:

- $\text{Person}(?a) \wedge \text{credit\_card}(?b) \wedge \text{has}(?a, ?b) \wedge \text{valid\_driver\_license}(?c) \wedge \text{has}(?a, ?c) \wedge \text{minimum\_age\_requirement}(?d) \wedge \text{meet}(?a, ?d) \wedge \text{enterprise}(?e) \rightarrow \text{qualifiedToRentFrom}(?a, ?e)$
- $\text{Person}(?a) \wedge \text{cash\_qualification}(?b) \wedge \text{meet}(?a, ?b) \wedge \text{valid\_driver\_license}(?c) \wedge \text{has}(?a, ?c) \wedge \text{minimum\_age\_requirement}(?d) \wedge \text{meet}(?a, ?d) \wedge \text{enterprise}(?e) \rightarrow \text{qualifiedToRentFrom}(?a, ?e)$

# Results

- For 9 out of 11 cases we generated one or more rules that accurately reflected the rental company's requirements.
- Precision: 100%
- Recall: 96%
  - The 4% missing was related to missing concepts in the ontology, e.g., home address, phone number

# Conclusion

- **Structured knowledge bases** in combination with **NLP** techniques can be used to automatically **formalize** knowledge in **free text**
- Can dramatically expand the amount of knowledge that can be automatically extracted

# Future Work

- Improve the precision and recall of the information retrieval and text mining methods
- Evaluation the rule extraction method to the biomedical domain
- Assemble (semi-) automated pipeline
  - Auto-generate domain ontology
  - Auto-select text fragments with rules
  - Generate rules (with more elaborate heads)