Chapter 5
Non monotonic rules

Based on slides from Grigoris Antoniou and Frank van Harmelen
In nonmonotonic rule systems, a rule may not be applied even if all premises are known because we have to consider contrary reasoning chains.

Now we consider defeasible rules that can be defeated by other rules.

Negated atoms may occur in the head and the body of rules, to allow for conflicts:
- \( p(X) \rightarrow q(X) \)
- \( r(X) \rightarrow \neg q(X) \)
Defeasible Rules

\[ p(X) \Rightarrow q(X) \]
\[ r(X) \Rightarrow \neg q(X) \]

- Given also the facts \( p(a) \) and \( r(a) \) we conclude neither \( q(a) \) nor \( \neg q(a) \)
  - This is a typical example of 2 rules blocking each other

- Conflict may be resolved using priorities among rules

- Suppose we knew somehow that the 1st rule is stronger than the 2nd
  - Then we could derive \( q(a) \)
Origin of Rule Priorities

- Higher authority
  - E.g. in law, federal law preempts state law
  - E.g., in business administration, higher management has more authority than middle management

- Recency

- Specificity
  - A typical example is a general rule with some exceptions

- We abstract from the specific prioritization principle
  - We assume the existence of an external priority relation on the set of rules
Rule Priorities

\[ r_1: p(X) \Rightarrow q(X) \]
\[ r_2: r(X) \Rightarrow \neg q(X) \]
\[ r_1 > r_2 \]

- Rules have a unique label
- The priority relation to be acyclic
In simple cases two rules are competing only if one head is the negation of the other.

But in many cases once a predicate $p$ is derived, some other predicates are excluded from holding:

- E.g., an investment consultant may base his recommendations on three levels of risk investors are willing to take: low, moderate, high.
- Only one risk level per investor is allowed to hold.
These situations are modelled by maintaining a conflict set $C(L)$ for each literal $L$.

$C(L)$ always contains the negation of $L$ but may contain more literals.
Defeasible Rules: Syntax

\[ r : L_1, ..., L_n \Rightarrow L \]

- \( r \) is the label
- \( \{L_1, ..., L_n\} \) the body (or premises)
- \( L \) the head of the rule
- \( L, L_1, ..., L_n \) are positive or negative literals
- A literal is an atomic formula \( p(t_1, ..., t_m) \) or its negation \( \neg p(t_1, ..., t_m) \)
- No function symbols may occur in the rule
A defeasible logic program is a triple \((F, R, >)\) consisting of

- a set \(F\) of facts
- a finite set \(R\) of defeasible rules
- an acyclic binary relation \(>\) on \(R\)

A set of pairs \(r > r'\) where \(r\) and \(r'\) are labels of rules in \(R\)
Lecture Outline

1. Introduction
2. Monotonic Rules: Example
3. Monotonic Rules: Syntax & Semantics
4. DLP: Description Logic Programs
5. SWRL: Semantic Web Rules Language
6. Nonmonotonic Rules: Syntax
7. Nonmonotonic Rules: Example
8. RuleML: XML-Based Syntax
Brokered trades take place via an independent third party, the broker.

The broker matches the buyer’s requirements and the sellers’ capabilities, and proposes a transaction when both parties can be satisfied by the trade.

The application is apartment renting an activity that is common and often tedious and time-consuming.
The Potential Buyer’s Requirements

- At least 45 sq m with at least 2 bedrooms
- Elevator if on 3rd floor or higher
- Pet animals must be allowed

• Carlos is willing to pay:
  - $300 for a centrally located 45 sq m apartment
  - $250 for a similar flat in the suburbs
  - An extra $5 per square meter for a larger apartment
  - An extra $2 per square meter for a garden
  - He is unable to pay more than $400 in total

• If given the choice, he would go for the cheapest option
• His second priority is the presence of a garden
• His lowest priority is additional space
Carlos’s Requirements – Predicates Used

- \( \text{size}(x,y) \), \( y \) is the size of apartment \( x \) (in sq m)
- \( \text{bedrooms}(x,y) \), \( x \) has \( y \) bedrooms
- \( \text{price}(x,y) \), \( y \) is the price for \( x \)
- \( \text{floor}(x,y) \), \( x \) is on the \( y \)-th floor
- \( \text{gardenSize}(x,y) \), \( x \) has a garden of size \( y \)
- \( \text{lift}(x) \), there is an elevator in the house of \( x \)
- \( \text{pets}(x) \), pets are allowed in \( x \)
- \( \text{central}(x) \), \( x \) is centrally located
- \( \text{acceptable}(x) \), flat \( x \) satisfies Carlos’s requirements
- \( \text{offer}(x,y) \), Carlos is willing to pay $ \( y \) for flat \( x \)
Carlos’s Requirements – Rules

r1: \( \Rightarrow \) acceptable(X)

r2: bedrooms(X,Y), \( Y < 2 \) \( \Rightarrow \) \( \neg \)acceptable(X)

r3: size(X,Y), \( Y < 45 \) \( \Rightarrow \) \( \neg \)acceptable(X)

r4: \( \neg \)pets(X) \( \Rightarrow \) \( \neg \)acceptable(X)

r5: floor(X,Y), \( Y > 2, \neg \)lift(X) \( \Rightarrow \) \( \neg \)acceptable(X)

r6: price(X,Y), \( Y > 400 \) \( \Rightarrow \) \( \neg \)acceptable(X)

r2 > r1, r3 > r1, r4 > r1, r5 > r1, r6 > r1
Carlos’s Requirements – Rules (2)

r7: size(X,Y), Y ≥ 45, garden(X,Z), central(X) ⇒
   offer(X, 300 + 2*Z + 5*(Y − 45))

r8: size(X,Y), Y ≥ 45, garden(X,Z), ¬central(X) ⇒
   offer(X, 250 + 2*Z + 5(Y − 45))

r9: offer(X,Y), price(X,Z), Y < Z ⇒ ¬acceptable(X)

r9 > r1
Representation of Available Apartments

bedrooms(a1,1)
size(a1,50)
central(a1)
floor(a1,1)
¬lift(a1)
pets(a1)
garden(a1,0)
price(a1,300)
Available Apartments (2)

<table>
<thead>
<tr>
<th>Flat</th>
<th>Bedrooms</th>
<th>Size</th>
<th>Central</th>
<th>Floor</th>
<th>Lift</th>
<th>Pets</th>
<th>Garden</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1</td>
<td>1</td>
<td>50</td>
<td>yes</td>
<td>1</td>
<td>no</td>
<td>yes</td>
<td>0</td>
<td>300</td>
</tr>
<tr>
<td>a2</td>
<td>2</td>
<td>45</td>
<td>yes</td>
<td>0</td>
<td>no</td>
<td>yes</td>
<td>0</td>
<td>335</td>
</tr>
<tr>
<td>a3</td>
<td>2</td>
<td>65</td>
<td>no</td>
<td>2</td>
<td>no</td>
<td>yes</td>
<td>0</td>
<td>350</td>
</tr>
<tr>
<td>a4</td>
<td>2</td>
<td>55</td>
<td>no</td>
<td>1</td>
<td>yes</td>
<td>no</td>
<td>15</td>
<td>330</td>
</tr>
<tr>
<td>a5</td>
<td>3</td>
<td>55</td>
<td>yes</td>
<td>0</td>
<td>no</td>
<td>yes</td>
<td>15</td>
<td>350</td>
</tr>
<tr>
<td>a6</td>
<td>2</td>
<td>60</td>
<td>yes</td>
<td>3</td>
<td>no</td>
<td>no</td>
<td>0</td>
<td>370</td>
</tr>
<tr>
<td>a7</td>
<td>3</td>
<td>65</td>
<td>yes</td>
<td>1</td>
<td>no</td>
<td>yes</td>
<td>12</td>
<td>375</td>
</tr>
</tbody>
</table>
Determining Acceptable Apartments

- If we match Carlos’s requirements and the available apartments, we see that
- flat a1 is not acceptable because it has one bedroom only (rule r2)
- flats a4 and a6 are unacceptable because pets are not allowed (rule r4)
- for a2, Carlos is willing to pay $300, but the price is higher (rules r7 and r9)
- flats a3, a5, and a7 are acceptable (rule r1)
Selecting an Apartment

\[ r_{10}: \text{cheapest}(X) \implies \text{rent}(X) \]
\[ r_{11}: \text{cheapest}(X), \text{largestGarden}(X) \implies \text{rent}(X) \]
\[ r_{12}: \text{cheapest}(X), \text{largestGarden}(X), \text{largest}(X) \implies \text{rent}(X) \]

\[ r_{12} > r_{10}, r_{12} > r_{11}, r_{11} > r_{10} \]

- We must specify that at most one apartment can be rented, using conflict sets:
  - \( C(\text{rent}(x)) = \{\neg\text{rent}(x)\} \cup \{\text{rent}(y) \mid y \neq x\} \)
Lecture Outline

1. Introduction
2. Monotonic Rules: Example
3. Monotonic Rules: Syntax & Semantics
4. DLP: Description Logic Programs
5. SWRL: Semantic Web Rules Language
6. Nonmonotonic Rules: Syntax
7. Nonmonotonic Rules: Example
8. RuleML: XML-Based Syntax
In accordance with the Semantic Web vision:
  - Make rules machine-accessible.

RuleML is an important standardization effort for rule markup on the Web.

Actually a family of rule markup languages, corresponding to different kinds of rule languages:
  - derivation rules, integrity constraints, reaction rules

Kernel: Datalog (function-free Horn logic)
RuleML (2)

- XML based
  - in the form of XML schemas
  - DTDs for earlier versions
- Straightforward correspondence between RuleML elements and rule components
<table>
<thead>
<tr>
<th>Component</th>
<th>RuleML</th>
</tr>
</thead>
<tbody>
<tr>
<td>program</td>
<td>rulebase</td>
</tr>
<tr>
<td>rule</td>
<td>Implies</td>
</tr>
<tr>
<td>head</td>
<td>head</td>
</tr>
<tr>
<td>body</td>
<td>body</td>
</tr>
<tr>
<td>&amp; of atoms</td>
<td>And</td>
</tr>
<tr>
<td>predicate</td>
<td>Rel</td>
</tr>
<tr>
<td>constant</td>
<td>Ind</td>
</tr>
<tr>
<td>var</td>
<td>Var</td>
</tr>
</tbody>
</table>
An Example

- The discount for a customer buying a product is 7.5 percent if the customer is premium and the product is luxury.
<Implies>
  <head>
    <Atom>
      <Rel>discount</Rel>
      <Var>customer</Var>
      <Var>product</Var>
      <Ind>7.5</Ind>
    </Atom>
  </head>
</Implies>
RuleML Representation (2)

<body>
  <And>
    <Atom>
      <Rel>premium</Rel>
      <Var>customer</Var>
    </Atom>
    <Atom>
      <Rel>luxury</Rel>
      <Var>product</Var>
    </Atom>
  </And>
</body>
</Implies>
Summary

• Horn logic is a subset of predicate logic that allows efficient reasoning, orthogonal to description logics
• Horn logic is the basis of monotonic rules
• DLP and SWRL are two important ways of combining OWL with Horn rules.
  - DLP is essentially the intersection of OWL and Horn logic
  - SWRL is a much richer language
Nonmonotonic rules are useful in situations where the available information is incomplete.

They are rules that may be overridden by contrary evidence.

Priorities are used to resolve some conflicts between rules.

Representation XML-like languages is straightforward.