Logic Programming
Queries

Michael Genesereth
Computer Science Department
Stanford University
True or False questions:
  e.g. Is Art the parent of Bob?

Fill-in-the-blanks questions:
  e.g. Art is the parent of ____?
  e.g. ____ is the parent of Bob?
  e.g. ____ is the parent of ____?

Compound questions:
  e.g. Is Art the parent of Bob or the parent of Bud?
  e.g. ____ has sons and no daughters?
Syntax
A **constant** is a string of lower case letters, digits, underscores, and periods *or* a string of ascii characters within double quotes.

\[
\begin{align*}
\text{a, b, c, 1, 2, 3, joe, cs151} \\
\text{f, g, pair, triple} \\
\text{p, q, r, person, parent, prefers}
\end{align*}
\]

A **variable** is either a lone underscore or a string of letters, digits, underscores, and periods beginning with an upper case letter.

\[
\begin{align*}
\text{X} \\ 
\text{Y23} \\ 
\text{Somebody} \\ 
\text{_}
\end{align*}
\]
Terms

Symbols
  art
  bob

Variables
  X
  Y23

Compound Terms
  pair(art, bob)
  pair(\(X, Y23\))
  pair(pair(art, bob), pair(\(X, Y23\)))

Query terms are not necessarily ground!
Atoms

- $p(a,b)$
- $p(a,x)$
- $p(Y,c)$

Atoms are like factoids in datasets except that they may contain variables.

Negations

- $\neg p(a,b)$

Literals (atoms or negations of atoms)

- $p(a,Y)$
- $\neg p(a,Y)$

An atom is a positive literal.
A negations is a negative literal.
Intuitive meaning: \textit{goal}(a,b) is true if \textit{p}(a,b) is true and \textit{q}(b) is false.
Queries May Contain Variables

goal(a,b) :- p(a,b) & ~q(b)

goal(X,Y) :- p(X,Y) & ~q(Y)
goal(X,X) :- p(X,Y) & ~q(Y)

goal(X,b) :- p(X,b) & ~q(b)
goal(X,b) :- p(X,Y) & ~q(Y)
goal(X,f(Y)) :- p(X,Y) & ~q(Y)
goal(X,Y) :- p(X,f(Y)) & ~q(Y)
Semantics
\[ p(a,b) \]
\[ p(b,c) \]
\[ p(c,d) \]
\[ p(d,c) \]

\[ + \]

\[ \text{goal}(X,Y) :- p(X,Y) \land p(Y,X) \]

\[ = \]

\[ \text{goal}(c,d) \]
\[ \text{goal}(d,c) \]
An instance of a query is a query in which all variables have been consistently replaced by ground terms.

Rule

\[
\text{goal}(X,Y) := \ p(X,Y) \ & \ \sim q(Y)
\]

Herbrand Universe

\{a, b\}

Instances

\[
\text{goal}(a,a) := \ p(a,a) \ & \ \sim q(a)
\]
\[
\text{goal}(a,b) := \ p(a,b) \ & \ \sim q(b)
\]
\[
\text{goal}(b,a) := \ p(b,a) \ & \ \sim q(a)
\]
\[
\text{goal}(b,b) := \ p(b,b) \ & \ \sim q(b)
\]
The result of applying a query to a dataset is defined to be the set of all $\psi$ such that

1. $\psi$ is the head of an instance of the rule,
2. every positive subgoal of the instance is in the dataset,
3. no negative subgoal of the instance is in the dataset.
Example

Dataset
- \( p(a,b) \)
- \( p(b,c) \)
- \( p(c,d) \)
- \( p(d,c) \)

Instances
- \( \text{goal}(a,a) :- p(a,a) \land p(a,a) \)
- \( \text{goal}(a,b) :- p(a,b) \land p(b,a) \)
- \( \text{goal}(a,c) :- p(a,c) \land p(c,a) \)
- \( \text{goal}(a,d) :- p(a,d) \land p(d,a) \)
- \( \text{goal}(b,a) :- p(b,a) \land p(a,b) \)
- \( \text{goal}(b,b) :- p(b,b) \land p(b,b) \)
- \( \text{goal}(b,c) :- p(b,c) \land p(c,b) \)
- \( \text{goal}(b,d) :- p(b,d) \land p(d,b) \)
- \( \text{goal}(c,a) :- p(c,a) \land p(a,c) \)
- \( \text{goal}(c,b) :- p(c,b) \land p(b,c) \)
- \( \text{goal}(c,c) :- p(c,c) \land p(c,c) \)
- \( \text{goal}(c,d) :- p(c,d) \land p(d,c) \)
- \( \text{goal}(d,a) :- p(d,a) \land p(a,d) \)
- \( \text{goal}(d,b) :- p(d,b) \land p(b,d) \)
- \( \text{goal}(d,c) :- p(d,c) \land p(c,d) \)
- \( \text{goal}(d,d) :- p(d,d) \land p(d,d) \)

Query
- \( \text{goal}(X,Y) :- \)
- \( \quad p(X,Y) \land p(Y,X) \)

Result
- \( \text{goal}(c,d) \)
- \( \text{goal}(d,c) \)
Example

Dataset
- p(a, b)
- p(b, c)
- p(c, d)
- p(d, c)

Instances
- goal(a, a) :- p(a, a) & ~p(a, a)
- goal(a, b) :- p(a, b) & ~p(b, a)
- goal(a, c) :- p(a, c) & ~p(c, a)
- goal(a, d) :- p(a, d) & ~p(d, a)
- goal(b, a) :- p(b, a) & ~p(a, b)
- goal(b, b) :- p(b, b) & ~p(b, b)
- goal(b, c) :- p(b, c) & ~p(c, b)
- goal(b, d) :- p(b, d) & ~p(d, b)
- goal(c, a) :- p(c, a) & ~p(a, c)
- goal(c, b) :- p(c, b) & ~p(b, c)
- goal(c, c) :- p(c, c) & ~p(c, c)
- goal(c, d) :- p(c, d) & ~p(d, c)
- goal(d, a) :- p(d, a) & ~p(a, d)
- goal(d, b) :- p(d, b) & ~p(b, d)
- goal(d, c) :- p(d, c) & ~p(c, d)
- goal(d, d) :- p(d, d) & ~p(d, d)

Query
- goal(X, Y) :-
  p(X, Y) & ~p(Y, X)

Result
- goal(a, b)
- goal(b, c)
Quiz

Dataset
- p(a,b)
- p(b,c)
- p(c,d)
- p(d,c)

Instances
- goal(a) :- p(a,a) & p(a,a)
- goal(a) :- p(a,b) & p(b,a)
- goal(a) :- p(a,c) & p(c,a)
- goal(a) :- p(a,d) & p(d,a)
- goal(b) :- p(b,a) & p(a,b)
- goal(b) :- p(b,b) & p(b,b)
- goal(b) :- p(b,c) & p(c,b)
- goal(b) :- p(b,d) & p(d,b)
- goal(c) :- p(c,a) & p(a,c)
- goal(c) :- p(c,b) & p(b,c)
- goal(c) :- p(c,c) & p(c,c)
- goal(c) :- p(c,d) & p(d,c)
- goal(d) :- p(d,a) & p(a,d)
- goal(d) :- p(d,b) & p(b,d)
- goal(d) :- p(d,c) & p(c,d)
- goal(d) :- p(d,d) & p(d,d)

Query

goal(X) :-
- p(X,Y) & p(Y,X)

Result
- goal(c)
- goal(d)
Dataset
- \( p(a,b) \)
- \( p(b,c) \)
- \( p(c,d) \)
- \( p(d,c) \)

Query
- \( \text{goal}(X,X) :\text{-} \)
  - \( p(X,Y) \) & \( p(Y,X) \)

Result
- \( \text{goal}(c,c) \)
- \( \text{goal}(d,d) \)

Instances
- \( \text{goal}(a,a) :\text{-} p(a,a) \) & \( p(a,a) \)
- \( \text{goal}(a,a) :\text{-} p(a,b) \) & \( p(b,a) \)
- \( \text{goal}(a,a) :\text{-} p(a,c) \) & \( p(c,a) \)
- \( \text{goal}(a,a) :\text{-} p(a,d) \) & \( p(d,a) \)
- \( \text{goal}(b,b) :\text{-} p(b,a) \) & \( p(a,b) \)
- \( \text{goal}(b,b) :\text{-} p(b,b) \) & \( p(b,b) \)
- \( \text{goal}(b,b) :\text{-} p(b,c) \) & \( p(c,b) \)
- \( \text{goal}(b,b) :\text{-} p(b,d) \) & \( p(d,b) \)
- \( \text{goal}(c,c) :\text{-} p(c,a) \) & \( p(a,c) \)
- \( \text{goal}(c,c) :\text{-} p(c,b) \) & \( p(b,c) \)
- \( \text{goal}(c,c) :\text{-} p(c,c) \) & \( p(c,c) \)
- \( \text{goal}(c,c) :\text{-} p(c,d) \) & \( p(d,c) \)
- \( \text{goal}(d,d) :\text{-} p(d,a) \) & \( p(a,d) \)
- \( \text{goal}(d,d) :\text{-} p(d,b) \) & \( p(b,d) \)
- \( \text{goal}(d,d) :\text{-} p(d,c) \) & \( p(c,d) \)
- \( \text{goal}(d,d) :\text{-} p(d,d) \) & \( p(d,d) \)
Dataset
- p(a,b)
- p(b,c)
- p(c,d)
- p(d,c)

Query
- goal(X,b) :- p(X,Y) & p(Y,X)

Result
- goal(c,b)
- goal(d,b)

Instances
- goal(a,b) :- p(a,a) & p(a,a)
- goal(a,b) :- p(a,b) & p(b,a)
- goal(a,b) :- p(a,c) & p(c,a)
- goal(a,b) :- p(a,d) & p(d,a)
- goal(b,b) :- p(b,a) & p(a,b)
- goal(b,b) :- p(b,b) & p(b,b)
- goal(b,b) :- p(b,c) & p(c,b)
- goal(b,b) :- p(b,d) & p(d,b)
- goal(c,b) :- p(c,a) & p(a,c)
- goal(c,b) :- p(c,b) & p(b,c)
- goal(c,b) :- p(c,c) & p(c,c)
- goal(c,b) :- p(c,d) & p(d,c)
- goal(d,b) :- p(d,a) & p(a,d)
- goal(d,b) :- p(d,b) & p(b,d)
- goal(d,b) :- p(d,c) & p(c,d)
- goal(d,b) :- p(d,d) & p(d,d)
Quiz

Dataset
- p(a,b)
- p(b,c)
- p(c,d)
- p(d,c)

Query
- \text{goal}(X,f(X)) \leftarrow p(X,Y) \land p(Y,X)

Result
- \text{goal}(c,f(c))
- \text{goal}(d,f(d))

Instances
- \text{goal}(a,f(a)) \leftarrow p(a,a) \land p(a,a)
- \text{goal}(a,f(a)) \leftarrow p(a,b) \land p(b,a)
- \text{goal}(a,f(a)) \leftarrow p(a,c) \land p(c,a)
- \text{goal}(a,f(a)) \leftarrow p(a,d) \land p(d,a)
- \text{goal}(b,f(b)) \leftarrow p(b,a) \land p(a,b)
- \text{goal}(b,f(b)) \leftarrow p(b,b) \land p(b,b)
- \text{goal}(b,f(b)) \leftarrow p(b,c) \land p(c,b)
- \text{goal}(b,f(b)) \leftarrow p(b,d) \land p(d,b)
- \text{goal}(c,f(c)) \leftarrow p(c,a) \land p(a,c)
- \text{goal}(c,f(c)) \leftarrow p(c,b) \land p(b,c)
- \text{goal}(c,f(c)) \leftarrow p(c,c) \land p(c,c)
- \text{goal}(c,f(c)) \leftarrow p(c,d) \land p(d,c)
- \text{goal}(d,f(d)) \leftarrow p(d,a) \land p(a,d)
- \text{goal}(d,f(d)) \leftarrow p(d,b) \land p(b,d)
- \text{goal}(d,f(d)) \leftarrow p(d,c) \land p(c,d)
- \text{goal}(d,f(d)) \leftarrow p(d,d) \land p(d,d)
Non-Examples

Dataset

\[ p(a, b) \]
\[ p(b, c) \]
\[ p(c, d) \]
\[ p(d, c) \]

Query

\[ \text{goal}(X, Y) : - \]
\[ p(X, Y) \land p(Y, X) \]

Not Results

\[ \text{goal}(c, d) \quad \text{goal}(a, b) \]
\[ \text{goal}(b, c) \quad \text{goal}(c, d) \]
\[ \text{goal}(c, d) \quad \text{goal}(d, c) \]

Too few. \quad \text{Too many.}
Query Sets

The result of applying a *set of queries* to a dataset is the union of the results of applying the queries to the dataset.

**Dataset**

\[
\{ p(a,b), p(b,c) \}
\]

**Queries**

\[
\begin{align*}
goal(X) & :\ p(X,Y) & \rightarrow & \{ goal(a), goal(b) \} \\
goal(Y) & :\ p(X,Y) & \rightarrow & \{ goal(b), goal(c) \}
\end{align*}
\]

**Result**

\[
\{ goal(a), goal(b), goal(c) \}
\]

*NB: A query set is effectively a disjunction.*
Safety
A rule is **safe** if and only if every variable in the head appears in some positive subgoal in the body and every variable in a negative subgoal appears in a prior positive subgoal.

Safe Rule:

$$\text{goal}(X, Z) :- \ p(X, Y) \ & \ q(Y, Z) \ & \ \neg r(X, Y)$$

Unsafe Rule:

$$\text{goal}(X, Z) :- \ p(X, Y) \ & \ q(Y, X)$$

Unsafe Rule:

$$\text{goal}(X, Y) :- \ p(X, Y) \ & \ \neg q(Y, Z)$$
Rule

\[ \text{goal}(X,Z) :- \ p(X,Y) \]

Herbrand Universe \( \{a, b\} \)

Dataset \( \{p(a,a)\} \)

Instances

\begin{align*}
\text{goal}(a,a) & : - \ p(a,a) \\
\text{goal}(a,a) & : - \ p(a,b) \\
\text{goal}(b,a) & : - \ p(b,a) \\
\text{goal}(b,a) & : - \ p(b,b) \\
\text{goal}(a,b) & : - \ p(a,a) \\
\text{goal}(a,b) & : - \ p(a,b) \\
\text{goal}(b,b) & : - \ p(b,a) \\
\text{goal}(b,b) & : - \ p(b,b)
\end{align*}

Results

\begin{align*}
\text{goal}(a,a) \\
\text{goal}(a,b)
\end{align*}
### Rule

\[ \text{goal}(X,Z) \ :- \ p(X,Y) \]

### Herbrand Universe
\{a, b, f(a), f(b), f(f(a)), \ldots\}

### Dataset
\{p(a,a)\}

### Instances

<table>
<thead>
<tr>
<th>Instances</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>goal(a,a) :- p(a,a)</td>
<td>goal(a,a)</td>
</tr>
<tr>
<td>goal(a,b) :- p(a,a)</td>
<td>goal(a,b)</td>
</tr>
<tr>
<td>goal(a,f(a)) :- p(a,a)</td>
<td>goal(a,f(a))</td>
</tr>
<tr>
<td>goal(a,f(b)) :- p(a,a)</td>
<td>goal(a,f(b))</td>
</tr>
<tr>
<td>goal(a,f(f(a))) :- p(a,a)</td>
<td>goal(a,f(f(a)))</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

### Unbound Variables in Head
Query

\[ \text{goal}(X) := p(X,Y) \land \neg p(Y,Z) \]

Herbrand Universe \{a, b, c\}

Dataset \{p(a, b), p(b, c)\}

What is the result?
Query

\[ \text{goal}(X) :- \ p(X,Y) \ & \ \neg p(Y,Z) \]

Possible Meanings

Find all \( X \) such that \( p(X,Y) \) is true and there is no \( Z \) for which \( p(Y,Z) \) is true.

Find all \( X \) such that \( p(X,Y) \) is true and there is some \( Z \) for which \( p(Y,Z) \) is false.
Query

\[ \text{goal}(X) := \text{p}(X, Y) \land \neg \text{p}(Y, Z) \]

Herbrand Universe  \{a, b, c\}

Dataset  \{\text{p}(a, b), \text{p}(b, c)\}

Results

Find all \(X\) such that \(\text{p}(X, Y)\) is true and there is no \(Z\) for which \(\text{p}(Y, Z)\) is true.

\{\text{goal}(b)\}

Find all \(X\) such that \(\text{p}(X, Y)\) is true and there is some \(Z\) for which \(\text{p}(Y, Z)\) is false.

\{\text{goal}(a), \text{goal}(b)\}
### Unbound Variables in Negation

#### Unsafe Rule

\[
goal(X) :- p(X,Y) \& \sim p(Y,Z)
\]

#### Herbrand Universe

\{a, b, c\}

#### Dataset

\{p(a,b), p(b,c)\}

#### Instances

<table>
<thead>
<tr>
<th>Instances</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>goal(a) :- p(a,b) &amp; \sim p(b,a)</td>
<td>goal(a)</td>
</tr>
<tr>
<td>goal(a) :- p(a,b) &amp; \sim p(b,b)</td>
<td>goal(a)</td>
</tr>
<tr>
<td>goal(a) :- p(a,b) &amp; \sim p(b,c)</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>goal(b) :- p(b,c) &amp; \sim p(c,a)</td>
<td>goal(b)</td>
</tr>
<tr>
<td>goal(b) :- p(b,c) &amp; \sim p(c,b)</td>
<td>goal(b)</td>
</tr>
<tr>
<td>goal(b) :- p(b,c) &amp; \sim p(c,c)</td>
<td>goal(b)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Predefined Concepts
Predefined Concepts

Evaluable Functions
- Arithmetic Functions (e.g. plus, times, min, max, etc.)
- String functions (e.g. concatenate, string matching, etc.)
- Other (e.g. converting between formulas and strings, etc.)
- Aggregates (e.g. sets of objects with given properties)

Evaluable Relations
- evaluate
- same, distinct, mutex
Evaluable term - constant, variable, $f(t_1, \ldots, t_n)$

- $f$ is a predefined function or user-defined function (later)
- *i.e. not a constructor / function constant*

$t_1, \ldots, t_n$ are evaluable terms

**Examples**

- `plus(2,3)` → 5
- `stringappend("abc","def")` → "abcdef"
- `stringify(vinay)` → "vinay"
- `symbolize("vinay")` → vinay

- `min(plus(2,3),times(2,3))` → 5

Many predefined functions are variadic, e.g. `plus(2,3,4)`.
Dataset \{h(a,2), w(a,3), h(b,4), w(b,2)\}

Possible Rule
\[
\text{goal}(X, \text{times}(H,W)) :- h(X,H) \& w(X,W)
\]

Results
\[
\text{goal}(a, \text{times}(2,3))
\]
\[
\text{goal}(b, \text{times}(4,2))
\]
evaluate(x,v)
  x is a term
  v is the value of x

Examples
  goal :- evaluate(times(2,3),6)

  goal :- evaluate(plus(times(2,3),4),10)

  goal(X,A) :-
    h(X,H) & w(X,W) & evaluate(times(H,W),A)

Safety: unbound variables allowed in second argument only.
Example

good(Z) :-
    evaluate(min(plus(2,3),times(2,3)),Z)

Result

good(5)
Aggregate operators are used to create sets of answers as terms and then count, add, average those sets.

Predefined Aggregates
- setofall
- countofall

Dataset \( \{p(a,b), p(a,c), p(b,d)\} \)

Examples
- countofall(Z, p(a,Z)) \(\rightarrow 2\)
- setofall(Z, p(a,Z)) \(\rightarrow [b,c]\)
Dataset \{p(a,b), p(a,c), p(b,d)\}

Example

\[
\text{goal}(X,L) :- \\
p(X,Y) \ & \ \\
evaluate(\text{countofall}(Z,p(X,Z)),L)
\]

Result \{\text{goal}(a,2), \text{goal}(b,1)\}

Example

\[
\text{goal}(X,L) :- \\
p(X,Y) \ & \ \\
evaluate(\text{setofall}(Z,p(X,Z)),L)
\]

Result \{\text{goal}(a,[b,c]), \text{goal}(b,[d])\}
Identity

\text{same}(t_1,t_2)\ is\ true\ iff\ t_1\ and\ t_2\ are\ identical

Difference

\text{distinct}(t_1,t_2)\ is\ true\ iff\ t_1\ and\ t_2\ are\ different
\text{mutex}(t_1,\ldots,t_n)\ is\ true\ iff\ t_1,\ldots,t_2\ are\ all\ different

Examples

\text{same}(a,a)\quad is\ true
\text{same}(a,b)\quad is\ false
\text{distinct}(a,a)\quad is\ false
\text{distinct}(a,b)\quad is\ true
\text{mutex}(a,b,c)\quad is\ true

Safety: No unbound variables allowed!!!
NB: This is **not** ordinary equality (e.g. $2+2 = 4$)

\[
\begin{align*}
\text{same}(\text{plus}(2,2), 4) & \quad \text{is false} \\
\text{distinct}(\text{plus}(2,2), 4) & \quad \text{is true}
\end{align*}
\]

NB: Use evaluate to get ordinary equality

\[
\begin{align*}
\text{evaluate}(\text{plus}(2,2), V) \ & \& \text{same}(V, 4) & \quad \text{is true} \\
\text{evaluate}(\text{plus}(2,2), 4) & \quad \text{is true}
\end{align*}
\]
Epilog provides a means to define new evaluable functions in terms of existing functions.

**Example**

\[
\begin{align*}
    f(X) & := \text{plus}(\text{pow}(X, 2), \text{times}(2, X), 1) \\
    \text{goal}(Z) & := \text{evaluate}(f(3), Z)
\end{align*}
\]

User-defined functions are quite useful in practice because they make some rules more readable and they can be evaluated very efficiently.

*NB: We won't be talking more about user-defined functions.*