Logic Programming

Query Optimization

Michael Genesereth
Computer Science Department
Stanford University
Two queries are **semantically equivalent** if and only if they produce identical results for every dataset.

Query 1:
\[
\text{goal}(X,Y) :- \ p(X) \ & \ r(X,Y) \ & \ q(X)
\]

Query 2:
\[
\text{goal}(X,Y) :- \ p(X) \ & \ q(X) \ & \ r(X,Y)
\]
Syntactically different but semantically equivalent queries may have different computational properties.

Query 1: $O(n^4)$

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

Query 2: $O(n^3)$

$$\text{goal}(X,Y) :- \ p(X) \ & \ q(X) \ & \ r(X,Y)$$
Types of Reformulation
   Logical - deleting and/or rearranging subgoals and rules
   Conceptual - changing vocabulary

Types of Logical Reformulation
   Rule Removal
   Subgoal Removal
   Subgoal Ordering

Types of Conceptual Reformulation
   Triples vs Wide Relations
   Minimal Spanning Trees
   Reification and Relationalization
Rule Removal
Useless Rules

Example:
\[
\text{goal}(X) :- \ p(X,Y) \ & \ q(Y) \ & \ \text{false}
\]

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

*Useless rules produce no results.*
Example:

\[
\begin{align*}
goal(X) & :\:- p(X,Y) & \& q(Y) & \& r(Y) \\
goal(X) & :\:- p(X,Y) & \& q(Y)
\end{align*}
\]

*Redundant rules produce only results produced by other rules, i.e. answers to one rule are a subset of answers to the other.*
Trickier Cases

Redundant Rules:

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

Non-Redundant Rules:

\[ \text{goal}(X) ::= p(X, b) \land q(b) \land r(Z) \]
\[ \text{goal}(X) ::= p(X, Y) \land q(Y) \land r(c) \]
A rule $r_1$ **subsumes** a rule $r_2$ if and only if it is possible to replace some or all of the variables of $r_1$ in such a way that the heads are the same and all of subgoals of $r_1$ are members of the body of $r_2$.

\[
\begin{align*}
goal(X) & :\ p(X,Y) & \& q(Y) \\
goal(X) & :\ p(X,b) & \& q(b) & \& r(Z)
\end{align*}
\]

Here, the first rule subsumes the second. We just replace $X$ in the first rule by itself and replace $Y$ by $b$, with the following result.

\[
goal(X) :- p(X,b) & \& q(b)
\]
Start with rule 1 and rule 2 as inputs where (a) the heads are identical and (b) neither rule contains any negations.

(1) Create a substitution in which each variable in rule 2 is bound to a distinct, new symbol.

(2) Create a **canonical dataset** consisting of the subgoals of rule 2 where all variables are replaced by these bindings.

(3) Substitute the bindings for the head variables in rule 1.

(4) Evaluate this modified rule on the dataset created in (2). If there are answers, then rule 1 subsumes rule 2. If not, then rule 1 does *not* subsume rule 2.
Example

Inputs

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

Substitution: \( \{x\leftarrow c1, z\leftarrow c3\} \)

Canonical Dataset: \( \{p(c1,b), q(b), r(c3)\} \)

Evaluate: \( p(c1,Y) \& q(Y) \)
Result: \( \{Y\leftarrow b\} \)

The first rule \textit{does} subsume the second.
Example

Inputs

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

Substitution: \(\{x \leftarrow c1, y \leftarrow c2\}\)

Canonical Dataset: \(\{p(c1, c2), q(c2)\}\)

Evaluate: \(p(c1, b) \& q(b) \& r(Z)\)

Result: failure

The first rule \textit{does not} subsume the second.
Rule Removal Technique

Compare every rule to every other rule (quadratic). If one rule subsumes another, it is okay to drop the subsumed rule.

NB: Applies only to rules with no negative subgoals and no predefined relations.

NB: The technique is sound in that it is guaranteed to produce an equivalent query.

NB: In the absence of any constraints on datasets to which the rules are applied, it is also guaranteed to be complete in that all surviving rules are needed for some dataset.

NB: In the face of constraints, it may be possible to drop rules that are not detected by this method, i.e. not complete.
If heads are not identical, they can sometimes be made identical by consistently replacing variables while avoiding clashes.

Original rules:
\[
\begin{align*}
goal(X) & :\, p(X,b) & \& q(b) & \& r(Z) \\
goal(U) & :\, p(U,V) & \& q(V)
\end{align*}
\]

Equivalent rules:
\[
\begin{align*}
goal(X) & :\, p(X,b) & \& q(b) & \& r(Z) \\
goal(X) & :\, p(X,V) & \& q(V)
\end{align*}
\]

There are other extensions for dealing with rules involving negations and built-ins and constraints.
Subgoal Removal
Original Rule:

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

Equivalent Reformulation:

\[
\text{goal}(X,Y) :- p(X,Y) \& q(Y)
\]
Subgoal Removal Technique

Accept query rule as input.

(1) Delete a subgoal.
(2) Check whether the resulting rule subsumes the original.
(3) If yes, continue. If no, try a different subgoal.

Output the result.
Original Rule:
\[
\text{goal}(X,Y) :- \ p(X,Y) \ & \ q(Y) \ & \ q(Z)
\]

Delete first subgoal - does not subsume (and not safe):
\[
\text{goal}(X,Y) :- \ q(Y) \ & \ q(Z) \quad \times
\]

Delete second subgoal - does not subsume:
\[
\text{goal}(X,Y) :- \ p(X,Y) \ & \ q(Z) \quad \times
\]

Delete third subgoal - subsumes original:
\[
\text{goal}(X,Y) :- \ p(X,Y) \ & \ q(Y) \quad \checkmark
\]
This technique is *sound* in that it is guaranteed to produce an equivalent query.
In the absence of any constraints, this method is guaranteed to be complete in that all surviving subgoals are needed for some dataset.

In the presence of constraints, it may be possible to drop more subgoals, i.e. not complete.

\[
\text{goal}(X,Y) \leftarrow \text{father}(X,Y) \land \text{male}(X)
\]

There are extensions that deal with constraints. See literature on the chase.
Subgoal Ordering
Original Rule

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

Reformulation

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

Original Rule

\[ \text{goal}(X, Y) \leftarrow p(X) \land r(X, Y) \land q(X) \]

\[ (n^2 + 2n) + n*(((n^2 + 2n) + n*(n^2 + 2n)) = n^4 + 3n^3 + 3n^2 + 2n) \]

Reformulation

\[ \text{goal}(X, Y) \leftarrow p(X) \land q(X) \land r(X, Y) \]
Analysis

Original Rule
\[
\text{goal}(X,Y) := p(X) \& r(X,Y) \& q(X)
\]
\[
(n^2 + 2n) + n*((n^2 + 2n) + n*(n^2 + 2n)) = n^4 + 3n^3 + 3n^2 + 2n
\]

Reformulation
\[
\text{goal}(X,Y) := p(X) \& q(X) \& r(X,Y)
\]
\[
(n^2 + 2n) + n*((n^2 + 2n) + 1*(n^2 + 2n)) = 2n^3 + 5n^2 + 2n
\]
Accept query rule as input.

(1) Create new query with head of input and empty body.

(2) Iterate through subgoals. On encountering one with all variables bound in subgoals of new query, add to new query and remove from original query. If none found, remove first subgoal, add to new query, and repeat.

Output the new query.

Example:

\[
\text{goal}(X,Y) \leftarrow p(X) \land r(X,Y) \land q(X)
\]
\[
\text{goal}(X,Y) \leftarrow
\]
Accept query rule as input.

(1) Create new query with head of input and empty body.

(2) Iterate through subgoals. On encountering one with all variables bound in subgoals of new query, add to new query and remove from original query. If none found, remove first subgoal, add to new query, and repeat.

Output the new query.

Example:

\[
\text{goal}(X,Y) := p(X) \& r(X,Y) \& q(X) \\
\text{goal}(X,Y) := p(X)
\]
Accept query rule as input.

(1) Create new query with head of input and empty body.

(2) Iterate through subgoals. On encountering one with all variables bound in subgoals of new query, add to new query and remove from original query. If none found, remove first subgoal, add to new query, and repeat.

Output the new query.

Example:

\[
goal(X, Y) :\leftarrow p(X) \& r(X, Y) \& q(X) \\
goal(X, Y) :\leftarrow p(X) \& q(X)
\]
Accept query rule as input.

(1) Create new query with head of input and empty body.

(2) Iterate through subgoals. On encountering one with all variables bound in subgoals of new query, add to new query and remove from original query. If none found, remove first subgoal, add to new query, and repeat.

Output the new query.

Example:

\[
\begin{align*}
goal(X, Y) & : = \ p(X) \ & \& \ r(X, Y) \ & \& \ q(X) \\
goal(X, Y) & : = \ p(X) \ & \& \ q(X) \ & \& \ r(X, Y)
\end{align*}
\]
Example
Example

SEND
+MORE
-----
MONEY
One Solution

digit(1)       digit(6)
digit(2)       digit(7)
digit(3)       digit(8)
digit(4)       digit(9)
digit(5)       digit(0)

goal(S,E,N,D,M,O,R,Y) :-
    digit(S) & digit(E) & digit(N) & digit(D) &
    digit(M) & digit(O) & digit(R) & digit(Y) &
    M!=0 & M!=S & M!=E & M!=N & M!=D &
    O!=S & O!=E & O!=N & O!=D & O!=M &
    evaluate(S*1000+E*100+N*10+D,U) &
    evaluate(M*1000+O*100+R*10+E,V) &
    evaluate(M*10000+O*1000+N*100+E*10+Y,W) &
    evaluate(plus(U,V),W)
Computational Analysis

Data

digit(1)       digit(6)
digit(2)       digit(7)
digit(3)       digit(8)
digit(4)       digit(9)
digit(5)       digit(0)

Rule

goal(S,E,N,D,M,O,R,Y) :-
digit(S) & digit(E) & digit(N) & digit(D) &
digit(M) & digit(O) & digit(R) & digit(Y) & ...

Analysis

10x10x10x10x10x10x10x10 = 10^8 = 100,000,000 cases
111,111,110 unifications
Running time ~minutes
Another Solution

goal(S,E,N,D,M,O,R,Y) :-
digit(S) & S!=0 &
digit(E) & E!=S &
digit(N) & N!=S & N!=E &
digit(D) & D!=S & D!=E & D!=N &
digit(M) & M!=0 & M!=S & M!=E & M!=N & M!=D &
digit(O) & O!=S & O!=E & O!=N & O!=D & O!=M &
digit(R) & R!=S & R!=E & R!=N & R!=D &
    R!=M & R!=O &
digit(Y) & Y!=S & Y!=E & Y!=N & Y!=D &
    Y!=M & Y!=O & Y!=R &
evaluate(S*1000+E*100+N*10+D,U) &
evaluate(M*1000+O*100+R*10+E,V) &
evaluate(M*10000+O*1000+N*100+E*10+Y,W) &
evaluate(plus(U,V),W)
Another Solution

\[\text{goal}(S,E,N,D,M,O,R,Y) \ :- \]
\[\text{digit}(S) \ & \ \text{mutex}(S,0) \ & \]
\[\text{digit}(E) \ & \ \text{mutex}(S,E) \ & \]
\[\text{digit}(N) \ & \ \text{mutex}(S,E,N) \ & \]
\[\text{digit}(D) \ & \ \text{mutex}(S,E,N,D) \ & \]
\[\text{digit}(M) \ & \ \text{distinct}(M,0) \ & \ \text{mutex}(S,E,N,D,M) \ & \]
\[\text{digit}(O) \ & \ \text{mutex}(S,E,N,D,M,O) \ & \]
\[\text{digit}(R) \ & \ \text{mutex}(S,E,N,D,M,O,R) \ & \]
\[\text{digit}(Y) \ & \ \text{mutex}(S,E,N,D,M,O,R,Y) \ & \]
\[\text{evaluate}(S*1000+E*100+N*10+D,XX) \ & \]
\[\text{evaluate}(M*1000+O*100+R*10+E,YY) \ & \]
\[\text{evaluate}(M*10000+O*1000+N*100+E*10+Y),ZZ) \ & \]
\[\text{evaluate}(XX+YY,ZZ) \]
Computational Analysis

Goal

goal(S,E,N,D,M,O,R,Y) :-
digit(S) & mutex(S,0) &
digit(E) & mutex(S,E) &
digit(N) & mutex(S,E,N) &
digit(D) & mutex(S,E,N,D) &
digit(M) & distinct(M,0) & mutex(S,E,N,D,M) &
digit(O) & mutex(S,E,N,D,M,O) &
digit(R) & mutex(S,E,N,D,M,O,R) &
digit(Y) & mutex(S,E,N,D,M,O,R,Y) & ...

Analysis

10x9x8x7x6x5x4x3 = 1,814,400 cases
5,989,558 unifications
Running time ~20 seconds
Goal

goal(S,E,N,D,M,O,R,Y) :-
    digit(S) & mutex(S,0) &
    digit(E) & mutex(S,E) &
    digit(N) & mutex(S,E,N) &
    digit(D) & mutex(S,E,N,D) &
    digit(M) & same(M,1) & mutex(S,E,N,D,M) &
    digit(O) & mutex(S,E,N,D,M,O) &
    digit(R) & mutex(S,E,N,D,M,O,R) &
    digit(Y) & mutex(S,E,N,D,M,O,R,Y) & ...

Analysis

10x9x8x7x6x5x4x3 = 320,400 cases
699,858 unifications
Running time < 2 seconds
Data
digit(9)
digit(5)
digit(6)
digit(7)
digit(7)
digit(1)
digit(0)
digit(0)
digit(8)
digit(2)
digit(3)
digit(4)

Analysis

\[ 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 = 36 \text{ cases} \]
Interpreted \(\sim 0\) seconds
Computational Analysis

Data
digit(9)
digit(5)
digit(6)
digit(7)
digit(7)
digit(1)
digit(0)
digit(0)
digit(8)
digit(8)
digit(2)
digit(3)
digit(4)

Analysis

\[ 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 = 36 \text{ unifications} \]

Interpreted \(~\sim\) 0 seconds
Narrow and Wide Relations
Represent wide relations as collections of binary relations.

Wide Relation:
\[
\text{student}(\text{Student}, \text{Department}, \text{Advisor}, \text{Year})
\]

Binary Relations:
\[
\begin{align*}
\text{student.major}(\text{Student}, \text{Department}) \\
\text{student.advisor}(\text{Student}, \text{Faculty}) \\
\text{student.year}(\text{Student}, \text{Year})
\end{align*}
\]

Always works when there is a field of the wide relation (called the key) that uniquely specifies the values of the other elements. If none exists, possible to create one.
Wide Relation:

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

Triples:

\[ p1(a,d) \quad p2(a,e) \]
\[ p1(b,d) \quad p2(b,e) \]
\[ p1(c,d) \quad p2(c,e) \]
Wide Relation:
\[ p(a,d,e) \]
\[ p(b,d,e) \]
\[ p(c,d,e) \]

Query: \[ \text{goal}(X) :- p(X,d,e) \]
Cost without indexing: 3  
Cost with indexing: 3

Triples:
\[ p1(a,d) \quad p2(a,e) \]
\[ p1(b,d) \quad p2(b,e) \]
\[ p1(c,d) \quad p2(c,e) \]

Query: \[ \text{goal}(X) :- p1(X,d) & p2(X,e) \]
Cost without indexing: 24  
Cost with indexing: 9
Minimal Spanning Trees
Social Isolation Cells
Vocabulary:
People - a, b, c, d, e, f, g, h, i, j, ...
Interaction - \( r/2 \)

Example:
\[
\begin{align*}
  r(a,b) \\
  r(a,e) \\
  r(b,a) \\
  r(b,c) \\
  r(c,b) \\
  r(d,e) \\
  r(e,a) \\
  r(e,d) \\
\end{align*}
\]

NB: Possible to represent undirected with only one factoid per arc rather than two, but we will ignore that for now.
Are two people are in the same cell?

\[
\text{goal}(a,e) \ :- \ \text{r}(a,e)
\]
\[
\text{goal}(a,e) \ :- \ \text{r}(a,Y) \ & \ \text{r}(Y,e)
\]
\[
\text{goal}(a,e) \ :- \ \text{r}(a,Y1) \ & \ \text{r}(Y1,Y2) \ & \ \text{r}(Y2,e)
\]
\[
\text{goal}(a,e) \ :- \ \text{r}(a,Y1) \ & \ \text{r}(Y1,Y2) \ & \ \text{r}(Y2,Y3) \ & \ \text{r}(Y3,e)
\]

Number of unifications for \text{goal}(a,e) \ (with indexing):

\[
8 \\
8 + 4*8 = 40 \\
8 + 4*(8 + 4*8)) = 168 \\
8 + 4*(8 + 4*(8 + 4*8)) = 680 \\
\text{Total:} \ 896\]
Precompute and store the transitive closure of $r$

$\begin{align*}
& r(a,b) & r(b,a) & r(c,a) & r(d,a) & r(e,a) \\
& r(a,c) & r(b,c) & r(c,b) & r(d,b) & r(e,b) \\
& r(a,d) & r(b,d) & r(c,d) & r(d,c) & r(e,c) \\
& r(a,e) & r(b,e) & r(c,e) & r(d,e) & r(e,d)
\end{align*}$

Are two people are in the same cell?

$\text{goal}(a,e) : - r(a,e)$

Number of unifications for $\text{goal}(a,e)$ (with indexing):

8

Number of factoids for $n$ objects:

$8 \times n$
Assign a number for each group and store with people

\[
\begin{align*}
  r(a,1) & \quad r(f,2) & \ldots \\
  r(b,1) & \quad r(g,2) & \ldots \\
  r(c,1) & \quad r(h,2) & \ldots \\
  r(d,1) & \quad r(i,2) & \ldots \\
  r(e,1) & \quad r(j,2) & \ldots 
\end{align*}
\]

Are two people are in the same cell?

\[
\text{goal}(a,e) :- r(a,N) \& r(e,N)
\]

Number of unifications for \( \text{goal}(a,e) \) (with indexing):

\[
2
\]

Number of factoids for \( n \) objects:

\[
n
\]
Lambda

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>goal(a,e)</td>
<td>p(a,Y1) &amp; p(Y1,Y2) &amp; p(Y2,Y3) &amp; p(Y3,e) &amp; false</td>
</tr>
</tbody>
</table>

680 unification(s)
Lambda

- Goal (a, e)
- Query: p(a, N) & p(e, N)

2 unifications

Vocabulary:
- p(a, 1)
- p(b, 1)
- p(c, 1)
- p(d, 1)
- p(e, 1)