Logic Programming
Optimization

Michael Genesereth
Computer Science Department
Stanford University
Last lecture:
  Transition Rules
  Closed Dynamic Systems
  Open Dynamic Systems

This lecture:
  Materialization of Views
  Differential Update
  Conceptual Reformulation of Dynamic Logic Programs
Materialization
Ruleset

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

Computation Cost

\( r \) computed multiple times in computing \( \text{goal} \)

Storage

Just \( p \) stored
\( O(n^2) \) in worst case
Example with $r$ Materialized

**Ruleset**

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

**Computation Cost**

$r$ computed just once

**Storage**

$r$ stored

$O(n)$ in worst case
Example with goal Materialized

**Ruleset**

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

**Computation Cost**

Lookup cost for `goal`

**Storage**

`goal` stored

\[O(n^3)\] in worst case
Conversion process. (1) Rename in all rules defining view to be materialized and replace in all other rules. (2) Write transition rules to relate old and new name.

Ruleset

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

Renaming

\[
\text{goal}(X,Y,Z) :\equiv rr(X) \& rr(Y) \& rr(Z) \\
r(X) :\equiv p(X,Y) \& p(Y,X)
\]

Transition Rules

\[
rr(X) \implies \neg rr(X) \\
r(X) \implies rr(X)
\]
Rule set

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

Old Materialized Dataset: \{rr(a), rr(b)\}

New Base Dataset: \{p(c,d), p(d,c)\}

Negative Update: \{rr(a), rr(b)\}

Positive Update: \{rr(c), rr(d)\}

New Materialized Dataset: \{rr(c), rr(d)\}
Differential Update
Conversion process. (1) Rename in all rules defining view to be materialized and replace in all other rules. (2) Write transition rules to relate old and new name. (3) *Remove rules defining the materialized view and replace with differential rules relating changes to base relations and changes to views.*
Ruleset

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

Materialization Version

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

Differential Version

\[\text{goal}(X,Y,Z) :- \text{rr}(X) \& \text{rr}(Y) \& \text{rr}(Z)\]
\[\text{pp}(X,Y) \implies \text{rr}(X)\]
\[\text{pm}(X,Y) \implies \neg \text{rr}(X)\]
Ruleset

\[
\begin{align*}
\text{pp}(X,Y) & \implies \text{rr}(X) \\
\text{pm}(X,Y) & \implies \neg \text{rr}(X)
\end{align*}
\]

Old Dataset: \{\text{rr}(a), \text{rr}(b)\}

Change Request: \{\text{pp}(c,d), \text{pp}(d,c)\}

Negative Update: {}

Positive Update: \{\text{rr}(c), \text{rr}(d)\}

New Dataset: \{\text{rr}(a), \text{rr}(b), \text{rr}(c), \text{rr}(d)\}
Tic Tac Toe

X |     | X
---|-----|---
   | O   |   
---|-----|---
   |     | X

cell(1,1,x)  
cell(1,2,b)  
cell(1,3,b)  
cell(2,1,b)  
cell(2,2,o)  
cell(2,3,b)  
cell(3,1,b)  
cell(3,2,b)  
cell(3,3,x)  
control(o)
Non-Differential Rules

\[\text{mark}(J,K) \& \text{cell}(M,N,R) \implies \neg\text{cell}(M,N,R)\]

\[\text{mark}(J,K) \& \text{control}(R) \implies \text{cell}(J,K,R)\]

\[\text{mark}(J,K) \& \text{cell}(M,N,R) \& \text{distinct}(R,b) \implies \text{cell}(M,N,R)\]

\[\text{mark}(J,K) \& \text{cell}(M,N,R) \& \text{distinct}(J,M) \implies \text{cell}(M,N,b)\]

\[\text{mark}(J,K) \& \text{cell}(M,N,R) \& \text{distinct}(K,N) \implies \text{cell}(M,N,b)\]
Differential Rules

\[ \text{mark}(M,N) \land \text{control}(R) \implies \text{cell}(M,N,R) \]

\[ \text{mark}(M,N) \implies \neg \text{cell}(M,N,b) \]
## Computational Benefit

### Number of playouts per second*

<table>
<thead>
<tr>
<th></th>
<th>Unindexed</th>
<th>Indexed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-differential</td>
<td>880</td>
<td>1070</td>
</tr>
<tr>
<td>Differential</td>
<td>1340</td>
<td>1780</td>
</tr>
</tbody>
</table>

* Larger is better.
Conceptual Reformulation
Materialization and differential update sometimes have computational benefit.

In *some* cases, none of the views may be suitable for materialization.

However it is *sometimes* possible to rewrite programs into equivalent programs that have suitable views.
samefamily
Axiomatization

redwin :-
    left(X) & redpath(X,Y) & right(Y)

redpath(X,X) :-
    cell(X,red)

redpath(X,Z) :-
    cell(X,red) & next(X,Y) & redpath(Y,Z)

Results
    Very expensive if path exists
    Can run forever if not
Improved Axiomatization

Axiomatization

redwin :-
    left(X) & redpath(X,Y,nil) & right(Y)

redpath(X,X,nil) :-
    cell(X,red)

redpath(X,Z,P) :-
    cell(X,red) & next(X,Y) & ~member(Y,P) &
    redpath(Y,Z,Y!P)

Results

Does not run forever
can take ~ 1 second to compute in bad cases
Result:
takes <1 millisecond to compute in worst cases
Better Still

Result:

takes <1 millisecond to compute in worst cases
Axiomatization

\[
\text{redwin} : - \\
\text{left}(X) \land \text{rep}(X,K) \land \text{rep}(Y,K) \land \text{right}(Y)
\]

Results

takes less than 1 \textit{millisecond} in the worst case
additional update cost negligible

Exercise
Write the transition rules for the better still axiomatization.