Datalog with negation

Serge Abiteboul

INRIA

6 mai 2009
Datalog with negation

Read Chapter 15 of [AHV]

transitive closure

\[ T(x, y) \leftarrow G(x, y) \]
\[ T(x, y) \leftarrow G(x, z), T(z, y). \]

complement \( CT \) of \( T \) (pairs of disconnected nodes in a graph \( G \))

\[ CT(x, y) \leftarrow \neg T(x, y) \]

To simplify, assume an active domain interpretation

datalog\( ^{-} \)

allow in bodies of rules, literals of the form \( \neg R_i(u_i) \)

\( \neg = (x, y) \) is denoted by \( x \neq y \)
notation: $J|S$ is restriction of $J$ to $S$

extend the immediate consequence operator

For $K$ over $sch(P)$, $A$ is $T_P(K)$ if

- $A \in K|edb(P)$, or
- $A \leftarrow A_1, \ldots, A_n$ an instantiation of a rule in $P$ such that
  1. if $A_i$ is a positive literal then $A_i \in K$
  2. if $A_i = \neg B_i$ then $B_i \notin K$

$T_P$ is not inflationary
Problems

$T_P$ may not have any fixpoint

- $P_1 = \{ p \leftarrow \neg p \}$

$T_P$ may have several minimal fixpoints containing $\mathbb{I}$

- $P_2 = \{ p \leftarrow \neg q, q \leftarrow \neg p \}$
- two minimal fixpoints (containing the $\emptyset$) : $\{ p \}$ and $\{ q \}$.

Now consider $\{ T_P^i(\emptyset) \}_{i>0}$

$T_P$ has a least fixpoint but sequence diverges

- $P_3 = \{ p \leftarrow \neg r; r \leftarrow \neg p; p \leftarrow \neg p, r \}$
- $T_{P_3}$ has a least fixpoint $\{ p \}$
- $\{ T_{P_3}^i(\emptyset) \}_{i>0}$ alternates between $\emptyset$ and $\{ p, r \}$

$T_P$ has a least fixpoint and $\{ T_P^i(\emptyset) \}_{i>0}$ converges to something else

- $P_4 = \{ p \leftarrow p, q \leftarrow q, p \leftarrow \neg p, q \leftarrow \neg p \}$
- $\{ T_{P_4}^i(\emptyset) \}_{i>0}$ converges to $\{ p, q \}$
- least fixpoint of $T_{P_4}$ is $\{ p \}$.
Rules of the form $P(x, y) \leftarrow P(x, y)$
- change the semantics of program
- force $T_P$ to be inflationnary so force convergence
- correspond to tautologies $p \lor \neg p$
- transitive closure example

Model theoretic semantics: Problems
some programs have no model
some have no least model containing I
when a program has several minimal models, choose between them
Semipositive datalog

only apply negation to \textit{edb} relations

semipositive program that is neither in datalog nor in CALC :

$$T(x, y) \leftarrow \neg G(x, y)$$
$$T(x, y) \leftarrow \neg G(x, z), T(z, y).$$

Intuition : one could eliminate negation from semi-positive programs by adding, for each \textit{edb} relation \(R'\), a new \textit{edb} relation \(\overline{R'}\) holding the complement of \(R'\) (w.r.t. the active domain), and replacing \(\neg R'(x)\) by \(\overline{R'}(x)\).

many nice properties of positive datalog
\(\Sigma_P\) has a unique minimal model \(J\) satisfying \(J|\text{edb}(P) = I\)
\(T_P\) has a unique minimal fixpoint \(J\) satisfying \(J|\text{edb}(P) = I\).

These coincide
complement of transitive closure is not a semi-positive program

closure under composition : stratified datalog
stratification of a datalog$^-$ program $P$

sequence of datalog$^-$ programs $P^1, \ldots, P^n$ and some
mapping $\sigma$ from idb($P$) to [1..n] such that

(i) $\{P^1, \ldots, P^n\}$ is a partition of $P$
(ii) for each $R$, all rules defining $R$ are in $P^{\sigma(R)}$
(iii) If $R(u) \leftarrow \ldots R'(v) \ldots$ is a rule in $P$, and $R'$ is an idb
relation, then $\sigma(R') \leq \sigma(R)$.
(iv) If $R(u) \leftarrow \ldots \neg R'(v) \ldots$ is a rule in $P$, and $R'$ is an idb
relation, then $\sigma(R') < \sigma(R)$.

each $P^i$ is called a stratum

the stratification of $P$ provides a parsing of $P$ as a sequence
of semipositive subprograms $P^1, \ldots, P^n$
Stratification examples

stratification of TCcomp

\[ T(x, y) \leftarrow G(x, y) \]
\[ T(x, y) \leftarrow G(x, z), T(z, y) \]
\[ CT(x, y) \leftarrow \neg T(x, y) \]

first stratum: first two rules (defining \( T \))
second stratum: third rule (defining \( CT \) using \( T \))
Stratification examples

\(P_7\) defined by

\[
\begin{align*}
    r_1 & : S(x) \leftarrow R'_1(x), \neg R(x) \\
    r_2 & : T(x) \leftarrow R'_2(x), \neg R(x) \\
    r_3 & : U(x) \leftarrow R'_3(x), \neg T(x) \\
    r_4 & : V(x) \leftarrow R'_4(x), \neg S(x), \neg U(x).
\end{align*}
\]

Then \(P_7\) has 5 distinct stratifications, namely,

\[
\begin{align*}
    \{r_1\}, \{r_2\}, \{r_3\}, \{r_4\} \\
    \{r_2\}, \{r_1\}, \{r_3\}, \{r_4\} \\
    \{r_2\}, \{r_3\}, \{r_1\}, \{r_4\} \\
    \{r_1, r_2\}, \{r_3\}, \{r_4\} \\
    \{r_2\}, \{r_1, r_3\}, \{r_4\}.
\end{align*}
\]

\(P_2 = \{p \leftarrow \neg q, q \leftarrow \neg p\}\)

no stratification
Testing stratification

Precedence graph $G_P$ of $P$
- vertexes: are the idb's of $P$
- edge $(R', R)$ with label + if $R'$ is used positively in some rule defining $R$
- edge $(R', R)$ with label - if $R'$ is used negative in some rule defining $R$

$P$ is stratifiable iff $G_P$ has no cycle containing a negative edge

Part of proof
$P$ is a program whose precedence graph $G_P$ has no cycle with negative edges
$C_1, \ldots, C_n$ the strongly connected components of $G_P$
$C_i \prec C_j$ : if there is an edge from $C_i$ to some node of $C_j$
$\prec$ is acyclic

Turn this partial order into a sort $C_{i_1}, \ldots, C_{i_n}$
This provides a stratification
Stratification : semantics

$P$ a program with stratification $\sigma = P^1, \ldots, P^n$ and $I$ and instance

$I_0 = I$

$I_i = I_{i-1} \cup P^i(I_{i-1}|edb(P^i))$

where $P^i$ is the semipositive semantics

$I_n$ is denoted $\sigma(I)$

Result : independent of the choice of a stratification

we denote it $P^{strat}(I)$

Result : $P$ stratified datalog$^-$ and $I$

1. $P^{strat}(I)$ is a minimal model of $\Sigma_P$
   whose restriction to $edb(P)$ equals $I$.

2. $P^{strat}(I)$ is a minimal fixpoint of $T_P$
   whose restriction to $edb(P)$ equals $I$.

3. $P^{strat}(I)$ is a “supported” model of $P$ relative to $I$
   ($J \subseteq T_P(J) \cup I$)

limited power
The well-founded semantics

accommodate incompleteness
3-valued instances: true, false, unknown
example: two players game
input $K$ with relation $moves$:

$$K(moves) = \{ \langle b, c \rangle, \langle c, a \rangle, \langle a, b \rangle, \langle a, d \rangle, \langle d, e \rangle, \langle d, f \rangle, \langle f, g \rangle \}$$

each player can move the position following a move edge
a player looses if he/she has no possible move
goal: compute the set of winning states

- \( d \) is winning: move to \( e \)
- \( f \) is winning: move to \( g \)

No winning strategy from \( a \), \( b \), or \( c \). Indeed, a given player can prevent the other from winning, essentially by forcing a non-terminating sequence of moves.

This will be the well-founded semantics for \( P_{\text{win}} \):

\[
\text{win}(x) \leftarrow \text{moves}(x, y), \neg \text{win}(y)
\]

(non stratifiable)

“3-valued model” \( J \) of \( P_{\text{win}} \), that agrees with \( K \) on \( \text{moves} \)

- \text{true} \hspace{1cm} \text{win}(d), \text{win}(f)
- \text{false} \hspace{1cm} \text{win}(e), \text{win}(g)
- \text{unknown} \hspace{1cm} \text{win}(a), \text{win}(b), \text{win}(c).

This will provide the well-founded semantics
3-valued instances

assume now that all facts \( R(u) \) are in the program as 
\[ R(u) \leftarrow \]
A 3-value instance: \( B(P) \rightarrow \{0, 1/2, 1\} \)
\( \mathbf{I}^0 \) false facts, \( \mathbf{I}^{1/2} \) unknown, \( \mathbf{I}^1 \) true
total instance if \( \mathbf{I}^{1/2} = \emptyset \)
E.g.: \( \mathbf{I}(p) = 1, \mathbf{I}(q) = 1, \mathbf{I}(r) = 1/2, \mathbf{I}(s) = 0 \)
written: \( \mathbf{I} = \{p, q, \neg s\} \)
\( \mathbf{I} \prec \mathbf{J} \) iff for each \( A \in B(P), \mathbf{I}(A) \leq \mathbf{J}(A) \)
(equivalently, \( \mathbf{I}^1 \subseteq \mathbf{J}^1 \) and \( \mathbf{I}^0 \supseteq \mathbf{J}^0 \))
Truth value of boolean combination of facts
\[ \hat{\mathbf{I}}(\beta \land \gamma) = \min\{\hat{\mathbf{I}}(\beta), \hat{\mathbf{I}}(\gamma)\} \]
\[ \hat{\mathbf{I}}(\beta \lor \gamma) = \max\{\hat{\mathbf{I}}(\beta), \hat{\mathbf{I}}(\gamma)\} \]
\[ \hat{\mathbf{I}}(\neg \beta) = 1 - \hat{\mathbf{I}}(\beta) \]
\[ \hat{\mathbf{I}}(\beta \leftarrow \gamma) = 1 \text{ if } \hat{\mathbf{I}}(\gamma) \leq \hat{\mathbf{I}}(\beta), \text{ and } 0 \text{ otherwise.} \]
careful
\[ p \leftarrow q \quad \text{and} \quad p \lor \neg q : \text{possibly different} \]
I satisfies $\alpha$ if $\hat{I}(\alpha) = 1$

win example

\[
\begin{align*}
\text{win}(a) & \leftarrow \text{moves}(a, d), \neg \text{win}(d) \\
\text{win}(a) & \leftarrow \text{moves}(a, b), \neg \text{win}(b)
\end{align*}
\]

first is true for $J$

$\hat{J}(\neg \text{win}(d)) = 0$, $\hat{J}(\text{moves}(a, d)) = 1$, $\hat{J}(\text{win}(a)) = 1/2$, $1/2 \geq 0$.

second is true

$\hat{J}(\neg \text{win}(b)) = 1/2$, $\hat{J}(\text{moves}(a, b)) = 1$, $\hat{J}(\text{win}(a)) = 1/2$, $1/2 \geq 1/2$

$\hat{J}(\text{win}(a) \lor \neg (\text{moves}(a, b) \land \neg \text{win}(b))) = 1/2$
3-valued minimal model for (extended) datalog

extended : datalog program with 0, 1/2 and 1 as literals in bodies

$3\text{-}T_P$ : of a 3-valued instance $I$ and $A \in B(P)$,

- 1 for some instantiation $A \leftarrow \text{body}$ and $\hat{I}(\text{body}) = 1$
- 0 for each instantiation $A \leftarrow \text{body}$ and $\hat{I}(\text{body}) = 0$
- $1/2$ otherwise

$P = \{ p \leftarrow 1/2 ; p \leftarrow q, 1/2 ; q \leftarrow p, r ; q \leftarrow p, s ; s \leftarrow q ; r \leftarrow 1 \}$

\[ 3\text{-}T_P(\{ \neg p, \neg q, \neg r, \neg s \}) = \{ \neg q, r, \neg s \} \]
\[ 3\text{-}T_P(\{ \neg q, r, \neg s \}) = \{ r, \neg s \} \]
\[ 3\text{-}T_P(\{ r, \neg s \}) = \{ r \} \]
\[ 3\text{-}T_P(\{ r \}) = \{ r \} \]
Result - 3-extended datalog programs

\[ P \text{ 3-extended datalog program} \]

1. \( 3-T_P \) is monotonic and the sequence \( \{3-T_P^i(\bot)\}_{i>0} \) is increasing and converges to the least fixpoint of \( 3-T_P \)

2. \( P \) has a \textbf{unique} minimal 3-valued model that equals the least fixpoint of \( 3-T_P \)

minimal is w.r.t. \( \prec \)
3-stable models of datalog$^\neg$

$P$ a datalog$^\neg$ program, $I$ a 3-valued instance over $\text{sch}(P)$

$P'$ ground version of $P$ given $I$

$\text{pg}(P,I)$ positivized ground version of $P$ given $I$ : replace each negative literal $\neg A$ by $\hat{I}(\neg A)$ (i.e., 0, 1 or $1/2$)

this is an extended datalog program

We denote its minimal model : $\text{conseq}_P(I)$

A 3-valued instance $I$ over $\text{sch}(P)$ is a 3-stable model of $P$ iff $\text{conseq}_P(I) = I$. 
Example: stable model

\[
P
\]

\[
p \leftarrow \neg r
q \leftarrow \neg r, p
s \leftarrow \neg t
t \leftarrow q, \neg s
u \leftarrow \neg t, p, s.
\]

3 3-stable models

\[
I_1 = \{p, q, t, \neg r, \neg s, \neg u\},
I_2 = \{p, q, s, \neg r, \neg t, \neg u\},
I_3 = \{p, q, \neg r\}.
\]
checking $I_3$

checking $I_3$ : positivized program

\[
\begin{align*}
p & \leftarrow 1 \\
q & \leftarrow 1, p \\
s & \leftarrow 1/2 \\
t & \leftarrow q, 1/2 \\
u & \leftarrow 1/2, p, s.
\end{align*}
\]

\[
\bot = \{\neg p, \neg q, \neg r, \neg s, \neg t, \neg u\}
\]

\[
3-T_{P'}(\bot) = \{p, \neg q, \neg r, \neg t, \neg u\}
\]

\[
(3-T_{P'})^2(\bot) = \{p, q, \neg r, \neg t\}
\]

\[
(3-T_{P'})^3(\bot) = (3-T_{P'})^4(\bot) = \{p, q, \neg r\}
\]

\[
\text{conseq}_P(I_3) = (3-T_{P'})^3(\bot) = I_3,
\]
each datalog\(\neg\) programs has at least one 3-stable model

\(P\) a datalog\(\neg\) program

The well-founded semantics of \(P\) \(P_{\text{wf}}(\emptyset) = \)

the 3-valued instance consisting of all positive and negative
facts belonging to all 3-stable models of \(P\)

\(P_{\text{wf}}(I) = P_{I}^{\text{wf}}(\emptyset)\)

example, \(P_{\text{win}}^{\text{wf}}(K) = J\)
Fixpoint characterization

previous description of the well-founded semantics
effective but very inefficient
more efficient one: “alternating fixpoint”

idea:

sequence \( \{I_i\}_{i \geq 0} \) of 3-valued instances
alternate between underestimates and overestimates of the
facts known in every 3-stable model of \( P \)

SEQUENCE

\[
\begin{align*}
I_0 &= \perp \quad \text{(all facts are false)} \\
I_{i+1} &= \text{conseq}_P(I_i)
\end{align*}
\]

each \( I_i \) is a total instance

observe that \( \text{conseq}_P \) is antimonotonic,
\( I \preceq J \) implies \( \text{conseq}_P(J) \preceq \text{conseq}_P(I) \)

since \( \perp \preceq I_1 \) and \( \perp \preceq I_2 \),

\[
I_0 \preceq I_2 \ldots \preceq I_{2i} \preceq I_{2i+2} \preceq \ldots \preceq I_{2i+1} \preceq I_{2i-1} \preceq \ldots \preceq I_1
\]
Fixpoint: examples

\[ P : \]
\[
P \leftarrow \neg r
\]
\[
q \leftarrow \neg r, p
\]
\[
s \leftarrow \neg t
\]
\[
t \leftarrow q, \neg s
\]
\[
u \leftarrow \neg t, p, s.
\]

\[ I_0 = \bot = \{ \neg p, \neg q, \neg r, \neg s, \neg t, \neg u \} \]
\[ I_1 = \{ p, q, \neg r, s, t, u \} \]
\[ I_2 = \{ p, q, \neg r, \neg s, \neg t, \neg u \} \]
\[ I_3 = \{ p, q, \neg r, s, t, u \} \]
\[ I_4 = \{ p, q, \neg r, \neg s, \neg t, \neg u \} \]
Fixpoint: examples

\( P_{\text{win}} \) and input \( K \)

for \( I_0 \), all move atoms are \textbf{false}

for each \( j \geq 1 \), \( I_j \)(moves) = \( K \)(moves)

\[
\begin{align*}
I_1 &= \{ \text{win}(a), \text{win}(b), \text{win}(c), \text{win}(d), \neg\text{win}(e), \text{win}(f), \neg\text{win}(f) \} \\
I_2 &= \{ \neg\text{win}(a), \neg\text{win}(b), \neg\text{win}(c), \text{win}(d), \neg\text{win}(e), \text{win}(f), \neg\text{win}(f) \} \\
I_3 &= I_1 \\
I_4 &= I_2
\end{align*}
\]
there are finitely many 3-valued instances for a given \( P \) 
these two sequences converge 
\( I_* : \) limit of increasing \( \{ I_{2i} \}_{i \geq 0} \) 
\( I^* : \) limit of decreasing \( \{ I_{2i+1} \}_{i \geq 0} \) 
\( I_* \prec I^* \) 

\em conseq_\!_P (I_*) = I^* \) and \( conseq_\!_P (I^*) = I_* \) 

\( I^* \) : 3-valued instance with facts known in both 

\[
I^*(A) = \begin{cases} 
1 & \text{if } I_*(A) = I^*(A) = 1 \\
0 & \text{if } I_*(A) = I^*(A) = 0 \text{ and} \\
1/2 & \text{otherwise.}
\end{cases}
\]
Theorem: $I^* = P^{wf}(\emptyset)$

Theorem

$P$ stratified datalog$^-$ program,

for each 2-valued instance $I$ over $edb(P)$, $P^{wf}(I) = P^{strat}(I)$. 
Example

input: binary relation $G$ + a unary relation $good$

$$bad(x) \leftarrow G(y, x), \neg good(y)$$
$$answer(x) \leftarrow \neg bad(x)$$

$K(G) = \{\langle b, c \rangle, \langle c, b \rangle, \langle c, d \rangle, \langle a, d \rangle, \langle a, e \rangle\}$, and
$K(good) = \{\langle a \rangle\}$.

as usual, we add the facts to program as unit clause $I_0 = \bot$ (containing all negated atoms).

omitting facts in good and $G$

<table>
<thead>
<tr>
<th></th>
<th>$bad$</th>
<th>$answer$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_0$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>$I_1$</td>
<td>${\neg a, b, c, d, e}$</td>
<td>${a, b, c, d, e}$</td>
</tr>
<tr>
<td>$I_2$</td>
<td>${\neg a, b, c, d, \neg e}$</td>
<td>${a, \neg b, \neg c, \neg d, \neg e}$</td>
</tr>
<tr>
<td>$I_3$</td>
<td>${\neg a, b, c, d, \neg e}$</td>
<td>${a, \neg b, \neg c, \neg d, e}$</td>
</tr>
<tr>
<td>$I_4$</td>
<td>${\neg a, b, c, d, \neg e}$</td>
<td>${a, \neg b, \neg c, \neg d, e}$</td>
</tr>
</tbody>
</table>

$I_* = I_* = I_* = I_3 = I_4$
Merci