Datalog with negation

Serge Abiteboul

INRIA

6 mai 2009

Datalog with negation

Read Chapter 15 of [AHV]

transitive closure

$$\begin{array}{lcl} T(x,y) & \leftarrow & G(x,y) \\ T(x,y) & \leftarrow & G(x,z), T(z,y). \end{array}$$

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

$$CT(x,y) \quad \leftarrow \quad \neg T(x,y)$$

To simplify, assume an active domain interpretation

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

$\neg = (x,y)$ is denoted by $x \neq y$

Fixpoint semantics : problems

notation : $\mathbf{J}|\mathbf{S}$ is restriction of $\mathbf{J}$ to $\mathbf{S}$

extend the immediate consequence operator

For $\mathbf{K}$ over $sch(P)$, A is $T_P(\mathbf{K})$ if

- $A \in \mathbf{K}|edb(P)$, 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 $A_i \in \mathbf{K}$
  2. if $A_i = \neg B_i$ then $B_i \notin \mathbf{K}$

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 **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 $p \vee \neg p$
- 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$^\neg$

only apply negation to *edb* relations

semipositive program that is neither in datalog nor in CALC :

$$\begin{array}{rcl} T(x,y) & \leftarrow & \neg G(x,y) \\ T(x,y) & \leftarrow & \neg G(x,z), T(z,y). \end{array}$$

Intuition : one could eliminate negation from semi-positive programs by adding, for each *edb* relation R', a new *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

Σ_P has a unique minimal model $\mathbf{J}$ satisfying $\mathbf{J}|edb(P) = \mathbf{I}$

T_P has a unique minimal fixpoint $\mathbf{J}$ satisfying $\mathbf{J}|edb(P) = \mathbf{I}$.

These coincide

complement of transitive closure is not a semi-positive program

closure under composition : stratified datalog$^\neg$

Stratified datalog$^\neg$

stratification of a datalog$^\neg$ program P

sequence of datalog$^\neg$ programs $P^1, \ldots, P^n$ and some mapping σ 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

$$\begin{array}{lcl} T(x,y) & \leftarrow & G(x,y) \\ T(x,y) & \leftarrow & G(x,z), T(z,y) \\ CT(x,y) & \leftarrow & \neg T(x,y) \end{array}$$

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

Stratification examples

P_7 defined by

$$\begin{array}{llcl} r_1 & S(x) & \leftarrow & R'_1(x), \neg R(x) \\ r_2 & T(x) & \leftarrow & R'_2(x), \neg R(x) \\ r_3 & U(x) & \leftarrow & R'_3(x), \neg T(x) \\ r_4 & V(x) & \leftarrow & R'_4(x), \neg S(x), \neg U(x). \end{array}$$

Then P_7 has 5 distinct stratifications, namely,

$$\begin{array}{l} \{r_1\}, \{r_2\}, \{r_3\}, \{r_4\} \\ \{r_2\}, \{r_1\}, \{r_3\}, \{r_4\} \\ \{r_2\}, \{r_3\}, \{r_1\}, \{r_4\} \\ \{r_1, r_2\}, \{r_3\}, \{r_4\} \\ \{r_2\}, \{r_1, r_3\}, \{r_4\}. \end{array}$$

$P_2 = \{p \leftarrow \neg q, q \leftarrow \neg p\}$
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

$C_1, ..., C_n$ 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 $C_{i_1}, ..., C_{i_n}$

this provides a stratification

Stratification : semantics

P a program with stratification $\sigma = P^1, ..., P^n$ and $\mathbf{I}$ and instance

$\mathbf{I}_0 = \mathbf{I}$

$\mathbf{I}_i = \mathbf{I}_{i-1} \cup P^i(\mathbf{I}_{i-1}|edb(P^i))$

where P^i is the semipositive semantics

$\mathbf{I}_n$ is denoted $\sigma(\mathbf{I})$

Result : independent of the choice of a stratification

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

Result : P stratified datalog$^{\neg}$ and $\mathbf{I}$

1. $P^{strat}(\mathbf{I})$ is a minimal model of Σ_P whose restriction to $edb(P)$ equals $\mathbf{I}$.
2. $P^{strat}(\mathbf{I})$ is a minimal fixpoint of T_P whose restriction to $edb(P)$ equals $\mathbf{I}$.
3. $P^{strat}(\mathbf{I})$ is a "supported" model of P relative to $\mathbf{I}$ ($\mathbf{J} \subseteq T_P(J) \cup \mathbf{I}$)

limited power

The well-founded semantics

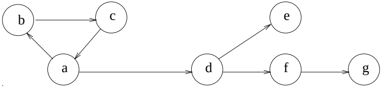
accommodate incompleteness

3-valued instances : true, false, unknown

example : two players game

input **K** with relation *moves* :

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



each player can move the position following a move edge

a player looses if he/she has no possible move

Game - ciontinued

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_{win} :

$$win(x) \quad \leftarrow \quad moves(x,y), \neg win(y)$$

(non stratifiable)

"3-valued model" **J** of P_{win}, that agrees with **K** on *moves*

true	$win(d), win(f)$
false	$win(e), win(g)$
unknown	$win(a), win(b), win(c)$.

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 $A \in \mathbf{B}(P)$, $\mathbf{I}(A) \leq \mathbf{J}(A)$

(equivalently, $\mathbf{I}^1 \subseteq \mathbf{J}^1$ and $\mathbf{I}^0 \supseteq \mathbf{J}^0$)

Truth value of boolean combination of facts

$$\begin{aligned}
\hat{\mathbf{I}}(\beta \wedge \gamma) &= \min\{\hat{\mathbf{I}}(\beta), \hat{\mathbf{I}}(\gamma)\} \\
\hat{\mathbf{I}}(\beta \vee \gamma) &= \max\{\hat{\mathbf{I}}(\beta), \hat{\mathbf{I}}(\gamma)\} \\
\hat{\mathbf{I}}(\neg\beta) &= 1 - \hat{\mathbf{I}}(\beta) \\
\hat{\mathbf{I}}(\beta \leftarrow \gamma) &= 1 \text{ if } \hat{\mathbf{I}}(\gamma) \leq \hat{\mathbf{I}}(\beta), \text{ and 0 otherwise.}
\end{aligned}$$

careful

$p \leftarrow q$ and $p \vee \neg q$: possibly different

3-valued instances - end

$\mathbf{I}$ *satisfies* α if $\hat{\mathbf{I}}(\alpha) = 1$

win example

$$\begin{array}{lcl} win(a) & \leftarrow & moves(a,d), \neg win(d) \\ win(a) & \leftarrow & moves(a,b), \neg win(b) \end{array}$$

first is true for $\mathbf{J}$
$\hat{\mathbf{J}}(\neg win(d)) = 0$, $\hat{\mathbf{J}}(moves(a,d)) = 1$, $\hat{\mathbf{J}}(win(a)) = 1/2$,
$1/2 \geq 0$.

second is true
$\hat{\mathbf{J}}(\neg win(b)) = 1/2$, $\hat{\mathbf{J}}(moves(a,b)) = 1$, $\hat{\mathbf{J}}(win(a)) = 1/2$,
$1/2 \geq 1/2$
$\hat{\mathbf{J}}(win(a) \vee \neg(moves(a,b) \wedge \neg win(b))) = 1/2$

3-valued minimal model for (extended) datalog

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

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

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

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

$$
\begin{array}{lcl}
\mathit{3\text{-}T_P}(\{\neg p, \neg q, \neg r, \neg s\}) & = & \{\neg q, r, \neg s\} \\
\mathit{3\text{-}T_P}(\{\neg q, r, \neg s\}) & = & \{r, \neg s\} \\
\mathit{3\text{-}T_P}(\{r, \neg s\}) & = & \{r\} \\
\mathit{3\text{-}T_P}(\{r\}) & = & \{r\}
\end{array}
$$

Result - 3-extended datalog programs

P 3-extended datalog program

1. $\mathit{3}\text{-}T_P$ is monotonic and the sequence $\{\mathit{3}\text{-}T_P^i(\perp)\}_{i>0}$ is increasing and converges to the least fixpoint of $\mathit{3}\text{-}T_P$
2. P has a **unique** minimal 3-valued model that equals the least fixpoint of $\mathit{3}\text{-}T_P$

minimal is w.r.t. $\prec$

3-stable models of datalog$^\neg$

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

P' ground version of P given $\mathbf{I}$

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

this is an extended datalog program

We denote its minimal model : $conseq_P(\mathbf{I})$

A 3-valued instance $\mathbf{I}$ over $sch(P)$ is a *3-stable model* of P iff $conseq_P(\mathbf{I}) = \mathbf{I}$.

Example : stable model

$$P$$

$$\begin{array}{rcl} p & \leftarrow & \neg r \\ q & \leftarrow & \neg r, p \\ s & \leftarrow & \neg t \\ t & \leftarrow & q, \neg s \\ u & \leftarrow & \neg t, p, s. \end{array}$$

3 3-stable models

$$\begin{array}{rcl} \mathbf{I}_1 & = & \{p, q, t, \neg r, \neg s, \neg u\}, \\ \mathbf{I}_2 & = & \{p, q, s, \neg r, \neg t, \neg u\}, \\ \mathbf{I}_3 & = & \{p, q, \neg r\}. \end{array}$$

checking $\mathbf{I}_3$

checking $\mathbf{I}_3$: positivized program

$$\begin{array}{lcl} p & \leftarrow & 1 \\ q & \leftarrow & 1, p \\ s & \leftarrow & 1/2 \\ t & \leftarrow & q, 1/2 \\ u & \leftarrow & 1/2, p, s. \end{array}$$

$$\begin{array}{l} \perp = \{\neg p, \neg q, \neg r, \neg s, \neg t, \neg u\} \\ \textit{3-}T_{P'}(\perp) = \{p, \neg q, \neg r, \neg t, \neg u\} \\ (\textit{3-}T_{P'})^2(\perp) = \{p, q, \neg r, \neg t\} \\ (\textit{3-}T_{P'})^3(\perp) = (\textit{3-}T_{P'})^4(\perp) = \{p, q, \neg r\} \\ \mathit{conseq}_P(\mathbf{I}_3) = (\textit{3-}T_{P'})^3(\perp) = \mathbf{I}_3, \end{array}$$

each datalog$^\neg$ programs has at least one 3-stable model

P a datalog$^\neg$ program

The *well-founded semantics* of P $P^{wf}(\emptyset) =$
the 3-valued instance consisting of all positive and negative facts belonging to all 3-stable models of P

$P^{wf}(\mathbf{I}) = P_{\mathbf{I}}^{wf}(\emptyset)$

example, $P_{win}^{wf}(\mathbf{K}) = \mathbf{J}$

Fixpoint characterization

previous description of the well-founded semantics
effective but very inefficient
more efficient one : "alternating fixpoint"
idea :
sequence $\{\mathbf{I}_i\}_{i \geq 0}$ of 3-valued instances
alternate between underestimates and overestimates of the facts known in every 3-stable model of P
SEQUENCE

$$\begin{array}{lcll} \mathbf{I}_0 & = & \perp & \text{(all facts are false)} \\ \mathbf{I}_{i+1} & = & \mathit{conseq}_P(\mathbf{I}_i) & \end{array}$$

each $\mathbf{I}_i$ is a total instance
observe that conseq_P is antimonotonic,
$I \prec J$ implies $\mathit{conseq}_P(J) \prec \mathit{consq}_P(I)$
since $\perp \prec \mathbf{I}_1$ and $\perp \prec \mathbf{I}_2$,

$$\mathbf{I}_0 \prec \mathbf{I}_2 \ldots \prec \mathbf{I}_{2i} \prec \mathbf{I}_{2i+2} \prec \ldots \prec \mathbf{I}_{2i+1} \prec \mathbf{I}_{2i-1} \prec \ldots \prec \mathbf{I}_1$$

Fixpoint : examples

P :

$$
\begin{array}{lcl}
p & \leftarrow & \neg r \\
q & \leftarrow & \neg r, p \\
s & \leftarrow & \neg t \\
t & \leftarrow & q, \neg s \\
u & \leftarrow & \neg t, p, s.
\end{array}
$$

$$
\begin{array}{lcl}
\mathbf{I}_0 = \bot = \{\neg p, \neg q, \neg r, \neg s, \neg t, \neg u\} & & \\
\mathbf{I}_1 & = & \{p, q, \neg r, s, t, u\}, \\
\mathbf{I}_2 & = & \{p, q, \neg r, \neg s, \neg t, \neg u\}, \\
\mathbf{I}_3 & = & \{p, q, \neg r, s, t, u\}, \\
\mathbf{I}_4 & = & \{p, q, \neg r, \neg s, \neg t, \neg u\}.
\end{array}
$$

Fixpoint : examples

P_{win} and input $\mathbf{K}$

for $\mathbf{I}_0$, all *move* atoms are **false**

for each $j \geq 1$, $\mathbf{I}_j(moves) = \mathbf{K}(moves)$

$$\begin{array}{lcl} \mathbf{I}_1 & = & \{win(a), win(b), win(c), win(d), \neg win(e), win(f), \neg win(\\ \mathbf{I}_2 & = & \{\neg win(a), \neg win(b), \neg win(c), win(d), \neg win(e), win(f), \neg \\ \mathbf{I}_3 & = & \mathbf{I}_1 \\ \mathbf{I}_4 & = & \mathbf{I}_2 \end{array}$$

Fixpoint

$$\mathbf{I}_0 \prec \mathbf{I}_2 \ldots \prec \mathbf{I}_{2i} \prec \mathbf{I}_{2i+2} \prec \ldots \prec \mathbf{I}_{2i+1} \prec \mathbf{I}_{2i-1} \prec \ldots \prec \mathbf{I}_1$$

there are finitely many 3-valued instances for a given P

these two sequences converge

$\mathbf{I}_*$: limit of increasing $\{\mathbf{I}_{2i}\}_{i\geq 0}$

$\mathbf{I}^*$: limit of decreasing $\{\mathbf{I}_{2i+1}\}_{i\geq 0}$

$\mathbf{I}_* \prec \mathbf{I}^*$

em $\text{conseq}_P(\mathbf{I}_*) = \mathbf{I}^*$ and $conseq_P(\mathbf{I}^*) = \mathbf{I}_*$

$\mathbf{I}_*^*$: 3-valued instance with facts known in both

$$\mathbf{I}_*^*(A) = \begin{cases} 1 & \text{if } \mathbf{I}_*(A) = \mathbf{I}^*(A) = 1 \\ 0 & \text{if } \mathbf{I}_*(A) = \mathbf{I}^*(A) = 0 \text{ and} \\ 1/2 & \text{otherwise.} \end{cases}$$

Results

Theorem : $\mathbf{I}_*^* = P^{wf}(\emptyset)$

Theorem

P stratified datalog$^\neg$ program,

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

Example

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

$$\begin{array}{lcl} bad(x) & \leftarrow & G(y,x), \neg good(y) \\ answer(x) & \leftarrow & \neg bad(x) \end{array}$$

$$\begin{array}{lcl} \mathbf{K}(G) & = & \{\langle b,c\rangle, \langle c,b\rangle, \langle c,d\rangle, \langle a,d\rangle, \langle a,e\rangle\}, \text{ and} \\ \mathbf{K}(good) & = & \{\langle a\rangle\}. \end{array}$$

as usual, we add the facts to program as unit clause
$\mathbf{I}_0 = \bot$ (containing all negated atoms).
omitting facts in good and G

	bad	*answer*
$\mathbf{I}_0$	$\emptyset$	$\emptyset$
$\mathbf{I}_1$	$\{\neg a, b, c, d, e\}$	$\{a, b, c, d, e\}$
$\mathbf{I}_2$	$\{\neg a, b, c, d, \neg e\}$	$\{a, \neg b, \neg c, \neg d, \neg e\}$
$\mathbf{I}_3$	$\{\neg a, b, c, d, \neg e\}$	$\{a, \neg b, \neg c, \neg d, e\}$
$\mathbf{I}_4$	$\{\neg a, b, c, d, \neg e\}$	$\{a, \neg b, \neg c, \neg d, e\}$

$\mathbf{I}_*^* = \mathbf{I}_* = \mathbf{I}^* = \mathbf{I}_3 = \mathbf{I}_4$

Merci