NegativeWrites Theory:
Handling Arithmetic Operations in Goal Attributes

Approach :

1. Given IDB ***I*** defined by rule set with minimum cardinality 2 in which at least one rule, denoted ***R***, possesses a goal attribute contains an arithmetic operation.
2. For all rules defining ***I***, rewrite the rules to exhibit a uniform goal attribute variable schema, except for appearances of the universal variables in the body such that the variables also appear in head arithmetic expressions.
3. Decompose ***R*** into a series of separate rules.
4. Apply NegativeWrites algorithm to the rules defining ***I*** after establishing a uniform attribute variable list across the defining rules.

Example :

P :

otherstuff(1,2) ;
b(1,2) ;
d(2) ;
e(2) ;
f(10,10) ;

stuff(X,Y) :- otherstuff(X,Y), notin a(X,Y) ;
a(X,Y+1) :- b(X,Y), c(Y) ;
c(X) :- d(X), e(X) ;

Rewrite 1 :

Remove arith ops from heads of rule targeted for NegativeWrites if goal has multiple rule definitions.

a(X,Y+1) :- b(X,Y), c(Y) ;

≡

{
a(A0,A1) :- a_b(A0,A1), a_c(A1) ;
a_b(X,Y+1) :- b(X,Y) ;
a_c(Y+1) :- c(Y) ;
}

Rewrite 3 : Via NegativeWrites

not_a(A0,A1) :- ¬a_b(A0,A1), ¬f(A0,A1), dom_stuff_att0(A0), dom_stuff_att1(A1)
not_a(A0,A1) :- ¬a_c(A0,A1), ¬f(A0,A1), dom_stuff_att0(A0), dom_stuff_att1(A1)

Rewrite 2 : B/c a_b, a_c, ***and*** c ***are IDBs.***

not_a_b(A0,A1+1) :- dom_not_a_att0(A0), dom_not_a_att1(A1), ¬b(A0,A1) ;

not_a_c(A0+1) :- dom_not_a_att1(A0), not_c(A0) ;

not_c(A0) :- dom_not_a_c_att0(A0), ¬d(A0) ;
not_c(A0) :- dom_not_a_c_att0(A0), ¬e(A0) ;

*Observe the rule definitions for ***not_a_b*** and ***not_a_c*** do not need arithmetic op rewrites because ***not_a_b*** and ***not_a_c*** are defined only by single rules across the entire program. Therefore, rewrites from cross-rule schema uniformity are not necessary.

P' :
otherstuff(1,2) ;
b(1,2) ;
d(2) ;
e(2) ;
f(10,10) ;

stuff(X,Y) :- otherstuff(X,Y), not_a(X,Y)
c(X) :- d(X), e(X)

// arithmetic op rewrite
a(A0,A1) :- a_b(A0,A1), a_c(A1) ;
a_b(X,Y+1) :- b(X,Y) ;
a_c(Y+1) :- c(Y) ;

// negated a rewrite
not_a(A0,A1) :- not_a_b(A0,A1), ¬f(A0,A1), dom_stuff_att0(A0), dom_stuff_att1(A1)
not_a(A0,A1) :- not_a_c(A0,A1), ¬f(A0,A1), dom_stuff_att0(A0), dom_stuff_att1(A1)

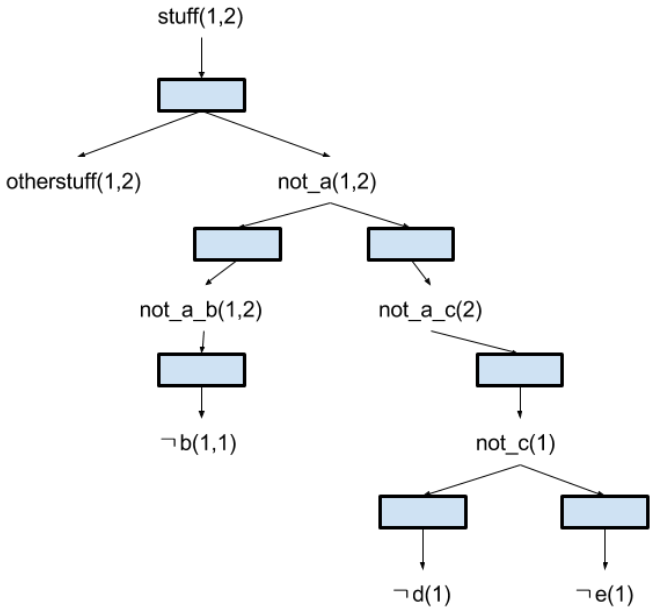
// negated ***a_b*** and ***a_c*** rewrites
not_a_b(A0,A1+1) :- dom_not_a_att0(A0), dom_not_a_att1(A1), not_b(A0,A1) ;
not_a_c(A0+1) :- dom_not_a_att1(A0), not_c(A0) ;

// negated ***b*** rewrite
not_b(A0,A1) :- dom_not_a_b_att0(A0), dom_not_a_b_att1(A1), ¬b(A0,A1) ;

// negated ***c*** rewrite
not_c(A0) :- dom_not_a_c_att0(A0), ¬d(A0) ;
not_c(A0) :- dom_not_a_c_att0(A0), ¬e(A0) ;

(domains from post-processing of evaluation not included for brevity)

Provenance graph:



//EOF