Advanced Data Modeling

Summer Semester 2008
- Exercises IV -

T*o be handed in before **2008-05-26, 23:59** via e-mail to sschenk@uni-koblenz.de, subject line: [ADM] ...*

1) Instances, Variants and Substitutions

1. Suppose Θ_1 and Θ_2 are substitutions and there exist substitutions σ_1 and σ_2, such that $\Theta_1 = \Theta_2 \ \sigma_1$ and $\Theta_2 = \Theta_1 \ \sigma_2$. Show that there exists a variable-pure substitution γ, such that $\Theta_1 = \Theta_2\gamma$.

   From $\Theta_1 = \Theta_2 \ \sigma_1$ and $\Theta_2 = \Theta_1 \ \sigma_2$ follows that $\Theta_1{=}\Theta_1\sigma_2\sigma_1$. Let γ'be $\sigma_1\sigma_2$ with the domain restricted to var(Θ_1). Obvisously γ' is variable pure. From $\rho 1\rho_2$ variable pure follows ρ_1 and $\rho 2$ are also variable pure. Hence, there must be a variable pure restriction γ of σ_1 to the domain var($\Theta 2$).

2. Which of the following clauses are Instances or Variants of each other?
   1. p(x, y, z) :- q(x, y), r(f(z))
   2. p(x, b, f(z)) :- q(x, b), r(f(f(z)))
   3. p(v, w, f(z)) :- q(v, b), r(f(f(z)))
   4. p(z, w, v) :- q(z, w), r(f(v))
   5. p(f(x), y, f(z)) :- q(f(x), y), r(f(f(z)))
   6. p(f(x), y, z) :- q(f(x), y), r(f(z))

Instance / variant of	1	2	3	4	5	6
1	v	-	-	v	-	-
2	i	v	i	i	-	-
3	i	i	v	i	-	-
4	v	-	-	v	-	-
5	i	i	-	i	v	i
6	i	-	-	i	-	v

2) The following Lemma shows that we only need to deal with Herbrand interpretations in order to find a model for any logic program:

Let C be a set of clauses and Σ be any signature containing all symbols used in C. The grounding of C with respect to Σ, denoted C* is the set of all ground instances of the signature Σ of clauses in C. Let I be an Herbrand interpretation and C be a set of clauses.

Prove that I |= C if and only if I |= C* .

a) I |= C* → I |= C .
Assume I |= C*. Without limiting generality we assume that for any two clauses C_1 and C_2 in C, the sets of variables used in C_1 and C_2 are disjoint. Let σ be any ground substitution over Σ. Then $C\sigma \subseteq C^*$. Hence, it is clear that I |= $C\sigma$. As we can choose any σ, it follows that I |= C.

a) I |= C → I |= C* .
Assume I |= C, but not I |= C*. Without limiting generality we assume that for any two clauses C_1 and C_2 in C, the sets of variables used in C_1 and C_2 are disjoint. Let σ be any ground substitution over Σ. Then $C\sigma \subseteq C^*$. It is clear that I |= $C\sigma$. As C* is the set of all ground instances of the signature Σ of clauses in C, there must be some σ for every clause C_1 in C*, such that $C_1 = C\sigma$. Then, however, I |= C_1, which is in conflict with our assumption.

3) Program Completion

1. Let the definition of a predicate symbol p be

   p(y) :- q(y), not r(a,y).
   p(f(z)) :- not q(z).
   p(b).

   Give a completion of p.

   1. p(X) :- (X = y), q(y), not r(a,y).
      p(X) :- (X = f(z)), not q(z).
      p(X) :- (X = b).
   2. p(X) :- ∃y (X = y), q(y), not r(a,y).
      p(X) :- ∃z (X = f(z)), not q(z).
      p(X) :- (X = b).
   3. p(X) :- ∃y (X = y), q(y), not r(a,y) ∨ ∃z (X = f(z)), not q(z) ∨ (X = b).
   4. p(X) = ∃y (X = y), q(y), not r(a,y) ∨ ∃z (X = f(z)), not q(z) ∨ (X = b).

2. Let P be a normal program and comp(P) it's completion. Prove that P is a logical consequence of comp(P). Hint: P is a logical consequence of comp(P) if
   I |= comp(P) → I |= P

   Let I be a model of comp(P). P is a logical consequence of comp(P) if I |= comp(P) → I |= P . As I models comp(P), every clause C in comp(P) is true in I. Now we apply the completion backwards:

   Replace every
   $A = G_1 \lor ... \lor G_n$
   by
   $A \rightarrow G_1 \lor ... \lor G_n$.
   $G_1 \lor ... \lor G_n \rightarrow A$.

   Clearly, I is also a model for the resulting program.
   As for all clauses C in a program P I |= P → I |=C, in the following we ignore
   $A \rightarrow G_1 \lor ... \lor G_n$.

   Then replace every
   $G_1 \lor ... \lor G_n \rightarrow A$.
   by
   $G_1 \rightarrow A$.
   ...
   $G_{n.} \rightarrow A$.

   Clearly, I is also a model for the resulting program P'.
   Each G_i in comp(P) results from a clause in P. I only contains ground instances of clauses in P. Now let G be the clause in P, G_i results from. As I is a model for P', all equalities and existentials in G_i must hold. We obtain a ground program P'' by removing all equalities and existentials from P'. As P'' is more general than P', obviously I |= P''. As the last step reverses the first two steps of the completion, P'' is a ground instantiation of P, hence I |= P.