Foundations of Databases

Exercises – Recursion in Relational Algebra and Calculus

March 15, 2005

1. Write a While$^+$ program which computes all nodes in a directed graph which do not lie on a directed cycle of even length. Do the same for undirected graphs.

2. Show that the variants of While, While$^+$, where instead of *"while change do"*, we have

   - *"while* $Expr \neq \emptyset$ *do"* in While
   - *"while* $Expr_1 \neq Expr_2$ *do"* in While$^+$

   have the same expressiveness as the original languages.

3. As for the proof of While$^+$ = CALC+μ^+, discuss how a While$^+$ program can be expressed by a CALC+μ^+ query.

4. Recall the definition of $\mu_T(\phi(T))$. Exhibit a formula $\phi(T)$ which has a unique minimal fixpoint on all inputs, and $\mu_T(\phi(T))$ terminates on all inputs but does not evaluate to the minimal fixpoint on any of them.

5. Express the queries in Ex. (1) in CALC+μ^+.

6. Simultaneous Induction: In the language CALC-μ, the fixpoint operator is applied to one relation at a time, i.e., we have $\mu_T(\phi(T))$ for some relation variable T. An extension of the language is that *several* relations can be simultaneously considered (similar as with the $\mathbf{T}_P$ operator, which is applied to all *idb* relations simultaneously). Thus, a generalized fixpoint operator could be of form

$$\mu_{T_1,\ldots,T_k}(\pi_1(T_1,\ldots,T_k),\ldots,\pi_k(T_1,\ldots,T_k))(T_i)$$

which "accesses" the result for T_i when the fixpoint is built for $T_1, \ldots, T_k$ using the formulas $\phi_1, \ldots, \phi_k$.

For example , we could have $k = 2$ and $\phi_1(T_1, T_2)(x, y) = G(x, y) \vee T_1(x, y) \wedge T_2(x)$, $\phi_2(x) = \exists y G(x, y) \wedge T(y) \leftrightarrow T_2(x)$.

(a) Write the formula which states that a belongs to the fixpoint for relation T_2

(b) How are ordinary fixpoint operators expressed as special case?

(c) Write a semantics definition for the generalized operator.

(d) Is the language with the generalized fixpoint operator more expressive?

7. Describe a While$^+$ program P and an input database $\mathbf{I}$ for it, such that computing $P(\mathbf{I})$ using iteration requires exponential time (in the size of P and $\mathbf{I}$). Do the same for CALC-μ^+.