Towards Determining Variable Overlap in Recursive Rule Expansions

David A. Briggs

Technical Report 87-33

April 13, 1987

Towards Determining Variable Overlap in Recursive Rule Expansions

David A. Briggs

April 13, 1987

Abstract

One approach to the evaluation of queries against relations defined by a recursive rule is to derive the series of rules that represent the resolution of multiple instances of the original rule, and apply each of these derived rules to the base set of tuples that are in the relation "non-recursively". We call these derived rules ***rule expansions.*** A significant source of economy in the evaluation is the recognition of repeating patterns of variable overlap in the literals of the rule expansions. In this paper we give conditions under which the variables of a rule expansion will be equated in terms of paths of the α-graph of Ioannidis. This characterization should prove useful in developing strategies for query evaluation.

The research community has recently devoted much attention to the efficient processing of queries against a relational data base that involve operations beyond those provided by the relational algebra. Although there has been some variety in the additional operations proposed, a common feature is the capability to pose queries that require taking the fixed point of an expression, so-called "recursive" queries, and a common linguistic vehicle for the expression of such queries are Horn clauses such as one finds in Prolog. We assume in the reader some familiarity with logic programming and suggest [Lloy84] for newcomers to the field. The goal is to find efficient techniques for the evaluation of queries of this sort, and [Banc86] surveys many of the methods that have been offered. A major focus of attention has been the identification of methods that exploit any selection conditions on the computed relation to avoid unnecessary computational effort, and the work of Henschen and Naqvi as presented in [Hens84] suggests a sound formal basis for obtaining significant economies. As detailed in [Brig87], the correctness of the Henschen-Naqvi technique rests on the pattern of variable overlap in successive expansions of the recursive rule. Repeated patterns of variable overlap among the literals of successive rule expansions suggest blocking the query into functional components and also serve in determining how constants provided as selection conditions propagate in the expansion. Recognizing the pattern of overlap is thus crucially important for an intelligent processing strategy.

Ioannidis, in [Ioan86], attacks another aspect of processing recursive queries, specifically, the detection that the number of rule expansions required to compute the recursive relation is bounded independently of the contents of the underlying base relations. He employs a representation of the recursive rule called the α-graph, and gives a graph characterization for the boundedness property for a restricted class of rules. In this paper we show how the α-graph representation can be used to identify patterns of variable overlap in rule expansions for a very general class of rules. We demonstrate that variable arguments in a rule expansion will be equated if and only if a certain kind of path connecting the two variables is present in the graph. It is hoped that this graph characterization will provide leverage on the larger goal of generating an efficient iterative program for queries against the recursive relation.

We deal with linear recursive rules, that is rules that contain a single instance of the recursive predicate in the tail of the rule, but we allow repeated variables in either the recursive consequent or recursive antecedent. The treatment presented here does not allow constants in either, but we anticipate no difficulty in their incorporation. Although in general the tail of the rule may contain many literals aside from the recursive antecedent, we will for simplicity assume they have all been joined into a single non-recursive base relation. In practice, this would often be a poor strategy, and we make the assumption for notational convenience only. Indeed, one of the hoped for benefits of this analysis is an identification of which of the base relation literals of the antecendent should be joined, and when. Under these assumptions, a general rule instance is given by the expression

$$P(X^0_{j_1}, \ldots, X^0_{j_n}) \Leftarrow P(X^0_{i_1}, \ldots, X^0_{i_n}) \wedge Q(X^0_1, \ldots, X^0_m)$$

where P is the n-ary recursive relation defined by the rule and Q is the join of the base literals. All variables occurring in the rule are of the form X^0_l, where $l \epsilon \{1, \ldots, p\}$ with p distinct variables in the rule. The subscripts are chosen so that the first m are used by the base relation Q, but the subscripts of the arguments of the recursive predicate P as it appears in the consequent and antecedent may be any value from 1 to p. The subscripts are used to distinguish variables within the rule. The superscripts will be used to distinguish variables in expansions of the rule in a manner to be described shortly.

The rule describes a way to derive tuples in the recursive relation from tuples "already" in it, or, alternatively, an inclusion relation that the recursive relation must satisfy. Under the usual semantics, the recursive relation is taken to be the minimal set, under inclusion, of all sets that satisfy the rule's inclusion relation. For this solution to be non-empty, there must be some tuples explicitly placed in P without recourse to the rule, and we will denote the collection of these tuples by P_0. The solution for P will be all the tuples that are either in P_0 or can be derived from tuples in P_0 by repeated applications of the rule. If all the base relations are finite and, as we have implicitly assumed all along, the arguments to the literals are only variables or constants, not function complexes, then the solution for P will be finite.

One might compute P by repeatedly applying the rule to tuples derived from the previous iteration, with P_0 supplying the tuples for the first iteration. An equivalent alternative is to derive a series of rules by resolving in additional instances of the original rule, and apply the rules so derived to P_0, and the analysis presented here bears on that approach.

We would like, therefore, a formal characterization of the rules that would be generated by repeatedly resolving in an instance of the original rule. To obtain such a characterization, we must describe the effect of resolution on the variables of the rules by defining the the substitutions that would take place. At each stage we will resolve on the recursive antecedent of the current rule and the recursive consequent of the original rule. Since variables may be repeated in either, we may be forced to apply a substitution to both the current rule and the instance of the original rule being brought in. Before presenting the argument for the general case, we will look at a specific example. Consider the rule

$$\begin{aligned} P(X^0_3, X^0_4, X^0_5, X^0_6, X^0_6, X^0_7) \Leftarrow\ & P(X^0_1, X^0_2, X^0_2, X^0_3, X^0_4, X^0_5) \wedge \\ & Q(X^0_1, X^0_2, X^0_3, X^0_4, X^0_5, X^0_6, X^0_7) \end{aligned}$$

Suppose we resolve two instances of this rule. We separate the variables of the second instance, making all of the superscripts 1 instead of 0, yielding

$$\begin{aligned} P(X^1_3, X^1_4, X^1_5, X^1_6, X^1_6, X^1_7) \Leftarrow\ & P(X^1_1, X^1_2, X^1_2, X^1_3, X^1_4, X^1_5) \wedge \\ & Q(X^1_1, X^1_2, X^1_3, X^1_4, X^1_5, X^1_6, X^1_7) \end{aligned}$$

We will resolve on the recursive antecedent of the first instance and the recursive consequent of the second, so we must unify the literals

$$\begin{gathered} P(X^0_1, X^0_2, X^0_2, X^0_3, X^0_4, X^0_5) \\ P(X^1_3, X^1_4, X^1_5, X^1_6, X^1_6, X^1_7) \end{gathered}$$

We would prefer to leave the literals of the first rule instance as they are, but the repeated occurrence of X^1_6 forces us to equate X^0_3 and X^0_4 wherever they occur. We could replace both of them with some variable that doesn't occur elsewhere in the rule, or replace one with the other, but they must be equated. We will replace X^0_4 with X^0_3, obtaining

$$\begin{aligned} P(X^0_3, X^0_3, X^0_5, X^0_6, X^0_6, X^0_7) \Leftarrow\ & P(X^0_1, X^0_2, X^0_2, X^0_3, X^0_3, X^0_5) \wedge \\ & Q(X^0_1, X^0_2, X^0_3, X^0_3, X^0_5, X^0_6, X^0_7) \end{aligned}$$

We can now define a substitution on the variables of the second rule instance to complete the unification

$$\begin{aligned} X^1_3 &\mapsto X^0_1 \\ X^1_4 &\mapsto X^0_2 \\ X^1_5 &\mapsto X^0_2 \\ X^1_6 &\mapsto X^0_3 \\ X^1_7 &\mapsto X^0_5 \end{aligned}$$

Applying that substitution and completing the resolution, we arrive at the rule

$$\begin{aligned} P(X_3^0, X_3^0, X_5^0, X_6^0, X_6^0, X_7^0) \Leftarrow\ & P(X_1^1, X_2^1, X_2^1, X_1^0, X_2^0, X_2^0)\wedge \\ & Q_1(X_1^1, X_2^1, X_1^0, X_2^0, X_2^0, X_3^0, X_5^0)\wedge \\ & Q_0(X_1^0, X_2^0, X_3^0, X_3^0, X_5^0, X_6^0, X_7^0) \end{aligned}$$

where the Q's have been subscripted to indicate the rule instance of their origin.

If we resolve in an additional separated instance of the rule, we must unify the literals

$$\begin{gathered} P(X_1^1, X_2^1, X_2^1, X_1^0, X_2^0, X_2^0) \\ P(X_3^2, X_4^2, X_5^2, X_6^2, X_6^2, X_7^2) \end{gathered}$$

Again, we are forced to equate X_1^0 and X_2^0 in the rule representing the resolution of two instances of the original rule. We will effect this by replacing X_2^0 with X_1^0. Once that has been done, we can define a substitution to apply to the separated rule instance and complete the resolution, generating a rule representing the resolution of three instances of the original rule

$$\begin{aligned} P(X_3^0, X_3^0, X_5^0, X_6^0, X_6^0, X_7^0) \Leftarrow\ & P(X_1^2, X_2^2, X_2^2, X_1^1, X_2^1, X_2^1)\wedge \\ & Q_2(X_1^2, X_2^2, X_1^1, X_2^1, X_2^1, X_1^0, X_1^0)\wedge \\ & Q_1(X_1^1, X_2^1, X_1^0, X_1^0, X_1^0, X_3^0, X_5^0)\wedge \\ & Q_0(X_1^0, X_1^0, X_3^0, X_3^0, X_5^0, X_6^0, X_7^0) \end{aligned}$$

Note that the instances of the Q predicate from the previous rule were both modified by the equivalencing substitution.

If we were to continue in this manner, at the rth stage we would define two substitutions : σ_r, a substitution to apply to the rule generated thus far, and γ_r, a substitution to apply to the instance of the original rule being resolved with the rule obtained so far. In the description that follows, we incorporate into the γ substitutions the separation of the original rule instance that we have accomplished by incrementing the subscripts, so that each γ_r will be a substitution applied to the variables of the original rule. The substitution γ_r will be used to bring in the $r+1$st instance of the original rule. Clearly, γ_0 can be the identity substitution. Given γ_r, define the binary relation R_{r+1} on $\{1, \ldots, n\}$, the argument positions of the recursive predicate, as

$$k\ R_{r+1}\ l \Longleftrightarrow j_k = j_l \vee \gamma_r(X_{i_k}^0) = \gamma_r(X_{i_l}^0)$$

Note that R_{r+1} is reflexive and symmetric. We next define an equivalence relation on $\{1,\ldots,n\}$, $\cong_{r+1}$, as the transitive closure of R_{r+1}. Using the equivalence relation, define a map $des_{r+1} : \{1,\ldots,n\} \longrightarrow \{1,\ldots,n\}$ that sends each member of an equivalence class to a single member of its class, the "designate" for the class. Next, define the equivalencing substitution, σ_{r+1} by

$$\gamma_r(X^0_{i_k}) \longmapsto \gamma_r(X^0_{i_{des_{r+1}(k)}})$$

We are now in a position to define γ_{r+1}, the substitution to be applied to the original rule to resolve it with the current rule.

$$\gamma_{r+1}(X^0_k) = \begin{cases} \gamma_r(X^0_{i_{des_{r+1}(l)}}) & \text{if } k = j_l \\ X^{r+1}_k & \text{else} \end{cases}$$

Before exhibiting the formal representation of the rule expansions, we provide some explanation for the preceding welter of formalism. We view the resolution of many instances of the base rule as an iterated resolution of rule instances. Each time we bring in another instance of the rule, we will unify the recursive antecedent of the clause obtained so far, $P(\nu_1,\ldots,\nu_n)$,with the recursive consequent of the rule instance, $P(X^0_{j_1},\ldots,X^0_{j_n})$. Since we are allowing the possibility of repeated variables in either the consequent or the antecedent, we may be forced to apply a substitution to both the clause obtained so far and the rule instance. We effect the resolution by first forcing any required equivalencing in the clause obtained so far with the σ substitutions, and use the γ substitution on the rule instance. In arriving at the correct specification of the σ substitution, consider the literals being unified. At the $r+1$st iteration, we will be unifying $\gamma_r(P(X^0_{i_1},\ldots,X^0_{i_n}))$, the antecedent of the last rule instance brought into the burgeoning rule, and $P(X^0_{j_1},\ldots,X^0_{j_n})$, so we need to determine any equivalencing of variables in the clause obtained so far forced by this unification and define σ_{r+1} accordingly. The equivalencing will only be of variables occuring in the $\gamma_r(P(X^0_{i_1},\ldots,X^0_{i_n}))$ literal, and the $\cong_{r+1}$ relation is meant to identify precisely those argument positions that will be forced equivalent. The somewhat obscure definition of $\cong_{r+1}$ is needed to correctly handle a situation such as $P(X,X,Y,Y,Z,Z)$ and $P(T,U,U,V,V,W)$, where unification will force all of X, Y, and Z to be equated. Once the equivalence classes have been identified, any variable within the class may be chosen as the one that will "survive" the impending σ substitution, which must be applied not only to the literal of the unification pair, but the entire clause obtained so far. Applying σ_{r+1} to the literal we obtain

$$\sigma_{r+1}\gamma_r(P(X^0_{i_1},\ldots,X^0_{i_n})) = \gamma_r(P(X^0_{i_{dee_{r+1}(1)}},\ldots,X^0_{i_{dee_{r+1}(n)}}))$$

In determining the γ_{r+1} substitution to apply to the rule, we map variables occurring in the head of the rule to their correspondent in the literal above. This map is well-defined even with repeated variables in the consequent, because of the equivalencing operation. Variables not occurring in the head are given a superscript to distinguish them from any variables occurring in the clause obtained so far.

We are now in a position to express $RULE_i$, the rule expansion obtained by resolving $i+1$ instances of the original rule.

$$\begin{aligned} RULE_i &= (\prod_{l'=1}^{i}\sigma_{l'})P(X^0_{j_1},\ldots,X^0_{j_n}) \Leftarrow \\ &\gamma_i P(X^0_{i_1},\ldots,X^0_{i_n}) \wedge \bigwedge_{l=0}^{i}(\prod_{l'=l+1}^{i}\sigma_{l'})\gamma_l Q(X^0_1,\ldots,X^0_m) \end{aligned}$$

Since substitutions do not in general commute, it must be understood that the product of the $\sigma_{l'}$'s are listed in decreasing order of subscripts.

We next define the α-graph of Ioannidis, originally given in [Ioan86]. Given a rule

$$P(X^0_{j_1},\ldots,X^0_{j_n}) \Leftarrow P(X^0_{i_1},\ldots,X^0_{i_n}) \wedge Q(X^0_1,\ldots,X^0_m)$$

we define a graph whose nodes are the distinct variables of the rule. We have a directed edge in the graph from ν to μ if and only if there is some $k\epsilon\{1,\ldots,n\}$ such that $\nu = X^0_{i_k}$ and $\mu = X^0_{j_k}$, in words, if ν is a variable occurring in the recursive antecedent and μ occurs in the recursive consequent in the same argument position. We have an undirected edge between two variables ν and μ if and only if both occur in the same non-recursive literal of the tail of the rule. For example, the rule

$$P(X,X,Y,Z) \Leftarrow P(V,W,X,X) \wedge Q(V,Y) \wedge R(W,Z)$$

would yield the graph of Figure 1. For our purposes the directed edges are

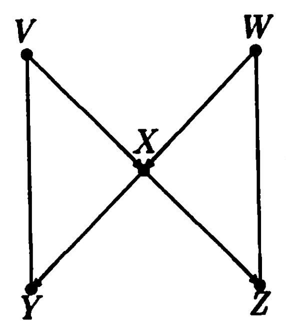

Figure 1:

the only ones of interest, so we eliminate all the undirected edges from the graph. Ioannidis terms the graph so restricted the *dynamic* α-graph.

We note that in a rule expansion, it is not the specific variable letters that occur in literals that matter, but rather the identification of all and only those argument positions that contain the same variable. This is just another way of saying that a renaming of the variables of the clause does not affect the meaning of the clause. We observe further that the variables in the literals of $RULE_i$ are all of the form $(\prod_{l'=l+1}^{i} \sigma_{l'})\gamma_l(\nu)$, where ν is a variable occurring in the original rule, and $l \epsilon \{0, \ldots, i\}$. It turns out that two such expressions for arbitrary variables ν and μ of the original rule will be equivalent if and only there is a certain kind of path linking the two variables in the dynamic α-graph. The paths we need are aberrant in that we permit traversing a directed edge in either the forward or reverse direction, but we keep track of the number of forward or reverse traversals we make in the path. One can think of such a path as mapping to a string of F's and R's, depending upon the direction of the traversal of an edge in the path. The result we claim is

The n-k-l Theorem. For all n, k, and l in **N**, and variables in the rule ν and μ,

$$\sigma_{n+k+l} \cdots \sigma_{n+1}\gamma_n(\nu) = \sigma_{n+k+l} \cdots \sigma_{n+k+1}\gamma_{n+k}(\mu)$$

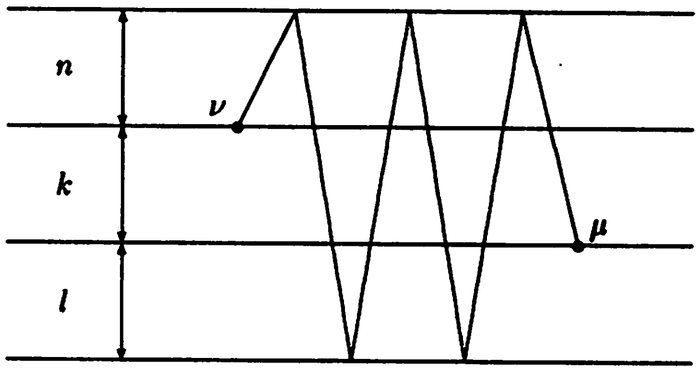

Figure 2:

if and only if there is a path in the dynamic α-graph from ν to μ such that

1. There are k more forward traversals in the path than there are reverse traversals.
2. If one keeps track of the number of forward and reverse traversals as one progresses through the path, there are never more than n more reverse traversals than forwards, and never more than $k + l$ more forward traversals than reverses.

Pictorially, if we arrange the nodes encountered in the path by levels, where each time we traverse an arc in the forward direction we drop a level, and each time we traverse an arc in the reverse direction we go up a level, the theorem states that the expressions of the equation are equal if and only if there is a path whose meanderings from ν are bounded above by n levels, below by $k + l$ levels, that ends at μ k levels below the start level (see Figure 2).

Proof($\Longleftarrow$). Note that a directed edge from a node v to a node w implies that for all $n \epsilon \mathrm{N}$, $\gamma_{n+1}(w) = \sigma_{n+1}\gamma_n(v)$. Starting with the expression $\sigma_{n+k+l}\cdots\sigma_{n+1}\gamma_n(\nu)$, we follow the path, peeling σ's out of the subscripts of the γ's as we move up levels, and trading them back in for a higher subscript as we move down. The bounds guarantee that we never exhaust our stock to trade as we go, and the overall difference in levels tells us that when we complete the path, we will be at the node for μ with a net loss of k σ's.

The proof in the other directions is more involved, and we need to climb up to it via a number of lemmas.

Lemma 1. For all $k, n \epsilon \mathbf{N}$, $n \geq k$, and for all variables ν, the superscripts of $\gamma_n(\nu)$ and $\sigma_{n+1} \cdots \sigma_{k+1}\gamma_k(\nu)$ are less than or equal to n.

Proof. An easy induction on n.

Lemma 2. For all $n \geq 0$, l and l', $\gamma_n(X^0_{i_{des_{n+1}(l)}}) = \gamma_n(X^0_{i_{des_{n+1}(l')}})$ implies $l \cong_{n+1} l'$.

Proof. Note that $l \cong_{n+1} des_{n+1}(l)$, for any l. The condition gives $des_{n+1}(l) \cong_{n+1} des_{n+1}(l')$, by the definition of R_{n+1}. The result then follows from transitivity of the equivalence relation $\cong_{n+1}$.

Note that $l \cong_{n+1} l'$ then implies that $des_{n+1}(l) = des_{n+1}(l')$, and thus that $\sigma_{n+1}\gamma_n(X^0_{i_l}) = \sigma_{n+1}\gamma_n(X^0_{i_{l'}})$.

Lemma 3. This is the special case of the theorem with $k, l = 0$. For all n, all variables X^0_r and X^0_s, if $\gamma_n(X^0_r) = \gamma_n(X^0_s)$ then there exists a path from X^0_r to X^0_s in the dynamic α-graph such that the number of forward arcs traversed equals the number of reverse arcs, and at any point in the path, the number of reverse arcs traversed is greater than or equal to the number of forward arcs (See Figure 3).

Proof. We induct on n. For $n = 0$, the variables must be the same, so the null path exists within the bound. Hypothesize the lemma true for n, and assume $\gamma_{n+1}(X^0_r) = \gamma_{n+1}(X^0_s)$. Now, if neither variable occurs in the recursive consequent, using the definition of γ_{n+1}, we must have $r = s$. The variables are thus the same, and the null path again suffices. We cannot have one variable occurring in the consequent and one not, for then the superscripts won't match (by lemma 1 and the definition of γ_{n+1}). The last

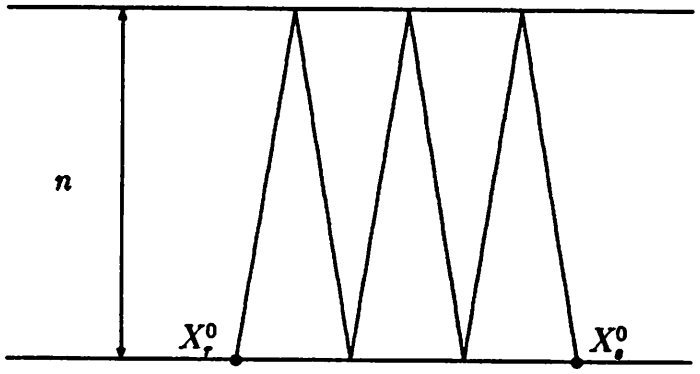

Figure 3:

case to consider is when both occur in the consequent. We would then have $r = j_t$ and $s = j_{t'}$ for some t and t', and by the definition of γ_{n+1},

$$\gamma_{n+1}(X^0_{j_t}) = \gamma_n(X^0_{i_{dec_{n+1}(t)}}) = \gamma_n(X^0_{i_{dec_{n+1}(t')}}) = \gamma_{n+1}(X^0_{j_{t'}})$$

By lemma 2, we have $t \cong_{n+1} t'$. If we can show a path from $X^0_{i_t}$ to $X^0_{i_{t'}}$ that doesn't drop more than one level, or rise more than n levels, we can paste it together with the directed arcs from $X^0_{i_t}$ to $X^0_{j_t}$ and from $X^0_{i_{t'}}$ to $X^0_{j_{t'}}$ and obtain the path we're looking for (see Figure 4). We have $t \cong_{n+1} t'$, so there exists a sequence $t = t_0, t_1, \ldots, t_k = t'$ such that for all $p \epsilon \{0, \ldots, k-1\}$, $t_p \; R_{n+1} \; t_{p+1}$, that is, either $j_{t_p} = j_{t_{p+1}}$ or $\gamma_n(X^0_{i_{t_p}}) = \gamma_n(X^0_{i_{t_{p+1}}})$. But this implies that for all $p \epsilon \{0, \ldots, k-1\}$, either there is a path from $X^0_{i_{t_p}}$ to $X^0_{i_{t_{p+1}}}$ that goes down one level and back up one level or, invoking the inductive hypothesis, that goes up no more than n levels and returns to the same level (see Figure 5). Combining these paths for each t_p of the sequence, we have the path we're after.

Lemma 4. The special case of the theorem for $k = 0$ and $l = 1$. For all n, and all variables X^0_r and X^0_s if $\sigma_{n+1}\gamma_n(X^0_r) = \sigma_{n+1}\gamma_n(X^0_s)$ then there exists a path in the dynamic α-graph from X^0_r to X^0_s such that the total number of forward traversals in the path equals the number of reverse traversals, and during the course of the path the count of reverse traversals less the

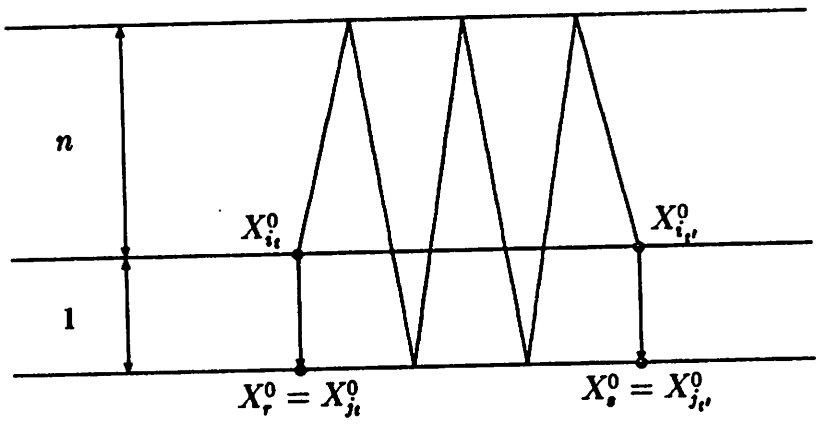

Figure 4:

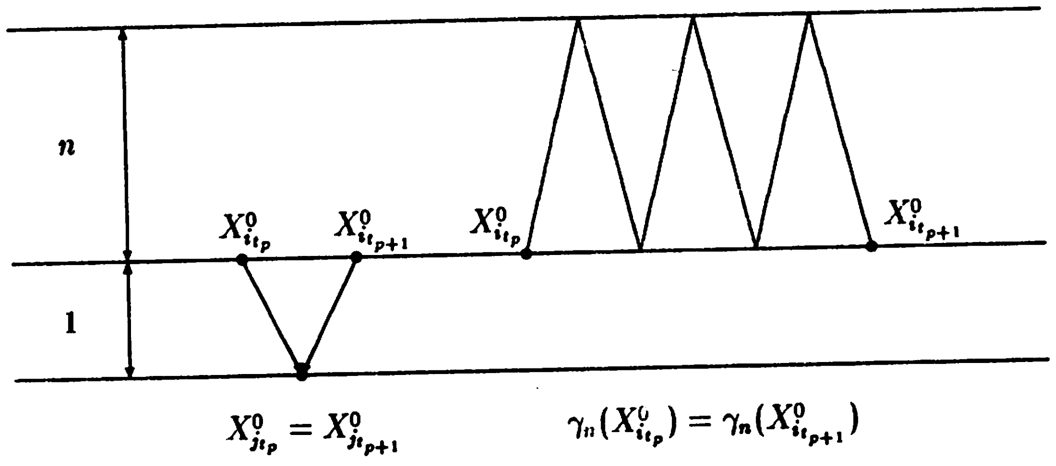
n
1
$X^0_{i_{t_p}}$
$X^0_{i_{t_{p+1}}}$
$X^0_{i_{t_p}}$
$X^0_{i_{t_{p+1}}}$
$X^0_{j_{t_p}} = X^0_{j_{t_{p+1}}}$
$\gamma_n(X^0_{i_{t_p}}) = \gamma_n(X^0_{i_{t_{p+1}}})$

Figure 5:

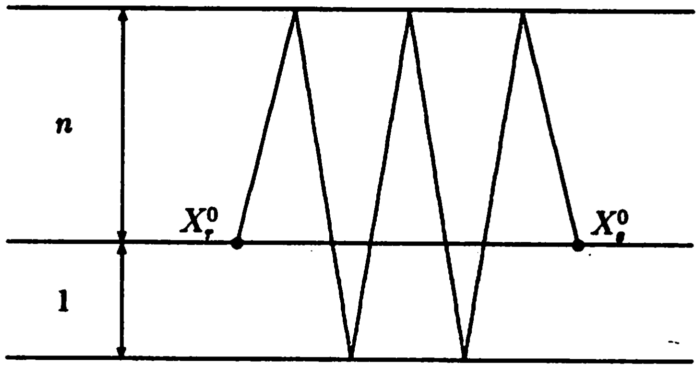

Figure 6:

count of forward traversals is bounded above by n and below by -1(see Figure 6).

Proof. Recalling the definition of the σ and γ substitutions, we have for any variable X_r^0

$$\sigma_{n+1}\gamma_n(X_r^0) = \begin{cases} \gamma_n(X_r^0) & \text{if } \gamma_n(X_r^0) \neq \gamma_n(X_{i_t}^0) \text{ for any } t \\ \gamma_n(X^0_{i_{des_{n+1}(t)}}) & \text{if } \gamma_n(X_r^0) = \gamma_n(X_{i_t}^0) \end{cases}$$

Considering the possibilities, we see that if the first alternative holds for both sides of the assumed equality, we can invoke lemma 2. If the first alternative holds for one, but not the other, we have a contradiction. If the second alternative holds for both, we have

$$\sigma_{n+1}\gamma_n(X_r^0) = \gamma_n(X^0_{i_{des_{n+1}(t)}}) = \gamma_n(X^0_{i_{des_{n+1}(t')}}) = \sigma_{n+1}\gamma_n(X_s^0)$$

for some t and t', with $t \cong_{n+1} t'$, by lemma 2. Using lemma 3, we can connect X_r^0 to $X_{i_t}^0$ and X_s^0 to $X_{i_{t'}}^0$, and connect $X_{i_t}^0$ to $X_{i_{t'}}^0$ as we did in the proof of lemma 3 (see Figure 7).

Lemma 5. The special case of the theorem with $l = 0$. For all n and k, if $\sigma_{n+k}\cdots\sigma_{n+1}\gamma_n(X_r^0) = \gamma_{n+k}(X_s^0)$,then there is a path from X_r^0 to X_s^0 that

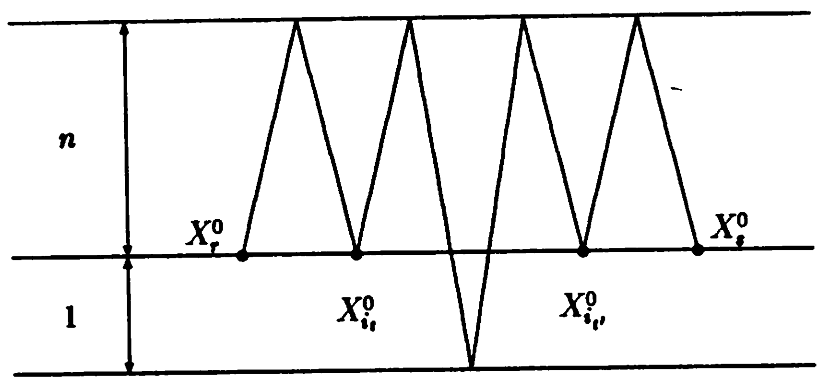

Figure 7:

has k more forward traversals than reverse traversals, with the number of forwards less the number of reverses never more than k and never less than $-n$.

The proof is by induction on k. For $k = 0$, it reduces to lemma 3. Assuming that the assertion is true for k, let

$$\sigma_{n+k+1} \cdots \sigma_{n+1} \gamma_n(X^0_r) = \gamma_{n+k+1}(X^0_s)$$

By definition,

$$\sigma_{n+k+1} \cdots \sigma_{n+1} \gamma_n(X^0_r) = \begin{cases} \sigma_{n+k} \cdots \sigma_{n+1} \gamma_n(X^0_r) & \text{if } \sigma_{n+k} \cdots \sigma_{n+1} \gamma_n(X^0_r) \neq \\ & \quad \gamma_{n+k}(X^0_{i_t}) \text{ for any } t \\ \gamma_{n+k}(X^0_{i_{dec_{n+k+1}(t)}}) & \text{if } \sigma_{n+k} \cdots \sigma_{n+1} \gamma_n(X^0_r) = \\ & \quad \gamma_{n+k}(X^0_{i_t}) \end{cases}$$

$$\gamma_{n+k+1}(X^0_s) = \begin{cases} X^{n+k+1}_s & \text{if } s \neq j_{t'} \text{ for any } t' \\ \gamma_{n+k}(X^0_{i_{dec_{n+k+1}(t')}}) & \text{if } s = j_{t'} \end{cases}$$

Any choice except the latter for both leads to a contradiction, so consider that situation. By the inductive hypothesis, we have a path from X^0_r to $X^0_{i_t}$

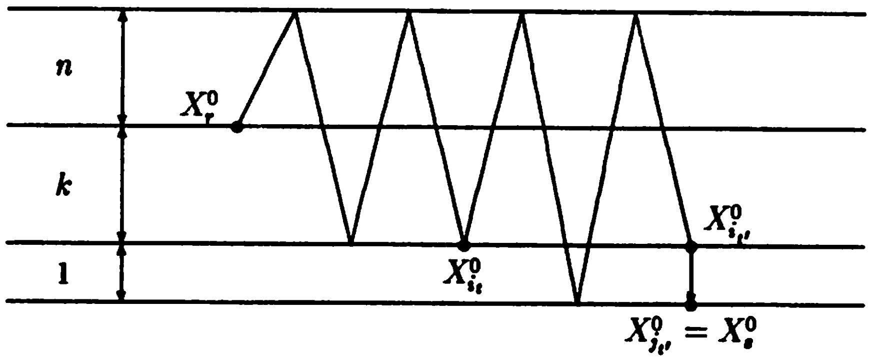

Figure 8:

that drops k levels, and stays within the bounds. Since

$$\gamma_{n+k}(X^0_{i_{des_{n+k+1}(t)}}) = \gamma_{n+k}(X^0_{i_{des_{n+k+1}(t')}}) \Rightarrow$$
$$\sigma_{n+k+1}\gamma_{n+k}(X^0_{i_t}) = \sigma_{n+k+1}\gamma_{n+k}(X^0_{i_{t'}})$$

we can invoke lemma 4 to get a path from $X^0_{i_t}$ to $X^0_{t'}$, and then drop down the arc to X^0_s (see Figure 8).

We can now turn to the only-if direction of the theorem.

The n-k-l Theorem.($\Longrightarrow$) For any n, k, and l,

$$\sigma_{n+k+l}\cdots\sigma_{n+1}\gamma_n(X^0_r) = \sigma_{n+k+l}\cdots\sigma_{n+k+1}\gamma_{n+k}(X^0_s)$$

implies the existence of a path from X^0_r to X^0_s in the dynamic α-graph such that

1. The number of forward traversals in the path is k more than the number of reverse traversals.

2. The difference of reverse traversals less forward traversals as one progresses through the path is bounded above by n and below by $-k-l$.

Proof. We induct on l. For $l = 0$, we use lemma 5. The statement of the premiss for $l + 1$ is

$$\sigma_{n+k+l+1} \cdots \sigma_{n+1}\gamma_n(X_r^0) = \sigma_{n+k+l+1} \cdots \sigma_{n+k+1}\gamma_{n+k}(X_s^0)$$

Opening up the definitions,

$$\sigma_{n+k+l+1} \cdots \sigma_{n+1}\gamma_n(X_r^0) = \begin{cases} \sigma_{n+k+l} \cdots \sigma_{n+1}\gamma_n(X_r^0) & \text{if } \sigma_{n+k+l} \cdots \sigma_{n+1}\gamma_n(X_r^0) \neq \\ & \gamma_{n+k+l}(X_{i_t}^0) \text{ for any } t \\ \gamma_{n+k+l}(X^0_{i_{des_{n+k+l+1}(t)}}) & \text{if } \sigma_{n+k+l} \cdots \sigma_{n+1}\gamma_n(X_r^0) = \\ & \gamma_{n+k+l}(X_{i_t}^0) \end{cases}$$

$$\sigma_{n+k+l+1} \cdots \sigma_{n+k+1}\gamma_{n+k}(X_s^0) = \begin{cases} \sigma_{n+k+l} \cdots \sigma_{n+k+1}\gamma_{n+k}(X_s^0) & \text{if } \sigma_{n+k+l} \cdots \sigma_{n+k+1}\gamma_{n+k}(X_s^0) \neq \\ & \gamma_{n+k+l}(X_{i_{t'}}^0) \text{ for any } t' \\ \gamma_{n+k+l}(X^0_{i_{des_{n+k+l+1}(t')}}) & \text{if } \sigma_{n+k+l} \cdots \sigma_{n+k+1}\gamma_{n+k}(X_s^0 = \\ & \gamma_{n+k+l}(X_{i_{t'}}^0) \end{cases}$$

If the first alternative holds for both, then we have the path by the inductive hypothesis. If the first alternative holds for one and the second for the other, we contradict the presumed equality. Finally, if the latter alternatives hold for both, we use lemma 5 to get paths from X_r^0 and X_s^0 to $X_{i_t}^0$ and $X_{i_{t'}}^0$, respectively. Lemma 4 then connects those two nodes in the necessary way (see Figure 9).

The dynamic α-graph of our first example is given in Figure 10. Note the path from X_1^0 to X_7^0

$$X_1^0, X_3^0, X_6^0, X_4^0, X_2^0, X_5^0, X_7^0$$

There is a net loss of two levels from X_1^0 in this path, but it never rises above the initial level of that variable, nor drops more than two levels below it. By the theorem, with $n = 0$, $k = 2$, and $l = 0$, we should have

$$\sigma_2\sigma_1\gamma_0(X_1^0) = \gamma_2(X_7^0)$$

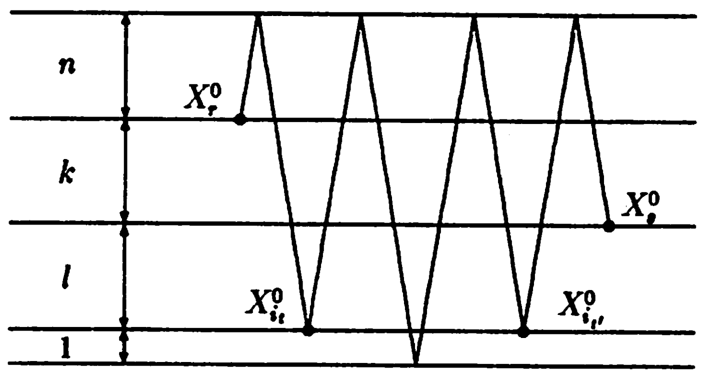

Figure 9:

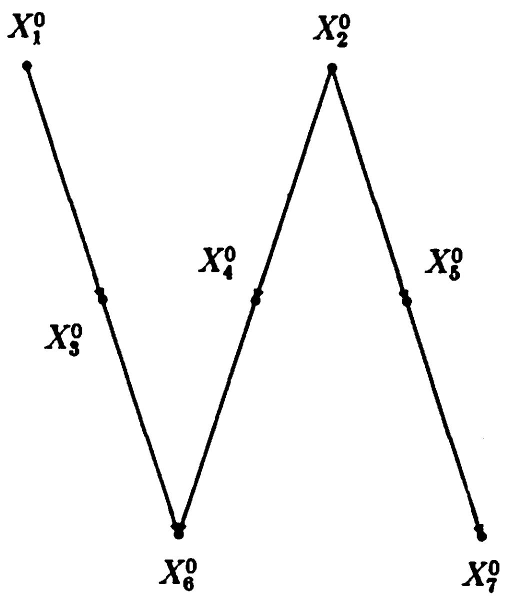

Figure 10:

Checking the first argument of Q_0 and the last argument of Q_2, we see that the variables are indeed the same.

One can obtain a regular expression for the paths between nodes by replacing each directed arc with two directed arcs, one forward and labelled with an 'F' and one directed in the opposite way and labelled 'R'. The graph is now a net over the alphabet {F, R}, and there are algorithms for determining the regular expression for all paths between two nodes. Of course, we are not interested in the specific sequence of letters in the path so much as the relative counts of the letters F and R, and we are currently investigating how to derive this information from the regular expression.

It is our belief that the characterization detailed in this paper will be useful in deriving a more cogent proof of a result claimed by Agrawal and Jagadish, and apparently independently by Ness, that any linear recursion can be expressed as a transitive closure ([Jaga86], [Naug87]) and in recognizing the boundedness property. We are hopeful that it will also serve the more general purpose of identifying repeated patterns of variable overlap.

References

[Agra86] Agrawal, R., and Jagadish, H., "On Bounded Linear Recursion", AT&T Bell Laboratories Technical Memorandum, 1986.

[Banc86] Bancilhon, F. , and Ramakrishnan, R., "An Amateur's Introduction to Recursive Query Processing", *Proceedings of SIGMOD '86 International Conference on Management of Data*, pps. 16–52.

[Brig87] Briggs, D., "A Reconsideration of the Termination Condition of the Henschen-Naqvi Technique", COINS TR 87-11, University of Massachusetts, 1987.

[Hens84] Henschen, L., and Naqvi, S., "On Compiling Queries on Recursive First-Order Data Bases", *JACM*, Vol. 31, January 1984, pps. 47–85.

[Ioan86] Ioannidis, Y., *Processing Recursion in Database Systems*, Ph. D. Thesis, University of California at Berkeley, 1986.

[Jaga86] Jagadish, H., and Agrawal, R., "A Study of Transitive Closure as a Recursion Mechanism", unpublished manuscript.

[Lloy84] Lloyd, J., *Foundations of Logic Programming*, Springer-Verlag, 1984.

[Naug87] Naughton, Jeffrey, personal communication.