

Answering Queries Using Views, Information Integration, CSCI 6967-01

1 Query Containment

Suppose Q_1 and Q_2 are two queries over the same database schema and the projection attributes of Q_1 and Q_2 are the same. Then, we say that Q_1 is contained in Q_2 or Q_1 is subsumed by Q_2 , denoted by $Q_1 \subseteq Q_2$, if all the tuples in Q_1 are also in Q_2 for all possible databases D over this schema. So, the subsumption relationship is based on the meaning of the queries, not the database contents. Generally, we say that Q_1 is contained in Q_2 if the contents of Q_1 can be obtained from Q_2 . In this sense, Q_1 is more restrictive and Q_2 . Hence, it contains a subset of the tuples in Q_2 . From a query rewriting perspective, we say that Q_1 can be used to answer Q_2 if the tuples in Q_1 are contained in Q_2 and the attributes in Q_2 can be obtained from Q_1 . We will give this definition of containment below.

Given two queries of the form (containing only predicates in the body)

$$Q_1 : H : -G_1, \dots, G_n$$

$$Q_2 : H' : -G'_1, \dots, G'_m$$

we say that $Q_1 \subseteq Q_2$ if there exists a substitution θ such that for each goal G'_i in Q_2 , there exists a subgoal G_j in Q_1 such that $G_j = G'_i\theta$, and each variable in $H'\theta$ is equivalent to a variable in H .

Note that two different subgoals G'_i, G'_k in Q_2 may map to the same predicate in Q_1 and some goals in Q_1 may not have an equivalent in Q_2 . This is true because Q_1 is more restrictive, i.e. it may have additional goals and it may make certain variables that are not necessary equivalent in Q_2 equivalent.

As an example, consider the following views:

$$\begin{aligned} V1(X, Y) &: -p(X, Y), r(Y, Y), s(Y, Z) \\ V2(X1, Y1) &: -p(X1, Y1), r(Y1, Z1), r(Z1, Y1) \end{aligned}$$

Now, given $\theta = \{X1 = X, Y1 = Y, Z1 = Y\}$, $V2\theta$ becomes:

$$V2'(X, Y) : -p(X, Y), r(Y, Y), r(Y, Y)$$

We can see that each predicate in $V2'$ have an equivalent view in $V1$, hence $V1 \subseteq V2$.

Given two queries of the form (containing built-in predicates the body)

$$Q_1 : H : -G_1, \dots, G_n, A_1, \dots, A_p$$

$$Q_2 : H' : -G'_1, \dots, G'_m, A'_1, \dots, A'_s$$

we say that $Q_1 \subseteq Q_2$ if there exists a substitution θ such that for each goal G'_i in Q_2 , there exists a subgoal G_j in Q_1 such that $G_j = G'_i\theta$, and each variable in $H'\theta$ is equivalent to a variable in H . In addition, each built-in predicate A'_i in Q_1 maps to $A'_i\theta$ that is implied by A_1, \dots, A_p .

The following are set of complete axioms for implication using built-in predicates:

- A1 : $X \leq X$
- A2 : $X < Y$ implies $X \leq Y$
- A3 : $X < Y$ implies $X \neq Y$
- A4 : $X \leq Y, X \neq Y$ imply $X < Y$
- A5 : $X \neq Y$ implies $Y \neq X$
- A6 : $X < Y, Y < Z$ imply $X < Z$
- A7 : $X \leq Y, Y \leq Z$ imply $X \leq Z$
- A8 : $X \leq Z, Z \leq Y, X \leq W, W \leq Y, W \neq Z$ imply $X \neq Y$

Note that this definition is sufficient to show containment, but it is not necessary. For example,

$$\begin{aligned} p(X, Y) &: - q(X, Y), r(U, V), r(V, U). \\ p(X, Y) &: - r(U, V), U \leq V. \end{aligned}$$

The first query is contained in the second one, but there exists no explicit mapping to prove this.

For example, suppose we are given the following views:

$$\begin{aligned} V3(X, Y) &: -p(X, Y), r(Y, Y), s(Y, Z), Y < Z, Z < 2 \\ V4(X1, Y1) &: -p(X1, Y1), r(Y1, Z1), r(Z1, Y1), Y1 < 5, Z1 < 10 \end{aligned}$$

Now, given $\theta = \{X1 = X, Y1 = Y, Z1 = Y\}$, $V4\theta$ and simplifying the built-in predicates, $V4$ becomes:

$$V4'(X, Y) : -p(X, Y), r(Y, Y), r(Y, Y), Y < 5$$

Again, each predicate in $V4'$ maps to a predicate in $V3$. Furthermore, $Y < 5$ is implied by $Y < Z, Z < 2$. Hence, $V3 \subseteq V4$.

2 Query Rewriting

Now, suppose we are given a set of view definitions in the Local As View (LAV) approach.

$$V1(X, Z) : -p(X, a), r(a, Z)$$

$V2(X, Z) : -p(X, Y), r(Y, Y)$
 $V3(Y) : -p(X, Y), r(Y, Z)$
 $V4(X) : -p(X, Y), r(Y, Z)$
 $V5(X, W) : -p(X, Y), s(Z, W)$

Now, given queries of the form:

$Q1(X) : -p(X, Y), r(Y, Z)$
 $Q2(X) : -p(X, Y), r(Y, Z), Y > 5$
 $Q3(X) : -p(X, Y), r(Y, Z), s(Z, W)$

the problem is to find a way to answer the queries using the views. As none of the base predicates (p, r, s) are stored explicitly, we cannot query them directly. Hence, we have to rewrite the queries completely using the above views.

The query rewriting problem is then rewriting a query Q as Q' such that the body of Q' refers only to the view names and possibly built-in predicates such that

- the query Q' is safe, all variables in the head refer to variables in the view heads,
- the built-in predicates refer to variables that are in the head of at least one view definition,
- if we replace the views in Q' with their definition to get expression Q^* then $Q^* \subseteq Q$.

Given a query Q , suppose all possible rewritings is given by Q_1, \dots, Q_n where each $Q_i \subseteq Q$. Then, we are given that $(Q_1 \cup \dots \cup Q_n) \subseteq Q$. Hence, the objective is to execute them all and take the union of the results to find the maximal set of answers to the query. In cases where this is not feasible, then we need to figure out the best rewritings to execute. This is the topic of query optimization that we will discuss later.

$RedCars(CarNo, Model, Year) : - CarDesc(CarNo, Model, red, Year).$
 $AntiqueCars(CarNo, Model, Year) : - CarDesc(CarNo, Model, Color, Year), Year < 1970.$
 $CarAndDriver(Model, Review) : - Review(Model, Review, 10).$

Consider the following three queries:

$Q1 : q1(CarNo, Review) : -CarDesc(CarNo, Model, C, Y), Review(Model, Review, Rating).$
 $Q2 : q2(CarNo, Review) : -CarDesc(CarNo, Model, C, Y), Review(Model, Review, 10).$
 $Q3 : q3(CarNo, Review) : -CarDesc(CarNo, Model, C, Y), Review(Model, Review, 10), Y < 1990.$

The following are the possible rewritings for each query:

$Q11 : q1(\text{CarNo}, \text{Review}) : \text{RedCars}(\text{CarNo}, \text{Model}, \text{Year}), \text{CarAndDriver}(\text{Model}, \text{Review})$
 $Q12 : q1(\text{CarNo}, \text{Review}) : \text{AntiqueCars}(\text{CarNo}, \text{Model}, \text{Year}), \text{CarAndDriver}(\text{Model}, \text{Review})$
 $Q21 : q2(\text{CarNo}, \text{Review}) : \text{RedCars}(\text{CarNo}, \text{Model}, \text{Year}), \text{CarAndDriver}(\text{Model}, \text{Review})$
 $Q22 : q2(\text{CarNo}, \text{Review}) : \text{AntiqueCars}(\text{CarNo}, \text{Model}, \text{Year}), \text{CarAndDriver}(\text{Model}, \text{Review})$
 $Q31 : q3(\text{CarNo}, \text{Review}) : \text{RedCars}(\text{CarNo}, \text{Model}, \text{Year}), \text{CarAndDriver}(\text{Model}, \text{Review}), \text{Year} < 1990$
 $Q32 : q3(\text{CarNo}, \text{Review}) : \text{AntiqueCars}(\text{CarNo}, \text{Model}, \text{Year}), \text{CarAndDriver}(\text{Model}, \text{Review}), \text{Year} < 1990$

What are the containment relationships? We note that even though before rewriting $Q1$ looks more general than $Q2$ and $Q2$ is more general than $Q3$, after rewriting $Q1$ and $Q2$ are equivalent with respect to both rewritings. Also, $Q32 = Q22$. But, $Q32 \subseteq Q22$. Note that while $Q3 \subseteq Q1$, $Q32$ and $Q11$ are incomparable.

A rewriting Q' of a query Q is said to be maximal if there is no rewriting Q'' such that $Q'' \not\subseteq Q'$ and $Q' \subseteq Q'' \subseteq Q$. In other words, maximal rewritings return the largest possible number of tuples for each query.

3 Computing Rewritings

To find the rewritings, we must try to answer each predicate in the body of the query using a view. There are two known algorithms for this problem.

3.1 Bucket Algorithm

In this algorithm, the idea is to find a way to answer the queries using views only. Given a query with k subgoals, we can limit our attention only to the solutions that use at most k views. Any solution with more subgoals is going to be contained in one of the solutions with k subgoals.

The algorithm works in two steps. In the first step, we find all views that can possibly answer one of the subgoals in the query. We create a bucket for each subgoal G and then place all relevant views V in each subgoal bucket if the body of V contains a predicate G' that unifies with G using a substitution θ such that the conjunction of the built-in predicates in V and the query are satisfiable after applying θ . In each bucket, we place the head of view after substitution θ .

In the second step, all possible combination of views from each bucket are considered. For each combination, we check whether the built-in predicates are satisfiable and there exists a subsumption mapping from the original query to (unfolded version of the) rewriting. Note that in this final step, we may end up unifying different variables to combine different occurrences of the same view predicate (different from the previous method).

Consider the following views:

$V1(X1, T1) : \neg p(X1, Y1), p(Y1, Z1), Z1 < 10, r(Y1, T1)$
 $V2(X2, Y2) : \neg p(X2, Y2), s(Y2, Z2), Y2 > 20$
 $V3(X3, Y3) : \neg r(X3, Y3), s(Y3, Y3), Y3 < 20$

and the following query:

$$Q(X, Y) : \neg p(X, Y), p(Y, Z), Y < 5, r(X, Y), s(X, Z)$$

We now create a bucket for each predicate:

$p(X, Y)$	$p(Y, Z)$	$r(X, Y)$	$s(X, Z)$
$V1(X, T1)$	$V1(Y, T1)$	$V1(X1, Y)$	$V2(X, Y2)$
$V1(X1, T1)$	$V1(X1, T1)$	$V3(X, Y)$	$V3(X3, Z)$
	$V2(Y, Z)$		

Note that the first p predicate does not unify with $V3$ due to the inconsistency in the built-in predicates. Now, we consider all combinations of these predicates and drop built-in predicates from the rewriting unless they are returned by the views.

$$\begin{aligned}
Q1(X, Y) &: \neg V1(X, T1), V1(Y, T1), V1(X1, Y), V2(X, Y2), Y < 5 \\
Q1(X, Y) &: \neg V1(X, T1), V1(Y, T1), V1(X1, Y), V3(X3, Z), Y < 5 \\
Q1(X, Y) &: \neg V1(X, T1), V1(Y, T1), V3(X, Y), V2(X, Y2), Y < 5 \\
Q1(X, Y) &: \neg V1(X, T1), V1(Y, T1), V3(X, Y), V3(X3, Z), Y < 5 \\
Q1(X, Y) &: \neg V1(X, T1), V1(X1, T1), V1(X1, Y), V2(X, Y2), Y < 5 \\
Q1(X, Y) &: \neg V1(X, T1), V1(X1, T1), V1(X1, Y), V3(X3, Z), Y < 5 \\
Q1(X, Y) &: \neg V1(X, T1), V1(X1, T1), V3(X, Y), V2(X, Y2), Y < 5 \\
Q1(X, Y) &: \neg V1(X, T1), V1(X1, T1), V3(X, Y), V3(X3, Z), Y < 5 \\
Q1(X, Y) &: \neg V1(X, T1), V2(Y, Z), V1(X1, Y), V2(X, Y2), Y < 5 \\
Q1(X, Y) &: \neg V1(X, T1), V2(Y, Z), V1(X1, Y), V3(X3, Z), Y < 5 \\
Q1(X, Y) &: \neg V1(X, T1), V2(Y, Z), V3(X, Y), V2(X, Y2), Y < 5 \\
Q1(X, Y) &: \neg V1(X, T1), V2(Y, Z), V3(X, Y), V3(X3, Z), Y < 5
\end{aligned}$$

An equal number will be generated by replacing $V1(X, T1)$ with $V1(X1, T1)$. We have to check whether each rewriting is contained in the original query. If not, then we must try to eliminate multiple occurrences of the same view by unifying them and repeat the containment check. This algorithm is guaranteed to return maximal rewritings.

3.2 Skolem Functions

We can replace all the variables in the view definition that do not appear in the body with a different Skolem function that ranges over the variables in the head of that view. For example, given the following views:

$$\begin{aligned}
V1(X1, T1) &: \neg p(X1, Y1), p(Y1, Z1), Z1 < 10, r(Y1, T1) \\
V2(X2, Y2) &: \neg p(X2, Y2), s(Y2, Z2), Y2 > 20 \\
V3(X3, Y3) &: \neg r(X3, Y3), s(Y3, Y3), Y3 < 20
\end{aligned}$$

We can rewrite them as follows (disregarding the built-in functions for the time being):

$$(1) p(X11, f1(X11, T11)) : \neg V1(X11, T11)$$

- (2) $p(f1(X12, T12), f2(X12, T12)) : -V1(X12, T12)$
 - (3) $p(f1(X13, T13), T13) : -V1(X13, T13)$
 - (4) $p(X21, Y21) : -V2(X21, Y21)$
 - (5) $s(Y22, f3(X22, Y22)) : -V2(X22, Y22)$
 - (6) $r(X31, Y31) : -V3(X31, Y31)$
 - (7) $s(Y32, Y32) : -V3(X32, Y32)$
-

Now, we can treat these rules as ordinary logic rules and rewrite the query to find rewritings. Suppose, we are given the following query:

$$Q(X, Y) : -p(X, Y), p(Y, Z), Y < 5, r(X, Y), s(X, Z)$$

We can use resolution to find all rewritings. One possible run is shown below.

$$Q(X, Y) : -p(X, Y), p(Y, Z), Y < 5, r(X, Y), s(X, Z)$$

(rule 1, $\{X11 = X, Y = f(X11, T11)\}$)

$$Q(X, f(X11, T11)) : -V1(X, T1), p(Y, Z), Y < 5, r(X, Y), s(X, Z)$$

We immediately see that we cannot use this rule since the head of the query now has a Skolem, which means we cannot return all the attributes in this query. Let's try a different route.

$$Q(X, Y) : -p(X, Y), p(Y, Z), Y < 5, r(X, Y), s(X, Z)$$

(rule 4, $\{X21 = X, Y21 = Y\}$)

$$Q(X, Y) : -V2(X, Y), p(Y, Z), Y < 5, r(X, Y), s(X, Z)$$

(rule 4, $\{X21 = Y, Y21 = Z\}$)

$$Q(X, Y) : -V2(X, Y), V2(Y, Z), Y < 5, r(X, Y), s(X, Z)$$

(rule 6, $\{X31 = X, Y31 = Y\}$)

$$Q(X, Y) : -V2(X, Y), V2(Y, Z), Y < 5, V3(X, Y), s(X, Z)$$

(rule 5, $\{Y22 = X, Z = f3(X22, Y)\}$)

$$Q(X, Y) : -V2(X, Y), V2(Y, Z), Y < 5, V3(X, Y), V2(X22, X)$$

Now, suppose we disregard the built-in predicates. If the resulting rewriting has no Skolem functions, then it is a maximal rewriting.

The built-in functions must be handled separately. One possible solution is to accumulate them as a contextual constraint. We add a special predicate called *context* and display all built-in constraints in this part.

-
- (1) $p(X11, f1(X11, T11)) : -V1(X11, T11)$
 - (2) $p(f1(X12, T12), f2(X12, T12)) : -V1(X12, T12), context(f2(X12, T12) < 10)$
 - (3) $p(f1(X13, T13), T13) : -V1(X13, T13)$
 - (4) $p(X21, Y21) : -V2(X21, Y21), context(Y21 > 20)$
 - (5) $s(Y22, f3(X22, Y22)) : -V2(X22, Y22), context(Y22 > 20)$
 - (6) $r(X31, Y31) : -V3(X31, Y31), context(Y31 < 20)$
 - (7) $s(Y32, Y32) : -V3(X32, Y32), context(Y32 < 20)$
-

Now, we can carry out the same resolution operation as before, but everytime we add a view head, we merge the context of these views also. Anytime, the context and the remaining built-in predicates are inconsistent, then we fail. Anytime, a built-in predicate in the query is implied by the context, we can remove it. If the remaining built-in predicates in the query have Skolem functions, then this is not a valid rewriting. However, if the query were to be modified by removing these conditions, then we have a valid rewriting.

$$Q(X, Y) : -p(X, Y), p(Y, Z), Y < 5, r(X, Y), s(X, Z)$$

(rule 4, $\{X21 = X, Y21 = Y\}$)

$$Q(X, Y) : -V2(X, Y), p(Y, Z), Y < 5, r(X, Y), s(X, Z), context(Y > 20)$$

Now, $Y < 5$ and $Y > 20$ is inconsistent. So, we fail in this rewriting.

Suppose we are given the following:

$$V1(X, T) : -p(X, Y), r(Y, Z), s(Z, T), Z < 5$$

(1) $p(X1, g(X1, T1)) : -V1(X1, T1)$

(2) $r(g(X2, T2), g2(X2, T2)) : -V1(X2, T2), context(g2(X2, T2) < 5)$

(3) $s(g2(X3, T3), T3) : -V1(X3, T3), context(g2(X3, T3) < 5)$

Given the query:

$$Q(X, T) : -p(X, Y), r(Y, Z), s(Z, T), Z <> 10, Y < 5$$

We can now write:

$$Q(X, T) : -p(X, Y), r(Y, Z), s(Z, T), Z <> 10, Y < 5$$

(rule 1, $X1 = X, Y = g(X, T1)$)

$$Q(X, T) : -V1(X, T1), r(g(X, T1), Z), s(Z, T), Z <> 10, g(X, T1) < 5$$

(rule 2, $X2 = X, T2 = T1, Z = g2(X, T1)$)

$$Q(X, T) : -V1(X, T1), V1(X, T1), s(g2(X, T1), T), g2(X, T1) <> 10, g(X, T1) < 5, context(g2(X, T1) < 5)$$

(note that $g2(X, T1) < 5$ implies $g2(X, T1) <> 10$, so it can be removed.)

(note also that we can merge two appearances of $V1(X, T1)$ into one)

$$Q(X, T) : -V1(X, T1), s(g2(X, T1), T), g(X, T1) < 5, context(g2(X, T1) < 5)$$

(rule 3, $X3 = X, T3 = T1 = T$)

$$Q(X, T) : -V1(X, T), V1(X, T), g(X, T) < 5, context(g2(X, T) < 5)$$

(simplify again)

$$Q(X, T) : -V1(X, T), g(X, T) < 5, context(g2(X, T) < 5)$$

This is not a valid rewriting as we cannot check the condition $Y < 5$. However, if the query would be simplified by removing this condition, then we have a rewriting. Note that context is not checked, it is a listing of what is true for $V1$.