

where conditions C_1, \dots, C_k, T refer only to variables t, p_1, \dots, p_m, q and D_1, \dots, D_l refer only to q, r_1, \dots, r_n . It is easily verified that this is equivalent to the sequence

$$\begin{aligned} temp(q) &\leftarrow Q(q), R_1(r_1), \dots, R_n(r_n), D_1, \dots, D_l \\ ans(t) &\leftarrow P_1(p_1), \dots, P_m(p_m), temp(q), C_1, \dots, C_k, T. \end{aligned}$$

In this example, variable q acts as a “pivot” around which the detachment is performed. More general forms of join detachment can be developed in which a set of variables serves as the pivot (see Exercise 6.6).

Tuple substitution chooses one of the underlying relations R_j and breaks the n -variable join into a set of $(n - 1)$ -variable joins, one for each tuple in R_j . Consider again a query of form (**) just shown. The tuple substitution of this on R_i is given by the “program”

for each r in R_i do

$$\begin{aligned} ans(s) &+\leftarrow R_1(s_1), \dots, R_{i-1}(s_{i-1}), R_{i+1}(s_{i+1}), \dots, R_n(s_n), \\ &(C_1, \dots, C_m, T)[s_i/r]. \end{aligned}$$

Here we use $+\leftarrow$ to indicate that ans is to accumulate the values stemming from all tuples r in (the value of) R_i ; furthermore, r is substituted for s_i in all of the conditions.

There is an obvious trade-off here between reducing the number of variables in the join and the number of tuples in R_i . In the INGRES optimizer, each of the R_i 's is considered as a candidate for forming the tuple substitution. During this process single-variable conditions may be applied to the R_i 's to decrease their size.

6.2 Global Optimization

The techniques for creating evaluation plans presented in the previous section are essentially *local* in their operation: They focus on clusters of contiguous nodes in a query tree. In this section we develop an approach to the *global* optimization of conjunctive queries. This allows a transformation of an algebra query that removes several joins in a single step, a capability not provided by the techniques of the previous section. The global optimization technique is based on an elegant Homomorphism Theorem.

The Homomorphism Theorem

For two queries q_1, q_2 over the same schema \mathbf{R} , q_1 is *contained* in q_2 , denoted $q_1 \subseteq q_2$, if for each \mathbf{I} over \mathbf{R} , $q_1(\mathbf{I}) \subseteq q_2(\mathbf{I})$. Clearly, $q_1 \equiv q_2$ iff $q_1 \subseteq q_2$ and $q_2 \subseteq q_1$. The Homomorphism Theorem provides a characterization for containment and equivalence of conjunctive queries.

We focus here on the tableau formalism for conjunctive queries, although the rule-based formalism could be used equally well. In addition, although the results hold for tableau queries over database schemas involving more than one relation, the examples presented focus on queries over a single relation.

Recall the notion of *valuation*—a mapping from variables to constants extended to be the identity on constants and generalized to free tuples and tableaux in the natural fashion.

$R \mid \begin{array}{cc} A & B \\ \hline x & y \\ \hline x & y \end{array}$	$R \mid \begin{array}{cc} A & B \\ \hline x & y_1 \\ x_1 & y_1 \\ x_1 & y \\ \hline x & y \end{array}$	$R \mid \begin{array}{cc} A & B \\ \hline x & y_1 \\ x_1 & y_1 \\ x_1 & y_2 \\ x_2 & y_2 \\ x_2 & y \\ \hline x & y \end{array}$	$R \mid \begin{array}{cc} A & B \\ \hline x & y_1 \\ x_1 & y \\ \hline x & y \end{array}$
$q_0 = (T_0, \langle x, y \rangle)$	$q_1 = (T_1, \langle x, y \rangle)$	$q_2 = (T_2, \langle x, y \rangle)$	$q_\omega = (T_\omega, \langle x, y \rangle)$
(a)	(b)	(c)	(d)

Figure 6.4: Tableau queries used to illustrate the Homomorphism Theorem

Valuations are used in the definition of the semantics of tableau queries. More generally, a *substitution* is a mapping from variables to variables and constants, which is extended to be the identity on constants and generalized to free tuples and tableaux in the natural fashion. As will be seen, substitutions play a central role in the Homomorphism Theorem.

We begin the discussion with two examples. The first presents several simple examples of the Homomorphism Theorem in action.

EXAMPLE 6.2.1 Consider the four tableau queries shown in Fig. 6.4. By using the Homomorphism Theorem, it can be shown that $q_0 \subseteq q_1 \subseteq q_2 \subseteq q_\omega$.

To illustrate the flavor of the proof of the Homomorphism Theorem, we argue informally that $q_1 \subseteq q_2$. Note that there is substitution θ such that $\theta(T_2) \subseteq T_1$ and $\theta(\langle x, y \rangle) = \langle x, y \rangle$ [e.g., let $\theta(x_1) = \theta(x_2) = x_1$ and $\theta(y_1) = \theta(y_2) = y_1$]. Now suppose that I is an instance over AB and that $t \in q_1(I)$. Then there is a valuation ν such that $\nu(T_1) \subseteq I$ and $\nu(\langle x, y \rangle) = t$. It follows that $\theta \circ \nu$ is a valuation that embeds T_2 into I with $\theta \circ \nu(\langle x, y \rangle) = t$, whence $t \in q_2(I)$.

Intuitively, the existence of a substitution embedding the tableau of q_2 into the tableau of q_1 and mapping the summary of q_2 to the summary of q_1 implies that q_1 is *more restrictive* than q_2 (or more correctly, *no less restrictive* than q_2 .) Surprisingly, the Homomorphism Theorem states that this is also a necessary condition for containment (i.e., if $q \subseteq q'$, then q is more restrictive than q' in this sense).

The second example illustrates a limitation of the techniques discussed in the previous section.

EXAMPLE 6.2.2 Consider the two tableau queries shown in Fig. 6.5. It can be shown that $q \equiv q'$ but that q' cannot be obtained from q using the rewrite rules of the previous section (see Exercise 6.3) or the other optimization techniques presented there.

R	A	B
	x	x
	x	y_1
	y_1	y_2
	\vdots	\vdots
	y_{n-1}	y_n
	y_n	x
	x	

$$q = (T, u)$$

(a)

R	A	B
	x	x
	x	

$$q' = (T', u)$$

(b)

Figure 6.5: Pair of equivalent tableau queries

Let $q = (\mathbf{T}, u)$ and $q' = (\mathbf{T}', u')$ be two tableau queries over the same schema \mathbf{R} . A *homomorphism* from q' to q is a substitution θ such that $\theta(\mathbf{T}') \subseteq \mathbf{T}$ and $\theta(u') = u$.

THEOREM 6.2.3 (Homomorphism Theorem) Let $q = (\mathbf{T}, u)$ and $q' = (\mathbf{T}', u')$ be tableau queries over the same schema \mathbf{R} . Then $q \subseteq q'$ iff there exists a homomorphism from (\mathbf{T}', u') to (\mathbf{T}, u) .

Proof Suppose first that there exists a homomorphism θ from q' to q . Let \mathbf{I} be an instance over \mathbf{R} . To see that $q(\mathbf{I}) \subseteq q'(\mathbf{I})$, suppose that $w \in q(\mathbf{I})$. Then there is a valuation ν that embeds \mathbf{T} into \mathbf{I} such that $\nu(u) = w$. It is clear that $\theta \circ \nu$ embeds \mathbf{T}' into \mathbf{I} and $\theta \circ \nu(u') = w$, whence $w \in q'(\mathbf{I})$ as desired.

For the opposite inclusion, suppose that $q \subseteq q'$ [i.e., that $(\mathbf{T}, u) \subseteq (\mathbf{T}', u')$]. Speaking intuitively, we complete the proof by applying both q and q' to the “instance” \mathbf{T} . Because q will yield the free tuple u , q' also yields u (i.e., there is an embedding θ of \mathbf{T}' into \mathbf{T} that maps u' to u). To make this argument formal, we construct an instance $\mathbf{I}_{\mathbf{T}}$ that is isomorphic to \mathbf{T} .

Let V be the set of variables occurring in \mathbf{T} . For each $x \in V$, let a_x be a new distinct constant not occurring in \mathbf{T} or \mathbf{T}' . Let μ be the valuation mapping each x to a_x , and let $\mathbf{I}_{\mathbf{T}} = \mu(\mathbf{T})$. Because μ is a bijection from V to $\mu(V)$, and because $\mu(V)$ has empty intersection with the constants occurring in \mathbf{T} , the inverse μ^{-1} of μ is well defined on $\text{adom}(\mathbf{I}_{\mathbf{T}})$.

It is clear that $\mu(u) \in q(\mathbf{I}_{\mathbf{T}})$, and so by assumption, $\mu(u) \in q'(\mathbf{I}_{\mathbf{T}})$. Thus there is a valuation ν that embeds \mathbf{T}' into $\mathbf{I}_{\mathbf{T}}$ such that $\nu(u') = \mu(u)$. It is now easily verified that $\nu \circ \mu^{-1}$ is a homomorphism from q' to q . ■

Permitting a slight abuse of notation, we have the following (see Exercise 6.8).

COROLLARY 6.2.4 For tableau queries $q = (\mathbf{T}, u)$ and $q' = (\mathbf{T}', u')$, $q \subseteq q'$ iff $u \in q'(\mathbf{T})$.

We also have

COROLLARY 6.2.5 Tableau queries q, q' over schema \mathbf{R} are equivalent iff there are homomorphisms from q to q' and from q' to q .

In particular, if $q = (\mathbf{T}, u)$ and $q' = (\mathbf{T}', u')$ are equivalent, then u and u' are identical up to one-one renaming of variables.

Only one direction of the preceding characterization holds if the underlying domain is finite (see Exercise 6.12). In addition, the direct generalization of the theorem to tableau queries with equality does not hold (see Exercise 6.9).

Query Optimization by Tableau Minimization

Although the Homomorphism Theorem yields a decision procedure for containment and equivalence between conjunctive queries, it does not immediately provide a mechanism, given a query q , to find an “optimal” query equivalent to q . The theorem is now applied to obtain just such a mechanism.

We note first that there are simple algorithms for translating tableau queries into (satisfiable) SPC queries and vice versa. More specifically, given a tableau query, the corresponding generalized SPC query has the form $\pi_j(\sigma_F(R_1 \times \cdots \times R_k))$, where each component R_i corresponds to a distinct row of the tableau. For the opposite direction, one algorithm for translating SPC queries into tableau queries is first to translate into the normal form for generalized SPC queries and then into a tableau query. A more direct approach that inductively builds tableau queries corresponding to subexpressions of an SPC query can also be developed (see Exercise 4.18). Analogous remarks apply to SPJR queries.

The goal of the optimization presented here is to minimize the number of rows in the tableau. Because the number of rows in a tableau query is one more than the number of joins in the SPC (SPJR) query corresponding to that tableau (see Exercise 4.18c), the tableau minimization procedure provides a way to minimize the number of joins in SPC and SPJR queries.

Surprisingly, we show that an optimal tableau query equivalent to tableau query q can be obtained simply by eliminating some rows from the tableau of q .

We say that a tableau query (\mathbf{T}, u) is *minimal* if there is no query (\mathbf{S}, v) equivalent to (\mathbf{T}, u) with $|\mathbf{S}| < |\mathbf{T}|$ (i.e., where \mathbf{S} has strictly fewer rows than \mathbf{T}).

We can now demonstrate the following.

THEOREM 6.2.6 Let $q = (\mathbf{T}, u)$ be a tableau query. Then there is a subset \mathbf{T}' of \mathbf{T} such that $q' = (\mathbf{T}', u)$ is a minimal tableau query and $q' \equiv q$.

Proof Let (\mathbf{S}, v) be a minimal tableau that is equivalent to q . By Corollary 6.2.5, there are homomorphisms θ from q to (\mathbf{S}, v) and λ from (\mathbf{S}, v) to q . Let $\mathbf{T}' = \theta \circ \lambda(\mathbf{S})$. It is straightforward to verify that $(\mathbf{T}', u) \equiv q$ and $|\mathbf{T}'| \leq |\mathbf{S}|$. By minimality of (\mathbf{S}, v) , it follows that $|\mathbf{T}'| = |\mathbf{S}|$, and (\mathbf{T}', u) is minimal. ■

Example 6.2.7 illustrates how one might minimize a tableau by hand.

R	A	B	C
u_1	x_2	y_1	z
u_2	x	y_1	z_1
u_3	x_1	y	z_1
u_4	x	y_2	z_2
u_5	x_2	y_2	z
u	x	y	z

Figure 6.6: The tableau (T, u)

EXAMPLE 6.2.7 Let R be a relation schema of sort ABC and (T, u) the tableau over R in Fig. 6.6. To minimize (T, u) , we wish to detect which rows of T can be eliminated. Consider u_1 . Suppose there is a homomorphism θ from (T, u) onto itself that eliminates u_1 [i.e., $u_1 \notin \theta(T)$]. Because any homomorphism on (T, u) is the identity on u , $\theta(z) = z$. Thus $\theta(u_1)$ must be u_5 . But then $\theta(y_1) = y_2$, and $\theta(u_2) \in \{u_4, u_5\}$. In particular, $\theta(z_1) \in \{z_2, z\}$. Because u_3 involves z_1 , it follows that $\theta(u_3) \neq u_3$ and $\theta(y) \neq y$. But the last inequality is impossible because y is in u so $\theta(y) = y$. It follows that row u_1 cannot be eliminated and is in the minimal tableau. Similar arguments show that u_2 and u_3 cannot be eliminated. However, u_4 and u_5 can be eliminated using $\theta(y_2) = y_1$, $\theta(z_2) = z_1$ (and identity everywhere else). The preceding argument emphasizes the global nature of tableau minimization.

The preceding theorem suggests an improvement over the optimization strategies described in Section 6.1. Specifically, given a (satisfiable) conjunctive query q , the following steps can be used:

1. Translate q into a tableau query.
2. Minimize the number of rows in the tableau of this query.
3. Translate the result into a generalized SPC expression.
4. Apply the optimization techniques of Section 6.1.

As illustrated by Examples 6.2.2, 6.2.7, and 6.2.8, this approach has the advantage of performing global optimizations that typical query rewriting systems cannot achieve.

EXAMPLE 6.2.8 Consider the relation schema R of sort ABC and the SPJR query q over R :

$$\pi_{AB}(\sigma_{B=5}(R)) \bowtie \pi_{BC}(\pi_{AB}(R) \bowtie \pi_{AC}(\sigma_{B=5}(R))).$$

R	A	B	C
	x	5	z_1
	x_1	5	z_2
	x_1	5	z
u	x	5	z

Figure 6.7: Tableau equivalent to q

The tableau (T, u) corresponding to it is that of Fig. 6.7. To minimize (T, u) , we wish to find a homomorphism that "folds" T onto a subtableau with minimal number of rows. (If desired, this can be done in several stages, each of which eliminates one or more rows.) Note that the first row cannot be eliminated because every homomorphism is the identity on u and therefore on x . A similar observation holds for the third row. However, the second row can be eliminated using the homomorphism that maps z_2 to z and is the identity everywhere else. Thus the minimal tableau equivalent to (T, u) consists of the first and third rows of T . An SPJR query equivalent to the minimized tableau is

$$\pi_{AB}(\sigma_{B=5}(R)) \bowtie \pi_{BC}(\sigma_{B=5}(R)).$$

Thus the optimization procedure resulted in saving one join operation.

Before leaving minimal tableau queries, we present a result that describes a strong correspondence between equivalent minimal tableau queries. Two tableau queries (\mathbf{T}, u) , (\mathbf{T}', u') are *isomorphic* if there is a one-one substitution θ that maps variables to variables such that $\theta((\mathbf{T}, u)) = (\mathbf{T}', u')$. In other words, (T, u) and (T', u') are the same up to renaming of variables. The proof of this result is left to the reader (see Exercise 6.11).

PROPOSITION 6.2.9 Let $q = (\mathbf{T}, u)$ and $q' = (\mathbf{T}', u')$ be minimal and equivalent. Then q and q' are isomorphic.

Complexity of Tableau Decision Problems

The following theorem shows that determining containment and equivalence between tableau queries is NP-complete and tableau query minimization is NP-hard.

THEOREM 6.2.10 The following problems, given tableau queries q, q' , are NP-complete:

- (a) Is $q \subseteq q'$?
- (b) Is $q \equiv q'$?
- (c) Suppose that the tableau of q is obtained by deleting free tuples of the tableau of q' . Is $q \equiv q'$ in this case?