Conjunctive Queries

= safe, Datalog rules:

$$H \text{ :- } G_1 \text{ \& } \cdots \text{ \& } G_n$$

- Most common form of query; equivalent to select-project-join queries.
- Useful for optimization of active elements, e.g., checking distributed constraints, maintaining materialized views.)
- Useful for information integration.

Applying a CQ to a Database

If Q is a CQ, and D is a database of EDB facts, then $Q(D)$ is the set of heads of Q that we get when we:

- Substitute constants for variables in the body of Q in all possible ways.
- Require all subgoals to become true.

Example

$$p(X,Y) \; :- \; q(X,Z) \;\&\; q(Z,Y)$$

- EDB = $\{q(1,2),\ q(2,3),\ q(3,4)\}$.
- Only substitutions that make subgoals both true:
  1. $X \rightarrow 1$; $Y \rightarrow 3$; $Z \rightarrow 2$.
  2. $X \rightarrow 2$; $Y \rightarrow 4$; $Z \rightarrow 3$.
- Yield heads $p(1,3)$ and $p(2,4)$.

Containment

$Q_1 \subseteq Q_2$ iff for every database D, $Q_1(D) \subseteq Q_2(D)$.

- Containment problem is NP-complete, but not a "hard" problem in practical situations (short queries, few pairs of subgoals with same predicate).
- Function symbols do not make problems more difficult.
- Adding negated subgoals and/or arithmetic subgoals, e.g., $X < Y$, makes things more complex, but important special cases.

Example

```
A: p(X,Y) :- r(X,W) & b(W,Z) & r(Z,Y)
B: p(X,Y) :- r(X,W) & b(W,W) & r(W,Y)
```

- Claim: $B \subseteq A$.
- In proof, suppose $p(x, y)$ is in $B(D)$. Then there is some w such that $r(x, w)$, $b(w, w)$, and $r(w, y)$ are in D.
- In A, make the substitution $X \rightarrow x$, $Y \rightarrow y$, $W \rightarrow w$, $Z \rightarrow w$.
- Thus, the head of A becomes $p(x, y)$, and all subgoals of A are in D.
- Thus, $p(x, y)$ is also in $A(D)$, proving $B \subseteq A$.

Testing Containment of CQ's

1. Containment mappings.
2. Canonical databases.

- Similar for basic CQ case, but (2) is useful for more general cases like negated subgoals.

Containment Mappings

Mapping from variables of CQ Q_2 to variables of CQ Q_1 such that

1. Head of Q_2 becomes head of Q_1.
2. Each subgoal of Q_2 becomes *some* subgoal of Q_1.
   - It is not necessary that every subgoal of Q_1 is the target of some subgoal of Q_2.

Example

A, B as above:

```
A: p(X,Y) :- r(X,W) & b(W,Z) & r(Z,Y)
B: p(X,Y) :- r(X,W) & b(W,W) & r(W,Y)
```

- Containment mapping from A to B: $X \rightarrow X$, $Y \rightarrow Y$, $W \rightarrow W$, $Z \rightarrow W$.
- No containment mapping from B to A. Subgoal $b(W, W)$ in B can only go to $b(W, Z)$ in A. That would require both $W \rightarrow W$ and $W \rightarrow Z$.

Example

```
C1: p(X) :- a(X,Y) & a(Y,Z) & a(Z,W)
C2: p(X) :- a(X,Y) & a(Y,X)
```

- Containment mapping from C_1 to C_2. $X \rightarrow X$, $Y \rightarrow Y$, $Z \rightarrow X$, $W \rightarrow Y$.
- No containment mapping from C_2 to C_1. Proof:
  a) $X \rightarrow X$ required for head.
  b) Thus, first subgoal of C_2 must map to first subgoal of C_1; Y must map to Y.
  c) Similarly, 2nd subgoal of C_2 must map to 2nd subgoal of C_1, so X must map to Z.
  d) But we already found X maps to X.

Containment Mapping Theorem

$Q_1 \subseteq Q_2$ iff there exists a containment mapping from Q_2 to Q_1.

Proof (If)

Let μ: $Q_2 \rightarrow Q_1$ be a containment mapping. Let D be any DB.

- Every tuple t in $Q_1(D)$ is produced by some substitution σ on the variables of Q_1 that makes Q_1's subgoals all become facts in D.
- Claim: $\sigma \circ \mu$ is a substitution for variables of Q_2 that produces t.
  1. $\sigma \circ \mu(F_i) = \sigma(\text{some } G_j)$. Therefore, it is in D.
  2. $\sigma \circ \mu(H_2) = \sigma(H_1) = t$.
- Thus, every t in $Q_1(D)$ is also in $Q_2(D)$; i.e., $Q_1 \subseteq Q_2$.

Proof (Only If)

Key idea: *frozen* CQ.

1. Create a unique constant for each variable of the CQ Q.
2. Frozen Q is a database consisting of all the subgoals of Q, with the chosen constants substituted for variables.

Example

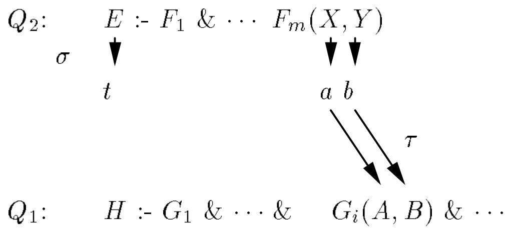
```
p(X) :- a(X,Y) & a(Y,Z) & a(Z,W)
```

Let x be the constant for X, etc. The relation for predicate a consists of the three tuples (x, y), (y, z), and (z, w).

Proof (Only If) Continued

Let $Q_1 \subseteq Q_2$. Let database D be the frozen Q_1.

- $Q_1(D)$ contains t, the "frozen" head of Q_1
  - ❖ Sounds gruesome, but the reason is that we can use the substitution in which each variable of Q_1 is replaced by its corresponding constant.
- Since $Q_1 \subseteq Q_2$, $Q_2(D)$ must also contain t.
- Let σ be the substitution of constants from D for the variables of Q_2 that makes each subgoal of Q_2 a tuple of D and yields t as the head.
- Let τ be the substitution that maps constants of D to their unique, corresponding variable of Q_1.



- $\tau \circ \sigma$ is a containment mapping from Q_2 to Q_1 because:
  a) The head of Q_2 is mapped by σ to t, and t is the frozen head of Q_1, so $\tau \circ \sigma$ maps the head of Q_2 to the "unfrozen" t, that is, the head of Q_1.
  b) Each subgoal F_i of Q_2 is mapped by σ to some tuple of D, which is a frozen version of some subgoal G_j of Q_1. Then $\tau \circ \sigma$ maps F_i to the unfrozen tuple, that is, to G_j itself.

Dual View of Containment Mappings

A containment mapping, defined as a mapping on variables, induces a mapping on subgoals.

- Therefore, we can alternatively define a containment mapping as a function on subgoals, thus inducing a mapping on variables.
- The containment mapping condition becomes: the subgoal mapping does not cause a variable to be mapped to two different variables or

constants, nor cause a constant to be mapped to a variable or a constant other than itself.

Example

Again consider

A: `p(X,Y) :- r(X,W) & b(W,Z) & r(Z,Y)`
B: `p(X,Y) :- r(X,W) & b(W,W) & r(W,Y)`

- Previously, we found the containment mapping $X \to X$, $Y \to Y$, $W \to W$, $Z \to W$ from A to B.
- We could as well describe this mapping as $r(X,W) \to r(X,W)$, $b(W,Z) \to b(W,W)$, and $r(Z,Y) \to r(W,Y)$.

Method of Canonical Databases

Instead of looking for a containment mapping from Q_2 to Q_1 in order to test $Q_1 \subseteq Q_2$, we can apply the following test:

1. Create a *canonical* database D that is the frozen body of Q_1.
2. Compute $Q_2(D)$.
3. If $Q_2(D)$ contains the frozen head of Q_1, then $Q_1 \subseteq Q_2$; else not.

- The proof that this method works is essentially the same as the argument for containment mappings:
  - ❖ The only way the frozen head of Q_1 can be in $Q_2(D)$ is for there to be a containment mapping $Q_2 \to Q_1$.

Example

C_1: `p(X) :- a(X,Y) & a(Y,Z) & a(Z,W)`
C_2: `p(X) :- a(X,Y) & a(Y,X)`

Here is the test for $C_2 \subseteq C_1$:

- Choose constants $X \to 0$, $Y \to 1$.
- Canonical DB from C_1 is

$$D = \{a(0,1),\ a(1,0)\}$$

- $C_1(D) = \{p(0), p(1)\}$.

- Since the frozen head of C_2 is $p(0)$, which is in $C_1(D)$, we conclude $C_2 \subseteq C_1$.
  - ✥ Note that the instantiation of C_1 that shows $p(0)$ is in $C_1(D)$ is $X \to 0$, $Y \to 1$, $Z \to 0$, and $W \to 1$.
  - ✥ If we replace 0 and 1 by the variables X and Y they stand for, we have the containment mapping from C_1 to C_2.

Saraiya's Containment Test

- Containment of CQ's is NP-complete in general.
- Sariaya's algorithm is a polynomial-time test of $Q_1 \subseteq Q_2$ for the common case that no predicate appears more than twice among the subgoals of Q_1.
  - ✥ They can appear any number of times in Q_2.
- The algorithm is a reduction to 2SAT and yields a linear-time algorithm.
- Our algorithm is more direct, but quadratic.

The Algorithm

Pick a subgoal of Q_2, and consider the consequences of mapping it to the two possible subgoals of Q_1.

- Follow all consequences of this choice: subgoals that must map to subgoals, and variables that must map to variables.
  - ✥ If we know $p(X_1, \ldots, X_n)$ must map to $p(Y_1, \ldots, Y_n)$, then infer that each X_i must map to Y_i.
  - ✥ If $p(X_1, \ldots, X_n)$ is a subgoal of Q_2, and we know X_i maps to some variable Z, and exactly one of the p-subgoals of Q_1 has Z in the ith component, then conclude $p(X_1, \ldots, X_n)$ maps to this subgoal.

One of two things must happen:

1. We derive a contradiction: a subgoal or variable that must map to two different things.
   - ✥ If so, try the other choice if there is one; fail if there is no other choice.

2. We close the set of inferences we must make.

    - Then we can forever forget about the question of how to map the determined subgoals and variables.

    - We have found one mapping that works and that can't interfere with the mapping of any other subgoals or variables, so we make another arbitrary choice if there are any unmapped subgoals.

Example

Let us test $C_1 \subseteq C_2$, where:

C_1: `p(B) :- a(A,B) & a(B,A) & b(A,C) & b(C,B)`
C_2: `p(X) :- a(X,Y) & b(Y,Z) & b(Z,W) & a(W,X)`

- Note this simple example omits some options: C_1 could have a predicate appearing only once in the body, and C_2 could have 3 or more occurrences of some predicates.

- Here is a description of inferences that might be made:

(1) Suppose $a(X,Y) \rightarrow a(A,B)$
(2) Then $X \rightarrow A$, $Y \rightarrow B$
(3) Now, $b(Y,Z) \rightarrow b(B,?)$
(4) Since there is no $b(B,?)$, fail
(5) Thus, we must map $a(X,Y) \rightarrow a(B,A)$
(6) Then $X \rightarrow B$ and $Y \rightarrow A$,
(7) $b(Y,Z) \rightarrow b(A,C)$, $Z \rightarrow C$,
(8) $b(Z,W) \rightarrow b(C,B)$, $W \rightarrow B$
(9) Now, $a(W,X)$ must map to $a(B,B)$
(10) Since $a(B,B)$ does not exist, fail

- Note, however, that if the last subgoal of C_1 were $b(C,A)$, we would have $W \rightarrow A$ at line (8) and $a(W,X) \rightarrow a(A,B)$ at line (9).

    - That completes the containment mapping successfully, with $X \rightarrow B$, $Y \rightarrow A$, $Z \rightarrow C$, and $W \rightarrow A$.

Generalization to Unions of CQ's

$P_1 \cup P_2 \cup \cdots \cup P_k \subseteq Q_1 \cup Q_2 \cup \cdots \cup Q_n$ iff for all P_i there exists some Q_j such that $P_i \subseteq Q_j$.

Proof (If)

Obvious.

Proof (Only If)

Assume the containment holds.

- Let D be the canonical (frozen) database from CQ P_i.
- Since the containment holds, and $P_i(D)$ surely includes the frozen head of P_i, there must be some Q_j such that $Q_j(D)$ includes the frozen head of P_i.
- Thus, $P_i \subseteq Q_j$.

Union Theorem Just Misses Being False

Consider generalized CQ's allowing arithmetic-comparison subgoals.

P_1: `p(X) :- e(X) & 10 <= X & X <= 20`

Q_1: `p(X) :- e(X) & 10 <= X & X <= 15`
Q_2: `p(X) :- e(X) & 15 <= X & X <= 20`

- $P_1 \subseteq Q_1 \cup Q_2$, but $P_1 \subseteq Q_1$ and $P_1 \subseteq Q_2$ are both false.

CQ Contained in Recursive Datalog

Test relies on method of canonical DB's; containment mapping approach doesn't work (it's meaningless).

- Make DB D from frozen body of CQ.
- Apply program to D. If frozen head of CQ appears in result, then yes (contained), else no.

Example

- CQ Q_1 is:

  Q_1: `path(X,Y) :- arc(X,Z) &`
  `arc(Z,W) & arc(W,Y)`

- Q_2 is the value of *path* in the following recursive Datalog program:

  r_1: `path(X,Y) :- arc(X,Y)`
  r_2: `path(X,Y) :- path(X,Z) & path(Z,Y)`

- Intuitively, Q_1 = paths of length 3; Q_2 = paths of length 1 or more.
- Freeze Q_1, say with 0, 1, 2, 3 as constants for X, Z, W, Y, respectively.

  $D = \{arc(0,1),\ arc(1,2),\ arc(2,3)\}$

- Frozen head is $path(0,3)$.
- Easy to infer that $path(0,3)$ is in $Q_2(D)$ — use r_1 three times to infer $path(0,1)$, $path(1,2)$, $path(2,3)$, then use r_2 to infer $path(0,2)$, $path(0,3)$.

Harder Cases

- Datalog program $\subseteq$ CQ: doubly exponential complexity. Reference: Chaudhuri, S. and M. Y. Vardi [1992]. "On the equivalence of datalog programs," *Proc. Eleventh ACM Symposium on Principles of Database Systems*, pp. 55–66.
- Datalog program $\subseteq$ Datalog program: undecidable.