

Processing Unions of Conjunctive Queries with Negation under Limited Access Patterns

Alan Nash

Department of Mathematics

Bertram Ludäscher

San Diego Supercomputer Center

University of California, San Diego

Data Integration

- **Problem:** Integrate sources with limited query capabilities
- **Example:** Global-as-View (GAV) integration:
 - Given: query Q against global view GV over source schemas S_i :

$$Q(\bar{x}) \leftarrow Q_{GV}(\bar{x}, \bar{y})$$

- Find: query plan P with subqueries Q_i against sources S_i with limited access patterns:

$$Q(\bar{x}) \leftarrow P(Q_1, \dots, Q_n)(\bar{x}, \bar{y})$$

Web Service Composition

- **Problem:** Declarative composition of web services into larger web service “workflows” or “dataflows.”
- **Idea:** Specify composite web service plans as declarative queries, then do query planning over sources with limited access patterns.
- **Example:** Given web service “relations” $WS_1(\text{in } y, \text{out } z)$ and $WS_2(\text{in } x, \text{out } y)$, the declarative query

$$Q(x, z) \leftarrow WS_2(x, y), WS_1(y, z)$$

becomes a web service plan $x \xrightarrow{WS_2} y \xrightarrow{WS_1} z$.

Access Patterns

Assume you have the following interface, which requires you enter an author, a title, or a subject.

author	<input type="text"/>
title	<input type="text"/>
subject	<input type="text"/>
publisher	<input type="text"/>

We model this as a relation $B(a, t, s, p)$ with three access patterns: iooo, oioo, ooio.

Queries Under Access Patterns

What kinds of queries can we answer?

- List all books
- Title and publisher for non-‘Springer’ books by ‘Knuth’
- Books by ‘Knuth’ or ‘Aho’

Classes of Queries

- $Q_1(a, t, s, p) \leftarrow B(a, t, s, p)$
- $Q_2(t, p) \leftarrow \neg B(\text{'Knuth'}, t, s, \text{'Springer'}),$
 $B(\text{'Knuth'}, t, s, p)$
- $Q_3(a, t, p) \leftarrow B(\text{'Knuth'}, t, s, p)$
 $Q_3(a, t, p) \leftarrow B(\text{'Aho'}, t, s, p)$
- $Q_4(a) \leftarrow B(a, t, s, p), L(t), C(b, t)$
 $Q_4(a) \leftarrow B(a, t, s, p), L(t), \neg C(a, t)$

These are CQ, CQ[¬], UCQ, and UCQ[¬] respectively.

Executable Queries

A UCQ[∇] query is *executable* if every variable appears first, positively, in an output slot in the body.

Q_3 can be annotated as follows

- $Q_3(a, t, p) \leftarrow B^{i000}(\text{'Knuth'}, t, s, p)$
 $Q_3(a, t, p) \leftarrow B^{i000}(\text{'Aho'}, t, s, p)$

Q_3 is executable; the rest are not.

Orderable Queries

A UCQ[∇] query is *orderable* if its subgoals can be reordered to get an executable query.

Q_2 can be reordered to get

- $Q'_2(t, p) \leftarrow B^{i000}(\text{'Knuth'}, t, s, p),$
 $\neg B^{i000}(\text{'Knuth'}, t, s, \text{'Springer'})$

Q_2 and Q_3 are orderable; Q_1 and Q_4 are not.

Feasible Queries

A UCQ[∇] query is *feasible* if it is equivalent to an executable UCQ[∇] query.

Q_4 is equivalent to

- $Q'_4(a) \leftarrow L^\circ(t), B^{\text{oi}^\circ\text{oo}}(a, t, s, p)$

Q_2 , Q_3 , and Q_4 are feasible; Q_1 is not.

The Answerable Part

Access Patterns: B^{i000} , B^{oioo} , B^{ooio} , L^o , C^{ii}

- $Q_4(a) \leftarrow B(a, t, s, p), L(t), C(b, t)$
 $Q_4(a) \leftarrow B(a, t, s, p), L(t), \neg C(a, t)$

Bindings: None

- $\text{ans}(Q_4)(a) \leftarrow$

Bindings: None

- $\text{ans}(Q_4)(a) \leftarrow$

The Answerable Part

Access Patterns: B^{i000} , B^{oi00} , B^{ooio} , L^o , C^{ii}

- $Q_4(a) \leftarrow B(a, t, s, p), L(t), C(b, t)$
 $Q_4(a) \leftarrow B(a, t, s, p), L(t), \neg C(a, t)$

Bindings: None

- $\text{ans}(Q_4)(a) \leftarrow L^o(t)$

Bindings: None

- $\text{ans}(Q_4)(a) \leftarrow L^o(t)$

The Answerable Part

Access Patterns: B^{i000} , B^{oioo} , B^{ooio} , L^o , C^{ii}

- $Q_4(a) \leftarrow B(a, t, s, p), L(t), C(b, t)$
 $Q_4(a) \leftarrow B(a, t, s, p), L(t), \neg C(a, t)$

Bindings: t

- $\text{ans}(Q_4)(a) \leftarrow L^o(t)$

Bindings: t

- $\text{ans}(Q_4)(a) \leftarrow L^o(t)$

The Answerable Part

Access Patterns: B^{i1000} , B^{oi100} , B^{ooio} , L^o , C^{ii}

- $Q_4(a) \leftarrow B(a, t, s, p), L(t), C(b, t)$
 $Q_4(a) \leftarrow B(a, t, s, p), L(t), \neg C(a, t)$

Bindings: t

- $\text{ans}(Q_4)(a) \leftarrow L^o(t), B^{oi100}(a, t, s, p)$

Bindings: t

- $\text{ans}(Q_4)(a) \leftarrow L^o(t), B^{oi100}(a, t, s, p)$

The Answerable Part

Access Patterns: B^{i000} , B^{oi00} , B^{ooio} , L^o , C^{ii}

- $Q_4(a) \leftarrow B(a, t, s, p), L(t), C(b, t)$
 $Q_4(a) \leftarrow B(a, t, s, p), L(t), \neg C(a, t)$

Bindings: t, a, s, p

- $\text{ans}(Q_4)(a) \leftarrow L^o(t), B^{oi00}(a, t, s, p)$

Bindings: t, a, s, p

- $\text{ans}(Q_4)(a) \leftarrow L^o(t), B^{oi00}(a, t, s, p)$

The Answerable Part

Access Patterns: B^{i000} , B^{oi00} , B^{ooio} , L^o , C^{ii}

- $Q_4(a) \leftarrow B(a, t, s, p), L(t), C(b, t)$
 $Q_4(a) \leftarrow B(a, t, s, p), L(t), \neg C(a, t)$

Bindings: t, a, s, p

- $\text{ans}(Q_4)(a) \leftarrow L^o(t), B^{oi00}(a, t, s, p)$

Bindings: t, a, s, p

- $\text{ans}(Q_4)(a) \leftarrow L^o(t), B^{oi00}(a, t, s, p), \neg C^{ii}(a, t)$

The Answerable Part

Access Patterns: B^{i000} , B^{oi00} , B^{ooio} , L^o , C^{ii}

- $Q_4(a) \leftarrow B(a, t, s, p), L(t), C(b, t)$
 $Q_4(a) \leftarrow B(a, t, s, p), L(t), \neg C(a, t)$

Bindings: t, a, s, p

- $\text{ans}(Q_4)(a) \leftarrow L^o(t), B^{oi00}(a, t, s, p)$

Bindings: t, a, s, p

- $\text{ans}(Q_4)(a) \leftarrow L^o(t), B^{oi00}(a, t, s, p), \neg C^{ii}(a, t)$

$\text{ans}(Q)$ can be computed efficiently
(in quadratic time).

The Answerable Part

For $Q \in \text{CQ}^-$:

- A subgoal is Q -answerable if there is an executable query E including that subgoal and subgoals from Q .
- If Q is unsatisfiable, then $\text{ans}(Q) := \text{false}$.
- Otherwise, $\text{ans}(Q)$ is the query given by the Q -answerable subgoals in Q .
- Head of $\text{ans}(Q)$: variables in the head of Q and in the body of $\text{ans}(Q)$.

If $Q \in \text{UCQ}^-$ with $Q = Q_1 \cup \dots \cup Q_k$,
then $\text{ans}(Q) := \text{ans}(Q_1) \cup \dots \cup \text{ans}(Q_k)$.

Query Containment

- Q_1 is *contained* in Q_2 ($Q_1 \sqsubseteq Q_2$)
if, for every database \mathcal{D} , $Q_1(\mathcal{D}) \subseteq Q_2(\mathcal{D})$.
- Q_1 is *equivalent* to Q_2 ($Q_1 \equiv Q_2$)
if $Q_1 \sqsubseteq Q_2$ and $Q_2 \sqsubseteq Q_1$.
- Checking containment of CQ or UCQ is NP-complete.
- Checking containment of CQ^\neg or UCQ^\neg is Π_2^P -complete.
- $\Pi_2^P := \text{coNP}^{\text{NP}}$. That is, Π_2^P is what can be computed in coNP with access to an NP oracle.

Outline

- We introduced access patterns.
- We defined *executable*, *orderable*, and *feasible*.
- *Executable* and *orderable* are syntactic notions. *Feasible* is a semantic notion that depends on equivalence.
- We have shown how to compute $\text{ans}(Q)$, the “answerable part” of Q .
- We present our main results.
- We show how to compute the *underestimate query* Q^u and the *overestimate query* Q^o .
- We show how to use Q^u and Q^o at runtime.

Summary of Definitions

- A UCQ^- query is *safe* if every variable appears positively in the body.
- A UCQ^- query is *executable* if every variable appears first, positively, in an output slot in the body.
- A UCQ^- query is *orderable* if its subgoals can be reordered to get an executable query.
- A UCQ^- query is *feasible* if it is equivalent to an executable UCQ^- query.
- $\text{ans}(Q)$ is the *answerable part* of Q .

The Feasibility Problem

Given $Q \in \text{UCQ}^\neg$, determine whether Q is feasible.

- Known: NP-complete for CQ and UCQ .
- We show: Π_2^P -complete for CQ^\neg and UCQ^\neg .

Results

Assume $Q, E \in \text{UCQ}^\neg$.

- $\text{ans}(Q)$ is executable.

Results

Assume $Q, E \in \text{UCQ}^\neg$.

- $\text{ans}(Q)$ is executable.
- Q is feasible iff $\text{ans}(Q) \equiv Q$.

Results

Assume $Q, E \in \text{UCQ}^\neg$.

- $\text{ans}(Q)$ is executable.
- Q is feasible iff $\text{ans}(Q) \equiv Q$.
- That is, UCQ^\neg feasibility reduces to containment.

Results

Assume $Q, E \in \text{UCQ}^\neg$.

- $\text{ans}(Q)$ is executable.
- Q is feasible iff $\text{ans}(Q) \equiv Q$.
- That is, UCQ^\neg feasibility reduces to containment.
- Feasibility of CQ^\neg and UCQ^\neg : Π_2^P -complete.

Results

Assume $Q, E \in \text{UCQ}^\neg$.

- $\text{ans}(Q)$ is executable.
- Q is feasible iff $\text{ans}(Q) \equiv Q$.
- That is, UCQ^\neg feasibility reduces to containment.
- Feasibility of CQ^\neg and UCQ^\neg : Π_2^P -complete.
- If $Q \sqsubseteq E$ and E is executable, then $Q \sqsubseteq \text{ans}(Q) \sqsubseteq E$.

That is, if there is a minimal executable query containing Q , it is equivalent to $\text{ans}(Q)$.

Compile-time vs. Runtime

- At compile time (we have the query Q but no database D) we can check whether Q is feasible.
- At runtime (we have the query Q and a database D) we can check whether we have $Q(D)$, regardless of whether Q is feasible.
- If we do not have exactly $Q(D)$, we can often quantify how close we are.

Under- and Overestimates

- Access patterns: $S^o, R^{oo}, B^{ii}, T^{oo}$
- $Q(x, y) \leftarrow \neg S(y), R(x, y), B(x, z)$
 $Q(x, y) \leftarrow T(x, y)$
- Underestimate:
 $Q^u(x, y) \leftarrow T(x, y)$
- Overestimate:
 $Q^o(x, y) \leftarrow R(x, y), \neg S(y)$
 $Q^o(x, y) \leftarrow T(x, y)$

Example

- $Q(x, y) \leftarrow \neg S(y), R(x, y), B(x, z)$
 $Q(x, y) \leftarrow T(x, y)$
- $Q^u(x, y) \leftarrow T(x, y)$
- $Q^o(x, y) \leftarrow R(x, y), \neg S(y)$
 $Q^o(x, y) \leftarrow T(x, y)$

T		R		S		B	
	1	2		1	2	5	
	3	4		5	6	7	
							1 8
							3 9
Q^u		Q^o		Q			
	1	2		1	2		1 2
	3	4		3	4		3 4
		Δ		5	6		

Algorithm Answer*

```
procedure ANSWER*(Q)
  (Qu, Qo) := PLAN*(Q)
  ansu := ANSWER(Qu, D)
  anso := ANSWER(Qo, D)
  Δ := anso \ ansu
  output ansu
  if Δ = ∅ then output "answer is complete"
  else
    output "answer not known complete;"
    output "possibly part of the answer:"
    output Δ
    if Δ has no null values then
      output "answer at least"  $\frac{|ans_u|}{|ans_o|}$  "complete"
```

Algorithm Plan*

```
procedure PLAN*(Q)
  for  $i := 1$  to  $n$  do
     $A_i := \text{ANSWERABLE}(Q_i, \mathcal{P})$ 
     $U_i := Q_i \setminus A_i$ 
     $Q_i^u := \begin{cases} A_i & \text{if } U_i = \emptyset \\ \perp & \text{otherwise} \end{cases}$ 
     $\bar{v} := \bar{x} \setminus \text{vars}(A_i)$ 
     $Q_i^o := A_i$  and  $(\bar{v} = \overline{\text{null}})$ 
   $Q^u := Q_1^u \vee \dots \vee Q_n^u$ 
   $Q^o := Q_1^o \vee \dots \vee Q_n^o$ 
  output  $Q^u, Q^o$ 
```

Algorithm Feasible*

```
procedure FEASIBLE( $Q$ )  
  ( $Q^u, Q^o$ ) := PLAN*( $Q$ )  
  if  $Q^u = Q^o$  then return true  
  else if  $Q^o$  contains null then return false  
  else return  $Q^o \sqsubseteq Q$ 
```

Summary

- Feasibility of UCQ^\neg : Π_2^P -complete.
- Feasibility of CQ^\neg : Π_2^P -complete.
- $\text{ans}(Q)$: minimal executable query containing Q .
- Unified algorithm for CQ , UCQ , CQ^\neg , UCQ^\neg .
- Runtime approximations.

Future Work

- Beyond UCQ[⊖]: FO (PODS 2004)
- Integrity constraints
- Views with access patterns
- Recursion