

CSE 6339 Web Search, Mining, and Integration

Answering Queries Using Views

(based on slides from Michalis Petropoulos @ SUNY-Buffalo)

Motivation

Local-As-View (LAV) Integration Approach

Data sources are described as views over the global schema

Global schema: ForSale(name, year, country, category)
Review(product, review, category)

French cars data source:

V1(name, year) :- ForSale(name, year, "France", "auto"),
year > 1990

Car review database:

V2(product, review) :- Review(product, review, "auto")

Query: q(X,Y,R):- ForSale(X,Y,C,"auto"),
Review(X,R,"auto"), Y > 1985

2

LAV Assumptions

1. There is a set of predicates that define the **global schema**
 - These do not exist as stored relations
2. Each data source has its capabilities defined by **views**, which are conjunctive queries (CQs) whose subgoals involve the global predicates
3. A **query** is a CQ over the global predicates
4. A **rewriting** is an expression (union of CQ's) involving the views
 - Ideally, the rewriting is equivalent to the query
 - In practice, we have to be happy with a rewriting maximally contained in the query

3

Interpretation of Views

- A view describes **some of the facts** that are available at the source
- A view does **not** define exactly what is at the source
 - Example: View V2(p, r) :- Review(p, r, "auto") says that the source has **some** Review-facts with third component "auto", **not all** of them
 - V2 could even be empty although Review(p, r, "auto") is not

4

Interpretation of Views (2)

In other words:

- The :- separator between head and body of a view definition should not be interpreted as "if"
- Rather, it is "only if"

5

Rewriting

French cars data source:

V1(name, year) :-
ForSale(name, year, "France", "auto"), year > 1990

Car review database:

V2(product, review) :- Review(product, review, "auto")

Query: q(X,Y,R):- ForSale(X,Y,C,"auto"),
Review(X,R,"auto"), Y > 1985.

Query rewriting: q'(X,Y,R) :- V1(X,Y), V2(X,R)

Note: Rewriting is not equivalent to the query, but we can't do any better

6

Formal Definition: Rewriting

Given a query Q and a set of view definitions V_1, \dots, V_n :

Q' is a **rewriting** of the query using V 's if it refers *only* to the views or to arithmetic predicates

Q' is an **equivalent rewriting** of Q using the V 's if Q' is equivalent to Q

Q' is a **maximally-contained rewriting** of Q w.r.t. L using the V 's if there is no other Q'' such that Q'' strictly contains Q' , and Q'' is contained in Q

7

Usability Conditions for Views

Query: $q(X,Z) :- r(X,Y), s(Y,Z), t(X,Z), Y > 5$

What can go wrong?

$V_1(A,B) :- r(A,C), s(C,B)$ (join predicate not applied)

$V_2(A,B) :- r(A,C), s(C,B), C > 1$ (predicate too weak)

$V_3(A) :- r(A,B), s(B,C), t(A,C), B > 5$:
needed argument is projected out. Can be recovered if we have a functional dependency $t: A \rightarrow C$

8

What Makes a Rewriting R Useful?

1. There must be no other rewriting containing R
2. When views in R are unfolded into global predicates, R is contained in the original query

9

Important Points

- To test containment of a rewriting in a query, we unfold the views in the rewriting first, then test CQ containment of the unfolding in the query
- The view definition describes what any tuples of the view look like, so CQ containment implies that the rewriting will provide only true answers

10

The Picture

Query: $q(X,Y) :- \text{sameTopic}(X,Y), \text{cites}(X,Y), \text{cites}(Y,X)$

Query rewriting: $q'(X,Y) :- V_1(X,Y), V_2(X,Y)$

Unfolding of the rewriting:
 $q''(X,Y) :- \text{cites}(X,Y), \text{cites}(Y,X),$
 $\text{sameTopic}(X,Y), \text{cites}(X,Z), \text{cites}(Y,W)$

Is there a containment mapping?

11

Important Points (2)

- There is no guarantee a rewriting supplies **any** answers to the query
- Comparing different rewritings by testing if one rewriting is contained in another must be done at the level of the folded views

12

Example

- Two sources might have similar views, defined by:
 $V2(C,D) :- \text{sameTopic}(C,D), \text{cites}(C,C1), \text{cites}(D,D1)$
 $V3(E,F) :- \text{sameTopic}(E,F), \text{cites}(E,E1), \text{cites}(F,F1)$
- But the sources actually have different sets of tuples

13

Example - Continued

- Then, the two rewritings:
 $q'(X,Y) :- V1(X,Y), V2(X,Y)$
 $q''(X,Y) :- V1(X,Y), V3(X,Y)$
have the same unfolding, but there is no reason to believe one rewriting is contained in the other
- One view could provide lots of tuples, the other, few or none

14

Important Points (3)

- On the other hand, when one rewriting, **folded**, is contained in another, we can be sure the first provides no answers the second does not

15

Example

- Here are two rewritings:
 $q'(X,Y) :- V1(X,Y), V2(X,Y)$
 $q''(X, \text{WSDL}) :- V1(X, \text{WSDL}), V2(X, \text{WSDL})$
- There is a containment mapping $q' \rightarrow q''$
 - Thus, $q'' \subseteq q'$ at the level of views
- No matter what tuples $V1$ and $V2$ represent, q' provides all answers q'' provides

16

Finding All Rewritings

- For conjunctive queries with no arithmetic predicates, the following holds:
If Q has an equivalent rewriting using V , then there exists one with no more conjuncts than Q [Levy, Mendelzon, Sagiv & Srivastava, PODS 95]
 - The rewriting problem is NP-complete
 - **Maximally-contained rewriting**: union of all conjunctive rewritings of the length of the query or less
- LMSS Test:**
- If a query has n subgoals, then we only need to consider rewritings with at most n subgoals
 - Any other rewriting must be contained in one with $\leq n$ subgoals

17

A Naive Algorithm

- Consider all rewrites containing up to as many views as Q has subgoals
- Test each unfolding for containment in Q
- Take the union of the contained ones
- Exponential and brute force
- Makes use of the LMSS test
- Can we do better?

18

Practical Algorithms

- **Bucket Algorithm**
- Inverse Rules Algorithm
- MINICON Algorithm
- Excellent survey:
 - **Answering Queries Using Views: A Survey**
 - By Alon Halevy
 - *VLDB Journal, 2000*
 - <http://citeseer.ist.psu.edu/halevy00answering.html>

19

The Bucket Algorithm

- Each subgoal g of Q must be "covered" by some view
- Make a list of candidates (buckets) per query subgoal
- Consider combinations of candidates from different buckets
- Not all combos are "compatible"
- Keep the compatible ones and minimize them
- Discard the ones contained in another
- Take their union

20

The Bucket Algorithm

$q(X,Y,R) :- \text{ForSale}(X,Y,C,\text{"auto"}), \text{Review}(X,R,\text{"auto"}), Y > 1985$

Step 1: For each subgoal, put the relevant sources into a bucket:

$V1(\text{name}, \text{year}) :- \text{ForSale}(\text{name}, \text{year}, \text{"France"}, \text{"auto"}), \text{year} > 1990$ **would be relevant**

$V3(\text{name}, \text{year}) :- \text{ForSale}(\text{name}, \text{year}, \text{"France"}, \text{"cheese"})$ **would be irrelevant**

Step 2: Take the Cartesian product of the buckets

Algorithm produces maximally contained rewriting

Ignores interactions between subgoals in Step 1

21

The Bucket Algorithm: Example

Mediated Schema:

$\text{Enrolled}(\text{std}, \text{dept})$ $\text{Registered}(\text{std}, \text{crs}, \text{yr})$ $\text{Course}(\text{crs}, \text{num})$

Views (data sources):

$V1(\text{std}, \text{num}, \text{yr}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{Course}(\text{crs}, \text{num}), \text{num} \geq 500, \text{yr} \geq 1992.$

$V2(\text{std}, \text{dept}, \text{crs}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{Enrolled}(\text{std}, \text{dept}).$

$V3(\text{std}, \text{crs}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{yr} \leq 1990.$

$V4(\text{std}, \text{crs}, \text{num}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{Course}(\text{crs}, \text{num}), \text{Enrolled}(\text{std}, \text{dept}), \text{num} \leq 100.$

Query:

$q(S,D) :- \text{Enrolled}(S, D), \text{Registered}(S, C, Y), \text{Course}(C, N), N \geq 300, Y \geq 1995$

Step 1: For each query subgoal, put the relevant sources into a bucket

22

The Bucket Algorithm: Example

$V1(\text{std}, \text{num}, \text{yr}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{Course}(\text{crs}, \text{num}), \text{num} \geq 500, \text{yr} \geq 1992.$

$V2(\text{std}, \text{dept}, \text{crs}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{Enrolled}(\text{std}, \text{dept}).$

$V3(\text{std}, \text{crs}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{yr} \leq 1990.$

$V4(\text{std}, \text{crs}, \text{num}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{Course}(\text{crs}, \text{num}), \text{Enrolled}(\text{std}, \text{dept}), \text{num} \leq 100.$

$q(S,D) :- \text{Enrolled}(S, D), \text{Registered}(S, C, Y), \text{Course}(C, N), N \geq 300, Y \geq 1995$

$S \rightarrow \text{std}, D \rightarrow \text{dept}$

Buckets		
$\text{Enrolled}(S,D)$	$\text{Registered}(S,C,Y)$	$\text{Course}(C,N)$
$V2(S,D,C)$		
	$V4(S,C,N)$	

Note: Arithmetic predicates don't pose a problem. (Why?)

23

The Bucket Algorithm: Example

$V1(\text{std}, \text{num}, \text{yr}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{Course}(\text{crs}, \text{num}), \text{num} \geq 500, \text{yr} \geq 1992.$

$V2(\text{std}, \text{dept}, \text{crs}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{Enrolled}(\text{std}, \text{dept}).$

$V3(\text{std}, \text{crs}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{yr} \leq 1990.$

$V4(\text{std}, \text{crs}, \text{num}) :- \text{Registered}(\text{std}, \text{crs}, \text{yr}), \text{Course}(\text{crs}, \text{num}), \text{Enrolled}(\text{std}, \text{dept}), \text{num} \leq 100.$

$q(S,D) :- \text{Enrolled}(S, D), \text{Registered}(S, C, Y), \text{Course}(C, N), N \geq 300, Y \geq 1995$

$S \rightarrow \text{std}, C \rightarrow \text{crs}, Y \rightarrow \text{yr}$

Buckets		
$\text{Enrolled}(S,D)$	$\text{Registered}(S,C,Y)$	$\text{Course}(C,N)$
$V2(S,D,C)$	$V1(S,N,Y)$	
$V4(S,C,N)$	$V2(S,D,C)$	
	$V4(S,C,N)$	

Note: * $V3$ doesn't work: arithmetic predicates not consistent
 * But why is $V4$ included? (because num is not in the unifying mapping.)

24

The Bucket Algorithm: Example

$V1(std,num,yr) :- Registered(std, crs, yr), Course(crs, num), num \geq 500, yr \geq 1992.$
 $V2(std,dept,crs) :- Registered(std, crs, yr), Enrolled(std, dept).$
 $V3(std,crs) :- Registered(std, crs, yr), yr \leq 1990.$
 $V4(std,crs,num) :- Registered(std, crs, yr), Course(crs, num), Enrolled(std, dept), num \leq 100.$

$q(S,D) :- Enrolled(S, D), Registered(S, C, Y), Course(C, N), N \geq 300, Y \geq 1995$

Buckets

Enrolled(S,D)	Registered(S,C,Y)	Course(C,N)
V2(S,D,C)	V1(S,N,Y)	V1(S,N,Y)
V4(S,C,N)	V2(S,D,C)	
	V4(S,C,N)	

$C \rightarrow crs, N \rightarrow num$

Note: Why did we exclude v4?

25

The Bucket Algorithm: Example

Step 2:

- Try all combos of views, one each from a bucket
- Test satisfaction of arithmetic predicates in each case
 - e.g., two views may not overlap, i.e., they may be inconsistent
- Desired rewriting = union of surviving ones

Query rewriting:

Enrolled(S,D)	Registered(S,C,Y)	Course(C,N)
V2(S,D,C)	V1(S,N,Y)	V1(S,N,Y)
V4(S,C,N)	V2(S,D,C)	
	V4(S,C,N)	

$q'(S,D) :- V1(S,N,Y), V2(S,D,C), Y \geq 1995$

How about $q''(S,D) :- V1(S,N,Y), V4(S,C,N), Y \geq 1995$?

26

Questions?

27