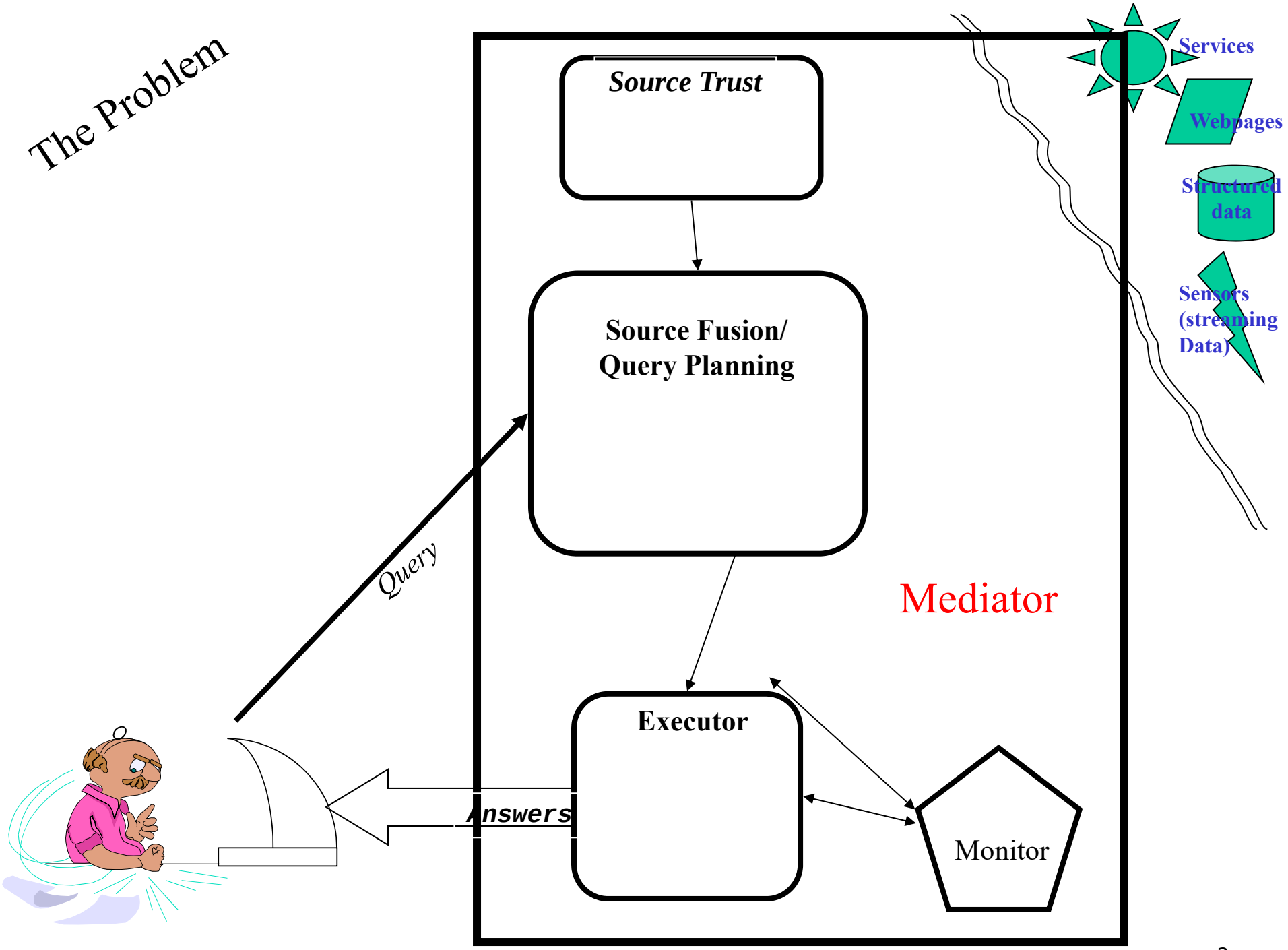
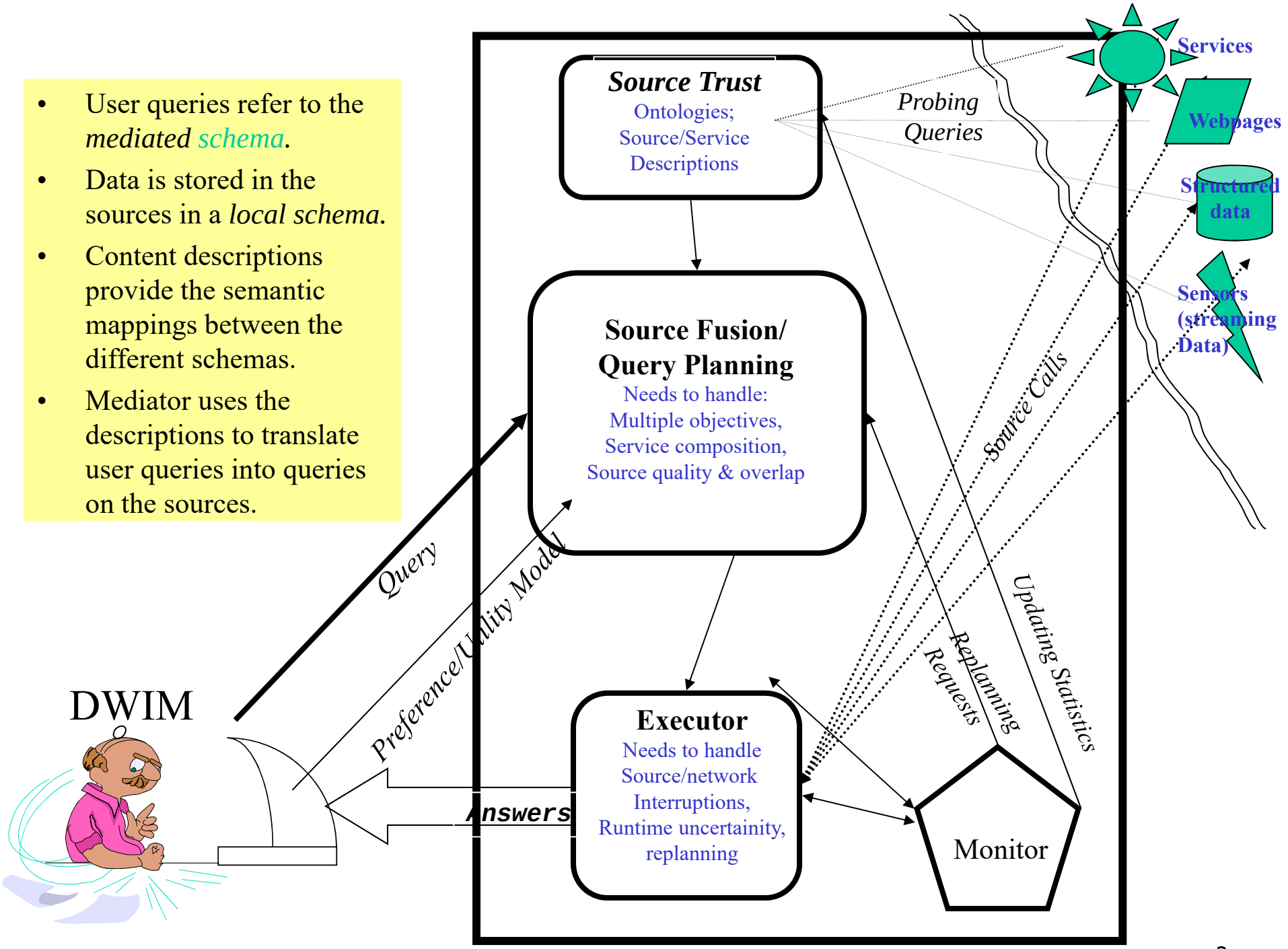
# Information Integration (Semantic Web done in Bottom-up manner)

)

# The Problem


Source Trust
Source Fusion/
Query Planning
Executor
Mediator
Monitor
Query
Answers
Services
Webpages
Structured
data
Sensors
(streaming
Data)

- User queries refer to the *mediated schema*.
- Data is stored in the sources in a *local schema*.
- Content descriptions provide the semantic mappings between the different schemas.
- Mediator uses the descriptions to translate user queries into queries on the sources.



Skeptic's corner

# Who is dying to have it? (Applications)

- WWW:
  - Comparison shopping
  - Portals integrating data from multiple sources
  - B2B, electronic marketplaces
- Science and culture:
  - Medical genetics: integrating genomic data
  - Astrophysics: monitoring events in the sky.
  - Culture: uniform access to all cultural databases produced by ~~countries in Europe~~ provinces in Canada
- Enterprise data integration
  - An average company has 49 different databases and spends 35% of its IT dollars on integration efforts

# 4/13: Data Integration (contd)

At Home Exam Stats

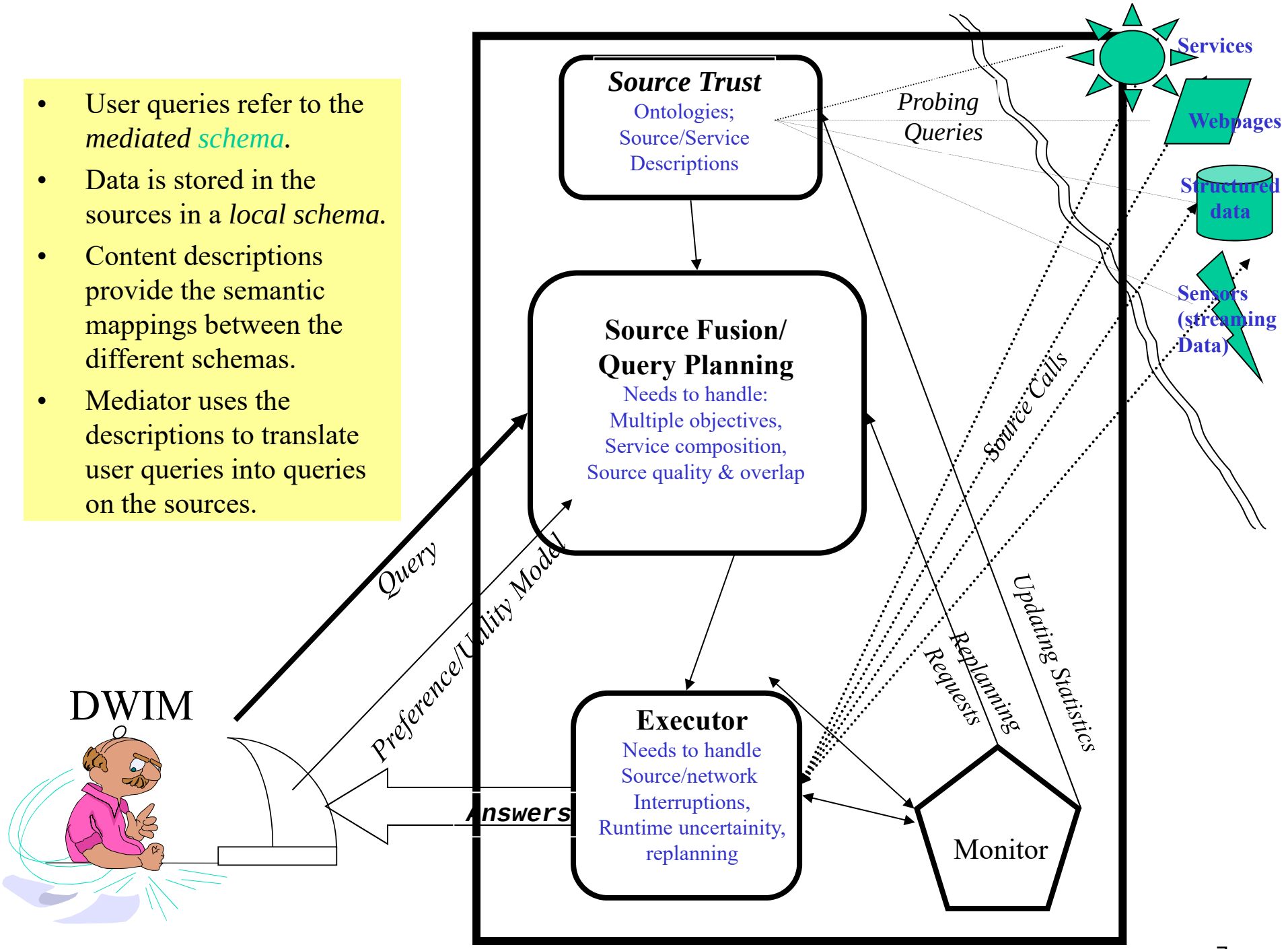
77.5/51.8/22

(61.5/38/13 for in-class)

Project B stats

100/68/34

- User queries refer to the *mediated schema*.
- Data is stored in the sources in a *local schema*.
- Content descriptions provide the semantic mappings between the different schemas.
- Mediator uses the descriptions to translate user queries into queries on the sources.



# Big Digression

The popularity of Web brings two broad challenges to Databases

- Integration of autonomous data sources
  - Data/information integration
  - Technically has to handle heterogeneous data too
    - But we will sort of assume that the sources are "quasi-relational"

I hope to do some of the DB/IR stuff once I complete the discussion of Information Integration stuff (which, in principle, subsumes the DB/IR stuff..)

- Supporting heterogeneous data (combining DB/IR)
  - This can be tackled in the presence of a single database
  - The issues are
    - How to do effective querying in the presence of structured and text data
      - E.g. Stuff I have Seen project
    - How to support IR-style querying on DB
      - Because users seem to know IR/keyword style querying more
      - (notice the irony here—we said structure is good because it supports structured querying)
    - How to support imprecise queries

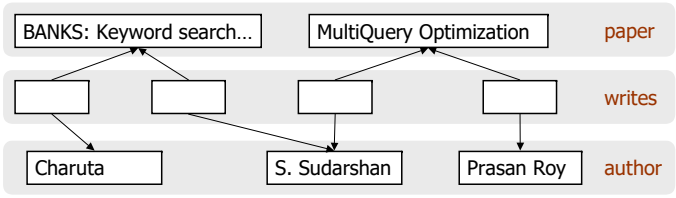
# BANKS: Keyword Search in DB

## Motivation

- Keyword search of documents on the Web has been enormously successful
  - Simple and intuitive, no need to learn any query language
- Database querying using keywords is desirable
  - SQL is not appropriate for casual users
  - Form interfaces cumbersome:
    - Require separate form for each type of query — confusing for casual users of Web information systems
    - Not suitable for ad hoc queries

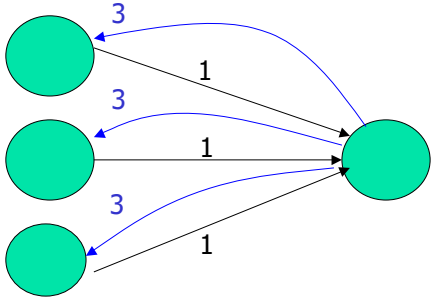
## Basic Model

- Database: modeled as a graph
  - Nodes = tuples
  - Edges = references between tuples
    - foreign key, other kind of relationships
    - Edges are directed.




## Edge Weight

- Weight of forward edge based on schema
  - e.g. citation link weights > writes link weights
- Weight of backward edge = indegree of edges pointing to the node




## BANKS Query Result Example

- Result of "Soumen Sunita"

Table = PAPER

| PAPERID | TITLE | YEAR |
|---|---|---|
| ChakrabartiSD98 | Mining Surprising Patterns Using Temporal Description Length. | |

Table = WRITES

| NAME | PAPERID |
|---|---|
| Soumen Chakrabarti | ChakrabartiSD98 |

Table = AUTHOR

| NAME | URL |
|---|---|
| Soumen Chakrabarti | |

Table = WRITES

| NAME | PAPERID |
|---|---|
| Sunita Sarawagi | ChakrabartiSD98 |

Table = AUTHOR

| NAME | URL |
|---|---|
| Sunita Sarawagi | |

Skeptic's corner

# Why isn't this just



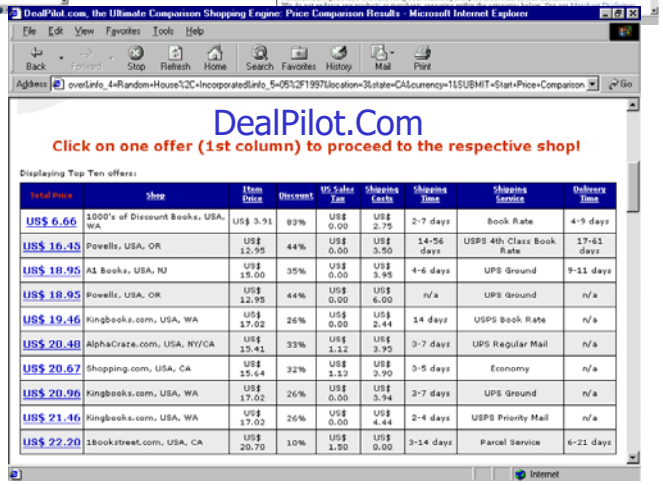
- Search engines do text-based retrieval of URLS
  - Works reasonably well for single document texts, or for finding sites based on single document text
    - Cannot integrate information from multiple documents
    - Cannot do effective "query relaxation" or generalization
    - Cannot link documents and databases
- The aim of Information integration is to support query processing over structured and semi-structured sources as well as services.

Skeptic's corner

# Are we talking "comparison shopping" agents?

- Certainly closer to the aims of these
- But:
  - Wider focus
    - Consider larger range of databases
    - Consider services
  - Implies more challenges
    - "warehousing" may not work
    - Manual source characterization/ integration won't scale-up





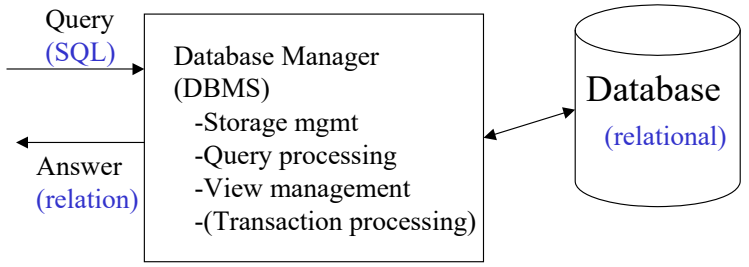
# Is it like Expedia/Travelocity/Orbitz…

- Surpringly, NO!
- The online travel sites don't quite need to do data integration; they just use SABRE
  - SABRE was started in the 60's as a joint project between American Airlines and IBM
  - It is the *de facto* database for most of the airline industry (who *voluntarily* enter their data into it)
    - There are very few airlines that buck the SABRE trend—SouthWest airlines is one (which is why many online sites don't bother with South West)
- So, online travel sites really are talking to a *single* database (not multiple data sources)…
  - To be sure, online travel sources do have to solve a hard problem. Finding an optimal fare (even at a given time) is basically computationally intractable (not to mention the issues brought in by fluctuating fares). So don't be so hard on yourself
    - Check out http://www.maa.org/devlin/devlin_09_02.html

Skeptic's corner

# Why isn't this just Databases / Distributed Databases

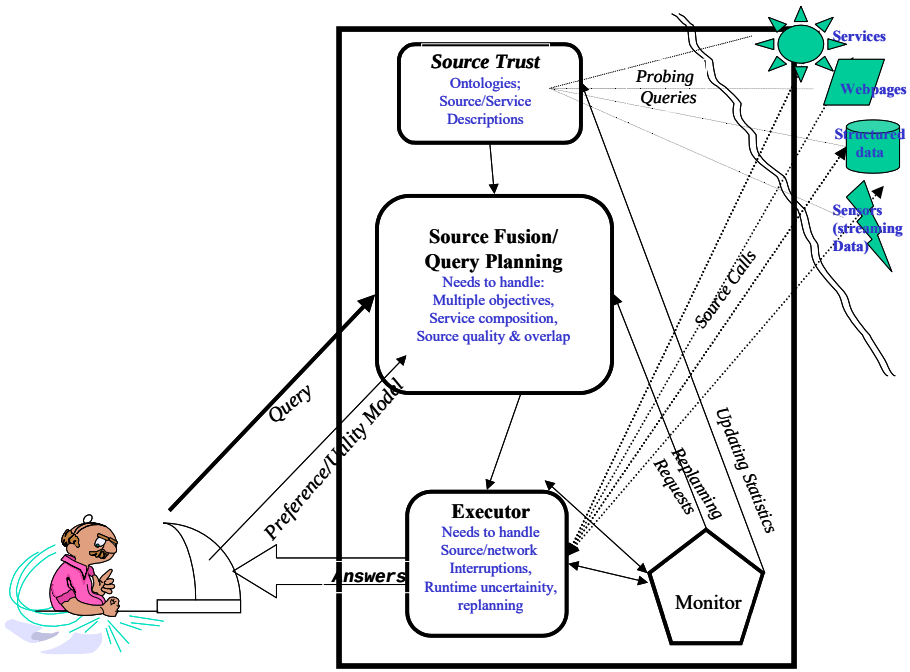
- No common schema
  - Sources with heterogeneous schemas (and ontologies)
  - Semi-structured sources
- Legacy Sources
  - Not relational-complete
  - Variety of access/process limitations
- Autonomous sources
  - No central administration
  - Uncontrolled source content overlap
- Unpredictable run-time behavior
  - Makes query execution hard
- Predominantly "Read-only"
  - Could be a blessing—less worry about transaction management
  - (although the push now is to also support transactions on web)



# Issues in Information Integration

# Overview

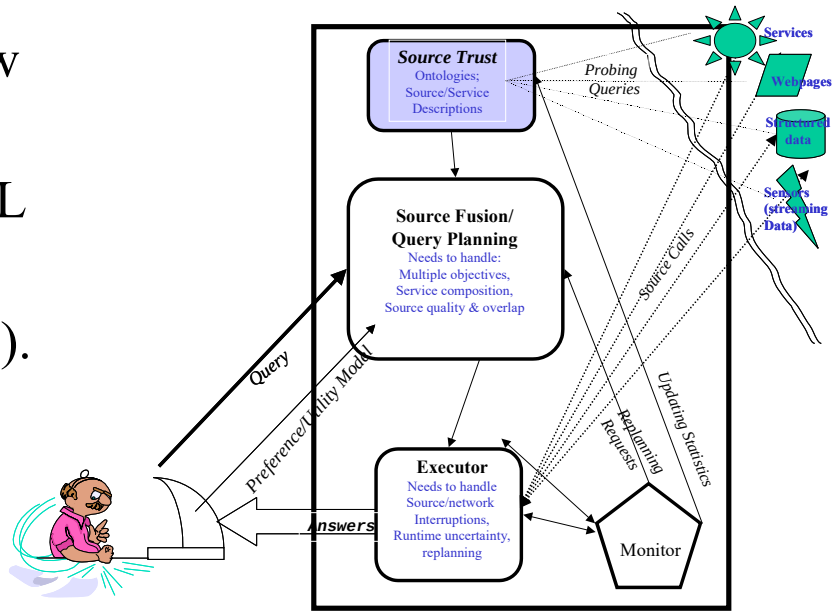
- User queries refer to the *mediated schema*.
- Data is stored in the sources in a *local schema*.
- Content descriptions provide the semantic mappings between the different schemas.
- Mediator uses the descriptions to translate user queries into queries on the sources.




*Schema: Template for the stored data*

# Source Descriptions

- Contains all meta-information about the sources:
  - Logical source contents (books, new cars).
  - Source capabilities (can answer SQL queries)
  - Source completeness (has *all* books).
  - Physical properties of source and network.
  - Statistics about the data (like in an RDBMS)
  - Source reliability
  - Mirror sources
  - Update frequency.



# 4/15

→Project 2 returned
→Current grade sheet put up
→Many slides added for 4/13

→Questions???

**Image 2 – Algorithm Menu**

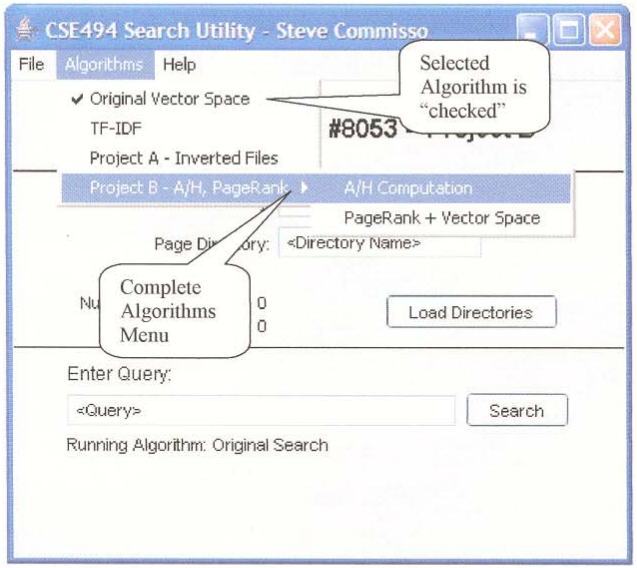
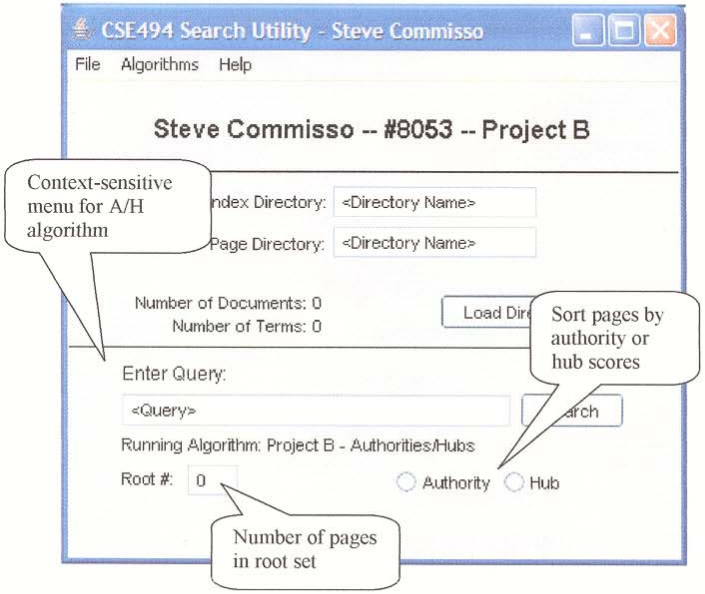
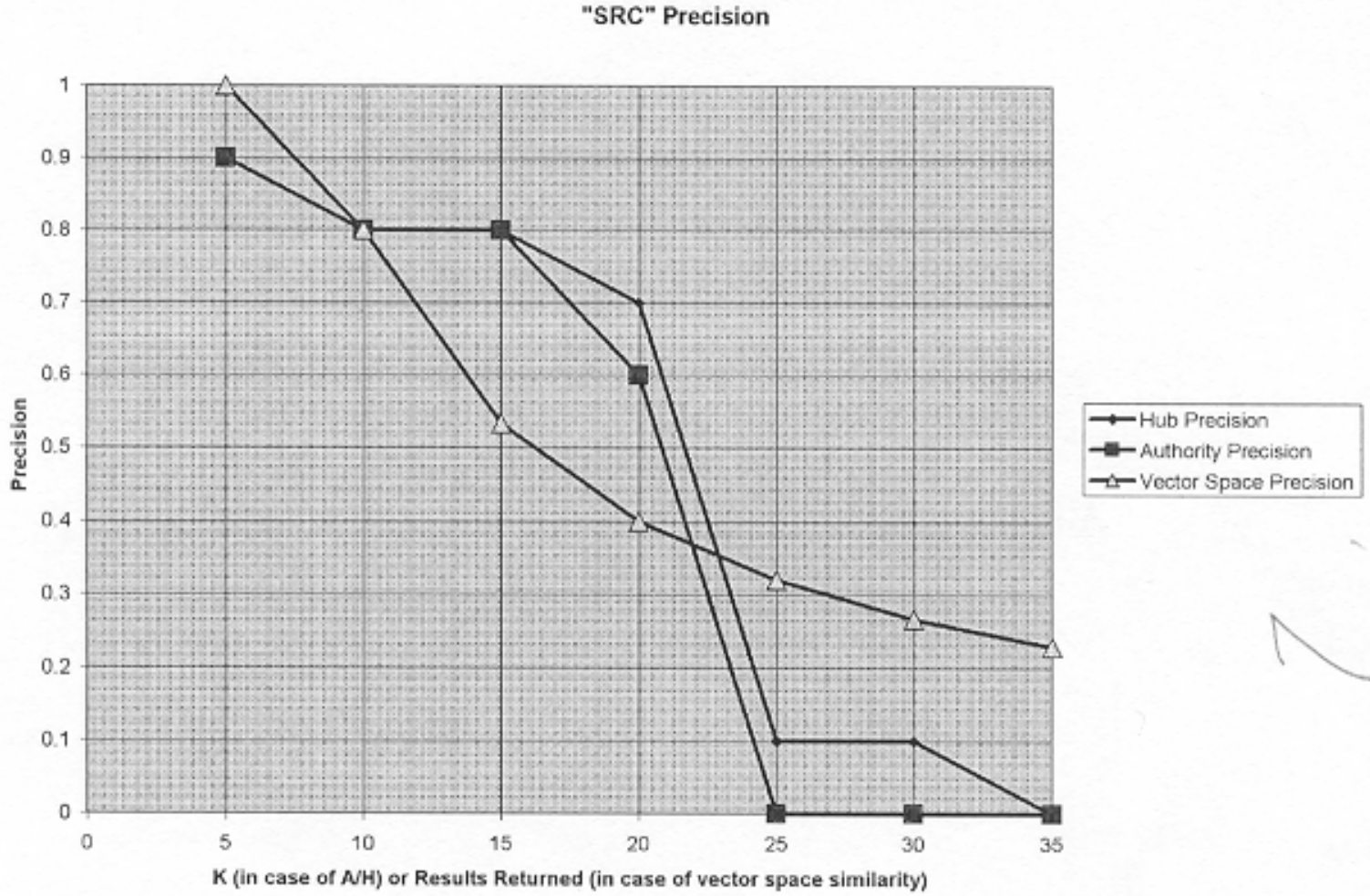



Image 3 – Authorities/Hubs Menu




"SRC" Precision
Precision
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
5
10
15
20
25
30
35
K (in case of A/H) or Results Returned (in case of vector space similarity)
Hub Precision
Authority Precision
Vector Space Precision

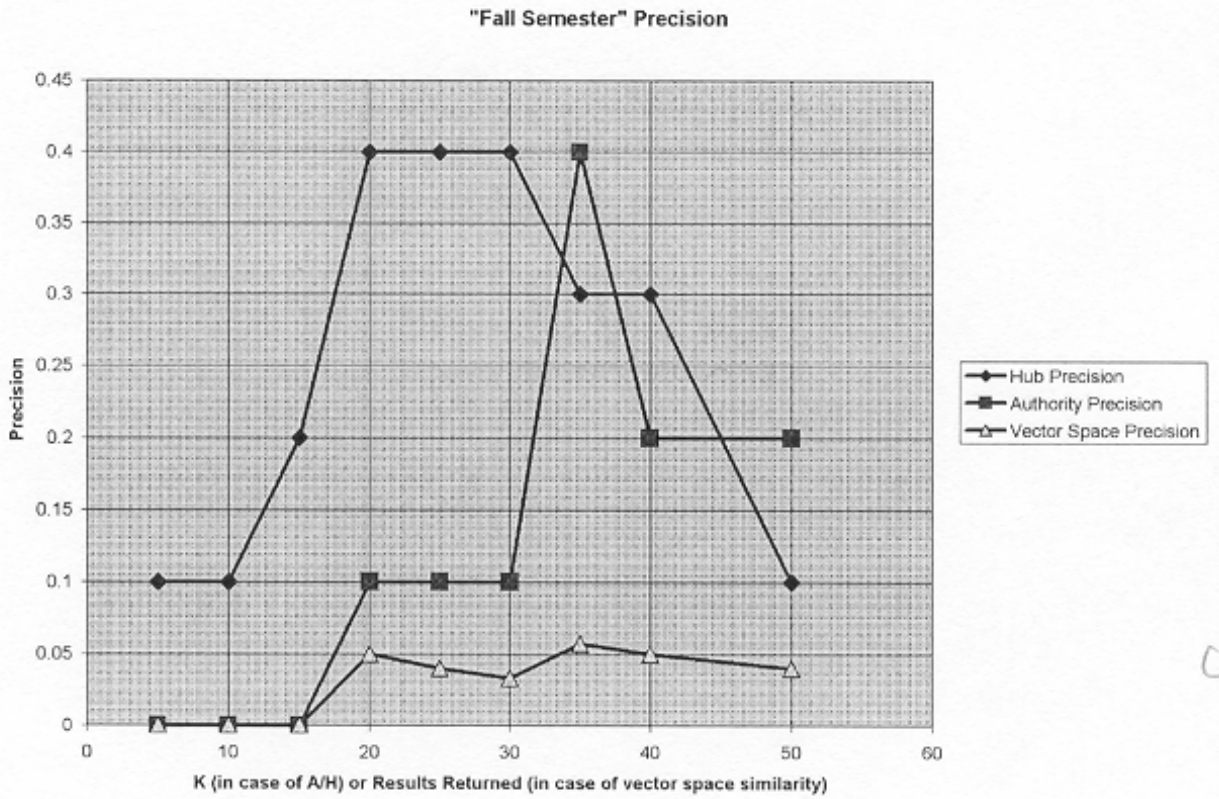



Figure 4: Search results are shown in the large text box below the top panel.

documents more often[1] Also, when A/H fails to return any relevant documents PageRank succeeds.

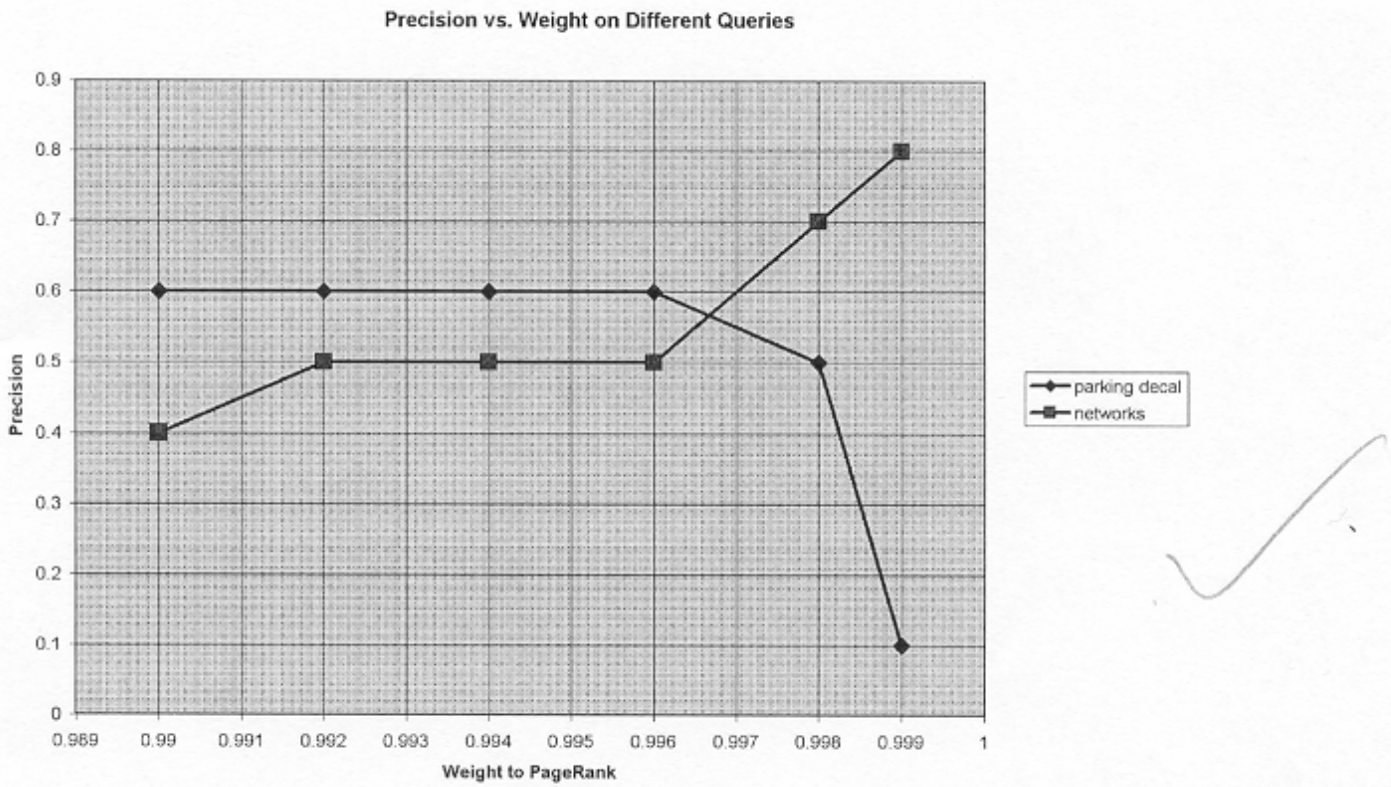



Figure 5: On different queries, the same weight given to PageRank has a different effect.

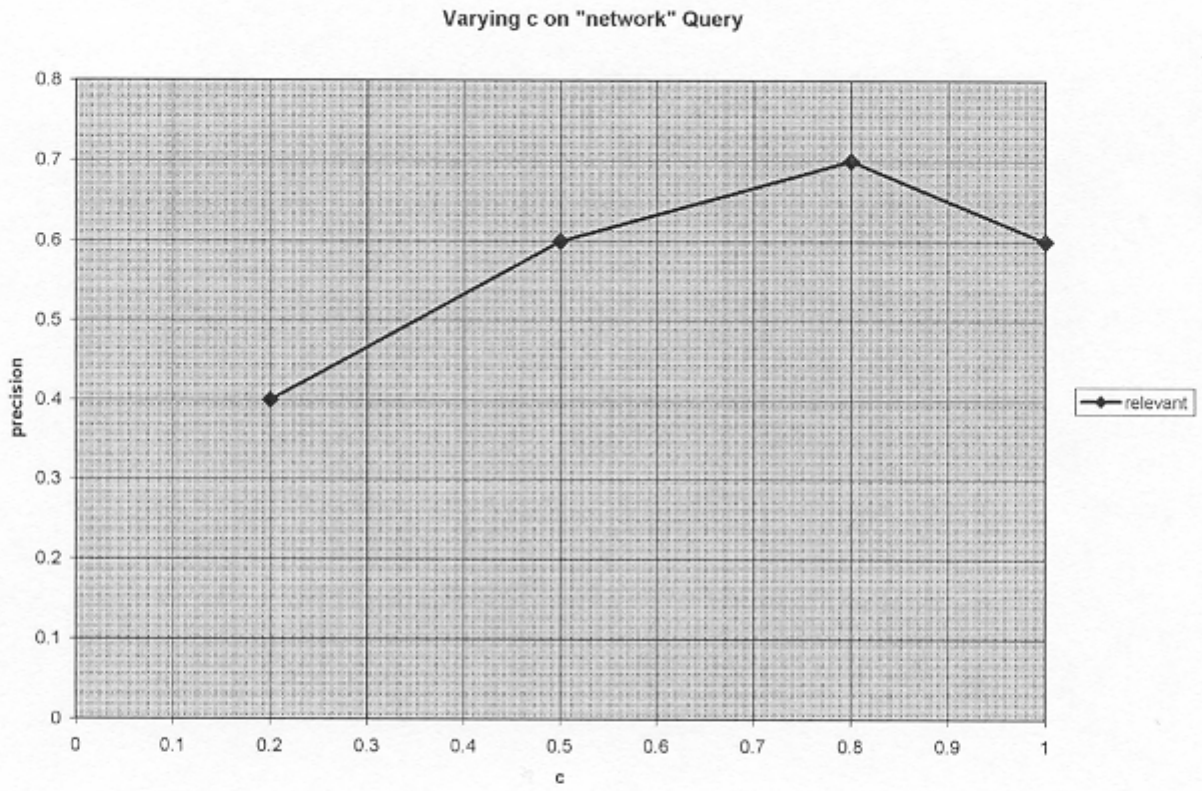


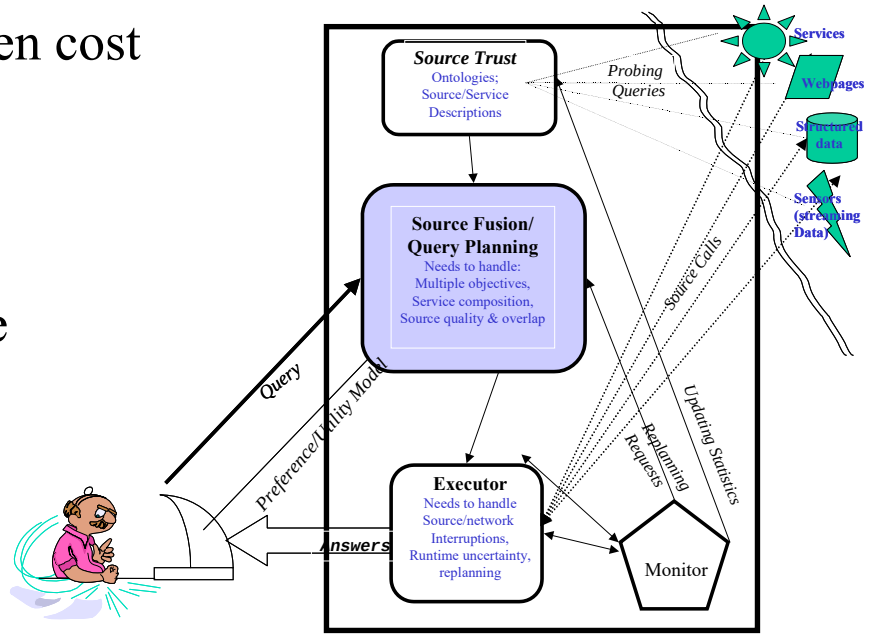

Figure 6: The random surfer matrix (K), when given a greater weight (i.e. when "c" is small), the precision of the results is poor. It is the case, however, that the matrix provides some precision when given some weight (i.e. when "c" is larger but not "too large").

# Source Access

- How do we get the “tuples”?
  - Many sources give “unstructured” output
    - Some inherently unstructured; while others “englishify” their database-style output
  - Need to (un)Wrap the output from the sources to get tuples
  - “Wrapper building”/Information Extraction
  - Can be done manually/semi-manually

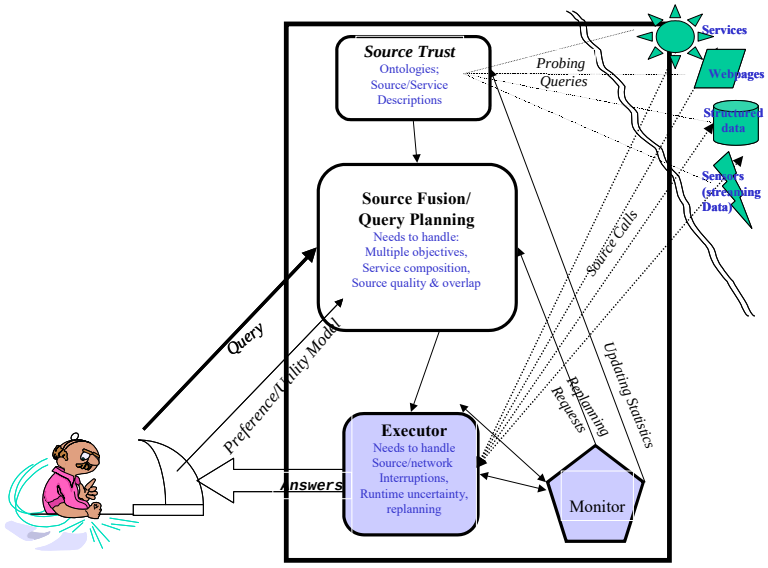
# Source Fusion/Query Planning

- Accepts user query and generates a plan for accessing sources to answer the query
  - Needs to handle tradeoffs between cost and coverage
  - Needs to handle source access limitations
  - Needs to reason about the source quality/reputation



# Monitoring/Execution

- Takes the query plan and executes it on the sources
  - Needs to handle source latency
  - Needs to handle transient/short-term network outages
  - Needs to handle source access limitations
  - May need to re-schedule or re-plan



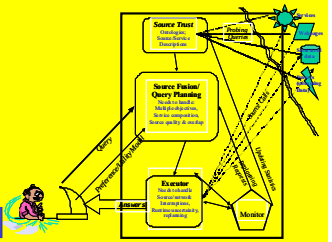
# Dimensions to Consider

- How many sources are we accessing?
- How autonomous are they?
- Can we get meta-data about sources?
- Is the data structured?
- Supporting just queries or also updates?
- Requirements: accuracy, completeness, performance, handling inconsistencies.
- Closed world assumption vs. open world?

# Models for Integration

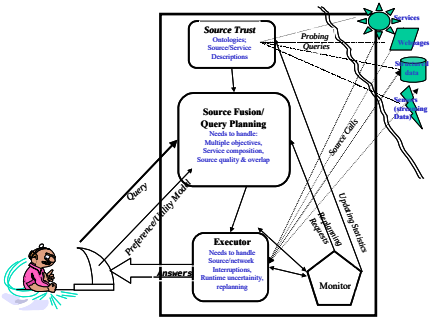
## Overview

- Motivation for Information Integration [Rao]
- Accessing Information Sources [Craig]
- Models for Integration [Rao]
- Query Planning & Optimization [Rao]
- Plan Execution [Craig]
- Standards for Integration/Mediation [Rao]
- Ontology & Data Integration [Craig]
- Future Directions [Craig]

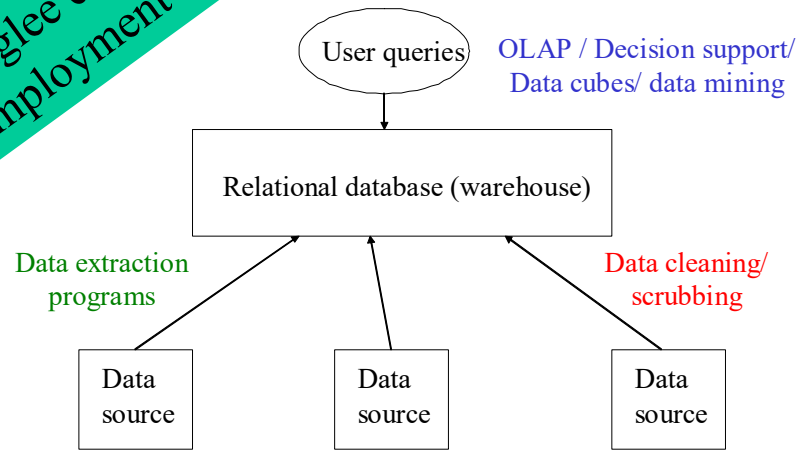


# Solutions for small-scale integration

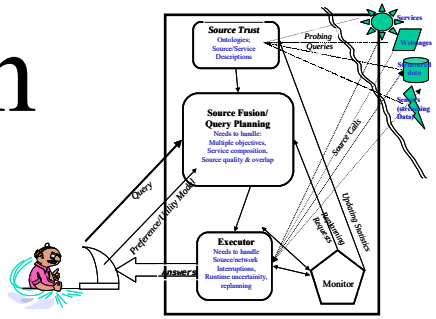
- Mostly ad-hoc programming: create a special solution for every case; pay consultants a lot of money.
- Data warehousing: load all the data periodically into a warehouse.
  - 6-18 months lead time
  - Separates *operational* DBMS from *decision support* DBMS. (not only a solution to data integration).
  - Performance is good; data may not be fresh.
  - Need to clean, scrub you data.



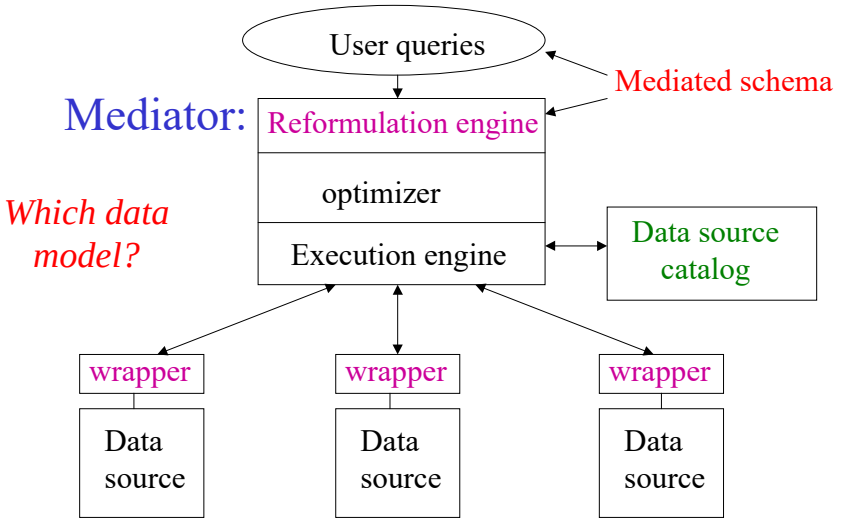

Junglee did this, for employment classifieds



# The Virtual Integration Architecture



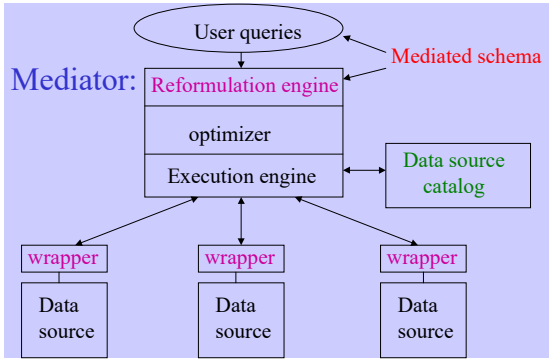

- Leave the data in the sources.
- When a query comes in:
  - Determine the relevant sources to the query
  - Break down the query into sub-queries for the sources.
  - Get the answers from the sources, and combine them appropriately.
- Data is fresh. Approach scalable
- Issues:
  - Relating Sources & Mediator
  - Reformulating the query
  - Efficient planning & execution




Garlic [IBM], Hermes[UMD];Tsimmis, InfoMaster[Stanford]; DISCO[INRIA]; Information Manifold [AT&T]; SIMS/Ariadne[USC];Emerac/Havasu[ASU]

# Desiderata for Relating Source-Mediator Schemas

- Expressive power: distinguish between sources with closely related data. Hence, be able to prune access to irrelevant sources.
- Easy addition: make it easy to add new data sources.
- Reformulation: be able to reformulate a user query into a query on the sources efficiently and effectively.
- Nonlossy: be able to handle all queries that can be answered by directly accessing the sources




## Reformulation

- Given:
  - A query Q posed over the mediated schema
  - Descriptions of the data sources
- Find:
  - A query Q' over the data source relations, such that:
    - Q' provides only *correct answers* to Q, and
    - Q' provides *all* possible answers to Q given the sources.

# Approaches for relating source & Mediator Schemas

- Global-as-view (GAV): express the mediated schema relations as a set of views over the data source relations
- Local-as-view (LAV): express the source relations as views over the mediated schema.
- Can be combined…?

"View" Refresher

```
CREATE VIEW  Seattle-view  AS

      SELECT  buyer, seller, product, store
      FROM    Person, Purchase
      WHERE   Person.city = "Seattle"    AND
              Person.name = Purchase.buyer
```

We can later use the views:

Virtual vs Materialized

```
      SELECT  name, store
      FROM    Seattle-view, Product
      WHERE   Seattle-view.product = Product.name  AND
              Product.category = "shoes"
```

Let's compare them in a movie Database integration scenario..

4/20

# Global-as-View

Mediated schema:

Movie(title, dir, year, genre),

Schedule(cinema, title, time).

Express mediator schema relations as views over source relations

[S1(title,dir,year,genre)]

[S2(title, dir,year,genre)]

[S3(title,dir), S4(title,year,genre)]

# Global-as-View

Mediated schema:

Movie(title, dir, year, genre),

Schedule(cinema, title, time).

Express mediator schema relations as views over source relations

```
Create View Movie AS
   select *  from S1      [S1(title,dir,year,genre)]
   union
   select  * from S2      [S2(title, dir,year,genre)]
   union                   [S3(title,dir), S4(title,year,genre)]
   select S3.title, S3.dir, S4.year, S4.genre
   from  S3, S4
   where S3.title=S4.title
```

Mediator schema relations are Virtual views on source relations

# Local-as-View: example 1

Mediated schema:

Movie(title, dir, year, genre),

Schedule(cinema, title, time).

Express source schema relations as views over mediator relations

Create Source S1 AS

select * from Movie

S1(title,dir,year,genre)

Create Source S3 AS

select title, dir from Movie

S3(title,dir)

Create Source S5 AS

select title, dir, year

from Movie

where year > 1960 AND genre="Comedy"

S5(title,dir,year), year >1960

Sources are "materialized views" of mediator schema

# GAV vs. LAV

Mediated schema:

Movie(title, dir, year, genre),

Schedule(cinema, title, time).

Source S4: S4(cinema, genre)

```
Create View Movie AS
  select NULL, NULL, NULL, genre
  from S4
Create View Schedule AS
  select cinema, NULL, NULL
  from S4.
```

*But what if we want to find which cinemas are playing comedies?*

Lossy mediation

```
Create Source S4
  select cinema, genre
  from Movie m, Schedule s
  where m.title=s.title
```

*Now if we want to find which cinemas are playing comedies, there is hope!*

# GAV vs. LAV

- Not modular
  - Addition of new sources changes the mediated schema
- Can be awkward to write mediated schema without loss of information
- Query reformulation easy
  - *reduces to view unfolding (polynomial)*
  - Can build hierarchies of mediated schemas
- Best when
  - Few, stable, data sources
  - well-known to the mediator (e.g. corporate integration)
    - Garlic, TSIMMIS, HERMES

- Modular--adding new sources is easy
- Very flexible--power of the entire query language available to describe sources
- Reformulation is hard
  - Involves answering queries only using views (can be intractable—see below)
- Best when
  - Many, relatively unknown data sources
  - possibility of addition/deletion of sources
    - Information Manifold, InfoMaster, Emerac, Havasu

# Reformulation in LAV: The issues

Query: Find all the years in which Zhang Yimou released movies.

Select year
 from movie M
 where M.dir=yimou

Not executable

Mediated schema:
 Movie(title, dir, year, genre),
 Schedule(cinema, title, time).

Create Source S1 AS
 select * from Movie
 S1(title,dir,year,genre)

Create Source S3 AS
 select title, dir from Movie
 S3(title,dir)

Create Source S5 AS
 select title, dir, year
 from Movie
 where year > 1960 AND genre="Comedy"
 S5(title,dir,year), year >1960

Sources are "materialized views" of Virtual schema

Q(y) :- movie(T,D,Y,G),D=yimou

Q(y) :- S1(T,D,Y,G) , D=yimou (1)

Q(y) :- S1(T,D,Y,G) , D=yimou
Q(y) :- S5(T,D,Y) , D=yimou (2)

Which is the better plan?
 What are we looking for?
 --equivalence?
 --containment?
 --Maximal Containment
 --Smallest plan?

# Maximal Containment

- Query plan should be sound and complete
  - Sound implies that Query should be contained in the Plan (I.e., tuples satisfying the query are subset of the tuples satisfying the plan
  - Completeness?
  - Traditional DBs aim for equivalence
    - Query contains the plan; Plan contains the query
      - Impossible
    - We want *all* query tuples that can be extracted given the sources we have
      - Maximal containment (no other query plan, which "contains" this plan is available)

P contains Q if P $\models$ Q
(exponential even for "conjunctive" (SPJ) query plans)

# The world of Containment

- Consider $Q_1(.)$ :- $B_1(.)$ $Q_2(.)$ :- $B_2(.)$
- $Q_1 \subseteq Q_2$ ("contained in") if the answer to $Q_1$ is a subset of $Q_2$
  - Basically holds if $B_1(x) \models B_2(x)$
- Given a query Q, and a query plan $Q_1$,
  - $Q_1$ is a sound query plan if $Q_1$ is contained in Q
  - $Q_1$ is a complete query plan if Q is contained in $Q_1$
  - $Q_1$ is a maximally contained query plan if there is no $Q_2$ which is a sound query plan for $Q_1$, such that $Q_1$ is contained in $Q_2$

# Computing Containment Checks

- Consider $Q_1(.)$ :- $B_1(.)$ $Q_2(.)$ :- $B_2(.)$
- $Q_1 \subseteq Q_2$ ("contained in") if the answer to $Q_1$ is a subset of $Q_2$
  - Basically holds if $B_1(x) \models B_2(x)$
  - (but entailment is undecidable in general… boo hoo)
- Containment can be checked through containment mappings, if the queries are "conjunctive" (select/project/join queries, without constraints) [ONLY EXPONENTIAL!!!--aren't you relieved?]
- m is a containment mapping from Vars($Q_2$) to Vars($Q_1$) if
  - ☆ m maps every subgoal in the body of $Q_2$ to a subgoal in the body of $Q_1$
  - 🕐 m maps the head of $Q_2$ to the head of $Q_1$

  Eg: Q1(x,y) :- R(x), S(y), T(x,y)  Q2(u,v) :- R(u), S(v)

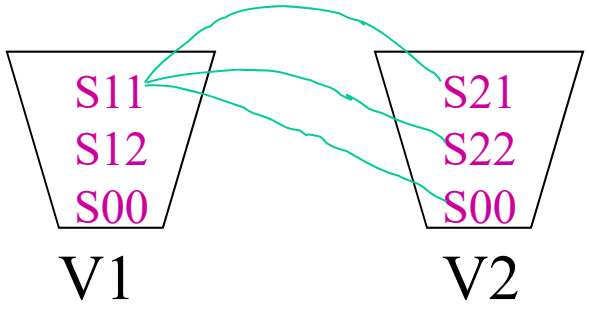
  Containment mapping: [u/x ; v/y]

# Reformulation Algorithms

Q(.) :- V1() & V2()

S11() :- V1()  S12 :- V1()
S21() :- V2()  S22 :- V2()
S00() :- V1(), V2()

## Bucket Algorithm




- Bucket algorithm
  - Cartesian product of buckets
  - Followed by "containment" check

[Levy]

$P_1$ contains $P_2$ if $P_2 \models P_1$

## Inverse Rules

Q(.) :- V1() & V2()

V1() :- S11()
V1() :- S12()
V1() :- S00()
V2() :- S21()
V2() :- S22()
V2() :- S00()

- Inverse Rules
  - plan fragments for mediator relations

# 4/22

Demo Schedule: Friday 4/30 or Monday 5/3?

Final Exam: Take-home or In-class?

Interactive Review on 5/4

# Take-home vs. In-class
## (The "Tastes Great/Less Filling" debate of CSE 494)

- More time consuming
  - To set, to take and *to grade*
    - Caveat: people tend to over-estimate the time taken to do the take-home since the don't factor out the "preparation" time that is interleaved with the exam time
- ..but may be more realistic
- Probably will be given on Th 6th and will be due by 11th evening. (+/-)

- Less time-consuming
  - To take (sort of like removing a bandage..)
  - and definitely to grade… ☺
- Scheduled to be on Tuesday, May 11th 2:40—4:30

# Interactive Review on 5/4

- A large (probably most?) of the class on 5/4 will be devoted to interactive semester review
- Everyone should come prepared with at least 3 topics/ideas that they got out of the class
  - You will be called in random order and you should have enough things to not repeat what others before you said.
- What you say will then be immortalized on the class homepage
  - See the *old* acquired wisdom page on the class page.

# Bucket Algorithm: Populating buckets

☆For each subgoal in the query, place relevant views in the subgoal's bucket

Inputs:

$Q(x)$:- $r_1(x,y)$ & $r_2(y,x)$

$V_1(a)$:-$r_1(a,b)$

$V_2(d)$:-$r_2(c,d)$

$V_3(f)$:- $r_1(f,g)$ & $r_2(g,f)$

Buckets:

| $r_1(x,y)$ |
| --- |
| $V_1(x), V_3(x)$ |

| $r_2(y,x)$ |
| --- |
| $V_2(x), V_3(x)$ |

# Combining Buckets

$R_1(x,b)$ & $R_2(c,x)$

- For every combination in the Cartesian products from the buckets, check containment in the query

$Q(x):- r_1(x,y)$ & $r_2(y,x)$

Bucket Algorithm will check all possible combinations

Candidate rewritings:

$Q'_1(x) :- V_1(x)$ & $V_2(x)$ X

$Q'_2(x) :- V_1(x)$ & $V_3(x)$ X

$Q'_3(x) :- V_3(x)$ & $V_2(x)$ X

$Q'_4(x) :- V_3(x)$ & $V_3(x)$ *

$r_1(x,y)$ $r_2(y,x)$

Buckets:

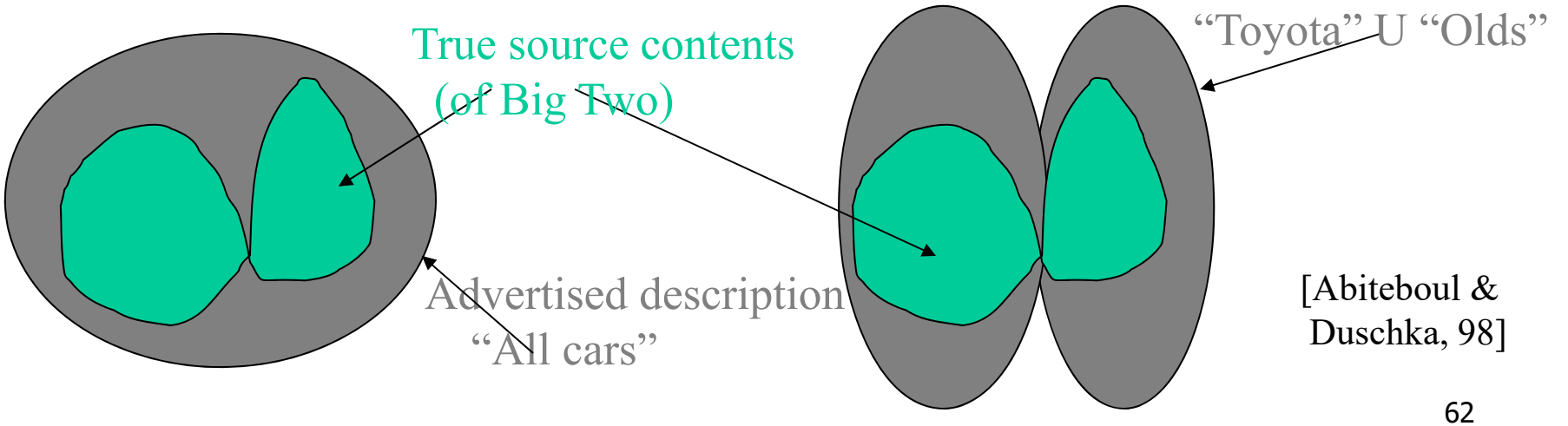
| $r_1(x,y)$ | $r_2(y,x)$ |
|---|---|
| $V_1(x), V_3(x)$ | $V_2(x), V_3(x)$ |

# Complexity of finding maximally contained plans in LAV

Big Two

- Complexity does change if the sources are not "conjunctive queries"
  - Sources as unions of conjunctive queries (NP-hard)
    - Disjunctive descriptions
  - Sources as recursive queries (Undecidable)
    - Comparison predicates
- Complexity is less dependent on the query
  - Recursion okay; but inequality constraints lead to NP-hardness
- Complexity also changes based on Open vs. Closed world assumption

You can "reduce" the complexity by taking a conjunctive query that is an upperbound.
→This just pushes the complexity to minimization phase

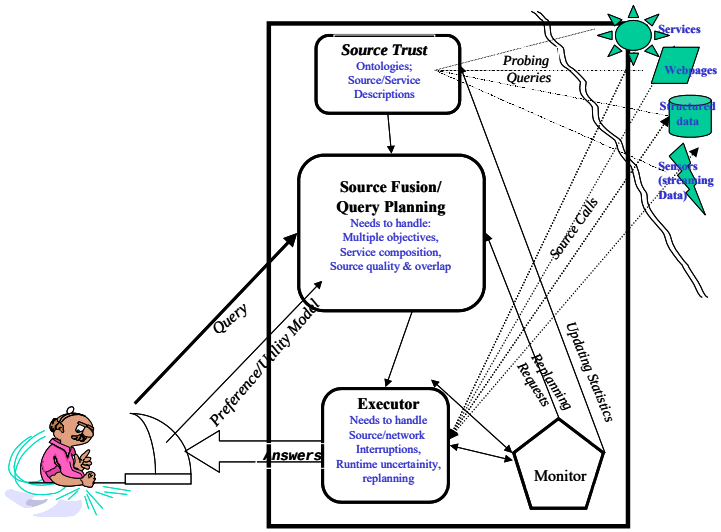


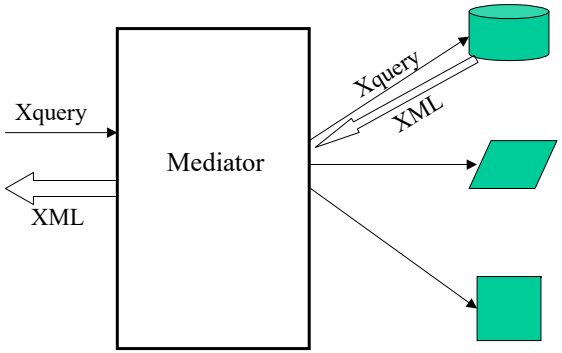

[Abiteboul & Duschka, 98]

# Is XML standardization a magical solution for Integration?

If all WEB sources standardize into XML format

- Source access (wrapper generation issues) become easier to manage
- BUT all other problems remain
  - Still need to relate source (XML)schemas to mediator (XML)schema
  - Still need to reason about source overlap, source access limitations etc.
  - Still need to manage execution in the presence of source/network uncertainities



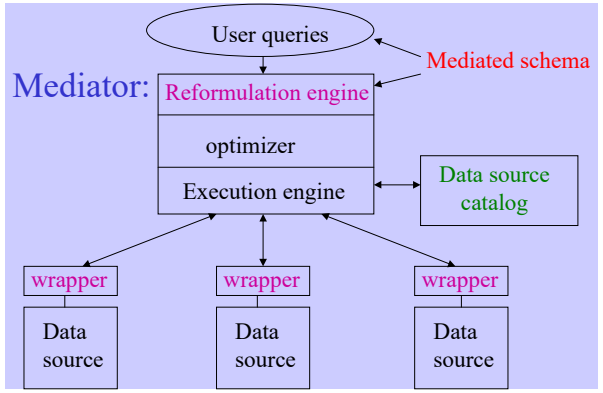
# Query Optimization Challenges

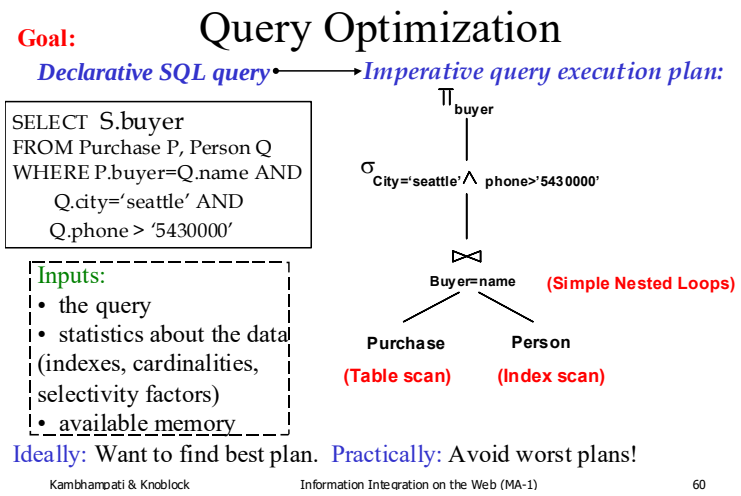
-- Deciding what to optimize

--Getting the statistics on sources

--Doing the optimization

We will first talk about reformulation level challenges



# Source Limitations

- Sources are not really fully-relational databases
  - Legacy systems
  - Limited access patters
    - (Can's ask a white-pages source for the list of all numbers)
  - Limited local processing power
    - Typically only selections (on certain attributes) are supported

- Access limitations modeled in terms of allowed ("feasible") binding patterns with which the source can be accessed
  - E.g. S(X,Y,Z) with feasible patterns *f,f,b* or *b,b,f*

# Access Restrictions & Recursive Reformulations

```
Create Source S1 as
   select *
   from Cites
   given paper1
Create Source S2 as
    select paper
    from ASU-Papers
Create Source S3 as
     select paper
     from  AwardPapers
     given paper
Query: select * from AwardPapers
```

$S1^{bf}(p1,p2)$ :- cites(p1,p2)
S2(p) :- Asp(p)
$S3^{b}(p)$ :- Awp(p)

Q(p) :- Awp(p)
Awp(p) :- Dom(p), $S3^{b}(p)$
Asp(p) :- S2(p)
Cites(p1,p2) :-Dom(p), $S1^{bf}(p)$

Dom(p) :- S2(p)
Dom(p) :- Dom(p1), S1(p1,p)

Recursive plan!!

[Kwok&Weld, 96; Duschka &Levy, 97]

# Managing Source Overlap

- Often, sources on the Internet have overlapping contents
  - The overlap is *not* centrally managed (unlike DDBMS—data replication etc.)
- Reasoning about overlap is important for plan optimality
  - We cannot possibly call all potentially relevant sources!
- Qns: How do we characterize, get and exploit source overlap?
  - Qualitative approaches (LCW statements)
  - Quantitative approaches (Coverage/Overlap statistics)

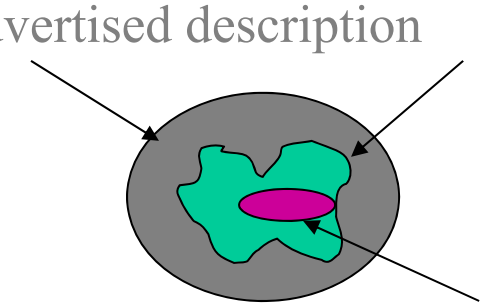
# Local Completeness Information

- If sources are incomplete, we need to look at each one of them.
- Often, sources are *locally complete*.
- Movie(title, director, year) complete for years after 1960, or for American directors.
- Question: given a set of local completeness statements, is a query Q' a complete answer to Q?

Problems:

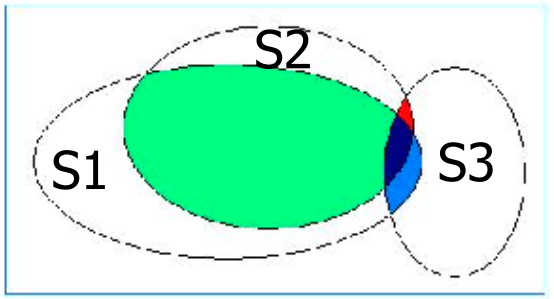
1. Sources may not be interested in giving these!
   → Need to learn
   → hard to learn!

2. Even if sources are willing to give, there may not be any "big enough" LCWs
   Saying "I definitely have the car with vehicle ID XXX is useless



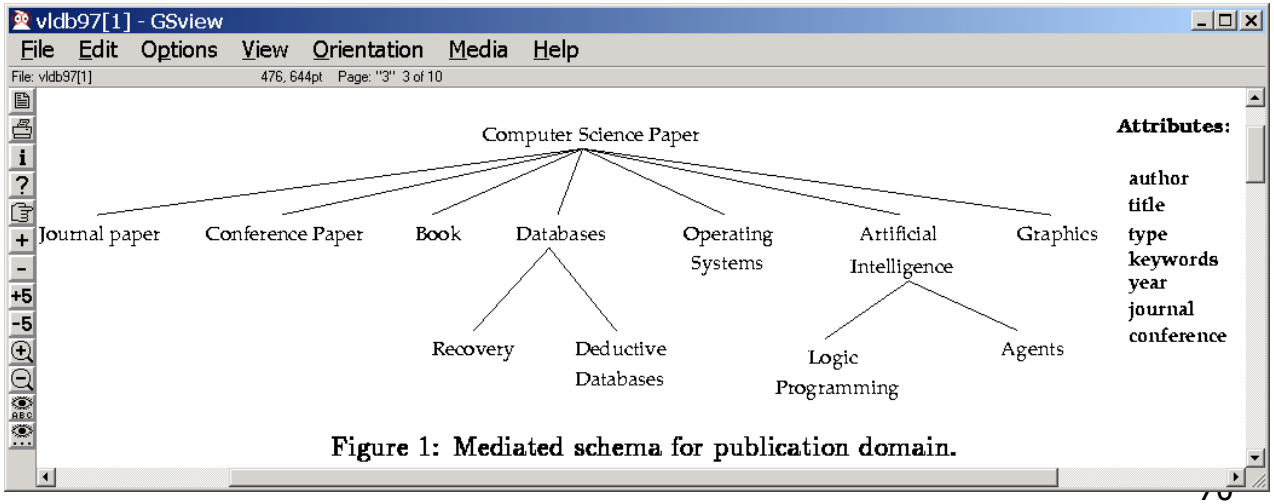
# Quantitative ways of modeling inter-source overlap

- Coverage & Overlap statistics [Koller et. al., 97]
  - $S_1$ has 80% of the movies made after 1960; while $S_2$ has 60% of the movies
  - $S_1$ has 98% of the movies stored in $S_2$
    - Computing cardinalities of unions given intersections




Extension of R

Who gives these statistics?
-Third party
-Probing



# BibFinder Case Study

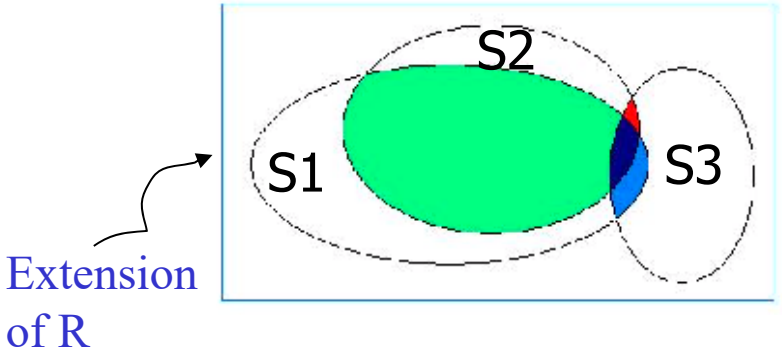
See the bibfinder slides

# What to Optimize

- Traditional DB optimizers compare candidate plans purely in terms of the time they take to produce *all* answers to a query.
- In Integration scenarios, the optimization is "*multi-objective*"
  - Total time of execution
  - Cost to first few tuples
    - Often, the users are happier with plans that give first tuples faster
  - Coverage of the plan
    - Full coverage is no longer an iron-clad requirement
      - Too many relevant sources, Uncontrolled overlap between the sources
    - Can't call them all!
  - (Robustness,
  - Access premiums…)

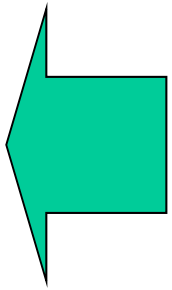
# Source Statistics Needed

- The size of the source relation and attributes;
  - The length and cardinality of the attributes;
  - the cardinality of the source relation;
- The feasible access patterns for the source;
- The network bandwidth and latency between the source and the integration system
- Coverage of the source S for a relation R denoted by P(S|R)
  - Overlap between sources $P(S_1..S_k \mid R)$



# Getting the Statistics

- Since the sources are autonomous, the mediator needs to actively gather the relevant statistics
  - Learning bandwidth and latency statistics
    - [Gruser et. al. 2000] use neural networks to learn the response time patterns of web sources
      - Can learn the variation of response times across the days of the week and across the hours of the day.
  - Learning coverages and overlaps
    - [Nie et. al. 2002] use itemset mining techniques to learn compact statistics about the spread of the mediator schema relations across the accessible sources
      - Can trade quality of the statistics for reduced space consumption

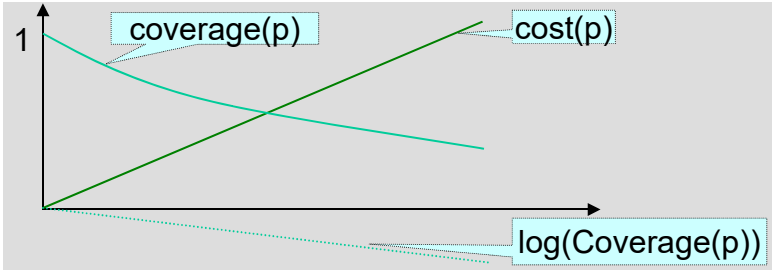


# Approaches for handling multiple objectives

- Do staged optimization
  - [Information Manifold] Optimize for coverage, and then for cost
- Do joint optimization
  - Generate all the non-dominated solutions (Pareto-Set)
  - Combine the objectives into a single metric
    - e.g. [Havasu/Multi-R]
      - » Cost increases additively
      - » Coverage decreases multiplicatively

utility(p) = w*log(coverage(p)) - (1-w)*cost(p)

- » The logarithm ensures coverage additive[Candan 01]

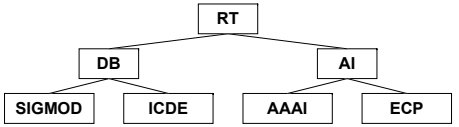


# Learning Coverage/Overlap Statistics

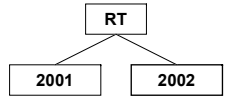
Challenge: Impractical to learn and store all the statistics for every query.

StatMiner: A threshold based hierarchical association rule mining approach

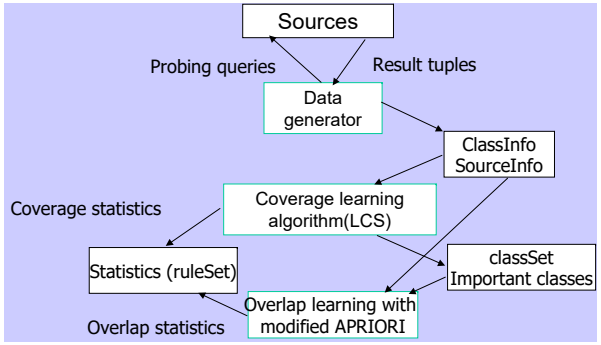
- Learns statistics with respect to "query classes" rather than specific queries
  - Defines query classes in terms of attribute-value hierarchies
  - Discovers frequent query classes and limits statistics to them
- Maps a user's query into it's closest ancestor class, and uses the statistics of the mapped class to estimate the statistics of the query.
- Handles the efficiency and accuracy tradeoffs by adjusting the thresholds.

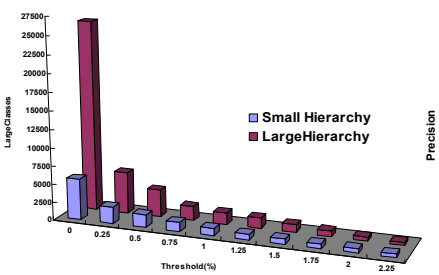


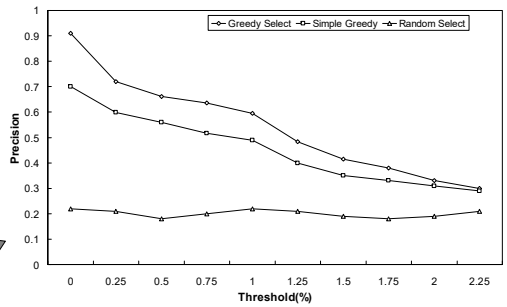

AV Hierarchy for the Conference Attribute




AV Hierarchy for the Year Attribute



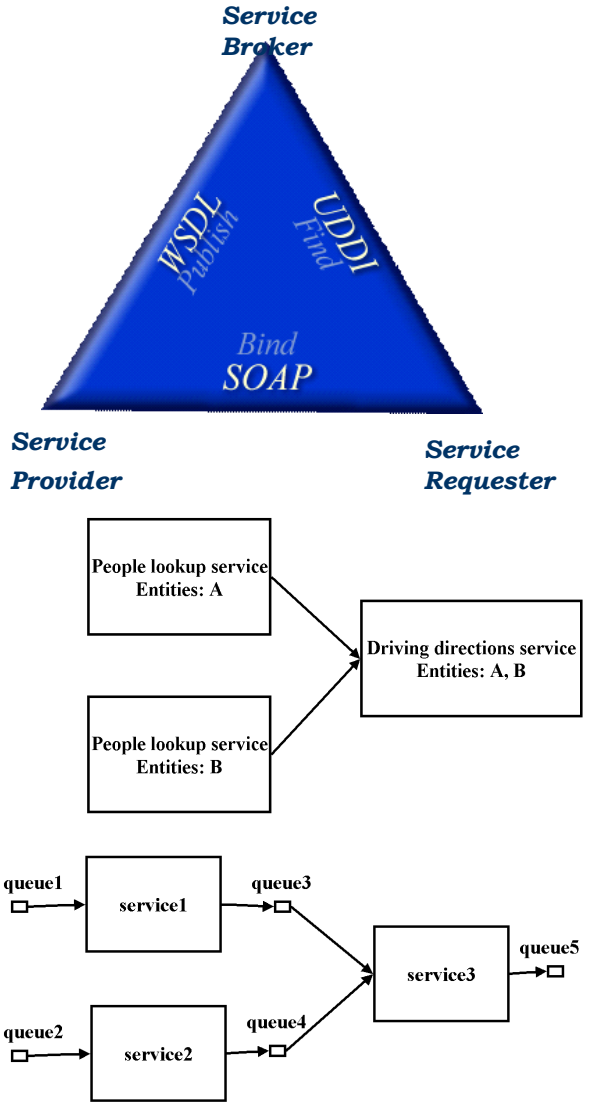

*Havasu [Nie et. al. 2002]*

# Techniques for optimizing response time for first tuples

- Staged approach: Generate plans based on other objectives and post-process them to improve their response time for first-k tuples
  - Typical idea is to replace asymmetric operators with symmetric ones
    - e.g. replace nested-loop join with symmetric hash join
      - e.g. Telegraph, Tukwila, Niagara
  - Problem: Access limitations between sources may disallow symmetric operations
  - Solution: Use joint optimization approach (e.g. Havasu) and consider the cost of first tuples as a component of plan utility
    - [Viglas & Naughton, 2002] describe approaches for characterizing the "rate" of answer delivery offered by various query plans.

# Integrating Services

- Source can be “services” rather than “data repositories”
  - Eg. Amazon as a composite service for book buying
  - Separating line is somewhat thin
- Handling services
  - Description (API;I/O spec)
    - WSDL
  - Composition
    - Planning in general
  - Execution
    - Data-flow architectures
      - See next part



# Impact of XML on Integration

**<u>If and when</u>** all sources accept Xqueries and exchange data in XML format, then

- Mediator can accept user queries in Xquery
- Access sources using Xquery
- Get data back in XML format
- Merge results and send to user in XML format

- How about now?
  - Sources can use XML adapters (middle-ware)