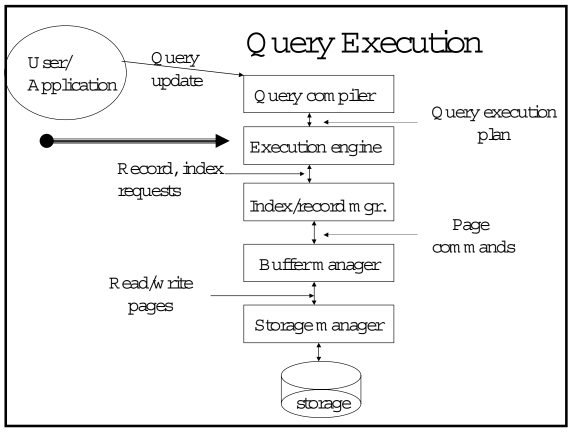
End of Query Optimization
Data Integration

May 24, 2004

Agenda

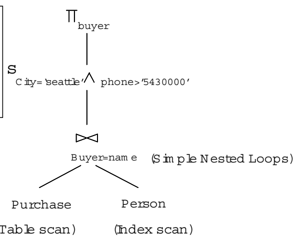
- Questions?
- Finish last bits of query optimization
- Data integration: the last frontier

Query Execution



Query Execution Plans

```
SELECT S.sname
FROM Purchase P, Person Q
WHERE P.buyer=Q.name AND
      Q.city='seattle' AND
      Q.phone > '5430000'
```



Query Plan:
- logical tree
- implementation choice at every node
- scheduling of operations.

Some operators are from relational algebra, and others (e.g., scan, group) are not.

We've Seen So Far

- Transformation rules
- The cost module:
  - Given a candidate plan: what is its expected cost and size of the result?
- Now: putting it all together.

Plans for Single-Relation Queries (Prep for Join ordering)

- Task: create a query execution plan for a single Select-project-group-by block.
- Key idea: consider each possible access path to the relevant tuples of the relation. Choose the cheapest one.
- The different operations are essentially carried out together (e.g., if an index is used for a selection, projection is done for each retrieved tuple, and the resulting tuples are pipelined into the aggregate computation).

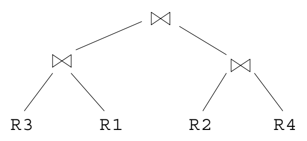
Example

```
SELECT S.sid
FROM Sailors S
WHERE S.rating=8
```

- If we have an Index on rating:
  - (1/NKeys(I)) * NTuples(R) = (1/10) * 40000 tuples retrieved.
  - Clustered index: (1/NKeys(I)) * (NPages(I)+NPages(R)) = (1/10) * (50+500) pages are retrieved (= 55).
  - Unclustered index: (1/NKeys(I)) * (NPages(I)+NTuples(R)) = (1/10) * (50+40000) pages are retrieved.
- If we have an index on sid:
  - Would have to retrieve all tuples/pages. With a clustered index, the cost is 50+500.
- Doing a file scan: we retrieve all file pages (500).

Determining Join Ordering

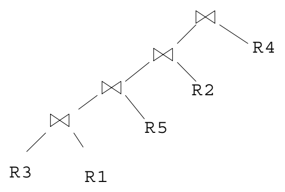
- R1 $\bowtie$ R2 $\bowtie$ …. $\bowtie$ Rn
- Join tree:



- A join tree represents a plan. An optimizer needs to inspect many (all?) join trees

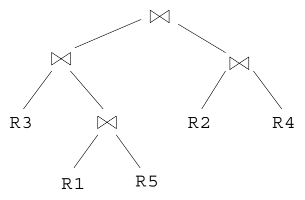
Types of Join Trees

- Left deep:



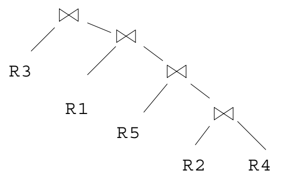
Types of Join Trees

- Bushy:



Types of Join Trees

- Right deep:



Problem

- Given: a query R1 $\bowtie$ R2 $\bowtie$ … $\bowtie$ Rn
- Assume we have a function cost() that gives us the cost of every join tree
- Find the best join tree for the query

Join Ordering by Dynamic Programming

- Idea: for each subset of {R1, …, Rn}, compute the best plan for that subset
- In increasing order of set cardinality:
  - Step 1: for {R1}, {R2}, …, {Rn}
  - Step 2: for {R1,R2}, {R1,R3}, …, {Rn-1, Rn}
  - …
  - Step n: for {R1, …, Rn}
- A subset of {R1, …, Rn} is also called a subquery

Dynamic Programming: step 1

- Step 1: For each {Ri} do:
  - Size({Ri}) = B(Ri)
  - Plan({Ri}) = Ri
  - Cost({Ri}) = (cost of scanning Ri)

Dynamic Programming: step i:

- Step i: For each Q in {R1, …, Rn} of cardinality i do:
  - Compute Size(Q)
  - For every pair of subqueries Q', Q'' s.t. Q = Q' U Q'' compute cost(Plan(Q') $\bowtie$ Plan(Q''))
  - Cost(Q) = the smallest such cost
  - Plan(Q) = the corresponding plan

A few practical considerations

- Heuristics for reducing the search space
  - Restrict to left linear trees
  - Restrict to trees "without cartesian product"
- Need more than just one plan for each subquery:
  - "interesting orders": save a single plan for every possible ordering of the result.
  - Why?

Query Optimization Summary

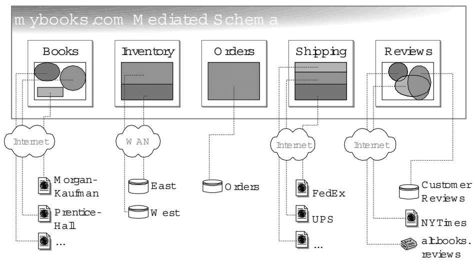
- Create initial (naïve) query execution plan.
- Apply transformation rules:
  - Try to un-nest blocks
  - Move predicates and grouping operators.
- Consider each block at a time:
  - Determine join order
  - Push selections, projections if possible.

Data Integration

What is Data Integration

- Providing
  - Uniform (same query interface to all sources)
  - Access to (queries; eventually updates too)
  - Multiple (we want many, but 2 is hard too)
  - Autonomous (DBA doesn't report to you)
  - Heterogeneous (data models are different)
  - Structured (or at least semi-structured)
  - Data Sources (not only databases).

The Problem: Data Integration



Uniform query capability across autonomous, heterogeneous data sources on LAN, WAN, or Internet

Motivation(s)

- Enterprise data integration; web-site construction.
- 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.
  - Environment: Puget Sound Regional Synthesis Model
  - Culture: uniform access to all cultural databases produced by countries in Europe.

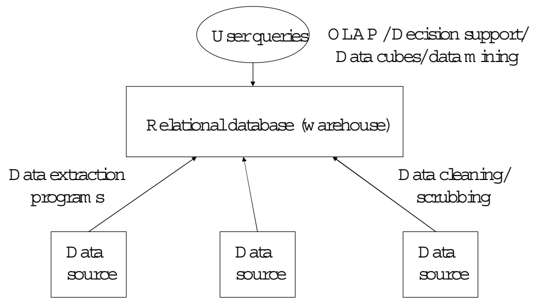
Discussion

- Why is it hard?
- How will we solve it?

Current Solutions

- 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.

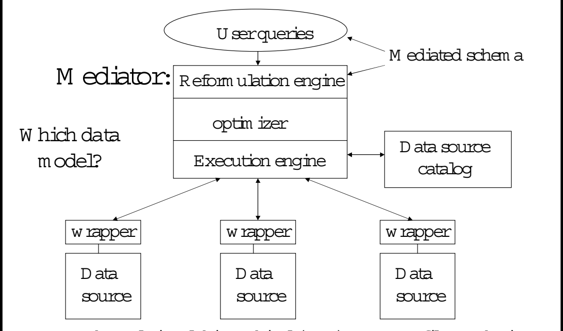
Data Warehouse Architecture



The Virtual Integration Architecture

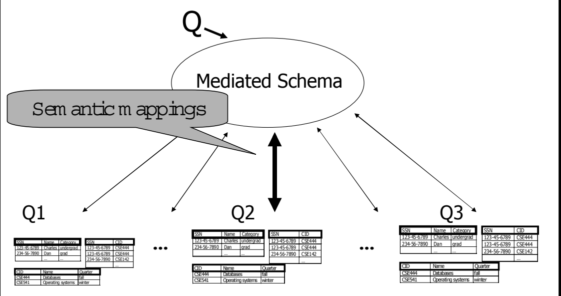
- Leave the data in the sources.
- When a query comes in:
  - Determine the relevant sources to the query
  - Break down the query into sub-queries for the sources.
  - Get the answers from the sources, and combine them appropriately.
- Data is fresh.
- Challenge: performance.

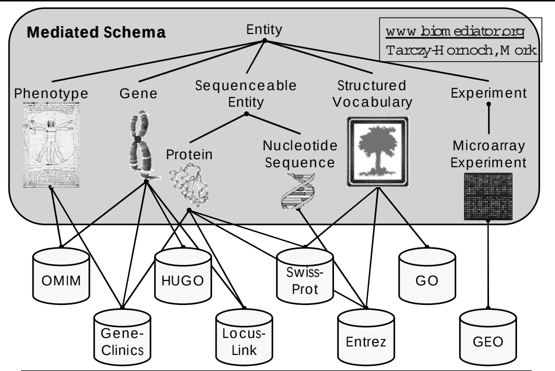
Virtual Integration Architecture



Sources can be: relational, hierarchical (IMS), structure files, web sites.

Data Integration: Higher-level Abstraction





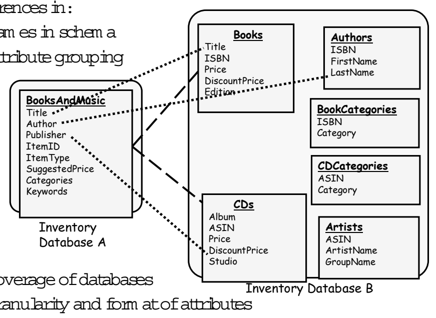
Query: For the micro-array experiment I just ran, what are the related nucleotide sequences and for what protein do they code?

Research Projects

- Garlic (IBM),
- Information Manifold (AT&T)
- Tsimmis, InfoMaster (Stanford)
- The Internet Softbot/Razor/Tukwila (UW)
- Hermes (Maryland)
- DISCO, Agora (INRIA, France)
- SIMS/Ariadne (USC/ISI)
- Many, many more!

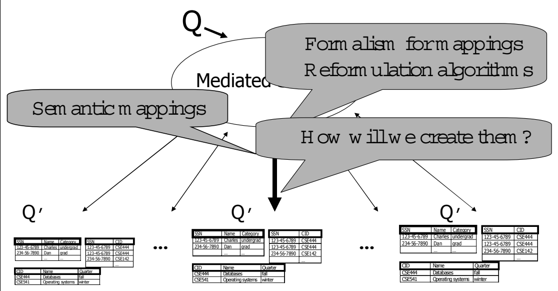
Semantic Mappings

- Differences in:
  - Names in schema
  - Attribute grouping



  - Coverage of databases
  - Granularity and format of attributes

Issues for Semantic Mappings



SSN	Name	Category
123-45-6789	Charles	undergrad
234-56-7890	Dan	grad
	...	...

SSN	CID
123-45-6789	CSE444
123-45-6789	CSE444
234-56-7890	CSE142
	...

CID	Name	Quarter
CSE444	Databases	fall
CSE541	Operating systems	winter

SSN	Name	Category
123-45-6789	Charles	undergrad
234-56-7890	Dan	grad
	...	...

SSN	CID
123-45-6789	CSE444
123-45-6789	CSE444
234-56-7890	CSE142
	...

CID	Name	Quarter
CSE444	Databases	fall
CSE541	Operating systems	winter

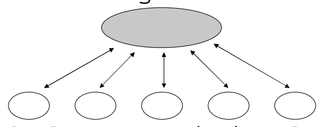
SSN	Name	Category
123-45-6789	Charles	undergrad
234-56-7890	Dan	grad
	...	...

SSN	CID
123-45-6789	CSE444
123-45-6789	CSE444
234-56-7890	CSE142
	...

CID	Name	Quarter
CSE444	Databases	fall
CSE541	Operating systems	winter

Beyond Data Integration

- Mediated schema is a bottleneck for large-scale data sharing

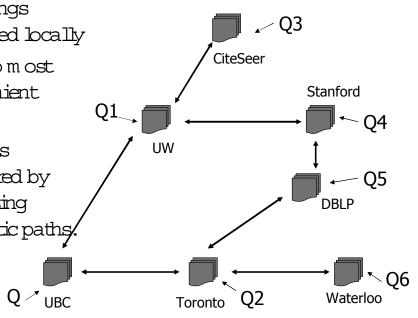


- It's hard to create, maintain, and agree upon.

Peer Data Management Systems

Piazza: [Tatarinov, H., Ives, Suciu, Mork]

- Mappings specified locally
- Map to most convenient nodes
- Queries answered by traversing semantic paths.



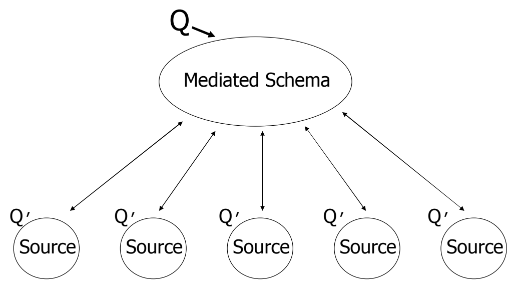
PDMS-Related Projects

- Hyperion (Toronto)
- PeerDB (Singapore)
- Local relational models (Trento)
- Edutella (Hannover, Germany)
- Semantic Gossiping (EPFL Zurich)
- Raccoon (UC Irvine)
- Orchestra (Ives, U. Penn)

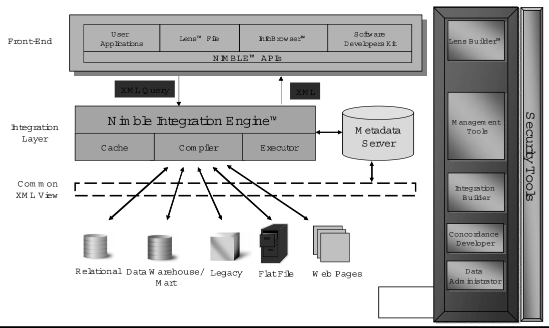
A Few Comments about Commerce

- Until 5 years ago:
  - Data integration = Data warehousing.
- Since then:
  - A wave of startups:
    - Nimble, MetaMatrix, Calixa, Composite, Enosys
  - Big guys made announcements (IBM, BEA).
  - [Delay] Big guys released products.
- Success: analysts have new buzzword – EII
  - New addition to acronym soup (with EAI).
- Lessons:
  - Performance was fine. Need management tools.

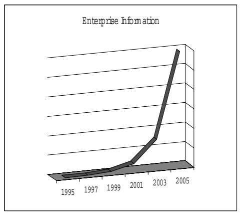
Data Integration: Before



Data Integration: After



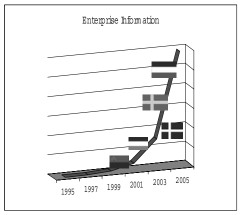
Sound Business Models



- Explosion of intranet and extranet information
- 80% of corporate information is unmanaged
- By 2004 30X more enterprise data than 1999
- The average company:
  - maintains 49 distinct enterprise applications
  - spends 35% of total IT budget on integration-related efforts

Source: Gartner, 1999

Sound Business Models



- Explosion of intranet and extranet information
- 80% of corporate information is unmanaged
- By 2004 30X more enterprise data than 1999
- The average company:
  - maintains 49 distinct enterprise applications
  - spends 35% of total IT budget on integration-related efforts

Source: Gartner, 1999

Dimensions to Consider

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

Outline

- Wrappers
- Semantic integration and source descriptions:
  - Modeling source completeness
  - Modeling source capabilities
- Query optimization
- Query execution
- Peer-data management systems
- Creating schema mappings

Wrapper Programs

- Task: to communicate with the data sources and do format translations.
- They are built w.r.t. a specific source.
- They can sit either at the source or at the mediator.
- Often hard to build (very little science).
- Can be "intelligent": perform source-specific optimizations.

Example

Transform:

```
<b> Introduction to DB </b>
<i> Phil Bernstein </i>
<i> Eric Newcomer </i>
Addison Wesley, 1999
```

into:

```
<book>
<title> Introduction to DB </title>
<author> Phil Bernstein </author>
<author> Eric Newcomer </author>
<publisher> Addison Wesley </publisher>
<year> 1999 </year>
</book>
```

Data Source Catalog

- 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.

Content Descriptions

- 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.
- Data integration system uses the descriptions to translate user queries into queries on the sources.

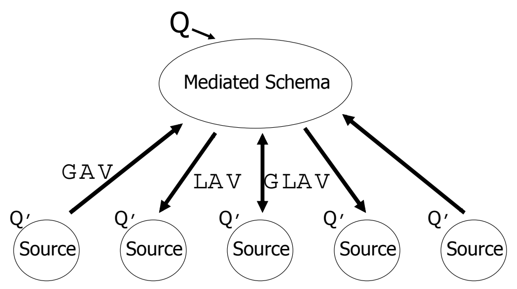
Desiderata from Source Descriptions

- 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.

Reformulation Problem

- 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 from to Q given the sources.

Languages for Schema Mapping



Global-as-View

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

Global-as-View: Example 2

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

Create View Movie AS [S1(title,dir,year)]
  select title, dir, year, NULL
  from S1
  union                  [S2(title, dir,genre)]
  select title, dir, NULL, genre
  from S2
```

Global-as-View: Example 3

```
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?

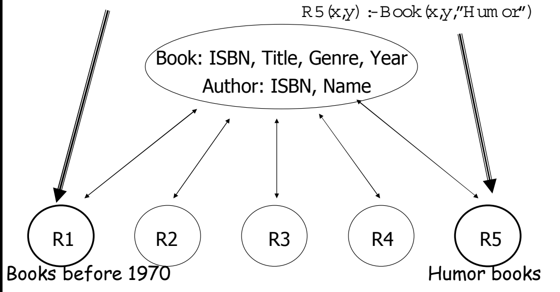
Global-as-View Summary

- Query reformulation boils down to view unfolding.
- Very easy conceptually.
- Can build hierarchies of mediated schemas.
- You sometimes loose information. Not always natural.
- Adding sources is hard. Need to consider all other sources that are available.

Local-as-View (LAV)

R1(x,y,n) :- Book(x, y, z, t), Author(x, n), t < 1970

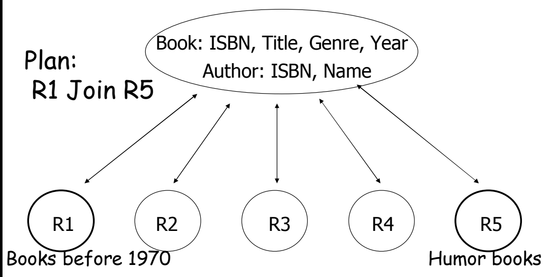
R5(x,y) :- Book(x,y,"Humor")



Query Reformulation

Query: Find authors of humor books

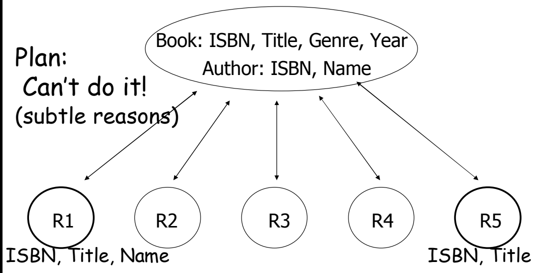
Plan:
R1 Join R5



Query Reformulation

Find authors of humor books before 1960

Plan:
Can't do it!
(subtle reasons)



Local-as-View: example 1

```
Mediated schema:
  Movie(title, dir, year, genre),
  Schedule(cinema, title, time).
Create Source S1 AS
  select * from Movie
Create Source S3 AS    [S3(title, dir)]
  select title, dir from Movie
Create Source S5 AS
  select title, dir, year
  from Movie
  where year > 1960 AND genre="Comedy"
```

Local-as-View: Example 2

```
Mediated schema:
  Movie(title, dir, year, genre),
  Schedule(cinema, title, time).
Source S4: S4(cinema, genre)
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!
```

Local-as-View Summary

- Very flexible. You have the power of the entire query language to define the contents of the source.
- Hence, can easily distinguish between contents of closely related sources.
- Adding sources is easy: they're independent of each other.
- Query reformulation: answering queries using views!

The General Problem

- Given a set of views V1,…,Vn, and a query Q, can we answer Q using only the answers to V1,…,Vn?
- Many, many papers on this problem.
- The best performing algorithm: The MiniCon Algorithm, (Pottinger & Levy, 2000).
- Great survey on the topic: (Halevy, 2001).

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?

Example

- Movie(title, director, year) (complete after 1960).
- Show (title, theater, city, hour)
- Query: find movies (and directors) playing in Seattle:

  Select m.title, m.director

  From Movie m, Show s

  Where m.title=s.title AND city="Seattle"
- Complete or not?

Example #2

- Movie(title, director, year), Oscar(title, year)
- Query: find directors whose movies won Oscars after 1965:

  select m.director

  from Movie m, Oscar o

  where m.title=o.title AND m.year=o.year AND o.year > 1965.
- Complete or not?

Query Optimization

- Very related to query reformulation!
- Goal of the optimizer: find a physical plan with minimal cost.
- Key components in optimization:
  - Search space of plans
  - Search strategy
  - Cost model

Optimization in Distributed DBMS

- A distributed database (2-minute tutorial):
  - Data is distributed over multiple nodes, but is uniform.
  - Query execution can be distributed to sites.
  - Communication costs are significant.
- Consequences for optimization:
  - Optimizer needs to decide locality
  - Need to exploit independent parallelism.
  - Need operators that reduce communication costs (semi-joins).

DDBMS vs. Data Integration

- In a DDBMS, data is distributed over a set of uniform sites with precise rules.
- In a data integration context:
  - Data sources may provide only limited access patterns to the data.
  - Data sources may have additional query capabilities.
  - Cost of answering queries at sources unknown.
  - Statistics about data unknown.
  - Transfer rates unpredictable.

Modeling Source Capabilities

- Negative capabilities:
  - A web site may require certain inputs (in an HTML form).
  - Need to consider only valid query execution plans.
- Positive capabilities:
  - A source may be an ODBC compliant system.
  - Need to decide placement of operations according to capabilities.
- Problem: how to describe and exploit source capabilities.

Example #1: Access Patterns

Mediated schema relation: Cites(paper1, paper2)

```
Create Source S1 as
  select *
  from Cites
  given paper1
Create Source S2 as
   select paper1
   from Cites
```

Query: select paper1 from Cites where paper2="Hal00"

Example #1: Continued

```
Create Source S1 as
  select *
  from Cites
  given paper1
Create Source S2 as
   select paper1
   from Cites
Select p1
From S1, S2
Where S2.paper1=S1.paper1 AND S1.paper2="Hal00"
```

Example #2: Access Patterns

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

Example #2: Solutions

- Can't go directly to S3 because it requires a binding.
- Can go to S1, get UW papers, and check if they're in S3.
- Can go to S1, get UW papers, feed them into S2, and feed the results into S3.
- Can go to S1, feed results into S2, feed results into S2 again, and then feed results into S3.
- Strictly speaking, we can't a priori decide when to stop.
- Need recursive query processing.

Handling Positive Capabilities

- Characterizing positive capabilities:
  - Schema independent (e.g., can always perform joins, selections).
  - Schema dependent: can join R and S, but not T.
  - Given a query, tells you whether it can be handled.
- Key issue: how do you search for plans?
- Garlic approach (IBM): Given a query, STAR rules determine which subqueries are executable by the sources. Then proceed bottom-up as in System-R.

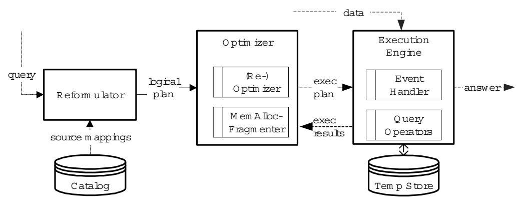
Matching Objects Across Sources

- How do I know that A. Halevy in source 1 is the same as Alon Halevy in source 2?
- If there are uniform keys across sources, no problem.
- If not:
  - Domain specific solutions (e.g., maybe look at the address, ssn).
  - Use Information retrieval techniques (Cohen, 98). Judge similarity as you would between documents.
  - Use concordance tables. These are time-consuming to build, but you can then sell them for lots of money.

Optimization and Execution

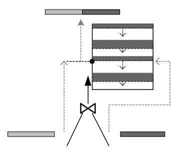
- Problem:
  - Few and unreliable statistics about the data.
  - Unexpected (possibly bursty) network transfer rates.
  - Generally, unpredictable environment.
- General solution: (research area)
  - Adaptive query processing.
  - Interleave optimization and execution. As you get to know more about your data, you can improve your plan.

Tukwila Data Integration System

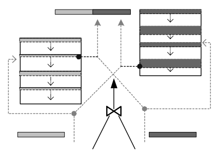


Novel components:
- Event handler
- Optimization-execution loop

Double Pipelined Join (Tukwila)



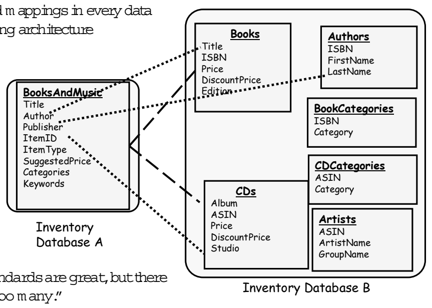
Hash Join
- 8 Partially pipelined: no output until inner read
- 8 Asymmetric (inner vs. outer) – optimization requires source behavior knowledge



Double Pipelined Hash Join
- 4 Outputs data immediately
- 4 Symmetric – requires less source knowledge to optimize

Semantic Mappings

- Need mappings in every data sharing architecture



- "Standards are great, but there are too many."

Why is it so Hard?

- Schemas never fully capture their intended meaning:
  - We need to leverage any additional information we may have.
- A human will always be in the loop.
  - Goal is to improve designer's productivity.
  - Solution must be extensible.
- Two cases for schema matching:
  - Find a map to a common mediated schema.
  - Find a direct mapping between two schemas.

Typical Matching Heuristics

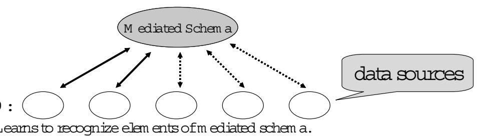
- We build a model for every element from multiple sources of evidences in the schemas
  - Schema element names
    - BooksAndCDs/Categories ~ BookCategories/Category
  - Descriptions and documentation
    - ItemID: unique identifier for a book or a CD
    - ISBN: unique identifier for any book
  - Data types, data instances
    - DateTime ,, Integer,
    - addresses have similar formats
  - Schema structure
    - All books have similar attributes

In isolation, techniques are incomplete or brittle:
Need principled combination.

Models consider *only* the two schemas.

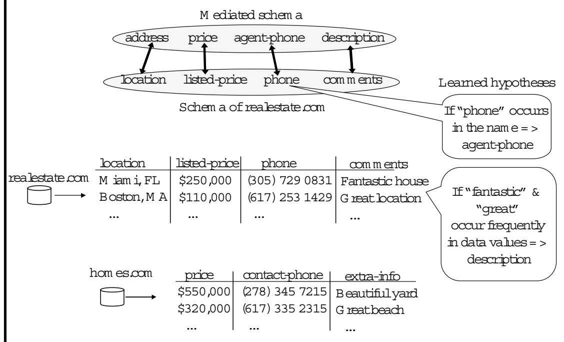
Using Past Experience

- Matching tasks are often repetitive
- Humans improve over time at matching.
  - A matching system should improve too!



- LSD:
  - Learns to recognize elements of mediated schema.
  - [Doan, Domingos, H., SIGMOD-01, MLJ-03]
    - Doan: 2003 ACM Distinguished Dissertation Award.

Example: Matching Real-Estate Sources



realestate.com

location	listed-price	phone	comments
Miami, FL	$250,000	(305) 729 0831	Fantastic house
Boston, MA	$110,000	(617) 253 1429	Great location
...	...	...	...

homes.com

price	contact-phone	extra-info
$550,000	(278) 345 7215	Beautiful yard
$320,000	(617) 335 2315	Great beach
...	...	...

Learning Source Descriptions

- We learn a classifier for each element of the mediated schema.
- Training examples are provided by the given mappings.
- Multi-strategy learning:
  - Base learners: name, instance, description
  - Combine using stacking.
- Accuracy of 70-90% in experiments.
- Learning about the mediated schema.

Multi-Strategy Learning

- Use a set of base learners:
  - Name learner, Naïve Bayes, Whirl, XML learner
- And a set of recognizers:
  - County name, zip code, phone numbers.
- Each base learner produces a prediction weighted by confidence score.
- Combine base learners with a meta-learner, using stacking.

The Semantic Web

- A web of structured data:
  - The 5-year old vision of Tim Berners-Lee
- How does it relate to data integration?
- How are we going to do it?
- Why should we do it? Do we need a killer app or is the semantic web a killer app?

The End