End of Query Optimization
Data Integration

May 24, 2004

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

Query Execution

User/Application

Query execution plan

Query compiler

Execution engine

Index/record mgr.

Buffer mgr.

Storage mgr.

Query Execution Plans

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

Query Plan:
• logical tree
• in plan entation choice at every node
• scheduling of operations.

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

If we have an index on rating:
- Clustering: \( \frac{1}{N_{\text{Keys}}} \times N_{\text{Tuples}} = \frac{1}{10} \times 40000 \) tuples retrieved.
- Clustering: \( \frac{1}{N_{\text{Keys}}} \times (N_{\text{Pages}} + N_{\text{Tuples}}) = \frac{1}{10} \times (50 + 500) \) pages are retrieved (= 55).
- Unclustering: \( \frac{1}{N_{\text{Keys}}} \times (N_{\text{Pages}} + N_{\text{Tuples}}) = \frac{1}{10} \times (50 + 40000) \) pages are retrieved.

If we have an index on sid:
- We 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).

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

Determining Join Ordering

Types of Join Trees

- Left deep:

![Left deep tree]

- Bushy:

![Bushy tree]

- Right deep:

![Right deep tree]

Problem

- Given a query \( R_1 \Join R_2 \Join \ldots \Join R_n \)
- Assume we have a function \( \text{cost()} \) 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 \( \{R_1, \ldots, R_n\} \), compute the best plan for that subset.
- In increasing order of set cardinality:
  - Step 1: for \( \{R_1\}, \{R_2\}, \ldots, \{R_n\} \)
  - Step 2: for \( \{R_1, R_2\}, \{R_1, R_3\}, \ldots, \{R_{n-1}, R_n\} \)
  - ...
  - Step n: for \( \{R_1, \ldots, R_n\} \)
- A subset of \( \{R_1, \ldots, R_n\} \) is also called a subquery.

Dynamic Programming: step 1

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

Dynamic Programming: step i:

- Step i: For each \( Q \) in \( \{R_1, \ldots, R_n\} \) of cardinality \( i \) do:
  - Compute \( \text{Size}(Q) \)
  - For every pair of subqueries \( Q', Q'' \) s.t. \( Q = Q' \cup Q'' \)
    - Compute \( \text{cost}(\text{Plan}(Q'), \text{Plan}(Q'')) \)
  - \( \text{Cost}(Q) = \text{the smallest such cost} \)
  - \( \text{Plan}(Q) = \text{the corresponding plan} \)

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 order”: 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 any, 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).
  - Perform once is good; data may not be fresh.
  - Need to clean, scrub you data.

Data Warehouse Architecture

- OLAP/Decision support/Data cubes/Data mining
- Data extraction programs
- Data cleaning/scrubbing
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

Mediator:
- Reformulation engine
- Execution engine

Sources:
- Data source
- Data source
- Data source

Challenges:
- Which data model?
- Data source catalog
- Execution engine

Mediated Schema

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

Data Integration: Higher-level Abstraction

Semantically mappings

Q1

Q2

Q3

Research Projects

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

Semantic Mappings

Books & Music

Title

Author

Publisher

Item ID

ItemType

Suggested Price

Categories

Keywords

Books

Title

ISBN

Price

Discount Price

Edition

CDs

Album

ASIN

Price

Discount Price

Studio

Inventory

Database A

Database B

Differences in:
- Names in schema
- Attribute grouping
- Coverage of databases
- Granularity and format of attributes
Issues for Semantic Mappings

Mediated Schema

Form all form mappings Reformulation algorithms

How will we create them?

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, M. cak]
・ Mappings specified locally
・ Mapping cost:
・ Convenient nodes
・ Queries answered by traversing semantic paths

Q1 Q2 Q3 Q4 Q5 Q6

UBC Waterloo
CiteSeer
Toronto
Q1 Q2 Q3
Q4 Q5 Q6

PDM S-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:
・ IBM, Oracle, CA, Informatica, Business Objects
・ Envelope of new systems
・ Big hype
・ Big release of new products.
・ Success as analysts have new buzzword - EII
・ New set of products from top 10 vendors.
・ Lessons:
・ Performance is as fine. Need management tools.

Data Integration: Before

Mediated Schema

Source Source Source Source Source Source
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 in 1999
- The average company:
  - Maintains 49 distinct enterprise applications
  - Spends 35% of total IT budget on integration-related efforts

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
- Wrapper Programs
  - Task: to communicate with the data sources and do form at translations.
  - They are built wrt 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.

- Wrapper Programs
  - Task: to communicate with the data sources and do form at translations.
  - They are built wrt 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:
- Introduction to DB
- Phil Bernstein
- Eric Neveu
- Addison Wesley, 1999

Into:
- <book>
  - <title>Introduction to DB</title>
  - <author>Phil Bernstein</author>
  - <author>Eric Neveu</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 Q given the sources.

Languages for Schema Mapping
- GAV
- LAV
- GLAV
- Mediated Schema
- Q
- Q'
- Source
- Source
- Source
- Source
- Source
- Source

8
Global-as-View

Mediated schema:
Movie(title, dir, year, genre),
Schedule(cinema, title, time).
Create View Movie AS
select * from S1
union
select * from S2
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
select title, dir, year, NULL
from S1
union
select title, dir, NULL, genre
from S2
But what if we want to find which cinemas are playing comedies?

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.

Global-as-View Summary

- Query reformulation boils down to view unfolding.
- Very easy conceptually.
- Can build hierarchies of mediated schema.
- Adding sources is hard. Need to consider all other sources that are available.

Local-as-View (LAV)

Book: ISBN, Title, Genre, Year
Author: ISBN, Name

Books before 1970
Humor books

Query Reformulation

Query: Find authors of humor books
Plan:
R1 Join R5

Books before 1970
Humor books
**Query Reformulation**

Find authors of humor books before 1960

**Plan:**

Can’t do it! (subtle reasons)

R1 ISBN, Title, Name
R2 ISBN, Title
R3 ISBN, Title
R4 ISBN, Title
R5 ISBN, Title

---

**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)

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 $V_1, \ldots, V_n$, and a query $Q$, can we answer $Q$ using only the answers to $V_1, \ldots, V_n$?
- Many, many papers on this problem.

---

**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 accessing 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

Create Source S1 as
select * from Cites given paper1

Create Source S2 as
select paperID from UW-Papers

Query: select * from AwardPapers

Example #1: Continued

Create Source S1 as
select * from Cites given paper1

Create Source S2 as
select paperID from UW-Papers

Query: select * from AwardPapers

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, tell 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 A. Halevy in source 2?
- If there are uniform keys across sources, no problem.
- If not:
  - Domain specific solutions (e.g., use the look at the address, etc.).
  - 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.

### Double Pipelined Join (Tukwila)

**Hash Join**
- Partially pipelined: no output until known match
- A symmetric inner vs. outer — optimization requires source behavior knowledge

**Double Pipelined Hash Join**
- Outputs data immediately
- Symmetric — requires less source knowledge to optimize

### Tukwila Data Integration System

- N novel components:
  - Event handler
  - Optimization-execution loop

### 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 evidence in the schema as:
  - Schema element names
  - Descriptions and dictionary entries
  - Data types, data instances
  - Addresses have similar formats
  - Schem a structure
    - A schema having 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, ML J-03]

Doan: 2003 ACM Distinguished Dissertation Award.

Example: Matching Real-Estate Sources

<table>
<thead>
<tr>
<th>location</th>
<th>listed-price</th>
<th>phone</th>
<th>comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miami, FL</td>
<td>$250,000</td>
<td>(305) 729 0831</td>
<td>Fantastic house</td>
</tr>
<tr>
<td>Boston, MA</td>
<td>$110,000</td>
<td>(617) 253 1429</td>
<td>Great location</td>
</tr>
</tbody>
</table>

LSD: Mediated Schema

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.
- A accuracy of 70-90% in experiments.
- Learning about the mediated schema.

Multi-Strategy Learning

- Use a set of base learners:
  - Name learner, naive Bayes, Whirl, XML learner
- 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