Data Integration Services

1 Introduction

** subject to change...

1.1 Definition of Data Integration

1.2 Motivation for data integration systems

- Users can focus on specifying *what* they want, not on *how* to obtain what they want. Instead of finding relevant sources, interacting with every source and combining data from different sources, a user can ask queries in a unified way.
  Particular examples:
  - Desire for reports that describe all parts of a merged organization (bank mergers, car dealerships, etc.).
- Facilitates new functionality (OLAP, Data mining)

1.3 Issues

- Heterogeneity of data sources
- Availability of data sources
- Dynamicity of individual data sources
- Autonomy of data sources.
- Correctness of the integrated view of the data
- Query performance

2 Data Integration Architectures

2.1 Attributes to categorize data integration architectures

- Autonomy
- Heterogeneity
- Distribution

2.2 Three approaches to data integration systems classification based on how data access is controlled

- Virtual approach
  Data is accessed on-demand, based on a user query (lazy approach).
- Warehouse approach
  Information is first fetched into a local repository to be queried later.

- Hybrid approach
  Data is selectively materialized.

2.3 Virtual approach

2.3.1 Problems common to all virtual view systems

2.3.2 Federated database systems

- Definition
- How queries are handled
- Additional problems raised by this specific architecture
- Known implementations

2.3.3 Mediated systems

- Definition
- Components
- How queries are handled
- Additional problems raised by this specific architecture
- Known implementations

2.4 Materialized approach (a.k.a. data warehouses)

- Definition
- Components
- How the materialized data is maintained
- How queries are handled
- Additional problems raised by this specific architecture See "Materialized View Maintenance" section.
- Known implementations

2.5 Hybrid approach

- Definition
- What data to materialize?
- How data is maintained

2.6 Comparative strengths and weaknesses of each architecture

3 Query Processing in Data Integration Systems

What is query processing in general terms? What stages does it involve? Why is it an issue to be discussed in data integration systems? What difficulties/complexities data integration systems bring that we need more than traditional query processing techniques?

3.1 Data Modeling

3.1.1 Languages

Alternatives will be discussed and Datalog will be briefly introduced as the choice of query and view language for the following subsections.

3.1.2 Modeling the Mediated Schema

3.1.3 Modeling the Data Sources

Data source descriptions which specify the relationship between the relations in the mediated schema and those in the local schemas at the sources should be provided. The description of a source specifies its

- contents
- attributes
- constraints on its contents
- completeness and reliability
- query processing capabilities

Besides, using the descriptions of the sources, we should be able to detect the following:

- overlapping and contradictory data among sources
- semantic mismatches among sources
- differing naming conventions for data values among sources

(* Source descriptions are also needed by the Wrapper for translation.) (* The sources may be in different data models. In that case, the translation between the mediated data model and the source data models is required. This should be the responsibility of the wrappers. Otherwise, the mediator would expand as new data sources were added to the integrated system. The mediator should be independent of the data sources. It is the unifying component in the system.)

3.2 Query Reformulation/Rewriting

Using the source descriptions, user query written in terms of the mediated schema is reformulated into a query that refers directly to the schemas of the sources. Goals to achieve:

- Semantical Correctness of the Reformulation: The answers obtained from the sources will be correct answers to the original query.

- Minimizing the Source Access: Sources that can not contribute any answer or partial answer to the query should not be accessed. In addition to avoiding access to redundant sources, we should reformulate the queries as specific as possible to each of the accessed sources to avoid redundant query evaluation.

3.2.1 Global As View (GAV) Approach

For each relation R in the mediated schema, a query over the source relations is written which specifies how to obtain R's tuples from the sources

3.2.2 Local As View (LAV) Approach

For each data source S, a rule over the relations in the mediated schema is written that describes which tuples are found in S. This is an application of a much broader problem called "Answering Queries using Views". (*It also applies to view design and selection in Data Warehousing.)

Query reformulation in LAV is more complex. However, it has important advantages compared to GAV: adding new sources and specifying constraints in LAV are easier.

- equivalent rewritings vs maximally-contained rewritings

- The Bucket Algorithm
- The Inverse-Rules Algorithm
- The Minicon Algorithm
- The Shared-Variable-Bucket Algorithm
- The CoreCover Algorithm
- Comparison of the Algorithms

3.2.3 Completeness and Complexity of Finding Query Rewritings

- source incompleteness
- recursive rewritings

3.2.4 Using Probabilistic Information

- for source completeness
- for overlap between parts of the mediated schema
- for overlap between information sources

3.2.5 Alternative Query Reformulation for Dynamic Information Integration

3.3 Query Optimization and Execution

What is query optimization? How is it achieved in traditional database systems? Why is it difficult in data integration systems?

- sources are autonomous; optimizer may not have any statistics or reliability info about the sources, query planning is difficult

- sources are heterogeneous; they may have different query processing capabilities
- data transfer time is not predictable due to the existence of the network environment, it is difficult to make cost estimates
- some sources may become unavailable (?)

3.3.1 Query Plan Generation

3.3.2 Query Execution

3.3.3 Adaptive Query Execution

- interleaving optimization and execution of the query

3.4 Query Translation

This is the responsibility of the source wrappers. This is a part of the execution and might go under the previous subsection. We will be connecting Query Processing section to Data Extraction section discussing the query translation stage inside query execution.

4 Materialized View Maintenance

4.1 Overview of what makes it difficult

- Large storage requirements
- Difficulty of schema changes in an established data warehouse
- Efficient updating of the view

4.2 Details of difficulties

4.2.1 Large storage requirements

- Description
- Solution: RAID

4.2.2 Difficulty of schema changes in an established data warehouse

- Description
- Solution: ¡¡needs further research from us¿¿

4.2.3 Efficient updating of views: Incremental updating vs. full refreshes

Each approach has different strengths

- Incremental update
  - Benefit: With slow-changing data sources, allows minimal view staleness without introducing high system load
  - Cost: Difficulty in knowing what source data has changed in the data sources

  - Cost: Protocols can be more complex than those for full refresh

- Full refresh
  - Benefit: Protocol can be simpler than incremental update
  - Benefit: Don't need to invent a way to discover a minimal set of what has changed in the data sources
  - Cost: In situations where ETL takes a long time, doing a full refresh may be so costly that it's done rarely, increasing data staleness.

5 Data Extraction

Combines techniques from DB and AI (NLP, Machine learning).

5.1 Sources

5.1.1 Types of sources

- Several ways to classify sources
  - structured/ semistructured/ unstructured (free text)
  - to be read by computer/ by human-being
  - cooperative/non-cooperative
  - push vs. pull?
- Specific sources:
  - full-fledged databases (relational, object-oriented, etc.)
  - legacy systems
  - knowledge bases
  - HTML/SGML
  - XML
  - structured files

5.1.2 Web sources as a special case

- Why special attention? Why extraction from web sources is especially hard?
  - Not designed for extracting data by computer (natural language)
  - On the other hand, Natural Language Processing algorithms don't work well, because many sentences are not grammatically complete
  - Form of HTML may change frequently
  - ...
- Introduction to semistructured data, OEM and XML
- Extracting structure from semistructured data
- Query languages for semistructured data and XML
- Storing semistructured data in relational DBMS

5.2 Techniques for extracting data. Wrappers

5.2.1 Wrapper. Definition and what functions it performs.

To access information from different heterogeneous data sources, we have to translate queries and data from one data model to another. This function is provided by wrappers around each individual data source. Wrapper converts queries into one or more queries understandable by the underlying data source and transforms results into the format understood by application (mediator).
Goal: make every source look like "database with limited capabilities (?)

5.2.2 Wrapper generation

- Issues
  - (see a section on data sources where for every source type it is explained what issues arise when we extract information from it)
  - Mediator systems usually require more complex wrappers than do most warehouse systems

- Ways of creating wrappers
  - **Manual**
    Why is it impractical for some sources.
    In case of Web sources:
    - big number of sources
    - new sources are added frequently
    - format of sources change
    So, high maintenance costs
  - **Semi-automatic (interactive)**
    Noted that only small part of the code deals with the specific access details of the source. the rest is common among wrappers or data transformation can be expressed in a declarative fashion (high-level). Graphical interface, programming by demonstration.
  - **Automatic**
    * site-specific or generic
    * usually needs training often supervised learning

- Tools for semi-automatic/automatic wrapper construction for structured/semistructured data
  - Template-based wrappers
  - Inductive learning techniques for automatically learning a wrapper (using labeled data)
    Inductive learning - task of computing some generalization from a set of examples
    Methods:
    * zero-order (decision tree learners)
    * first-order (inductive logic programming)
      - bottom-up/top-down approaches

- Tools for data extraction from unstructured documents
  - Using ontologies and conceptual models to extract and structure information from data-rich, unstructured documents.
  - Using heuristic approaches to find record boundaries in web documents.

5.2.3 Filters

If a wrapper returns a superset of what query wants, we can filter the results of the query

5.3 Interfaces of sources to the world

5.3.1 Local interfaces

Difficulties caused for data integration May require some integration software to be installed on the data source's computer

5.3.2 Network interfaces

- File-oriented protocols
  - Description
  - Examples (HTTP, FTP, SMB /NFS, WebDAV)
- Object-oriented protocols
  - Description
  - Examples (CORBA, COM/DCOM, Java RMI)
- Database-oriented protocols
  - Description
  - Examples (ODBC, JDBC,OLE DB)
- Text terminal protocols
  - Description
  - Difficulties introduced by format that's designed primarily to satisfy user esthetics rather than software parsing
- Directory service protocols
  - Description
  - Appearance of limited use to primarily login security
  - LDAP, NIS+

6 Systems

- TSIMMIS (Stanford project)
- Ariadne (USC)
- Araneus (Web=Bases, University of Rome, Italy)
- Infomaster
- ...

7 Open Questions and Research Issues

8 Concluding Remarks