Distributed Data Storage and Management
Part VIII: Heterogeneous Distributed Databases

Saptarshi Pyne
Assistant Professor
Department of Computer Science and Engineering
Indian Institute of Technology Jodhpur, Rajasthan, India 342030

CSL4030 Data Engineering Lectures 22, 23
September 29th, October 4th, 2023

What we discussed in the last class

- Strategies for high availability in a distributed database
  - The backup coordinator
  - The bully algorithm
  - The CAP theorem
  - The BASE properties for multi-transaction operations

Remaining sub-topics for distributed databases

- Multidatabase systems for heterogeneous distributed databases
- Distributed directory systems for managing data
  - The lightweight directory access protocol (LDAP)

Heterogeneous distributed databases

Many apps communicate across multiple sites having different databases with

- different data models
- different query languages
- different transaction management schemes
- different schemas
- attributes with same name but different meanings
- different data formats (e.g., floating point numbers with different precisions)
- different natural languages (‘Munich’ vs. ‘München’)

Hence, an additional software is required to coordinate multiple databases. Such a software is called a **multidatabase system**.

Query processing in a multidatabase system (MDBS)

App designers usually build their in-house MDBS. Off-the-shelf MDBSes are almost nonexistent.

App designers write **wrappers** to translate global queries into local site-specific queries.

MDBS vs. mediators

A **mediator** integrates data from different sources to provide a global view. They also allow read queries on the global view. E.g., useful for news feed aggregator apps like ‘inoreader’.

Mediators can not do transaction processing. MDBSes are supersets of mediators and can provide a global view along with transaction processing (reads + writes).

(MDBSes and mediators are combinedly referred to as **virtual databases**.)

Transaction processing by an MDBS

A local deadlock between local transactions at a particular site is easy to detect by the local DBMS. However, how does an MDBS detect a global deadlock?
Example: At site S1, transaction T2 follows T1, whereas, at S2, T1 follows T2.

For each site, the MDBS creates a special data item called the **ticket**. Any global transaction that wants to read/write data at a site, must first write lock the corresponding ticket.

There exist many global transaction processing protocols based on the idea of tickets.

Cloud

- **Late 1990s:** Software-as-a-service
  Vendors provided specialized software online such as the modern-day Microsoft 365.

- **Early 2000s:** Cloud computing
  Vendors started providing generic computing power where the users can run any software of their choosing.

- **Cloud storage:** Large companies such as Amazon and Google started offering their idle data storage at a subscription fee.

Widely used cloud storages

- Amazon S3 (simple storage service)
- Google Bigtable
- Apache Cassandra (formerly Facebook Cassandra, Lakshman and Malik et al.)
- Yahoo Sherpa (on which Microsoft Azure, one of the top cloud computing services, runs)

What kind of apps runs on cloud storages?

Apps that require

- high **availability** and
- high **scalability** (millions of users joining every quarter)

Cloud architecture: Case study of Yahoo PNUTS

- Yahoo Sherpa uses the **Yahoo PNUTS** architecture
- Data representation, range query
- Partition data into tables, tablets, tablets of tablets
- Partitioning function, partition table
- Tablet controller site
- Requests: put(key, value) and get(key)
- Request routers (The Google Bigtable does not involve request routers. It uses Google File System or GFS which is a distributed file system implemented on interconnected remote computers.)

Cloud architecture: Case study of Yahoo PNUTS (contd.)

- BASE transactions
- Test-and-set function
- Keeping log files up-to-date using distributed-messaging (or a distributed file system in case of Google Bigtable)
- Partitioning indices of a table into secondary indices for load balancing
- Asynchronous JavaScript and XML (AJAX) apps on the cloud

Challenges with cloud storage services

- Heavily dependent on the network infrastructure
- **Security:** Your data is held by a cloud vendor which is a separate organization
- **Legality:** Your data might be held at a different country that follows different data privacy laws. Suppose, you have developed an app in India and have been using Amazon S3 for storing your user data. Amazon S3 might put some fragments of your data in their Chinese servers. The Chinese govt. may have legal rights to claim access to that data.

References

- A. SILBERSCHATZ, H.F. KORTH, S. SUDARSHAN (2011), Database System Concepts, McGraw Hill Publications, 6th Edition.
  - Chapter 19. Distributed Databases

- Paper: Bronson et al., “TAO: Facebook’s Distributed Data Store for the Social Graph”, 2013 USENIX Annual Technical Conference (USENIX ATC ‘13).
  - Video: https://www.usenix.org/conference/atc13/technical-sessions/presentation/bronson

Thank you