DDB Functionalities by Major DMBS Products

Haibin Liu

Shcherbak Maryna

Nassrat Hatem

Outline

- Introduction
- Distributed Security
- Distributed Concurrency Control
- Distributed Query Optimization

Introduction

- Data replication
  - Reproducing data to and from sites to improve local service response time and data availability
  - Helpful when recovering lost or corrupt data
- Reliability
  - probability that the system under consideration does not experience any failures in a given time interval
- Fragmentation
  - how to fragment
  - number of copies to be replicated
  - fragment allocation

Introduction

- *Security* – Maryna
  - The type of data determines the amount of security needed
  - Certain data must be available only to authorized users and be protected from others
- *Query optimization* – Hatem
  - Minimize response time
  - Maximize throughput
  - Minimize optimization costs
- *Concurrency control* – Haibin
  - Maintains consistency of the DB in a multi-user distributed environment
  - trade-off between consistency of the DB and a high level of concurrency

Distributed Security

Security

- *What is security?*
  - "A system or set of steps that helps keep data from prying eyes by utilizing passwords, encryption and hiding data" (ComputerHope).

Security

- *Why do we need security?* [1]
  - On-line banking
    - Privacy of account data
  - E-Commerce merchants
    - Customer, order, payment data need to be kept secure
  - HR departments
    - Allow updates of personal information;
    - Protect certain managerial information from unauthorized access
  - Medical data
    - Confidentiality

Distributed DB Security

- *Multilevel security*
  - MLS/DBMS - multilevel secure database management system supports users and data at different security levels
  - Users are given different security level access
  - To view: information at or below the user's classification level
  - To update: data at the user's security level only

Security features

- *Authorization*
  - Centralized
    - Access is granted at a remote site
    - More secure
    - Disadvantages: probability of an error is higher; computationally expensive to maintain replicated user clearance tables
  - Distributed
    - Access is granted at the user's home site
    - Easy to implement and handle
    - Reliable links needed. One of the sites is compromised - the whole system is not secure anymore

Security features

- *Inference*
  - Problem description
    - "The process of users posing queries and deducing unauthorized information from the legitimate responses that they receive" [1]
    - There are data mining tools that can deduce sensitive patterns
  - Solutions proposed
    - Try to infer sensitive data ourselves
    - Build an inference controller between the DB and the data mining tool. It will be able to detect the user's motives and prevent inference

FEA-TURE	ORACLE	DB2	SQL SERVER 2005	MYSQL
Authent.	Supports	Supports	Supports	Supports
Authoriz.	Supports	Supports	Supports	Supports
Virtual Private DB (VPD)	Supports. VPD developed by Oracle	No support	No support	No support
Encryption in DB	Supports	Supports	Supports [27]	Supports [25]
Network encryp-tion	Supports	Additional IBM products (Tivoli) support network encryption.	Supports	Supports [25]
Auditing	Supports	Supports	Supports	No support

Distributed Concurrency Control

Distributed Concurrency Control

- Distributed database system: a system in which different parts of the database are stored at several sites which are interconnected into a network [2].

- The level of concurrency determines the performance of distributed DBMS

  transaction response time

  system throughput

- Attempts to find the balance between the consistency and the concurrency of distributed DBMS.

Distributed Concurrency Control

- Concurrency control mechanism in the distributed environment:

1. manages read and write requests from different transactions,

2. maintains the global consistency of the distributed database

3. ensures that the termination of the processes is not prevented by phenomena such as deadlock.

Distributed Concurrency Control

- During the concurrent operation of any set of transactions, the job of the concurrency control ensures three fundamentals [3]:

  1. Each transaction sees a consistent picture of the database;

  2. Each transaction terminates eventually;

  3. After all the transactions terminate, the final database is consistent.

Distributed Concurrency Control

- Concurrently executed transactions may interfere in operations of Read/write, write/read and write/write.

  Dirty reads: a transaction reads data written by concurrent uncommitted transactions

  Lost update: if a second transaction read an item for update after the first transaction has read it, but before the first transaction has committed

  Non-repeatable reads: a transaction re-reads data that it has previously read and finds it modified

  Phantom read: a transaction re-executes a query, finding a set of data not equal to a previous one although the search condition remains unchanged

Distributed Concurrency Control

- One design is to send all conflicting requests to a master site for final resolution. If communication costs over the network are low, this design will be beneficial.

- The other design is to have both the database and its control mechanism distributed. Each site storing data has its own local concurrency control that makes decisions on database conflicts occurring at that site.

Distributed Concurrency Control

- Distributed concurrency control algorithms generally fall into three major categories:
  - *locking algorithms*
  - *timestamp algorithms*
  - *optimistic algorithms*

- Four typical algorithms:
  - *Two-phase locking algorithm (2PL)*
  - *Wound-wait locking algorithm (WW)*
  - *Basic timestamp ordering algorithm (BTO)*
  - *Timestamp-based, optimistic concurrency control algorithm (OPT).*

Distributed Concurrency Control

- 2PL prevents conflicts because they occur using locking, resolving global deadlocks via a centralized deadlock detection scheme.
- WW is similar to 2PL, except that it uses timestamps and aborts to prevent deadlocks.
- BTO uses timestamps to order transactions and to abort transactions when conflicting accesses occur.
- OPT only checks for conflicts when a transaction is ready to commit, and it uses aborts to resolve them.

Distributed Concurrency Control

- The two-phase commit protocol is a required component for distributed databases that use synchronous replication.
  1. the coordinating node issues a transaction to all affected nodes,
  2. wait for each node to acknowledge that it is prepared to commit,
  3. issues a commit order to the affected nodes,
  4. the coordinating node issues a rollback instruction to all nodes if not all affected nodes acknowledge that they are prepared to commit after the first phase.

- There is a short period of vulnerability between commit times at different nodes during the second phase when data consistency could possibly be lost due to a node or network failure.

Distributed Concurrency Control in Major Products

- ***Microsoft***

  Microsoft facilitates customers with Microsoft Distributed Transaction Coordinator (MS DTC). It works with a two-phase commit protocol.

  In the prepare phase:

  MS DTC of the local server sends a request to all the servers used in the transaction to start a session which in turn return a success or failure acknowledgement.

  If one of the servers acknowledges the local server with a failure message, the transaction can be rolled back.

  In the commit phase:

  when all the servers return a success message, the local server sends all the remote servers a message to commit which in turn return a success or failure acknowledgement. In case of failure, the entire transaction is rolled back across the servers.

Distributed Concurrency Control in Major Products

- Microsoft SQL Server 2005 [4] :
  - “snapshot isolation” (SI)
  - “read committed snapshot isolation” (RCSI)

  Concurrency is increased in the system: the row-versioning based isolations levels allow the reader to get to a previously committed value of the row without blocking.

  SQL Server must keep multiple old versions of a row when it is updated.

Distributed Concurrency Control in Major Products

- ***Oracle***

  Oracle8 and up also employs the two-phase commit to maintain concurrency control [5]. The Oracle engine automatically takes care of the commit or rollback of all transactions.

  A major problem: when one of the nodes participating in a distributed transaction fails while the transaction is in the prepare phrase. When the failure is for a long period of time, the data locked on all the other nodes will not be available for other transactions. This will cause a lot of transactions to rollback due to deadlocks.

  Oracle DBMS further introduces an advanced queuing technique called the message queuing functionality to deal with the problem of deadlock [5].

Distributed Concurrency Control in Major Products

- ***IBM***

  IBM DB2: Distributed Database Connection Services (DDCS) [6]. for DBMS that implements the Distributed Relational Database Architecture (DRDA) application server specification. A two-phase commit is also supported.

- ***MySQL***

  MySQL 5.03 and up provides server-side support for XA transactions [7].

  XA allows multiple separate transactional resources to participate in a global transaction.

  XA also enables resource managers to join transactions, to perform two-phase commit, and to recover in-doubt transactions following a failure.

Distributed Concurrency Control in Major Products

- ***MySQL***

  Applications that use global transactions involve one or more Resource Managers (RM) and a Transaction Manager (TM) .

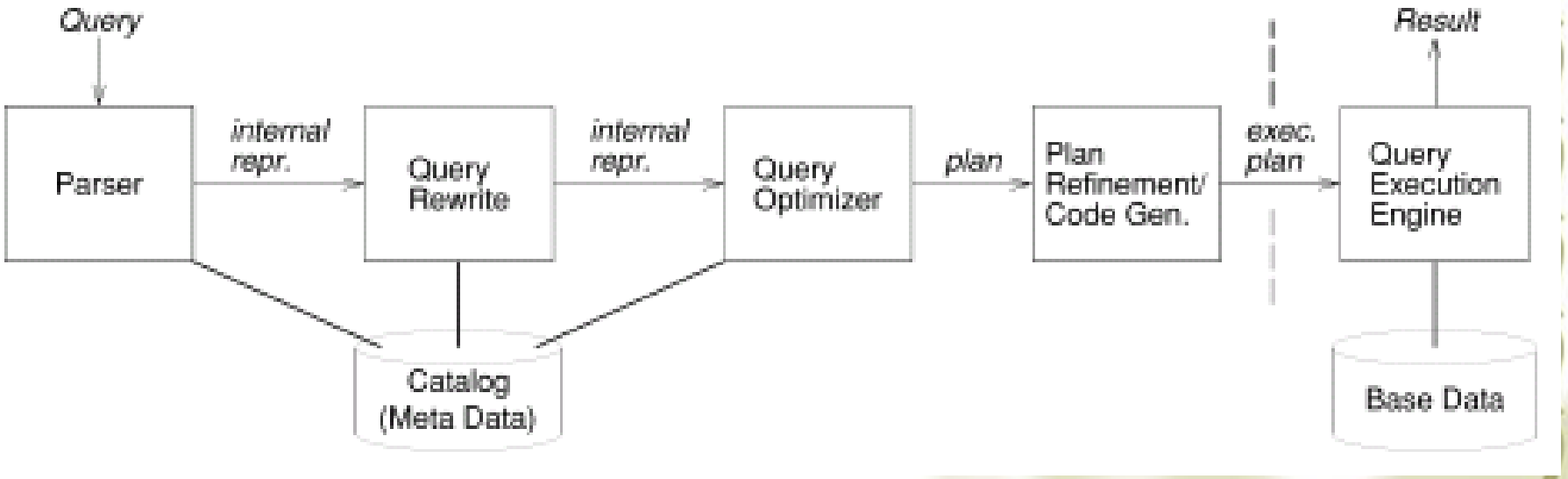
  A RM provides access to transactional resources. A TM coordinates the transactions that are part of a global transaction. It communicates with the RMs that handle each of these transactions.

  A MySQL server that handles XA transactions within a global transaction acts as a Resource Manager .

  A client program that connects to the MySQL server acts as the Transaction Manager [7].

Distributed Query Optimization and Execution Techniques

Query Processing (Overview)



[8]

Query Optimizer

- Sub-plan enumeration
- Cost Modelling
  - Classic
  - Distributed
  - Response Time
- Indexing
- Genetic algorithm
- Time to Optimize

Input: SPJ query q on relations $R_1, \ldots, R_n$
Output: A query plan for q

```
1:  for i = 1 to n do {
2:      optPlan({R_i}) = accessPlans(R_i)
3:      prunePlans(optPlan({R_i}))
4:  }
5:  for i = 2 to n do {
6:      for all S ⊆ {R_1, ..., R_n} such that |S| = i do {
7:          optPlan(S) = ∅
8:          for all O ⊂ S do {
9:              optPlan(S) = optPlan(S) ∪ joinPlans(optPlan(O), optPlan(S − O))
10:             prunePlans(optPlan(S))
11:         }
12:     }
13: }
14: return optPlan({R_1, ..., R_n})
```

[8]

Query Optimization Through Execution Techniques

- Row Blocking
- Multicast optimization
- Partitioning Data
- Semi-Joins
  - Hash Semi-joins
  - Nested Loop
  - Sort-Merge
- Top N Queries

Image Src: http://www.freewheelchairmission.org

Query Optimization Through Execution Techniques

- Query Shipping
- Data Shipping
- Hybrid Shipping

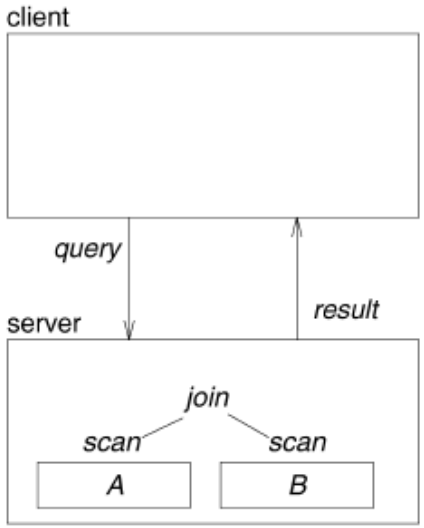


Fig. 8. Query shipping.

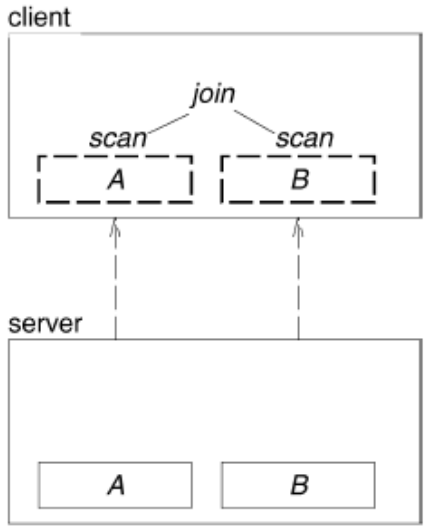


Fig. 9. Data shipping.

[8]

Optimization in Oracle

- Response Time Model
- Heuristics using Data Usage Statistics
- Bitmap Indexing
- Optimization Level
- Recognizing ∅ joins
- hash, sort-merge and nested-loop semi joins

Optimization in DB2

- Assume records are load balanced
- Exploits pipelining join queries, for first record queries
- heuristics to push selections and projections down the query tree

PostgreSQL

- Standard Optimizations
- Genetic Algorithm
  - uses an innovative algorithm that models the Human Genome System, where solution paths are modeled as chromosomes

P(t) generation of ancestors at a time t
P''(t) generation of descendants at a time t

```
+=========================================+
|>>>>>>>>>>>  Algorithm GA  <<<<<<<<<<<<<|
+=========================================+
| INITIALIZE t := 0                       |
+=========================================+
| INITIALIZE P(t)                         |
+=========================================+
| evaluate FITNESS of P(t)                |
+=========================================+
| while not STOPPING CRITERION do         |
|   +-----------------------------------+ |
|   | P'(t)  := RECOMBINATION{P(t)}     | |
|   +-----------------------------------+ |
|   | P''(t) := MUTATION{P'(t)}         | |
|   +-----------------------------------+ |
|   | P(t+1) := SELECTION{P''(t) + P(t)}| |
|   +-----------------------------------+ |
|   | evaluate FITNESS of P''(t)        | |
|   +-----------------------------------+ |
|   | t := t + 1                        | |
+===+=====================================+
```

[9]

Data Shipping

- Object Oriented DB Systems
  - ObjectStore
  - BeSS
  - SHORE
  - O_2

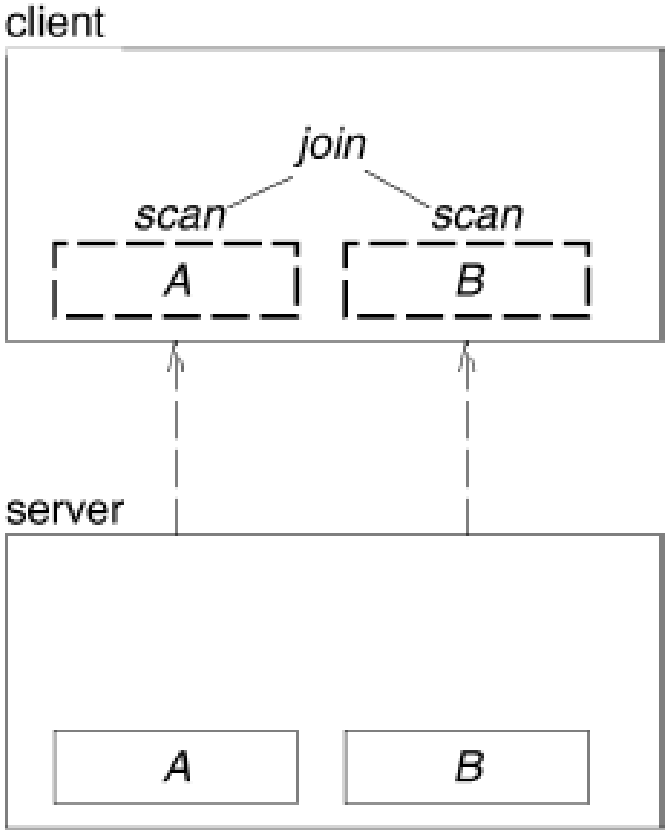


Fig. 9. Data shipping.

[8]

Hybrid Shipping

- Mix Between Query and Data Shipping
- Mostly implemented in heterogeneous DB Systems
  - UniSQL
  - ORION-2
  - SAP R/3 application systems

Conclusion

- DDBMS technology has great potential due to the rapid growth of demand.
- It would be difficult for a product to compete without distributed features.
- What we have presented highlighted the most important aspects of the functionalities in a distributed environment.
- In future work, we will investigate functionalities which are not covered by this paper.

Thank you

Questions?

References

[1] Kose I. Distributed database security. Data and network security – GYTE. Computer engineering. 2002.

[2] Balter R, Berard P, and Decitre P. Why control of the concurrency level in distributed systems is more fundamental than deadlock management. Proceedings of the First ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing 1982; 183.

[3] Bernstein PA, Goodman N. Concurrency control in distributed database systems. ACM Comput. Surv 1981; 13: 185.

[4] Delaney K, Guerrer F. Database concurrency and row level versioning in SQL server 2005. from http://www.microsoft.com/technet/prodtechnol/sql/2005/cncrrncy.mspx

[5] Oracle9i Database Concepts Release 2 (9.2). Data concurrency and consistency. from http://www.lc.leidenuniv.nl/awcourse/oracle/server.920/a96524/c21cnsis.htm

[6] Walt N. DBMS server comparison supplement. from http://www.dbmsmag.com/9611d57.html

[7] Horstmann J. Inside MySQL 5.0: A DBA's perspective. from http://corpo.mandriva.com/xwiki/bin/download/Main/Technology/mysql_wp_inside50

[8] Kossmann D. The state of the art in distributed query processing. ACM Comput Surv 2000; 32: 422-469.

[9] PostgreSQL Global Development Group. PostgreSQL: Documentation: Manuals: PostgreSQL 7.2: Genetic algorithms. 2006.