Mariposa

Anastassia Ailamaki
http://www.cs.cmu.edu/~natassa

Outline

- Why we need to rethink everything
  - All current DBMSs architected in the late 1970s
  - why the world is different now
- Mariposa
  - a wide-area distributed database system

The World is Different

- 7X 24 a serious requirement almost everywhere!
- End-to-end issue
  - RAID not the (complete) answer
  - Require wide area network replication
- The web changes everything
  - not a client-server protocol!
  - Stateless
- Warehousing is a new application area
  - typical app is data mining
  - queries run forever
The World is Different

- Multiprocessor architectures common
  - Clusters – NUMA – MPP

- The gizmo revolution is coming
  - mobile clients
  - disconnected operation
  - coke machine as a data base

The World is Different

- SQL-3
  - components (blades, extenders, OLE, Corba) in DB
  - multiple language support required
  - inheritance required

Assumptions (good for LAN?)

- Static data allocation
  - Moving objects across sites done manually

- Single administrative structure
  - Site selection done by optimizer

- Uniformity
  - Processors are of same speed
  - Any op can be done at any site
  - Everyone has disk to store anything
  - Everyone has same collection of data types
### Requirements

- 7 X 24 operation: wide area replication + mobility
  - understood by the DBMS
  - transactionally consistent
  - fastest mechanism is to move the log
- Incredible scalability
  - the biggest single system is not enough
- No global synchronization
- Total local autonomy
- Easily configurable policies

### Mariposa: a good start

- Economic paradigm for federated query processing
  - each query has a budget
  - each site is an independent contractor
  - federator acts like a general contractor, trying to solve query under the budget

### Mariposa: a good start

- Flexible heterogeneous replication
  - master-slave or peer-to-peer
  - bounded out-of-date-ness
- Mobile (and disconnected) sites ok
  - out-of-date replica
Mariposa Data Model

- A collection of fragments of a SQL-3 table
- Range partitioning
- Type conversion of data types when federated
- Each "owned" by a local DBA

Local DBMS -- Storage Model

- Store segments: fragments
  - I.e. the unit of federation
- Also the unit of movement between disk and cache (segmented storage)
- Site may buy and sell fragments
- Need “split” and “coalesce” to keep variable length segments reasonably sized

  Shades of the Burroughs B5000!

Local DBMS – System Services

- DBMS provides buffer pool, file system
  - Can provide file system abstraction easily
- Thread management from compiler
- Reliable message delivery from network
- DBMS is only application running on the machine
  - No need for a scheduler

  Very thin OS will do.....
Main ideas

- Queries and bid curves.
- Fragment as basic unit.
- In the Mariposa economic system, there are 3 kinds of entities:
  - Client
  - Broker
  - Name Server or Execution Server
- Bidding process
  - Use Rush rule system
    - on event where condition do action

Main ideas (cont…)

- Query → parser → optimizer → fragmenter → query plan → query broker.
- Optimizer is a single-site cost-based optimizer.
- Query broker follows Mariposa bidding protocol and forwards the query plan to potential bidder site.
- Each bidder responds with a bid ($cost, time)
- Broker selects bid according to budget
- Broker forwards the query plan to the selected execution site and execute the query.

The bidding process

- Query Q has budget B(t)
- Broker receives query plan (Q1…Qn) and B(t)
  - Each Qi can be 1-var restriction or join
- Expensive bid protocol
  - Broker sends requests to possible bidders
    - Answer: (Cost, Delay, Expiration)
  - Notification of winning bidders + losers
- Purchase order protocol
  - Send to most relevant site (which processes query)
    - Answer: Bill of services
  - Site may refuse
- Pros & cons of two protocols?
Replication

- Staleness factor indicates how old copy is
- Contract initial copy using select*
- Negotiate to buy updates

- How prevent monopolies (simply)?

- How discard copies?
  - Drop – sell – ignore
  - Master copy can only be sold

Experimental Results

- Mariposa vs. Static Optimizer
  - Average response time per query for static optimizer degrades faster than when using query broker

- Workload Distribution
  - The Mariposa distributes load more evenly among sites (15% variation as compare with 40% when using static optimizer)

- Query Brokering Overhead
  - Bidding adds <1% overhead on response time

Improvements

- Query decomposition into economic units of work
  - bottom-up
  - top down
  - heuristic decomposition

- Change economic plan midflight if needed
  - how to tell things have changed
  - what to do

- Partial answers are often a good idea
  - can it be done without knowing how much of the answer the user will want?
Storage Model – Open Issues

- When to coalesce and split segments
- LRU a bad model for eviction

Conclusions

- Mariposa outperforms static query optimizer
- The bidder algorithm is quite primitive
  - Crude estimate of overall resource consumption
- Provides good load balancing in case the amount of work cannot be accurately predicted
  - Because of data skew
  - The probe phase of distributed hash-join