Mariposa: A wide-area distributed database

Slides originally by Shahed Alam
Edited by Cody R. Brown, Nov 15, 2009

Why is Mariposa Important?

- Wide-area (WAN) differ from Local-area (LAN) databases.
  - Each individual site is set up differently:
    - with different access methods.
    - with different data-type extensions.
    - different site administrative structures.
  - Optimization is hard:
    - traditional optimizers do not work.
    - centralized distributed optimizers do not scale.
  - Traditional LAN assumptions do not hold for today’s WANs!

- Why use the same software for LANs?

Outline

1. Assumptions for DDBMS
2. Motivation for Mariposa
3. Economics in Mariposa
4. Mariposa architecture
5. Bidding process
6. Storage and Name resolution
7. Experiment and Conclusion

Assumptions in Traditional LAN Distributed DBMS

- Static data allocation
  - Objects can’t quickly change sites.
  - Manual transfer of data is required from site to site.
- Single administrative structure
  - Central optimizer splits queries and sends them out.
  - No site can refuse work, even under excessive load.
- Uniformity
  - Optimizer assumes all sites have same hardware, network, ample space, etc.

For WAN, these assumptions are less plausible!

Motivation

- Why not plausible?
  - Building for a non-uniformed, multi-admin WAN environment!

- For this environment we will need new goals!
  - Need new set of assumptions!
  - Requires new architecture!

Motivation: Assumptions

- Scalability to a large number of sites
  - No assumptions that will limit this!
- Data mobility
  - Easily change “home” of an object and remain available.
- No global synchronization
  - Schema changes should not force synchronization.
- Total local autonomy
  - Total control over its own resources, including what to run and store.
- Easily configurable policies
  - Easily change individual rules of sites by local administrators.
Outline

1. Assumptions for DDBMS
2. Motivation for Mariposa
3. Economics in Mariposa

Economics in Mariposa

• Apply a microeconomic paradigm for query and storage optimization:
  – clients and servers have accounts with a network bank.
  – users allocate a budget to each query.
  – query administered by broker which obtains bids.
  – fragments (objects) are the units of storage that are bought and sold and can be split or coalesced.
  – servers buy objects, advertise its services, bids on queries.

• Goal is to optimize revenue!

Economics in Mariposa

• Why a microeconomic structure?
  – Supports a large number of sites.
  – Sites can easily join and leave by buying or selling objects.
  – Data mobility: objects have no “home” just current owner which can change.

• Object replication based on payment for frequency of updates among copy holders.
  – Name servers use the same policy for metadata.

• Makes sense: sites want to maximize their profit per unit of operating. Competitive query execution.

Mariposa architecture

Outline

1. Assumptions for DDBMS
2. Motivation for Mariposa
3. Economics in Mariposa
4. Mariposa architecture

A few more details…

• Rush
  – Low level, efficient scripting rule language.
  – Included in Mariposa, done for performance reasons.
  – Storage manager, bidder, broker coded in Rush, but can be done in any language.

• Strides
  – Fragmenter groups operations in strides which can be done in parallel.
  – Sub-queries in a stride must complete before any sub-queries in next stride.
  – Used as synchronization.
Outline

1. Assumptions for DDBMS
2. Motivation for Mariposa
3. Economics in Mariposa
4. Mariposa architecture
5. Bidding process

Bidding process

- Each query has a budget \( B(t) \).
  - This is a budget which can decrease over time.
- Each query fragmented into sub-queries.
  - Can be split into parallel strides.
- Broker solves sub-queries using:
  - Expensive Bid Protocol.
  - Purchase Order Protocol.

Expensive Bid protocol

- Two phases:
  1. Request for bids:
     - Send portion of query plan being bid.
     - Bidder sends back a triplet \((C, D, E)\):
       - \( C \) = Cost
       - \( D \) = Delay (time to process query)
       - \( E \) = Expiration date of offer
  2. Notify the winning bidder (may notify losers).
- This process used only for complex queries as it is expensive (overhead: many expensive messages).
- Use Purchase order protocol for most queries.

Purchase order protocol

- Send subquery to bidder which most likely would win bid.
  - Done by keeping track of query-history.
- Site processes request and sends a “bill.”
- Can refuse bid and return to broker or pass it on.
- Cons: Probable budget deficit!
  - Since do not know bill which site will charge.

Finding bidders

- Finding bidders
  - Servers post “advertisements” with name servers.
  - Name servers store “ad-tables.”
    - Advertisements in form of “yellow pages.”
    - Several more specific ads available.
  - Brokers examine ad-tables to locate bidders.
  - Brokers remember sites that bid successfully.

Bid Acceptance

- Collection of bids for sub-queries are prefer in each stride.
- Bids are not guaranteed to be accepted.
  - Brokers must do it themselves, or inform users.
- Only simple query can perform exhaustive search of bids.
  - Non-optimal heuristic bottom-up greedy algorithm implemented for determining winner bids.
Setting the bid price

- Remember, bidder sends reply in form \((C, D, E)\) to broker.
- Cost:
  - CPU, I/O (naive), Network resource.
  - Optimization: Billing rate per fragment, adjust cost based on current load bid on hot-list items even if server does not have data.
- Delay:
  - Time to process under zero load or current load + safety factor.
- Expiration:
  - Set arbitrarily.
- Enforces load balancing.

Outline

1. Assumptions for DDBMS
2. Motivation for Mariposa
3. Economics in Mariposa
4. Mariposa architecture
5. Bidding process
6. Storage and Name resolution

Storage Management

- Manages fragments to maximize profits in local execution component.
- Buying and selling fragments.
  - Put items on hot-list for purchase.
  - Sells fragments to evict for new fragments.
- Splitting or coalescing fragments.
  - Break fragments that have high revenues, to lower copies (to redirect traffic to oneself).
- Works in harmony with Bidder:
  - Bidder bids on fragments the Storage Manager wants.
  - Declines to bid on fragments Storage Manager has not interest in, or wants to sell.

Naming and Name service

- Unlike traditional centralized name servers, Mariposa has a decentralized name registration system.
- Names are unordered sets of attributes.
- Each object has four structures for naming:
  - Internal names
  - Full names
  - Common names
  - Name contexts
    - Share certain features

Name resolution and discovery

- Every client-server has local name cache to resolve object names.
- Broker queries name-server if a match is not found.
- There exists multiple name-servers.
  - Uses advertisements to find clients.
- Broker choose name-server based on quality-of-service (staleness of metadata).
Experimental Evaluation

• Test **Purchase order vs. Expensive Bid Protocol** in LAN vs. WAN environments.
  – Only involves Broker:
    • Purchase Order: 4.52s
    • Expensive Bid: 14.08s
  – Test Expensive Bid to show how data is moved to closer sites for repeated-queries.
    – Result: all 3 tables move to site that starts the query.

• **Conclusion:** Expensive Bid Protocol only used when Purchase Order can’t be.

Conclusion

• Scheduling actions in distributed systems is difficult:
  – Large number of sites and choices per action.
  – Expensive global syncs.
  – Supporting heterogeneous systems/capabilities.
  – Timing varying load levels.
  – Site entering/leaving the system.

• **Microeconomic model well suited to these problem!**
  – Bidding allows us to adapt to environment.
  – Bidding is not too expensive!

Epilogue

• Where is Mariposa now?
  – Mariposa -> Cohera -> PeopleSoft -> Oracle