

The Microeconomic Paradigm

- Servers bid on queries
- Broker decides winners
- Each site tries to maximize its profit
- Distributed advertisement service
- Fragment - Basic unit of storage – a tradable commodity
- Servers join system by buying objects; leave by selling all objects

11/9/09

EECS 584, Fall 2009

7

The Microeconomic Paradigm

- How does this achieve our requirements?
 - Joining / leaving mechanism
 - allows scale up
 - Sites buy / sell objects as per business strategy
 - easier data mobility
 - Distributed updates, contracts for replication
 - no global synchronization
 - Each site bids on queries of interest
 - local autonomy
 - Policies easily configurable for each site

11/9/09

EECS 584, Fall 2009

8

Challenges

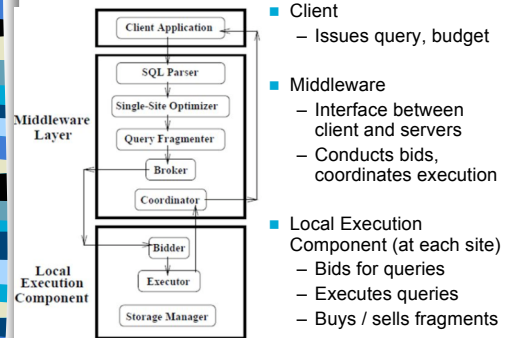
- Some queries may not be solvable!
 - Sites have freedom to choose when to bid, and how much price to demand
- Buying / Selling can be difficult
 - Site can refuse to give up objects
 - Site wanting to leave may not find buyers for its data

11/9/09

EECS 584, Fall 2009

9

Architecture



- Client
 - Issues query, budget
- Middleware
 - Interface between client and servers
 - Conducts bids, coordinates execution
- Local Execution Component (at each site)
 - Bids for queries
 - Executes queries
 - Buys / sells fragments

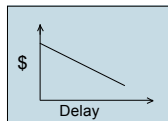
11/9/09

EECS 584, Fall 2009

10

Architecture – Client

- Client:
 - Gives Query and Budget $B(t)$ to middleware.
 - Budget – Price as a function of time (Bid Curve)



11/9/09

EECS 584, Fall 2009

11

Architecture – Middleware

- SQL Parser:
 - Parses the query
 - Requests metadata from some name server.
 - Metadata: type information, location, staleness ('quality')
 - Metadata also a tradable commodity
 - Decides name server to use based on desired quality, available budget
 - Outputs parse tree to bidder

11/9/09

EECS 584, Fall 2009

12

Architecture – Middleware

- Single Site Optimizer:
 - Prepares single site execution plan (Assumes all data is at single site)
- Fragmenter:
 - Breaks down single site plan into sub-queries based on fragments accessed
 - Groups sub-queries into *strides*
 - Queries in each *stride* can run in parallel
 - Next stride executed after previous one has completed

11/9/09

EECS 584, Fall 2009

13

Architecture – Middleware

- Broker:
 - Receives sub-query plans {Q1,Q2.....Qn}; budget B(t)
 - For each query Qi: find the most appropriate server
 - Looks up candidate sites using name servers, contacts sites for bids
 - Decides winners
 - Needs to plan so that total query is solvable within budget

11/9/09

EECS 584, Fall 2009

14

Architecture – Middleware

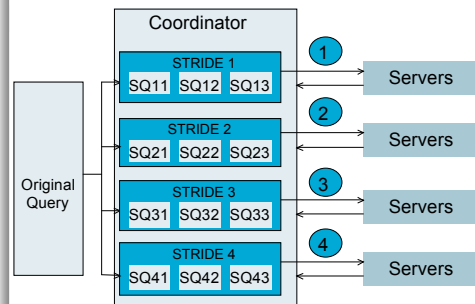
- Coordinator:
 - Coordinates execution of query strides
 - Queries within each stride executed in parallel
 - Different strides executed sequentially
 - Assembles final result, sends it to client

11/9/09

EECS 584, Fall 2009

15

Architecture – Middleware



11/9/09

EECS 584, Fall 2009

16

Architecture – Local Component

- Bidder
 - Bids for queries
 - Quotes price + expected delay
- Executor
 - Executes sub-queries given to the server
- Storage Manager
 - Manages local storage
 - Buy / Sells fragments

11/9/09

EECS 584, Fall 2009

17

The Bidding Process

- Two protocols:
 - Expensive Protocol:
 - Phase I
 - Broker sends bid request to potential bidders
 - Bidders reply with: (Ci, Di, Ei)
 - [C=cost, D=delay, E=expiration date]
 - Phase II
 - Broker notifies winners.
 - High communication cost.

11/9/09

EECS 584, Fall 2009

18

The Bidding Process

- Purchase Order Protocol:
 - Broker sends each sub query to the most likely winner.
 - If site accepts, it returns result with bill.
- Site can refuse, or pass bid on to another site.
- Cost might exceed budget

11/9/09

EECS 584, Fall 2009

19

Finding Bidders

- Servers advertise services through name servers.
- Ad Table has service details
 - types of queries, price, expected delay, etc.
- Brokers look up Ad Table to find bidders.
- Consider servers most likely to own fragments referenced in query.
- Remember previous winners, include them in current bid.

11/9/09

EECS 584, Fall 2009

20

Bid Acceptance

- Goal: Choose optimal bid combination
- Number of combinations exponential
- Greedy heuristic to decide winners
- C = Aggregate Cost
- D = Aggregate Delay
- Constraint: $C \leq B(D)$

11/9/09

EECS 584, Fall 2009

21

Setting Bid Price

- Naïve Strategy
 - Maintain billing rate for *CPU*, *I/O* per site
 - Don't bid when object doesn't exist at site.
- Improvements:
 - Maintain billing rate *per fragment*
 - Bid on fragments above threshold
 - guide direction of business
 - Adjust bid for current load
 - Helps in load balancing
 - Bid on objects which site would like to get

11/9/09

EECS 584, Fall 2009

22

Storage Management

- Goal: Maximize revenues / unit time
 - Buy / Sell fragments
 - Split / Coalesce fragments
- Buying:
 - Estimate value of fragment (using revenue history)
 - Hot List – list of desired fragments
 - offer price = value of fragment – value of alternate fragments + price received

11/9/09

EECS 584, Fall 2009

23

Storage Management

- Selling
 - offer price > value of fragment – value of alternate fragments + price received
 - To leave Mariposa – Sell all fragments
- Splitting / Coalescing fragments
 - Too few fragments – hinders parallelism
 - Too many fragments – high overhead
 - Look for balance

11/9/09

EECS 584, Fall 2009

24

Name Service

- Object names bound to list of attributes (location, types, etc)
- Ask name server for details
- Use bidding to decide which name server to ask

- Risk: Name server data may be out of date
 - Name servers periodically update their data
 - Quality of Service → rate of update
- Use {quality, price, delay} to decide bid

11/9/09

EECS 584, Fall 2009

25

Experimental Results

- Bidding system tends to choose efficient plans
 - Sites bid according to capability, load, resources
- Market dynamics prompt data movement
 - Sites buy / sell fragments based on business advantage
 - Can lead to more efficient partitioning
- With time, query plans improve

11/9/09

EECS 584, Fall 2009

26