CS 347: Parallel and Distributed Data Management

Notes 01: Introduction

Hector Garcia-Molina

In CS245: Centralized DB system

Software:
- Application
- SQL Front End
- Query Processor
- Transaction Proc.
- File Access

• Simplifications:
  - single front end
  - one place to keep locks
  - if processor fails, system fails, ...

In CS347

• Multiple processors (+ memories)
• Heterogeneity and autonomy of “components”

Multiple processors

• Opportunity for parallelism
• Opportunity for reliability
• Synchronization issues

⇨ To illustrate synchronization problems: Two Generals Problem

The one general problem (Trivial!)

G

Troops ➔ Battlefield

The two general problem:

Blue army ➔ Enemy ➔ Red army

Blue G <------------------------> Red G

messengers
Rules:
- Blue and red army must attack at same time
- Blue and red generals synchronize through messengers
- Messengers can be lost

How Many Messages Do We Need?

How Many Messages Do We Need?

How Many Messages Do We Need?

How Many Messages Do We Need?

Stated problem is Impossible!

- **Theorem**: There is no protocol that uses a finite number of messages that solves the two-generals problem (as stated here)

Probabilistic Approach?

- Send as many messages as possible, hope one gets through...
Eventual Commit

• Eventually both sides attack...

  - assume blue starts...
  - attack ASAP
  - retransmits
  - retransmits

  on my way!

BG  RG

Eventual Commit

• One message sent every time unit
• Probability of success one message is $p$
• What is probability that red commits by time $t$?

BG  RG

Eventual Commit

  - attack ASAP
  - retransmits
  - retransmits

  on my way!

BG  RG

• $C(1) = p$

• $C(2) = p + (1-p)p$

• $C(3) = p + (1-p)p + (1-p)^2p$

• $C(4) = p + (1-p)p + (1-p)^2p + (1-p)^3p$

Eventual Commit

$C(t)$

- $p$
- $-0.1$
- $-0.0$
- $0.1$
- $0.2$
- $0.3$
- $0.4$
- $0.5$
- $1.0$

$t$

$0$

$1$

$2$

$3$

$4$

$5$
**Eventual Commit**

- How expensive is protocol?
- \( E = \text{expected number of messages} \)
- Homework: compute \( E \) (function of \( p \))

**2-Phase Eventual Commit**

- Eventually both sides attack...
  - Assume blue starts...
  - \( \text{ready to attack?} \)
  - \( \text{yes, at your disposal} \)
  - \( \text{attack ASAP} \)
  - \( \text{ack} \)

**Commit Protocols**

- Will study commit protocols like these...

**Heterogeneity**

- Select new investments
  - Application
    - Stock ticker tape
    - RDBMS
    - Files
  - Portfolio
  - History of dividends, ratios...

**Autonomy**

- Example: unable to get statistics for query optimization
- Example: blue general may have mind of his (or her) own!

**So, in CS347 we study data management with multiple processors and possible autonomy, heterogeneity**

- Impact on:
  - Data organization
  - Query processing
  - Access structures
  - Concurrency control
  - Recovery
Renewed Interest in Distributed/Parallel Data Processing!
- Massive web data, manage with many computers
- How to crawl and search the web?
- Peer-to-peer systems manage huge amounts of data
- Data from many sources (e.g., comparison shopping): how to integrate?
- Sensor Networks: data generated by many sensors/devices, need to analyze
- Multi-player games (e.g., Second Life): tons of distributed data

It's the Economy, Stupid!
- Example: Multi-player games

Logistics
- TEXTBOOK: No required textbook. You’ll be expected to read several research papers.
- CLASS WEB PAGE: http://www.stanford.edu/class/cs347 Will contain homework assignments, course news, etc. Be sure to check it periodically.
- ASSIGNMENTS: about 5 homeworks
- GRADING: Homeworks: 20%, Midterm 30%, Final: 50%.

Tentative Syllabus 2014 (Part I)

<table>
<thead>
<tr>
<th>DATE</th>
<th>TOPIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday March 31</td>
<td>Introduction [N01]</td>
</tr>
<tr>
<td>Wednesday April 2</td>
<td>Data Fragmentation [N02]</td>
</tr>
<tr>
<td>Monday April 7</td>
<td>Query processing [N03]</td>
</tr>
<tr>
<td>Wednesday April 9</td>
<td>Query processing &amp; Optimization [N04]</td>
</tr>
<tr>
<td>Monday April 14</td>
<td>Concurrency Control, Failures [N05]</td>
</tr>
<tr>
<td>Wednesday April 16</td>
<td>Reliable Data Management [N06]</td>
</tr>
<tr>
<td>Monday April 21</td>
<td>Reliable Data Management [N06]</td>
</tr>
<tr>
<td>Wednesday April 23</td>
<td>Replicated Data Management [N07]</td>
</tr>
<tr>
<td>Monday April 28</td>
<td>Partitions, Entity Resolution [N11]</td>
</tr>
<tr>
<td>Wednesday April 30</td>
<td>Midterm</td>
</tr>
</tbody>
</table>
### Tentative Syllabus 2014 (Part II)

<table>
<thead>
<tr>
<th>DATE</th>
<th>TOPIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday May 5</td>
<td>Peer to Peer Systems [N08]</td>
</tr>
<tr>
<td>Wednesday May 7</td>
<td>Peer to Peer Systems [N08]</td>
</tr>
<tr>
<td>Monday May 12</td>
<td>Map-Reduce &amp; Pig [N09]</td>
</tr>
<tr>
<td>Wednesday May 14</td>
<td>Map-Reduce &amp; Pig [N09]</td>
</tr>
<tr>
<td>Monday May 19</td>
<td>Other Open Source Systems [N09b]</td>
</tr>
<tr>
<td>Wednesday May 21</td>
<td>Other Open Source Systems [N09b]</td>
</tr>
<tr>
<td>Monday May 28</td>
<td>Distributed IR [N10]</td>
</tr>
<tr>
<td>Wednesday June 2</td>
<td>Time [N12]</td>
</tr>
<tr>
<td>Wednesday June 4</td>
<td>Publish/Subscribe Systems [N13]</td>
</tr>
<tr>
<td>Friday June 6</td>
<td>8:30 am!!! FINAL EXAM</td>
</tr>
</tbody>
</table>

### Interesting New Systems
- Storm (from Twitter)
- S4 (from Yahoo)
- Cassandra (key-value store)
- Hive (SQL over Hadoop)
- Pregel (graph execution)
- ZooKeeper (replicated data)
- Spark or Spark (Berkeley?)
- H-Base
- HyRacks (UC Irvine)
- MemCache-D
- Pnuts
- Dynamo (Amazon)
- Mega-Store (Google)
- Paxos
- G-Store (UC Santa Barbara)
- Elastras (UC Santa Barbara)
- Tao (Facebook)

### Concepts you should be familiar with:
- CS245: query plan, cost estimation, join algorithms, recovery, logging,…
- Interconnection networks (bus, mesh, hypercube,…)
- Computer networks (LAN, WAN,…)

### Introductory topics
- Database architectures
- Client-server systems
- Distributed vs. parallel DB systems
- Cloud Computing

### DB architectures

1. **Shared memory**
   ```plaintext
   P   P   ...   P
   \_\_\_\_
   M
   ```

2. **Shared disk**
   ```plaintext
   P   P   ...   P
   \_\_\_\_
   M
   ```
DB architectures

(2B) Shared data storage (disk or file?)
- storage area network (SAN)
- Hadoop/Google file system

(3) Shared nothing

(4) Hybrid example

(4) Hybrid example 2
- WAN
- LAN #1
- LAN #2

(5) Unusual?
- Datacycle (Broadcast disks)
- Entire DB broadcast

also in: Microsoft SQLServer Parallel Data Warehouse
(5) Unusual — Sorting network

![Sorting network diagram]

(5) Unusual — processor per track or processor per disk

![Processor per track or disk diagram]

Related idea in Oracle Exadata "DB machine"

“Small” processors + “tiny” memories

(6) Unusual — sensor networks

![Sensor network diagram]

Data collection node

Related ideas:

Sensor
Battery

Issues for selecting architecture

- Reliability
- Scalability
- Geographic distribution of data
- Data “clusters”
- Performance
- Cost

Client-Server Systems
(or how to partition software)

![Client-Server Systems diagram]

Application
Front End
Query Processor
Transaction Processing
File Access

Client-server

Client-Server Systems
(or how to partition software)

Application
Front End
Query Processor
Transaction Processing
File Access

Client-server
**Client-Server Systems**
(or how to partition software)

- **Application**
- **Front End**
- **Query Processor**
- **Transaction Processing**
- **File Access**

---

**Transaction Servers**

- Clients ship transactions consisting of 1 or more SQL commands

  E.g., Open DataBase Connectivity (ODBC)
  (standard API)

---

**Data Servers**

- Client requests pages or records
- Popular for OODB systems

---

**Issues**

- Object granularity
- Where is data cached?
- Where is locking done?

---

**Basic Tradeoff**

- Offloading work to clients
- Data transmitted

---

**Note:** Similar issues arise when we partition software/functionality within server

- Reserve hotel room
- Data transmitted
- Where is data cached?
- Where is locking done?
Parallel or distributed DB system?
• More similarities than differences!

• Typically, parallel DBs:
  - Fast interconnect
  - Homogeneous software
  - High performance is goal
  - Transparency is goal

• Typically, distributed DBs:
  - Geographically distributed
  - Data sharing is goal (may run into heterogeneity, autonomy)
  - Disconnected operation possible

Cloud Computing
• Is CC just a marketing term??
  - utility (like power)
  - data or CPU cycles?
  - many processors, many storage units
  - business model

Is CC a subset, superset, disjoint from, or overlaps with:
• grid computing
• distributed computing
• Web 2.0
• Cluster Computing
• Peer-to-peer computing
• software as a service
• client-server computing
• data center as a computer
• massively parallel computing

Clash of the Clouds (Economist April 4, 2009)
CC Issues

- Customer lock-in
- Privacy
- Standards
- Software licensing

Next

- How to describe distributed data
- Query processing in parallel DBs
- Query processing in distributed DBs

Query processing in parallel DBs:

- Typically: we can distribute/ partition/ sort... data to make certain DB operations (e.g., Join) fast

Query processing in distributed DBs:

- Typically: we are given data distribution; we need to find query processing strategy to minimize cost (e.g., communication cost)