Scalable Clusters: Architecture and Software
Lecture 12: Case Studies

April 14, 1998

Outline

• Announcements
  – project presentations: 04/28/98
  – final reports due 6pm, 05/05/98

• This lecture: Putting it all together
  – case studies
    • Berkeley NOW: (killer networks, fast communication) [ Linda ]
    • Beowulf: (cost-effective parallel computing) [ Perrin ]
    • Inktomi: (cost-effectiveness, scalability, availability)
  – conclusion
    • course summary
    • future directions
Inktomi

- Bay Area startup: February 1996
  - Eric Brewer (Assistant Professor)
  - Paul Gauthier (Graduate Student)

- Products
  - “HotBot” Web Search Engine
    - indexes 54 million full-text documents (~1997)
    - web crawler reads 10 million documents a day
    - claimed more complete than Altavista
  - Traffic Server
    - internet cache for web pages
    - 3488 ops/sec in a half-terabyte configuration

- Technology
  - cluster of commodity workstations and PCs, Myrinet, disk
  - one generation lag: price, reliability advantages

HotBot: Internals

- 3 components (numbers as of mid-1997)
  - web crawler
    - searches the web for pages (documents) by following hypertext links
    - multithreaded: ~10,000 threads
    - cluster of ~20 Pentium Pro’s and Myrinet
  - database
    - stores information about the pages
    - off-the-shelf 4 GB disks in RAID configuration: about 400 of these
    - database is partitioned: documents are randomly distributed
  - web server (HTTP interface)
    - handles users’ queries on the database
    - cluster of ~50 UltraSparcs and Myrinet + reserve capacity (for burstiness)
      - nodes are heterogeneous: 5 -> 10 -> 22
      - custom web server: does not matter that they do not run NT
      - query involves all nodes: each node runs 50-80 threads
    - user preferences and ad revenues tracked using an Informix database
HotBot: Web Server Organization

static partitioning of read-only data onto disks, main memory

4/14/98

HotBot: A Simple Query (Step 1)

Request received by the manager
Manager selects a worker to process the request

4/14/98
### HotBot: A Simple Query (Step 2)

Request redirected to selected worker

![Diagram of network with Up: 1, 2, 3, 4]

- Internet
- ethernet
- Myrinet

- Memory
- Disks

4/14/98

---

### HotBot: A Simple Query (Step 3)

Worker sends queries to all nodes in the server

Each node works on its portion of the database and sends back response

![Diagram of network with Up: 1, 2, 3, 4]

- Internet
- ethernet
- Myrinet

- Memory
- Disks

4/14/98
HotBot: A Simple Query (Step 4)

Responses collected by the designated worker
Worker returns response to client (customize for user preferences, browser type)

HotBot: Fault Management

Original design: Node 2 transparently takes over responsibility
New design: Just pretend that portion of database is temporarily unavailable
Inktomi: Advantages from Clustering

• Price/Performance
  – commodity components
    • ~$2M versus $15M (Altavista)
    • reason: low-margin components, single database versus replicated (6 SMPs)
  – performance less crucial: not limited by communication overhead, BW

• Growth and Scalability
  – system can grow with demand: 5 -> 10 -> 22 UltraSparcs
    • no need to guess the load, just ADAPT
  – CPU/Memory/Disk can be tuned separately

• Fault management
  – hot-swap disks, machines
  – Inktomi moved from Berkeley to San Jose without service disruption

• Inktomi’s edge is software for clusters
  – multithreading, coordinating communication, fault management, etc.

Course Summary

Lecture 12: Case Studies
- Beowulf, Berkeley NOW, Inktomi

Lecture 11: Resource Coordination
- Work Stealing, Implicit Scheduling, DCS

Lecture 10: Fault-tolerance
- Horus, Calypso, WolfPack

Lectures 8, 9: Software DSM
- TreadMarks, AURC, CRL, Shasta, Munin

Lectures 6, 7: Messaging Layers
- AM, FM, VMMC-2, U-Net, VIA

Lecture 5: Cluster NIs
- SHRIMP, Hamlyn, Memory Channel

Lecture 4: Commodity Machines
- T3E, Challenge, Origin

Lecture 3: Full-custom Machines
- *T, iWarp, J-Machine, Alewife

Lectures 1, 2: Technology Trends
Future Directions: Cluster Hardware

- **Processors**
  - track microprocessor development curve
    - wider superscalar, more speculation, multithreaded processors

- **Networks**
  - switched, lower latency, higher bandwidth
  - improved reliability

- **Network interfaces**
  - VIA: protected user-level access to the network
    - remote DMA operations in addition to messaging operations
    - remote atomic operations?
  - programmable/reconfigurable devices
    - enables application-specific firmware
  - unfortunately, still connected to a peripheral (I/O) bus
  - good news: performance of these buses is improving (I2O)

Future Directions: Cluster Middleware

- **Messaging**
  - predictable, high performance on multi-party traffic
  - improved flow-control algorithms for dynamic process model

- **Shared memory**
  - software shared memory protocols
    - weaker consistency models, hybrid hardware-software approaches
  - use of application information to offset overhead of underlying primitives

- **Fault-tolerance and availability**
  - exploit improved hardware reliability, *cooperative* resources
  - 24×7 availability without sacrificing performance

- **Resource coordination**
  - interactive response + good performance for sequential, parallel jobs
  - physical view: heterogenous pool of PCs, different networks
  - logical view: dedicated machine with predictable behavior (QoS)
Future Directions: Cluster Applications

• First generation:
  – clusters can be used as viable alternatives to custom parallel machines
    • good performance on some applications
  – arguments of scalability and availability

• Second generation:
  – enlarge domain of applications that are efficiently supported on clusters
    • requires advances in cluster hardware and middleware
  – really demonstrate cost-effectiveness, scalability, and availability
  – “killer app” for clusters?