Distributed Systems

Introduction

Paul Krzyzanowski
pxk@cs.rutgers.edu

What can we do now that we could not do before?

Technology advances

Networking

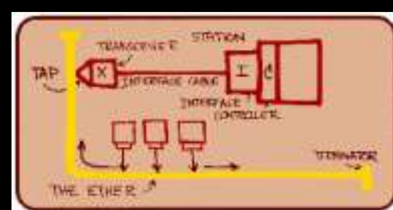
Processors

Memory

Storage

Protocols

Networking: Ethernet - 1973, 1976



June 1976: Robert Metcalfe presents the concept of *Ethernet* at the National Computer Conference

1980: Ethernet introduced as de facto standard (DEC, Intel, Xerox)

Network architecture

348 - >35,000x faster

LAN speeds
- Original Ethernet: 2.94 Mbps
- **1985**: thick Ethernet: 10 Mbps
  1 Mbps with twisted pair networking
- **1991**: 10BaseT - twisted pair: 10 Mbps
  Switched networking: **scalable bandwidth**
- **1995**: 100 Mbps Ethernet
- **1998**: 1 Gbps (Gigabit) Ethernet
- **1999**: 802.11b (wireless Ethernet) standardized
- **2001**: 10 Gbps introduced
- **2005**: 100 Gbps (over optical link)

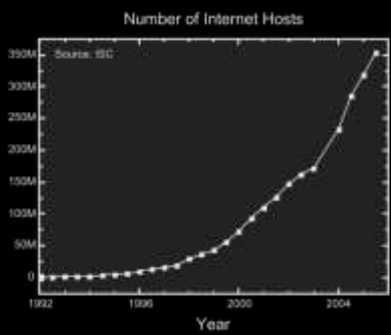
Network Connectivity

570million more hosts

Then:
- Large companies and universities on Internet
- Gateways between other networks
- Dial-up bulletin boards
- 1985: 1,961 hosts on the Internet

Now:
- One Internet (mostly)
- 2008: 570,937,778 hosts on the Internet
- Widespread connectivity
  High-speed WAN connectivity: 1- >50 Mbps
- Switched LANs
- Wireless networking

Network Connectivity



Computing power

Computers got…

- Smaller
- Cheaper
- Power efficient
- Faster

Microprocessors became technology leaders

Computing Power

1974: Intel 8080
2 MHz, 6K transistors

2004: Intel P4 Prescott
3.6 GHz, 125 million transistors

2006: Intel Core 2 Duo
2.93 GHz, 291 million transistors

Storage: RAM

9,000x cheaper
4,000x more capacity

year	$/MB	typical
1977	$32,000	16K
1987	$250	640K-2MB
1997	$2	64MB-256MB
2007	$0.06	512MB-2GB+

Storage: disk

129,000x cheaper in 20 years
18,750x more capacity

Recording density increased over 60,000,000 times over 50 years

1977: 360KB floppy drive – $1480
$11,529 / GB (but 2,713 5½" disks!)

1987: 40 MB drive for – $679
$16,975K / GB

2008: 750 GB drive for – $99
$0.13 / GB

Music Collection

4,207 Billboard hits

- 18 GB
- Average song size: 4.4 MB

Today

- Download time per song @12.9 Mbps: 3.5 sec
- Storage cost: $2.38

Approx 20 years ago (1987)

- Download time per song, V90 modem @44 Kbps: 15 minutes
- Storage cost: $305,55

Protocols

Faster CPU →

more time for protocol processing

- ECC, checksums, parsing (e.g. XML)
- Image, audio compression feasible

Faster network →

→ bigger (and bloated) protocols

- e.g., SOAP/XML, H.323

Why do we want to network?

- Performance ratio
  - Scaling multiprocessors may not be possible or cost effective
- Distributing applications may make sense
  - ATMs, graphics, remote monitoring
- Interactive communication & entertainment
  - work and play together: email, gaming, telephony, instant messaging
- Remote content
  - web browsing, music & video downloads, IPTV, file servers
- Mobility
- Increased reliability
- Incremental growth

Problems

Designing distributed software can be difficult

- Operating systems handling distribution
- Programming languages?
- Efficiency?
- Reliability?
- Administration?

Network

- disconnect, loss of data, latency

Security

- want easy and convenient access

Building and classifying distributed systems

Flynn's Taxonomy (1972)

number of <u>instruction streams</u>
and <u>number of data streams</u>

SISD

- traditional uniprocessor system

SIMD

- array (vector) processor
- Examples:
  - GPUs – Graphical Processing Units for video
  - APU (attached processor unit in Cell processor)
  - SSE3: Intel's Streaming SIMD Extensions
  - PowerPC AltiVec (Velocity Engine)
  - GPGPU (General Purpose GPU): AMD/ATI, Nvidia
  - Intel Larrabee (late 2008?)

MISD

- Generally not used and doesn't make sense
- Sometimes (rarely!) applied to classifying redundant systems

MIMD

- multiple computers, each with:
  - program counter, program (instructions), data
- parallel and distributed systems

Subclassifying MIMD

memory

- shared memory systems: <u>multiprocessors</u>
- no shared memory: networks of computers, <u>multicomputers</u>

interconnect

- bus
- switch

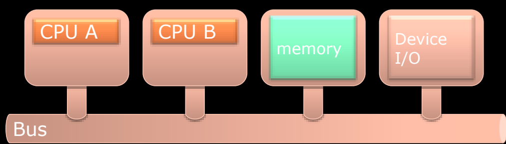
delay/bandwidth

- tightly coupled systems
- loosely coupled systems

Bus-based multiprocessors

SMP: Symmetric Multi-Processing
All CPUs connected to one bus (backplane)

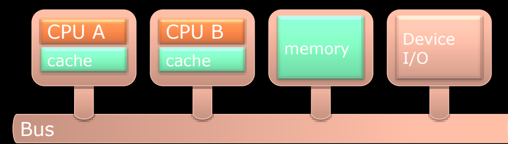
Memory and peripherals are accessed via shared bus.
System looks the same from any processor.



Bus-based multiprocessors

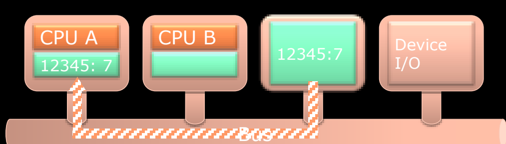
Dealing with bus overload
- add local memory

CPU does I/O to cache memory
- access main memory on cache miss



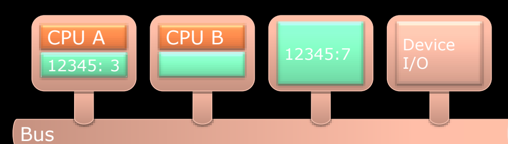
Working with a cache

CPU A reads location 12345 from memory



Working with a cache

CPU A modifies location 12345

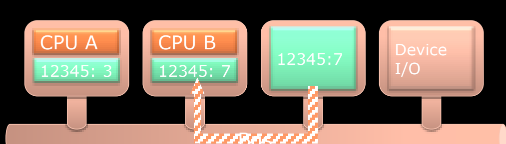


Working with a cache

CPU B reads location 12345 from memory

Gets old value

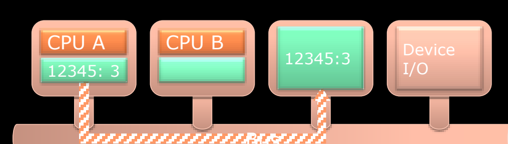
Memory not coherent!



Write-through cache

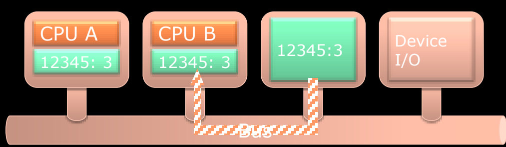
Fix coherency problem by writing all values through bus to main memory

CPU A modifies location 12345 – ***write-through***
main memory is now coherent



Write-through cache … continued

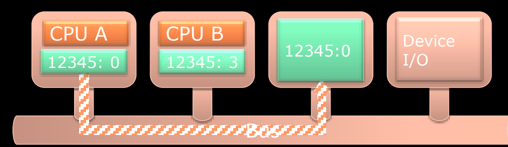
CPU B reads location 12345 from memory
- loads into cache



Write-through cache

CPU A modifies location 12345
- write-through

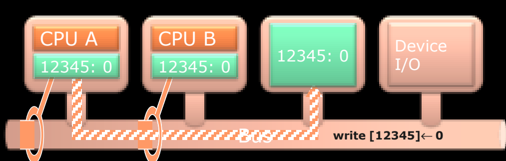
Cache on CPU B not updated
Memory not coherent!



Snoopy cache

Add logic to each cache controller:
monitor the bus

Virtually all bus-based architectures use a snoopy cache

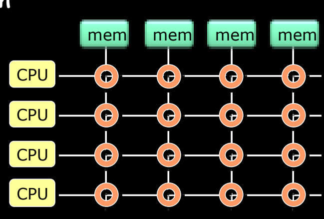


Switched multiprocessors

- Bus-based architecture does not scale to a large number of CPUs (8+)

Switched multiprocessors

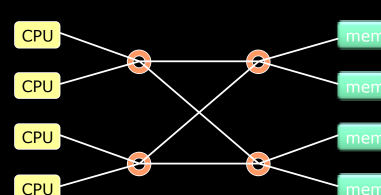
Divide memory into groups and connect chunks of memory to the processors with a **crossbar switch**



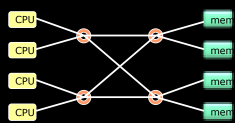
n^2 crosspoint switches – expensive switching fabric

Crossbar alternative: omega network

Reduce crosspoint switches by adding more switching stages



Crossbar alternative: omega network



with n CPUs and n memory modules:
need $\log_2 n$ switching stages,
each with $n/2$ switches

Total: **$(n\log_2 n)/2$ switches**.

- Better than n^2 but still a quite expensive
- delay increases:
  1024 CPU and memory chunks
  overhead of 10 switching stages to memory and 10 back.

NUMA

- Hierarchical Memory System
- Each CPU has local memory
- Other CPU's memory is in its own address space
  - slower access
- Better *average* access time than omega network if most accesses are local
- Placement of code and data becomes difficult

NUMA

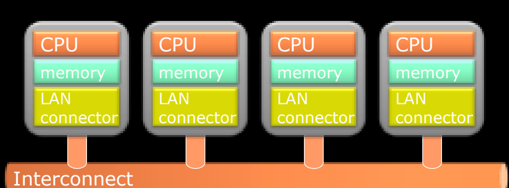
- SGI Origin's ccNUMA
- AMD64 Opteron
  - Each CPU gets a bank of DDR memory
  - Inter-processor communications are sent over a HyperTransport link
- Linux 2.5 kernel
  - Multiple run queues
  - Structures for determining layout of memory and processors

Bus-based multicomputers

- No shared memory
- Communication mechanism needed on bus
  - Traffic much lower than memory access
  - Need not use physical system bus
    - Can use LAN (local area network) instead

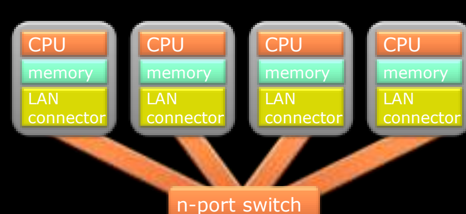
Bus-based multicomputers

Collection of workstations on a LAN



Switched multicomputers

Collection of workstations on a LAN



Software

Single System Image

Collection of *independent* computers that appears as a *single system* to the user(s)

- *Independent*: autonomous
- *Single system*: user not aware of distribution

Distributed systems software

Responsible for maintaining single system image

You know you have a distributed system when the crash of a computer you've never heard of stops you from getting any work done.

– *Leslie Lamport*

Coupling

Tightly versus loosely coupled software

Tightly versus loosely coupled hardware

Design issues: Transparency

High level: hide distribution from users
Low level: hide distribution from software

- **Location transparency**: users don't care where resources are
- **Migration transparency**: resources move at will
- **Replication transparency**: users cannot tell whether there are copies of resources
- **Concurrency transparency**: users share resources transparently
- **Parallelism transparency**: operations take place in parallel without user's knowledge

Design issues

Reliability

- **Availability**: fraction of time system is usable
  - Achieve with redundancy
- **Reliability**: data must not get lost
  - Includes security

Performance

- Communication network may be slow and/or unreliable

Scalability

- Distributable vs. centralized algorithms
- Can we take advantage of having lots of computers?

Service Models

Centralized model

- No networking
- Traditional time-sharing system
- Direct connection of user terminals to system
- One or several CPUs
- Not easily scalable
- Limiting factor: number of CPUs in system
  - Contention for same resources

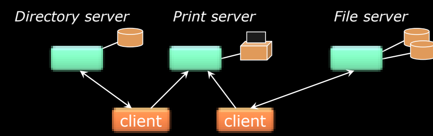
Client-server model

Environment consists of **clients** and **servers**

Service: task machine can perform

Server: machine that performs the task

Client: machine that is requesting the service



Workstation model

assume client is used by one user at a time

Peer to peer model

- Each machine on network has (mostly) equivalent capabilities
- No machines are dedicated to serving others
- E.g., collection of PCs:
  - Access other people's files
  - Send/receive email (without server)
  - Gnutella-style content sharing
  - SETI@home computation

Processor pool model

What about idle workstations (computing resources)?

- Let them sit idle
- Run jobs on them

Alternatively…

- Collection of CPUs that can be assigned processes on demand
- Users won't need heavy duty workstations
  - GUI on local machine
- Computation model of Plan 9

Grid computing

Provide users with seamless access to:

- Storage capacity
- Processing
- Network bandwidth

Heterogeneous and geographically distributed systems

Grid Computing

- Provide users with seamless access to:
  - Storage capacity
  - Processing
  - Network bandwidth
- Heterogeneous and geographically distributed systems
- Build a "supercomputer" on the fly via networked, loosely coupled computers

Cloud Computing

Resources are provided as a network (Internet) service

- Software as a Service (SaaS)
- Google Apps
- Salesforce.com

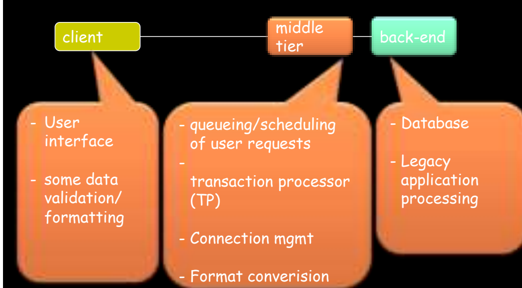
Multi-tier client-server architectures

Two-tier architecture

Common from mid 1980's-early 1990's

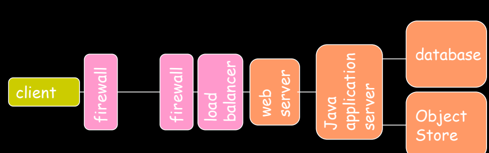
- UI on user's desktop
- Application services on server

Three-tier architecture



Beyond three tiers

Most architectures are multi-tiered



The end.