Hálózatba kapcsolt erőforrás platformok és alkalmazásaik

Simon Csaba
TMIT
2018

Legal Download in EU

- http://index.hu/kultur/2013/10/30/uhd/

Zárójel: distributed storage?

CAP

CAP tétel

- *CAP = Consistency, Availability, Partition Tolerance*
  - Elosztott rendszerekben
  - Egyszerre nem lehet mind a három célt elérni

- http://www.julianbrowne.com/article/viewer/brewers-cap-theorem

CAP tulajdonságok

- CAP - mozaikszó
  - Konzisztencia (consistency)
  - Rendelkezésre állás (availability)
  - Partíció tolerancia (partition tolerance)
- Elosztott rendszerekben egyidejűleg legfeljebb két fenti tulajdonság garantálható teljesen

- A. Fox, E. Brewer, „Harvest, yield, and scalable tolerant systems”. *Workshop on Hot Topics in Operating Systems, 1999*.

http://shark-collection.googlecode.com/svn/docs/Technique/Architect/%5B8%5DHarvest,%20Yield,%20and%20Scalable%20Tolerant%20Systems.pdf

CAP tétel

- „C" - Konzisztencia: bármely időpontban, bármely csomóponttól lekérdezhetjük bármelyik „atomi" adatot ugyanaz az érték
- „A" - Rendelkezésre állás: minden működő csomóponthoz érkező kérésre tud válaszolni
- „P" - Partíció tolerancia: adott kérésre hálózati partíció esetén is végre lehet hajtani az írás vagy olvasás műveletet
- **CAP tétel**: elosztott rendszerben a hálózat partíciója esetén a rendszer műveletei nem lesznek atomiak és/vagy az adategységei elérhetetlenek lesznek

CAP elvárások a gyakorlatban

- CAP tétel szerint nem valósítható meg egy működő elosztott adatbázis
  - Ugyanis az nagyszámú, egymástól független elemből állna
  - Ez meg „behozza” a particionálás veszélyét a rendszerbe
- Ugyanakkor globális cégek működtetnek ilyen rendszereket. Hogyan?
  - A három feltétel egyikének lazításával
  - Pl: ha a formális konzisztencia követelménynél gyengébb elvárás megengedése
    - Amazon Dynamo - „A”, „P” tulajdonságokat garantálja
  - Pl. ha a késleltetés minimalizálását helyezik előtérbe
    - Yahoo PNUTS – „P” tulajdonságot minden esetben garantálja

Blockchain alapú P2P megoldások

Zeronet

- Dynamic Web
- P2P
- Integrated BitCoin

- ZeroNet and IPFS: uncensorable auto-scaling BitTorrent powered websites
- ZeroNet is created for fast, dynamic websites, IPFS is more like a storage solution

http://bitcoinist.com/interview-developer-zeronet/

Ping: zeronet.io Go

Pinging zeronet.io, IP: 104.156.231.236 located in United States from multiple locations:

Location	ISP	Loss	Sent	Last	Avg	Best	Worst	StDev	MTR	Chart
Canada, BC, Vancouver	Telus	0%	100	26.1	25.95	25.37	26.5	0.3	show	
Canada, BC, Vancouver	Shaw	0%	100	33.43	36.02	32.41	126.44	9.55	show	
USA, WA, Seattle	PCCW	0%	100	18.16	18.18	17.92	18.62	0.14	show	
USA, WA, Seattle	Softlayer	0%	100	17.98	18.08	17.86	18.79	0.15	show	
	cane-fmt2	0%	100	1.51	1.62	1.22	11.75	1.05	show	
	aNET	0%	100	13.07	13.15	12.98	15.39	0.24	show	
	ayer	0%	100	38.66	38.82	38.56	39.88	0.23	show	
	t	0%	100	51.46	51.7	51.07	63.37	1.67	show	
	rietta/RamNode	0%	100	63.9	63.97	63.82	64.72	0.14	show	
	ouse/RamNode	0%	100	78.08	78.11	77.92	78.71	0.12	show	
	vIT/RamNode	0%	100	141.22	141.26	141.07	141.78	0.12	show	
	lu/BuyVM	0%	100	150.35	150.59	150.13	159.24	1.27	show	
	derService	0%	100	151.5	151.33	151.16	152.02	0.14	show	
Italy, Milan	Prometeus	0%	100	158.14	158.39	158.09	161.52	0.52	show	
Russia, Moscow	WebDC/FirstVDS	0%	100	202.2	203.77	201.74	206.64	1.86	show	
Russia, Tomsk	Tomgate/Berihoster	0%	100	255.7	255.48	255.07	256.24	0.22	show	
Singapore	Digital Ocean	0%	100	168.26	168.28	168.07	168.85	0.13	show	
Japan, Tokyo	Vultr	0%	100	108.88	108.9	108.72	109.45	0.13	show	
Australia, Sydney	Vultr	0%	100	167.14	167.01	166.75	168.34	0.22	show	
China, Guizhou	China Telecom	100%	100	-	-	-	-	-	show	
China, Henan	China Unicom	100%	100	-	-	-	-	-	show	
China, Guangzhou	Tencent	100%	100	-	-	-	-	-	show	
China, Beijing	Aliyun	100%	100	-	-	-	-	-	show	
China, Shenzhen	Aliyun	100%	100	-	-	-	-	-	show	
China, Anhui	China Mobile	100%	100	-	-	-	-	-	show	
China, Jiangsu	China Mobile	100%	100	-	-	-	-	-	show	
China, Jiangsu	China Unicom	100%	100	-	-	-	-	-	show	
China, Jiangsu	China Telecom	100%	100	-	-	-	-	-	show	
China, Shanghai	Aliyun	100%	100	-	-	-	-	-	show	

ZeroNet
@HelloZeroNet

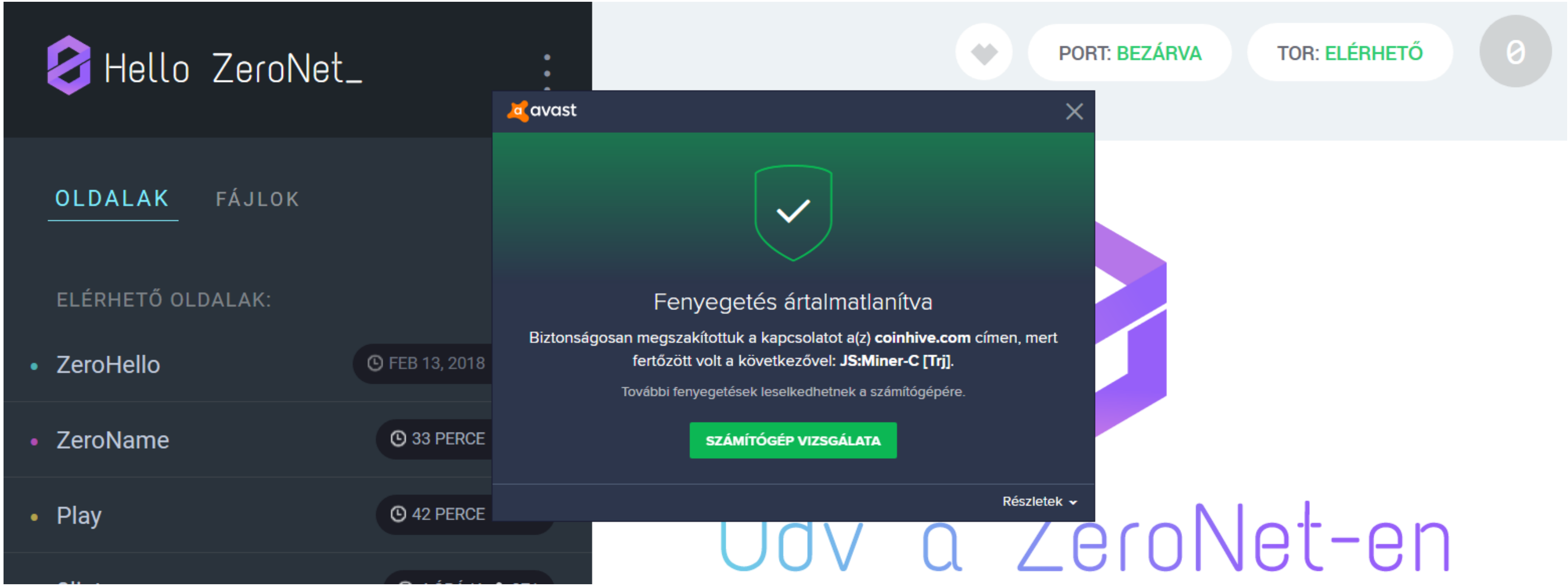
Követés

ZeroNet.io webpage got blocked in China: We are officially marked as threat to government censorship.

13:06 - 2017. okt. 13.

Kínai hozzáférés?

Vírusveszély!



IPFS

- Interplanetary File System
  - combines the concept of linking the distributed file sharing to a file system
  - Hashing individual files

- Ethereum
  - a single shared computer that is run by the network of users and on which resources are parceled out and paid for by *Ether*
  - *finance, the internet-of-things, farm-to-table produce, electricity sourcing and pricing, and sports betting*

- Filecoin
  - incentive for storage, on top of IPFS
  - distributed electronic currency similar to Bitcoin
  - proof-of-retrievability component, which re-quires nodes to prove they store a particular file

Ethereum - https://www.cryptokitties.co/

SWARM

- Distributed **storage** platform and content distribution service
  - a native base layer service of the ethereum web 3 stack
- Features
  - DDOS-resistant, zero-downtime, fault-tolerant, censorship-resistant
  - self-sustaining due to a built-in incentive system
    - uses peer to peer accounting
    - allows trading resources for payment

http://www.epointsystem.org/~nagydani/homepage

Összefoglalás

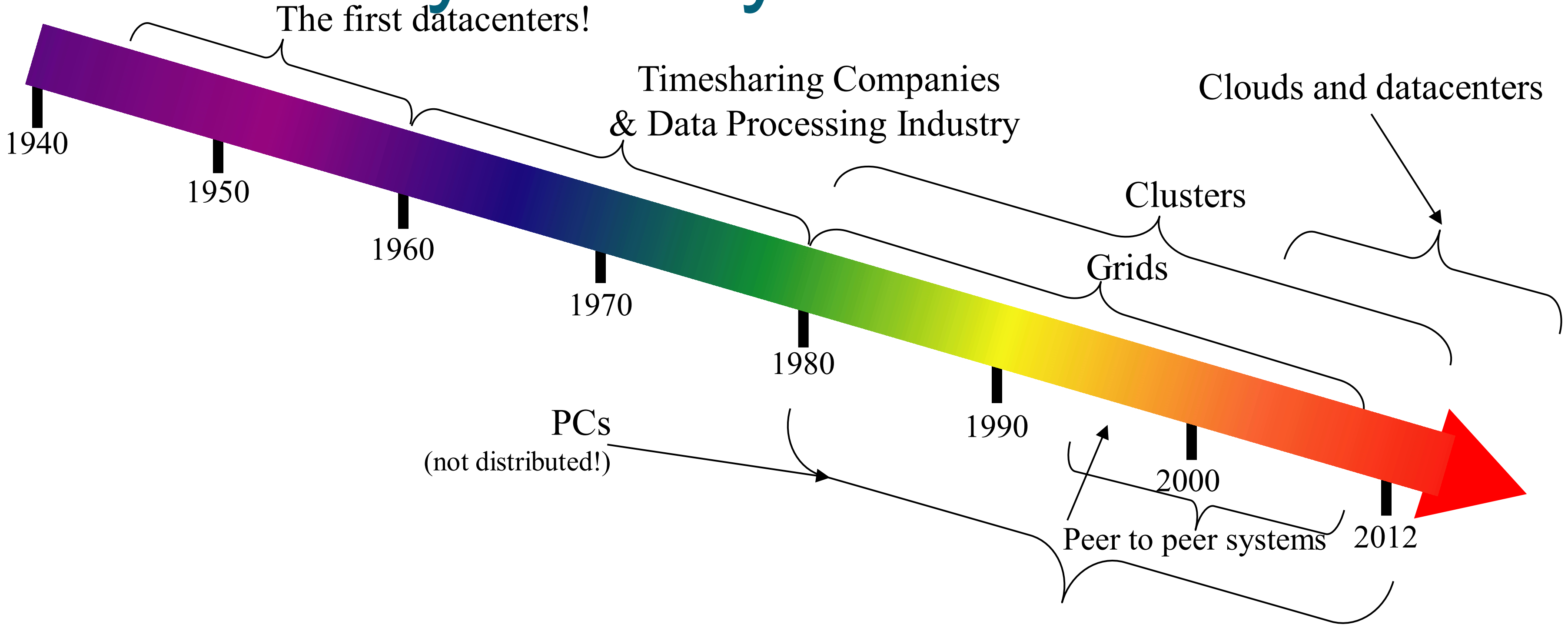
- P2P > fájlcserélés
  - Lehetőség van a résztvevők identitásának elrejtésére
  - DHT alapú
- Blockchain alapú új javaslatok (2014/15-től kezdődően)
  - Distributed storage
  - Cenzorship
  - Incentive
  - Dynamic

Klaszterek

P2P Computing vs Grid Computing

- Differ in Target Communities
- Grid system deals with more complex, more powerful, more diverse and highly interconnected set of resources than P2P.
- VO

"A Cloudy History of Time"

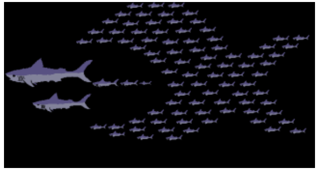


"A Cloudy History of Time"

First large datacenters: ENIAC, ORDVAC, ILLIAC
Many used vacuum tubes and mechanical relays

1940
1950
1960
1970
1980
1990
2000
2012

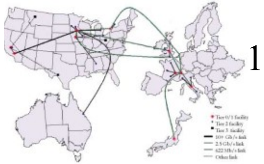
Berkeley NOW Project
Supercomputers
Server Farms (e.g., Oceano)



P2P Systems (90s-00s)
- Many Millions of users
- Many GB per day

Data Processing Industry
- 1968: $70 M. 1978: $3.15 Billion

Timesharing Industry (1975):
- Market Share: Honeywell 34%, IBM 15%,
- Xerox 10%, CDC 10%, DEC 10%, UNIVAC 10%
- Honeywell 6000 & 635, IBM 370/168,
  Xerox 940 & Sigma 9, DEC PDP-10, UNIVAC 1108



Grids (1980s-2000s):
- GriPhyN (1970s-80s)
- Open Science Grid and Lambda Rail (2000s)
- Globus & other standards (1990s-2000s)

Clouds

Parelell computing

Multiprocessing

- Flynn's Taxonomy of Parallel Machines
  - How many Instruction streams?
  - How many Data streams?
- SISD: Single I Stream, Single D Stream
  - A uniprocessor
- SIMD: Single I, Multiple D Streams
  - Each "processor" works on its own data
  - But all execute the same instrs in lockstep
  - E.g. a vector processor or MMX

Flynn’s Taxonomy

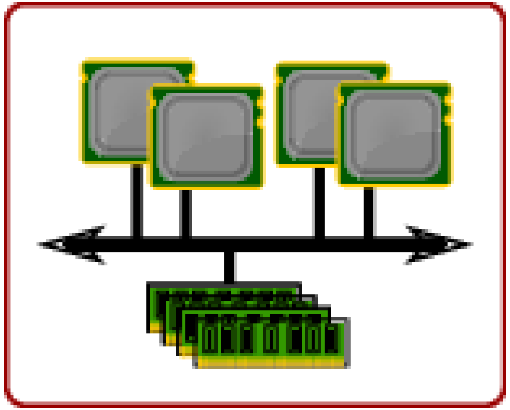
- MISD: Multiple I, Single D Stream
  - Not used much
  - Stream processors are closest to MISD
- MIMD: Multiple I, Multiple D Streams
  - Each processor executes its own instructions and operates on its own data
  - This is your typical off-the-shelf multiprocessor (made using a bunch of “normal” processors)
  - Includes multi-core processors

Multiprocessors

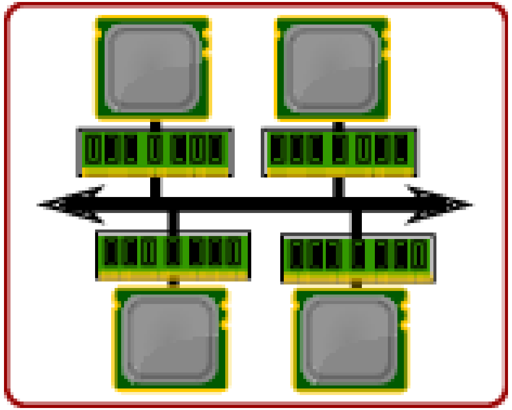
- Why do we need multiprocessors?
  - Uniprocessor speed keeps improving
  - But there are things that need even more speed
    - Wait for a few years for Moore's law to catch up?
    - Or use multiple processors and do it now?
- Multiprocessor software problem
  - Most code is sequential (for uniprocessors)
    - MUCH easier to write and debug
  - Correct parallel code very, very difficult to write
    - ***Efficient*** and correct is even harder
    - Debugging even more difficult (Heisenbugs)

MIMD Multiprocessors

Centralized Shared Memory

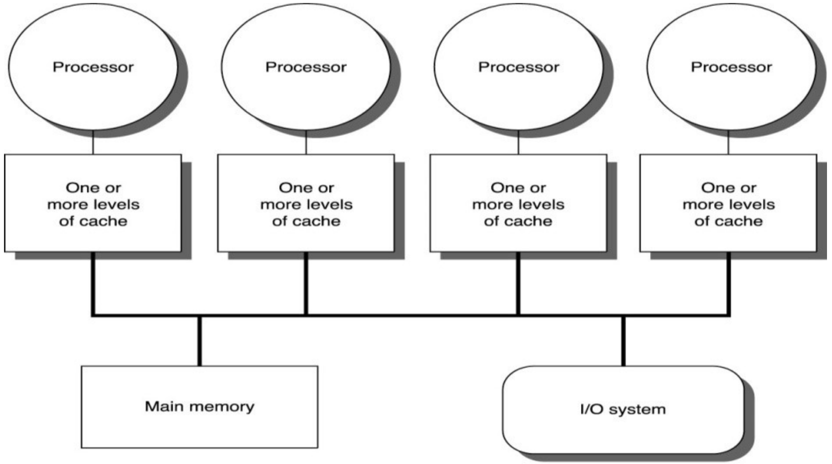


Distributed Memory

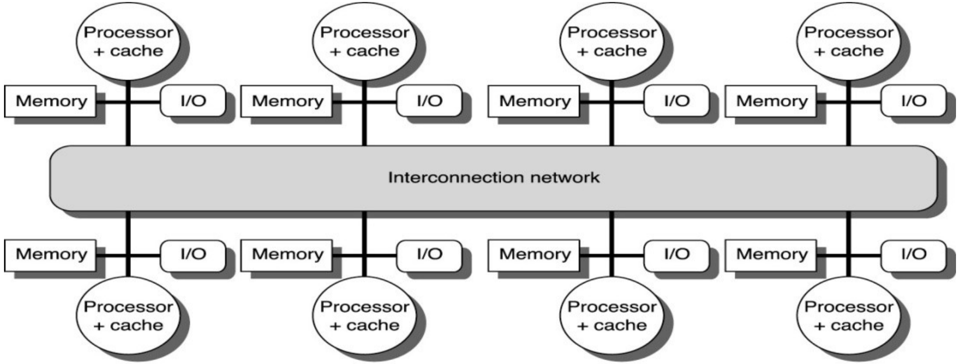


MIMD Multiprocessors

Centralized Shared Memory



Distributed Memory



Centralized-Memory Machines

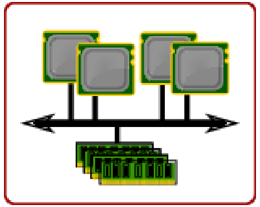
- Also “Symmetric Multiprocessors” (SMP)
- “Uniform Memory Access” (UMA)
  - All memory locations have similar latencies
  - Data sharing through memory reads/writes
  - P1 can write data to a physical address A,
    P2 can then read physical address A to get that data

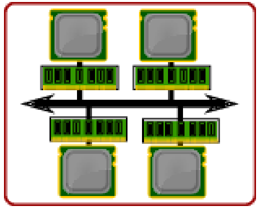
- Problem: Memory Contention
  - All processor share the one memory
  - Memory bandwidth becomes bottleneck
  - Used only for smaller machines
    - Most often 2,4, or 8 processors

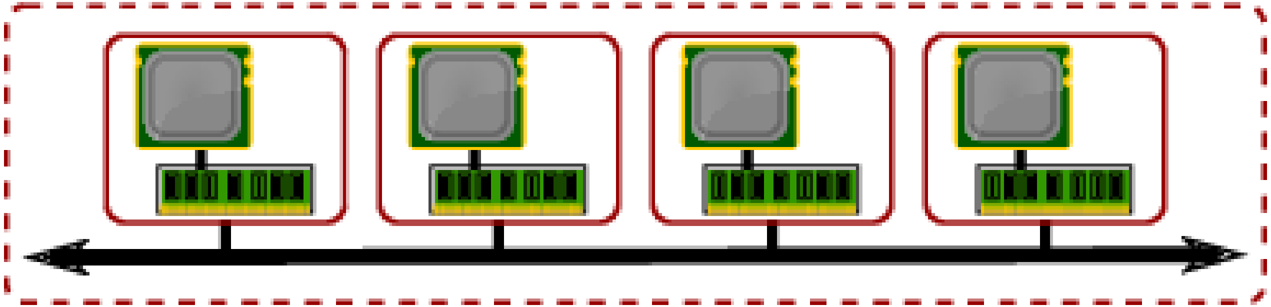
Distributed-Memory Machines

- Two kinds
  - Distributed Shared-Memory (DSM)
    - All processors can address all memory locations
    - Data sharing like in SMP
    - Also called **NUMA** (non-uniform memory access)
    - Latencies of different memory locations can differ (local access faster than remote access)
  - Message-Passing
    - A processor can directly address only local memory
    - To communicate with other processors, must explicitly send/receive messages
    - Also called multicomputers or clusters
- Most accesses local, so less memory contention (can scale to well over 1000 processors)

Message-Passing Machines





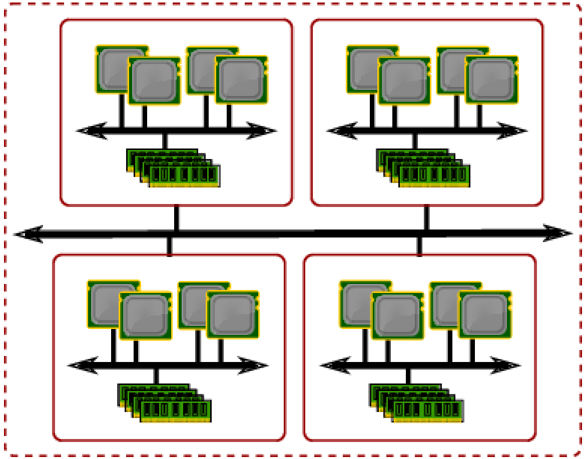


Message-Passing Machines

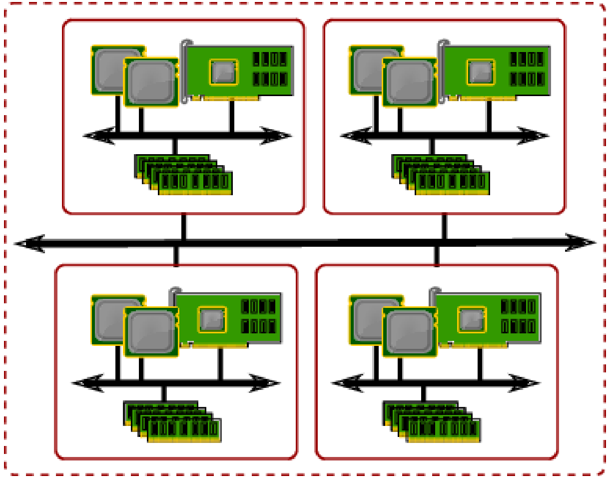
- A cluster of computers
  - Each with its own processor and memory
  - An interconnect to pass messages between them
  - Producer-Consumer Scenario:
    - P1 produces data D, uses a SEND to send it to P2
    - The network routes the message to P2
    - P2 then calls a RECEIVE to get the message
  - Two types of send primitives
    - Synchronous: P1 stops until P2 confirms receipt of message
    - Asynchronous: P1 sends its message and continues
  - Standard libraries for message passing:
    Most common is MPI – Message Passing Interface

Hybrid architectures

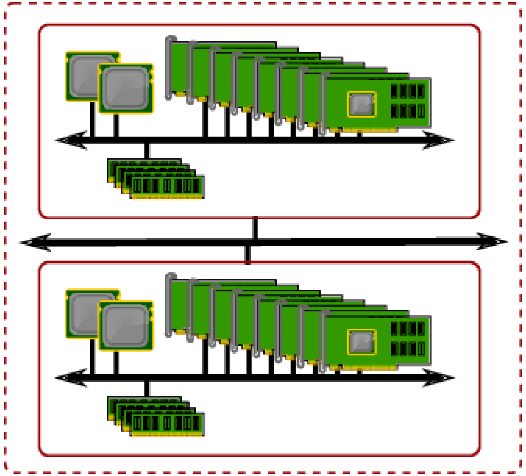
Fat cluster



GPU-accelerated



GPU cluster



Communication Performance

- Metrics for Communication Performance
  - Communication Bandwidth
  - Communication Latency
    - Sender overhead + transfer time + receiver overhead
  - Communication latency hiding
- Characterizing Applications
  - Communication to Computation Ratio
    - Work done vs. bytes sent over network
    - Example: 146 bytes per 1000 instructions

Parallel Performance

- Serial sections
  - Very difficult to parallelize the entire app
  - Amdahl's law

$$\text{Speedup}_{\text{Overall}} = \frac{1}{(1 - F_{\text{Parallel}}) + \dfrac{F_{\text{Parallel}}}{\text{Speedup}_{\text{Parallel}}}} \Rightarrow$$

$$\begin{array}{l} \text{Speedup}_{\text{Parallel}} = 1024 \\ F_{\text{Parallel}} = 0.5 \\ \hline \text{Speedup}_{\text{Overall}} = 1.998 \end{array} \qquad \begin{array}{l} \text{Speedup}_{\text{Parallel}} = 1024 \\ F_{\text{Parallel}} = 0.99 \\ \hline \text{Speedup}_{\text{Overall}} = 91.2 \end{array}$$

- Large remote access latency (100s of ns)
  - Overall IPC goes down

$$\text{CPI} = \text{CPI}_{\text{Base}} + \text{RemoteRequestRate} \times \text{RemoteRequestCost}$$

This cost reduced with multi-core

$$\text{CPI}_{\text{Base}} = 0.4 \qquad \text{RemoteRequestCost} = \frac{400\text{ns}}{0.33\text{ns/Cycle}} = 1200 \text{ Cycles} \qquad \text{RemoteRequestRate} = 0.002$$

$$\text{CPI} = 2.8$$

We need at least 7 processors just to break even!

Message Passing Pros and Cons

- Pros
  - Simpler and cheaper hardware
  - Explicit communication makes programmers aware of costly (communication) operations
- Cons
  - Explicit communication is painful to program
  - Requires manual optimization
    - If you want a variable to be local and accessible via LD/ST, you must declare it as such
    - If other processes need to read or write this variable, you must explicitly code the needed sends and receives to do this