

**ECE7995**

**(2) Objective of Caching and Prefetching:**

**Avoiding Lagging Latency and Taking Advantage of Bountiful Bandwidth**

## Memory Hierarchy vs. Caching/Prefetching

- **Memory hierarchy** consists of registers, processor caches, DRAM memory, local disks, and remote disks accessible via local network or Internet (listed in the order of their distances to processor).
- The **closer** a component is to processor, the **faster** it is to service requests from processor.
- The **furtherer** a component is to processor, the **larger** its capacity is.
- Caching/Prefetching aim to **quickly** access large set of data.
  - Caching is to temporarily buffer data in a faster storage device for efficient **reuse**.
  - Prefetching is to **speculatively** fetch data into a faster storage device **before** an expected request for them is generated to a slower storage device.

How to measure the performance of the data transferals between levels of the hierarchy?

➔ Latency and bandwidth

# Latency and Bandwidth

- Latency is the **entire** time period from an initialization of a request data to the time the data is delivered.
- Bandwidth is the total number of requests serviced in a **unit** time period.
- To understand the metrics, think of hypothesize assembly line for manufacturing automobiles:
  - It takes four hours for building a Ford Taurus, starting from parts to a drivable car → Latency
  - Every four minutes there is a Ford Taurus leaving assembly line → Bandwidth
- Which is more meaningful in defining “quickly”? Its **depends**.
  - If a large amount of data are requested in a **bursty** fashion, bandwidth is more relevant.
  - If a fraction of data is requested in a synchronous fashion, latency quantifies its **end-to-end** access time.
- In many cases, small latency is more difficult to achieve than high bandwidth.

## Imbalance between Bandwidth and Latency Improvements

- Consistent trend in data transfer – bandwidth improves much more quickly than latency.

Milestone	1	2	3	4	5	6
<b>Microprocessor</b>	16-bit address/bus, microcoded	32-bit address/bus, microcoded	5-stage pipeline, on-chip I & D caches, FPU	2-way superscalar, 64-bit bus	Out-of-Order, 3-way superscalar	Superpipelined, on-chip L2 cache
<b>Product</b>	Intel 80286	Intel 80386	Intel 80486	Intel Pentium	Intel Pentium Pro	Intel Pentium 4
<b>Year</b>	1982	1985	1989	1993	1997	2001
<b>Die size (mm<sup>2</sup>)</b>	47	43	81	90	308	217
<b>Transistors</b>	134,000	275,000	1,200,000	3,100,000	5,500,000	42,000,000
<b>Pins</b>	68	132	168	273	387	423
<b>Latency (clocks)</b>	6	5	5	5	10	22
<b>Bus width (bits)</b>	16 bits	32 bits	32 bits	64 bits	64 bits	64 bits
<b>Clock rate (MHz)</b>	12.5	16	25	66	200	1500
<b>Bandwidth (MIPS)</b>	2	6	25	132	600	4500
<b>Latency (nsec)</b>	320	313	200	76	50	15

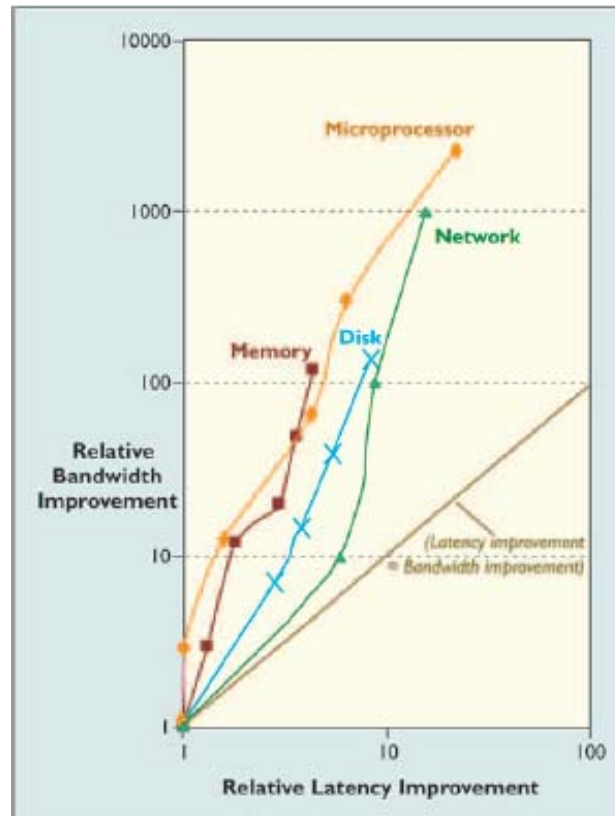
## Imbalance between Bandwidth and Latency Improvements (cont'd)

Memory Module	DRAM	Page Mode DRAM	Fast Page Mode DRAM	Fast Page Mode DRAM	Synchronous DRAM	Double Data Rate SDRAM
Module width	16 bits	16 bits	32 bits	64 bits	64 bits	64 bits
Year	1980	1983	1986	1993	1997	2000
Mbits/DRAM chip	0.06	0.25	1	16	64	256
Die size (mm <sup>2</sup> )	35	45	70	130	170	204
Pins/DRAM chip	16	16	18	20	54	66
Bandwidth (MB/s)	13	40	160	267	640	1,600
Latency (nsec)	225	170	125	75	62	52

## Imbalance between Bandwidth and Latency Improvements (cont'd)

<b>Local Area Network</b>	Ethernet	Fast Ethernet	Gigabit Ethernet	10 Gigabit Ethernet		
<b>IEEE Standard</b>	802.3	802.3u	802.3ab	802.3ae		
<b>Year</b>	1978	1995	1999	2003		
<b>Bandwidth (Mb/s)</b>	10	100	1000	10000		
<b>Latency (msec)</b>	3000	500	340	190		
<b>Hard Disk</b>	3600 RPM	5400 RPM	7200 RPM	10000 RPM	15000 RPM	
<b>Product</b>	CDC Wren1 94145-36	Seagate ST41600	Seagate ST15150	Seagate ST39102	Seagate ST373453	
<b>Year</b>	1983	1990	1994	1998	2003	
<b>Capacity</b>	0.03 Gbytes	1.4 Gbytes	4.3 Gbytes	9.1 Gbytes	73.4 Gbytes	
<b>Disk form factor</b>	5.25 inch	5.25 inch	3.5 inch	3.5 inch	3.5 inch	
<b>Media diameter</b>	5.25 inch	5.25 inch	3.5 inch	3.0 inch	2.5 inch	
<b>Interface</b>	ST-412	SCSI	SCSI	SCSI	SCSI	
<b>Bandwidth (MB/s)</b>	0.6	4	9	24	86	
<b>Latency (msec)</b>	48.3	17.1	12.7	8.8	5.7	

## Imbalance between Bandwidth and Latency Improvements (cont'd)



- Performance gains of microprocessor and network: **1000-2000X** in bandwidth and **20-40X** in latency.
- For memory and disks, bandwidth advances are also **much greater** than their gains in latency.

## Imbalance between Bandwidth and Latency Improvements (cont'd)

	Processor	Memory Module	LAN	Disk
Annual Latency Improvement (all milestones)	1.17	1.07	1.12	1.09
Annual Capacity Improvement (all milestones)	--	1.52	--	1.48
Annual Bandwidth Improvement (all milestones)	1.50	1.27	1.39	1.28
Time for Bandwidth to Double in Years	1.7	2.9	2.1	2.8
Capacity Improvement in Time for Bandwidth to Double	--	3.4	--	3.0
Latency Improvement in Time for Bandwidth to Double	1.3	1.2	1.3	1.3

Average improvement for all milestones (last two decades)

	Processor	Memory Module	LAN	Disk
Annual Latency Improvement (last 3 milestones)	1.22	1.06	1.13	1.09
Annual Capacity Improvement (last 3 milestones)	--	1.49	--	1.37
Annual Bandwidth Improvement (last 3 milestones)	1.55	1.30	1.78	1.29
Time for Bandwidth to Double in Years	1.6	2.7	1.2	2.7
Capacity Improvement in Time for Bandwidth to Double	--	2.9	--	2.4
Latency Improvement in Time for Bandwidth to Double	1.4	1.2	1.2	1.3

Average improvement for the three latest milestones (more recent trends)

- In the time that bandwidth doubles, latency improves by no more than a factor of 1.2 to 1.4.
- Bandwidth improves by at least the square of the improvement in latency.

## Reasons for Bountiful Bandwidth but Lagging Latency

- Moore's law helps bandwidth more than latency.
  - Moore's law: a periodic doubling in the number of transistors per chip, due to scaling and larger chips. (doubles 22-24 months, recently).
  - More transistors helps bandwidth (think of halving each stage in an assembly line)
  - But more transistors and longer distances on larger chips limits gains in latency.
- Distance limits latency.
  - Distance sets a lower bound to latency.
  - Ultimately latency is bounded by light speed (300m → latency > 1us).
- Bandwidth is generally easier to sell.
  - While marketing of performance, it's easier to sell higher bandwidth than to sell lower latency. (e.g. network: 10Gbps bandwidth vs 10 us latency).
  - This non-technical reason leads to more investment in improvement of bandwidth.
  - This is in contrast with automotive industry, which advertises time to accelerate from 0-60 mph rather than top speed.

## Reasons for Bountiful Bandwidth but Lagging Latency (cont'd)

- Latency helps bandwidth.
  - For example, spinning disks faster reduces the rotational latency, which also helps to improve read/write rate.
- Bandwidth hurts latency.
  - Bandwidth is usually improved at the expense of latency.
  - For example, increasing disk density (bytes per inch in disk track and number of tracks) helps bandwidth but hurts latency of random access because of more time needed for disk head positioning.
- Operating system overhead hurts latency.
  - OS in an I/O stack generates overhead, which can be amortized by large data transfer but plays a large part of the latency for small data requests.

# Coping with Bountiful Bandwidth but Lagging Latency

- Latency is important in making system responsive.
- Three basic proven techniques to cope with the imbalance.
- Caching: hide long delay to slower devices by keeping some previously requested data in faster devices.
  - It is almost ubiquitous in the storage hierarchy – processor cache, memory buffer cache, and cache built inside a hard drive ...
  - Caching is an extremely successful practice
  - But how do you know which data would be reused? → temporal locality
- Prefetching: take advantage of high bandwidth by piggybacking future accesses with current ones.
  - Performance of large transfer is determined by bandwidth, while small transferal costs latency.
  - But how do we know which data would be requested in the future? → spatial locality

**How does prefetching affect latency? It depends.**

**→ should do aggregate accesses in the non-critical path**

## Coping with Bountiful Bandwidth but Lagging Latency (cont'd)

- Replication: store multiple copies of data at multiple devices/sites, and access data from the place of the smallest latency.
  - For example, ISPs often use multiple sites to reduce latency (CDN: Content Delivery Network).
  - But how to keep replicas coherent in a small overhead?
  - The three techniques would increase in popularity with the increasingly unbalanced improvements between latency and bandwidth.
- These three techniques can be used at the same time.
  - But would they conflict with each other? If yes, how to make them work cooperatively?
- This course will focus on caching and prefetching and address the aforementioned “*how*”s.

Acknowledgements: most contents covered in the lecture are adapted from “*Latency Lags Bandwidth*” by D.A.Patterson