Latency, Throughput
And the memory Hierarchy

Some numbers

- Time: $1\,\text{sec} = 10^3 ms = 10^6 \mu s = 10^9 ns$
- Data Size:
  - $\text{KB} = 10^3\ bytes$
  - $\text{MB} = 10^6\ bytes$
  - $\text{GB} = 10^9\ bytes$
  - $\text{TB} = 10^{12}\ bytes$
  - $\text{PB} = 10^{15}\ bytes$
- Compter Clock Rate (since 2003) ~3GHz = 3×10^9 cycles per second.

Example of Latency vs. throughput

- **Latency:** time between joining the line and leaving the cashier.
- **Throughput:** Number of people coming out of the store per minute.
- **Minimal latency**: Time at Cashier.
- Customer cares about latency.
- Store cares mostly about throughput (dollars per minute) and to some degree latency (customer happier).

Definitions

- **Latency:** total time to process one unit from start to end.
- **Throughput:** number of units processed in one minute.
- Is $Throughput = \frac{1}{Latency}$?
- **Not necessarily!**

Wholesale vs. retail

Wholesale

Costco

Buying water in costco and driving the truck to it's stand:
2 hours
10 cents / bottle

(water cache)

Retail

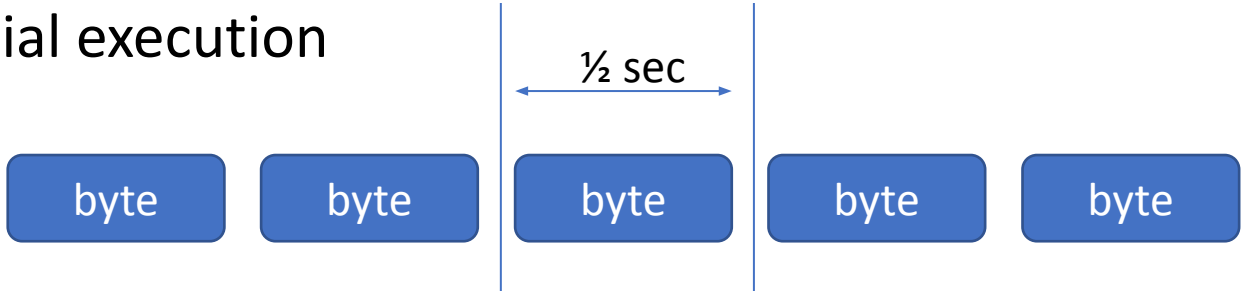
Walking to the truck and buying a water bottle: ½ minute
1$ / bottle

Different latencies for different units

- If unit is a package of 32 water bottles:
  - **Latency** from entering Costco to having the water bottles ready for sale: 2 hours.
  - **Throughput:** number of packages sold in one Costco per hour. Maybe 20
  - $Throughput \gg \frac{1}{Latency}$
- If unit is one water bottle:
  - **Latency** from requesting a bottle to having the bottle in your hand: ½ minute
  - **Throughput:** Number of water bottles bought in an hour: Maybe 10
  - $Throughput \ll \frac{1}{Latency}$

Why $Throughput \neq \frac{1}{Latency}$?

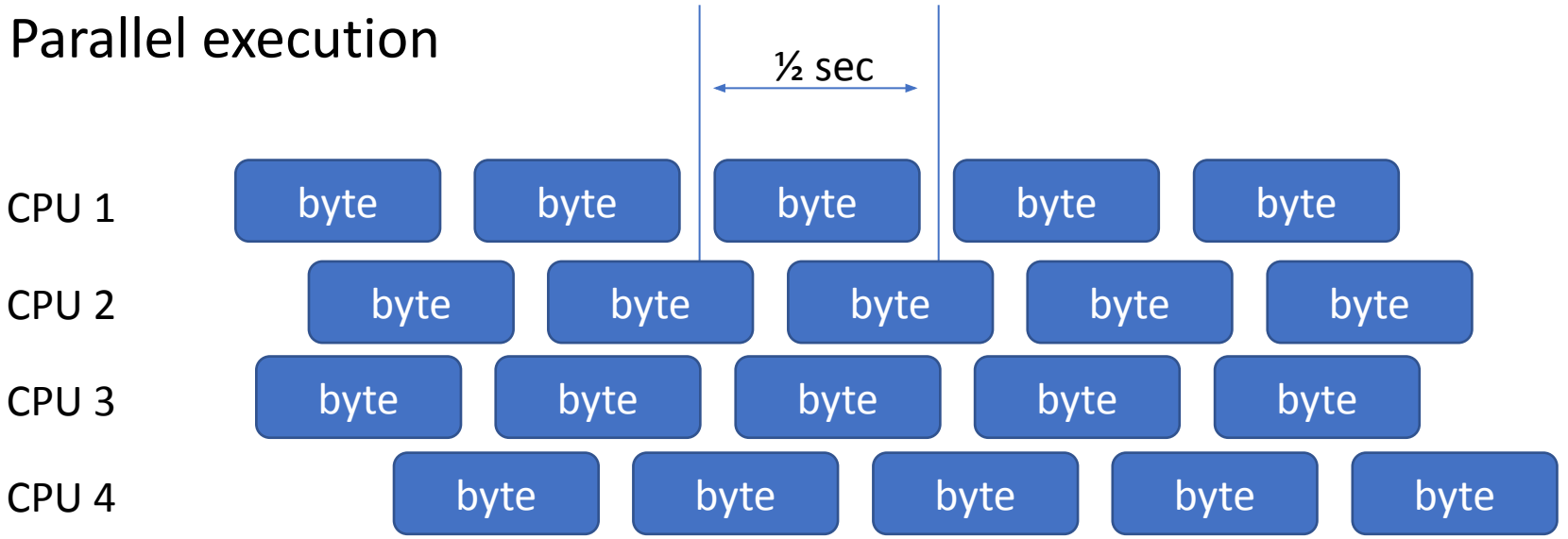
Sequential execution



Latency = ½ sec

Throughput = 2 byte/sec

Parallel execution

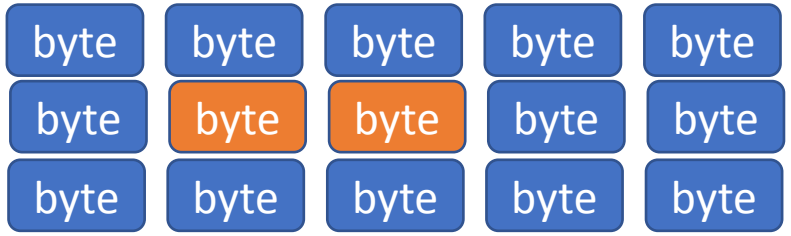


Latency = ½ sec

Throughput = 8 byte/sec

Reading data blocks from disk

- The latency for reading a random block from a spinning Disk is 10ms.
- Why? Because of the mechanics of moving the reading head.
- Instead of reading one byte, read a whole block!
- If all of the data in the block is useful:
  - Latency: 10ms (100Byte/sec for random access)
  - Throughput: 100MB/sec (for sequential access)



- Caching: a method for achieving low latency by predicting access.

When do we need low latency?

- When interacting with our cell phone or laptop.
- When playing games online: https://tetris.com/play-tetris
- When checking in to a flight.
- Anything interactive – working on a jupyter notebook.
- Running javascript inside a browser
- Latency = reaction time < 100ms

When do we need high throughput

- When we have a large amount of data to process (50TB),
  - **we care** about throughput – number of seconds to process 1TB
  - **we don't care** about latency = processing one record,
- Example: transfer 50TB of data to the cloud.
  - Regular home line: upload speed 6Mbps = 6/8 MBps -> 771 days ~ 2 years
  - Fast University line: upload speed 100Mbps = 100/8 MBps -> 46 days
  - Dedicated fiber: upload speed 10Gbps = 10/8 GBps -> 11 hours
  - Dedicated fiber costs 10-100K$ per year.

The fastest and cheapest option: Fedex

- Latency: 24 hour
- Throughput using 50TB snowball:
  50TB / 24 hours.
- Cost ~ 200$

Snowball is a petabyte-scale data transport solution that uses devices designed to be secure to transfer large amounts of data into and out of the AWS Cloud. Using Snowball addresses common challenges with large-scale data transfers including high network costs, long transfer times, and security concerns. Customers today use Snowball to migrate analytics data, genomics data, video libraries, image repositories, backups, and to archive

Summary for part 1

- When providing an interactive experience, we care more about latency than throughput.
- When processing TB, we care more about throughput than about latency.
- Transmitting TB through the internet is slow and expensive.
- Sending a physical disk through Fedex is cheap and high bandwidth.