[544] Intro to Big Data Systems

Tyler Caraza-Harter

Outline

Course Overview

- Introduction
- Resources

Resources

- Overview
- Compute
- Memory
- Storage
- Network

Deployment

Introductions

Tyler Caraza-Harter (Section 1 instructor)

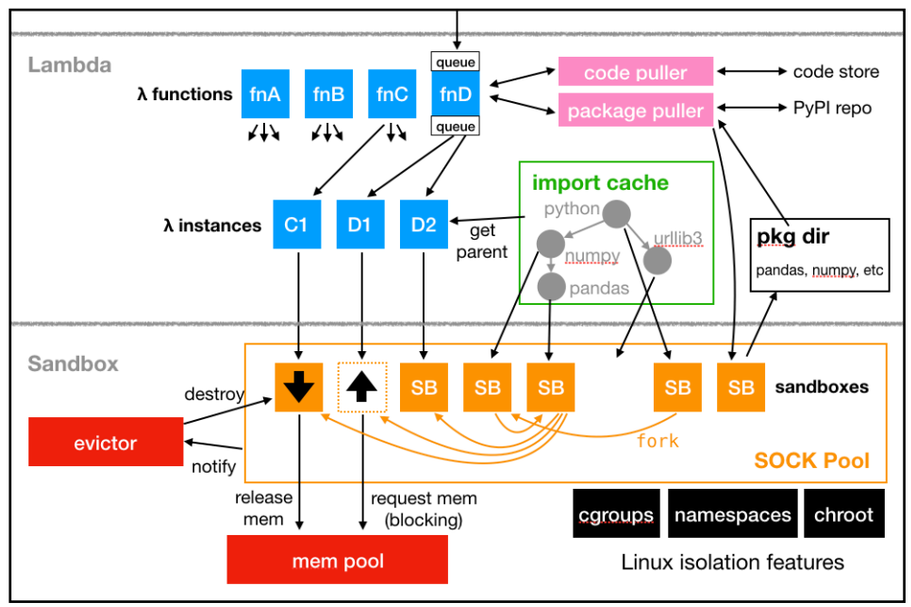
- Long time Badger
- Email: tharter@wisc.edu
- Just call me "Tyler" (he/him)

Industry experience

- Worked at Microsoft on SQL Server and Cloud
- Other internships/collaborations:
  Qualcomm, Google, Facebook, Tintri, Bauplan

Open source

- OpenLambda (serverless cloud platform)
- https://github.com/open-lambda/open-lambda



Who are You?

Year in school? Major?

What CS courses have people taken before?

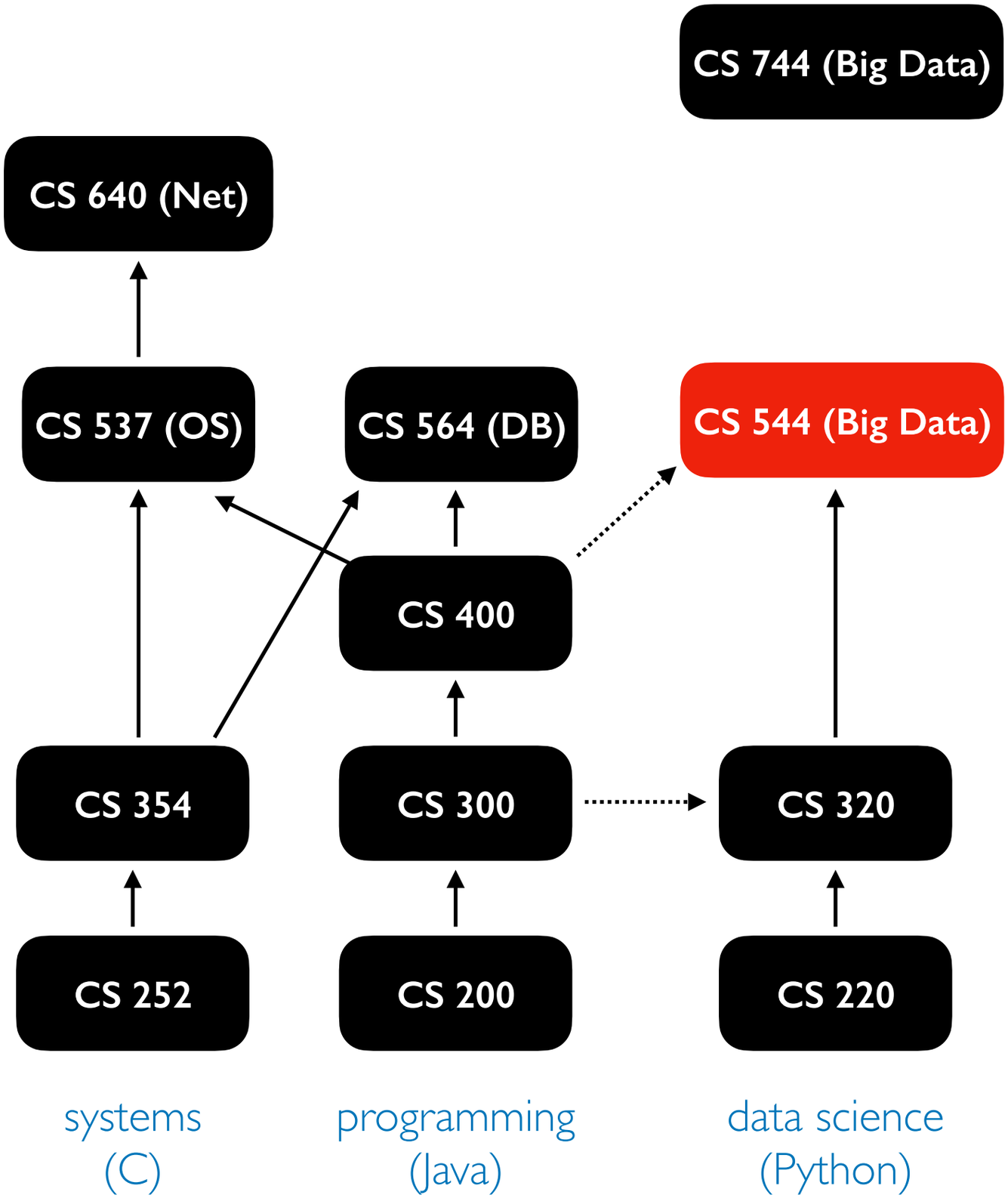
- 320? 400? 537/564/640?

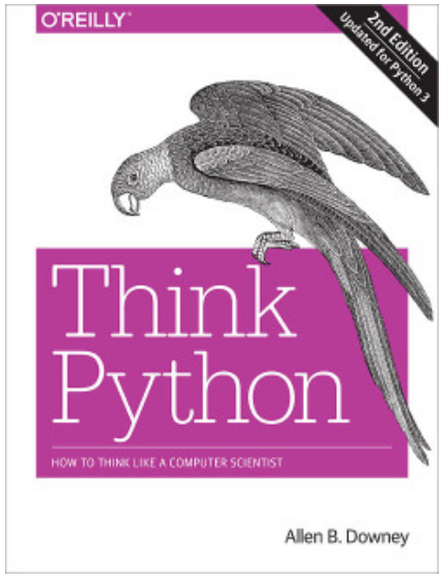
Please fill the "Who are you?" form (**due today**):
https://tyler.caraza-harter.com/cs544/s26/forms.html

Why?

- Help me get to know you
- Get extra credit

Related courses





- most coding will be in Python (400 folks will need to pick this up)
- first third of course will cover some foundations from operating systems, networking, and databases
- 744 will cover some similar systems, but from the research perspective (544 is hands on)

What are "*systems*"?

Some major categories of software

- analysis code (run once, get results)
- applications (often long running, maybe a website)
- systems (manage resources, like storage space)

Other code uses systems. For example, without an operating system, your analyis code couldn't read files.

Whatever kind of programming you are doing,
knowing how systems work with help you **use resources better**!

What are "*big data systems*"?

Some major categories of software

- analysis code (run once, get results)
- applications (often long running, maybe a website)
- systems (manage resources, like storage space)

Other code uses systems. For example, without an operating system, your analyis code couldn't read files.

As data grows, we need to optimize our code and/or use more resources

Big data systems manage resources that are:

- distributed (cluster of machines)
- specialized (e.g., GPUs)

What will you learn in 544?

Learning objectives

- Deploy distributed systems for data storage and analytics
- Demonstrate competencies with tools and processes necessary for loading data into distributed storage systems
- Write programs that use distributed platforms to efficiently analyze large datasets
- Produce meaning from large datasets by training machine learning models in parallel or on distributed systems
- Measure resource usage and overall cost of running distributed programs
- Optimize distributed analytics programs to reduce resource consumption and program runtime
- Demonstrate competencies with cloud services designed to store or analyze large datasets

Grades

Thresholds

- A >= 94
- AB >= 90
- B >= 82
- BC >= 72
- C >= 65
- D >= 60
- F < 60

Tyler's Fall 2025 Section (students who reported major and didn't drop)

- All students: 38% earned an A
- Non-CS students: 33% earned an A

Components

- Final Exam (20 points)
- 3 Midterms (12 points each, 36 points total)
- Online Policy Quiz (2 points)
- 8 Programming Projects (3 points each, 24 points total)
- 6 Hand-in Worksheets (2 points each, 12 points total)
- 2 Skill Demos (3 points each, 6 points total)
- 4 Extra Credit Points (for TopHat, Instructor Endorsements on Piazza, etc)

Schedule: Projects and Midterms

Before Spring Break

- 5 project releases
- 3 midterms (all cumulative)

After Spring Break

- 3 project releases
- 1 final (cumulative)

CS 544 is a lot of work. It will be the most advanced computing work done by many data science majors. Consider, how many credits are you taking this semester?

```
Week 01: --------
Week 02: --P-----
Week 03: ----M---
Week 04: --P-----
Week 05: --------
Week 06: --P-----
Week 07: M-------
Week 08: --P-----
Week 09: --------
Week 10: M-P-----
Week 11:  break
Week 12: --P-----
Week 13: --P-----
Week 14: ----P---
Week 15: --------
```

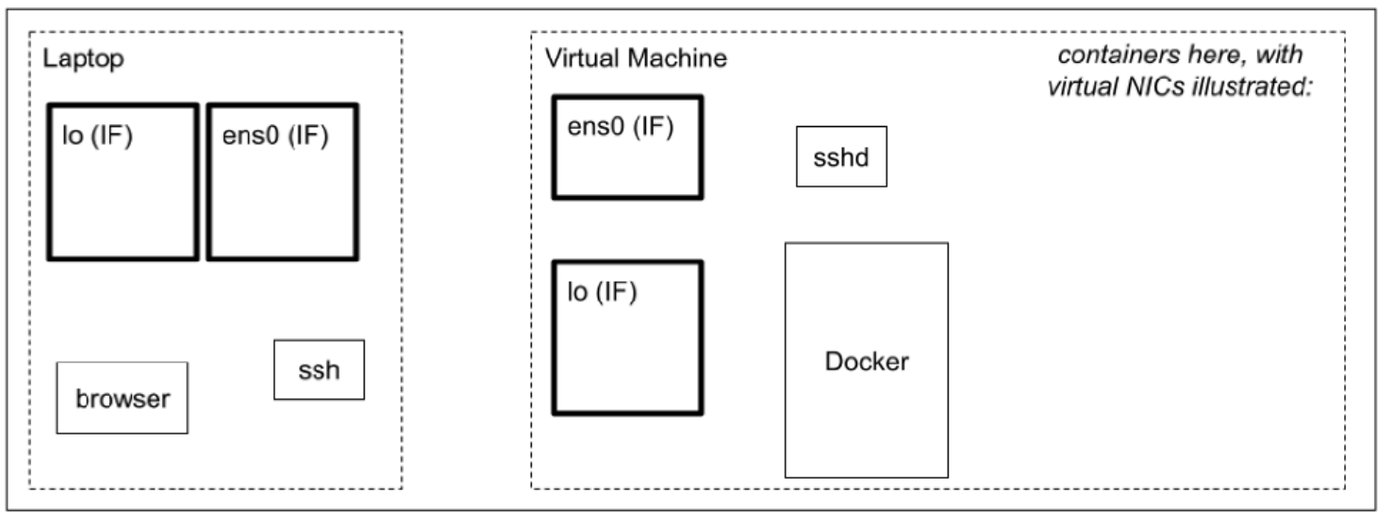
Exam Philosophy

- We have 4 exams and heavy weight relative to projects because mastery of concepts is becoming increasingly important, relative to code writing.
- Everything is cumulative because it is important to study for long-term retention. If you study, ace an exam, and forget, what was the point?
- Multiple choice is not realistic, but you'll often whiteboard problems with fellow engineers, so you'll often be asked to draw things. Example (Fall 2025):

Question 12 (5 points). Consider the following shell commands, and illustrate below how traffic flows from your laptop browser to the virtual machine to the docker container(s).

```
docker run -d -p 127.0.0.1:7000:8000 demo # Virtual machine
docker run -d -p 127.0.0.1:3000:4000 demo # Virtual machine
ssh <user>@<VM_name> -L localhost:6000:localhost:7000 # Laptop
# open http://localhost:6000 in browser on Laptop
```

Be sure to label port numbers on interfaces (IF), when known. Arrow directions should indicate whether a process is listening on a port (port => process) or sending to a port (process => port).

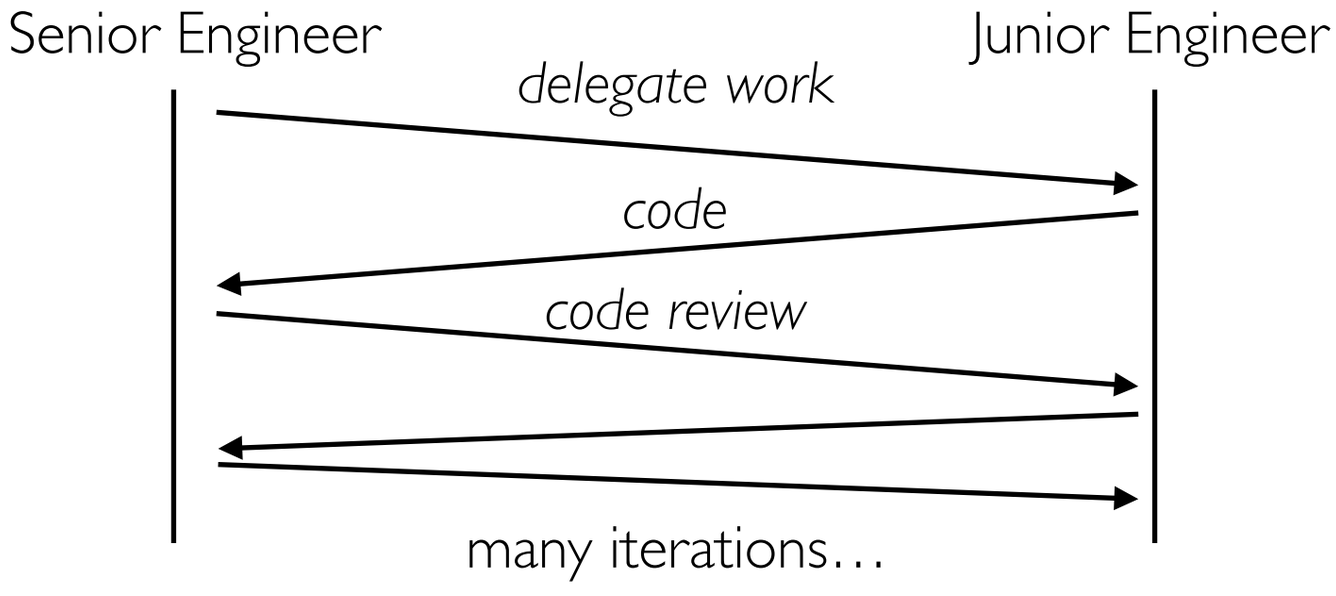


Project Philosophy: Embrace AI

- For better or worse, AI is rapidly changing the jobs we'll have.
- CS 544 will be rapidly changing too, based on the future jobs I imagine (which could be completely wrong).
- Course started in Spring 2023, but it will feel like a new course for the foreseeable future. I'll be constantly experimenting with how to teach, given AI advances.
- You'll be expected to use specific AI-based tools on specific projects, and the project complexity will reflect that.
- We will be embracing AI-assisted coding this semester. We will NOT be vibe coding. You are expected to understand and "own" the code you produce, regardless of tooling used to produce it.

Why embrace AI? To stay relevant.

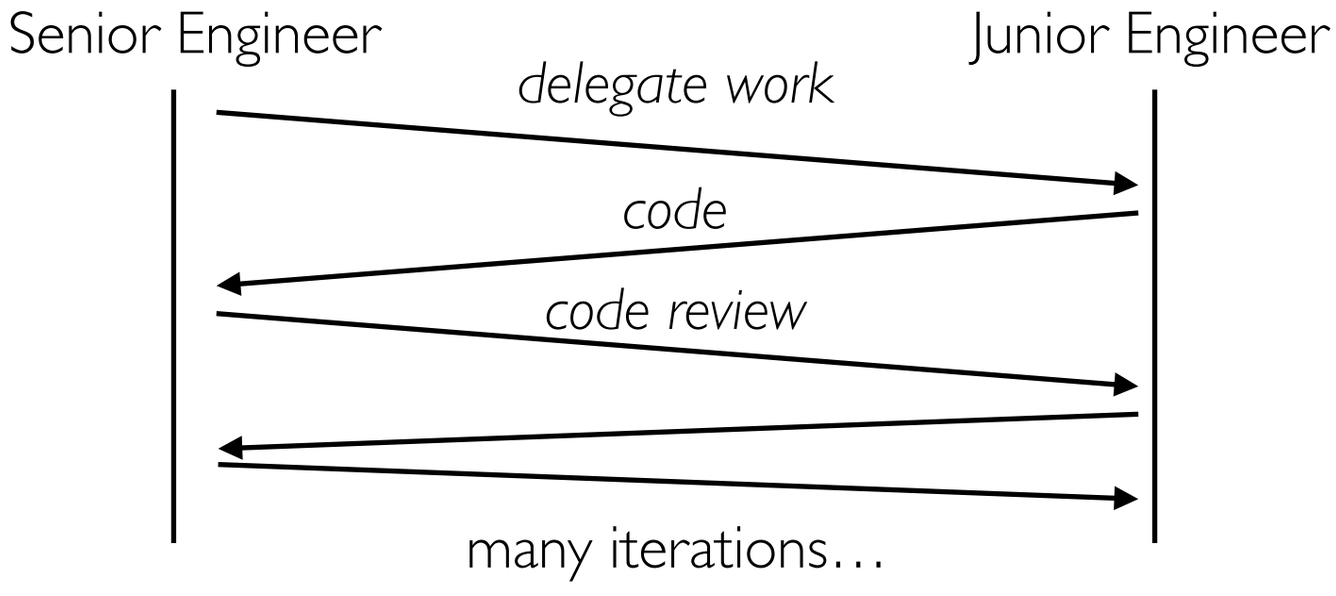
Scenario 1, Pre AI:



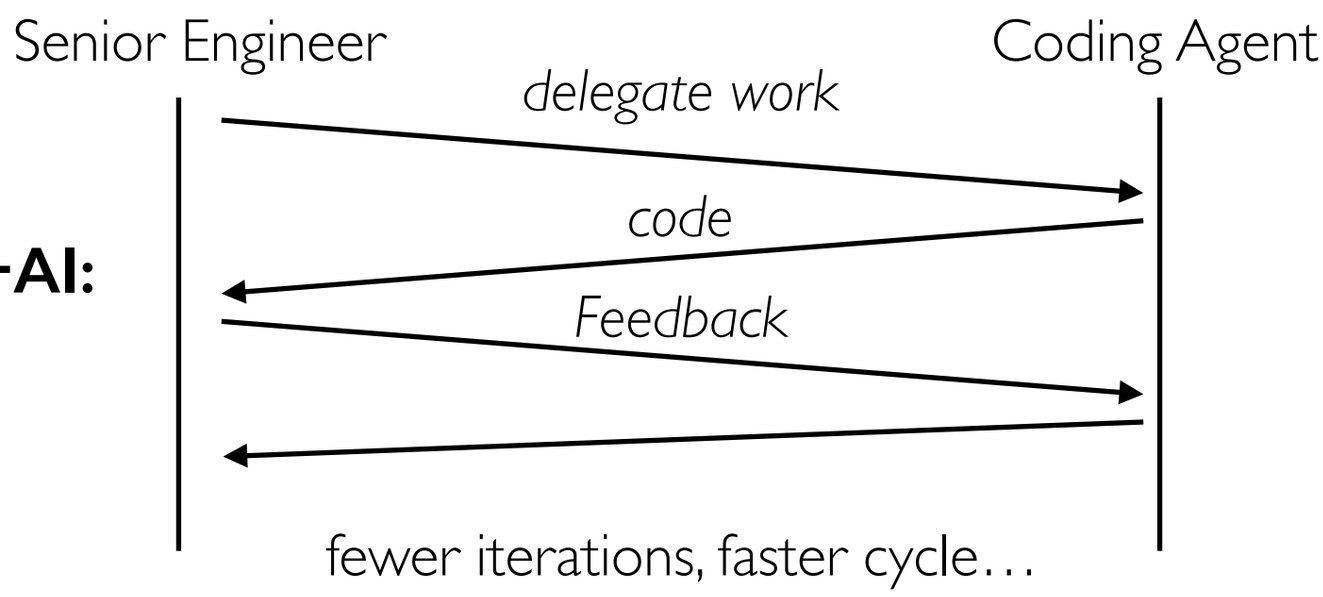
Senior engineers have always grumbled about this. It's more fun to write code than read somebody else's code. Code review is all responsibility, and no fun.

Why embrace AI? To stay relevant.

Scenario 1, Pre AI:



Scenario 2, Senior+AI:

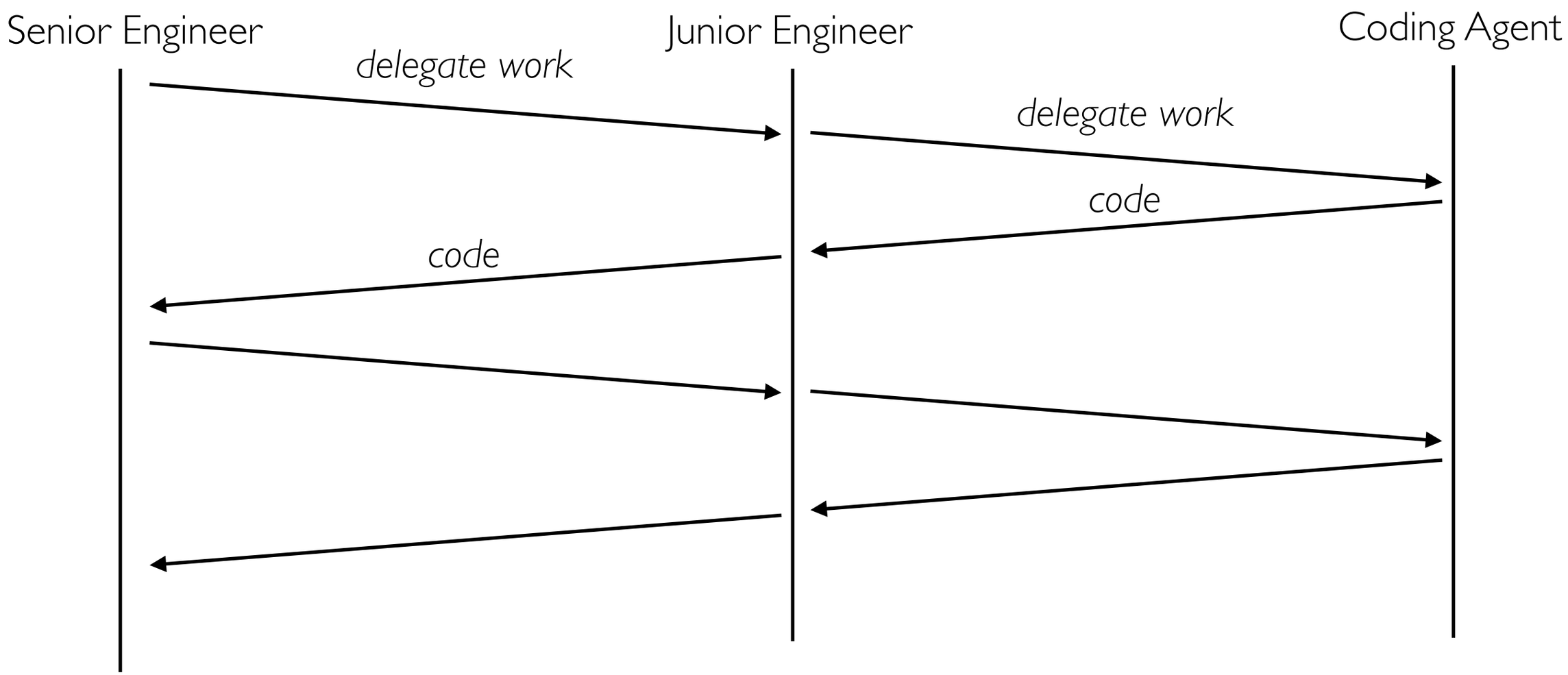


this is an existential threat to junior engineers

Junior with AI. Vibe Coding?

Vibe coding: tell AI what to do, and trust the results. Don't understand or question the code.

Scenario 3, Junior Vibe Coder:

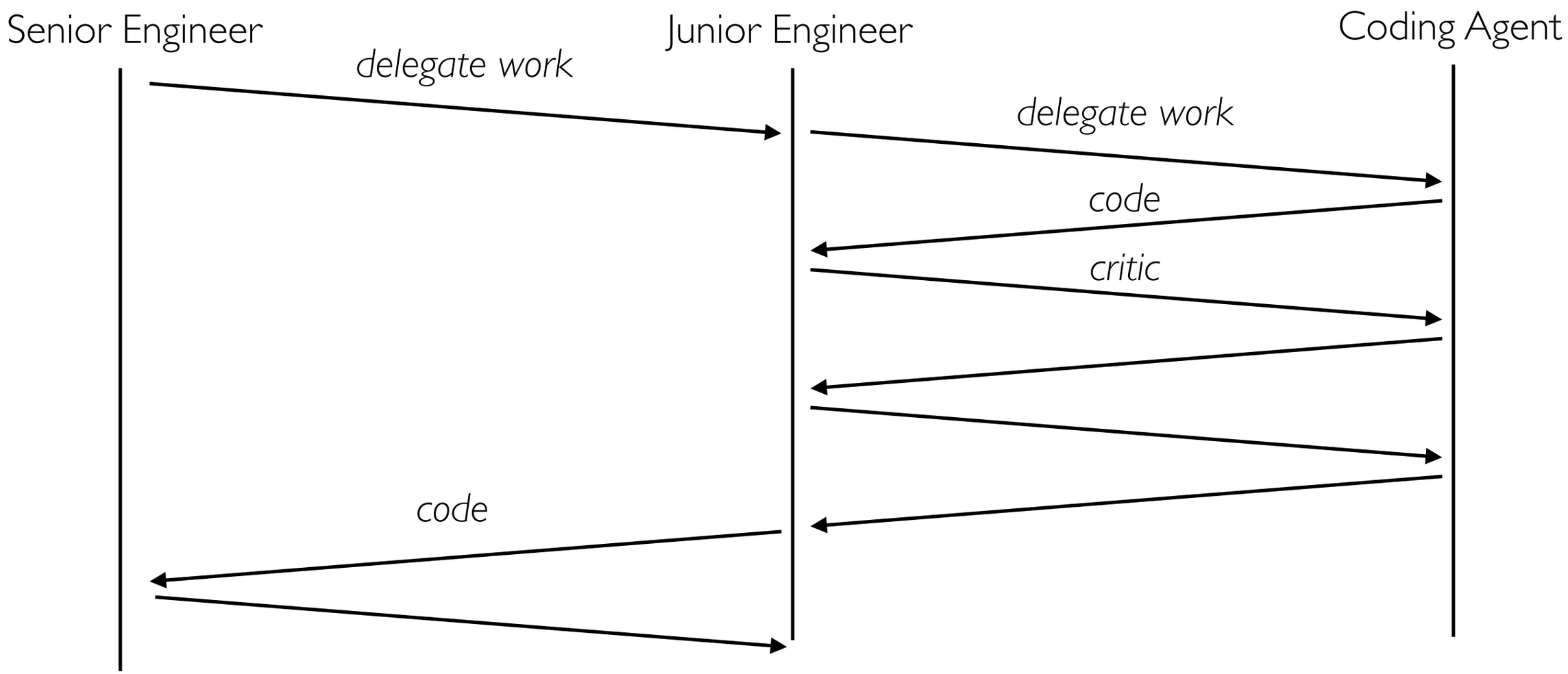


For the senior engineer, this is MUCH worse than scenario 1 or 2.
Junior feels like they're fast, but they're just slowing things down, getting between senior and agent.
They also learn nothing, so will never become senior.

Junior as Delegator and Critic

Vibe coding: tell AI what to do, and trust the results. Don't understand or question the code.

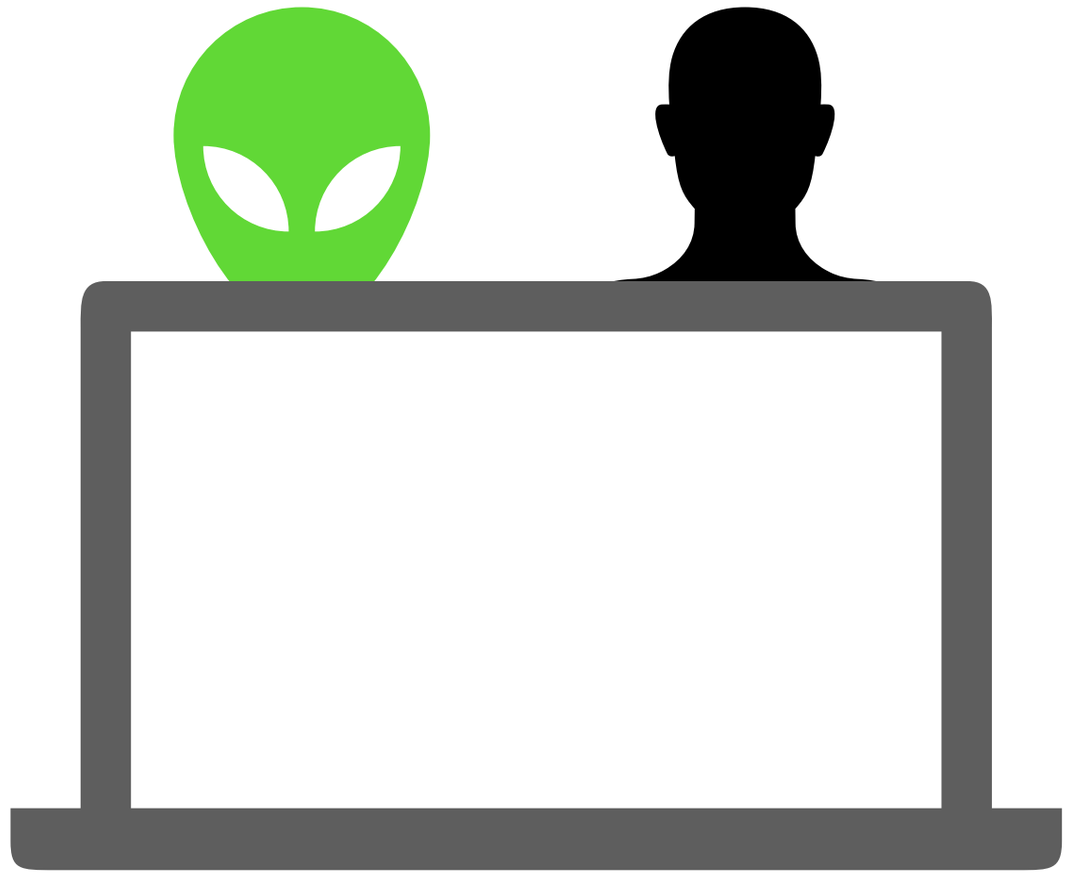
Scenario 4, Junior as Delegator and Critic:



Even if Coding Agent is "better" than junior, it is easier to critique than write code.
When AI justifies decisions, junior learns. Most importantly, senior engineer sees value in having junior on the team (directly working with AI would be more feedback cycles).

Junior understands the code and takes responsibility for it.

Pair programming with an alien?



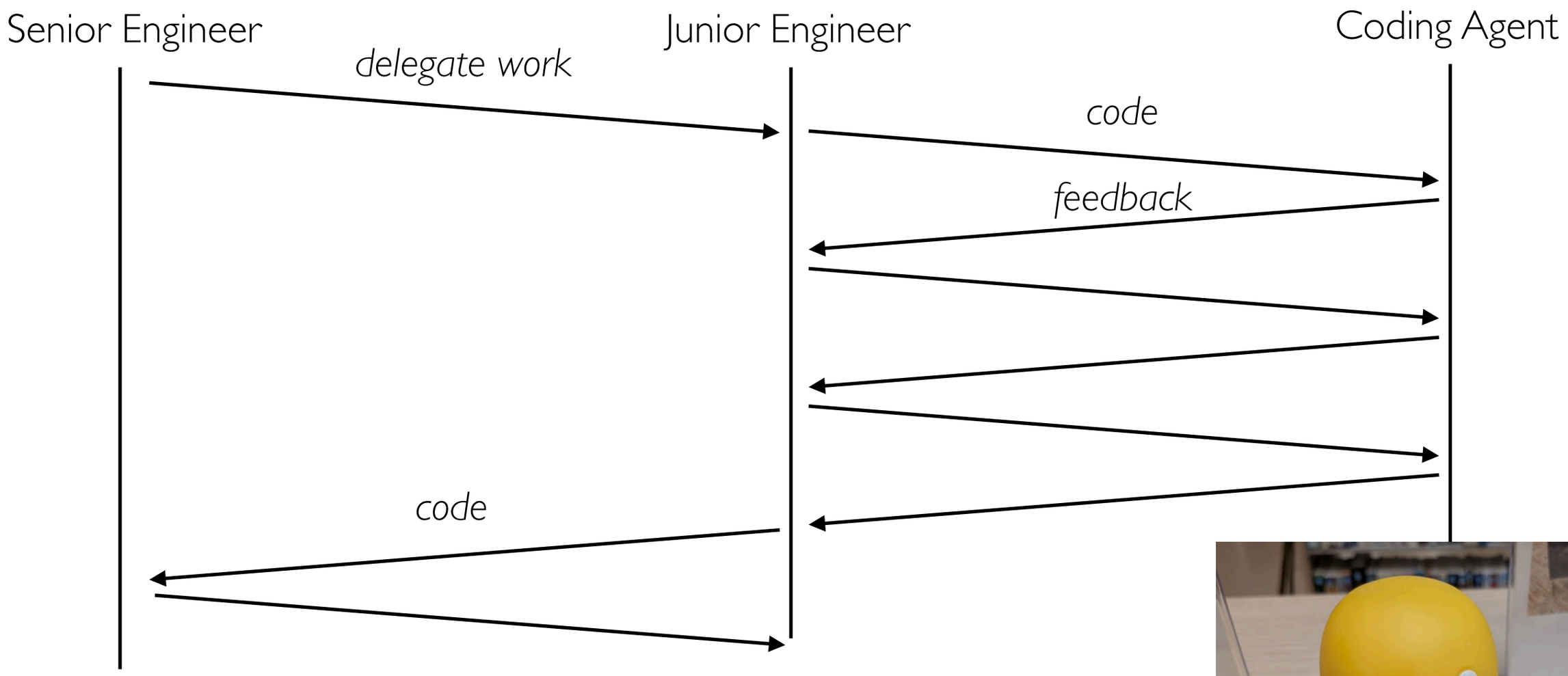
AI agents are like a different kind of intelligence.

AI is smarter than you in many ways: during training, it has consumed more material than you have in your life. It can write code faster and read documentation faster than you ever will.

You also have advantages: better at remembering the point of what is being done, better sense of what the customer needs, better aesthetic (appreciation for simplicity).

AI as Second Code Reviewer

Scenario 5, AI as Second Code Reviewer:



Decreases review load on senior engineer. Junior still exercises their programming muscles and is able to learn from fast feedback. Even if agent feedback is wrong, defending your decisions to the AI will make you a more rigorous thinker. AI is the ultimate rubber duck.

Scenario Recap

- Scenario 1: Senior + Junior
- Scenario 2: Senior + AI (already often beating scenario 1)
- Scenario 3: Vibe Coding
- Scenario 4: Delegate and Critique
- Scenario 5: AI as Code Reviewer

focus of CS 544

New skill: delegation is difficult, and pre AI, junior engineers never delegated anything to anyone!

Outline

Course Overview

- Introduction
- Resources

Resources

- Overview
- Compute
- Memory
- Storage
- Network

Deployment

Main Websites

1. https://tyler.caraza-harter.com/cs544/s26/schedule.html
   - schedule, course content, how to get help
   - links to all other resources/tools
   - some lecture recordings (review only)

2. https://git.doit.wisc.edu/cdis/cs/courses/cs544/s26/main
   - project specifications
   - lecture demo code

3. Canvas
   - announcements
   - quizzes
   - grade summaries
   - zoom office hours

1. TopHat (me asking you questions during lecture)
   - Optional, but earn extra credit for correct answers!

2. Piazza (asking questions of general interest)
   - our goal: responses <24 hours
   - don't post >5 lines of project code

Note: Piazza and email are not for debugging help because teaching debugging is most effective in-person (during office hours)

3. Email (asking questions of individual interest)
   - everybody will be assigned a TA contact
   - our goal: responses <2 business days
   - feel free to escalate by CC'ing instructor on same thread after that
   - if contacting multiple staff members about same issue, please keep all on the same thread (don't start multiple threads)

4. GitLab
   - you'll be given a **private** repo for each project
   - we'll provide feedback on GitLab

5. Anki Flash Cards
   - memory terms, basic ideas using flash cards and spaced repetition

Outline

Course Overview

Resources

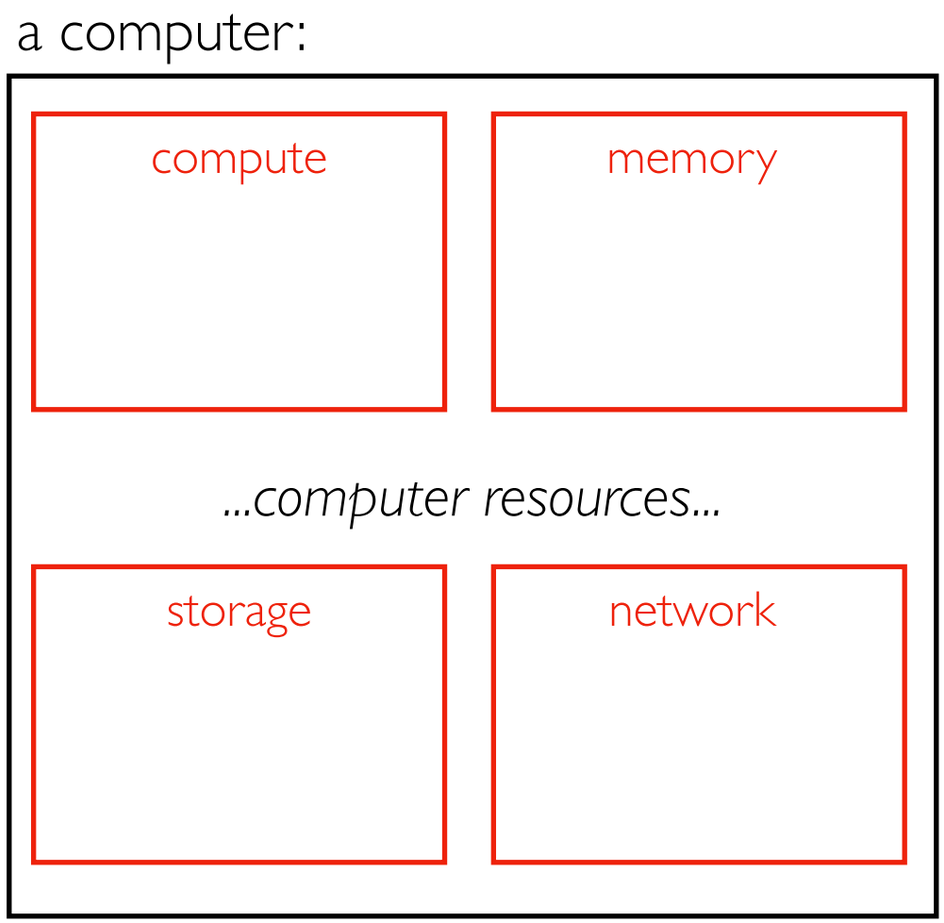
- Overview
- Compute
- Memory
- Storage
- Network

Deployment

Categories of resources

Systems: software for managing computer resources

Other kinds of software (analysis code, applications) rely on systems.

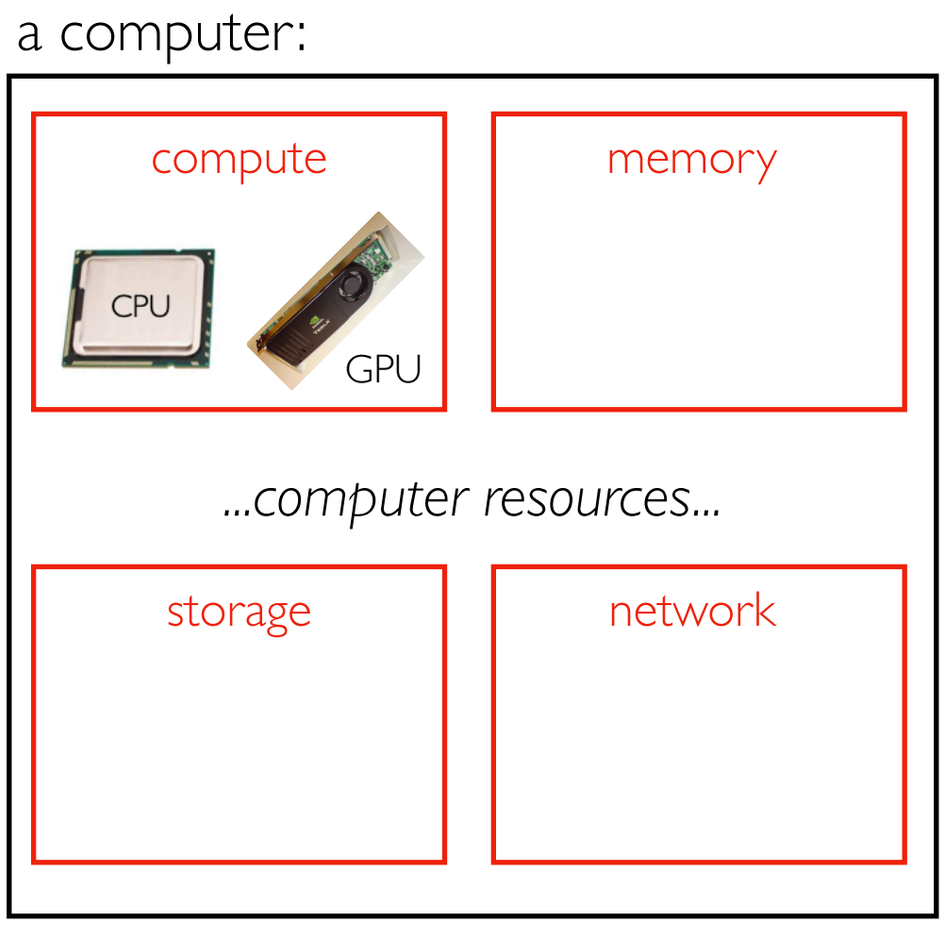


Categories of resources

Systems: software for managing computer resources

Other kinds of software (analysis code, applications) rely on systems.

computational resources execute code



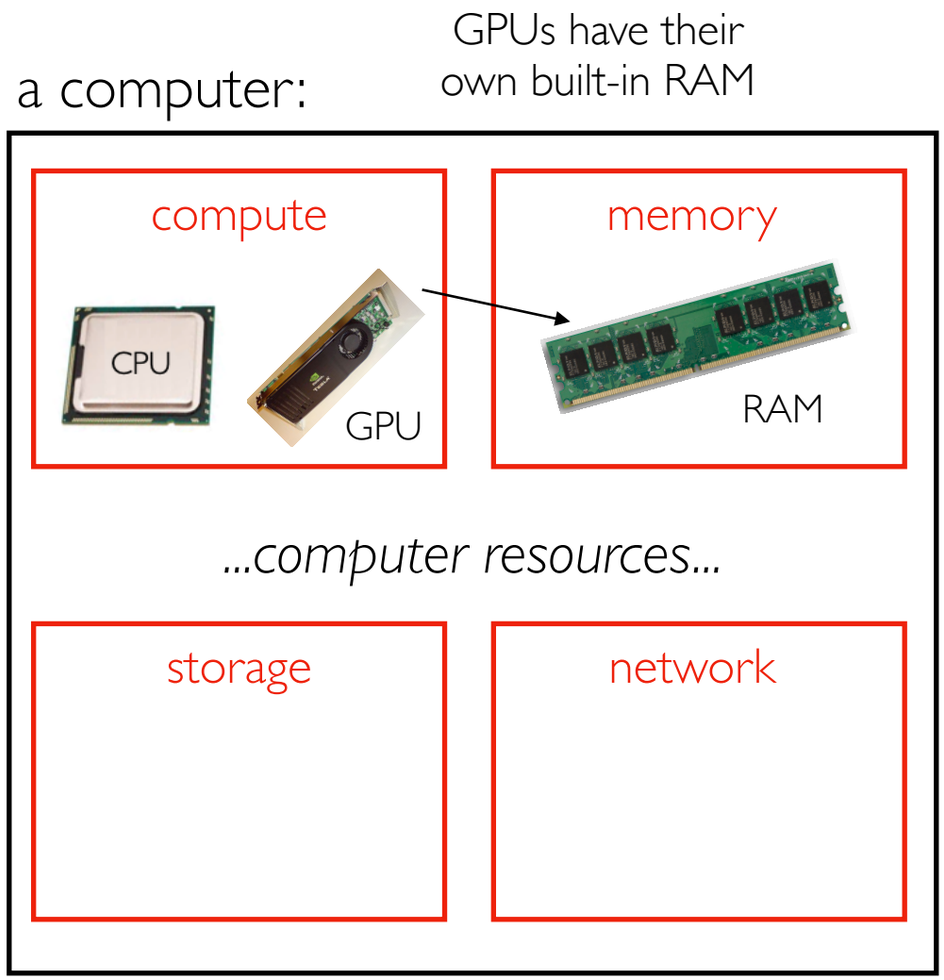
central processing unit (CPU), graphics processing unit (GPU)

Categories of resources

Systems: software for managing computer resources

Other kinds of software (analysis code, applications) rely on systems.

computational resources execute code



memory holds data for active usage

random-access memory (RAM)

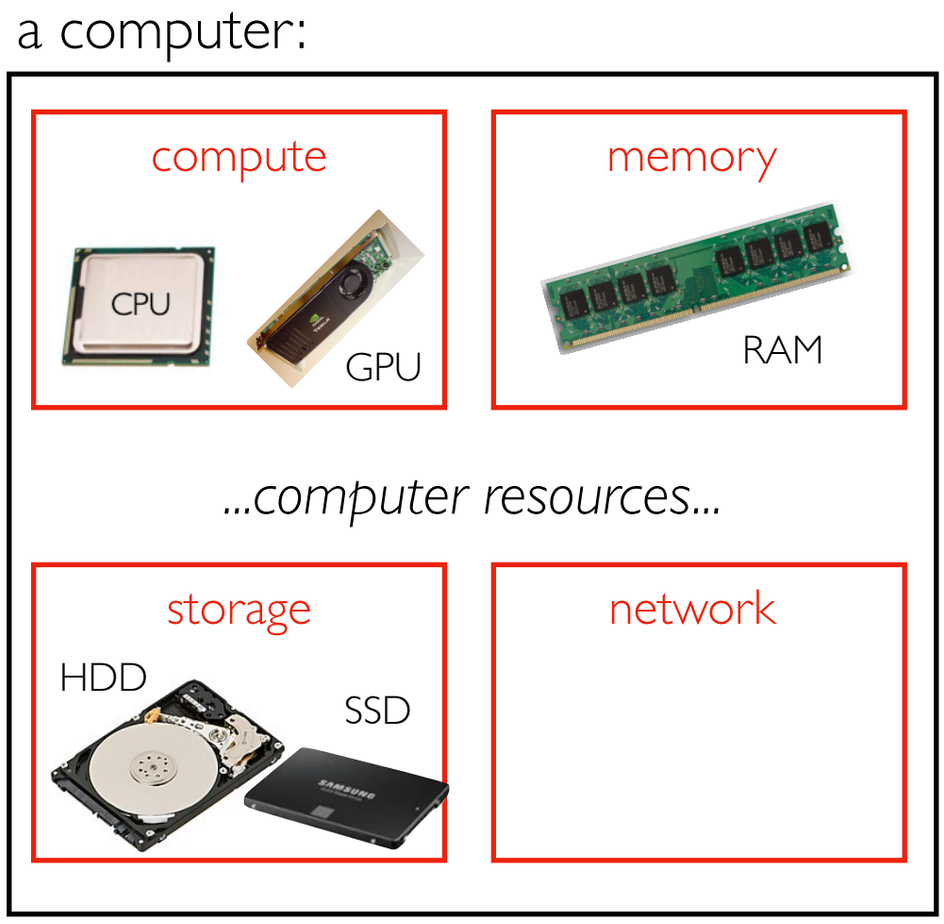
Categories of resources

Systems: software for managing computer resources

Other kinds of software (analysis code, applications) rely on systems.

computational resources execute code

storage holds long-term data



memory holds data for active usage

hard disk drive (HDD), solid-state disk (SSD)

https://en.wikipedia.org/wiki/Hard_disk_drive

Categories of resources

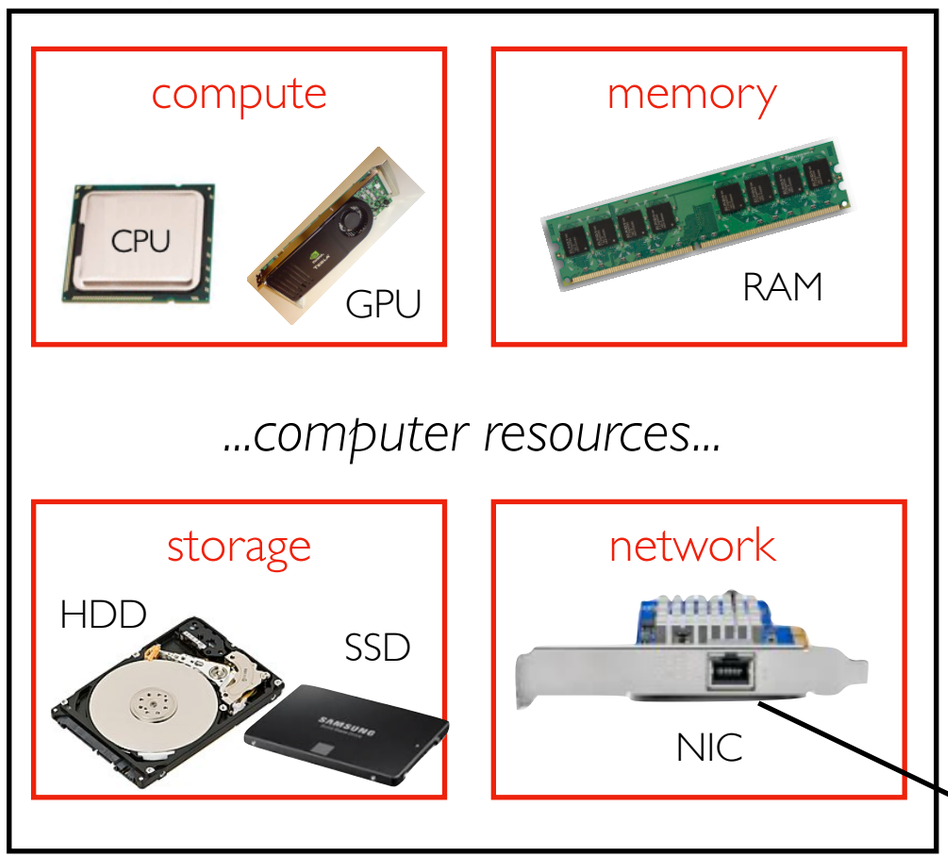
Systems: software for managing computer resources

Other kinds of software (analysis code, applications) rely on systems.

computational resources execute code

storage holds long-term data

a computer:



memory holds data for active usage

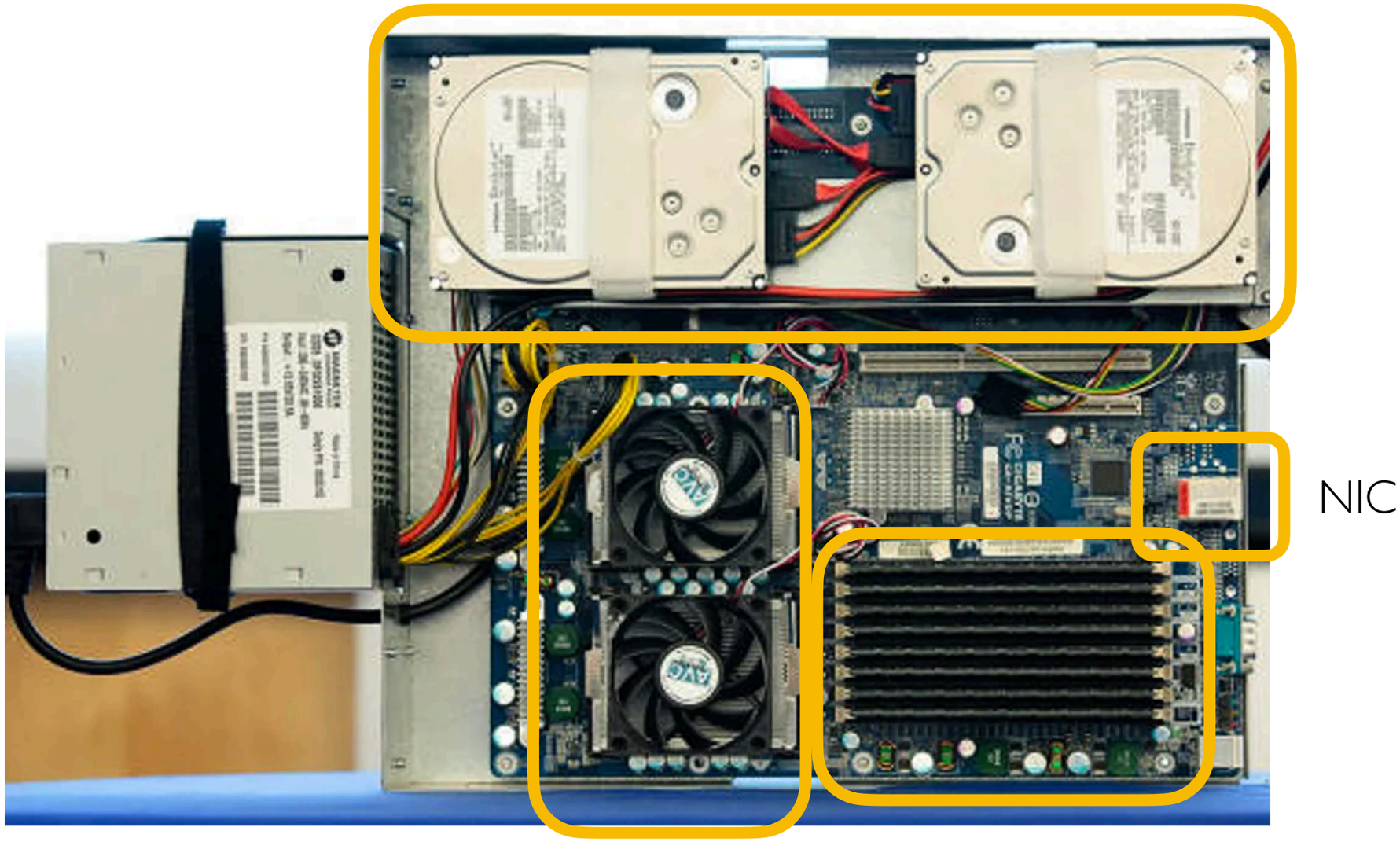
network provides communication between computers

network interface card (NIC)

switches/routers forward packets between computers

https://en.wikipedia.org/wiki/Network_switch#/media/File:2550T-PWR-Front.jpg

A real server



https://www.cnet.com/tech/tech-industry/google-uncloaks-once-secret-server-10209580/

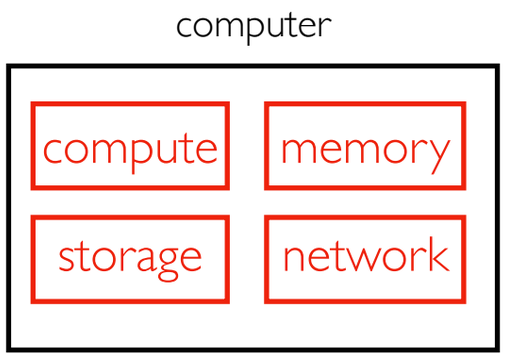
Big Data

Potential problems as datasets grow

- might run too slowly
- might not be able to run at all (for example, not enough memory)

Solutions:

- more efficient code
- use more resources



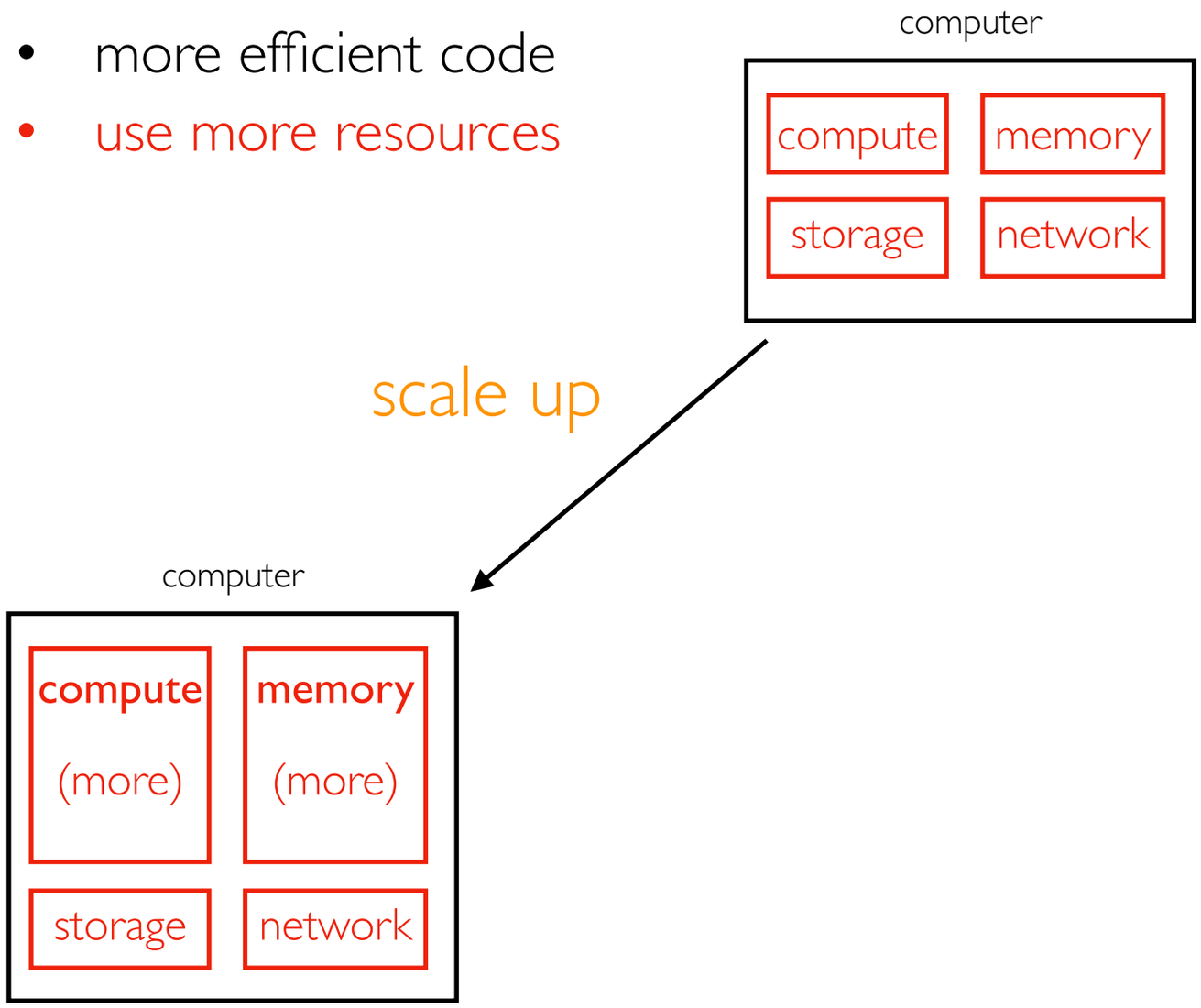
Big Data

Potential problems as datasets grow

- might run too slowly
- might not be able to run at all (for example, not enough memory)

Solutions:

- more efficient code
- use more resources



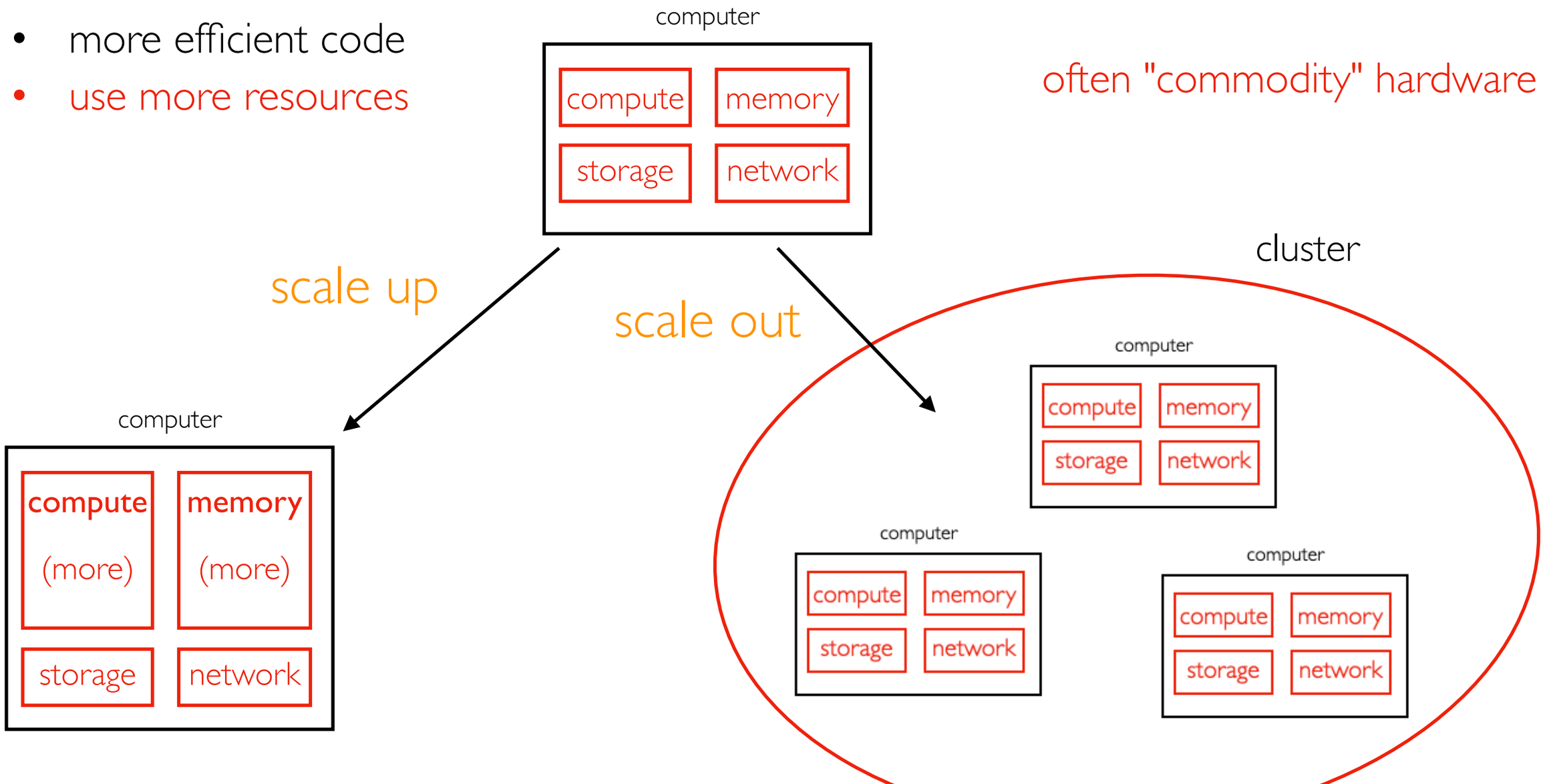
Big Data

Potential problems as datasets grow

- might run too slowly
- might not be able to run at all (for example, not enough memory)

Solutions:

- more efficient code
- use more resources



often "commodity" hardware

Outline

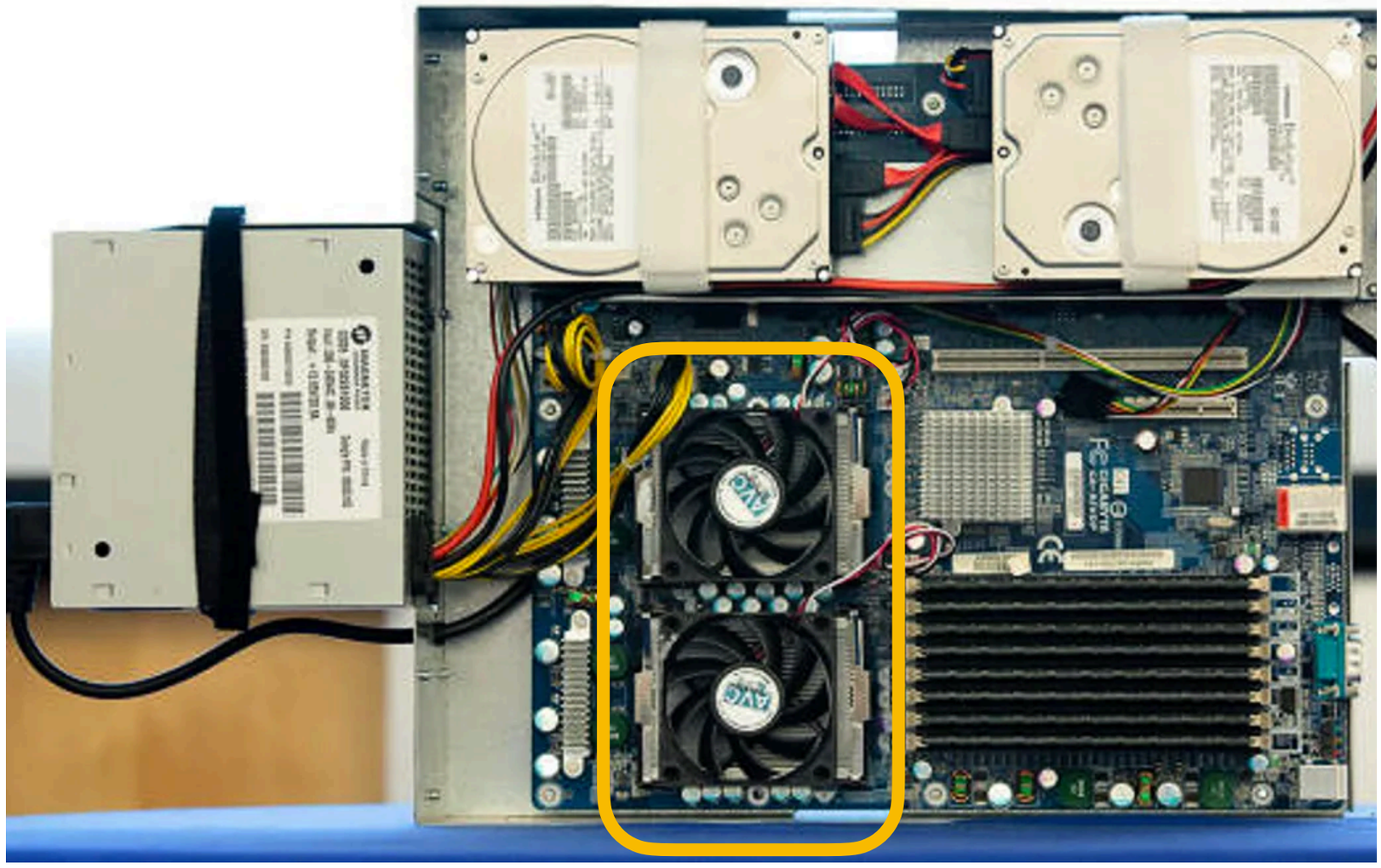
Course Overview

Resources

- Overview
- Compute
- Memory
- Storage
- Network

Deployment

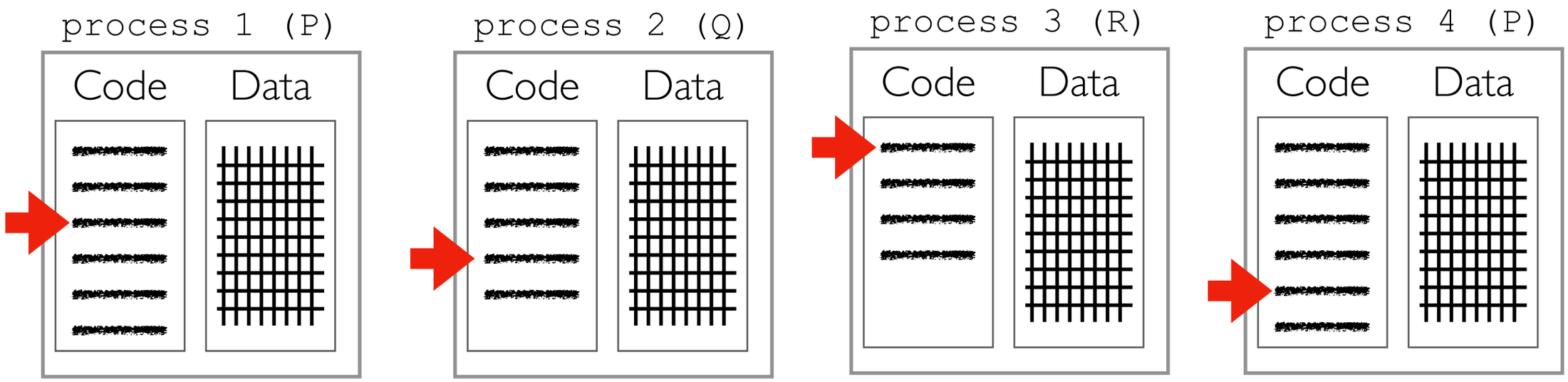
Compute

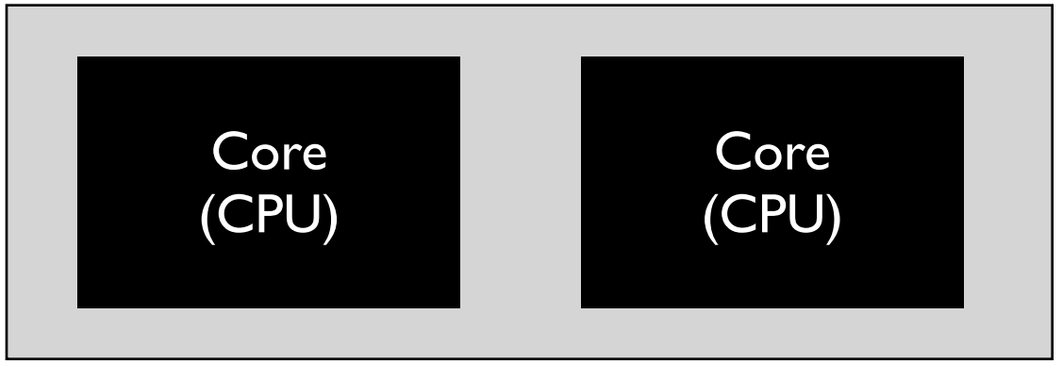


Some computers have multiple CPUs. Modern CPUs typically have multiple cores. Each core works like a CPU and runs programs by executing instructions.

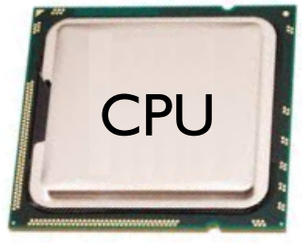
How do cores run machine code?

the operating system "schedules" tasks on cores
(decides when they get to run)

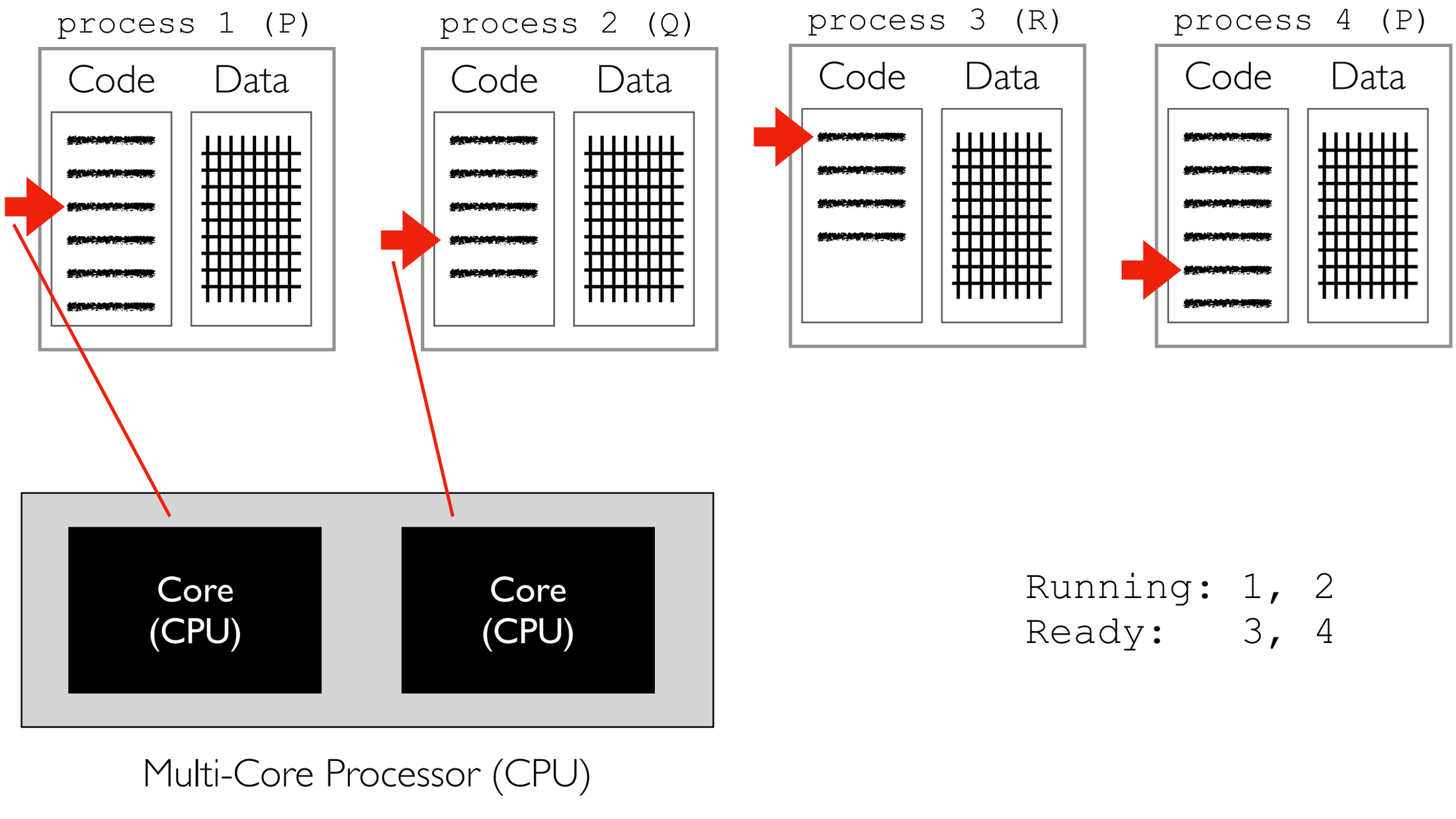


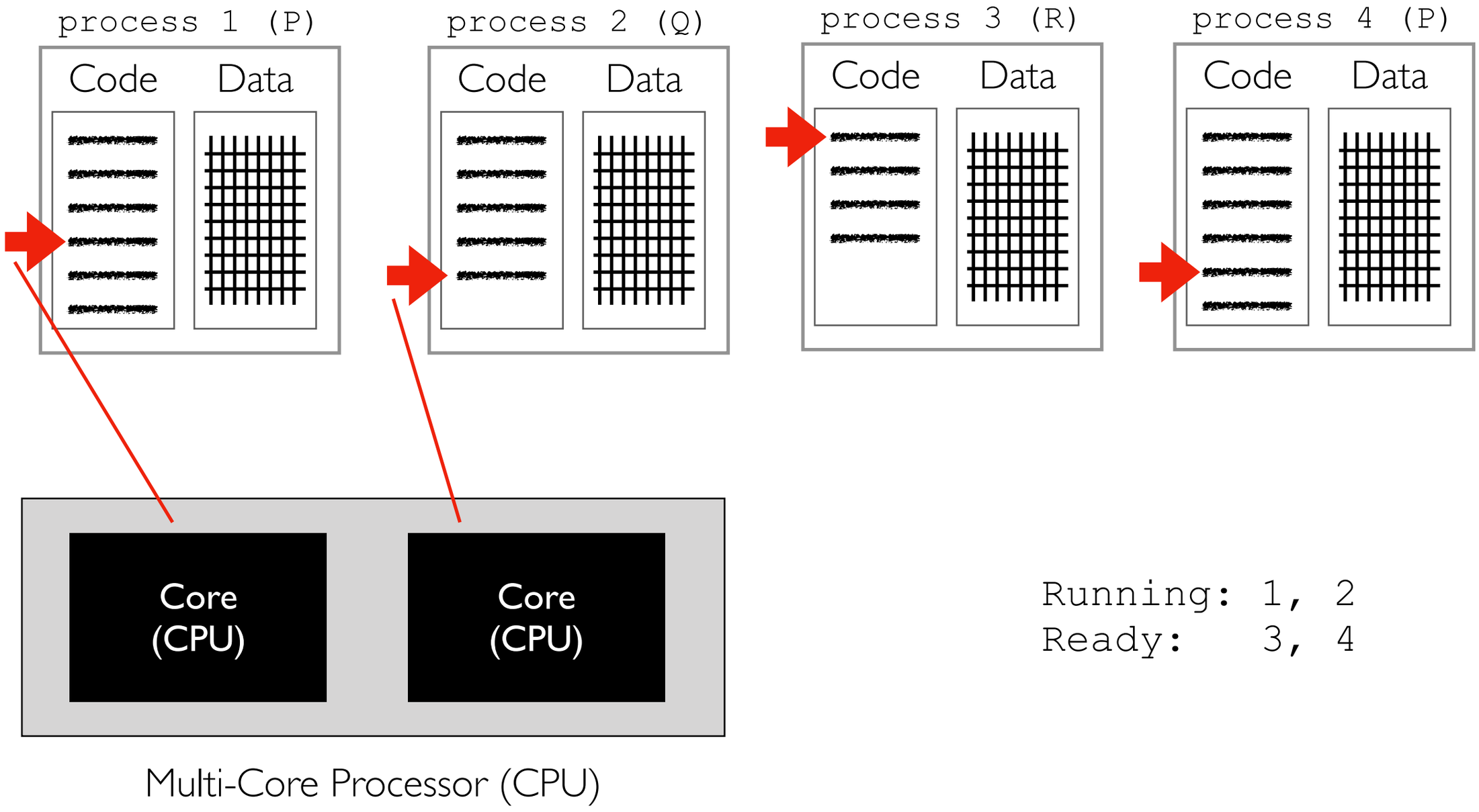


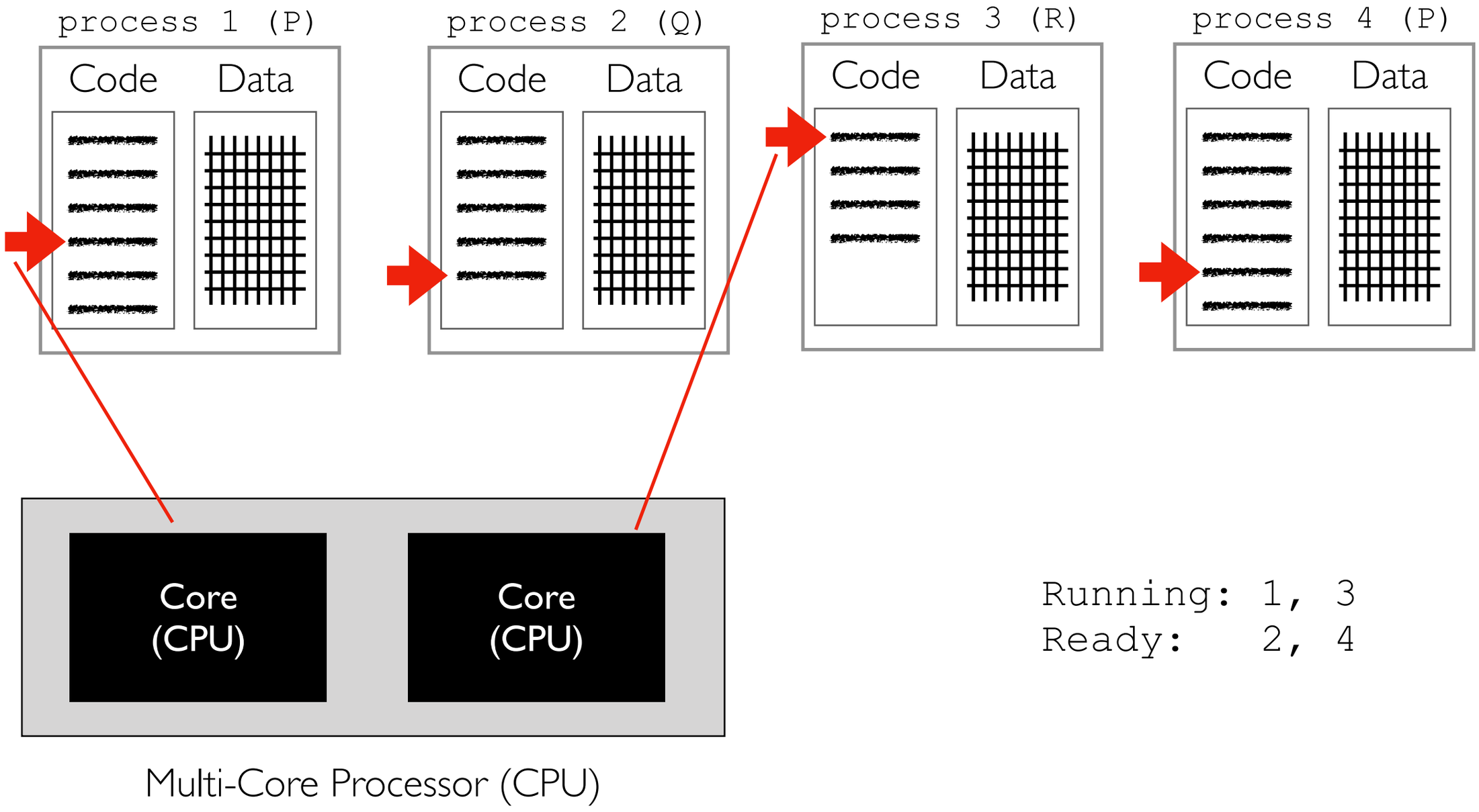
Multi-Core Processor (CPU)

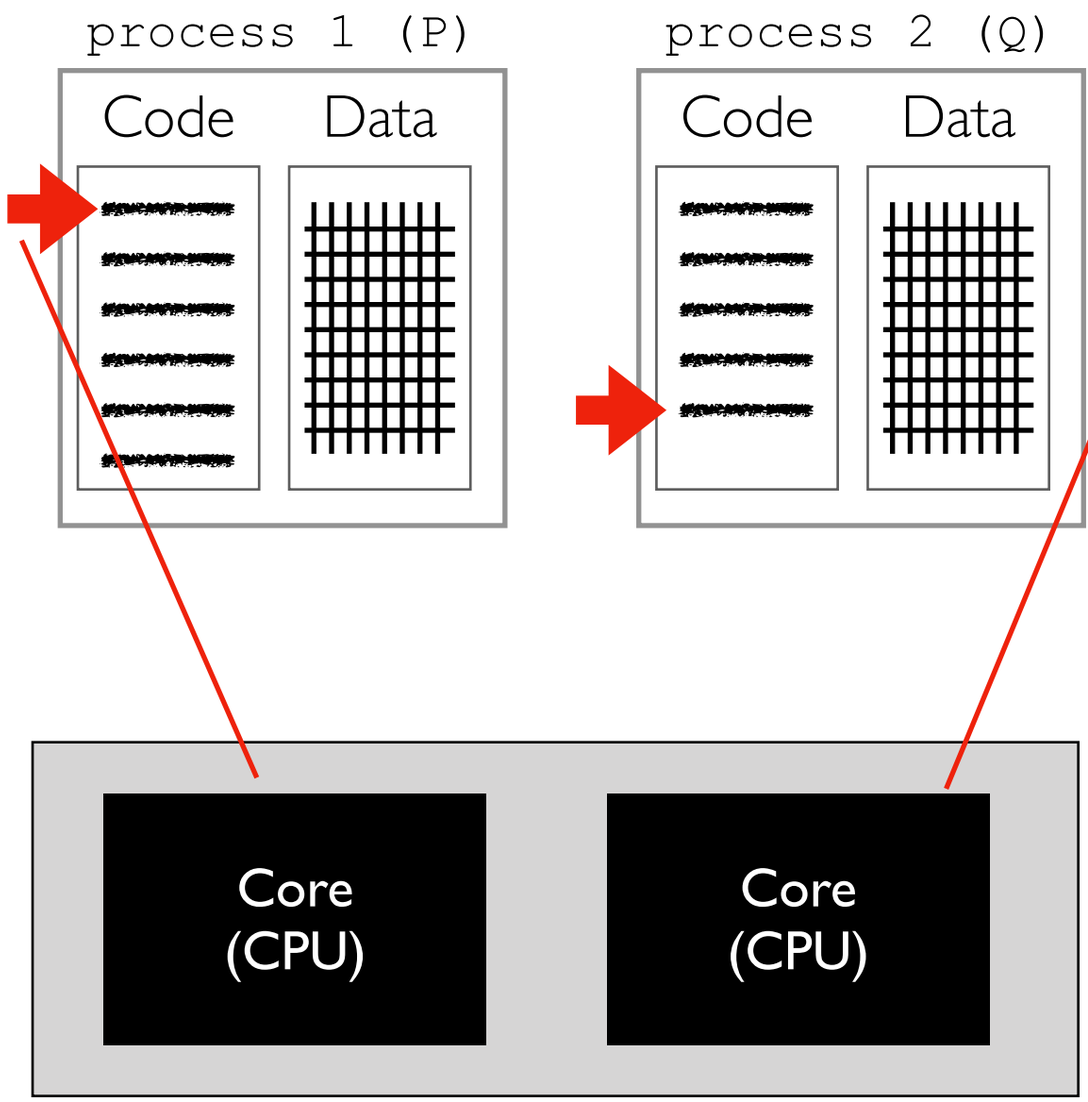


a core can execute instructions for one (or in some cases two) tasks at a time




process 1 (P)
Code
Data
process 2 (Q)
Code
Data
process 3 (R)
Code
Data
process 4 (P)
Code
Data
Core
(CPU)
Core
(CPU)
Multi-Core Processor (CPU)
Running: 1, 2
Ready: 3, 4


process 1 (P)
Code
Data
process 2 (Q)
Code
Data
process 3 (R)
Code
Data
process 4 (P)
Code
Data
Core
(CPU)
Core
(CPU)
Multi-Core Processor (CPU)
Running: 1, 3
Ready: 2, 4

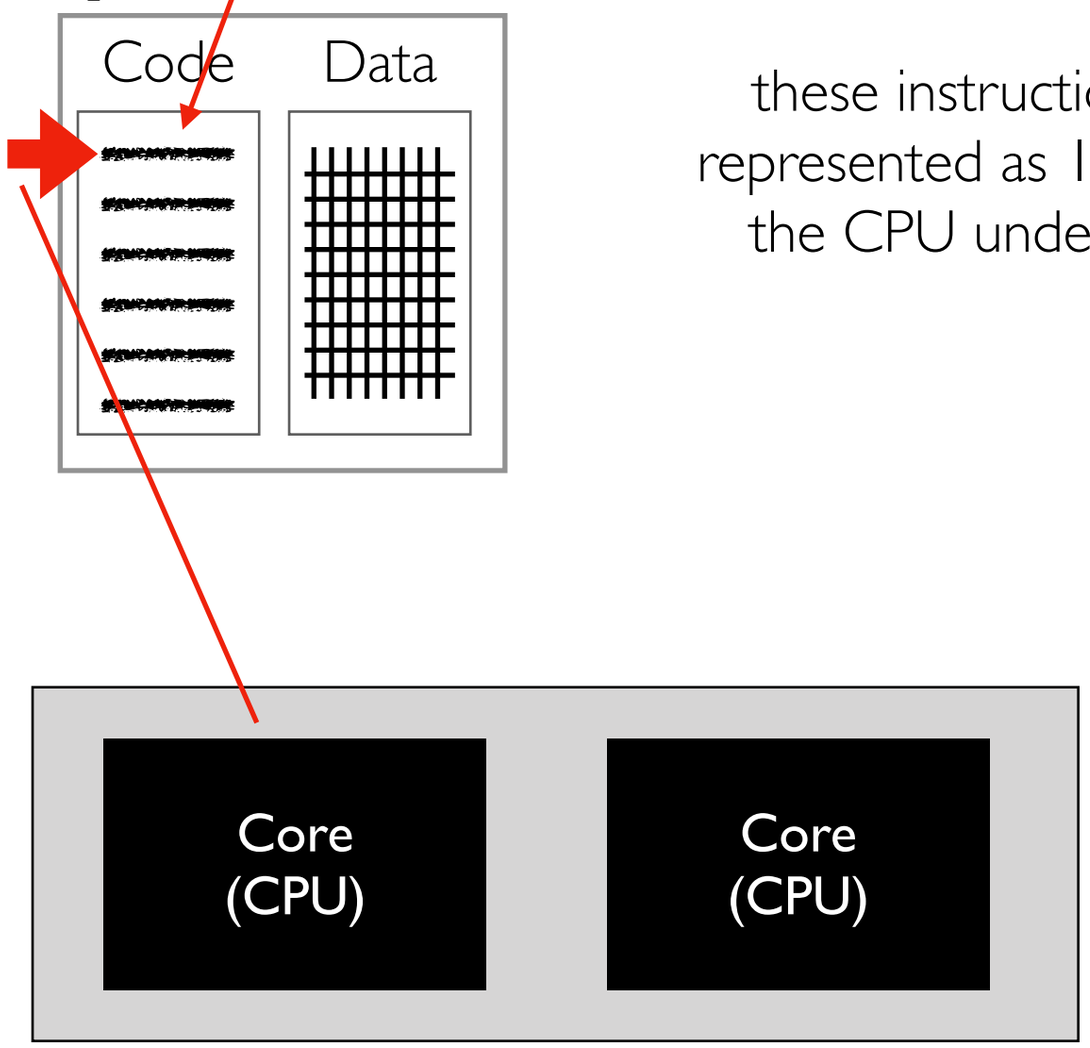


Multi-Core Processor (CPU)

the more cores we have, the more tasks we can run simultaneously (that is, in "parallel")

```
Running: 1, 3
Ready:   2, 4
```

these instructions are in "machine code"
that the CPU can understand



Multi-Core Processor (CPU)

these instructions are
represented as 1's and 0's
the CPU understands

https://pages.cs.wisc.edu/~deppeler/cs354/reference/x86-cheat-sheet.pdf

arithmetic
two operand instructions

```
addl src,dst   dst = dst + src
subl src,dst   dst = dst - src
imull src,dst  dst = dst * src
sall src,dst   dst = dst << src (aka shll)
sarl src,dst   dst = dst >> src (arith)
shrl src,dst   dst = dst >> src (logical)
xorl src,dst   dst = dst ^ src
andl src,dst   dst = dst & src
orl src,dst    dst = dst | src
```

one operand instructions

```
incl dst       dst = dst + 1
decl dst       dst = dst - 1
negl dst       dst = -dst
notl dst       dst = ~dst
```

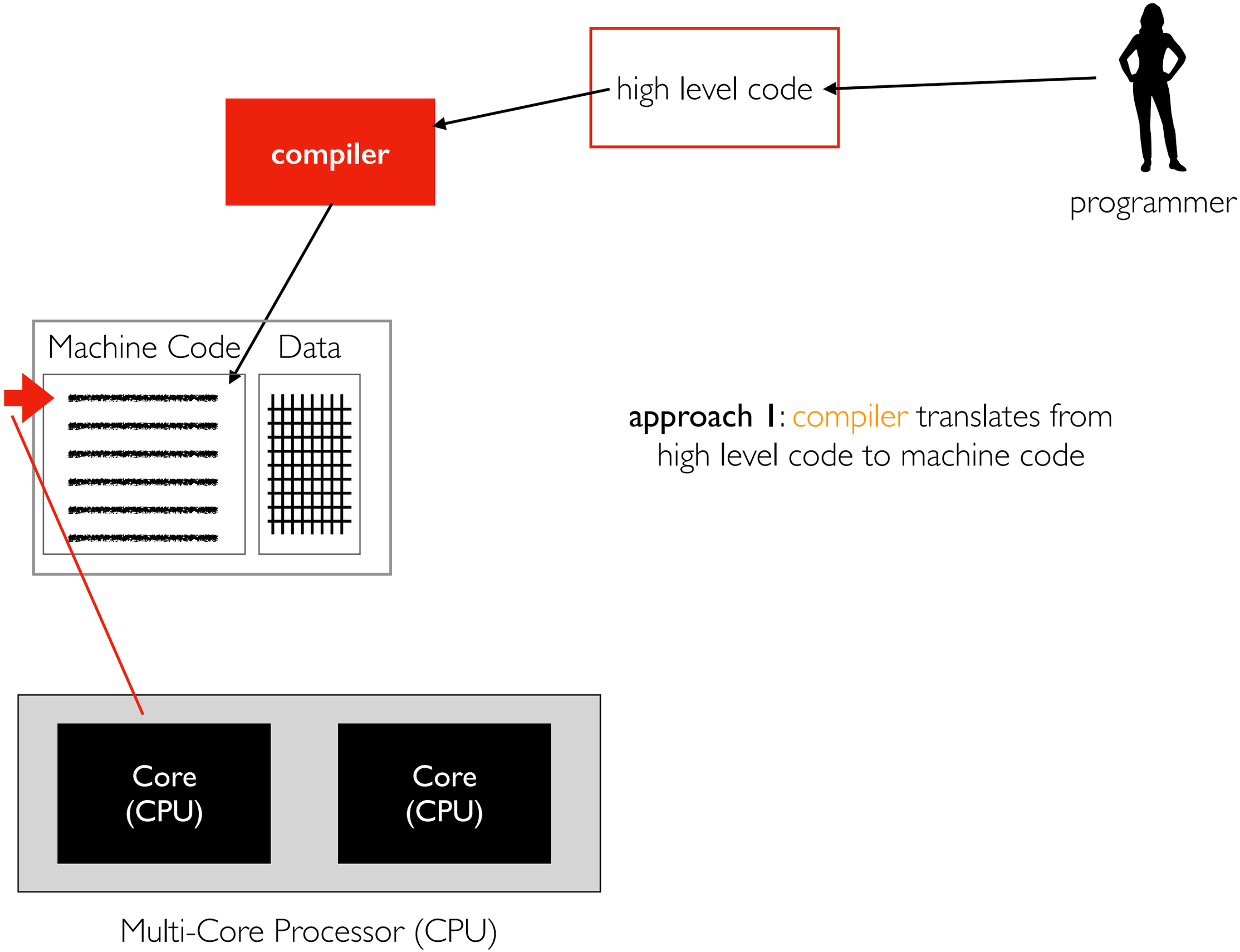
arithmetic ops set CCs implicitly

```
cf=1 if carry out from msb
zf=1 if dst==0,
sf=1 if dst < 0 (signed)
of=1 if two's complement
     (signed) under/overflow
```

how do we bridge the gap between "high level"
code (Python/Java/etc) and machine code?

how do we bridge the gap between "high level" code (Python/Java/etc) and machine code?

Note: we'll primarily write Python this semester, but it helps to explore this in general to understand how systems like Spark work (which is written in Scala and uses the Java Virtual Machine)



approach 1: compiler translates from high level code to machine code

high level code

programmer

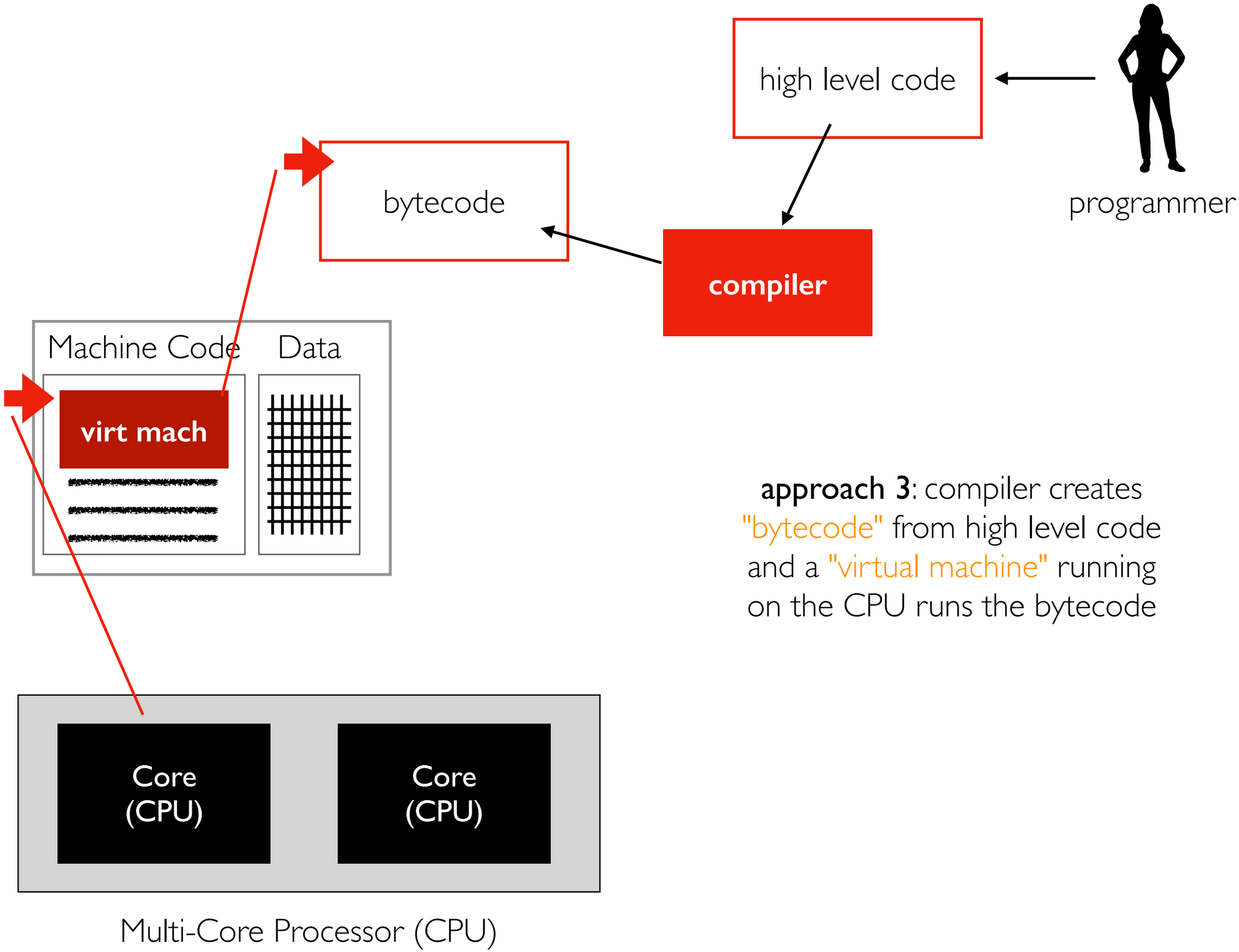
Machine Code | Data

interpreter

Core (CPU) | Core (CPU)

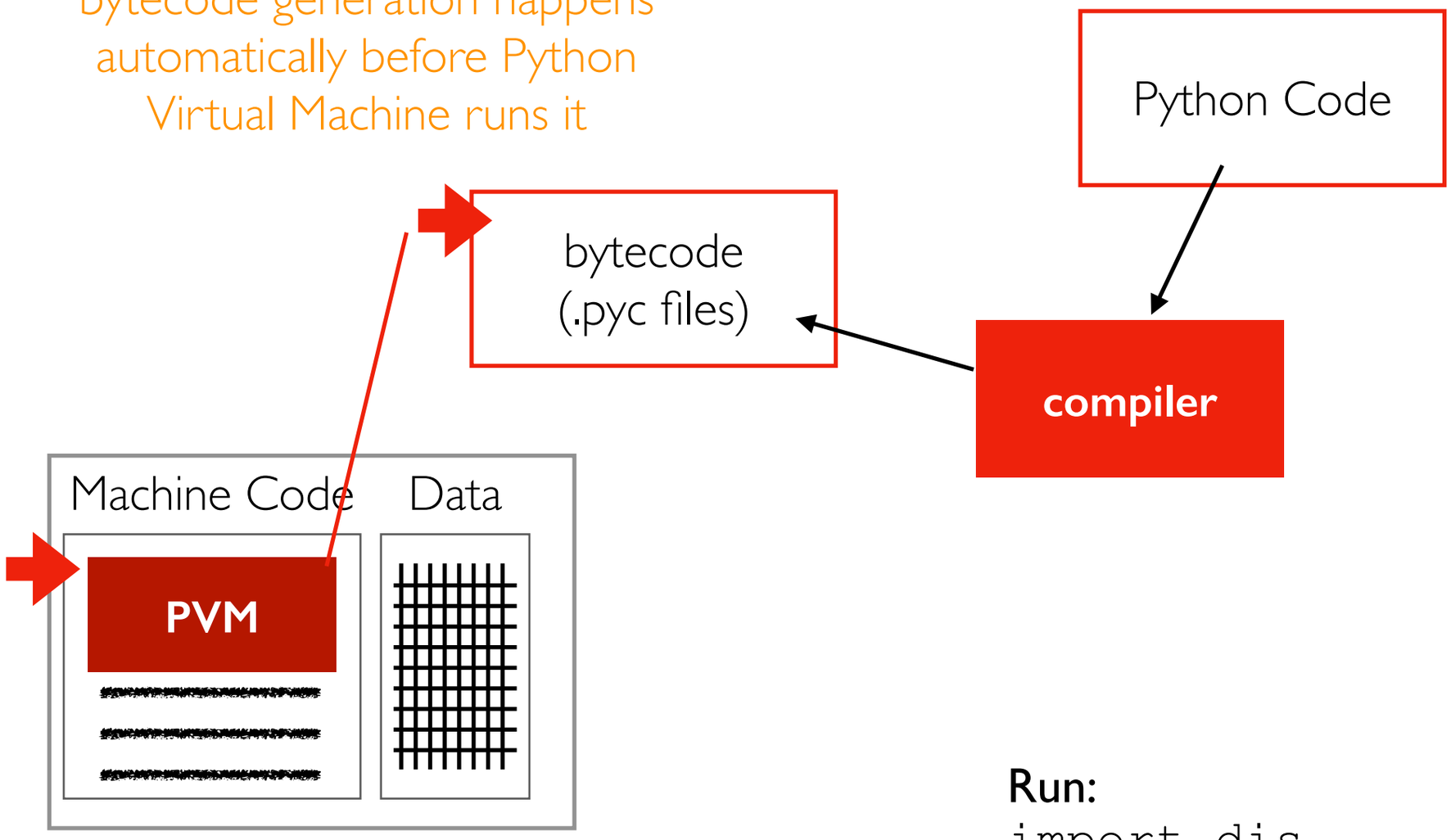
Multi-Core Processor (CPU)

approach 2: CPU runs an interpreter program that loops over programmer's code and runs it



approach 3: compiler creates "bytecode" from high level code and a "virtual machine" running on the CPU runs the bytecode

when you run "python3 ..."
bytecode generation happens
automatically before Python
Virtual Machine runs it



Run:

```
import dis
dis.dis("z = x + y * 2")
```

when you run "python3 ..."
bytecode generation happens
automatically before Python
Virtual Machine runs it

Python Code

bytecode
(.pyc files)

compiler

Machine Code Data

PVM

JVM bytecode is "closer"
to machine code

Java Code

bytecode
(.class files)

Java compiler

javac mycode.java

java mycode

Machine Code Data

JVM

when you run "python3 ..."
bytecode generation happens
automatically before PVM runs it

Python Code

bytecode
(.pyc files)

compiler

Jython
compiler

Machine Code Data

PVM

JVM bytecode is "closer"
to machine code

Java Code

bytecode
(.class files)

Java compiler

javac mycode.java

java mycode

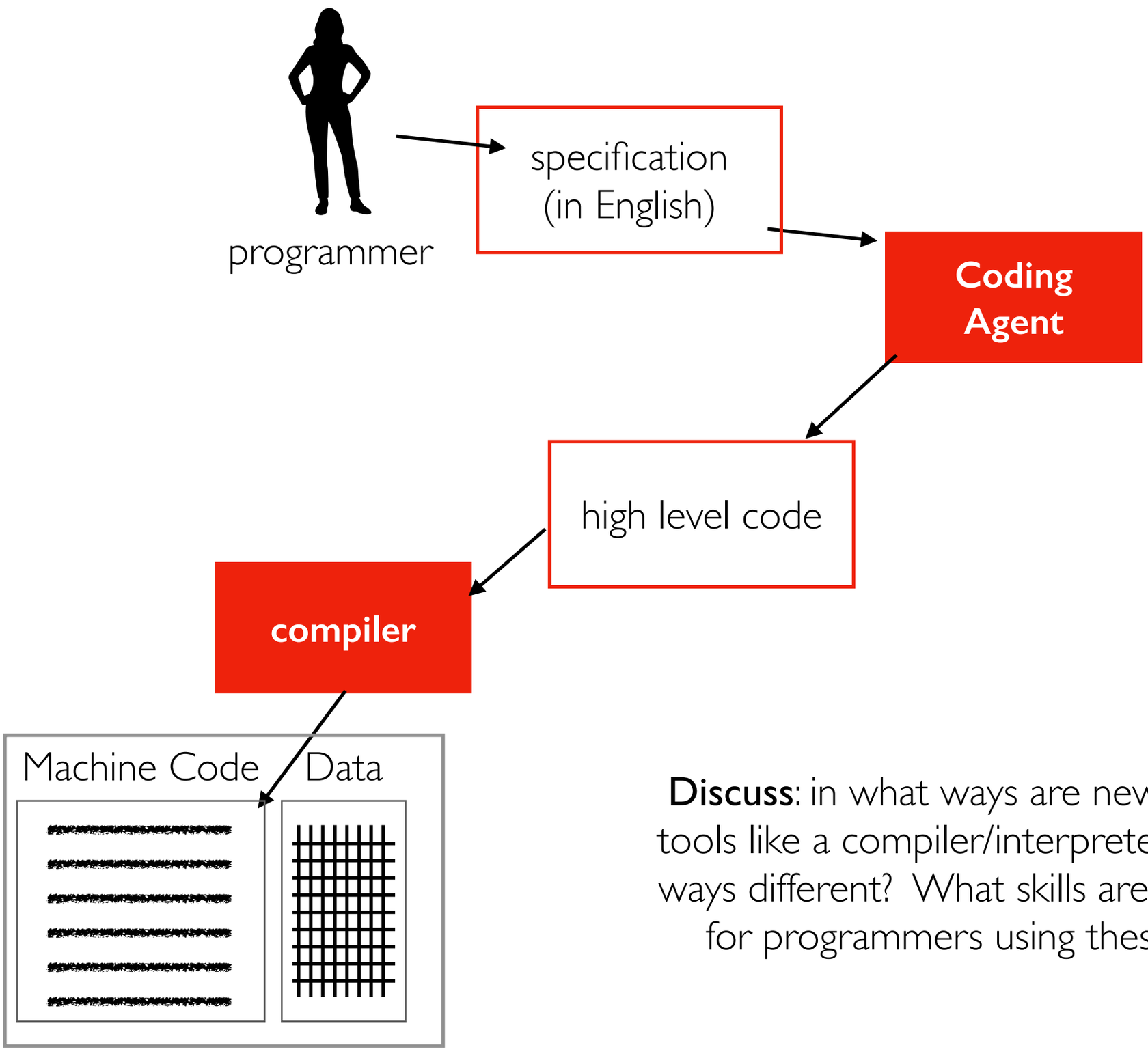
Machine Code Data

JVM

Scala compiler

Scala Code

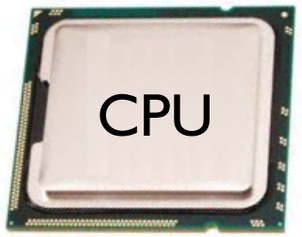
The Future?



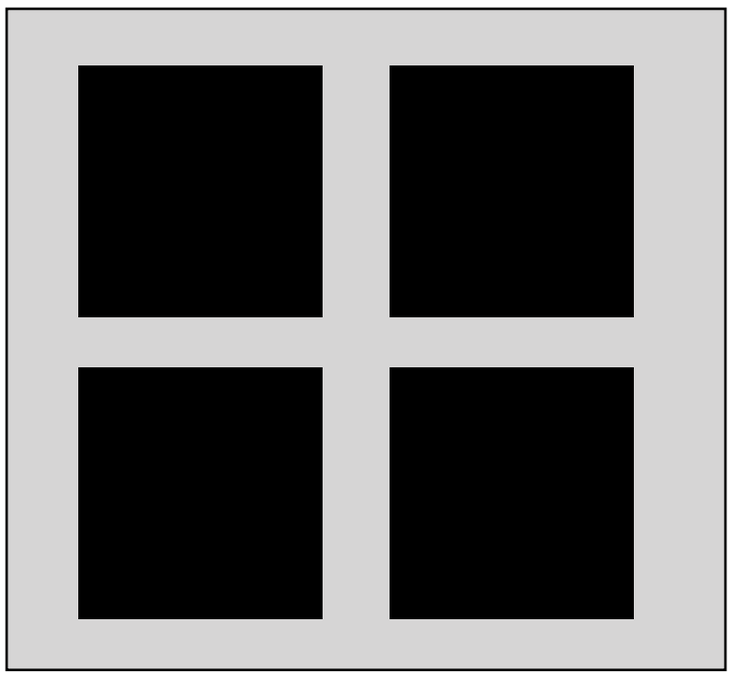
Discuss: in what ways are new AI-based tools like a compiler/interpreter? In what ways different? What skills are important for programmers using these tools?

What are alternatives to CPUs for compute?

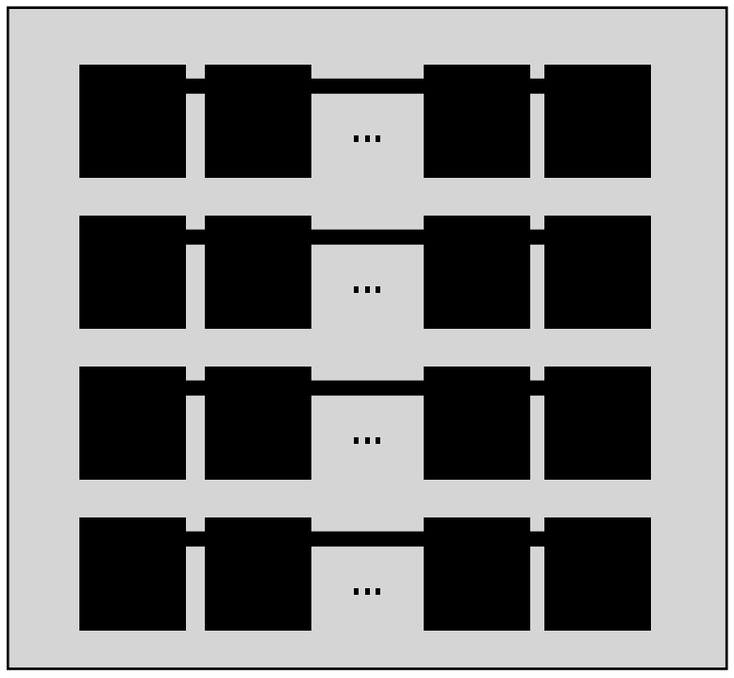
GPUs (graphical processing units) are an alternative compute resource.



https://en.wikipedia.org/wiki/Nvidia_Tesla



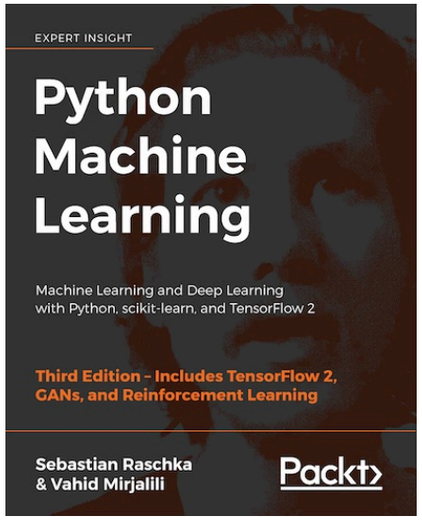
few cores that are fast,
flexible, independent



many cores that are slow,
float-optimized, coordinated

GPU vs. CPU: Cost Comparison

TOP HAT



The GPU is 30% cheaper but 28x faster at floating-point operations!

Specifications	Intel® Core™ i7-6900K Processor Extreme Ed.	NVIDIA GeForce® GTX™ 1080 Ti
Base Clock Frequency	3.2 GHz	< 1.5 GHz
Cores	8	3584
Memory Bandwidth	64 GB/s	484 GB/s
Floating-Point Calculations	409 GFLOPS	11300 GFLOPS
Cost	~ $1000.00	~ $700.00

https://sebastianraschka.com/books.html

Resource metric: FLOPS (floating-point operations per second)

- floating-point ops: add, mult, etc (how to weight?)
- prefixes: K (thousand), M (million), G (billion), T (trillion)
  - ➡ how many TFLOPS does the above GPU provide?

How to measure?

- a "benchmark", a program that we run for the sake of measuring performance

Outline

Course Overview

Resources

- Overview
- Compute
- Memory
- Storage
- Network

Deployment

RAM: Random Access Memory

What is "Random"?

"Random" means we can jump around and access data from different locations efficiently.

In contrast, some devices that hold data are only efficient sequentially:

https://en.wikipedia.org/wiki/Cassette_tape

Bits

RAM holds bits.

A "bit" holds a 0 or a 1 (two possible values).

2 bits can hold 00, 01, 10, 11 (four possible values).

N bits can hold 2^N possible values.

Representation

Different encodings/representations decide what a combination of bits mean.

bits			
000	A	0	−4
001	B	1	−3
010	C	2	−2
011	D	3	−1
100	E	4	0
101	F	5	1
110	G	6	2
111	H	7	3

colors
images
floats
etc.

Bytes

A byte is 8 bits, so can hold $2^8 = 256$ possible values.

RAM is "byte addressable"

- each byte of data has it's own address the CPU can use to access it
- extracting a single bit from a byte actually involves more steps than using the whole byte

Units:

- 1 KB = 1024 bytes (or sometimes 1000 bytes)
- 1 MB = 1024 KB (or sometimes 1000 KB)
- 1 GB = 1024 MB (or sometimes 1000 MB)
- 1 TB = 1024 GB (or sometimes 1000 GB)

RAM Characteristics

Characteristics

- small (for example, your course VM will have ~8 GB)
- volatile (contents lost upon reboot)
- fast (much faster that storage devices)

Some uses

- actively used data (e.g., Python list, program code, DataFrame)
- copies of "hot" data (frequently accessed) from storage

Outline

Course Overview

Resources

- Overview
- Compute
- Memory
- Storage
- Network

Deployment

Block devices: storing 0s and 1s

Hard Disk Drives (HDDs)

- 0s/1s stored on spinning magnatized platter
- moving head reads/writes data

Solid State Disks (SSDs)

- 0s/1s stored in charged cells
- no moving parts (faster)

Both are "block devices"

- data is read/written in blocks of many bytes (for example, 0.5 KB)
- reading 1 byte or 1 block takes same time

HDD and SSD Characteristics

Characteristics

- large (> 1 TB devices are affordable)
- nonvolatile (contents are NOT lost upon reboot)
- slow (much slower that memory)

Some uses

- large datasets
- data that needs to be preserved long term

Metrics

Capacity

- how much data can be stored?
- measured in bytes (for example, 500 GB)

Throughput

- how fast can data be read/written?
- measure in bytes/second (for example, 200 MB/s)
- throughput will depend on access pattern (for example, spinning disks have low throughput for random accesses)

Latency

- how long does it take to do one I/O (e.g., 10 ms)

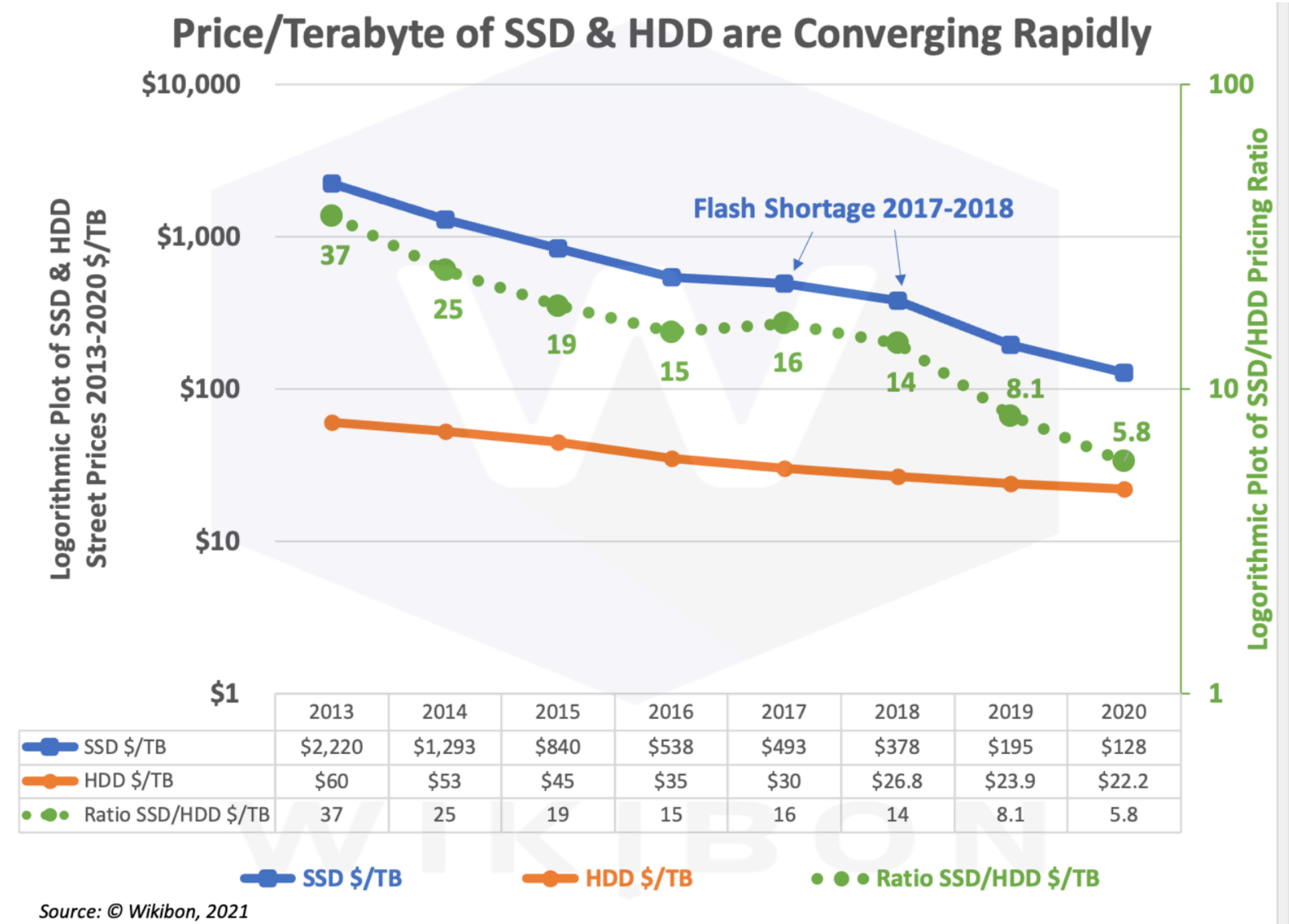


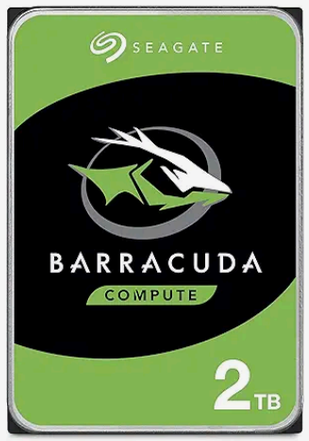
Figure 3 – Flash & HDD Pricing 2013 – 2020

Source: © Wikibon, 2021. Wikibon uses historical data is from multiple sources, including IDC, Gartner, Kitguru, Nidec, Trendfocus, and Wells Fargo LLC

https://wikibon.com/qlc-flash-hamrs-hdd/

Update: Amazon's "Overall Pick" in Aug 2025

Overall Pick



Seagate BarraCuda 2TB Internal Hard Drive HDD – 3.5 Inch SATA 6Gb/s 7200 RPM 256MB Cache – Frustration Free Packaging (ST2000DM008/ST2000DMZ08)

4.6 ★★★★☆ (101.8K)

2K+ bought in past month

$67.17

prime Today

FREE delivery **Today 5 PM - 10 PM**

Add to cart

More Buying Choices

$42.04 (8+ used & new offers)

$34/TB

Overall Pick

Samsung 990 EVO Plus SSD 2TB, PCIe Gen 4x4, Gen 5x2 M.2 2280, Speeds Up-to 7,250 MB/s, Upgrade Storage for PC/Laptops, HMB Technology and Intelligent Turbowrite 2.0, (MZ-…

4.8 ★★★★★ (6.1K)

10K+ bought in past month

Limited time deal

$119.99 List: ~~$176.99~~

prime Overnight

FREE delivery **Overnight 7 AM - 11 AM**

Add to cart

More Buying Choices

$116.11 (16+ used & new offers)

$60/TB

(1.8x more expensive)

Outline

Course Overview

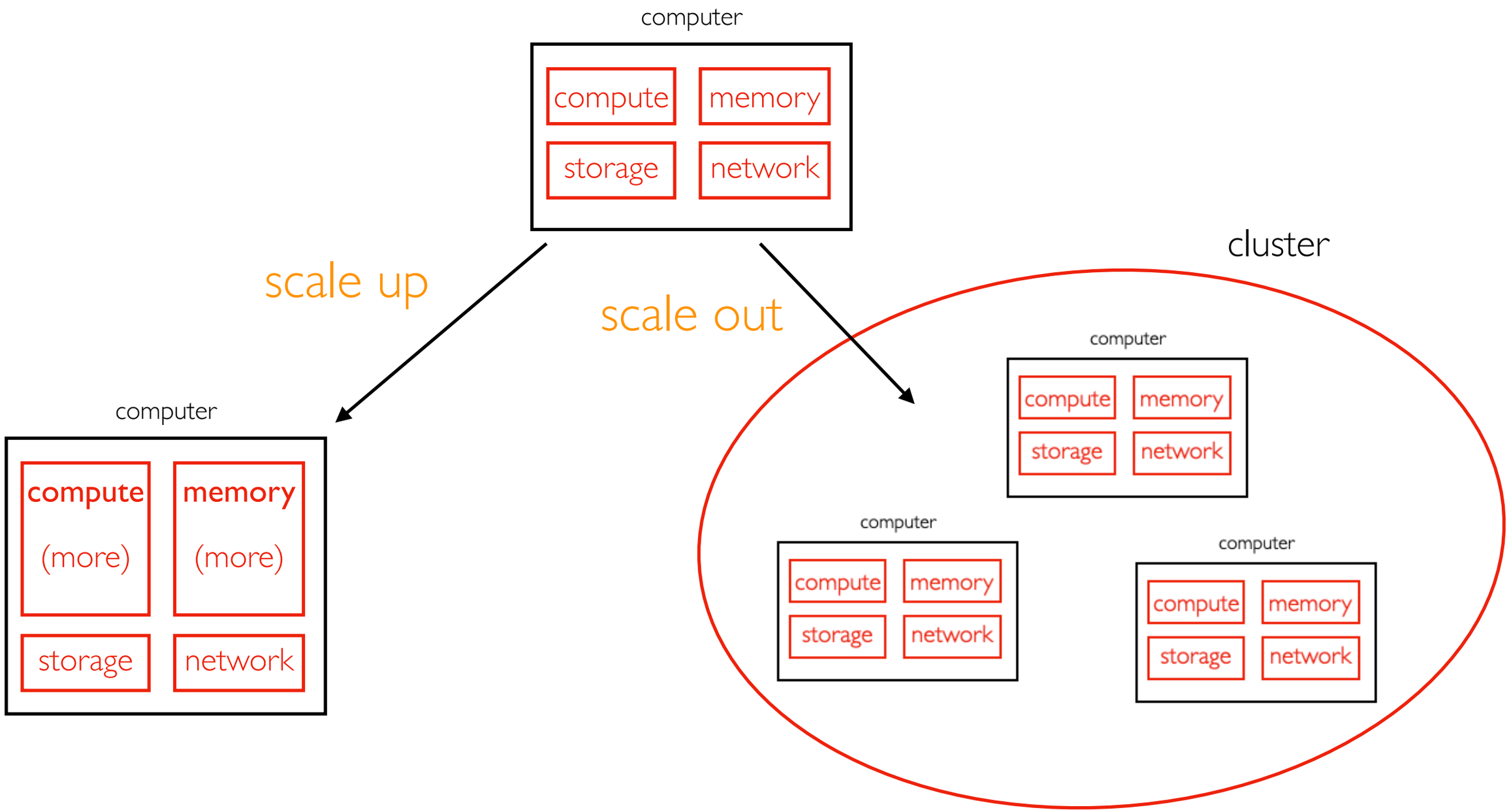
Resources

- Overview
- Compute
- Memory
- Storage
- Network

Deployment

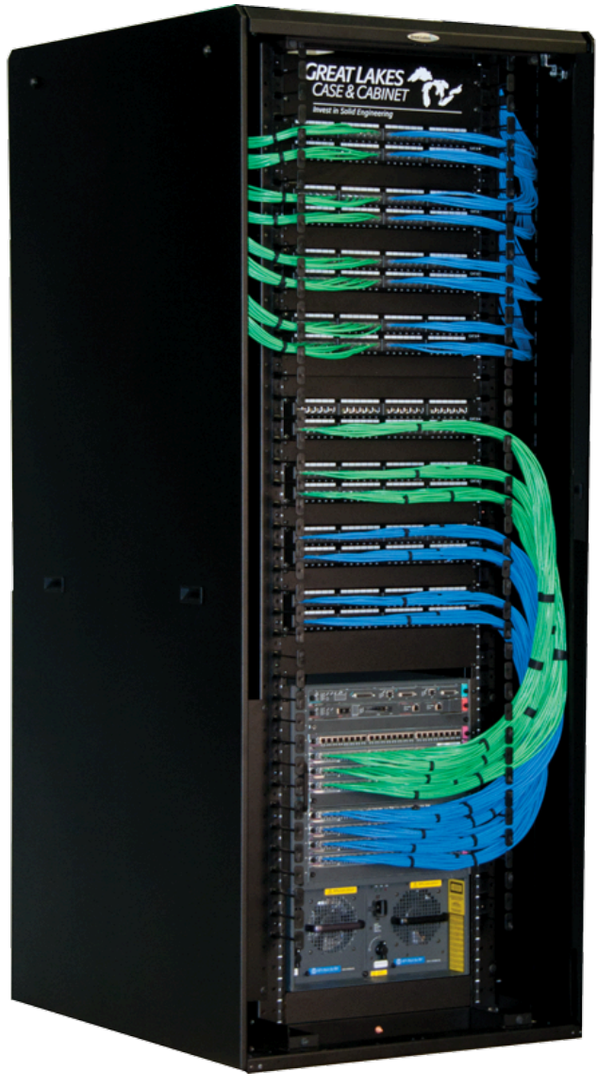
Network

When scaling out, many nodes (computers) will be communicating via a network.



Network

Server



Rack

https://www.dotmagazine.online/issues/digital-infrastructure-and-transforming-markets/data-center-models

https://buy.hpe.com/us/en/servers/proliant-dl-servers/proliant-dl10-servers/proliant-dl20-server/hpe-proliant-dl20-gen10-plus-e-2336-2-9ghz-6-core-1p-16gb-u-4sff-500w-rps-server/p/p44115-b21?ef_id=Cj0KCQiAt66eBhCnARIsAKf3ZNFJsg49UV6Zm33R7IkRqi-XOd_JECmdyqNMAm2CKLSm_F-z6JTYDTQaAgM1TEALw_wcB:G:s&s_kwcid=AL!13472!3!331628972784!!!g!318267171339!!170791836 9!670764174 19&gclsrc=aw.ds&gclid=Cj0KCQiAt66eBhCnARIsAKf3ZNFJsg49UV6Zm33R7IkRqi-XOd_JECmdyqNMAm2CKLSm_F-z6JTYDTQaAgM1TEALw_wcB

https://www.server-rack-online.com/gl910ent-4048sss.html?utm_medium=shoppingengine&utm_source=googlebase&utm_source=google&utm_medium=cpc&adpos=&scid=scplpgl910ent-4048sss&sc_intid=gl910ent-4048sss&gclid=Cj0KCQiAt66eBhCnARIsAKf3ZNEMYINPAA0RFGQIF0DsieCM6oh7i3kuJvJIpnmJAIOpAJ3RWT11QMAaAqRnEALw_wcB

Network

Server

Data Center

https://www.dotmagazine.online/issues/digital-infrastructure-and-transforming-markets/data-center-models

https://buy.hpe.com/us/en/servers/proliant-dl-servers/proliant-dl10-servers/proliant-dl20-server/hpe-proliant-dl20-gen10-plus-e-2336-2-9ghz-6-core-1p-16gb-u-4sff-500w-rps-server/p/p44115-b21?ef_id=Cj0KCQiAt66eBhCnARIsAKf3ZNFJsg49UV6Zm33R7IkRqi-XOd_JECmdyqNMAm2CKLSm_F-z6JTYDTQaAgMTEALw_wcB:G:s&s_kwcid=AL!13472!3!331628972784!!!g!318267171339!!17079183691670764174 19&gclsrc=aw.ds&gclid=Cj0KCQiAt66eBhCnARIsAKf3ZNFJsg49UV6Zm33R7IkRqi-XOd_JECmdyqNMAm2CKLSm_F-z6JTYDTQaAgMTEALw_wcB

https://www.server-rack-online.com/gl910ent-4048sss.html?utm_medium=shoppingengine&utm_source=googlebase&utm_source=google&utm_medium=cpc&adpos=&scid=scplpgl910ent-4048sss&sc_intid=gl910ent-4048sss&gclid=Cj0KCQiAt66eBhCnARIsAKf3ZNEMYINPAA0RFGQIF0DsieCM6oh7i3kuJvJlpnmJAIOpAJ3RWT11QMAaAqRnEALw_wcB

Topology

Example configuring Hadoop File System (HDFS) to store data based on network topology:

python Example

```
#!/usr/bin/python3
# this script makes assumptions about the physical environment.
#  1) each rack is its own layer 3 network with a /24 subnet, which
# could be typical where each rack has its own
#     switch with uplinks to a central core router.
#
#             +-----------+
#             |core router|
#             +-----------+
#            /             \
#   +-----------+        +-----------+
#   |rack switch|        |rack switch|
#   +-----------+        +-----------+
#   | data node |        | data node |
#   +-----------+        +-----------+
#   | data node |        | data node |
#   +-----------+        +-----------+
#
```

https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/RackAwareness.html

Metrics

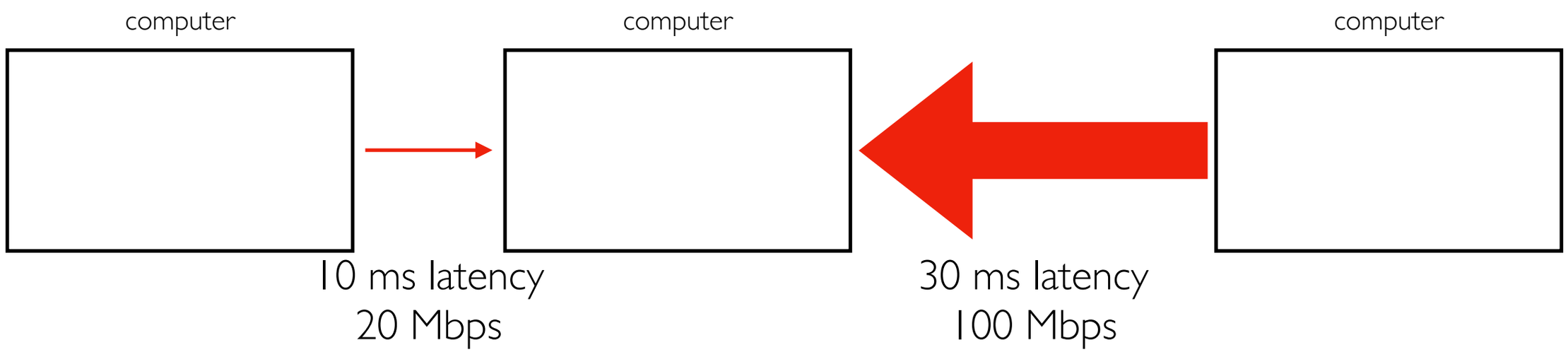
TOP HAT

Latency

- how long does it take to send messages between two points
- seconds, milliseconds (ms), etc

Bandwidth/Throughput

- how many **bits** can be sent per second?
- Mbps (mega bits per second -- note lower case "b")
- What is faster, 10 Mbps or 10 MB/s?



Outline

Course Overview

Resources

- Overview
- Compute
- Memory
- Storage
- Network

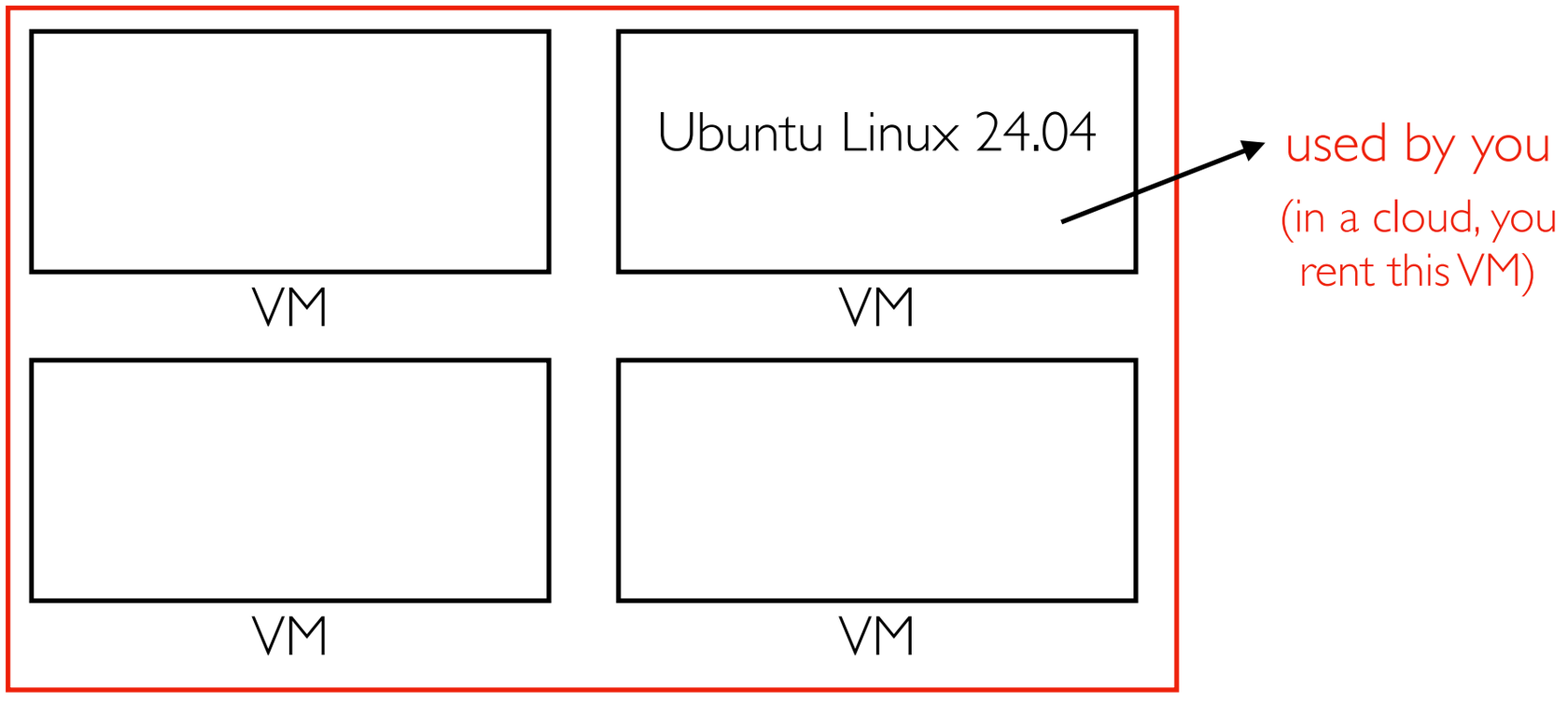
Deployment

Virtual Machines

Deployment means running code somewhere

- often a major undertaking when working with clusters

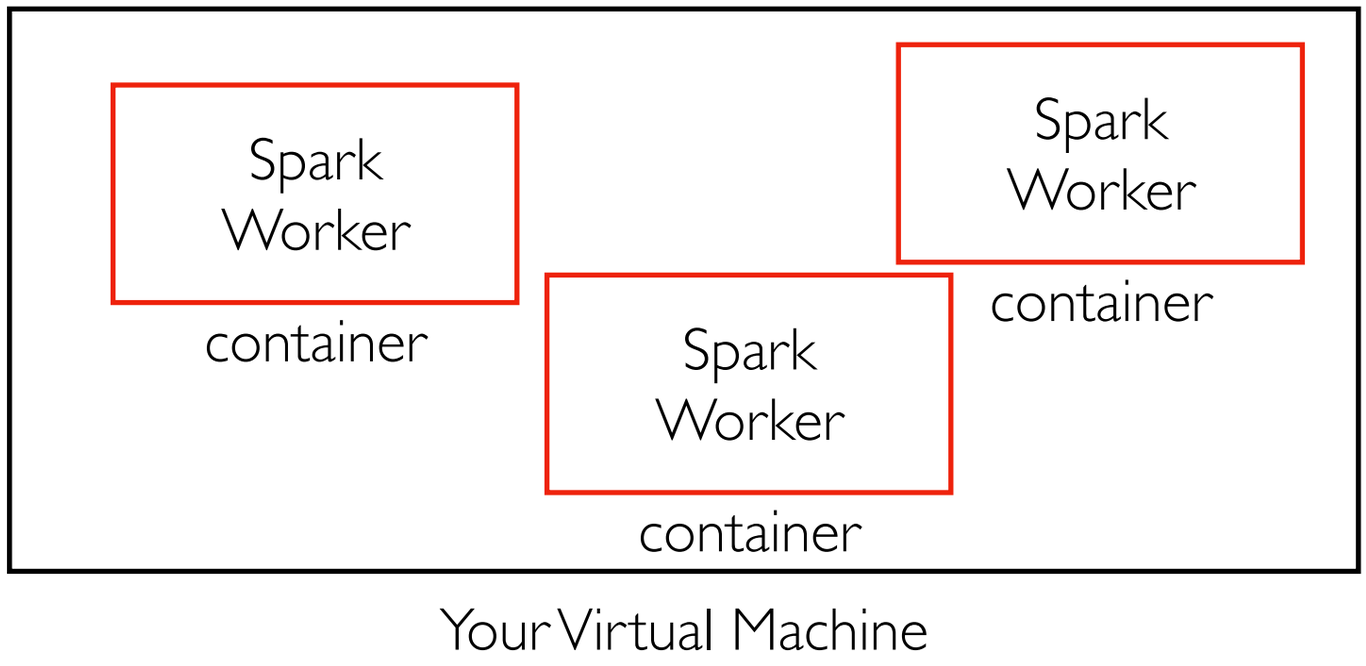
We'll be deploying systems on using virtual machines (VMs).



Docker Containers

Containers are a lightweight alternative to virtual machines.

You'll run Docker containers this semester to have your own "mini cluster"



Resources of the "cluster" are limited to those of a single VM, so we'll scale projects accordingly. But the techniques will apply to large clusters and datasets.

Conclusion

Systems manage resources like compute, memory, storage, and networking.

Big data systems use specialized or distributed resources to make it faster to work with large datasets.

We'll deploy these systems using containers and VMs.

Tasks for next time:

- read syllabus, become familiar with course websites
- introduce yourself via the welcome form