

[544] Intro to Big Data Systems

Tyler Caraza-Harter

Outline

Course Overview

- Introductions
- Main sites: tyler.caraza-harter.com, Semester Repo, Canvas
- Other tools: Email, TopHat, Piazza, GitLab, Anki

Resources

- Overview
- Compute
- Memory
- Storage
- Network

Deployment

Introductions

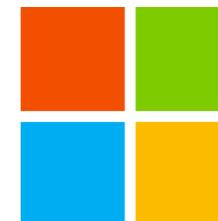
Tyler Caraza-Harter (Section I instructor)

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



Industry experience

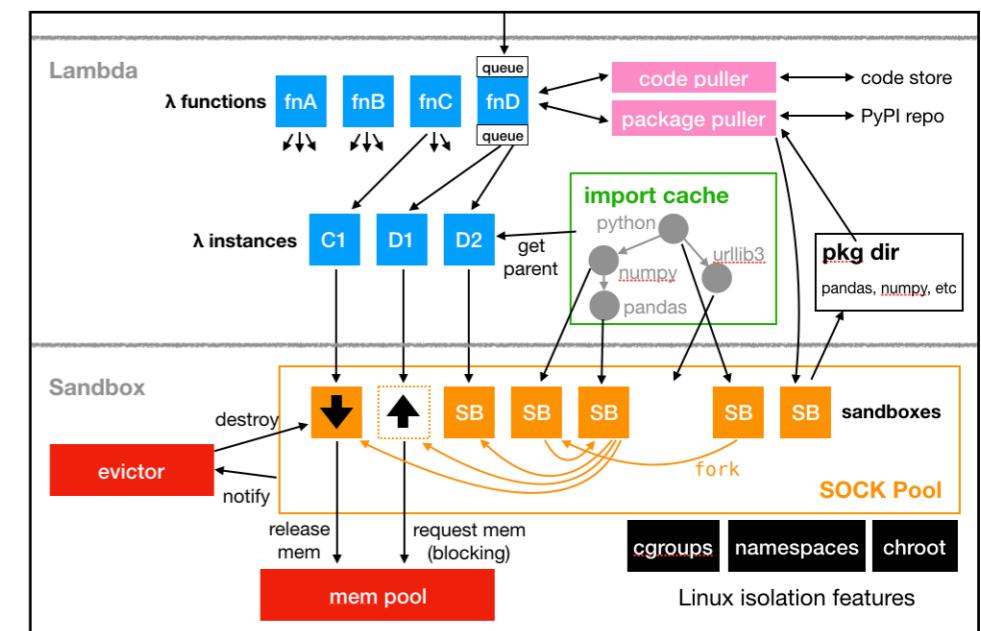
- Worked at Microsoft on SQL Server and Cloud
- Other internships/collaborations:
Qualcomm, Google, Facebook, Tintri, Bauplan



Open source

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

Meena Syamkumar teaches section 2!



Who are You?

Year in school? Major?

What CS courses have people taken before?

- 320? 400? 537/564/640?

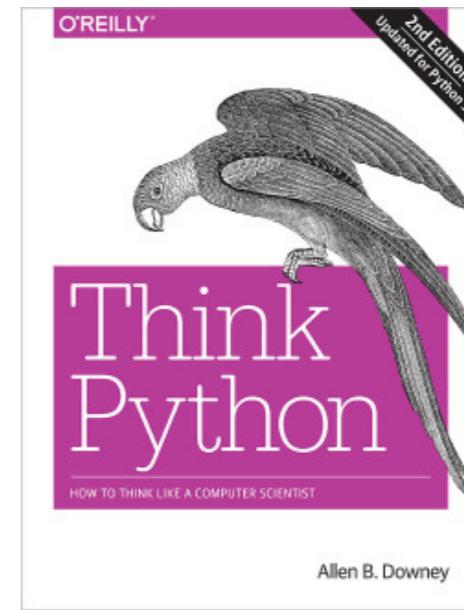
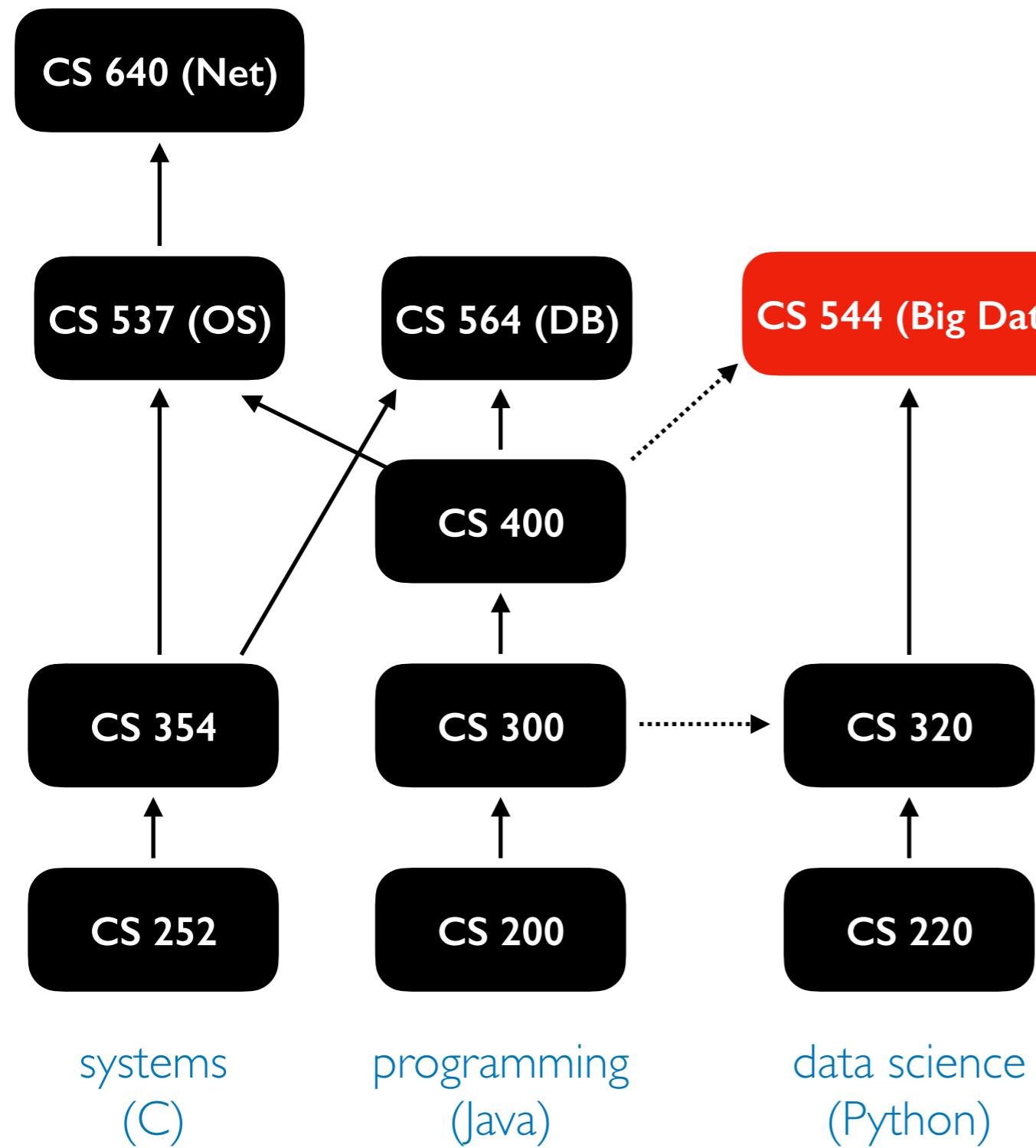
Please fill this form (**due today**):

<https://forms.gle/SaDVNUA2DvgzUMQd7>

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)

Honors Credit

CS 544 is “Honors Optional”

- Only available to undergraduates
- If you enroll in honors, you’ll attend the regular lectures and do the regular course work, but there will be some additional work
- The additional work will not influence your grade, but if the quality of work for the extras is not adequate, you’ll need to drop honors before receiving your final grade

Additional Work for Honors Enrollments

- Read 3 research papers and discuss them as a group
- Write a paper about one of those publications
- Do 2 extra programming projects

More details soon...

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 analysis code couldn't read files.

Whatever kind of programming you doing,
knowing how systems work will 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 analysis 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

What will you learn today?

Learning objectives

- recall course logistics and policies
- describe different kinds of hardware resources
- compare scale up to scale out approaches
- compare different approaches for running code on a CPU

Outline

Course Overview

- Introductions
- Main sites: tyler.caraza-harter.com, Semester Repo, Canvas
- Other tools: Email, TopHat, Piazza, GitLab, Anki

Resources

- Overview
- Compute
- Memory
- Storage
- Network

Deployment

Main Websites

1

<https://tylercaraza-harter.com/cs544/f25/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/f25/main>

- project specifications
- lecture demo code

3

Canvas

- announcements
- quizzes
- grade summaries
- zoom office hours

Outline

Course Overview

- Introductions
- Main sites: tyler.caraza-harter.com, Semester Repo, Canvas
- Other tools: Email, TopHat, Piazza, GitLab, Anki

Resources

- Overview
- Compute
- Memory
- Storage
- Network

Deployment

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 2 days
- 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

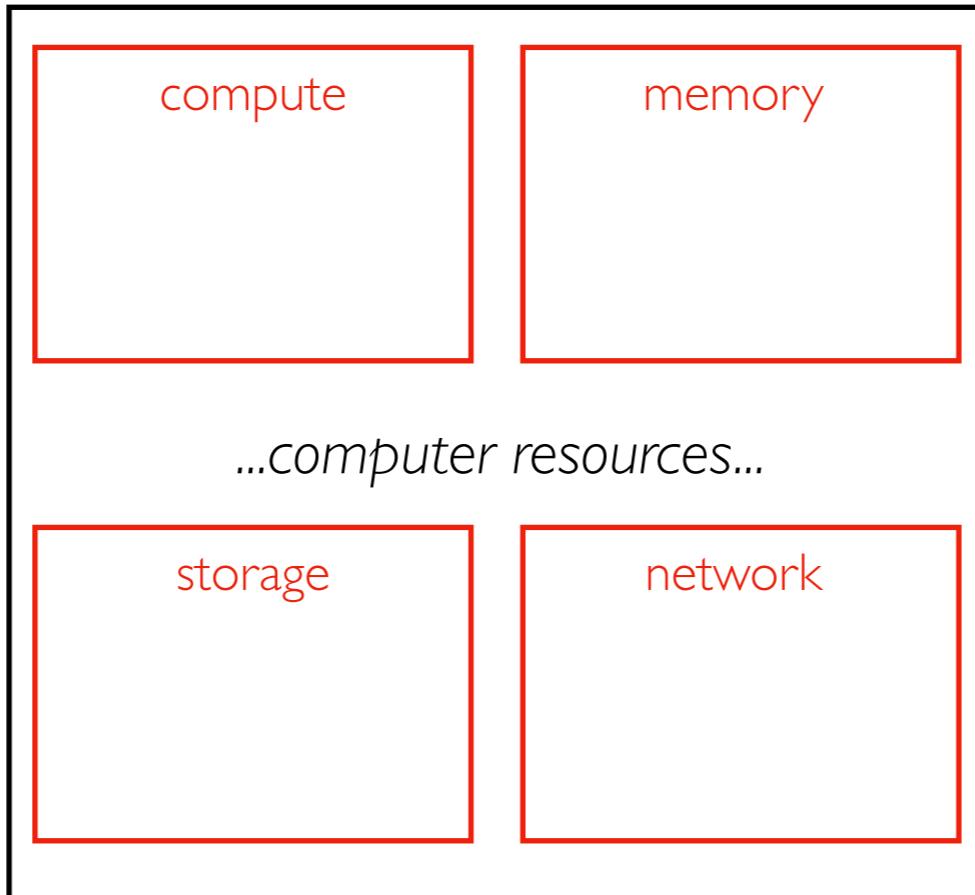
Deployment

Categories of resources

Systems: software for managing computer **resources**

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

a computer:



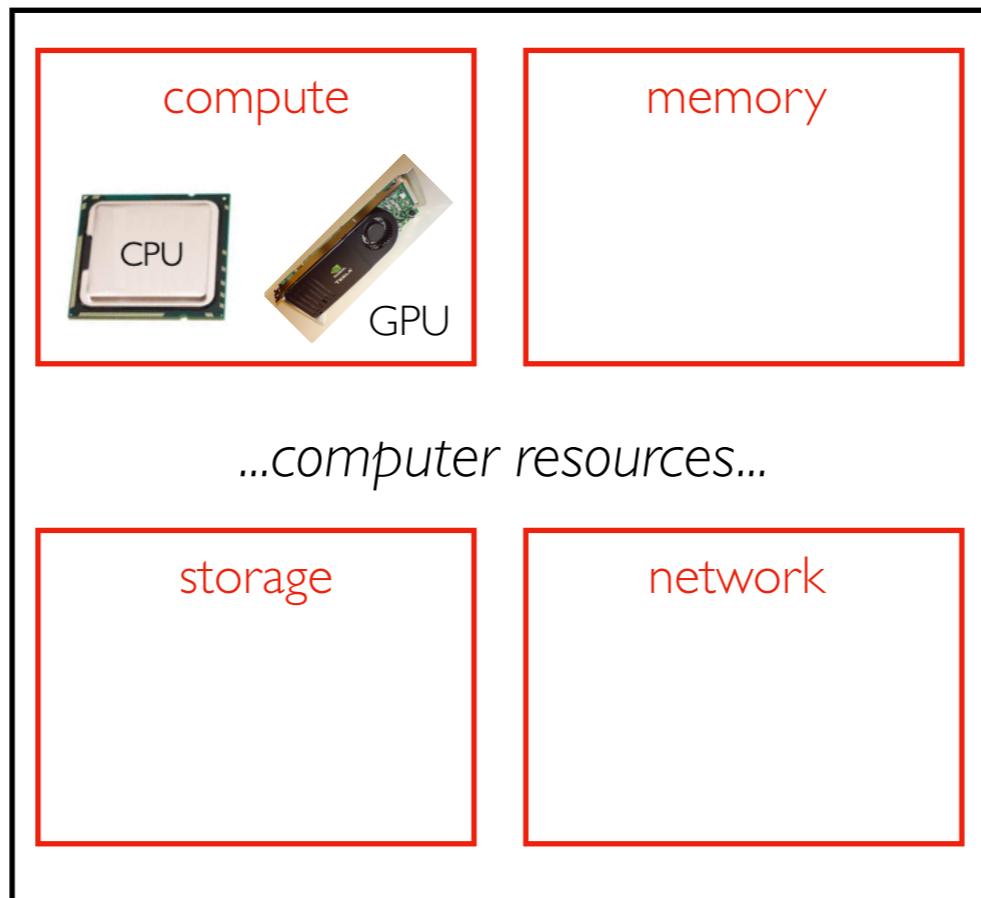
Categories of resources

Systems: software for managing computer **resources**

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

a computer:

computational resources
execute code



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

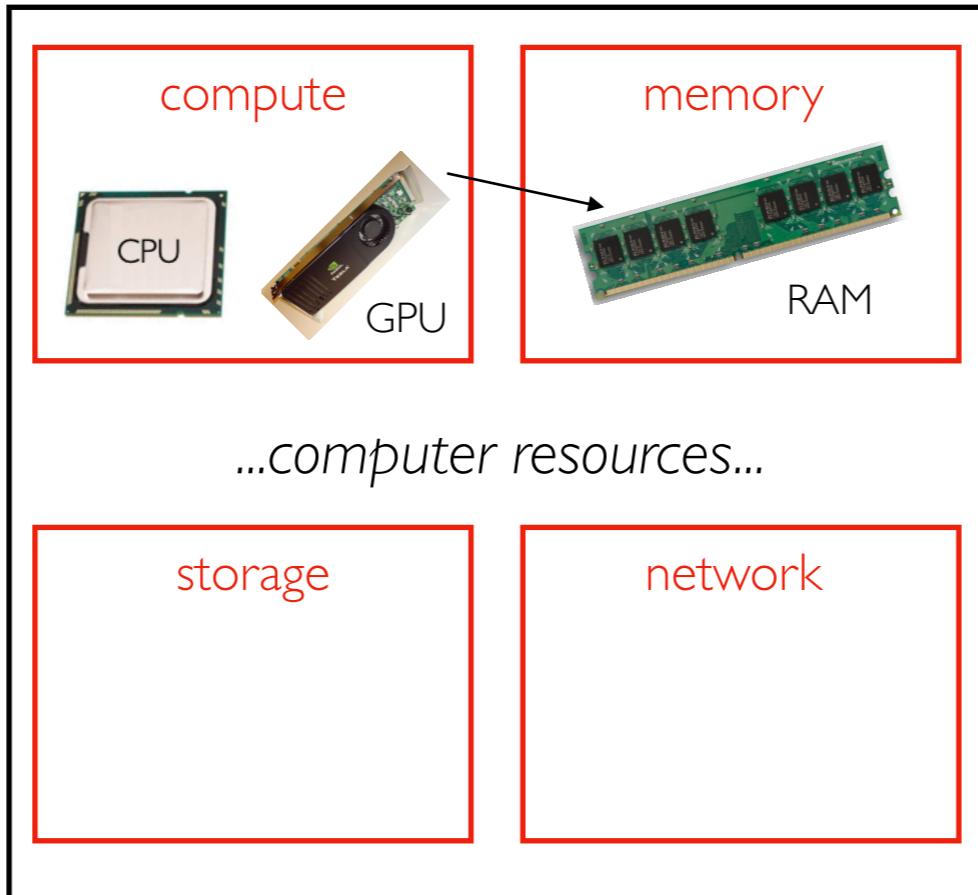
Categories of resources

Systems: software for managing computer **resources**

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

computational resources
execute code

a computer:
GPUs have their
own built-in RAM



memory holds data
for active usage

random-access memory (RAM)

Categories of resources

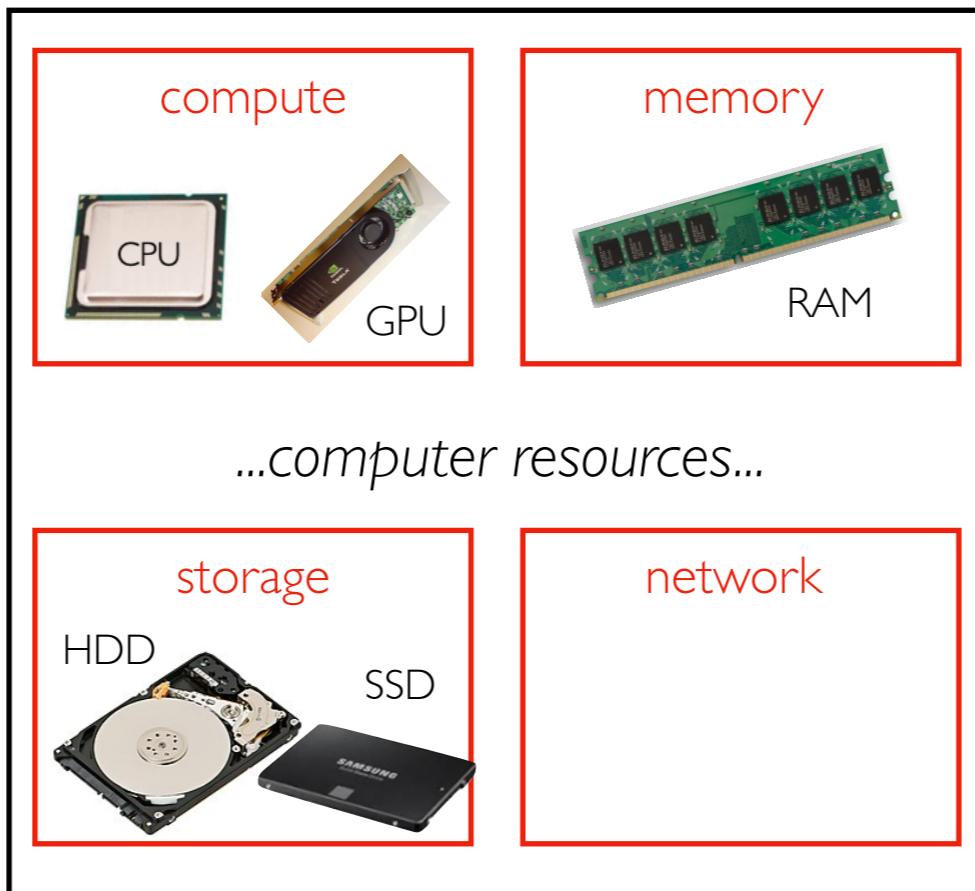
Systems: software for managing computer **resources**

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

a computer:

computational resources
execute code

storage holds
long-term data



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

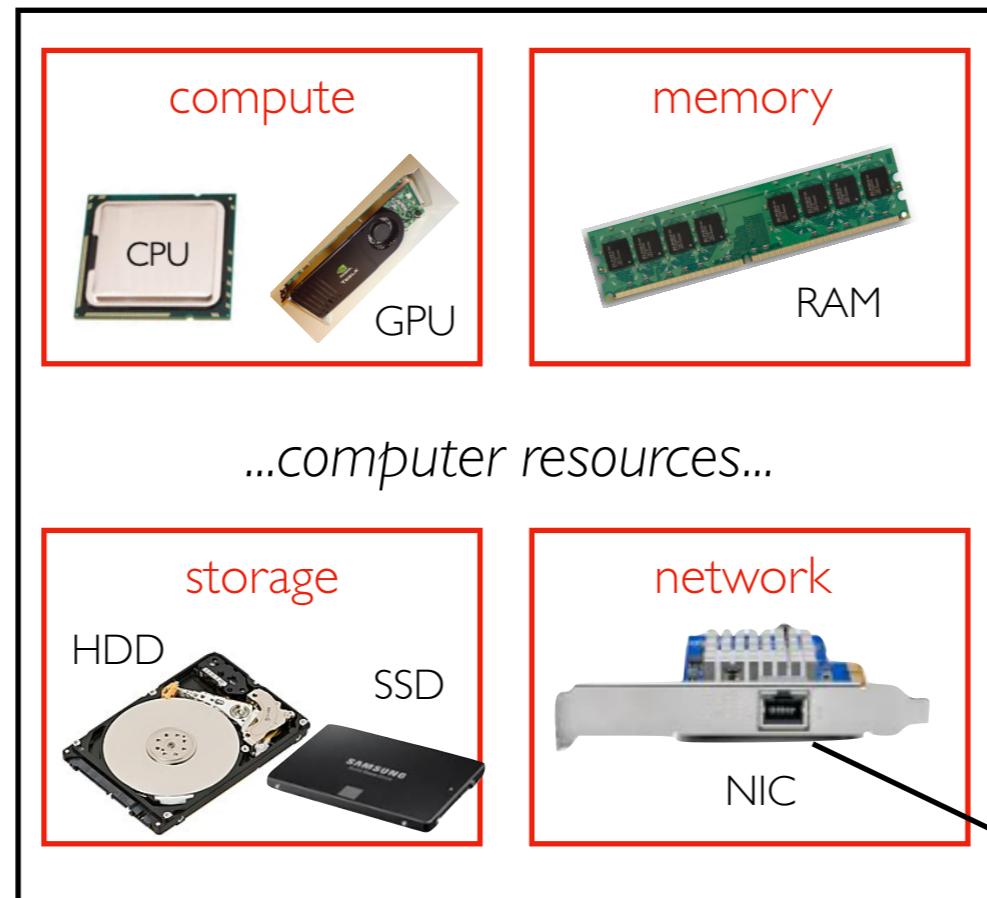
memory holds data
for active usage

Categories of resources

Systems: software for managing computer **resources**

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

a computer:



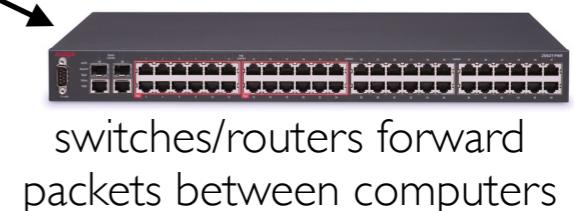
computational resources
execute code

storage holds
long-term data

memory holds data
for active usage

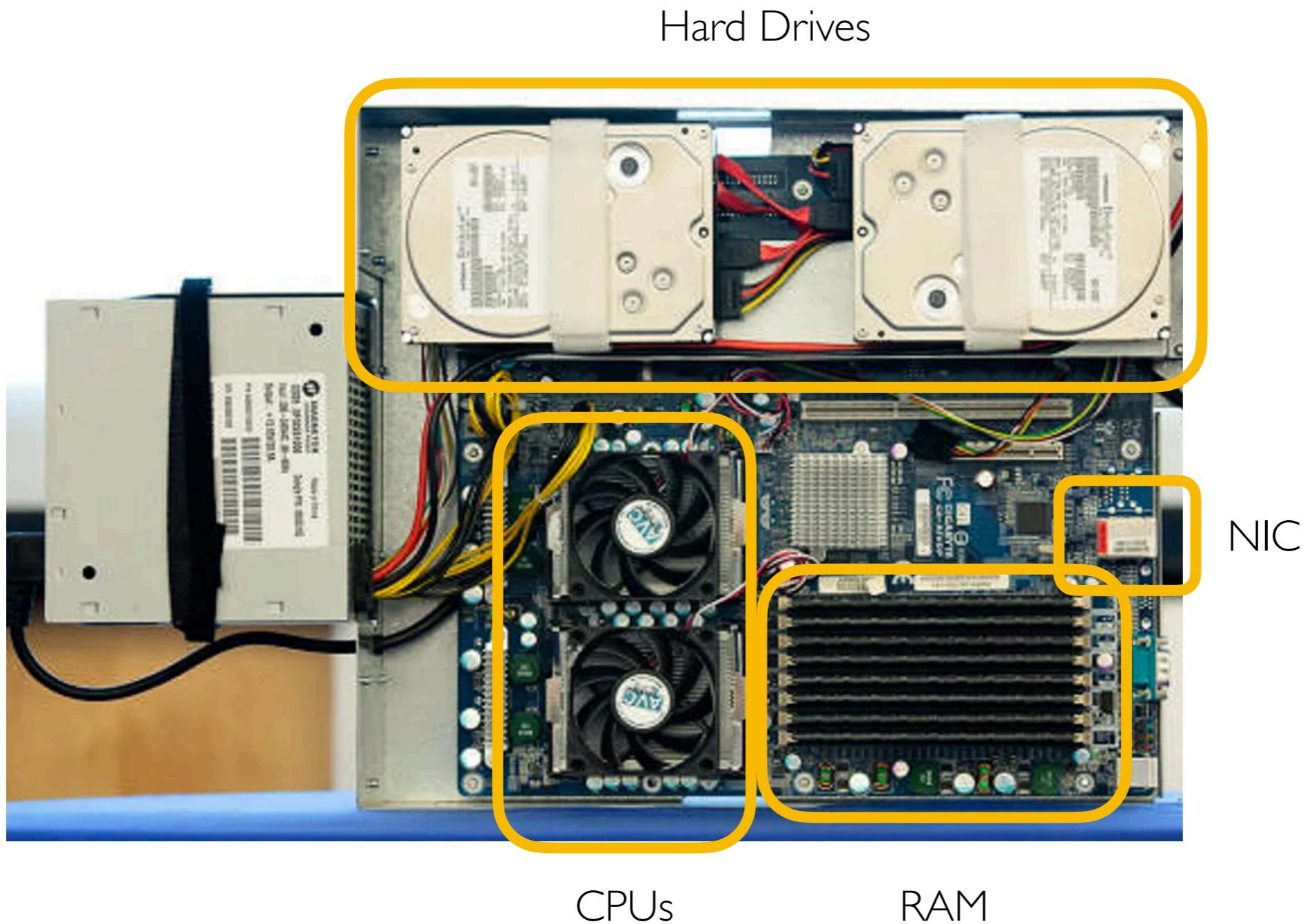
network provides
communication between
computers

network interface card (NIC)



switches/routers forward
packets between computers

A real server



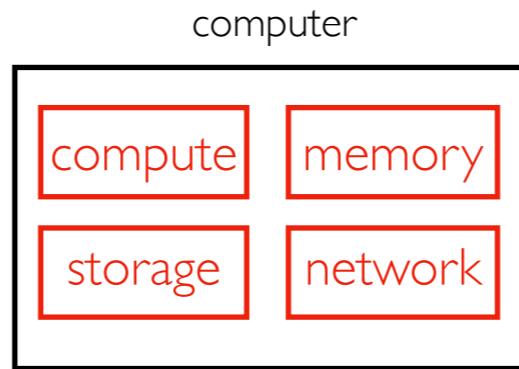
Big Data

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



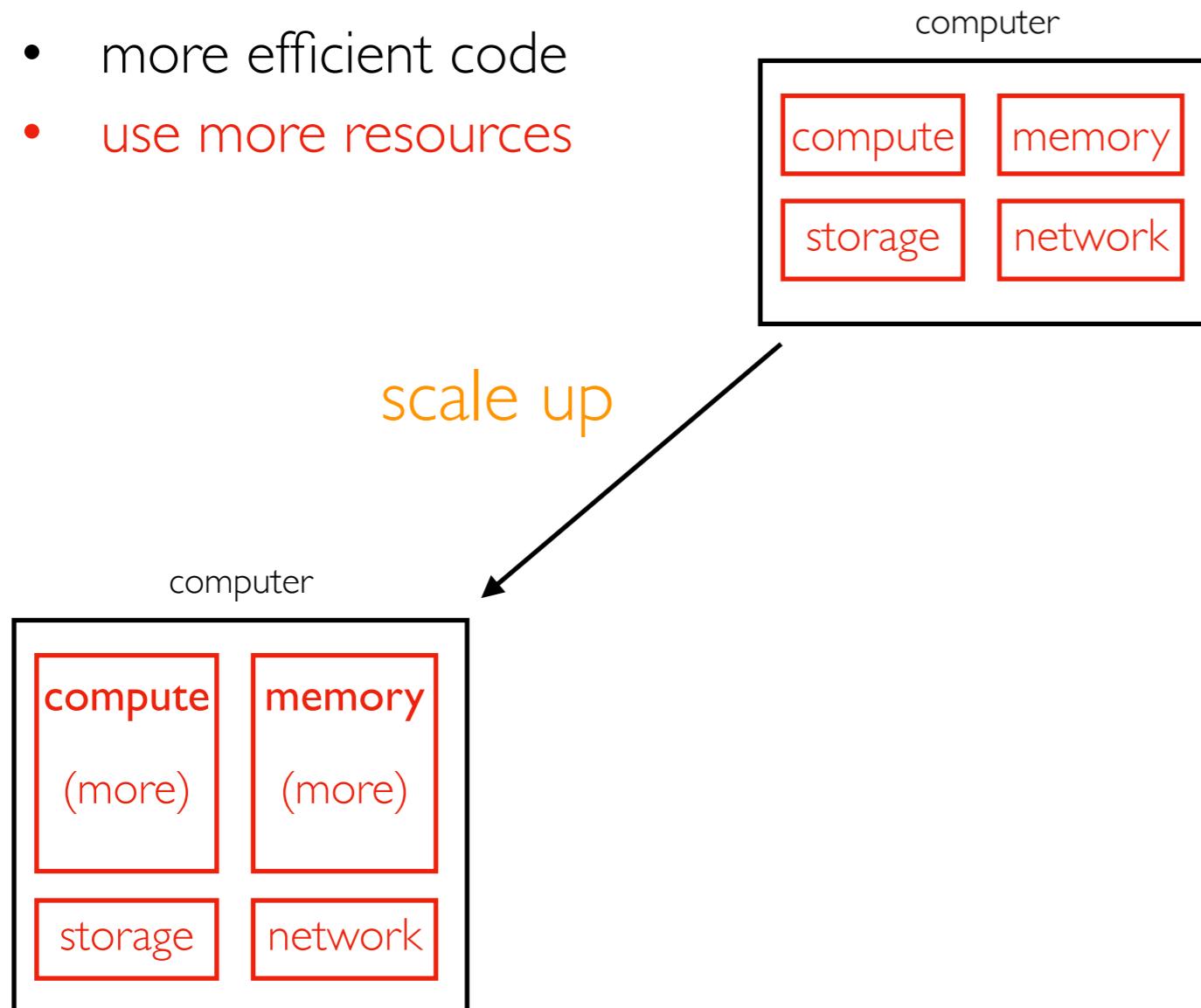
Big Data

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



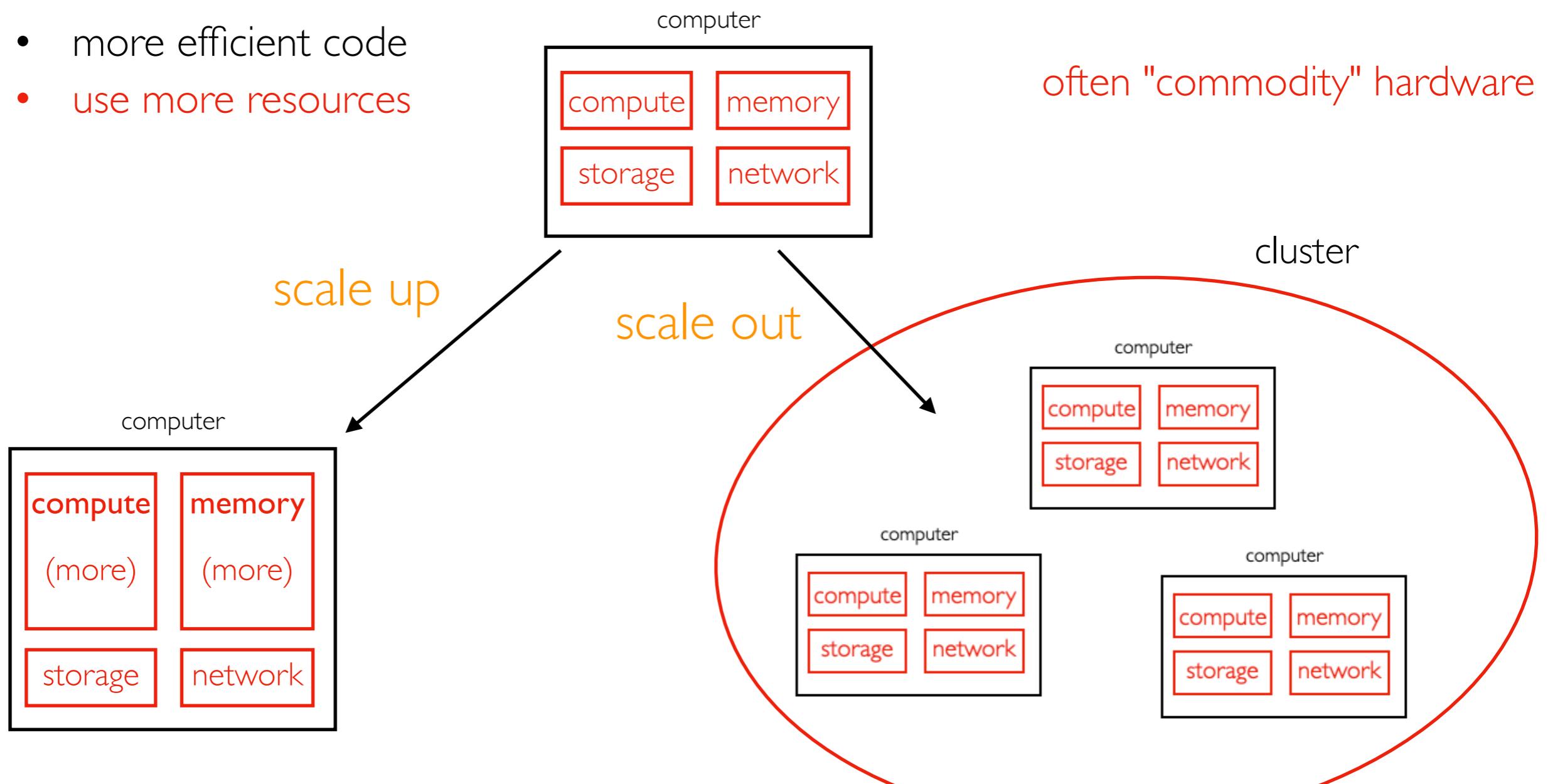
Big Data

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



Outline

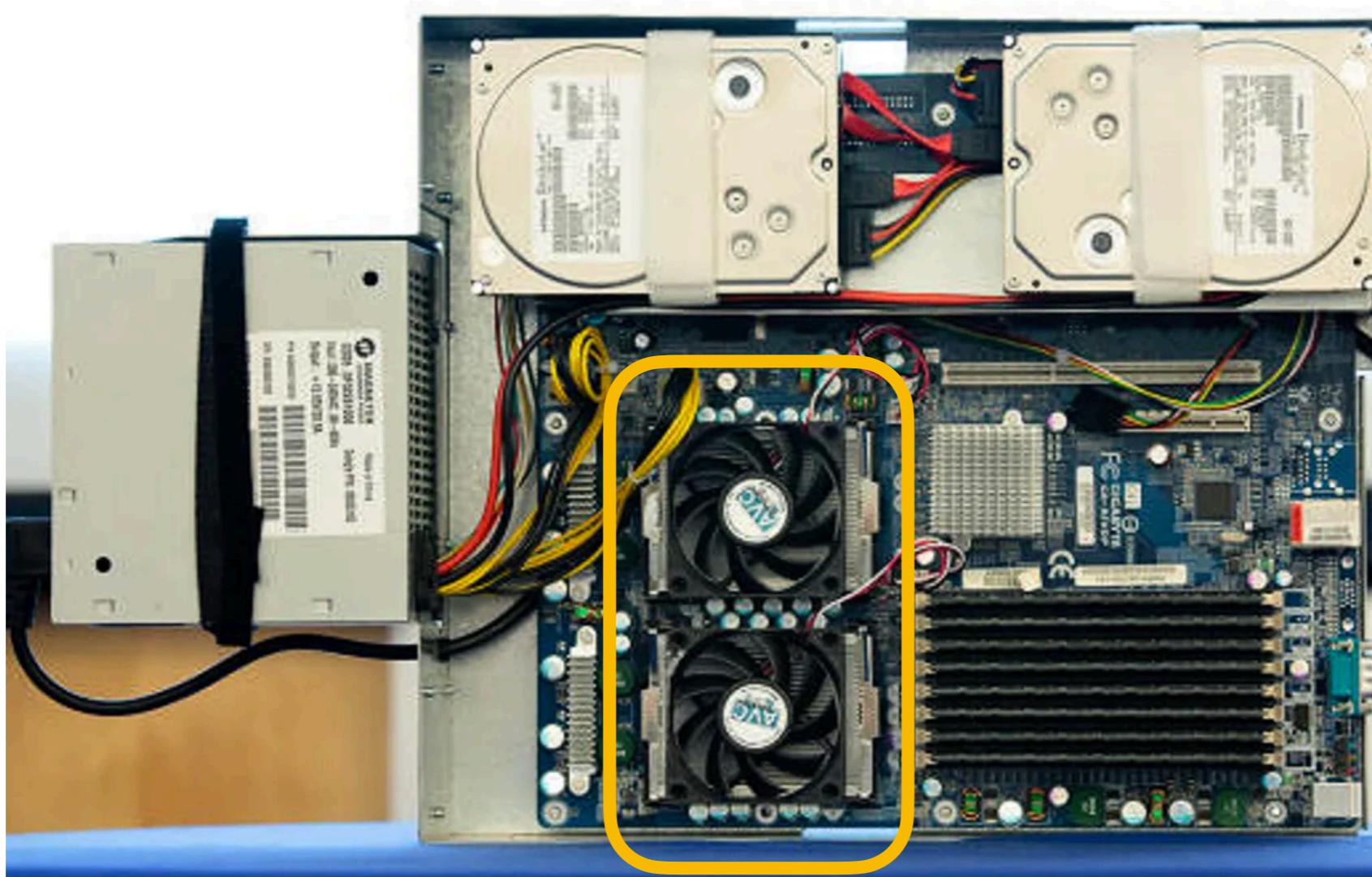
Course Overview

Resources

- Overview
- Compute
- Memory
- Storage
- Network

Deployment

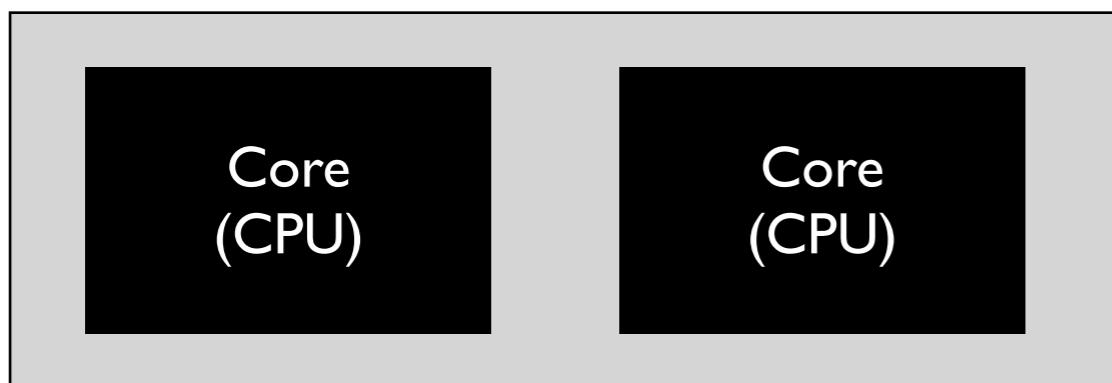
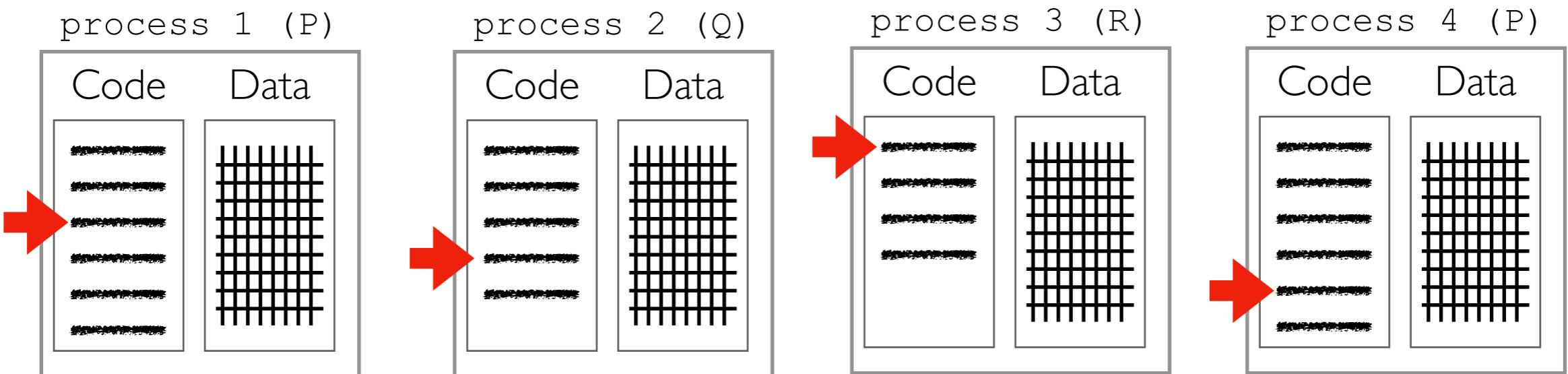
Compute



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

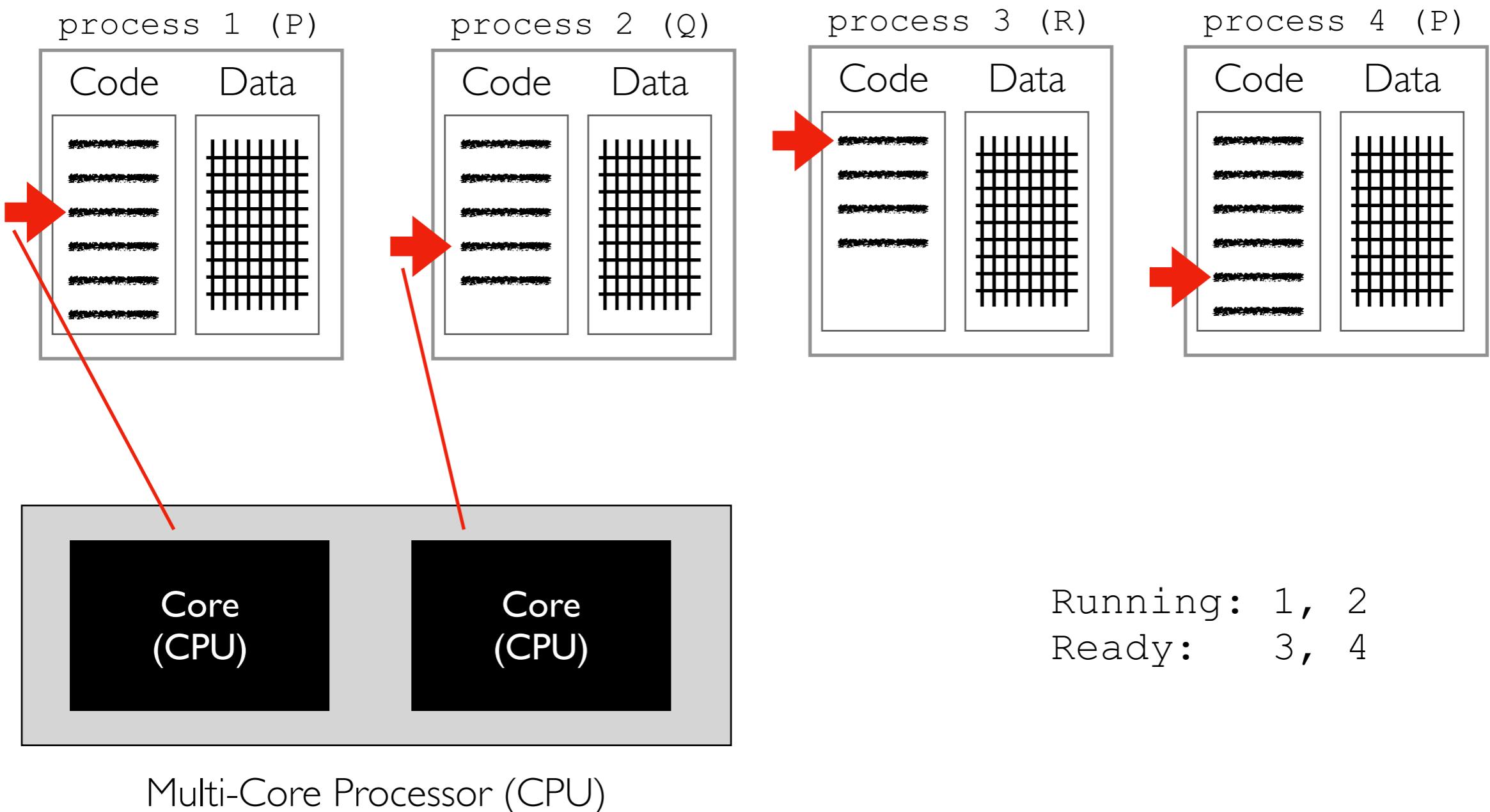
How do cores run machine code?

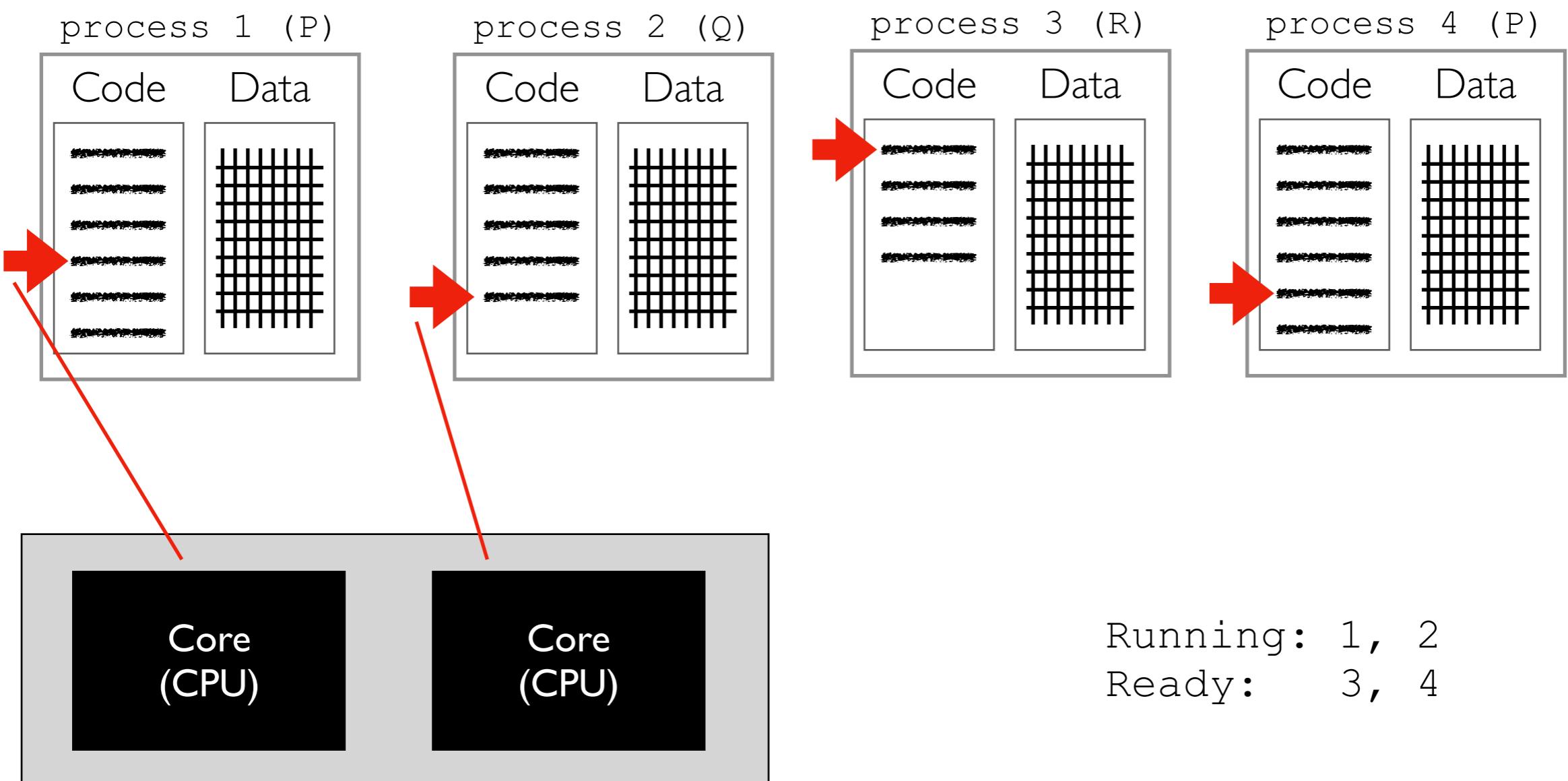
the operating system "schedules" tasks on cores
(decides when they get to run)



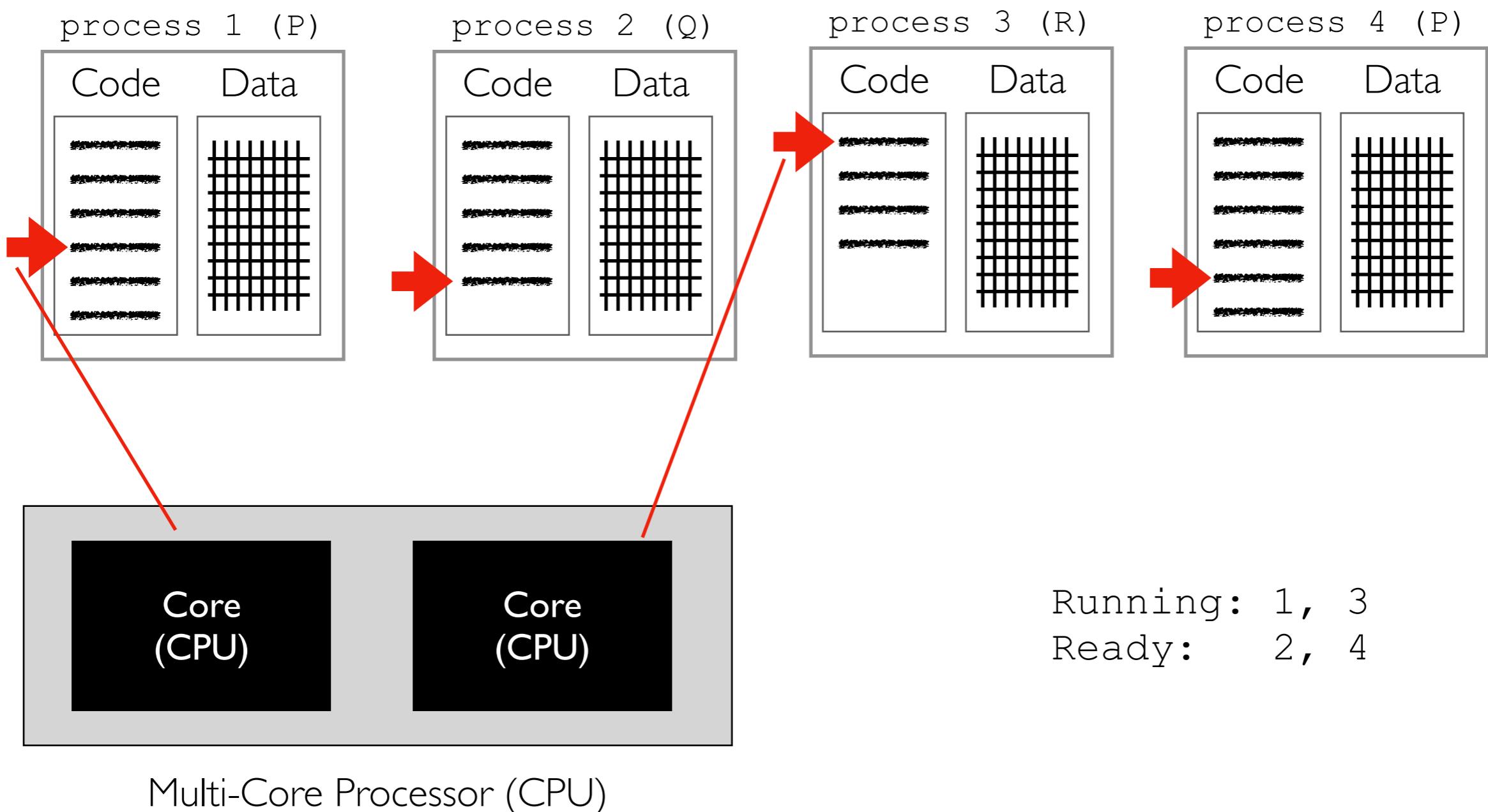
Multi-Core Processor (CPU)

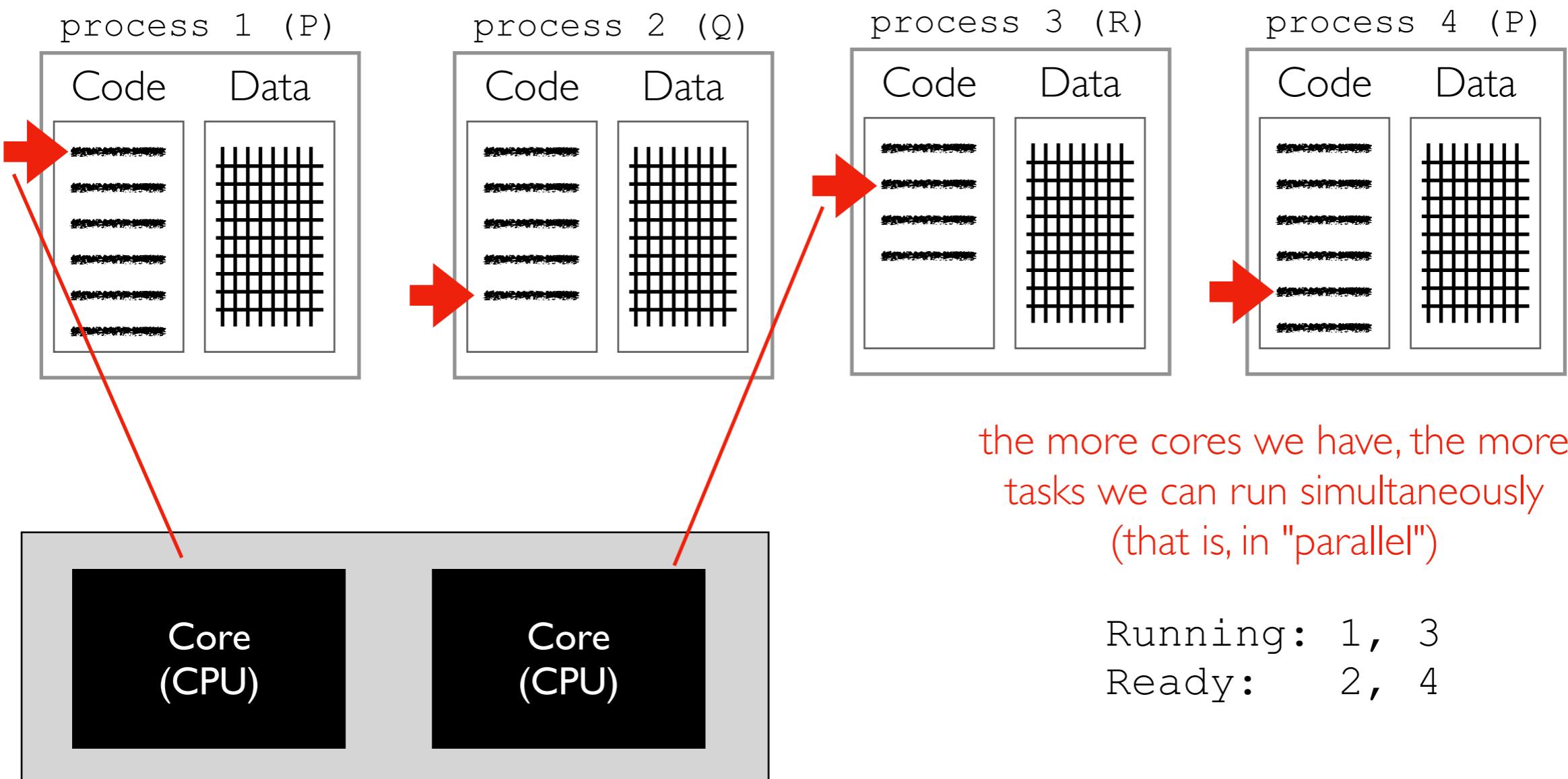
a core can execute instructions for one (or in some cases two) tasks at a time





Multi-Core Processor (CPU)

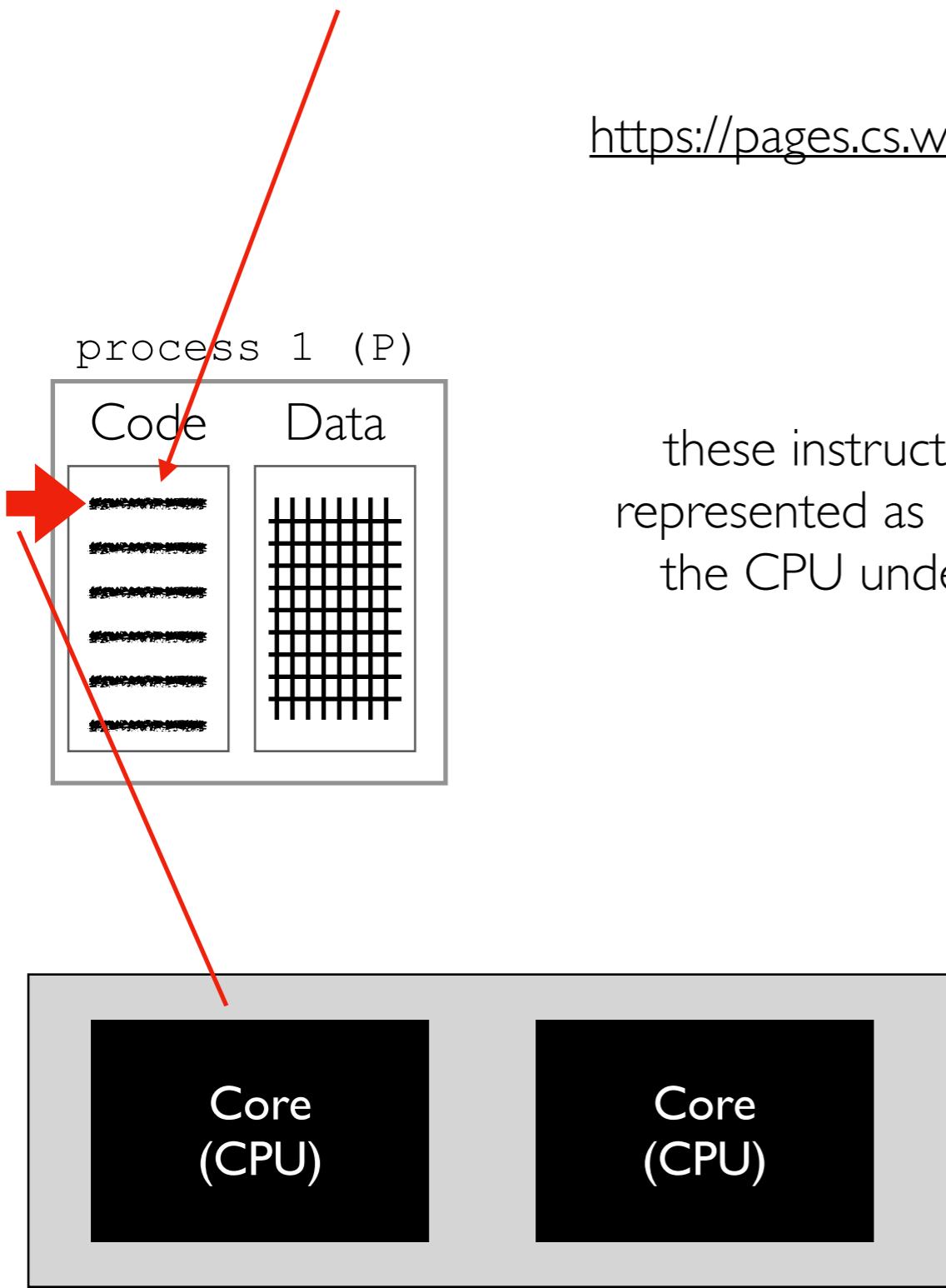




the more cores we have, the more tasks we can run simultaneously
(that is, in "parallel")

Running: 1, 3
Ready: 2, 4

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



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

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

```
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

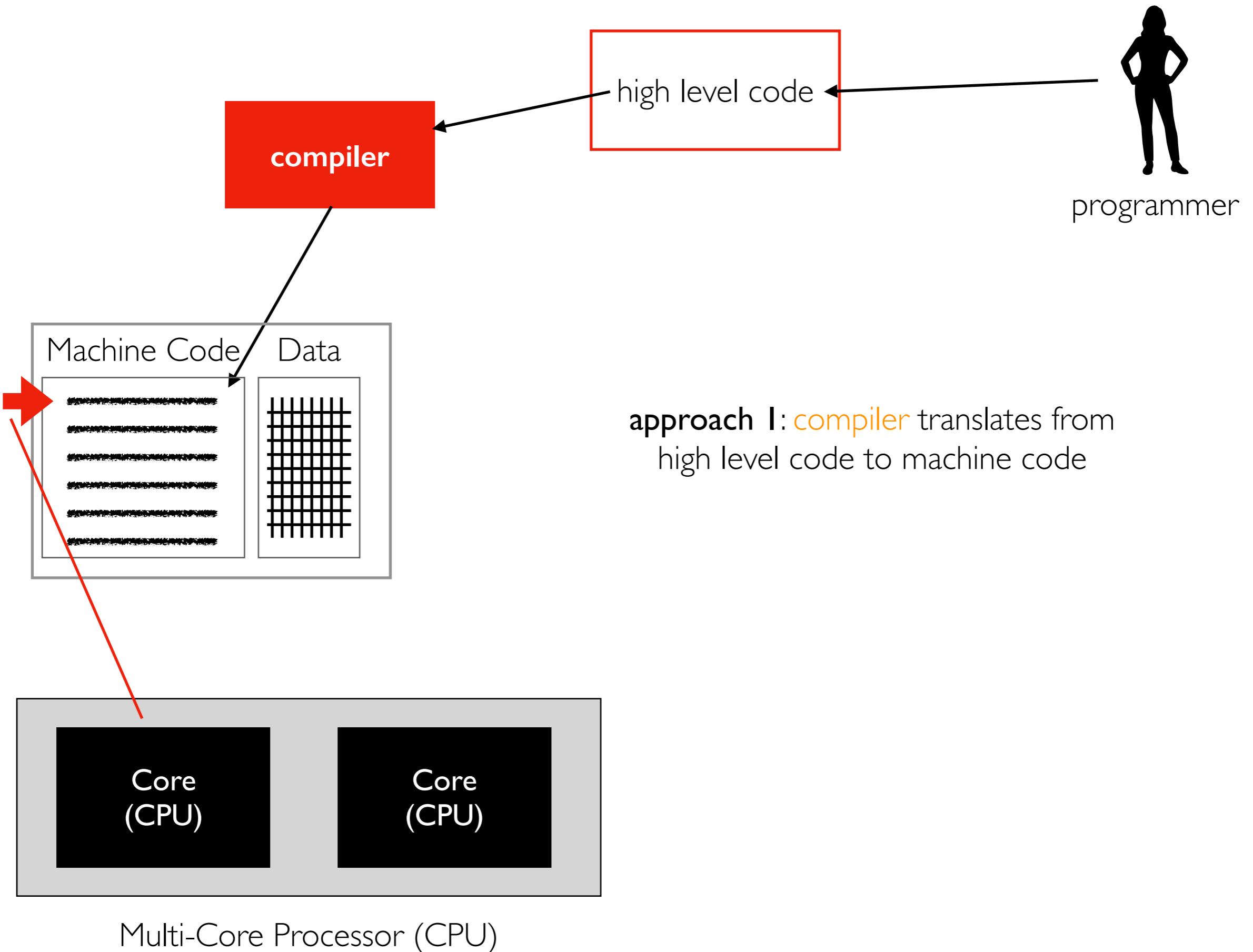
arithmetic ops set CCs implicitly
cf=1 if carry out from msb
zf=1 if dst==0,
sf=1 if dst < 0 (signed)
of=1 if two's complement
(signed) under/overflow
```

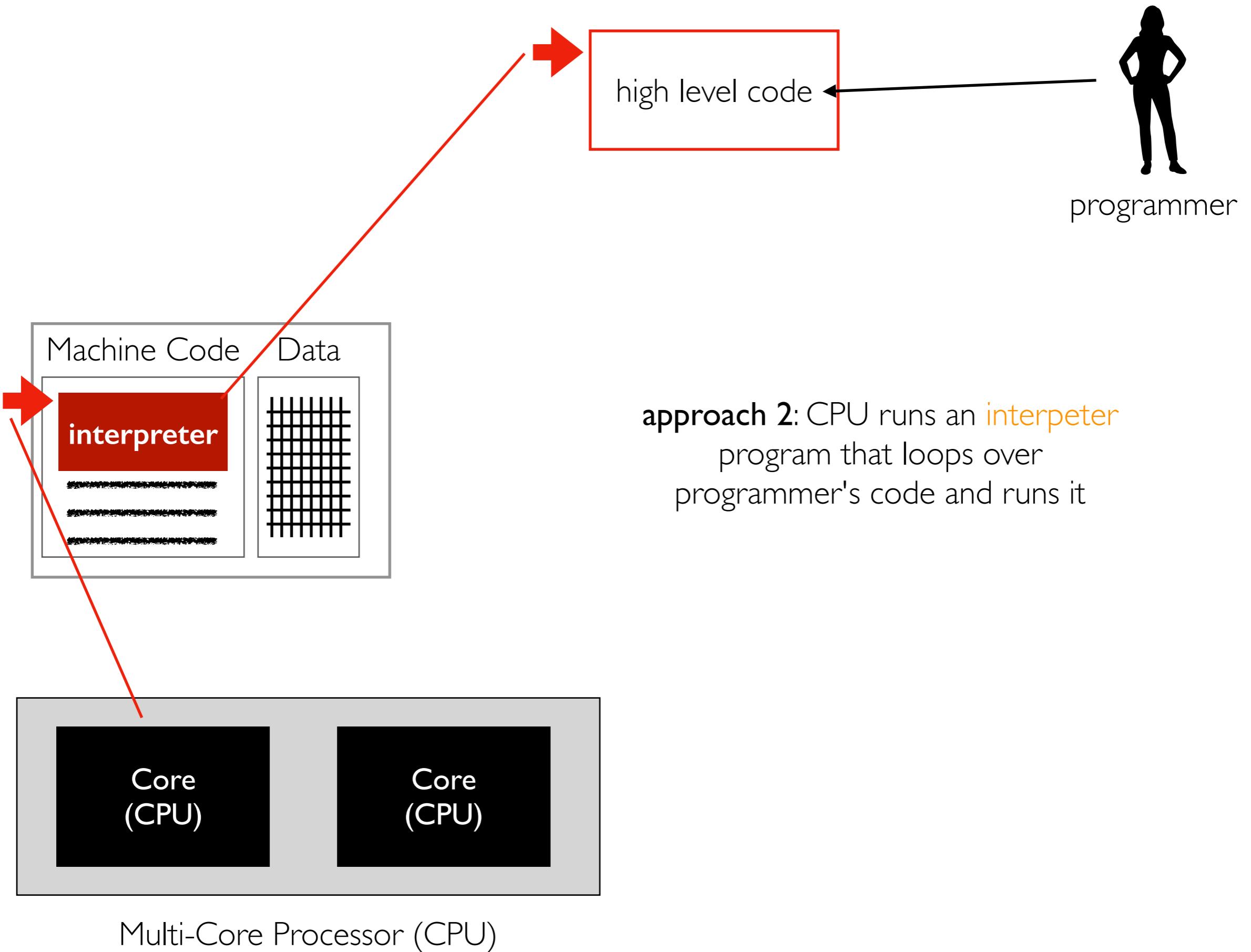
how do we bridge the gap between "high level"
code (Python/Java/etc) and machine code?

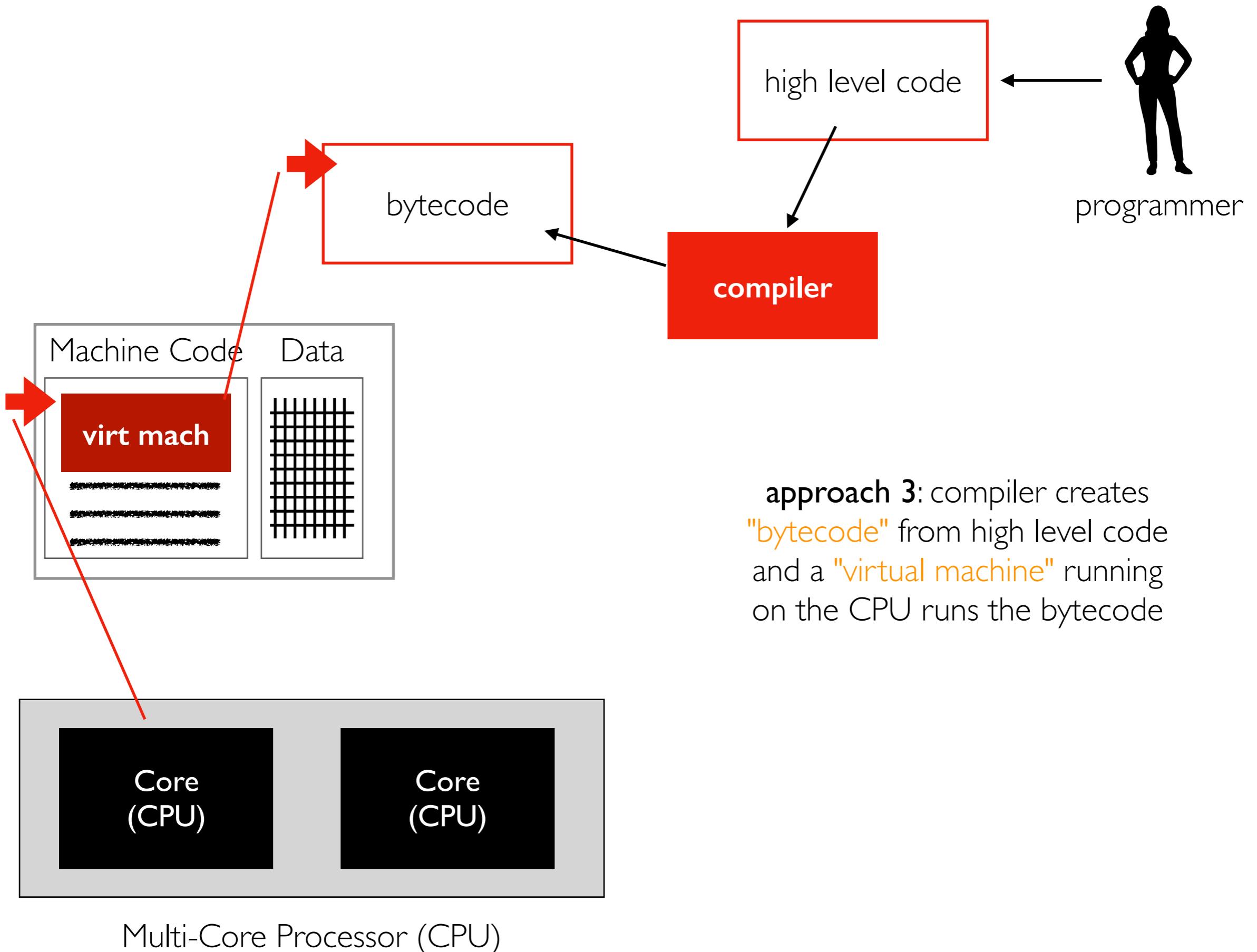
Multi-Core Processor (CPU)

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

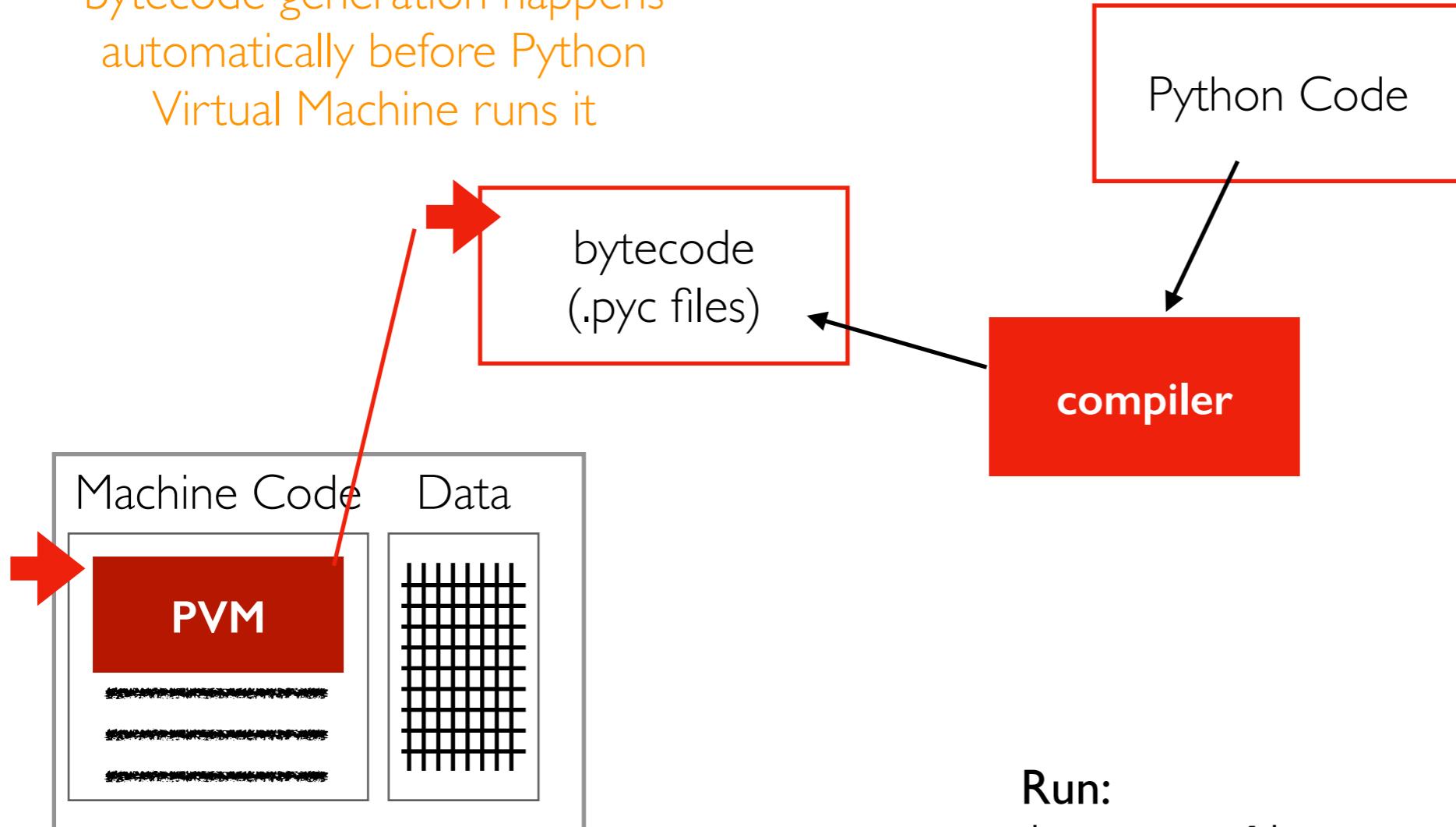
Note: we'll primarily write Python this semester, but it helps to explore this in general to understand how systems like Spark work (which is written in Scala and uses the Java Virtual Machine)





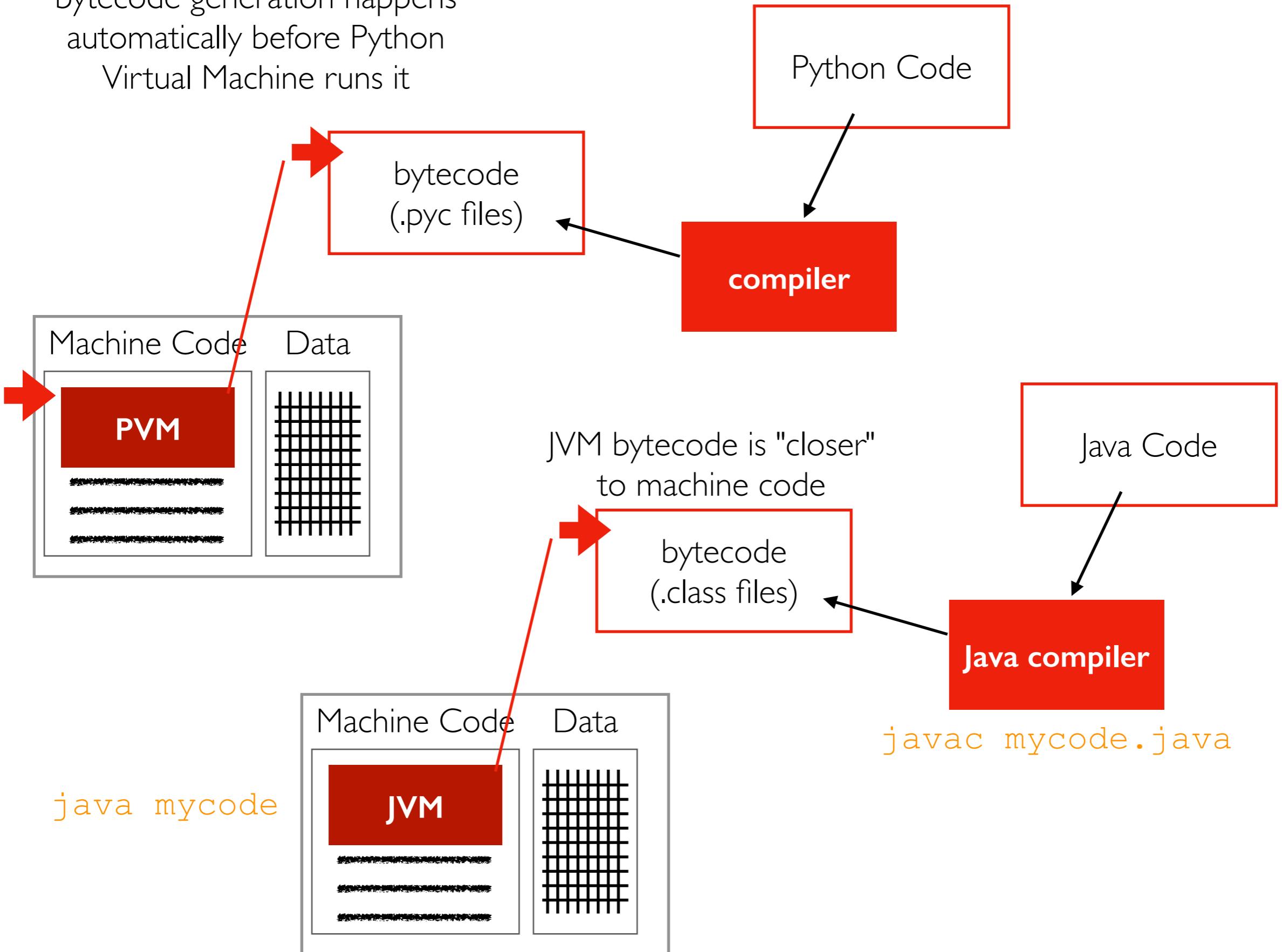


when you run "python3 ..."
bytecode generation happens
automatically before Python
Virtual Machine runs it

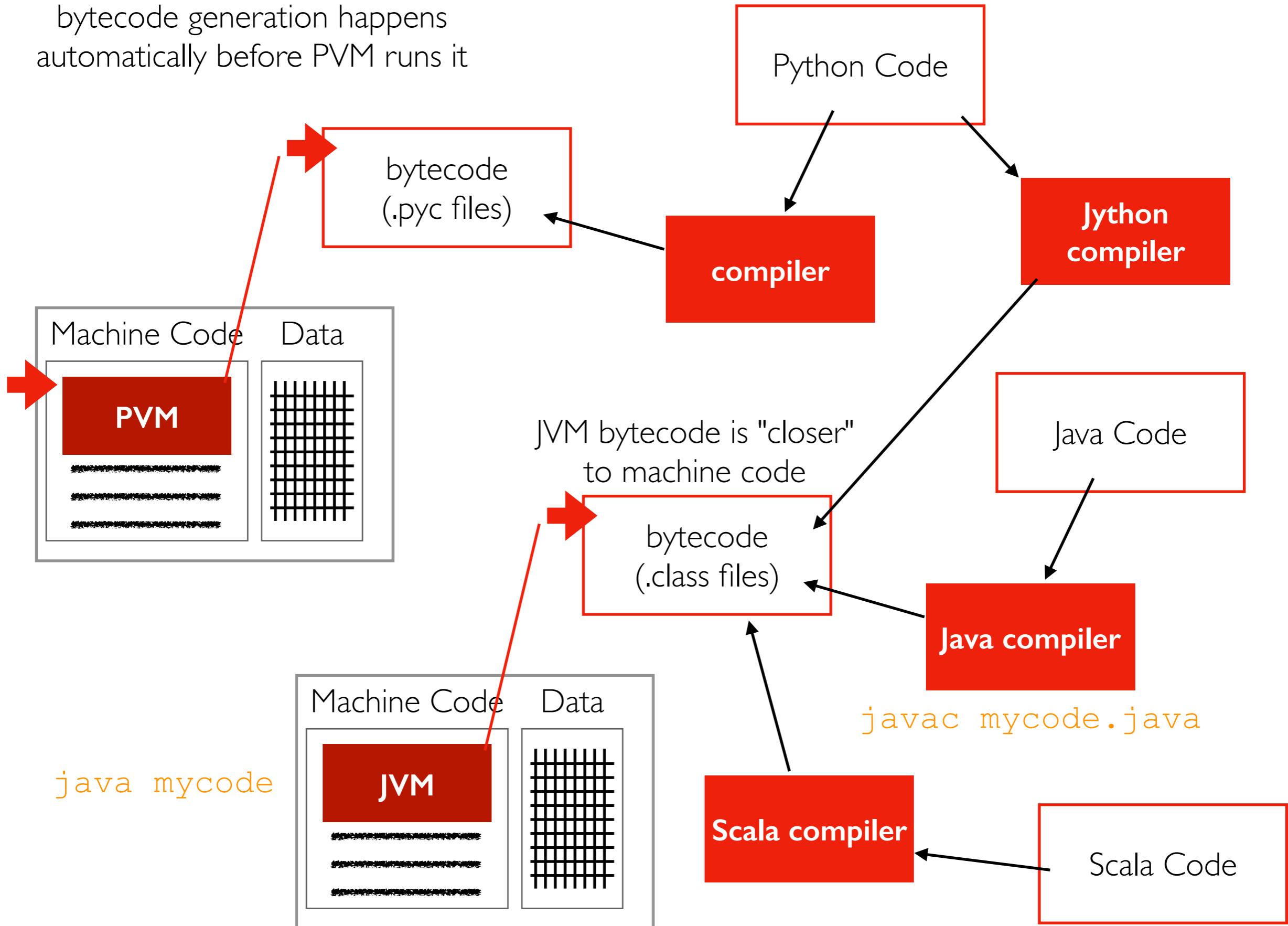


Run:
import dis
dis.dis("z = x + y * 2")

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

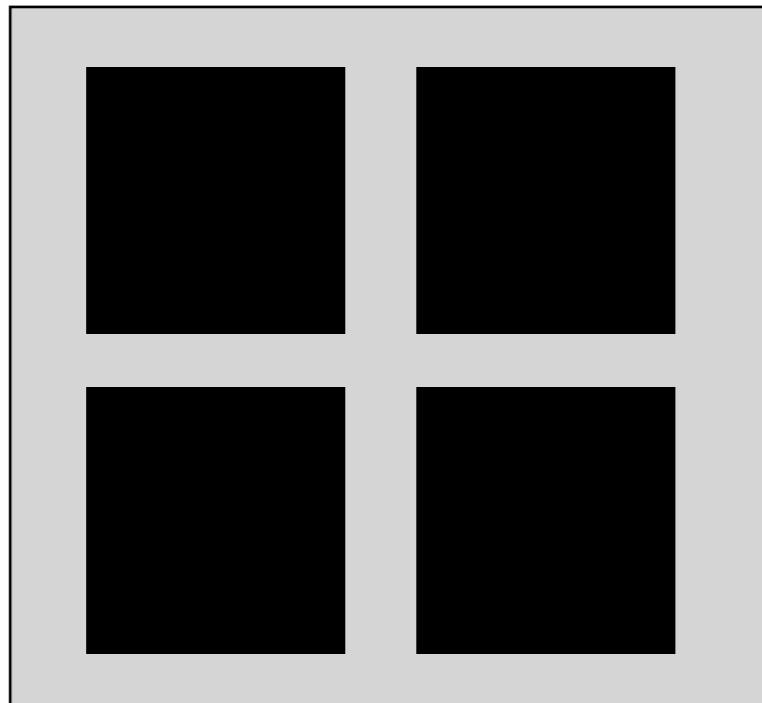
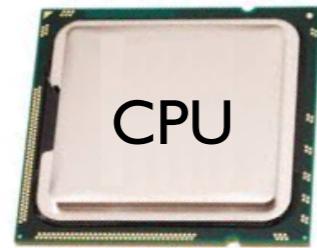


when you run "python3 ..."
 bytecode generation happens
 automatically before PVM runs it

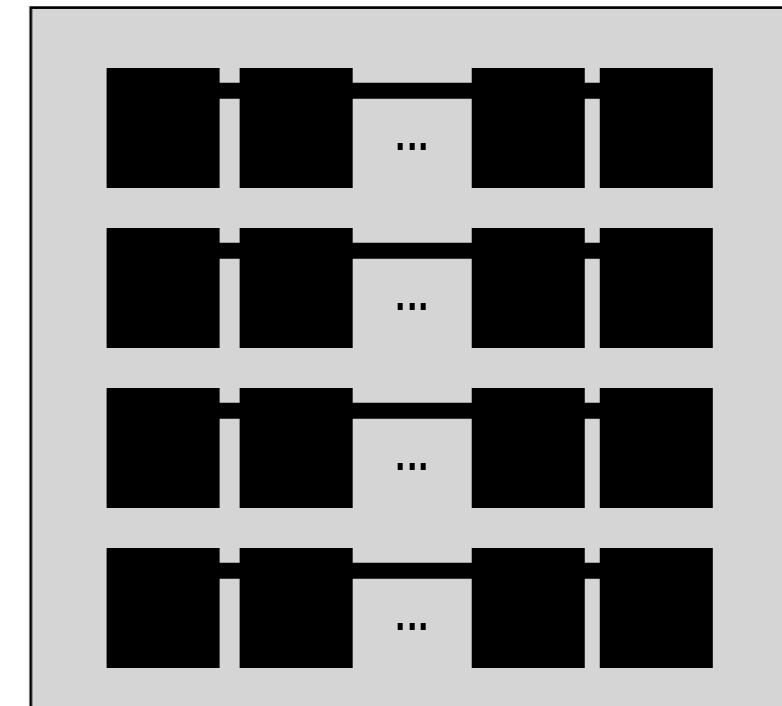


What are alternatives to CPUs for compute?

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



few cores that are fast,
flexible, independent

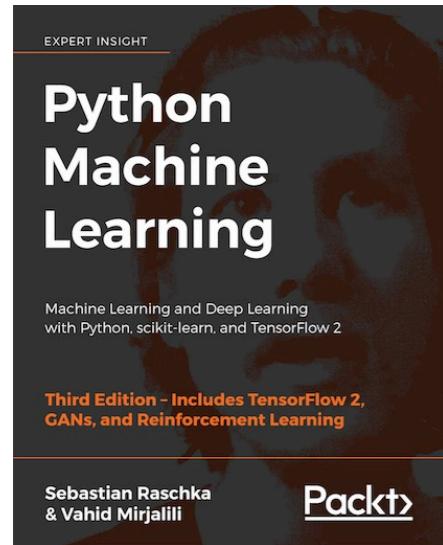


many cores that are slow,
float-optimized, coordinated

TOP HAT

GPU vs. CPU: Cost Comparison

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

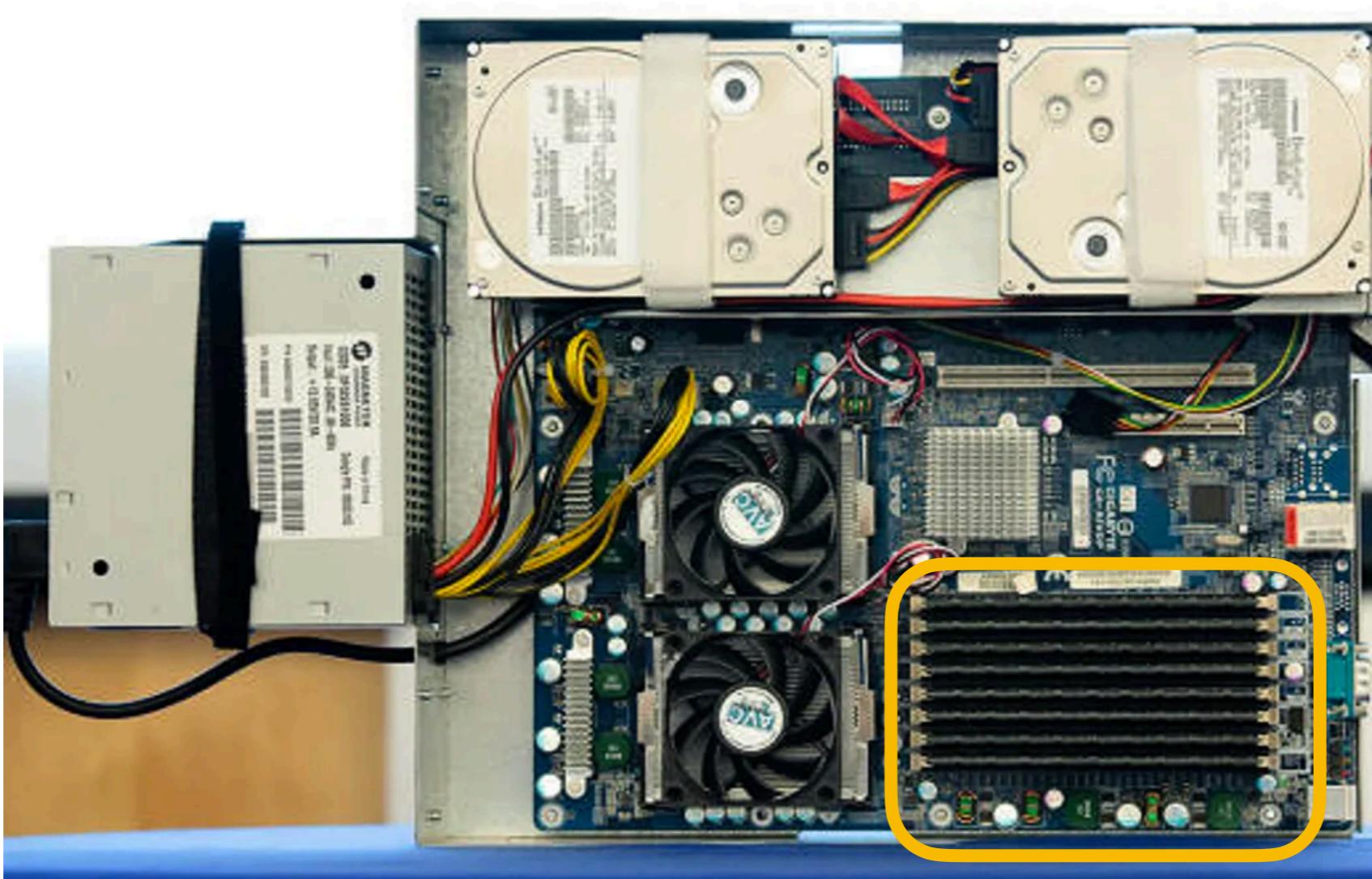
Course Overview

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**:



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	colors
011	D	3	-1	images
100	E	4	0	floats
101	F	5	1	etc.
110	G	6	2	
111	H	7	3	

Bytes

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

RAM is "byte addressable"

- each byte of data has its 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 than 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**)



Solid State Disks (**SSDs**)



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

- 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 than 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)

Price/Terabyte of SSD & HDD are Converging Rapidly

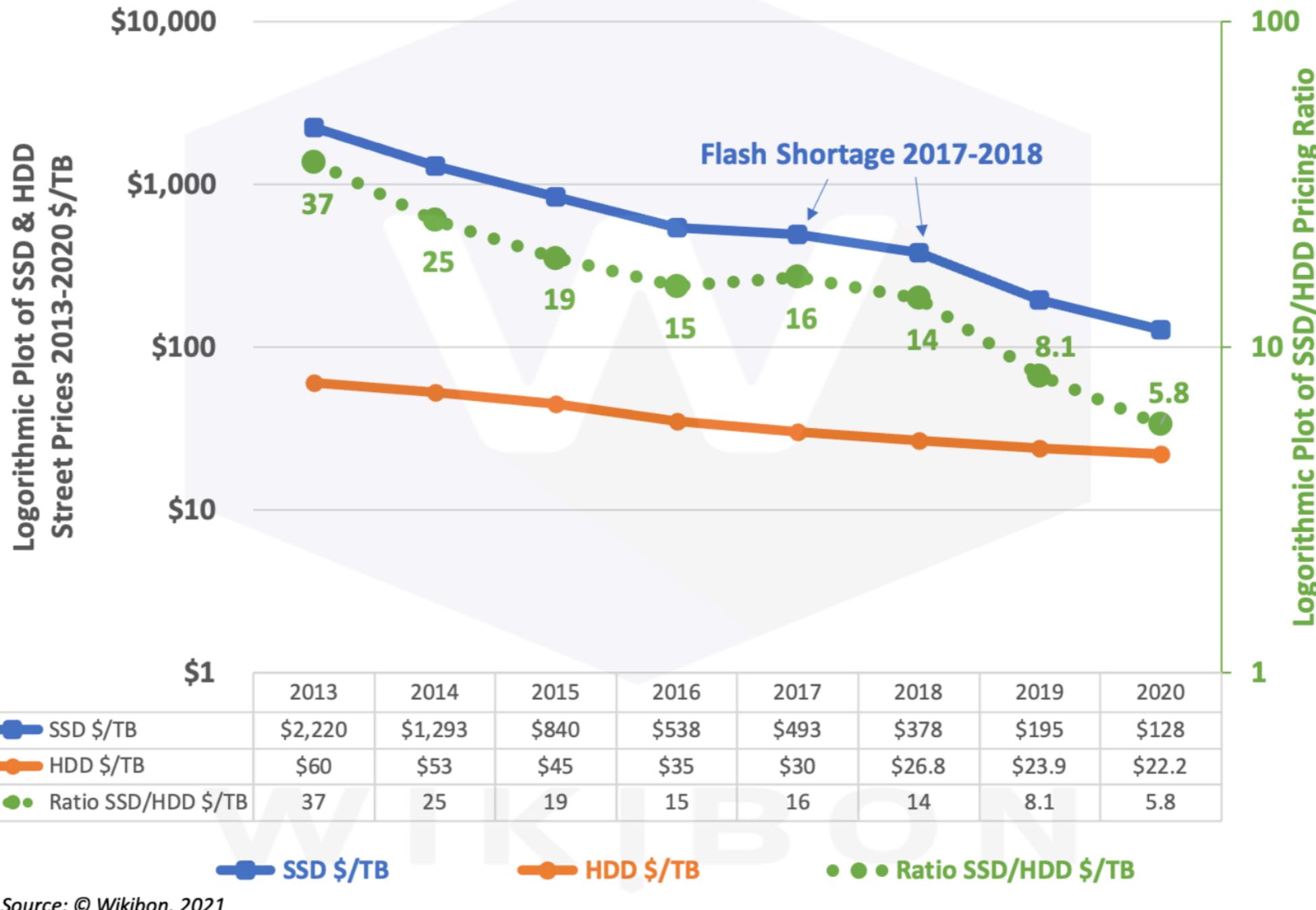
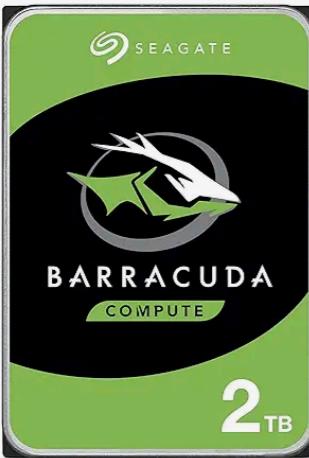


Figure 3 – Flash & HDD Pricing 2013 – 2020

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

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
\$67¹⁷
✓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

\$119⁹⁹ 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

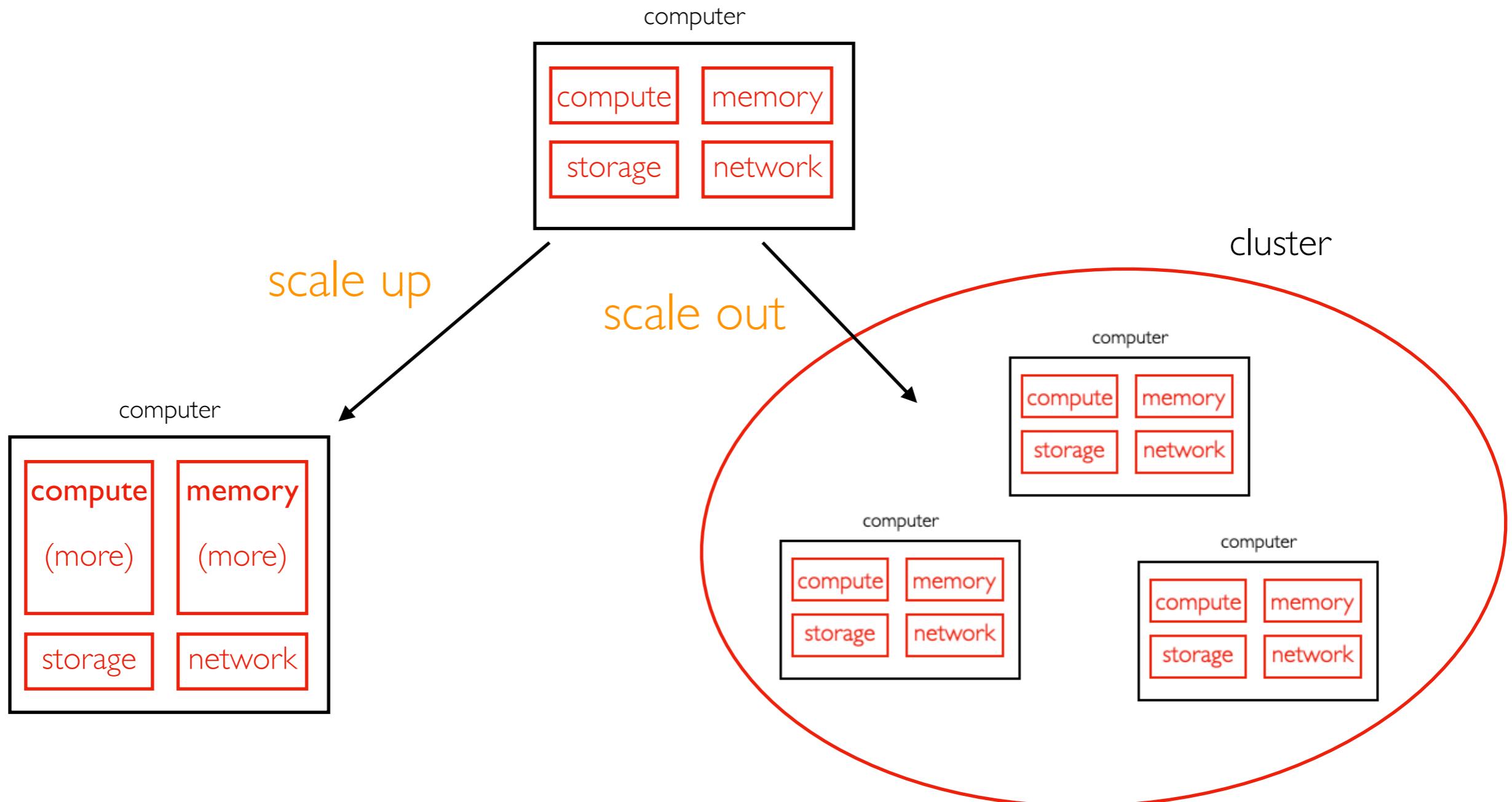
Resources

- 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>

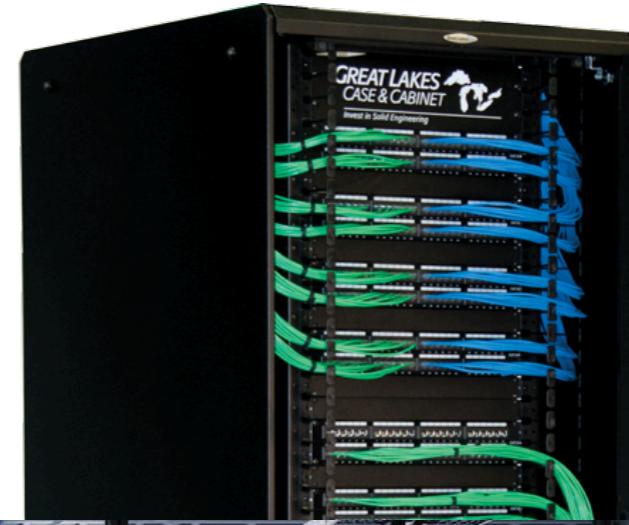
https://buy.hpe.com/us/en/servers/proliant-dl-servers/proliant-dl10-servers/proliant-dl20-server/hpe-proliant-dl20-gen10-plus-e-2336-2-9ghz-6-core-1p-16gb-u-4sff-500w-rps-server/p/p41115-b21?ef_id=Cj0KCQiAt66eBhCnARlsAKf3ZNFjsg49UV6Zm33R7lkRqi-XOd_JECmdyqNMAm2CKLSm_F-z6jTYDTQaAgMTEALw_wcB&s_kwcid=AL!13472!3!33!628972784!!g!318267171339!!1707918369!67076417419&gclid=Cj0KCQiAt66eBhCnARlsAKf3ZNFjsg49UV6Zm33R7lkRqi-XOd_JECmdyqNMAm2CKLSm_F-z6jTYDTQaAgMTEALw_wcB

https://www.server-rack-online.com/g1910ent-4048sss.html?utm_medium=shoppingengine&utm_source=googlebase&utm_source=google&utm_medium=cpc&adpos=&scid=scplpg1910ent-4048sss&sc_intid=g1910ent-4048sss&gclid=Cj0KCQiAt66eBhCnARlsAKf3ZNEMYINPAA0RFGQIF0DsieCM6oh7i3kuJvJlpnmJAloPaj3RWT11QMAaAqRnEALw_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/p4115-b21?ef_id=Cj0KCQiAt66eBhCnARlsAKf3ZNFjsg49UV6Zm33R7lkRqi-XOd_JECmdyqNMAm2CKLSm_F-z6jTYDTQaAgMTEALw_wcB&s_kwcid=AL!13472!3!33!628972784!!g!318267171339!!1707918369!67076417419&gclid=Cj0KCQiAt66eBhCnARlsAKf3ZNFjsg49UV6Zm33R7lkRqi-XOd_JECmdyqNMAm2CKLSm_F-z6jTYDTQaAgMTEALw_wcB

https://www.server-rack-online.com/g1910ent-4048sss.html?utm_medium=shoppingengine&utm_source=googlebase&utm_source=google&utm_medium=cpc&adpos=&scid=scplpg1910ent-4048sss&sc_intid=g1910ent-4048sss&gclid=Cj0KCQiAt66eBhCnARlsAKf3ZNEMYINPAA0RFGQIF0DsieCM6oh7i3kuJvJlpnmJAIOpAJ3RWT11QMAaAqRnEALw_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 |
#          +-----+      +-----+
#          "           "
```

Metrics

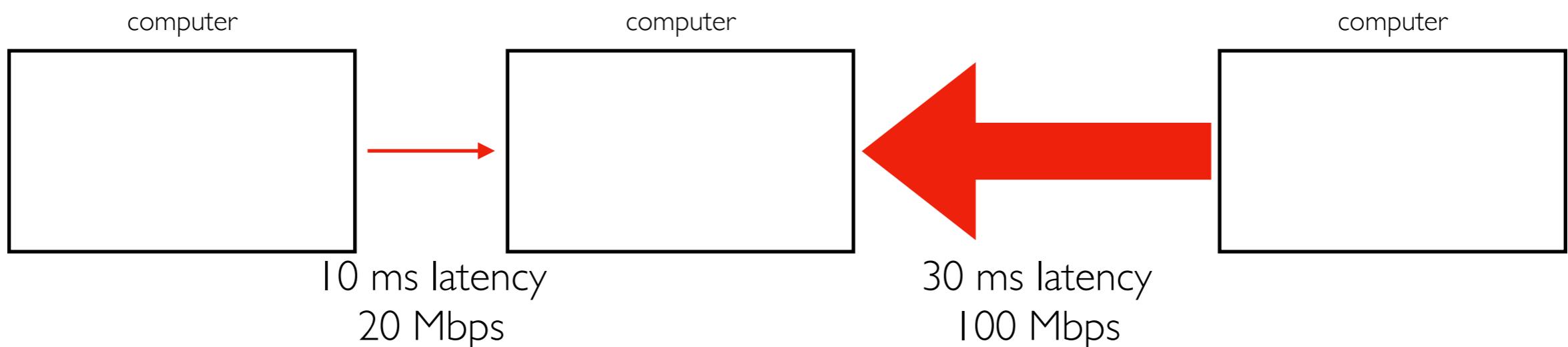
TOP HAT

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

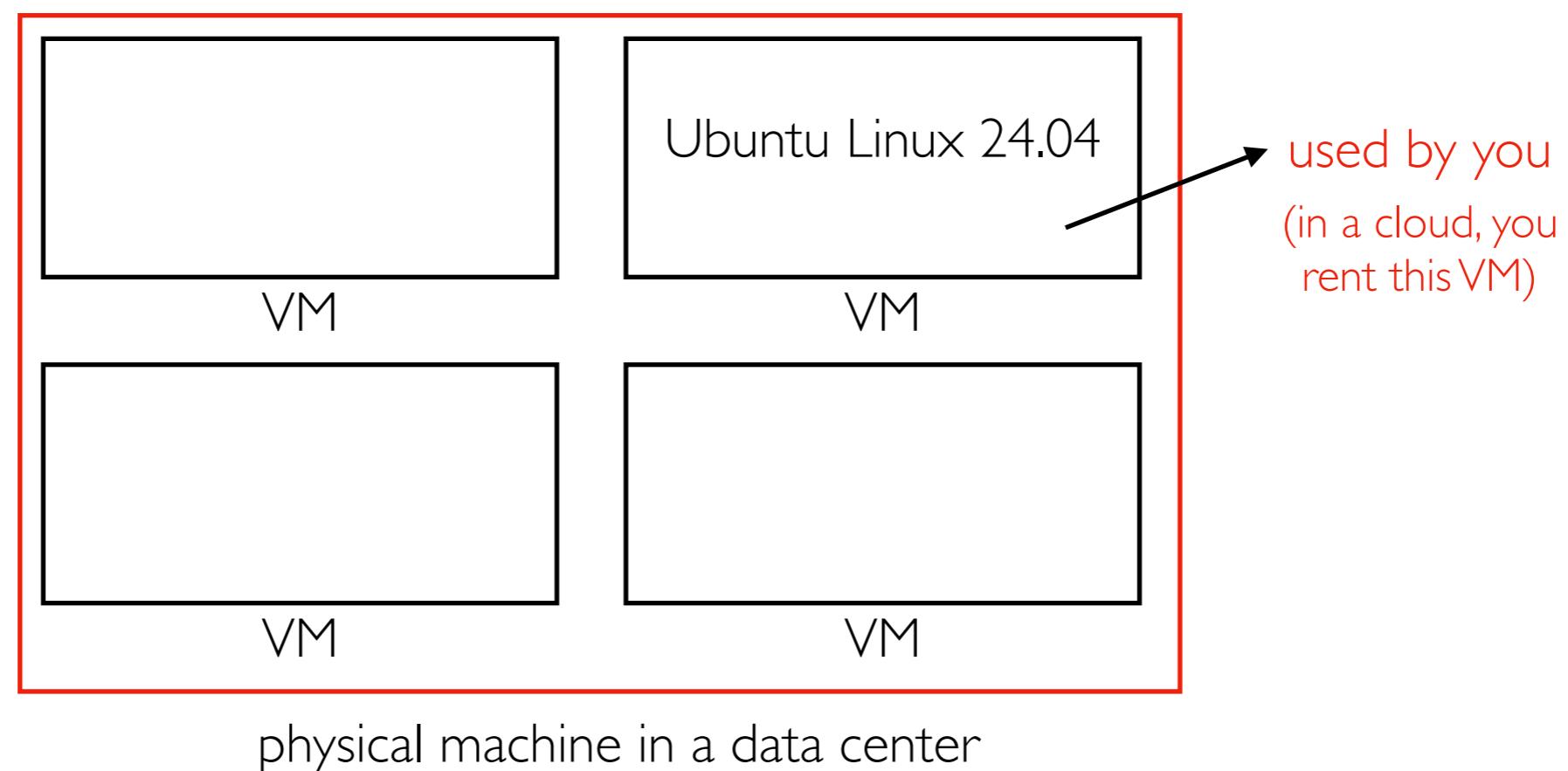
Deployment

Virtual Machines

Deployment means running code somewhere

- often a major undertaking when working with clusters

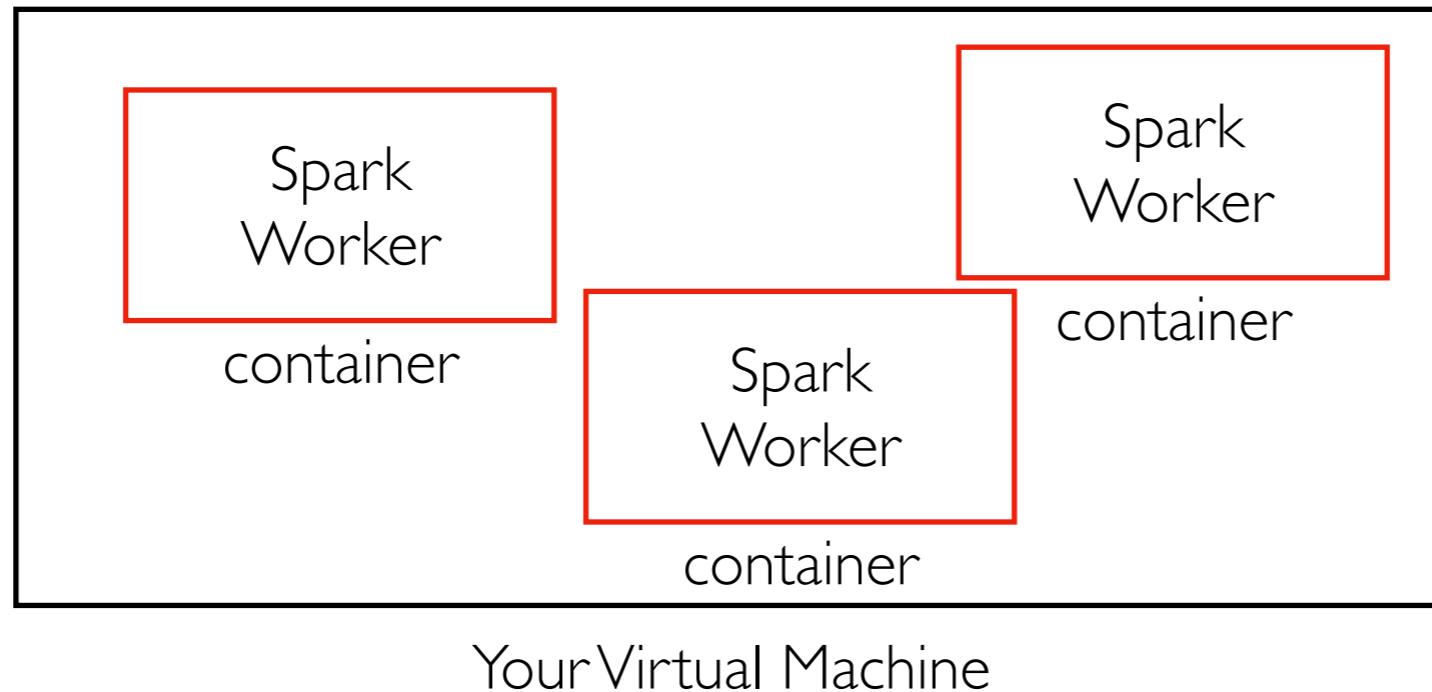
We'll be deploying systems on using virtual machines (VMs).



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