

Where We Are

Machine Learning Systems

Big Data

Cloud

Foundations of Data Systems

2010 - Now

2000 - 2016

1980 - 2000

Logistics

- PA2 released
 - 2 Weeks to finish
- Please participate in guest talk discussion
 - Will initiate discussion threads after every guest talk
 - Develop an understand of technological trend is the most important outcome of this course
- Next guest talk: Ray by Stephanie Wang
- Reading summary
 - The 2 page limit is meant to reduce your workload, but if you want to go over 2 pages, just do it.

Logistics

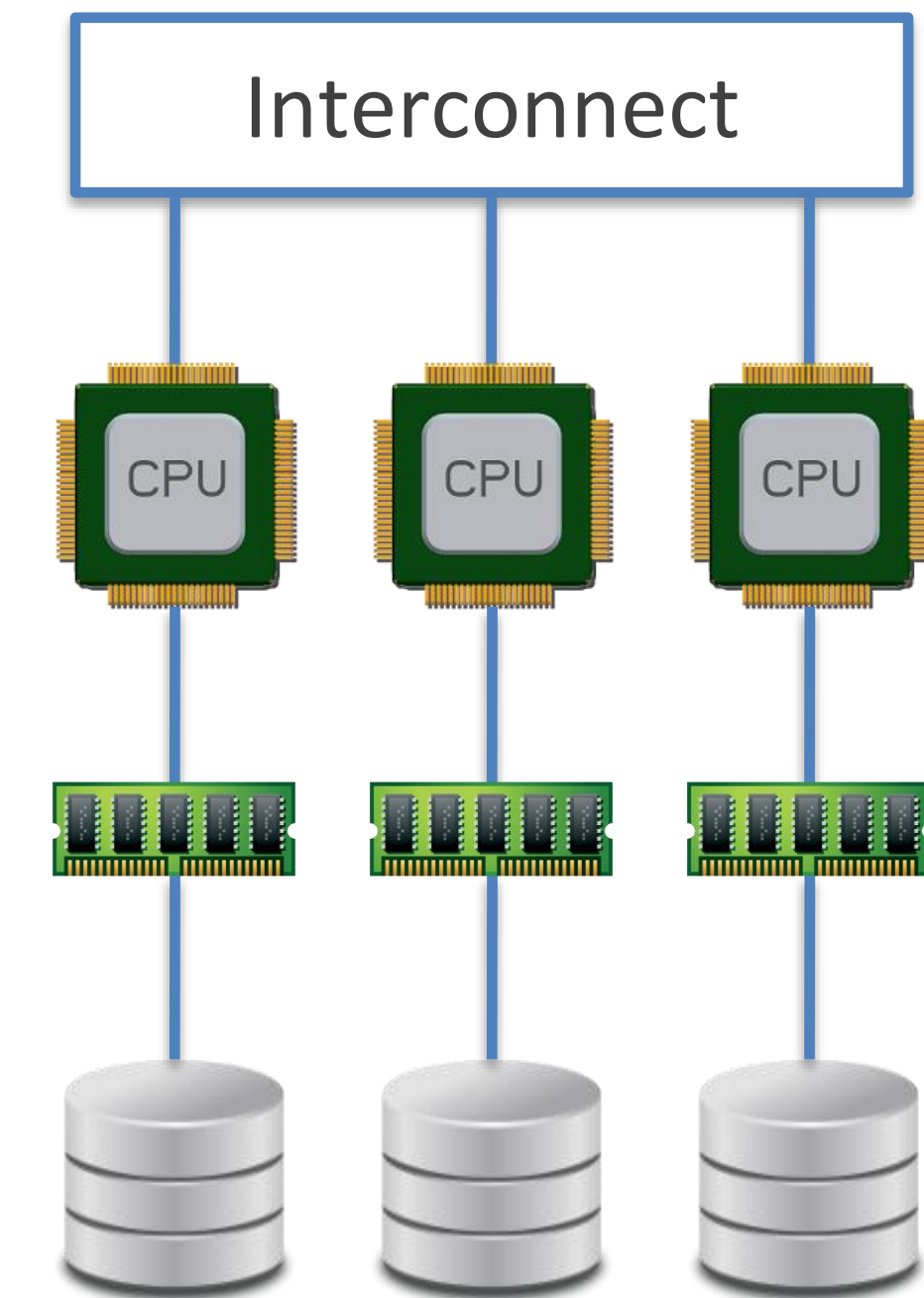
- PA3 will be released in end of week 8
- Scribe duties
 - The scribe scheduling was tentative
 - Please do scribe the day you signed up, even if the topic was changed from the schedule
 - If you really dislike the topic – write an email to instructors

Problems Distributed Systems Need to Solve

- **How to Distribute Data?**
 - **Replicate / Partition the data**
 - **Replicate -> redundancy -> fault tolerance**
 - **Partition -> scalability -> hot partition problem**
- How to Distribute Compute?
 - Batch processing
 - Streaming processing
- How to Coordinate/Synchronize?

Partitioning challenges

- How to partition and how to index?
- How to add or remove nodes?
- How to route the requests and execute queries?



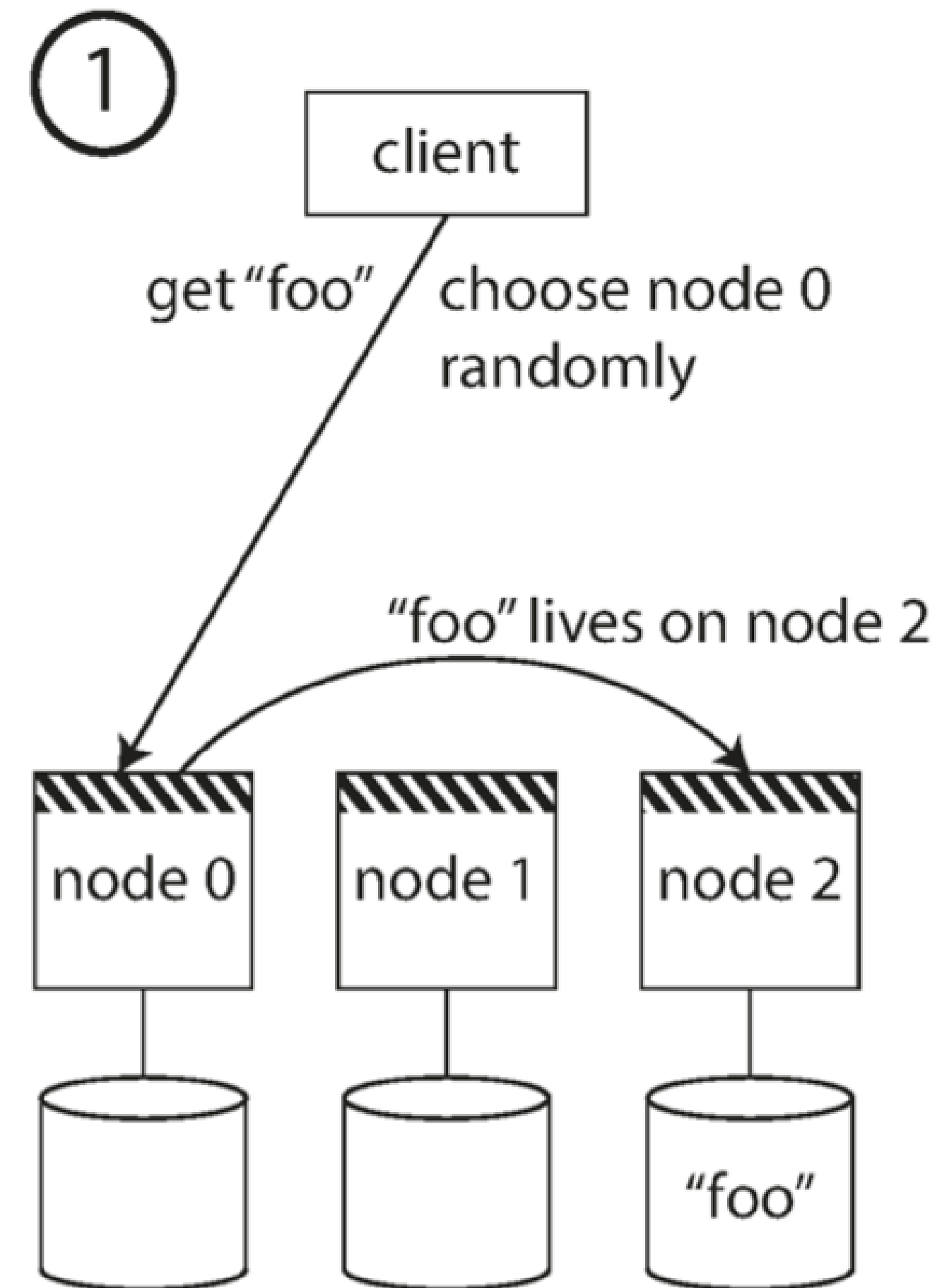
Shared-Nothing
Parallelism

Routing paradigm #1

- Contact any node (e.g., a round-robin load balancer),
- If the node has the data copy, respond.
- If not, forward, receives the reply, and passes the reply along to the client.

Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications

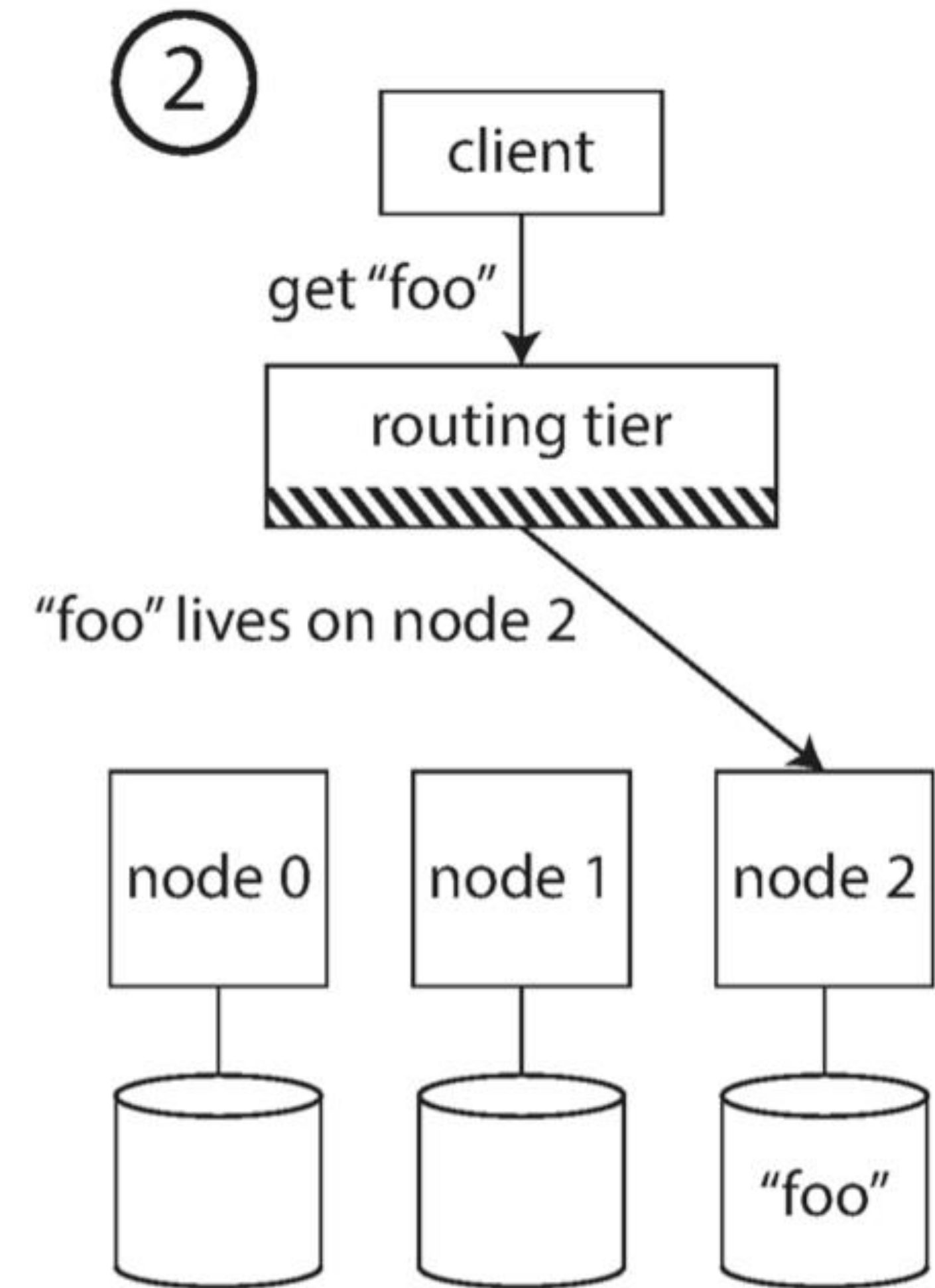
Ion Stoica; Robert Morris, David Karger, M. Frans Kaashoek, Hari Balakrishnan†
MIT Laboratory for Computer Science
chord@lcs.mit.edu
<http://pdos.lcs.mit.edu/chord/>



//// = the knowledge of which partition is assigned to which node

Routing paradigm #2

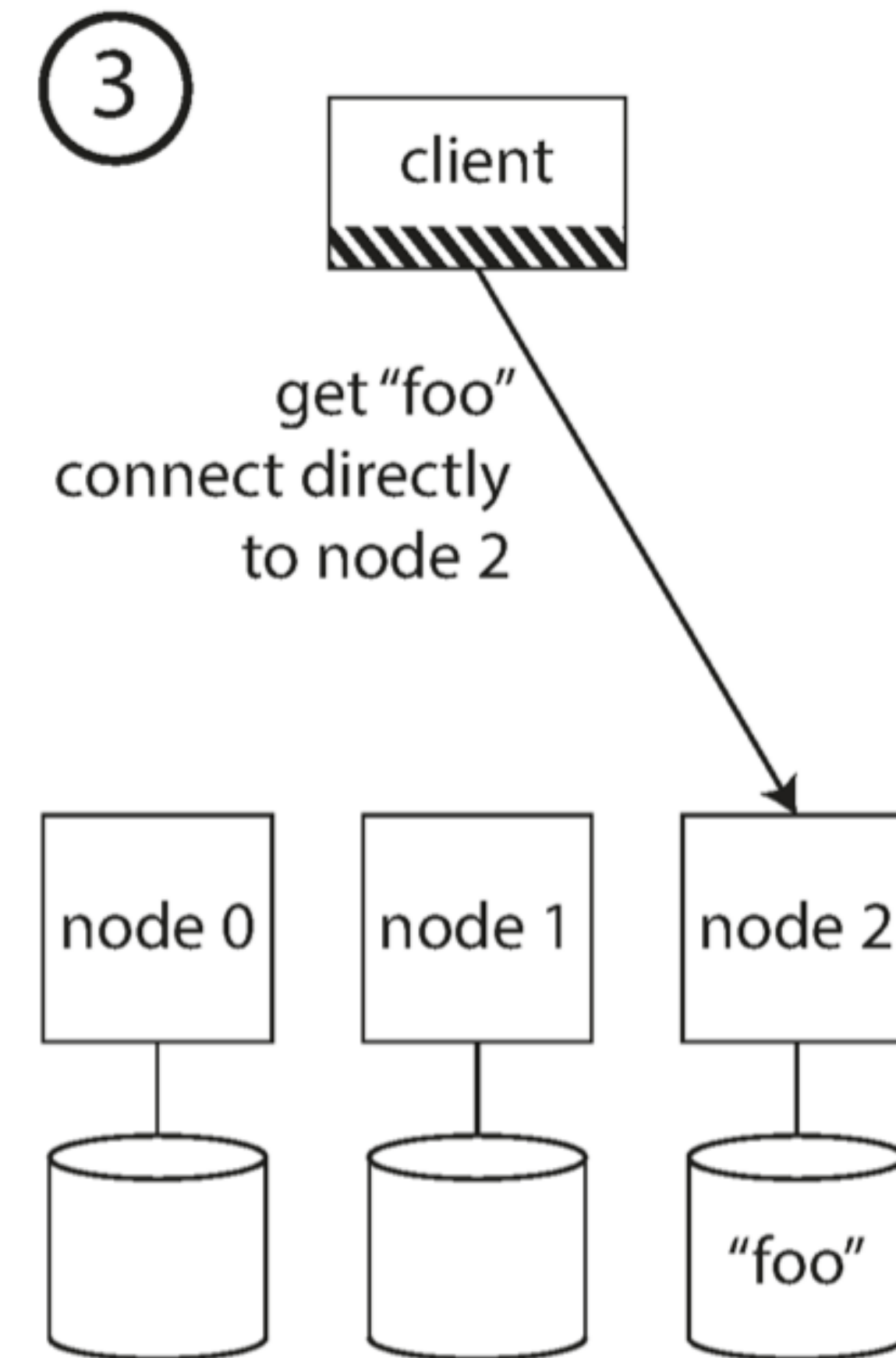
- Send all requests to a routing tier first.
- The routing tier forward all the requests.



//// = the knowledge of which partition is assigned to which node

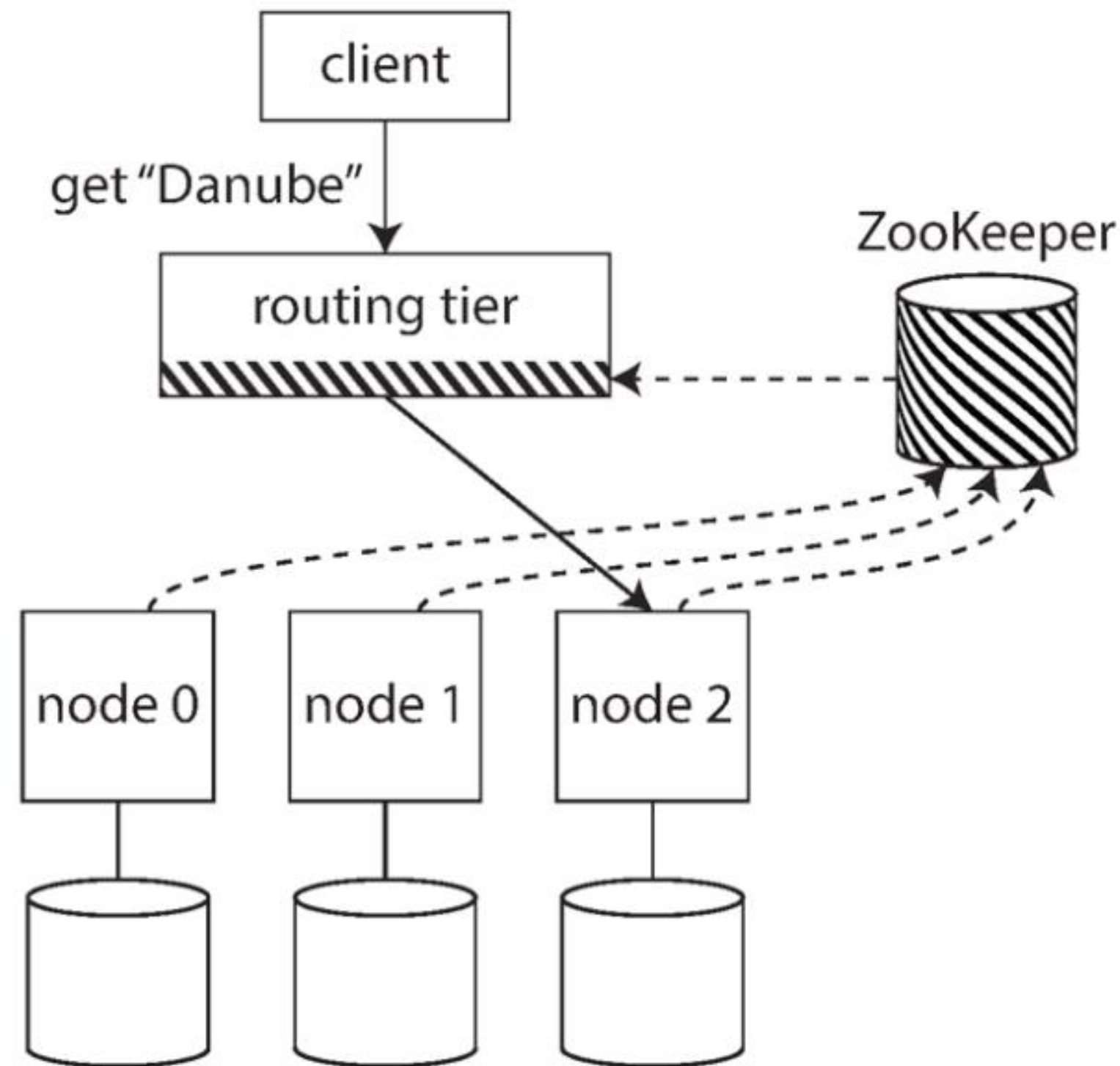
Routing paradigm #3

- The client is aware of the partitioning and the assignment of partitions to nodes.
- No intermediary.



//// = the knowledge of which partition is assigned to which node

ZooKeeper

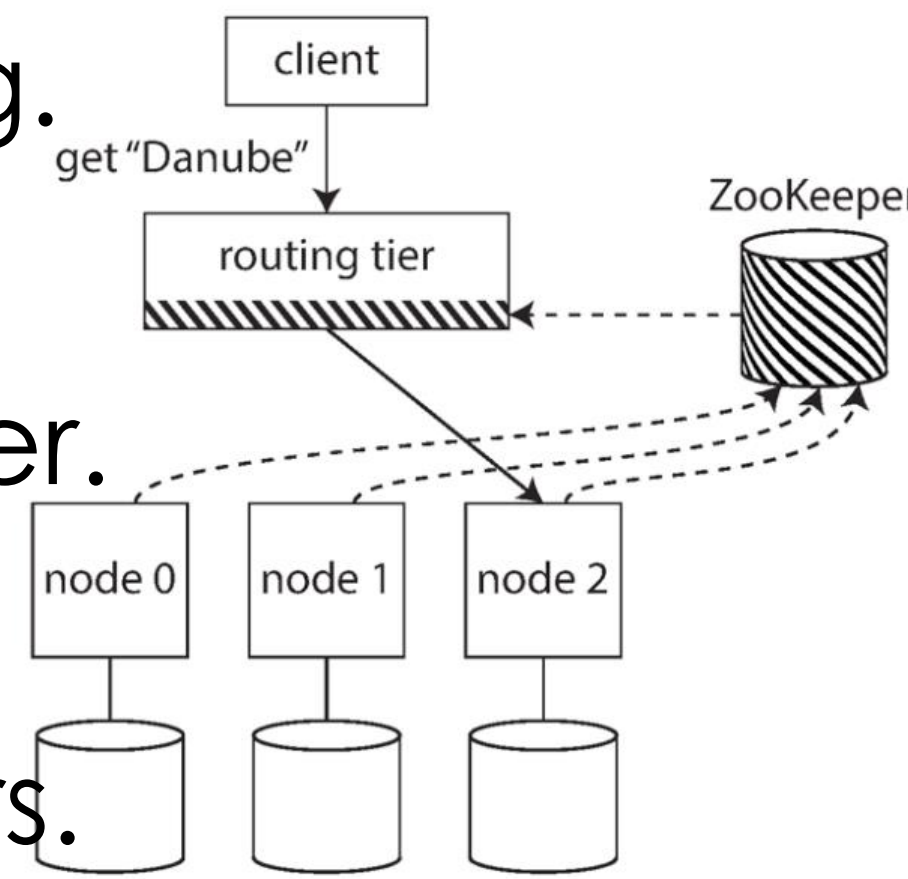


Key range	Partition	Node	IP address
A-ak — Bayes	partition 0	node 0	10.20.30.100
Bayeu — Ceanothus	partition 1	node 1	10.20.30.101
Ceara — Deluc	partition 2	node 2	10.20.30.102
Delusion — Frenssen	partition 3	node 0	10.20.30.100
Freon — Holderlin	partition 4	node 1	10.20.30.101
Holderness — Krasnoje	partition 5	node 2	10.20.30.102
Krasnokamsk — Menadra	partition 6	node 0	10.20.30.100
Menage — Ottawa	partition 7	node 1	10.20.30.101
Otter — Rethimnon	partition 8	node 2	10.20.30.102
Reti — Solovets	partition 9	node 0	10.20.30.100
Solovyov — Truck	partition 10	node 1	10.20.30.101
Trudeau — Zywiec	partition 11	node 2	10.20.30.102

//// = the knowledge of which partition is assigned to which node

ZooKeeper

- Each node registers itself in ZooKeeper.
- ZooKeeper maintains the mapping.
- Other actors (different in three paradigms) subscribe to ZooKeeper.
- Whenever the partition mapping changes, ZooKeeper notifies actors.



Key range	Partition	Node	IP address
A-ak — Bayes	partition 0	node 0	10.20.30.100
Bayeu — Ceanothus	partition 1	node 1	10.20.30.101
Ceara — Deluc	partition 2	node 2	10.20.30.102
Delusion — Frenssen	partition 3	node 0	10.20.30.100
Freon — Holderlin	partition 4	node 1	10.20.30.101
Holderness — Krasnoje	partition 5	node 2	10.20.30.102
Krasnokamsk — Menadra	partition 6	node 0	10.20.30.100
Menage — Ottawa	partition 7	node 1	10.20.30.101
Otter — Rethimnon	partition 8	node 2	10.20.30.102
Reti — Solovets	partition 9	node 0	10.20.30.100
Solovyov — Truck	partition 10	node 1	10.20.30.101
Trudeau — Zywiec	partition 11	node 2	10.20.30.102

//// = the knowledge of which partition is assigned to which node

Takeaway

- The benefits of Partitioning and Replication.
- The challenges of Partitioning and Replication.
- The tradeoffs of different strategies.
- Replication: single-leader, multiple-leader, leaderless
- Partition: Key range, hash, hybrid.
 - Partition rebalancing strategies: fixed, dynamic
 - Partition routing

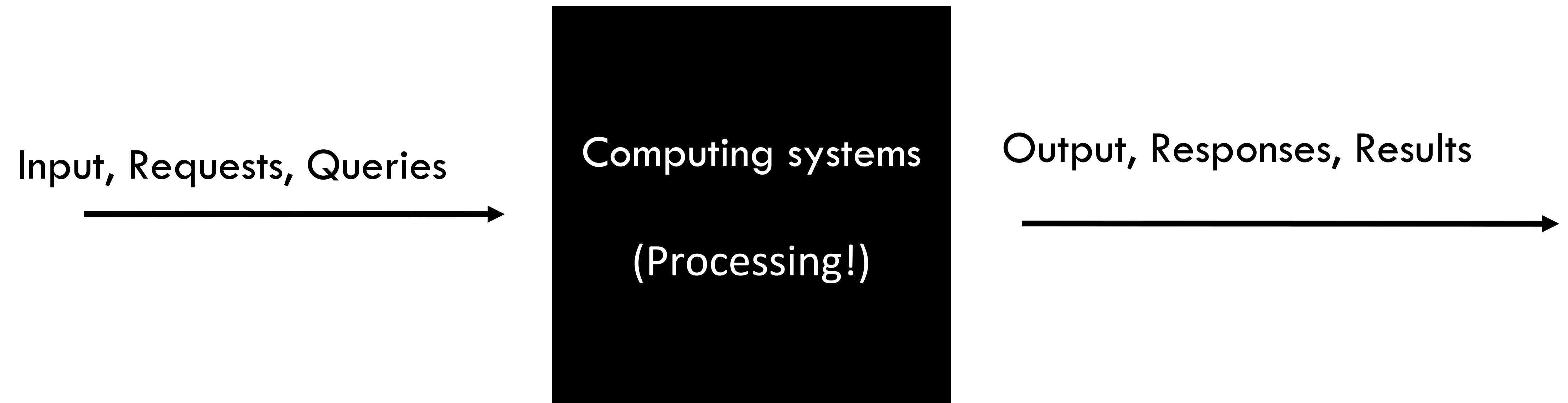
Problems Distributed Systems Need to Solve

- How to Distribute Data?
 - Replicate / Partition the data
 - Replicate -> redundancy -> fault tolerance
 - Partition -> scalability
- **How to Distribute Compute?**
 - Batch processing
 - Streaming processing
- How to Coordinate/Synchronize?

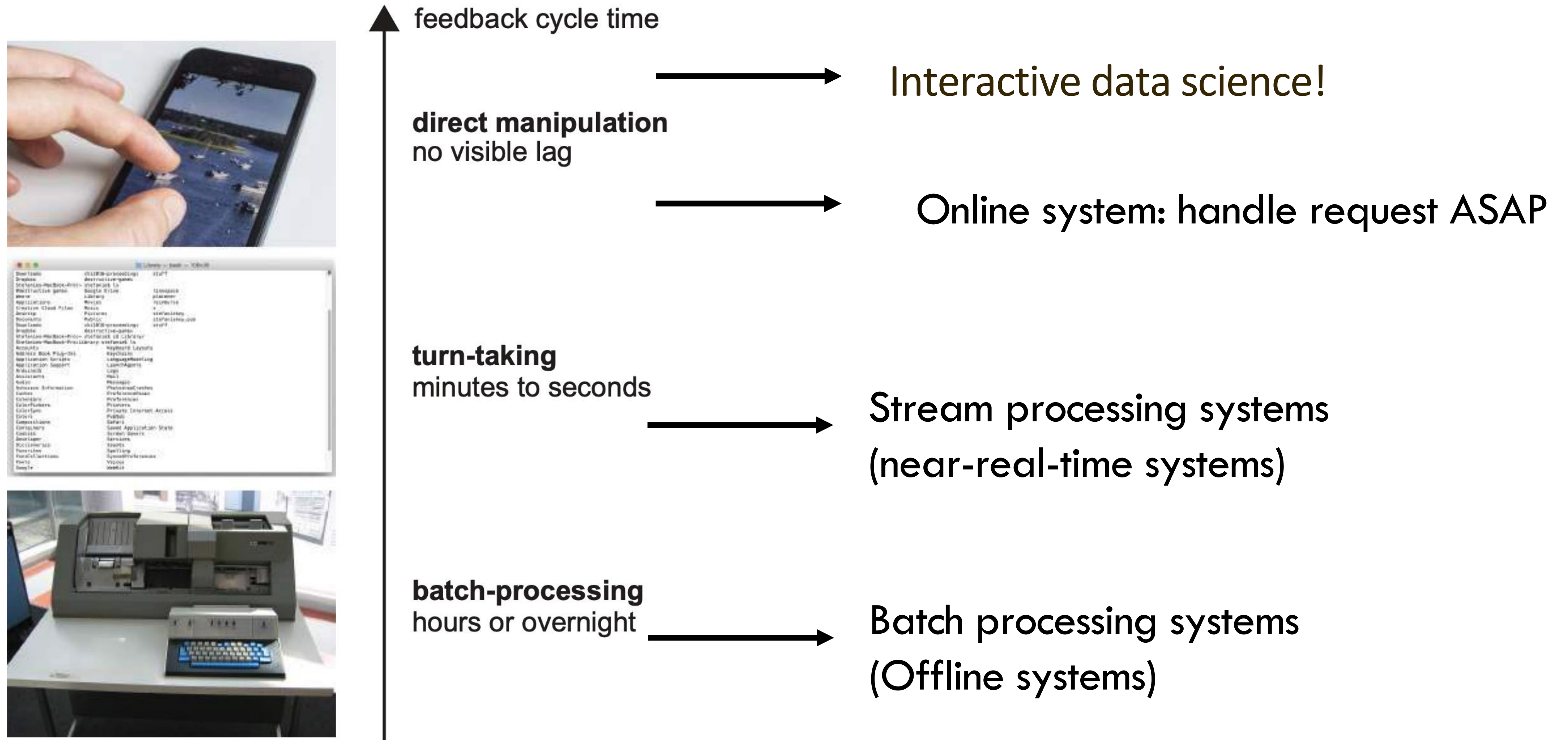
Today's topic: Batch Processing

- Overview
- IO & Unix pipes
- MapReduce
- Beyond MapReduce

Basic Computing System Paradigm



Processing latency



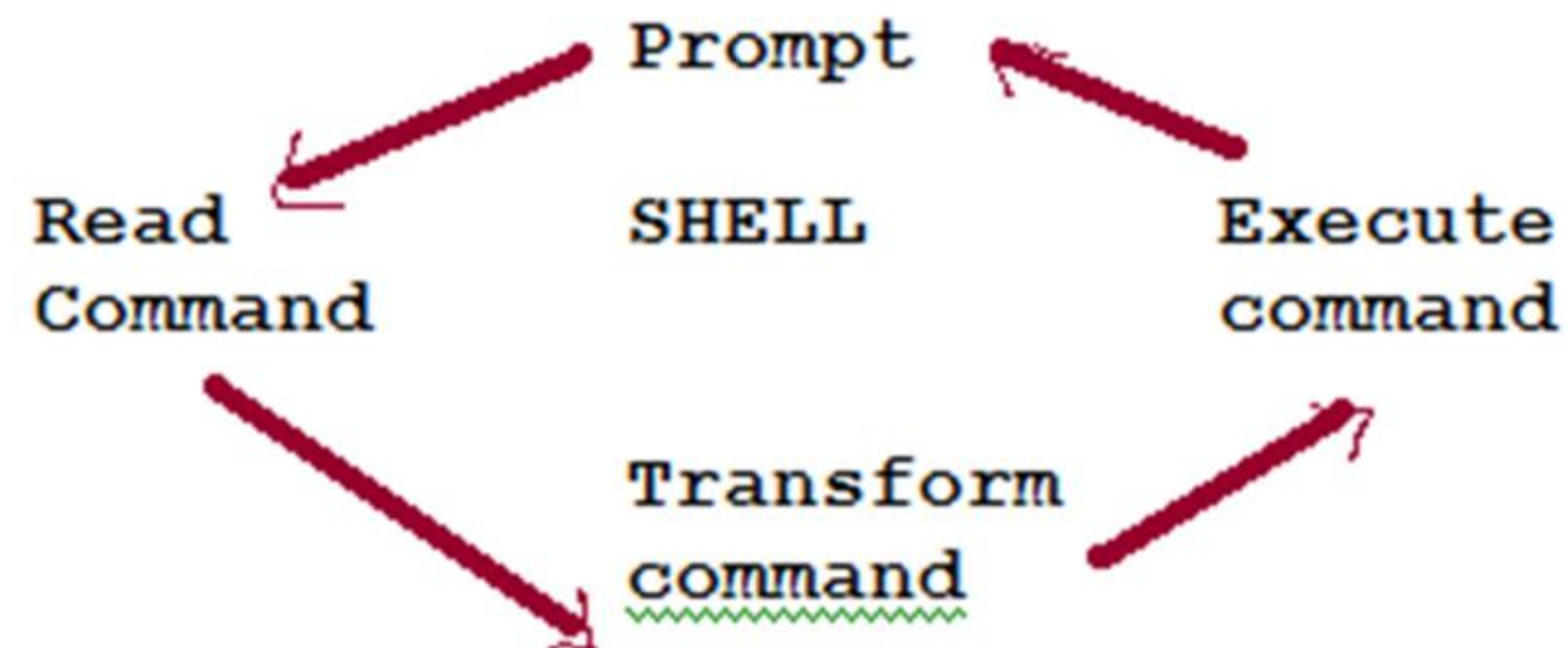
Today's topic: Batch Processing

- Overview
- IO & Unix pipes
- MapReduce
- Beyond MapReduce

The shell

A command line interpreter that provides the interface to Unix OS.

```
hao@HaoPC: /mnt/e/projects x + v
hao@HaoPC: /mnt/e/projects/projects/courses/dsc204a-w24$ ls
Gemfile      README.md  _includes  sass       announcements.md  faq.md      resources.md  syllabus.md
Gemfile.lock  _announcements  _layouts  _site      assets           index.md    schedule.md
LICENSE      _config.yml  _modules  _staffers  assignments.md  officehours.md  staff.md
hao@HaoPC: /mnt/e/projects/projects/courses/dsc204a-w24$ |
```



Shell example

“****rc”

Run commands



```
hao@HaoPC: /mnt/e/projects x + v
hao@HaoPC: /mnt/e/projects/projects/courses/dsc204a-w24$ ls -lah ~/.
total 256K
drwxr-xr-x 30 hao hao 4.0K Feb 16 14:01 .
drwxr-xr-x 3 root root 4.0K Jul 2 2021 ..
drwxr-xr-x 2 hao hao 4.0K Sep 1 00:10 .aws
drwxr-xr-x 5 hao hao 4.0K Nov 4 2021 .azure
-rw----- 1 hao hao 25K Feb 16 14:01 .bash_history
-rw-r--r-- 1 hao hao 220 Jul 2 2021 .bash_logout
-rw-r--r-- 1 hao hao 4.4K Jan 1 22:17 .bashrc
-rw----- 1 hao hao 21K Apr 13 2023 .boto
drwxr-xr-x 3 hao hao 4.0K Apr 29 2023 .bundle
drwxr-xr-x 11 hao hao 4.0K Jun 18 2023 .cache
drwxr-xr-x 12 hao hao 4.0K Aug 16 2023 .config
drwxr-xr-x 3 hao hao 4.0K Nov 21 2022 .cupy
drwxr-xr-x 3 hao hao 4.0K Oct 3 2021 .eclipse
drwxr-xr-x 4 hao hao 4.0K Apr 29 2023 .gem
-rw-r--r-- 1 hao hao 96 Jun 13 2023 .gitconfig
drwx----- 2 hao hao 4.0K Jun 22 2022 .gnupg
drwxr-xr-x 3 hao hao 4.0K May 12 2023 .gsutil
drwxr-xr-x 3 hao hao 4.0K Nov 22 2022 .ipython
drwxr-xr-x 2 hao hao 4.0K Nov 22 2022 .jupyter
drwxr-xr-x 3 hao hao 4.0K Jan 1 2023 .kube
drwxr-xr-x 2 hao hao 4.0K Jul 2 2021 .landscape
drwx----- 7 hao hao 4.0K Jan 6 02:09 .local
-rw-r--r-- 1 hao hao 0 Feb 23 09:36 .motd_shown
drwxr-xr-x 2 hao hao 4.0K Apr 28 2023 .ngrok
-rw----- 1 hao hao 18 Jun 22 2023 .node_repl_history
drwxr-xr-x 7 hao hao 4.0K Apr 30 2023 .npm
drwx----- 3 hao hao 4.0K Dec 30 2022 .nv
drwxr-xr-x 8 hao hao 4.0K Apr 28 2023 .nvm
-rw-r--r-- 1 hao hao 807 Jul 4 2023 .profile
drwxr-xr-x 23 hao hao 4.0K Aug 18 2023 .pycharm_helpers
-rw----- 1 hao hao 4.1K Aug 25 22:12 .python_history
drwxr-xr-x 2 hao hao 4.0K Nov 28 2022 .ray
drwxr-xr-x 4 hao hao 4.0K Jan 1 22:21 .rbenv
drwxr-xr-x 2 hao hao 4.0K Feb 20 2023 .skyplane
drwxr-xr-x 2 hao hao 4.0K Dec 10 12:23 .ssh
-rw-r--r-- 1 hao hao 0 Jul 24 2021 .sudo_as_admin_successful
drwxr-xr-x 2 hao hao 4.0K Dec 10 2022 .vim
-rw----- 1 hao hao 32K Jan 7 18:34 .viminfo
-rw-r--r-- 1 hao hao 355 Nov 4 2021 .vimrc
drwxr-xr-x 5 hao hao 4.0K Mar 12 2022 .vscode-server-insiders
-rw-r--r-- 1 hao hao 215 May 15 2023 .wget-hsts
-rw-r--r-- 1 hao hao 0 Sep 12 23:26 calculate_flops.py
-rwxr-xr-x 1 hao hao 3.6K Sep 13 00:39 estimate_throughput.py
drwxr-xr-x 6 hao hao 4.0K Jun 18 2023 logs-env
drwxr-xr-x 12 hao hao 4.0K Aug 21 2023 my_site
-rw-r--r-- 1 hao hao 646 Sep 2 01:48 perf_model.py
-rw-r--r-- 1 hao hao 333 Sep 2 02:13 test.py
```

Useful shell commands

- Shell already has a collection of rich commands
 - Some Useful commands
 - uptime, cut, date, cat, finger, hexdump, man, md5sum, quota,
 - mkdir, rmdir, rm, mv, du, df, find, cp, chmod, cd
 - unname, zip, unzip, gzip, tar
 - tr, sed, sort, uniq, ascii
 - Type “man command” to read about shell commands

What do these shell commands do?

- `cat dups.txt | sort | uniq`
- `cat dups.txt | sort -V | uniq`
- `cat dups.txt | sort -V | uniq > outfile.txt`
- `tr "a" "e" < z.txt`
- `cat z.txt | tr a e`

Batch processing with Unix Tools

```
cat /var/log/nginx/access.log | ①  
  awk '{print $7}' | ②  
  sort | ③  
  uniq -c | ④  
  sort -r -n | ⑤  
  head -n 5 ⑥
```

```
4189 /favicon.ico  
3631 /2013/05/24/improving-security-of-ssh-private-keys.html  
2124 /2012/12/05/schema-evolution-in-avro-protocol-buffers-thrift.html  
1369 /  
915 /css/typography.css
```

- Read the log file.
- Split each line into fields by white space, output only the 7th element (requested URL).
- Alphabetically sort
- Filter out repeated lines.
- Sort it again based on the line number (-n)
- Output the first five lines.

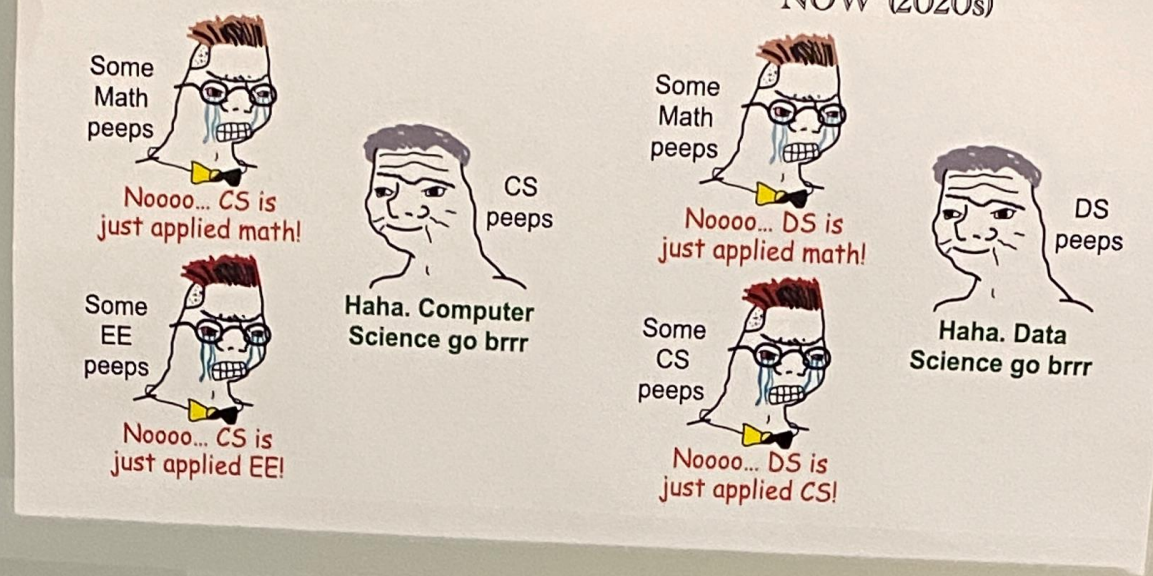
Unix philosophy

Style

A number of maxims have gained currency among the builders and users of the UNIX system to explain and promote its characteristic style:

- (i) Make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new "features."
- (ii) Expect the output of every program to become the input to another, as yet unknown, program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.
- (iii) Design and build software, even operating systems, to be tried early, ideally within weeks. Don't hesitate to throw away the clumsy parts and rebuild them.
- (iv) Use tools in preference to unskilled help to lighten a

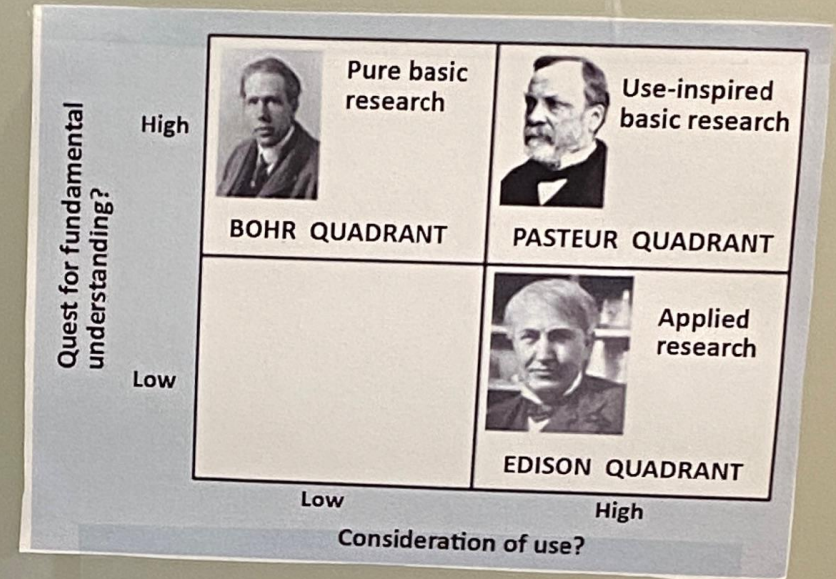
1902 THE BELL SYSTEM TECHNICAL JOURNAL, JULY-AUGUST 1978



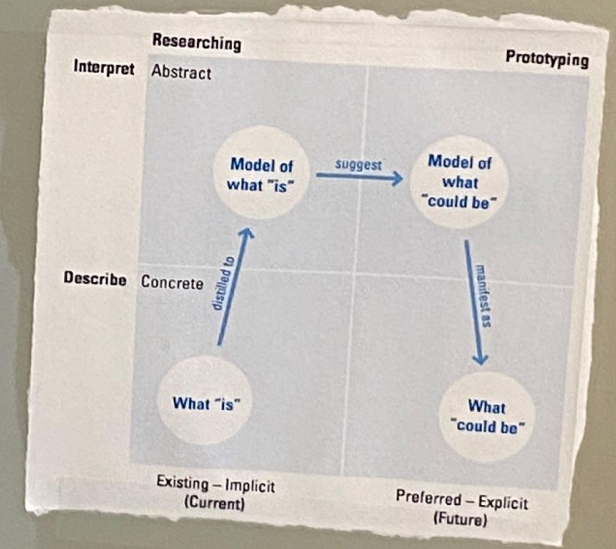
Heilmeier's Catechism

When George Heilmeier was the ARPA director in the mid 1970s he had a standard set of questions he expected every proposal for a new research program to answer.

1. What is the problem, why is it hard?
2. How is it solved today?
3. What is the new technical idea; why can we succeed now?
4. What is the impact if successful?
5. How will the program be organized?
6. How will intermediate results be generated?
7. How will you measure progress?
8. What will it cost?

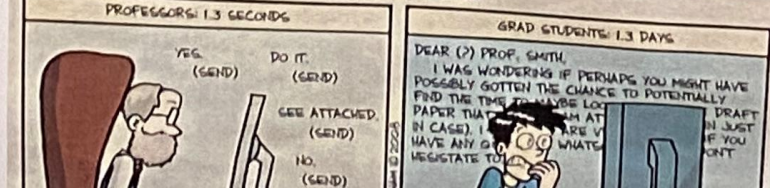


Clients Users



Style
A number of maxims have gained currency among the builders and users of the UNIX system to explain and promote its characteristic style:
(i) Make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new "features."
(ii) Expect the output of every program to become the input to another, as yet unknown, program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.
(iii) Design and build software, even operating systems, to be tried early, ideally within weeks. Don't hesitate to throw away the clumsy parts and rebuild them.
(iv) Use tools in preference to unskilled help to lighten a

AVERAGE TIME SPENT COMPOSING ONE E-MAIL



Unix philosophy

- Make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new “features”.
- Expect the output of every program to become the input to another, as yet unknown, program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.
- Design and build software, even operating systems, to be tried early, ideally within weeks. Don't hesitate to throw away the clumsy parts and rebuild them.
- Use tools in preference to unskilled help to lighten a programming task, even if you have to detour to build the tools and expect to throw some of them out after you've finished using them.

Unix philosophy

- Make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new “features”.
- Expect the output of every program to become the input to another, as yet unknown, program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.
- Design and build software, even operating systems, to be tried early, ideally within weeks. Don't hesitate to throw away the clumsy parts and rebuild them.
- Use tools in preference to unskilled help to lighten a programming task, even if you have to detour to build the tools and expect to throw some of them out after you've finished using them.

Unix philosophy

- Make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new “features”.
- Expect the output of every program to become the input to another, as yet unknown, program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.
- Design and build software, even operating systems, to be tried early, ideally within weeks. Don't hesitate to throw away the clumsy parts and rebuild them.
- Use tools in preference to unskilled help to lighten a programming task, even if you have to detour to build the tools and expect to throw some of them out after you've finished using them.

Unix philosophy

- Make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new “features”.
- Expect the output of every program to become the input to another, as yet unknown, program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.
- Design and build software, even operating systems, to be tried early, ideally within weeks. Don't hesitate to throw away the clumsy parts and rebuild them.
- Use tools in preference to unskilled help to lighten a programming task, even if you have to detour to build the tools and expect to throw some of them out after you've finished using them.

Transparency and experimentation

- The input files to Unix commands are normally treated as immutable.
 - Run most commands without damaging the input files.
- You can end the pipeline at any point, pipe the output into `less`, and look at it to see if it has the expected form.
 - Great for debugging.
- You can write the output of one pipeline stage to a file and use that file as input to the next stage.
 - Restart process.

The biggest limitation of Unix tools is that they **run only on a single machine** — and that's where tools like Hadoop come in.

Today's topic: Batch Processing

- Overview
- IO & Unix pipes
- MapReduce
 - HDFS - infrastructure
 - Job execution
 - Programming models
 - Workflow
- Beyond MapReduce

Batch processing with Unix Tools

```
cat /var/log/nginx/access.log | ①  
  awk '{print $7}' | ②  
  sort | ③  
  uniq -c | ④  
  sort -r -n | ⑤  
  head -n 5 ⑥
```

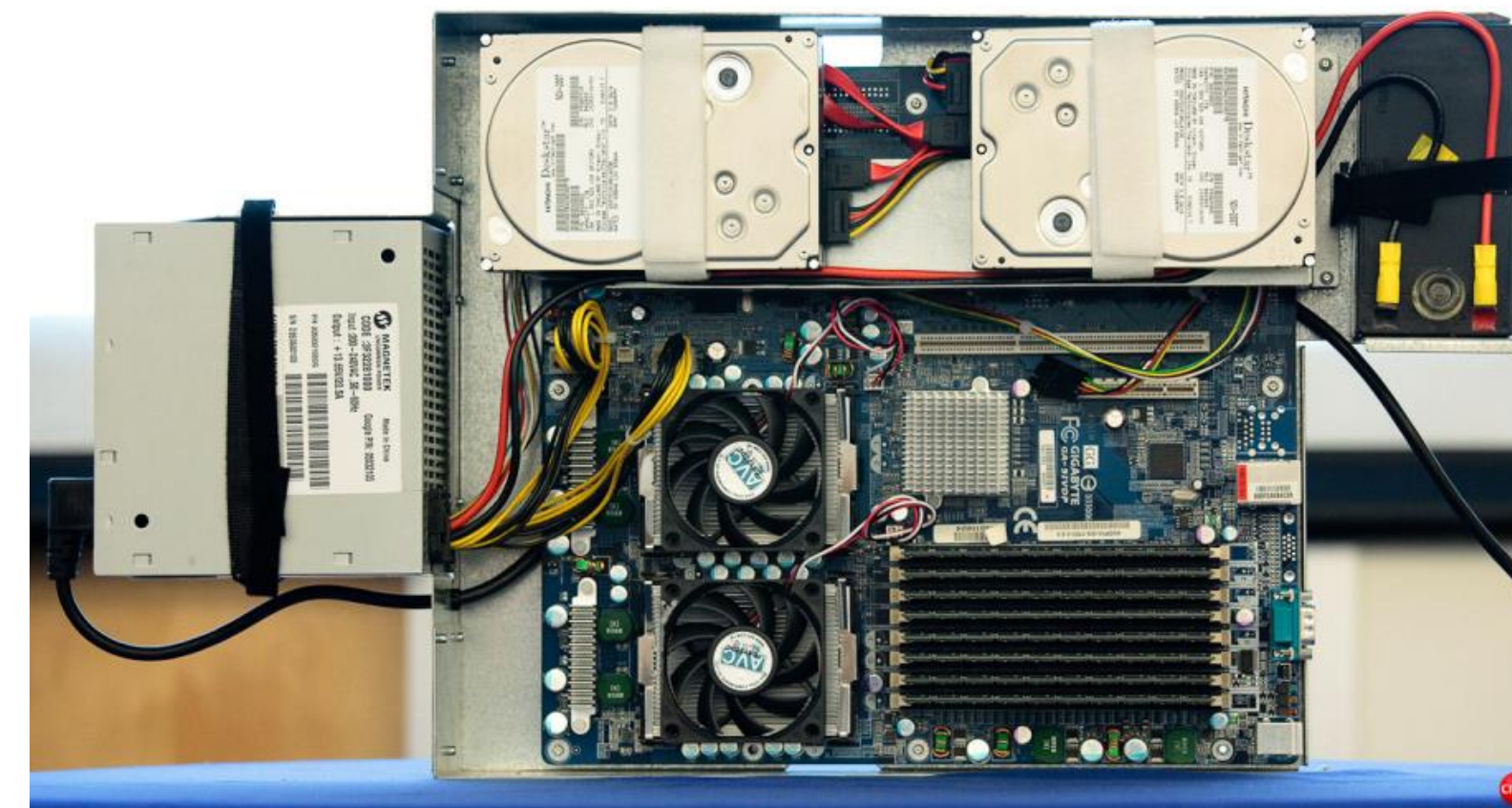
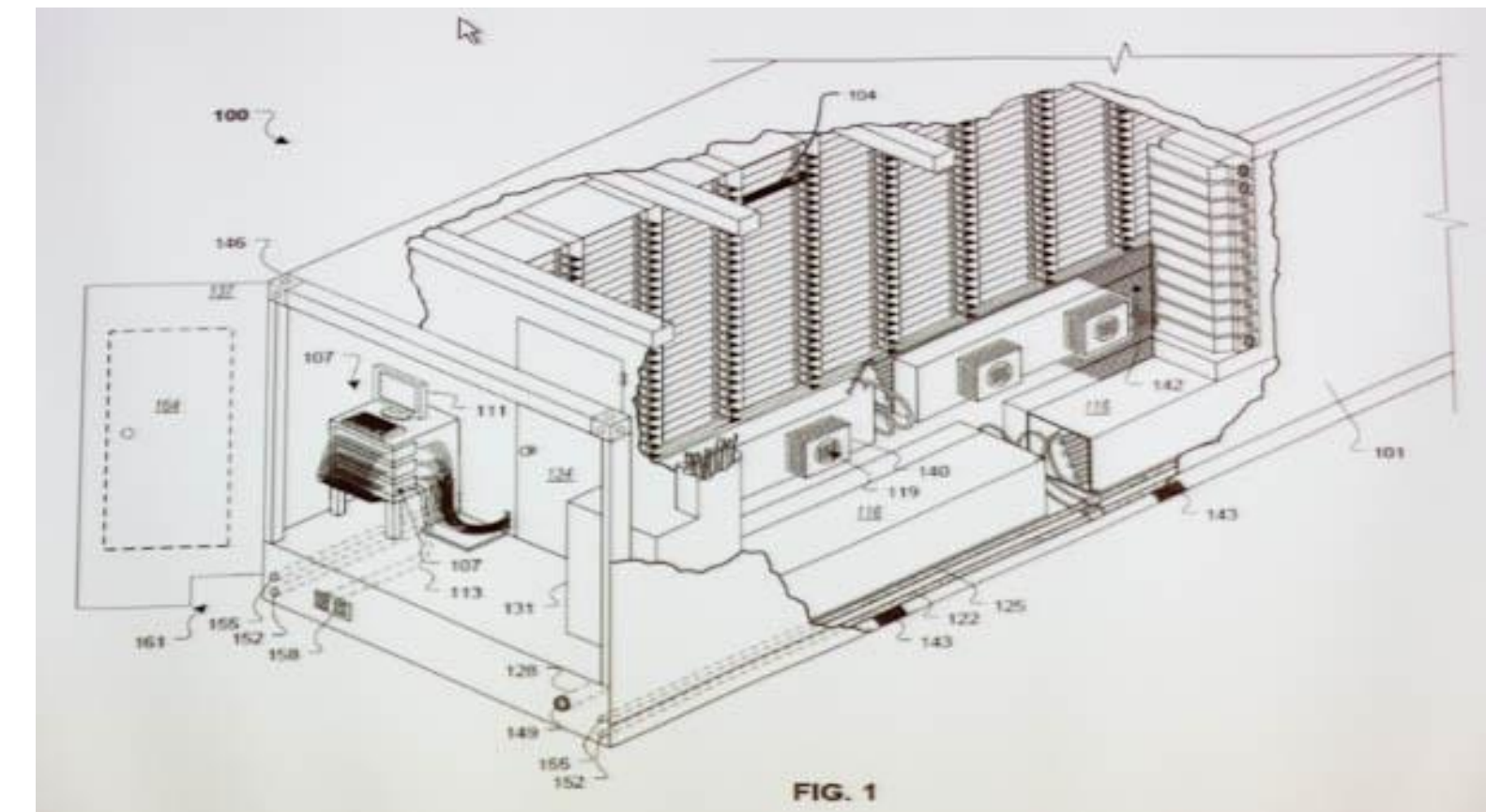
```
4189 /favicon.ico  
3631 /2013/05/24/improving-security-of-ssh-private-keys.html  
2124 /2012/12/05/schema-evolution-in-avro-protocol-buffers-thrift.html  
1369 /  
915 /css/typography.css
```

- Read the log file.
- Split each line into fields by white space, output only the 7th element (requested URL).
- Alphabetically sort
- Filter out repeated lines.
- Sort it again based on the line number (-n)
- Output the first five lines.

Today's topic: Batch Processing

- Overview
- IO & Unix pipes
- MapReduce
 - HDFS - infrastructure
 - Programming models
 - Job execution
 - Workflow
- Beyond MapReduce

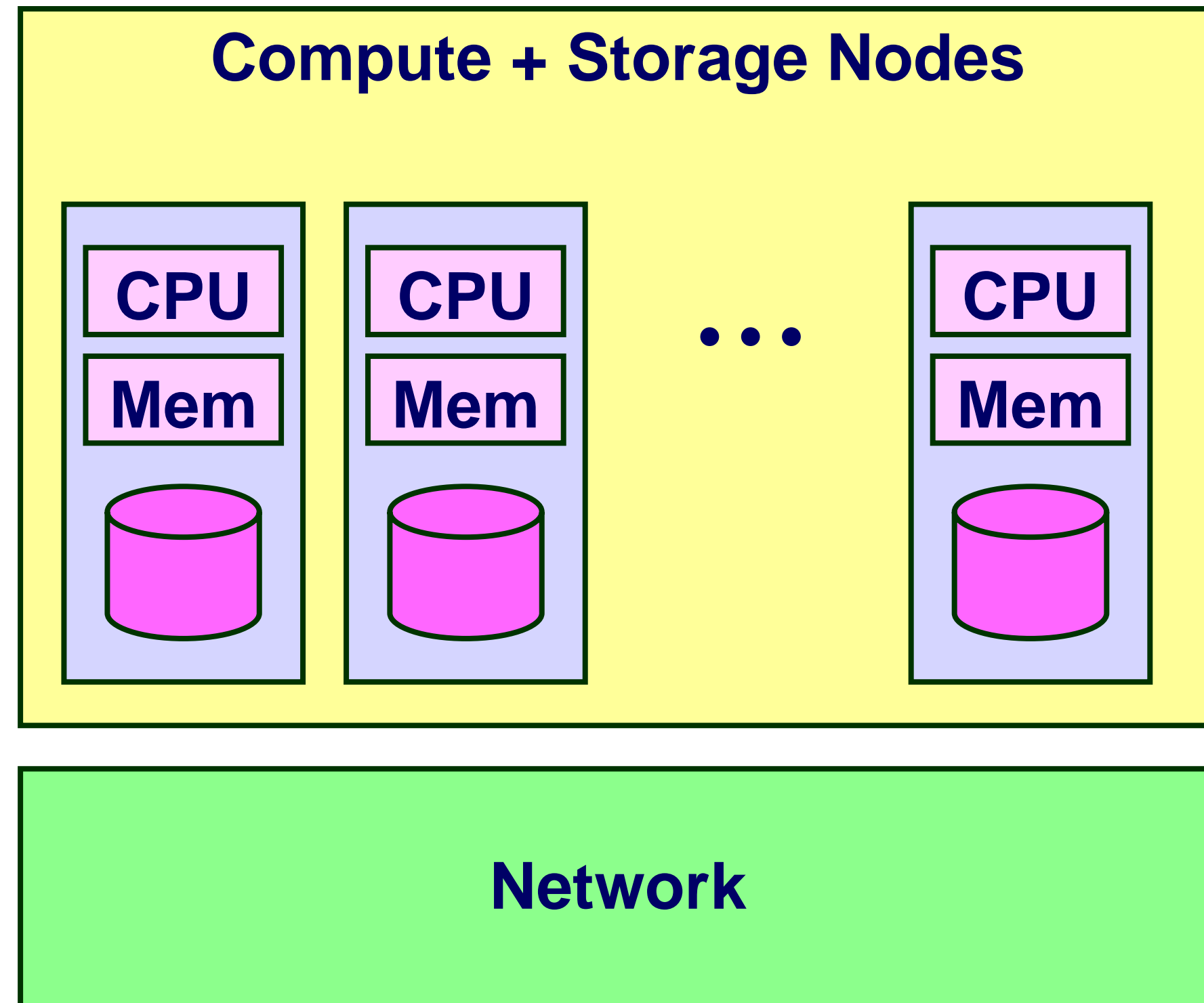
Google Data Centers



- Dalles, Oregon
- Hydroelectric power @ 2¢ / KW Hr
- 50 Megawatts
- Enough to power 60,000 homes

- Engineered for maximum modularity & power efficiency
- Container: 1160 servers, 250KW
- Server: 2 disks, 2 processors

Typical Cluster Machine



Compute + Storage Nodes

- Medium-performance processors
- Modest memory
- 1-2 disks

Network

- Conventional Ethernet switches
 - 10 Gb/s within rack
 - 100 Gb/s across racks

Machines with Disks

Lots of storage for cheap

- 3 TB @ \$150
(5¢ / GB)
- Compare 2007:
0.75 TB @ \$266
35¢ / GB



WD Black 3TB Performance Desktop Hard Disk Drive -
7200 RPM SATA 6 Gb/s 64MB Cache 3.5 Inch -
WD3003FZEX

by Western Digital

★★★★★ 1,615 customer reviews | 28 answered questions

List Price: \$249.99

Price: **\$149.99** ✓Prime

You Save: \$100.00 (40%)

In Stock.

Ships from and sold by Amazon.com. Gift-wrap available.

Want it Friday, Oct. 30? Order within **55 mins** and choose **Two-Day Shipping** at checkout. [Details](#)

Capacity: **3 TB**

1 TB \$71.79 ✓Prime	2 TB \$116.99 ✓Prime	3 TB \$149.99 ✓Prime	4 TB \$198.00 ✓Prime
5 TB \$260.99	6 TB \$298.91		

Drawbacks

- Long and highly variable delays
- Not very reliable

Not included in HPC Nodes

Oceans of Data, Skinny Pipes

1 Terabyte

- Easy to store
- Hard to move



No more blaming connection speeds for your losses.

Verizon FiOS – the fastest Internet available.

Plans as low **\$39.99/month** (up to 5 Mbps).
Plus, order online & **get your first month FREE!**

Enter your home phone number below to check availability.

Don't have a Verizon phone number? [Qualify your address.](#)

Disks	MB / s	Time
Seagate Barracuda	115	2.3 hours
Seagate Cheetah	125	2.2 hours
Networks	MB / s	Time
Home Internet	< 0.625	> 18.5 days
Gigabit Ethernet	< 125	> 2.2 hours
PSC Teragrid Connection	< 3,750	> 4.4 minutes

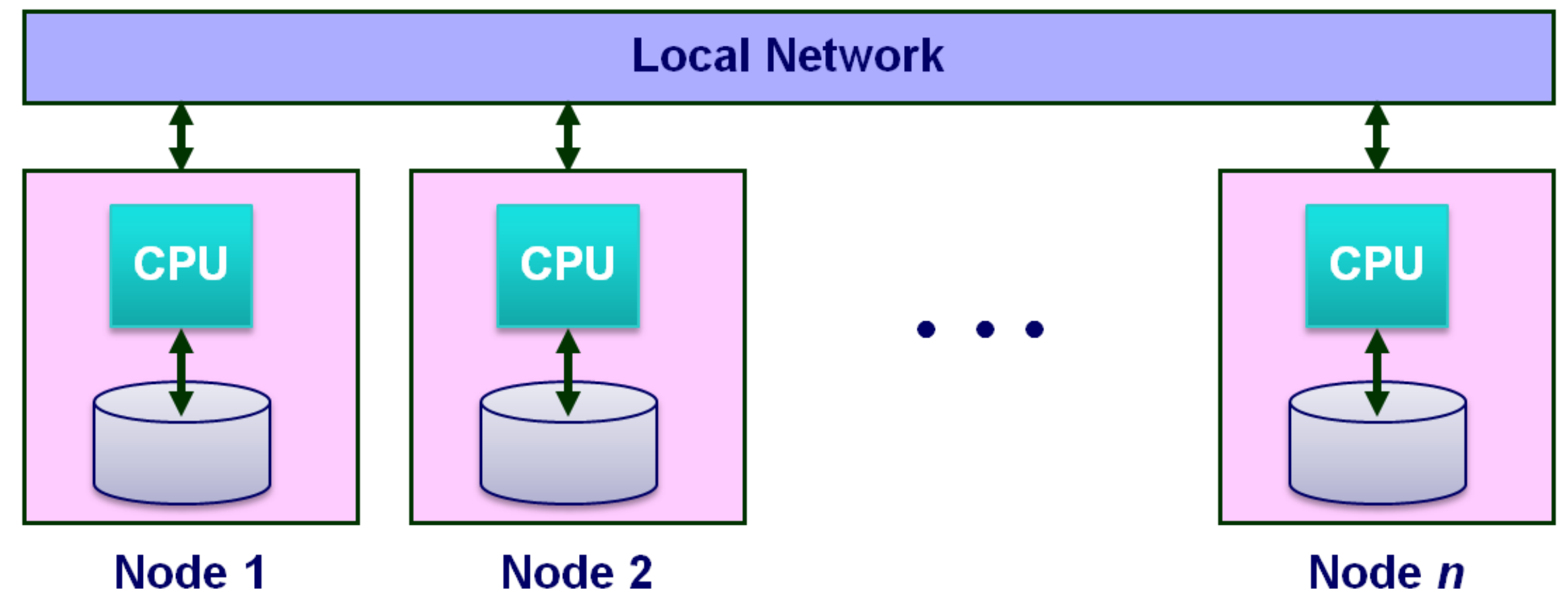
Data-Intensive System Challenge

For Computation That Accesses 1 TB in 5 minutes

- Data distributed over 100+ disks
 - Assuming uniform data partitioning
- Compute using 100+ processors
- Connected by gigabit Ethernet (or equivalent)

System Requirements

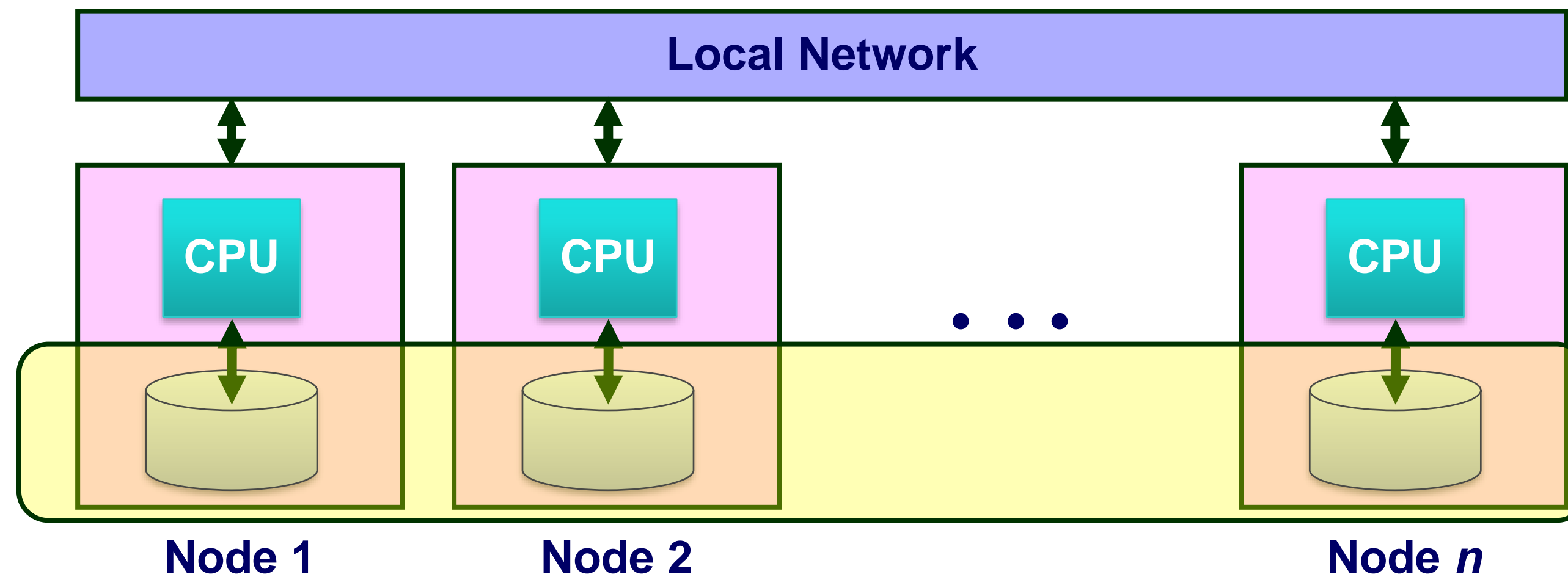
- Lots of disks
- Lots of processors
- Located in close proximity
 - Within reach of fast, local-area network





Hadoop Project

File system with files distributed across nodes



- Store multiple (typically 3 copies of each file)
 - If one node fails, data still available
- Logically, any node has access to any file
 - May need to fetch across network (ideally, leverage locality for perf.)

Map / Reduce programming environment

- Software manages execution of tasks on nodes