

▷ Yogesh Simmhan

▷ simmhan@iisc.ac.in

▷ Department of Computational and Data Sciences

▷ Indian Institute of Science, Bangalore



DS256 (3:1)

Scalable Systems for Data Science



Distributed File Systems

Scale-out data storage using GFS/HDFS

**Tutorials on Fri
1130-1pm, CDS 202**

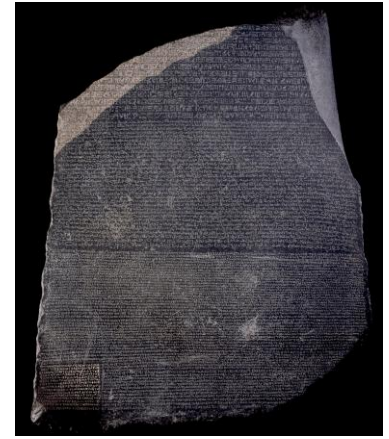
Why do we need a file system?

- ▷ Persistence of data
 - Reliability
- ▷ Logical means to organize and access data
- ▷ Access data by processes
 - Performance
- ▷ Sharing of data across users
 - Access control

History of Storage Media



Cave Paintings

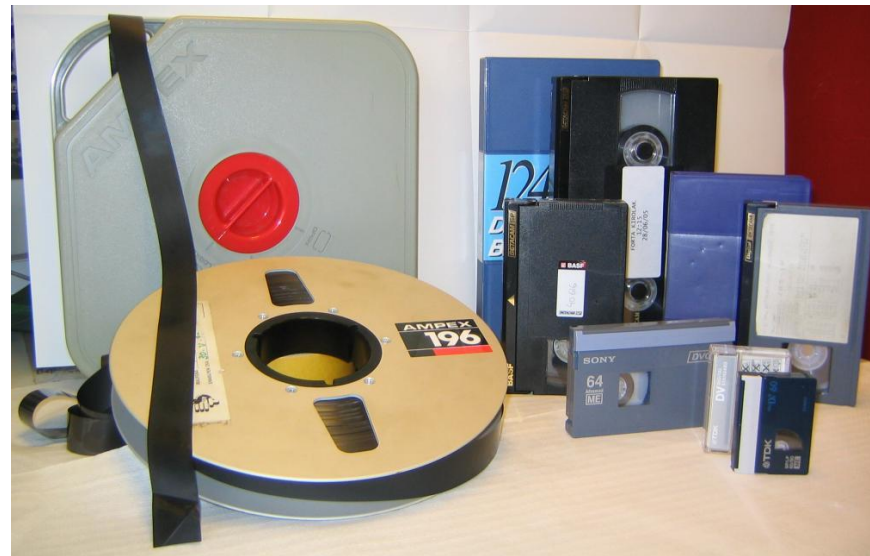


Tablets and Carvings



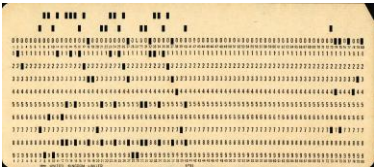
Parchment and Scrolls

History of *Modern* Storage Media



<https://en.wikipedia.org/wiki/Phonograph>
https://en.wikipedia.org/wiki/Cassette_tape
<https://en.wikipedia.org/wiki/Videotape>

History of *Digital* Storage Media



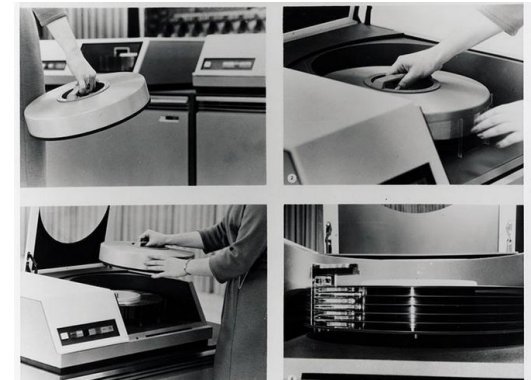
Punched Cards,
~1700~1900
“Jacquard”



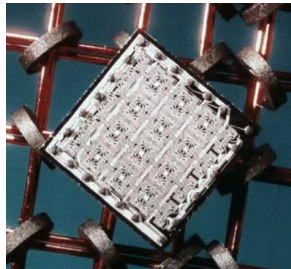
CRT, ~1950



Magnetic Drum & Tape, ~1950



Magnetic Disk, ~1960



DRAM, ~1980



Laser Disc, ~1980



Floppy Disk, ~1980



Hard Disk, ~1980

Integrated Circuitry
Memory, ~1970



Compact Flash, USB, ~2000



Solid State
Drive, ~2010

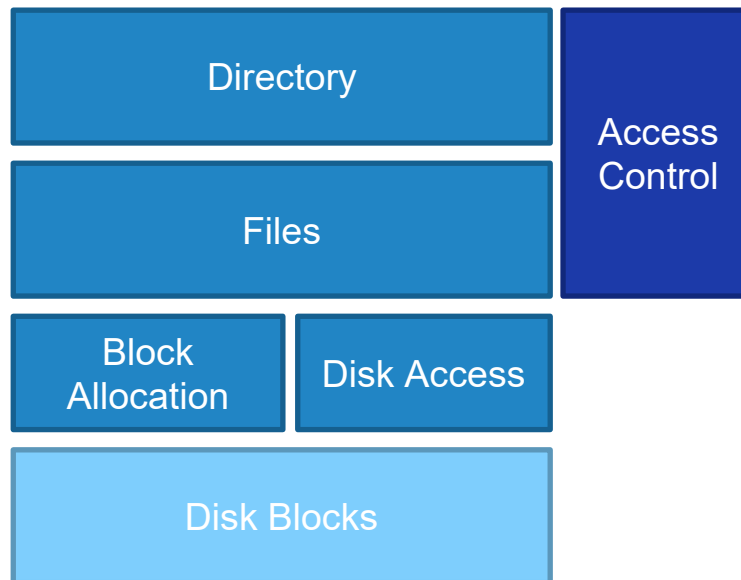
<https://www.computerhistory.org/timeline/memory-storage/>
https://en.wikipedia.org/wiki/Floppy_disk
https://en.wikipedia.org/wiki/Punched_card
https://en.wikipedia.org/wiki/Solid-state_drive

What is a file system?

- ▷ Various storage media that evolve over decades
- ▷ *Can we abstract away the media from the access interface?*
- ▷ Logical view of information
 - File, Directories
- ▷ Attributes
 - Name, type, size, access control, time
- ▷ Logical operations
 - Create, Write, Read, Delete

File System Modules

Directory module:	relates file names to file IDs
File module:	relates file IDs to particular files
Access control module:	checks permission for operation requested
File access module:	reads or writes file data or attributes
Block module:	accesses and allocates disk blocks
Device module:	performs disk I/O and buffering



Unix iNodes

Maps logical file system to physical blocks on disk

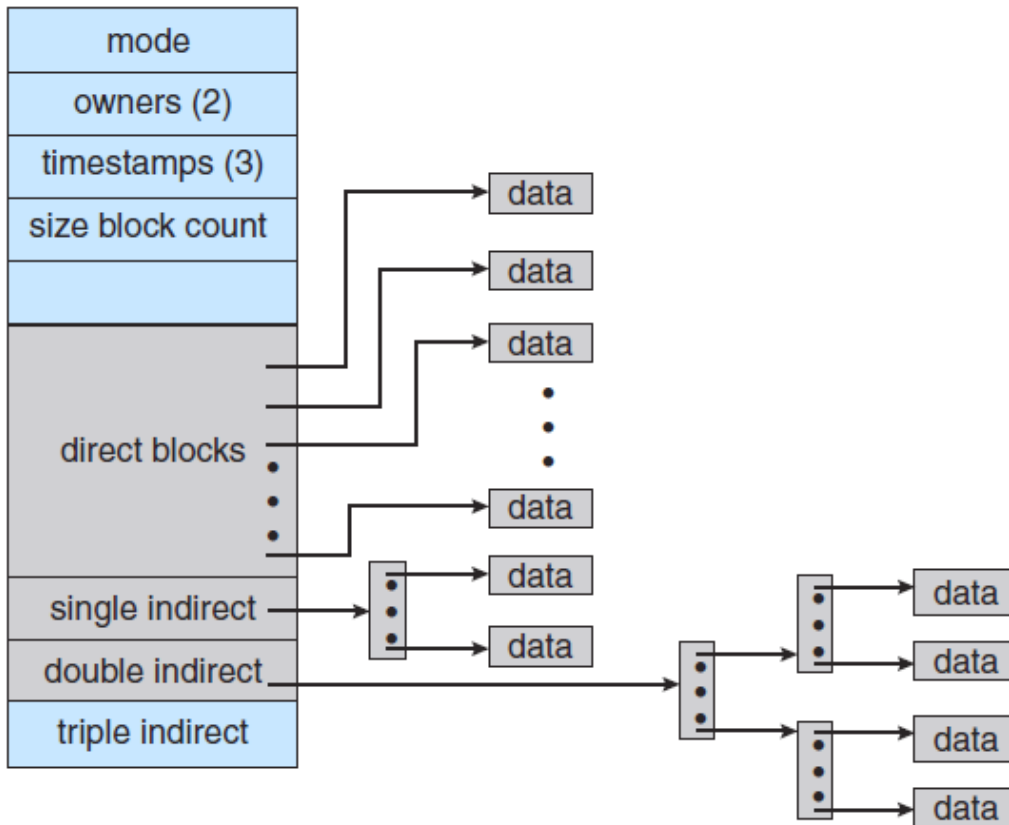
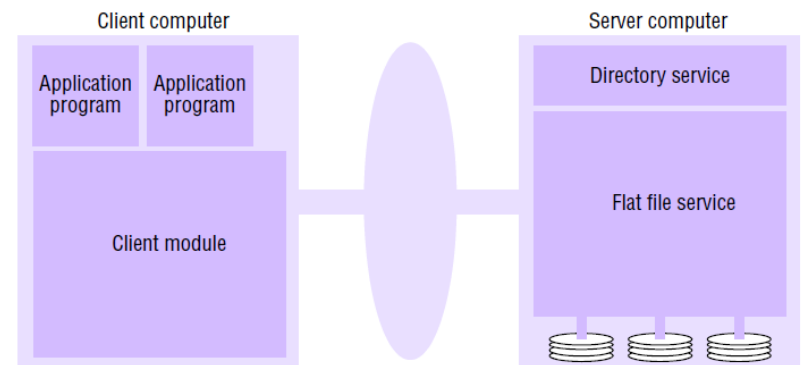


Figure 12.9 The UNIX inode.

What is a distributed file system?

- ▷ *Location where the files are stored and clients that access them can be on different machines*



- ▷ **Client-Server File System**
 - Many clients, one server with data, e.g., NFS
- ▷ **Cluster File System**
 - Many clients, Many servers in a single cluster, e.g., GFS
- ▷ **Peer to Peer File System**
 - Many clients and servers, serving both roles, e.g., Chord

Why do we need a DFS?

- ▷ Remote Access
- ▷ Performance
 - Bandwidth, Wide-area locality
- ▷ Reliability
- ▷ Capacity
- ▷ Access restrictions
 - Physical, network

Toward Exascale Systems

HDD: 100-150MB/s
SSD: 500 MB/s

Table 1. Amdahl's laws applied to various system powers.

Operations per second	RAM	Disk I/O bytes/s	Disks for that bandwidth at 100 Mbytes/s/disk	Disk byte capacity (100x RAM)	Disks for that capacity at 1 Tbyte/disk
10^9	Gigabyte	10^8	1	10^{11}	1
10^{12}	Terabyte	10^{11}	1,000	10^{14}	100
10^{15}	Petabyte	10^{14}	1,000,000	10^{17}	100,000
10^{18}	Exabyte	10^{17}	1,000,000,000	10^{20}	100,000,000

▷ Amdahl's (Other) Laws

- **Parallelism**—if a computation has a serial part S and a parallel component P , then the maximum speedup is $1/S$.
- **Balanced system**—a system needs a bit of disk I/O per second per instruction per second
- **Memory**—the RAM Mbyte capacity to CPU MIPS ratio (α) in a balanced system is 1
- **Input/output**—programs do one I/O per 50,000 instructions

Petascale Computational Systems, Bell, Gray and Szalay, IEEE Computer, 2006

GrayWulf: Scalable Clustered Architecture for Data Intensive Computing, HICSS, 2009

Rules of Thumb in Data Engineering, Jim Gray, Prashant Shenoy, International Conference on Data Engineering (ICDE), 2000

Amdahl's Balanced System Law, today

- ▷ 16 cores x 2GHz CPU
 - 32×10^9 instructions per second
- ▷ SAS-4 *cumulative* Disk controller speed: 22.5 Gbit/s
 - 22×10^9 bits per second
- ▷ 32GB RAM
- ▷ Caveats to Rules of Thumb
 - CPU architecture have a lot more complexity and pipelining
 - Sequential vs. Random access I/O workloads
 - As RAM becomes cheaper, alpha ratio is increasing from 1 to 4

DFS Characteristics

- ▷ Transparency
 - Access, Performance, Scalability
- ▷ Concurrency & Consistency
- ▷ Replication
- ▷ Fault Tolerance
- ▷ Efficiency
- ▷ Heterogeneity
- ▷ Security

Storage Systems

Figure 12.1 Storage systems and their properties

	<i>Sharing</i>	<i>Persistence</i>	<i>Distributed cache/replicas</i>	<i>Consistency maintenance</i>	<i>Example</i>
Main memory	✗	✗	✗	1	RAM
File system	✗	✓	✗	1	UNIX file system
Distributed file system	✓	✓	✓	✓	Sun NFS
Web	✓	✓	✓	✗	Web server
Distributed shared memory	✓	✗	✓	✓	Ivy (DSM, Ch. 6)
Remote objects (RMI/ORB)	✓	✗	✗	1	CORBA
Persistent object store	✓	✓	✗	1	CORBA Persistent State Service
Peer-to-peer storage system	✓	✓	✓	2	OceanStore (Ch. 10)

Big Data Sizes in the era of Deep Learning

- ▷ Recent Machine and Deep learning success is *because* we have **large datasets**
 - Artificial Neural Networks are 50+ years old
- ▷ **Netflix Prize (2006):** *Recommendation system for movies using 100M ratings for 17k movies from 500k users*
- ▷ **ImageNet (2009):** *14M images with 20k categories*
- ▷ **IBM Watson beats Jeopardy (2011):** *“5500 independent experiments of 2000 CPU hours each generating 10 GB of error-analysis data using massively parallel architecture”*
- ▷ **Google Brain recognizes Cats (2012):** *10M 200x200 pixel images, 20k classes, 1000 machines with 16k cores*
- ▷ **Facebook recognizes Humans (2014):** *4.4 M labelled faces from 4k people each with 1k samples*
- ▷ **GPT-3 for text/code/??? generation (2020):** 400B tokens from Common Crawl, 19B tokens from WebText2 , 67B tokens from books, 3B tokens from Wikipedia
- ▷ **ChatGPT (2022):** Based on InstructGPT PPO-ptx model with 175B params fine-tunes GPT-3 using RL based on **33k** human prompts.

Training the ChatGPT Model...

Step 1

Collect demonstration data and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3.5 with supervised learning.



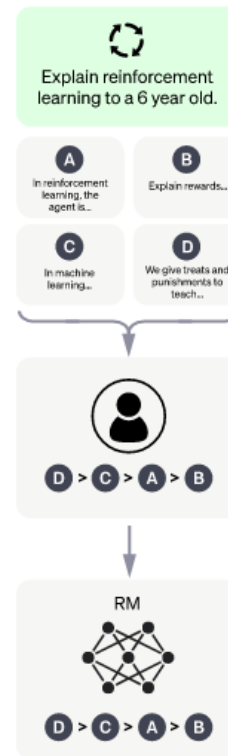
Step 2

Collect comparison data and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Step 3

Optimize a policy against the reward model using the PPO reinforcement learning algorithm.

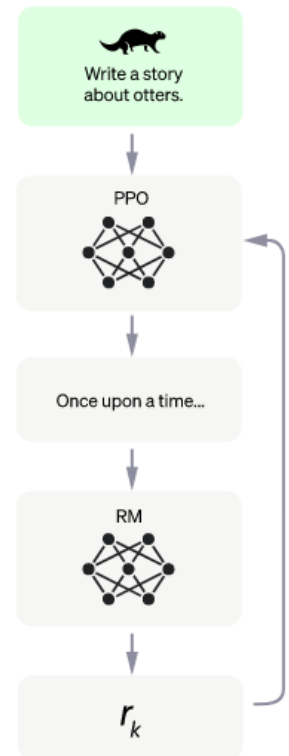
A new prompt is sampled from the dataset.

The PPO model is initialized from the supervised policy.

The policy generates an output.

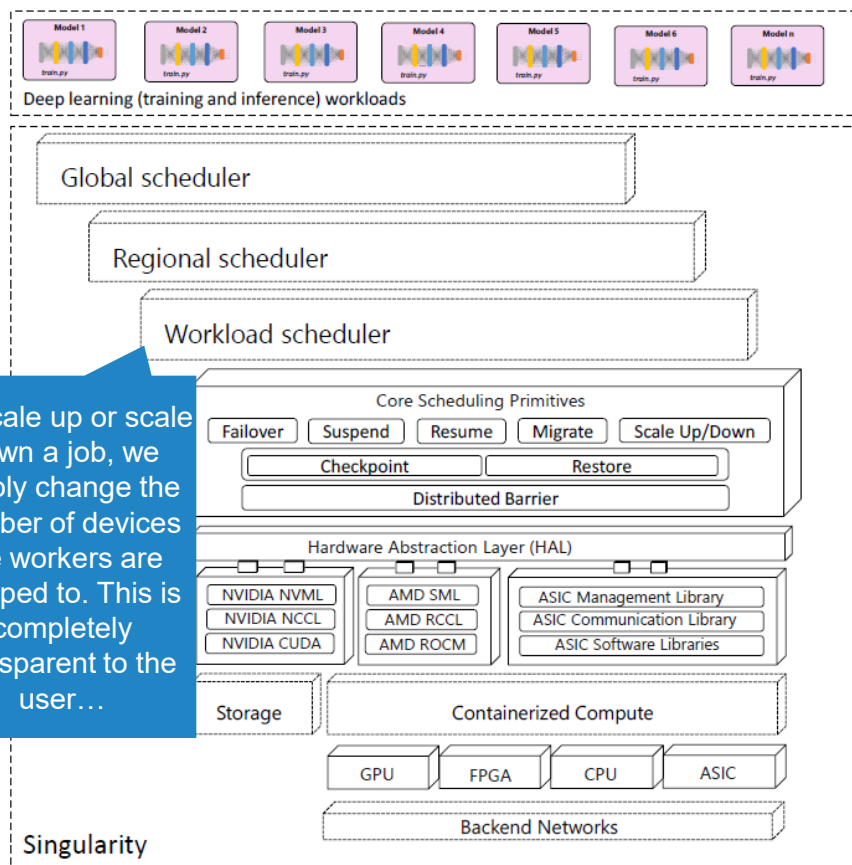
The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



<https://openai.com/blog/chatgpt/>

...on Azure “Singularity” Cloud for AI Workloads



“Microsoft invested \$1 billion in OpenAI in 2019, and in return OpenAI has built its AI models on Microsoft's Azure AI supercomputing technologies...”

<https://www.zdnet.com/article/microsoft-were-bringing-chatgpt-to-the-azure-openai-cloud-computing-service/>

Singularity: Planet-Scale, Preemptive and Elastic Scheduling of AI Workloads, arxiv, Feb 2022

- ▷ Incremental memory checkpointing of workers to storage for reliability
 - CPU and GPU
 - Used for recovery and elastic resizing
- ▷ De-duplicate CPU mem. with checksum
 - Train & loader processes across workers
 - For a worker across time

Model	User-chkpt (GB)	Singularity checkpoint (GB)					
		4-GPU			8-GPU		
		S_G	S_{Cr}	S_{Cr}^i	S_G	S_{Cr}	S_{Cr}^i
BERT	1.3	1.26	2.09	0.027	1.27	4.19	0.052
DenseNet169	0.11	0.26	4.39	2.88	0.42	8.73	5.63
PyramidNet	0.19	0.19	1.05	0.029	0.19	2.09	0.057
GPT-2*	24.2	NA	NA	NA	33.05	4.8	5.2
ResNet50	0.2	0.35	4.0	2.57	0.51	8.01	4.94
InternalQ	4.0	5.4	1.53	0.035	5.4	3.05	0.11
InternalT*	274	NA	NA	NA	162	13.1	0.19

Table 4. Checkpoint Size. Compares checkpoint sizes of Singularity with user-level checkpoints. S_G is GPU state, S_{Cr} is the size of first CRIU dump (all workers), S_{Cr}^i is size of subsequent (incremental) dumps. * on GPT-2 and InternalT indicates 32-worker configs (larger model size).

50% of latency goes to Azure Blob Storage

Model	Latency (s)					
	16-to-16		16-to-8		8-to-16	
	Total	Transfer	Total	Transfer	Total	Transfer
BERT	36	17	45	26	34	15
DenseNet169	64	40	69	45	75	51
PyramidNet	28	7	31	12	30	9
ResNet50	58	37	67	45	59	39
InternalQ	46	24	68	44	49	24
GPT-2*	72	44	122	85	97	51
InternalT*	141	98	228	165	181	121

Table 5. Latencies of migration and resizing. This table shows the end-to-end latency of migration (16-to-16), scale-up (8-to-16) and scale-down (16-to-8). (m -to- n) indicates migration from m GPUs to n GPUs. *GPT-2 and InternalT used 32 workers (we report 32-to-32, 32-to-16, and 16-to-32)



End of Lecture 3

▷ REMINDER

- *Read the GFS article before the next class*