Kia ora

KIA ORA TĀTOU

GREETINGS ALL

KO CORRAL DE TIERRA,
MONTEREY CALIFORNIA
TE MAUNGA

CORRAL DE TIERRA, MONTEREY CALIFORNIA IS THE MOUNTAIN

KO 'AU 'AU CHANNEL-
BETWEEN MAUI AND LANAI
TE AWA

'AU 'AU CHANNEL- BETWEEN MAUI AND LANAI IS THE RIVER

NŌ WEYMOUTH, UK AHAU

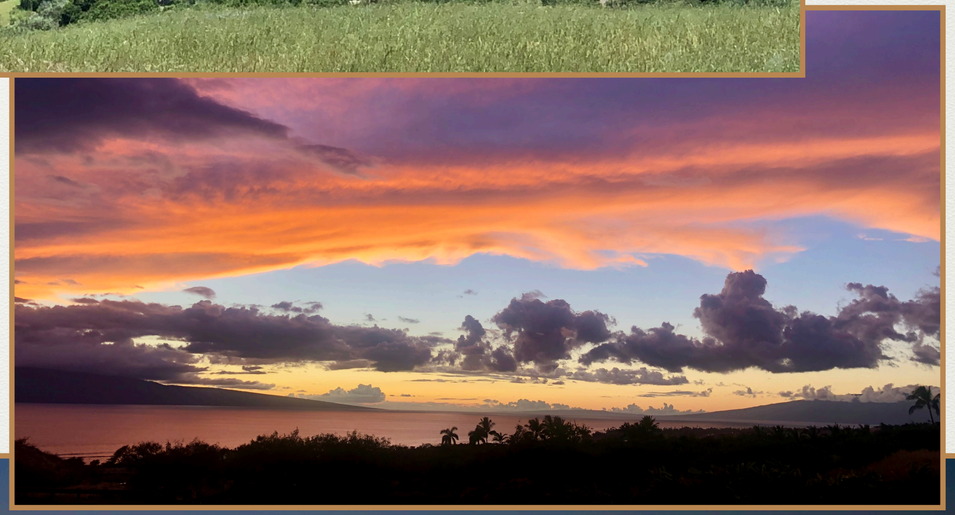
I AM FROM WEYMOUTH, UK

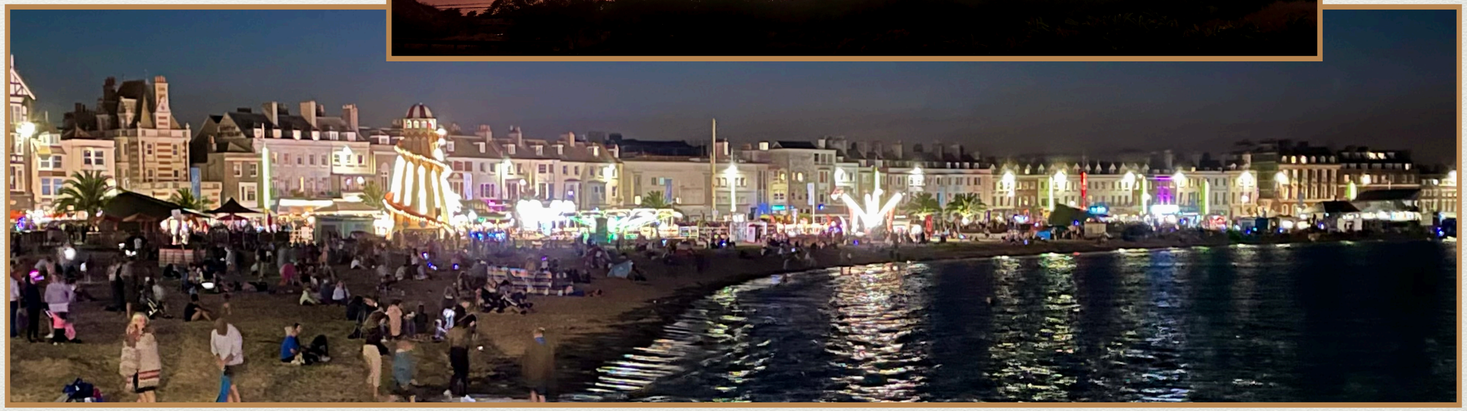
KO COCKCROFT
TŌKU WHĀNAU

COCKCROFT IS MY FAMILY

KO ADRIAN TŌKU INGOA

MY NAME IS ADRIAN





COMMUNICATING MEGABYTES IN MICROSECONDS?

An exploration into how LLM training works

Adrian Cockcroft - OrionX.net

(Illustrated with photographs taken by the author on the way here…)

Bora Bora - Most of the pictures were taken here by Adrian

Overview

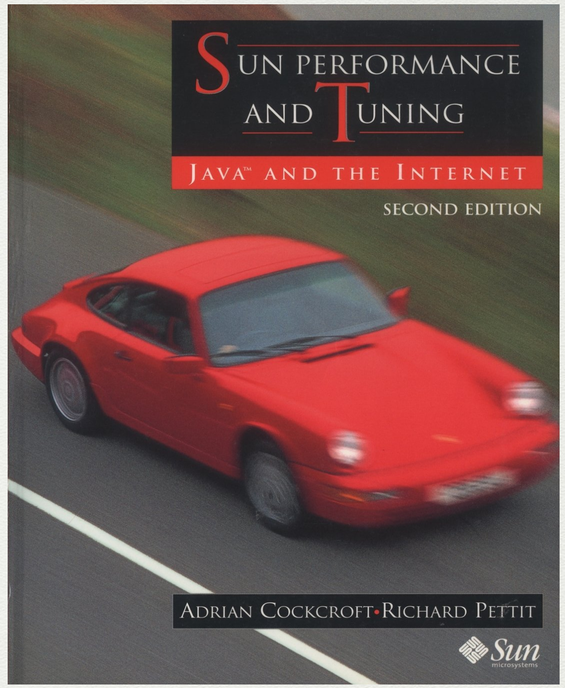
- Motivations: Why do I care about this? Why will it matter?
- How training, fine tuning, and inference workloads are structured
- What does the hardware and software stack and network look like
- How it behaves, how is it changing over time, observability
- Discussion

A curious scissor-tail sergeant fish

Why do I care?

I have a long history as a performance specialist, figuring out new workloads as they appear, helping people optimize and capacity plan for lower latency and cost.

However I'm not a hands-on user of LLMs so this talk is part of my effort to read and understand what is going on, and I'm probably missing a few things, so please point them out!



Distinguished Engineer for Sun performance team 1999-2001, Chief Architect for Sun High Performance Computing Business 2002-2004

NETFLIX | OSS

Chief Architect for Netflix transition to cloud and open source program 2010-2013 "Goal for Netflix to be the least profitable large customer for AWS"

Why will it matter?

Training and operating LLM based workloads is extremely slow and expensive.

As applications transition from research to production, efficiency, scalability and cost optimization become critical.

Big incentive to try to approach GPT-4 quality using "From Scratch" or lower cost models running on very expensive hardware.

	Quality_score x/5 (as evaluated by gpt-4)	Avg cost / request	Cost multiplier (mixtral)
gpt-3.5-turbo	3.560	$0.0027	2.87
gpt-4 (0613)	3.873	$0.0576	61.35
gpt-4-turbo (1106-preview)	4.209	$0.0230	24.47
llama-2-7b-chat-hf	3.300	$0.0003	0.31
llama-2-13b-chat-hf	3.415	$0.0005	0.51
llama-2-70b-chat-hf	3.598	$0.0019	2.04
codellama-34b-instruct-hf	3.593	$0.0019	2.01
mistral-7b-instruct-v0.1	3.441	$0.0003	0.29
mixtral-8x7b-instruct-v0.1	3.944 (3.952*)	$0.0009	1

anyscale.com cost to query a specialist RAG based LLM (Oct 2023)

Instance Size	vCPUs	Memory (GiB)	GPUs (H100)	Network Bandwidth (Gbps)	EBS Bandwidth (Gbps)	Local Storage (TB)
p5.48xlarge	192	2048	8	3200	80	8 x 3.84

AWS p5dn instance ~$100/hr

Why will it matter?

Training and operating LLM based workloads is extremely slow and expensive.

As applications transition from research to production, efficiency, scalability and cost optimization become critical.

Big incentive to try to approach GPT-4 quality using "From Scratch" or lower cost models running on very expensive hardware.

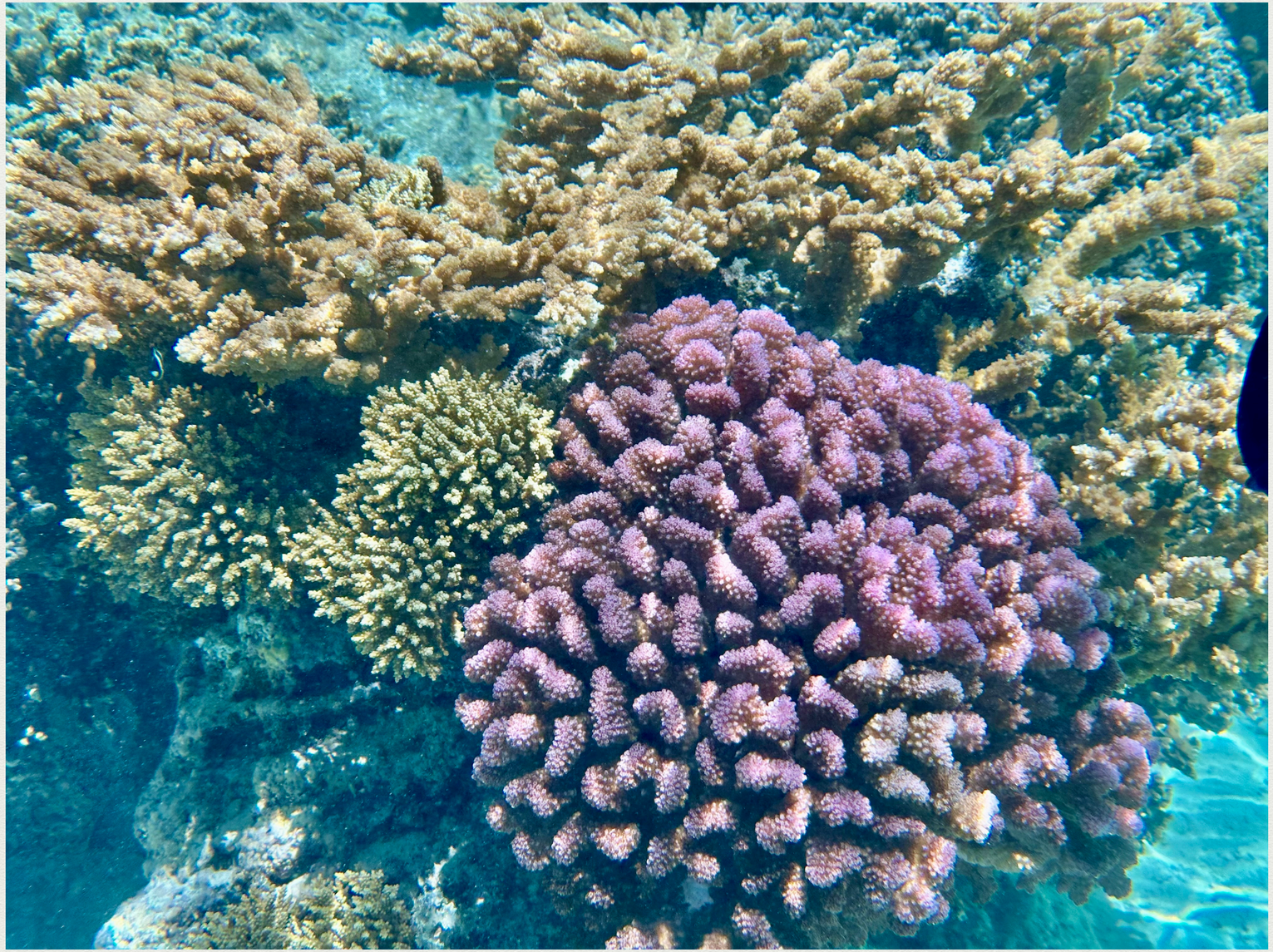
	Quality_score x/5 (as evaluated by gpt-4)	Avg cost / request	Cost multiplier (mixtral)
gpt-3.5-turbo	3.560	$0.0027	2.87
gpt-4 (0613)	3.873	$0.0576	61.35
gpt-4-turbo (1106-preview)	4.209	$0.0230	24.47
llama-2-7b-chat-hf	3.300	$0.0003	0.31
llama-2-13b-chat-hf	3.415	$0.0005	0.51
llama-2-70b-chat-hf	3.598	$0.0019	2.04
codellama-34b-instruct-hf	3.593	$0.0019	2.01
mistral-7b-instruct-v0.1	3.441	$0.0003	0.29
mixtral-8x7b-instruct-v0.1	3.944 (3.952*)	$0.0009	1

anyscale.com cost to query a specialist RAG based LLM (Oct 2023)

Instance Size	vCPUs	Memory (GiB)	GPUs (H100)	Network Bandwidth (Gbps)	EBS Bandwidth (Gbps)	Local Storage (TB)
p5.48xlarge	192	2048	8	3200	80	8 x 3.84

AWS p5dn instance ~$100/hr

3.2 Terabits! Why so much bandwidth and how is it used?



Brain Coral

- **Training foundation models**

Extremely large scale HPC style batch job on dedicated network of 10,000+ GPUs running for months to produce industry leading models like GPT4 or Llama-2.

Smaller more specialized models typically use 100s of GPUs for several days. This is where a lot of experimentation and iteration is occurring.

- **Pre-Training, Fine Tuning**

Same workload structure as training but up to hundreds of GPUs for tens of hours. Fine tuning large models needs a lot of GPU memory that drives the GPU count up.

https://llama-2.ai/llama-2-fine-tuning/
https://github.com/facebookresearch/llama-recipes/blob/main/docs/LLM_finetuning.md
https://community.juniper.net/blogs/sharada-yeluri/2023/10/03/large-language-models-the-hardware-connection

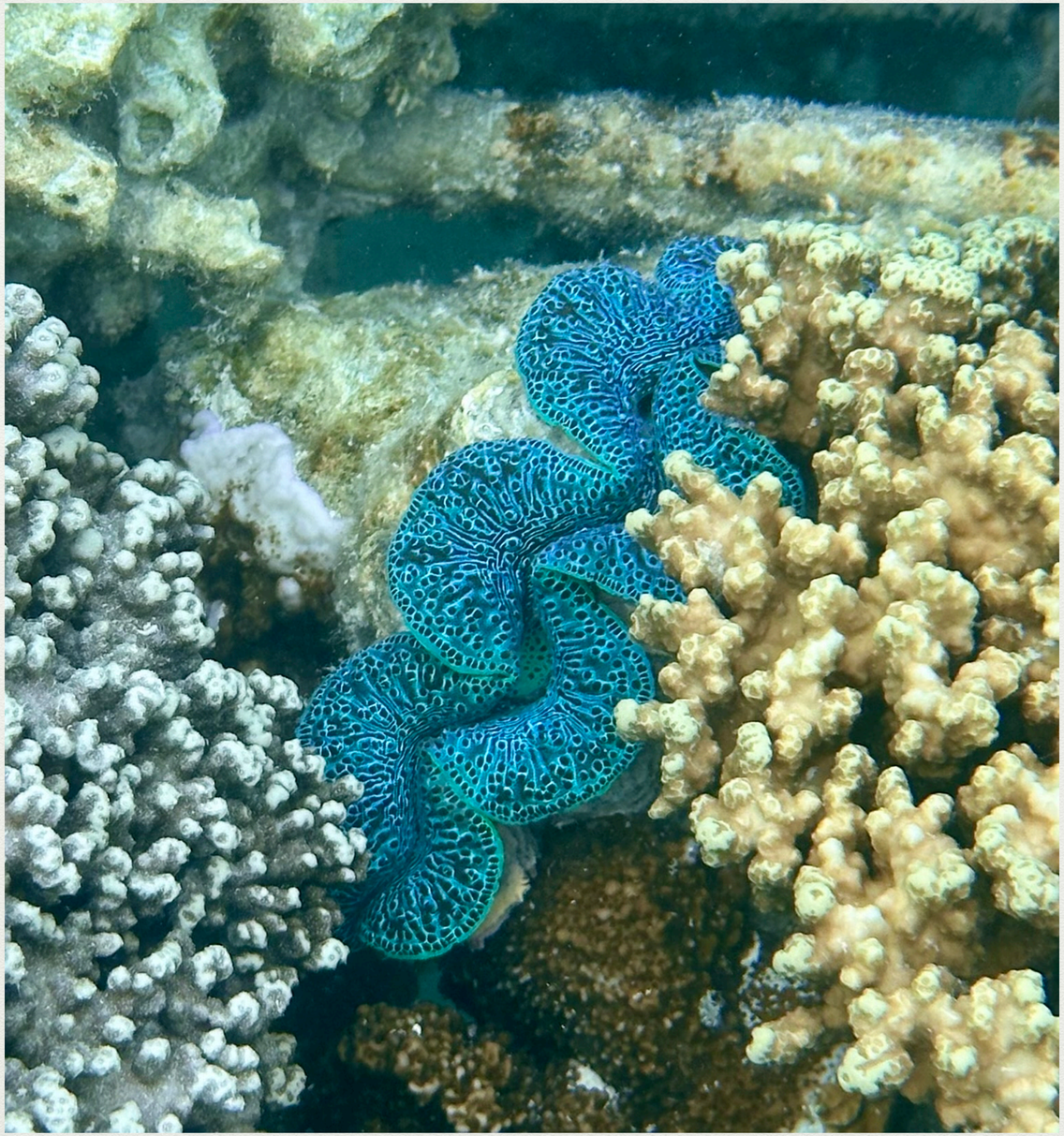
- **Retrieval Augmented Generation**

Reference materials, questions and answers, are coded and stored in a vector database that is matched and merged with the response. Most databases now support vector indexing, which is a fuzzy matching lookup.

- **Inference to support an application**

Horizontally scaled responses to users, with models tuned to reduce their size, usually running on a single GPU or a small number of GPUs in a single node, with customized prompts and retrieval augmented generation. Examples - Langchain, Ray etc.

Database lookups, and single node, not a focus for this talk.



Giant Clam Surrounded by Coral

Example: ClimateGPT by Erasmus.ai

https://arxiv.org/pdf/2401.09646.pdf

Well written paper describing all the steps taken to produce a climate expert LLM that has been trained on all the research and related documents in this area, including translation into multiple languages.

Several approaches described, including fine tuning a Llama-2 base model, and creating their own model from scratch.

Model	Base Model	Tokens	LR	GPU Hours
ClimateGPT-70B	Llama-2 70B	4.2B	$1 \cdot 10^{-5}$	2,182
ClimateGPT-13B	Llama-2 13B	4.2B	$1 \cdot 10^{-5}$	301
ClimateGPT-7B	Llama-2 7B	4.2B	$1 \cdot 10^{-5}$	157
ClimateGPT-FSC-7B	-	319.5B	$3 \cdot 10^{-4}$	14,131
ClimateGPT-FSG-7B	-	323.7B	$3 \cdot 10^{-4}$	14,288

Table 1: ClimateGPT model variants.

NVIDIA H100 GPUs, 8 per node connected by NVLINK, and 20 nodes connected via Infiniband = 160 GPUs for almost four days for the From Scratch runs with 7Billion parameters.

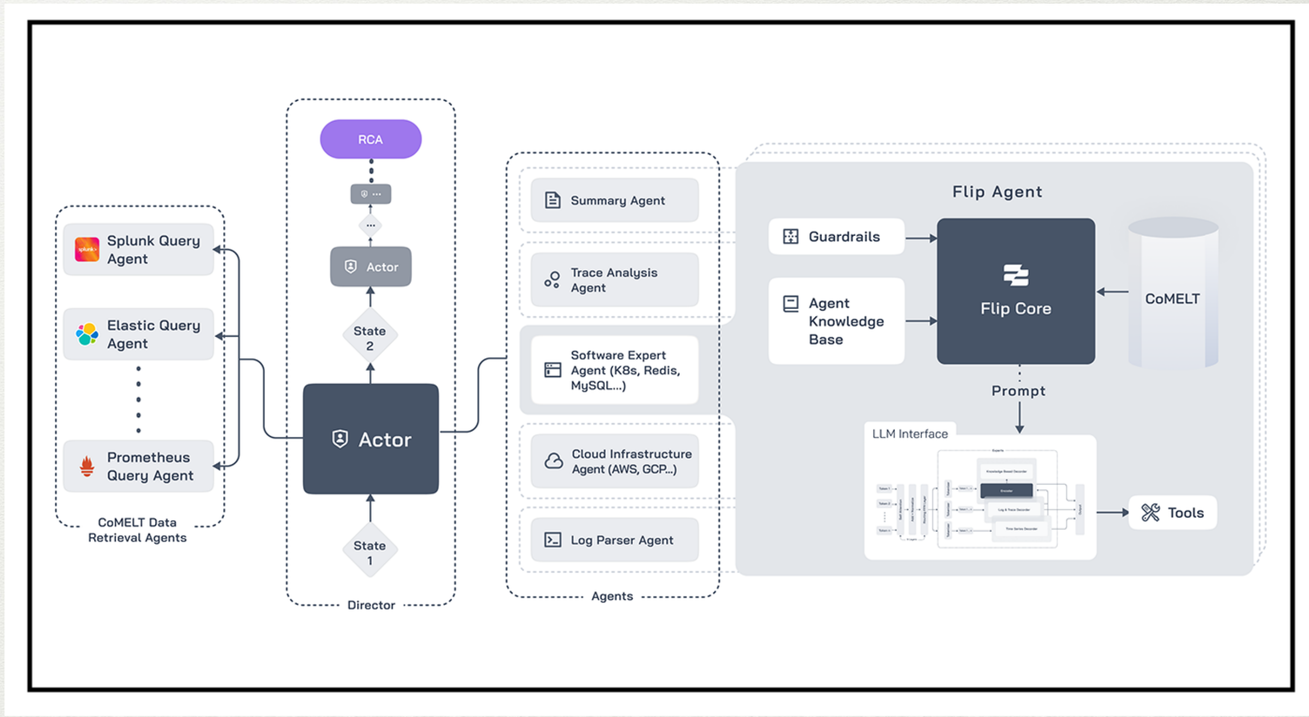
MoE Example: Flip.ai Intelligent Actors

https://www.flip.ai/blog/introducing-flip-ai-system-of-intelligent-actors

Mixture of Experts - Combining multiple specialized intelligent actors with intermediate state-stores into a "microservices-like" data flow to get better results for real-time DevOps incident analysis.

Table 1: Model Evaluation Performance Scores (normalized scale 0-1) for Categories

Category	Flip	GPT-4	Mixtral	Sample Tasks
Log Understanding	**0.77**	0.58	0.4	*Log Parsing, Classification, Generative Log NER*
Code, Bugs, Exceptions	**0.80**	0.71	0.68	*Code Explain, Exception Cause Reasoning*
Software Engineering Reasoning	0.81	**0.84**	0.68	*Software Engineering Q&A with Explanation*
Cloud DevOps Reasoning	**0.88**	0.76	0.6	*Cloud Certification, DevOps Scenario Reasoning*
Trace Summary	**0.85**	0.73	0.62	*Summarize Multiple Flame Graphs*
Database Debugging	**0.85**	0.7	0.56	*Database Performance, Slow Query Debugging*
Log & Metric Summarization	**0.95**	0.26	0.17	*Summarize RCA from Metric & Log evidence*
Metric Timeseries	**0.86**	0.45	0.33	*Summarize Cause of Spikes, Forecast Latency Spikes*



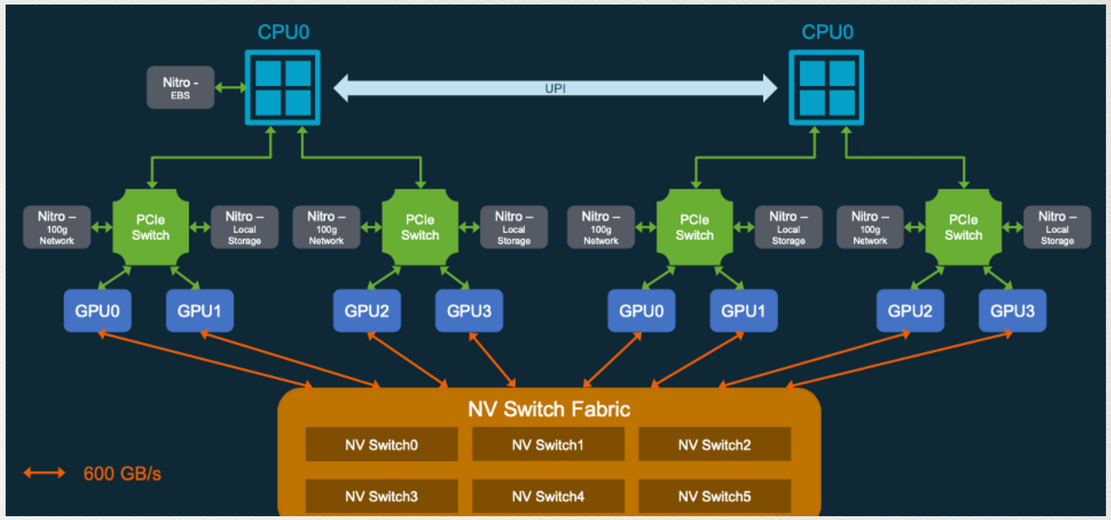
Another interesting multi-stage MoE for coding https://www.codium.ai/products/alpha-codium-v2/

Fire Coral and Goat Fish

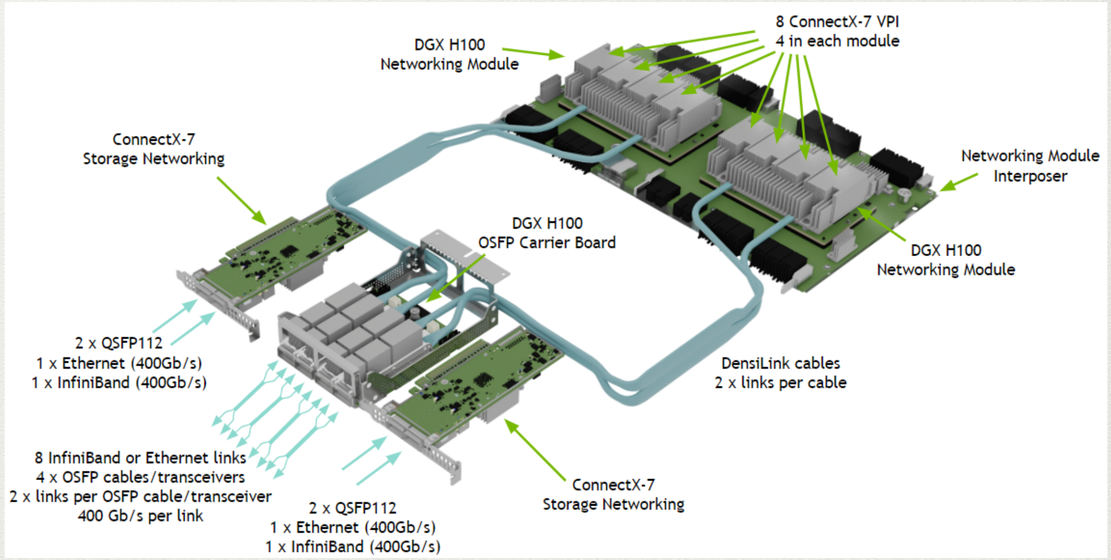
What does the hardware configuration look like?

Previous generation p4d A100 architecture on AWS uses PCI buses and Nitro for offloading 4x100Gbit/s network and virtualization.

Current NVIDIA DGX H100 architecture drives 8x400Gbit/s network directly, one per GPU



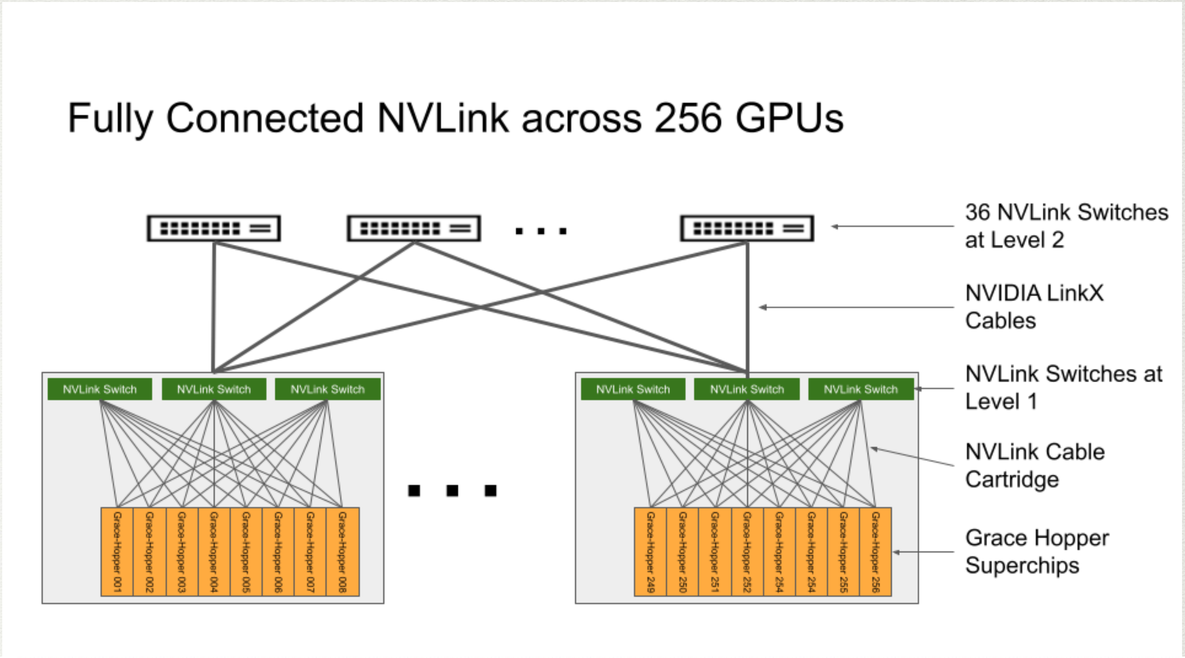
AWS p4d A100 Architecture with Nitro EFA networking over PCI
https://aws.amazon.com/blogs/compute/amazon-ec2-p4d-instances-deep-dive/

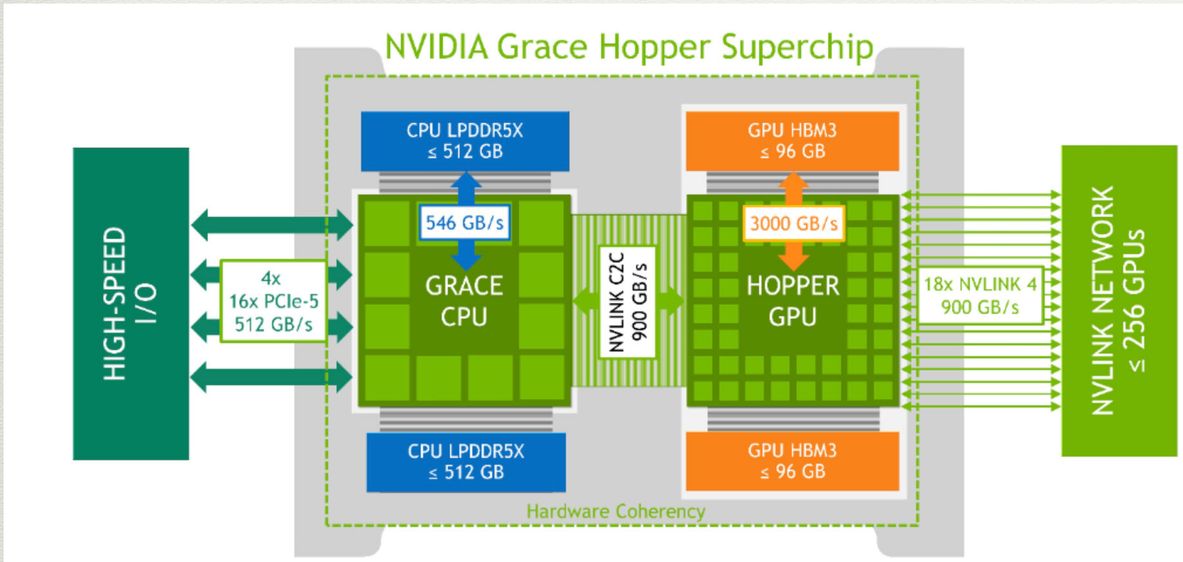


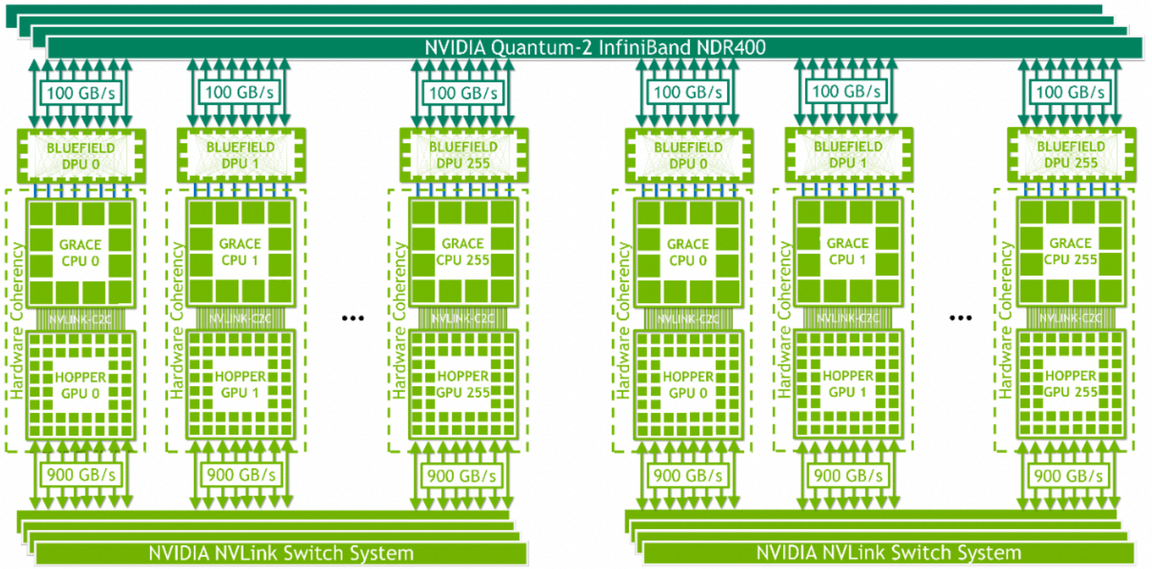
NVIDIA DGX H100 Architecture NVLINK direct access from GPU

What will the hardware configuration look like?

Next generation NVIDIA hardware increases GPUs per coherent shared memory node from 8 to 256. CXL promises to do this eventually for everyone else... see my Petalith blog post for more on this.







https://medium.com/@adrianco/supercomputing-predictions-custom-cpus-cxl3-0-and-petalith-architectures-b67cc324588f

How is work scheduled?

Pipeline, tensor, and data parallelism, a technique called PTD-P

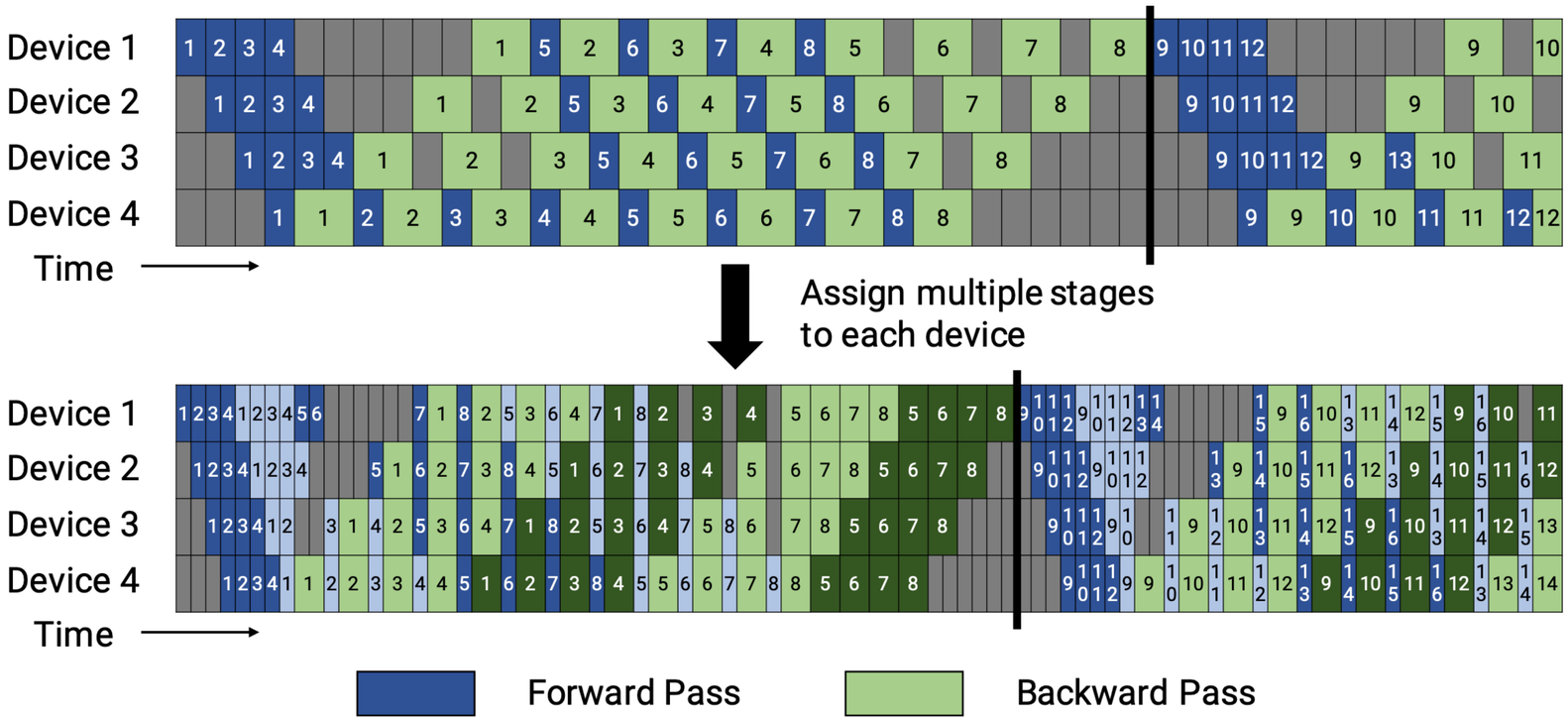


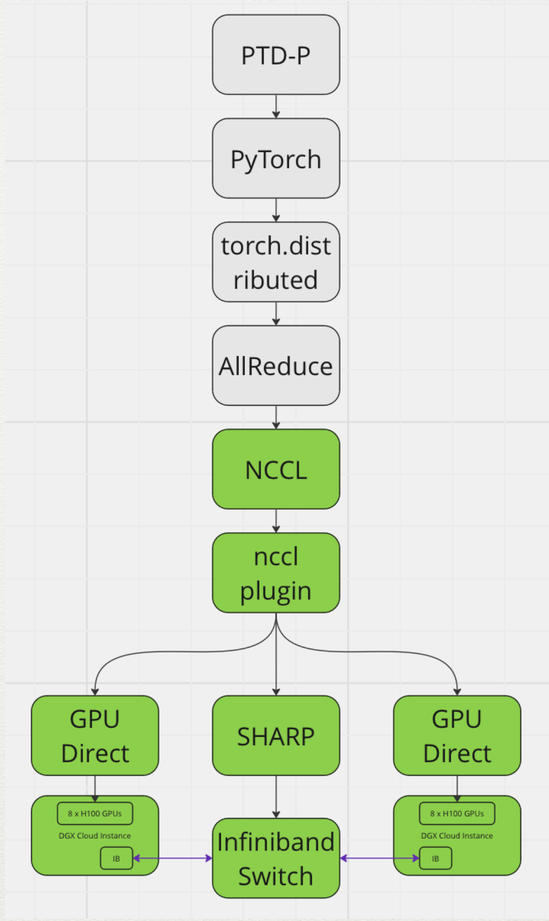
Figure 4: Default and interleaved 1F1B pipeline schedules. The top figure shows the default non-interleaved 1F1B schedule. The bottom figure shows the interleaved 1F1B schedule, where each device is assigned multiple chunks (in this case, 2). Dark colors show the first chunk and light colors show the second chunk. The size of the pipeline bubble is smaller (the pipeline flush happens sooner in the interleaved timeline).

Efficient Large-Scale Language Model Training on GPU Clusters Using Megatron-LM. https://arxiv.org/pdf/2104.04473.pdf

What does the networking software stack look like?

Top level code written in Python, using PyTorch, can be deployed as a container on Kubernetes, using some bypass optimization.

NVIDIA Mellanox InfiniBand SHARP improves the performance of collective operations by processing data aggregation and reduction operations as it traverses the network, eliminating the need for sending data multiple times between endpoints. It's also implemented in the NVSwitch connecting the latest GPU models, 400GFlops per switch chip.



NeMo Framework Training Containe

Software Component	Version
Transformer Engine	v1.1
PyTorch	2.1.0
NeMo	1.22.0
Megatron Core	0.4.0
PyTorch Lightning	2.0.7
Hydra	1.2.0
Kubernetes	1.27.4
Helm	3.12.1
GPU Operator	23.3.2
Network Operator	23.1.0
KubeFlow Operator	1.6.0

https://developer.nvidia.com/blog/upgrading-multi-gpu-interconnectivity-with-the-third-generation-nvidia-nvswitch/

Efficient Large-Scale Language Model Training on GPU Clusters Using Megatron-LM https://arxiv.org/pdf/2104.04473.pdf

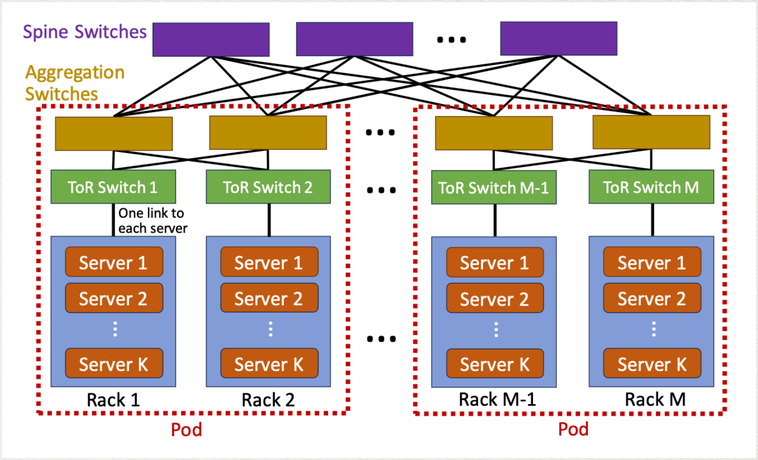
Coral Gardening

What does the network topology look like?

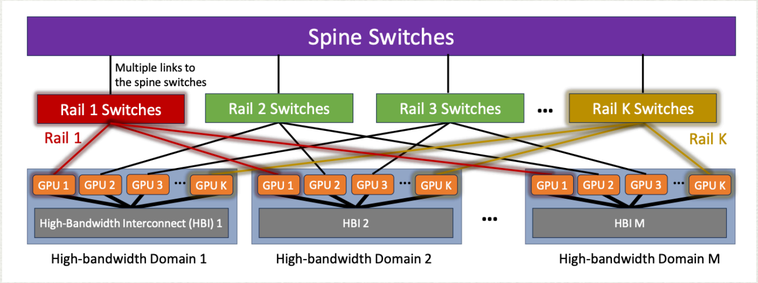
Conventional datacenter Clos topologies don't work well for LLM training. Very specific traffic patterns occur that can be optimized by Rail Only connections, forwarding other traffic indirectly.

# of GPUs (N)	Switch Radix	SOTA #SW	Rail-only #SW	SOTA #TR	Rail-only #TR	Cost Reduc-tion
32768	32	7168	3072	262144	131072	54%
	64	2560	1536	196608	131072	37%
	128	1280	256	196608	65536	75%
	256	384	128	131072	65536	60%
65536	64	5120	3072	393216	262144	37%
	128	2560	1536	393216	262144	37%
	256	1280	256	393216	131072	75%

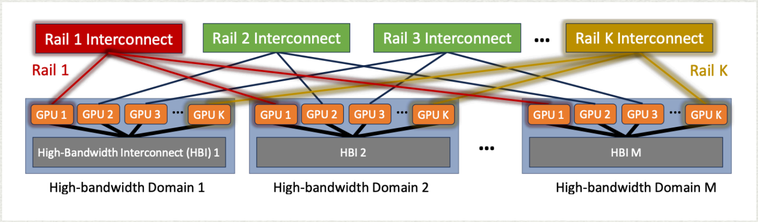
Clos



SOTA Rail and Spine



Rail Only



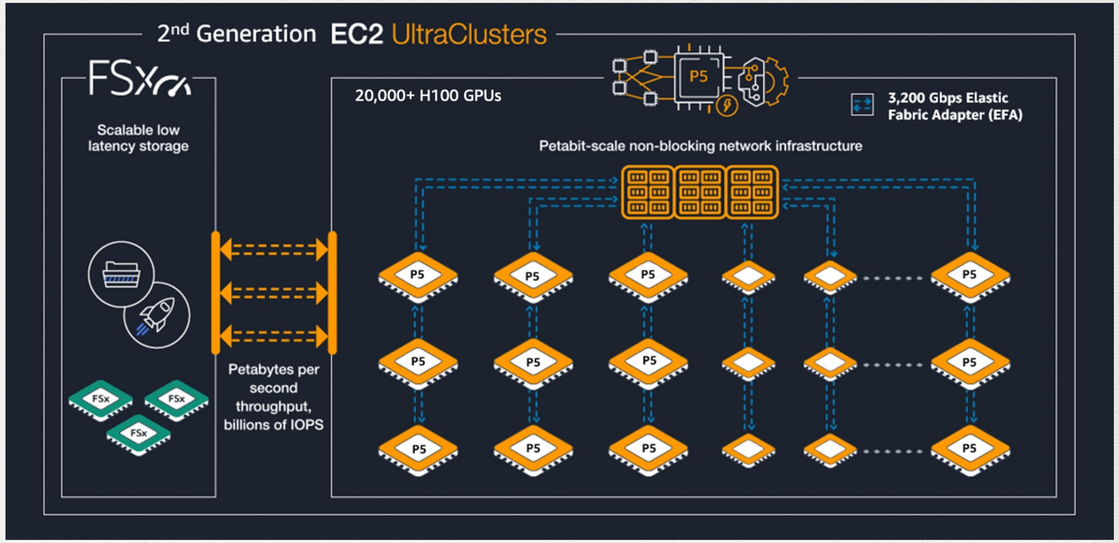
Optimized Network Architectures for LLM Training with Billions of Parameters https://arxiv.org/pdf/2307.12169v2.pdf

What does AWS network topology look like?

EFA Non-blocking petabit scale multipath optimized, no head-of-line blocking, fast internal retries using Scalable Reliable Datagram (SRD) via Libfabric over ethernet.

Seems generally useful for HPC workloads, but not optimal for LLM rail based traffic pattern?

AWS recently announced direct support for current NVIDIA DGX hardware, and a future integration of Nitro with DGX for 16384 x GH200 "Project Ceiba", perhaps that's an optimization for this specific workload?



Ceiba could be about 1 Exaflop if NVIDIA submit a Top500 entry?

https://aws.amazon.com/blogs/hpc/second-generation-efa-improving-hpc-and-ml-application-performance-in-the-cloud/

https://press.aboutamazon.com/2023/11/aws-and-nvidia-announce-strategic-collaboration-to-offer-new-supercomputing-infrastructure-software-and-services-for-generative-ai

https://nvidianews.nvidia.com/news/aws-nvidia-strategic-collaboration-for-generative-ai

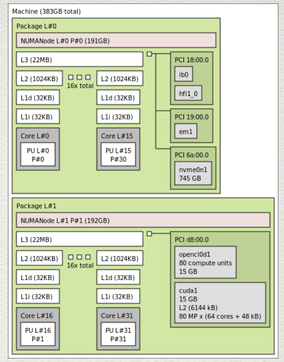
Coral

Behaviors and Observability

"At scale, distributed training is communication-intensive. When training a trillion-parameter model on 3072 GPUs, our implementation used an effective bisection bandwidth of 892 GB/s for pipeline-parallel communication, and 13 TB/s for data-parallel communication."

Hardware topology database - HWLOC

https://www.open-mpi.org/projects/hwloc/



https://swsmith.cc/posts/efa-best-practices.html

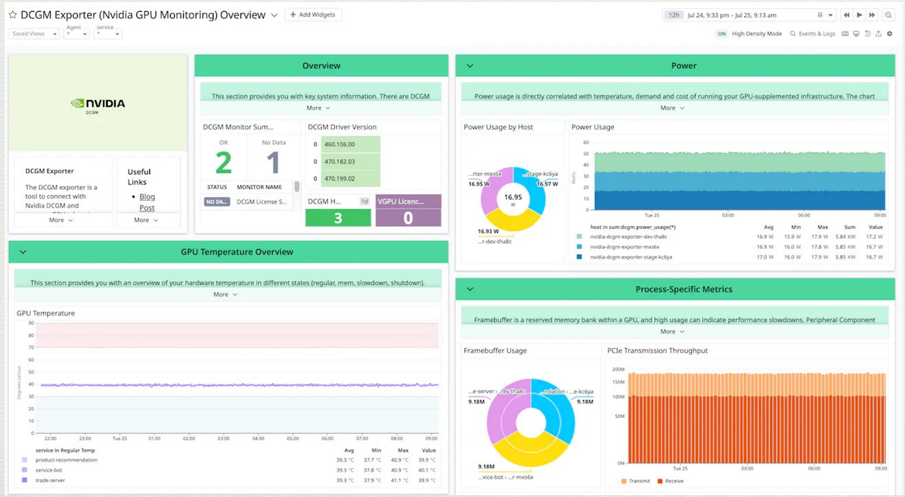
alinux2

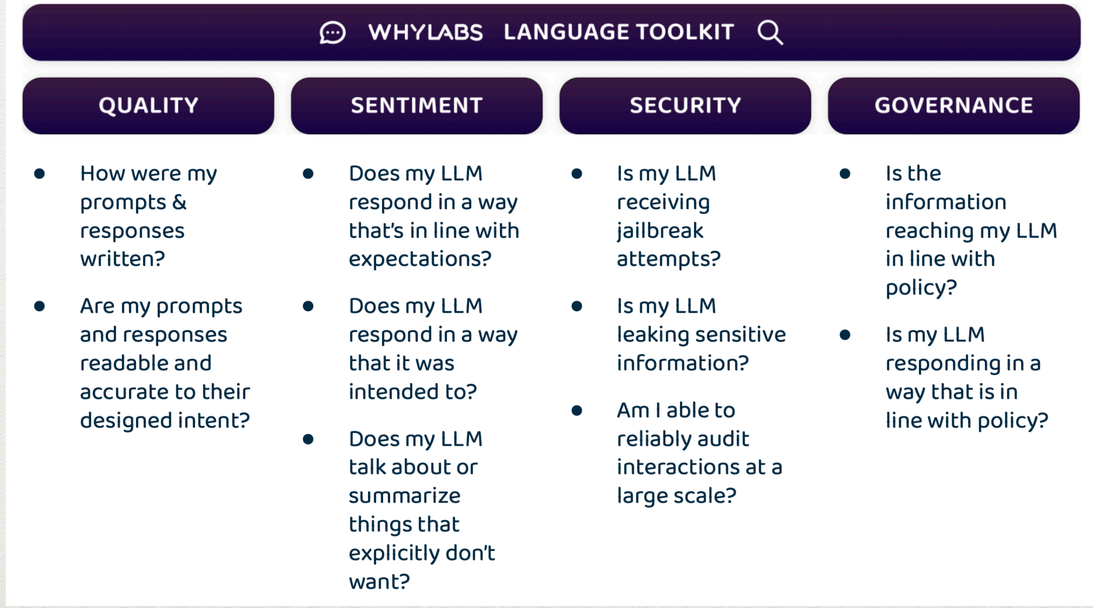
SHELL

```
cat /sys/class/infiniband/rdmap0s6/ports/1/hw_counters/tx_pkts
cat /sys/class/infiniband/rdmap0s6/ports/1/hw_counters/rx_pkts
```

Bernease Herman https://www.youtube.com/watch?v=Gxs6VpP3Sww

https://www.datadoghq.com/monitoring/nvidia-gpu-monitoring/





Sergeant Fish

Discussion

adrian.cockcroft@orionx.net
@adrianco@mastodon.social

Miro Board Notes

https://bit.ly/llm-observability

	Quality_score x/5 (as evaluated by gpt-4)	Avg cost / request	Cost multiplier (mixtral)
gpt-3.5-turbo	3.560	$0.0027	2.87
gpt-4 (0613)	3.873	$0.0576	61.35
gpt-4-turbo (1106-preview)	4.209	$0.0230	24.47
llama-2-7b-chat-hf	3.300	$0.0003	0.31
llama-2-13b-chat-hf	3.415	$0.0005	0.51
llama-2-70b-chat-hf	3.598	$0.0019	2.04
codellama-34b-instruct-hf	3.593	$0.0019	2.01
mistral-7b-instruct-v0.1	3.441	$0.0003	0.29
mixtral-8x7b-instruct-v0.1	3.944 (3.952*)	$0.0009	1

Anyscale.com deployment costs for LLM comparison
Ray RAG expert
https://www.anyscale.com/blog/a-comprehensive-guide-for-building-rag-based-llm-applications-part-1#cost-analysis

MegaTron 2020 Paper
https://arxiv.org/pdf/1909.08053.pdf

MegaTron 2021 Paper
https://arxiv.org/pdf/2104.04473.pdf

NVIDIA Launcher

NeMo Megatron

Nephele

Slurm

Pyxis

Enroot

NFS

FSx Lustre

AWS Launcher

Parallel Cluster

AWS Operation
https://docs.nvidia.com/nemo-framework/user-guide/latest/cloudserviceproviders.html#aws

https://github.com/aws-samples/awsome-distributed-training/tree/main/3.test_cases/2.nemo-launcher

Observability

Monitoring LLMs: what is new:
https://www.perplexity.ai/search/what-is-new-rTCSdjBnTnieoUK0TnDIbg?s=c

What is Langsmith?
https://docs.smith.langchain.com/

LLMOps Tools:
https://research.aimultiple.com/llmops-tools/

Why Model Monitoring is Crucial for LLMs
https://www.linkedin.com/pulse/model-monitoring-age-large-language-models-llms-frank-la-vigne/

LLM Inference Langchain and Whylabs with Bernease Herman Video
https://www.youtube.com/watch?v=GpsEKjqPJSww
Slides
https://docs.google.com/presentation/d/1aEgCH3huNBl_DBZTO-00VQxcIw7fBkQWc3WRPBd7awIe/edit#slide=id.g258f1905564_0_643

Cost optimization Infery-LLM from deci.ai
https://deci.ai/blog/how-to-reduce-llm-inference-cloud-cost/

DataDog Monitoring GPUs and LLMs
https://www.datadoghq.com/monitoring/nvidia-gpu-monitoring/

RWKV Eagle 10-100x faster inference, open source
https://blog.rwkv.com/p/eagle-7b-soaring-past-transformers

Enroot
Import a Docker container but optimize it for HPC operations by removing some of the isolation
https://github.com/NVIDIA/enroot

Erasmus.ai ClimateGPT Paper
https://arxiv.org/pdf/2401.09646.pdf

Pyxis
Plugin for Slurm that uses Enroot to run an HPC job
https://github.com/NVIDIA/pyxis

NeMo Framework User Guide
https://docs.nvidia.com/nemo-framework/user-guide/latest/index.html

NeMo Megatron Launcher
https://github.com/NVIDIA/NeMo-Megatron-Launcher

NeMo Framework Training Containe

Software Component	Version
Transformer Engine	v1.1
PyTorch	2.1.0
NeMo	1.22.0
Megatron Core	0.4.0
PyTorch Lightning	2.0.7
Hydra	1.2.0
Kubernetes	1.27.4
Helm	3.12.1
GPU Operator	23.3.2
Network Operator	23.1.0
Kubeflow Operator	1.8.0

Network Traffic Patterns

Pipeline, tensor, and data parallelism, a technique we call PTD-P
At scale, distributed training is communication-intensive. When training a trillion-parameter model on 3072 GPUs, our implementation used an effective bisection bandwidth of 892 GB/s for pipeline-parallel communication, and 13 TB/s for data-parallel communication.

Network traffic analysis for PTD-P
"Optimized Network Architectures for Large Language Model Training with Billions of Parameters"
https://arxiv.org/pdf/2307.12169.pdf
https://arxiv.org/pdf/2307.12169v2.pdf

NVIDIA GPUDirect
https://developer.nvidia.com/gpudirect

NVIDIA Collective Communications Library
https://developer.nvidia.com/nccl
Video 2023
https://www.nvidia.com/en-us/on-demand/session/gtcspring23-s51111/

NCCL Test
common.cu/main()
run()
benchTime()
startColl()
sendRecvStartColl()
ncclSend()
ncclGroupEnd()
ncclGroupEndInternal()
nccl-net plugin

NCCL Test/benchmark code in C
https://github.com/NVIDIA/nccl-tests/blob/master/src/common.cu

LLM Training Stack

PTD-P

PyTorch

torch.distributed

AllReduce

NCCL

nccl plugin

GPU Direct

SHARP

GPU Direct

Infiniband Switch

NVIDIA SHARP - Scalable Hierarchical Aggregation and Reduction Protocol
https://developer.nvidia.com/blog/accelerating-io-in-the-modern-data-center-computing-and-io-management/

AWS Networking

AWS p5 instances
https://aws.amazon.com/blogs/aws/new-amazon-ec2-p5-instances-powered-by-nvidia-h100-tensor-core-gpus-for-accelerating-generative-ai-and-hpc-applications/

AWS plugin for NCCL
https://github.com/aws/aws-ofi-nccl

Open Fabric Interface Working Group Libfabric
https://github.com/ofiwg/libfabric

AWS efa plugin for libfabric
https://github.com/ofiwg/libfabric/tree/main/prov/efa/src

HWLOC Portable Hardware Locality (includes CXL)
https://www.open-mpi.org/projects/hwloc/

LTTNG Linux Trace Toolkit Next Generation
https://lttng.org/docs/v2.13/#doc-viewing-and-analyzing-your-traces

Intro to EFA slides from 2019
https://pages.awscloud.com/rs/112-TZM-766/images/2019_0324-CMP_Slide-Deck.pdf

Fine Tuning LLaMA 2
https://github.com/facebookresearch/llama-recipes/blob/main/docs/LLM_finetuning.md

Observability and Monitoring LLMs
Bijit Ghosh - CTO at Deutsche Bank
https://medium.com/@bijit211987/llm-observability-and-monitoring-925f93242ccf

https://medium.com/@bijit211987/ways-to-monitor-llm-behavior-c068fba53932