Sia: Heterogeneity-aware, goodput-optimized ML-cluster scheduling

by Suhas J. Subramanya @CMU et al.

.

.

Presenter: Ke Siyun (Garfield)*

Contact: garfield.ke@connect.um.edu.mo

(mc35080)

* Assigned collaborator withdrawn from this course.

Slides

Sia: Heterogeneity-aware, goodput-optimized ML-cluster scheduling

- Introduction
- Background details and related work
- Sia design and implementation
- Experiments
- Conclusion

Slides

Introduction

Scheduling of deep learning (DL) clusters:

- Multiple users submit jobs to DL clusters to train their models.
- A scheduler assigns resources (i.e. GPU time) to jobs.
- Current clusters might consist of mixed types of GPUs
  - (Homogenous – same type in a cluster)
  - **(Heterogenous – mixed types)**

Overview of Sia

What is Sia?

- A deep learning (DL) clusters' scheduler.
- Outperforms (at least matches) representative schedulers.

Features of Sia?

- Match jobs (configurations) with resources (GPU)
- Adaptable to changes (e.g. batchsize)
- Support elastic scaling of "hybrid" parallel jobs

Key features

Sia- a new scheduler designed for DL clusters that are BOTH...

- **Heterogeneous** *-containing different types of GPUs, and
- **Resource-adaptive** *-able to adjust the resources allocated to jobs dynamically.

(* More details following)

to optimize the *goodput* of these DL clusters, while original schedulers usually only focus on one of the criteria.

Goodput:
A measure of useful work done.

Background details and related work

Training a deep neural network (DNN)...

- Iterate through epochs with the dataset, in each epoch/ minibatch in epochs,
- minimizes loss function over the minibatch of samples.
- updates model parameters (optimizer)

The DL jobs are usually able to be **parallelized** across multiple GPUs on a single/multiple node

Parallelization of DL jobs

Most training jobs use *synchronous data parallelism (DP)*

- a set of GPUs,
- each GPU receives a model replica,
- computes gradients on a partition of the minibatch(local batch size).
- gradients reduced on all GPUs (synchronizes)

Some jobs use *model parallelism*

- when the model being trained is too large to fit in a single GPU’s memory

(Other strategies exist)

*Elastic and resource-adaptive DL jobs

Elastic Resizing & Adaptive Job :

- Data-parallel DL jobs (minibatch sizes) can be **resized** over time.
- Achieved by checkpointing and restarting on a different number of GPUs.
- Jobs can adapt to assigned resources.

  Example: Minibatch size can be increased with more GPUs.
- Different minibatch sizes have different impacts.

  increaed per-GPU compute and scalability.

*Resource heterogeneity

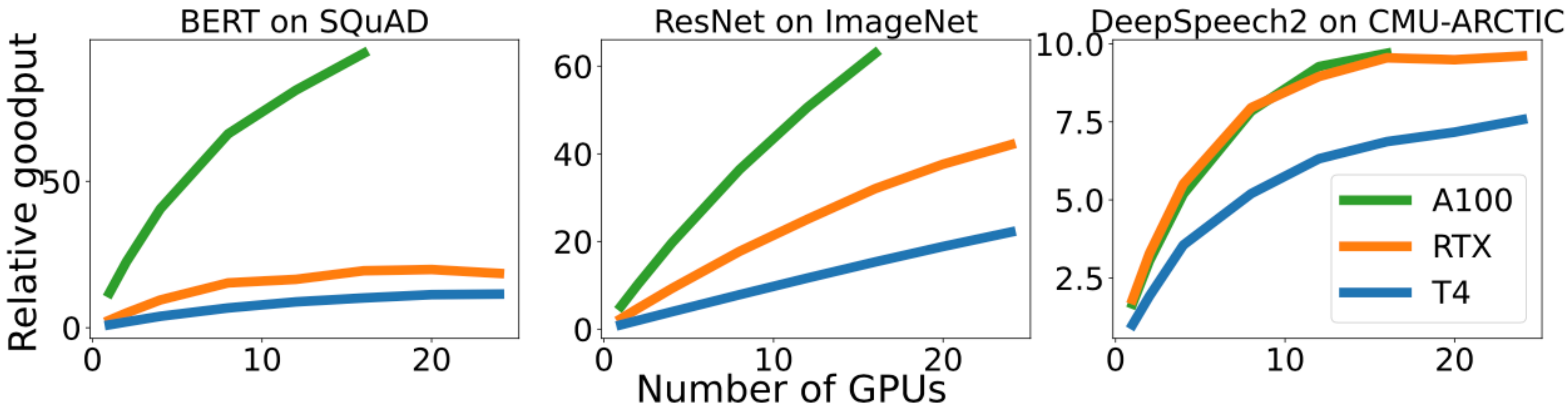
Different types of GPUs might be deployed to a cluster.

(e.g. the cluster upgrades meet the rapid advancement of GPUs)

(may be differed in memory size, performance etc.)

DL jobs might perform differently over varied GPU types.

(figure: different speedups/scalability of 3 DL jobs on 3 types of GPUs)



*Current DL cluster schedulers -example

Heterogeneity-aware schedulers

(Gavel, as state of art example)

- Consider differences among GPU **types**
- Run the jobs with user-specified **number** of GPUs ("*Rigid*" jobs)

- No elastic scaling, not adaptable to resource assignments.

Adaptivity-aware schedulers

(Pollux)

- Assume the cluster is deployed with same **type** of GPUs
- Adaptable with **number** of GPUs involved ("No-*rigid*" jobs)

- Allows adjustment in resources (e.g. specify different batchsize ...)

Current DL cluster schedulers

Heterogeneity-aware schedulers (Gavel)

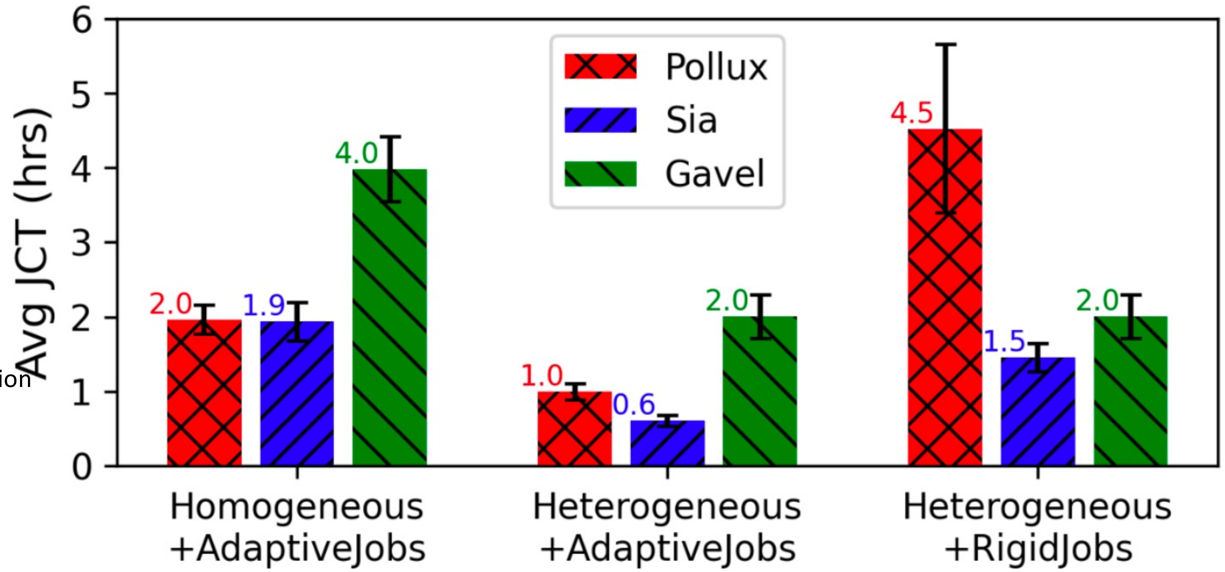
RIGHT✅ Good with rigid jobs on **<u>hetero</u>**geneous cluster (3 GPU types)

Adaptivity-aware schedulers (Pollux)

LEFT✅ Good with non-rigid jobs on **<u>homo</u>**geneous cluster

MIDDLE❌ Both not perfect with non-rigid jobs on **<u>hetero</u>**geneous cluster (still faster?)

JCT-
Job
Completion
Time



DL cluster schedulers summary

Many DL cluster schedulers only accommodate fix **number** of GPUs, not

- elasticity (**resizing**),
- resource-**adaptivity**,
- and heterogeneity (**types**)

Sia is designed to solve the issues.

Sia design and implementation

For Sia's workflow to...

- considers every possible assignment of GPUs (number and type)
- selects the best resource assignment

which is challenging because,

- the search space is huge, and
- profiling all possible job-allocations is prohibitively expensive,

Features

‚Sia introduces a new **scheduling approach**...

- ILP formulation
- efficiently manage the large search space of possible resource assignments
- addresses both the GPU types and numbers* (*job adaptability).

And to find optimized configurations (per-job && per-GPU-type throughput models), Sia ...

- at first bootstraps from observing **a few mini-batches**
- effectively refine as the job runs.

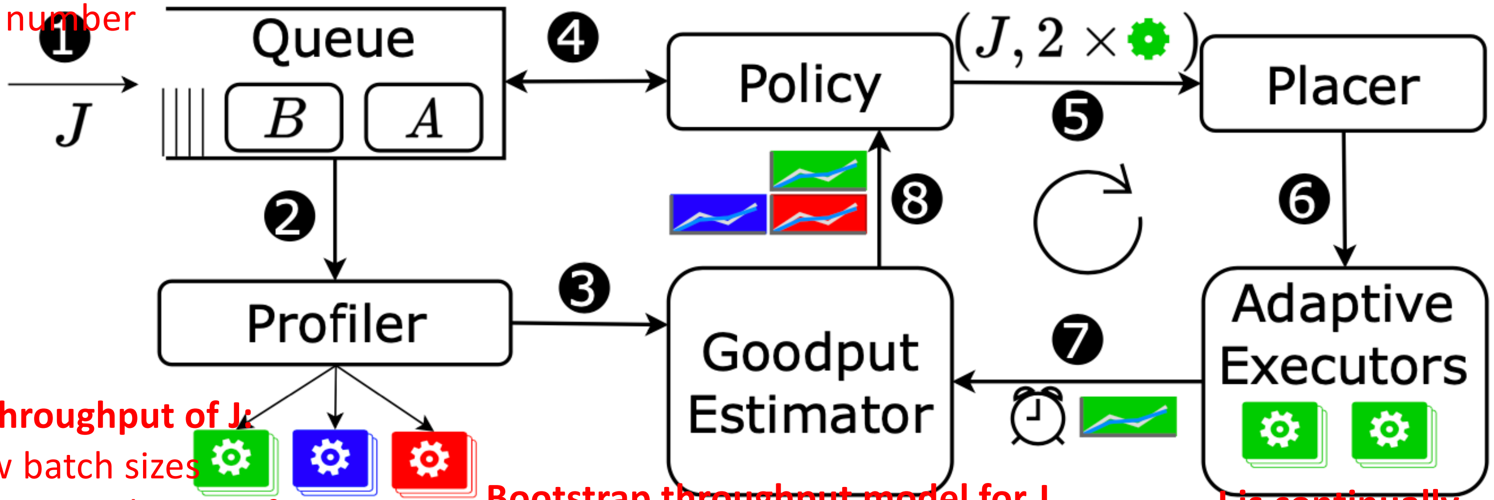
Job lifecycle

J declares its demands:

1- max batch size

2- max GPU number

J stays in queue until allocated GPUs



Sia "profile" the throughput of J:

1- using a few batch sizes

2- attempting on each type of GPUs

Bootstrap throughput model for J

(on each each type of GPUs)

J is continually monitored/optimized

Bootstrap throughput model

- Sia obtained the throughput of a Job on ONE(1) GPU of a new type (says, type B).
- How to estimate throughput of such Job on multiple (N) type B GPUs?

$$\text{est-xput}_B(N) = \frac{\text{xput}_B(1)}{\text{xput}_A(1)} * \text{xput}_A(N)$$

Used a ratio to a known type of GPU

Experiments shows this kind of bootstrap is accurate enough

Implementation

Open-source AdaptDL framework AdaptDL

https://github.com/petuum/adaptdl

- *Resource-adaptive* DL training & scheduling framework
- Claimed efficient resource management and lower training time compared to others

Experiments

Sia is compared with state-of-the-art schedulers in

- both homogeneous and heterogeneous clusters
- using real-world workloads

Experimental workloads

Real world workloads used for evaluation:

- Philly
  - 100k jobs executed over two months
  - multiple GPU types at Microsoft
- Helios
  - 3.3M jobs over six months
  - heterogeneous cluster, 6k GPUs

...

Hardware

4 types of GPUs used to build the experiment environment (physical + simulated **Homogeneous/Heterogeneous** clusters)

- `t4` – [Cloud] `g4dn.12xlarge` AWS EC2 instance with 4 NVIDIA T4 (16GB VRAM) GPUs.
- `rtx` – [On-prem] *commodity* node with 8 NVIDIA RTX 2080Ti (11GB VRAM) GPUs and 50Gb/s Ethernet.
- `a100` – [On-prem] *high-performance* NVIDIA DGX-A100 node with 8 NVIDIA A100 (40GB VRAM) GPUs and 1.6Tb/s Infiniband.
- `quad` – [On-prem] *workstation* node with 4 NVIDIA Quadro RTX6000 (24GB VRAM) GPUs and 200 Gb/s Infiniband.

Models involved

- Real world DL training tasks

Table 2. Models used in our evaluations.

Size	Task	Model	Dataset	Target Metric	Batch Sizes	Optimizer
S	Image Classification	ResNet18 [16]	CIFAR-10 [30]	94% Top-1 acc	[128 - 4096]	SGD
M	Question-Answering	BERT [12]	SQuAD [46]	0.88 F1 score	[12 - 384]	AdamW [33]
	Speech Recognition	DeepSpeech2 [6]	CMU-ARCTIC [28]	25% word err	[20 - 640]	SGD
L	Object Detection	YOLOv3 [47]	PASCAL-VOC [14]	85% mAP	[8 - 512]	SGD
XL	Image Classification	ResNet50 [16]	ImageNet-1k [11]	75% Top-1 acc	[200, 12800]	SGD
XXL	LLM Finetuning	2.8B GPT [45]	SQuAD	0.88 F1 score	[48, 384]	AdamW

Results

- Sia overperforms current state-of-the-art schedulers.

Trace	Policy	JCT		Makespan	Avg. GPU-hours/job	Contention		Avg. job restarts
		Avg.	p99			Avg.	Max.	
Philly	Sia	**0.6h ± 0.1**	**9.5h**	**14.2 ± 1.9h**	**4.0 ± 0.7**	**6.9**	**31**	**2.9**
	Pollux	1.0 ± 0.1h	14.9h	24.5 ± 7.9h	5.6 ± 1.1	7.2	42	5.8
	Gavel+TJ	1.9 ± 0.3h	30.0h	33.8 ± 8.6h	9.0 ± 6.3	9.9	56	5.7
Helios	Sia	**0.7 ± 0.1h**	**10.9h**	**14.9 ± 1.7h**	**4.8 ± 0.7**	**7.4**	**32**	**3.4**
	Pollux	1.0 ± 0.2h	15.0h	25.5 ± 8.0h	5.9 ± 0.7	6.9	47	5.3
	Gavel+TJ	2.5 ± 0.9h	38.7h	43.0 ± 10.9h	12.1 ± 3.7	9.2	48	7.5
new-Trace	Sia	**0.7 ± 0.1h**	**4.6h**	**52.2 ± 1.3h**	**3.0 ± 0.1**	**13**	**69**	5.0
	Pollux	1.5 ± 0.2 h	10.3h	62.3 ± 4.6h	3.4 ± 0.2	22	85	5.4
	Gavel+TJ	11.3 ± 3.0h	98.1h	110 ± 21.5h	6.4 ± 1.1	96	243	**4.5**

Results

- Sia overperforms current state-of-the-art schedulers.

- Adapting hybrid parallel jobs
- Attribution of primary benefits
- Finish Time Fairness
- ...

Conclusion

- Sia improves job completion times (JCT) by 30-93% while using 12-60% fewer GPU hours.
  - derived from 3 real-world environments

- Quick to evaluate GPU clusters with many GPU types and thousands of GPUs.
  - scalability up to 2000 GPUs

Performance of Sia

Matches state-of-the-art schedulers in their target domains (Gavel, Pollux etc.);

Outperforms state-of-the-art schedulers in **union** of their domains (Adaptivity + Heterogeneity), and;

The **first** cluster scheduler able to elastically scale hybrid parallel jobs.

...

References

References

[1] Hu, Sia, and Heh. https://www.britannica.com/topic/Hu-Egyptian-religion, 2022 (accessed December 10, 2022).

[2] Sia. https://en.wikipedia.org/wiki/Sia, 2022 (accessed December 10, 2022).

[3] petuum/adaptdl. https://github.com/petuum/adaptdl/tree/osdi21-artifact, 2022 (accessed January 2022).

[4] AWS Tranium. https://aws.amazon.com/machine-learning/trainium/, 2023 (accessed April 2023).

[5] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, OSDI'16, USA, 2016. USENIX Association.

[6] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182. PMLR, 2016.

[7] Sanjith Athlur, Nitika Saran, Muthian Sivathanu, Ramachandran Ramjee, and Nipun Kwatra. Varuna: scalable, low-cost training of massive deep learning models. In *Proceedings of the Seventeenth European Conference on Computer Systems*, pages 472–487, 2022.

[8] Shubham Chaudhary, Ramachandran Ramjee, Muthian Sivathanu, Nipun Kwatra, and Srinidhi Viswanatha. Balancing efficiency and fairness in heterogeneous gpu clusters for deep learning. In *Proceedings of the Fifteenth European Conference on Computer Systems*, EuroSys '20.

[9] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *CoRR*, 2015.

[10] Henggang Cui, Hao Zhang, Gregory R. Ganger, Phillip B. Gibbons, and Eric P. Xing. Geeps: Scalable deep learning on distributed gpus with a gpu-specialized parameter server. In *Proceedings of the Eleventh European Conference on Computer Systems*, EuroSys '16. Association for Computing Machinery, 2016.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, 2019. Association for Computational Linguistics.

[13] Steven Diamond and Stephen Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17(1):2909–2913, 2016.

[14] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[15] Juncheng Gu, Mosharaf Chowdhury, Kang G. Shin, Yibo Zhu, Myeongjae Jeon, Junjie Qian, Hongqiang Liu, and Chuanxiong Guo. Tiresias: A GPU cluster manager for distributed deep learning. In *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*, pages 485–500, Boston, MA, February 2019. USENIX Association.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[17] Qinghao Hu, Peng Sun, Shengen Yan, Yonggang Wen, and Tianwei Zhang. Characterization and prediction of deep learning workloads in large-scale gpu datacenters. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15, 2021.

[18] Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Mia Xu Chen, Dehao Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V. Le, Yonghui Wu, and Zhifeng Chen. *GPipe: Efficient Training of Giant Neural Networks Using Pipeline Parallelism.* 2019.

[19] Changho Hwang, Taehyun Kim, Sunghyun Kim, Jinwoo Shin, and KyoungSoo Park. Elastic resource sharing for distributed deep learning. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*, pages 721–739, 2021.

[20] Petuum Inc. petuum/adaptdl: Resource-adaptive cluster scheduler for deep learning training., April 2021.

[21] Myeongjae Jeon, Shivaram Venkataraman, Amar Phanishayee, Junjie Qian, Wencong Xiao, and Fan Yang. Analysis of {Large-Scale}{Multi-Tenant}{GPU} clusters for {DNN} training workloads. In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, pages 947–960, 2019.

[22] Zhe Jia, Blake Tillman, Marco Maggioni, and Daniele Paolo Scarpazza. Dissecting the graphcore ipu architecture via microbenchmarking. *arXiv preprint arXiv:1912.03413*, 2019.

[23] Zhihao Jia, Matei Zaharia, and Alex Aiken. Beyond data and model parallelism for deep neural networks. *Proceedings of Machine Learning and Systems*, 1:1–13, 2019.

[24] Tyler Johnson, Pulkit Agrawal, Haijie Gu, and Carlos Guestrin. AdaScale SGD: A user-friendly algorithm for distributed training. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4911–4920. PMLR, 13–18 Jul 2020.

[25] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture*, pages 1–12, 2017.

[26] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

[27] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[28] John Kominek and Alan W Black. The cmu arctic speech databases. In *Fifth ISCA workshop on speech synthesis*, 2004.

[29] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*, 2014.

[30] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, 2009.

[31] Gary Lauterbach. The path to successful wafer-scale integration: The cerebras story. *IEEE Micro*, 41(6):52–57, 2021.

[32] Tan N. Le, Xiao Sun, Mosharaf Chowdhury, and Zhenhua Liu. Allox: Compute allocation in hybrid clusters. EuroSys '20. Association for Computing Machinery, 2020.

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[34] Kshiteej Mahajan, Arjun Balasubramanian, Arjun Singhvi, Shivaram Venkataraman, Aditya Akella, Amar Phanishayee, and Shuchi Chawla. Themis: Fair and efficient {GPU} cluster scheduling. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*, pages 289–304, 2020.

[35] Andrew Makhorin. Glpk (gnu linear programming kit). *http://www. gnu. org/s/glpk/glpk. html*, 2022.

[36] Sam McCandlish, Jared Kaplan, and Dario Amodei. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*, 2018.

[37] Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*, 2018.

[38] Xupeng Miao, Yujie Wang, Youhe Jiang, Chunan Shi, Xiaonan Nie, Hailin Zhang, and Bin Cui. Galvatron: Efficient transformer training over multiple gpus using automatic parallelism. *Proc. VLDB Endow.*, 16(3), 2022.

[39] Deepak Narayanan, Aaron Harlap, Amar Phanishayee, Vivek Seshadri, Nikhil R. Devanur, Gregory R. Ganger, Phillip B. Gibbons, and Matei Zaharia. Pipedream: Generalized pipeline parallelism for dnn training. In *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, SOSP '19. Association for Computing Machinery, 2019.

[40] Deepak Narayanan, Keshav Santhanam, Fiodar Kazhamiaka, Amar Phanishayee, and Matei Zaharia. {Heterogeneity-Aware} cluster scheduling policies for deep learning workloads. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*, pages 481–498, 2020.

[41] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. Efficient large-scale language model training on gpu clusters using megatron-lm. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '21. Association for Computing Machinery, 2021.

[42] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[43] Yanghua Peng, Yixin Bao, Yangrui Chen, Chuan Wu, and Chuanxiong Guo. Optimus: An efficient dynamic resource scheduler for deep learning clusters. In *Proceedings of the Thirteenth EuroSys Conference*, EuroSys '18, New York, NY, USA, 2018. Association for Computing Machinery.

[44] Aurick Qiao, Sang Keun Choe, Suhas Jayaram Subramanya, Willie Neiswanger, Qirong Ho, Hao Zhang, Gregory R. Ganger, and Eric P. Xing. Pollux: Co-adaptive cluster scheduling for goodput-optimized deep learning. In Angela Demke Brown and Jay R. Lorch, editors, *15th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2021, July 14-16, 2021*. USENIX Association, 2021.

[45] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[46] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2016.

[47] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[48] Alexander Sergeev and Mike Del Balso. Horovod: fast and easy distributed deep learning in tensorflow. *CoRR*, 2018.

[49] Noam Shazeer, Youlong Cheng, Niki Parmar, Dustin Tran, Ashish Vaswani, Penporn Koanantakool, Peter Hawkins, HyoukJoong Lee, Mingsheng Hong, Cliff Young, Ryan Sepassi, and Blake Hechtman. Mesh-tensorflow: Deep learning for supercomputers. In *Advances in Neural Information Processing Systems*, 2018.

[50] Dharma Shukla, Muthian Sivathanu, Srinidhi Viswanatha, Bhargav Gulavani, Rimma Nehme, Amey Agrawal, Chen Chen, Nipun Kwatra, Ramachandran Ramjee, Pankaj Sharma, et al. Singularity: Planet-scale, preemptible, elastic scheduling of ai workloads. *arXiv preprint arXiv:2202.07848*, 2022.

[51] Prasoon Sinha, Akhil Guliani, Rutwik Jain, Brandon Tran, Matthew D Sinclair, and Shivaram Venkataraman. Not all gpus are created equal: characterizing variability in large-scale, accelerator-rich systems. *arXiv preprint arXiv:2208.11035*, 2022.

[52] Muthian Sivathanu, Tapan Chugh, Sanjay S Singapuram, and Lidong Zhou. Astra: Exploiting predictability to optimize deep learning. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 909–923, 2019.

[53] Jakub M Tarnawski, Deepak Narayanan, and Amar Phanishayee. Piper: Multidimensional planner for dnn parallelization. In *Advances in Neural Information Processing Systems*, 2021.

[54] John Thorpe, Pengzhan Zhao, Jonathan Eyolfson, Yifan Qiao, Zhihao Jia, Minjia Zhang, Ravi Netravali, and Guoqing Harry Xu. Bamboo: Making preemptible instances resilient for affordable training of large DNNs. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, pages 497–513, Boston, MA, April 2023. USENIX Association.

[55] Colin Unger, Zhihao Jia, Wei Wu, Sina Lin, Mandeep Baines, Carlos Efrain Quintero Narvaez, Vinay Ramakrishnaiah, Nirmal Prajapati, Pat McCormick, Jamaludin Mohd-Yusof, Xi Luo, Dheevatsa Mudigere, Jongsoo Park, Misha Smelyanskiy, and Alex Aiken. Unity: Accelerating DNN training through joint optimization of algebraic transformations and parallelization. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, pages 267–284, Carlsbad, CA, July 2022. USENIX Association.

[56] Wencong Xiao, Romil Bhardwaj, Ramachandran Ramjee, Muthian Sivathanu, Nipun Kwatra, Zhenhua Han, Pratyush Patel, Xuan Peng, Hanyu Zhao, Quanlu Zhang, et al. Gandiva: Introspective cluster scheduling for deep learning. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 595–610, 2018.

[57] Wencong Xiao, Shiru Ren, Yong Li, Yang Zhang, Pengyang Hou, Zhi Li, Yihui Feng, Wei Lin, and Yangqing Jia. Antman: Dynamic scaling on gpu clusters for deep learning. In *Proceedings of the 14th USENIX Conference on Operating Systems Design and Implementation*, OSDI'20. USENIX Association, 2020.

[58] Dan Zhang, Safeen Huda, Ebrahim Songhori, Kartik Prabhu, Quoc Le, Anna Goldie, and Azalia Mirhoseini. A full-stack search technique for domain optimized deep learning accelerators. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '22. Association for Computing Machinery, 2022.

[59] Hao Zhang, Yuan Li, Zhijie Deng, Xiaodan Liang, Lawrence Carin, and Eric Xing. Autosync: Learning to synchronize for data-parallel distributed deep learning. In *Advances in Neural Information Processing Systems*, 2020.

[60] Lianmin Zheng, Zhuohan Li, Hao Zhang, Yonghao Zhuang, Zhifeng Chen, Yanping Huang, Yida Wang, Yuanzhong Xu, Danyang Zhuo, Eric P. Xing, Joseph E. Gonzalez, and Ion Stoica. Alpa: Automating inter- and Intra-Operator parallelism for distributed deep learning. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*. USENIX Association, 2022.

[61] Pengfei Zheng, Rui Pan, Tarannum Khan, Shivaram Venkataraman, and Aditya Akella. Shockwave: Fair and efficient cluster scheduling for dynamic adaptation in machine learning. *arXiv preprint arXiv:2210.00093*, 2022.

Q & A

Slides