Vrije Universiteit Amsterdam
`https://www.overleaf.com/read/vhvbxytqwpgf#b21f28`

Honours Programme, Problem Statement

Precision Capacity Planning: Simulating the Cost-Energy Trade-offs of LLM Training on Homogeneous Clusters

---

Author: Thijs van den Heuvel (2849429)

1st supervisor: Prof. Dr. Ir. Cav. Alexandru Iosup
daily supervisor: MSc. Dante Niewenhuis

A report submitted in fulfillment of the requirements for the Honours Programme, which is an excellence annotation to the VU Bachelor of Science degree in Computer Science/Artificial Intelligence/Information Sciences
version 1.0

December 12, 2025

Abstract

The training of Large Language Models (LLMs) has evolved into a massive industrial operation, where single training runs are costing tens of millions of dollars and consuming gigawatt-hours of energy. As models scale from billions to trillions of parameters, the financial and environmental sustainability of AI is becoming a critical and global challenge. Datacenters training these kinds of massive LLMs rely currently on a "good enough" human judgment or massive over-provisioning to manage these workloads, leading to significant inefficiencies in capital and energy use. Simulation offers a way to optimize these systems without the cost of physical experimentation. However, state-of-the-art simulators like OpenDC currently lack the specialized workload models required to accurately capture the non-functional properties of modern LLM training loops, such as gradient synchronization latency and checkpointing overheads.

This report proposes to bridge this gap by developing a parametric LLM training workload model for OpenDC. We aim to simulate the cost and energy trade-offs of training on homogeneous clusters, enabling datacenter operators to identify the best possible configuration where performance is maximized before diminishing returns and excessive carbon emissions set in.

Keywords

Large Language Models, LLMs, Simulation, Datacenter, Cost-Energy

1 Introduction

Artificial Intelligence (AI) has shifted from a niche research field to a key factor of global datacenter demand. This AI era has introduced a new class of workloads, like the training of Large Language Models (LLMs). These type of workloads are not short-lived inference tasks or traditional web services, but they are synchronous, monolithic, and resource-intensive training jobs that occupies thousands of GPUs for months at a time [4, 11] . Estimating the carbon footprint of models like BLOOM reveals that training alone can emit over 50 tonnes of CO2[9]. Efficiency in these systems is not just a performance metric, but a necessity for both financial and environmental sustainability.

Training is estimated to account for 10-40% of the total energy consumed by an LLM in its lifetime[3]. This is substantial and optimizing this to reduce even a few percent would cut costs and emissions. Discrete-event simulators could get every bit of performance out of a system resulting into millions of dollars in operational savings and significant reduction in carbon emissions. These simulators Act as a critical lever in this optimization, enabling companies to identify these efficiency gains without the capital risk of physical experimentation.

Datacenter configurations where all compute nodes (GPUs) are identical, an example being 1000 NVIDIA H100s, is the current industrial standard for training Frontier Models to minimize the "straggler effect" and synchronization latency found in heterogeneous environments. Even in these heterogeneous clusters, there is still a lot of time lost on synchronization. A simple background process can stall thousands of others during gradient synchronization[8]. Optimizing LLM training is notoriously difficult because physical experimentation is expensive. Operators often over-provision hardware to ensure stability, avoiding the risk of potential downtime but leading to massive resource waste[13]. Current simulators also fail to model the "system-level" components of training, specifically Checkpointing mechanisms (saving model states to disk in case of failure) and gradient synchronization (network communication between clusters). Recent surveys insidate that 30-40% of training time can be lost due to these overheads if not optimized, but standard simulators still treat them as negligible[16, 17].

Key prior work relies on the "Scaling Laws" of Neural Networks[5], which which predict the raw compute requirements for training models based on their size. To meet these demands, industry has adopted distributed training systems like Megatron-LM[11], which split the workload across thousands of GPUs. In the field of simulation, tools like OpenDC[10] and ASTRA-sim have successfully modeled general cloud and network behaviors. Still, a gap remains: recent surveys[14] and production traces[7] show that current simulators fail to account for the "system-level" friction of training, specifically the massive overheads of checkpointing and synchronization. This is a critical gap, as existing tools cannot accurately predict the true energy and financial impact of large-scale training workloads.

So this leads us to the main research question: **How can we utilize discrete-event simulation to determine the cost-optimal and energy-optimal homogeneous cluster configuration for training Large Language Models?** This will become the first open-source LLM Training Workload Model for OpenDC that specifically accounts for reliability overheads like checkpointing and gradient synchronization. Allowing researchers to study "Green AI" infrastructure without needing access to supercomputers to do these experiments. This project will contribute to the scientific community by providing a new tool for simulating LLM training, enabling more efficient and sustainable AI development. It will also help me develop my skills in simulation, distributed systems, and AI, preparing me for a career in this rapidly evolving field.

2 Background

2.1 The lifecycle of Large Language Models: Training vs Inference

It is crucial to distinguish between the two primary phases of an LLM's lifecycle: Training and Inference.
Inference is the process of generating a response from a trained model. It is stateless, highly parallelizable, and typically consumes a negligible amount of energy per request. Text classification for example consumes only 0.002Wh per request[3].
Training is the focus of this research, involving a stateful, monolithic workload that is iterating billions of parameters across thousands of GPUs. While inference accounts for the majority of accumulated energy usage over years of deployment, training is the most capital and resource intensive phase per unit of time. Training for GPT-4 is estimated to have an energy consumption of 51,772 to 62,318 MWh[3]. This is the equivalent to around 5,000 to 6,000 average American households' annual electricity consumption. Optimization of this phase is critical because unlike inference, which can be distributed across edge

devices, training requires a concentrated, high-performance datacenter environment where inefficiencies scale with cluster size.

2.2 Homogeneous Cluster and 3D Parallelism

To manage the computational scale of LLM training, the industry has standardized on 3D parallelism. This is a hybrid strategy that combines Data, Tensor and Pipeline parallelism to partition models across thousands of GPUs[11]. To support this synchronization, operators almost exclusively deploy Homogeneous Clusters, where all the compute nodes are identical. The "straggler effect," in which slower nodes cause the entire training task to be delayed because quicker nodes must wait for slower nodes to complete, is minimized by enforcing homogeneity[7]. However, even in these homogeneous environments, recent studies using production traces have shown that hardware homogeneity does not guarantee performance homogeneity. Software-level inefficiencies, such as uneven pipeline partitioning or pauses from garbage collection, can still cause a significant synchronization delay across nodes. These "software stragglers" can cause up to 45% of allocated GPU resources to be wasted, even in clusters with identical hardware specification[7].

2.3 Checkpointing and Fault Tolerance

Training runs take a very long time, often multiple weeks or months. In this period, hardware failures are statistically bound to happen. To make sure fault tolerance is effective, training systems implement checkpointing: The process of periodically pausing the computation to save the model's parameters and optimizer states to persistent storage (SSDs). Although this is necessary, this procedure introduces a significant Checkpointing Overhead. For every save operation, standard asynchronous checkpointing can stop training for more than 1.3 seconds [17]. In a large-scale cluster where saving occurs frequently to mitigate failure risks, these stalls represent a massive efficiency loss. Techniques like "Gradient-Assisted Checkpointing" attempt to reduce this stall time to sub-second levels[17], but their impact on total energy consumption at the scale of 10,000+ GPUs has not yet been simulated.

2.4 Discrete-Event Simulation (OpenDC)

The problem with analyzing these inefficiencies on real hardware is the cost. One cannot simply experiment with different configurations on a 10,000 GPU cluster or simply "pause" a $50 million training run to test a new scheduling hypothesis. Discrete-Event Simulation (DES) offers a solution by modeling the system as a sequence of events in time. OpenDC[10] is a state-of-the-art DES platform that has successfully modeled cloud datacenter workloads and serverless functions. However, current simulators often treat jobs as generic units of compute duration and assume that small-scale node behavior is still correct when scaled up[14]. They lack the specific workload models required to simulate the internal dynamics of LLM training, specifically the network bursts of 3D parallelism and the I/O stalls of checkpointing, rendering them insufficient for precise capacity planning in the context of LLM training.

3 Problem

We identified a critical gap in the current simulation tools available: **There is no accessible instrument to accurately predict the cost-efficiency and energy impact of LLM training configurations on homogeneous clusters**. While there is a growing body of research that has focused on Model Optimization, reducing computational complexity by up to 35% through techniques like sparse training and quantization[12], far less attention has been given to System Optimization. Even with highly compressed models, the infrastructure overheads of fault tolerance and synchronization persist. Current simulators often focus on the former (compute time), ignoring the latter (system-level overheads), creating a blind spot in energy estimation. Specifically, we identify the following issues:

- **The Fallacy of Idealized Execution**: Existing simulators normally model training under the assumption of ideal execution, treating homogeneous clusters as perfectly synchronized straggler free systems. This ignores the non-deterministic delays of 3D parallelism. As revealed from production traces, even in homogeneous clusters designed for stability, "software stragglers" (caused by uneven pipeline partitioning or garbage collection pauses) can cause the system to waste up to 45% of its allocated GPU resources[7]. Current tools are unable to capture these synchronization inefficiencies, resulting in overly optimistic performance and energy estimates that do not reflect real-world efficiency losses.

- **Ignored Reliability Costs**: Fault tolerance is mandatory for the month-long duration of training large LLMs on thousands of GPUs. However, the energy and time cost of Checkpointing is rarely modeled in planning tools. Data shows that standard asynchronous checkpointing can stall GPU computation for over 1.3 seconds per save interval[17]. In a large-scale cluster with frequent checkpoints, these delays accumulate, leading to a significant and unoptimized energy overhead that stays regardless of how optimized the model architecture itself is.

- **The Absence of Multi-Objective Trade-off Analysis**: Due to it being almost impossible to predict these system-level overheads, operators are forced to rely on massive over-provisioning or guessing. There is a lack of research quantifying the trade-off between speed, money and carbon emissions. For example, slowing down training slightly to accommodate a more energy-efficient checkpointing schedule could save megawatt-hours of energy, but without a simulator to quantify this trade-off, these optimizations are considered too risky to attempt on live production hardware.

4 Related Work

Numerous studies have explored the challenges and solutions associated with training large language models (LLMs) in distributed computing environments. However, there is a notable gap in the literature regarding the simulation of LLM training workloads on homogeneous clusters, particularly int he context of energy efficiency and fault tolerance. In this section we review the related work that is highly relevant to our research.

4.1 Training Performance Modeling

Several works have focused on modeling the performance of distributed training workloads. Early analytical models, such as the scaling laws proposed by Kaplan et al. [5], showed the relationship between model size, dataset size, and compute requirements (FLOPs). However, these models go by idealized hardware conditions for execution and fail to account for system-level inefficiencies. Other studies have focused on trace simulations to capture these behaviors, for example, Lumos [6] uses execution traces to predict training time with high accuracy (3.3% error) by modeling the computation and communication dependencies. Echo [1] simulates distributed training at scale by tracing runtime workloads and estimating the combined communication overheads. While this approach works good for performance prediction, these approaches treat reliability overheads as negligible, completely ignoring the impact of these fault tolerance mechanisms on total job duration.

4.2 Infrastructure and Fault Tolerance

Distributed training needs to be reliable, especially at scale. This reliability though, is often a major bottleneck for LLM training. As models scale to thousands of GPUs, hardware failures are statistically bound to happen during long training runs. So this process always requires robust fault tolerance mechanisms. The standard approach to this is Checkpoint/Restart (C/R), which periodically saves the model state to persistent storage. However, this process introduces significant I/O overhead. Recent studies like GoCkpt [17] and FlowCheck [2] have proposed techniques to lessen this cost, such as gradient-assisted checkpoints and asynchronous state saving. System-level analyses from ByteDance [7] and meta [18] show that ”stragglers”, slow nodes caused by software congestion or hardware issues, can waste up to 45% of cluster resources. Despite these papers, most simulators do not yet have the capability to simulate these stragglers using models or simulate the complex I/O bursts associated with checkpointing, leading to unrealistic energy and performance estimates.

4.3 Simulation Gaps

a survey of end-to-end training simulators [14] showed a critical gap in the firld: current tools focus on computation and communication while forgetting the checkpointing mechanisms and fault recovery procedures. While hardware-centric simulators like ASTRA-sim [15] provide cycle accurate modeling of network collectives, they are often too low level for long term capacity planning. On the other hand, high level cloud simulators like OpenDC [10] excel at resource management but lack the specific workload models for stateful, synchronous AI training. This research aims to bridge this gap by integrating a reliability-aware workload model into OpenDC, being the first simulation of the cost-energy trade-offs in fault tolerance LLM training.

5 Research Question(s)

This project will focus on **How can we utilize discrete-event simulation to determine the cost-optimal and energy-optimal homogeneous cluster configurations for training Large Language Models?**. We focus specifically on the Training phase of LLMs, excluding inference, as training represents the most capital-intensive and critical bottleneck for infrastructure planning. We constrain our research to Homogeneous Clusters (identical GPUs and network topology), which is the industry standard for minimizing hardware-induced latency. Within this scope, we will focus on modelling the "System-level" non-functional properties, specifically Checkpointing Overhead and Gradient Synchronization, rather than low-level chip microarchitecture. We use the OpenDC simulator as our experimental platform.

5.1 RQ1: How can we model the non-functional properties of LLM training workloads, specifically gradient synchronization latency and checkpointing I/O bursts, for discrete-event simulation

This research question aims to first abstract the complex reality of a distributed job into a smaller mathematical model that defines its key performance characteristics. We need to do this because existing simulators treat jobs as generic units of compute duration and assume that small-scale node behavior is still correct when scaled up. This will fail to capture the massive efficiency losses from synchronization delays and checkpointing stalls that can occur at scale. This part is challenging because it first requires a deep understanding of the mechanics of the training process, and then simplifying these complex structures into a simple model without losing the key dynamics. We must accurately model the "software- stragglers" that occur even in homogeneous clusters.

5.2 RQ2: How to implement this model within the OpenDC ecosystem to accurately measure power consumption and throughput on homogeneous GPU clusters?

A theoretical model is insufficient without a practical implementation. It must be validated within a simulation engine to run scenarios at a scale and collect accurate metrics. This will provide a reusable software tool for the scientific community, allowing researchers to study "Green AI" infrastructure without needing access to expensive supercomputers. OpenDC is primarily designed for cloud workloads (stateless requests). Extending it to support stateful, long-running training jobs requires a different logic for mechanisms for checkpointing and handling the complex synchronization patterns of training.

5.3 RQ3: What is the relationship between cluster size, checkpointing frequency, and training cost efficiency

To validate the model, we will run experiments verifying its accuracy and provide the "Precision Capacity Planning" from the title. We aim to identify the optimal configuration where adding more GPUs would yield diminishing returns due to increased synchronization overheads. This will answer the core societal and economic question of "Are we wasting money and energy by building clusters that are too big?" Operators currently lack the data to make these trade-off decisions, leading to over-provisioning and resource waste. It requires designing a good experimental setup that isolates the key variables to prove causality. We must also simulate the "What-if" scenarios of techniques like Gradient-Assisted Checkpointing to measure their potential benefits at larger scale (over 10,000 GPUs) that has never been tested before in an actual cluster.

6 Approach

In this section, we will outline the approach needed to address the research questions posed in the previous section. Each research question is based on the previous question. Thus, we will first address RQ1, then RQ2, and finally RQ3.

6.1 RQ1: Workload Characterization (Modeling)

To define the LLM training workload and its characteristics, we will synthesize a Parametric Performance model. To obtain this model, we will perform a literature review and analyze real-world performance traces from data centers. With these traces, we will identify key performance metrics and patterns that are representative of the LLM training workload. Using this information, we will develop a parametric model that captures the essential characteristics of the workload. We will extend this model to include the "Wasted Time" equations, which mathematically represent the latency of checkpointing as a function of bandwidth and frequency. A formal Mathematical specification of a training job will be the outcome of this research question, that will also define when computation pauses and how long synchronization takes, ready for simulation.

6.2 RQ2: Integration into OpenDC (Implementation)

The implementation phase will focus on translating the mathematical model into the OpenDC simulator codebase. We will extend the OpenDC interface to add the LLM training workload as a new type of workload that can be simulated. This will involve converting the formal definitions of "Wasted Time" and gradient synchronization into executable code within the OpenDC workload interface. We will implement a SyntheticLLMWorkload class that dynamically calculates resource demands based on high-level inputs, such as model parameter count and cluster size, rather than static traces. To ensure the accuracy of this implementation, we will verify the simulator's behavior and output against known truth logs from open-source training runs (BLOOM as example), ensuring that our synthetic model accurately reproduces real-world energy and performance metrics.

6.3 RQ3: Experimental evaluation

To evaluate the trade-offs between cluster scale and efficiency, we will conduct a factorial design experiment. We will simulate a fixed training workload across a spectrum of Homogeneous Cluster configurations, varying the node count from 128 to 16,384 GPUs. We will also check some "What-If" scenarios and see how a standard asynchronous checkpointing stacks up against an optimized Gradient-Assisted Checkpointing technique. This experimental setup allows up to isolate the impact of synchronization overheads, resulting in the data needed to quantify the precise energy cost of fault tolerance at scale.

Validation and Analysis

To interpret and validate the experimental data, we will perform a trade-off analysis. In this analysis, we will plot the simulation results to visualize the most efficient cluster configuration for time to completion, total energy consumption and capital costs. By quantifying the relationship between cluster size and efficiency losses, we will derive concrete capacity planning guidelines. This analysis aims to demonstrate how a system-level optimization can yield significant energy-savings, giving a scientific basis for more sustainable infrastructure design.

7 Plan

For this project, i have planned about 40 weeks of work, completing by October 2026. The work is divided into four phases, aligned with the research questions defined in the previous sections.

7.1 Phase 1: RQ1 - Definition and modeling (weeks 1-8)

This phase will result in a finalized mathematical workload specification and a completed literature review chapter.

- **Weeks 1-2 (Literature Review):** Set up the OpenDC codebase to get familiar with the simulator and create a local experiment to understand how ti works. Conduct a literature review on Scaling laws for LLMs, to gather the mathematical formulas needed for compute requirements.
- **Weeks 3-5 (Literature Review):** Analyze the recent systems papers (GoCkpt, Byte Dance) to extract formulas for I/O latency and model the ”Software Stragglers” in the homogeneous clusters.
- **Weeks 6-8(Model Formalization):** Synthesize findings into a draft ”Parametric LLM Workload Model” specification. Formalize the mathematical equations for ”Gradient Synchronization” and ”Checkpointing Overhead”.

7.2 Phase 2: RQ2 - Design and Implementation (weeks 9-20)

This phase will result in a functional OpenDC prototype implementation capable of simulating a full training run.

- **Weeks 9-10 (Architecture Design):** Analyze the OpenDC workload interface and design the class hierarchy for SyntheticLLMWorkload.
- **Weeks 11-14 (Core Logic Implementation):** Implement the basic compute logic (duration based on FLOPs) and 3D parallelism, which simulates the data/tensor /pipeline stages.
- **Weeks 15-18 (Overhead Logic Implementation):** Implement the Gradient Synchronization barrier logic and the checkpointing logic (I/O stall events based on the waste formulas). Run initial validation experiment with a small dummy cluster to verify it’s stable.
- **Weeks 19-20 (Refinement):** Refine the code for performance and work on the documentation for the API.

7.3 Phase 3: RQ3 - Validation and Experimentation (Weeks 21-32)

This phase will result in a complete dataset of experimental results and finalized charts.

- **Weeks 21-23 (Validation Experiments):** Configure the simulator to match the hardware specifications of the BLOOM training cluster. Run simulations to compare energy/duration against the public logs and calibrate the model parameters accordingly.
- **Weeks 24-27 (Experimentation and Execution):** Design and execute the full factorial experiment. Run batches for a baseline asynchronous checkpointing run and an optimized Gradient-Assisted Checkpointing run, using clusters of variable sizes (128 to 16,384 GPUs).
- **Weeks 28-32 (Data Processing):** Process raw output logs into usable datasets. Generate initial visualizations like charts of throughput vs cluster size and advanced plots for cost vs energy.

7.4 Phase 4: Analysis and Writing (Weeks 33-40)

- **Weeks 33-35 (Drafting Results):** Draft the Methodology, Implementation and Results chapters. Get started on a Discussion and Conclusion, focussing on the societal impact of energy efficiency.
- **Weeks 36-37 (Review and Revision):** Submit the first full draft to supervisors and revise the document based on feedback.
- **Weeks 38-40 (Final polishing):** Finalize formatting and references. Prepare the final presentation slides.
- **Week 40 (Submission):** Submit the final thesis document.

8 Conclusion

In conclusion, this project aims to develop a comprehensive simulation framework for evaluating the energy efficiency of large language model (LLM) training workloads in homogeneous clusters. We identified a clear gap in simulating the training phrase of LLM lifecycles. By developing a specialized workload model for OpenDC, we move beyond generic abstractions to capture the precise friction of 3D parallelism and Checkpointing Overhead. Integrating mathematical models of synchronization latency and real-world straggler effects allows us to simulate the behavior of homogeneous clusters. This research provides the first open-source instrument capable of quantifying the trade offs between speed, cost and carbon emissions for LLM training, ensuring that future AI systems can be deployed sustainably at scale with the most efficient use of resources.

References

[1] Y. Feng, Y. Chen, K. Chen, J. Li, T. Wu, P. Cheng, C. Wu, W. Wang, T.-Y. Ho, and H. Xu. Echo: Simulating distributed training at scale, 2024.

[2] Z. Huang, H. Nie, H. Jia, B. Jiang, J. Guo, J. Lu, R. Wen, B. Lyu, S. Zhu, and X. Wang. Flowcheck: Decoupling checkpointing and training of large-scale models. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, page 1334–1349. ACM, Mar. 2025.

[3] Z. Ji and M. Jiang. A systematic review of electricity demand for large language models: evaluations, challenges, and solutions. *Renewable and Sustainable Energy Reviews*, 225:116159, Jan. 2026.

[4] Z. Jiang, H. Lin, Y. Zhong, Q. Huang, Y. Chen, Z. Zhang, Y. Peng, X. Li, C. Xie, S. Nong, Y. Jia, S. He, H. Chen, Z. Bai, Q. Hou, S. Yan, D. Zhou, Y. Sheng, Z. Jiang, H. Xu, H. Wei, Z. Zhang, P. Nie, L. Zou, S. Zhao, L. Xiang, Z. Liu, Z. Li, X. Jia, J. Ye, X. Jin, and X. Liu. Megascale: Scaling large language model training to more than 10, 000 gpus, 2024.

[5] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models, 2020.

[6] M. Liang, H. T. Kassa, W. Fu, B. Coutinho, L. Feng, and C. Delimitrou. Lumos: Efficient performance modeling and estimation for large-scale llm training, 2025.

[7] J. Lin, Z. Jiang, Z. Song, S. Zhao, M. Yu, Z. Wang, C. Wang, Z. Shi, X. Shi, W. Jia, et al. Understanding stragglers in large model training using what-if analysis. *arXiv preprint arXiv:2505.05713*, 2025.

[8] J. Lin, Z. Jiang, Z. Song, S. Zhao, M. Yu, Z. Wang, C. Wang, Z. Shi, X. Shi, W. Jia, Z. Liu, S. Wang, H. Lin, X. Liu, A. Panda, and J. Li. Understanding stragglers in large model training using what-if analysis, 2025.

[9] A. S. Luccioni, S. Viguier, and A.-L. Ligozat. Estimating the carbon footprint of bloom, a 176b parameter language model, 2022.

[10] F. Mastenbroek, G. Andreadis, S. Jounaid, W. Lai, J. Burley, J. Bosch, E. van Eyk, L. Versluis, V. van Beek, and A. Iosup. Opendc 2.0: Convenient modeling and simulation of emerging technologies in cloud datacenters. In *2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pages 455–464, 2021.

[11] D. Narayanan, M. Shoeybi, J. Casper, P. LeGresley, M. Patwary, V. A. Korthikanti, D. Vainbrand, P. Kashinkunti, J. Bernauer, B. Catanzaro, A. Phanishayee, and M. Zaharia. Efficient large-scale language model training on gpu clusters using megatron-lm, 2021.

[12] K. R. Narsepalle. Energy-efficient training and inference in large language models: Optimizing computational and energy costs. *International Journal of Computer Applications*, 187(14):1–13, Jun 2025.

[13] D. Patterson, J. Gonzalez, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D. So, M. Texier, and J. Dean. Carbon emissions and large neural network training, 2021.

[14] J. Svedas, H. Watson, N. Laubeuf, D. Moolchandani, A. Nada, A. Singh, D. Biswas, J. Myers, and D. Bhattacharjee. A survey of end-to-end modeling for distributed dnn training: Workloads, simulators, and tco, 2025.

[15] W. Won, T. Heo, S. Rashidi, S. Sridharan, S. Srinivasan, and T. Krishna. Astra-sim2.0: Modeling hierarchical networks and disaggregated systems for large-model training at scale. In *2023 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, page 283–294. IEEE, Apr. 2023.

[16] W. Xu, X. Huang, S. Meng, W. Zhang, L. Guo, and K. Sato. An efficient checkpointing system for large machine learning model training. In *Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis (SC-W)*, pages 896–900, 11 2024.

[17] K. Zhang, Y. Chen, Z. Hu, W. Lin, J. Xu, and W. Chen. Gockpt: Gradient-assisted multi-step overlapped checkpointing for efficient llm training, 2025.

[18] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang, and L. Zettlemoyer. Opt: Open pre-trained transformer language models, 2022.