SPARX: A Resource-aware Partitioning Framework for Distributed Inference on Heterogeneous Devices

Washington R. D. Silva
washingtonrds@estudante.ufscar.br
Federal University of Sao Carlos
Sorocaba, Sao Paulo, Brazil

Abstract

The increasing integration of complex AI workloads into constrained, heterogeneous edge environments necessitates distribution of the inference computation to ensure execution feasibility. However, the combinatorial explosion of hardware and model variants makes infrastructure provisioning a complex design space to navigate. Hence, we introduce SPARX, a framework that enables infrastructure providers to analytically verify distributed deployment plans prior to physical hardware procurement. Our preliminary results demonstrate SPARX's capability to guarantee minimal performance requirements across diverse server configurations.

ACM Reference Format:
Washington R. D. Silva. 2026. SPARX: A Resource-aware Partitioning Framework for Distributed Inference on Heterogeneous Devices. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (SIGMETRICS '26).* ACM, New York, NY, USA, 2 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

As the transition toward 6G transforms cellular infrastructure into an "AI-Native" ecosystem, infrastructure providers face a critical provisioning dilemma: Deep Neural Networks (DNNs) are becoming increasingly complex, yet the edge computing landscape is evolving into a highly heterogeneous environment. Modern edge servers are now hybrid clusters, integrating Application-Specific Integrated Circuits (ASICs) alongside programmable networking devices such as Data Processing Units (DPUs) and Field Programmable Gate Arrays (FPGAs) [5].

This convergence creates a paradox between AI computational demand and edge hardware constraints. Hence, to execute such workloads on these devices without violating strict Service Level Agreements (SLAs), the inference computation must be distributed. Instead of running a model monolithically, which risks execution feasibility, the model is partitioned across multiple devices and inference is performed collaboratively between them [5]. To achieve this, the DNN is converted into an Intermediate Representation (IR) modeled as a Directed Acyclic Graph (DAG); which can then be split into subgraphs and assigned to diverse compute units.

SIGMETRICS '26, Ann Arbor, MI

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/XXXXXXX.XXXXXXX

However, effectively distributing these subgraphs is non-trivial, since heterogeneous devices possess fundamentally different architectures and computing capacities – ranging from the massive parallel throughput of GPUs to the custom spatial pipelining of FPGAs. Consequently, the same operation will exhibit different execution times and energy costs depending on its hardware mapping. Furthermore, splitting a model forces intermediate data tensors to traverse internal interconnects or networks, introducing communication overheads.

Due to this complex design space and the combinatorial explosion of model and hardware variants, providing adequate infrastructure for an AI-Native environment is a highly complex challenge. Since physical benchmarking all possible combinations is economically unscalable, performance modeling has emerged as the only practical approach to optimize system dimensioning [1].

While recent literature acknowledges this need, it primarily targets large-scale, homogeneous environments [1–3], leaving heterogeneous edge provisioning underexplored [4]. Therefore, to enable distributed inference in this heterogeneous hardware landscape, we introduce SPARX (Scalable Partitioning for AI Resource eXploration). SPARX is a design space exploration framework that enables infrastructure providers to evaluate different server configurations under distinct constraints. By doing so, they can analytically verify distributed deployment plans for a given AI application and guarantee minimal performance requirements before committing to any physical hardware purchases or deployments.

2 Methodology

To enable the partitioning of a given AI model across the heterogeneous edge continuum, SPARX functions as the core engine for a "Distributed Inference Planning as a Service" (DIPaaS) architecture. As illustrated in Fig. 1, the framework systematically resolves the multidimensional search space of hardware specs, DNN models, operational constraints, and latency requirements through three phases:

- **Profiling and Abstraction:** The pipeline begins by converting the trained DNN into a DAG, where nodes denote layer-level operations (e.g., convolutions) characterized by structural attributes, and edges represent tensor dependencies. Simultaneously, SPARX profiles the edge hardware, retrieving a specification dictionary containing processing throughput (GFLOPS), thermal design power (TDP), memory capacity, and internal bandwidths (PCIe/Network).
- **Cost Estimation and Contention:** By parsing the DAG's structural attributes, SPARX calculates the FLOP count and output data size for every node. It constructs a composite latency model aggregating raw computation time with data

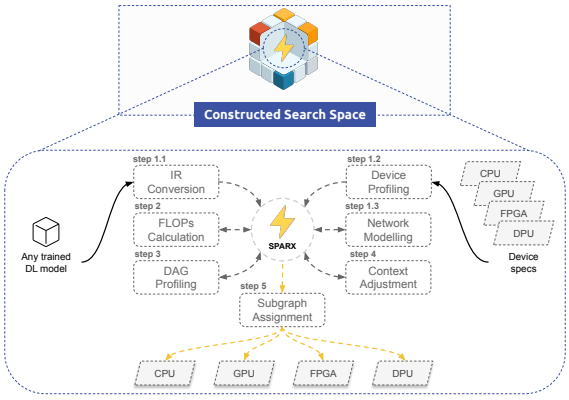

Figure 1: Model partitioning workflow.

movement overheads, scaling this against the device's TDP to project the energy footprint for each hardware target. To reflect multi-tenant edge environments, SPARX incorporates a configurable contention parameter. This allows infrastructure providers to simulate various background workloads and analytically verify whether inference SLAs can be guaranteed even under highly restricted resource conditions.

- **Resource-Aware Partitioning:** Finally, SPARX synthesizes these inputs to derive the optimal deployment strategy. This resource allocation challenge is mathematically formulated as a Multiprocessor Scheduling with Precedence Constraints (MPSPC) problem. The partitioner enforces strict boundary constraints: DAG precedence dependencies, device memory limits, and the application's global latency budget. Because the framework is highly modular, it can optimize objective functions for several scenarios (e.g., minimizing total energy footprint), supporting a spectrum of optimization models ranging from exact mathematical techniques to sub-optimal approximations.

3 Preliminary Results

To validate the DIPaaS architecture, we evaluated SPARX's modularity in a Mobile Augmented Reality (MAR) scenario, which imposes a strict 20 ms global latency deadline. We profiled two fundamental models (YOLO11n and ResNet-18) across three simulated heterogeneous edge server tiers: High-Performance (Scenario A), Intermediate (Scenario B), and Constrained (Scenario C). All tiers were modeled with a mix of GPUs, CPUs, and FPGAs.

We used a SCIP exact solver with a multi-stage optimization objective: For Scenarios A and B, we first minimized the energy footprint (1-stage), and then compressed the makespan (2-stage) to free resources earlier. For Scenario C, we employed a 3-stage optimization pipeline (3-stage) that sequentially enforces load balancing, energy minimization, and makespan minimization to guarantee execution feasibility while achieving the same benefits from previous scenarios.

For comparison, we also evaluated simple greedy heuristics designed according to each optimization objective: h_es minimizes energy, while h_ct minimizes makespan. Since these algorithms rely on the standardized input generated during the Profiling and Abstraction phase, any optimization strategy can be seamlessly integrated. The exact solver and heuristics demonstrated here are merely examples.

Our preliminary results verified SPARX's adaptability across diverse deployment scenarios, demonstrating the exact solver's capability to guarantee the SLA across all tested configurations. In Scenario C, for instance, the exact multi-stage solver compressed the ResNet-18 makespan from the 20 ms boundary down to 9.41 ms while preserving the baseline energy footprint. As illustrated in Fig. 2, while h_es and h_ct estimated a lower total energy footprint, they achieve this by centralizing the workload on the FPGA. In contrast, the multi-stage solver successfully balanced the ResNet-18 workload across all devices, ensuring operational stability at an energy tradeoff.

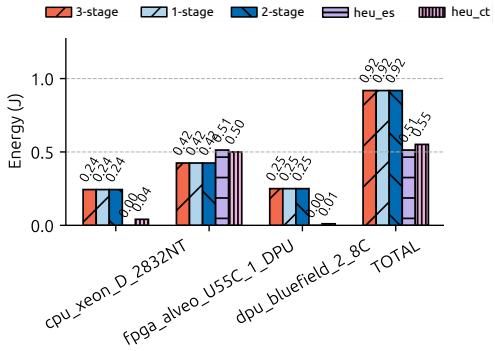

Figure 2: Estimated energy consumption (in Joules) for the workload distributed in Scenario C.

However, the exact solver required several minutes to find an optimal solution (limited in this evaluation to a 10-minute timeout). Conversely, the heuristics computed deployment plans in under 1 millisecond, but their efficacy depends on several factors, such as the DAG structure and resource contention. For instance, under high GPU contention in Scenarios A and B, h_ct failed to meet the SLA entirely.

To prevent the localized bottlenecks observed in h_es and h_ct while avoiding the overhead of exact solvers, we plan to explore robust meta-heuristics (e.g., Simulated Annealing) to provide a near-optimal, time-bounded optimization strategy.

References

[1] Sho Ko et al. 2024. DFModel: Design Space Optimization of Large-Scale Systems Exploiting Dataflow Mappings. *arXiv preprint arXiv:2412.16432* (2024), 1–22.

[2] Sumit Kumar et al. 2025. Simulating LLM training workloads for heterogeneous compute and network infrastructure. In *Proceedings of the 2nd Workshop on Networks for AI Computing*. ACM, Coimbra, Portugal, 105–107. doi:10.1145/3748273.3749212.

[3] Seonho Lee et al. 2025. Forecasting GPU Performance for Deep Learning Training and Inference. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*. ACM, Rotterdam, Netherlands, 493–508.

[4] Md. Maruf Hossain Shuvo et al. 2023. Efficient Acceleration of Deep Learning Inference on Resource-Constrained Edge Devices: A Review. *Proc. IEEE* 111, 1 (2023), 42–91. doi:10.1109/JPROC.2022.3226481.

[5] Guolin Sun et al. 2025. Toward AI-Native Task Orchestration for Collaborative Computing in SAGSINs. *IEEE Communications Magazine* 63, 12 (2025), 112–118. doi:10.1109/MCOM.004.2400304.