고려대학교 전기전자공학부 컴퓨터 시스템 연구실 소개

(지도교수: 안정섭)

jsahn@korea.ac.kr
https://Jeongseob.github.io

2024-03 기준으로 작성됨

컴퓨터 시스템 연구실

- 2024년 재설립 (2017 가을 아주대학교에서 시작)
- 연구 방향: 컴퓨터 시스템을 빠르고 효율적으로 설계 하는 방법 탐구
- 구성원: 박사과정 3명, 석사과정 1명, 학부연구생 2명

EuroSys 2023 논문 발표
이탈리아 로마

컴퓨터 시스템 동계학술대회 2023
강원도 평창

ATC 2022 논문 발표
미국 캘리포니아

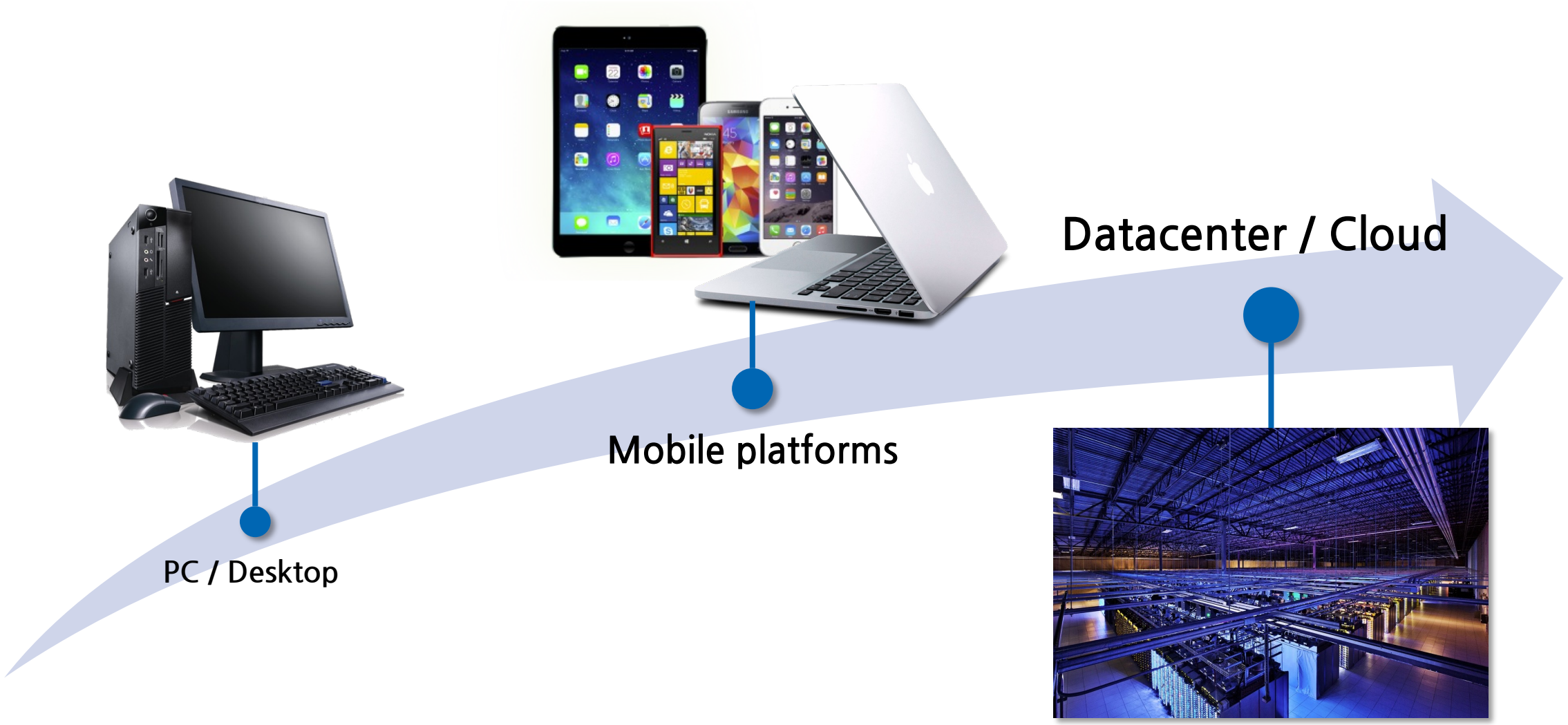
지도교수: 안정섭 (Jeongseob Ahn)

- Associate Professor @ Korea University
  - School of Electrical Engineering
  - Department of Communication Engineering

- Research Interest
  - Computer Systems and Architecture
  - Cloud & Datacenter Computing
  - Systems for Artificial Intelligence

KAIST	MICHIGAN	ORACLE Labs	AJOU UNIVERSITY 1973	KOREA UNIVERSITY 1905
PhD	Research Fellow	Researcher	Assistant/Associate Professor	Associate Professor
Feb. 2015	Feb. 2015 ~ Jan. 2016	Feb. 2016 ~ Jul. 2017	Sep. 2017 ~ Feb. 2024	Mar. 2024 ~ Current

컴퓨팅 패러다임의 변화



컴퓨팅 패러다임의 변화

Expensive: 100s of millions $$ to construct and operate

→ Need to build **fast** and **efficient** datacenter servers

어떤 연구를 하는지?

- 분야: 시스템 및 아키텍처
  - 프로세서 (CPU/GPU/NPU) 구조
  - 운영체제, 가상 머신
  - 병렬 및 분산 처리 시스템
  - 인공지능 시스템
  - 데이터센터 / 클라우드 컴퓨팅

연구주제 1: 빠르고 효율적인 LLM 서빙 시스템

- 연구 내용
  - 거대 모델(예: GPT-3)는 컴퓨팅 자원을 매우 많이 필요로 하고 특히 메모리 부족이 심각함
  - 원활하고 빠른 추론 서비스를 위해 다수의 GPU를 통한 분산 및 병렬처리 시스템 구축
  - 대규모 GPU 자원을 효율적으로 관리하는 시스템 및 소프트웨어 기술 연구
    - 많은 LLM 추론 요청을 처리할 수 있는 기술 개발
    - 비용 효율적인 학습 시스템 개발
- 우대 사항 (필수 X)
  - GPU 아키텍처 이해 및 CUDA 프로그래밍 경험
  - PyTorch 경험

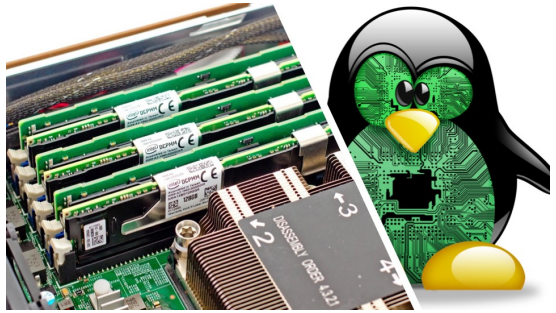
vLLM

PyTorch

관련 논문: Fast and Efficient Model Serving Using Multi-GPUs with Direct-Host-Access (EuroSys '23)

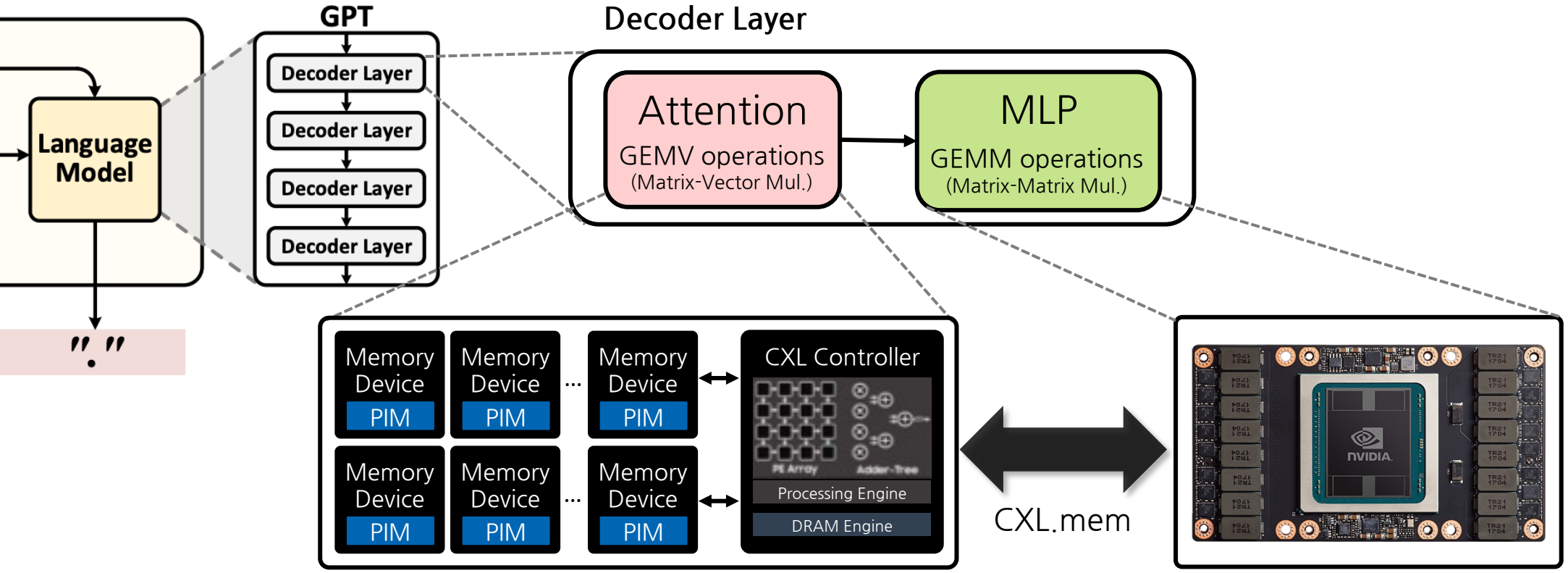
연구주제 2: 운영체제 메모리 관리 최적화

- 연구 내용
  - 서버 컴퓨터의 코어 및 메모리 양은 계속해서 증가
  - 앞으로는 성능이 다른 여러 메모리 장치가 등장
  - 대용량 메모리를 효율적으로 사용하기 위한 메모리 관리 기법 필요!
    - 데이터센터 환경을 고려한 운영체제 및 라이브러리 수준에서 관리 기술 연구
    - 클라우드 컴퓨팅을 고려한 하이퍼바이저 레벨에서 관리 기술 연구
- 우대 사항 (필수 X)
  - 리눅스 프로그래밍 경험자



관련 논문: Exploring the Design Space of Page Management for Multi-Tiered Memory Systems (USENIX ATC '21)

연구주제 3: NPU 및 PIM 아키텍처 설계



CXL-based Memory Expander with Near Data Processing Architecture

GPU

- 연구 내용
  - LLM 추론 가속을 위한 NPU 및 PIM 기술 연구

최근 3년 연구 결과: 최우수 컨퍼런스 논문 3편

Exploring the Design Space of Page Management for Multi-Tiered Memory Systems

Jonghyeon Kim, Wonkyo Choe, and Jeongseob Ahn
Ajou University

Abstract

With the arrival of tiered memory systems comprising various types of memory, such as DRAM and SCM, the operating system support for memory management is becoming increasingly important. However, the way that operating systems currently manage pages was designed under the assumption that all the memory has the same capabilities based on DRAM. This oversimplification leads to non-optimal memory usage in tiered memory systems. This study performs an in-depth analysis of page management schemes in the current Linux design extending NUMA to support systems equipped with both DRAM and SCM (Intel's DCPMM). In such multi-tiered memory systems, we find that the critical factor in performance is not only the *access locality* but also the *access tier* of memory. When considering both characteristics, there are several alternatives to page placement. However, current operating systems only prioritize access locality. This paper explores the design space of page management schemes, called *AutoTiering*, to use multi-tiered memory systems effectively. Our evaluation results show that our proposed techniques can significantly improve performance for various workloads, compared to the stock Linux kernel, by unlocking the potential of the multi-tiered memory hierarchy.

1 Introduction

With the advent of in-memory computing, such as data analytics, key-value stores, and graph processing, the demand for high-density DRAM has been steadily increasing in recent years [27]. However, due to the challenge of scaling DRAM density, a new class of memory has received attention to bridge the performance gap between DRAM and SSD. For example, Intel recently unveiled its non-volatile memory based on 3D Xpoint technology, called Optane DC Persistent Memory Module (DCPMM) that provides more density than DRAM whil
vendors such
adopted such
services [4,

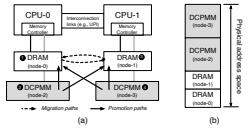


Figure 1: Software-managed tiered memory system augmented on the NUMA architecture

Since modern server systems are built with the Non-Uniform Memory Access (NUMA) architecture, future large-memory systems will take the shape of tiered memory augmented on traditional NUMA architecture, called *multi-tiered memory*. Figure 1 presents a real-world multi-tiered memory system used throughout this study. Each compute chip has two types of memory: DRAM (upper-tier) and Intel's DCPMM (lower-tier). We configure both DRAM and DCPMM to be fully exposed to software as memory.

This paper presents that the recent advancement in Linux [15] and tiered memory studies [16, 20, 35] do not lead to optimal page placement in multi-tiered memory systems. As the new class of memory becomes part of the main memory, the critical factor in performance is not only the *access locality* but also the *access tier* of memory. However, current page placement schemes have been established for DRAM-only NUMA architecture and only consider locality between threads and memory [2, 8, 12, 13, 21, 38]. As a result, the current design is far from exploiting the potential benefits of multi-tiered memory systems. For example, suppose the local DRAM becomes full when promoting pages from the lower-tier (DCPMM) to the upper-tier (DRAM) memory. In this case, the current state of the art leaves the page on the
ailability of the remote
cision is reasonable for
e there is no difference
lti-tiered memory sys-

USENIX ATC '21

Memory Harvesting in Multi-GPU Systems with Hierarchical Unified Virtual Memory

Sangjin Choi* *KAIST* Taeksoo Kim* *KAIST* Jinwoo Jeong *Ajou University* Rachata Ausavarungnirun *KMUTNB*
Myeongjae Jeon *UNIST* Youngjin Kwon *KAIST* Jeongseob Ahn† *Ajou University*

Abstract

With the ever-growing demands for GPUs, most organizations allow users to share the multi-GPU servers. However, we observe that the memory space across GPUs is not effectively utilized enough when consolidating various workloads that exhibit highly varying resource demands. This is because the current memory management techniques were designed solely for individual GPUs rather than shared multi-GPU environments.

This study introduces a novel approach to provide an illusion of virtual memory space for GPUs, called hierarchical unified virtual memory (HUVM), by incorporating the temporarily idle memory of neighbor GPUs. Since modern GPUs are connected to each other through a fast interconnect, it provides lower access latency to neighbor GPU's memory compared to the host memory via PCIe. On top of HUVM, we design a new memory manager, called memHarvester, to effectively and efficiently harvest the temporarily available neighbor GPUs' memory. For diverse consolidation scenarios with DNN training and graph analytics workloads, our experimental result shows up to 2.71× performance improvement compared to the prior approach in multi-GPU environments.

1 Introduction

As the demand for GPUs explodes, it is now a common practice in both academia and industry to equip multiple GPUs in a single server and make them shareable. Many enterprises in the industry have built large GPU clusters comprised of a set of multi-GPU servers to satisfy the demand for a variety of workloads from deep learning [1, 13, 18, 26, 36] to graph applications [6, 10, 19, 31] while saving the infrastructure cost by sharing. However, as a downside, achieving high GPU resource efficiency in such multi-GPU servers remains a challenge. Figure 1 presents that the current memory management technique is
environments where multiple jobs are running across GPUs independently. Although a small amount of memory ranging from hundreds of MB to a few GB remains idle in one or a few GPUs, other GPUs under heavy memory pressure rely on the host memory as a swap device that is significantly slower than remote GPUs within the same server.

Meanwhile, GPU vendors have faced the challenge of scaling the memory capacity of single GPUs. To overcome the limited capacity of GPUs, a train of previous studies provides an illusion of infinite memory space with the host memory [11, 14, 17, 25, 28]. However, none of the work does utilize the idle memory of neighbor GPUs in commodity multi-GPU systems. As modern GPU servers are commonly equipped with 8~16 GPUs connected via high-speed interconnect such as NVLink, accessing the idle memory of neighbor GPUs is much faster than that of the host. For instance, NVIDIA DGX-2 has 16 GPUs with point-to-point connections through NVLink 3.0, yielding a large pool of 512GB GPU memory at 600GB/s bidirectional bandwidth [23]. On the other hand, swapping GPU memory to host DRAM via the latest PCIe 4.0 could utilize up to 32GB/s bandwidth only.

In this study, we introduce a new approach providing an illusion of virtual memory space for GPUs called *hierarchical unified virtual memory (HUVM)* comprised of local GPU, spare memory of neighbor GPUs, and the host memory. HUVM opens up a new opportunity for memory virtualization by increasing the effective memory space with minimal performance overhead. When the local GPU memory does not have free space, HUVM leverages the spare[1] memory in neighbor GPUs *as a victim cache* between the GPU and host instead of directly swapping out data to the host memory.

However, it is challenging to effectively and efficiently harvest the spotty-available, small fraction of neighbor GPUs' memory because the amount of idle memory is highly variable and unknown a priori. Beyond the single GPU perspective, we develop the memory management scheme for modern
to find the best

*Co-first authors
†Corresponding aut
interchangeably.

USENIX ATC '22

Fast and Efficient Model Serving Using Multi-GPUs with Direct-Host-Access

Jinwoo Jeong, Ajou University, Suwon, Korea
Seungsu Baek, Ajou University, Suwon, Korea
Jeongseob Ahn, Ajou University, Suwon, Korea

Abstract

As deep learning (DL) inference has been widely adopted for building user-facing applications in many domains, it is increasingly important for DL inference servers to achieve high throughput while preserving bounded latency. DL inference requests can be immediately served if the corresponding model is already in the GPU memory. Otherwise, it needs to load the model from host to GPU, adding a significant delay to inference. This paper proposes *DeepPlan* to minimize inference latency while provisioning DL models from host to GPU in server environments. First, we take advantage of the `direct-host-access` facility provided by commodity GPUs, allowing access to particular layers of models in the host memory directly from GPU without loading. Second, we parallelize model transmission across multiple GPUs to reduce the time for loading models from host to GPU. We show that a single inference can achieve a 1.94× speedup compared with the state-of-the-art pipelining approach for `BERT-Base`. When deploying multiple BERT, RoBERTa, and GPT-2 instances on a DL inference serving system, `DeepPlan` shows a significant performance improvement compared to the pipelining technique and stable 99% tail latency.

CCS Concepts: **• Computer systems organization; • Software and its engineering → Software system structures;**

Keywords: DNN model serving, Direct-host-access, Parallel-transmission

ACM Reference Format:
Jinwoo Jeong, Seungsu Baek, and Jeongseob Ahn. 2023. Fast and Efficient Model Serving Using Multi-GPUs with Direct-Host-Access. In *Eighteenth European Conference on Computer Systems (EuroSys '23), May 9–12, 2023, Rome, Italy.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3552326.3567508

ACM ISBN 978-1-4503-9487-1/23/05…$15.00
https://doi.org/10.1145/3552326.3567508

Due to the increasing demand to utilize deep neural networks (DNNs) in many user-facing applications, it is becoming increasingly important to provide deep learning (DL) inference with low latency [8, 13, 17, 27]. To serve incoming inference requests within the strict latency constraints (e.g., service level objectives), a straightforward approach is to cache models in the GPU memory, as depicted in Figure 1a. However, the downside of this approach is that inference servers need to be over-provisioned for the peak load, increasing the operation cost of servers. A promising way to reduce the cost of GPU servers is to allow the number of models to extend beyond the GPU memory limit [20], leading to fewer GPU servers. Once GPU memory becomes insufficient to add a new model, we can reclaim the GPU memory space occupied by an inactive model and load the active model. If an inference request arrives at a model not ready in the GPU memory, it starts loading the corresponding model to GPU on-demand [34, 37] (Figure 1b). The remaining challenge is to minimize the (cold-start) time for loading DL models to GPU memory, which significantly delays inference. For instance, loading a `BERT-Base` model takes 40ms if the model is available in host memory, while a single inference on the model cached in the GPU memory is complete within 9.35ms for NVIDIA V100.

A recent inspiring study presented populating model transmission per layer granularity [6], enabling inference to start before the entire model is loaded, as shown in Figure 1c. This approach hides the time for loading layers by overlapping it with the computation. Since DNN models comprise a sequence of layers, we can separate the inference computation layer-by-layer. Once the first layer is loaded, the inference starts immediately. While performing the inference on the first layer, it loads the next layer simultaneously. However, to make such pipelining technique effective, it is required that the computation time must be sufficiently longer than the loading time. Otherwise, the computation cannot proceed until the corresponding layer is completely loaded, called *pipeline stall.* Since recent DNN models such as BERT and GPT have large layers that take a substantial loading time, it is challenging to fully overlap such layer loading time with the computation.

In this study, we explore three techniques to minimize the performance impact of loading models: executing layers

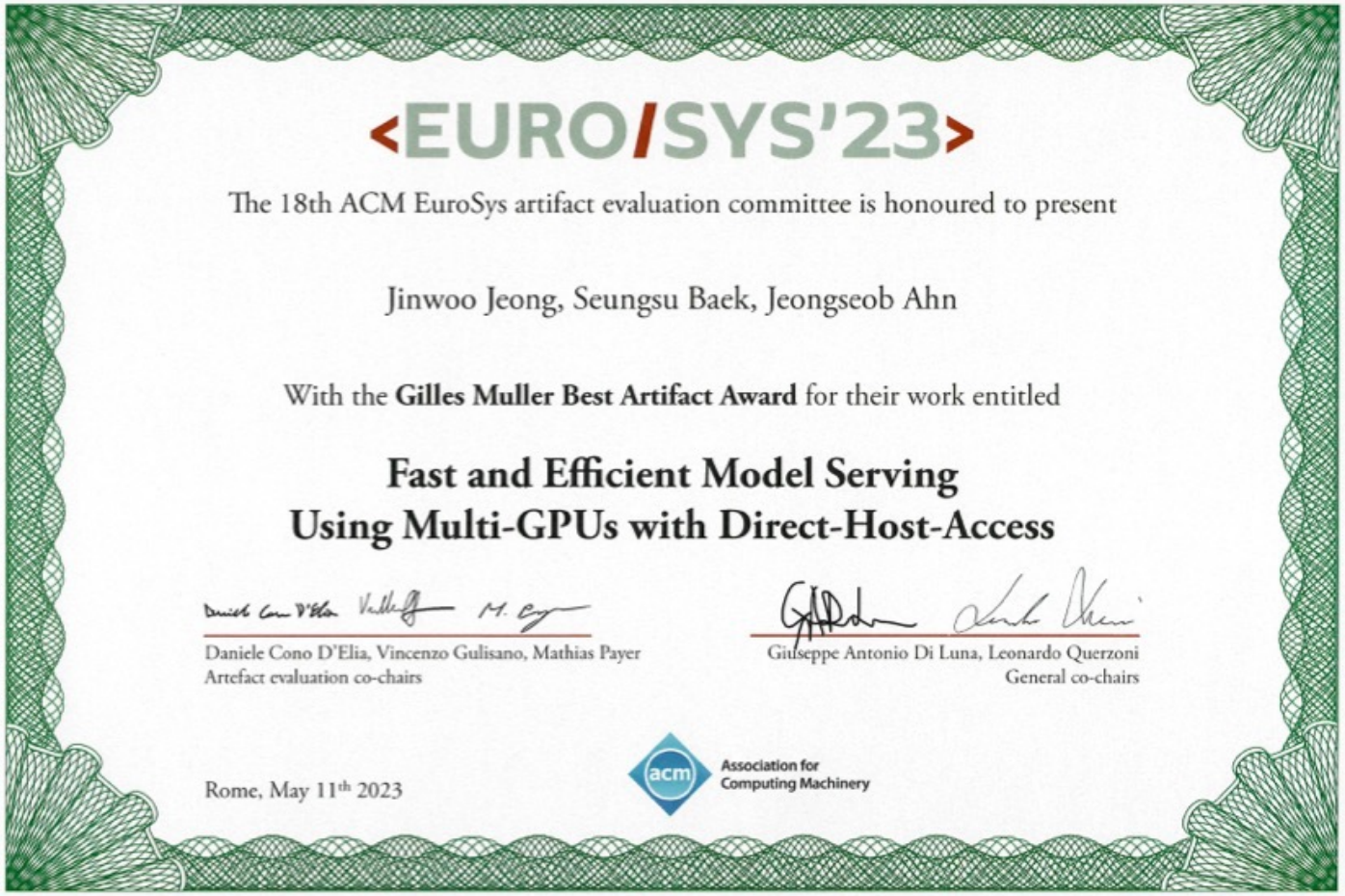
EURO/SYS ACM SIGOPS IN EUROPE

Best Artifact Award @ EuroSys 2023

EuroSys 2023
@EuroSys_conf · **Follow**

#eurosys23 GILLES MULLER BEST ARTIFACT AWARD:
Fast and Efficient Model Serving Using Multi-GPUs with Direct-Host-Access
Jinwoo Jeong (Ajou University),
Seungsu Baek (Ajou University),
Jeongseob Ahn (Ajou University)

Congratulations!



<EURO/SYS'23>

The 18th ACM EuroSys artifact evaluation committee is honoured to present

Jinwoo Jeong, Seungsu Baek, Jeongseob Ahn

With the **Gilles Muller Best Artifact Award** for their work entitled

Fast and Efficient Model Serving Using Multi-GPUs with Direct-Host-Access

Daniele Cono D'Elia, Vincenzo Gulisano, Mathias Payer
Artefact evaluation co-chairs

Giuseppe Antonio Di Luna, Leonardo Querzoni
General co-chairs

Rome, May 11th 2023

Association for Computing Machinery

컴퓨터 시스템 연구실 인턴 모집 (상시)

- 리눅스 커널
  - 운영체제 동작 원리에 대해서 심도 있는 이해
  - 커널 프로그래밍 경험을 통해서 운영체제 기술 습득 및 최적화
- 병렬 컴퓨팅
  - 병렬 처리 하드웨어 및 구동되는 소프트웨어 원리 이해
  - CPU 및 GPU에서 병렬로 동작하는 코드 구현에 관심 있는 학생
- 대상: 시스템 개발 및 연구에 관심이 있는 학생 누구나
  - 운영체제 / 시스템 프로그래밍 / 컴퓨터구조 관심있는 학생 누구나

문의: jsahn@korea.ac.kr

자주 물어보는 질문들

- 해당 연구에 관심이 있는데 실력이 부족한 것 같아요...
  - 연구에 대한 관심과 동기 부여가 가장 중요
  - 기본적인 C/C++ 프로그래밍 실력 + 필수과목 수강 (컴퓨터구조, 운영체제)
- 해당 연구에 관심이 있는데 어디서 시작할까요?
  - 연구 참여 인턴쉽

컴퓨터 시스템 연구에 관심이 있다면 연구실 구성원 아무에게나 연락주세요!

연락처: jsahn@korea.ac.kr