Tersiteab Adem

734-596-8373 | teadem at umich dot edu | linkedin | github | website
2765 Bob and Betty Beyster Building, 2260 Hayward, Ann Arbor, 48109

Education

University of Michigan — Ann Arbor, MI
Doctoral program in Computer Science & Engineering — *Sept. 2022 – May 2027 (expected)*
- Topics: Computer Architecture, Computer Systems and Sustainable Computing
- Advisor: Prof. Valeria Bertacco

Addis Ababa University — Addis Ababa, Ethiopia
Bachelor of Science in Electrical and Computer Engineering — *Dec. 2020*
- Specialization: Computer Engineering

Publication

- Evergreen: Comprehensive Carbon Modeling for Emission-Performance Tradeoffs **[IISWC'24]**
  Tersiteab Adem, Andrew McCrabb, Vidushi Goyal, Valeria Bertacco

- Octal: Efficient Automatic Data-Oblivious Program Transformations to Eliminate Side-Channel Leakage **[DevSec'24]**
  Biniyam Tiruye, Lauren Biernacki, Tersiteab Adem, Petros Mola, Todd Austin

Projects

Parallel Simulation of Coarse-Grained Reconfigurable Array using Structural Simulation Toolkit *Jan 2024 - April 2024*
- Implemented CGRA designed to accelerate matrix-matrix multiplication via systolic arrays using SST.
- This SST simulation of a PE-array achieves $5 - 24\times$ speedup as compared to running RTL simulation for different simulation time using a single core.
- As SST allows parallelism, able to achieve a simulation time speedup of up to $42\times$ on multiple cores.

Memory Access Pattern Recognition Using Search Algorithms *Sept 2023 - Dec 2023*
- Memory-intensive applications experience system performance issues mainly due to memory access latency. One of the ways to mitigate this issue is to prefetch data ahead of time to reduce the processor's wait time.
- Implemented search algorithms to find the memory access pattern being followed by programs to help inform memory prefetching.
- Hill climbing, simulated annealing, and the evolutionary genetic algorithm are used to search for the access pattern in a program's memory address trace.

COCO: A Compiler-based Obliviousness Checker and Optimizer *Sept 2022 - Dec 2022*
- Data oblivious programs prevent data access pattern leakage as their memory access is independent of the input data
- COCO uses taint analysis to track sensitive data and its usage in conditional instructions.
- Performers the following optimizations for improved performance while maintaining data obliviousness: Loop Unrolling, Loop Fusion, and Frequent Path Loop Invariant Code Motion (FPLICM).
- Transforms programs into their data oblivious equivalents by converting logical operators to bit-wise operators, when possible.

ACRE: Explainable Random Forest Process in Memory Accelerator *Sept 2022 - Dec 2022*
- Explainable Artificial Intelligence (XAI) has been developed to provide explanations for results obtained from AI models. There is currently a need to improve the performance of XAI models.
- A proposed solution aims to parallelize parts of the XAI algorithm, specifically inference tasks explainable Random Forest, and accelerate it using Process-in-Memory.
- It reduces the large bus traffic associated with this workload by placing comparator nodes in the memory for simple comparisons.

Benchmark Suite for explainable AI *June 2021 - Oct 2021*

- Gathered and curated different explainable AI models that range across both input data domains (images, tabular and data, text) and types of models (Decision Tree, Random Forest, NN).
- Developed metrics to measure the performance of explainable AI models quantitatively.
- Tested the performance of SHAP and LIME explanation methods regarding how faithful the interpretations are to the predictions made by different target models across image, text, and tabular datasets.
- Compiled and organized interpretable ML algorithms with metrics to evaluate them.
- Github link: https://github.com/tersiteab/BeXAI

Graph Application Acceleration through Optimal Vertex Placement *June 2021 - Oct 2021*

- Identified major bottleneck of graph application that operates on power-law distribution graphs.
- Tested different graph reordering techniques to get better performance in terms of execution speed.
- Measured the overall speedup obtained when using preprocessed graphs of selected graph applications on the baseline platform and OMEGA accelerator.
- Github link: https://github.com/tersiteab/graph-reorder-1

Work Experience

- Undergraduate Research Intern at University of Michigan — *June 2021 - Oct 2021*
- Software Engineer at Tohey Technologies — *Jan 2021 - June 2022*
- Data Science Intern at Hamoye — *Jan 2019 - June 2020*

Services

- Graduate Society of Women Engineer's (GradSWE at UM)- Secretary — *May 2024 - Present*
- Reviewer for University of Michigan graduate student admission — *2024*
- Lab Manager — *Jan 2024 - Present*
- CSE Graduate Student Organization at University of Michigan Ann Arbor (CSEG at UM) - Recruitment Chair — *Sept 2023 - April 2024*
- African Undergraduate Research Adventure (AURA) - Student Mentor — *May 2023 - August 2023*

Courseworks

- EECS 583 – Advanced Compilers
- EECS 573 – Microarchitecture
- EECS 570 – Parallel Computer Architecture
- EECS 592 – Foundations of AI
- EECS 598-09 – Privacy Enhancing Technologies
- EECS 598-010 – Quantum Computing Systems & Architecture (audit)

Technical Skills

Languages: Java, Python, C/C++, SQL (Postgres), JavaScript, HTML/CSS, R
Frameworks: React, Node.js, Flask, JUnit, WordPress, Material-UI, FastAPI
Developer Tools: Git, Docker, TravisCI, Google Cloud Platform, VS Code, Visual Studio, PyCharm, IntelliJ, Eclipse
Libraries: pandas, NumPy, Matplotlib

Awards

Very Great Distinction Award *Dec 2020*

- Received for scoring the highest grade from Computer Engineering Major from School of Electrical and Computer department