Swastik Mittal
smittal@ncsu.edu • (224) 388-2914 • mittalswastik.github.io

August 14, 2025

Hiring Manager
NVIDIA — Systems Team

I'm a PhD candidate at NC State with Dr. Frank Mueller, graduating in 2025, applying to NVIDIA's Systems team. My work spans security, scheduling, and performance—from Linux kernel paths to compiler/runtime layers and GPU execution.

My path began with **T-Pack (ISORC '21)**, where I instrumented the **Linux network layer** in a distributed real-time system to perform **packet-level execution-time analysis**, enabling **early intrusion detection with marginal overhead**. Working at the packet and timing layers showed how breaking execution into finer stages improves real-time threat detection.

Next, **T-Tex (ICCPS '25)** brought those ideas into **parallel runtimes**: a timing-aware security pipeline for **OpenMP** using **LLVM/Clang instrumentation** (passes), **OMPT and Context-Switch monitoring** to capture **microsecond-scale telemetry** and flag anomalies while staying within application budgets. Alongside peers' OpenMP-RT work on task scheduling, this formed a strong base for real-time capabilities in parallel programs.

Building on that foundation, my dissertation explores **controlled execution for OpenMP task offloads on NVIDIA GPUs**. I built an **LD_PRELOAD** interposer that routes CUDA kernels into **priority-queued streams** with **budgeted execution** and **admission control** for periodic and sporadic tasks. I also experimented with **partitioning SM resources (e.g., via Green Contexts)** to improve high-priority latency while keeping throughput stable under contention.

Beyond research, I care about **systems performance end-to-end**. At **AIM Intelligent Machines**, I combined **CUDA MPS**, perf/eBPF, and Nsight to cut robotics control-loop latency and turn traces into concrete runtime changes. In **OpenMP-Q**, I add **quantum devices as additional accelerators** via a quantum wrapper for task offloading. At **Uber**, I worked on the Go toolchain to improve **instruction cache (I-cache) locality** in a large production codebase.

I’d be excited to contribute across kernel scheduling, runtime/driver isolation, admission control, observability, and performance tuning in NVIDIA’s compute stack. **Thank you for your time**—I’d welcome a conversation.

Sincerely,
Swastik Mittal