# Vivek L. Kale

**Phone:** 217-369-7996 | **Email**: viveklkale@gmail.com | **LinkedIn**: linkedin.com/in/vlkale
**Github**: github.com/vlkale | **Website**: vlkale.github.io | **US Citizen with Secret Clearance**

## Professional Summary

- Highly skilled computational scientist and software developer with expertise in high performance computing (HPC), runtime systems, and parallel programming models for GPU-based clusters.
- Proven track record of contributions to parallel programming standards, open-source software for AI-assisted HPC tools for profiling and debugging, and research on adaptive load balancing.
- Effective communicator and collaborator with a strong record of publications and software projects.

## Relevant Experience

**Sandia National Laboratories**

*Principal Member of Technical Staff II* *July 2024 - Present*

- Pathfinding and software engineering for tools for Kokkos integrated with (1) HPC performance monitoring and feedback via LDMS and (2) PMPI and adaptive runtime systems for MPI.
- Developed AI-assisted HPC Tools through LLMs (coderosetta.com) and autotuning (TAU+APEX) for Kokkos applications run on NVIDIA GPUs, resulting in a poster presentation at GTC 2025.
- Research and pathfinding on the use of AI chips, e.g., Cerebras WSE-3, for science simulations.
- Submitted two proposals on correctness tools for HPC, each with $1.5M in funding for 3 years.

*Senior Member of Technical Staff* *August 2022 - June 2024*

- Developed and maintained Kokkos Tools for the CMake and Spack build system, tooling overheads, CI/CD, auto-tuning, and nvtx/roctx/vtune integration, leading to 15 merged github PRs.
- Developed a debugging tool that detected 7 common Kokkos user bugs by analyzing LLVM IR of Kokkos programs via symbolic execution, leading to a paper at SC24's Correctness workshop.
- Implemented prototype LLVM OpenMP feature for index set splitting of an OpenMP loop, leading to a 1.2x speedup for an OpenMP + CUDA benchmark and to OpenMP 6.0's new split directive.
- Drafted standards for OpenMP multi-GPU features for NVIDIA DGX, and for GPUs for AWS, Google Cloud, and OCI, leading to 19 proposed features for OpenMP versions 6.1 and 7.0.

**Brookhaven National Laboratory** *Assistant Computational Scientist* *May 2019 - August 2022*

- Implemented OpenMP user-defined multi-GPU scheduling for LLVM, offering 2.1x speedup over using MPI parallelization, leading to papers at IWOMP 2020 and BCB 2021.
- Implemented performance optimizations in LLVM for OpenMP asynchronous GPU offloading that achieved a 1.2x speedup, leading to a paper at SC22's HiPar workshop.
- Developed performance benchmarks that evaluated 5 major vendor OpenMP GPU implementations, leading to an ACM journal paper and an IWOMP 2021 workshop paper.
- Demonstrated technical leadership as technical project manager for the ECP SOLLVE project, submitting 12 ECP milestone reports, organizing 7 GPU hackathons, and defining 3 project KPIs.

**Charmworks** *Software Engineer* *May 2018 - May 2019*

- Implemented, tested and experimented with User-defined Loop Schedules (UDS) for OpenMP, leading to a paper at IWOMP 2018 and a prototype library for LLVM and GCC.
- Added the UDS feature to RAJA and Charm++'s CkLoop, with 1 github PR merged in Charm++.

**USC** - **Information Sciences Institute** *Computer Scientist* *Dec 2016 - May 2018*

- Performance analysis and optimization of 3-D image reconstruction application on NVIDIA GPUs via CUPTI and auto-tuning, leading to a performance-enhanced CUDA version of the application.
- Developed tuning support for coordinated loop scheduling and load balancing in Charm++, leading to a 1.2x speedup on a particle-in-cell benchmark code and a Best Poster Candidate at SC18.

**Charmworks** *Software Developer* *Jan 2016 - Dec 2016*

- Extended Charm++ to offer a novel runtime system capability of coordinating inter-node load balancing and intra-node loop scheduling, leading to 2 github PRs merged in Charm++.

**University of Illinois** *Postdoctoral Associate* *Jul 2015 - Dec 2015*

- Sped up a plasma-physics Fortran MPI+OpenACC code by 1.2x via a combination of GPU offload optimizations and loop transformations on an NVIDIA K80 GPU.

## Education

- Ph.D (Doctor of Philosophy), Computer Science, 2015, University of Illinois at Urbana-Champaign

  **Dissertation**: *Low-Overhead Scheduling to Improve Performance of Scientific Applications*
- B.S. (Bachelor of Science), Computer Science, 2007, University of Illinois at Urbana-Champaign

## Technical Skills

**Languages**: C, C++, CUDA, python, Fortran, Java, bash, csh, VHDL, Matlab;
**Libraries**: OpenMP (gomp, llvm), Kokkos, MPI (mpich), Charm++, OpenACC (pgi), Globus, mpi4py, pyomp, matplotlib, pandas, numpy;
**Tools**: Kokkos Tools, PMPI, ompt, PAPI, nvtx, NVIDIA Nsight, tau, hpcToolkit, VTune, clang-tidy, KLEE, gprof, gdb;
**Utilities**: git, cmake, spack, vi, clang-format, gnuplot, emacs, autoconf, LaTeX, docker;

## Open-source Software Projects

1. **OpenMP multi-GPU support**: User-defined multi-GPU loop scheduling for clang/LLVM OpenMP.
   *Repository*: `https://github.com/vlkale/taskGPUSched`

2. **Kokkos Tools**: Kokkos Tools and runtime systems for C++.
   *Repository*: `https://github.com/kokkos/kokkos-tools`

3. **MPI Slack Predictor**: MPI runtime tool using libunwind to predict slack trace
   *Repository*: `https://github.com/vlkale/slack-trace`