Kaushik G Kulkarni

Address	201N Goodwin Ave, Urbana, IL 61801 USA
Email	kgk2@illinois.edu
Web	https://kaushikcfd.github.io

Education

Aug '17 – *Present*	Ph.D. candidate in Computer Science University of Illinois at Urbana-Champaign, Urbana, IL *Domain-Specific Code Transformations in Computational Science based on the Polyhedral Model* *Advised by Prof. Andreas Klöckner*
Aug '13 – May '17	Bachelor of Technology in Mechanical Engineering Indian Institute of Technology, Bombay *Advised by Prof. Shiva Gopalakrishnan*

Experience

Aug '20 – *Present*	Research Assistant *National Center for Supercomputing Applications, UIUC*
Aug '17 – Aug' 20	Graduate Research/Teaching Assistant *Computer Science Dept., UIUC*
May '19 – Aug '19	Givens Associate *Argonne National Laboratory, Illinois*
Jan '16 – May '17	Teaching Assistant for Introduction to Numerical Analysis *Mathematics Dept., IITB*
May '16 – July '16	Software Engineering Intern *Morgan Stanley Strats and Modelling, Mumbai*

Awards and Achievements

2023	SIAM Travel Award for Conference on Computational Science and Engineering (*to be held in Amsterdam, NL*)
2022	Graduate College Presentation Award to present at SIAM Conference on Parallel Processing for Scientific Computing
2021	SIAM Travel Award for Conference on Computational Science and Engineering
2019	SIAM Travel Award for Conference on Computational Science and Engineering (Spokane, WA)
2016	Undergraduate Research Award (URA01) by Indian Institute of Technology, Bombay
2013	*Kishore Vaigyanik Protsahan Yojana Fellowship* Award
2013	Certificate of Merit for being among the State Top 1% in National Standard examination in Physics and Chemistry

Publications

[1] Tianjiao Sun, Lawrence Mitchell, Kaushik Kulkarni, Andreas Klöckner, David Ham, and Paul HJ Kelly. "A study of vectorization for matrix-free finite element methods." *The International Journal of High Performance Computing Applications* 34, no. 6 (2020): 629-644.

[2] Awan Bhati, Rajat Sawanni, Kaushik Kulkarni, and Rajneesh Bhardwaj. "Role of skin friction drag during flow-induced reconfiguration of a flexible thin plate." *Journal of Fluids and Structures* 77 (2018): 134-150.

Talks

[1] "A Two Stage IR for Transforming Array Expressions (not just) for GPUs," UIUC Compiler Seminar, 2022.

[2] "Transforming (not just) DG-FEM Array Expressions (not just) on GPUs," as part of Minisymposium on Flexible, Performance Portable Software for Partial Differential Equations at SIAM Conference on Parallel Processing for Scientific Computing, online, 2022.

[3] "UFL to GPU: Generating near-roofline action kernels," as part of Minisymposterium on Automatic Simulation Code Generation Tools and their Applications at SIAM Conference on Computational Science and Engineering, online, 2021..

[4] "UFL to GPU: Generating near-roofline action kernels," FEniCS workshop, online, 2021.

[5] "Optimizing Finite Element Operator Evaluation on GPUs," Firedrake USA workshop, Seattle, WA, 2020.

[6] "Transformation-based Code Optimizations for Finite Element Methods," as part of Minisymposium on Performance Portability through Source-to-source code translation, SIAM Conference on Computational Science and Engineering, Spokane, WA, 2019.

Key Software Contributions

(†: Primary author/Maintainer, ∗: Significant contributions)

Loopy† — An intermediate representation and transformation toolkit for programs with *single exit-point* loops for targets with SPMD execution models. My work introduced the callables abstraction within the IR that eventually enabled it to interact with the computer algebra systems in projects like Firedrake, MIRGE-Com. Documentation: https://documen.tician.de/loopy.

Pytato† — An intermediate representation for n-d array operations that supports the *pure* subset of Numpy operations. The IR can be targeted to Loopy IR, XLA/JAX, etc. Pytato has been used to deploy a production Discontinuous Galerkin solver which is capable of delivering roofline performance while a computational scientist writes Numpy-like code. Documentation: https://documen.tician.de/pytato.

feinsum† — Toolkit for managing databases of code-transformations on *Batched Einstein Summations*. Unlike typical computational primitive libraries that ship optimized computational kernel in source or binary form, *feinsum* records code-transformations from a trivial schedule, resulting in relative savings from not having to flush the operands to the DRAM. Documentation: https://feinsum.readthedocs.io.

Arraycontext∗ — Provides a subset of Numpy operations that are avaialable to both lazy and eager modes of execution. At runtime, these operations can be retargeted to array implementations, such as Pytato, JAX, PyOpenCL, etc. Documentation: https://documen.tician.de/arraycontext.

SYMOXIDE†	A computer algebra system written purely in RUST. Demonstrated ∼5× speedup w.r.t. state of the art implementations in PYTHON for expression traversals of production Scientific Computing workloads. Documentation: https://docs.rs/symoxide.
FIREDRAKE*/OP2*	A multi-IR UFL compiler for implementing FEM solvers. I contributed a domain-specific compilation strategy for 1-form assembly on GPUs. Documentation: https://firedrakeproject.org
ISL-Rs†	Rust bindings for Integer Set Library to access polyhedral analysis primitives, where the implementation is responsible for automatic cleanup of objects as per RUST semantics. Documentation: https://docs.rs/isl-rs/.
Miscellaneous	Minor improvements to PETSc, PyCUDA, PyOpenCL, PyMLIR, etc.

Research

Near-Roofline Timestepping of Discontinuous Galerkin Operators. DG-FEM workloads comprise of many fine-grained array operations that can push them into the memory-bound regime. Within a lazy-evaluation array framework, we develop a novel kernel fusion algorithm to deliver near-roofline perfromance for end-to-end evaluation of a timestepping stage. The algorithm has been integrated to run the simulations at the *Center for Exascale-enabled Scramjet Design*, a collaboration between multiple departments at UIUC.

Dense Linear Algebra Computational Primitives. Computational primitives serve as the abstraction layer between an optimizing compiler and a performance engineer. We proposed *Batched Einstein Summations* as a new computational primitive to overcome the shortcomings of BLAS-primitives by recording code-transformation within a database. To reduce the database size, we proposed canonicalization algorithms to reuse the knowledge between *ismorphic* classes of expressions.

Compiling UFL to GPUs. Unified Form Language (UFL) is a domain-specific language to represent variational forms in FEM solvers. The wide range of kernels that can be targeted from UFL makes it challenging to solve the latency hiding - synchronization tradeoff for a target micro-architecture. We proposed a parametric transformation algorithm along with a cost-model based auto-tuning strategy to deliver performance portability for a suite of weak-forms seen in real world applications. In collaboration with the FIREDRAKE PROJECT team at Imperial College, we integrated our algorithm into the FIREDRAKE system.

Array Programming Paradigms and Intermediate Representations. Array programs not only provide a close-to-math expressibility but also trivially lower to intermediate representations are closer to SIMD architectures making it easier to engineer optimizing compilers targeting such hardwares. We developed PYTATO, one such IR that lowers *n*-d array programs to computation graphs comprising of *pure*-Array Ops that can be targeted to OPENCL, CUDA, JAX.