Expression Re-writing and Code Specialization for Finite Element Integration

Fabio Luporini, Imperial College London
David A. Ham, Imperial College London
Paul H.J. Kelly, Imperial College London

The numerical solution of partial differential equations using the finite element method is one of the key applications of high performance computing. Local assembly is its characteristic operation. This entails the execution of a problem-specific kernel to numerically evaluate an integral for each element in the discretized problem domain. Since the domain size can be huge, executing efficient kernels is fundamental. Their optimization is, however, a challenging issue. Even though affine loop nests are generally present, the short trip counts and the complexity of mathematical expressions make it hard to determine a single or unique sequence of successful transformations. Therefore, we present the design and systematic evaluation of COFFEE, a domain-specific compiler for local assembly kernels. COFFEE manipulates abstract syntax trees generated from a high-level domain-specific language for PDEs by introducing domain-aware composable optimizations aimed at improving instruction-level parallelism, especially SIMD vectorization, and register locality. It then generates C code including vector intrinsics. Experiments using a range of finite-element forms of increasing complexity show that significant performance improvement is achieved.

Categories and Subject Descriptors: G.1.8 [**Numerical Analysis**]: Partial Differential Equations - Finite element methods; G.4 [**Mathematical Software**]: Parallel and vector implementations

General Terms: Design, Performance

Additional Key Words and Phrases: Finite element integration, local assembly, compilers, optimizations, SIMD vectorization

ACM Reference Format:
Fabio Luporini, Ana Lucia Varbanescu, Florian Rathgeber, Gheorghe-Teodor Bercea, J. Ramanujam, David A. Ham, and Paul H. J. Kelly, 2014. COFFEE: an Optimizing Compilerfor Finite Element Local Assembly. *ACM Trans. Arch. & Code Opt.* V, N, Article A (January YYYY), 1 pages.
DOI:http://dx.doi.org/10.1145/0000000.0000000

This research is partly funded by the MAPDES project, by the Department of Computing at Imperial College London, by EPSRC through grants EP/I00677X/1, EP/I006761/1, and EP/L000407/1, by NERC grants NE/K008951/1 and NE/K006789/1, by the U.S. National Science Foundation through grants 0811457 and 0926687, by the U.S. Army through contract W911NF-10-1-000, and by a HiPEAC collaboration grant. The authors would like to thank Dr. Carlo Bertolli, Dr. Lawrence Mitchell, and Dr. Francis Russell for their invaluable suggestions and their contribution to the Firedrake project.
Author's addresses: Fabio Luporini & Paul H. J. Kelly, Department of Computing, Imperial College London; David A. Ham, Department of Computing and Department of Mathematics, Imperial College London;

DOI:http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

2. PRELIMINARIES

3. AUTOMATED EXPRESSION RE-WRITING

4. CODE SPECIALIZATION

5. PERFORMANCE EVALUATION

5.1. Experimental Setup

6. RELATED WORK

7. CONCLUSIONS

REFERENCES