OpenUH – An Open Source OpenACC Compiler

Xiaonan Tian, Rengan Xu, Yonghong Yan, Zhifeng Yun, Sunita Chandrasekaran, Barbara Chapman
Department of Computer Science, University of Houston
Email: {xtian2, rxu6, yyan3, zyun, schandrasekaran, bchapman}@uh.edu , http://web.cs.uh.edu/~openuh

Introduction

- OpenACC is an emerging directive-based programming model for programming accelerators that typically enable non-expert programmers to achieve portable and productive performance of their applications.
- We constructed a prototype open-source OpenACC compiler OpenUH which is based on a branch of main stream Open64 compiler. The experiences could be applicable to other compiler implementation efforts.
- We provide multiple loop mapping strategies in the compiler on how to efficiently distribute parallel loops to the GPGPU accelerators. Our findings provide guidance for users to adopt suitable loop mappings depending on their application characteristics.
- OpenUH compiler adopts a source-to-source approach and generates readable CUDA source code for GPGPUs. This gives users opportunities to understand how the loop mapping mechanism are applied and to further optimize the code manually. It also allows us to leverage the advanced optimization features in the backend compilation step by the CUDA compiler.

Loops Transformation

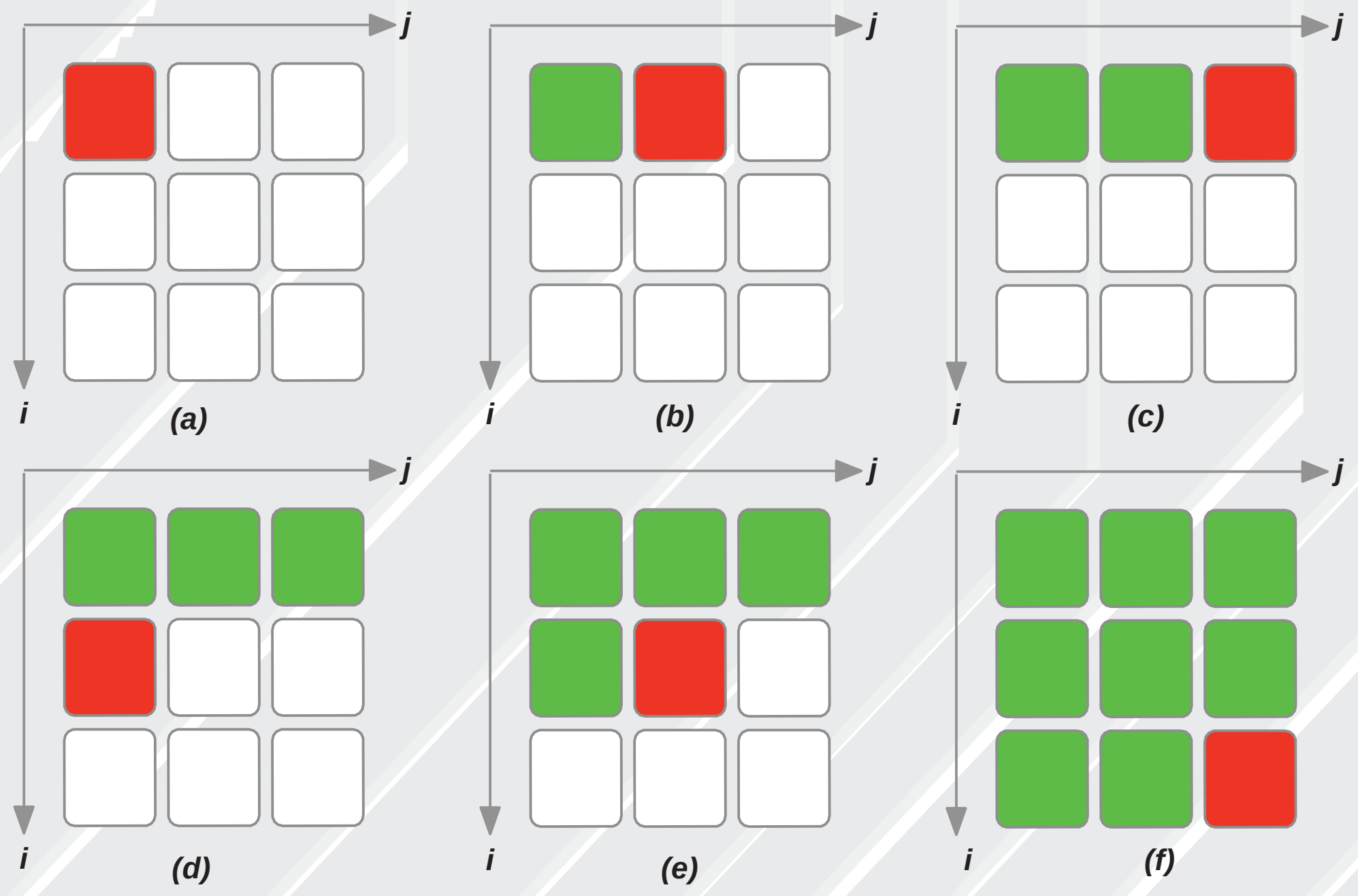

Fig 1: Triple Nested Loop Iteration Distribution

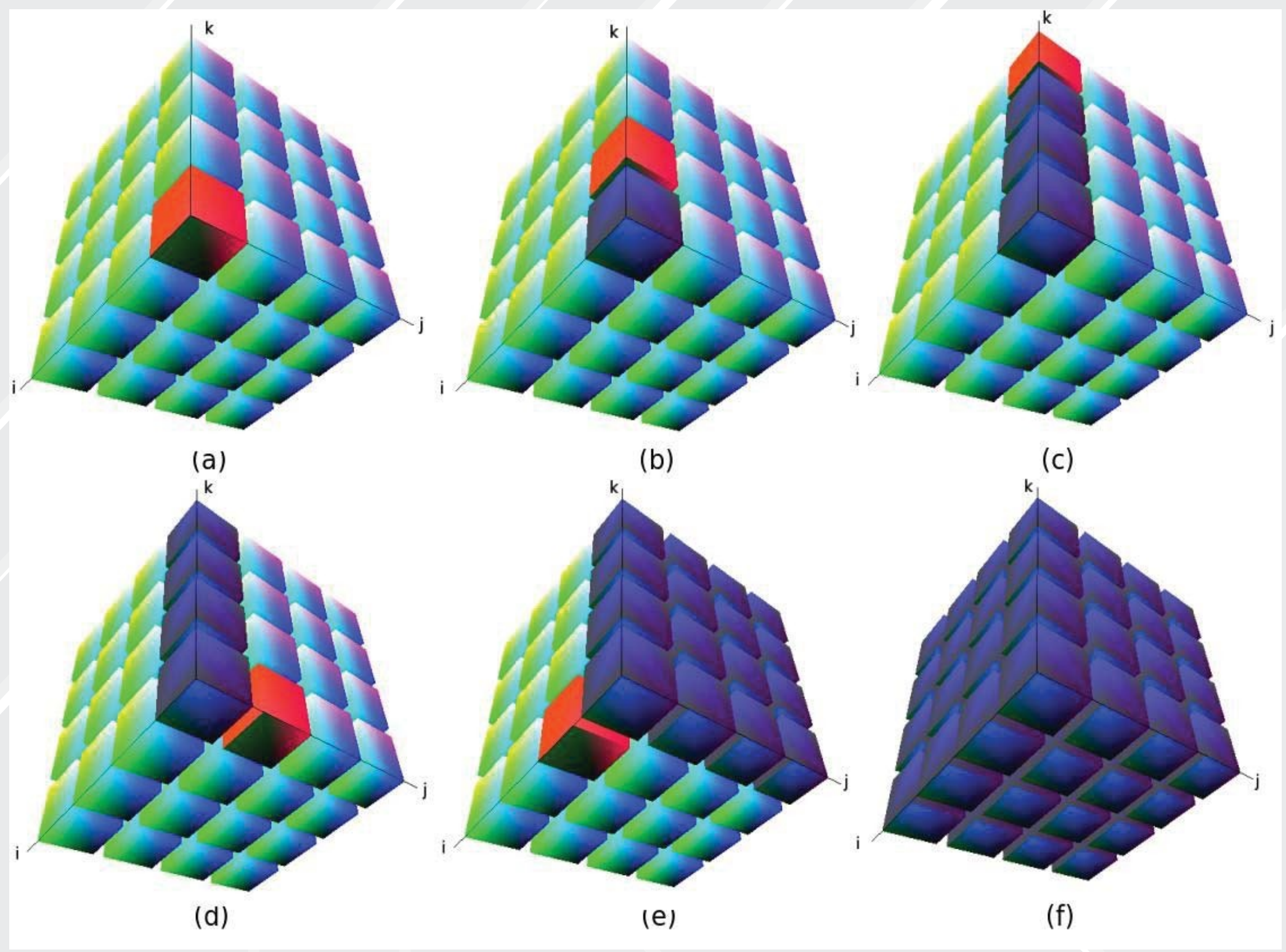

Fig 2: Double Nested Loop Iteration Distribution

OpenACC Implementation in OpenUH

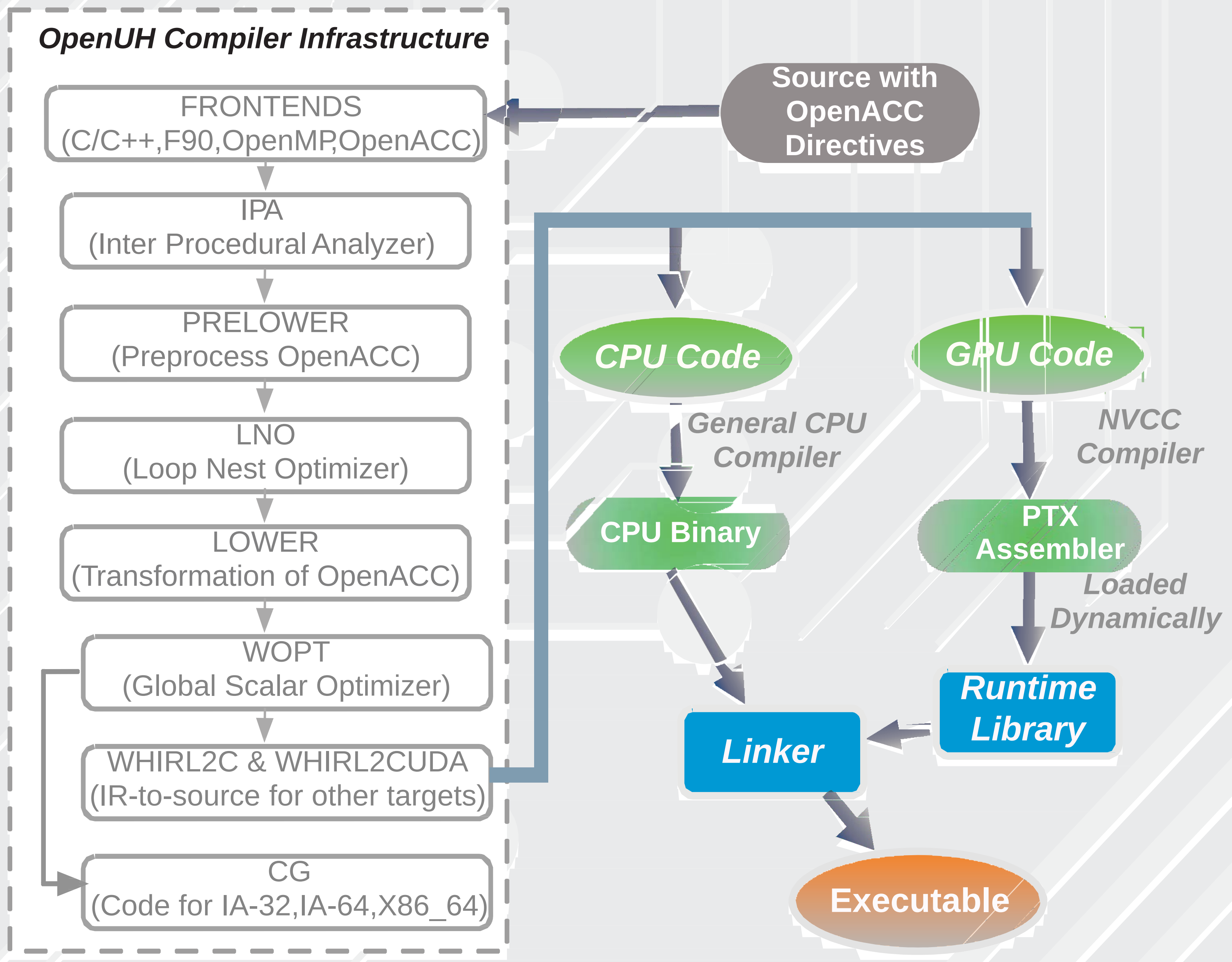
OpenUH Compiler Infrastructure
FRONTENDS
(C/C++,F90,OpenMP,OpenACC)
IPA
(Inter Procedural Analyzer)
PRELOWER
(Preprocess OpenACC)
LNO
(Loop Nest Optimizer)
LOWER
(Transformation of OpenACC)
WOPT
(Global Scalar Optimizer)
WHIRL2C & WHIRL2CUDA
(IR-to-source for other targets)
CG
(Code for IA-32,IA-64,X86_64)
Source with OpenACC Directives
CPU Code
GPU Code
General CPU Compiler
NVCC Compiler
CPU Binary
PTX Assembler
Loaded Dynamically
Runtime Library
Linker
Executable

Fig 3: OpenUH Framework for OpenACC

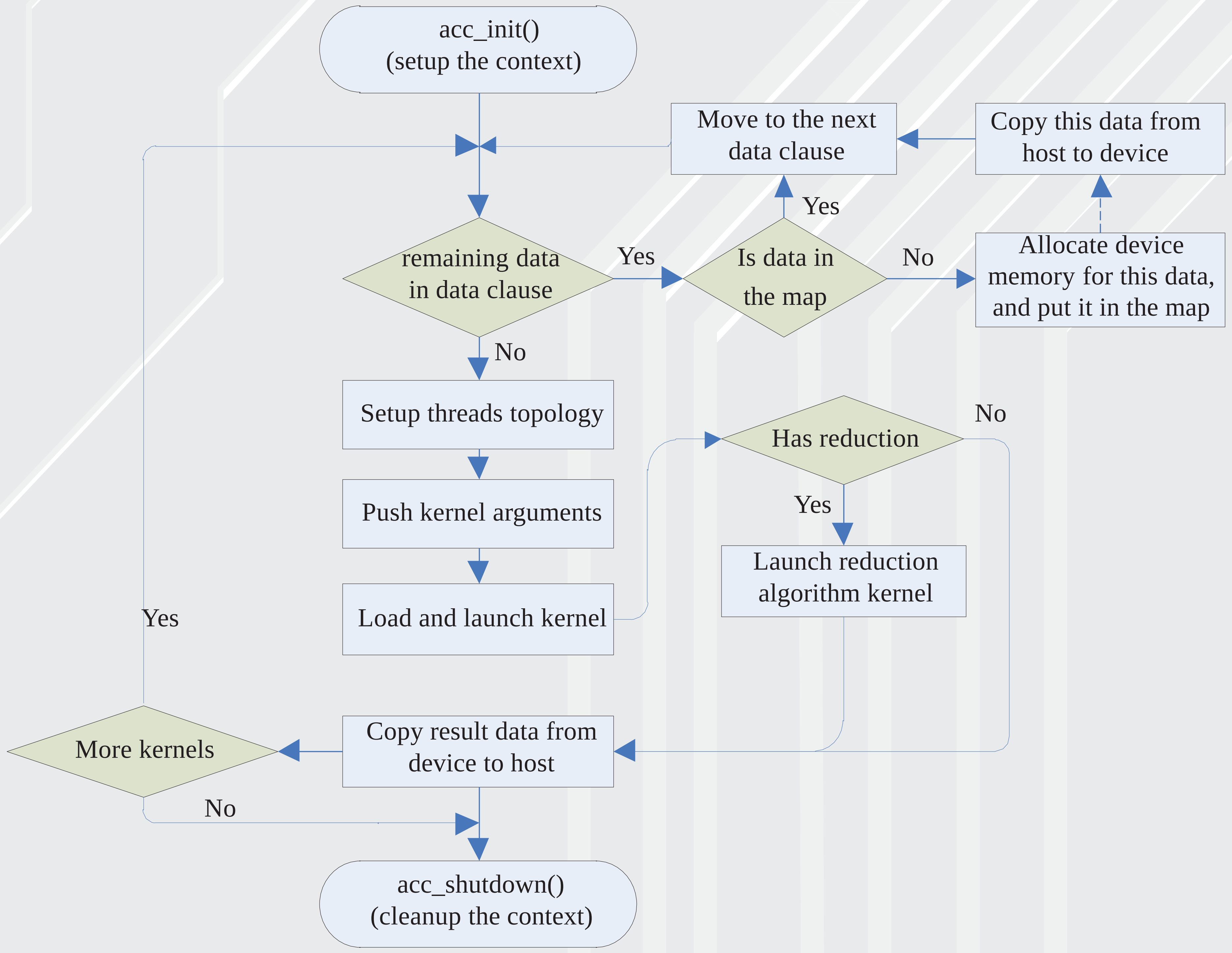
acc_init()
(setup the context)
remaining data in data clause
Yes
Is data in the map
Yes
Move to the next data clause
No
Allocate device memory for this data, and put it in the map
Copy this data from host to device
No
Setup threads topology
Push kernel arguments
Load and launch kernel
Has reduction
Yes
Launch reduction algorithm kernel
No
Copy result data from device to host
More kernels
Yes
No
acc_shutdown()
(cleanup the context)

Fig. 4: Execution Flow with OpenACC Runtime Library

Results

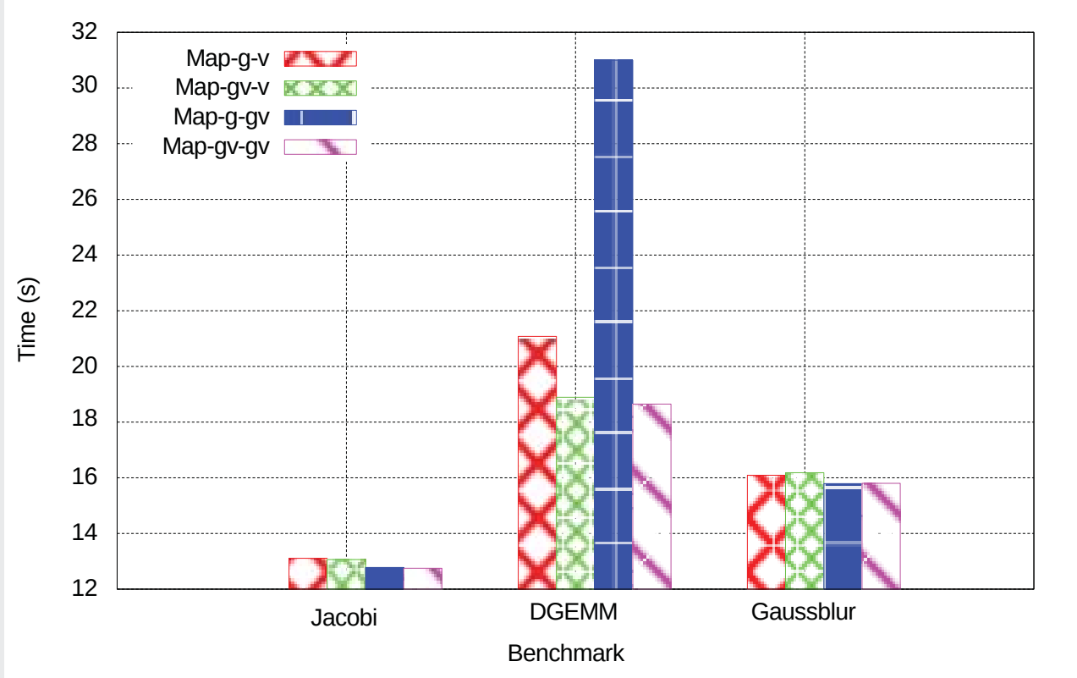

Fig. 5: Performance of Double Nested Loop Mapping

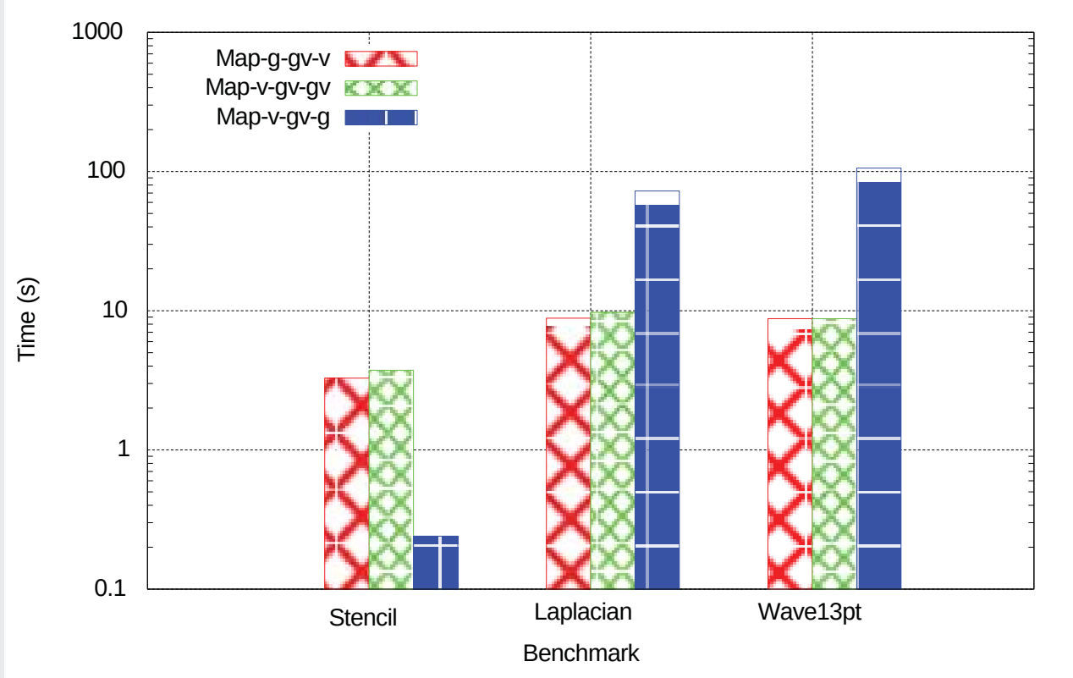

Fig. 6: Performance of Triple Nested Loop Mapping

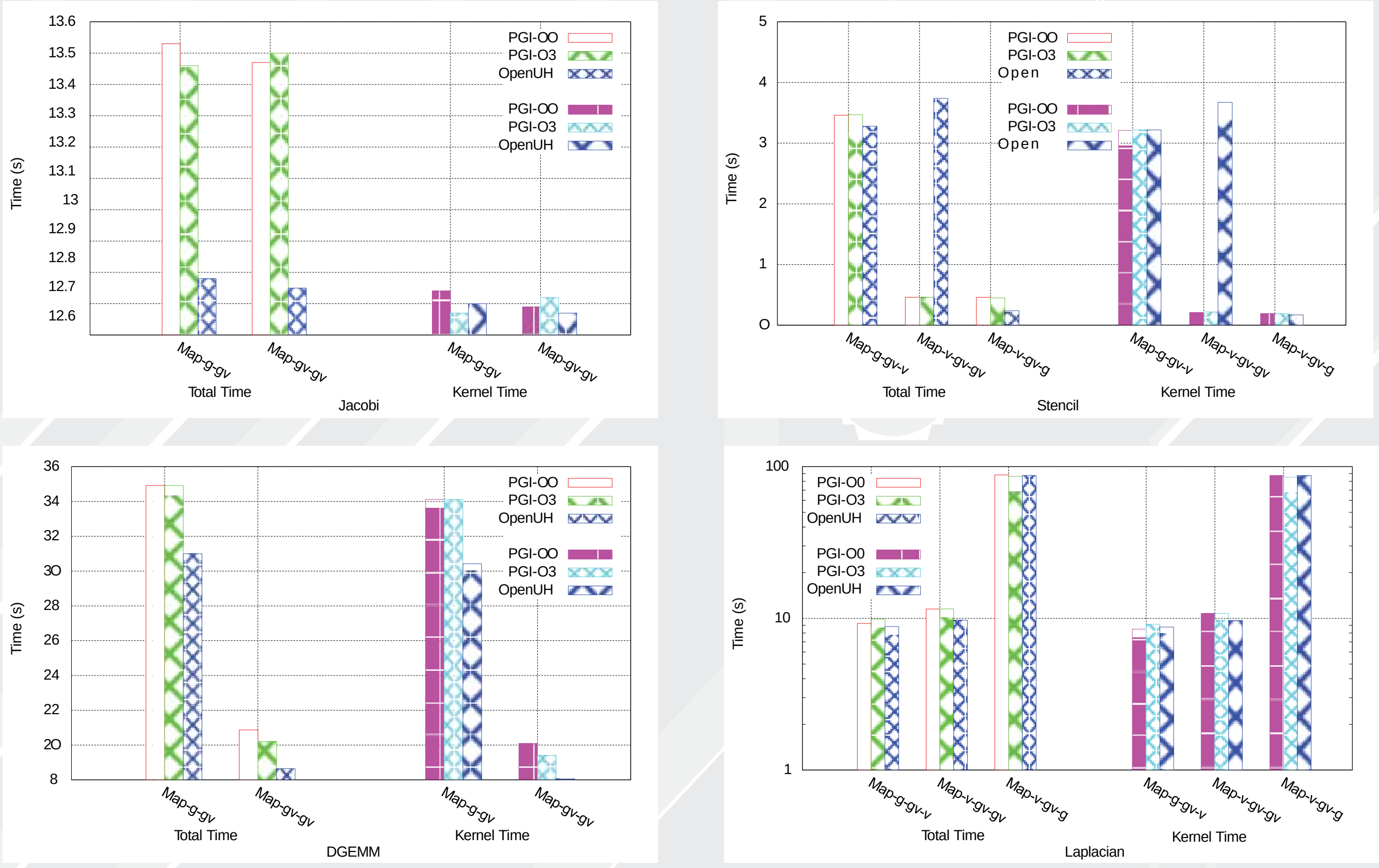

Fig. 7: Performance Comparison between PGI and OpenUH

Conclusion

- An open-source OpenACC compiler is created using OpenUH compiler framework
- Loop mapping mechanisms are designed to translate single loop, double loop and triple nested loop
- Competitive performance compared to a commercial OpenACC compiler
- Explore advanced compiler analysis and transformation techniques to further improve the performance in the future

References

Rengan Xu, Xiaonan Tian, Yonghong Yan, Sunita Chandrasekaran, and Barbara Chapman. Reduction Operations in Parallel Loops for GPGPUs, in *PMAM 2014*, Feb., 2014, Orlando, Florida, USA

Xiaonan Tian, Rengan Xu, Yonghong Yan, Zhifeng Yun, Sunita Chandrasekaran, Barbara Chapman. Compiling a High-Level Directive-Based Programming Model for GPGPUs , In *LCPC2013*, Sep. 2013, San Jose, CA, USA

Acknowledgment

This research was supported by NVIDIA and Department of Energy under Award Agreement No. DE-FC02-12ER26099. We also thank PGI for providing the compiler and support for the evaluation