UNIVERSITY NAME

DOCTORAL THESIS

Thesis Title

Author:
John SMITH

Supervisor:
Dr. James SMITH

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

Research Group Name
Department or School Name

May 9, 2023

Declaration of Authorship

I, John SMITH, declare that this thesis titled, "Thesis Title" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

UNIVERSITY NAME

Abstract

Faculty Name
Department or School Name

Doctor of Philosophy

Thesis Title

by John SMITH

The landscape of deep learning compiler frameworks has evolved rapidly with the development of various tools, such as TVM, deeptools, TensorFlow, DLVM, nGraph, and Glow. These frameworks offer unique optimizations to address computation and data movement challenges in deep learning accelerators (DLAs). These approaches include graph or IR level optimizations related to intra node memory access optimizations, operator fusion, and various tiling techniques. Despite their unique approaches, these frameworks primarily concentrate on node level optimizations that focus on increasing the performance of executing a scheduled operation in the graph and overlook the potential for inter-node data reuse optimizations within on-chip memory resources.

OnSRAM, a scratchpad management framework build to work with deep learning compilers, addresses this gap by focusing on internode scratchpad management in DLAs. OnSRAM exploits the static graph representations of deep learning models by identifying data structures that can be pinned to on-chip memory based on their reuse rate and cost of transfer from main host memory. OnSRAM has been implemented and evaluated on single DLA that contains a monolithic scratchpad and is integrated as part of a custom deep learning compiler framework.

In this work, we extend the capabilities of OnSRAM by introducing dynamic scratchpad allocation for static graph execution models using any number of scratchpads and allowing the implementation to be interfaced as a compiler extension rather than a core part of a bespoke compiler. This enhancement allows for more fine-grained control over on-chip memory resources, providing increased flexibility and adaptability to better accommodate diverse deep learning accelerators and memory access patterns. By optimizing inter-node data movement and storage across multiple scratchpads, our approach further reduces energy consumption and latency associated with inter-node communication.

Acknowledgements

The acknowledgments and the people to thank go here, don't forget to include your project advisor…

Contents

List of Figures

List of Tables

For/Dedicated to/To my…

Chapter 1

Deep Learning

1.1 Deep Learning Architectures

1.1.1 Perceptron

1.1.2 CNN

1.1.3 RNN

1.1.4 LSTM

1.2 Main Section 2

Chapter 2

Deep Learning Compilers

2.1 Graph Level Frameworks

2.1.1 TVM

2.1.2 TensorFlow

2.2 OnSRAM