THE UNIVERSITY OF CHICAGO

INCENTIVIZING FLEXIBILITY AND COOPERATION IN COMPUTER SYSTEMS USING FEEDBACK MECHANISMS

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCE
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

BY
MUHAMMAD HUSNI SANTRIAJI

CHICAGO, ILLINOIS
JUNE 2022

Dedication Text

Epigraph Text

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

ABSTRACT

Many computer systems and applications, from small embedded systems to large datacenter have deployment requirements. Meeting this deployment requirement in a dynamic environment is challenging and requires flexibility from the application and the system.

Flexibility is the ability to trade-off the value of one measure space by adjusting the value of another measure space. For example, both DNN and approximate computing applications can reduce their runtime latency by sacrificing their output accuracy. However, managing this flexibility is difficult. Prior approaches do not incentivize flexibility and cooperation. In the single stakeholder scenario where applications come from one stakeholder, they do not cooperate with the application and system knobs which makes the deployment inefficient in terms of energy, output accuracy, and performance. In the multistakeholder scenario, they do not incentivize the flexibility of applications which make flexible application produce higher output error.

Our first contribution is GRAPE and MERLOT, a hardware feedback mechanism to meet latency requirements while reducing energy usage. GRAPE is a hardware control system for GPU that provides a soft guarantee to meet the performance requirements. Meanwhile, MERLOT is a real-time hardware scheduler that provides a hard real-time guarantee.

Our second contribution is ALERT, runtime management for Deep Neural Networks that decrease output error or energy usage while meeting latency requirements. ALERT cooperates the whole flexibility from both application and the system. ALERT uses a probabilistic feedback mechanism that predicts the energy, performance, and output accuracy of the applications during the runtime.

The third contribution is SIM, runtime management that incentivizes the flexibility of applications in the multistakeholder scenario. Prior approaches inadvertently disincentivize the flexibility by enforcing flexible applications to adapt to meet their deployment requirement, thus encouraging greedy behavior where every stakeholder deploys networks that consume as

many resources as possible. SIM instead only enforces the adaptation to an application that holds the most resources. In each iteration, on behalf of the applications, SIM would make an application to a configuration that minimizes their output error such that the resource usage is either less than the application that holds most resources or higher as long as there are enough slack resources. SIM incentivize the deployment of flexible application by giving opportunity for all of the applications to fight for the slack resources by being flexible.

CHAPTER 1

INTRODUCTION

1.1 Motivation

Many computer systems and applications, from small embedded systems to large datacenter have deployment requirements. For example, DNN deployed in remote sensing infrastructure needs to minimize the output error while meeting latency deadline and energy budget. Meanwhile, a software application deployed in a large data center needs to minimize energy and meet Quality of Service (QoS). Meeting this deployment requirement in a dynamic environment is challenging and requires flexibility from the application and the system.

Flexibility is the ability to trade-off the value of one measure space by adjusting the value of another measure space. For example, both DNN and approximate computing applications can reduce their runtime latency by sacrificing their output accuracy. Meanwhile, the computer hardware can run in different DVFS configurations to adjust its performance and energy usage. However, managing this flexibility is difficult. Prior approaches do not incentivize flexibility and cooperation. In the single stakeholder scenario where applications come from one stakeholder, they do not cooperate with the application and system knobs which makes the deployment inefficient in terms of energy, output accuracy, and performance. In the multistakeholder scenario, they don't incentivize the flexibility of applications which make flexible application produce higher output error.

In this dissertation, we aim to incentivize the flexibility and cooperation in computer systems using a feedback mechanism. We focus on addressing these questions:

- How do we incentivize the hardware flexibility for interactive GPU applications?
- How do we incentivize the flexibility and cooperation of both application and system knobs for DNN deployment?

- How do we incentivize the flexibility and cooperation of anytime algorithms in the multistakeholder deployment scenario?

1.2 Contributions

This dissertation addresses the flexible management of applications and systems. We answer the first question by proposing GRAPE and MERLOT, a hardware feedback mechanism to meet latency requirements while reducing energy usage. We answer the second question by proposing ALERT, runtime management for Deep Neural Networks that decrease output error or energy usage while meeting latency requirements. We answer the third question by proposing SIM, runtime management that incentivizes the flexibility of applications in the multistakeholder scenario. We describe the overview of contributions below.

1.2.1 GRAPE

Many applications have performance requirements (e.g., real-time deadlines or quality-of-service goals) and we can save tremendous energy by tailoring resource usage so the application just meets its performance using the minimal resources. This problem is a classic constrained optimization: the performance goal is the constraint and energy consumption is the objective to be optimized. While several existing hardware approaches solve unconstrained optimizations (i.e., maximizing performance or minimizing energy), we are not aware of a hardware approach that minimizes GPU energy under an externally defined performance constraint. Therefore, we propose GRAPE, a hardware control system for GPUs that coordinates core usage, wavefront/warp action, core speed, and memory speed to deliver user-specified performance while minimizing energy. We implement GRAPE in VHDL (to demonstrate feasibility) and as an extension to GPGPU-Sim (for performance and power measurement).

1.2.2 MERLOT

Correct functioning of embedded systems requires strict timing guarantees. Traditionally, enforcing timing guarantees is the operating system's responsibility. The OS scheduler assigns sufficient resources to an application task to ensure it meets its deadline. Meeting hard real-time deadlines requires the scheduler to be conservative and allocate for the worst case timing; when behavior is not worst case, extra resources are allocated and energy is wasted. Some software schedulers reduce this energy waste by recognizing when an application is ahead of a worst case schedule and reclaiming unneeded resources, but they are fundamentally limited by (1) overhead and (2) a lack of visibility into low-level resource usage. Therefore, this paper advocates hardware assistance for energy management of hard real-time tasks. Specifically, we propose MERLOT, a hardware-based resource manager for GPUs that enforces software-specified timing guarantees with minimal energy. We implement MERLOT in VHDL and find that its performance, power, and area overheads are minuscule. We implement MERLOT in GPGPU-Sim to test timing and energy consumption and compare to two software-only approaches: one that always allocates for worst case timing and an intelligent approach that reduces resource usage when it recognizes better than worst case behavior.

1.2.3 ALERT

An increasing number of software applications incorporate runtime Deep Neural Networks (DNNs) to process sensor data and return inference results to humans. Effective deployment of DNNs in these interactive scenarios requires meeting latency and accuracy constraints while minimizing energy, a problem exacerbated by common system dynamics.

Prior approaches handle dynamics through either (1) system-oblivious DNN adaptation, which adjusts DNN latency/accuracy tradeoffs, or (2) application-oblivious system adaptation, which adjusts resources to change latency/energy tradeoffs. In contrast, this paper

improves on the state-of-the-art by coordinating application- and system-level adaptation. ALERT, our runtime scheduler, uses a probabilistic model to detect environmental volatility and then simultaneously select both a DNN and a system resource configuration to meet latency, accuracy, and energy constraints. We evaluate ALERT on CPU and GPU platforms for image and speech tasks in dynamic environments.

1.2.4 SIM

Recent work has proposed shared sensing infrastructure, multi-tenant embedded systems that different scientists can apply to deploy neural networks to process information collected from a third-party sensor. Examples include the Array of Things in Chicago and the National Ecological Observatory. Scheduling on these devices is difficult. They want to accommodate as much science as possible while keeping the neural network accuracy as high as possible. They are all subject to dynamic workload changes (as networks from different scientists enter and exit the system). Prior work has proposed anytime networks that flexibly operate across a wide range of accuracy/latency tradeoffs to minimize neural network error in dynamic environments. These anytime networks achieve great results when deployed by a single stakeholder (e.g., in a traditional embedded system); however, we find that anytime networks by themselves are a poor solution to the problem of shared sensing infrastructure as the networks are deployed by different stakeholders (i.e., the different scientists). Specifically, we find that traditional multi-stakeholder schedulers and traditional embedded scheduling based on worst-case execution time sacrifice accuracy compared to an optimal schedule. Perhaps worse, we find that applying prior approaches to SSI disincentivizes the deployment of flexible, anytime networks and encourages greedy behavior where every stakeholder deploys networks that consume as many resources as possible. In this paper, we identify this problem and propose a scheduler that incentivizes Anytime networks by rewording stakeholders that deploy such flexible networks and detecting and punishing bad actors that are inflexible and

consume the most resources.