GPU Performance Auto-tuning using Machine Learning

Tianyi David Han and Tarek S. Abdelrahman

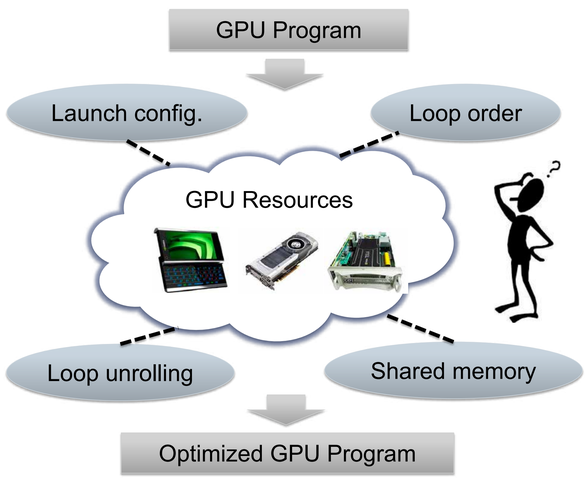
The Edward S. Rogers Sr. Department of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

NVIDIA CUDA RESEARCH CENTER

Summary: We explore the use of machine learning techniques to automatically select the best optimization configurations for computational photography applications.

Motivation

Why Auto-tuning?



Why Machine Learning?

	Performance	Speed	Effort
Analytical Modeling	●	●	●
Empirical Search	●	●	●
Machine Learning	●	●	●

Tuning Problem

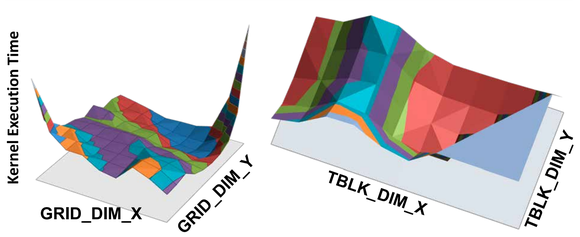
Computational photography applications

```
#kernel main tblock(BY,BX) thread(TY,TX)
for each row
  #unroll UF
  for each col {
    ... = input_img[row][col];
    // computation
    ... = input_img[col][row];
    // computation
    ...
    output_img[row][col] = ...;
  }
```

LOOP_ORDER

L1_CACHING

Optimization	# of Parameters	# of Configs
Loop order	1	2
Input caching	1	2
Inner loop unrolling	1	4
Launch configuration	4	≈ 6K
Total # of valid optimization configurations ≈ 38K		



Machine-Learning-based Auto-tuning

Predict the best-performing combination and configuration of optimizations for a kernel, by automatically building models based on how other kernels perform under various optimization configurations

Training Kernels

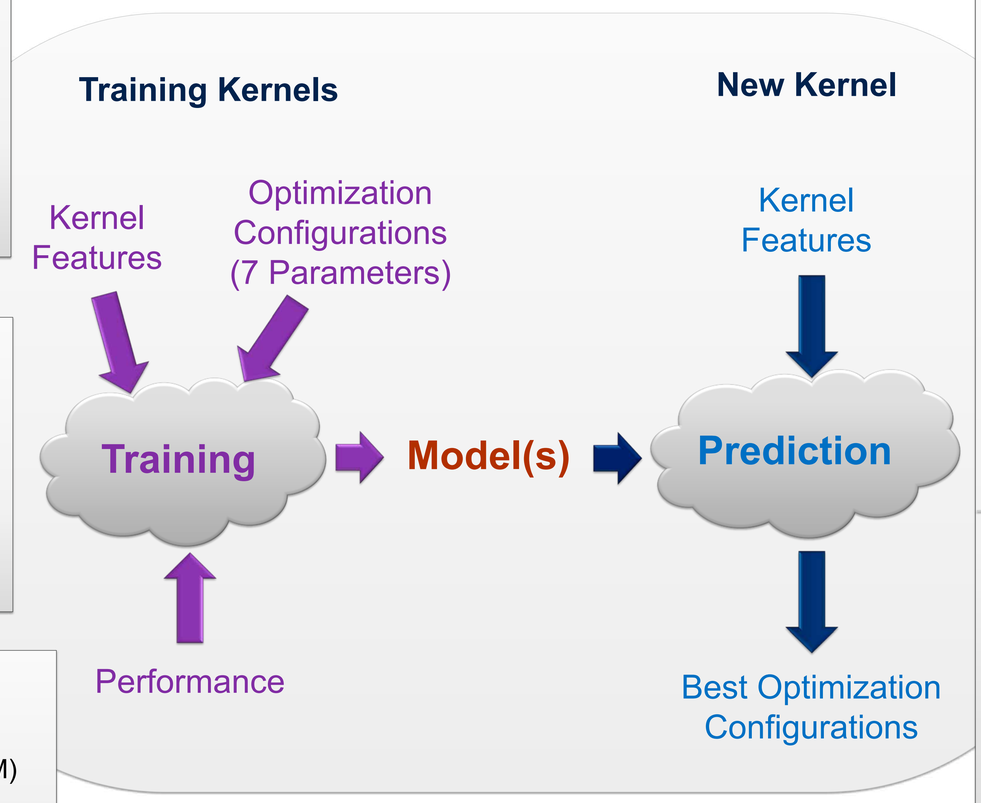
- Synthetically generated
- Random # of memory reads in the inner loop body
- Random computation length
- Random instruction dependencies

Kernel Features

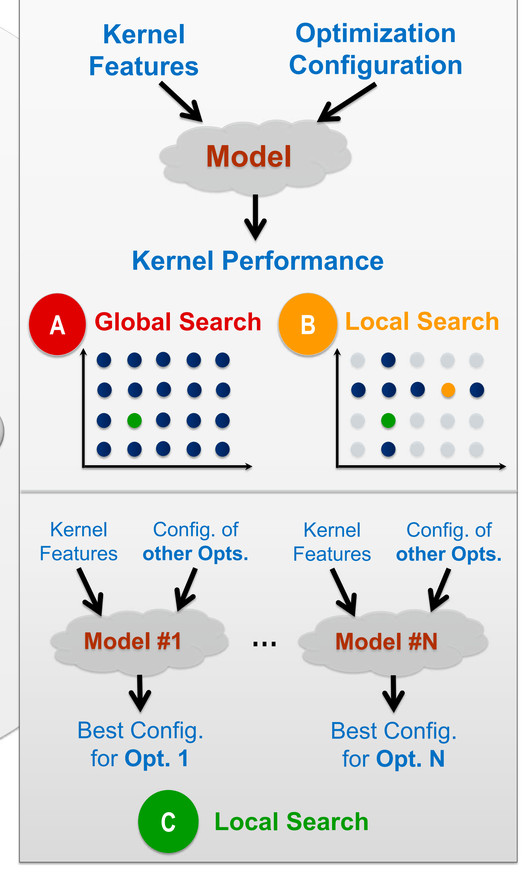
- Compile-time / run-time
- "Raw": memory access index expression
- Derived: # of transactions induced by memory access

Learning Algorithm

- Support Vector Machine (SVM)
- Decision Tree
- Random Forest



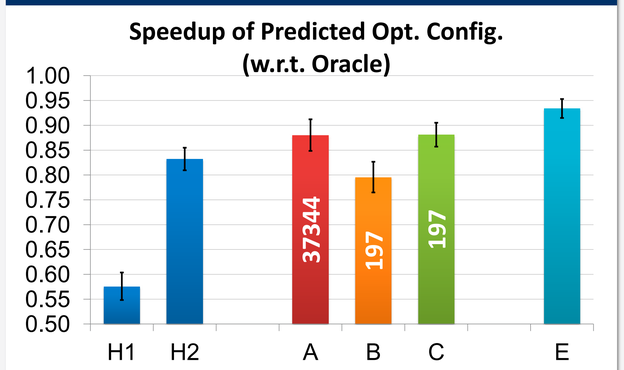
Learning Formulation



Experiment Setup

Kernels	• 120 synthetic kernels • 4 memory reads in inner loop body
GPU Platform	NVIDIA Tesla M2090, CUDA 5.5
Training Set	25% of the kernels (30 kernels)
Test Set	The remaining 90 kernels
Kernel Features	"Raw" features • Coefficients of each memory access index expression • Length of interleaved computation • Register usage
Learning Algorithm	Random Forest

Results



H1 -- a simple heuristic
H2 -- a heuristic with GPU architectural expertise
E -- empirical (local) search

What's Next

Study other optimization spaces and application domains

Further explore the design space of learning-based auto-tuning