GPU Approximation Acceleration For Scientific Applications*

Ang Li and Shuaiwen Leon Song
High Performance Computing Group, Pacific Northwest National Laboratory (PNNL)

Pacific Northwest NATIONAL LABORATORY

Proudly Operated by Battelle Since 1965

Overview

- Approximate computing, the technique that sacrifices certain amount of accuracy in exchange for substantial performance boost or power reduction, is one of the most promising solutions to enable power control and performance scaling towards exascale.
- In this position paper, we introduce a transparent, tractable and portable design framework for SFU-driven approximate acceleration on GPUs, providing fine-grained tuning for performance and accuracy trade-offs. Our design is software-based and requires no hardware or application modifications.

SFU Design & Implementation

Table 1: Invoking SP Transcendental Functions via CUDA and PTX APIs

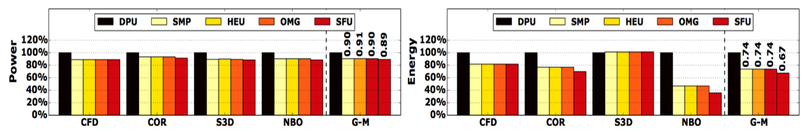
Func.	CUDA API Intrinsics		PTX API Instructions	
	SP-Accurate Version	SFU-Approximate Version	SP-Accurate Version	SFU-Approximate Version
x/y	x/y	__fdividef(x,y) & –ftz=true	div.rn.f32 %f3,%f1,%f2;	div.approx.ftz.f32 %f3,%f1,%f2;
$1/x$	1/x	Non-Provided	rcp.rn.f32 %f2,%f1;	rcp.approx.ftz.f32 %f2,%f1;
$\sqrt{x}$	sqrtf(x)	Non-Provided	sqrt.rn.f32 %f2,%f1;	sqrt.approx.ftz.f32 %f2,%f1;
$1/\sqrt{x}$	1/sqrtf(x)	rsqrtf(x) & –ftz=true	sqrt.rn.f32 %f2,%f1; rcp.rn.f32 %f3,%f2;	rsqrt.approx.ftz.f32 %f2,%f1;
x^y	powf(x)	__powf(x) & –ftz=true	Very Complex	lg2.approx.ftz.f32 %f3,%f1; mul.ftz.f32 %f4,%f3,%f2; ex2.approx.ftz.f32 %f5,%f4;
e^x	expf(x)	__expf(x) & –ftz=true	Very Complex	mul.ftz.f32 %f2,%f1, 0f3FB8AA3B; ex2.approx.ftz.f32 %f3,%f2;
$\log(x)$	logf(x)	__logf(x) & –ftz=true	Very Complex	lg2.approx.ftz.f32 %f2,%f1; mul.ftz.f32 %f3,%f2, 0f3F317218;
$\sin(x)$	sinf(x)	__sinf(x) & –ftz=true	Very Complex	sin.approx.ftz.f32 %f2,%f1;
$\cos(x)$	cosf(x)	__cosf(x) & –ftz=true	Very Complex	cos.approx.ftz.f32 %f2,%f1;

Table 2: Invoking DP Transcendental Functions via CUDA and PTX APIs

Func.	CUDA API Intrinsics		PTX API Instructions	
	DPU-Accurate Version	SFU-Approximate Version	DPU-Accurate Version	SFU-Approximate Version
x/y	x/y	Non-Provided	div.rn.f64 %fd3,%fd1,%fd2;	rcp.approx.ftz.f64 %fd3,%fd2; mul.f64 %fd5,%fd3,%fd1;
$1/x$	1/x	Non-Provided	rcp.rn.f64 %fd2,%fd1;	rcp.approx.ftz.f64 %fd2,%fd1;
$\sqrt{x}$	x/y	Non-Provided	sqrt.rn.f64 %fd2,%fd1;	rsqrt.approx.ftz.f64 %fd2,%fd1; rcp.approx.ftz.f64 %fd3,%fd2;
$1/\sqrt{x}$	1/sqrt(x)	Non-Provided	sqrt.rn.f64 %fd2,%fd1; rcp.rn.f64 %fd3,%fd2;	rsqrt.approx.ftz.f64 %fd2,%fd1;

- Invoking SP/DP transcendental functions via CUDA and PTX APIs.

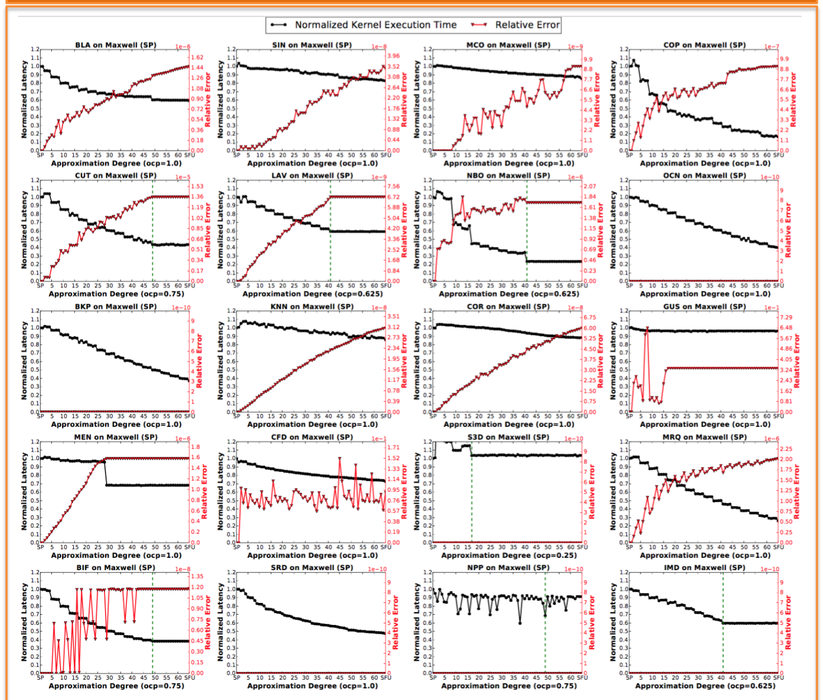
Measurement & Observation



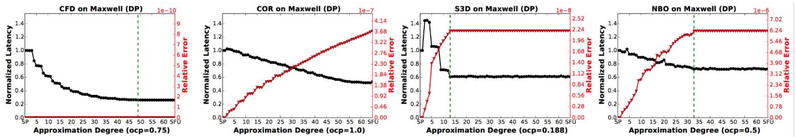
Figures show that:

- The power consumption using SPU/DPU is slightly higher than that using SFU, except for x/y in SP.
- Due to the huge performance differences between the SP and DP versions on the Maxwell platform, the overall energy consumption of DP versions (including their SFU approximations) is significant higher than that of the SP versions, despite of their lower power.

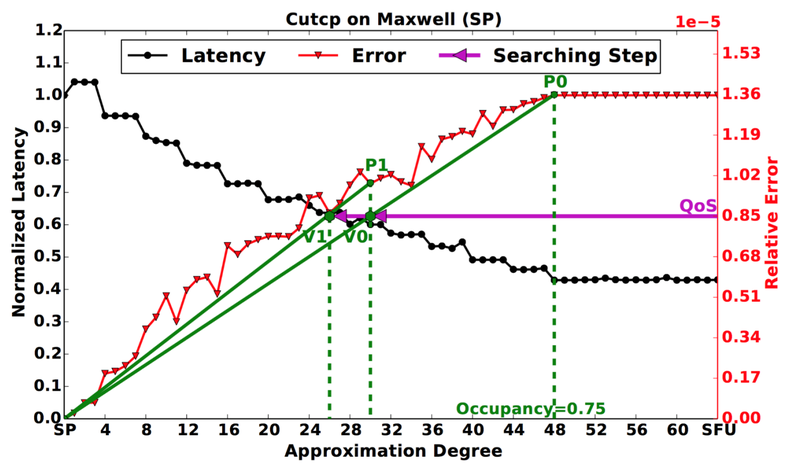
SFU-Driven Approximation Acceleration



- Performance-Accuracy Trade-offs for SP Applications on Maxwell GPU. The green dot line is based on the occupancy (i.e., ocp in the x-label).

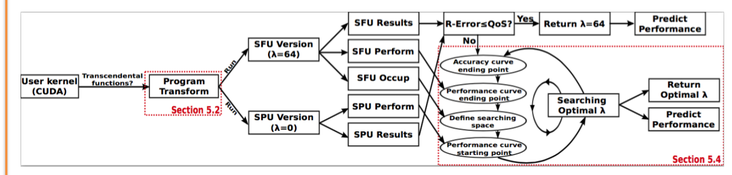


- Performance-Accuracy Trade-offs for DP Applications on Maxwell GPU.



- The proposed linear-approaching method(HEU) to locate the optimal λ for cutcp on a Maxwell GPU. The searching process terminates after two steps when QoS is satisfied.

The Overall Framework



- SFU-Driven Transparent Approximate Acceleration Framework.

Validation

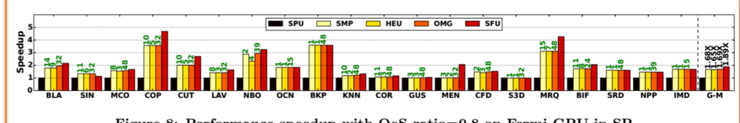


Figure 8: Performance speedup with QoS_ratio=0.8 on Fermi GPU in SP.

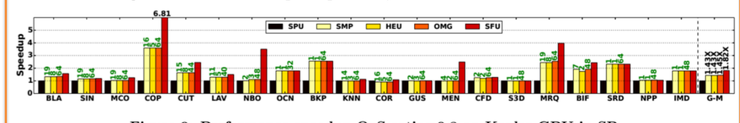


Figure 9: Performance speedup QoS_ratio=0.8 on Kepler GPU in SP.

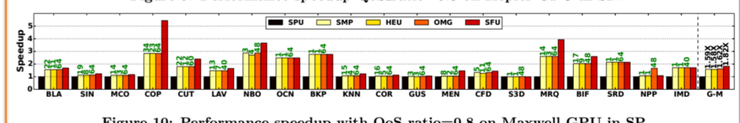


Figure 10: Performance speedup with QoS_ratio=0.8 on Maxwell GPU in SP.

- Figure 8, 9 and 10 illustrate the results for applying our framework to locate the optimal approximate degree of the 20 SP applications on the three GPU platforms with the QoS ratio7=0.8, respectively.

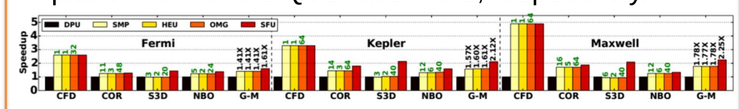


Figure 11: Performance speedup QoS_ratio=0.8 on Fermi, Kepler and Maxwell GPUs in DP.

- Figure 11 shows the results for the 4 DP applications. In these four figures, SPU/DPU is the baseline with no approximation. SFU is the maximum attainable speedup via the proposed approach when all the transcendental functions are calculated by the SFUs.

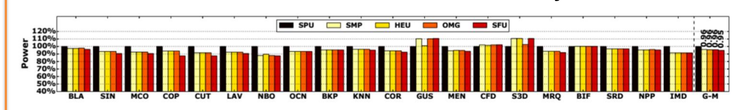


Figure 12: Normalized power reduction with QoS_ratio=0.8 on Maxwell Jetson-TX1 in SP.

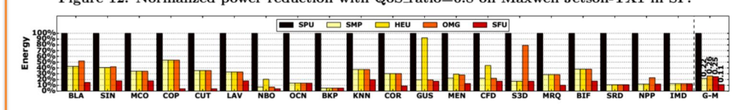


Figure 13: Normalized energy reduction with QoS_ratio=0.8 on Maxwell Jetson-TX1 in SP.

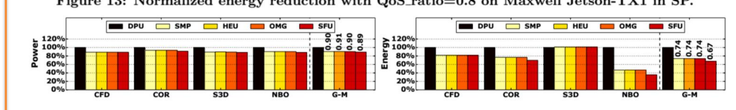


Figure 14: Normalized power and energy reduction with QoS_ratio=0.8 on Maxwell Jetson-TX1 in DP.

- Figure 12, 13 and 14 illustrate that the normalized power and energy reduction for SP and DP on the Maxwell Jetson-TX1 GPU (Platform 5 in Table 3) for calculating the transcendental functions in the 20 SP and 4 DP applications via the proposed methods (SMP, HEU and OMG, which is the most optimal can be achieved at that QoS level) under the QoS ratio=0.8.

U.S. DEPARTMENT OF ENERGY

* Initial results has been published in: SFU-Driven Transparent Approximation Acceleration on GPUs, Ang Li, Shuaiwen Leon Song, et.al., 30th ACM International Conference on Supercomputing (ICS), Istanbul, Turkey, 2016.