Enhancing Java Performance: vector API or iGPU?

Comparing the vector API and a novel SVM Buffer Java API for integrated GPU

Graduate

Reto Ehrensperger

Introduction: With its Vector API, Java provides an abstraction layer for executing vector operations on CPUs supporting Advanced Vector Extension (AVX) to speed up execution. On the other hand, shared virtual memory (SVM) API in OpenCL 2.0 enables the same virtual memory address the GPU and the application processor, reducing the overhead of memory operations. In this master thesis, we explore if integrated graphics cards (iGPUs) supported by SVM can improve performance compared to Vector API. We develop SVMBuffer API, an extension of OpenCL's SVM API for Java.

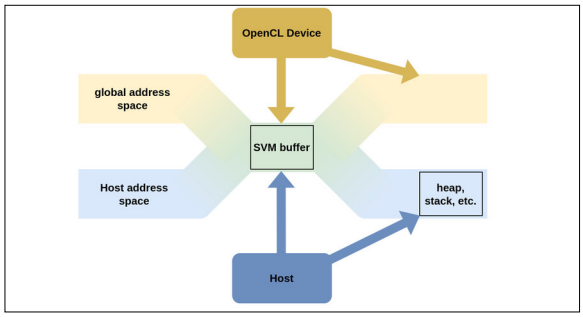
Approach / Technology: We explore the architecture and implementation of the SVMBuffer API and explain its integration with Java applications and the underlying OpenCL platform. We assess the performance implications of using SVMBuffer API compared to Vector API in various computational scenarios, namely, Discrete Fourier Transform (DFT), Matrix Multiplication (MM), Gaussian Blur (GB), and the Black-Scholes model (BS).

Result: The SVMBuffer API is particularly promising in operation-intensive scenarios such as DFT. DFT performance is improved from a workload size of 65536 by 1.4, compared to Vector API. At a workload size of 131072, performance can be increased by a factor of two.

If identical operations are performed on the iGPU and AVX, iGPU shows better performance only on DFT. Due to the slower memory access of the iGPU compared to the CPU, the Vector API outperforms the SVMBuffer API in MM, GB, and BS benchmarks.

Finally, we show that with specifically tailored kernels, matrix multiplication and Gaussian blur performance can be improved compared to translating the algorithm from Vector API to SVMBuffer API. Investigation of specialised kernels that reuse existing buffers and reduce the time spent on memory operations is recommended for further performance gain.

Representation of the address space of a SVM Buffer
Intel (OpenCL™ 2.0 Shared Virtual Memory Overview)

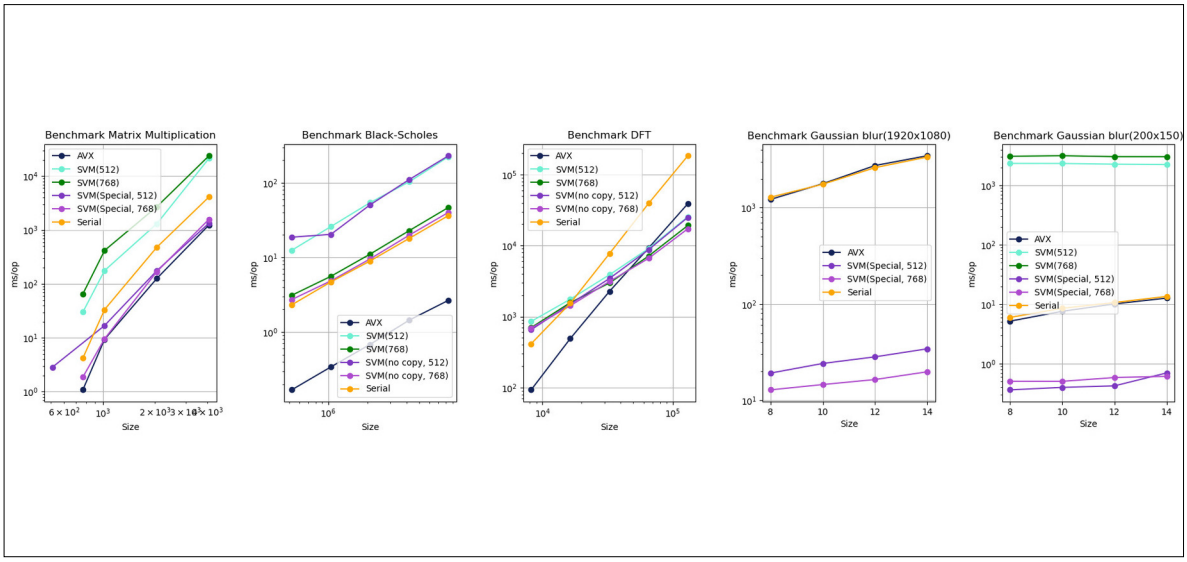


Hardware platforms used in benchamarking
Own presentment

Machine 1	
Processor	Intel® Core™ i7-1065G7
Cores	4 (8 Threads)
RAM	8 GB
GPU	Intel® Iris® Plus Graphics G7. Compute Units: 64, Max work group size: 256, Cores: 512
OpenCL (GPU)	OpenCL 3.0 NEO
OS	Manjaro Vulcan 23.1.0 (Linux Kernel 6.1.62-1)
Machine 2	
Processor	Intel® Core™ i7-1185G7
Cores	4 (8 Threads)
RAM	32 GB
GPU	Intel® Iris® Xe Graphics. Compute Units: 96, Max work group size: 512, Cores: 768
OpenCL (GPU)	OpenCL 3.0
OS	Windows 10

Table 2. Experimental Platforms

Benchmark performance results
Own presentment



Advisor
Prof. Dr. Mitra Purandare

Co-Examiner
Dr. Raphael Polig, IBM Research GmbH, Rüschlikon, ZH

Subject Area
Computer Science

Project Partner
Martin Stypinski (Veemg GmbH), Binz, ZH