codeplay®

Enable AI & HPC to be Open, Safe and Accessible to All

SYCL-BLAS as a oneMKL backend

Ouadie EL FAROUKI

Staff Software Engineer

oneAPI Community Forum - Math Special Interest Group - July 12, 2023

Summary

I. Introduction

II. SYCL-BLAS

III. SYCL-BLAS & oneMKL

IV. Future plans

V. Q&A

Introduction

BLAS

- **B**asic **L**inear **A**lgebra **S**ubroutines : Standardized in the late 70s/80s while trying to make it available & re-usable for everyone. Still in use today thanks to its **interface**.

- Main motives :
  - *Reusability through a common interface within higher level libraries & application.*
  - *Exploring hardware capabilities for efficiency & accuracy.*

- **BLAS** consists of 3 different levels of operations:
  - *level 1 : vector operations*
  - *level 2 : matrix - vector operations*
  - *level 3 : matrix - matrix operations*

Introduction

Who we are

Focus & missions :

- Expanding & maintaining SYCL BLAS library *(Operators, performance, portability etc..)*.
- Building & maintaining CI & benchmarks of the SYCL BLAS library across different platforms.
- Integrating & maintaining SYCL-BLAS as a oneMKL backend.

Introduction

BLAS Libraries

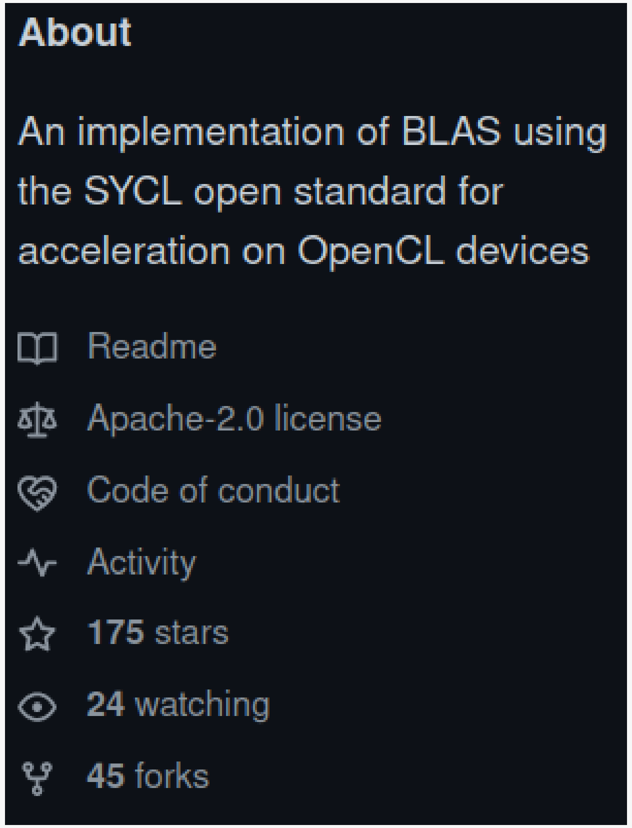
- Many implementations have been developed as standalone libraries, each with a specific '*goal in mind*' :
  - **Open-source**: *OpenBlas, clBLAST, LAPACK*
  - **Proprietary**:
    - Intel MKL (*Intel CPUs & GPUs*)
    - cuBLAS (*NVIDIA GPUs*)
    - rocBLAS (*AMD GPUs*)

- The open-source implementations usually offer a functional portability along with the possibility of tuning a specific routine on specific platforms.
- The proprietary implementations usually offer fine-tuned performance for corresponding native hardware.

SYCL-BLAS

Overview

- It's a SYCL & C++ based BLAS implementation started in 2015.
- It aims to be the reference BLAS implementation for tuneable performance and **portability**, as a community open-source project.
- Many papers have been published about it :
  - *Aliaga, José I., Ruymán Reyes, and Mehdi Goli. "**SYCL-BLAS: leveraging expression trees for linear algebra.**" Proceedings of the 5th International Workshop on OpenCL. 2017.*
  - *Aliaga, José I., Ruyman Reyes, and Mehdi Goli. "**SYCL-BLAS: combining expression trees and kernel fusion on heterogeneous systems.**" Parallel Computing is Everywhere 32 (2018): 349.*
  - *Sabino, Thales, and Mehdi Goli. "**Toward Performance Portability of Highly Parametrizable TRSM Algorithm Using SYCL.**" International Workshop on OpenCL. 2021.*



About

An implementation of BLAS using the SYCL open standard for acceleration on OpenCL devices

github.com/codeplaysoftware/sycl-blas/

SYCL-BLAS

Overview

- SYCL-BLAS Follows modern C++ specifications :
  - Can be used as Header-only library *(no linking needed, simple #include<sycl_blas.h>)*
  - Uses template meta-programming for maximum flexibility and performance *(Expression trees similar to Eigen's approach).*

- Developed to be the reference SYCL-Based implementation :
  - It can run on any SYCL compatible device *(through supporting SYCL implementation)*.
  - It can be compiled with different SYCL compilers :
    - *DPC++ / icpx*
    - *ComputeCPP*
    - *HIPSycl (OpenSYCL)*

SYCL-BLAS

Performance & Portability

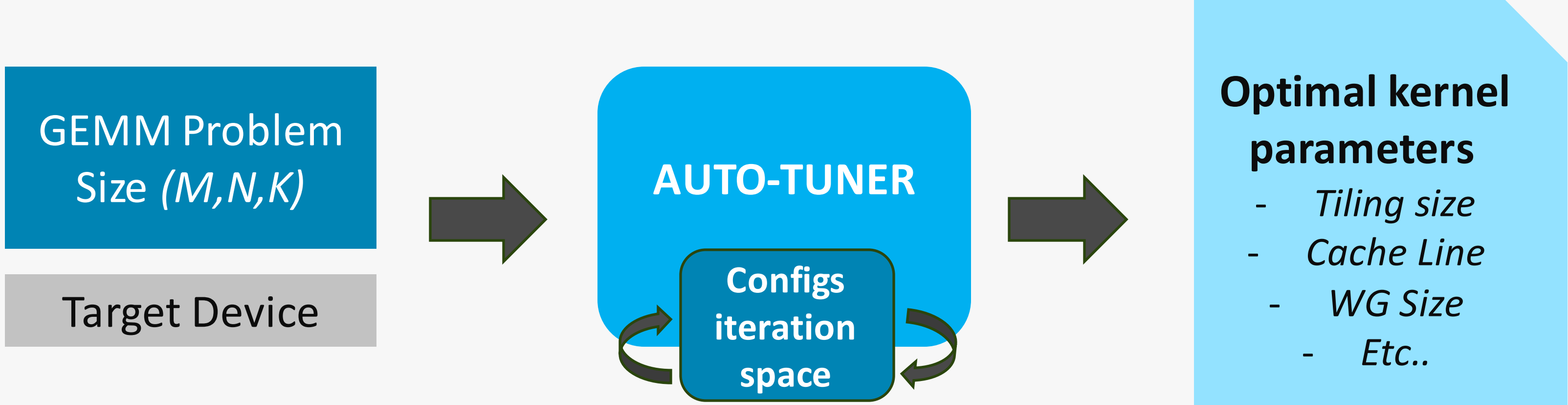
- SYCL BLAS Operators/Kernels support & can be tuned for specific targets thanks to templated parametrizations :
    - *NVIDIA GPUs*
    - *AMD GPUs*
    - *INTEL GPUs*
    - *Default CPU*

```
~ >> cmake -GNinja .. \
           -DSYCL_COMPILER={dpcpp | computecpp | hipsycl .. } \
           -DTUNING_TARGET={DEFAULT_CPU | NVIDIA_GPU | ..} \
           -DDPCPP_SYCL_TARGET={spir64 | nvptx64-nvidia-cuda | ..}
           -DDPCPP_SYCL_ARCH={sm_XY | gfxXYZ .. }
```

SYCL-BLAS

Auto-tuning : GEMM

- Automatically calculate the optimal parameters for GEMM on the given platform, maximizing performance*.



GEMM Problem Size *(M,N,K)*

Target Device

AUTO-TUNER

Configs iteration space

Optimal kernel parameters

- *Tiling size*
- *Cache Line*
- *WG Size*
- *Etc..*

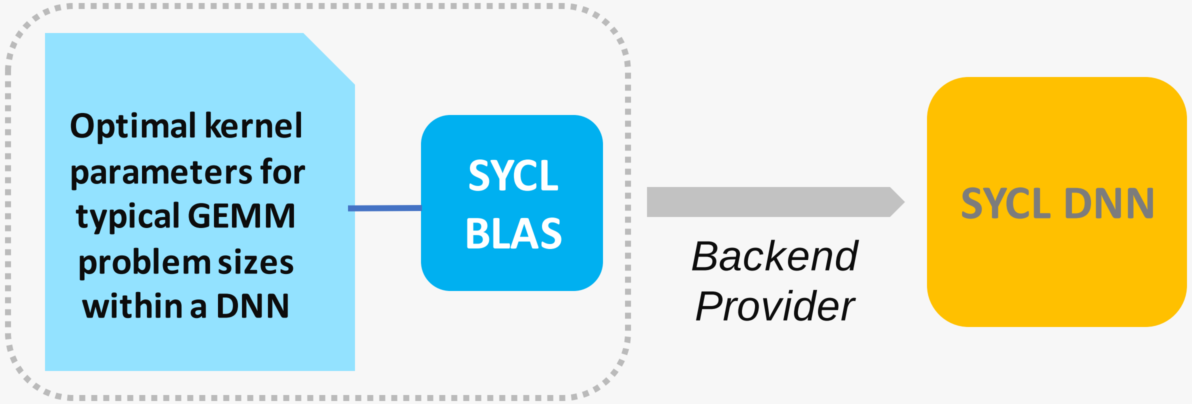
Template parameters used once selected to instantiate GEMM kernels at compile time.

*: *github.com/codeplaysoftware/sycl-blas/tools/auto_tuner*

SYCL-BLAS

Some ongoing use-cases

- SYCL-DNN *(A Deep Neural Networks SYCL Library) exposes an optional* ***SYCL-BLAS backend*** *to be used for some operations (e.g., Memory handling, Matrix Multiplication, Reduction etc..).*

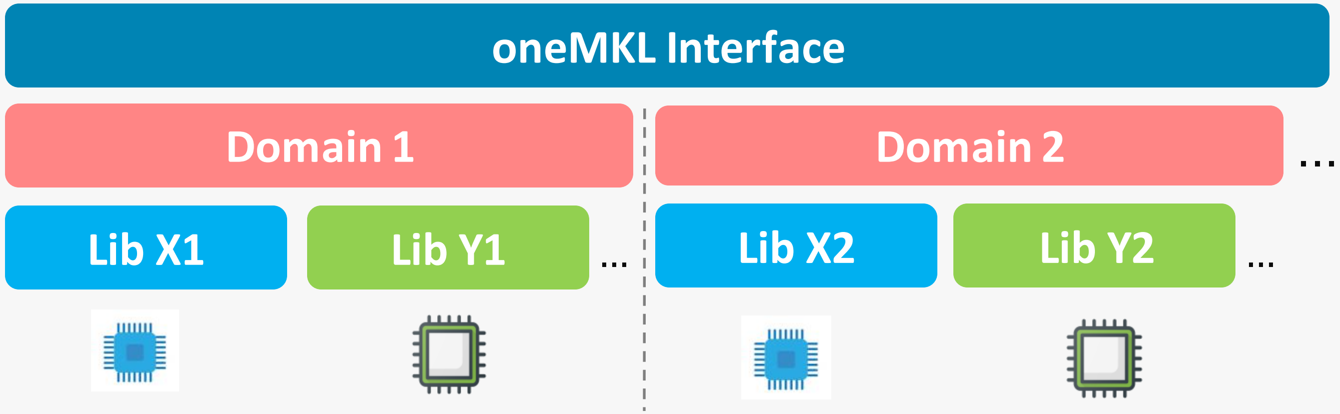

Optimal kernel parameters for typical GEMM problem sizes within a DNN
SYCL BLAS
Backend Provider
SYCL DNN

- A SYCL Based *JPEG-Compression* app has been developed using SYCL-BLAS Gemm as a **tunable building block** for performance portability.

OneMKL

Overview

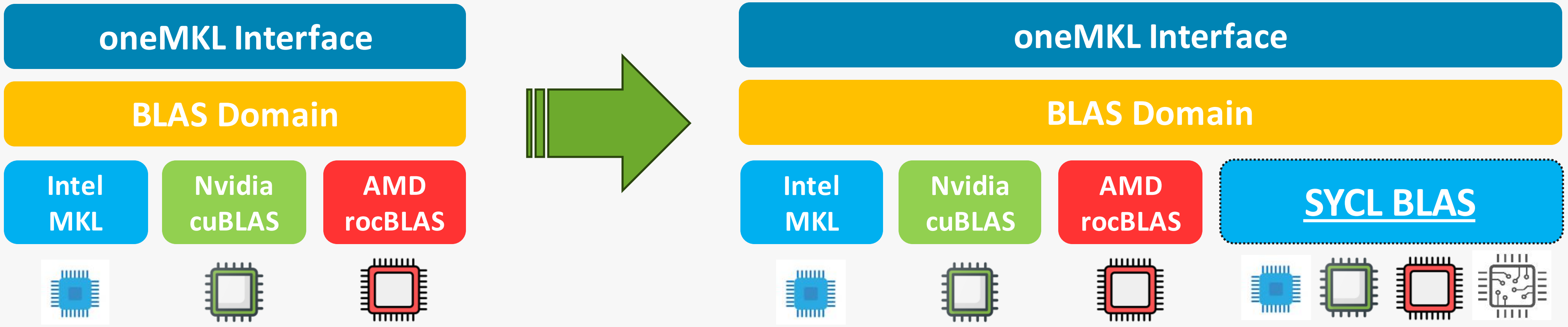
- oneAPI Math Kernel Library :
  - Open-source implementation of the DPC++ oneMKL Spec & part of the oneAPI 10 core specs.
  - It's a library gathering optimized common mathematical routines grouped into *'domains'* : ***BLAS***, *LAPACK, RNG, DFT etc..*
  - It supports multiple devices through relevant libraries (***backends***) underneath.



SYCL-BLAS & oneMKL

SYCL-BLAS Backend

- For the BLAS domain in oneMKL, there are backends relying on proprietary third-party libraries *(MKL, cuBLAS, rocBLAS),* each supporting a set of native devices.
- Introducing SYCL-BLAS as a BLAS backend to oneMKL aims to support portability in *oneMKL* interface within the backend itself while remaining open-source :



SYCL-BLAS & oneMKL

SYCL-BLAS Backend

- To use SYCL-BLAS as backend, users don't need to install it by themselves. It's taken care of by the built-in configurations.

```
  ~ >> cmake -GNinja .. \
             -DENABLE_SYCLBLAS_BACKEND=ON \
             -DENABLE_MKLCPU_BACKEND=OFF \
             -DENABLE_MKLGPU_BACKEND=OFF \
             -DTARGET_DOMAINS=blas
```

- Use local version of *SYCL-BLAS* by specifying target directory.

```
  ~ >> cmake -GNinja .. \
             -DENABLE_SYCLBLAS_BACKEND=ON \
             -DENABLE_MKLCPU_BACKEND=OFF \
             -DENABLE_MKLGPU_BACKEND=OFF \
             -DTARGET_DOMAINS=blas \
             -Dsycl_blas_DIR=[INSTALLED DIR
OF SYCLBLAS headers]
```

SYCL-BLAS & oneMKL

Usage Model

- **Run-time dispatching :** The application is linked with the oneMKL library, and the backend is loaded at run-time based on device vendor.

```
#include "oneapi/mkl.hpp"

gpu_dev = sycl::device(sycl::gpu_selector_v);

sycl::queue gpu_queue(gpu_dev);

oneapi::mkl::blas::column_major::gemm(gpu_queue, transA, transB, m, ...);
```

SYCL-BLAS & oneMKL

Usage Model

- **Compile-time dispatching** : The application used a templated backend selector API where the template parameters specify the required backend and third-party libraries. The application is linked with the required oneMKL backend wrapper libraries.

```
#include "oneapi/mkl.hpp"

gpu_dev = sycl::device(sycl::gpu_selector_v);
sycl::queue gpu_queue(gpu_dev);

oneapi::mkl::backend_selector<oneapi::mkl::backend::syclblas>
gpu_selector(gpu_queue);

oneapi::mkl::blas::column_major::gemm(gpu_selector, transA, transB, ...);
```

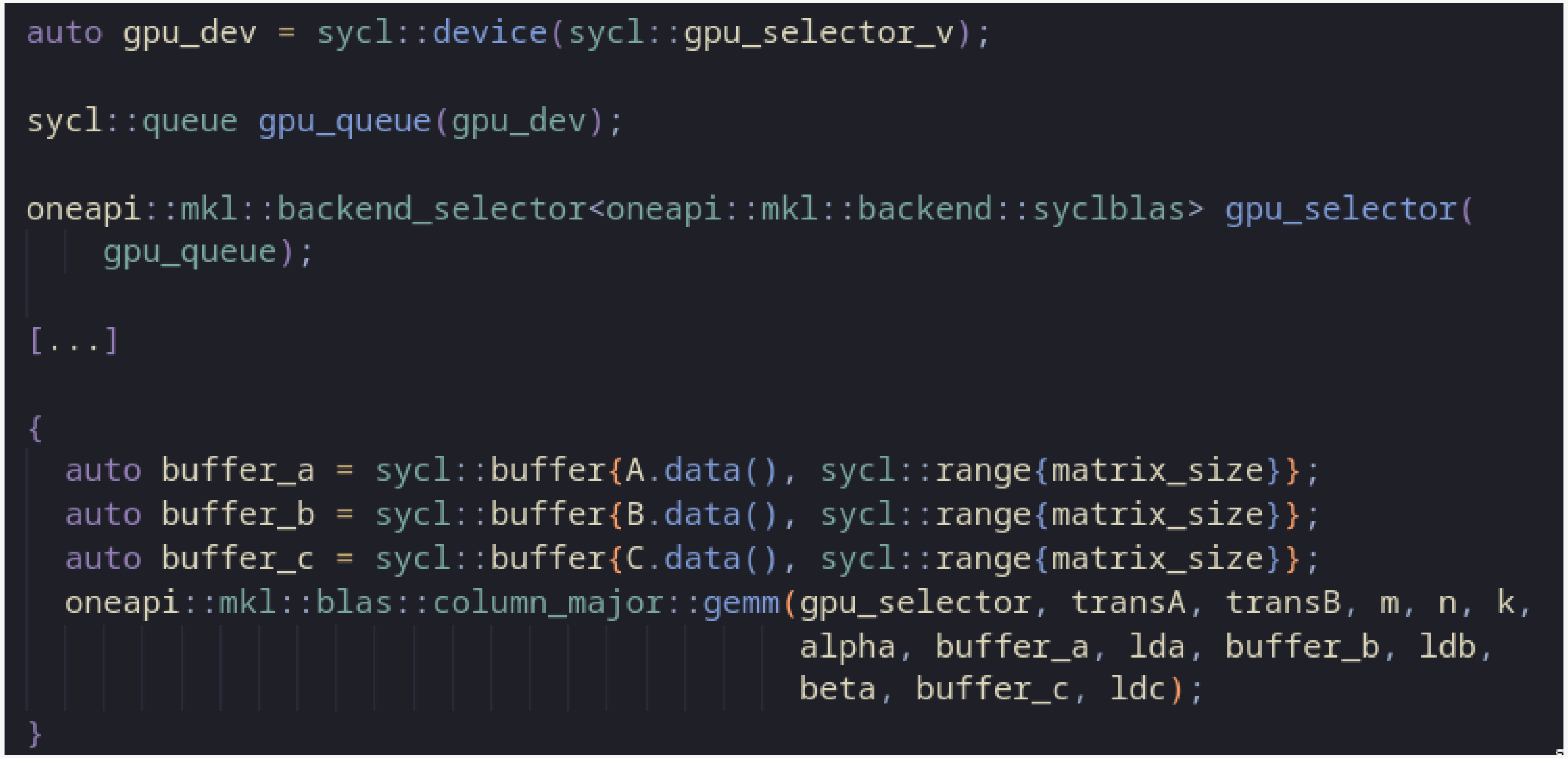
SYCL-BLAS & oneMKL

Example

Device Selection

Backend Selection

Preparing buffers & launching kernel



```
auto gpu_dev = sycl::device(sycl::gpu_selector_v);

sycl::queue gpu_queue(gpu_dev);

oneapi::mkl::backend_selector<oneapi::mkl::backend::syclblas> gpu_selector(
    gpu_queue);

[...]

{
  auto buffer_a = sycl::buffer{A.data(), sycl::range{matrix_size}};
  auto buffer_b = sycl::buffer{B.data(), sycl::range{matrix_size}};
  auto buffer_c = sycl::buffer{C.data(), sycl::range{matrix_size}};
  oneapi::mkl::blas::column_major::gemm(gpu_selector, transA, transB, m, n, k,
                                        alpha, buffer_a, lda, buffer_b, ldb,
                                        beta, buffer_c, ldc);
}
```

Future plans

SYCL-BLAS is in active development and there are more upcoming features.

- USM full support.
- Row-major support.
- Increase operator implementation *(currently ~60%)*.
- Complex type support.

SYCL-BLAS is the portable backend of oneMKL.

Update

Name changing of the library to align with *oneAPI*'s strategy & emphasize its main feature : **portability**.

SYCL-BLAS

Update

Name changing of the library to align with *oneAPI*'s strategy & emphasize its main feature : **portability**.

portBLAS

codeplay®

Enable AI & HPC to be Open, Safe and Accessible to All

Q&A

@codeplaysoft

info@codeplay.com

codeplay.com

Notices & Disclaimers
Intel technologies may require enabled hardware, software or service activation.
No product or component can be absolutely secure.
Your costs and results may vary.