IWOCL / SYCLcon 2021

On measuring the maturity of SYCL implementations by tracking historical performance improvements

Wei-Chen Lin
and Tom Deakin, Simon McIntosh-Smith

Introduction

Select: | Compile with: | Benchmark on:

Representative set of HPC style mini-apps

- Memory-bandwidth bound:
  - BabelStream
  - CloverLeaf (complex, high kernel count)
- Compute-bound
  - BUDE

Historical SYCL compilers

- ComputeCpp (Codeplay)
  - Jul. 2018 ~ Nov. 2020
- oneAPI DPC++ (Intel)
  - Mar. 2020 ~ Jan. 2021
- hipSYCL (Heidelberg University)
  - Sep. 2019 ~ Jan. 2021

Alternative HPC frameworks

- CUDA
- OpenCL
- OpenMP
- Kokkos

HPC Platforms w/ SYCL support

- Intel Cascade Lake Xeon CPU
- AMD Rome CPU
- Intel Gen 9.5 GPU
- NVidia V100 GPU

SYCL landscape

oneAPI

hipSYCL

ComputeCpp

	DPC++	ComputeCpp	hipSYCL
OpenMP (CPU)	●	○	●
OpenCL SPIR (CPU/GPU)	●	●	◐[a]
Intel Level Zero (CPU/GPU)	●	○	◐[a]
Nvidia (GPU)	◐[b]	◐[c]	●[d]
AMD ROCm (GPU)	○	○	●

[a] Built on DPC++, experimental and work in progress
[b] PTX, experimental
[c] PTX, experimental, incomplete and discontinued
[d] CUDA

Hardware platform

Name	Architecture	Short name	Device Type	Peak Mem. BW (GB/s)	Peak FP32 FLOP/s (GFLOP/s)
NVIDIA Tesla V100	Volta	v100-isambard	Discrete GPU	900	14000
Intel UHD P630 (Intel Xeon E2176G)	Gen9.5	uhdp630-devcloud	Integrated GPU + CPU	42.6	460
Intel Xeon Gold 6230 (20-cores)	Cascade Lake	cxl-isambard	HPC CPU (2-socket)	281.6	4096
AMD EPYC 7742 (64-cores)	Zen2 (Rome)	rome-isambard	HPC CPU (2-socket)	409.6	9216

BabelStream

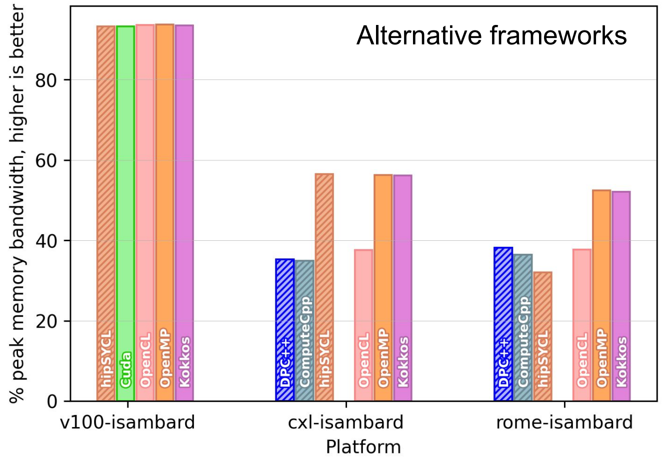
- Port of the STREAM benchmark to many languages, SYCL included
- Memory-bandwidth bound
- Single kernel; repeated for consistency
- Measurements in GB/s
- Source code publicly available on GitHub
  - https://github.com/UoB-HPC/BabelStream

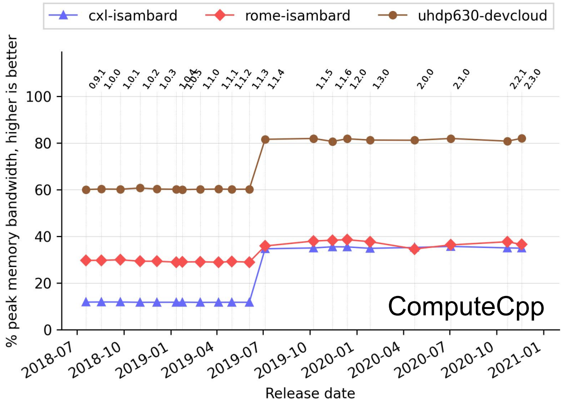
Algorithm 1 BabelStream Triad kernel

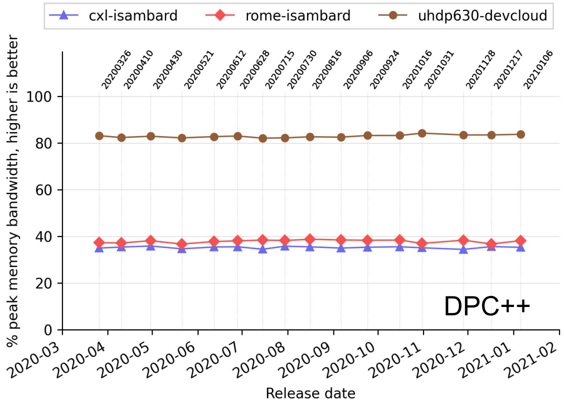
procedure TRIAD($a[], b[], c[], scalar, n$) ▷ a, b, c are arrays of size n
    for $i \leftarrow 0, n$ **do**
        $a[i] \leftarrow b[i] + scalar * c[i]$

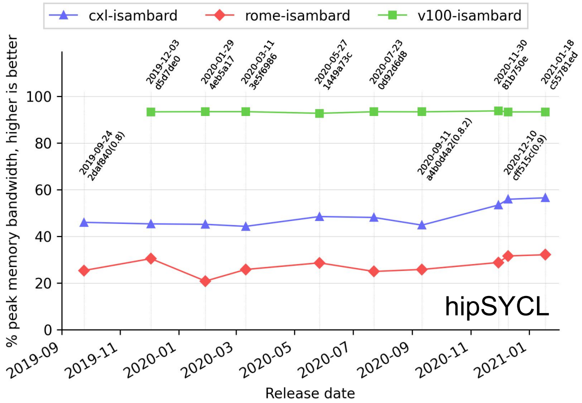
Results: BabelStream

- ComputeCpp: ~ 70% performance of alternative frameworks
- DPC++: Highly consistent
- hipSYCL: Major performance uplift after version 0.9



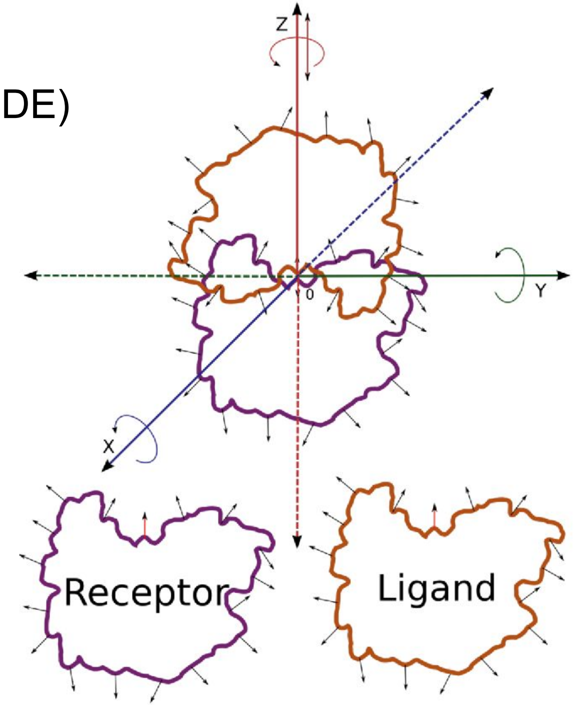






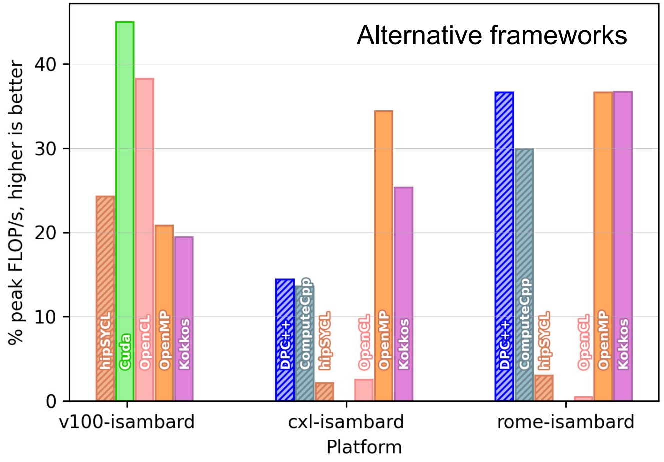
miniBUDE

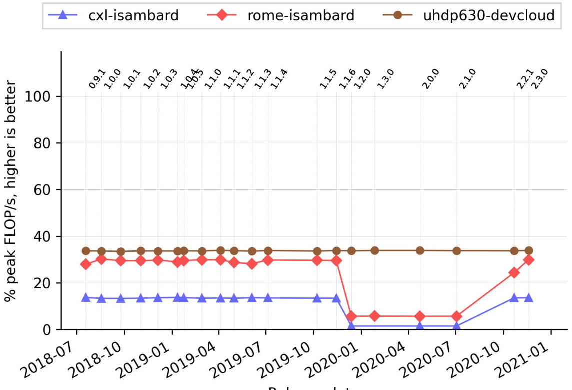
- Proxy application of the Bristol University Docking Engine (BUDE)
- Compute bound
- Single kernel; repeated for consistency
- No hierarchical parallelism
- Measurements in GFLOPS/s
- Source code publicly available on GitHub
  - https://github.com/UoB-HPC/bude-portability-benchmark



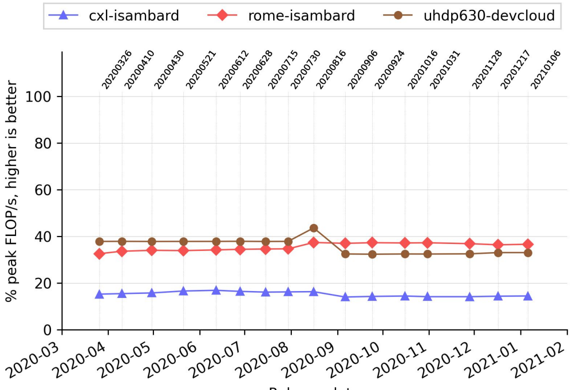
Results: BUDE

- ComputeCpp: Performance regressions resolved recently
- DPC++: Highly consistent, competitive
- hipSYCL: Consistently low performance on the CPU end

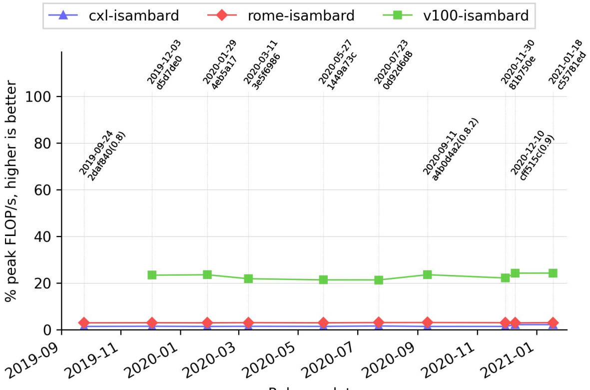




ComputeCpp



DPC++



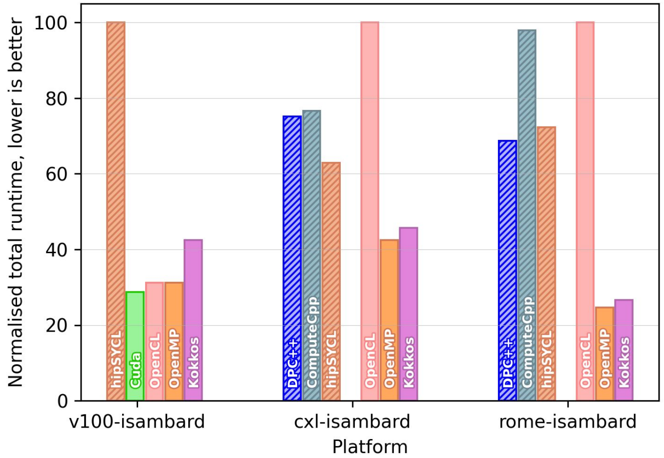
hipSYCL

CloverLeaf

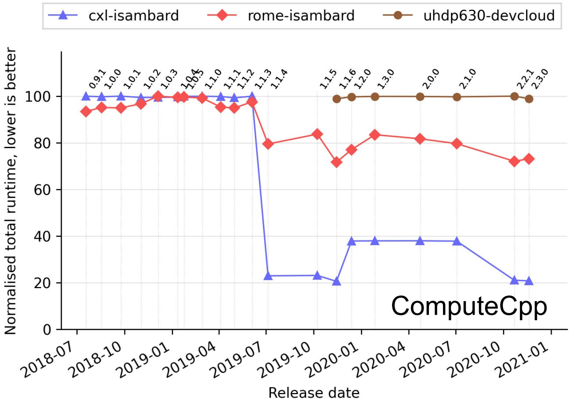
- Proxy application for 2D hydrodynamics
- Memory-bandwidth bound; kernel traverses structured grid
- 170+ unique kernels, largest codebase (8kLOC)
- Measurements in total runtime for 2995 iterations
- Source code publicly available on GitHub
  - https://github.com/UoB-HPC/cloverleaf_sycl

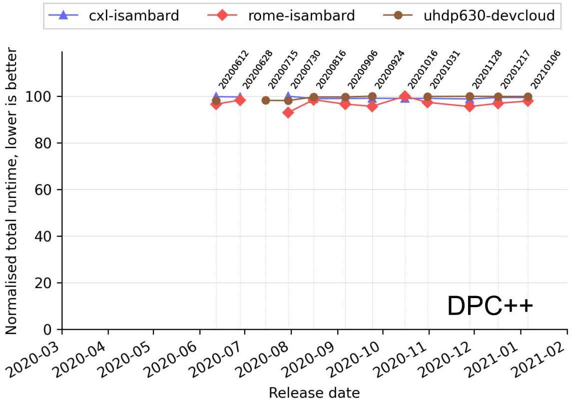
Results: CloverLeaf

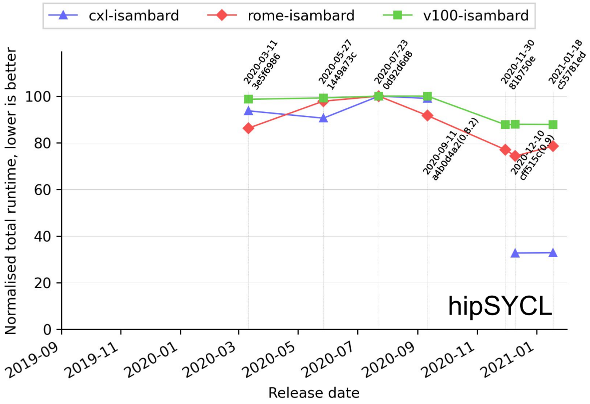
- ComputeCpp: Major improvements (~ 80%)
- DPC++: Highly consistent
- hipSYCL: Significant improvements (~40%)



Alternative frameworks







Note: normalised runtime per chart; values are not comparable across compilers

Summary

- All SYCL implementations approaching maturity
  - SYCL2020 alignment
  - Improved software and hardware platform support
  - Stabilised performance, trending upwards
- All mini-apps are on publicly available GitHub repositories