14th International Workshop on OpenCL and SYCL

IWOCL 2026

Evaluating the AdaptiveCpp Single-Pass (SSCP) SYCL compiler for GROMACS on Modern AMD Accelerators

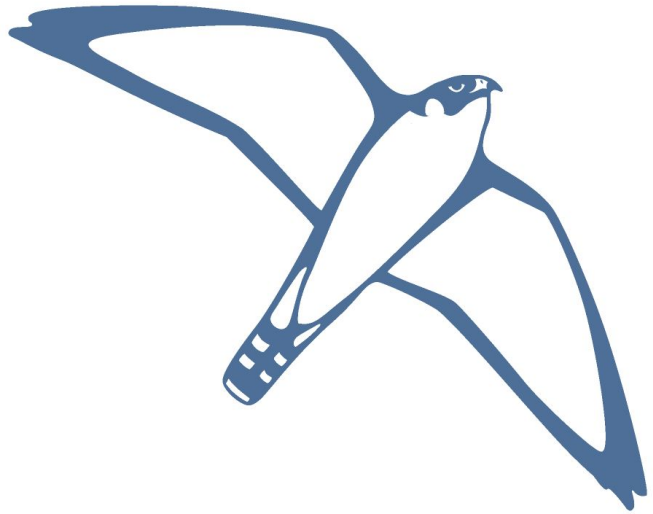
Bálint Soproni, Heidelberg University

Bálint Soproni, Aksel Alpay and Vincent Heuveline
Heidelberg University

May 6-8, 2026 | Heilbronn, Germany | iwocl.org

Introduction: GROMACS

- Molecular dynamics software package
- Very popular: 27000 citations
- Heavily uses GPUs
- AMD GPUs are targeted via AdaptiveCpp
- Large HPC systems use AMD GPUs
  - LUMI, Frontier, Hunter
- Does not use the most recent AdaptiveCpp compiler
- New compiler Improved performance on medium sized problems



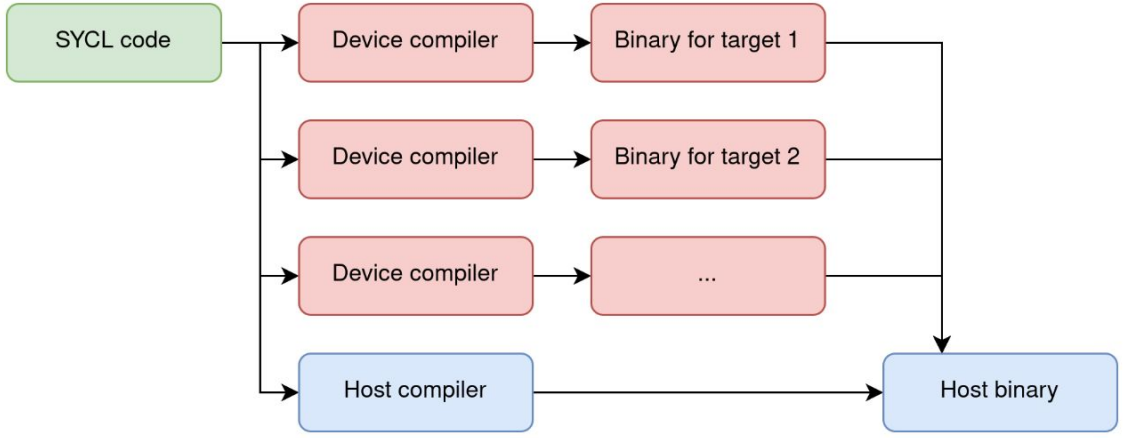
Logo of the GROMACS project [3]

Evaluate the performance of the AdaptiveCpp SSCP compiler on AMD devices

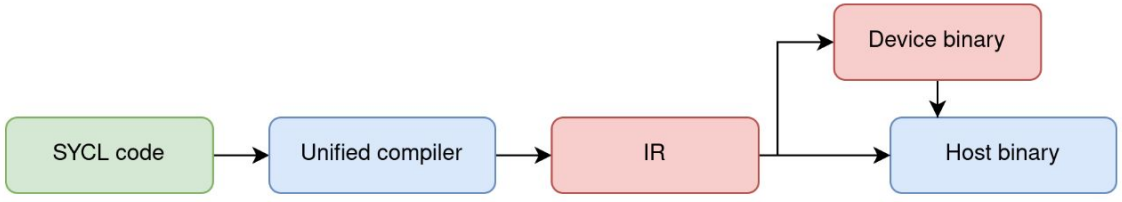
Introduction: AdaptiveCpp SYCL Compilation Flows

- SMCP
  - **S**ingle source
  - **M**ultiple **C**ompilation **P**asses
  - Ahead-of-time compilation
- SSCP
  - **S**ingle source
  - **S**ingle **C**ompiler **P**ass
  - Just-in-time compilation
- SSCP allows additional optimization at JIT time
- Usual code specialization via Macros is not available

SMCP Compiler



SSCP Compiler



Figures from Alpay et.al [1]

Implementation: Builtins

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

SMCP

- Multipass paradigm uses preprocessor macros
- Not used branch is eliminated

SSCP

- During compile time both branches are included
- At runtime the JIT compiler treats `is_amd` as a constant
- The optimizer removes the unused branch

```
#if defined(__SYCL_DEVICE_ONLY__) && defined(__AMDGCN__)
   reduceForceJAmdDpp(f, tidxi, aidx, a_f);
+ #elif GMX_ACPP_HAVE_GENERIC_TARGET
+   namespace jit = sycl::AdaptiveCpp_jit;
+   namespace rq = jit::reflection_query;
+   bool is_amd = rq::reflect<reflection_query::compiler_backend>()
+                 == compiler_backend::amdgpu;
+   jit::compile_if(is_amd, [&](){
+     reduceForceJAmdDpp(f, tidxi, aidx, a_f);
+   });
+   jit::compile_if(!is_amd, [&](){
+      reduceForceJShuffle(f, itemIdx, tidxi, aidx, a_f);
+   });
#else
   reduceForceJShuffle(f, itemIdx, tidxi, aidx, a_f);
#endif
```

MR open at: https://gitlab.com/gromacs/gromacs/-/merge_requests/5671

Implementation: Refactoring the Non-Bonded Kernel

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

- Preprocessor macros controlled
  - Array of Structs / Structs of Arrays
  - Special datatype for MI210
  - unroll factor
- For SSCP requires control flow based specialization
- Refactored the code
- Possible ODR violation on the host-device boundary

```
#if defined(__AMDGCN__) && defined(__gfx90a__)
  using FCiFloat3 = AmdPackedFloat3;
#else
  using FCiFloat3 = Float3;
#endif
....
#if defined(__SYCL_DEVICE_ONLY__) && defined(__AMDGCN__)
  FCiFloat3 fCiBuf_[c_superClusterSize]; // i force buffer
#else
  float fCiBufX_[c_superClusterSize] = { 0.0F }; // i force buffer
  float fCiBufY_[c_superClusterSize] = { 0.0F }; // i force buffer
  float fCiBufZ_[c_superClusterSize] = { 0.0F }; // i force buffer
#endif
```

```
bool amd    = jit::reflect<rq::compiler_backend>() == amdgpu;
bool cuda   = jit::reflect<rq::compiler_backend>() == ptx;
bool gfx90a = jit::reflect<rq::target_arch>()      == 0x90a;
jit::compile_if(amd && gfx90a, [&](){
  auto kernel_config = nbnxm_arch_config<...>();
  nbnxmKernel_impl<...>(kernel_config, ...);
});
jit::compile_if(amd && !gfx90a, [&](){
  auto my_config=nbnxm_arch_config<...>();
  nbnxmKernel_impl<...>(kernel_config, ...);
});
```

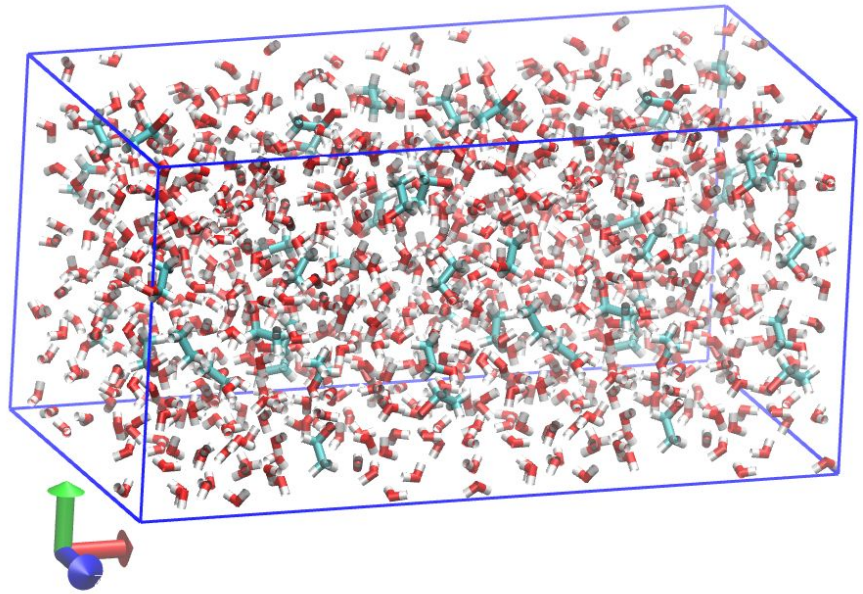
MR open at: https://gitlab.com/gromacs/gromacs/-/merge_requests/5670

h nbnxm_sycl_kernel_body.h +651 -542

Performance Evaluation

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

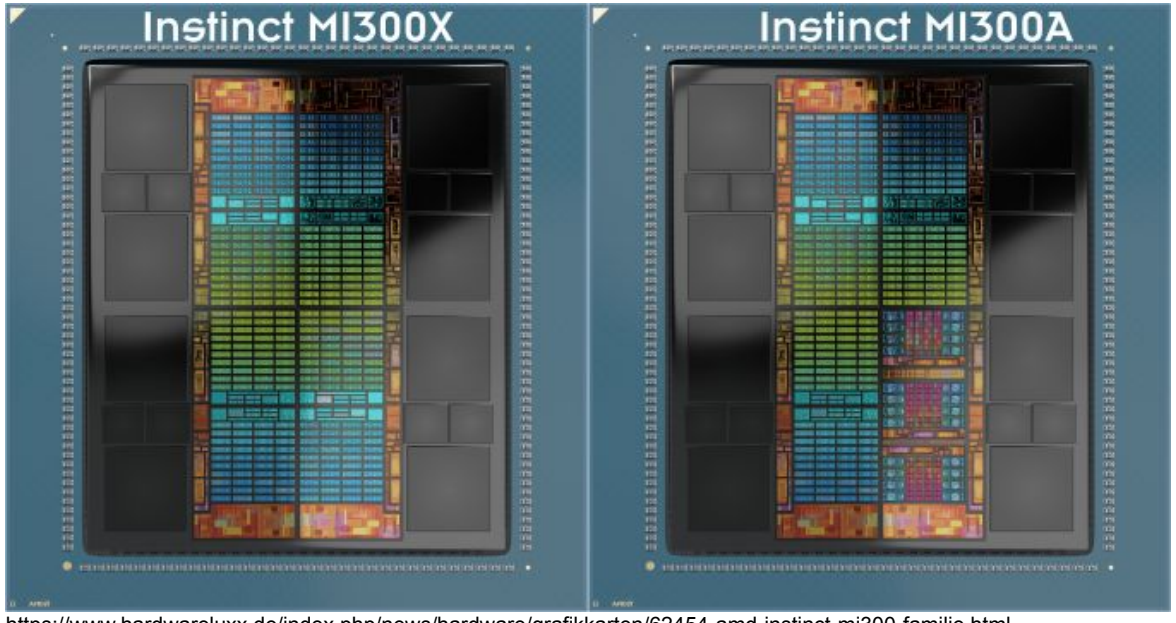
- Boxes of differently sized ethanol solutions
  - from 6,000 to 3 million atoms
  - covers the usual simulation sizes
- Simulation methods
  - Reaction field (rf) } single queue
  - Potential shift (psh)
  - Force switching (fsw)
  - Potential switching (psw) } concurrent queues
- Methodology end-to-end performance
  - at least 30 seconds for each simulation run
  - 5/3 runs for each box (small/large)
  - Comparing peak performances
- Software versions
  - Based on GROMACS 2025.4
  - AdaptiveCpp 2025.10
  - Available as Docker image[2]



3000 atom Grappa box

Performance Evaluation: Test Systems

- MI300X & MI300A
  - CDNA III architecture
  - At the time highest performance chips
  - Deployed at Hunter, available in AMD devcloud
  - 304 and 228 CUs
- MI210
  - CDNA II architecture
  - MI250s are Deployed at LUMI-G
  - 104 CUs
  - StreamHPC development server

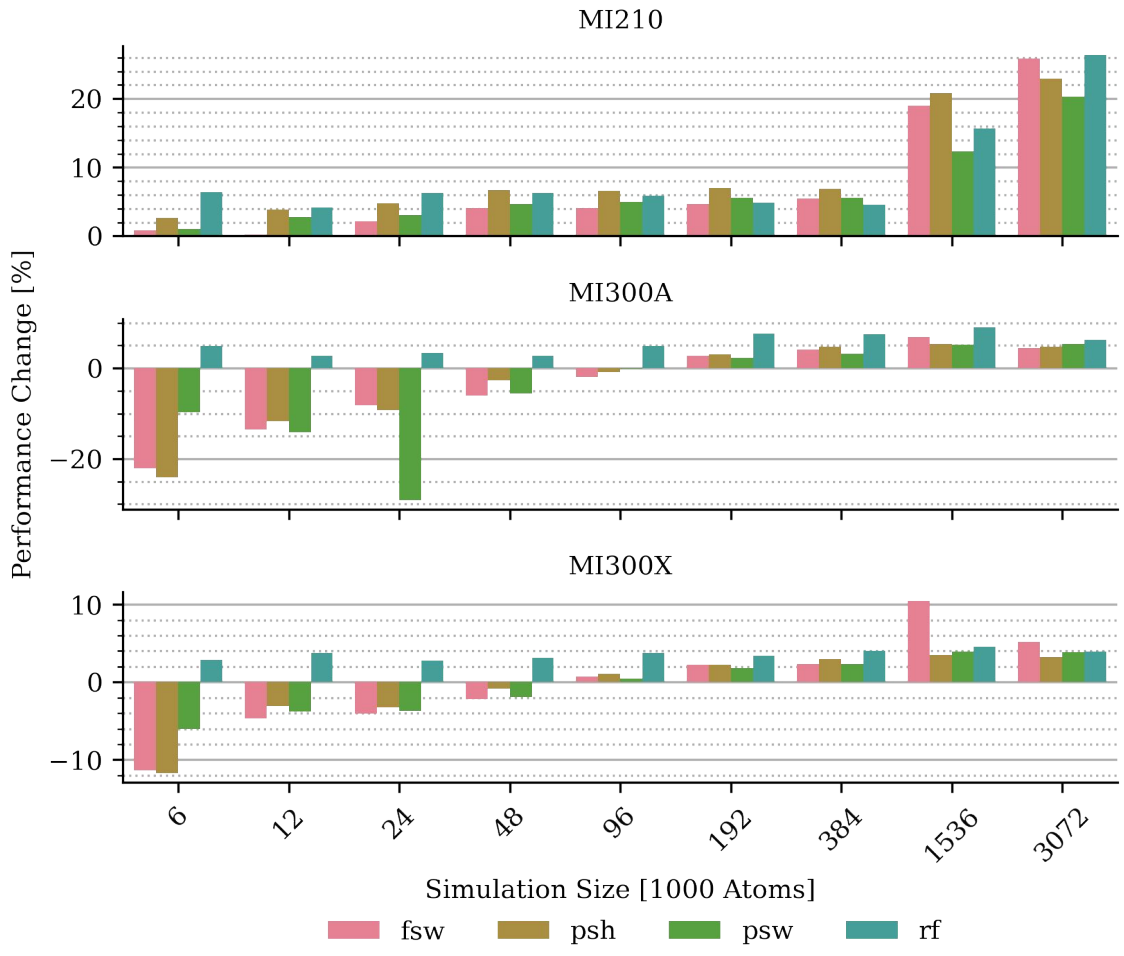


https://www.hardwareluxx.de/index.php/news/hardware/grafikkarten/62454-amd-instinct-mi300-familie.html

End to End Performance

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

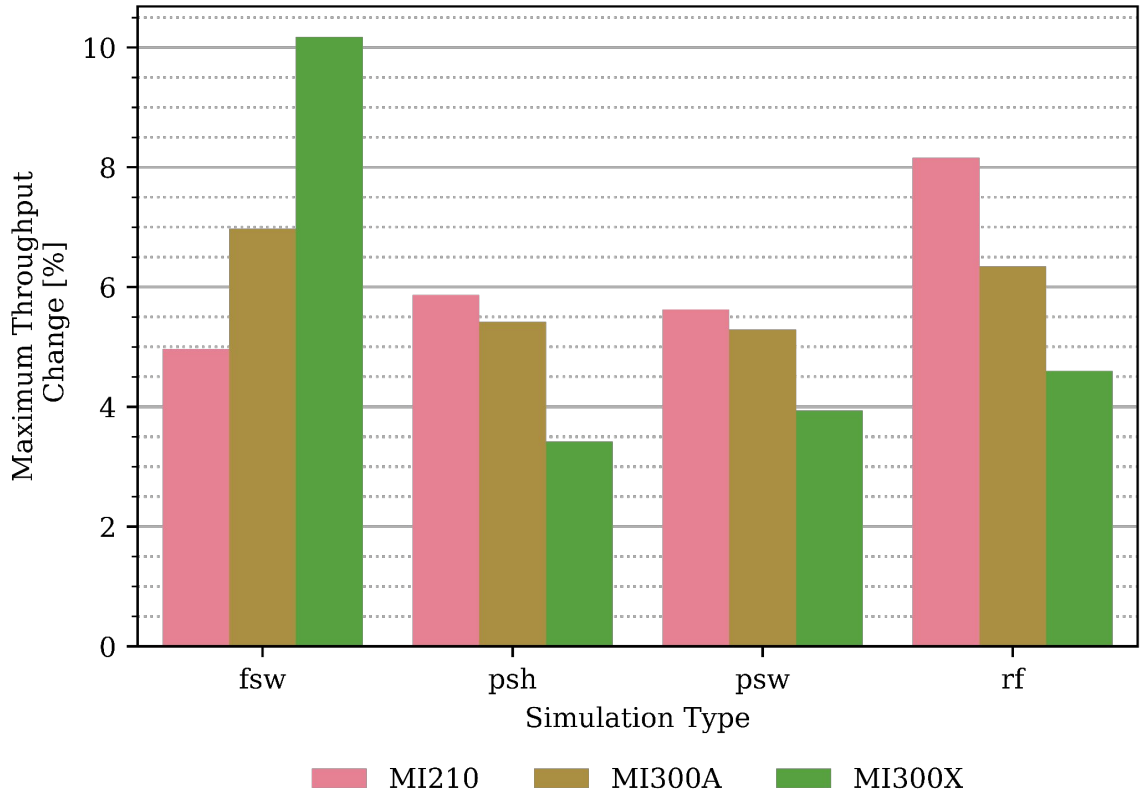
- MI210
  - Better performance for all configurations
  - Up to 25% improvement for 3M atom case
- MI300A & MI300X
  - Small multi queue problems large regression
  - up to 30%, 12% regression
  - Larger problems show 5%-10% improvement
  - Single queue problems (rf) show constant improvement 2-8%
  - Possible runtime overhead for multi-queue scheduling



Throughput Change

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

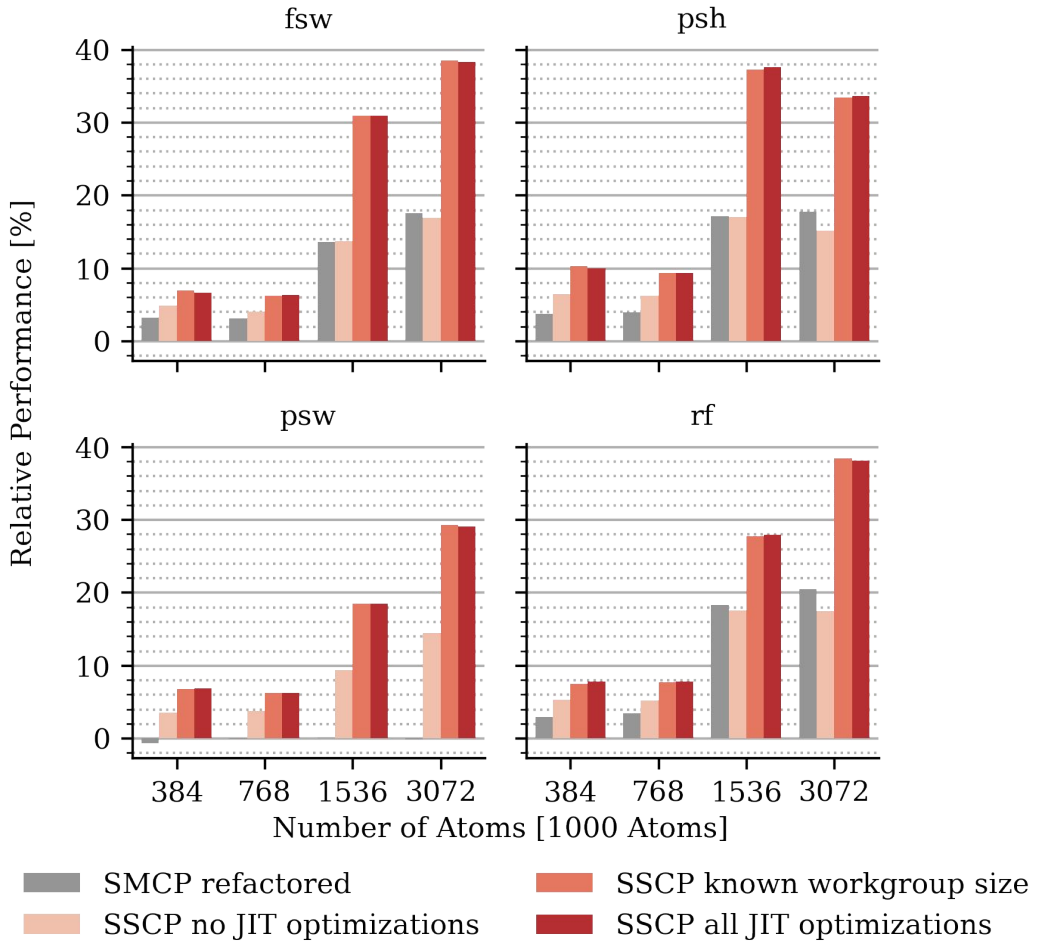
- Change in peak throughput measured across all simulations
- Throughput as the number of atoms processed per second
- Performance increases not limited for niche simulation setups
- Up to 10% improvement for MI300X fsw simulations



Performance of the Non-bonded Kernel

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

- Refactoring Impact (SMCP)
  - Performance improvement: Up to 15%
  - Improvements for fsw, psh, and rf methods
  - No improvement for psw
- SSCP JIT Compiler Impact
  - Performance improvement: Up to 35% for largest simulations
  - IT compiler considers work-group sizes during code generation
  - Confirms previous findings on AMD hardware



Future Work

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

- Further investigation into the underlying mechanisms profiling assembly analysis
  - Investigation of psw simulation anomalies
  - Concurrent queue handling
- Evaluation of portability for other backends (Intel, Nvidia)
- Finis upstreaming the current code changes

Conclusion

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

- AdaptiveCpp SSCP compiler enabled in GROMACS with limited changes
- More involved refactoring also improved the performance of the SMCP version
- JIT time optimizations are beneficial
- Generic JIT compiler is viable for complex production codebases

Acknowledgments

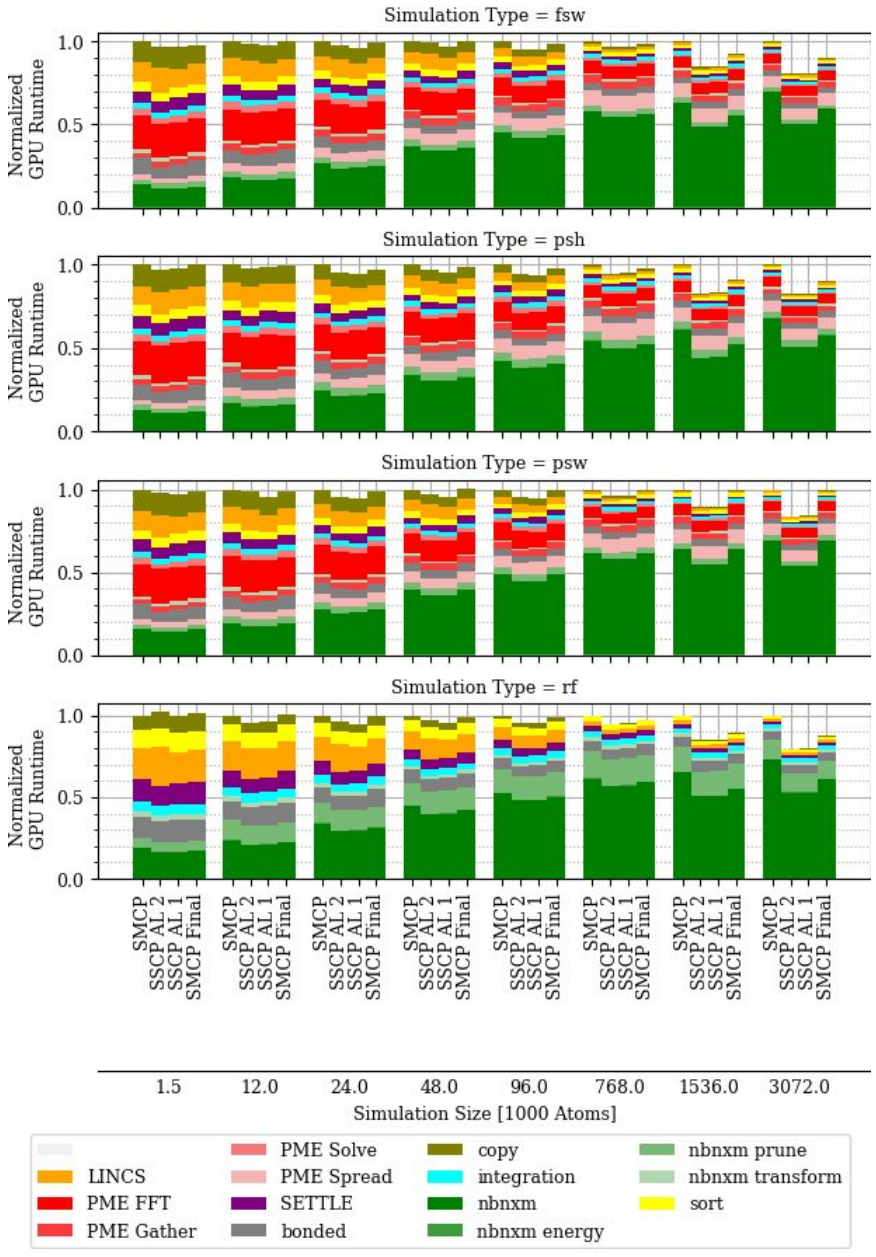
UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Andrey Alekseenko & Szilárd Páll (KTH Royal Institute of Technology)

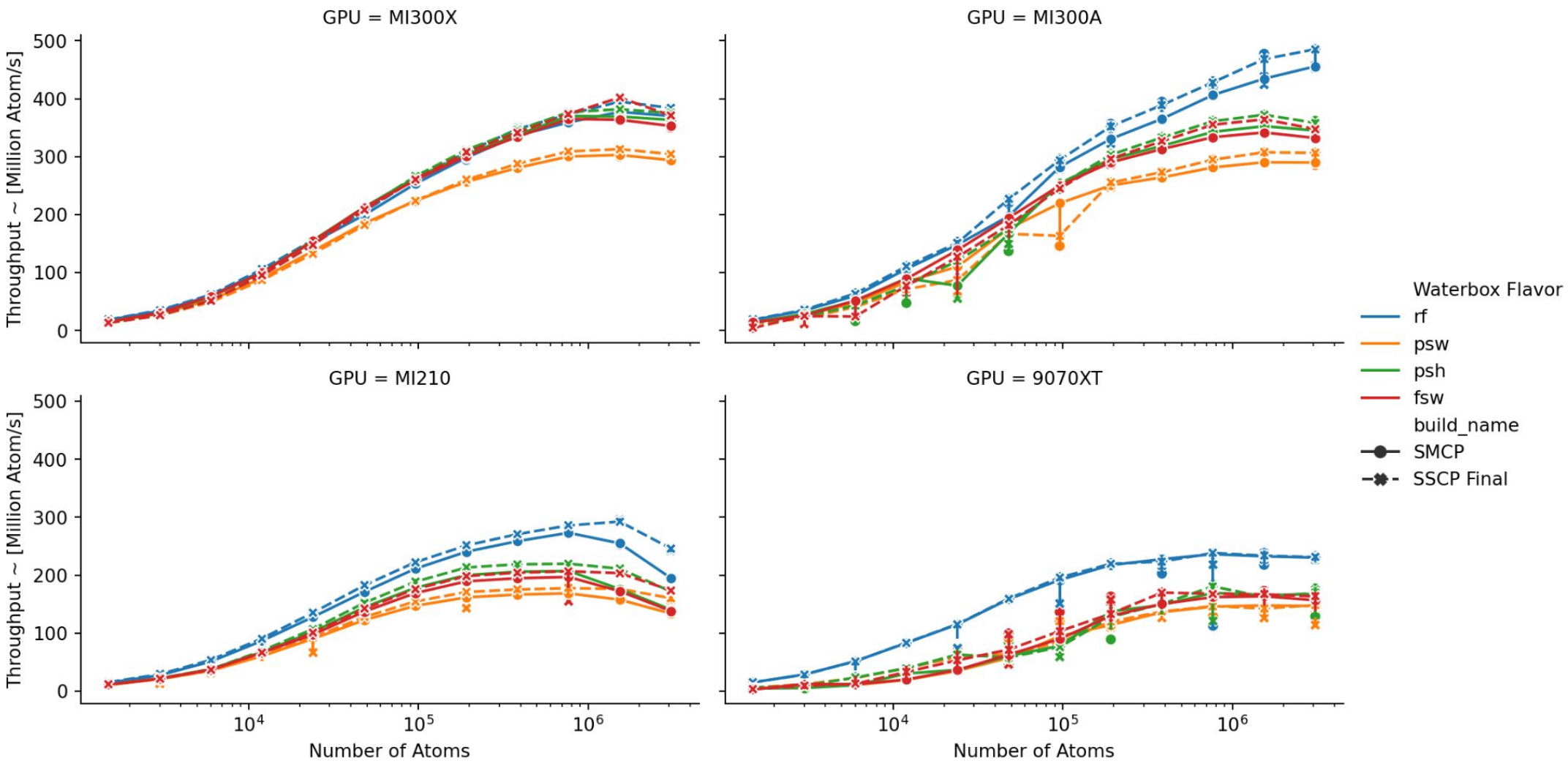
HLRS and StreamHPC for hardware access

GPU time on MI210

- Most of the performance gains can be attributed to the non-bonded kernel
- GPU kernel performance improvement explains the performance improvements but not the slow downs
-



Throughput Absolut Plot



Performance Improvement Geometric mean

UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

Table 1: Level 1 compared to SMCP adaptivity geometric mean

GPU	fsw	psh	psw	rf
9070XT	6.8%	3.7%	6.0%	0.4%
MI210	9.4%	10.8%	8.3%	9.5%
MI300A	3.8%	3.7%	3.4%	6.7%
MI300X	3.9%	2.5%	2.5%	3.8%

Table 2: Level 2 compared to level 1 adaptivity geometric mean

GPU	fsw	psh	psw	rf
9070XT	0.1%	3.6%	-0.1%	0.4%
MI210	0.3%	0.5%	0.2%	0.3%
MI300A	0.2%	0.1%	0.6%	0.1%
MI300X	0.4%	0.1%	0.5%	-0.1%

References

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

[1]: Aksel Alpay and Vincent Heuveline. “One Pass to Bind Them: The First Single-Pass SYCL Compiler with Unified Code Representation Across Backends”. In: Proceed- ings of the 2023 International Workshop on OpenCL. IWOCL ’23. Cambridge, United Kingdom: Association for Computing Machinery, 2023. isbn: 9798400707452. doi: 10.1145/3585341.3585351.

[2]: https://hub.docker.com/r/sbalint98/acpp-gmx-paper

[3]: https://manual.gromacs.org/documentation/current/index.html