A shared compilation stack for distributed-memory parallelism in stencil DSLs

George Bisbas *[1] Anton Lydike *[2] Emilien Bauer *[2] Nick Brown *[3]
Mathieu Fehr [2] Lawrence Mitchell Gabriel Rodriguez-Canal [2] Maurice Jamieson [3]
Paul H. J. Kelly [1] Michel Steuwer [4] Tobias Grosser [5]

[1]Imperial College London, UK [2]University of Edinburgh, UK [3]EPCC, University of Edinburgh, UK
[4]Technische Universität Berlin, Germany [5]University of Cambridge, UK
*authors equally contributed

The problem: Monolithic Domain-specific languages

Tailored to their domain, but actually lots of **common generic concepts!**
✅Performance ✅Productivity ✅Portability

Technical challenges:
⛔Independent/Siloed ⛔Lack of code reuse ⛔Separate ⛔Short lifespan

Societal challenges:
⛔Disjoint communities ⛔Lack of knowledge transfer

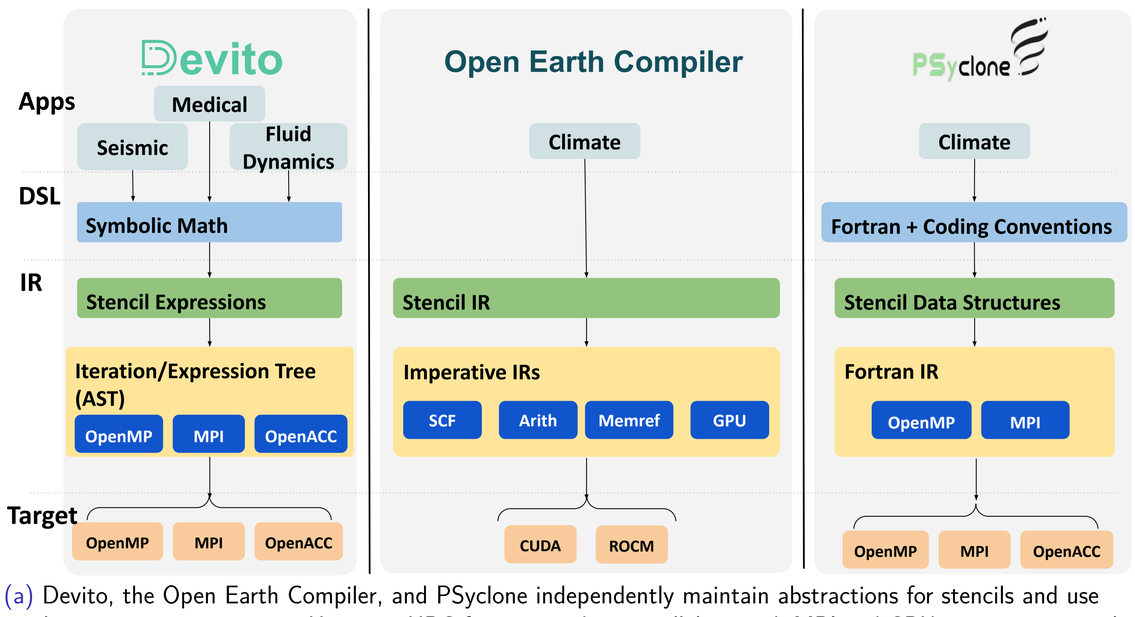
We propose: Using compiler technology!

Contributing a shared compilation stack for HPC in stencil DSLs:
✅Performance ✅Productivity ✅Portability

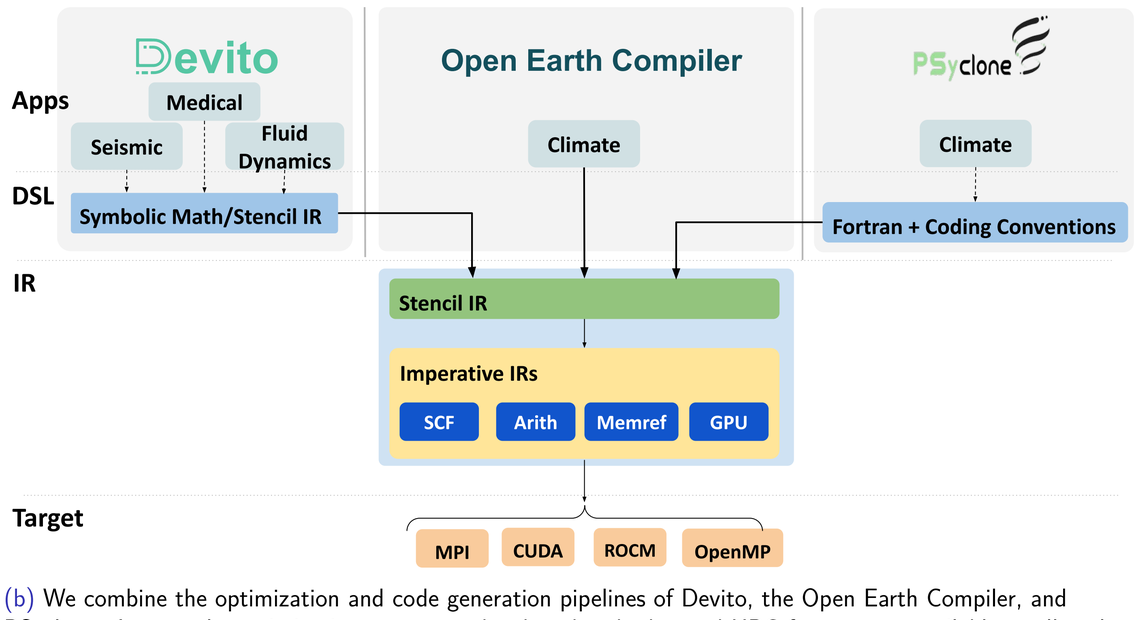
Technical benefits:
✅Composability ✅Code reuse ✅Interoperability ✅Longevity

Societal benefits:
✅Connected communities ✅Extensive knowledge transfer

Our work enables reuse of HPC and target-specific abstractions across DSL and compiler frameworks and offers synergies across DSL communities while maintaining the community-tailored interfaces of each DSL compiler.



(a) Devito, the Open Earth Compiler, and PSyclone independently maintain abstractions for stencils and use similar imperative constructs. However, HPC features such as parallelism with MPI and GPUs are not universal.



(b) We combine the optimization and code generation pipelines of Devito, the Open Earth Compiler, and PSyclone. As a result, optimization passes can be shared and advanced HPC features are available to all tools.

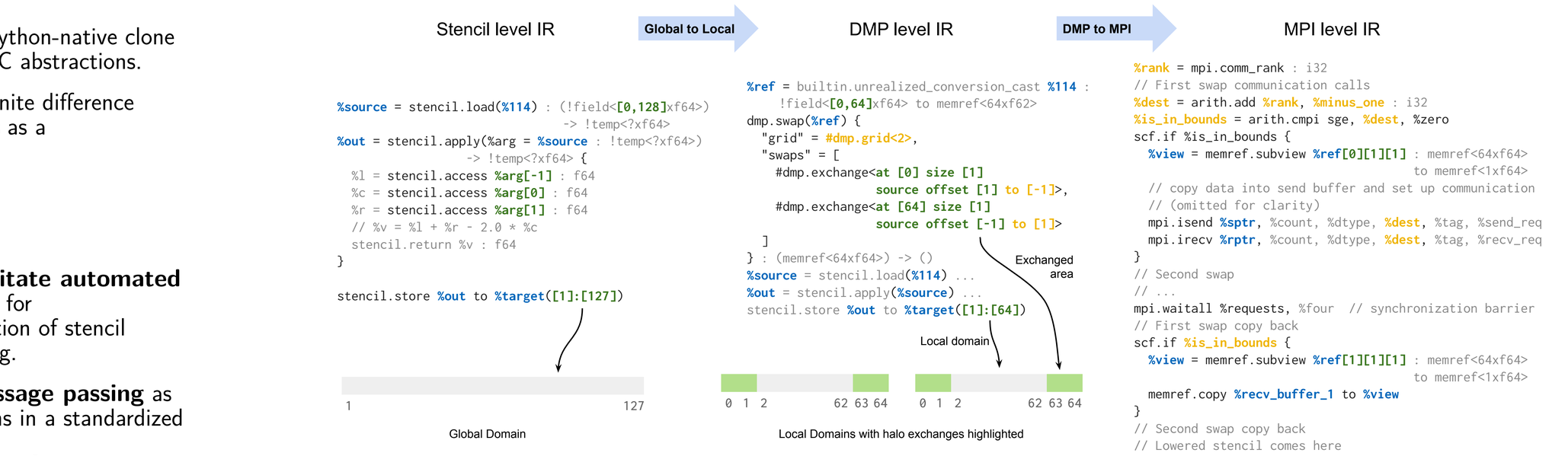
Proposed solution

- **Idea**: MLIR offers a unified IR but needs bridging with HPC concepts.
- **Fill the gap**: Introducing HPC-specific abstractions for interoperability with MLIR dialects.
- **How**: Utilizing xDSL, a python-native clone of MLIR, and building HPC abstractions.
- **Case**: Focus on explicit finite difference (FD) stencil computations as a representative case study.

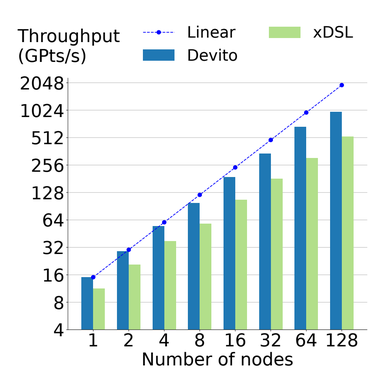
Contributions

- An **SSA dialect to facilitate automated domain decomposition** for distributed-memory execution of stencil kernels via message-passing.
- An **SSA dialect for message passing** as a set of modular operations in a standardized SSA-based IR.
  ✅**[Upstreamed to MLIR!]**
- A prototype implementation of **abstraction-sharing compilation stack for two HPC stencil-DSL compilers, PSyclone and Devito**, based on the concepts of *SSA* and *Region* and utilizing the MLIR and xDSL compiler frameworks.
- A **performance evaluation** demonstrating that our approach is competitive for a range of FD stencil computations, compared to the existing domain-specific compiler stacks, for **CPU shared- and distributed-memory parallelism, GPUs and FPGAs running at scale** on ARCHER2 and Cirrus.

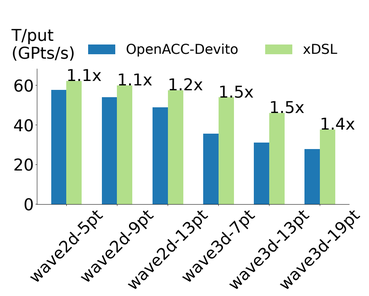
A stencil computation being transformed to a rank-local *stencil* + *dmp*, and then lowered to MPI. The data being operated on, shape and halo information, and communication-related information showcase how we enrich the IR with relevant information to perform rewrites at every level of abstraction.



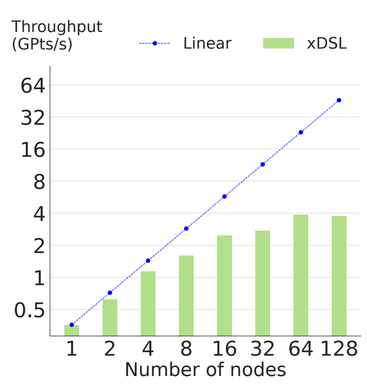
Performance evaluation of selected benchmarks, higher is better:



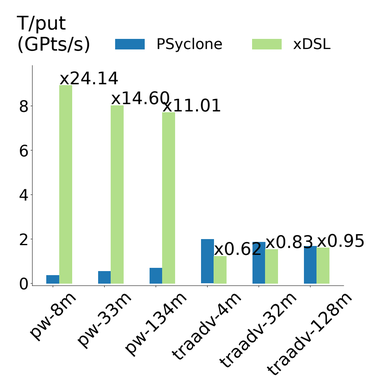
(a) Strong scaling of the acoustic wave kernel, space discretization order of 4, is competitive against Devito's highly optimized MPI modes.



(b) xDSL's lowerings through CUDA outperform Devito's tiled OpenACC kernels for more than 1.5x when it comes for 3D kernels on an NVIDIA V100.



(c) Multi-node strong scaling CPU throughput for xDSL-PSyclone for tracer advection $[512, 512, 128]$ on ARCHER2



(d) xDSL-PSyclone single node GPU (Cirrus) throughput, where tracer advection benchmark performance is limited by the MLIR *scf* parallel lowering transformations.

References

1. Fehr M., et al. "Sidekick compilation with xDSL", arXiv:2311.07422 (2024)
2. Luporini, F., et al. "Architecture and performance of Devito, a system for automated stencil computation." ACM TOMS 46.1 (2020): 1-28

EXCALIBUR
EPSRC Engineering and Physical Sciences Research Council
archer2