# NektarIR

## Towards code-generation of highly efficient finite element kernels for computational fluid dynamics using the MLIR compiler infrastructure

Edward Erasmie-Jones[(1)], Giacomo Castiglioni[(2)], David Moxey[(1)]
*(1) King's College London, United Kingdom. (2) CSCS, Switzerland*

## Aims

- Create a **high-level abstraction** of the elemental operators for the construction of PDE solvers using the spectral/hp element method
- Separate optimizations from the concerns of domain scientists through **target independent optimizations** using the MLIR compiler development infrastructure
- **Ensure hardware extensibility and facilitate JIT compilation** using LLVM

## Spectral/*hp* Element Method and Elemental Operators

- A **high-order** finite element method for solving partial differential equations used in HPC applications such as computational fluid dynamics
- Implementation requires frequent evaluation of **elemental operators** within elemental regions of a mesh of the solution domain [1], such as:

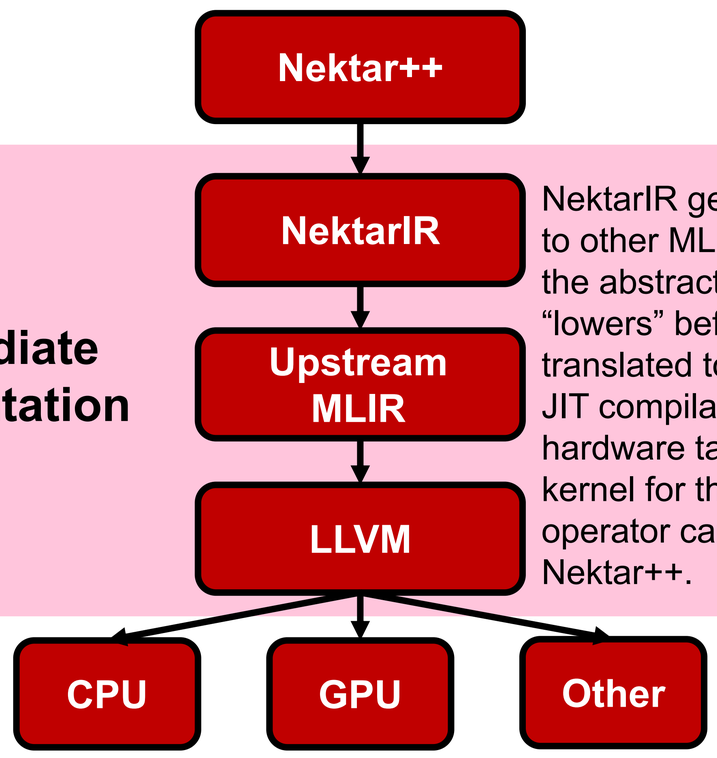
**Backward Transform**

$$u(\xi_{1i}, \xi_{2j}) = \sum_{p=0}^{P} \sum_{q=0}^{Q} \hat{u}_{pq} \phi_{pq}(\xi_{1i}, \xi_{2j}) \quad \forall i, j$$

**Inner product with respect to expansion basis**

$$I_{pq} = \sum_{i=0}^{N_1} \sum_{j=0}^{N_2} u(\xi_{1i}, \xi_{2j}) w_i w_j \phi_{pq}(\xi_{1i}, \xi_{j2}) \quad \forall p, q$$

## Intermediate Representation




NektarIR gets converted to other MLIR dialects as the abstraction level "lowers" before being translated to LLVM for JIT compilation to a hardware target. The JIT kernel for the elemental operator can be called in Nektar++.

### Nektar++

An open-source spectral/*hp* element framework for the development of high-fidelity PDE solvers [2].

### NektarIR

A high-level abstraction of the elemental operators in the spectral/hp element method developed as an MLIR dialect

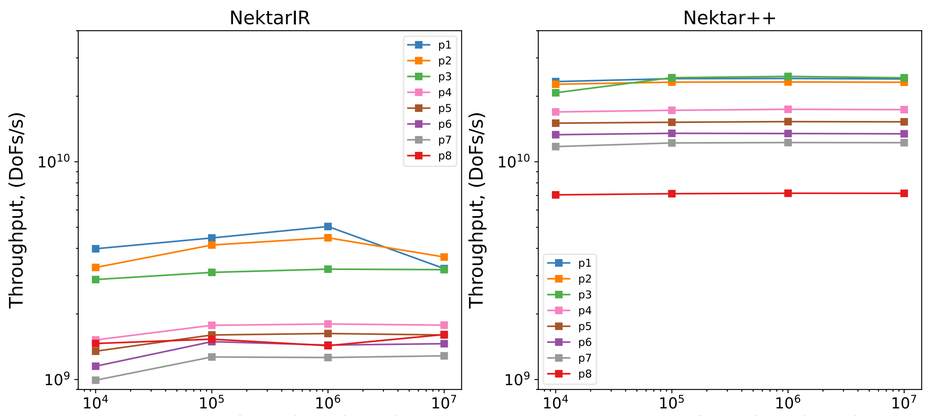
### MLIR

MLIR dialects describe common programming constructs at different, target independent, abstraction levels [3].

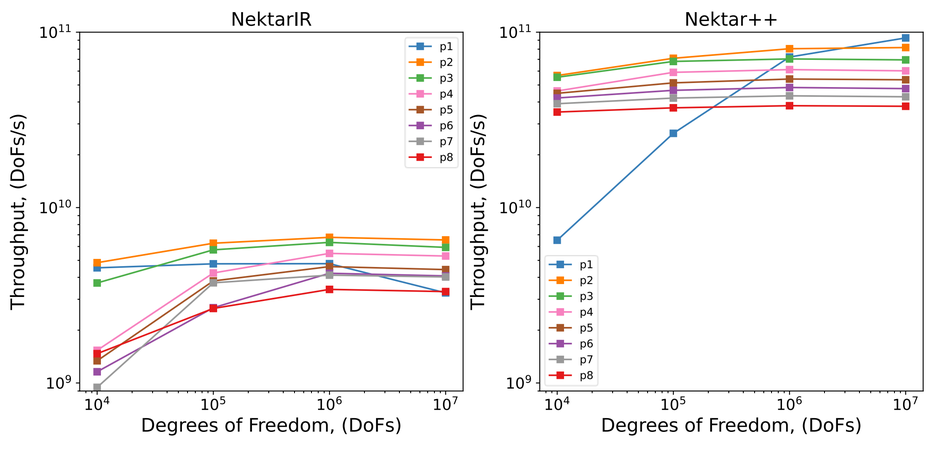
### LLVM

As a backend to our pipeline, LLVM facilitates the compilation of the elemental operators to different hardware targets.

## Initial Benchmarking: Backward Transform

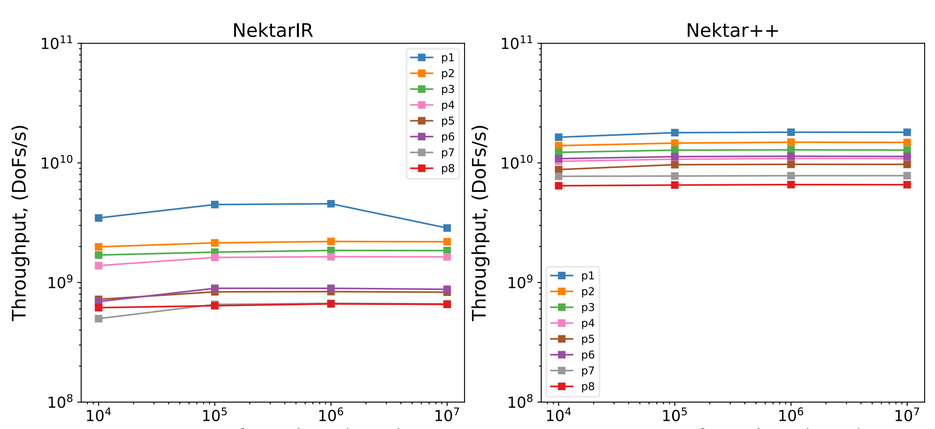


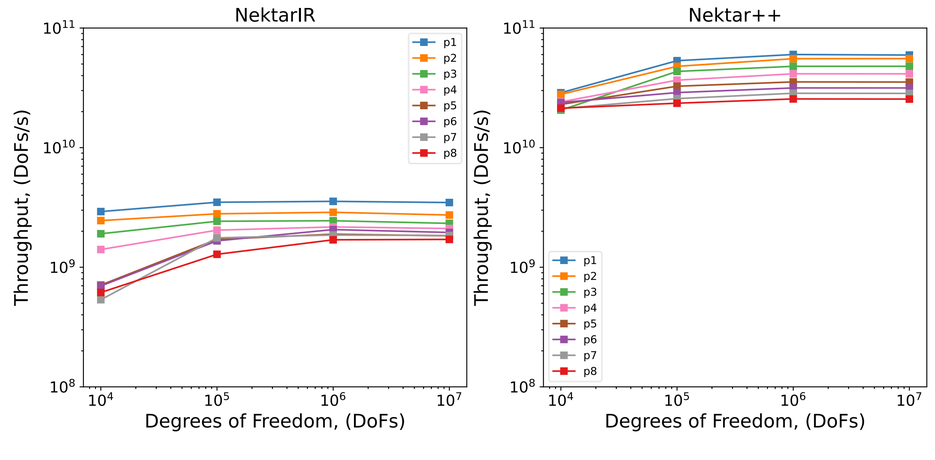

Comparison of scalar (top) and vectorized (bottom) throughput performance for two elemental operators on quadrilateral elements. Everything run on 64 cores across two AMD EPYC 9554 CPUs




NektarIR and Nektar++ use different vectorization strategies and results show Nektar++ has better performance. Differences in scalar performance requires further investigation and shows need for optimization.

## Initial Benchmarking: Inner Product WRT Basis




## Acknowledgements

EEJ would like to thank ACM SIGHPC for their generous support with the PASC2025 student travel grant.

DM acknowledges the Royal Academy of Engineering under their research chair scheme

## References

[1]: Karniadakis, G.E. and Sherwin, S.J. (2005) Spectral/hp Element Methods for CFD. Oxford University Press, Oxford. http://dx.doi.org/10.1093/acprof:oso/9780198528692.001.0001
[2]: C.D. Cantwell, D. Moxey, A. Comerford, A. Bolis, G. Rocco, G. Mengaldo, D. De Grazia, S. Yakovlev, J.-E. Lombard, D. Ekelschot, B. Jordi, H. Xu, Y. Mohamied, C. Eskilsson, B. Nelson, P. Vos, C. Biotto, R.M. Kirby, S.J. Sherwin, Nektar++: An open-source spectral/hp element framework, Computer Physics Communications, Volume 192, 2015, Pages 205-219, ISSN 0010 4655, https://doi.org/10.1016/j.cpc.2015.02.008.
[3]: https://mlir.llvm.org/