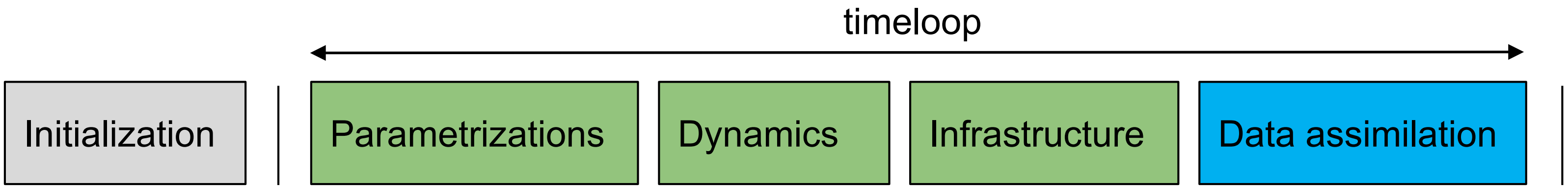
Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Eidgenössisches Departement des Innern EDI
Bundesamt für Meteorologie und Klimatologie MeteoSchweiz

PP IMPACT and HPC overview

X. Lapillonne[1], C. Osuna[1], D. Hupp[1], D. Alexeev[5], V. Cherkas[1], R. Dietlicher[1], E. Germann[1], F. Gessler[1], M. Jacob[4], A. Jocksch[3], J. Jucker[2],C. Müller[1], M. Röthlin[1], W. Sawyer[3], U. Schättler[4], André Walser[1]
[1]MeteoSwiss, [2]C2SM, [3]CSCS, [4]DWD, [5]Nvidia
06.09.2022, COSMO-GM WG6 – IMPACT parallel session

OpenACC port overview



- Most components for NWP Regional and global application ported, optimization work ongoing
- Support for both double and mixed precision
- Some components, e.g. ecRad,I need to be merged, some need to be ported
- Regular testing on builbot infrastructure

Status of the OpenACC port to GPU

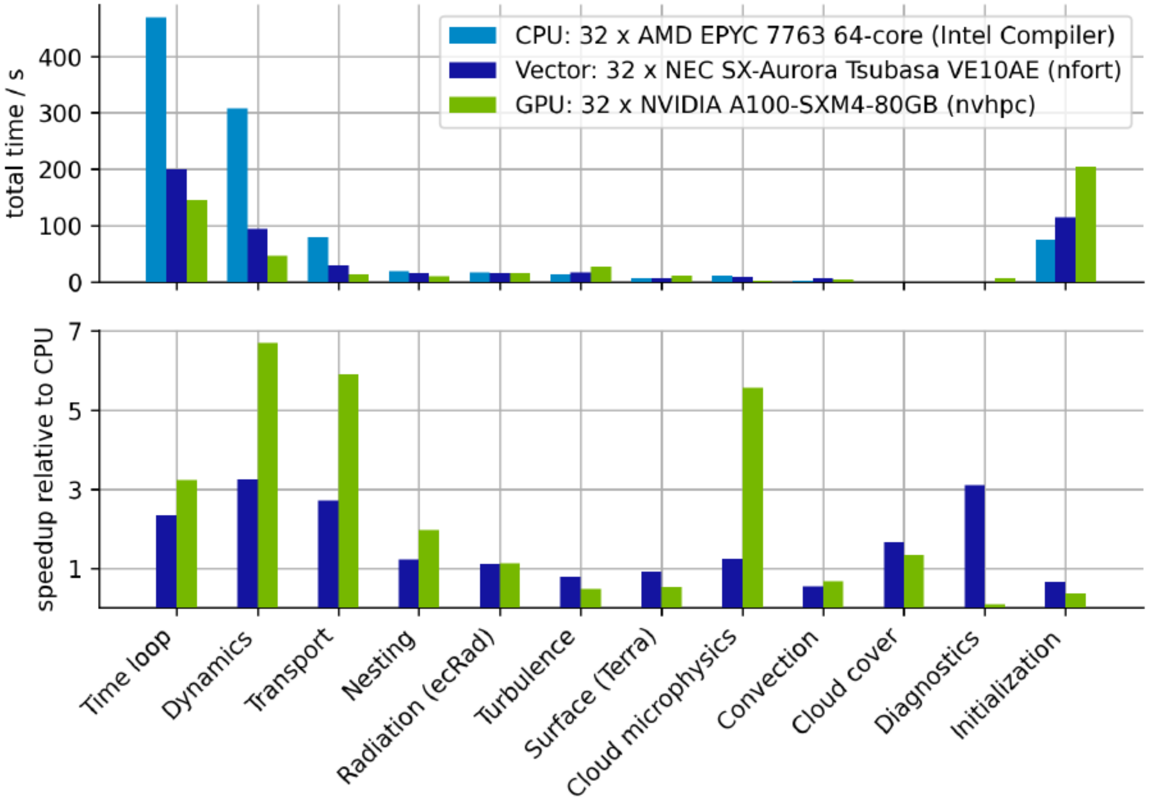
	ported	merged
nh_solve		
nh_hdiff		
transport		
2 way nesting		
convection		
Microphysics (graupel)		
radiation		
radheat		
Surface (terra)		
cover		
turbulence		
Sea-ice		
sso		
Non-or. Wave drag		
2 mom. microphysics		

	ported	merged
NWP diagnostic		
DA: LHN		
DA: conv. operatior		
DA: IAU (Incr. Anal. Update)		
SPPT		

	Ready
	In progress
	On CPU + data copy and interface ported
	Not started

Detailed missing features list: https://gitlab.dkrz.de/icon/wiki/-/wikis/GPU-development/todo-list

Current performance



CPU vs GPU vs NEC
Experiment:
- ~DWD deterministic forecast Global R2B**8**
- 5 242 880 cells (10 km) + Nested grid over Europe 845 340 cells (5 km)
- Output disabled

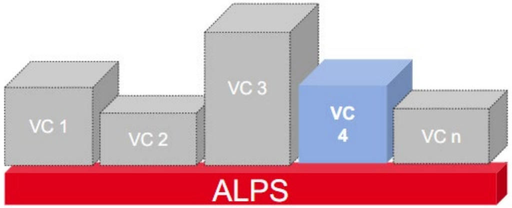
Performance and energy
- GPU 3x faster than GPU
- 32 GPUs: 663 s/day 268 Wh/day
- 32 VEs: 914 s/day 292 Wh/day

Source: Marek Jacob, DWD, using ICON branch with ecRad on GPU

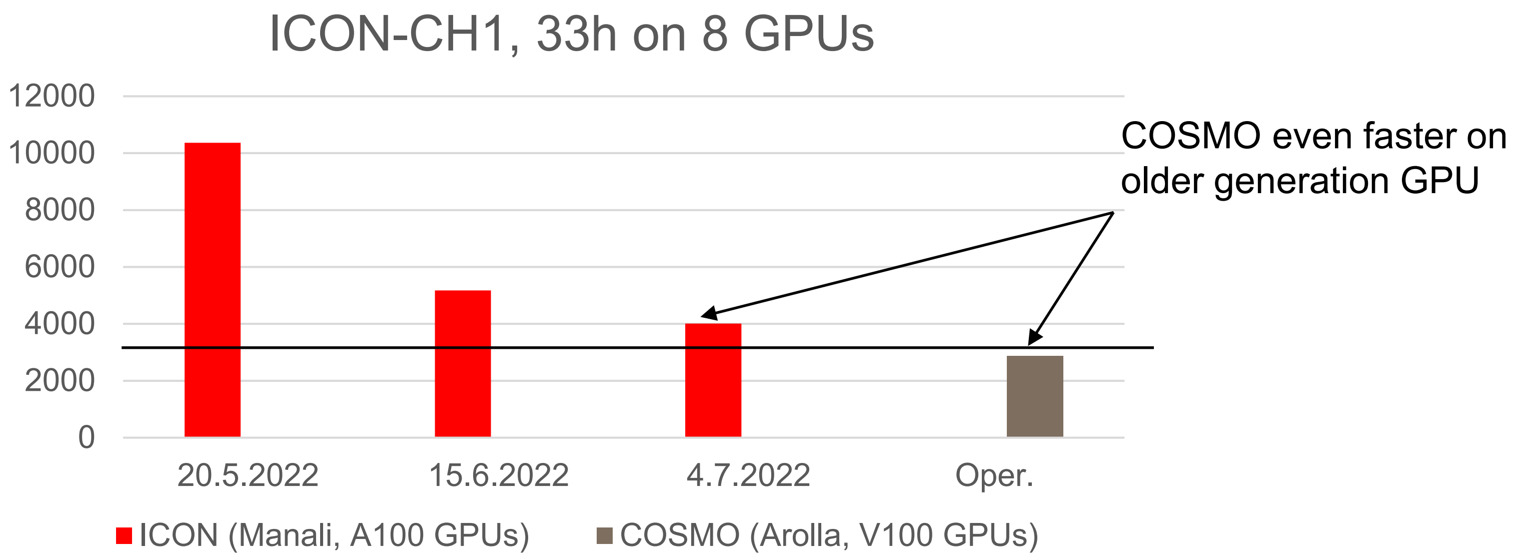
ICON-22 Performance

New MeteoSwiss system HPC Computing Services on Alps Plattform

- GPU nodes:
  - 4 x NVIDIA A100
  - 1 x AMD Epyc 64-cores CPU
- CPU nodes:
  - 2 x AMD Epyc 64-cores CPUs
- 2 Virtual Clusters (VC)
  - Production: 42 GPU / 15 CPU Nodes
  - R&D: 30-50 GPU / ~15 CPU Nodes (elastic)



ICON-CH1 on Alps (MeteoSwiss)



- Issues with the new system + slower GPU performance as compared to COSMO – required time to solution < 3000 s
- First optimization brought some improvement.
- ICON ca 2-3x slower than COSMO for same configuration and hardware

Domain Specific Language (DSL) in weather and climate – really ?

- DSL : computer language restricted to a particular domain
- We need performance to reach time to solution
- Separation of concern between domain and computer scientist
- Single source code for multiple target architectures
- Possible to write a new backend when a new technology emerged
- Allow aggressive optimization without degrading readability of user code
- Allow optimization across components – data centric optimization

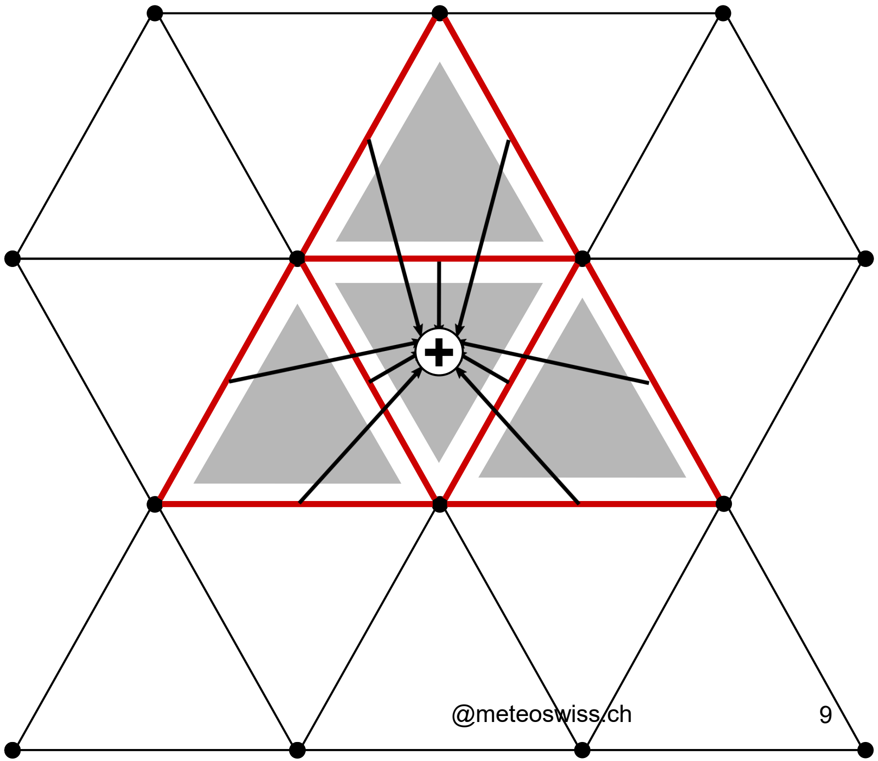
High level DSL for ICON

- Need to support unstructured grid, such as ICON grid
- New abstraction (e.g. neighbors operations)
- Focus on usability, productivity. Should be usable for domain scientist
- High level python dsl (gt4py)
- Development work in the EXCALIM project
  - High resolution use cases
  - Re-write code components using python DSL framework

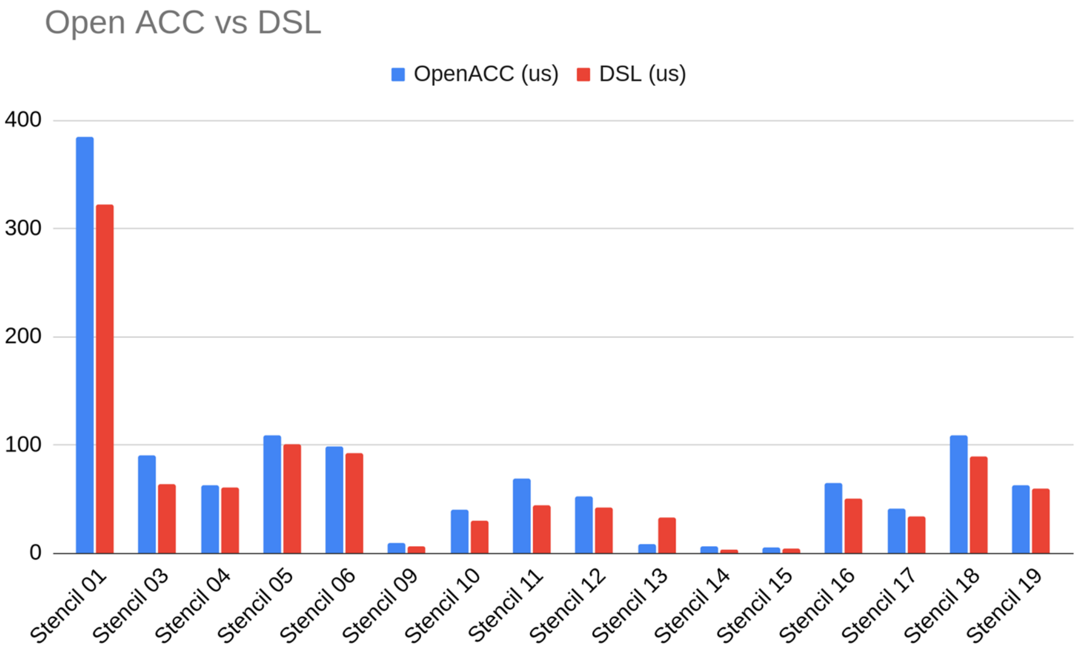
EXCLAIM

Python DSL notation example (gt4py) : Neighbor Chains

```
@field_operator
def intp(
    fe: Field[[EdgeDim], float],
    w: Field[[CellDim, C2E2C2EDim], float],
) -> Field[[CellDim], float]:
    f_c = neighbor_sum(w * f_e(C2E2C2E), axis=C2E2C2EDim)
    return f_c
```



Performance of ICON dycore (DSL) prototype



DSL : Dusk/ Dawn

- Stencil by stencil translation 10-20% faster DSL compared to OpenACC.
- Prototype DSL dycore about 40% (1.4x) faster then OpenACC - not fully optimized. Dry dycore only.

Current state gt4py re-write

- Dry dycore almost completely translated to gt4py (7 of 110 stencils still in progress)
  - Performance the same as dusk/dawn (tested for 20 stencils)
- Tracer advection partially translated (20 stencils), continuing work

- Dry dycore + tracer advection are 60%-70% of the runtime of a full run

- Next focus for dycore: optimization and robustness

- Also ongoing: microphysics using gt4py; focus here is good language support (example: if-else)

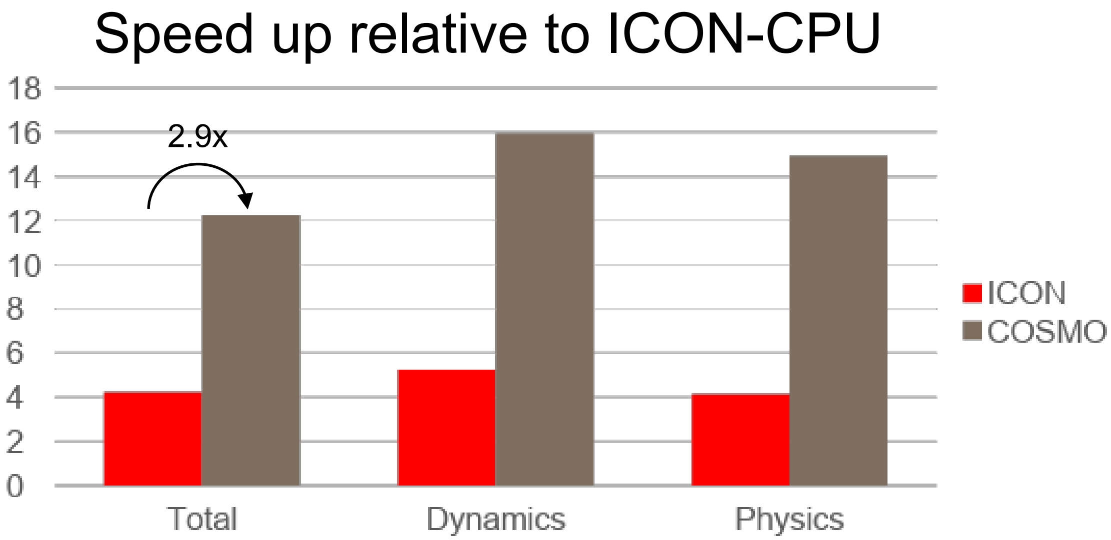
Conclusions

- First version of ICON model ported to GPU using OpenACC compiler directives
- Most components for global and regional NWP ported, and shall be soon available in icon-nwp master
  - basic version: Q4 2022
  - complete version: std NWP configurations tested with buildbot: Q4 2023
- Reasonable first performance, 3x faster than CPU, but still potential for optimization as still 2x time slower than COSMO
- Training of the ICON developers to work with OpenACC will be organized
- NWP application can likely benefit from DSL development in EXCLAIM, in particular for the dynamical core which shall be used once available

Additional slides

COSMO-ICON performance comparison on Daint

Socket to socket comparison: operational ICON-CH2 / COSMO-2E (2 km) 8 Nodes, 1h, P100 GPU vs Intel Xeon E5 12 cores (Piz Daint, CSCS, GPU node) - timeloop only



- ICON CPU vs GPU : 4.2x speedup
- ICON is 2.9x slower than COSMO on GPU for an equivalent setup.

14

Porting and optimization challenges

OpenACC optimizations

- GPU and CPU working asynchronously
  - Reduces launch overhead
- Bundling similar loop constructs into single GPU kernels
  - Improves cache reuse
  - Reduces launch overhead
- Compiler assisted / manual inlining of function calls
  - Required for complex (deep call-trees) GPU kernels
  - Enables optimizations above

Conceptual challenges

- Tiling for surface and turbulence
  - Implicitly introduces sub-blocking which leads to underutilized GPUs
- Physics initialization on CPU
  - Prohibitively slow because of unsuitable nproma and MPI settings for CPU
- Radiation sub-blocking
  - Radiation (ec-rad) has an additional dimension which can be parallelized Sub-blocking as a memory optimization
- Code management
  - Disruptive code changes are challenging
  - ecrad: juggling diverse Institutes