Hewlett Packard
Enterprise

LUMI - Delivering Real-World Application Performance at Scale through Collaboration

Samuel Antao[3], Aniello Esposito[2], Alessandro Fanfarillo[3], Alistair Hart[2], Nicholas Malaya[3], George Markomanolis[3], Diana Moise[2], Andrei Poenaru[2], Fredrik Robertsén[1], Peter Wauligmann[2]

[1] CSC, LUMI; [2] HPE; [3] AMD; Presenter

May 11, 2023

Overview

Introduction to the LUMI system

HPL: the road to the Top3

GROMACS: leveraging new hardware instructions

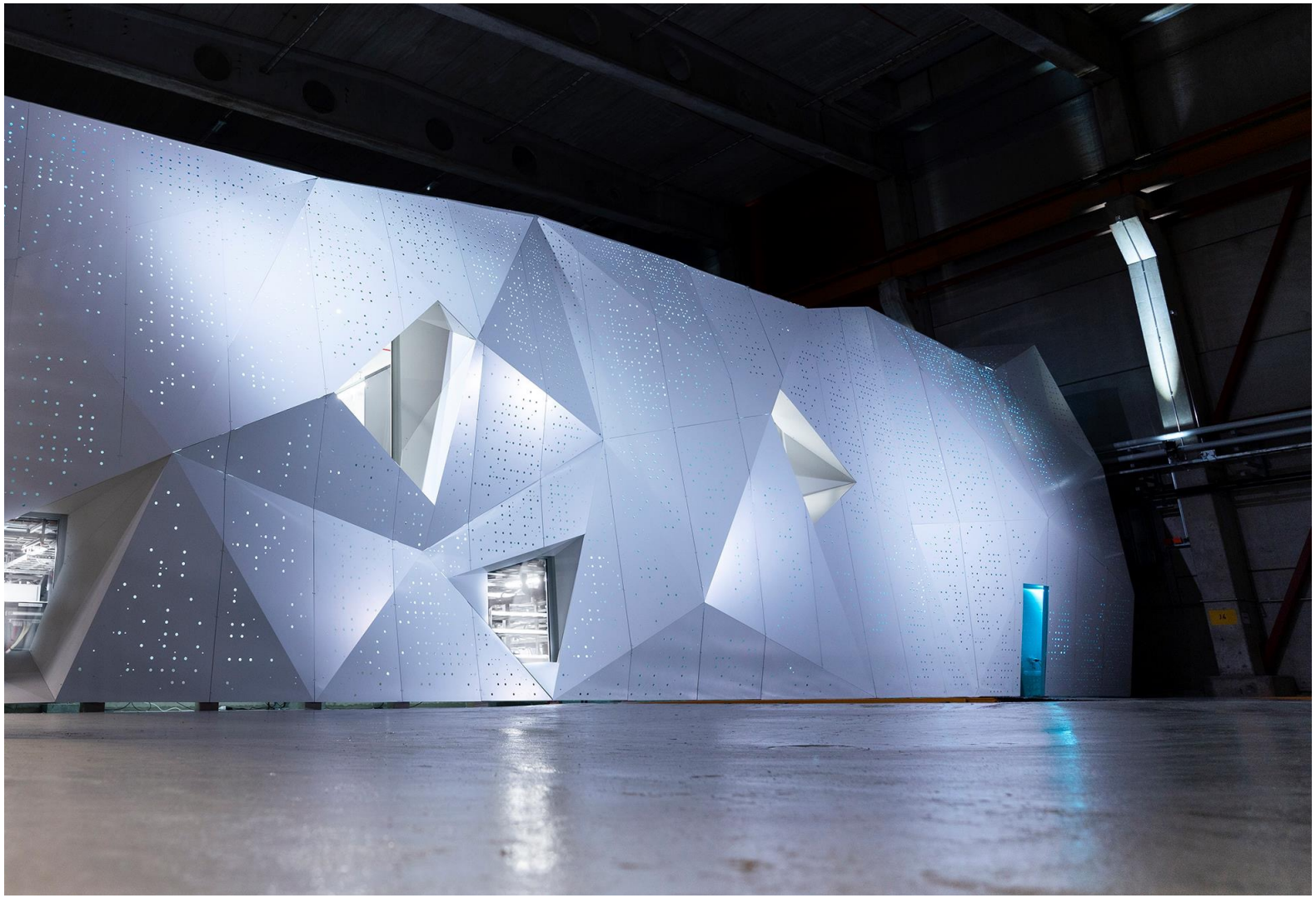
GRIDTOOLS: a lesson in memory alignment

ICON: driving improved compiler optimisations

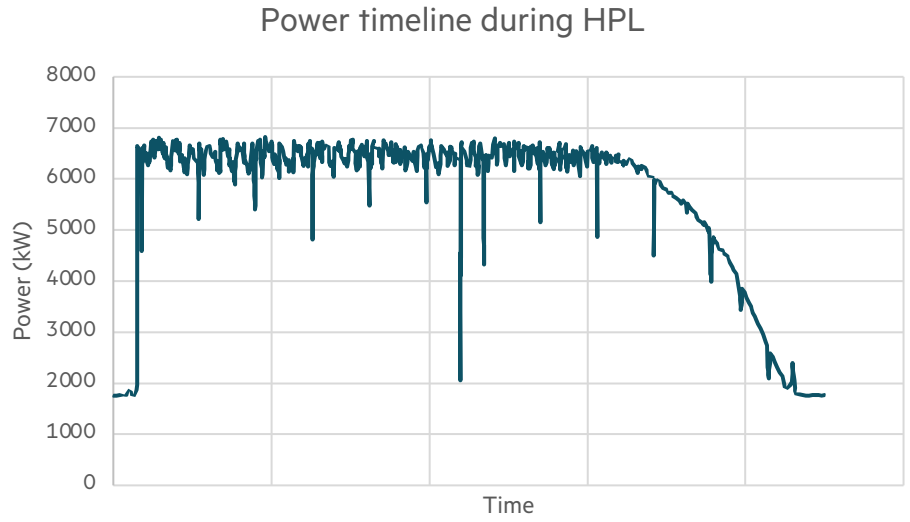
RCCL: tuning performance for ML and AI workloads

What is LUMI?

- LUMI is 1 of 3 EuroHPC pre-exascale machines
- Machine located in Kajaani, Finland
- Hosted by Lumi consortium
- Coordinated by CSC – IT center for science
- Consortium made up of 10 countries, and the EuroHPC joint undertaking: Finland, Belgium, the Czech Republic, Denmark, Estonia, Iceland, Norway, Poland, Sweden, and Switzerland
- ~200 M€ budget (TCO)
- CPU partition installed Summer 2021,
- GPU nodes began arriving in Summer 2022
- Renewable energy, 100% hydro power and heat reuse through district central heating



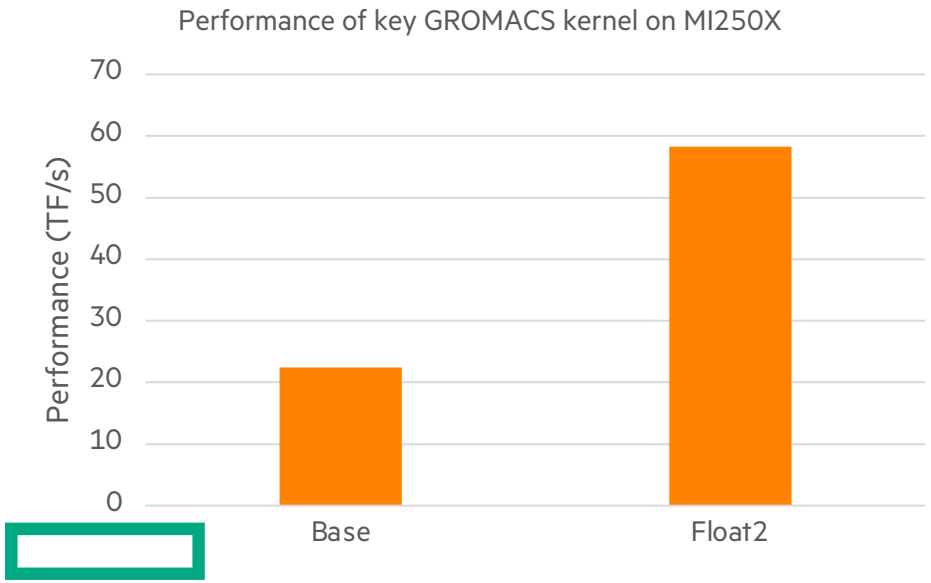
HPL



- HPL is the milestone on which systems are judged
- LUMI is Top3 in the world with more than 309 PF/s
  - as well as No. 1 in Europe
  - as well as No. 2 in the world for HPL-MxP and No. 3 for HPCG
  - Plus, only 1 large system is ahead in the Green500 ranking
- And there is more to come with the ongoing expansion to 375 PF/s
- HPL is not simply an artificial demonstration of DGEMM computational performance
  - Getting a successful HPL run means that every node and every part of the fabric has been thoroughly vetted
  - Means the system has become production-ready, so user applications perform well from Day 1
- Also, the software improvements for HPL feed down to real user applications
  - e.g., the DGEMM BLAS tuning
    – the AMD TENSILE library already benefits application codes like CP2K, PyFR, BDAS
  - Improvements in the low-level drivers and BIOS to improve both performance and stability
  - the Low Noise Mode tunings to reduce OS-derived noise on the compute nodes and increase performance at scale

GROMACS

- GROMACS AQP is limited by FP32 rate: single-precision, compute-bound, "non-bonded force" kernel
- AMD MI250X GPU introduces “packed FMA32” operations
  - Permits two component vector instructions to operate in FP32 in parallel (i.e., double throughput)
  - Introduces new instructions, using the FP64 datapath used to execute two FP32 operations
  - New operations are: pk_FMA, pk_ADD, pk_MUL, pk_MOV
- Requires only modest refactoring of code
  - Public example at: https://www.amd.com/en/technologies/infinity-hub/mini-hacc
  - The benefits can be even more than 2x



Original

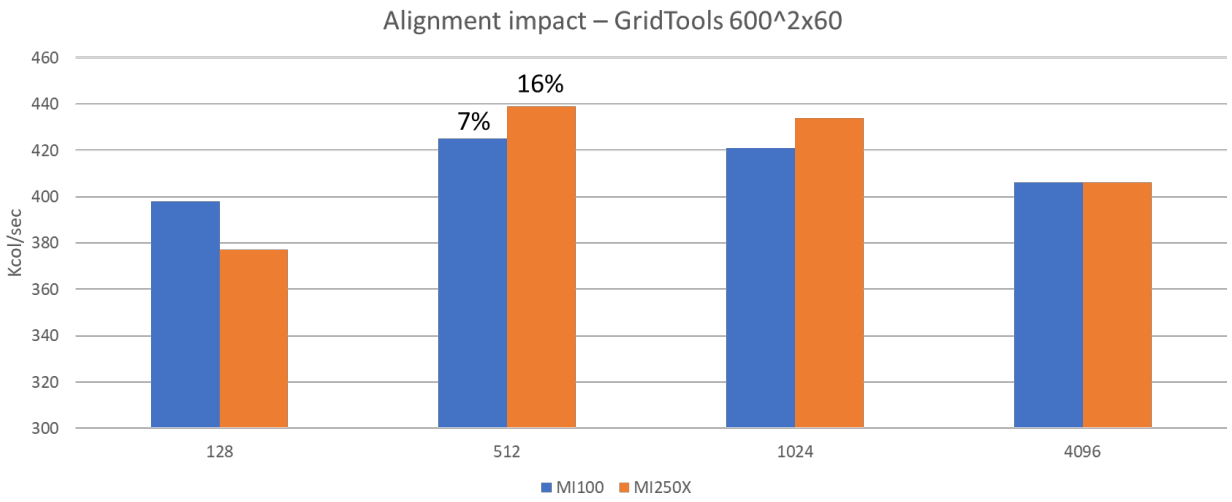
```
float vxi = 0.0f, vyi = 0.0f, vzi = 0.0f;
for (int j = threadIdx.x; j < count1; j += blockDim.x) {
  float dx = xx1[j] - xxi;
  float dy = yy1[j] - yyi;
  float dz = zz1[j] - zzi;
  float dist2 = dx*dx + dy*dy + dz*dz;
  if (dist2 < fsrrmax2) {
    float rtemp = (dist2 + rsm2)*(dist2 + rsm2)*(dist2 + rsm2);
    float f_over_r = massi*mass1[j]*(1.0f/sqrt(rtemp) -
        (ma0 + dist2*(ma1 + dist2*(ma2 + dist2*(ma3 +
        dist2*(ma4 + dist2*ma5))))));
    vxi += fcoeff*f_over_r*dx;
    vyi += fcoeff*f_over_r*dy;
    vzi += fcoeff*f_over_r*dz;
  }
}
```

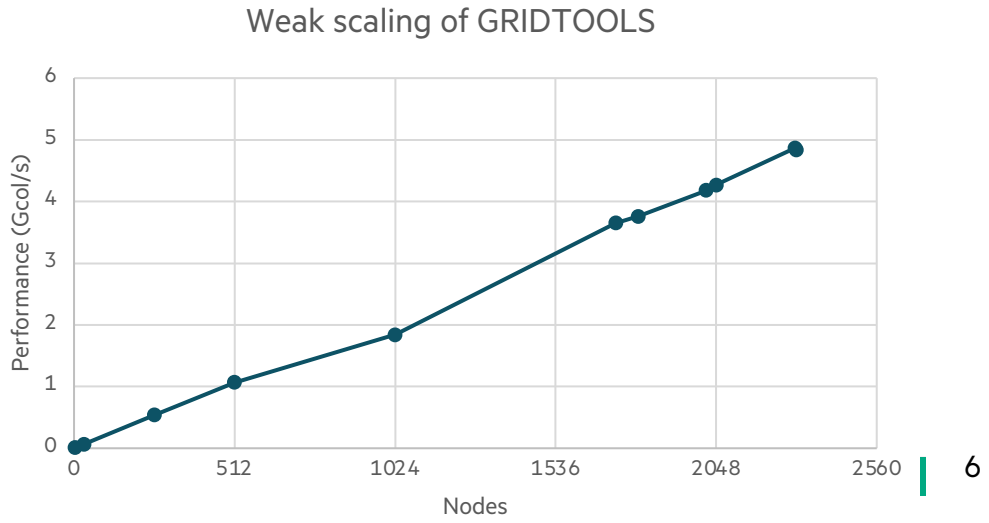
Modified to use Packed FMA32

```
float vxi = 0.0f, vyi = 0.0f, vzi = 0.0f;
  for (int j = threadIdx.x; j < count1; j += 2*blockDim.x) {
    float2 dx = {xx1[j] - xxi, xx1[j + blockDim.x] - xxi};
    float2 dy = {yy1[j] - yyi, yy1[j + blockDim.x] - yyi};
    float2 dz = {zz1[j] - zzi, zz1[j + blockDim.x] - zzi};
    float2 dist2 = dx*dx + dy*dy + dz*dz;
    bool check[2] = {dist2.x < fsrrmax2, dist2.y < fsrrmax2};
    if (check[0] || check[1]) {
      float2 rtemp = (dist2 + rsm2)*(dist2 + rsm2)*(dist2 + rsm2);
      float2 mass1_2 = {mass1[j],mass1[j + blockDim.x]};
      float2 sqrt_rtemp = {sqrtf(rtemp.x),sqrtf(rtemp.y)};
      float2 f_over_r = massi*mass1_2*(1.0f/sqrt_rtemp -
          (ma0 + dist2*(ma1 + dist2*(ma2 + dist2*(ma3 +
          dist2*(ma4 + dist2*ma5))))));
      float2 vxi_tmp = fcoeff*f_over_r*dx;
      float2 vyi_tmp = fcoeff*f_over_r*dy;
      float2 vzi_tmp = fcoeff*f_over_r*dz;
      vxi += check[0] ? vxi_tmp.x : 0.0f;
      vxi += check[1] ? vxi_tmp.y : 0.0f;
      vyi += check[0] ? vyi_tmp.x : 0.0f;
      vyi += check[1] ? vyi_tmp.y : 0.0f;
      vzi += check[0] ? vzi_tmp.x : 0.0f;
      vzi += check[1] ? vzi_tmp.y : 0.0f;
    }
  }
```

GRIDTOOLS

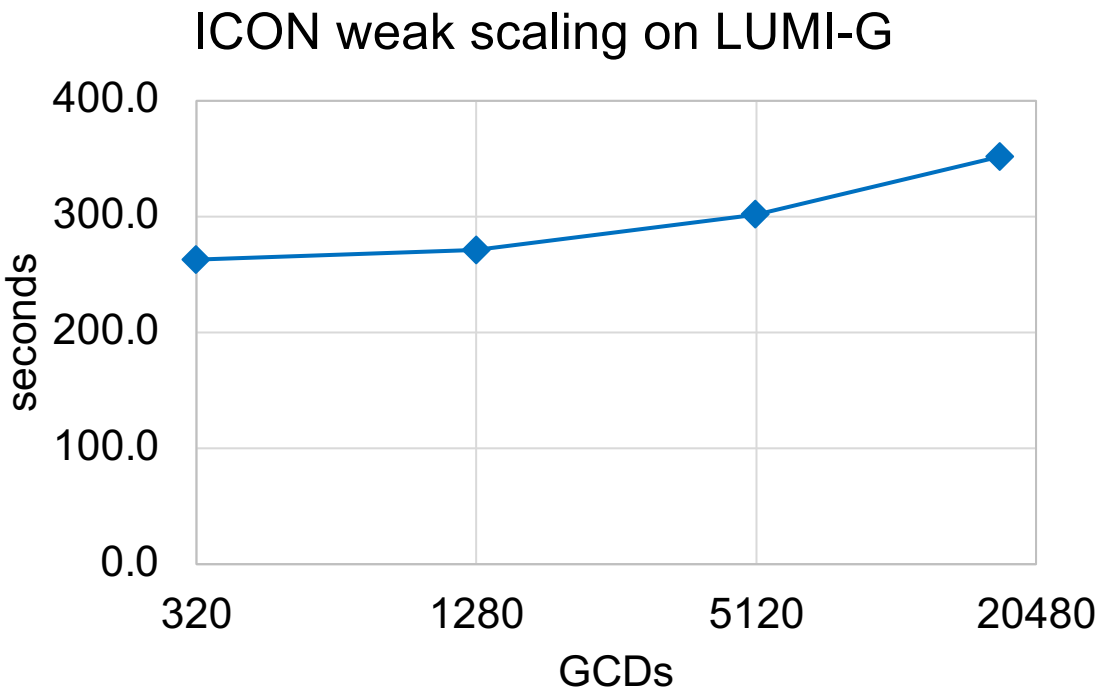
- GRIDTOOLS is a stencil framework used in key weather and climate applications
  - Implements stencil operations in a domain-specific language. Mostly memory bound.
- Initial performance did not deliver the full memory BW of MI250X on each GCD (logical GPU)
  - Default code uses hardcoded memory alignment of 128 bytes for GPU layer
  - Inherited from development work on other GPUs
- Changing hardcoded alignment to 512 bytes improved performance on MI250X by 16% (+130 GB/s)
  - You need to tune for your architecture: 7% improvement on MI100, but nothing on Nvidia A100
- Using Packed FP32 FMA instructions provided another ~3% improvement (stencil to dot product)
- Successfully weak-scales to run on the full LUMI-G system

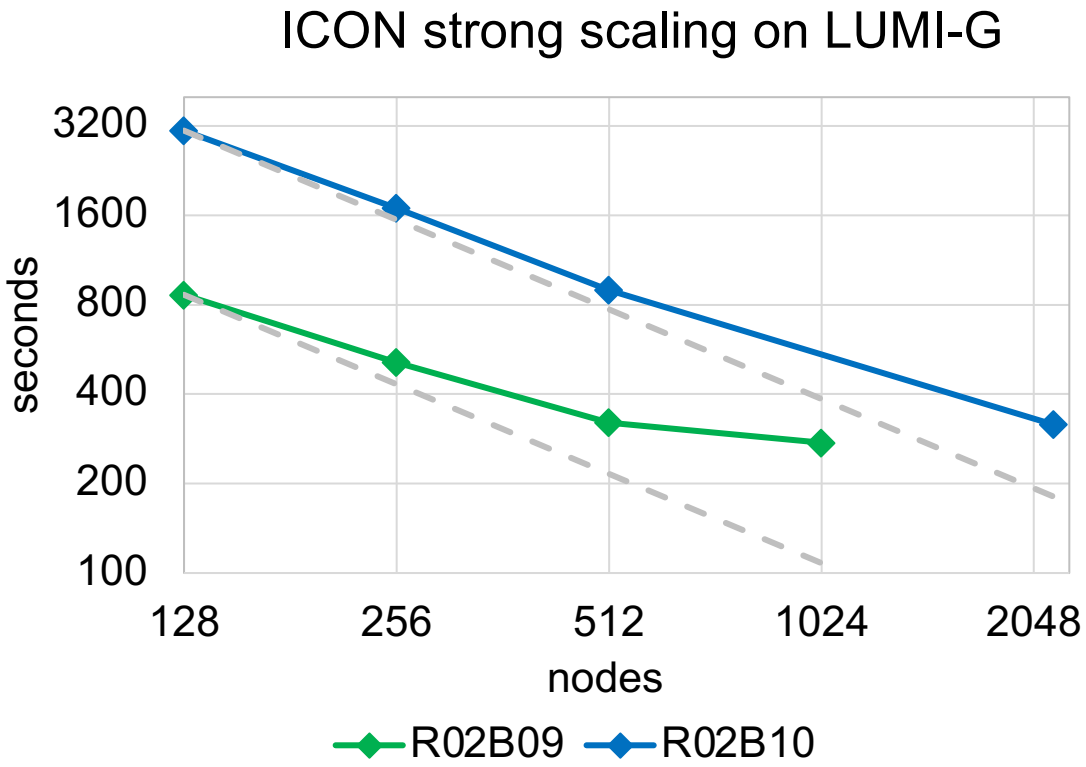




ICON (1)

- ICON is a key weather and climate modelling framework developed by DWD and numerous partners
- The Fortran code is extended by OpenACC directives for GPU offloading
- Until the LUMI installation, the GPU version was working exclusively on NVIDIA hardware
- HPE made a great effort to extend the OpenACC support of the Cray Compilation Environment (CCE)





ICON (2)

- Weak spots in CCE's code generation were worked around in the ICON code and patched in later versions

CCE-14.0.2: 161 µs
CCE-15.0.1: 95 µs

```
!$ACC PARALLEL
!$ACC LOOP SEQ
DO j = M-1, 2, -1
  !$ACC LOOP
  DO i = N1, N2
    A(i,j)=A(i,j)+A(i,j+1)*B(i,j)
  ENDDO
ENDDO
!$ACC END PARALLEL
```

CCE-14.0.2: 84 µs
CCE-15.0.1: 83 µs

```
!$ACC PARALLEL
!$ACC LOOP
DO i = N1, N2
!$ACC LOOP SEQ
  DO j = M-1, 2, -1
    A(i,j)=A(i,j)+A(i,j+1)*B(j,j)
  ENDDO
ENDDO
!$ACC END PARALLEL
```

- The latest compilers and libraries should be used to obtain the best performance with ICON
- Compiler improvements will benefit all OpenACC applications running on LUMI

	CCE 14.0.2	CCE 15.0.1
rocm 5.0.2	236 seconds	226 seconds
rocm 5.3.0	not available	222 seconds

RCCL: ML on Slingshot

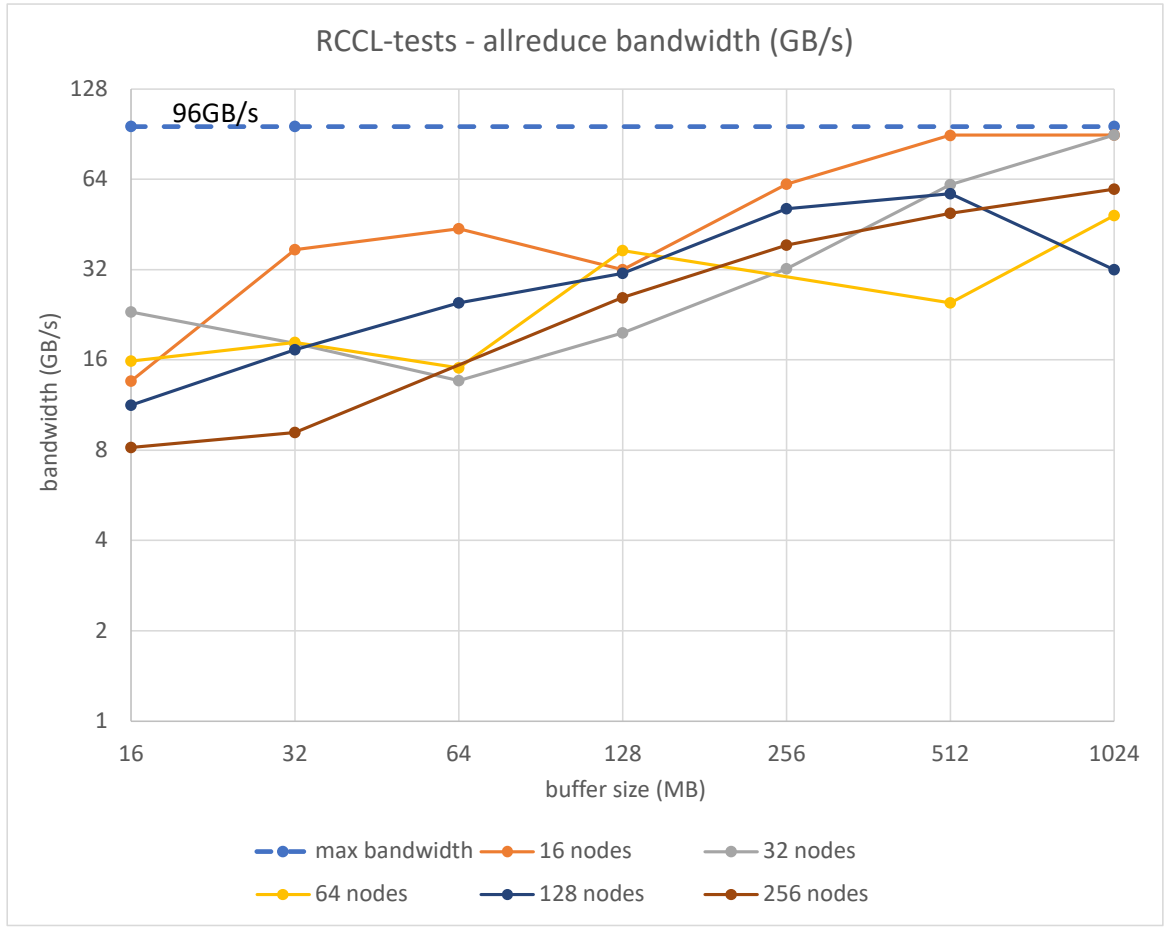
- ML/AI codes are a significant part of the production workload on LUMI
- These codes typically use the AMD **RCCL** communication framework

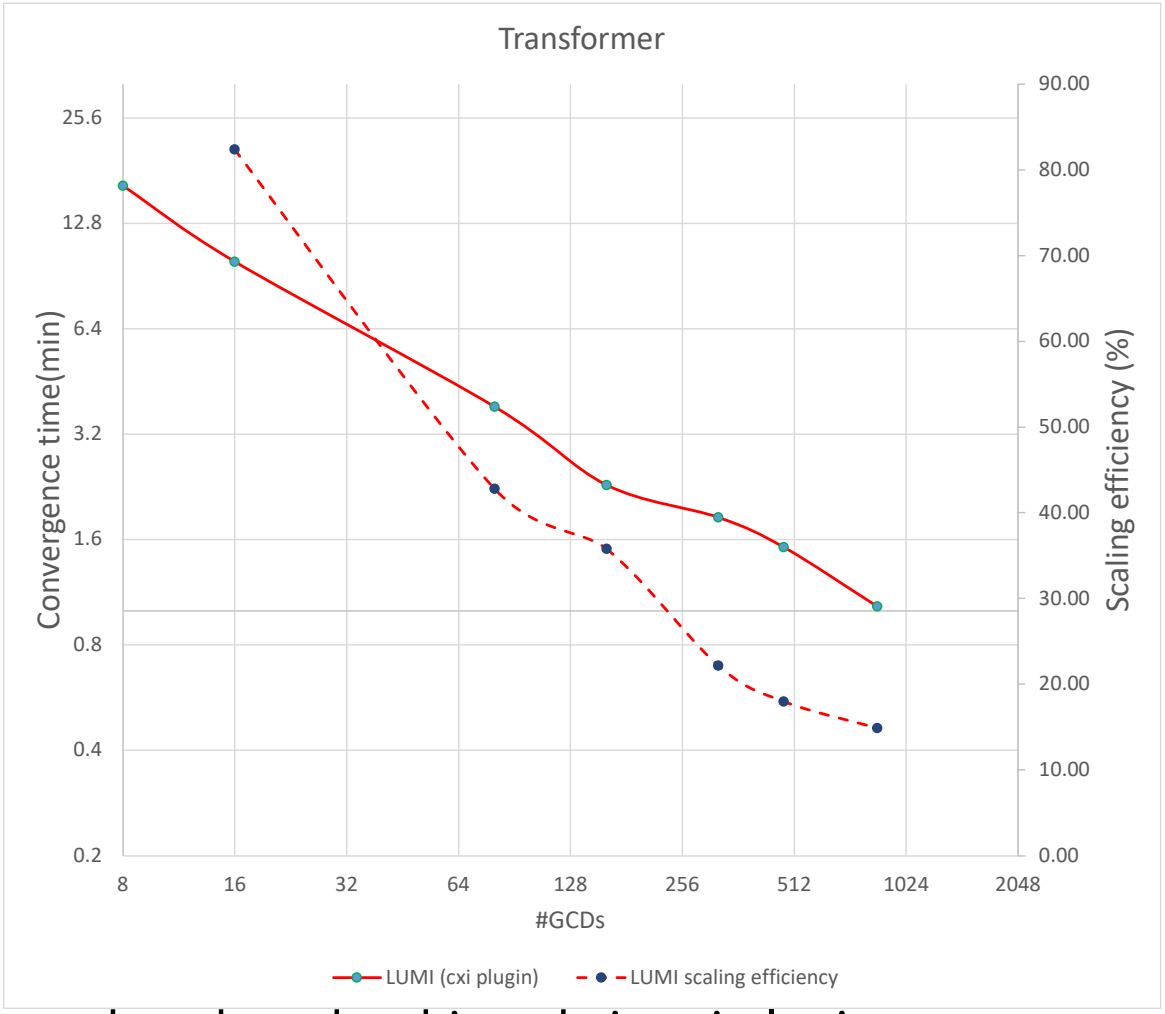
- AMD and HPE worked closely together to ensure RCCL performed optimally on the HPE Slingshot network
  - Developed a **cxi plugin** that interfaces between the RCCL communication layer and the Slingshot drivers
  - Leverage Open-fabrics interface implementation
  - The cxi plugin is a runtime dependency that allows RCCL to leverage transport layers other than the default ones
    – e.g. sockets, which are either not supported or perform poorly

- Allowed HPE to demonstrate scalability of ML benchmarks (Transformer, ResNet50, SSD) out to 230 nodes

- The benchmark-driven improvements are already being leveraged by LUMI users
  - Megatron-Deepspeed – scaling GPT-2 models up to 128 nodes (with up to 8.4B parameters, 178B tokens)
  - Early experiences with distributed deep learning on LUMI
    – https://www.lumi-supercomputer.eu/experiences-of-czech-scientists-pilot-testing-the-gpu-partition-of-lumi/
  - Lead the work to several large scale projects, including NLP:
    – https://www.lumi-supercomputer.eu/research-group-created-the-largest-finnish-language-model-ever-with-the-lumi-supercomputer/

RCCL Scalability on LUMI





Allreduce RCCL bandwidth and scalability for Transformer benchmark achieved via cxi plugin

Conclusions

- The system is the fastest in Europe and still expanding
- The LUMI system is operational and supports a diverse workload of applications
  - a mix of research areas, applications and programming models
- A diverse and representative acceptance benchmark suite was key
  - it ensured the system was ready to support this production workload from Day 1
  - it drove improvements in the system software, the device drivers and the Programming Environment
- These benefits feed down to all applications running on the system

- But we are not done: more and more user codes are migrating to the system for large-scale simulation
- The LUMI User Support Team is ready to help
  - with porting, optimisation and scaling

Thank you