EPFL

Hamza Chouh

SKA Swiss Days
02.09.2024

A co-design approach for the SKA Science Data Processor system

Main tasks

- Profiling and kernel analysis
  - Performance measurements on various hardware
  - Roofline analysis

- Benchmark suite for procurement:
  - Compute kernels
  - I/O
  - Networking

- Experimental work on HPC

Benchmark suite

- Includes a set of relevant workloads for radio astronomy applications such as FFT, gridding/degridding and deconvolution cleaning
- Other HPC characterisation benchmarks: HPCG, OSU MPI, I/O benchmark
- Meant to be portable and executed by vendors in order to assess potential HPC systems
- Tests + measurements on various clusters with different types of hardware (AMD/Intel/ARM CPUs, AMD/Nvidia GPUs)

Hamza Chouh

Intel DPC++ tests

- Objective: evaluation of the Intel oneAPI DPC++ compiler for HPC applications
- Criteria:
  - Ease of use
  - Performance
  - Portability
  - Support perspectives

Intel DPC++ tests: gridding/degridding

Hamza Chouh

- Pros:
  - Easy to implement
  - Portable (CPU, AMD/Intel/Nvidia GPUs, FPGA)
  - Unique code for all hardware, choose at runtime
  - Good performance
- Cons:
  - Needs a specific compiler
  - ARM64 architectures?
  - Might end up with several code versions anyway for specific optimizations

Intel DPC++ tests: SAXPY example

Hamza Chouh

CUDA

```
__global__ void saxpy(
    float* z, const float a, const float* x, const float* y, const size_t n)
{
  const auto i = blockIdx.x * blockDim.x + threadIdx.x;
  if (i >= n)
    return;
  z[i] = a * x[i] + y[i];
}

int main(int argc, char* argv[])
{
  const size_t n_items = 1e9;
  const float a = 2.f;

  float* x;
  gpuErrchk(cudaMallocManaged(&x, n_items * sizeof(float)));
  float* y;
  gpuErrchk(cudaMallocManaged(&y, n_items * sizeof(float)));
  float* z;
  gpuErrchk(cudaMallocManaged(&z, n_items * sizeof(float)));

  std::iota(x, x + n_items, 0.f);
  std::iota(y, y + n_items, 0.f);

  saxpy<<<(n_items + 127) / 128, 128>>>(z, a, x, y, n_items);
}
```

DPC++

```
int main(int argc, char* argv[])
{
  sycl::queue queue;

  constexpr size_t n_items = 1e9;
  const float a = 2.f;

  auto x = syclx::malloc_shared<float>(n_items, queue);
  auto y = syclx::malloc_shared<float>(n_items, queue);
  auto z = syclx::malloc_shared<float>(n_items, queue);

  std::iota(x.begin(), x.end(), 0.f);
  std::iota(y.begin(), y.end(), 0.f);

  queue
      .parallel_for(sycl::range<1>(n_items),
          [=](sycl::id<1> i) { z[i] = a * x[i] + y[i]; })
      .wait();
}
```

Intel DPC++ tests: gridding/degridding

Hamza Chouh

- Gridding performance results:

Version	Input copy bandwidth (GB/s)	Compute time (ms)	Output copy bandwidth (GB/s)
DPC++	4,1	0,9	2,7
CUDA base version	2,7	3,7	2,9
CUDA optimized version	2,7	0,5	2,9

Intel DPC++ tests: gridding/degridding

- Degridding performance results:

Version	Input copy bandwidth (GB/s)	Compute time (ms)	Output copy bandwidth (GB/s)
DPC++	4,1	6,5	2,7
CUDA base version	2,7	10,8	2,9
CUDA optimized version	2,7	0,9	2,9

EPFL

Quick overview of the SEAMS project

SEAMS

Sustainable & Energy Aware Methods for SKA

EPFL

What is SEAMS ?

- French-Swiss Collaboration

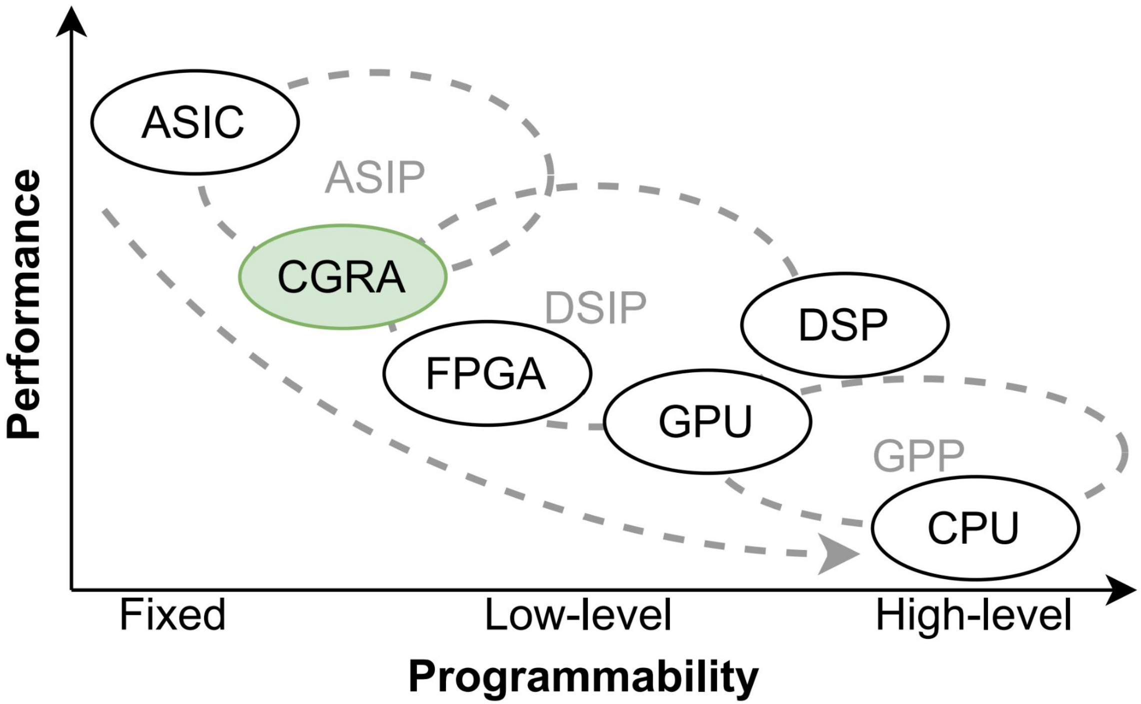
- Effort done in partnership with SKACH

- Focus on hardware and data processing

SEAMS Objectives

- Aims to demonstrate the **usage of accelerators** for the data processing part (**SDP**) associated with the SKA
  - Focus on **Coarse-Grained Reconfigurable Architectures** (**CGRAS**)
- Focus on the SKA's SDP computationally intensive kernels
  - FFT / iFFT
  - Gridding
  - nuFFT
- Aims at better energy efficiency than the classical CPU + GPU approach using
  - Objective: ~4x more energy efficient than GPUs

Advantages of CGRAs



SKA/SEAMS MiniApp effort

- We created a **MiniApp** repository aiming to gather the most common and computationally intensive radio-astronomy kernels

- The repo provides:
  - **One library per kernel**
  - Benchmarking tools to measure
    - time to solution
    - **scalability**
    - **power consumption**
  - Configurable way to play with CPUs, GPUs and **CGRAs FPGA** implementations

- Ongoing work to tune the High Level synthesizers in order to reproduce on-par results between the CPU/GPUs implementation and the FPGA.

EPFL

SKA/Seams MiniApp next

- We will continue to integrate new backend (GPUs, New hardware, etc) as part of the RACOON effort

- We will continue to integrate new kernels
  - We are interested by all the most computationally expensive kernels in your data processing pipeline (including Cleaning, calibration, etc)

- Do not hesitate to reach our team #team-racoon on slack if you are interested

More Information about SEAMS ?

- Please come to the presentation of Denisa and Rubén on Wednesday

10:05 **Scaling Sustainable Computing: Advanced ML Techniques for Enhancing the Efficiency in the SKA Regional Data Centers** 20m

Speaker: Dr Denisa-Andreea Constantinescu (EPFL Embedded Systems Lab)

- Website: https://seams-project.com/