How to use Hunter?

Björn Dick (HLRS),   Paul Saumet (HLRS),   AMD

Software environment

- Cray Programming Environment (CPE)
- As known from Hermit / Hornet / Hazel Hen
- Provides compilers:
  - Cray compiler   (C/C++ compilers based on LLVM)
  - AMD compiler   (LLVM-based)
  - GNU
- provides highly optimized versions of BLAS/LAPACK, FFTW, Trilinos, HDF5, NetCDF
- Easy to switch between compilers   (libs are automatically adapted)
- Furthermore provides debugger (gdb4hpc) and robust, comprehensive as well as easy-to-use performance analysis tools (perftools)

Memory model

- MI300A features "unified shared memory"

- i.e. CPU cores and GPU attached to same memory:

- → not necessary to copy data

- → less effort and overhead

- → **allows *incremental* porting w/o sacrificing performance**
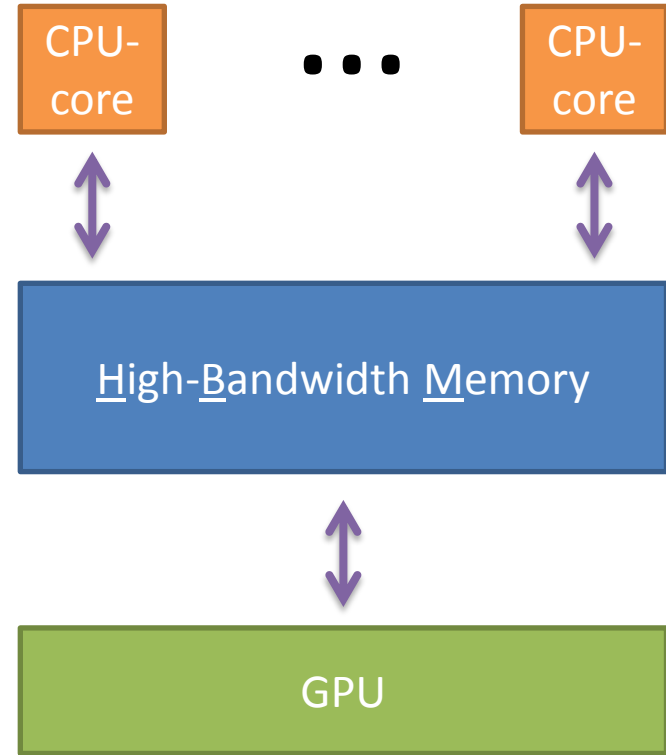
Programming models

::::: ::::: ::::: ::::: ::::: ::::: ::::: ::::: ::::: ::::: ::::

There are exceptions

Name	Approach	Effort required	Performance expected	Steering possible	Comments
HIP	native				• Code can be generated from CUDA by means of hipify script • Fortran not (directly) supported!
OpenMP (target offloading)	directives				suggested
stdpar (PSTL / do cuncurrent)	automagic ally				

Programming models - HIP

CPU CODE	GPU CODE	APU CODE

```double* in_h = (double*)malloc(Msize);```
```double* out_h = (double*)malloc(Msize);```
	```double* in_h = (double*)malloc(Msize);```
	```double* out_h = (double*)malloc(Msize);```
	```hipMalloc(&in_d, Msize);```
	```hipMalloc(&out_d, Msize);```
		```double* in_h = (double*)malloc(Msize);```
		```double* out_h = (double*)malloc(Msize);```

```for (int i=0; i<M; i++) // initialize```
```   in_h[i] = …;```
 	
```cpu_func(in_h, out_h, M);```
	```for (int i=0; i<M; i++) // initialize```
	``` in_h[i] = …;```
	```hipMemcpy(in_d,in_h,Msize);```
	```gpu_func<< >>(in_d, out_d, M);```
	```hipDeviceSynchronize();```
	```hipMemcpy(out_h,out_d,Msize);```
		```for (int i=0; i<M; i++) // initialize```
		``` in_h[i] = …;```
		
		```gpu_func<< >>(in_h, out_h, M);```
		```hipDeviceSynchronize();```

```for (int i=0; i<M; i++) // CPU-process``` ``` … = out_h[i];```	```for (int i=0; i<M; i++) // CPU-process``` ``` … = out_h[i];```	```for (int i=0; i<M; i++) // CPU-process``` ``` … = out_h[i];```

- GPU memory allocation on Device
- Explicit memory management between CPU & GPU
- Synchronization Barrier

AMD
together we advance_

Programming models - OpenMP

OpenMP® CODE

```
#pragma omp requires unified_shared_memory
 double* in_h = (double*)malloc(Msize);
 double* out_h = (double*)malloc(Msize);




for (int i=0; i<M; i++) // initialize
    in_h[i] = …;

#pragma omp target
{ … }


for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```

running on

CPU

GPU

CPU

Synchronous execution, but might be wise to use this opportunity to switch over to asychronous tasking approach (i.e. nowait clause)!

- ~~GPU memory allocation on Device~~
- ~~Explicit memory management between CPU & GPU~~
- Synchronization Barrier

AMD
together we advance_

C++

```cpp
std::transform( // needs <algorithm>
    std::execution::par_unseq, // <-- needs <execution>
    indices.begin(), indices.end(), grid.begin(),
    [](size_t index){
        return expensive_calculation(index);
    }
);
```

Fortran

```fortran
do concurrent (k = 1:n, j = 1:n, i = 1:n)
   do_stuff(i, j, k)
end do
```

Programming models - Compiler support

Language	Model	Compiler			Comments
		Cray	AMD	GNU	
Fortran	HIP	✗	✗	✗	but possible via ISO_C_BINDING
	OpenMP	✓	✓	?	GNU uses ROCm under the hood for OpenMP/OpenACC offloading. MI300A not supported yet. But *might* be possible with recent enough ROCm.
	OpenACC	✓	✓	?	
	stdpar	✓	?	?	not working yet (but should)
C	HIP	✓	✓	✗	Cray passes HIP sections to AMD compiler. Might be possible for GNU as well.
	OpenMP	✓	✓	?	cf. Fortran
	OpenACC	✗	✗	✗	
C++	HIP	✓	✓	✗	cf. C
	OpenMP	✓	✓	?	cf. Fortran
	OpenACC	✗	✗	✗	
	stdpar	✓	✗	✗	

Programming models - Interoperability

- Possible to mix them in your application

- → allows for e.g. using OpenMP for majority of code but HIP in very hot kernels

- But within a kernel / offloading unit, you can use a single model only.

Need assistance?

- Join our "Hunter Code Preparation Workshop"
  - July 15th to 19th
  - hybrid
  - Port your workflow to CPE on Hawk & identify hot loops
  - AMD might shape porting efforts w.r.t. OpenFOAM, GROMACS, etc. based on your profile gathered in this workshop
  - https://www.hlrs.de/training/2024/hpe1

- Another workshop in November
  - test and improve your code on Hunter
  - More information will follow