Acceleration made easy - Speeding up your science codes with (fairly) little effort

Wolfgang Hayek
eResearch NZ 2016

New Zealand eScience Infrastructure

Overview

1. What accelerators are and when to use them
2. Accelerating Python with PyOpenCL/PyCUDA
3. Accelerating C, C++, and Fortran with OpenACC
4. Summary

What accelerators are and when to use them

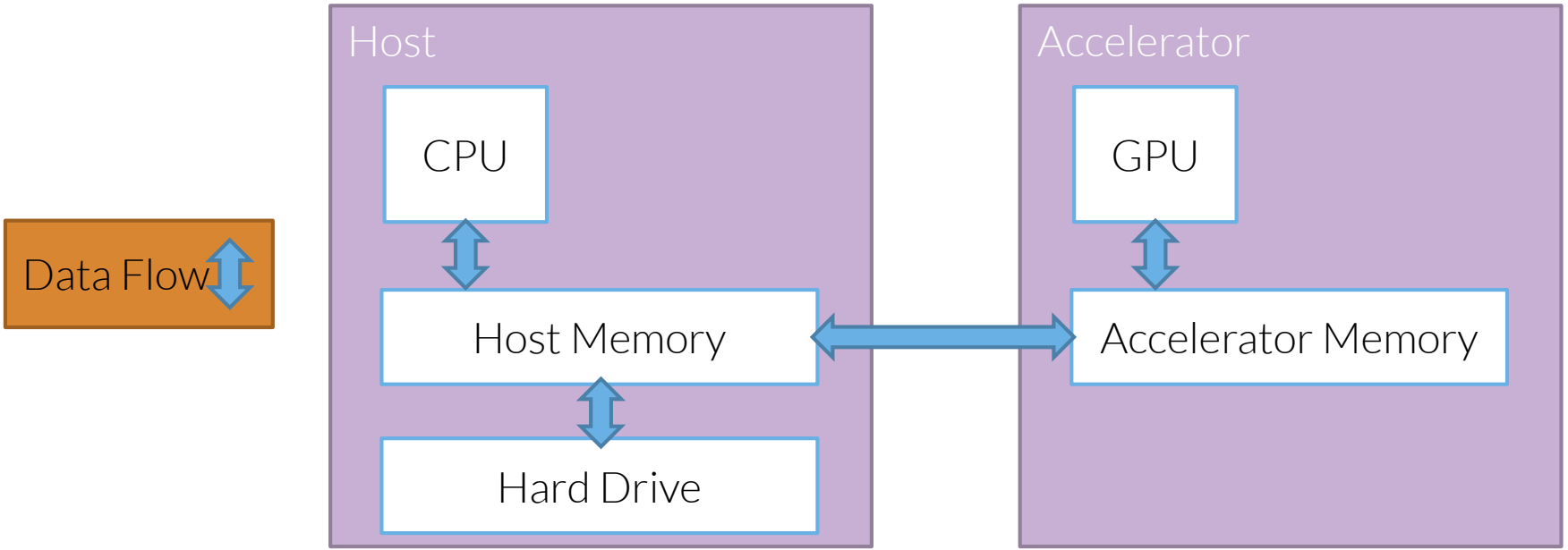
Accelerators

- Accelerators: co-processors that perform some functions faster than CPUs
- Applications include
  - Signal processing, Cryptography, Graphics
  - Scientific computing
- We will focus on scientific computing with GPUs
- Ongoing NeSI project: “Exploring tsunami and flooding code speed-up using GPU processing”

Using accelerators

- CPU – mostly serial computation (each step in a program is executed in order)
- GPU – highly parallel computation (many steps executed at the same time)
- Accelerators only useful for parallel compute-dominated applications (not, e.g., disk I/O)

Typical Accelerator Computing (today!)



Data needs to travel – this takes time and adds complexity

Accelerating Python with PyOpenCL/PyCUDA

PyOpenCL/ PyCUDA

- Python: popular multi-purpose computer language
- PyOpenCL/PyCUDA: extension packages for accelerators
- Full access to OpenCL/CUDA APIs
- We will focus on PyOpenCL Array syntax
- Same code runs on laptop and on HPC (e.g. NeSI Pan cluster, cloud, …)

PyOpenCL/ PyCUDA

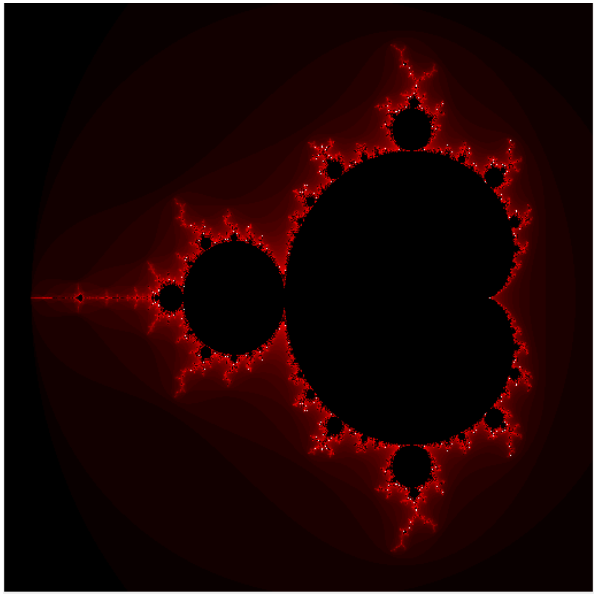
NumPy

```
# Program loads data into “data_np”




# Compute part
x_np = 2 * data_np
result_np = 1.4 * np.exp(-x_np * x_np)




# Rest of program
```

PyOpenCL Arrays

```
# Program loads data into “data_np”
# Program initialises accelerator

# Copy data to accelerator
data_cl = cla.to_device(queue, data_np)

# Compute part – entirely on accelerator!
x_cl = 2 * data_cl
result_cl = 1.4 * clm.exp(-x_cl * x_cl)

# Copy data back to host memory
result_np = result_cl.get()

# Rest of program
```

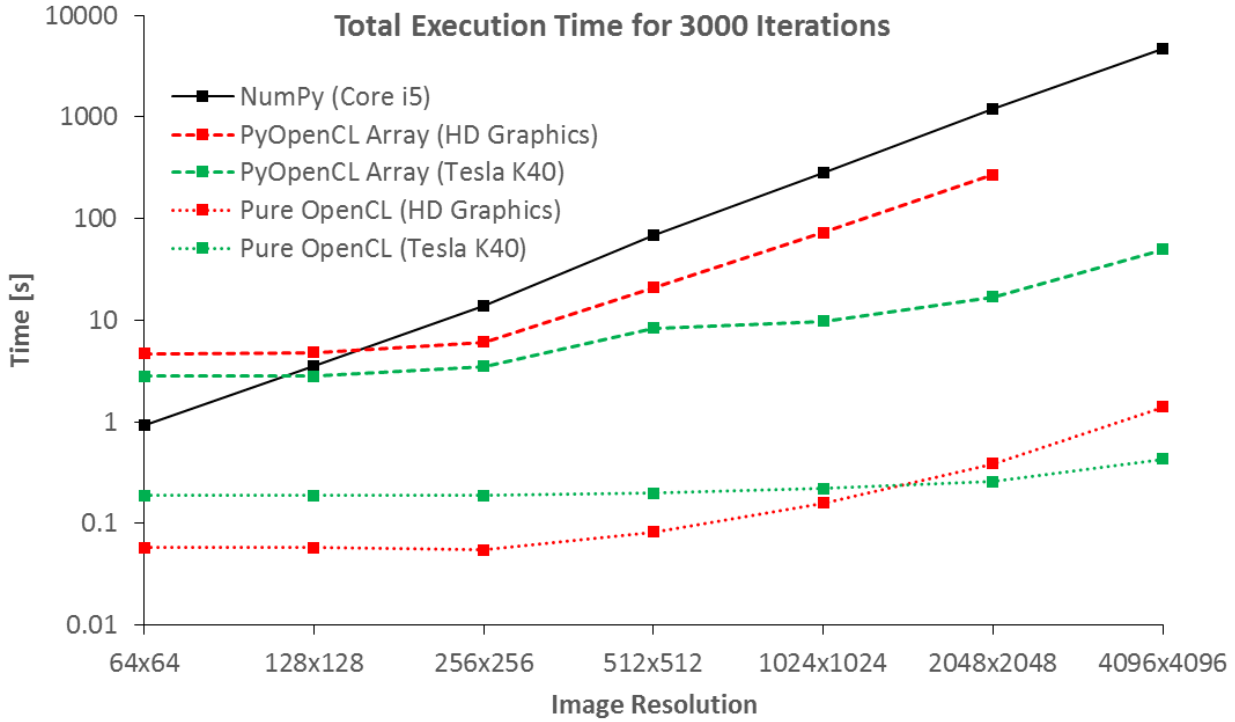
PyOpenCL/ PyCUDA



- Mandelbrot is idealised case - perfect match for GPUs due to inherent parallelism

PyOpenCL/ PyCUDA

Mandelbrot Performance



PyOpenCL/ PyCUDA

- Good speed-up possible with PyOpenCL Arrays, but need large work sizes
- Hand-coded OpenCL is more efficient - just as with NumPy vs. C/Fortran
- NeSI can help with getting you started!

Accelerating C, C++, and Fortran with OpenACC

OpenACC

- Various ways to use accelerators in C, C++ and Fortran: scientific libraries, CUDA, OpenCL, …
- OpenACC: compiler directives for “offloading” loops, main code stays on CPU
- Useful for existing code (NeSI tsunami project)

OpenACC

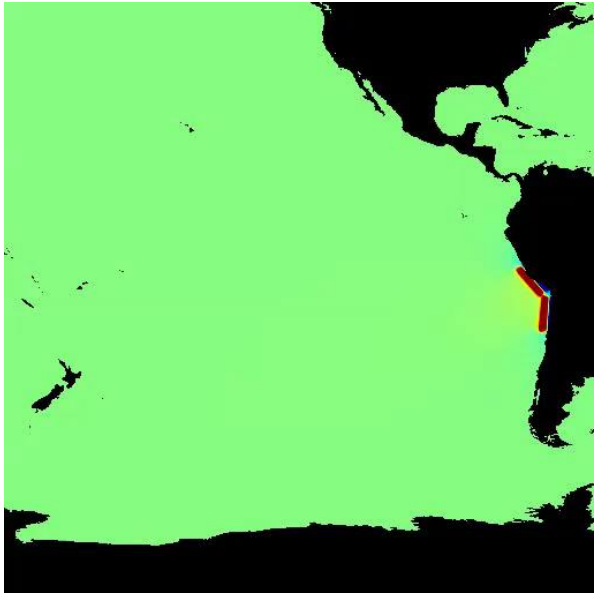
C

```
/* Program loads data into “data” */


/* Compute part */


for (int i = 0; i < n; i++) {
   float x;
   x = 2.0 * data[i];
   result[i] = 1.4 * exp(-x*x);
}

/* Rest of program */
```

C with OpenACC

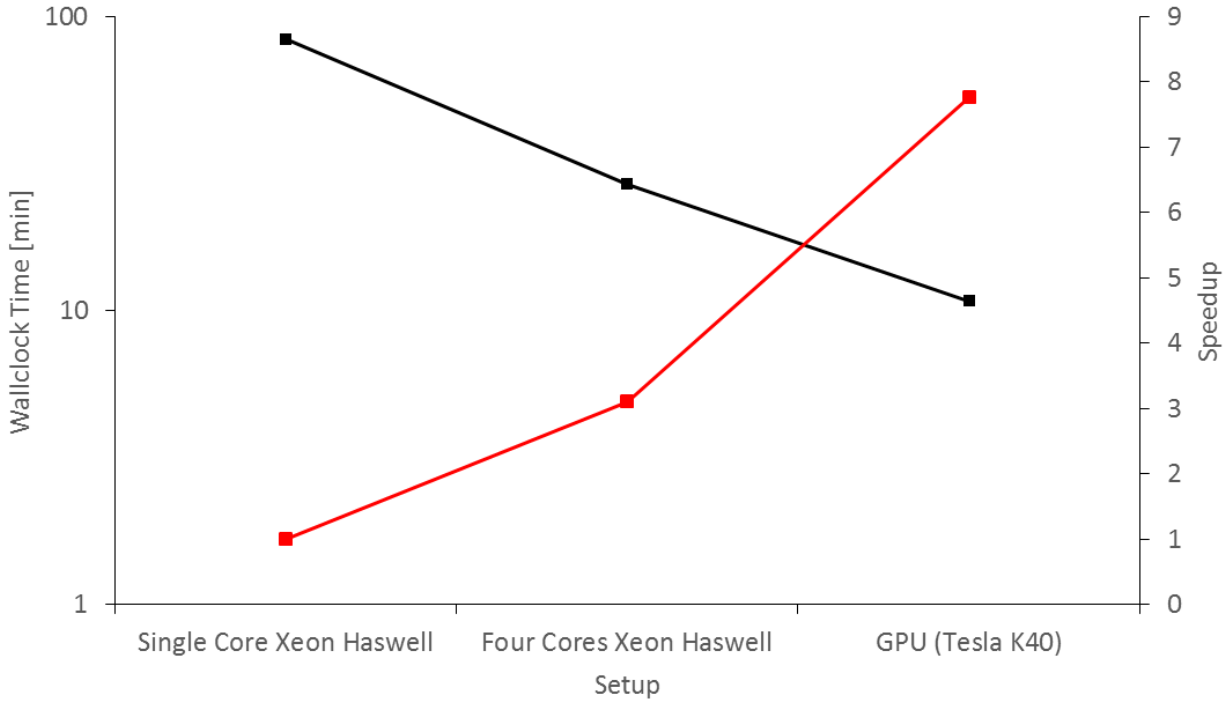
```
/* Program loads data into “data”
    Program initialises accelerator */

/* Offload compute part to accelerator */
#pragma acc kernels \
 copy(data[0:n], result[0:n])
for (int i = 0; i < n; i++) {
   float x;
   x = 2.0 * data[i];
   result[i] = 1.4 * exp(-x*x);
}

/* Rest of program */
```

NeSI Tsunami Project



- Basilisk (www.basilisk.fr)
- Shallow Water simulation of tsunami wave
- Model by Emily Lane, NIWA

Basilisk Performance (preliminary!)



OpenACC

- Good speed-ups, but data locality can be complex
- Hardware knowledge needed for optimisations
- NeSI can help with getting you started!

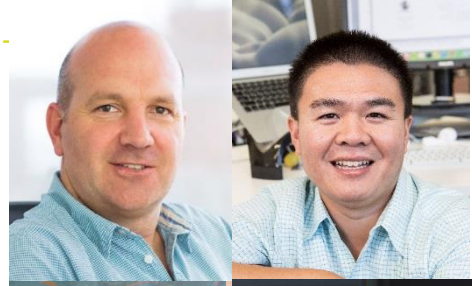
Summary

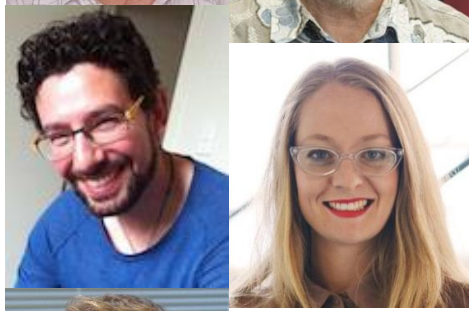
Summary

- PyOpenCL/PyCUDA and OpenACC: easy ways to use accelerators
- Workloads need to be sufficiently large
- Some algorithms may need to be changed
- Talk to us if you would like to try!

NeSI at eResearch NZ 2016

Tuesday 11:15 (Queenstown)	NeSI Update	Nick Jones
Tuesday 11:45 (Remarkables)	Growing NZ's Researcher's Computing Capability	Georgina Rae & John Rugis
Tuesday 14:30 (Queenstown)	An HPC Implementation of the Finite Element Method	John Rugis
Tuesday 16:00 (Queenstown)	The NeSI National Platform Framework	Michael Uddstrom
Tuesday 16:30 (Queenstown)	Early Experiences with Cloud Bursting	Jordi Blasco
Tuesday 16:30 (Remarkables)	NeSI – NZGL Alliance	Dan Sun & Nic Mair
Tuesday 17:00 (Queenstown)	Acceleration Made Easy	Wolfgang Hayek






NeSI
New Zealand eScience Infrastructure

Thanks for listening!