PAX @ Durham

PAX-HPC face-to-face meeting

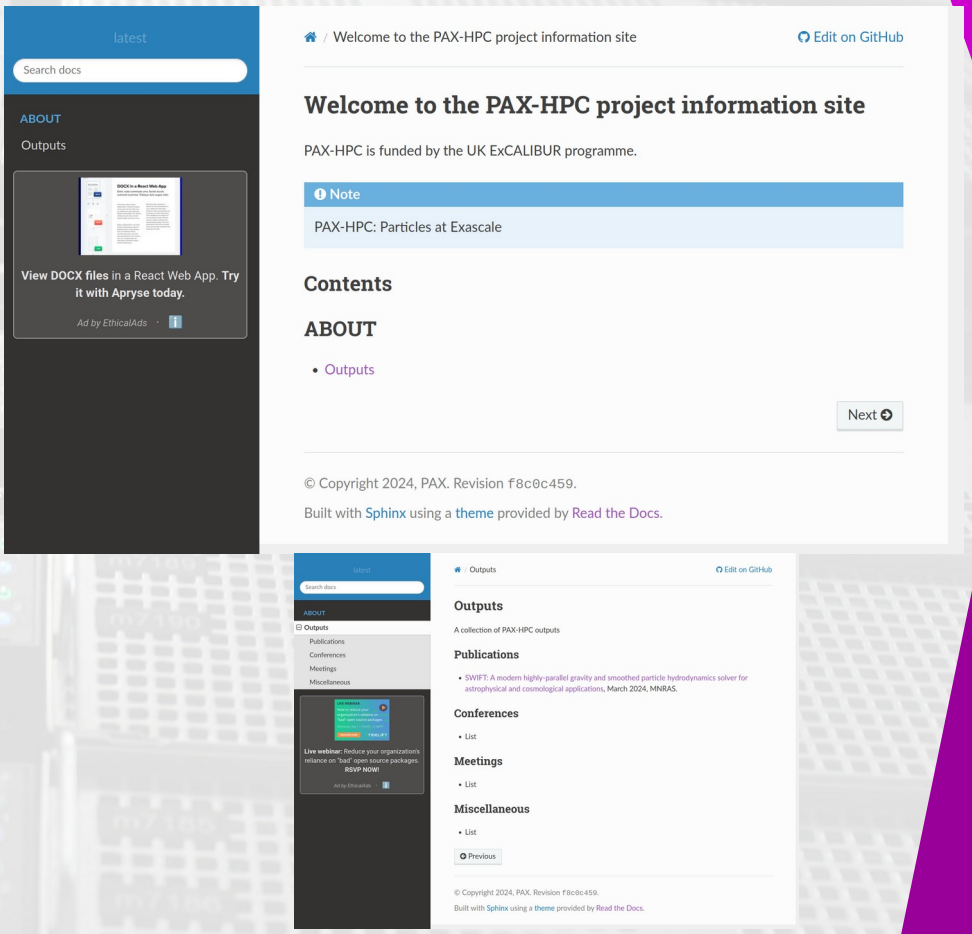
Lancaster

Alastair Basden + others

Durham University / DiRAC

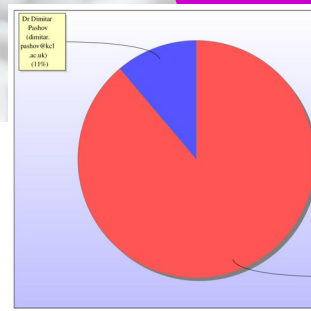
A new website?

- https://pax-hpc.readthedocs.io
- For information purposes
  - Is this what we want?
    - (in lieu of anything better)
  - Can be used for storing outputs
- Anyone can add content:
  - Fork and clone the git repo
  - Make changes
  - Submit a merge request
    - Which I then authorise
  - Website is then automatically updated
- Note - no private section
  - e.g. may not be suitable for minutes



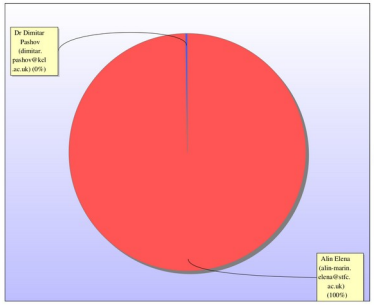
DiRAC time: Tursa and COSMA8

- We had a DiRAC allocation from October 2023-April 2024
  - 2 quarters-worth of allocation
  - 36/50k GPU hours used
  - 0/100k core hours used
  - Job sizes of up to 4 nodes
    - (>99% 1 node)



Total Allocation
Dr Dimitar Pashov (dimitar.pashov@kcl.ac.uk)
Alin Elena (alin-marin.elena@stfc.ac.uk)

A total of 814 jobs were submitted on es-tursa-gpu in project dp291 (Materials Science/PAX) during the period Oct 2023 - Mar 2024.



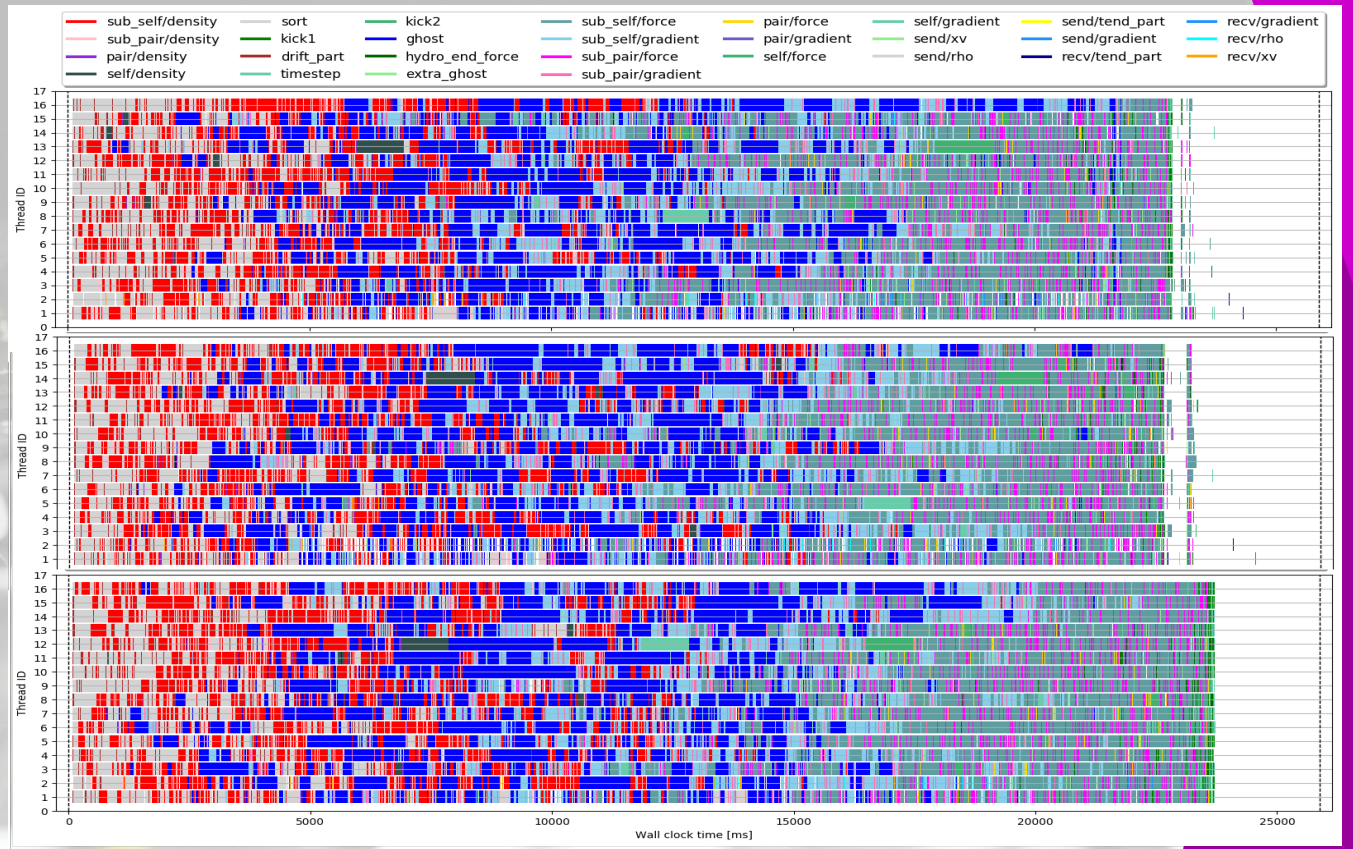
Alin Elena (alin-marin.elena@stfc.ac.uk)
Dr Dimitar Pashov (dimitar.pashov@kcl.ac.uk)

SWIFT communications work

- Thanks to Peter Draper
- SWIFT uses internal task scheduling based on pthreads
  - Lower level control than OpenMP
  - Tasks scheduled when data locked and no prerequisites
- MPI communications are tasks (asynchronous)
    - Overlap between comms and computation
  -

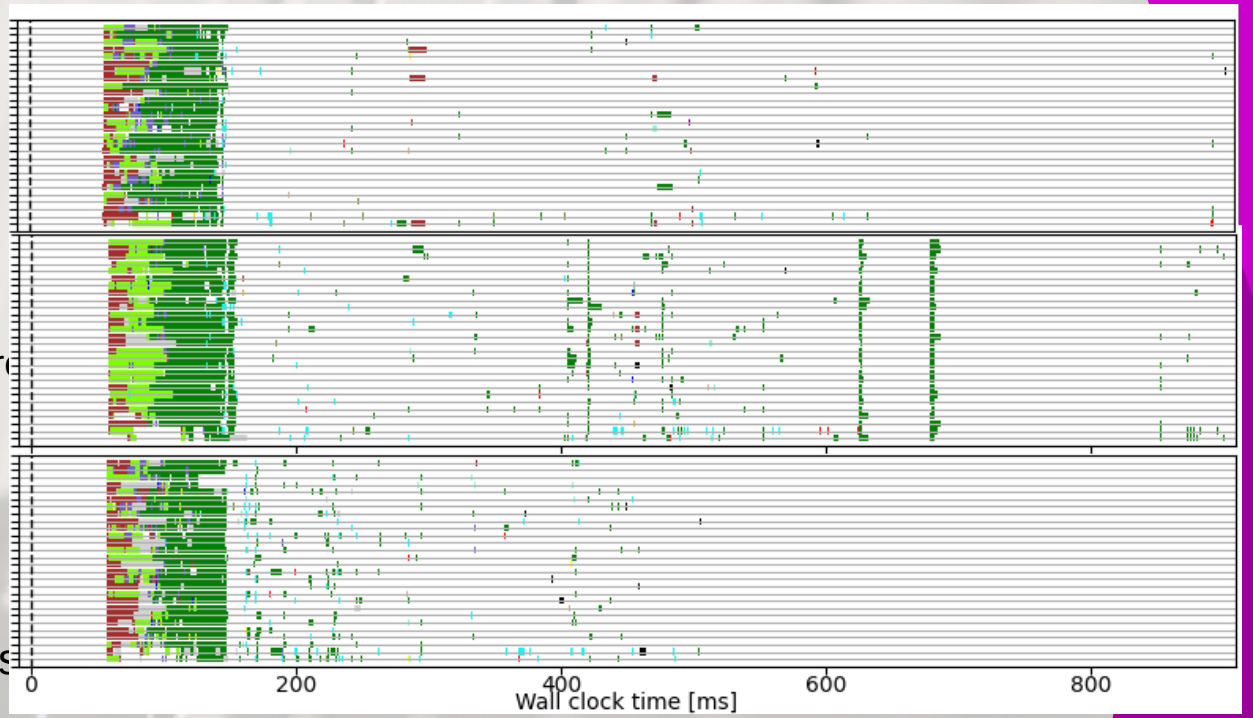
Task plots

- Ranks are kept busy
- Some inbalance



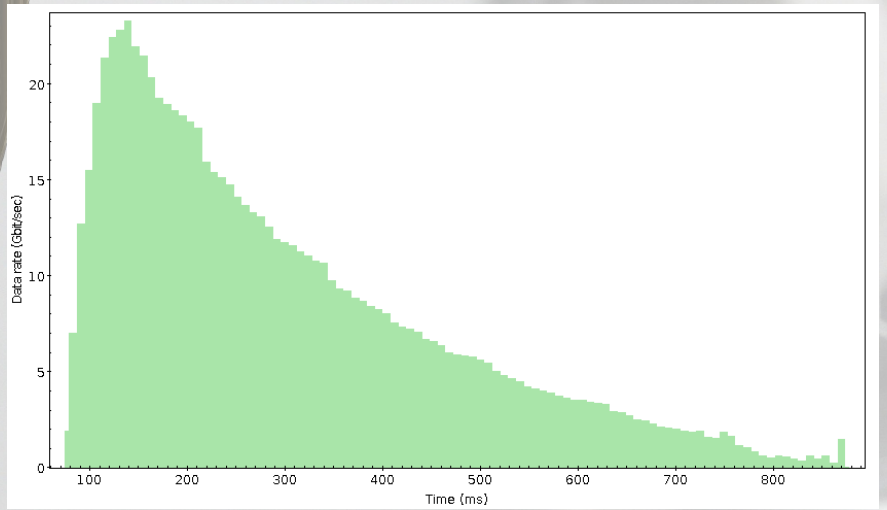
With limited computation

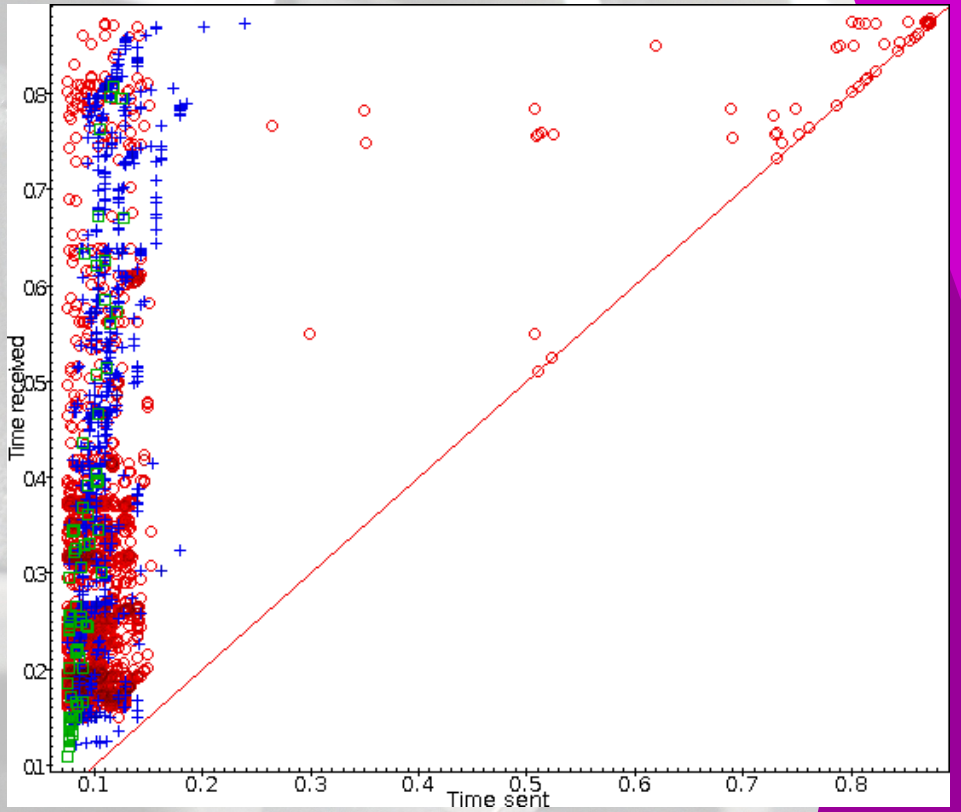
- A lot of dead time
- MPI progression?
- Difficult to work out
  - No direct timing of data transfers
    - Just when the async MPI calls are made
  - Task ordering is non-reproducible
- Internal logger implemented
  - MPI busy during the dead times



Time sent vs time received

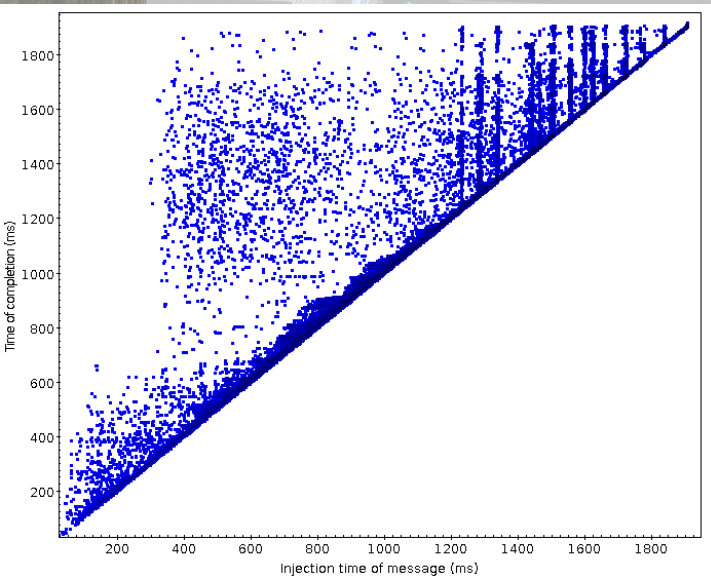
- Takes a long time to process the messages
- Not bandwidth limited
  - Expected peak of 25Gb/s

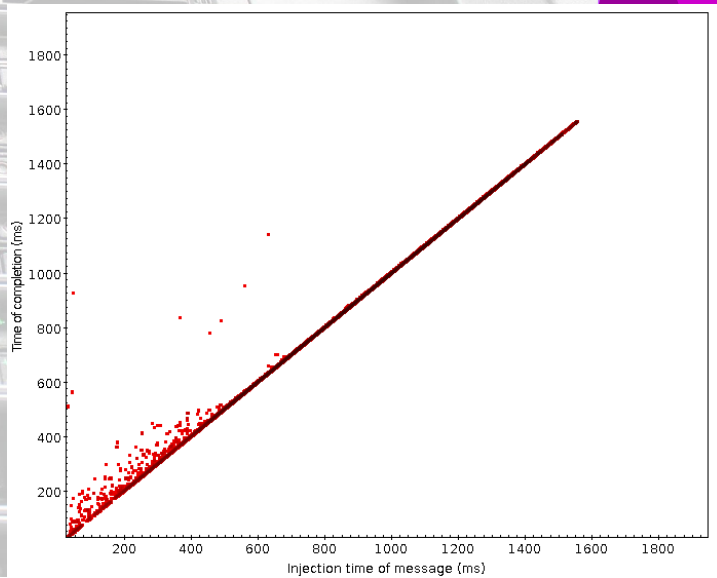




RDMA simulator

- Significant improvement over MPI
- Implemented in SWIFT - RDMA SWIFT
  - No real improvement
  - Due to time spent on data movement into and from the RDMA (or MPI) buffers





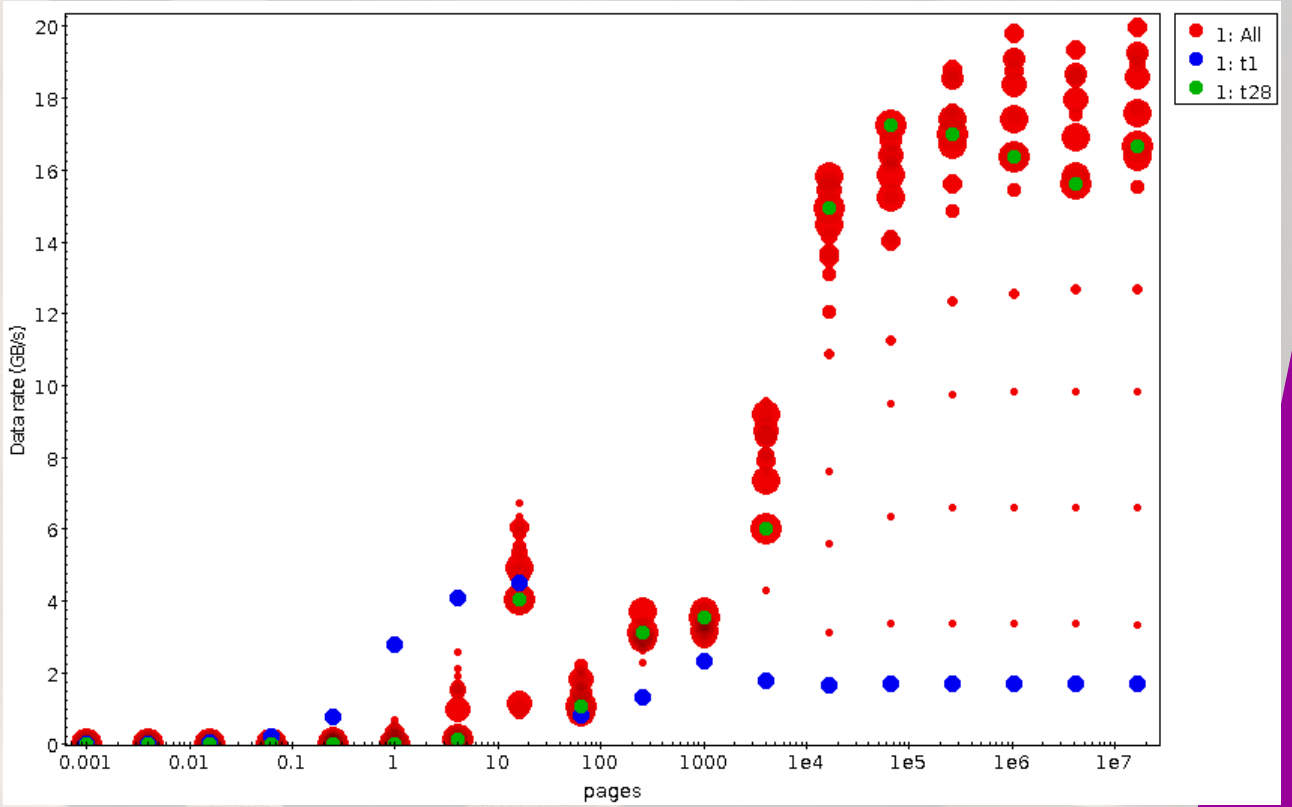
Threaded RDMA

- Using extra threads to do the memory copies
  - Sees an improvement
  - Main issue is not making use of full memory bandwidth
  - So not a problem with MPI
    - Though this is helping to hide some of the implementation details
  - Might be better to share the memory with RDMA
    - A job for SWIFT-2
    - Not straightforward to implement in MPI (hidden from user)
    - Could look at splitting sends into smaller parts (more tasks)
      - But tests don’t show improvement - probably due to overheads

Single vs multiple copying threads

How many theads to use?

- Not all of them!



The future

- This isn’t easy to integrate with MPI
  - Probably requires a redevelopment of MPI!
- Wait for better hardware?
  - Reduced fabric and memory latency
  - Direct memory sharing (CXL composability)
    - Using an MPI shared-memory communicator
  - MPI hides a lot of complication, so would be a difficult decision to move away from

SWIFT on GPU

- Sarah Johnston, Durham - PhD part-funded by Dell/AMD
  - Looking at the gravity tasks
  - Good progress being made
    - Now working with AMD and NVIDIA GPUs
      - Performance improvements required
  - Next steps to enable multiple streams/multiple GPUs
  - Good occupancy shown in profiling
  - Watch this space
- Abouzied Nasar, Manchester - PAX
  - Looking at the tasking and smoothed particle hydrodynamics
- Taking different but compatible approaches

Genoa performance

- AMD Genoa - 96 and 128 core processors
  - ~2x performance gain over Milan for SWIFT
  - >20% energy reduction per science done

CPU type	mean time	cores	frequency	max boost
bergamo:	4662.218	256	2.25	3.1
genoa:	4476.0693	192	2.4	3.7
milan:	8145.8936	128	2.45	3.5
rome:	9302.112	128	2.6	3.3

- Impressive gain for 1 generation (mostly AVX512-related)

Is GPU the future?

- Some speed-ups
  - 2-4x?
- But speed-up per £? Or per Watt?
- At the cost of RAM
  - Limiting maximum size of capability jobs

Hardware landscape

- Recent changes:
  - Doubling of memory bandwidth in one AMD generation (Milan to Genoa)
  - Doubling of network bandwidth (HDR to NDR)
    - No latency improvement
  - First CXL systems
    - Nothing of note yet, but CXL3.1 is interesting
      - Shared memory fabrics
  - Rapid increase in small-float/int performance in GPU
  - SSD storage - 1PB in 1U for £100k
  - Composability options increasing
  - AIRR systems
  - The Pre-Exascale system (EPCC)
  - Intel and AMD GPUs
  - UKRI emphasis on federation

Future HPC requirements

- What type of system does PAX require?
  - Most future UK systems will be primarily GPU
  - DiRAC will still take the bespoke design route
    - Optimised RAM, cores, storage, accelerators, fabric, etc
  - What would be ideal for our workloads?
    - Now is the time to feed into the designs
    - Can we give concrete examples
    - Please send me your thoughts, ideas, requirements, etc.

Conclusions

- SWIFT-2 likely to need a rewrite
  - Faster processors and RAM helps
  - Memory movement is key
- What would our ideal system/systems look like?
- Would more DiRAC time be useful?
- Please help update the documentation