

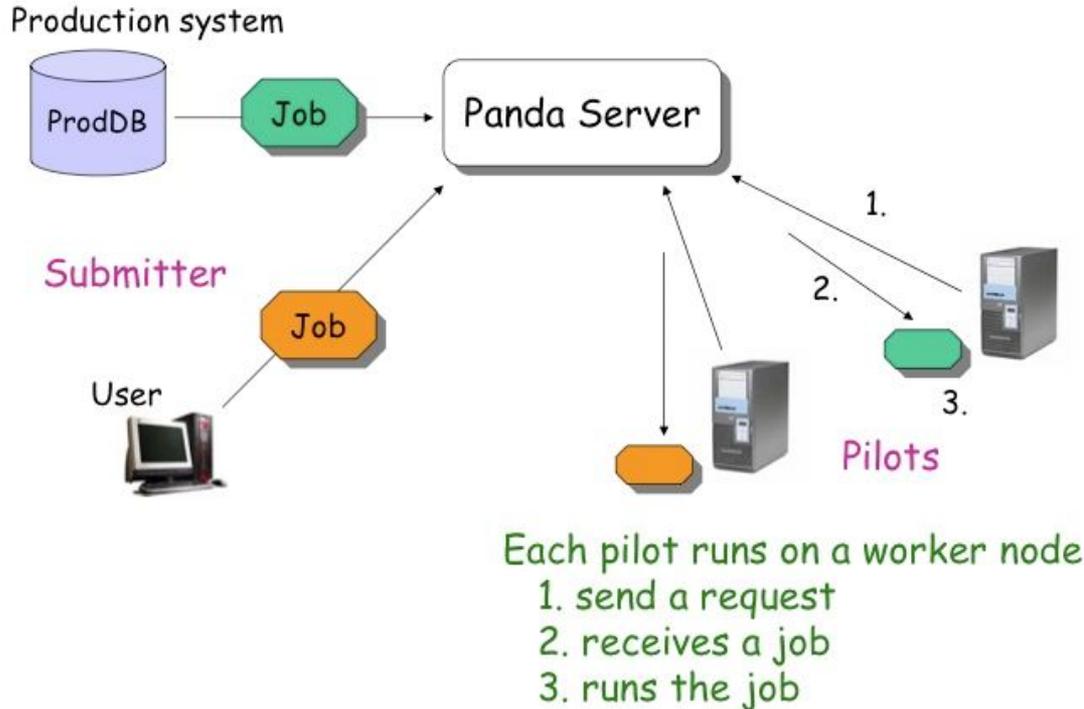
Brokering to heterogeneous resources: ATLAS

Rod Walker, LMU 7th May 2023

Which resources?

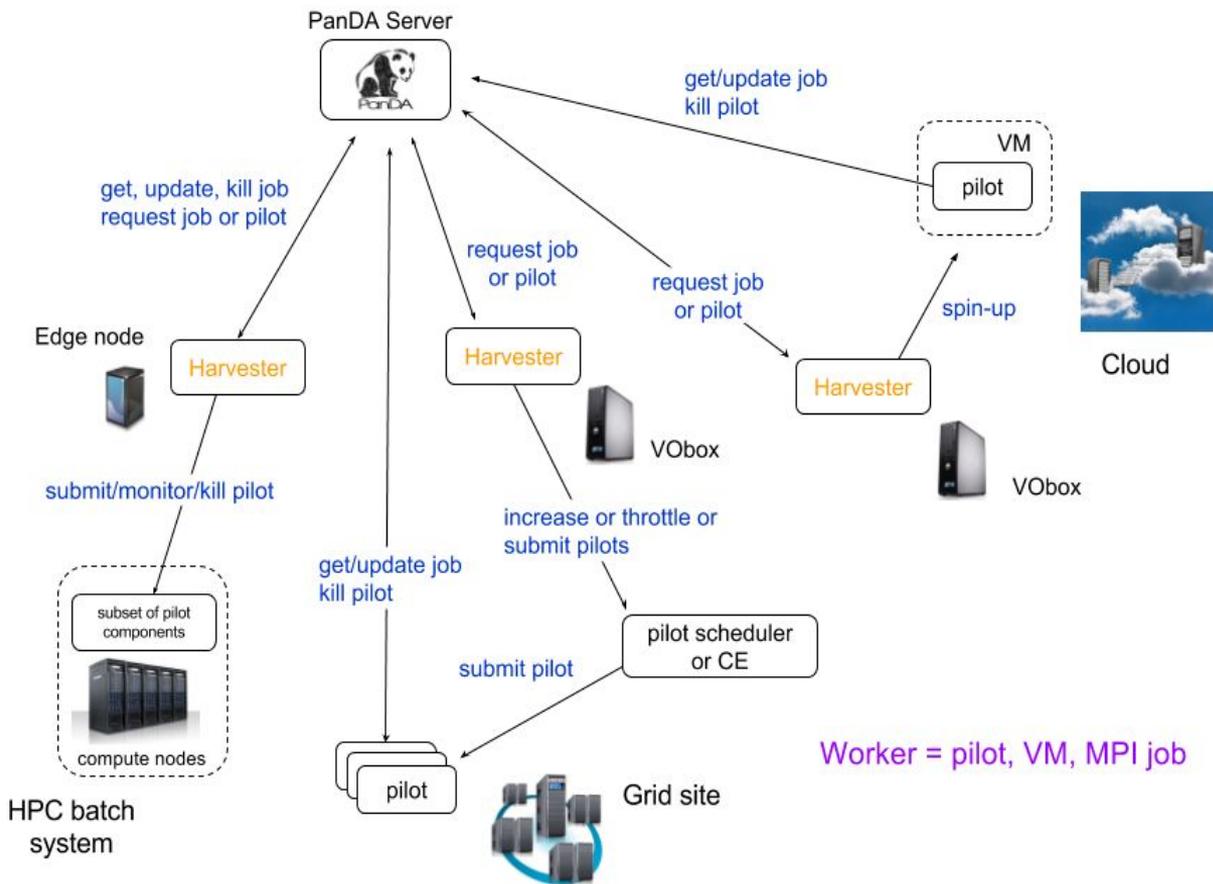
- ARM
 - ARMv7,8,9
- GPU
 - Nvidia, AMD, Intel
- x86 instruction sets
 - SSE4.2, avx2, avx512
- Hybrid GPU with significant CPU
 - offloading some parts to GPU

ATLAS distributed computing: Panda



- Pull model
 - assumes homogeneous resources in Panda Queue(PQ)
- Multiple PQs allowed per resource, but restricts following priorities
 - e.g. leave vCPUs idle to run hi-mem jobs
- Unified PQs have 4 sub-resources
 - SCORE/MCORE/*_HIMEM

Harvester - pilot factory



- pull-mode pilot stream per **sub-resource**, according to queued jobs and priority
 - e.g. only HIMEM if best
 - prevents thrashing
 - granularity is limited by scaling
 - new arch dimension?
- supports push-mode
 - pilot submitted with pre-loaded job
 - job requirements passed to batch
 - walltime, RSS, cores but also arch
 - fine granularity

Architecture, Vendor, Instructions, Model

- Added support for brokerage of GPU, CPU arch/vendor & instruction sets(~2019)
- Early use-cases
 - intel-only, to get identical maths hardware optimization for validation
 - user analysis on GPU
 - new major SW release needed SSE4.2 and we still had cpus not supporting it
- Want to use same method for all heterogeneous resources
 - expected to need brokerage for avx2, GPU models and ARM
- CRIC architectures associated to Panda Queues - homogeneous!
 - aarch64, x86_64, nvidia
 - most sites have nothing set, so implicitly x86_64
- Intel/AMD agreed [nomenclature](#) v1-4:[sse2,sse4.2,avx2,avx512], e.g. x86-64-v3 includes avx2
 - corresponds to CXXFLAG -march=x86-64-v3
 - probably only need vN rather than particular instruction set

Associated PandaQueue architectures

Type	Architectures	Vendor	Instructions	Model	Ops
cpu	aarch64, excl				 

ATLAS build optimizations

- ATLAS builds are using only v1, except in test builds(v2)
 - no significant out-of-the-box gain
 - not surprising since no dedicated development
- SSE4.2 dependency was introduced via GAUDI build
 - in 2020 had to retire some very old nodes to avoid missing instruction crashes
 - SSE4.2 believed to have been removed from GAUDI again, at some point
- Remaining SSE4.2 dependency comes from LCG (OpenBLAS)
 - was avx2 but reduced to sse4.2 after complaint
- Could immediately use SSE4.2(v2) for ATLAS builds, since already have dependency through LCG, i.e. no loss in CPU
- Requiring v3 would lose 5-10% of resources, but decreasing with time
 - potentially mopped up by old releases, other VOs, etc.
- Using v2/3 as standard would motivate developers to optimize for new instruction sets - although probably still no large gain.
- Have validated sim/reco on aarch64 build - v7?
 - unclear if optimization per version brings significant improvement

Brokerage - choose a PandaQueue

- Task architecture

sw_platform<@base_platform><#host_cpu_spec><&host_gpu_spec>

- host_cpu_spec is architecture<-vendor<-instruction_set>>
- host_gpu_spec is vendor<-model>.

- 'excl' means task must explicitly request this feature

- e.g. nodes with GPU also have x86_64 cpu, but should not run non-gpu jobs here

- production request or user client can set architecture

- '#aarch64', '#x86_64-* -v3', #x86_64&nvidia-kt100

Type	Architectures	Vendor	Instructions	Model
gpu	nvidia	nvidia, excl		
cpu	x86_64	intel	avx2	

Using multiple architectures for a task

- Can tell task to run only on ARM, or only on x86
- If ARM and x86 validated for a particular release
 - want to run task on either ARM or x86, choosing the corresponding SW?
 - this does not work today
- TODO:
 - some syntax, `task.architecture='#x86_64 | #aarch64'` handled in brokerage
 - job `sw_platform` set to one or the other once brokerage sets the site
 - this is the software setup by the pilot for the job
- In theory: same for optimized builds for different CPU/instruction sets
 - `'#x86_64 | #x86_64-* -avx2'` - need speed improvement to go to this bother
 - build-time, validation, cvmfs space
- On a mixed PQ, pilot probes WN arch or instruction set and chooses the right setup
 - no thrashing because job always finds a sw build they can use

Scheduling payloads in whole-node slot

- Complimentary workloads on a node
 - HPC node with GPU: mix jobs to use 100% cpu and 100% GPU
 - athena offloading algorithms to GPU, in future. Will not use full GPU.
 - mix io/cpu bound payloads on standard CPU node
 - problems
 - stay within other limits: scratch space, RSS, diskio
 - finish at the same time
- Requires some sort of scheduling, if not provided by local batch system
 - overlay BS: GlideinWMS, CloudScheduler, TARDIS - no network on HPC!
 - add most basic case to pilot - niced background MCORE G4 sim
 - stand-alone Condor scheduler service started and used by pilot
- Need to see the concrete use-cases

Conclusion

- Ready for heterogeneous resources in homogeneous Panda Queues
 - CPU(arch-vendor-vN), GPU(vendor-model)
 - some tuning required for GPU - little experience so far
- Minor work needed to run same task on PQs with aarch64 or x86_64
 - have a validated aarch64 release, but no ARM resources yet
- Minor work to choose optimized build on mix of instruction sets
 - dynamically choose aarch64/x86_64 seems unnecessary: separate PQs
 - would need all payloads to have builds for both
- Open questions:
 - instruction set optimizations worth the trouble? Performance improvement %?
 - hybrid GPU/CPU node scheduling
 - multi-GPU, similar to multi-Core
 - multi-node, e.g. for DASK