HyperQueue

Overcoming Limitations of HPC Job Managers

Böhm, Beránek, Cima, Macháček, Jha, Kočí, Jansík, Martinovič

IT4Innovations, VSB-TU Ostrava, Czech Republic

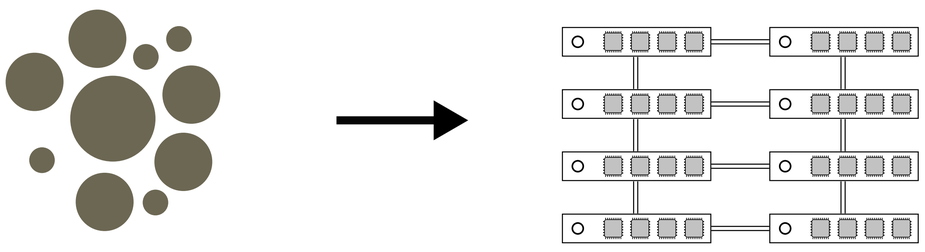
IT4Innovations/hyperqueue

https://def.it4i.io

HyperQueue lets you easily execute millions of embarrassingly parallel tasks on a Slurm/PBS cluster with dynamic load balancing and NUMA-aware scheduling

Motivation 1

Many simple tasks → **Slurm/PBS Cluster**



Execute a large number of simple, embarrassingly parallel tasks on an HPC cluster as easily as possible.

Challenges 2

Task duration x Job manager overhead
- Tasks can be very short (milliseconds)
- Workflows can contain a large number of tasks
- HPC job managers introduce a lot of overhead per each task

Challenge #1: **Keep overhead per task minimal**

Task core granularity x Node utilization
- Tasks use diverse (often small) amounts of threads/cores
- HPC nodes contain many (tens, hundreds) cores
- HPC job managers often allow only a single job per node

Challenge #2: **Achieve high node utilization**

Limitations of existing approaches 3

Submit tasks directly as Slurm/PBS jobs
- 👎 High overhead per task → limited task throughput
- 👎 Usually only a single job per node allowed → inefficient node usage
- 👎 Manual aggregation of tasks into jobs is required to reduce overhead

Run a meta-scheduler outside of Slurm/PBS
e.g. Snakemake
- 👍 More tasks may be submitted than what Slurm/PBS allows
- 👎 No dynamic task load balancing → inefficient node usage
- 👎 Manual aggregation of tasks into jobs is required to reduce overhead

Distribute tasks inside each Slurm/PBS job
e.g. Dask, GNU Parallel
- 👍 Load balancing across nodes inside a job
- 👎 No load balancing across jobs
- 👎 Manual aggregation of tasks into jobs is required to reduce overhead

Run both outside Slurm/PBS and inside its jobs
Merlin
- 👍 Automatically load balances tasks into jobs
- 👍 Load balances tasks inside jobs
- 👎 Queues with fixed concurrency → balancing not fully dynamic
- 👎 Does not allow specification of CPU task requirements
- 👎 Nontrivial to deploy

HyperQueue 4

Runtime designed for executing a large number of tasks.

- **Transparent task execution on top of a Slurm/PBS cluster**
  - Tasks are distributed amongst jobs, nodes and cores
  - Slurm/PBS jobs with HQ workers are started automatically
- **Dynamic load balancing across jobs**
  - NUMA-aware work-stealing scheduler
  - Supports core pinning, task priorities, task arrays
  - Nodes and tasks may be added/removed on the fly
- **Scalability**
  - Low overhead per task (~100 μs)
  - Handles hundreds of nodes and millions of tasks
  - *Output streaming* avoids creating many files on network filesystems
- **Easy deployment**
  - Single binary, no installation, depends only on *libc*
  - No elevated privileges required
- **Open source**

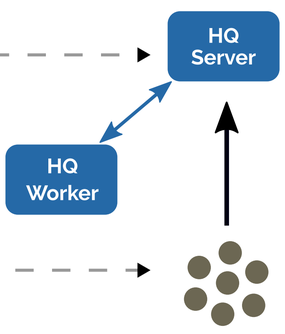
Usage example 5

```
# Start the HyperQueue server (on a login node)
$ hq server start

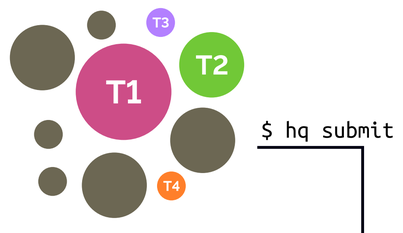
# Ask for computational resources
(PBS)   $ qsub <qsub-params> -- hq worker start
(Slurm) $ sbatch <sbatch-params> --wrap "hq worker start"

# Submit a HyperQueue job with 100 tasks
$ hq submit --cpus=4 --array=1-100 my-computation.sh

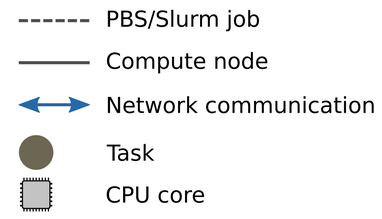
# Check the state of submitted jobs
$ hq jobs
+----+-------------------+----------+-------+
| Id | Name              | State    | Tasks |
+----+-------------------+----------+-------+
|  1 | my-computation.sh | FINISHED | 1     |
+----+-------------------+----------+-------+
```

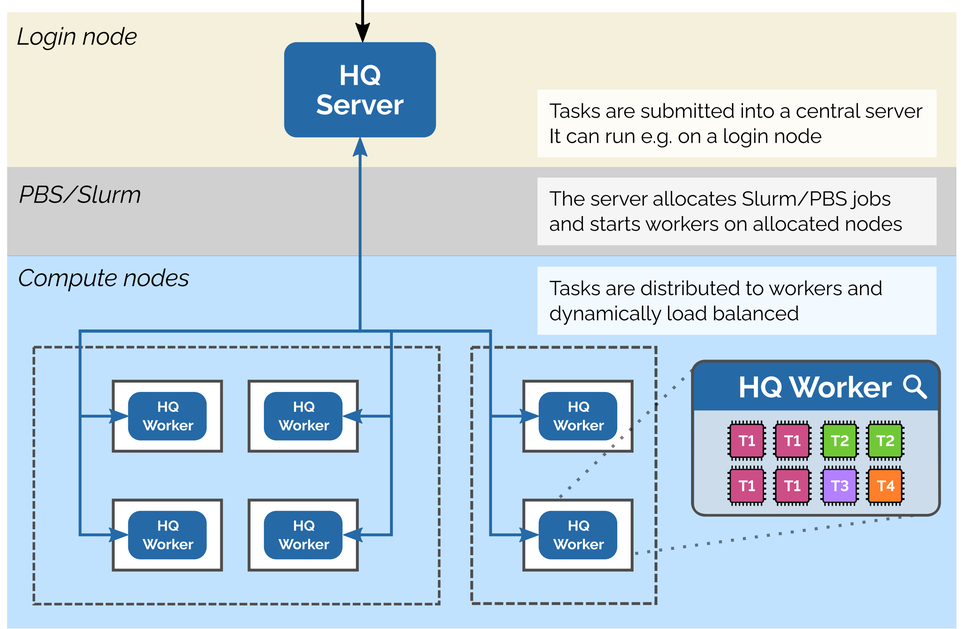


Architecture 6



$ hq submit





This work was supported by the LIGATE project. This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 956137. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Italy, Sweden, Austria, the Czech Republic, Switzerland.
This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90140).

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER