Cybermedia Center
Osaka University, Japan
SC22 BOOTH 1613

Toward a Practical Cloud Bursting Operation on SQUID

Background

The on-premise supercomputing systems in CMC are sometimes faced with a surge of the computing demand. This situation causes a longer wait time from when a user submits a job until when the job starts. In order to alleviate the peak, we have built SQUID as our new on-premise supercomputing system with the idea of offloading the workloads on an on-premise supercomputing system to cloud computing resources. This idea is referred to as *cloud bursting.*

Job
Offloading the workload
Cloud

Fig. 1 Overview of cloud bursting.

Environment

The cloud bursting environment on SQUID allows the users to execute their jobs without their being aware of Azure. The following two ideas achieve transparency in terms of usage.

- Cloud bursting queue which dynamically allocates their jobs to SQUID or Azure.
- CPU computing nodes on SQUID and Azure access Lustre on SQUID with NFS.

However, the cloud bursting environment discourages the users from executing their jobs on Azure. This reason is they have to be aware of Azure in terms of performance and monetary cost.

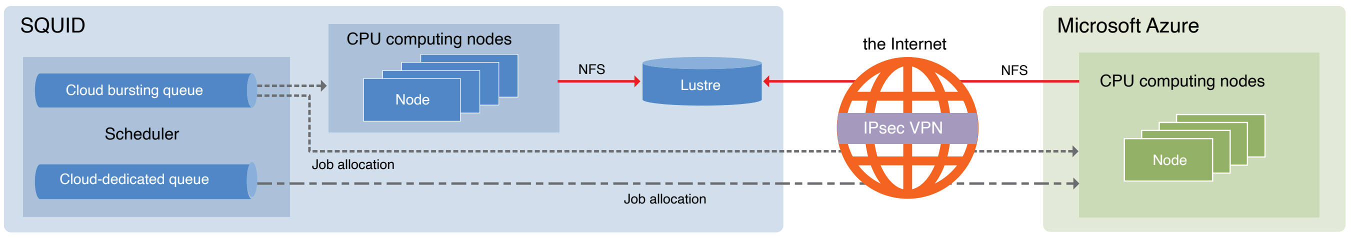

Fig. 2 The cloud bursting environment on SQUID.

Operation view

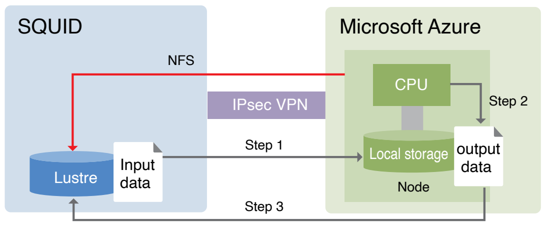

Fig. 3 The staging functionality.

Step 1	The users manually copy input data to a local storage before executing their jobs. (stage-in)
Step 2	The jobs access the input/output data on the local storage.
Step 3	The users manually copy the output data to Luster after executing the jobs. (stage-out)

- The two queues on SQUID
  1. Cloud bursting queue : Transparency in terms of usage, low cost for the users
  2. Cloud-dedicated queue : Immediate provision of computing resources, high cost for the users.

To improve transparency in terms of performance and cost, we propose three operations on SQUID which combine the staging functionality and the two queues.
Policy 1. Staging functionality unable + Cloud bursting queue
Policy 2. Staging functionality enable + Cloud-dedicated queue
Policy 3. Improved staging functionality enable + Cloud bursting queue (unfinished implementation)

Transparency	Usage	Performance	Cost
Policy 1	easy	low	low
Policy 2	difficult	high	high
Policy 3	easy	high	low

Tab. 1 The three operation policy.

Performance profile

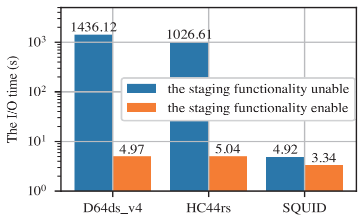

Fig. 4 The I/O time with the staging functionality unable and enable (BT-IO).

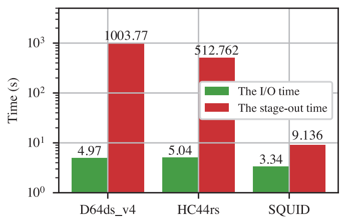

Fig. 5 The I/O time and the stage-out time with the staging functionality enable (BT-IO).

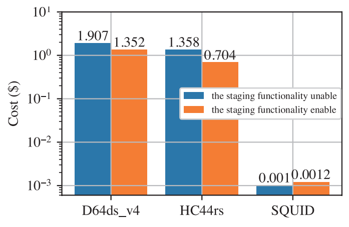

Fig. 6 The cost with the staging functionality unable and enable (BT-IO).

Contact : sc22@ais.cmc.osaka-u.ac.jp https://www.cmc.osaka-u.ac.jp/