About Us：Cybermedia Center, Osaka University

As a resource provider of knowledge and technology derived from advanced researches conducted in Osaka University, the Cybermedia Center (CMC) offers support in the areas of large-scale computation, information communication, multimedia content and education. The center also works closely with educational and research organizations within Osaka University, as well as with industries and institutes outside the University. By sharing its resources and encouraging local communities to use its facilities for public lectures and other events, CMC has helped to create a more internationally-oriented IT society for the region.

Location Map

Location

University-Wide Services

Large-Scale Computer System, we provide a high-performance computing environment, consisting of OCTOPUS and SQUID, to both the academic and industrial communities. Part of the overall computer system is provided, as a computational resource, to the national High-Performance Computing Infrastructure(HPCI).

Information Media Education Multimedia Language Education, we have implemented a consistent curriculum, from the basics of computer utilization to advanced subject matter, while the Computer Assisted Language Learning System supports foreign language learning and cross-cultural understanding in accordance with each individual's language-proficiency level.

Cybermedia Commons is an active learning space for students, exploiting a wide variety of the Cybermedia Center's information technology, to support student's active learning and research activities.

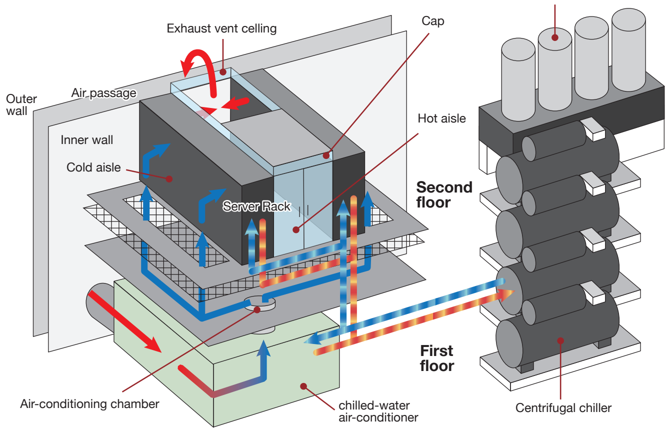
Cybermedia Commons

Digital Library provides academic information databases and remote access to electronic journals. It is equipped with multimedia terminals and public network jacks with an authentication system.

Repair and Maintenance of the Information Network, a high-speed, stable and reliable campus-wide network environment, as well as wireless access networks, as information infrastructure for supporting the educational, research, and social contribution activities of Osaka University.

Academic Cloud improves the integration of computing resources scattered across the university. The objectives of the system are to optimize administrative operations, enhance security, and reduce costs.

IT Core Annex is a two-story steel-frame data center housing large-scale computers. The perimeter wall is designed with gently curving surface and light-permeable metal panels, to harmonize with the surrounding environment.



Cooling mechanism in IT Core Annex

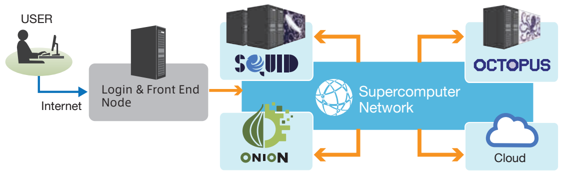
Cybermedia Center

Osaka University, Japan

SC22 BOOTH 1613

Large-scale Computing Systems at the Cybermedia Center

Overview of High-Performance Computing Environment at the CMC



Large-scale computing systems (OCTOPUS and SQUID) and data aggregate infrastructure (ONION) are deployed on CMC-Supercomputer network, a.k.a CMC-SCinet, a low-latency and wide-bandwidth network. This architectural design allows users to access to large-scale storage systems, perform large-scale high-performance computation and analysis on our large-scale computing systems.

Large-scale Computing System

OCTOPUS

OCTOPUS is short for Osaka university Cybermedia cenTer Over-Petascale Universal Supercomputer. OCTOPUS is a cluster system being operated since December 2017. This system is composed of General purpose CPU nodes, GPU nodes, Large-scale shared-memory nodes, and Xeon Phi nodes, total 319 nodes. These nodes and large-scale storage EXAScaler (Lustre 3.1 PB) are interconnected on InfiniBand EDR (100 Gbps) and form a cluster.

Table 1 Data Sheet of OCTOPUS

Type of nodes	General purpose CPU	GPU	Large–scale shared-memory	Xeon Phi
CPU	Intel Xeon Skylake (2.6 GHz, 12 cores) x 2		Intel Xeon Skylake (2.0 GHz, 16 cores) x 8	Intel Xeon Phi KNL (1.3 GHz, 64 cores)
OS	RHEL 7.3			
# of nodes (total)	236	37	2	44
# of cores (total)	5,664	888	256	2,816
# of memory (total)	45	7	12	8
Peak performance	471.2 TFLOPS	16.4 TFLOPS	858.3 TFLOPS	117.1 TFLOPS
Accelerator		NVIDIA Tesla P100 x 148		

SQUID

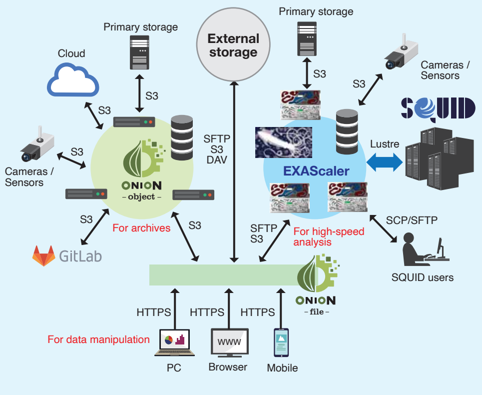
SQUID is short for Supercomputer for Quest to Unsolved Interdisciplinary Datascience. SQUID is a new cluster system being operated since May 2021. This system is composed of General purpose CPU nodes, GPU nodes, and Vector nodes, total 1,598 nodes. These nodes and large-scale storage EXAScaler (Lustre 21.2 PB) are interconnected on InfiniBand HDR (200 Gbps) and form a cluster.

Table 2 Data Sheet of SQUID

Type of nodes	General purpose CPU	GPU	Vector
CPU	Intel Xeon Icelake (2.4 GHz, 38 cores) x 2		AMD EPYC Rome (2.8 GHz, 24 cores)
OS	CentOS 8.4		
# of nodes (total)	1,520	42	36
# of cores (total)	115,520	3,192	864
# of memory (total)	389 TB	22 TB	5 TB
Peak performance	8.871 PFLOPS	6.797 PFLOPS	0.922 PFLOPS
Accelerator		NVIDIA A100 x 336	NEC SX-Aurora TSUBASA Type20A x 288

Data Aggregation Infrastructure

ONION



ONION stands for Osaka university Next-generation Infrastructure for Open research and open innovatioN. ONION is a new data aggregation infrastructure that is linked to SQUID. ONION consists of ONION-object (AWS S3 compatible object storage) , ONION-file (storage service using Nextcloud) , and EXAScaler (a parallel file system based on Lustre) .

ONION makes it easy for users to data between your PC and large-scale computing system. In addition, ONION can be used in a variety of ways, such as immediate sharing of calculation results with those who do not have a SQUID or OCTOPUS account and manipulating data from a smartphone. Of course, it can also be used to store and share research data in the laboratory.

Table 3 EXAScaler (on SQUID)

Effective capacity (HDD)	20 PB
Effective capacity (NVMe)	1.2 PB
Max number of inodes	Approx. 8.8 Billion
Max expected effective throughput (HDD)	Over 160 GB/s
Max expected effective throughput (NVMe)	Write : Over 160 GB/s Read : Over 180 GB/s

Table 4 ONION-object

Effective capacity	950 TiB * We plan to expense sequentially
Data protection method	Erasure Coding (Data chunk:4 + Parity chunk:2)

Contact : sc22@ais.cmc.osaka-u.ac.jp http://www.hpc.cmc.osaka-u.ac.jp/en/

Cybermedia Center

Osaka University, Japan

SC22 BOOTH 1613

AI assisted job scheduler / Profile guided vector optimization

AI assisted job scheduler: Cloud Burst Optimization with Deep Q Network

Background

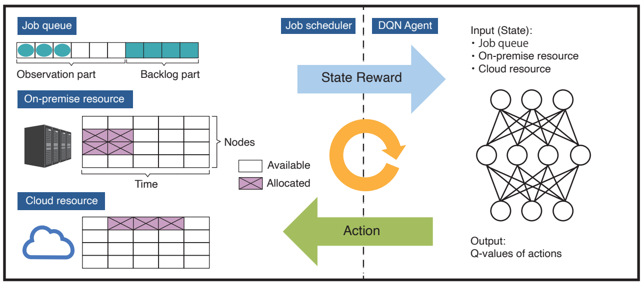
- Cloud bursting becomes attractive for HPC systems to prevent an increase of job waiting time under high load.
- However, it is still difficult to control the tradeoff between job waiting time and cloud cost.

Proposal

- Job scheduler with DQN (Deep Q Network) that can optimize the tradeoff of cloud bursting

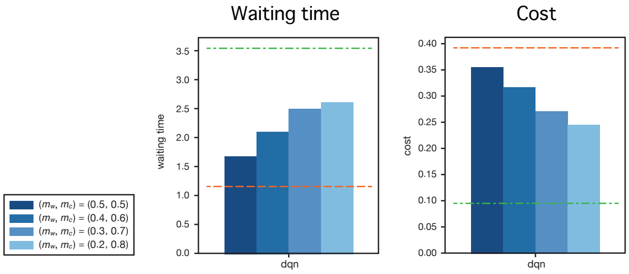
Architecture

- Job scheduler provides state of job queue and on-premise and cloud resources to DQN when scheduling a job
- DQN returns action that shows the scheduler should assign on-premise or cloud resources to the job or skip scheduling
- Job scheduler schedules the job based on the action
- Job scheduler provides reward to DQN for evaluating the action based on a waiting time and a cloud cost



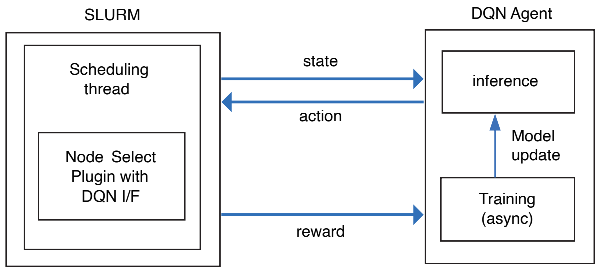
Results

- Proposed architecture can control tradeoff between waiting time and cloud costs



Future Work

- Evaluation and implementation of the proposed method into the SLURM scheduler



Profile guided vector optimization for SX-Aurora TSUBASA

Background

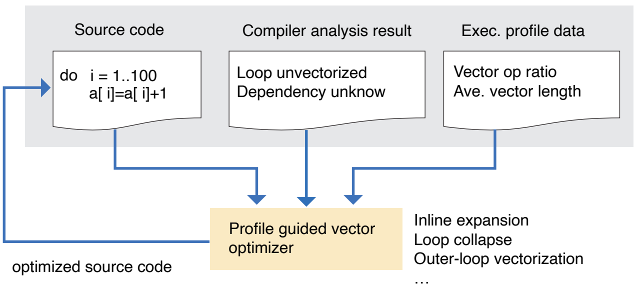
- Realization of vector optimization by users without HW knowledge

Proposal

- Automatic source-to-source translation tool by Profile Guided Vector Optimization (PGVO) for SX-Aurora TSUBASA

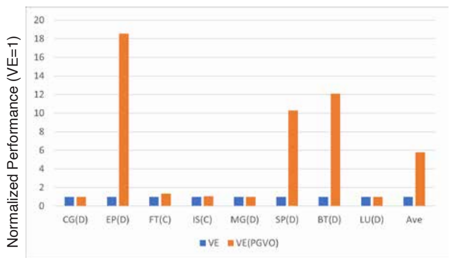
Architecture

- PGVO uses source codes, compiler analysis results and execution profile data as inputs
- PGVO outputs translated source codes



Results

- Significant performance improvement by PGVO compared to automatic vectorization compiler with 3/8 workloads* of NPB
(* Human-optimized codes that assumes tool behavior are evaluated)



- NPB 3.4.2 (omp version)
- VE: SX-Aurora 20B 1VE(8core) + ncc/nfort

- ☐ EP/SP/BT: PGO achieves great improvement
- ☐ CG/MG: Compiler already achieves good performance. No room for PGVO.
- ☐ FT/IS/LU: Some room for optimization, but current PGVO achieves little or no improvement.

Future Work

- Implement as a tool and confirm the feasibility
- Evaluate with more workloads
- Support more vectorization technologies

These works were carried out in Joint Research Laboratory for Integrated Infrastructure of High Performance Computing and Data Analysis
https://www.nri.cmc.osaka-u.ac.jp/

Cybermedia Center

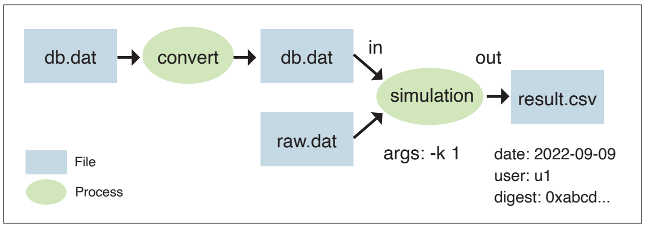
Osaka University, Japan

SC22 BOOTH 1613

Provenance Recording System for Research Data Management

Background

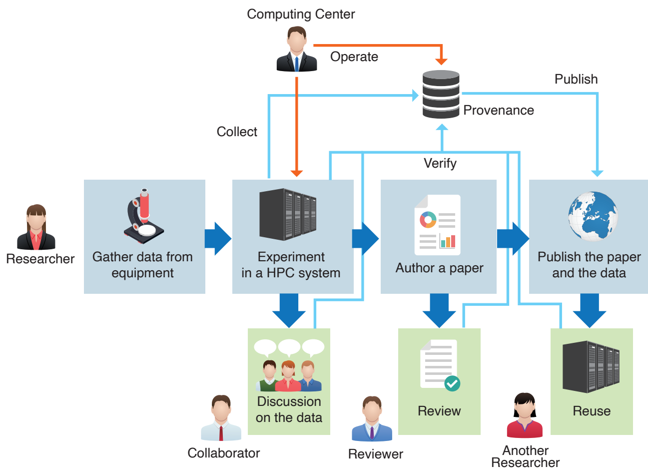
- Growing importance of research data management (RDM)
  - To ensure **reproducibility** (transparency): Preserving data that provide evidence of research results
  - To improve **reusability**: Promoting the global sharing of knowledge and increasing research efficiency
- Provenance, which identifies the input data and the process used to obtain data, should be secured for reproducibility and reusability
- HPC systems generate data through simulations and experiments, but there is no established method to manage the provenance of the data
- A system that implements RDM on HPC systems is needed



Example of a provenance

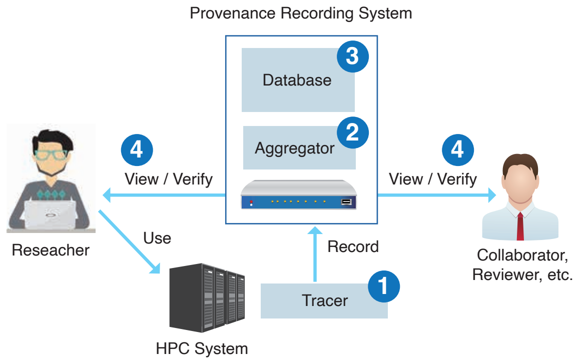
Requirements for Provenance Recording System

- Automatically record the provenance and the metadata (date/user created, etc.) of a file generated in a HPC system
- Support a typical HPC environment: workload manager (Slurm), MPI, etc.
- Minimalize impacts on performance and user's operations
- Secure the records not to be falsified
- Provide interfaces to verify that a file has not been fabricated/falsified

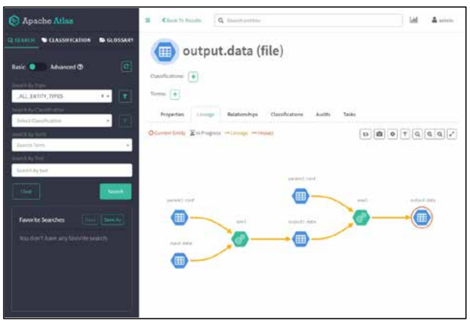


Use cases

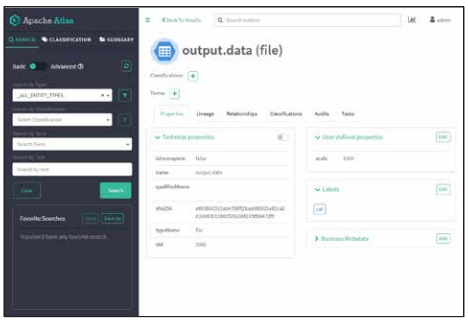
Prototype



1. Tracer captures system call invocations (exec(), open(), write(), etc.) of a user program. BPF, a low overhead observability scheme in Linux kernel is used for the capture. Tracer also captures metadata (date created, SHA-256, etc.). No modifications in the user program and operations are required.
2. Aggregator builds a provenance of files from the history of the system call invocation: a file read/written by a process is an input/output of the process in the provenance. Parallel processes by MPI are aggregated.
3. The provenance and the metadata are stored in Apache Atlas (an open-source data catalog).
4. Find and verify the provenance and the metadata of a file (shown below).



Show the provenance of a file



Show the metadata of a file

This work was carried out in Joint Research Laboratory for Integrated Infrastructure of High Performance Computing and Data Analysis
https://www.nri.cmc.osaka-u.ac.jp/

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

Cybermedia Center
Osaka University

Cybermedia Center
Osaka University, Japan
SC22 BOOTH 1613

ns-3-based Interconnect Simulator for Network Simulation with Job Scheduling

Background : Aim of Interconnect Design in Supercomputing Systems is Changing

A variety of jobs are performed on today's supercomputing systems. The number of compute nodes requested by such jobs is diverse and then much inter-node communication take place.
⇨ Interconnects of supercomputing systems should be **designed using simulators to examine the performance in communication.**

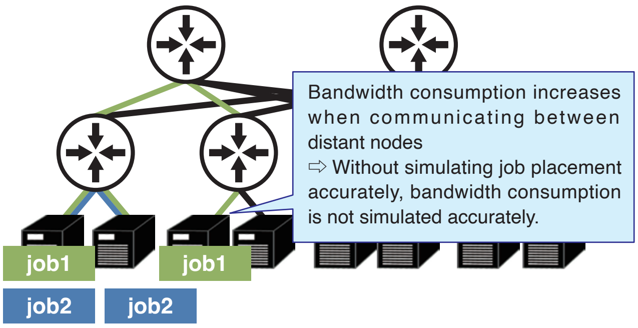
Traditional Supercomputing Systems	
Expected Workload	Computation-intensive MPI (Message Passing Interface) jobs.
Aim of Interconnect Design	Focus on the cost to increase the number of compute nodes.
Method of Interconnect Design	・Select from stable and mature technologies such as Fat-tree and ECMP. ・Parameters are determined by calculations and other simple estimates.

Next Supercomputing System	
Expected Workload	Communication-intensive jobs using distributed processing frameworks.
Aim of Interconnect Design	Focus on the performance to accelerate inter-node communication.
Method of Interconnect Design	・Select from stable and mature technologies and/or state-of-the-art technologies such as DragonFly and adaptive routing. ・Parameters are determined by simulations to examine interconnect performance.

Problem: The Effects of Job Scheduling Are Missed by Existing Network Simulators

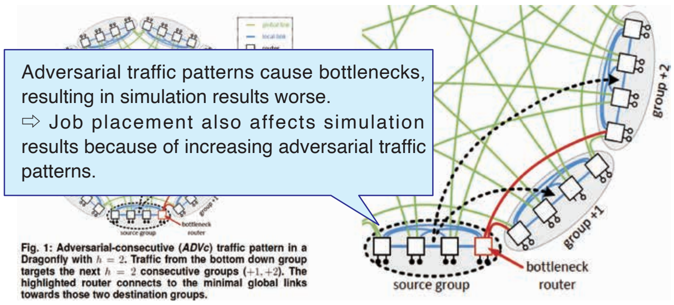
When simulating interconnects in a supercomputing system, the simulation result is incorrect in the case of using only existing network simulators. The reason is **existing network simulators cannot reproduce job placement by job schedulers.**

・**Traffic patterns* are changed by job placement.**
(*A set of communications within a certain time period)



Example of link capacity consumption depending on job placement

・**Adversarial traffic patterns**** cause misunderstanding of the network performance. (**Traffic patterns that degrade network performance)



Adversarial traffic patterns in DragonFly topology [1]

[1] P. Fuentes, E. Vallejo, C. Camarero, R. Beivide and M. Valero, "Throughput Unfairness in Dragonfly Networks under Realistic Traffic Patterns," 2015 IEEE International Conference on Cluster Computing, 2015, pp. 801-808, doi: 10.1109/CLUSTER.2015.136.

Proposal : ns-3-based Interconnect Simulator for Interconnect Design (In-Progress)

To achieve network simulation with job scheduling, we decided to implement a job scheduling function as a module for ns-3.

・**Assets**: Interconnect research results are implemented in ns-3, **enabling simulations using state-of-the-art technologies.**
・**Expandability**: ns-3 is modularized, making it **easy to expand the job scheduling** functionalities.
・**Packet-level simulation**: accurate simulation of network latency should **reduce performance estimation errors.**

Parameter file

・Topology
・Routing algorithm
・TCP congestion control algorithm
・**Job scheduling algorithm**

Job scripts

・Number of request nodes
・**Contents of communication** (prepared communication pattern or pcap file) and **computation time** (communication interval)

User Input

Step 0	Setup topology as input files
Step 1	According to the job scheduling algorithm, the job scheduler **dynamically places traffic generators using the ns3 Schedule function**
Step 2	Present simulation results including job runtime and bottleneck links/switches

By repeating Step 0 ~ 2 with different inputs,
interconnect design based on quantitative comparisons will be achieved.

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

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*.

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.

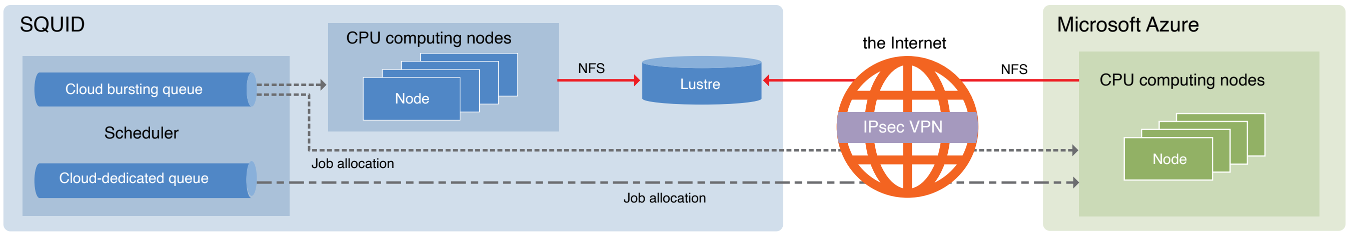


Fig. 2 The cloud bursting environment on SQUID.

Operation view

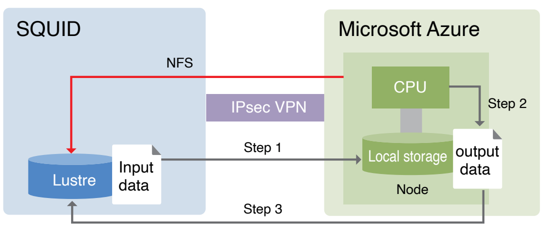


Fig. 3 The staging functionality.

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

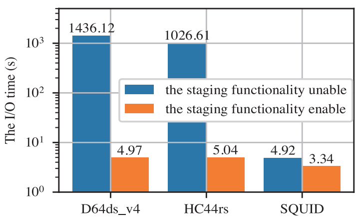


Fig. 4 The I/O time with the staging functionality unable and enable (BT-IO).

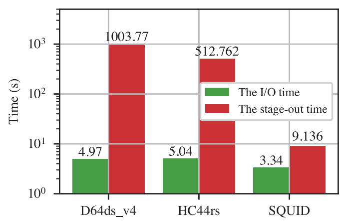


Fig. 5 The I/O time and the stage-out time with the staging functionality enable (BT-IO).

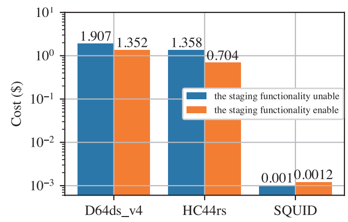


Fig. 6 The cost with the staging functionality unable and enable (BT-IO).

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