富岳の運用状況報告
Operation status of Fugaku

2024/08/07

Operation and Computer Technology Division,
RIKEN Center for Computational Science

UPDATE

Operation Schedule of Fugaku

- Calculated at 384 nodes / a rack

August	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
	Thu.	Fri.	Sat.	Sun.	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.
Available ratio (# of Racks)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)
	<= medium scale jobs =>																			<= large scale jobs => <= medium scale jobs =>										

September	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	
	Sun.	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.	Sun.	Mon.	
Available ratio (# of Racks)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	100% (414)	
	<= medium scale jobs =>													<= large scale jobs = > <= medium scale jobs =>																

(Date : JST)

- Large scale job execution period (tentative)
  - About 50% (216racks/82,944 nodes) of full nodes are for this, 50% are for shared use
    - 08/20 (Tue.) 15:00 - 08/23 (Fri.) 15:00
    - 09/17 (Tue.) 17:00 - 09/20 (Fri.) 17:00

- Medium scale job execution period (tentative)
  - Target jobs will be executed with high priority among other large jobs after large scale jobs period finish
    - 08/05 (Mon.) 15:00 - 08/09 (Fri.) 15:00 (The start time may be earlier or later)
    - 08/23 (Fri.) 15:00 - 08/27 (Tue.) 15:00 (This will start at the end of the half size of full nodes execution)
    - 09/04 (Wed.) 15:00 - 09/08 (Sun.) 15:00 (The start time may be earlier or later)
    - 09/20 (Fri.) 15:00 - 09/24 (Tue.) 15:00 (This will start at the end of the half size of full nodes execution)

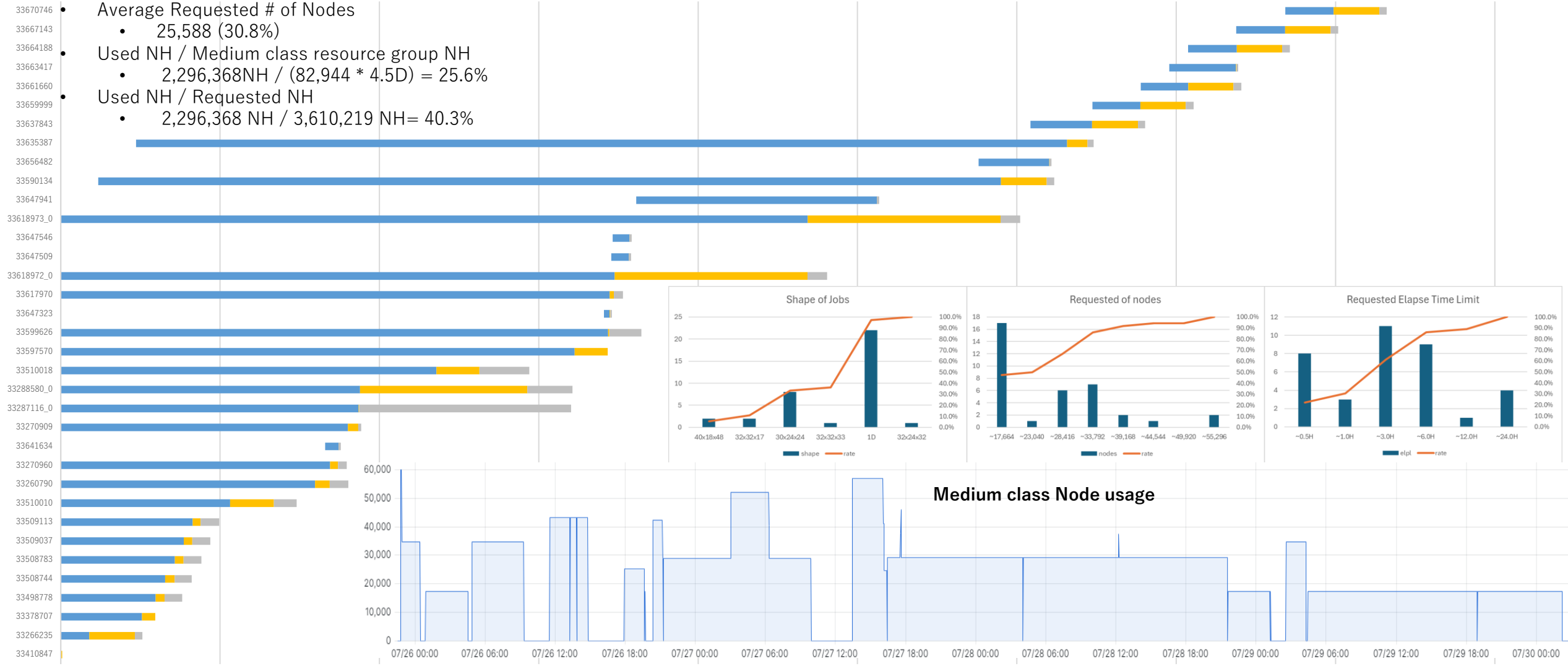
- Login nodes and Fugaku Website are available during the unavailable period of Compute nodes except the maintenance of login nodes and Fugaku Website.
- # of available compute nodes may change depending on the situation of an operation.
- Some services (login nodes, Fugaku Website, commands (pjsub, pjstat and etc)) may be temporarily unavailable due to the system trouble.

Scheduling status of medium class jobs execution in July.

middleジョブ実行状況

UPDATE

- # of executed jobs (ELAPSE_TIM > 0)
  - 36
- Average Requested # of Nodes
  - 25,588 (30.8%)
- Used NH / Medium class resource group NH
  - 2,296,368NH / (82,944 * 4.5D) = 25.6%
- Used NH / Requested NH
  - 2,296,368 NH / 3,610,219 NH= 40.3%



Resource group configuration for Shared Use (1/3)

- for all users

resource group name	# of nodes (min. / max.)	Shape*1 (1/2/3D)	# of concurrency	Max. Time Limit (hour)	Node allocation method	Node allocation unit	Remarks
large	385/12,288	1:12,288 2:144x720 3:48x51x48	unlimited	24	torus*2	48 nodes(2*3*8)	
small	1/384	1:384 2:138x288 3:46x18x48	unlimited	72	torus*2 mesh noncont(default)	・torus:12 nodes(2*3*2) ・mesh:1 node(greater than or equal to the require nodes) ・noncont:1 node(equal to the require nodes)	
small-s1	1/384	1:384 2:138x1104 3:46x69x48	unlimited	1	torus*2	12 nodes(2*3*2)	Jobs are moved from small by system
small-s2	1/384	1:384 2:138x384 3:46x24x48	unlimited	72 (more than 1h)	torus*2	12 nodes(2*3*2)	
small-s3	1/1	-	unlimited	72	torus*2 mesh noncont(default)	1 node	
small-s5	1/1	-	unlimited	1	torus*2 mesh noncont(default)	1 node	
small-torus	1/384	1:384 2:24x32 3:8x6x16	unlimited	24	torus*2	12 nodes(2*3*2)	
int	1/12	1:12 2:138x288 3:46x18x48	unlimited	6	torus*2 mesh noncont(default)	・torus:12 nodes(2*3*2) ・mesh:1 node(greater than or equal to the require nodes) ・noncont:1 node(equal to the require nodes)	・Interactive job only. ・# of concurrency submit limit: 3 Jobs/User
middle	12,289/55,296	1:55,296 2:144x576 3:48x36x48	unlimited	24	torus*2	48 nodes(2*3*8)	Jobs are executed during designated period.
huge	55,297/82,944	1:82,944 2:144x576 3:48x36x48	unlimited	24	torus*2	48 nodes(2*3*8)	・You need to pass a preliminary examination. ・Jobs are executed during designated period.

*1 In the 2 and 3 dimension‘s description, it shows the maximum value of each coordinate.

*2 Allocated number of nodes may be more than required number of nodes.

Resource group configuration for Shared Use (2/3)

- for users to cooperate in energy conservation (Fugaku points) in FY2024

resource group name	# of nodes (min. / max.)	Shape*1 (1/2/3D)	# of concurrency	Max. Time Limit (hour)	Node allocation method	Node allocation unit	Remarks
f-pt	1/12,288	1:12,288 2:144x816 3:48x51x48	unlimited	24	torus*2	12 nodes(2*3*2)	**Fugaku Points in FY2024** You can execute the priority job up to the calculation resource amount according to the Fugaku point.

- Low priority jobs(spot) in FY2024

resource group name	# of nodes (min. / max.)	Shape*1 (1/2/3D)	# of concurrency	Max. Time Limit (hour)	Node allocation method	Node allocation unit	Remarks
spot-large	385/12,288	1:12,288 2:144x816 3:48x51x48	unlimited	12	torus*2	48 nodes(2*3*8)	Low priority
spot-small	1/384	1:384 2:138x288 3:46x18x48	unlimited	12	torus*2 mesh noncont(default)	・torus:12 nodes(2*3*2) ・mesh:1 node(greater than or equal to the require nodes) ・noncont:1 node(equal to the require nodes)	Low priority
spot-int	1/12	1:12 2:138x288 3:46x18x48	unlimited	6	torus*2 mesh noncont(default)	・torus:12 nodes(2*3*2) ・mesh:1 node(greater than or equal to the require nodes) ・noncont:1 node(equal to the require nodes)	・Interactive job only. ・# of concurrency submit limit: 3 Jobs/User ・Low priority
spot-middle	12,289/55,296	1:55,296 2:144x576 3:48x36x48	unlimited	12	torus*2	48 nodes(2*3*8)	・Jobs are executed during designated period. ・Low priority

- for users in General/Industrial Use Projects (Proprietary Use)

resource group name	# of nodes (min. / max.)	Shape*1 (1/2/3D)	# of concurrency	Max. Time Limit (hour)	Node allocation method	Node allocation unit	Remarks
prior	1/12,288	1:12,288 2:144x816 3:48x51x48	unlimited	24	torus*2	48 nodes(2*3*8)	Higher priority
extra	1/384	1:384 2:138x288 3:46x18x48	unlimited	72	torus*2 mesh noncont(default)	・torus:12 nodes(2*3*2) ・mesh:1 node(greater than or equal to the require nodes) ・noncont:1 node(equal to the require nodes)	・Higher priority ・Resources are consumed at a higher rate than usual

- for users in Government Initiated Projects in FY2024

resource group name	# of nodes (min. / max.)	Shape*1 (1/2/3D)	# of concurrency	Max. Time Limit (hour)	Node allocation method	Node allocation unit	Remarks
ppu2024	1/12,288	1:12,288 2:144x816 3:48x51x48	unlimited	24	torus*2	48 nodes(2*3*8)	Highest priority

*1 In the 2 and 3 dimension‘s description, it shows the maximum value of each coordinate.

*2 Allocated number of nodes may be more than required number of nodes.

Resource group configuration for Shared Use (3/3)

Remarks of job submission

- Job acceptance is limited
  - 1,000 jobs except step and bulk sub jobs / group (Changed from 10,000 jobs)
  - 50,000 step sub jobs / group
  - 2,000 bulk sub jobs / group
  - 40,000 jobs except step and bulk sub jobs / system
  - 42,000 bulk sub jobs / system

- “spot-*” are resource groups for low priority jobs.
  - Low priority jobs are executed when compute nodes are free.
  - If there are some normal priority jobs, low priority jobs are not executed regardless of when you submit your low priority jobs.

- You can check size of each resource group (RSCGRP_SIZE) by executing pjstat command with --rsc option.

ex.
login$pjstat --rsc

RSCUNIT	RSCUNIT_SIZE	RSCGRP	RSCGRP_SIZE
rscunit_ft01[ENABLE,START]	24x23x24	large[ENABLE,START]	48x51x48
rscunit_ft01[ENABLE,START]	24x23x24	small[ENABLE,START]	41424

login$

note: Some resource group such as small shows only the number of nodes.

If you'd like to check more information, please refer to 2.2.1 on the Job Operation Software End-user's Guide

Manuals/Language environment

UPDATE

- Users guides
  - Users Guide - 利用およびジョブ実行編 / Use and job execution **Ver. 1.41**
  - Users Guide – 言語開発環境編 / Language and development environment Ver. 1.28.1

- Language environment
  - Ver. 4.11.1 tcsds-1.2.39 (default)
  - Ver. 4.10.0 tcsds-1.2.38
  - Ver. 4.9.0 tcsds-1.2.37

- Intel oneAPI Base & HPC Toolkit is available on Intel login nodes
  - Current version 2024.1.0
  - Module file is available
    - module load [Module file name corresponding to the component]

- Arm compiler ver. 23.10
  - Module file is available
    - module load [Module file name corresponding to the component]

障害情報

現象	原因/回避策	修正予定
同サイズのラージページがノードにより獲得できる場合とできない場合がある	【原因】BIOノードでLinuxが予約しているメモリ領域が存在することが判明 【回避】メモリ領域を強制開放する仮対処を設定中 ※OSの修正パッチ適用→経過観察中(8/25-) ※10月にラージページ用メモリ領域の設定を変更しましたが、ノーマルページを利用するジョブに影響が出たため、11月より設定を戻しました (11/3-)	経過観察中
電力のMAX値が想定した値にならない	【原因】 調査中	未定
clangモードにおいて、入れ子クラスに対するコンストラクタのチェックで、結果異常となる場合がある	【原因】コンパイラ障害 【回避】回避方法はありません。	未定
clangモードにおいて、std::regexのコンストラクタで複数文字を表現する正規表現を含む不正な範囲の正規表現を指定すると、結果異常となる場合がある	【原因】コンパイラ障害 【回避】回避方法はありません。	未定
テンプレート関数を宣言したC++プログラムをtradモードのc++17仕様で翻訳すると不正なエラーが出力される場合がある	【原因】コンパイラ障害 【回避】翻訳時オプション-std=c++14、または、-std=gnu++14を指定する。	未定
ジョブが実行直前にエラーとなり実行されない（PJM CODE=180）	【原因】システム内部のキャッシュの肥大化 【回避】回避方法はありません。ジョブの再投入をお願いします。	未定
スパースファイルの操作で、ファイルシステムへのアクセスがハングする場合がある	【原因】ファイルシステム側のioctl(2) の処理誤り 【回避】回避方法はありません。ジョブがアクセス中のスパースファイルに対し、異なるジョブやログインノードからファイルを操作はお控えください.	2024年10月

Failure information

Matters	Cause / workaround	to be revised
Some nodes may not get requested memory size from the large page area.	[cause] OS reserved some memory area on BIO nodes [workaround] temporary measures that release the memory area forcibly is applied. ※ Applied OS fix patch→Under observing (8/25-)	Under observing
Maximum value of power is not expected value.	[cause] under investigating	TBD
The execution result may be incorrect when checking the constructor for the nested class in clang mode.	[cause] compiler failure [workaround] No preventive measures.	TBD
The execution result may be incorrect when specify an invalid range of regular expressions including regular expressions that express multiple characters in the constructor of std::regex in clang mode.	[cause] compiler failure [workaround] No preventive measures.	TBD
Compiling to the c++17 specification in trad mode a C++ program that declares a template function may output an invalid error.	[cause] compiler failure [workaround] Specify compiler option -std=c++14 or -std=gnu++14.	TBD
Job fails immediately before RUN (PJM CODE=180)	[cause] compiler failure [workaround] No preventive measures. Resubmit your job.	TBD
Handling a sparse file may hang via FEFS or second-layer storage cache.	[cause] File system ioctl (2) handling failure [workaround] This failure may occur when a job is accessing a sparse file by handling the file from a different job or login node.	October 2024

ログインノード障害 （ログインノード#5, #6）

新規

以下の時間帯におきまして、ログインノードで障害が発生しておりました。
現在は復旧しております。
ご迷惑をおかけし申し訳ありません。

Unable to log in to login node (login node #5, #6)

New

The login node failed during the following time period.
It is now recoverd. We apologize for the inconvenience.

Period: 2024/07/17 14:27 - 2024/07/17 15:00 (JST)
Node: login node #5

Period: 2024/07/19 00:33 - 2024/07/19 01:31 (JST)
Node: login node #6

プリポスト環境の障害（大容量メモリノード#1/ppm01）

新規

以下の時間帯におきまして、プリポスト環境（大容量メモリノード#1/ppm01）で障害が発生しておりました。
現在は復旧しております。
ご迷惑をおかけし申し訳ありません。

Pre/Post Environment failure (Large memory node #1/ppm01)

New

Large memory node #1 in the Pre/Post environment went down due to failures at the following times.
It is now recoverd. We apologize for the inconvenience.

Period: 2024/07/13 16:00 - 2024/07/13 18:37 (JST)
Period: 2024/07/22 14:50 - 2024/07/22 15:28 (JST)

OAuth認証によるログインサービスの提供について(2024/07/01～)

新規

HPCIで富岳資源をご利用の皆様

HPCIでは、2024年07月よりアクセストークンに基づくOAuth認証でのログインノードおよびCSGWノードへのログインの提供を開始しております。

OAuth認証のご利用方法は、次のマニュアルへ記載しております。
・マニュアル名: HPCIログインマニュアル OAuth対応版
・URL: https://www.hpci-office.jp/for_users/hpci_info_manuals

また、OAuth認証を用いて富岳ログインノードおよびCSGWノードにログインする際は以下の通り個別のホスト名を指定するようお願いいたします。

$ hpcissh login1.fugaku.r-ccs.riken.jp
$ hpcissh csgw1.fugaku.r-ccs.riken.jp

また、現在ご利用いただいておりますGSI認証(GSISSH)については、2024年度をもって提供を修了いたしますので、HPCIによるシングルサインオンを利用したSSHログインをご利用頂く場合はOAuth認証にご移行いただくようお願い申し上げます。

Provision of Login Services via OAuth Authentication (From: 2024/07/01)

New

Dear Fugaku System Users with used HPCI

Starting from July 2024, HPCI has commenced the provision of log-in access to Fugaku Login nodes and CSGW nodes using OAuth authentication based on access tokens.

For details on how to use OAuth authentication, please refer to the following manual:

* User's Guide HPCI Login Manual For OAuth
* URL: https://www.hpci-office.jp/en/for_users/hpci_info_manuals

Furthermore, when log-in into Fugaku Login nodes and CSGW nodes using OAuth authentication, please specify the individual host names as follows:

```
$ hpcissh login1.fugaku.r-ccs.riken.jp
$ hpcissh csgw1.fugaku.r-ccs.riken.jp
```

Additionally, please note that the GSI authentication (GSISSH) that you are currently using will be discontinued by the end of the FY2024. Therefore, if you wish to continue using SSH login with single sign-on provided by HPCI, we kindly request that you transition to OAuth authentication.

OpenFOAM（OpenCFD版）の利用に関する注意事項

新規

OpenFOAM-v2206以降のバージョンで、解析開始までの時間がかかる現象が確認されました。具体的には以下のような現象となります。

・時間が経っても標準出力がなかなか出力されない
・標準出力のExecutionTime 、ClockTimeの差が大きい

第二階層ストレージのキャッシュミスを削減するため、次のオプションを指定して実行することにより、本現象を回避できます。

```
#PJM --llio cn-read-cache=off
```

Note on the use of OpenFOAM（OpenCFD）

New

It was confirmed that it takes some time after OpenFOAM-v2206 version, before start of the time step iteration.
・Standard outputs take a long time to output.
・The difference between ExecutionTime and ClockTime in the standard output is large.

The following option is available as workaround to reduce 2nd layer cache miss hit.

```
#PJM --llio cn-read-cache=off
```

LLIO利用制限超過検知機能の強化について

新規

LLIO利用制限超過検知機能を強化いたしました。
今回の強化により、従来ではLLIO利用制限超過の検知ができなかったジョブも検知できるようになります。
ただし、LLIO利用制限を超過したファイルパスを通知できない場合があります。

ファイルパスを通知できない場合、"job_events --llio"コマンドでは以下のように表示されます。

```
login $ job_events --llio
JOBID     FILEPATH
123456802 The path could not be found.
```

ファイルパスを通知できない場合、Zendesk経由での通知は、以下のような通知になります。

```
Detected path:
Sorry.  The path could not be found.
```

LLIO利用制限超過が発生した場合は「利用手引書 利用およびジョブ実行編」の「階層化ストレージ>留意事項」を参照の上、ご対応お願いいたします。

"job_events --llio"コマンドによるLLIO利用制限超過のファイルパス表示については以下の記事をご参照ください。
　運用情報 LLIO利用制限超過したファイルパス表示コマンドのご提供

job_eventsコマンドに関しては、利用手引書並びに以下のFAQ記事をご参照ください。
　FAQ job_eventsコマンドについて

Enhancement of LLIO Usage Limit Exceeded Detection Function.

New

We have enhanced the LLIO usage limit excess detection feature.
With this enhancement, jobs that were not previously detected as exceeding the LLIO usage limit will now be detected.
However, the file path that exceeded the LLIO usage limit may not be notified.

If the file path cannot be notified, the "job_events --llio" command will show the following:

```
login $ job_events --llio
JOBID     FILEPATH
123456802 The path could not be found.
```

If the file path cannot be notified, the notification via Zendesk will be as follows:

```
Detected path:
Sorry.  The path could not be found.
```

If the LLIO usage limit is exceeded, please refer to "Layered storage>Important Notices" in the "Users Guide - Use and job execution -" and take appropriate action.

For information on displaying the file path that exceeded the LLIO usage limit using the "job_events --llio" command, please refer to the following article:
Operation Command to display file paths that exceed LLIO usage limit

For the job_events command, please refer to the User Guide and the following FAQ article:
FAQ About the job_events Command

省エネルギー実行へのご協力のお願い

新規

ジョブ実行時にコアリテンションの設定が可能となるジョブサイズを9,216ノードまで拡大し、9,216ノード以下のジョブには、ジョブ投入時にデフォルトでコアリテンションが適用されるように変更します。
変更は、1カ月程度の検証期間を設けたのちに実施します。検証期間中に、ご自身のジョブに影響があるかをご確認ください。

検証期間中、ジョブサイズが4,609ノード以上9,216ノード以下のジョブは、ジョブスクリプトにジョブ実行時の計算コアのリテンション状態遷移を行う設定（retention_state=1）をして頂くことで、コアリテンションの影響を確認することができます。検証期間中はデフォルト設定は変更されませんので、ジョブスクリプトに設定の上、ご確認よろしくお願いします。
消費エネルギーの削減にご協力お願いします。

<table>
<tr><th>ジョブサイズ</th><th>現行設定</th><th>検証期間中(1ヶ月程度)</th><th>検証期間以降</th></tr>
<tr><td>4,608ノード以下</td><td colspan="2">コアリテンション有効
(デフォルト・変更可能)</td><td rowspan="2">コアリテンション有効
(デフォルト・変更可能)</td></tr>
<tr><td>4,609～9,216
ノード</td><td rowspan="2">コアリテンション無効
(デフォルト・変更不可)</td><td>コアリテンション無効
(デフォルト・
ジョブスクリプトに設定することでコアリテンション有効へ変更可能)</td></tr>
<tr><td>9,217ノード以上</td><td colspan="2">コアリテンション無効
(デフォルト・変更不可)</td></tr>
</table>

New

Linaro Forge Hands-on Tutorial

- Date and time: 15:00-17:00pm JST, Sep. 11st
- Venue: online
- Agenda:
  - 10 minute overview of Forge
  - 50 minute DDT hands on exercise on Fugaku
  - 50 minute MAP / PR hands on exercise on Fugaku
- Lecturer: Mr. Rudy Shand (Field Application Engineer Affiliation, Linaro Limited)
- Register at: https://riken-jp.zoom.us/webinar/register/WN_lH92uSbaRp6UjnDxquE82A
- Registration Deadline: Sep. 4th
- Note: no more than 36 people can attend this tutorial.

New

Linaro Forge Hands-on Tutorial

- Abstract

Linaro Forge combines DDT for parallel high-performance application debugging, MAP for performance profiling and optimization advice, and Performance Reports for summarizing and characterizing both scalar and MPI application performance. Linaro Forge supports various CPU and GPU architectures and parallel programming models, including MPI, CUDA, OpenACC and OpenMP.

We will provide an overview of Linaro Forge, a cross platform, integrated environment for debugging and optimizing parallel codes at any scale. We will provide hands-on demonstrations and exercises of how Linaro Forge reduces development time, simplifies debugging, and eases application performance enhancement on Fugaku.

- Topics
  - ***Ensuring Program Correctness with Linaro DDT***: Using sample codes, we will walk through the major capabilities of the debugger to illustrate how DDT can debug applications ranging from a single thread to large scale.
  - ***Performance Engineering with Performance Reports and MAP***: We will illustrate how you can understand the nature of your application's performance. We will introduce best practices to attain and maintain optimal performance.

「サテライト富岳」の提供開始（7/16 15:30より）

新規

7/11のユーザブリーフィングにて周知させていただきましたとおり、
7/16 15:30に「サテライト富岳」（AWS Graviton3E 環境）の提供を開始しました。

ご利用方法につきましては以下の利用手引書をご参照ください。
https://www.fugaku.r-ccs.riken.jp/doc_root/ja/user_guides/SatelliteFugakuUserGuide/

参考：
第40回ユーザブリーフィング資料 p.22-23
https://www.fugaku.r-ccs.riken.jp/doc_root/ja/guidances/SystemReport_202407.pdf

Satellite Fugaku is now opened (from 3:30pm on the 16th of July).

New

As annouced in the Fugaku User Briefing on the 11th of July, Satellite Fugaku (AWS Graviton3E) is opend at 3:30pm on the 16th of July.

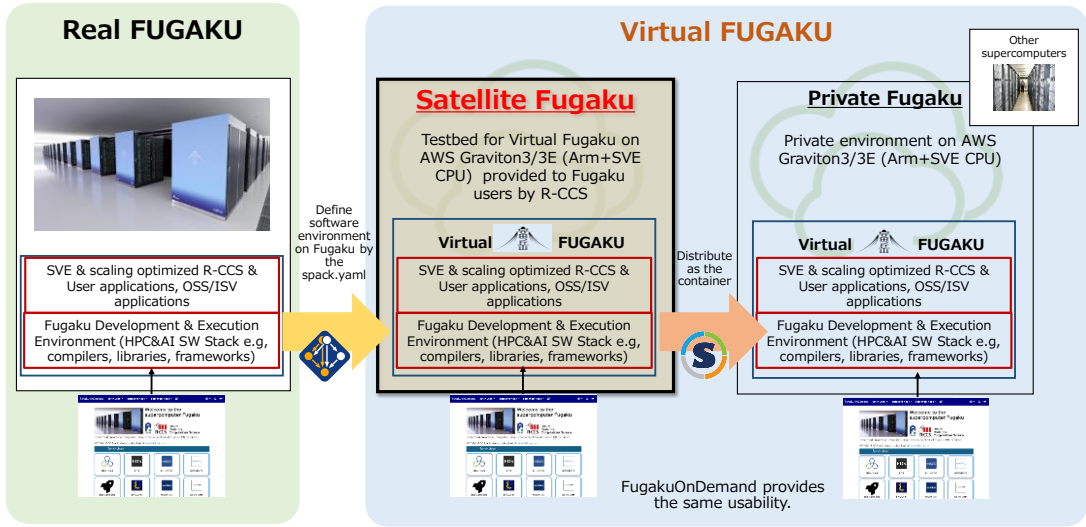
Please refer the following user guide and learn how to use Satellite Fugaku.
https://www.fugaku.r-ccs.riken.jp/doc_root/en/user_guides/SatelliteFugakuUserGuide/

References:
The 40th Fugaku User Briefing Report#1 p.22-23 (Japanese only)
https://www.fugaku.r-ccs.riken.jp/doc_root/en/guidances/SystemReport_202407.pdf

「サテライト富岳」公開のお知らせ (7/16 15:30オープン)

再掲

- 「バーチャル富岳」構想の一翼を担う「サテライト富岳」を来週オープン
  - 「バーチャル富岳」は、「富岳」で整備されたソフトウェア環境を富岳以外でも利用可能とすることでHPCエコシステムの構築を目指すR-CCSが推めるプロジェクト
    - HPCアプリケーションの流通性の向上
    - HPCシステム構築の手間の大幅削減
    - 最先端かつ生きたソフトウェア環境で常に更新されるソフトウェア環境
    - 第一歩として、A64FXと同一アーキのGraviton CPUをターゲットにプロジェクトを始動
  - 「サテライト富岳」は、「バーチャル富岳」のテスト環境および「バーチャル富岳」を構成するソフトウェアの整備環境
    - まずは、Graviton3Eを「富岳」利用者へ開放。アプリ移植や動作確認・簡易性能評価にお使いいただけます。
    - バーチャル富岳を進化させるために、不足ライブラリなどご要望を承ります。
    - 資源には限りがあること、ご理解ください。
  - バーチャル富岳のソフトウェア環境は、コンテナ化
    → 誰でも入手・利用可能に
    (バーチャル富岳コンテナを導入したシステム＝「プライベート富岳」)
    - アプリケーション配付の手段としてプライベート富岳向けコンテナへの同梱をご検討ください



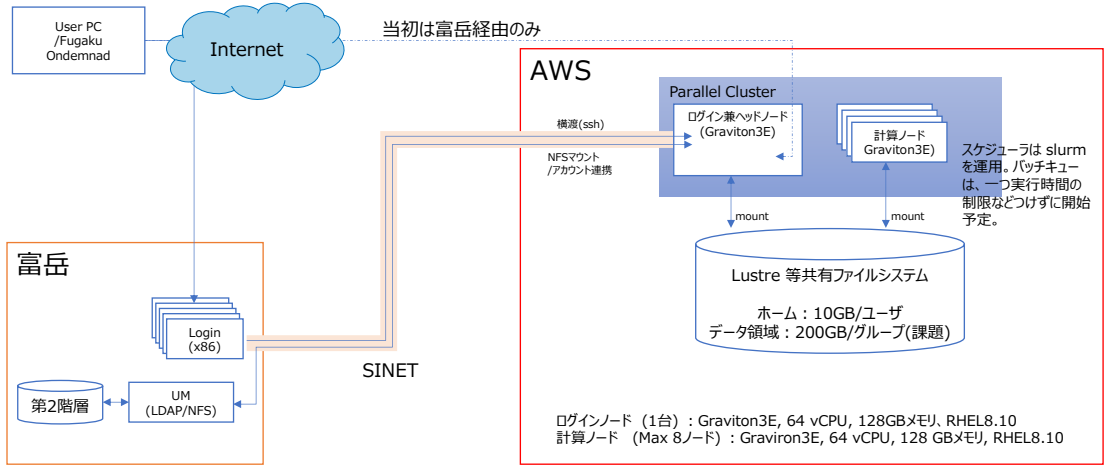
「バーチャル富岳」ご紹介

「サテライト富岳」の概要

再掲

サテライト富岳　システム概要

- **Graviton3Eのログインノードを常設**
- **計算ノードはMAX８ノードで開始（８ノード以上はご相談ください）**



1

サテライト富岳のソフト環境＝バーチャル富岳 バージョン1

- **富岳で利用頻度の高いトップ100 OSSとそのビルド環境をSPACKで管理**
  - gcc 8.5.0 & 14.1.0 + EFA対応版OpenMPI
  - 代表的OSS
    - genesis
    - scale
    - gromacs ~sve
    - openfoam
    - quantum-espresso
    - metis
    - parmetis
    - gsl
    - tmux
    - petsc
    - openbabel
    - paraview+osmesa+python~qt
    - gnuplot
    - lammps
    - julia
    - py-scipy
    - py-numpy
    - py-mpi4py
    - py-matplotlib
    - py-ase
    - py-toml
    - py-scikit-learn
    - py-pandas

1

【サテライト富岳】ネットワークメンテナンスのお知らせ（9/10）

新規

サテライト富岳をご利用いただきありがとうございます。

サテライト富岳のネットワークメンテナンスを実施いたします。
以下の時間帯で10秒程度の通信断が複数回発生する可能性があります。

実施日：2024年9月10日(火) 0:00 - 2:00

ご迷惑をおかけしますが、よろしくお願いいたします。

[Satellite Fugaku] Network Maintenance Information (Sep 10th)

New

Thank you for using Satellite Fugaku.

This is the announcement for maintenance on Satellite Fugaku's network.
Around ten seconds network disconnection for several times might occur in the following period.

Period: 2024/09/10(Tue) 0:00 - 2:00（JST）

Sorry for inconvenience, and thank you for your cooperation.

2024年度の電力運用について

再掲

- 現時点で，2024年度も計算ノードの部分停止は実施しない見込みですが，引き続き省電力運用へのご協力をお願いします.

- 電力量資源の配分
  - 2023年度と同様に，各課題には配分済みの計算資源量に応じた電力量資源を配分します.
  - accountj_pt コマンドで電力量資源を確認できます.
  - 電力量資源を使いきってもジョブの実行制限は行いません.

- ジョブ実行時の電力設定
  - 引き続き消費エネルギーが最小のモードで実行をお願いします.
    - ノーマルモード
    - エコモード
    - ブーストエコモード
    - ブーストモード
      - EDP (Energy Delay Product)が増加しない範囲であれば単独利用は可
  - 4,608ノード以下のジョブは core retention が有効

制限事項：2024年度 Boostモード利用について

再掲

富岳ではシステムの消費電力を削減するため，当面の間 Boostモード (2.2GHz)のみの利用を制限しています．
利用可能になりましたら，連絡させて頂きます．

富岳では実行モードの設定より，エネルギー効率が良いジョブ実行を行うことができます．
以下が代表的な3種類のモードですが，消費エネルギーが最小のモードで実行をお願いします．
・ノーマルモード
・エコモード
・ブーストエコモード

実行モードは，ジョブスクリプトのオプションで以下を指定します．
・ブーストエコモードの場合
#PJM -L "freq=2200,eco_state=2"
・エコモードの場合
#PJM -L "eco_state=2"

以下はブーストモードのみ使用する場合の例ですが，このモードは利用しないでください。
・ブーストモードのみの場合
#PJM -L "freq=2200"

ジョブ毎の消費エネルギーは，ジョブスクリプトに以下の行を追加し，stats ファイルでジョブの電力・エネルギー情報を確認することができます．。
#PJM -s

statsファイルにジョブの電力・エネルギー情報が以下のように出力されます．
AVG POWER CONSUMPTION OF NODE (IDEAL) : 3595.445134
ENERGY CONSUMPTION OF NODE (IDEAL) : 905.290095

AVG POWERは、各ノードの平均電力の合計を表しており，単位はWです．
上記の例は32ノードで実行しているので，1ノードあたりの平均電力は 約112W です．
ENERGYは，ジョブの開始から終了までの全ノードのエネルギー消費量を表しており，単位はWhです．

実行モードの設定による実行性能への影響の有無および大小は実行するプログラムにより異なります．
性能低下が許容範囲で，電力量の削減効果が認められる場合は、そのモードでの実行をお願いします．

詳細は以下をご参照ください．
・利用手引書 －利用およびジョブ実行編
7. 電力制御機能
https://www.fugaku.r-ccs.riken.jp/doc_root/ja/user_guides/use_latest/PowerControlFunction/index.html

2024年度富岳ポイントについて

再掲

- 省電力運用にご協力いただいたユーザに富岳ポイントを付与します.
  獲得したポイントに応じて付与した計算資源量を上限とし，優先的にジョブを実行することができます.
  - 対象課題：有償課題を除く課題
  - ノード時間積は追加されません
- ポイントの計算方法
  - ジョブ実行時に削減された電力量に基づき計算します.
  - 富岳ポイントは、ジョブ単位で付与します
  - 1ポイント＝420NS（ノード秒）
  - 富岳ポイント $= \frac{(\text{実行されたジョブのノード時間積} \times \text{ノード時間積あたりの消費電力量の基準値} - \text{実行されたジョブの消費電力量})}{\text{ノード時間積あたりの消費電力量の基準値}}$
    - ノード時間積あたりの消費電力量の基準値 = 配分電力量 / 配分計算資源量
    - 電力消費が多いジョブの場合，富岳ポイントはマイナスになる場合があります
  - 獲得ポイント数は、accountj_pt で確認できます
    - 詳細は，利用者支援ツール 使用手引書 (利用者編) 3.1.12.富岳ポイント表示をご確認ください
- 優先ジョブ実行
  - 優先ジョブ実行可能期間は、期末の9月，3月の2か月を除いた稼働日
    - A期課題　　：　4月－翌年3月末
    - B期課題　　：10月－翌年9月末
    - 随時募集課題：期間内
  - 付与された優先利用用計算資源量を上限として，優先的にジョブを実行できます
    - 専用のリソースグループを設定します
    - ノード時間積の追加はありません．各課題の計算資源量がなくなると優先実行できません
    - 使いきれなかった場合，翌期(例：A期課題の場合、翌年4月以降)への繰越はありません

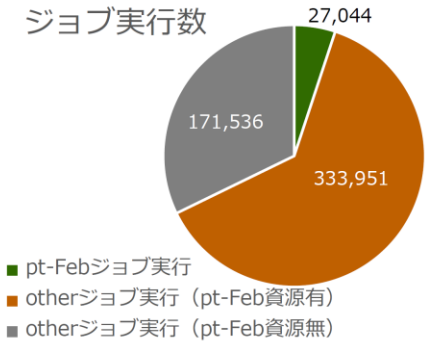
富岳ポイント利用状況

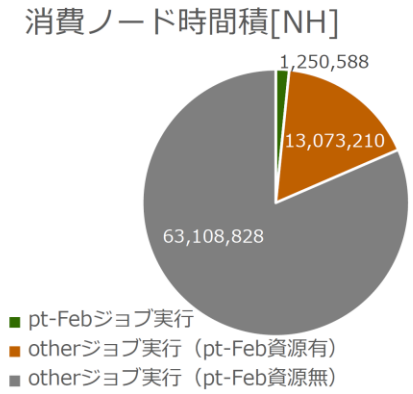
更新

- 2023年度の富岳ポイントの利用状況を分析した結果，優先ジョブ実行の権利を持っているのに，優先ジョブ実行されず，通常優先度のリソースグループで実行されていました.

項目	ジョブ数	ノード時間積[NH]
期間内ジョブ実行	532,531	77,432,626
pt-Febジョブ実行	27,044	1,250,588
otherジョブ実行（pt-Feb資源有）	333,951	13,073,210
otherジョブ実行（pt-Feb資源無）	171,536	63,108,828

other
pt-Feb以外のリソースグループ





- 2024年度も富岳ポイントを付与しておりますが，2024/8/5時点で富岳ポイントを獲得されている174課題中利用されている課題は15課題です.
  獲得したポイントに応じて付与した計算資源量を上限とし，優先的にジョブを実行することができますのでご利用ください.
  - 以下は一例です．富岳ポイントは獲得されていますが，ポイントを利用して優先ジョブ実行されていない課題.

GROUP	富岳ポイント	獲得資源(NH)
hpXXXXXX	1,900,000	223,142.8
hpXXXXXX	980,000	114,684.0
hpXXXXXX	760,000	88,711.8
hpXXXXXX	726,000	84,756.6

Power Operation in FY2024

Reprint

- At this time, no partial compute node shutdowns are expected for FY2024, but we appreciate your ongoing cooperation with energy saving operations.

- Power resource allocation
  - Power resource has been allocated to each project according to the computing resource already allocated.
  - Remaining power resource can be viewed by command accountj_pt
  - Even if you run out of power resource, there are no restrictions on running the jobs.

- Power mode on job execution
  - Please use the mode that consumes the least amount of energy.
    - Normal mode
    - Eco Mode
    - Boost Eco
    - Boost
      - You can use boost mode if EDP (Energy Delay Product) does not increase.
  - Core retention is valid on jobs using less than 4,609 nodes

Restriction：About using of Boost mode in FY2024

Reprint

Due to reduce power consumption, we are limiting the sole use of Boost mode (2.2GHz) for a while. We will contact you when it becomes available.

Fugaku can perform energy-efficient job execution by the execution mode setting.
The following are three typical modes.
Please use the mode that consumes the least amount of energy.

- Normal mode
- Eco Mode
- Boost Eco Mode

The execution mode is specified in the job script options as follows.

- In case of Boost Eco mode
  #PJM -L "freq=2200,eco_state=2"

- In case of Eco mode
  #PJM -L "eco_state=2"

The following is an example of using only the boost mode, but DO NOT USE this mode.

- In case of Boost mode only
  #PJM -L "freq=2200"

The energy consumption of each job can be checked by adding the following line to the job script and checking the stats file.
#PJM -s

The power and energy information for the job is output to the stats file as follows.
AVG POWER CONSUMPTION OF NODE (IDEAL) : 3595.445134
ENERGY CONSUMPTION OF NODE (IDEAL) : 905.290095

AVG POWER represents the total average power of each node, in W.
The above example is run on 32 nodes, so the average power per node is about 112W.
ENERGY represents the energy consumption of all nodes from the start to the end of the job, in Wh.
Whether or not the execution mode setting has an effect on execution performance, and how large the effect is, depends on the program being executed.
If the performance degradation is acceptable and the power reduction effect is recognized, please run the job in the mode you used.

Please refer following for details.
Users Guide - Use and job execution
7. Power control function
https://www.fugaku.r-ccs.riken.jp/doc_root/en/user_guides/use_latest/PowerControlFunction/index.html

Fugaku Points in FY2024

Reprint

- We will give Fugaku points to users who cooperate in energy conservation
  Fugaku points are awarded based on the amount of electricity saved through energy conservation.
  You can execute the priority job up to the calculation resource amount according to the Fugaku point
  - Excluding Fee-based Access Projects
  - Computing resources are not added
- How to calculate Fugaku points
  - Calculate points based on electric energy reduced when you execute jobs
  - Fugaku point is given on job basis.
  - 1 Fugaku point ＝ 420 node・seconds
  - Fugaku point $= \frac{\text{(used node hours by jobs x standard value of energy per node hour} - \text{used energy by jobs)}}{\text{standard value of energy for each node hour}}$
    - distributed average energy = distributed energy / distributed computing resources
    - For jobs that consume a lot of electricity, the Fugaku point may be minus
  - You can check Fugaku points by command “accountj_pt ”
    - For more details, please refer to the manual.
      User support tools User's Guide (for User) documentation 3.1.12. Displaying Fugaku Points
- Priority job execution
  - Priority job execution periods are system operating days except September and March
  - You can execute priority jobs up to distributed computing resources for priority job execution
    - Dedicated resource group will be set up
    - Computing resources are not added. When you used all computing resources for each project, you can not execute priority jobs
    - If not fully used, the remaining computing resources for priority jobs are not carried over to the next period.

効率良くジョブを実行する方法

効率良くジョブを実行するために，以下の設定を試してみてください.

- ジョブ投入時の形状を指定しない，IO専有を指定している場合は指定しない
  - ジョブの形状(2次元，3次元)を指定しないでノード数のみを指定するとスケジューリングされやすくなります.
  - IO専有の指定がある場合もスケジューリングに時間がかかるので，指定しないことでスケジューリングされやすくなります.

- 指定経過時間をできるだけ短くする
  - スケジューラは指定経過時間に基づいてスケジューリングを行いますので，指定経過時間が短いほうがスケジューリングされやすくなります.
  - 最低実行時間指定機能を利用し指定経過時間を短く指定することでスケジューリングされやすくなります.

- smallリソースグループを利用する場合は，torusを指定する.
  - smallジョブでtorusを指定していると，largeジョブの隙間で実行される場合があります.
  - 特に経過時間が1時間以下の場合は，大幅に待ち時間が短縮できる場合があります.
  - ただし，torusを指定することによりスケジューリング単位が12ノード単位になるので注意が必要です.

Tips for Effective Job Execution

For efficient program execution, try the following

- Not specifying the shape when submitting jobs, and not specifying I/O exclusive-mode
  - Specifying only the number of nodes without specifying the shape (2D or 3D) makes it easier to be scheduled
  - If I/O exclusive-mode is specified, waiting time will take longer.

- Shorten the specified elapsed time as much as possible
  - Scheduler schedules jobs based on the specified time, so the shorter the specified elapsed time, the easier it is to be scheduled.
  - Shorter the specified elapsed time using the minimum elapsed time limit value, the easier it is to be scheduled.

- Specify torus when using small resource group
  - When specifying torus on small jobs, jobs may be executed in gaps between large jobs.
  - Waiting time can be significantly reduced, especially if the specified elapsed time is within one hour.
  - Note that the scheduling unit is 12 nodes when torus is specified.

ノード規模に応じた1ノードに割り当て可能な限界プロセス数について

ノード（プロセス）規模に応じてシステムが使用するメモリ量が変わる箇所があります。
MPIのメモリ使用量を元に限界プロセス数を示します。

マニュアル「Development Studio MPI使用手引書 6.11.1 メモリ使用量の見積り式」で算出したメモリ使用量を元に、リソースグループlarge領域最大ノード数、中規模最大ノード数、1/2規模最大ノード数、全ノード数に応じた1ノードに割り当てるプロセスの数(以下、ppn)の限界値を算出しました。
※この表で示す限界ppnは見積り式から算出した目安の値です。

ノード規模	ラージページの限界ppn	ノーマルページの限界ppn
12,288(large最大ノード数)	48(*)	48(*)
55,296(中規模最大ノード数)	29	36
82,944(1/2規模最大ノード数)	24	29
158,976(全ノード数)	17	21

(*)ppn=48の場合においてメモリの限界となる最大ノード数は以下の通りです。
・ラージページ利用時　：19,114
・ノーマルページ利用時：30,696

マニュアルに記載されている見積り式はMPI_INITルーチンが呼び出された直後の1プロセスあたりのメモリ使用量を算出するものです。
このうち通信アルゴリズムなどにより変化する値を除外したもの（C_Proc×N_Proc＋C_Base）とppnの積が「24GiB」以下となるノード数とppnの組み合わせを求めることで、1ノードに割り当て可能な限界プロセス数とします。

詳細は、富岳ウェブサイトをご確認ください。
https://www.fugaku.r-ccs.riken.jp/faq/20240207_01

Maximum number of processes that can be allocated to a node according to the node size

Depending on the size of the node (process), the amount of memory used by the system may vary.
In this example, the limit number of processes is calculated based on the memory usage of MPI.

Based on the memory usage calculated in "6.11 Memory Usage Estimation Formulae and Tuning Guidelines" of the manual "Development Studio MPI User's Guide" the limit value of the number of processes (Hereafter referred to as ppn) to be allocated to one node according to the scale was calculated.
* The limit ppn shown in this table is the approximate value calculated from the estimation formula.

maximum number of nodes	Large page limit ppn	Normal page limit ppn
12,288(Resource group large)	48(*)	48(*)
55,296(Medium-scale)	29	36
82,944(Half the size of the full node)	24	29
158,976(Full nodes)	17	21

(*) When ppn=48, the maximum number of nodes allowed is as follows:
- When using large pages: 19,114
- Normal page usage: 30,696

The estimation formula described in the manual calculates the memory usage per process immediately after the MPI_INIT routine is called.
The maximum number of processes that can be allocated to 1 node is determined by determining the "combination of the number of nodes and ppn" where the product of ppn and the value that changes (C_Proc × N_Proc + C_Base) due to communication algorithms is 24GiB or less.
Please check Fugaku Website for details.
https://www.fugaku.r-ccs.riken.jp/en/faq/20240207_01

適切なアカウント管理のお願い

「富岳」ローカルアカウントは、発行時にユーザ毎に「外国為替及び外国貿易法」（外為法）に基づき審査を行っており、第三者が利用すると法律違反となる場合があります。

「富岳」利用規則に則り、第三者に利用させることが無いよう適切な管理をお願いします。

違反した場合は、アカウント停止等の厳格な処置を行います。

「富岳」の複数ユーザによる公開鍵の共有等によるアカウントの共有も違反対象です。

課題代表者の方々には、課題参加者への「富岳」利用規則の周知をお願いします。

適切なアカウント管理をお願いします。

※スーパーコンピュータ「富岳」利用規則(PDF)
https://www.hpci-office.jp/materials/f_riken_fugaku_kitei.pdf

Management of your account properly.

Fugaku account is issued to each user in accordance with any laws and regulations regarding security export control.

So, it may be against the law for a third party to use your account.
Please manage your account appropriately in accordance with the Regulations for Use of the supercomputer Fugaku.

If you are against the law, R-CCS will take strict action, such as suspending your account.

It is also against the law to share the Fugaku account with more than one Fugaku user by sharing the public key.

The project leader should inform the members of the Regulations for Use of the supercomputer Fugaku.

Please manage your account appropriately.

Regulations for Use of the supercomputer Fugaku (PDF)
https://www.hpci-office.jp/materials/f_riken_fugaku_kitei_en.pdf

2024年度低優先度ジョブ実行(spot)について

- 2024年度より低優先度ジョブ実行利用を以下のとおり変更します
  - 2023年度まで：課題に付与した計算資源量を98%消費した際に利用可能
  - 2024年度以降：通年で随時利用可能

- 対象課題：有償課題を除く課題
- 対象リソースグループ：spot-large, spot-small, spot-int, spot-middle
- 課金RATE：rate=0%（無償）
- その他
  - 低優先度ジョブはノードが空いている場合に実行されます
  - 低優先度ジョブのリソースグループの設定は，優先度以外は通常リソースグループ(spot-XXXのXXX部分)と同じです

Low priority jobs(spot) in FY2024

- Change the use of low-priority job(spot) execution as follows
  - FY 2023: Available when 98% of the assigned computing resources are consumed
  - FY 2024: Available at any time

- project: Excluding Fee-based Access Projects
- Target resource group: spot-large, spot-small, spot-int, spot-middle
- Billing RATE: rate=0% (free)
- Note
  - Low priority jobs are executed when compute nodes are free.
  - The configuration of low priority job's resource groups is the same as the normal resource groups, except for the job's priority.

ログインノードに対して使用メモリ量の制限を設定しました

ログインノードの安定化を目的として、以下の通りログインノードに対して使用メモリ量の制限を設定しました。

対象ノード: login1-6
変更内容 ：各ログインノードにおける全利用者の合計使用メモリ量の制限を148GiBに設定します。
変更日時 : 2024/05/30 13:00 - 14:00

留意事項:
- 変更作業に伴うノード、サービスの停止等はありません。
- ログインノードで全利用者の合計使用メモリ量が148GiBを超過した場合は、使用メモリ量の多いプロセスを強制終了します。

ログインノードの安定した運用を行うために、ご理解とご協力をお願い致します。

Sets the memory usage limit for the login node

To ensure stable operation of the login node, the system sets the memory usage limit for the login node as follows:

Target node: login1-6
Changes : Sets the total memory limit for all users on each login node to 148GiB.
Modified : 2024/05/30 13:00 - 14:00

Notes:
- There are no nodes, service outages, etc. associated with change operations.
- If the total amount of memory used by all users on the login node exceeds 148GiB, the process with the largest amount of memory used is forcibly terminated.

We appreciate your understanding and cooperation to ensure stable operation of the login node.

ログインノードのメモリ使用について

ログインノードの安定化を目的として、5/30にログインノード(login1-6)は、全利用者の合計使用メモリ量を148GiBに制限しました。
これに伴い、6/24より、制限を超過した場合、システムにて使用メモリ量の多いプロセスを強制終了し、強制終了したプロセスを利用者様に通知します。
利用者様におかれましては、以下の対応をお願い致します。

・ログインノードにて、以下の目安を超過するメモリを使用するプロセスや大量プロセスの生成はお控えください。
・使用メモリ量の大きいプロセスを生成する必要がある場合はプリポスト環境や富岳本体をご利用ください。

ログインノード 1台で以下の資源量を超過しないようご注意願います。

1利用者様の最大資源量の目安

資源	目安
最大スレッド数	8 スレッド
最大メモリ容量	12 GB

ログインノードの安定運用に、ご理解とご協力お願い致します。

About Memory usage of login nodes

To stabilize login nodes, as of May 30, login nodes (login1-6) have limited the total memory usage of all users to 148GiB. Accordingly, from June 24, if the limit is exceeded, the system will forcibly terminate processes that use a large amount of memory and notify users of the forcibly terminated processes.
Users are requested to take the following actions.
- On the login node, do not create a process that uses more memory than the following guideline or a large number of processes.
- If you need to create a process that uses a large amount of memory, use a Pre/Post environment or Fugaku main unit.

Make sure that one login node does not exceed the following resources:

Guideline of the maximum amount of resources for users

Resource	Guideline
Maximum number of threads	8 threads
Maximum memory	12 GB

We appreciate your understanding and cooperation for stable operation of the login node.

参考資料 / References

Index

Index

ログインノードおよびクラウドストレージゲートウェイのデフォルト文字コードについて

- ログインノードおよびクラウドストレージゲートウェイのデフォルトの文字コードは「en_US.utf8」です.
- 「富岳」の計算ノードのデフォルトの文字コードは「C」で，ログインノードとは異なるので注意してください.
- ログインノードの文字コードの変更を行う場合は，ログインノードの~/.bashrc に下記のように「export LANG=C」等を追記してください.

Node	デフォルト 文字コード
ログインノード (login.fugaku.r-ccs.riken.jp)	en_US.utf8
クラウドストレージゲートウェイ (csgw.fugaku.r-ccs.riken.jp)	en_US.utf8
「富岳」計算ノード	C

```
$ env | grep LANG
LANG= en_US.utf8
$ vim ~/.bashrc
export LANG=C     ## 追加
$ bash
$ env | grep LANG
LANG=C
```

About the default character encoding for Login Nodes and Cloud Storage Gateway Nodes

- Default character encoding of the Login nodes and the Cloud storage gateway nodes is "en_US.utf8".
- Please note that default character encoding of Fugaku compute node is "C", which is different from the Login nodes.
- If you want to change the character encoding, please add “export LANG=C” or etc. to ~/.bashrc file like following.

Nodes	Default character encoding
Login node (login.fugaku.r-ccs.riken.jp)	en_US.utf8
Cloud storage gateway node (csgw.fugaku.r-ccs.riken.jp)	en_US.utf8
Fugaku compute node	C

```
$ env | grep LANG
LANG= en_US.utf8
$ vim ~/.bashrc
export LANG=C    ## Added
$ bash
$ env | grep LANG
LANG=C
```

「富岳」利用準備課題ヒアリング内容への対応状況について (1/3)

設問内容：利用にあたり困ったことはありましたか

No	回答内容	対応状況
1	vcoordファイルによる資源の割り当てを用いると、ジョブ実行時にエラーになってしまいます。	SPACK利用時にLD_LIBRARY_PATHが書き換えられ，異なるverのDLLが呼ばれていたのが原因. 解決済み
2	ドキュメントの所在や記載がやや分かりづらかった。実機上での試行錯誤を通じて解決した。	継続対応中
3	富岳ポータルに掲載されているチューニングガイドはFORTRANのものであるため，C言語やC++でのチューニングに関する資料が欲しいです。	プログラミングガイド・チューニング編について，C/C++の事例を追加
4	ドキュメントの検索方法がドキュメントごとにしかできないため、探すのが手間だった。特にドキュメント内で別の資料を参照とあるのにリンク等ですぐ見に行けないのが使いにくかった。	全文検索システムを導入済み

「富岳」利用準備課題ヒアリング内容への対応状況について (2/3)

設問内容：利用環境として、ライブラリ、アプリケーションについての要望はありますか

No	回答内容	対応状況
1	tensorflowのC ++サポート。	導入済み
2	SPACKにGMTを導入いただけると助かります。	導入済み
3	我々の計算にはFFTと大行列対角化が大きな時間を占めますので、その高効率化を望みます。	FFT並びに対角化は高速化済み．これ以上の高速化には何らかのアルゴリズムの進展かつ新規開発が必要になるため難しい． FFTはSSL2，fftw，ffteを利用シーンに応じて使い分ける．対角化は理研のeigen_FSを利用するなど，利用シーンに応じたTipsを整備
4	Spackが安定して利用できないなど、一部のストレージが不安定なことが原因と思われる問題があった。改善願いたい。	Spack自体については，9月の更新でかなり改善する見込み
5	Singularityなどのコンテナ仮想化環境。	Singularityを導入済み．ユーザガイドを公開済み． 利用講習会を実施予定
6	アプリによっては特定バージョンのOSS（たとえばPython）を必要とする場合があり、そういった要望を受け付けていただいて、スピーディーに対応していただけると大変有益です。	OSSのインストールについては，ベストエフォートでの対応． ユーザ自身でインストールする方法も案内する
7	並列ライブラリの強化（高速化）。	SSL2，理研ライブラリ，OSSライブラリの利用を検討してほしい．OSSの版用ライブラリの高速化は別途開発が必要になるため難しい．
8	Anaconda	対応予定なし

「富岳」利用準備課題ヒアリング内容への対応状況について (3/3)

設問内容：利用環境として、ライブラリ、アプリケーションについての要望はありますか

No	回答内容	対応状況
9	リモートデスクトップなどを用いて可視化まで富岳でできると嬉しいです．また、可視化にはTecplotが導入されていると助かります．	VNCが利用可能．Tecplotは運用面とコスト面で検討した結果，導入が難しいと判断．ユーザー個人で契約したライセンスの利用をお願いする．
10	A64FXの性能をフルに発揮するよう、弊社プログラムの最適化を進めていきたいが、富士通様のチューニングマニュアルが、私たち素人には大変高度で難しい。企業ユーザーを想定したより平易な内容のドキュメントの公開を希望します。また弊社は「非構造格子」流体解析手法を採用しており、「非構造格子」向けのチューニング手法に関してドキュメントを公開して頂ければありがたいです。	ドキュメントの拡充を検討する。個別のアプリケーションの事例については例えばRISTと共催の "Meetings for application code tuning on A64FX computer systems" を参照ください。
11	可能であれば、Spack によりインストール済みのパッケージについては、コンパイル時の条件設定やコンパイル用スクリプト等を公開いただければと思います。我々のように自身でプログラムを最適にコンパイルする必要があるユーザには大変参考になると思います。	ビルド時のログはパブリックインスタンス内に保存されているので，必要に応じて情報を提供可能
12	疎行列を扱っているとSIMD化できない計算ばかりと言って良いでしょう。 固体力学のFEM解析なら係数行列の一行あたりの演算量も多いですが、流体解析だと一行当たりの非零項数が少なく、スカラー演算にならざるを得ない。非構造格子だとELL方式も難しい。動作クロックをもう少し上げて欲しかった。 1CMG当たりのメモリーが少ないのは不満です。HBM2のメモリーだから顕著には影響が現れていないのかもしれませんが、スペックを見る限り、キャッシュメモリーが小さいのではないでしょうか。	富岳では拡張されたSIMDに相当するSVE機構を採用したgather(間接ロード）/scatter(間接ストア）の命令がサポートされています。プログラム・データの構成にも依存しますが、gather性能はスカラーロードよりも概して性能が高い結果が知られています。疎行列の計算が間接インデックスを含むループ処理であれば、コンパイラが最適化した結果、gatherを含む処理がベクトル化処理の適用をうけたことをコンパイルリストでご確認いただき、もしベクトル化されていなければ、オプションやディレクティブの指定・ソースの部分的書き換えなどで、ベクトル化が実現できるかをご検討ください。scatter性能はスカラーストアと比較して高く無いことが多いと知られています。 FEM流体計算アプリケーションにおいても要素・節点の処理順序を最適化したコーディングを行って、計算速度を高めた実績例もありますので、論文等をご参照の上、同様なリファクタリングが可能かをご検討ください。 動作クロック・メモリサイズ・キャッシュサイズについてのご意見は今後の参考とさせていただきます。
13	It would be great if some open-source codes could be accessed directly from the system, e.g., Quantum Espresso, CP2K for materialsmodelling. And more specific instructions on compilation of some common open-source code software, e.g., the makefile settings, pathsettings, etc., would be useful.	Spack外の提供は原則として行いません 典型的なmakefile設定の提供は検討中
14	情報が少なすぎるので、チューニング結果、コンパイル方法、ノウハウなどが公のブログなどに書けるような雰囲気になってほしい。	公開情報を増やすことを検討中

Status of response to Fugaku Preliminary Use Project's hearing content (1/3)

Question : Did you have any difficulties in using the system?

No	Answers	Current status
1	We have the job execution error using vcoord file.	LD_LIBRARY_PATH was changed when using SPACK, and different DLL was called. This issue is fixed.
2	The location and description of the documents were a bit confusing. We solved the problem through trial and error on Fugaku.	In progress
3	Tuning guides on Fugaku website are written for FORTRAN, so we want them for C/C++.	We added examples of C/C++ to "Programming Guide (Tuning) "
4	The searching for documents was for each document, which was difficult. In particular, it was inconvenient to be able to quickly refer to a link to another documents in the document.	Full-text search system was implemented

Status of response to Fugaku Preliminary Use Project's hearing content (2/3)

Question : Do you have any requests for library or applications for user's environment?

No	Answers	Current status
1	Support C++ on tensorflow	Supported
2	Installation of GMT on SPACK	Supported
3	Optimization of libraries for FFT and Large matrix diagonalization	Libraries for FFT and Large matrix diagonalization are optimized for Fugaku (More optimization is difficult because new development is needed). We made tips on how to use SSL2, fftw and ffte as FFT, to use RIKEN eigen_FS as large matrix diagonalization according to the situation.
4	Stabilization of Spack and file system.	Spack will be stabilized in the September update. Operation of the filesystem has been tuned.
5	Container virtualization environment like Singularity	Singularity is available. User guide is available, and lecture will be held soon.
6	Some apps need specific version of OSS (like Python)., thus It would be great if you could respond quickly to these requests.	Installation of OSS is a best-effort response. We will announce the way to install OSS by yourself.
7	Optimization of parallel libraries.	Please use SS2, RIKEN's libraries and OSS libraries. Optimization of OSS libraries is difficult because new development is needed.
8	Anaconda	No plan

Status of response to Fugaku Preliminary Use Project's hearing content (3/3)

Question : Do you have any requests for library or applications for user's environment?

No	Answers	Current status
9	It would be great if we can do all visualization on Fugaku using like a remote desktop. We want Tecplot for visualization.	VNC is available. Installation of Tecplot is difficult due to the cost. If you want to use it, please purchase an individual license by your self.
10	We want to optimize our programs for utilizing the performance of A64FX, but Fujitsu's documents are difficult for us amateurs. So we want simple documents, and we are grate if the documents for tuning methods for unstructured lattices are available.	We will add more documents. For examples of individual applications, please see "Meetings for application code tuning on A64FX computer systems" on HPCI office website.
11	If the setting, script and etc for installation of OSS provided by Spack are available, users who optimize programs by myself will be grate.	Log is stored in a public instance, so it will be available on demand.
12	We think it is difficult to apply optimization using SIMD extensions for sparse matrix. If you're doing an FEM analysis of solid mechanics, there's a lot of computation per row of the coefficient matrix. But in fluid analysis, the number of non-zero terms per line is small, and scalar operations have to be used. The ELL method is also difficult for unstructured lattices. We want more CPU freq, and more memory per 1CMG. Because of HBM2, the effect may not be noticeable, but we think the cache memory is small from the specification.	Instructions (gather/indirect load and scatter/indirect store) with SVE are available on Fugaku. Depending on the structure of a program and data, performance of gather is higher than that of scalar load. If the sparse matrix calculation is a loop operation with indirect indices, see whether the operation including gather is vectorized as a result of compilation. If not, please try to rewrite the source code to be vectorized. In most of cases, the performance of scatter is not higher than that of scalar store. In FEM fluid calculation applications, there are some examples of improved computation speed by optimizing the processing order of elements and nodes. Please refer to the paper and consider whether similar refactoring is possible. Thank you for your comments on CPU freq., memory size and cache size.
13	It would be great if some open-source codes could be accessed directly from the system, e.g., Quantum Espresso, CP2K for materials modelling. And more specific instructions on compilation of some common open-source code software, e.g., the makefile settings, path settings, etc., would be useful.	We will support Spack only, and we are considering for offering typical setting of Makefiles.
14	There is little information about tuning results, compile method and other know-how, so we are grate if these information is available on blogs or etc.	We will add more open information about them.

「富岳」加速プログラム課題ヒアリング内容への対応状況について (1/4)

- 富岳のスケジューリングシステムにおいて、ノード間通信が他のジョブと干渉しないようなノード割り当てができることが望ましい。トーラス指定時に、ノード形状の回転が入ると使用資源量がかわるし、プログラム内で tofu 座標を直接用いる際に(YB)軸を含む入れ替えへの対応は難しいことがあるので、複数のstrict 候補の指定あるいは、回転を抑止できるとよい。また、他のジョブの I/O と関係していると思われる実行時間の大幅な遅れが起こることがある。I/O の重いジョブはリソースを別にするなどの対策ができると助かる。
  - smallでは他のジョブの影響を受けないようなスケジューリングは難しいです（離散スケジューリングを行っているため）
  - large等は原則他のジョブの通信の影響は受けませんが，第2階層は共有のため影響を受ける場合があります
  - y軸固定のstrictについては検討したが実現は難しい
- ログインノードのCPUの増強。現状、混雑時にコンパイルに時間がかかり、非効率になっている。
  - 短期的には、コンパイルに利用する前などに https://status.fugaku.r-ccs.riken.jp のログインノード稼働情報状況を確認し、負荷が低いホストを選択してご利用ください
  - 2月保守で全てのログインノードのメモリを増強しました（96GiB => 192GiB）
- ログイン時やjob投入時に割り当て資源量に対する消費率を表示するなど、不注意による資源浪費を防ぐ仕組み。
  - motdで情報を表示するようにしました
- 重点課題の時は毎月京使用量の報告があったが、富岳になってからは無くなった。計算資源をスムーズに利用していくうえで、チームIDごとの利用状況が毎月報告されることを希望する。
  - ポータルで月次の集計表示画面が利用可能
- 富岳上でのリソースグループについて、クオータを設定可能なサブグループを設定出来るようにして欲しい。各課題はテーマごとにグループを形成しており、課題内のリソース配分は話し合って担当分をオーバーしないように利用している。しかし、参加人数が多いと管理が難しい。例えばポータル上で、課題代表者か副代表者がサブグループを作成してクオータを設定し、リソースグループに所属するメンバー各人をサブグループに振り分ける事ができるとうれしい。
  - ユーザ単位にリソースグループの利用制限を設定することは可能．サブグループを作成し，課題代表者が自由に操作するのはできない（課題をサブグループとし，複数課題全体の資源量をコントロールすることは可能）．

「富岳」加速プログラム課題ヒアリング内容への対応状況について (2/4)

- 富岳上でのSPACKを用いたオープンソースソフトウェア(OSS)の利用に関し、フロントエンド環境でのSPACKの有効化とOSSのロードにとても時間がかかる。ログインするたびにとても時間がかかるので、時間短縮に向けて改善をお願いしたい。
  - I/O負荷次第でそのようになる。Spackが生成するmodulefileやsingularityを使うと若干改善する可能性あり
- public Spackで利用されているコンパイラが古いバージョンで留まっている期間が長いので、コンパイラの更新に伴ってpublic Spackをコンパイルし直してほしい。
  - 言語環境の更新のたびに全てのパッケージを再ビルドするのは様々な負荷が大きいため、特に要望があれば個別対応する。
- 富岳のポータルサイトに掲示されるバグやシステム障害の情報を、希望者にはメールで送付するようにしてほしい。
  - 2022/4よりメール配信開始
- 「特殊な場合だけの問題」と「多くの人に影響する問題」が区別されずにリストされているように見受けられる。それぞれの記事のタイトルに、問題の重要度を3段階くらいで簡潔に表記しておいてもらえると良い。
  - 多くの人に影響する問題は「重要なお知らせ」にも掲載することを検討する。
- 優先度高のキュー（large_fastやsmall_fastという類）があるとありがたい。ただし、割当ノード時間の何％までを優先度高のキューで使って良いといったルールが必要
  - システムとしては対応可能だが，HPCIでの運用ルールの合意が必要
- 通信時間や通信速度、通信するノード間距離（ホップ数）、Tofu ユニット間の距離などを詳細に調査したところ、ノード間距離（あるいは、Tofu ユニット間距離）が長い通信によって、他の通信に大きな遅延が発生していることが明らかとなった。メモリースループットによって実効性能が決まっている他の多くのアプリケーションに共通する課題であると考えられ、早期のシステム調整を望む
  - システム調整（インタコネクト）についてシステム側の対応は困難
  - CPU・メモリ性能の高さに比較してインタコネクトの性能が相対的に低いことは指摘の通り
  - weak-scalingベンチマークにおける並列化効率はノード間通信性能の上限値の他に、アプリケーション依存の要因が大きい場合も多いので、並列化率を向上するために富岳においてどの様な対策が可能であるかを個別に検討するためにはアプリ開発者・富岳担当技術者でタスクフォース等を編成して対応することが必要

「富岳」加速プログラム課題ヒアリング内容への対応状況について (3/4)

- プリポストノードでMatlabを使えるようにして欲しい
  - ライセンスの問題から全ユーザへの提供は難しい（ライセンス持ち込みなら可能）
- デバッグ専用のキューが欲しい．デバッグで短いJobを投入しても非常に長い時間待たされることがある．また関連して，可能であればRUNになってから実際に計算が始まるまで数分程度かかることがあるのを改善してほしい (デバッグの効率を上げたいため)
  - 会話型ジョブはプライオリティを高く設定しているので活用してほしい
  - ジョブ実行に時間がかかる原因の一つにファイルアクセスに時間がかかることが考えられる．llio_transfer等の活用を検討してほしい
- おそらく「富岳」ノードの問題で，特に大規模なノードを使った計算で，配列アクセス時などにSIGBUSでランダムに計算が落ちる問題があり，計算資源が有効に使われない時が多々あり，改善して欲しい．ただし，7/16のファームウェアアップデートで改善されている？
  - ファームウェアアップデートで改善済です
- (メッシュモードでのノード割り当て，およびそれに合わせた通信アルゴリズムを使わない限り)大並列のMPI通信の速度が他のスパコンと比較して遅い問題がある．MPI通信を高速化する方法もしくは改善予定があれば教えて欲しい．
  - アプリケーションの通信特性を無視して、Tofu上にMPIランクを配置してしまうと、経路競合とホップ数の増加で通信時間が伸びてしまいます。実際にどのくらいの並列度から、どういうノード形状で、どのMPI通信に関することかを記し、サポートデスクに問い合わせください。
  - ランク配置の最適化手法を検討しています
- 運用スケジュールが富岳専用サイトでしか確認できず、スパコンの保守と同じタイミングでサイトもダウンするので再開の情報が見えないのが不便でした
  - メンテ中はメンテ時間を記載したページを表示し参照できるようにしました

「富岳」加速プログラム課題ヒアリング内容への対応状況について(4/4)

- (あまりプログラミングが得意でない人が)手っ取り早く高速化するためのキーとなる最低限のポイントを押さえた資料が欲しい．高速化の網羅的な資料があるが，キーとなる点を飲みを分かりやすくまとめた資料が欲しい．
  - ウェブサイトで以下の資料が公開されているので参考にしてほしい
    - プログラミングの観点から全般的に知っておくべき項目をスライドにまとめた参考資料
      「プログラミングガイド・プログラミング共通編」
    - アプリの高速化に影響を与えるプログラミング上の技法全般をスライドにまとめた参考資料
      「プログラミングガイド・プログラミング共通編」
    - CPUの機構や特性の観点から項目毎にスライドにまとめた参考資料
      「プログラミングガイド・プロセッサ編」
    - 実際のアプリにおいては計算特性が異なることを踏まえて、タイプ毎に高速化の指針をまとめた参考資料
      「アプリケーションのタイプ別CPU性能チューニング」
  - アプリケーションの高速化をはかる場合，通常は上記の様な各種の背景情報を踏まえた上で総合的なアプローチを取る必要がある．特にコンパイラの機能を活用した基本的な高速化の指針を比較的簡潔にまとめた部分は「プログラミングガイド・プログラミング共通編」の中の「コンパイラによる高速化」の節にある．まずはこちらをご覧いただくのが良いと思われる．
- 言語環境が変わるたびに新たなバグが出る．プログラムのバグなのかシステム側のバグなのか不明な事がほとんどで，対応に多大な時間を取られる
  - コンパイラ，MPIの改善・安定化に努めます．他のコンパイラ/環境で動いていた場合は，ヘルプデスクへ問い合わせて欲しい．GCCとLLVMが利用可能
- 富岳の富士通コンパイラ（frtpx/fccpx/mpifrtpx)はまだ古いOMPしか使えない（最近のOMPバージョン4.5/5.0/5.1が使えない）
  - ベンダへ新機能の対応状況と今後の対応予定を問い合わせるので，ヘルプデスクへ問い合わせて欲しい．

Status of response to Program for Promoting Research on the Supercomputer Fugaku's hearing content (1/4)

- I hope that the inter-node communication is not disturbed by other jobs on Fugaku. Amount of used resource will vary by rotation of node assignment, and it is difficult to deal with the replacement of node coordinates (YB axis). So I wish I could specify the axis of rotation. Elapse time may be extended by other job's IO, thus I wish that computer resource is separated from other jobs.
  - It is difficult to allocate nodes in such a way that they are not disturbed by other jobs on small due to noncont node allocation.
  - jobs on large are not disturbed by other jobs, but file IO on 2nd layer storage may be disturbed because its storage is shared among users.
  - We considered applying "strict with fixed Y axis', but found it difficult to apply.
- I hope the # of CPUs will increase. Compiling takes long time, and it is inefficient.
  - In the short term, please see the status of login nodes on Fugaku website, and select login node with low load.
  - We changed the memory capacity of all login nodes from 96GiB to 192GiB on the system maintenance in February 2022
- I hope the method to prevent loss of resources due to carelessness by displaying the resource usage when I login or submit jobs.
  - "motd" displays usage status when logging in.
- We hope the monthly reporting of each project's resource usage.
  - Monthly report is available on Fugaku's user portal.
- We want to make sub groups in our group and to set a resource quota for each sub group.
  - You can set resource quota for each users. Project's leader can not make sub groups and not set resource quota for each sub group.

Status of response to Program for Promoting Research on the Supercomputer Fugaku's hearing content (2/4)

- It takes a long time to load Spack and OSSs. Please improve the loading time.
  - Depending on the I/O load, it will be like that. module files generated by Spack and singularity may improve the situation.
- We hope that public instance of Spack will be updated with compiler update.
  - Rebuilding all the packages for every language update is a lot of work, so we will respond with a request,
- We hope the mail announce about bug information, system failures and etc on Fugaku website.
  - We started e-mail distribution of Fugaku operation information in April.
- We think limited problems and common problems are listed without distinguished, so, for example, we hope each notice is labeled as importance with 3 levels.
  - We will consider that problems effect many users will be listed as "Important notice"
- We hope resource groups with high priority will be available. In that case, we need to have a rule about how many resources can be used for those resource groups.
  - Those resource group can be made, but we need to agree on the operational rules with HPCI.
- We studied communication time, communication performance, communication distance between nodes (hops) and distance between Tofu units, and found that the communication is disturbed by long distance communication between nodes or Tofu units. This is a common problem for many applications that it's effective performance is determined by memory throughput. So we hope that the system will be adjusted as soon as possible.
  - Further system adjustments for inter-connection are difficult. As you pointed out, performance of inter-connection is worse than performance of CPU and memory throughput. In addition to the upper bound on the inter-node communication performance, the parallelization efficiency is also highly application-dependent. Discussions are needed between application developers and system operators to consider measures to improve parallelization efficiency.

Status of response to Program for Promoting Research on the Supercomputer Fugaku's hearing content (3/4)

- We want to use Matlab on pre/post nodes.
  - It is difficult for everyone to use it because of the license (Available if you bring your own license).
- We want a debug queue, because it waits for a long time to execute a job with a short elapsed time. We want to improve the waiting time for starting calculation after the status turned to RUN (we want to improve the efficiency of debugging).
  - Please use interactive job because it has higher priority.
  - One of the reasons why it takes time to execute a job is that it takes time to access files. Please try llio_transfer.
- We have the trouble that our program using lots of nodes randomly had SIGBUS during array access, and it is often the case that computational resources are not used effectively. This problem may have been fixed by firmware update on 7/16.
  - This problem is fixed by the firmware update.
- Without using node allocation method with mesh mode and appropriate communication algorithm, Highly parallel MPI communication performance is inferior to other supercomputers. Please let us know how we can improve the performance or if you have any plans for improvement.
  - If you arrange ranks on Tofu ignoring the communication characteristics of the application, communication time will increase due to route contention and increased hop count. Describe the parallelism, geometry, MPI algorithm and etc, and contact the support desk.
  - We are studying optimization methods for rank allocation
- The website was down during maintenance, so it was inconvenient to check the maintenance schedule.
  - The maintenance schedule is now displayed during maintenance.

Status of response to Program for Promoting Research on the Supercomputer Fugaku's hearing content (4/4)

- We want to see a document that deals with how to easily optimize a program to a minimum level with minimal technology for beginners. There are comprehensive documents on speedup, but we would like to see documents that summarize the key points in an easy-to-understand manner.
  - You can see following documents on Fugaku website.
    - Reference with slides of items that you should know in general from a programming perspective.
      Programming Guide (Programming common part)
    - Reference with slides on general programming techniques that affect app speed.
      Programming Guide (Programming common part)
    - Reference summarized in slides for each item from the viewpoint of CPU mechanism and characteristics.
      Programming Guide (Processors)
    - References that summarizes speed-up guidelines for each type of application, taking into account the different computational characteristics of actual applications.
      CPU performance tuning based on the type of application
  - When speeding up an application, it is usually necessary to take a holistic approach based on the various background information described above. In particular, a relatively concise section on basic speedup guidelines utilizing compiler functions is written in "Programming Guide (Programming common part) ". Please take a look at this first.
- Every time the new language environment is released, bugs appear. It is difficult to determine whether the bug is caused by my program or the system software, and it takes a lot of time to deal with.
  - We will work to improve and stabilize compilers, MPI and etc. If it works with other systems or compilers, please contact helpdesk. GCC and LLVM are available.
- Fugaku's Fujitsu compilers (frtpx/fccpx/mpifrtpx) can only handle old version of OMP (OMP Ver.4.5/5.0/5.1 are not available)
  - We will check with the vendor for the status and schedule of new features. Please contact helpdesk.

その他

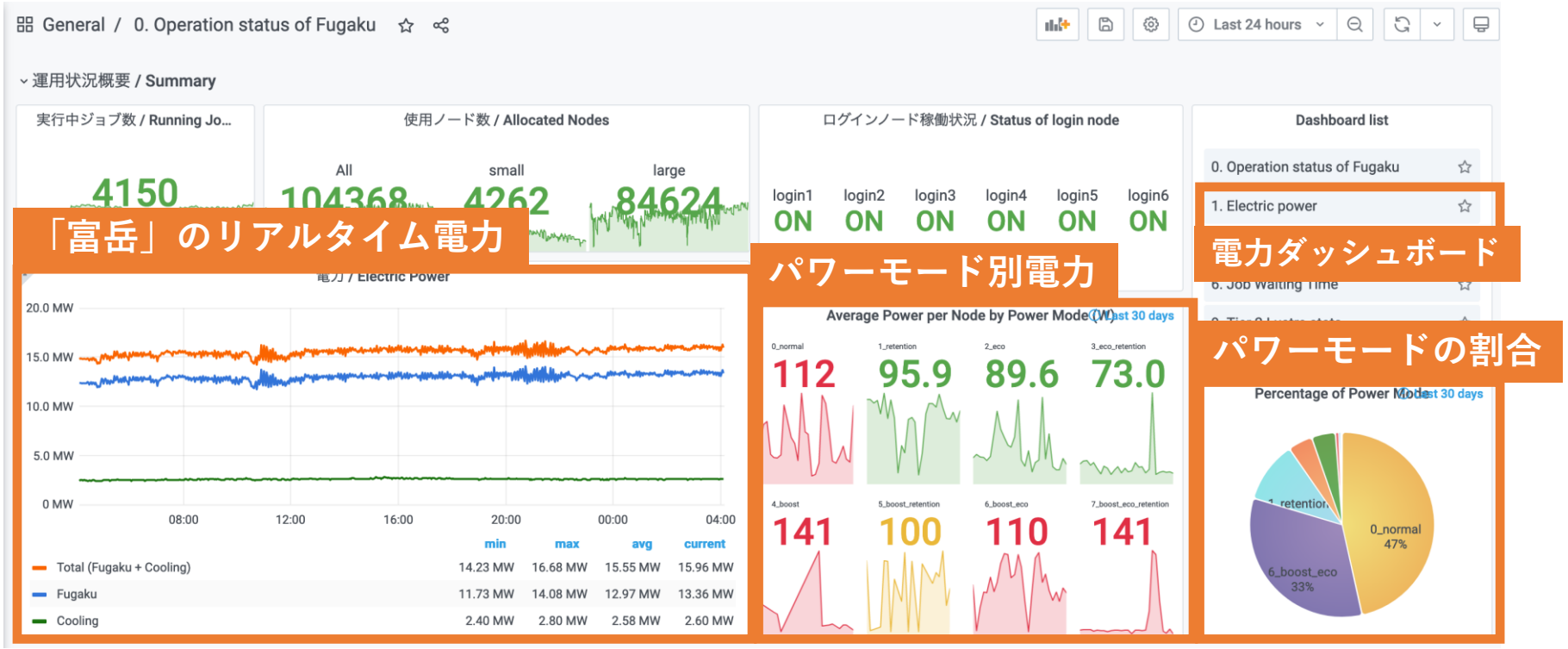
- 最大経過時間制限(24H)の拡大は可能か
  - 運用状況をみながら，拡大可能か引き続き検討します
- mpiexec同時実行数制限 (128) を緩和できないか
  - システムへの負荷から，同時実行数を制限しています.
  - 現状では，システムで同時実行数をコントロールすることが困難なため，同時実行数を緩和することは困難です

Others

- Can the maximum elapsed time limit (24H) be expanded?
  - We will continue to examine the feasibility of expansion while monitoring the operational status.
- Can the mpiexec concurrency limit (128) be relaxed?
  - The number of concurrent executions is limited due to the load on the system.
  - Currently, it is difficult to relax the number of concurrent executions because it is difficult to control the number of concurrent executions in the system

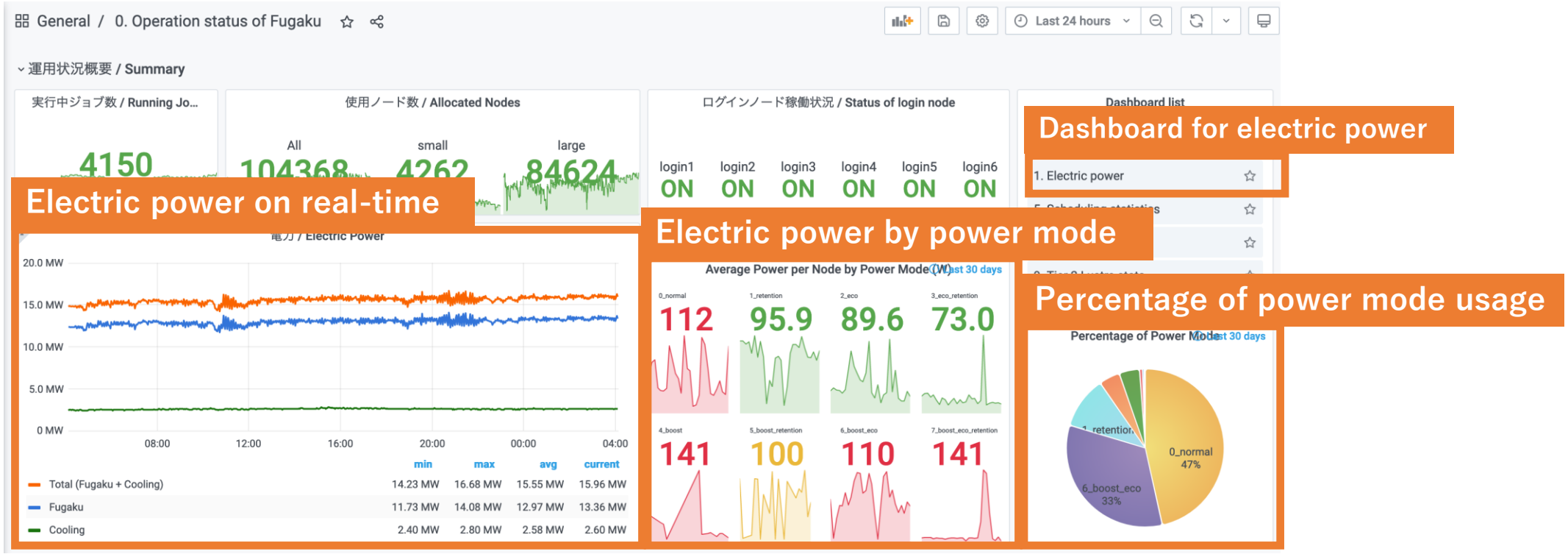
省エネルギー実行に関する統計情報

- **「富岳」運用ステータス**ページにて、ご協力頂いている省エネルギー実行に関する統計情報を掲載しています
  - https://status.fugaku.r-ccs.riken.jp/



Statistics on Energy Saving

- **Fugaku Operation Status** site shows statistics on energy saving operation.
  - https://status.fugaku.r-ccs.riken.jp/



large の電力設定変更について

10月の保守以降，largeのジョブ実行時にコア リテンションを指定することが可能となっています．この設定を，2022/10/25 10:00以降デフォルト設定に変更しました．

なお，4,608ノードを超えるジョブはシステム側でコア リテンションをオフにします．

変更内容：

- large の4,608ノード以下のジョブ実行時のパワーモード指定（ジョブ実行時のパワーノブ操作）のデフォルト設定を 「リテンション遷移しない」から「リテンション遷移を許可する」に変更しました
  - 4,608ノードの制限は，様子をみながら拡大する予定です
- ジョブ実行時のパワーモード指定（ジョブ実行時のパワーノブ操作）において，計算コアのリテンション状態遷移の可否（retention_state）が指定できます
- ジョブ実行中に，PowerAPIで計算コアのリテンション状態遷移の可否（retention_state）を操作できます

変更日時：

- 2022/10/25 10:00　※この日時より前に投入されたジョブは対象外です

設定変更による影響：

- リテンション遷移を許可することで，どの程度実行時間が長くなるかはジョブの処理内容により異なりますので，各ジョブでどのような影響があるかご確認ください．
- retention_state=1が有効な場合，コア上でプロセスが動作していない場合に，より低電力状態(リテンション状態)に遷移します．
  - コアがリテンション状態の場合，プログラムを実行するには実行状態に遷移する必要があります．遷移時間は3ms
  - コアの状態遷移に要する時間が発生するため，ジョブ全体の実行時間が長くなる場合があります
- retention_state=1が有効で，次の場合にリテンション状態に遷移します
  - wfi命令が発行された場合
    - wfi命令はcpu idle時(iowait、同期待ちなど)に呼ばれます．futex，mutexなどの同期処理はwfi命令が使用されているためリテンション状態に遷移します pthreadなどが該当します．
  - コア間ハードウェアバリアを使用した同期待ちはリテンション状態に遷移しません．コア間ハードウェアバリアは富士通言語環境の並列処理ライブラリで使用可能です
- 影響が大きい場合は， リテンション遷移を許可しない設定 retention_state=0 を指定してジョブを投入してください．

Changing power settings in resource group large

Core retention in large is available after October's maintenance, and core retention in large is default setting.

The system turns off core retention when the number of nodes is over 4,608.

Description：

- Default setting for the power mode specification during job execution (power knob operation during job execution) of jobs using less than 4,609 nodes in large changes from "No retention transition" to "Allow retention transition".
    - limitation of the number of nodes will be expanded in the future
- Users can specify whether the compute core is allowed to transition to the retention state (retention_state) in the power mode specification during job execution (power knob operation during job execution).
- Users can control whether the compute core is allowed to transition to the retention state (retention_state) in PowerAPI during job execution.

Change date：

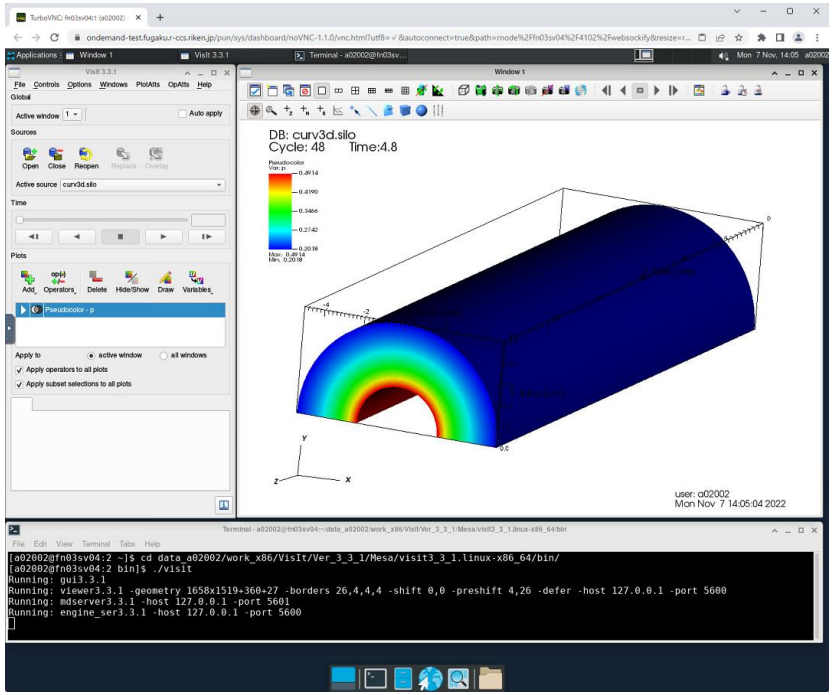
- 2022/10/25 10:00 (JST) (This change does not apply to jobs submitted before the change date)

Effect of changing settings：

- How long the execution time will be depends on the program by allowing the transition to the retention state. So please check the effect of the retention state by running the job.
- In case of setting retention_state=1, core goes into the retention state when no processes are running on the core.
    - When the retention state, core must go into the running state to execute program. The transition time is 3 ms.
    - The execution time may be longer because the time required for core state transitions.
- In case of retention_state=1, the core goes into the retention state by following conditions
    - When wfi instruction is issued
        - wfi instruction is called when core is idle mode (iowait, synchronization waits and etc). Core goes into the retention state because synchronization order like futex, mutex and etc uses wfi instruction. pthread and so on are also applicable.
    - Synchronization waits using an Inter-Core Hardware Barrier do not transition to the retention state. Inter-Core Hardware Barrier are available in parallel processing libraries in Fujitsu language environments.
- If the impact of the retention state is significant, please submit job by specifying retention_state=0 (not allowed transition to the retention state)

Open OnDemand

- 11月7日より提供開始（富岳ポータルから利用可能）
- Open OnDemandとは、Webブラウザを通して富岳の計算ノードを利用するためのWebポータル
- リモートデスクトップやJupyterLabなどのGUIアプリケーションを簡易に利用可能



- Available from November 7th in the Fugaku portal
- Open OnDemand is a web portal for using Fugaku's calculation nodes through a web browser
- Easy to use GUI applications such as remote desktop and JupyterLab

ホーム領域で発生するi-node超過(Disk quota exceeded)が出力される場合の回避方法

データ領域に十分なi-nodeの空きがあるにも関わらず，i-node超過が発生する場合があります.

これは利用されているアプリケーションやコマンドの一時ファイルの作成先が，ホーム領域に設定されており，ホーム領域のi-node超過が発生している可能性があります.

一時ファイルの作成先として利用される環境変数TMPDIRは, ログイン時にホーム領域が設定されます.

一時ファイルの作成先をホーム領域以外に設定することでi-node超過を回避することができます.
利用されているアプリケーションやコマンドの一時ファイル保存先を確認してください.

以下に$TMPDIRの設定変更例を示します.

設定例

$TMPDIR をデータ領域に設定します.
制限に余裕のあるデータ領域を選択することを推奨します.

```
$ mkdir -p /vol0n0m/data/your_group/your_id/tmp
$ export TMPDIR=/vol0n0m/data/your_group/your_id/tmp
```

ホーム領域で発生するi-node超過(Disk quota exceeded)が出力される場合の回避方法

.bashrcでTMPDIRを設定することで, ログイン時からホーム領域以外に$TMPDIRを設定することが可能です.
設定例

User specific aliases and functions
export TMPDIR=/vol0n0m/data/your_group/your_id/tmp

また, これまではモジュールパッケージを読み込んだ際もホーム領域が設定されておりましたが,2022年10月保守以降は, モジュールパッケージを読み込んでもTMPDIRは設定されません.

ジョブ投入時に，利用するデータ領域がホーム領域と異なるファイルシステム（volume）の場合は，環境変数 PJM_LLIO_GFSCACHE の指定が必要です.

詳細は「ファイルシステムの運用変更に伴いジョブ投入時にvolume指定」をご確認ください.

参考
１．[運用情報]ホーム領域の容量制限について
２．ホーム領域と異なるファイルシステム（volume）を利用するジョブについて

How to avoid output of inode exceeding (Disk quota exceeded) occurring in the home area

The inode might be exceeded even though there is enough space in the data area.

In this case, the temporary file creation destination for the used application or command is set in the home area,
and the i-node in the home area might be exceeded.

The TMPDIR environment variable used to create temporary files sets the home area at login.

You can avoid inode overage by setting the temporary file to a location other than the home area.
Check the temporary file storage location for the application or command being used.

The following is an example of changing the $TMPDIR setting.

Configuration Examples

Set $TMPDIR to the data area.
We recommend that you choose a data area that has more room for limits.

```
$ mkdir -p /vol0n0m/data/your_group/your_id/tmp
$ export TMPDIR=/vol0n0m/data/your_group/your_id/tmp
```

How to avoid output of inode exceeding (Disk quota exceeded) occurring in the home area

By setting TMPDIR in your .bashrc, you can set $TMPDIR outside of your home area at login.

```
------
# User specific aliases and functions
export TMPDIR=/vol0n0m/data/your_group/your_id/tmp
------
```

Previously, the home area was set when a module package was loaded.
After maintenance in October 2022, TMPDIR will not be set when the module package is loaded.

If the data area to be used is a file system (volume) different from the home area when the job is submitted,
the PJM_LLIO_GFSCACHE environment variable must be specified.

For details, see "Volume must be specified at job submission".

Reference
1. About the quota limit of the home area
2. Volume must be specified at job submission

ジョブスクリプト雛形作成コマンド

ジョブスクリプトの雛形を作成するコマンド(make_jobscript)は、主にファイルシステムに関連して必須である項目などを記載したスクリプトの雛形を出力します．グループ名を指定するオプション--gmaneが必須オプションのため，オプションを指定しない場合エラーとなります．

使用方法
make_jobscript [OPTION ...]
  --gname groupname
    ジョブ実行に使用するグループ名を指定してください。
  --distribute-common-file path_to_file1,path_to_file2,...
    すべての計算ノードから読み込まれるファイル（共通ファイル）を指定してください．
  --use-directory path_to_directory1,path_to_directory2,...
    ジョブで使用するディレクトリを指定してください．
  --use-spack
    Spack を使用する場合に指定してください．

指定されなかったオプションに関連する記述もコメントとして出力されます．
参考となるドキュメントのURLも出力されるので、詳細はそちらをご参照ください．

コマンドの使用方法の詳細については、下記のページをご参照ください。
5.1.4. ジョブスクリプトの雛形作成コマンド

Release of the command for creating template of job script

A command (make_jobscript) to create a template for a job script has been released.
The group specification option “--gname” is a required option, omitting it results in an error.

Usage:

```
make_jobscript [OPTION ...]
    --gname groupname
        Specify the group name to use in the job.
    --distribute-common-file path_to_file1,path_to_file2,...
        Specify the common files to be read from all compute nodes.
    --use-directory path_to_directory1,path_to_directory2,...
        Specify the directories used in the job.
    --use-spack
        Specify if you want to use spack.
```

Descriptions related to options not specified are also output as comments.
The URL of the reference document is also output, so please refer to it for details.

For more information about using the command, see the following pages:
5.1.4. The command for creating template of job script

ログインノード利用時のターミナル表示について

2023/01/31 19:00より、ログイン時に以下の情報メッセージがターミナルに出力されるようになりました。

[低優先度ジョブの利用案内]
低優先度ジョブが利用可能となった課題を表示します。
課題名と低優先度ジョブのリソースグループへの案内文を表示します。
課題の割り当て資源量の98%以上を利用した場合に、低優先度ジョブが利用可能となります。

Terminal display when using the login node

Beginning at 1/31 13:00, upon login, the following informational message in the terminal.

[Usage information for low priority jobs]
Displays the projects for which low priority jobs are available.
Displays the Project Name and the Invitation Messages to Resource Groups for Low Priority Jobs.
Projects have used more than 98% of the allocated resources can use low priority jobs.

OSS利用ログ採取

- Spackを通じて利用されたOSSのログ採取を開始します。
- 具体的には、`spack load`で指定されたパッケージ名を含むログが、以下のディレクトリに生成されます。

  `/vol0004/data/applog/YYYYMMDD/JOBID_SUBJOBID/`

- ログ採取を希望されない方は、以下の設定をお願いします。

1. `/vol0004/apps/oss/spack/etc/spack/config.yaml`を、`~/.spack`にコピー。
2. `~/.spack/config.yaml`の最後の行を、「`logdir: ''`」に変更。

- 開始日： 7/19(火) 9:00

※ 事前でも問題ありません。

OSS logging

- We'll start logging the usage of OSS via Spack.
- A file which contains the names of the packages you used with "spack load" is generated in:
  /vol0004/data/applog/YYYYMMDD/JOBID_SUBJOBID/
- Please configure it as follows if you don't prefer the logging.

1. Copy /vol0004/apps/oss/spack/etc/spack/config.yaml to ~/.spack
2. Modify the last line of ~/.spack/config.yaml into logdir: ''

- Start date: 9:00am on July 19th

Note: you can do this before the start date.

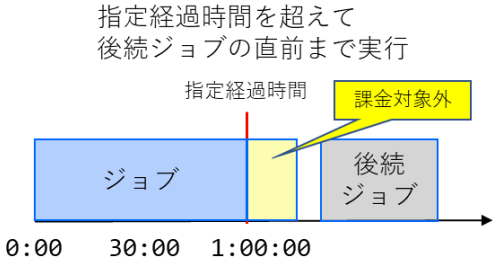
最低実行時間を指定したジョブ実行

- 富岳では，一部リソースグループを除き，経過時間指定を範囲で指定することができます．通常の実行では，指定経過時間になるとジョブが終了しますが，この機能を使うと指定経過時間後もノードが空いていれば引き続きジョブを実行することができます．
  ジョブ投入時に指定した最低実行時間（min_limit）以降に使用された計算資源は課金対象外となります．
  - スケジューリングは最低実行時間で行われるため，短く指定することでスケジューリングされやすくなるというメリットもあります．

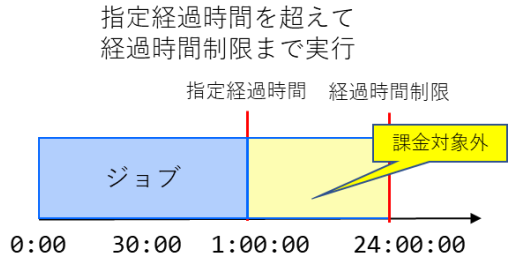
指定方法

```
{-L | --rsc-list} elapse=min_limit- (*) min_limit の後ろにハイフンが必要
{-L | --rsc-list} elapse=min_limit-max_limit
```

- 経過時間がmin_limitに達しても，経過時間が最長でmax_limitに達するまで実行を継続し，min_limit以降は課金対象外となります．
  - max_limitは経過時間上限+72Hまで指定可能
  - max_limitを指定しない場合(例: elapse=30:00-)は，max_limitは経過時間上限+72Hになります
- ただし，経過時間がmin_limitを超えた場合に，後続のジョブが割り当てられた場合，経過時間がmax_limit未満でもジョブは強制終了されます．そのため，ジョブ側でチェックポイントリスタートなどを利用してジョブの打ち切りに対応する必要があります．
  - チェックポイントリスタートには，VelocなどのVelocの利用を推奨します．
    VeloC 利用ガイド [https://www.fugaku.r-ccs.riken.jp/doc_root/ja/user_guides/VeloC_Guide/]
    VeloC Webpage [https://veloc.readthedocs.io/en/latest/toc.html]



後続ジョブが割り当てられた場合



後続ジョブがない場合

（参考）：ジョブ運用ソフトウェア　エンドユーザ向けガイド - 2.3.2.3 ジョブの経過時間制限値の指定

Jobs specifying the minimum elapsed time limit value

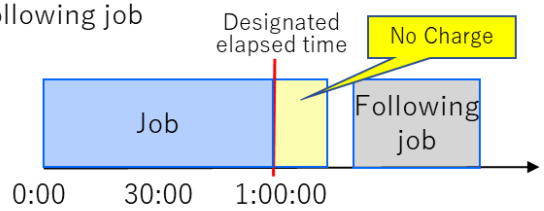
- You can use the following formats to specify the elapsed time limit value for the job on Fugaku except some resource groups. On normal job execution, jobs will terminate to reach the elapsed time limit. But by specifying the minimum elapsed time limit value (min_limit), your job can run after the minimum elapsed time limit unless other jobs will not be scheduled on nodes used by your job.
  The compute resource used by your job after the minimum elapsed time limit will no longer be charged.
  - The job scheduling time is decided based on the minimum elapsed time limit, so short specified elapsed time will make short waiting time to be executed.

How to specify

{-L | --rsc-list} elapse=min_limit- (*) Hyphen required after min_limit {-L | --rsc-list} elapse=min_limit-max_limit

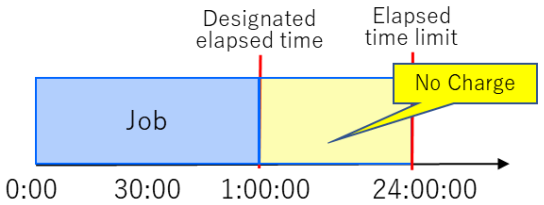
- After the elapsed time reaches min_limit, job execution can continue to run until the elapsed time reaches max_limit, and used compute resource over min_limit will be no charge.
  - max_limit can be specified for up to “the max. amount of elapse time limit +72H”
  - If max_limit is not specified (e.g., elapse=30:00-), “the max. amount of elapse time limit +72H” is applied to max_limit.
- Howethe elapsed ver, even when time is less than max_limit, the job is forcibly terminated when other job is allocated on nodes used by your job. So you need to be prepared for the job termination using check-point restart or something like.
  - We recommend using VeloC as a check-point restart.
    VeloC User Guide [https://www.fugaku.r-ccs.riken.jp/doc_root/en/user_guides/VeloC_Guide/]
    VeloC Webpage [https://veloc.readthedocs.io/en/latest/toc.html]

Job execution exceeds the designated elapsed time and terminates immediately before the following job



Following job is allocated

Executes up to the elapsed time limit after exceeding the designated elapsed time



No following job

ref: Job Operation Software End-user's Guide - 2.3.2.3 Specifying the elapsed time limit value for a job

MATLAB

- 富岳フロントエンドサーバー向けに，インストール済みのMATLAB環境 (Intel版) を用意しています.
- 富岳ユーザが利用するには，ユーザ自身が所有しているライセンスをログインノード，プリポスト環境に持ち込み，アクティベーションする必要があります.
- 利用申請および利用方法
  - https://www.fugaku.r-ccs.riken.jp/doc_root/ja/user_guides/Commercial_Software/matlab.html
- 留意事項
  - R-CCSではライセンスは提供しません.
  - ライセンスは Individual license に対応しています.
  - アプリケーションの利用に関する質問・サポートは，MathWorks社のヘルプセンターにお問い合わせ下さい.
  - 富岳アカウント申請時の情報に基づき，非居住者である場合は，利用の可否を判断させていただきます. また，安全保障輸出管理に基づく審査のための審査票の提出が必要になります．その場合追加の情報提供を依頼する場合があります.

MATLAB

- R-CCS provides a pre-installed MATLAB (Intel version) environment for the front-end servers of Fugaku.
- To use the MATLAB environment, a Fugaku user must only bring his/her own license to the login nodes or the pre/post environment and activates the license by him/herself.
- Application method and how to use
  - https://www.fugaku.r-ccs.riken.jp/doc_root/en/user_guides/Commercial_Software/matlab.html
- Note
  - R-CCS does not provide a license.
  - “Individual license” is only valid for activation.
  - For questions or assistance in using the application, please contact the MathWorks help center.
  - If you are a non-resident, we need to submit a non-resident check form to abide by compliance. In such cases, additional information may be requested. Finally, we will approve/deny your application based on the fact.

プリポスト環境

- プリポスト環境のジョブ混雑状況の改善の試みとして、新たに4つのキュー(gpu1, gpu2, mem1, mem2)を試験的に追加しました。

キュー名 (Partition Name)	ノード名 (Nodes)	デフォルト経過時間 (DefaultTime)	最大経過時間 (MaxTime)	ジョブあたりデフォルトCPU数 (cpus-per-task)	ジョブあたり最大CPU数 (MaxTRESPerJob=cpu)	ジョブあたりデフォルトメモリ量 (DefMemPerNode)	ジョブあたり最大メモリ量 (MaxMemPerNode)	ジョブあたり最大ジョブ実行数 (MaxJobsPerUser)	ジョブあたり最大ノード数 (MaxNodes)
gpu1	pps[01-06]	30min	3h	1	72	2638MB	190000MB	5	1
gpu2	pps[07-08]	30min	24h	1	36	2638MB	95000MB	1	1
mem1	ppm01	30min	3h	1	224	27636MB	6190568MB	5	1
mem2	ppm02	30min	24h	1	56	27636MB	1547642MB	1	1

- 新しく追加したキューは、従来のppsq, ppmqより優先的にジョブが実行されますので、今後ジョブ投入される場合は活用をご検討ください。
- また今後、リソース量を指定せずにジョブ投入した場合に、デフォルト値が設定されます。ジョブ投入時に要求CPU数、メモリ量、経過時間をジョブスクリプトもしくはコマンドオプションで指定することをお願い致します。
- なお、Slurmのキュー設定につきましては、今後利用状況に応じてパラメタ調整を予定しております。また現在、ppsq, ppmqの使用は非推奨となっており、5月中に廃止することを予定しています。ご協力よろしくお願い致します。

Pre/Post Environment

- To improve the congestion in the Slurm queues (partitions), we added four new queues (gpu1, gpu2, mem1, mem2).

キュー名 (Partition Name)	ノード名 (Nodes)	デフォルト経過時間 (DefaultTime)	最大経過時間 (MaxTime)	ジョブあたりデフォルトCPU数 (cpus-per-task)	ジョブあたり最大CPU数 (MaxTRESPerJob=cpu)	ジョブあたりデフォルトメモリ量 (DefMemPerNode)	ジョブあたり最大メモリ量 (MaxMemPerNode)	ジョブあたり最大ジョブ実行数 (MaxJobsPerUser)	ジョブあたり最大ノード数 (MaxNodes)
gpu1	pps[01-06]	30min	3h	1	72	2638MB	190000MB	5	1
gpu2	pps[07-08]	30min	24h	1	36	2638MB	95000MB	1	1
mem1	ppm01	30min	3h	1	224	27636MB	6190568MB	5	1
mem2	ppm02	30min	24h	1	56	27636MB	1547642MB	1	1

- The newly added queues have priority over the existing ppsq and ppmq, please consider using them when submitting jobs.
- Also, when a job is submitted without specifying the amount of resources, default values will be applied. Please specify the number of required CPUs, amount of memory, and elapsed time in the job script or command options when submitting a job.
- We plan to optimize the parameters of the Slurm configurations based on the actual usage. The queues (ppsq and ppmq) are deprecated and will be discontinued by the end of next May. Thank you for your cooperation.

可視化処理環境について

- Tecplot 360
  - ISV可視化ソフトウェア
    - CFD Visualization &Analysis Tools（https://www.tecplot.com/）
  - 「富岳」利用準備課題ヒアリングでの導入要望
  - ライセンス

https://www.hulinks.co.jp/software/da_visual/tecplot/price

製品名	税込価格
年間ライセンス	
Tecplot 360 Network Single Facility 年間ライセンス 1セッション DL版	￥ 816,200
Tecplot 360 Network Multi Facility 年間ライセンス 1セッション DL版	￥ 1,611,500
無期限ライセンス	
Tecplot 360 2021 Single DL版	￥ 908,600
Tecplot 360 2021 Network Single Facility 1セッション DL版	￥ 1,815,000
Tecplot 360 2021 Network Multi Facility 1セッション DL版	￥ 3,630,000

個人利用

不特定多数の
第三者ユーザーへ提供：
通常価格 x 1.5倍
x 同時利用セッション

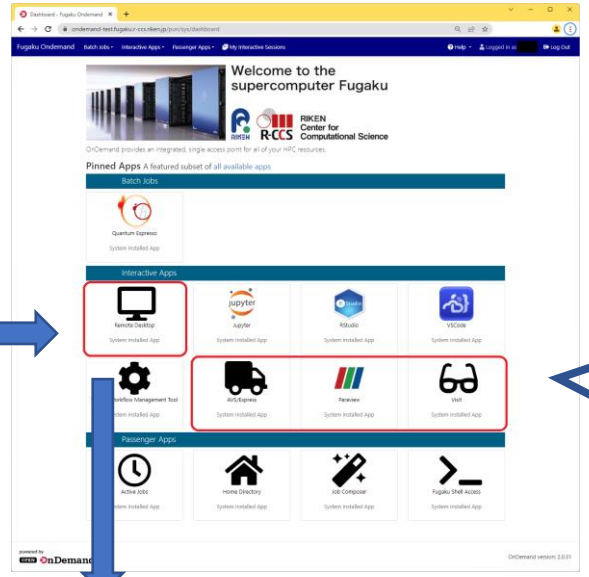
コスト面での検討を行った結果、現時点ではユーザー向けへの提供が難しいと判断し、現時点ではプリポストノードでの利用は個人で契約したライセンスの利用をお願いしたい。個人で導入する利用者が多い場合は理研側で整備することも検討する。

可視化処理環境について

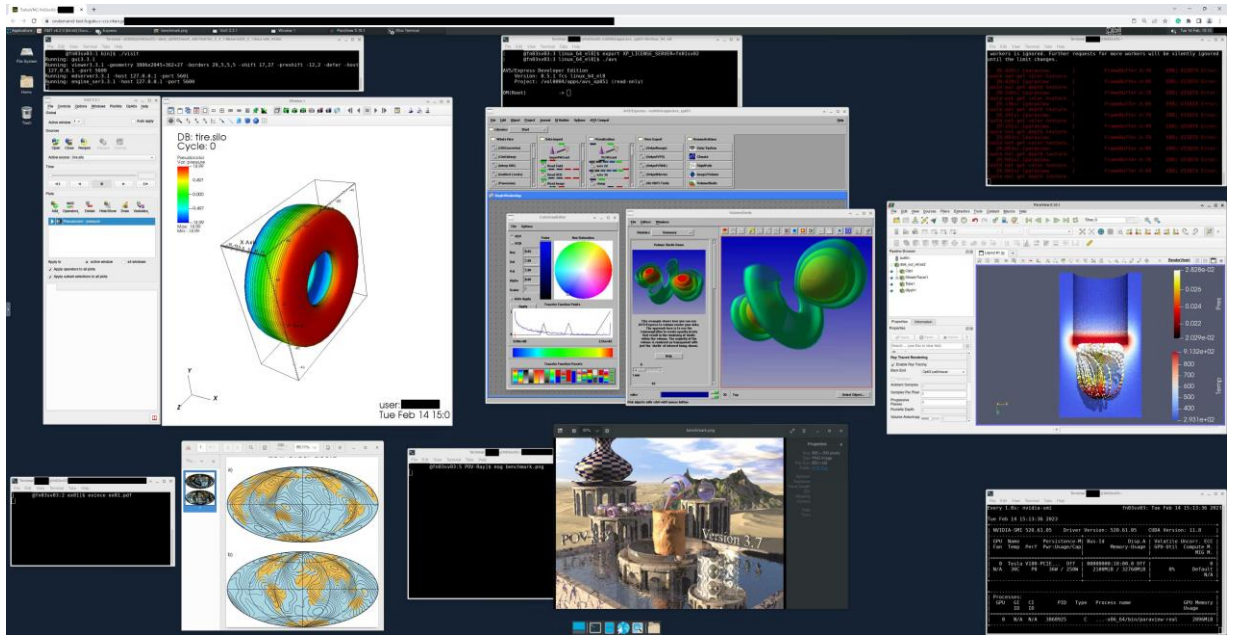
- Fugaku OnDemand（プリポスト環境）
  - リモートデスクトップ
    - ParaView
    - VisIt
    - AVS/Express
  - インテラクティブ・セッション

OnDemand側にインストール

```
$ paraview
$ visit
$ avs
```



可視化アプリケーションのインテラクティブ・セッションを起動

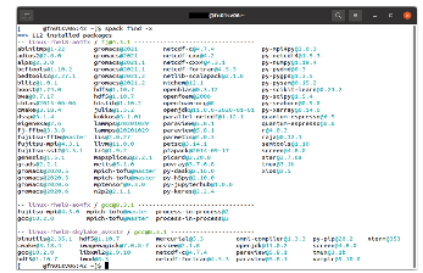


Xfce リモートデスクトップ

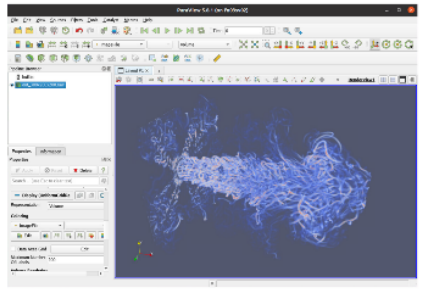
詳細は富岳ポータル掲載の可視化 ガイドを参照
https://www.fugaku.r-ccs.riken.jp/doc_root/ja/user_guides/viz/

可視化処理環境について

- 可視化ガイドの情報更新
  - Spackによるアップデート
  - ParaView Pythonモジュール
- プリポストノード
- 計算ノード
- 利用方法
  - CLI
  - GUI
  - Client/Server
  - VNC（GNOME）



CLI (Command Line Interface)



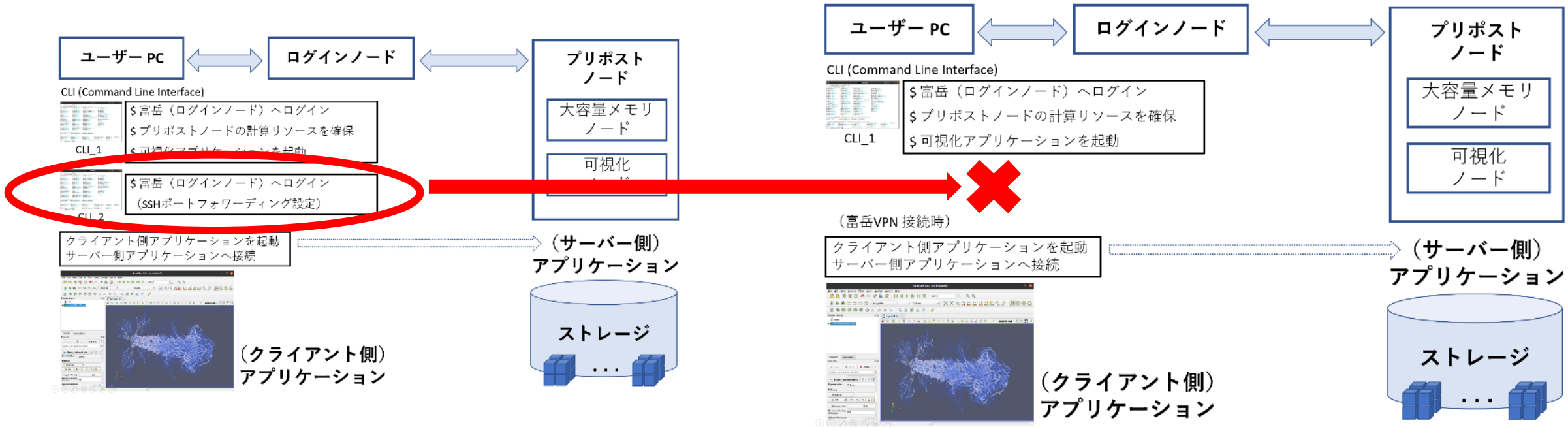
GUI (Graphical User Interface

Data courtesy:
Former Advanced Visualization Research Team (RIKEN R-CCS)

詳細は富岳ウェブサイト掲載の可視化 ガイドを参照
https://www.fugaku.r-ccs.riken.jp/doc_root/ja/user_guides/viz/

可視化処理環境について

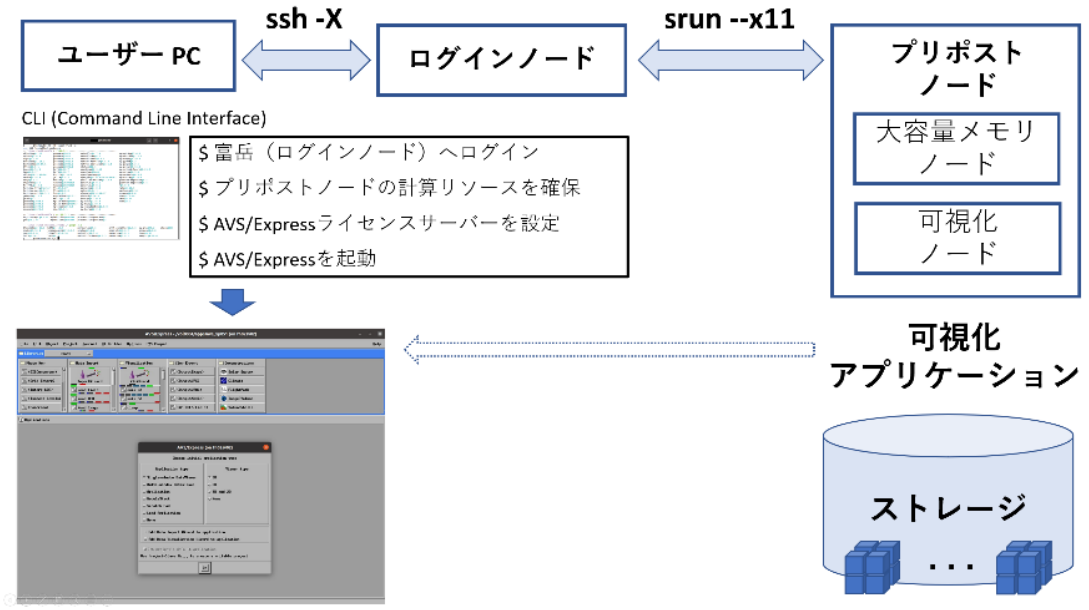
- SSHポートフォワーディング　➔　富岳VPN接続
  - Client/Server型アプリケーション
  - VNC（GNOMEデスクトップ）

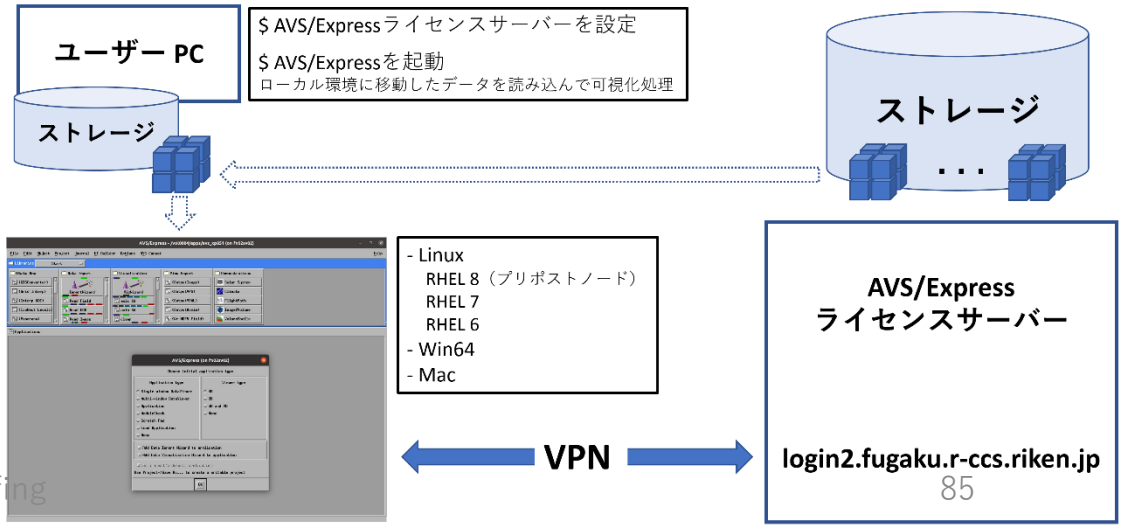


詳細は富岳ウェブサイト掲載の可視化 ガイドを参照
https://www.fugaku.r-ccs.riken.jp/doc_root/ja/user_guides/viz/

可視化処理環境について

- AVS/Express Developer
  - ISV汎用可視化ソフトウェア
  - ライセンス・サーバー
    - login2.fugaku.r-ccs.riken.jp
- プリポスト・ノード
  - /vol0004/apps/avs_xp851
- ローカルPC（富岳VPN接続）
  - Linux, Windows, Mac OSX





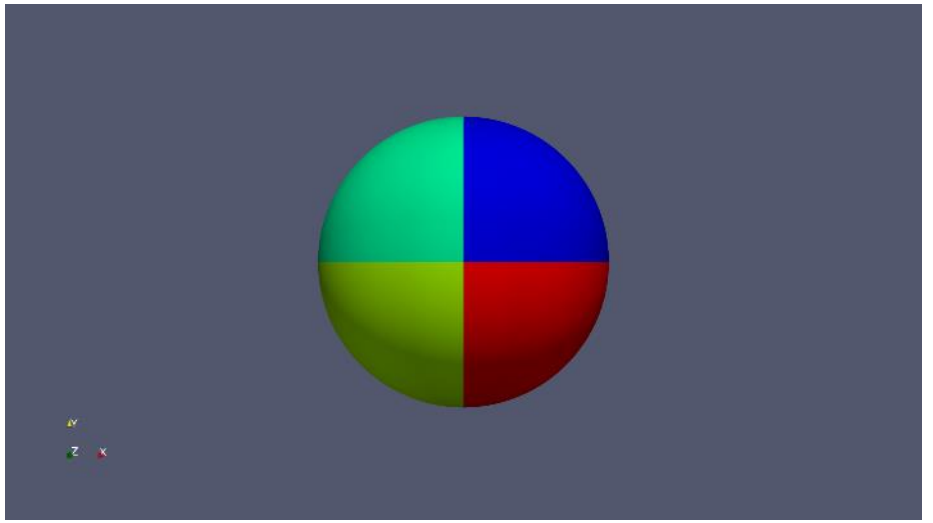
詳細は富岳ウエブサイト掲載の可視化 ガイドを参照
https://www.fugaku.r-ccs.riken.jp/doc_root/ja/user_guides/viz/

可視化処理環境について

- ParaView Pythonモジュール
  - CLI での利用
  - プリポストノード
    - pvbatch
    - pvpython
  - 計算ノード
    - pvpython

pvbatch利用例：
「計算ノード」$ spack load paraview@5.8.1%fj@4.6.1
「計算ノード」$ mpirun –n 4 pvbatch sample_script.sh

出力：画像ファイル



詳細は富岳ウェブサイト掲載の可視化 ガイドを参照
https://www.fugaku.r-ccs.riken.jp/doc_root/ja/user_guides/viz/

可視化処理環境について

- Generic Mapping Tools (GMT)
  - 「富岳」利用準備課題ヒアリングでの導入要望
    - 「SPACKにGMTを導入いただけると助かります。」
      - 8/15にSpack Public Instanceへ導入済み
  - ビルドオプション（デフォルト）
    - オン：+blas+lapack
    - オフ：~docs~ffmpeg~fftw~gdal~ghostscript~glib~graphicsmagick ~pcre
  - プリポスト環境
    - バージョン：gmt@6.2.0
    - コンパイラ：gcc@11.2.0
      - PDF出力は可能
  - 計算ノード
    - バージョン：gmt@6.2.0
    - コンパイラ：fj@4.8.0
      - PDF出力が行えない

可視化処理環境についてのご要望や　　ご質問は富岳サポートサイト(Zendesk)にてお気軽にご連絡お願い致します。

Visualization Environment

- Tecplot 360
  - ISV Visualization Software
    - CFD Visualization & Analysis Tools（https://www.tecplot.com/）
  - Users' request from a Hearing Survey
  - License

https://www.hulinks.co.jp/software/da_visual/tecplot/price

製品名	税込価格
年間ライセンス	
Tecplot 360 Network Single Facility 年間ライセンス 1セッション DL版	¥ 816,200
Tecplot 360 Network Multi Facility 年間ライセンス 1セッション DL版	¥ 1,611,500
無期限ライセンス	
Tecplot 360 2021 Single DL版	¥ 908,600
Tecplot 360 2021 Network Single Facility 1セッション DL版	¥ 1,815,000
Tecplot 360 2021 Network Multi Facility 1セッション DL版	¥ 3,630,000

Single User (Local Use)

For providing to unspecified users:
1.5 x Regular Price x
Sessions at same time

Taking into consideration the cost, at this time it seems difficult to provide this software to the users. Therefore, we kindly ask to use their own Tecplot license on the Pre/Post Processing Nodes. However, we will still consider to install it on the RIKEN side, depending on the number of users using their own licenses.

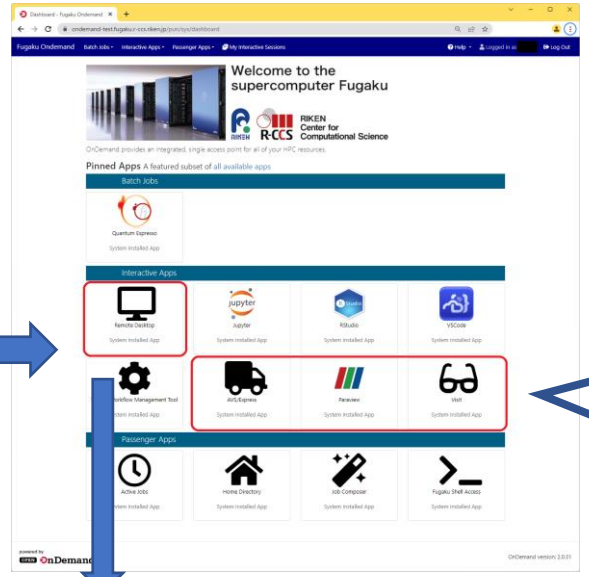
Visualization Environment

- Fugaku OnDemand (Pre/Post)
  - Remote Desktop
    - ParaView
    - VisIt
    - AVS/Express

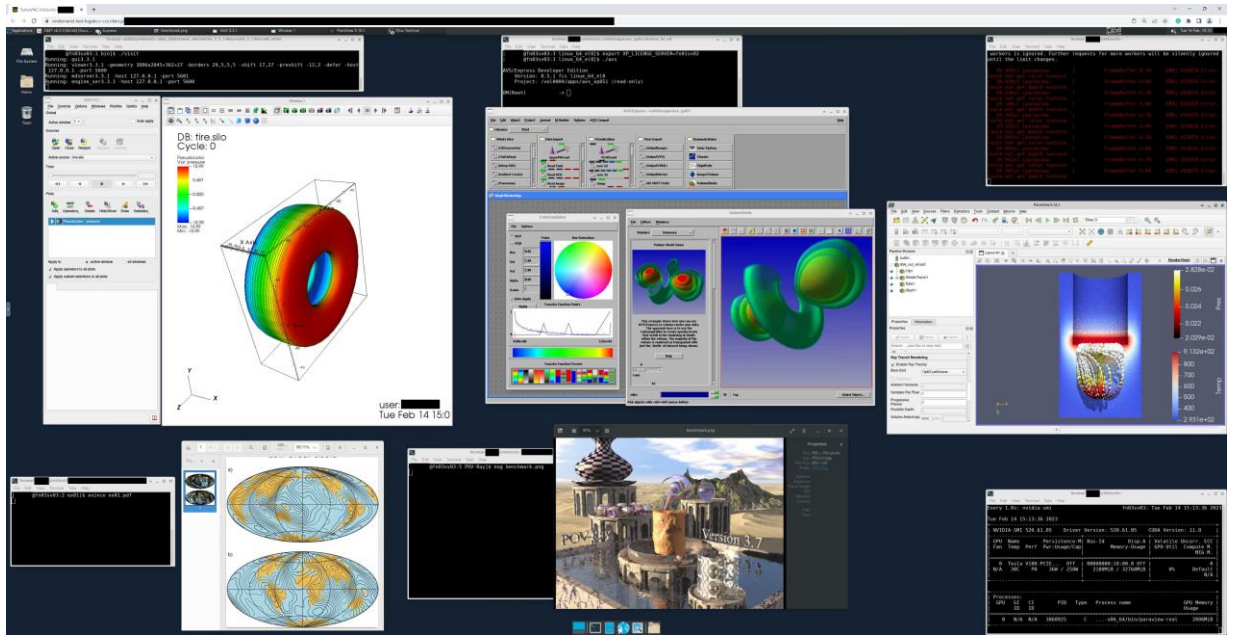
Installed on the OnDemand side

```
$ paraview
$ visit
$ avs
```

  - Interactive Session



Start an interactive session for the pre-installed visualization applications



Xfce-based Remote Desktop

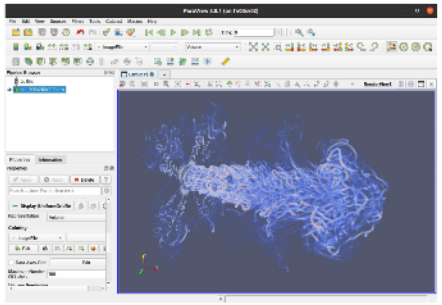
See the “Visualization User Guide” on the Fugaku website for more details.
https://www.fugaku.r-ccs.riken.jp/doc_root/en/user_guides/viz/

Visualization Environment

- Information update of “Visualization User Guide”
  - Spack package update
  - ParaView Python Module
- Pre/Post Nodes
- Compute Nodes
- Usage Scenarios
  - CLI
  - GUI
  - Client/Server
  - VNC（GNOME）

CLI (Command Line Interface)

GUI (Graphical User Interface

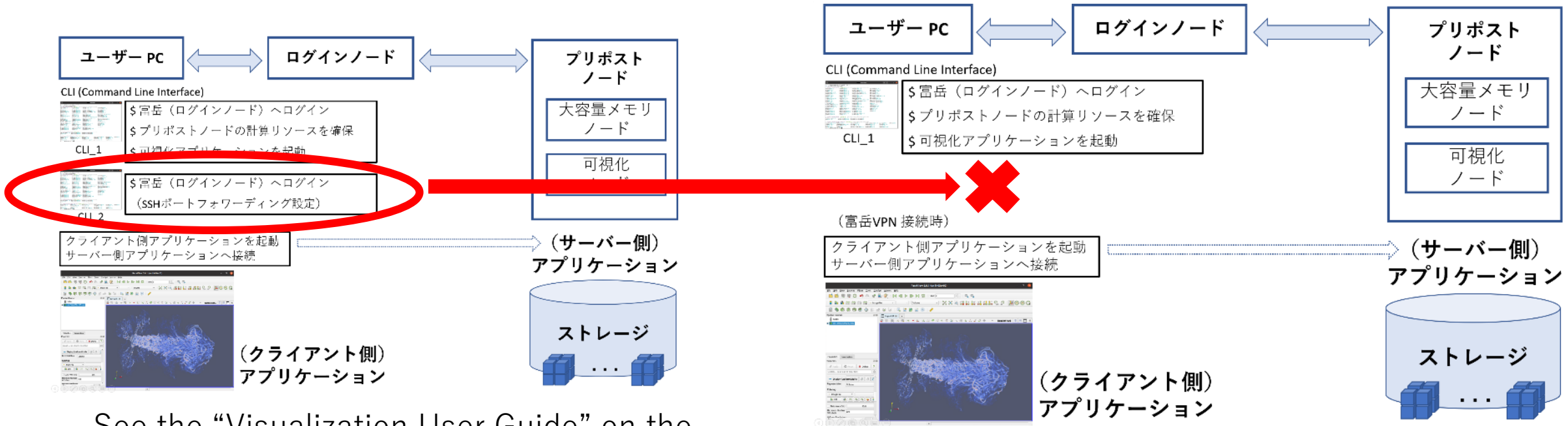


Data courtesy:
Former Advanced Visualization Research Team (RIKEN R-CCS)

See the “Visualization User Guide” on the Fugaku website for more details.
https://www.fugaku.r-ccs.riken.jp/doc_root/en/user_guides/viz/

Visualization Environment

- SSH Port Forwarding ➔ Fugaku VPN Service
  - Client/Server Application
  - VNC (GNOME Desktop)

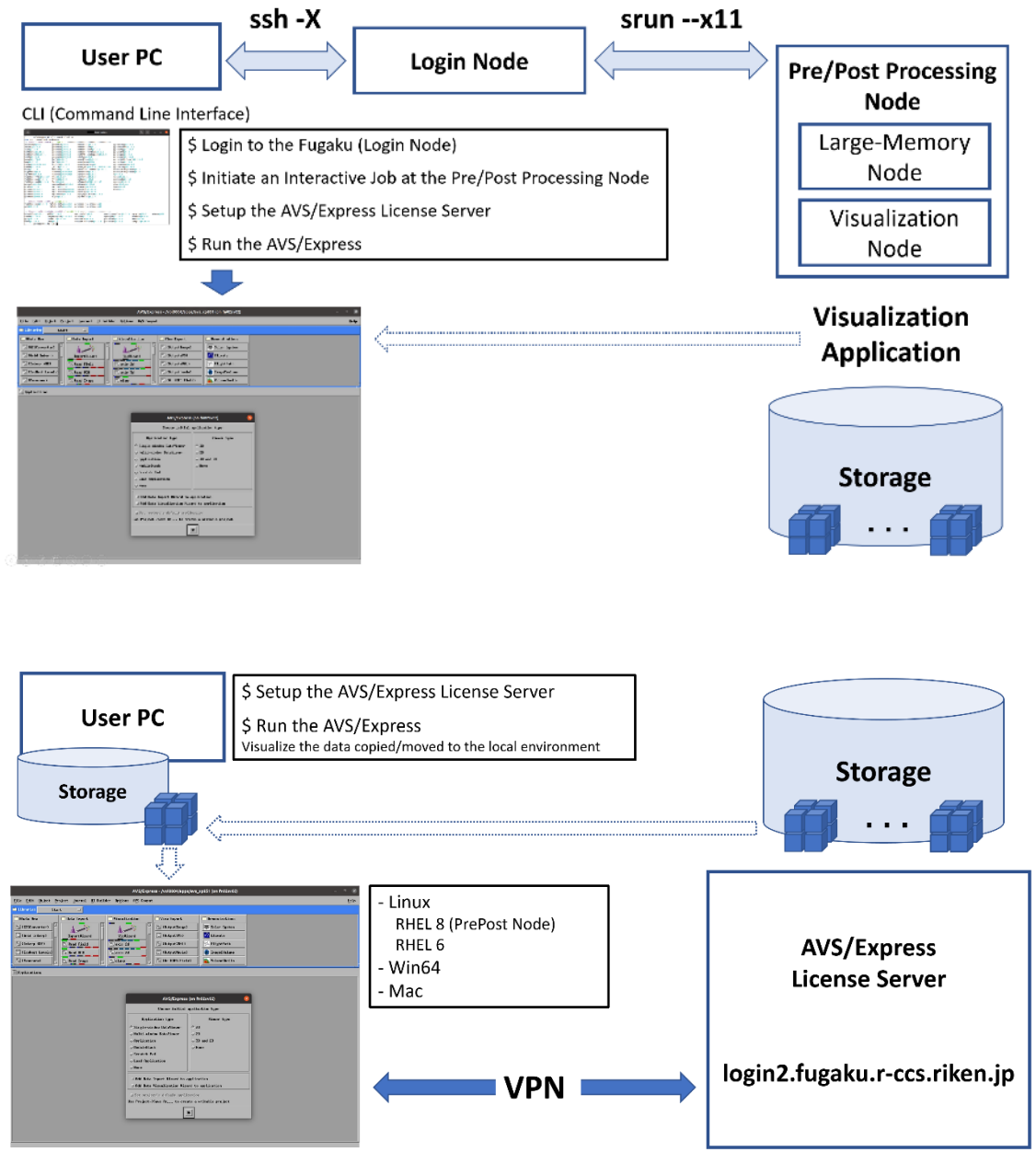


See the “Visualization User Guide” on the Fugaku website for more details.
https://www.fugaku.r-ccs.riken.jp/doc_root/en/user_guides/viz/

Visualization Environment

- AVS/Express Developer
  - ISV Visualization Application
  - License Server
    - login2.fugaku.r-ccs.riken.jp
- Pre/Post Processing Nodes
  - /vol0004/apps/avs_xp851
- Local PC
  - Fugaku VPN Service
  - Linux, Windows, Mac OSX

See the “Visualization User Guide” on the Fugaku website for more details.
https://www.fugaku.r-ccs.riken.jp/doc_root/en/user_guides/viz/



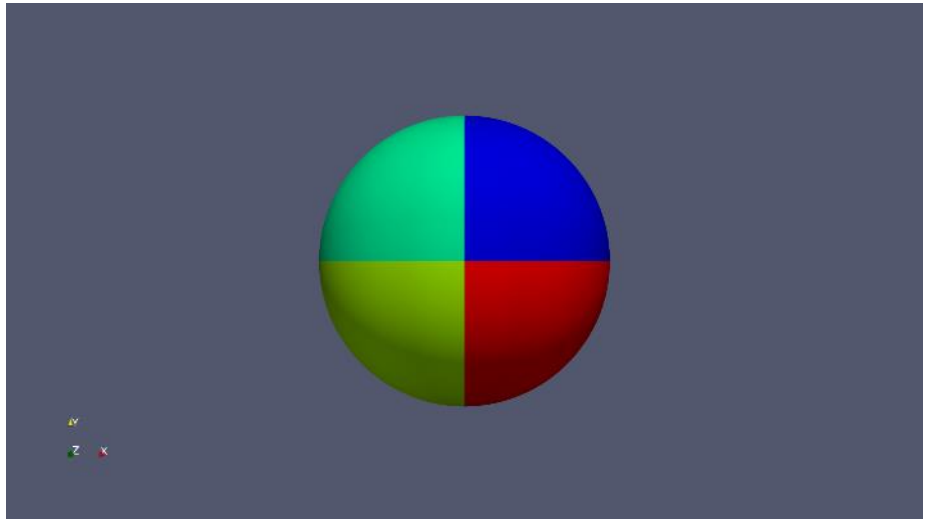
Visualization Environment

- ParaView Python Module
  - Usage via CLI
  - Pre/Post processing Nodes
    - pvbatch
    - pvpython
  - Compute Nodes
    - pvpython

Example usage of pvbatch：

```
[Compute]$ spack load paraview@5.8.1%fj@4.6.1
[Compute]$ mpirun –n 4 pvbatch sample_script.sh
```

Output: Image File



See the “Visualization User Guide” on the Fugaku website for more details.
https://www.fugaku.r-ccs.riken.jp/doc_root/en/user_guides/viz/

Visualization Environment

- Generic Mapping Tools (GMT)
  - Users' request from a Hearing Survey
    - To provide GMT via SPACK
      - Installed on Spack Public Instance: Aug. 15
  - Build Option (Default)
    - ON : +blas+lapack
    - OFF: ~docs~ffmpeg~fftw~gdal~ghostscript~glib~graphicsmagick ~pcre
  - Pre/Post Environment
    - Version : gmt@6.2.0
    - Compiler: gcc@11.2.0
      - PDF output is available
  - Compute Node
    - Version : gmt@6.2.0
    - Compiler: fj@4.8.0
      - PDF output is not working

Please, fee free to contact via Fugaku Support Site (Zendesk) regarding the requests and questions on visualization environment.

計算ノードのファイルシステム構成変更について (1/2)

- 9月の保守作業で，計算ノードのファイルシステムのマウント構成を変更しました
- この構成変更に伴い，既存ジョブの実行時間が増減，または実行不可となる場合があります．変更内容を確認し，ジョブの見直しをお願いします

- 構成変更の内容
  - 第2階層への直接アクセスのパスを削除し，第2階層ストレージキャッシュのパスのみに変更されています

	第2階層ストレージ（FEFS）のパス	第2階層ストレージのキャッシュのパス名
変更前	/vol000x/<groupname>	/vol000x_cache/<groupname>
変更後	deleted	/vol000x/<groupname>

  - ジョブから /home/<groupname>，/data/<groupname>にアクセスした場合も，第2階層ストレージのキャッシュ経由でのアクセスとなります

- “2ndfs”の追加
  - ジョブ（計算ノード）から直接アクセス可能なファイルシステムです
  - 全てのグループが利用可能です

計算ノードのファイルシステム構成変更について (2/2)

- [既にLLIOを利用されている方へ]
  - アクセスパス /vol000x_cache は削除されました
  - llio_transfer で指定するパスは "_cache" なしのパスで指定してください

```
llio_transfer ./a.out
mpiexec -stdout-proc ./output.%j/%/1000r/stdout -stderr-proc ./output.%j/%/1000r/stderr ./a.out
llio_transfer --purge ./a.out
```

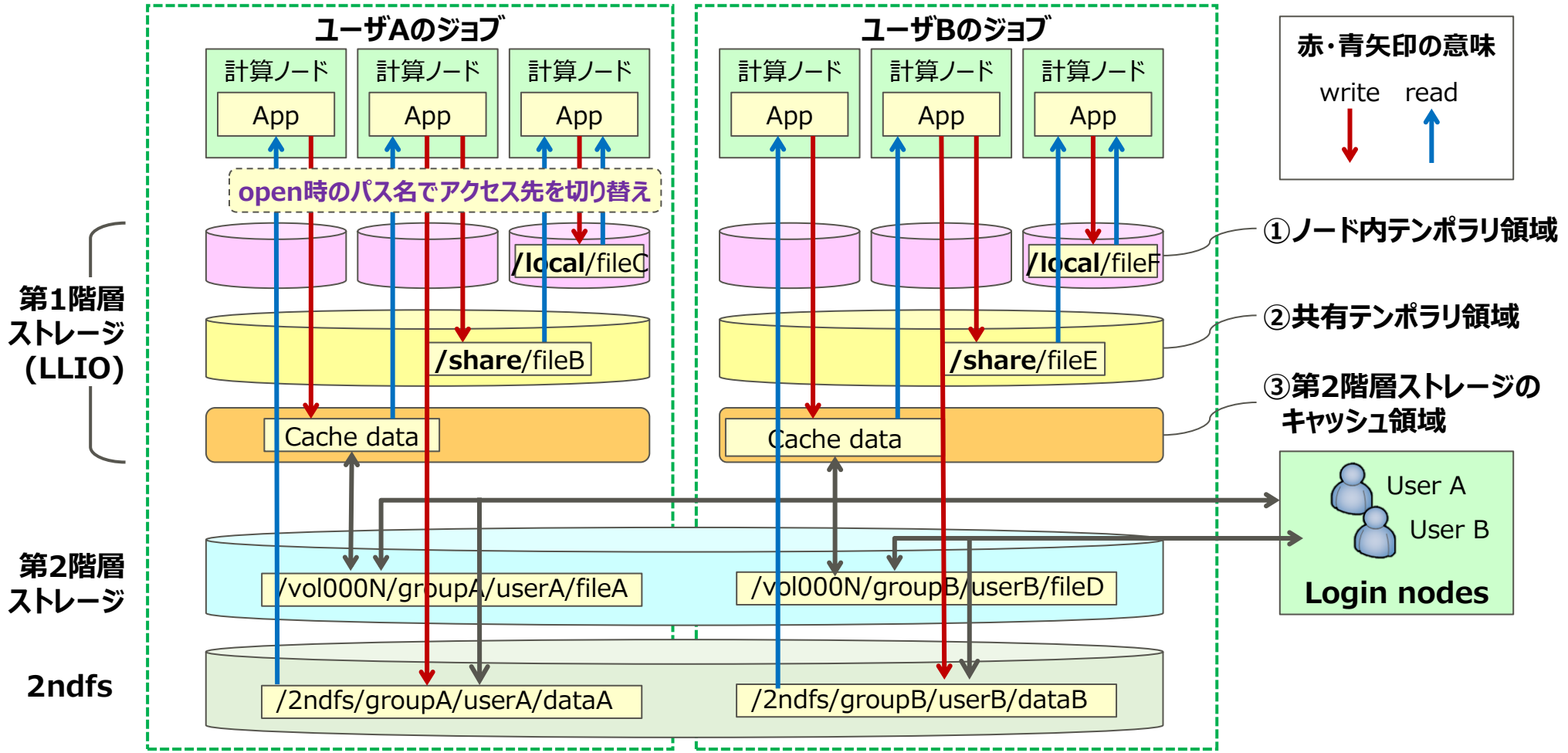
- [FEFSのみを利用されている方へ]
  - データ領域に対するアクセスは全て第2階層ストレージキャッシュ（LLIO）経由です（“2ndfs”は除く）
  - [利用手引書]で利用方法や留意事項を確認して利用してください
  - a.out のような各プロセスからアクセスするファイルは通常アクセスでは時間がかかる場合があるので， llio_transferを使ってください

- [ジョブ投入オプションのdefault変更]
  - ジョブ投入時のオプションの defaultが変更されています
  - llio-sio-read-cache
    - default を on に変更．第2階層ストレージから計算ノードへ読み込んだファイルを第1階層ストレージにキャッシュします
  - llio-async-close
    - default を off に変更．第1階層ストレージおよび第2階層ストレージ上のファイルのクローズを同期クローズにします
  - mpiexec-std-emptyfile
    - default を on に変更．空ファイルが作成されなくなります
  - llio-stripe-count
    - default を 1 から 24 に変更し，ジョブ内共有ファイル，第2階層ストレージキャッシュへのアクセスを並列分散アクセスにします
    - これによってジョブが同時にアクセス可能なファイル数、同時に open 可能なファイル数がストライプ分でカウントされることからLLIOに関する利用手引書の「留意事項」の制限に合致する場合があります
    - 留意事項の制限に該当しLLIOを利用できない場合は，2ndfs の利用やストライプカウントを下げることを検討してください

2ndfsについて (1/3)

- 2ndfsは，ジョブから第2階層ストレージに直接アクセス可能なファイルシステムです
  - 保守前の第2階層ストレージと同じ
- LLIOを利用したジョブ実行に不具合が出る場合などに利用してください
  - LLIO利用時の制限事項に抵触する場合など
- アクセスパス (すべてのグループが利用可能)
  - /2ndfs/<groupname>/
- 割り当て容量（申請により拡大は可能ですが，大きな領域を割り当てることはできません）
  - Disk Quota: 5TiB
  - i-node : 1.5M
- 注意事項
  - 2ndfsは，第2階層ストレージのキャッシュ領域の対象ではありません
  - 以下の操作は，2ndfsに高負荷を与える場合があるためお控えください
    1. 同一ディレクトリに対する1000プロセス以上からの一斉ファイル作成
    2. 2ndfs上の同一ディレクトリに，十万以上のファイルを作成
    3.以下を同時に行う処理
       a.同一ディレクトリに複数プロセスからファイル作成
       b.上記ディレクトリ上のファイルを複数プロセスから参照

2ndfsについて (2/3)



2ndfsについて (3/3)

- 利用方法
  - ファイルを /2ndfs/<groupname>/下 に置き、ジョブからアクセスしてください。
    例）入力ファイルを2ndfs に置き、LLIO上でジョブを実行する場合
    (2ndfs上で実行する場合は、llio_transferは利用できません)

```
#!/bin/sh
#PJM -L  "node=384"
#PJM -L  "rscgrp=small"
#PJM --mpi "max-proc-per-node=4"

INPUT=/2ndfs/<groupname>/<username>/input

llio_transfer ./a.out
mpiexec -stdout-proc ./output.%j/%/1000r/stdout ￥
        -stderr-proc ./output.%j/%/1000r/stderr ￥
        ./a.out  ${INPUT}
llio_transfer --purge ./a.out
```

- 使用状況の確認方法
  - accountdコマンドで確認できます。

```
$ accountd -f /vol0001
```

LLIO領域の選択例

検討順	領域名	参照範囲	アクセスパス	最大容量
1	ノード内テンポラリ領域	計算ノード内	$PJM_LOCALTMP	87GiB
2	ジョブ内共有テンポラリ領域	ジョブ内	$PJM_SHAREDTMP	87GiB * #nodes
3	第2階層ストレージのキャッシュ領域	ジョブに割り当てられた全ての計算ノード	/vol0n0m/data/groupname	87GiB * #nodes
4	2ndfs	全ての計算ノード	/2ndfs/groupname	5TiB (default)

領域名	適用事例
ノード内テンポラリ領域	ランク・ノード毎に独立した中間ファイルや一時ファイルの作成・参照
ジョブ内共有テンポラリ領域	ランク・ノード間でファイルを参照，巨大なファイルの操作
第2階層ストレージのキャッシュ領域	標準出力・標準エラー出力など、保存されるファイルの出力
2ndfs	LLIOの利用制限にかかるアクセス等を実施する必要がある場合

The mount configuration of file system of the compute node was changed (1/2)

- We changed the mount configuration of the file system of the compute node in the maintenance work in September 2021.
- Due to this configuration change, the job elapse may be longer or shorter, or job may not run. Please see the details of the work and review your jobs.

- [Details of configuration change]
  - All access paths to the second-layer storage are via the cache area of second-layer storage.

	2nd layer storage (FEFS) pathname	2nd layer storage cache pathname
before	/vol000x/<groupname>	/vol000x_cache/<groupname>
after	deleted	/vol000x/<groupname>

  - The access /home/<groupname> and /data/<groupname> are also via the cache area of second-layer storage.

- [New FS : “2ndfs”]
  - You can access “2ndfs” directly from the jobs or the compute node (not via LLIO.)
  - All groups can use it

The mount configuration of file system of the compute node was changed (2/2)

- [for users who already use LLIO]
  - Access path "/vol000x_cache" is deleted.
  - The path specified to a llio_transfer command must be changed to a path without "_cache".

```
llio_transfer ./a.out
mpiexec -stdout-proc ./output.%j/%/1000r/stdout -stderr-proc ./output.%j/%/1000r/stderr ./a.out
llio_transfer --purge ./a.out
```

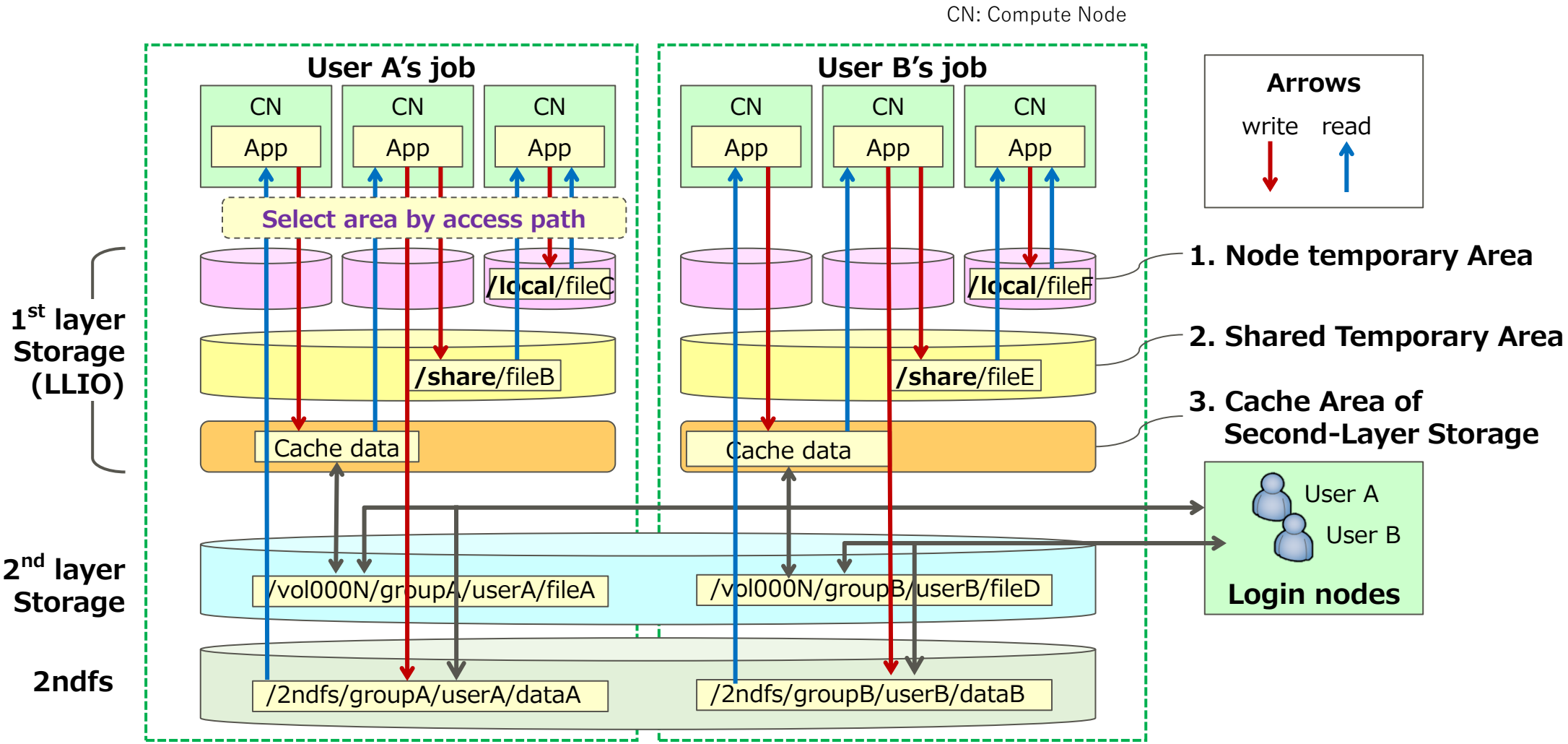
- [for users who use FEFS only]
  - All accesses to the second-layer storage is changed via the cache area of second-layer storage (LLIO.)
  - Please use LLIO after reading the user manual
  - It may take time to read common access files (like a.out) that all processes read, so please use llio_transfer command for these files.

- [Changing default settings of pjsub command]
  - llio-sio-read-cache
    - Default is ON, so cache files in first-layer storage when reading the files from the second-layer storage to the compute node.
  - llio-async-close
    - Default is OFF, so the close process is synchronous close on LLIO.
  - mpiexec-std-emptyfile
    - Default is ON, so do not create empty file.
  - llio-stripe-count
    - Default is 24, so access to Shared temporary area and cache area of second-layer storage is parallel distributed access
    - In this setting, number of files that a job can open at the same time and number of files that can be used by a job are counted by each stripe, so there may be restrictions on the use of LLIO.
    - If there are restrictions on the use of LLIO, please change the stripe count or use 2ndfs

2ndfs (1/3)

- “2ndfs” is a filesystem that jobs can access directly.
  - 2ndfs is the same of the 2nd storage before
- Please use 2ndfs when you have problems with job execution using LLIO.
  - ex) use 2ndfs when you meet the restriction of LLIO
- Access path (all groups can use)
  - /2ndfs/<groupname>/
- Assigned capacity (Expandable with application, but not large space)
  - Disk Quota: 5TiB
  - i-node : 1.5M
- Note
  - Cache area of 2nd layer storage is not available on 2ndfs
  - Please do not perform the following operations because they may cause a heavy load on 2ndfs.
    1. Simultaneous file creation in the same directory from more than 1000 processes
    2. Create 100,000 or more files in the same directory on 2ndfs
    3. Process that performs the following simultaneously
       a. Create files in the same directory from multiple processes
       b. Reference files in the above directory from multiple processes

2ndfs (2/3)


CN: Compute Node
User A's job
User B's job
CN
App
Select area by access path
/local/fileC
/local/fileF
/share/fileB
/share/fileE
Cache data
1st layer Storage (LLIO)
2nd layer Storage
2ndfs
/vol000N/groupA/userA/fileA
/vol000N/groupB/userB/fileD
/2ndfs/groupA/userA/dataA
/2ndfs/groupB/userB/dataB
Arrows
write
read
1. Node temporary Area
2. Shared Temporary Area
3. Cache Area of Second-Layer Storage
User A
User B
Login nodes

2ndfs (3/3)

- How to use
  - Store files in “/2ndfs/<groupname>/” and access it from jobs.
    Ex.) When storing input files on 2ndfs and run the job on LLIO.
    (llio_transfer is not available when you run the job on 2ndfs)

```
#!/bin/sh
#PJM -L  "node=384"
#PJM -L  "rscgrp=small"
#PJM --mpi "max-proc-per-node=4"

INPUT=/2ndfs/<groupname>/<username>/input

llio_transfer ./a.out
mpiexec -stdout-proc ./output.%j/%/1000r/stdout ¥
        -stderr-proc ./output.%j/%/1000r/stderr ¥
        ./a.out  ${INPUT}
llio_transfer --purge ./a.out
```

- How to see the usage of 2ndfs
  - Use accountd command

```
$ accountd -f /vol0001
```

ex.) LLIO area selection

Order of Consideration	Area Name	Range of References	Access Path	Maximum Size
1	Node Temporary Area	In the compute node	$PJM_LOCALTMP	87GiB
2	Shared temporary area	In the job	$PJM_SHAREDTMP	87GiB * #nodes
3	Cache Area of Second-Layer Storage	Assigned compute nodes for the job	/vol0n0m/data/groupname	87GiB * #nodes
4	2ndfs	All compute nodes	/2ndfs/groupname	5TiB (default)

Area Name	Example of Applying LLIO
Node Temporary Area	Create and reference intermediate and temporary files independently for each rank and node
Shared temporary area	Reference files between ranks and nodes, manipulate large files
Cache Area of Second-Layer Storage	Output of the saved files such as stdout or stderr
2ndfs	When you have restrictions on the use of LLIO

小規模リソースグループにおける電力設定の変更について

- システム電力の省電力化を目的とし，small/small-free/int/int-freeなどの384ノード以下のジョブを実行する小規模リソースグループにおいて，ジョブ実行時のリテンション遷移に関するデフォルト設定を変更します.

- 変更内容
  - small/small-free/int/int-free のジョブ実行時のパワーモード指定（ジョブ実行時のパワーノブ操作）のデフォルト設定を 「リテンション遷移しない」から「リテンション遷移を許可する」に変更します
  - ジョブ実行時のパワーモード指定（ジョブ実行時のパワーノブ操作）において，計算コアのリテンション状態遷移の可否（retention_state）を指定できるようになります
  - ジョブ実行中に，PowerAPIで計算コアのリテンション状態遷移の可否（retention_state）を操作できるようになります
  - IOノードについてもジョブ実行時のパワーモード指定のデフォルトを「リテンション遷移しない」から「リテンション遷移を許可する」に変更します（追加実施）

- 変更（追加実施）日時
  - 2022/05/17 14:30　※この日時より前に実行開始したジョブは本変更は適用されません

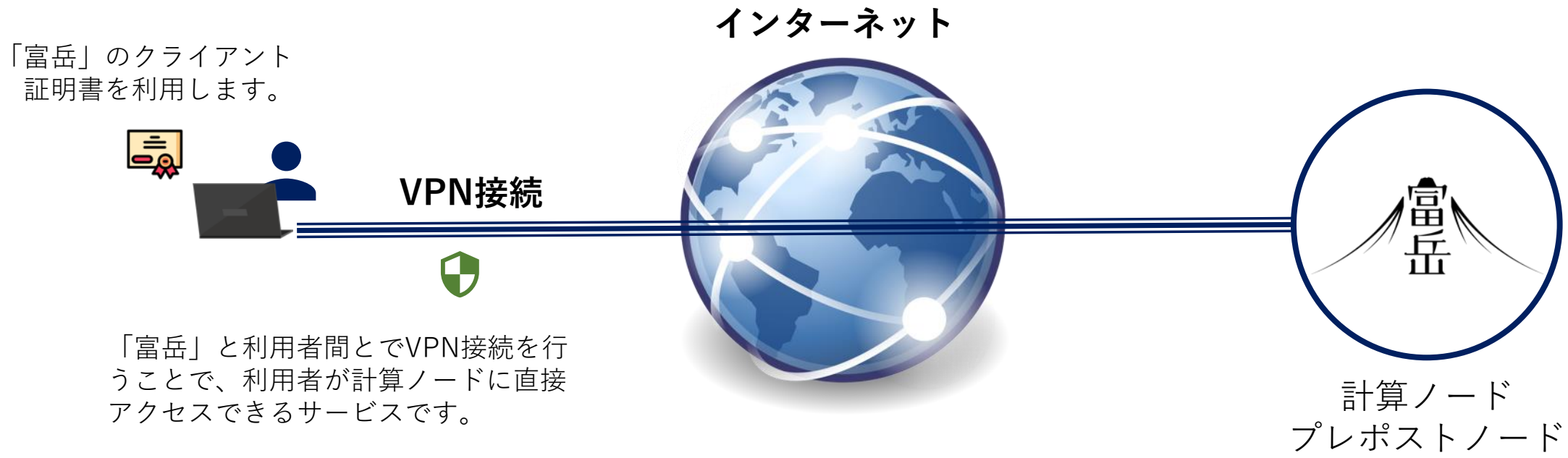
- 設定変更による影響
  - リテンション遷移を許可することで，計算コアの状態遷移に要する時間が長くなるため，ジョブ全体の実行時間が長くなる場合があります.
  - リテンション遷移の詳細な条件についてはFAQで説明します。
  - リテンション遷移を許可しない場合はジョブ投入時のパワーモード指定にて以下を指定してください.
    retention_state=0

Changing power settings in small scale resource groups

- In order to save system power, the default settings for retention transition during job execution are changed for small scale resource groups execute jobs by less or equal 384 nodes such as small/small-free/int/int-free.
- Description
  - The default setting for the power mode specification during job execution (power knob operation during job execution) of small/small-free/int/int-free changes from "No retention transition" to "Allow retention transition".
  - Users can specify whether the compute core is allowed to transition to the retention state (retention_state) in the power mode specification during job execution (power knob operation during job execution).
  - Users can control whether the compute core is allowed to transition to the retention state (retention_state) in PowerAPI during job execution.
  - Same as compute nodes, I/O node‘s default setting for the power mode specification during job execution (power knob operation during job execution) changes from “No retention transition” to “Allow retention transition“ (Additional implementation).
- Change (additional implementation) date：
  2022/05/17 14:30 (JST)　*This change does not apply to jobs started before this date.
- Effect of changing settings
  - Since the time required for the state transition of the compute core increases by allowing the retention transition, the whole execution time of the job may increase.
  - Detailed conditions for the retention transition are described in the FAQ.
  - If you do not allow the retention transition, please specify the following in the power mode specification at job submission.
    retention_state=0

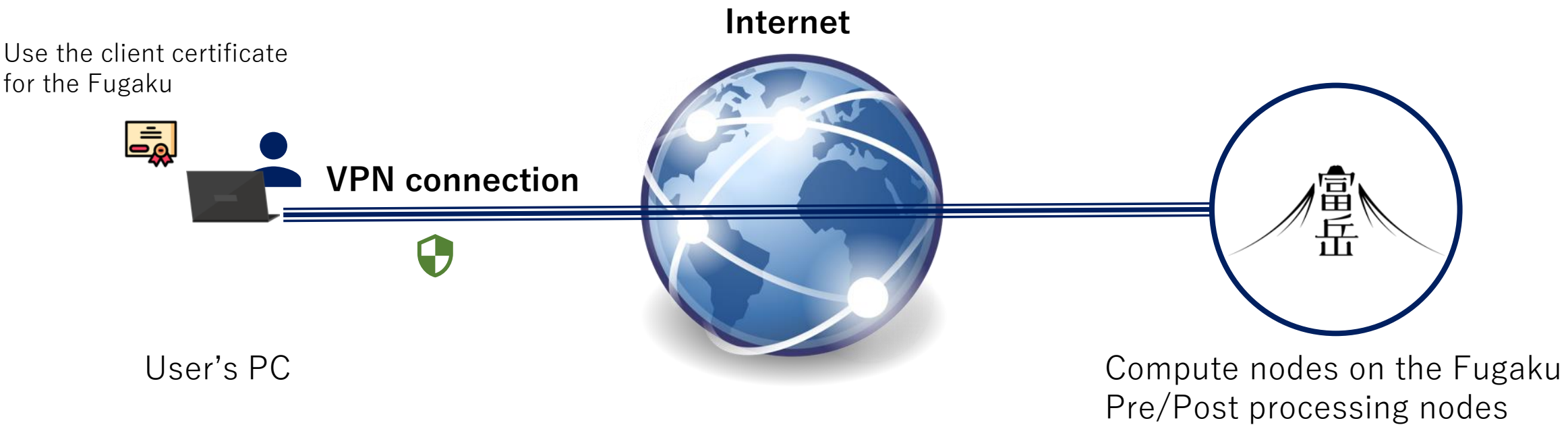
「富岳」VPN接続サービス

サービス利用者は、「富岳」VPN接続サービスを利用して、自宅や出先等から
インターネット経由で「富岳」の計算ノード/プレポストノードに直接アクセスできます。
https://www.fugaku.r-ccs.riken.jp/doc_root/ja/user_guides/vpn_guide/



Fugaku VPN Service

Users can directly access the compute nodes and the Pre/Post nodes on the supercomputer Fugaku via the VPN service.
https://www.fugaku.r-ccs.riken.jp/doc_root/en/user_guides/vpn_guide/



job_eventsコマンド

- job_events コマンドにて、以下を確認できます。
  ・LLIO（第2階層ストレージキャッシュ）とFEFS（第2階層ストレージ）にて
  evictの影響を受けたジョブ
  ・パワーキャッピングに達したジョブ
  ・OOM（Out of memory）の影響を受けたジョブ
  ・ハード故障の影響を受けたジョブ
  ・LLIO利用制限超過が発生したジョブ
- 使用方法
  - login $ job_events [-g [GROUP_NAME]]
- 実行例
  - login $ job_events

```
JOBID      MD   USER   GROUP   ST  JOB_START           JOB_END             MESSAGES
1111111    NM   user01 group01 EXT 2021/03/01 16:23:13 2021/03/01 16:23:25 Filesystem I/O error
2222222[2] BU   user01 group01 EXT 2021/03/02 14:47:49 2021/03/02 14:57:51 POWER CAPPING:2021/03/02 14:50:00
2222222[3] BU   user01 group01 EXT 2021/03/02 14:47:49 2021/03/02 14:57:51 POWER CAPPING:2021/03/02 14:50:00,Filesystem I/O error
3333333    NM   user01 group01 EXT 2021/08/17 15:55:00 2021/08/17 17:00:00 Out Of Memory
4444444    NM   user01 group01 EXT 2021/08/18 12:00:00 2021/08/17 15:00:00 POWER CAPPING:2021/08/18 14:53:00,Filesystem I/O error,Out Of Memory
5555555    NM   user01 group01 EXT 2023/02/01 15:55:00 2022/02/25 17:10:02 Hardware error
6666666    NM   user01 group01 EXT 2023/02/01 16:55:00 2022/02/25 18:10:02 LLIO Limit Over
```

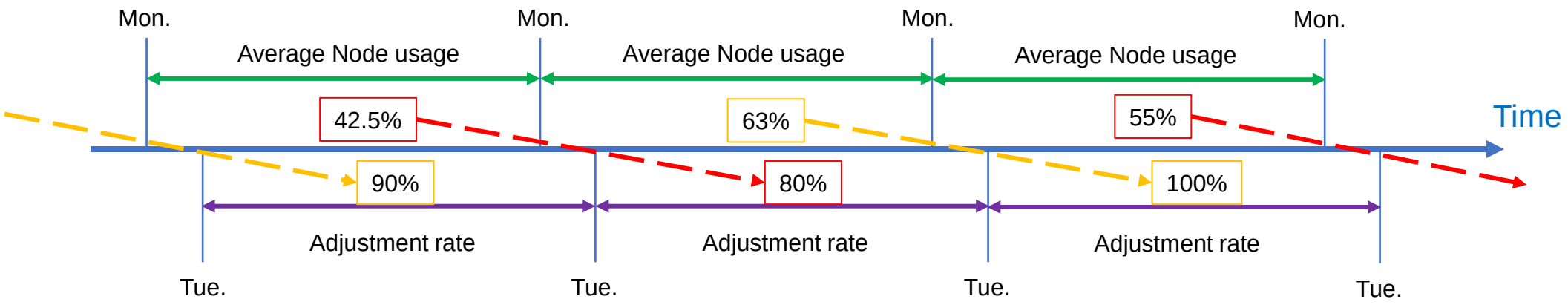
About the job_events Command

- You can use the job_events command to check the following.
  - Evict-affected jobs in LLIO (Second-layer storage cache) or FEFS (Second-layer storage).
  - Jobs that have reached power capping.
  - OOM(Out of Memory)-affected jobs.
  - Jobs affected by a hardware failure
  - LLIO Usage Limit Exceeded Jobs
- How to use
  - login $ job_events [-g [GROUP_NAME]]
- Execution example
  - login $ job_events

```
JOBID       MD  USER    GROUP    ST   JOB_START            JOB_END                MESSAGES
JOBID      MD USER   GROUP   ST  JOB_START    JOB_END MESSAGES
1111111    NM user01 group01 EXT 2021/03/01 16:23:13 2021/03/01 16:23:25 Filesystem I/O error
2222222[2] BU user01 group01 EXT 2021/03/02 14:47:49 2021/03/02 14:57:51 POWER CAPPING:2021/03/02 14:50:00
2222222[3] BU user01 group01 EXT 2021/03/02 14:47:49 2021/03/02 14:57:51 POWER CAPPING:2021/03/02 14:50:00,Filesystem I/O error
3333333    NM user01 group01 EXT 2021/08/17 15:55:00 2021/08/17 17:00:00 Out Of Memory
4444444    NM user01 group01 EXT 2021/08/18 12:00:00 2021/08/17 15:00:00 POWER CAPPING:2021/08/18 14:53:00,Filesystem I/O error,Out Of Memory
5555555    NM user01 group01 EXT 2023/02/01 15:55:00 2022/02/25 17:10:02 Hardware error
6666666    NM user01 group01 EXT 2023/02/01 16:55:00 2022/02/25 18:10:02 LLIO Limit Over
```

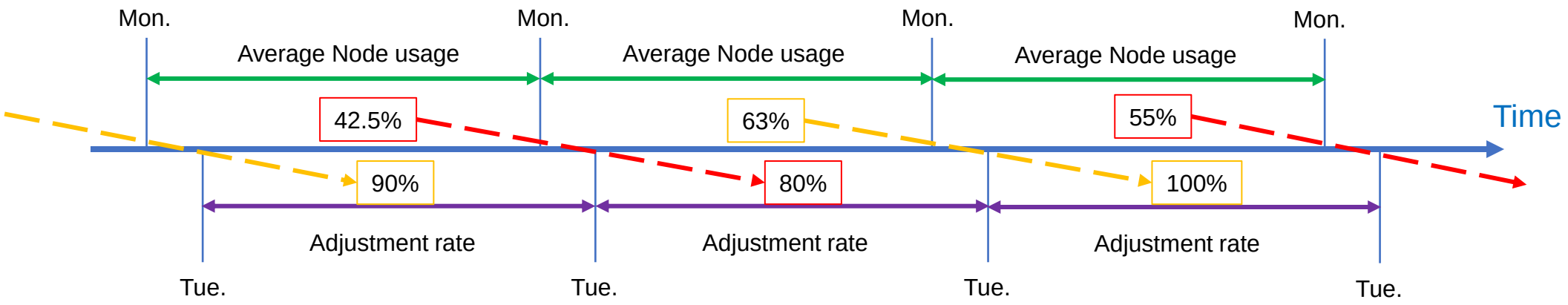
ジョブ利用実績の変動制導入について

- 計算ノードの利用促進を目的として，リソースグループ small と large に対して，計算ノードの利用状況に応じたジョブの利用実績が変動する仕組みを2021/6/22より導入しました.
- この仕組みでは，特定期間のリソースグループのノード利用率に基づいて翌期間の調整率が決まり，ジョブの利用実績が変わります （対象リソースグループのシステム全体での利用率が低い場合，ジョブの利用実績が割引になります）.
- 調整率等の各種パラメータは「ジョブ調整率表」を参照してください.
  - 各種パラメータは予告なく変更される場合があるのでご注意ください.



Variable accounting system based on node usage

- To improve the node usage, we introduced a variable accounting system based on node usage to resource group "large" and "small" at 22nd June 2021.
- This system decides an adjustment rate based on node usage during a certain period, and used resource of a job will vary depending on the rate.
  (When node usage of target resource group is low, used resource on that resource group will be discounted.)
- Please refer "Accounting adjustment rate" for parameters of this system
  - Please note that the parameters will be changed without notice.



計算資源の利用の定義

$$\sum_{n=1,\#jobs} \left(n\text{番目のジョブに割り当てられたノード数} \times n\text{番目のジョブの経過時間} \right)$$

n番目のジョブに割り当てられたノード数

＝ スケジューラがジョブに割り当てたノード数
≠ ジョブスクリプトまたはpjsub実行時に指定したノード数
≠ 実際にジョブの実行中に使用したノード数

- 384ノード以下のジョブは，指定したノード割り当て方式で，割り当てられるノード数が変わります．
  - torus : 12ノード単位(2*3*2)でノードが割り当てられます．
  - mesh : 指定されたmeshを形成するのに必要なノード数が割り当てられます．
  - noncont (デフォルト指定) : 指定されたノード数が割り当てられます．この場合，隣接したノードが割り当てられない場合があるので，ノード間通信が他のジョブの影響を受ける場合があります．
- 385ノード以上のジョブは48ノード単位(2*3*8)(Shelf単位)でノードが割り当てられます．そのため，ジョブ投入時に指定したノード数よりも多いノードがジョブに割り当てられる場合があります．48ノード単位(2*3*8)で構成される直方体でノードを指定すると，この無駄を少なくすることができます．Shelf単位でノードが割り当てられるため，ジョブは第1階層を占有することができます．
- ノードの空き状況によっては，指定した形状を回転させて割り当てることがあるため，ジョブを投入するごとに割り当てられるノード数が変わることがあります．':strict'オプションを指定すると回転を抑止できますが，ジョブの実行開始時刻が遅くなる場合があります

n番目のジョブの経過時間

＝ 実際にジョブの実行にかかった時間
＝ ステータスが「RUN」になっている時間 （再実行された場合は全ての実行時間の和）
≠ ジョブスクリプトまたはpjsub実行時に指定した経過時間

※ 原則資源返却は行いません

Definition of used computational resource

$$\sum_{n=1,\#\,jobs}\left(\begin{array}{c}\#\ of\ assigned\ nodes\\ to\ n\text{-}th\ job\end{array}\times\begin{array}{c}Elapsed\ time\ of\\ n\text{-}th\ job\end{array}\right)$$

of assigned nodes to n-th job

= # of nodes assigned by the scheduler

≠ # of requested nodes that is defined in job script or option of pjsub command

≠ # of nodes used by job

- In case of jobs with less than 385 nodes, # of assigned nodes varies by specified node allocation mode.
  - torus : node allocation unit is 12 nodes (2*3*2)
  - mesh : # of nodes needed to make specified mesh pattern Is assigned to the job
  - noncont (default) : Specified # of nodes is assigned to the job. Adjacent nodes may not always be allocated to the jobs. thus, inter-node communication may be disturbed by other jobs.
- In case of jobs with more than 384 nodes, node allocation unit is 48 nodes (2*3*8). Therefore, # of assigned nodes is sometimes larger than # of requested nodes. You can minimize # of unused nodes by requesting nodes consisting of 48 nodes (2*3*8). The job can occupy the 1st layer storage that assigned to it.
- # of assigned nodes varies when you submit job because the scheduler allocates nodes to job with rotation depending on scheduling status. You can prevent this rotation using ':strict' option, but the start time of job may be postponed.

Elapsed time of n-th job

= running time of job

= time with "RUN" state (if re-run, time is sum of all runs.)

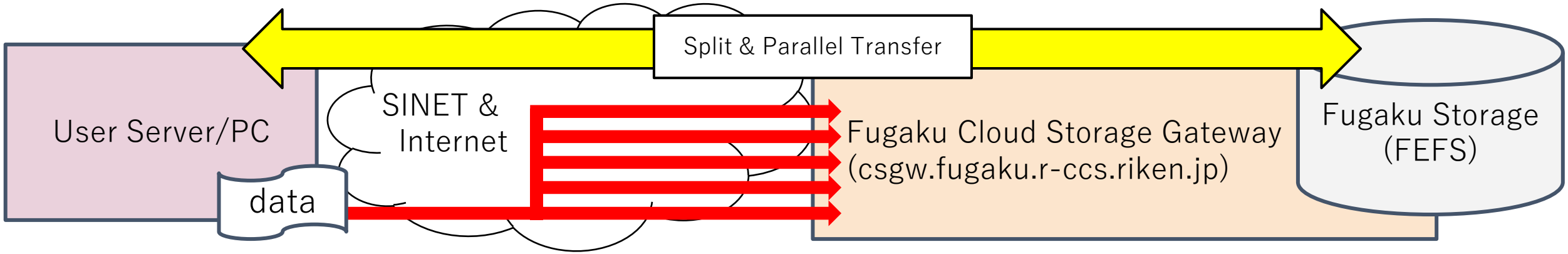
≠ requested time that is defined in job script or option of pjsub command

※ Used computational resource will NOT be refunded under any condition

Fugaku High Speed Transfer Guide now available

Users Data can be transferred between a user's server and Fugaku using GridFTP.

- Guide URL:
  - (JA) https://www.fugaku.r-ccs.riken.jp/doc_root/ja/user_guides/high_speed_io_guide_1.0/
  - (EN) https://www.fugaku.r-ccs.riken.jp/doc_root/en/user_guides/high_speed_io_guide_1.0/
- GridFTP is installed and setup on the Fugaku Cloud Storage Gateway(csgw.fugaku.r-ccs.riken.jp).
- In order to use it, the GridFTP client needs to be installed on the user's server.
- GridFTP is a protocol for parallel data transfer,
  single large file can be split and transferred in parallel.
- SSH public key authentication and GSI authentication are supported,
  with GSI authentication requiring an HPCI proxy certificate.



AIフレームワーク

AIフレームワークTensorFlow-2.2.0, PyTorch-1.7.0をアップデートしました．物体検知Mask R-CNNの高速動作をサポートします．旧版PyTorch-1.5.0/1.6.0, TensorFlow-2.1.0, Chainer-4.5.0も配置してますが，旧言語環境で構築しているため動作保証できません．富岳向け高速化ライブラリを使用します．質問・要望はサポートデスクまでご連絡下さい．

- インストール場所
  第2階層FS上：/home/apps/oss/...
- インストールパッケージバージョン
  PyTorch ver.1.7.0, ver.1.6.0
  TensorFlow-2.2.0, Horovod ver.0.20.3
  oneDNN ver.2.1.0L01_aarch64
  Python ver.3.8.2
    mpi4py ver.3.0.3，pandas ver.1.2.2,
    numpy ver.1.19.0, scipy ver.1.5.2,
    h5py ver.2.8.0，netcdf ver.1.5.6,
    OpenCV ver.4.3, fapp ver.1.0.0
- 制限事項
  - TensorFlowにてfipp/fapp/PAでHWカウンタ情報が取得できません(修正時期未定).
- LICENCE
  提供したパッチおよび手順はOSSに提供(Upstream化)予定であり、Upstream化された際はPyTorchは修正BSDライセンス，TensorFlowはApatch2.0に準拠します。
- 参考資料
  https://github.com/fujitsu/tensorflow/wiki/TensorFlow-oneDNN-build-manual-for-FUJITSU-Software-Compiler-Package-(TensorFlow-v2.2.0)
  https://github.com/fujitsu/pytorch/wiki/PyTorch-DNNL_aarch64-build-manual-for-FUJITSU-Software-Compiler-Package-(PyTorch-v1.7.0)
- 実行方法
  上記URLに利用手順の記載があります．example配下にイメージ読み込み無しのResNet-50ベンチマークの実行例と して，ノード内MPI並列有/無，train/inference，tracer，fipp/fapp/PAの取得例がございます．また自然言語処理のOpenNMT，Bert, 物体検知のMask R-CNNなどのサンプルも入っており，高速化ライブラリを読み込めるようになっています．パスなど調整の上ご利用可能です．ご参照下さいませ．フレームワークごとに利用できる機能や性能は異なります．
  パスを設定することで，好きな場所に配置しても実行できます．Python moduleはfile数が多いため，多並列ではMDSアクセス高負荷になりimportlibの時間がかかります．tarなどで固めてLLIO_transferや/tmp などへ展開することで，I/O処理の高速化が可能です．
- 今後の対応予定
  制限事項の解除，オプショナル機能・モジュールの追加，Spackによるユーザ環境でのbuild，Singularityコンテナ提供，富士通言語環境への追従．PyTorch, oneDNNバージョンアップへの追従．旧バージョンへの対応，FX700での提供.

AI framework

We updated the AI frameworks, TensorFlow-2.2.0, PyTorch-1.7.0 for Fugaku. These versions support the high performance training of Mask R-CNN for object detection. The old version PyTorch-1.5.0/1.6.0, TensorFlow-2.1.0, Chainer-4.5.0 are also installed, but these may not work, because it is built in the old language environment. These AI frameworks use the customized library for Fugaku. For support, please contact the support desk.

- Installed location:
  on the second level FS
  /home/apps/oss/...
- Installed package version
  PyTorch ver.1.7.0, ver.1.6.0
  TensorFlow-2.2.0, Horovod ver.0.20.3
  oneDNN ver.2.1.0L01_aarch64
  Python ver.3.8.2
    mpi4py ver.3.0.3, pandas ver.1.2.2,
    numpy ver.1.19.0, scipy ver.1.5.2,
    h5py ver.2.8.0, netcdf ver.1.5.6,
    OpenCV ver.4.3, fapp ver.1.0.0
- Limitation
  - TensorFlow: HW counter information cannot be acquired with fipp / fapp / PA (correction schedule undecided).
- License
  These patches and procedures will be provided to OSS (upstreaming), and PyTorch will comply with the modified BSD license and TensorFlow will comply with the Apatch 2.0 license when being upstreamed.
- References
  https://github.com/fujitsu/tensorflow/wiki/TensorFlow-oneDNN-build-manual-for-FUJITSU-Software-Compiler-Package-(TensorFlow-v2.2.0)
  https://github.com/fujitsu/pytorch/wiki/PyTorch-DNNL_aarch64-build-manual-for-FUJITSU-Software-Compiler-Package-(PyTorch-v1.7.0)
- Execution
  The usage is written on the above URL. Under the example directory, there are some samples of ResNet-50 benchmark without image reading, which are MPI parallel / serial in node, train / inference , tracer and fipp / fapp / PA. It also contains samples such as OpenNMT and Bert for natural language processing, Mask R-CNN for Object detection. These examples can load high-speed libraries. You can be used these after adjusting the path etc. Please refer. The functions and performance that can be used differ depending on the framework.
  By setting the path, you can execute it even if you place it anywhere you want use. Since the Python module has a large number of files, MDS access becomes heavy and importlib takes time under the massive parallels. I/O time can be reduced by putting it by LLIO_transfer with tar compress.
- Future plan
  Release of restrictions. Addition of optional functions / modules. Build in user environment by Spack. Provision of Singularity container. Follow-up to the Fujitsu language environment. Follow-up PyTorch, oneDNN version. Correspondence to the old version. Provision AI framework on the FX700.

アプリケーション情報の収集について

5/17より，運用改善を目的に実行するアプリケーションの以下の情報を収集します.

- コンパイラ情報
- シンボルテーブル
- 共有ライブラリ情報
- MD5 ハッシュチェックサム
- 実行ファイルの種類や文字列

これらアプリケーション情報の収集を回避するには，ジョブ実行時に以下の環境変数を定義してください.

#PJM –x “APPLOG=NO”

情報収集はmpiexec の開始時のみに行うため，実行中のジョブの性能に影響は出ません.
また，収集によるジョブの実行時間への影響はほとんどありません.

※利用手引書　記載予定

Collection of Application Information

The following information of applications will be collected for the sake of improvement of the system operation since May 17th.

- Compiler information
- Symbol table
- Shared library information
- MD5 hash checksum
- Type and strings of executable file

You can inhibit collecting of those information by setting the following environment variable:

#PJM –x “APPLOG=NO”

Information collection is performed only at the beginning of an execution of mpiexec, so it does not affect the performance of running jobs.
Also, it does not so much affect execution time of jobs

Note: This will be written on Users Guide.

LLIOの提供について

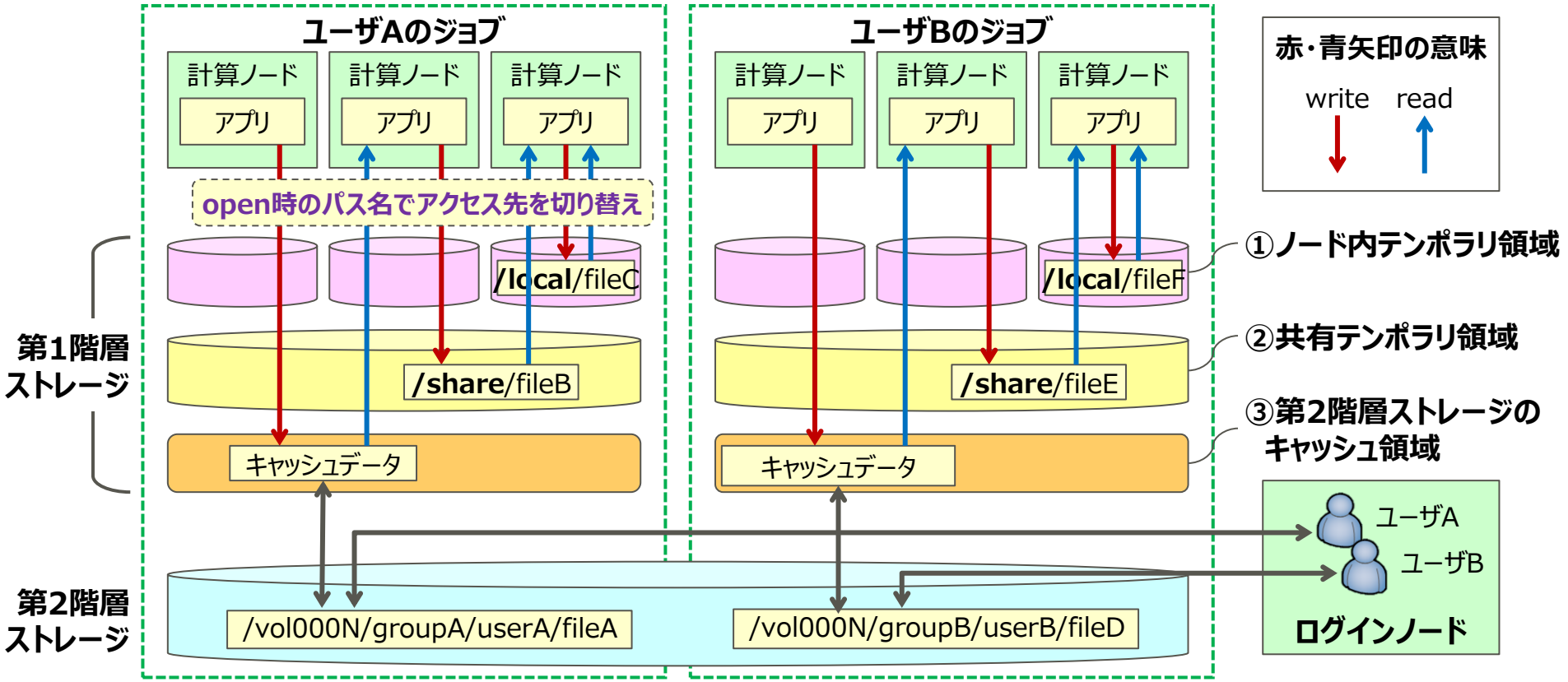
全リソースグループでLLIO機能(3機能)が利用できます．ジョブのI/O時間を短縮できる可能性などがあります．
富岳ポータルの情報も参考にしてください（https://www.fugaku.r-ccs.riken.jp/operation/20210510_01）
制限事項は，後述の留意事項を参照してください

- ノード内テンポラリ領域
  - 同一計算ノード内での共有のみ可能（第2階層とは独立した領域）
    - 「京」のランク番号ディレクトリのように，ノード内で多数の一時ファイルを扱う場合に効果があります
  - ジョブ開始時に初期化され，終了時に削除されます
  - マウントポイント：/local

- 共有テンポラリ領域
  - ジョブに割り当てられた計算ノード間で参照できます（第2階層とは独立した領域）
  - ジョブ開始時に初期化され，終了時に削除されます
  - マウントポイント：/share

- 第2階層ストレージのキャッシュ領域
  - ジョブ内のすべての計算ノードから参照できます
  - 計算ノードからの書出し要求とは非同期に第2階層ストレージに書き出します
  - ジョブ開始時に初期化され，終了時に削除されます
  - 書出しの途中でジョブ実行可能時間制限を超えるなどしてジョブが中断された場合は，第2階層ストレージのキャッシュ領域は削除されます
  - 書き出されなかったファイルは，未書出しファイルの一覧としてファイルに出力します
  - マウントポイント：/vol0004， /vol0006，etc…

LLIO：ファイルシステム構成



参考：LLIO ユーザーズガイド，ジョブ運用ソフトウェア エンドユーザ向けガイド

LLIO：留意事項

※（制限事項）は今後の運用状況に応じて変更される場合があります

- 第2階層ストレージのキャッシュ領域使用時のファイル指定方法がマニュアル記載と異なります
- 以下の制限で利用してください。制限を超えるとENOMEMでエラーになります。（制限事項）
  - ジョブが同時にopenする最大ファイル数は、LLIOが提供する3つの領域の合計で，1,024 * (ジョブが使用する計算ノード数) 以下にする
    - 同一のファイルを複数の計算ノードからopenする場合は，計算ノード数によらず1ファイルとしてカウント
    - llio_transferで転送し共有ファイルとして利用する場合は，異なるファイルとしてカウントするため、同時にopenする計算ノード数でカウント
    - LLIOのストライプカウントが2以上の場合，ストライプ毎に1ファイルとしてカウント
      - 1ファイルでもストライプカウントが4ならば，ファイル数は4とカウント
- 1つのファイルを複数のプロセスから利用する場合，以下の2つの条件を両方満たすようにしなければ，I/Oがスローダウンします
  1. 同一ファイルを利用するプロセスが存在するノード数が7,000以下
  2. 同一ファイルを利用するプロセスの総数が28,000以下 (計算ノードあたりのプロセス数の上限なし)
- ストライプ数の指定は，最大24まで (制限事項)

LLIO：留意事項

※（制限事項）は今後の運用状況に応じて変更される場合があります

- 共有テンポラリ領域のブロック数やinode使用数が取得できません(dfコマンドで出力されません)
- LLIOの容量: 約87GiB/ノード
  - [第2階層ストレージのキャッシュ領域] = 87GiB - [共有テンポラリ領域] - [ノード内テンポラリ領域]
    - [第2階層ストレージのキャッシュ領域]の最小容量は128MiB
- 第2階層ストレージのキャッシュ領域使用時，ファイルの拡張属性の置換を行うと拡張属性が削除される場合があります
- 第2階層ストレージのキャッシュ領域使用時に異なる計算ノードから以下を同時に実施した場合，LLIO経由でファイルサイズを正しく取得できない場合があります

計算ノードA	計算ノードB
truncate	read
truncate	write
truncate	truncate

- 第2階層ストレージのキャッシュ領域または共有テンポラリ領域使用時にwriteと削除を並行して行った場合，第1階層ストレージ上にファイルデータが残存することがあります （ジョブ終了時に削除されます）
- 第2階層ストレージのキャッシュ領域または共有テンポラリ領域使用時にこれら領域にあるファイルに対して，SUID または SGID を落とす chmod() と他の属性情報を更新するchmod() を並行に実行した場合，他の属性情報を更新する chmod()が反映されない場合があります
- 全ての計算ノードから読み込まれるファイル(共通ファイル)は，llio_transferコマンドを利用することを推奨します

LLIO：ノード内テンポラリ領域の使い方

- ジョブ投入時に，ノード内テンポラリ領域のサイズを指定
  - 最小：0MiB，最大：87GiB

```
ex.) Command line option
$ pjsub --llio localtmp-size=10Gi job.sh
```

```
ex.) specification in job script
#PJM --llio localtmp-size=10Gi
```

Example 1

```
#!/bin/bash
#PJM -L "node=1"
#PJM --llio localtmp-size=10Gi

### 1. プログラムprog1がノード内テンポラリ領域${PJM_LOCALTMP}に
###    計算結果out.dataを出力
prog1 -o ${PJM_LOCALTMP}/out.data

### 2. ファイルout.dataをプログラムprog2が読み込んで、ディレクトリ${PJM_LOCALTMP}に
###    最終的な計算結果result.dataを出力
prog2 -i ${PJM_LOCALTMP}/out.data -o ${PJM_LOCALTMP}/result.data

### 3. ジョブの終了前にファイルresult.dataをジョブ実行時の
###    ディレクトリ${PJM_JOBDIR}(第2階層ストレージ上)に、
###    ジョブID${PJM_JOBID}をつけた名前に変えて退避
cp ${PJM_LOCALTMP}/result.data ${PJM_JOBDIR}/result_${PJM_JOBID}.data
```

環境変数名	意味
PLE_RANK_ON_NODE	MPIプロセスを生成したとき、同一計算ノードで生成した各々のプロセスに対して、0～“当該計算ノード内生成プロセス数-1” の値が設定される
PJM_LOCALTMP	第1階層ストレージ上のノード内テンポラリ領域のパス

Example 2

```
#!/bin/bash -x
#PJM -L "node=48"
#PJM -L "rscgrp=small"
#PJM -L "elapse=1:00:00"
#PJM --mpi "max-proc-per-node=4"
#PJM --llio localtmp-size=30Gi # ノード内テンポラリ領域の容量を指定
#PJM --llio perf # LLIO性能情報の出力:ノード数に応じて出力量が増えるので注意
#PJM -s

### ノード内の1プロセスからa.outをコピー
mpiexec sh -c 'if [ ${PLE_RANK_ON_NODE} == 0 ]; then ¥
                   cp -f ./a.out   ${PJM_LOCALTMP} ; ¥
               fi’

# 標準出力/標準エラー出力への出力がない場合はファイルを作成しない
export PLE_MPI_STD_EMPTYFILE=off

mpiexec ${PJM_LOCALTMP}/a.out
```

LLIO：共有テンポラリ領域の使い方

- ジョブ投入時に，共有テンポラリ領域のサイズを指定
  - sharedtmp-sizeの値に，割り当てられた計算ノード数を乗じた値が共有テンポラリ領域のサイズです
  - 指定がない場合は0Bになります

```
ex.) Command line option
$ pjsub --llio sharedtmp-size=10Gi jobscript.sh
```

```
ex.) specification in job script
#PJM --llio sharedtmp-size=10Gi
```

Example

```
#!/bin/bash
#PJM -L "node=1"
#PJM -L "rscgrp=small"
#PJM --llio sharedtmp-size=10Gi # 共有テンポラリ領域の容量を指定

### 1. プログラムprog1が共有テンポラリ領域${PJM_SHAREDTMP}に
###    計算結果result.dataを出力
prog1 -o ${PJM_SHAREDTMP}/result.data

### 2. ジョブの終了前にファイルresult.dataをジョブ実行時の
###    ディレクトリ${PJM_JOBDIR}(第2階層ストレージ上)に，
###    ジョブID${PJM_JOBID}をつけた名前に変えて退避
cp ${PJM_SHAREDTMP}/result.data ${PJM_JOBDIR}/result_${PJM_JOBID}.data
```

環境変数名	意味
PJM_SHAREDTMP	第1階層ストレージ上の共有テンポラリ領域のパス

LLIO：第2階層ストレージのキャッシュ領域の使い方

Example 1

```
#!/bin/bash
#PJM -L "node=12"
#PJM -L "rscgrp=small"

export PLE_MPI_STD_EMPTYFILE=off

### 1. /vol0004/<groupname>/<username>/a.outを
###     第2階層ストレージのキャッシュ経由で利用する
mpiexec /vol0004/<groupname>/<username>/a.out
```

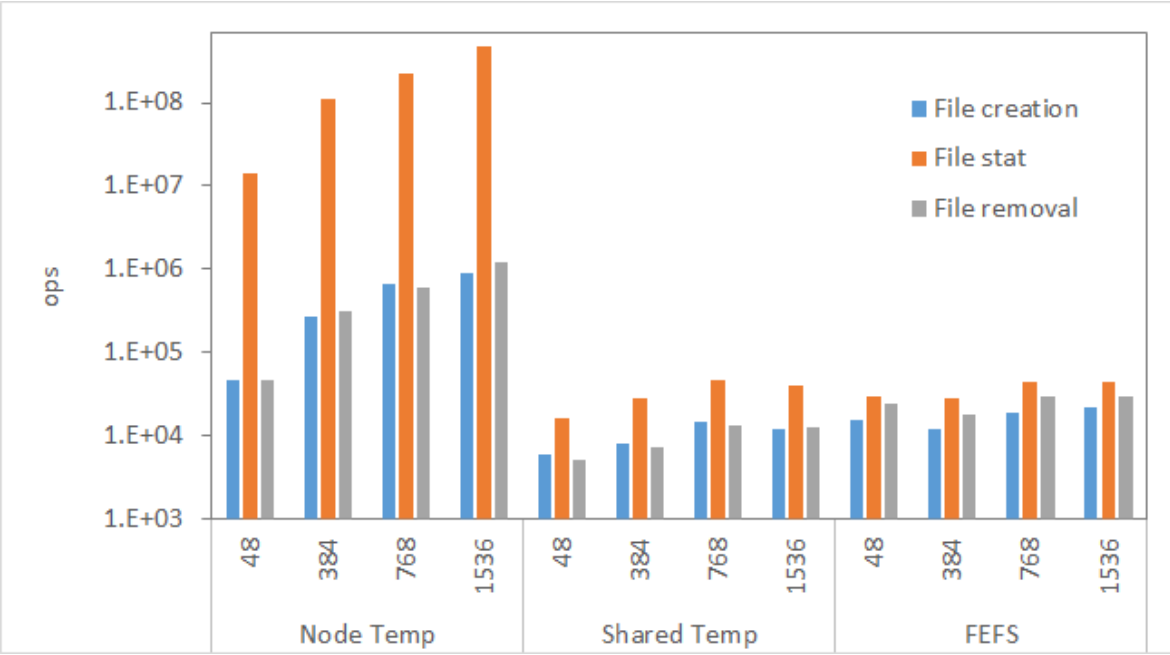
Example 2

```
#!/bin/bash
#PJM -L "node=384"
#PJM -L "rscgrp=small"
#PJM --mpi "max-proc-per-node=4"

export PLE_MPI_STD_EMPTYFILE=off

### 1. a.outを第2階層ストレージのキャッシュにコピーする
llio_transfer /vol0004/<groupname>/<username>/a.out

### 2. a.outを実行
mpiexec -stdout-proc ./output.%j/%/1000r/stdout ¥
        -stderr-proc ./output.%j/%/1000r/stderr ¥
        /vol0004/<groupname>/<username>/a.out

### 3. a.outを第2階層ストレージのキャッシュから削除
llio_transfer --purge /vol0004/<groupname>/<username>/a.out
```

llio_transferコマンドは全ての計算ノードから読み込まれるファイル(共通ファイル)を，ジョブが使用する全てのSIOにコピーします

共有テンポラリ領域の利用における注意

- 共有テンポラリ領域を使用する際に，ファイル数が多いと，ファイルのopenなどのメタデータ処理に関わる操作に時間を要する可能性があります
- ファイル数をできるだけ少なくすることにより性能改善が見込めますので，利用の際は上記の点に留意してください
- ファイル容量などで問題が無ければ，ノード内テンポラリ領域を用いることを推奨します
- 共有テンポラリ領域および第2階層ストレージのメタデータ処理は，ユーザ間で共通のサーバの資源を使用しています．一方でノード内テンポラリ領域およびllio_transferを併用した第2階層ストレージキャッシュのメタデータ処理はSIO単位で行われるため，他ジョブの影響による性能ブレが発生しにくく，また，使用するノード規模（SIO数）によって処理能力が向上することが期待できます



参考：MDTEST（4プロセス/ノード：file per rank）による性能（横軸の数字はノード数）

- Node Temp: ノード内テンポラリ領域
- Shared Temp: 共有テンポラリ領域
- FEFS: 第2階層ストレージ（FEFS）

Use of LLIO

LLIO is available on all resource groups. There are possibilities such as minimizing I/O time.

Please refer to the following information on the Fugaku portal. (https://www.fugaku.r-ccs.riken.jp/en/operation/20210510_01)

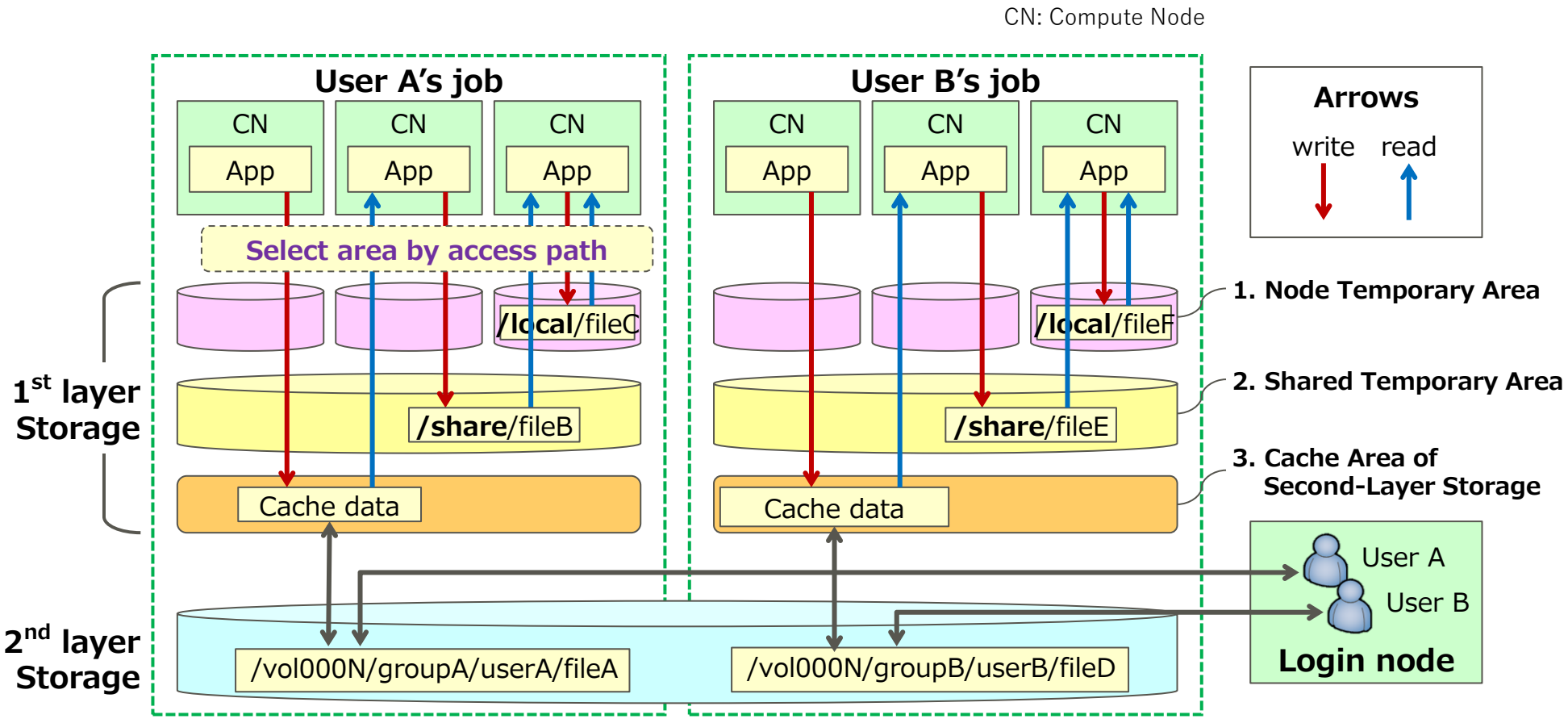
See “attention regarding using LLIO”

- Node Temporary Area
  - Only intra-node sharing is available (independent from 2nd layer storage)
    - Effective when many files are used in one node like K computer’s rank number directory
  - Initialized at job start, and deleted at job end
  - Mount point：/local

- Shared Temporary Area
  - Inter-node sharing within a job is available (independent from 2nd layer storage)
  - Initialized at job start, and deleted at job end
  - Mount point：/share

- Cache Area of 2nd Layer Storage
  - All compute nodes within a job can access
  - Data is written asynchronously for a write request from a compute node
  - Initialized at job start, and deleted at job end
  - A job may be interrupted partway through writing to second-layer storage because it exceeds the job execution time limit or for other reasons. In this event, the cache area of second-layer storage is deleted.
  - The unwritten files are output as an unwritten file list, which is a listing of incompletely written files
  - Mount point : /vol0004, /vol0006, etc…

LLIO：Filesystem structure



ref: LLIO User's Guide, Job Operation Software End-user's Guide

LLIO : attention regarding using LLIO

※（restriction）may be changed in the future

- Access path for cache area of 2nd storage is different from on the way manual.
- Please use with the following restrictions. ENOMEM raises an error if the limit is exceeded. (Restriction)
  - The maximum number of files that a job can open at the same time is 1,024 * (the number of compute nodes used by the job) or less, which is the total of the three areas provided by LLIO.
    - When opening the same file from multiple compute nodes, it is counted as one file regardless of the number of compute nodes.
    - When transferring with llio_transfer and using it as a shared file, it is counted as a different file, so it is counted by the number of compute nodes opened at the same time.
    - If a stripe count of LLIO is 2 or more, each stripe is counted as one file
      - If even one file has a stripe count of 4, the number of files is counted as 4
- When using one file from multiple processes, I / O will slow down unless both of the following two conditions are fulfilled.
  1. The number of nodes where processes that use the same file exist is 7,000 or less.
  2. The total number of processes that use the same file is 28,000 or less (No upper limit on the number of processes per compute node)
- Maximum stripe count is 24 (restriction)

LLIO : attention regarding using LLIO

※（restriction）may be changed in the future

- Access path for cache area of 2nd storage is different from on the way manual.
- Please use LLIO under the following restrictions as compute nodes may be stopped. (restriction)
  - Don’t access nor do I/O one file by more than 1,151 nodes at the same time in 2nd storage cache or Shared Temporary Area.
  - Four or fewer processes per node
- Maximum stripe count is 24 (restriction)
- Maximum number of files in Node Temporary Area is 10M files / node (restriction)
- You can’t get number of blocks and i-node in shared temporary area (df command shows no data)
- Capacity of LLIO : about 87GiB / node
  - [Capacity of cache area of 2nd storage] = 87GiB - [Capacity of shared temporary area] - [Capacity of node temporary area]
    - Minimum capacity of cache are of 2nd storage is 128MiB
- There is a case that changing the file attribute may delete the file attribute on cache are of 2nd storage.
- There is a case that you can’t get correct file size via LLIO when some compute nodes execute the following instructions at the same time on cache area of 2nd storage.

Node A	Node B
truncate	read
truncate	write
truncate	truncate

- There is a case that a file may exist on 1st layer storage when you write and delete the file at the same time on cache area of 2nd storage or shared temporary area. (the file will be deleted at job end)
- There is a case that chmod() may not work when you execute chmod() that remove SUID or SGID and chmod() that change other attributes to files on cache are of 2nd layer storage of shared temporary are at the same time.
- Use “llio_transfer” command to make file cache in 2nd storage cache used by all compute nodes.

LLIO : How to use LLIO Node Temporary Area

- Specify the capacity of Node Temporary Area when you submit job
  - Minimum: 0MiB, Maximum: 87GiB

```
ex.) Command line option
$ pjsub --llio localtmp-size=10Gi job.sh
```

```
ex.) specification in job script
#PJM --llio localtmp-size=10Gi
```

Example 1

```
#!/bin/bash
#PJM -L "node=1"
#PJM --llio localtmp-size=10Gi

### 1. prog1 writes results to out.data on Node Temporary Area ${PJM_LOCALTMP}
prog1 -o ${PJM_LOCALTMP}/out.data

### 2. prog2 reads out.data and writes results to result.data
###     in dir ${PJM_LOCALTMP}
prog2 -i ${PJM_LOCALTMP}/out.data -o ${PJM_LOCALTMP}/result.data

### 3. Copy result.data in ${PJM_LOCALTMP} to dir (${PJM_JOBDIR})
###     on 2nd layer storage as result_${PJM_JOBID}.data before job termination
cp ${PJM_LOCALTMP}/result.data ${PJM_JOBDIR}/result_${PJM_JOBID}.data
```

Environment variable name	description
PLE_RANK_ON_NODE	serial number of MPI processes in the same node. range; 0 - (number of MPI processes in the same node-1)
PJM_LOCALTMP	Path of Node Temporary Area

Example 2

```
#!/bin/bash -x
#PJM -L "node=48"
#PJM -L "rscgrp=small"
#PJM -L "elapse=1:00:00"
#PJM --mpi "max-proc-per-node=4"
#PJM --llio localtmp-size=30Gi # Capacity of Node Temopraru Area
#PJM --llio perf # Output of LLIO performance information
                 # Notice: output increases depending on number of CN
#PJM -s

### Copy a.out by 1 process in a node
mpiexec sh -c 'if [ ${PLE_RANK_ON_NODE} == 0 ]; then ¥
                   cp -f ./a.out   ${PJM_LOCALTMP} ; ¥
               fi'

# suppress the creation of zero-size STDOUT/STDERR files
export PLE_MPI_STD_EMPTYFILE=off

mpiexec ${PJM_LOCALTMP}/a.out
```

LLIO : How to use Shared Temporary Area

- Specify the capacity of Shared Temporary Area when you submit job
  - Capacity of Shared temporary Area is sharedtmp-size times number of allocated compute nodes
  - No specification means 0B

```
ex.) Command line option
$ pjsub --llio sharedtmp-size=10Gi jobscript.sh
```

```
ex.) specification in job script
#PJM --llio sharedtmp-size=10Gi
```

Example

```
#!/bin/bash
#PJM -L "node=1"
#PJM -L "rscgrp=small"
# Specify the capacity of Shared Temporary Area
#PJM --llio sharedtmp-size=10Gi
### 1. prog1 writes results to out.data on Shared Temporary Area ${PJM_SHAREDTMP}
prog1 -o ${PJM_SHAREDTMP}/result.data

### 2. Copy result.data in ${PJM_SHAREDTMP} to dir (${PJM_JOBDIR})
###     on 2nd layer storage as result_${PJM_JOBID}.data before job termination
cp ${PJM_SHAREDTMP}/result.data ${PJM_JOBDIR}/result_${PJM_JOBID}.data
```

Environment variable name	description
PJM_SHAREDTMP	Path of Shared Temporary Area

LLIO : How to use Cache Area of 2nd layer storage

Example 1

```
#!/bin/bash
#PJM -L "node=12"
#PJM -L "rscgrp=small"

export PLE_MPI_STD_EMPTYFILE=off

### 1. Use /vol0004/<groupname>/<username>/a.out via
###    cache area of 2nd later storage
mpiexec /vol0004/<groupname>/<username>/a.out
```

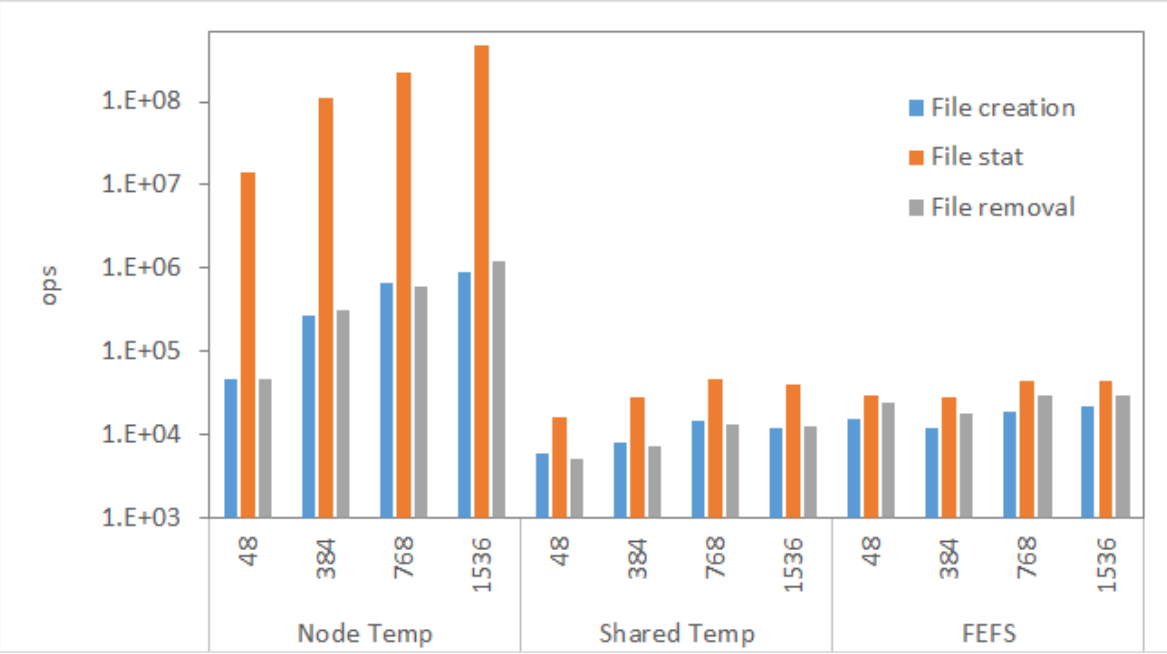
Example 2

```
#!/bin/bash
#PJM -L "node=384"
#PJM -L "rscgrp=small"
#PJM --mpi "max-proc-per-node=4"

export PLE_MPI_STD_EMPTYFILE=off

### 1. Copy a.out to Cache area of 2nd layer storage
llio_transfer /vol0004/<groupname>/<username>/a.out

### 2. Execute a.out
mpiexec -stdout-proc ./output.%j/%/1000r/stdout ¥
        -stderr-proc ./output.%j/%/1000r/stderr ¥
        /vol0004/<groupname>/<username>/a.out

### 3. Delete a.out in Cache Area of 2nd layer storage.
llio_transfer --purge /vol0004/<groupname>/<username>/a.out
```

“llio_transfer” command copies common files that all compute nodes read to all SIOs that job uses.

Notice for using the shared temporary area

- There are possibilities of performance degradation in meta-data operations such as opening files in accessing a large number of files at the shared temporary area.
- There would be performance improvements by reducing the number of files. Please also understand the above situation.
- The node temporary area is recommended if your data can be stored in its disk space.
- The shared temporary area and the second layer storage use a common server in meta-data operations. Therefore your jobs also face interference by other jobs. On the other hand, the node temporary area and the second-layer storage cache with llio_transfer are free from such interference because meta-data operations are done inside SIO. Besides, performance improvement is expected by increasing the number of compute nodes (SIOs).



MDTEST results (4 processes/node：file per rank) where numbers in the horizontal axis represent the number of compute nodes

- Node Temp: The node temporary area
- Shared Temp: The shared temporary area
- FEFS: The second layer storage (FEFS)

Arm ログインノードで作成したプログラムを計算ノードで実行する方法

1. コンパイル方法
   - Arm ログインノードで以下のmoduleでコンパイル
     - module load arm21/21.0

2. 計算ノードでの実行方法
   - 環境変数 LD_LIBRARY_PATH に以下を追加
     - /vol0004/apps/arm-lib/arm-linux-compiler-21.0_Generic-AArch64_RHEL-7_aarch64-linux/lib
     - /vol0004/apps/arm-lib/arm-linux-compiler-21.0_Generic-AArch64_RHEL-7_aarch64-linux/lib/clang/11.0.0/armpl_links/lib
     - /vol0004/apps/arm-lib/armpl-21.0.0_A64FX_RHEL-7_arm-linux-compiler_21.0_aarch64-linux/lib
     - /vol0004/apps/arm-lib/gcc-10.2.0_Generic-AArch64_RHEL-7_aarch64-linux/lib64
     - /vol0004/apps/arm-lib/gcc-10.2.0_Generic-AArch64_RHEL-7_aarch64-linux/lib

注意：実行できることを保証するものではありません.

How to execute Programs compiled by Arm compiler on Compute nodes

1. How to compile
   - Load following module and compile programs on Arm login nodes
     - module load arm21/21.0

2. How to execute programs on Compute nodes
   - Add following settings to LD_LIBRARY_PATH
     - /vol0004/apps/arm-lib/arm-linux-compiler-21.0_Generic-AArch64_RHEL-7_aarch64-linux/lib
     - /vol0004/apps/arm-lib/arm-linux-compiler-21.0_Generic-AArch64_RHEL-7_aarch64-linux/lib/clang/11.0.0/armpl_links/lib
     - /vol0004/apps/arm-lib/armpl-21.0.0_A64FX_RHEL-7_arm-linux-compiler_21.0_aarch64-linux/lib
     - /vol0004/apps/arm-lib/gcc-10.2.0_Generic-AArch64_RHEL-7_aarch64-linux/lib64
     - /vol0004/apps/arm-lib/gcc-10.2.0_Generic-AArch64_RHEL-7_aarch64-linux/lib

Note: This is provided "AS IS" without any warranty of any kind.

RAM disk（tmpfs）の提供について

- ユーザが自由に利用可能な RAM disk を作成しました．設定不要で利用できます．
  - マウントポイント：/worktmp
- ノード内でのみ共有可能
  - ノード間での共有は不可
- 利用可能期間
  - ジョブ実行開始 - ジョブ実行終了
  - ジョブ実行開始時に毎回初期化されます
- 通常のファイルシステムと同様に利用可能
  - ファイルの作成，読み書き，実行バイナリの起動
- ノードあたりの利用可能容量上限
  - ノードに搭載するメモリを使用します．以下の容量が利用可能な目安になります．
    - 32GiB - アプリケーションが利用する1ノードあたりのメモリ量 (GiB) - MPIの使用するメモリ量 (GiB) - システムソフト使用メモリ量 (GiB)
      ※上限以上の書き込みはOOM killerによりプロセスが削除されるため，一度ジョブを実行し，ノード単位の最大メモリ使用量を把握した上での利用を推奨します
- 留意点
  - メモリ枯渇によるジョブ実行の失敗が頻発する可能性があります．使用に際しては十分にご注意ください
  - /dev/shmは使用せず， /worktmp を使用してください

Providing RAM disk (tmpfs)

- You can use RAM disk without any settings.
  - Mount point : /worktmp
- Intra-node sharing is available.
  - Sharing among nodes is NOT available
- Term of validity
  - During job execution (from job start to job end)
  - RAM disk is initialized when the job is started.
- You can use RAM disk as normal file system
  - create, read, write and execute files on RAM disk
- Available upper limit per node
  - RAM disk uses memory of a node. Available capacity is following;
    - 32GiB - used by application / node (GiB) - used by MPI (GiB) - used by system software (GiB)
      * OOM killer will kill processes when memory is over used. You should run a job and check your job's memory utilization before using RAM disk.
- Notice
  - Job fault can be frequent due to memory shortage. Please be careful when using RAM disk.
  - Never use /dev/shm, please use /worktmp

ジョブ実行結果（標準出力・標準エラー出力）の出力方法の変更

7/11以降，ジョブ実行結果（標準出力・標準エラー出力）のデフォルトの出力方法が変更となりました.

【変更前】 ジョブ単位の出力

```
${jobname}.${jobid}.[out|err]
```

【変更後】ジョブ単位の出力とmpiexecのランク単位の出力

```
${jobname}.${jobid}.[out|err]                          # mpiexec実行以外の出力結果
${jobname}.${jobid}.[out|err].${mpiexec}.${rank}       # mpiexec実行時の出力結果
```

* mpiexec：ジョブスクリプト内で何回目に実行されたmpiexecかを示す番号．1から始まる数字
* rank：ランク番号

【留意事項】

高並列のジョブの場合，従来と同じくファイルシステム負荷を軽減するためにディレクトリを分けて出力してください.

具体的な指定方法については，利用手引書の「6.4.1. 標準出力／標準エラー出力／標準入力の指定方法」を参考にしてください.

Change the output method of job execution results (standard output/standard error output).

Default output method of the job execution results (standard output/standard error output) was changed at 2020/7/11

[Before]
files were output per each job
${jobname}.${jobid}.[out|err]

[After]
files are output per a job and files are output per each rank of mpiexec
${jobname}.${jobid}.[out|err] # output except from mpiexecs
${jobname}.${jobid}.[out|err].${mpiexec}.${rank} # output from mpiexec
* mpiexec : a number indicating the number of times mpiexec was executed in the job script.
Number begins from 1.
* rank : Rank number

[Notes]
Please output files in separate directories to reduce the file system load when you execute highly parallel jobs.
For more information, please refer to “6.4.1 How to specify standard output / standard error output / standard input” in the Supercomputer Fugaku Users Guide - Use and job execution -

グループ間のファイル共有について

- グループ間のファイル共有は，ディレクトリのアクセス権（FEFSのACL機能）を利用して行います
- グループ毎に共有領域(/share/hp200xxx)を作成しています
  - 初期設定では所有グループのメンバーのみがアクセス可
- このディレクトリ配下のアクセス権を適宜変更して，特定のグループ・ユーザがアクセスできるように設定します
- この領域のディスク容量は，ディレクトリの所有グループが受け持ちます
  - グループのディスク容量が足りない場合は，適宜データ領域の拡大申請をしてください
  - グループの使用ディスク容量 = ホーム/データ領域(/home/hp200xxx) + 共有領域(/share/hp200xxx)
- 作成例
  - > mkdir /share/hp200xxx/toGroupA　#GroupAに公開するdir:toGroupAを作成
  - > setfacl -m g:GroupA:rx toGroupA　# toGroupAにGroupAのアクセス権限:rxを付与
  - > setfacl -m d:g:GroupA:rx toGroupA　# toGroupA配下に作成されるファイルにdefaultでGroupAのアクセス権限:rxを付与
  - dir:toGroupAに共有したいファイルを作成（toGroupA配下の権限はtoGroupAと同等）

- アクセス権の設定方法の詳細は，利用者ポータルの利用手引書を参照してください.
  - 利用者ポータル＞ドキュメント＞利用手引書 利用およびジョブ実行編＞3.4. リソース＞3.4.6.2. ACLを利用したファイルの共有例

File sharing among groups

- You can share files among groups using ACL function of FEFS
- We made directories (/share/hp200xxx) for file sharing
  - Only owner group member can access it at initial setting.
- If you want to share files among groups or users, you need to change the setting of this directory
- Disk quota of this directory is charged for owner group of this directory.
  - If disk quota is shortage, you need to apply to expand disk quota
  - group's disk quota = home/data area (/home/hp200xxx) + shared area (/share/hp200xxx)
- example:
  - > mkdir /share/hp200xxx/toGroupA # make directory for sharing files with GroupA
  - > setfacl -m g:GroupA:rx toGroupA # add ACL (rx) toGroupA for GroupA's access
  - > setfacl -m d:g:GroupA:rx toGroupA # add ACL (rx) to files in toGroupA for GroupA's access
  - make files that you want to share with GroupA in dir : toGroupA

- For more information. please refer to "3.4.6.2. File sharing examples of using ACL" in the supercomputer Fugaku Users Guide - Use and job execution -

第2階層ストレージにおけるI/O障害の回避策

第2階層ストレージにおけるI/O障害に関して，ファイル構成やファイルI/Oパターンを以下のようにすることで，この障害を回避できる可能性があります

1. ファイル構成
    1. 問題が発生しやすいファイル構成
       実行モジュールと同じディレクトリにI/O対象のファイルを配置
       例： 実行モジュール：/path/to/dir/jobdir/a.out
            I/O対象ファイル：/path/to/dir/jobdir/stdout.001 stdout.002 ‥‥stdout.999
    2. 問題を回避できる可能性のあるファイル構成
       実行モジュールとは異なるディレクトリにI/O対象のファイルを配置
       例： 実行モジュール：/path/to/dir/jobdir/a.out
            I/O対象ファイル：/path/to/dir/jobdir/log/stdout.001 stdout.002 ‥‥stdout.999
2. ファイルI/Oパターン
    1. 問題が発生しやすいI/Oパターン
       1つのファイルに対し，複数ノードからwriteするケース
       - ディスク装置のwrite性能を超えた場合，計算ノードからのwrite要求が完了しない
       - O_TRUNC指定でopenしたファイルへのwriteではファイル単位に処理がシリアライズする
       例： mpiexec -of FILE ./a.out
    2. 問題を回避できる可能性のあるI/Oパターン
       - ディスク装置側を満遍なく利用するために，プロセス単位で出力先ファイルを分ける
       - O_TRUNC指定が不要な場合には，open時に指定しないようにプログラムを修正する
       例： mpiexec -of-proc FILE ./a.out
       （注意）プロセス数が多い場合は，ファイル作成に時間を要することがあります

Workaround for file I/O error due to file system failure at the second-layer storage

Concerning file I/O error due to the file system failure, it has turned out that there would be a chance to prevent I/O accesses from the problem as follows.

1. Layout of I/O target files
    1. Layout which easily faces the problem
       Placing I/O target files in the same directory with the executing module
       e.g., Executing module: /path/to/dir/jobdir/a.out
       I/O target files: /path/to/dir/jobdir/stdout.001 stdout.002 ・・・・stdout.999
    2. Layout which may prevent from the problem
       Placing I/O target files in the different directory from the executing module
       e.g., Executing module: /path/to/dir/jobdir/a.out
       I/O target files: /path/to/dir/jobdir/log/stdout.001 stdout.002 ・・・・stdout.999

2. I/O access patterns
    1. I/O accesses which easily face the problem
       I/O accesses towards a single file from multiple compute nodes
       - Once write accesses exceed disk performance, write requests from compute nodes do not complete.
       - In write accesses for a file opened with "O_TRUNC" mode, write requests are serialized.

       e.g., mpiexec -of FILE ./a.out
    2. I/O accesses which may prevent from the problem
       - File-per-process based I/O accesses in order to leverage potential of disk systems
       - Not to use "O_TRUNC" mode if it is not necessary.

       e.g., mpiexec -of-proc FILE ./a.out

富岳サポートサイト「コミュニティ」スペースの開設

富岳の利用者間で相互に情報交換をはかっていただくためのスペースとして、富岳サポートサイト内に「コミュニティ」を開設いたしました。富岳を利用する上で他の利用者の方からの意見を参考までにききたい、あるいはご自身で知り得た情報を他の利用者と共有したい、等の互助目的でご自由に活用いただきたいと考えています。

「コミュニティ」では主に技術的な話題が多くなるとの想定で、以下のトピックを設けました。

システム
ジョブ関連
ソフトウェア
プログラミング
会議告知等一般

「コミュニティ」の各トピック毎に形式的な管理者がつきますが、管理者は各トピックの専門家ではなく、基本的には利用者間の情報交流を見守らせていただき、場合によってはチケット発行を提案する等の最小限の助言をさせていただく可能性があります。

利用者からの技術的な問い合わせは、これまで通り当サイトの「新しいお問い合わせ」および「過去のお問い合わせ」メニューが窓口となりますので、引き続きご活用願います。「お問い合わせ」メニューから寄せられる各種ご相談・ご質問に対しては階層的なサポート体制をとって、適切なサポートのご提供を目指しています。「お問い合わせ」メニューへ寄せていただいた各種ご相談・ご質問とその回答内容の内で、共有する価値があると判断されるものについては漸次FAQへ掲載する作業を進める方針としていますが、まだ迅速な記事掲載まで至っていないことから、FAQと補完的に「コミュニティ」スペースもご利用いただければと考えています。
「コミュニティ」スペースの開設は運用側でも初めての取り組みですので、当スペースについてのご意見・ご要望等も「コミュニティ」へ発言いただければ参考にさせていただきます。
尚「コミュニティ」スペースでの使用言語は自由となりますが、英語圏の利用者も多いことから、意見を求める様なコメントは英語で書き入れていただく方が、反応が多いかもしれません。

"Community" space is now available in Fugaku Support Site

To help exchanging information among users of Fugaku, we have started "community" inside the Fugaku support site. Please feel free to use "community" for the purpose of mutual assistance in using Fugaku, such as asking opinions from other users, or sharing information with other users.

We have set up the following topics in the "community".

System
Job
Software
Programming
Meeting announcements and general

There is a formal administrator in each of the topics. The administrator is not an expert of the topic, and basically watches over information exchanges among users, and may provide minimal advice such as issuing a ticket if appropriate.

For technical inquiries from users to support staffs, please continue to use the "New Inquiry" and "Past Inquiries" menus on this site. We aim to provide appropriate support by adopting a hierarchical support system for various questions and requests received from the "New Inquiry" menu. Of the various questions and answers that have been processed through the "New Inquiry" menu, we will reformat their contents and post them in FAQ sections, if judged worth sharing. We hope that the "community" space can be also used as a complement to the FAQ.

Since starting the "community" space is a new experience to the operation side as well, we will appreciate your opinions and requests regarding as a comment to a topic.

Please note that the space is opened as language free, and both of English comments and Japanese comments can be posted.

大規模ジョブ実行について

通常運用とは別に1/2規模（82,944ノード）と全系規模（152,064ノード）のジョブを実行する大規模ジョブ実行期間を設けます．全系規模（152,064ノード）のジョブ実行は別途公募があります．

- 実施期間
  - 1/2規模：毎月第3火曜日15:00から次の金曜日 15:00まで

大規模ジョブ実行を希望する利用者の方は事前審査が必要です．この事前審査は，ジョブ実行時の消費電力が電力設備の供給能力を超過することを防止する目的で実施します．

- 事前審査
  - 1/2規模
    - 過去に1/2規模以上で実行したことがあるジョブはその際のジョブIDを，新規ジョブは，投入予定のジョブを事前に以下の条件で実行し，そのジョブIDを審査受付期間内に審査窓口へ連絡してください．
    - 審査用ジョブの要件
      - 実行規模：9,216～12,288ノード（24～32ラック）
      - 実行時間：IO処理の時間を除いた実行時間が10分以上
- 審査受付期間
  - 1/2規模：実施当月の第3月曜日10:00まで（但し，実施当月の第3月曜日が休日の場合は，同月の第2金曜日10:00まで）
- 審査窓口
  - https://fugaku.zendesk.com/hc/ja/

Execution of large-scale jobs

In addition to normal operation, Fugaku has a large-scale job execution period for executing half size of full nodes (-82,944 nodes) and almost full nodes scale (-152,064 nodes) jobs. Call for proposals of full nodes scale job execution.

- Schedule
  - half size of full nodes scale : every month 3rd Tuesday 15:00 - 3rd Friday 15:00 (JST)

Users who wish to execute jobs during these periods need to undergo a preliminary examination and obtain permission to execute the job. This preliminary examination is carried out for the purpose of preventing the power consumption of the job at the time of job execution from the supply capacity of the power equipment.

- Preliminary examination
  - half size of full nodes scale:
    - For jobs that have been executed on a scale of 1/2 or more in the past, please inform the job ID that was executed in the past to the examination office. For new jobs, please execute the job to be submitted in advance under the following conditions, and inform the job ID to the examination office.
    - Review job requirements
      - Execution scale : 9,216 - 12,288 nodes (24 - 32 racks) Execution time: Execution time excluding IO processing time is 10 minutes or more
- Examination acceptance period
  - half size of full nodes scale : Until 10:00 on the 3rd Monday of the implementation month (JST) (Until 10:00 on the 2nd Friday of the implementation month when the 3rd Monday is a holiday)
- Examination office / contact information
  - https://fugaku.zendesk.com/hc/en-us/

現在発生しているファイルシステム障害とジョブへの影響について

[事象]
2021年12月保守以降、ファイルシステムの障害によるレスポンス低下のため、ログインノードの応答遅延やジョブの経過時間制限値超過による終了などの事象が複数回発生しております。
本事象の背景には、原因が異なる3つの障害を確認しています。

障害1. FEFSの障害によるファイルシステムサーバのダウン
障害2. ファイルシステムの通信で異常発生
障害3. ファイルシステムサーバの切替時の高負荷

障害1については2022年2月保守での修正適用をもって解消しました。
障害2については原因調査中です。暫定対処済み
障害3については2023年4月保守での修正適用をもって解消しました。

[回避方法]
ありません。
本事象は間欠的なものであるため、ジョブが経過時間制限値超過で終了した場合はジョブを再投入いただくようお願いします。

[修正時期]
障害1については2022年2月保守にて修正適用が完了しました。
障害2については原因調査中のため、修正時期は未定です。
障害3については2023年4月保守にて修正適用が完了しました。

Current filesystem failures and their impact on jobs

[Phenomenon]
Since the system maintenance in December 2021, due to poor response caused by a file system failure, delayed response of login nodes and termination of jobs due to exceeded elapsed time limits occurred multiple times.
Three failures with different causes are identified in the background of this incident.

Failure 1. File system server down due to FEFS failure
Failure 2. An error during file system communication
Failure 3. High load when switching the file system server

Failure 1 was resolved by the revision in the system maintenance in February 2022.
The cause of failure 2 is under investigation. We applied a temporary measure.
Failure 3 was resolved by the revision in the system maintenance in April 2023.

[Preventive measures]
There is no preventive measures.
The phenomenon occurs intermittently, so please resubmit the job if it ends with the exceeded elapsed time limit.

[Scheduled time to fix]
Failure 1 was fixed by the revision in the system maintenance in February 2022.
Failure 2 is under investigation, so the time to fix is undecided.
Failure 3 was fixed by the revision in the system maintenance in April 2023.