

# DDBJスパコン概要説明

2019.03.25 2018年度第2回先進ゲノム支援情報解析講習会

国立遺伝学研究所 DDBJセンター  
小笠原 理

# 目次

- \* 遺伝研スパコンの目的
- \* 遺伝研スパコンの構成
- \* スパコンの機能の紹介
  - \* (1) Univa Grid Engine, (2) Singularity, (3) ハイブリッドクラウド、(4) 大容量データのやりとり、(5) 開発環境について
- \* アカウント申請、課金、お問い合わせ先

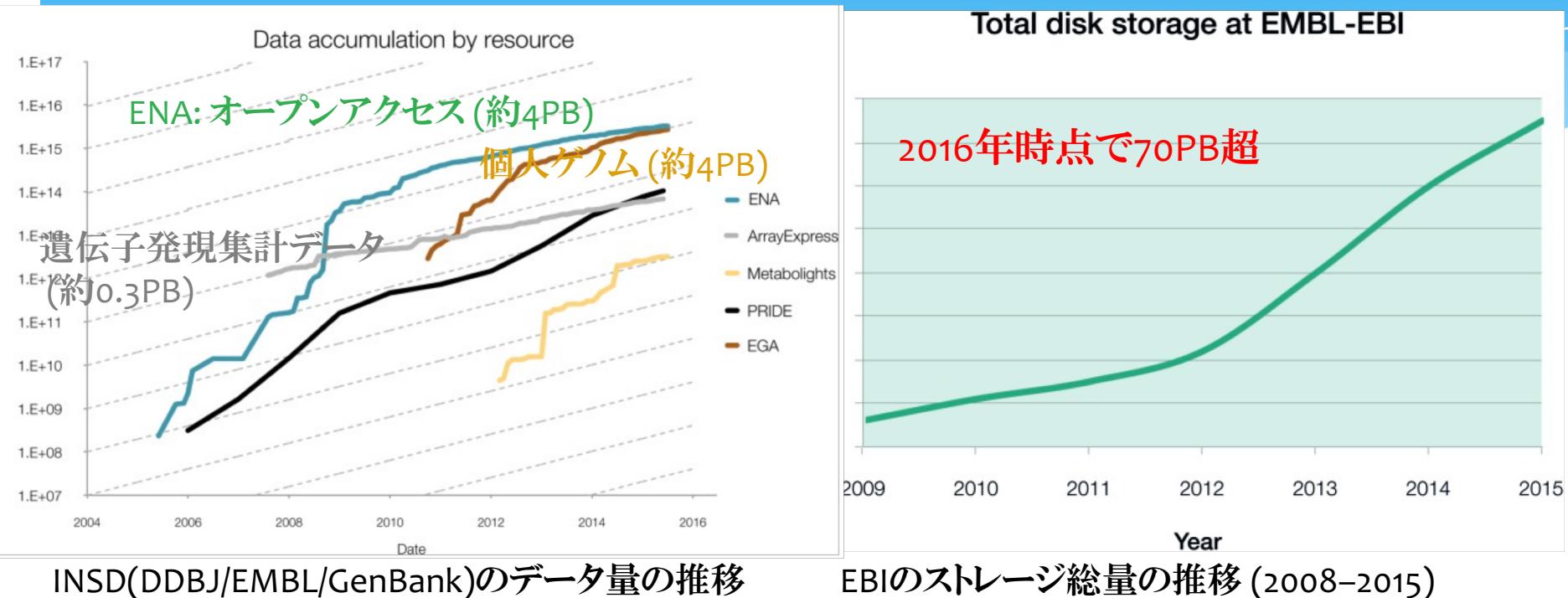
# 遺伝研スパコンの目的

1. 国際塩基配列データベース構築のための計算基盤
2. 大学共同利用機関法人としての生命医学系研究者への計算資源の提供

2019年3月稼働開始の第5期遺伝研スパコンは基本的な構成や使い方は2012年導入の第4期遺伝研スパコンを踏襲しているが、新たに日本人個人ゲノムデータのデータベース構築と計算資源の提供へ対応した。

# データ量とストレージ保有の推移

[Nucleic Acids Res.](#). 2016 Jan 4; 44(Database issue): D20–D26.



2030年までに8千3百万人の希少疾患患者関連、2億4千8百万人のがん患者のゲノムが読まれると予想されている。

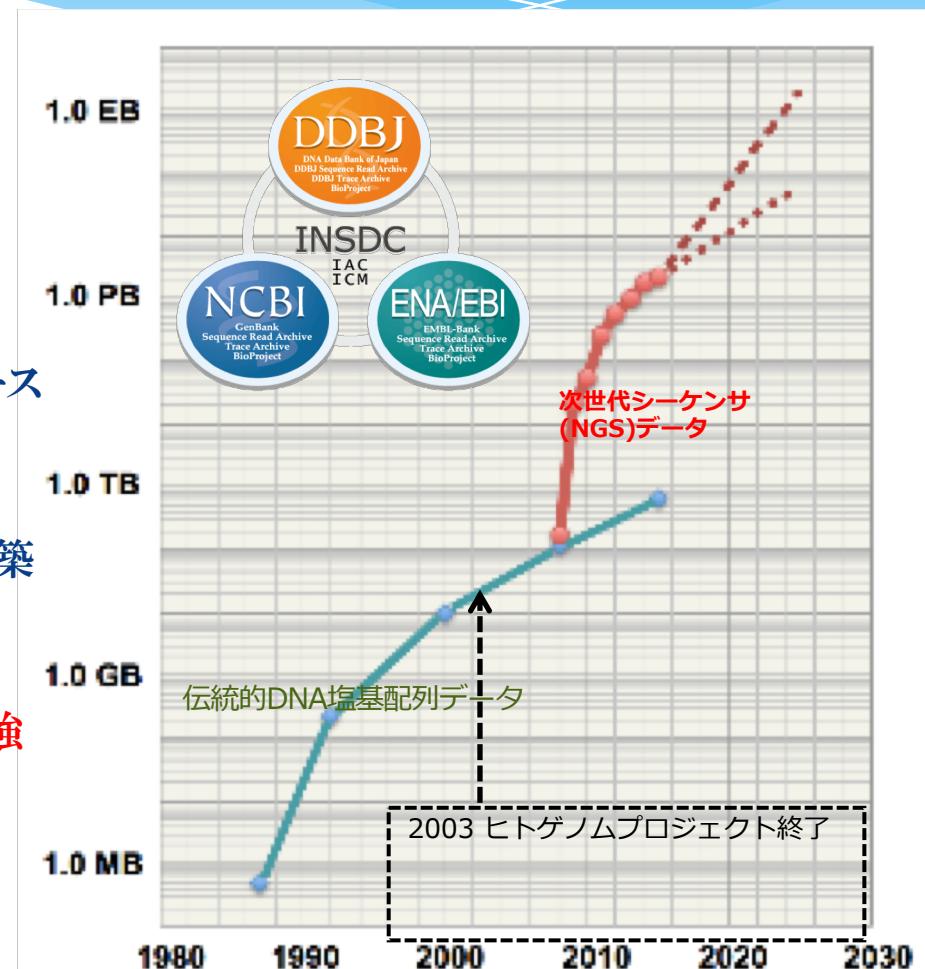
Knoppers and Joly, Introduction: the why and whither of genomic data sharing , Human Genetics (2018) 137:569

1EB (whole exome)~10EB(whole genome)?

# 国際塩基配列データベース (International Nucleotide Sequence Database ; INSD)

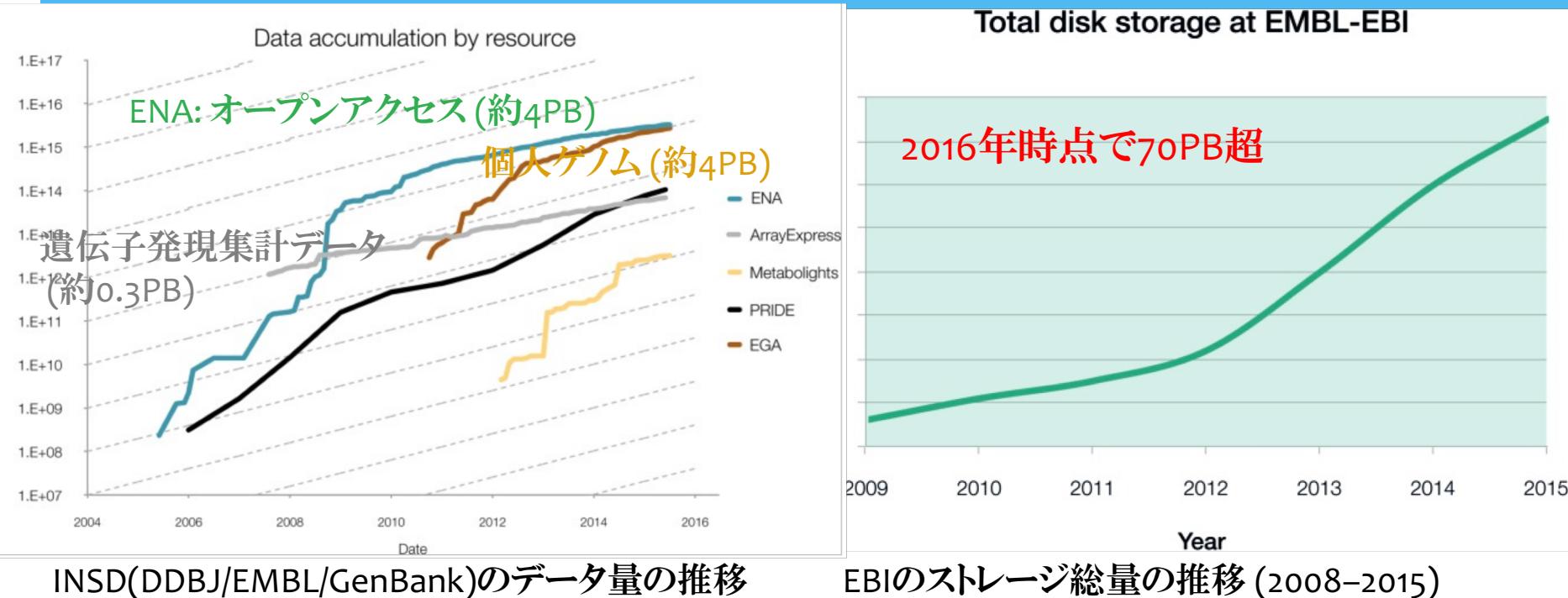
- \* 1977年 塩基配列決定法 (Sanger法)
- \* 1981年 EMBL(欧) 1982年 GenBank (米)  
設立
- \* 1984年～1987 遺伝研にDDBJ設置。
- \* 1993年 日米欧特許庁データを  
DDBJ/EMBL/GenBankから公開開始
- \* 2007年 遺伝研スパコン(第3期)
- \* 2008年 次世代シーケンサーのデータベース  
構築開始(SRA, ERA, DRA)
- \* 2012年 遺伝研スパコン更新(第4期)
- \* 2013年 個人ゲノムデータベース (JGA)構築  
開始
- \* 2018年2月 スパコン課金サービスの開始
- \* 2018年3月 遺伝研スパコンストレージ増強
- \* 2019年3月 遺伝研スパコン更新(第5期)

国際塩基配列データベース  
(DDBJ)データ量推移



# データ量とストレージ保有の推移

[Nucleic Acids Res.](#). 2016 Jan 4; 44(Database issue): D20–D26.



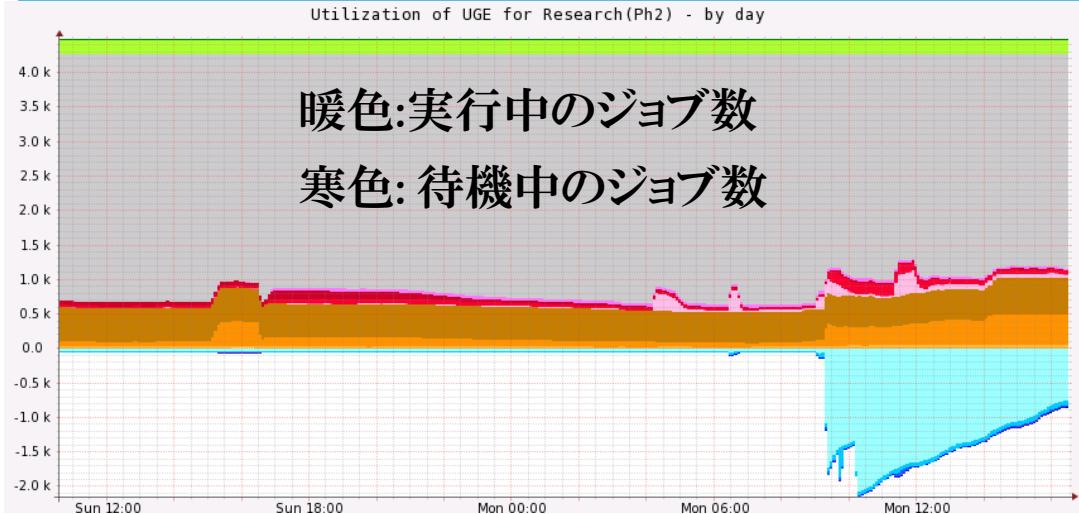
INSD(DDBJ/EMBL/GenBank)のデータ量の推移

EBIのストレージ総量の推移 (2008–2015)

2030年までに8千3百万人の希少疾患患者関連、2億4千8百万人のがん患者のゲノムが読まれると予想されている。

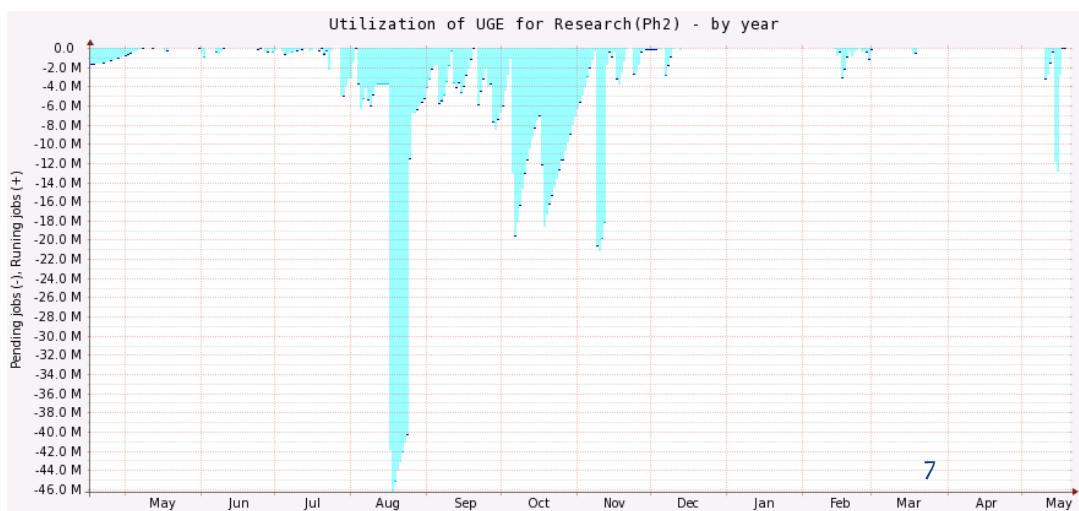
Knoppers and Joly, Introduction: the why and whither of genomic data sharing , Human Genetics (2018) 137:569

# 遺伝研スパコンの混雑状況



2012年導入の前スパコンは、  
2012年導入分(Phase 1)と  
2014年中間増強分(Phase 2)が  
あったが、  
左図はPhase 2のみの図。

平均すると常に3000程度のジョブが  
流れている状態



年間の統計(2018年)

# 生命医学系のデータ解析のための要件

(1) 整数演算の多数ジョブ (2) 大容量ストレージとInput/Output (3) セキュリティ (4) Pipeline

伝統的スパコン(京  
など)

## タスクによる要件の違い

遺伝研スパコン

産総研ABCiなど

### シミュレーション

- 64bit floating point
- Memory Bandwidth
- Random Access Memory
- Sparse Matrices
- Distributed Memory jobs
- Synchronous I/O  
multinode
- Scalability Limited Comm
- Low Latency  
High Bandwidth
- Large Coherency Domains help sometimes
- O typically greater than I
- O rarely read
- Output is data

### ビッグデータ分析

- 64bit and Integer Important
- Data analysis Pipelines
- DB including No SQL
- MapReduce/Spark
- Millions of jobs
- I/O bandwidth limited
- Data management limited
- Many task parallelism
- Large-data in and Large-data out
- I and O both important
- O is read and used
- Output is data

### Deep Learning

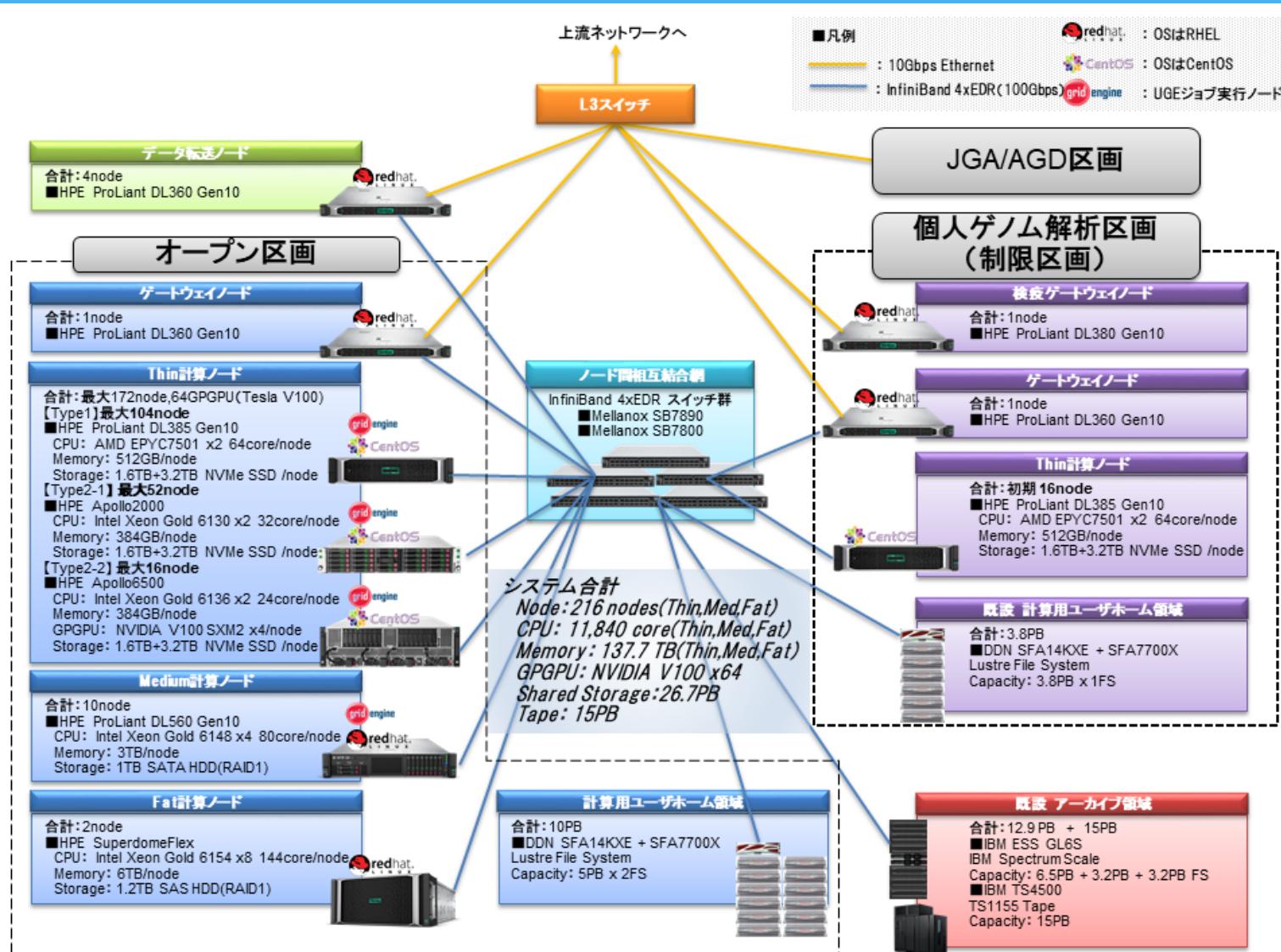
- Lower Precision  
(fp32, fp16)
- FMAC @ 32 and 16 okay
- Inferencing can be 8bit (TPU)
- Scaled integer possible
- Training dominates dev
- Inference dominates pro
- Reuse of training data
- Data pipelines needed
- Dense FP typical SGEMM
- Small DFT, CNN
- Ensembles and Search
- Single Models Small
- I more important than O
- Output is models

Rick Stevens, "A Vision of Exascale: Simulation, Data and Learning"

<https://www.slideshare.net/insideHPC/a-vision-for-exascale-simulation-and-deep-learning>

# 遺伝研スパコン(2019)の構成

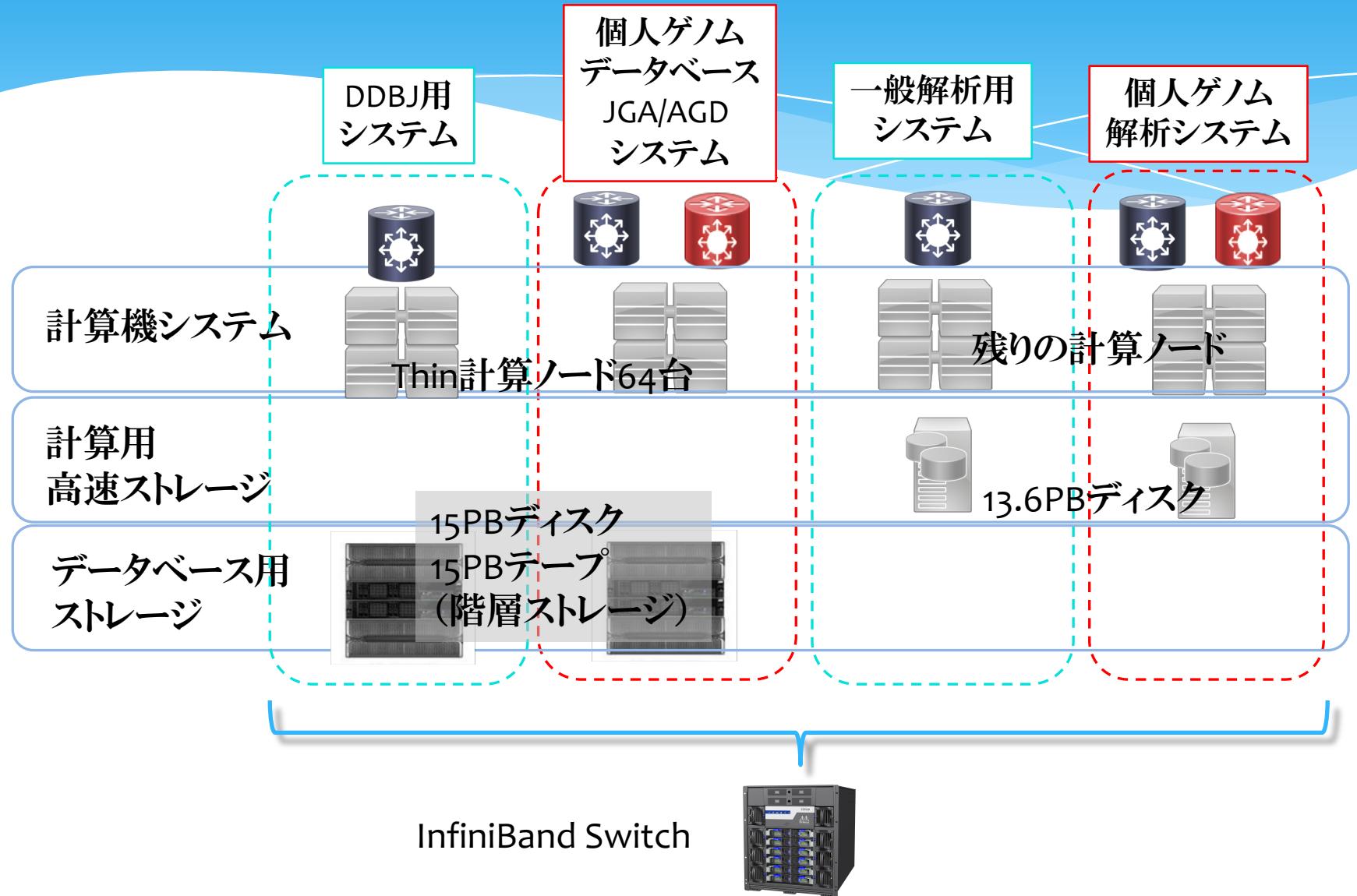
<https://sc2.ddbj.nig.ac.jp/index.php/systemconfig>



# 遺伝研スパコン(2019)構成概要

		2012年導入遺伝研スパコン(第4期)	2019年導入遺伝研スパコン(第5期)	コメント
(1) 計算機システム	分散メモリ型計算機(Thin計算ノード)	計算機システム554台 (約10,800コア)	計算機システム <b>204台</b> (約11,000コア)	204台中1/3がIntel, 2/3がAMD製CPU。
	共有メモリ型計算機(Fat, Medium計算ノード)	10TBメモリ計算機1台 2TBメモリ計算機10台	12TBメモリ計算機1台、 3TBメモリ計算機10台	204台中64台がDDBJ専用。 <b>課金枠は最大で残りの半数(70台)想定。</b>
(2) ストレージシステム	計算用高速ストレージ	合計7PB	合計13.6PB	一般計算用
	データベース用ストレージ	<u>5.6PB</u>	<u>30PB</u>	DDBJ専用(又は課金)
(3) ノード間相互結合網(ネットワークシステム)	InfiniBand 4×QDR (32Gbps),		InfiniBand 4×EDR (100Gbps)	Fat tree IP over IB

# 各サブシステムへの割り当ては可変



# Thin計算ノードの種類

Thin計算ノードの1/3はIntel, 2/3はAMDのCPUを搭載することでコア数を稼いでいる。  
GPU搭載ノード(HPE Apollo6500)は16台。各ノードにGPUが4基搭載されている。(合計64台)

Thin計算ノード種別	機種	CPU種	ノード当りCPU数	ノード数
Type1 25GbE無し	HPE ProLiant DL385 Gen10	AMD EPYC	2	72
Type1 25GbE有り	HPE ProLiant DL385 Gen10	AMD EPYC	2	64
Type2 GPU無し	HPE Apollo 2000 Gen10	Intel Xeon	2	52
Type2 GPU有り	HPE Apollo 6500 Gen10	Intel Xeon	2	16
			合計	204



# Thin計算ノード(GPUなし)

## ■Thin計算ノード Type1 25GbE有り

AMD製CPU EPYCを搭載したHPE ProLiant DL385 Gen10を、64ノード構成で提案致します。



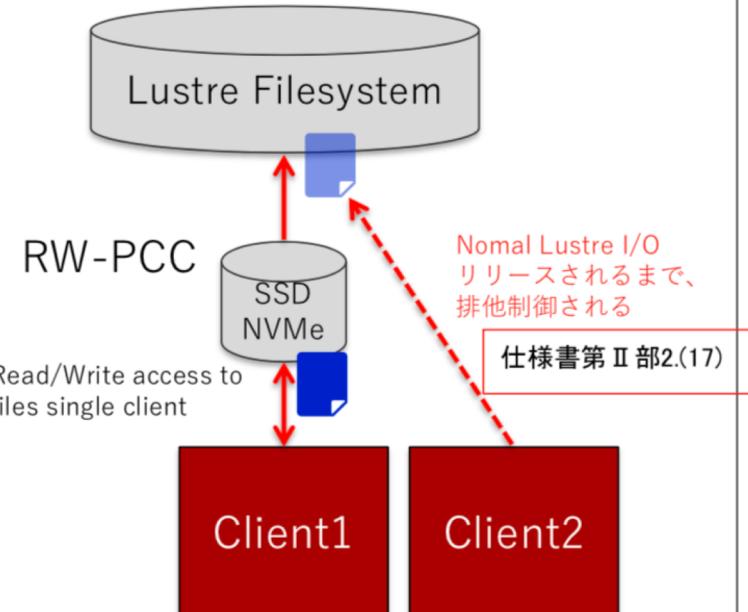
## ■提案構成概要(ノード当り)

構成要素	構成	員数	備考
CPU	AMD EPYC 7501 (2.0GHz, 32core)	2 基	
メモリ	32GB DDR4-2666 DIMM	16 枚	ノード総量:512GB
ディスク	1.6TB NVMe SSD	1 基	
	3.2TB NVMe SSD	1 基	
ネットワーク	InfiniBand 4xEDR	2 ポート	
	25Gbps Ethernet	2 ポート	
	1Gbps Ethernet	4 ポート	内1ポートはiLOと共有可

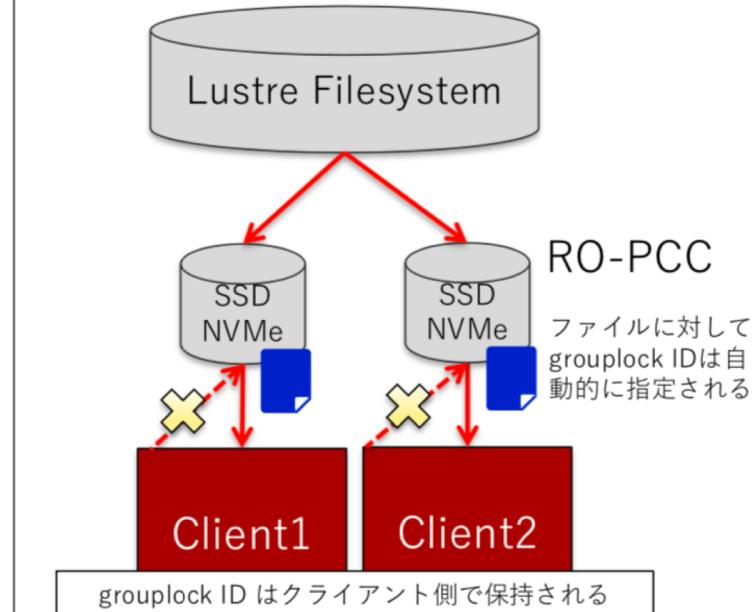
# SSD使用方法

## (1) 高速ストレージ(Lustre FS)に対するキャッシュ

ReadWrite Persistent Client Cache (RW-PCC)は、HSM データマネージメント機能を用いて実装されています。ファイルが Readwrite PCC のファイルシステム領域に置かれるとき、そのファイルは単一のクライアントのみI/O が可能です。そのファイルを使用しているジョブを終了しリリースされるまで、他のクライアントはそのファイルにアクセスできません。そのデメリットよりもRW-PCC を有効化して、単一のクライアントでDB のようなランダムI/O 集約型のアプリケーションに最適化するメリットがあります。



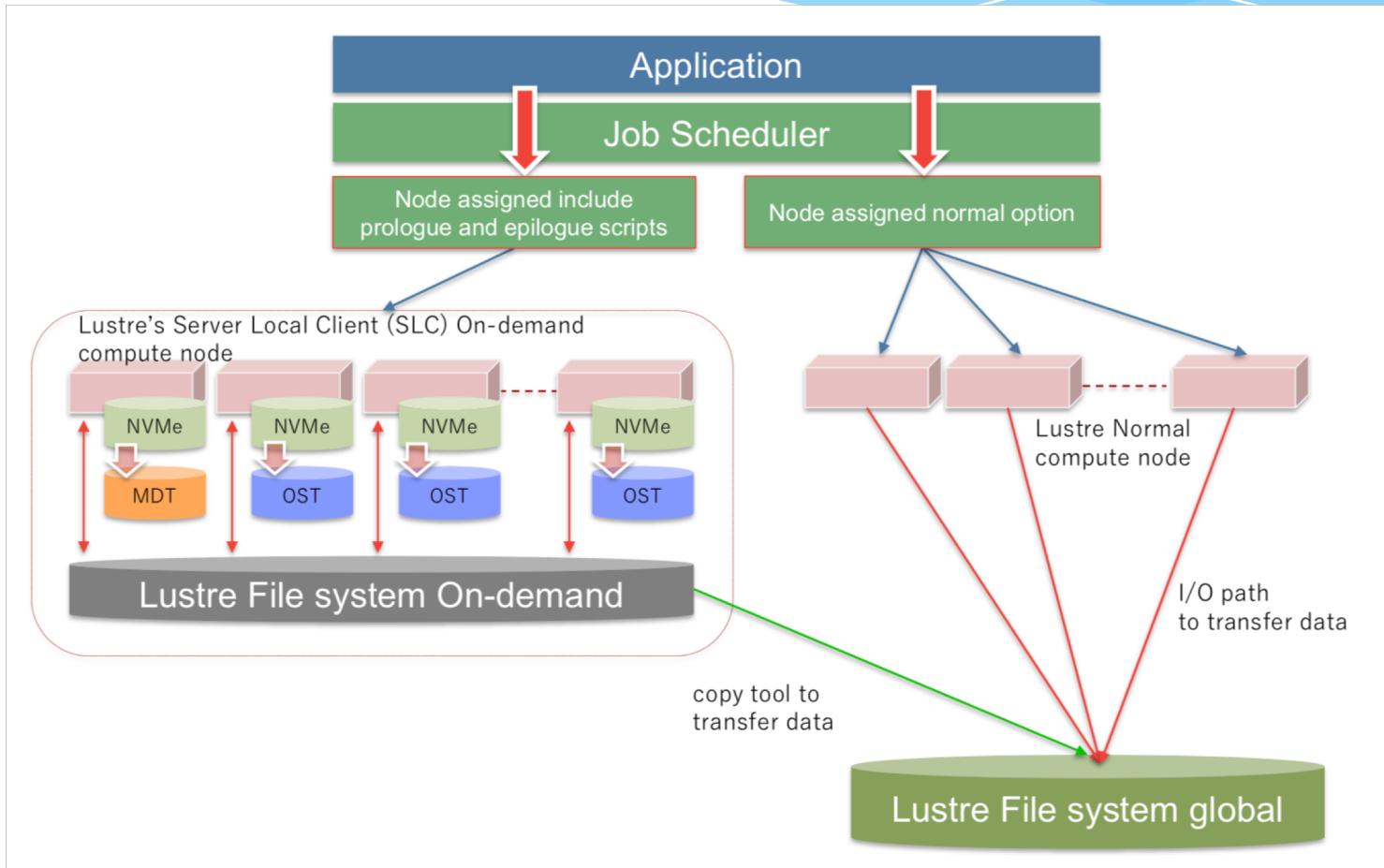
Readonly Persistent Client Cache(RO-PCC)は、ReadWrite Persistent Client Cache と同じフレームワークを利用していますが、HSM 機能によるポリシーはRO-PCC の動作時には使用されません。その代わりに、ファイルを他のクライアントから更新されないように保護するため、grouplock 機能を使います。ファイルが RO-PCC にある間、grouplock ID はクライアント側で保持され、同じID を持つクライアントは同じファイルをRead することが可能です。この設定により複数のクライアントでパラレルリードキャッシュを実現可能とします。



# SSD使用方法

## (2) SSDを束ねてLustre FSを構成する

### Lustre's Server Local Client (SLC) On-demand



# Thin計算ノード(GPUあり)

## ■Thin計算ノード Type2 GPU有り

Intel製CPU Xeon、及びNVIDIA製GPUアクセラレータを搭載したHPE Apollo 6500 Gen10を、16ノード構成で提案致します。

Apollo 6500は、4UサイズでGPUアクセラレータを最大8基搭載可能です。



## ■提案構成概要(ノード当り)

構成要素	構成	員数	備考
CPU	Intel XeonG 6136 (3.0GHz, 12core)	2 基	
メモリ	16GB DDR4-2666 DIMM	12枚	ノード総量:384GB
ディスク	1.6TB NVMe SSD	1 基	
	3.2TB NVMe SSD	1 基	
GPU	NVIDIA TESLA V100 SXM2	4 基	
ネットワーク	InfiniBand 4xEDR	2 ポート	
	1Gbps Ethernet	4 ポート	内1ポートはiLOと共有可

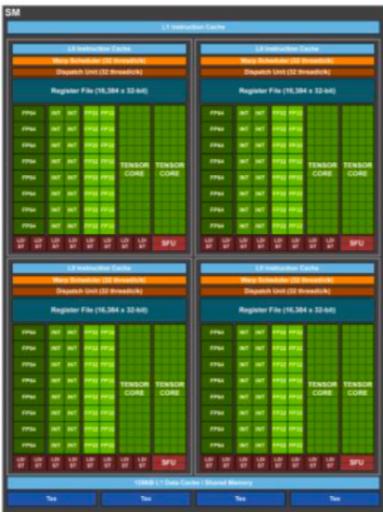
# NVIDIA Tesla V100

## 最先端のデータセンター向けGPU

NVIDIA® Tesla® V100 はAI、HPCそしてグラフィクス処理を高速化する史上最高のデータセンター向け GPU です。

最新のNVIDIA Volta™アーキテクチャにより、Tesla V100は1基で最大100CPU分のパフォーマンスを発揮し、かつては不可能と考えられていた課題に取り組むデータサイエンティスト、研究者、エンジニアを強力に支援します。

## VOLTA GV100 SM



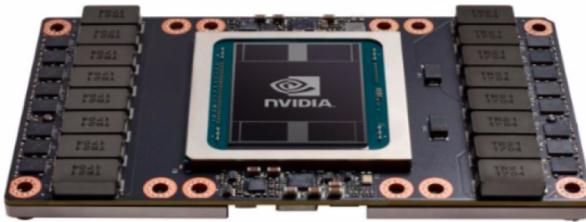
### 生産性のために刷新された設計

大容量・高速なL1 キャッシュ・テンソル演算の加速化

GV100	
FP32 units	64
FP64 units	32
INT32 units	64
Tensor Cores	8
Register File	256 KB
Unified L1/Shared memory	128 KB
Active Threads	2048

- 完全に新しいISA
- スケジューラを倍増
- 簡素化された命令発行ロジック
- 改善されたSIMT モデル

もっとも容易に  
プログラミングできるSM



Tesla V100 SXM2外観

仕様	Tesla V100 SXM2
GPUアーキテクチャ	NVIDIA Volta
NVIDIA Tensorコア	640
NVIDIA CUDAコア	5,120
倍精度演算性能	7.8 TFLOPS
単精度演算性能	15.7 TFLOPS
行列演算性能	125 TFLOPS
GPUメモリ	32/16 GB HBM2
メモリ帯域幅	900 GB/sec
ECC	対応
GPU間接続帯域	300 GB/sec
システム接続	NVIDIA NVLink
フォームファクタ	SXM2最大消費電力
最大消費電力	300 W
冷却方式	パッシブ(冷却ファンなし)
対応計算API	CUDA, DirectCompute, OpenCL™, OpenACC

# 遺伝研スパコンの機能の紹介

1. Univa Grid Engine
2. Singularityコンテナの利用
3. ハイブリッドクラウド
4. 大容量データのやりとり
5. 開発環境について

# Univa Grid Engine

<https://web.archive.org/web/20151018220217/https://blogs.oracle.com/templedf/?page=1>

<https://blogs.oracle.com/templedf/?page=1>

3 captures category: Grid

22 Oct 2012 - 18 Oct 2015

Go DEC OCT NOV  
Ring... 18 2013 2015 201 ▾ Abo

Monday Nov 30, 2009

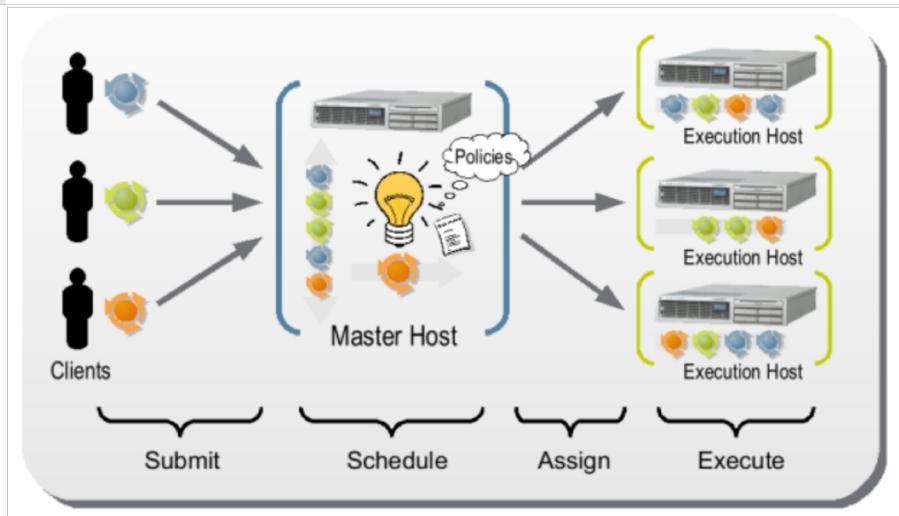
## Sun Grid Engine for Dummies

By templedf on Nov 30, 2009

I've recently been asked for a **really** introductory doc on [Sun Grid Engine](#), and I was dismayed to realize that there really isn't anything like that out there. Even the [Beginner's Guide](#) I wrote has some fairly high expectations of the reader's experience level. So, this post will be my attempt at a truly introductory introduction to Sun Grid Engine.

### Let's Begin at the Beginning

Andreas Haas  
Richard Hierlmeier  
Tim Bray  
Constantin Gonzalez  
Bryan Cantrill  
Dan Baigent  
Daniel Green  
Miha Ahronovitz  
Scott Dickson



Jobの種類は4種類

1. Interactive job
2. Batch job
3. Array job
4. Parallel (MPI) job

# 遺伝研スパコンに於ける UGEキュー構成

## 研究用medium計算ノード

queue name	# of nodes	slot (スロット数)				memory (GB)		
		in use / total		disabled	in use	required / total		
medium.q	10	186	/ 800	23%	0	2688	9816	/ 30720 31%

## 研究用thin計算ノード

\*gpu.qとshort.qは実ノードが同じため、in use(使用メモリ量)が同じ値となります。

queue name	# of nodes	slot (スロット数)				memory (GB)		
		in use / total		disabled	in use	required / total		
epyc.q	66	2117	/ 4224	50%	192	3935	18541	/ 33792 54%
gpu.q	14	0	/ 112	0%	0	262	0	/ 2688 0%
intel.q	46	16	/ 1472	1%	0	342	192	/ 17664 1%
login.q	6	43	/ 384	11%	0	80	344	/ 3072 11%
login_gpu.q	2	0	/ 48	0%	0	16	0	/ 768 0%
short.q (-l short)	14	1	/ 224	0%	0	262	10	/ 2688 0%
	148	2177	/ 6464	33%	192	4897	19087	/ 60672 31%



# Singularity

<https://singularity.lbl.gov/>

Singularity enables users to have full control of their environment. Singularity containers can be used to package entire scientific workflows, software and libraries, and even data.

This means that **you don't have to ask your cluster admin to install anything for you** - you can put it in a Singularity container and run.

- Encapsulation of the environment
- Containers are image based
- No user contextual changes or root escalation allowed
- No root owned daemon processes

# Singularity : HPC用コンテナ

ユーザーは自分が好きなOS、ミドルウェアなどの環境ごとプログラムをコンテナに詰めて自分の環境からスパコンにシステムを移行できる。

The screenshot shows a PLOS ONE research article titled "Singularity: Scientific containers for mobility of compute". The article is open-access and peer-reviewed. It was published on May 11, 2017, with the DOI <https://doi.org/10.1371/journal.pone.0177459>. The page features a navigation bar with "Publish", "About", and "Browse" buttons. Below the title, there are sections for "Article", "Authors", "Metrics", "Comments", and "Related Content". The "Abstract" section contains the following text:

Here we present Singularity, software developed to bring containers and reproducibility to scientific computing. Using Singularity containers, developers can work in reproducible environments of their choosing and design, and these complete environments can easily be copied and executed on other platforms. Singularity is an open source initiative that harnesses the expertise of system and software engineers and researchers alike, and integrates seamlessly into common workflows for both of these groups. As its primary use case,

1. ユーザーは自分の計算機の上でroot権限でコンテナを作る。
  - sandboxモード
  - Writable containerモード (\*.img ファイル)
2. ユーザーは、作ったコンテナをプロダクションコンテナファイル (\*.simg または \*.img) に固めてスパコン上にコピーする。
  - \*.simg ファイルはリードオンリー
  - コンテナの中からは \$HOME, /dev/, /tmp, /proc, /sys が最初から見えている。
3. ユーザーはコンテナをスパコン上で実行する。
  - Web アプリなどはスパコン SE が手伝う。

Table 1. Container comparison.

	Singularity	Shifter	Charlie Cloud	Docker
Privilege model	SUID/UserNS	SUID	UserNS	Root Daemon
Supports current production Linux distros	Yes	Yes	No	No
Internal image build/bootstrap	Yes	No*	No*	No***
No privileged or trusted daemons	Yes	Yes	Yes	No
No additional network configurations	Yes	Yes	Yes	No
No additional hardware	Yes	Maybe	Yes	Maybe
Access to host filesystem	Yes	Yes	Yes	Yes**
Native support for GPU	Yes	No	No	No
Native support for InfiniBand	Yes	Yes	Yes	Yes
Native support for MPI	Yes	Yes	Yes	Yes
Works with all schedulers	Yes	No	Yes	No
Designed for general scientific use cases	Yes	Yes	No	No
Contained environment has correct perms	Yes	Yes	No	Yes
Containers are portable, unmodified by use	Yes	No	No	No
Trivial HPC install (one package, zero conf)	Yes	No	Yes	Yes
Admins can control and limit capabilities	Yes	Yes	No	No

In addition to the default Singularity container image, a standard file, Singularity supports numerous other formats described in the table. For each format (except directory) the suffix is necessary for Singularity to identify the image type.

\*relies on Docker

\*\*with security implications

\*\*\*depends on upstream

<https://doi.org/10.1371/journal.pone.0177459.t001>

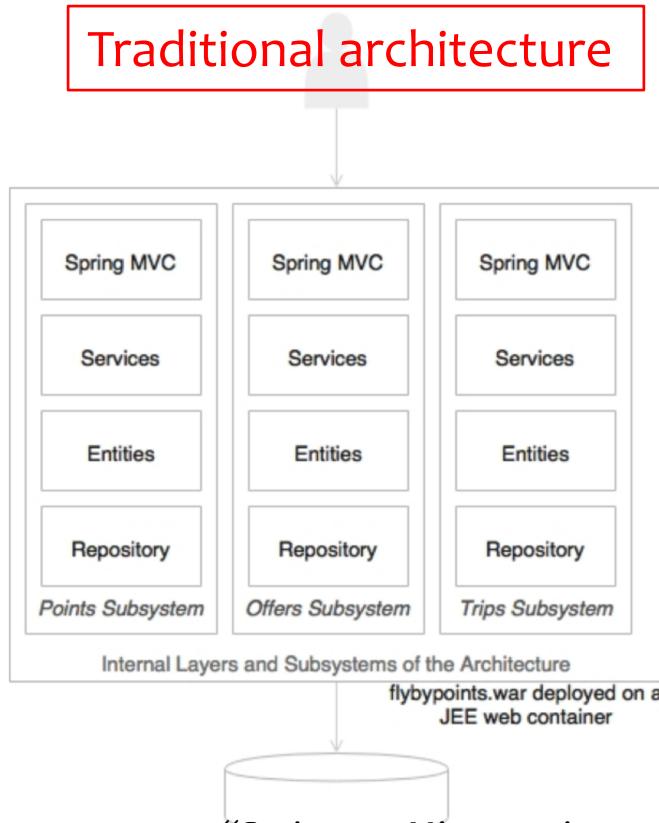
Kurtzer GM, Sochat V, Bauer MW (2017) Singularity: Scientific containers for mobility of compute. PLOS ONE 12(5): e0177459.

<https://doi.org/10.1371/journal.pone.0177459>

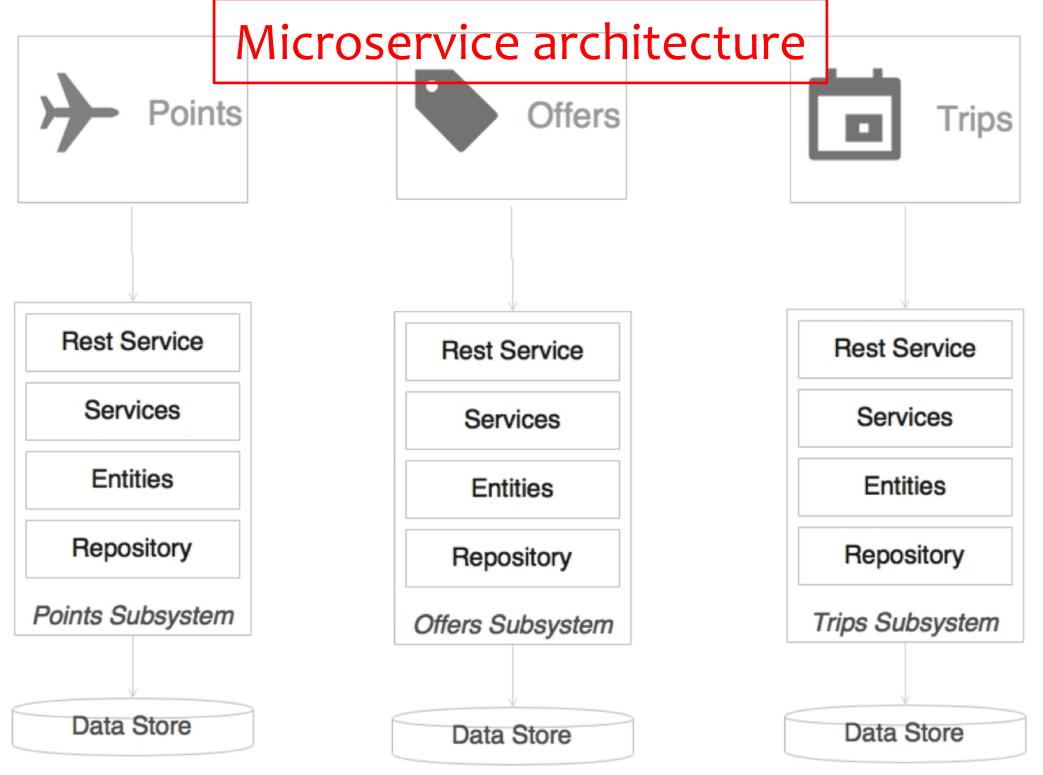
<http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0177459>

# コンテナの作法: Microservice

Traditional architecture



Microservice architecture



“Spring 5.0 Microservices, 2<sup>nd</sup> ed.” by Rajesh R V Published by Packt Publishing, 2017

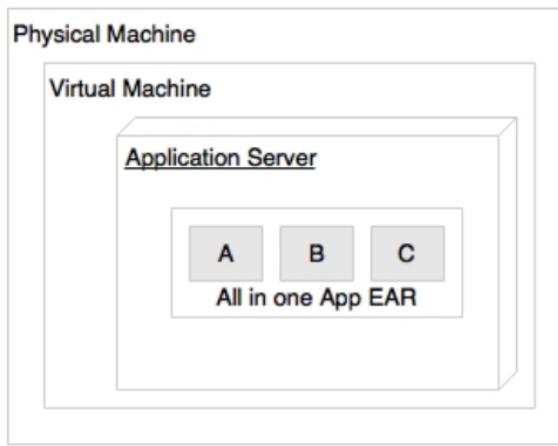
Microserviceは機能ごとに独立の実行ファイルとして動作する。(DBMSやWebサーバーなどのミドルウェアも実行ファイルの中に内蔵する。)

- デプロイメント、スケールアウトが容易(パブリッククラウドでも遺伝研スパコンでもデプロイの手間は同じ)
- 機能の追加更新が容易
- 各microservice間の通信はRESTやRabbitMQなどで行う。(疎結合、言語非依存・混在可能)

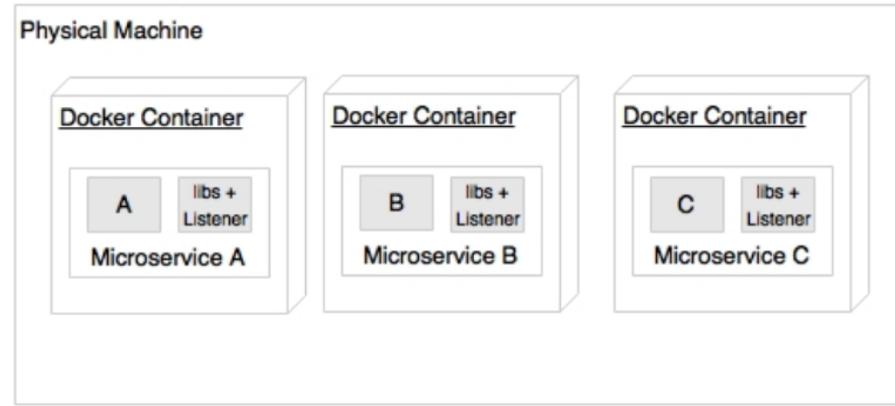
# Containers and Microservices

Container technologies such as Docker also help us keep the infrastructure footprint as minimal as possible compared to hypervisors such as VMWare or Hyper-V.

EAR:  
Enterprise  
ARchive file  
of Java EE



Traditional Deployment



Microservices Deployment

As shown in the preceding diagram, microservices are typically deployed in Docker containers, which encapsulate the business logic and needed libraries. This helps us quickly replicate the entire setup on a new machine or on a completely different hosting environment or even to move across different cloud providers. As there is no physical infrastructure dependency, containerized microservices are easily portable.

# Bioinformatics関連ツールについては Singularityコンテナイメージを置いています

Index of /singularity/		
<a href="#">..</a>		
<a href="#">abricate:0.2--0</a>	14-Dec-2017 22:25	238702623
<a href="#">abricate:0.2--1</a>	19-Jan-2018 21:48	285487135
<a href="#">abricate:0.3--0</a>	19-Jan-2018 21:49	285290527
<a href="#">abricate:0.4--pl5.22.0_0</a>	14-Dec-2017 22:25	215334943
<a href="#">abricate:0.5--pl5.22.0_0</a>	14-Dec-2017 22:26	426184735
<a href="#">abricate:0.7--pl5.22.0_0</a>	14-Dec-2017 22:27	599994399
<a href="#">abrujin:2.1b--py27_0</a>	14-Dec-2017 22:27	42717215
<a href="#">abundancebin:1.0.1--0</a>	17-Dec-2017 08:33	4280351
<a href="#">abyss-k128:2.0.1--boost1.60_0</a>	14-Dec-2017 22:30	162574367
<a href="#">abyss-k128:2.0.1--boost1.61_0</a>	14-Dec-2017 22:30	121327647
<a href="#">abyss-k128:2.0.2--boost1.64_0</a>	14-Dec-2017 22:31	405561375
<a href="#">abyss-k128:2.0.2--boost1.64_1</a>	14-Dec-2017 22:32	405561375
<a href="#">abyss-k128:2.0.2--boost1.64_2</a>	14-Dec-2017 22:33	434728991
<a href="#">abyss:1.5.2--boost1.61_0</a>	14-Dec-2017 22:27	24842271
<a href="#">abyss:1.5.2--boost1.61_1</a>	14-Dec-2017 22:28	51752991
<a href="#">abyss:1.5.2--boost1.61_2</a>	14-Dec-2017 22:28	51752991
<a href="#">abyss:1.5.2--boost1.61_3</a>	14-Dec-2017 22:28	85667871
<a href="#">abyss:1.5.2--boost1.61_4</a>	14-Dec-2017 22:28	85659679
<a href="#">abyss:1.9.0--0</a>	14-Dec-2017 22:28	29532191
<a href="#">abyss:1.9.0--boost1.60_1</a>	19-Jan-2018 21:49	29265951
<a href="#">abyss:1.9.0--boost1.61_2</a>	14-Dec-2017 22:28	30806047
<a href="#">abyss:1.9.0--boost1.61_3</a>	14-Dec-2017 22:28	57389087
<a href="#">abyss:1.9.0--boost1.61_4</a>	14-Dec-2017 22:28	57389087
<a href="#">abyss:1.9.0--boost1.61_5</a>	14-Dec-2017 22:29	92807199
<a href="#">abyss:1.9.0--boost1.61_6</a>	14-Dec-2017 22:29	92807199
<a href="#">abyss:2.0.1--boost1.60_0</a>	14-Dec-2017 22:29	31629343
<a href="#">abyss:2.0.1--boost1.60_1</a>	14-Dec-2017 22:29	55260750

ツールの種類としては約2500種類  
バージョン違いを含めると約8000個の  
コンテナイメージが提供されている。

## Bioconda

⇒ Biocontainers (Docker container)  
⇒ Singularity Container と変換されている。

**inutano** 7:28 PM

starについてはイメージが間違っていますね。3)のコードの中で指定しているのは bioconductor の starr ですけど、使いたい STAR はそれじゃないです。  
`star:2.5.3a--0.1` とかがそうです。 (edited)

```
singularity exec /usr/local/biotools/s/star:2.5.3a--0.1 STAR -h
```

これなら通ります

確かに `hisat2:2.0.0beta--py27_0` はperlないよっつって落ちるんですけど

```
singularity exec /usr/local/biotools/h/hisat2:2.1.0--  
py36pl5.22.0_0.1 hisat2 -h
```

これは通るんでhisat2についてはイメージが悪いのかもです。2.0使わないといけないのでなければこっち使ってねと案内するのがよさそうですね。 (edited)

**inutano** 7:34 PM



biocontainers の検索も使い物にならんので、使いたいツールのイメージがどれか分かりにくいのは結構問題ですね。。。現状は <https://quay.io/organization/biocontainers> で検索するのが一番見つけやすいです。。

 quay.io

**Quay**

Quay is the best place to build, store, and distribute your containers.  
Public repositories are always free. (13 kB) ▾

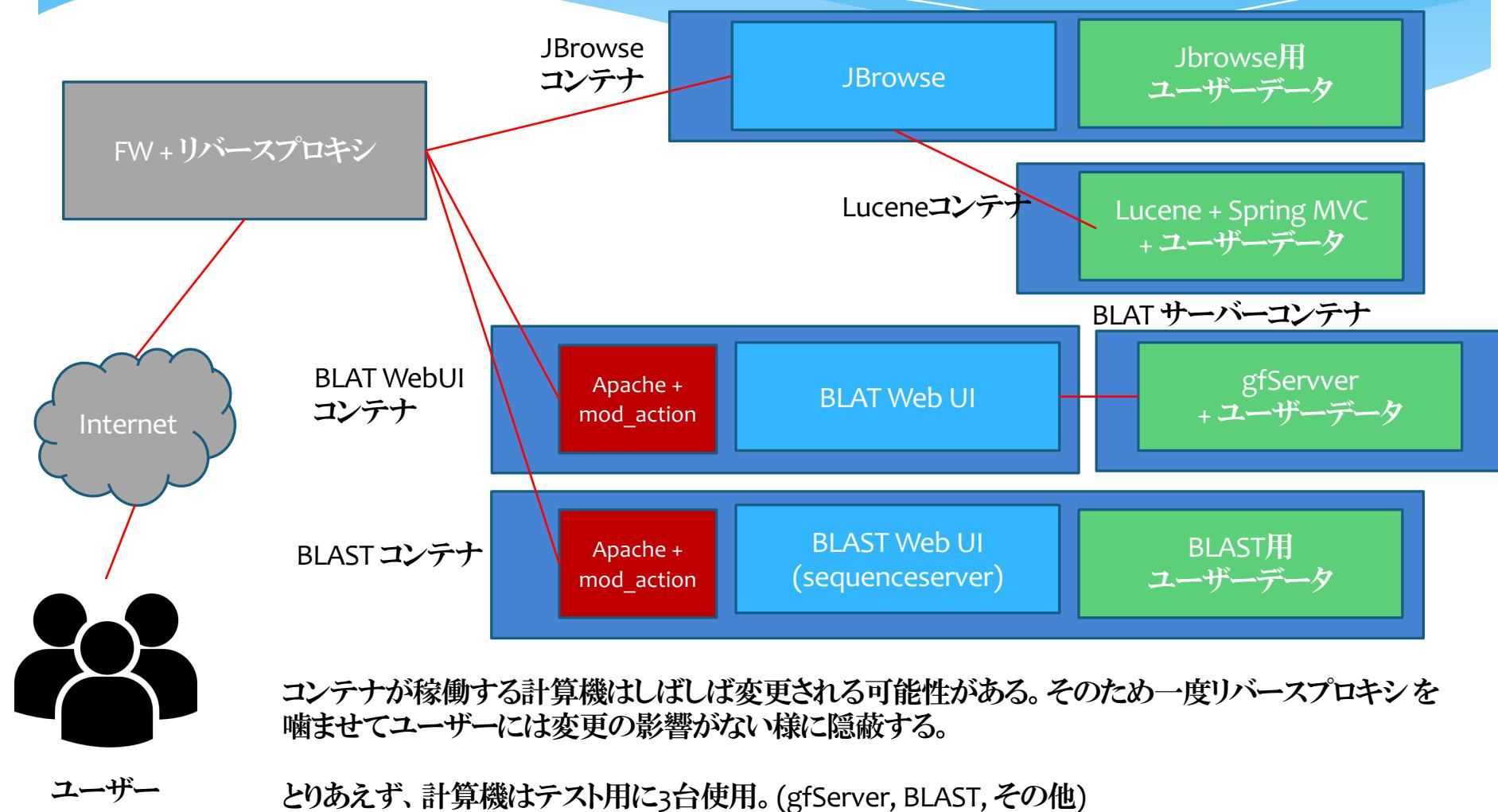


元々のコンテナイメージが悪い場合もある。

コンテナイメージの検索

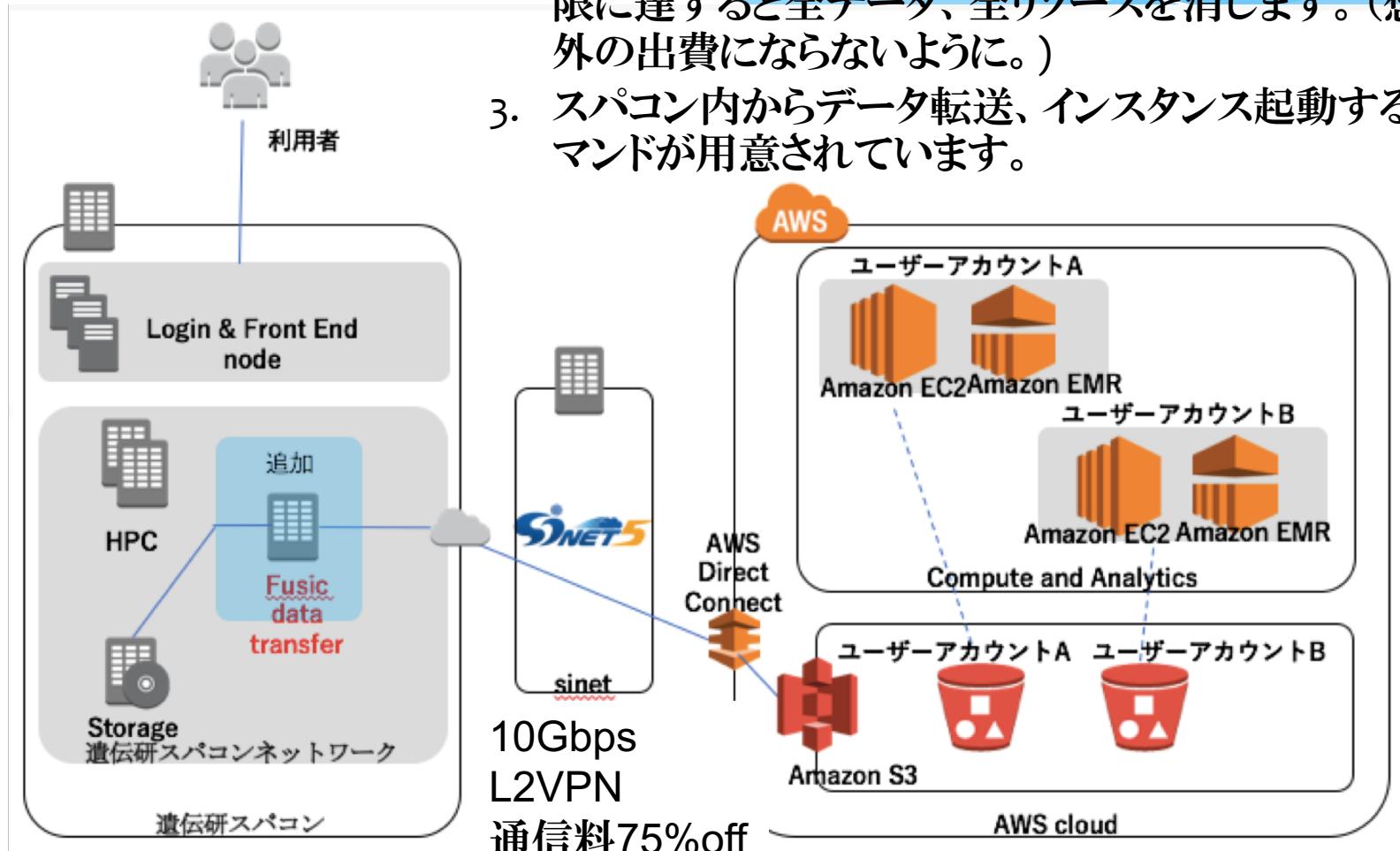
# Singularityでサーバーを立てる

ElasticSearch, BLAT, BLAST用には、すでにあるWebアプリを使い、JBrowseとのリンクのためにApache mod\_actionsでWeb画面にリンクを動的に埋め込んで表示する。



# ハイブリッドクラウド

1. 請求書払いで払えます。各自でAWSに契約・支払いしてください。
2. 利用上限に近づくとアラートメールが来ます。利用上限に達すると全データ、全リソースを消します。(想定外の出費にならないように。)
3. スパコン内からデータ転送、インスタンス起動するコマンドが用意されています。



# 大容量データのやりとり

遺伝研のネットワーク帯域幅は30Gbpsになる。

10Gbps = 100TB/day

- \* scp, sftp
- \* Aspera (上限10Gbps)
- \* Grid FTP (特に海外のスパコンとのデータのやりとりなど)
- \* ディスクの搬送(予定)
  - \* ネットワークの条件が悪いときなど。80TB程度のRAIDを郵送します。

# GCC, Intel Compiler, PGI Compiler

- \* C, C++, Fortran用コンパイラとして、GCCの他にIntel Parallel Studio XEおよびPGI Professional Editionが導入されています。
- \* 付属のインテルMKLは、工学、科学、金融系アプリケーションのパフォーマンスを最大限に引き出すために、並列化、最適化された演算ルーチンのライブラリです。
- \* また、GPU環境への対応としてNVIDIA社製 CUDA Toolkit、Nvidia Docker2、NVIDIA Data Center GPU Manager(DCGM)を導入しています。

## コンパイラと付属ユーティリティ詳細

PGI Fortran 77/90/95/2003/2008 コンパイラ

PGI FORTRAN77 専用コンパイラ

PGI C11 コンパイラ

PGI ANSI C++14 コンパイラ(Linux, OS X)

OpenACC コンパイラ機能 (Fortran/C/C++)

PGI CUDA Fortran

Open MPI 2.1.2(Linux版)、MPICH 3.2 ライブラリ(OS X版)

分散メモリ用 Pre-Build Scalapack 2.0.2 (Linux, OS X)

マイクロソフト MS-MPI ライブラリ (Windows版)

OpenMP/MPI GUI 対応 PGDBG シンボリックデバッガ (※)

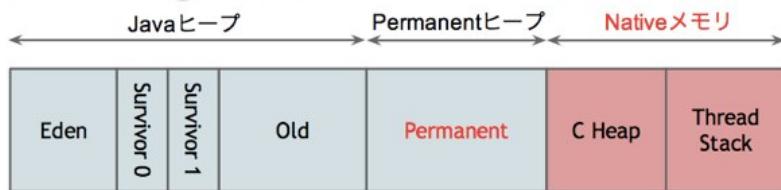
OpenMP/OpenACC GUI 対応 PGPROF 性能解析プロファイル

Microsoft MPI (MS-MPI) 対応の並列デバッガ搭載

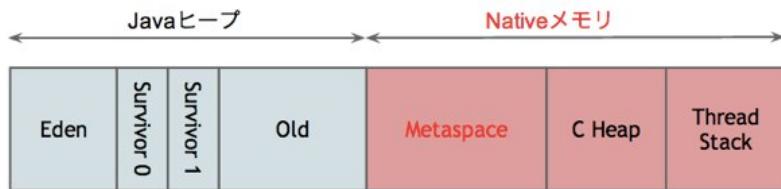
OpenBLAS project sourceにに基づいた BLAS/LAPACK

# Javaに関するFAQ

## ~Java7 HotSpotVM



## Java8 HotSpotVM



<http://equj65.net/tech/java8hotspot/>

- \* Javaプログラムから使われるメモリは、  
(1)JVMが管理している「Javaヒープメモリ」  
(2)OSが管理している「ネイティブメモリ」の  
2種類がある。
- \* glibc 2.10以降で arena機能が導入された  
ことにより、Javaプログラムにとってネイ  
ティブメモリ側のメモリ確保量が不必要に  
大きくなる結果となっております。Javaプ  
ログラムを使う際には環境変  
数 `MALLOC_ARENA_MAX`に小さな値(2  
とか3とか)を設定してください、というのが  
結論になります。
- \* 64bitOSでは `MALLOC_ARENA_MAX` のデ  
フォルト値が128(64MBのブロックを最大  
で128個取るので、128スレッドがmallocし  
たときで8GBの仮想メモリを使用)。

# Python, R, Ruby, Perl, ...

- \* 3<sup>rd</sup> partyのライブラリのインストールが出来ないという質問が頻繁に来ますが、これらの言語では\$HOMEの下にインストール可能ですので、virtualenvのような環境管理ツールをお好みに応じて駆使してください。

# ユーザーアカウント発行基準

国立遺伝学研究所(遺伝研)は大学共同利用機関の一員として国内の大学・国公立研究機関所属の研究者に対する遺伝学を中心としたライフサイエンス系の研究・教育のためのリソース提供を目的としてスーパーコンピューターシステム(スパコン)の運営を行っています。

一方、スーパーコンピューターシステムは外国為替及び外国貿易法(外為法)の輸出管理規制の対象となっており、兵器製造目的などで使用すると処罰の対象となります。

このような背景からスパコンでは具体的には以下の方々を対象にログインユーザーアカウントの発行を行っております。

- \* 外為法の定める国内居住者でありかつ大学あるいは国公立研究機関の教員
- \* 1.の共同研究者、あるいは1.の指導の下にある研究者、学生、委託企業の研究者など  
(留学生、海外への転任、海外の研究者を含む)

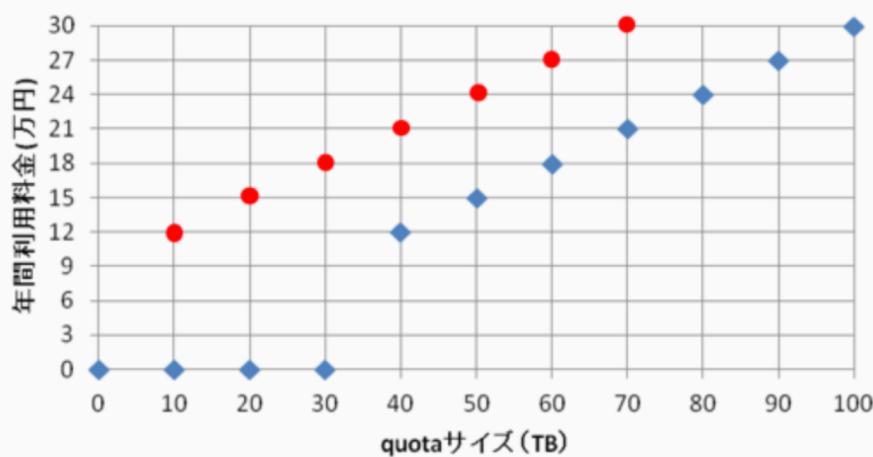
申請時に責任者を指定する欄がありますが上記2.に該当される方は1.の方を責任者に指定してください。(1.の方は申請者と責任者が同一で構いません。)責任者単位でUNIXグループが作られます。

# 課金について

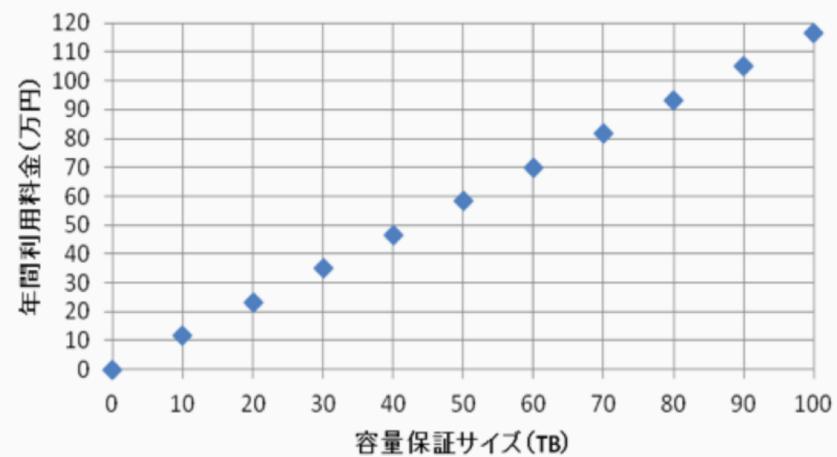
- \* 計算ノード
  - \* UGEを利用した優先利用(Advanced Reservation)
  - \* ノード貸し
- \* ストレージ
  - \* 高速ストレージに対するQuota設定
  - \* アーカイブストレージの領域占有

# 課金: ストレージ

高速ストレージ(quota設定)



アーカイブストレージ(容量保証)



○赤点が個人ゲノム解析環境の価格

◆青点が遺伝研スパコン一般ユーザーの価格

・責任者単位でのquota設定量の合計で30TBまで無料で利用可能です。

・10TBあたりの年間利用料金は11.7万円です。

・10TB単位で申請してください。

# 問い合わせ先

<https://sc.ddbj.nig.ac.jp/ja/application>をご一読の上以下の要領でお問い合わせください。

## その他お問い合わせ

国立遺伝学研究所スーパーコンピュータシステム利用に関する質問をメールで受け付けています。

なお、よくあるご質問とその回答について、"[よくある質問\(FAQ\)](#)"にまとめてありますのでまずはそちらをご覧ください。

- メールの送信先：[sc-helpdesk@nig.ac.jp](mailto:sc-helpdesk@nig.ac.jp)
- メールでお問い合わせいただく際には、以下の項目を明記してください。

- お名前（必須）
- 所属（必須）
- ログインアカウント
- お問い合わせ件名（必須）
- お問い合わせ内容（問題が再現する手順など、なるべく詳細に）