はじめてのHPCによる深層学習
～スーパーコンピューターに触れてみる～

ohtaman

第61回 Machine Learning 15minutes! V-expo (2021/11/27)

自己紹介

太田 満久 (ohtaman)

- CDTO (Chief Data Technology Officer) at BrainPad Inc.
- Google Developers Expert (Machine Learning).
- An organizer of TensorFlow User Group Tokyo.

研究会・イベント
- ASE研究会
- セミナー
- シンポジウム・ワークショップ等
- イベント報告
- JCAHPC
- 講習会
  - 講習会開催予定
  - 講習会教育教材
- 「若手・女性利用者推薦」成果報告会

第159回お試しアカウント付き並列プログラミング講習会「第4回 GPUミニキャンプ～DL編～」

共催：東京大学情報基盤センター、エヌビディア合同会社、PCクラスタコンソーシアム（実用アプリケーション部会）
後援：株式会社ディー・エヌ・エー、株式会社ブレインパッド

講習会開催に関するお知らせ
新型コロナウイルス感染症対策のため、東京大学本部指針に基づき、今後講習会実施を中止させていただく可能性がございます。何卒ご了承いただきますようお願い申し上げます。（2020.4.7）

開催趣旨

本ミニキャンプでは、これからGPUを利用される方またはすでに利用されているが効率化を進めたい方、スパコンのGPUを利用したい方、を対象に、情報基盤センター（以降、センター）に設置されたスーパーコンピュータ Wisteria/BDEC-01 を活用した実践を行います。受講料は無料です。
本イベントは、新型コロナウィルス感染症の拡大を受け、オンラインでの開催となります。

https://www.cc.u-tokyo.ac.jp/events/lectures/159/

ABCIミニキャンプ2021/05

ABCI ミニキャンプ募集要項

ABCI ミニキャンプでは、CUDA、OpenACC、ディープラーニングなど GPU に関連する様々な課題を持ち込み、メンターからの助言を受けながら ABCI を活用して課題を解決することを目指します。Slack とオンライン会議システムを使って進めていきます。また ABCIとGPU のスペシャリストがメンターとして参加します。
今回は主にディープラーニングの課題を対象とし、新たに運用開始予定の計算ノード (A) が提供されます。

■日時
- 5 月 12 日 (水) 10:00 - 17:00
- 5 月 14 日 (金) 10:00 - 17:00

■開催方法
- オンラインで開催

■対象者・参加資格
- ABCI 利用者 (ABCI アカウント保有者)
- ABCI、オンライン会議システム (Zoom の予定) および Slack への接続が可能な PC とネットワーク環境から参加できる方
- 取り組みたい課題をご自身で用意できる方
- 利用者のプログラム等をメンターが見ることを了承いただける方
- 基本的に 2 日間ご参加いただきますが、途中の一時的な離席は自由

■提供計算資源の詳細
1 チームあたり (1 人参加も可能)、1 ノード (8GPU)、最大 10TB のディスクスペース (クラウドストレージ) を期間中占有利用できます。

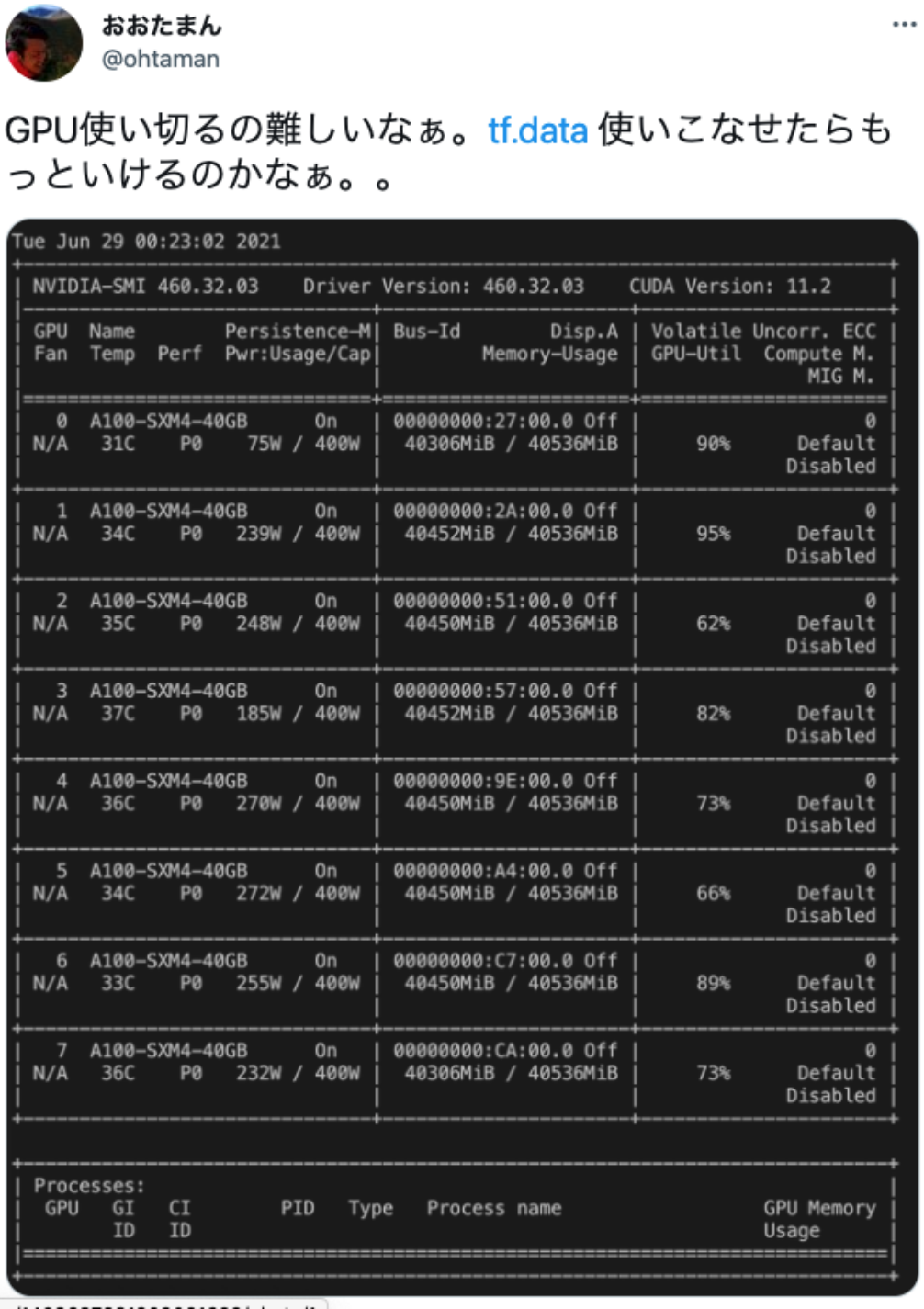
https://abciug.abci.ai/minicamp202105-p53

GPUミニキャンプを通じて知ったこと

1. スパコンは特殊な装置ではない。使い方さえ知っていれば、案外簡単に使える。
2. スパコンは学術研究以外にも使える（利用規約や運用方針による）。
3. スパコンは思っていたよりも安い。

スパコンを使うメリット

1. 大量のマシンによる並列分散処理で、モデルの訓練時間を短縮できる
2. クラウドサービスよりも安価に利用できる



参考: ABCI の料金体系

- ポイント制（1ポイント220円）
- 3.0ポイント/時間
  (A100x8, 72コア, 480GBメモリ, 2021年度の料金設定)

ABCIサービス料金表 (2021年度)

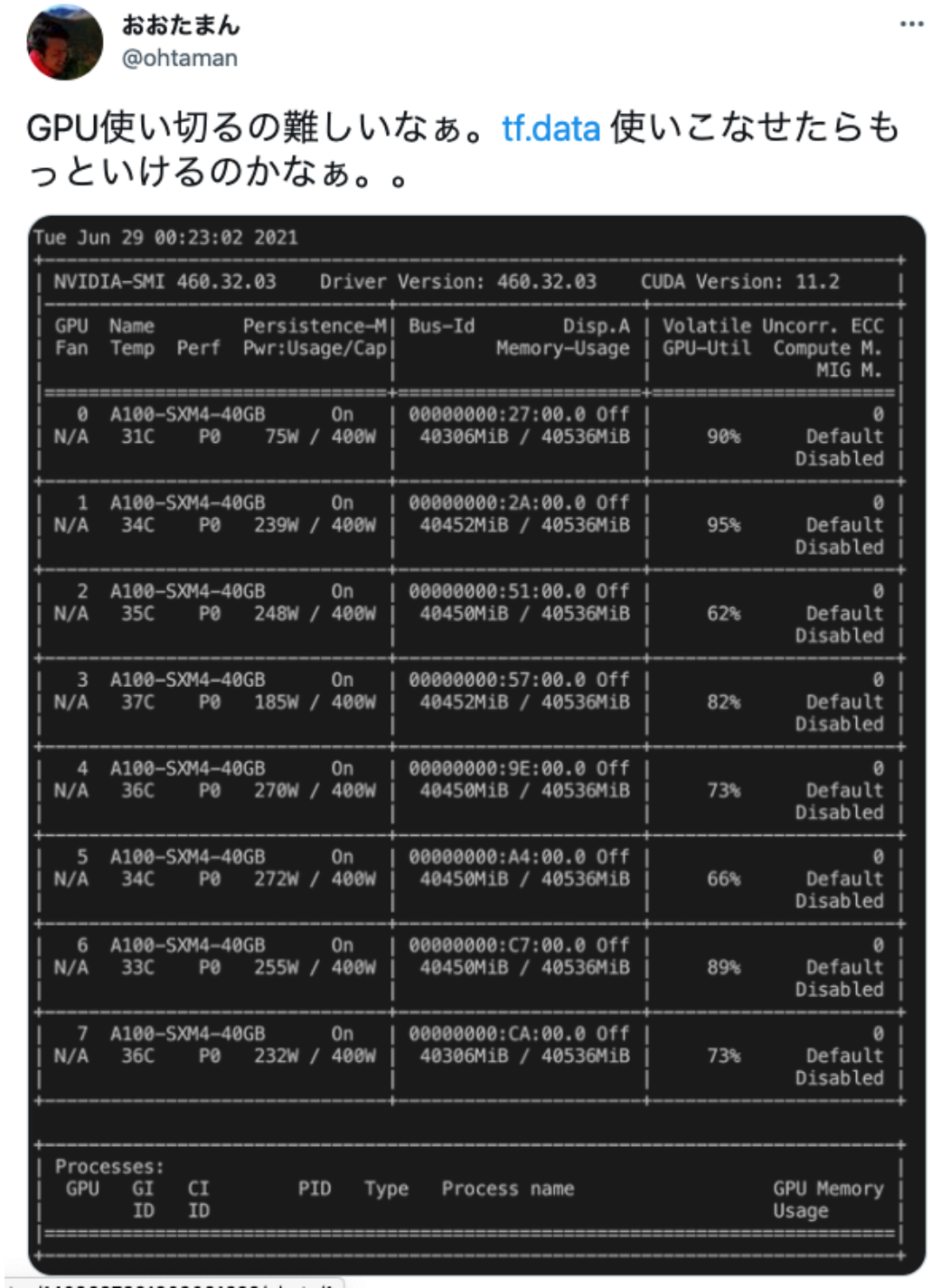
2021年度は、1 ABCIポイント当たり220円(税込)です。ただし、初回ポイント取得時のみ、最初の1,000ポイントを19万8千円(税込)で取得できます。

サービス	計算資源タイプ	計算資源概要	バッチ及びインタラクティブジョブ	予約
計算資源	<計算ノード(V)>			
	rt_F	4GPU, 40コア, 360GBメモリ	1.0 ポイント/時間	36 ポイント/日 (1.5 ポイント/時間相当)
	rt_G.large	4GPU, 20コア, 240GBメモリ	0.9 ポイント/時間	NA
	rt_G.small	1GPU, 5コア, 60GB メモリ	0.3 ポイント/時間	
	rt_C.large	20コア, 120GBメモリ	0.6 ポイント/時間	
	rt_C.small	5コア, 30GBメモリ	0.2 ポイント/時間	
	<メモリインテンシブノード>			
	rt_M.large	8コア, 800GBメモリ	0.4 ポイント/時間	NA
	rt_M.small	4コア, 400GBメモリ	0.2 ポイント/時間	
	<計算ノード(A)>			
	rt_AF New	8GPU, 72コア, 480GBメモリ	3.0 ポイント/時間	108 ポイント/日 (4.5 ポイント/時間相当)
	rt_AG.small New	1GPU, 9コア, 60GB メモリ	0.5 ポイント/時間	NA
ストレージ	グループ領域 (共有ディスク)	5 ポイント/TB・月 (ABCIアカウントに割当てられる200GBのホーム領域は無償)		
	ABCIクラウドストレージ	0.0001 ポイント/GB・日 (従量制)		

スパコンを使うメリット

1. 大量のマシンによる並列分散処理で、モデルの訓練時間を短縮できる
2. クラウドサービスよりも安価に利用できる

でも、利用イメージがわかないので始められない...



というわけで、主に深層学習のモデル構築を想定したスパコン（HPC）の利用イメージを紹介します

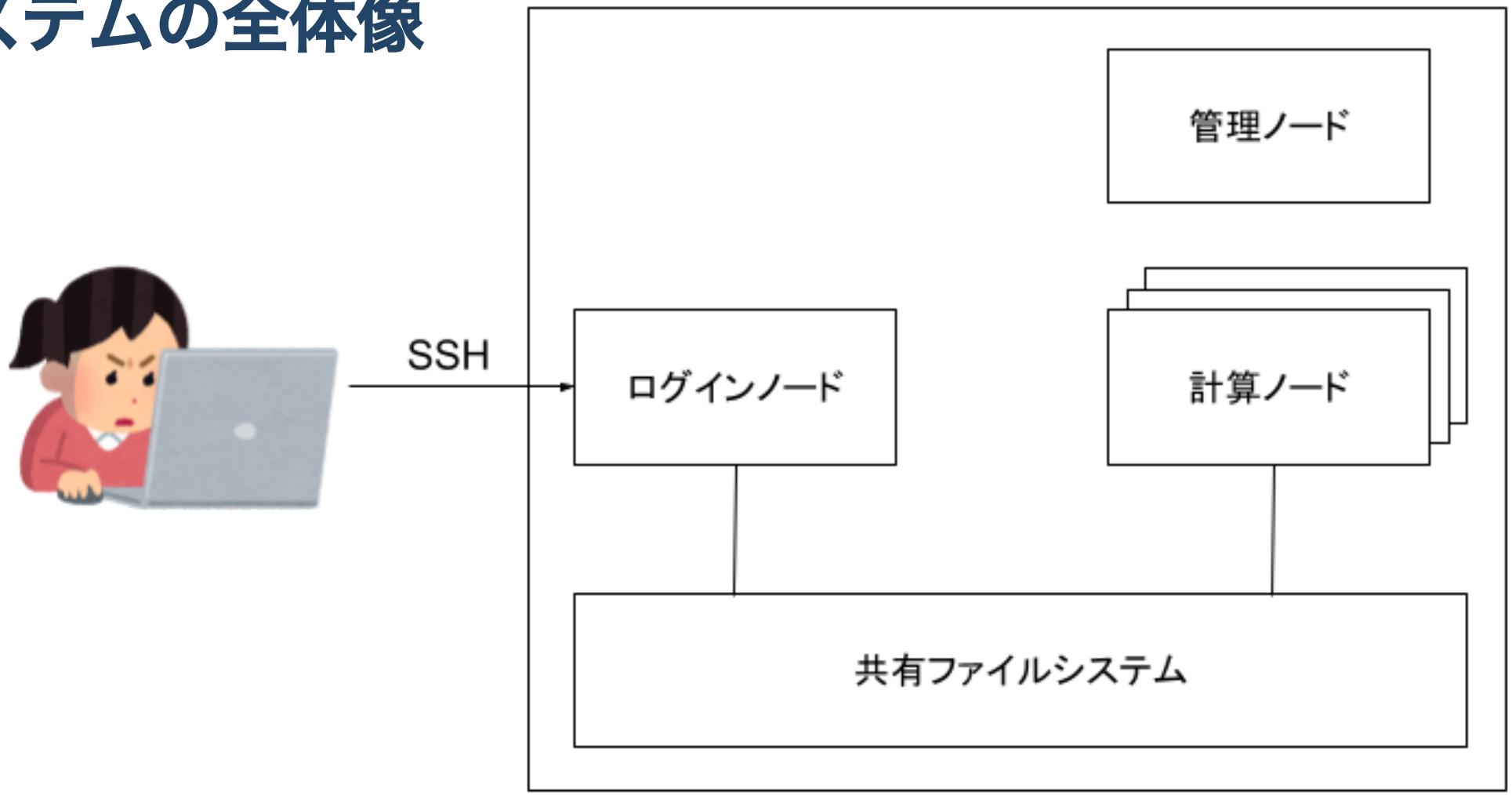
覚えておくと良さそうなこと

1. ジョブスケジューラー
2. Environment Modules

覚えておくと良さそうなこと

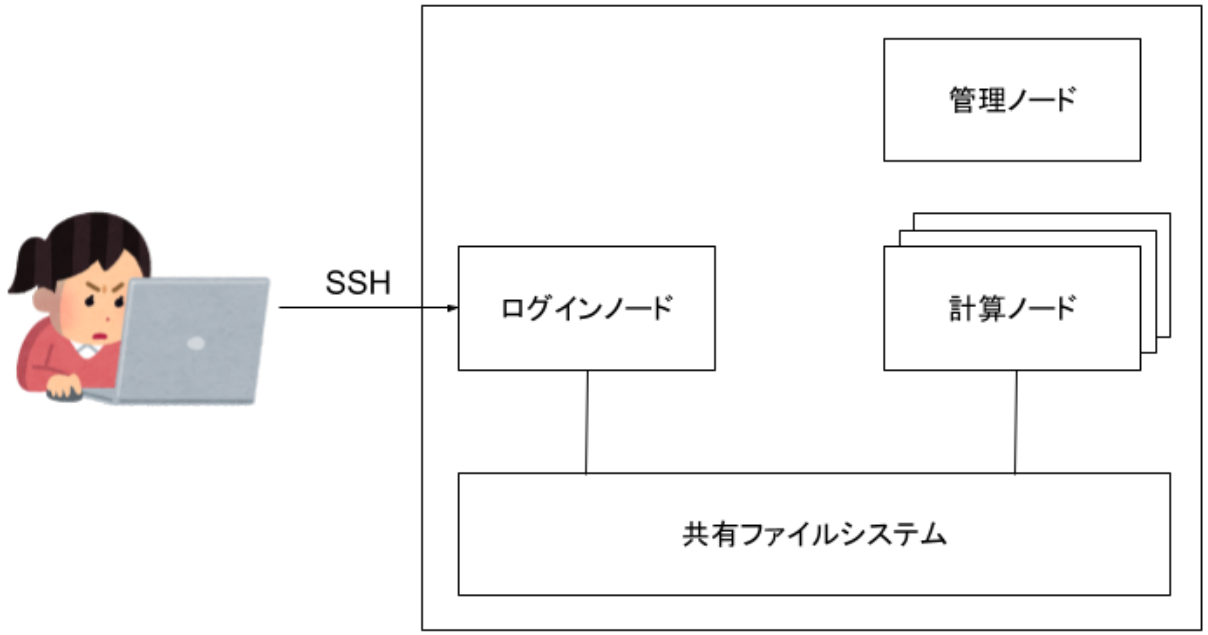
1. **ジョブスケジューラ**: 複数人からの依頼に応じてリソースの割り当てを行う
2. Environment Modules

システムの全体像



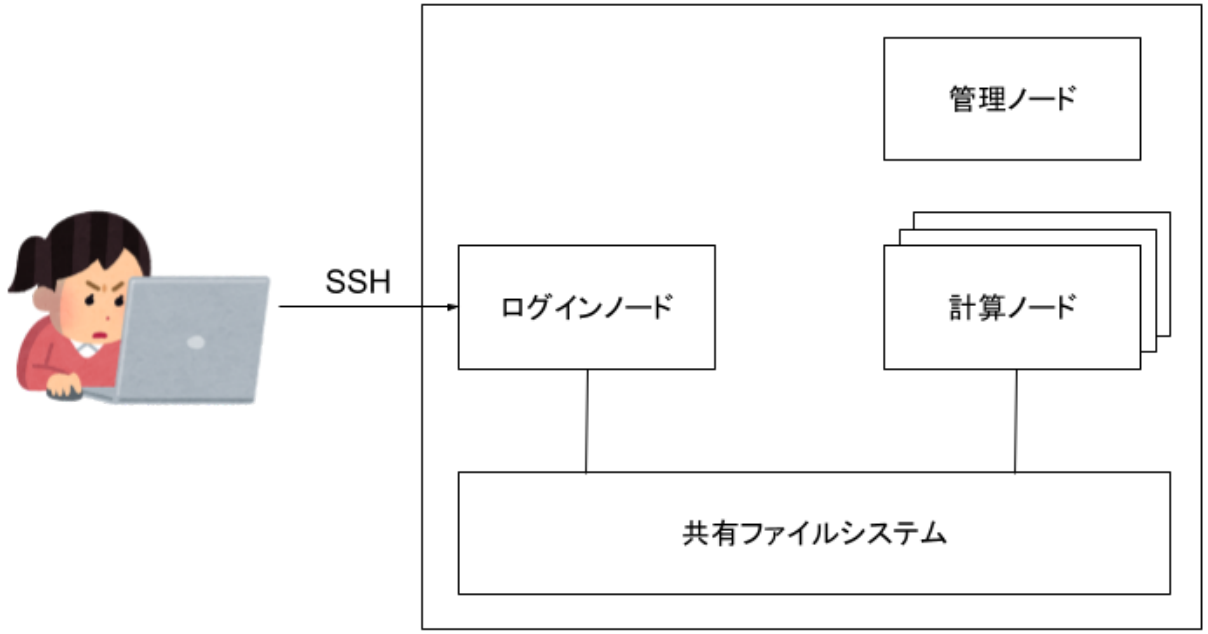
システムの全体像

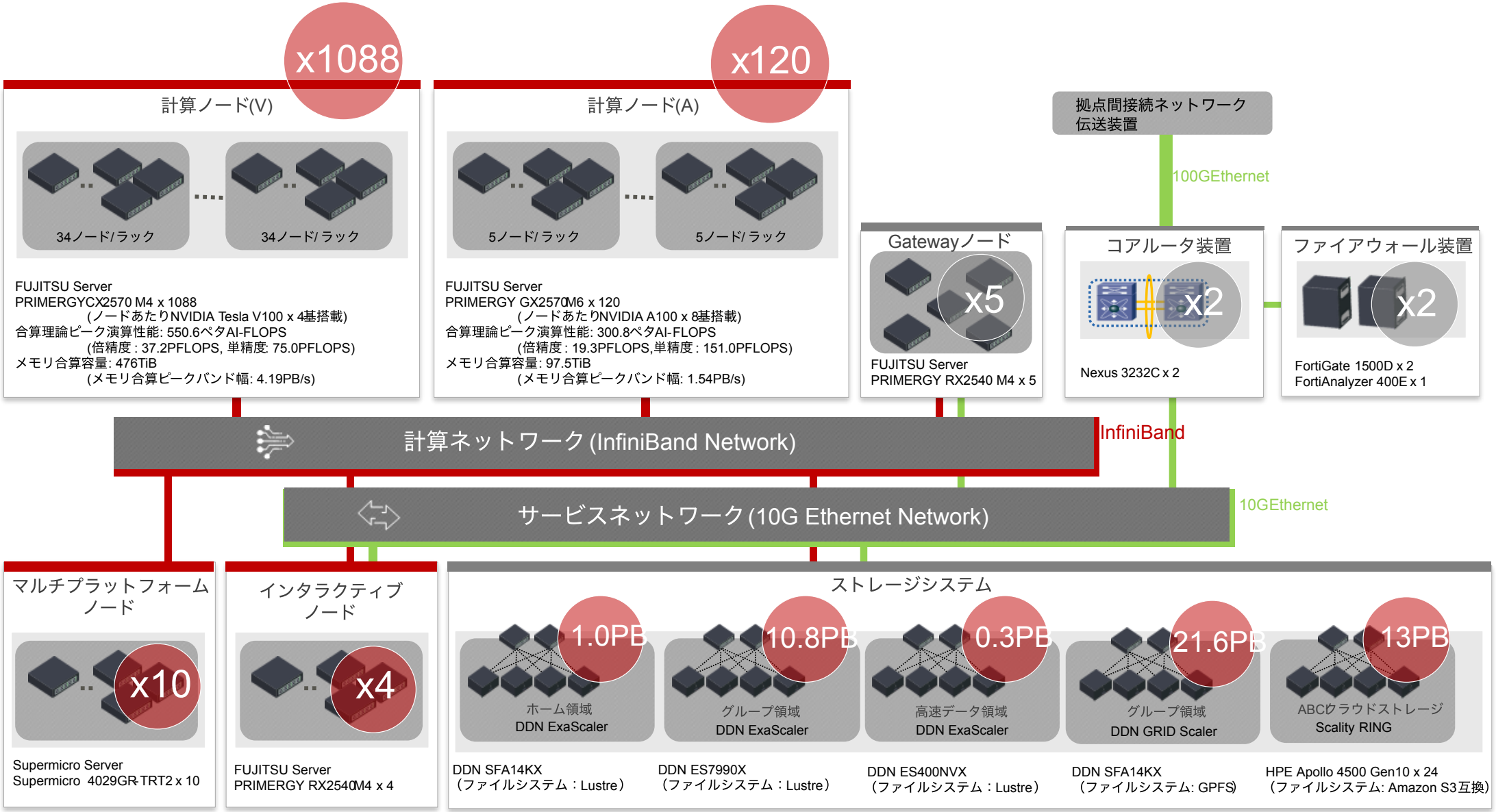
1. ログインノード
   ユーザーがログインして各種設定やジョブの投入を行う
2. 計算ノード
   ジョブを受け取り、計算を行う
3. 管理ノード
   リソースやキューの管理を行う



ジョブ実行の流れ

1. ログインノードにログイン
2. 計算に必要なファイルの準備
3. (optional) インタラクティブジョブで動作確認
4. バッチジョブを実行
5. 結果の確認





ABCIシステムの概要: https://docs.abci.ai/ja/system-overview/

覚えておくと良さそうなこと

1. ジョブスケジューラ: 複数人からの依頼に応じてリソースの割り当てを行う
2. **Environment Modules**: 利用するライブラリの管理を行う

Environment Modules の利用例

`module` コマンドを用いて、使いたいライブラリの使いたいバージョンをロード

```
$ # 利用可能なモジュールの確認
$ module avail
--------------------------- /usr/share/Modules/modulefiles ---------------------------
dot  module-git  module-info  modules  null  use.own

------------------------------- /usr/share/modulefiles -------------------------------
mpi/mpich-x86_64
$ # モジュールのロード
$ module load mpi
$ # ロード済みモジュールの確認
$ module list
Currently Loaded Modulefiles:
 1) mpi/mpich-x86_64
```

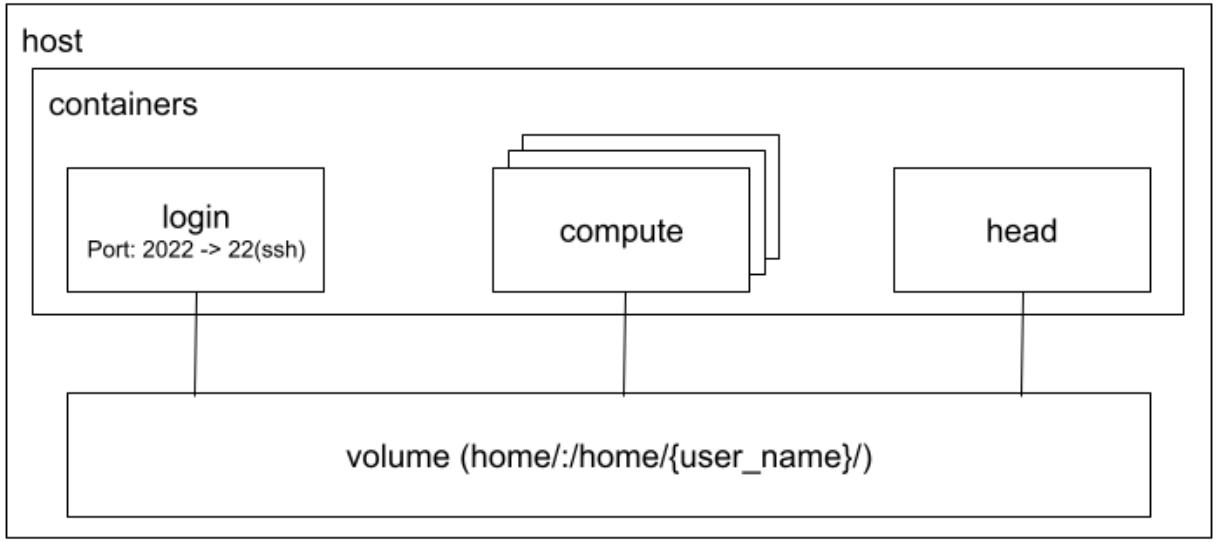
でも、触ってみないと実感が湧かないよ...

ということで、疑似環境を用意しました

https://github.com/ohtaman/docker-compose-hpc

docker-compose-hpc

- docker-compose による構成
- ホストのディレクトリを共有
- OpenPBS の利用
- Environment Modules の利用



クラスタの構築

```
host$ docker-compose up -d
```

ログイン

ログインノード（login）が SSH を待ち受けている

```
host$ docker-compose ps
             Name                           Command             State           Ports
-------------------------------------------------------------------------------------------
docker-compose-hpc_compute_1   /bin/sh -c /entrypoint.sh   Up
docker-compose-hpc_compute_2   /bin/sh -c /entrypoint.sh   Up
docker-compose-hpc_head_1      /bin/sh -c /entrypoint.sh   Up
docker-compose-hpc_login_1     /bin/sh -c /entrypoint.sh   Up      0.0.0.0:2022->22/tcp
```

ホストのユーザー名ログインノード (login) でログイン

```
host$ ssh -i ssh/id_rsa -p 2022 localhost
```

ジョブの実行

スクリプトの中身を確認

```
$ cat mnist.sh
#!/bin/sh
#PBS -q workq
#PBS -l nodes=1:ppn=1
#PBS -l mem=2G

# qsubを実行したディレクトリに移動
cd "${PBS_O_WORKDIR:-$(pwd)}"

source .venv/bin/activate
python mnist.py
```

ジョブの実行

`qsub` コマンドでジョブを送信

```
$ qsub job.sh
```

`qstat` コマンドでジョブの状態を確認。実行中のジョブがない場合は何も表示されない。

```
$ qstat
Job id              Name             User              Time Use S Queue
----------------  ---------------- ----------------  -------- - -----
0.head              job.sh           mitsuhisa.ota     00:00:00 R workq
```

計算ノードの追加

docker-compose なので、計算ノードを（仮想的に）好きなだけ追加できる

```
host$ docker-compose up -d --scale compute=2
```

MPI による分散処理

MPI を用いると、複数の計算ノードで並列計算できる (以下の例では `nodes=2:ppn=2` で2ノードで2プロセスずつ実行するよう指定)

```
$ cat mpi.sh
#!/bin/sh
#PBS -q workq
#PBS -l nodes=2:ppn=2
#PBS -l mem=2G

module load mpi
mpirun hostname
```

MPI による分散処理

ログを見ると、2つの計算ノード（572085e2d423, f565b6b4a09b）それぞれで2プロセスずつ実行されていることがわかる

```
$ qsub mpi.sh
... (ジョブが終了するまで待機する)
$ cat mpi.sh.o1
572085e2d423
572085e2d423
f565b6b4a09b
f565b6b4a09b
```

Optuna による並列分散最適化

機械学習で複数ノードを利用する場合、1つのモデルを複数ノードで訓練することもできるが、パラメーター探索などで大量のモデルを構築したい場合は、各ノードにモデルを割り当て、それぞれ独立に訓練すると、実装がシンプルになる。

Optuna による並列分散最適化

Optuna を使って MPI を用いたパラメーター探索を行う場合は、以下のように適切な storage を指定するだけ

```
def main():
    # Optuna で分散並列最適化を行うには、storage を指定する
    # https://www.slideshare.net/pfi/20201023-optunalectureatnagoyauni-238946529
    study = optuna.create_study(
        direction='maximize',
        storage='sqlite:///db.sqlite3',
    )
    study.optimize(objective, n_trials=5)
...
```

※ SQLite ではなく PostgreSQL の方が安心

スパコン利用の注意点

1. 個人利用はできないことが多い（利用規約や運営方針による）
2. 初期費用（最低のチャージ）が発生することがある
3. 一般のクラウドサービスと異なり、サービング環境はないため訓練のみ利用可能
4. グランドチャレンジなど優先的なプロジェクトがあるため、常時実行するようなジョブには向いていない

まとめ

1. スパコンは民間利用もできて、安い。使い方も難しくない。
2. 体験していただけるように docker-compose で疑似環境を作ってみたので触ってみてね

参考資料

- docker-compose-hpc
  - スライドで紹介した疑似環境
- 一週間でなれる！スパコンプログラマ
  - MPIやスパコンの基本からわかりやすく解説している記事
- ABCI 2.0 User Guide
  - ABCI (AI橋渡しクラウド) のユーザーガイド
- 不老におけるOptunaを利用した分散ハイパーパラメータ最適化 - 今村秀明（名古屋大学 Optuna講習会）
  - Optuna の紹介スライド