TSUBAMEシリーズにおける ブラウザベース利用機能の導入

野村 哲弘

東京科学大学 情報基盤センター

Institute of SCIENCE TOKYO

この発表の概要

- 東工大/科学大のスパコンTSUBAMEシリーズにおいて提供してきたブラウザ経由利用について、そのモチベーションと独自実装、Open OnDemandへの乗り換えまでの流れを報告する
  - SWoPP2020でTSUBAME3導入直後に発表しており、その内容を多く含みます

- 本発表における禁句: それ、Open OnDemandでできます（当時は存在を知らなかった）

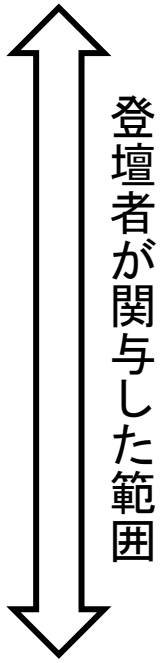
自己紹介

- 野村 哲弘 / Akihiro Nomura / Science Tokyo(東京科学大学) 情報基盤センター
- TSUBAMEの中の人の一人
  - 2011年より東工大(現: 東京科学大)
  - TSUBAME2以降の運用・3以降の仕様策定に従事
  - スパコンに新しいモノを入れて使えるようにするのが趣味
    - TSUBAME2: Caffe, ~~TensorFlow~~ (DLL Hellに敗北), Score-P
    - TSUBAME3: Jupyter Notebook / Lab 起動スクリプト→2020/4にWebサービス化
    - その他: Vim, CMake, OpenACC対応GCC, CUDAなどの新しいバージョン
  - TSUBAMEシリーズの各種運用の黒幕(大抵の場合)
    - このスパコン使いにくいという声は極力拾っていきたい所存
      (とはいえ、何かとのトレードオフで仕方なく使いづらくしている面もあり)
  - Xの @TSUBAME_sc やWebサイト・マニュアル類も担当

東工大/科学大 TSUBAMEシリーズの歴史

Institute of SCIENCE TOKYO

- TSUBAME1.0～1.2 (2006-2010)
  - 655ノード
  - 1.2はGPUを用いた世界初の大規模スパコン
- TSUBAME2.0/2.5 (2010-2017)
  - 1408ノード(+α)
  - 1ノード3GPU, 2 IB HCA (QDR, 40Gbps)
- TSUBAME3.0 (2017-2024)
  - 540ノード
  - 1ノード4GPU, 4 Omnipath HFI (100Gbps)
- TSUBAME4.0 (2024-2030)
  - 240ノード
  - 1ノード4GPU, 4 IB HCA (NDR200, 200Gbps)



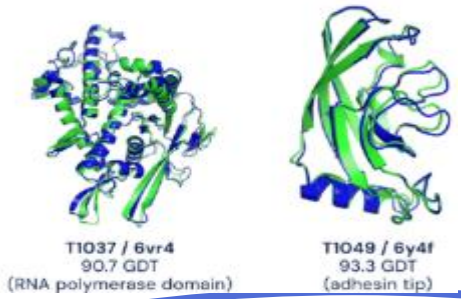
以下、TSUBAMEN.x 系をまとめて TSUBAMEN もしくは TN と表記します (例: TSUBAME2.0/2.5 → T2)

TSUBAME4.0

データ・計算科学・AI融合のためのもっとみんなのスパコン

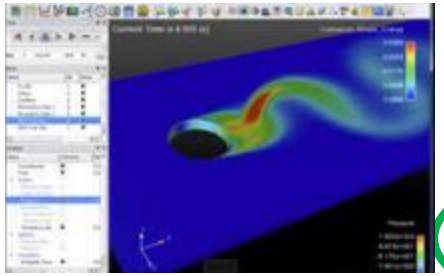
Institute of SCIENCE TOKYO

対話的データ解析

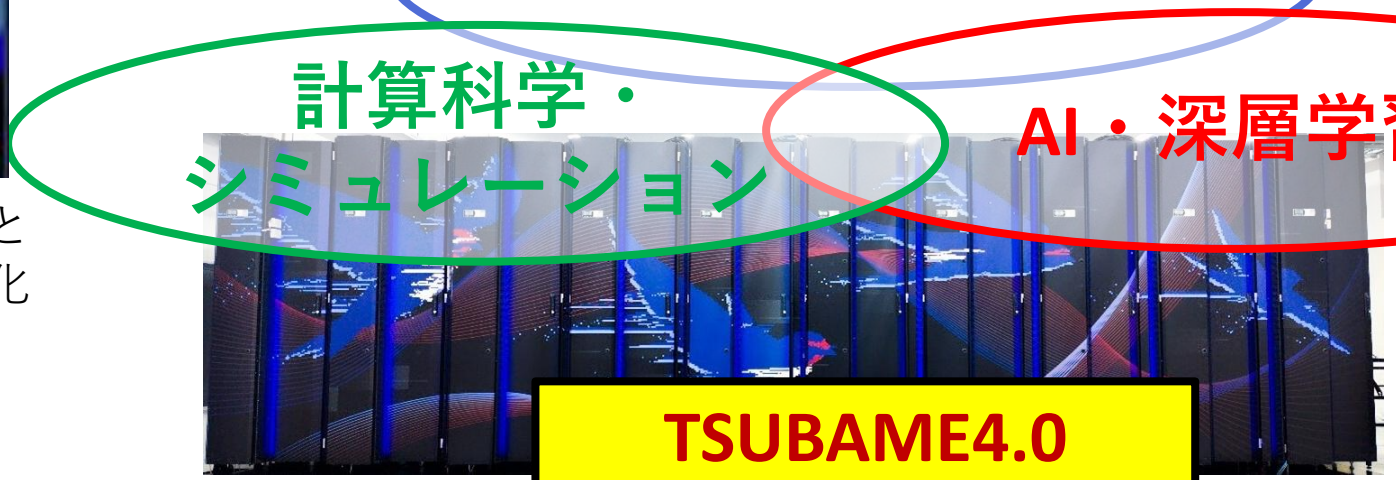


深層学習との融合による
シミュレーション革新

SNSのフォロー関係解析



シミュレーションと
リアルタイム可視化



深層学習による
画像認識・生成AI

TSUBAME4.0の概要

計算ノード（**単一構成**）：
HPE Cray XD665 ×**240台**
H100 GPU 計960台 **← 全ノードにGPU**
総演算性能：
- **66.8 PFlops**（倍精度）
- **952 PFlops**（半精度）

共有ストレージ：
HPE Cray ClusterStor E1000
総容量：
- 44 PByte（ハードディスクベース）
- 327 TByte（SSDベース）

30ラックに格納
- 計算ノード 23ラック
- ストレージ・管理 7ラック

東工大(現: 東京科学大)
すずかけ台キャンパスG4-A棟に設置

構築業者：Hewlett Packard Enterprise

2024/11 ランキング
Top500: No. 36 (39.42PFlops)
Green500: No. 30 (48.56GFlops/W)

TSUBAME4.0でのリソース分割

- いろいろなTSUBAMEのユーザ
  - **CPUもGPUもあるだけ使う**
  - **GPUはあるだけ、CPUは最低限**
  - **CPUしか使わない**
- TSUBAME4の計算ノードは「パワフル」すぎる
  - GPUが4台、各GPUも強力
  - CPUコア数 計192コア
- けど、台数が少ない
  - TSUBAME2の頃は1400台
  - TSUBAME4では240台

→ **動的に分割してみんなで使う**

  - AltairとHPEが頑張りました

資源タイプ	CPUコア数	メモリ(GB)	GPU数	ローカルSSD容量(GB)
node_f	192	768	4	1920
node_h	96	384	2	960
node_q	48	192	1	480
node_o	24	96	1/2	240
gpu_1	8	96	1	240
gpu_h	4	48	1/2	120
cpu_160	160	368	0	960
cpu_80	80	184	0	480
cpu_40	40	92	0	240
cpu_16	16	36.8	0	96
cpu_8	8	18.4	0	48
cpu_4	4	9.2	0	24

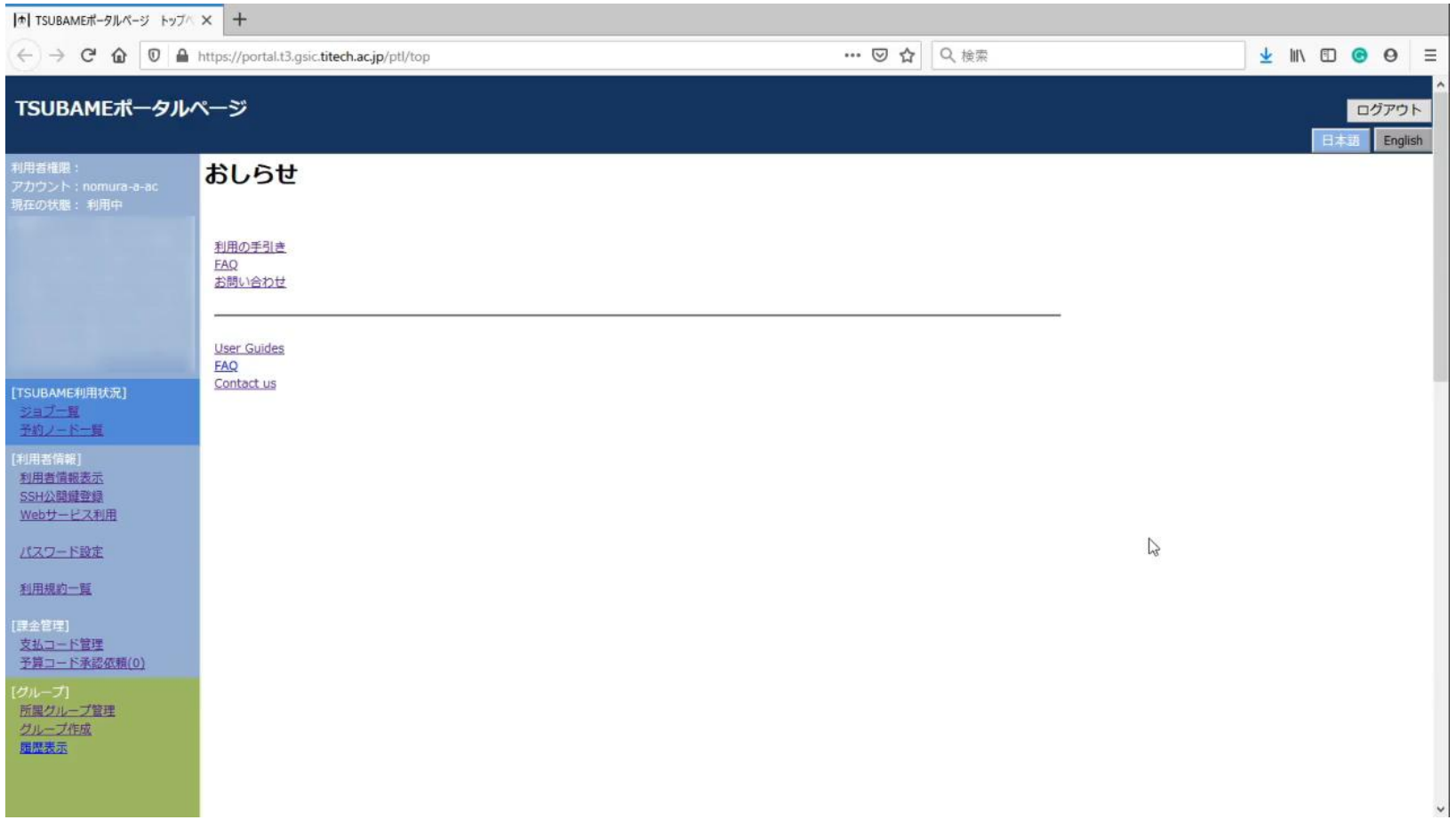
ユーザは自分にあったサイズの資源タイプを使う

(**費用負担も使ったサイズに比例**)

スパコン初学者にとっての壁

- スパコンTSUBAMEシリーズのユーザ層
  - 科学大 学内ユーザ(教員・学生) - 「みんなのスパコン」
  - 共同研究等に基づく他機関・企業の研究者
  - TSUBAME共同利用・HPCI/JHPCN等の学外利用プログラムによる学術利用・産業利用ユーザ
- スパコンの計算ノードで何かをするために必要な知識
  - SSHおよび公開鍵認証 ← 教育コスト超高 Zoomでトラブルシューティングできますか?
    - X転送・ポートフォワーディング (可視化・Jupyterなどを利用する場合)
  - Linuxの基本的なコマンド
  - ジョブスケジューラの概念・ジョブスクリプト
  - アクセラレータ(GPU)や並列処理(MPI, OpenMP, OpenACC)
  - ドメイン・実際に動かすアプリの知識
  - 本質的には、オレンジ色の箇所の知識しか要らないはず
    - あるには越したことないけど、後からでよい

TSUBAME3のWebアプリ実行機能(2020)



SSHを知らなくてもいつでもTSUBAMEを使える

- 利用者ポータルからジョブ投入に必要な情報を入れるとジョブが作られる
- ジョブで確保された計算ノード内でJupyter Labが起動し、リバースプロキシ経由でユーザのブラウザに表示されえる
- Jupyterで何かやると計算ノードのCPU/GPUで計算が走る

SSHやジョブスクリプトは操作に出てこない

ネタバレ: 当時Open OnDemand[2013]を知っていれば...

さて、これを手で実現するとどうなる?

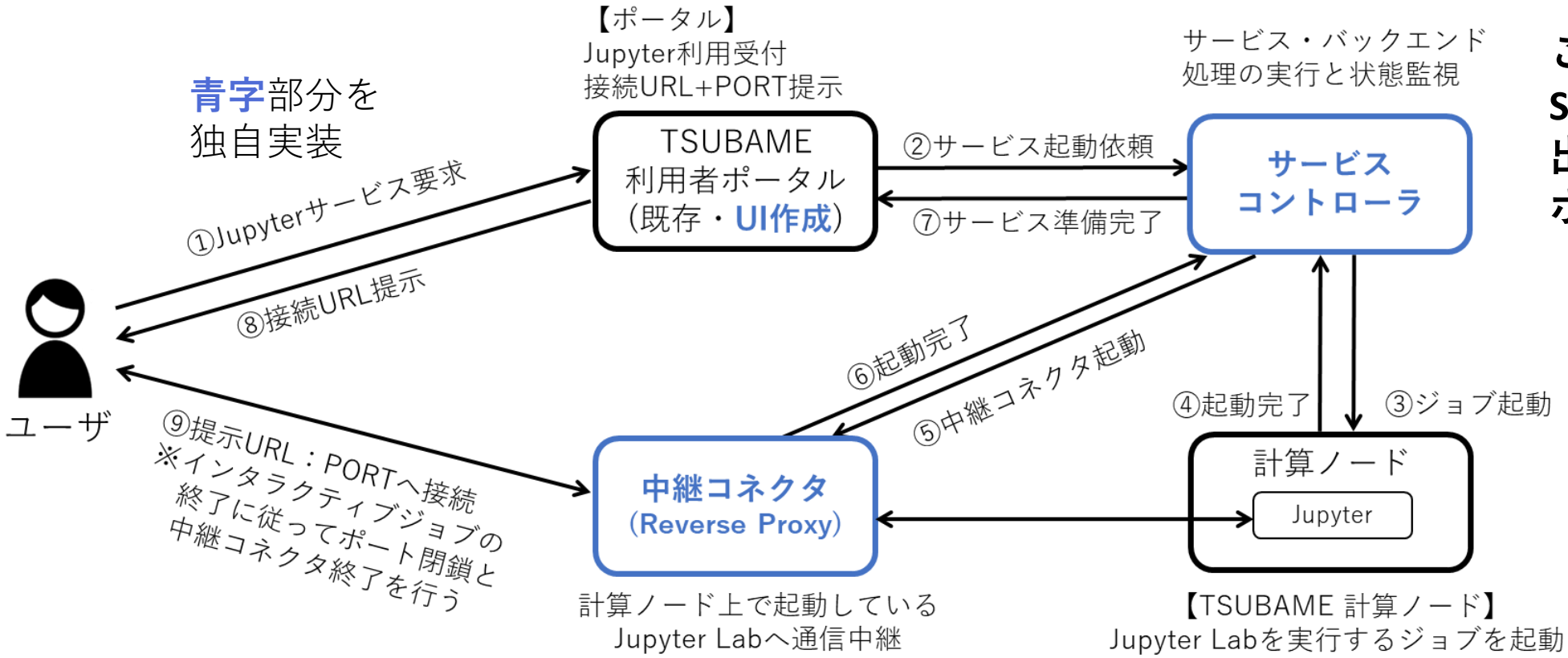
- 一切システムの助けなしにやるには、こんな感じ
  - Jupyterのインストール (`pip install --user jupyterlab`)
  - 計算ノード確保
  - Jypyterの起動(を、ジョブスクリプト内で行う)
  - Jupyterが待ち受けている計算ノードまで、SSHポートフォワードを設定
  - 理屈はわかるが、正直面倒すぎてやって(調べて)られない
- Jupyter起動自動化コマンド `jupyterrun` (2019公開, T4でも利用可能)
  - ユーザは `jupyterrun` を、AGEの`qrsh`と同じオプションで起動するだけ
  - 実行すると以下のようなメッセージが表示されるので、別ターミナルへコピペ

```
To enable port forwarding, execute this command in another terminal:
ssh -l username -L 8888:r0i1n2:12345 login.t3.gsic.titech.ac.jp
Then, you can connect to http://localhost:8888/
```

  - http://localhost:8888/ につなぐと、計算ノードで動くJupyterが待っている
  - 約40行のスクリプトでだいぶ省力化できた
    - **この自動化を突き詰めるとどこまでできるか → 実装してみた**

TSUBAME3 Webアプリ実行機能の設計

- ユーザはTSUBAMEの利用者ポータルからWebアプリケーション(Jupyter, Code Server, NoVNC)を起動、裏でジョブの管理等を行い、HTTPSリバースプロキシ経由でWebアプリの画面をユーザに提示



この図のどこにも SSH・公開鍵が 出てきていないのが ポイント

ジョブ投入UI

- qsubコマンドのオプションをWeb UIで指定する
- 後述のインタラクティブジョブ専用キューも指定可能
- リソース不足(ノードが空いてない)時にジョブをキャンセルする動作がデフォルト
  - AGEのqsubにおける -now y に相当
  - ノードが空くまで待つこともできる

t3jpttools Pythonモジュール

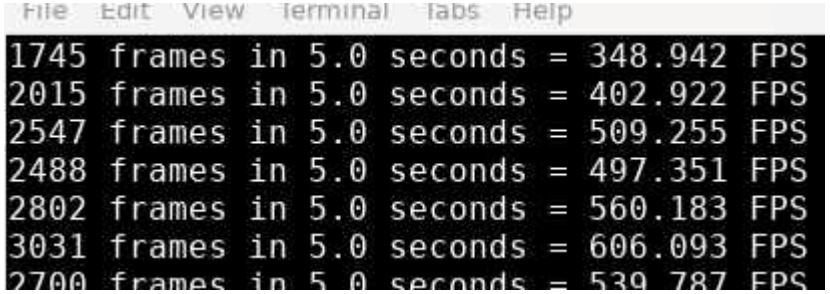
- Jupyter LabのUIだけではできない各種調整用の独自モジュール

- `create_kernel`(カーネル名, Environment moduleのリスト)
  - 実行するとJupyterのカーネル(環境)設定ファイルを作成
  - Jupyterの機能でカーネルを切り替えることで、指定したEnvironment moduleに対応する環境変数が設定されるので、CUDA等を利用可能に

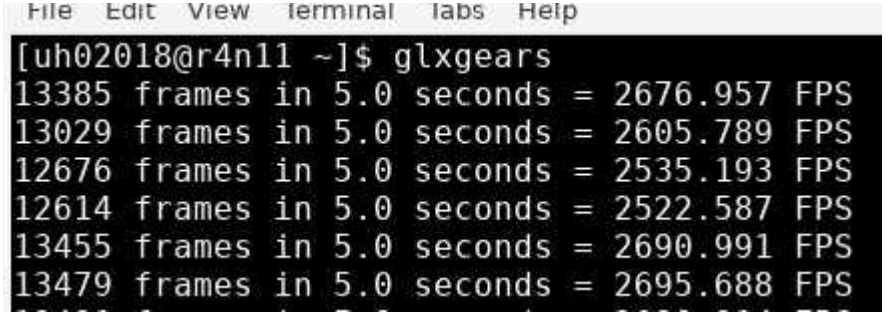
- TSUBAME3ではできていなかったこと
  - virtualenv, pyenv, Anaconda等のPython仮想環境への対応
    - どこまで面倒を見るべきか… 現状はこの辺を使える人は`jupyterrun`も使えるのでそっちで対応
  - 任意のPythonバージョンの選択機構(部分的に対応済み)

T3で対応したWebアプリケーション

- 最初はJupyterのみから始め、1年に1個ずつ対応アプリを増やした
  - Jupyter Lab
    - Pythonだけかとおもったら簡易的なエディタ・ターミナルもあって意外と使えた
  - Code Server (VS CodeのWeb対応クローン)
    - ログインノードでVS Code Remoteのプロセスが異常生成される事態が頻発していた
  - NoVNC (VNCのブラウザ実装)
    - X11使えたらよいのではと探してみたらあった
    - ユーザからのX転送・可視化関係の問い合わせでX11実装との相性問題が疑われるものが多く、標準的なものを用意したかった
    - VirtualGLを使った計算ノードGPUを使っての描画アクセラレーションにも対応 (T4 OODではデフォルトで有効に)
      → X11関係の問い合わせの大半を「Webブラウザ使え」で一蹴できるようになった
      - 余談: MobaXTermのX11もわりと相性問題起こさないし、導入楽なのでおすすめ



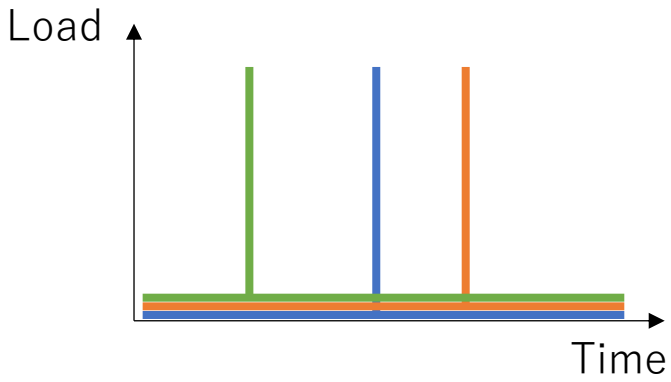
R-CCS Cloud FX700 (GPUないマシン)



TSUBAME4 Interactive Queue w/ VritualGL

インタラクティブジョブ専用キュー

- Webブラウザを用いた対話的利用をやるうえで、計算ノードをいつでも確保できる環境は必須
- TSUBAME3, 4は混み過ぎていて、このような条件を通常ジョブでは満たせない
- 計算ノードの1%を対話利用専用として切り出して、「性能度外視」キューを構築
  - ノード内に多数のユーザを詰め込むことを前提、GPUは共用、CPUも1人当たり2コア程度
  - CPUメモリはSSD上のswapfileで水増し
  - 1人1ジョブまで、複数ジョブ同時実行や複数ノードジョブは走らない
  - 最大実行時間は24h、Web UIのデフォルトは2h
    - 忘れられたころに掃除される
  - その代わり、100人以上が待ち時間なしで即時実行できる
    - 科学大におけるHPC分野の専門科目講義を1クラス分収容できる
  - Jupyter Notebookを読んで簡単なサンプルを実行する・ファイル編集するにはこれぐらいで充分

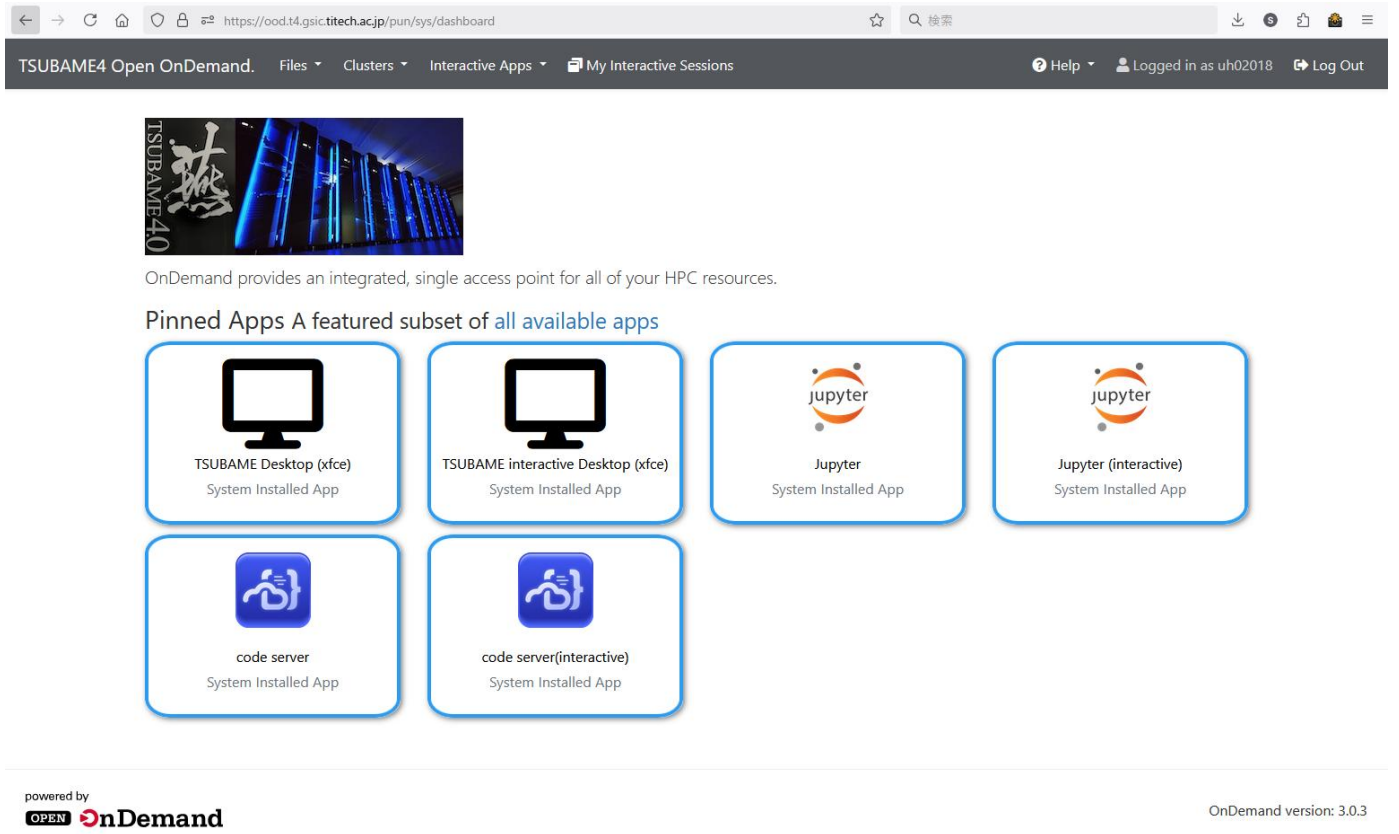


TSUBAME4.0 Open OnDemandへ移行

- TSUBAME3のものは独自実装、TSUBAME利用者ポータルへの作りこみ
  - TSUBAMEにべったりで他のシステムへ移植不可能
  - システム提供ベンダが変われば東工大(当時)ですら活用不能になる
- TSUBAME4導入時には国内でもOpen OnDemandが普及し始めてきた
  - では、乗り換えましょう
- まずはTSUBAME3でできていたことの再現
  - Jupyter
  - Code Server
  - X11
- ログインノードの仮想コンソール・ファイル転送
- まだできていないこと
  - スケジューラ周り (Jub Submitter, Active Jobs)
    - ジョブ簡易投入
    - 非OODジョブの管理(状態確認etc)
    - (あるけどあまりうまく動いてないので隠してる)
  - ウィジェット関係
    - ストレージの混み具合はできるだけ早期に対応したい
- ログインはメール認証(ポータルと独立)
  - 4月にPassKey(FIDO2)対応予定!!

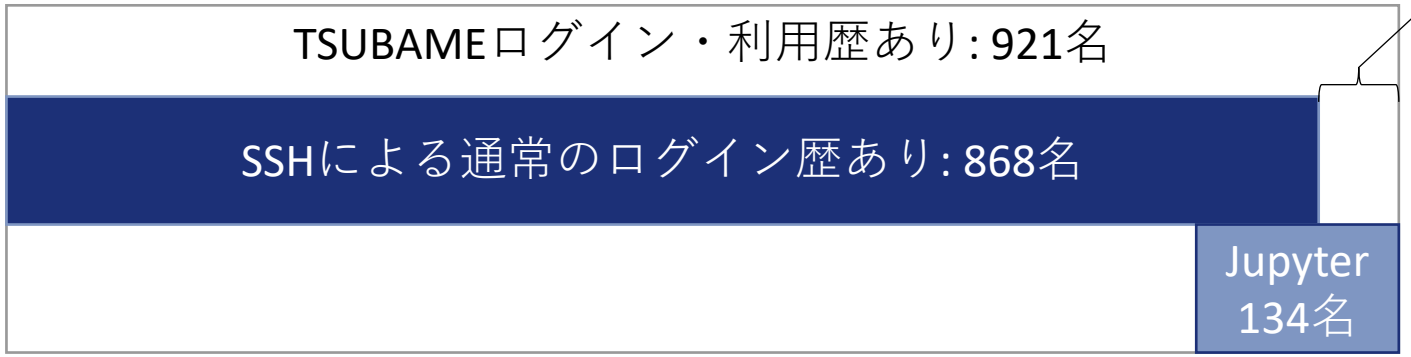
サーバスペック (AMに話題になったので)

AMD EPYC 7443 (2.85GHz, 24core, 200W) x 2
Mem 256GiB, SSD 1.6TB, IB HDR 200Gbps, Ethernet 10Gbps
(一部は認証用のKeycloak VMに使ってる)

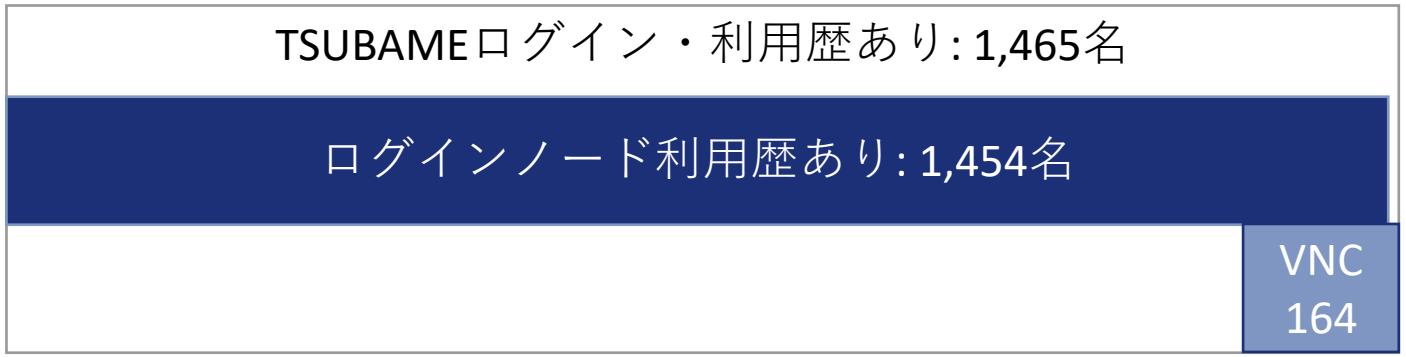


Web経由の利用でユーザは増えた?

- 端的に言えば、Yes
- TSUBAME3に導入後3か月のユーザ動向: シェア14.5%, 新規発掘ユーザ6%



- TSUBAME4運用開始後8か月(11月末まで)のユーザ動向:



- OOD経由でもログインノードを使えるようになった
- OOD自体のログイン人数の統計を取っていない(今後の課題)
- ユーザの絶対数が増えている

まとめ

- TSUBAME4のOpen OnDemandに至るまでの「前史」(TSUBAME3時代)
  - Jupyter Labを簡単につかいたいな から始まる独自実装の旅
  - Code Server・NoVNC、欲しかったので追加
    - Jupyterの段階で何かが欲しくなるのは読めていたので、追加できるような構造に
- TSUBAME4でのOOD利用状況
  - 思ったほどユーザは広がっていない
    - 現状提供している物がビギナー向けなものが多く、スパコン使い倒す勢には不十分?
  - カスタマイズ性に難ありか
    - JupyterのPython環境切り替えなど、T3時代にできていた一部の機能を取りこぼしている
      - 現状はjupyterrunスクリプトに丸投げ、そちらではVirtualEnvなども利用可能
  - バージョンアップ・アプリを増やすことができる環境を準備中
    - いまのところセンター教員が触れる検証環境がない → 準備中(年度末)