2021年2月2日
「不老」オンライン利用説明会（一般および企業利用）

大島聡史（名古屋大学情報基盤センター）

Type IIサブシステムの機械学習ユーザ向け情報とクラウドシステムの時刻指定ジョブ実行について

本資料で紹介する内容

- Type IIサブシステムの機械学習ユーザ向け情報
  - ソフトウエア環境の整備
  - コンテナの活用
  - SSD
  - JupyterLab (Jupyter Notebook)
- クラウドシステムの時刻指定ジョブ実行
  - クラウドシステムのサポートする利用形態とその使い方
    - バッチジョブ実行
    - 時刻指定実行
      - 時刻指定バッチジョブ実行
      - 時刻指定インタラクティブ実行

本資料で紹介する内容

- Type IIサブシステムの機械学習ユーザ向け情報
  - ソフトウエア環境の整備
  - コンテナの活用
  - SSD
  - JupyterLab (Jupyter Notebook)
- クラウドシステムの時刻指定ジョブ実行
  - クラウドシステムのサポートする利用形態とその使い方
    - バッチジョブ実行
    - 時刻指定実行
      - 時刻指定バッチジョブ実行
      - 時刻指定インタラクティブ実行

機械学習アプリ・フレームワークの整備状況について

- 「不老」ではmoduleコマンドでアプリケーションを管理
- Type IIサブシステムにはTensorFlowやPyTorchなどがmoduleで用意されている

- 足りないモジュールがある場合や最新のフレームワークが使いたい場合は……
  - pip, venv, condaなどを用いてローカルディレクトリにインストールする
    - ほとんどのPython関係モジュールはこれらで簡単に整備可能
  - コンテナを利用する
    - Singularityが利用可能、Dockerコンテナを実行できる
    - Docker HubやSingularity Hubに公開されているコンテナを簡単に利用できる
    - NVIDIAの提供するGPU環境向けコンテナ（NGC）も有用
      - GPUを用いて機械学習を行えるようなコンテナが整備されている

利用例1：公開されているコンテナをそのまま利用する

- Docker HubやSingularity Hubなどで公開されているコンテナをそのまま使う方法
- インタラクティブジョブで実行する例

```
[a49979a@flow-cx02 ~]$ pjsub --interact -L rscgrp=cx-workshop,jobenv=singularity
インタラクティブジョブが起動されて計算ノードにログインした状態になる
[a49979a@cx120 ~]$ module load singularity
[a49979a@cx120 ~]$ singularity shell --nv docker://<イメージの在処>/
Singularity> これ以降、入力したコマンドはコンテナ上で実行される
```

- バッチジョブで実行する例
  - 以下のようなファイルを用意
  - pjsubコマンドでジョブを投入

```
#!/bin/bash
#PJM -L rscgrp=cx-workshop
#PJM -L jobenv=singularity
module load singularity
singularity run --nv docker://<イメージの在処>/
singularity exec --nv docker://<イメージの在処>/ コンテナ上で実行したいコマンド
```

- 初回実行時はコンテナイメージのダウンロードに時間がかかるが、~/.singularity/cache/以下にキャッシュされ、二回目からは起動時間が短縮される
- runはコンテナを起動するのみ（組み込まれたコマンドを実行）
- execは引数で指定したコマンドを実行

利用例2：.defファイルでコンテナイメージを作成して利用する

- .defファイル：コンテナのインストール（構築）手順を書いたもの、定義ファイル
  - DockerにおけるDockerfileに相当
- Docker HubからTensorFlowのコンテナを持ってくるだけの例
  - test.def

```
Bootstrap: docker
From:tensorflow/tensorflow:latest-gpu
```

  - 実行例

```
[a49979a@cx120 ~]$ singularity build -f ./test.sif ./test.def
INFO:    Starting build...
Getting image source signatures
 (省略)
INFO:    Creating SIF file...
INFO:    Build complete: ./test.sif
[a49979a@cx120 ~]$ singularity shell --nv ./test.sif
Singularity>
```

- さらに%postにapt/yumやpipなどを書いてソフトウェア環境を揃えたコンテナを作るという使い方が一般的
- 詳しい書き方はドキュメントを参照

```
Bootstrap: docker
From:tensorflow/tensorflow:latest-gpu

%post
apt-get install -y emacs
pip3 install pandas
```

利用例3：手持ちのDockerコンテナから変換して利用する

- 既に別環境で構築済みのDockerコンテナ（イメージ）を持っている場合、変換すればSingularityで利用できる
  - ※動かせないものがある可能性については未調査

既存環境（手元のPCなど）にてDockerコンテナをイメージファイルに保存する

```
[a49979a@cx120 ~]$ docker save 既存のイメージ > ./tmp.tar
```

保存したファイルをscpやsftpで「不老」にコピーする

コピーしたファイルをsingularity buildで変換する

```
[a49979a@cx120 ~]$ singularity build ./container.sif docker-archive:./tmp.tar
```

変換後は通常のコンテナイメージとして利用可能

```
[a49979a@cx120 ~]$ singularity shell --nv ./container.sif
Singularity>
```

コンテナイメージを編集したい場合は？

- コンテナイメージ自体を編集してオリジナルのコンテナを作ることも可能
- システムの都合上、計算ノードに搭載されたSSD上で作業を行う必要がある

- 作業手順
  1. sandboxコンテナを作成する（sandboxコンテナに変換する）
     - `singularity build -s ./workdir_on_ssd docker://container`
  2. 書き込み用のオプションを付けてコンテナを起動し、更新作業を行う
     - `singularity shell -w -f ./workdir_on_ssd`
  3. コンテナイメージに変換して利用する（変換せずに使い続けても構わないが、SSD上のデータはジョブ終了時に消えるため注意が必要）
     - `singularity build -f container.sif ./workdir_on_ssd`

Type IIサブシステムに搭載されたSSDについて

- Type IIサブシステムの各計算ノードには6.4TBのNVMe SSDが搭載されている
- ジョブの設定とリソースグループによってSSDの見え方は異なるが、ホットストレージより高いアクセス性能が期待できる
  - 機械学習に限らず、同じファイルに何度もアクセスしたりランダムアクセスが多い場合に有用

- そのまま使う、BeeONDとして使う、NVMeshとして使う、の3つの使い方がある
  - そのまま使う
    - ジョブ実行時に各ノードの専用ディレクトリにマウント、ジョブの開始終了時に削除される
  - BeeONDとして使う
    - ジョブ実行時のみ有効な共有ストレージ、複数ノードから同じファイルが見えるようになる
  - NVMeshとして使う
    - ジョブ実行時以外もデータが残るBeeOND、利用には申請が必要

JupyterLab (Jupyter Notebook)

- 機械学習利用者に人気のあるJupyterLabも利用可能
  - moduleで導入済み、もちろん個別にインストールしても良い
  - 利用手順：バッチジョブやインタラクティブジョブでjupyterコマンドを実行し、その出力情報を見てsshポート転送を行い、Webブラウザでアクセスする
  - バッチジョブスクリプト例

    job_jupyter_single.sh

```
#!/bin/bash
#PJM -L rscgrp=cx-single
module load gcc
module load python/3.7.6
module load jupyterlab/2.1.5
jupyter notebook --notebook-dir=${HOME} --ip='*' --port=8888 --no-browser --allow-root
```

  - 接続手順

    上記バッチジョブが実行開始した後、
    $ ssh -L 8888:計算ノード名:8888 ユーザID@flow-cx.cc.nagoya-u.ac.jp
    のようにSSHポート転送を行い、ブラウザで　http://127.0.0.1:8888/?token=……　にアクセスすることで利用可能。GPUも利用可能。

本資料で紹介する内容

- Type IIサブシステムの機械学習ユーザ向け情報
  - ソフトウエア環境の整備
  - コンテナの活用
  - SSD
  - JupyterLab (Jupyter Notebook)
- クラウドシステムの時刻指定ジョブ実行
  - クラウドシステムのサポートする利用形態とその使い方
    - バッチジョブ実行
    - 時刻指定実行
      - 時刻指定バッチジョブ実行
      - 時刻指定インタラクティブ実行

クラウドシステムの利用形態

- バッチジョブ実行 → Type I, II, III同様のバッチジョブ実行、ログインノードからジョブ投入
- 時刻指定実行
  - 時刻指定バッチジョブ実行
  - 時刻指定インタラクティブ実行

  → 実行時刻を指定した実行
  → 専用のWebシステムを用いて操作
  → 指定にあわせて仮想マシンが起動する

- 全100ノードをバッチジョブ実行用と時刻指定実行用に分けて利用
  - 割合は利用状況を見て調整
- 準占有制度による利用も可能（バッチジョブ実行のみ対応）
  - 4ソケットCPU+384GiBメモリを一ヶ月間使い続けることが可能

時刻指定実行の特徴

- 通常のバッチジョブ実行は、ジョブスケジューラによる調整の結果で実際のジョブ開始時刻が決まる
- 時刻指定実行は、ジョブ開始時刻を指定してジョブの予約をすることができる
  - 専用のWebシステムから予約を行う
  - HWトラブルなど特別な事情がない限り、指定した時刻に開始する
  - 計算資源が足りない場合や同時実行制限を超える場合は予約ができない

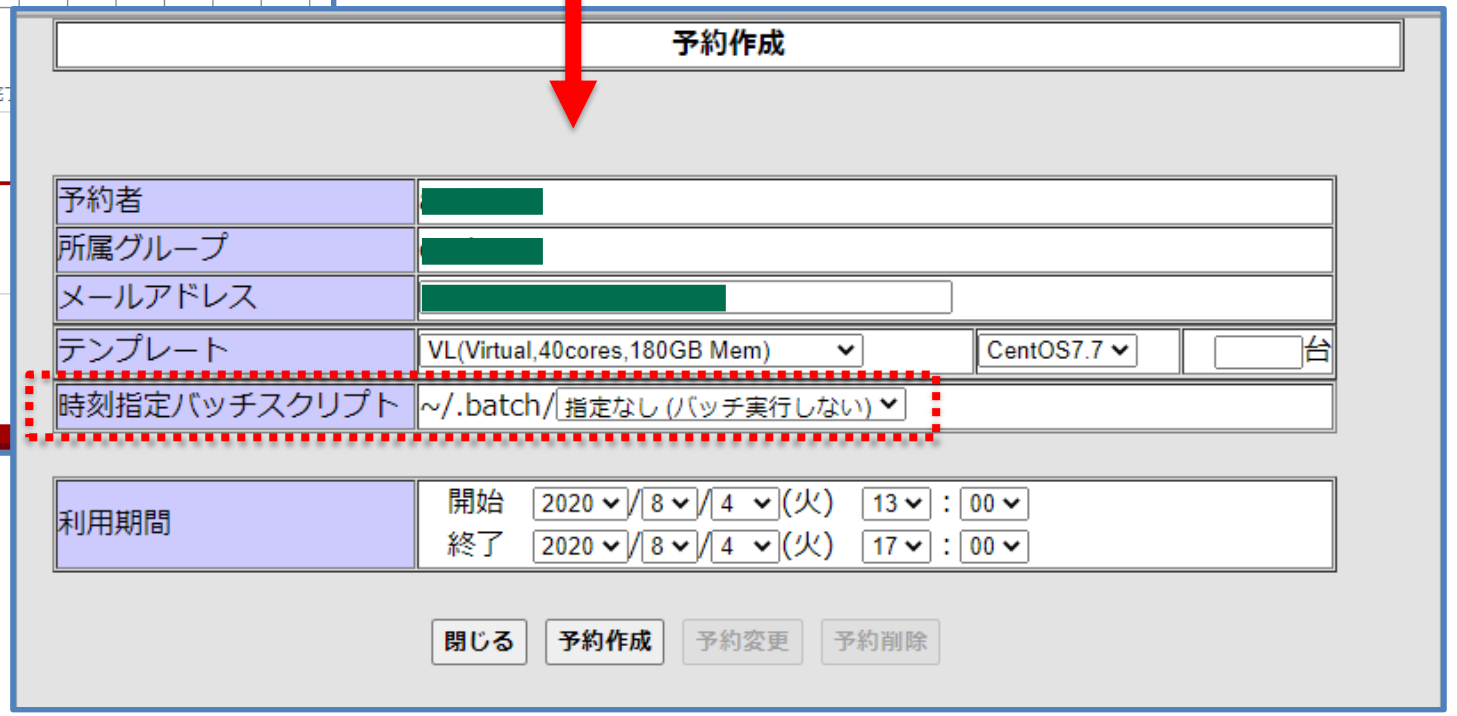
- 時刻指定実行の種類
  - 時刻指定バッチジョブ実行
    - 指定した時刻になると、Webシステムから選択しておいたスクリプトが自動的に実行開始される
  - 時刻指定インタラクティブ実行
    - 指定時刻にあわせて仮想マシンが用意され起動される
    - ログインノードからクラウドシステム上の計算資源にsshアクセスできるようになる
    - ログインのための情報はメールで送られてくる

時刻指定実行の操作手順

- 時刻指定バッチジョブ
  - あらかじめバッチジョブスクリプトを用意しておく
  - ブラウザで専用Webシステムにアクセス
  - 実行時刻、利用資源、実行するスクリプトを選択して登録する
  - 予約した時刻になるとジョブが自動的に実行開始される
  - 実際にジョブが実行された時間分だけポイントが消費される
- 時刻指定インタラクティブ実行
  - ブラウザで専用Webシステムにアクセス
  - 実行時刻と利用資源を選択して登録する
  - メールでsshアクセス情報（IPアドレス情報）が送られてくる
  - 予約した時刻以降、ログインノードからsshでアクセス可能になる
  - ログインしていたかどうかとは関係なく、確保した時間分だけポイントが消費される（Webから明示的に終了させればその時点で消費が止まる）

専用Webシステム（UNCAI）の操作イメージ

1. 予約表から使いたい時間帯を指定、または「予約作成」ボタン
2. 具体的な利用資源と予約時間帯を指定する



その他、変更、中止などもWeb予約表から行う

時刻指定インタラクティブ実行の確認メールの例

ご予約されていたリソースが利用可能となりました。
予約内容、および、割り当てられたFloating IPをご連絡いたします。
Floating IPで対象のノードにアクセスが可能です。

Reserve ID : xxxxxxxxxxxxxxx
(予約ID)

User : a49999a
(予約者)

Template : VX
(テンプレート)

OS image : CentOS-7.7
(OSイメージ)

Number of host(s) : 1
(予約数)

Batch script : (No execution)
(時刻指定バッチスクリプト)

Reservation term : 2020/06/27 14:00 - 2020/06/27 20:00
(利用期間)

Floating IP address
(Floating IP)
xx.xx.xx.xx

←ログインノードからここに書かれたIPアドレスにsshできる

クラウドシステムでもJupyterLab(Jupyter notebook)を利用可能

- バッチジョブ実行で利用する
  - Type IIサブシステムと同様の使い方（リソースグループをクラウドシステム向けに変更する）
  - GPUがないため、あまり高い性能は期待できない

- UNCAIから利用する
  - 10コアCPUから利用できるため、使い方次第では経済的
  - 準備と接続に少し手間がかかる
    - moduleが用意されていないため、自分でインストールする必要がある（簡単）
    - 現状ではSSHポート転送が2段階必要
  - 時刻指定バッチジョブを使えば仮想マシン起動と同時にJupyterLabを起動可能

start_script_jupyter_conda

```
#!/bin/bash
eval "$(~/miniconda3/bin/conda shell.bash hook)"
conda activate testenv
jupyter notebook --notebook-dir=${HOME} --ip='*' --port=8888 --no-browser --allow-root
```