



OACIS講習会 (session2)

村瀬洋介^{1,2} , 内種岳詞^{1,2}

1: 理化学研究所, 2: JST CREST

2015/10/21 OACIS公開ソフト講習会



- session1
 - 13:00 – 13:20 概要説明
 - 13:20 – 13:50 シミュレーション実行ハンズオン
- session2
 - 14:00 – 14:50 simulator, analyzer, host登録のデモ
- session3
 - 15:00 – 15:50 各自の環境に合わせたOACISのセットアップ（希望者）

- Simulatorの登録
 - Simulatorの要件
 - パラメータの渡し方
 - 結果のファイルの取り込み方
 - ジョブ投入時の挙動
- Analyzerの登録
 - Analyzerとは？
 - Analyzerの要件
 - ジョブ投入時の挙動

Simulator 登録

Simulatorの要件

以下の要件を満たすコマンドをSimulatorとして登録できる

- 引数またはJSONでパラメータを受け取ること（次スライドで解説）
- 出力ファイルはカレントディレクトリ以下に作られること
- 以下の名前のファイルがあっても干渉しないこと
 - `_input.json`, `_output.json`, `_status.json`, `_time.txt`,
`_version.txt`, `_log.txt`, `_stdout.txt`, `_stderr.txt`
- 正常終了で0、異常終了で0以外の返り値を返すこと

パラメータの渡し方

- 引数渡しの場合
 - 登録したコマンドにパラメータが順番に引数として渡される。ただし最後は乱数の種
 - `~/simulator.out <p1> <p2> <p3> ... <seed>`
- JSON渡しの場合
 - 実行時に `_input.json` というファイルが作成され、その中にパラメータが記述される
 - `{ “p1” : 30, “p2” : 10, “_seed” : 12345 }`

出力ファイルの取り込み方

- 基本的にはカレントディレクトリ以下にファイルを出力するようにすればよいだけ
- プロットしたい結果がある場合には
 - “_output.json” というファイル名でJSON形式で出力する
 - キーの名前は任意で良い
- bmp, png, jpgなどの形式で出力しておけば一括閲覧できる (epsは不可)

```
{
  velocity: 0.25841666666666667,
  flow: 0.077524999999999968
}
```

Simulatorの登録

New Simulator

Name

Definition of Parameters
_seed seed -
Random number seed

[Add Parameter](#)

Pre process script

Command

Print version command

Input type

Support mpi

Support omp

Description

Executable_on localhost

Simulatorの名前

Parameterの定義
名前、型、デフォルト値

プリプロセスのコマンド
(詳細後述)

Simulation実行コマンド

バージョン取得コマンド

JSON入力か引数入力か

MPI, OpenMP並列化か

どのようなSimulatorか
についてのメモ

実行可能なホスト

Simulator登録時のTips

- コマンドは、ホームディレクトリからの相対パスで書くのがオススメ
 - 絶対パスだとホストごとに異なる場合がある
- ログインノードでしかできないコマンドはプリプロセスとして登録する
 - 入力ファイルの準備など
- Simulatorのバージョンを記録すると一括削除や置換ができる

- MPI並列の場合、 `mpiexec -n $OACIS_MPI_PROCS ...` というコマンドを実行コマンドとして登録する
- OpenMP並列の場合、 `OMP_NUM_THREADS`環境変数に指定した値が入る

Create a new parameter set on: echo

p1 (Integer)	<input type="text" value="0"/>
p2 (Float)	<input type="text" value="1.0"/>
Target # of Runs	<input type="text" value="1"/>
Submitted to	<input type="text" value="localhost"/>
Priorities of Runs	<input type="text" value="normal"/>
MPI procs	<input type="text" value="4"/>
OMP threads	<input type="text" value="2"/>
	<input type="button" value="Create"/> <input type="button" value="Cancel"/>

Run作成時に指定可能に

ジョブスクリプトの例

```
#!/bin/bash
export LANG=C
export LC_ALL=C

# VARIABLE DEFINITIONS -----
OACIS_JOB_ID=5624ae1364663800e7d40000
OACIS_IS_MPI_JOB=true
OACIS_MPI_PROCS=4
OACIS_OMP_THREADS=2
OACIS_PRINT_VERSION_COMMAND="echo 'version 1'"

# PRE-PROCESS -----
if [ `basename $(pwd)` != ${OACIS_JOB_ID} ]; then # for manual submission
  mkdir -p ${OACIS_JOB_ID} && cd ${OACIS_JOB_ID}
  if [ -e ../${OACIS_JOB_ID}_input.json ]; then
    ¥mv ../${OACIS_JOB_ID}_input.json ./_input.json
  fi
fi
echo "{" > ../${OACIS_JOB_ID}_status.json
echo " ¥"started_at¥": ¥"date`¥", " >> ../${OACIS_JOB_ID}_status.json
echo " ¥"hostname¥": ¥"hostname`¥", " >> ../${OACIS_JOB_ID}_status.json

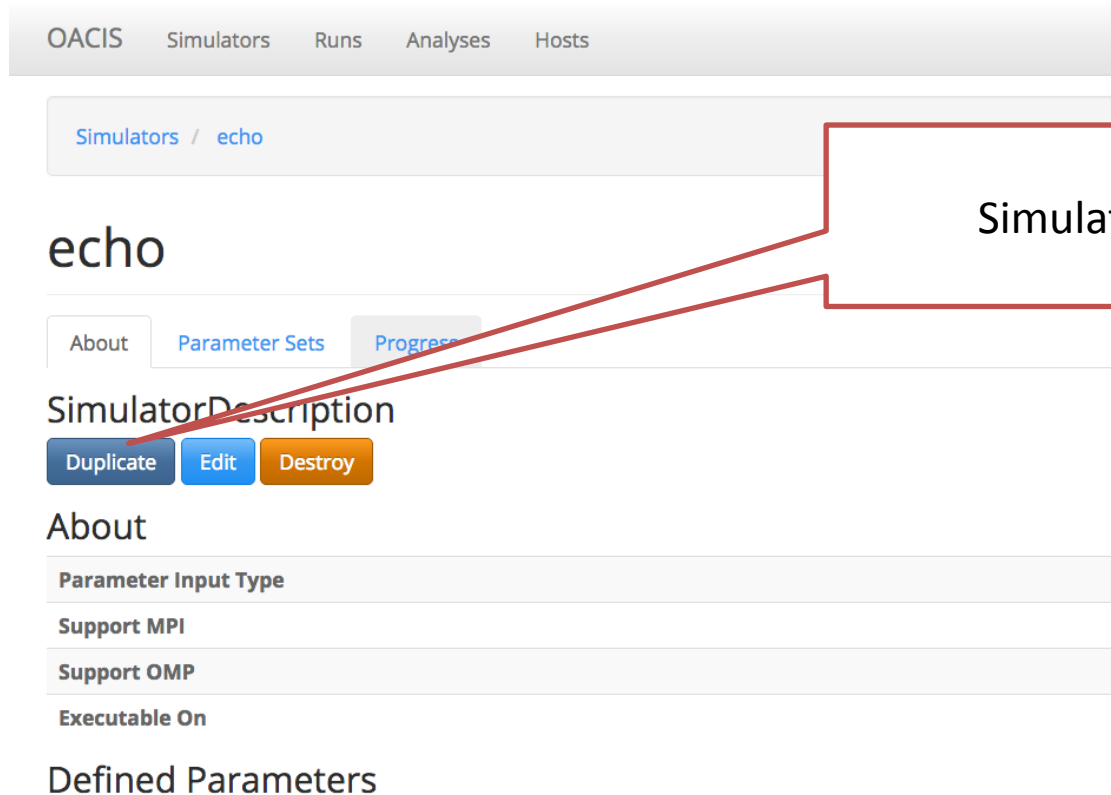
# PRINT SIMULATOR VERSION -----
if [ -n "$OACIS_PRINT_VERSION_COMMAND" ]; then
  (eval ${OACIS_PRINT_VERSION_COMMAND}) > _version.txt
fi

# JOB EXECUTION -----
export OMP_NUM_THREADS=${OACIS_OMP_THREADS}
{ time -p { { echo 3 1.0 1473031699; } 1>> _stdout.txt 2>> _stderr.txt; } } 2>> ../${OACIS_JOB_ID}_time.txt
RC=$?
echo " ¥"rc¥": $RC, " >> ../${OACIS_JOB_ID}_status.json
echo " ¥"finished_at¥": ¥"date`¥" >> ../${OACIS_JOB_ID}_status.json
echo "}" >> ../${OACIS_JOB_ID}_status.json
```

Runの画面から実際に生成されたスクリプトを確認することも可能

Simulatorの複製

- ちょっと仕様の違うSimulatorを新たに登録する場合、既存のSimulatorを複製して作成すると楽。



OACIS Simulators Runs Analyses Hosts

Simulators / echo

echo

About Parameter Sets Progress

SimulatorDescription

[Duplicate](#) [Edit](#) [Destroy](#)

About

Parameter Input Type

Support MPI

Support OMP

Executable On

Defined Parameters

Simulatorの複製

Analyzer 登録

Analyzerとは

- Runの実行後に行うポスト処理をAnalyzerという形で登録することができる
 - リモートホストでは実行できない処理、追加で実行したくなった処理をAnalyzerとして登録するとよい
- OACISには2種類のAnalyzerを登録可能。
 - Runに対するAnalyzer
 - 結果の可視化
 - ParameterSetに対するAnalyzer
 - ParameterSet配下のすべてのRunに対する統計処理

Analyzer登録

- Simulator登録とほぼ同様
- 異なるのは実行時に既存のRunの結果が `_input/` にコピーされること
- どのような形式で入力ファイル (Runの結果ファイル) が準備されるかはドキュメントを参照

Edit Analyzer

Name:

Type:

Definition of Parameters

Pre process script:

Command:

Print version command:

Input type:

Support mpi:

Support omp:

Auto run:

Description:

Executable on: localhost

Host for Auto Run:

自動実行

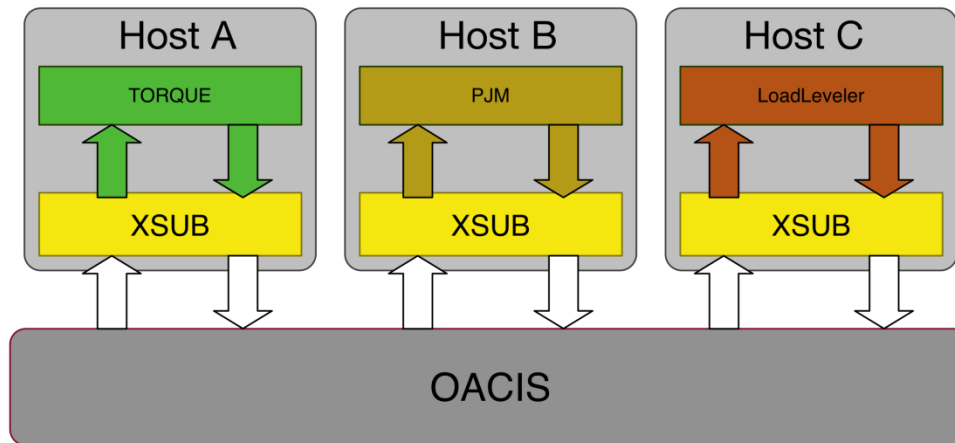
- Runの終了時にAnalyzerを自動的に作成することができる (auto_runフラグ)
 - Runに対するAnalyzer : Yes/No から選択
 - ParameterSetに対するAnalyzer: Yes/First run only/No から選択

Host登録

ホストの登録は以下の手順で行う

- 鍵認証でリモートログインできるようにする
 - パスフレーズが必要な場合は、ssh-agent を利用して
パスフレーズを入力した状態でOACISを再起動
- xsubを導入する
- OACISにホストを登録する

xsubとは？



- ジョブスケジューラは仕様が様々で、方言も存在
- その違いを吸収するためのスクリプトをリモートホストにおいて、OACISからは `xsub`, `xstat`, `xdel` という統一したコマンドを呼べるようにしている
- 現在はRubyで実装されている (が、将来はpythonで実装したい…)

Create New Runs

# of Runs	<input type="text" value="1"/>	
Submitted to	<input type="text" value="k"/>	
Priorities of Runs	<input type="text" value="normal"/>	
MPI procs	<input type="text" value="1"/>	
elapse	<input type="text" value="1:00:00"/>	Format: / [^] \d+:\d{2}:\d{2}\$/
node	<input type="text" value="1"/>	Format: / [^] \d+(x\d+){0,2}\$/
shape	<input type="text" value="1"/>	Format: / [^] \d+(x\d+){0,2}\$/

Runの作成時にジョブカードに必要なパラメータが求められるようになる

xsubの導入

- リモートホストで
 - git clone <https://github.com/crest-cassia/xsub.git>
- .bash_profile を編集。(OACISはリモートログインするときにはbashログインシェルからコマンドを実行する)
 - export PATH= “\$HOME/xsub/bin:\$PATH”
 - export XSUB_TYPE= “torque”
- 現状、サポートされているのは、none, torque, FX10, K, SR16000のみ。必要に応じて追加するので、ご相談ください

OACISへのホストの登録

OACIS Simulators Runs Analyses Hosts

New host

Name

Hostname

Polling Status

User

Port

Ssh key

Work base dir

Mounted work base dir

Max num jobs

Polling interval

MPI processes ~

OMP threads ~

Executable Simulators NagelSchreckenberg
 echo

Executable Analyzers

SSHの接続先の情報

ジョブに使うディレクトリ
(この下にディレクトリを掘って
ジョブが実行される)

ジョブを何本まで投げるか

ジョブの完了確認や投入を何
秒ごとに行うか

このホストで実行が許される
MPI,OpenMPの並列数
Runを作るときにチェック

実行可能シミュレーター

ステージングがある場合

- プリプロセスで実行プログラムなど必要なものをすべて、カレントディレクトリにコピー
- 実行コマンドをカレントディレクトリからのコマンドにする
 - `mpirexec -n $OACIS_MPI_PROCS ./a.out`
- カレントディレクトリのファイルすべてをステージイン、ステージアウトする仕様
 - ランクディレクトリ未対応

まとめ

- Simulator登録、Host登録、Analyzer登録のデモを行った
 - これらの登録は煩雑だが、一度登録すれば便利に使うことができる
 - メーリングリスト
 - `oacis-users@googlegroups.com`
 - アップデートの通知など
- ⇒ Session 3 ではハンズオン形式で各自の環境でお試しいただく