

バッチジョブ入門(bash)

Akitomo Enokizono

Sampleコード

- 去年の蜂谷さんのsampleコードを参考に、簡略化・bash化しました
 - Fun4Allをつかったバッチジョブは今回はなし
- まず以下のファイルを自分の適当なディレクトリにコピーしてください
 - /sphenix/user/enokiaki/sample/batchjob_sample_v2.tar.gz
- 解凍してください
 - tar xvzf batchjob_sample_v2.tar.gz

```
sphnx04:tmp>ls batchjob_sample_v2
README  example0/  example1/  example2/  example3/  example4/  test.C
```

example0

- バッチジョブを実行するマシンでの環境変数について理解しよう

```
example0/  
├─ condor.job  
└─ showenv.sh
```

condor.job

```
Executable = showenv.sh  
Output     = condor_example0.out  
Error      = condor_example0.err  
Log        = condor_example0.log  
Queue
```

showenv.sh

```
#!/bin/bash  
  
echo "User name : " $USER  
echo "Home directory: " $HOME  
echo "Path : " $PATH  
echo "Current directory : " $PWD  
echo "Hostname : " $HOSTNAME  
echo "ROOTSYS : " $ROOTSYS
```

- 1) まずは"showenv.sh"をそのまま実行して自分のsphinx解析マシン上での環境変数
を確かめよう
- 2) "condor_submit condor.job"でcondor上でshowenv.shを実行して、結果の
condor_example0.outをチェックしてみよう
- 3) 最後にcondor.jobに"Getenv = True"という行を書き足して再度実行して結果を見
てみよう

condorコマンドとステータス

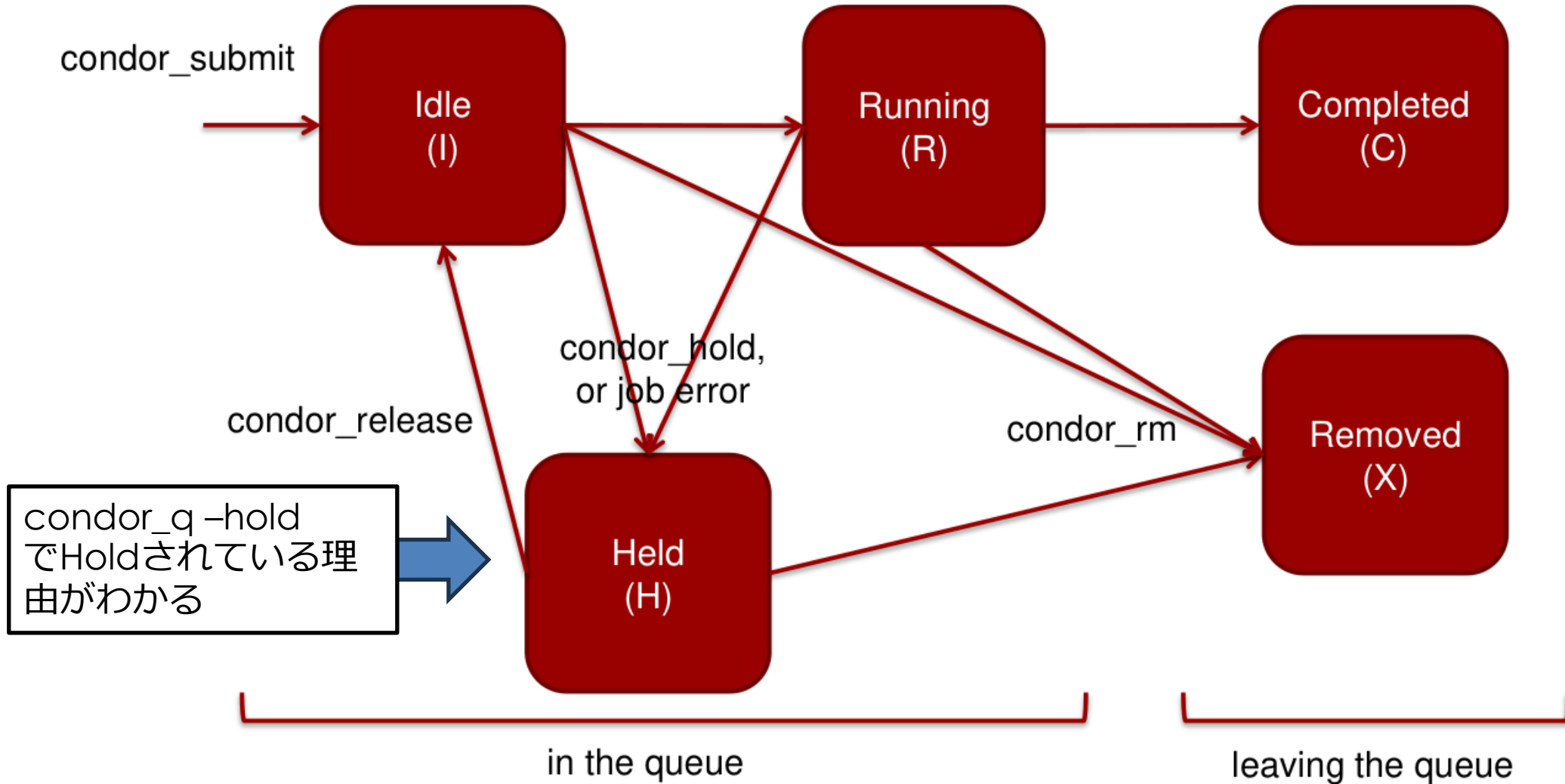
蜂谷さんの前回のバッチジョブ入門より

CONDORコマンド

- `condor_submit <ジョブファイル>` : ジョブを投げる
- `condor_q` : ジョブの状況を確認する
 - `condor_q -long` : 細かい状況
- `condor_rm <ユーザー名>` : ジョブを停止・削除
 - `Condor_rm <ジョブID>`
- `condor_status` : コンピュータの状況を見る
 - `condor_status -submitters`

condorコマンドとステータス

https://indico.cern.ch/event/611296/contributions/2604378/attachments/1471728/2277589/TannenbaumT_Policy.pptx



example1

- ROOTスクリプトをcondor上で実行しよう
- バッチジョブを実行するマシンでの実行ディレクトリの場所について理解しよう

```
example1/  
├─ condor.job  
├─ roottest.sh  
└─ workdir
```

condor.job

```
Executable      = roottest.sh  
Output          = condor_example1.out  
Error           = condor_example1.err  
Log             = condor_example1.log  
Getenv         = True  
Initialdir     = workdir  
Queue
```

roottest.sh

```
#!/bin/bash  
  
#source /opt/sphenix/core/bin/sphenix_setup.sh -n  
  
echo "User name : " $USER  
echo "Home directory: " $HOME  
echo "Path : " $PATH  
echo "Current directory : " $PWD  
echo "Hostname : " $HOSTNAME  
echo "ROOTSYS : " $ROOTSYS  
  
root -b -g ../test.C
```

- まずroottest.shを実行して../test.C（1つ上のディレクトリにあるROOTマクロ）を実行しよう
- 次にcondorを使い同じROOTマクロを"workdir"ディレクトリ内で実行してみよう
 - ジョブが失敗したら、エラーメッセージを確認してなにが問題か確かめて、修正して正しく再処理してみよう

example1

- test.C

```
void test(const int runnum=40000)
{
    std::cout<<"Processing run "<<runnum<<std::endl;
    TFile *f = new TFile(Form("testfile_run%d.root",runnum),"recreate");

    // You will process events, then write histograms or tree branches here.

    f->Write();
    f->Close();
}
```

example2

- 色々なバッチファイルの作り方について学ぼう

```
example2/  
└─ roottest2.sh  
└─ makejob.sh
```

roottest2.sh

```
#!/bin/bash  
  
#source /opt/sphenix/core/bin/sphenix_setup.sh -n  
  
echo "User name : " $USER  
echo "Home directory: " $HOME  
echo "Path : " $PATH  
echo "Current directory : " $PWD  
echo "Hostname : " $HOSTNAME  
echo "ROOTSYS : " $ROOTSYS  
  
root -b -q ../test.C\($1\)
```

makejob.sh

```
#!/bin/bash  
  
runs=( 35122 35223 35224 35421 35442 )  
  
for run in "${runs[@]}" ; do  
  
    jobfile="condor_run${run}.job"  
  
    echo "Executable    = roottest2.sh"           > $jobfile  
    echo "Arguments     = ${run}"                 >> $jobfile  
    echo "Output        = condor_run${run}.out"   >> $jobfile  
    echo "Error         = condor_run${run}.err"   >> $jobfile  
    echo "Log           = condor_run${run}.log"   >> $jobfile  
    echo "Getenv        = True"                   >> $jobfile  
    echo "Queue"                               >> $jobfile  
  
    condor_submit ${jobfile}  
  
done
```

- まずroottest2.shに適切なRun番号を引数にして実行しよう
 - roottest2.sh 12345
- 次にmakejob.shで複数のRunごとのcondor_runXXXXX.jobファイルを作成してcondorに投げよう

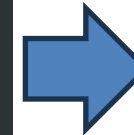
example3

- 色々なバッチファイルの作り方について学ぼう

```
example3/  
└─ makejob.sh  
└─ roottest2.sh
```

makejob.sh

```
#!/bin/bash  
  
runs=( 35122 35223 35224 35421 35442 )  
  
jobfile="condor_allrun.job"  
  
echo "Executable = roottest2.sh" > $jobfile  
echo "Log = condor_runall.log" >> $jobfile  
echo "Getenv = True" >> $jobfile  
  
for run in "${runs[@]}"; do  
    echo "#-----" >> $jobfile  
    echo "Arguments = ${run}" >> $jobfile  
    echo "Output = condor_run${run}.out" >> $jobfile  
    echo "Error = condor_run${run}.err" >> $jobfile  
    echo "Queue" >> $jobfile  
  
done  
  
condor submit ${jobfile}
```



condor_allrun.job

```
Executable = roottest2.sh  
Log = condor_runall.log  
Getenv = True  
#-----  
Arguments = 35122  
Output = condor_run35122.out  
Error = condor_run35122.err  
Queue  
#-----  
Arguments = 35223  
Output = condor_run35223.out  
Error = condor_run35223.err  
Queue  
#-----  
Arguments = 35224  
Output = condor_run35224.out  
Error = condor_run35224.err  
Queue  
#-----  
Arguments = 35421  
Output = condor_run35421.out  
Error = condor_run35421.err  
Queue  
#-----  
Arguments = 35442  
Output = condor_run35442.out  
Error = condor_run35442.err  
Queue
```

- 次にmakejob.shで複数のRunのジョブを記述したファイル (condor_allrun.job) を作成してcondorに投げよう

example4

- 色々なバッチファイルの作り方について学ぼう

condor.job

run.list

```
example4/  
├── condor.job  
├── run.list  
└── roottest2.sh
```

```
Executable = roottest2.sh  
Arguments  = $(arg1)  
Output     = condor_run$(arg1).out  
Error      = condor_run$(arg1).err  
Log        = condor_runall.log  
Getenv     = True  
Queue arg1 from run.list
```

```
35122  
35223  
35224  
35421  
35442
```

- もし引数が2つ以上ある場合は例えばcondor.job内で

Arguments = \$(run) \$(intt) \$(DAC0cut)

Queue run intt DAC0cut from run.list

のように記述し、run.listを

35122 intt0 35

35223 intt5 30

35224 intt0 20 のように記述しておけばよい

その他のオプション

- +JobFlavor (or +MaxRuntime) [*https://batchdocs.web.cern.ch/local/submit.html](https://batchdocs.web.cern.ch/local/submit.html)


Setting the job flavour in the submit file is achieved like this:

```
+JobFlavour = "longlunch"
```

Setting manually can be achieved by placing the following in your submit file:

```
+MaxRuntime = Number of seconds
```

```
espresso      = 20 minutes
microcentury  = 1 hour
longlunch     = 2 hours
workday       = 8 hours
tomorrow      = 1 day
testmatch     = 3 days
nextweek      = 1 week
```



- Request_memory
 - 標準は1500MB
 - それ以上、例えば4000MB欲しい時は
Request_memory 4000
と要求する