TITech Grid 上での Condor を用いた Gaussian の利用

■はじめに

本手引きは TITech Grid 上で、Gaussian のジョブを Condor を用いて実行するための方法について解説したものです。Titech GRID 上では Gaussian98 及び Gaussian03 を利用できます。

■TITech Grid における Condor の構成

TITech Grid 上の Condor システムは以下のような構成となっています。
Condor の詳細については別資料 (xx) を参照してください。

・Submit ホスト (ジョブ投入ホスト)
tgn002001.g.gsic.titech.ac.jp (172.17.2.1), tgn003001 (172.17.3.1), tgn003002 (172.17.3.2), tgnbin (172.17.1.1)

・計算ホスト
センターA (256 cpu/ 128 ノード), センター B1 および B2 (256 cpu/ 128 ノード), サテライトノード(168 cpu/ 82 ノード)
計 680 cpu / 340 ノード (1 ノード当り Pentium III 1.4GHz x 2 cpu, 1GB メモリ, 60GB スクラッチディスク領域)

■Titech GRID へのログイン

Submit ホストへ ssh または telnet でログインしてください。

■Titech GRID 上での Gaussian 環境変数の設定

Gaussian を実行するためには、最初に環境変数の設定が必要です。下記のとおり実行してから、Gaussian を利用するようにしてください。

csh の場合

```
% setenv   g98root /usr/local              または    % setenv   g03rooot   /usr/local
% source   $g98root/g98/bsd/g98.login      または    % source   $g03root/g03/bsd/g03.login
% setenv   GAUSS_SCRDIR   /work
```

bash の場合

```
$ export   g98root=/usr/local            または     $ export   g03root=/usr/local/
$ .  $g98root/g98/bsd/g98.profile        または   $  .  $g03root/g03/bsd/g03.profile
$ export   GAUSS_SCRDIR=/work
```

■Condor を用いた Gaussian の利用

(1) ジョブの作成

NQS などのバッチシステムに Gaussian ジョブを投入して実行する場合、従来次のようなスクリプトファイルを作成していました。

```
#!/bin/csh -f
#QSUB -l mpp_p=2
#QSUB -q E

setenv   g98root   /usr/local
source $g98root/g98/bsd/g98.login
setenv   GAUSS_SCRDIR /work

cd   ${HOME}/gaussian

g98 < h2o.dat >& h2o.out
```

例 1：h2o.csh

Condor に Gaussian ジョブを投入する場合、上記のようなスクリプトファイルに加えて、実行プログラムやログファイル等を記述した Submit ファイル(次項参照) を作成する必要があります。また、Condor のジョブを実行する際には、ジョブを投入するマシンから実行するマシンへのデータ転送は行われません。そのため、入出力ファイル等は各実行マシンから参照できるよう、NFS で共有されている /home 以下にこれらのファイルを置くようにしてください。なお、G98、G03 などの Gaussian 実行コマンドは、あらかじめ全てのマシンで共有されている /usr/local にインストールされています。

(2) Submit ファイルの作成

Submit ファイルは Condor システムにジョブの情報を知らせるための設定ファイルです。前項で示したスクリプトファイル h2o.csh を Condor ジョブとして実行するための Submit ファイルは次のようになります。

```
Universe       = vanilla
Executable   = h2o.csh
Error          = h2o.err
Log            = h2o.log
Environment = HOME=/home/cc/gusr17
Initialdir      = /home/cc/gusr17/test/G98
Queue          1
```

例 2：シェルスクリプトをジョブとした submit ファイル

Submit ファイルでは「設定項目＝内容」の形式で 1 項目につき 1 行づつ記述していきます。上記の設定項目の意味は次のとおりとなります。

- Universe ： ジョブの種類を表します。Gaussian ジョブの場合は常に vanilla を指定してください。
- Executable ： 実際に実行するコマンドを記述します。ここでは h2o.csh シェルスクリプトをコマンドとして指定します。
- Error ： 標準エラーを記録するファイル名を指定します。
- Log ： ジョブの実行ログが逐次的に記述されるファイル名を指定します。
- Environment ： 実行時の環境変数を指定できます。「環境変数＝値」の形式で記述し、複数ある場合は「;」で区切ります。
- Initialdir ： Error、Log などのファイルが保存されるディレクトリを指定します。ここでは、/home/cc/gusr17/test/G98 以下に h2o.err, h2o.log が保存されます。
- Queue ： ジョブの投入数を指定できます。上記の場合、このジョブが 1 つキューに入り、実行されます。この項目は「=」がないのに注意してください。また、必ず Submit ファイルの最後に記述してください。

以上の例では h2o.csh スクリプトの中で Gaussian への入出力ファイルを指定していましたが、Condor の Submit ファイルの中で直接、実行プログラムや入出力ファイルを指定することもできます。ただしこの場合は、Condor へのジョブの投入時に Gaussian 実行のために必要な環境変数があらかじめ設定されている必要があります。環境の設定については「■Titech GRID 上での Gaussian 環境変数の設定」を参照ください。

```
Universe    = vanilla
Executable = /usr/local/g98/g98
Initialdir     = /home/cc/gusr17/test/G98
Input         = h2o.dat
Output      = h2o.out
Error         = h2o.log
Getenv      = True
Queue       1
```

例 3：実行コマンドをジョブとした submit ファイル h2o.sub

上記の submit ファイルは、基本的には例 1 h2o.csh の最後の 1 行

g98 < h2o.dat > h2o.out

の部分をジョブとして指定したことになります。例 2 で説明されなかった設定項目の意味はつぎのとおりです。

Input ： 標準入力として読み込まれるファイルを指定します。 Initialdir で指定されたディレクトリ内にある必要があります。

Output ： 標準出力の保存先となるファイルを指定します。Initialdir で指定されたディレクトリに生成されます。

Getenv ： submit マシン上での環境変数を引き継ぐための設定です。Gaussian ジョブの場合には常に True を指定してください。

これまでの submit ファイルの例では、どちらもあるひとつのジョブを投入して実行していますが、Condor では、入力ファイルと出力ファイルの違う複数のジョブを同時に実行することもできます。異なる入出力ファイルを指定して、 Gaussian ジョブを複数同時実行する場合の submit ファイルの例を以下に示します。

```
Universe      = vanilla
Executable    = /usr/local/g98/g98
Intialdir     = /home/cc/gusr17/test/G98/
Input         = sample.$(process)
Output        = sample.$(cluster).$(process)
Error         = sample.$(cluster).$(process)
Log           = sample.log
Getenv        = True
Notification  = always
Notify_user   = gusr17@xxx.titech.ac.jp
Queue         3
```

例 4：入出力ファイルが異なる複数ジョブの実行

Submit ファイル内で使用されている $(process) と $(cluster) は、このファイル内でのみ使用できるマクロ変数になります。Condor ではジョブが投入された際、それぞれのジョブにジョブ番号と、そのジョブ内のプロセス番号が割り振られますが、$(cluster) がこのジョブ番号を表し、$(process) がプロセス番号を表します。例 4 を実行した場合の例をつぎに示します。ジョブ投入時にジョブ番号が 5 であったとして、Queue 3 を指定しているのでプロセス番号 0, 1, 2 の計 3 プロセスが同時に実行されます。

```
例 4 の submit ファイルが ジョブ番号 5 の場合…
プロセス番号 0 のジョブ:  g98 < sample.0.dat > sample.5.0.out   エラー出力は sample.5.0.err
プロセス番号 1 のジョブ:  g98 < sample.1.dat > sample.5.1.out   エラー出力は sample.5.1.err
プロセス番号 2 のジョブ:  g98 < sample.2.dat > sample.5.2.out   エラー出力は sample.5.2.err
以上のジョブが生成され、同時に実行されます。
```

```
また、それぞれの入出力ファイルは/home/cc/gusr17/test/G98/ ディレクトリが参照されます
Log は追加記述されるので sample.log のみで良いです
```

cluster と process

また、Notification、Notify_user の項目は、ジョブの終了をメールで自動通知するための設定になります。Notification= Always となっているので、実行が終了し次第、gusr17@xxx.titech.ac.jp 宛に通知メールが送られます。

(3) 環境の確認

Condor ジョブが実行される計算ノードのことを Condor プールといいます。Condor プールの現在の状態を確認したい場合は condor_status コマンドを使用します。

```
% condor_status
Name               OpSys     Arch   State     Activity   LoadAv Mem   ActvtyTime

vm1@tgn002002.cl.t LINUX     INTEL  Owner     Idle       0.920  504   0+00:08:29
vm2@tgn002002.cl.t LINUX     INTEL  Unclaimed Idle       0.000  504   0+03:38:30
vm1@tgn002003.cl.t LINUX     INTEL  Owner     Idle       0.810  504   0+00:03:28
vm2@tgn002003.cl.t LINUX     INTEL  Unclaimed Idle       0.000  504   0+03:33:30

                     Machines Owner Claimed Unclaimed Matched Preempting

        INTEL/LINUX       518    22       3       493       0          0
              Total       518    22       3        49       0          0
```

condor_status による確認

Owner はそのマシンの所有者によって使われている状態、Claimed は Condor のジョブが動作している状態、Unclaimed は空き資源でジョブが実行されていな状態をそれぞれ示しています。

(4) ジョブの投入

ジョブを投入するためには submit マシン上で condor_submit コマンドを使用します。

```
% condor_submit h2o.sub
Submitting job(s).
Logging submit event(s).
1 job(s) submitted to cluster 27.
```

condor_submit によるジョブの投入

condor_submit コマンドの引数には submit ファイルを指定します。上記の例では、h2o.sub を Condor ジョブとして投入しています。正しく投入されれば、そのジョブのジョブ番号(cluster)が表示されます。ここでは 27 がジョブ番号となります。

(5) 実行状態の表示

投入したジョブの状態を確認する場合には condor_q コマンドを使用します。

```
% condor_q
-- Submitter: tgn002001.g.gsic.titech.ac.jp : <172.17.2.1:32784> : tgn002001.g.gsic.titech.ac.jp
 ID      OWNER            SUBMITTED     RUN_TIME ST PRI SIZE CMD
27.0   gusr17          6/16 16:35   0+00:00:00 I  0   0.0  g98
1 jobs; 1 idle, 0 running, 0 held
```

condor_q による実行中の表示

その submit マシンのキューに投入されているジョブを表示します。項目 ST はステータスを表します。I は Idle(実行待ち)、R は Running(実行中)、H は Hold(停止)をそれぞれ意味します。

(6) ジョブ終了の確認

終了を確認するには、(1)log ファイルを見る、(2)メールによる結果の確認、(3)condor_history コマンド、の 3 通りの確認方法があります。それぞれ確認できる内容がことなりますので、適切な方法を使用してください。

(1) log ファイルの確認

submit ファイル内の Log の項目で指定したファイルに計算に使用した時間などが出力されています。

```
000 (8135.000.000) 05/25 19:10:03 Job submitted from host: <128.105.146.14:1816>
...
001 (8135.000.000) 05/25 19:12:17 Job executing on host: <128.105.165.131:1026>
...
005 (8135.000.000) 05/25 19:13:06 Job terminated.
        (1) Normal termination (return value 0)
                Usr 0 00:00:37, Sys 0 00:00:00  -  Run Remote Usage
                Usr 0 00:00:00, Sys 0 00:00:05  -  Run Local Usage
                Usr 0 00:00:37, Sys 0 00:00:00  -  Total Remote Usage
                Usr 0 00:00:00, Sys 0 00:00:05  -  Total Local Usage
        9624  -  Run Bytes Sent By Job
        7146159  -  Run Bytes Received By Job
        9624  -  Total Bytes Sent By Job
        7146159  -  Total Bytes Received By Job
```

log ファイルの内容

(2) メールによる確認

Submit ファイル内の Notify_user で指定したメールアドレスに次のような通知メールが送られます。

```
From condor   Wed Jun 16 17:09:23 2004
Date: Wed, 16 Jun 2004 17:09:23 +0900
From: condor@tgn002001.g.gsic.titech.ac.jp
To: gusr17@tgn002001.g.gsic.titech.ac.jp
```

```
Subject: [Condor] Condor Job 30.0
This is an automated email from the Condor system
on machine "tgn002001.g.gsic.titech.ac.jp".   Do not reply.
```

メールによる確認

(3) condor_history コマンド

```
%condor_history
ID     OWNER          SUBMITTED      RUN_TIME     ST  COMPLETED      CMD
1.0    gusr20         3/17 10:32     0+00:03:03   C   3/17 10:35     /home/cc/gusr20
3.0    gsic005        5/8  11:51     0+00:00:00   X   5/8  11:52     /bin/hostname-a
4.0    gsic005        5/8  11:55     0+00:00:01   X   5/8  11:55     /bin/hostname
5.0    gsic005        5/13 16:36     0+00:00:02   C   5/13 16:36     /home/exp/gsic0
```

condor_history による履歴の確認

condor_history では、submit マシン上で既に実行が終了、あるいは中断されて condor_q では表示できないジョブを確認することができます。項目 ST はステータスを表します。C はそのジョブが正常に実行、完了したことを表し、X は condor_rm によってジョブを削除されたことを表しています。

■その他の機能

Condor を利用するにあたり、以下のコマンドを用いてジョブの操作が可能です。

(1) condor_rm

実行中のジョブを中断し、キューから削除したい場合は condor_rm コマンドを使用します。C ジョブを削除するには、condor_submit を実行した submit マシンで condor_rm を実行する必要があります。また、削除できるのは自分のジョブのみです。削除の対象はジョブ番号(cluster)もしくは、ジョブ番号.プロセス番号(process)で指定することができます。

```
例：ジョブ ID が 5 の場合。現在、キューに 5.0, 5.1, 5.2 のジョブがある場合
   % condor_rm 5
で、5.0, 5.1, 5.2 全てのジョブが削除されます。
  % condor_rm 5.1
で、5.1 のみ削除されます。また、自分のジョブ全てを削除したい場合は
  % condor_rm -all
で削除可能です
```

condor_rm によるジョブの削除

(2) condor_hold & condor_release

キュー内のジョブを一時的に停止したい場合は condor_hold コマンドを実行します。停止したいジョブが現在実行中であれば、そのジョブは計算ノードで kill されます。停止中のジョブは

condor_q で確認すると、ST が H になり、停止している間はジョブが計算ノードに割り振られることはありません。停止しているジョブを再スタートするには condor_release コマンドを実行します。

```
ジョブの停止
% condor_hold ジョブ番号.プロセス番号
ジョブの再開
%condor_release ジョブ番号.プロセス番号
```

condor_hold＆condor_release によるジョブの停止･再開

(3) condor_prio

投入したジョブの優先度を変更することができます。なお、優先度の高いジョブから実行が開始されます。優先度を変更できるのは自分のジョブのみです。優先度の指定は下記のように、+(数字)で優先度を高く、-(数字)で優先度を低くします。

```
% condor_q
-- Submitter: perdita.cs.wisc.edu : <128.105.165.34:1027> :
 ID      OWNER            SUBMITTED     RUN_TIME ST PRI SIZE CMD
  1.0   frieda           6/16 06:52   0+00:02:11  R   0   0.0   my_job
% condor_prio +5 1.0
% condor_q
-- Submitter: perdita.cs.wisc.edu : <128.105.165.34:1027> :
 ID      OWNER            SUBMITTED     RUN_TIME ST PRI SIZE CMD
  1.0   frieda           6/16 06:52   0+00:02:13  R   5   0.0   my_job
```

condor_prio による優先度の変更

作成	秋山智宏	日本 SGI 株式会社
監修	山梨　毅	東京工業大学　学術情報部情報基盤課

発行	平成 16 年 7 月 9 日