Utilização do Ambiente de processamento paralelo do LAD

contato.lad@pucrs.br

suporte.lad@pucrs.br

Índice:

Introdução

O LAD é um laboratório que provê recursos computacionais de alto desempenho para os grupos de pesquisa da PUCRS e pesquisadores externos ligados a esses grupos. Nele, os pesquisadores podem contar com a disposição de um grande conjunto de máquinas e dispositivos de armazenamento para a realização de tarefas com alta demanda computacional.

O LAD presta serviços de alocação de servidores 24x7 e de clusters de processamento paralelo. Por meio deste documento trataremos sobre a infraestrutura e a utilização do ambiente de processamento paralelo do laboratório, servindo como um manual para os usuários. Também abordaremos a utilização do MPI, que é bastante comum neste tipo de ambiente.

1 Infraestrutura

No LAD, temos uma máquina responsável pela gerência dos recursos disponíveis nos clusters, a qual é chamada de "Marfim". Podemos dizer que se trata de uma "central de acessos", onde os usuários irão alocar e acessar as máquinas dos clusters. Por esse motivo, na Marfim não deve ser feito nenhum processamento pesado.

Atualmente o LAD possui 5 clusters. Sâo eles: Amazônia, Atlântica, Cerrado, Pantanal e Nimbus. Parte dos equipamentos do LAD são reservados para alocação 24x7, onde os usuários tem acesso administrativo a uma máquina física ou virtual para hospedagem de serviços que devem ficar disponíveis 24 horas por dia. Outra parte é utilizada para processamento em cluster, onde os usuários fazem alocações momentâneas de máquinas com maior capacidade computacional para a execução de aplicações paralelas. Para mais informações sobre cada cluster, acesse:

http://www3.pucrs.br/portal/page/portal/ideia/Capa/LAD/LADInfraestrutura/LADInfraestruturaHardware

1.1 Acesso ao LAD

Na PUCRS temos uma rede chamada "PORTOALEGRE". Essa é a rede comum da universidade, administrada pelo GTIT e onde praticamente todas as estações de trabalho e laboratórios estão conectados. A PUCRS também possui outras redes, porém somente a rede Wi-Fi denominada "PUCRS" faz parte da rede PORTOALEGRE.

A infraestrutura do LAD é uma subrede autônoma da rede PORTOALEGRE, ou seja, a equipe tem autonomia na configuração e administração da subrede, porém está sob as politicas do GTIT para acessos externos ao ambiente do LAD. O acesso ao LAD é permitido para toda a rede PORTOALEGRE. Redes Wi-Fi como "eduroam" e "portal" são consideradas inseguras pelo GTIT e, por isso, não tem acesso ao LAD. Para usuários localizados fora da universidade, o acesso é permitido apenas através de um IP estático da internet. Caso o usuário possua um, o mesmo poderá solicitar ao LAD a liberação para seu IP.

Quando estamos conectados à internet, temos um ponto de acesso que possui um endereço IP visível na internet, esse é o que chamamos aqui de "IP da internet" (ou IP público). Existem muitos sites onde é possível analisar sobre qual IP uma máquina ou uma rede se conecta à internet. Consideramos um IP "estático" da internet quando a máquina ou rede utiliza sempre o mesmo endereço IP para se conectar à internet. Os usuários comuns de internet banda larga costumam utilizar IP's dinamicamente disponibilizados pelas operadoras, ou seja, toda vez que o usuário se autentica na operadora (por exemplo, quando o modem é reiniciado), a operadora provê um endereço IP diferente, inviabilizando a criação de regras para acesso ao LAD.

Universidades e grandes empresas possuem contratos especiais com as operadoras, utilizando IP's estáticos que permitem a criação das regras.

Para acessar os serviços do LAD é necessário possuir uma conta. Para solicitar uma conta para um projeto/grupo existente, basta mandar a solicitação (informando o nome completo, e-mail e matricula do requerente) para o e-mail "suporte.lad@lista.pucrs.br" com cópia ao coordenador do grupo, para que o mesmo aprove o cadastro da nova conta. No caso da criação de um novo grupo, a solicitação deve ser feita no sistema do Instituto Idéia através do site:

http://www3.pucrs.br/portal/page/portal/ideia/Capa/

As contas cadastradas são nominais, ou seja, cada usuário possuí uma conta e é responsável pelo que é feito através dela. Por esse motivo, passamos algumas indicações básicas:

- Que as contas/senhas não sejam compartilhadas a terceiros
  - Senhas anotadas em locais acessíveis por terceiros podem gerar utilização indevida da conta.
- Utilize uma senha complexa, contendo letras, números e/ou caracteres especiais.
  - Muitos estudos indicam que as senhas mais seguras são as mais extensas, mesmo que ofereçam um grau menor de complexidade.
- Não esqueça de alterar a senha no primeiro acesso, pois só então ela será 100% privada.

Após obter uma conta, deve-se efetuar o login no servidor de acesso do LAD através de SSH, conforme exemplo abaixo executado em um terminal Linux:

```
# ssh <usuário>@marfim.lad.pucrs.br
```

Atenção: A notação "<>" é usada para variáveis e também será usada em outros exemplos neste documento. No caso acima, quando o comando for executado, "<usuário>" deve ser alterado para o nome de usuário (username) da conta, por exemplo "joao.silva".

Para usuários de máquinas com o Sistema Operacional Windows, existem diversas ferramentas para se fazer o acesso SSH, Por exemplo, o "PuTTY" (http://www.putty.org).

Para mudar a senha, após feito o SSH digite o comando abaixo e siga as instruções que serão apresentadas:

```
# yppasswd
```

1.2 Pontos de Montagem

O LAD possui alguns diretórios montados a partir de equipamentos de armazenamentos compartilhados na rede. No ambiente de cluster, os diretórios "/usr/local" (onde estão as aplicações) e "/home" (onde estão os diretórios dos usuários) estão, fisicamente, em um destes equipamentos e não nos discos de cada máquina.

Por serem diretórios compartilhados na rede, estes serão idênticos, independente de qual máquina do LAD o usuário estiver. Sendo assim, qualquer modificação feita em seus arquivos ou sub diretórios, na prática serão aplicados no equipamento compartilhado, se refletindo nas demais máquinas. Também, como o acesso é feito através da rede, a leitura e escrita de arquivos podem ser mais lentos quando comparados a um disco local.

1.3 Cópia de Arquivos/Diretórios

É possível o usuário realizar cópias de arquivos e diretórios entre a sua máquina local e o LAD, sempre através do protocolo SSH. Caso o Sistema Operacional do usuário seja o Linux, o usuário poderá copiar os arquivos através do comando "SCP". Em máquinas Windows (ou até mesmo Linux), outras ferramentas estão disponíveis para cópia de arquivos por SSH, por exemplo, o "FIleZilla" (https://filezilla-project.org).

O uso do SCP é definido abaixo:

scp <ArquivoOrigem> <LocalDestino>

Ou no caso de diretórios, para realizar a cópia devemos apenas acrescentar o parâmetro "-r" como abaixo:

scp -r <DiretórioOrigem> <LocalDestino>

Para copiar arquivos ou diretórios do LAD para a sua máquina local, temos a definição abaixo, notando que o usuário deverá estar realizando o comando localmente da sua máquina, ou seja, não conectado à Marfim.

scp <usuário>@marfim.lad.pucrs.br:<ArquivoOuDiretórioComCaminhoCompleto> <DiretórioLocalDestino>

Por exemplo, o usuário "joao.silva" deseja copiar o arquivo "arquivo.txt" localizado em "/home/joao.silva/testes/" para o seu diretório local "/home/joao/Desktop/":

scp joao.silva@marfim.lad.pucrs.br:/home/joao.silva/testes/arquivo.txt /home/joao/Desktop/

Outro exemplo, o usuário “joao.silva” deseja copiar todo o diretório “diretorio_1” localizado em “/home/joao.silva/testes/” para o seu diretório local “/home/joao/Desktop/”:

scp -r joao.silva@marfim.lad.pucrs.br:/home/joao.silva/testes/diretorio_1/ /home/joao/Desktop/

Já para copiar um arquivo da sua máquina local para o seu diretório no LAD, temos a definição abaixo, notando que o usuário deverá executar o comando também da sua máquina local.

scp <ArquivoComCaminhoCompleto> <usuário>@marfim.lad.pucrs.br:<DiretórioComCaminhoCompleto>

Por exemplo, o usuário “joao.silva” deseja copiar o arquivo “arquivo.txt” localizado na sua máquina local em “/home/joao/Desktop/” para o seu diretório no LAD “/home/joao.silva/teste/”:

scp /home/joao/Desktop/arquivo.txt joao.silva@marfim.lad.pucrs.br:/home/joao.silva/testes/

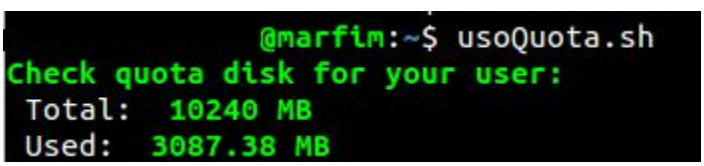
Outro exemplo, o usuário “joao.silva” deseja copiar todo o diretório “diretório_1” localizado na sua máquina local em “/home/joao/Desktop/” para o seu diretório no LAD “/home/joao.silva/teste/”:

scp -r /home/joao/Desktop/diretorio_1 joao.silva@marfim.lad.pucrs.br:/home/joao.silva/testes/

3 Quotas

O LAD possui um sistema de quota para limitar o uso de armazenamento nos diretórios de usuários (Home). Inicialmente, cada usuário possui uma quota de 10 Gigabytes. A quota pode ser visualizada quando é feito o login na Marfim.

Também é possível analisar a sua quota executando o comando “usoQuotas.sh”, como abaixo:



Havendo necessidade de um espaço de armazenamento maior, o usuário pode contactar o suporte para conversar sobre uma possível mudança ou a utilização de um novo ponto de montagem (com tamanho limitado) para uso livre do grupo (sem distinção por usuário).

4 Torque

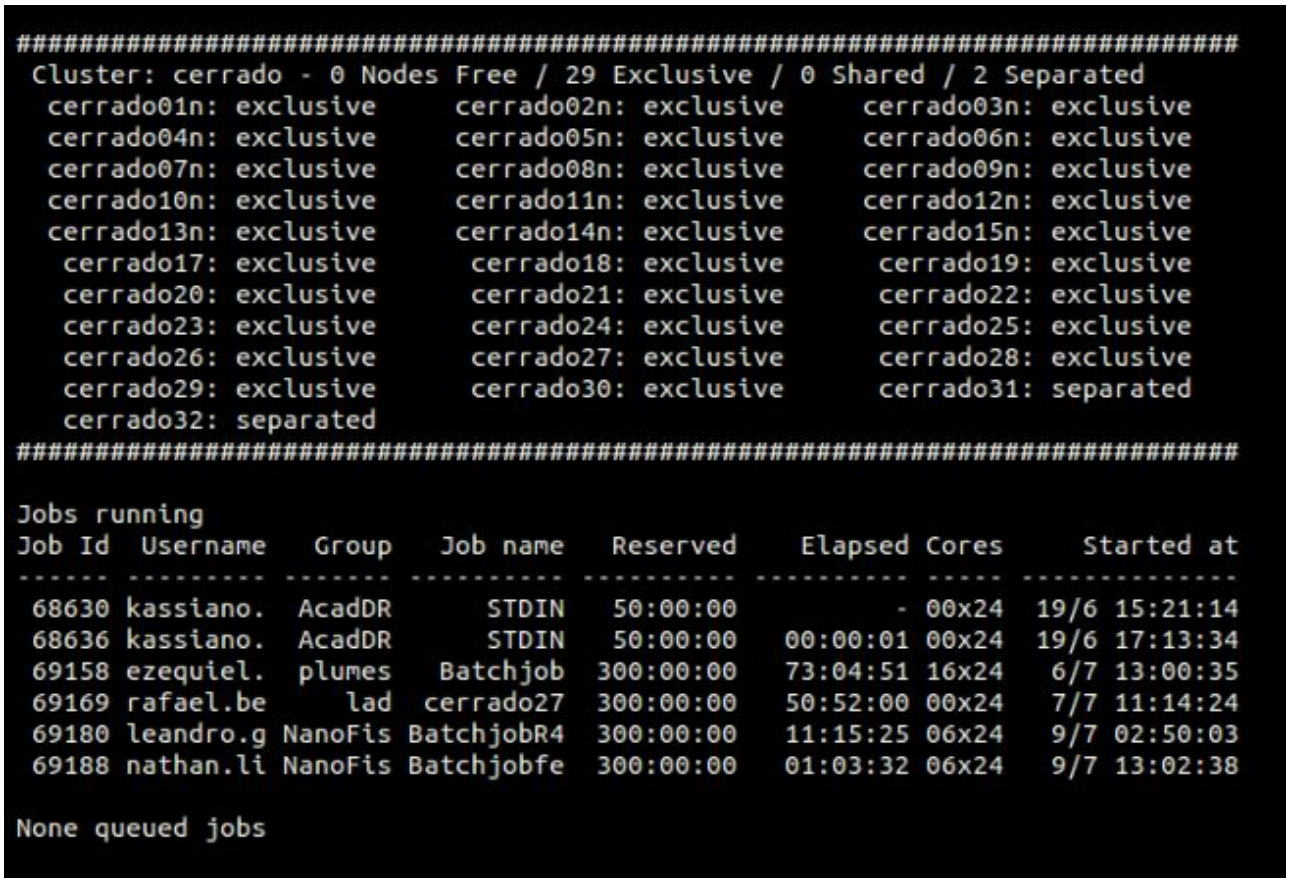
No nosso servidor de acesso (Marfim) temos o gerenciador de recursos chamado Torque, o qual é responsável pela gerência de alocações das máquinas dos clusters. Para a utilização do ambiente LAD, o Torque provê alguns comandos:

```
Programs to use the cluster:
Torque:   qsub      qdel     qstat      pbsnodes   qnodes
```

4.1 Comando "qstat"

Comando usado para verificar os estados atuais das máquinas e dos Jobs* que estão sendo executados. Ao executar o comando, haverão as informações como na imagem abaixo para cada cluster, obviamente com algumas mudanças, pois são clusters distintos com recursos e alocações diferentes.

*Job é uma sequência de processos que o usuário deseja executar em uma ou mais máquinas.



No exemplo acima temos o cluster cerrado, onde podemos analisar 2 tipos de dados. São eles:

1. **Estado do Cluster e das máquinas**, os quais possuem as seguintes informações:
    a. **cerrado**: nome do cluster.
    b. **cerradoXX**: nome da máquina.
    c. **free**: máquina com todos os processadores livres.
    d. **exclusive**: máquina com todos os processadores alocados por um ou mais usuários.
    e. **separeted**: máquina com alocação momentânea usando um ambiente diferente, onde é retirada do cluster para casos específicos.
    f. **shared**: máquina com apenas alguns processadores alocados, possibilitando outros usuários à usarem os processadores livres.

2. **Estado dos Jobs e filas**, os quais possuem as seguintes informações:
    a. **Jobs running**: atuais Jobs sendo executados no cluster.
    b. **Queued jobs**: Jobs aguardando a liberação de recursos necessários para a execução.

c. **Job Id**: número de identificação do Job.

d. **Username**: usuário que submeteu o Job.

e. **Group**: grupo do usuário.

f. **Job name**: nome do Job. Pode ser especificado no comando de alocação ou é obtido a partir do nome do arquivo Batchjob (falaremos mais sobre formas de alocação nos próximos capítulos).

g. **Reserved**: tempo reservado para o Job, ou seja, o tempo máximo que o Job poderá executar, podendo ser finalizado ou cancelado antes desse período (falaremos mais sobre os motivos para o termino antecipado nos próximos capítulos).

h. **Elapsed**: tempo que o Job está executando.

i. **Cores**: número de máquinas "X" número de processadores cada. ex.: 02x16 = 2 máquinas alocadas com 16 processadores cada uma. Caso o número de máquinas seja "00", quer dizer que o usuário especificou manualmente qual(ais) máquina(s) deseja alocar.

j. **Started at**: data e horário que o Job começou a ser executado.

4.2 Comando "pbsnodes"

Comando usado para verificar os estados de cada máquina (individualmente) de forma mais detalhada.

Ex.: máquina "cerrado27"

```
cerrado27
     state = job-exclusive
     np = 24
     properties = cluster-Cerrado
     ntype = cluster
     jobs = 0/69169.marfim.lad.pucrs.br, 1/69169.marfim.lad.pucrs.br, 2/69169.marfim.lad.pucrs.br, 3/69169.marfim.lad.pucrs.br, 4/69169.marfim.lad.pucrs.br, 5/69169.marfim.lad.pucrs.br, 6/69169.marfim.lad.pucrs.br, 7/69169.marfim.lad.pucrs.br, 8/69169.marfim.lad.pucrs.br, 9/69169.marfim.lad.pucrs.br, 10/69169.marfim.lad.pucrs.br, 11/69169.marfim.lad.pucrs.br, 12/69169.marfim.lad.pucrs.br, 13/69169.marfim.lad.pucrs.br, 14/69169.marfim.lad.pucrs.br, 15/69169.marfim.lad.pucrs.br, 16/69169.marfim.lad.pucrs.br, 17/69169.marfim.lad.pucrs.br, 18/69169.marfim.lad.pucrs.br, 19/69169.marfim.lad.pucrs.br, 20/69169.marfim.lad.pucrs.br, 21/69169.marfim.lad.pucrs.br, 22/69169.marfim.lad.pucrs.br, 23/69169.marfim.lad.pucrs.br
     status = rectime=1436461942,varattr=,jobs=69169.marfim.lad.pucrs.br,state=free,netload=59837039,gres=,loadave=0.08,ncpus=24,physmem=24677460kb,availmem=72634176kb,totmem=73505872kb,idletime=181807,nusers=2,nsessions=2,sessions=1749 2298,uname=Linux cerrado27 3.2.0-23-generic #36-Ubuntu SMP Tue Apr 10 20:39:51 UTC 2012 x86_64,opsys=linux
     gpus = 0
```

No exemplo acima temos "cerrado27", onde podemos analisar os seguintes dados:

state: estado atual da máquina.
np: número de processadores da máquina.
properties: contém o nome do cluster da máquina.
jobs: contém o número de cada processador e qual Job o alocou. ex.: 10/1234.marfim.lad.pucrs = processador número 10 está sendo utilizado pelo Job id "1234".

status: informações adicionais da máquina, como tempo de atividade, memória RAM (physmem), etc.

4.3 Comando "qdel"

Comando usado para remover algum Job criado, em execução ou na fila (aguardando liberação dos recursos). Somente usuários administradores do LAD possuem a permissão de remover Jobs de outros usuários, os demais podem remover apenas seus próprios Jobs. O uso dele é bastante simples:

qdel <job id>
exemplo: qdel 1234

4.4 Comando "qsub"

Comando usado para submeter um Job, podendo ser no modo interativo ou no modo Batchjob. Relembrando que Job se trata de uma série de processos que o usuário deseja executar em uma ou mais máquinas.

4.4.1 Parâmetros do "qsub"

O comando possui muitos parâmetros, todos podem ser consultados com detalhes utilizando o comando "man". O man é um comando de manual, que é utilizado para orientar os usuários na utilização de comandos do Linux, descrevendo funções do comandos e seus parâmetros. Para manual do "qsub", execute o comando:

man qsub

Os principais parâmetros serão detalhados abaixo:

-I = Alocação no modo interativo.

-V = Modo Verbose, mostra algumas informações adicionais do sistema Torque.

-M <email do usuário> = Define o e-mail para alerta de iniciação e finalização do job.

-d <diretório de trabalho> = Define o diretório no qual a alocação será direcionada. Caso esse parâmetro não seja especificado, o padrão é "/home/<usuário>/". Caso seja alocação Batchjob, será nessa pasta que será criado o arquivo com as saídas do Job (outputs dos comandos executados).

-l = Parâmetro mais importante. É a definição dos recursos de que o usuário deseja alocar. Há políticas de limitação (definidas pela equipe de suporte) feitas no Torque, por isso, a alocação pode ser recusada (o comando exibe um alerta de que o ambiente não oferece os recursos solicitados). Caso o usuário

deseje mais recursos do que é permitido nas politicas, o mesmo deve entrar em contato com o suporte para avaliação da necessidade.

- É importante salientar que as definições deste parâmetro se tratam apenas dos recursos reservados, e não os que serão consumidos pela aplicação. Portanto, os comandos de execução da aplicação (podem variar para cada uma) definem como os recursos reservados serão utilizados.

[nodes=<número de máquinas> | <nome da máquina>] [:ppn=<nº de processadores por máquina>] [:cluster-<nome do cluster>,walltime=<tempo a ser reservado>]

exemplo1: -l nodes=2:ppn=16:cluster-Atlantica,walltime=05:00:00

No exemplo acima, o usuário define que pretende alocar 2 máquinas do cluster Atlantica com 16 processadores cada, por um período máximo de 5 horas. Em "nodes" é possível especificar qual máquina deseja alocar, ex.: "nodes=atlantica01", e para especificar mais de uma máquina, use o exemplo abaixo.

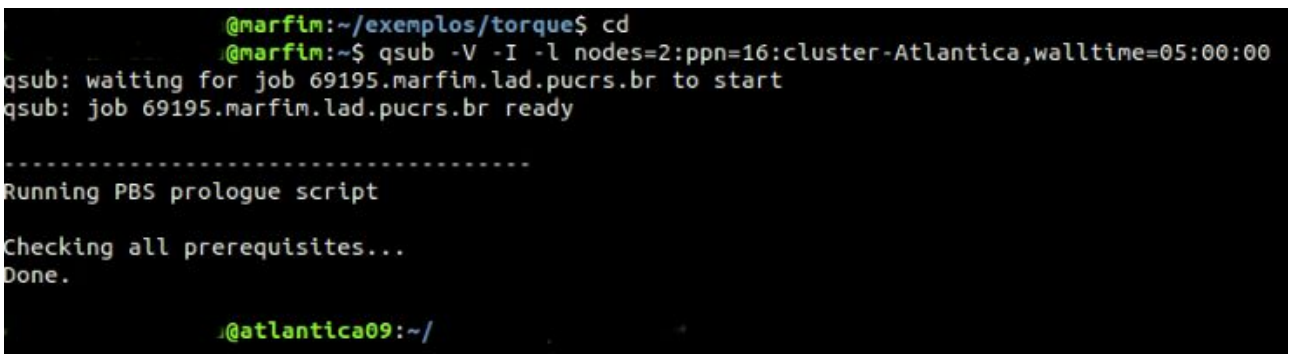
[nodes=<nome da máquina>:ppn=<ppn>+<nome da máquina>:ppn=<ppn>+...] [:cluster-<nome do cluster>,walltime=<tempo a ser reservado>]

exemplo2: -l nodes=atlantica01:ppn=16+atlantica05:ppn=16+atlantica10:ppn=16:cluster-Atlantica,walltime=05:00:00

No exemplo acima, o usuário define que pretende alocar as máquinas "atlantica01","atlantica05" e "atlantica10", todas com 16 processadores cada, por um tempo máximo de 5 horas.

4.5 Modo Interativo

É iniciada uma sessão para o usuário em uma das máquinas alocadas, onde o usuário é transferido da sua sessão atual (marfim) para essa sessão criada, como se o usuário tivesse se conectado à maquina alocada por SSH. O Job fica vinculado à essa nova sessão. Caso o usuário saia da mesma (comando "exit", por exemplo) ou a conexão seja interrompida (considerando toda a conexão, desde a máquina local do usuário, até a máquina final), o Job também é encerrado.



No exemplo acima, o usuário alocou 2 máquinas do cluster Atlantica e sua sessão foi transferida para a máquina atlantica09. Nesse caso o usuário

alocou duas máquinas, com isso o acesso à essas máquinas está liberado para ele, podendo se conectar à qualquer uma através de "ssh".

Atenção: Uma sessão ssh comum não é igual a sessão iniciada pelo Torque (através da alocação), pois as variáveis de ambiente criadas na alocação podem vir a ser utilizadas pela aplicação. Assim, é indicado que as aplicações sejam executadas através da sessão de alocação (na mesma máquina, inclusive). Caso o usuário queira fazer outras conexões com a(s) maquina(s), é aconselhado que novas sessões sejam abertas por ssh (como em um novo terminal).

4.6 Modo Batchjob

O usuário submete seus comandos em um script para ser executado em uma das máquinas alocadas. Quando acabar a execução do script, o job é finalizado e o usuário poderá ver as saídas (retornos) dos comandos executados em um arquivo ".o" (gerado no diretório especificado no parâmetro "-d" do comando qsub).
Um exemplo de Batchjob está localizado em "/home/<usuário>/exemplos/torque/Batchjob".

Lembrando que o "walltime" é exatamente como no modo Interativo, se trata do número máximo de horas alocadas, não quer dizer que ficará executando todo esse tempo. Caso ultrapasse o tempo de walltime, a execução será interrompida e o Job encerrado. Podemos analisar o arquivo de exemplo na imagem abaixo:

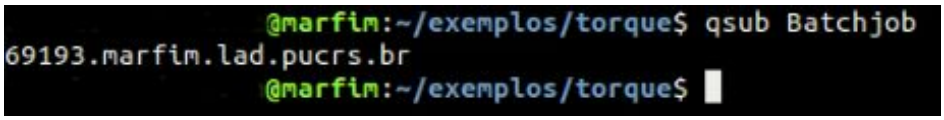
```
#!/bin/bash

################-> are comments
################-> "#PBS" are Batch Script commands

#PBS -m abe

################ Verbose mode

#PBS -V

################

################ Change these parameters according to your requisites

#PBS -l nodes=1:ppn=16:cluster-Atlantica,walltime=05:00:00

################ Where:
################ nodes = number of nodes requested
################ ppn = number of cores per node
################ cluster-Atlantica / cluster-Gates = cluster name
################ walltime = max allocation time

################ Please, change this e-mail address to yours

#PBS -M user.name@acad.pucrs.br

################

#PBS -r n

################ Output options

#PBS -j oe

################

################ Please, change this directory to your working dir.

#PBS -d /home/user.name/exemplos/torque/

################
echo Running on host `hostname`
echo
echo Initial Time is `date`
echo
echo Directory is `pwd`
echo
echo This jobs runs on the following nodes:
echo `cat $PBS_NODEFILE | uniq`
echo
echo JOB_ID:
echo `echo $PBS_JOBID`
echo ################

############# Command example, if using MPI

#mpirun -machinefile nodefile -np 16 program_mpi

################

############# If running a sequential or openMP program

#./program
sleep 10
################

echo Final Time is `date`
```

No modo Batchjob, todas as linhas que se iniciam com "#PBS" são comentários que serão interpretados como parâmetros do comando "qsub" (já

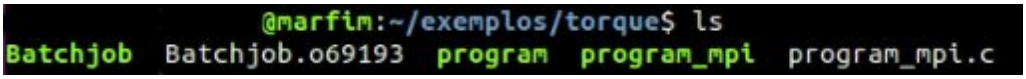
vistos anteriormente), ou seja, o usuário deve especificar os parâmetros do "qsub" no próprio script. As linhas que não iniciem especificamente com "#PBS" serão interpretadas normalmente como qualquer script bash (como comandos ou comentários). Por exemplo, linhas iniciadas em "##PBS" não serão interpretadas pelo qsub e linhas iniciadas em "PBS" serão interpretadas como um comando, PBS, que não existe.

Os comandos desejados pelo usuário devem ser inseridos após todos os parâmetros do "qsub". O próprio arquivo de exemplo contém informações sobre os parâmetros e locais para a inserção dos comandos desejados para a execução ("Command example..." e "If Running a sequential...").

Para submeter o Job no modo Batchjob, apenas execute o comando "qsub" seguido do caminho relativo do script.

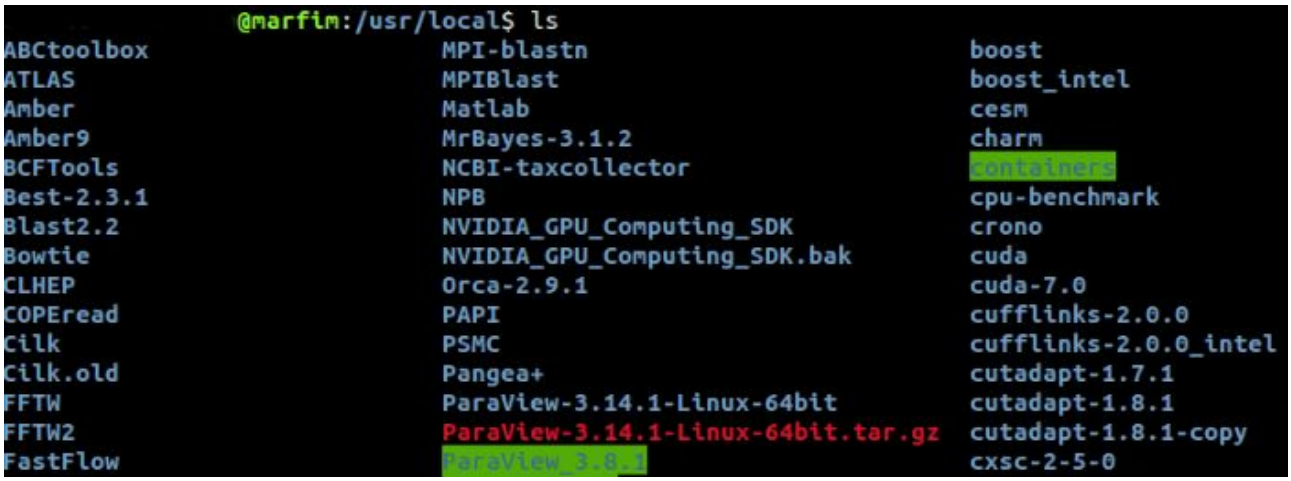


No exemplo acima temos um exemplo de submissão do script de exemplo citado anteriormente (como o arquivo está no mesmo diretório do usuário, não é necessário passar o caminho completo). O Job vai para a fila e é iniciado, executado e finalizado sem intervenções do usuário, ou seja, o usuário não precisa se manter conectado ao LAD para a execução. Ao finalizar o Job, teremos o arquivo com as saídas dos comandos executados no diretório especificado no parâmetro "-d". Caso o parâmetro não seja especificado, será criado no home do usuário "/home/<usuário>/". Na figura abaixo, o diretório definido foi "/home/<usuário>/exemplos/torque/". O nome do arquivo com os resultados sempre será "NomeDoJob"+".o"+"JobId" e o usuário terá total acesso à ele.



5 Aplicações

Na imagem abaixo, podemos ver algumas das aplicações que estão disponivéis em "/usr/local/".



As aplicações localizadas nesse diretório são instaladas pelos administradores do LAD. O usuário pode instalar aplicações que deseje no seu diretório "/home/<usuário>", sendo possível também solicitar a instalação de alguma aplicação no "/usr/local", contatando o suporte do LAD por email.

5.1 Scripts "-vars.sh"

As aplicações estão localizadas em uma unidade de armazenamento acessível para todas as máquinas dos clusters, no diretório "/usr/local/". Para a utilização das aplicações, temos os scripts "<aplicação>-vars.sh", os quais estão localizados dentro do diretório de cada aplicação. Caso a aplicação possua mais de uma versão, existirá um script para cada versão da mesma e um script sem versão definida, que será apenas um link para a versão mais recente.

Esses scripts são responsáveis por preparar todo o ambiente para a utilização da aplicação. O mais comum é a definição de variáveis de ambiente (variáveis da sessão do usuário). Na inicialização da sessão, diversas variáveis já são criadas para o uso do próprio sistema, e entre elas existe uma lista de diretórios chamada "PATH". O PATH contém os diretórios com os binários (comandos do linux) para executarmos sem precisar digitar todo o caminho do arquivo e sim apenas o "comando" em si.

Quando apenas executamos os scripts "-vars.sh" (Ex: "./script-vars.sh"), as variáveis definidas por ele serão desfeitas após a sua execução, ou seja, as variáveis serão "válidas" somente para a execução do próprio script, não se aplicando a sessão como um todo. Para que as variáveis ali definidas sejam aplicadas na sessão do usuário, devemos executar os scripts através do comando "source" (source script-vars.sh). O comando "source" irá aplicar à sessão atual todas alterações nas variáveis de ambiente feitas pelo script, e não apenas durante a execução do mesmo.

Como citado anteriormente, o mais comum destes scripts é a definição de variáveis de ambiente. Com o comando "export" podemos visualizar todas as variáveis que já estão definidas no sistema, como é mostrado na imagem abaixo:

```
            @marfim:/usr/local/Lammps$ export
declare -x HISTCONTROL="ignoreboth"
declare -x HISTTIMEFORMAT="[%D %T] "
declare -x HOME="/home/              "
declare -x INTEL_LICENSE_FILE="/usr/local/intel/licensas/intel"
declare -x LANG="en_US.UTF-8"
declare -x LC_ADDRESS="pt_BR.UTF-8"
declare -x LC_IDENTIFICATION="pt_BR.UTF-8"
declare -x LC_MEASUREMENT="pt_BR.UTF-8"
declare -x LC_MONETARY="pt_BR.UTF-8"
declare -x LC_NAME="pt_BR.UTF-8"
declare -x LC_NUMERIC="pt_BR.UTF-8"
declare -x LC_PAPER="pt_BR.UTF-8"
declare -x LC_TELEPHONE="pt_BR.UTF-8"
declare -x LC_TIME="pt_BR.UTF-8"
declare -x LD_LIBRARY_PATH="/usr/local/intel/intel-cpp/lib/intel64:/usr/local/intel/intel-cpp/mkl/lib/em64t:/usr/local/intel/intel-fort/lib/in
tel64:/usr/local/intel/intel-fort/mkl/lib/em64t::/usr/local/torque-2.5.11/lib:/usr/local/torque-2.5.11/lib/xpbs:/usr/local/torque-2.5.11/lib/x
pbsmon"
declare -x LESSCLOSE="/usr/bin/lesspipe %s %s"
declare -x LESSOPEN="| /usr/bin/lesspipe %s"
declare -x LOGNAME="              "
declare -x LS_COLORS="rs=0:di=01;34:ln=01;36:hl=44;37:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:su=37;41:sg=30;43:ca=30;4
1:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arj=01;31:*.taz=01;31:*.lzh=01;31:*.lzma=01;31:*.zip=01;31:*.z=01;31:*.Z=01;31
:*.dz=01;31:*.gz=01;31:*.bz2=01;31:*.bz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.rar=01;31:*.ace=01;31:*.zoo=01;31:
*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.jpg=01;35:*.jpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;3
5:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v
=01;35:*.mkv=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rm
vb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.axv=01;35:*.anx=01;35:*.o
gv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*
.wav=00;36:*.axa=00;36:*.oga=00;36:*.spx=00;36:*.xspf=00;36:"
declare -x MAIL="/var/mail/              "
declare -x OLDPWD="/usr/local"
declare -x PATH="/usr/local/intel/intel-cpp/bin/intel64:/usr/local/intel/intel-fort/bin/intel64:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/
bin:/sbin:/bin:/usr/games:/usr/local/crono/bin:/usr/local/crono/sbin:/usr/X11R6/bin:/usr/local/torque-2.5.11/bin:/usr/local/torque-2.5.11/sbin
:/usr/local/ladscripts"
declare -x PWD="/usr/local/Lammps"
declare -x SHELL="/bin/bash"
declare -x SHLVL="1"
declare -x SSH_CLIENT="192.168.1.220 36102 22"
declare -x SSH_CONNECTION="192.168.1.220 36102 192.168.1.79 22"
declare -x SSH_TTY="/dev/pts/18"
declare -x TERM="xterm"
```

Como exemplo, podemos ver o script "-vars.sh" da aplicação "Lammps", que é simples e não aplica muitas configurações. Esse adiciona à variável "PATH", um caminho para o diretório que contém os binários da aplicação.

```
            @marfim:/usr/local$ cd Lammps/
            @marfim:/usr/local/Lammps$ ls
Lammps-10Sept2010  Lammps-14Jul2014  lammps-10Sept2010-vars.sh  lammps-14Jul2014-vars.sh  lammps-vars.sh
            @marfim:/usr/local/Lammps$ cat lammps-vars.sh
#!/bin/bash
#
# Inserir uma chamada ao script no arquivo .bashrc do usuário da seguinte forma:
#
# source /usr/local/Lammps/lammps-vars.sh
#

export PATH=/usr/local/Lammps/Lammps-14Jul2014/bin:$PATH
```

Ao executarmos o script com o comando "source", poderemos ver a alteração feita na variável de ambiente "PATH":

```
@marfim:/usr/local/Lammps$ source lammps-vars.sh
```

```
            @marfim:/usr/local/Lammps$ export
declare -x HISTCONTROL="ignoreboth"
declare -x HISTTIMEFORMAT="[%D %T] "
declare -x HOME="/home/
declare -x INTEL_LICENSE_FILE="/usr/local/intel/licensas/intel"
declare -x LANG="en_US.UTF-8"
declare -x LC_ADDRESS="pt_BR.UTF-8"
declare -x LC_IDENTIFICATION="pt_BR.UTF-8"
declare -x LC_MEASUREMENT="pt_BR.UTF-8"
declare -x LC_MONETARY="pt_BR.UTF-8"
declare -x LC_NAME="pt_BR.UTF-8"
declare -x LC_NUMERIC="pt_BR.UTF-8"
declare -x LC_PAPER="pt_BR.UTF-8"
declare -x LC_TELEPHONE="pt_BR.UTF-8"
declare -x LC_TIME="pt_BR.UTF-8"
declare -x LD_LIBRARY_PATH="/usr/local/intel/intel-cpp/lib/intel64:/usr/local/intel/intel-cpp/mkl/lib/em64t:/usr/local/intel/intel-fort/lib/in
tel64:/usr/local/intel/intel-fort/mkl/lib/em64t::/usr/local/torque-2.5.11/lib:/usr/local/torque-2.5.11/lib/xpbs:/usr/local/torque-2.5.11/lib/x
pbsmon"
declare -x LESSCLOSE="/usr/bin/lesspipe %s %s"
declare -x LESSOPEN="| /usr/bin/lesspipe %s"
declare -x LOGNAME="                "
declare -x LS_COLORS="rs=0:di=01;34:ln=01;36:hl=44;37:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:su=37;41:sg=30;43:ca=30;4
1:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arj=01;31:*.taz=01;31:*.lzh=01;31:*.lzma=01;31:*.zip=01;31:*.z=01;31:*.Z=01;31
:*.dz=01;31:*.gz=01;31:*.bz2=01;31:*.bz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.rar=01;31:*.ace=01;31:*.zoo=01;31:
*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.jpg=01;35:*.jpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;3
5:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v
=01;35:*.mkv=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rm
vb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.axv=01;35:*.anx=01;35:*.o
gv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*
.wav=00;36:*.axa=00;36:*.oga=00;36:*.spx=00;36:*.xspf=00;36:"
declare -x MAIL="/var/mail/marlon.sanhudo"
declare -x OLDPWD="/usr/local"
declare -x PATH="/usr/local/Lammps/Lammps-14Jul2014/bin:/usr/local/intel/intel-cpp/bin/intel64:/usr/local/intel/intel-fort/bin/intel64:/usr/lo
cal/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/crono/bin:/usr/local/crono/sbin:/usr/X11R6/bin:/usr/local/torque-2
.5.11/bin:/usr/local/torque-2.5.11/sbin:/usr/local/ladscripts"
declare -x PWD="/usr/local/Lammps"
```

5.2 Arquivo ".bashrc"

No diretório "home" do usuário (/home/<usuário>/) temos um arquivo chamado ".bashrc", o qual é automaticamente executado quando uma sessão do usuário é iniciada. Caso o usuário sempre utilize uma mesma aplicação (por exemplo, "LAST"), não é necessário sempre estar executando "source /usr/local/LAST/last-vars.sh", apenas é necessário a inserção desse comando ao final do arquivo ".bashrc". Desta forma, sempre que o usuário fizer o login, o comando é executado. Não aconselhamos usar esse método, pois podem haver conflitos entre bibliotecas de diferentes aplicações que um mesmo usuário execute.

O modo aconselhado é, sempre que for utilizar uma aplicação, antes executar "source" para o arquivo "-vars.sh" da mesma, evitando que bibliotecas de outras aplicações possam gerar conflitos. No caso do batchjob, o comando pode ser inserido antes da(s) linha(s) com o(s) comando(s) de execução. Assim, caso o usuário queira usar uma outra aplicação e desfazer as definições de variáveis feitas, somente deverá encerrar a sessão atual e iniciar outra, pois as variáveis modificadas terão efeito somente na sessão atual do usuário.

6 Processamento Paralelo e MPI

O processamento paralelo se trata de uma forma de processamento onde são usados múltiplos processadores para resolver um mesmo problema, visando um menor tempo de execução. Por padrão, a grande maioria das aplicações não são paralelas, ou seja, apenas um processo é criado e, consequentemente, apenas um processador (core/núcleo) irá executar o

processo. Para que uma aplicação execute de forma paralela, é necessário que:

1. Seu código tenha sido preparado para isso, ou seja, os desenvolvedores já a tenham a programado para executar paralelamente.
2. Tenha sido compilada e instalada com as bibliotecas necessárias para tal.
    a. Dependendo das bibliotecas utilizadas, pode ser que a aplicação precise ser executada através de algum sistema "extra" (como o mpirun), que crie o ambiente para execução.

Existem várias formas de programar para que a aplicação execute em paralelo. Uma delas é através do MPI (Message Passing Interface), o qual o uso é muito comum entre os usuários do LAD. O MPI é um padrão para comunicação de dados entre processos, ou seja, permite que diferentes processos troquem mensagens entre si.

Existem diversas implementações deste padrão, como OpenMPI, IntelMPI, MPICH e outros. A utilizada como padrão do LAD é a OpenMPI, mas outras estão instaladas no "/usr/local" e o usuário pode utilizá-las. É importante frisar que o usuário só utilizará alguma outra implementação do MPI se ele mesmo compilar a aplicação (desenvolvida por ele ou a partir de seu código fonte aberto), pois a aplicação precisa ser executada pela mesma implementação do MPI que foi utilizada em sua compilação. Para aplicações instaladas no "/usr/local", o usuário não precisa se preocupar com a implementação utilizada, pois a mesma será configurada no script "-vars.sh" da aplicação.

Estas implementações do MPI envolvem um conjunto de de ferramentas e comandos para compilação e execução da aplicação. No momento da execução, ela cria os diversos processos em um ambiente para a troca de mensagens que pode estar espalhado em diferentes máquinas. Esse ambiente utiliza diversas bibliotecas capazes de utilizar e otimizar o uso de diferentes ambientes de rede.

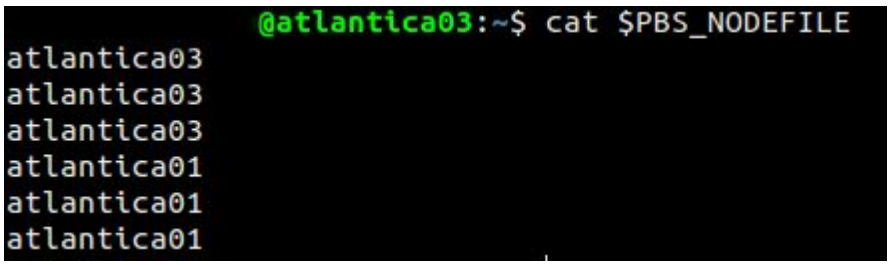
6.1 Variável "PBS_NODEFILE"

Sempre que o usuário faz alguma alocação, uma variável de ambiente é criada. Essa variável denomina-se "PBS_NODEFILE" e nela contém o caminho para um "machinefile" próprio da alocação feita, comumente utilizada em aplicações MPI que veremos a seguir. Esse "machinefile" contém o nome de cada máquina, tantas vezes quanto o número de processadores alocados por cada.

Ao executarmos o comando "export", após feita uma alocação, podemos notar a variável e o caminho que ela possui, como na figura abaixo:

Podemos também executar o comando “cat” juntamente com “$PBS_NODEFILE” para exibir o conteúdo do arquivo que a variável define.

Seguindo esse raciocínio, no exemplo abaixo podemos perceber que alocamos duas máquinas com três processadores cada, pois o nome de cada máquina é repetido três vezes.



6.2 Execução de aplicações com MPI

Para que uma implementação do MPI possa fazer a criação dos processos nas diferentes máquinas, é utilizado um “machinefile” como o contido na variável “PBS_NODEFILE”. No próprio arquivo de exemplo de Batchjob possui um exemplo de execução MPI. Um outro exemplo é o que segue:

```
mpirun -machinefile $PBS_NODEFILE -np 32 program_mpi
```

Para o exemplo acima, alocamos duas máquinas com 16 processadores cada. Então definimos a variável “PBS_NODEFILE” como o machinefile e 32 como o número de processos que o MPI irá criar para a execução do nosso programa “program_mpi”. O mpirun percorrerá o arquivo machinefile criando um processo em cada maquina nele referenciado até chegar ao número de processos definidos no parâmetro “np”. Se o arquivo machinefile não for declarado, todos os processos serão disparados na máquina atual. Se o machinefile tiver um número menor de referências do que o número de processos declarados no “np”, o mpirun percorrerá as referências do arquivo de forma recursiva.

Porém os usuários devem ter em mente que as máquinas do LAD não possuem fisicamente todos os processadores especificados, a maioria das máquinas possuem 2 processadores físicos com X núcleos cada, todas com Hyper-threading.

Hyper-threading é uma tecnologia da Intel que torna cada núcleo físico capaz de executar 2 processos a cada ciclo de clock, assim, o sistema operacional identifica 2 núcleos “virtuais” de processamento para cada núcleo físico. Por exemplo: Em uma máquina com 2 processadores (cada um com 2 núcleos de processamento). Sem Hyper-threading, ao executarmos o comando “htop” veremos 4 núcleos. Porém, caso os processadores tenham a tecnologia da Intel, veremos 8 núcleos (virtuais).

O uso do Hyper-threading não significa que cada core tenha o dobro de desempenho, simplesmente contém algumas contém algumas features que, na maioria das vezes, trazem ganhos no tempo total de processamento. Em algumas aplicações o desempenho pode até piorar quando utilizado todos os núcleos virtuais. Por esse motivo, o indicado é que o usuário faça simulações iguais, mas com diferentes números de processos por máquina. Por exemplo, se uma máquina tem 16 núcleos virtuais (8 físicos), o usuário pode testar a execução de 8 processos por máquina.

Para testar com diferentes números de processos, o usuário poderá criar o seu próprio "machinefile" a partir do "PBS_NODEFILE". Isso é fundamental, pois se o usuário apenas alterar o parâmetro "-np" do MPI, os processos poderão não se espalhar adequadamente entre as máquinas alocadas. Por exemplo, se o usuário reservou 2 máquinas com 16 processadores cada e deseja usar 8 processadores de cada maquina, não adianta apenas mudar o parâmetro "-np" de 32 para 16, pois no "PBS_NODEFILE" as 16 primeiras linhas correspondem a uma única máquina. Desta forma, 16 processos rodariam em uma das máquinas enquanto a outra não executará nenhum. Para resolver esse problema, podemos criar um arquivo, denominado "nodefile" (por exemplo), e remover as referências duplicadas, usando o exemplo abaixo:

```
cat $PBS_NODEFILE | uniq > nodefile
```

Com esse comando, criamos o arquivo "nodefile" com apenas uma referencia de cada máquina alocada. Lembrando que esse comando também pode ser inserido no script BatchJob.

```
@atlantica09:~/exemplos/torque$ cat $PBS_NODEFILE | uniq
atlantica09
atlantica03
```

Assim, podemos especificar quantos processos desejarmos, pois ao chegar ao final do arquivo (em "atlantica03") e ter mais processos para executar, o MPI irá voltar para o inicio do arquivo (em "atlantica09"). Ou seja, os processos serão divididos igualmente entre as máquinas (dependendo do número de processos escolhido e o número de máquinas). Agora só precisamos executar o "mpirun" utilizando o novo arquivo machinefile.

```
mpirun -machinefile nodefile -np 16 program_mpi
```

Para fins de teste e acompanhamento da execução, é possível acessar as máquinas alocadas por ssh (máquinas não alocadas não permitem acesso) para visualizar os processos em execução e a utilização dos processadores. Um comando que pode ser usado para tal visualização é o "htop", para isso basta o usuário acessar a(s) máquina(s) alocada(s) e executar o comando.

7 Permissões dos Arquivos

Todos os arquivos no Linux possuem 3 tipos de permissões associadas a 3 tipos de usuários.

As permissões são de:

- leitura, representada pela letra “r” (read);
- escrita, representada pela letra “w” (write);
- execução, representada pela letra “x” (execute).

Os tipos são:

- O proprietário (usuário) do arquivo;
- Os usuários de um determinado grupo (por padrão, é o mesmo grupo do usuário);
- Outros usuários.

Podemos verificar as permissões dos arquivos através do comando “ls”, com o uso dos parâmetros “l” e “a” (“-la”).

Esse comando listará todos os objetos do local atual, mas permite especificar outro local após o parâmetro “-la”. Essa lista conterá informações sobre o tipo do objeto, permissões, proprietário (usuário) e grupo.

Na imagem acima, a primeira letra de cada arquivo é o tipo de objeto, exemplo: “d” é diretório e “l” é um link simbólico a outro arquivo. Após a primeira letra vêm as permissões do arquivo, com as 3 sequências de 3 letras. Também apresenta o proprietário, o “root” (nesse caso), e o grupo, que também se chama “root”.

A primeira sequência são as permissões do proprietário do arquivo, a segunda sequência é dos usuários do “grupo dono“ do arquivo (não necessariamente é o grupo ao qual o proprietário pertence) e a terceira sequência são as permissões para os outros usuários.

Para fazer alguma alteração nas permissões de um certo arquivo ou diretório, temos o comando “chmod”. O uso dele é bastante simples:

chmod <permissão> <arquivo/diretório>
chmod -R <permissão> <diretório>

O parâmetro “-R” quer dizer recursivo, ou seja, aplicará a alteração da permissão recursivamente para todos os objetos localizados dentro do diretório. Caso o usuário mude a permissão de um diretório sem o parâmetro de recursividade, as permissões afetarão apenas o diretório e não seu conteúdo interno.

A variável “permissão” pode ser definida de várias formas, para esse texto, explicaremos a definição com 3 números, para a permissão do usuário, grupo e outros, respectivamente. O número para cada um é obtido a partir do valor em decimal do binário formado pela sequência de 0 (permissão não concedida) e 1 (permissão concedida) para cada tipo de permissão. Caso no lugar da letra tenha um sinal “-” significa que o objeto não possui aquela permissão.

Permissão	Binário	Decimal
---	000	0
--x	001	1
-w-	010	2
-wx	011	3
r--	100	4
r-x	101	5
rw-	110	6
rwx	111	7

Exemplos

chmod 775 arquivo.pl

No caso acima, no arquivo "arquivo.pl", estamos atribuindo todas as permissões (leitura, escrita e de execução) para o proprietário e para o grupo. Para os outros usuários, atribuindo somente as permissões de leitura e execução.

chmod 777 arquivo.pl

No caso anterior, estamos atribuindo total acesso (leitura, escrita e execução) ao arquivo para todos os tipos de usuários.

chmod -R 750 pasta

No exemplo acima, estamos atribuindo ao diretório "pasta" e recursivamente para todo o conteúdo nele contido, as 3 permissões para o proprietário, somente leitura e execução para o grupo proprietário e nenhuma permissão para os outros usuários.