Sistema de Submissão Simplificada (GO)

Introdução

Avaliando a utilização do sistema e tentando sempre facilitar a utilização dos recursos, foram identificadas algumas facilidades que não estavam presentes nas ferramentas tradicionais de grid de forma que sua ausência prejudicava a adoção do grid por parte dos usuários. Por essa razão, foi desenvolvido um método de submissão simplificado, batizado de go, que oferece as seguintes vantagens:

- Submissão simplificada da simulação
- Acompanhamento do progresso durante a execução
- Salvar e recuperar checkpoints
- Suporte a grandes arquivos de entrada e saída
- Flexibilidade na descrição de arquivos de saída
- Utilização automática de diretórios temporários
- Facilitar depuração de erros

Ainda estão previstas novas melhorias, como a cópia de grandes arquivos em duas etapas, um sistema de acompanhamento de processos via web e um sistema que direcione automaticamente os processos para os recursos mais adequados.

Esse sistema permite a criação de templates para aplicações específicas, facilitando o seu uso. A lista das aplicações suportadas por esse mecanismo está descrita abaixo. Porém esse sistema permite a utilização de qualquer aplicação, incluindo as desenvolvidas ou compiladas pelos próprios usuários. **Recomendamos o uso dessa ferramenta para todos os usuários (básicos e avançados)**

Modos de operação

O sistema GO aceita o envio de processos para o cluster local ou através da estrutura de grid. No primeiro caso, os processos estarão restritos ao cluster central do GridUnesp, mas não é necessário gerar os certificados proxy, usados apenas no envio para o grid.

A operação local é a padrão, e não é necessário fazer nada para usá-la. Também **não** é necessário criar um proxy com o comando "myproxy-logon".

Para enviar o processo para a estrutura de grid, use a opção "-grid". Por exemplo:

```
go caso.go -grid
```

Aplicativos suportados

O sistema permite o envio de um programa desenvolvido e compilado pelo usuário. Porém conta também com modelos simplificados para algumas as aplicações que damos suporte. Esperamos expandir o número de aplicações suportadas.

Nome	Versão	Serial	SMP	MPI	Chekpoint	Progress
[[BEAST]]	1.6.1	S	S	N	N	S
[[Clustalw]]	1.82	N	N	S	N	S
[[Crystal]]	14.1	S	N	?	?	S
[[Dacapo]]	r474	S	N	S	S	S
[[Dalton]]	2013.4	S	N	S	?	S
[[Dirac]]	12.3	S	N	S	?	S
[[Espresso]]	3.2.0	S	N	S	?	S
[[Gamess]]	2013	S	N	S	?	ni
[[Gamit]]	10.5	S	N	N	?	ni
[[Garli]]	2.0	S	N	S	N	S
[[Gaussian]]	09	S	S	S	S	S
[[GPAW]]	2012 07	S	N	S	S	S
[[Gromacs]]	4.5.4	S	S	S	T	S
[[Gromacs]]	4.6.5	S	S	S	T	S
[[IM]]	12_17_2009	S	N	N	S	S
[[IMa2]]	8.26.11	S	N	N	N	S
[[LAMMPS]]	18_10_2011	S	ni	S	?	S
[[Mathematica]]	9	S	N	S	?	S

[[MOPAC]]	13.294	S	?	N	N	S
[[MrBayes]]	3.1.2	S	N	S	?	ni
[[Mugsy]]	1.2.1	S	N	N	N	S
[[NAMD]]	2.9	ni	N	S	?	ni
[[OpenFOAM]]	2.1	S	N	S	?	?
[[Orca]]	r1730	S	N	?	S	S
[[Python]]*	2.7	S	S	S	-	-
[[POY]]	4.1.2	S	N	S	?	ni
[[QuantumEspresso]]	4.3	S	N	S	?	S
[[R]]	3.1	S	N	N	-	-
[[RAxML]]	7.3.0	S	S	S	N	S
[[Siesta]]	3.2.4	S	N	N	N	S

Legenda:

S	recurso implementado
N	recurso não suportado pelo software
T	recurso em fase de testes
ni	recurso disponibilizado pelo software, mas não implementado no go
?	informação não disponível

[*] Módulos adicionais de Python

- SciPy
- NumPy
- matplotlib
- mpi4py
- networkx
- emcee
- decorators
- xlrd/xlwd
- libsbml

Aplicativos compilados pelo próprio usuário

Com o go é possível também enviar arquivos desenvolvidos ou compilados pelo próprio usuário.

Mais detalhes estão [[SubmissaoSimplificadaCustom|aqui]].

Utilização

- Crie um arquivo de parâmetros (ex: caso.go) como explicado abaixo
- Crie um certificado de proxy (myproxy-logon) com uma duração maior que a do seu job
- Execute "go caso.go"

Isso cria e envia o seu job para o grid. Caso não queira enviar imediatamente, utilize a opção "-nosubmit". O arquivo de submissão será criado com o nome de "caso.go.submit".

Caso você tenha especificado checkpoints ou arquivos de progresso, eles serão automaticamente copiados para esse diretório.

IMPORTANTE: NÃO USE ESPAÇOS NOS NOMES DOS ARQUIVOS

Opções

Opções gerais

- CASE - Nome do caso. Define o nome de arquivos de entrada, log e saida. **Padrão**: (nome do arquivo de parâmetros, sem a extensão)
- HOSTS - Número de hosts (processos) para a execução. **Padrão**: 1
- QUEUE - Fila do processo. Opções: short (até 24 horas), medium (até 7 dias), long (30 dias) **Padrão**: medium (24 horas)
- LARGE_FILES - Suporte a arquivos de simulação maiores que 8GB. **Padrão**: false
- NOTIFY - Escolhe quais notificação serão enviadas por email. "NOTIFY=" desabilita o envio.

Opções avançadas (geralmente não são necessárias!)

- EXEC - executável
- TRANSFER_EXEC - se transfere o executável
- ARGS - argumentos
- INPUT - arquivos de entrada, **separados por espaço**
- OUTPUT - arquivos de saída, **separados por espaço**
- PROGRESS - arquivo que exibe o progresso da simulação
- WAIT_PROGRESS - tempo de atualização do progresso, em segundos. **Padrão**: 15 min.
- CHECKPOINT - arquivo de checkpoint
- WAIT_CHECKPOINT - tempo de atualização do checkpoint, em segundos. **Padrão**: 3 horas.
- HOSTS - Número de hosts (processos) para a execução. **Padrão**: 1
- PPH - Process per host. Opção para programas SMP. **Padrão**: 1. Máximo: 8
- MPI - Se o executável deve ser chamado em ambiente MPI (True ou False).
- SHARED_FS - Utiliza um sistema de arquivos compartilhado (útil para aplicações MPI)
- CORE_MEM - Memória RAM por core, em GB. **Padrão**: 2. Máximo: 16

Exemplo passo-a-passo

Segue um exemplo completo utilizando o software "Orca". As opções específicas de cada aplicação estão a seguir.

Prepare os arquivos de entrada

Prepare os arquivos de entrada da aplicação. Nesse caso usaremos nome_do_caso.inp (arquivo do orca).

IMPORTANTE: NÃO USE ESPAÇOS NOS NOMES

caso_de_teste.inp

```
! HF SVP

* xyz 0 1
  C 0.0 0.0 0.0
  O 0.0 0.0 1.13
*
```

Prepare o arquivo .go

caso_de_teste.go

```
APP=orca
QUEUE=short
```

Crie um proxy - Apenas necessário se usado com a opção -grid

```
myproxy-logon
```

Submeta a simulação

Local (padrão e recomendado)

```
go caso_de_teste.go
```

Grid (alternativa)

```
go caso_de_teste.go -grid
```

Verificando o status

Você pode verificar o status com o comando: squeue -u $USER (local) ou condor_q $USER -globus (grid).

Cancelando a execução

Para interromper uma simulação, utilize o comando: scancel _jobid_ (local) ou condor_rm jobid (grid) , onde *jobid* é o identificador do job, disponível no arquivo de log ou no comando de status.