Executando múltiplas tarefas seriais em um único job:

1) Executando N vezes o mesmo binário (serial), sem passar parâmetros

Essa é a forma mais simples. Se a aplicação não necessita utilizar parâmetros de entrada e é executada diretamente, é necessário apenas configurar o ambiente do SLURM corretamente.

O exemplo abaixo executará 48 vezes a mesma aplicação, sendo 24 em cada nó.

Script de submissão exemplo_1_simples.srm

```
#!/bin/bash
#SBATCH --nodes=2                 #Numero de Nós
#SBATCH --ntasks-per-node=24      #Numero de tarefas por Nó
#SBATCH --ntasks=48               #Numero total de tarefas MPI
#SBATCH -p treinamento            #Fila (partition) a ser utilizada
#SBATCH -J exemplo_1              #Nome job

#Exibe os nós alocados para o Job
echo $SLURM_JOB_NODELIST
nodeset -e $SLURM_JOB_NODELIST

cd $SLURM_SUBMIT_DIR

#Configura o executavel
EXEC=/scratch/app/NPB3.3.1-MZ/bin/cg.C.x

#exibe informações sobre o executável
/usr/bin/ldd $EXEC

srun --resv-ports --nodes 1 --ntasks=24 $EXEC > $PWD/log1.txt &
srun --resv-ports --nodes 1 --ntasks=24 $EXEC > $PWD/log2.txt &
wait
<==========>
```

Quando o “srun” é executado utilizando um binário serial, ele disparará a execução do número de vezes que foi definido pelo “--ntasks”.
No exemplo acima, executará 24 vezes o benchmark cg do NPB.

2) Executando N vezes o mesmo binário (serial), sem passar parâmetros através de um script

No exemplo anterior o job colocará a saída das execuções em um arquivo para cada “srun”, agregando assim a saída de 24 execuções em um único arquivo de log.

Uma forma de contornar isso é utilizar um script intermediário.

No exemplo abaixo, a linha do srun executará um script, que por sua vez executará as 24 instâncias da aplicação, direcionando a saída para arquivos diferentes.

Script de submissão exemplo_2_simples_com_script.srm

```
#!/bin/bash
#SBATCH --nodes=2                 #Numero de Nós
#SBATCH --ntasks-per-node=24      #Numero de tarefas por Nó
#SBATCH --ntasks=48               #Numero total de tarefas MPI
#SBATCH -p treinamento            #Fila (partition) a ser utilizada
#SBATCH -J exemplo_2              #Nome job

#Exibe os nós alocados para o Job
echo $SLURM_JOB_NODELIST
nodeset -e $SLURM_JOB_NODELIST

cd $SLURM_SUBMIT_DIR
```

```
#Configura o script intermediario
SCRIPT=${PWD}/script_intermediario_2.sh

srun --resv-ports --nodes 1 --ntasks=1 $SCRIPT run_node1 &
srun --resv-ports --nodes 1 --ntasks=1 $SCRIPT run_node2 &
wait
```

<==>

script_intermediario_2.sh

```
#!/bin/bash
EXEC=/scratch/app/NPB3.3.1-MZ/bin/cg.C.x

RUN=${1}

#exibe informações sobre o executável
/usr/bin/ldd $EXEC

$EXEC > ${PWD}/${RUN}_1_log.txt &
$EXEC > ${PWD}/${RUN}_2_log.txt &
$EXEC > ${PWD}/${RUN}_3_log.txt &
$EXEC > ${PWD}/${RUN}_4_log.txt &
$EXEC > ${PWD}/${RUN}_5_log.txt &
$EXEC > ${PWD}/${RUN}_6_log.txt &
$EXEC > ${PWD}/${RUN}_7_log.txt &
$EXEC > ${PWD}/${RUN}_8_log.txt &
$EXEC > ${PWD}/${RUN}_9_log.txt &
$EXEC > ${PWD}/${RUN}_10_log.txt &
$EXEC > ${PWD}/${RUN}_11_log.txt &
$EXEC > ${PWD}/${RUN}_12_log.txt &
$EXEC > ${PWD}/${RUN}_13_log.txt &
$EXEC > ${PWD}/${RUN}_14_log.txt &
$EXEC > ${PWD}/${RUN}_15_log.txt &
$EXEC > ${PWD}/${RUN}_16_log.txt &
$EXEC > ${PWD}/${RUN}_17_log.txt &
$EXEC > ${PWD}/${RUN}_18_log.txt &
$EXEC > ${PWD}/${RUN}_19_log.txt &
$EXEC > ${PWD}/${RUN}_20_log.txt &
$EXEC > ${PWD}/${RUN}_21_log.txt &
$EXEC > ${PWD}/${RUN}_22_log.txt &
$EXEC > ${PWD}/${RUN}_23_log.txt &
$EXEC > ${PWD}/${RUN}_24_log.txt &
wait
```

<=======>

3) Executando N vezes o mesmo binário (serial), passando parâmetros através de um script

A utilização de parâmetros por parte do executável vai depender de cada caso.
Se ele lê um arquivo de entrada, é possível criar um arquivo para cada execução independente.

Script de submissão exemplo_3_com_parametros_em_arquivo.srm

```
#!/bin/bash
#SBATCH --nodes=2                 #Numero de Nós
#SBATCH --ntasks-per-node=24      #Numero de tarefas por Nó
#SBATCH --ntasks=48               #Numero total de tarefas MPI
#SBATCH -p treinamento            #Fila (partition) a ser utilizada
#SBATCH -J exemplo_3              #Nome job

#Exibe os nós alocados para o Job
echo $SLURM_JOB_NODELIST
nodeset -e $SLURM_JOB_NODELIST
cd $SLURM_SUBMIT_DIR

#Configura o script intermediario
SCRIPT=${PWD}/script_intermediario_3.sh
```

```
srun --resv-ports --nodes 1 --ntasks=1 $SCRIPT run_node1 &
srun --resv-ports --nodes 1 --ntasks=1 $SCRIPT run_node2 &
wait
```

<==>

script_intermediario_3.sh

```
#!/bin/bash
EXEC=/scratch/app/NPB3.3.1-MZ/bin/cg.C.x
RUN=${1}
#exibe informações sobre o executável
/usr/bin/ldd $EXEC
$EXEC -f ${PWD}/${RUN}_1_entrada > ${PWD}/${RUN}_1_log.txt &
$EXEC -f ${PWD}/${RUN}_2_entrada > ${PWD}/${RUN}_2_log.txt &
$EXEC -f ${PWD}/${RUN}_3_entrada > ${PWD}/${RUN}_3_log.txt &
$EXEC -f ${PWD}/${RUN}_4_entrada > ${PWD}/${RUN}_4_log.txt &
$EXEC -f ${PWD}/${RUN}_5_entrada > ${PWD}/${RUN}_5_log.txt &
$EXEC -f ${PWD}/${RUN}_6_entrada > ${PWD}/${RUN}_6_log.txt &
$EXEC -f ${PWD}/${RUN}_7_entrada > ${PWD}/${RUN}_7_log.txt &
$EXEC -f ${PWD}/${RUN}_8_entrada > ${PWD}/${RUN}_8_log.txt &
$EXEC -f ${PWD}/${RUN}_9_entrada > ${PWD}/${RUN}_9_log.txt &
$EXEC -f ${PWD}/${RUN}_10_entrada > ${PWD}/${RUN}_10_log.txt &
$EXEC -f ${PWD}/${RUN}_11_entrada > ${PWD}/${RUN}_10_log.txt &
$EXEC -f ${PWD}/${RUN}_12_entrada > ${PWD}/${RUN}_11_log.txt &
$EXEC -f ${PWD}/${RUN}_13_entrada > ${PWD}/${RUN}_12_log.txt &
$EXEC -f ${PWD}/${RUN}_14_entrada > ${PWD}/${RUN}_13_log.txt &
$EXEC -f ${PWD}/${RUN}_15_entrada > ${PWD}/${RUN}_14_log.txt &
$EXEC -f ${PWD}/${RUN}_16_entrada > ${PWD}/${RUN}_15_log.txt &
$EXEC -f ${PWD}/${RUN}_17_entrada > ${PWD}/${RUN}_16_log.txt &
$EXEC -f ${PWD}/${RUN}_18_entrada > ${PWD}/${RUN}_17_log.txt &
$EXEC -f ${PWD}/${RUN}_19_entrada > ${PWD}/${RUN}_18_log.txt &
$EXEC -f ${PWD}/${RUN}_20_entrada > ${PWD}/${RUN}_19_log.txt &
$EXEC -f ${PWD}/${RUN}_21_entrada > ${PWD}/${RUN}_20_log.txt &
$EXEC -f ${PWD}/${RUN}_22_entrada > ${PWD}/${RUN}_21_log.txt &
$EXEC -f ${PWD}/${RUN}_23_entrada > ${PWD}/${RUN}_22_log.txt &
$EXEC -f ${PWD}/${RUN}_24_entrada > ${PWD}/${RUN}_23_log.txt &
wait
```

<=======>

Importante observar que:

- Nesse caso, é necessário gerar previamente um arquivo de entrada diferente para cada execução.

3.1) Exemplo passando parâmetros diretamente:

É possível também passar diferentes parâmetros diretamente para o script

Script de submissão exemplo_3.1_com_parametros_direto.srm

```
#!/bin/bash
#SBATCH --nodes=2                 #Numero de Nós
#SBATCH --ntasks-per-node=24      #Numero de tarefas por Nó
#SBATCH --ntasks=48               #Numero total de tarefas MPI
#SBATCH -p treinamento            #Fila (partition) a ser utilizada
#SBATCH -J exemplo_3.1            #Nome job

#Exibe os nós alocados para o Job
echo $SLURM_JOB_NODELIST
nodeset -e $SLURM_JOB_NODELIST

cd $SLURM_SUBMIT_DIR

#Configura o script intermediario
SCRIPT=${PWD}/script_intermediario_3.1.sh

srun --resv-ports --nodes 1 --ntasks=1 $SCRIPT parametro_1 parametro_2 parametro_3 parametro_4
parametro_5 parametro_6 parametro_7 parametro_8 parametro_9 parametro_10 parametro_11
```

```
parametro_12 parametro_13 parametro_14 parametro_15 parametro_16 parametro_17 parametro_18
parametro_19 parametro_20 parametro_21 parametro_22 parametro_23 parametro_24 &

srun --resv-ports --nodes 1 --ntasks=1 $SCRIPT parametro_25 parametro_26 parametro_27
parametro_28 parametro_29 parametro_30 parametro_31 parametro_32 parametro_33 parametro_34
parametro_35 parametro_36 parametro_37 parametro_38 parametro_39 parametro_40 parametro_41
parametro_42 parametro_43 parametro_44 parametro_45 parametro_46 parametro_47 parametro_48 &
wait
```

<==>

script_intermediario_3.1.sh

```
#!/bin/bash
EXEC=/scratch/app/NPB3.3.1-MZ/bin/cg.C.x

RUN=${1}

#exibe informações sobre o executável
/usr/bin/ldd $EXEC

$EXEC ${1} &
$EXEC ${2} &
$EXEC ${3} &
$EXEC ${4} &
$EXEC ${5} &
$EXEC ${6} &
$EXEC ${7} &
$EXEC ${8} &
$EXEC ${9} &
$EXEC ${10} &
$EXEC ${11} &
$EXEC ${12} &
$EXEC ${13} &
$EXEC ${14} &
$EXEC ${15} &
$EXEC ${16} &
$EXEC ${17} &
$EXEC ${18} &
$EXEC ${19} &
$EXEC ${20} &
$EXEC ${21} &
$EXEC ${22} &
$EXEC ${23} &
$EXEC ${24} &

wait
```

<=======>

Importante observar que:

- Nesse caso, a linha do “srun” passará 24 parâmetros diferentes para o script intermediário.
- O script intermediário pega os 24 diferentes parâmetros de entrada e os distribui para as 24 execuções.

4) Uma outra forma seria criar diferentes scripts intermediários completos, cada um contando todos os parâmetros necessários, sem a necessidade de passar valores do script de submissão

Script de submissão exemplo_4_com_scripts_completos.srm

```
#!/bin/bash
#SBATCH --nodes=2                 #Numero de Nós
#SBATCH --ntasks-per-node=24      #Numero de tarefas por Nó
#SBATCH --ntasks=48               #Numero total de tarefas MPI
#SBATCH -p treinamento            #Fila (partition) a ser utilizada
#SBATCH -J exemplo_4            #Nome job
```

```
#Exibe os nós alocados para o Job
echo $SLURM_JOB_NODELIST
nodeset -e $SLURM_JOB_NODELIST

cd $SLURM_SUBMIT_DIR

srun --resv-ports --nodes 1 --ntasks=1 ${PWD}/script_intermediario_4_completo.1.sh &
srun --resv-ports --nodes 1 --ntasks=1 ${PWD}/script_intermediario_4_completo.2.sh &
wait
```

<==>

script_intermediario_4_completo.1.sh

```
#!/bin/bash
EXEC=/scratch/app/NPB3.3.1-MZ/bin/cg.C.x

RUN=${1}

#exibe informações sobre o executável
/usr/bin/ldd $EXEC

$EXEC parametro_1.1 parametro_1.2 parametro_1.3  ... &
$EXEC parametro_2.1 parametro_2.2 parametro_2.3  ... &
$EXEC parametro_3.1 parametro_3.2 parametro_3.3  ... &
$EXEC parametro_4.1 parametro_4.2 parametro_4.3  ... &
$EXEC parametro_5.1 parametro_5.2 parametro_5.3  ... &
$EXEC parametro_6.1 parametro_6.2 parametro_6.3  ... &
$EXEC parametro_7.1 parametro_7.2 parametro_7.3  ... &
$EXEC parametro_8.1 parametro_8.2 parametro_8.3  ... &
$EXEC parametro_9.1 parametro_9.2 parametro_9.3  ... &
$EXEC parametro_10.1 parametro_10.2 parametro_10.3  ... &
$EXEC parametro_11.1 parametro_11.2 parametro_11.3  ... &
$EXEC parametro_12.1 parametro_12.2 parametro_12.3  ... &
$EXEC parametro_13.1 parametro_13.2 parametro_13.3  ... &
$EXEC parametro_14.1 parametro_14.2 parametro_14.3  ... &
$EXEC parametro_15.1 parametro_15.2 parametro_15.3  ... &
$EXEC parametro_16.1 parametro_16.2 parametro_16.3  ... &
$EXEC parametro_17.1 parametro_17.2 parametro_17.3  ... &
$EXEC parametro_18.1 parametro_18.2 parametro_18.3  ... &
$EXEC parametro_19.1 parametro_19.2 parametro_19.3  ... &
$EXEC parametro_20.1 parametro_20.2 parametro_20.3  ... &
$EXEC parametro_21.1 parametro_21.2 parametro_21.3  ... &
$EXEC parametro_22.1 parametro_22.2 parametro_22.3  ... &
$EXEC parametro_23.1 parametro_23.2 parametro_23.3  ... &
$EXEC parametro_24.1 parametro_24.2 parametro_24.3  ... &
wait
```

<=======>

script_intermediario_4_completo.1.sh

```
#!/bin/bash
EXEC=/scratch/app/NPB3.3.1-MZ/bin/cg.C.x

RUN=${1}

#exibe informações sobre o executável
/usr/bin/ldd $EXEC

$EXEC parametro_25.4 parametro_25.5 parametro_25.6  ... &
$EXEC parametro_26.4 parametro_26.5 parametro_26.6  ... &
$EXEC parametro_27.4 parametro_27.5 parametro_27.6  ... &
$EXEC parametro_28.4 parametro_28.5 parametro_28.6  ... &
$EXEC parametro_29.4 parametro_29.5 parametro_29.6  ... &
$EXEC parametro_30.4 parametro_30.5 parametro_30.6  ... &
$EXEC parametro_31.4 parametro_31.5 parametro_31.6  ... &
$EXEC parametro_32.4 parametro_32.5 parametro_32.6  ... &
$EXEC parametro_33.4 parametro_33.5 parametro_33.6  ... &
```

```
$EXEC parametro_34.4 parametro_34.5 parametro_34.6  ... &
$EXEC parametro_35.4 parametro_35.5 parametro_35.6  ... &
$EXEC parametro_36.4 parametro_36.5 parametro_36.6  ... &
$EXEC parametro_37.4 parametro_37.5 parametro_37.6  ... &
$EXEC parametro_38.4 parametro_38.5 parametro_38.6  ... &
$EXEC parametro_39.4 parametro_39.5 parametro_39.6  ... &
$EXEC parametro_40.4 parametro_40.5 parametro_40.6  ... &
$EXEC parametro_41.4 parametro_41.5 parametro_41.6  ... &
$EXEC parametro_42.4 parametro_42.5 parametro_42.6  ... &
$EXEC parametro_43.4 parametro_43.5 parametro_43.6  ... &
$EXEC parametro_44.4 parametro_44.5 parametro_44.6  ... &
$EXEC parametro_45.4 parametro_45.5 parametro_45.6  ... &
$EXEC parametro_46.4 parametro_46.5 parametro_46.6  ... &
$EXEC parametro_47.4 parametro_47.5 parametro_47.6  ... &
$EXEC parametro_48.4 parametro_48.5 parametro_48.6  ... &

wait


<=======>
```

Importante observar que:

- Nesse caso é necessário um script específico para cada linha do “srun”
- Esse script deverá conter todos os parâmetros necessários para a execução da aplicação, representados por “*parametro_X.Y*”

Alterar informações do job, após a submissão:

Executar o comando scontrol para exibir as informações do job:

```
[aluno1001@sdumont12 ~]$ scontrol show jobid 123456 -dd

JobId= 123456 JobName=exemplo_1
   UserId=aluno1001(32363) GroupId=treinamento(2058)
   Priority=1458 Nice=0 Account=treinamento QOS=normal
   JobState=PENDING Reason=Priority Dependency=(null)
   Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
   DerivedExitCode=0:0
   RunTime=00:00:00 TimeLimit=10:00:00 TimeMin=N/A
   SubmitTime=2017-07-28T17:04:40 EligibleTime=2017-07-28T17:04:40
   StartTime=2017-07-29T20:55:47 EndTime=Unknown
   PreemptTime=None SuspendTime=None SecsPreSuspend=0
   Partition=treinamento_phi AllocNode:Sid=sdumont12:10197
   ReqNodeList=(null) ExcNodeList=(null)
   NodeList=(null)
   NumNodes=14-14 NumCPUs=336 CPUs/Task=1 ReqB:S:C:T=0:0:*:*
   Socks/Node=* NtasksPerN:B:S:C=24:0:*:* CoreSpec=*
   MinCPUsNode=24 MinMemoryNode=0 MinTmpDiskNode=0
   Features=(null) Gres=(null) Reservation=(null)
   Shared=OK Contiguous=0 Licenses=(null) Network=(null)
   Command=/scratch/treinamento/aluno1001/exemplo_1_simples.srm
   WorkDir=/scratch/treinamento/aluno1001
   StdErr=/scratch/treinamento/aluno1001/slurm-123456 .out
   StdIn=/dev/null
   StdOut=/scratch/treinamento/aluno1001/slurm-123456 .out
   BatchScript=
```

Para alterar a fila (partição) do job:

```
[aluno1001@sdumont12 ~]$ scontrol update JobId=123456 Partition=treinamento_gpu
```

Exibir novamente as informações do job:

```
[aluno1001@sdumont12 ~]$ scontrol show jobid 123456 -dd

JobId=123456 JobName=exemplo_1
   UserId=aluno1001(32363) GroupId=treinamento(2058)
   Priority=1468 Nice=0 Account=treinamento QOS=normal
   JobState=RUNNING Reason=None Dependency=(null)
   Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
   DerivedExitCode=0:0
   RunTime=00:00:08 TimeLimit=10:00:00 TimeMin=N/A
   SubmitTime=2017-07-28T17:04:40 EligibleTime=2017-07-28T17:04:40
   StartTime=2017-07-28T17:14:41 EndTime=2017-07-29T03:14:41
   PreemptTime=None SuspendTime=None SecsPreSuspend=0
   Partition=treinamento_gpu AllocNode:Sid=sdumont12:10197
   ReqNodeList=(null) ExcNodeList=(null)
   NodeList=sdumont[5000-5013]
   BatchHost=sdumont5000
   NumNodes=14 NumCPUs=336 CPUs/Task=1 ReqB:S:C:T=0:0:*:*
   Socks/Node=* NtasksPerN:B:S:C=24:0:*:* CoreSpec=*
     Nodes=sdumont[5000-5013] CPU_IDs=0-23 Mem=64000
   MinCPUsNode=24 MinMemoryNode=62.50G MinTmpDiskNode=0
   Features=(null) Gres=(null) Reservation=(null)
   Shared=OK Contiguous=0 Licenses=(null) Network=(null)
   Command=/scratch/treinamento/aluno1001/exemplo_1_simples.srm
   WorkDir=/scratch/treinamento/aluno1001/
   StdErr=/scratch/treinamento/aluno1001/slurm-123456.out
   StdIn=/dev/null
   StdOut=/scratch/treinamento/aluno1001/slurm-123456.out
   BatchScript=
```

Através do comando “scontrol update” é possível alterar vários parâmetros do job, como por exemplo:

- O momento em que o job entrará em execução:

[aluno1001@sdumont12 ~]$ scontrol update JobId=123456 StartTime=now+30days
... later ...
[aluno1001@sdumont12 ~]$ scontrol update JobId=123456 StartTime=now