Simple mode d'emploi de "Linux Utility for Resource Management" (SLURM) v2.6

Mise en garde

Ce texte se veut minimaliste expliquant que les ordres les plus élémentaires pour pouvoir utiliser SLURM (version 2.6).

Informations générales et conventions

1. le répertoire par default est celui du lancement des ordres SLURM. Pour travailler dans un autre répertoire il faut utiliser "cd" de linux dans votre script ;
2. ce mode d'emploi se limite aux ordres suivants : srun, sbatch, squeue, scancel, smap, sview ;
3. "#BATCH" est la directive SLURM correspondant à "#$" pour SGE ;
4. "%j" dans un nom de fichier sera remplacé à l'exécution par le numéro identifiant le job donné par SLURM. Utile pour les array jobs.

Lexique, comparaison SLURM <=> SGE

(voir les MAN pages de SLURM voir : http ://slurm.schedmd.com/man_index.html)
(voir la ROSETTA comparant SLURM et SGE : http ://slurm.schedmd.com/rosetta.pdf)

SLURM	SGE	signification
srun	qsub	soumission de travaux parallèles en ligne de commande
salloc	qsub	allocation de ressources en ligne de commande
sbatch	qsub	soumission de travaux par script de soumission
smap	qmon	information graphique sur les travaux/jobs et des partitions/queues
sview	qmon	information graphique sur les travaux/jobs et des partitions/queues
sinfo		donne l'état des noeuds et partitions en mode non graphique
squeue	queue	information non graphique sur les travaux/jobs et des partitions/queues
sstat	qstat	information non graphique sur l'état des jobs et étapes de calcul
scancel	qdel	arrêter/tuer un job
scontrol	qmon	modifier les paramètres de slurm
partition	queue	queue de lancement de programmes avec de paramètres définis par le gestionnaire

Options/directives essentielles dans les scripts/ordres de sbatch ou la commande srun

(voir : http ://slurm.schedmd.com/sbatch.html, http ://slurm.schedmd.com/sbatch.html ou srun.html)
"#SBATCH " suivi d'un blanc identifie la directive SLURM suivi d'une option expliquée ci-après

option	signification
%j	numéro du job donné au moment de l'exécution (variable)
%A et %a	numéros d'un array job et de sa soutache à l'execution
-J ou --job-name="your job-name"	nom du travail/job
-a ou --array="1-100"	définit le nombre de jobs dans l'array job
-o ou --output="job-name_%j.stdout"	sortie/output de messages système du travail/job
-e ou --error="job-name_%j.stderr"	sortie/output de erreurs du job. Manquant, erreurs dans -o
--partition="partition/queue"	la queue de soumission du job, travail
--mail-type="type"	"Type" de mail peut être BEGIN, END, ALL, FAIL, REQUEUE
--mail-user="user@site"	recevoir un courriel suivant les instructions du --mail-type
--mem="nombre de M/G/T"	mémoire en mega/giga/terabits par noeud
--mem-per-cpu="nombre de M/G/T"	mémoire en mega/giga/terabits par coeur/task
-N "nombre" ou --nodes="nombre"	nombre de noeuds
-n "nombre" ou --ntasks="nombre"	nombre de tache (tasks/cores)
--ntasks-per-node="nombre"	nombre de tache (tasks/cores) pour un noeud
--time=dd-hh:mm:ss	temps de calcul, aussi --time=dd-hh, --time=hh:mm:ss, --time=hh
--workdir=	répertoire de travail
--export=[ALL|NONE|variables]	copie l'environnement sur le noeud de calcul
--gres=gpu:2	"gres" correspond aux ressources consommable. Ici il s'agit du gpu
--cpus-per-task="1"	nombre de CPUs par tache
--cores-per-socket="6"	restreint l'accès aux noeuds $\leqslant$6 coeurs par processeur
--constraint="nom du noeud"	le travail/job s'exécutera sur un noeud particulier
--share/exclusive	les noeuds "sont/ne sont pas" réservés par le travail/job

Examples de sbatch

- ***job séquentiel simple***

```
#!/bin/sh
#SBATCH --partition=xxx
#SBATCH --time=0 :20 :00
#SBATCH --output=j%j.stout
#SBATCH --cpus-per-task=1
#SBATCH --mem=4096M
# #SBATCH --nodes=1
#SBATCH --ntasks=2
echo "----------------------------------------------------------"
echo "hostname = $(hostname)"
echo "SLURM_JOB_NAME = $SLURM_JOB_NAME"
echo "SLURM_SUBMIT_DIR = $SLURM_SUBMIT_DIR"
echo "SLURM_JOBID = $SLURM_JOBID"
echo "SLURM_JOB_ID = $SLURM_JOB_ID"
echo "SLURM_NODELIST = $SLURM_NODELIST"
echo "SLURM_JOB_NODELIST = $SLURM_JOB_NODELIST"
echo "SLURM_TASKS_PER_NODE = $SLURM_TASKS_PER_NODE"
echo "SLURM_JOB_CPUS_PER_NODE = $SLURM_JOB_CPUS_PER_NODE"
echo "SLURM_TOPOLOGY_ADDR_PATTERN= $SLURM_TOPOLOGY_ADDR_PATTERN"
echo "SLURM_TOPOLOGY_ADDR = $SLURM_TOPOLOGY_ADDR"
echo "SLURM_CPUS_ON_NODE = $SLURM_CPUS_ON_NODE"
echo "SLURM_NNODES = $SLURM_NNODES"
echo "SLURM_JOB_NUM_NODES = $SLURM_JOB_NUM_NODES"
echo "SLURMD_NODENAME = $SLURMD_NODENAME"
echo "SLURM_NTASKS = $SLURM_NTASKS"
echo "SLURM_NPROCS = $SLURM_NPROCS"
echo "SLURM_MEM_PER_NODE = $SLURM_MEM_PER_NODE"
echo "SLURM_PRIO_PROCESS = $SLURM_PRIO_PROCESS"
echo "SLURM_ARRAY_TASK_ID = $SLURM_ARRAY_TASK_ID$SLURM"
echo "----------------------------------------------------------"
#srun --ntasks=2 –gres=gpu :1 -l echo "CUDA_VISIBLE_DEVICES=$CUDA_VISIBLE_DEVICES"
# srun --gres=gpu :1 -l env
# srun -n 1 --gres=gpu :1 -l /opt/software/nvidia4/sdk/C/bin/linux/release/deviceQuery –noprompt
# srun -n 1 --gres=gpu :1 -l /opt/software/nvidia4/sdk/C/bin/linux/release/deviceQuery –noprompt
srun –ntasks=2 --gres=gpu :1 -l /opt/software/nvidia.debian/sdk-4.2.9/C/bin/linux/release/nbody -fp64
-benchmark -n=300000
```

Note si l'on ne passe pas de paramètres, on ne doit pas nécessairement/absolument utiliser "srun"

- ***array job***

```
#!/bin/bash
#SBATCH -J job_array_max_times_max_array
#SBATCH –partition=debug
#SBATCH -t 00 :05 :00
max_times=$(expr 2)
## ================================================================
## the second value of –array must be identical with max_array on the following line
#SBATCH –array=1-3
max_array=$(expr 3)
echo "max_times = $max_times" "max_array = $max_array"
## ================================================================
# do not use #SBATCH –mail-type and #SBATCH –mail-user.
# Otherwise you will receive one mail per array sub_job
# below %A==job_id, %a==sub_job_id
# if one miss them the information of the array job appears in wave_array_job.stout but it is incomplete
#SBATCH -o job_array_max_times_max_array_%A_%a.stout # if –error or -e is absent, includes
also the errors
#SBATCH –mem=10000M
#SBATCH –mem-per-cpu=4000M
# for array jobs the number of simultaneus jobs is given by –array above
#SBATCH –nodes=1 # -N 1
#SBATCH –ntasks-per-node=2 # -n 2
#SBATCH –ntasks=2
```

```
## ================================================================
# this array job performs simultaneously "max_array" tasks, "max_times" times in a do loop
# the final job using –dependency=singleton is executed only if all the preceding jobs are finished
## ================================================================

echo "————————————————————————————————————————"
echo "hostname = $(hostname)"
echo "SLURM_JOB_NAME = $SLURM_JOB_NAME"
echo "SLURM_SUBMIT_DIR = $SLURM_SUBMIT_DIR"
echo "SLURM_JOBID = $SLURM_JOBID"
echo "SLURM_JOB_ID = $SLURM_JOB_ID"
echo "SLURM_NODELIST = $SLURM_NODELIST"
echo "SLURM_JOB_NODELIST = $SLURM_JOB_NODELIST"
echo "SLURM_TASKS_PER_NODE = $SLURM_TASKS_PER_NODE"
echo "SLURM_JOB_CPUS_PER_NODE = $SLURM_JOB_CPUS_PER_NODE"
echo "SLURM_TOPOLOGY_ADDR_PATTERN = $SLURM_TOPOLOGY_ADDR_PATTERN"
echo "SLURM_TOPOLOGY_ADDR = $SLURM_TOPOLOGY_ADDR"
echo "SLURM_CPUS_ON_NODE = $SLURM_CPUS_ON_NODE"
echo "SLURM_NNODES = $SLURM_NNODES"
echo "SLURM_JOB_NUM_NODES = $SLURM_JOB_NUM_NODES"
echo "SLURMD_NODENAME = $SLURMD_NODENAME"
echo "SLURM_NTASKS = $SLURM_NTASKS"
echo "SLURM_NPROCS = $SLURM_NPROCS"
echo "SLURM_MEM_PER_NODE = $SLURM_MEM_PER_NODE"
echo "SLURM_PRIO_PROCESS = $SLURM_PRIO_PROCESS"
echo "SLURM_ARRAY_TASK_ID = $SLURM_ARRAY_TASK_ID"
echo "————————————————————————————————————————"

# USER Commands
# move to the directory where the data files locate
# cd data_dir

# run the program
## ================================================================
# do loop of "max_array" simultaneous jobs
# the do loop is executed "max_times" times
## ================================================================
i_array=$(expr $SLURM_ARRAY_TASK_ID)
for i_times in ‘seq 1 $max_times‘; do
echo "i_times = $i_times" "i_array = $i_array"
echo "SLURM_ARRAY_TASK_ID = $SLURM_ARRAY_TASK_ID"
srun wave_dvr_potentials.x < wave_dvr_potentials_slab_low_wf_chulkov_Ag_001_$i_array.rd >
print_wave_dvr_potentials_slab_low_wf_chulkov_Ag_001_$i_array.out
i_array=$(expr $SLURM_ARRAY_TASK_ID + $max_array)
done
# execute the sweep program
srun –dependency=singleton ./comp_integration.x < comp_integration_test.rd > print_comp_integration_test.ou
## ================================================================

echo "end of the array job"
# end of the user commands
```

Note : i) ci-dessus %A est le numéro de l'array job, %a est le numéro du sub-job ; ii) pour l'instant je n'ai pas trouvé comment restreindre le nombre de sub-jobs simultanément lancés.