RESSOURCES DE CALCUL MCIA

MCIA

Khodor.Khadra@math.u-bordeaux.fr

Du 27 juin au 01 juillet 2022

Sessions antérieures : 2017, 2018, 2019, 2020

RESSOURCES DE CALCUL MCIA

1. Qu'est-ce que MCIA ?
2. Quelques définitions
3. Cluster CURTA
4. Les modules
5. Compilation, débogage
6. Soumission des jobs : batch ou interactif
7. Les points de reprise
8. Stockage des données
9. Post-traitement
10. Contact en cas de problème

1.

Qu'est ce que MCIA ?

RESSOURCES DE CALCUL MCIA

Mésocentre de Calcul Intensif Aquitain

Site Web :
http://www.mcia.fr
https://redmine.mcia.fr/projects/mcia

Objectif : mettre à disposition des laboratoires de recherche et des entreprises d'Aquitaine un plateau technique de qualité et un lieu d'échange d'expériences et de compétences dans le domaine du calcul intensif

Guide de l'utilisateur :
https://redmine.mcia.fr/projects/cluster-curta/wiki/Guide_de_l'utilisateur

RESSOURCES DE CALCUL MCIA

Connexion au site par authentification CAS :

https://redmine.mcia.fr/projects/cluster-curta/wiki/Guide_de_l'utilisateur#Redmine-le-site-collaboratif-du-M%C3%A9socentre

En particulier, pour poster un ticket (demande particulière ou signaler un problème)

2.
Quelques définitions

RESSOURCES DE CALCUL MCIA

Une plateforme de calcul comprend un ou plusieurs clusters différents

Un cluster de calcul est un ensemble de N nœuds de calcul identiques qui communiquent entre eux par l'intermédiaire d'un réseau pour le calcul parallèle

Un nœud de calcul = une machine de calcul qui comprend :

- **sa mémoire vive et son disque dur local**
- **plusieurs processeurs à plusieurs cœurs de calcul chacun**

RESSOURCES DE CALCUL MCIA

Un processus de calcul est défini par :

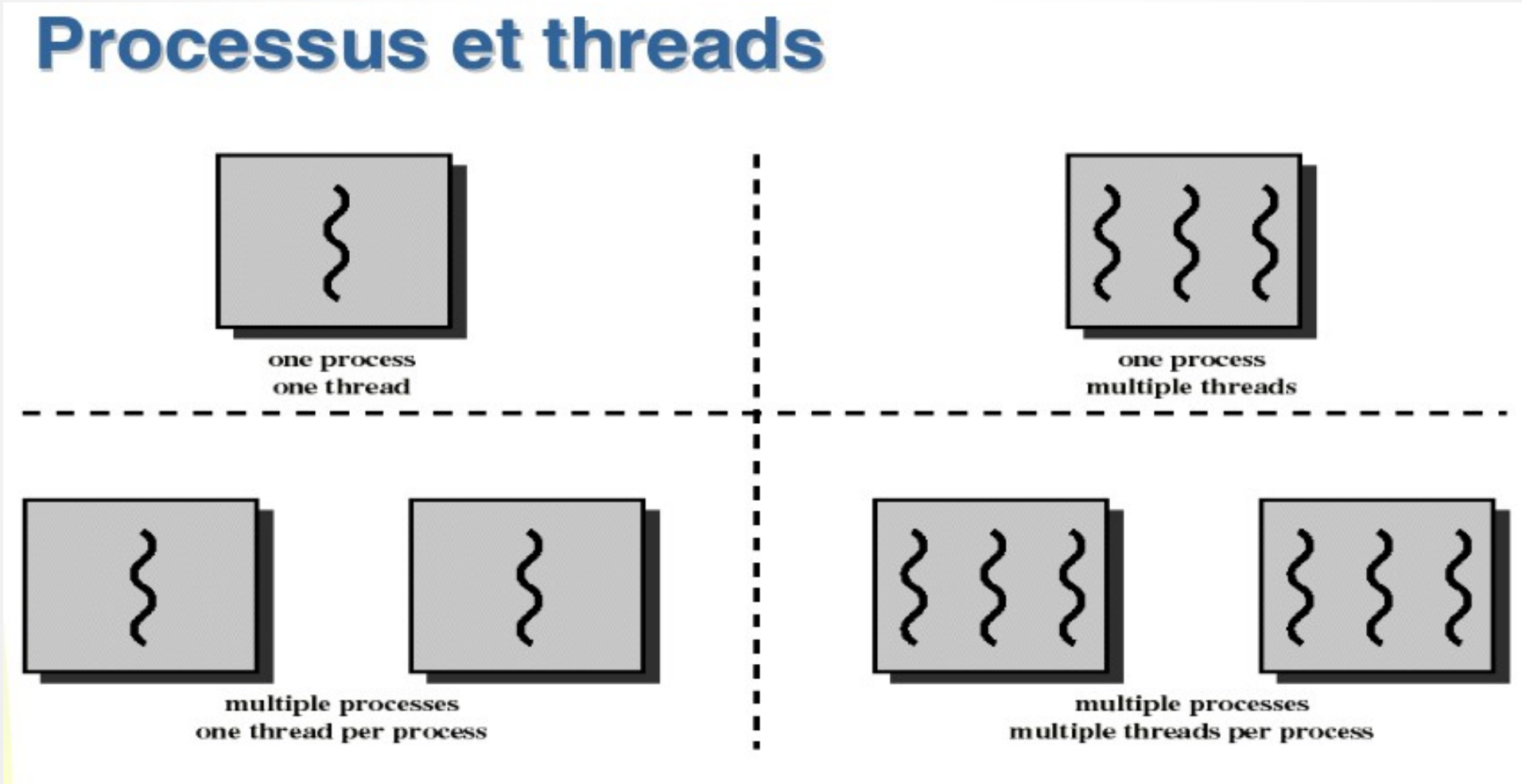
- **un ensemble d'instructions à exécuter un programme**
- **un espace mémoire pour les données de travail**

Un thread (tâche) est une unité d'exécution rattachée à un processus, chargée d'exécuter une partie du processus.

Un processus est vu comme étant un ensemble de ressources (espace d'adressage, fichiers, périphériques...) que ses threads (flots de contrôle ou processus légers) partagent.

RESSOURCES DE CALCUL MCIA

Processus et threads



RESSOURCES DE CALCUL MCIA

Un job de calcul = un ensemble de processus liés à l'exécution d'un code calcul

Pour un calcul séquentiel, un processus est rattaché à un seul cœur de calcul

Pour un calcul parallèle, P processus tournent sur C cœurs de calcul, on peut choisir P > C mais il est préférable de choisir P=C, c'est à dire un seul processus par cœur de calcul

3.

Cluster CURTA

RESSOURCES DE CALCUL MCIA

MCIA contient un cluster de calcul curta

https://redmine.mcia.fr/projects/cluster-curta/wiki

4.

Les modules

RESSOURCES DE CALCUL MCIA

Un logiciel de calcul a besoin pour être exécuté d'un certain nombre de modules (compilateurs, bibliothèques scientifiques, ...) pré-installés

Les modules sont accessibles sur l'ensemble des nœuds de calcul

Ils sont classés en différentes catégories (compilateurs, bibliothèques de calcul parallèle, bibliothèques d'algèbre linéaire, …) et il existe différentes versions pour chaque catégorie

Il peut y avoir des dépendances (pré-requis) dans le chargement d'un module : par exemple un module M2 ne peut être chargé avant que le module M1 ne le soit.

RESSOURCES DE CALCUL MCIA

Commandes sur les modules

Pour avoir un aperçu des modules existants il suffit de taper la commande

module available ou module av

Pour charger un module :

module load nom_module ou module add nom_module

Cette commande positionne automatiquement un certain nombre de variables d'environnement nécessaires à l'utilisation de bibliothèques

Pour connaître la liste des modules qui sont chargés dans votre environnement :

module liste

Pour supprimer un module de votre environnement :

module remove nom_module ou module rm nom_module

Pour supprimer tous les modules de votre environnement :

module purge

RESSOURCES DE CALCUL MCIA

Lorsque vous chargez les mêmes modules de façon régulière, plutôt que de les charger à chaque fois manuellement dans une fenêtre terminal, il est possible de les charger une fois pour toute dans le fichier .bashrc qui se trouve dans votre ($HOME). Ainsi, à chaque connexion sur Curta, ces modules seront chargés automatiquement

Lorsqu'au moins un module est chargé dans le fichier .bashrc, en tapant dans une fenêtre terminal respectivement les commandes
module initadd nom_module ou module initrm nom_module
le module nom_module se rajoute ou se supprime de la liste des modules directement dans le fichier .bashrc

Quand on ne précise pas la version d'un module, c'est la version stable qui est chargée (taper la commande module liste pour vérifier le numéro de version)

RESSOURCES DE CALCUL MCIA

Chaque utilisateur de Curta a la possibilité d'installer sa propre bibliothèque et de créer le module associé dans la partition /gpfs/softs/contrib/modulefiles

Il suffit qu'il fasse la demande pour accéder à cette partition

Il en fait bénéficier l'ensemble de la communauté

Il a la charge de la maintenance du module

5.

Compilation, débogage, exécution

Compilation

Pour le compilateur GNU, on n'a pas besoin de charger le module, et on utilise respectivement pour le Fortran et C les compilateurs gfortran et gcc

Pour le compilateurs intel, on peut par exemple charger le module :

module add compiler/intel/numero_de_version

et on utilise respectivement pour le Fortran et C les compilateurs ifort et icc

Pour un code parallèle, on peut par exemple charger le module :

module add mpi/intel/numero_de_version

et on utilise respectivement pour le Fortran, C et C++ les compilateurs

mpif90 mpicc mpicxx

Débogage

Si le code nécessite un débogage, utiliser l'option -g de compilation.

Débogage basique : « print écran » dans le code

Outils de débogage :

- **Débogueur GNU**
  - **pas de chargement de module**
  - **commande : gdb**
- **DDT**
  - **très efficace en mode parallèle**
  - **interface graphique**
  - **chargement du module module add tools/ddt/numero_de_version**
  - **commande : ddt**

Une fois le processus de débogage terminé, inhiber l'option -g car elle ralentit les temps d 'exécution. On utilise souvent l'option de base d'optimisation -O

Exécution

Pour un code séquentiel :
./nom_executable

Pour un code parallèle MPI à np processus :
mpirun -n np ./nom_executable

6.
Soumission de jobs : batch ou interactif

RESSOURCES DE CALCUL MCIA

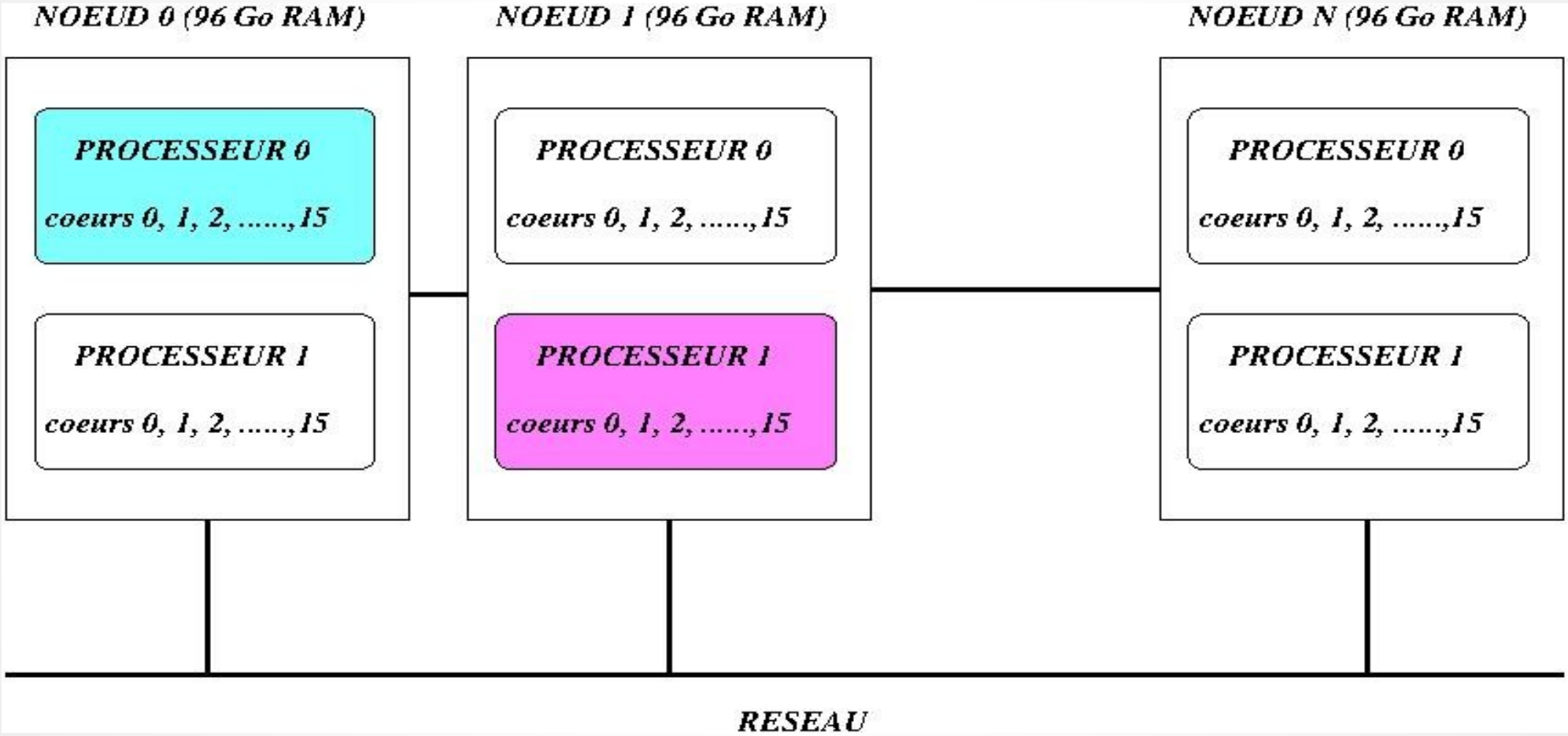
Le gestionnaire de jobs du cluster Curta est SLURM
https://redmine.mcia.fr/projects/cluster-curta/wiki/Slurm

Allocation des ressources pour chaque job de calcul.
Pour un job donné, réservation de N nœuds de calcul et de P processus de calcul par nœud. Pour optimiser les calculs, on choisira P=C coeurs réservés par nœud.

Système d'ordonnancement de jobs très puissant en fonction des ressources informatiques et des temps de calcul demandés

RESSOURCES DE CALCUL MCIA

Exemple de réservation de 2 nœuds et 16 processus (sur 16 coeurs) par nœud :



Les nœuds et coeurs sont allouées à l'utilisateur jusqu'à la fin de l'exécution de son job et un autre job ne peut pas venir se greffer sur les coeurs réservés d'un nœud. Les coeurs ne sont pas forcément alloués sur le même processeur.

RESSOURCES DE CALCUL MCIA

2 modes de soumission des jobs :

- **mode batch : pas de connexion directe sur les nœuds, le job est placé automatiquement dans une partition (file), il est en mode run (R) ou queue (Q) selon la disponibilité des nœuds et coeurs**

- **mode interactif : connexion directe sur un des nœuds afin d'exécuter le code sur les N nœuds**

Travail en mode batch

Création d'un fichier de batch (script shell)

On nomme le fichier de batch comme on le souhaite, c'est un fichier texte de quelques lignes. Il contient des commentaires de texte, des instructions #SBATCH et des commandes shell UNIX standard.

On y précise :

- **le nom du job**
- **une estimation du temps réel maximum d'exécution, si le job a fini plus tôt il s'arrête avant ce temps indiqué**
- **le nombre de nœuds et le nombre de processus par nœud**
- **les modules utilisés s'ils ne sont déjà chargés dans le fichier .bashrc**
- **une suite de commandes (positionnement dans le répertoire de travail, compilation, exécution, …)**

Le fichier de batch

Exemple de fichier batch :

```
#SBATCH –job-name=nom_job
#SBATCH -o nom_job.out
#SBATCH -e nom_job.err
#SBATCH --time=hh:mm:ss   # estimation du temps de calcul
#SBATCH  -N  nombre_de_noeuds # nombre de nœuds a reserver
#SBATCH  --tasks-per-node   nombre_de_processus_par_noeud
module add nom_compilateur
module add nom_bibliothèque
cd /chemin_repertoire_de_travail
echo $SLURM_JOB_NODELIST          # numero des nœuds sur lesquels le job tourne
time ./nom_executable                                    # pour un code séquentiel
time mpirun -n  nombre_total_processus ./nom_executable  # pour un code parallèle MPI
```

Toute instruction #SBATCH est commentée avec ##SBATCH

Tout commentaire de commande UNIX ou de texte doit être précédé du symbole # et ne doit pas comporter d'accent

Quelques commandes

Pour exécuter un batch : sbatch nom_fichier_batch
Le job est placé dans une partition d'un cluster en fonction des ressources et des temps de calcul demandés

Pour connaître l'état global des jobs : squeue -a ou squeue -l
Pour connaître l'état de ses propres jobs : squeue -l | grep user_id
Ces commandes permettent de contrôler l'état d'un job qui tourne à tout moment : les nœuds sur lesquels il tourne et le temps écoulé

Pour connaître l'ensemble des partitions : sinfo -l

A l'exécution d'un fichier de batch , le job est rattaché à un numéro
Pour supprimer un job en cours : scancel numerojob

Fichiers de sorties batch

Deux fichiers de sortie sont générés à la fin du job dans le répertoire où a été exécuté le fichier de batch :

- **un fichier nom_job.Onumerojob qui contient la sortie standard « print ecran ». Attention ce fichier n'est généré qu'à la fin de l'exécution, pour contrôler ses résultats au cours du calcul, penser à générer son propre fichier « print »**

- **un fichier nom_job.enumerojob qui contient les temps de calcul à la fin du job (si la commande time a été indiquée au moment de l'exécution du code) ainsi que les éventuels messages d'erreur (si le code de calcul ne s'est pas déroulé comme prévu jusqu'à la fin)**

Travail en mode interactif

Réservation des noeuds et coeurs en mode interactif

La réservation utilise le gestionnaire de batch SLURM, donc on retrouve les mêmes options qu'en mode batch

Par exemple pour réserver un nœud de calcul avec P processus par noeud :
srun -X --time=hh:mm:ss -N 1 --tasks-per-node P --pty /bin/bash -i

- **L'option -X permet d'obtenir le déport graphique (elle n'est pas obligatoire)**
- **hh, mm, ss : nombre d'heures, minutes et secondes du temps maximum d'exécution du code**
- **Des nœuds vous ont été spécialement réservés pour la formation, rajouter pour y accéder l'option --reservation=formation-02**

Lorsque les ressources demandées ne sont pas disponibles, cette commande répond avec un message d'attente

Les commandes SLURM précédemment en mode batch sont aussi valides en mode interactif

Travail en mode interactif

Réservation des noeuds et coeurs en mode interactif

La réservation utilise le gestionnaire de batch SLURM, donc on retrouve les mêmes options qu'en mode batch

Pour réserver plusieurs nœuds n de calcul avec p processus par nœud :

salloc --time=01:00:00 -N n --tasks-per-node P

Si ces ressources sont disponibles, on voit alors affichée l'information suivante :
salloc: Granted job allocation numero_job
ainsi que la liste des nœuds réservés

La commande srun hostname fournit aussi la liste des nœuds réservés

On peut alors se connecter directement sur un des n nœuds réservés via la commande ssh [-X] nom_noeud, charger les modules, compiler et exécuter à partir de ce nœud le job sur les n nœuds

Estimation du temps d'un job

Estimer le temps de calcul sur un petit jeu de données, ce qui peut permettre d'extrapoler ainsi le « temps max » pour un plus grand jeu de données. Prévoir environ + 20% par rapport à cette estimation

Si vous avez estimé un temps trop court par rapport à la réalité, le job s'arrête à la fin du temps demandé. Si vous avez estimé un temps trop long par rapport à la réalité, le job se termine en temps réel.

Ne pas donner un temps de calcul très élevé si le code ne le nécessite pas. Vous risquez de vous bloquer dans les priorités des files d'attente des jobs.

RESSOURCES DE CALCUL MCIA

Estimation de la mémoire vive (RAM)

Il n'est pas indispensable de préciser la mémoire vive dans la soumission d'un job mais cela est recommandé afin d'avoir un contrôle de la taille des ressources allouées dans votre code. La mémoire à estimer est la somme des mémoires requises par chaque processus d'un job sur l'ensemble des nœuds de calcul où le job va tourner.

Règles de bonne pratique :

- **sachant qu'un nœud de calcul a 32 coeurs et 96 Go de RAM, l'idéal est de faire tourner un job avec un ensemble de processus dont chacun est rattaché à un coeur de calcul et nécessite moins de 3 Go de RAM**
- **allouer 1 cœur pour chaque tranche de 3 Go de RAM, même si votre programme n'utilise pas tous les cœurs.**

Si vous avez estimez une mémoire insuffisante, le job s'arrête immédiatement

On peut estimer la mémoire en faisant tourner le code sur un jeu de données de taille raisonnable de façon interactive, repérer via la commande top dans la colonne RES la mémoire allouée, puis extrapoler si possible de « façon linéaire » la mémoire sur des tailles de données plus grandes. Prévoir environ + 20% par rapport à votre estimation.

Noeuds exclusifs

Par défaut les nœuds réservés ne sont pas exclusifs, c'est à dire plusieurs jobs peuvent se partager le même nœud et donc sa RAM

Un nœud de calcul devient exclusif quand on réserve tous ses coeurs, en l'occurence ici 32, même s'ils ne sont pas tous utilisés.

Utile de réserver des nœuds exclusifs pour des applications qui nécessitent beaucoup de mémoire vive.

Il peut arriver de réserver un ou plusieurs nœuds exclusifs (donc la totalité des coeurs par noeud) parce qu'on a besoin de la totalité de la RAM de tous les nœuds alors que le job ne va pas forcément tourner sur tous les coeurs des nœuds. Dans ces cas, il n'est pas nécessaire de spécifier la mémoire pendant la soumission du job, par défaut cela prendra toute la mémoire des nœuds.

Ne rendre les nœuds exclusifs qu'en cas de réel besoin car les autres coeurs d'un même nœud peuvent servir pour d'autres jobs

Choix optimal du nombre de nœuds et de coeurs

Un utilisateur souhaite faire tourner un code parallèle sur un nombre total de cœurs CT

Question : comment répartir les CT cœurs sur un ensemble de N nœuds et de C cœurs par nœud, c.a.d comment choisir de façon optimale N et C tels que N*C = CT ?

Si on pose CN le nombre total de coeurs par nœud, le réflexe optimal est de choisir C=CN et donc N = CT/CN

La règle n'est pas aussi simple car elle dépend de :

- **la charge des ressources (les CN cœurs ne sont pas tous disponibles par nœud)**
- **la RAM que nécessite le job, un utilisateur peut être amené à réserver plus de nœuds avec juste quelques cœurs par nœud**

Notion d'affinité

Lorsqu'une partition de coeurs est réservée au sein d'un même nœud, un processus qui tourne sur un coeur du nœud peut se ballader d'un coeur à un autre au sein de ce noeud

Cela peut influer sur les temps de calcul

Pour éviter cela, on a la possibilité de rattacher un processus à un même coeur tout au long du calcul

Par exemple en chargeant les modules compiler/intel/xxx et mpi/openmpi/yyy
on lance un job MPI à np processus avec l'option --bind-to-core :
mpirun --bind-to-core -n np ./nom_executable

6.

Les points de reprise

RESSOURCES DE CALCUL MCIA

INDISPENSABLE

Dans les impressions, en dehors des sorties standard « print », penser à générer dans le code de calcul des fichiers de sortie pour les résultats avec des points de reprise (checkpointing) à intervalles réguliers

Pourquoi ?

- **les plateformes de calcul (locales ou nationales) avec des clusters de centaines voire des milliers de cœurs, pour le calcul parallèle ne permettent pas de faire tourner des jobs de plus de quelques heures**

- **avoir le contrôle des calculs à intervalle de temps régulier et en cas de nécessité arrêter le code en cas de divergence des calculs**

- **en cas d'arrêt brutal des machines, reprendre le calcul à partir du dernier point de reprise et non pas de l'état initial**

8.

Stockage des données

RESSOURCES DE CALCUL MCIA

A la création d'un compte, chaque utilisateur dispose automatiquement de 3 espaces de stockage :

Un espace HOME dédié à la mise en oeuvre des codes, logiciels, leur compilation et leur exécution sur des petits volumes de données afin de tester leur bon comportement. Ce répertoire est sauvegardé
https://redmine.mcia.fr/projects/cluster-curta/wiki/FS_home

Un espace SCRATCH pour les calculs avec de plus gros volumes de données. Ce répertoire n'est pas sauvegardé et les données sont disponibles pendant 3 mois. Taper la commande touch pour la disponibilité des données sur du plus long terme.
https://redmine.mcia.fr/projects/cluster-curta/wiki/FS_scratch

RESSOURCES DE CALCUL MCIA

Transfert de fichiers compte courant ↔ MCIA

Les éventuels transferts de fichiers ou répertoires entre votre compte courant et MCIA se font via la commande scp

•Par exemple, à partir de votre compte courant :

- **transfert compte MCIA → compte MCIA :**
  scp -r user_id@curta.mcia.fr:/home/user_id/xxx /home/.
- **transfert compte courant → compte MCIA :**
- **scp -r /home/yyy user_id@curta.mcia.fr:/home/user_id/.**

9.
Post-traitement

RESSOURCES DE CALCUL MCIA

Comment exploiter graphiquement ses résultats de calcul ?

La visualisation déportée
https://redmine.mcia.fr/projects/cluster-curta/wiki/Visualisation_d%C3%A9port%C3%A9e

Procédure sshfs : visualiser directement à partir de son poste de travail local en montant à distance des données de Curta sans les recopier
https://redmine.mcia.fr/projects/cluster-curta/wiki/Guide_de_l'utilisateur#Acc%C3%A9der-%C3%A0-curta-par-un-montage-sshfs

10.

Contact en cas de problème

RESSOURCES DE CALCUL MCIA

Poster un ticket :

https://redmine.mcia.fr/projects/cluster-curta/wiki/Guide_de_l'utilisateur#Redmine-le-site-collaboratif-du-M%C3%A9socentre

RESSOURCES DE CALCUL MCIA

MERCI POUR VOTRE ATTENTION

Khodor.Khadra@math.u-bordeaux.fr