Une Introduction aux Calculs Haute Performance

Ivan LABAYE équipe nanomatériaux magnétiques et modélisation

IMMM
Le Mans

Plan de l'exposé

- Calculs Haute Performance , Pour quoi Faire ?
- Modèles de calculateurs (HPC)
- Outils de Programmation HP
- Exemples sur Aselkam (UMMTO)
- Conclusions
- => suite en atelier calcul //

Calculs Haute Performance
Pour quoi Faire ?

- Dans de nombreux domaines de la science, les problèmes traités actuellement nécessitent de plus en plus de ressources informatiques (puissance de calcul et stockage)
- Les performances individuelles des processeurs et les capacités mémoire évoluent ..... mais pas assez vite pour traiter toutes ces informations en des temps raisonnables
  - Comment faire les calculs de demain avec les ordinateurs d'aujourd'hui?

Calculs Haute Performance
Pour quoi Faire ?

- Besoins de vitesse de calcul
  - Les fréquences des processeurs ont considérablement augmenté ces 30 dernières années (grand public)
    - (1981 IBM PC 2MHz)
    - (2015 Intel Core i7 4960X 4.0 GHz)
  - Mais ces fréquences plafonnent :
    - Pb de refroidissement / de consommation électrique.....
    - AMD FX-8370 Refroidissement à l'He liquide ..... 8,7 GHz

Calculs Haute Performance
Pour quoi Faire ?

- Besoins de mémoire vive
  - Mémoires des ordinateurs personnels
    - (1981 IBM PC 2MHz / 16ko Ram (ext 256 ko))
    - (2015 Intel Corei7 3GHz / 32 Go 2k€)
  - Mais les fréquences des mémoires suivent moins vite :
    - « 1,6 GHz (800MHz réels) » temps d'accès 60ns
    - Vitesse processeur > vitesse mémoire
      - Les Processeurs possèdent des mémoires cache (L1-L3) rapides, petites et chères.....

Calculs Haute Performance
Pour quoi Faire ?

- Les calculs actuels réclament des vitesses et des tailles mémoire de plus en plus grandes .....

  - Sommes nous coincés par les limites précédentes ????
  - NON ---> Recours au « Supercalculateurs »
  - Utilisation de plusieurs UC sur des machines hybrides (Multicoeurs / Multiprocesseurs / Accélérateurs)
  - Mémoire distribuée sur plusieurs UC (Noeuds)

Calculs Haute Performance
Pour quoi Faire ?

Exemple Concret :

Mise à jour d'un champ dipolaire
d'un système magnétique
(i.e mise à jour d'un tableau)
Après analyse =>
[90-98 % du temps de calcul total du programme !!]

Calculs Haute Performance
Pour quoi Faire ?

```
void   Magnet::VariationChampDemagnetisant(REEL dSx,REEL dSy,REEL dSz)          {
for (int k=0;k<NbSitesT;k++)
     {if (k!=iSpin)
        {REEL dx   = -(S[k].X-S[iSpin].X);
         REEL dy   = -(S[k].Y-S[iSpin].Y);
         REEL dz   = -(S[k].Z-S[iSpin].Z);
         REEL UR    = 1.0/NORM(dx,dy,dz);
         REEL UnSurR3=(UR*UR*UR);
         REEL Fact = 3*(dx*dSx+dy*dSy+dz*dSz)*UnSurR3*UR*UR;
         S[k].HD[0] +=(dx*Fact-dSx*UnSurR3);
         S[k].HD[1] +=(dy*Fact-dSy*UnSurR3);
         S[k].HD[2] +=(dz*Fact-dSz*UnSurR3);
         }
      }//fin du for
} // fin de la fonction VariationChampDemagnetisant
```

Calculs Haute Performance
Pour quoi Faire ?

```
void   Magnet::VariationChampDemagnetisant(REEL dSx,REEL dSy,REEL dSz)          {

for (int k=0;k<NbSitesT;k++)

     {Calcul}//fin du for

} // fin de la fonction VariationChampDemagnetisant
```


```
Processeur 1 :
for (int k=0;k<(NbSitesT/2);k++)
    {Calcul}//fin du for
```

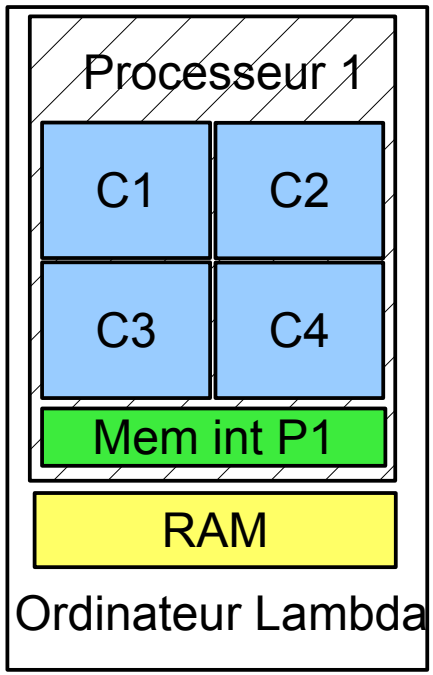
```
Processeur 2 :
for (int k=(NbSitesT/2)+1;k<NbSitesT;k++)
    {Calcul}//fin du for
```

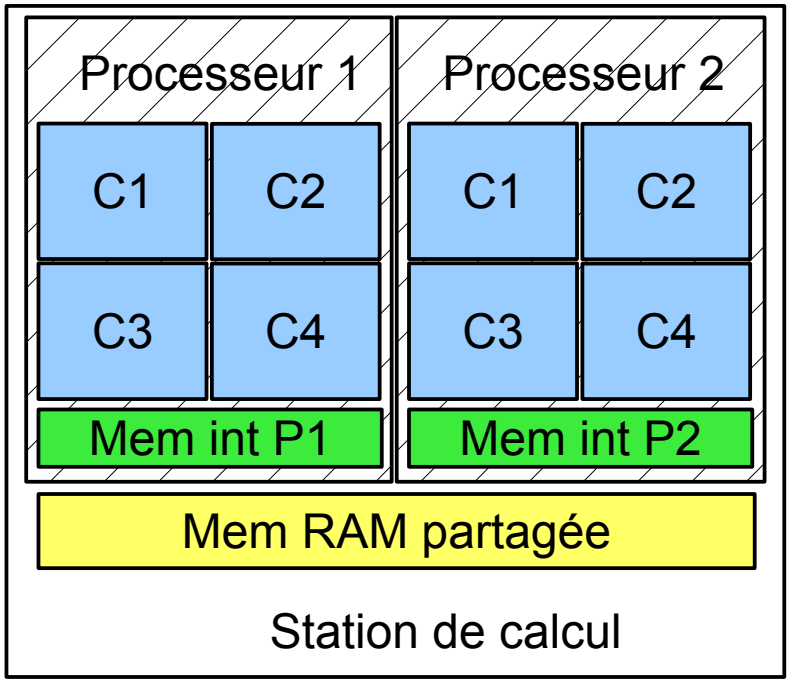
Modèles de calculateurs (HPC)

Sur quelles machines faire ce type de calculs ?

Modèles de calculateurs (HPC)

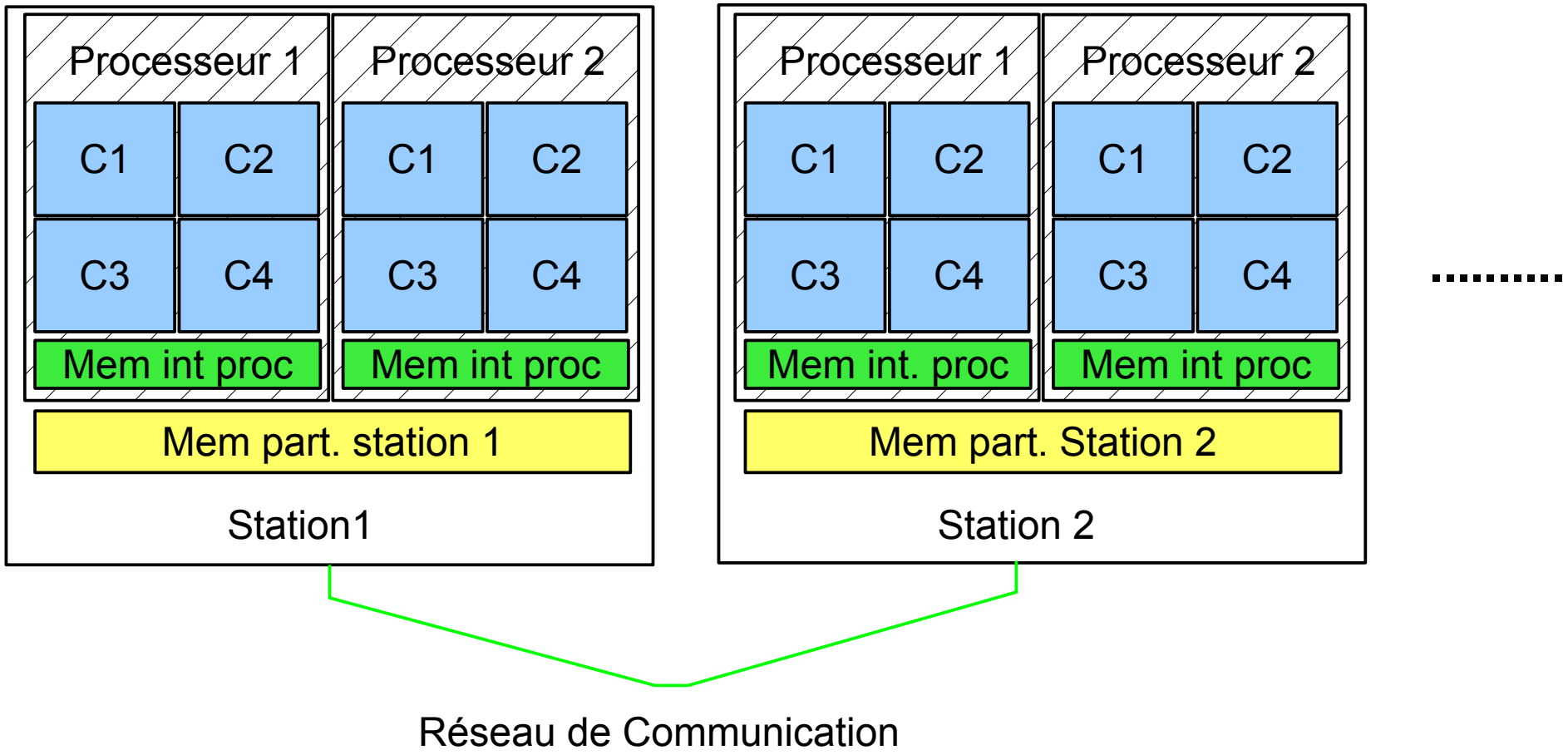
- Architecture d'ordinateur de Bureau / de Station de Calcul





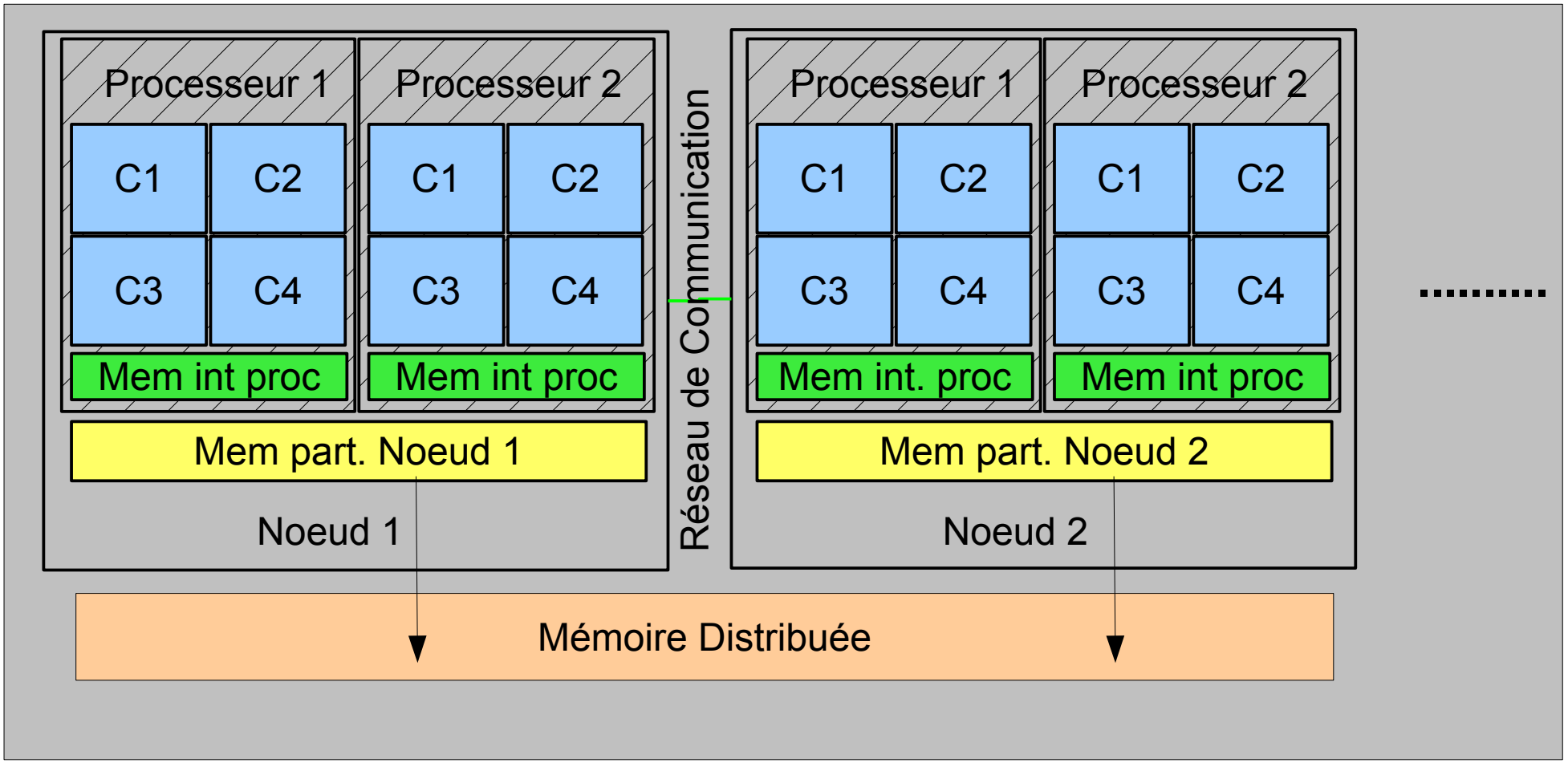
Modèles de calculateurs (HPC)

- Architecture Simplifiée de HPC



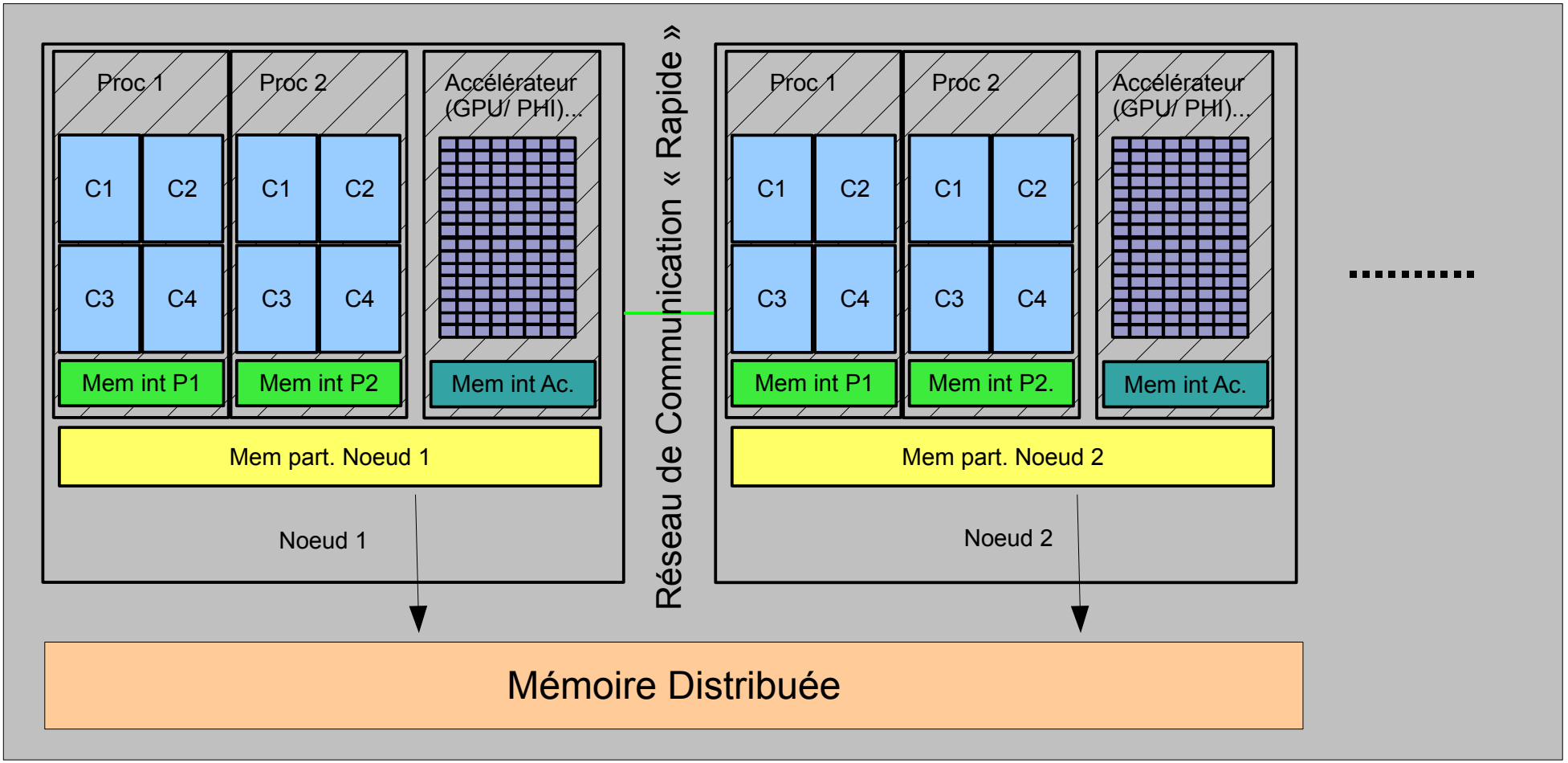
Modèles de calculateurs (HPC)

- Architecture Virtuelle de HPC



Modèles de calculateurs (HPC)

- Architecture Hybride de HPC



Modèles de calculateurs (HPC)

- Puissance des calculateurs Actuels (TOP 500 11/2016)

Nom	PetaFlops	Nb Cores	GPU/ACCE	Pays	Power
Sunway	93	10 649 600	Custom	Chine	15,4MW
Tianhe-2	35	3 120 000	Xeon PHI	Chine	18 MW
Titan	17,6	560 640	Tesla K20x	USA	8,2 MW
Sequoia	17,2	1 572 864	Custom	USA	7,9 MW

http://www.top500.org/

Modèles de Programmes

Avec quels modèles de programmes
faire ce type de calculs ?

Outils de Programmation HP /paradigmes de Prog.

- On peut classer les modèles de programmes en 4 catégories (Taxonomie de Flynn):
  - Single Instruction Single Data

    Cas le plus simple : pas de parallélisme

Outils de Programmation HP /paradigmes de Programmation

- On peut classer les modèles de programmes en 4 catégories (Taxonomie de Flynn):
  - Single Instruction Single Data
  - Single Instruction Multiple Data

    Cas des calculs type vectoriels/ parallélisme classique

Outils de Programmation HP /paradigmes de Programmation

- On peut classer les modèles de programmes en 4 catégories (Taxonomie de Flynn):
  - Single Instruction Single Data
  - Single Instruction Multiple Data
  - Multiple Instruction Single Data

    Très peu fréquent / instructions processeur

Outils de Programmation HP /paradigmes de Programmation

- On peut classer les modèles de programmes en 4 catégories (Taxonomie de Flynn):
  - Single Instruction Single Data
  - Single Instruction Multiple Data
  - Multiple Instruction Single Data
  - Multiple Instruction Multiple Data
    - Cas de calculs sur architecture hétérogène

Outils de Programmation HP /paradigmes de Programmation

Comment écrire des programmes tirant parti de la parallélisation ?

Comment paralléliser ?

- On identifie les parties de code les plus gourmandes
- On étudie la possibilité de parallélisation
- On choisit un modèle de parallélisme pour ce code
- On modifie le code (introduction du parallélisme)
- On teste le comportement du code parallélisé
  - Résultats identiques au code série ?
- On teste la mise à l'échelle (Tcalc(Nproc))

Et si on organisait un congrès de physique et chimie quantique ?

- Après avoir lancé un appel au congrès et tenu à jour la liste des participants (+ pleins d'autres choses ;-))
- On se décide d'appeler un par un les participants pour leur demander s'ils préféreraient allez voir un match de la JSK plutôt que de visiter Athyenni?
- Gros travail , il y a 2500 personnes à contacter , le responsable des sorties est débordé….
  - Il décide de demander à 24 collègues de passer chacun 100 coups de téléphone de la liste complète qu'il a séparé en 25 sous listes….

Et si on organisait un congrès de physique et chimie quantique ?

- Deux méthodes s'offrent au responsable
  - 1) On projète sur le mur d'une salle la liste complète des noms et numéros de téléphone des participants
    - Chaque collègue dans cette salle voit l'ensemble des numéros ainsi que la partie qui lui est attribuée.
    - Quand ils ont toutes les réponses de leur liste, chaque collègue note le nombre de « jsk » et « Athyenni » obtenu
    - À la fin, le responsable peut additionner le nombre de réponses pour chaque option et peut décider de maintenir ou non chaque sortie ….
- On appellera cette méthode **Open**space **M**ulti **P**hone

Et si on organisait un congrès de physique et chimie quantique ?

- Deuxième méthode
  - 2) Le responsable donne à chaque collègue la sous-liste (100 N°) qu'il doit appeler
    - Chaque collègue rentre chez lui et appelle ces 100 numéros, il comptabilise le nombre de choix « jsk » et « Athyenni ».
    - Le responsable appelle un peu plus tard les collègues et leur demande le nombre de réponses pour chaque option. Il fait la somme et peut décider de maintenir ou non chaque sortie ....
- On appellera cette méthode **M**ulti **P**hone **I**ndividuel

Outils de Programmation HP /paradigmes de Programmation

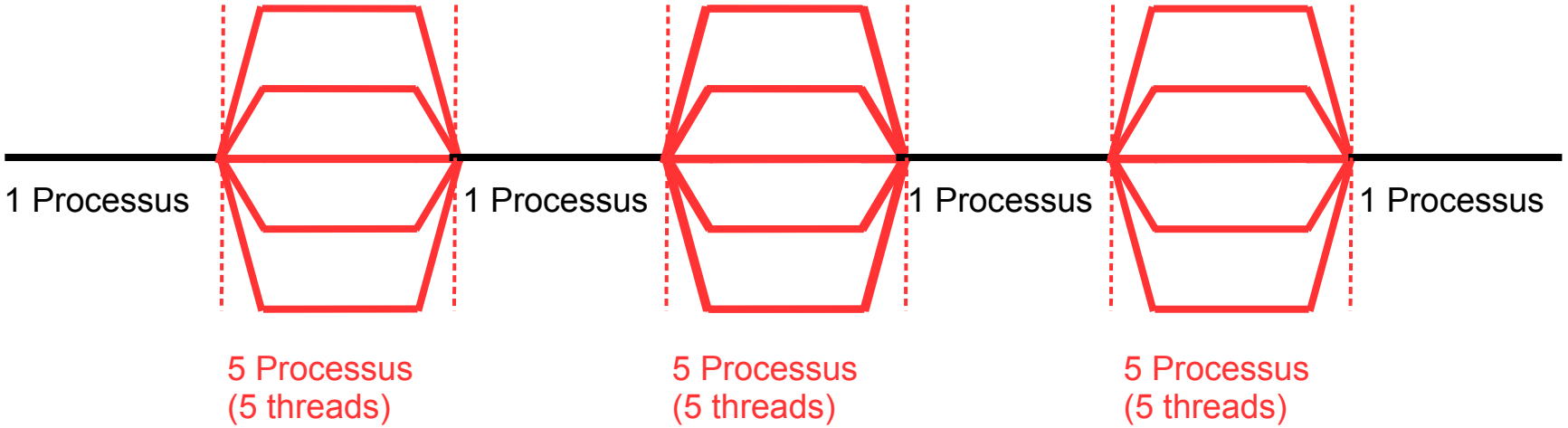
- Open MP (Multi-Processing) (1997-....)
  - Modèle efficace dédié aux architectures à mémoire partagée (accès rapide)
  - + Directives de compilations « simples »
  - + Disponible pour plusieurs langages (C/C++/Fortran)
  - + pas de communications explicites à gérer
  - + modifs de code « souvent » simples (ex des boucles)
  - + Type Multi-Threads (vu comme un process unique)
  - - Inadapté aux architectures à mémoire distribuée
    - Limité à un nb de Processus lié au nb de Coeurs d'un noeud

Outils de Programmation HP /OpenMP Exemple

```
....
#include <omp.h>
int main (int argc, char * const argv[]) {
int Num_Thread;
int Nb_Threads;
....

for(int i=0;i<NT;i++)
     { .....
     VariationChampDemagnetisant(dSx,dSy,dSz);
     }
.....
   return 0;
}
// ------------------------------------------------------------------------ //
void   Magnet::VariationChampDemagnetisant(REEL dSx,REEL dSy,REEL dSz)          {
#pragma omp parallel for
for (int k=0;k<NbSitesT;k++)
     {
     REEL dx   = -(S[k].X-S[iSpin].X);
     REEL dy   = -(S[k].Y-S[iSpin].Y);
     REEL dz   = -(S[k].Z-S[iSpin].Z);
     REEL UR    = 1.0/NORM(dx,dy,dz);
     REEL UnSurR3=(UR*UR*UR);
     REEL Fact = 3*(dx*dSx+dy*dSy+dz*dSz)*UnSurR3*UR*UR;
     S[k].HD[0] +=(dx*Fact-dSx*UnSurR3);
     S[k].HD[1] +=(dy*Fact-dSy*UnSurR3);
     S[k].HD[2] +=(dz*Fact-dSz*UnSurR3);
     }//fin du for k
}
```

Outils de Programmation HP /OpenMP Exemple

- Déroulement du programme :



Outils de Programmation HP /OpenMP Exemple

Compilation :

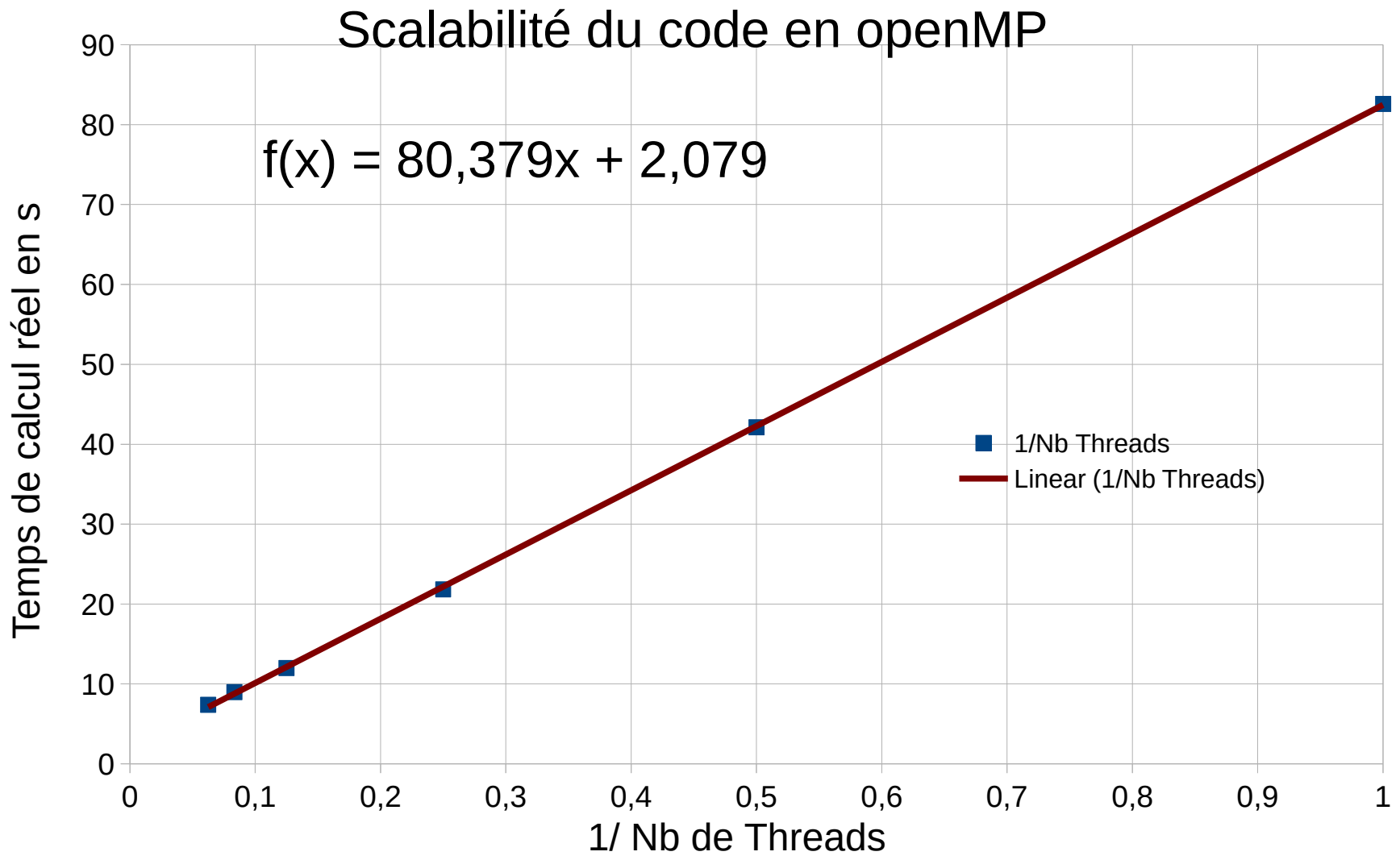
```
$ g++-4.2 -O3 -o TestOMP -fopenmp mainOMP_new.cpp
```

Execution :

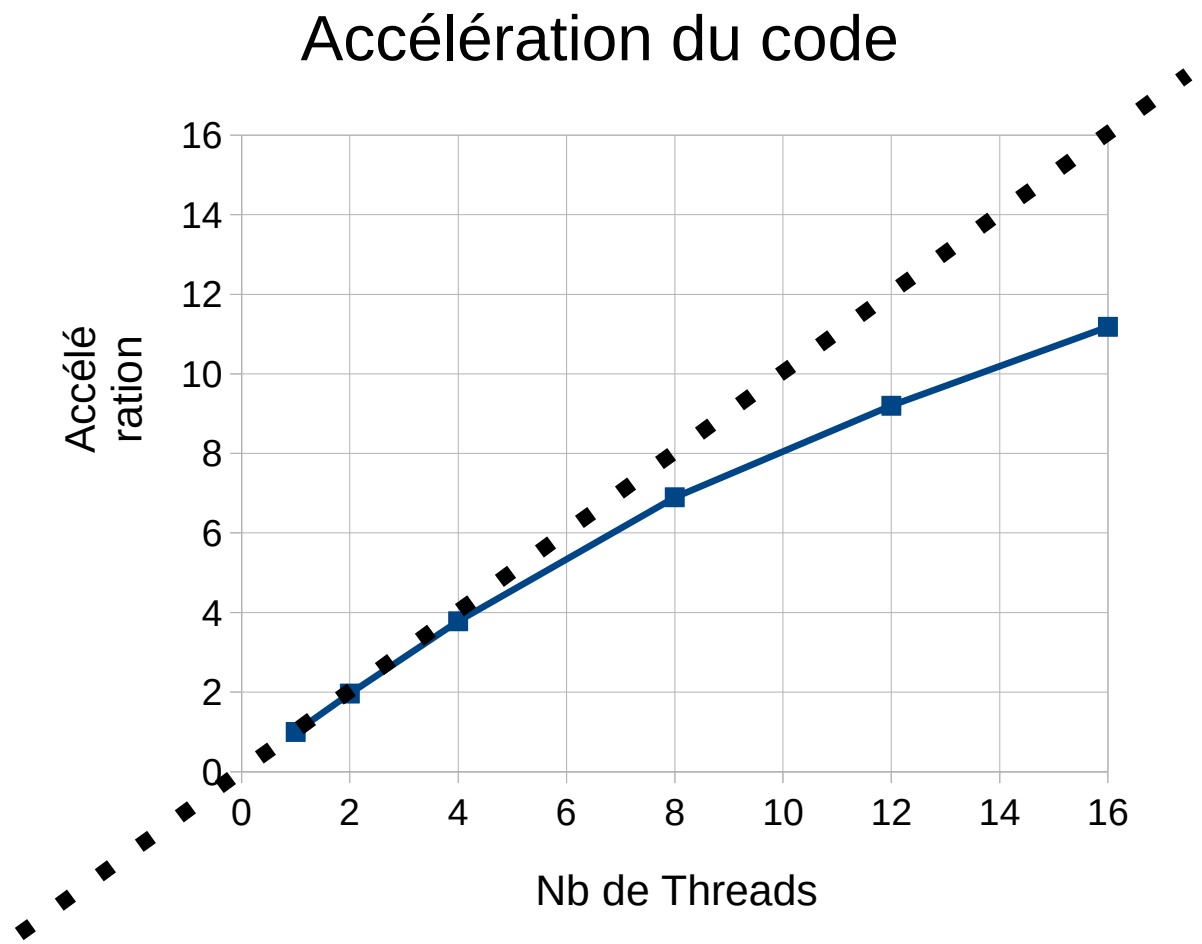
```
$ export OMP_NUM_THREADS=2
$ time ./TestOMP NPARTICULES NBoucles &
```

Npart=30000	NBoucles=200000			
Nb threads	real (s)	user (s)	1/Nb Threads	Acc
1	82,6	82,3	1	1
2	42,12	82,55	0,5	1,96
4	21,84	83,23	0,25	3,78
8	11,98	86,61	0,13	6,89
12	8,98	92,8	0,08	9,2
16	7,39	98,72	0,06	11,18

Outils de Programmation HP /OpenMP Exemple



Outils de Programmation HP /OpenMP Exemple



Outils de Programmation HP /OpenMP Exemple

f(x) = 80,379x + 2,079 Signification :

$$TCalc(Np) = \frac{Tcalc^{//}}{Np} + TCalc^{\text{série}}$$
$$avec\ TCalc^{//} = 80{,}4\ s$$
$$et\ TCalc^{\text{série}} = 2{,}1\ s$$

$$Acc(Np) = \frac{TCalc(1)}{TCalc(Np)}$$

$$Acc(\infty) = \frac{TCalc(1)}{TCalc(\infty)} => \frac{TCalc^{//}}{TCalc^{\text{série}}} + 1$$

$$Acc(\infty) = 39{,}7$$

L'accélération Maximale est limitée par le rapport Tcalc // / Tcalc série

Outils de Programmation HP /MPI

- Message Passing Interface (MPI) (1991-....)
  - Modèle dédié aux architectures à **mémoire distribuée** (vitesse d'accès très dépendante du type de réseau)
  - + Fonctionne sur des machines Massivement //

    La Seule solution sur les machines actuelles (Np grand)
  - + disponible pour plusieurs langages (C/C++/Fortran)
  - - communications /parallèlisation explicites (à gérer)
  - - performances dépendantes du type de pb et du « Hardware »
    - ⚹ modifs de code « souvent » délicates pour être optimales

Outils de Programmation HP /MPI

```
// ------------------------------------------------------------------------ //
void   Magnet::VariationChampDemagnetisant(REEL dSx,REEL dSy,REEL dSz)          {
static int      k;
static int      fink = NbSitesT   ;
static int      TailleTranche= (fink)/(nbProcess);
static int      Reste = (fink)%(nbProcess);
static int      debk = (numProcess)*TailleTranche;
double          R,cond =1;
double          dx,dy,dz;
double          Fact,UnSurR3;
fink = debk+TailleTranche;
if((numProcess+1)==nbProcess) fink+=Reste;
for (k=debk;k<fink;k++)
     {
     Calcul
     }//fin des trois for
}
```

```
int main (int argc,char **argv)
{......
MPI_Init (&argc,  &argv);              /* initialize   MPI system      */
MPI_Comm_rank (MPI_COMM_WORLD, &numProcess);  /* my place in MPI system    */
MPI_Comm_size (MPI_COMM_WORLD, &nbProcess );    /* size of MPI system      */
......
Magnet Mag(&D);
tDeb = time(&dum);
..... Faire les calculs
tFin = time(&dum);
if (numProcess==0) COUT<<"temps en s ="<<tFin-tDeb<<ENDL;
MPI_Barrier(MPI_COMM_WORLD);
MPI_Finalize();
return(0);
}
```

Outils de Programmation HP /MPI

- Message Passing Interface (MPI)
  - Le Programme est lancé en N exemplaires qui ne diffèrent entre eux que par leur numéro (numProcess)
  - Le travail affecté à chaque processeur (Core) doit être explicite (qui fait quoi)
  - Les échanges de données (Mem Distribuée...) se font par des communications explicites point à point(send /receive...) ou collectives (scater/gather)
  - Les lectures/écritures sont généralement faites par un Process unique qui distribue (Com coll) les données aux autres

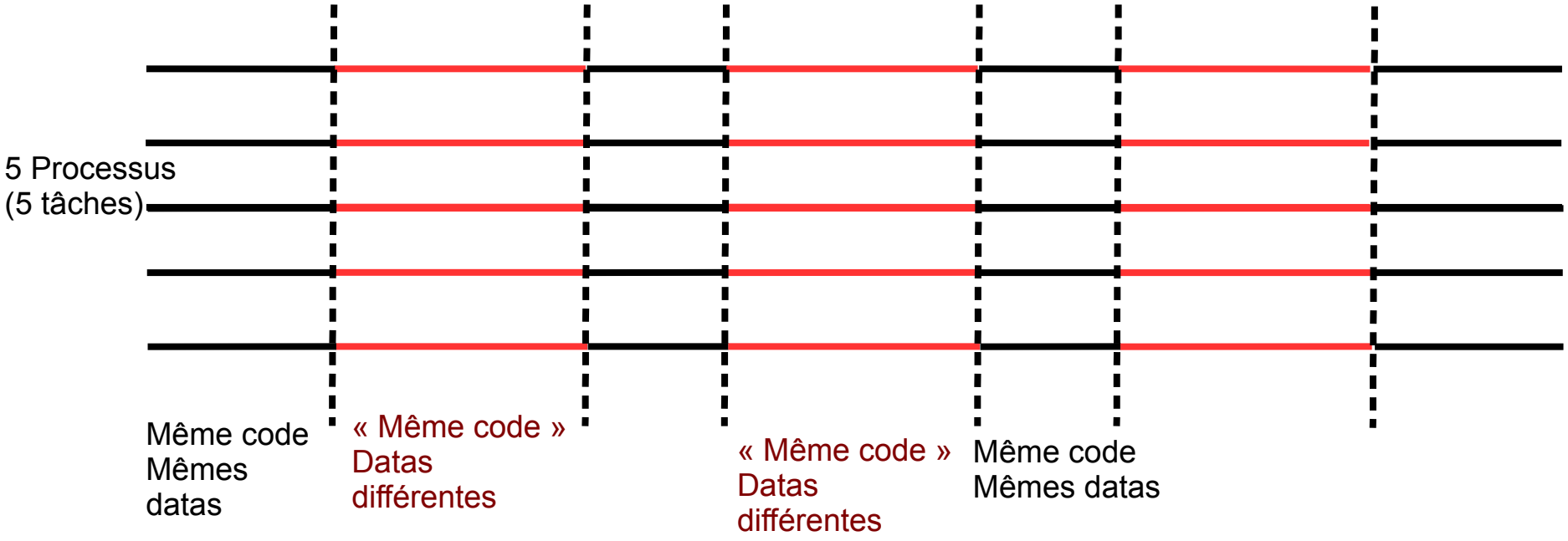
Outils de Programmation HP /MPI

$$TCalc(Np) = \frac{Tcalc^{//}}{Np} + TCalc^{\text{série}} + TComm(Np, architecture, vol\ données)$$

$$Acc(Np) = \frac{TCalc(1)}{TCalc(Np)} = f(TComm)$$

Le réseau de communication est un goulet d'étranglement potentiel pour l'efficacité de la parallèlisation

- Déroulement du programme :



On aura une succession de travail identique et de travail différentié !!!!
Le programme initial sera cloné en N tâches (tâches MPI)

Exemple de Cluster : Aselkam (UMMTO)

- Aselkam (Cluster BullX)
  - 32 Noeuds de Calcul aselkam[11-43]
    - 2 HexaCores par Noeud (2X6 Coeurs)
      - 24Go Ram
      - Intel(R) Xeon(R) CPU X5670 @ 2.93GHz
    - Total= 32 x 12 =384 Coeurs
  - 1 Noeud de Frontale aselkam0
  - 1 Noeud de serveur de Fichiers aselkam1 (partage NFS des /home … /opt
  - 1 Noeud de visualisation aselkam10
  - Réseau infiniband (40Gb/s)
  - Onduleur

Cluster Aselkam (TO)

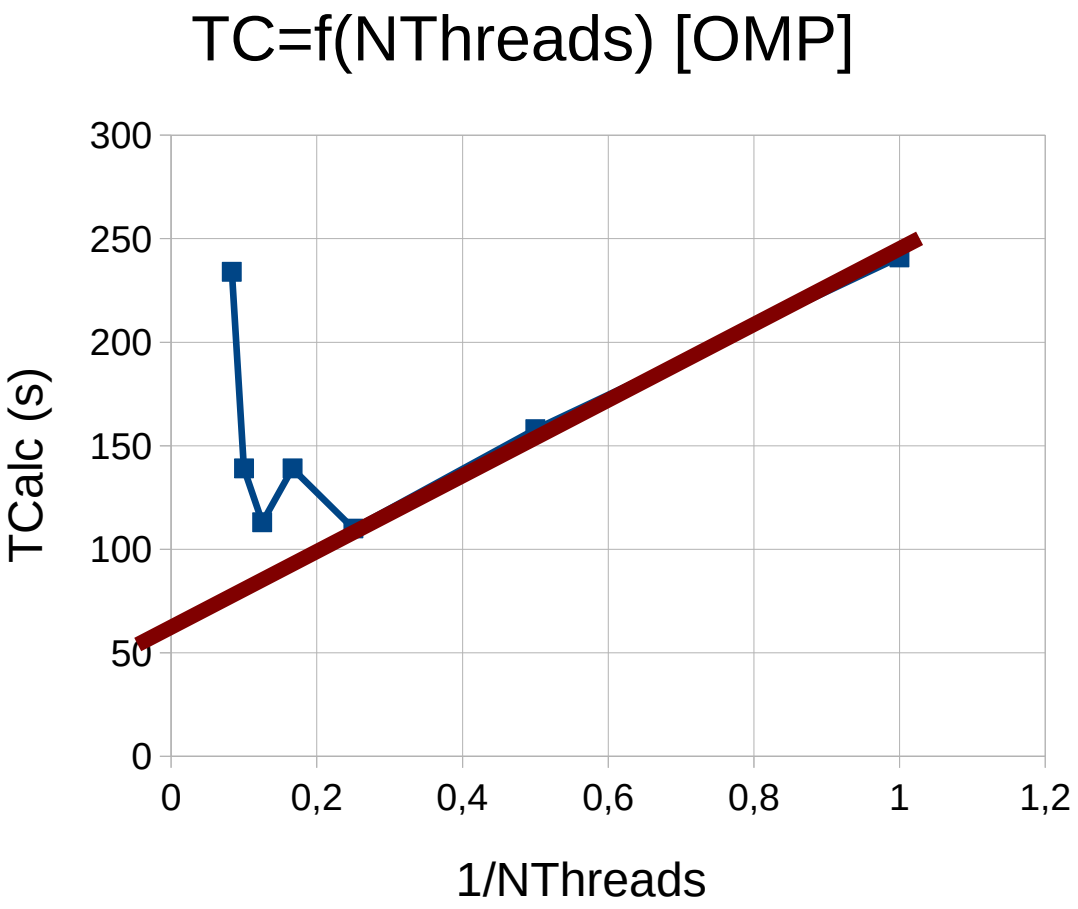
- Aselkam (Cluster BullX)
  - Pour utiliser un cluster
    - S'inscrire (demande login)
    - Préciser les logiciels dont vous avez besoin
    - Apprendre à l'utiliser au mieux de ses capacités
    - Ne jamais oublier qu'il est partagé entre plusieurs utilisateurs (espace disque / Nb de Noeuds)
      - Faire le ménage dans ses fichiers temporaires locaux et distants
      - Toujours envoyer les calculs à l'aide du gestionnaire de Jobs (Slurm : sbatch script_de_lancement)

Cluster Aselkam (TO)
Exemple d'utilisation: (Qespresso)

- Avec OpenMP
  - $sbatch OMP-Qe.sh

```
#!/bin/bash --login

#  Quantum espresso
#  Ce Batch est ecrit pour une utilisation sur Aselkam  adapted from  /home/OM10075/QEspressoTEST/RunTest.run
#  Chaque Noeud aselkam[11-42] possede 12 Coeurs
#  Le nombre de process à lancer est donc au Max de 12 (OMP)
#  Le nombre actuel(Np) est à renseigner avec la directive (Slurm)
#  #SBATCH --cpus-per-task=Np
#  Comme on utilise un seul noeud: #SBATCH  --nodes=1

#SBATCH –nodes=1            .....  Nb de Noeuds utilisés (ici 1)
#SBATCH --time=24:00:00
#SBATCH --account=[IL10105]
#SBATCH –cpus-per-task=12 ..... Nb de Threads utilisés (ici 12 car 12 Coeurs/Noeud)

# Chemin et Nom de l'executable
EXEC_DIR=/opt/quantum-espresso/src/espresso-5.1_par/bin
EXEC_BIN=pw.x
INPUT_QE=si.scf.in
OUTPUT_QE=ex.out

# Lancement du calcul
${EXEC_DIR}/${EXEC_BIN} < ${INPUT_QE} > ${OUTPUT_QE}
```

Cluster Aselkam (TO)
Exemple d'utilisation: (Qespresso)

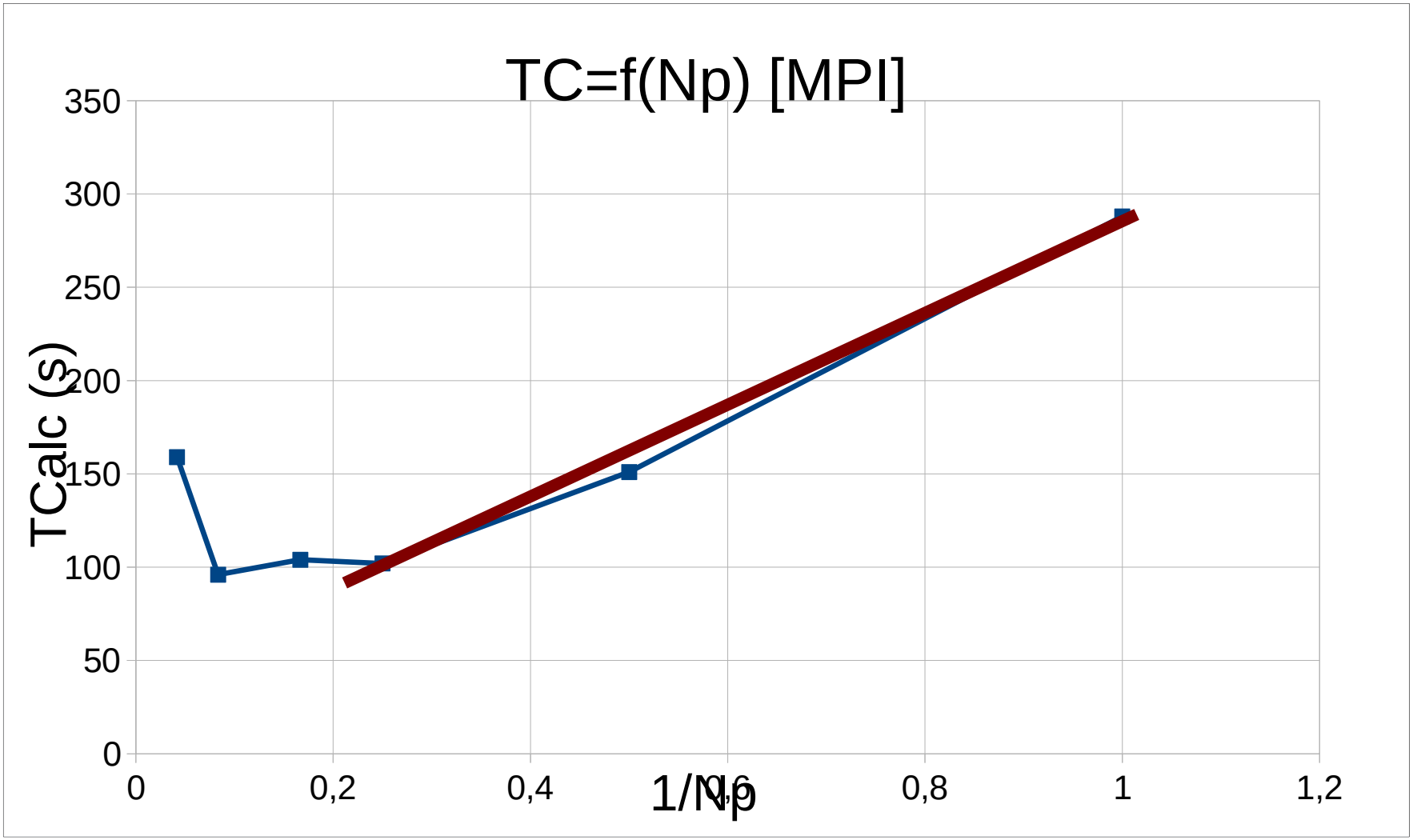


Cluster Aselkam (TO)
Exemple d'utilisation: (Qespresso)

- Avec MPI
  - $sbatch MPI-Qe.sh

```
#!/bin/bash --login
#  Quantum espresso
#  Ce Batch est ecrit pour une utilisation sur Aselkam  inspiré de /home/OM10075/QEspressoTEST/RunTest.run
#  Chaque Noeud aselkam[11-42] possede 12 Coeurs
#  Le nombre actuel(Np) est à renseigner avec la directive (Slurm)
#  #SBATCH  --ntasks=Np

#SBATCH --ntasks=24
#SBATCH --nodes=2
#SBATCH --time=24:00:00
#SBATCH --account=[IL10105]
#SBATCH --cpus-per-task=1

EXEC_DIR=/opt/quantum-espresso/src/espresso-5.1_par/bin
EXEC_BIN=pw.x
# lancement du calcul

export OMP_NUM_THREADS=1

mpirun -n 24  ${EXEC_DIR}/${EXEC_BIN} < si.scf.in> ex.out
```

Cluster Aselkam (TO)
Exemple d'utilisation: (Qespresso)



Conclusions

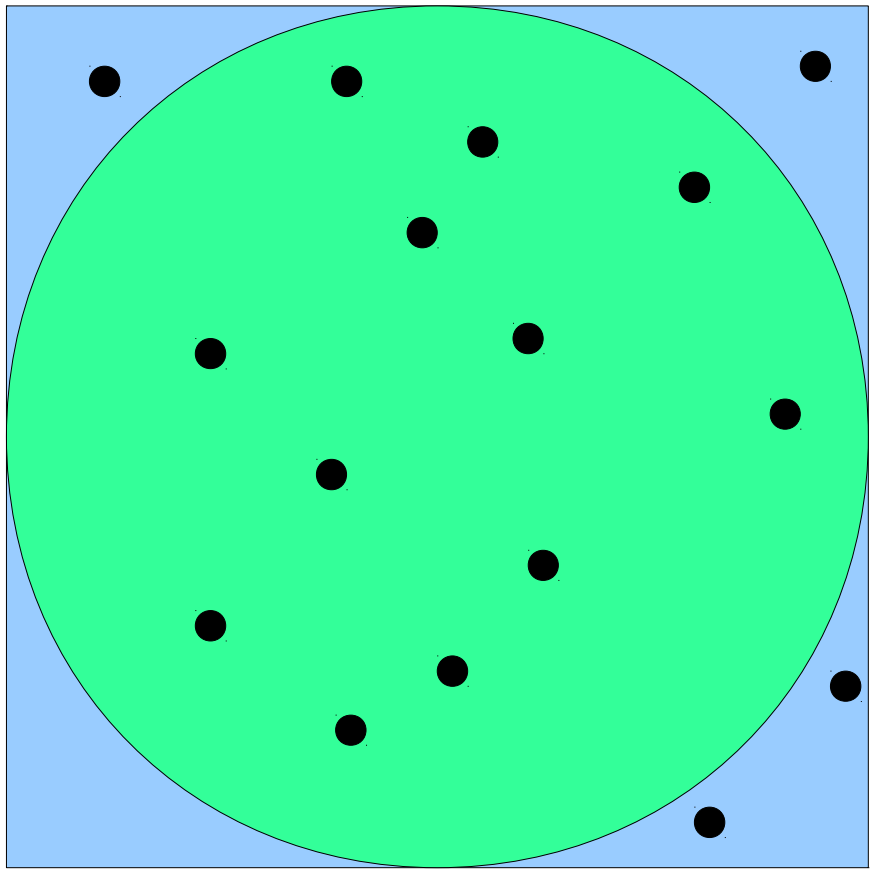
- Le Calcul parallèle devient la norme du calcul scientifique lourd
- Il permet d'envisager des Calculs autrefois quasi-infaisables
- Les logiciels de calcul scientifique sont tous passés à des versions //
- Travailler sur ces machines demande:
  - Un petit apprentissage
  - Une dose de responsabilité et de discipline (ne pas laisser des fichiers inutiles sur disque, ne pas vampiriser le temps de calcul, ne pas lancer ses calculs sur la frontale,....)

Conclusions

- Travailler sur cette machine vous permet :
  - De lancer des calculs longs sans craindre les coupures d'alimentation électrique (involontaires ou de Sonelgaz)
  - De lancer des calculs qui seront plus rapides et/ou plus nombreux
  - De disposer d'un espace de stockage fiable mais limité
  - De ne pas craindre les virus de sa machine (Window$)
  - De disposer de logiciels installés et utilisables (pour peu que l'on laisse le temps au responsable de la machine de s'en occuper)

Atelier /Workshop
//

On veut calculer la valeur approchée de π
par une méthode stochastique (MC) .



Un joueur maladroit mais uniformément maladroit, joue aux fléchettes.

Si l'on fait le rapport entre le nombre de fléchettes arrivant dans le cercle inscrit et le nombre de fléchettes dans le carré, on converge (lentement) vers le rapport des deux surfaces à savoir π/4

Si le nombre de lancers est très grand nous aurons une estimation de π

RDV en TP pour essayer de réaliser ce calcul avec une programmation paralléle OpenMP/MPI

Sahit