HPC : SIMULATION DU PROBLÈME À N-CORPS

Partie 1 : Parallélisation MPI

Jérôme Courtois, Mickael Chen

24 janvier 2020

Université Pierre-et-Marie Curie

CONTEXTE

N corps dans l'espace s'attirent mutuellement. On modélise cette attraction avec la loi de la gravitation universelle et le principe fondamentale de la dynamique :

- $F_{Cj \to Ci} = \frac{m_i m_j}{r_{ij}^3}(P_i - P_j)$
- $\sum_{\substack{j=1 \\ j \neq i}}^{j=N} F_{C_j \to C_i} = m_i A_i$

Une solution analytique de ce système différentiel existe pour deux ou trois corps. Au delà le problème est complexe, chaotique et dépend fortement des conditions aux limites

Modélisation

Un corps *Ci* est représenté par :

- une masse m_i
- un vecteur position $\mathsf{P_i} = (\mathsf{px_i}, \mathsf{py_i}, \mathsf{pz_i})$
- un vecteur vitesse $\mathsf{V_i} = (\mathsf{vx_i}, \mathsf{vy_i}, \mathsf{vz_i})$
- un vecteur accélération $\mathsf{F_i} = (\mathsf{fx_i}, \mathsf{fy_i}, \mathsf{fz_i})/\mathsf{m_i}$

On cherche à calculer la position des corps à chaque pas de temps t.

Schéma LeapFrog

A un temps t :

1- Kick : $V_{i,t+1/2} = V_{i,t} + A_{i,t}.dt/2$

2- Drift : $P_{i,t+1} = P_{i,t} + V_{i,t+1/2}$

3- Calcul des $A_{i,t}$

4- Kick : $V_{i,t+1} = V_{i,t+1/2} + A_{i,t}.dt/2$

Schéma LeapFrog

A un temps t :

1- Kick : $V_{i,t+1/2} = V_{i,t} + A_{i,t}.dt/2$

2- Drift : $P_{i,t+1} = P_{i,t} + V_{i,t+1/2}$

3- Calcul des $A_{i,t}$

4- Kick : $V_{i,t+1} = V_{i,t+1/2} + A_{i,t}.dt/2$

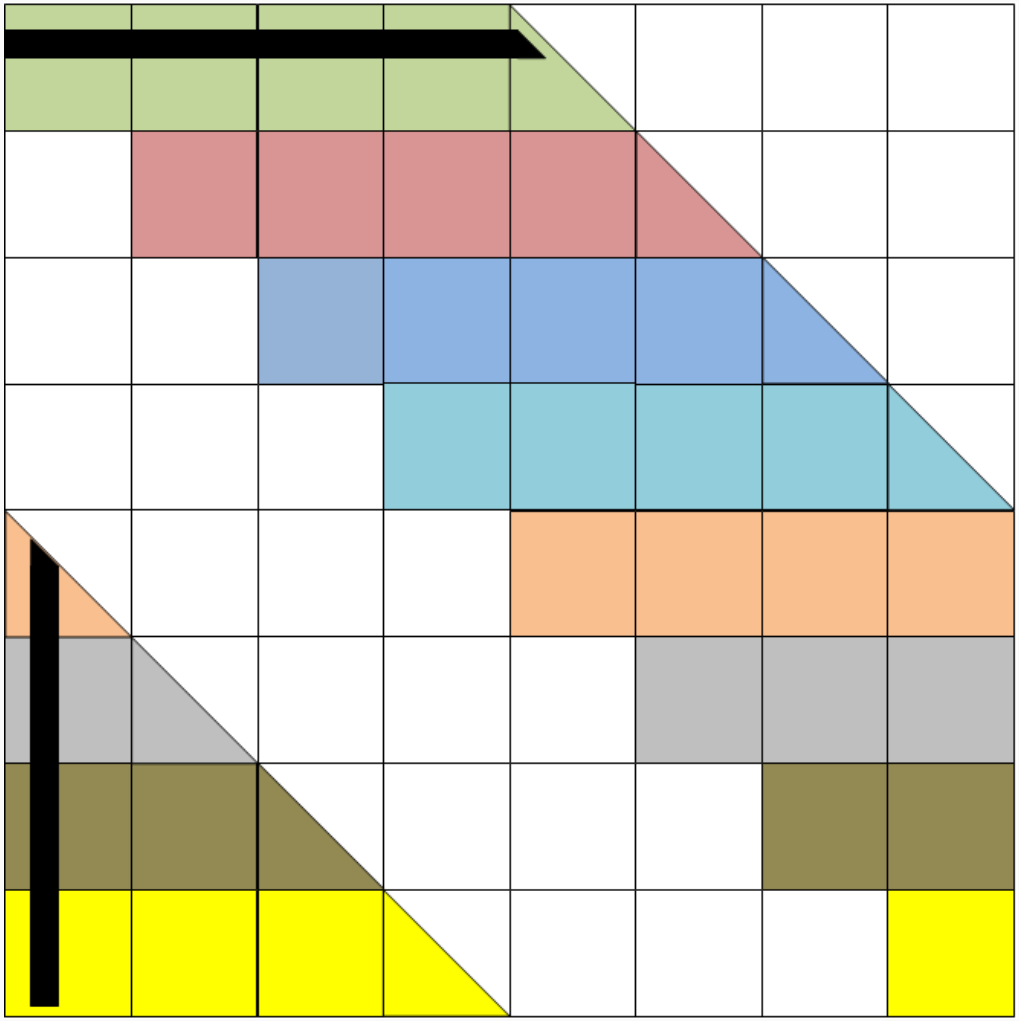
Mise à jour de l'accélération A_i

- $A_i = F_i / m_i$
- $F_i = \sum_{\substack{j=1 \\ j \neq i}}^{j=N} F_{C_j \to C_i}$
- $F_{C_j \to C_i} = \frac{m_i m_j}{(r_{ij}^2 + \epsilon^2)^{\frac{3}{2}}} (P_i - P_j)$

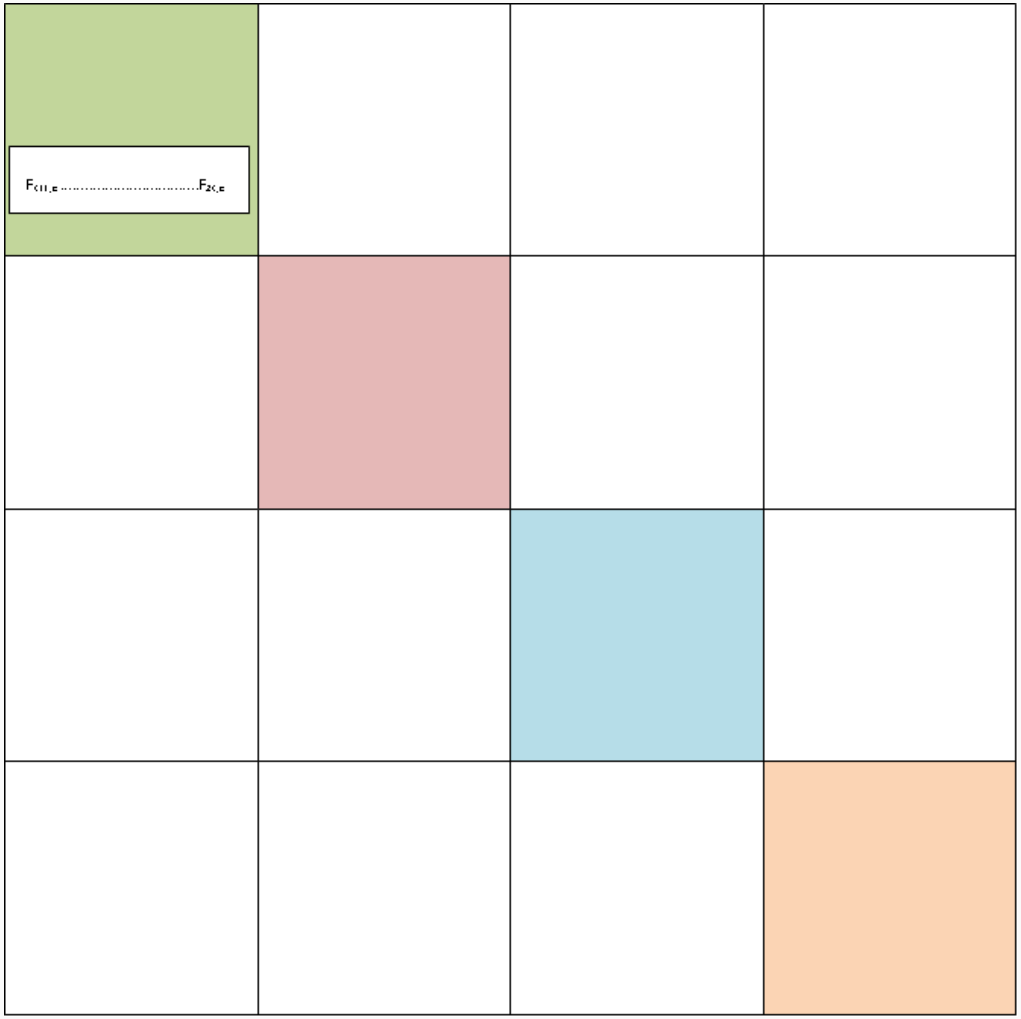
Mise à jour de l'accélération A_i

- $A_i = F_i / m_i$
- $F_i = \sum_{\substack{j=1 \\ j \neq i}}^{j=N} F_{C_j \to C_i}$
- $F_{C_j \to C_i} = \frac{m_i m_j}{(r_{ij}^2 + \epsilon^2)^{\frac{3}{2}}} (P_i - P_j)$

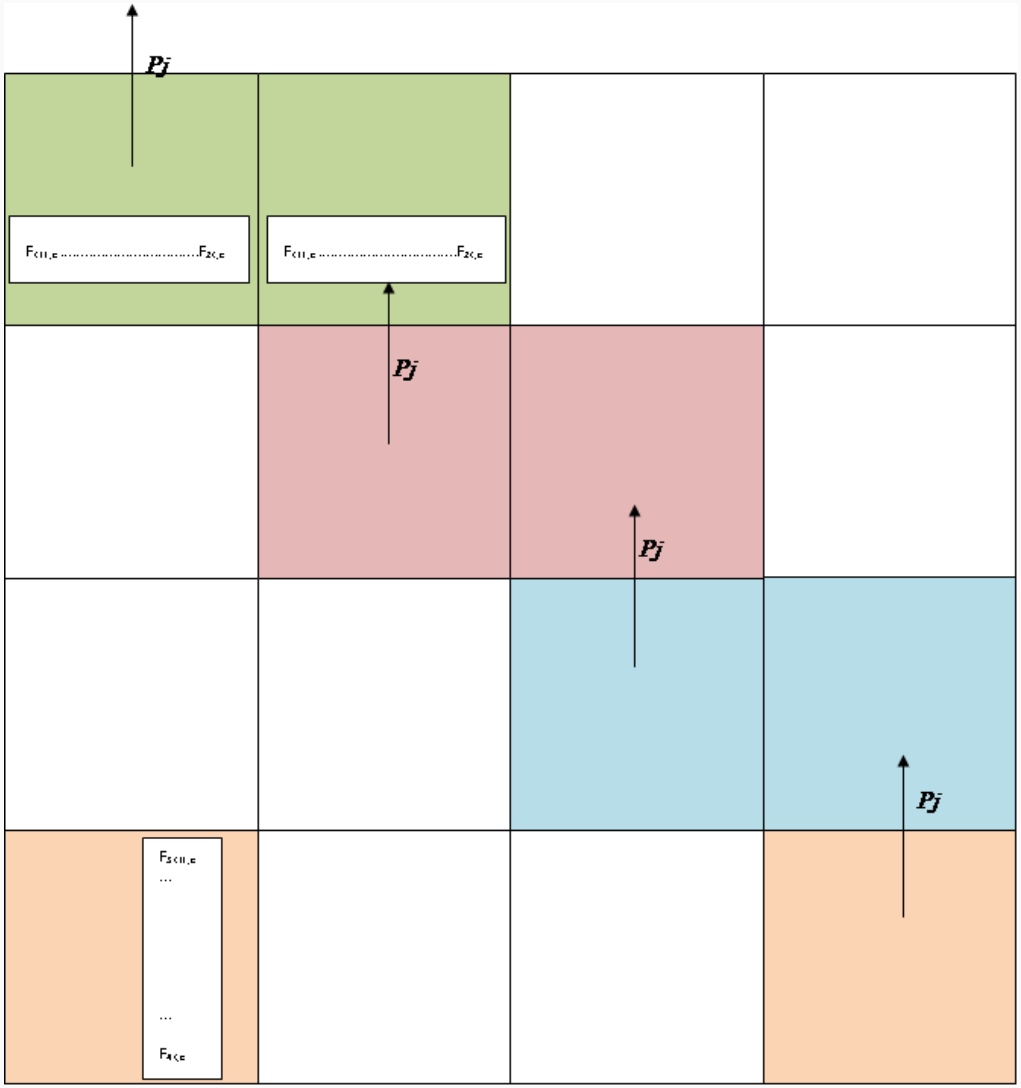
La matrice des $F_{i,j}$ pour 8 processus



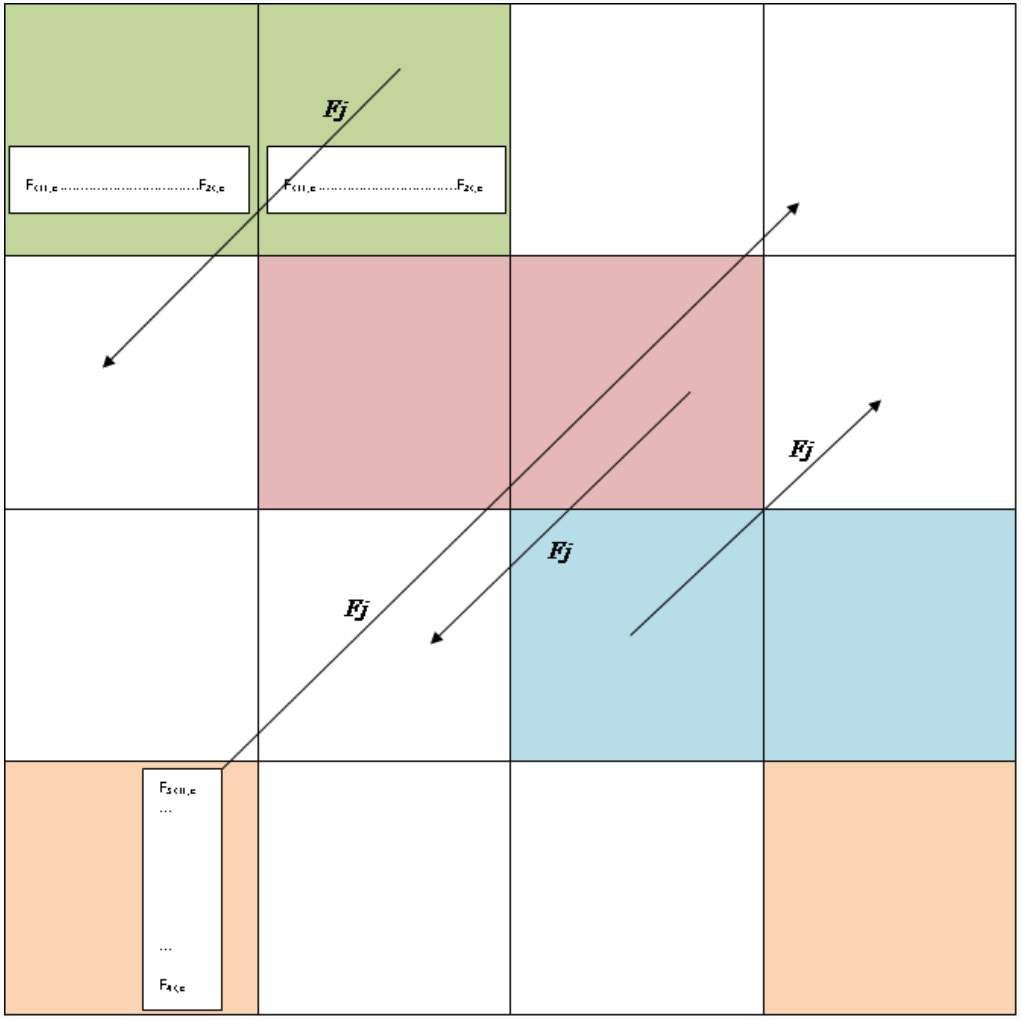
La matrice des $F_{i,j}$ pour 4 processus



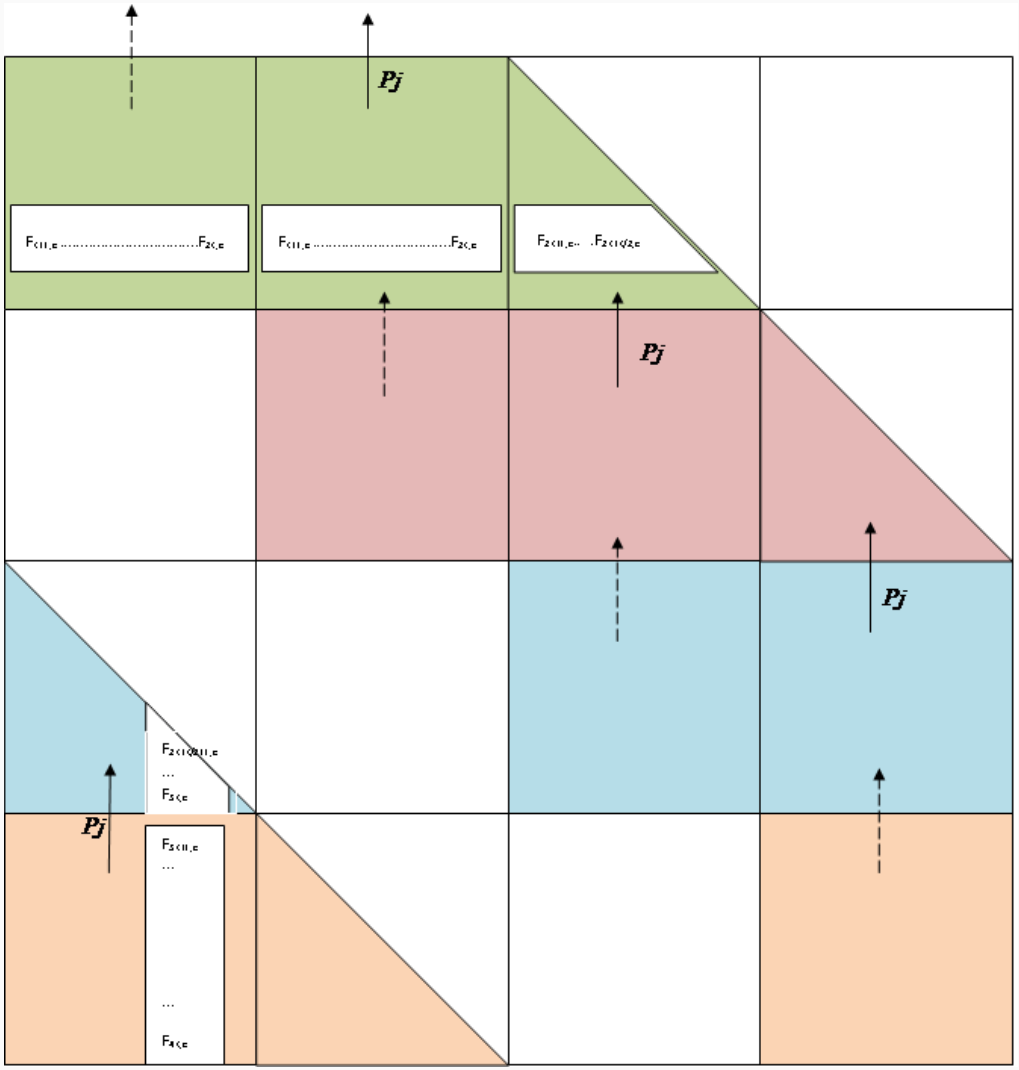
La matrice des $F_{i,j}$ pour 4 processus



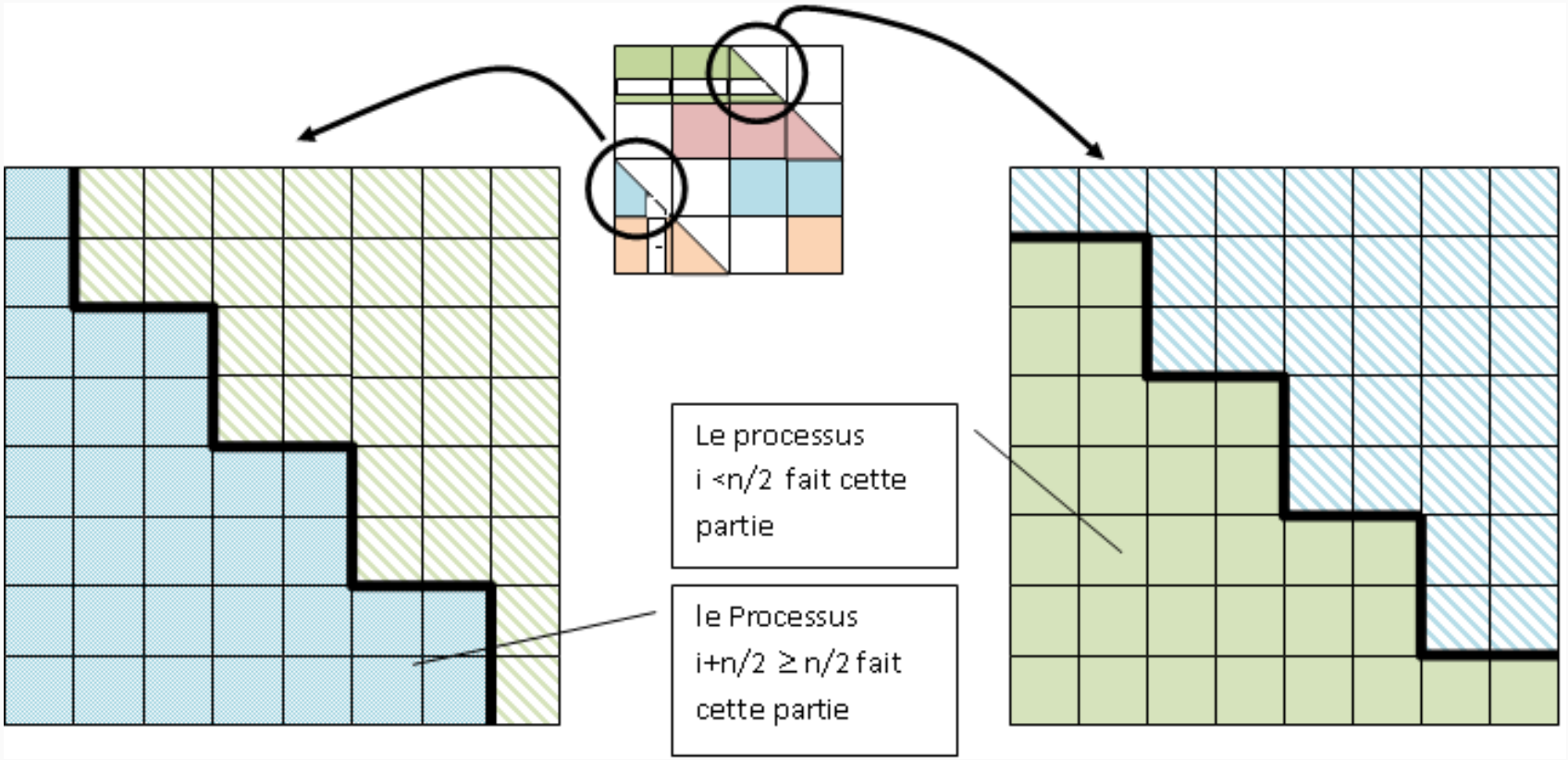
La matrice des $F_{i,j}$ pour 4 processus



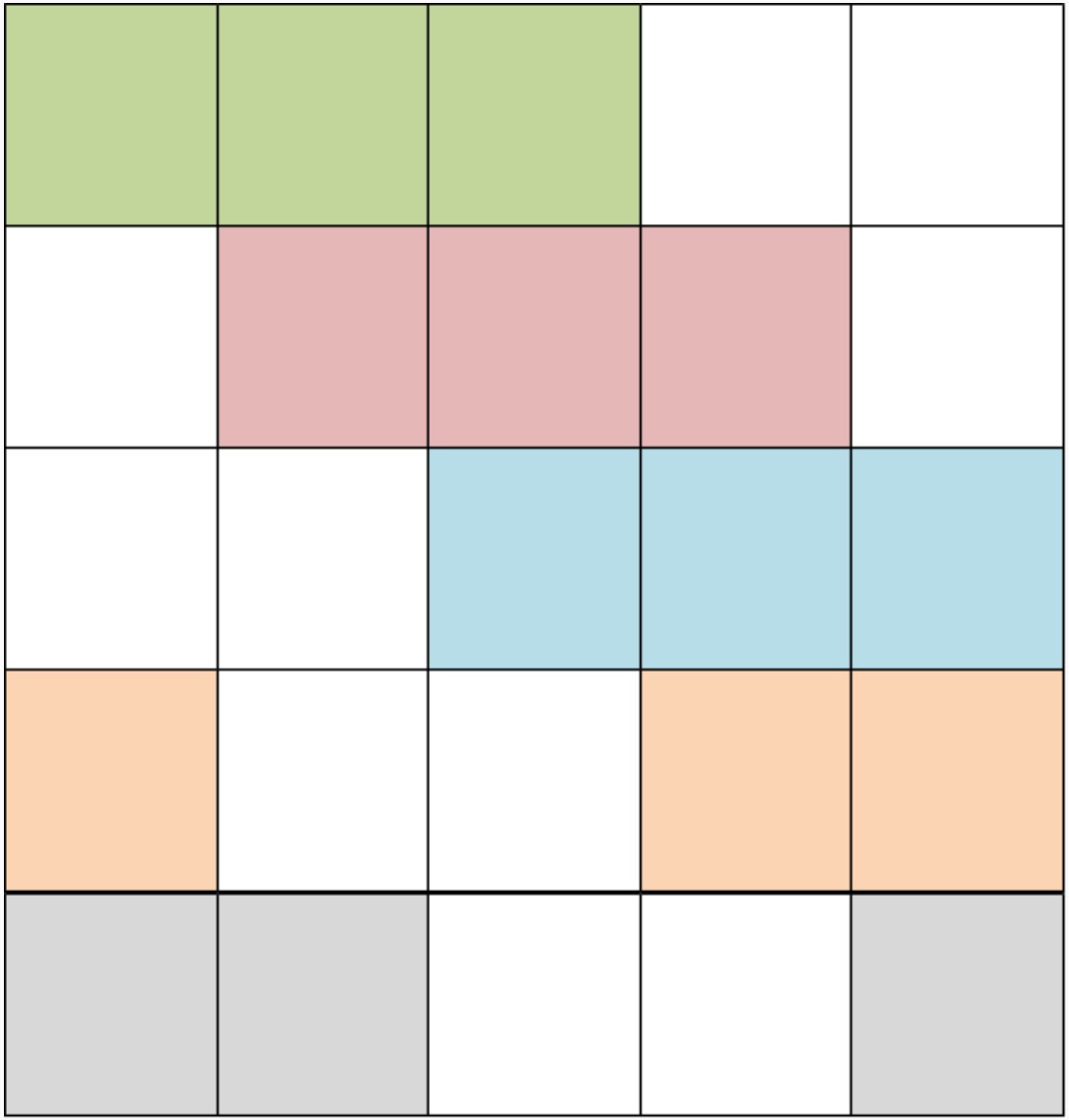
La matrice des $F_{i,j}$ pour 4 processus



Bloc de calculs finaux des $F_{i,j}$



La matrice des $F_{i,j}$ pour 5 processus



Communications

- A l'étape 1 : aucune
- Pour chacune des $nb_{proc}/2$ itérations suivantes :
  - Circulation des P_j
  - Retourner les sommes partiels des F_{ij} au processus j

$\Rightarrow$ Structure de tableaux pour faciliter ces communications
$\Rightarrow (nb_{proc}/2) * 6$ envois et réceptions par processus (et par pas de temps)

Extensibilité mémoire

Chaque processus alloue :

- N/nb_{proc} positions P_i : $(N/nb_{proc}) * 3 * sizeof(float)$
- N/nb_{proc} vitesses V_i : $(N/nb_{proc}) * 3 * sizeof(float)$
- N/nb_{proc} forces F_i : $(N/nb_{proc}) * 3 * sizeof(float)$
- N/nb_{proc} positions P_j : $(N/nb_{proc}) * 3 * sizeof(float)$
- N/nb_{proc} forces F_j : $(N/nb_{proc}) * 3 * sizeof(float)$
- N masses m_i : $N * sizeof(float)$

Recouvrement des communications

Nécessité d'allouer des buffers d'envois et de réceptions supplémentaires :

- P_{recv} pour préparer la réception des blocs de positions suivants.
- F_{send} pour envoyer le résultat de son calcul au processus concerné.
- F_{recv} pour recevoir les résultats calculés par les autres processus.

Pour la lecture initiale du fichier : On recouvre l'envoi des corps à un processus i par le chargement des corps à envoyer au processus $i+1$.
$\rightarrow$ Evite aussi d'allouer la mémoire nécessaire au stockage de l'ensemble des corps.

Recouvrement des communications

Pour un processus k, pour i de 1 à $nb_{proc}/2$:

- Commencer la transmission de P_j vers P_{recv} de $k-1$
- Calcul du bloc courant des interactions
- ...
- ...
- ...
- Echanger F_j et F_{send} puis envoyer F_{send} vers F_{recv} de $k+i$
- Terminer les communications sur P_j
- Echanger P_{recv} et P_j pour préparer l'itération suivante.

Recouvrement des communications

Pour un processus k, pour i de 1 à $nb_{proc}/2$:

- Commencer la transmission de P_j vers P_{recv} de $k-1$
- Calcul du bloc courant des interactions
- Terminer la réception dans F_{recv}
- Mettre à jour $Fi = Fi + F_{recv}$
- Terminer l'envoi de F_{send}
- Echanger F_j et F_{send} puis envoyer F_{send} vers F_{recv} de $k+i$
- Terminer les communications sur P_j
- Echanger P_{recv} et P_j pour préparer l'itération suivante.

Tests de l'implémentation

- Affichage de la somme totale des forces
- Affichage du nombre de $F_{i,j}$ effectivement calculés
- Affichage des positions de quelques corps et comparaison avec la version séquentielle
- Visualisation avec Nemo des versions séquentielles et parallèles

Comparaison données séquentielles / parallèles

```
################################################################################
Time now ( Step number) : 1.000000 (10)
Computation time = 1.377275 seconds
Interactions computed: 268419072
  Nb interactions / second: 194891415.246
  Gflop/s = 2.241 (11.5 flop with mutual)
-0.530790       0.434387        -0.373471
-0.060369       0.041643        -0.015160
-0.571420       -0.194220       0.106860
0.055681        0.269762        0.113298
1.529727        -0.679312       -0.315905
0.278874        -0.016547       -0.115331
1.673661        -1.171955       -0.461954
-0.278818       -0.074569       0.093147
-0.155913       0.116043        0.080295
-0.433044       0.307067        0.321900
Sum (force): (0.000000071129762, -0.000000031664968, -0.000000006053
Temps de demarrage= 8.338495
temps total =23.490285
8807311@ssh:~/Documents/hpc/NBODY directSequentielle$
```

```
################################################################################
Time now ( Step number) : 1.000000 (10)
Computation time = 0.259011 seconds
Interactions computed: 268419072
  Nb interactions / second: 1036322938.817
  Gflop/s = 11.918 (11.5 flop with mutual)
tnow= 1.000000, rang= 2 Computation time = 0.263644 seconds
-0.530786       0.434383        -0.373466
-0.060368       0.041646        -0.015164
-0.571416       -0.194219       0.106859
0.055683        0.269762        0.113296
1.529729        -0.679307       -0.315903
0.278871        -0.016543       -0.115330
1.673662        -1.171958       -0.461950
-0.278821       -0.074568       0.093146
-0.155911       0.116043        0.080300
-0.433044       0.307067        0.321899
Sum (force): (0.000000013038516, 0.000000007916242, -0.000000015832484
```

16384 corps, à 1s, 10 premiers corps (pas de temps = 0.1 s)

Comparaison données séquentielles / parallèles

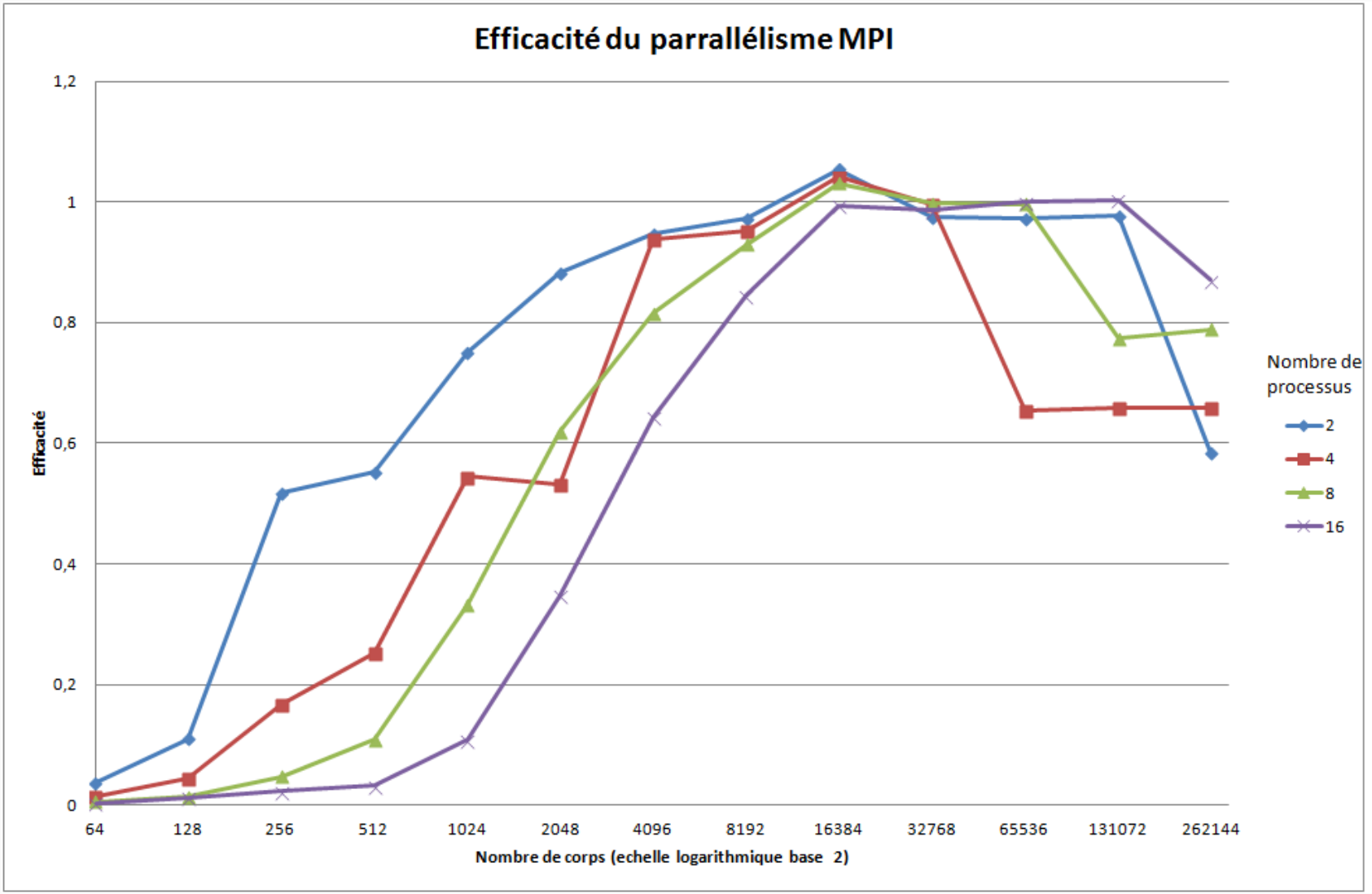
```
########################################################################
Time now ( Step number) : 0.200000 (2)
Computation time = 321.791056 seconds
Interactions computed: 68719214592
  Nb interactions / second: 213552282.850
  Gflop/s = 2.456 (11.5 flop with mutual)
-0.056988       -0.126304       0.096913
0.120689        0.148192        -0.236251
-0.020453       -0.143093       -0.197552
-0.245739       0.116884        0.066547
0.393393        0.335431        -0.174352
-0.363159       0.354155        -0.190540
-0.359836       0.432028        0.312073
0.443504        0.269464        0.008171
-0.108535       0.073376        -0.446325
0.434745        0.061833        -0.214702
Sum (force): (-0.000000011175871, 0.000000015366822, 0.000000020314474)
```

```
tnow= 0.200000, rang= 0 Computation time = 42.248945 seconds

########################################################################
Time now ( Step number) : 0.200000 (2)
Computation time = 42.248945 seconds
Interactions computed: 68719214592
  Nb interactions / second: 1626530901.437
  Gflop/s = 18.705 (11.5 flop with mutual)
tnow= 0.200000, rang= 3 Computation time = 42.573441 seconds
tnow= 0.200000, rang= 7 Computation time = 42.566630 seconds
tnow= 0.200000, rang= 5 Computation time = 42.570570 seconds
tnow= 0.200000, rang= 1 Computation time = 42.582674 seconds
tnow= 0.200000, rang= 6 Computation time = 42.567869 seconds
tnow= 0.200000, rang= 2 Computation time = 42.575917 seconds
-0.056988       -0.126304       0.096913
0.120689        0.148192        -0.236251
-0.020453       -0.143093       -0.197552
-0.245738       0.116885        0.066547
0.393392        0.335431        -0.174352
-0.363159       0.354155        -0.190540
-0.359836       0.432028        0.312073
0.443504        0.269463        0.008171
-0.108534       0.073376        -0.446325
0.434746        0.061833        -0.214702
Sum (force): (-0.000000023166649, -0.000000011175871, 0.000000012310920)
```

262144 corps, à 0.2s, 10 premiers corps (pas de temps = 0.1 s)

Évolution de l'Efficacité



Nous vous remercions
de votre attention.