TP - 1
Parallélisation du champ proche
pour un problème à N-corps

Le calcul du champ de force dû à N particules implique de calculer pour chaque particule son interaction avec toutes les autres particules. Un tel calcul a une complexité en N^2 et n'est donc pas envisageable lorsque N est grand. La méthode des multipoles rapide décompose ce calcul en deux contributions : le champ proche et le champ lointain. pour obtenir une complexité linéaire.

Le domaine de simulation est découpé en $P \times P$ boîtes avec $P = 2^h$. Dans chaque boîte le nombre et la position des particules sont construites de manière aléatoire. Le calcul du champ proche nécessite de calculer les contributions (potentiel et force) entre les particules de la boîte A avec l'ensemble de ses boîtes voisines. Le potentiel p_i et la force F_i agissant sur la particule i en position X_i dû aux particules j dans la boîte B en position Y_j ont pour expression

$$p_i = \sum_{Y_j \in B} \frac{1}{\|X_i - Y_j\|} \text{ et } F_i = \sum_{Y_j \in B} \frac{X_i - Y_j}{\|X_i - Y_j\|^2}.$$

Pour chaque boîte ou feuille, on construit le vecteur, `interactions`, des boîtes voisines. Pour la boîte de numéro $k = (i, j)$ la liste des voisins, ou des interactions, est donnée par la liste $\mathcal{V}_k = \{\text{boîtes } B \text{ existantes de coordonnée}(i \pm 1, j \pm 1)\}$. Le calcul des interactions dues à l'ensemble des particules de ces boîtes est calculé à l'aide de l'algorithme suivant

```
void compute_near_seq(std::vector<LeafT> &grid) {
// boucle sur les feuilles ou boîtes du domaine
  for (int k = 0; k < grid.size(); ++k) {

    auto my_part = grid[k].particles;  // tableau des particules de la feuille
    auto n1 = grid[k].nb_part;         // nombre de particules de la feuille

    for (auto &p2p_inter : grid[k].interactions) { // Boucle sur les feuilles voisines
      if (grid[p2p_inter].nb_part > 0) {
      // Calcul des contributions d'une feuille voisine sur la feuille courante
        p2p_outer(n1,  my_part,  grid[p2p_inter].nb_part,  grid[p2p_inter].particles);
      }
    }
    // Calcul des contributions dans la feuille
    p2p_inner(n1,  my_part);
  }
}
```

Listing 1: La fonction, `compute_near_seq`, calcule le champ proche.

L'objectif de ce TP est de paralléliser par différentes approches le calcul du champ proche à l'aide d'OpenMP.

A Approche non optimale

Dans cette première approche, nous considérons la fonction `p2p_outer` qui calcule les interactions sur les particules de la boîte A à partir des particules de la boîtes B.

```
        p2p_outer(n_A,  particules_A,  n_B,  particules_B);
```

A.1 Optimisation

Dans un premier temps, vous devez optimiser les fonctions `p2p_inner` et `p2p_outer` à l'aide des pragma SIMD d'OpenMP.

A.2 Parallélisation

On considère maintenant trois différentes méthodes de parallélisation. On parallélise le calcul de la boucle sur les feuilles. Un thread calcule toutes les contributions pour une feuille.

Q1 Utiliser un parallélisme de boucle `#pragma omp for`

Q2 Utiliser un parallélisme de boucle mais avec une approche en tâche

Q3 Utiliser un parallélisme de tâche. Une tâche calcule toutes les contributions pour une feuille.

Comparer ces trois approches parallèles en calculant l'accélération du code. On rappelle que l'accélération, S, est défini par

$$S = \frac{\text{temps séquentiel}}{\text{temps parallèle}}$$

B Approche optimale

Pour diminuer la complexité des calcul,s on considère la deuxième loi de Newton $F_{i,j} + F_{j,i} = 0$, qui lie l'interaction F_{ij} entre la particule i et la particule j avec l'interaction réciproque $F_{ji} = -F_{ij}$. Nous utilisons cette remarque pour éviter de calculer deux fois les contributions. Pour cela, la liste des interactions ne considère plus que les boîtes voisines ayant un indice plus petit que celui de la boîte courante. Le calcul de la liste d'interactions se fait en positionnant le booléen `mutual` à `true`.

```
build_interactions(grid,  true)
```

Nous remplaçons maintenant dans le calcul `p2p_outer_seq` la fonction `p2p_outer` par la fonction

```
        p2p_mutual(n_A,  particules_A,  n_B,  particules_B);
```

qui met à jour les contributions du potentiel et de la force dans les deux boîtes. Les particules dans les deux boîtes sont modifiées.

Modifier le code issu de la question Q3 pour qu'il donne le même résultat.

Indication : écrire le graphe de tâches pour les deux premières boîtes.