

DE LA RECHERCHE À L'INDUSTRIE



# Mini-projet du cours de programmation GPU

Pierre KESTENER

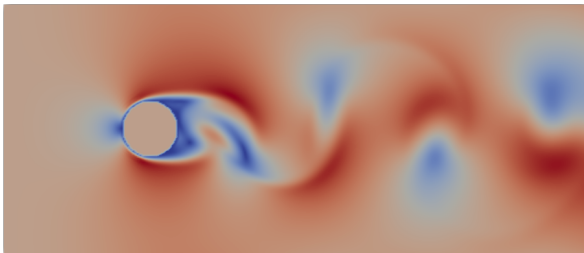
*CEA / DRF / IFRU / DEDIP  
Master spécialisé HPC-IA, Mines  
Paritech, Sophia Antipolis*

[www.cea.fr](http://www.cea.fr)

*11 janvier 2021*



- LBM (Lattice Boltzmann Method) : simulation d'écoulement fluide



- ▶ théorie cinétique des gaz
- ▶ fonction de distribution  $f(\vec{\mathbf{r}}, \vec{\mathbf{c}}, t)$ , avec  $\vec{\mathbf{r}} \in \mathbb{R}^3$  et  $\vec{\mathbf{c}} \in \mathbb{R}^3$
- ▶ interprétation de  $f(\vec{\mathbf{r}}, \vec{\mathbf{c}}, t) d\vec{\mathbf{r}} d\vec{\mathbf{c}}$  : nombre de particules dans le volume élémentaire  $d\vec{\mathbf{r}}$  et de vitesse comprise entre  $\vec{\mathbf{c}}$  et  $\vec{\mathbf{c}} + d\vec{\mathbf{c}}$
- ▶ les **grandeurs hydrodynamiques macroscopiques usuelles** (densité, vitesse, énergie, ...) sont définies comme les moments de la fonction de distribution :
  - ▶ **densité** :  $\rho(\vec{\mathbf{r}}, t) = \int f(\vec{\mathbf{r}}, \vec{\mathbf{c}}, t) d\vec{\mathbf{c}}$
  - ▶ **quantité de mouvement** :  $\vec{\mathbf{p}}(\vec{\mathbf{r}}, t) = \int f(\vec{\mathbf{r}}, \vec{\mathbf{c}}, t) \vec{\mathbf{c}} d\vec{\mathbf{c}} = \rho \vec{\mathbf{v}}$
  - ▶ **énergie cinétique**  $e_{cin}(\vec{\mathbf{r}}, t) = \int f(\vec{\mathbf{r}}, \vec{\mathbf{c}}, t) \|\vec{\mathbf{c}}\|^2 d\vec{\mathbf{c}} = \rho \vec{\mathbf{v}}$
- ▶ si on sait faire évoluer dans le temps  $f$ , alors on peut connaître la dynamique du mouvement du fluide  $\Rightarrow$  équation de boltzmann

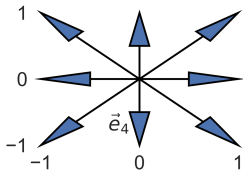
équation de Boltzmann est une équation de transport qui décrit comment évolue dans le temps la distribution des particules

$$\frac{\partial f}{\partial t} + \vec{c} \cdot \nabla_{\vec{r}} f + \vec{F} \cdot \frac{\partial f}{\partial \vec{c}} = \left( \frac{\partial f}{\partial t} \right)_{\text{coll}}$$

- **Problème** : pour simuler numériquement, il faut discrétiser  $f(\vec{\mathbf{r}}, \vec{\mathbf{c}}, t)$  sachant que  $\vec{\mathbf{r}} \in \mathbb{R}^3$  et  $\vec{\mathbf{c}} \in \mathbb{R}^3 \Rightarrow$  le nombre de variables est rapidement gigantesque ( $\sim N^6$ )  
exemple :  $N = 100 \Rightarrow N^6 = 10^{12}$  soit 8 TBytes de mémoire vive juste pour stocker  $f$  à un pas de temps, c'est impraticable en pratique.
- **Solution** : on discrétise l'espace (variable  $\vec{\mathbf{r}}$  sur une grille  $N^3$  et la vitesse  $\vec{\mathbf{c}}$  sur un petit nombre (quelque unités à quels dizaines); cela s'appelle la méthode dite de Boltzmann sur réseau qui donne des simulations numériques en bon accord avec les observations expérimentales.  
avantages principaux des méthodes LBM : facile à mettre en œuvre numériquement et facile à paralléliser.

De manière simplifiée, on peut dire que l'évolution temporelle de la fonction de distribution (où plus exactement des  $q$  composantes de  $f$ ) est calculée en itérant les deux opérations suivantes :

- ▶ collision :  $f_i(\vec{x}, t + \delta_t) = f_i(\vec{x}, t) + \frac{f_i^{eq}(\vec{x}, t) - f_i(\vec{x}, t)}{\tau_f}$ ,  
avec  $0 \leq i < q$   
le terme de collision choisi ici modélise un retour (relaxation) vers la distribution d'équilibre
- ▶ stream :  $f_i(\vec{x} + \vec{e}_i, t + 1) = f_i(\vec{x}, t)$   
les particules modélisées par  $f_i$  se déplacent dans la direction  $e_i$



**Objectif :** En se basant soit sur la version python, soit sur la version C++, et en utilisant le cours de programmation des GPUs, proposer une version parallélisée de la méthode LBM.

- ▶ Choisir le modèle de programmation adapté pour la parallélisation :
  - ▶ numba (ou Cupy) pour python<sup>1</sup>
  - ▶ OpenACC pour C++
  - ▶ CUDA/C++ pour les plus courageux ;)
- ▶ Fournir un mini rapport détaillant votre démarche (identification des noyaux de calcul) et une étude de performance : on pourra par exemple présenté une courbe de speedup (rapport du temps de calcul de la version séquentielle fournie sur le temps de calcul de la version parallèle).

---

1. éventuellement essayer legate sachant qu'il faudra bien analyser les messages d'erreur et contourner les fonctionnalités non encore implantées dans legate/cunumeric