La méthode PIC en OpenCL

Anaïs Crestetto

Journée GPU de Strasbourg

25 février 2010.

Sommaire

The PIC method

- We consider the Vlasov-Poisson system :

$$\begin{cases} \frac{\partial f}{\partial t}(x,v,t) + v\frac{\partial f}{\partial x}(x,v,t) - E(x,t)\frac{\partial f}{\partial v}(x,v,t) = 0 \\ \frac{\partial E}{\partial x}(x,t) = \rho(x,t) = 1 - \int_{-\infty}^{+\infty} f(x,v,t)dv \end{cases}.$$

- The Particle-In-Cell (PIC) method :

we consider N macro-particles : position x_k, velocity v_k and weight $\omega_k = \omega$,

we approach f by : $f_N(x,v,t) = \sum_{k=1}^{N} \omega_k \delta(x - x_k(t))\delta(v - v_k(t))$.

- We solve the electric field equation on a mesh.

- We move the macro-particles with the Newton's law : $\frac{dx_k}{dt} = v_k$ and $\frac{dv_k}{dt} = -E$.

Algorithm

We know x_k^n, v_k^n for each particle $k = 0, \ldots, N-1$, E_i^n at each grid point $i = 0, \ldots, N_x - 1$.

- Explicit Euler's scheme (for example) for moving the particles :

$v_k^{n+1} = v_k^n - \Delta t E_{i_k}^n$, with i_k such that $x_k^n \in [x_{i_k}, x_{i_k+1}]$
$x_k^{n+1} = x_k^n + \Delta t v_k^n$,

- density ρ_i^{n+1} calculated by linear interpolation : we need the position of the particles located in $[x_{i-1}, x_i]$ and $[x_i, x_{i+1}]$ at time $n+1$,
- electric field E_i^{n+1} : we need the densities ρ_j^{n+1}, for $j < i$.

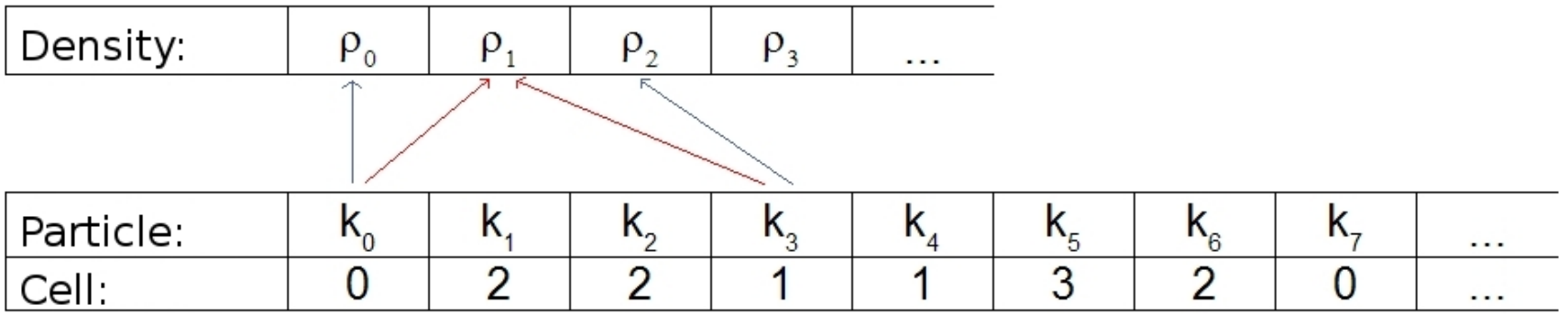
How to parallelize ?

We work with particles ($N \sim 1\,000\,000$) and on a grid ($N_x \sim 128$).

- Particle move : one work-item per particle,
- E : one work-item for each grid point.

ρ computing

First idea and first problem : move the particles and compute their contribution to ρ in the same Kernel $\longrightarrow$ problem when two particles add at the same time their contribution to the same grid point :

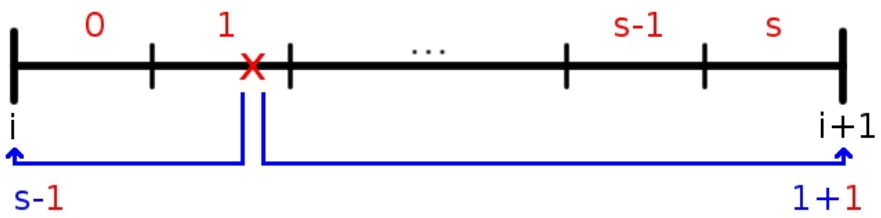


k_0 reads ρ_1^n,
k_3 reads ρ_1^n,
k_0 adds its contribution to ρ_1^n and writes ρ_1 : $\rho_1 = \rho_1^n + contr(k_0)$,
k_3 adds its contribution to ρ_1^n and overwrites ρ_1 : $\rho_1 = \rho_1^n + contr(k_3)$.
Finally $\rho_1 = \rho_1^n + contr(k_3)$ instead of $\rho_1 = \rho_1^n + contr(k_0) + contr(k_3)$.

Atomic addition

Solution : use the atomic addition to add the contributions of particles.

- When a particle reads ρ and adds its contribution, the other work-items are stopped until this addition is finished.
- Constraint : we have to add integer values !
- Cut each cell and transform the contributions into integer values :



if $x_k \in [x_i, x_{i+1}]$ and if k is in the subcell j, this particle adds $s-j$ to $contr_i$ and $1+j$ to $contr_{i+1}$.

- When we compute E we use the good densities $\rho_i = \frac{contr_i}{s+1}\frac{\omega}{\Delta x}$.

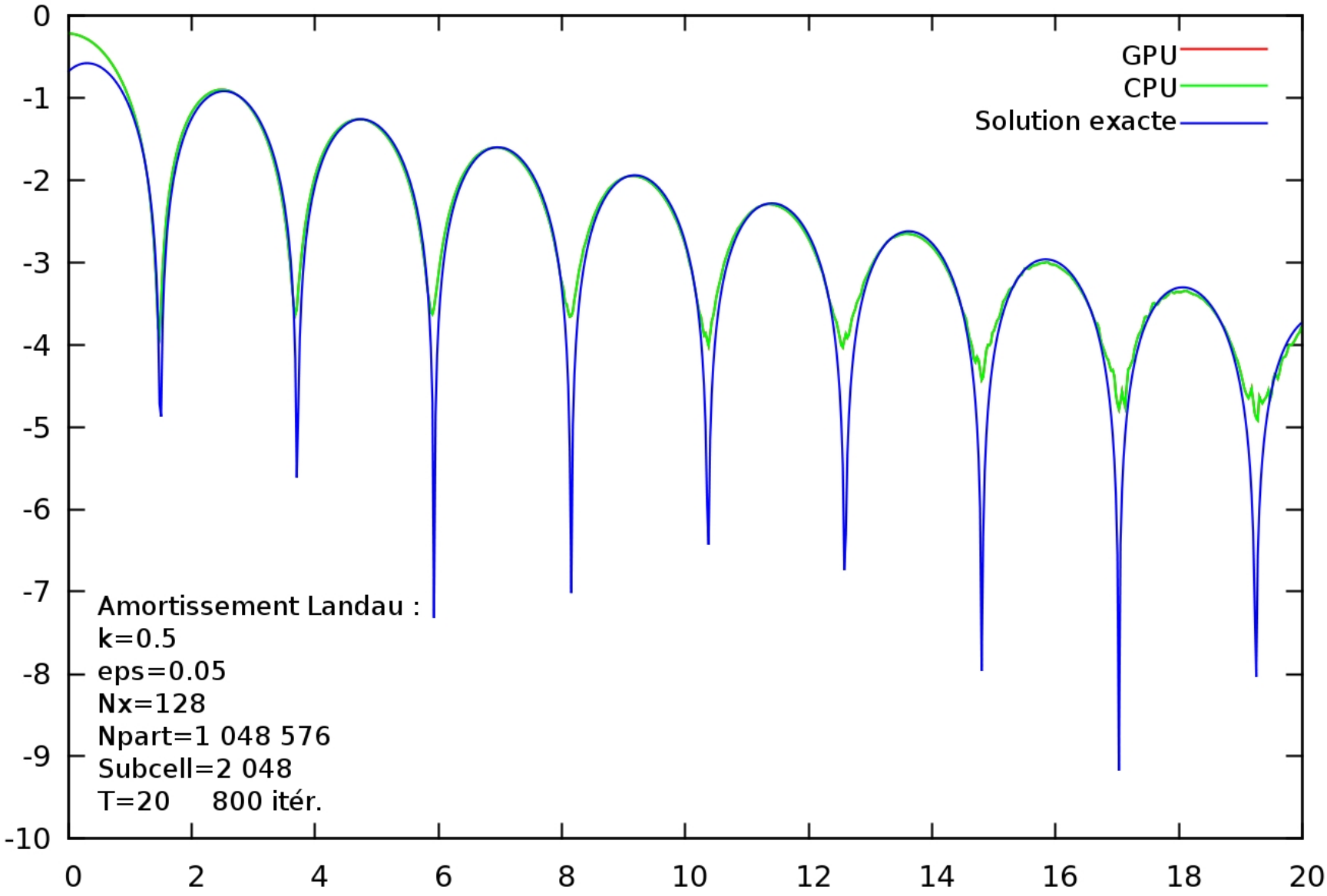
Speedups

524 288 particles	CPU	NVIDIA GeForce GTX 260	ATI Radeon HD 5750
Time	37s.	15s.	5s.
Speedup		**2.47**	**7.40**

1 048 576 particles	CPU	NVIDIA GeForce GTX 260	ATI Radeon HD 5750
Time	75s.	30s.	9s.
Speedup		**2.50**	**8.33**

2 097 152 particles	CPU	NVIDIA GeForce GTX 260	ATI Radeon HD 5750
Time	151s.	61s.	19s.
Speedup		**2.48**	**7.95**

Graph


GPU
CPU
Solution exacte
Amortissement Landau :
k=0.5
eps=0.05
Nx=128
Npart=1 048 576
Subcell=2 048
T=20 800 itér.
0
-1
-2
-3
-4
-5
-6
-7
-8
-9
-10
0
2
4
6
8
10
12
14
16
18
20

Reference

Fast parallel Particle-To-Grid interpolation for plasma PIC simulations on the GPU,
G. Stantchev, W. Dorland, N. Gumerov,
J. Parallel Distrib. Comput. 68 (2008).