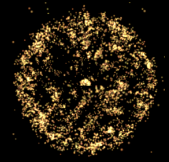
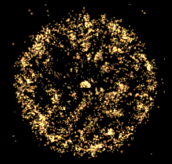


N-test szimuláció molekuladinamika GPU-n

RMKI GPU nap 2010

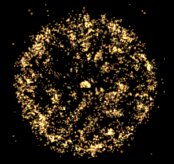
Jurek Zoltán
MTA SZFKI, Röntgendiffrakciós csoport



Molekuladinamika

Számítógépes szimuláció atomcsoportok, molekulák tulajdonságinak megértésére és leírására az atomok közötti mikroszkópikus kölcsönhatások alapján.

- klasszikus részecskék
- kölcsönhatások → klasszikus erők
  (származhat kvantummechanikai számolásból)
- klasszikus mozgásegyenletek
- csatolt közönséges diff.egyenlet-rendszer
  kezdetiérték problémájának megoldása



Mozgásegyenletek

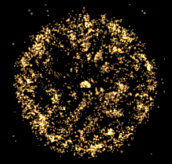
$$m_i \ddot{\boldsymbol{r}}_i = \boldsymbol{f}_i \qquad \boldsymbol{f}_i = -\frac{\partial}{\partial \boldsymbol{r}_i}\mathcal{U}$$

Potenciál pl.:

$$\mathcal{U}_{\text{non-bonded}}(\boldsymbol{r}^N) = \sum_i u(\boldsymbol{r}_i) + \sum_i \sum_{j>i} v(\boldsymbol{r}_i, \boldsymbol{r}_j) + \dots .$$

ahol:

$$\boldsymbol{r}^N = (\boldsymbol{r}_1, \boldsymbol{r}_2, \dots \boldsymbol{r}_N)$$

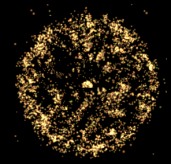


Párpoteciálok

Például:

$$v^{\mathrm{LJ}}(r) = 4\varepsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right]$$

$$v^{\mathrm{Coulomb}}(r) = \frac{Q_1 Q_2}{4\pi\epsilon_0 r}$$



Numerikus módszerek

Mozgásegyenletek:

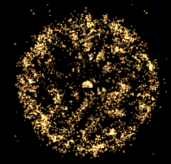
$$m_i \ddot{\boldsymbol{r}}_i = \boldsymbol{f}_i \qquad \boldsymbol{f}_i = -\frac{\partial}{\partial \boldsymbol{r}_i}\mathcal{U}$$

Kezdeti érték:

$$\{\boldsymbol{r}^N(t=0), \boldsymbol{p}^N(t=0)\}$$

Feladat:

$$\{\boldsymbol{r}^N(t), \boldsymbol{p}^N(t)\} \rightarrow \{\boldsymbol{r}^N(t+\delta t), \boldsymbol{p}^N(t+\delta t)\}$$



Numerikus módszerek

Ha az erők csak a helytől függnek:

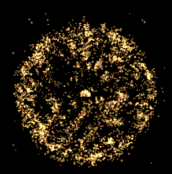
$$f_i = f_i(\boldsymbol{r}^N)$$

akkor pl. a (közkedvelt) Velocity Verlet algoritmus:

$$\boldsymbol{p}_i(t + \tfrac{1}{2}\delta t) = \boldsymbol{p}_i(t) + \tfrac{1}{2}\delta t \boldsymbol{f}_i(t)$$

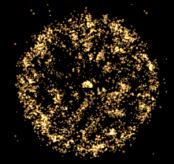
$$\boldsymbol{r}_i(t + \delta t) = \boldsymbol{r}_i(t) + \delta t \boldsymbol{p}_i(t + \tfrac{1}{2}\delta t)/m_i$$

$$\boldsymbol{p}_i(t + \delta t) = \boldsymbol{p}_i(t + \tfrac{1}{2}\delta t) + \tfrac{1}{2}\delta t \boldsymbol{f}_i(t + \delta t)$$



Molekuladinamika programcsomagok

Code	Version	Release Target	Notes
VMD	1.8.7	Released	Free download from VMD site
NAMD	2.7 beta	September 2009	Nightly build being rolled into beta 2
HOOMD	0.8.1	Released	Multi-GPU support in 0.8.1
HMMER	0.9.1	Released	GPU-HMMER available now
Autodock	0.9	Beta	From Silicon Informatics
CHARMM	Beta	December 2009	
Amber	Alpha patch		Generalized Born only, 1 GPU
GROMACS	4.0	Released	CUDA client based on OpenMM adds support for implicit solvent
LAMMPS	Alpha	Released (alpha)	2 pair styles ported to GPU



Coulomb N-test rendszer

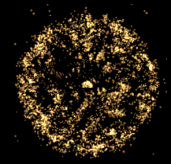
CUDA program

Kölcsönhatás:

$$v^{\text{Coulomb}}(r) = \frac{Q_1 Q_2}{4\pi\epsilon_0 r}$$

Numerikus instabilitás miatt regularizálás:

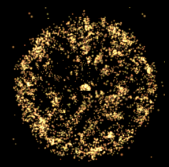
$$\frac{Q_1 Q_2}{4\pi\epsilon_0} \frac{1}{\sqrt{r^2 + r_0^2}}$$



Feladat: Coulomb erők kiszámítása

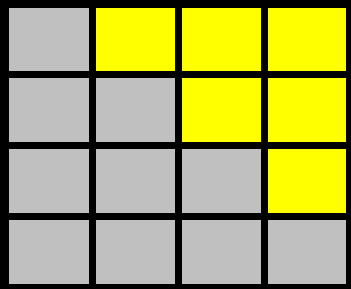
$$\vec{f}_i = Q_i \sum_j^{N_j} Q_j \frac{1}{((\vec{r}_i - \vec{r}_j)^2 + r_0^2)^{3/2}} \vec{r}_i \quad , \qquad i = 1..N_i$$

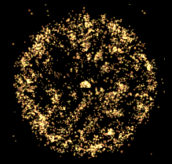
$$\textit{műveletek száma}: \quad N_i N_j \quad , \qquad \sim O(N^2)$$



CPU

```
for (i=0; i<n; i=i+1) {
  for (j=i+1; j<n; j=j+1) {
    dx0 = xi[i][0] - xj[j][0];
    dx1 = xi[i][1] - xj[j][1];
    dx2 = xi[i][2] - xj[j][2];
    yyy = (dx0*dx0 + dx1*dx1) +  (dx2*dx2+eps2) ;
    xxx =  q[i]*q[j] / ( sqrtf(yyy) * yyy ) ;
    ai[i][0] += xxx * dx0 ;
    ai[i][1] += xxx * dx1 ;
    ai[i][2] += xxx * dx2 ;
    ai[j][0] -= xxx * dx0 ;
    ai[j][1] -= xxx * dx1 ;
    ai[j][2] -= xxx * dx2 ;
  }
}
```





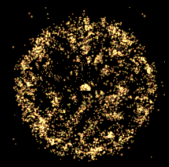
N-test szimuláció

Regularizált Coulomb párkölcsönhatás

Futásidők (N=65536)

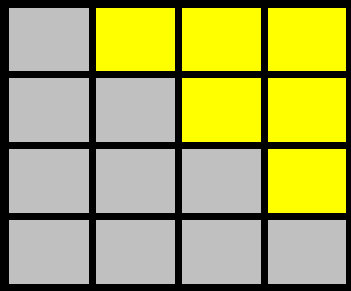
	1 CPU	PGI acc. (trivial)	CUDA kód
Idők [s]	52		

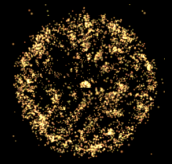
CPU: Intel Xeon E5520 @ 2.27GHz , 1 core, gcc



Portland accelerator (GPU)

```
#pragma acc region
{
  for (i=0; i<n; i=i+1) {
    for (j=i+1; j<n; j=j+1) {
      dx0 = xi[i][0] - xj[j][0];
      dx1 = xi[i][1] - xj[j][1];
      dx2 = xi[i][2] - xj[j][2];
      yyy = (dx0*dx0 + dx1*dx1) +  (dx2*dx2+eps2) ;
      xxx =  q[i]*q[j] / ( sqrtf(yyy) * yyy ) ;
      ai[i][0] += xxx * dx0 ;
      ai[i][1] += xxx * dx1 ;
      ai[i][2] += xxx * dx2 ;
      ai[j][0] -= xxx * dx0 ;
      ai[j][1] -= xxx * dx1 ;
      ai[j][2] -= xxx * dx2 ;
    }
  }
}
```

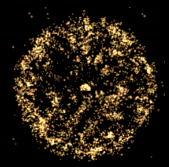




Portland accelerator (GPU)

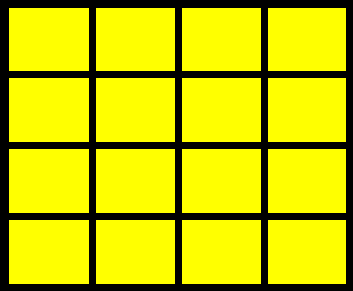
Fordítási üzenet:

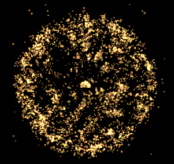
```
237, Accelerator restriction: size of the GPU copy
of an array depends on values computed in this loop
238, Accelerator restriction: size of the GPU copy
of 'm' is unknown
     Accelerator restriction: size of the GPU copy
of 'xi' is unknown
     Accelerator restriction: one or more arrays
have unknown size
     Loop not vectorized: data dependency
```



Portland accelerator (GPU)

```
#pragma acc region
{
  for (i=0; i<n; i=i+1) {
    for (j=0; j<n; j=j+1) {
      dx0 = xi[i][0] - xj[j][0];
      dx1 = xi[i][1] - xj[j][1];
      dx2 = xi[i][2] - xj[j][2];
      yyy = (dx0*dx0 + dx1*dx1) +  (dx2*dx2+eps2) ;
      xxx =  q[i]*q[j] / ( sqrtf(yyy) * yyy ) ;
      ai[i][0] += xxx * dx0 ;
      ai[i][1] += xxx * dx1 ;
      ai[i][2] += xxx * dx2 ;
      //  ai[j][0] -= xxx * dr[0] ;
      //  ai[j][1] -= xxx * dr[1] ;
      //  ai[j][2] -= xxx * dr[2] ;
    }
  }
}
```

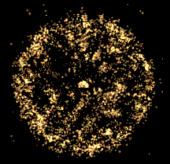




Portland accelerator (GPU)

Fordítási üzenet:

```
235, Generating copyin(xi[0:natom-1][0:2])
     Generating copyin(m[0:natom-1])
     Generating copyout(ai[0:natom-1][0:2])
     Generating compute capability 1.0 kernel
     Generating compute capability 1.3 kernel
237, Loop is parallelizable
     Accelerator kernel generated
    237, #pragma acc for parallel, vector(32)
         Non-stride-1 accesses for array 'xi'
         Non-stride-1 accesses for array 'ai'
244, Complex loop carried dependence of 'ai'
prevents parallelization
     Loop carried reuse of 'ai' prevents
parallelization
     Inner sequential loop scheduled on accelerator
```



N-test szimuláció

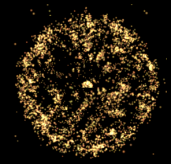
Regularizált Coulomb párkölcsönhatás

Futásidők (N=65536)

	1 CPU	PGI acc. (naív)	CUDA kód
Idők [s]	52	5 (~10x)	

CPU: Intel Xeon E5520 @ 2.27GHz , 1 core

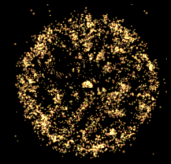
GPU: nVidia GTX280



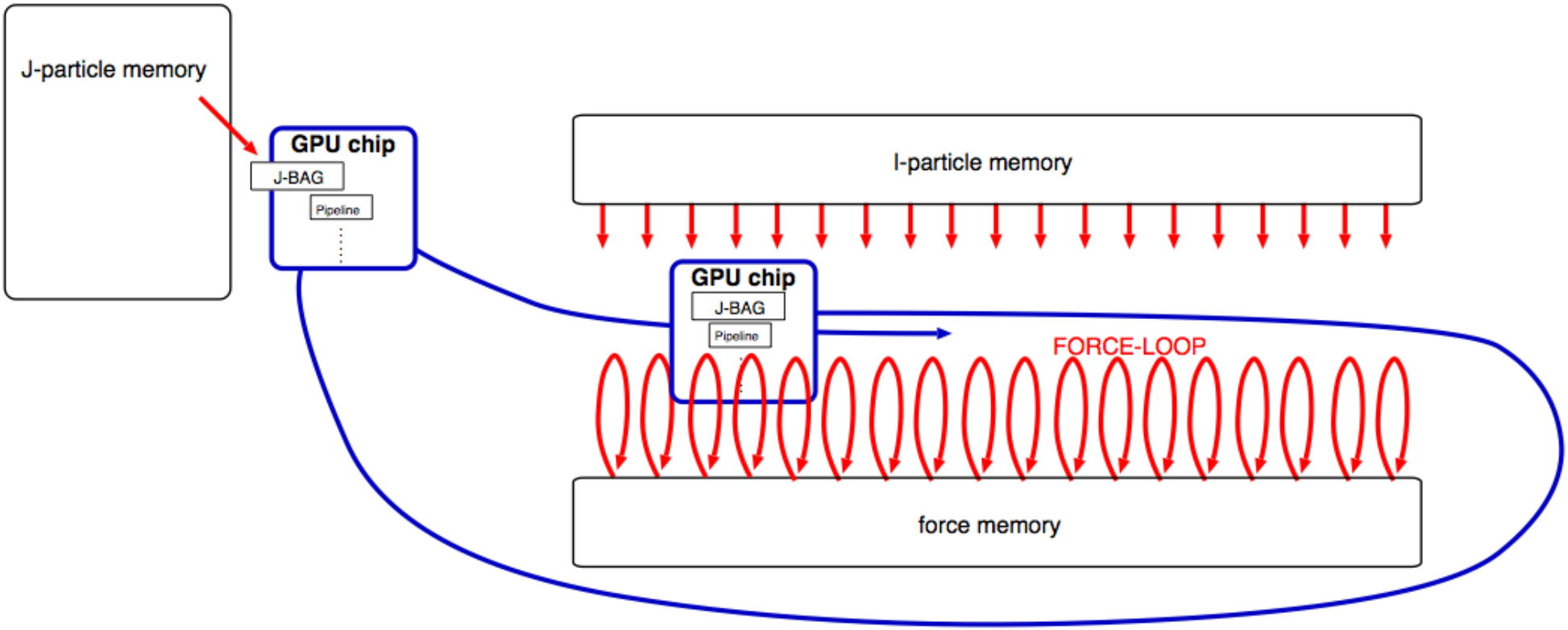
„Chamomile Scheme" (CUDA)

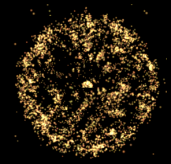
T. Hamada, T. Iitaka, http://arxiv.org/abs/astro-ph/0703100
http://progrape.jp/cs/
Gordon Bell díj (2009.nov.)

$$\vec{f}_i = Q_i \sum_j^{N_j} Q_j \frac{1}{((\vec{r}_i - \vec{r}_j)^2 + r_0^2)^{3/2}} \vec{r}_i \ , \qquad i = 1..N_i$$

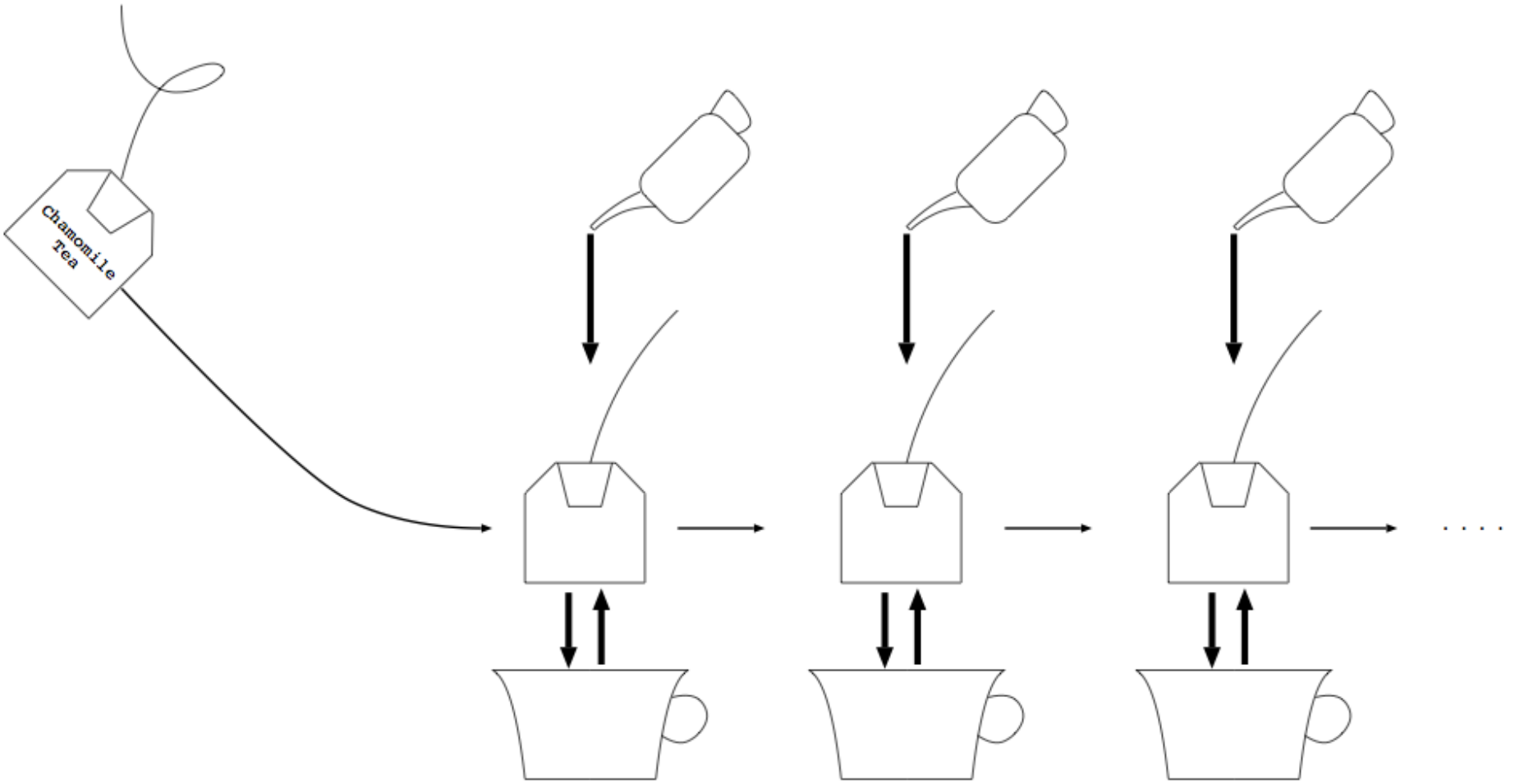


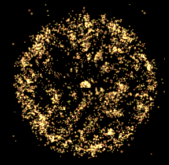
„Chamomile Scheme" (CUDA)



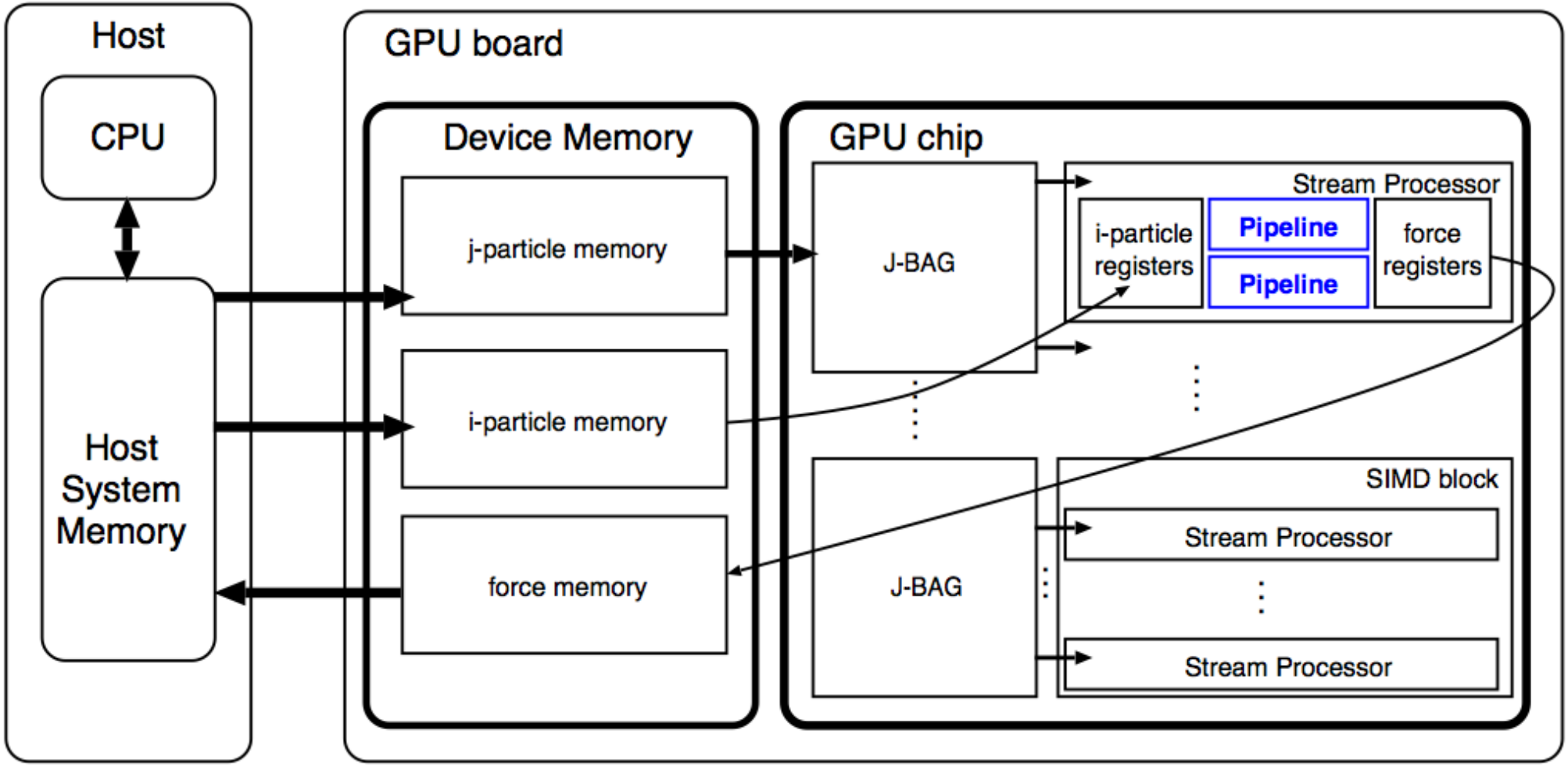


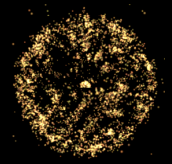
„Chamomile Scheme" (CUDA)





„Chamomile Scheme" (CUDA)





N-test szimuláció

Regularizált Coulomb párkölcsönhatás

Futásidők (N=65536)

	1 CPU	PGI acc. (naív)	CUDA kód
Idők [s]	52	5 (~10x)	0.27 (~200x)

CPU: Intel Xeon E5520 @ 2.27GHz , 1 core

GPU: nVidia GTX280