Many-core and Accelerator-based Computing for
Physics and Astronomy Applications

December, 2009 Stanford, CA

Molecular dynamics simulation of multi-phase flow on micro-scale using CUDA

Feiguo Chen, Wei Ge & Jinghai Li

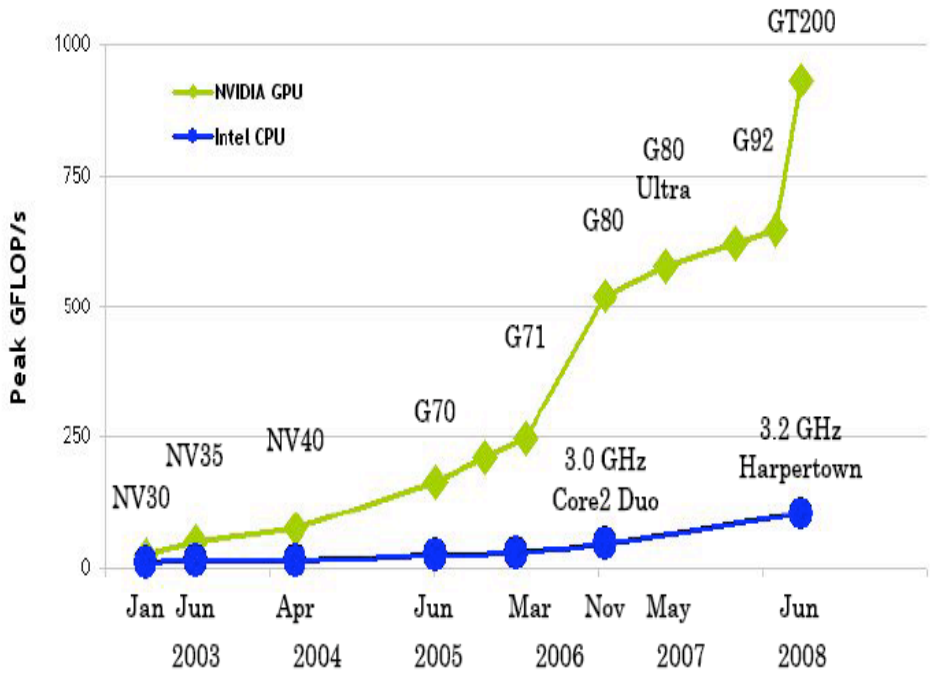
State Key Laboratory of Multi-Phase Complex System,
Institute of Process Engineering, Chinese Academy of Sciences

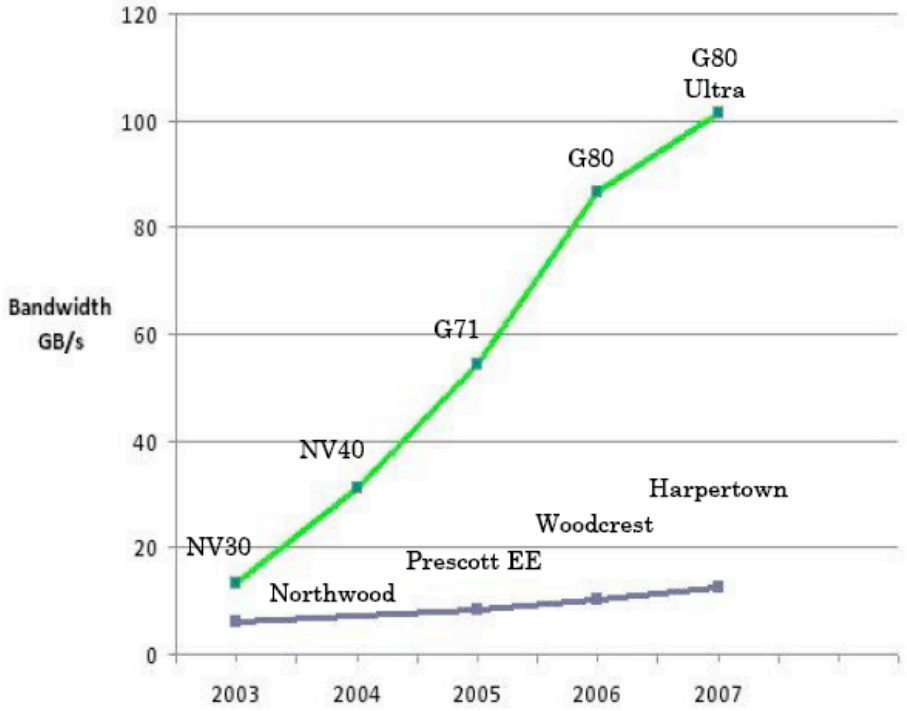
Outline

- Background: GPU computing & MD
- GPU-based discrete algorithm
- Applications
  - *lid-driven cavity flow*
  - *multiphase flow*
  - *vaporization of liquid*
- Conclusions

GPU computing

GPU: much higher performance

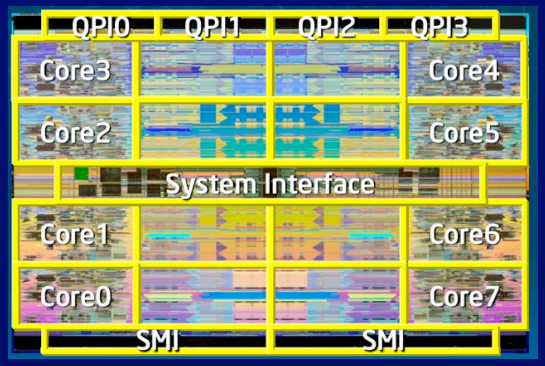




FLOPS and Memory Bandwidth for CPU and GPU

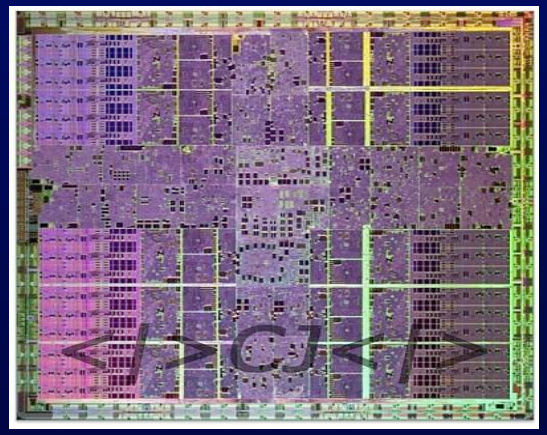
(NVIDIA, 2008)

GPU computing



Intel Nehalem CPU
8 cores
85 Gflops@2.67GHz

VS



nVIDIA G200 GPU
240 cores !!!
936 Gflops (SP)

chip die *(pictures from internet)*

GPU computing:

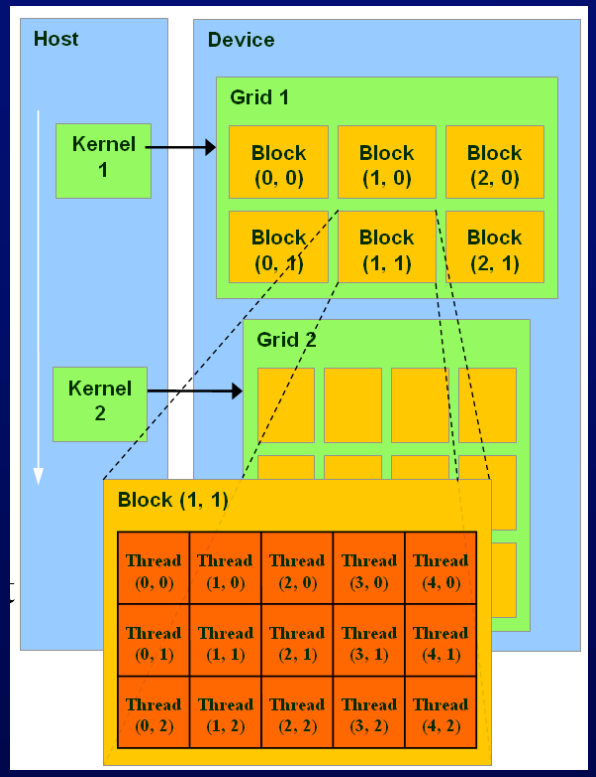
- high performance
- high efficiency
- low power consumption
- low cost

- easy to expand
- difficult to handle

CUDA ...

GPU computing

CUDA: Compute Unified Device Architecture



GPU programming model

(NVIDIA, 2007)

"CPU" painting robot

smile face

"GPU" painting robot

Mona Lisa

(videos from YouTube)

#1
CPU

#2
GPU

Discrete methods

microscopic scale

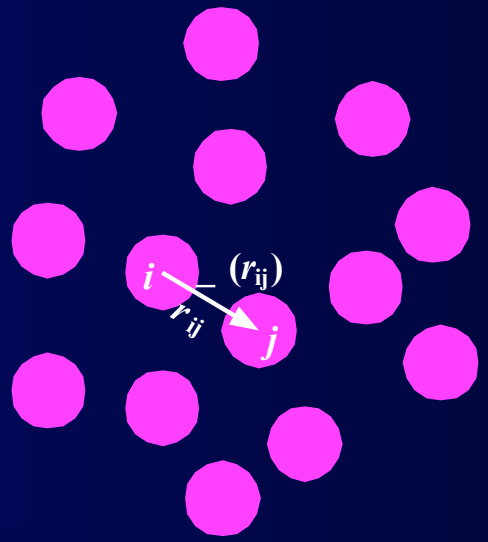
MD, DSMC, PPM ...

mesoscopic scale

LBM, DPD...

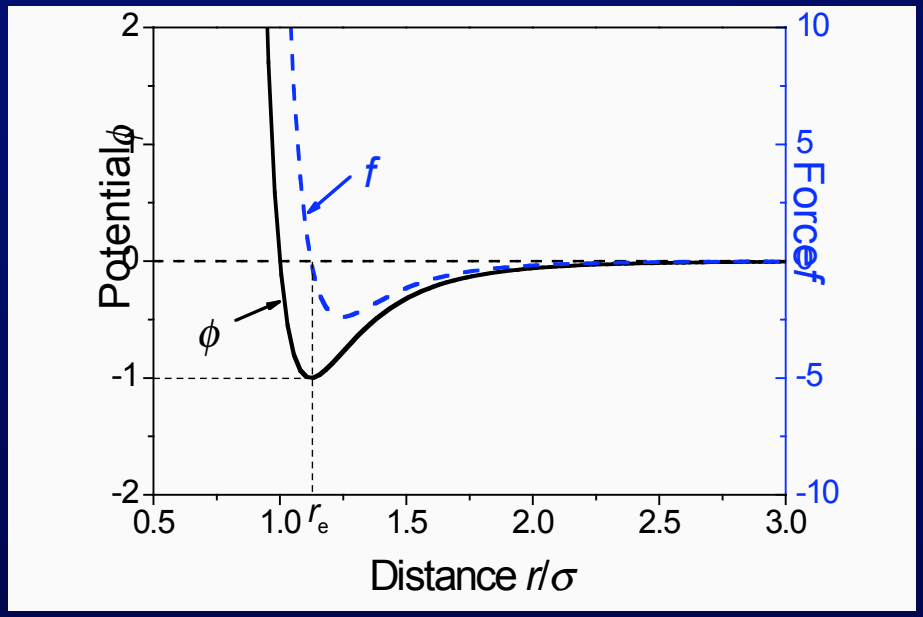
macroscopic scale

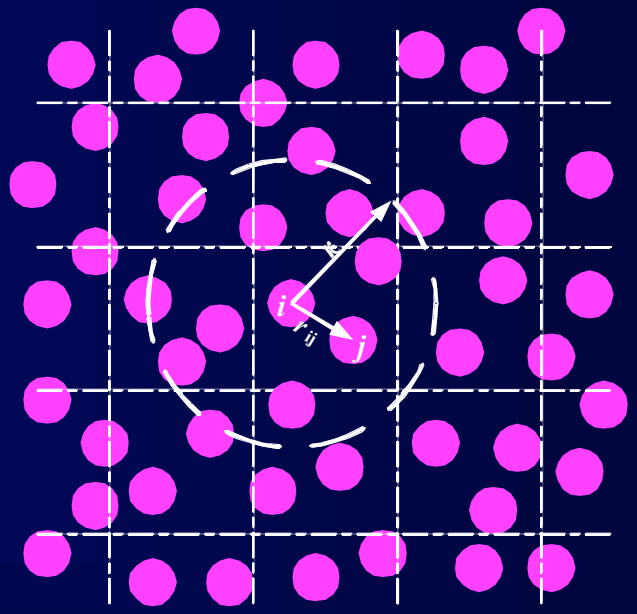
SPH, DEM, FPM, MaPPM ...



Discrete methods

Molecular dynamics





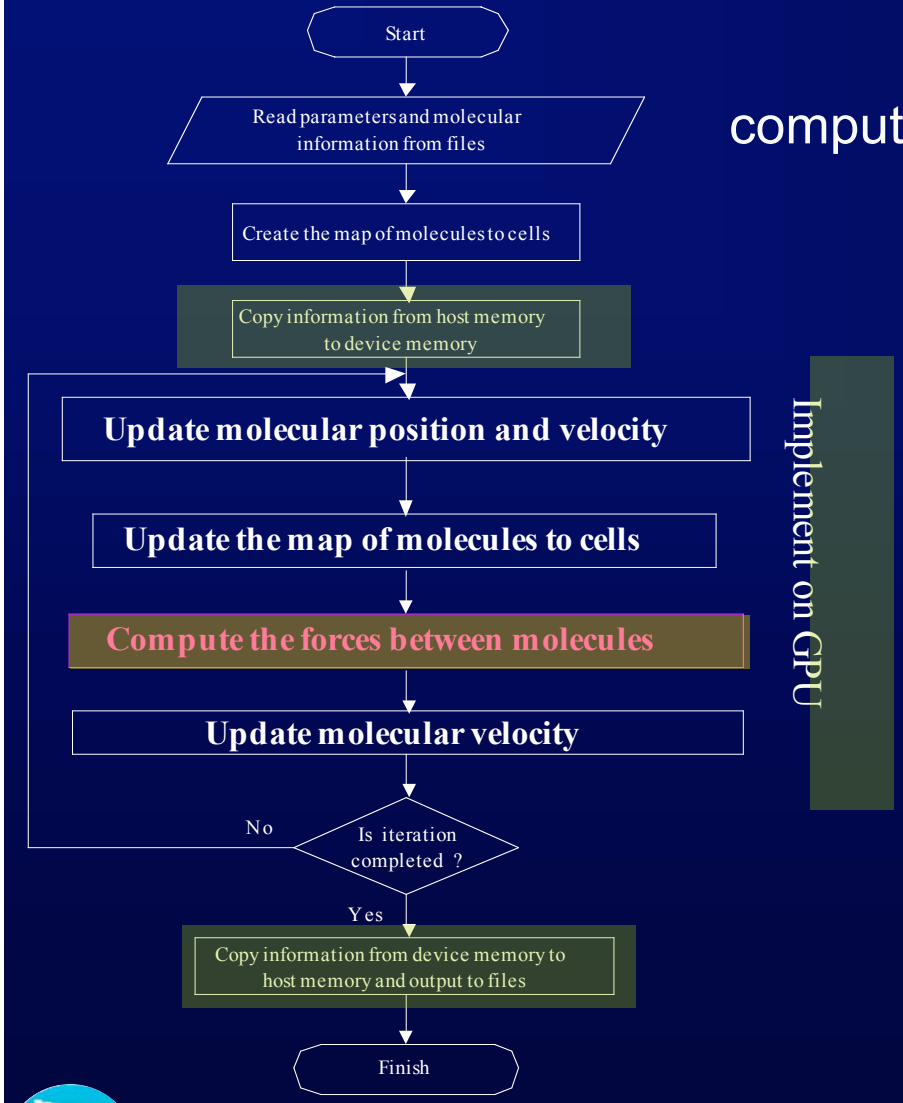
pair-molecular interaction, $f = -\nabla\phi$

short range
truncated interaction

saving computing time

interacting distance
cutoff scheme

GPU-based discrete algorithm

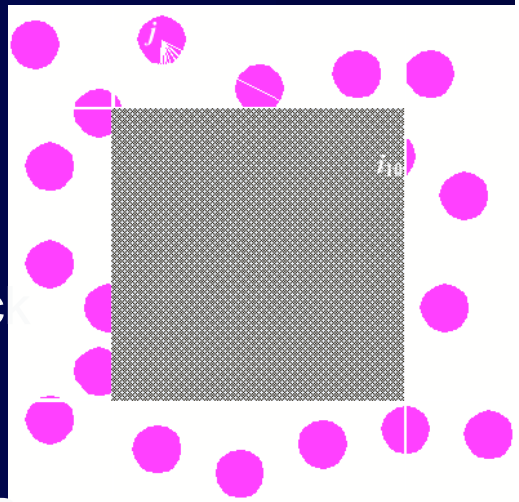


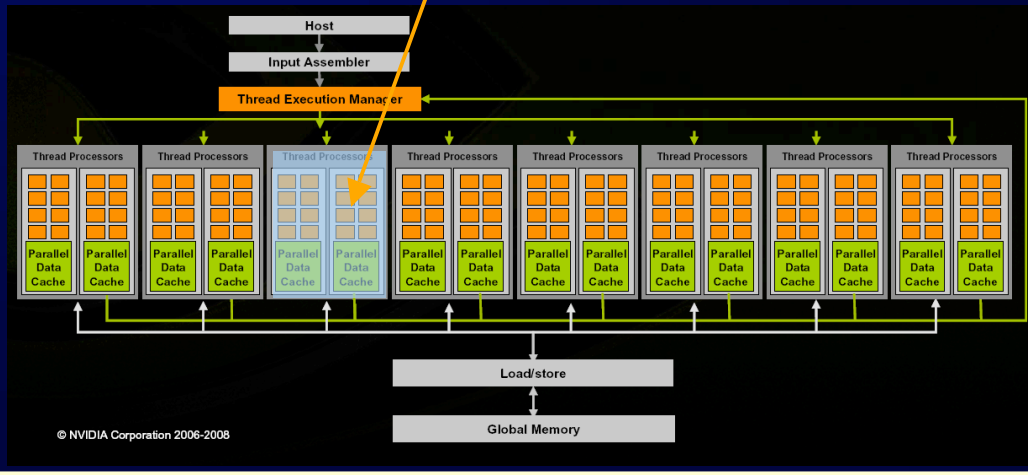
computing scheme:

linked cell

thread bloc

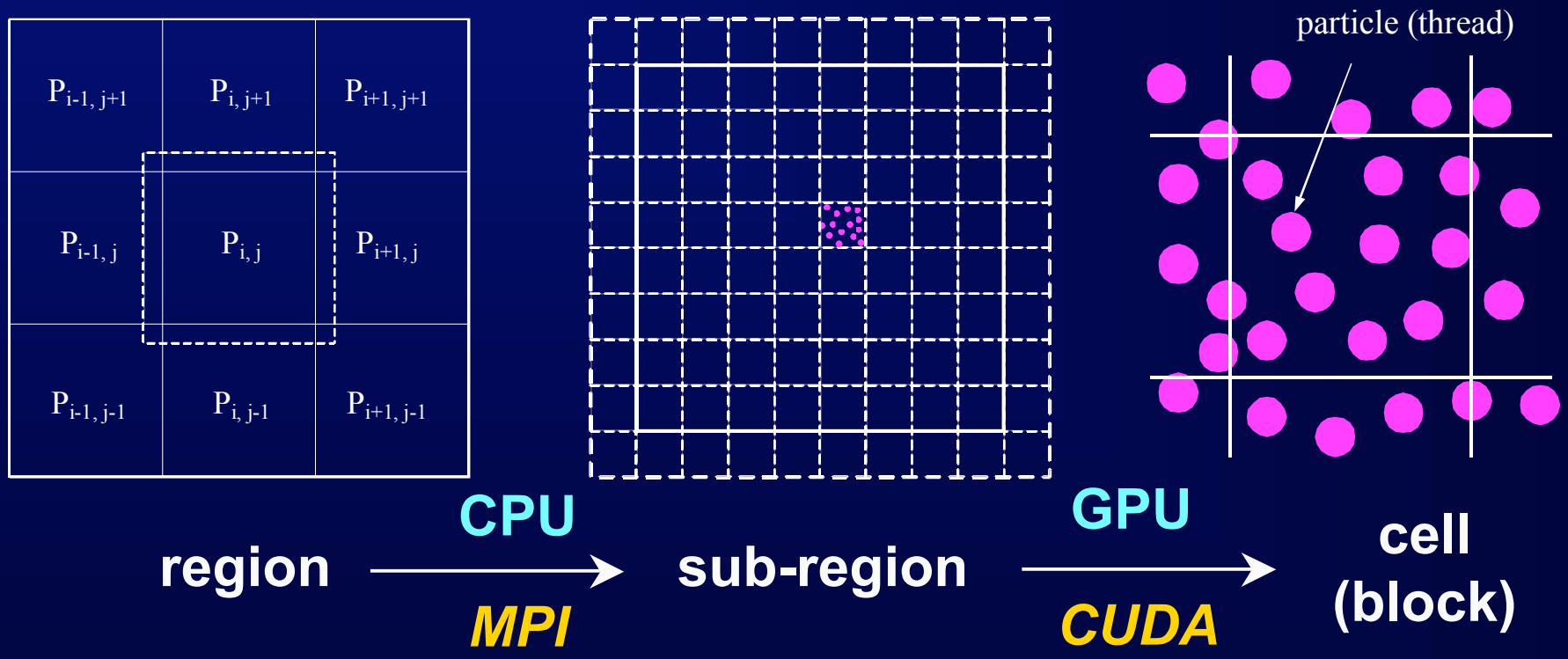
stream MP



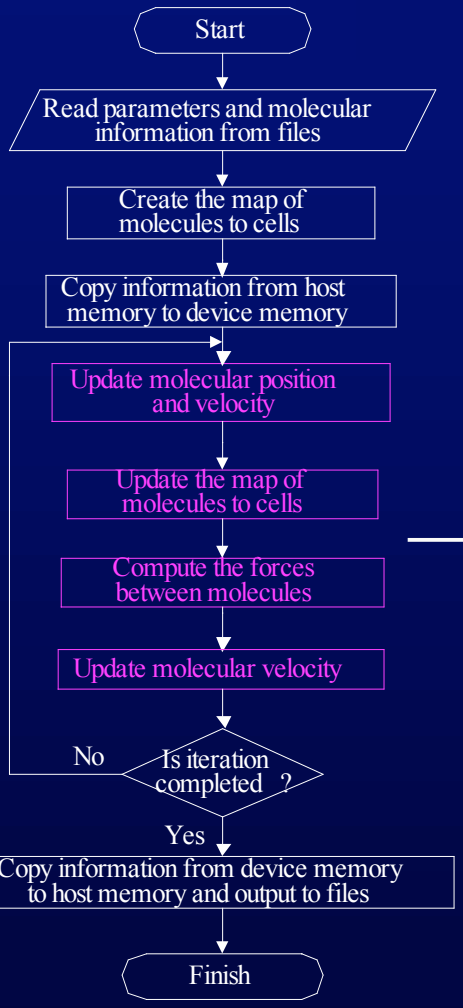


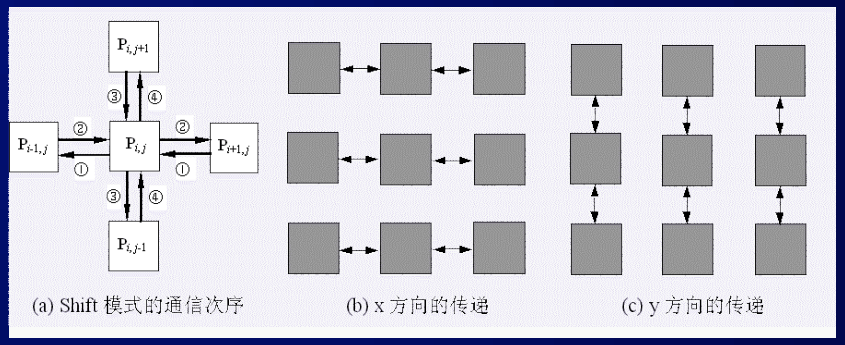
GPU-based discrete algorithm

multilevel parallel mode: inter-GPUs & inner-GPU

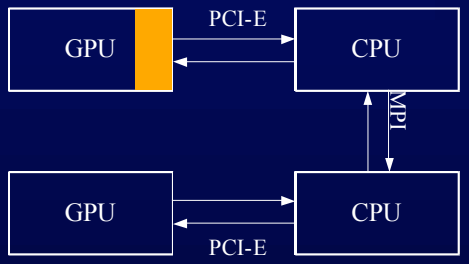


GPU-based discrete algorithm

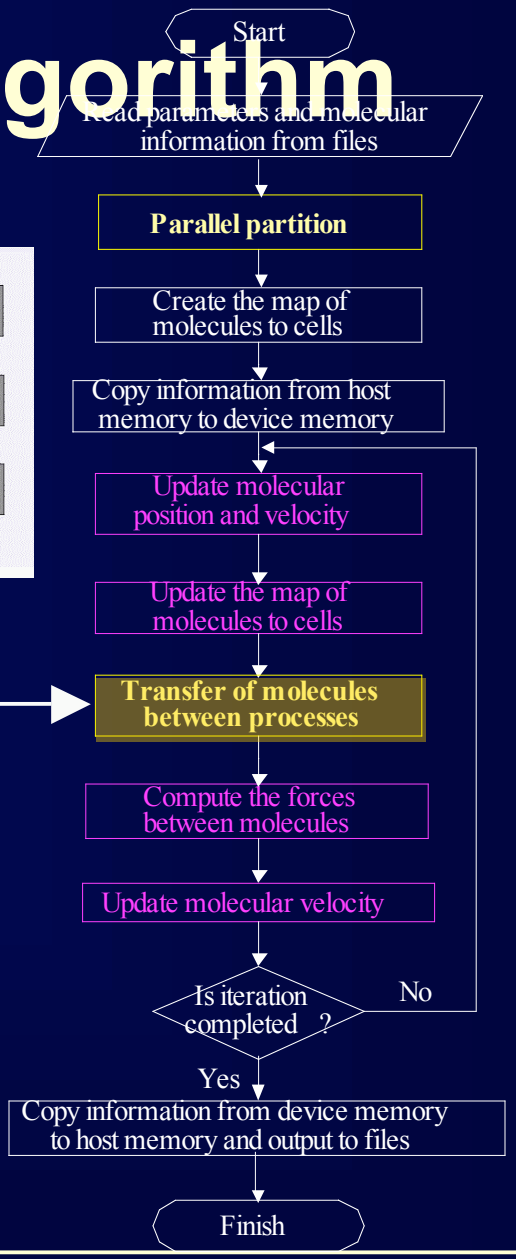




shift communication mode



GPU-CPU-CPU-GPU transferring



Performance

Tesla C870 GPU vs one core of Xeon E5430@2.66GHz CPU

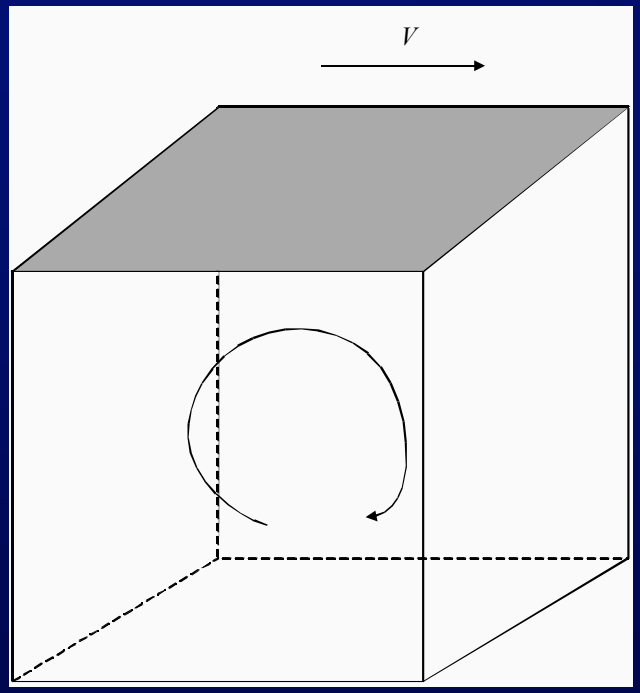
7.29×10^5 molecules/(CPU or GPU) in 100×100×100 region (n=0.8, T=1)

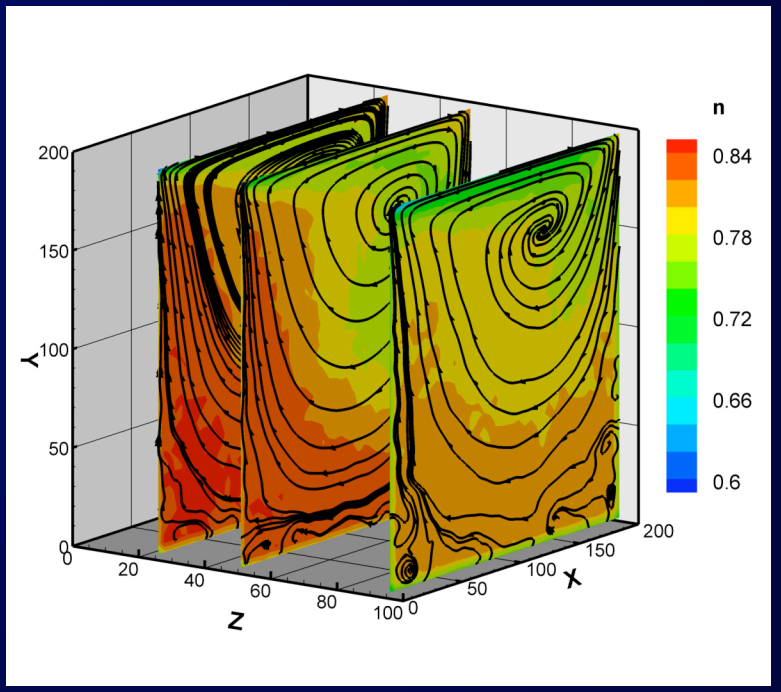
	Movement s/timestep	Mapping s/timestep	Force s/timestep	Comm. s/timestep	Total s/timestep	Perform. Gflops	Speedup
CPU	0.036	0.046	14.56		14.67	2.05	1
GPU	0.042	0.146	0.46		0.68	44.30	**21.6**
GPU+MPI	0.037	0.206	0.45	0.115*	0.81	37.19	18.1
CPU, force			14.56		29.07	2.07	1
GPU, force			0.46		0.48	65.49	**31.6**
CPU, r_c=20	0.074	0.106	1054.83		1055.09	1.82	1
GPU, r_c=20	0.042	0.144	16.57		16.79	114.83	**63.1**

* gigabit ethernet

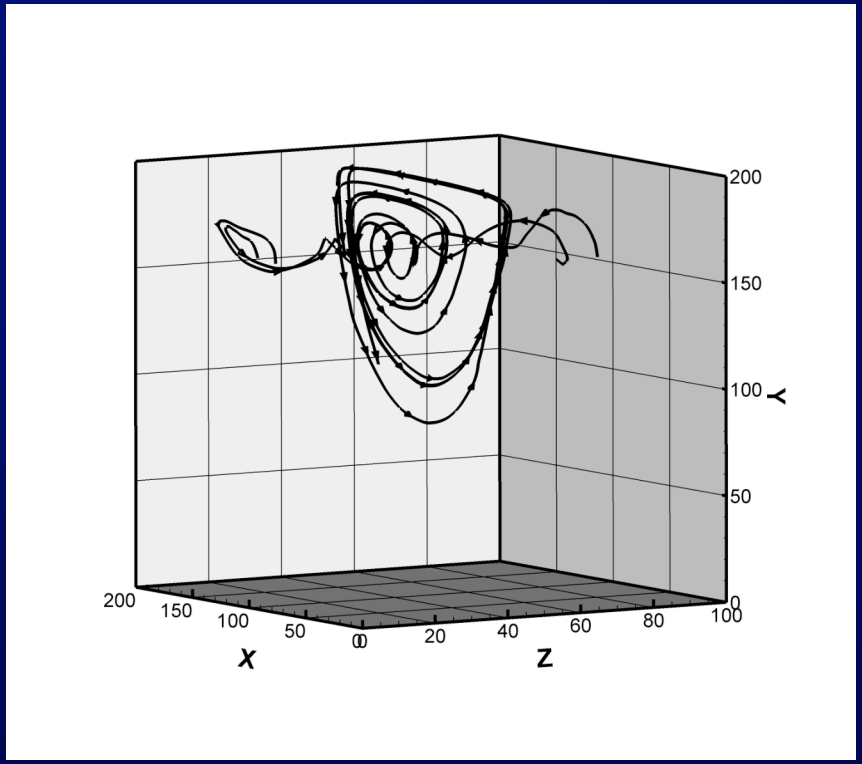
GPU achieves the speedup of **20~60**x over CPU !!

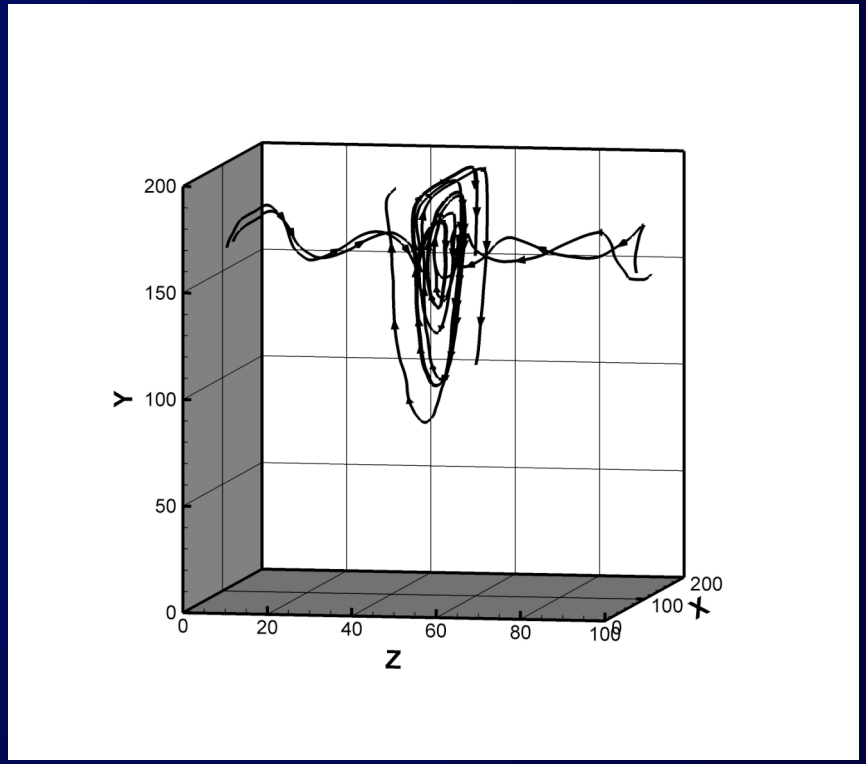
Application
-- lid-driven cavity flow



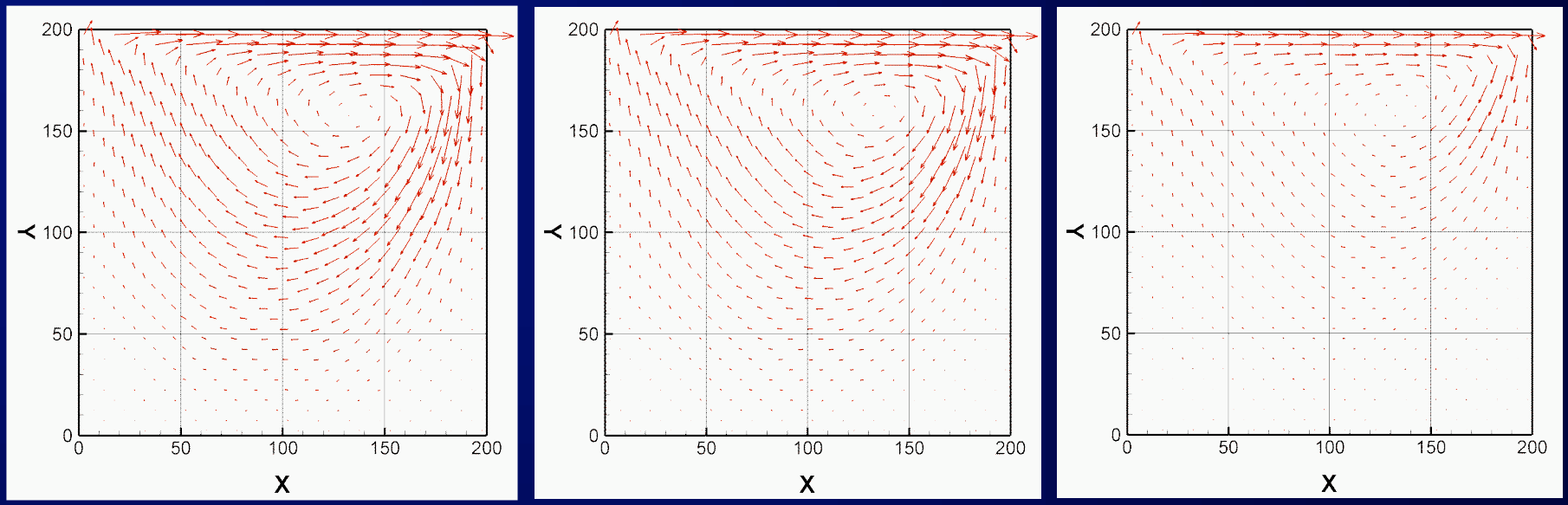


	ρ	T	μ	c	L×H×D	U	Re	Ma
dimensionless value	0.8	0.8	2.28	5.42	200×200×100	2	**140.4**	0.37
dimensional value	1350kg/m^3	96K	2.08×10^{-3}P	861m/s	68nm×68nm×34nm	318m/s	**140.4**	0.37





Three-dimensional structure of the primary eddy, as seen from different angles

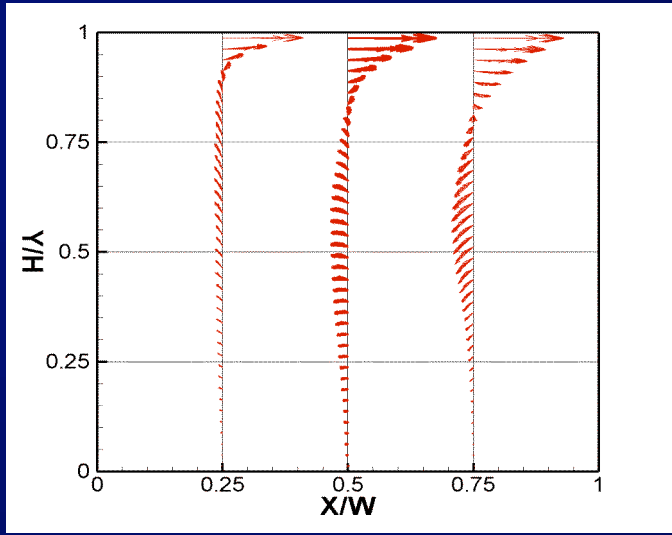


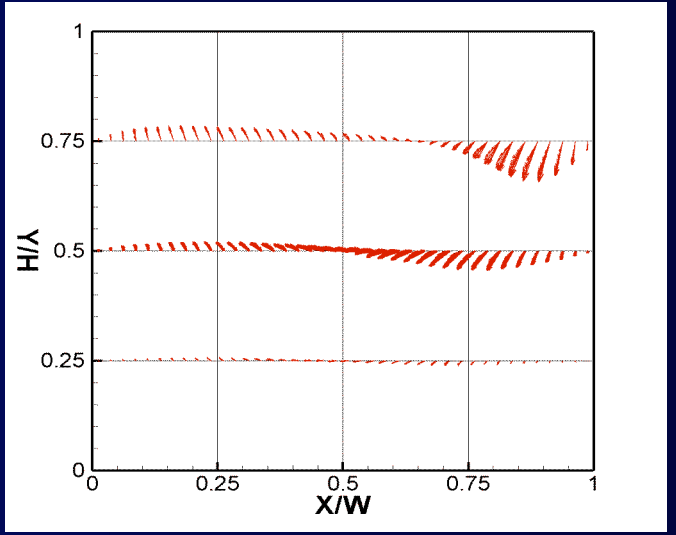
(a) *Z/D*=0.5 (b) *Z/D*=0.25 (c) *Z/D*=0.05

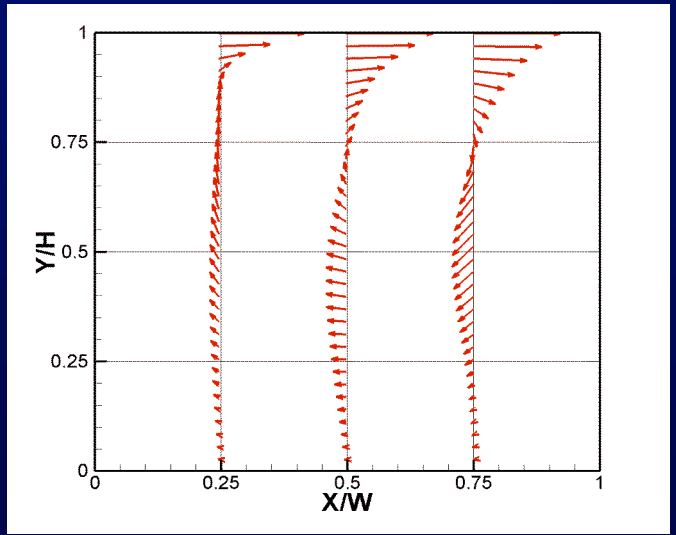
Velocity vector profiles at *Z/D*=0.5, 0.25, 0.05

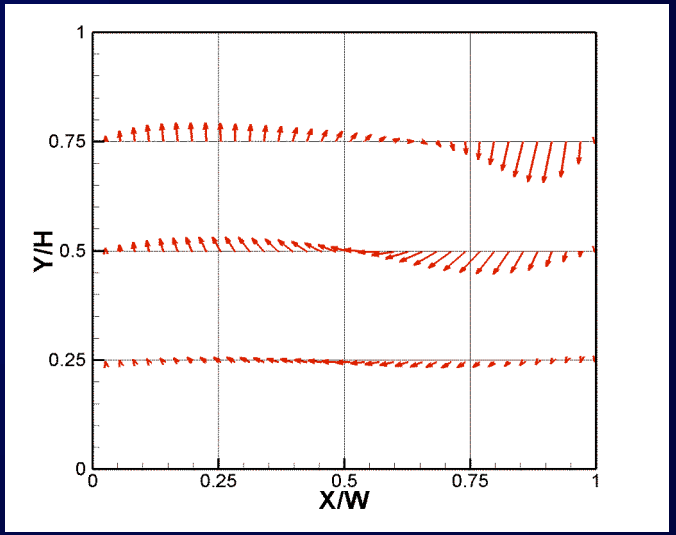
MD

CFD









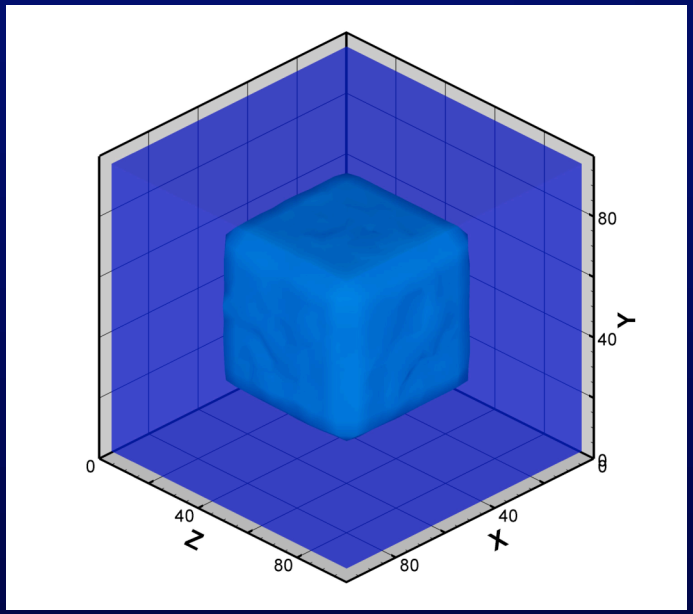
X/W=0.25, 0.5, 0.75

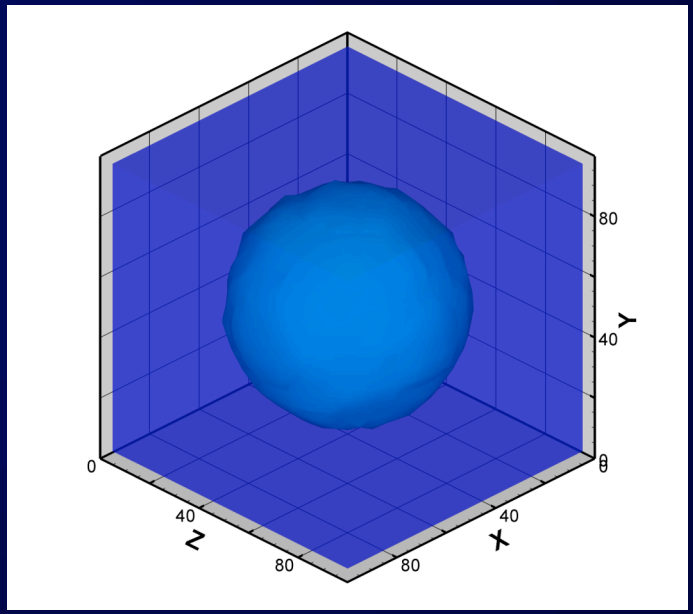
Y/H=0.25, 0.5, 0.75

Velocity vector profiles

Application
-- micro-scale multiphase flow

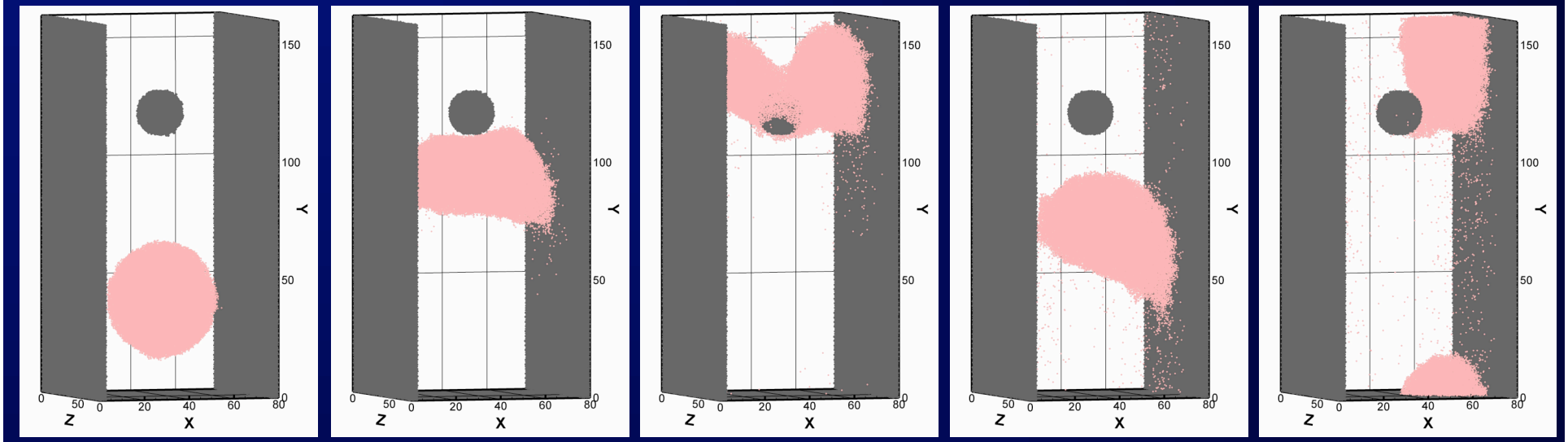
Multiphase MD simulation: gas-liquid relaxation





Initial cubic bubble (t=0) relaxes to (nearly) spherical shape at equilibrium (t=200)

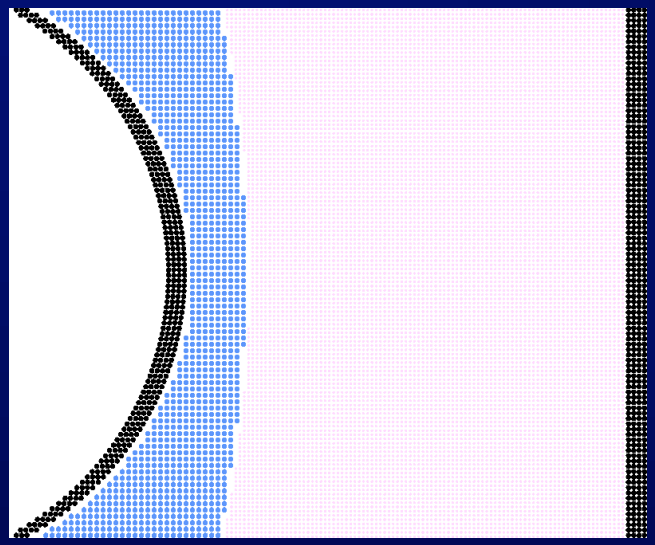
Multiphase MD simulation: particle-bubble interaction in liquid flow

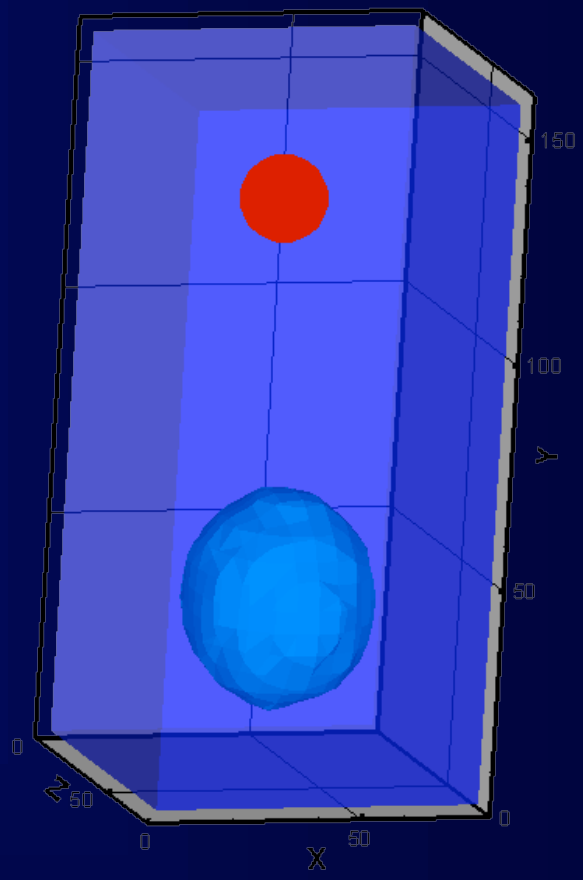


(a) t=0 (b) t=300 (c) t=500 (d) t=900 (e) t=1,250

Snapshots of gas-liuqid-solid MD simulation at different times

CPU→GPU: *from Angstroms to Microns*

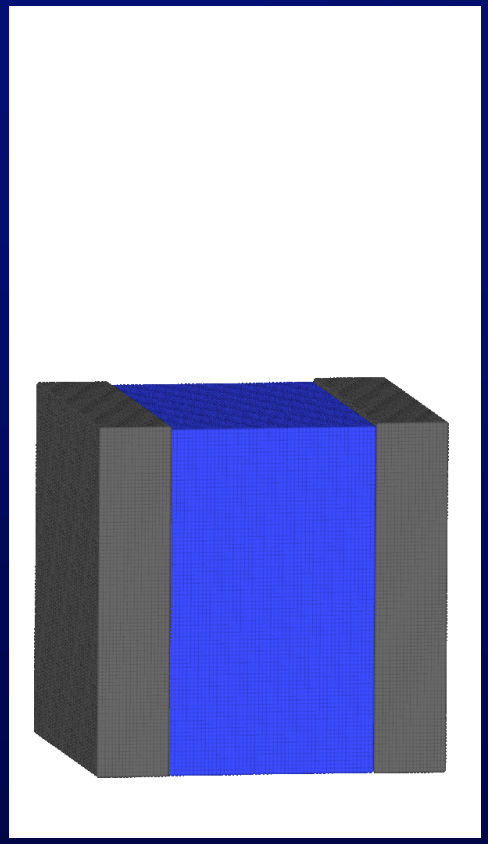




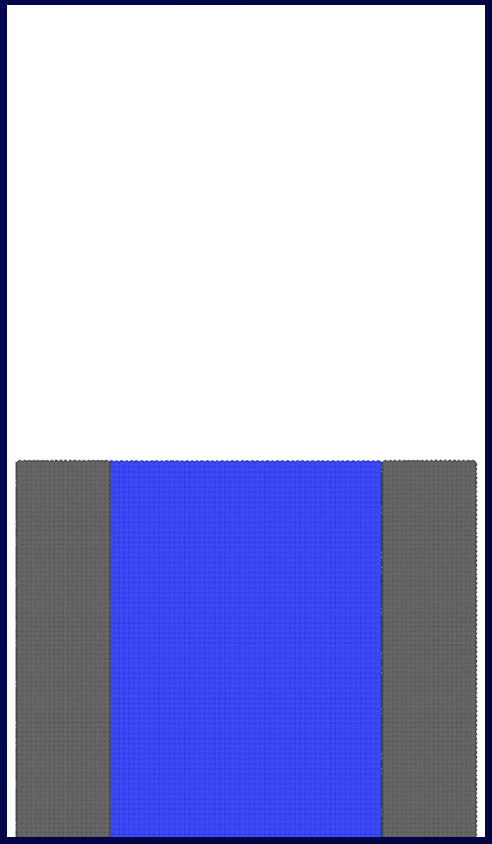
CPU: ~10^4 2D molecules nano-scale

GPUs: ~10^8 3D molecules (sub)micro-scale

Application
-- vaporization of liquid



vaporization with open boundary conditions



vaporization with rebounding boundary conditions

Conclusion & Prospect

- Molecular dynamics simulation has wide applications on micro-scale flow.
- GPU-based parallel computation may provide a feasible way to bring the method into practical use.
- More powerful and straightforward GPU computing is desired.

Fermi & CUDA 3.0 ?

Problems

CUDA memory allocation

Sample code:

```
float *d_M1, *d_M2, *d_M3;
…
cudaMalloc((void**) &d_M1, mem_size));
…
cudaMalloc((void**) &d_M2, mem_size));
…
cudaMalloc((void**) &d_M3, mem_size));
```

time cost

t_1 ~10^2 ms

v

v

t_2 ~1 ms

=

t_3

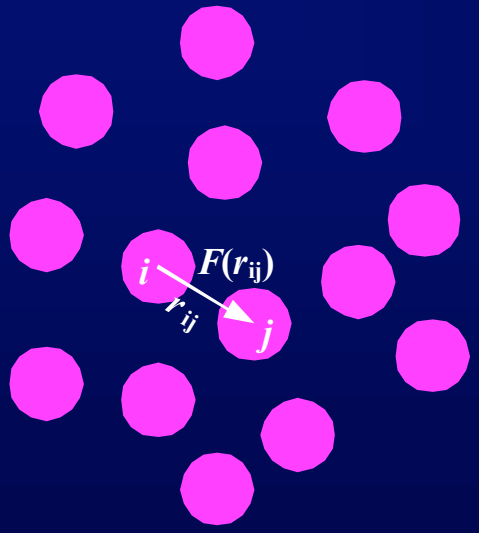
Why? Are there solutions to shorten this time?

The same problem exists in the case of cudaMallocHost() function.

Problems

Applying Newton's 3rd Law

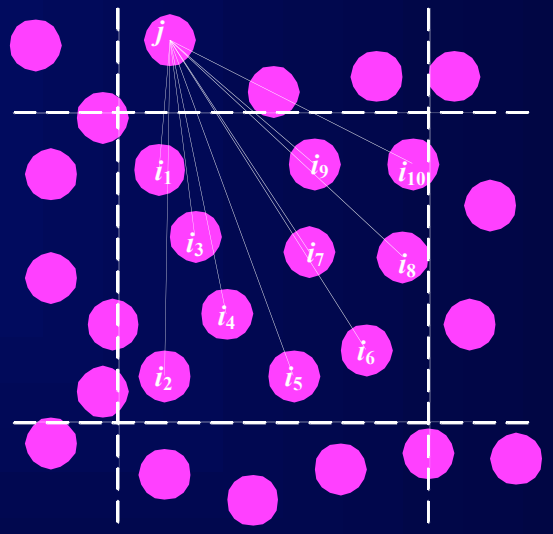
serial computing



$j \rightarrow i$ $\quad F_i = F_i + F_{ij}$

$i \rightarrow j$ $\quad F_j = F_j - F_{ij}$

parallel computing



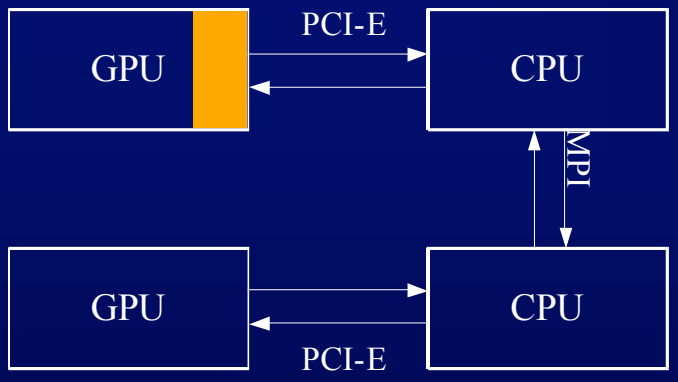
$F_{in} = F_{in} + F_{in,j}$ atomic op. ?

~~$F_j = F_j - F_{in,j}$~~ ?

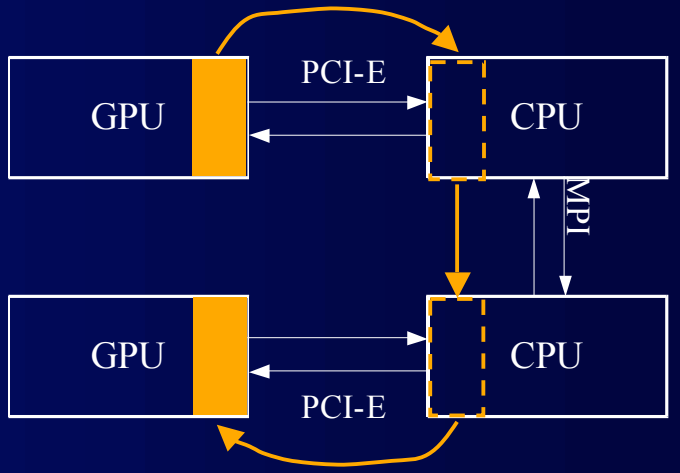
w/o 3rd law ?

Problems

GPU memory DMA ?



GPU-CPU-CPU-GPU transferring



DMA transferring

Thanks for your attention!