Fast Binding Site Mapping using GPUs and CUDA

Bharat Sukhwani **Martin C. Herbordt**

Computer Architecture and Automated Design Laboratory
Department of Electrical and Computer Engineering
Boston University
http://www.bu.edu/caadlab

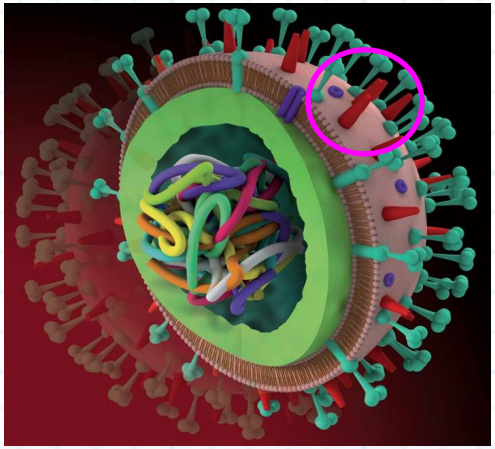
*** This work supported, in part, by the U.S. NIH/NCRR**

Why Bother?

Problem: *Combat the bird flu virus*

Method: *Inhibit its function by "gumming up"*
Neuraminidase, a surface protein, with an inhibitor

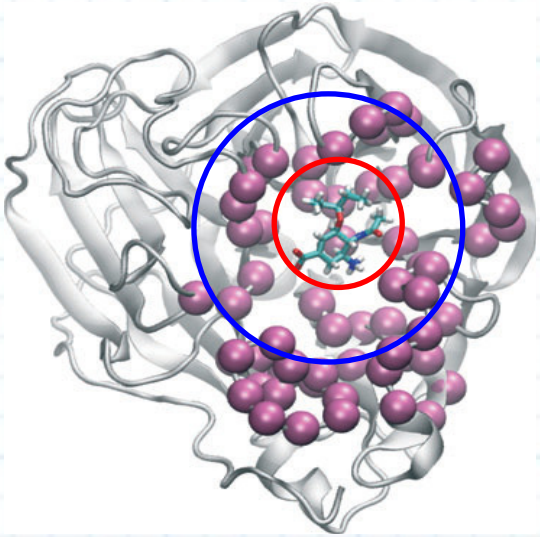
- Neuraminidase helps release progeny viruses from the cell.



Procedure*:

- Search protein surface for likely sites

- Find a molecule that binds there (and only there)



Binding site mapping:

- Very compute intensive: Usually run on clusters

- GPU based desktop alternative

*Landon, et al. Chem. Biol. Drug Des 2008
#From ***New Scientist*** www.newscientist.com/channel/health/bird-flu

Outline

- Overview of Binding Site Mapping
  - *Rigid Docking*
  - *Energy Minimization*
- Overview of NVIDIA GPUs / CUDA
- Rigid Docking on GPU
- Energy Minimization on GPU
- Results

Binding Site Mapping

Purpose: Identification of hot spots

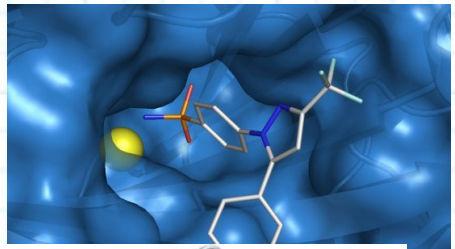
Significance: Very effective for drug-discovery

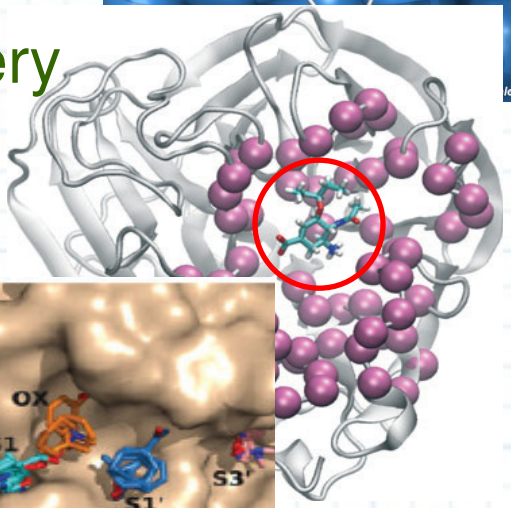
Rationale:

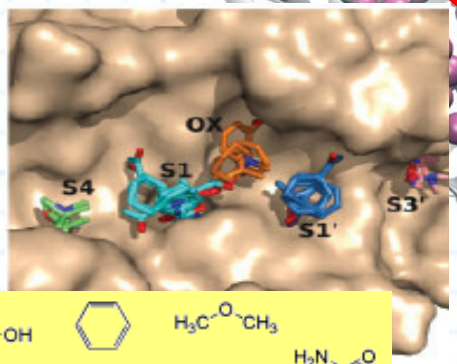
- *Hot spots are major contributors to the binding energy*
- *They bind a large variety of small molecules*

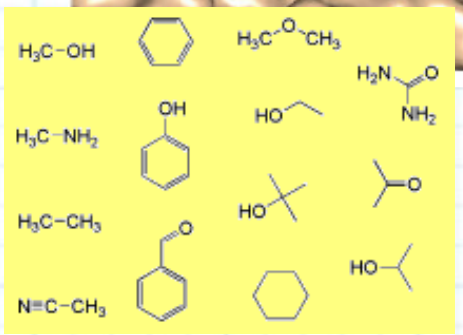
Process: Docking small probes

- *Rigid Docking*
- *Energy Minimization*



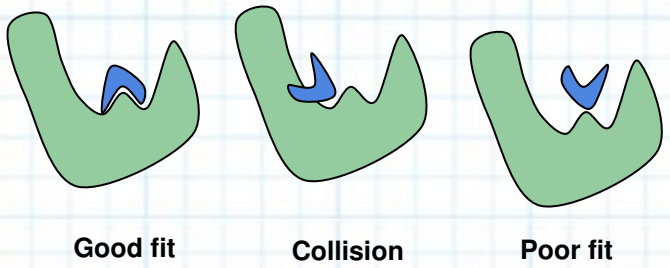




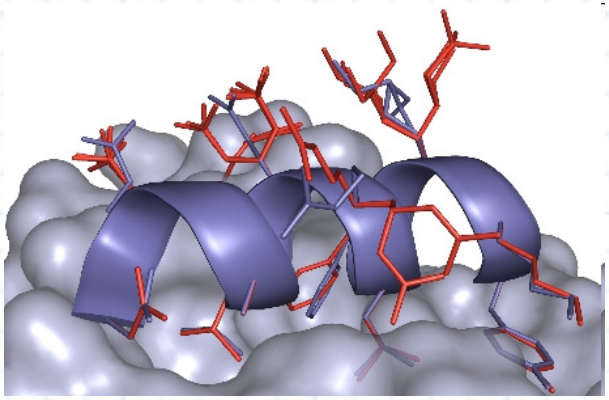


Mapping: Two Step Process

- Rigid Docking of Probes into Protein
  - *Grid-based computation*
  - *Exhaustive 6D search*
  - *Find an approximate conformation*

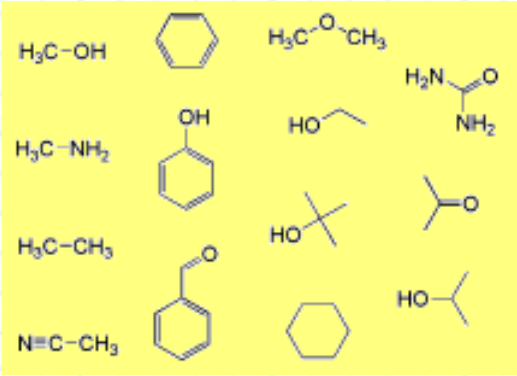


- Local refinement – Energy Minimization
  - *Model the flexibility in the side-chains*



FTMap*

- 16 small molecule probes
- Dock each probes into the protein
  - *500 rotations – 10^6 translations per rotation*
  - *30 minutes on a single CPU*
- Energy minimize 2000 conformations per protein-probe complex
  - *Up to 30 seconds per conformation*
  - *16 hours per probe!*

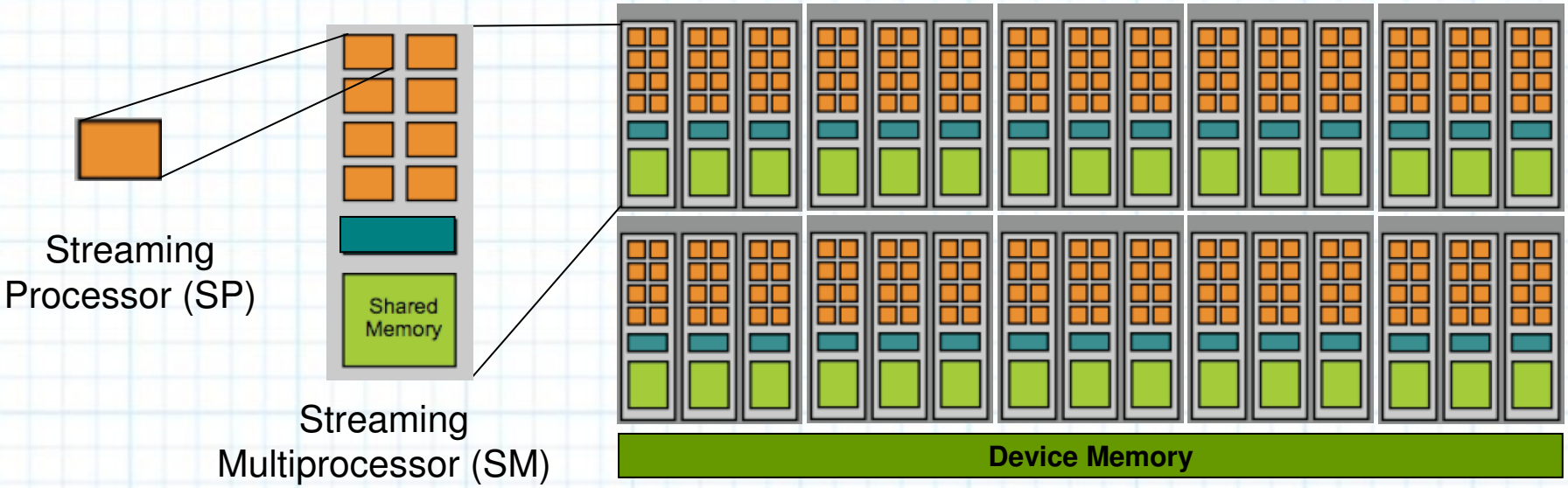


* Brenke R, Kozakov D, Chuang G-Y, Beglov D, Mattos C, and Vajda S. Fragment-based identification of druggable "hot spots" of proteins using Fourier domain correlation, Bioinformatics.

Outline

- Overview of Binding Site Mapping
  - *Rigid Docking*
  - *Energy Minimization*
- Overview of NVIDIA GPUs / CUDA
- Rigid Docking on GPU
- Energy Minimization on GPU
- Results

NVIDIA GPU Architecture



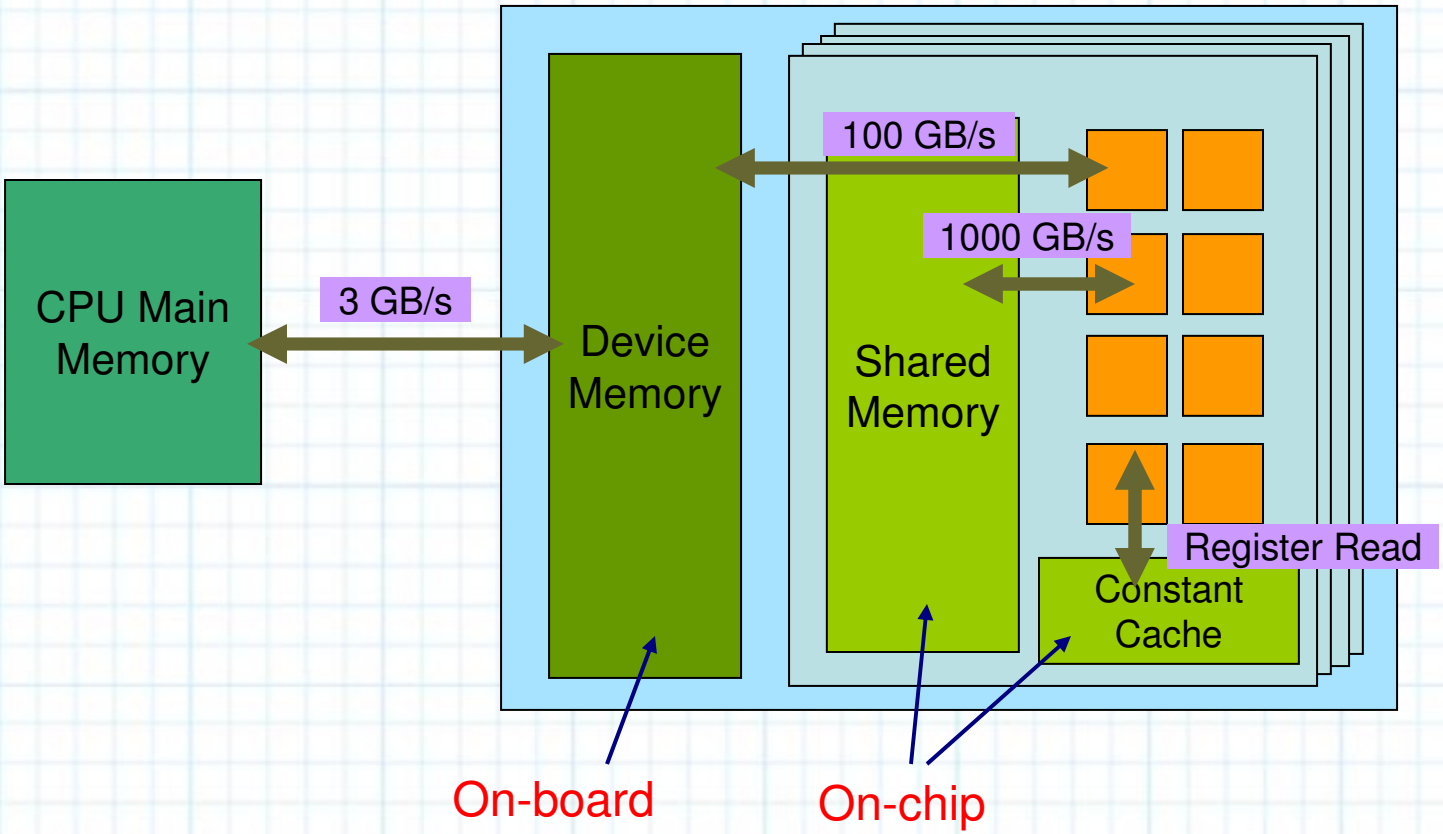
8 SPs per SM

NVIDIA Tesla C1060 Architecture

- *30 Multiprocessors*
- *240 Processor cores*
- *4 GB Device memory*
- *1.3 GHz Clock*

* Source: NVIDIA Corporation

Memory Hierarchy



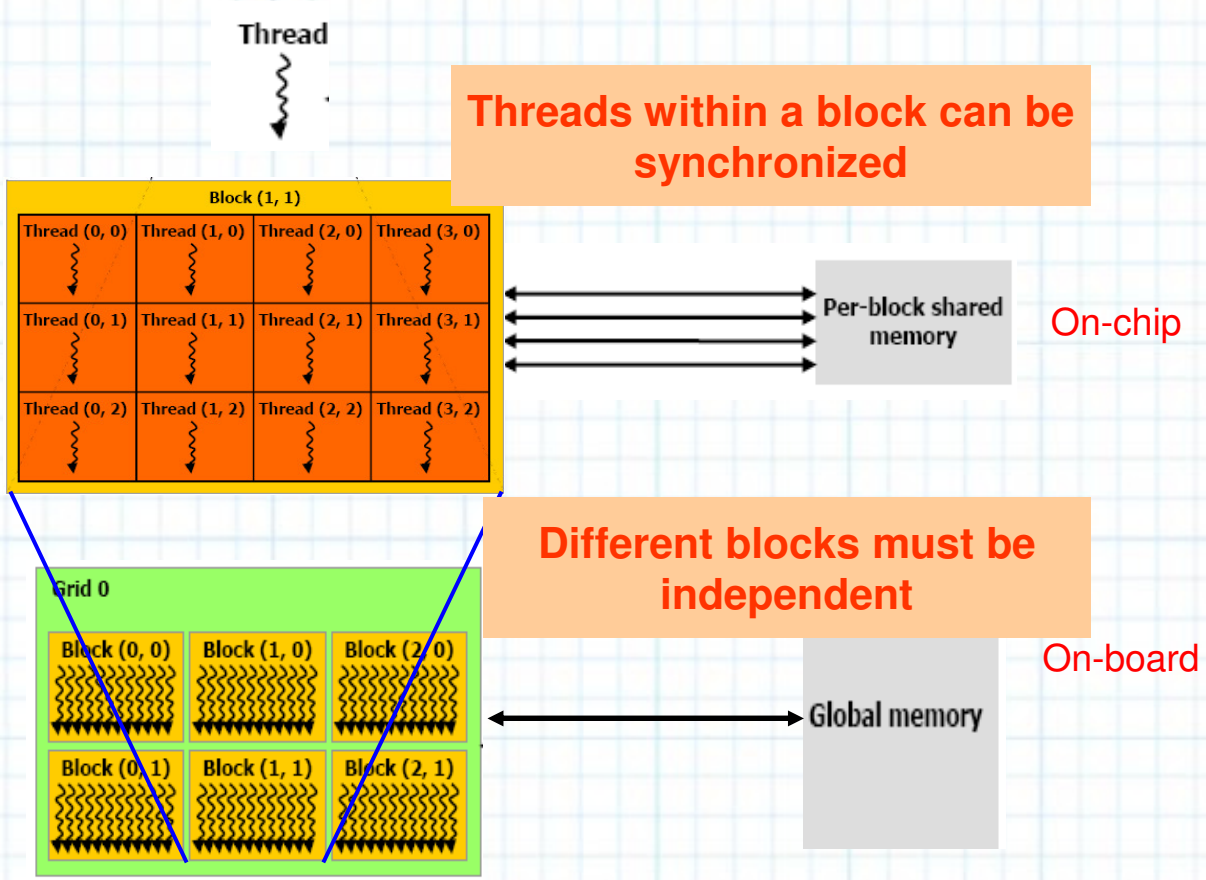
* Source: NVIDIA Corporation

CUDA Programming Model

Thread

Block of Threads

Grid of Blocks



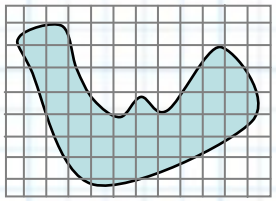
* Source: NVIDIA Corporation

Outline

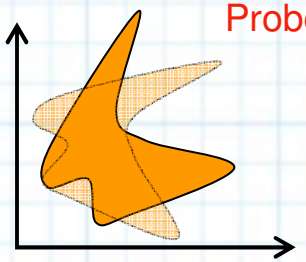
- Overview of Binding Site Mapping
  - *Rigid Docking*
  - *Energy Minimization*
- Overview of NVIDIA GPUs / CUDA
- Rigid Docking on GPU
- Energy Minimization on GPU
- Results

Rigid Docking: Procedure

Protein

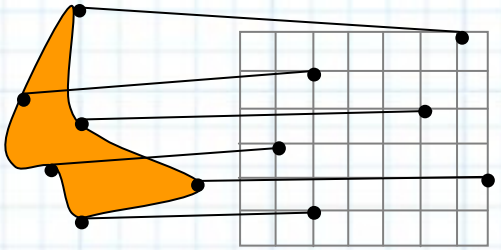


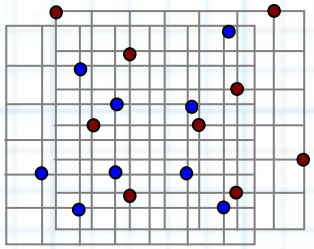
Probe



Rotation

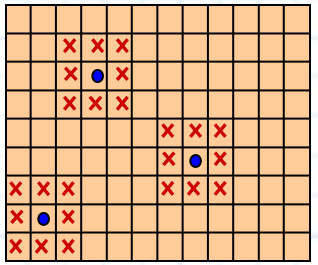
Grid Assignment





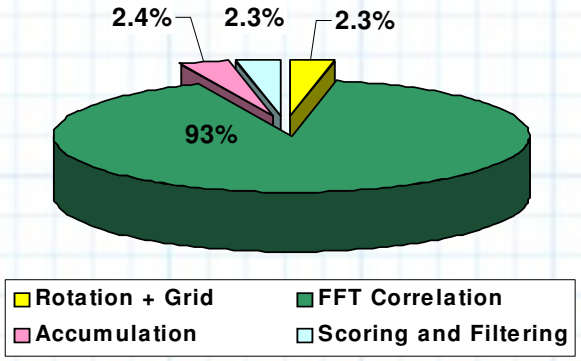
Pose Score: 3D FFT Correlation

Scoring and Filtering



PIPER Rigid Docking Program

- Structural Bioinformatics lab at BU
- Complex energy functions
- Top scorer in CAPRI* challenge



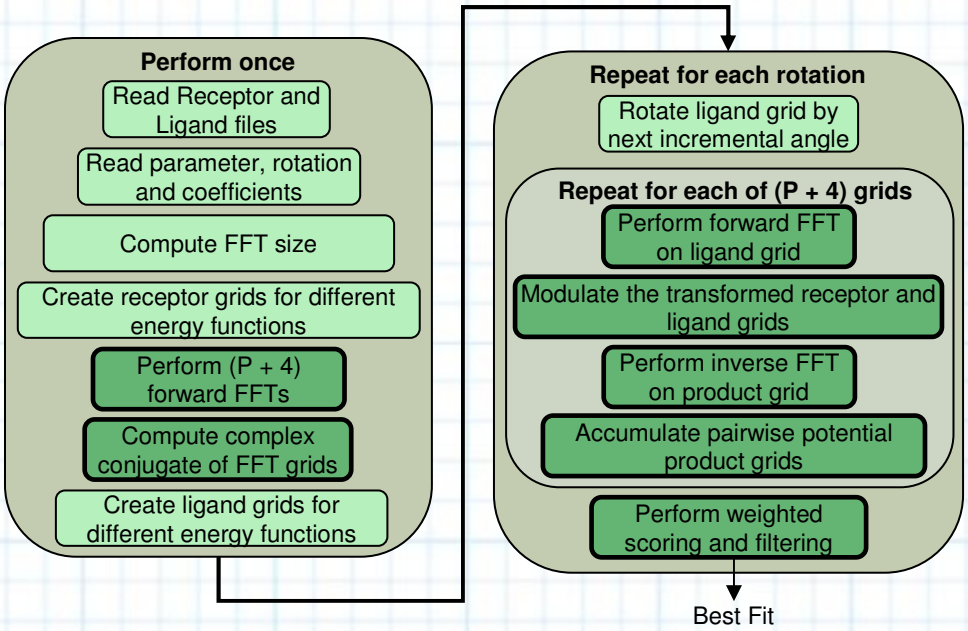
$$E = E_{shape} + w_2 E_{elec} + w_3 E_{desol}$$

$$E_{shape} = E_{attr} + w_1 E_{repul}$$

$$E_{elec} = E_{born} + E_{coulomb}$$

$$E_{desol} = \sum_{k=0}^{P-1} E_{pairpot\_k}$$

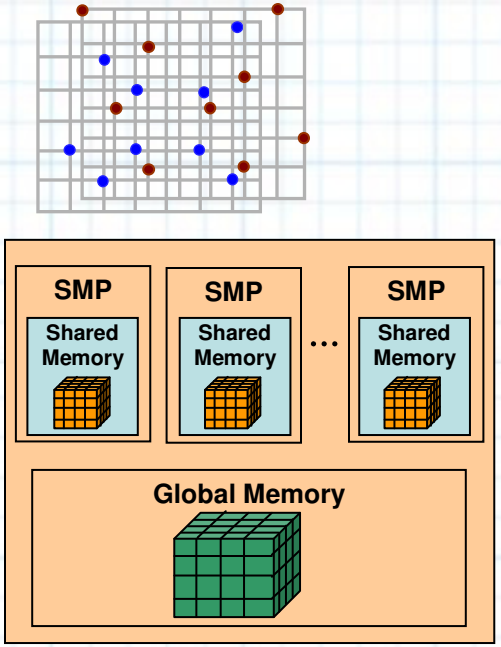
Up to 22 FFT correlations are required

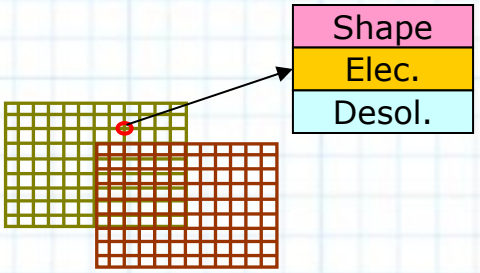


* Janin, J., Henrick, K., Moult, J., Eyck, L., Sternberg, M., Vajda, S., Vakser, I., and Wodak, S. CAPRI: A critical assessment of predicted interactions. Proteins, 52 (2003), 2-9

Rigid Docking on GPUs - Correlation

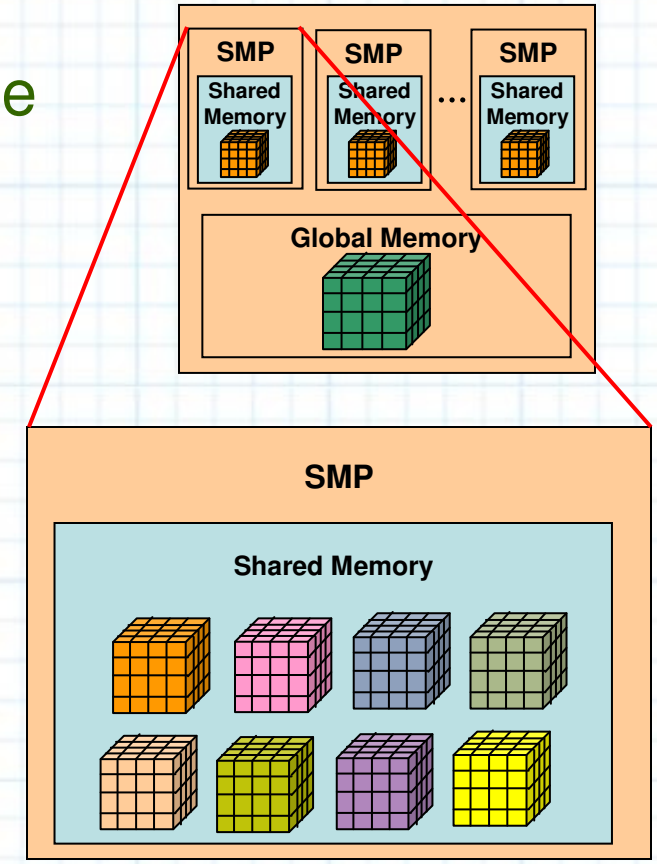
- Direct Correlation (better than FFT!)
  - *For small grid sizes*
  - *Replaces FFT, voxel-voxel summation, IFFT*
- Each multiprocessor accesses both the grids
  - *Protein grid on the global memory*
  - *Probe grid duplicated on shared memories*
- Multiple correlations together
  - *Voxel represents multiple energy functions*





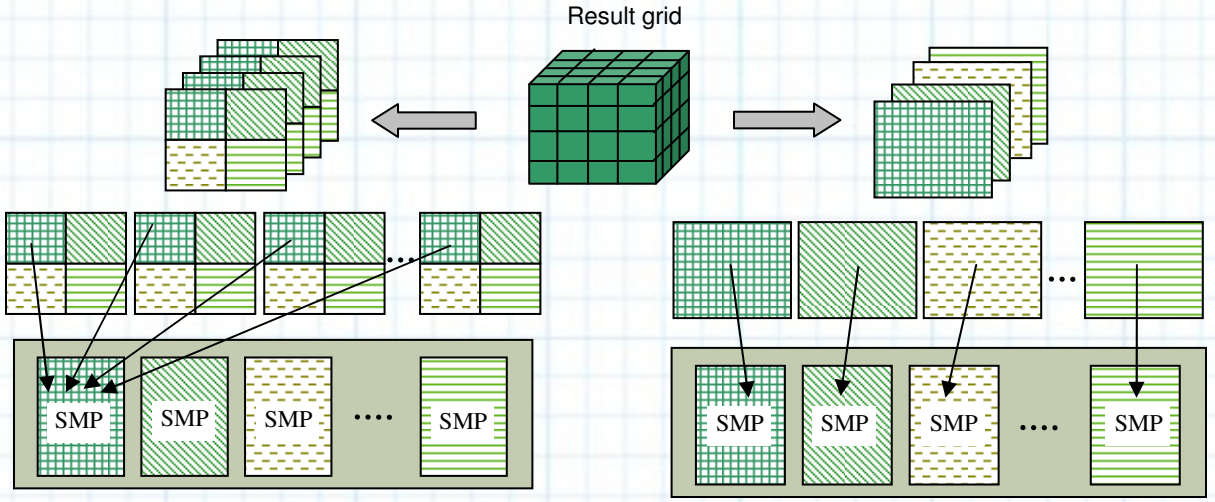
Direct Correlation on GPUs

- Shared memory limits the probe size
  - *With 8 correlations – 8 cubed*
  - *Probe grids are typically 4 cubed*
- Multiple rotations together
  - *8 rotations*
  - *Effectively loop-unrolling*
  - *Multiple computations per global memory fetch*
  - *2.7x additional performance improvement*



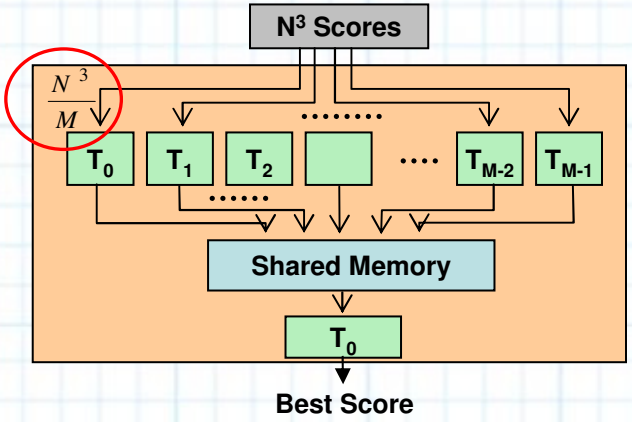
Direct Correlation on GPUs

- Distribution of work among threads / blocks
  - *Scheme 1: Entire 2D-plane to a thread block*
  - *Scheme 2: Part of the 2D-plane to a thread block*
  - *Both yield similar results*

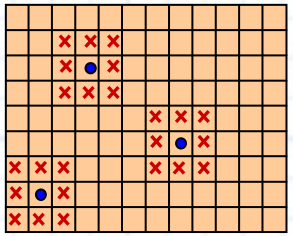


Scoring and Filtering on GPUs

- Score Computation
  - *Divide work among different threads*
  - *Sync and Serialize to find the best-of-the-best*
  - *Only one multiprocessor utilized*



- Flagging for exclusion



  - *Serial code – Exclusion bit-vector*

(N³ entries)

1	1	0	0	0	···	1	0

  - *GPU Solution 1 – Exclusion index array*

(100 entries)

4	5	16	28	···	45

  - *GPU Solution 2 – Exclusion bit-vector on GPU global memory*

(N³ entries)

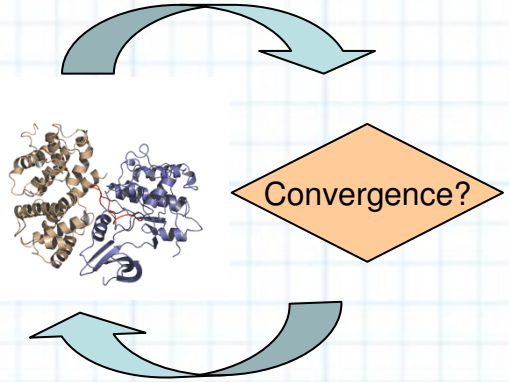
1	1	0	0	0	···	1	0

Outline

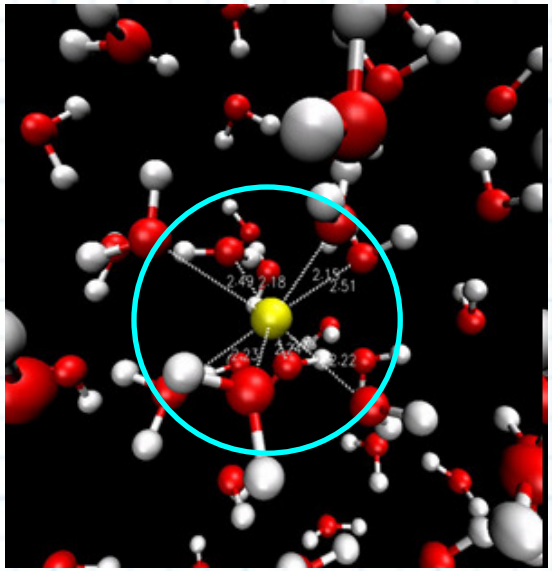
- Overview of Binding Site Mapping
  - *Rigid Docking*
  - *Energy Minimization*
- Overview of NVIDIA GPUs / CUDA
- Rigid Docking on GPU
- Energy Minimization on GPU
- Results

Energy Minimization

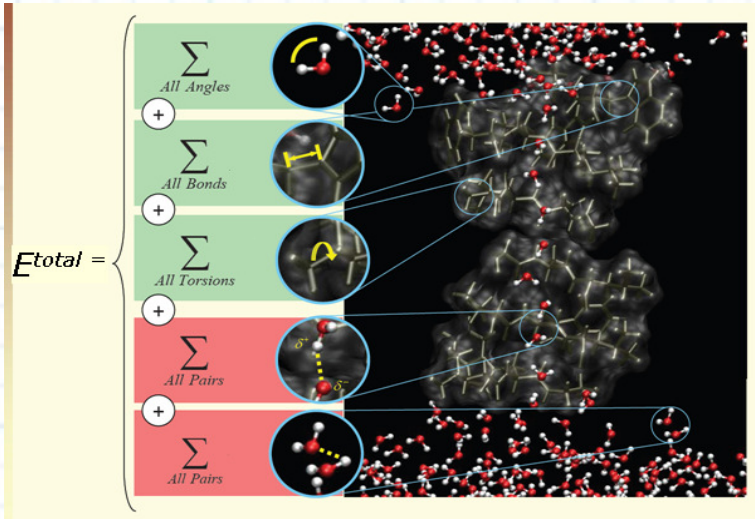
- Minimizing energy between two molecules
  - *Iterative process*
  - *Optimization moves*
- Used to model flexible side chains



N-body problem with a cut-off



$$E^{total} = \underbrace{E^{vdw} + E^{elec}}_{non\text{-}bonded} + \underbrace{E^{bond} + E^{angle} + E^{torsion}}_{bonded}$$



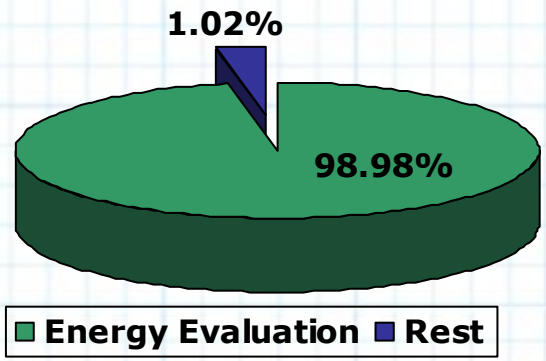
Looks like MD, but it's not

- Performed on a local region
  - *Many fewer atoms, typically few thousand*
- Much smaller atom neighborhoods
  - *Very small cut-off radius*
- Move to the next position
  - *Coordinate adjustments - No motion / velocity updates*
- No cell-lists / efficient filtering
  - *Refinement step; close to dest. - small motions*
- Neighbor lists are very sparse, with non-uniform distribution

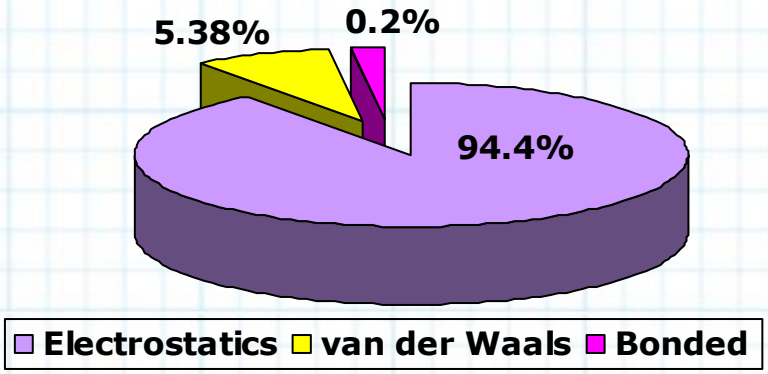
Different geometry

Different computations

Energy Minimization step of FTMap



FTMap Minimization Step



Energy evaluation phase

Absolute time ~ 10 ms per iteration (on a single core)

$$E^{total} = \underbrace{E^{vdw} + E^{elec}}_{non\text{-}bonded} + \underbrace{E^{bond} + E^{angle} + E^{torsion}}_{bonded}$$

FTMap Electrostatics Model

$$E^{total} = E^{vdw} + E^{elec} + E^{bond} + E^{angle} + E^{torsion}$$

Analytic Continuum Electrostatics (ACE)

Atom Self Energy: Electrostatic energy due to the charge itself

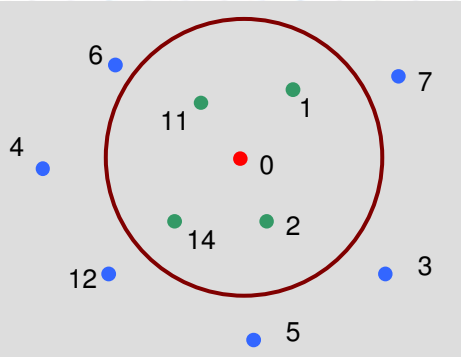
$$E_i^{self} = \frac{q_i^2}{2\varepsilon_s R_i} + \sum_{k \neq i} E_{ik}^{self} \qquad E_{ik}^{self} = \frac{\tau q_i^2}{\omega_{ik}} e^{-\left(\frac{r_{ik}^2}{\sigma_{ik}^2}\right)} + \frac{\tau q_i^2 \tilde{V}_k}{8\pi}\left(\frac{r_{ik}^3}{r_{ik}^4 + \mu_{ik}^4}\right)^4$$

Pairwise interaction – Generalized Born eqn.: Electrostatic energy due to the presence of other charges

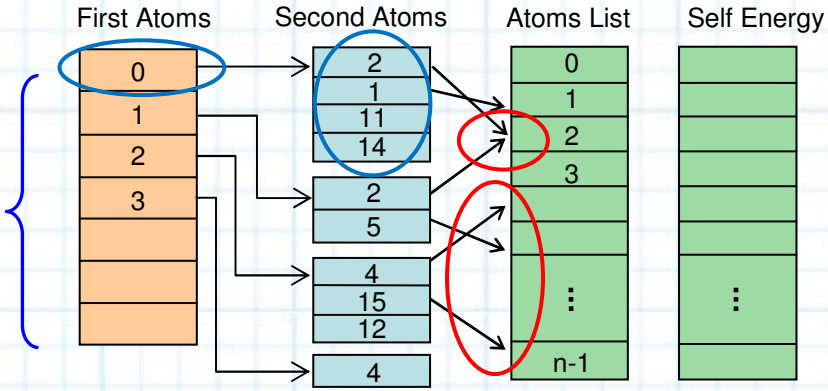
$$E_{ij}^{int} = 332 \sum_{j \neq i} \frac{q_i q_j}{r_{ij}} - 166\ \tau \sum_{j \neq i} \frac{q_i q_j}{\sqrt{r_{ij}^2 + \alpha_i \alpha_j e^{-\left(r_{ij}^2 / 4\alpha_i \alpha_j\right)}}}$$

Born Radii – depends on E^{self}

FTMap Data Structure - Neighbor Lists



Cycle through 1st atoms – update partial energies of both



- Random updates for second atoms
  - *Can't distribute the atoms list across multiprocessors*
- Write conflicts
  - *Second atom might appear in multiple lists*
- Not suitable for parallel implementations

- *Memory conflicts during updates*
- *Serialization during accumulation*

Energy Minimization on GPU – Challenges

- Little to no data reuse (within and across iterations)
- Small computation per iteration

Inherent to the algorithm

- Multiple accumulations – self energy of each atom must be computed

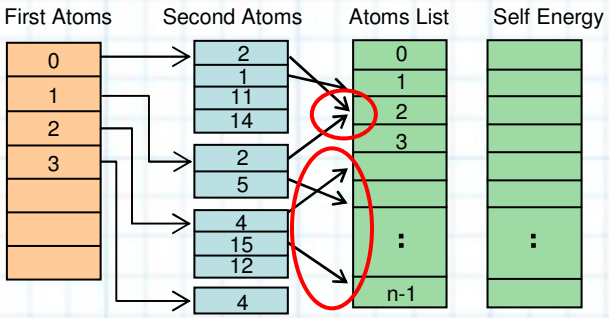
Born Radii – depends on E^{self}

$$E_i^{self} = \frac{q_i^2}{2\varepsilon_s R_i} + \sum_{k \neq i} E_{ik}^{self}$$

$$E_{ij}^{int} = 332 \sum_{j \neq i} \frac{q_i q_j}{r_{ij}} - 166\ \tau \sum_{j \neq i} \frac{q_i q_j}{\sqrt{r_{ij}^2 + \alpha_i \alpha_j e^{-\left(r_{ij}^2 / 4\alpha_i \alpha_j\right)}}}$$

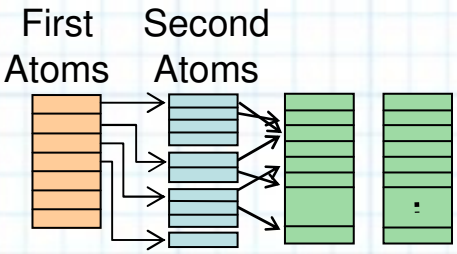
- Large data transfer time
- Random updates – write conflicts
- Accumulation requires serialization

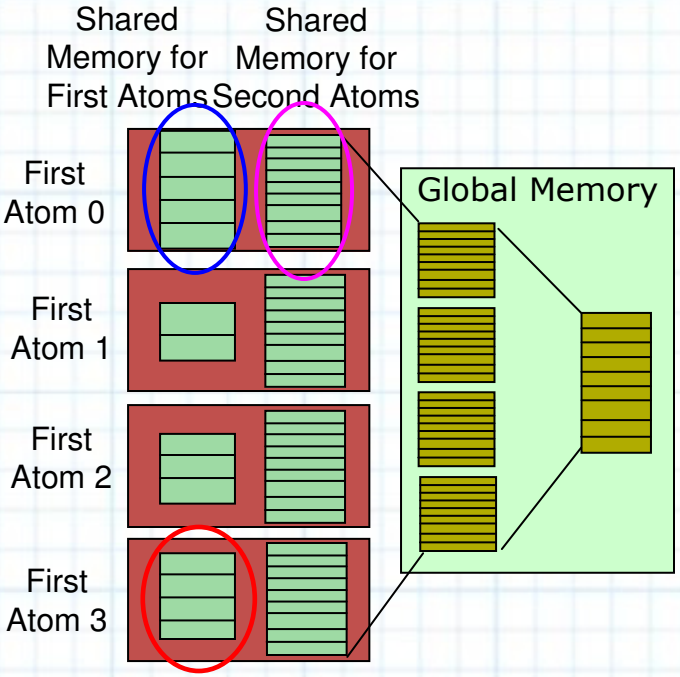
Architecture related



Neighbor Lists on GPUs

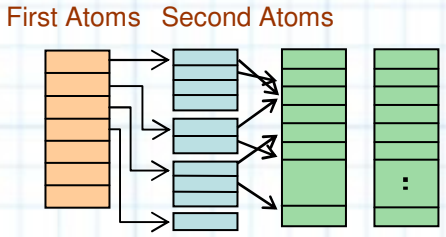
- Separate energy arrays for first and second atoms
  - *Allows parallel updates by multiple threads*
- Multiple copies of arrays for second atom
  - *One in each thread block*
  - *Parallel updates – no conflicts*
- First arrays reduced to single values
  - *Within the shared memory*
- Second atoms arrays merged by moving to global memory
  - *Large copy and accumulation time*
  - *Slow*





Modified Data Structure - Pairs List

- 2D neighbor ⇨ 1D pair list
  - *Each pair contains atom indices and types*
- Distribute pairs across multiple threads
  - *More uniform work distribution*
- Compute partial energies in parallel
- Perform accumulations serially



	Atom index		Atom Type	
Pair #	Atom 1	Atom 2	Atom 1	Atom 2
0	0	2	T5	T1
1	0	1	T5	T3
2	0	11	T5	T2
3	0	14	T5	T4
4	1	2	T3	T1
5	1	5	T3	T3
6	2	4	T1	T8
7	2	15	T1	T7
8	2	12	T1	T4
9	3	4	T5	T8

Pairs List on GPUs – Initial Attempts

- Pairs distributed on different threads
  - *Energy of an atom computed in different multiprocessors*
  - *Serialization during accumulation*
- Accumulation on GPU
  - *From global memory: Slow*
- Accumulation on host
  - *Fast, but requires energy arrays to be transferred every iteration*
  - *2x-3x speedup*

	Atom index		Self energy	
Pair id	Atom 1	Atom 2	Atom 1	Atom 2
0	0	2		
1	0	1		
2	0	11		
3	0	14		
4	1	2		
5	1	5		
6	2	4		
7	2	15		
8	2	12		
9	3	4		

Pairs List on GPUs – Improved Scheme

Pairs list with two changes

- Conflicts due to random occurrence of second atoms
  - *Split forward and reverse pair list*
  - *Process only the first atom of each list*
- Indeterminate distribution requires serialization during accumulation
  - *Statically map the pairs onto GPU threads*
  - *New data structure: Assignment tables*

	Atom index		Self energy	
Pair id	Atom 1	Atom 2	Atom 1	Atom 2
0	0	2		
1	0	1		
2	0	11		
3	0	14		
4	1	2		
5	1	5		
6	2	4		
7	2	15		
8	2	12		
9	3	4		

Split Pairs List

- Forward list: Same as before
- Reverse list: Treat every second atom as a first atom

- Process only the first atoms of each list
- Adds determinism => Better distribution

Forward List

	Atom index		Self energy	
Pair id	Atom 1	Atom 2	Atom 1	Atom 2
0	0	2		
1	0	1		
2	0	11		
3	0	14		
4	1	2		
5	1	5		
6	2	4		
7	2	15		
8	2	12		
9	3	4		

Reverse List

	Atom index		Self energy	
Pair id	Atom 1	Atom 2	Atom 1	Atom 2
0	1	0		
1	2	0		
2	2	1		
3	4	2		
4	4	3		
5	5	1		
6	11	0		
7	12	2		
8	14	0		
9	15	2		

Static Mapping - Assignment Table

- Pairs can be grouped by first atom
- Groups mapped to different thread blocks
  - *Look for next block with enough threads*
- One pair per thread (multiple if $N_{pair} > N_{threads}$)
- Reverse Assignment table for the second atoms

	Thread Id	Pair Id	Atom 1	Atom 2	Master	Num. Atoms	
Thread Block 0	0	0	0	2	1	4	Group 0
	1	1	0	1	0	4	
	2	2	0	11	0	4	
	3	3	0	14	0	4	
	4	9	3	4	1	1	Group 3
Thread Block 1	5	4	1	2	1	2	Group 1
	6	5	1	5	0	2	
	7	6	2	4	1	3	Group 2
	8	7	2	15	0	3	
	9	8	2	12	0	3	

Unused threads used by next group

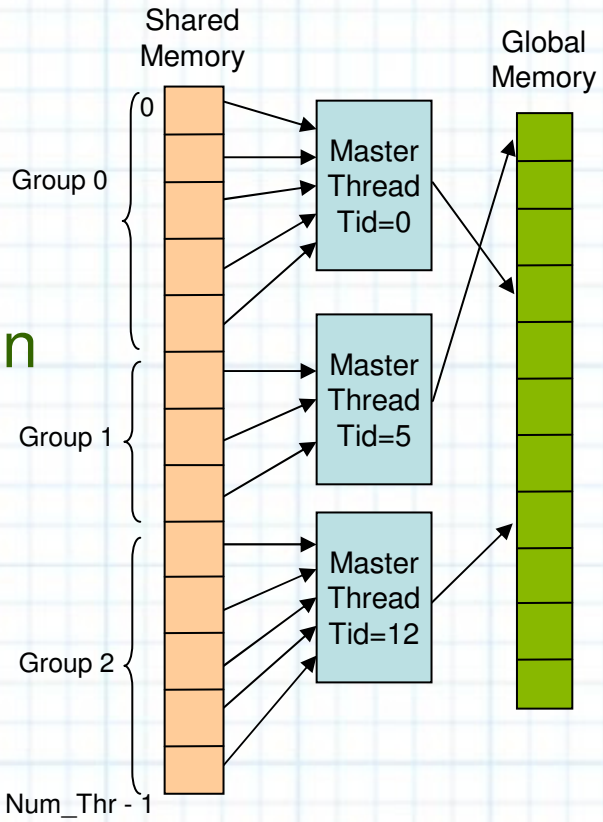
Does not fit on TB_0

Computing and Accumulating Energies

- Threads store partial energies in shared memory
  - *Address = Local Thread Id*

Thread Id	Pair Id	Atom 1	Atom 2	Master	Num. Atoms
0	0	0	2	1	4
1	1	0	1	0	4
2	2	0	11	0	4
3	3	0	14	0	4
4	9	3	4	1	1

- Master thread performs accumulation
  - *'N' locations starting from its thread id*
- Multiple parallel accumulations per thread block (from shared memory)



Outline

- Overview of Binding Site Mapping
  - *Rigid Docking*
  - *Energy Minimization*
- Overview of NVIDIA GPUs / CUDA
- Rigid Docking on GPUs - PIPER
- Energy Minimization on GPUs - FTMap
- Results

Results - Speedups

Speedups for Rigid Docking Step

Computation (Per Rotation)	Serial Runtime (ms)	GPU Runtime (ms)	Speedup	
			vs. 1 core	vs. 4 cores*
Rotation + Grid Assignment	80	80	1x	--
Correlations	3600	13.5	267x	70x
Accum. Of Desolvation Terms	180	1	180x	--
Scoring and Filtering	200	30	6.67x	--
Total time per rotation	**4060**	**125.5**	**32.6x**	**11x**

Results - Speedups

Speedups for Energy Minimization Step

2260 atoms

9780 atom-pairs

Computation	Serial Runtime (ms)	GPU Runtime* (ms)	Speedup
Self Energy	6.15	0.23	26.7x
Pairwise Interaction	2.75	0.19	17x
van der Waals	0.5		
Force Updates	0.95	0.14	6.7x
Optimization move	0.005	0.005	1x

**** Overall Speedup on EM computations : 18.5x***

**** Overall FTMap speedup (including overhead): 15x***

** GPU runtimes include data transfer time*

Results – Precision Analysis

- Single vs Double Precision
  - *RMSD error on force values for first iteration:* 10^{-6}
  - *Convergence in 50 iterations (as opposed to 600)*
  - *Error on final energy and force values*
    - *Energy : 10^{-3}*
    - *Forces : 10^{-5}*
  - *Error on atom coordinates after minimization*
    - *0.5 Å*
- Exact match for double precision
  - *Atom coordinates within* 10^{-5} Å
  - More complex mapping on GPU – Similar speedup numbers

Conclusion

- GPUs can deliver high performance
  - *Even for double precision computations*
- To obtain good performance:
  - *Alternate algorithms are often needed*
  - *Restructuring of data-structures is crucial*
  - *Efficient use of memory hierarchy is essential*
- Getting it right on the GPU is easy …
  - *… getting good performance is not so much!*

Thank You!