



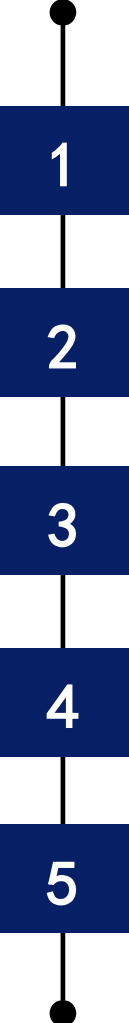
SW_GROMACS: Accelerate GROMACS on Sunway TaihuLight

Tingjian Zhang

SHANDONG UNIVERSITY

National Supercomputer Center in Wuxi

Contents



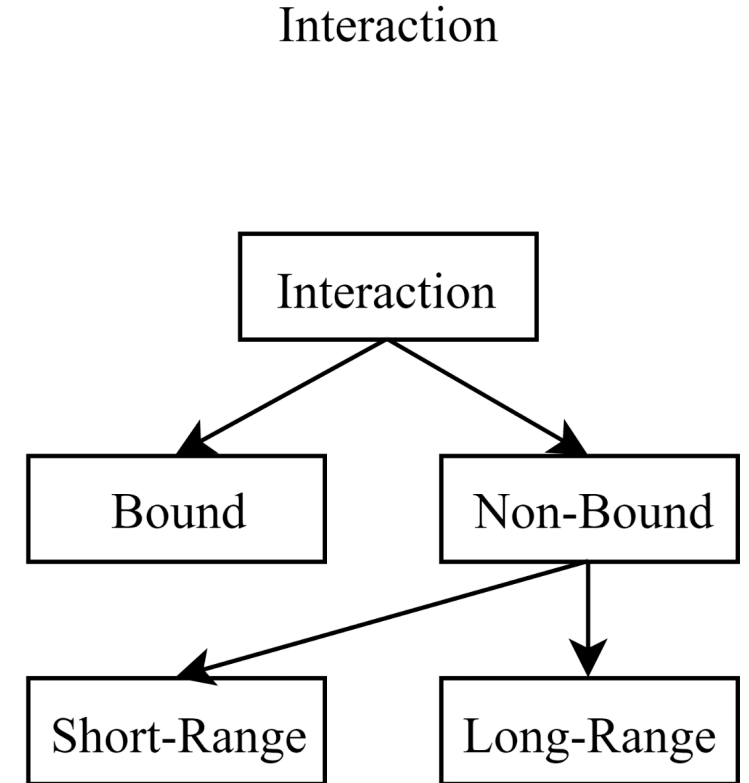
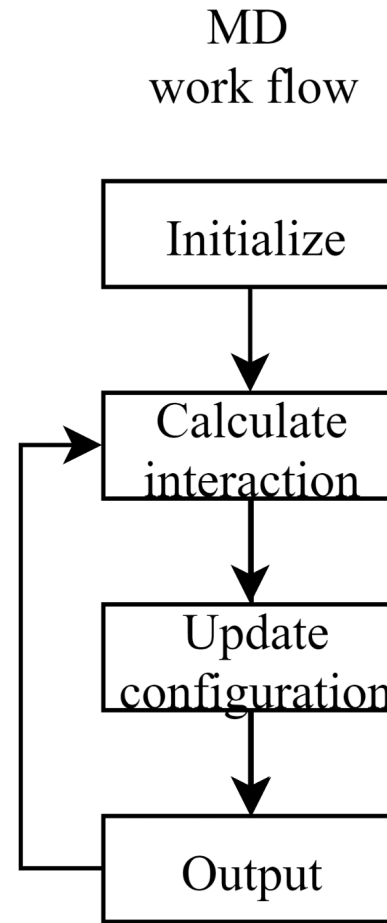
1	Background
2	Strategies
3	Evaluation
4	Conclusion
5	Future Work

Part **1**

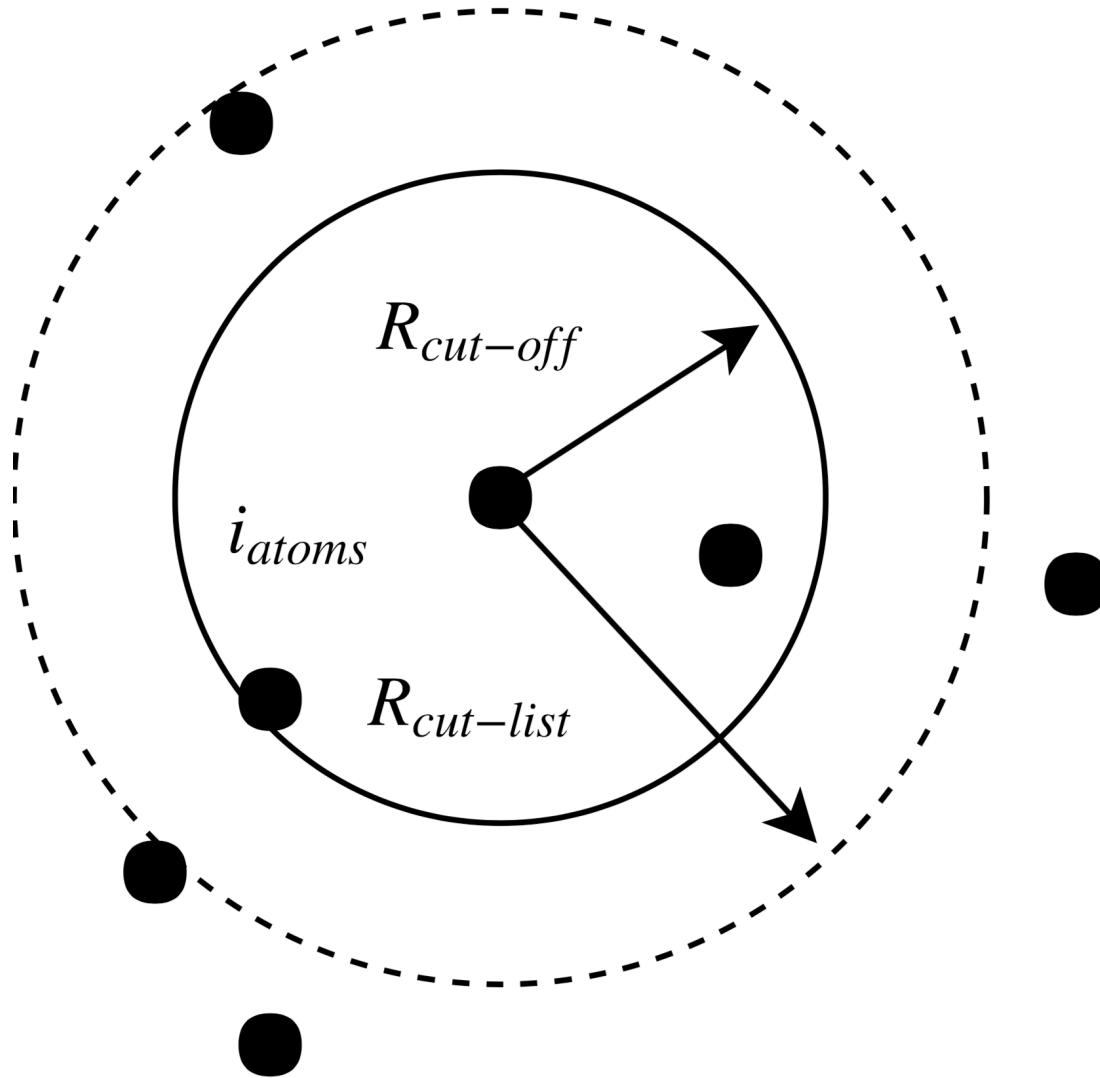
Background

Simple Introduction of GROMACS

- GROMACS: One of most popular Modular Dynamic applications
- Have a wide used area in the field of chemical and bi-molecular system
- The performance of GROMACS on SUNWAY TaihuLight is not very well

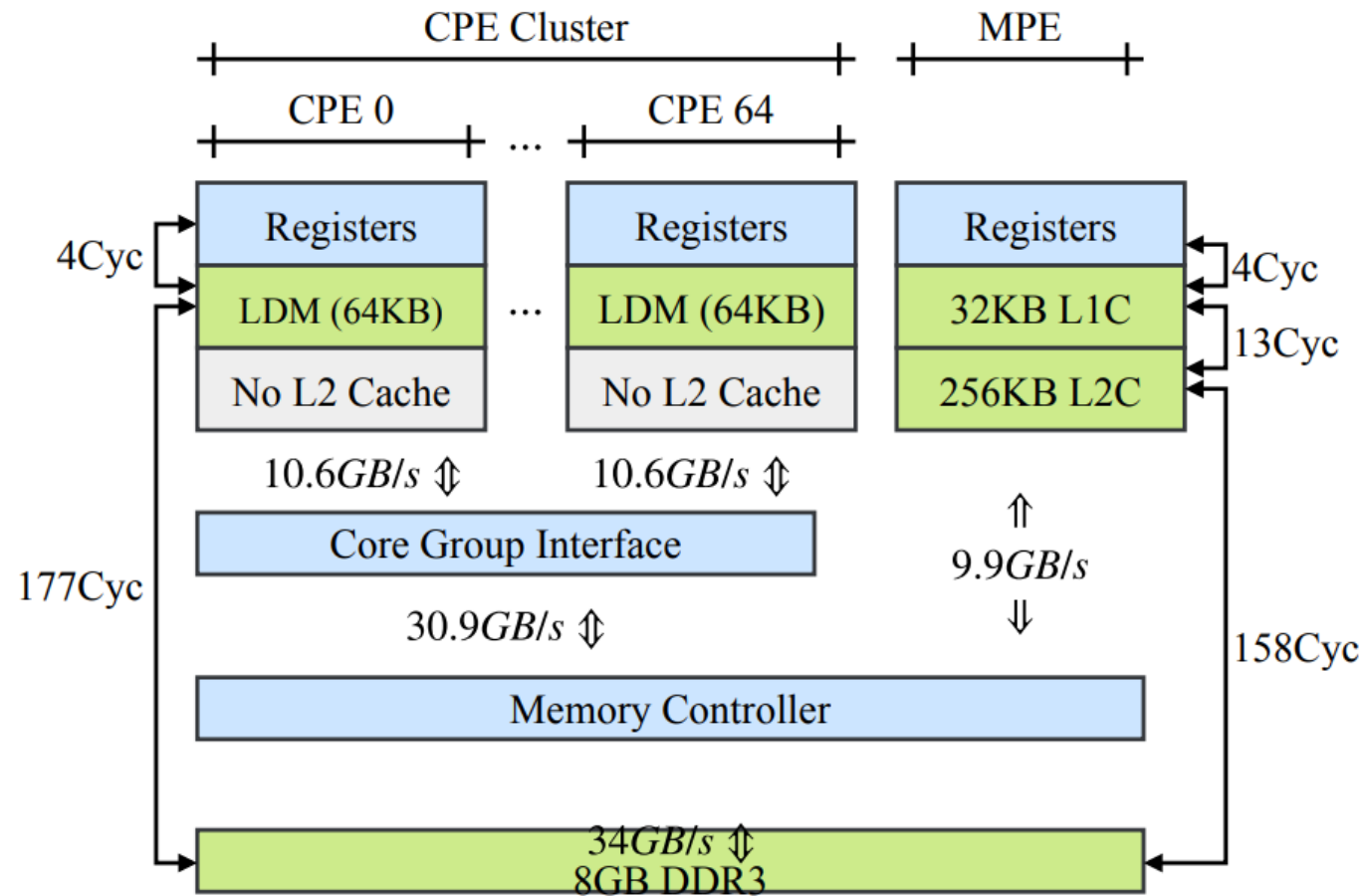


Short-Range Interaction of GROMACS



- Make a neighbor list for every particle
- Calculate interactions basing on the neighbor list
- Update the information of every particle
- Neighbor list will be rebuilt in a certain step

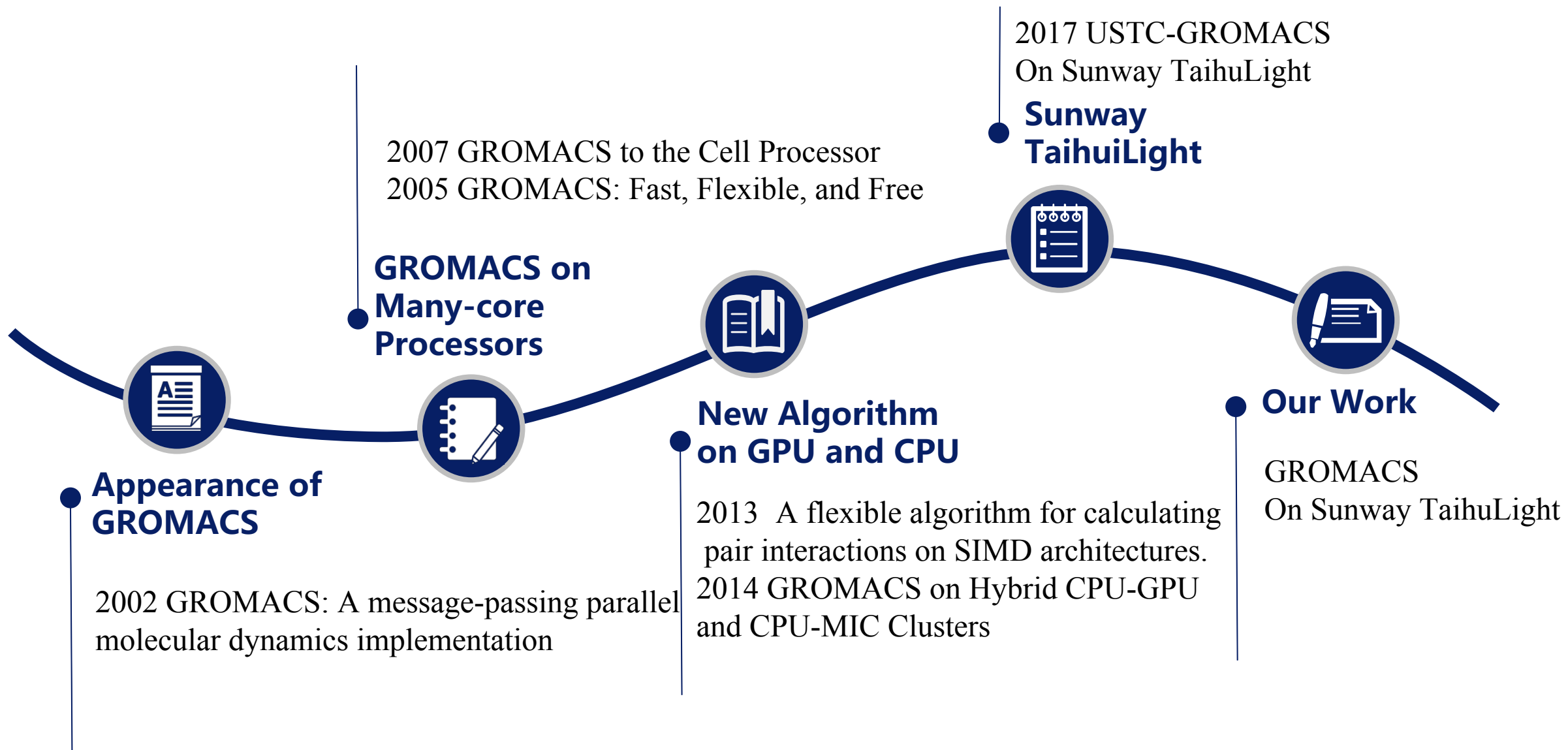
The SW26010



- Low Bandwidth
- Small LDM
- No Cache in LDM

memory architecture
of one CG in SW26010

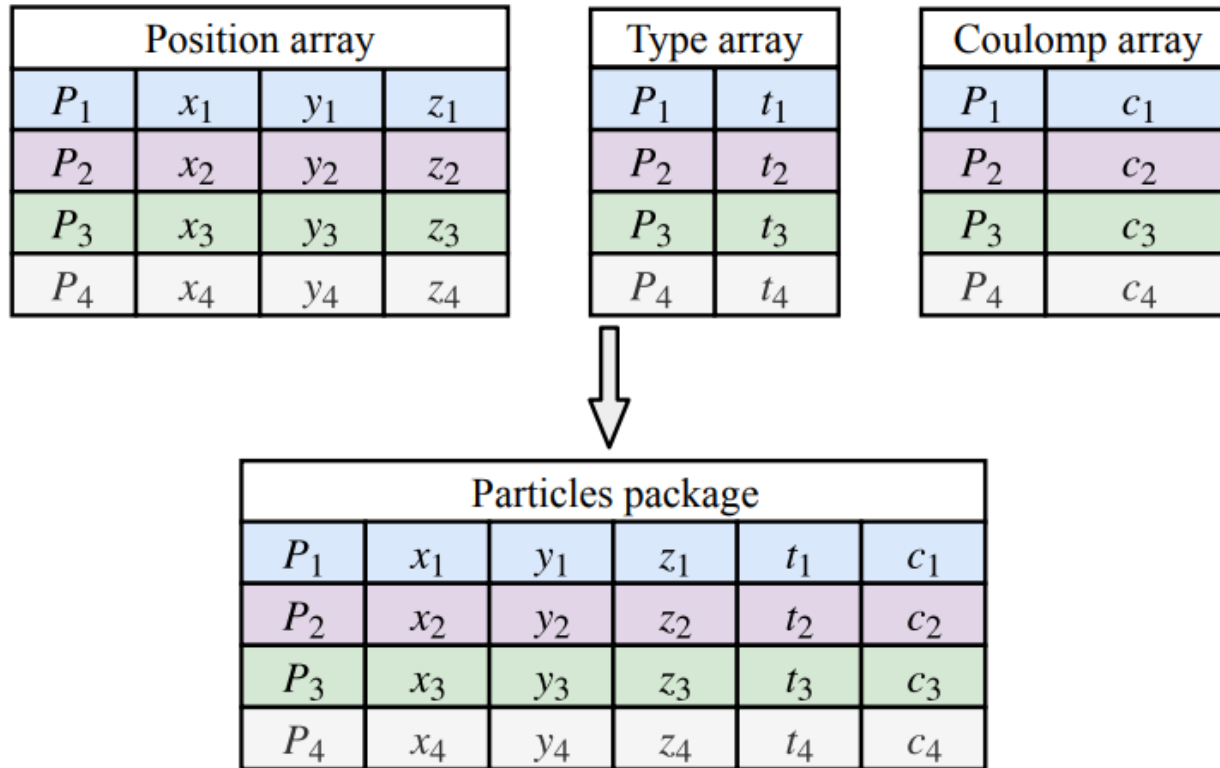
Related Works





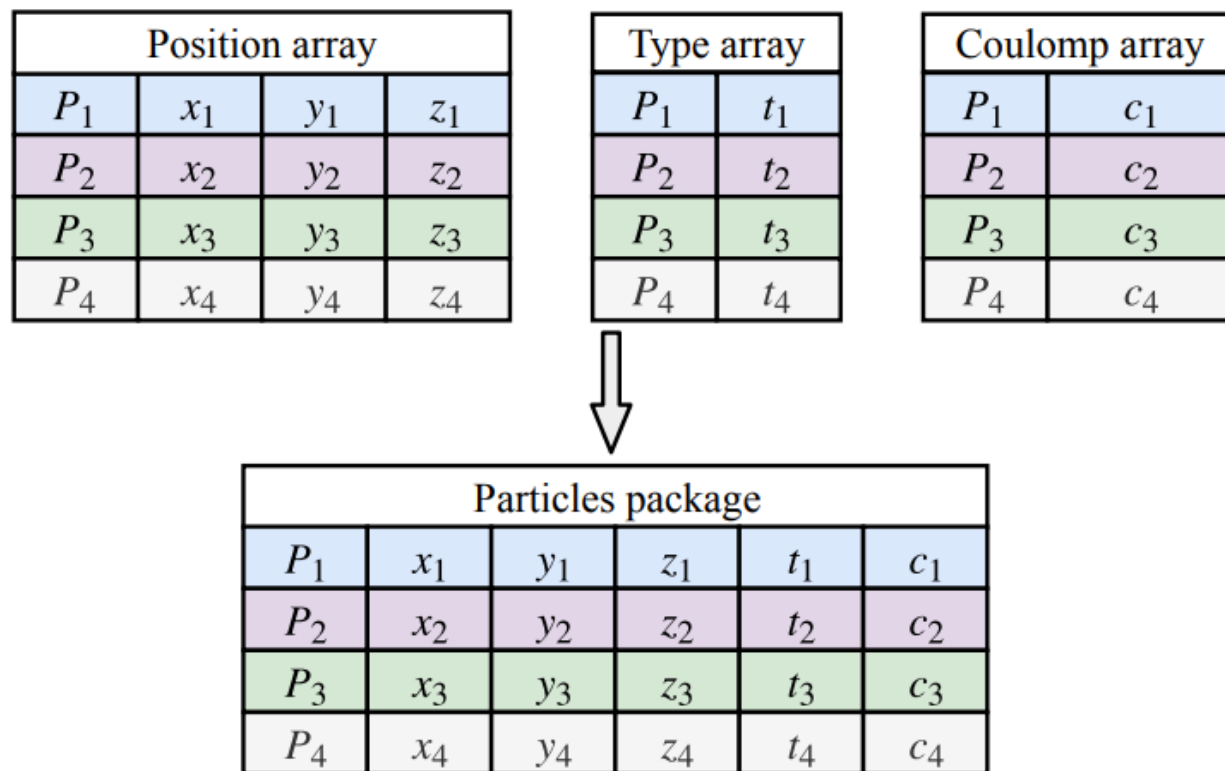
Part 2 Strategies

Strategies Restructure Data

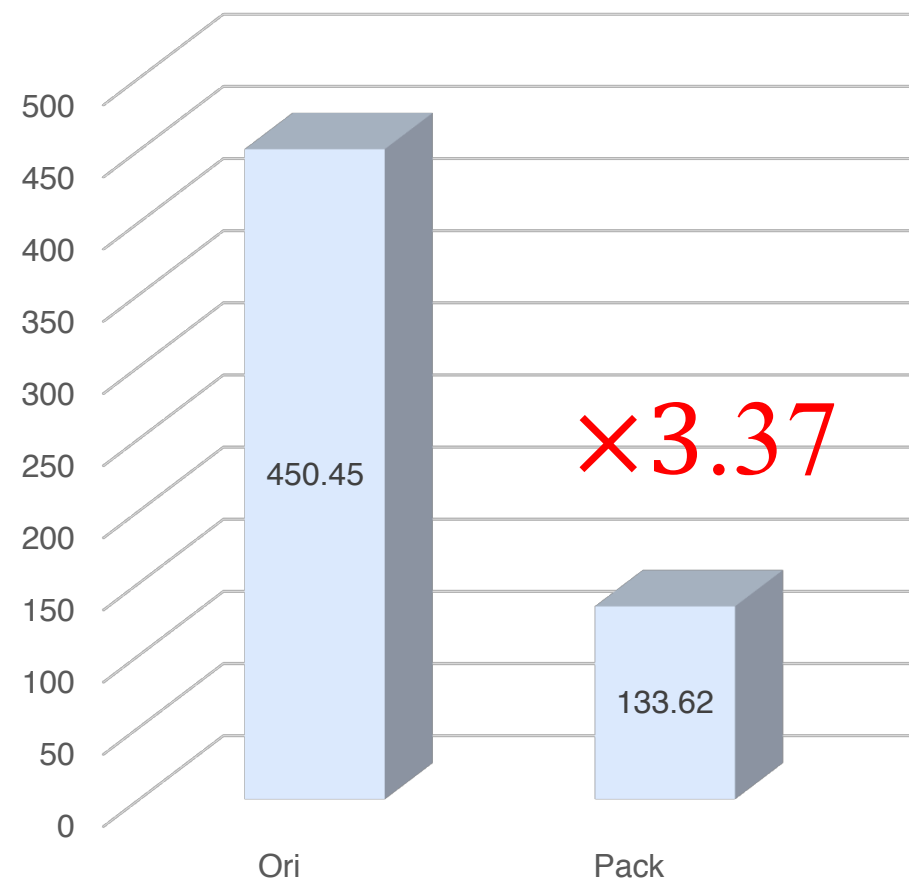


- Make the data of same particle contiguous
- Improve the bandwidth
- Reduce the time of memory access

Strategies Restructure Data

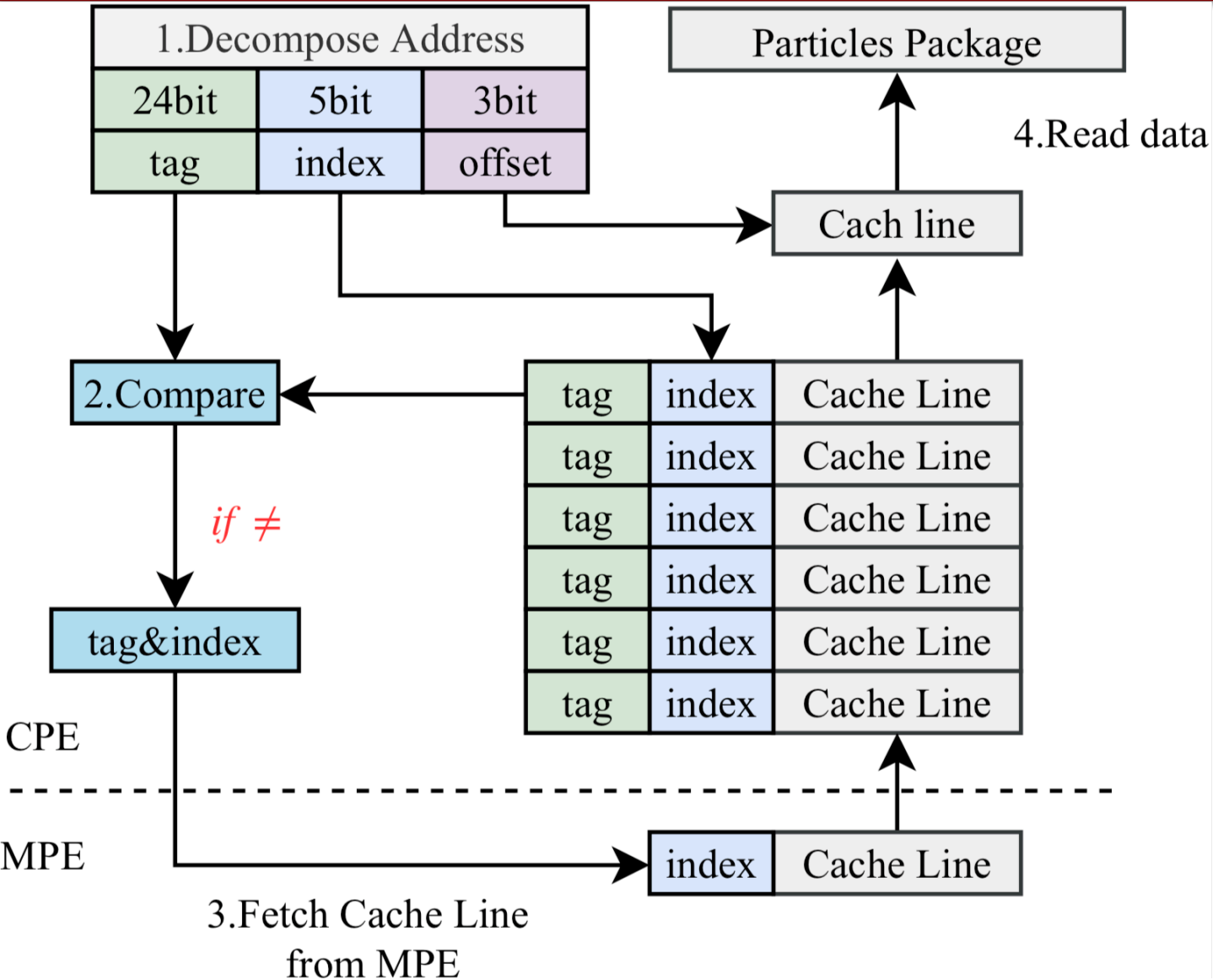


Restructure Data



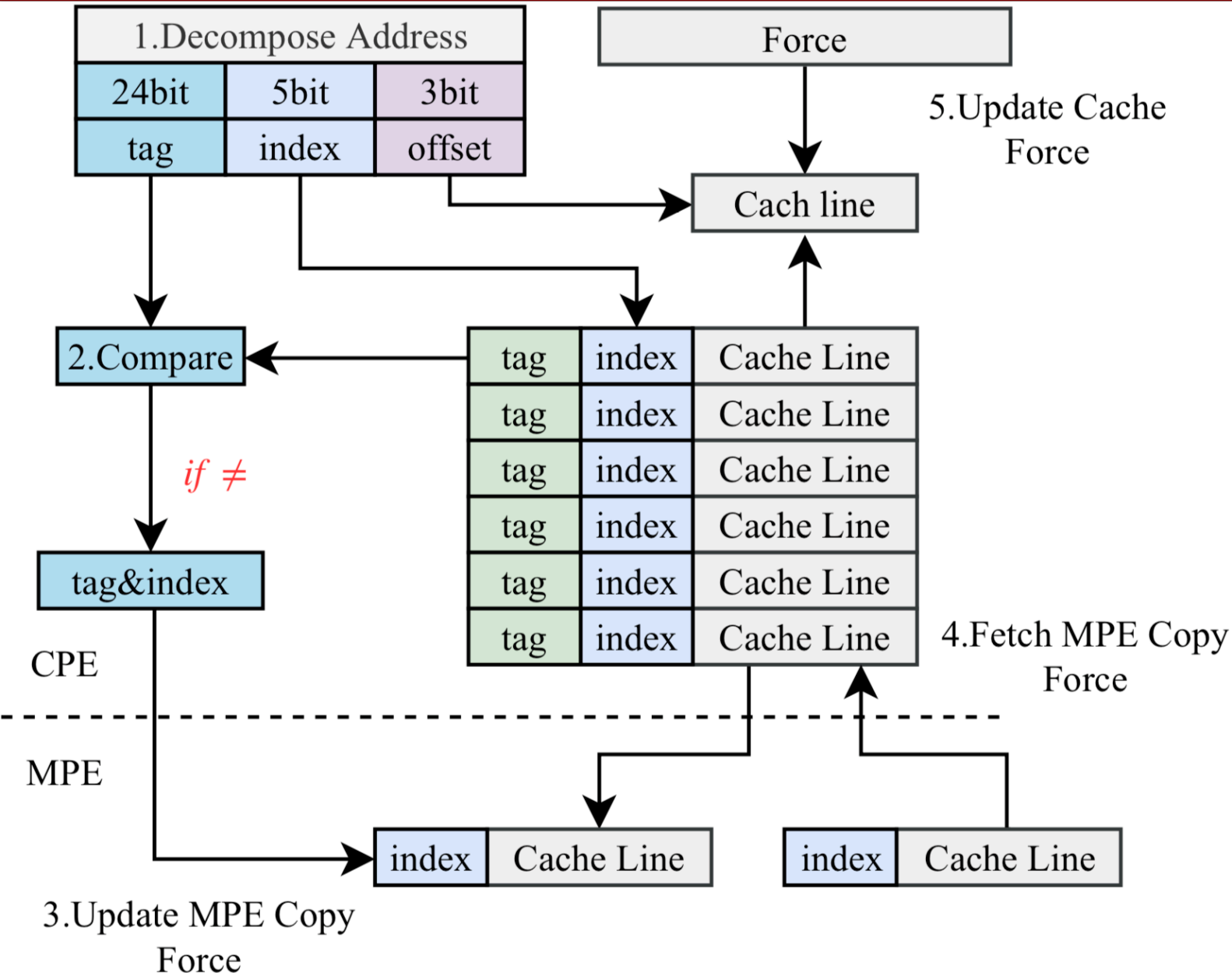
■ Time calculate 1.2M particles 1000 iterations on 1CG(/s)

Strategies Read Cache



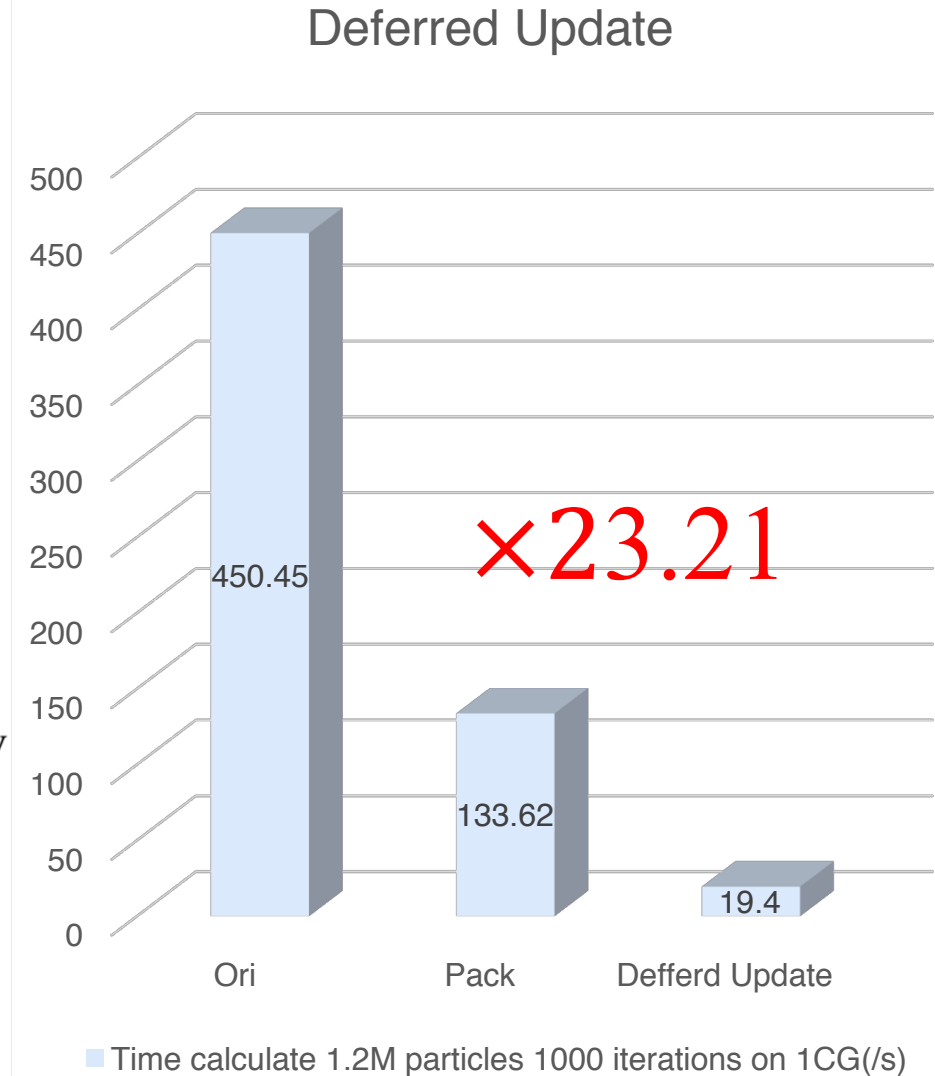
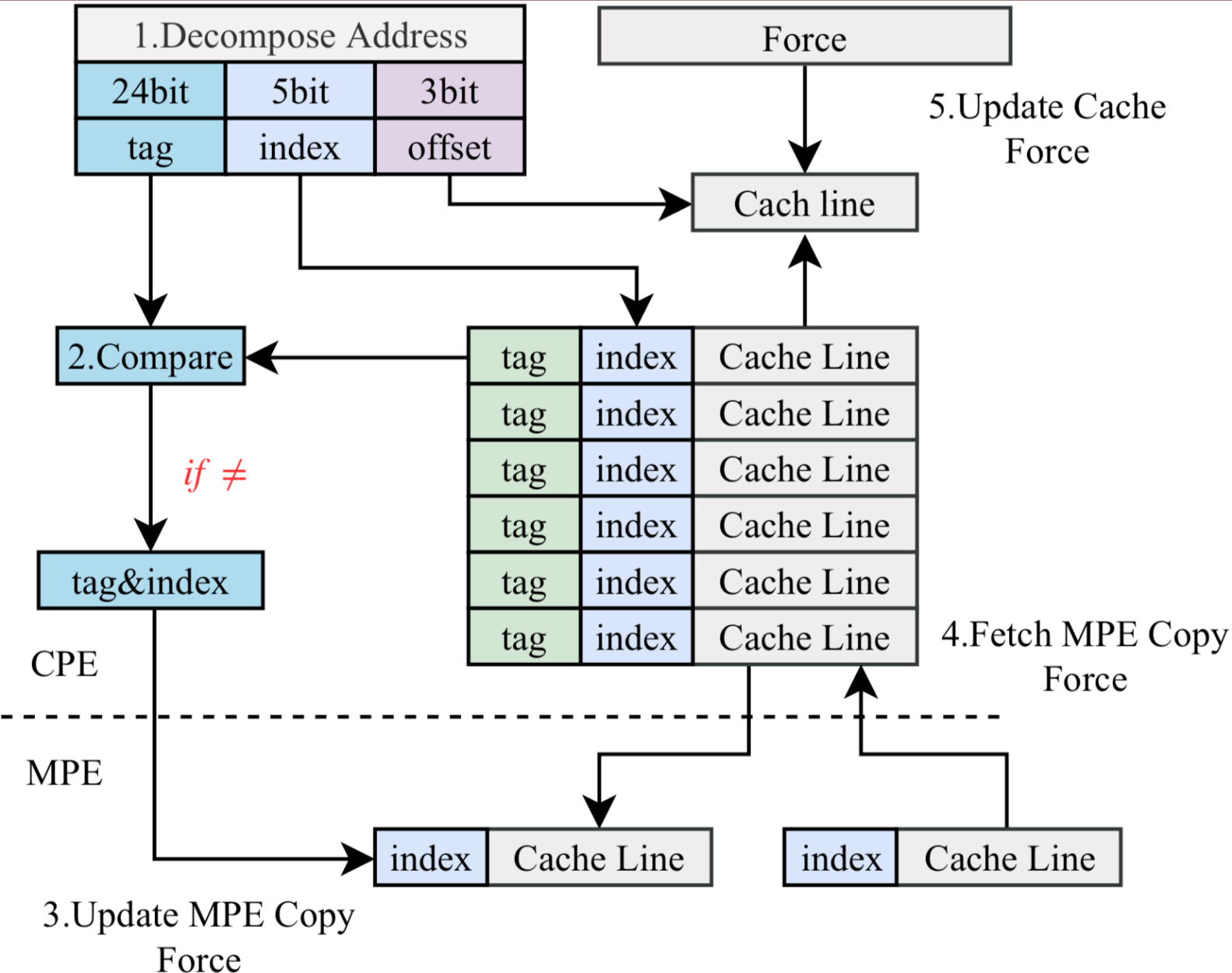
- Increase bandwidth
- Reuse the data in LDM
- Reduce the times fetch data from main memory

Strategies Deferred Update



- Increase bandwidth
- Reuse the data in LDM
- Reduce the times update main memory

Strategies Deferred Update



Strategies Vectorization

Particles Package					
P_1	x_1	y_1	z_1	t_1	c_1
P_2	x_2	y_2	z_2	t_2	c_2
P_3	x_3	y_3	z_3	t_3	c_3
P_4	x_4	y_4	z_4	t_4	c_4



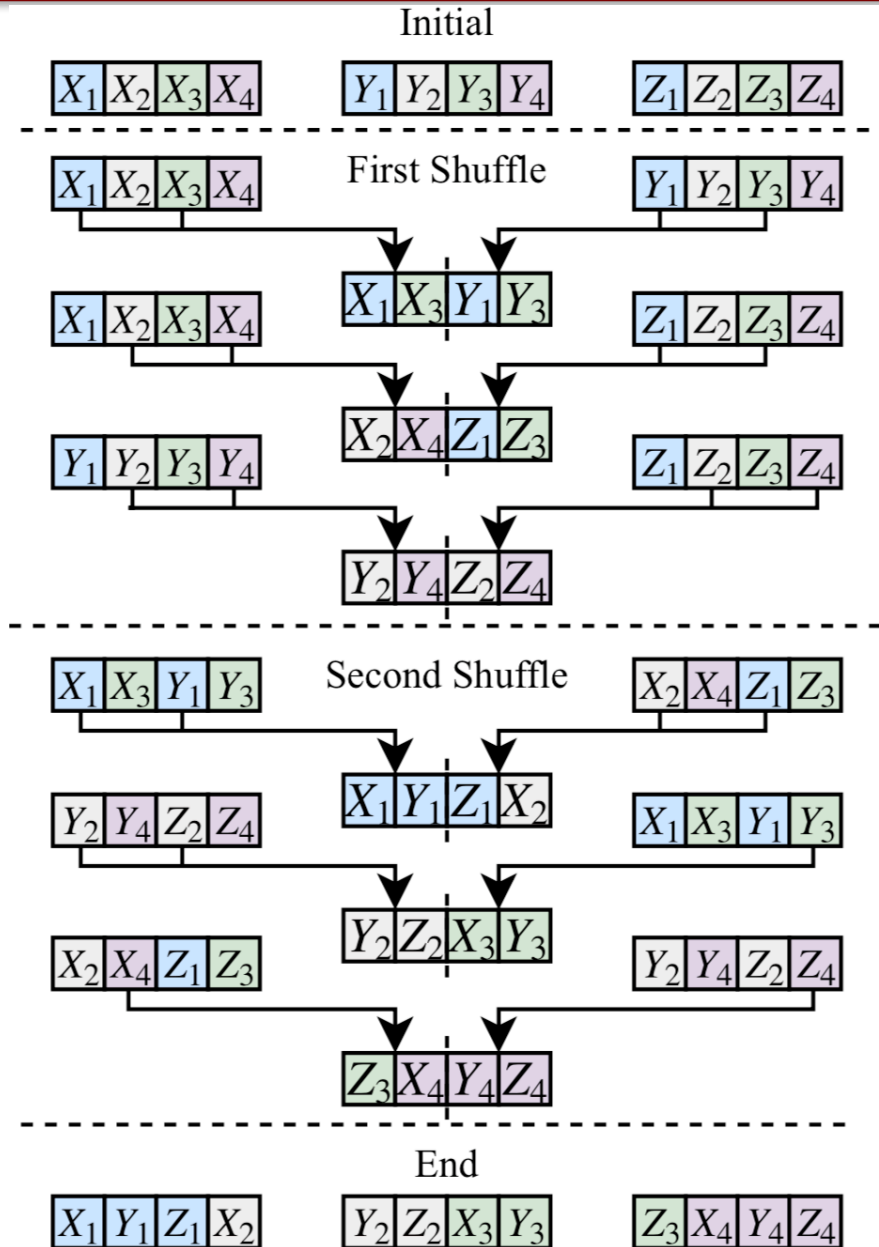
P
x
y
z
t
c

Cal
 \Leftrightarrow

Particles Package			
P_1	P_2	P_3	P_4
x_1	x_2	x_3	x_4
y_1	y_2	y_3	y_4
z_1	z_2	z_3	z_4
t_1	t_2	t_3	t_4
c_1	c_2	c_3	c_4

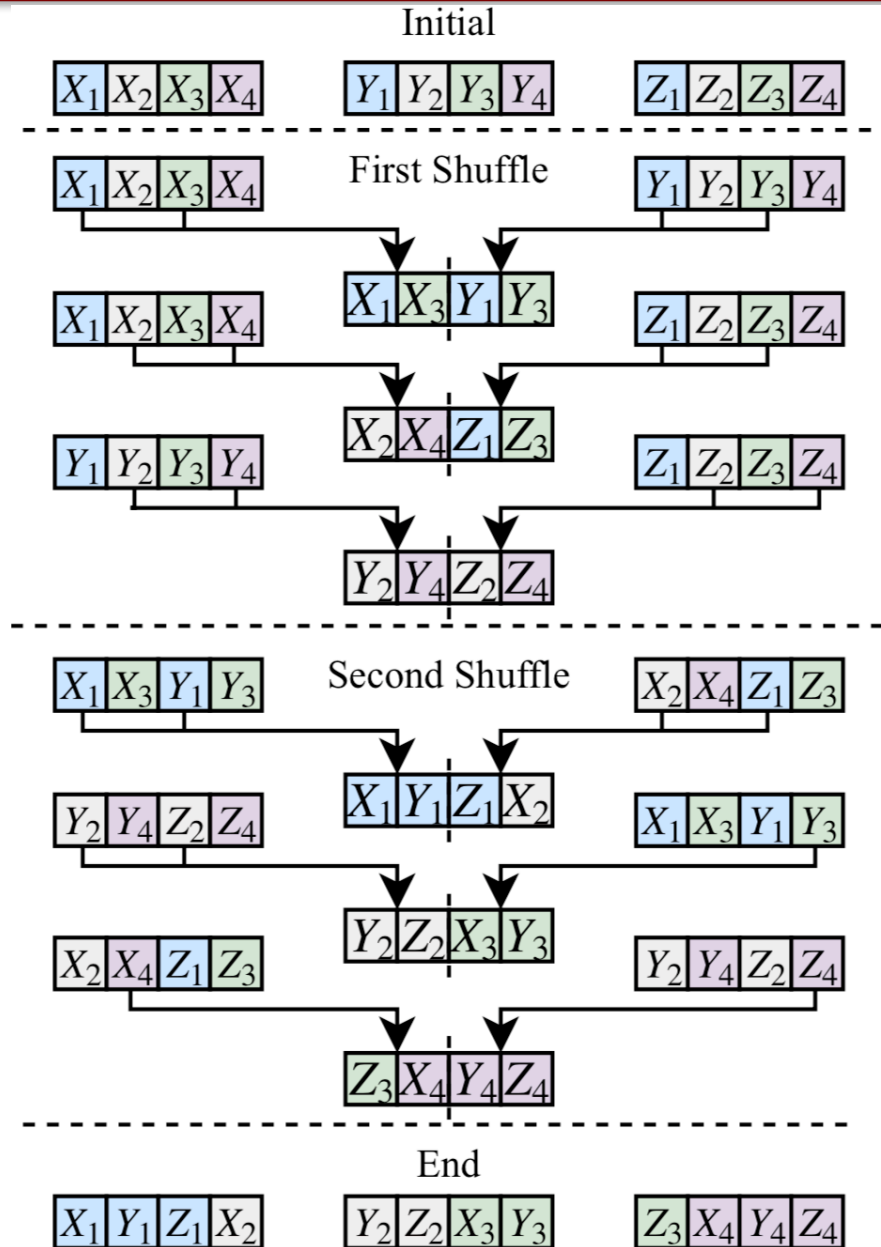
- Make registers could fetch the four floats once
- Save the pre-treatment time
- Translate the data by Vshuffle
- Reduce the post-treatment time

Strategies Vectorization

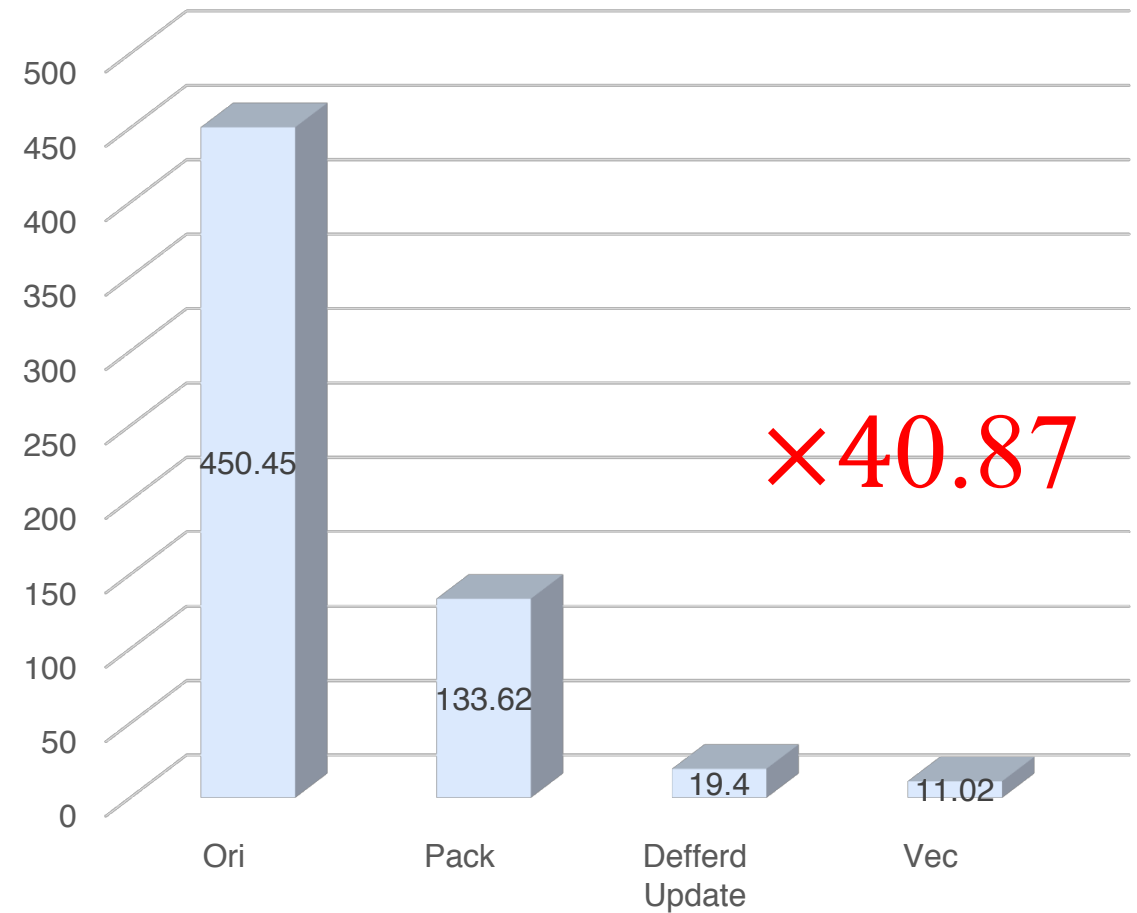


- Make registers could fetch the four floats once
- Save the pre-treatment time
- Translate the data by Vshuffle
- Reduce the post-treatment time

Strategies Vectorization



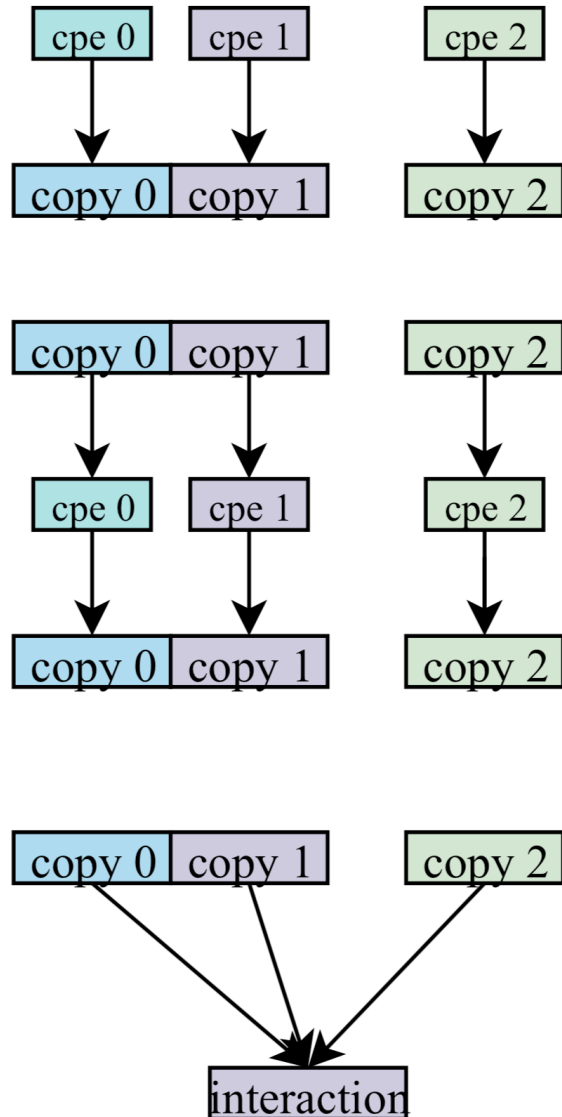
Vectorization



Time calculate 1.2M particles 1000 iterations on 1CG(/s)

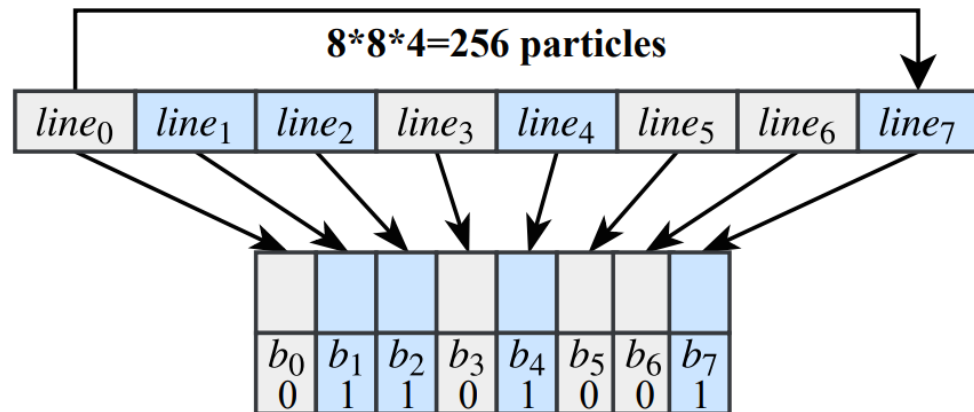
Strategies Many Copies

initiation step



- Keep a copy of the interaction array for every CPE
- All CPEs update their own interaction array
- Reduce interaction array copies into one

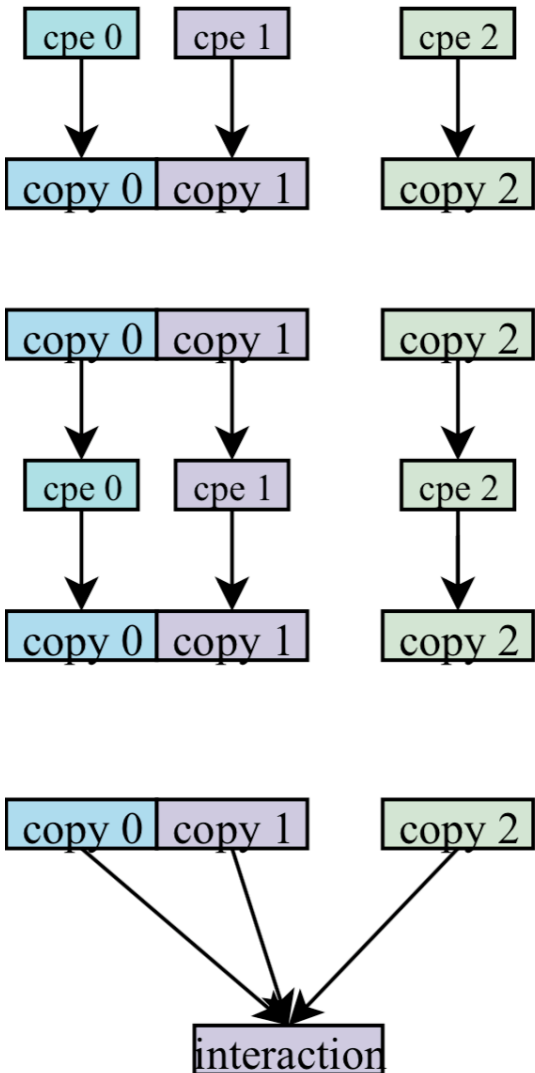
Strategies Bit-Map



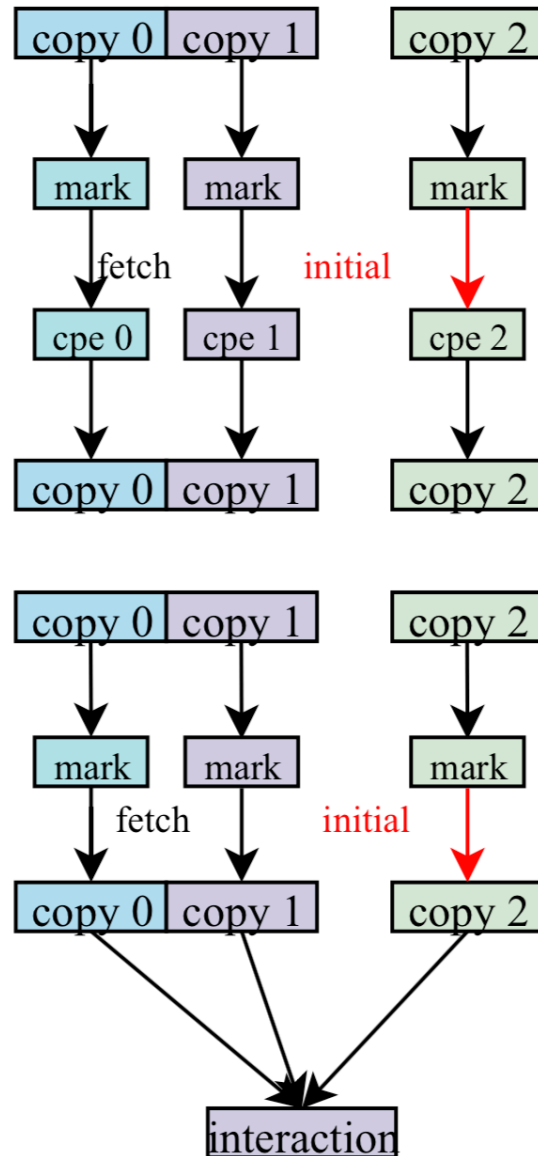
- Keep the update state of a cache line in one bit
- Desert the initialization step
- Reduce the reduction step

Strategies Bit-Map

initiation step



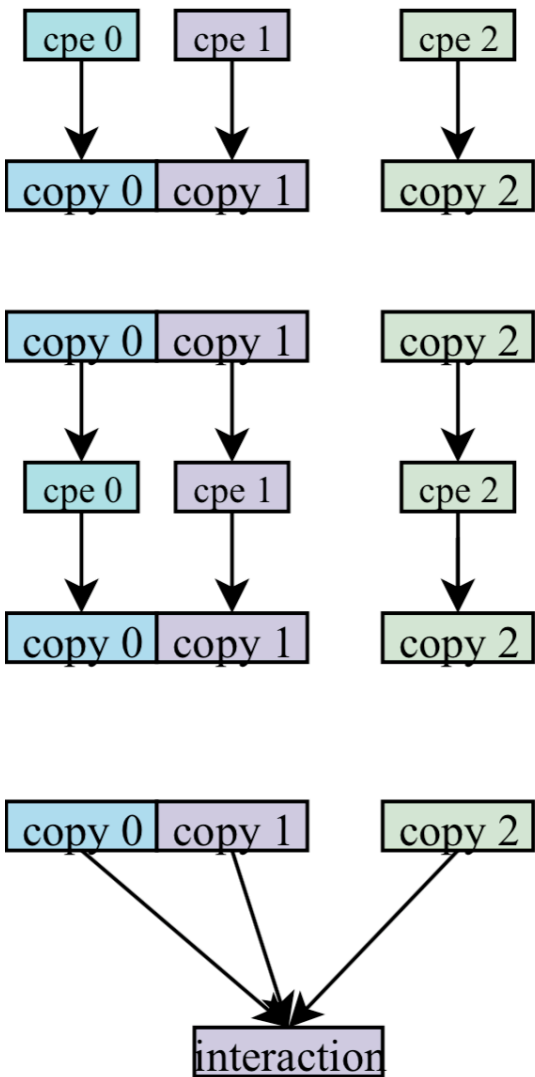
without initiation



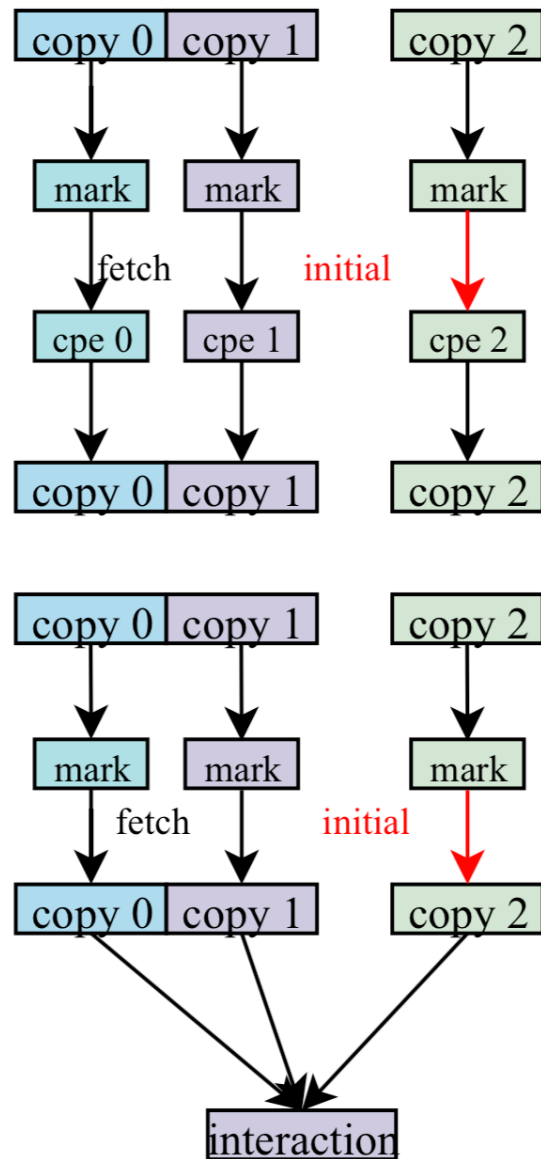
- Keep the update state of an cache line in one bit
- Desert the initialization step
- Reduce the reduction step

Strategies Bit-Map

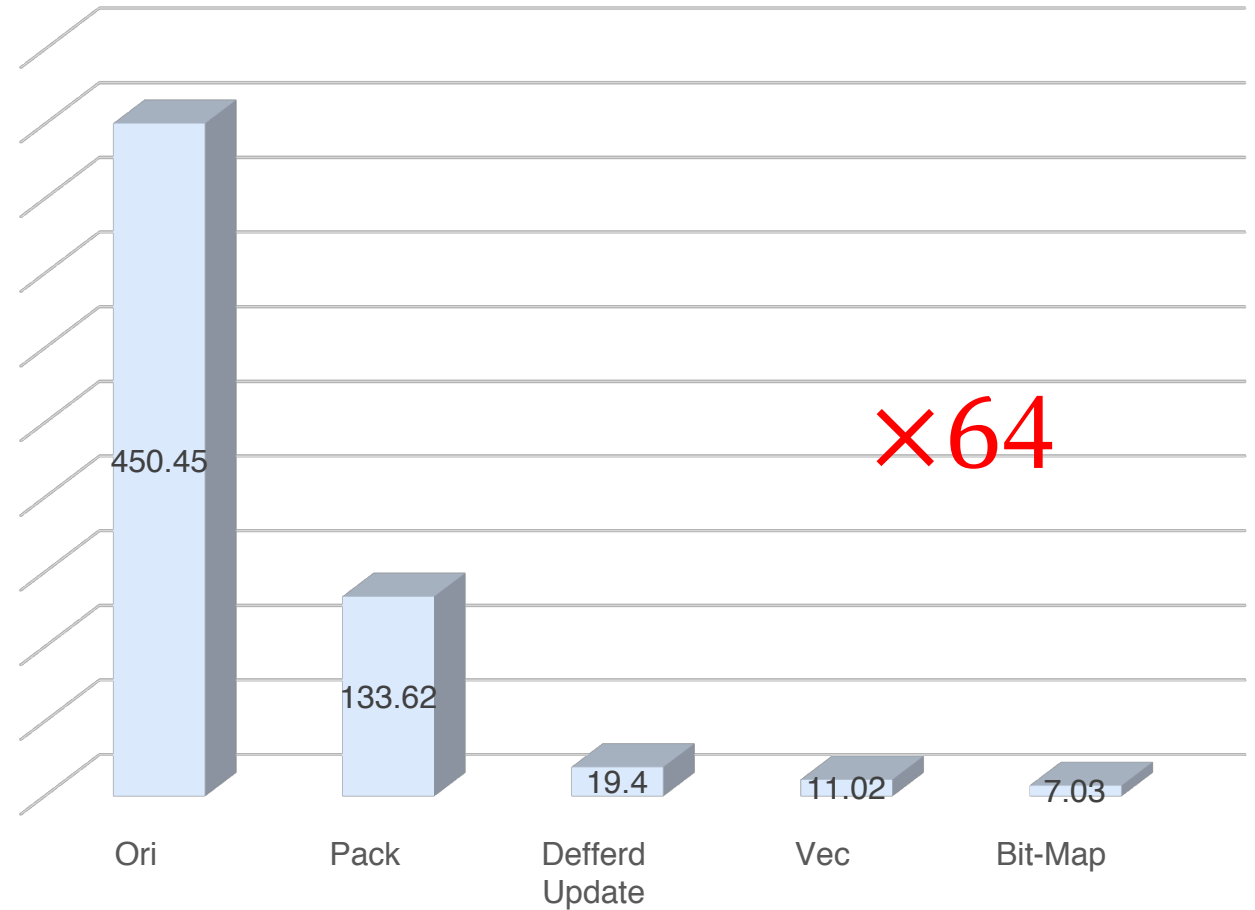
initiation step



without initiation



Bit-Map



Time calculate 1.2M particles 1000 iterations on 1CG(/s)

Other Optimization

- Optimize the make list.
- Optimize the MPI with RDMA. Re-implement the ALL to ALL communication with RDMA.
- Optimize the output step.

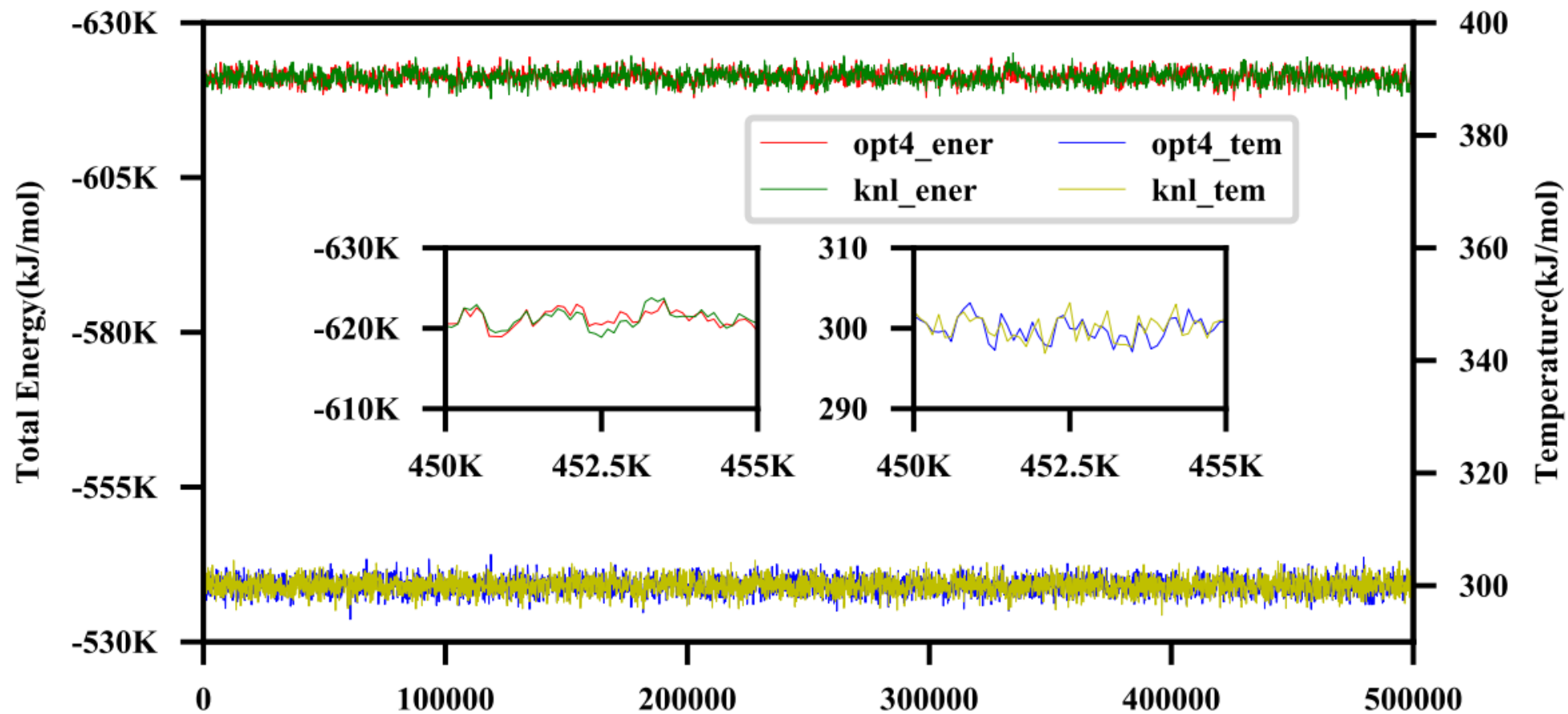
Part **3**

Evaluation

Benchmark

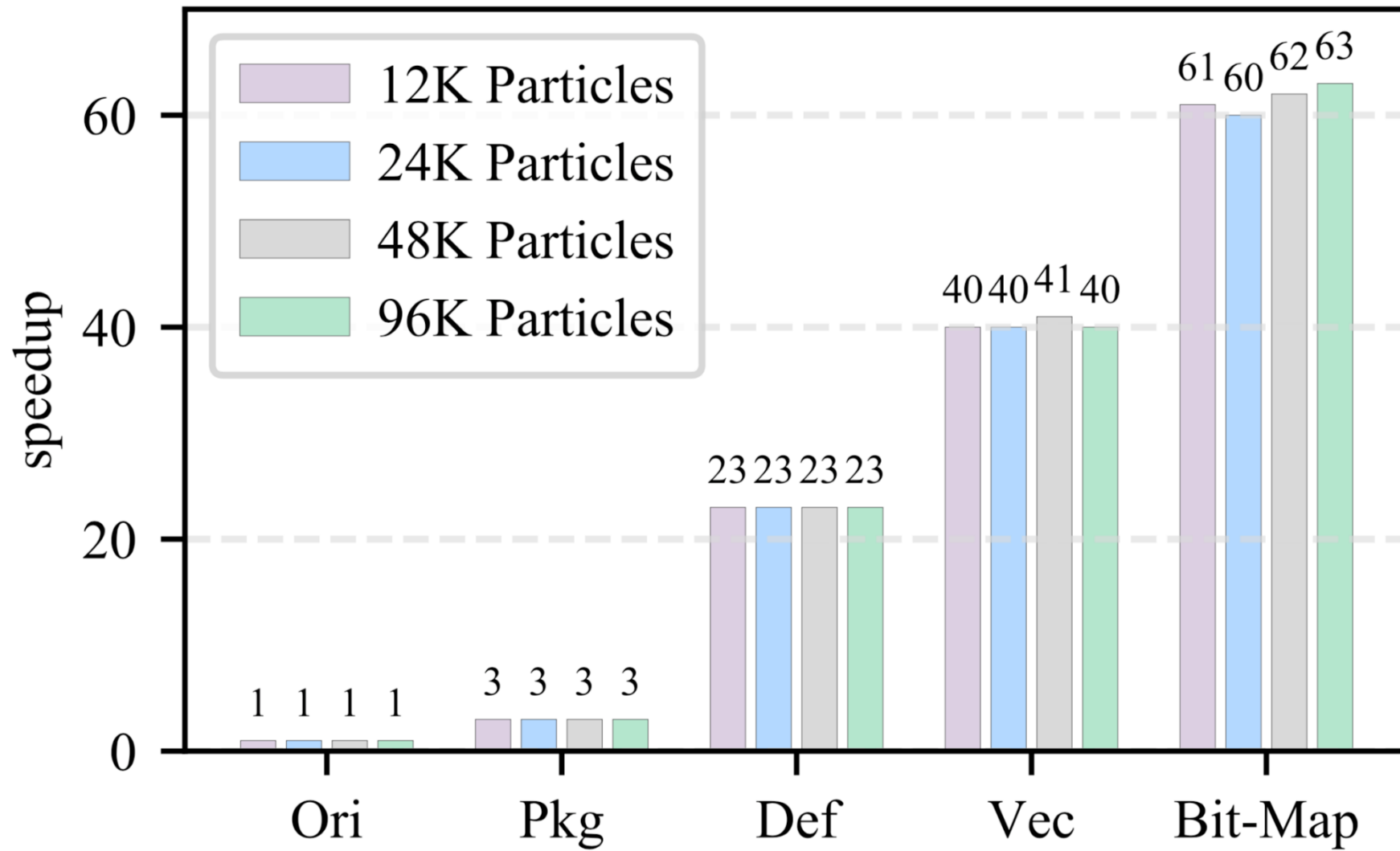
Key variable	Value
Particles number	0.9k~3,000K
Nlist	10
Ns_type	grid
coulombtype	PME
rlist	1.0
cutoffscheme	varlet

Accuracy



**Thre Energy and Temperature of the original code and the optimization code
in the benchmark of 3000K particles and 50000 time steps**

Performance Evaluation Performance of different strategies



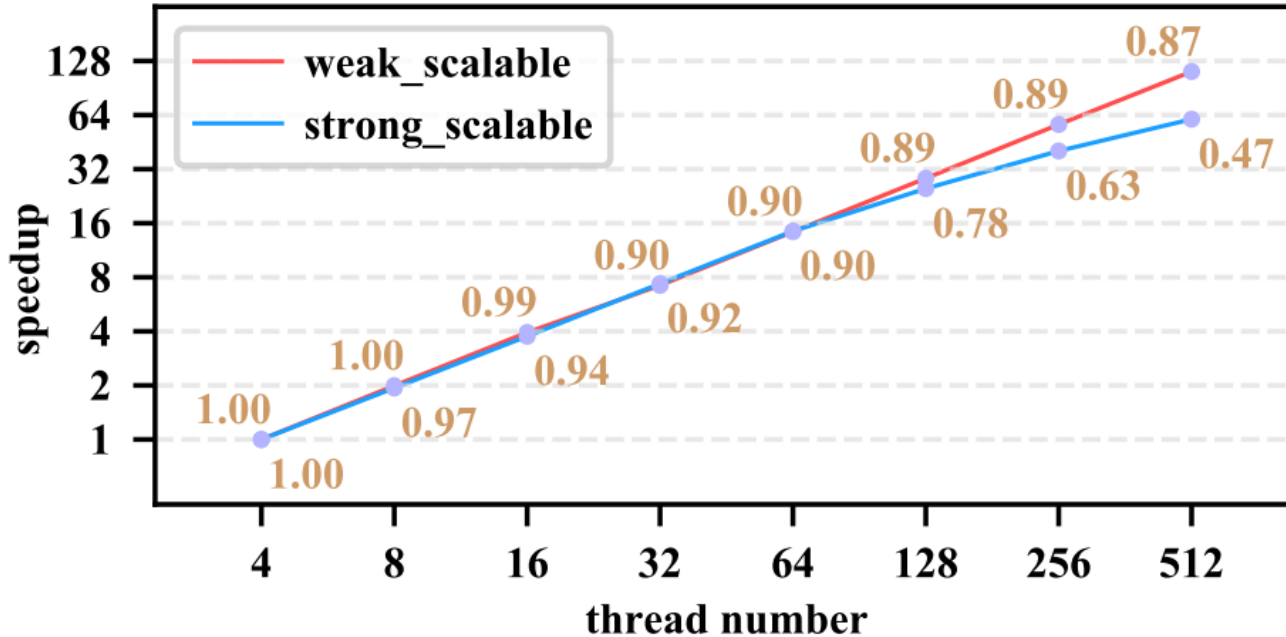
Acceleration of different benchmarks in one node in 1000 time steps

Performance Evaluation Compare with Other Strategies

Strategies	Target		Acceleration	
	appliciations	Interaction	Kernel	Entire
Hybrid Memory Update	GROMACS	Short-Range	16	5
Redundant Computation Approach	LAMMPS	L-J	32	16
Without Bit-Map	GROMACS	Short-Range	40	—
Bit-Map	GROMACS	Short-Range	63	30

Some related works in SUNWAY TaihuLight

Performance Evaluation Strong&Weak Scalable



The weak and strong scalability in different scales
In the strong scalability we keep the number of particles 1500K
In the weak scalability we keep every thread contains 6K particles

Equation to calculate
the strong and weak scalability

$$Eff_{weak}(N) = \frac{T_4}{T_N}$$

$Eff_{weak}(N)$: The weak scalability in N thread
 T_N : Time N thread spend

$$Eff_{strong}(N) = \frac{T_4}{\frac{N}{4} \times T_N}$$

$Eff_{strong}(N)$: The strong scalability in N thread
 T_N : Time N thread spend

Part **4**

Conclusion

Conclusion

- We accelerate the Short-Range interaction in GROMACS on SUNWAY-TaihuLight whose performance is much better than before
- The bandwidth of our implementation has achieved the peak bandwidth.
- We propose a new strategy to solve the write conflict with little performance loss. It could be use in many kind of programs with write-conflict.
- Our strategies could also be used in many other kinds Modular Dynamice applications, including LAMMPS, AMBER and so on

Part **4**

Future Work

Future Work

- We may change the cache strategy to reduce the cache miss rate.
- Change the size of our particle package.
- Optimize the calculation



Thanks !

Tingjian Zhang

张庭坚

sdubeyhhhh@gmail.com

sdubeyhhhh@163.com