Optimization of the GTC on Tianhe-2 supercomputer

About This Work

The Gyrokinetic Particle Simulation of Fusion Plasmas on Tianhe-2 Supercomputer

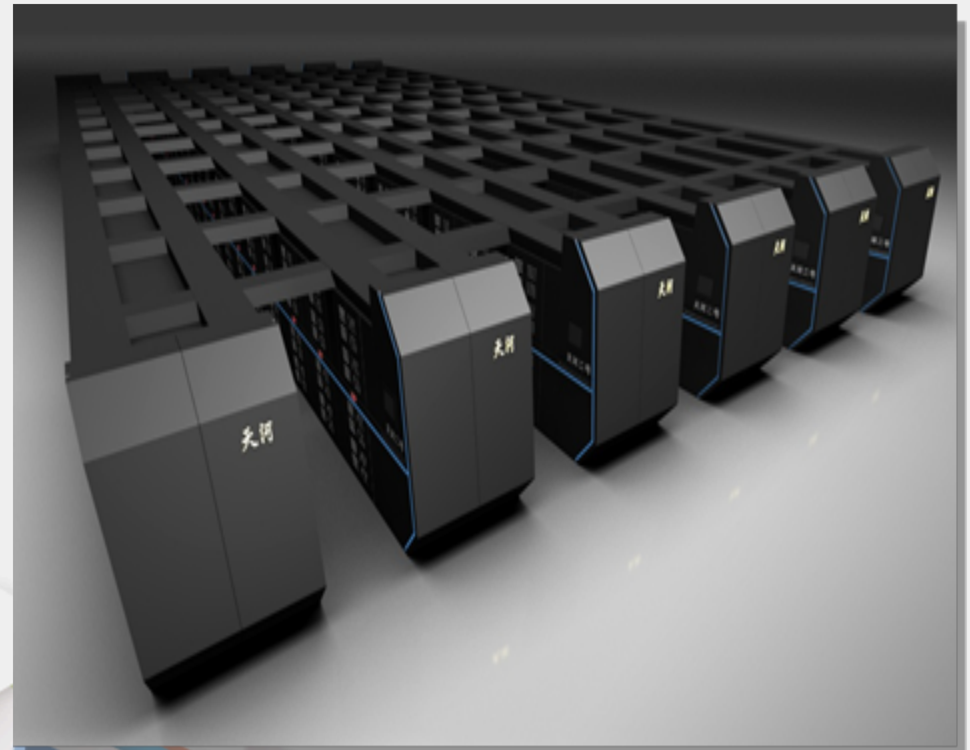
Endong Wang[1], Shaohua Wu[1], Qing Zhang[1], Jun Liu[1], Wenlu Zhang[2], Zhihong Lin[3,4], Yutong Lu[5,6], Yunfei Du[5,6], Xiaoqian Zhu[7]

[1] State Key Laboratory of High-end Server & Storage Technology, Inspur, Jinan, China
[2] Institute of Physics, Chinese Academy of Science, Beijing, China
[3] Department of Physics and Astronomy, University of California, Irvine, California, USA
[4] Fusion Simulation Center, Peking University, Beijing, China
[5] School of Computer Science, National University of Defense Technology, Changsha, China
[6] National Supercomputer Center in Guangzhou, Sun Yat-sen University, Guang Zhou, China
[7] Academy of Ocean Science and Engineering, National University of Defense Technology, Changsha, China

Inspur's achievements on the GTC optimization

- 2013.3-2014.12(CPU Computing)
  - Optimization of the MPI communication
  - Particle sorting
- 2015.6-2016.4 (CPU+KNC Computing)
  - Cooperation of CPU and MIC
    - Minimizing data transfer between host and co-processor
  - Vectorization of the Kernel functions
    - Pushe, pushi, chargei, et al.

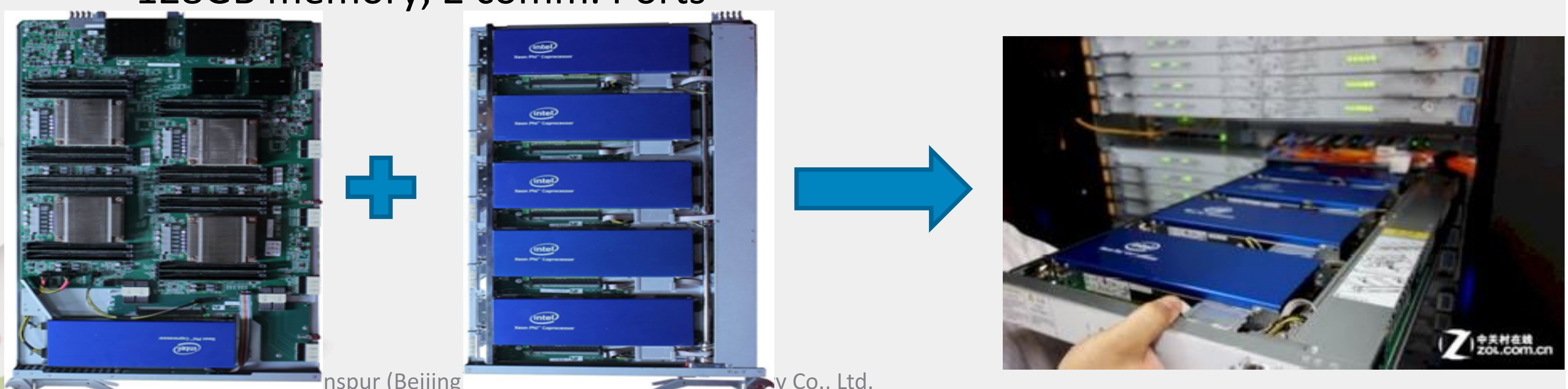
Tianhe-2 (MilkyWay II)



- No.1 @Top500 since 6, 2013
- Co-Developed by NUDT and Inspur

Compute blade of Tianhe-2

- 125 Rack
  - Each rack has 8 frame, each frame has 8 blade.
- Compute Blade
  - CPM module + APU module
  - 128GB memory, 2 comm. Ports



Analysing the bottleneck

- Data transfer between host and coprocessor
  - PCI-E v2: 16 GB/s
  - DDR Memory: ~70 GB/s
  - MIC Memory: 360 GB/s
- Random memory access
  - High cache miss rate
- Poor vectorization
  - Pushe, Pushi, Chargei, et al.

Hotspots

- Pushe
  - The force on each particle is interpolated from the grid points that surrounds the particle.
  - Then the particles are forces to move by solving the equations of motion with a second order low-storage Runge-Kutta scheme.
- Shifte
  - Pick the particles that need to be sent
  - Fill the hole (**sequential**)
  - Exchange the particles among MPI processes

Optimization methods- for Pushe

- Pushe
  - Improve cache hit rates by sorting
  - Sorting
    - A quick sort method given in [1] was implemented.
    - Particles were sorted by “jtgc0” for every 16 iterations.
  - Vectorizing all the loops
    - Merge the loops into a global one
    - Partition it to small blocks that fit the L1 cache

[1] K. J. Bowers, Accelerating a Particle-in-Cell Simulation Using a Hybrid Counting Sort. Journal of Computational Physics, 2001, 173: 393-411

Optimization methods-for Pushe

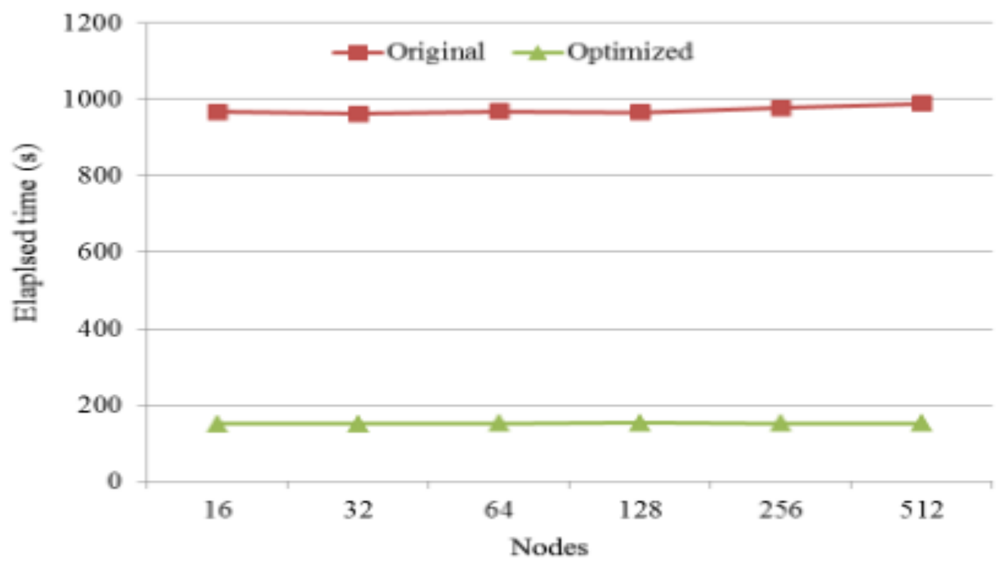


Fig. 4. Runtime of the pushe kernel before and after optimization. The "Original" indicates runtime of the original pushe kernel, while the "Optimzed" indicates runtime of the optimized pushe kerne.

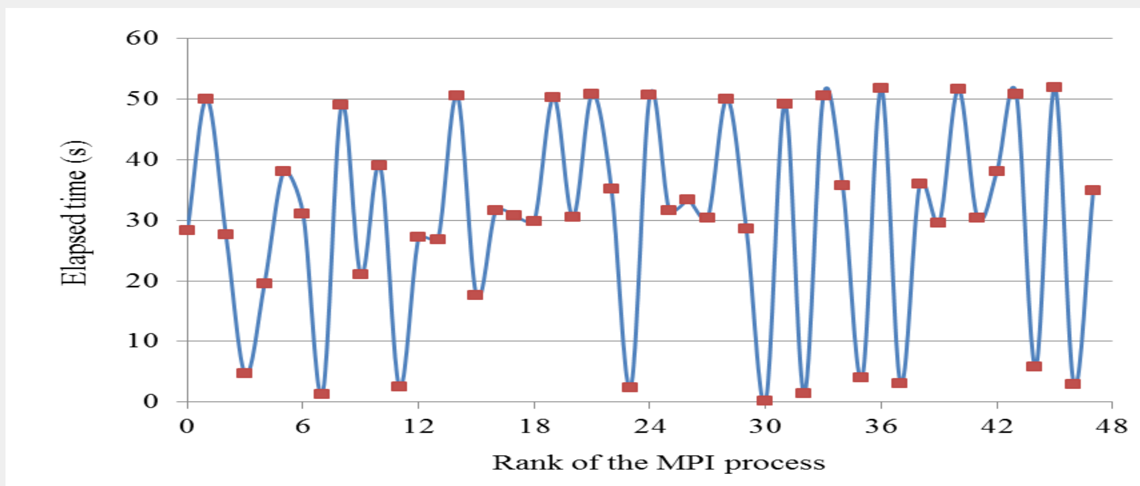
SPEEDUP:6.4X

Optimization methods- for Shifte

- MPI communication in the original code
  - It occurs among adjacent processes.

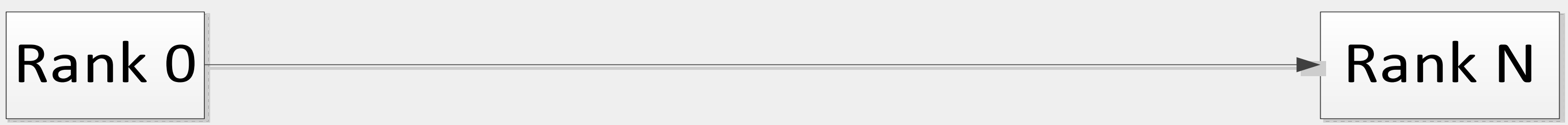
Rank 0 → Rank 1 → …… → Rank N-1 → Rank N

  - Imbalance



Optimization methods - for Shifte

- Optimization
  - The particles will be sent to the destination process directly.
  - Non-blocked communication.



Optimization methods - for Shifte

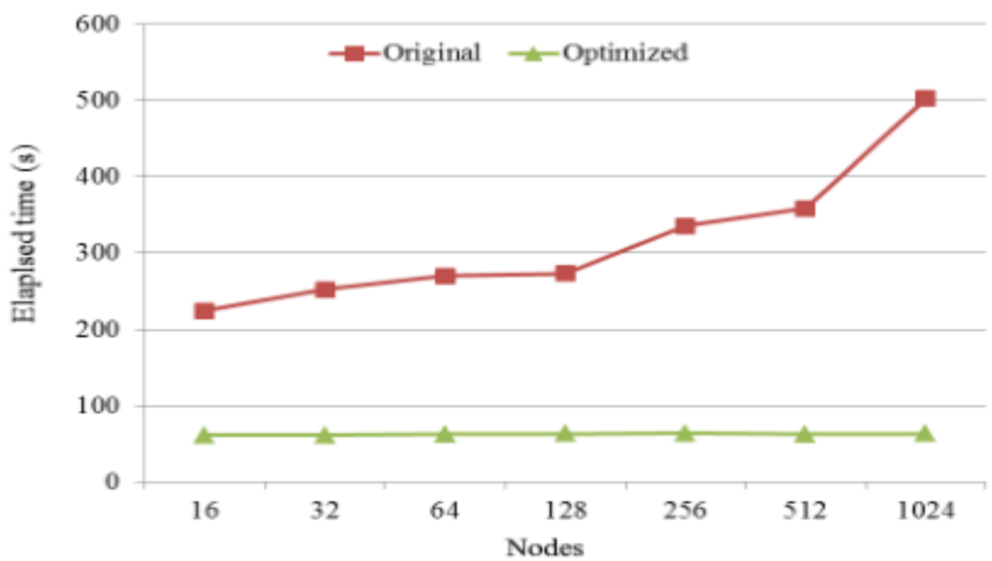


Fig. 3. Runtime of the shifte kernel before and after optimization. The "Original" indicates for the runtime of the original shift kernel, while the "Optimzed" indicates for the runtime of the optimized kernel.

7.9x performance improvement

Optimization methods - Others

- Data transfer between CPU and MIC
  - Particle arrays are decomposed between CPU and MIC
  - In the main loop, the CPU and Xeon Phi conduct particle computations concurrently. Only the particles that need to be exchanged among MPI processes are copied back to CPUs.
  - After the MPI communication, the particles receiving from other MPI processes are decomposed between CPUs and Xeon Phis again.

Testing results of Tianhe-2: Weak Scalability

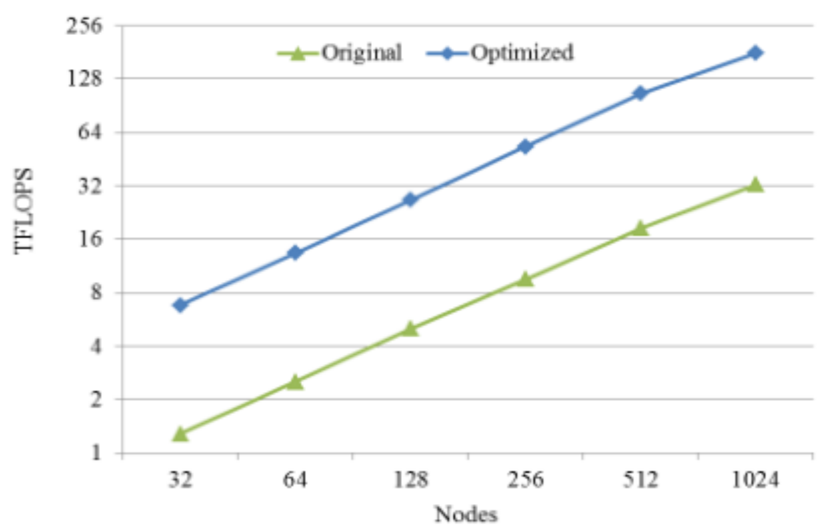


Fig. 5. Weak scaling of the GTC from 32 to 1024 Tianhe-2 nodes. The number of particles are increased proportionally to the number of nodes.

TABLE II. SPEEDUP ON TIANHE-2 FOR THE WEAK SCALING

Kernels	Nodes					
	32	*64*	*128*	*256*	*512*	*1024*
pushe	6.3	6.3	6.3	6.4	6.4	6.0
shifte	4.1	4.3	4.3	5.2	5.7	7.9
GTC	5.3	5.3	5.3	5.6	5.8	5.5

Testing results of Tianhe-2:Strong Scalability

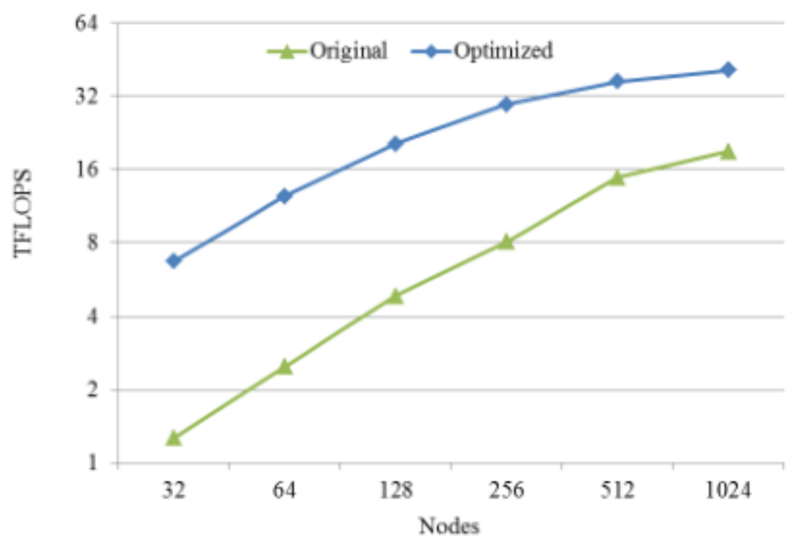


Fig. 6. Strong scaling of the GTC from 32 to 1024 Tianhe-2 nodes. The number of particles are fixed with the number of nodes increased.

TABLE III. SPEEDUP ON TIANHE-2 FOR THE STRONG SCALING

Kernels	Nodes					
	32	*64*	*128*	*256*	*512*	*1024*
pushe	6.4	6.2	5.9	5.3	4.2	2.8
shifte	4.1	3.8	2.7	2.8	2.0	2.0
GTC	5.3	5.0	4.2	3.6	2.5	2.2

Thank You