Supercomputer Simulations of Structure Formation in the Universe

Tomoaki Ishiyama
Chiba University, Japan
Outline

• How to perform large simulations on supercomputers?

• How to analyze big data generated from large simulations?
Cosmological simulation (dark matter only)

360 degree panoramic video for head mounted display is available on http://4d2u.nao.ac.jp/English/
Structures of dark matter halos

- Central Cusp
  - Einasto profile
  - NFW profile

\[ \rho(r) = \frac{\rho_S}{(r/r_s)[1 + (r/r_s)]^2} \]

- Numerous subhalo
  - \( \frac{dn}{dm} \sim m^{-(1.8-2)} \)
- Triaxial
- Non Universality
  - Weak dependence on the halo mass
  - halo to halo variation
    - halo formation epoch

Impact on the galaxy formation, Dark matter detection experiment

Ishiyama+ 2013
History of large cosmological $N$-body simulations (dm only)

Benchmark simulations are excluded.

![Graph showing the history of large cosmological $N$-body simulations (dm only). The graph plots the number of simulation particles against the year of each simulation. The x-axis represents the year from 1970 to 2020, and the y-axis represents the number of simulation particles from $10^2$ to $10^{14}$. The simulations are plotted as points on the graph, with markers indicating key years and simulation names such as Hubble, Bode, Ostriker, Millenium, and Horizon.](image-url)
History of large cosmological N-body simulations (dm only)

$10^{14-15}$ particles at 2020 (Exa era) ???

- Historically, the largest simulations in each era have been achieved using the largest supercomputers in that era.
- Together with the growth of supercomputers, always thinking about:
  - How to simulate?
  - How to analyze?
- This situation facilitates the development of new algorithm
How to increase the number of particles?

- Parallel computing on supercomputer
- Sophisticated numerical algorithm
  - Solves the forces from distant particles by multipole expansions
    (Tree method: Barnes and Hut, 1986) $O(N^2) \rightarrow O(N \log N)$

$$\frac{d^2 r_i}{dt^2} = \sum_{j \neq i}^N G m_j \frac{r_j - r_i}{|r_j - r_i|^3}$$
K computer

- **SPARC64™ VIIIfx, 2.0GHz octcore (128Gflops / CPU)**
  - Total 82944 nodes (663552 CPU core), 10.6 Pflops peak speed
- **16 GB memory / core, Total 1.3PB memory**
- **6D torus network**

World’s fifth fastest supercomputer

© RIKEN
Technical issues and our strategies

- Dense structures form everywhere via the gravitationally instability
  - Dynamic domain update with a good load balancer

- Gravity is the long-range force
  - Hierarchical communication

- $O(N \log N)$ is still expensive
  - SIMD
  - GPU, Intel MIC
Massively Parallel N-body code: GreeM

- TreePM poisson solver
  - Short-range: Tree
  - Long-range: Particle-Mesh (PM)
- Dynamic domain decomposition with good load balancer
- Novel communication algorithm for all-to-all communication
- Highly optimized gravity kernel with handy SIMD
- Flat MPI or MPI+OpenMP
- 2-10 times faster than Gadget-2 (Springel 2005)

Ishiyama, Fukushige, Makino, 2009
Ishiyama, Nitadori, Makino, 2012
Dynamic domain decomposition

- △ Space filling curve
- ○ multi section
  - Enables each node to know easily where to perform short communications
- × equal #particles
- △ equal #interactions
- O equal calculation time

Springel 2005

Gadget-2

Ishiyama, Fukushima, Makino, 2009
Hierarchical communication algorithm

Ishiyama, Nitadori, Makino, 2012

- Large all-to-all communication occurs on PM (for FFT)
- Alltoallv communication is difficult to accelerate by both of software and hardware
- The size of traffic is not uniform
- Difficult to find best route
- Strongly depend on the performance of network used

- Split alltoallv by each dimension
  - Avoid skew communication and network congestion
Gravity kernel

An extension of Phantom-GRAPE
(Nitadori+ 2006, Tanikawa+ 2012a, 2012b)

\[
g_{P3M}(R) = \begin{cases} 
1 - \frac{1}{140} \left(224R^3 - 224R^5 + 70R^6 + 48R^7 - 21R^8\right) & (0 \leq R \leq 1) \\
1 - \frac{1}{140} \left(12 - 224R^2 + 869R^3 - 840R^4 + 224R^5 + 70R^6 - 48R^7 + 7R^8\right) & (1 \leq R \leq 2) \\
0 & (2 \leq R)
\end{cases}
\]

1. One if-branch is reduced
2. optimized for a SIMD hardware with FMA

\[
g_{P3M}(R) = 1 + R^3 \left(\frac{8}{5} + R^2 \left(\frac{8}{5} + R \left(-\frac{1}{2} + R \left(-\frac{12}{35} + R \frac{3}{20}\right)\right)\right)\right)
- S^6 \left(\frac{3}{35} + R \left(\frac{18}{35} + R \frac{1}{5}\right)\right) & (0 \leq R \leq 2) \\
S \equiv \max(0, R - 1),
\]

- Implemented with SIMD builtin functions (Fujitsu C++ compiler)
- Unrolled eight times by hand

Typically, automatic vectorization by compilers is insufficient
Gravity kernel

- 17 FMA and 17 non-FMA operations for a interaction
  - Theoretical limit is 12Gflops/core
  - (16Gflops for peak)

- **11.65** Gflops on a simple kernel benchmark
  - **97%** of the theoretical limit (73% of the peak)
Highlight results

Performance results on K computer

Scalability ($2048^3 - 10240^3$)
- Excellent strong scaling
- $10240^3$ simulation is well scaled from 24576 to 82944 (full) nodes of K computer

Performance ($12600^3$)
- The average performance on full system is $\sim 5.8$ Pflops,
- $\sim 55\%$ of the peak speed

Ishiyama et al. 2012 (arXiv: 1211:4406), SC12 Gordon Bell Prize Winner
Comparison with other codes

- $1024^3$, 320Mpc/h, 512CPU cores
  - GreeM 20,763 sec, Gadget-2 44,752 sec
- $512^3$, 1Gpc/h, 256CPU cores
  - GreeM 1,678 sec, Gadget-2 3,577 sec
- $512^3$, 21Mpc/h, 256CPU cores
  - GreeM 10,756 sec, Gadget-2 62,005 sec

Our code is 2-10 times faster than Gadget-2

- GreeM (Ishiyama et al. 2012) on K computer (10.6Pflops)
  - $2.54\times10^{-11}$ sec / substep / particle
- HACC (Habib et al. 2012) on BG/Q 48 rack (10Pflops)
  - $1.07\times10^{-10}$ sec / substep / particle

Our code is ~4 times faster than HACC
$N = 8192^3 = 549,755,813,888$ particles

$L = 1.12 \text{ Gpc/h}$

$m = 2.2 \times 10^8 \text{ Msun/h}$

Planck Cosmology

11x larger volume, 4x better mass res, compared to Millennium Run (Springel+ 2005)

$N = 2160^3$

$L = 0.5 \text{ Gpc/h}$

$m = 8.6 \times 10^8 \text{ Msun/h}$

WMAP1 ($\sigma_8=0.9$)

~ 100 hours on 131,072 CPU cores of K computer

Data size : ~1PB

Ishiyama et al., PASJ, 2015

http://hpc.imit.chiba-u.jp/~nngc
Data size

- $8192^3$ particles → 16TB/snapshot
  - 3D positions, velocities are stored as float
  - Particle ID is stored as 64bit integer
- Typically > 50 snapshots are needed to construct halo merger histories
  - ~ $1$PB / $8192^3$ simulation → **Could not perform many runs**

- There are some data reduction techniques
  - On the fly analysis
    - Halo finding, power spectrum
  - Irreversible compression
    - Positions → 24 bit fixed integer
    - Velocities → 16 bit float
    - ID → 40 bit (insufficient for $>10321^3$ simulations…)

32bytes/particle

20bytes/particle
Big data analysis

• Basically, heavy analysis are done on supercomputers
  • Halo finding (fof)
  • Merger tree construction
  • power spectrum
• We have to parallelize codes for analysis
• Sometimes tuning for different architectures is needed
  • Intel, Power, Sparc …
  • GPU, Xeon-Phi...
• On further analysis (e.g., density structures of halos, tracking orbits…), trial and error is needed
  • Analysis on supercomputers is annoying
    • Job does not run immediately
Efficient IO access

- It is best to analyze on one server if possible
- But one snapshot uses 16TB, reading all particles takes 160,000 sec via 1Gbps network
- Splitting one snapshot into many files according to positions
- Only read files including particles in interested regions

0 (IO format)
\( n \) (Number of particles)
0 0.5 0 0.5 0 0.5 (position info)
. (header)
. particle 0 (position, velocity, ID)
particle 1
. .
particle \( n-1 \)
Data transfer

- If 100 Mb/s network is available
  - ~10TB / day
  - ~100 days / 1PB
- Typically, effective speed is less than 10Mb/s
  - < 1TB / day
  - > 3 years / 1PB
Data transfer

• If 100 Mb/s network is available
  • ~10TB / day
  • ~100 days / 1PB
• Typically, effective speed is less than 10Mb/s
  • <1TB / day
  • >3 years / 1PB ……
• Delivery by car
  • 3 days / 1PB
  • From Kobe to Chiba
    (from Kyoto to Tokyo + 100km, ~600km journey)
Summary

- We developed a massively parallel N-body code
  - ~5.8Pflops is achieved on K computer, which corresponds to ~55% of the peak speed
  - Enable to perform gravitational trillion-body problem within practical time
  - We could perform simulations, 2-10 times faster than public codes like Gadget-2

- Data size limits the number of large simulations
- Data transfer is becoming major bottleneck
- 1PB might be the limit that one person can treat (in my personal opinion)