ciuk_new_bg

01/12/2019 09:55

Gas Density

Temperature

Dark Matter

Shocks

Enrichment

SWIFTsimIO

Post-processing the next-generation of cosmological simulations

Josh Borrow

Institute for Computational Cosmology, Durham University

Movitation

We run **large-scale simulations** of the universe with up to **100 billion particles**.

Largest simulations now producing **petabytes of data**.

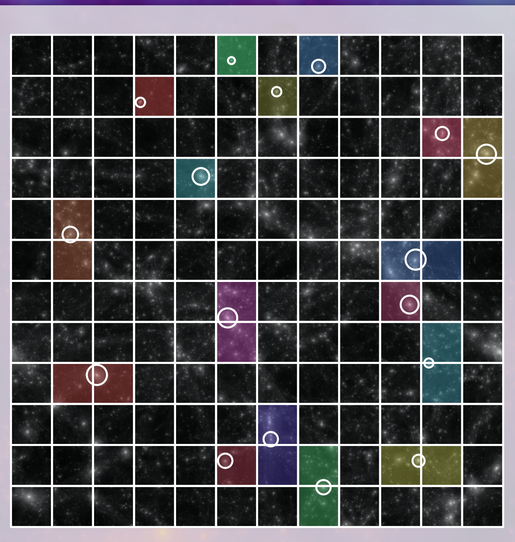
Individual snapshots are around **10 TB**, and represent a **huge dynamic range**.

However, **individual objects** (galaxies) in these simulations **only represent < 1 GB** of data.

Need to **efficiently extract** these objects!

0

Metadata



Key to solving **big data challenges**: producing enough **metadata** to efficiently slice the data at a later stage.

Physicists think spatially – package (very cheap) **spatial metadata with outputs**.

Run **on-the-fly object finders** to deal with huge **dynamic range**, along with a **top-level grid**.

Store each file **ordered by top-level cell**.

Figure 1: Left shows the top-level cell grid (projected in 2D) of a typical cosmological volume simulated with the SWIFT code. Objects identified by the on-the-fly object finder are shown as white circles, with the top-level cells identified by swiftsimio to read from the snapshot highlighted in various colours. This reduces the data size sigfnicantly; each cell contains only around a hundred thousand particles.

1

Reading Data

Metadata is **stored for every object** in the simulation, including properties such as mass, size, temperature, etc. such that **often it is not necessary to go back to the particle data**.

When it is necessary, thanks to the **spatial metadata**, the **time to read** a fixed volume of data is **completely independent** of the **size of the dataset** (see Figure 2).

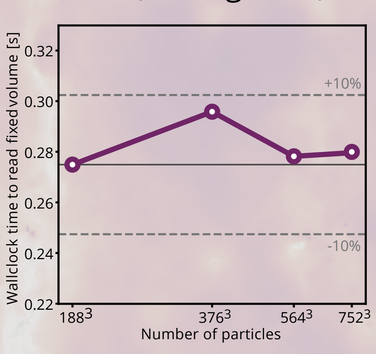


Figure 2: Left shows the (constant) time to access a fixed volume of data (here much larger than any object in a typical simulation) as a function of the size of the dataset. The rightmost dataset represents over 100 Gb of particle data. The dashed lines show a range of ±10% in read time.

2

Visualisation

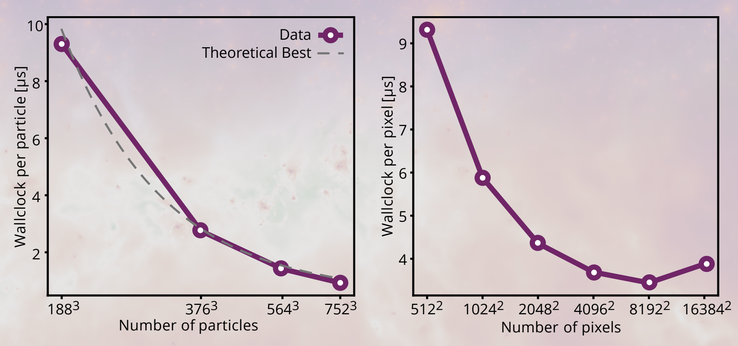


Figure 3: The left panel shows the cost of making a fixed 4096x4096 image of a dataset of different sizes. The cost per particle actually decreases as the (particle) resolution increases as each particle is smoothed over fewer pixels. `swiftsimio` shows very close agreement to theoretical best scaling here. The right panel shows how the cost per pixel scales as a function of the image resolution for a fixed particle count (188^3). This should be constant, but overheads dominate for small images, with large images generally being cheaper.

Once loaded, **very cheap to visualise** data.

Accelerated routines with `numba` to produce SPH-smoothed visualisations.

See background for examples!

Visualisation is **not just for** making pretty **pictures**; **projected quantities** map directly to **astronomical observables**.

3

Conclusion

`swiftsimio` allows users of the SWIFT simulation code deal with **huge snapshots trivially** through the use of **spatial metadata**.

It turns a **petascale big data** analysis problem into something **simple** to perform even on a **laptop computer**.

The code is **available** on GitHub (`swiftsim/swiftsimio`) and on PyPI.

4

Durham University

DiRAC

5 million light-years