Building Bigger Boxes: Towards True Cosmological Volumes with EAGLE-XL

Josh Borrow

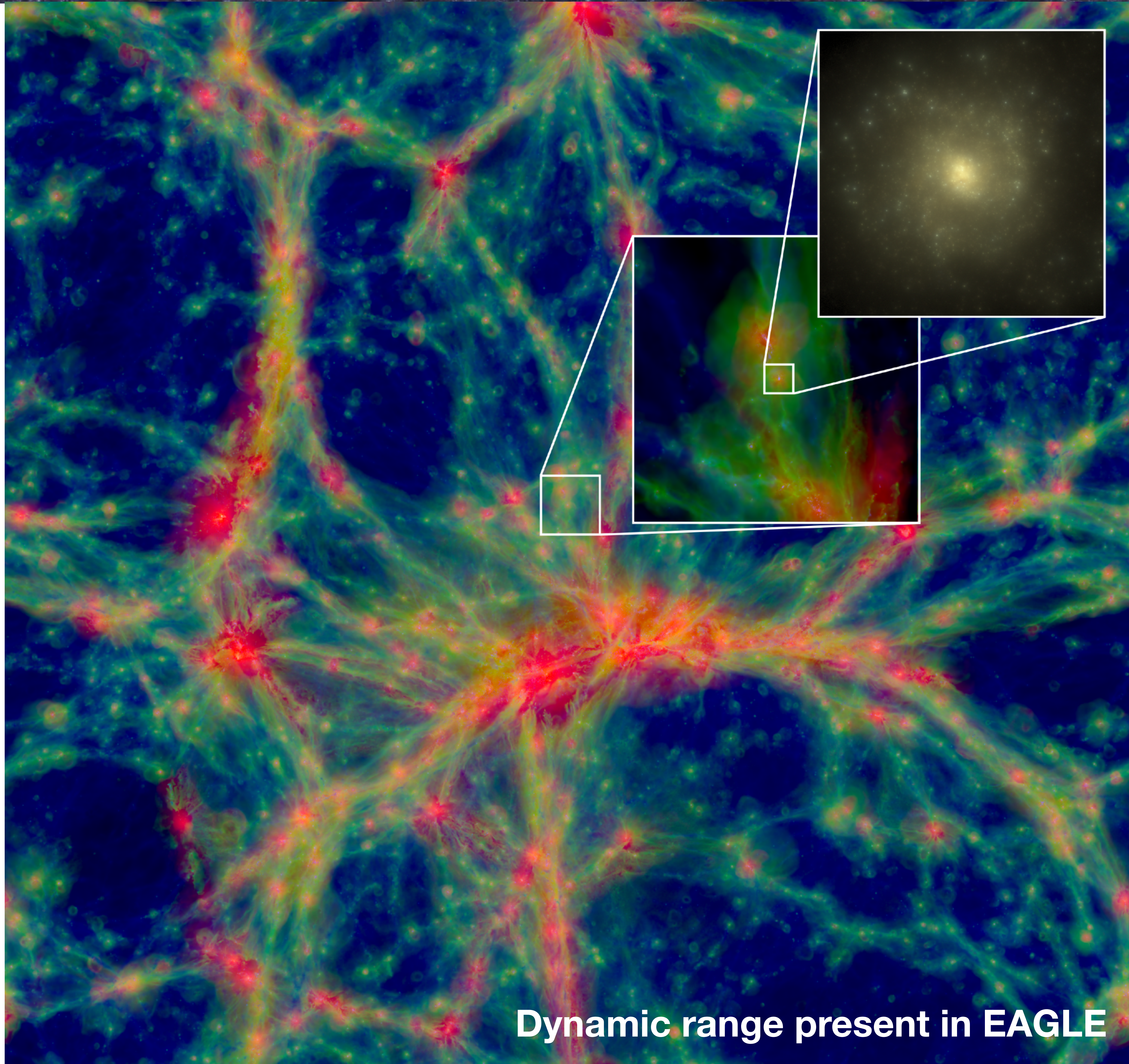
ICC, Durham

& Matthieu Schaller, Richard Bower, Peter Draper, Pedro Gonnet, James Willis, Alexei Borrisov, David Barnes, Joop Schaye …

MIT, Cambridge MA | 22nd April 2019

Simulations of the Universe

- Illustris-TNG, EAGLE, Horizon-AGN, SIMBA, ...
- All aim to solve the same problem: simulate both the large-scale structure of the Universe and the physics central to galaxy formation
- State-of-the-art typically use ~3-10 B resolution elements
- Always want bigger and better.



What do they teach us?

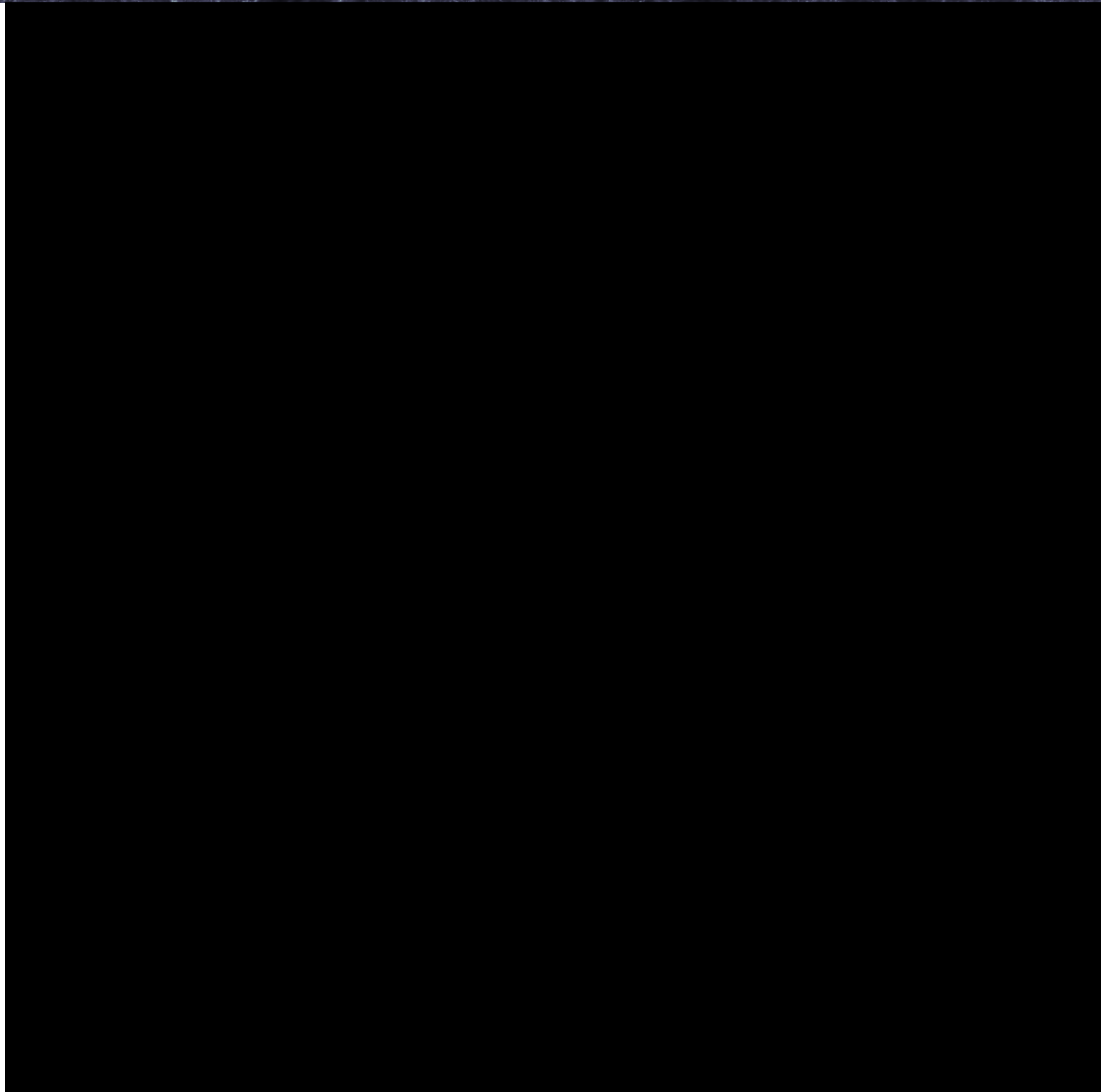
- Allow us to track individual galaxies over time

- "Real" astronomers can only see a snapshot of the Universe

- We can e.g. see how different types of galaxies evolve differently, how a spiral turns into an elliptical...

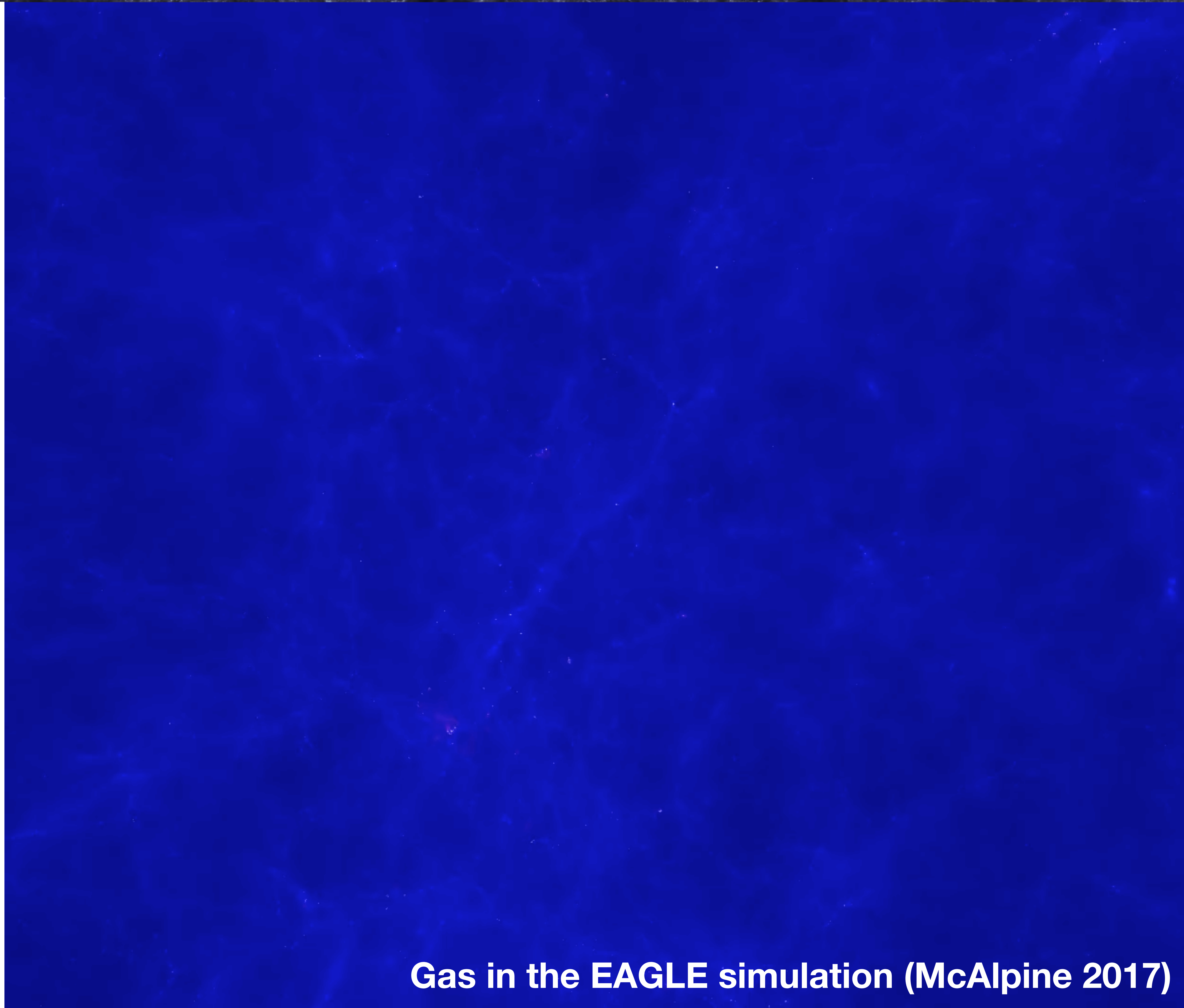
What do they teach us?

- Allow us to track individual galaxies over time
- "Real" astronomers can only see a snapshot of the Universe
- We can e.g. see how different types of galaxies evolve differently, how a spiral turns into an elliptical...



The EAGLE Project Code

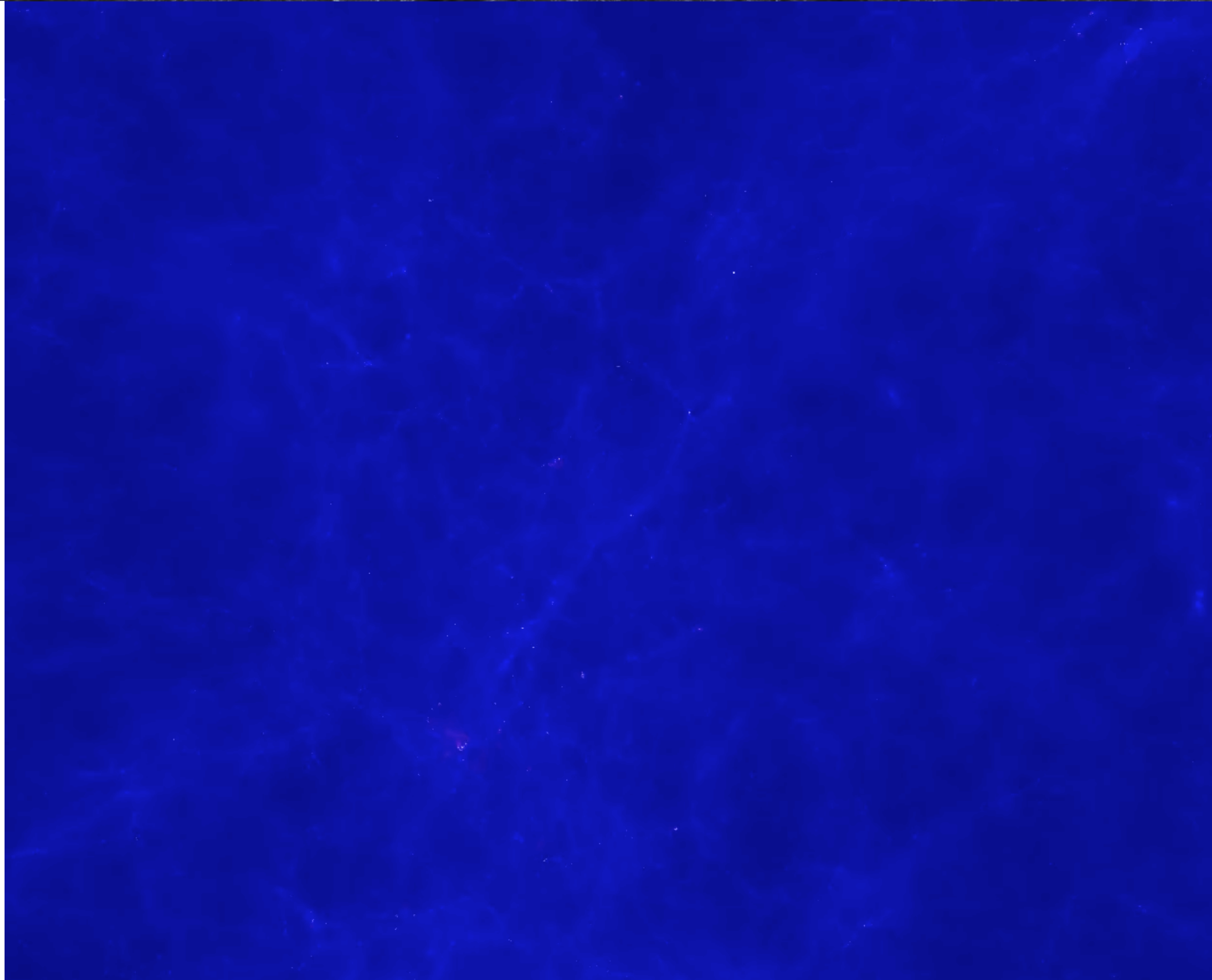
- Built on P-Gadget3
- Uses SPH for hydrodynamics, Tree-PM method for gravity
- Designed to run on ~512 single-core nodes (MPI only).
- **3.5 Bn gas particles**
- EAGLE used 16x256 cores for ~40d of wall-clock time.



Gas in the EAGLE simulation (McAlpine 2017)

The EAGLE Project Code

- Built on P-Gadget3
- Uses SPH for hydrodynamics, Tree-PM method for gravity
- Designed to run on ~512 single-core nodes (MPI only).
- **3.5 Bn gas particles**
- EAGLE used 16x256 cores for ~40d of wall-clock time.

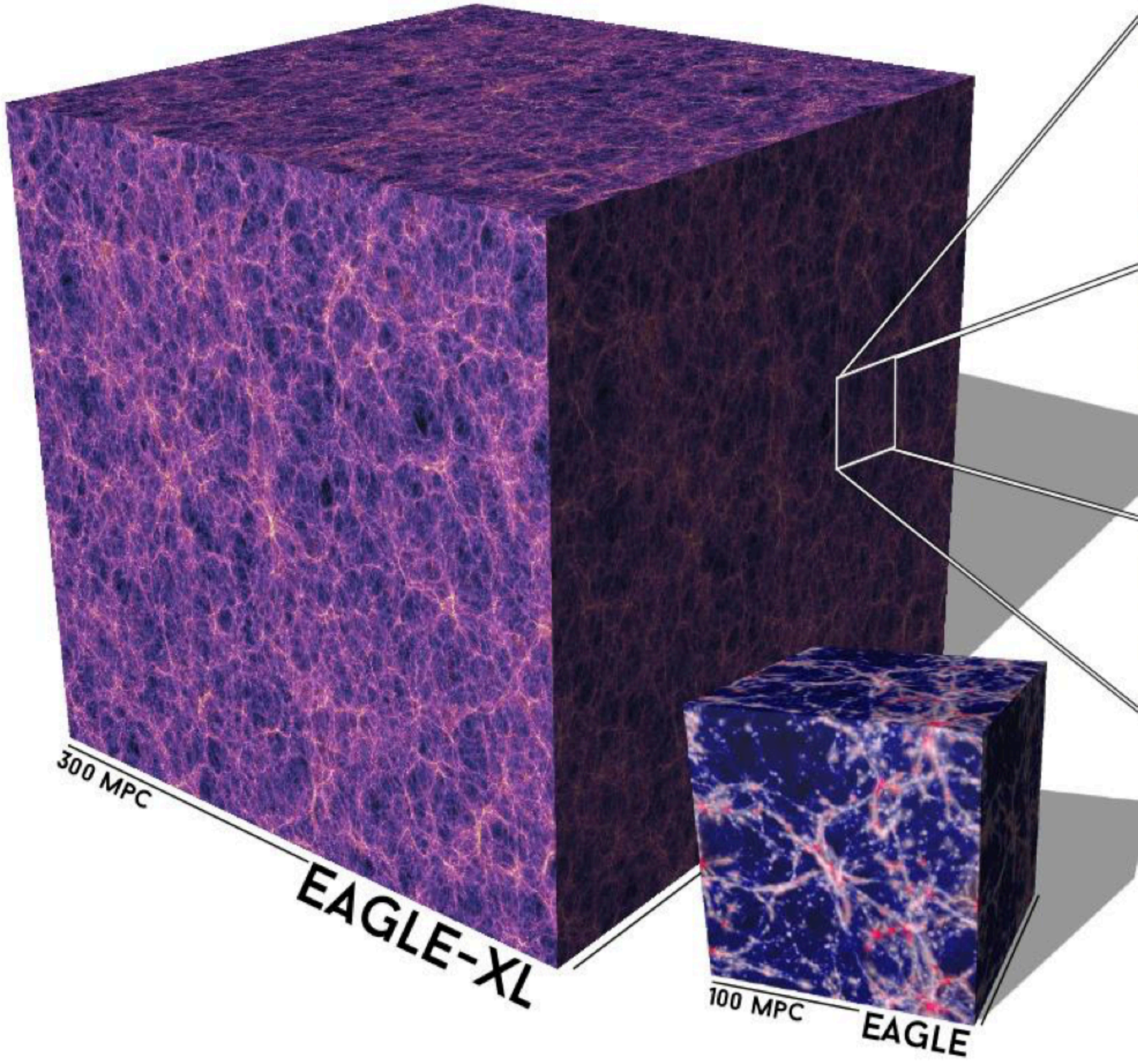


Gas in the EAGLE simulation (McAlpine 2017)


EAGLE-XL

EAGLE-XL

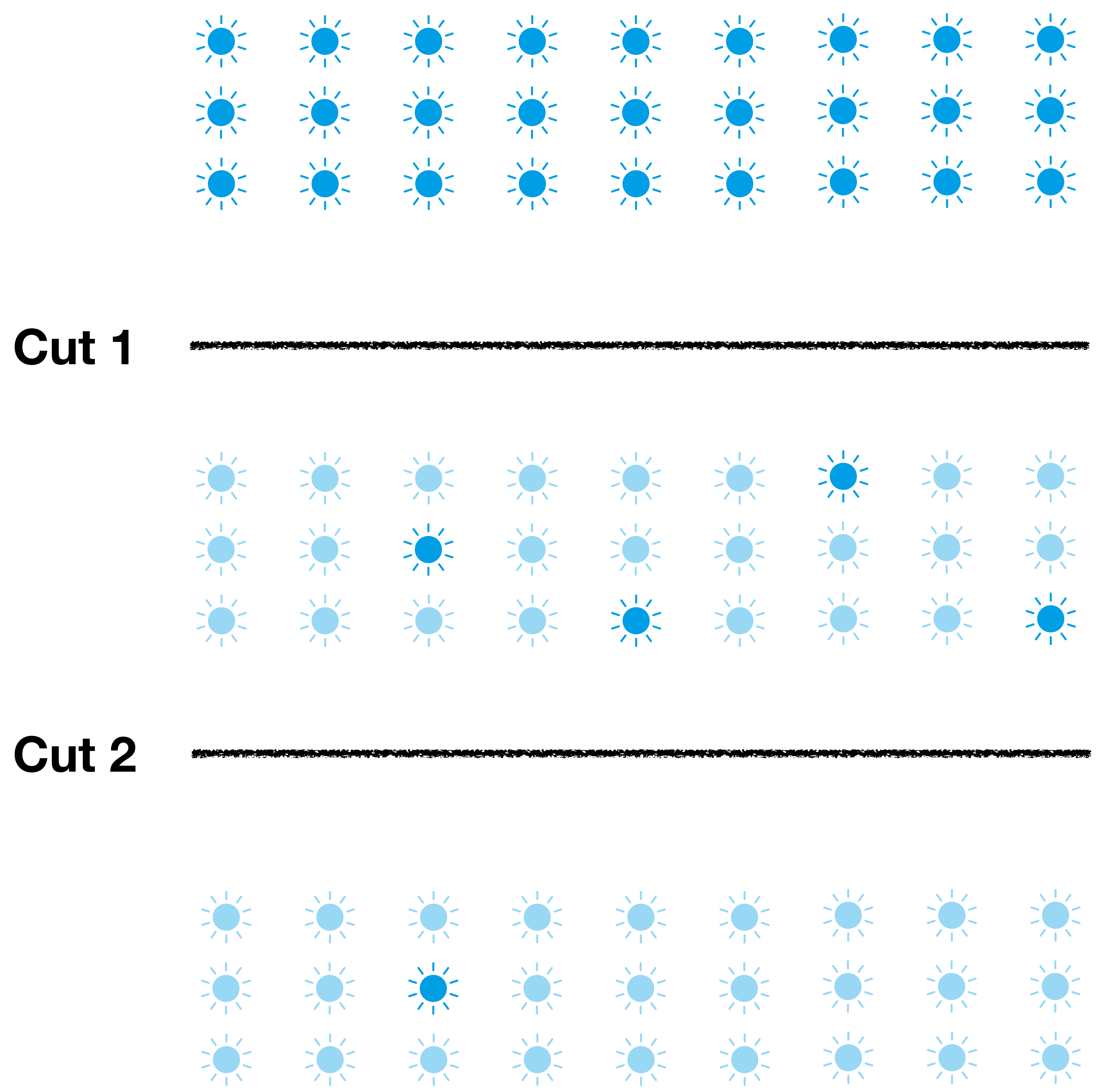
- Same resolution as original EAGLE
- Baryonic particles with mass 10^6 solar masses
- 27x the volume, 27x the particles.
- **~91 billion gas particles** - 300 B particles to integrate.



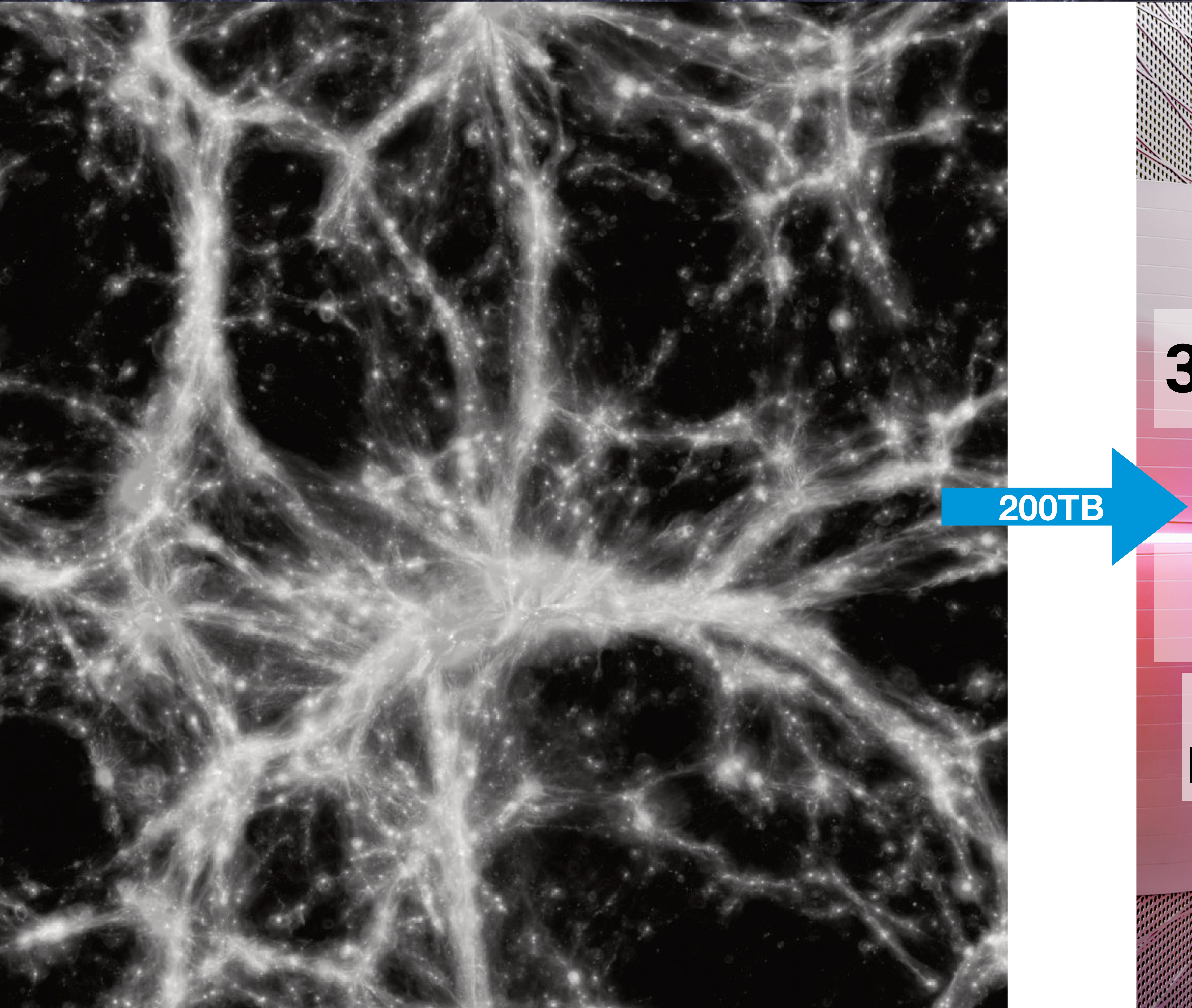
But why?

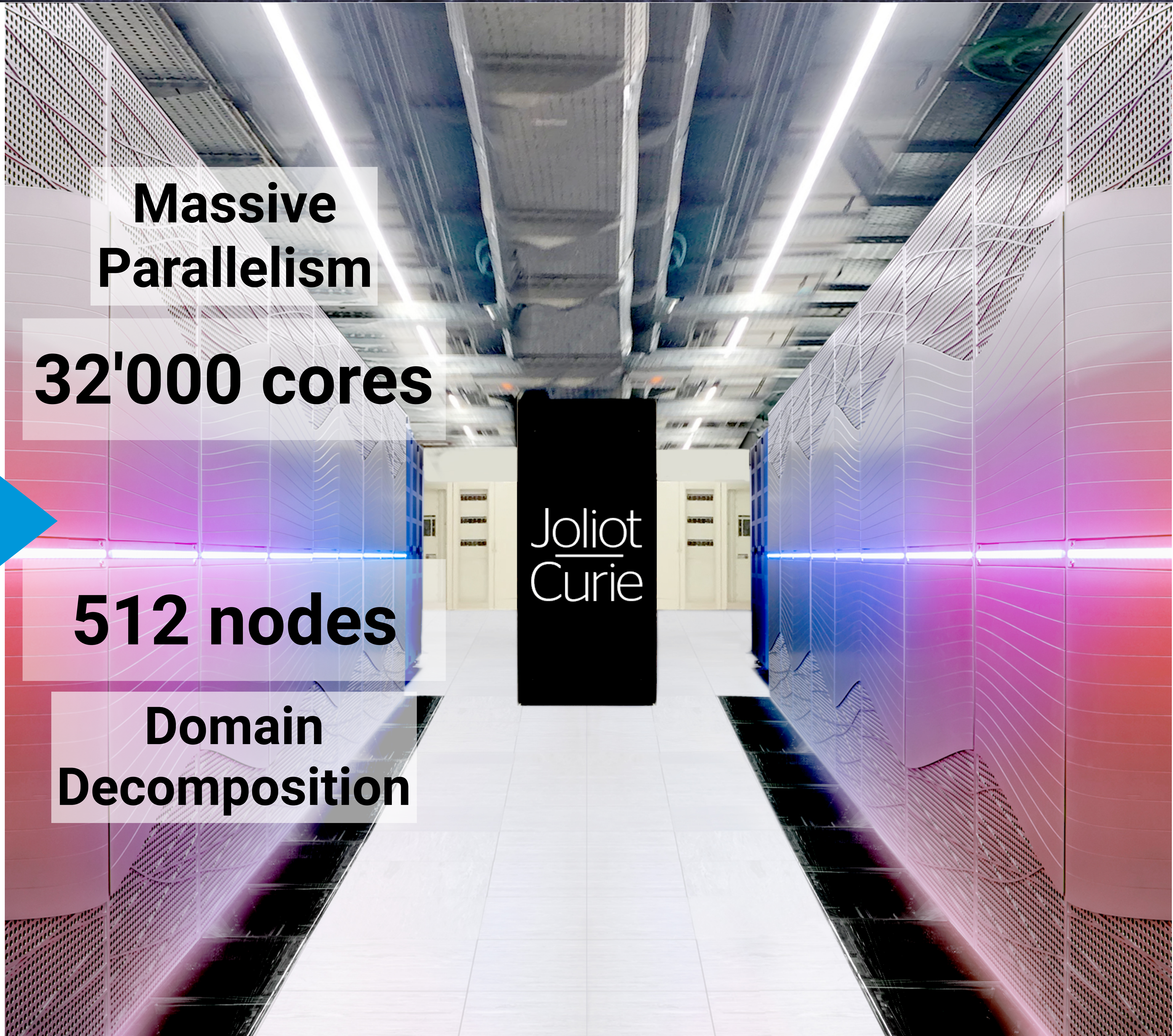
- **Statistics**: need to cut many times in different dimensions, leaving e.g. 2 galaxies in a given bin; imagine cutting 1000 galaxies by 10% four times for independent variables.

- **Cosmology**: large-scale effects of galaxy formation are felt out to at least ~15 Mpc; this is over 10% of the box-size of previous runs.



The EAGLE-XL challenge





Our Options

Run on same number of nodes

Requires very large nodes.

Original nodes had
192 Gb of RAM

These would need 6 Tb each

Would also need 3 years+ of wallclock time

Run on many more nodes

Needs a code designed to run on 512 processors to run on 16'0000

Requires us to be able to weak-scale

Would require a huge amount of architecture work

At least ~200 M CPU Hours

Our Options

Run on same number of nodes

...s very large...

Orig... ...ad
192 ... AM

Thes... ...d need ... each

Woul... also need 3 yea...+ of
wallclock time

Run on many more nodes

Needs a code designed to run on 512 processors to run on 16'0000

Requires us to be able to weak-scale

Would require a huge amount of architecture work

At least ~200 M CPU Hours

Our Options

Run on same number of nodes

...es very large...

Orig... ...ad
192 ... AM

Thes... ...d need... ...each

Woul... also need 3 yea...s+ of
wa... ...me

Run on many more nodes

Ne... ...ode design... ...n on
51 2 pr... ...rs to r... ...16'0...00

Requires us... ...ble to weal...

Wo... ...equire a huge a... ...nt of
...architecture wor...

At least ~200 M CPU Hours

Our Options

Run on same number of nodes

…es very large…

Orig… …ad
192 … …AM

Thes… …d nee… …each

Woul… also need 3 yea… + of
wallclock time

Run on many more nodes

Ne… …ode design… …n on
5…2 pr… …ors to r… …16'0…00

Requires us… …ble to weal…

Wo… …equire a huge a… …nt of
architecture wor…

Nuclear option

Throw everything out and start again

Start with an empty source file and design the code for modern architectures

Requires ~years of human time

At least ~200 M CPU Hours

Our Options

Run on same number of nodes

...s very large...

Orig... ...had
192 ... ...AM

Thes... ...d need... ...each

Would also need 3 years+ of
wallclock time

Run on many more nodes

Ne... ...ode design... ...n on
5...2 pr... ...rs to r... ...16'0...00

Requires us... ...ble to weal...

Wo... ...equire a huge a... ...nt of
...architecture wor...

At least ~200 M CPU Hours

Nuclear option

Thro... ...again

St... ...sour... ...nd
...de fo...
...nitect...

Req... ...ime

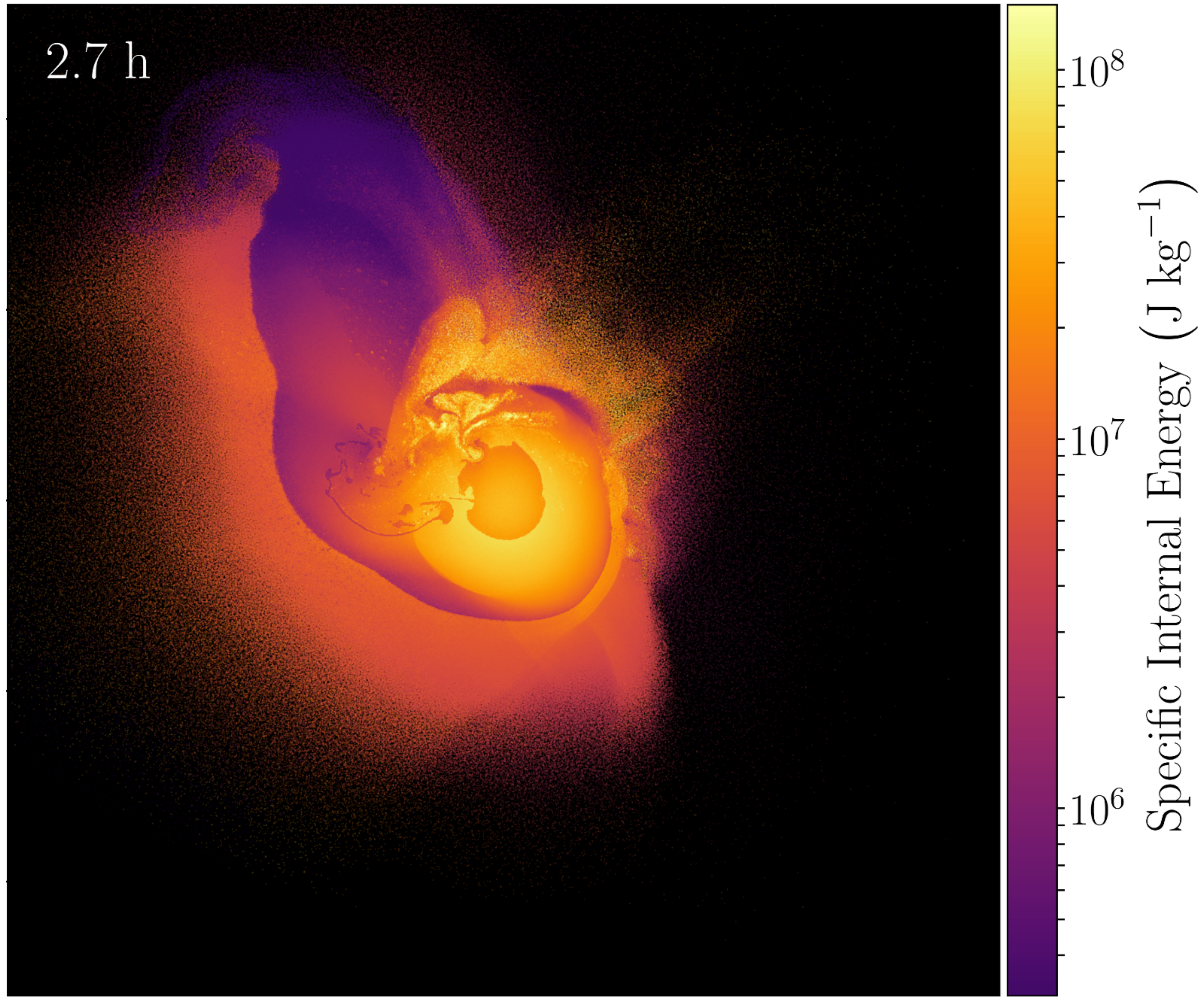
Turns out, humans are cheaper than computers...


SWIFT

SWIFT

- SWIFT (SPH With Inter-dependent Fine-grained Tasking) is a brand new, **open source** (and open development!) code for astrophysics.

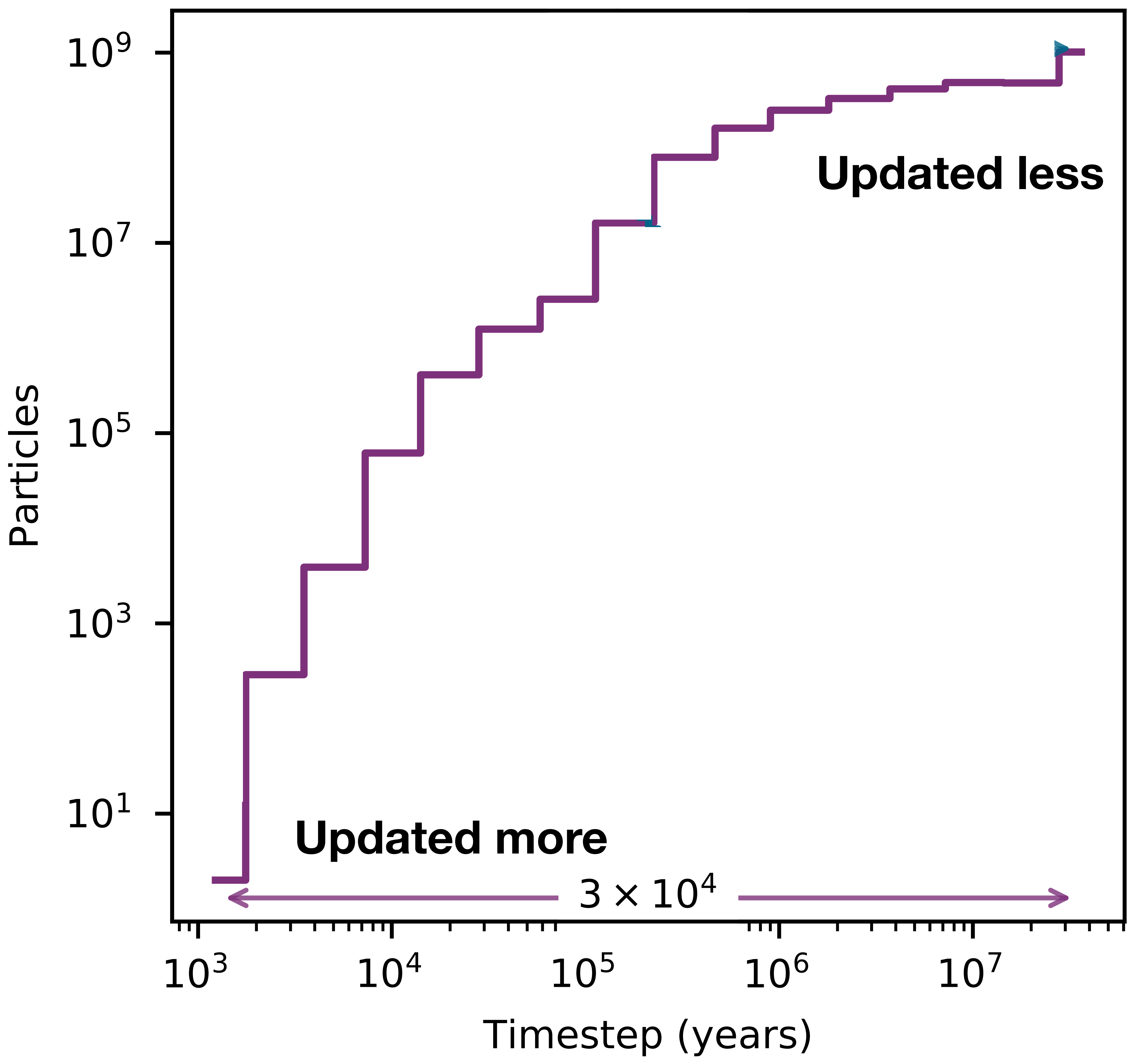
- Designed primarily for cosmological simulations, but adaptable enough to do e.g. planetary impacts.



The highest resolution planetary impacts simulation ever performed, made possible with SWIFT
Kegerreis+, 2019

A quick note on adaptivity

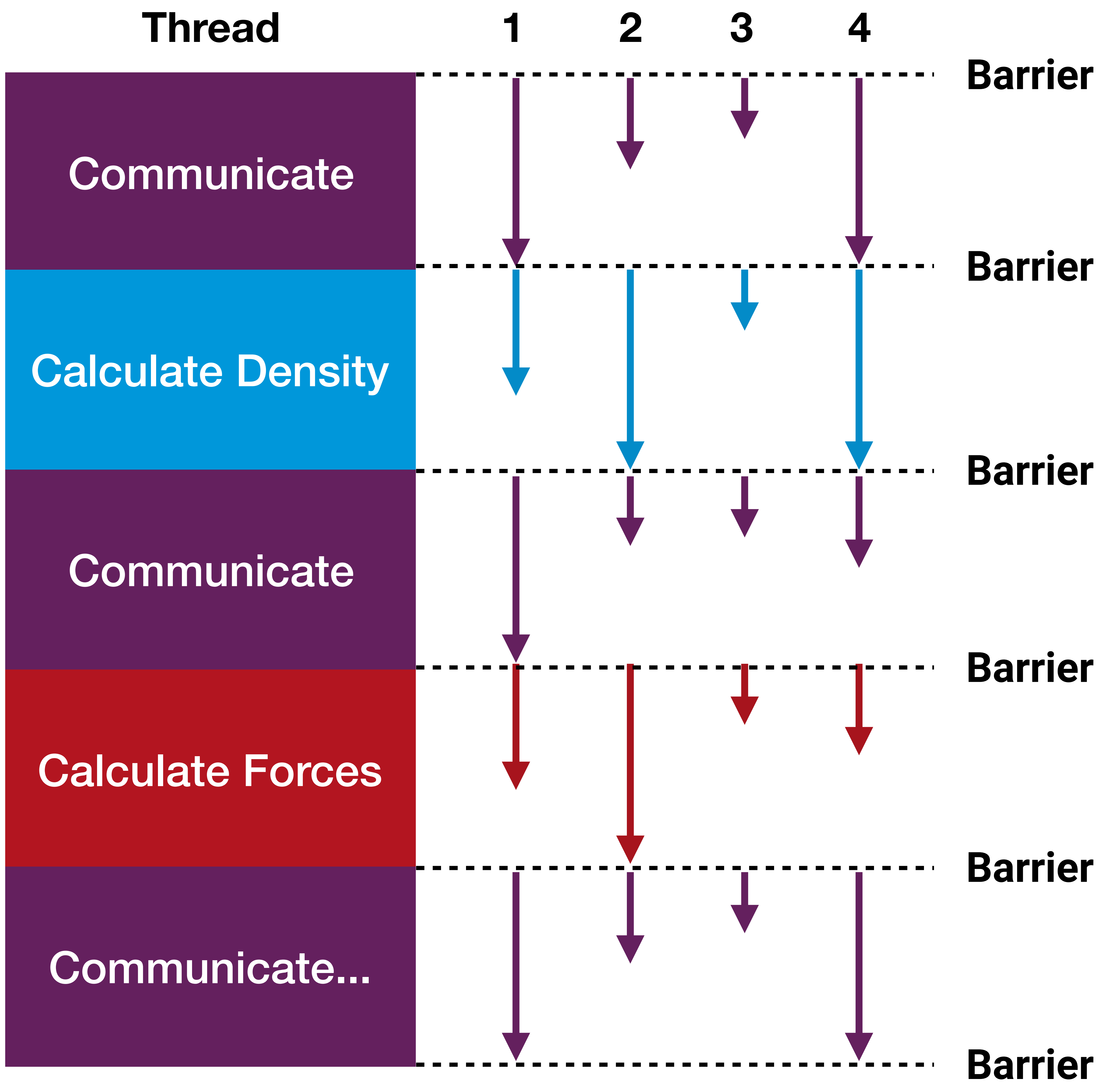
- 10^{11} range in density within the simulation
- Leads to 10^4 range in time-step!
- Vast majority of particles are in under-dense regions
- Very few particles need to be updated frequently, but we have lots overall!



Time-step variation in the EAGLE-50 volume. Borrow+, SPHERIC 2018, 2018

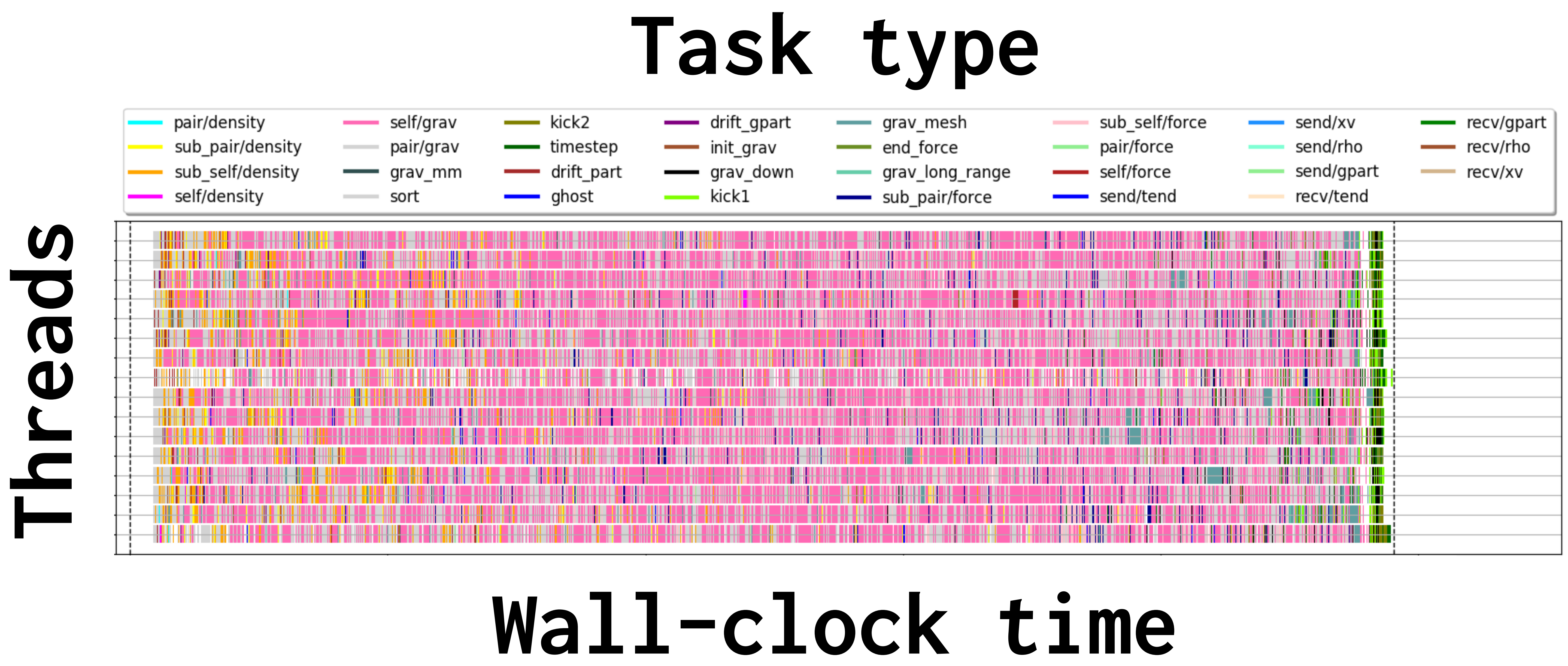
Parallelism Strategy

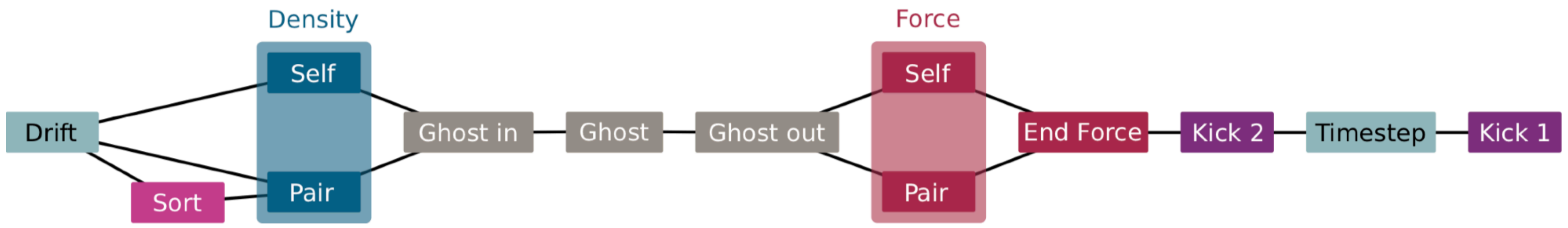
- Typical strategy is 'branch-and-bound'
- Can lead to large differences in compute time per rank (thread)
- Hard to load-balance multiple operations



Modern Architechtures

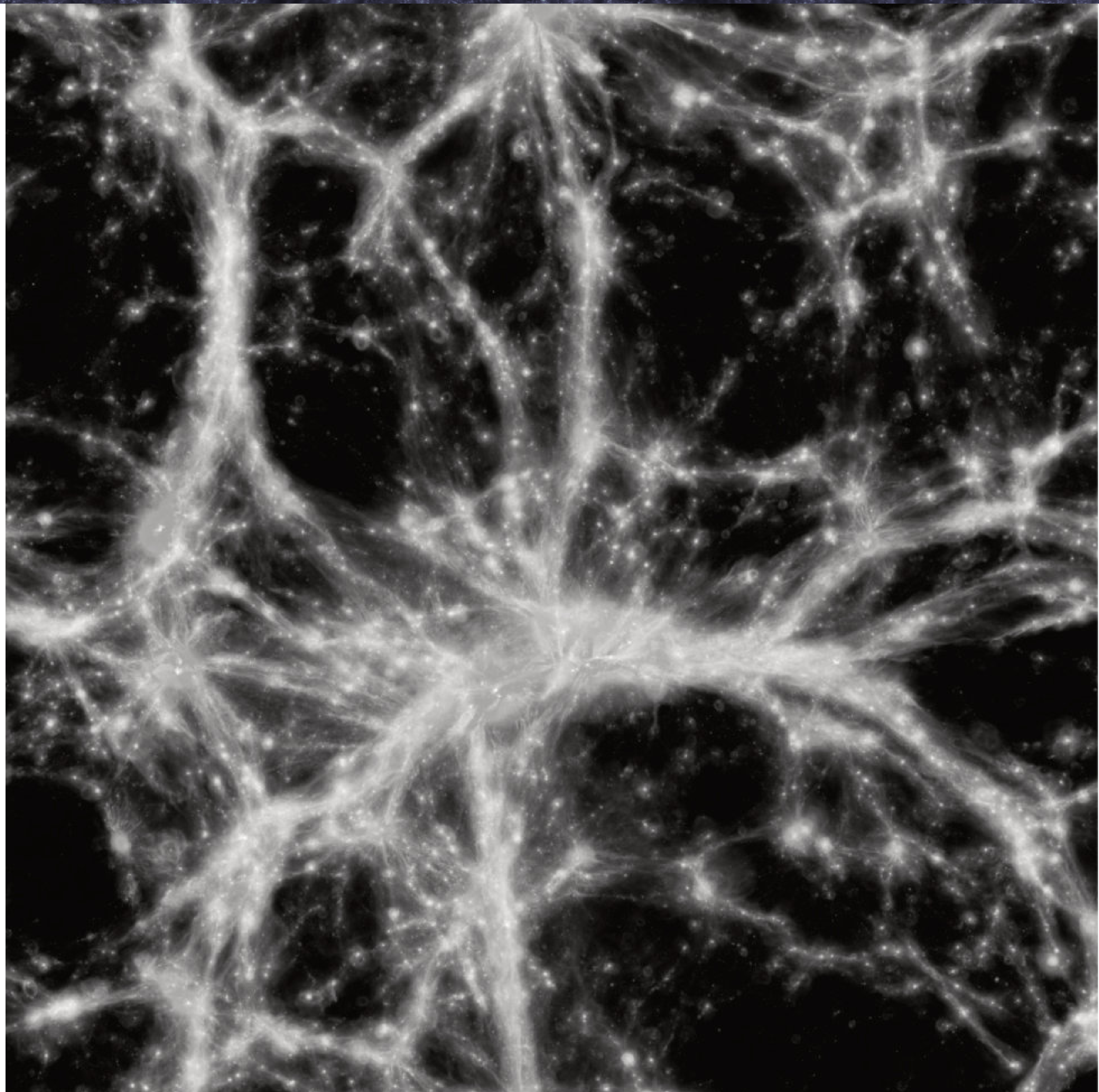
- SWIFT is a hybrid (MPI + threads) code
- It implements task-based parallelism using the QuickSched library
- This allows for better load-balancing on single nodes
- Uses the many-core nodes much more efficiently





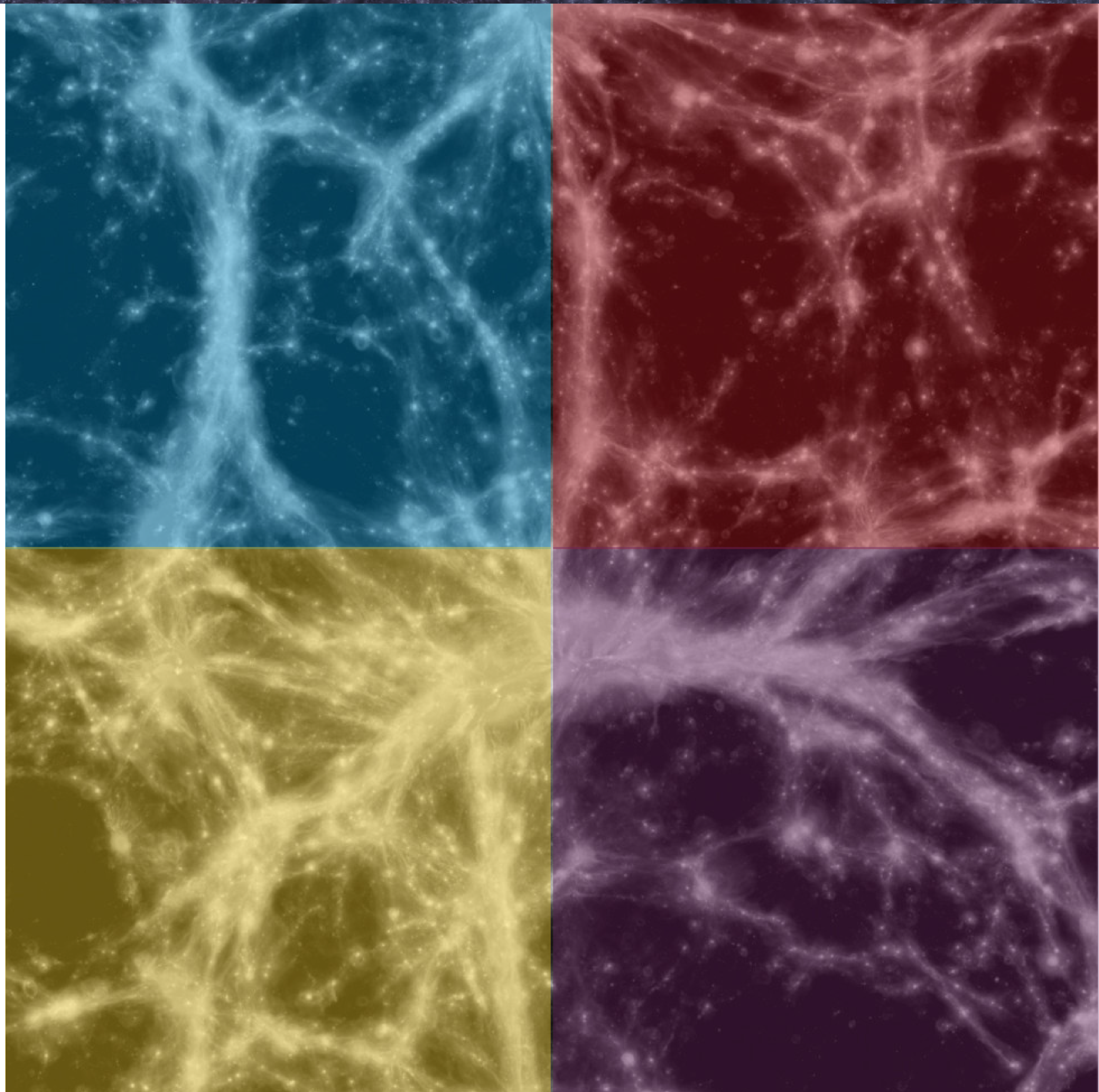
The problem of multi-node science

- How do I split a cosmological volume into N, where N ~ 100s, pieces?
- Two problems:
  - Communication Balance
  - Memory Balance



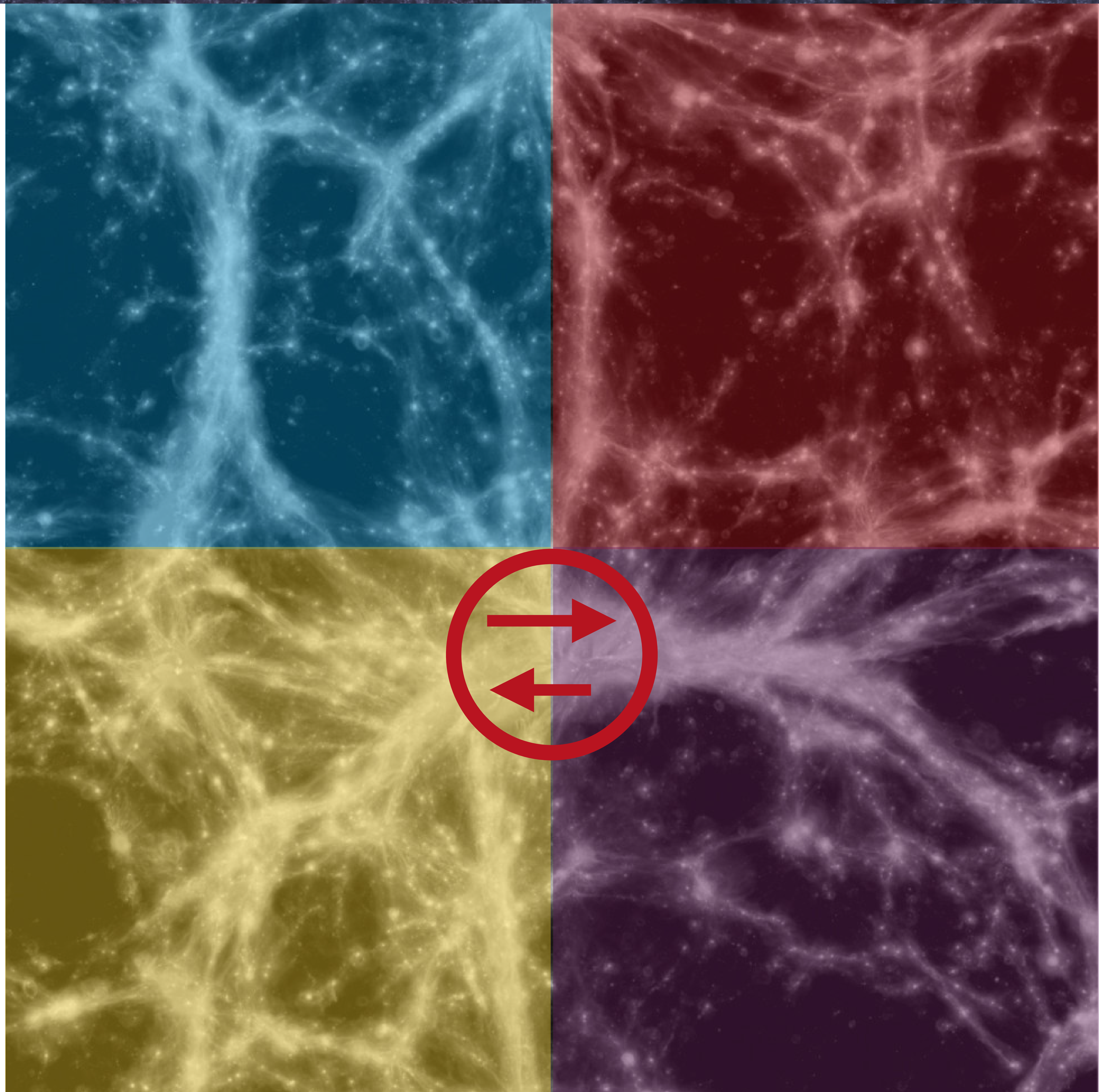
The problem of multi-node science

- How do I split a cosmological volume into N, where N ~ 100s, pieces?
- Two problems:
    - Communication Balance
    - Memory Balance



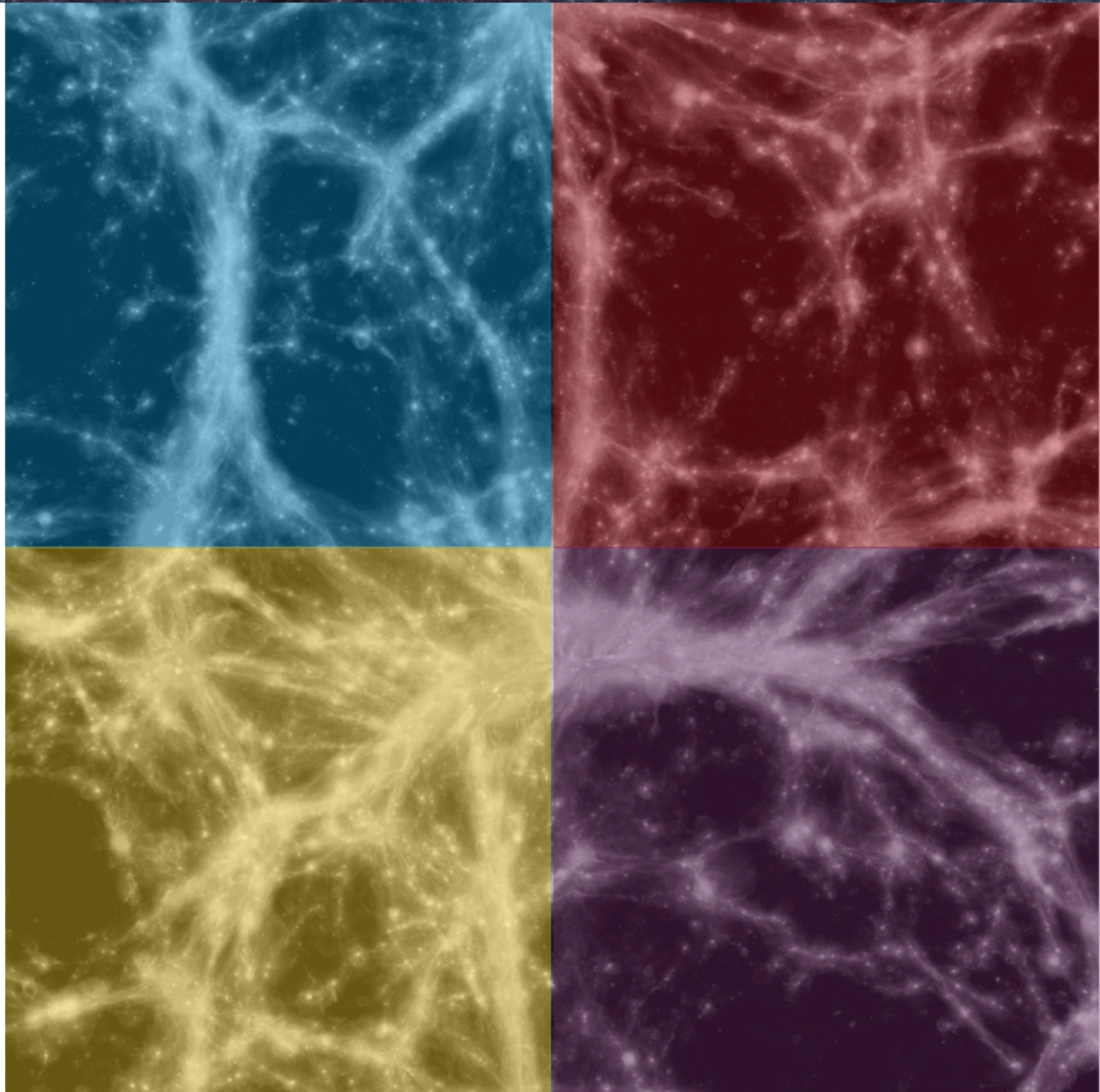
The problem of multi-node science

- How do I split a cosmological volume into N, where N ~ 100s, pieces?
- Two problems:
  - Communication Balance
  - Memory Balance



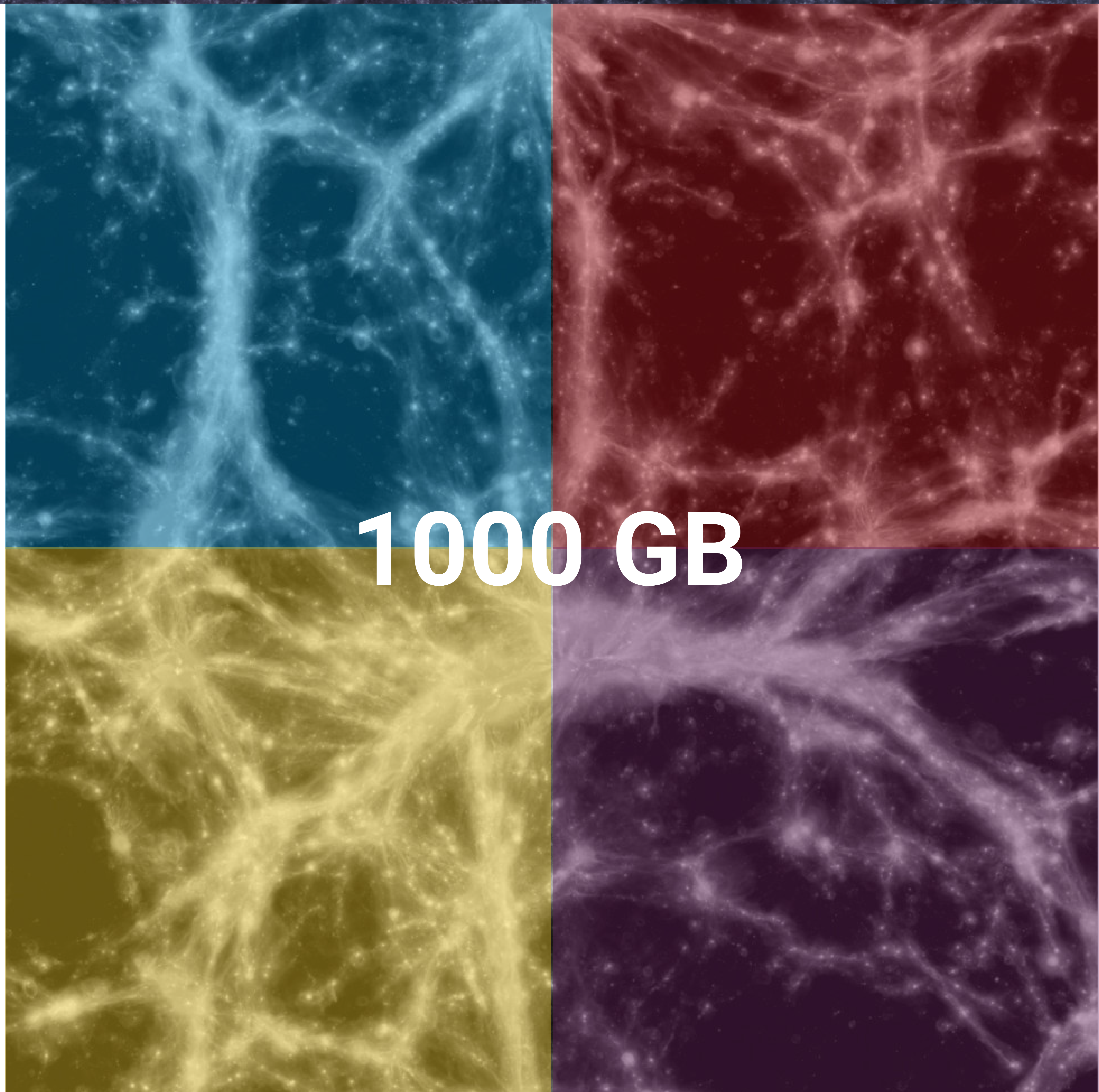
The problem of multi-node science

- How do I split a cosmological volume into N, where N ~ 100s, pieces?
- Two problems:
    - Communication Balance
    - Memory Balance



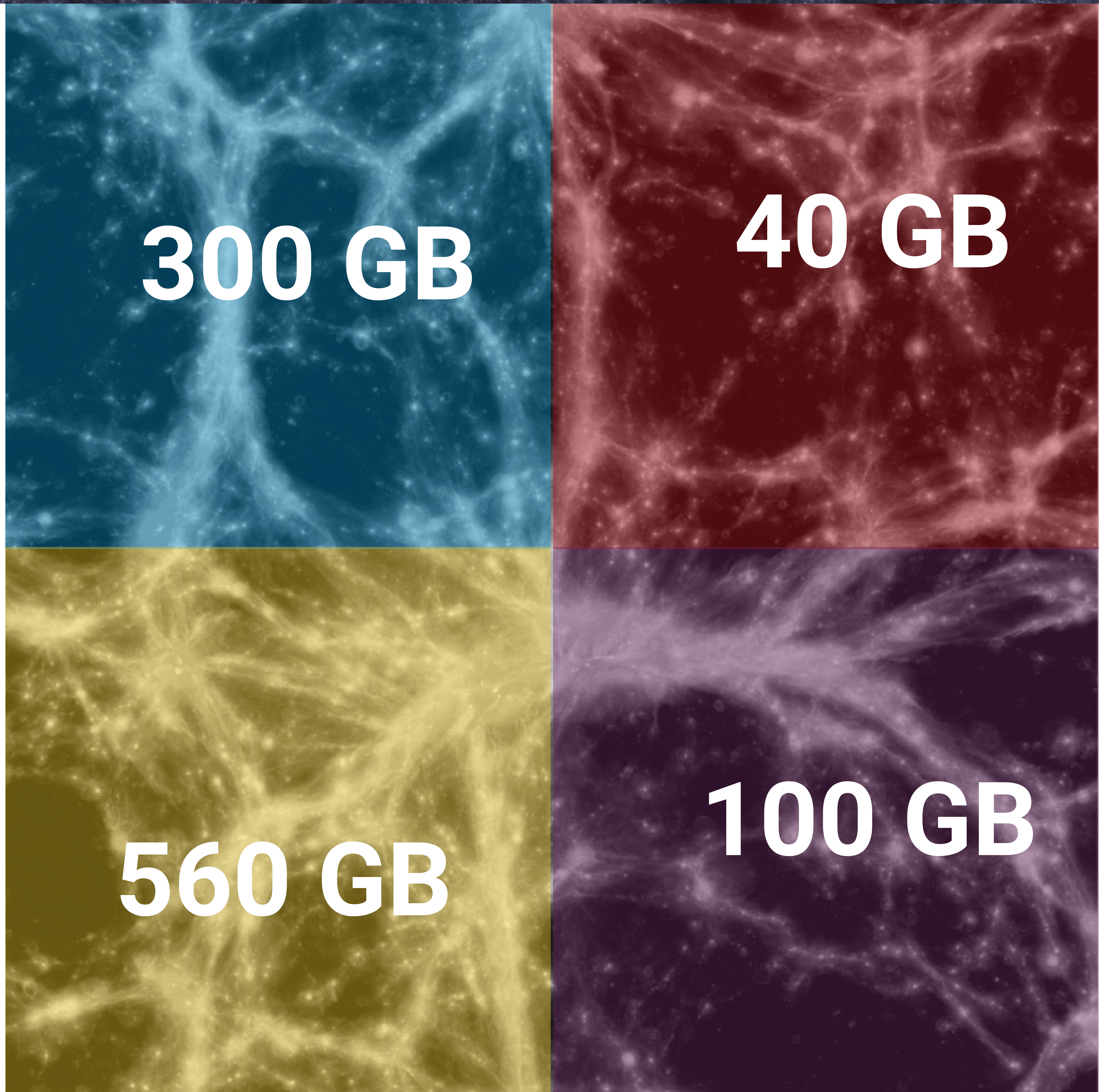
The problem of multi-node science

- How do I split a cosmological volume into N, where N ~ 100s, pieces?
- Two problems:
  - Communication Balance
  - Memory Balance



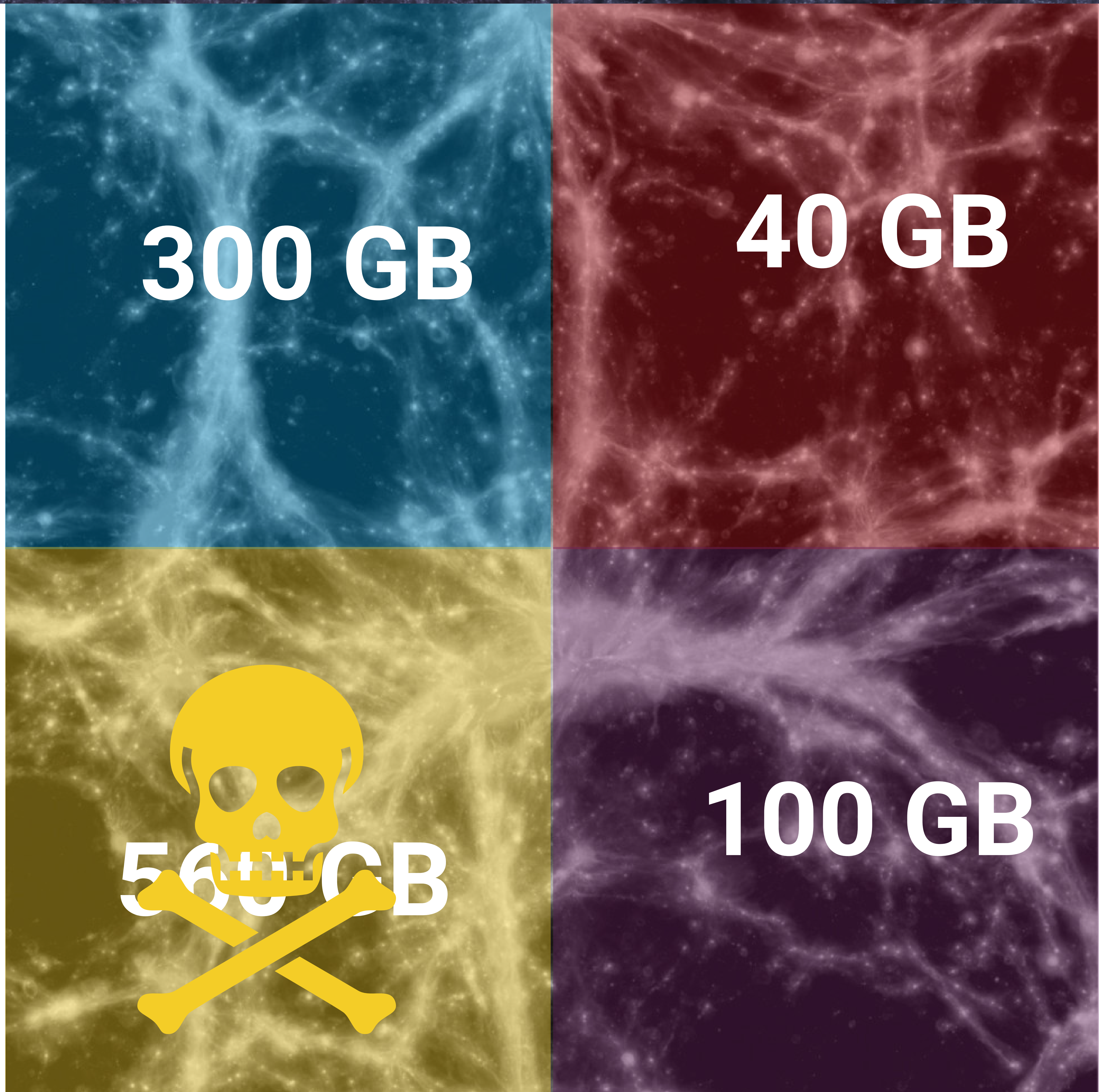
The problem of multi-node science

- How do I split a cosmological volume into N, where N ~ 100s, pieces?
- Two problems:
  - Communication Balance
  - Memory Balance



The problem of multi-node science

- How do I split a cosmological volume into N, where N ~ 100s, pieces?
- Two problems:
  - Communication Balance
  - Memory Balance



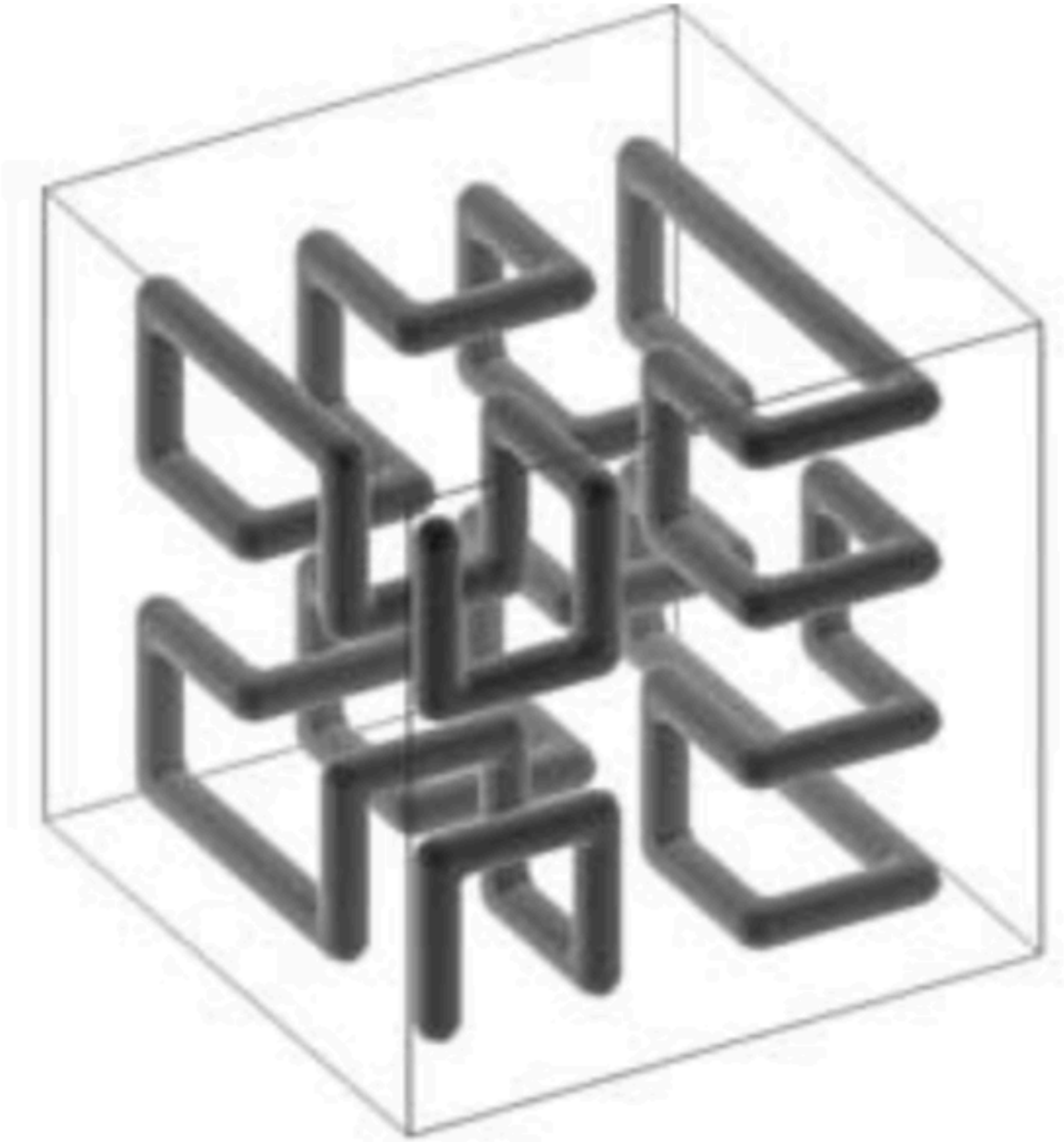
The problem of multi-node science

- Can fill the space with a "space-filling curve" and then chop it up so that each piece of string has the same number of particles

- Achieves perfect memory balance

- Reduces the median communication cost by minimising "surface area"

- Says nothing about the tail of communication!



Peano-Hilbert space-filling curve used by GADGET-2, Springel 2005

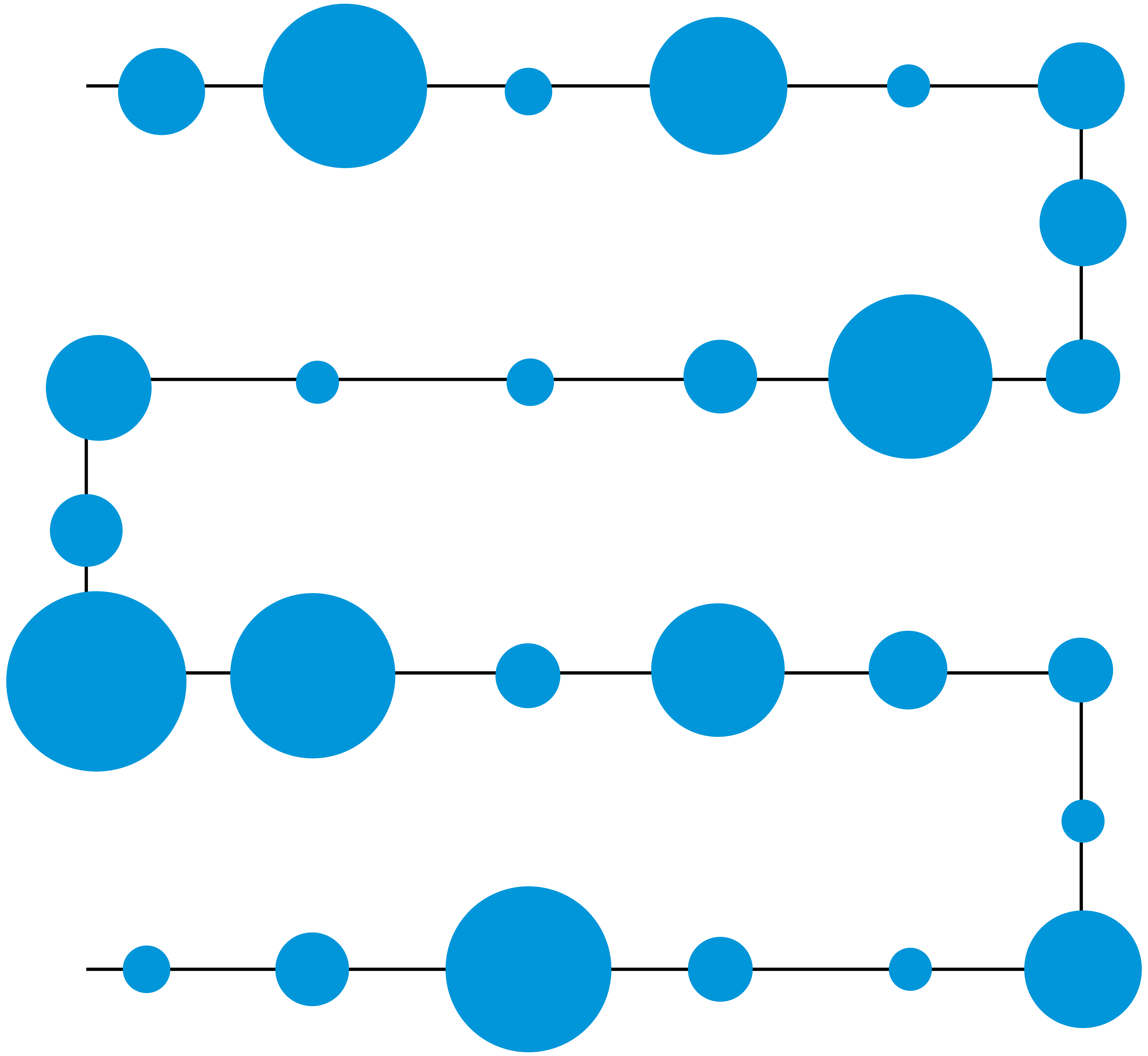
The problem of multi-node science

- Can fill the space with a "space-filling curve" and then chop it up so that each piece of string has the same number of particles

- Achieves perfect memory balance

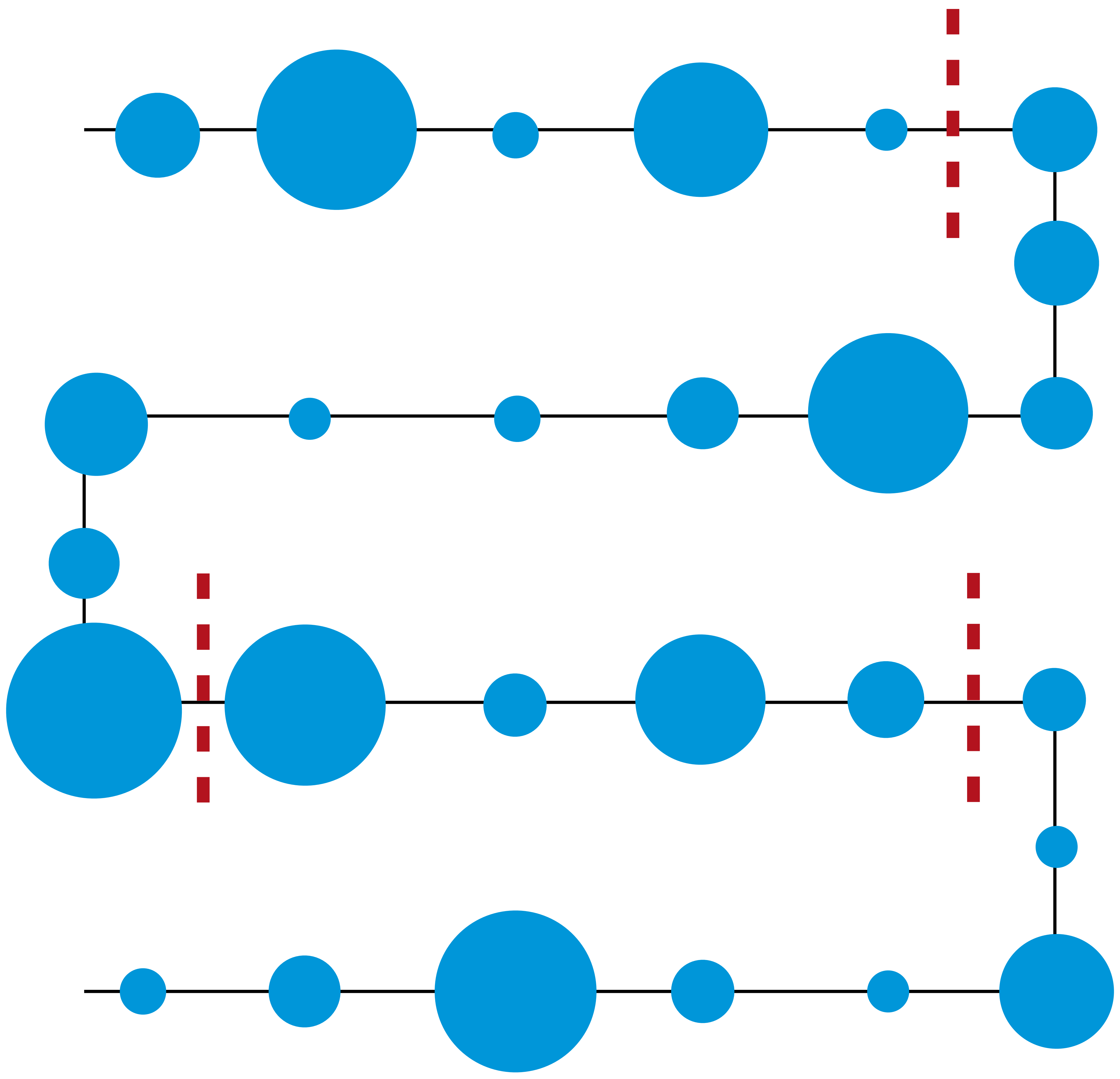
- Reduces the median communication cost by minimising "surface area"

- Says nothing about the tail of communication!



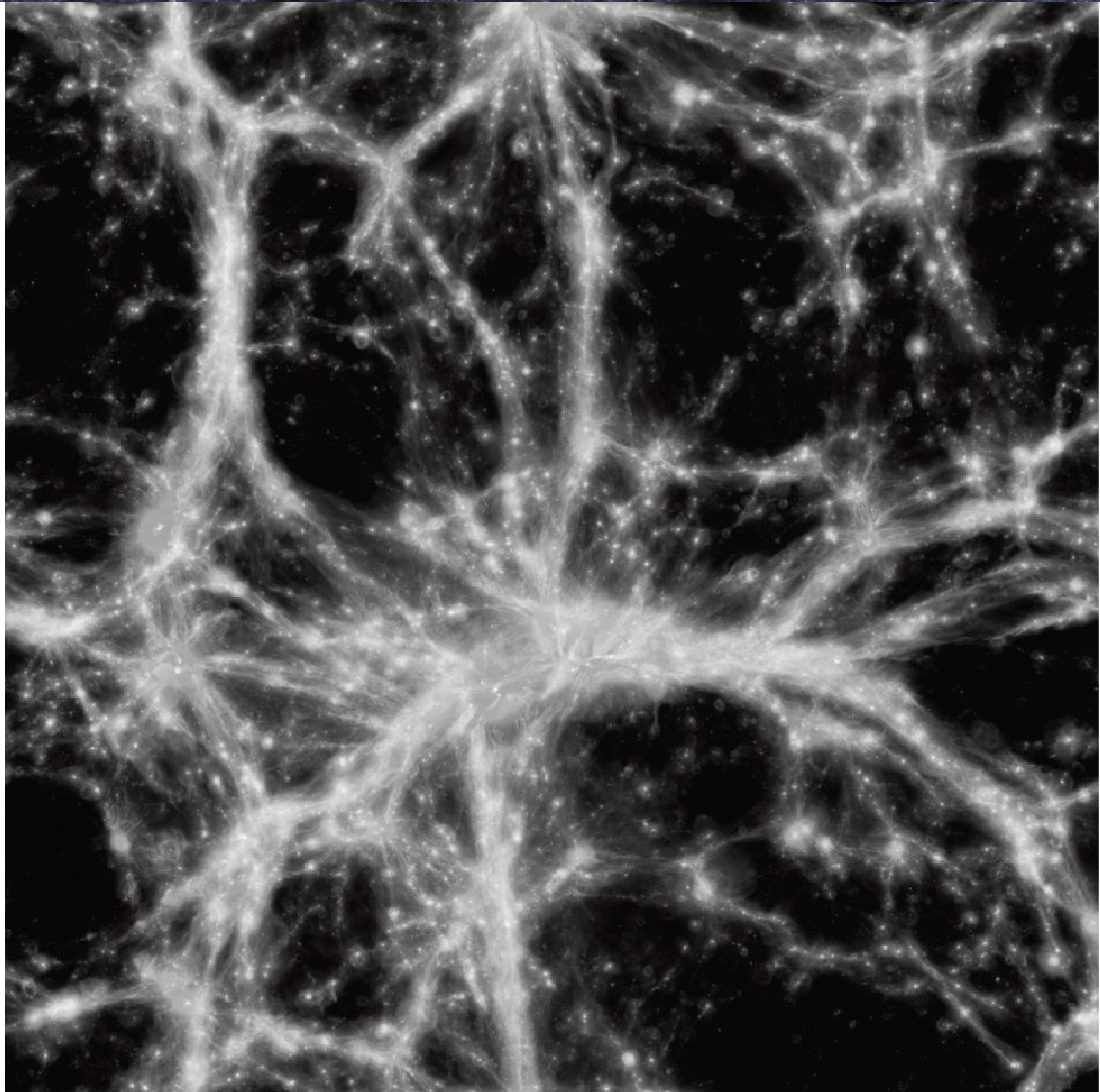
The problem of multi-node science

- Can fill the space with a "space-filling curve" and then chop it up so that each piece of string has the same number of particles
- Achieves perfect memory balance
- Reduces the median communication cost by minimising "surface area"
- Says nothing about the tail of communication!



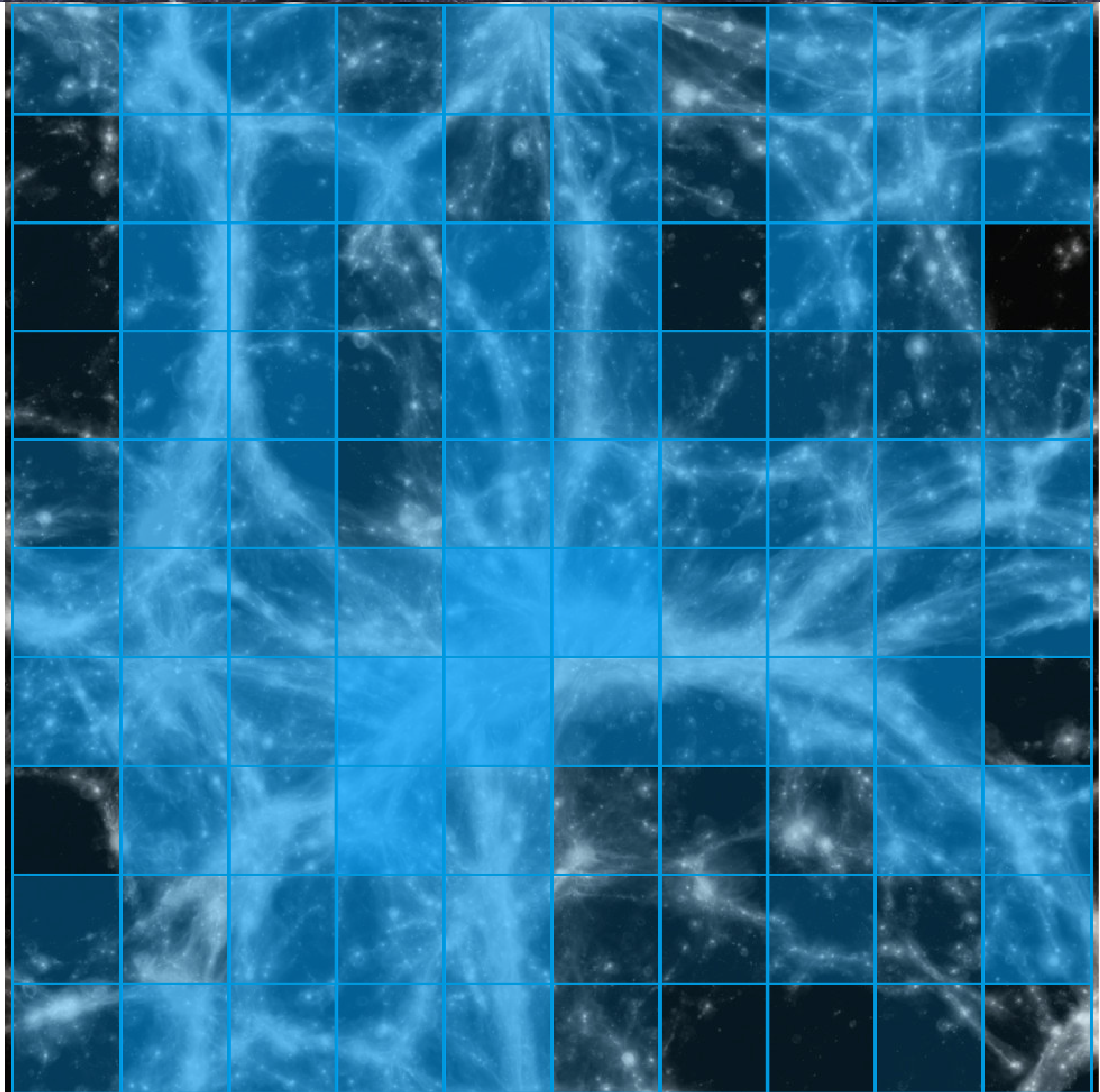
Can we do better?

- SWIFT uses a domain decomposition strategy that explicitly balances work
- Rough memory balance comes along with this
- Bisects the task graph, not the spatial distribution of particles!



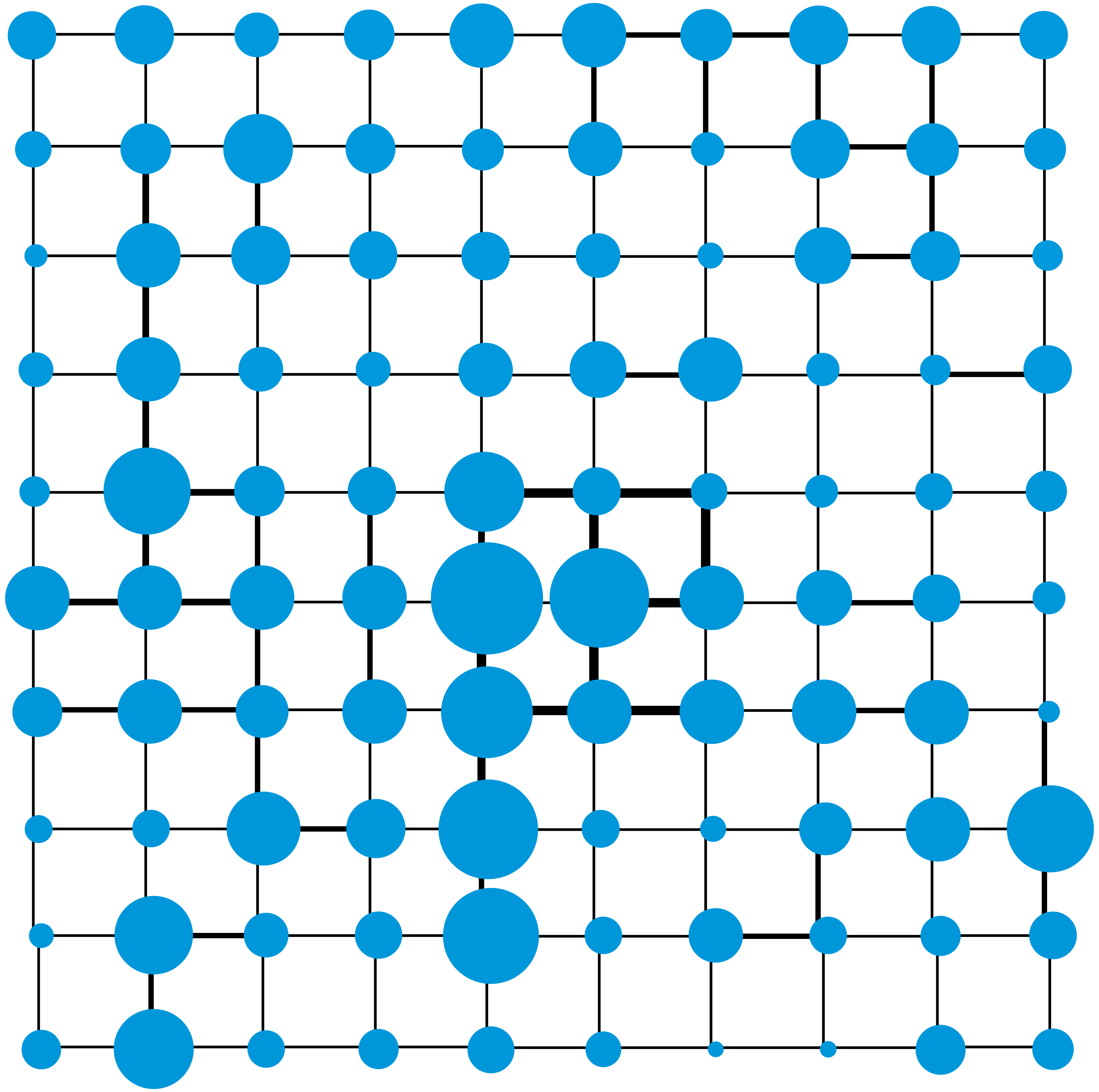
Can we do better?

- First, split the domain into a fixed number of top-level cells.
- On the right, we show the top-level cells coloured by the number of tasks that are associated with them.
- NB: This is just an illustration



Can we do better?

- Size of nodes represents number of tasks associated with cell
- Thickness of edge represents cost of communication
- Use (par)METIS to bisect this graph to:
  - Evenly split work
  - Minimise communication



Can we do better?

- SWIFT uses a domain decomposition strategy that explicitly balances work
- Rough memory balance comes along with this
- Bisects the task-weighted top-level cell graph, not the spatial distribution of particles!

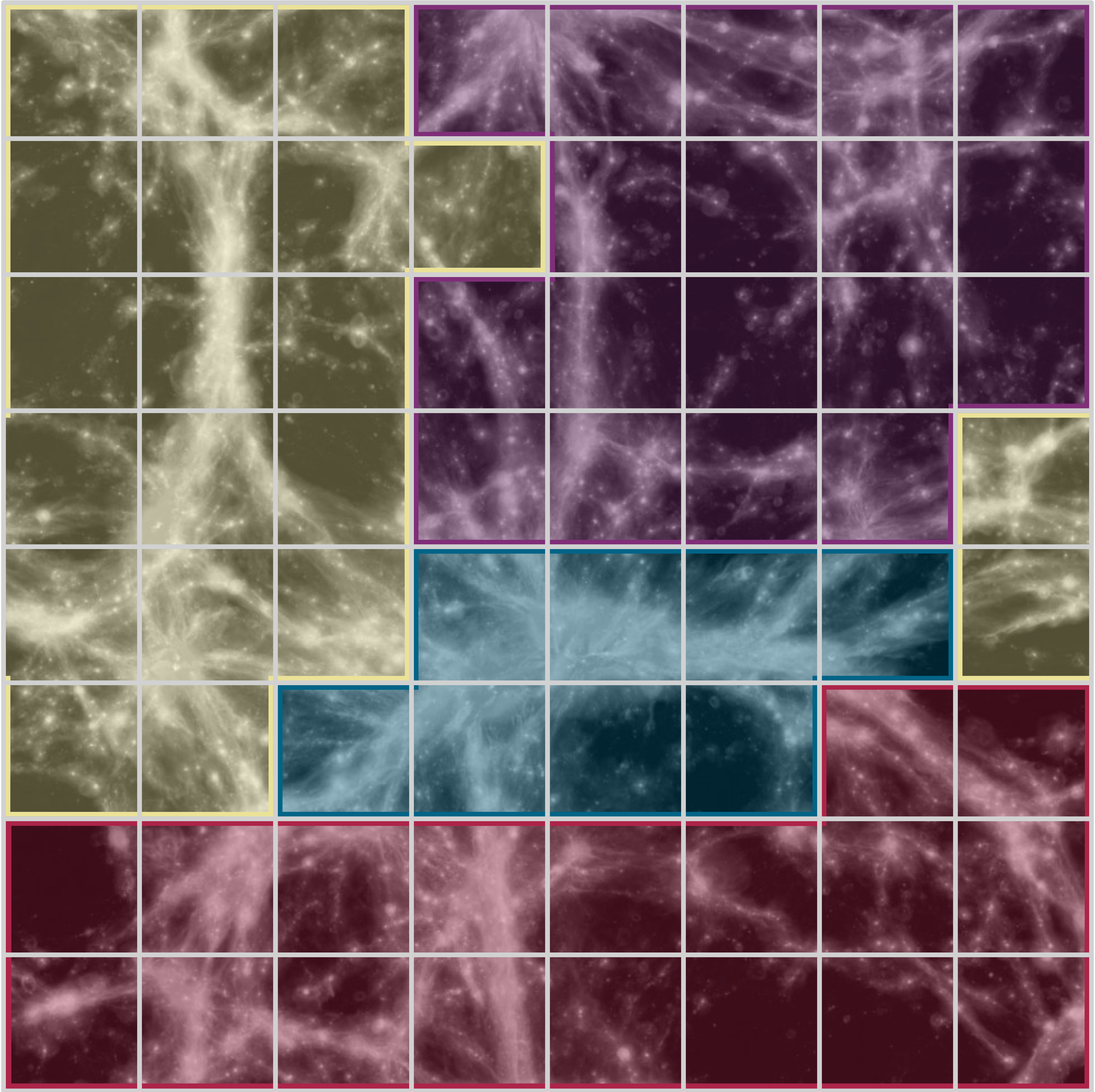
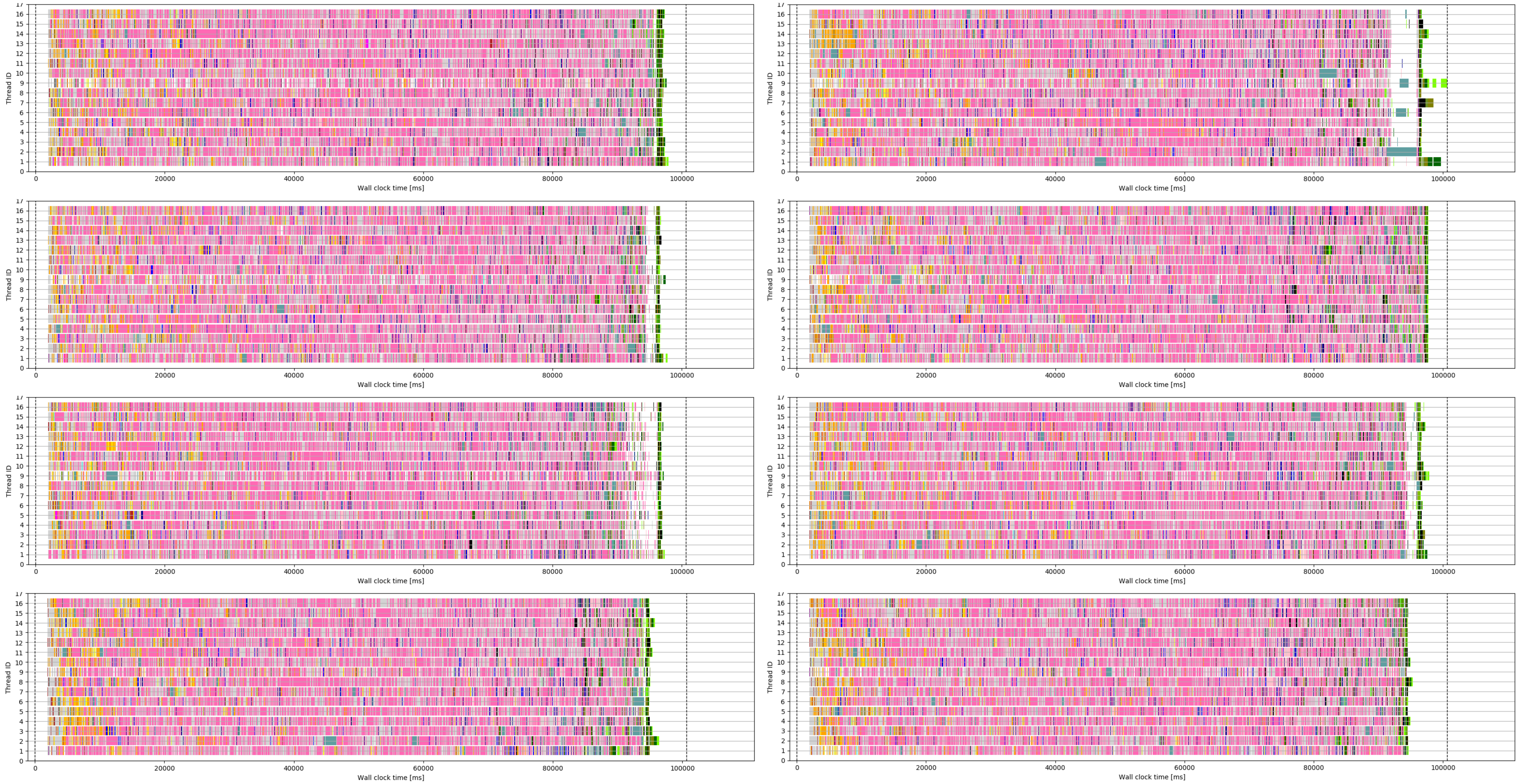


Illustration of top-level domain decomposition in SWIFT
Borrow+, SPHERIC 2018, 2018

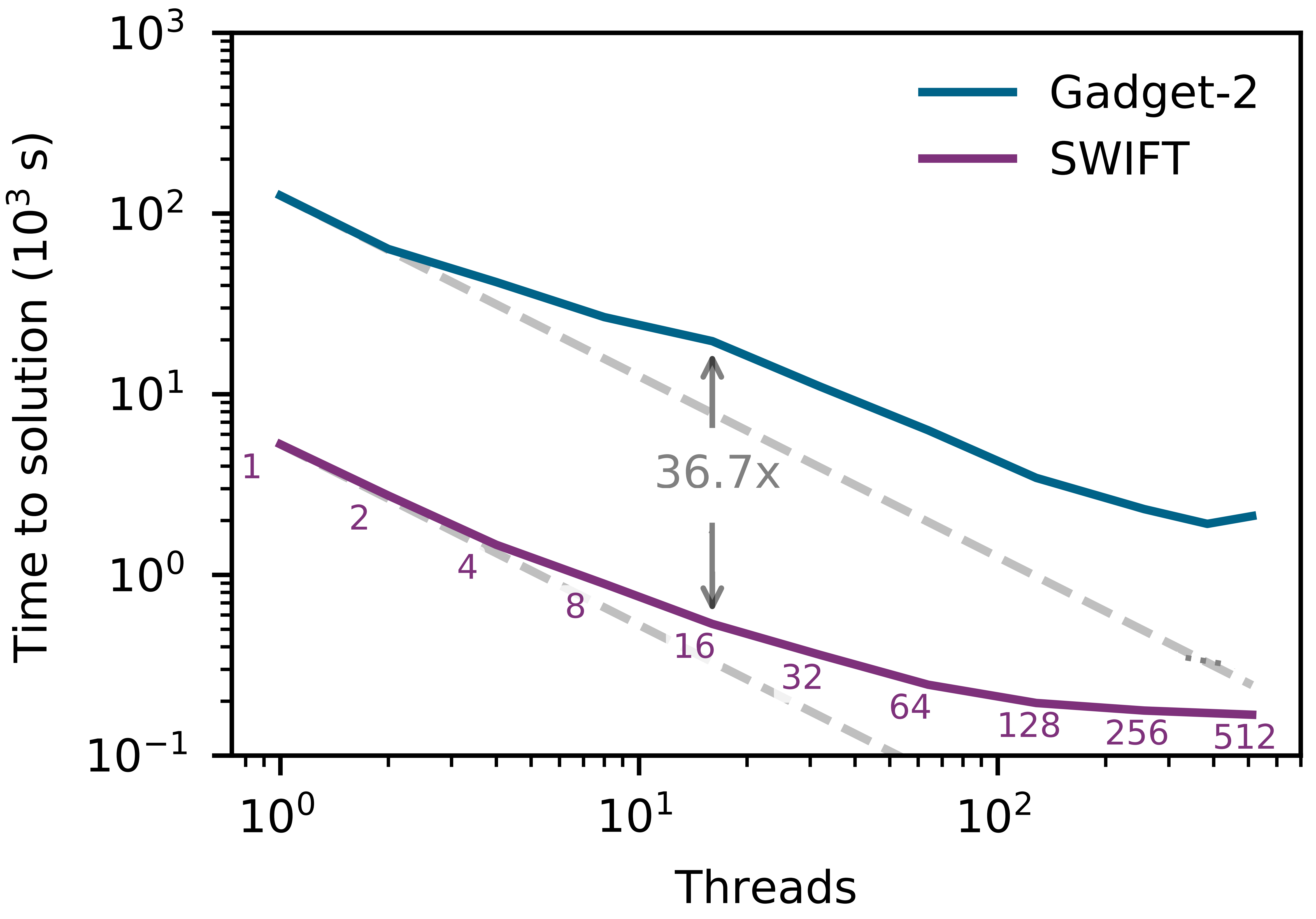
Communication-balanced Tasking



Results: Strong scaling

- Strong-scaling: for a given problem, throw more cores at it and see how much faster you go

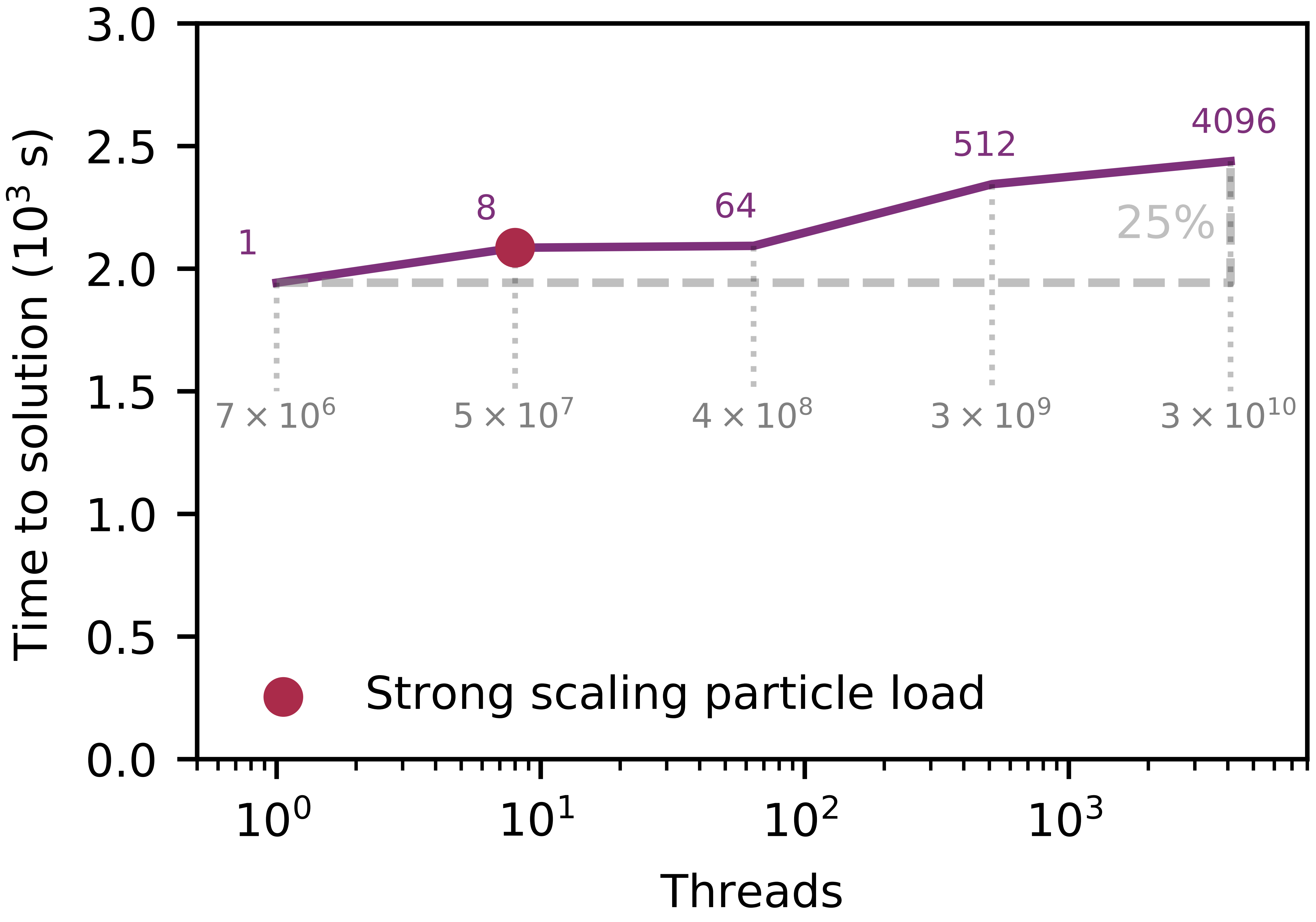
- SWIFT shows a 37x improvement over GADGET-2 (hydro-only), and about 20x with hydro+gravity.



SWIFT strong scaling on the EAGLE-25 z=0.1 box
Borrow+, SPHERIC 2018, 2018

Results: Weak scaling

- Weak-scaling: for a given *load*, copy it again and again and see how much speed you lose
- SWIFT shows *near perfect* weak scaling (this is extremely rare).
- Shown at the worst-case position: z=0.1 after all structure has formed



SWIFT weak scaling on the replicated EAGLE-25 z=0.1 box
Borrow+, SPHERIC 2018, 2018

Conclusions

- EAGLE-XL is only possible with SWIFT
- The EAGLE model is now fully open-source with it's implementation in SWIFT
- EAGLE-XL will open up a new era of statistics in computational cosmology
- SWIFT will allow us to explore other projects that were previously intractable
- Look out for the papers & open data (eventually...)