Simulations of Non-Equilibrium Flows

Alvaro Zamora, Elliott Slaughter

Moore’s Law and Science

Faster computation → higher fidelity physics

- Higher resolution
- More physical phenomena
  - Astrophysical fluids: MHD, chemistry/cooling, dozens of scalar fields, radiative transfer, viscosity, cosmic ray pressure

Navier-Stokes Equation

Probably most famous fluid equation, but not the most fundamental one.

Excerpt from Millenium Problem Statement

EXISTENCE AND SMOOTHNESS OF THE NAVIER–STOKES EQUATION

CHARLES L. FEFFERMAN

The Euler and Navier–Stokes equations describe the motion of a fluid in $\mathbb{R}^n$ ($n = 2$ or 3). These equations are to be solved for an unknown velocity vector $u(x,t) = (u_i(x,t))_{1\le i\le n} \in \mathbb{R}^n$ and pressure $p(x,t) \in \mathbb{R}$, defined for position $x \in \mathbb{R}^n$ and time $t \ge 0$. We restrict attention here to incompressible fluids filling all of $\mathbb{R}^n$. The *Navier–Stokes* equations are then given by

$$\frac{\partial}{\partial t}u_i + \sum_{j=1}^{n} u_j \frac{\partial u_i}{\partial x_j} = \nu \Delta u_i - \frac{\partial p}{\partial x_i} + f_i(x,t) \qquad (x \in \mathbb{R}^n, t \ge 0), \tag{1}$$

$$\operatorname{div} u = \sum_{i=1}^{n} \frac{\partial u_i}{\partial x_i} = 0 \qquad (x \in \mathbb{R}^n, t \ge 0) \tag{2}$$

with initial conditions

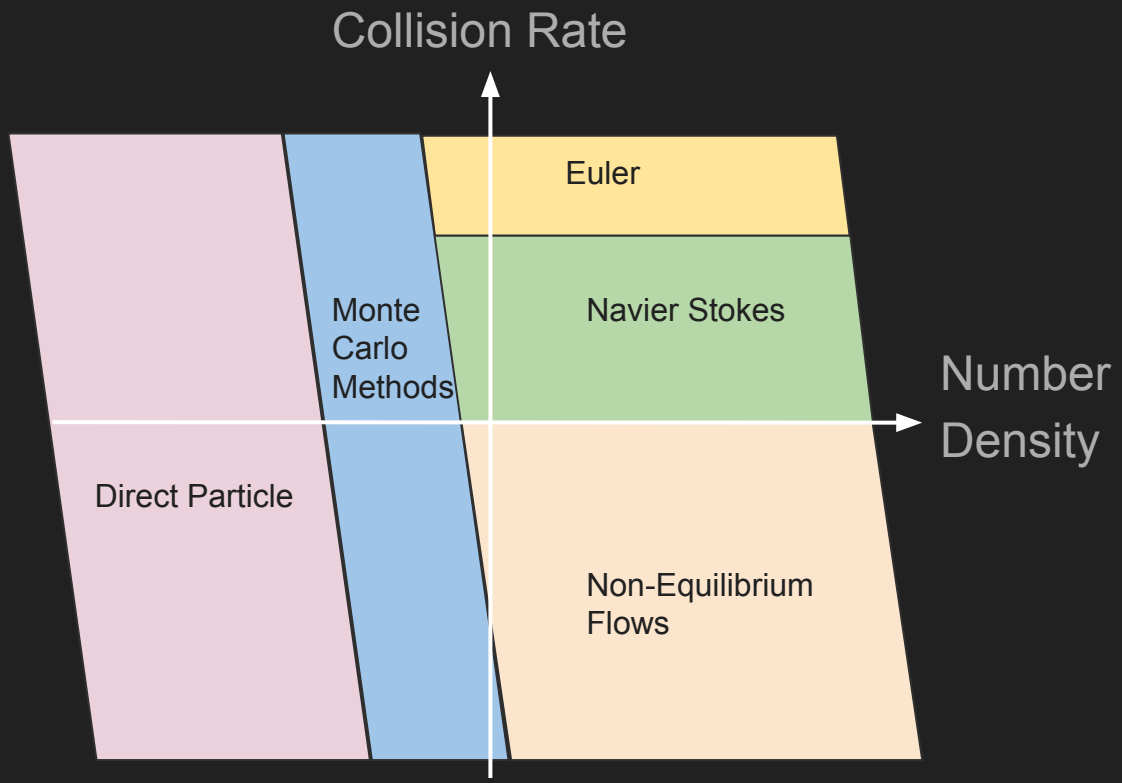
$$u(x,0) = u^\circ(x) \qquad (x \in \mathbb{R}^n). \tag{3}$$

Here, $u^\circ(x)$ is a given, C^∞ divergence-free vector field on $\mathbb{R}^n$, $f_i(x,t)$ are the components of a given, externally applied force (e.g. gravity), ν is a positive coefficient (the viscosity), and $\Delta = \sum_{i=1}^{n} \frac{\partial^2}{\partial x_i^2}$ is the Laplacian in the space variables. The *Euler equations* are equations (1), (2), (3) with ν set equal to zero.

Simulating Fluids

Fluids are many-particle systems

Having many particles is **usually** sufficient but not always sufficient to make use of NS/E approximation



Navier-Stokes/Eulerian Approximation

In the fluid approximation (large particle number), the fundamental quantity of interest is the phase space distribution.

$$f(t, x, y, z, v_x, v_y, v_z)$$

By making the at-equilibrium approximation, you replace the velocity distribution with a Gaussian. (Euler)

$$\frac{\rho}{(2\pi RT)^{D/2}} \exp\left(-\frac{(\boldsymbol{\xi}-\boldsymbol{u})^2}{2RT}\right)$$

Navier-Stokes is a first-order correction with a particular stress-tensor.

$$f = f^{(0)}\left(1 - \frac{1}{n}\left(\frac{2kT}{m}\right)^{1/2}\mathbf{A}\cdot\boldsymbol{\nabla}\ln(T) - \frac{2}{n}\mathbf{B}:\nabla\mathbf{c}_0\right)$$

Navier-Stokes/Eulerian Approximation

Eulerian Hydrodynamics:

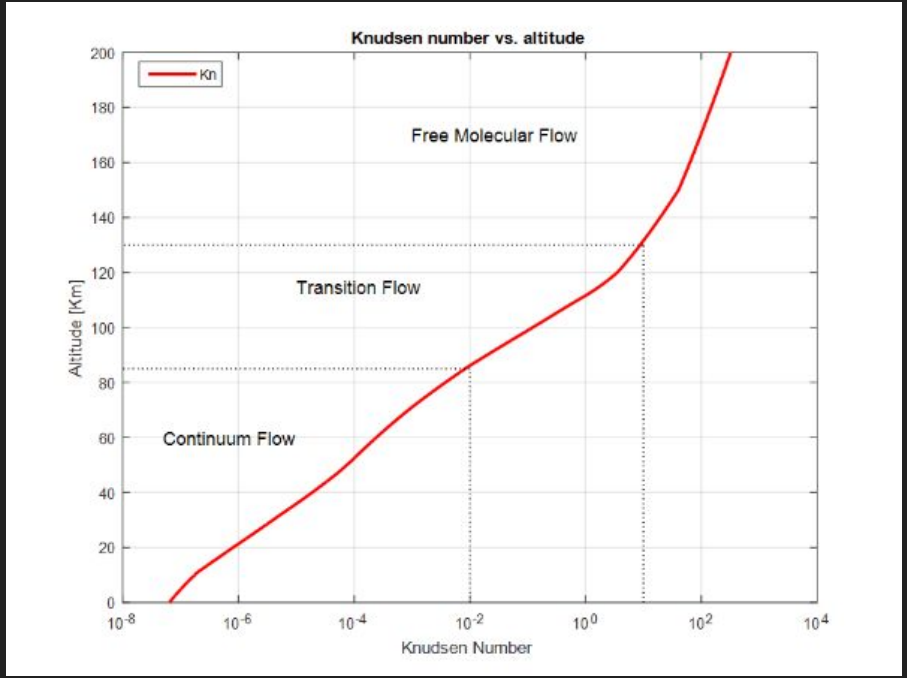
- Continuum, sheet, manifold
- **ZERO** mean free path
- No infinitesimal elements exchange mass, energy, momentum
    - Well-defined notion of a “parcel”
- Only thing relevant is how the continuum deforms due to non-uniform advection

Navier-Stokes:

- Non-zero mean free path --> “viscosity”
- Diffusion of momentum and energy
- Parcels exchange mass, energy, momentum

Validity of Approximation

For virtually all earthly and astrophysical gases, this is a great approximation that saves you from simulation three additional dimensions



Fadgyas et. al. 2018

A Comprehensive Toolbox

How do we go beyond Navier-Stokes?

- Burnett Equations & Super-Burnett Equations are an attempt
- Without uniqueness, you can’t avoid reverting back to 6D

There are kinetic schemes that do 6D, but there have been many challenges

- Timestep to take depends on collision rate
- Some physical parameters are fixed

CDUGKS: A Pivotal Moment in the KS Saga

- GKS
  - A Gas-Kinetic BGK Scheme for the Navier–Stokes Equations and Its Connection with Artificial Dissipation and Godunov Method, (Kun Xu 2001)
- UGKS
  - A unified gas-kinetic scheme for continuum and rarefied flows, (Kun Xu, Juan-Chen Huang 2010)
- DUGKS
  - Discrete unified gas kinetic scheme for all Knudsen number flows: Low-speed isothermal case (Zhaoli Guo, Kun Xu, and Ruijie Wang, 2013)
- CDUGKS
  - Coupled discrete unified gas kinetic scheme for the thermal compressible flows in all Knudsen number regimes (Hongtao Liu, Mingchi Kong, Qing Chen, Liang Zheng, Yong Cao 2018)

CDUGKS: A Pivotal Moment in the KS Saga

- Asymptotic-Preserving Scheme
  - The zero-th and first order approximations of the update rule are **EXACTLY** those of Eulerian and Navier-Stokes
  - In the zero collision regime, the update rule is exactly that of collisionless advection
- The update rules guarantee non-negative conserved variables
  - No matter how strong a shock, will never have nonphysical values.
- The timestep criterion is **independent of the collision rate**
  - In Navier-Stokes, one must take smaller timesteps at higher viscosity to resolve the faster diffusion.
  - Fixed cost across all collision rates
- Handles variable Prandtl number. Many previous kinetic schemes assumed Pr=1, but e.g. monatomic gases have Pr = ⅔.

State of KS Saga Codes

The papers in the KS Saga:

- Closed source codes
- Results presented are 1D or low res 2D.

MP-CDUGKS (this implementation) is one of the first if not the first open source massively parallel implementation of an asymptotic-preserving, non-negative preserving, kinetic scheme with fixed computational cost across ALL fluid regimes.

Why hasn’t this been done before?

Consider the following

1. 256^6 grid ≈ 1e14 elements
2. 1000 grid updates @ 100 floating point ops each

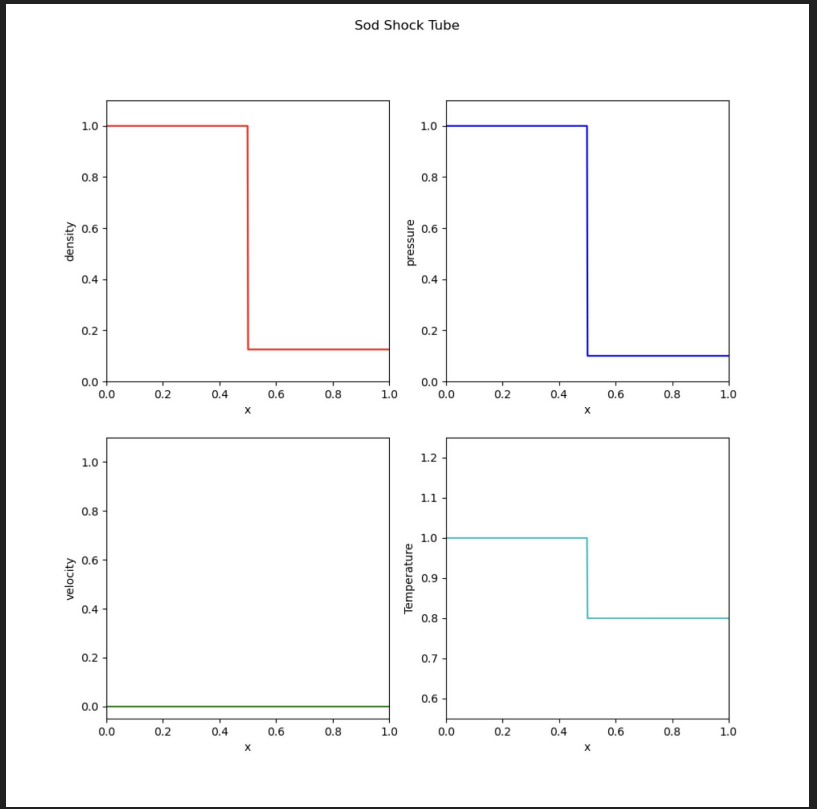
A total of 1e19 floating point operations.

Now imagine 512^6 or 1024^6 grid sizes, plus other physics.

Squarely in the realm of exascale+ computing. However, can study fluids in 1D/2D for now on commodity hardware.

Results

Sod Shock

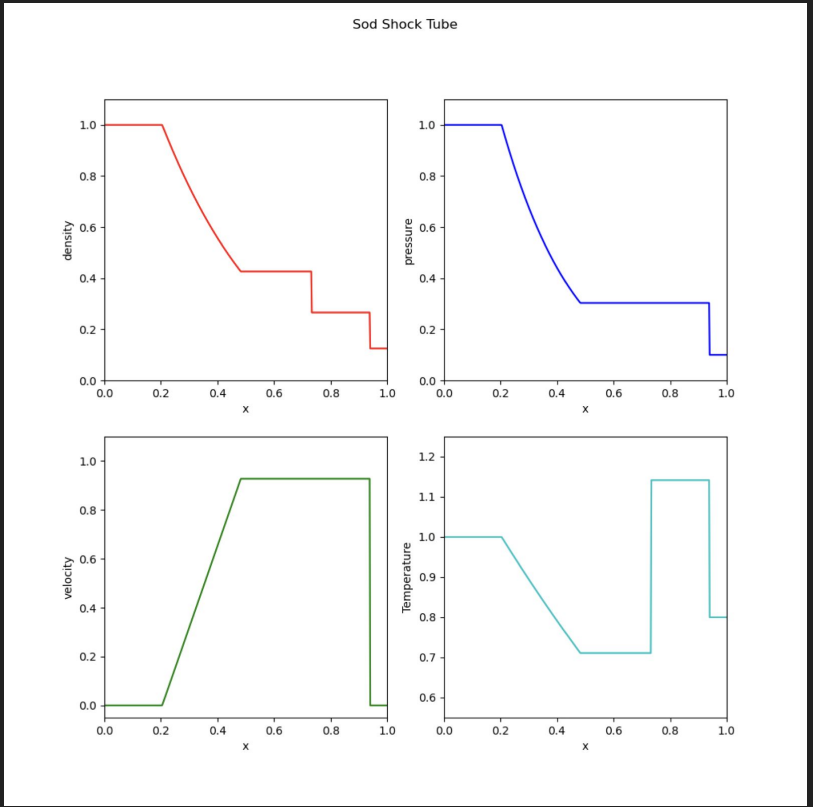

Sod Shock Tube
density
pressure
velocity
Temperature
x
x
x
x

Sod Shock

Analytic Eulerian

Need

- Correct shape of rarefaction fan
- Track contact discontinuity and shock



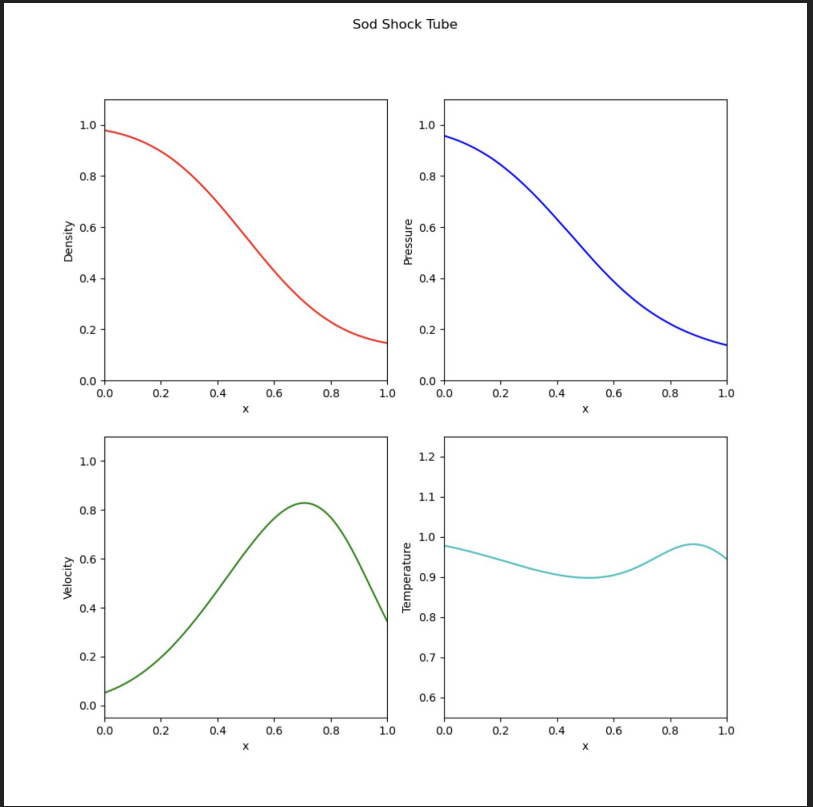
Sod Shock

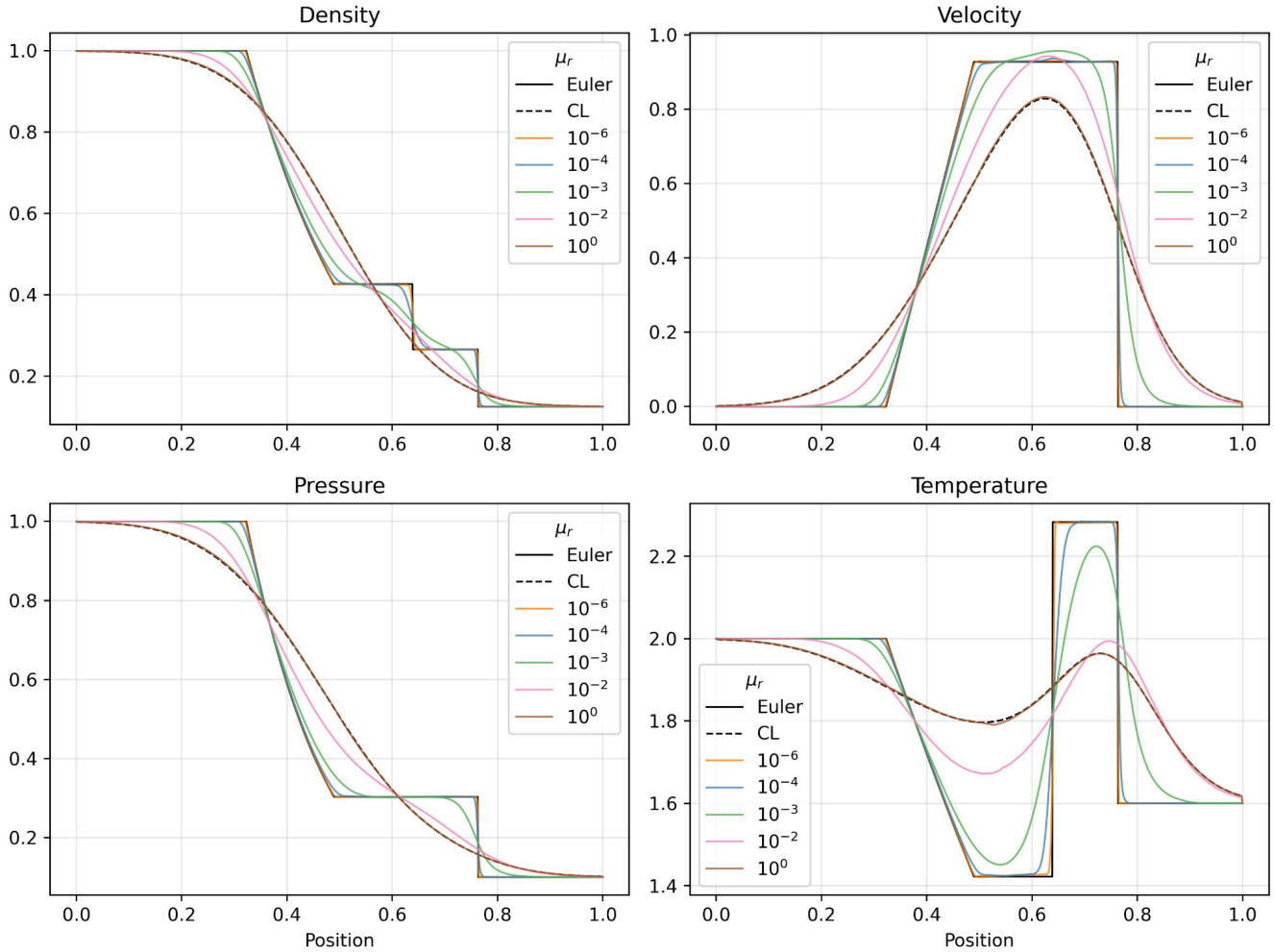
Need

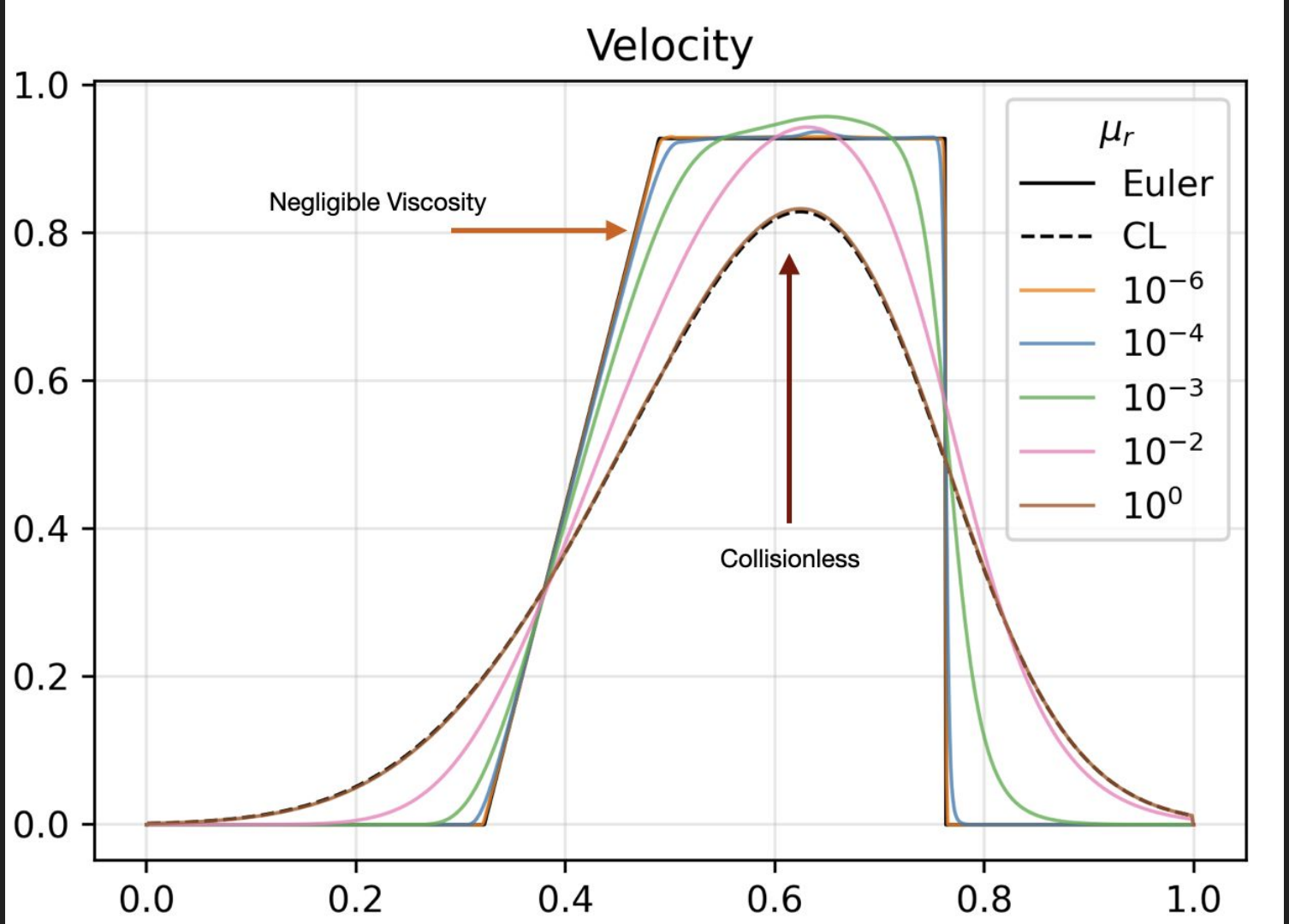
- Correct shape of rarefaction fan
- Track contact discontinuity and shock

In collisionless regime everything just diffuses

Analytic Collisionless

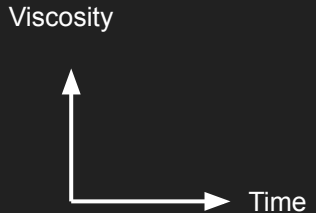



Sod Shock Tube Viscosity Test (t = 0.15)
Density
Velocity
Pressure
Temperature
μ_r
Euler
CL
10^{-6}
10^{-4}
10^{-3}
10^{-2}
10^{0}
Position
Position


Velocity
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Negligible Viscosity
Collisionless
μ_r
Euler
CL
10^{-6}
10^{-4}
10^{-3}
10^{-2}
10^{0}

Sine Wave Perturbation

Multimodal distributions are well beyond Navier-Stokes



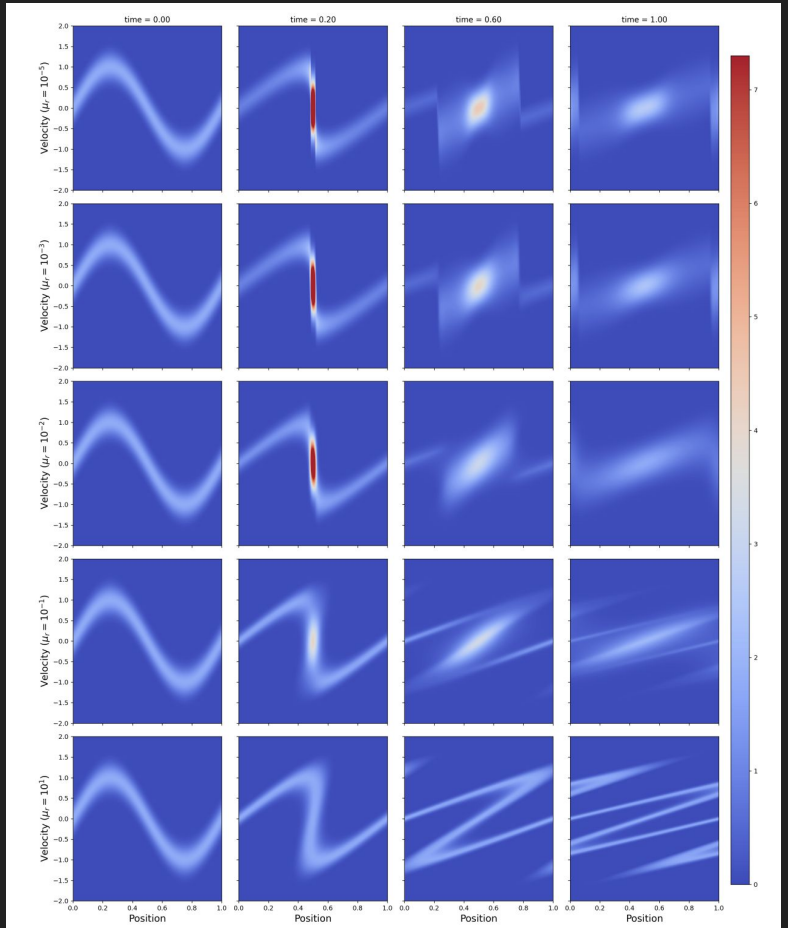
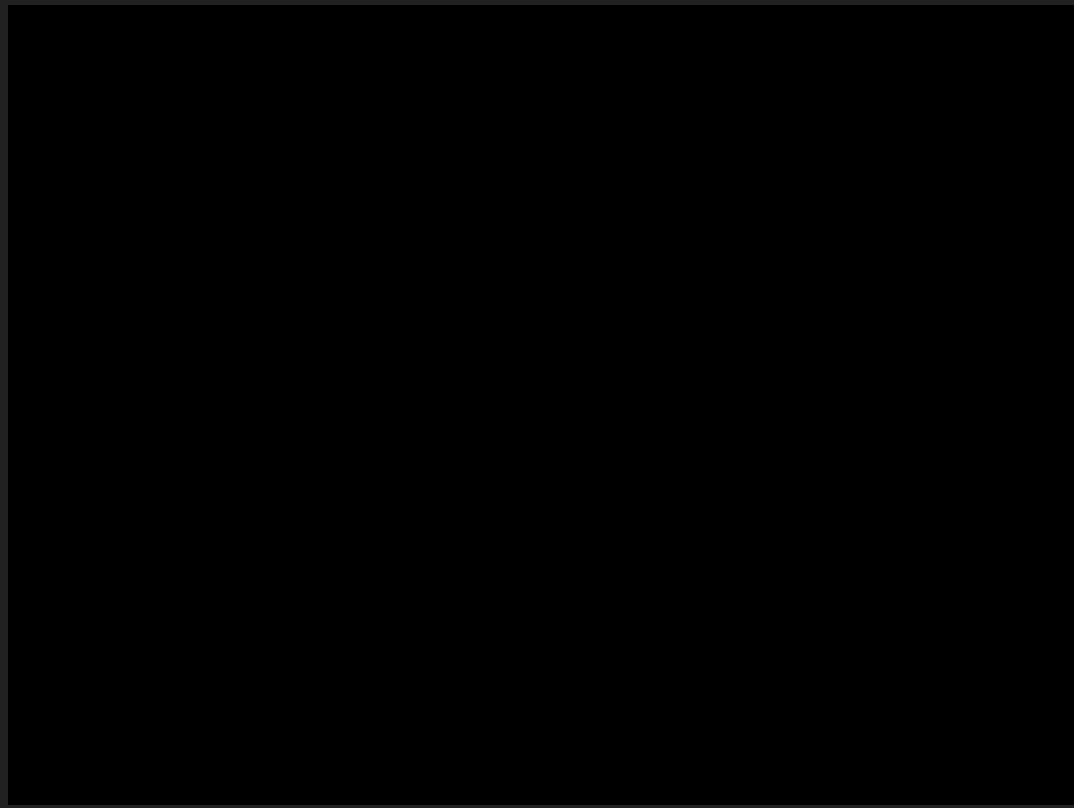


Figure 9. Phase space evolution of the sine wave perturbation at four different points in time. In order from left to right: initial conditions ($t = 0$), circa max density ($t = 0.2$), outflow (with shock for lower viscosity runs) ($t = 0.6$). The final column shows the shock interaction in the Eulerian/NS regime (upper rows), and shows streams overlapping in the free streaming regime (lower rows). To better display the lower densities, the colorbar range here is limited to half the maximum of that in the top row at $t = 0.2$.

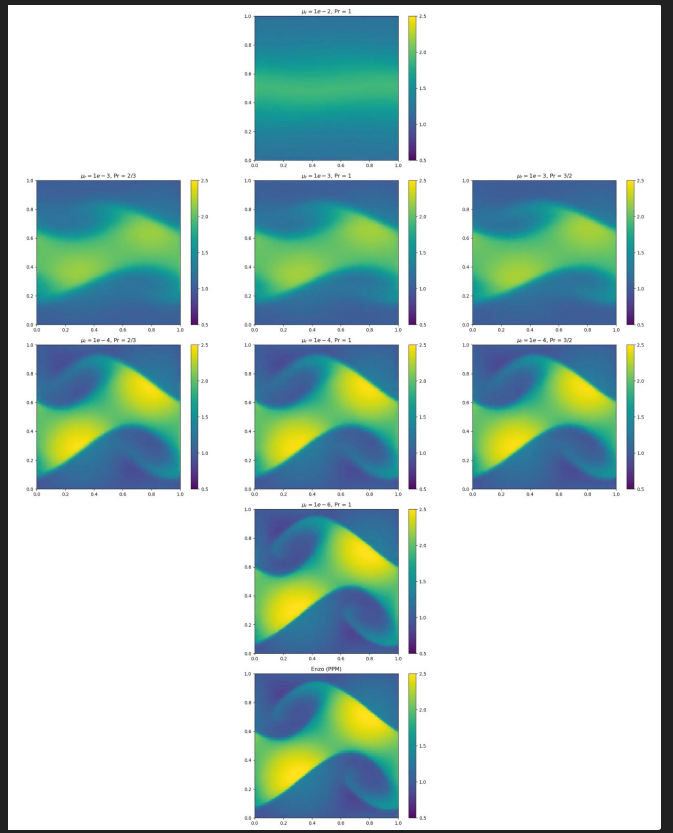
Kelvin-Helmholtz Instability



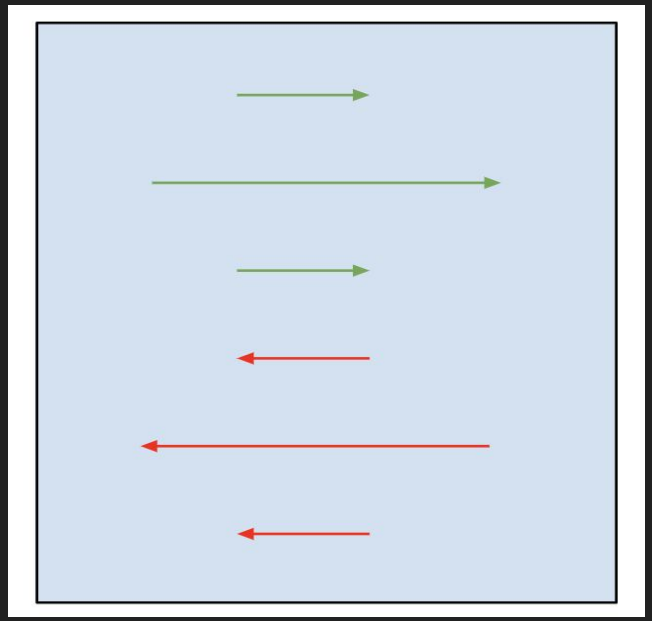
Kelvin-Helmholtz Instability

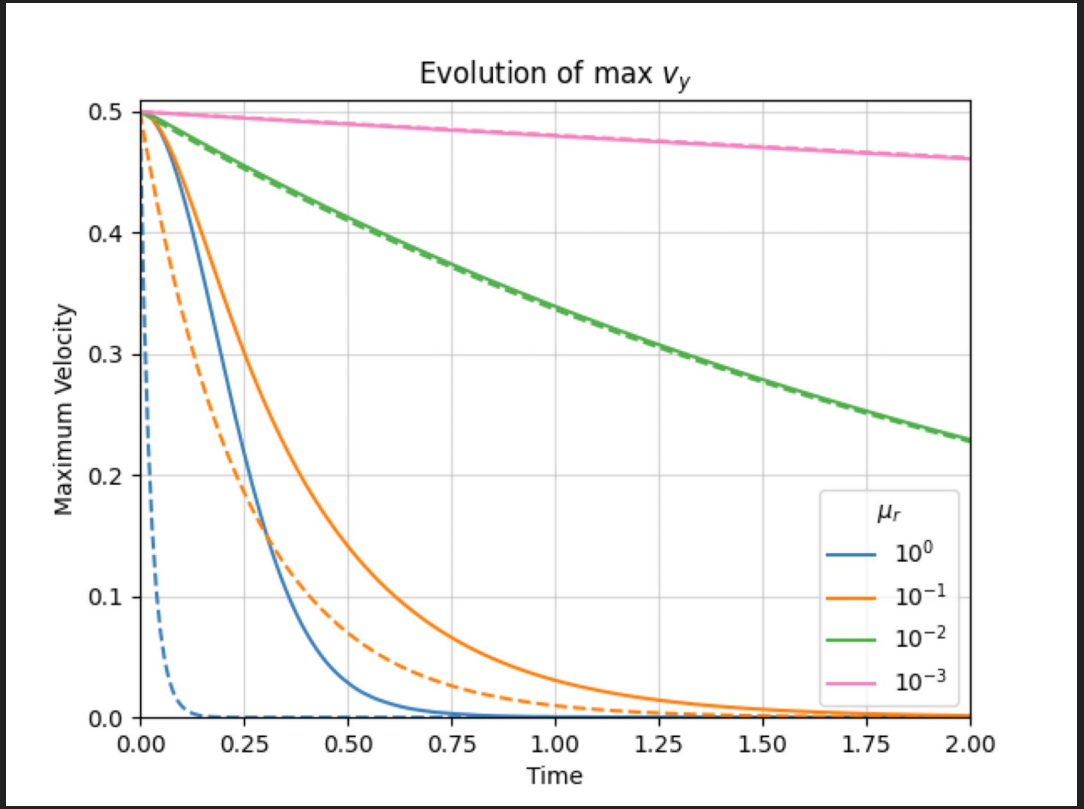
Prandtl Number

Viscosity



Shearing Box



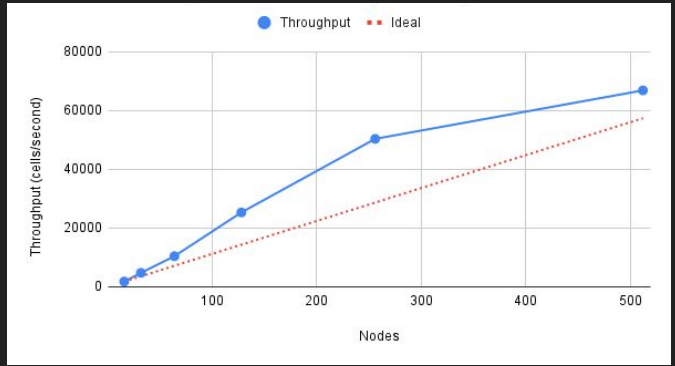


Performance Tuning

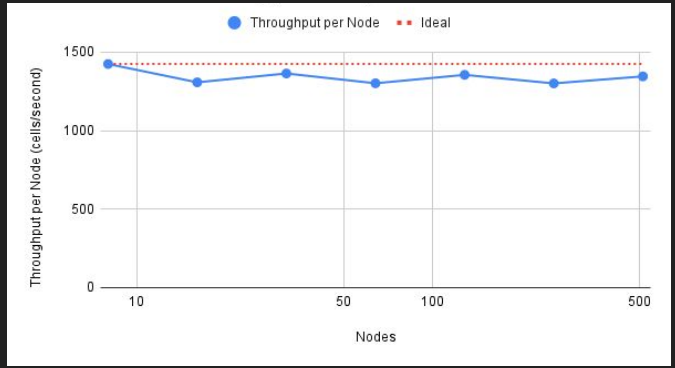
Performance Tuning Overview

- Starting point: code did not scale on 1 node
- Today: strong and weak scaling to 512 nodes
- Rest of this talk: how did we get here?

Strong Scaling

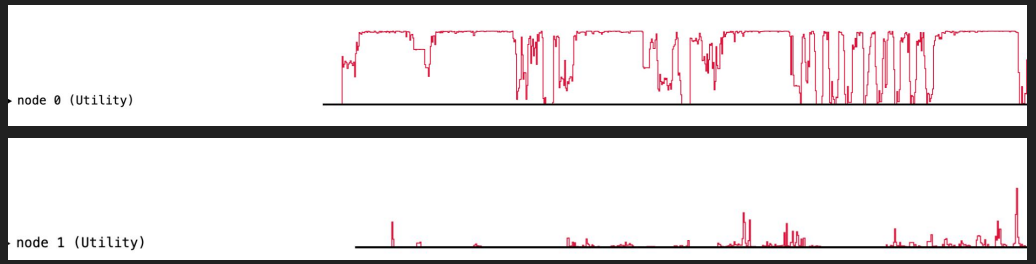


Weak Scaling



Tuning Part 1: First Profiles

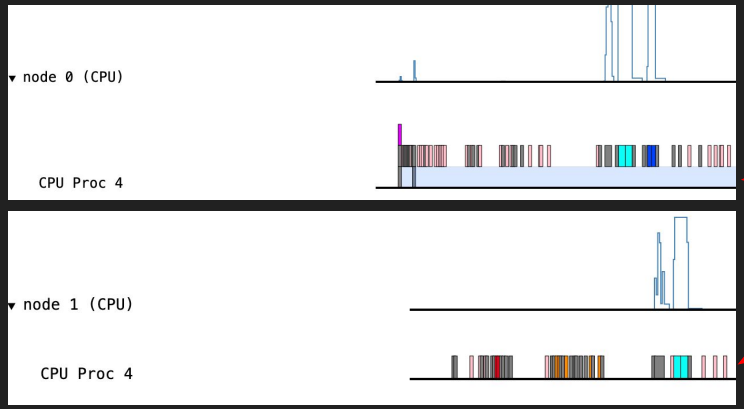
First order of business: what's the utility processor doing?



Node 0 is full

Node 1 is empty

Hypothesis: is control replication enabled?



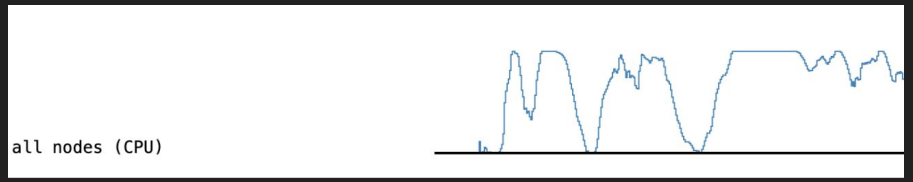
Nope!

Top-level task is only on node 0

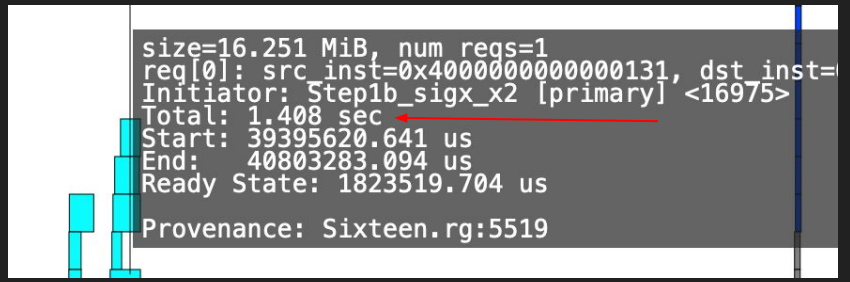
Solution: add
__demand(__replicable)
to top-level task

Tuning Part 2: Adventures in Core Binding

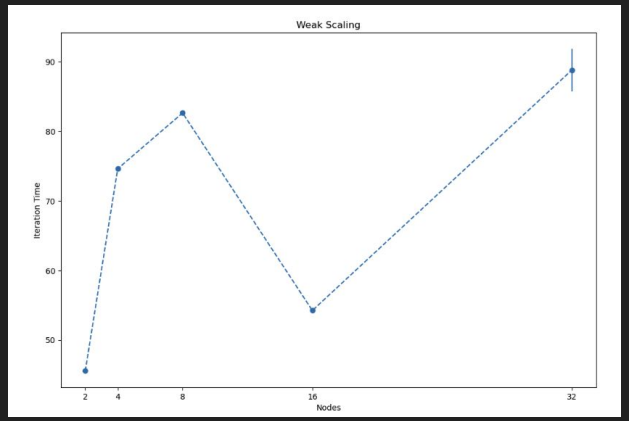
Better (but not perfect) CPU utilization now



1.4 second copies?!!! (for only 16 MiB)



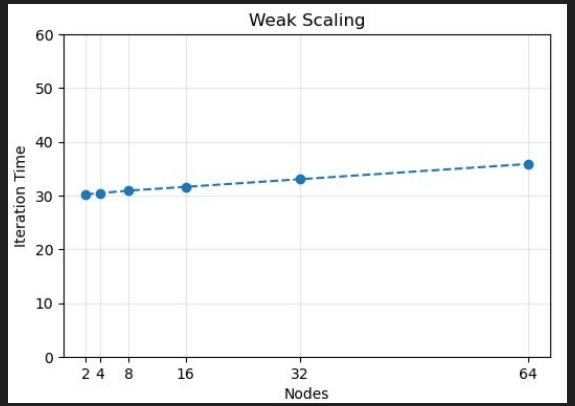
Weak scaling still all over the place



Hypothesis: core binding is off

Solution: bind to sockets

Weak scaling is dramatically better now (but still not perfect)



Tuning Part 3: New Machine, New Problems

Moved to Piz Daint for access to more nodes

Problem: now freezing (suspect OOM)

Solution (part 1): add
__demand(__inner)
to top-level task (to avoid inline mapping)

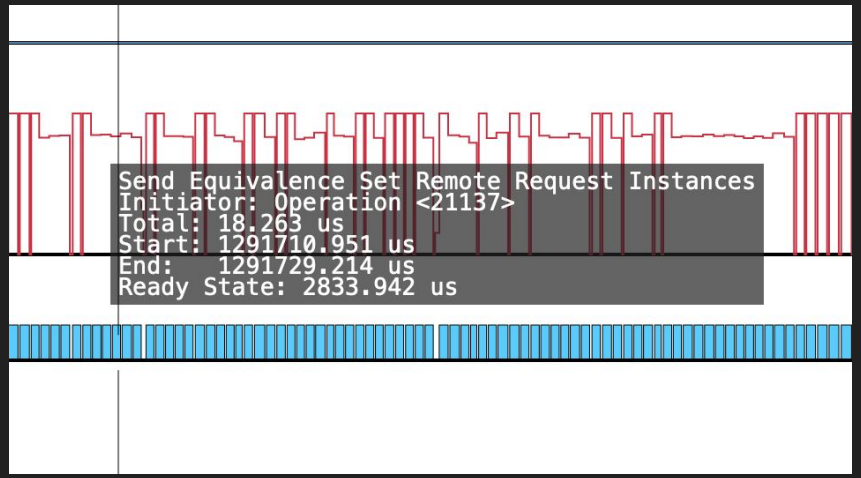
Problem: still OOM

Solution (part 2): destroy coloring data structures
used for partitioning

Code scales now

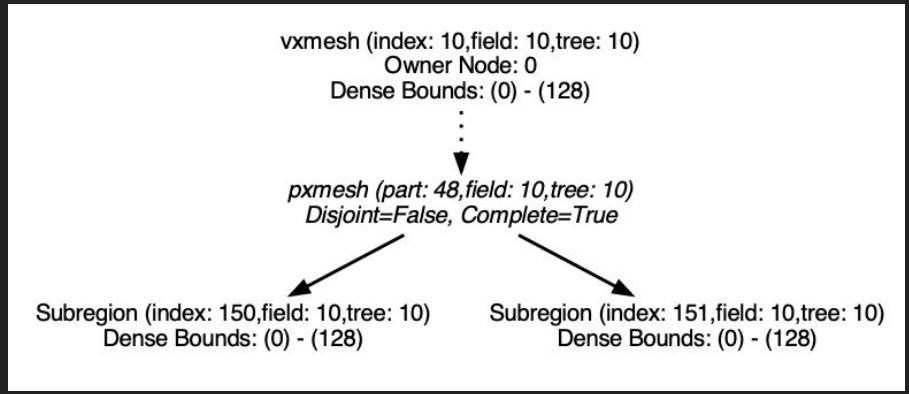
Tuning Part 4: Startup Time

Problem: startup time is growing, not going to get out to large enough node counts



Legion is moving a lot of metadata

Hypothesis: missing disjoint/complete partitions

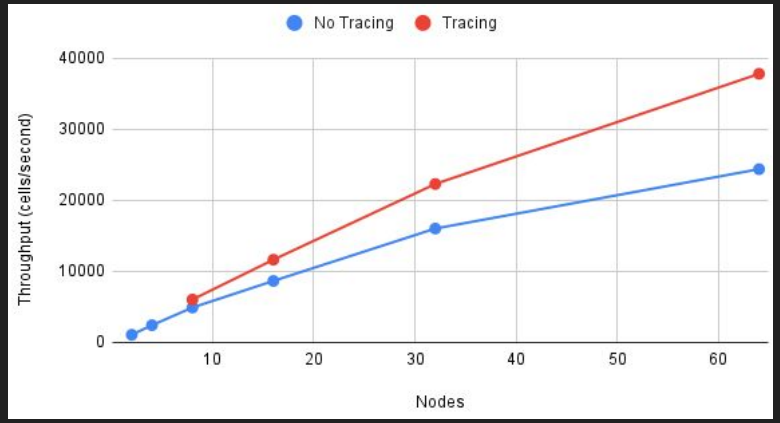


Solution: replicate this region (it's read-only, every copy identical)

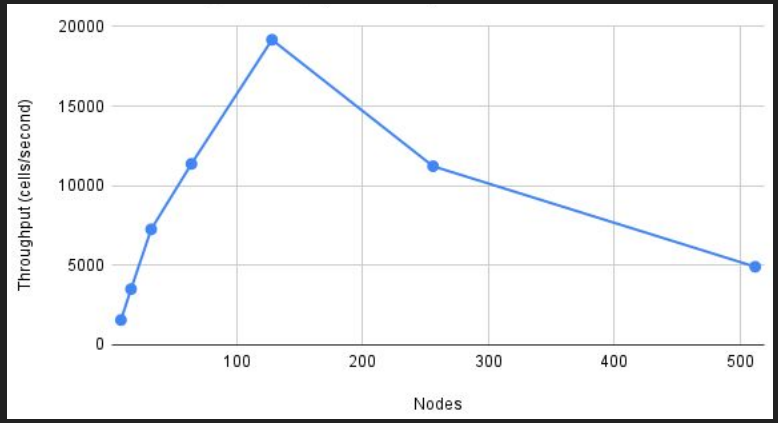
Tuning Part 5: Strong Scaling

Weak scaling performance was ok at this point, but strong scaling had room to improve

Hypothesis: need tracing to drive overheads low enough



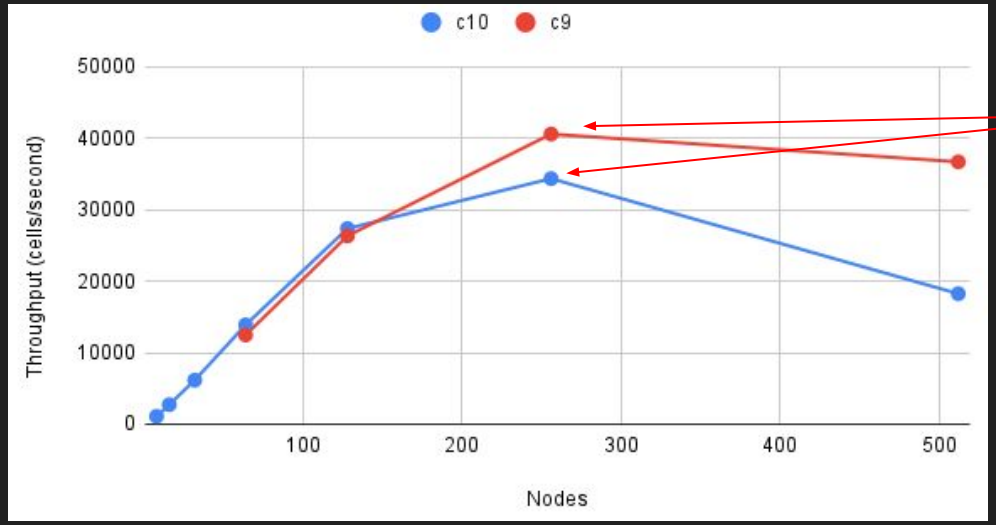
Sherlock (looks ok)



Piz Daint (ouch, why does it fall off so hard?)

Tuning Part 6: Core Binding, Again

Hypothesis: we're squeezing the runtime too hard and it needs more cores



Reducing the number of application cores actually improves scalability

(At a small cost to performance at lower node count)

Tuning: Summary

- Control replication
- Bind to sockets (if system has more than one per node)
- Inner task
- Destroy temporary data structures (colorings)
- Replicate region to enable disjoint/complete partition of read-only data
- Tracing
- Reduce application cores to balance usage with runtime
- Overall, most of this happened over a couple of weeks

Tuning: Things to Try Next

- GPUs: work, but need to debug performance
- Collective instances for replicated, read-only data
- Clean up tracing code to account for infrequent tasks (e.g., I/O)