Exascale simulations of binary mergers for LIGO

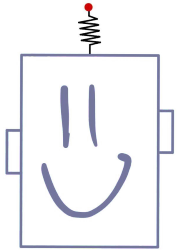
Kyle Nelli

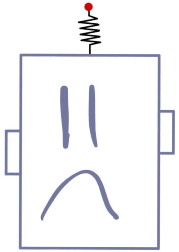
9/13/2023

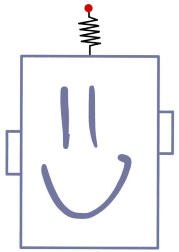
Background

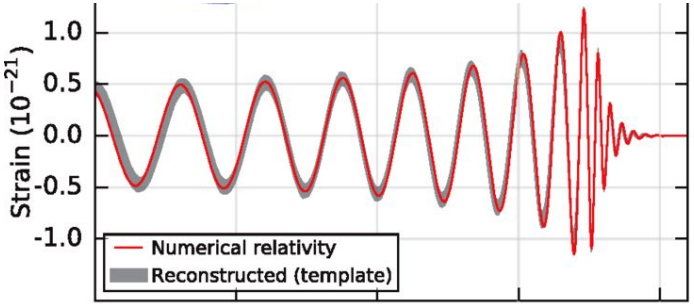
GW Theory vs Observation

- ➔ LIGO + current NR codes =
- ➔ LIGO upgrades + current NR codes =
- ➔ LIGO upgrades + new NR codes =









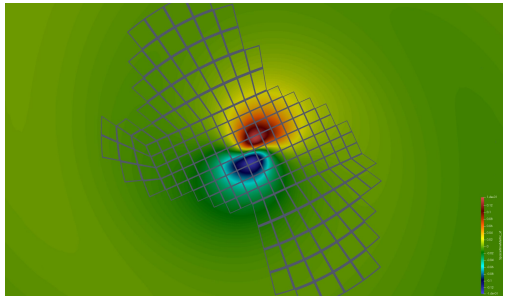
GW150914

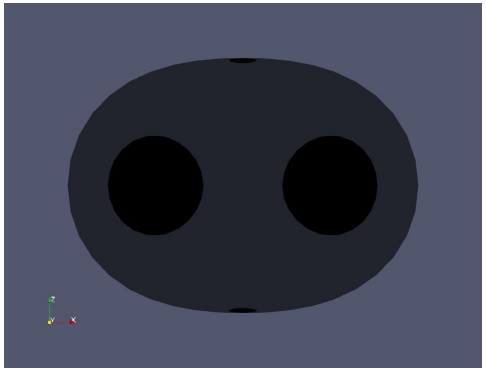
What’s holding current NR codes back?

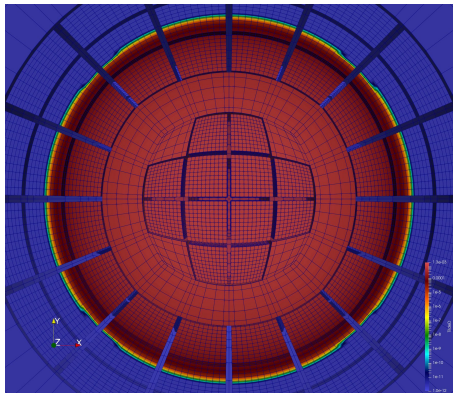
- ➔ Accuracy
- ➔ Speed

- ➔ Numerical methods
  - ◆ Spectral vs Finite difference (FD)
- ➔ Parallelization

SpECTRE - a lead developer







- ➔ Open source
- ➔ Astro- & gravitational physics
- ➔ Spectral methods (Disc. Galerkin (DG))
- ➔ DG + FD hybrid
- ➔ Exascale
- ➔ **Task-based parallelism**

Abstract Concept

MPI vs Task Parallelism

Standard Message Passing Interface (MPI)

- ➔ Prescription for parallelization
- ➔ Non-threaded (all cores run the same code)
- ➔ Synchronous
- ➔ Not easily load balanced
- ➔ 30 years experience

Task-based Parallelism

- ➔ Asynchronous
- ➔ Small amount of code = “task”
- ➔ Threaded
- ➔ “Auto” load balancing
- ➔ Nearest neighbor communication
- ➔ Little experience with PDEs

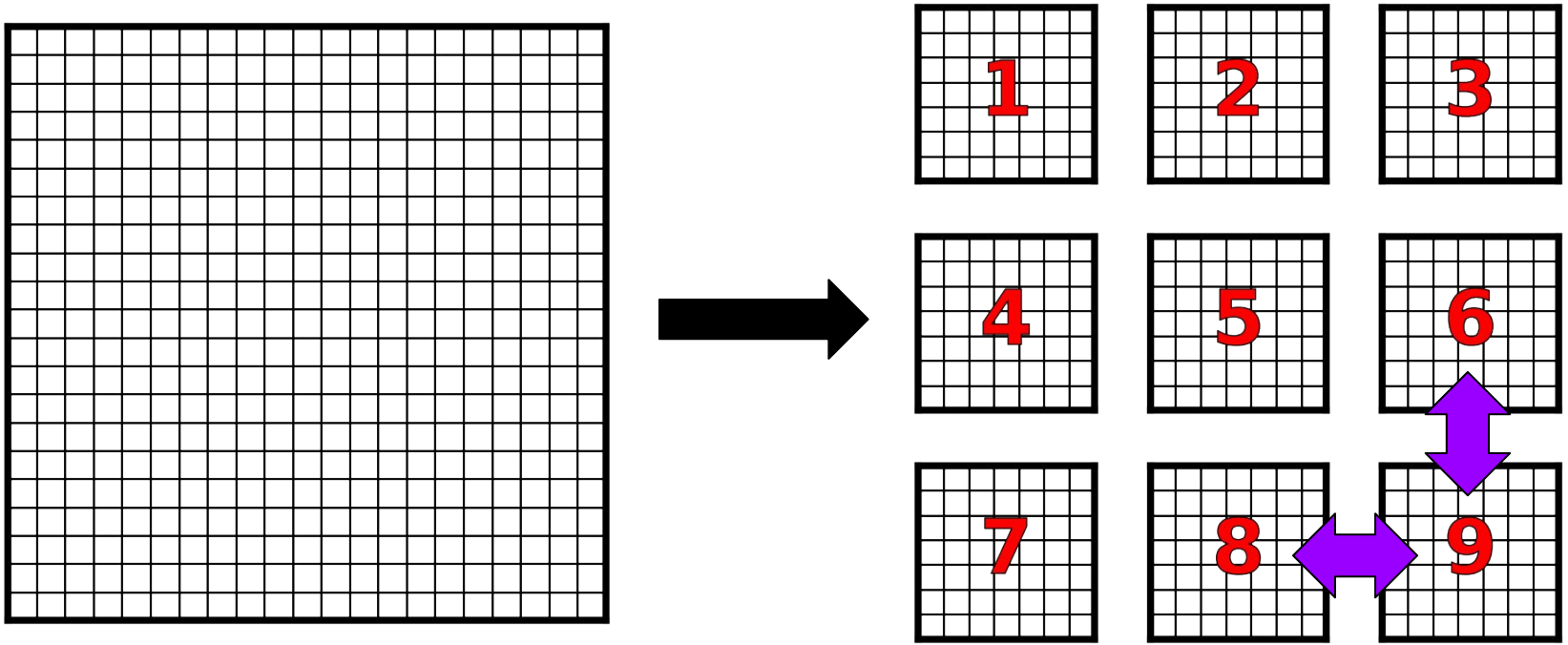
Our type of “tasks”: Solving PDEs!

$$\frac{du}{dt} = f(u, \frac{du}{dx}, ...)$$

$$u(t_2) = u(t_1) + \Delta t \frac{du}{dt}$$

The work done in the task

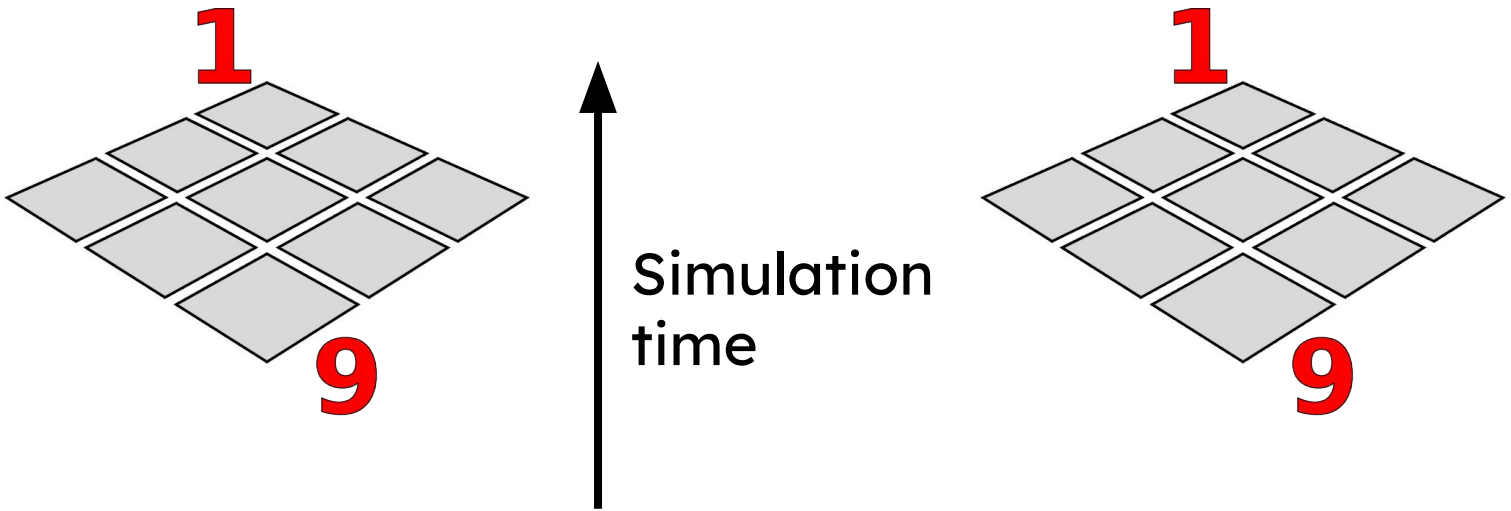
Parallelization



MPI vs task parallelism

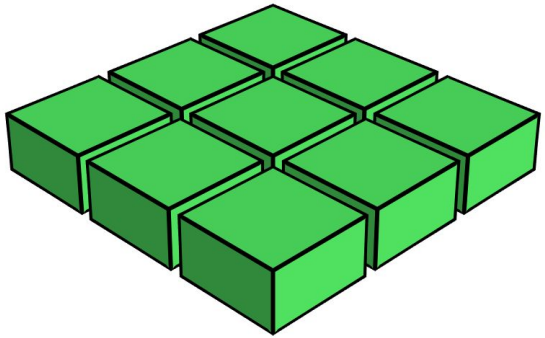
Synchronous

Asynchronous

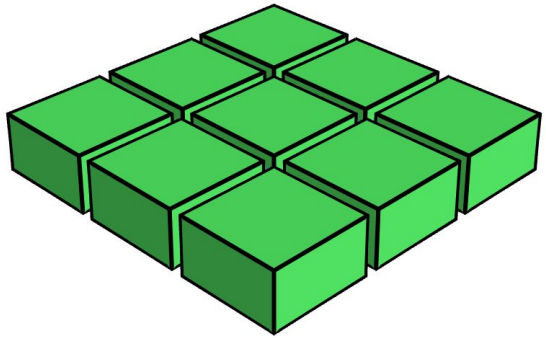


MPI vs task parallelism

Synchronous

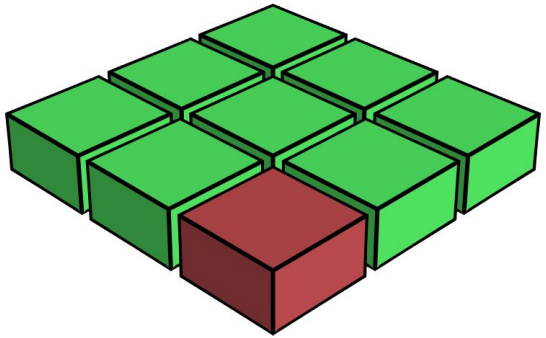


Asynchronous

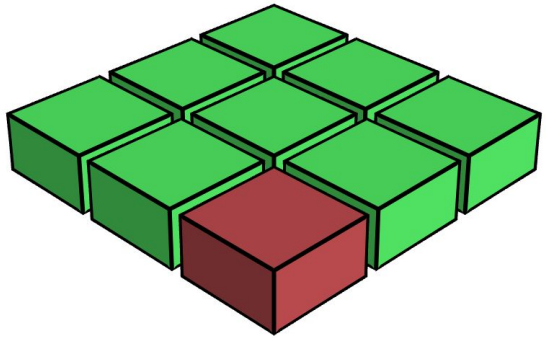


MPI vs task parallelism

Synchronous

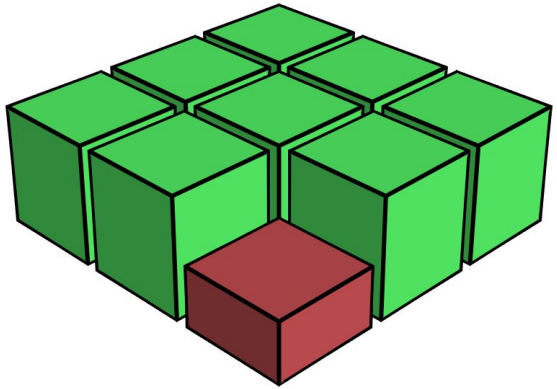


Asynchronous

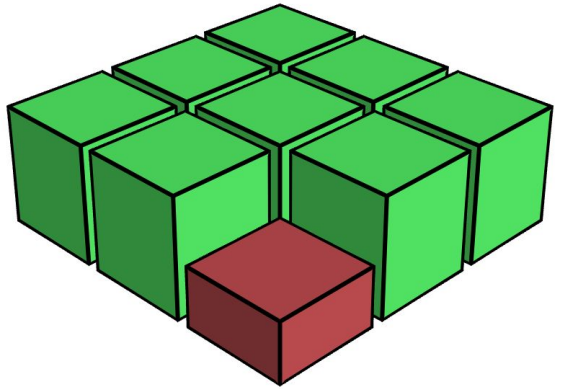


MPI vs task parallelism

Synchronous

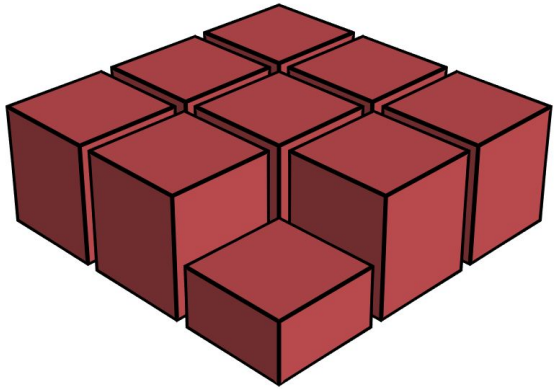


Asynchronous

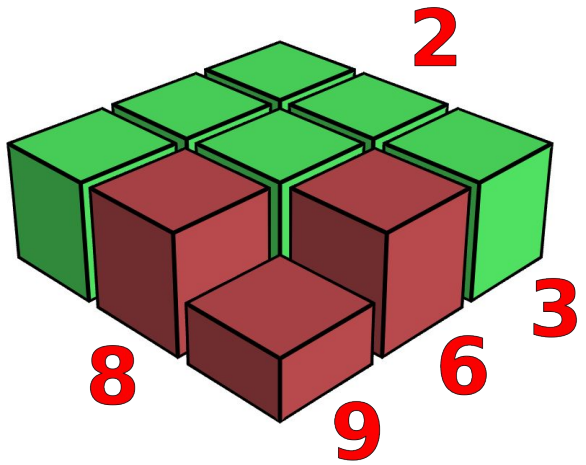


MPI vs task parallelism

Synchronous

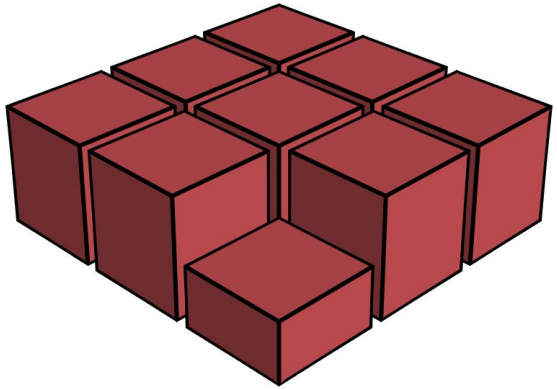


Asynchronous

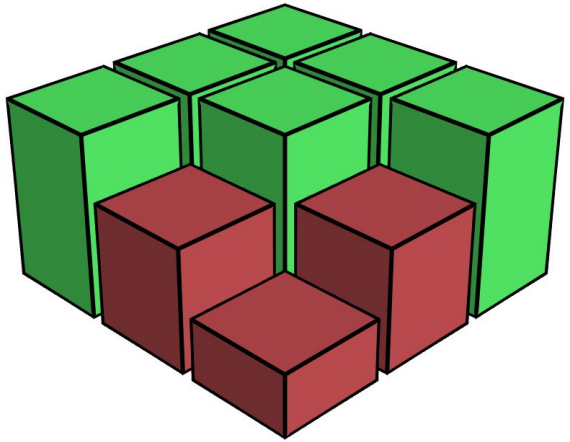


MPI vs task parallelism

Synchronous

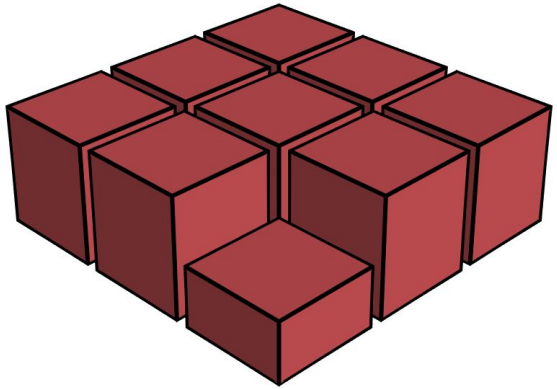


Asynchronous

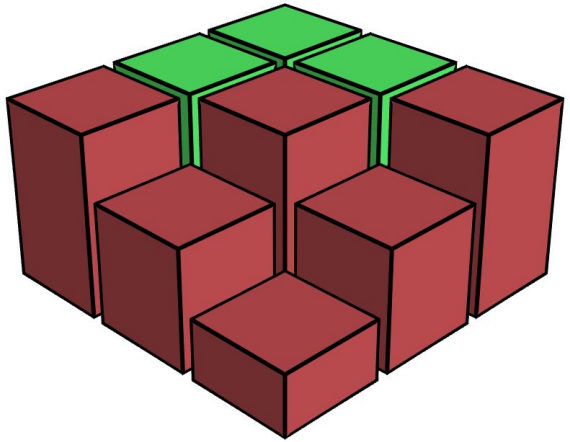


MPI vs task parallelism

Synchronous

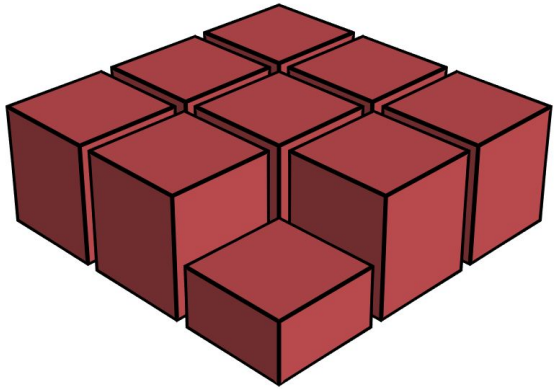


Asynchronous

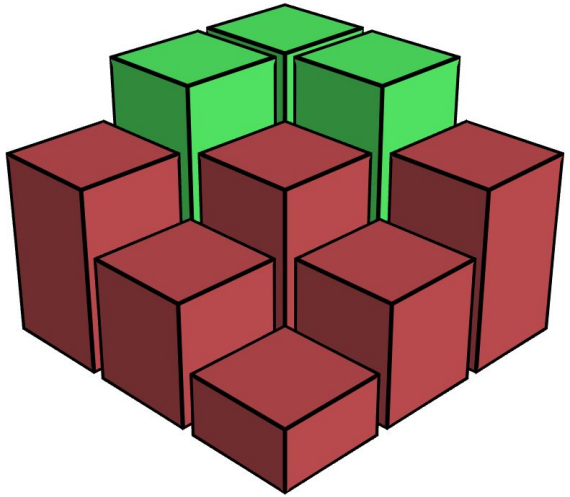


MPI vs task parallelism

Synchronous

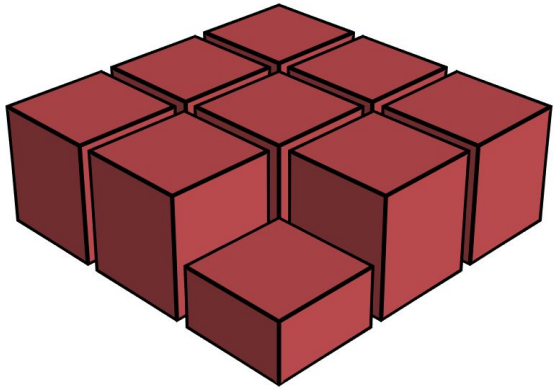


Asynchronous

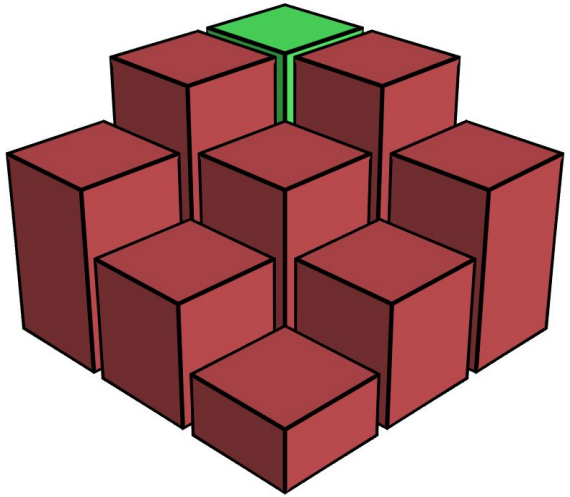


MPI vs task parallelism

Synchronous

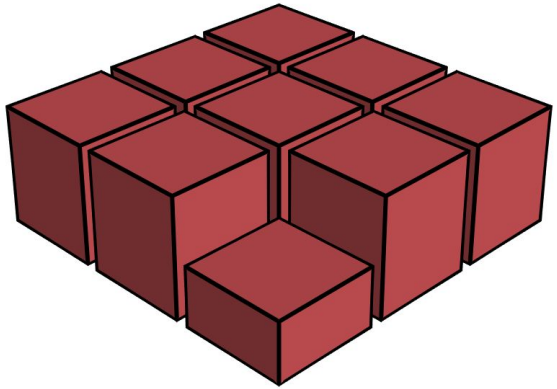


Asynchronous

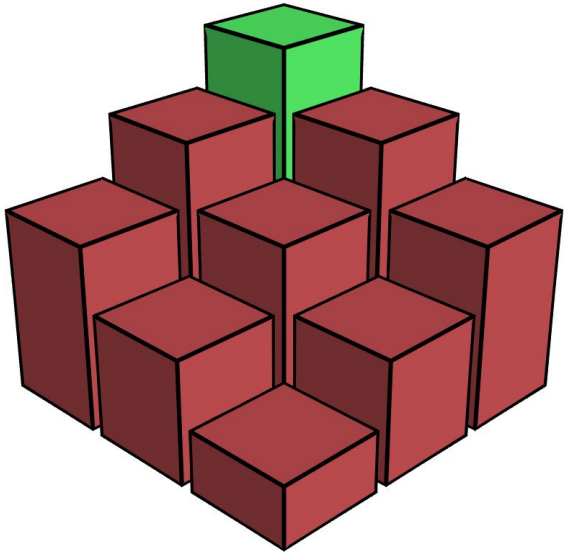


MPI vs task parallelism

Synchronous

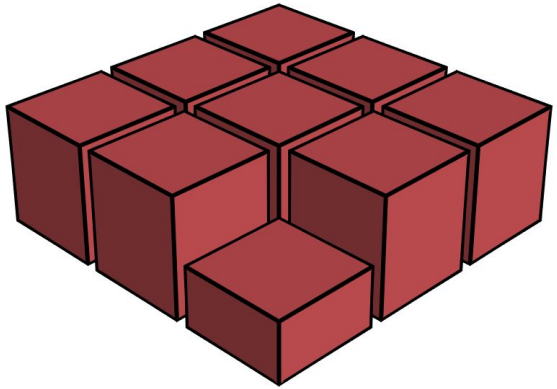


Asynchronous

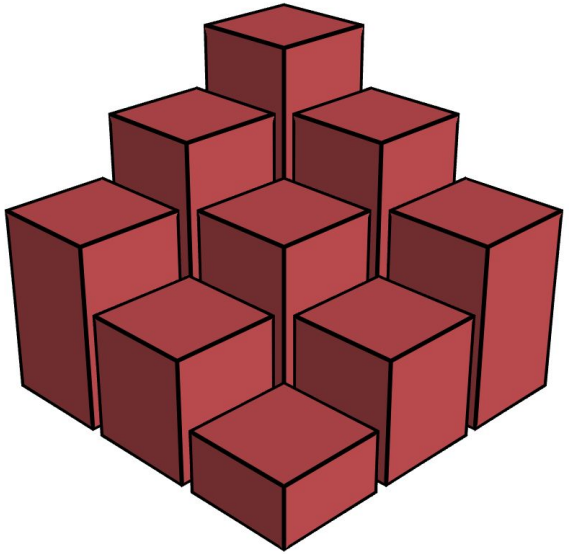


MPI vs task parallelism

Synchronous



Asynchronous



Globally shared variable (global state)

- ➔ Single value shared across all elements
- ➔ Used in evolution
- ➔ Same value, same simulation time
- ➔ Time dependent
- ➔ “Expire”
- ➔ Must be updated

$$\frac{du}{dt} = f(u, \frac{du}{dx}, ...)$$

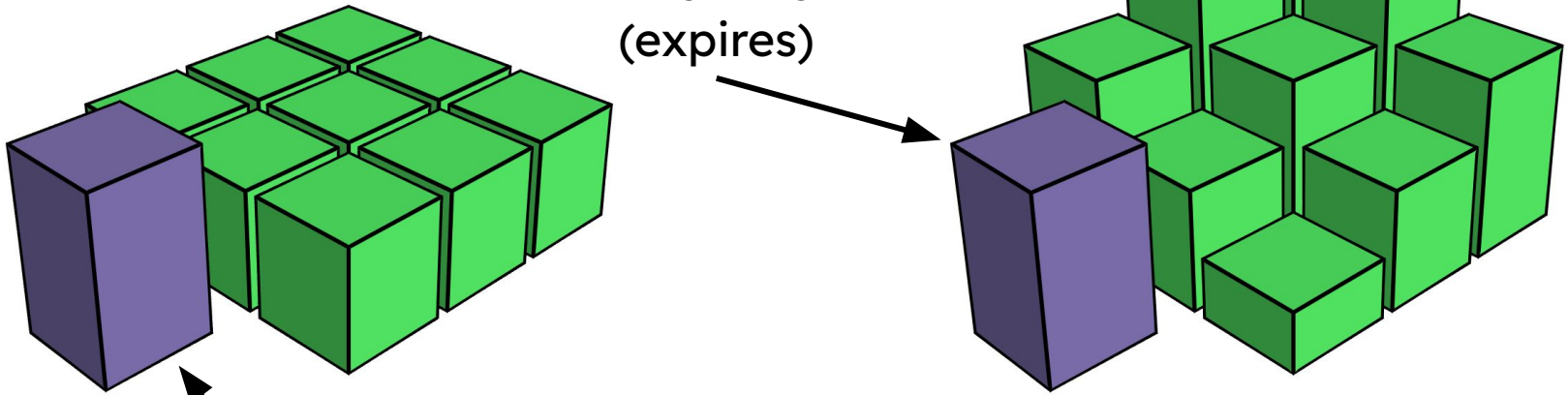
Global state comparison

Synchronous

Asynchronous

Simulation time

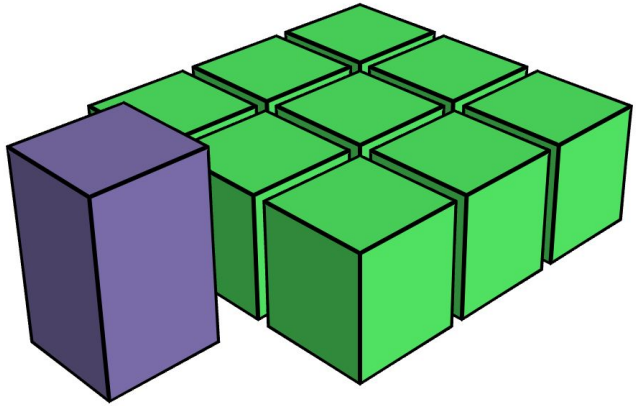
Valid until this time (expires)



Global State

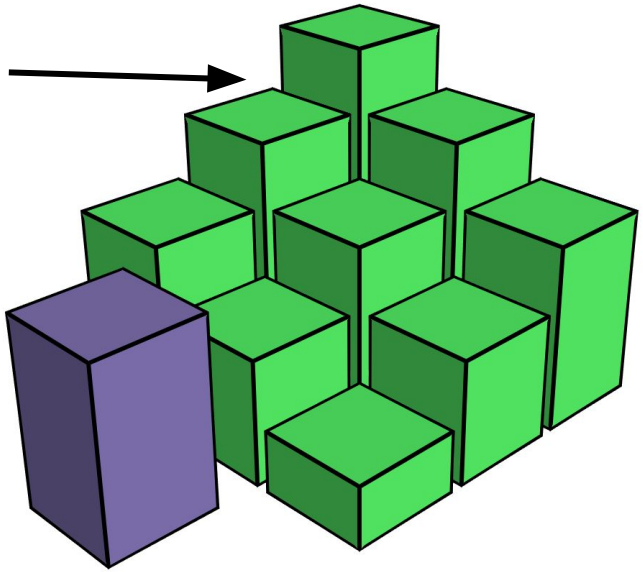
Global state comparison

Synchronous



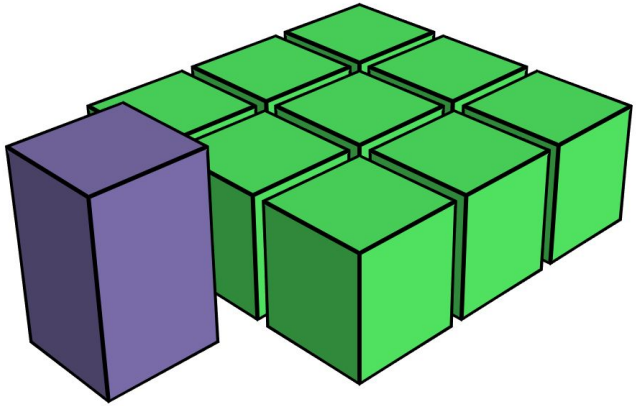
Asynchronous

Impossible configuration

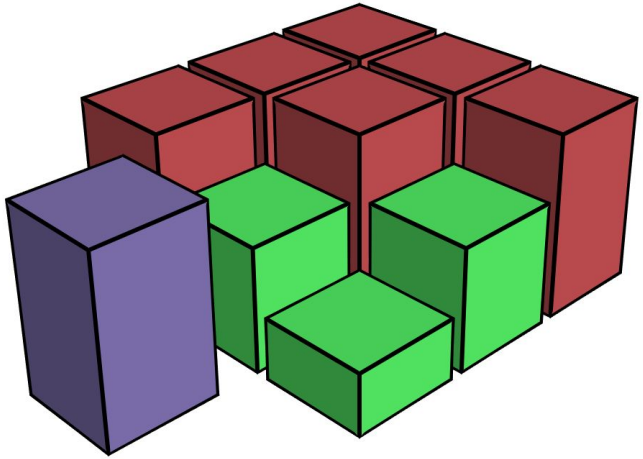


Global state comparison

Synchronous

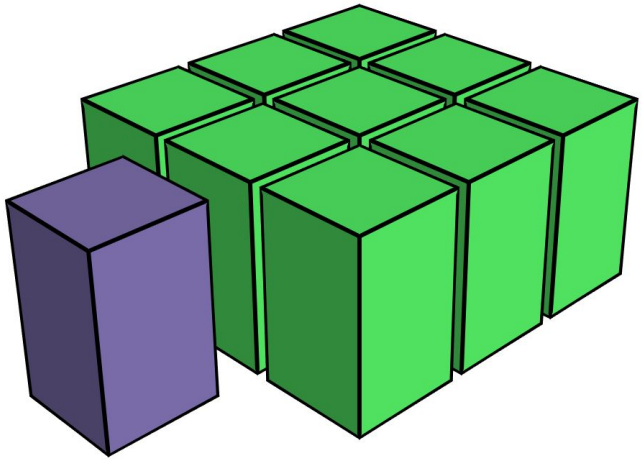


Asynchronous

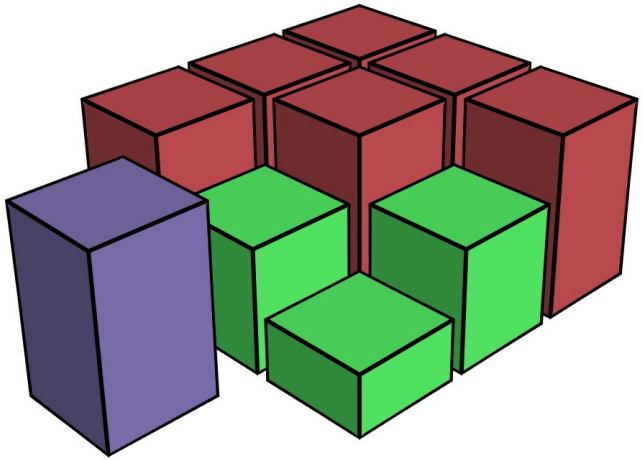


Global state comparison

Synchronous

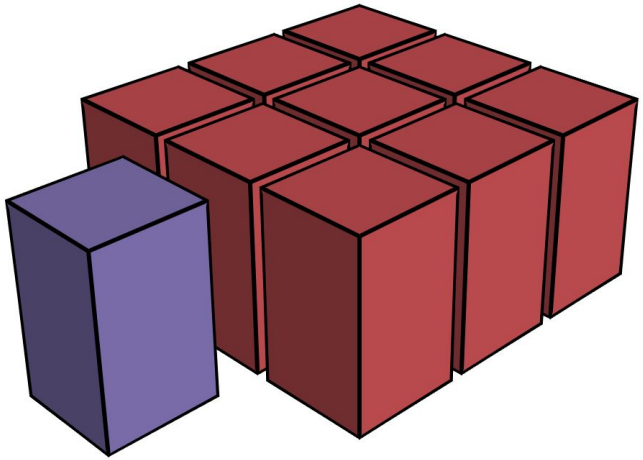


Asynchronous

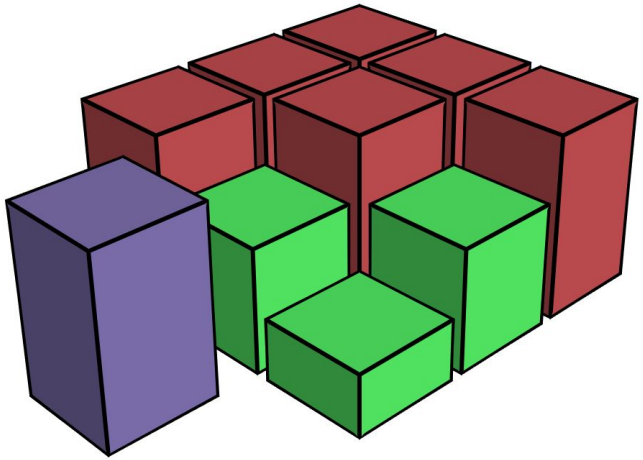


Global state comparison

Synchronous



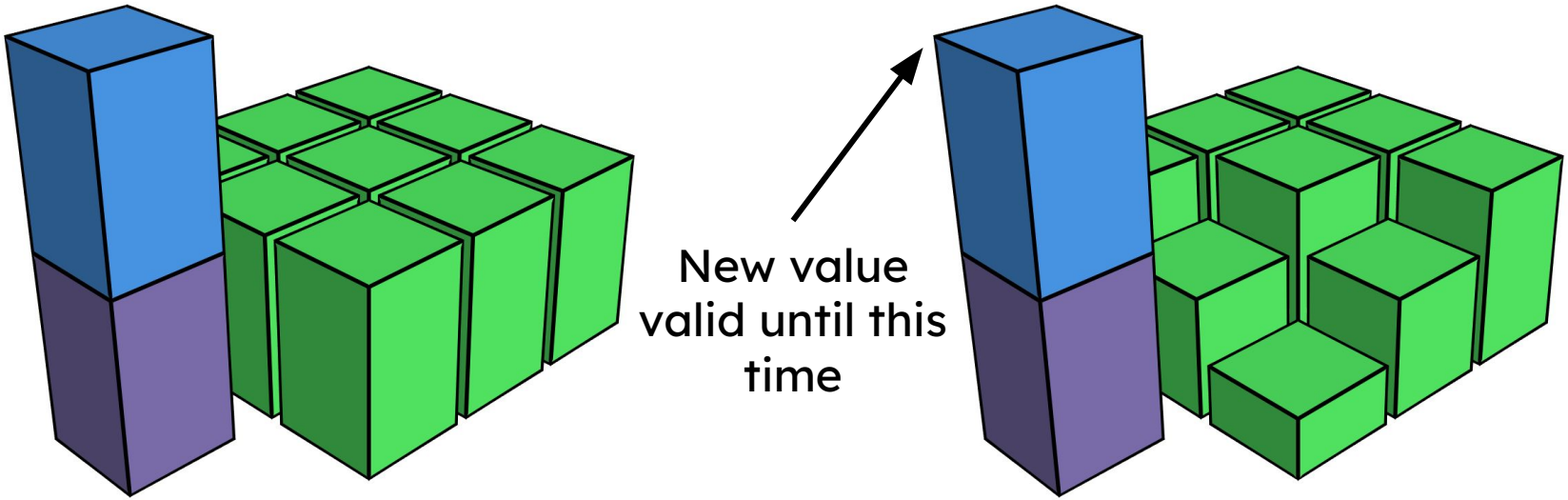
Asynchronous



Global state comparison

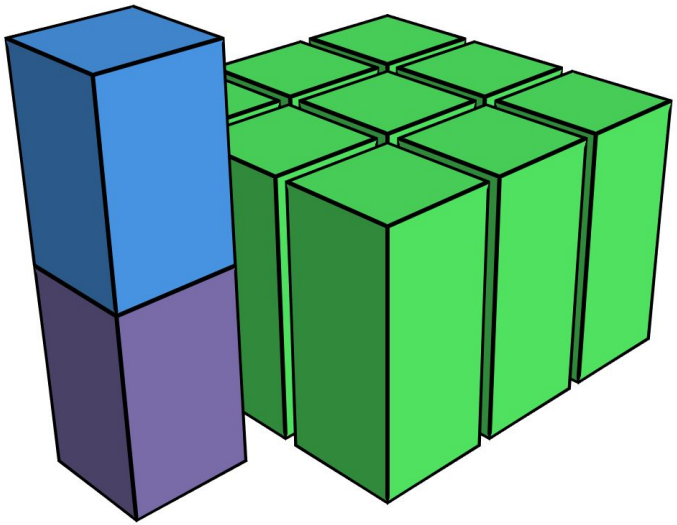
Synchronous

Asynchronous

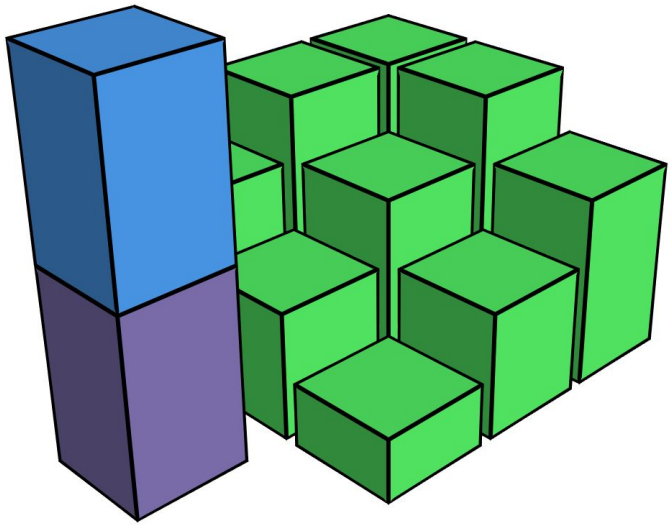


Global state comparison

Synchronous

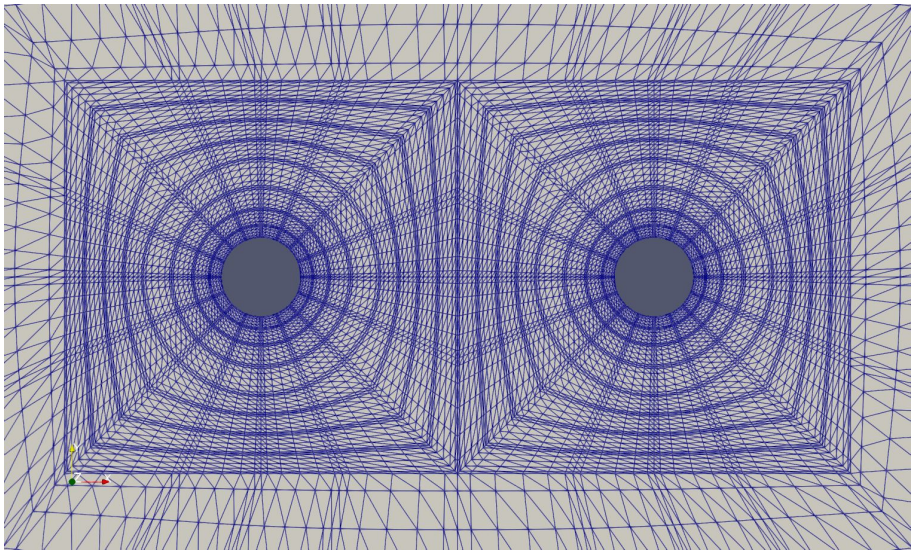


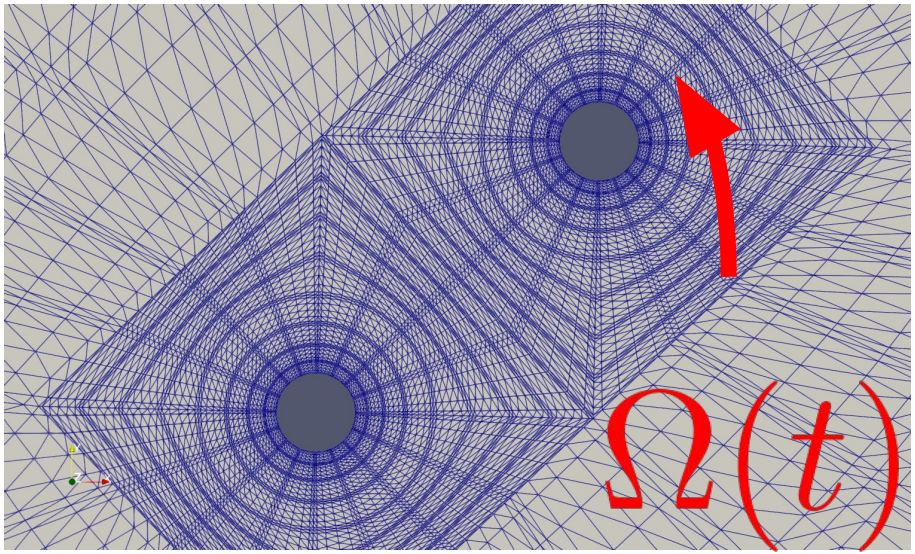
Asynchronous



Concrete Example

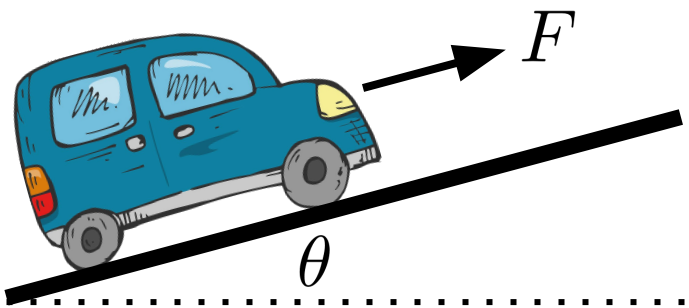
Time dependent rotation mappings





Cruise control for a car

$$\frac{dv}{dt} = F - g\sin\theta$$



Error $\longrightarrow Q(t) = v_0 - v(t)$

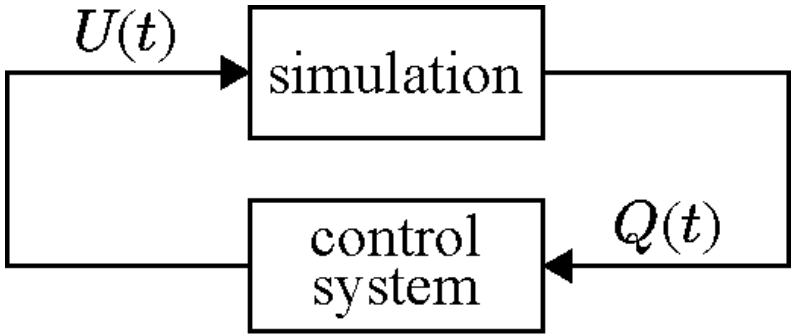
Signal $\longrightarrow U(t) = K_1 Q(t) + K_2 \frac{dQ(t)}{dt} + K_3 \frac{d^2Q(t)}{dt^2}$

$$U(t) \sim \frac{dv}{dt} \implies U(t) = F(t)$$

Cruise control for binary black holes

$$Q(t) = v_0 - v(t) \rightarrow Q(t) = \Omega_{\rm BH} - \Omega(t)$$

$$U(t) \sim \frac{dv}{dt} \rightarrow U(t) = \frac{d\Omega}{dt}$$



Ω = Global State

More complexity!

- ➔ 9 global states, not 1
- ➔ Different expiration times
- ➔ Different update intervals
- ➔ Potential for deadlocks

What is a deadlock?

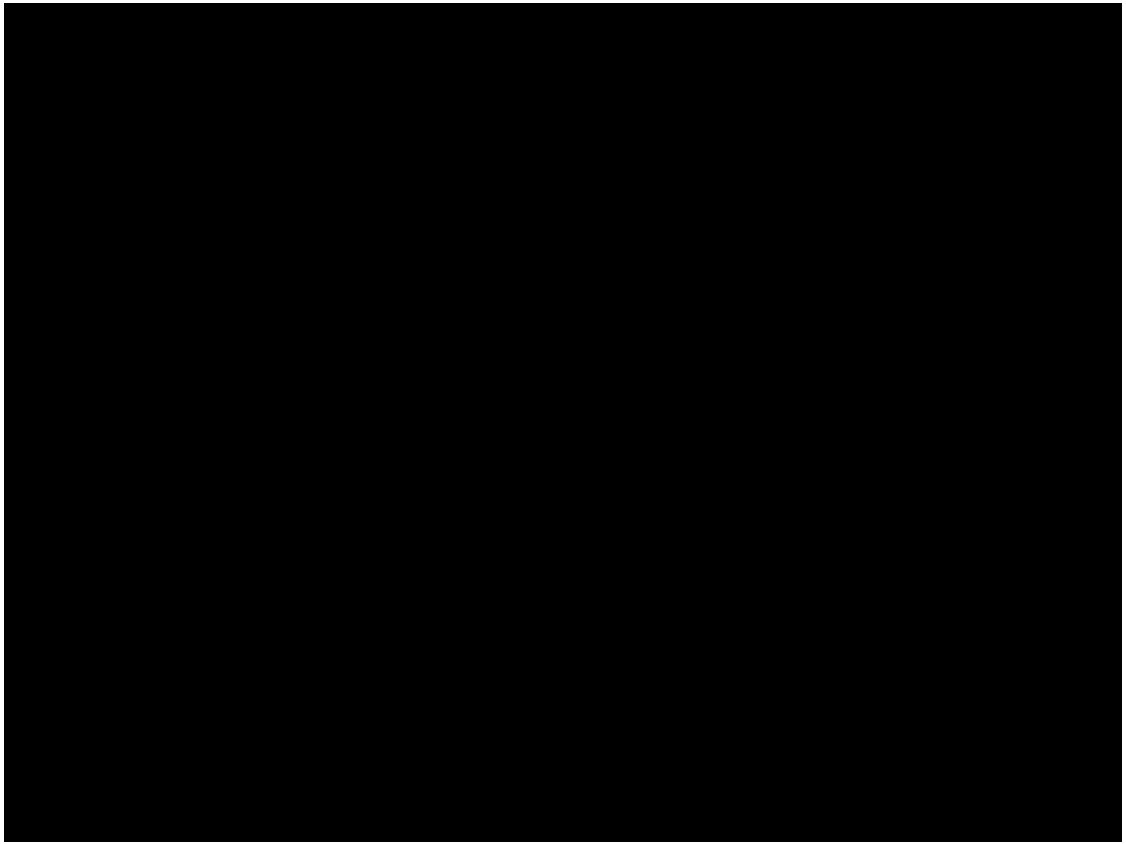
- ➔ An error … but also not …
- ➔ Simulation just *stops*
- ➔ Cyclic dependency
- ➔ Elements waiting for elements
- ➔ $Q(t)$ before $U(t)$
- ➔ $Q(t)$ before expiration
- ➔ $U(t)$ before/@ expiration

How to debug a deadlock

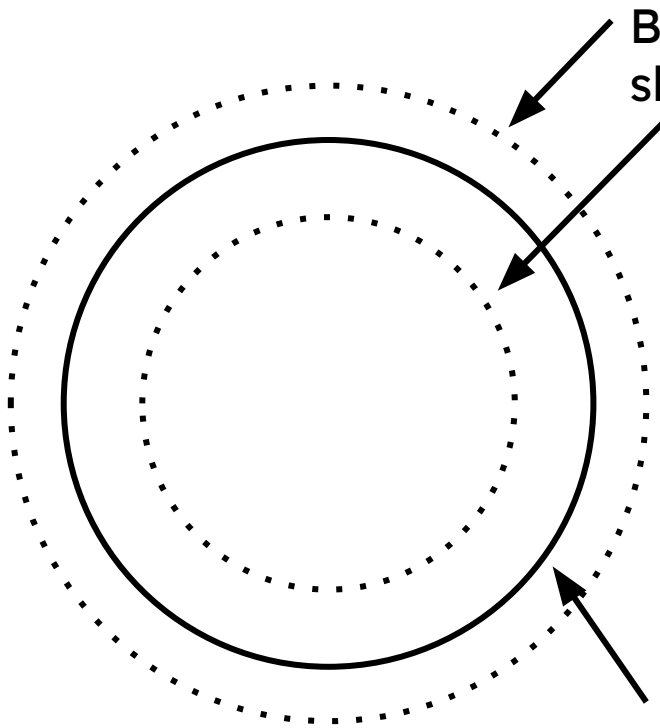
- ➔ Expiration time of global state
- ➔ Time of global state update
- ➔ Measurement for global state update
- ➔ Simulation time of each element
- ➔ What went wrong?
  - ◆ Data not sent?
  - ◆ Data not received?
  - ◆ Unexpected times
- ➔ Implemented data dump

Simulation Use Cases

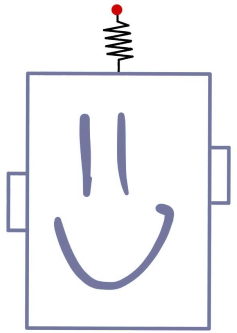
Binary neutron star simulations



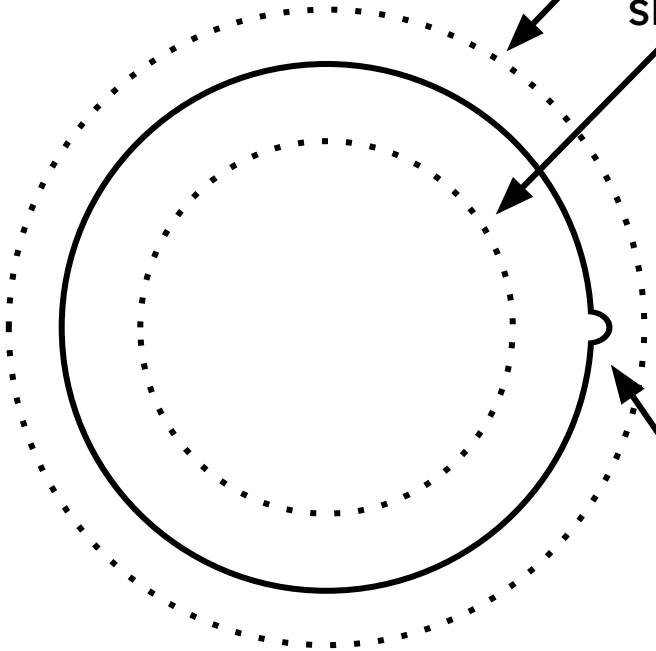
High mass ratio binary black hole simulations



$$Y_{\ell m}(\theta, \phi)$$

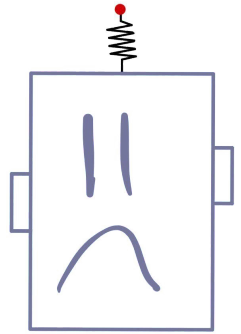


High mass ratio binary black hole simulations



Boundary of single spherical shell element

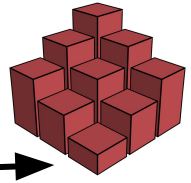
$$Y_{\ell m}(\theta, \phi)$$



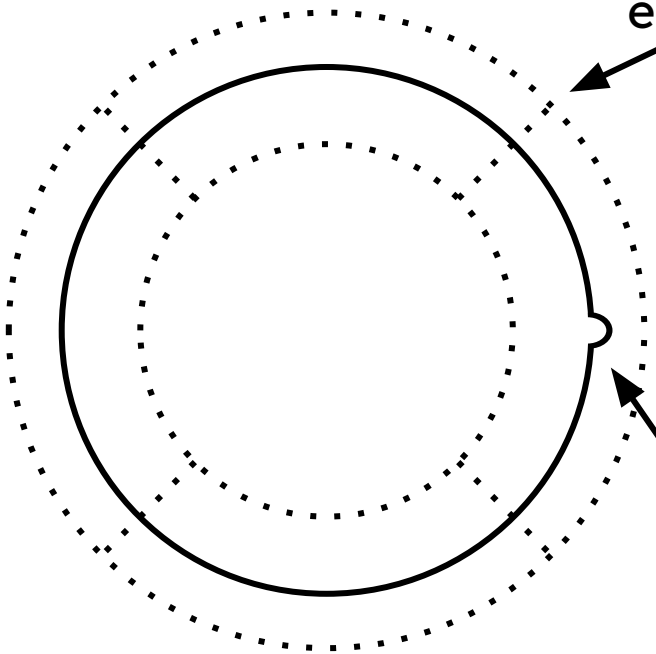
Almost Spherical horizon distorted by smaller black hole

Asynchronous

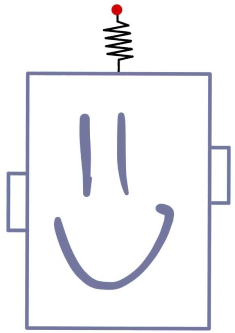
High mass ratio binary black hole simulations



Boundary of many wedge elements

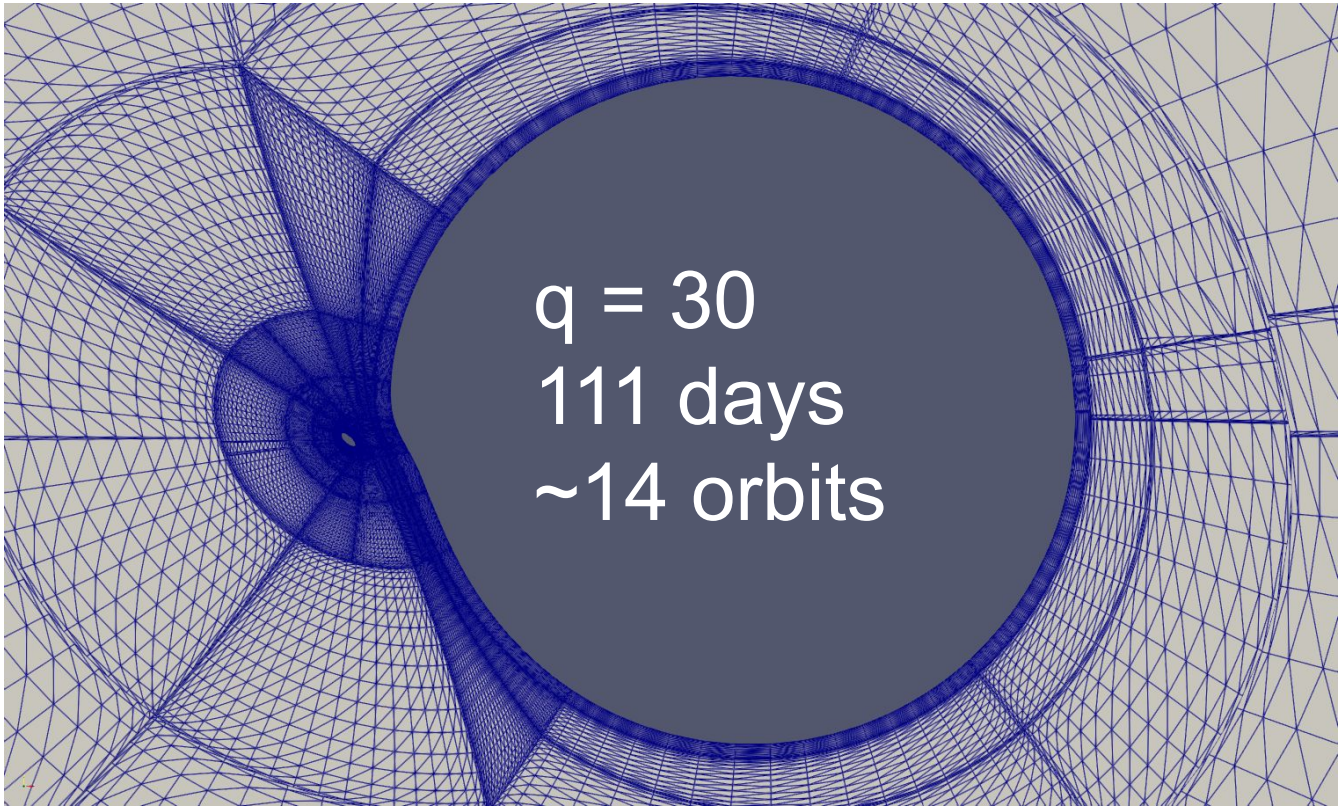


Discontinuous Galerkin



Almost Spherical horizon distorted by smaller black hole

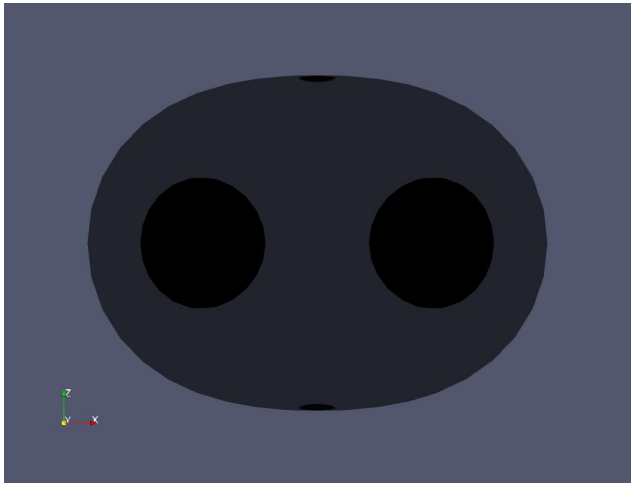
High mass ratio binary black hole simulations



Future Work

Future work

- ➔ Short term
  - Finishing touches for global state
  - Merge BHs robustly
  - Merger to ringdown transition
  - Various code improvements
    - Optimize parallel interpolation routines
- ➔ Long term
  - Code description/task-based paper
  - Small mass ratio BBHs with SpECTRE
  - High mass ratio BBHs with SpECTRE
  - BNSs with SpECTRE



Summary

- ➔ LIGO upgrades + new NR codes
- ➔ MPI vs task parallelism
  - ◆ Synchronous vs asynchronous
- ➔ Global state
  - ◆ Example: BBH angular velocity
  - ◆ Control theory to update
  - ◆ Avoid deadlocks
- ➔ Use cases
  - ◆ BNSs
  - ◆ BBHs
- ➔ Future research
  - ◆ Code development for above concepts
  - ◆ Papers related to efficient and scalable BBHs with SpECTRE