On-the-fly Analysis of Weighted Ensemble Simulations Using Trajectory Streaming

Jamie Rowe

GSoC Mentors: Oliver Beckstein, Jeremy Leung, Lawson Woods, Lillian Chong

A Short Intro to Weighted Ensemble Simulations

- The Weighted Ensemble (WE) approach is an enhanced sampling technique that aims to accelerate rare events and yield estimates of nonequilibrium observables such as rate constants.
- WESTPA is used to run WE simulations – it handles the resampling of walkers and starts the simulations

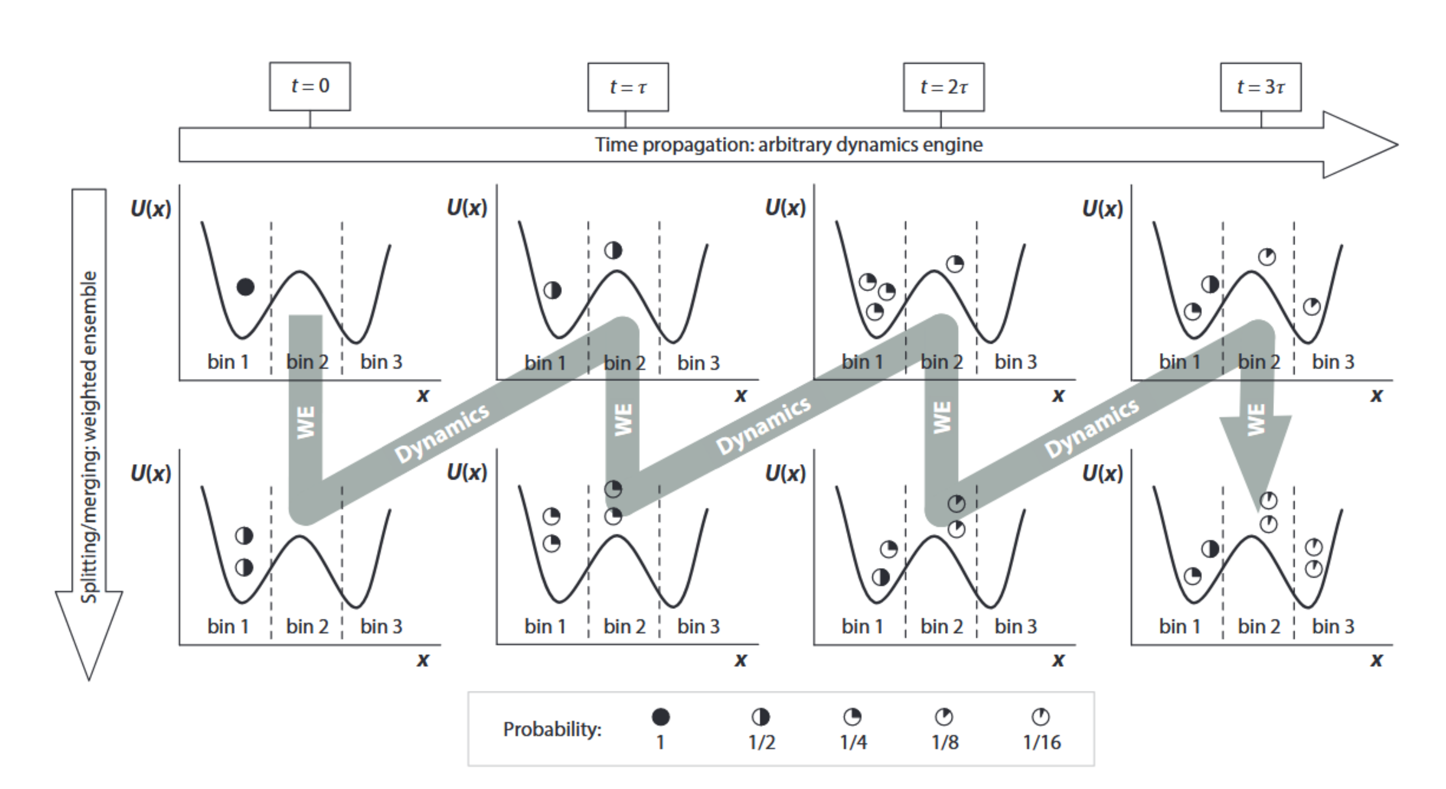
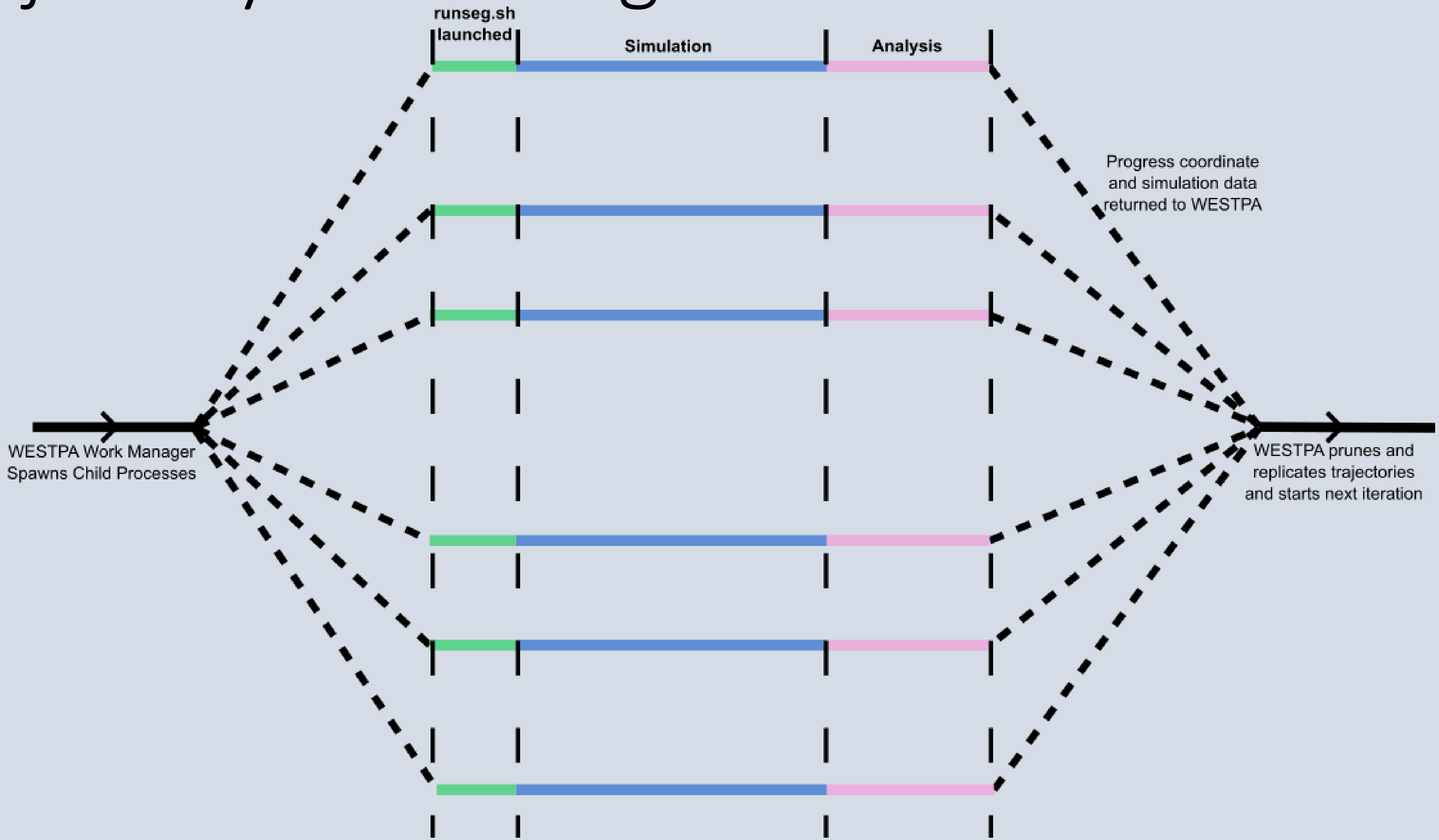


Image from *Zuckerman DM, Chong LT. Weighted Ensemble Simulation: Review of Methodology, Applications, and Software. Annual Review of Biophysics 2017;46:43–57.*

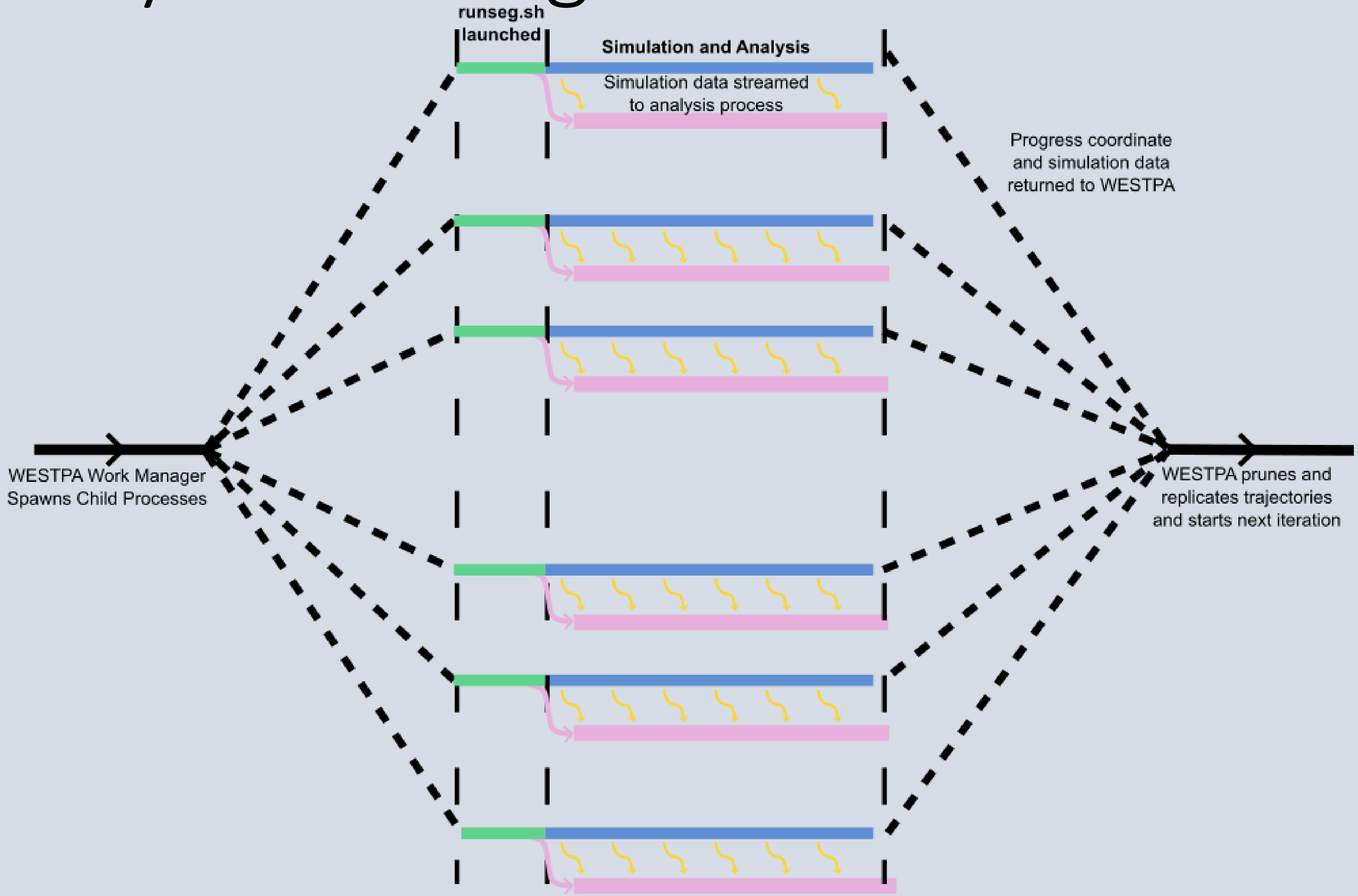
WESTPA

On-the-fly Analysis of Walkers using Trajectory Streaming



runseg.sh launched
Simulation
Analysis
Progress coordinate and simulation data returned to WESTPA
WESTPA Work Manager Spawns Child Processes
WESTPA prunes and replicates trajectories and starts next iteration

On-the-fly Analysis of Walkers using Trajectory Streaming



Weighted Ensemble Simulations: A Test Case for Streaming

- From previous talks we have seen how to use **IMDClient** and **IMDReader** to analyse MD trajectories on-the-fly
- Simulations are started using:

```
gmx mdrun -imdwait -imdport 8889
```

- An MDAnalysis universe is created using:

```
u = mda.Universe("topol.tpr","imd://localhost:8889”)
```

- **How can we handle the hundreds of simulations being run simultaneously in our WE simulation?**

Challenges of Applying Trajectory Streaming to WE Walkers

1. Port Uniqueness

IMD simulations need to be assigned a unique port

How do you assign a port to each walker and guarantee that it is not in use by another process or another WESTPA walker?

2. Port Competition

What if we find a free port, but another process (from WESTPA or another program) grabs it before our simulation starts?

3. Port Communication

WESTPA runs 100s of segments in each iteration.

How can we ensure that analysis script #87 is communicating on the same port as simulation #87?

Solution: Let the Engine Choose the Port

Instead of *telling* the simulation a port, what if we let the *engine find one itself*?

- We use a "wrapper script" that launches the simulation and requests **Port 0**
- The simulation engine binds to the first free port it finds and **prints that port number** to its output.
- The wrapper script *captures* this port number and then launches the analysis process, *passing it the correct port*.
- This solution **only works for GROMACS**.
- NAMD and LAMMPS do not support "Port 0" binding. They *must* be given a port upfront, which throws us into potential "Port Contention" problems.

The *TrajectoryStreamer* Class

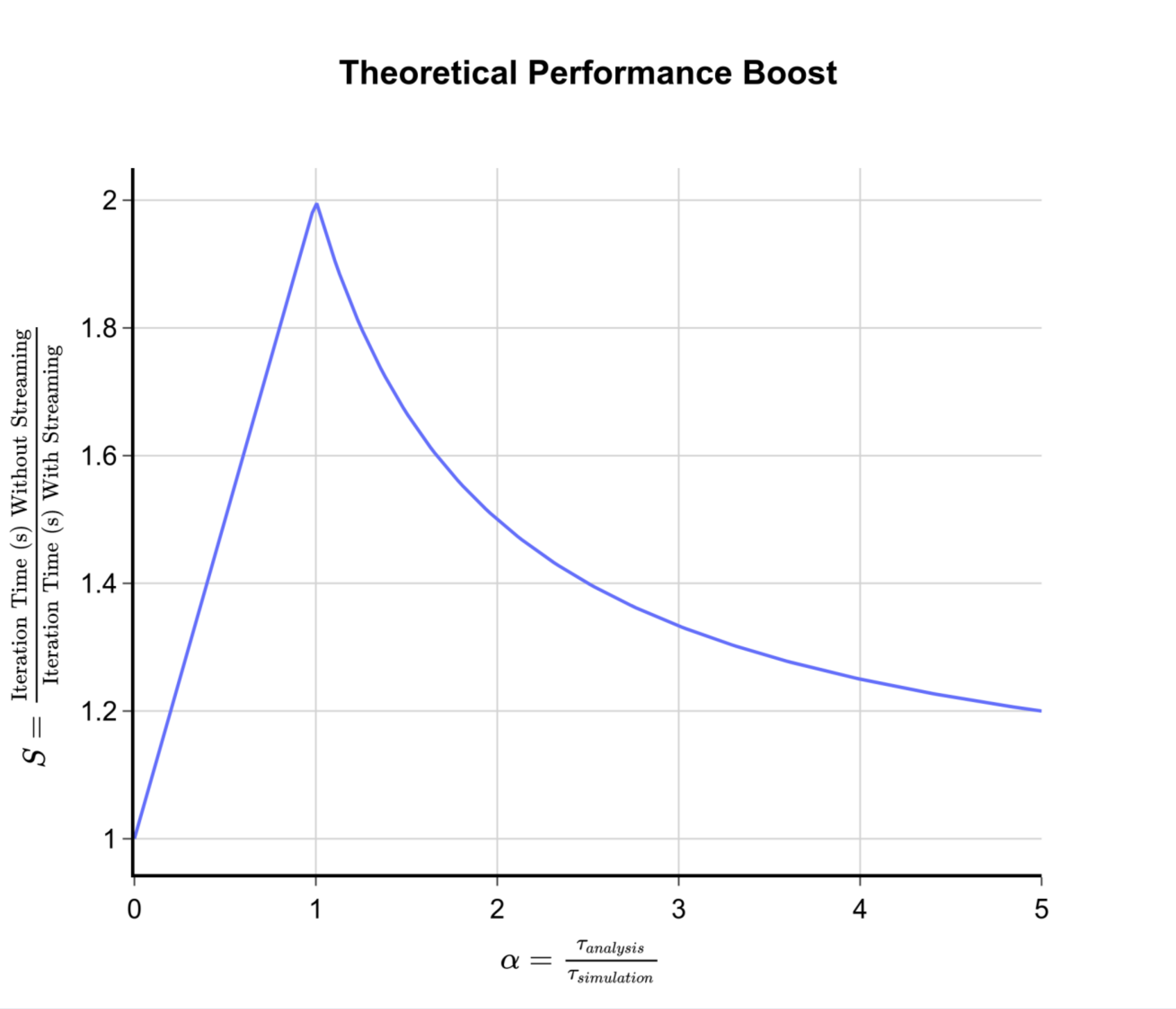
- We've built all this logic into a single class: *TrajectoryStreamer.*
- The class aims to abstract away the complexity of port assignment, contention and communication
- It's designed to be called by the runseg.sh script that WESTPA executes for each walker.
- **Availability:** Find it in this WESTPA fork:
  - https://github.com/jpkrowe/westpa/tree/traj-streaming-tools

The *TrajectoryStreamer* Class: Example Code

```
from westpa.tools.trajectory_streaming import TrajectoryStreamer


stream = TrajectoryStreamer("gromacs", "template.gro", "gmx_imd mdrun -s
seg.tpr -o seg.trr -c seg.gro -e seg.edr -cpo seg.cpt -g seg.log -nt 5 -
imdwait -imdport 0")


u = stream.start_sim_and_get_universe(timeout=30)
ag = u.select_atoms("protein and name CA")
for ts in u.trajectory:
        dists = self_distance_array(ag, box=ts.dimensions)


stream.end_sim()
```

What Speed Up Can We Expect?

- Without streaming, a single walker takes $\tau_{sim} + \tau_{analysis}$ to complete
- When performed in parallel we expect a single walker to take $\approx \max(\tau_{sim}, \tau_{analysis})$ to complete
- The speed up is therefore: $S = \frac{1+\alpha}{max(1,\alpha)}$, where $\alpha = \frac{\tau_{\text{analysis}}}{\tau_{\text{simulation}}}$

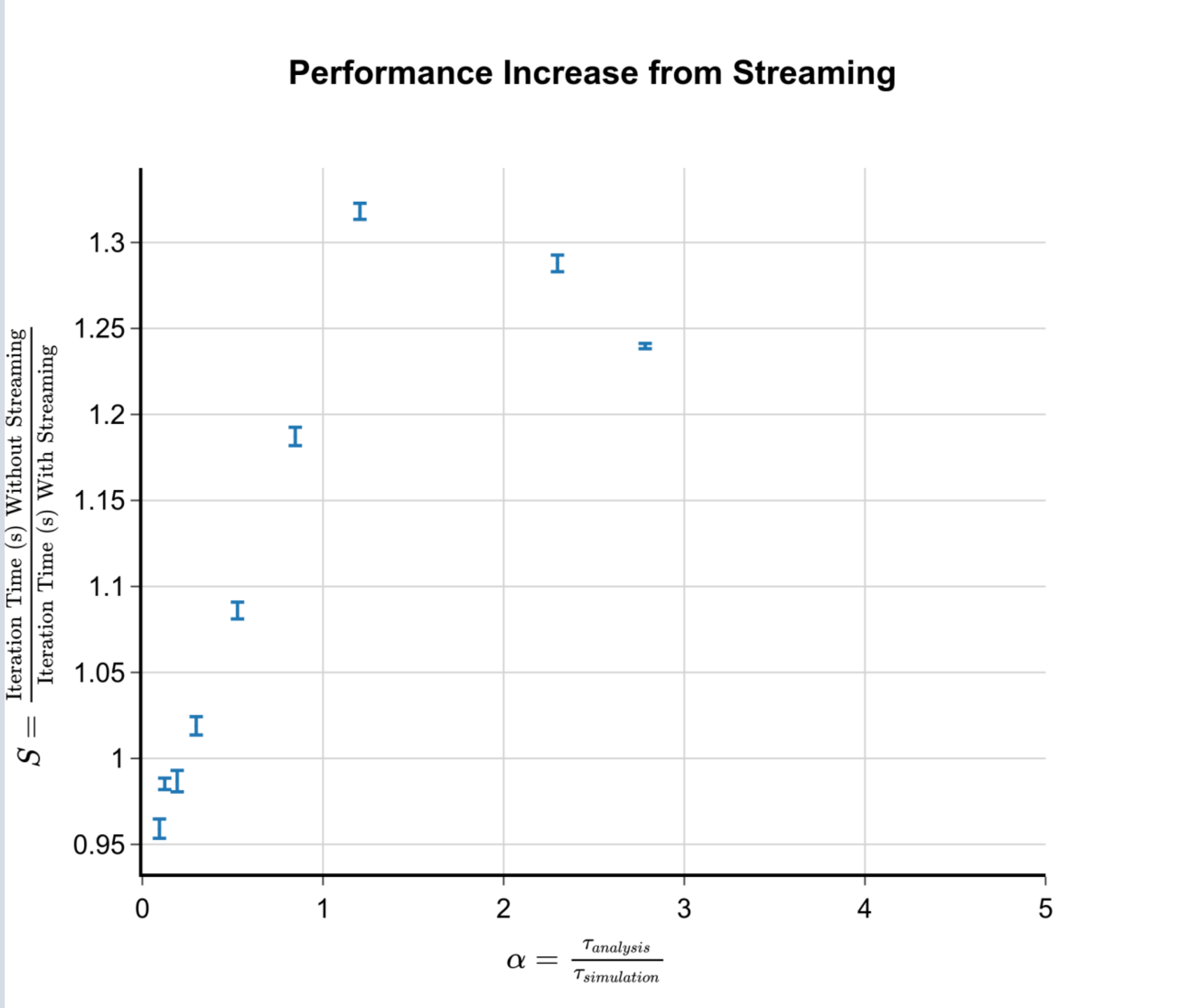


Benchmarking On a Real System

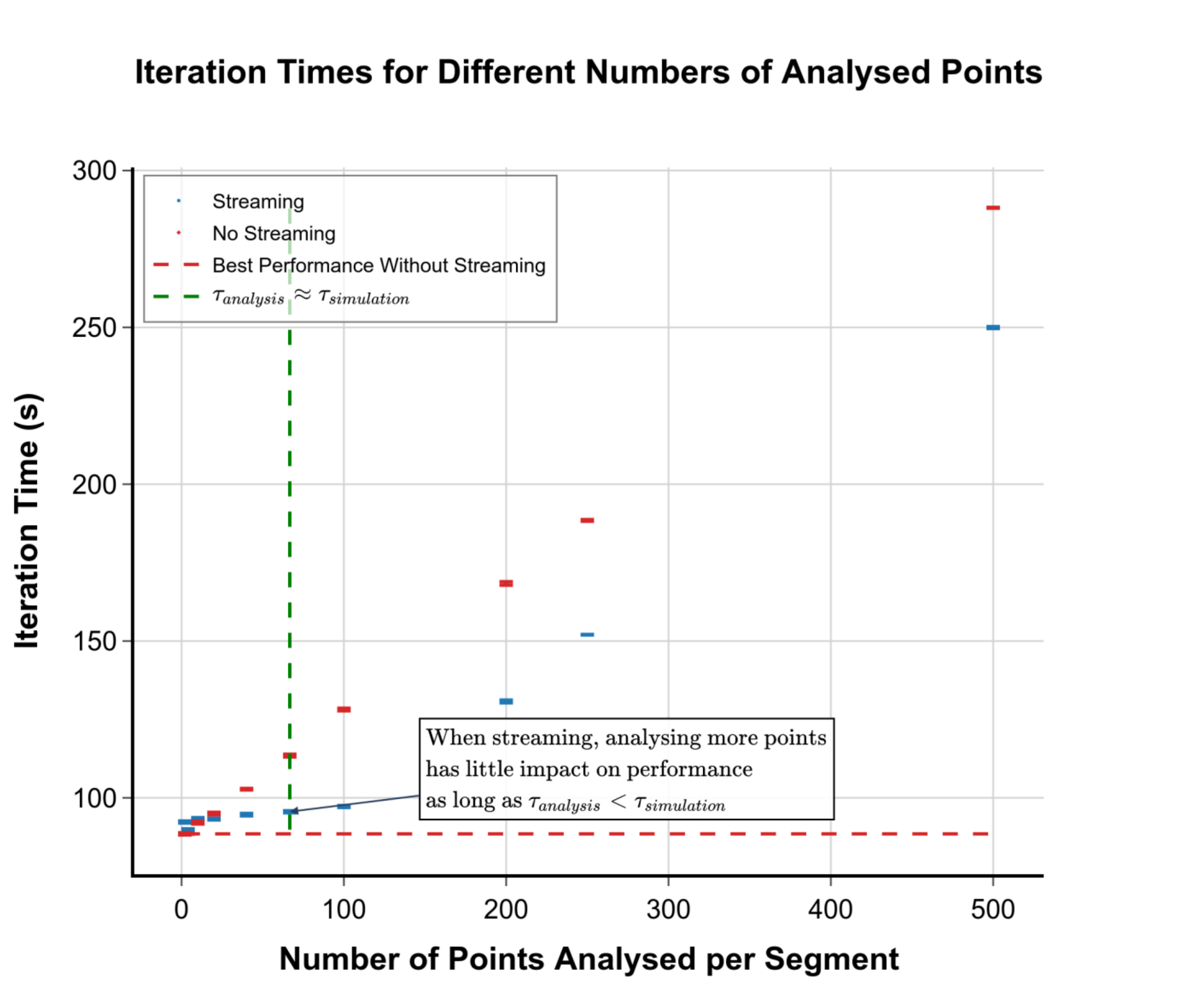
- Test System: Collagen fibril (≈3000 amino acid residues)
- Analysis: pairwise distances between residues

~1.3x

Max Real-World Speedup



Streaming Allows More Analysis “For Free”



When streaming, analysing more points has little impact on performance as long as $\tau_{analysis} < \tau_{simulation}$

When Is Streaming Useful?

This gives a simple test to see if streaming will speed up the WE run:

- Measure the Simulation Time for one segment.
- Measure the Analysis Time for that same segment.

- If the Analysis Time is **very short** streaming is **not** worth the setup and overhead.

Streaming provides a speedup as long as:

- Analysis Time ≤ Simulation Time
- If you're in this zone, you can **increase the amount/complexity of analysis** with little extra time cost.

What Next?

- **Improve NAMD & LAMMPS Integration**
  - Right now, they still face the "Port Contention" risk. The next goal is to engineer a robust solution that makes them as reliable as the GROMACS "Port 0" method.
- **Enable Multi-Node Architectures**
  - *Currently:* Analysis and simulation for a single worker are "locked" to the same node.
  - *Aim:* Permit Analysis and Simulation to run on separate nodes

WESTPA

Google
Summer of Code

MD
ANALYSIS

Acknowledgements

GSoC Mentors: Oliver Beckstein, Jeremy Leung, Lawson Woods, Lillian Chong

MDAnalysis and WESTPA communities

Thanks For Listening!