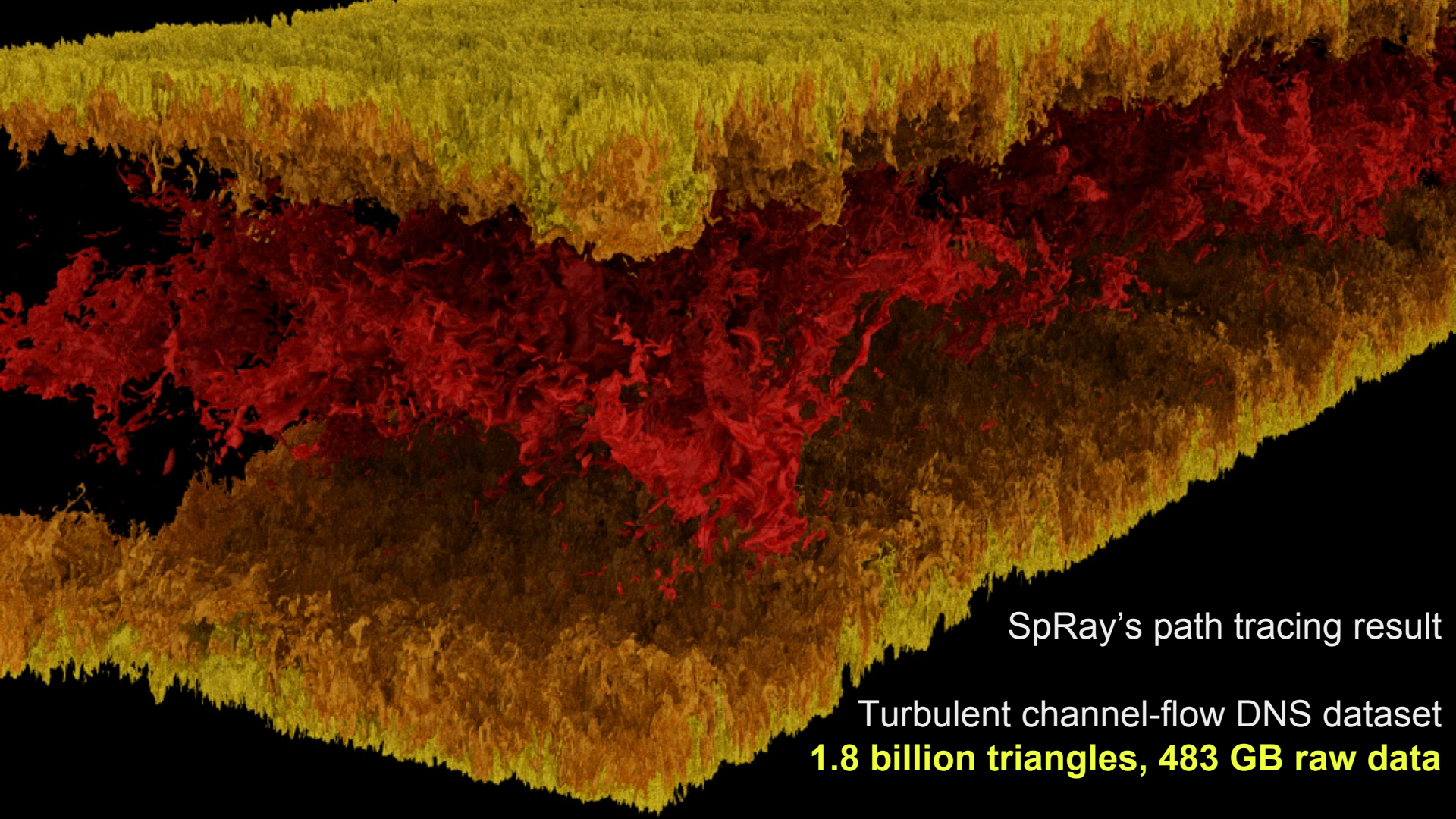
SpRay: Speculative Ray Scheduling for Large Data Visualization

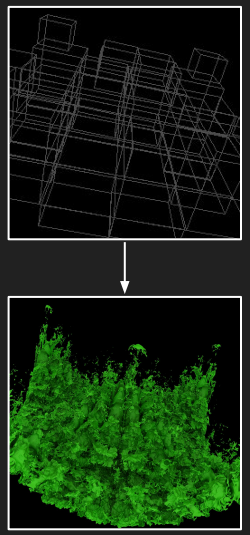
Hyungman Park[*‡] Donald Fussell[†] Paul Navrátil[‡]

[*]Electrical and Computer Engineering [†]Computer Science [‡]Texas Advanced Computing Center
The University of Texas at Austin

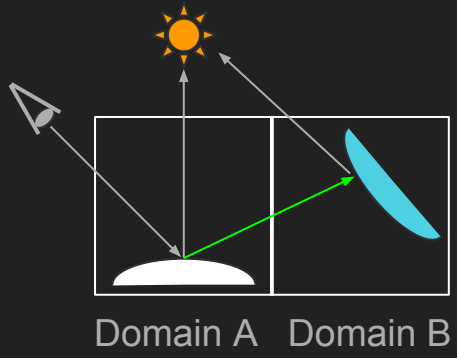

SpRay's path tracing result
Turbulent channel-flow DNS dataset
1.8 billion triangles, 483 GB raw data

Challenges for rendering large data

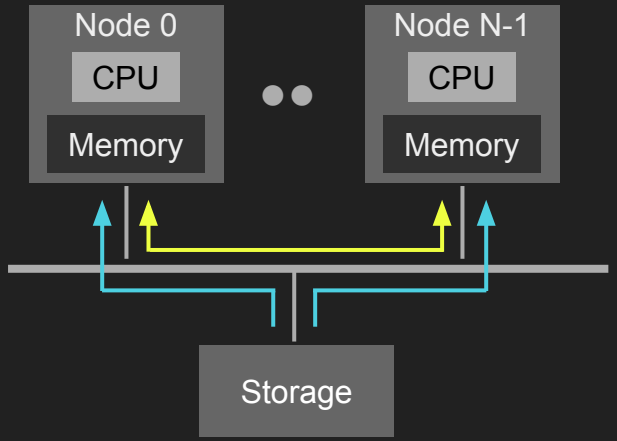
Subdivide the scene into domains



Rays traverse different domains



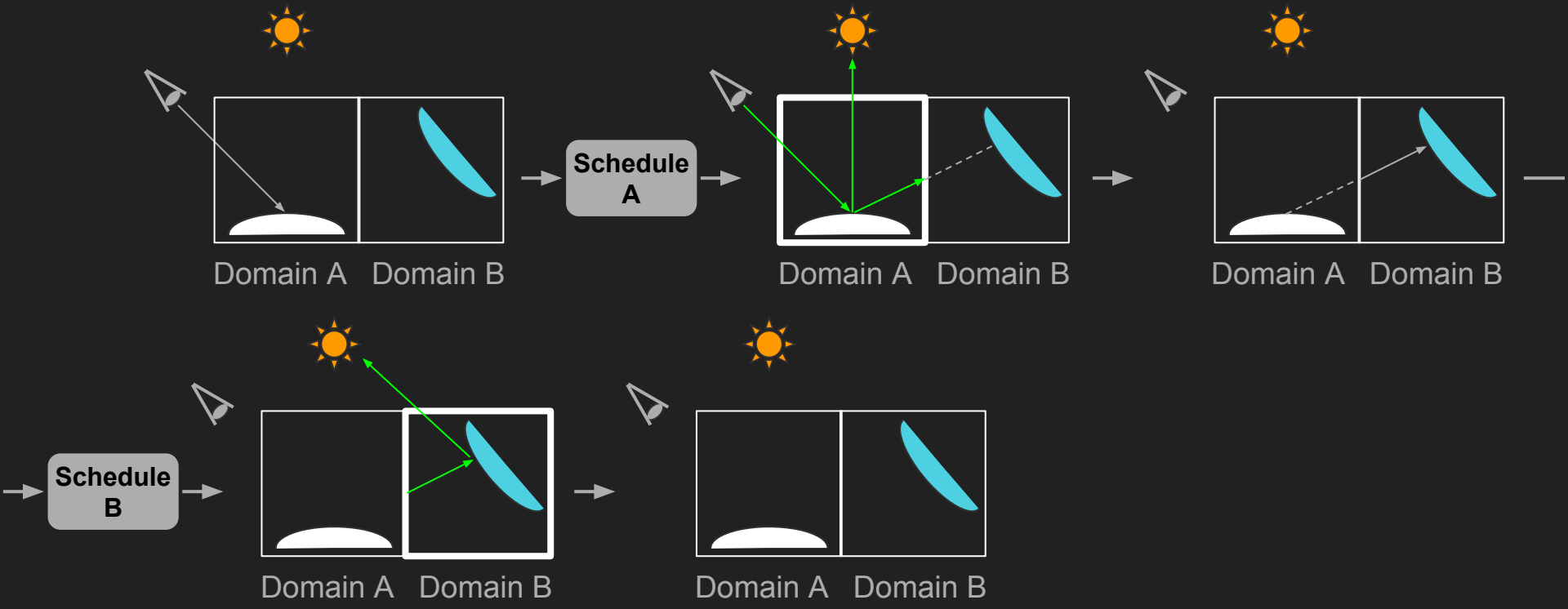
Send rays and/or load domains as needed



Come up with a smart **schedule** that reduces expensive **I/O costs** while maximizing **locality and parallelism** for ray processing

Problem

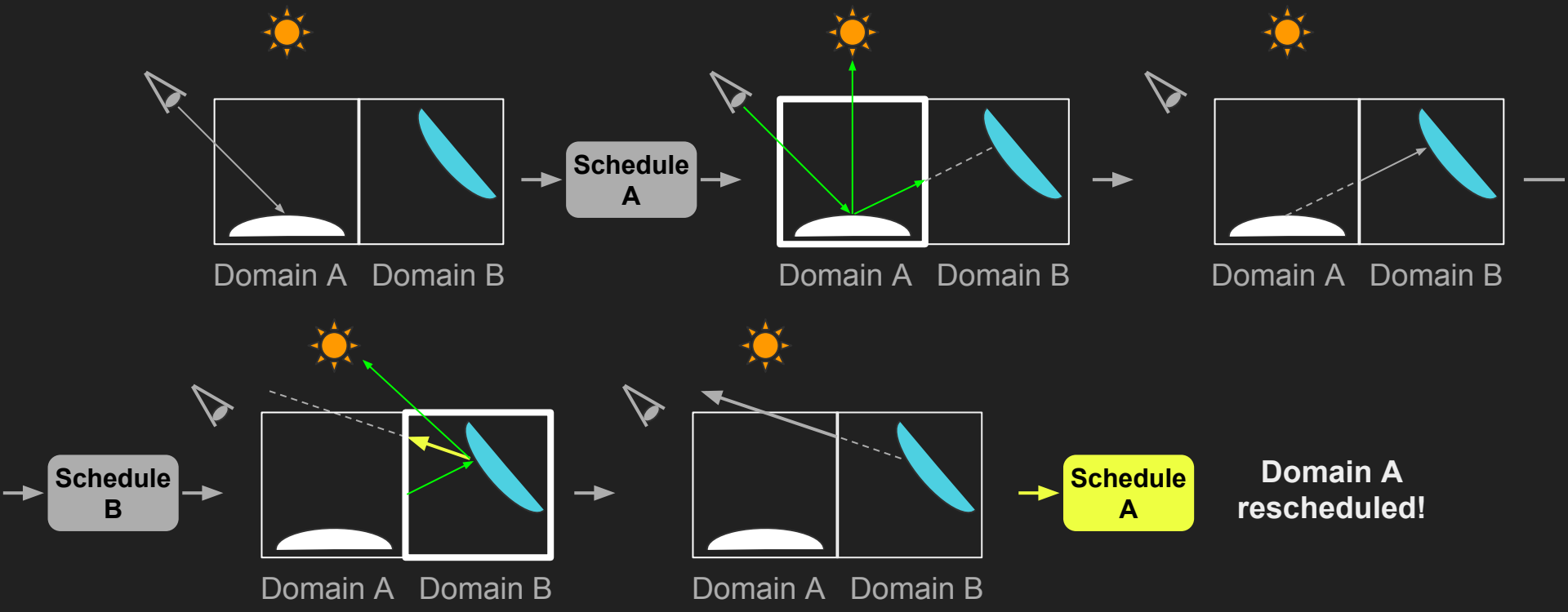
Baseline algorithm*: ray queuing and scheduling



* [Pharr 1997] [Navrátil 2013] [Son 2017]

Secondary rays result in frequent rescheduling

Problem: costly I/O operations for out-of-core rendering

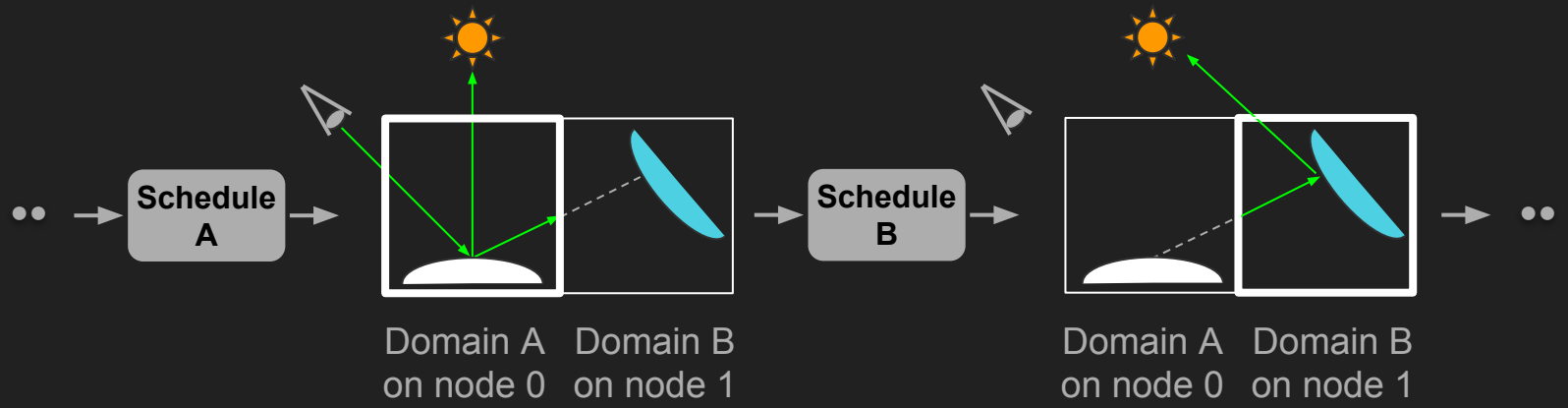


Traversal order results in low processor utilization

Q: Which domain should a ray be placed in?

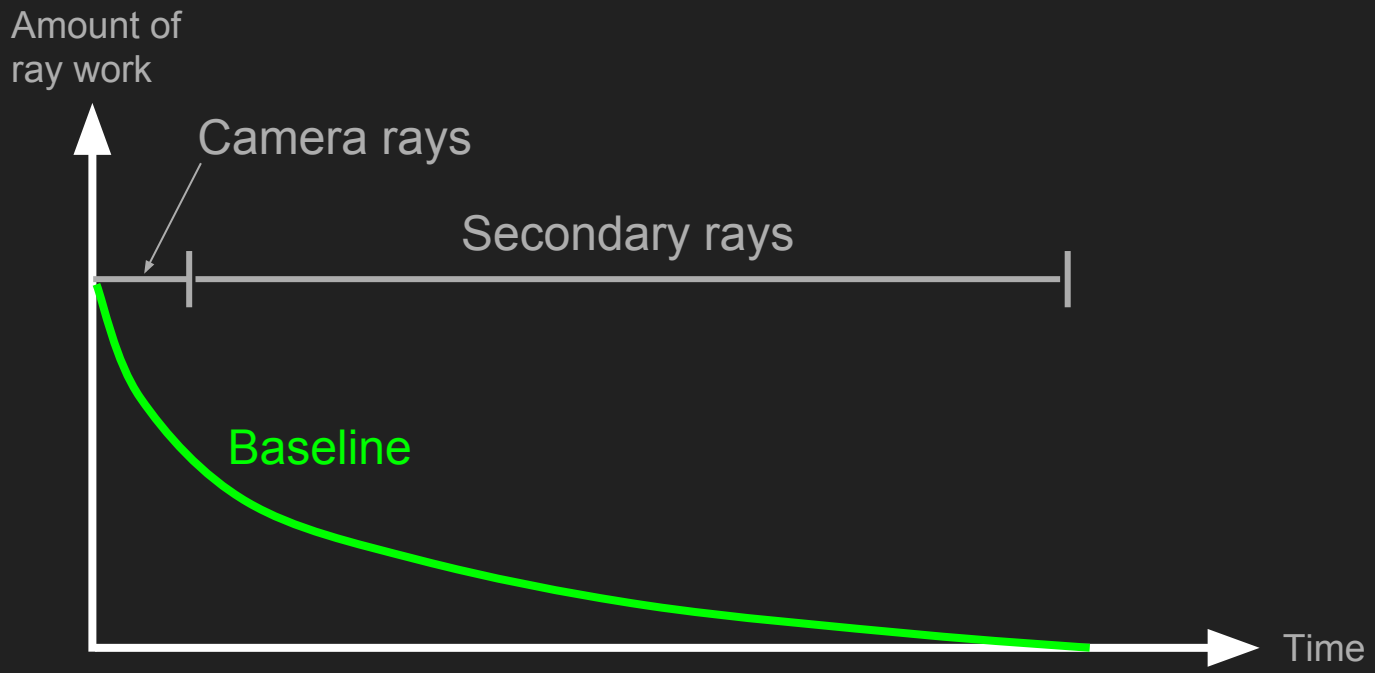
Or, which domain might have the first hit point for a given ray?

A: Most methods assume **the domain closest to the ray origin** is mostly likely



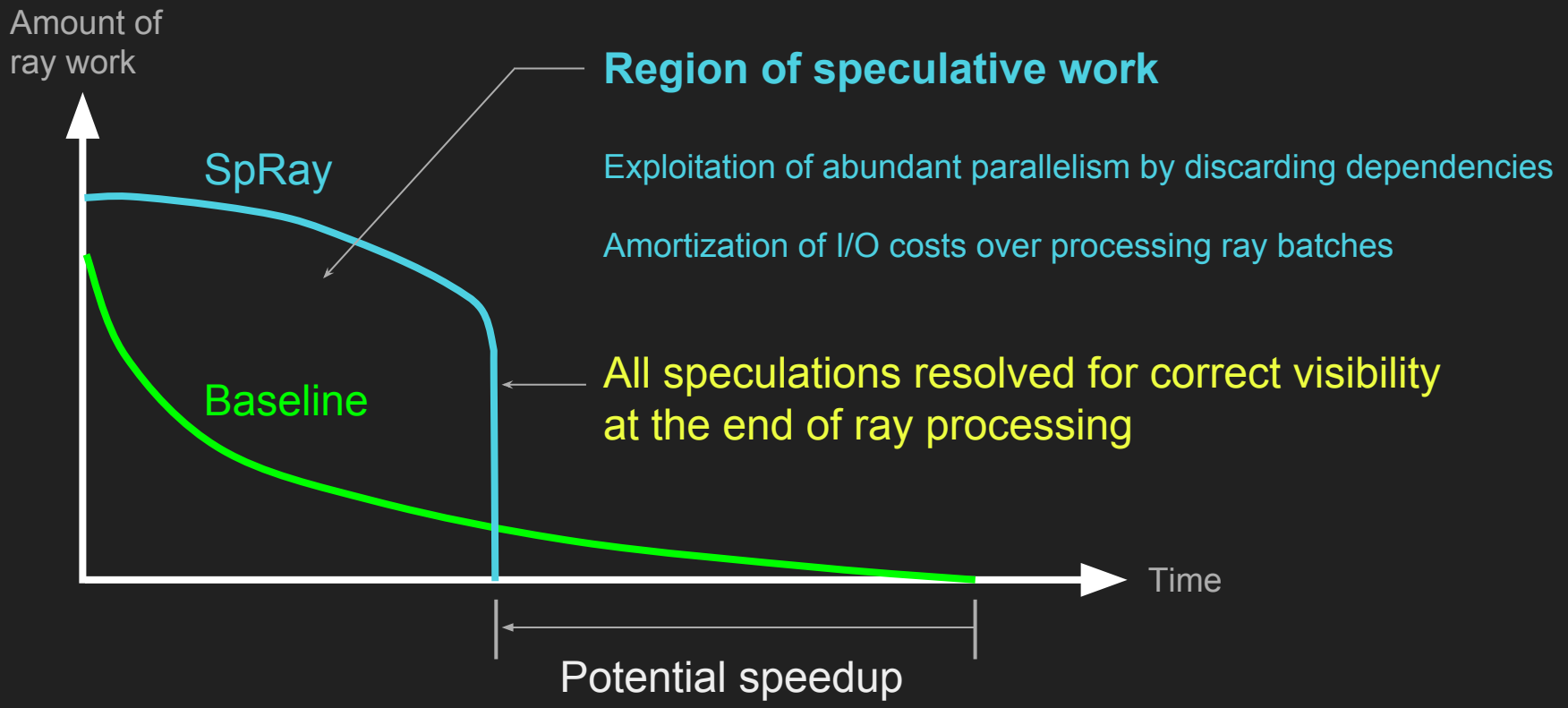
Only one node is active at a time

Amount of work is drastically reduced over time



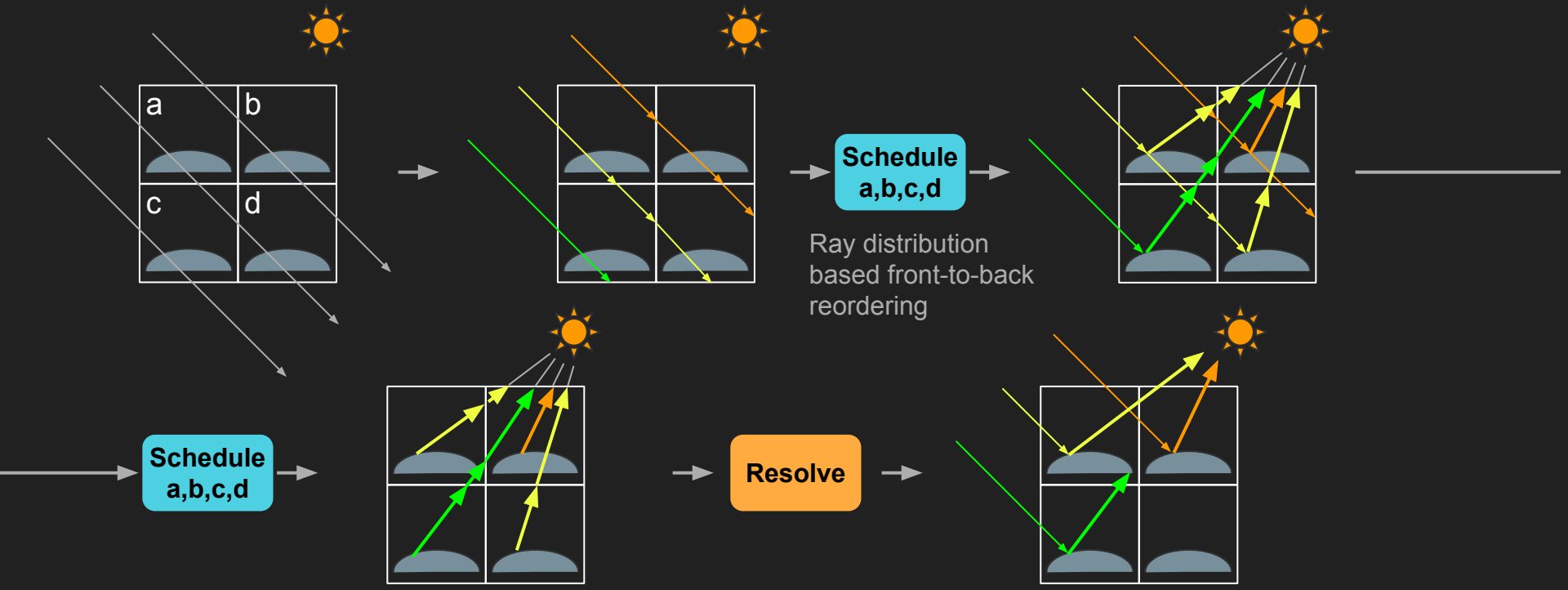
SpRay

Speculate first and resolve later



Amortize loading costs over ray processing

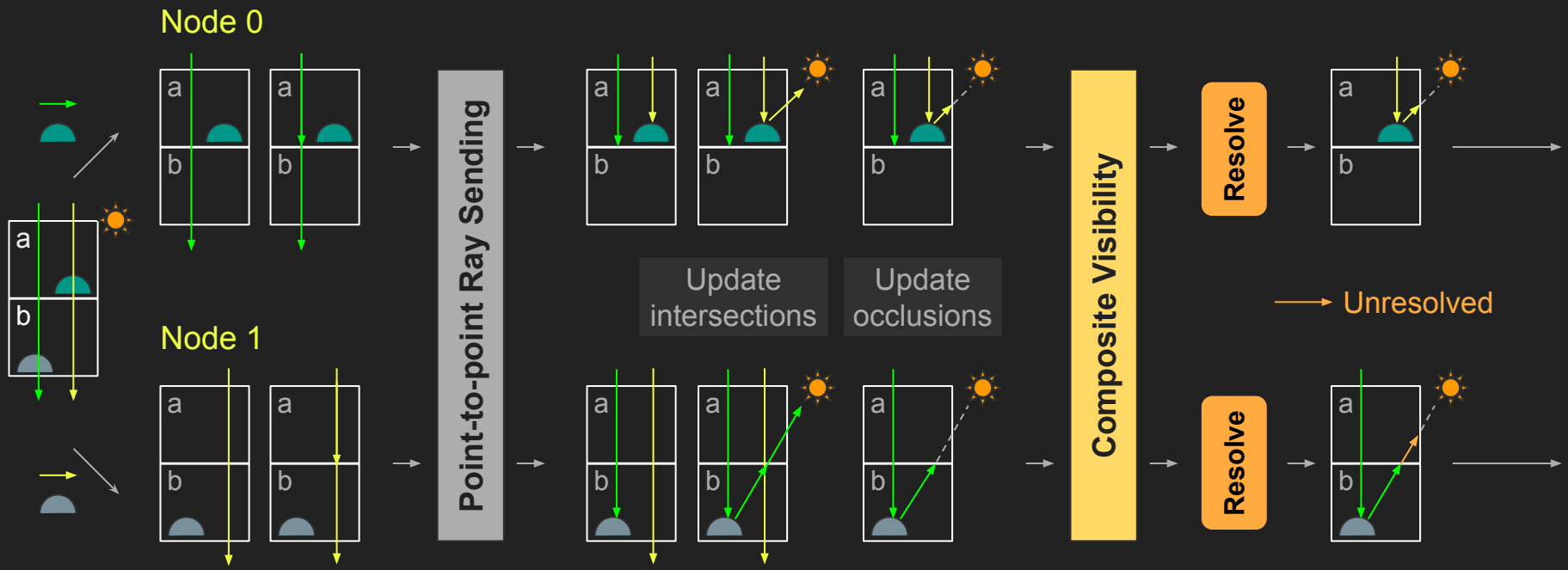
SpRay assumes that **all domains are equally likely to have the first hit point** for a given ray, speculatively performing ray placement, ray-primitive tests, and creation of new rays



We need only **up to 2 loads per domain for one bounce worth of speculative rays**

Speculate intersection points and occlusions

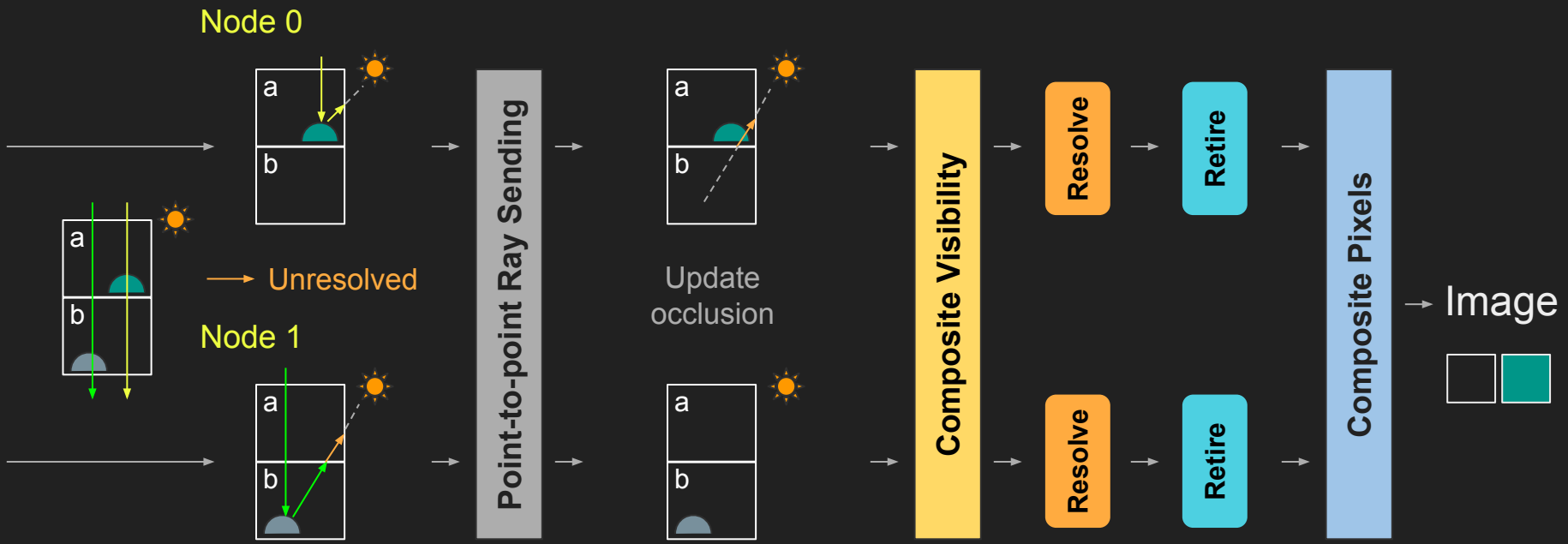
Use case: 2 nodes, each owning a domain



Perform another round to resolve speculative shadow rays

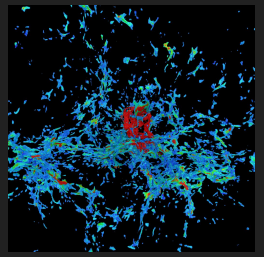
Resolve speculative shadows and retire pixels

Use case: 2 nodes, each owning a domain

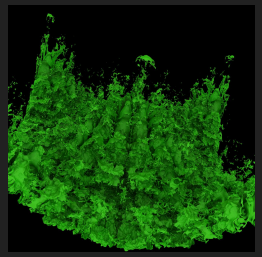


Evaluation

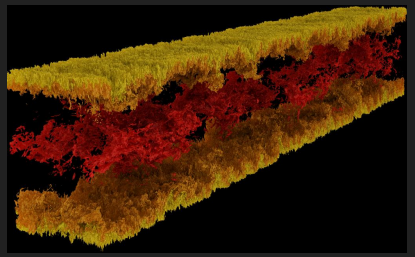
Experimental setup



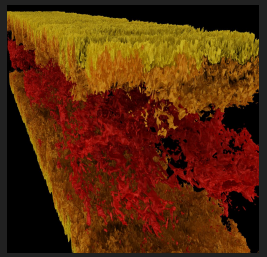
Enzo
940 GB raw data
16 domains
6 M triangles



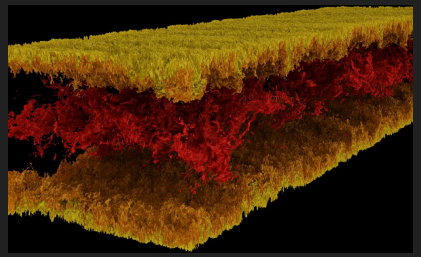
RM
8 GB raw data
52 domains
108 M triangles



DNS1-side
483 GB raw data
720 domains
0.9 B triangles



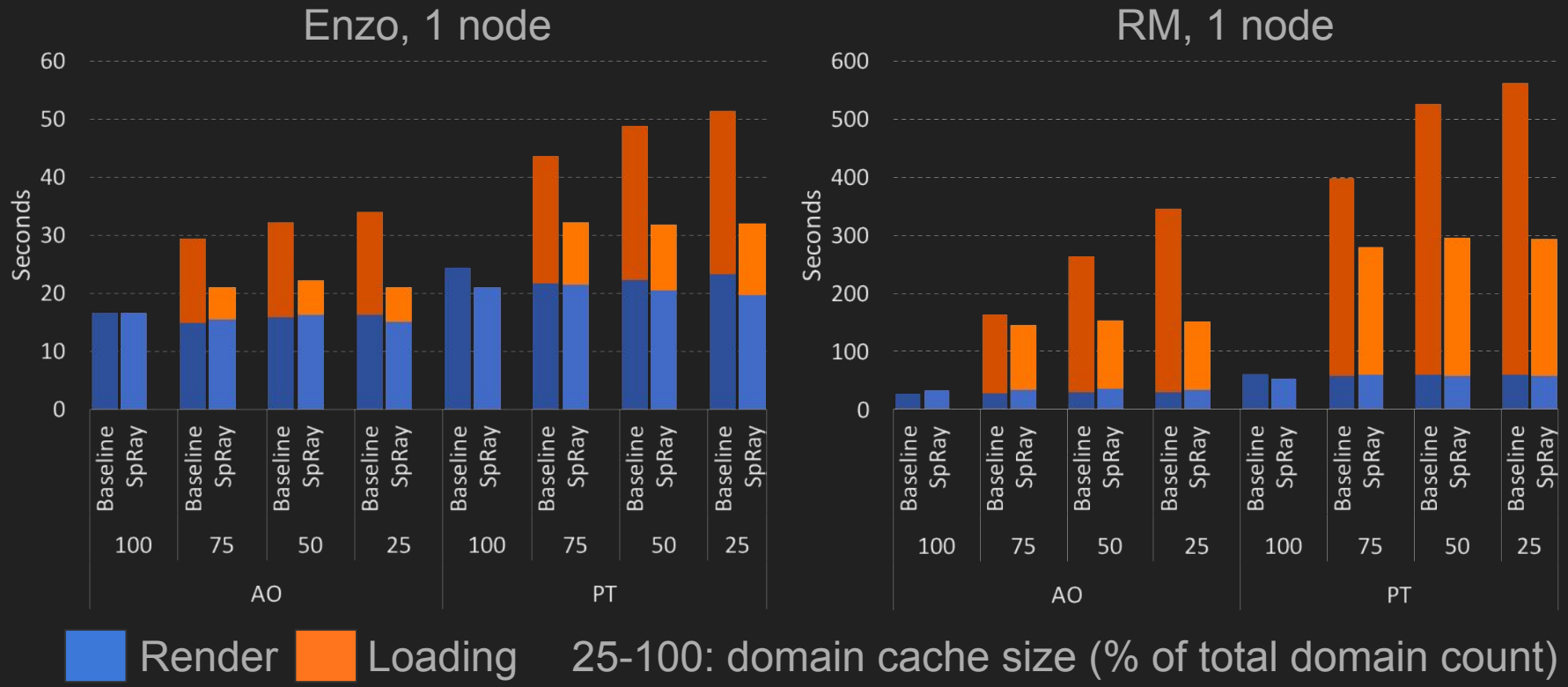
DNS1-back
483 GB raw data
720 domains
0.9 B triangles



DNS2
483 GB raw data
1427 domains
1.8 B triangles

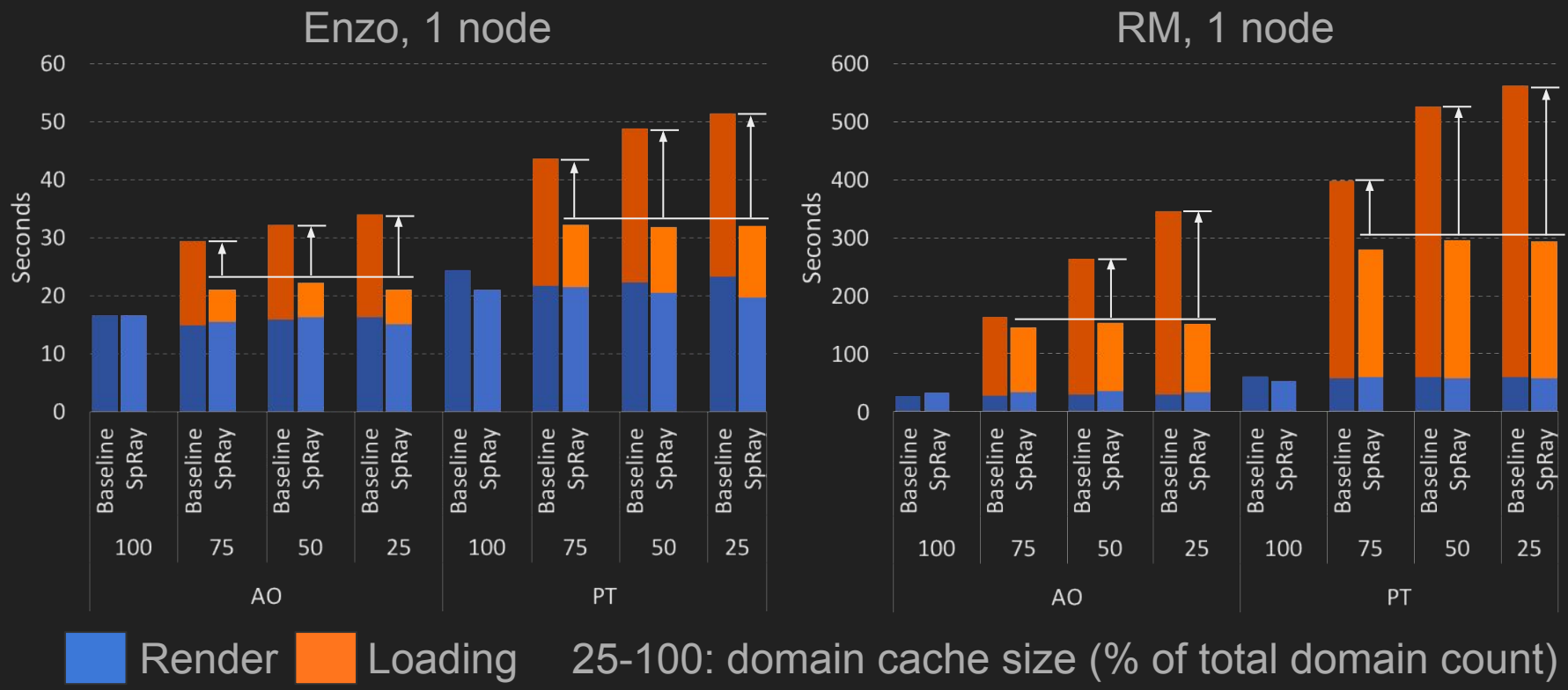
- Compared SpRay with Baseline for in situ and out-of-core use cases
- Limited the domain cache size to emulate out-of-core rendering
- 1-bounce ambient occlusion (AO) and 3-bounce path tracing (PT)
- 32 camera rays per pixel, 64 shadow rays per hit point, 1 Megapixel image

Out-of-core: up to 2.3x speedup with fewer loads



Up to 3.2x for the loading time

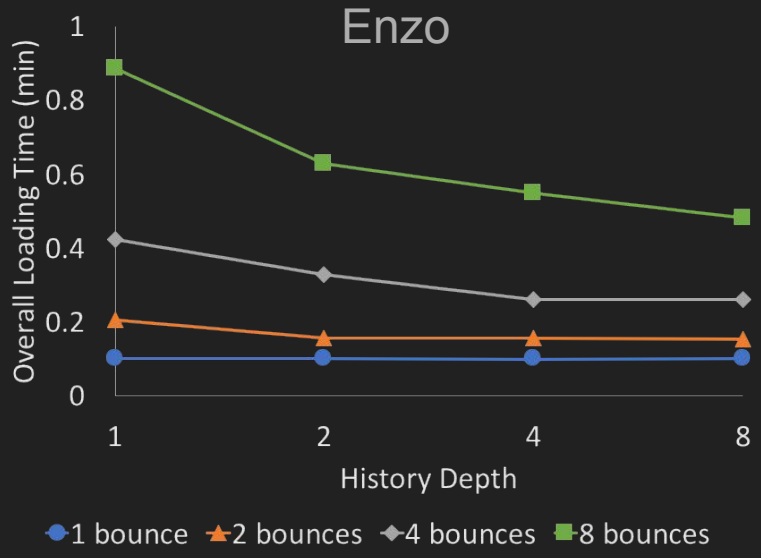
Out-of-core: effective with larger data

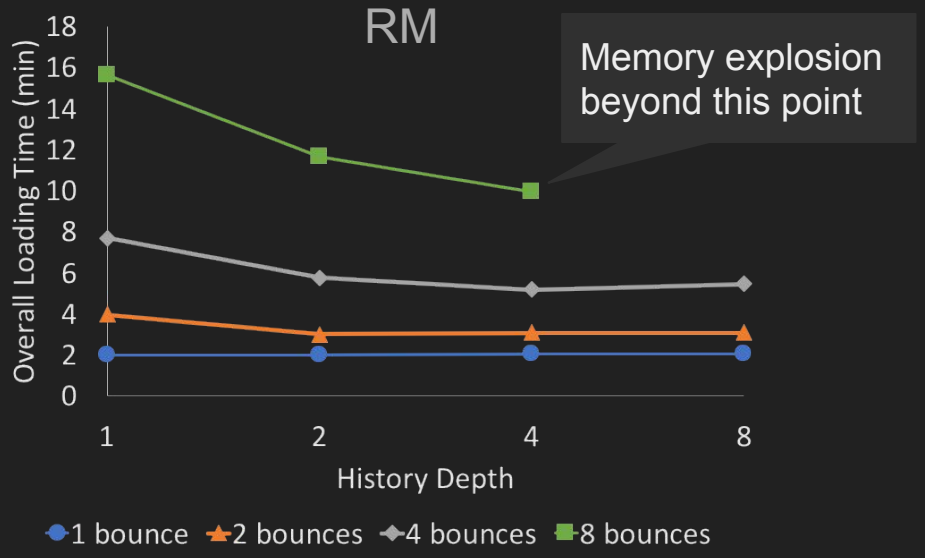


Loading costs become more amortized as memory pressure grows

Out-of-core: performance vs. speculation level

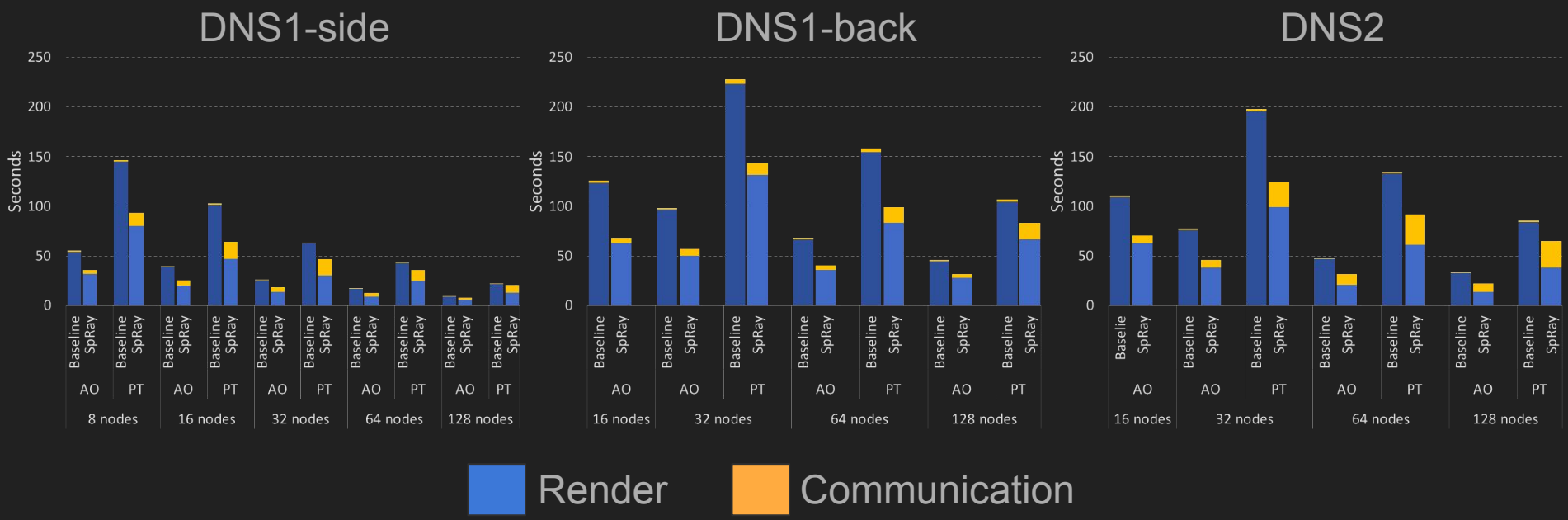
Path tracing, 1 node, 50% domain cache sizes





Overall loading time reduced as the speculation level (history depth) grows

In situ: 1.9x speedup overall with comm. overhead



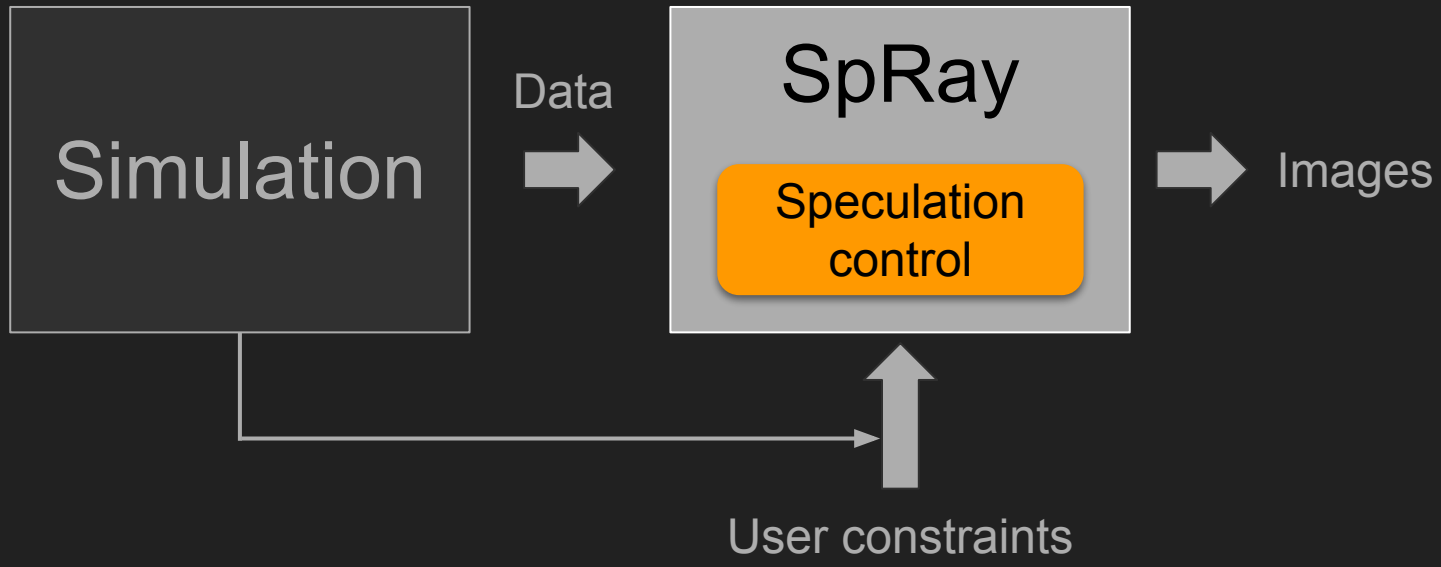
We could further achieve up to 2.3x speedup by overlapping communication

Summary

SpRay: a system for speculative ray scheduling

- Introduced the speculate-first-and-resolve-later concept for rendering large data subdivided into domains
- Showed SpRay outperforming typical methods for ray scheduling in both out-of-core and in situ rendering scenarios on a supercomputing environment

Vision: a system with controlled speculation



Thank you

https://github.com/TACC/SpRay

Acknowledgments

National Science Foundation grant ACI-1339863
Intel Visualization Center of Excellence award through the IPCC program
Nick Malaya and Bob Moser at UT Austin for providing the DNS data

References

[Pharr 1997]

Rendering Complex Scenes with Memory-Coherent Ray Tracing
Matt Pharr, Craig Kolb, Reid Gershbein, Pat Hanrahan, SIGGRAPH 1997

[Navrátil 2013]

Exploring the Spectrum of Dynamic Scheduling Algorithms for Scalable Distributed-Memory Ray Tracing
Paul A. Navrátil, Hank Childs, Donald S. Fussell, Calvin Lin, IEEE TVCG 2013

[Son 2017]

Timeline Scheduling for Out-of-Core Ray Batching
Myungbae Son and Sung-Eui Yoon, HPG 2017