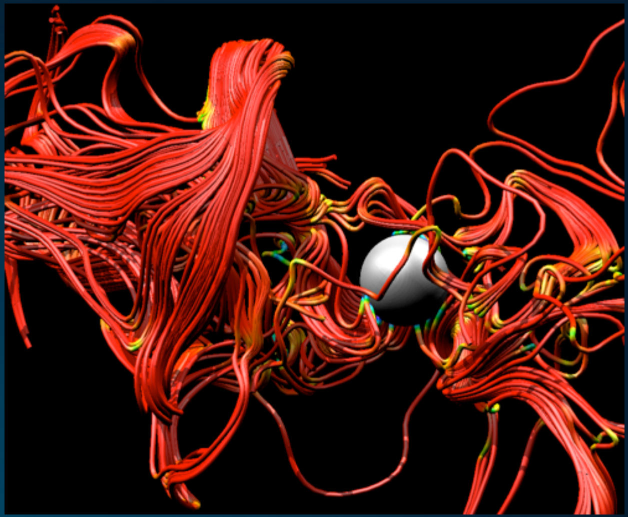
VisWeek 2011
VIS • INFOVIS • VAST



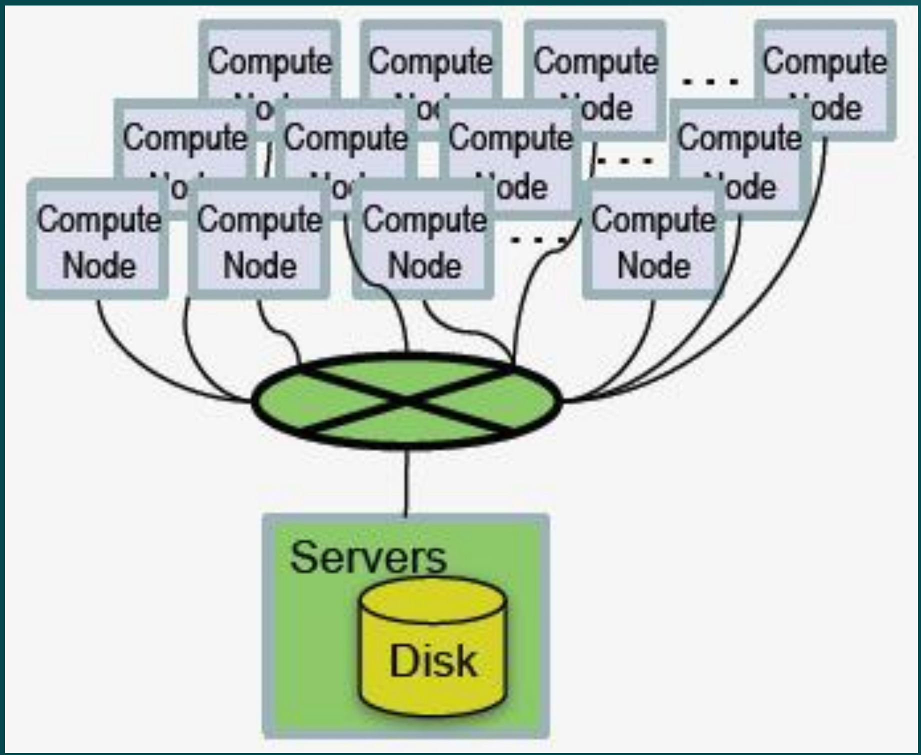
Evaluating the Benefits of An Extended Memory Hierarchy for Parallel Streamline Algorithms

David Camp (LBL, UC Davis), Hank Childs (LBL, UC Davis), Amit Chourasia (SDSC), Christoph Garth (UC Davis),
& Kenneth I. Joy (UC Davis)

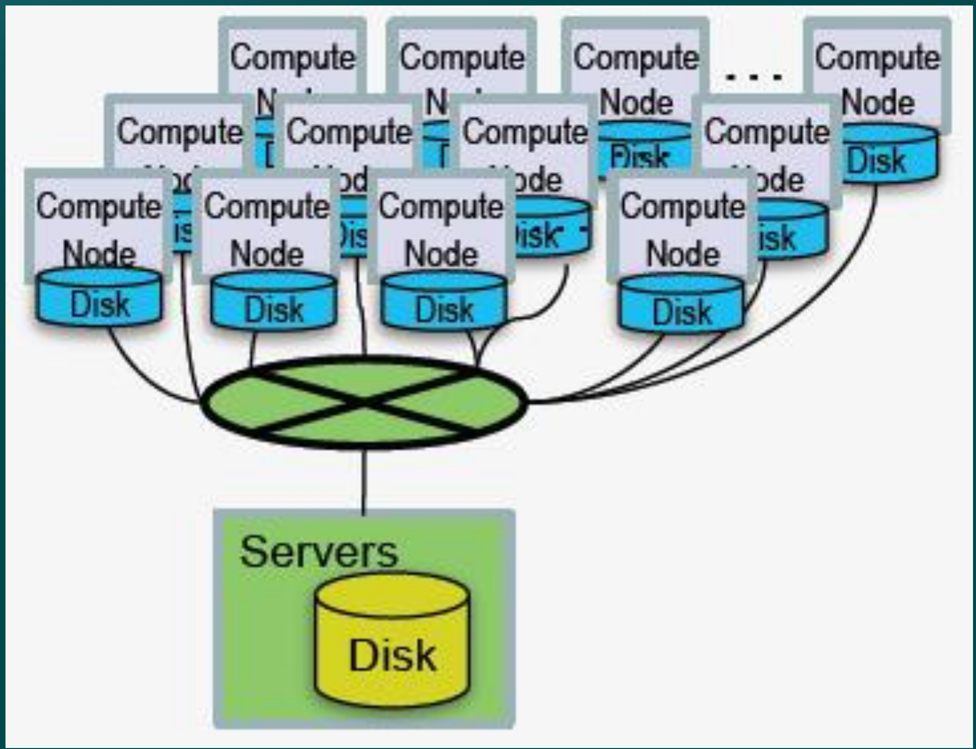
Outline

- New emerging I/O configuration for supercomputers
  - Can we use this new configuration to improve streamline performance.
- Streamline Algorithm
  - Parallelize-over-Seeds
- Results
  - Up to three times faster
  - Model that can determine benefit

Emerging I/O Configuration

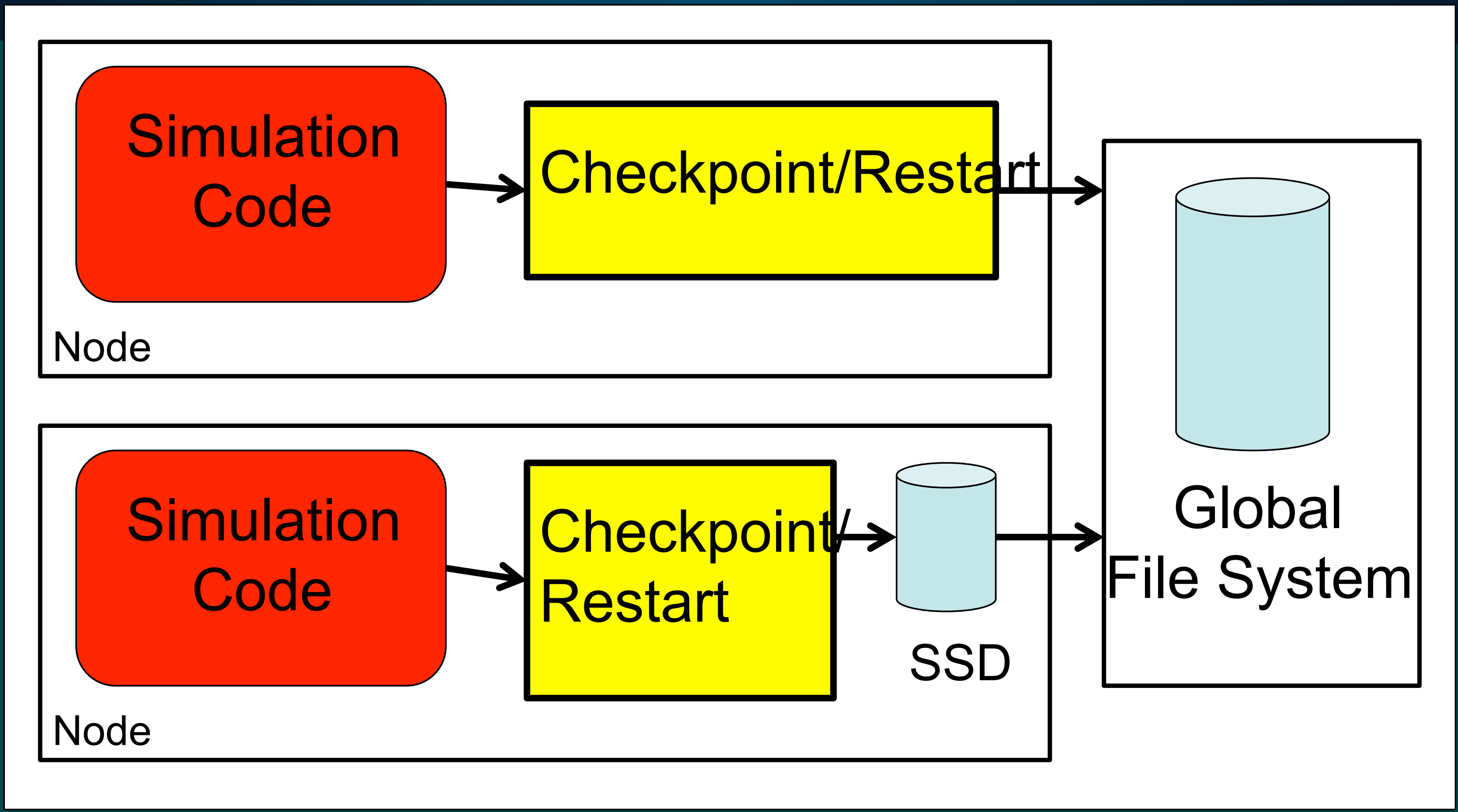


Common I/O Configuration



Emerging I/O Configuration

Simulation Codes

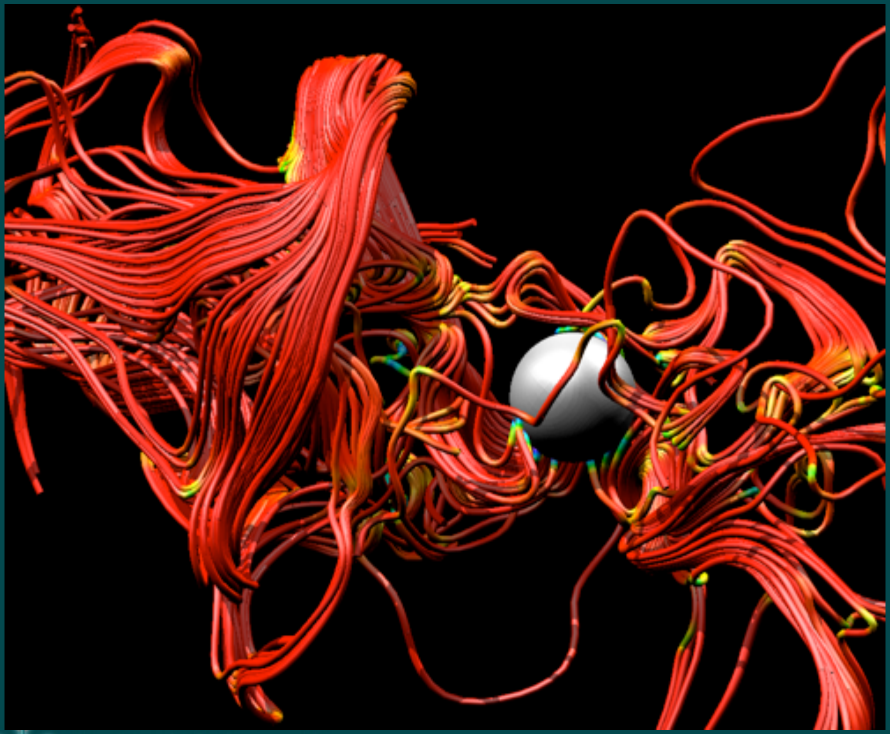


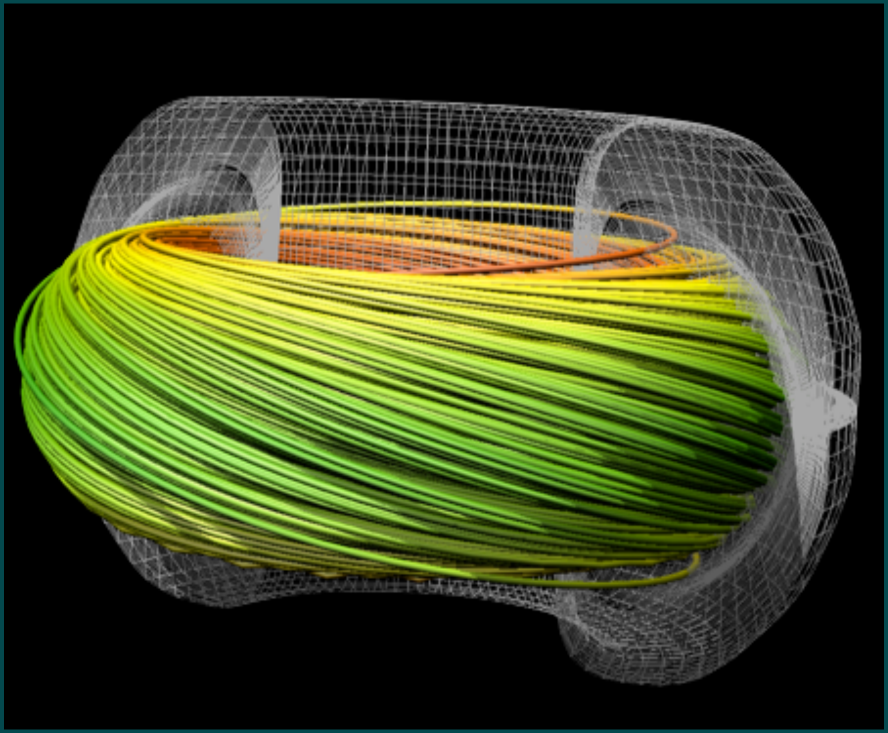
Research Questions

- New design is primarily made for data producers (those who store data), not data consumers (those who read data).
- But can we use the SSDs to benefit data consumers?
  - IDEA: use the SSDs to extend the memory hierarchy and reduce redundant reads. In effect increases local memory size.
- Research questions:
  - Can the extended memory hierarchy improve performance?
  - To what extent?

What Type of Vis Algorithm Can Benefit From an Extended Memory Hierarchy?

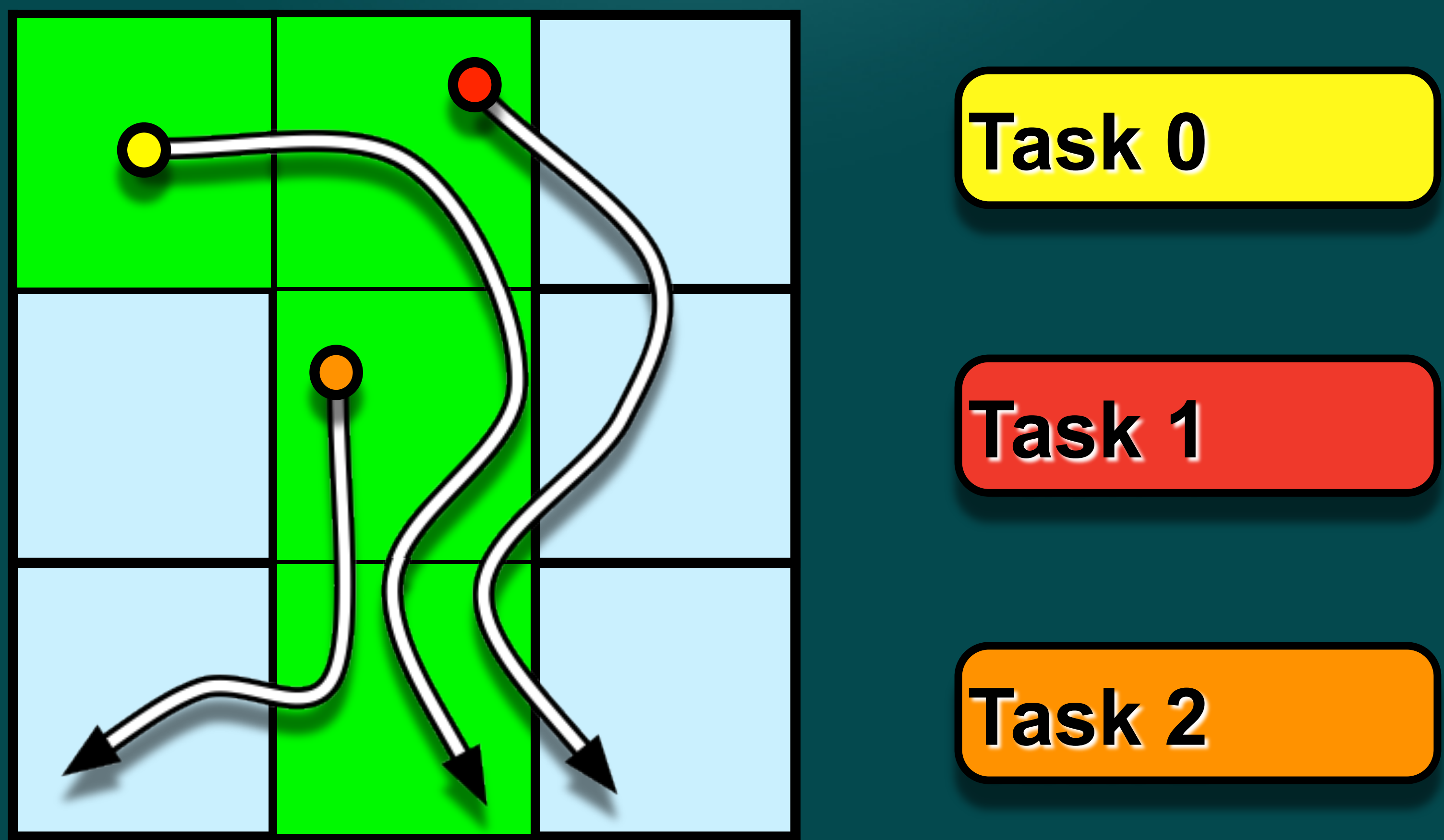
- Answer: Algorithms that repeatedly read data
  - Streamline algorithm - Parallelize-over-Seeds



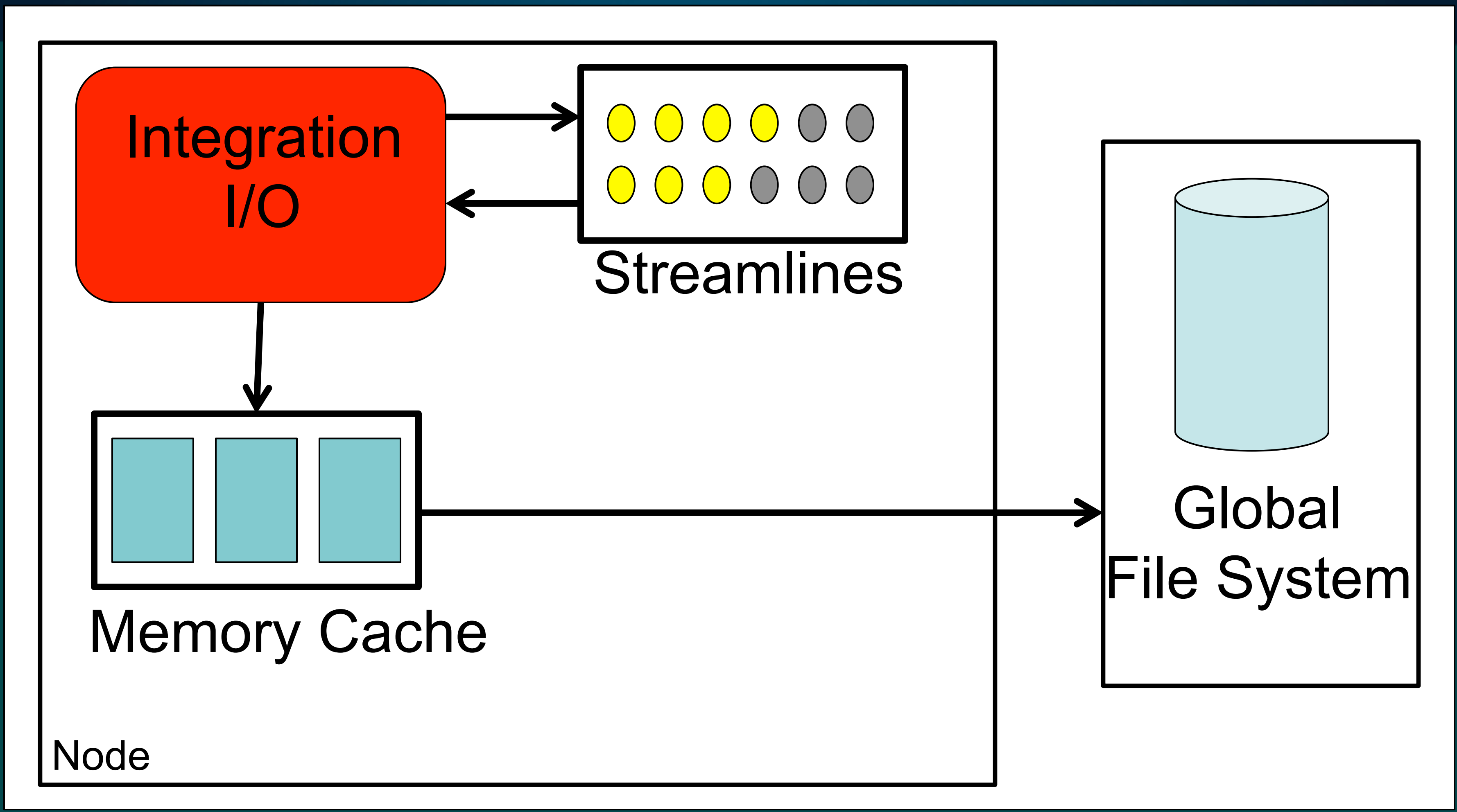


Streamline Algorithm - Parallelize-over-Seeds

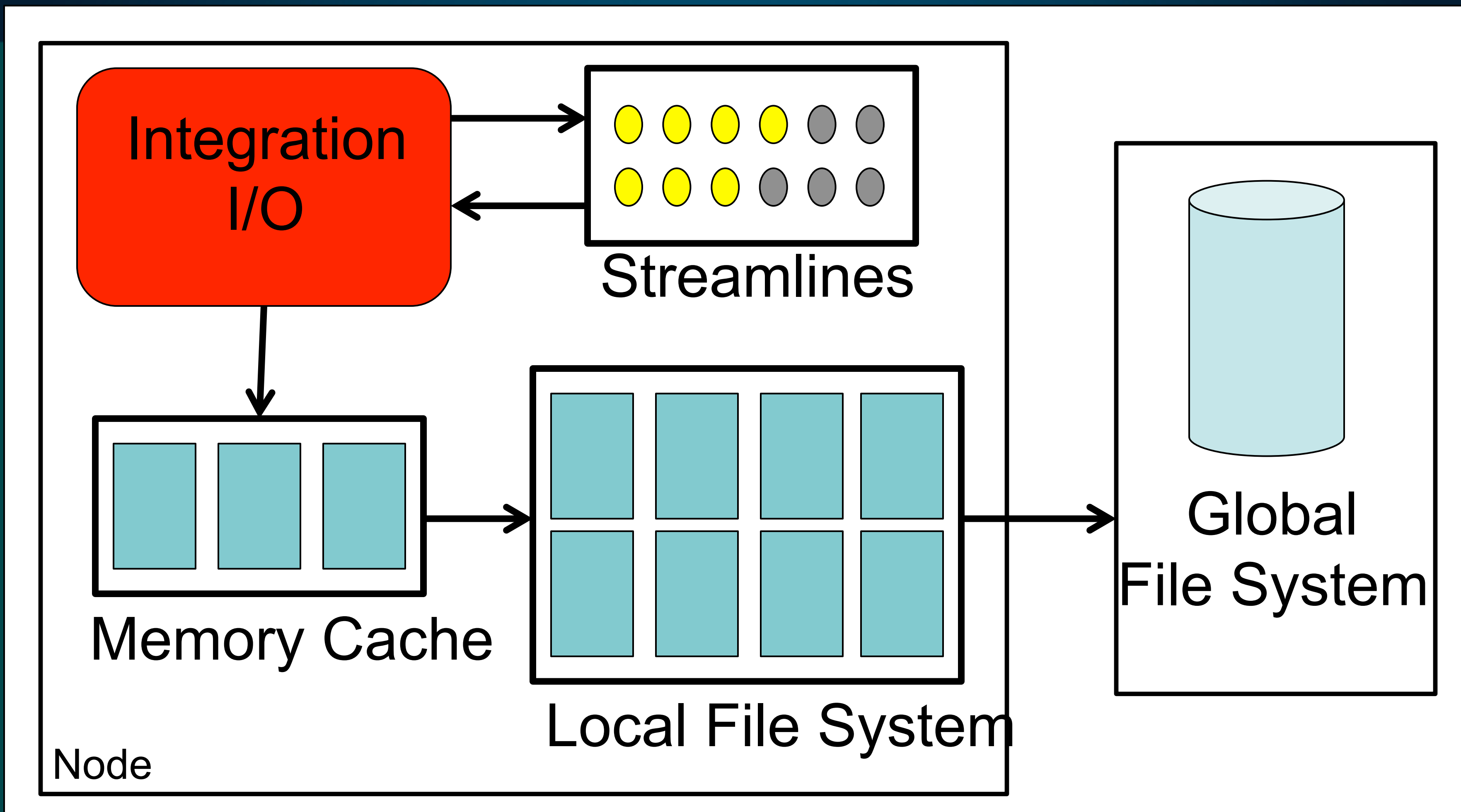
- Example: 2d data set with 9 data blocks and 3 seed points partitioned over 3 processors



Parallelize-over-Seeds Algorithm

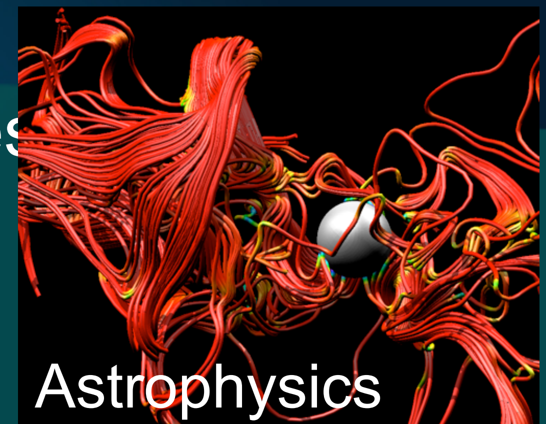


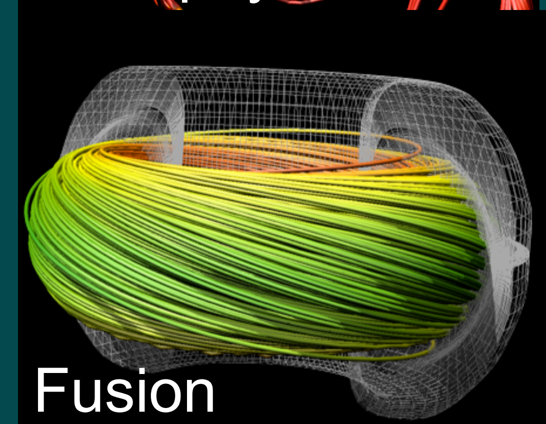
Parallelize-over-Seeds Algorithm with the Extended Memory Hierarchy

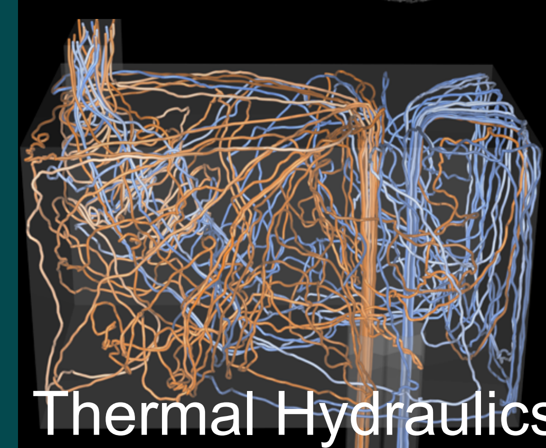


We designed experiments to illuminate the benefits of an extended memory hierarchy for streamlines.

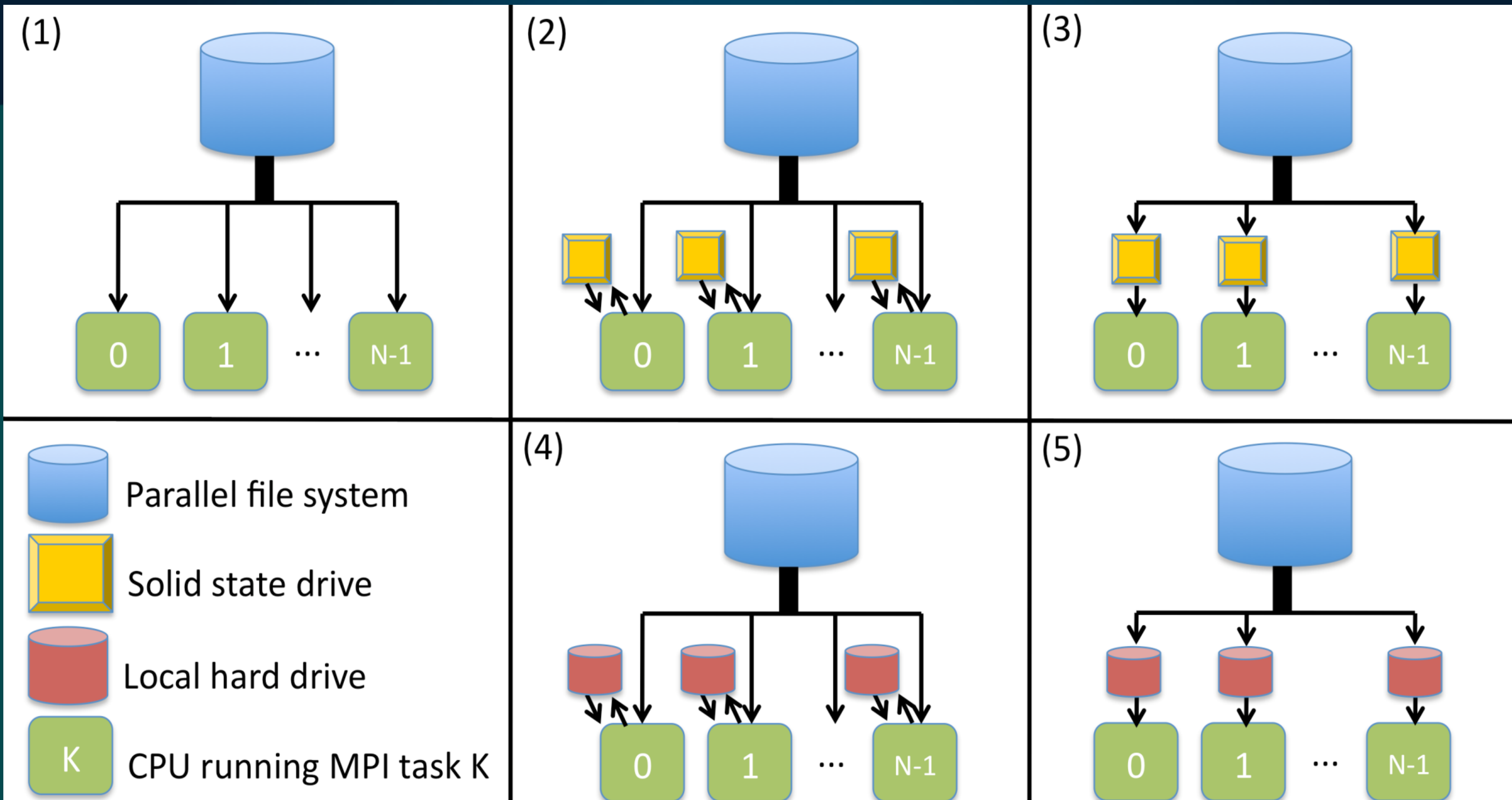
- Goal was to cover complexities of streamlines
  - Seed set size
  - Seed set distribution
  - Data set size
  - Vector field complexity
- 30 experiments varied over:
  - Five different I/O configurations
  - Three data sets
  - Two seed set sizes (2,500 and 10,000)







I/O Configurations



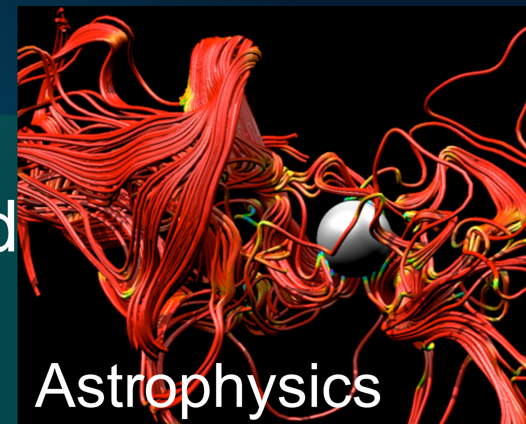
Runtime Environment

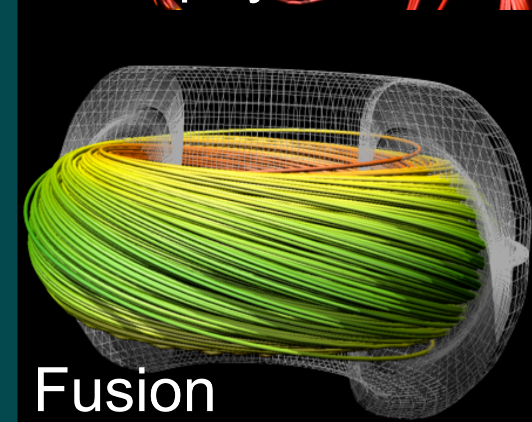
- Dash is a prototype system at San Diego Supercomputer Center
  - Each node has
    - Two quad Intel processors
    - GPFS, local SSD and local hard drive
- VisIt
  - End user visualization and analysis tool for large data sets
- Benchmarks were performed during production use
  - Each benchmark run was performed using 64 cores (8 nodes)

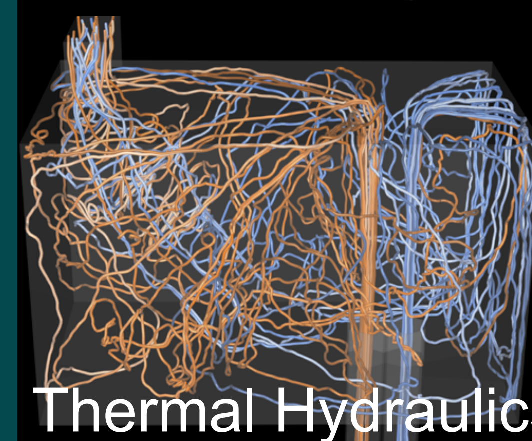
Measurements

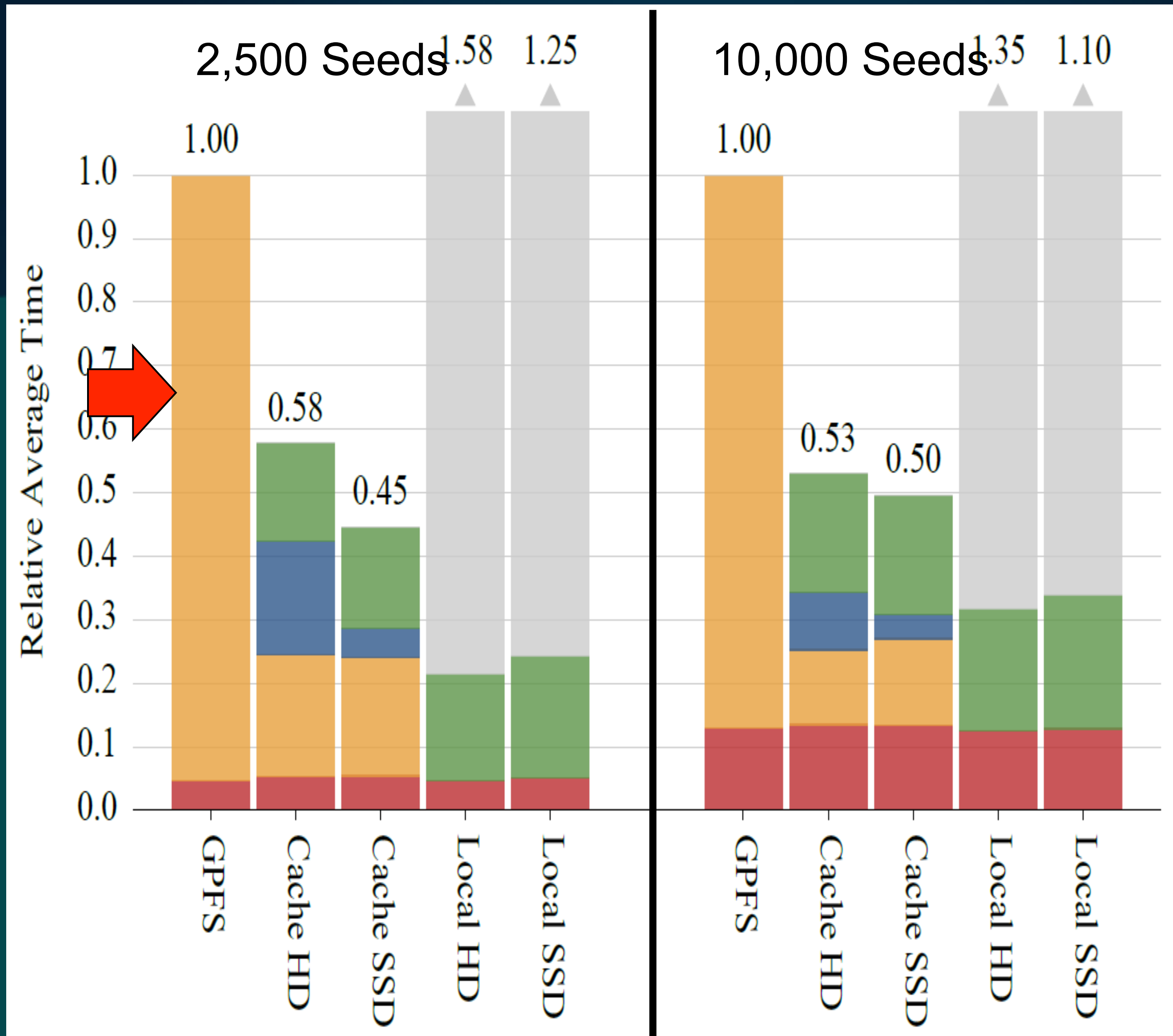
- Time to calculate streamlines
  - Total time = integration time + Data load
  - Integration time
  - I/O time
    - Data retrieval and storage time

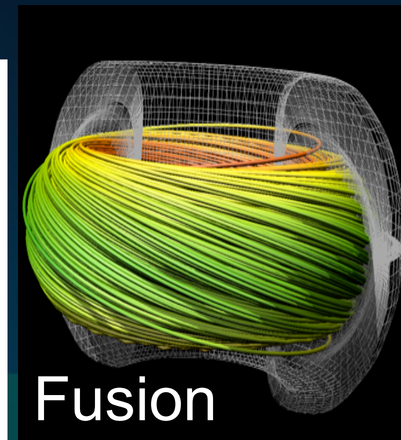
- For our 30 tests…
  - Three different data sets
  - Five I/O configurations
  - Two seeding sizes (2,500 and 10,000)











GPFS fetches

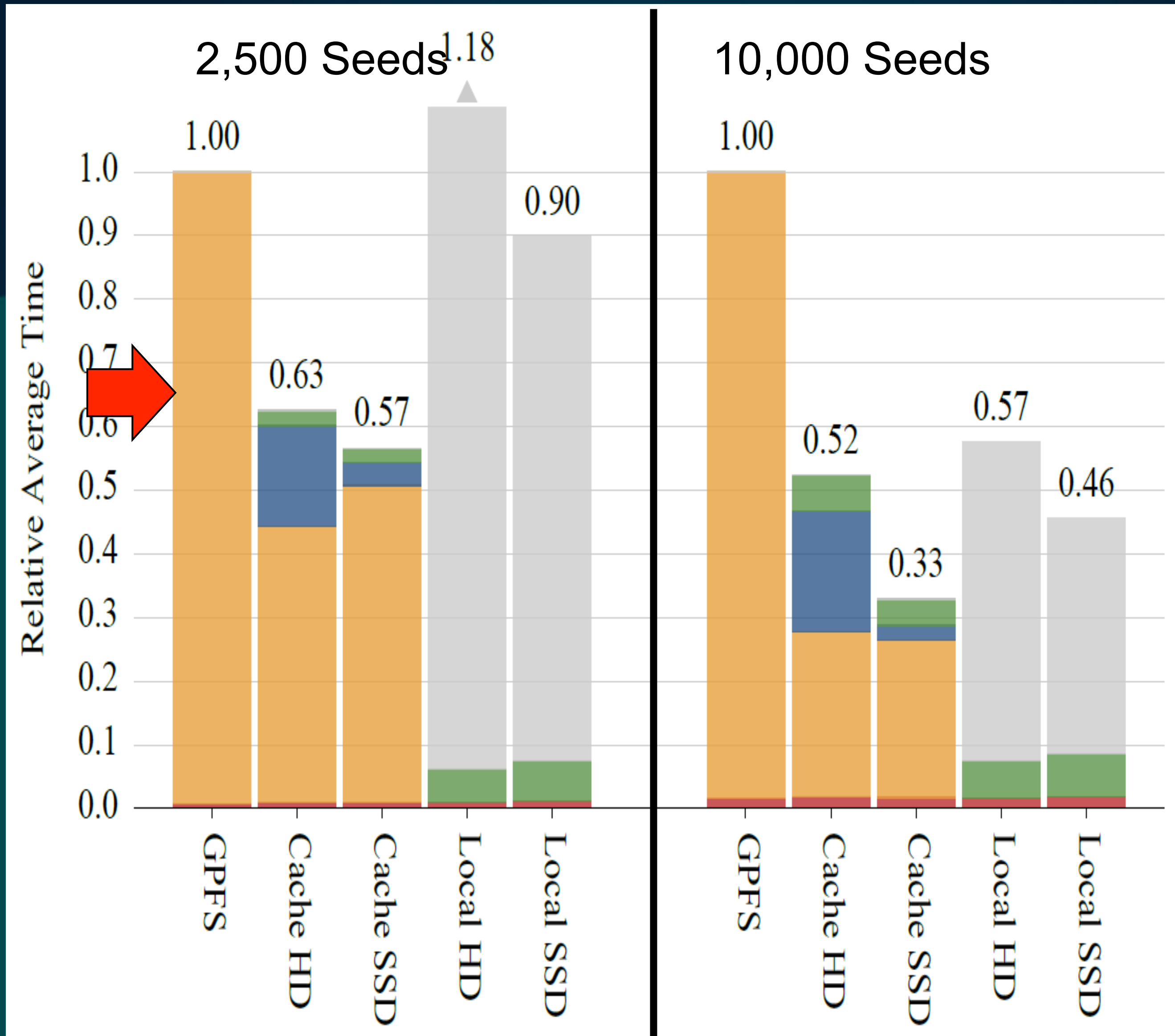
Integration time

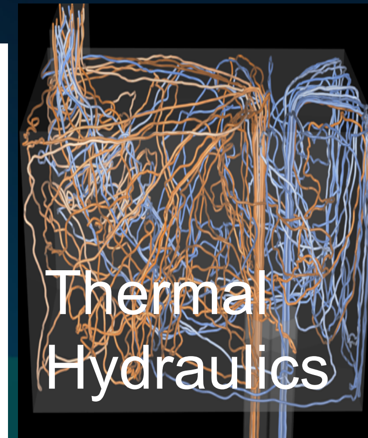
Local disk Stores

Local disk Fetches

Local copy time

Conf.	N_{GF}	N_{LF}	N_{LS}	% Accel	Conf.	N_{GF}	N_{LF}	N_{LS}	% Accel





GPFS fetches

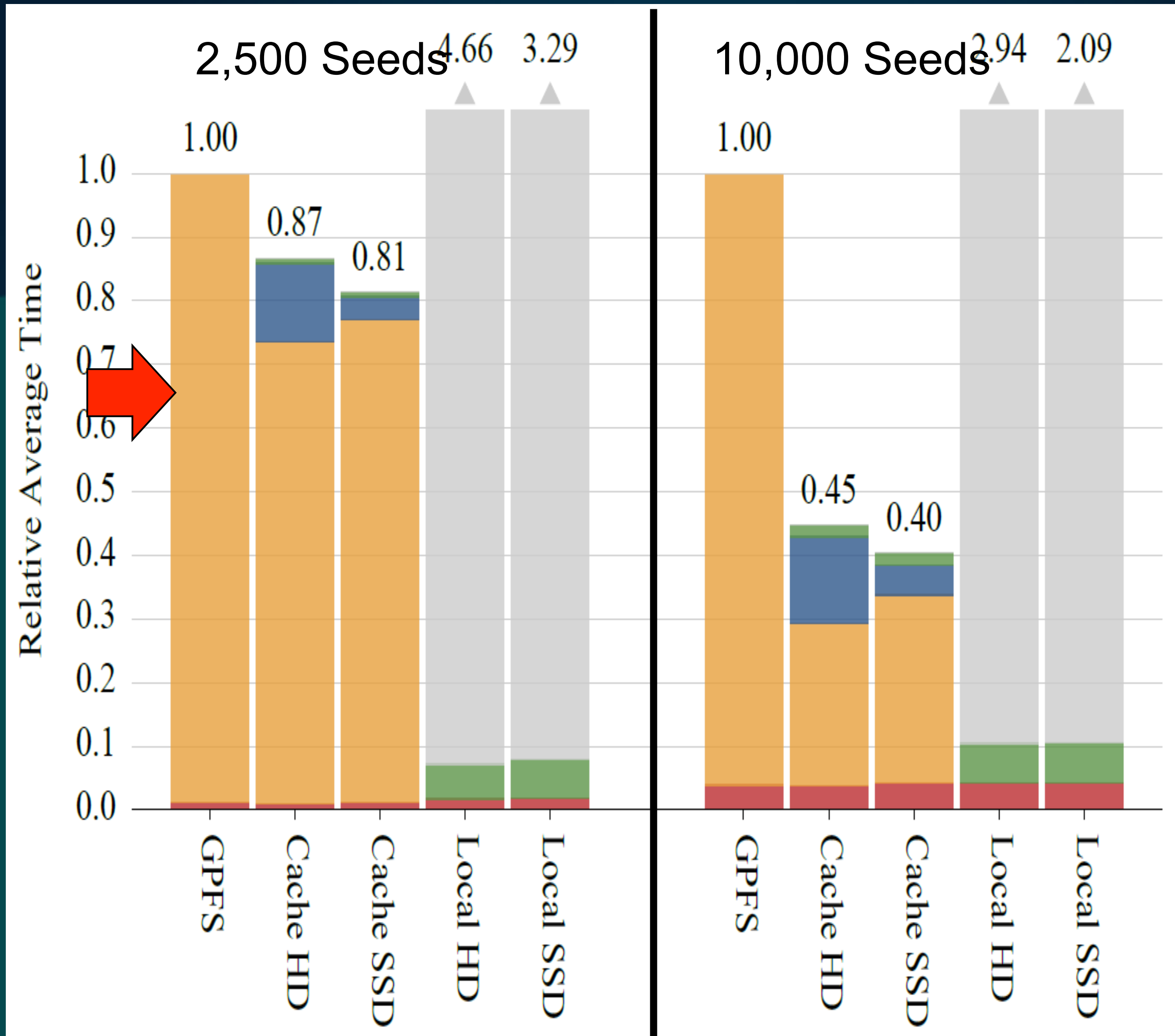
Integration time

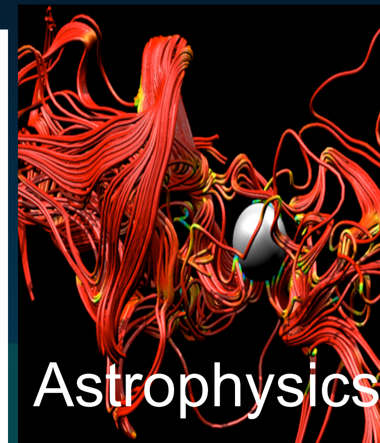
Local disk Stores

Local disk Fetches

Local copy time

Conf.	N_{GF}	N_{LF}	N_{LS}	% Accel	Conf.	N_{GF}	N_{LF}	N_{LS}	% Accel





GPFS fetches

Integration time

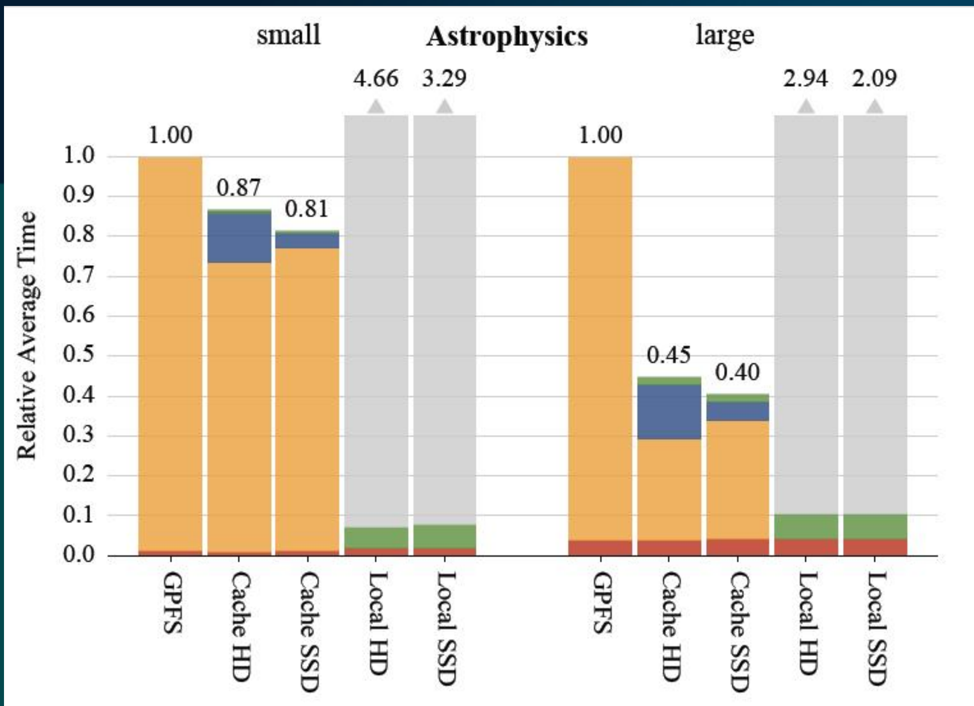
Local disk Stores

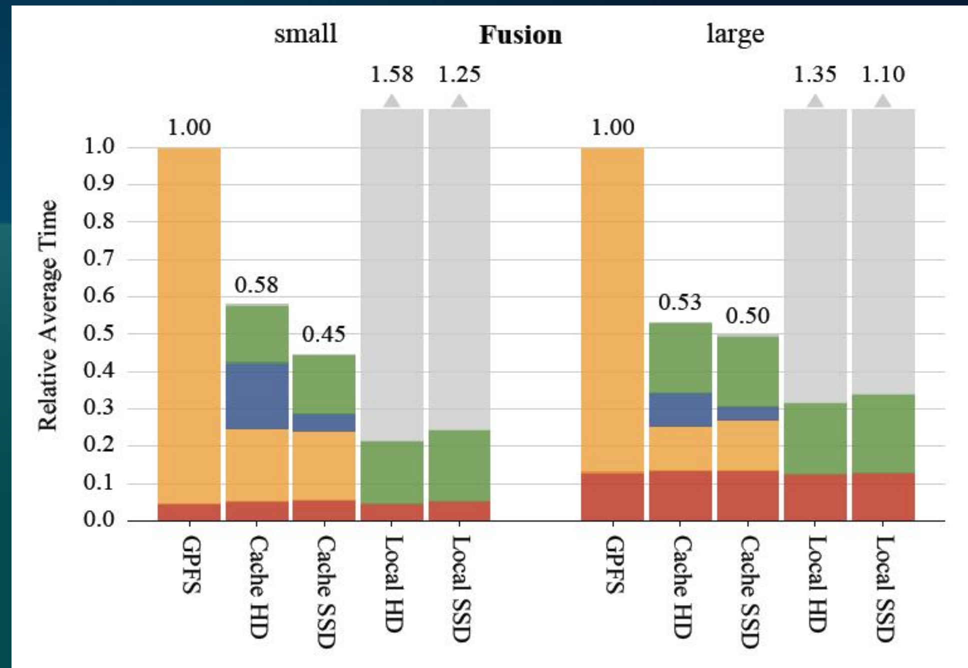
Local disk Fetches

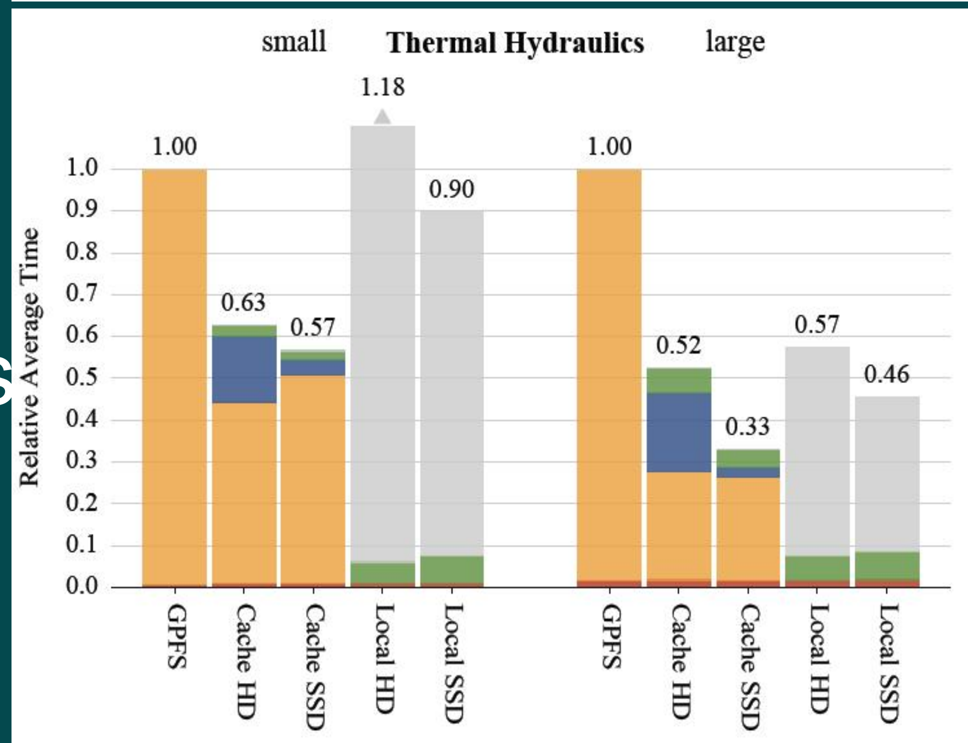
Local copy time

Conf.	N_{GF}	N_{LF}	N_{LS}	% Accel	Conf.	N_{GF}	N_{LF}	N_{LS}	% Accel

Results







Cache SSD won all tests

		I/O Conf.	T_{Total}	T_{Total*}	T_{IO}	N_{GF}	T_{GF}	N_{LF}	T_{LF}	N_{LS}	T_{LS}	$\%_{Accel}$
Astro	small	GPFS	25.17s	25.17s	24.89s (98%)	5275	24.89s	–	–	–	–	0%
		Cache HD	21.81s	21.81s	21.60s (99%)	4456	18.31s	819	0.20s	3275	3.10s	16%
		Cache SSD	20.47s	20.47s	20.20s (98%)	4456	19.14s	819	0.18s	3275	0.88s	16%
		Local HD	117.37s	1.80s	1.37s (76%)	–	–	5275	1.37s	–	–	100%
		Local SSD	82.85s	1.97s	1.53s (77%)	–	–	5275	1.53s	–	–	100%
	large	GPFS	40.80s	49.80s	39.21s (96%)	9994	39.21s	–	–	–	–	0%
		Cache HD	18.26s	18.26s	16.71s (91%)	7059	10.40s	2935	0.77s	5890	5.55s	29%
		Cache SSD	16.48s	16.48s	14.75s (89%)	7059	12.03s	2935	0.78s	5890	1.95s	29%
		Local HD	119.83s	4.26s	2.54s (59%)	–	–	9994	2.54s	–	–	100%
		Local SSD	85.16s	4.28s	2.57s (59%)	–	–	9994	2.57s	–	–	100%
Fusion	small	GPFS	80.90s	80.90s	77.20s (95%)	51251	77.20s	–	–	–	–	0%
		Cache HD	46.79s	46.79s	42.48s (90%)	8642	15.55s	42609	12.55s	8453	14.38s	83%
		Cache SSD	36.07s	36.07s	31.68s (87%)	8642	15.05s	42609	12.85s	8453	3.78s	83%
		Local HD	128.07s	17.31s	13.61s (78%)	–	–	51251	13.61s	–	–	100%
		Local SSD	101.13s	19.68s	15.53s (78%)	–	–	51251	15.53s	–	–	100%
	large	GPFS	107.04s	107.04s	93.16s (87%)	79467	93.16s	–	–	–	–	0%
		Cache HD	56.79s	56.79s	42.39s (74%)	9907	12.59s	69560	20.16s	9714	9.64s	87%
		Cache SSD	53.07s	53.07s	38.78s (73%)	9907	14.57s	69560	20.14s	9714	4.07s	87%
		Local HD	144.65s	33.89s	20.60s (60%)	–	–	79467	20.60s	–	–	100%
		Local SSD	117.75s	36.30s	22.62s (62%)	–	–	79467	22.62s	–	–	100%
Thermal Hydraulics	small	GPFS	98.88s	98.88s	98.22s (99%)	19085	98.22s	–	–	–	–	0%
		Cache HD	61.81s	61.81s	60.85s (98%)	10278	42.88s	8807	2.39s	9888	15.58s	46%
		Cache SSD	55.92s	55.92s	54.95s (98%)	10278	49.20s	8807	2.16s	9888	3.59s	46%
		Local HD	116.48s	6.04s	5.13s (84%)	–	–	19085	5.13s	–	–	100%
		Local SSD	88.85s	7.38s	6.29s (85%)	–	–	19085	6.29s	–	–	100%
	large	GPFS	220.68s	220.68s	217.23s (98%)	48188	217.23s	–	–	–	–	0%
		Cache HD	115.39s	115.39s	111.47s (96%)	14099	57.38s	34089	12.54s	13717	41.54s	71%
		Cache SSD	72.56s	72.56s	68.80s (94%)	14099	54.56s	34089	9.13s	13717	5.10s	71%
		Local HD	126.84s	16.40s	12.93s (78%)	–	–	48188	12.93s	–	–	100%
		Local SSD	100.44s	18.97s	15.14s (79%)	–	–	48188	15.14s	–	–	100%

Model To Determine How Much The Extended Memory Hierarchy Can Benefit

- Baseline (GPFS)
  - Total Time = (Integration Time) + (Number of Blocks) * (Time to Fetch Data)
  - $T_{total} = T_{Int} + N_{GF} * A_{GF}$
- Extended Memory Hierarchy
  - $T_{total} = T_{int} + N_{\underline{A}} * T_{GF} + N_{\underline{A}} * T_{LS} + N_{A} * T_{LF}$

VisWeek 2011 VIS • INFOVIS • VAST

Model To Determine How Much The Extended Memory Hierarchy Can Benefit

- Baseline (GPFS)
  - Total Time = (Integration Time) + (Number of Blocks) * (Time to Fetch Data)
  - $T_{total} = T_{Int} + N_{GF} * A_{GF}$

- Extended Memory Hierarchy
  - $T_{total} = T_{int} + N_{\underline{A}} * T_{GF} + N_{\underline{A}} * T_{LS} + N_{A} * T_{LF}$

(Number of Data Blocks) * (Time to Fetch Data from Global File S

Model To Determine How Much The Extended Memory Hierarchy Can Benefit

- Baseline (GPFS)
  - Total Time = (Integration Time) + (Number of Blocks) * (Time to Fetch Data)
  - $T_{total} = T_{Int} + N_{GF} * A_{GF}$
- Extended Memory Hierarchy
  - $T_{total} = T_{int} + N_{\underline{A}} * T_{GF} + N_{\underline{A}} * T_{LS} + N_{A} * T_{LF}$

(Number of Data Blocks) * (Time to Store Data to Local Dri

Model To Determine How Much The Extended Memory Hierarchy Can Benefit

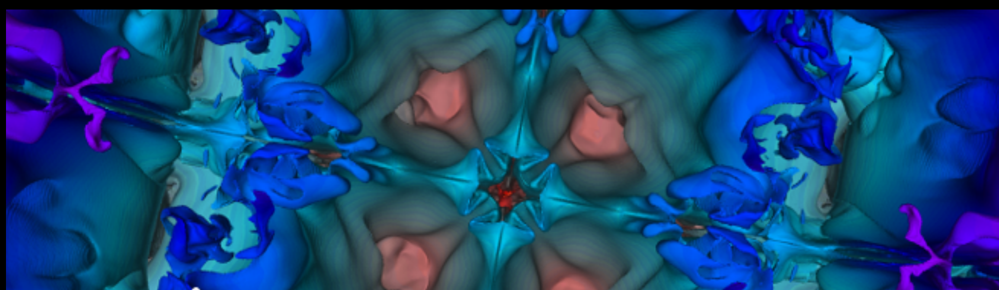
- Baseline (GPFS)
  - Total Time = (Integration Time) + (Number of Blocks) * (Time to Fetch Data)
  - $T_{total} = T_{Int} + N_{GF} * A_{GF}$
- Extended Memory Hierarchy
  - $T_{total} = T_{int} + N_{\underline{A}} * T_{GF} + N_{\underline{A}} * T_{LS} + N_{A} * T_{LF}$

(Number of Reused Data Blocks) * (Time to Load Data from Loca

Summary

- The I/O architecture of supercomputers is evolving.
- We designed and implemented a streamline algorithm that made use of an extended memory hierarchy.
- We found that the performance of a common visualization algorithm (streamline) can be improved by up to a factor of three.

EGPGV 2012 in Cagliari, Italy
Co-located with EuroGraphics 2012

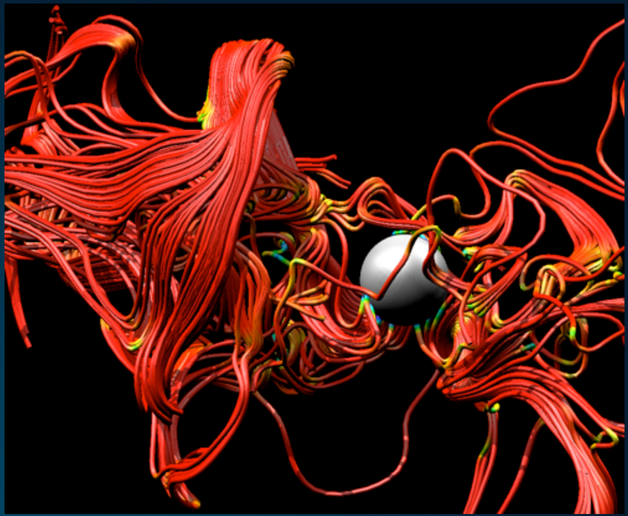


- Dates:
- February, 12, 2011: tentative data for paper submission
- Symposium: May 13-14
- Important:
- Best papers (up to 3) invited for TVCG submissions

Conference Chair:
Fabio Marton,
Visual Computing Group, CRS4, Italy

Papers Co-Chairs:
Torsten Kuhlen
RWTH, Aachen University

Thank you for your attention!!!



Evaluating the Benefits of An Extended Memory Hierarchy for Parallel Streamline Algorithms

David Camp (LBL, UC Davis), Hank Childs (LBL, UC Davis), Amit Chourasia (SDSC), Christoph Garth (UC Davis),

& Kenneth I. Joy (UC Davis)