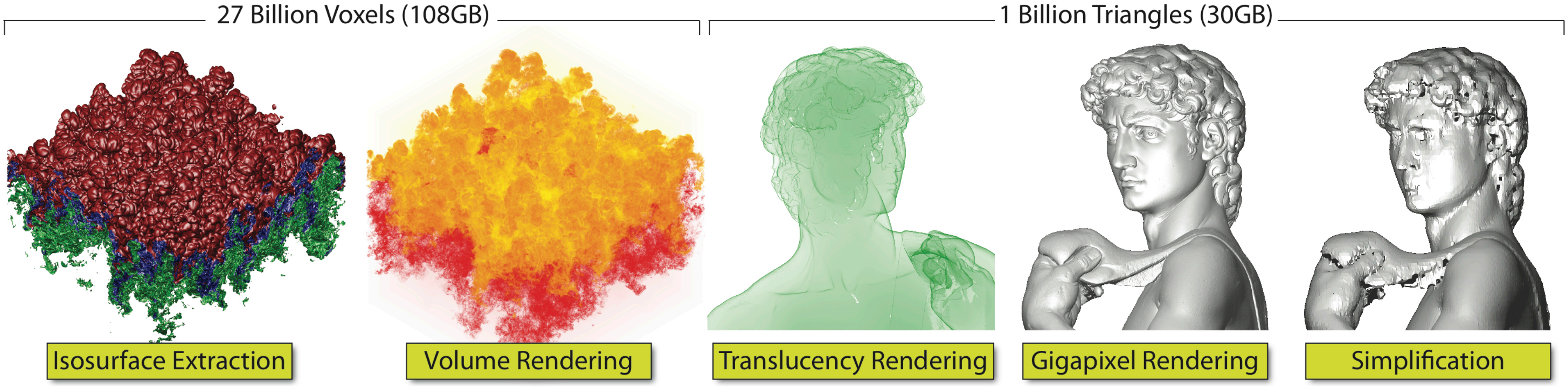


Parallel Visualization on Large Clusters Using MapReduce

Huy T. Vo
Polytechnic Institute of New York University (NYU-Poly)

Jonathan Bronson *(SCI Institute)*, Brian Summa *(SCI Institute)*,
Joao Comba *(UFRGS, Brazil)*, Juliana Freire *(NYU-Poly)*,
Bill Howe *(eScience Institute)*, Valerio Pascucci *(SCI Institute)*,
Claudio T. Silva *(NYU-Poly)*

Why Cloud?

- Multi-tenancy, cost-effective platform
- Simple programming model (MapReduce)
- Scalable computing
- Data-intensive
- Works great in the web and database community

Objectives

- Can we use cloud computing for Vis?
  - Efficiency
  - Scalability
  - LARGE DATA handling

Objectives

- Can we use cloud computing for Vis?
  - Efficiency
  - Scalability
  - LARGE DATA handling

Evaluation with 3 core visualization algorithms

Outline

- MapReduce and Hadoop overview
- Core visualization algorithms in MapReduce
  - Rendering, Isocontouring, Simplification
- Performance results
  - Hadoop baseline
  - Visualization algorithms

What is MapReduce?

- A lightweight parallel framework
  - Two data-parallel phases: Map & Reduce
- Fault-tolerance
- I/O Scheduling

MapReduce Programming Pipeline

INPUT: list of key-value pairs of (k1,v1)

MAP: (k1,v1) ➔ [list of (k2,v2)]

SHUFFLE: combine (k2,v2) ➔ (k2, [list of v2])

REDUCE: (k2, [list of v2]) ➔ [list of v3]

OUTPUT: list of values v3

MapReduce Programming Pipeline

INPUT: list of key-value pairs of (k1,v1)

MAP: (k1,v1) → [list of (k2,v2)]

SHUFFLE: (k2,v2) → (k2, [list of v2])

REDUCE: (k2, [list of v2]) → [list of v3]

OUTPUT: list of values v3

Fixed pipeline

MapReduce Programming Model

INPUT: list of key-value pairs of (k1,v1)

MAP: (k1,v1) → [list of (k2,v2)]

SHUFFLE: (k2,v2) → (k2, [list of v2])

REDUCE: (k2, [list of v2]) → [list of v3]

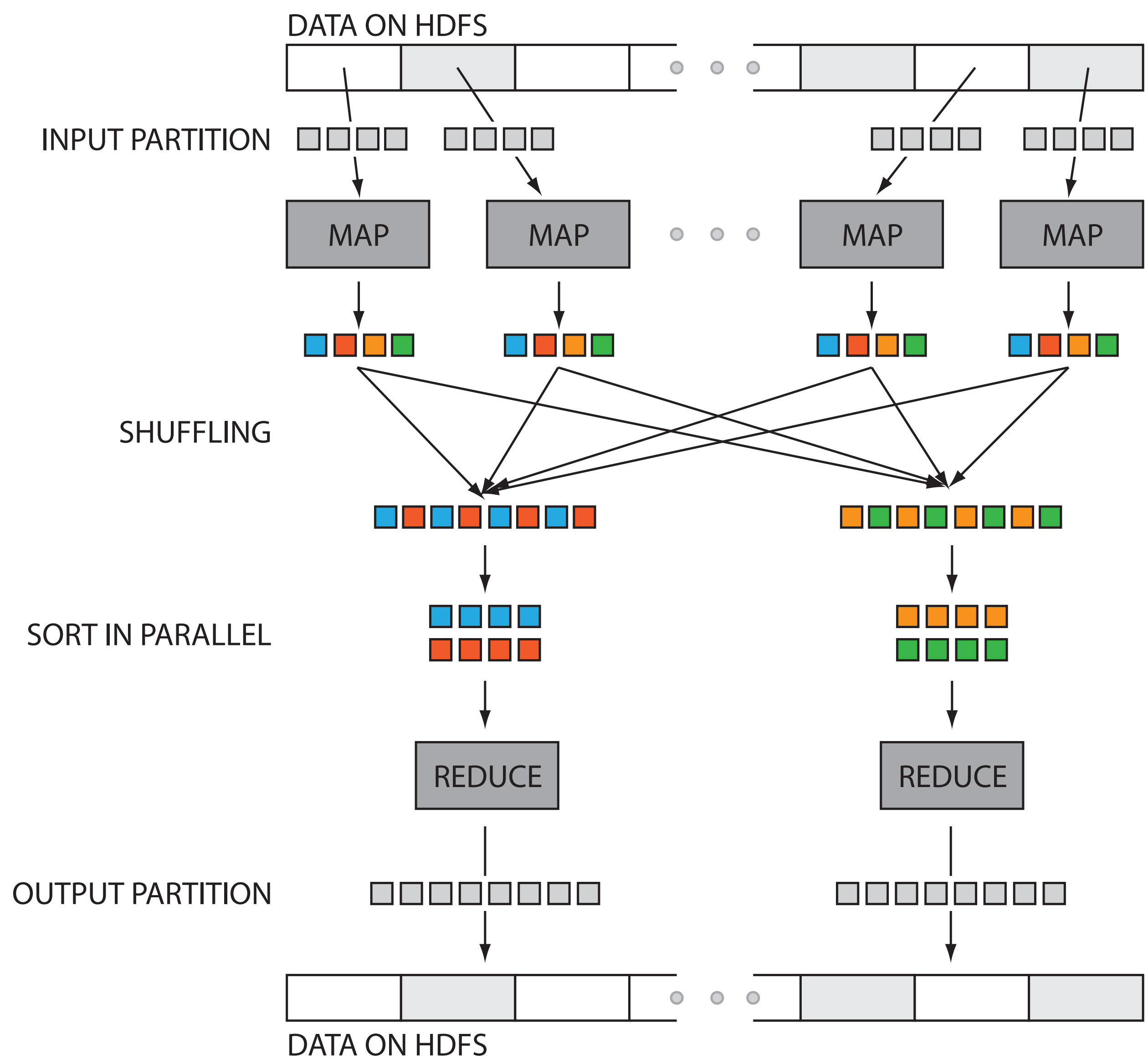
OUTPUT: list of values v3

User-defined and run in parallel

Hadoop is MapReduce + HDFS

- MapReduce implementation from Yahoo
- With its own distributed filesystem (HDFS)
- Java-based but support C++ map and reduce functions
  - Can incorporate C++ libraries

Hadoop Architecture



Visualization Algorithms with MapReduce

- Surface and volume rendering
- Regular grids isosurface extraction
- Triangular mesh simplification
- Can be chained together
- LARGE DATA!

Rendering: Rasterization vs. Ray Tracing

- Rasterization!
- Hadoop platform → graphics card (MapReduce pipeline → graphics pipeline)
  - Mapper: rasterizer and geometry shader
  - Reducer: fragment shader and composition
- Full pipeline control → rendering effects

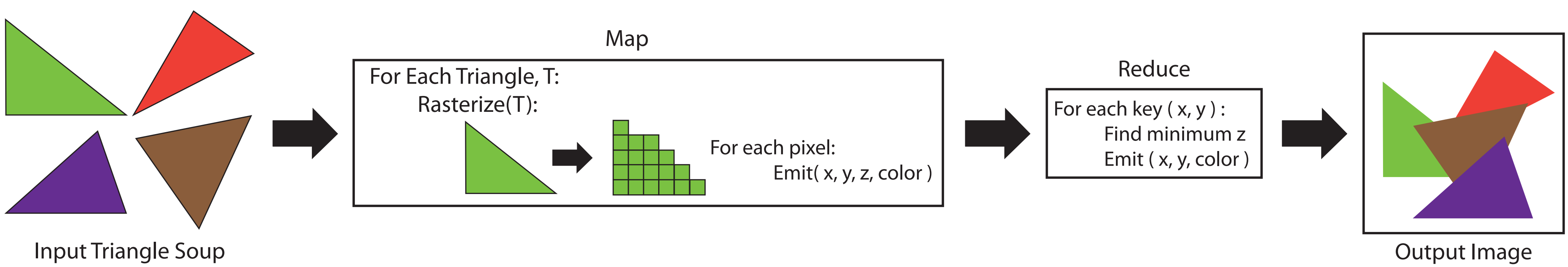
MapReduce Surface Rendering

INPUT: k1=N/A, v1=triangle vertices

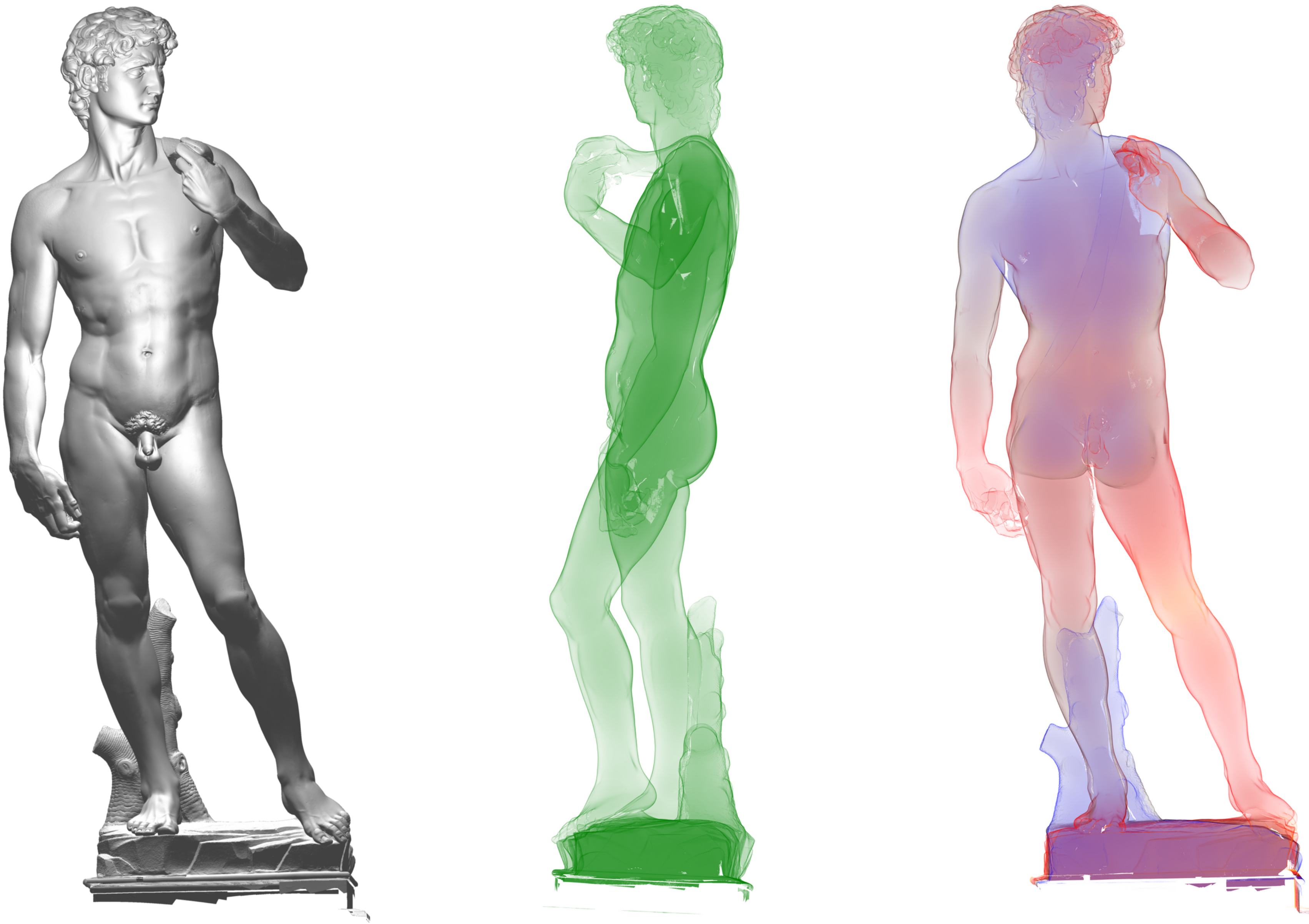
MAP: k2=pixel location, v2=(depth, color)

REDUCE: v3=composited pixel color

OUTPUT: pixel colors



1 GigaPixel of 1 Billion Tris



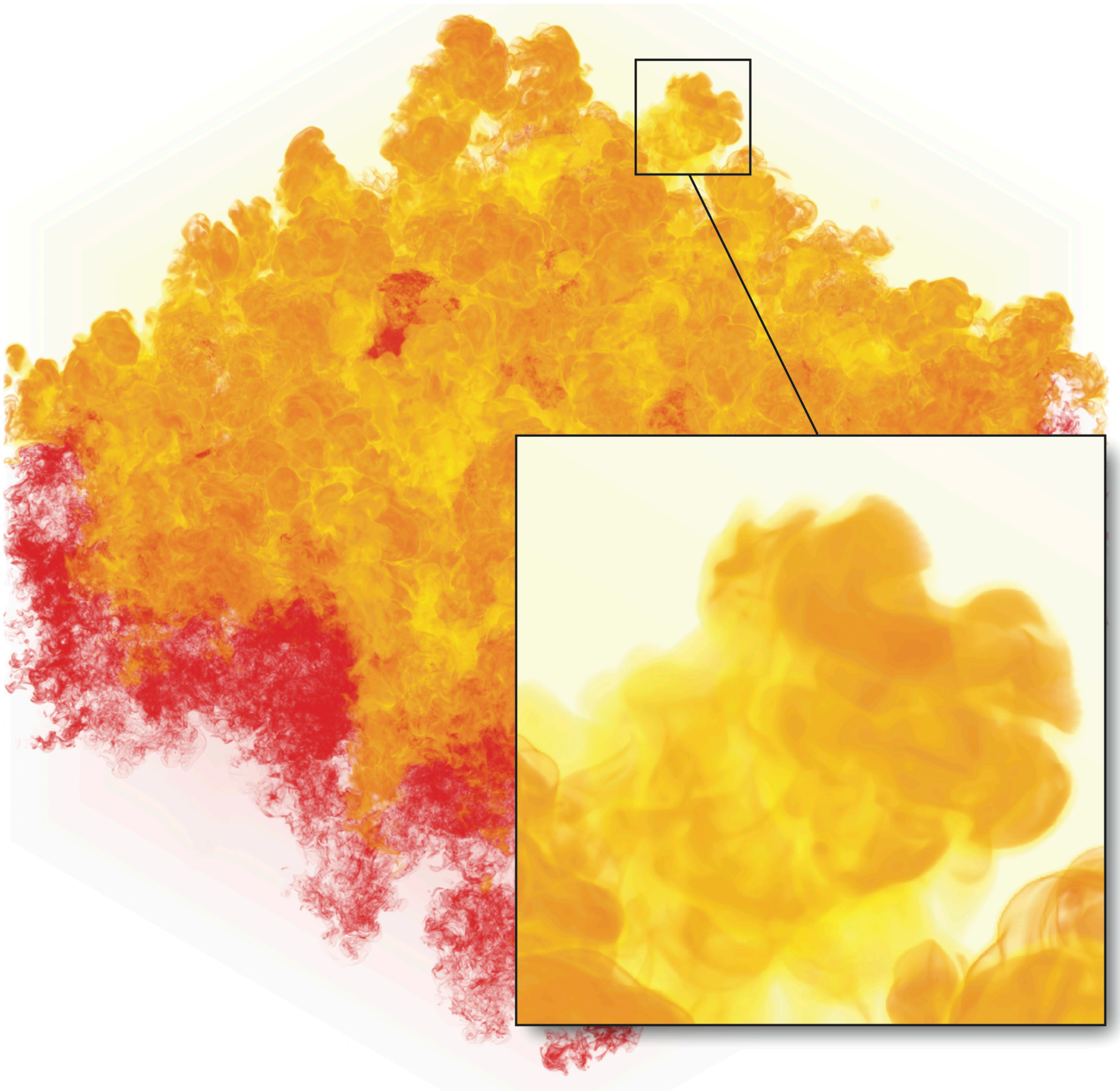
MapReduce Volume Rendering

Decompose primitives into triangles

MAP: v2=(depth, scalar)

REDUCE: perform integration and color lookup before composition

27 Billion Voxels Rendering



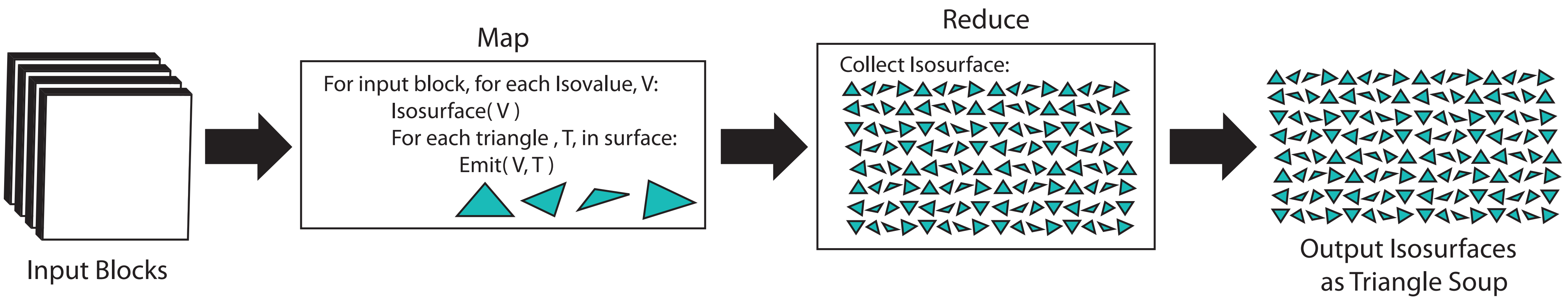
MapReduce Isosurface Extraction

Marching Cube on regular grids

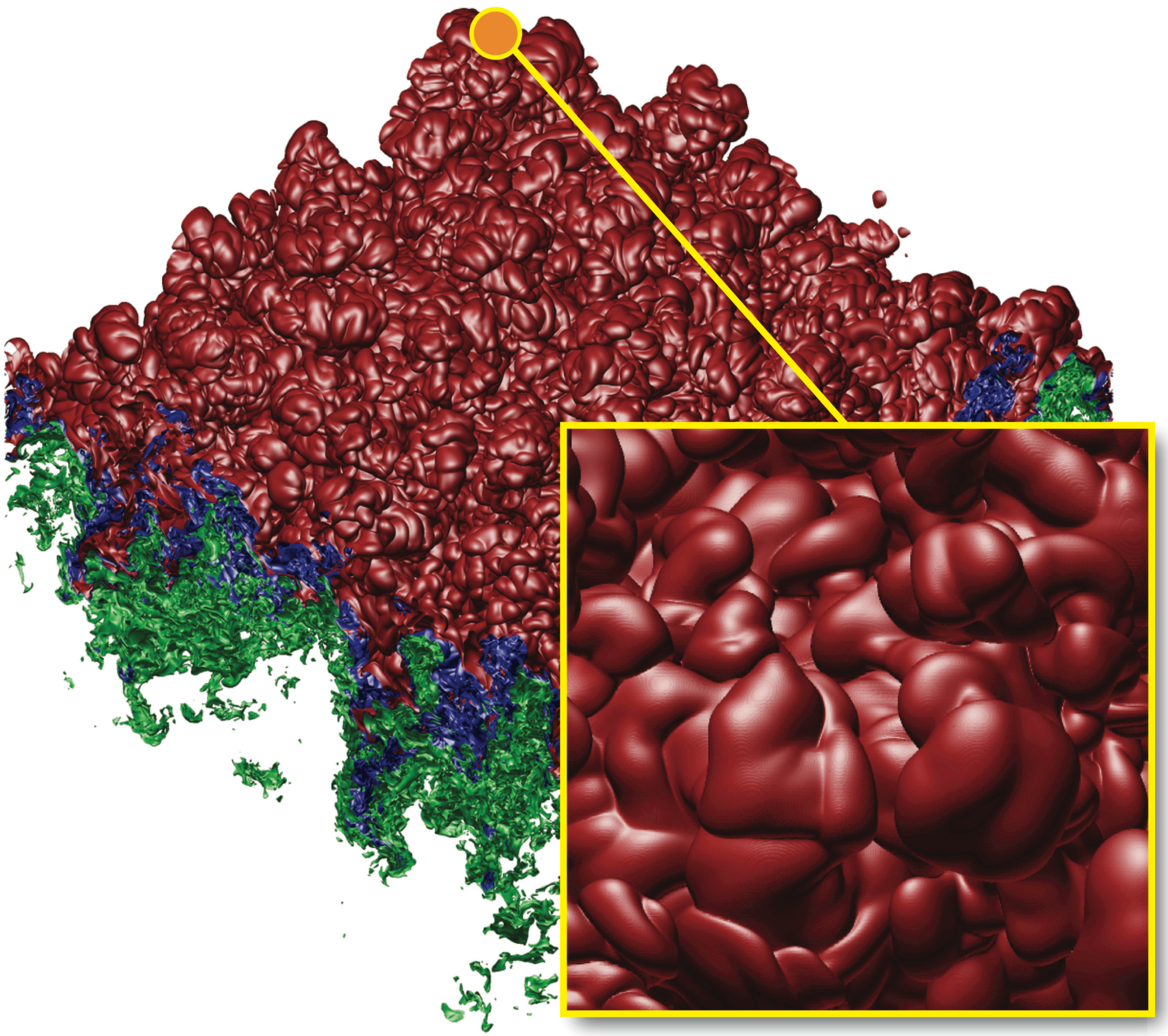
INPUT: k1=slice index, v2=slice grid points

MAP: k2=iso-value,v2=extracted triangles

REDUCE: k3=k2, v3=combined triangles



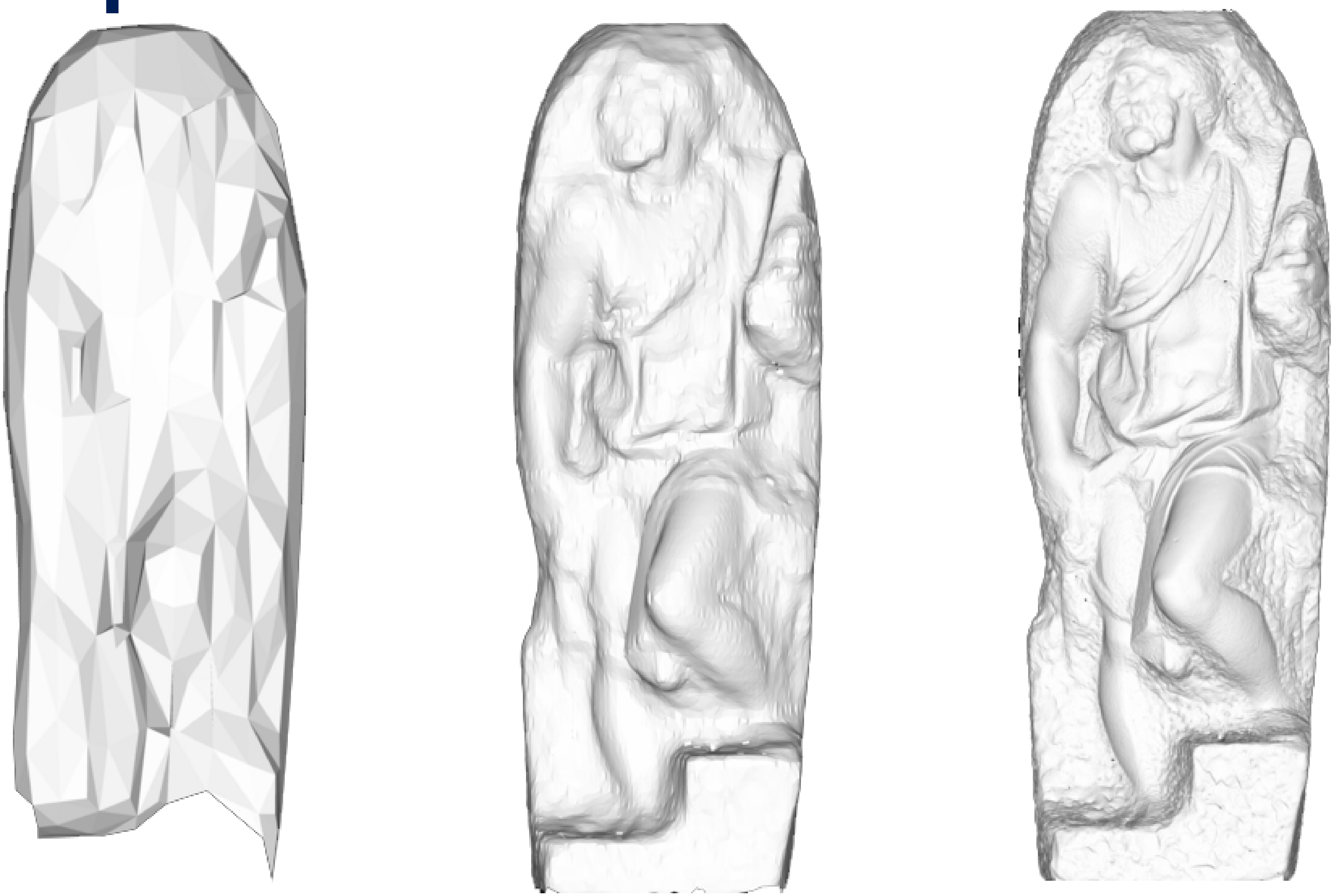
Isosurface + Rendering



MapReduce Surface Simplification

- Vertex clustering [Lindstrom and Silva 01]
- Clustering and re-building triangles both require data shuffling → 2 MR Jobs
- JOB1: bins vertices into regular grid and compute representative vertex locations
- JOB2: Creating representative triangles

Simplification of St. Matthew



8x8x8 64x64x64 512x512x512

Performance Results

- Hadoop baseline
  - A shared CLuE cluster, shared 768 cores
  - A private cluster: 60 nodes, 240 cores
- Visualization algorithms
  - Only on private cluster machines

Hadoop Baseline

WEAK-SCALING OF DATASIZE VS. THE NUMBER OF TASKS (on Cluster)

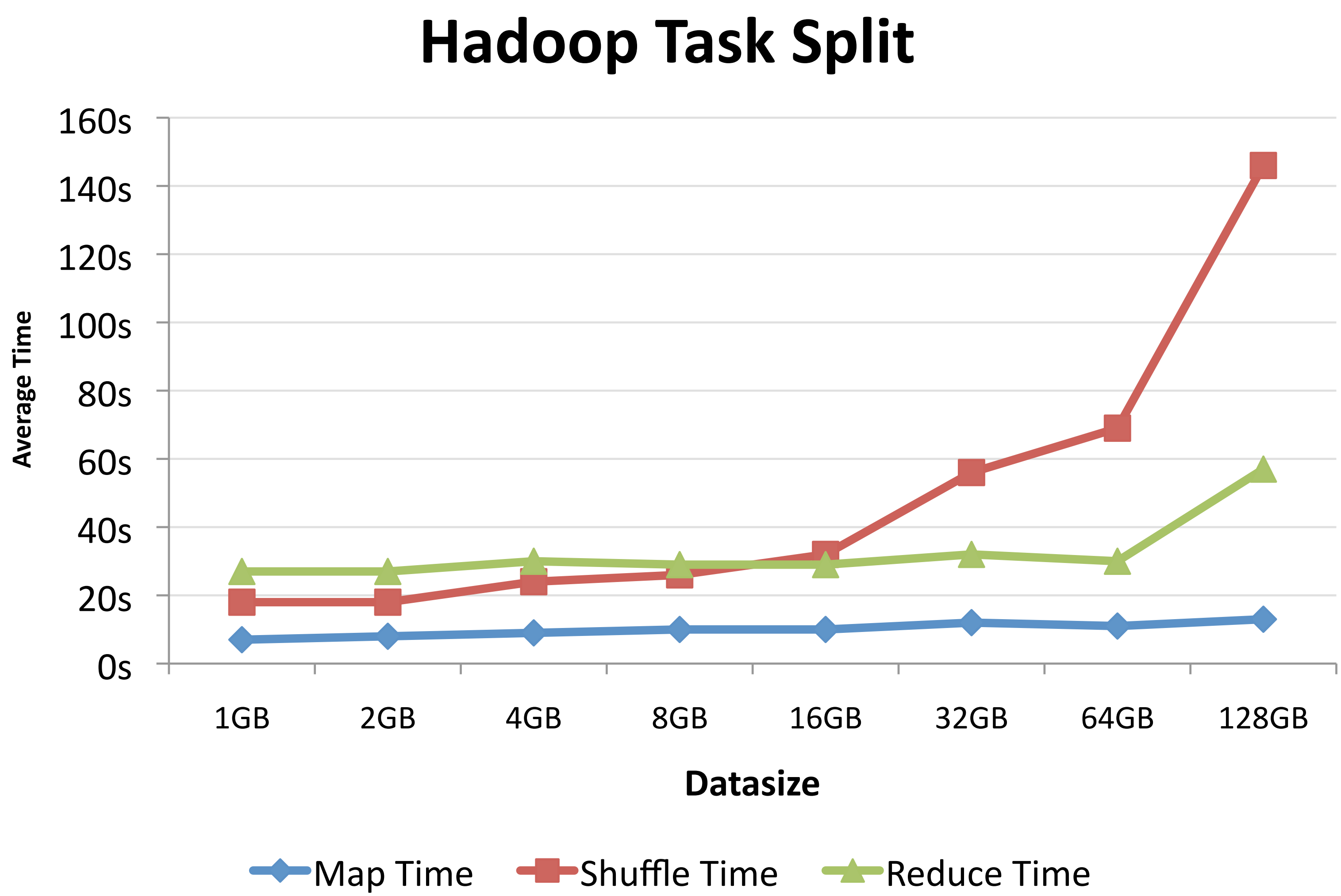
Datasize	#Maps	#Reduces	Map Time	Shuffle Time	Reduce Time	Total Time	I/O Rate	Data Rate
1GB	16	1	7s	18s	27s	63s	84 MB/s	16 MB/s
2GB	32	2	8s	18s	27s	66s	161 MB/s	31 MB/s
4GB	64	4	9s	24s	30s	75s	283 MB/s	55 MB/s
8GB	128	8	10s	26s	29s	78s	545 MB/s	105 MB/s
16GB	256	16	10s	32s	29s	90s	944 MB/s	182 MB/s
32GB	512	32	12s	56s	32s	130s	1308 MB/s	252 MB/s
64GB	1024	64	11s	69s	30s	153s	2222 MB/s	428 MB/s
128GB	2048	128	13s	146s	57s	320s	2125 MB/s	410 MB/s

- High latency/overhead (30s)
- High I/O cost (>5x data size)
- Scale well with data-size

WEAK-SCALING (on CLuE)

Datasize	Total Time	I/O Rate	Data Rate
1GB	971s	5 MB/s	1 MB/s
2GB	946s	11 MB/s	2 MB/s
4GB	986s	22 MB/s	4 MB/s
8GB	976s	44 MB/s	8 MB/s
16GB	1059s	80 MB/s	15 MB/s

Heavy I/O during Shuffling



Surface Rendering

WEAK SCALING (RESOLUTION)

Resolution	St. MATTHEW (13 GB) #M/R	St. MATTHEW CLuE time	St. MATTHEW Cluster time	St. MATTHEW File Written	ATLAS (18 GB) #M/R	ATLAS CLuE time	ATLAS Cluster time	ATLAS File Written
1.5 MP	256/256	1min 54s	46s	33MB	273/273	1min 55s	46s	41MB
6 MP	256/256	1min 42s	46s	147MB	273/273	2min 11s	46s	104MB
25 MP	256/256	1min 47s	46s	583MB	273/273	2min 12s	46s	412MB
100 MP	256/256	1min 40s	46s	2.3GB	273/273	2min 12s	46s	1.6GB
400 MP	256/256	2min 04s	46s	10.9GB	273/273	2min 27s	47s	5.5GB
1.6 GP	256/256	3min 12s	1min08s	53.14GB	273/273	3min 55s	55s	37.8GB
6.4 GP	256/256	9min 50s	2min55s	213GB	273/273	10min 30s	1min58s	151.8GB

WEAK SCALING (RESOLUTION AND REDUCE)

Resolution	St. MATTHEW (13 GB) CLuE #R	St. MATTHEW 256M time	St. MATTHEW Cluster #R	St. MATTHEW 480M time	ATLAS (18 GB) CLuE #R	ATLAS 256M time	ATLAS Cluster #R	ATLAS 480M time
1.5 MP	4	1min 13s	8	46s	4	1min 18s	8	46s
6 MP	8	1min 18s	15	46s	8	1min 19s	15	45s
25 MP	16	1min 18s	30	46s	16	1min 51s	30	46s
100 MP	32	2min 04s	60	47s	32	1min 52s	60	47s
400 MP	64	2min 04s	120	49s	64	2min 34s	120	46s
1.6 GP	128	4min 45s	240	1min06s	128	5min 06s	240	55s
6.4 GP	256	9min 50s	480	2min14s	256	10min 30s	480	1min41s

DAVID (1 Billion Triangles, 30GB)

	1.5 MP	6 MP	25 MP	100 MP	400 MP	1.6 GP	6.4 GP
Time	59s	59s	59s	59s	1m 1s	1m 40s	1m 47s

Surface Rendering

WEAK SCALING (RESOLUTION)

Resolution	St. MATTHEW (13 GB) #M/R	St. MATTHEW CLuE time	St. MATTHEW Cluster time	St. MATTHEW File Written	ATLAS (18 GB) #M/R	ATLAS CLuE time	ATLAS Cluster time	ATLAS File Written
1.5 MP	256/256	1min 54s	46s	33MB	273/273	1min 55s	46s	41MB
6 MP	256/256	1min 42s	46s	147MB	273/273	2min 11s	46s	104MB
25 MP	256/256	1min 47s	46s	583MB	273/273	2min 12s	46s	412MB
100 MP	256/256	1min 40s	46s	2.3GB	273/273	2min 12s	46s	1.6GB
400 MP	256/256	2min 04s	46s	10.9GB	273/273	2min 27s	47s	5.5GB
1.6 GP	256/256	3min 12s	1min08s	53.14GB	273/273	3min 55s	55s	37.8GB
6.4 GP	256/256	9min 50s	2min55s	213GB	273/273	10min 30s	1min58s	151.8GB

WEAK SCALING (RESOLUTION AND REDUCE)

Resolution	St. MATTHEW (13 GB) CLuE #R	St. MATTHEW CLuE 256M time	St. MATTHEW Cluster #R	St. MATTHEW Cluster 480M time	ATLAS (18 GB) CLuE #R	ATLAS CLuE 256M time	ATLAS Cluster #R	ATLAS Cluster 480M time
1.5 MP	4	1min 13s	8	46s	4	1min 18s	8	46s
6 MP	8	1min 18s	15	46s	8	1min 19s	15	45s
25 MP	16	1min 18s	30	46s	16	1min 51s	30	46s
100 MP	32	2min 04s	60	47s	32	1min 52s	60	47s
400 MP	64	2min 04s	120	49s	64	2min 34s	120	46s
1.6 GP	128	4min 45s	240	1min06s	128	5min 06s	240	55s
6.4 GP	256	9min 50s	480	2min14s	256	10min 30s	480	1min41s

DAVID (1 Billion Triangles, 30GB)

	1.5 MP	6 MP	25 MP	100 MP	400 MP	1.6 GP	6.4 GP
Time	59s	59s	59s	59s	1m 1s	1m 40s	1m 47s

vs. 30 hours [Ize et al. 11]

Volume Rendering

TETRAHEDRAL MESH VOLUME RENDERING (on Cluster)

Model	# Tetrahedra	#Triangles	Time	#Fragments	Read	Write
Spx	0.8 millions	1.6 millions	3m 29s	9.8 billions	320 GB	473 GB
Fighter	1.4 millions	2.8 millions	2m 20s	5.3 billions	172 GB	254 GB
Sf1	14 millions	28 millions	6m 53s	16.8 billions	545 GB	807 GB
Bullet	36 millions	73 millions	4m19s	12.7 billions	412 GB	610 GB

STRUCTURED GRID VOLUME RENDERING (on Cluster)

Model	Grid Size	#Triangles	Time	#Fragments	Read	Write
RT27	3072^3 floats	161 billions	19m 20s	22.2 billions	1.2 TB	1.6 TB

Volume Rendering

TETRAHEDRAL MESH VOLUME RENDERING (on Cluster)

Model	# Tetrahedra	#Triangles	Time	#Fragments	Read	Write
Spx	0.8 millions	1.6 millions	3m 29s	9.8 billions	320 GB	473 GB
Fighter	1.4 millions	2.8 millions	2m 20s	5.3 billions	172 GB	254 GB
Sf1	14 millions	28 millions	6m 53s	16.8 billions	545 GB	807 GB
Bullet	36 millions	73 millions	4m19s	12.7 billions	412 GB	610 GB

STRUCTURED GRID VOLUME RENDERING (on Cluster)

Model	Grid Size	#Triangles	Time	#Fragments	Read	Write
RT27	3072^3 floats	161 billions	19m 20s	22.2 billions	1.2 TB	1.6 TB

vs. 22 seconds on 1728 cores [Howison et al. 10]

Isosurfacing

#Iso	Richtmyer-Meshkov (7.6GB)		Rayleigh-Taylor (108GB)	
	Total Time	Written	Total Time	Written
1	30s	1.78GB	39s	8.4GB
2	31s	5.9GB	39s	11.1GB
4	45s	22.5GB	1m 5s	62.0GB
8	45s	52.7GB	1m 25s	155.9GB
16	1m 26s	112.4GB	2m 50s	336.6GB

Isosurfacing

#Iso	Richtmyer-Meshkov (7.6GB)		Rayleigh-Taylor (108GB)	
	Total Time	Written	Total Time	Written
1	30s	1.78GB	39s	8.4GB
2	31s	5.9GB	39s	11.1GB
4	45s	22.5GB	1m 5s	62.0GB
8	45s	52.7GB	1m 25s	155.9GB
16	1m 26s	112.4GB	2m 50s	336.6GB

vs. 250 seconds on 64 cores by [Isenburg et al. 10]

Simplification

Size	St MATTHEW (13 GB)					ATLAS (18 GB)				
	CLuE Time		Cluster Time		Output Size	CLuE Time		Cluster Time		Output Size
	Job 1	Job 2	Job 1	Job 2		Job 1	Job 2	Job 1	Job 2	
8^3	5m 45s	52s	58s	56s	22 KB	5m 45s	52s	54s	55s	23 KB
16^3	3m 54s	49s	58s	55s	98 KB	3m 54s	49s	54s	54s	105 KB
32^3	3m 51s	49s	55s	54s	392 KB	3m 51s	49s	51s	52s	450 KB
64^3	3m 40s	49s	57s	54s	1.6 MB	3m 40s	49s	55s	55s	1.9 MB
128^3	4m 12s	49s	55s	58s	6.4 MB	4m 12s	49s	52s	52s	7.5 MB
256^3	3m 50s	49s	55s	55s	26 MB	3m 50s	49s	55s	55s	30 MB

Simplification

Size	St MATTHEW (13 GB)					ATLAS (18 GB)				
	CLuE Time		Cluster Time		Output Size	CLuE Time		Cluster Time		Output Size
	Job 1	Job 2	Job 1	Job 2		Job 1	Job 2	Job 1	Job 2	
8^3	5m 45s	52s	58s	56s	22 KB	5m 45s	52s	54s	55s	23 KB
16^3	3m 54s	49s	58s	55s	98 KB	3m 54s	49s	54s	54s	105 KB
32^3	3m 51s	49s	55s	54s	392 KB	3m 51s	49s	51s	52s	450 KB
64^3	3m 40s	49s	57s	54s	1.6 MB	3m 40s	49s	55s	55s	1.9 MB
128^3	4m 12s	49s	55s	58s	6.4 MB	4m 12s	49s	52s	52s	7.5 MB
256^3	3m 50s	49s	55s	55s	26 MB	3m 50s	49s	55s	55s	30 MB

Job 2 operates on decimated vertices ➔ much faster

Hadoop Lessons

- Scale well where data-parallelism fits
- Performance is sensitive to intermediate data size
- Easy to use, but hard to configure
- Lack the ability for chaining jobs
- Data upload cannot be done in parallel

Objectives

- Can we use cloud computing for Vis?
  - Efficiency
  - Scalability
  - LARGE DATA handling

Conclusions

- Visualization can operate on the cloud!
  - Efficiency: high overhead but comparable performance (for data-parallelism)
  - Scalability: limit by intermediate data size
- Capable of visualizing LARGE DATA if
  - Interactivity is not required
  - Techniques can be data-parallelized

Future Work

- Try other MapReduce implementations
  - MapReduce-MPI, Cascading, Piccolo
- Try other programming paradigms
  - DryadLINQ, Sector/Sphere
- Using structured data storage (DBs) back-ends

Acknowledgements

- NSF, DoE, IBM, NVIDIA

This work was supported in part by the National Science Foundation (CCF-08560,CCF-0702817, CNS-0751152, CNS-1153503, IIS-0844572, IIS-0904631, IIS-0906379, IIS-1153728, and NIH ITKv4), the Department of Energy, CNPq (processes 200498/2010-0, 569239/2008-7, and 491034/2008-3), IBM Faculty Awards and NVIDIA Fellowships. This work was also performed under the auspices of the U.S. Department of Energy by the University of Utah under contract DE-SC0001922 and DE-FC02-06ER25781 and by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344, LLNL-JRNL-453051.

- Datasets: Stanford Graphics Lab, Marc Levoy (the new David scan model), Bill Cabot, Andy Cook and Paul Miller at LLNL (the Rayleigh-Taylor dataset)

Question?

Thank you!