Mochi: Visual Log-Analysis Based Tools for Debugging Hadoop

Jiaqi Tan

Xinghao Pan, Soila Kavulya,
Rajeev Gandhi, Priya Narasimhan

PARALLEL DATA LABORATORY
Carnegie Mellon University

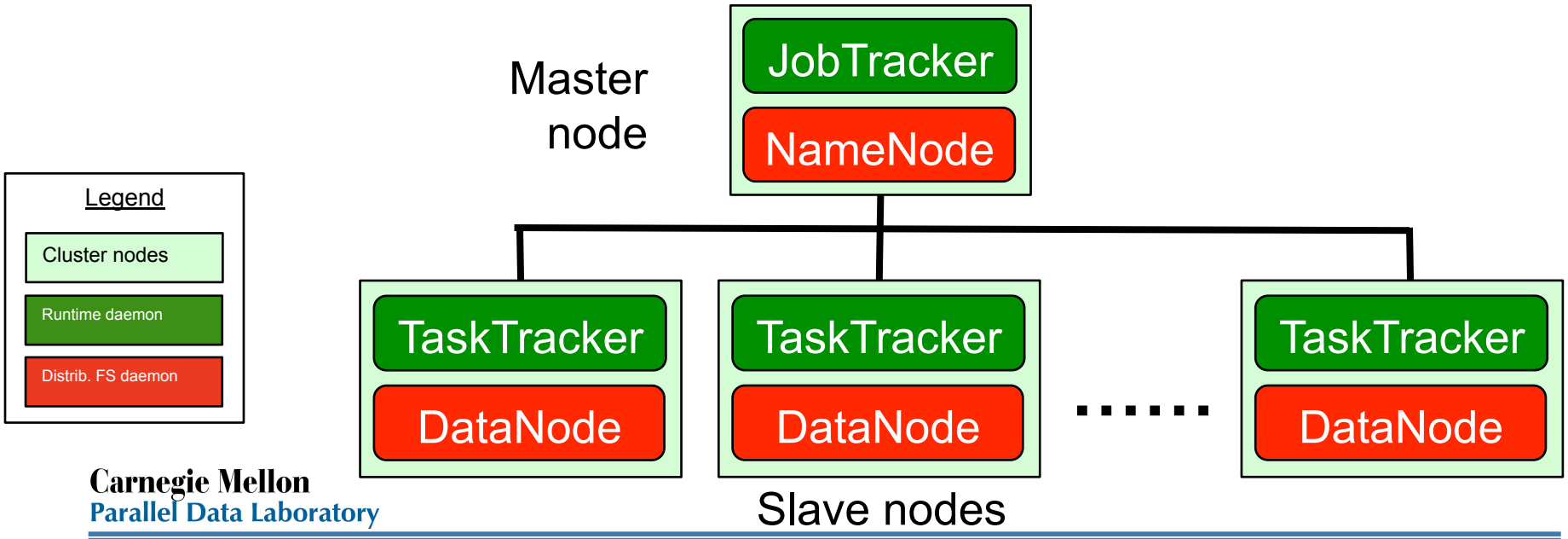
Carnegie Mellon
Parallel Data Laboratory

Motivation

- Debugging cloud computing programs is hard
  - Involves both computation and data movement
  - Large-scale, distributed, many interdependencies
  - Focus: debugging MapReduce programs
- Current tools (traditional debuggers) lacking
  - Too fine-grained, too much information
    e.g., `jstack, jprof`
  - Do not show higher-level MapReduce abstractions
    i.e., information not at level of maps, reduces
- Need to expose all factors affecting MapReduce program performance

Architecture of Hadoop

- Runtime and Distributed Filesystem (HDFS)
- Master/Slave: one Master, many Slaves
- Runs arbitrary MapReduce user code
- Daemons natively generate activity logs



Mochi: Approach

- Log analysis
  - Extract views of node-local execution: SALSA [USENIX WASL 08]
  - Correlate execution across nodes: distributed control-flow, distributed data-flow
  - Correlate execution and filesystem layers: *Job-centric Data-flow* (JCDF)
- Visualization of behavior
  - Show behavior along three dimensions: Time, space, and (data) volume

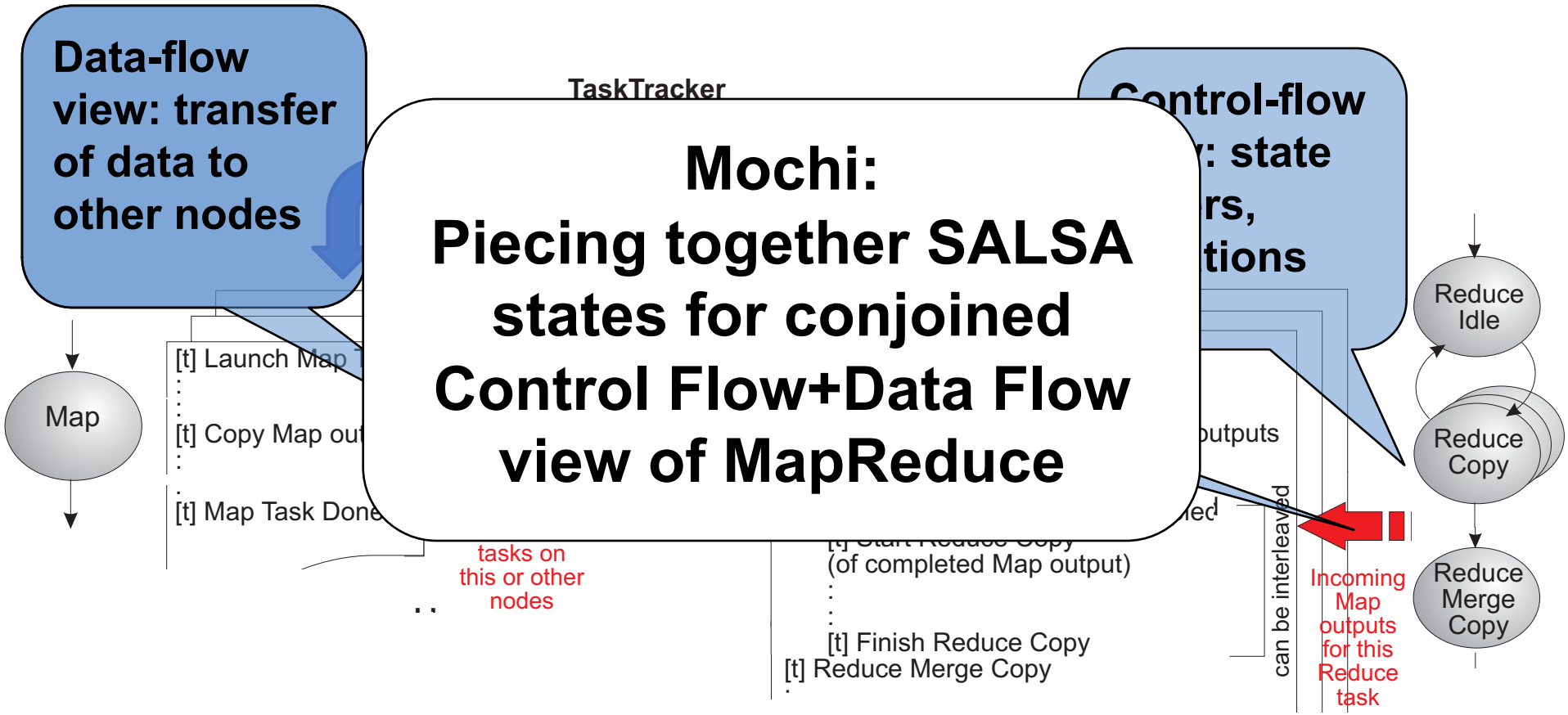
Goals

- Extract program behavior at MapReduce level of abstraction
  - Factors affecting program performance
  - Aspects of framework not visible to user code
    e.g. task scheduling, data distribution
- Expose different perspectives of performance
  - Present different levels of aggregation
    e.g. time-aggregates, space-aggregates
- Remain transparent to Hadoop
  - No added instrumentation, work in production environments

Outline

- State-machine extraction from logs
- Abstraction of MapReduce Execution: Job-centric Data-flow (JCDF)
- Visualization
- Case-studies
- Future Work

State-machine Extraction



Mochi

SALSA

Execution (MapReduce) layer state-machine view (per-node)

Distributed FileSystem (HDFS) state-machine view (per-node)

Prior work

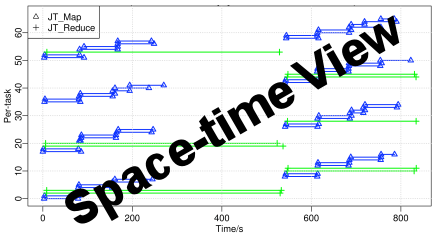
Mochi: Log Analysis

Conjoined Control-flow + Data-flow: Job-Centric Data-Flow (JCDF) (across all nodes)

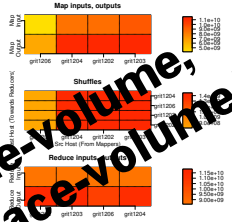
Causal flows of data and processing: Realized Execution Paths (REP)

New Contributions

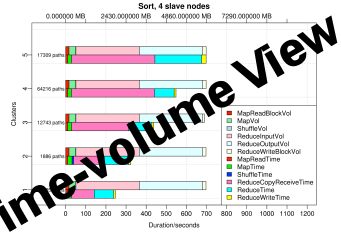
Mochi: Visualization



Space-time View



Time-volume, Space-volume View

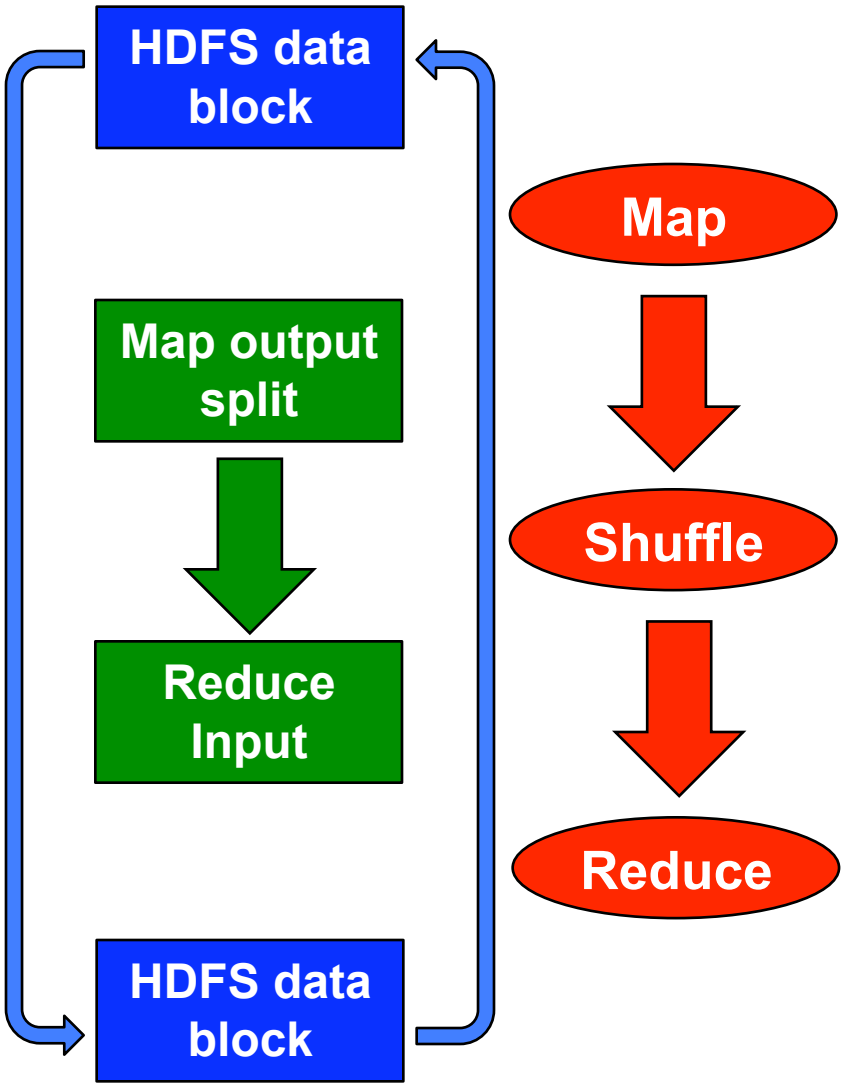


Time-volume View

Carnegie Mellon
Parallel Data Laboratory

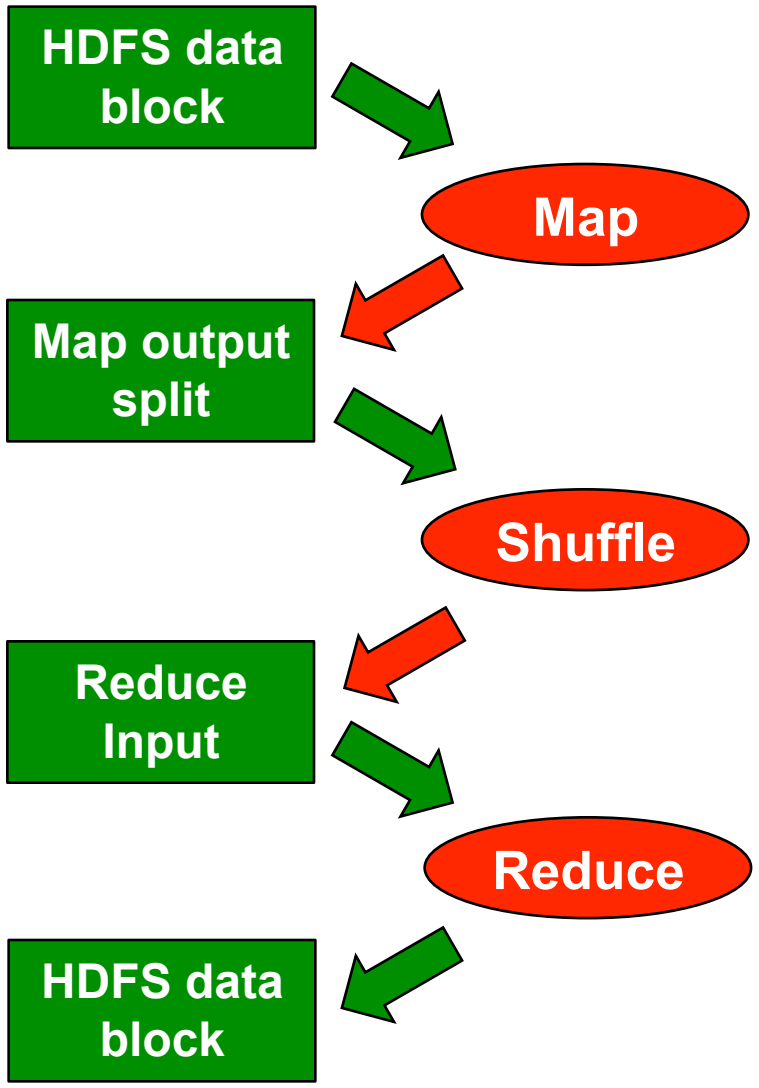
Control-Flows, Data-Flows

- **Control-flow**
  - Distributed flow of execution across nodes: *Map* → *Reduce via Shuffles*
- Distributed data-flow
  - **Data paths of Map outputs shuffled to Reduces**
  - **HDFS data blocks read into and written out of jobs**



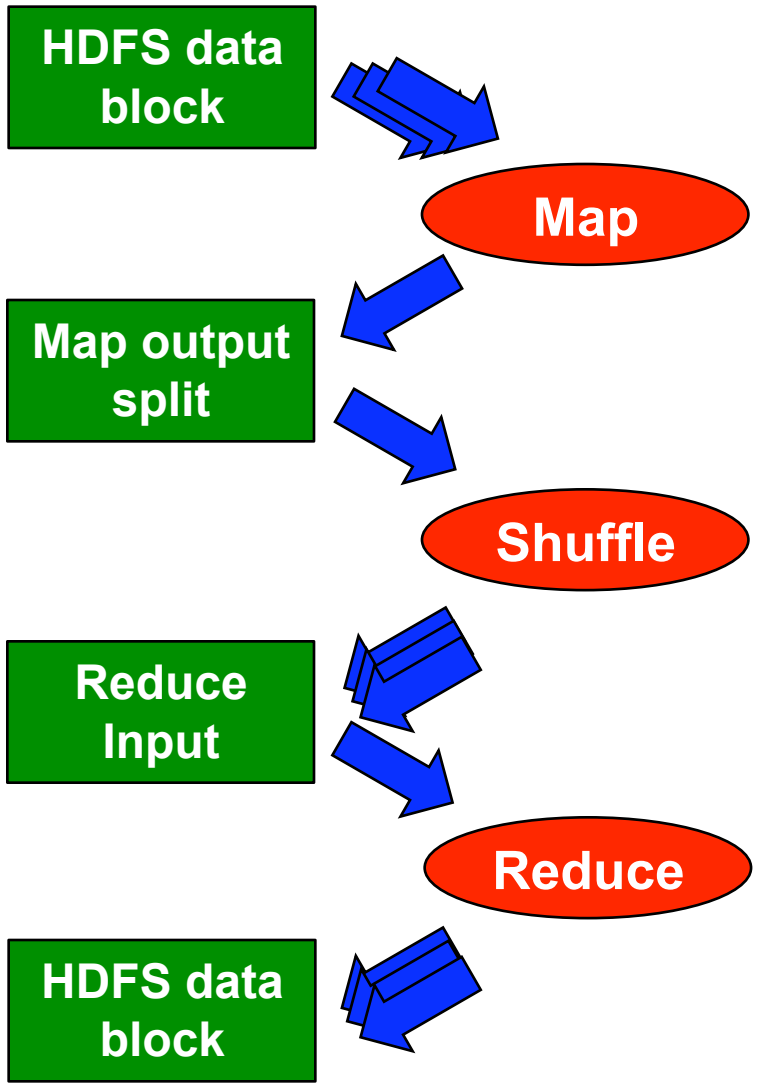
Job-Centric Data-Flows (JCDF)

- Control+Data Flows = JCDF
- Correlate **data** and **execution** paths in time: co-occurring:
  - Block-reads, maps;
  - Block-writes, reduces
- Create **conjoined causal paths (REP)**



Realized Execution Paths (REP)

- JCDF (Job-centric Data-Flows)
  - Large graph of all causal flows
  - Contains all causal flows distributed through system
- **REP**
  - Single causal path through execution
  - Extract every REP from JCDF using Depth-First Search



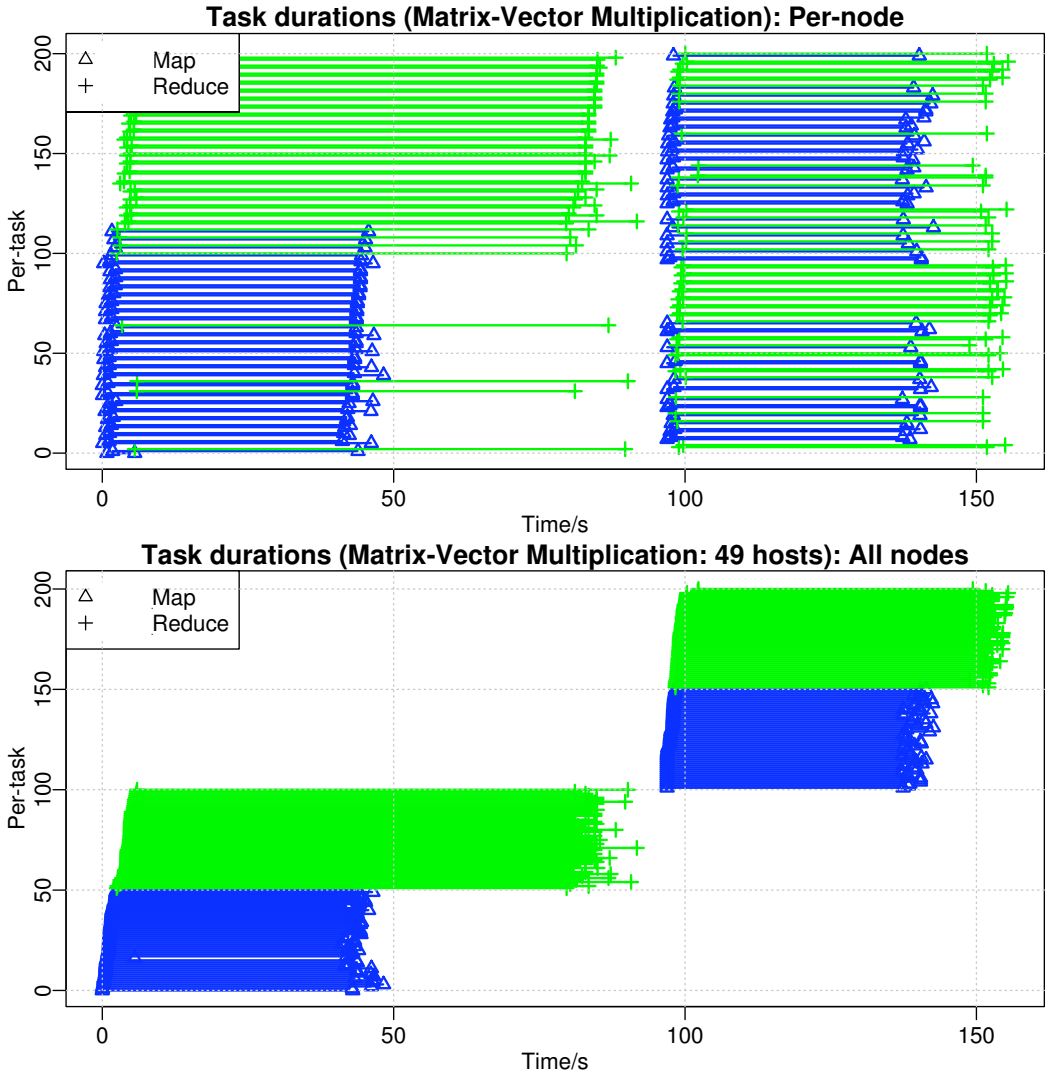
Validation: Visualizations

- Testbed
  - Yahoo! M45 4000-processor cluster
  - Production environment: no added instrumentation
  - Hadoop 0.18.3 with Hadoop-on-Demand
- Workloads:
  - Real CMU user workloads:
    Matrix-Vector Multiplication [CMU-ML-08-117]
  - Benchmarks: SleepJob (No-op), Sort
- Processing

  C++ Log parsing + GNU R graph generation

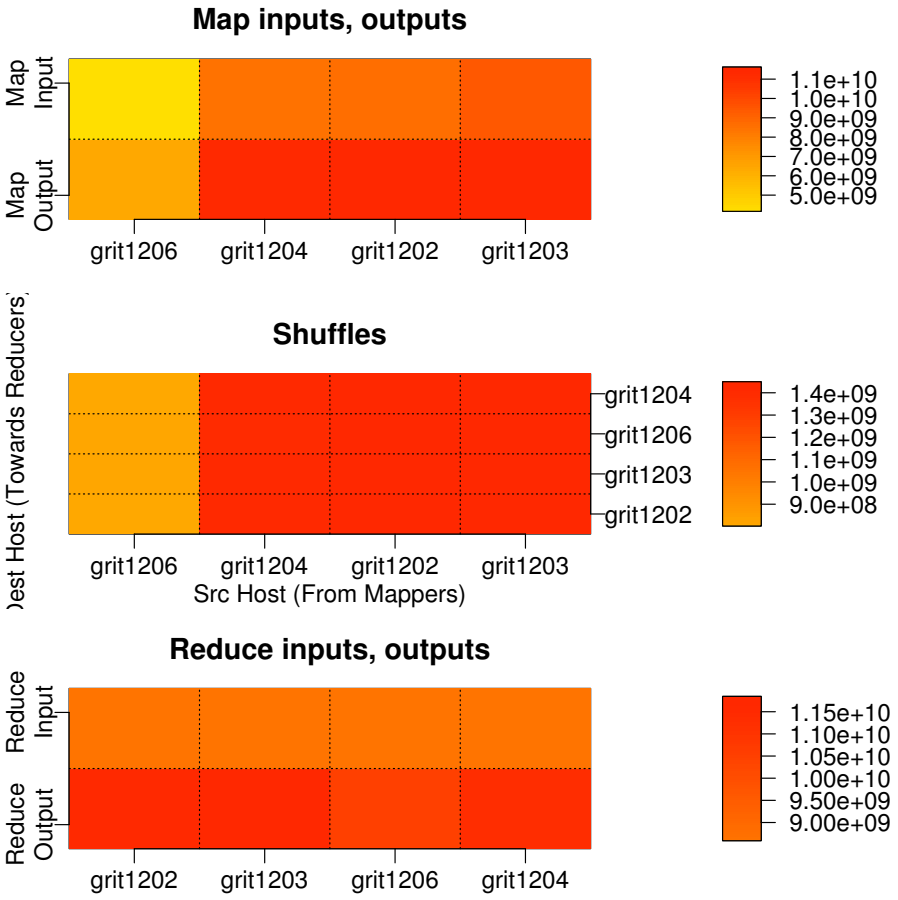
‘Swimlanes’ (Space-Time)

- Task-execution in time, across nodes
- Each task gets a line for its duration
- Shows where Hadoop job and nodes spend their time



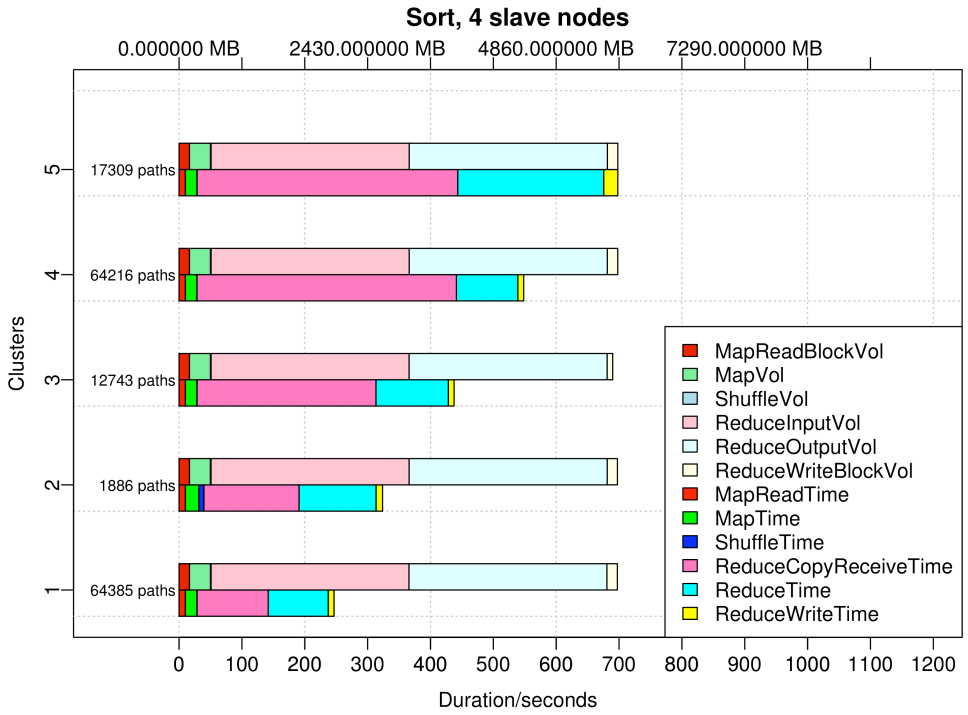
MIROS

- MIROS (Map Inputs, Reduce Outputs, Shuffles)
- Aggregates data volumes across:
  - All tasks per node
  - Entire job
- Shows skewed data flows, bottlenecks



Realized Execution Paths (REP)

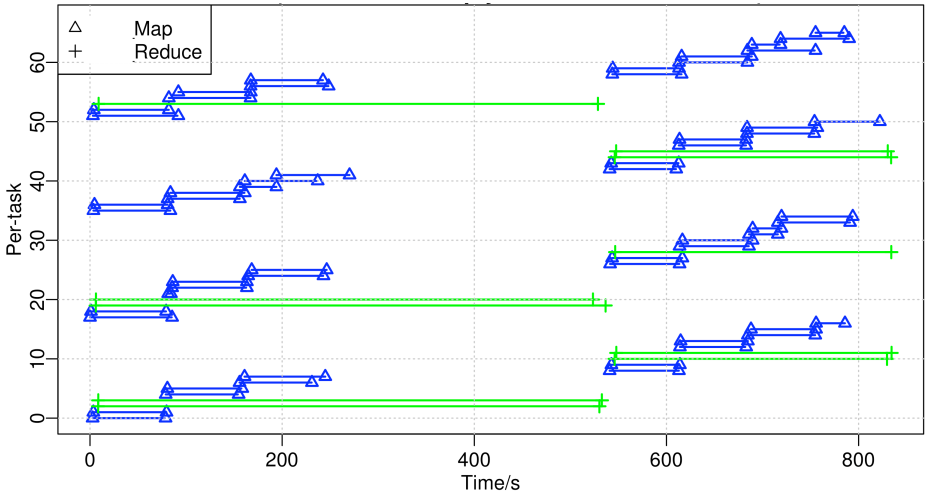
- Per-flow Volume-Duration correlation
- Flows clustered for scalable visualization
- For each stage in flow, shows:
  - Duration spent
  - Input+output volumes
- Compares time taken vs. volume processed



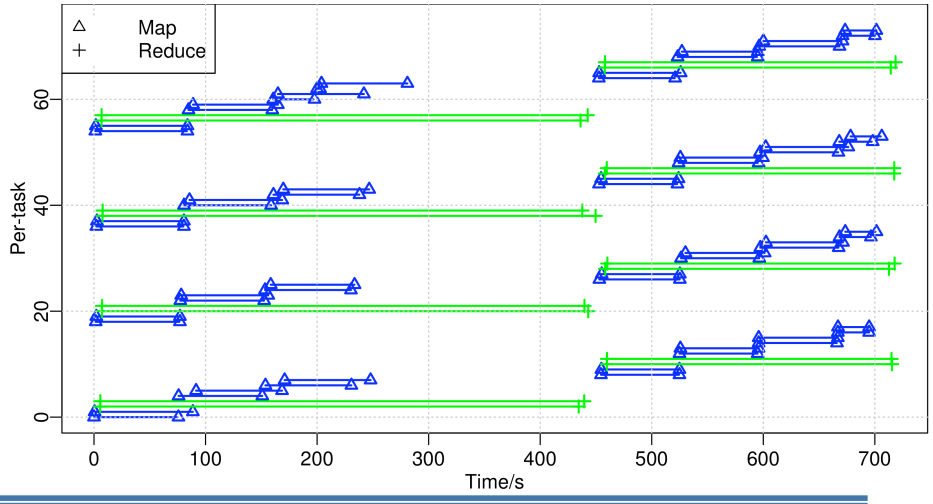
Opportunities for Optimization

- User M45 workload:
  - Matrix-Vector Multiplication
  - 5-node cluster
- Before: Some nodes idle during reduce
- Adjusted # reducers
- After: All nodes had reducers; 15% faster

Before



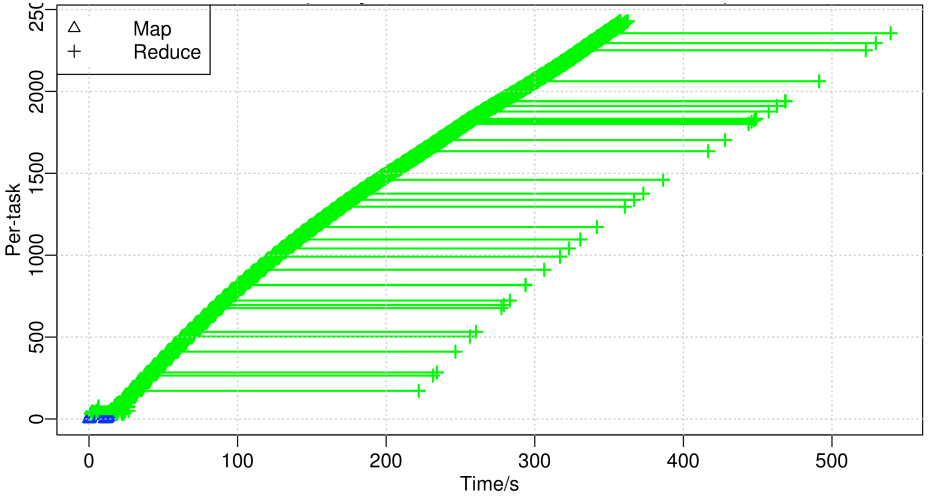
After



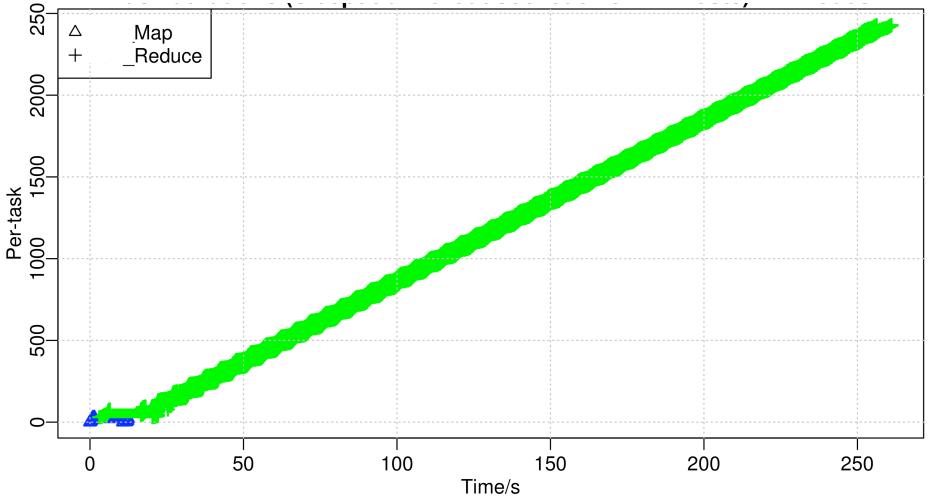
Debugging: Delayed Socket Creation

- M45 workload: "No-op" Sleep job
  - Each Map/Reduce sleeps for 100 ms
- Tasks had unusually long durations: exactly 3 minutes
- Identified to be bad routing interaction
- Disabled Java IPv6

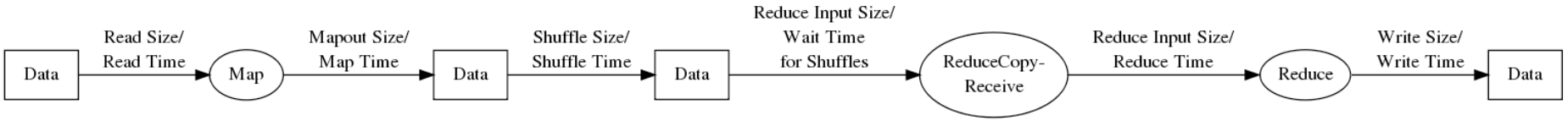
Before



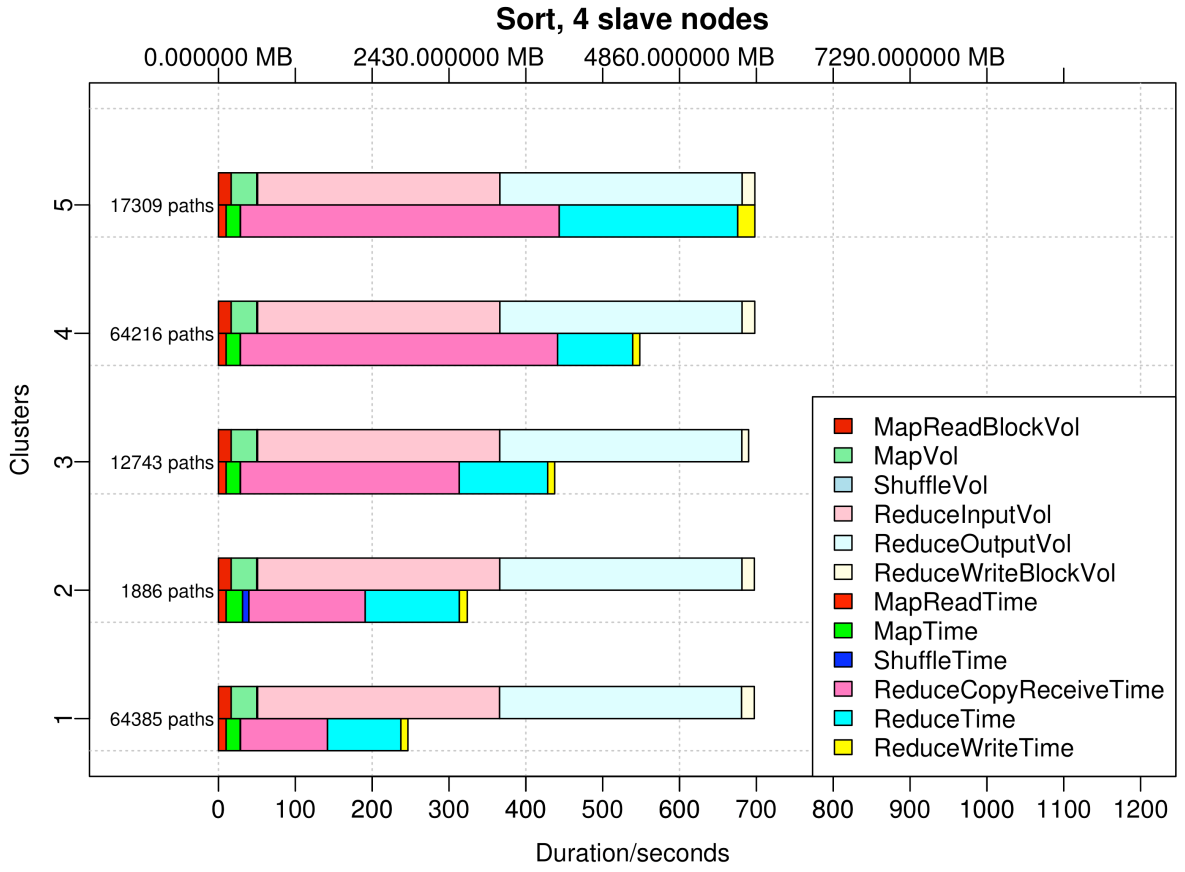
After



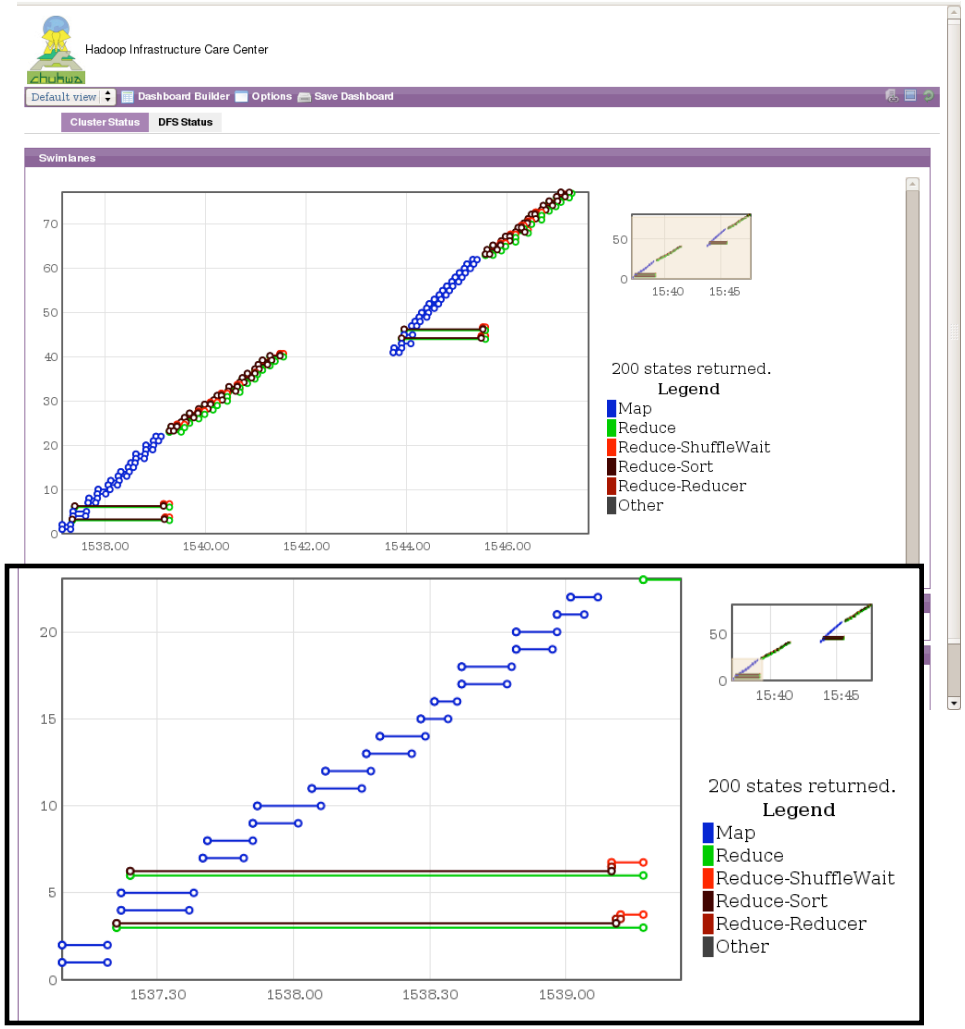
Characterizing Jobs using REPs



- Each REP: instantiation of the generic execution
- Plot of Sort workload: Significant time spent waiting on shuffles



Next Steps



- Collaboration with Yahoo! – we are implementing Mochi for the Hadoop Chukwa project
- Web-based visualization widgets for Hadoop Infrastructure Care Center
- "Swimlanes" currently available in Chukwa 0.2 (CHUKWA-279)

Conclusion

- Extracting program behavior with log analysis
  - Distributed Control+Data-Flow
  - Job-centric Data-flows (JCDF), Realized Execution Paths (REP)
- Visualizations
  - Swimlanes – Task execution in time and space
  - MIROS – Aggregate data-flows across cluster
  - REP – Volume-Duration correlation
- Case-studies
  - Performance debugging and optimization
  - Job characterization: Bottleneck stages

References

- [CMU-ML-08-117] U. Kang, C. Tsourakakis, A. P. Appel, C. Faloutsos, J. Leskovec. HADI: Fast Diameter Estimation and Mining in Massive Graphs with Hadoop. *CMU ML Tech Report*, CMU-ML-08-117, 2008.
- [USENIX WASL 08] J. Tan, X. Pan, S. Kavulya, R. Gandhi, P. Narasimhan. SALSA: Analyzing Logs as StAte Machines. *First USENIX Workshop on Analysis of System Logs (WASL)*, San Diego, CA, Dec 2008.

Other work:

- X. Pan, S. Kavulya, J. Tan, R. Gandhi, P. Narasimhan. Ganesha: Black-Box Diagnosis for MapReduce Systems. *Second Workshop on Hot Topics in Measurement & Modeling of Computer Systems (HotMetrics)*, Seattle, WA, Jun 2009.