# SparkBench: A Comprehensive Spark Benchmarking Suite Characterizing In-memory Data Analytics

Min LI,, Jian Tan, Yandong Wang, Li Zhang,
Valentina Salapura, Alan Bivens

IBM TJ Watson Research Center

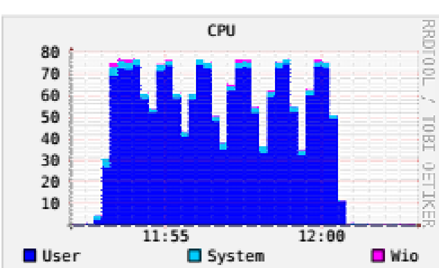
# SparkBench Overview*

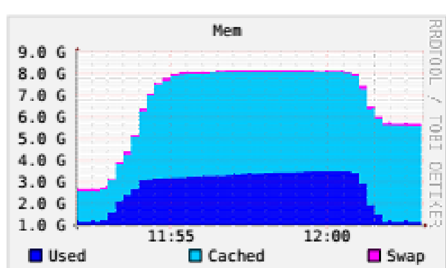
- A Spark benchmarking suite charactering in memory data analysis to provide guidance of Spark system design and performance optimization
  - A data generator automatically generates input data sets with various sizes
  - Diverse and representative workloads ( extensible to new workloads )
    - Machine learning: Logistic regression, support vector machine, matrix factorization
    - Graph processing: pagerank, svdplusplus, triangle count
    - Streaming: twitter, pageview
    - SQL query applications: hive,RDDRelation
  - Explore different parameter configurations easily
  - Reported Metrics:
    - supported: job execution time, input data size, data process rate
    - under development: shuffle data, RDD size, resource consumption, integration with monitoring tool
- Workload characterization and study of parameter impacts
  - Diverse and representative date sets: Wikipedia, Google web graph, Amazon movie review
  - Charactering workloads in terms of resource consumption, data access patterns and time information, job execution time, shuffle data
  - Studying the impact of Spark configuration parameters

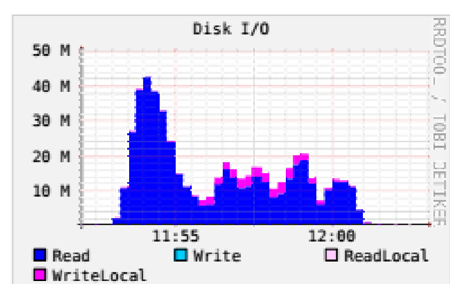
* A paper currently under submission : “SPARKBENCH: a Spark Benchmarking Suite Characterizing Large-scale in Memory Data Analysis”

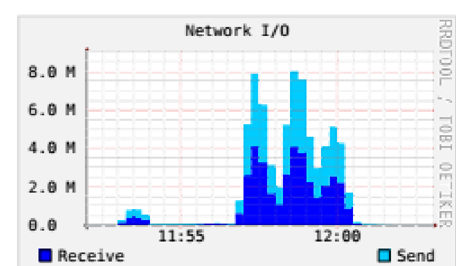
# What SparkBench is designed for?

- Provide quantitative comparison for different platforms and hardware cluster setups
  - e.g. the comparison between IBM Power system VS Intel System. IBM cloud VS Amazon cloud
- Provide quantitative comparison for Spark system optimization
- Enable in-depth study of performance implication of Spark system in various aspects
  - workload characterization, parameter impact, scalability, fault tolerance
- Provide insights and guidance for cluster sizing and provisioning
  - If a user aims to provision a spark cluster for usage, what will the performance look like?
  - Help identify resource bottleneck



# Workloads and Data Sets

| Application Type | Workload | Input Dataset |
|---|---|---|
| Machine Learning | Logistic Regression | Wikipedia |
| | Support Vector Machine | Wikipedia |
| | Matrix factorization | Amazon Movie Review |
| Graph Computation | PageRank | Google Web Graph |
| | SVD++ | Amazon Movie Review |
| | TriangleCount | Amazon Movie Review |
| SQL engine | Hive | E-commerce |
| | RDD Relation | E-commerce |
| Streaming Application | Twitter | Twitter |
| | Page review | PageView DataGen |

Table 1: SparkBench Workloads

| Data sets | Description |
|---|---|
| Wikipedia | 6,938,018 Articles |
| Google Web Graph | 875713 nodes, 5105039 edges |
| Amazon Movie Review | 7,911,684 reviews<br>889,176 movies, 253,059 users |
| E-commerce Transactions | Orders: 38275 entries,8 columns<br>Items: 240332 entries,7 columns |

Table 2: Data sets used by SPARKBENCH.

# Agenda

- Workload Characterization
- Impact of Parameter Configuration
- An End-to-end Example of Running SparkBench

# Application Characterization

# Machine learning – large data sets

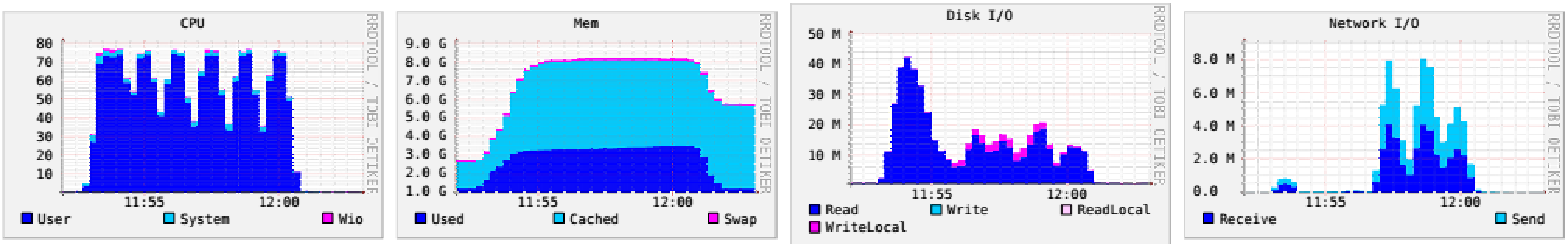



**Figure 3: Resource Consumption of LogisticRegression.**

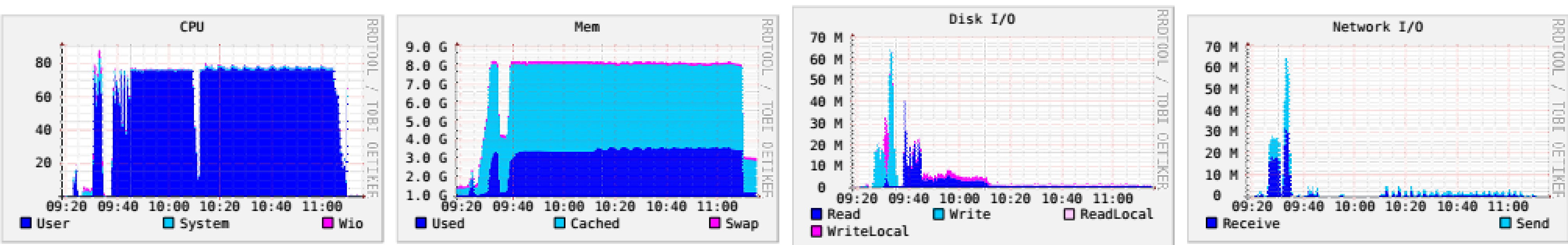



**Figure 4: Resource Consumption of SVM.**

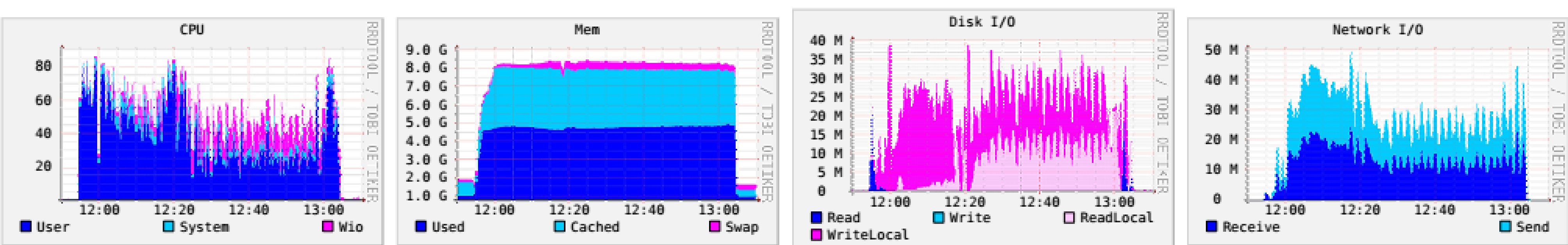



**Figure 5: Resource Consumption of MFAmazonMovie.**

# Machine learning

- bottlenecked resources: CPU
- OS cache has been used extensively
- few disk IO and network IO
- cpu and memory usage are relatively stable

# Graph computation – large data sets

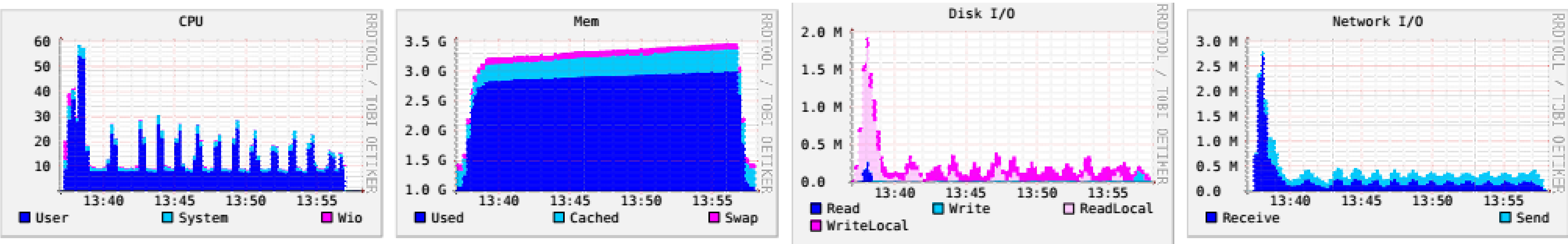



**Figure 6: Resource Consumption of PageRank.**

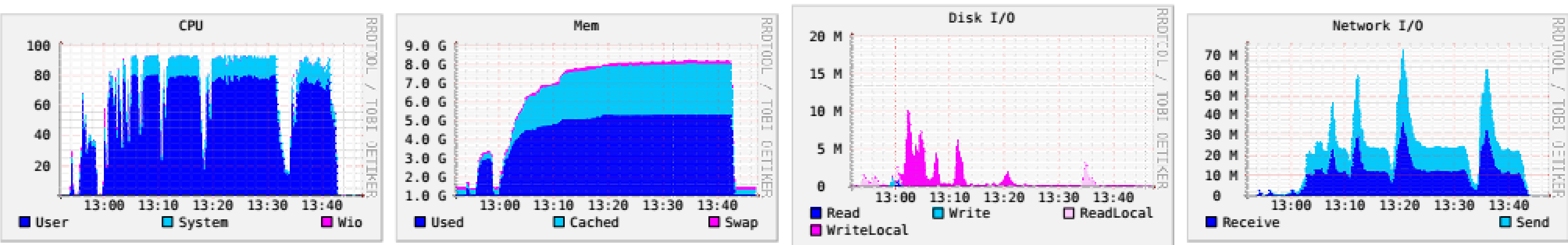



**Figure 7: Resource Consumption of SVDPlusPlusAmazonMovie.**

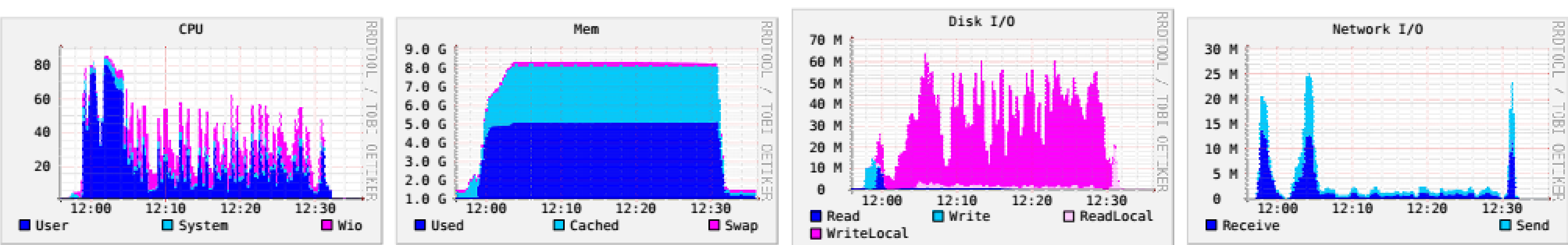



**Figure 8: Resource Consumption of TriangleCount.**

# Graph Computation

- PageRank and triangle count
  - bottlenecked resources: memory
  - CPU and network usage has peaks
- SVD plusplus and shortest path look alike
  - increased memory usage
  - few network IO
  - bursty disk IO

# SQL-like queries – large data sets

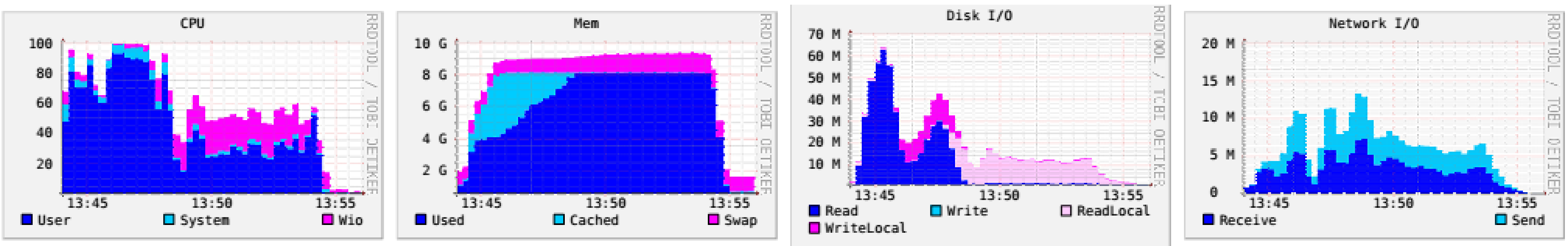



**Figure 9: Resource Consumption of HiveSQL.**

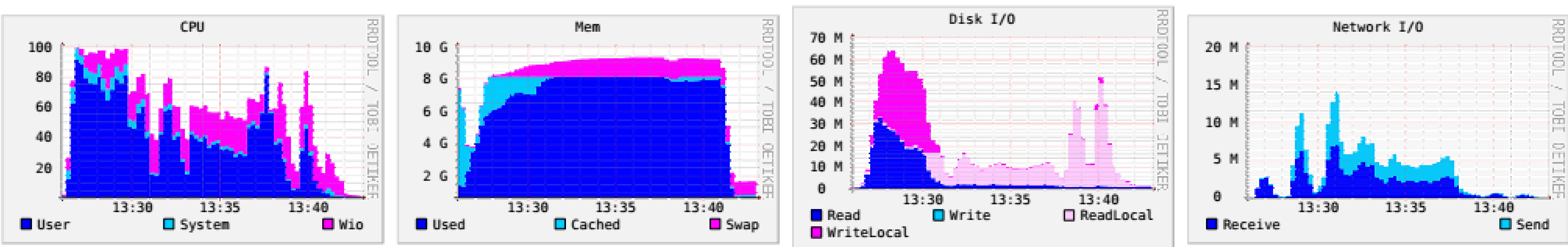



**Figure 10: Resource Consumption of RDDRelation.**

# SQL-like Queries

- Less resource intensive
- suggest to co-run multiple queries to improve system resource utilization

# Streaming applications – large data sets

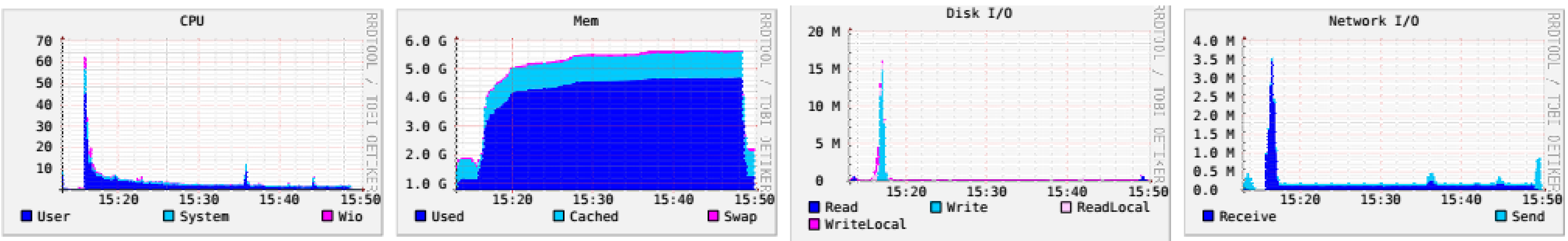



**Figure 11: Resource Consumption of TwitterStreaming.**

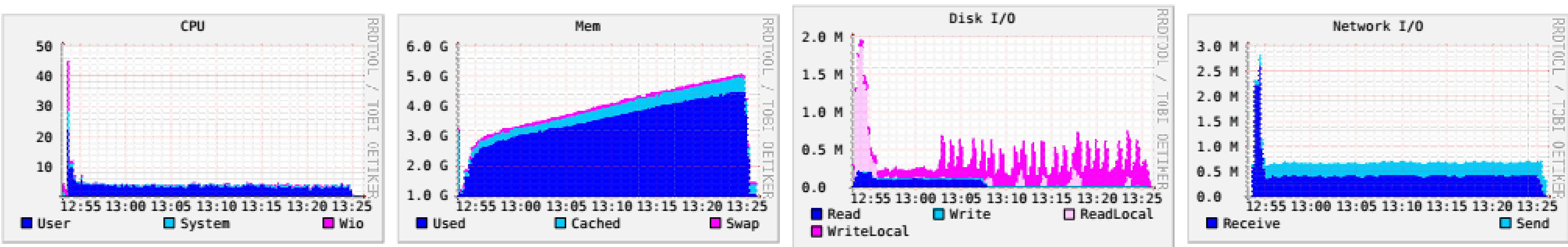
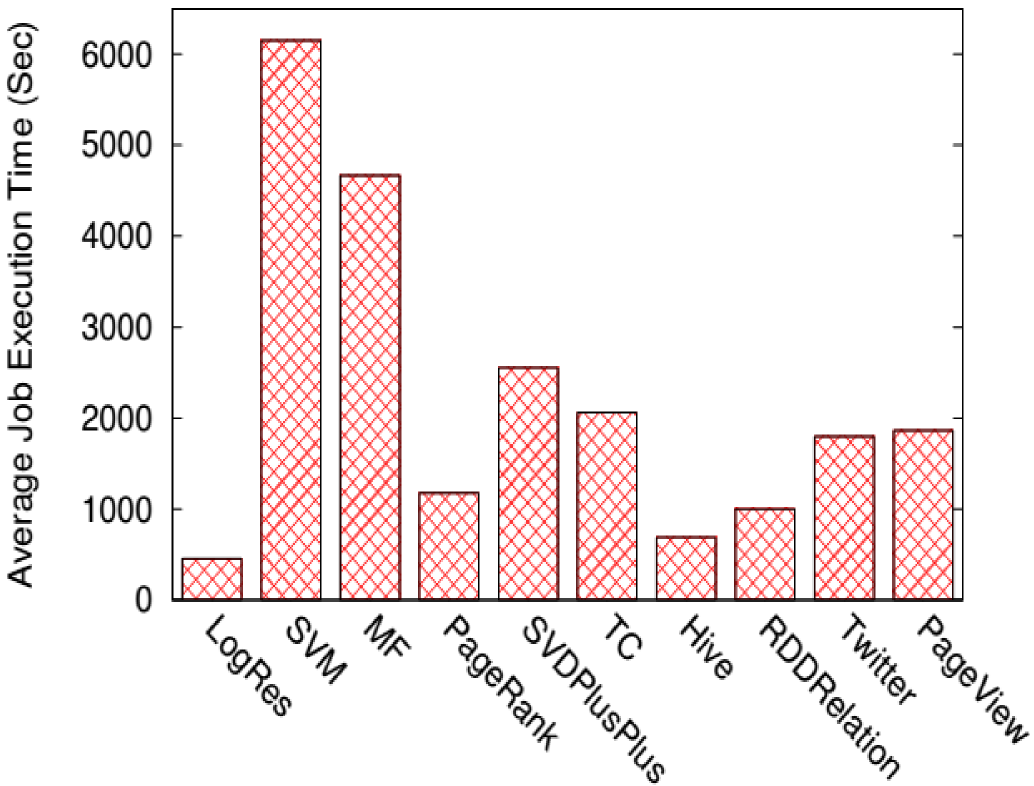


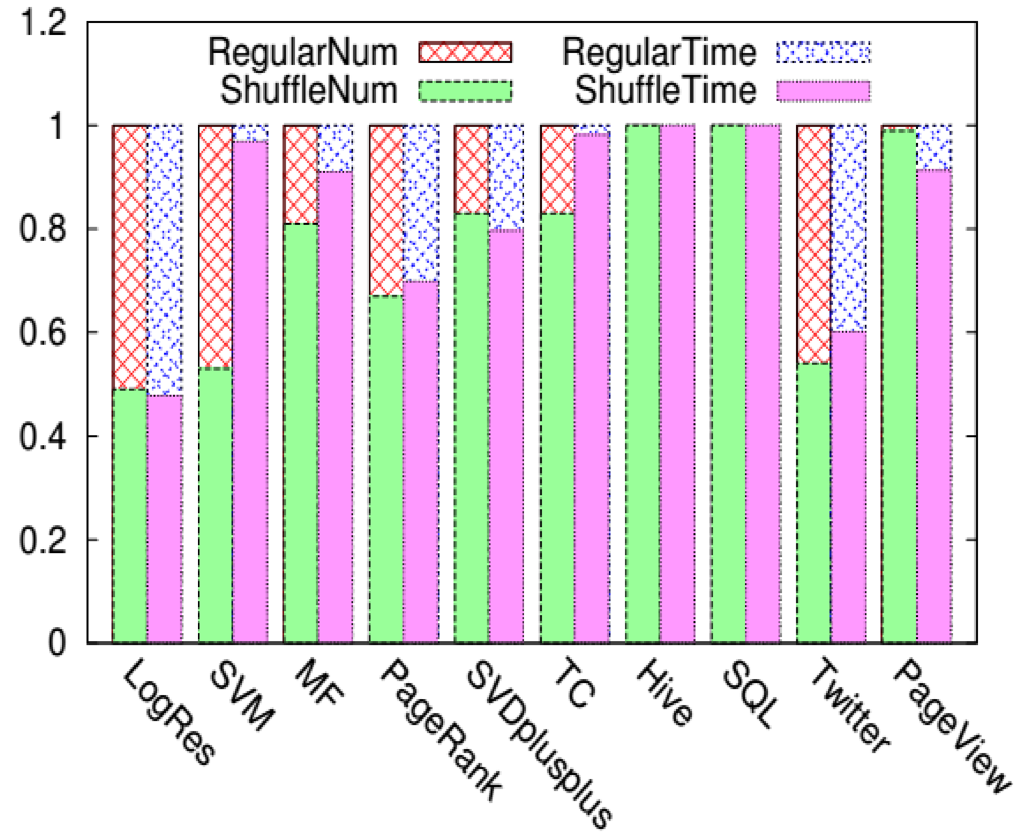

**Figure 12: Resource Consumption of PageViewStreaming.**

# Streaming Applications

- bimodal pattern of CPU utilization
- increased use of memory
- few disk and network IO

# Workload shuffling VS regular tasks



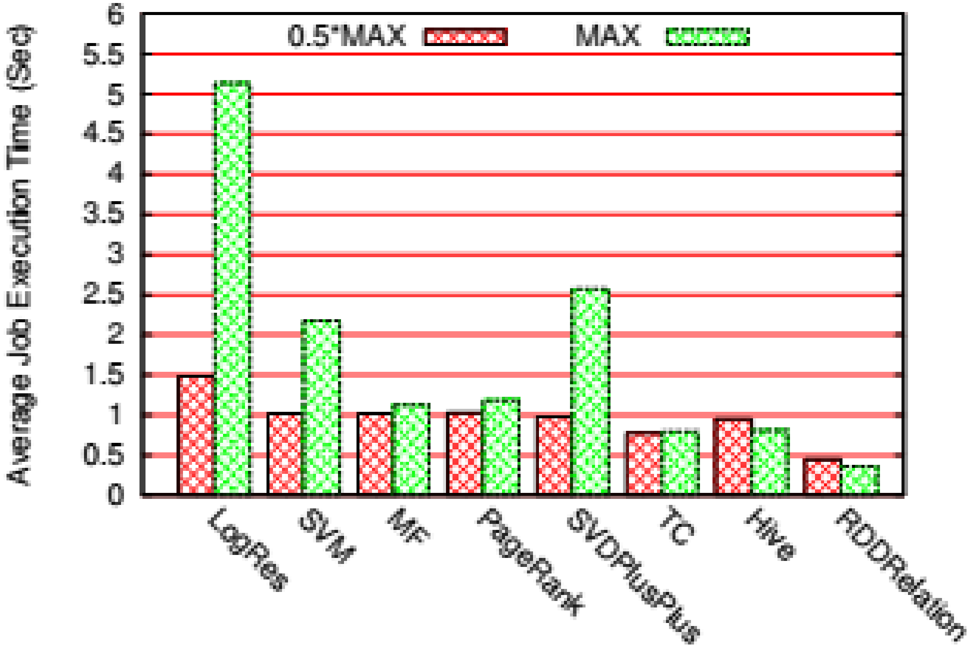



# Workload pattern

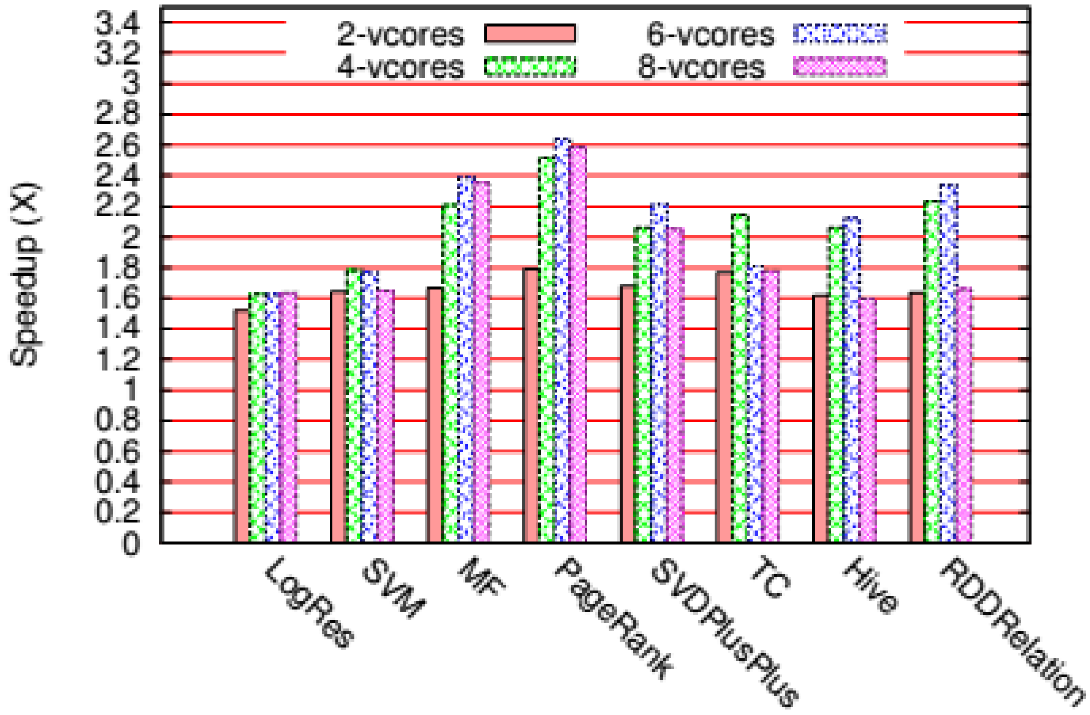
| Workload | Execution Time(Sec) | Input H/M/D/N | Shuffle R/W | Stages/Tasks | Bottlenecked resources |
|---|---|---|---|---|---|
| LogRes | 453 | $162G/23G/0/0.62G$ | 4G/4G | 10/2062 | CPU |
| SVM | 6151 | 223G/7G/0/8M | 27 G/28 G | 23/4410 | CPU |
| MF | 4667 | 45G/23G/370G/21.4G | 524 G/502 G | 87/19K | Shuffle |
| PageRank | 1177 | 71M/7G/0/88M | 708M/713M | 42/17K | Memory |
| SVD++ | 2553 | 198M/26G/1.6G/185M | 310 G/23 G | 22/9111 | CPU |
| TriangleCount | 2059 | 5G/4G/14G/10.6G | 13 G/568 G | 8/4814 | Shuffle |
| Hive | 688 | 39G/291M/29G/0.08G | 51G/56G | 12/9805 | Shuffle |
| RDDRelation | 472 | 39G/0/99.7G/0 | 10.6G/11.8G | 12/9805 | CPU,Shuffle |
| Twitter | 1800 | 0/0/0/0 | 14M/12M | 11K/23K | Memory |
| PageView | 1862 | 0/0/0/0 | 233 M/242 M | 1223/61K | Memory |

# Impact of Spark Configuration

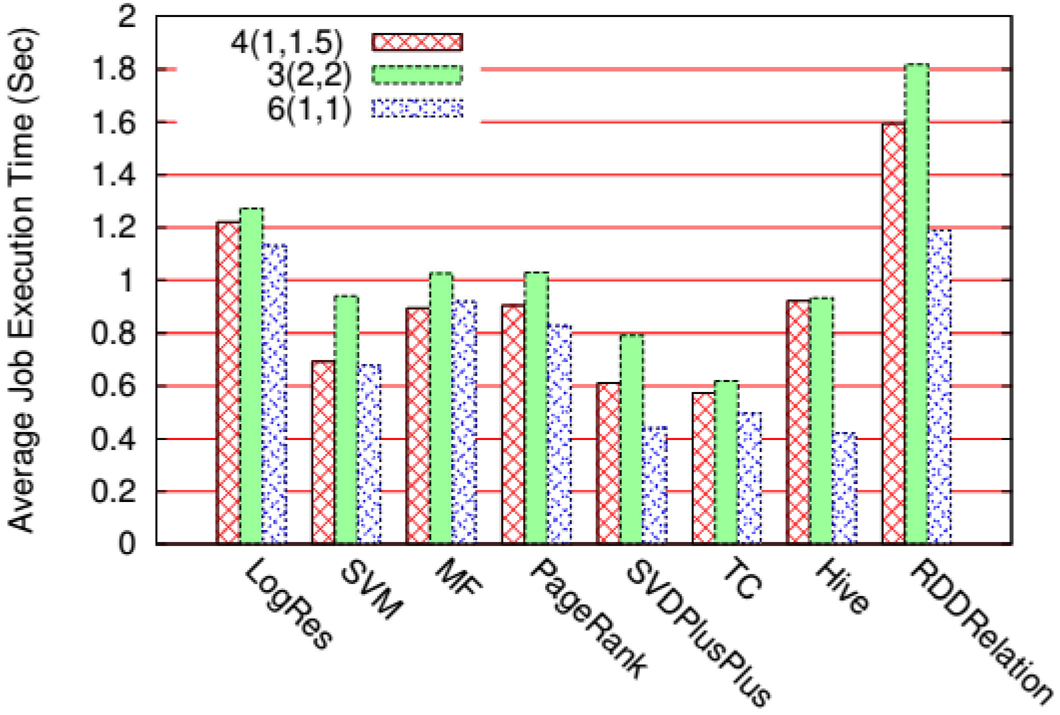
# Impact of RDD cache size



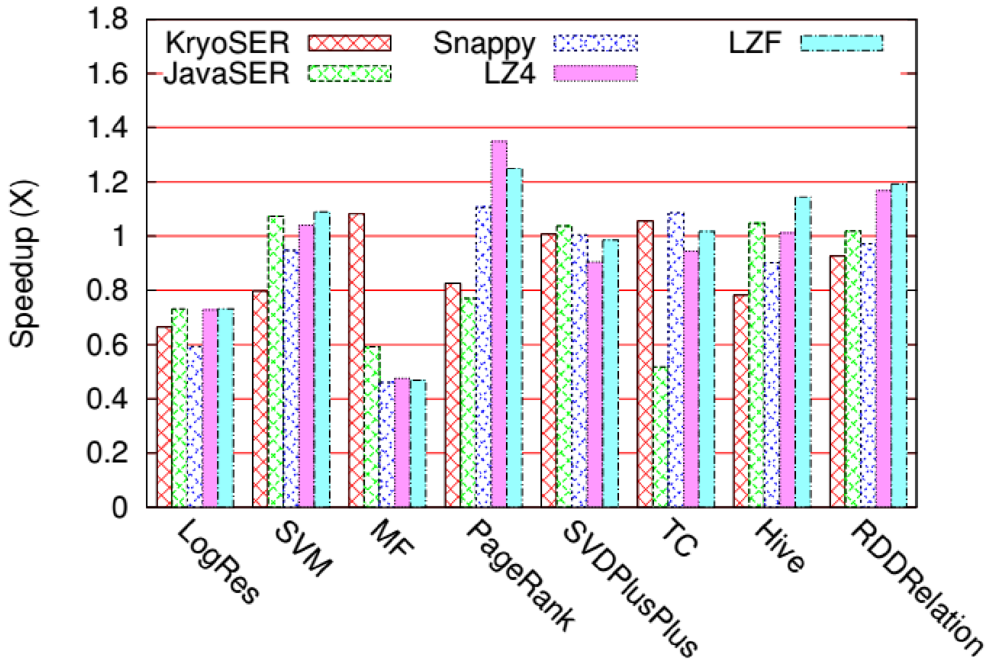
# Impact of task parallellism

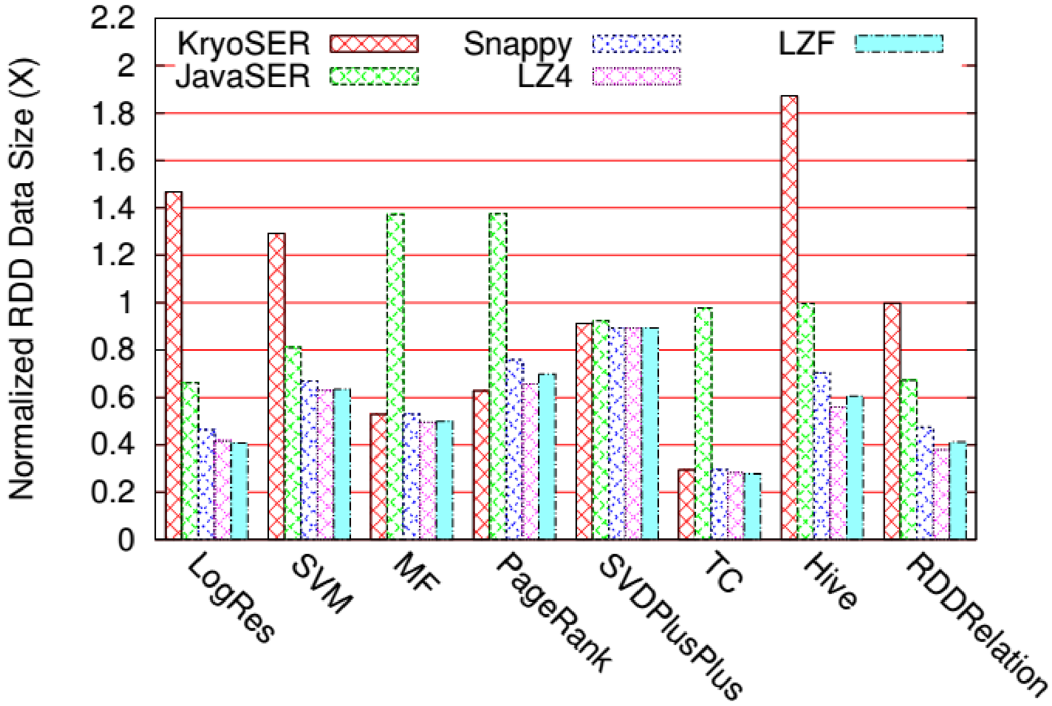


# Impact of executor configuration



# Impact of memory



# Overall Observation

- memory is intensively used across all workloads
  - ShuffleMapTasks use OS cache to store intermediate data
- Machining learning workloads are CPU intensive
- Demand of graph computation workloads varies from different workloads, generally resource intensive
- SQL can be resource demanding
- Streaming workloads demand light resources yet is memory hungry
- While memory usage is stable, other resource usage can be bursty.
- Parameter configuration impact the performance signicantly

# Example

# Example(1/4): Using SparkBench to Study the Impact of Parameter Configuration

- Download the package:
  - git clone https://bitbucket.org/lm0926/sparkbench
- Setup spark cluster, optionally Ganglia
- configure bin/config.sh file to point to the Spark master
- To run each workload individually,
  - cd to the workload directory
  - mvn package/ sbt package
  - bin/gen_data.sh
  - bin/run.sh

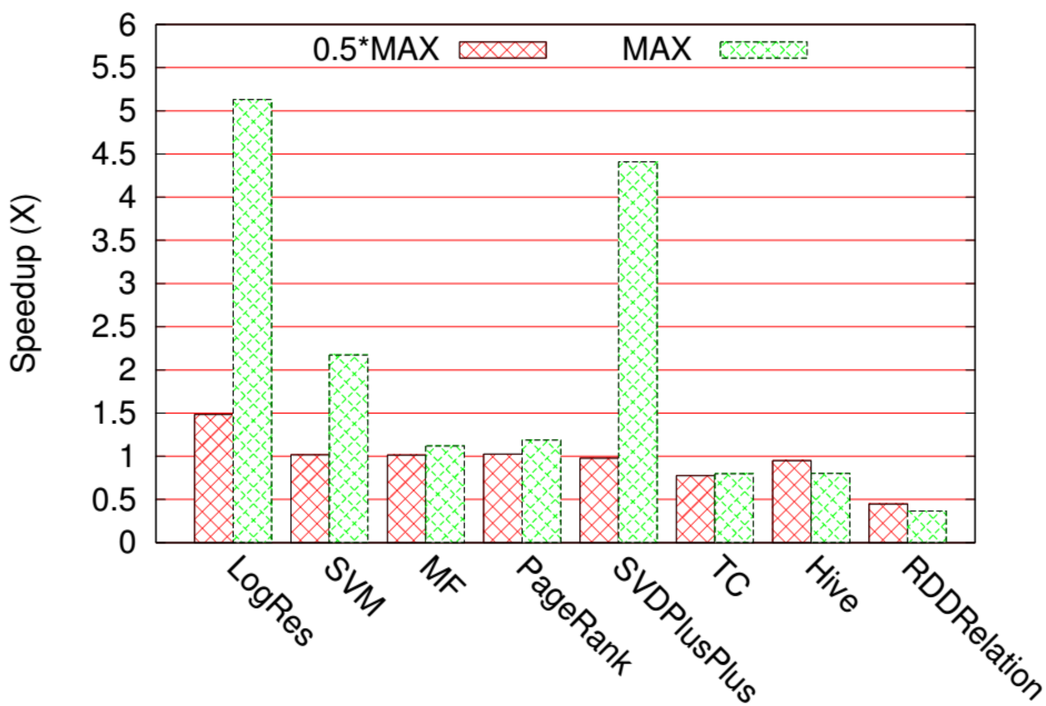
# Example(2/4): Using SparkBench to Study the Impact of Parameter Configuration

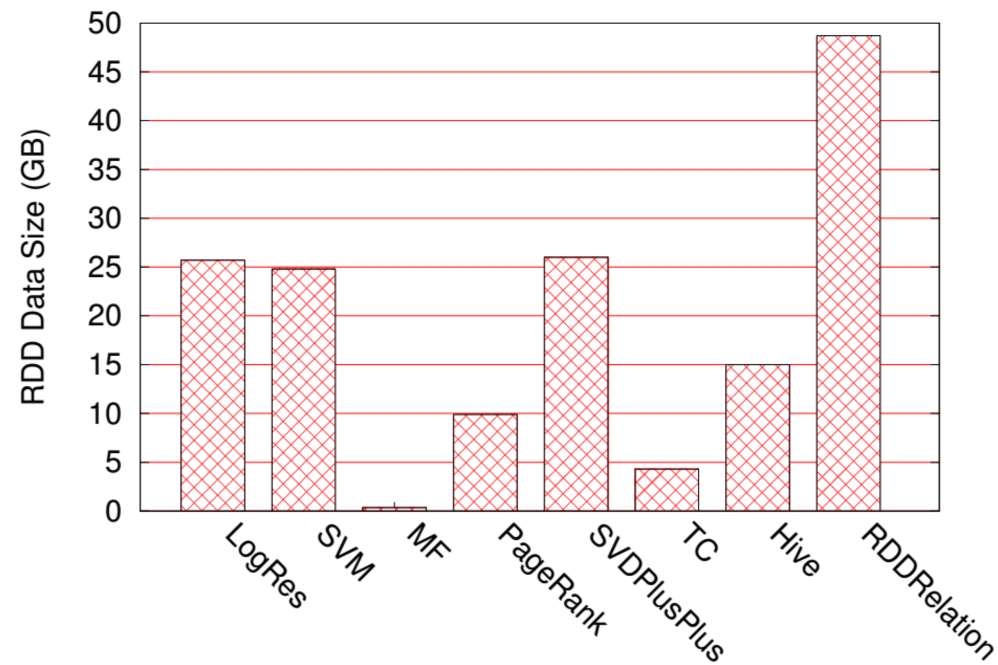
- configuration of SparkBench
  - bin/config.sh,
    - change memoryFraction to zero  or different values
  - [workload]/bin/config.sh
    - specify the workload parameters such as number of iterations, the number of points generated in the input dataset.
- to view result in the bench.report
  - reports the job execution time, the data process rate, the input data set size

# Example (3/4) : A Screenshot

```
[limin@minli1 SparkBench]$ ls -d */
common/              DecisionTree/  kmeans_java/       LinearRegression/    MatrixFactorization/  num/       PregelOperati
on/  sql/        StronglyConnectedComponent/  SVM/ bin/        ConnectedComponent/  kmeans/        LabelPropagation/  Logi
sticRegression/  PageRank/  ShortestPaths/    streaming/  SVDPlusPlus/     TriangleCount/

[limin@minli1 LogisticRegression]$ bin/gen_data.sh;
========== preparing LogisticRegression data ==========
Deleted /Bench/LogisticRegression/Input
Spark assembly has been built with Hive, including Datanucleus jars on classpath
15/03/17 12:49:09 INFO Slf4jLogger: Slf4jLogger started
15/03/17 12:49:09 INFO Remoting: Starting remoting
15/03/17 12:49:09 INFO Remoting: Remoting started; listening on addresses :[akka.tcp://sparkDriver@minli1.sl.cloud9.ibm.co
m:39579]
15/03/17 12:49:09 INFO RemoteActorRefProvider$RemotingTerminator: Shutting down remote daemon.
15/03/17 12:49:09 INFO RemoteActorRefProvider$RemotingTerminator: Remote daemon shut down; proceeding with flushing remote
 transports.
15/03/17 12:49:09 INFO RemoteActorRefProvider$RemotingTerminator: Remoting shut down.

[limin@minli1 LogisticRegression]$ bin/run.sh
========== running LogisticRegression bench ==========
rm: `/Bench/LogisticRegression/Output': No such file or directory
Spark assembly has been built with Hive, including Datanucleus jars on classpath
15/03/17 13:05:08 INFO Slf4jLogger: Slf4jLogger started
15/03/17 13:05:08 INFO Remoting: Starting remoting
15/03/17 13:05:08 INFO Remoting: Remoting started; listening on addresses :[akka.tcp://sparkDriver@minli1.sl.cloud9.ibm.co
m:60481]
15/03/17 13:05:13 INFO FileInputFormat: Total input paths to process : 720
training Mean Squared Error = 0.036943336666666764
15/03/17 13:07:22 INFO RemoteActorRefProvider$RemotingTerminator: Shutting down remote daemon.
15/03/17 13:07:22 INFO RemoteActorRefProvider$RemotingTerminator: Remote daemon shut down; proceeding with flushing remote
 transports.
15/03/17 13:07:22 INFO RemoteActorRefProvider$RemotingTerminator: Remoting shut down.

[limin@minli1 spark_app]$ cat bench.report
LogisticRegression-gendata 2015-03-17-12:49:35 114.771000 32822.729247 285.984519
LogisticRegressionConfig nexample 400000000 nCluster 4 EPS 0.5 npar 720 ProbOne 0.2 niter 3 memoryFraction 0.79
LogisticRegression 2015-03-17-13:05:33 137.848000 24912.460407 180.724133
LogisticRegressionConfig nexample 400000000 nCluster 4 EPS 0.5 npar 720 ProbOne 0.2 niter 3 memoryFraction 0.79
```

# Example(4/4): Visualizing the Impact of RDD Cache Size on Job Execution Time



# Conclusion

- SparkBench is a comprehensive Spark-specific benchmarking suite
  - easy to use
  - can be used for various scenarios: performance comparison, cluster provisioning, in-depth study of Spark

Available for download:

https://bitbucket.org/lm0926/sparkbench