PRESTO

Distributed Machine Learning and Graph Processing with Sparse Matrices

Shivaram Venkataraman*, Erik Bodzsar#
Indrajit Roy+, Alvin AuYoung+, Rob Schreiber+

*UC Berkeley, #U Chicago, +HP Labs

Big Data, Complex Algorithms

PageRank
(Dominant eigenvector)

Recommendations

Machine learning + Graph algorithms

Anomaly detection
(Top-K eigenvalues)

User Importance
(Vertex Centrality)

Large-Scale Processing Frameworks

Data-parallel frameworks – MapReduce/Dryad (2004)

- Process each *record* in parallel
- Use case: Computing sufficient statistics, analytics queries

Graph-centric frameworks – Pregel/GraphLab (2010)

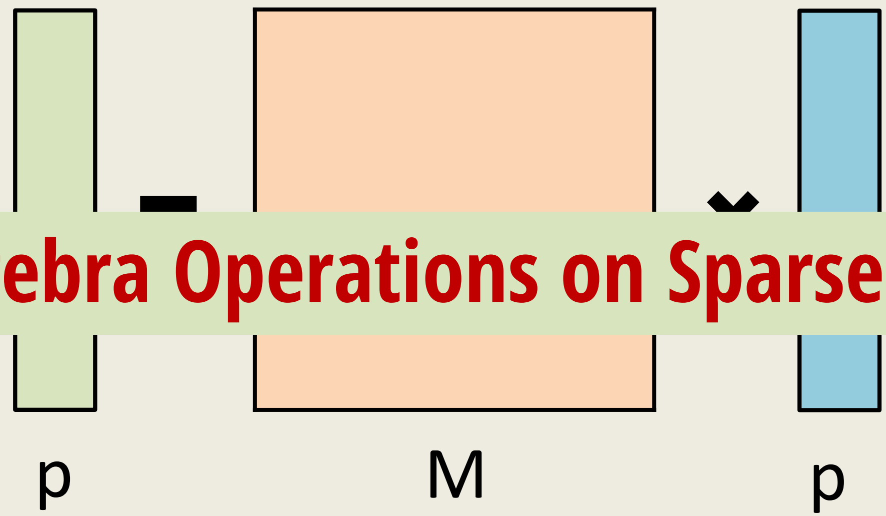
- Process each *vertex* in parallel
- Use case: Graphical models

Array-based frameworks – MadLINQ (2012)

- Process *blocks* of array in parallel
- Use case: Linear Algebra Operations

PageRank using Matrices

Simplified algorithm repeat { p = M*p }



Linear Algebra Operations on Sparse Matrices

Power Method
Dominant eigenvector

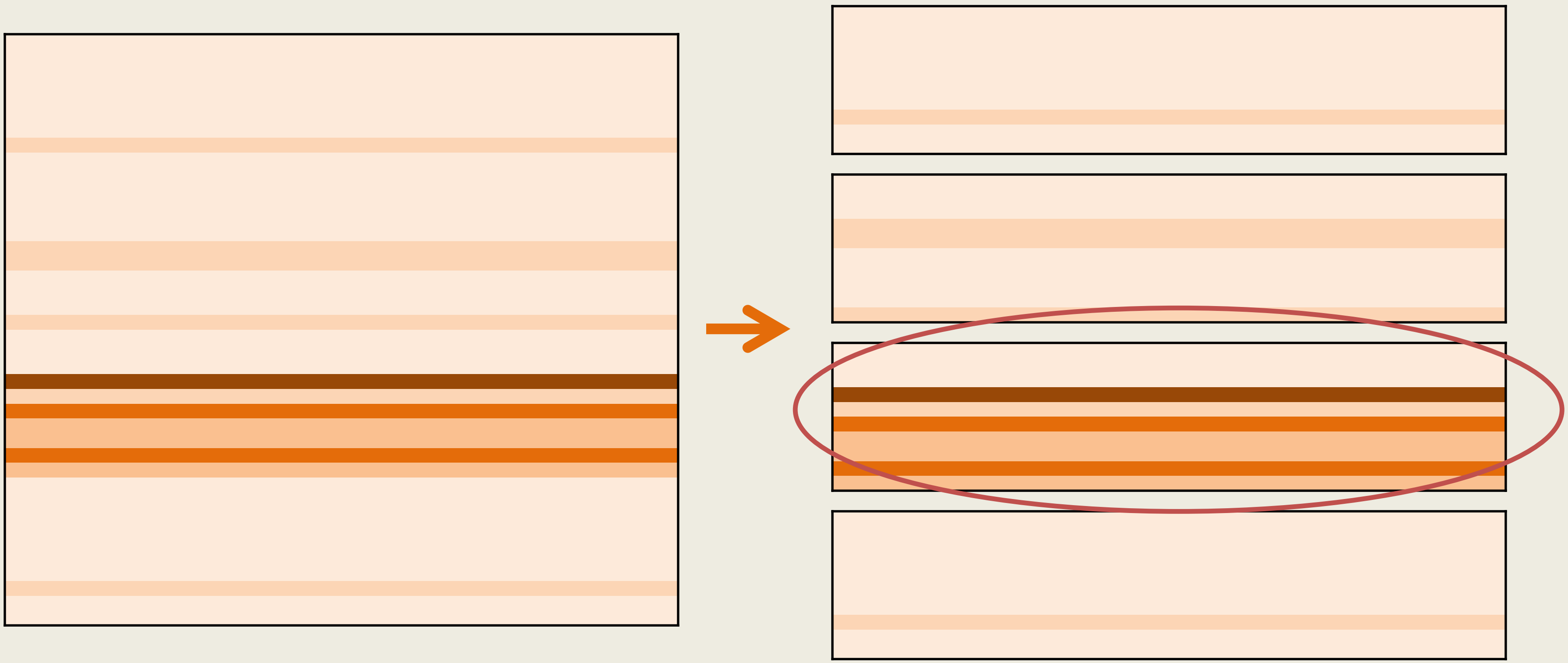
M = web graph matrix
p = PageRank vector

Presto

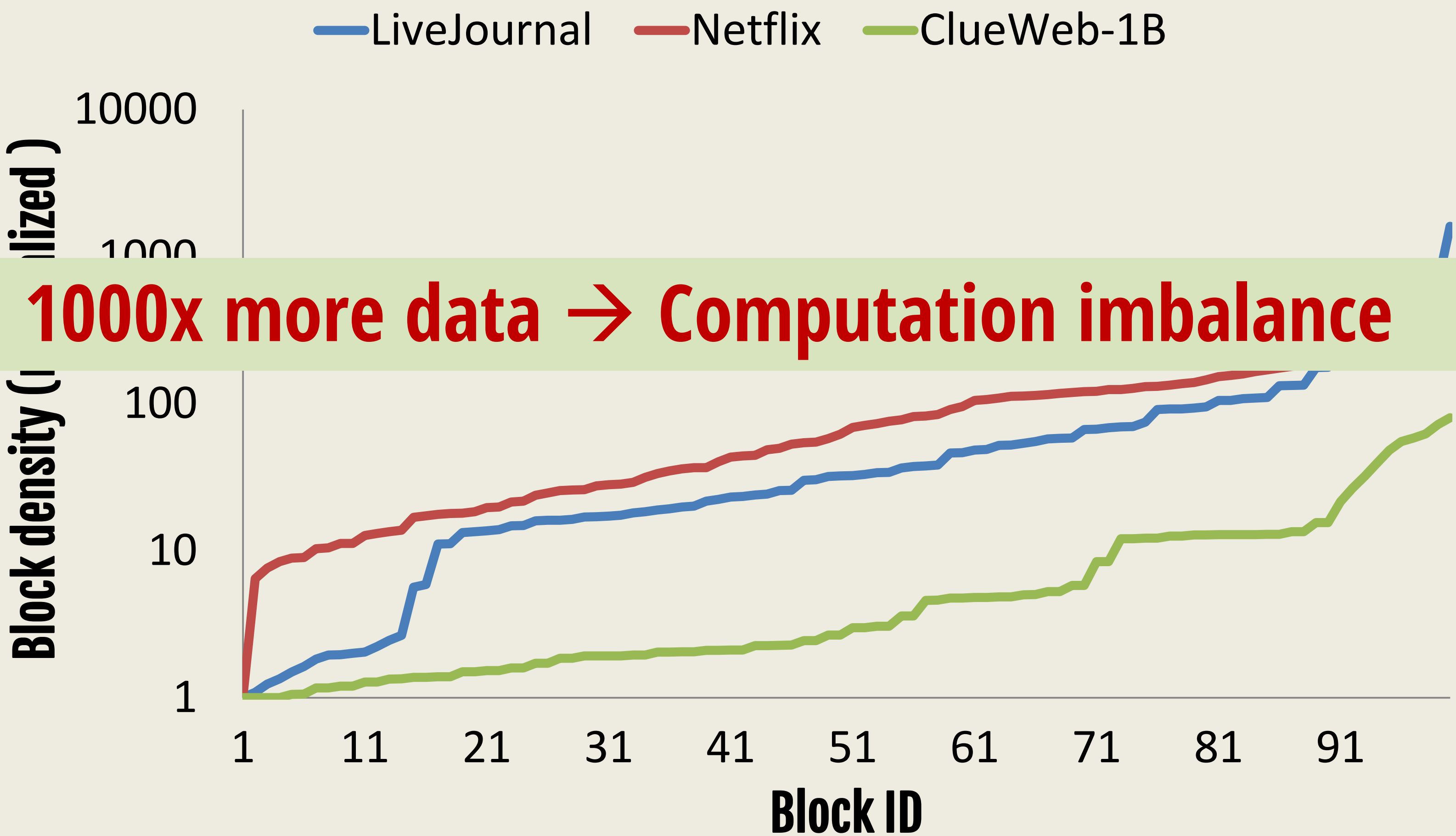
Large-scale machine learning and graph processing on sparse matrices

Extend R – make it scalable, distributed

Challenge 1 – Sparse Matrices



Challenge 1 – Sparse Matrices



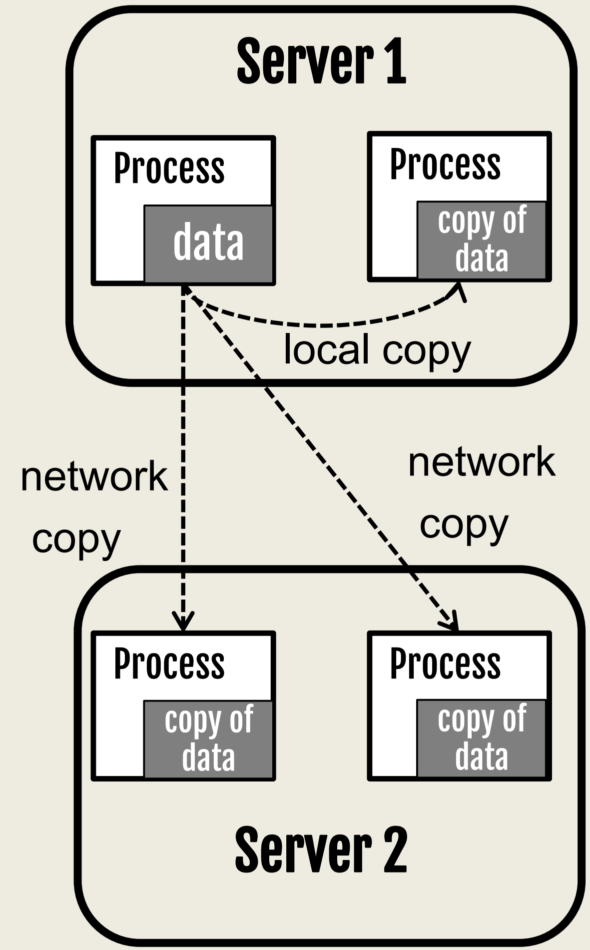
1000x more data → Computation imbalance

Challenge 2 – Data Sharing

Sparse matrices →
Communication overhead

Sharing data through pipes/network

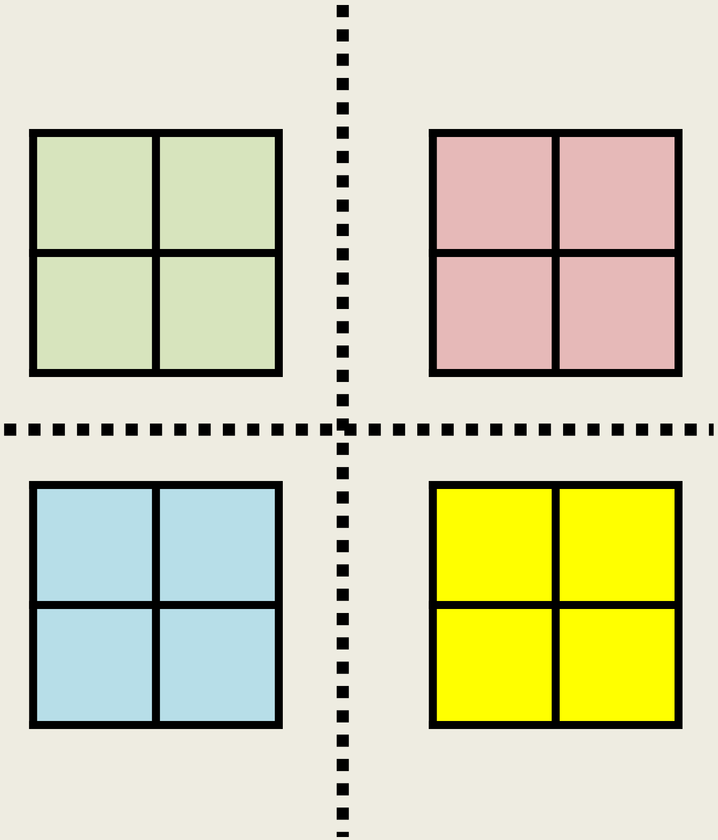
Time-inefficient (sending copies)
Space-inefficient (extra copies)



Outline

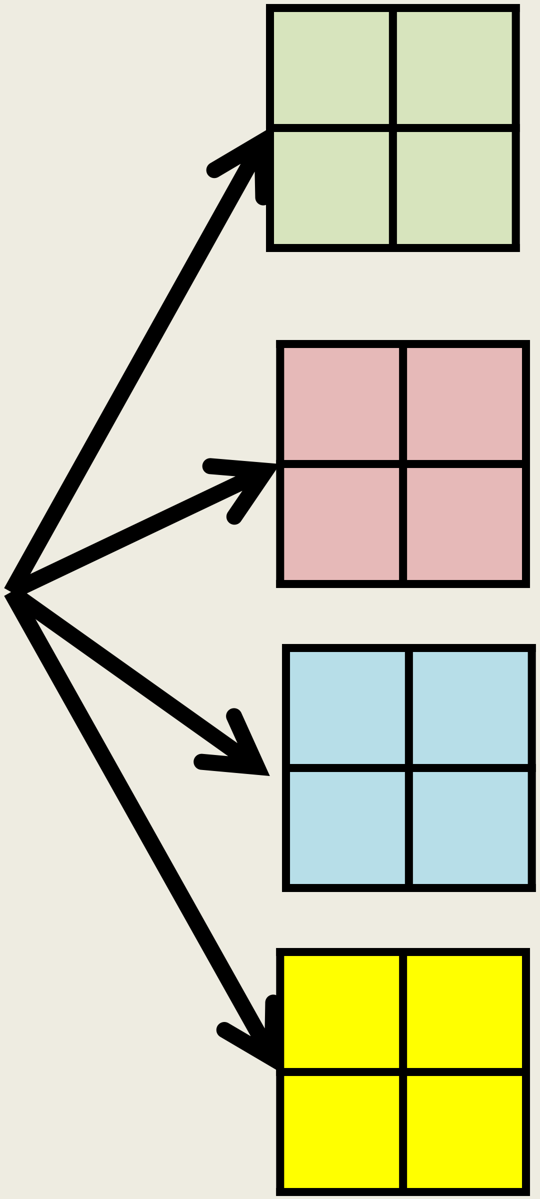
- Motivation
- Programming model
- Design
- Applications and Results

darray

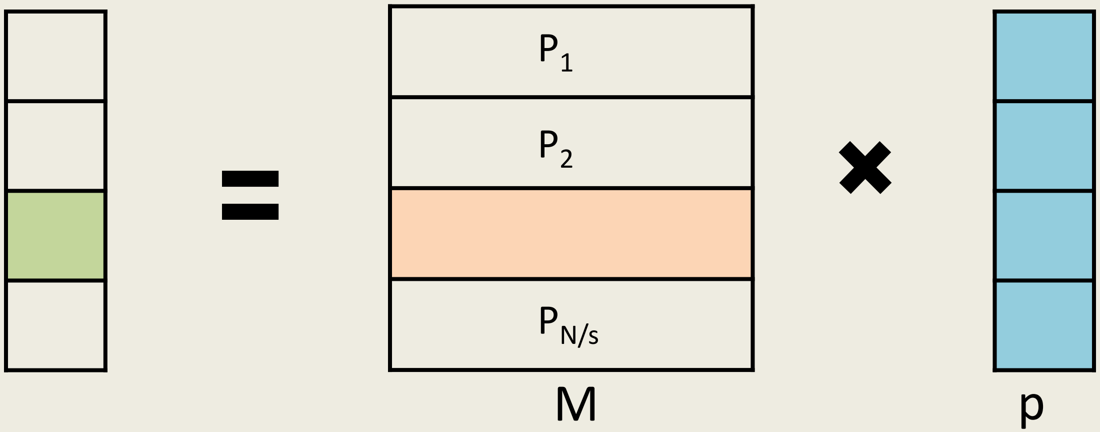


foreach

$f(x)$



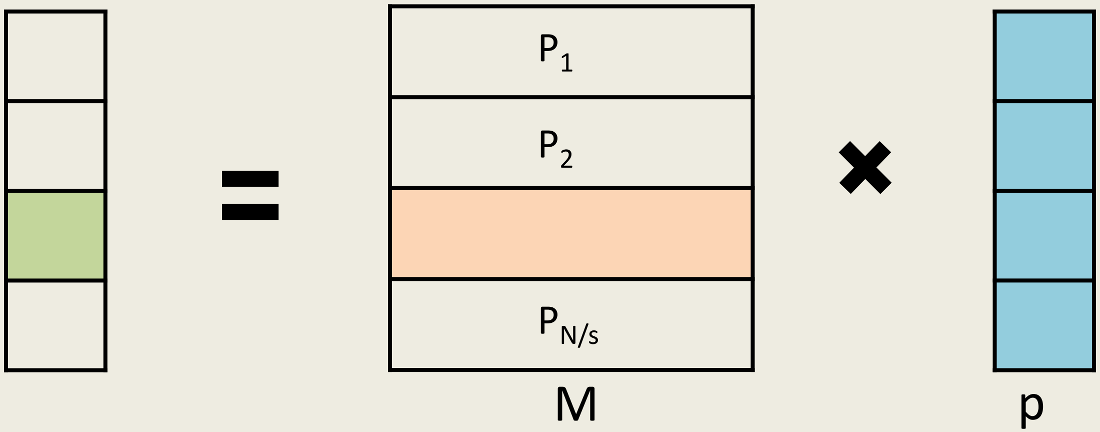
PageRank Using Presto




```
M ← darray(dim=c(N,N),blocks=(s,N))
P ← darray(dim=c(N,1),blocks=(s,1))
while(..){
  foreach(i,1:len,
     calculate(m=splits(M,i),
               x=splits(P), p=splits(P,i)) {
        p ← m*x
     }
)}
```

Create Distributed Array

PageRank Using Presto

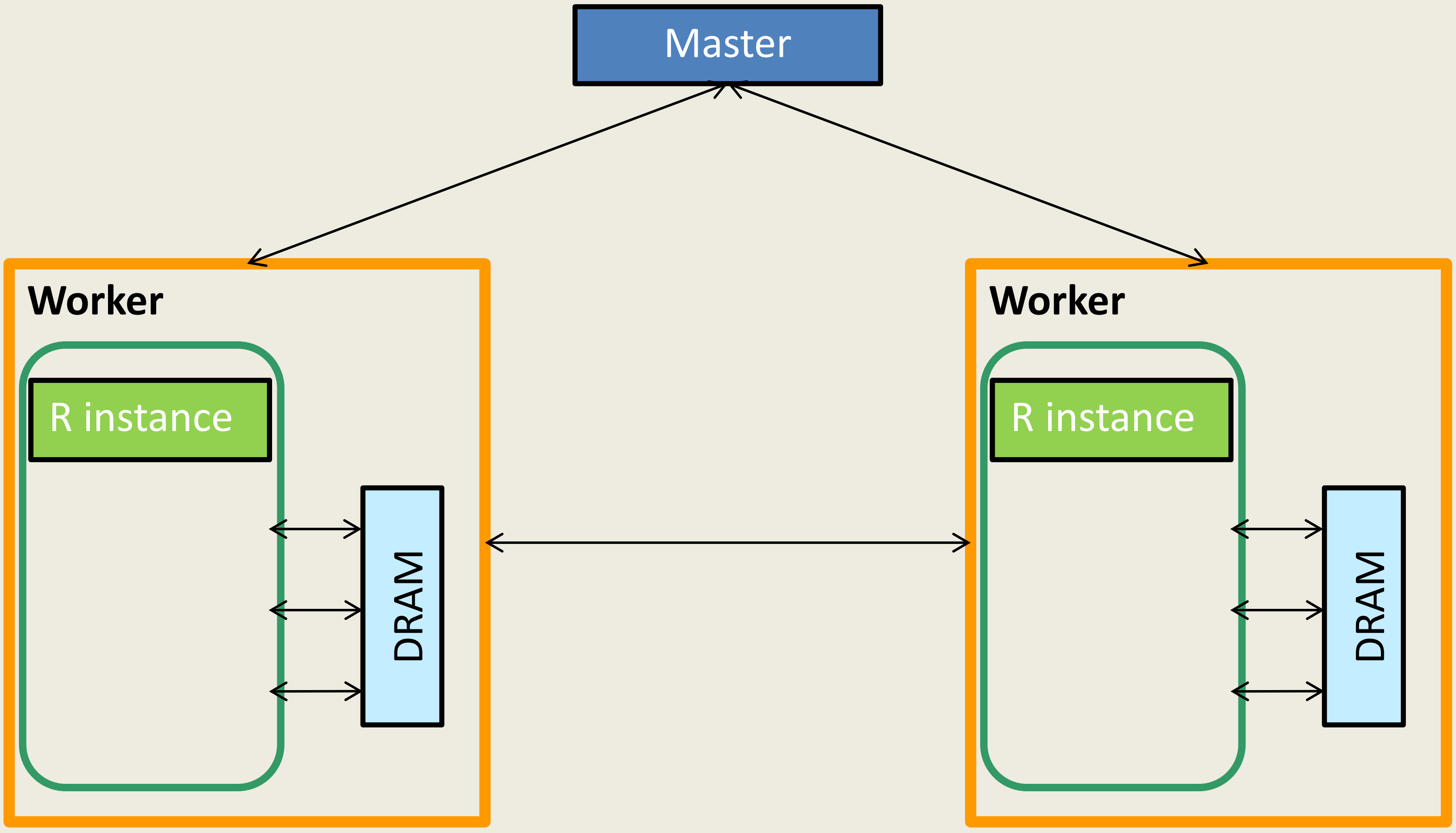



```
M ← darray(dim=c(N,N),blocks=(s,N))
P ← darray(dim=c(N,1),blocks=(s,1))
while(..){
  foreach(i,1:len,
     calculate(m=splits(M,i),
               x=splits(P), p=splits(P,i)) {
         p ← m*x
     }
)}
```

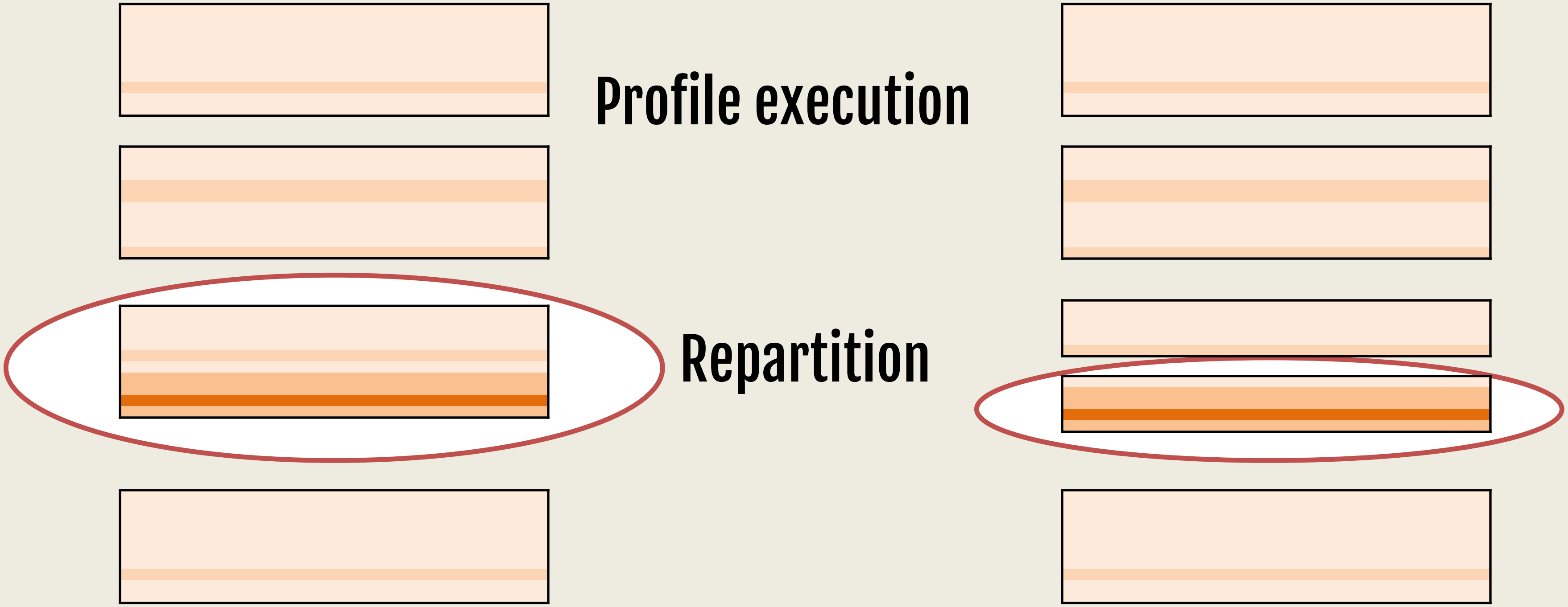
Execute function in a cluster

Pass array partitions

Presto Architecture



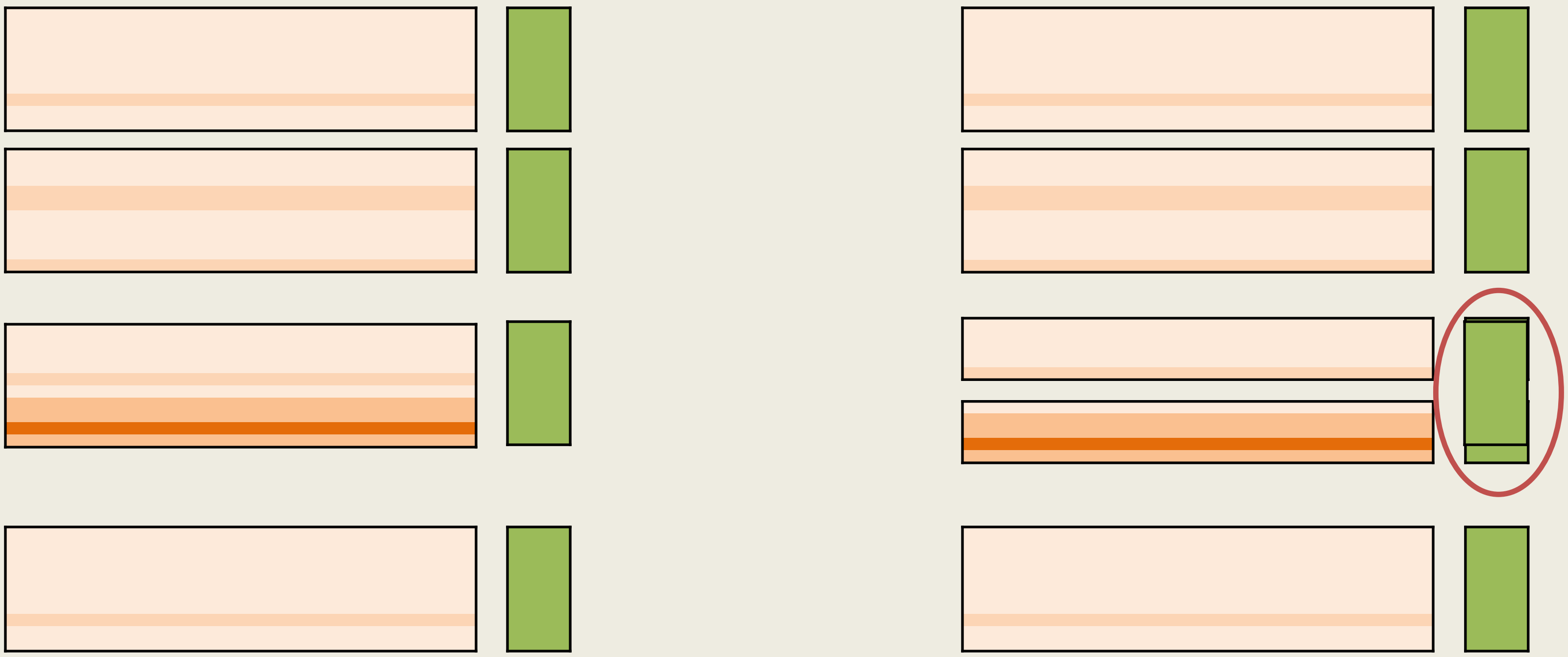
Repartitioning Matrices



Partition if $\frac{\max(t)}{median\ (t)} > \delta$

Maintaining Size Invariants

```
invariant(mat,vec, type=ROW)
```



Sharing Distributed Arrays

Goal: Zero-copy sharing across cores

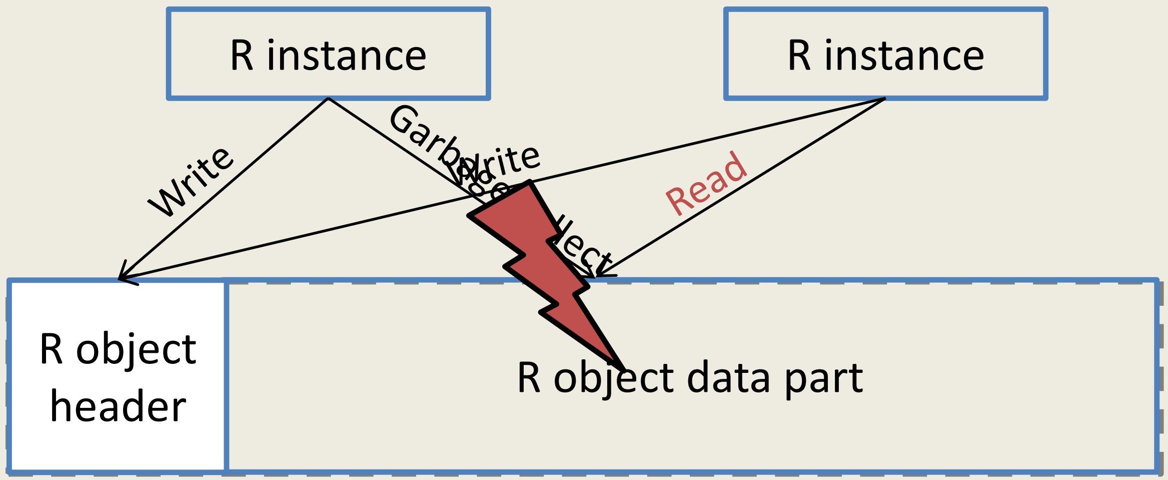
Immutable partitions → Safe sharing

Versioned distributed arrays

Data Sharing Challenges

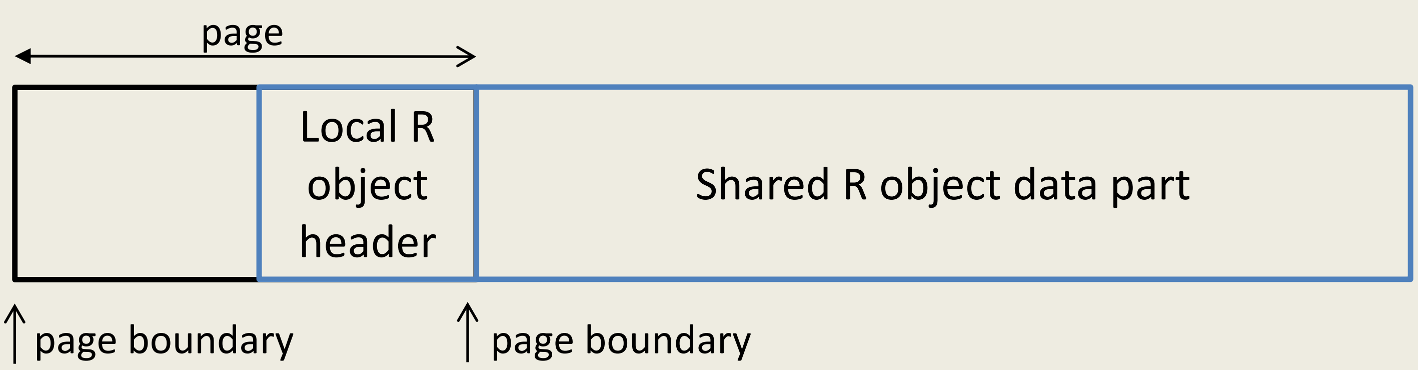
1. Garbage collection

2. Header conflicts



Overriding R’s allocator

Allocate process-local headers. Map data in shared memory



Outline

- Motivation
- Programming model
- Design
- Applications and Results

demo

demo

demo

demo

5 node cluster **8 cores per node**

PageRank on 1.5B edge Twitter data


bigten.hpl.hp.com - PuTTY
venkshiv@110-1:~/presto-demo$ R

Applications Implemented in Presto

Application	Algorithm	Presto LOC
PageRank	Eigenvector calculation	41
Triangle counting	Top-K eigenvalues	121
Centrality measure	Graph algorithm	132
k-path connectivity	Graph algorithm	30
k-means	Clustering	71
Sequence alignment	Smith-Waterman	64

Fewer than 140 lines of code

Evaluation Overview

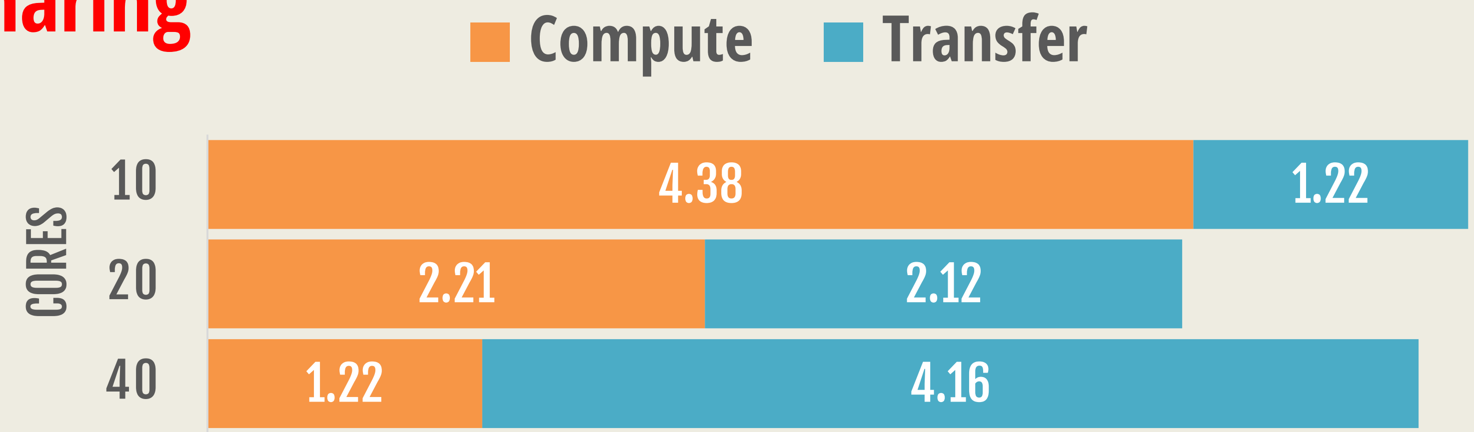
Evaluation Setup
- 25 machine cluster
- Machine: 24 cores, 96GB RAM, 10Gbps network

Data-sharing benefits – 1.5B edge Twitter graph
Repartitioning analysis – 6B edge Web-graph

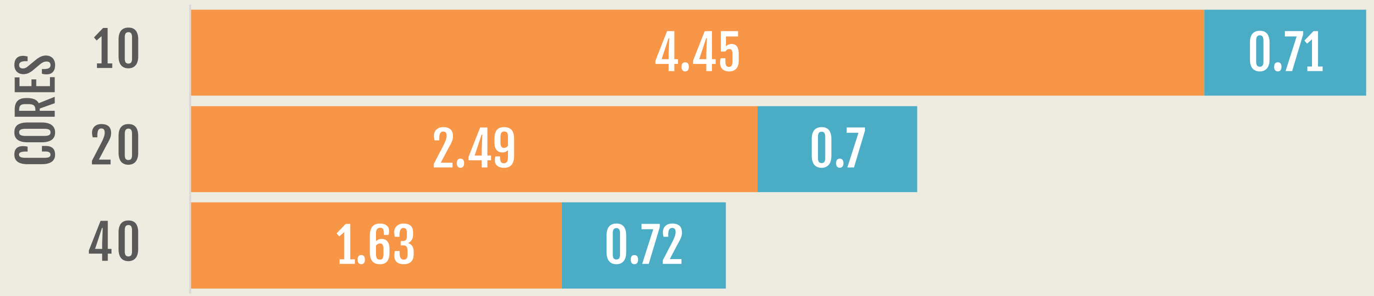
Faster than Spark and Hadoop using in-memory data
Collaborative Filtering using Netflix dataset

Data sharing benefits

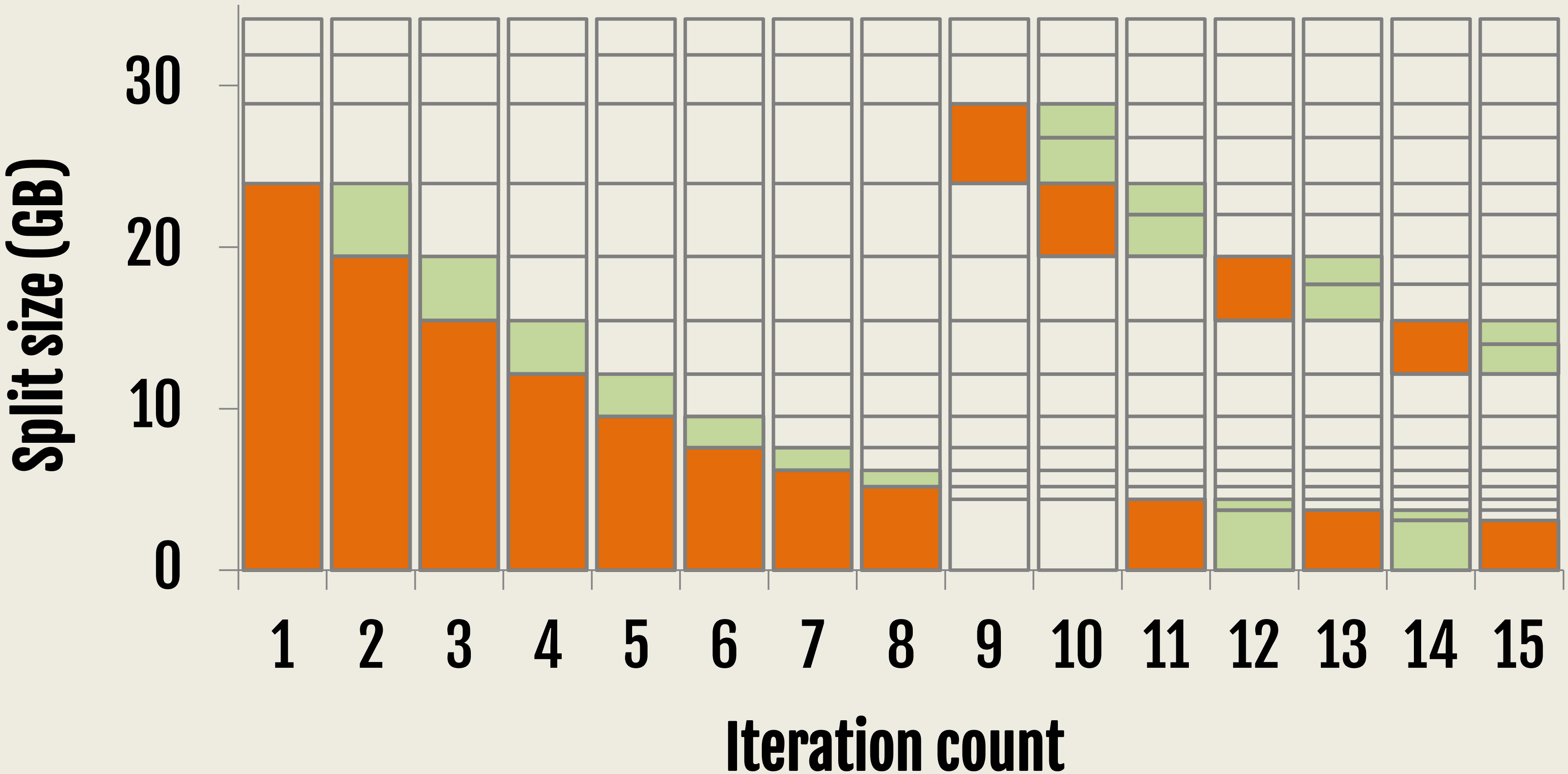
No sharing



Sharing

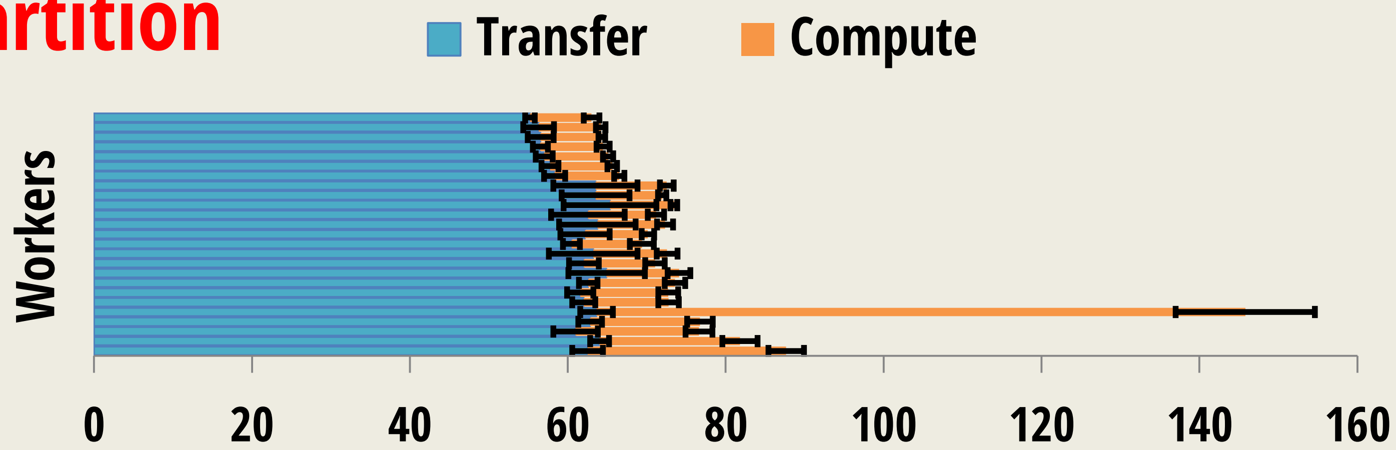


Repartitioning Progress

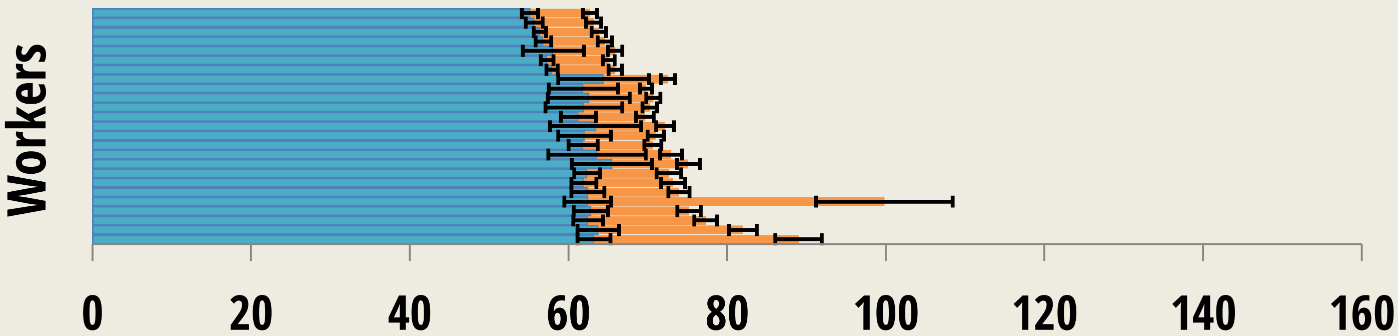


Repartitioning benefits

No Repartition



Repartition



Related Work

Large scale data processing frameworks
- MapReduce, Dryad, Spark, GraphLab

Matrix Computations – Ricardo, MadLINQ

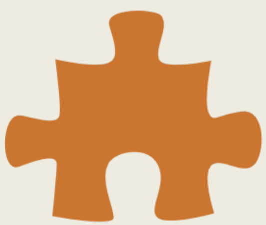
HPC systems – ARPACK, Combinatorial BLAS

Multi-core R packages – doMC, snow, Rmpi

Locality-based scheduling

Presto

Caching partitions



Co-partitioning matrices

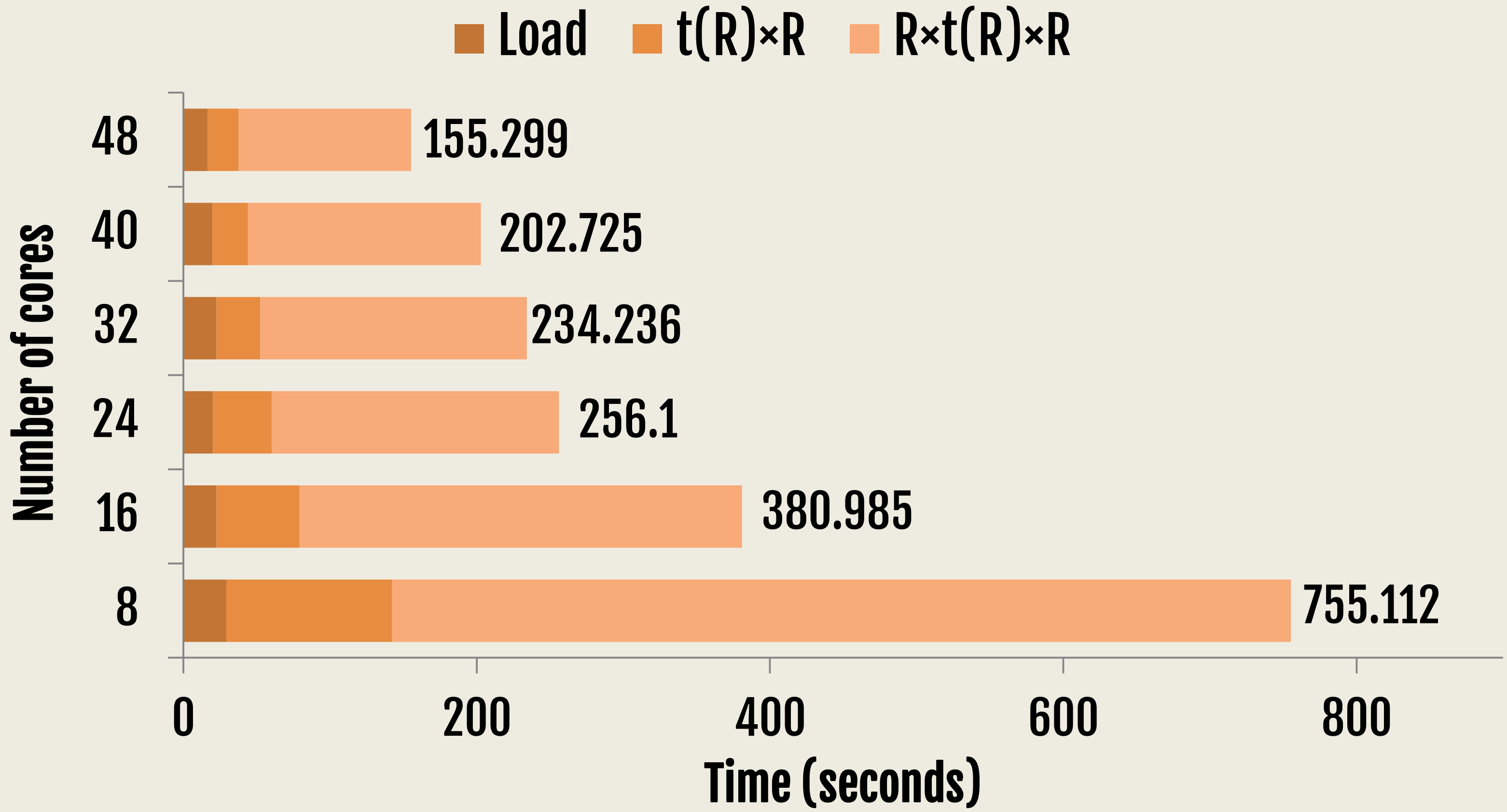
Conclusion

Presto: Large scale array-based framework extends R

Challenges with Sparse matrices
Repartitioning, sharing versioned arrays

Backup Slides

Netflix Collaborative Filtering



Repartitioning benefits