



# BabelMR: A Polyglot Framework for Serverless MapReduce

Fabian Mahling, Paul Rößler, **Thomas Bodner**, Tilmann Rabl

thomas.bodner@hpi.de

SDA Workshop | 1 September 2023

# You are a data professional

- Build and run a complex data pipeline at scale

- Requirements:
  1. Pipeline stages in different languages, environments, and data engines
  2. Enable distributed data-parallel execution
  3. Operate underlying infrastructure

- What do you do with little expertise in… ?
  1. Porting applications to distributed computing frameworks
  2. Distributed and parallel data processing
  3. Cluster management

# BabelMR: A Serverless System for Arbitrary Containerized MapReduce Applications

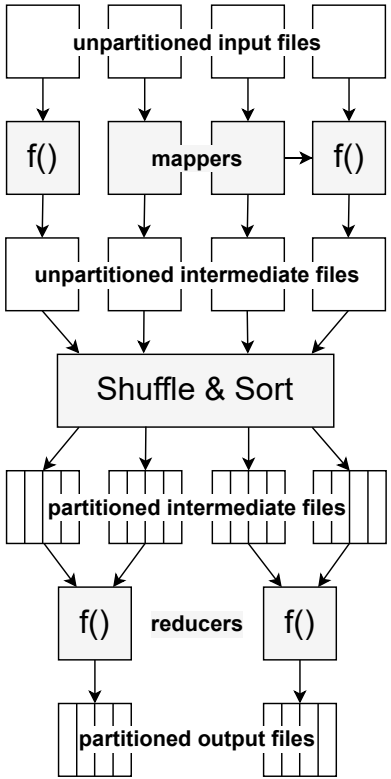
- Wraps arbitrary containerized applications to maximize portability
  - Applications are black boxes to the system
- Exposes MapReduce programming model to simplify data-parallel execution
  - Pragmatic choice over richer models
- Builds on serverless infrastructure to simplify cluster management
  - Current commercial public clouds (AWS, Azure, GCP, …)

# Agenda

- Primer
  - The MapReduce Programming Model
  - Function as a Service Platforms
- BabelMR
  - Programming Interface
  - System Architecture
  - Microbenchmarks of Function Containers and Filesystems
- Evaluation
  - BabelMR Building Blocks
  - Systems Comparison

# The MapReduce Programming Model

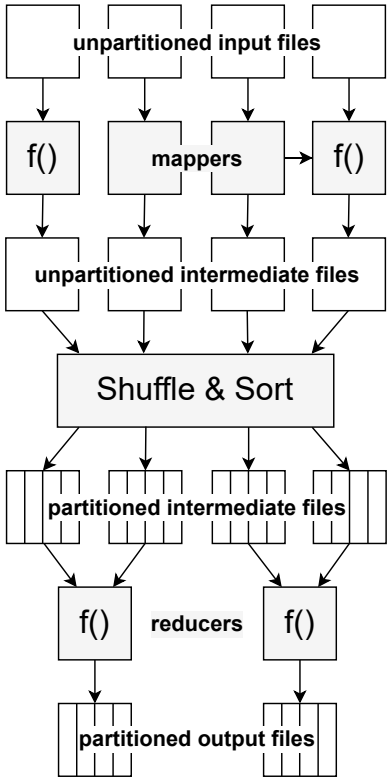
- Inspired by functional programming
- Higher order functions (map, reduce, …) call user-defined lower order functions
- All functions are side effect free
- Simple parallelization model for arbitrary computations



# The MapReduce Programming Model

- Inspired by functional programming
- Higher order functions (map, reduce, …) call user-defined lower order functions
- All functions are side effect free
- Simple parallelization model for arbitrary computations

*"The user just has to setup the cluster"*



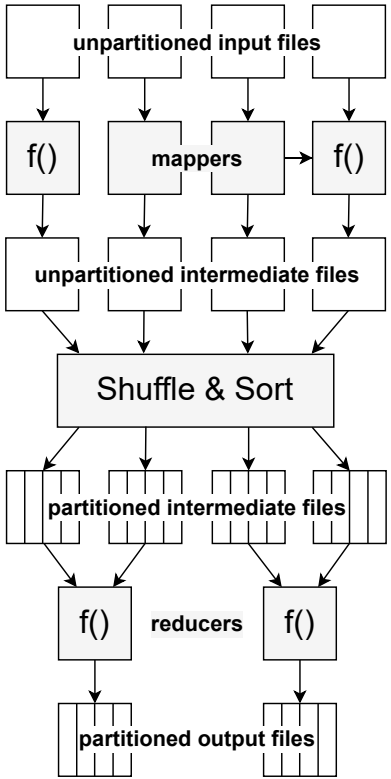
# The MapReduce Programming Model

- Inspired by functional programming
- Higher order functions (map, reduce, …) call user-defined lower order functions
- All functions are side effect free
- Simple parallelization model for arbitrary computations

*”The user just has to setup the cluster”*

---

*Instance type? Cluster size? Pricing model?*



# Function as a Service Platforms

- Users write pieces of code, packaged as ZIP or container image
  - Much like map functions in MapReduce
  - Image may contain entire operating system user space
- Providers transparently schedule, load balance, and scale user code
  - Operational simplicity
- Can startup thousands of small compute units in milliseconds
  - Elastic scalability

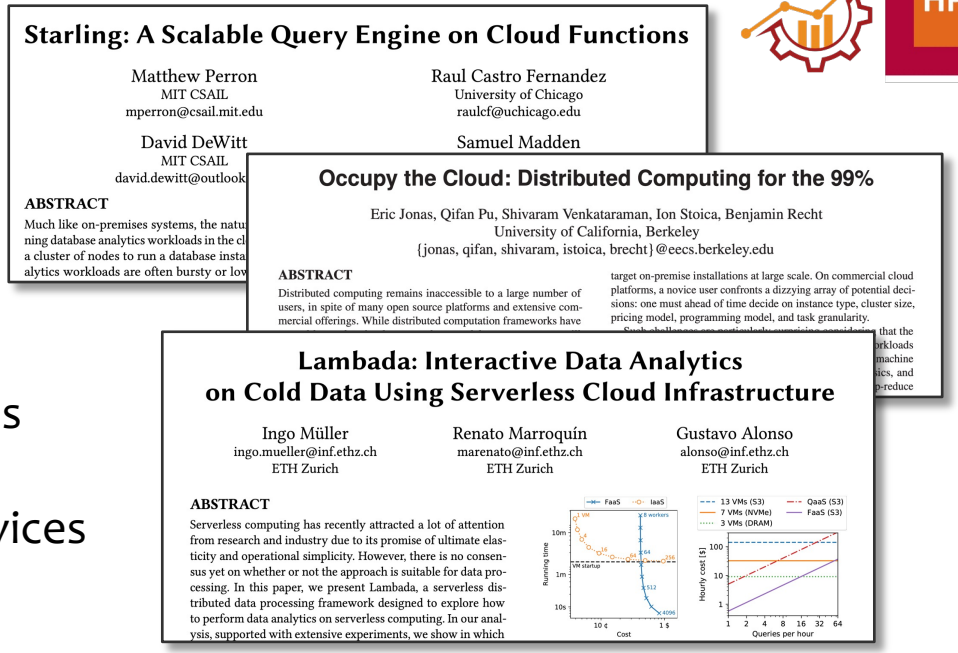
Source: AWS

Source: Google

Source: Azure

# Serverless Data Analytics

- Exploit benefits of FaaS platforms while working around limitations
- Efficient batch-start of many functions
- Efficient I/O to serverless storage services
- Cost-efficient staged shuffle
- Mitigation of straggling service requests
- …



**Starling: A Scalable Query Engine on Cloud Functions**

Matthew Perron
MIT CSAIL
mperron@csail.mit.edu

Raul Castro Fernandez
University of Chicago
raulcf@uchicago.edu

David DeWitt
MIT CSAIL
david.dewitt@outlook

Samuel Madden

ABSTRACT

Much like on-premises systems, the natu
ning database analytics workloads in the cl
a cluster of nodes to run a database insta
alytics workloads are often bursty or lo

**Occupy the Cloud: Distributed Computing for the 99%**

Eric Jonas, Qifan Pu, Shivaram Venkataraman, Ion Stoica, Benjamin Recht
University of California, Berkeley
{jonas, qifan, shivaram, istoica, brecht}@eecs.berkeley.edu

ABSTRACT

Distributed computing remains inaccessible to a large number of users, in spite of many open source platforms and extensive commercial offerings. While distributed computation frameworks have

target on-premise installations at large scale. On commercial cloud platforms, a novice user confronts a dizzying array of potential decisions: one must ahead of time decide on instance type, cluster size, pricing model, programming model, and task granularity.

**Lambada: Interactive Data Analytics on Cold Data Using Serverless Cloud Infrastructure**

Ingo Müller
ingo.mueller@inf.ethz.ch
ETH Zurich

Renato Marroquín
marenato@inf.ethz.ch
ETH Zurich

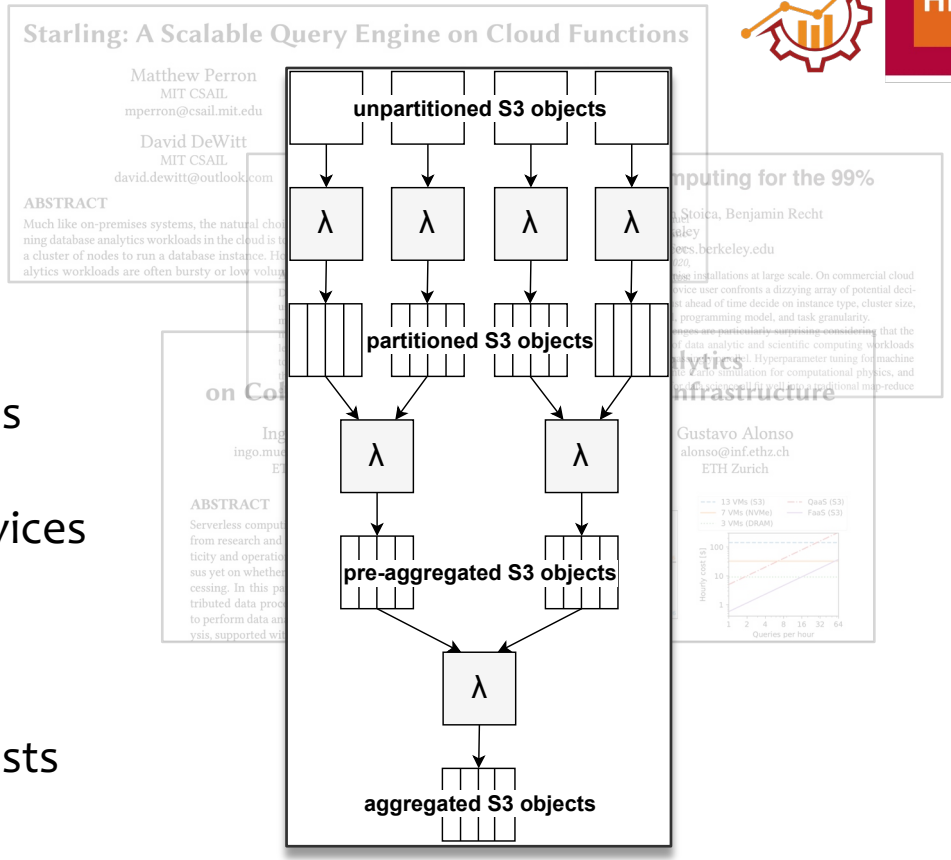
Gustavo Alonso
alonso@inf.ethz.ch
ETH Zurich

ABSTRACT

Serverless computing has recently attracted a lot of attention from research and industry due to its promise of ultimate elasticity and operational simplicity. However, there is no consensus yet on whether or not the approach is suitable for data processing. In this paper, we present Lambada, a serverless distributed data processing framework designed to explore how to perform data analytics on serverless computing. In our analysis, supported with extensive experiments, we show in which

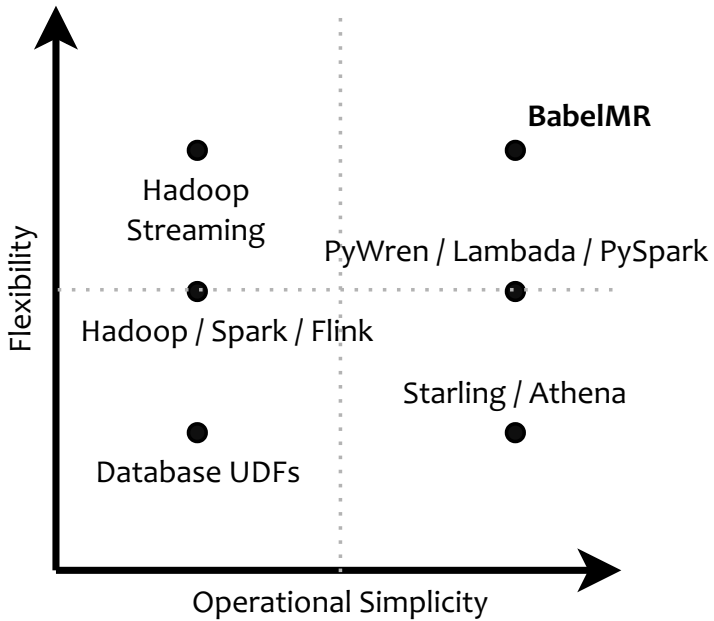
# Serverless Data Analytics

- Exploit benefits of FaaS platforms while working around limitations
- Efficient batch-start of many functions
- Efficient I/O to serverless storage services
- Cost-efficient staged shuffle
- Mitigation of straggling service requests
- …



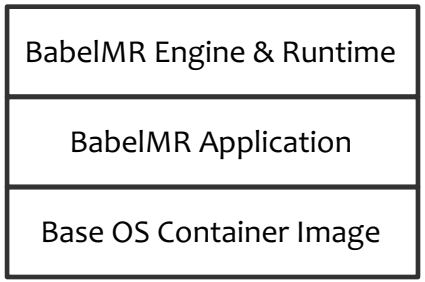

Blueprint System Architecture

# Systems for Simplified Data Analytics

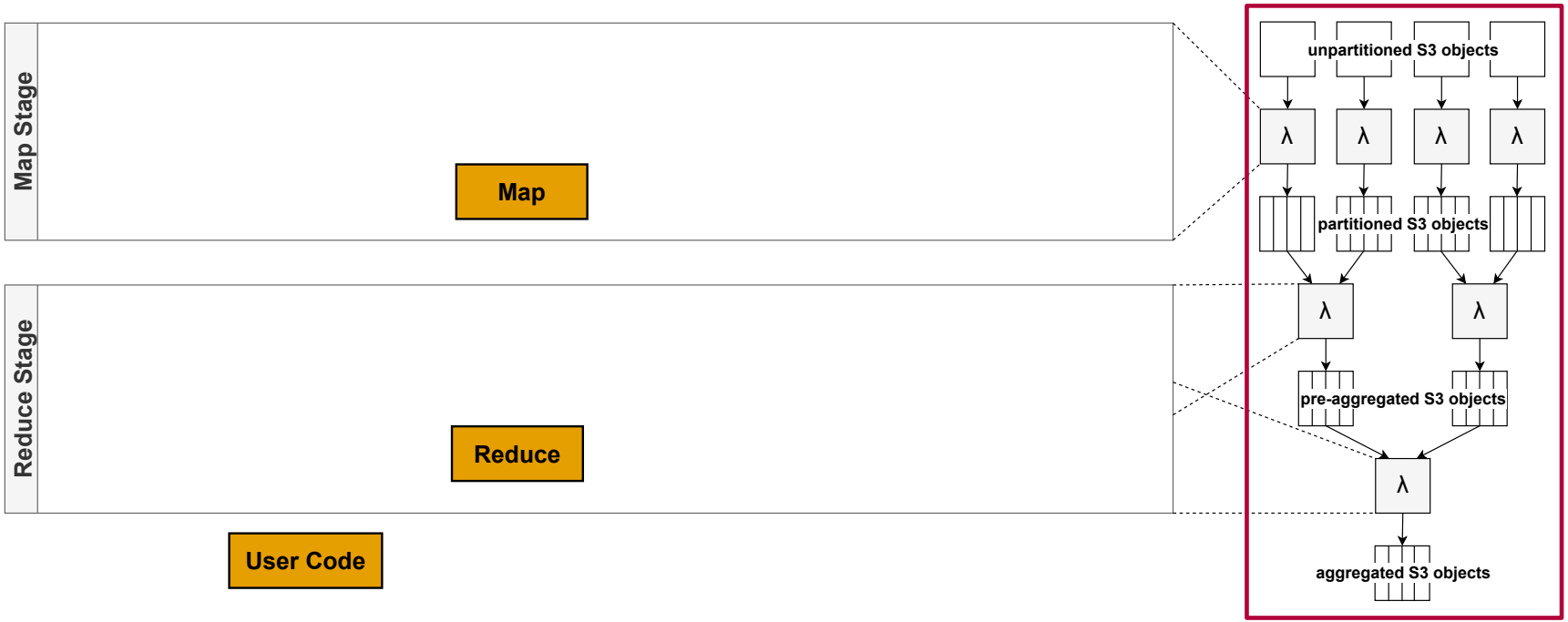


# BabelMR Programming Interface

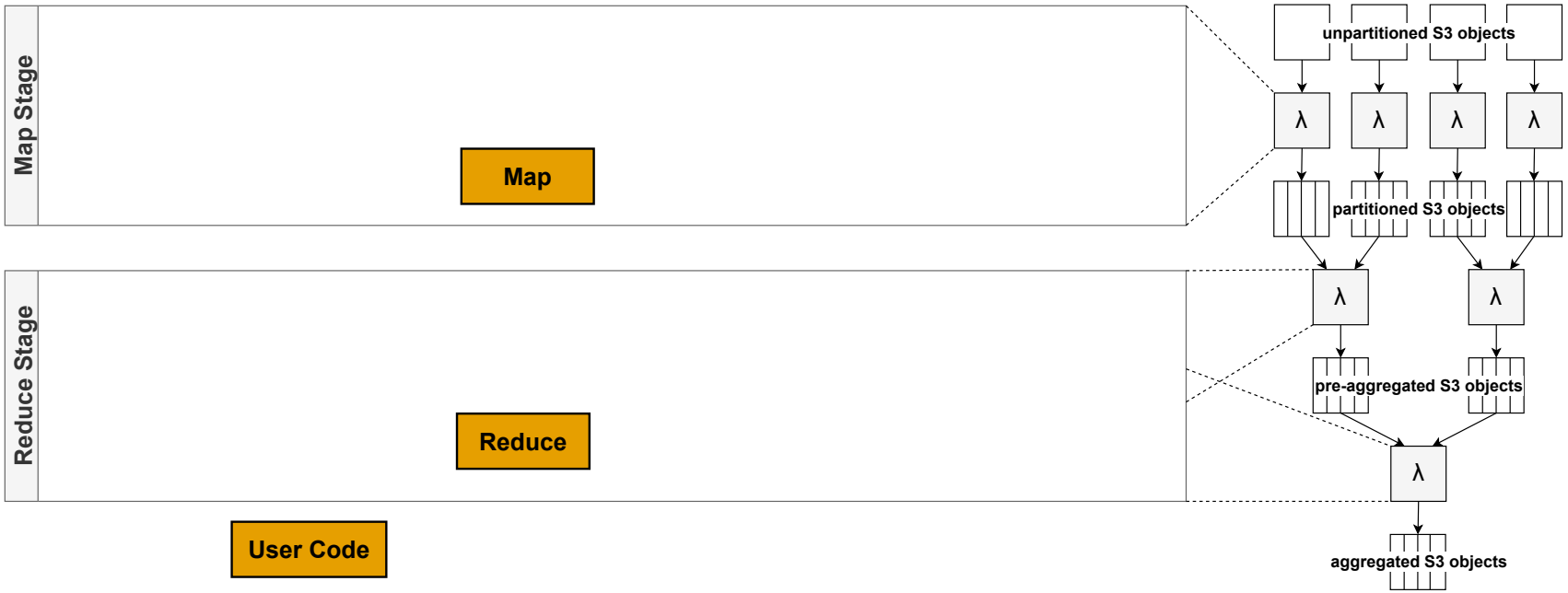
- BabelMR application
  - Container images for batched map and reduce functions
  - Cloud storage locations for inputs and outputs
  - Key-value attributes
- BabelMR engine
  - User images contain layer with BabelMR engine
  - Engine integrates with cloud storage and file formats
  - Custom Lambda runtime orchestrates interaction



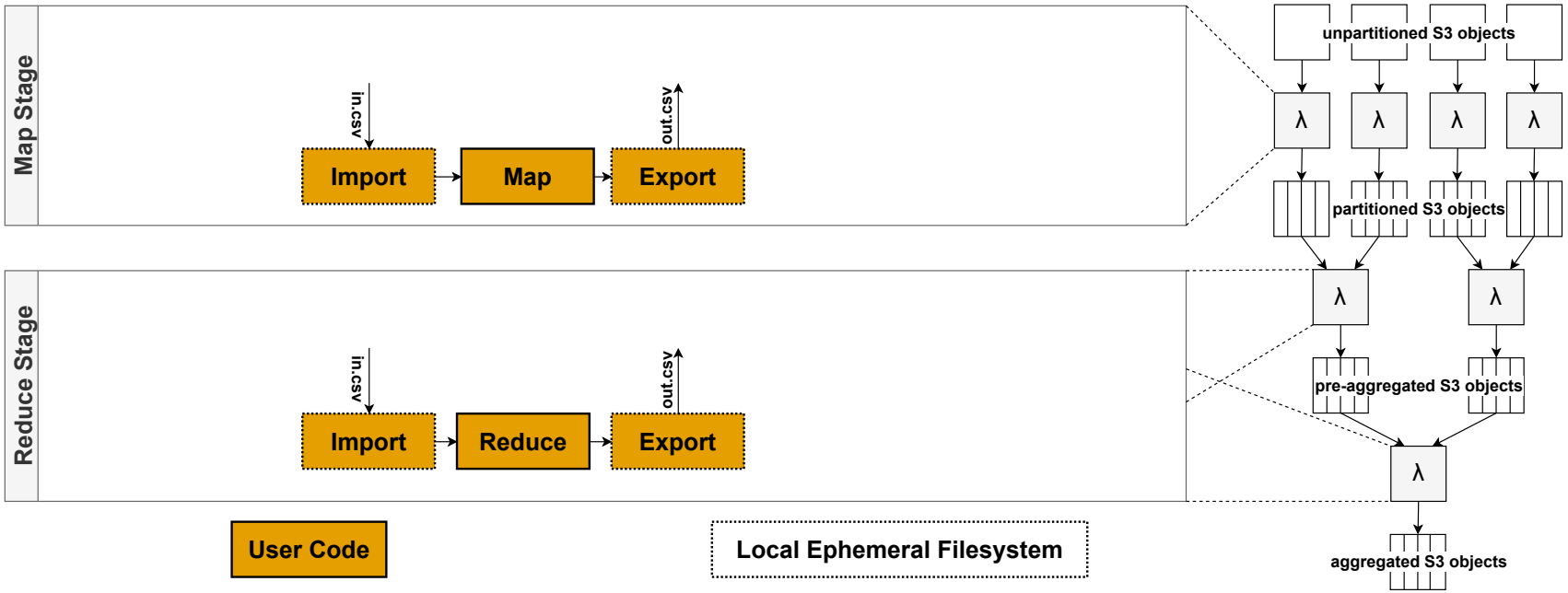
# BabelMR System Architecture



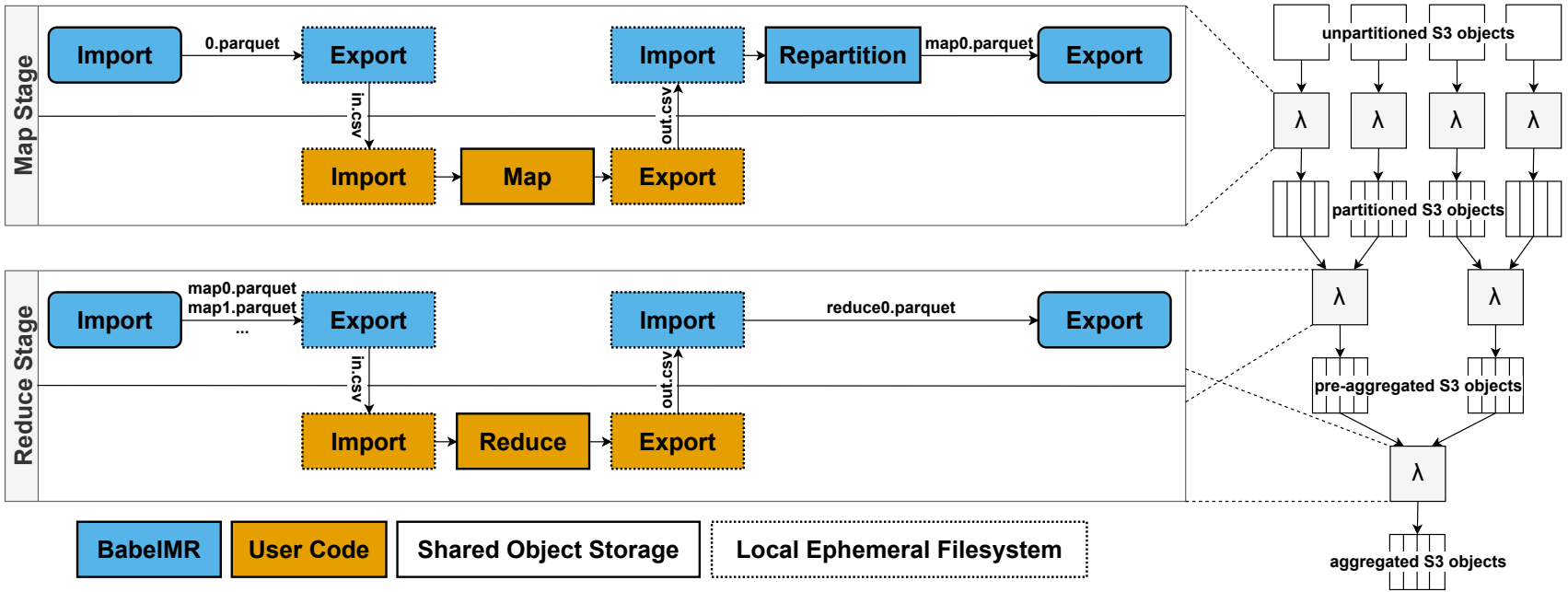
# BabelMR System Architecture



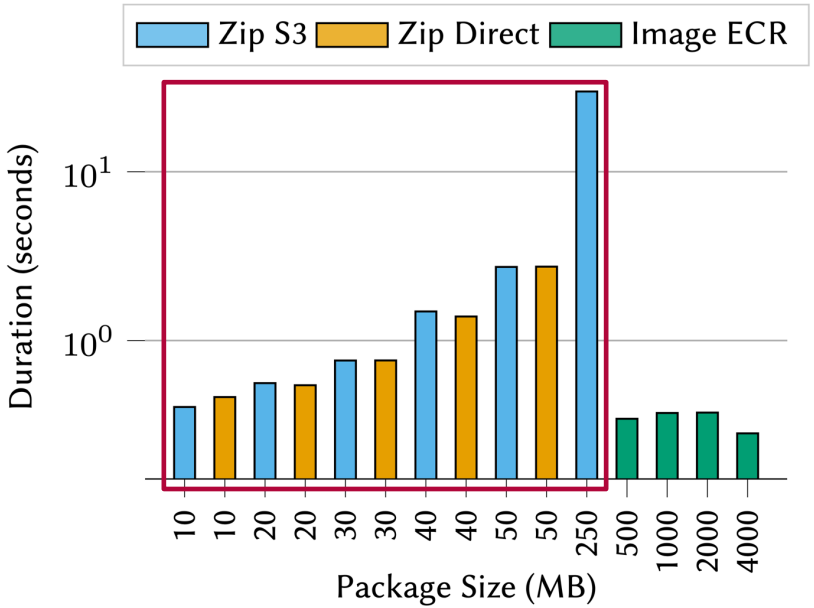
# BabelMR System Architecture



# BabelMR System Architecture

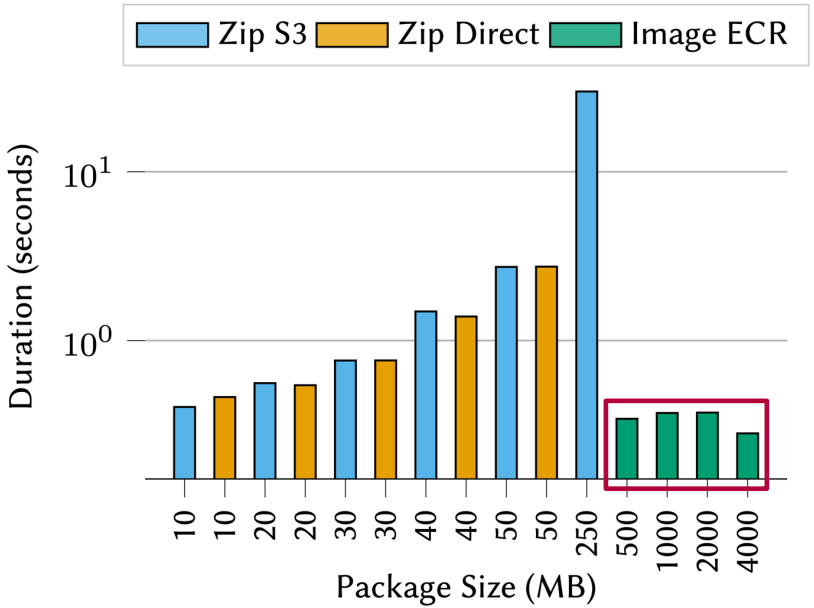


# Elasticity of Container-based Functions



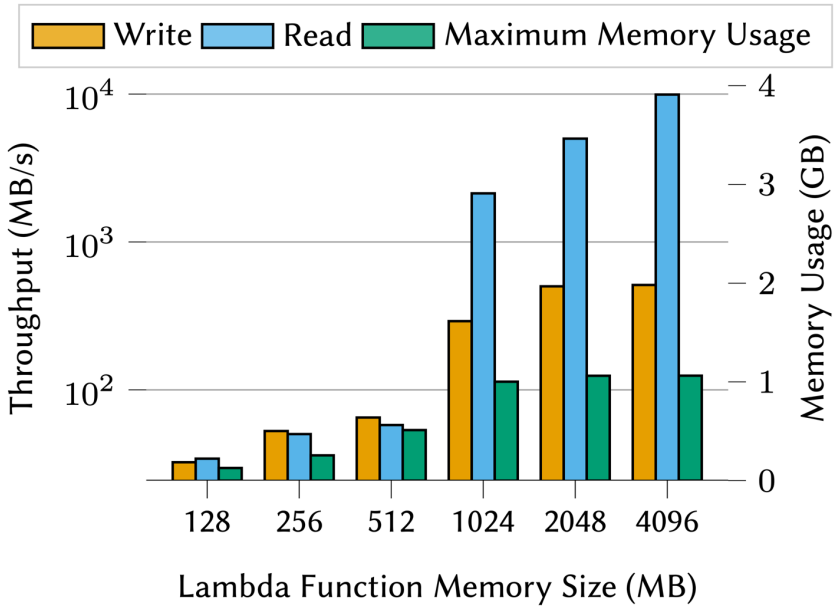

Container Startup Times

# Elasticity of Container-based Functions



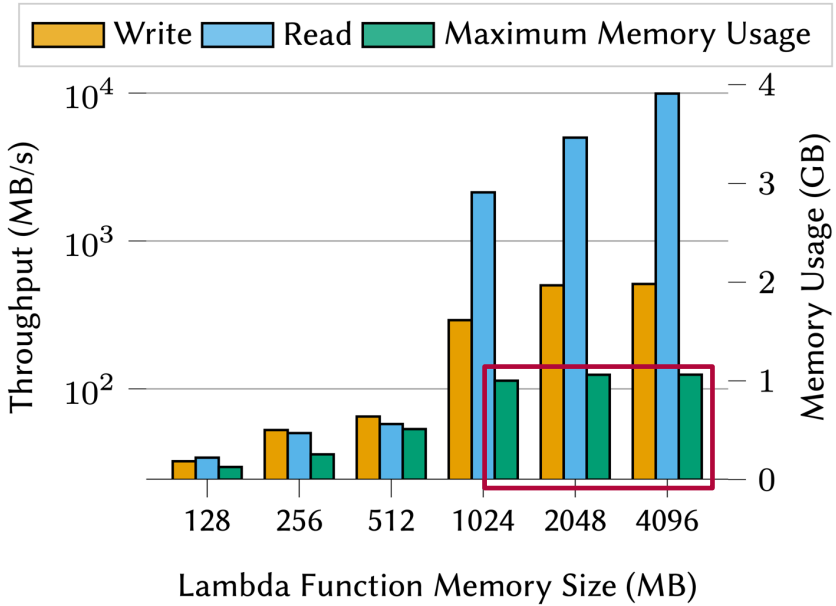

Container Startup Times

# Efficiency of Function Filesystems




Filesystem Performance

# Efficiency of Function Filesystems

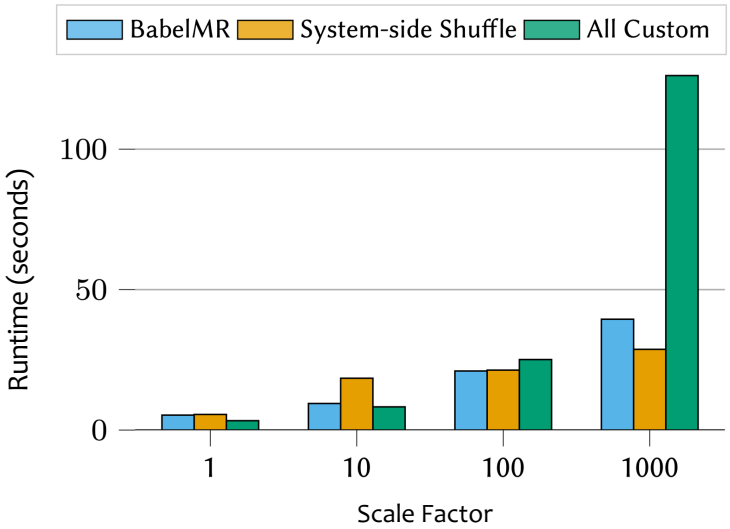



Filesystem Performance

# Evaluation of Building Blocks

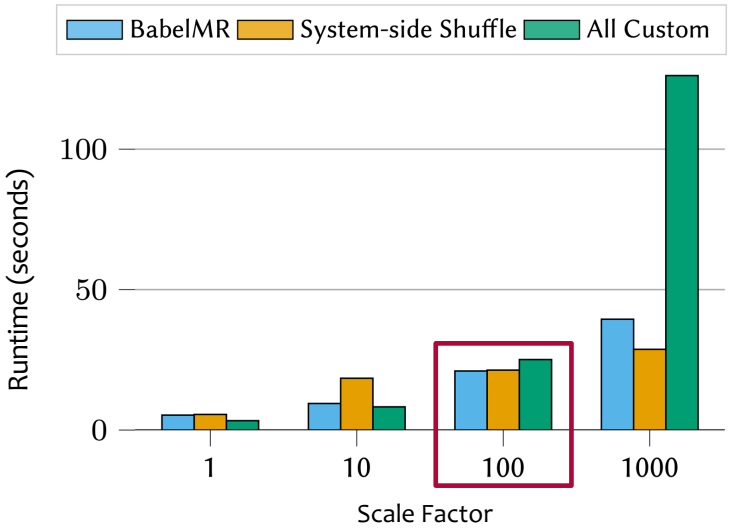
- Workload and data
  - TPC-H Q1 and TPCx-BB Q1 written in *best effort* C#, Go, and Python
  - Data at scale factors (sf) 1, 10, 100, and 1000 stored in #sf files, formatted in Parquet and CSV
- BabelMR system setup and configurations
  - AWS services: Lambda, EC2, S3
  - Lambda-based workers with 5,120 MB RAM and 512 MB SSD (between 1 - #sf)
  - EC2-based coordinator with 8 vCPUs and 16 GB RAM (c5.2xlarge)
  - All custom code, system-side shuffle, BabelMR scan and shuffle
- Runtimes averaged over 10 warm runs
- Measured from May to July 2023 in AWS region us-east-1 for ~$1,500

# Building Blocks improve Execution Efficiency

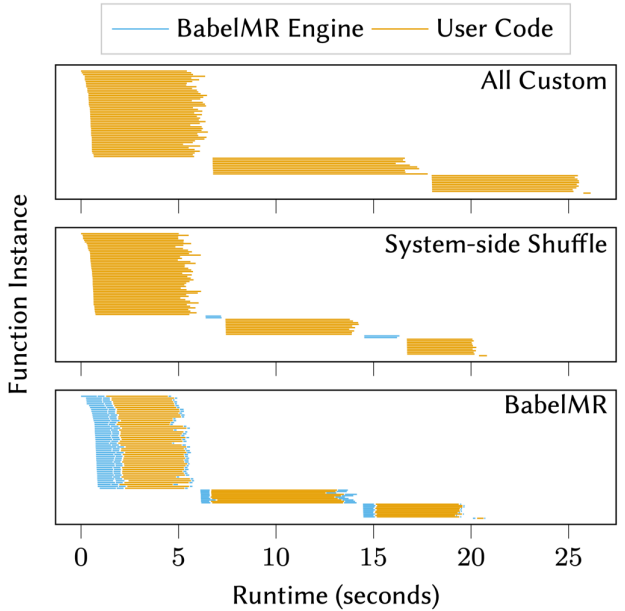



Runtimes for TPCx-BB Q1 in C#

# Building Blocks improve Execution Efficiency




Runtimes for TPCx-BB Q1 in C#

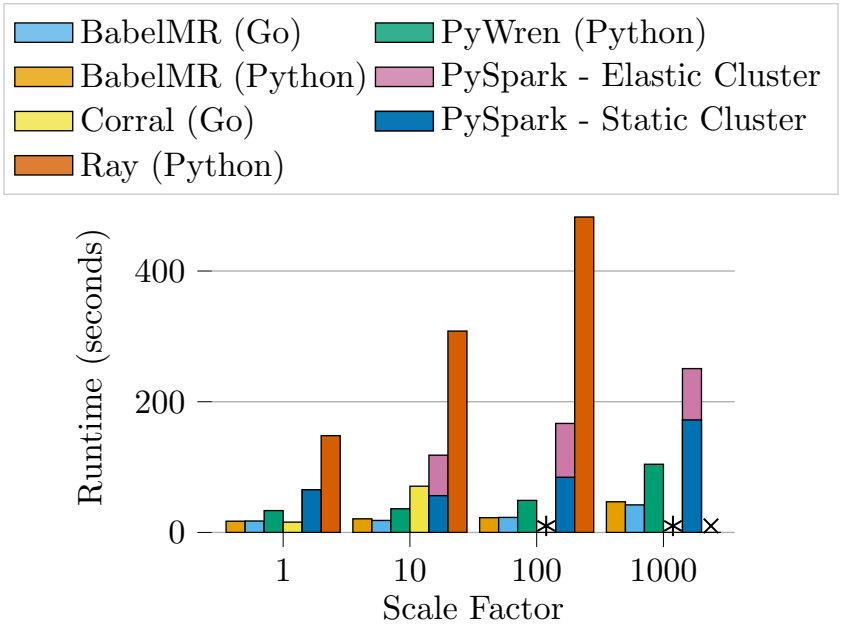



Runtime Breakdown per Function

# End-to-End Evaluation

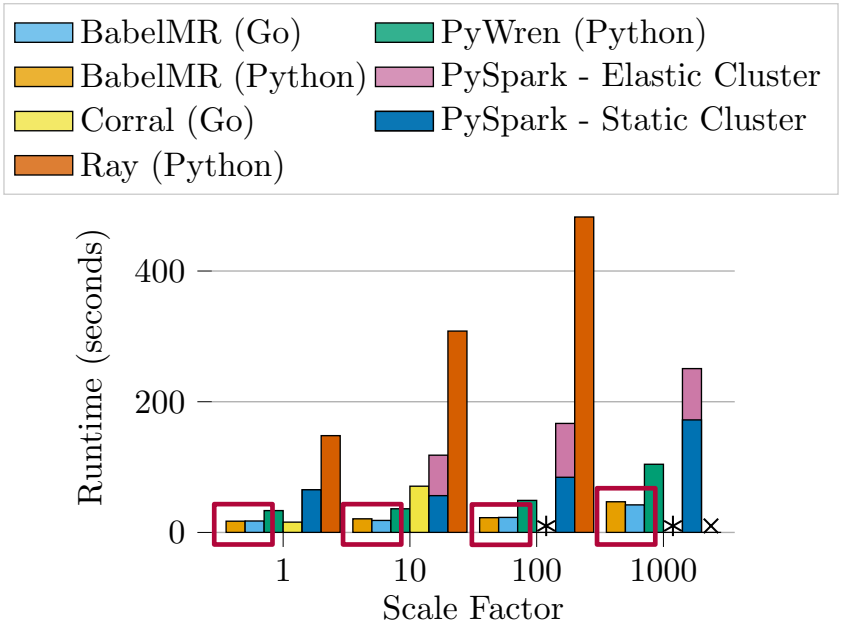
- Setup for serverless systems Corral, PyWren, and BabelMR
  - AWS services: Lambda, EC2, S3
  - Lambda-based workers with 5,120 MB RAM and 512 MB SSD (between 1 - #sf)
  - EC2-based coordinator with 8 vCPUs and 16 GB RAM (c5.2xlarge)
- PySpark setup
  - AWS EMR 6.11
  - EMR used 1, 4, 40, and 400 workers with 16 vCPUs and 32 GB RAM
  - Elastic and static clusters
- Ray setup
  - AWS Glue 4.0
  - Glue used Z.2x machines with equivalent amount of resources
  - Only elastic clusters

# System Performance Comparison



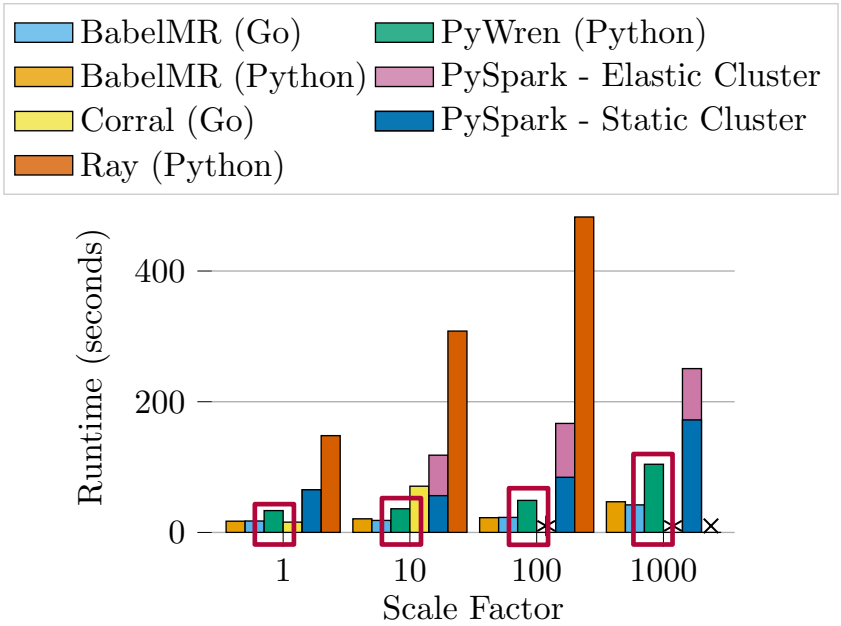

Runtimes for TPC-H Q1

# System Performance Comparison



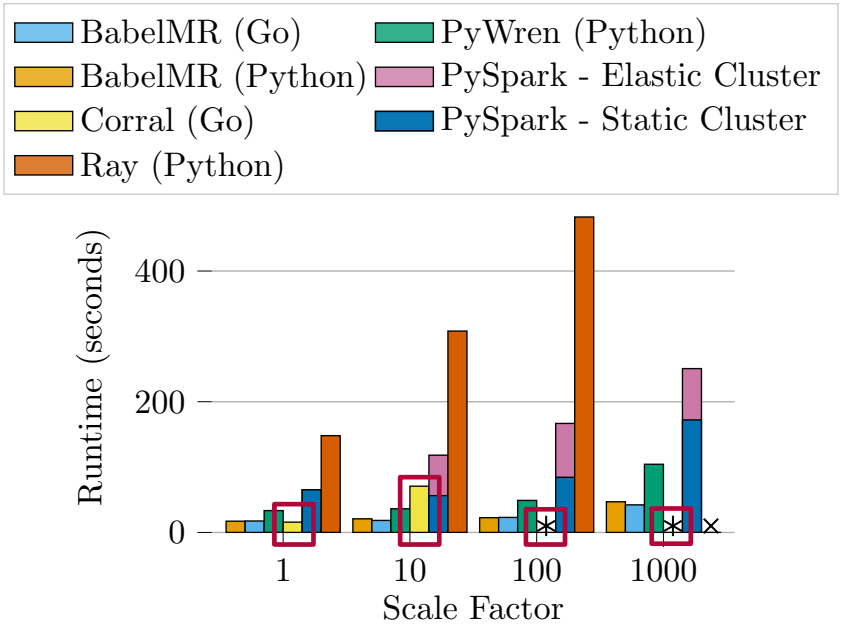

Runtimes for TPC-H Q1

# System Performance Comparison



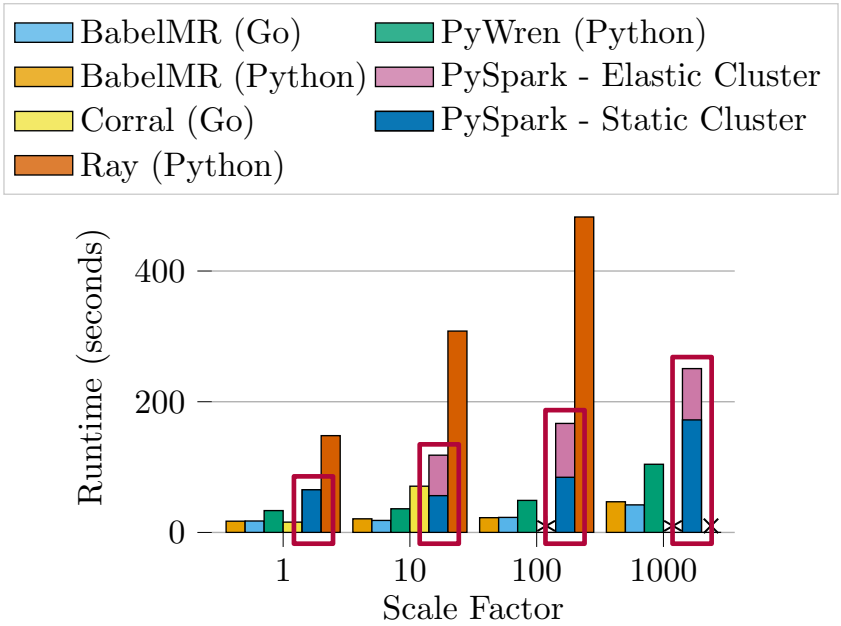

Runtimes for TPC-H Q1

# System Performance Comparison



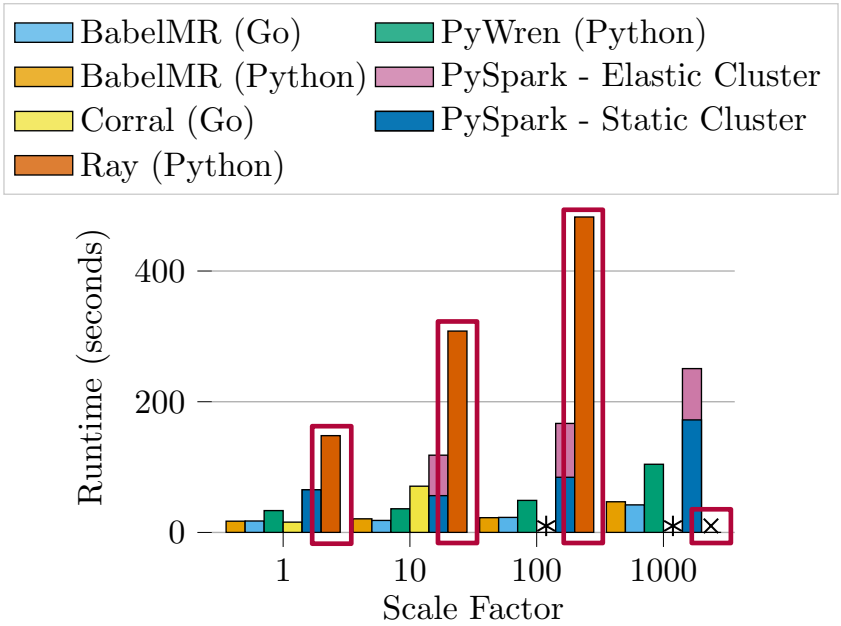

Runtimes for TPC-H Q1

# System Performance Comparison



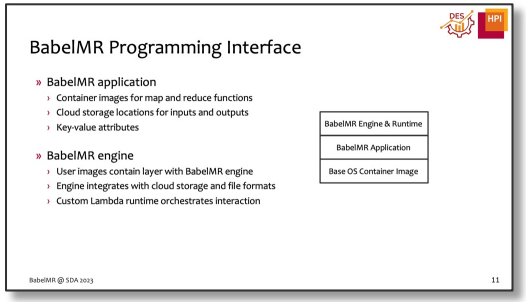

Runtimes for TPC-H Q1

# System Performance Comparison

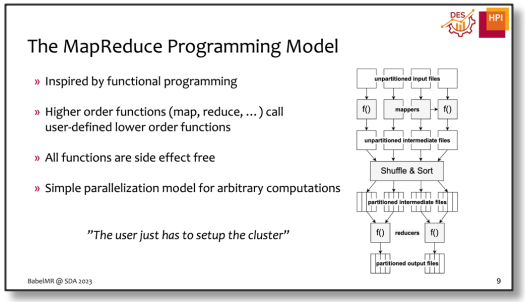



Runtimes for TPC-H Q1

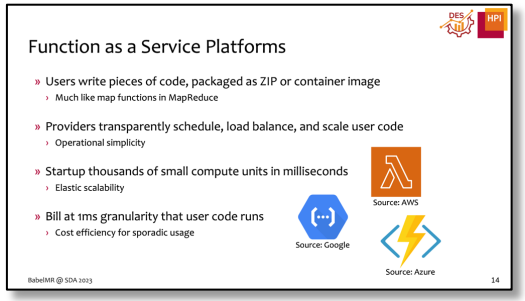
# BabelMR Summary



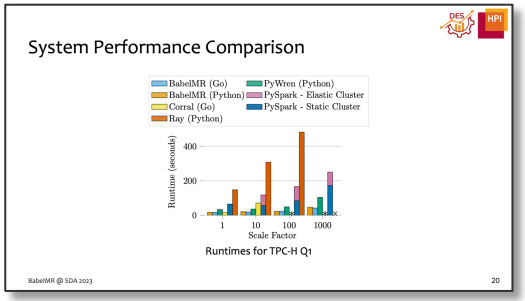

Wraps arbitrary containers to maximize flexibility




Exposes MapReduce model to simplify parallel execution

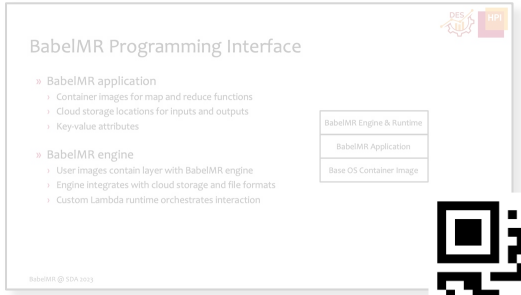



Builds on serverless infrastructure to simplify cluster management

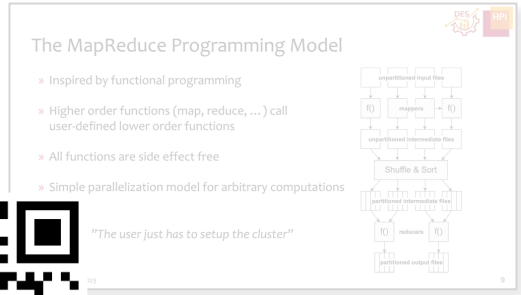



Performs as state-of-the-art serverless systems

# BabelMR Summary

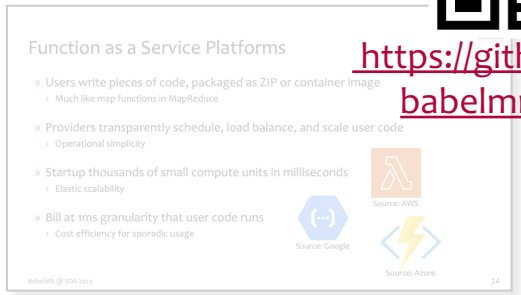



Wraps arbitrary contain
to maximize flexibility

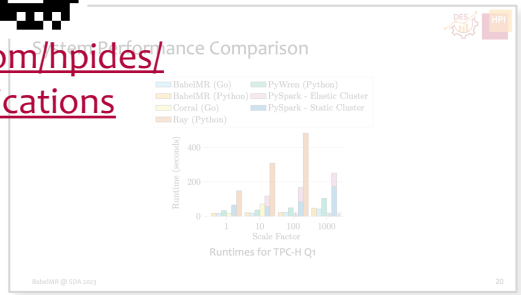



oses MapReduce model
implify parallel execution

https://github.com/hpides/babelmr-applications




Builds on serverless infrastructure
to simplify cluster management




Performs as state-of-the-art
serverless systems