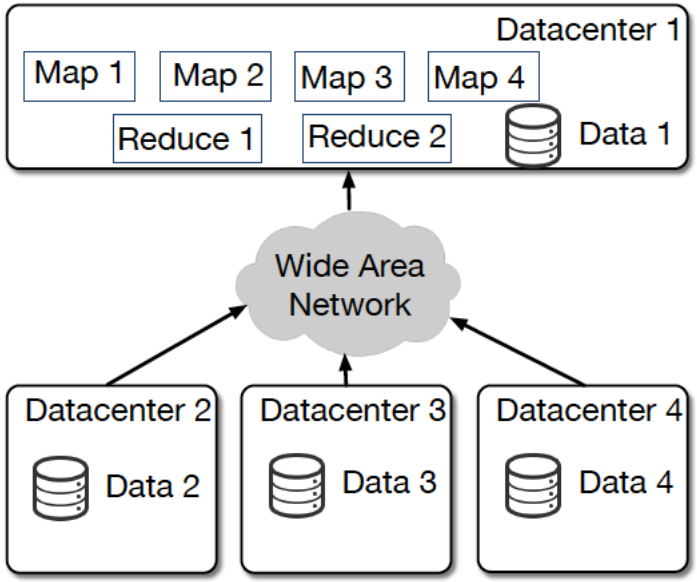
Flutter: Scheduling Tasks Closer to Data Across Geo-Distributed Datacenters

INFOCOM 2016

Big Data Processing

- Mostly stored in geographically distributed datacenters around the world
- Traditional way
  - A centralized fashion
  - Transfer all data across world to a single datacenter

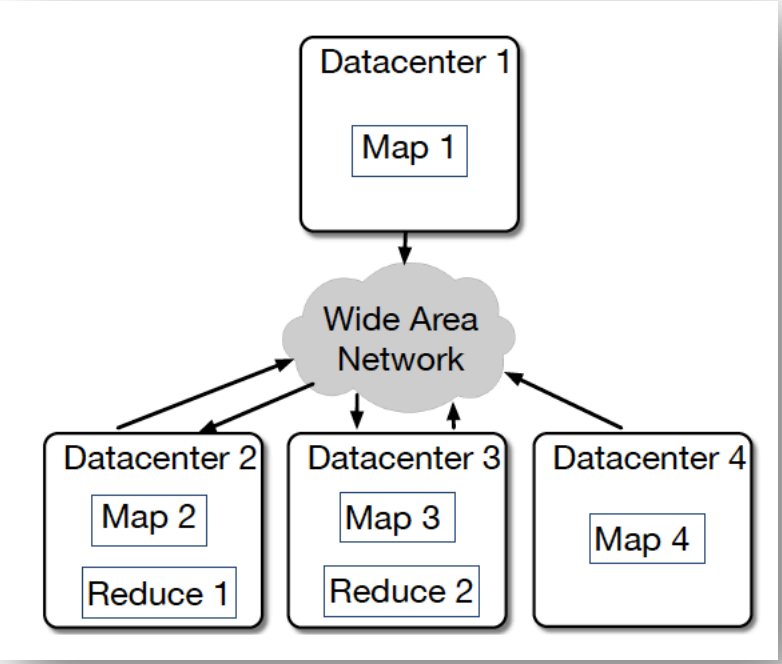


Big Data Processing

- Impractical
  - Legal reason or privacy concerns: can not move data across country boundaries
  - Bandwidth or time cost: prohibitive to move large volume of data across geo-distributed datacenters
- A better design
  - Move the computation tasks to where the data is
  - Process data locally within the same datacenter
  - The intermediate data size is much smaller than input data

Big Data Processing

- Goal:
  - Minimize the job completion time by placing the tasks at their respective best possible datacenters

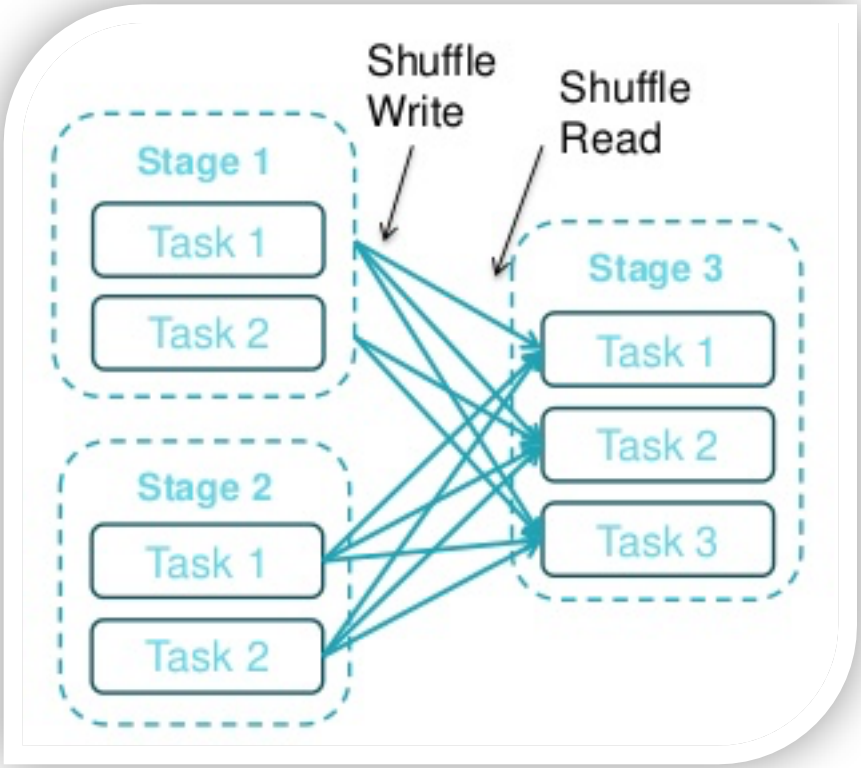


Flutter

- Online scheduling
  - Stage by stage
  - Make adjustment based on the current job progress
- Stage-aware
  - Minimize the completion time of each stage
- Network-aware
  - Consider the available inter-datacenter bandwidth

Stage in Spark

- A set of independent tasks all computing the same function
- The job is transferred into DAG of tasks first
- The DAGScheduler runs these stages in topological order.



Observation

- The intra-datacenter bandwidth is high
  - Sufficient enough for typical Spark application
- The bandwidth across datacenters is lower
  - Varies significantly for different inter-datacenter links
- Bottleneck
  - Transfer times of intermediate data across datacenters

Problem Formulation

- Only schedules tasks within a stage

Shuffle time

Task execution time

$$\underset{X}{\text{lexmin}} \quad \max_{i,j}(x_{ij} \cdot (c_{ij} + e_{ij})) \qquad (1)$$

For each task

$$\text{s.t.} \quad \sum_{j=1}^{d} x_{ij} = 1, \qquad \forall i \in \mathcal{N} \qquad (2)$$

Not exceed the affordable # of tasks in datacenter *j*

For each datacenter

$$\sum_{i=1}^{n} x_{ij} \leq f_j, \qquad \forall j \in \mathcal{D} \qquad (3)$$

Transfer time of the task *i* to datacenter *j*

$$c_{ij} = \max_{k \in s_i}(m_{d_k j}/b_{d_k j}), \quad \forall i \in \mathcal{N}, \ \forall j \in \mathcal{D} \ (4)$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i \in \mathcal{N}, \ \forall j \in \mathcal{D} \ (5)$$

Assign task *i* to datacenter *j*

Solution

- Transform into a Nonlinear Programming Problem
  - Separable convex objective function
- Transform the Nonlinear Programming Problem into a LP
  - λ-representation
    - Transform to a piecewise-linear function
  - Totally unimodular constraint matrix
    - Integer solution

Separable Function

- Express as the sum of functions of the individual decision variables
- $f(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i)$

- Convert each individual function $f_i(x_i)$ to a piecewise linear function

λ-representation

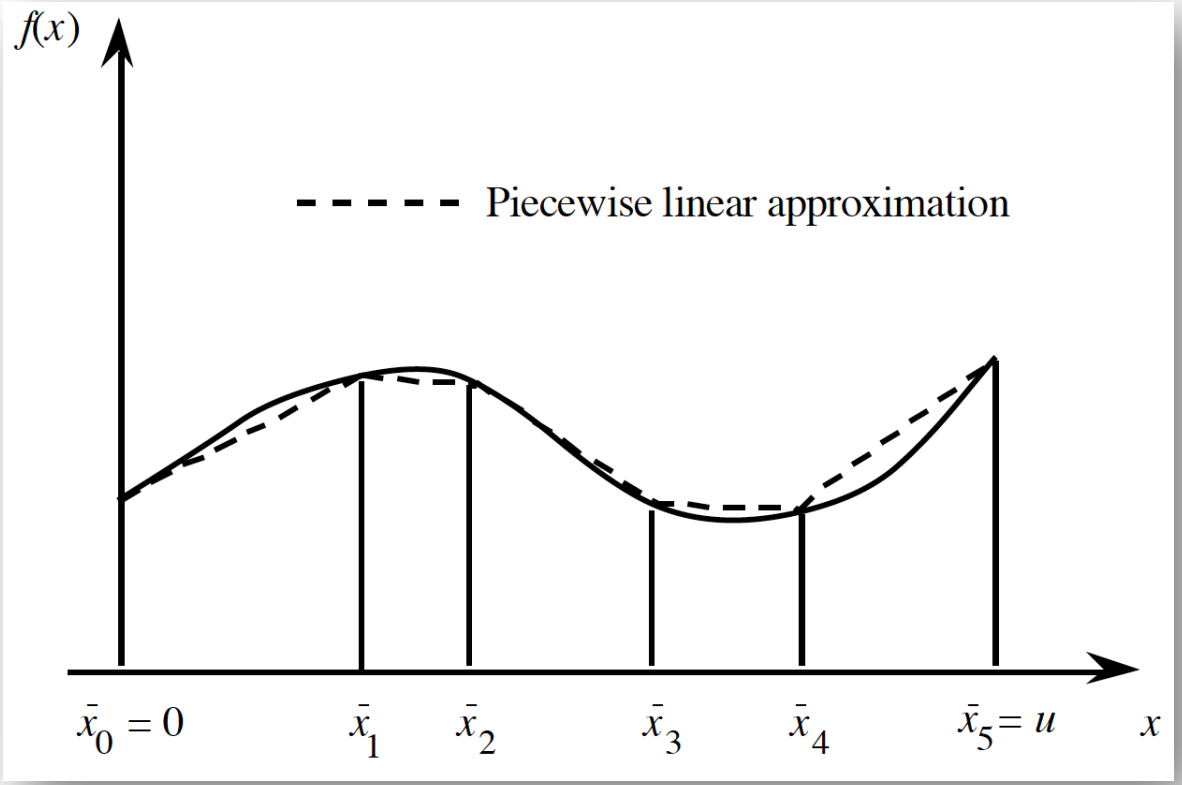
- x_{ij} equals to the weighted combination of λ_h

$$f(x) = \sum_{h \in \mathcal{P}} f(h)\lambda_h$$

$$\sum_{h \in \mathcal{P}} h\lambda_h = x$$

$$\sum_{h \in \mathcal{P}} \lambda_h = 1$$

$$\forall \lambda_h \in R^+, \forall h \in \mathcal{P}$$



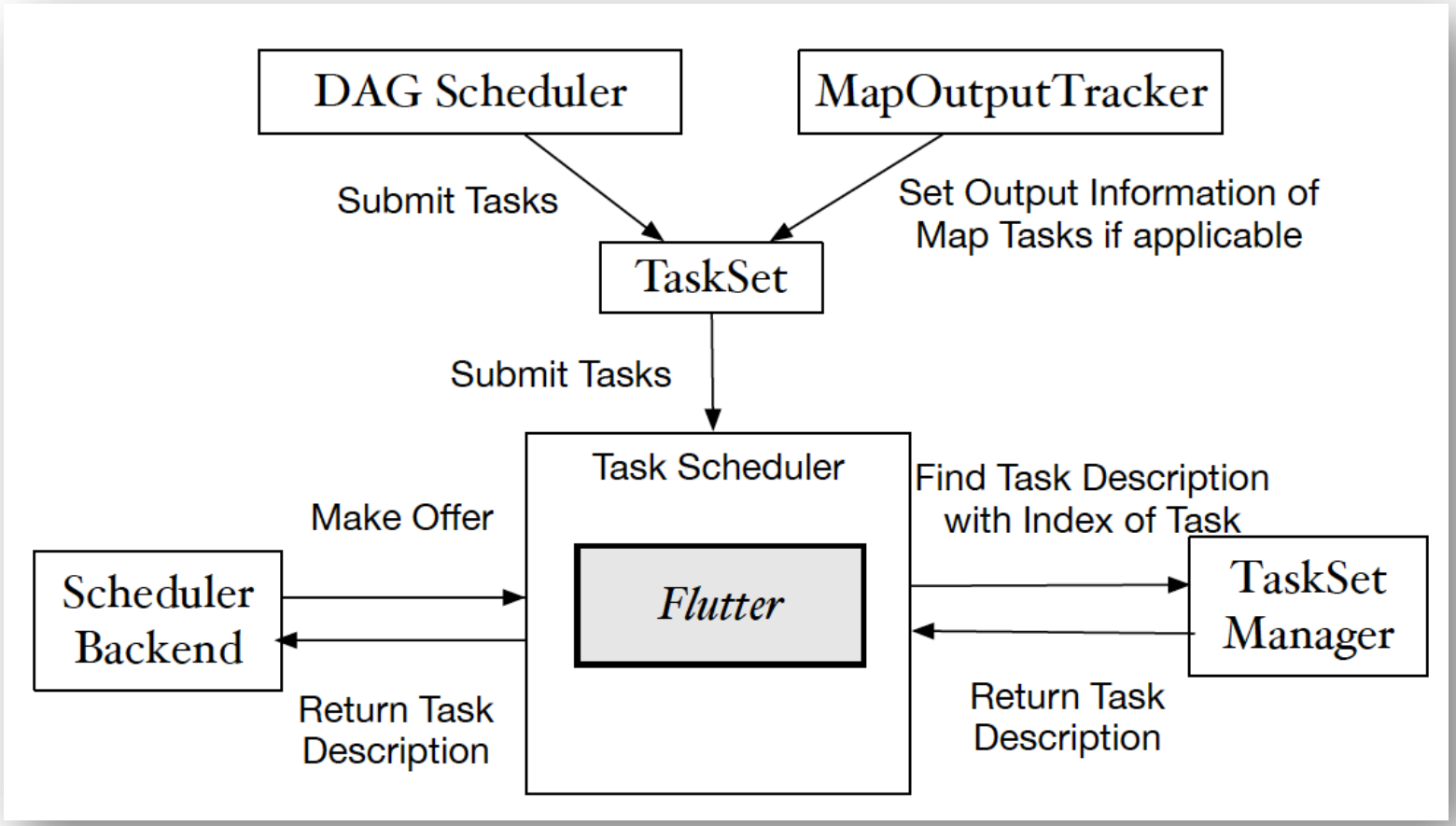
Totally Unimodular Matrix

- The determinant of each square submatrix of A is 0 or ± 1
- The solution of $Ax = b$ must integral
- Find the integer solution by solving LP directly

$$Ax = b \iff x = A^{-1}b \iff \forall i : x_i = \frac{det(A^i)}{det(A)}$$

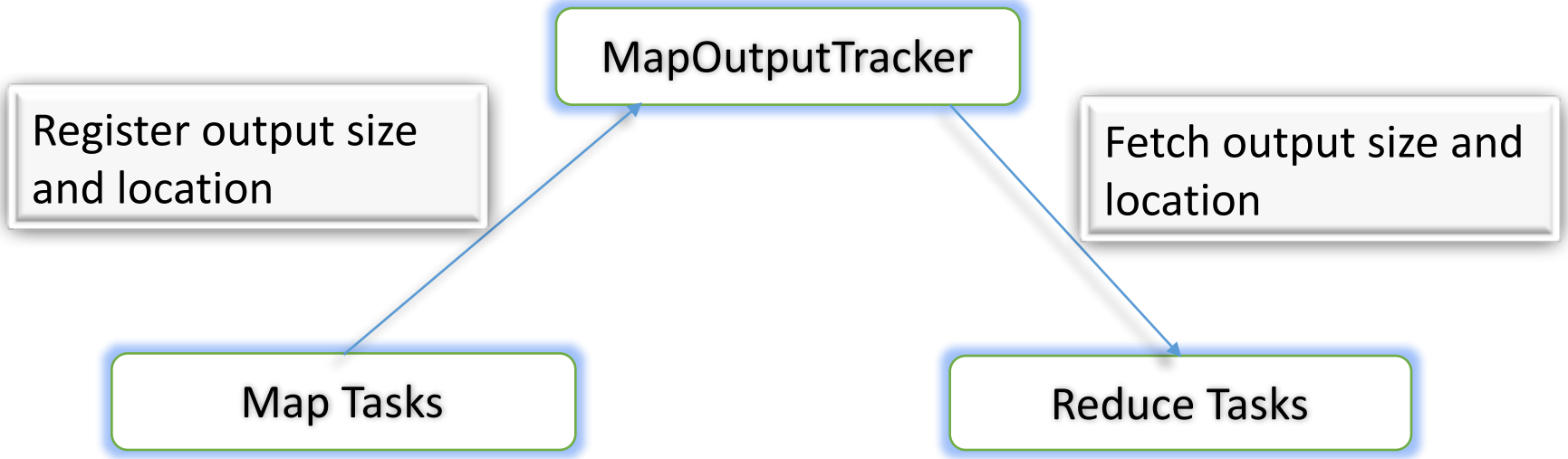
Implementation

- Job → DAG



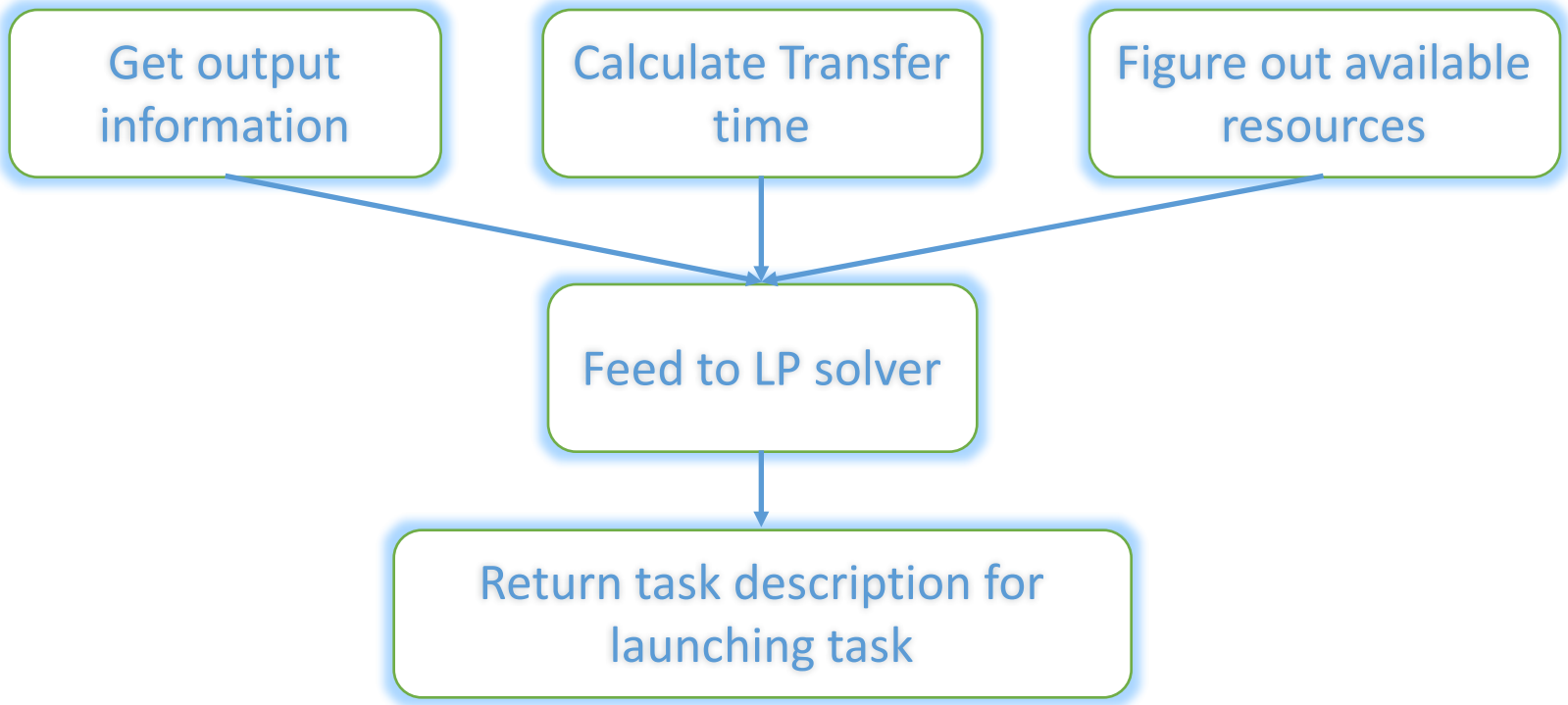
Implementation

- Obtaining Outputs of the Map Tasks
  - Save the map output information in TaskSet



Implementation

- Task Scheduling with Flutter
- Only deal with reduce tasks



Experiment

- Baseline: Delay scheduling, default task scheduler in Spark
- Main challenge
  - Data locality – For efficiency, must run tasks near their input data
  - Task picked by policy may not have data on free nodes
- Solution
  - If a local task can not be launched, launch other tasks first
  - A task waits for a limited amount of time for a scheduling opportunity on a node with data for it
- Only check locality for map tasks
  - Reduce tasks normally read roughly equal amounts of data from all nodes

Experiment

- Job completion time

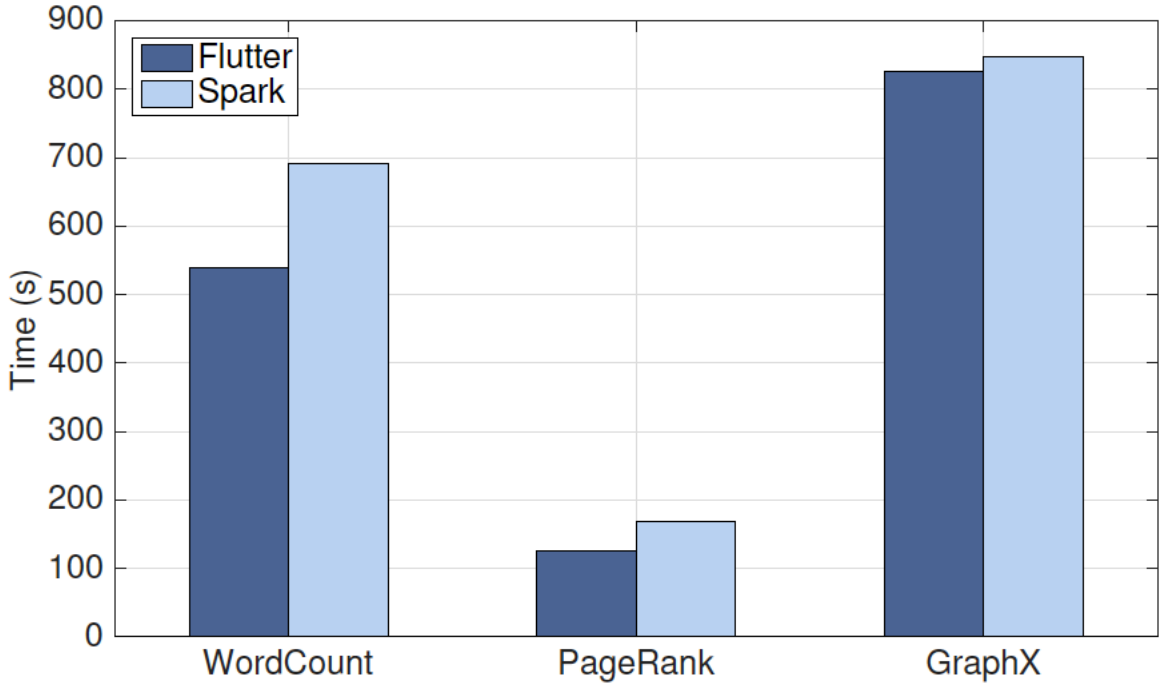
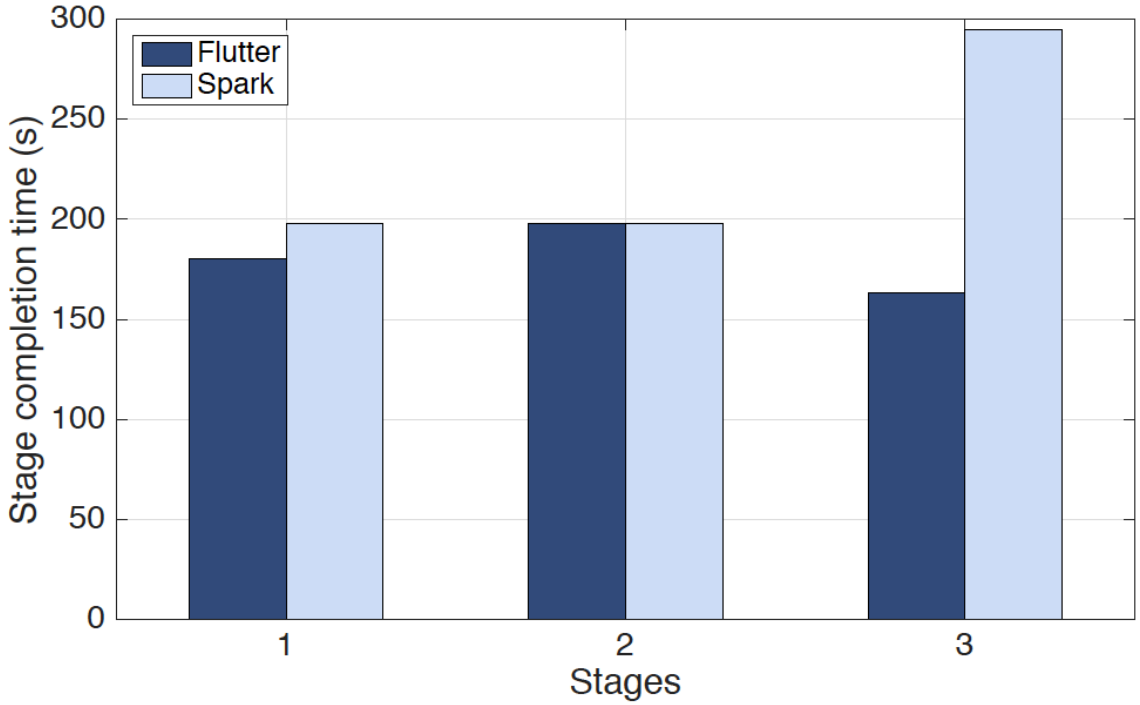


Fig. 3. The job computation times of the three workloads.

Experiment

- Stage completion time



(a) WordCount

Experiment

- Data volume transferred across datacenters

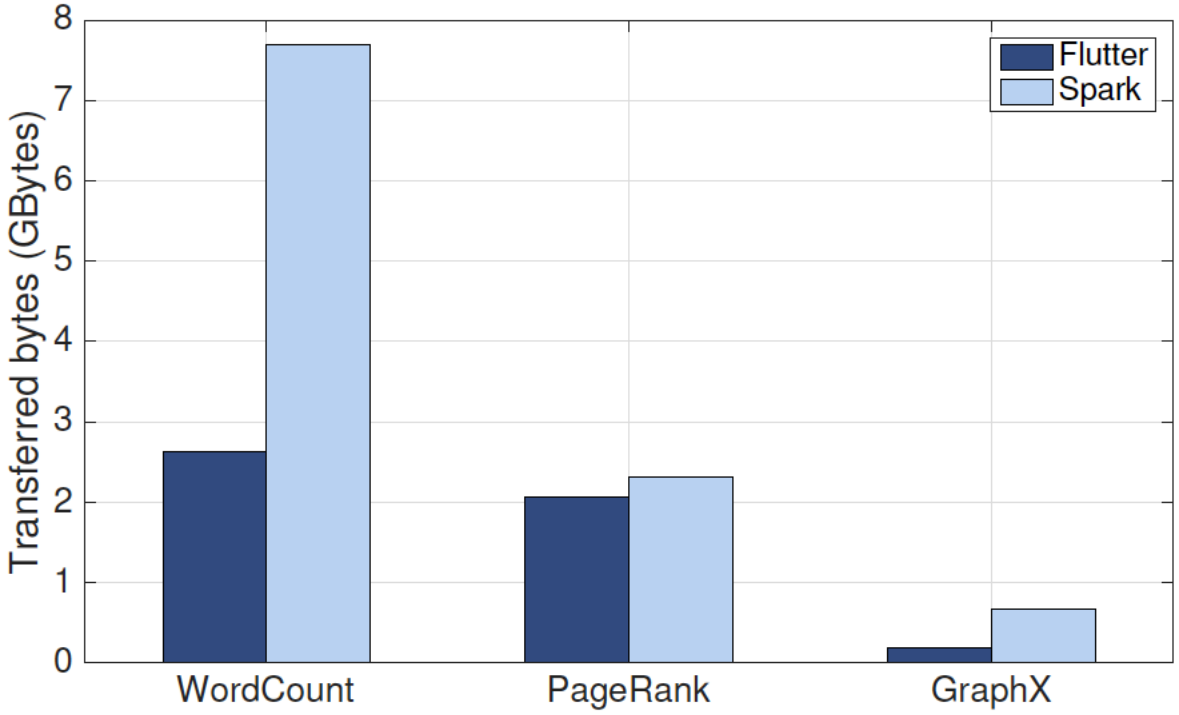


Fig. 5. The amount of data transferred among different datacenters.

Thank you ^^