Adaptive Partitioning in Spark

Prabhagaran & Arun Sundaramoorthy

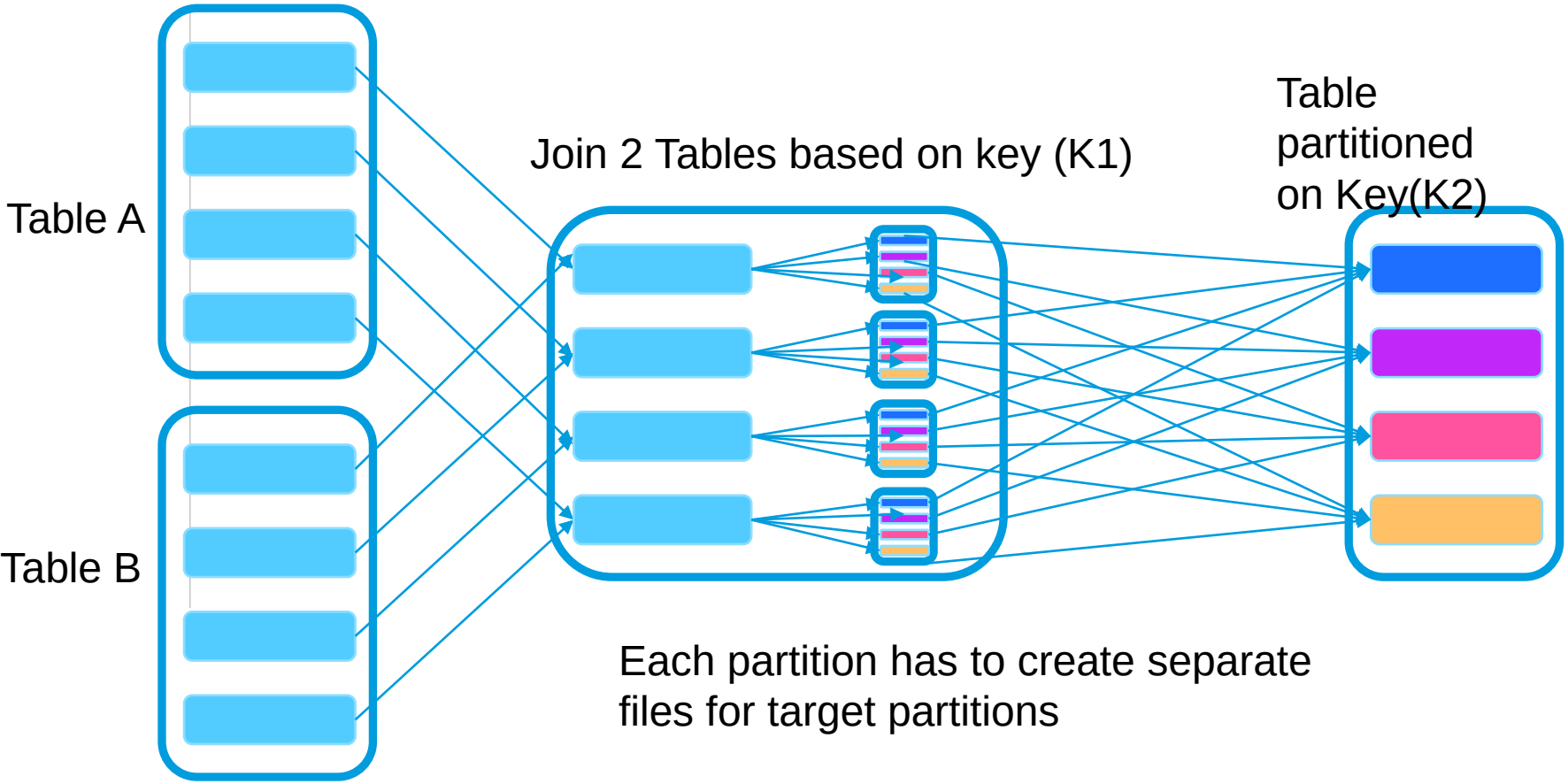
Adaptive Partitioning in Spark

About us

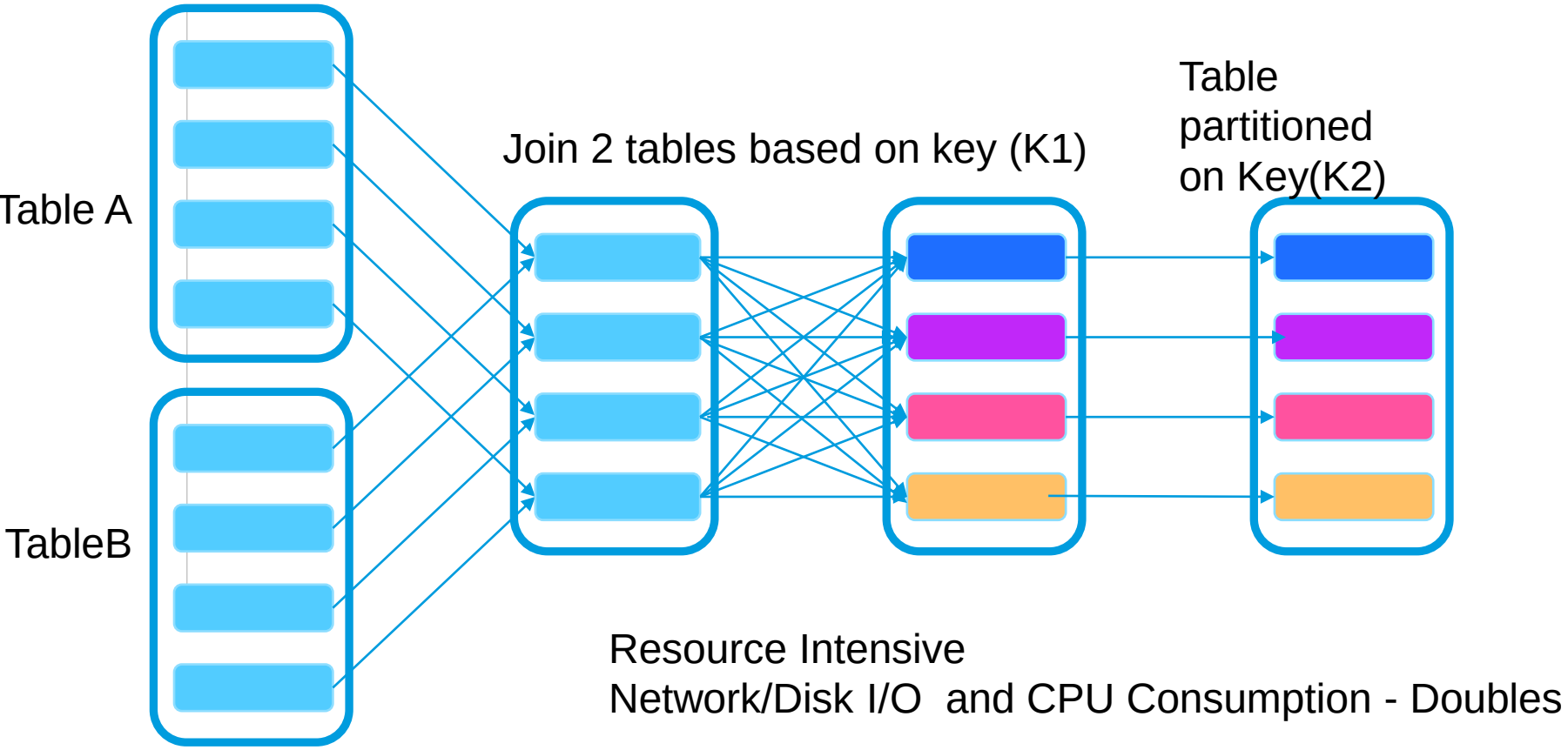
- Arun S
  - 15+ years in Data related projects
  - 7+ years of experience in USA working for Fortune 500 companies like BestBuy, CapitalOne, Hartford and Albertsons
  - PayPal (Built various Data platforms including transactions views, Customer, Product views profile/account management etc.)
  - Linkedin - https://www.linkedin.com/in/arun-sundaramoorthy-12104a6
- Prabhagaran S
  - Hands-on Big Data Engineer
  - One of the key technology experts in the PayPal Enterprise Data Services Team
  - Holds a Masters Degree in Computer Applications(MCA) from Sathyabama University
  - Linkedin - https://www.linkedin.com/in/prabhagaran-ks-a7830a13/

Table of Contents

Normal join on Partitioned Table in Spark



Join with 2 Shuffles



Resource Intensive
Network/Disk I/O and CPU Consumption - Doubles

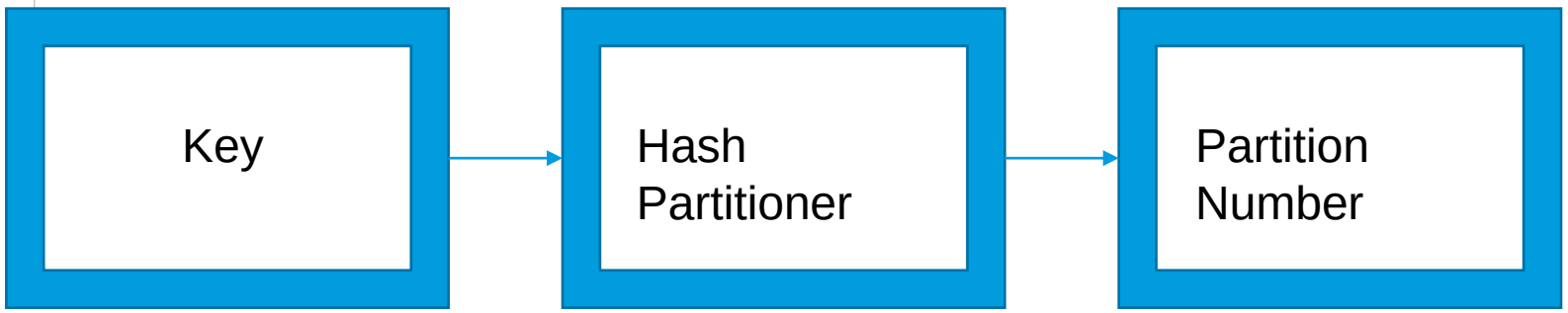
Optimizing the Join

- Build the Adaptive Partitioning Information
- Use the Partition information to distribute the data based on the join key as well as the Target table Partitions
- Example -
  - //val adaptivePartitions = Map("2014-05-01" -> Array(1, 3), "2014-03-04" -> Array(0, 1))

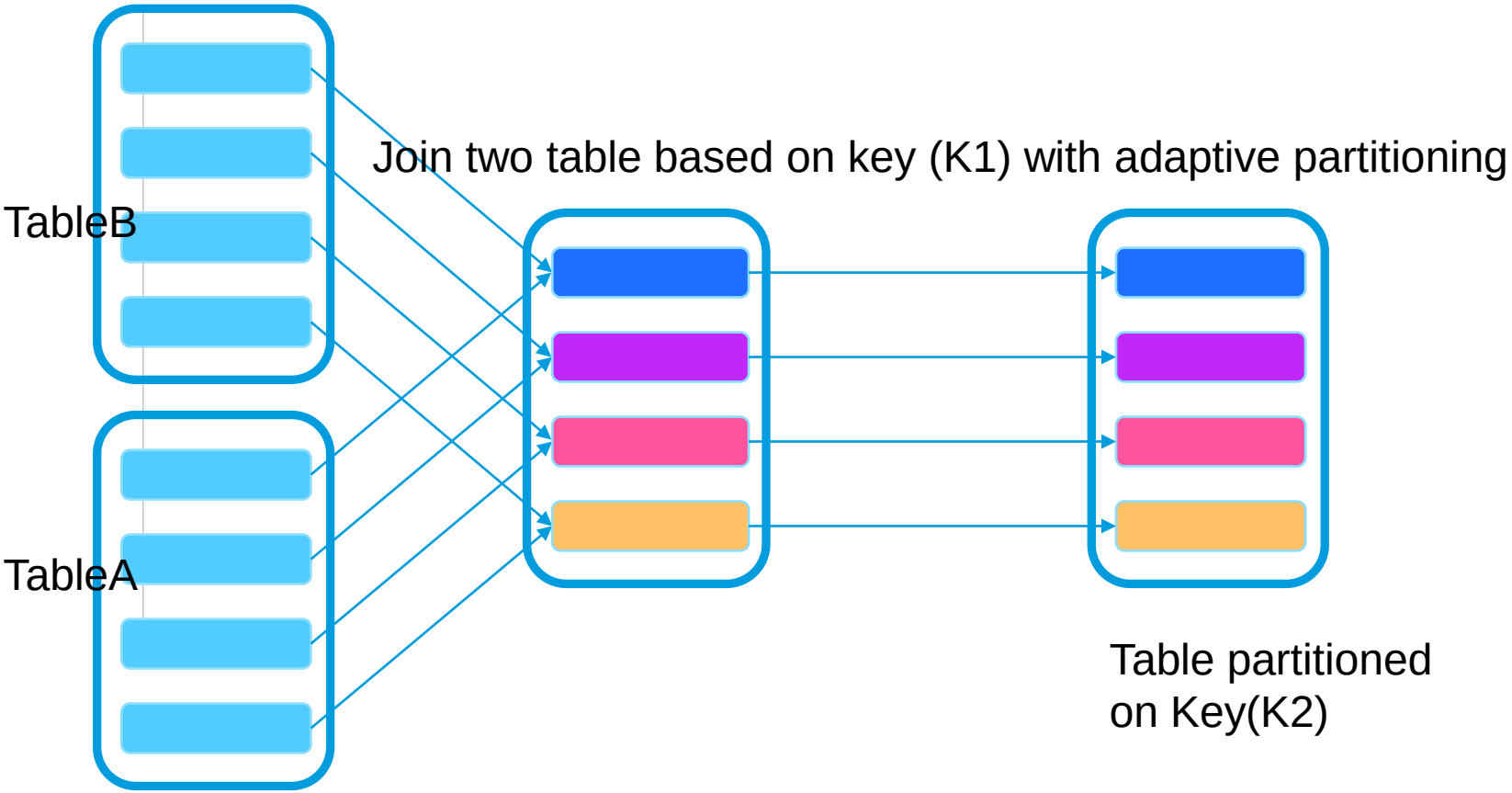
Key	Starting Partition Number	Number of Partitions
2014-05-01	1	3
2014-03-04	0	1

Adaptive Partitioning – Flow

Key	Starting Partition Number	Number of Partitions
2014-05-01	1	3
2014-03-04	0	1



Using Adaptive Partitioning in Spark



Adaptive Partitioning in Spark SQL

- Preserving the internal row format.
- No need of to many java object creation
- Performing custom hash partitioner in spark SQL
- Create UDFs to assign the partition number based on adaptive partitioning information and then use CLUSTER BY on the derived partition number to distribute the data based on key and target table partitions.

Advantages

- Precomputing the partitioning information will not consume much resources but it would aid a lot in performing the actual processing
- History rebuild on partitioned table is resource-intensive process and this would help in performing the rebuild effectively
- Optimized data distribution
- Avoid too many small files in *hdfs*

Q & A

October 5-6, 2017
Indian Institute of Science, Bangalore
www.modsummit.com

Register early and get the best discounts

April 23-28, 2018
Indian Institute of Science, Bangalore
www.developersummit.com