



WHEN TWO CHOICES ARE NOT ENOUGH

Balancing at Scale in Distributed Stream Processing

Muhammad Anis Uddin Nasir¹, Gianmarco de Francisci Morales^{3c}, Nicolas Kourtellis^{2y}, Marco Serafini^{3c}
¹KTH Royal Institute of Technology, ²Telefonica Research Barcelona, ³Qatar Computing Research Institute
anisu@kth.se⁵, gdfm@acm.org^c, nicolas.kourtellis@telefonica.com^y, msrafini@qf.org.qa^e



Abstract

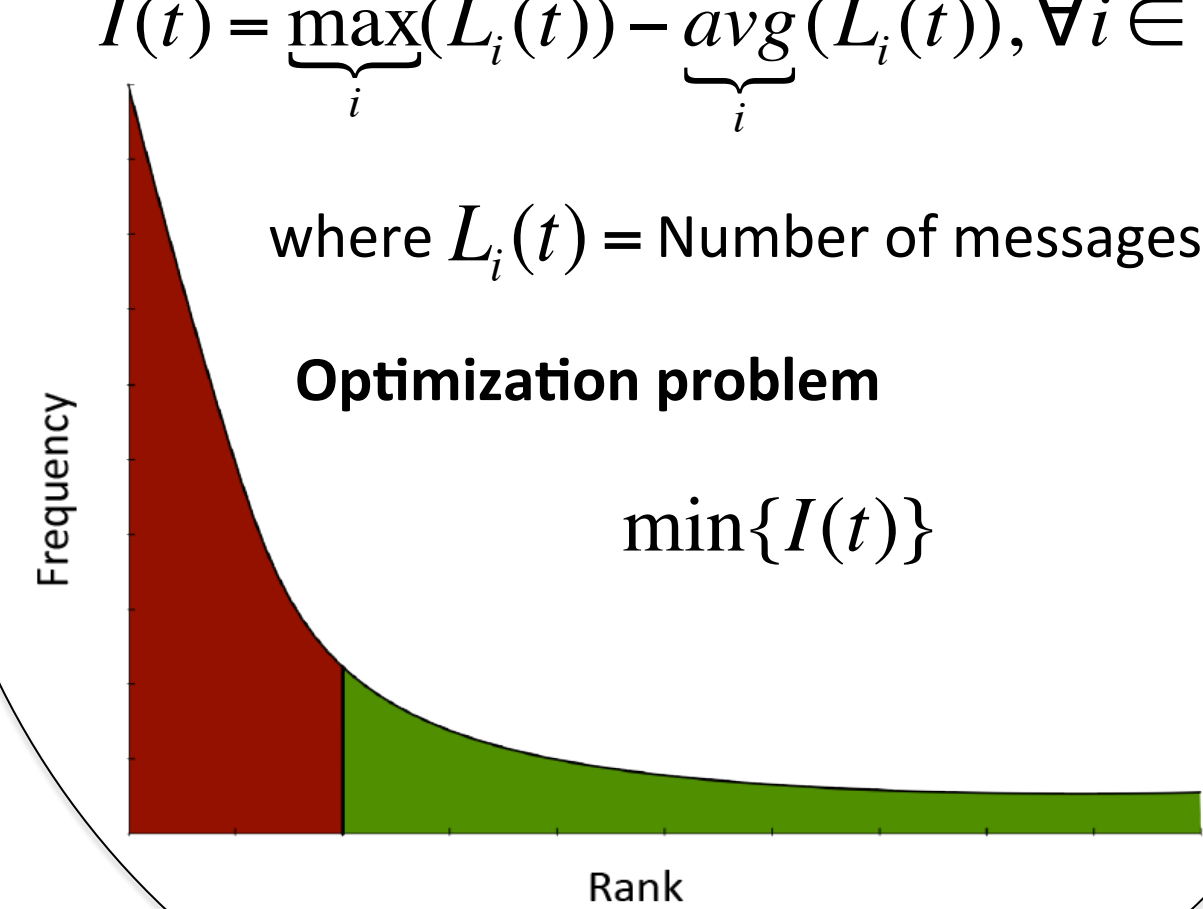
- Given a **highly skewed input stream**, **minimize the load imbalance** across the machines to achieve **better hardware utilization, higher throughput and lower processing latency**
- We propose two novel algorithms : a) **D-Choices** and b) **W-Choices**. Both the algorithms operate by identifying the head of the input stream and placing the head on more than two workers

Imbalance

Problem Statement

Load

Worker



Past Work: ICDE 2015

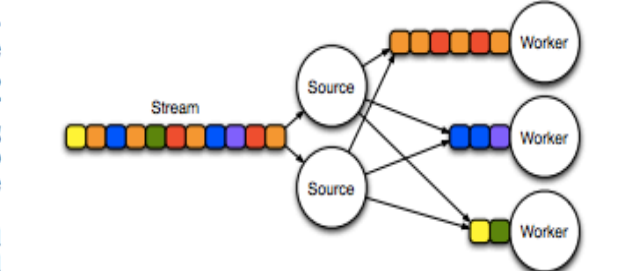
The Power of Both Choices: Practical Load Balancing for Distributed Stream Processing Engines

Muhammad Anis Uddin Nasir¹, Gianmarco De Francisci Morales^{3c}, David Garcia-Soriano³, Nicolas Kourtellis², Marco Serafini³

¹KTH Royal Institute of Technology, Stockholm, Sweden
²Yahoo Labs, Barcelona, Spain
³Qatar Computing Research Institute, Doha, Qatar

¹anisu@kth.se, ²gdfm@acm.org, ³davidgs@yahoo-inc.com
^{3c}kourtellis@yahoo-inc.com, ³msrafini@qf.org.qa

Abstract—We study the problem of load balancing in distributed stream processing engines, which is exacerbated in the presence of skew. We introduce PARTIAL KEY GROUPING (PKG), a new stream partitioning scheme that adapts the classical “power of two choices” to a distributed streaming setting by leveraging two novel techniques: key splitting and local load estimation. In so doing, it achieves better load balancing than key grouping while being more scalable than shuffle grouping.

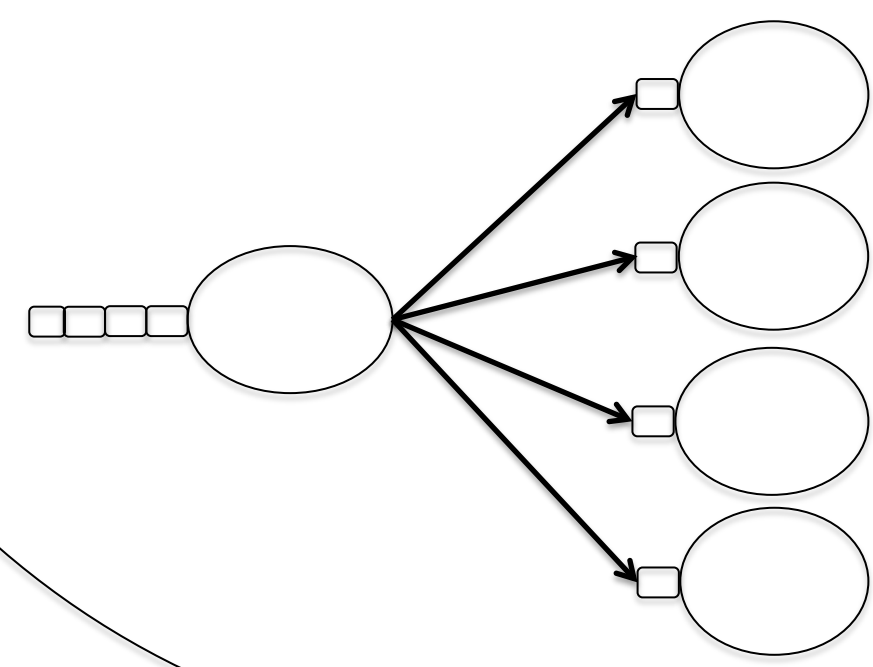
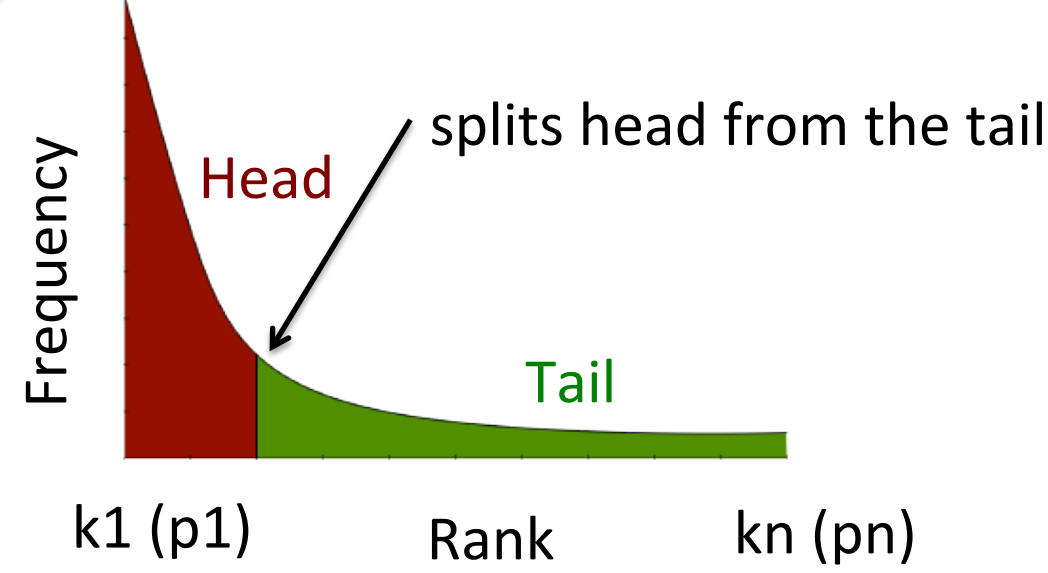


We test PKG on several large datasets, both real-world and synthetic. Compared to standard load balancing, PKG reduces the load imbalance by up to 40% in throughput and up to 45% in latency when deployed on a real Storm cluster.

Fig. 1: Load imbalance generated by skew in the key distribution when using key grouping. The color of each message represents its key.

1. INTRODUCTION

Solution Overview



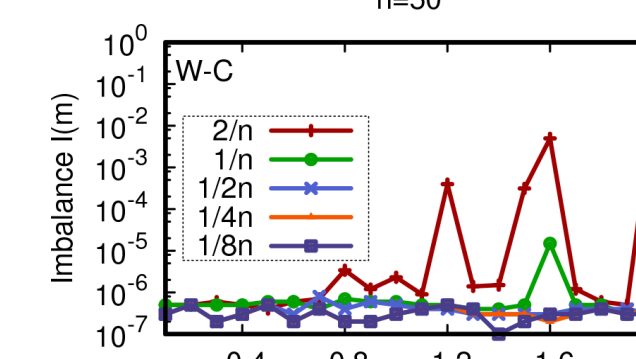
Solution Overview

Observation

PKG guarantees nearly perfect load balance for $p1 \leq 1/(5n)$

Solution

Stream Summary to handle the Head



D-Choices:

- adapts to the frequencies
- allows subset of workers

W-Choices:

- Independent of frequencies
- allows all the workers

Algorithm:

- Assign head to the set of d/w workers
- Handle tail using power of both choices

Solution Overview

Pseudocode

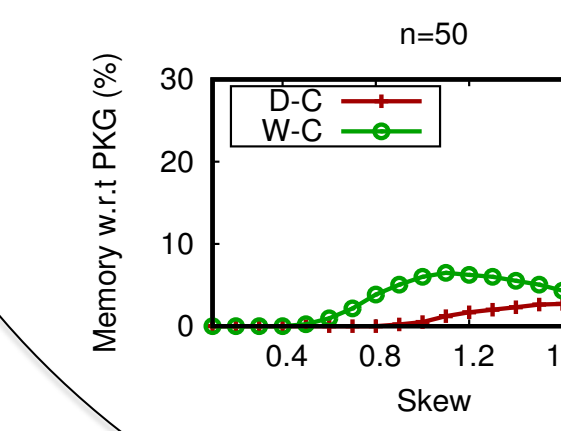
Algorithm 1: Stream partitioning algorithm.

```

upon message  $m = (k, v)$ 
 $\mathcal{H} \leftarrow \text{UPDATE\_SPACE\_SAVING}(k)$ 
 $d \leftarrow 2$  // Default as in PKG
if  $k \in \mathcal{H}$  then
  if D-CHOICES then
     $d \leftarrow \text{FIND\_OPTIMAL\_CHOICES}()$ 
  else if W-CHOICES then
     $d \leftarrow n$ 
 $w \leftarrow \text{MINLOAD}(\mathcal{F}_1(k), \dots, \mathcal{F}_d(k))$ 
send( $w, m$ )

```

Memory Overhead



Experimental Setup

Dataset

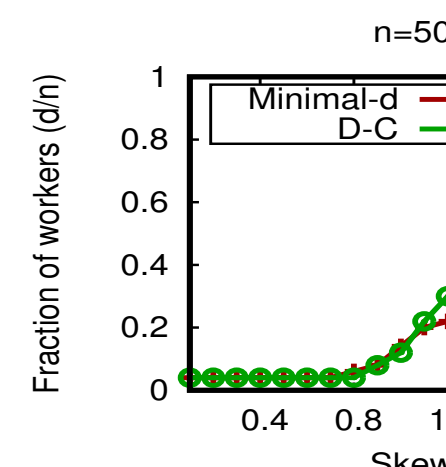
Dataset	Symbol	Messages	Keys	$p_1(\%)$
Wikipedia	WP	22M	2.9M	9.32
Twitter	TW	1.2G	31M	2.67
Cashtags	CT	690k	2.9k	3.29
Zipf	ZF	10^7	$10^4, 10^5, 10^6$	$\propto \frac{1}{\sum x^{-\alpha}}$

Algorithms

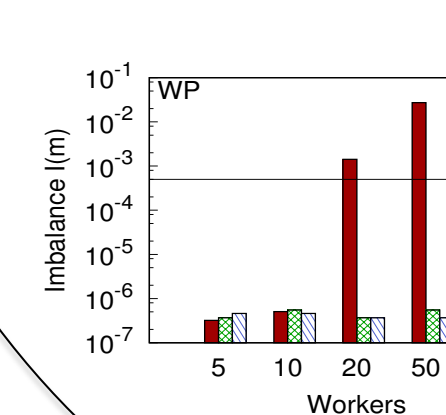
Symbol	Algorithm	Head vs. Tail
D-C	D-Choices	Specialized on head
W-C	W-Choices	
RR	Round-Robin	
PKG	Partial Key Grouping	Treats all keys equally
SG	Shuffle Grouping	

Experimental Results

Estimation

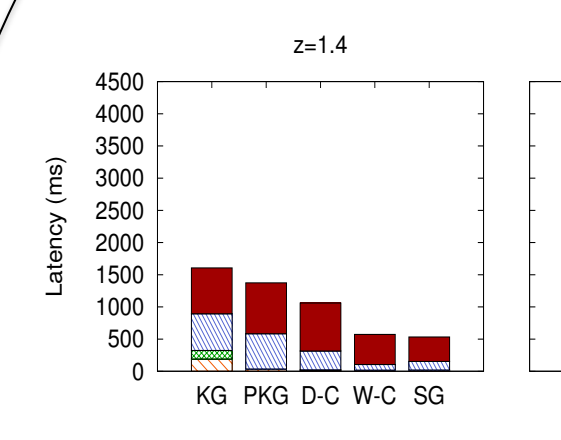


Load Imbalance

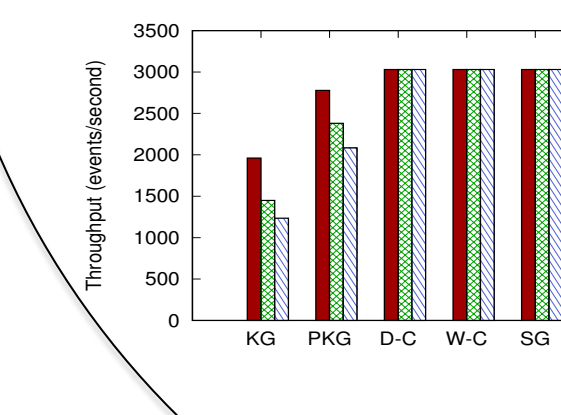


Experimental Results

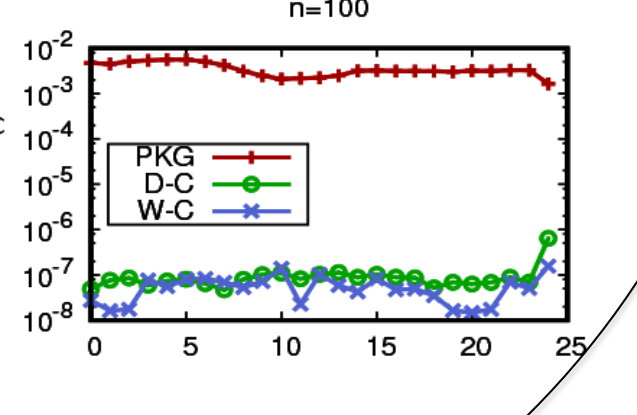
Latency



Throughput



Imbalance over time



Analysis

$$\text{minimize}_d f(d; \mathcal{D}, \theta) = d \times |\mathcal{H}_{\mathcal{D}, \theta}|$$

$$\text{subject to } \mathbb{E}[I(m)] \leq \epsilon.$$

Expanding the constraint we get:

$$\sum_{i \in \mathcal{H}} p_i + \left(\frac{b}{n}\right)^d \sum_{h < i \leq |H|} p_i + \left(\frac{b}{n}\right)^2 \sum_{i > |H|} p_i \leq \left(\frac{b}{n}\right) + \epsilon$$

where

$$b = n - n \left(\frac{n-1}{n}\right)^{h \times d}$$

Conclusion

- We propose two algorithms to achieve load balance at scale for DSPEs
- Use heavy hitters to separate the head of the distribution and process on larger set of workers
- Improvement translates into 150% gain in throughput and 60% gain in latency over PKG

Future Work

- Dynamic Load Balancing for stateful operators
 - Key Migration
 - Partition Migration
 - Queuing Theory
- Load Balancing in Heterogeneous Cluster
 - Load Prediction
 - Worker Communication