Transforming Blocks of Data with DStreams

Janani Ravi
CO-FOUNDER, LOONYCORN

www.loonycorn.com

Overview

Differentiate between stateful and stateless transformations

Apply stateful transformations

- across all entities in the stream
- on a window of entities in a stream

Implement these transformations using Python

Stream Transformations

Stateless

Transformations which are applied on a single RDD

Stateful

Transformations which accumulate across multiple RDDs

Stateless Transformations

Batch processing from a file

All data available in a single RDD

map(), reduceByKey(), filter()

All transformations used so far have been stateless

Stateful Transformations

Apply to streaming data

Include data from more than one RDD

Accumulate data across a longer time interval

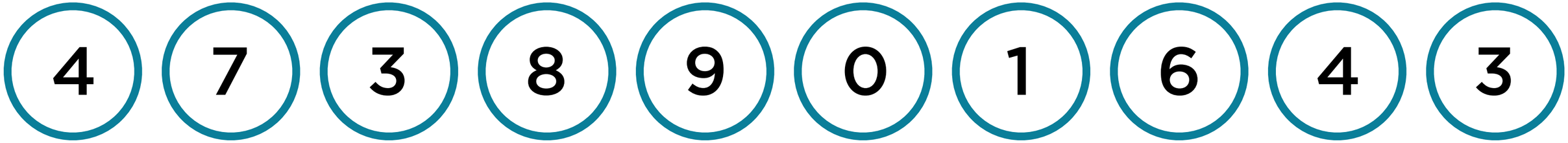
- **entire stream**
- **window**

Stateful Transformations

Accumulate information across all or a window of entities in a stream

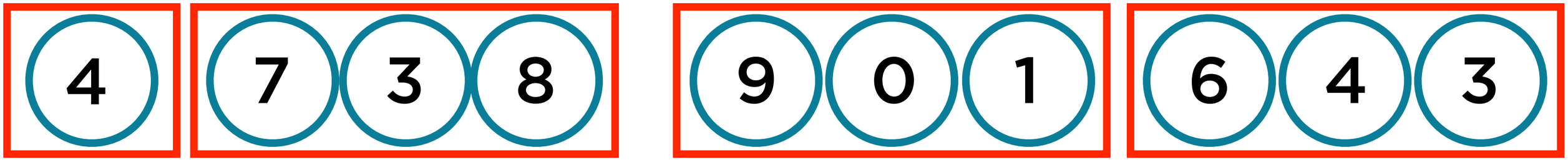
Summarizing Data in a Stream

Summarizing Stream Data



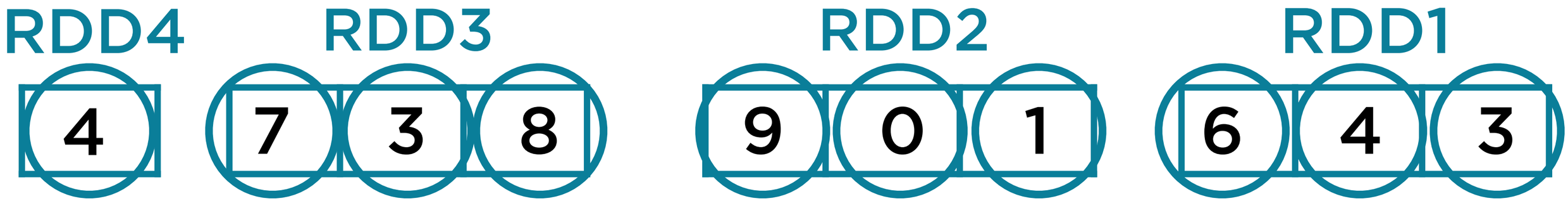
A stream of integers

Summarizing Stream Data



Grouped into batches

Summarizing Stream Data

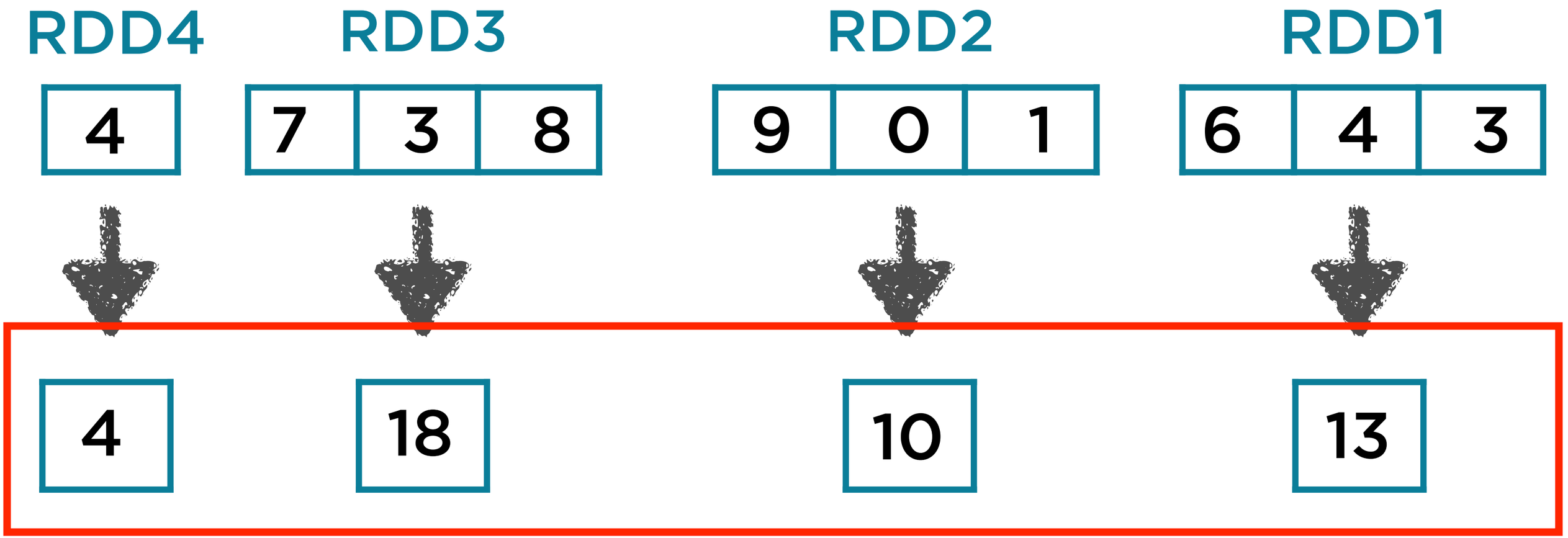


Each group is an RDD within the DStream

Summarizing Stream Data

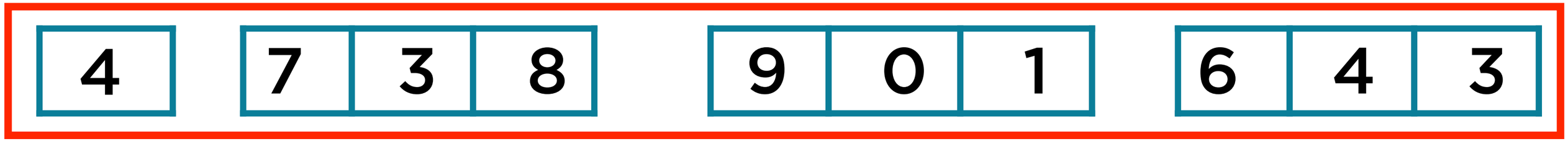
Apply the sum() operation to the DStream

Summarizing Stream Data



The result is another DStream

Summarizing Stream Data

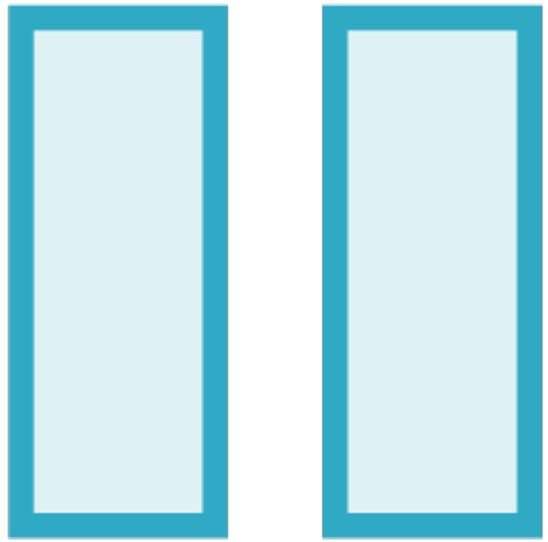


What if you want to keep a **running total** of all integers in the stream?

Pair RDDs have a method called **updateStateByKey(updateFn)**

This summarizes data across all elements in a stream

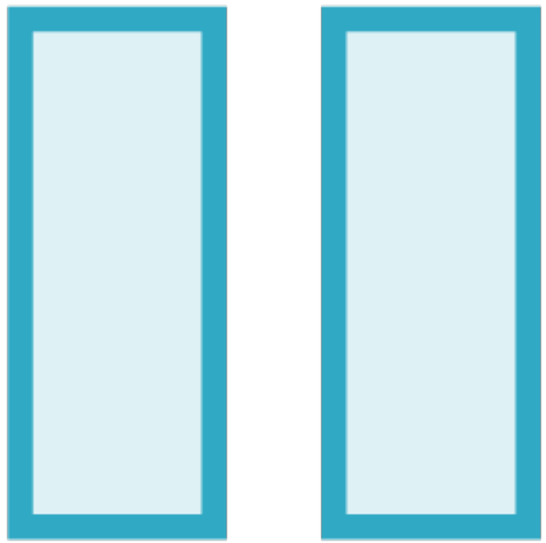
Pair RDDs



Pair RDDs

Every element in the data set is a key-value pair

Pair RDDs



Every element in the dataset is a key-value pair

Represented as a tuple in Python i.e. (word, count)

Special transformations apply to pair RDDs

- keys(), values()
- groupByKey()
- reduceByKey()

Summarizing Stream Data

updateStateByKey(fn)

Apply a summarizing operation on all values with the same key in a DStream

Pair RDDs

Initialize the state to some value

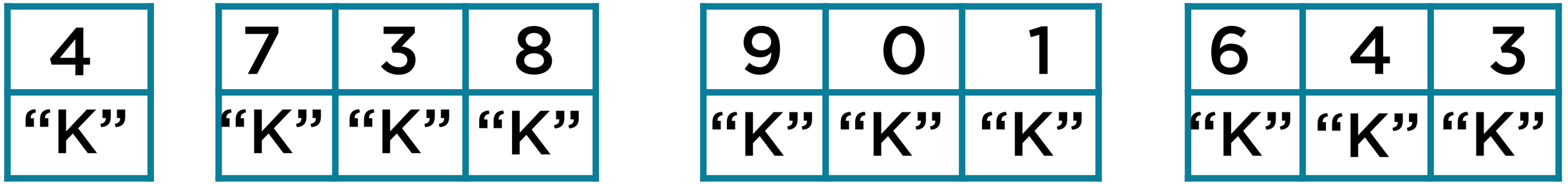
Specify an update function which updates the current state based on the values in the stream

Apply this update function to all existing keys

Summarizing Stream Data

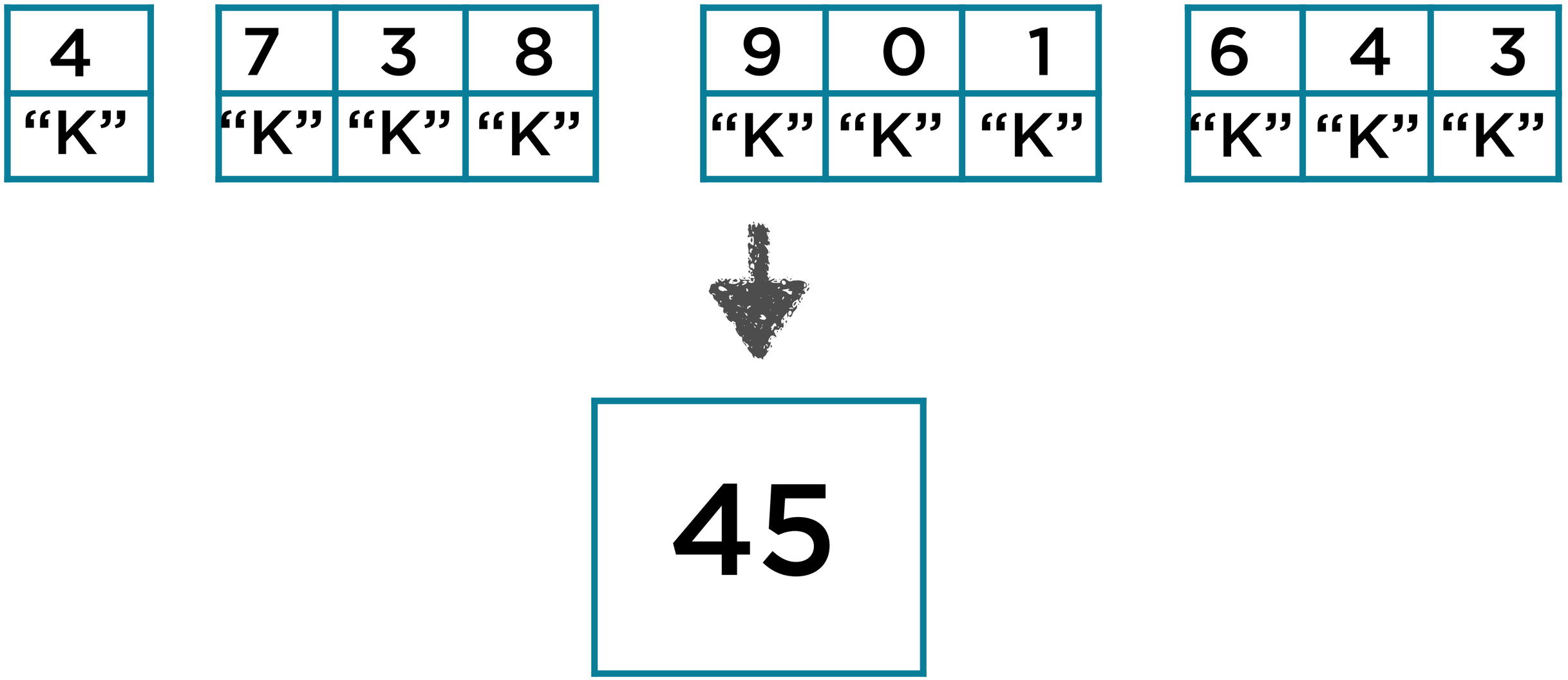
Assign the same key to each integer to make every element a key-value pair

Summarizing Stream Data



The updateStateByKey() will sum all values with the same key

Summarizing Stream Data

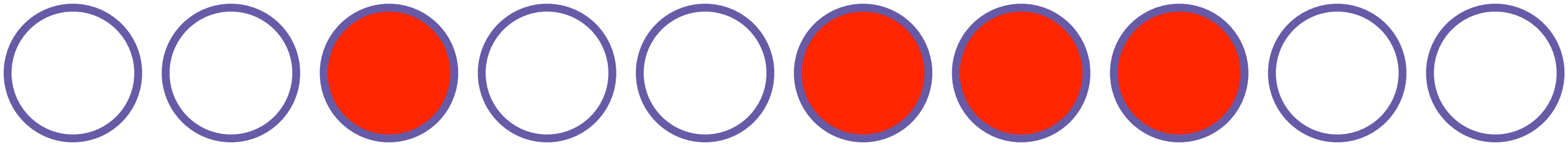


Demo

Apply updateStateByKey() on a DStream to count the number of occurrences of each word in a text stream

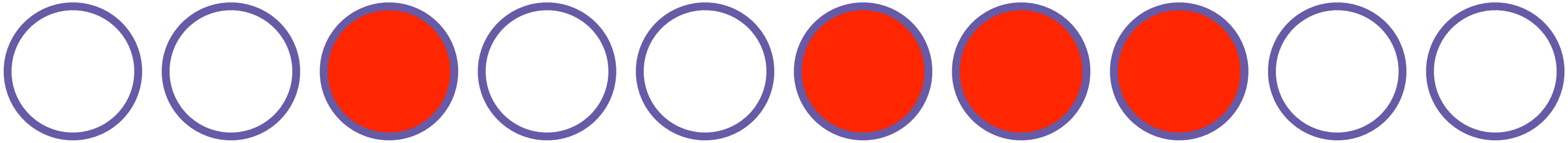
Summarizing Data Over a Window

Window Operations



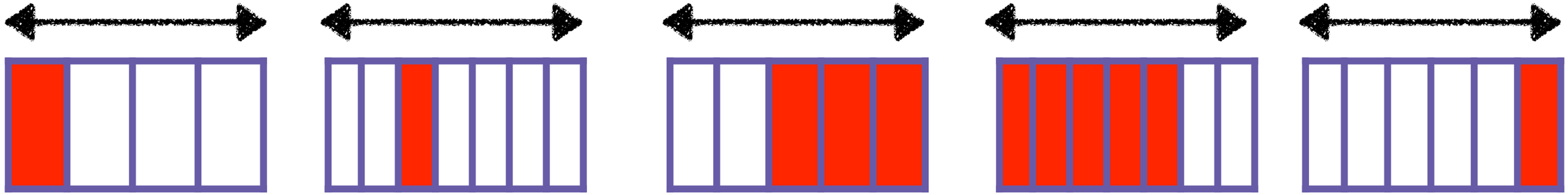
A stream of logs for a website

Window Operations



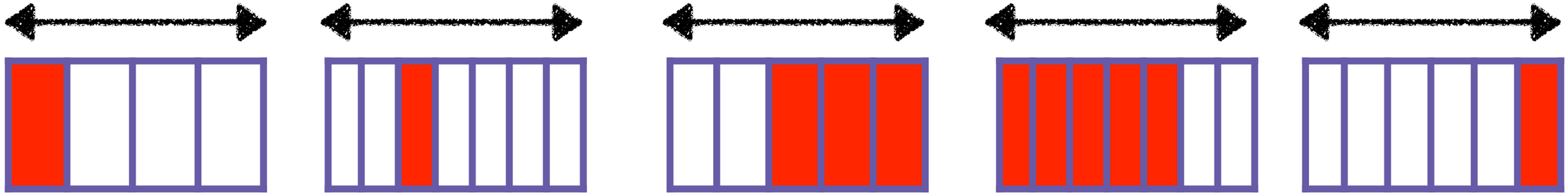
Check whether a new deployment caused a spike in errors

Window Operations



Messages are grouped into RDDs based on a batchInterval

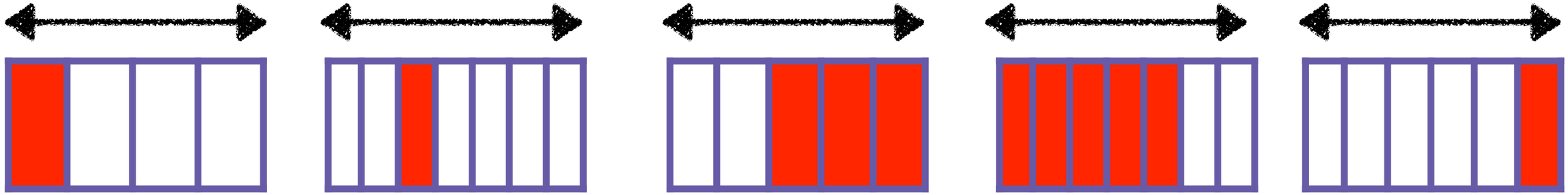
Window Operations



batchInterval ~ short duration of time

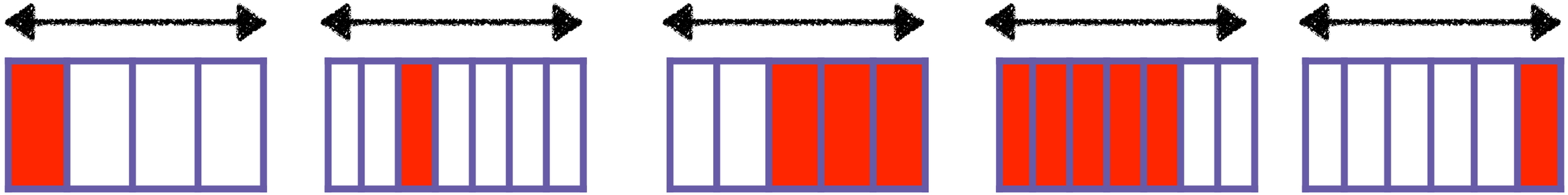
e.g. 10 seconds

Window Operations



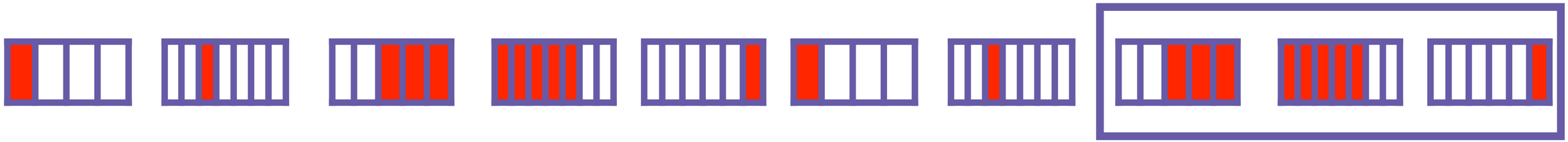
Certain kinds of analysis are best over longer durations

Window Operations



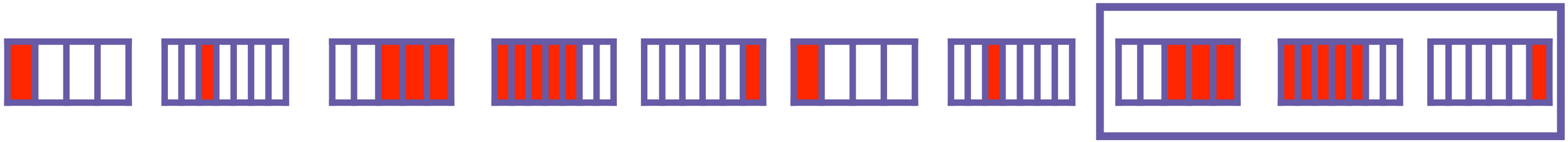
Cumulative error rate over a 30 second timeframe

Window Operations



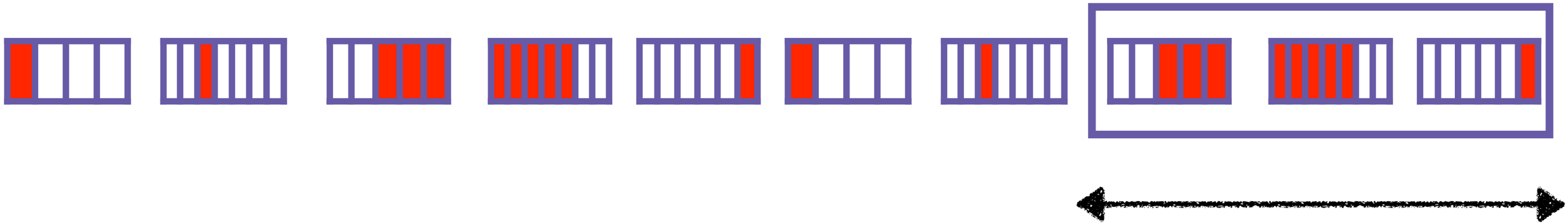
Cumulative error rate over a 30 second timeframe

Window Operations



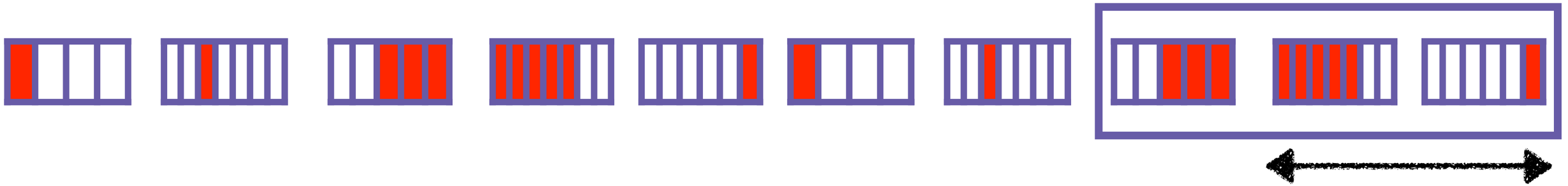
Define a sliding window
for cumulative summaries

Window Operations



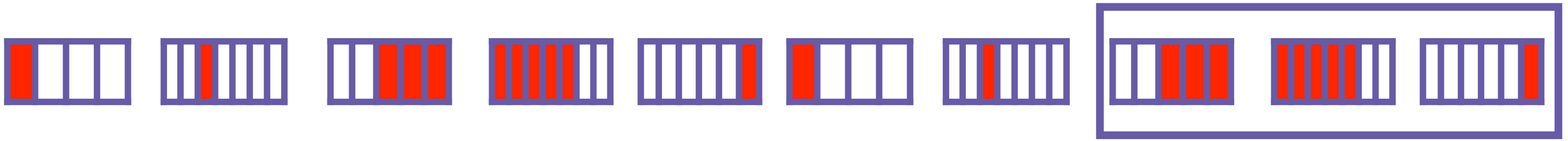
Window size = 30 seconds

Window Operations



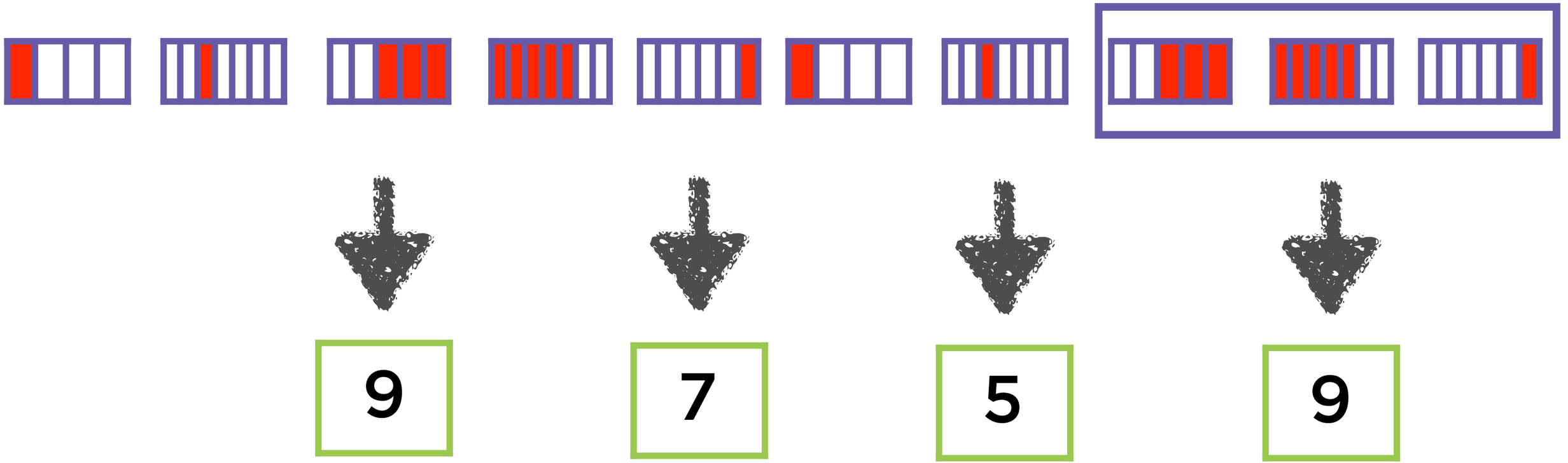
Sliding interval = 20 seconds

Window Operations



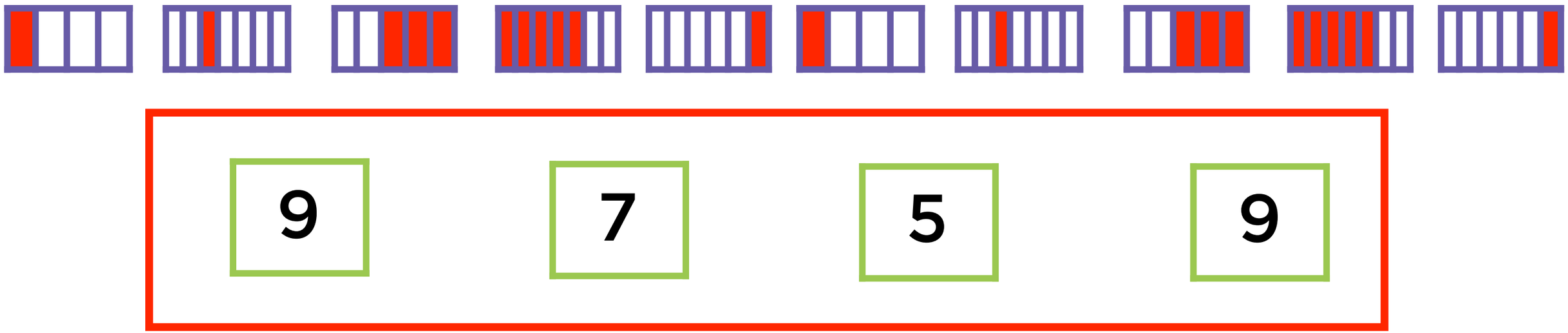
All RDDs within a window are treated as a single combined RDD

Window Operations



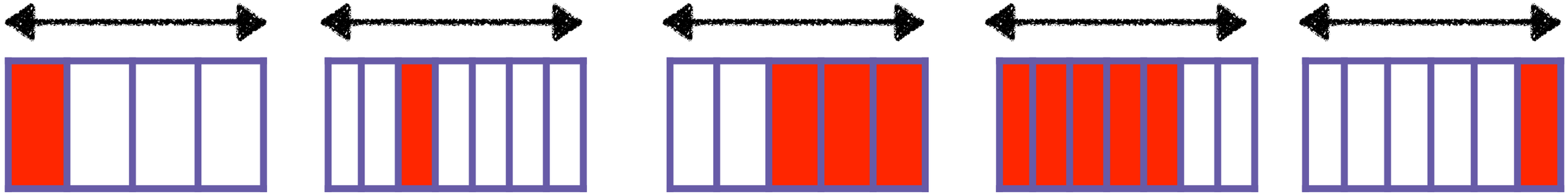
countByWindow()

Window Operations



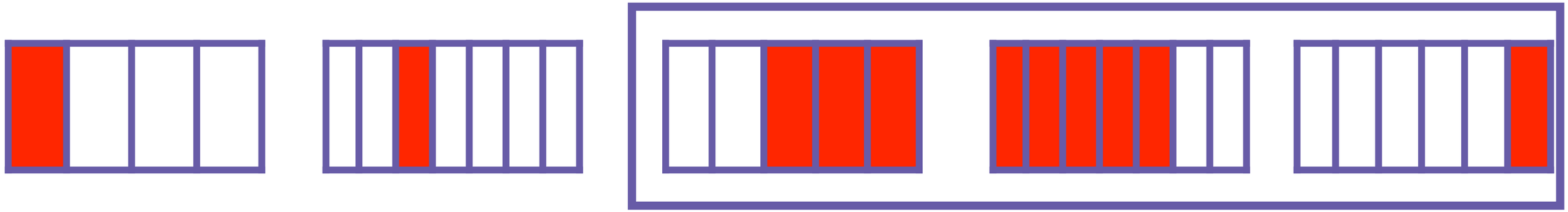
Resultant DStream where the summary is accumulated per window

Window Operations



Batch interval = 10 seconds

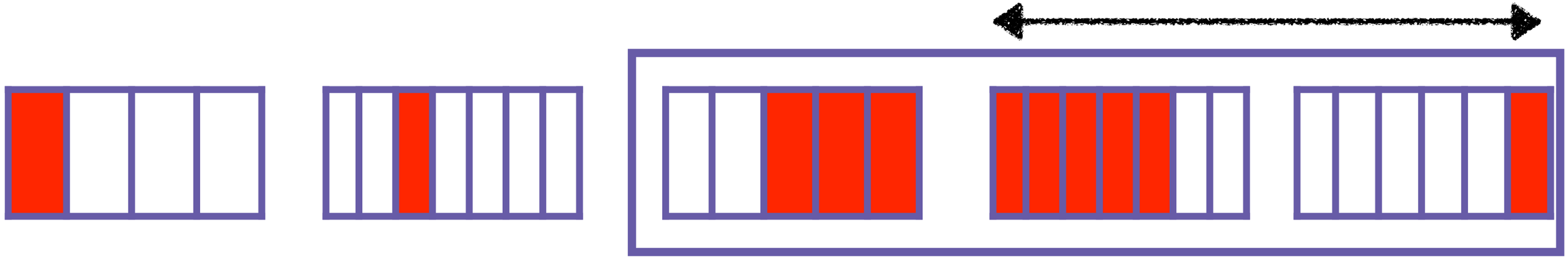
Window Operations



Batch interval = 10 seconds

Window size = 30 seconds

Window Operations



Batch interval = 10 seconds

Window size = 30 seconds

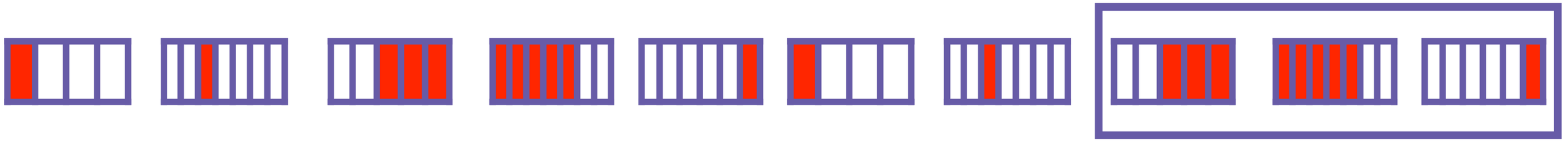
Sliding interval = 20 seconds

Demo

Count the number of messages in a window interval

Count the number of error messages in a window

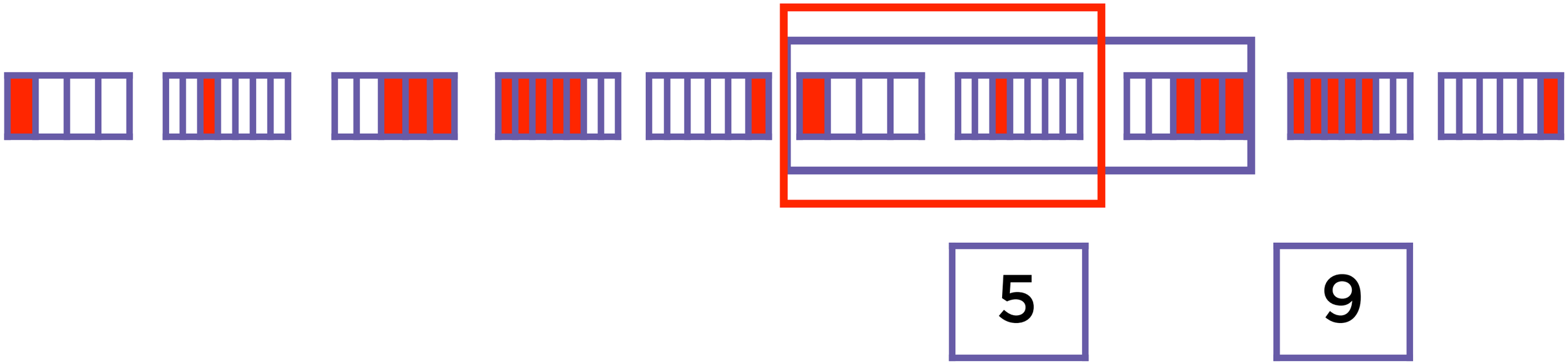
Summary and Inverse Functions



9

The summary function performs the sum() operation

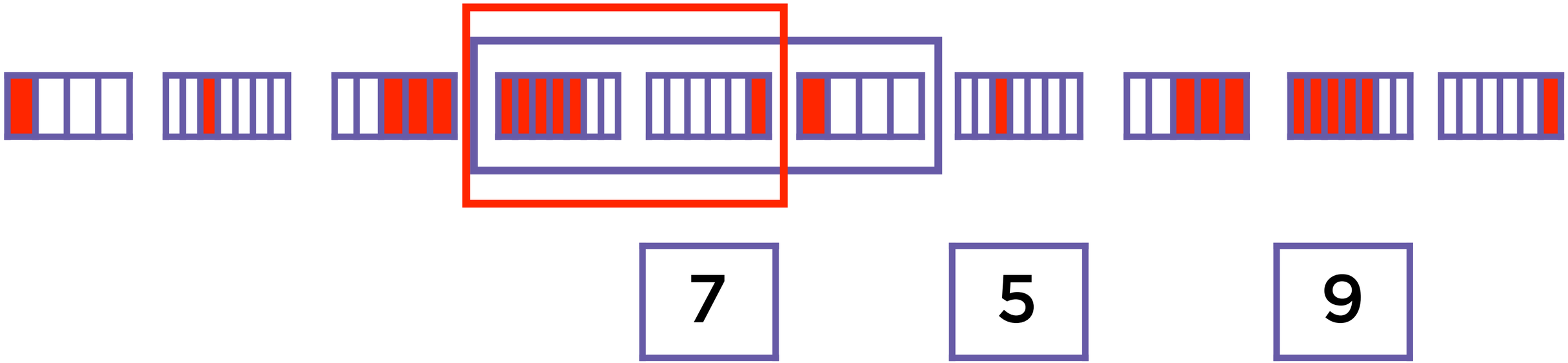
Summary and Inverse Functions



2 RDDs entered the window 9 + 2 = 11

2 RDDs left the window 11 - 6 = 5

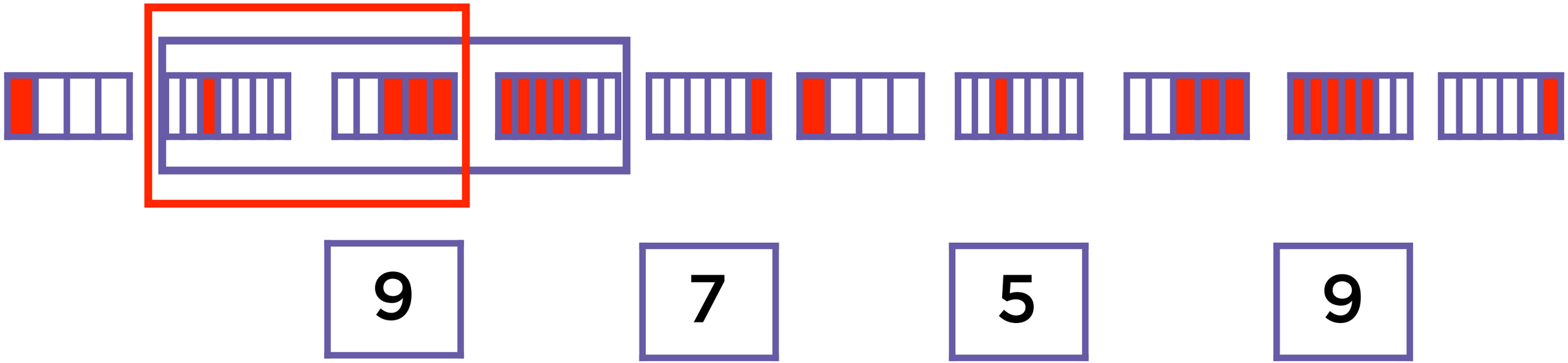
Summary and Inverse Functions



2 RDDs entered the window 5 + 6 = 11

2 RDDs left the window 11 - 4 = 7

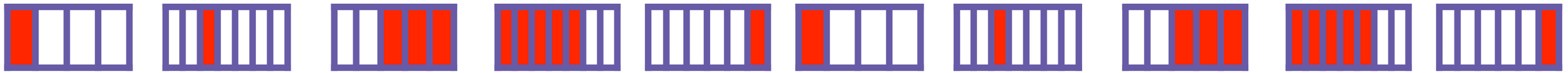
Summary and Inverse Functions



2 RDDs entered the window 7 + 4 = 11

2 RDDs left the window 11 - 2 = 9

Summary and Inverse Functions



2 RDDs entered the window — Summary function

2 RDDs left the window — Inverse function

Demo

Sum the integers received in a window interval

Demo

Count the different types of error messages received from a stream of logs

Overview

Stateful transformations across all entities in a stream

Stateful transformations using window operations

Implementing these in streaming applications using Python