Performance

Ajay Kumar Yadav

Strategic Cloud Engineers,
Google Cloud

CD

Agenda

Course Intro

Monitoring

Logging and Error Reporting

Troubleshooting and Debugging

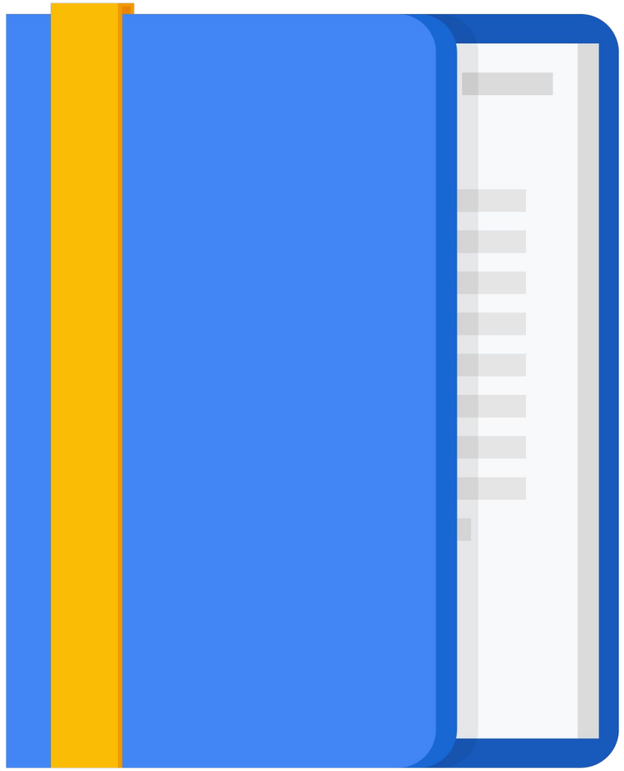
Performance

Testing and CI/CD

Reliability

Flex Templates

Course Summary



Google Cloud

CD

Performance

Agenda

Pipeline design

Data shape

Sources, sinks & external systems

Shuffle & Streaming Engine

Performance

Agenda

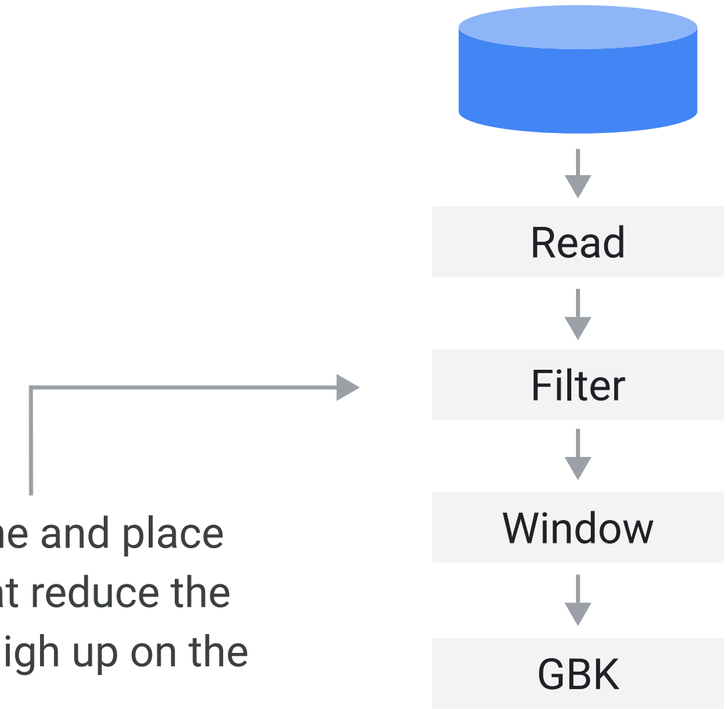
Pipeline design

Data shape

Sources, sinks & external systems

Shuffle & Streaming Engine

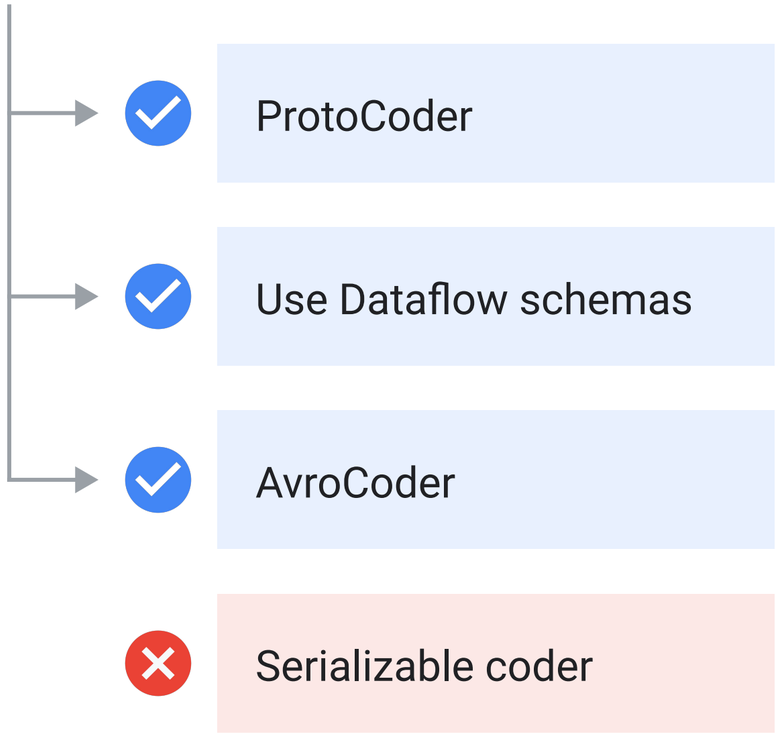
Design: Topology



Filter early in pipeline and place transformations that reduce the volume of data as high up on the graph as possible.

Design: Coders

Utilize efficient coders



Annotating a custom data type [Java]

```
@DefaultCoder(AvroCoder.class)
public class MyCustomDataType {
  ...
 }
```

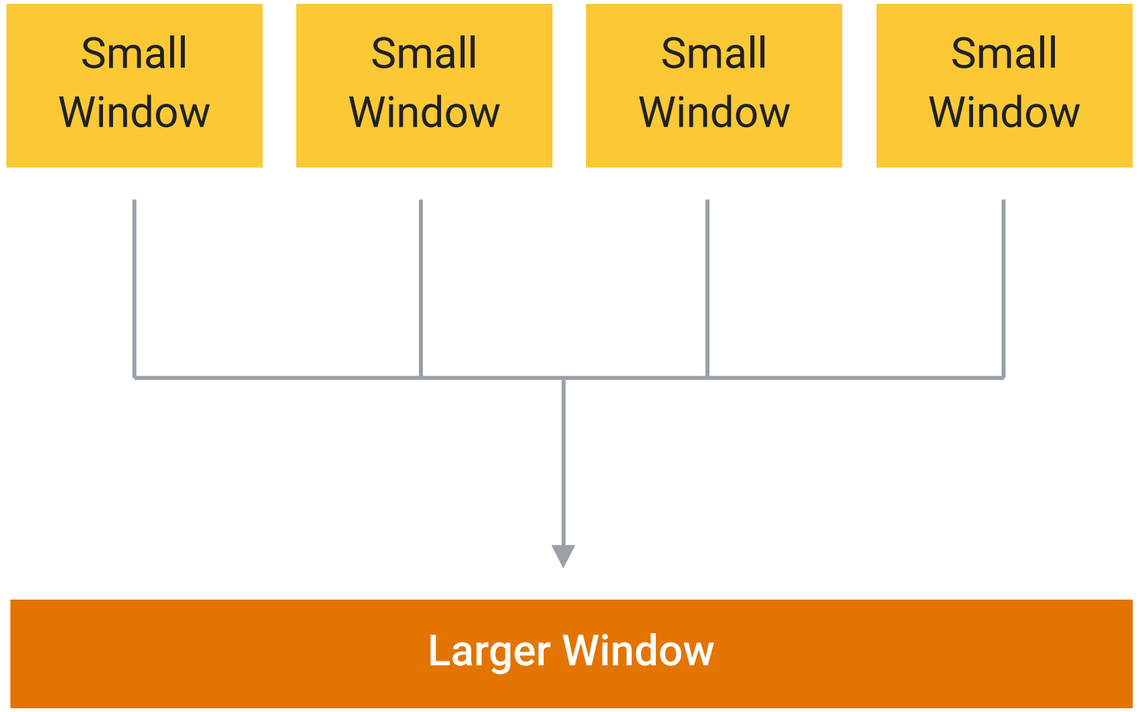
Set a default coder [Java]

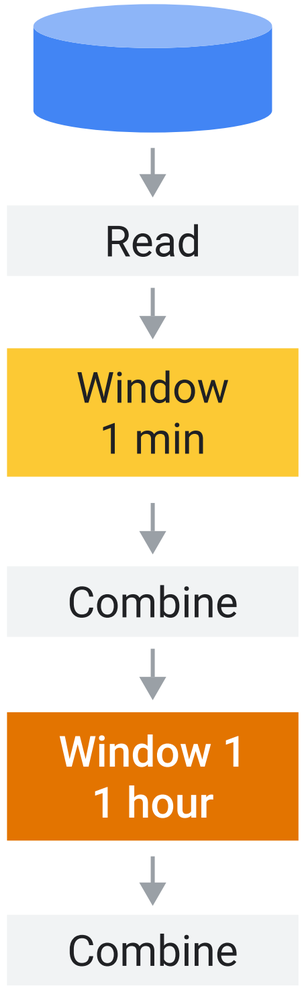
```
PipelineOptions options =
PipelineOptionsFactory.create();
Pipeline p = Pipeline.create(options);


CoderRegistry cr = p.getCoderRegistry();
cr.registerCoder(Integer.class,
BigEndianIntegerCoder.class);
```

Set a default coder [Python]

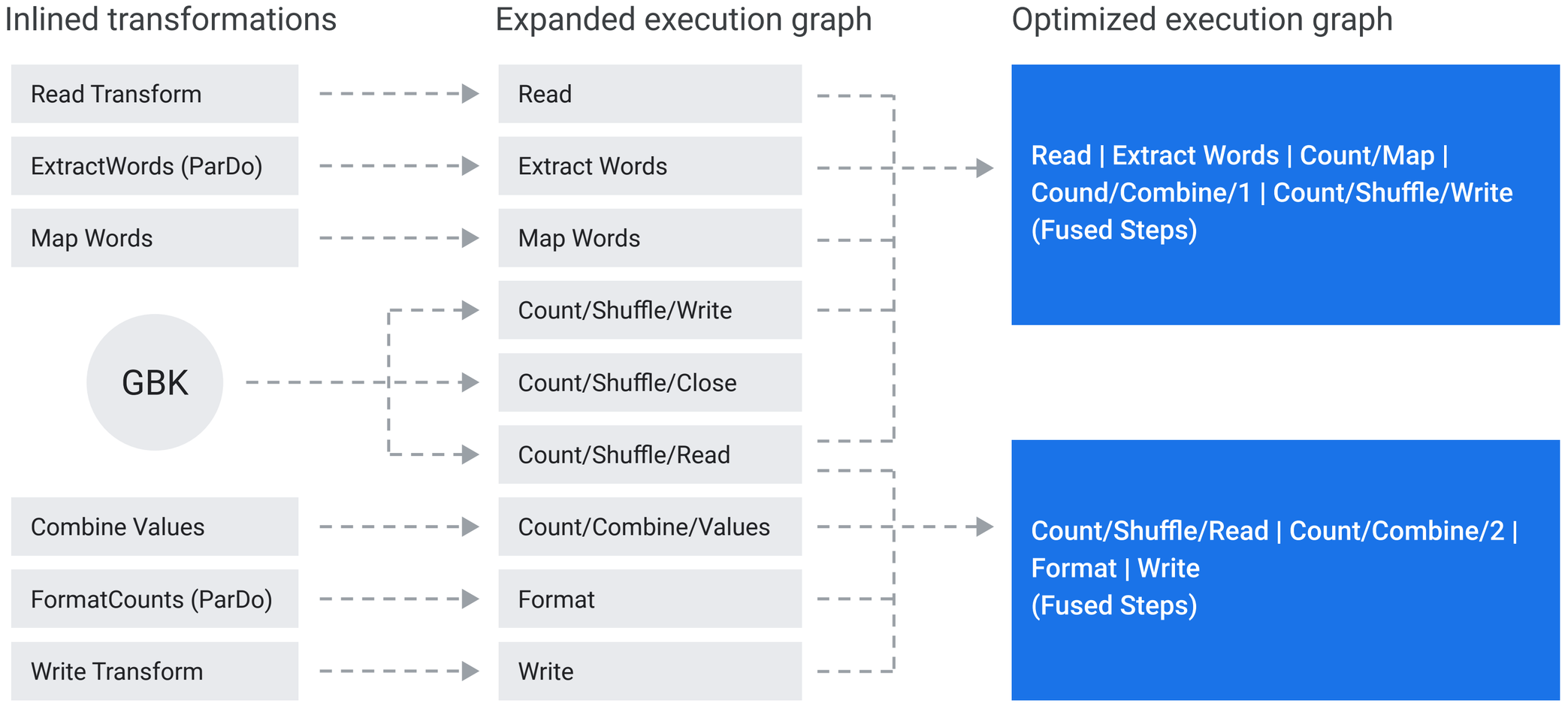
```
apache_beam.coders.registry.register_coder(i
nt, BigEndianIntegerCoder)
```

Design: Window considerations



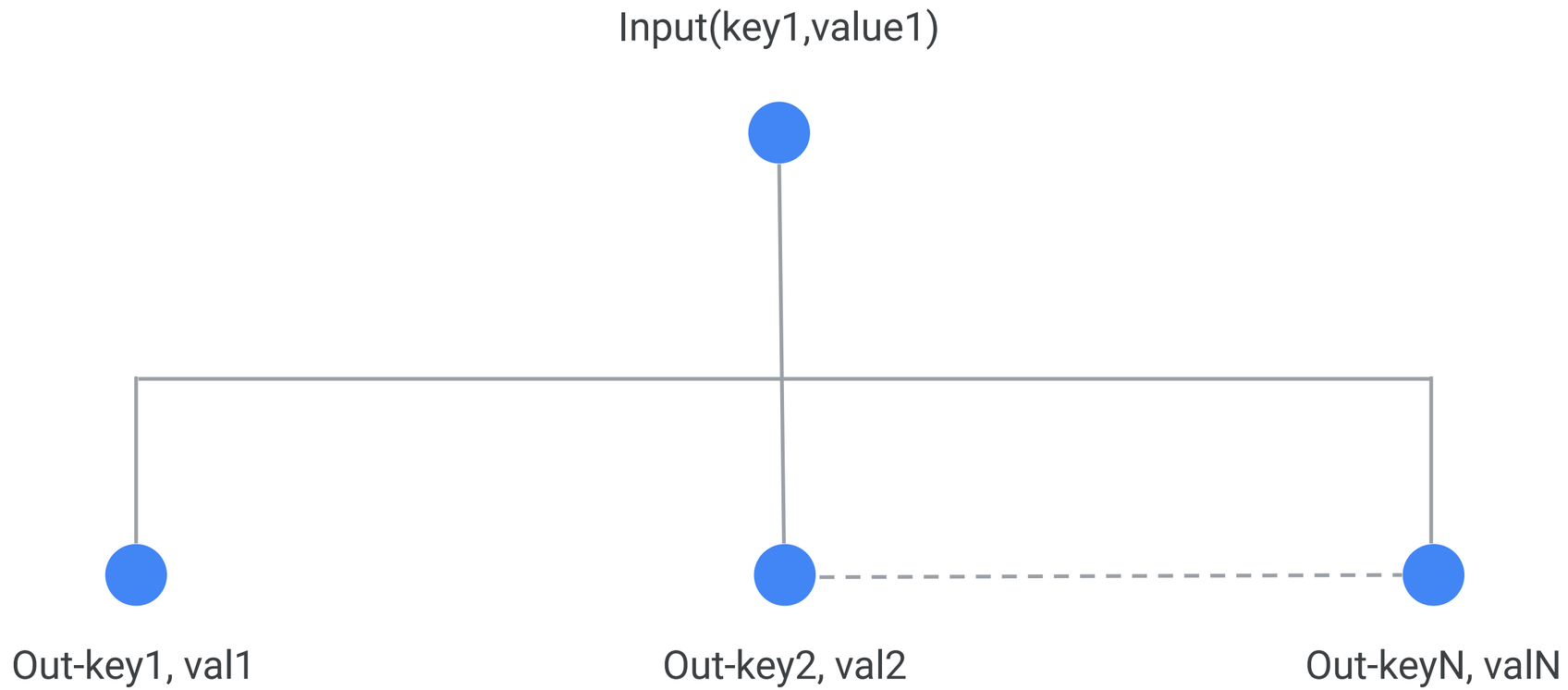


Design: Graph optimization



Design: Graph optimization

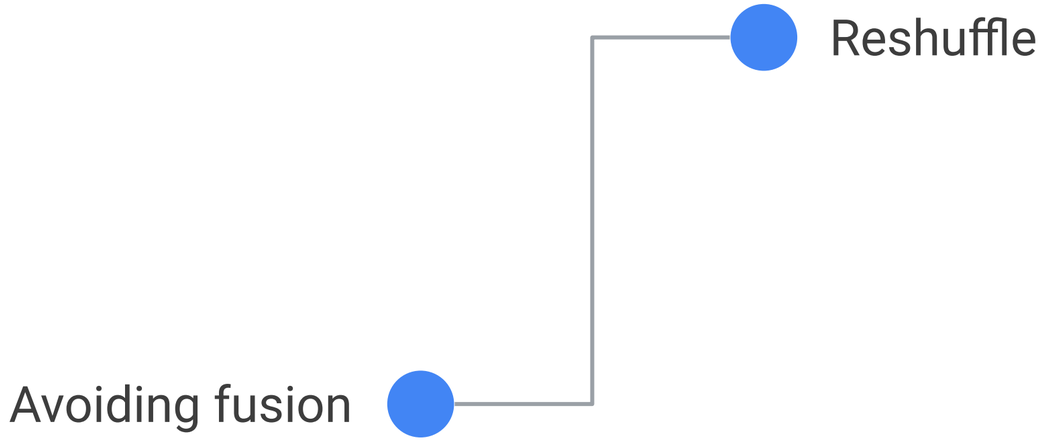
Fanout transformation



Design: Graph optimization

Avoiding fusion

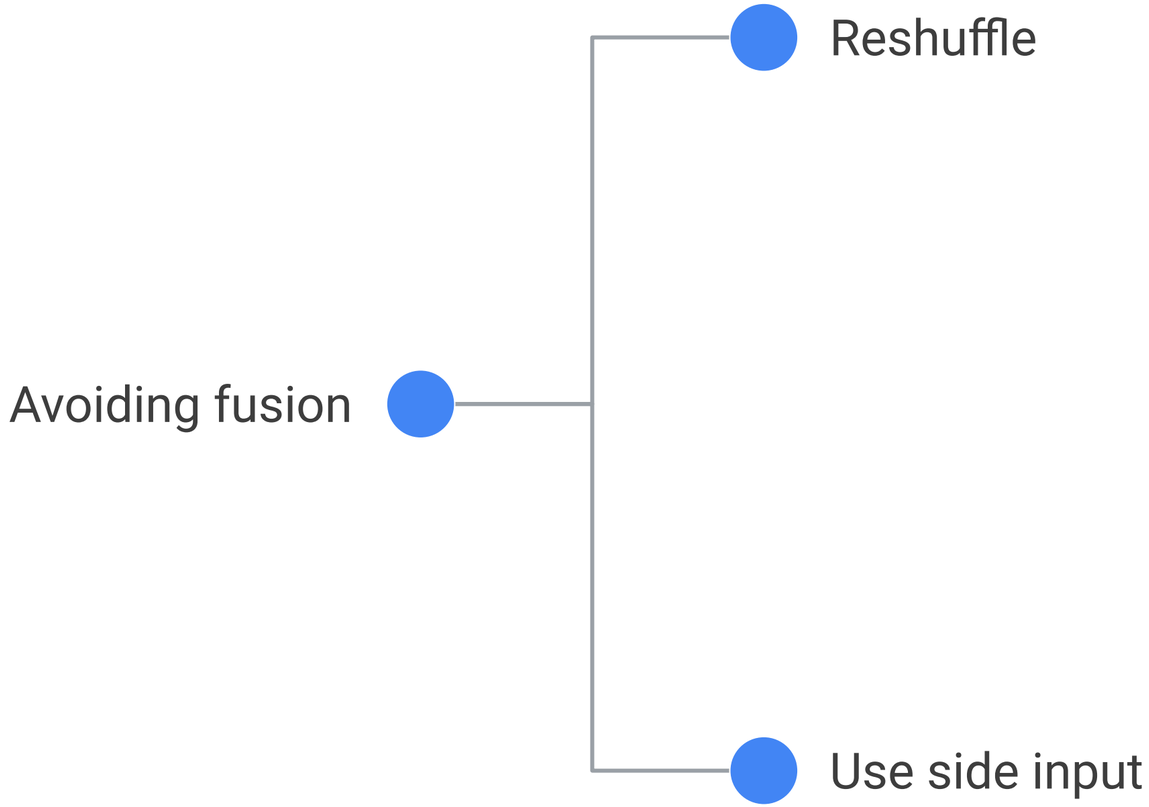
Design: Graph optimization




```
output_collection = output_collection |
'Reshuffle' >> beam.Reshuffle()
```

```
input.apply("Reshuffle",Reshuffle.viaRandomKey())
```

Design: Graph optimization




```
output_collection = output_collection |
'Reshuffle' >> beam.Reshuffle()
```

```
input.apply("Reshuffle",Reshuffle.viaRandomKey())
```

```
messages.apply(ParDo.of(new DoFn<PubsubMessage,
TableRow>() {
@ProcessElement
public void processElement(ProcessContext c) {
TableRow sideInputData =
c.sideInput(mapData).get(key);
// Do something with side input
).withSideInputs(mapData));
```

```
result = ( main_input | 'ApplyCrossJoin' >>
beam.FlatMap(cross_join,rights=beam.pvalue.AsIter
(side_input)))
```

Design: Logging

1 Logging considerations

Design: Logging

1	Logging considerations	• Strike a balance b/w excessive logging vs no/little logging at all.

Design: Logging

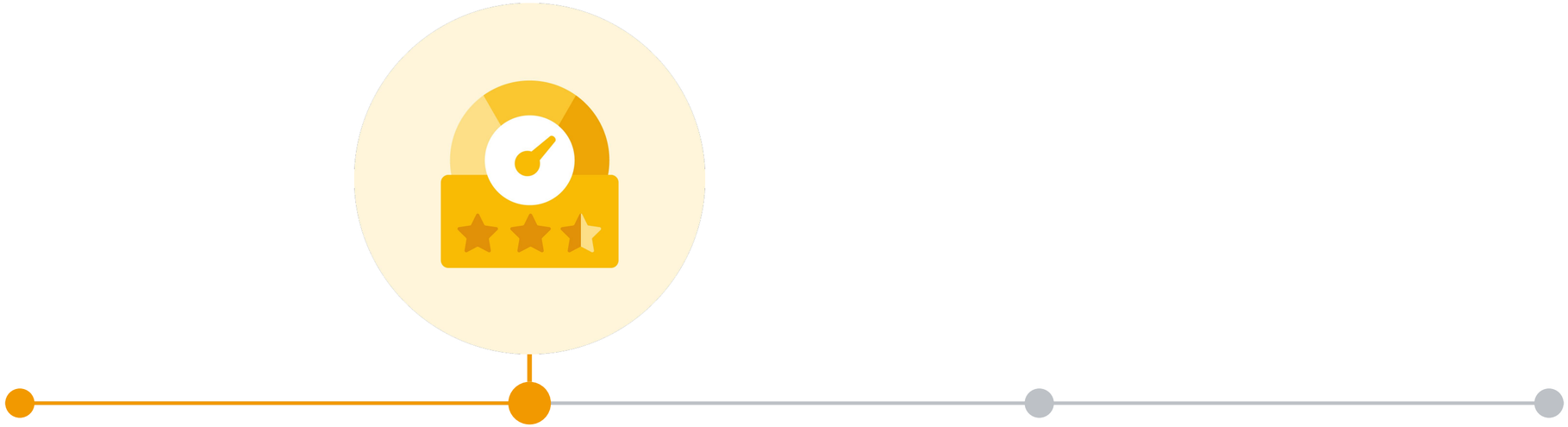
1	Logging considerations	• Strike a balance b/w excessive logging vs no/little logging at all. • Avoid logging at info level against PCollection element granularity.

Design: Logging

1	Logging considerations	• Strike a balance b/w excessive logging vs no/little logging at all. • Avoid logging at info level against PCollection element granularity.
2	Use a dead letter pattern	• Use a dead letter pattern followed by a count per window (ex: 5 mins) for reporting data errors.

Performance

Agenda



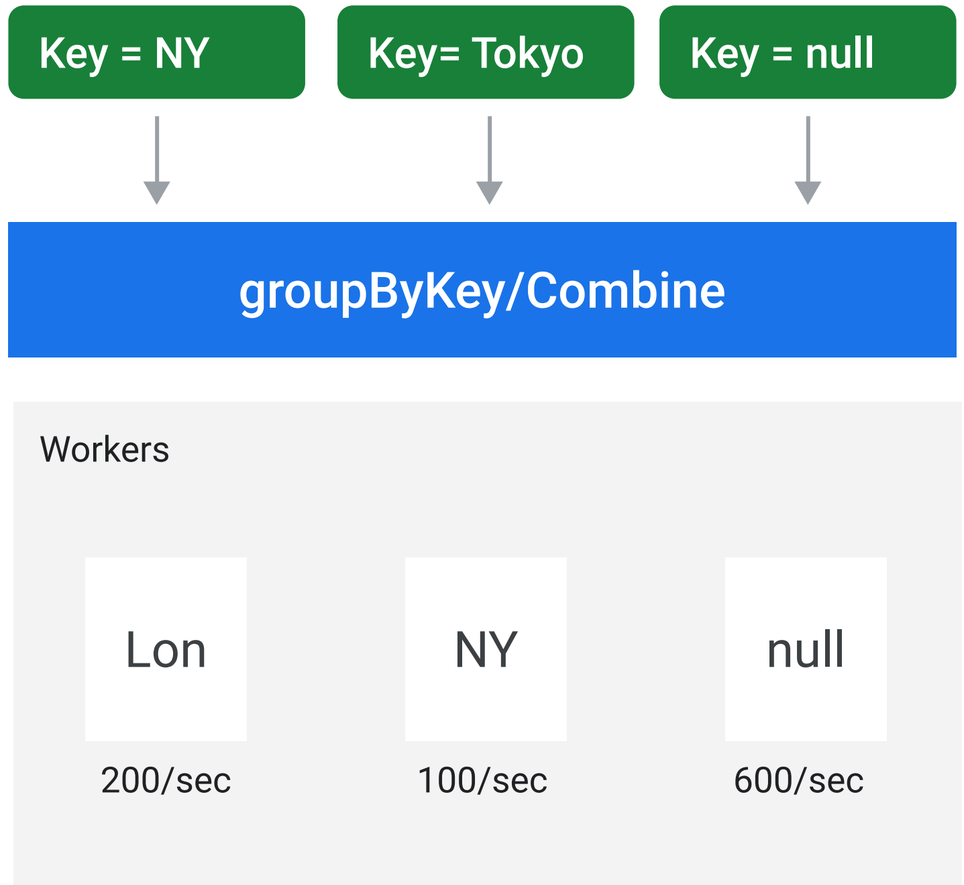
Pipeline design

Data shape

Sources, sinks & external systems

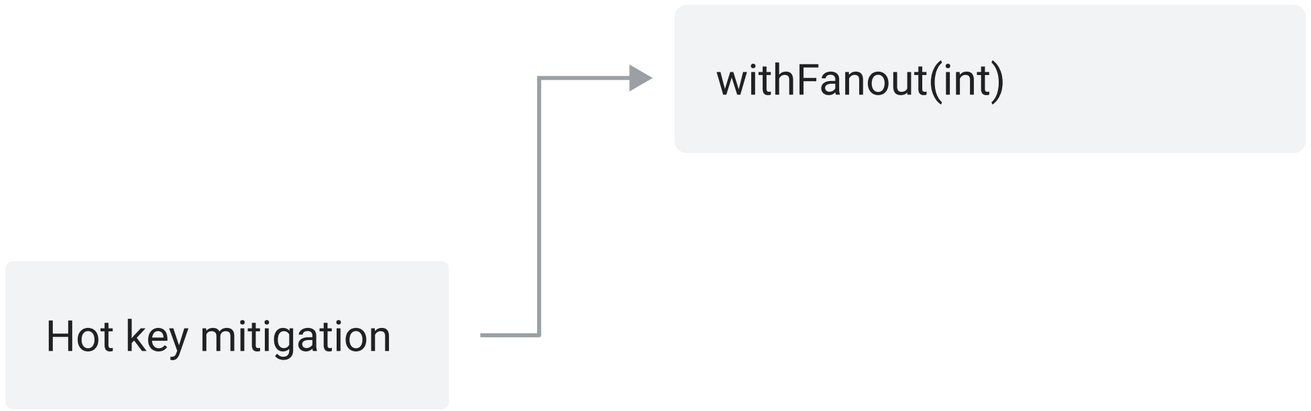
Shuffle & Streaming Engine

Design: Data skew



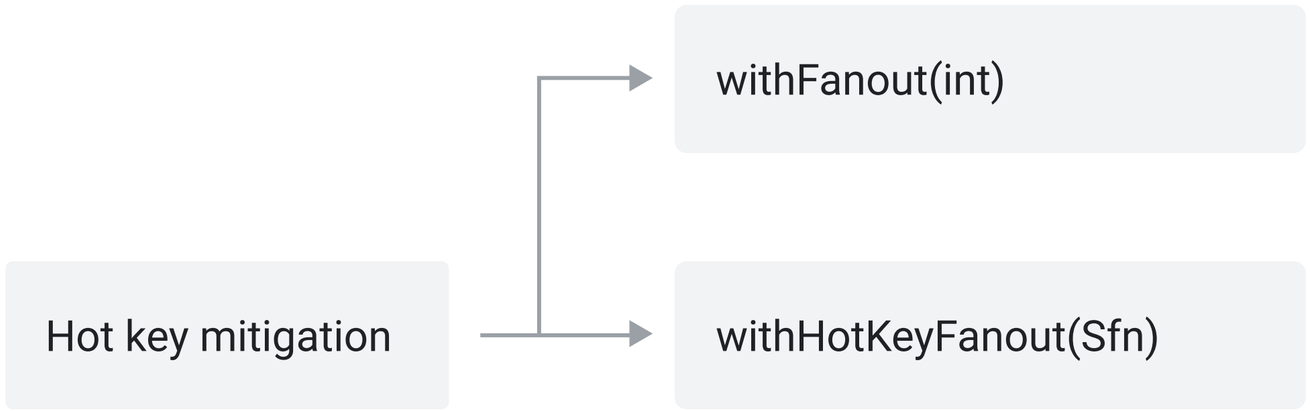
Design: Data skew

Log hotkeys by setting hotKeyLoggingEnabled to true



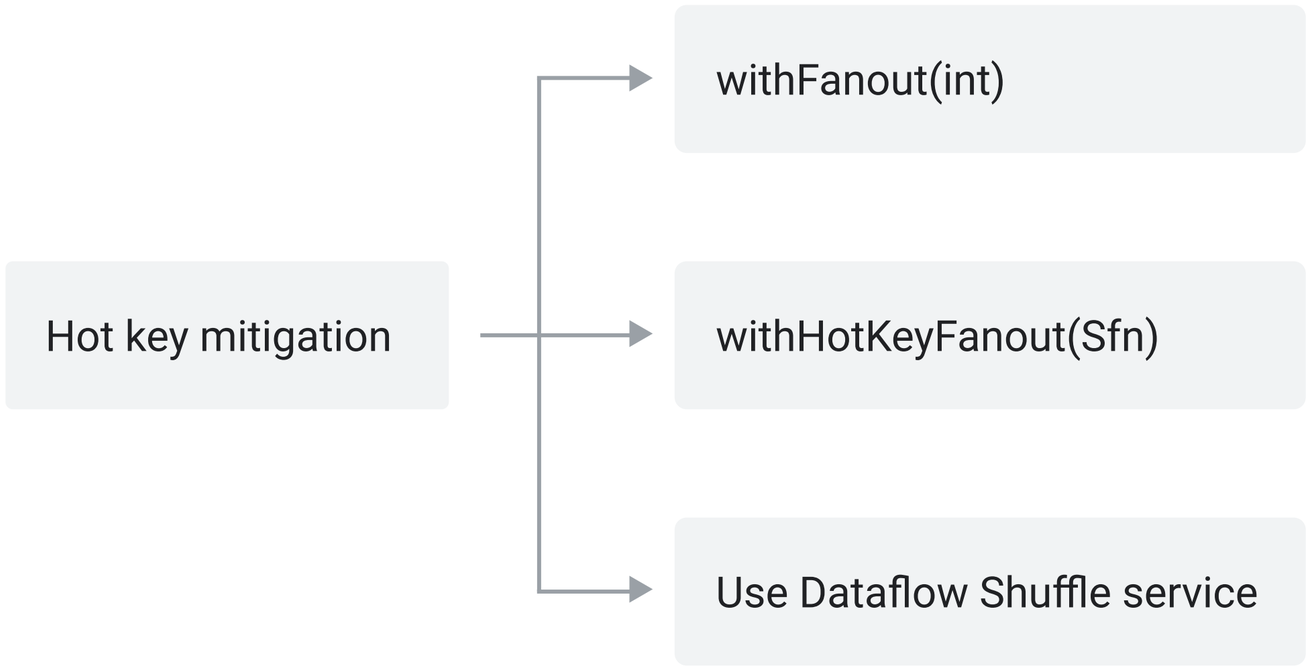
Design: Data skew

Log hot keys by setting hotKeyLoggingEnabled to true



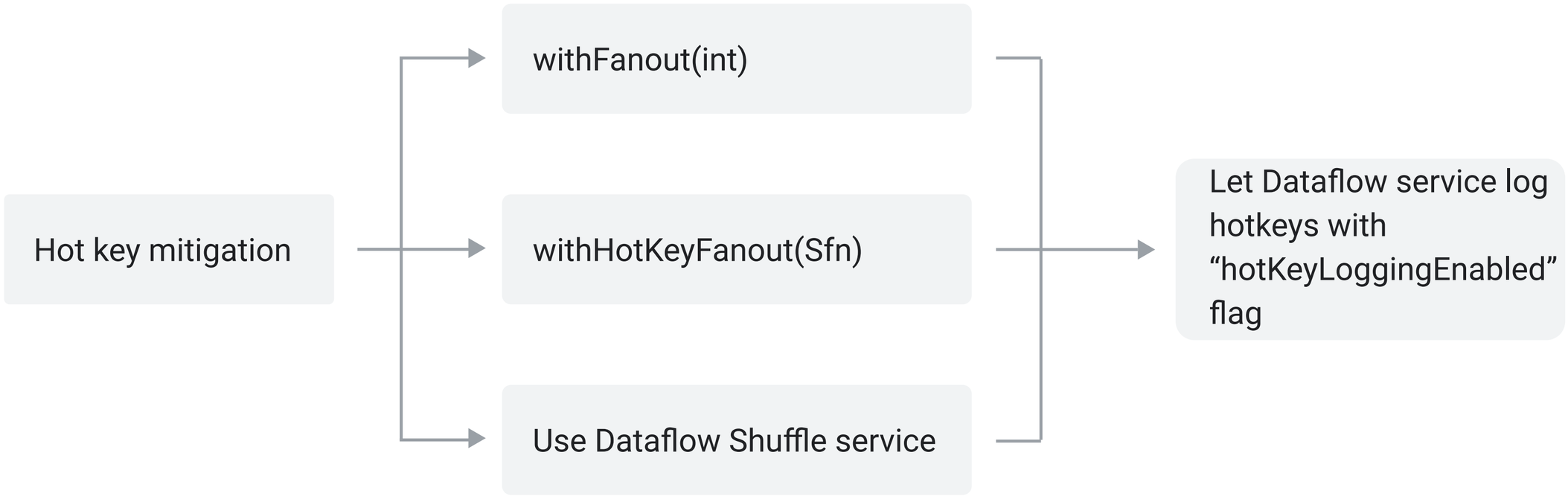
Design: Data skew

Log hot keys by setting hotKeyLoggingEnabled to true



Design: Data skew

Log hot keys by setting hotKeyLoggingEnabled to true



Design: Key space and parallelism

Low parallelism (too few keys)

1. Increase number of keys
   Example: If windows are distinct, use composite (window + key) keys.
2. If reading from files, prefer reading from splittable compression formats like Avro

parallelism = no. of keys

Design: Key space and parallelism

Low parallelism (too few keys)

1. Increase number of keys
   Example: If windows are distinct, use composite (window + key) keys.
2. If reading from files, prefer reading from splittable compression formats like Avro

Too-high parallelism (too many keys)

1. If key space is very large, consider using hashes separating keys out internally.
2. "Re-use" processing keys from the past that are not active

parallelism = no. of keys

Performance

Agenda

Pipeline design

Data shape

Sources, sinks & external systems

Shuffle & Streaming Engine

Source, Sinks & External Systems

TextIO + Compressed file

Source, sinks, and external systems

TextIO + Compressed file

Only one machine can read the compressed file.

Source, sinks, and external systems

TextIO + Compressed file

- Only one machine can read the compressed file.
- Fused stages will need to run on the same worker that read the data.

Source, Sinks & External System

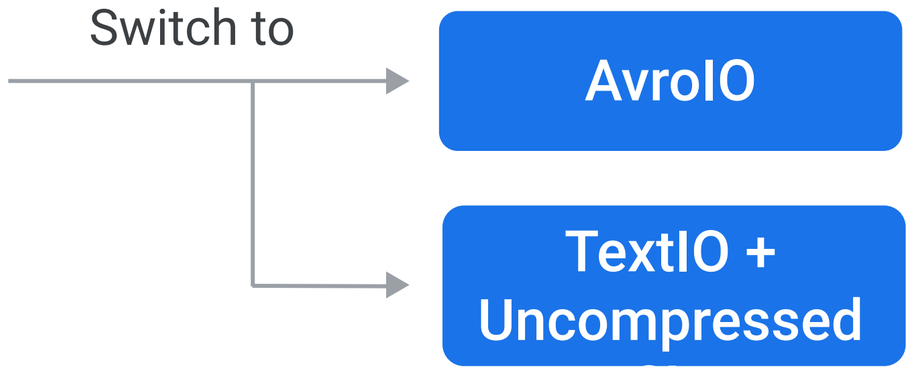
TextIO + Compressed file

- Only one machine can read the compressed file.
- Fused stages will need to run on the same worker that read the data.
- A single machine will need to push all the data from the file to all other machines.

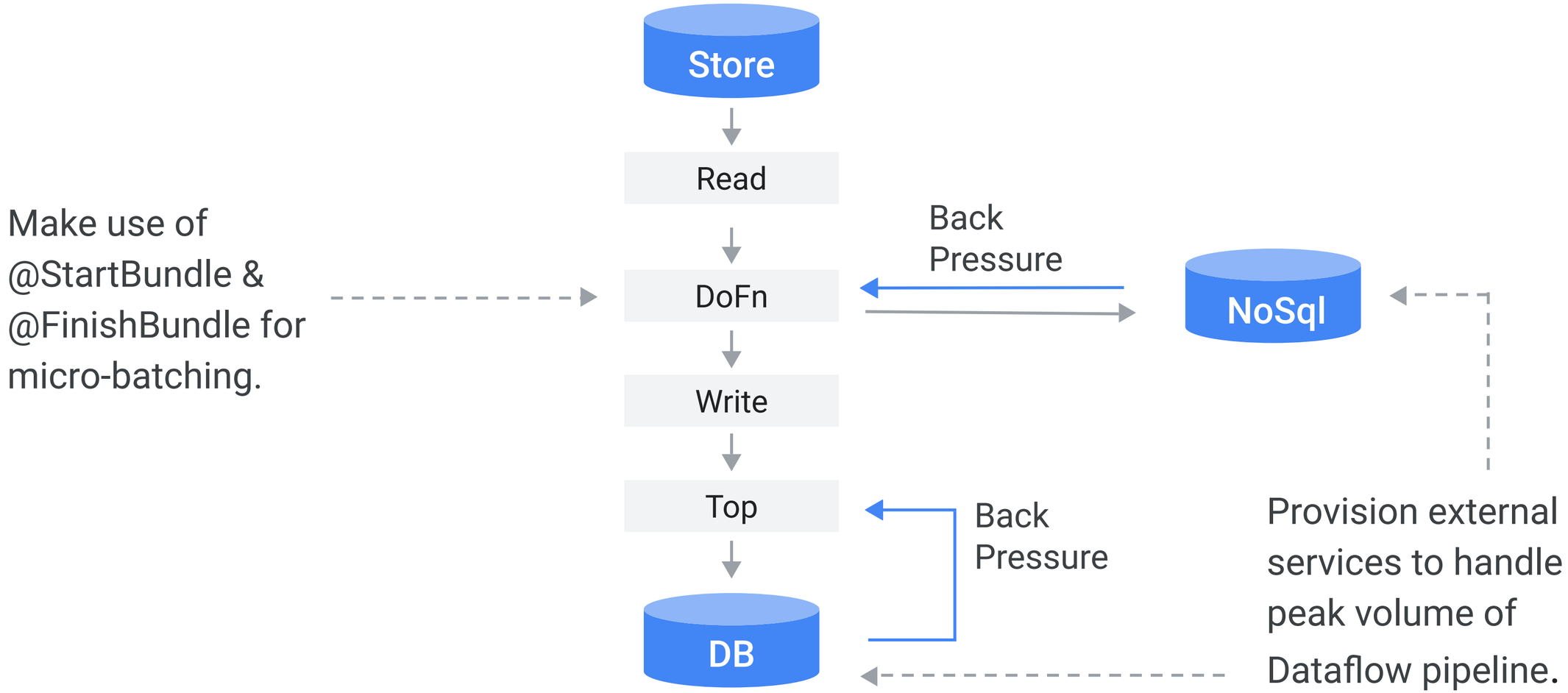
Source, Sinks & External System

TextIO + Compressed file

- Only one machine can read the compressed file.
- Fused stages will need to run on the same worker that read the data.
- A single machine will need to push all the data from the file to all other machines.

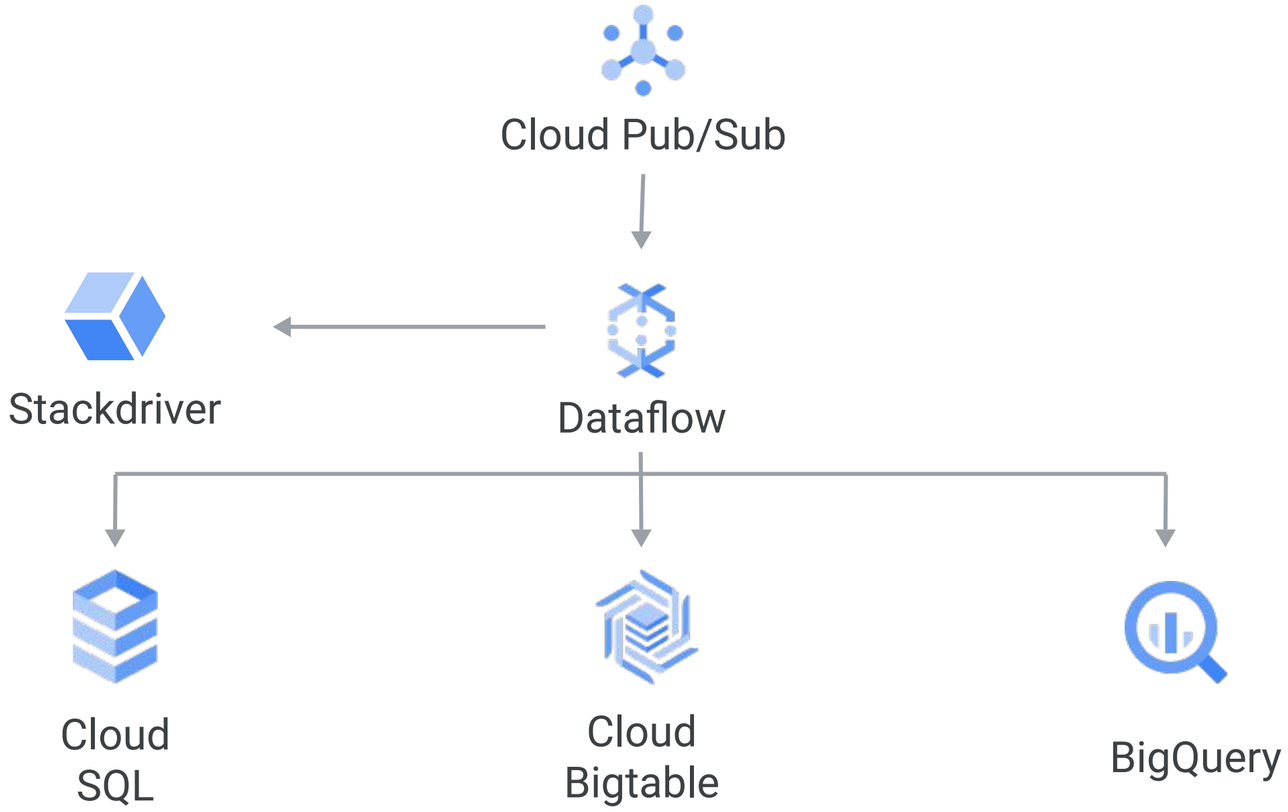


Source, Sinks & External Systems



Source, Sinks & External Systems

Colocation (i.e same region/zone) helps in performance



Performance

Agenda

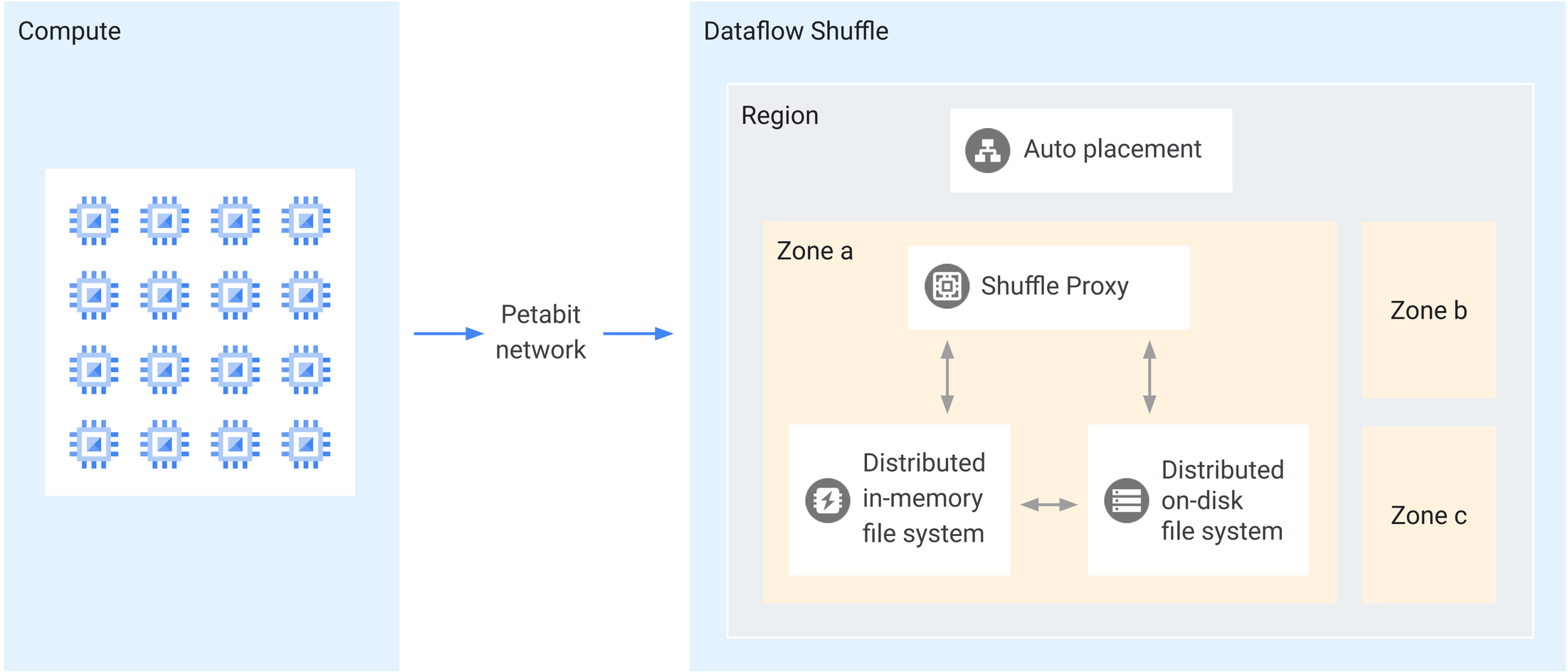
Pipeline design

Data shape

Sources, sinks & external systems

Shuffle & Streaming Engine

Dataflow Shuffle service



Benefits of Cloud Dataflow Shuffle

- Faster execution time of batch pipelines.

Benefits of Cloud Dataflow Shuffle

- Faster execution time of batch pipelines.
- A reduction in consumed CPU, memory, and persistent disk storage resources on the worker VMs.

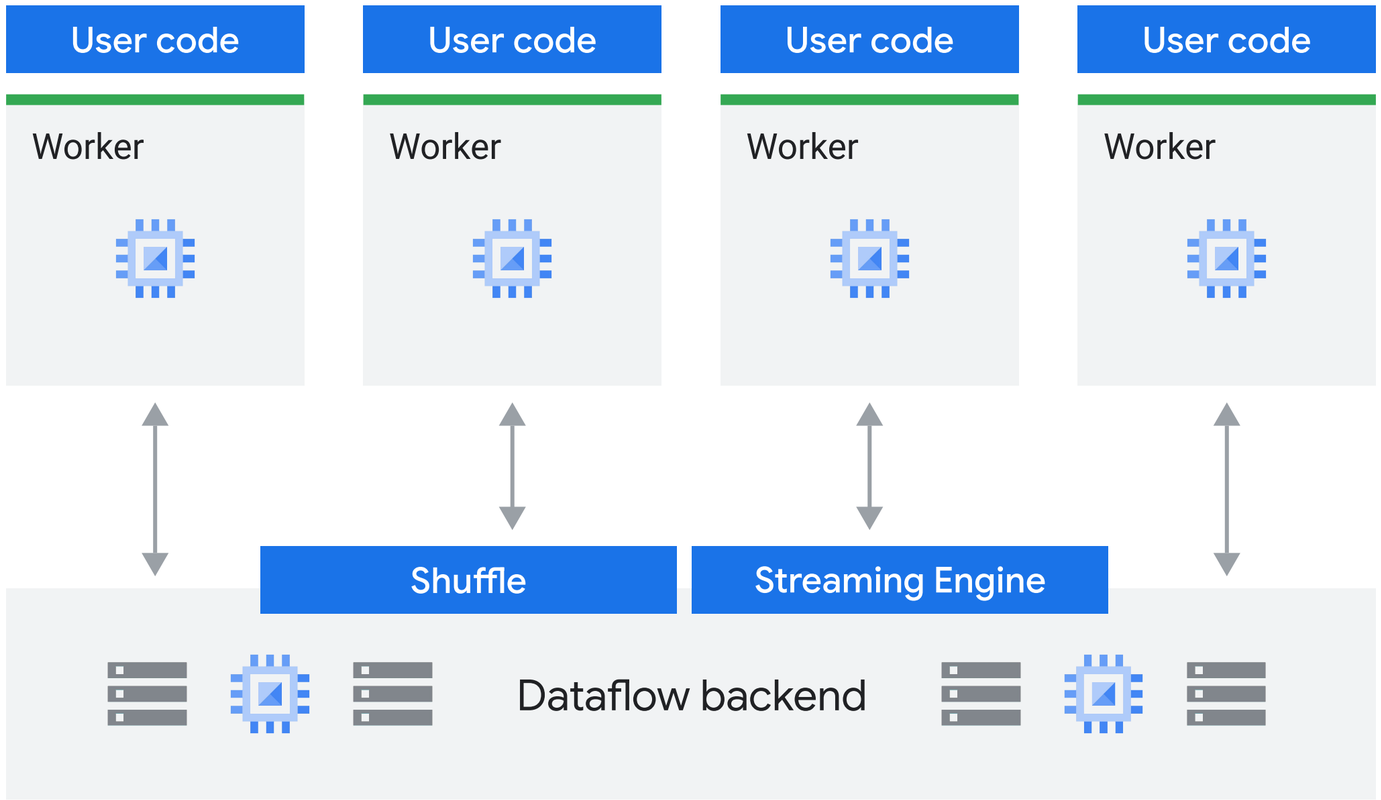
Benefits of Cloud Dataflow Shuffle

- Faster execution time of batch pipelines.
- A reduction in consumed CPU, memory, and persistent disk storage resources on the worker VMs.
- Better autoscaling.

Benefits of Cloud Dataflow Shuffle

- Faster execution time of batch pipelines.
- A reduction in consumed CPU, memory, and persistent disk storage resources on the worker VMs.
- Better autoscaling.
- Better fault tolerance.

Dataflow shuffle and streaming service



Summary