wasi-data

Support for embarrassingly parallel algorithms and distributed computation for data streams

Problem

- Input data is far beyond gigabyte-scale
- I/O-bound
- Distributed
- Must be resilient

API

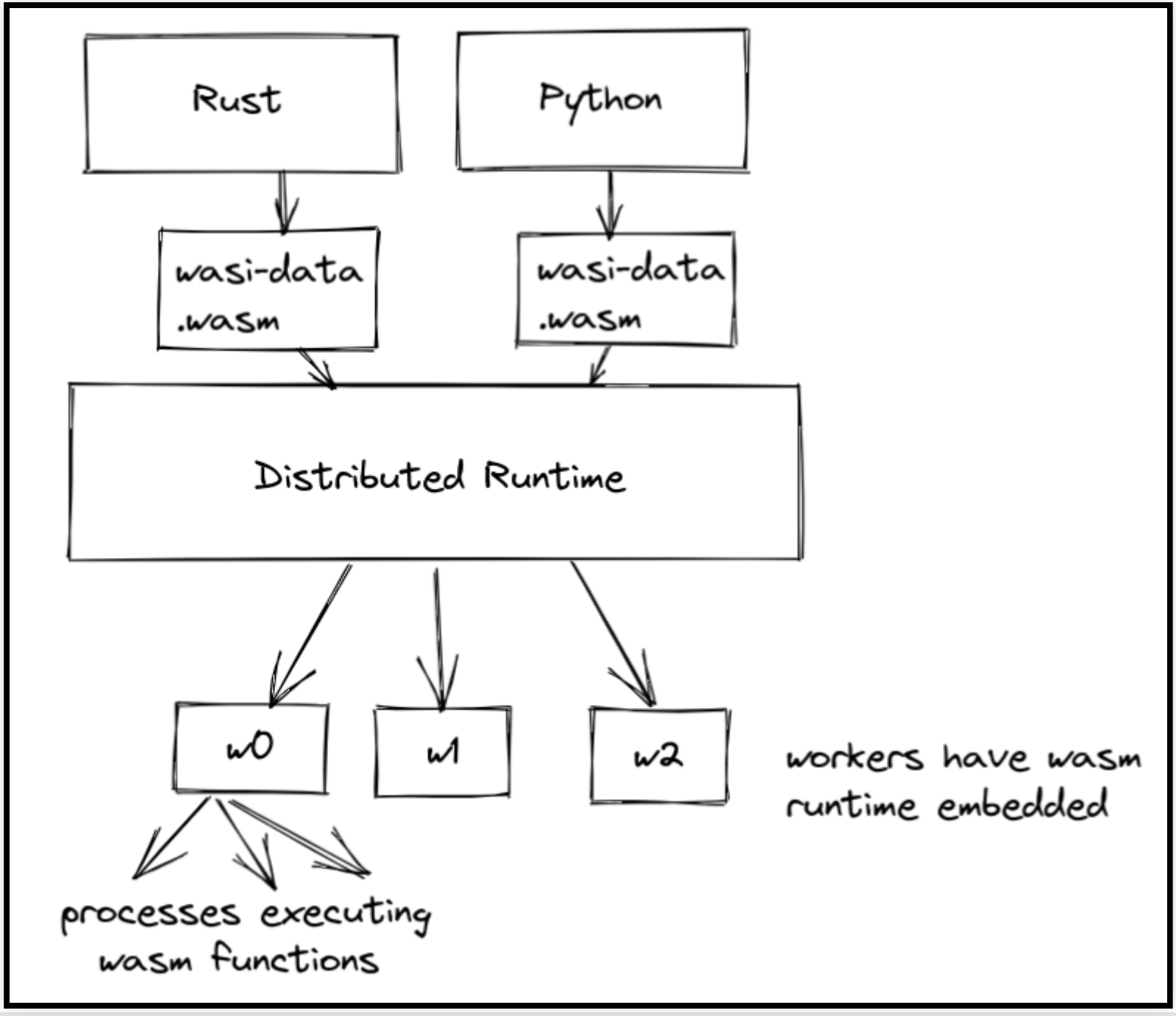
```
DataSet<Row<A,B,C…>>

map(func (Row<A,B,C>) Row<…>)

reduce(Row<out>, func(Row<out>, Row<orig>) Row<out>)
```

map-reduce

Specialization of split-apply-combine


Rust
Python
wasi-data
.wasm
wasi-data
.wasm
Distributed Runtime
w0
w1
w2
workers have wasm
runtime embedded
processes executing
wasm functions

Real world frameworks based on map-reduce

There are so many

To name a few

- Apache Hadoop
- Apache Spark
- Apache Flink
- Timely Dataflow
- Apache Beam
- …
- Google Cloud Dataflow
- IBM Streams
- Twister2
- …

Distributed map-reduce

Requires an implementation to connect processes performing map and reduce phases.

- Distributed file system
- Distributed database
- Streaming from mappers to reducers
- Sharding

Why WASM and WASI

- Portable
- Host and language-independent
- Reliability and Isolation
- Composable WASM modules
- Highly performant distributed computation (SIMD, hardware acceleration)

Example

```
createDataSet([
    Row(a=1, b=2., c='string1', d=date(2000, 1, 1)),
    Row(a=2, b=3., c='string2', d=date(2000, 2, 1)),
    Row(a=4, b=5., c='string3', d=date(2000, 3, 1))
])

DataSet<...> input = // [...]
DataSet<...> reduced = input
  .groupBy(/*define key here*/)
  .reduce(/*do something*/);
```