

MapReduce (Hadoop) with Povray

Jan Schlenker & Sebastian Sams

Agenda

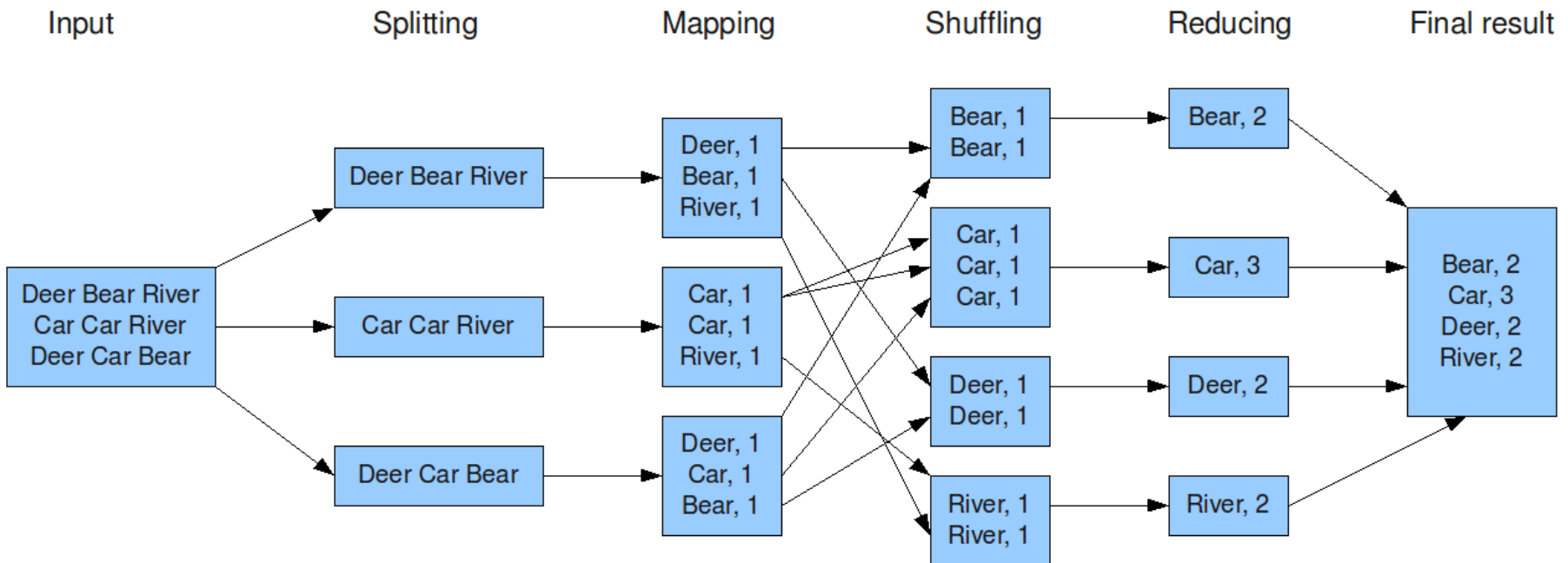


- MapReduce & Hadoop
- Amazon Elastic MapReduce
- Idea & Implementation
- Live Demo

MapReduce & Hadoop



The overall MapReduce word count process





```
public static class TokenizerMapper extends
    Mapper<Object, Text, Text, IntWritable>{

    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(Object key, Text value, Context context) throws
        IOException, InterruptedException {

        StringTokenizer itr = new StringTokenizer(value.toString());
        while (itr.hasMoreTokens()) {
            word.set(itr.nextToken());
            context.write(word, one);
        }
    }
}
```



```
public static class IntSumReducer extends
    Reducer<Text,IntWritable,Text,IntWritable> {

    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values, Context
        context) throws IOException, InterruptedException {

        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
        result.set(sum);
        context.write(key, result);
    }
}
```

Amazon Elastic MapReduce



Amazon Elastic
MapReduce

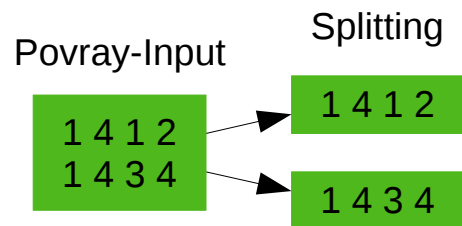
Idea & Implementation



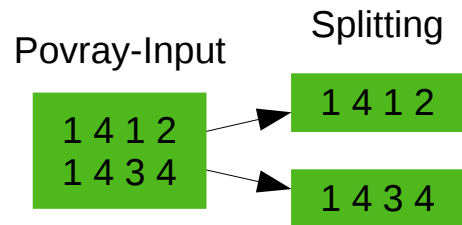
Povray-Input

```
1 4 1 2  
1 4 3 4
```

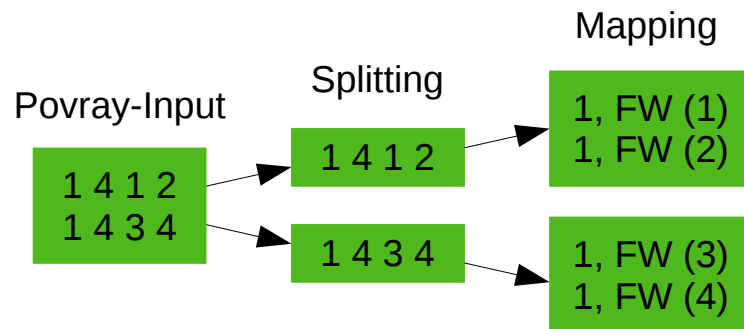
Idea & Implementation



Idea & Implementation

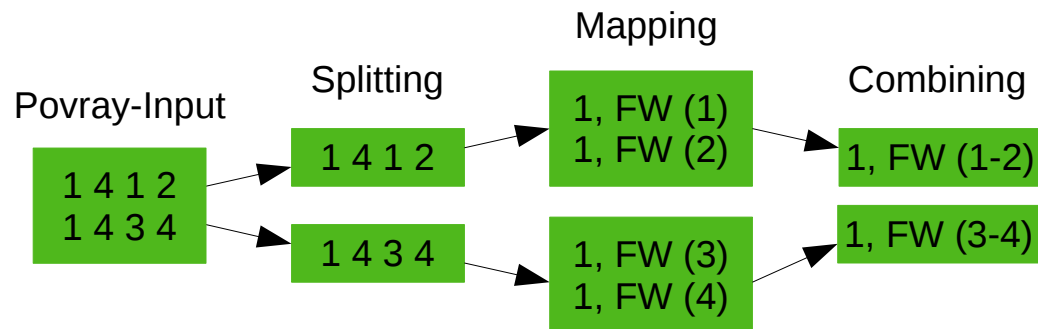


Idea & Implementation



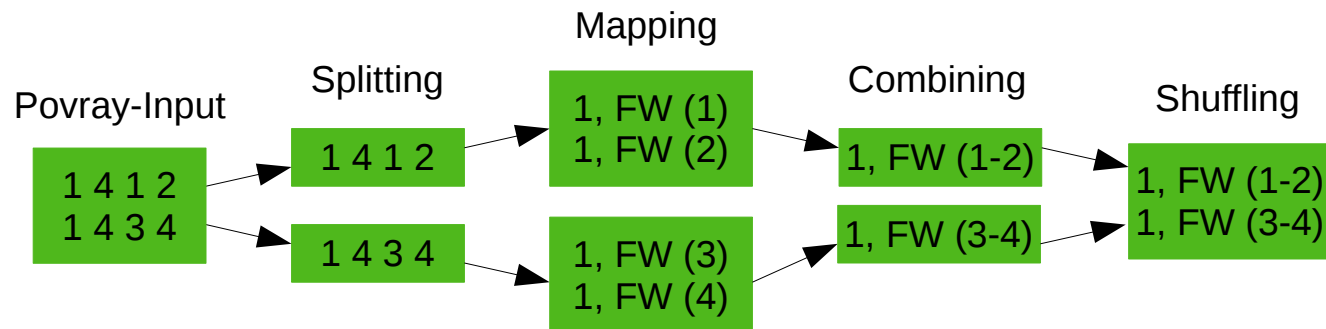
FW = FrameWritable

Idea & Implementation



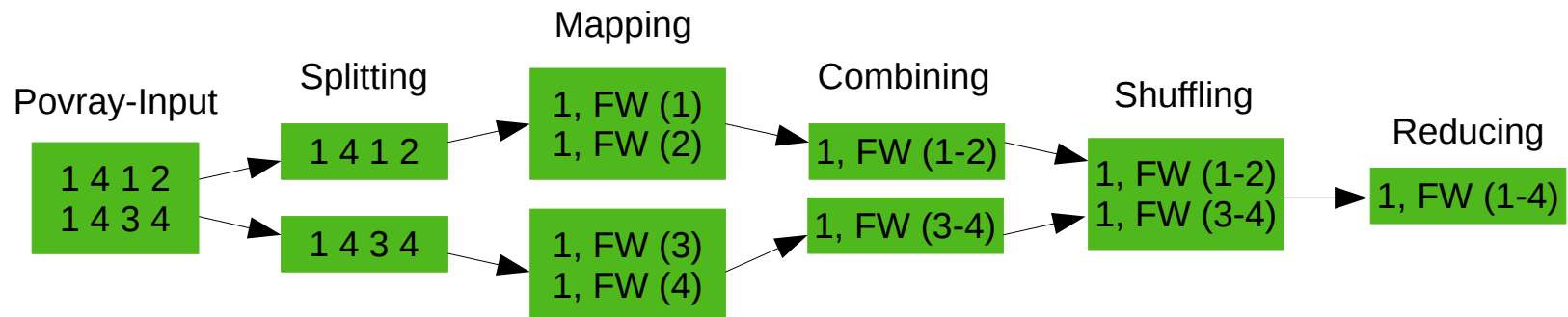
FW = FrameWritable

Idea & Implementation



FW = FrameWritable

Idea & Implementation



FW = FrameWritable



Live Demo