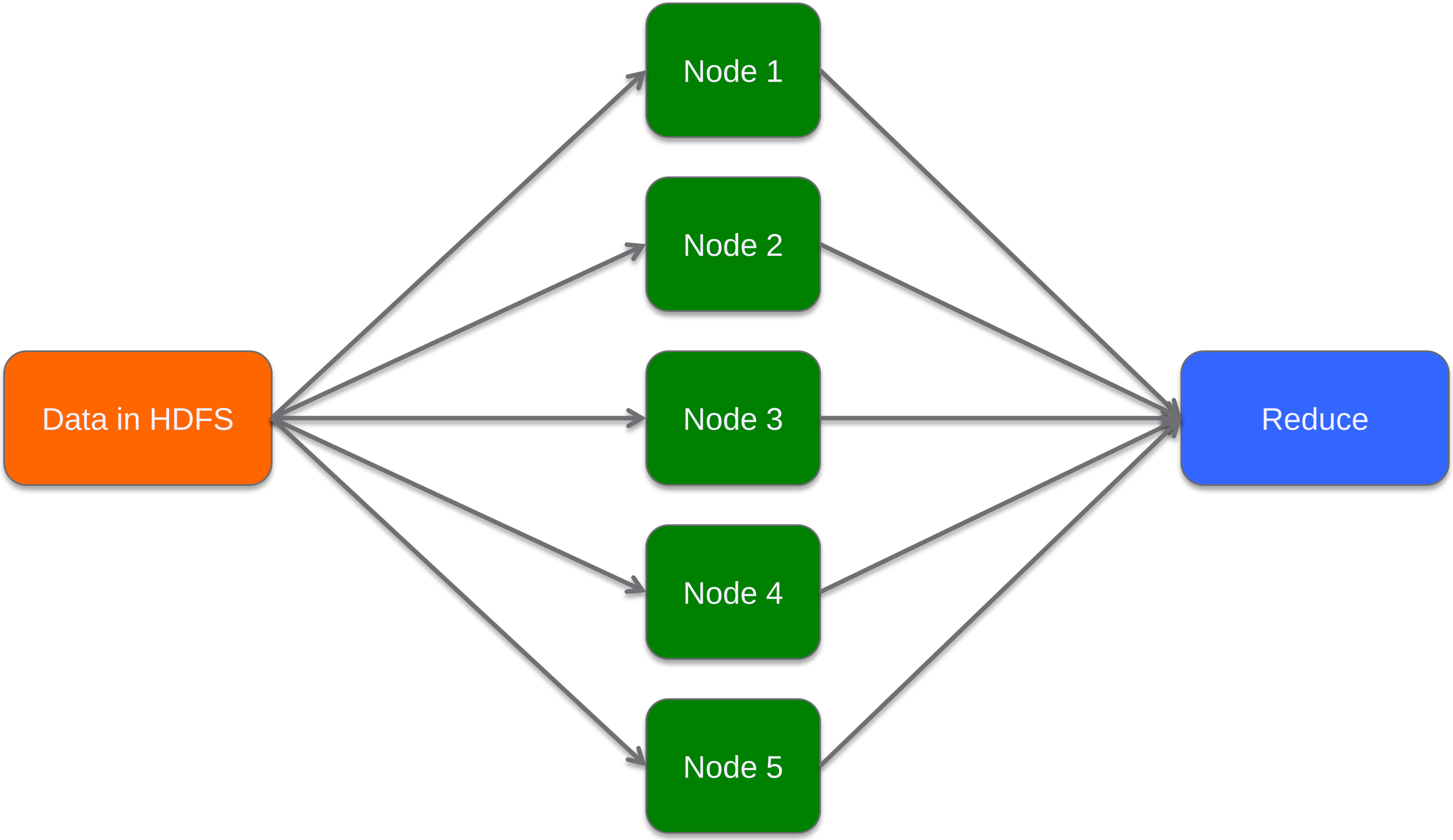
Big Data in Chapel: Working with HDFS

Tim Zakian

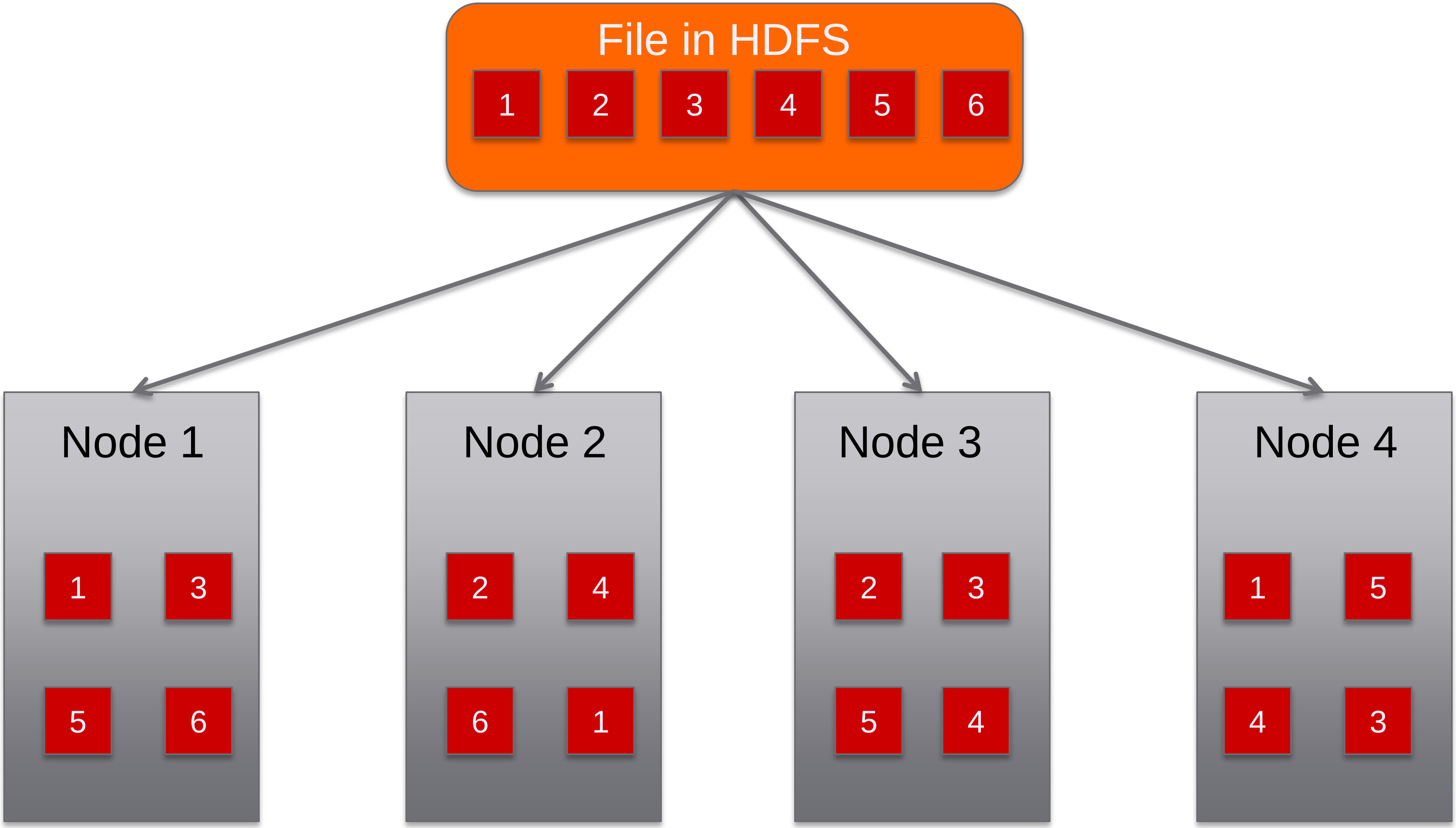
Michael Ferguson

Brad Chamberlain

What are HDFS and mapreduce?



What are HDFS and mapreduce?



What does I/O look like in Chapel?

```
var fl = open(“hello.txt”, iomode.cw),   // Open a file
    ch = fl.writer();                    // Create a writer channel
ch.writeln(“Hello World”);               // Write some data
ch.close();                              // Close the writer
fl.close();                              // Close the file
```

What does HDFS I/O look like in Chapel?

```
var hdfs = hdfs_chapel_connect(“default”, 0);   // Connect to HDFS
var fl = hdfs.hdfs_chapel_open(“hello.txt”, iomode.cw),    // Open a file
    ch = fl.writer();                    // Create a writer channel
ch.writeln(“Hello World”);               // Write some data
ch.close();                              // Close the writer
fl.close();                              // Close the file
hdfs.hdfs_chapel_disconnect();           // Disconnect from HDFS
```

Representing Data Records

formatted data file

```
beer/name: Sausa Weizen
beer/beerId: 47986
beer/brewerId: 10325
beer/ABV: 5.00
beer/style: Hefeweizen
review/appearance: 2.5
review/aroma: 2
review/palate: 1.5
review/taste: 1.5
review/overall: 1.5
review/time: 1234817823
review/profileName: stcules
review/text: ...
```

Chapel record type

```
record Beer {
  var name: string;
  var beerId: int;
  var brewerId: int;
  var ABV: real;
  var style: string;
  var appearance: real;
  var aroma: real;
  var palate: real;
  var taste: real;
  var overall: real;
  var time: int;
  var profileName: string;
  var text: string;
}
```

Applying a Reduction

```
const regEx = “beer/name: (.*)\\s*beer/beerID: (.*) …”;

const num_buckets = 5,
      max = 6,
      min = 0;
//
// Use a user-defined reduction in order to histogram
// the records into bins:
//
var c = myHisto reduce HDFSiter(“beers.txt”, Beer, regEx);
```

What about other file systems?

- Created an API so other distributed file systems can plug into Chapel easily.
- Could then do mapreduce with any file system you wanted.

Next Steps

- Evaluate Performance
- Gain User Experience
- Generalize HDFSiter()
- Support Lustre and Ceph

For more information, see:

$CHPL_HOME/doc/technotes/README.hdfs