Hadoop/Spark Config

CMPT 732, Fall 2017

Config Objects

There have been config objects around, but we haven't used them much. In MapReduce and Spark:

```
Configuration conf = this.getConf();
Job job = Job.getInstance(conf, "word count");
```

```
conf = SparkConf().setAppName('word count')
sc = SparkContext(conf=conf)
```

```
spark = SparkSession.builder.appName('example').getOrCreate()
```

The Command Line

... actually, that's not true. We have modified them with command line switches:

```
yarn jar wordcount.jar WordCount -D mapreduce.job.reduces=3 …
```

```
spark-submit --num-executors=4 --executor-memory=4g …
spark-submit --conf spark.executor.memory=4g …
```

Both of these have the effect of modifying the configuration object (and thus the behaviour of the jobs).

In Code

Config options can also be modified in code (where that makes sense, e.g. not the Spark driver memory):

```
Configuration conf = this.getConf();
conf.setInt("mapreduce.job.running.map.limit", 5);
conf.setInt("mapreduce.reduce.memory.mb", 4096);
Job job = Job.getInstance(conf, "word count");
```

```
conf = SparkConf().setAppName('word count') \
    .set('spark.shuffle.compress', False)
sc = SparkContext(conf=conf)
```

```
spark = SparkSession.builder \
    .config('spark.sql.shuffle.partitions', '100') \
    .getOrCreate()
```

Config Options

There are options to tune jobs in many, many ways:

- YARN configuration options
- HDFS configuration options
- MapReduce configuration options
- Spark Configuration options (plus more for Spark+YARN, SQL, etc.)

Filesystems

In both MapReduce and Spark, we have always accepted the default input filesystem: local files when running locally; HDFS on the cluster.

On our cluster, the default filesystem (Hadoop config `fs.defaultFS`) is `hdfs://nml-cloud-149.cs.sfu.ca:8020`, i.e. our HDFS server. This can be overridden with the path URL.

These are equivalent (for MapReduce & Spark):

```
TextInputFormat.addInputPath(job,
    new Path("/user/me"));
TextInputFormat.addInputPath(job,
    new Path("hdfs://nml-cloud-149.cs.sfu.ca:8020/user/me"));
```

```
sc.textFile('/user/me')
sc.textFile('hdfs://nml-cloud-149.cs.sfu.ca:8020/user/me')
```

There are other URL formats you can access (MapReduce or Spark, input or output):

```
sc.textFile('file:///mnt/share/inputs')
sc.textFile('s3a://s3key:s3secret@bucket/')
```