Spark Lesson 3

6 questions

1 point

1.
Check all true statements about the Directed Acyclic Graph Scheduler

- [x] A DAG is used to track dependencies of each partition of each RDD
- [ ] If a partition is lost, the DAG is traversed forward to check what other steps are affected
- [ ] The DAG is managed by the cluster manager
- [ ] Each transformation is executed as soon as it is called on a RDD

1 point

2.
Why is building a DAG necessary in Spark but not in MapReduce?

- ( ) In order to make a computation distributed at large scale
- ( ) For resiliency: it is necessary to make sure a partition can be recovered in case it is lost.
- (x) Because MapReduce always has the same type of workflow, Spark needs to accommodate diverse workflows.

1 point

3.
What are the differences between an action and a transformation? Mark all that apply

- [x] A transformation is lazy, an action instead executes immediately.
- [x] A transformation is from worker nodes to worker nodes, an action between worker nodes and the Driver (or a data source like HDFS)
- [ ] An action always triggers a shuffle.
- [ ] An action always writes the disk.

1 point

4.
Generally, which are good stages to mark a RDD for caching in memory?

- [ ] The first RDD, just after reading from disk, so we avoid reading from disk again.
- [ ] Every 2 or 3 transformations, to keep a recent backup.
- [x] At the start of an iterative algorithm.
- [x] After data cleaning, parsing and validation.

---

1 point

5.
What are good cases for using a broadcast variable? Mark all that apply

- [x] Copy a large configuration dictionary to all worker nodes
- [x] Copy a large lookup table to all worker nodes
- [x] Copy a small/medium sized RDD for a join
- [ ] Broadcast a Python module to all worker nodes

---

1 point

6.
We would like to count the number of invalid entries in this example dataset:

```
invalid = sc.accumulator(0)
d = sc.parallelize(["3", "23", "S", "99", "TT"]).foreach(count_invalid)
```

What would be a good implementation of the count_invalid function?

- ○

```
def count_invalid(element):
    try:
        int(element)
    except:
        invalid = invalid.add(1)
```

- ◉

```
def count_invalid(element):
    try:
        int(element)
    except:
        invalid.add(1)
```

```
def count_invalid(element):
    try:
        int(element)
    except:
        invalid = invalid + 1
```

```
def count_invalid(element):
    try:
        int(element)
    except:
        invalid.accumulate(1)
```

I understand that submitting work that isn't my own may result in permanent failure of this course or deactivation of my Coursera account. Learn more about Coursera's Honor Code

mingda zhang

Submit Quiz