2. experiment changing the number of executors

October 11, 2018

0.1 Changing the number of workers

The effect of changing the number of workers

- When you initialize SparkContext, you can specify the number of workers.
- Usually the recommendation is for one worker per core.
- But the number of workers can be smaller or larger than the number of cores

```
In [1]: #start the SparkContext
        import findspark
        findspark.init()

        from time import time
        from pyspark import SparkContext
        for j in range(1,10):
            sc = SparkContext(master="local[%d]"%(j))
            t0=time()
            for i in range(10):
                sc.parallelize([1,2]*1000000).reduce(lambda x,y:x+y)
            print("%2d executors, time=%4.3f"%(j,time()-t0))
            sc.stop()
```

```
 1 executors, time=11.636
 2 executors, time=9.269
 3 executors, time=9.099
 4 executors, time=8.301
 5 executors, time=8.644
 6 executors, time=9.177
 7 executors, time=8.734
 8 executors, time=8.360
 9 executors, time=8.361
```

0.2 Summary

- This machine has 4 cores
- Increasing the number of executors from 1 to 3 speeds up the computation
- From 3 and up you have fluctuations in performance.
- More than one worker per core is usually unhelpful