大规模分布式系统第二次实验报告
员司雨 17307110448

PartI：统计其中各类文件的数量

一. 实验介绍

mrjob 是编写能够在 hadoop 上运行的 python 程序最简单的途径。如果使用 mrjob，可以在本地测试的代码，甚至不需要安装 hadoop 或者在选择的集群上运行。

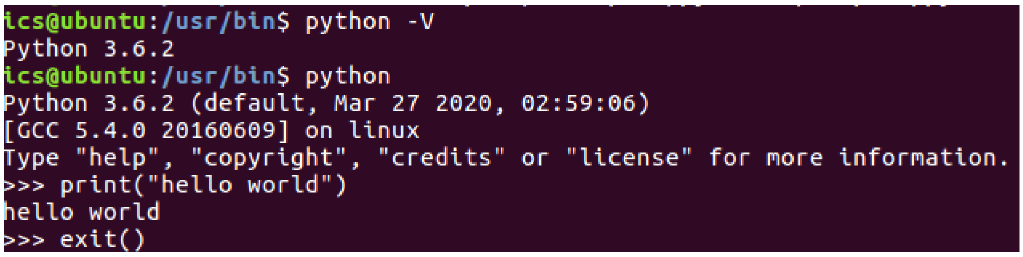
另外，mrjob 可以和亚马逊的 EMR（Elastic MapReduce）服务无缝集成。只要设置完毕，就可以运行在 EMR 上，像在自己的笔记本上运行一样简单。

二. 实验环境

1. Ubuntu18.04
2. jdk 1.8.0_131
3. hadoop 2.7.7
4. Python3.6.2
5. mrjob 包

三. 实验过程

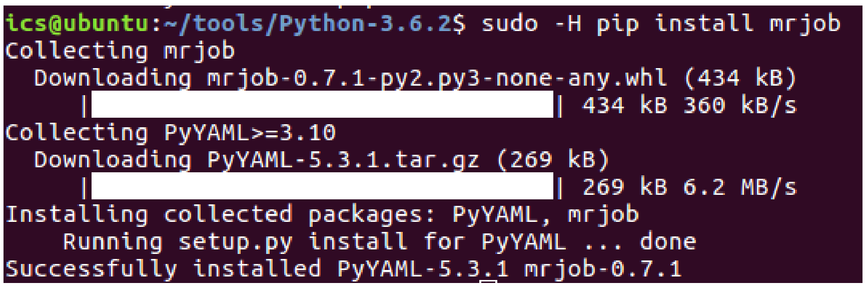
1. Python 版本查看



```
ics@ubuntu:/usr/bin$ python -V
Python 3.6.2
ics@ubuntu:/usr/bin$ python
Python 3.6.2 (default, Mar 27 2020, 02:59:06)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print("hello world")
hello world
>>> exit()
```

Python 版本为 3.6.2

2. mrjob 包安装



```
ics@ubuntu:~/tools/Python-3.6.2$ sudo -H pip install mrjob
Collecting mrjob
  Downloading mrjob-0.7.1-py2.py3-none-any.whl (434 kB)
     |████████████████████████████████| 434 kB 360 kB/s
Collecting PyYAML>=3.10
  Downloading PyYAML-5.3.1.tar.gz (269 kB)
     |████████████████████████████████| 269 kB 6.2 MB/s
Installing collected packages: PyYAML, mrjob
    Running setup.py install for PyYAML ... done
Successfully installed PyYAML-5.3.1 mrjob-0.7.1
```

使用 pip 安装 mrjob 包

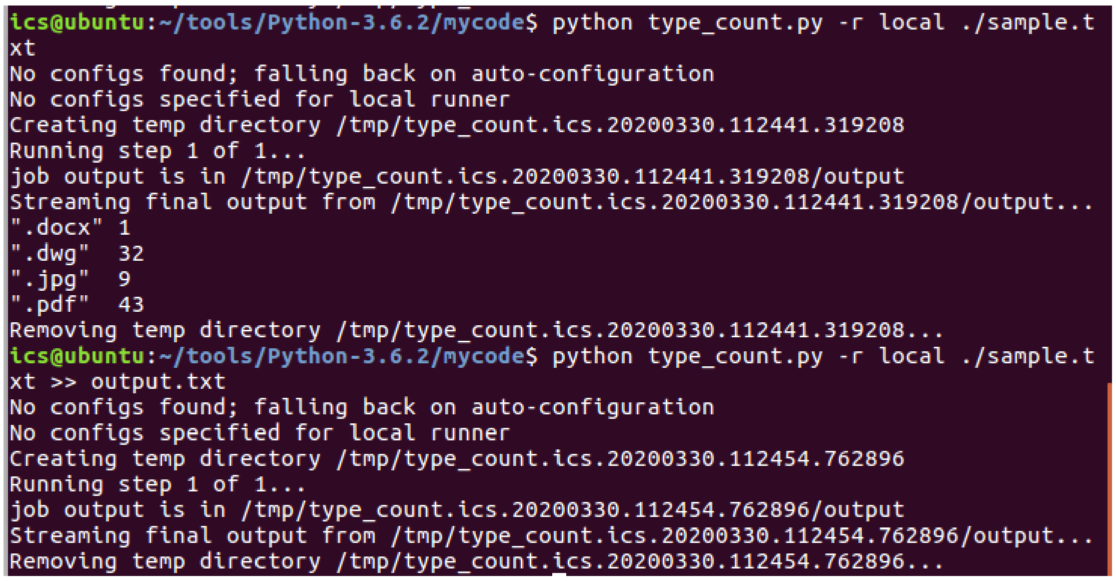
3. Python 代码编写

```python
from mrjob.job import MRJob
```

```
import os
class MRFILE_TYPE_Counter(MRJob):

  def mapper(self, key, line):
    temp = line.split(' ')
    F = temp[-1]
    f = os.path.splitext(F)
    filename,ty = f
    yield ty, 1

  def reducer(self, word, occurrences):
    yield word, sum(occurrences)

if __name__=='__main__':
  MRFILE_TYPE_Counter.run()
```

参见：type_count.py

4. 实验结果



```
ics@ubuntu:~/tools/Python-3.6.2/mycode$ python type_count.py -r local ./sample.t
xt
No configs found; falling back on auto-configuration
No configs specified for local runner
Creating temp directory /tmp/type_count.ics.20200330.112441.319208
Running step 1 of 1...
job output is in /tmp/type_count.ics.20200330.112441.319208/output
Streaming final output from /tmp/type_count.ics.20200330.112441.319208/output...
".docx" 1
".dwg"  32
".jpg"  9
".pdf"  43
Removing temp directory /tmp/type_count.ics.20200330.112441.319208...
ics@ubuntu:~/tools/Python-3.6.2/mycode$ python type_count.py -r local ./sample.t
xt >> output.txt
No configs found; falling back on auto-configuration
No configs specified for local runner
Creating temp directory /tmp/type_count.ics.20200330.112454.762896
Running step 1 of 1...
job output is in /tmp/type_count.ics.20200330.112454.762896/output
Streaming final output from /tmp/type_count.ics.20200330.112454.762896/output...
Removing temp directory /tmp/type_count.ics.20200330.112454.762896...
```

实验成功结果，具体参加：output.txt

PartII：按文件的字节数大小降序排序输出文件名

一. 实验介绍

mrjob 是编写能够在 hadoop 上运行的 python 程序最简单的途径。如果使用 mrjob，可以在本地测试的代码，甚至不需要安装 hadoop 或者在选择的集群上运行。

另外，mrjob 可以和亚马逊的 EMR（Elastic MapReduce）服务无缝集成。只要设置完毕，就可以运行在 EMR 上，像在自己的笔记本上运行一样简单。

二. 实验环境

6. Ubuntu18.04
7. jdk 1.8.0_131
8. hadoop 2.7.7
9. Python3.6.2
10. mrjob 包

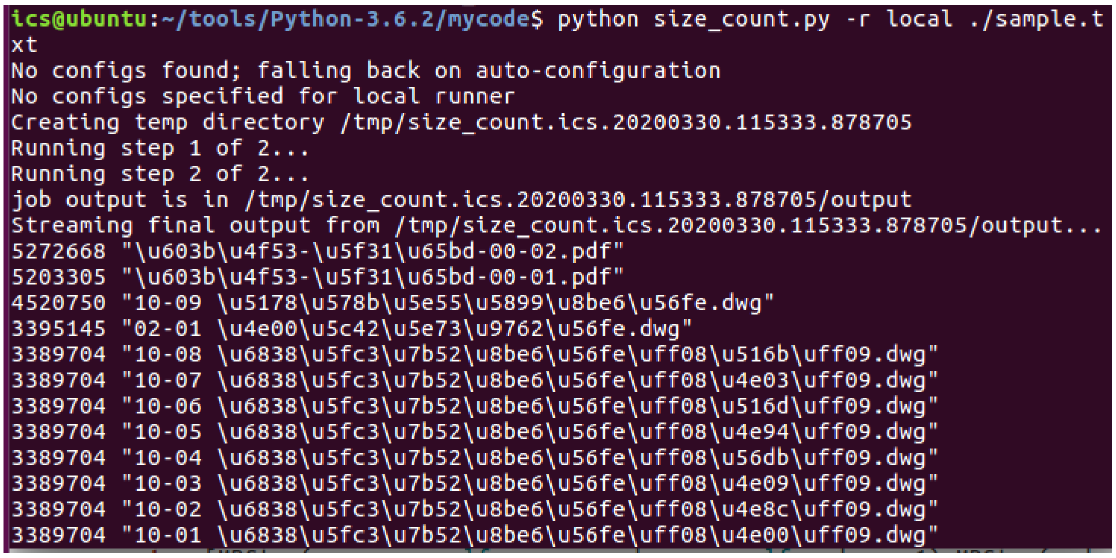
三. 实验过程

1. Python 代码编写

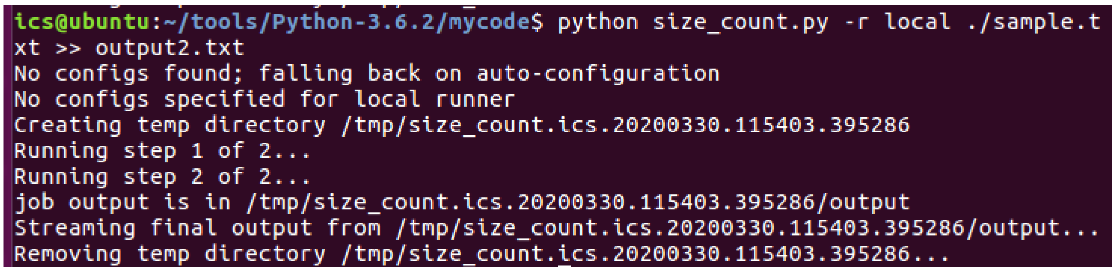
```python
from mrjob.job import MRJob
from mrjob.step import MRStep
import heapq
import os
class MRFILE_SIZE_Counter(MRJob):

  def mapper(self, key, line):
    temp = line.split()
    Size = temp[2]
    if ‘,’ in temp[-2]:
      F = temp[-1]
    else:
      F = temp[-2]+' '+temp[-1]
    yield (int(Size.replace(',','')),F),1

  def reducer_1(self, key, value):
    yield None, key

  def reducer_2(self, _, value):
    for s, f in heapq.nlargest(85, value):
      yield s,f
  def steps(self):
    return[MRStep(mapper = self.mapper,reducer = self.reducer_1),MRStep(reducer = self.reducer_2)]

if __name__=='__main__':
  MRFILE_SIZE_Counter.run()
```

参见：type_count.py

2. 实验结果



```
ics@ubuntu:~/tools/Python-3.6.2/mycode$ python size_count.py -r local ./sample.t
xt
No configs found; falling back on auto-configuration
No configs specified for local runner
Creating temp directory /tmp/size_count.ics.20200330.115333.878705
Running step 1 of 2...
Running step 2 of 2...
job output is in /tmp/size_count.ics.20200330.115333.878705/output
Streaming final output from /tmp/size_count.ics.20200330.115333.878705/output...
5272668 "\u603b\u4f53-\u5f31\u65bd-00-02.pdf"
5203305 "\u603b\u4f53-\u5f31\u65bd-00-01.pdf"
4520750 "10-09 \u5178\u578b\u5e55\u5899\u8be6\u56fe.dwg"
3395145 "02-01 \u4e00\u5c42\u5e73\u9762\u56fe.dwg"
3389704 "10-08 \u6838\u5fc3\u7b52\u8be6\u56fe\uff08\u516b\uff09.dwg"
3389704 "10-07 \u6838\u5fc3\u7b52\u8be6\u56fe\uff08\u4e03\uff09.dwg"
3389704 "10-06 \u6838\u5fc3\u7b52\u8be6\u56fe\uff08\u516d\uff09.dwg"
3389704 "10-05 \u6838\u5fc3\u7b52\u8be6\u56fe\uff08\u4e94\uff09.dwg"
3389704 "10-04 \u6838\u5fc3\u7b52\u8be6\u56fe\uff08\u56db\uff09.dwg"
3389704 "10-03 \u6838\u5fc3\u7b52\u8be6\u56fe\uff08\u4e09\uff09.dwg"
3389704 "10-02 \u6838\u5fc3\u7b52\u8be6\u56fe\uff08\u4e8c\uff09.dwg"
3389704 "10-01 \u6838\u5fc3\u7b52\u8be6\u56fe\uff08\u4e00\uff09.dwg"
```



```
ics@ubuntu:~/tools/Python-3.6.2/mycode$ python size_count.py -r local ./sample.t
xt >> output2.txt
No configs found; falling back on auto-configuration
No configs specified for local runner
Creating temp directory /tmp/size_count.ics.20200330.115403.395286
Running step 1 of 2...
Running step 2 of 2...
job output is in /tmp/size_count.ics.20200330.115403.395286/output
Streaming final output from /tmp/size_count.ics.20200330.115403.395286/output...
Removing temp directory /tmp/size_count.ics.20200330.115403.395286...
```

实验成功结果，具体参加：output2.txt

3. 细节介绍

(1) 由于 hadoop 默认是对 key 做升序排序输出，而我们的要求是倒序，所以引入包 heapq.

(2) 由于输出是 unicode 编码，故而需要转化为中文。本地在线转换代码如下

```python
temp1 = []

address2 = "D:\output2.txt"
with open(address2, 'r') as f1:
  for line in f1:
    line = line.strip('\n')
    temp1.append(line)

ls0 = []
for ele in temp1:
  a = ele.split('\t')
  a[1] = a[1].encode('utf-8').decode('unicode_escape')
  ls0.append(a)

f = open("D:/output2_rewrite.txt", "w")
for element in ls0:
  f.write(str(element[0])+'\t'+element[1]+'\n')
```

```
f.close()
```

代码及输出结果参加 file_rewrite.py 和 output2_rewrite.txt。