

استاد: محمدعلی نعمت بخش دستیاران: فاطمه ابراهیمی، پریسا لطیفی، امیر سرتییی

**تمرین سوم:** زمان اجرا

درس: تحلیل سیستم دادههای حجیم

نام و نامخانوادگی: فاطمه مومنی

آدرس گیت: https://github.com/FatemehMomeni/BD3.git

- لطفا پاسخ تمارین حتما در سامانهی کوئرا ارسال شود.
- لطفا پاسخهای خود را در خود سند سوال نوشته و در قالب یک فایل PDF ارسال کنید.
  - نام سند ارسالی {Name Family}-{student number}}
    - تمامی فایلهای مورد نیاز این تمرین در این لینک قابل دسترس است.
      - خروجی از هر مرحلهی تمرین را در سند خود بارگذاری کنید.

در این تمرین هدف ما مقایسه زمان اجرا در هدوپ و اسپارک است.

برای این منظور ۴ فایل داده متنی با حجمهای ۱، ۵، ۱۰ و ۱۲ گیگابایتی در اختیار شما قرار گرفته است که انتظار میرود با نوشتن برنامه ی شمارش کلمات عملیات نگاشت-کاهش را برای داده ها بر روی هدوپ و اسپارک انجام دهید. نتایج را گزارش و مقایسه ای بین آنها انجام دهید.

آدرس فايلها:

/user/ebrahimi/hw3-data

نمونهی دستور اسپارک را با client mode هم امتحان کرده و تفاوت حالت cluster و rull را بیان کنید.

### عملیات نگاشت-کاهش بر روی هدوپ

reducer.py و mapper.py و فایل ۱ و شکل ۲ مشاهده می شود. این برنامه در قالب دو فایل mapper.py و mapper.py نو شته شده است. درفایل mapper.py پس از دریافت ورودی، کلمات هر خطر برا ساس فا صله بین آنها جدا شده و در خروجی به صورت زوج (word,1) چاپ می شود. فایل reducer.py با دریافت خروجی مرحله قبل، تعداد تکرار هر کلمه را شمارش کرده و در خروجی به صورت زوج (word,count) چاپ می کند.

شکل ۱: برنامه شمارش کلمات، عملیات نگاشت

شكل ٢: برنامه شمارش كلمات، عمليات كاهش

به منظور اجرای این برنامه، از دستور نشان داده شده در شکل ۳، استفاده شده است. در این دستور، پس از آدرس پکیج، فایلهای ورودی، فایلهای mapper و reducer و مسیر ذخیره خروجی، داده می شود.

```
fatemeh_momeni@MasterPC:~$ hadoop jar /home/hduser/hadoop/hadoop/share/hadoop/to ols/lib/hadoop-streaming-2.10.1.jar -files ./mapper.py,./reducer.py -mapper mapp er.py -reducer reducer.py -input /user/ebrahimi/hw3-data/filelG.txt -output /use r/fatemeh_momeni/hadoop_outputlG
```

شکل ۳: دستور اجرای برنامه عملیات نگاشت-کاهش بر روی هدوپ

متاسفانه این دستور با موفقیت اجرا نشد. شکل ۴ بخش انتهایی خطای اجرای این دستور را نشان میدهد.

شکل ۴: بخشی از خطای اجرای ناموفق دستور راهاندازی هدوپ

## عملیات نگاشت–کاهش بر روی اسیارک در cluster mode

برنامه شـمارش کلمات در شـکل ۵ مشـاهده میشـود. در این برنامه، پس از پیکربندی spark context، فایل ورودی از مسـیر داده شـده با تابع ()textFile خوانده شـده و توسـط تابع ()flatMap نگاشـتی روی تابع درون آن که خطوط فایل ورودی را براسـاس فاصـله بین کلمات جدا می کند، اعمال می شـود. سـپس، تابع ()map نگاشـت دیگری روی خروجی بهد ست آمده از این مرحله اعمال می کند و زوج (word,1) که همان خروجی عملیات نگاشت است را بهد ست می دهد. در آخر، با تابع (reduceByKey) عملیات کاهش براسـاس کلید (کلمات)، انجام شـده و خروجی نهایی توسـط تابع saveAsTextFile()

شکل ۵: برنامه شمارش کلمات با عملیاتهای اسیارک

برای اجرای برنامه مربوط به عملیات نگاشت-کاهش بر روی اسپارک در حالت کلاستر، دستور نشان داده شده در شکل ۶ اجرا شدهاست.

```
fatemeh_momeni@MasterPC:~$ spark-submit --class org.apache.spark.examples.SparkP
i_--master yarn --deploy-mode cluster /home/fatemeh_momeni/spark_mr.py
```

#### شکل ۶: دستور اجرای عملیات نگاشت-کاهش روی اسپارک در حالت cluster

متاسفانه این دستور پس از مدت زمان زیادی با خطای نشان دادهشده در شکل ۷ متوقف شد.

شکل ۷: خطای راهاندازی اسیارک در حالت cluster

## عملیات نگاشت-کاهش بر روی اسیارک در client mode

در این حالت نیز از برنامه شمارش کلمات نشان داده شده در شکل  $\alpha$  استفاده شده است. برای اجرای برنامه مربوط به عملیات نگاشت-کاهش بر روی اسپارک در حالت کلاینت، دستور نشان داده شده در شکل  $\alpha$  اجرا شده است.

```
fatemeh_momeni@MasterPC:~$ spark-submit --class org.apache.spark.examples.SparkP
i --master yarn --deploy-mode client /home/fatemeh_momeni/spark_mr.py
```

#### شکل ۸: دستور اجرای عملیات نگاشت-کاهش روی اسیارک در حالت client

پس از اجرای دستور فوق، همان گونه که در شکل ۹ مشاهده می شود، به مسیری که فایل خروجی در آن ذخیره شده است رفته و با دستور h محتویات آن را چاپ می کنیم.

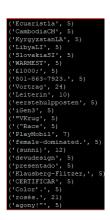
```
fatemeh_momeni@MasterPC:~$ 1s
hw3-data mapper.py.save reducer.py spark_output
mapper.py map_reduce spark_mr.py user_shell_exporter.sh
fatemeh_momeni@MasterPC:~$ cd spark_output/
fatemeh_momeni@MasterPC:~/spark_output$ 1s
part-00000 part-00002 part-00004 part-00006 _SUCCESS
part-00001 part-00003 part-00005 part-00007
fatemeh_momeni@MasterPC:~/spark_output$ h -cat /user/fatemeh_momeni/spark_output
/part-00000
```

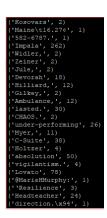
شكل ٩: دستور چاپ محتويات فايل خروجي

بخشی از خروجی مربوط به هر یک از فایلها در شکل ۱۰ ،شکل ۱۱ ،شکل ۱۲ وشکل ۱۳ آورده شده است.

```
('Sesame-Soy', 2)
('Duch,', 6)
('Duch,', 6)
('Dandful', 2)
('aight...', 2)
('squishes'', 2)
('Waz', 51)
('MUTOBLography', 2)
('Martinsburg,', 2)
('mso-hansi-font-family:Calibri;', 22)
('seemsThe', 2)
('duck', 52)
('duck', 52)
('duck', 52)
('Stumpt', 2)
('Chuffest', 4)
("Hey!', 2)
('Invaluable:', 15)
('chorale's', 2)
('Oboe", 2)
('Tahoe-LAFS,', 2)
('psrfect'Cheap', 2)
('mste', 5)
```

```
('flickering).', 3)
('Nope:', 15)
('192.168.85.16,', 3)
('Triggerx', 3)
('Allpro', 7)
('Lauras', 3)
('Paradaens', 3)
('Dossogne', 3)
('baki', 3)
('bhoomi', 7)
('Mantralay's', 3)
('padyforessay.Web', 3)
('padi?', 3)
('RatEN3', 3)
('Padi?', 44)
('=23Kg.', 3)
('wonder!!!!!', 3)
('slinhoven', 6)
('Carmero,', 6)
('Carmero,', 6)
('L',700-2,100', 3)
('P-Diady', 3)
('Paris".', 3)
```





شکل ۱۳: شی از خروجی مربوط به فایل file12G.txt

شکل ۱۲: خشی از خروجی مربوط به فایل file5G.txt

شکل ۱۱: بخشی از خروجی مربوط به فایل file10G.txt

شکل ۱۰: بخشی از خروجی مربوط به فایل file1G.txt

# تفاوت حالت client و cluster در اسپارک

در حالت spark driver ،client روی ماشینی که کار از آن ارسال شدهاست اجرا می شود. به این معنی که درایور به صورت محلی و با وارد کردن دستور spark-submit اجرا شده و وظایف روی ماشینهای کارگر درون cluster، اجرا می شوند. هم چنین golهای درایور، در ماشین محلی قابل دسترسی می باشند. در این حالت، اگر یک جلسه کاربر که در حال اجرای spark-submit است خاتمه یابد، برنامه نیز با شکست خاتمه می یابد. این حالت برای تست و debugging به کار می رود و برای کارهای تولیدی استفاده نمی شود. با توجه به این که در این حالت داده ها باید بین ماشین درایور (ماشین محلی) و ماشین های کارگر جابجا شوند، بسته به تاخیر شبکه، ممکن است عملکرد برنامه کاهش یابد و زمان اجرا طولانی تر شود.

در حالت cluster، برنامه درایور می تواند روی هر یک از ماشینهای درون cluster اسپارک اجرا شود. درواقع، روی هر یک از گرههای موجود و در دسترس cluster می تواند راهاندازی شود. در این حالت، خاتمه یافتن یک جلسه باعث خاتمه یافتن کل برنامه نمی شود و برنامه می تواند بر روی cluster در حال اجرا بماند. با توجه به این که در این حالت درایور و ماشینهای کار گر درون cluster هستند، سرباره جابجایی داده بین cluster و ماشینی که کار از آن ارسال شده است کاهش می یابد.