1)

Hadoop و spark دو فریمورک متن باز هستند که برای پردازش کلان داده ها (big data) به وجود آمده اند. منظور از big data داده هایی هستند که حجم و نوع داده در آن ها بسیار زیاد است. از Hadoop برای نگهداری و مدیریت و پردازش big data با استفاده از مدل برنامه نویسی نگاشت کاهش (map reduce) استفاده می شود.

:HDFS

Hadoop برای ذخیره داده ها از سیستم فایلی به نام HDFS استفاده می کند. در HDFS برای ذخیره داده از معماری master-slave (ارباب-برده) استفاده می شود.یک Name Node و چند Data Node داریم. داده ی ورودی به تعدادی بلاک تقسیم و این بلاک ها میان data node ها برای ذخیره سازی توزیع می شود. همچنین هر بلاک در چند data node تکرار می شود که حاصل این کار خطا پذیری کم در Hadoop است. برای مثال زمانی که یک node از کار افتاد می توان بلاک درخواستی را از node دیگر گرفت. Name node که همان node ارباب ما می شود می داند که دقیقا کدام data node حاوی کدام بلاک ها است و در کجای کلاستر قرار گرفته است. همچنین مدیریت دسترسی برای خواندن نوشتن حذف و ... در data node ها توسط name node انجام می شود.

:Map Reduce

از map reduce برای پردازش اطلاعات ذخیره شده (در مرحله قبل با HDFS) با استفاده از node های داخل cluster به صورت موازی استفاده می شود. از مزایای این روش می توان به ساده بودن و خطا پذیری کم (با توجه به ساختار ذخیره داده در Hadoop) اشاره کرد. در map reduce نیز یک job tracker داریم که به عنوان master node عمل می کند که task ها را به task tracker ها (node برده) اختصاص می دهد. Node ارباب بر همه ی پردازش های task tracker ها نظارت داشته و در صورتی که پردازشی fail شود دوباره اجرا می شود.map reduce در Hadoop به زبان جاوا نوشته شده است.

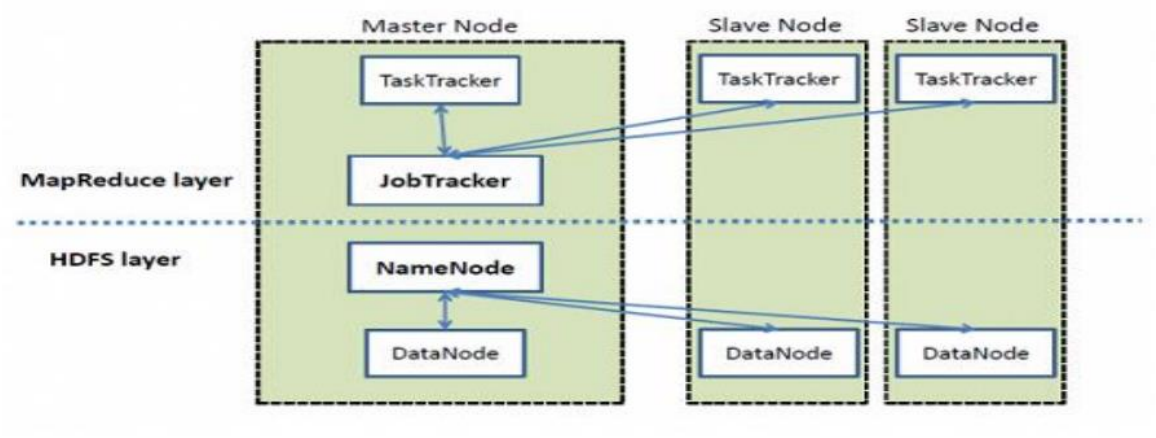
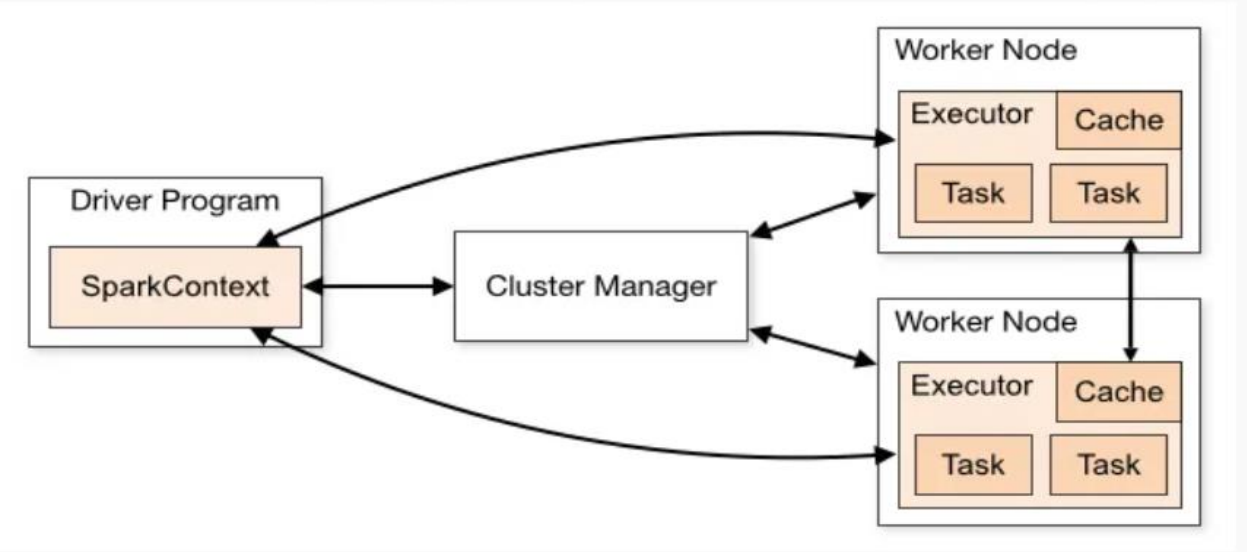


Figure 1 ساختار HDFS و Map-reduce

در ورژن های Hadoop 1.x برای پردازش دیتا فقط از map reduce استفاده می کرد اما از ورژن 2.x کامپوننتی به نام yarn در Hadoop اراعه شد. yarn وظیفه مدیریت منابع(resource manger) و زمان بندی پردازش ها را که قبلا بره عهده map reduce بود را بر عهده گرفت.

Spark:

فریمورک spark نیز از معماری master-slave استفاده می کند. یک node مستر داریم به اسم driver و تعدادی worker (برده) داریم. وقتی که یک برنامه spark اجرا می شود یک context ساخته می شود و عملیاتی که باید انجام شود در نود های worker پخش می شود و مشخصا اینجا هم مثل Hadoop یک بخش مدیریت منابع داریم که اختصاص منابع را مدیریت می کند. (می توان از مدیریت منابع Hadoop yarn هم استفاده کرد)



دو تا مفهوم مهم در spark مطرح هست:

:Resilient Distributed Dataset (RDD)

Spark از RDD برای سریع تر کردن عملیات های map-reduce استفاده می کند. RDD ساختار اساسی داده ها در spark است. RDD ها مجموعه ای از object های هر نوع داده هستند.هم چنین تغییر ناپذیر و در چند تا node از cluster ساکن می شوند و به صورت موازی پردازش انجام می شود. در spark دو عملیات روی RDD انجام می شود:transform (مثل مپ کردن یا فیلتر کردن و..) وaction (مثل reduce کردن و count by key و ...) که بعد از یک RDD جدید ساخته می شود.

ویژگی های ساختارRDD :

1- Lazy evaluation: به این معنی است که در transform ها بلافاصله اجرا نمی شوند و زمانی که یک action به آن ها نیاز دارد اجرا می شوند. با استفاده از مفهوم dag در spark این transform ها track می شوند.
2- In memory computation: تمام پردازش ها در spark به جای هارد دیسک در Ram انجام می شوند که باعث افزایش سرعت پردازش ها می شود.هم چنین داده ها نیز در RAM ذخیره می شوند.
3- Fault Tolerance: مانند Hadoop از تحمل خطای بالایی برخوردارند به خاطر تکرار شدن دیتا در تعداد زیادی node

:Directed Acyclic Graph(DAG)

Dag یا همان گراف جهت دار بدون حلقه یک گرافی هست که راس های گراف RDD های ما هستند و یال های آن عملیات ها یی هستند که روی آن ها انجام می شوند.(transform و action ها) این گراف کمک می کند تا پردازش هایی که قرار است انجام شود با بهترین ترتیب انجام شود تا سریع تر پردازش کنیم. برای مثال اگر چند تا عمل map , filter, reducebykey داریم . در مرحله اول map و فیلتر انجام شود و در مرحله دوم reducebykey

همچنین ساختار این گراف باعث می شود که اگر RDD از دست رفت با توجه والد و عملیات مر بوطه اش بازیابی شود.

Spark	Hadoop
پردازش ها روی حافظه 100 RAM برابر و روی دیسک 10 برابر سریع تر از Hadoop است	**از آنجایی که پردازش ها در دیسک انجام می شوند سرعت پردازش پایین بوده و برای پردازش های real time مناسب نیست**
با توجه به حافظه RAM هزینه بالاتری دارد	**چون از حافظه دیسک استفاده می شود هزینه راه اندازی cluster کم تر است**
به دلیل استفاده از RDD و DAG کنترل خطایی بالایی دارد	**به دلیل این که داده به چند بلاک تقسیم و هر بلاک در چند ماشین ذخیره می شود کترل خطای بالایی دارد**
پردازش ها به صورت batch ,real-time وبا ساختار گراف انجام می شود.(مناسب برای social media هایی مانند اینستاگرام و فسیبوک و...)	**پردازش ها به صورت batches انجام می شود. Map-reduce طی چند مرتبه انجام می شود. خواندن داده ها از ,cluster پردازش آن ها و نوشتن داده ها در cluster**
چند api برای زبان های مختلف دارد و یک interface در قالب command line برای تعامل دارد	**برای کار با Hadoop یک api سطح پایین وجود دارد برای پردازش داده ها و یک مقدار پیچیده و به تعداد کد بیش تری نیاز دارد**
با زبان scala نوشته شده و زبان های java و R و python را پشتیبانی می کند	**به زبان java نوشته شده ولی می توان برنامه ها را با زبان های R , python , c++ نوشت**
بزرگترین کلاستر شناخته شده از 8000 تا node استفاده کرده است.	**دارای مقیاس پذیری بالایی است. با توجه به گزارش یاهو از 42000 node با hadoop استفاده شده است**

شباهت:

هر دو متن باز هستند. در هر دو پردازش ها روی تعدادی node به صورت موازی اجرا می شود.

Spark بیش تر برای کاربرد های real-time به خاطر سرعت بالا استفاده می شود.hadoop برای کاربرد هایی که هزینه کم تری مد نظر باشد استفاده می شود .

سوال 2:

سوال3:

در این کد ابتدا با استفاده از کتابخانه pyspark یک شی از sparkcontext ساختیم و درون متغیر sc قرار دادیم. با این کار برنامه به کلاستر وصل می شود که البته کلاستر اینجا لوکال هست. بعد با استفاده از sc که ساختیم محتویات دیتا ستی که توی دایرکتوری فعلی هست رو میخواند و در قالب rdd ذخیره می کند. پس از آن با mapflat هر خط رو با space هایی که بین کلمات هست جدا می کنیم . بعد با filter کلمه هایی که کاراکتر اولشون حروف الفبا نیست رو کنار میزنیم و بعد از اون هر کلمه رو به صورت (1,کلمه) map می کنیم و بعد با reducebykey کلید هایی که یکسان هستند رو value هاشون رو جمع می کنیم و در آخر نیز نتیجه رو در دایرکتوری out ذخیره می کنیم.(اسم فایل part0000 معمولا ذخیره می شود.)

(در صورتی که برنامه اجرا می شود نباید دایرکتوری out در دایرکتوری فعلی باشد و باید قبل اجرا پاک و توسط برنامه ساخته شود در غیر این صورت خطای این دایرکتوری هم اکنون وجود دارد می دهد)