**2.ce este spark?**

Spark este un alt framework care face acelasi lucru ca si MapReduce. Dar il face mult mai repede.

Spark-ul nu inlocuieste Hadoop, ci extinde MapReduce

**3.Cum a aparut?**

A fost lucrearea de doctorat a lui Matei Zaharia

Matei Zaharia s-a nascut in Romania, dar apoi familia lui s-a mutat in Canada unde a absolvit unitatsitatea Waterloo.

Cand a fost intrebat cum i-a venit ideea de a incepe un asemenea proiect, el a spus ca a lucrat cu primii utilizatori importanti care foloseau Hadoop (Facebook and Yahoo!) si-a vazut limitarile. Asa ca s-a gandit sa faca un framework mai general, cu mai multe functii.  
  
O alta directie importanta pe care s-a axat a fost cea a Machine Learning-ului. Unde algoritmii sunt iterativi ( iar MapReduce la fiecare iteratie scrie rezultatele intermediare pe disc)

**4. 100 TB de date -> recorduri**

explic

**5. Spark vs MapReduce**

Cel mai mare avantaj este ca Spark incearca sa tina datele in memorie, in timp ce MapReduce dupa o serie de pasi scrie datele din memorie pe disk, apoi le citeste si continua procesarea datelor.

**6. Spark vs MapReduce**

Doar pastrand datele in memorie, viteza de procesare a Spark-ului creste de 80 de ori

7 **Spark vs MapReduce**

Este mult mai general si mai usor de folosit, pentru ca are mai multe metode si sunt mult mai intuitive

**8. Procesarea datelor**

Sa ne imaginam ca avem datele memorate ca o colectie sa spunem lista.  
Pe o masina, pe un singur fi de executie putem sa facem acest lucru in felul urmator:

for element in colectie, aplica functia asta

Daca am avea **N** fire de executie putem sa creem un wrapper care sa imparta lista in **N** parti si apoi fiecare fir de executie va procesa o parte din acea lista. Sunt limbaje de programare care au implementat astfel de colectii (ParArray, ParVector)

Cum putem sa procesam datele atunci cand sunt memorate pe mai multe masini?

**9.RDD**

Ideea spark-ului este de a folosi un API cat mai asemanator cu celelalte colectii. Adica sa creeze o abstractie care sa se comporte la fel ca o lista (sa spunem)

Procesarea in cluster presupune ca fiecare masina are cate o bucata din colectia mea. Cu alte cuvinte RDD-urile pot fi privite ca o simpla colectie, iar Spark-ul va face management-ul datelor in spate.

Desenez la tabla o forma geometrica si o impart pe bucati.

**10. Executia unui program in spark**

Programele in spark sunt de tipul master/slave

Executia unui program spark

1. Driver-ul ruleaza aplicatia Spark si creeaza SparkContext-ul de la inceput
2. SparkContextul se conecteaza la managerul clusterului (Mesos/ YARN) care aloca resursele
3. Spark primeste executorii din claster, acetia vor face computatia si vor stoca datele din aplicatie
4. Driver-ul trimite codul aplicatiei catre executori.
5. SparkContextul trimite task-uri executorilor si acestia ruleaza ce li se cere.

**11. Cum se creeaza un RDD**

**­**

**12. Laziness**

Map = exact ca un for doar ca e scris mai usor. Aplic o functie, o transformare pe fiecare element din acea colectie. Rezultatul este un nou RDD

Filter = aplic o functie predicat pe fiecare element si creez un un RDD doar cu acele elemente pentru care predicatul este true

Count = returnez numarul de elemente din RDD

Foreach = la fel ca si map-ul aplic o functie peste fiecare element din colectie. Dar nu intoarce nimic ca rezultat.

**13. WordCount**

Sa explic flatMap: Daca as fi folosit map, aveam un RDD[Array[String]] asa am doar RDD[String]

\_ din reducebyKey semnifica: orice element, nu ma intereseaza numele lui pentru ca o sa-l folosesc doar aici

14. code examples

creez la tabla DAG-ul

spark-ul se opreste atunci cand a gasit 10 elemente care trec de filtru

**15. poate o sa povestesc pe acelasi exemplu**

Sunt mai multe tipuri de persist:

In memorie

Pe disc

Si in memorie si pe disc

Trebuiesa tinem cont de faptul ca atunci cand tinem in memorie anumite colectii,vom ramane cu mai putina memorie pentru calcularea celorlalte RDD-uri

Atunci cand alegem sa tinem un RDD pe disc trebuie sa evaluam daca e variant cea mai buna si daca nu cumva scrierea pe disc si citirea dureaza mai mult decat calcularea inca o data a respectivului RDD

**16. O noua abstractie - dataFrame**

As fi putut sa scriu ***un nou API*** , dar de fapt, API-ul folosit pentru RDD-uri a ramas aproape acelasi, cu foarte mici modificari, iar API-ul de la RDD-ul este aproape la fel ca operatiile pe care le poti face pe orice colectie in scala. Ceea ce e un avantaj. Exista continuitate.

**17.quizz**

Ce se intampla daca am avea urmatorul cod:

ce se intampla pe driver?

NIMIC pentru ca foreach este o actiune si se va executa pe worker. Rezultatul este intors la driver, dar foreach nu intoarce nici un rezultat

**18.quizz**

19.

Prin diverse mecanisme se genereaza date, care sunt modelate cu ajutorul Machine Learning-ului,

prin Big data putem pune aceste informatii intr-un context si sa formam cunostinte, si incepe sa se vorbeasca deja despre Internet of knowlodge.

iar aceste cunosinte, daca sunt manipulate asa cum trebuie pot crea aplicatii la care inca nici nu ne ganditi ca e posibil sa fie.

Dar totul pleaca de la colectia de date, tehnologiile cu care pot fi procesate aceste date (Spark) si modalitatile de la procesa datele brute (Machine Learning)

20.

Google brain team au facut urmatorul experiment: Au creat 3 retele neuronale: Alice, Bob si Eve. Alice si Bob au schimbat initial niste numere random intre ei (o posibila cheie)  
  
 Alice trebuie sa trimita un mesaj criptat lui Bob

Bob trebuia sa de decripteze mesajul

Eve trebuia sa incerce sa intercepteze mesajul si sa-l descifreze.

Dupa 15.000 de iteratii Bob putea sa decripteze mesajul de la Alice, iar Eve reusea sa descrifreze doar 8 din cei 16 biti.

Dar inca nu a fost pus in practica pentru sistemul este prea complex pentru a fi inteles si oamenii se tem sa nu aiba vulnerabilitati.

Alt experiement interesant este cel in care 2 algoritmi se antrenau reciproc: unul invata sa minta si celalalt sa descopere cand e mintit.

Acum imaginati-va puterea machine-learningului aplicata in Cloud, peste BigData. Incetul cu incetul incepe sa prinda contur un nou concept DsaaS = Data Science as a Service.