Hadoop/Storm/Spark

lunedì 23 settembre 2019 14:43

HADOOP

È l'implementazione delle Map Reduce sviluppata da Apache. Deve essere fornito un codice scritto in Java/C++, dove esprimi un codice che prende un certo tipo Tin in input e che restituisce dopo le fase di reduce il tipo Tout in output.

↳ <Key, Tout>

L'obiettivo che ci si pone con Hadoop è suddividere i dati su cui vogliamo lavorare in tante macchine per fare in modo di parallelizzare l'accesso ai dati che è la principale bottleneck di un sistema parallelo di questo tipo perché con il tempo performance e velocità di connessione sono migliorate ma l'accesso ai dati no.
Si usa un file system chiamato HDFS che ci permette di avere una replica dei dati in modo che sia garantito sempre l'accesso ai dati ma deve anche gestire la distribuzione di questi dati.

Fusione è una possibile ottimizzazione che possiamo avere in Hadoop e mi permette di evitare due map in una sola map.

STORM

Qua si lavora su uno stream in parallelo e non su dati che sono già sulla nostra macchina.
Storm ci permette di avere una buona performance nell'esecuzione delle funzioni sullo stream ma è particolarmente adatto solamente per lavori complicati, per le cose semplici non è troppo adatto.
Storm organizza la computazione in una sorta di grafo in cui abbiamo dei nodi di vario tipo:

→ Spouts: sono nodi che hanno solo l'output e producono lo stream

→ Bolt: nodi che prendono un input, fanno qualcosa e poi producono un output sempre come stream.

→ Topologies: sono un'unione di nodi Spouts e di nodi bolts.

Il grafo di Storm ci permette di gestire la fault tolerance perché siamo sicuri che quando viene prodotto uno stream i dati arriveranno fino alla fine dello stream (nodo finale)

SPARK

È una versione più recente di Storm ma permette più o meno di fare le stesse cose sporcando comunque su stream.
Permette di fare due cose:

→ SparkSQL ci fa lavorare su un db distribuito usando SQL

→ Stream: riusciamo a lavorare su uno stream con delle funzioni più recenti e migliori rispetto a quelli di Storm

→ MLlib

→ GraphX

In particolare Spark ci fa lavorare sugli stream e ogni volta che arriva una tappa su cui lavorare la memorizziamo anche in un database.
Ci sono vari modi per aggiornare questo database:
- Possiamo ricreare ogni volta che arrivano nuovi dati
- Possiamo fare un append
- Possiamo fare un update