Apache Storm :::

- Apache Storm is an Elastically scalable , Highly avaaibale and Fault tolerant distributed real time stream processing engine .
-
- Storm is a stateless technology and zookeeper is responsible to storing the state of master and supervisor machines .
-
- Storm architecture :::
- One JVM machine runs a process called Nimbus . It acts as a master process .
- There can be N worker nodes which run a process called supervisor .
- All these machines are being managed by zookeeper .
- Here Nimbus is the master node responsible for assignment and management of work being executed

in worker nodes. It monitors them for failures and balance work among them .

-
- Streaming patterns
- Streaming Joins :::
- Some bolts join single single elements of two streams emitting from two spouts/bolts and some join tuples in a finite window of time
- Batching topology Pattern . In this case we keep on holding some data in memory for some time and finally emit the data to next bolt and acknowledgement the all tuples together .
- Basic pattern ::: Processing one at a time and applying some filter logic or small transformation logic .
- In Memory caching a field grouping combo . If we need to save some counts and average on the basis of some field in the tuple then all the tuples with that same field should go

to the same thread holding that cache .

-
- What are various examples of real time systems .
- Real time Trends , Real time conversations millions of conversations keep going in real time , Product recommendation in real time for adds , Real time search like you will be able to search all the tweets happened in last seconds or few milli seconds . After some milliseconds you will be able to search a tweet. Particle physics , Fraud detection of credit card transaction that has to be done in few milli seconds only .
- Suppose a stream of data is coming at 1 Million events/s then the system must be able to handle the requests and do analytics is a way that processing rate >= ingestion rate . But if Ingestion rate will become very high then system will not be more real

time .

- Now how to increase the processing rate . Answer is parallelism . Suppose some process is putting the stream on Kafka topic with a very high number of partitions then there can be Kafka stream workers reading it parallely and doing some analytics . More parallelism can match the ingestion rate .
- Now suppose if we keep large number of portions of the topic and equal number of threads check the stream and see if some data is having some special keywords like ironman , batman then it can count them and emit a new stream having their count and a final collector process can sum it up in redis and produce the real time stream of final count .
- Problems ::: Very difficult to setup . Large Kafka cluster is required .
- Fault tolerance is not good as if some fails the process will migrate to new

thread and data backed in Kafka will be moved to data worker as well .

- Almost once gives undercount of analytics and at-least once gives over count of some analytics and exactly once is exact . Almost once is cheapest and at least once is middle and exactly once is costliest .
-
- Hadoop vs Storm ::: Hadoop is very fast batch analytics and Storm is realtime analytics . If the data rate coming is higher than the time a Hadoop job will run then real time processing needs will not be met .
-
- Grouping :::
- Shuffle grouping is road robin to next bolt .
- Field grouping is based on some field and hashing and same things will keep going on same bolt .
- All Grouping all things will be going to one bolt .

- Global Groping is like broadcast .