# storm

## storm简介

Storm是一个免费并开源的分布式实时计算系统。利用Storm可以很容易做到可靠地处理无限的数据流，像Hadoop批量处理大数据一样，Storm可以实时处理数据。Storm简单，可以使用任何编程语言。

在Storm之前，进行实时处理是非常痛苦的事情: 需要维护一堆消息队列和消费者，他们构成了非常复杂的图结构。消费者进程从队列里取消息，处理完成后，去更新数据库，或者给其他队列发新消息。

这样进行实时处理是非常痛苦的。我们主要的时间都花在关注往哪里发消息，从哪里接收消息，消息如何序列化，真正的业务逻辑只占了源代码的一小部分。一个应用程序的逻辑运行在很多worker上，但这些worker需要各自单独部署，还需要部署消息队列。最大问题是系统很脆弱，而且不是容错的：需要自己保证消息队列和worker进程工作正常。

Storm完整地解决了这些问题。它是为分布式场景而生的，抽象了消息传递，会自动地在集群机器上并发地处理流式计算，让你专注于实时处理的业务逻辑。

## Storm的特点

Storm有如下特点：

编程简单：开发人员只需要关注应用逻辑，而且跟Hadoop类似，Storm提供的编程原语也很简单

高性能，低延迟：可以应用于广告搜索引擎这种要求对广告主的操作进行实时响应的场景。

分布式：可以轻松应对数据量大，单机搞不定的场景

可扩展： 随着业务发展，数据量和计算量越来越大，系统可水平扩展

容错：单个节点挂了不影响应用

消息不丢失：保证消息处理

不过Storm不是一个完整的解决方案。使用Storm时你需要关注以下几点：

如果使用的是自己的消息队列，需要加入消息队列做数据的来源和产出的代码

需要考虑如何做故障处理：如何记录消息队列处理的进度，应对Storm重启，挂掉的场景

需要考虑如何做消息的回退：如果某些消息处理一直失败怎么办？

## Storm的应用

跟Hadoop不一样，Storm是没有包括任何存储概念的计算系统。这就让Storm可以用在多种不同 的场景下：非传统场景下数据动态到达或者数据存储在数据库这样的存储系统里（或数据是被实时操控其他设备的控制器(如交易系统)所消费）

Storm有很多应用：实时分析，在线机器学习(online machine learning)，连续计算(continuous computation)，分布式远程过程调用(RPC)、ETL等。Storm处理速度很快：每个节点每秒钟可以处理超过百万的数据组。它是可扩展(scalable)，容错(fault-tolerant)，保证你的数据会被处理，并且很容易搭建和操作。

例如Nathan Marz提供的例子，产生Twitter的趋势信息。Twitter从海量推文中抽取趋势信息，并在本地区域和国家层级进行维护。这意味者一旦一个案例开始出现，Twitter的话题趋势算法就能实时的鉴别出这个话题。这个实时的算法就是通过在Storm上连续分析Twitter数据来实现的。

推荐系统:典型应用淘宝,京东推荐

网站统计:实时销量,流量统计,如淘宝双11效果图

## storm原理

### storm集群结构

主节点(Nimbus),工作节点(Supervisor),作业(topologies,拓扑,死循环)

Nimbus和Supervisors之间所有的协调工作是通过一个Zookeeper集群

Nimbus进程和Supervisors进程是无法直接连接和无状态的;所有的状态维持在Zookeeper中货保存在本地磁盘上

这意味着你可以kill -9 Nimbus货Supervisors进程,而不需要做备份.

这种设计导致storm集群具有令人难以置信的稳定性,即无耦合

### 原理

Nimbus负责在集群分发的代码,topo只能在Nimbus机器上提交,将任务分配给其他机器,和故障检测

Supervisor监听分配给它的节点,根据Nimbus的委派在必要时启动和关闭工作进程,每个工作进程执行topology的一个子集,一个运行中的topology由很多运行在很多机器上的工作进程组成

在Storm中有对于流stream的抽象,流是一个不间断的无界的连续tuple,注意Storm在建模事件流时,把流中的事件抽象为tuple即元组