达观数据应对大规模消息处理的经验

前言

达观数据作为一家提供大数据处理的公司，经常会遇到类似于客户上用户点击行为的需求。这样的上报请求不要求马上返回处理结果， 而是需要后台将一系列的上报数据进行统一归档整理挖掘之后将结果数据呈现给客户。上述的的业务需求形成了一个经典的生产者（producer）消费者（consumer）模型， 也就是说我们需要一个系统在生产者（客户上报数据）和消费者（后台数据处理）进行沟通。换句话说我们需要一个各个系统之间通信的消息系统。

然而好消息是有一个消息系统正好是为了应对这种业务场景而生的，这就是kafka。那么kafka到底是一个什么样的消息系统？有什么特点？实际吞吐表现又如何呢？带着这些问题，我们一起来了解一下。

**Kafka**简介：

首先根据官网（http://kafka.apache.org/intro）介绍，我们知道kafka是一个分布式流处理平台，一个可处理企业级消息发布/订阅的消息系统，并且具有高容错性和消费及时性，那么它是怎么做到这一点的呢？接着往下看。

一， 主题和日志：

原理简介：

主题(topic)和日志(log)的设置kafka的一大特色，一个kafka集群可以创建多个topic， 每一个topic都相当于一个消息队列，这也就意味着我们可以将不同格式的数据发布到不同的topic中，减小消费这些数据时的逻辑难度。那么每一个topic中处理的数据结构是怎样的呢？我们先来看一张topic的解剖图：

从图中我们可以看到， 消息传送过来时kafka会通过负载均衡将这消息最终写入到磁盘上一个特定分区（partition）。由于在同一个partition上这些消息都是顺序存储的， 所以对一个特定的分区每一条消息都会有一个基于起始位置的偏移量（offset）， 因此我们在后续的消费时只需要指明从哪个partition中的哪个offset开始消费，就能达到重复消费的目的。

**Q&A**：

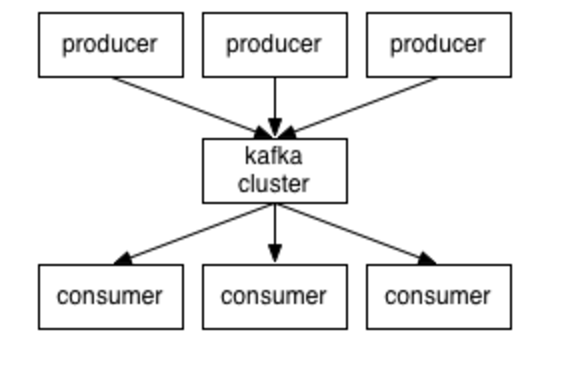
Q:虽然kafka已经有partition的方式来增加负载，但是他的数据最终是写入到磁盘中，比如机械磁盘 而磁盘的写入效率是低下的， 难道我们需要增大一个topic的负载需要给它设置更多的partition吗？

A:机械磁盘驱动器的吞吐量跟寻道延时是强关联的，也就是说，线性写速度远大于随机读写：例如在67200rpm SATA RAID-5 的磁盘阵列中， 随机写的速度大约是100k/s, 然而线性写的速度可以达到600M/s，后者大约是前者的6000倍。如前文所说， kafka采用的即是后者, 利用操作系统的read-ahead和write-behind技术，极大的提升访问性能；设置partition的数量固然可以增大topic的负载，但是partition过多会导致cpu计算量增大，所以对于不同配置的机器， 需要根据业务场景设置不同数量的partition。

Q: 偏移量offset的存储类型是什么， 如果我的消息足够大，那么offset是否会重新置0， 如果置0，以后的消费是否会紊乱？

A: kafka offset 是一个日志序列号（ log sequence number），不必担心offset 长度的问题。那么这个日志序列号到底有多大呢，举个例子：如果你一个partition一天接收1Tb的日志， 这个offset至少可以使用1百万年。

Q: 写入磁盘的日志会被永久保留吗？如果想删除过期的消息， 需要怎么操作？

A:可以通过配置文件中log.retention参数设置过期时间，超过过期时间的消息会被系统删除，删除的消息不可再被重新消费。

二，分布式集群

通过前文的介绍我们已经瞭解到kafka通过partition和顺序读写磁盘的方式达到很高的吞吐量，可是单台机器的吞吐量再高一旦该机发生故障宕掉会对业务产生灾难性影响，怎么处理这个问题呢？想必你已经知道了，那就是采用集群的方式，一旦一台机器发生故障客户端可以选择链接其它的机器， 保证业务稳定性。根据官网说明，我们知道在kafka配置文件中有一个复制因子的控制参数（default.replication.factor）， 表示一份数据会做的备份次数，所以要想保证数据完整性，当设置复制因子为n时，允许n-1台实例（broker）宕掉。

Q&A:

Q:搭建kafka集群时有没有什么小细节需要值得注意的？

A:kafka官网（http://kafka.apache.org）已经有详细的搭建过程，在此不赘述。建议正式项目中不要采用伪集群（多个broker运行在同一台物理机上）的搭建方式，而且zookeeper集群和kafka集群最好不要出现在同一台实体机上，这样会影响kafka顺序读写的效率。

三，生产者消费者和消费的强顺序：

上面讲了那么多，无非是要实现一个队列的数据结构，由此可以想见对于kafka的一个topic来说，生产消费逻辑应该是这样的（如下图），有很多生产者向topic中写入数据，另外一端则有许多的消费者消费数据。

然而实际上kafka生产者消费者模式有他的特殊性那么kafka这个队列是怎样实现入队和出队的呢？接下来我们一起来看看kafka的生产者消费者模式。

生产者：生产者（producer）顾名思义，就是向kafka队列中发布消息的，是一个入队的操作。生产者的功能是在选中的topic中选择一个partion 然后向这个partition中发送数据。选择partition的过程就是一个负载均衡的方式， 比如可以采用轮询或者自己设定partition选择函数来实现负载均衡。当然如果你使用封装的api比如（https://github.com/dpkp/kafka-python）就可以不需要关心负载均衡的问题。

消费者： 消费者（consumer）的功能是从队列中读出数据并进行相应的逻辑处理，但是kafka的消费者有他的特殊之处。kafka增加了一个组（group）的概念，一个topic可以有多个group， 当多个consumer从属于一个组时，一条消息将被发往所有的组，但是在组内，这条消息只会被一个consumer消费。由此说来一个group才是一个“逻辑消费者（logic consumer）”。相关逻辑如下图所示：

图中展示了由2个broker组成的集群在一个有4个partition的topic被分属于2个group的6个consumer消费的原理图。

消息强顺序：通过上图我们知道某一个消息点的消费情况，那么一个消息流的消费情况会是怎样的呢？其实在高等级的api中由于指定了负载均衡的规则，生产者发布两条不同的消息数据时会根据相应的规则发送到一个特定的partition中，在消费时会按照同样的规则从partition中取出出数据，这样就能保证两条数据的顺序，从而达到消息的强顺序的特性。

Q&A：

Q:对于一个具有多个consumer的topic，我要实现一条消息被多个consumer消费和一条消息只被一个consumer消费，那我需要怎么设置group？

A:将多个consumer设置为同一个组可以实现一条消息只被这个组中的一个consumer消费， 将所有的consumer都设置为不同的组，一条消息将会被所有的consumer消费。

**Kafka**性能测试：

测试环境：

1， broker 数量：3

2， 备份因子数：2

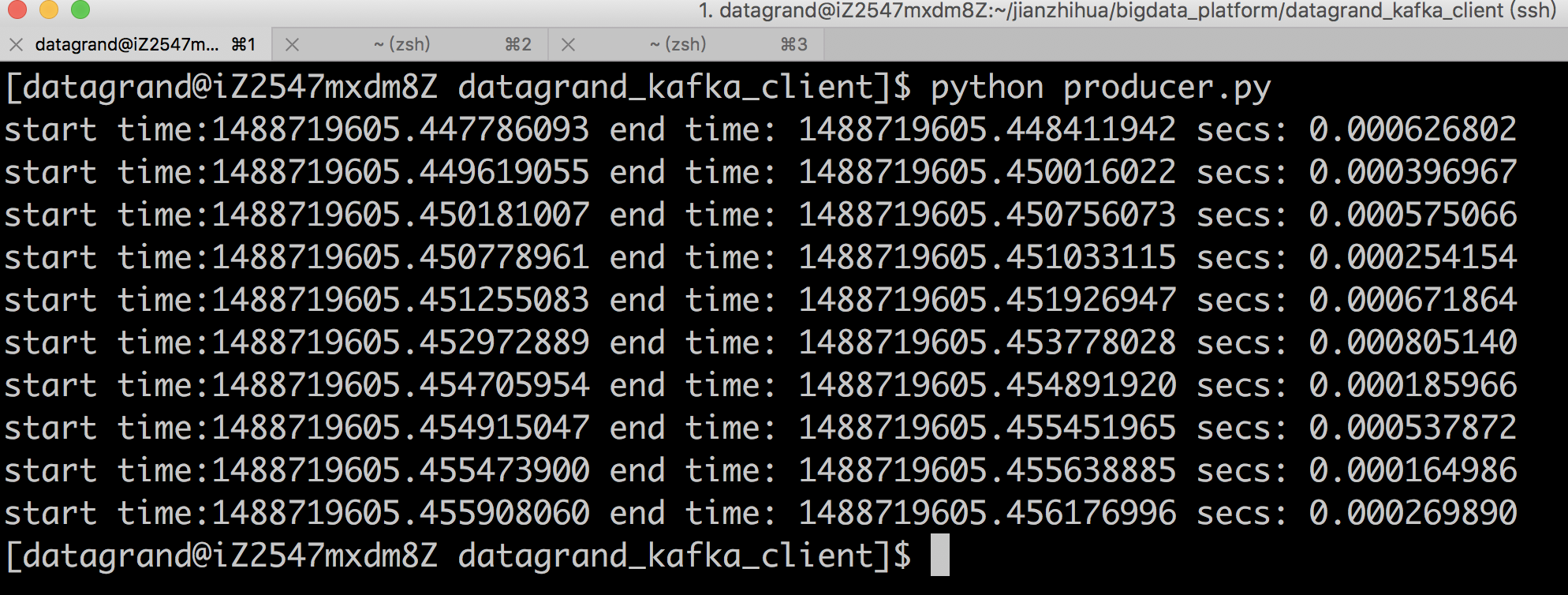
3， 磁盘信息：200G普通机械硬盘

4， cpu参数：8核8线程

5， 语言： Python2.7

6, 客户端： kafka-python

7, partition 数量： 5

单进程producer测试（如下图）：

统计上图数据可知单进程发送消息数： 10 条， 平均延时：0.004488707， qps竟然可以达到2000，这样的成绩无疑是惊人的。那么在多进程的情况下你的表现还会好吗？我们设置10个进程，看看kafka在10个进程下的延时会有较大的变化吗？如下图（打印消息过多，截取部分图）：

由图可知10 个进程在每个进程发送10条消息，平均延时为0.00050380466秒, qps也接近2000，由此可见在多进程的情况下表现依然良好。

总结：

通过上面的描述相信你对kafka作为一个消息系统使用已经有一些瞭解了， 但是其实kakfa不是只能做消息系统使用， 根据官网的介绍kafka还可以用作网站行为追踪（Website Activity Tracking）、数据监控， 流处理等，如果你正好想使用kafka希望这篇文章能对你有所帮助， 当然如果有不同的想法，欢迎讨论， 谢谢！