07 | 消息积压了该如何处理?

李玥 · 消息队列高手课

你好，我是李玥。这节课我们来聊一聊关于消息积压的问题。

据我了解，在使用消息队列遇到的问题中，消息积压这个问题，应该是最常遇到的问题了，并且，这个问题还不太好解决。

我们都知道，消息积压的直接原因，一定是系统中的某个部分出现了性能问题，来不及处理上游发送的消息，才会导致消息积压。

所以，我们先来分析下，在使用消息队列时，如何来优化代码的性能，避免出现消息积压。然后再来看看，如果你的线上系统出现了消息积压，该如何进行紧急处理，最大程度地避免消息积压对业务的影响。

优化性能来避免消息积压

在使用消息队列的系统中，对于性能的优化，主要体现在生产者和消费者这一收一发两部分的业务逻辑中。对于消息队列本身的性能，你作为使用者，不需要太关注。为什么这么说呢?

主要原因是，对于绝大多数使用消息队列的业务来说，消息队列本身的处理能力要远大于业务系统的处理能力。主流消息队列的单个节点，消息收发的性能可以达到每秒钟处理几万至几十万条消息的水平，还可以通过水平扩展 Broker 的实例数成倍地提升处理能力。

而一般的业务系统需要处理的业务逻辑远比消息队列要复杂，单个节点每秒钟可以处理几百到几千次请求，已经可以算是性能非常好的了。所以，对于消息队列的性能优化，我们更关注的是，**在消息的收发两端，我们的业务代码怎么和消息队列配合，达到一个最佳的性能。**

1. 发送端性能优化

发送端业务代码的处理性能，实际上和消息队列的关系不大，因为一般发送端都是先执行自己的业务逻辑，最后再发送消息。**如果说，你的代码发送消息的性能上不去，你需要优先检查一下，是不是发消息之前的业务逻辑耗时太多导致的。**

对于发送消息的业务逻辑，只需要注意设置合适的并发和批量大小，就可以达到很好的发送性能。为什么这么说呢?

我们之前的课程中讲过 Producer 发送消息的过程，Producer 发消息给 Broker，Broker 收到消息后返回确认响应，这是一次完整的交互。假设这一次交互的平均时延是 1ms，我们把这 1ms 的时间分解开，它包括了下面这些步骤的耗时：

- 发送端准备数据、序列化消息、构造请求等逻辑的时间，也就是发送端在发送网络请求之前的耗时；
- 发送消息和返回响应在网络传输中的耗时；
- Broker 处理消息的时延。

如果是单线程发送，每次只发送 1 条消息，那么每秒只能发送 1000ms / 1ms * 1 条 /ms = 1000 条 消息，这种情况下并不能发挥出消息队列的全部实力。

无论是增加每次发送消息的批量大小，还是增加并发，都能成倍地提升发送性能。至于到底是选择批量发送还是增加并发，主要取决于发送端程序的业务性质。简单来说，只要能够满足你的性能要求，怎么实现方便就怎么实现。

比如说，你的消息发送端是一个微服务，主要接受 RPC 请求处理在线业务。很自然的，微服务在处理每次请求的时候，就在当前线程直接发送消息就可以了，因为所有 RPC 框架都是多线程支持多并发的，自然也就实现了并行发送消息。并且在线业务比较在意的是请求响应时延，选择批量发送必然会影响 RPC 服务的时延。这种情况，比较明智的方式就是通过并发来提升发送性能。

如果你的系统是一个离线分析系统，离线系统在性能上的需求是什么呢？它不关心时延，更注重整个系统的吞吐量。发送端的数据都是来自于数据库，这种情况就更适合批量发送，你可以批量从数据库读取数据，然后批量来发送消息，同样用少量的并发就可以获得非常高的吞吐量。

2. 消费端性能优化

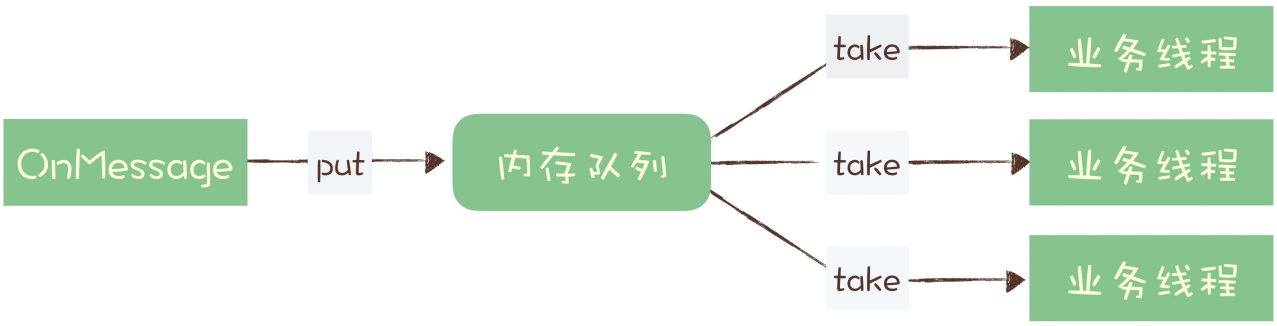
使用消息队列的时候，大部分的性能问题都出现在消费端，如果消费的速度跟不上发送端生产消息的速度，就会造成消息积压。如果这种性能倒挂的问题只是暂时的，那问题不大，只要消费端的性能恢复之后，超过发送端的性能，那积压的消息是可以逐渐被消化掉的。

要是消费速度一直比生产速度慢，时间长了，整个系统就会出现问题，要么，消息队列的存储被填满无法提供服务，要么消息丢失，这对于整个系统来说都是严重故障。

所以，我们在设计系统的时候，**一定要保证消费端的消费性能要高于生产端的发送性能，这样的系统才能健康的持续运行。**

消费端的性能优化除了优化消费业务逻辑以外，也可以通过水平扩容，增加消费端的并发数来提升总体的消费性能。特别需要注意的一点是，**在扩容 Consumer 的实例数量的同时，必须同步扩容主题中的分区（也叫队列）数量，确保 Consumer 的实例数和分区数量是相等的。**如果 Consumer 的实例数量超过分区数量，这样的扩容实际上是没有效果的。原因我们之前讲过，因为对于消费者来说，在每个分区上实际上只能支持单线程消费。

我见到过很多消费程序，他们是这样来解决消费慢的问题的：



它收消息处理的业务逻辑可能比较慢，也很难再优化了，为了避免消息积压，在收到消息的 OnMessage 方法中，不处理任何业务逻辑，把这个消息放到一个内存队列里面就返回了。然后它可以启动很多的业务线程，这些业务线程里面是真正处理消息的业务逻辑，这些线程从内存队列里取消息处理，这样它就解决了单个 Consumer 不能并行消费的问题。

这个方法是不是很完美地实现了并发消费？请注意，这是一个非常常见的错误方法！ 为什么错误？因为会丢消息。如果收消息的节点发生宕机，在内存队列中还没来及处理的这些消息就会丢失。关于“消息丢失”问题，你可以回顾一下我们的专栏文章《05 | 如何确保消息不会丢失？》。

消息积压了该如何处理？

还有一种消息积压的情况是，日常系统正常运转的时候，没有积压或者只有少量积压很快就消费掉了，但是某一个时刻，突然就开始积压消息并且积压持续上涨。这种情况下需要你在短时间内找到消息积压的原因，迅速解决问题才不至于影响业务。

导致突然积压的原因肯定是多种多样的，不同的系统、不同的情况有不同的原因，不能一概而论。但是，我们排查消息积压原因，是有一些相对固定而且比较有效的方法的。

能导致积压突然增加，最粗粒度的原因，只有两种：要么是发送变快了，要么是消费变慢了。

大部分消息队列都内置了监控的功能，只要通过监控数据，很容易确定是哪种原因。如果是单位时间发送的消息增多，比如说是赶上大促或者抢购，短时间内不太可能优化消费端的代码来

提升消费性能，唯一的方法是通过扩容消费端的实例数来提升总体的消费能力。

如果短时间内没有足够的服务器资源进行扩容，没办法的办法是，将系统降级，通过关闭一些不重要的业务，减少发送方发送的数据量，最低限度让系统还能正常运转，服务一些重要业务。

还有一种不太常见的情况，你通过监控发现，无论是发送消息的速度还是消费消息的速度和原来都没什么变化，这时候你需要检查一下你的消费端，是不是消费失败导致的一条消息反复消费这种情况比较多，这种情况也会拖慢整个系统的消费速度。

如果监控到消费变慢了，你需要检查你的消费实例，分析一下是什么原因导致消费变慢。优先检查一下日志是否有大量的消费错误，如果没有错误的话，可以通过打印堆栈信息，看一下你的消费线程是不是卡在什么地方不动了，比如触发了死锁或者卡在等待某些资源上了。

小结

这节课我们主要讨论了 2 个问题，一个是如何在消息队列的收发两端优化系统性能，提前预防消息积压。另外一个问题是，当系统发生消息积压了之后，该如何处理。

优化消息收发性能，预防消息积压的方法有两种，增加批量或者是增加并发，在发送端这两种方法都可以使用，在消费端需要注意的是，增加并发需要同步扩容分区数量，否则是起不到效果的。

对于系统发生消息积压的情况，需要先解决积压，再分析原因，毕竟保证系统的可用性是首先要解决的问题。快速解决积压的方法就是通过水平扩容增加 Consumer 的实例数量。

思考题

课后请你思考一下，在消费端是否可以通过批量消费的方式来提升消费性能？在什么样场景下，适合使用这种方法？或者说，这种方法有什么局限性？欢迎在留言区与我分享讨论。

感谢阅读，如果你觉得这篇文章对你有一些启发，也欢迎把它分享给你的朋友。

精选留言 (108)

大白给小白讲故事
2019-08-06

1、要求消费端能够批量处理或者开启多线程进行单条处理
2、批量消费一旦某一条数据消费失败会导致整批数据重复消费
3、对实时性要求不能太高，批量消费需要Broker积累到一定消费数据才会发送到Consumer

作者回复: 👍👍👍

共 5 条评论>　　89

约书亚
2019-08-06

批量消费有意义的场景要求：1.要么消费端对消息的处理支持批量处理，比如批量入库 2. 要么消费端支持多线程/协程并发处理，业务上也允许消息无序。3. 或者网络带宽在考虑因素内，需要减少消息的overhead。
批量消费的局限性：1. 需要一个整体ack的机制，一旦一条靠前的消息消费失败，可能会引起很多消息重试。2. 多线程下批量消费速度受限于最慢的那个线程。
但其实以上局限并没有影响主流MQ的实现了批量功能。

共 8 条评论>　　69

iLeGeND
2019-08-06

老师好，我一直理解，消息积压不是一种正常的现象吗？来不及处理的消息先在消息队列中存着，缓解下游系统的压力，让上下游系统在时间上解偶，，听了今天的课，感觉理解的不太一样，希望老师解答一下

作者回复: 消息积压是正常现象，积压越来越多就需要处理了。

就像一个水库，日常蓄水是正常的，但下游泄洪能力太差，导致水库水位一直不停的上涨，这个就不正常了。

共 10 条评论>　　59

严青

2019-12-02

我没读过这篇文章之前我看过别人面试的解决消息积压的方法：

（1）临时扩容，增加消费端，用硬件提升消费速度。

（2）服务降级，关闭一些非核心业务，减少消息生产。

（3）通过日志分析，监控等找到挤压原因，消息队列三部分，上游生产者是否异常生产大量数据，中游消息队列存储层是否出现问题，下游消费速度是否变慢，就能确定哪个环节出了问题

（4）根据排查解决异常部分。

（5）等待积压的消息被消费，恢复到正常状态，撤掉扩容服务器。

到此，问题解决

 48

C J J

2019-08-08

一、如何预防消息积压?

1、发送端提高并发及批量大小；

2、消费端增加实例且同步宽容分区；

二、如何处理消息积压?

1、消费端扩容；

2、服务降级；

3、异常监控。

共 2 条评论> 25

linqw

2019-08-06

尝试回答下课后习题，老师有空帮忙看下哦

消费端进行批量操作，感觉和上面的先将消息放在内存队列中，然后在并发消费消息，如果机器宕机，这些批量消息都会丢失，如果在数据库层面，批量操作在大事务，会导致锁的竞争，并且也会导致主备的不一致。如果是一些不重要的消息如对日志进行备份，就可以使用批量操作之类的提高消费性能，因为一些日志消息丢失也是可以接受的。

作者回复: 非常好!

共 4 条评论> 26

Jxin
2019-08-06

1.无法提升消费业务效率（仅受消费业务自身逻辑影响），但可以提高mq中堆积消息消费的整体吞吐量（批推比单推mq耗时较短）。
2.数据增量同步，监控信息采集。（非核心业务的稳定大数据流操作）。
3.批处理意味数据积累和大数据传输，这会让单次消费的最长时延变长。同时批量操作为了保证当前批量操作一致性，在个别失败的情况下会引发批量操作重试。

作者回复: 总结的非常好!

 23

lecy_L
2019-08-16

消息积压处理：
1、发送端优化，增加批量和线程并发两种方式处理
2、消费端优化，优化业务逻辑代码、水平扩容增加并发并同步扩容分区数量
查看消息积压的方法：
1、消息队列内置监控，查看发送端发送消息与消费端消费消息的速度变化
2、查看日志是否有大量的消费错误
3、打印堆栈信息，查看消费线程卡点信息

作者回复: 👍👍👍

共 3 条评论> 13

SunshineBoy
2020-03-04

如何判断增加多少consumer消费实例的个数?

作者回复: 你可以简单计算一下，消费并行度：单实例平均消费tps * 消费并行度 > 生产消息的总tps

消费并行度 = min（consumer实例数，分区数量）

 12

亚洲舞王.尼古拉斯赵...
2019-08-06

如果使用了批量消费的方式，那么就需要批量确认，如果一次消费十条消息，除了第七条消费失败了，其他的都处理成功了，但是这中情况下broker只能将消费的游标修改成消息7，而之后的消息虽然处理成功了，但是也只能使用类似于拉回重传的方式再次消费，浪费性能，而且这种批量消费对于消费者的并发我觉得不是很友好，可能消费者1来了取走了十条消息在处理，这时候消费者2过来了也想取十条消息，但是他需要等待消费者1进行ack才可以取走消息，不知道说的对不对，请老师指正

作者回复: 是这样的。

12

长期规划
2019-09-02

老师，如果onMessage方法中，收到消息后不确认，等真正处理完消息再确认，就可以了吧，这样就可以用内存队列了

作者回复: 理论上是可以的，但你要注意，像RocketMQ，采用默认配置的时候，onMessage方法结束后，如果没抛异常，默认就会自动确认了。

共 2 条评论>

10

天涯煮酒
2019-09-06

一开始不理解为啥Producer发得慢了会导致消息积压，明明是发得少了Broker端的消息应该更少啊

后来想到前面章节有提到消息事务，Producer首先会开启事务并发送一个半消息，再执行业务逻辑，最后提交事务，如果发得慢了会导致半消息在Broker中的时间增长，导致积压

共 3 条评论>

7

涛涛
2020-04-13

临时扩容消息分区，已堆积的消息会转移到新分区上吗?

作者回复: 不会的。

共 4 条评论>

6

传志
2020-03-09

老师想问下，为什么生产端性能问题怎么会引起消息堆积呀

作者回复: 生产端发送慢不会引起消息积压的。

6

grey927
2019-08-26

发送端如果多线程，如何实现之前您在《确保消息不会丢失》这篇文章中说的：
发送端 把发送消息标识为每次增加1的效果？

作者回复: 可以使用一个自增的原子变量，比如Java中的AtomicLong。

共 2 条评论>

5

Stenvien
2019-08-11

如果消费者消费异常，即使多次消费也无法成功处理（如消息格式异常），导致一直无法成功ack此条消息，这种场景一般要怎么处理？

我想到有2种：
1. 不做任何处理，消费者会一直卡在此消息的处理上，那么后面的所有消息都没机会处理了，只能靠监控发现消费延迟，发出告警，人工修复。这种处理方式会导致一条有问题的消息就影响了整个业务
2. 数据库存储此异常的消息，并发告警，人工修复，仍然ack此消息，继续消费后面的消息。但是，若对消息的处理顺序有依赖，若没有成功处理此异常消息，消费的后面的消息的处理可能会有问题

是否有更好的处理方式？

作者回复: 有的消息中间件提供了“死信队列”的功能，它会自动把这种反复消费都失败的消息丢到一个特殊的死信队列中，避免一条消息卡主队列的情况。

5

快快
2019-08-19

老师，kafka扩容时会发出reblance吧

作者回复: 不仅是扩容时候，只要是consumer和partition有一方的数量变化，都会触发reblance。

4

guoguo
2019-08-09

只能在消费端开多个线程并行消费。

题目所说的批量消费是什么意思，多次取消息，放在内存里，批量消费？消息不会丢么？

作者回复: 一次取一批消息，等这一批消息都成功了，再提交最后一条消息的位置作为新的消费位置。如果其中任何一条失败，则认为整批都失败。

4

Get it
2020-04-15

老师，你好
想请问下在消费端 一般通过先处理业务逻辑再确认消息以保证消息不丢失
但是如果业务逻辑有未知风险会持续抛出异常导致消息一直无法消费导致积压一般如何解决呢

作者回复: 有些消息队列有针对这个问题的解决方案，比如RocketMQ有死信队列，对于多次消费的失败的消息，扔到死信队列中，避免坏消息卡住队列。

3

睡在床板下
2020-04-05

老师，像rabbitmq 消费无法扩充分区，那么只能扩充消费者吗?

作者回复: 一般来说，队列（分区）数量最好和消费者的实例数保持一致，这样能达到最佳的消费性能。

3