10 | 索引拆分：大规模检索系统如何使用分布式技术加速检索？

2020-04-17 陈东

检索技术核心20讲 进入课程 >

讲述：陈东
时长 14:30 大小 13.29M

你好，我是陈东。

在互联网行业中，分布式系统是一个非常重要的技术方向。我们熟悉的搜索引擎、广告引擎和推荐引擎，这些大规模的检索系统都采用了分布式技术。

分布式技术有什么优点呢？**分布式技术就是将大任务分解成多个子任务，使用多台服务器共同承担任务，让整体系统的服务能力相比于单机系统得到了大幅提升。**而且，在第 8 讲中我们就讲过，在索引构建的时候，我们可以使用分布式技术来提升索引构建的效率。

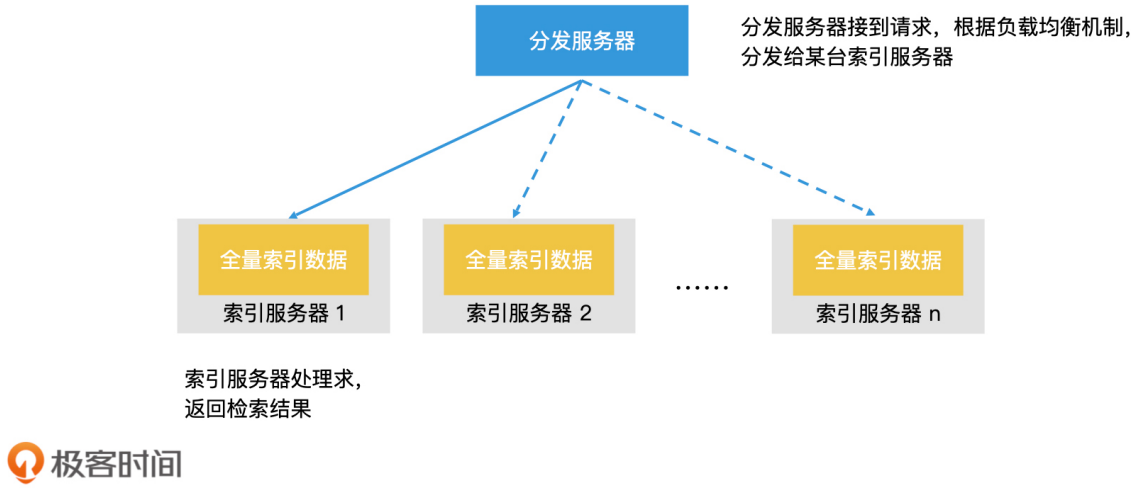
那今天，我们就来聊一聊，大规模检索系统中是如何使用分布式技术来加速检索的。

简单的分布式结构是什么样的?

一个完备的分布式系统会有复杂的服务管理机制，包括服务注册、服务发现、负载均衡、流量控制、远程调用和冗余备份等。在这里，我们先抛开分布式系统的实现细节，回归到它的本质，也就是从“让多台服务器共同承担任务”入手，来看一个简单的分布式检索系统是怎样工作的。

首先，我们需要一台接收请求的服务器，但是该服务器并不执行具体的查询工作，它只负责任务分发，我们把它叫作**分发服务器**。真正执行检索任务的是**多台索引服务器**，每台索引服务器上都保存着完整的倒排索引，它们都能完成检索的工作。

当分发服务器接到请求时，它会根据负载均衡机制，将当前查询请求发给某台较为空闲的索引服务器进行查询。具体的检索工作由该台索引服务器独立完成，并返回结果。



简单的分布式检索系统

现在，分布式检索系统的结构你已经知道了，那它的效率怎么样呢？举个例子，如果一台索引服务器一秒钟能处理 1000 条请求，那我们同时使用 10 台索引服务器，整个系统一秒钟就能处理 10000 条请求了。也就是说，这样简单的分布式系统，就能大幅提升整个检索系统的处理能力。

但是，这种简单的分布式系统有一个问题：它仅能提升检索系统整体的“吞吐量”，而不能缩短一个查询的检索时间。也就是说，如果单机处理一个查询请求的耗时是 1 秒钟，那不管我们增加了多少台机器，单次查询的检索时间依然是 1 秒钟。所以，如果我们想要缩短检索时间，这样的分布式系统是无法发挥作用的。

那么，我们能否利用多台机器，来提升单次检索的效率呢？我们先来回顾一下，在前面讨论工业级的倒排索引时我们说过，对于存储在磁盘上的大规模索引数据，我们要尽可能地将数据加载到内存中，以此来减少磁盘访问次数，从而提升检索效率。

根据这个思路，当多台服务器的总内存量远远大于单机的内存时，我们可以把倒排索引拆分开，分散加载到每台服务器的内存中。这样，我们就可以避免或者减少磁盘访问，从而提升单次检索的效率了。

即使原来的索引都能加载到内存中，索引拆分依然可以帮助我们提升单次检索的效率。这是因为，检索时间和数据规模是正相关的。当索引拆分以后，每台服务器上加载的数据都会比全量数据少，那每台服务器上的单次查询所消耗的时间也就随之减少了。

因此，索引拆分是检索加速的一个重要优化方案，至于索引应该如何拆分，以及拆分后该如何检索，工业界也有很多不同的实现方法。你可以先自己想一想，然后我们再一起来看看，工业界一般都是怎么做的。

如何进行业务拆分？

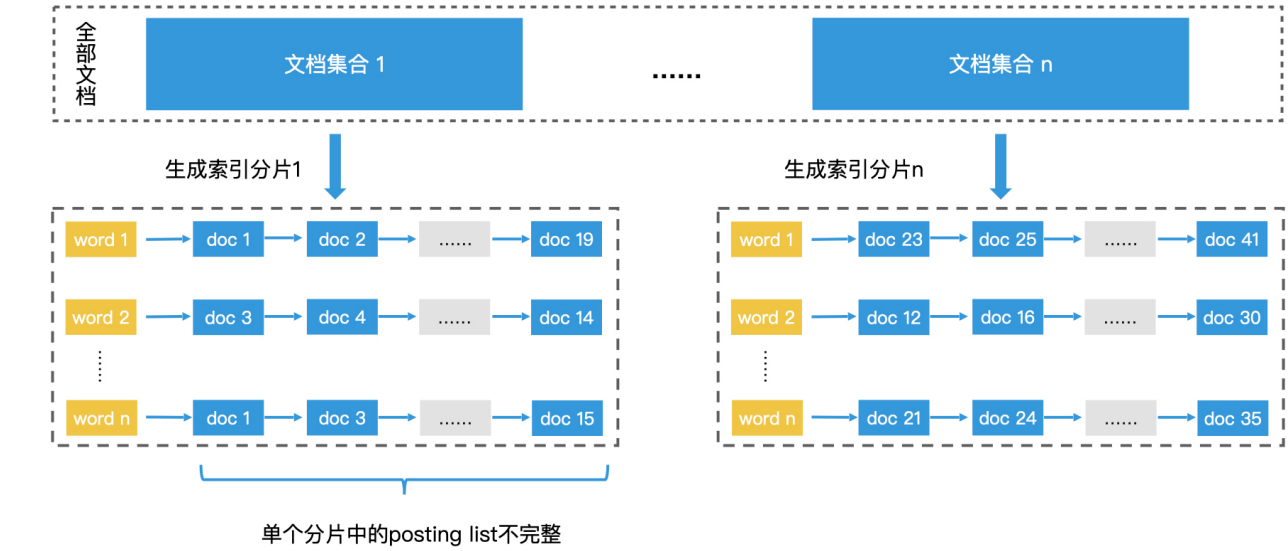
首先，在工业界中一个最直接的索引拆分思路，是根据业务进行索引拆分。那具体该如何拆分呢？

我来举个例子。在图书管理系统中，有许多不同国籍的作家的作品。如果我们将它们分成国内作品和国外作品两大类，分别建立两个倒排索引，这就完成了索引拆分。索引拆分之后，我们可以使用不同的服务器加载不同的索引。在检索的时候，我们需要先判断检索的是国内作品还是国外作品，然后在检索界面上做好选择，这样系统就可以只在一个索引上查询了。如果我们不能确认是哪类作品，那也没关系，系统可以在两个索引中并行查找，然后将结果汇总。

你会看到，基于业务的拆分是一个实用的索引拆分方案，在许多应用场景中都可以使用。但是这种方案和业务的耦合性太强，需要根据不同的业务需求灵活调整。那我们有没有更通用的技术解决方案呢？你可以先想一下，然后我们一起来讨论。

如何基于文档进行拆分？

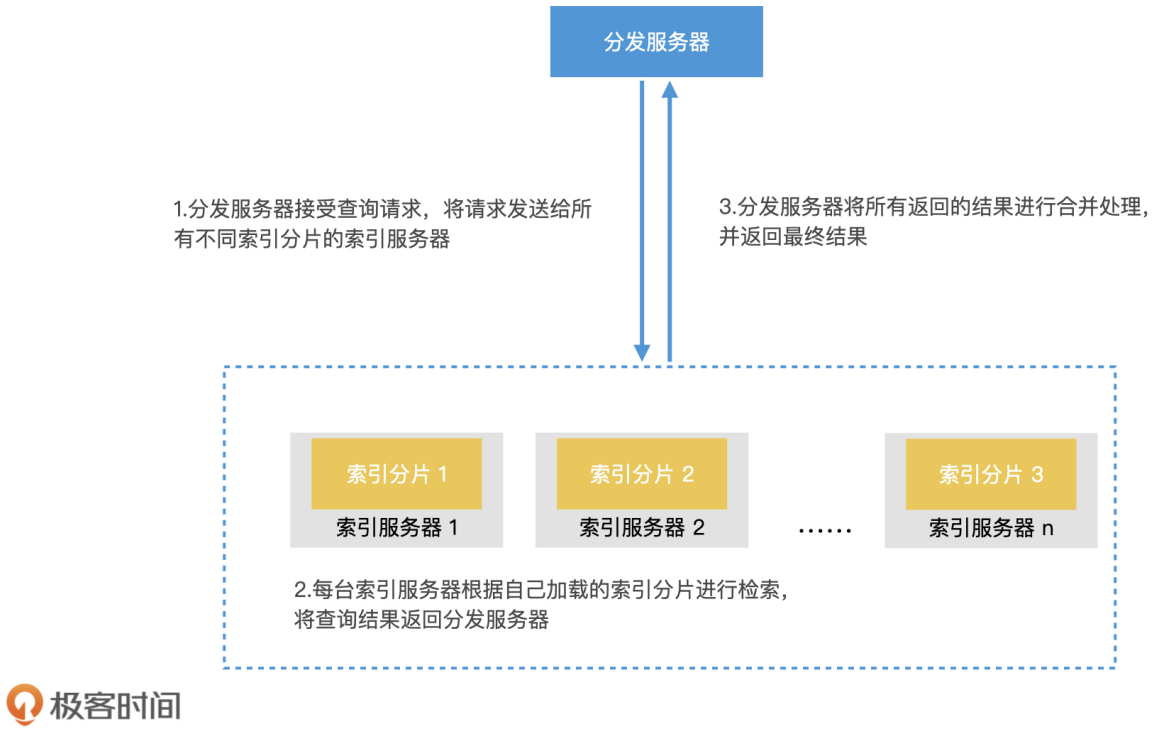
以搜索引擎为例，一个通用的方案是借鉴索引构建的拆分思路，将大规模文档集合随机划分为多个小规模的文档集合分别处理。这样我们就可以基于文档进行拆分，建立起多个倒排索引了。其中，每个倒排索引都是一个索引分片，它们分别由不同的索引服务器负责。每个索引分片只包含部分文档，所以它们的 posting list 都不会太长，这样单机的检索效率也就得到了提升。



基于文档拆分索引

但是，这样拆分出来的任意一个单独的索引分片，它检索出来的结果都不完整，我们还需要合并操作才能得到最后的检索结果。因此，对于基于文档进行拆分的分布式方案，我们的检索流程可以总结为 3 个步骤：

1. 分发服务器接受查询请求，将请求发送给所有不同索引分片的索引服务器；
2. 每台索引服务器根据自己加载的索引分片进行检索，将查询结果返回分发服务器；
3. 分发服务器将所有返回的结果进行合并处理，再返回最终结果。



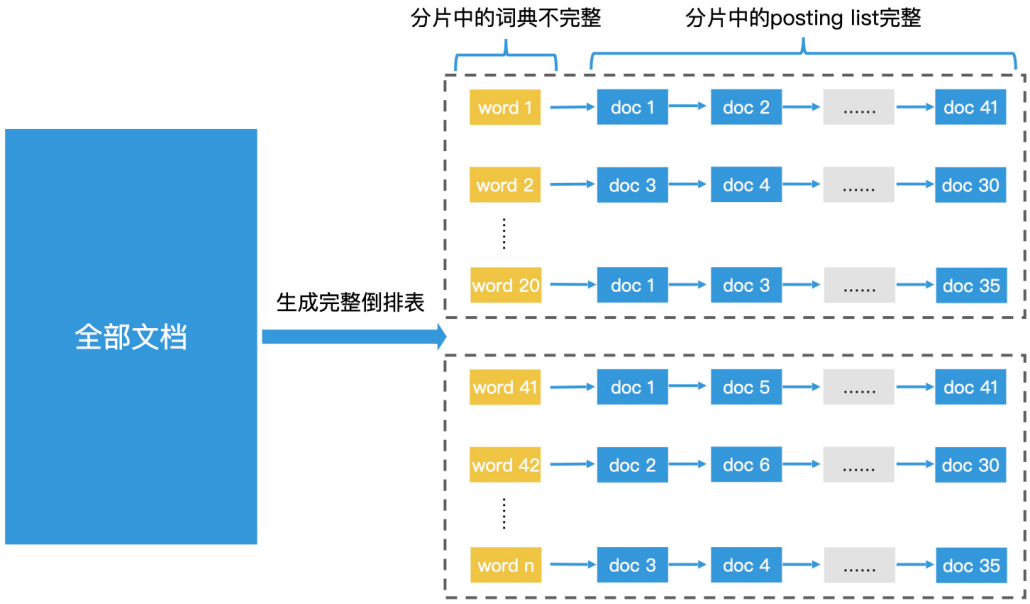
基于文档拆分索引的检索过程

这种基于文档拆分的方案是随机划分的，所以我们可以不用关心业务细节。而且每个索引分片的大小都能足够相近，因此，这种拆分方式能很均匀地划分检索空间和分担检索负载。并且，如果我们将索引数据分成合适的份数，是有可能将所有数据都加载到内存中的。由于每个索引分片中的文档列表都不长，因此每台机器对于单个请求都能在更短的时间内返回，从而加速了检索效率。

但是，分片的数量也不宜过多。这是因为，一个查询请求会被复制到所有的索引分片上，如果分片过多的话，每台加载索引分片的服务器都要返回 n 个检索结果，这会带来成倍的网络传输开销。而且，分片越多，分发服务器需要合并的工作量也会越大，这会使得分发服务器成为瓶颈，造成性能下降。因此，对于索引分片数量，我们需要考虑系统的实际情况进行合理的设置。

如何基于关键词进行拆分？

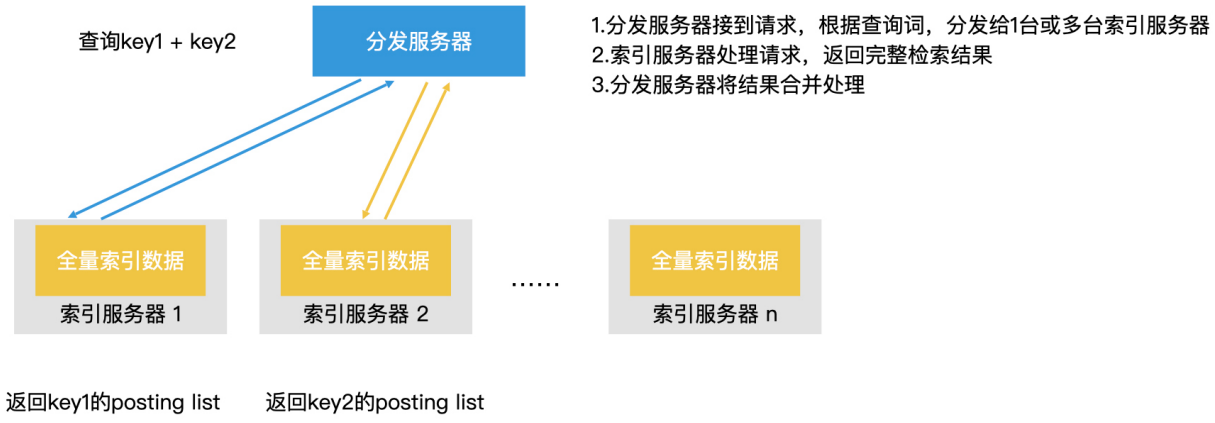
在搜索引擎中，为了解决分片过多导致一次请求被复制成多次的问题，我们还可以使用另一种拆分方案，那就是基于关键词进行拆分。这种方案将词典划分成多个分片，分别加载到不同的索引服务器上。每台索引服务器上的词典都是不完整的，但是词典中关键词对应的文档列表都是完整的。



极客时间

基于关键词拆分索引

当用户查询时，如果只有一个关键词，那我们只需要查询存有这个关键词的一台索引服务器，就能得到完整的文档列表，而不需要给所有的索引服务器都发送请求；当用户同时查询两个关键词时，如果这两个关键词也同时属于一个索引分片的话，那系统依然只需要查询一台索引服务器即可。如果分别属于两个分片，那我们就需要发起两次查询，再由分发服务器进行结果合并。



极客时间

基于关键词拆分索引的检索过程

也就是说，在查询词少的情况下，如果能合理分片，我们就可以大幅降低请求复制的代价了。

但是这种切分方案也带来了很多复杂的管理问题，比如，如果查询词很多并且没有被划分到同一个分片中，那么请求依然会被多次复制。再比如，以及如果有的关键词是高频词，那么对应的文档列表会非常长，检索性能也会急剧下降。此外，还有新增文档的索引修改问题，系统热点查询负载均衡的问题等。

因此，除了少数的高性能检索场景有需求以外，一般我们还是基于文档进行索引拆分。这样，系统的扩展性和可运维性都会更好。

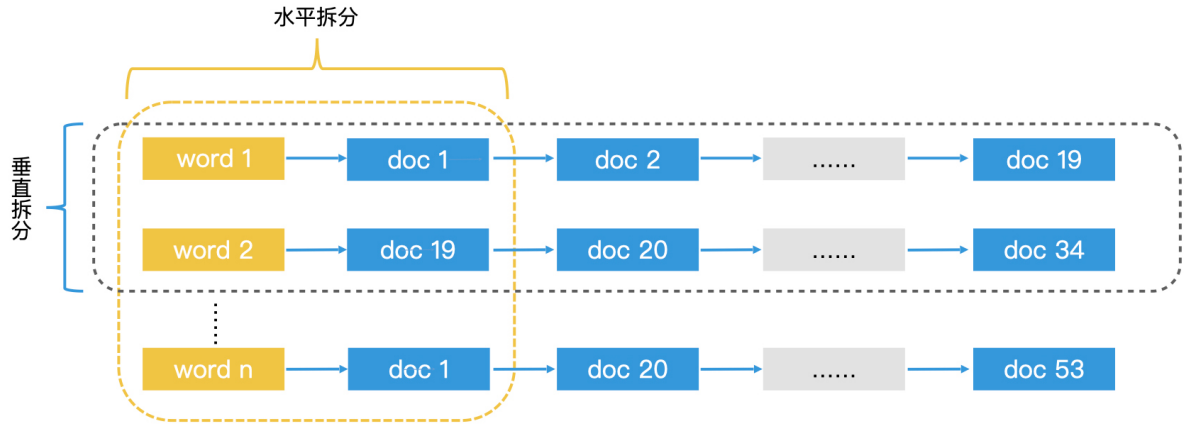
重点回顾

好了，今天的内容就先讲到这里。我们一起来总结一下，你要掌握的重点内容。

首先，利用分布式技术，我们可以将倒排索引进行索引拆分。索引拆分的好处是：一方面是能将更多的索引数据加载到内存中，降低磁盘访问次数，使得检索效率能得到大幅度的提升；另一方面是基于文档的拆分，能将一个查询请求复制成多份，由多台索引服务器并行完成，单次检索的时间也能得到缩短。

其次，除了搜索引擎，其他大规模数据检索引擎，如广告引擎、推荐引擎等也都使用了类似的索引拆分技术。只是由于它们处理的对象不是文档，因此对于拆分方式的命名也不同。

一般来说，根据处理对象将倒排索引进行拆分，每个索引分片都可能有完整的词典，但 posting list 不完整，这种拆分方案叫作**水平拆分**。如果是根据倒排索引中的关键词进行拆分，每个索引分片的词典都不完整，但是词典中的关键词对应的 posting list 是完整的，这种拆分方案叫作**垂直拆分**。



极客时间

水平拆分和垂直拆分

总之，**合理的索引拆分是分布式检索加速的重要手段，也是工业界的有效实践经验。**因此，我希望你能好好地理解今天的内容。

课堂讨论

为什么说基于文档拆分的方案会比基于关键词拆分的方案更好维护？你可以结合以下 2 个问题来考虑一下：

1. 当有新文档加入时，会影响多少台索引服务器？
2. 当某些关键词是热点，会被大量查询时，每台服务器的负载是否均衡？

欢迎在留言区畅所欲言，说出你的思考过程和最终答案。如果有收获，也欢迎把这篇文章分享给你的朋友。

上一篇　09 | 索引更新：刚发布的文章就能被搜到，这是怎么做到的?

下一篇　11 | 精准Top K检索：搜索结果是怎么进行打分排序的?

精选留言 (8)

写留言

一步
2020-04-17

Redis Cluster 技术相当于按照关键词进行拆分，直接定位到 要查询的 key 在哪个 slot

作者回复: 你举的这个例子很有意思。其实Redis cluster是一个kv存储，而不是一个倒排索引。对于kv，你既可以说它是按关键词拆分的，也可以说是按文档拆分的。

当然，如果kv中的v是一个结果集合列表的话，这就是一个典型的基于关键词拆分的倒排索引了。在我们不需要进行多个关键词合并的场景下，这样的使用方案是很适合的。

1　1

那时刻
2020-04-17

基于文档或关键字拆分，类似于数据库的分库分表操作。基于文档拆分的好处在于分摊网络和io的压力。

作者回复: 是的。你会看到，数据库的分库分表其实也是一样的思路。因此，也希望大家能将一个技术进行横行对比，这样能更好地融会贯通，举一反三。

1

峰
2020-04-17

本来按照题目写了下答案，但感觉还不如跳出来回答这个问题，看着太像传统数据库横向拆分纵向拆分，然后引入中间件，然后就有手工分库分表那一坨解决方案。跳出这个思路，横向纵向分片只是分散数据到各个节点的手段，上层应该提供策略屏蔽这些手段的差异，针对具体的分片方式做优化，比如热点，那就针对这个分片多点副本。

展开

作者回复: 是的。后面抽象成了水平拆分和垂直拆分，其实就和数据库的拆分理念很相似了。包括业务拆分，其实也和分库很相似。因此，许多设计理念都是可以相互借鉴，融会贯通的。

那时刻
2020-04-17

尝试回答老师在回复里的问题。不明白的地方请老师指正。
1.按文档拆分，新增加的文档可以只加到一台增量索引的机器上班，因为查询的时候有按照关键字的合并
2.我觉得可以按照业务拆分来减少查询的复制，比如按照文档类型 军事，娱乐来把文档分区，这样查询关键字的时候，比如这个关键字属于军事类型就只去军事类型文档分区找...

展开

作者回复: 1.没错。如果使用了全量索引+增量索引机制的话，对于新增文档，其实只需要先分片到对应的索引服务器上，然后加入这台服务器的增量索引即可。
2.业务拆分尽管和业务耦合紧密，不过它可以同时兼具文档拆分和关键词拆分的优点(也可以理解为，业务拆分可以在两个方向进行抽象，分别变成文档拆分和关键词拆分)。
业务既能对文档进行分片，也能在查询时指定只去一个分片查询，而不是所有分片都查询。因此在一些简单的应用场合中也是可以考虑的。
对应到数据库设计，就是分库的问题。

Mq
2020-04-17

基于文档的方案在有新文档加入时只会影响到有文档的那台服务器，基于关键词的拆分会影响到有关键词的所有服务器。
热点关键词问题我怎么觉得基于文档跟基于关键词的划分都有，并且基于文档划分的影响范围更大。因为基于文档的划分所有索引服务器都保存了完整的key，也就意味着热点key来了后会导致所有索引服务器负载高，基于热点key的划分还只会影响到热点key的那台...
展开

作者回复: 对于热点的问题，如果所有的服务器的负载都同时上升，其实这是我们期望的事情，这时候没有服务器是“热点”，我们在运维时只要无差别扩容就行。
相反，如果有的服务器查询负载很低，但有的服务器查询负载很高，那么这时候就存在“热点”问题了，我们需要针对特殊的一小撮服务器进行加副本扩容。但这时候可能其他服务器其实还是足够空闲的，这就造成了资源浪费。而且，如果第二天热点切换了的话，那我们是不是还要将原热点的副本下线，然后上线其他热点的副本?这样就会给运维带来很大的复杂性。

1

_你说了不算
2020-04-17

老师，我们的广告投放引擎在数据检索这块就走了不少路，es过滤和内存过滤两种方式都用过，最终还是用了内存过滤，原因是后者服务器的cpu和内存状态更好，不知道是不是我们用es的姿势不对。最近引擎系统cpu一直报警，除了加机器，就是加机器，想请教下老师，有遇到过类似的这种情况吗？假如按照文章中提到的索引拆分的方式，具体的落地方案老师能不能指点一二？还有我们的数据是AE通过管理后台存在mysql的，那么mysql和...
展开

作者回复: 1.关于es的使用，的确是要了解了相关检索技术，已经了解了es以后，才能发挥出es的优势。如果你们想走这条路线的话，那需要多花点时间深入了解。
2.你所谓的内存过滤，我的理解就是你们自研系统，在内存中建立索引处理。那么这样的话，你可以结合这个专栏的内容看看如何优化，比如说索引拆分，你们可以指定一个固定分片数，然后在离线环节就拆分好；然后结合全量索引+增量索引的机制，也能保证索引更新时的性能；还有倒排检索加速(参考两篇加餐)，应该也是有帮助的。
3.MySQL和es的数据同步问题，其实有许多工具可以做，比如说logstash等。而一致性问题，需要你进行监控和周期性检查，避免有错误。还可以进行周期性完整重建索引的方式，将之前可能已经造成的不一致进行修复。

pedro
2020-04-17

按照老师的思路大致可以理一下，当新的文档加入，势必会根据新文档的关键词再拆分，影响的服务器得根据具体场景来定，可能会影响多台服务器，而基于文档拆分会直接将新增文档加入到特定的服务器中，对于热点关键词，肯定会被频繁访问，一台服务器会不堪重负，而冷点关键词估计会在服务器中发霉，热点关键词可能需要多台服务器进行支持，并增加相应的网关进行负载均衡。...

展开

作者回复: 热点问题的确很常见。一个通用的解法就是为热点分片增加副本。
此外，对于随机划分的文档拆分方案而言，由于随机的特性，它出现热点分片的概率相对较低，而且即使出现了，它也可以通过再次随机划分，或者有规划地划分来完成检索负载均衡的问题。
而基于关键词划分的方案，因为还有一个“相关的关键词要被划分到同一个分片”的限制，因此在调整分片上能做的事情就很有限了。这也是它不如文档拆分更灵活的原因。


每天晒白牙
2020-04-17

提高的吞吐量而非检索效率
简单的分布式检索系统是指每台索引服务器保存了全量的索引数据，然后加机器，这种方式只能提高系统整体的"吞吐量"，而不能缩短检索时间从而加速检索效率

通过拆分提高检索效率...

展开

作者回复: 总结得很好。我觉得可以给你补两个问题。有精力的童鞋也都可以想想。
1.如果索引是使用“全量索引+增量索引”，再基于文档拆分，那么一个新文档加入时，它是会加入到所有索引服务器的增量索引中，还是可以只加入到一台服务器的增量索引中?
2.基于文档拆分，会造成查询请求被复制多份，那除了基于关键词拆分，我们使用业务拆分的方案是否也能避免这个问题?