40-Redis的下一步：基于NVM内存的实践

你好，我是蒋德钧。

今天这节课是咱们课程的最后一节课了，我们来聊聊Redis的下一步发展。

这几年呢，新型非易失存储（Non-Volatile Memory，NVM）器件发展得非常快。NVM器件具有容量大、性能快、能持久化保存数据的特性，这些刚好就是Redis追求的目标。同时，NVM器件像DRAM一样，可以让软件以字节粒度进行寻址访问，所以，在实际应用中，NVM可以作为内存来使用，我们称为NVM内存。

你肯定会想到，Redis作为内存键值数据库，如果能和NVM内存结合起来使用，就可以充分享受到这些特性。我认为，Redis发展的下一步，就可以基于NVM内存来实现大容量实例，或者是实现快速持久化数据和恢复。这节课，我就带你了解下这个新趋势。

接下来，我们先来学习下NVM内存的特性，以及软件使用NVM内存的两种模式。在不同的使用模式下，软件能用到的NVM特性是不一样的，所以，掌握这部分知识，可以帮助我们更好地根据业务需求选择适合的模式。

拼课微信：699250

NVM内存的特性与使用模式

Redis是基于DRAM内存的键值数据库，而跟传统的DRAM内存相比，NVM有三个显著的特点。

首先，**NVM内存最大的优势是可以直接持久化保存数据**。也就是说，数据保存在NVM内存上后，即使发生了宕机或是掉电，数据仍然存在NVM内存上。但如果数据是保存在DRAM上，那么，掉电后数据就会丢失。

其次，**NVM内存的访问速度接近DRAM的速度**。我实际测试过NVM内存的访问速度，结果显示，它的读延迟大约是200~300ns，而写延迟大约是100ns。在读写带宽方面，单根NVM内存条的写带宽大约是1~2GB/s，而读带宽约是5~6GB/s。当软件系统把数据保存在NVM内存上时，系统仍然可以快速地存取数据。

最后，**NVM内存的容量很大**。这是因为，NVM器件的密度大，单个NVM的存储单元可以保存更多数据。例如，单根NVM内存条就能达到128GB的容量，最大可以达到512GB，而单根DRAM内存条通常是16GB或32GB。所以，我们可以很轻松地用NVM内存构建TB级别的内存。

总结来说，NVM内存的特点可以用三句话概括：

- 能持久化保存数据；
- 读写速度和DRAM接近；
- 容量大。

现在，业界已经有了实际的NVM内存产品，就是Intel在2019年4月份时推出的Optane AEP内存条（简称AEP内存）。我们在应用AEP内存时，需要注意的是，AEP内存给软件提供了两种使用模式，分别对应着使用了NVM的容量大和持久化保存数据两个特性，我们来学习下这两种模式。

第一种是Memory模式。

这种模式是把NVM内存作为大容量内存来使用的，也就是说，只使用NVM容量大和性能高的特性，没有启用数据持久化的功能。

例如，我们可以在一台服务器上安装6根NVM内存条，每根512GB，这样我们就可以在单台服务器上获得3TB的内存容量了。

在Memory模式下，服务器上仍然需要配置DRAM内存，但是，DRAM内存是被CPU用作AEP内存的缓存，DRAM的空间对应用软件不可见。换句话说，**软件系统能使用到的内存空间，就是AEP内存条的空间容量**。

第二种是App Direct模式。

这种模式启用了NVM持久化数据的功能。在这种模式下，应用软件把数据写到AEP内存上时，数据就直接持久化保存下来了。所以，使用了App Direct模式的AEP内存，也叫做持久化内存（Persistent Memory，PM）。

现在呢，我们知道了AEP内存的两种使用模式，那Redis是怎么用的呢？我来给你具体解释一下。

基于NVM内存的Redis实践

当AEP内存使用Memory模式时，应用软件就可以利用它的大容量特性来保存大量数据，Redis也就可以给上层业务应用提供大容量的实例了。而且，在Memory模式下，Redis可以像在DRAM内存上运行一样，直接在AEP内存上运行，不用修改代码。

不过，有个地方需要注意下：在Memory模式下，AEP内存的访问延迟会比DRAM高一点。我刚刚提到过，NVM的读延迟大约是200~300ns，而写延迟大约是100ns。所以，在Memory模式下运行Redis实例，实例读性能会有所降低，我们就需要在保存大量数据和读性能较慢两者之间做个取舍。

那么，当我们使用App Direct模式，把AEP内存用作PM时，Redis又该如何利用PM快速持久化数据的特性呢？这就和Redis的数据可靠性保证需求和现有机制有关了，我们来具体分析下。

为了保证数据可靠性，Redis设计了RDB和AOF两种机制，把数据持久化保存到硬盘上。

但是，无论是RDB还是AOF，都需要把数据或命令操作以文件的形式写到硬盘上。对于RDB来说，虽然Redis实例可以通过子进程生成RDB文件，但是，实例主线程fork子进程时，仍然会阻塞主线程。而且，RDB文件的生成需要经过文件系统，文件本身会有一定的操作开销。

对于AOF日志来说，虽然Redis提供了always、everysec和no三个选项，其中，always选项以fsync的方式落盘保存数据，虽然保证了数据的可靠性，但是面临性能损失的风险。everysec选项避免了每个操作都要实时落盘，改为后台每秒定期落盘。在这种情况下，Redis的写性能得到了改善，但是，应用会面临秒级数据丢失的风险。

此外，当我们使用RDB文件或AOF文件对Redis进行恢复时，需要把RDB文件加载到内存中，或者是回放AOF中的日志操作。这个恢复过程的效率受到RDB文件大小和AOF文件中的日志操作多少的影响。

所以，在前面的课程里，我也经常提醒你，不要让单个Redis实例过大，否则会导致RDB文件过大。在主从集群应用中，过大的RDB文件就会导致低效的主从同步。

拼课微信：699250

我们先简单小结下现在Redis在涉及持久化操作时的问题：

- RDB文件创建时的fork操作会阻塞主线程；
- AOF文件记录日志时，需要在数据可靠性和写性能之间取得平衡；
- 使用RDB或AOF恢复数据时，恢复效率受RDB和AOF大小的限制。

但是，如果我们使用持久化内存，就可以充分利用PM快速持久化的特点，来避免RDB和AOF的操作。因为PM支持内存访问，而Redis的操作都是内存操作，那么，我们就可以把Redis直接运行在PM上。同时，数据本身就可以在PM上持久化保存了，我们就不再需要额外的RDB或AOF日志机制来保证数据可靠性了。

那么，当使用PM来支持Redis的持久化操作时，我们具体该如何实现呢？

我先介绍下PM的使用方法。

当服务器中部署了PM后，我们可以在操作系统的/dev目录下看到一个PM设备，如下所示：

```
/dev/pmem0
```

然后，我们需要使用ext4-dax文件系统来格式化这个设备：

```
mkfs.ext4 /dev/pmem0
```

接着，我们把这个格式化好的设备，挂载到服务器上的一个目录下：

```
mount -o dax /dev/pmem0  /mnt/pmem0
```

此时，我们就可以在这个目录下创建文件了。创建好了以后，再把这些文件通过内存映射（mmap）的方式映射到Redis的进程空间。这样一来，我们就可以把Redis接收到的数据直接保存到映射的内存空间上了，而这块内存空间是由PM提供的。所以，数据写入这块空间时，就可以直接被持久化保存了。

而且，如果要修改或删除数据，PM本身也支持以字节粒度进行数据访问，所以，Redis可以直接在PM上修改或删除数据。

如果发生了实例故障，Redis宕机了，因为数据本身已经持久化保存在PM上了，所以我们可以直接使用PM上的数据进行实例恢复，而不用再像现在的Redis那样，通过加载RDB文件或是重放AOF日志操作来恢复了，可以实现快速的故障恢复。

当然，因为PM的读写速度比DRAM慢，所以，**如果使用PM来运行Redis，需要评估下PM提供的访问延迟和**

拼课微信：699250

访问带宽，是否能满足业务层的需求。

我给你举个例子，带你看下如何评估PM带宽对Redis业务的支撑。

假设业务层需要支持1百万QPS，平均每个请求的大小是2KB，那么，就需要机器能支持2GB/s的带宽（1百万请求操作每秒 * 2KB每请求 = 2GB/s）。如果这些请求正好是写操作的话，那么，单根PM的写带宽可能不太够用了。

这个时候，我们就可以在一台服务器上使用多根PM内存条，来支撑高带宽的需求。当然，我们也可以使用切片集群，把数据分散保存到多个实例，分担访问压力。

好了，到这里，我们就掌握了用PM将Redis数据直接持久化保存在内存上的方法。现在，我们既可以在单个实例上使用大容量的PM保存更多的业务数据了，同时，也可以在实例故障后，直接使用PM上保存的数据进行故障恢复。

拼课微信：699250

小结

这节课我向你介绍了NVM的三大特点：性能高、容量大、数据可以持久化保存。软件系统可以像访问传统DRAM内存一样，访问NVM内存。目前，Intel已经推出了NVM内存产品Optane AEP。

这款NVM内存产品给软件提供了两种使用模式，分别是Memory模式和App Direct模式。在Memory模式时，Redis可以利用NVM容量大的特点，实现大容量实例，保存更多数据。在使用App Direct模式时，Redis可以直接在持久化内存上进行数据读写，在这种情况下，Redis不用再使用RDB或AOF文件了，数据在机器掉电后也不会丢失。而且，实例可以直接使用持久化内存上的数据进行恢复，恢复速度特别快。

NVM内存是近年来存储设备领域中一个非常大的变化，它既能持久化保存数据，还能像内存一样快速访问，这必然会给当前基于DRAM和硬盘的系统软件优化带来新的机遇。现在，很多互联网大厂已经开始使用NVM内存了，希望你能够关注这个重要趋势，为未来的发展做好准备。

每课一问

按照惯例，我给你提个小问题，你觉得有了持久化内存后，还需要Redis主从集群吗？

欢迎在留言区写下你的思考和答案，我们一起交流讨论。如果你觉得今天的内容对你有所帮助，也欢迎你分享给你的朋友或同事。

精选留言：

- Kaito 2020-11-25 00:10:53
  有了持久化内存，是否还需要 Redis 主从集群？

  肯定还是需要主从集群的。持久化内存只能解决存储容量和数据恢复问题，关注点在于单个实例。

  而 Redis 主从集群，既可以提升集群的访问性能，还能提高集群的可靠性。

  例如部署多个从节点，采用读写分离的方式，可以分担单个实例的请求压力，提升集群的访问性能。而且当主节点故障时，可以提升从节点为新的主节点，降低故障对应用的影响。

- 写点啥呢 2020-11-25 09:14:29
  请问蒋老师，Redis将PM用作内存模式的话，是否需要修改Redis代码。我理解内存模式是对程序透明的，虽然PM可以把数据持久化保存，但是如果Redis进程把它看做内存，如果希望进程启动能够自动回复，就会涉及到进程内存空间的恢复，OS里是没有这个功能的，是不是应该需要Redis来做个事情，才可以直接从PM保存的上一次数据中作为新进程的内存空间，而不再需要通过RDB或者AOF来做数据持久化？ [3赞]

- 大土豆 2020-11-25 12:06:40
  老师问下一个redis集群的问题，现在最多人用的3主3从的redis cluster方案，3个主节点同时也是哨兵实例吧？你怎么看这种集群部署方案 [1赞]

- Lemon 2020-11-25 10:23:04
  肯定还是需要的，两者是互补的。

  NVM 给了数据存储方面的新方案，但目前用作 PM 的读写速度比 DRAM 慢，不使用主从集群仍会有明显的访问瓶颈。【过大的实例在主从同步时会有影响（缓存、带宽）】

  而集群是为了高可用，分散了数据的访问和存储，便于拓展与维护。对于单实例而言，即便单实例恢复的再快，挂了对业务仍会有影响。

  感觉 NVM 内存用作 PM 有点像第 28 将的 Pika，如果把 SSD 换为 NVM ，岂不是都再内存中操作？是否可以解决 Pika 操作慢的缺点？ [1赞]

- yyl 2020-11-25 11:48:48
  问题：有了持久化内存，是否还需要 Redis 主从集群？
  解答：需要，主从集群解决的单点故障问题，而且还能起到一定的负载分担。而NVM解决的是数据丢失

- 尿布 2020-11-25 08:36:45
  pika

- 林林要加油鸭 2020-11-25 00:14:57
  沙发！

拼界微信：699250