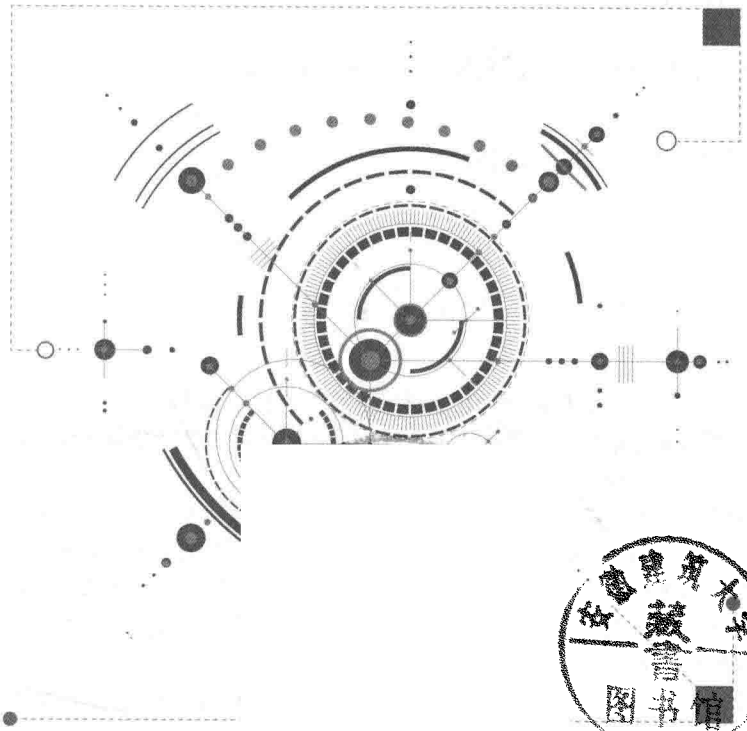
异步图书
www.epubit.com

分布式缓存

——原理、架构及Go语言实现

胡世杰 著

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS



分布式缓存

——原理、架构及Go语言实现

胡世杰 著

人民邮电出版社
北京

图书在版编目（CIP）数据

分布式缓存 ：原理、架构及Go语言实现 / 胡世杰著
. -- 北京 ：人民邮电出版社，2019.1
ISBN 978-7-115-49138-1

Ⅰ. ①分… Ⅱ. ①胡… Ⅲ. ①互联网络－网络服务器
Ⅳ. ①TP368.5

中国版本图书馆CIP数据核字(2018)第186771号

内 容 提 要

随着互联网的飞速发展，各行各业对互联网服务的要求也越来越高，互联网系统很多常见的存储类场景都面临着容量和稳定性风险。此时，本地缓存已无法满足需要，分布式缓存由于其高性能、高可用性等优点迅速被广大互联网公司接受并使用。

本书共有 3 个部分，每个部分都有 3 章。第 1 部分介绍基本功能的实现，主要介绍基于 HTTP 的 in memory 缓存服务、HTTP/REST 协议、TCP 等。第 2 部分介绍性能相关的内容，我们将集中全力讲解从各方面提升缓存服务性能的方法，主要包括 pipeline 的原理、RocksDB 批量写入等。最后一个部分则和分布式缓存服务集群有关，主要介绍分布式缓存集群、节点的再平衡功能等。本书选择用来实现分布式缓存的编程语言是当前流行的 Go 语言。

本书适合从事缓存方面工作的工程师或架构师，也适合想要学习和实现分布式缓存的读者。

◆ 著　　胡世杰
责任编辑　武晓燕
责任印制　焦志炜

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市君旺印务有限公司印刷

◆ 开本：800×1000　1/16
印张：12
字数：158 千字　　2019 年 1 月第 1 版
印数：1－2 400 册　　2019 年 1 月河北第 1 次印刷

定价：49.00 元

读者服务热线：(010)81055410　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字 20170147 号

作者简介

胡世杰，上海交通大学硕士，目前在七牛云任技术专家，是私有云存储服务的负责人。

他是分布式对象存储系统专家，在该领域拥有多年的架构、开发和部署经验，精通 C、C++、Perl、Python、Ruby、Go 等多种编程语言，熟悉 ElasticSearch、RabbitMQ 等各种开源软件。之前他还写过一本关于分布式对象存储的图书——《分布式对象存储——原理、架构及 Go 语言实现》。本书是他的第二部作品。

除了自己写作，他还致力于技术图书的翻译，是《JavaScript 面向对象精要》《Python 和 HDF5 大数据应用》《Python 高性能编程》等多部著作的译者。

序

写作本书的目的

随着互联网的飞速发展，各行各业对互联网服务的要求也越来越高，服务架构能撑起多大的业务数据？服务响应的速度能不能达到要求？我们的架构师每天都在思考这些问题。

对于数据库或者对象存储等服务来说，它们受限于自己先天的设计目标，往往不能具有很好的性能，响应时间通常是秒级。此时就需要高性能的缓存来为我们的服务提速了，缓存服务的响应时间通常是毫秒级，甚至小于 1ms。缓存服务需要被设置在其他服务的前端，客户端首先访问缓存，查询自己的数据，仅当客户端需要的数据不存在于缓存中时，才去访问实际的服务。从实际的服务中获取到的数据会被放在缓存中，以备下次使用。

缓存的设计目标就是尽可能地快，但它引起了其他的问题。比如目前业界使用较多的缓存服务有 Memcached 和 Redis 等，它们都是内存内缓存（in memory cache），单节点最大的容量不能超过整个系统的内存。且一旦服务器重启，对于 Memcached 来说就是内容彻底丢失；Redis 稍好一点，但也要花费不少时间从磁盘上的数据文件中重新读入内存。

本书将要实现的分布式缓存，利用了 Facebook 的 RocksDB 库，在普通的读写性能上不弱于 Redis，而且它能够突破内存容量的限制，非常快速地重启节点。

本书用来实现分布式缓存服务的编程语言是 Go 语言。

缓存和存储的关系

缓存和存储虽然都可以用来保存数据，但是它们的设计目标从一开始就截然不同。存储（storage）是非易失的，被存储的内容通常都会被期望永久保存，直到用户主动删除。存储对性能也有一定的要求，比如要求整个系统的吞吐量必须达到每秒多少字节等。但是在单个请求的响应时间上，存储就不会有很高的要求，因为用户也明白存储数据的重要性，愿意为了确保数据的安全支付较多的时间开销。

而缓存（cache）则被用来提升访问资源的速度。在设计的一开始，我们就明白缓存数据是允许丢失的，虽然在本书的实现中，缓存也会被保存在磁盘上，但是和存储的各种副本备份等防丢失措施相比就差得太远了，更不用说有些时候我们还会特意要求缓存有一个生存时间，将超出生存时间的缓存项目淘汰，以回收空间给新的项目。

缓存的目的就是要快速存取，所以本书会花大量的篇幅来介绍各种增强读写性能的措施，并且会将其与目前业界公认的高速缓存 Redis 进行对比。对比的指标主要包括请求的平均响应时间和每秒处理请求的数量。

分布式缓存集群的好处

用集群来提供服务的许多优点是单节点的服务无法相比的。

首先，单节点的扩展性不好，我们知道网络吞吐量和缓存容量会受到硬件的限制。对于单节点来说，这个上限就是主机硬件接口的数量；对于集群来说，它可以提供的硬件数量不受单块主板插槽数量的限制，只需要增加新的节点就可以了。

其次，单节点的性价比很低，一台高端设备能提供的服务往往弱于同样价格、由多台低端设备组建的集群能提供的服务。

最后，集群的容错率高于单节点的。一台服务器死机对于一个有 10 台服务器的集群来说损失了 10%的处理能力，但是对于单节点服务来说就是损失了 100%的能力。

本书的目标读者

如果你是互联网服务的工程师或架构师，正在寻找一个更适合你业务场景的缓存实现，本书可以给你一些启发。

如果你对缓存技术感兴趣，那么在读完本书之后你能够学到很多相关的知识。

在协议方面，本书主要涉及 HTTP/REST 和 TCP 两种协议，如果你对这些协议一无所知，那完全读懂本书可能会比较困难。

本书会介绍 Go 语言的一些基础，但更多时候则是专注于 Go 语言高阶技巧的应用。所以本书的读者需要对 Go 语言有一些基本的了解。

内容简介

本书共分为 3 个部分，每个部分都有 3 章。第 1 部分为基本功能的实现。

在第 1 章中，我们会实现一个基于 HTTP 的 in memory 缓存服务，满足缓存的 Set、Get、Del 等基本操作，然后介绍性能测试的工具 cache-benchmark，并将其与 Redis 进行对比。

Go 语言的 HTTP 服务框架虽然便利，但也会给我们的服务带来性能问题。我们会在第 2 章中将缓存服务改为使用 HTTP/REST 协议和 TCP 协议的混合接口，HTTP/REST 协议用于各种管理功能，TCP 协议则用于高速访问缓存。

我们会在第 3 章介绍 RocksDB，并借助 RocksDB 来实现缓存数据的持久化，同时也让我们的缓存容量得以突破内存的限制。

本书的第 2 部分为性能相关，我们将通过多种方法从各方面提升缓存服务的性能。

第 4 章将介绍 pipeline 的原理，在不改变任何服务端实现的情况下，仅通过改变客户端的行为，让缓存的读写效率得到提升。

在第 5 章中，我们利用 RocksDB 批量写入的功能来提升缓存的写入性能。

在第 6 章中，我们会用异步操作的方式来提升缓存的读取性能。

本书的最后一个部分和分布式缓存服务集群有关。

我们会在第 7 章中实现分布式缓存集群，利用一致性散列来实现节点的分片，并利用 gossip 协议来实现节点之间的对话。

当整个集群的容量渐满时，我们需要添加新的节点，这些新的节点加入集群后会导致新旧节点之间容量的不平衡，所以我们会在第 8 章讲解如何实现节点的再平衡。

然而并不是所有的缓存实现都需要节点再平衡。在第 9 章中，我们会介绍缓存

的生存时间和超时机制，实现 FIFO 的缓存淘汰策略。有了缓存淘汰策略，我们就可以不需要节点再平衡了。

如何下载和运行本书中的代码

本书的代码使用 Go 语言实现，Go 编译器的版本是 1.9.2，开发和运行环境是 Ubuntu 16.04。

本书中所有的 Go 语言代码都可以在 GitHub 和异步社区上找到，在 Linux 环境中可以用 git 命令下载：

```
git clone https://github.com/stuarthu/go-implement-your-cache-server.git
```

GitHub 是一个在线的软件项目管理仓库，Ubuntu 下的 Git 客户端可以用 apt-get 下载安装：

```
sudo apt-get install git
```

编译 Go 代码需要运行 Go 编译器，读者可以在 Go 语言官网下载最新的 Go 编译器。

本书代码下载完成后还需要将 GOPATH 环境变量设置为本书源码根目录。这样，Go 编译器才能知道 Go 包的搜索路径。设置完 GOPATH 之后，我们还需要用 go get 命令下载 3 个第三方 Go 包，它们会被本书代码使用：

```
go get github.com/go-redis/redis;
go get github.com/hashicorp/memberlist;
go get stathat.com/c/consistent。
```

Redis 是一个 Go 语言的 Redis 客户端，cache-benchmark 工具用到了它。cache-benchmark 是缓存服务的性能测试工具，我们会在本书的第 4 章中详细介绍它

的实现。

memberlist 是一个基于 gossip 协议的集群节点通信库，可以管理集群成员以及失败检测。consistent 是一个一致性散列的 Go 实现。我们会在第 7 章中实现分布式缓存集群时用到这两个 Go 包。

致谢

感谢我的妻子黄静和岳父黄雪春在我写书的日子里对我的支持，让我能够没有后顾之忧地写作。感谢人民邮电出版社的武晓燕编辑在本书写作和出版过程中的大力协助。

资源与支持

本书由异步社区出品，社区（https://www.epubit.com/）为您提供相关资源和后续服务。

配套资源

本书提供如下资源：

- 本书源代码。

要获得以上配套资源，请在异步社区本书页面中点击 配套资源，跳转到下载界面，按提示进行操作即可。注意：为保证购书读者的权益，该操作会给出相关提示，要求输入提取码进行验证。

提交勘误

作者和编辑尽最大努力来确保书中内容的准确性，但难免会存在疏漏。欢迎您将发现的问题反馈给我们，帮助我们提升图书的质量。

当您发现错误时，请登录异步社区，按书名搜索，进入本书页面，点击“提交勘误”，输入勘误信息，点击“提交”按钮即可。本书的作者和编辑会对您提交的勘误进行审核，确认并接受后，您将获赠异步社区的 100 积分。积分可用于在异步社区兑换优惠券、样书或奖品。

扫码关注本书

扫描下方二维码，您将会在异步社区微信服务号中看到本书信息及相关的服务提示。

与我们联系

我们的联系邮箱是 contact@epubit.com.cn。

如果您对本书有任何疑问或建议，请您发邮件给我们，并请在邮件标题中注明本书书名，以便我们更高效地做出反馈。

如果您有兴趣出版图书、录制教学视频，或者参与图书翻译、技术审校等工作，可以发邮件给我们；有意出版图书的作者也可以到异步社区在线提交投稿（直接访问 www.epubit.com/selfpublish/submission 即可）。

如果您是学校、培训机构或企业，想批量购买本书或异步社区出版的其他图书，也可以发邮件给我们。

如果您在网上发现有针对异步社区出品图书的各种形式的盗版行为，包括对图书全部或部分内容的非授权传播，请您将怀疑有侵权行为的链接发邮件给我们。您的这一举动是对作者权益的保护，也是我们持续为您提供有价值的内容的动力之源。

关于异步社区和异步图书

“异步社区”是人民邮电出版社旗下 IT 专业图书社区，致力于出版精品 IT 技术图书和相关学习产品，为作译者提供优质出版服务。异步社区创办于 2015 年 8 月，提供大量精品 IT 技术图书和电子书，以及高品质技术文章和视频课程。更多详情请访问异步社区官网 https://www.epubit.com。

“异步图书”是由异步社区编辑团队策划出版的精品 IT 专业图书的品牌，依托于人民邮电出版社近 30 年的计算机图书出版积累和专业编辑团队，相关图书在封面上印有异步图书的 LOGO。异步图书的出版领域包括软件开发、大数据、AI、测试、前端、网络技术等。

异步社区

微信服务号

目录

第 1 部分　基本功能

第 2 部分 性能相关

第 3 部分 服务集群