**安徽大学**

**本科毕业论文（设计、创作）**

**题　　目：　　基于负载特征的Redis数据库性能预测**

**学生姓名：**　 　周元核 **学号：** E31714013

**院（系）： 计算机科学与技术学院 专业：　　网络工程**

**入学时间：　 2017　　　　年　　7　　　月**

**导师姓名：　　窦辉 　　 职称/学位：　副教授**

**导师所在单位：　计算机科学与技术学院**

**完成时间：　 2021　　　　年　　　5　　　　月**

**基于负载特征的Redis数据库性能预测**

**摘 要**

Redis是互联网发展的产物，作为Key—value的高效存储系统，为互联网应用提供高效的解决方案。

Redis的应用非常广泛，根据它的特性，可以想象许多的应用场景，Redis数据库的性能（如延时、吞吐量）对这些大数据应用非常关键。在同样的硬件资源配置和软件配置参数下，Redis数据库的性能会受到其面临的负载特征的直接影响。例如，当写入数据总量大于Redis的最大内存空间时会触发写磁盘操作，进而影响到后续的读写数据性能。因此研究面向负载特征的Redis性能预测很有意义。

本文为了研究面向负载特征的Redis性能预测，通过模拟不同的读写请求负载。修改数据量负载和访问数据类型负载，收集在不同的负载下Redis的执行能力和延迟，获得性能数据，为Redis的运行时性能优化提供理论依据。Redis的进行性能预测可以对程序的负载能力提供更加合理准确的依据，而符合Redis性能预测的服务器可以被认为是合理合适的比对等机更快、更适合于特定的用户交互。Redis其支持的多种类型的数据结构意味着源数据有类型多样的存储需求，本次选题中，目的在于探究在不同负载情况下，Redis服务器的数据处理和执行能力会受到怎样的影响。

关键词：Redis；负载特征；YCSB；性能预测

**Performance prediction of redis database based on load characteristics**

**Abstract**

Redis is the product of the development of Internet. As an efficient Key value storage system, Redis provides efficient solutions for Internet applications.

Redis is widely used. According to its characteristics, we can imagine many application scenarios. The performance of Redis database (such as latency and throughput) is very critical to these big data applications. Under the same hardware resource configuration and software configuration parameters, the performance of Redis database will be directly affected by the load characteristics it faces. For example, when the total amount of data written is greater than the maximum memory space of Redis, the write to disk operation will be triggered, which will affect the subsequent data reading and writing performance. Therefore,it is significant to study the performance prediction of Redis based on load characteristics.

In order to study the performance prediction of Redis based on load characteristics, this paper simulates different read-write request loads. Modify the data load and access data type load, collect the execution ability and delay of Redis under different loads, and obtain the performance data, which provides a theoretical basis for the runtime performance optimization of Redis. The performance prediction of Redis can provide a more reasonable and accurate basis for the load capacity of the program, and the server that meets the performance prediction of Redis can be considered as reasonable and appropriate, faster than peer and more suitable for specific user interaction. Redis supports a variety of data structures, which means that the source data has a variety of storage requirements. The purpose of this topic is to explore how the data processing and execution ability of Redis server will be affected under different load conditions.

**Keywords: Redis， load characteristics， YCSB， performance prediction**

目 录

[abstract 3](#_Toc71189072)

[目 录 4](#_Toc71189073)

[2 研究背景与相关工作 5](#_Toc71189074)

[2.1 非关系数据库（NoSQL） 5](#_Toc71189075)

[2.2 Redis 应用场景 5](#_Toc71189076)

[2.3 Redis 的技术特点 5](#_Toc71189077)

[2.4 YCSB 的技术特点 6](#_Toc71189078)

[2.4.1 简化参数调节 6](#_Toc71189079)

[2.4.2 典型的负载类型 6](#_Toc71189080)

[3 基于负载特征的性能测试流程 7](#_Toc71189081)

[3.1 硬件配置 7](#_Toc71189082)

[3.2 软件配置 7](#_Toc71189083)

[3.2.1 Redis配置 7](#_Toc71189084)

[3.2.2 配置YCSB 8](#_Toc71189085)

[3.3 负载类型选择 8](#_Toc71189086)

[3.4 数据收集 8](#_Toc71189087)

[3.4.1 单个测试流程 8](#_Toc71189088)

[3.4.2 测试脚本编写与数据收集 10](#_Toc71189089)

[4 建模与验证 11](#_Toc71189090)

[4.1 数据预处理与可视化 11](#_Toc71189091)

[4.2 数据预测模型建立 12](#_Toc71189092)

[4.3 数据预测模型测试 17](#_Toc71189093)

[4 总结与体会 17](#_Toc71189094)

[参考文献 18](#_Toc71189095)

[致谢 18](#_Toc71189096)

# 引言

* 1. Redis应用广泛

Redis是互联网发展的产物，作为Key—value的高效存储系统，为互联网应用提供高效的解决方案。Redis基于c语言开发，支持网络、可基于内存进行短期缓存，亦可进行持久化的日志型存储。

Redis的应用目前已经十分广泛，国内的淘宝，新浪，以及国外的Git Hub等都在使用Redis的缓存服务。常见的应用场景有：缓存使用，网站并发量大的情况下，很多人使用了这套解决方案，而且逐渐有取代Memcached，成为首选服务端缓存的组件。包括持久型数据、 状态数据的缓存；排队使用，许多网站有秒杀、抢购的功能，这种功能即时性非常强，需要用高效的机制，发布库存的时候，可以把库存信息存在Redis ，这样用户对Redis 服务器压力就减少，这样就不会造成短时间的阻塞；异步通信，部分服务作为生成者，另一部分则作为消费者，可以代替传统的webservice 或者http ，高效的作为服务之间的通信；[1]日志记录，Redis 与 Logstash 一起，处理了分布式日志系统问题。Logstash 的Agent 收集日志发送到Redis ，另外一台中央Logstash 从Redis 读取日志，集中处理。

* 1. Redis存储机制

相比于传统的连接型数据库，其存数据存放在内存中而不是硬盘中，读写、查询、修改、删除等操作都是直接作用于内存，因此Redis和传统数据库速度差距就好比硬盘和内存的读写速度差距。[2]

此外Redis可以通过线程优化数据资源调用，Redis将数据缓存到内存时会，为了保证执行效率，会将其分片处理。因此Redis作为缓存使用时候也具有优秀的读写效率，其高速读写的特性来自于数据存储在内存中[3]，但是同时也容易受到内存或CPU读写能力限制。

* 1. 研究内容

在使用redis缓存技术之前，必须构筑分散型数据存储系统。为了进一步提高redis分散型数据存储的效率，还必须考虑数据库的负荷，所以在开发过程中需要构筑另一分散型存储系统。以往的相关数据库使用CPU或存储器来评估成本，但系统的存储方式和数据类型不太全面。在实际执行过程中，实时统计了多源数据。处理任务的线索完成后，将完成对未服务的主线程的响应。基于此，确定数据存储的有效性的影响因素，实现分散型数据的可靠性评价和存储的扩展性。

因此本文通过基于负载特征对Redis性能测试实验；对在不同负载特性下Redis服务器吞吐量和时延等性能指标信息，选择朴素贝叶斯多项式模型，对不同负载情况下数据库性能进行建模。经过验证为数据库进行性能评估。

# 2 研究背景与相关工作

当前各大互联网公司在提供web服务时，有众多基于Redis的数据存储服务器，Redis支持的多种类型的数据结构刚好满足了源数据类型多样的存储需求，分析Redis服务器的负载能力，为Redis的运行时性能优化提供依据，对数据库有着重要意义。

## 2.1 非关系数据库（NoSQL）

NoSQL是一种区别于关系型数据库，为了适应互联网进程而提出的全新的数据库类型，其跳出传统的行列关系，采用哈希表进行存储和查询，虽然服务器部署并不像传统数据库一样简单快捷，但是在面向局部数据查询修改时，通常其效率比传统关系型数据库有显著的提高。NoSQL数据库种类繁多，但是一个共同的特点都是去掉关系数据库的关系型特性。数据之间无关系，使得数据库普遍性上来说容易扩展。同时在也在架构的层面上带来了可扩展的能力。此外实际已经证实，在大数据量、高操作量的情况下，NoSQL数据库都具有非常优秀的读写性能，尤其在大数据量下，相对传统连接型数据库来说表现更加优秀。这得益于它的无关系性，数据库的结构简单。

## 2.3 Redis 的技术特点

对于不同的常用数据类型，Redis有相同的指令和针对不同数据类型的特殊指令，同时Redis采用和Java相同思想的多线程技术，这是Redis的主要特点之一，多线程处理，每线程优化系统资源，基于线程存储查找修改删除等方式操作Redis数据[4]。根据实际情况恢复线程，大大提高Redis的使用效率，加快数据采集速度，Redis的最大优点是可以将数据缓存在内存中，并将数据存储在分区中，具有较高的读写效率。

（1） 高性能。由于Redis不是SQL数据库，Redis的所有操作都直接在内存中操作。与MySQL 等传统关系数据库相比，它的运行速度与内存读写与硬盘读写的差距相当；

（2） Redis支持hash、set、string、list等数据类型的二进制操作；

（3） 支持事务，操作都是原子化的，所谓原子化就是改变数据或全部实现，或全部不执行；

（4） Redis还支持发布/订阅、通知、密钥过期等功能。基于以上优势，缓存采用Redis技术最大的应用场景。缓存技术可以大大减轻服务器的压力，减少数据丢失，提高系统中数据的一致性和完整性。[5]

它通常被称为数据结构服务器，因为值（value）可以是 字符串(String)， 哈希(Map)， 列表(list)， 集合(sets) 和 有序集合(sorted sets)等类型。并且提供了基本的数据结构list、set、map，还支持java、C++ 、.net等众多语言调用。Redis存储的数据一般放在内存当中，这样加快了读取速度。通常局限点来说，Redis也以消息队列的形式存在，作为内嵌的List存在，满足实时的高并发需求。而通常在一个电商类型的数据处理过程之中，有关商品，热销，推荐排序的队列，通常存放在Redis之中，期间也包扩Storm对于Redis列表的读取和更新[6]。

Redis的优点也是其缺点，由于Redis数据存储在内存中，数据库容量受到物理内存的限制，在用作海量数据的高性能读写时会遇到较大的限制。 因此，研究在物理内存确定的情况下，Redis数据库读写能力随负载类型的变化，对Redis的优性能化很有意义。

## 2.4 YCSB 的技术特点

YCSB英文全称为Yahoo! Cloud Serving Benchmark (YCSB) 。是Yahoo公司的一个用来对云服务数据库进行基础测试的工具。

## 2.4.1 简化参数调节

调节配置数据库参数在数据库测试系统中至关重要，但是在数据测试工具中即使是实现相同的功能，可调节参数也千差万别，例如，为了实现找到决定系统运行时间的功能，PostgreSQL需要调节大约 170 个参数，而 Apache Cassandra 则有155个参数可调节。[7]

在参数量较大的情况下，手动找到在实验中相对重要的参数是非常困难的事情，因此在部分数据库测试软件中配置了自动调节功能。[3,4]主要可分为：基于搜索的方法和基于学习的方法两种，在这两种情况下，一个从一组参数和一系列有效值开始，然后调谐系统尝试为每个参数找到最佳值，以最大化或最小化指定目标（例如吞吐量或延迟）。调谐所需的时间取决于需要调整的参数。随着参数数量的增加，基于搜索的方法的搜索空间大大扩展，因此查找所有参数的最佳值的速度变慢。基于 ML 的方法需要一个较大的培训数据集，因为本次实验中选用了采用ML方法的YCSB 。

## 2.4.2 典型的负载类型

YCSB将负载类型分为六类。

YCSB-A：模拟重写型负载。对数据库的操作中读取和写入操作各占50%，操作的记录的分布为Zipfian分布。

YCSB-B：数据库的操作中读操作占95%，写操作占5%，操作的记录的分布为Zipfian分布以，是读为主要工作负载测试的参数设置。

YCSB-C：所有操作均为读操作，操作的记录的分布为Zipfian分布，是以读为全部工作负载测试的参数设置。

YCSB-D：读操作占95%，插入操作占5% ，操作的记录的分布为最近分布。

YCSB-E：查找操作占95%，插入操作占5%，操作的记录的分布为Zipfian分布，且限制单次最大查找长度为100，是针对短区间生成的测试工作负载。

YCSB-F：读取操作占50%，读取后修改并写回的操作占50%。是修改写回生成的测试工作负载。[8]

# 3 基于负载特征的性能测试流程

## 3.1 硬件配置

实验物理环境为学生个人桌面电脑，处理器为AMD R5-2600，CPU主频为3.85GHz, 主机内存为32G，内存频率为2400MHz. 物理环境为实验预留了较为充足的空间，不会出现限制服务器的瓶颈，因测得的数据更加有说服力。

从安装<https://www.vmware.com/>下载并安装VMware Station，创建一个系统为ubuntu18.04虚拟机作为服务器，再创建一个系统同为ubuntu18.04的虚拟机作为客户端，在VMware Station设置配置网络互通，进行数据测试。

实验物理环境为：

表格 1 实验物理机器环境

|  |  |  |
| --- | --- | --- |
| 组件 | 型号 | 主频/内存 |
| CPU | R5-2600 | 3.85GHz |
| 内存 | 海盗船 | 32G |
| 虚拟服务器和客户机系统 | ubuntu-18.04 | - |
| 服务器内存分配 | - | 4G |
| 客户端内存分配 | - | 4G |

## 3.2 软件配置

### 3.2.1 Redis配置

在官网https://Redis.io/下载Redis（版本号为5.0.1）至虚拟机一，解压缩。在文件目录下，执行Redis-server开启服务器，生成配置文件。在文件安装目录下打开配置文件Redis.conf，注释“ bind 127.0.0.1”并设置密码，重新启动服务器，开放访问来源ip，方便进行下一步测试。

关闭Redis内置的读写保护程序，执行config set stop – writes – on – bgsave - error no。其中的“stop-writes-on-bgsave-error”的意思是在服务器被访问创建一个新的名为dump rdb的后台进程，若此进程创建时出现内存充满的情况，则会认为服务器被恶意数据攻击，停止访问，no则表示关闭该功能，由于本次实验进行基于负载特征的数据库运行能力预测。在测试过程中必然会遇到内存充满的情况，因此需要关闭该功能。

### 3.2.2 配置YCSB

从官网<https://github.com/brianfrankcooper/YCSB/releases/tag/0.10.0下载YCSB>数据包（版本号0.15.0），并编译成可执行脚本文件。将文件导入到客户端中，为了能正确识别服务器的Redis数据库，需要在在工作负载配置中，添加Redis的服务器地址和端口号，YCSB的数据库配置和负载类型在同位置，在安装目录下找到对应的工作负载workload，

主要参数说明：

表格 2 YCSB测试主要参数

|  |  |
| --- | --- |
| recordcount | load和run操作中，使用的YCSB实例记录数 |
| operationcount | load和run操作中，使用的YCSB实例操作数 |
| readproportion | 进行read的操作占所有操作的比例 |
| updateproportion | 进行update的操作占所有操作的比例 |
| insertproportion | 进行insert的操作占所有操作的比例 |
| scanproportion | 进行scan的操作占所有操作的比例 |
| requestdistribution | 使用什么分布来选择要操作的记录 |

## 3.3 负载类型选择

本次实验中，使用的负载类型为：

表格 3 本次实验选用的负载类型

|  |  |
| --- | --- |
| YCSB-A | 模拟重写型负载。对数据库的操作中读取和写入操作各占50%，操作的记录的分布为Zipfian分布。 |
| YCSB-B | 数据库的操作中读操作占95%，写操作占5%，操作的记录的分布为Zipfian分布以，是读为主要工作负载测试的参数设置。 |
| YCSB-C | 所有操作均为读操作，操作的记录的分布为Zipfian分布，是以读为全部工作负载测试的参数设置。 |
| YCSB-D | 读操作占95%，插入操作占5% ，操作的记录的分布为最近分布。 |
| YCSB-E | 查找操作占95%，插入操作占5%，操作的记录的分布为Zipfian分布，且限制单次最大查找长度为100，是针对短区间生成的测试工作负载。 |

## 3.4 数据收集

## 3.4.1 单个测试流程

为了得到负载和数据库性能的指标，需要修改负载类型、数据量和操作数，进行测试得到数据模型。本次测试需要获得的数据有：吞吐量、单指令平均时延。

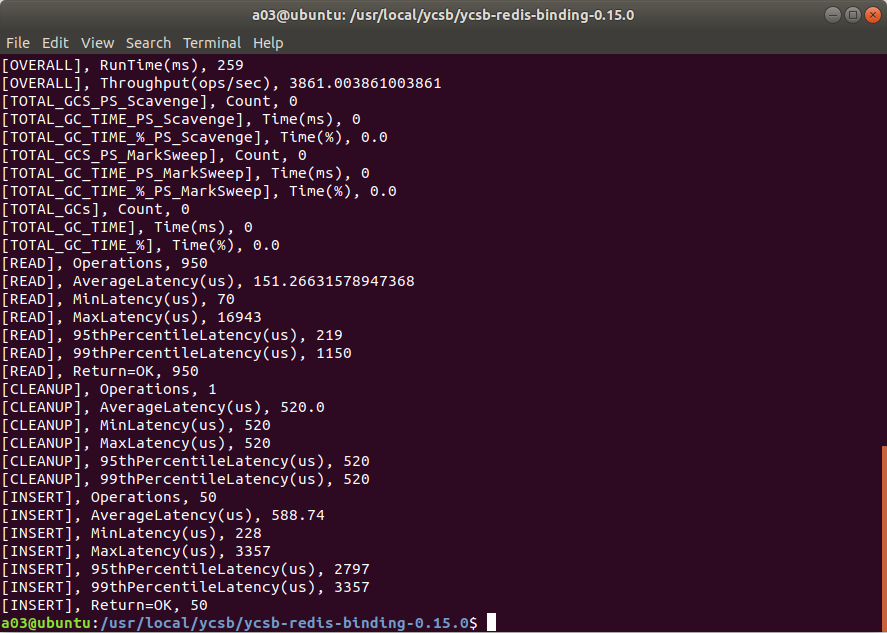


图例 1 单次测试流程

数据测试，单个数据测试流程分三步。

1.导入对应的工作负载，指令为load Redis -s -P workloads/workloada，导入数据库的初始数据。-P参数用来加载property文件。这个例子中，我用来加载workloada中的参数文件。负载类型来自workloada中预先的设置，文件中 recordcount意思是在Redis中的预先导入的数据数，每一个Key都是hash类型，我们可以重执行上面的命令，Redis指的是服务器类型。如不填写，默认为dummy客户端，不操作数据库，而将YCSB对数据库的操作输出到控制台。可以实用这个特性查看各种负载是如何执行到服务器中的。-s 参数表示要求Client向stderr输出状态报告。

2.使用工作负载进行测试，指令为run Redis -s -P workloads/workloada。和导入阶段主要的不同是，告诉服务器这一段指令意味着导入数据已结束，可以直接执行。



图例 2 单次性能测试结果展示

3.将得到的数据导出到OUTPUT.txt文件中。

## 3.4.2 测试脚本编写与数据收集

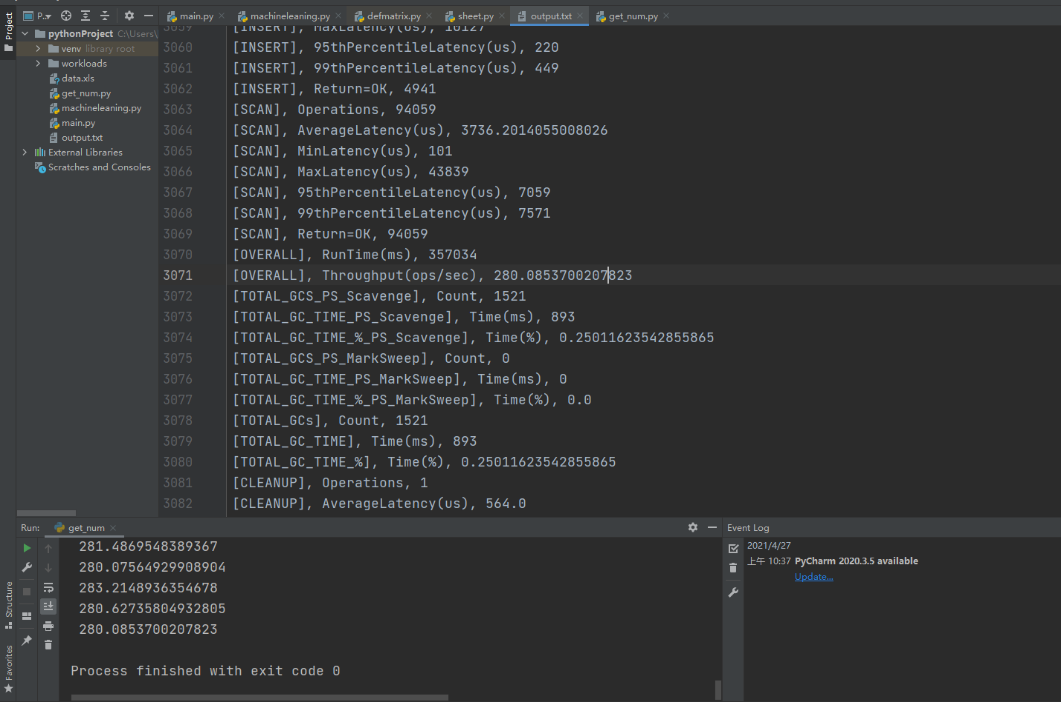
拟测试的数据量从1，000到100，000，以1000为步长。数据量和测试量都非常巨大，由人工测量时间空间误差较大，易受物理环境变化影响，因此编写python脚本进行测试。



图例 3 脚本测试流程

脚本文件在客户机中运行，从文件中载入预先输入的工作负载类型,修改初始数据数操作数总数。将工作负载写入到workload中，随后执行单次工作负载的流程，得到的数据输入到OUTPUT.txt中，重复上述过程，直到从一千到十万数据测试完成。此时OUTPUT.txt中得到的就是当前工作负载的全部测试数据。第二步、修改工作负载类型，重复上述过程，直到全部工作负载均测试完成。

得到的数据文件中包含许多不必要信息，我们只需要提取出其中的吞吐量和运行时间等数据。通过正则表达式可以快速提出。



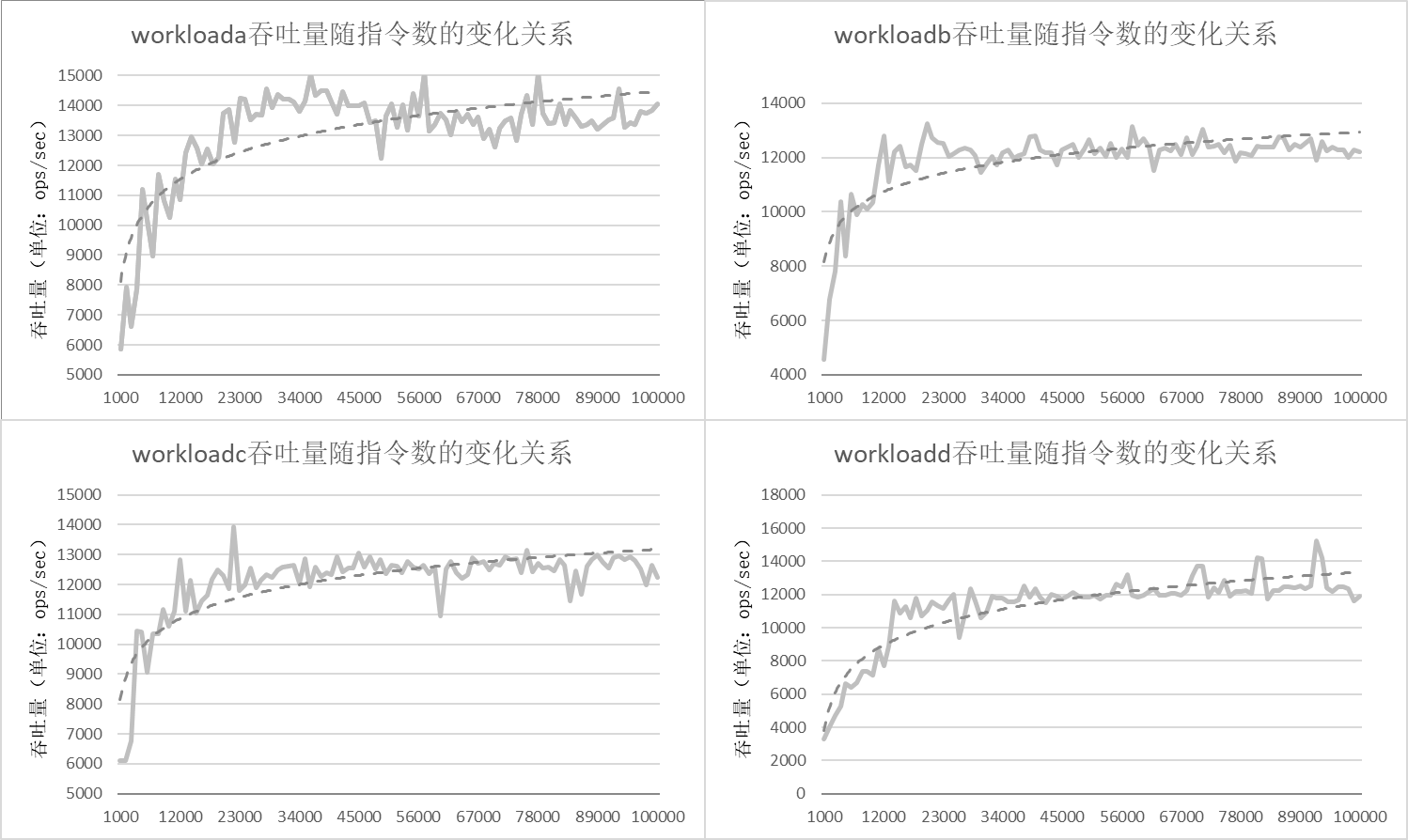
图例 4 workload-A性能测试指标

# 4 建模与验证

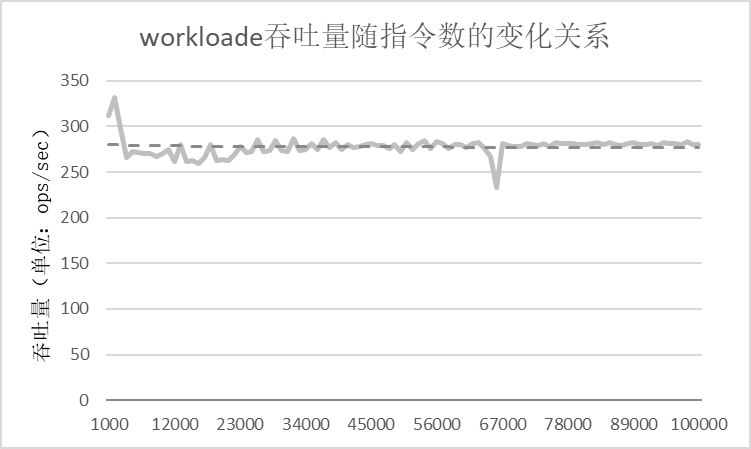
## 4.1 数据预处理与可视化

将实验得到的数据，绘制成曲线。可以看出，workloada ，workloadb ，workloadc ，workloadd四种负载下，单位内执行时间随数据量增大而增加，而且存在不同的上限。总执行时间（延时）则近似呈现正比例关系。对于workloade，由于服务器的处理能力受到查询的长度的限制，总吞吐量很低且则稳定。

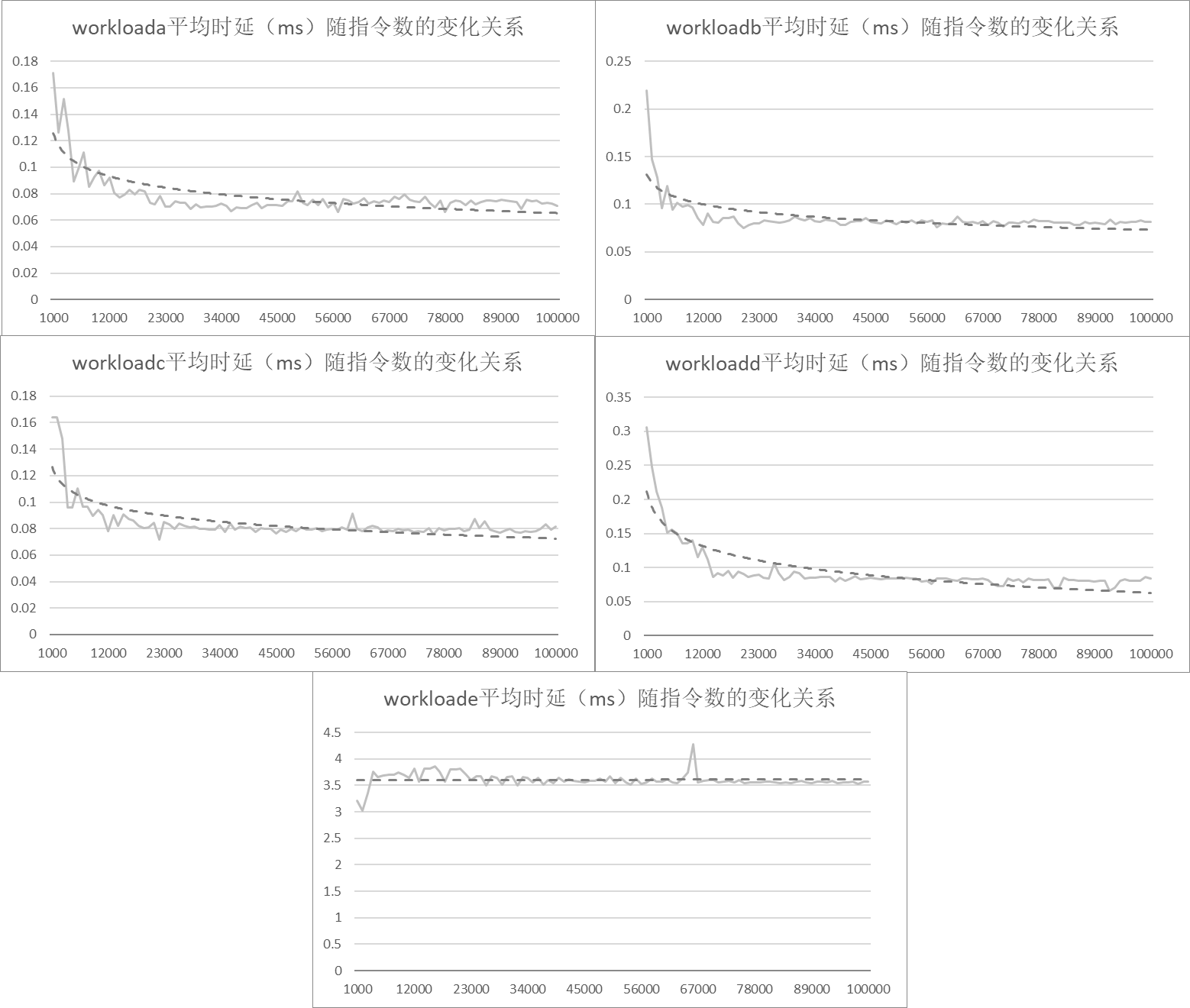
平均时延随着吞吐量增加而降低，且存在下限，



图例 5 workloadA-D 吞吐量随指令数变化情况



图例 6 workload-E吞吐量随操作数变化情况



图例 7 workload A-E 平均时延(ms)随指令数(条)变化情况

## 4.2 数据预测模型建立

分析数据图像可知，在A,B,C,D四种负载特征下，吞吐量随数据变化的图像成非线性关系，数据大多集中在一条满足函数关系的曲线上，采用多项式回归建模，修改多项式最高次数，依次测试回归结果， workload-A的回归模型测试结果如下表格所示：

表格 4 workloadA 对性能指标进行贝叶斯多项式建模

|  |  |  |  |
| --- | --- | --- | --- |
| 最高项次数 | Workloada吞吐量随访问数变化 | 最高项次数 | Workloada平均时延随访问数变化 |
| 二次 |  | 二次 |  |
| 三次 |  | 三次 |  |
| 四次 |  | 四次 |  |
| 五次 |  | 五次 |  |
| 六次 |  | 六次 |  |
| 七次 |  | 七次 |  |
| 八次 |  | 八次 |  |
| 九次 |  | 九次 |  |

据函数图像分析，Redis在workloada负载条件下，吞吐量随操作数变化的最适合选用的多项式次数为五次。每条指令的平均延时（ms）数学模型的多项式最高次数选择为六次。

**Workloada 吞吐量(ops/sec):**

**Workloada 单位时延(ms):**

表格 5 WorkloadB 对性能指标进行贝叶斯多项式建模结果

|  |  |  |  |
| --- | --- | --- | --- |
| 最高项次数 | Workload-B吞吐量随访问数变化 | 最高项次数 | Workload-B平均时延随访问数变化 |
| 二次 |  | 二次 |  |
| 三次 |  | 三次 |  |
| 四次 |  | 四次 |  |
| 五次 |  | 五次 |  |
| 六次 |  | 六次 |  |
| 七次 |  | 七次 |  |
| 八次 |  | 八次 |  |
| 九次 |  | 九次 |  |

据函数图像分析，Redis在Workload-B负载条件下，吞吐量随操作数变化的最适合选用的多项式次数为七次。每条指令的平均延时（ms）数学模型的多项式最高次数选择为七次。Workload上限较为平滑，在操作数为25000时达到上限，约为12300 ops/sec。平均实验与吞吐量近似成反比例关系，

**Workloadb 吞吐量(****ops/sec):**

**Workloadb 单位时延(ms):**

WorkloadC-E的数据建模过程大致相同，因此在此处仅展示结果，其它欠拟合和过拟合结果不再赘述。

**Workload-C 吞吐量(ops/sec):**

**Workload-C 单位时延(ms):**

**Workload-D 吞吐量(ops/sec):**

**Workload-D 单位时延(ms):**

**Workload-E 吞吐量(ops/sec):**

**Workload-E 单位时延(ms):**

表格 5 WorkloadC-E 对性能指标进行贝叶斯多项式建模结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 负载类型 | 吞吐量 | 负载类型 | | 平均单位时延 |
| c |  | c |  | |
| d |  | d |  | |
| e |  | e |  | |

# 4 总结与体会

作为一名大学生，在此次数据库负载性能预测的可行方向和对数据库优化方案的探究过程中，我既为实验室的提供了一定的帮助，又深化了对数据库这方面的理解。毕业设计有利于缓解大学生就业压力，开拓视野。一个专业能力强的大学毕业生不但不会成为社会的就业压力，相反还能通过自主创业活动来增加就业岗位，以缓解社会的就业压力。 有利于大学生自我价值实现。本次毕业设计同时也是我个人兴趣方向，对我个人搭建的服务器也有一定帮助，可以把自己的兴趣与专业知识紧密结合，做自己最感兴趣，最愿意做和自己认为最值得做的事情，实现自我价值。有利于大学生自身素质的提高。通过毕业设计，我可以充分调动自己的主观能动性，改变自身就业心态，自主学习，独立思考，并学会自我调节与控制。

参考文献

[1] 宁方美，贺雪梅，牟晋娟，Spring Boot集成Redis缓存技术在企业一卡通系统中的应用[J].电子技术与软件工程，2019(24):133-134.

[2] Kanellis， Konstantinos， Ramnatthan Alagappan， and Shivaram Venkataraman. "Too many knobs to tune? towards faster database tuning by pre-selecting important knobs." In 12th {USENIX} Workshop on Hot Topics in Storage and File Systems (HotStorage 20). 2020.

[3] Cao， Zhichao， Siying Dong， Sagar Vemuri， and David HC Du. "Characterizing， modeling， and benchmarking RocksDB Key-value workloads at Facebook." In 18th {USENIX} Conference on File and Storage Technologies ({FAST} 20)， pp. 209-223. 2020.

[4] Cooper， Brian F.， Adam Silberstein， Erwin Tam， Raghu Ramakrishnan， and Russell Sears. "Benchmarking cloud serving systems with YCSB." In Proceedings of the 1st ACM symposium on Cloud computing， pp. 143-154. 2010.

[5] Osemwegie， Omoruyi， Kennedy Okokpujie， Nsikan Nkordeh， Charles Ndujiuba， John Samuel， and Uzairue Stanley. "Performance Benchmarking of Key-Value Store NoSQL Databases." International Journal of Electrical and Computer Engineering 8， no. 6 (2018): 5333.

[6] 申德荣， 于戈， 王习特，等. 支持大数据管理的NoSQL系统研究综述[J]. 软件学报， 2013(08):96-113.

[7] 杨武军， 张继荣， 屈军锁. 内存数据库技术综述[J]. 西安邮电学院学报， 2005.

[8]Dimosthenis Kyriazis Cloud forward: From distributed to complete computing[J] Future Generation Computer Systems, 2018, 78

[9]Gregory Chockler, Seth Gilbert, Vincent Gramoli et al. Reconfigurable distributed storage for dynamic networks[J] Journal of Parallel and Distributed Computing, 2008, 69(1)

[10]Hamza Ben-Ammar, Yassine Hadjadj-Aoul, Gerardo Rubino et al. On the performance analysis of distributed caching systems using a customizable Markov chain model[J] Journal of Network and Computer Applications, 2019

致谢

经过几个月的忙碌，我的毕业设计已经基本完成了。作为一名大四生，由于个人经验匮乏，难免有许多地方考虑不周全。如果没有窦辉老师的督促指导，以及同学们的建议提醒，要完成这个设计是十分困难的。在此，衷心感谢她们的支持与帮助。

其次，感谢大学四年来所有的老师，是他们为我打下了夯实的专业基础知识。没有这些专业基础，我不可能完成这次毕业设计，尤其是Adobe Flash Professional 、Adobe Photoshop 软件和Adobe Premiere Pro 的综合运用。

最后感谢学院我的母校——安徽大学，这个将近四年以来给予我知识和关怀的地方。