



【Java限流算法详解及实现】

原创

Hhzyy99

于 2024-05-26 23:10:19 发布

阅读量789

收藏 16

点赞数 15

分类专栏: 微服务 Spring

文章标签: java 算法 开发语言



微服务 同时被 2 个专栏收录

0 订阅 12 篇文章

Java限流算法详解及实现

在高并发场景下，限流是保护系统的重要手段。限流算法可以帮助我们在流量过大时进行合理的控制，避免系统崩溃。本文将详细介绍几种常见的限流算法在Java中的实现。

Java限流算法详解及实现

Java限流算法详解及实现

- 一、令牌桶算法 (Token Bucket)
  - 理论介绍
  - 实现步骤
  - 代码实现
- 二、漏桶算法 (Leaky Bucket)
  - 理论介绍
  - 实现步骤
  - 代码实现
- 三、计数器算法 (Counter)
  - 理论介绍
  - 实现步骤
  - 代码实现
- 四、滑动窗口算法 (Sliding Window)
  - 理论介绍
  - 实现步骤
  - 代码实现
- 结论

一、令牌桶算法 (Token Bucket)

理论介绍

令牌桶算法是一个常用于网络流量整形和流量控制的算法。它的工作原理是系统以固定的速率生成令牌，并将其放入令牌桶中。当请求到来时，需出一定数量的令牌才能继续处理请求。如果桶中没有足够的令牌，则请求会被拒绝或等待。

实现步骤

- 初始化一个令牌桶，设置桶的容量和令牌生成速率。
- 定时向令牌桶中添加令牌，但不能超过桶的最大容量。
- 当请求到来时，检查桶中是否有足够的令牌：
  - 如果有足够的令牌，则允许请求，并从桶中取出相应数量的令牌。
  - 如果没有足够的令牌，则拒绝请求或让请求等待。

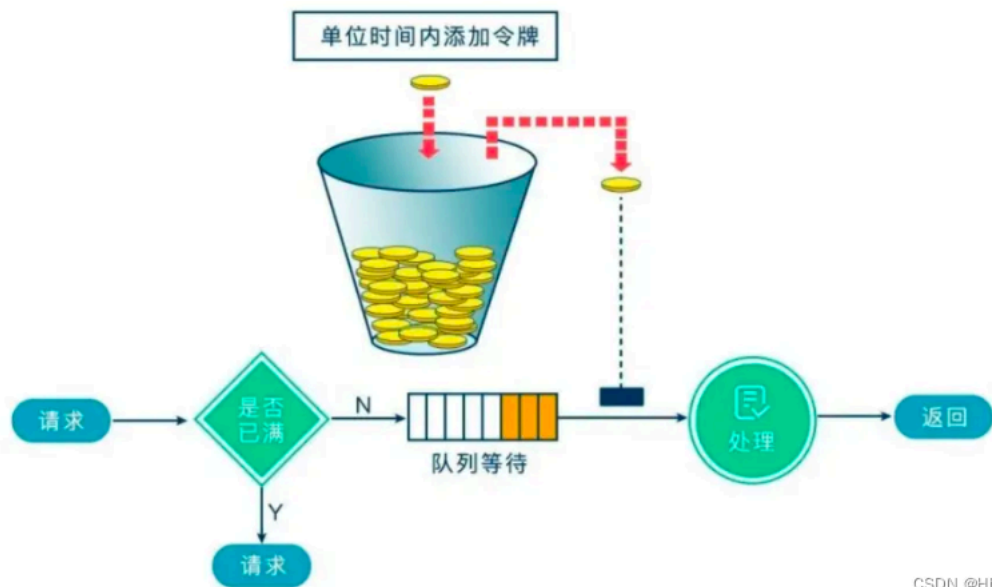
代码实现

```
1 import java.util.concurrent.atomic.AtomicLong;
2
3 public class TokenBucket {
4     private final long capacity;
5     private final long refillRate;
6     private AtomicLong tokens;
7     private long lastRefillTimestamp;
8 }
```



Hhzyy99 关注

```
9 public TokenBucket(long capacity, long refillRate) {
10     this.capacity = capacity;
11     this.refillRate = refillRate;
12     this.tokens = new AtomicLong(capacity);
13     this.lastRefillTimestamp = System.nanoTime();
14 }
15
16 public synchronized boolean tryConsume(long numTokens) {
17     refill();
18     if (tokens.get() >= numTokens) {
19         tokens.addAndGet(-numTokens);
20         return true;
21     }
22     return false;
23 }
24
25 private void refill() {
26     long now = System.nanoTime();
27     long tokensToAdd = (now - lastRefillTimestamp) * refillRate / 1_000_000_000;
28     if (tokensToAdd > 0) {
29         tokens.getAndAdd(Math.min(tokensToAdd, capacity - tokens.get()));
30         lastRefillTimestamp = now;
31     }
32 }
33 }
```



## 二、漏桶算法 (Leaky Bucket)

### 理论介绍

漏桶算法也是一种常用的流量整形和流量控制算法。它的工作原理是系统将请求放入一个漏桶中，桶以固定的速率漏水（处理请求）。当桶满时，新丢弃。

### 实现步骤

1. 初始化一个漏桶，设置桶的容量和漏水速率。
2. 定时以固定速率从漏桶中漏水（处理请求）。
3. 当请求到来时，检查漏桶是否已满：
  - 如果未滿，则将请求放入桶中。
  - 如果已滿，则拒绝请求。

### 代码实现

```
1 import java.util.concurrent.LinkedBlockingQueue;
2
3
```



Hhzy99

关注

```
4 public class LeakyBucket {
5     private final int capacity;
6     private final long leakRate;
7     private final LinkedBlockingQueue<Long> bucket;
8     private long lastLeakTimestamp;
9
10    public LeakyBucket(int capacity, long leakRate) {
11        this.capacity = capacity;
12        this.leakRate = leakRate;
13        this.bucket = new LinkedBlockingQueue<>(capacity);
14        this.lastLeakTimestamp = System.nanoTime();
15    }
16
17    public synchronized boolean tryConsume() {
18        leak();
19        if (bucket.remainingCapacity() > 0) {
20            bucket.offer(System.nanoTime());
21            return true;
22        }
23        return false;
24    }
25
26    private void leak() {
27        long now = System.nanoTime();
28        long numLeaks = (now - lastLeakTimestamp) * leakRate / 1_000_000_000;
29        for (long i = 0; i < numLeaks && !bucket.isEmpty(); i++) {
30            bucket.poll();
31        }
32        lastLeakTimestamp = now;
33    }
34 }
```



### 三、计数器算法 (Counter)

#### 理论介绍

计数器算法是最简单的限流算法。它在一个固定时间窗口内计数请求数量，如果请求数量超过限制，则拒绝请求。

#### 实现步骤

1. 设置一个时间窗口和请求上限。
2. 在时间窗口内计数请求数量。
3. 当请求数量超过上限时，拒绝请求。
4. 在新时间窗口开始时重置计数器。

#### 代码实现

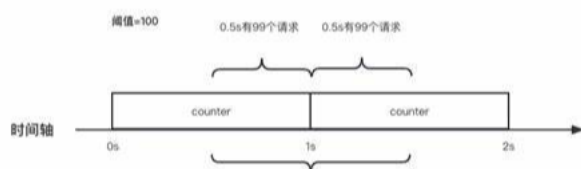
```
1 import java.util.concurrent.atomic.AtomicInteger;
2
3 public class CounterLimiter {
```



Hhzy99

关注

```
4 private final int limit;
5 private final long interval;
6 private AtomicInteger counter;
7 private long startTime;
8
9 public CounterLimiter(int limit, long interval) {
10     this.limit = limit;
11     this.interval = interval;
12     this.counter = new AtomicInteger(0);
13     this.startTime = System.nanoTime();
14 }
15
16 public synchronized boolean tryConsume() {
17     long now = System.nanoTime();
18     if (now - startTime > interval) {
19         counter.set(0);
20         startTime = now;
21     }
22     if (counter.incrementAndGet() <= limit) {
23         return true;
24     }
25     return false;
26 }
27 }
```



CSDN @Hhzy99

## 四、滑动窗口算法 (Sliding Window)

### 理论介绍

滑动窗口算法改进了计数器算法，通过使用多个小窗口来统计请求数量，从而更加准确地反映瞬时流量。

### 实现步骤

1. 将时间窗口分成多个小窗口。
2. 在每个小窗口内计数请求数量。
3. 滑动窗口在新请求到来时更新，计算总请求数量。
4. 如果总请求数量超过限制，则拒绝请求。

### 代码实现

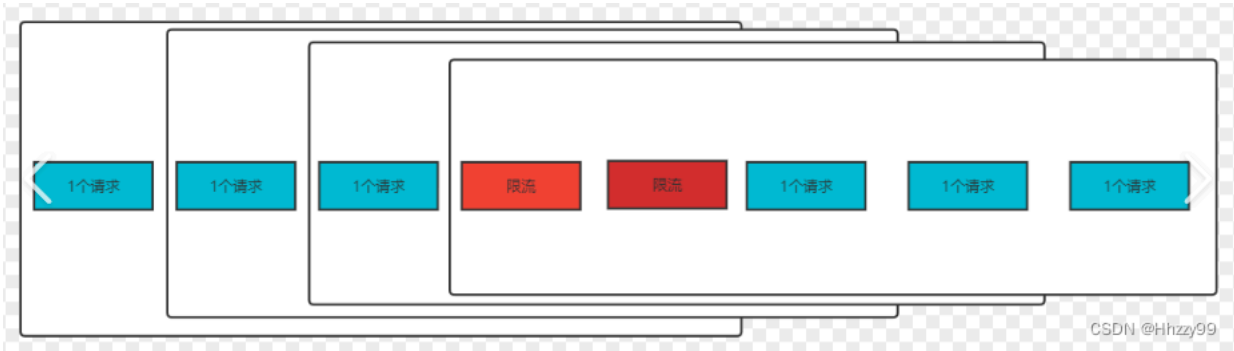
```
1 import java.util.LinkedList;
2 import java.util.Queue;
3
4 public class SlidingWindowLimiter {
5     private final int limit;
6     private final long windowSize;
7     private final Queue<Long> timestamps;
8
9     public SlidingWindowLimiter(int limit, long windowSize) {
10         this.limit = limit;
11         this.windowSize = windowSize;
12         this.timestamps = new LinkedList<>();
13     }
14
15     public synchronized boolean tryConsume() {
16         long now = System.nanoTime();
17         long boundary = now - windowSize;
18         while (!timestamps.isEmpty() && timestamps.peek() < boundary) {
19             timestamps.poll();
20         }
21         if (timestamps.size() < limit) {
22             timestamps.add(now);
23             return true;
24         }
25         return false;
26     }
27 }
```



Hhzy99

关注

```
10         while (!timestamps.isEmpty() && timestamps.peek() < boundary) {
19             timestamps.poll();
20         }
21         if (timestamps.size() < limit) {
22             timestamps.offer(now);
23             return true;
24         }
25         return false;
26     }
27 }
```



CSDN @Hhzy99

结论

限流算法在高并发系统中至关重要。通过合理地选择和实现限流算法，我们可以有效地保护系统免受流量突增的冲击。希望本文对大家理解和实现限流有所帮助。

文章知识点与官方知识档案匹配，可进一步学习相关知识

算法技能树 首页 概览 63056 人正在系统学习中



JAVA快快

微信公众号 >

Java相关技术交流，大家一起学习进步呀！

令牌桶算法示例(java实现)

lanjiangyu的

令牌桶算法实例demo

Java项目如何实现限流？

chenxuyuan的

众所周知，服务器能处理的请求数是有限的，如果请求量特别大，我们就可能需要做限流。限流处理的姿势：要么就让请求等待，要么就把请求给扔了从系统架构来看，手

java限流(漏桶+令牌桶)实现 java令牌桶限流

限流算法基本上有三种:漏桶、令牌桶和信号量这三种,本文是关于漏桶和令牌桶的限流的实现方式。限流注意的点是多线程下限流算法的修改并发问题 一、令牌桶 1.话不多

一文详解 Java 限流常见的四种限流算法

常见的四种限流算法,分别是:固定窗口算法、滑动窗口算法、漏桶算法、令牌桶算法。 二、限流算法 1. 固定窗口 1.1 实现原理 固定窗口又称固定窗口(又称计数器算法,Fix

java实现令牌桶限流

限流是对某一时间窗口内的请求数进行限制，保持系统的可用性和稳定性，防止因流量暴增而导致的系统运行缓慢或宕机。常用的限流算法有令牌桶和漏桶，而Google

手撕漏桶&令牌桶限流算法（Java版）

健身变秃，coding

手撕漏桶限流算法、手撕令牌桶限流算法

java中常见的限流算法详细解析

1. 验证限流以及容器限流 2. 服务端限流 2.1 固定时间窗口 2.2 滑动时间窗口 2.3 漏桶算法 2.4 令牌桶算法 前言 以下的文章参考了一些具体的资料加深了解 B站:Java限流

java实现令牌桶限流器 java中使用令牌桶达到限流

java实现令牌桶限流器 publicabstractclassRateLimiter{ /\*\* \* 限流单位时间段,秒 \*/ protectedfinalLong time; /\*\* \* 时间单位 \*/ protectedfinalTimeUnit timeUnit; /\*\* \* 限流时间

常用的限流算法有哪些？你听说过几种？ 最新发布

yuiezt的

限流，就是指限制流量请求的频次。在高并发情况下，它是一种保护系统的策略，避免了在流量高峰时系统崩溃，造成系统的不可用。

Java常见限流用法介绍和实现

YY-帆S的

在固定时间窗口的基础上进行优化，对大的时间窗口进行划分，每个小窗口对应大窗口中的不同时间点，每个窗口独立计数。又如在最后1个毫秒内请求了100个请求，下

Java并发系列之 第一篇:接口限流算法:漏桶算法&令牌桶算法

第一部分:漏桶算法 漏桶算法是一种简单但有效的接口限流算法。它的原理类似于一个漏水的桶,请求进来后按固



Hhzy99

关注

基于令牌桶算法的Java限流实现  
基于令牌桶算法的Java限流实现。项目需要使用限流措施，查阅后主要使用令牌桶算法实现，为了更灵活的实现限流，就自己实现了一个简单的基于令牌桶算法的限流实

java 限流策略 jijianshuai的i  
概要 在大数据量高并发访问时，经常会出现服务或接口面对暴涨的请求而不可用的情况，甚至引发连锁反映导致整个系统崩溃。此时你需要使用的技术手段之一就是限流

Java限流实现  
NULL 博文链接：<https://bajian1013.iteye.com/blog/2382409>

详解Java分布式IP限流和防止恶意IP攻击方案  
同时，也可以使用其他技术和工具来实现限流和防止恶意IP攻击，例如使用机器学习算法来识别恶意IP。本文提供了一种可行的Java分布式IP限流和防止恶意IP攻击方案。

令牌桶Java实现  
令牌桶 Java 源码 不限制桶大小

详解Springboot分布式限流实践  
限流算法是分布式限流实践的核心部分，常见的限流算法有令牌桶算法、漏桶算法、计数器限流算法等。1. 令牌桶算法 令牌桶算法的原理是系统会以一个恒定的速度往桶

详解Java常用排序算法-插入排序  
在Java中，插入排序算法可以通过以下代码实现：``java public class Insertion { public static void insertionSort(int[] arr) { int n = arr.length; for (int i = 1; i < n; i++) { int key =

详解Java常用排序算法-冒泡排序  
Java排序算法之冒泡排序详解 冒泡排序（Bubble Sort）是一种简单的排序算法，它重复地遍历要排序的数列，一次比较两个元素，如果它们的顺序错误就交换位置。这个

java实现的RC4加密解密算法示例  
"java实现的RC4加密解密算法示例" RC4加密解密算法是Symmetric-key block cipher的一种，使用同一个密钥进行加密和解密。java实现的RC4加密解密算法可以通过以T

令牌桶算法限流 weixin\_33819479的  
限流 限流是对某一时间窗口内的请求数进行限制，保持系统的可用性和稳定性，防止因流量激增而导致的系统运行缓慢或宕机。常用的限流算法有令牌桶和漏桶，而Gc

限流 -- 令牌桶算法 白羊座的时  
一、什么是令牌桶算法？三、单个Jvm应用的限流框架四、分布式的限流框架令牌桶算法是一种流量控制策略，用于限制请求数。它通过维护一个固定容量的“令牌桶”，并

【Java】四种方案实现限流 人生苦短，我用pyt  
详细介绍了四种常见的限流算法：固定窗口、滑动窗口、漏桶和令牌桶。每种算法都有具体的代码示例和适用场景，帮助读者理解和选择最适合自己的限流策略。

关于我们 招贤纳士 商务合作 寻求报道 400-660-0108 kefu@csdn.net 在线客服 工作时间 8:30-22:00  
公安备案号11010502030143 京ICP备19004658号 京网文〔2020〕1039-165号 经营性网站备案信息 北京互联网违法和不良信息举报中心  
家长监护 网络110报警服务 中国互联网举报中心 Chrome商店下载 账号管理规范 版权与免责声明 版权申诉 出版物许可证 营业执照  
©1999-2024北京创新乐知网络技术有限公司



Hhzyy99

码龄4年

Java领域新星创作者

VIP

64

1万+

7305

17万+

原创

周排名

总排名

访问

等级

4594

4375

2176

1525

2547

积分

粉丝

获赞

评论

收藏





























私信


关注

AI圈早知道，每日最新动态

了解全球AI新鲜事！

立即参与





Hhzyy99

关注



搜博主文章

热门文章

【Spring框架】爆gan一万七千字，超详细的AOP技术详解，你真的不来看看吗？ 64359

IDEA报错：Error: java: 错误：不支持发行版本5 10518

【Java数据结构与算法】Day2-高级排序（希尔、归并、快速、计数） 7771

【SpringBoot】一文带你入门SpringBoot 7395

【Spring框架】爆gan两万六千字，助你通关IoC和DI 6860

分类专栏

前端	2篇
AI	3篇
大数据技术	1篇
项目实践	3篇
微服务	12篇
Spring	8篇

最新评论

ChatGPT：人工智能助手的新时代  
CSDN-Ada助手: 恭喜你这篇博客进入【CSDN月度精选】榜单，全部的排名请看 hi ...

ChatGPT：人工智能助手的新时代  
CSDN-Ada助手: 恭喜你这篇博客进入【CSDN月度精选】榜单，全部的排名请看 hi ...

【SSM整合】对Spring、SpringMVC、M...  
m0\_65709426: 请问公众号是啥啊

深入理解与实践Seata：分布式事务解决...  
CSDN-Ada助手: 恭喜你，获得了 2023 博客之星评选的入围资格，请看这个帖子 ...

ChatGPT：人工智能助手的新时代  
CSDN-Ada助手: 恭喜你这篇博客进入【CSDN月度精选】榜单，全部的排名请看 hi ...

最新文章

秒懂 Yarn：安装与配置详解

生成式AI的未来：对话系统与自主代理的双重探索

面对AI发展中的伦理挑战：构建透明、公平和隐私保护的未來

2024年 11篇      2023年 36篇

 Hhzyy99

关注



2022年 17篇

目录

代码实现

二、漏桶算法 (Leaky Bucket)

理论介绍

实现步骤

代码实现

三、计数器算法 (Counter)

理论介绍

实现步骤

代码实现

四、滑动窗口算法 (Sliding Window)

理论介绍

实现步骤

代码实现

结论