

Elasticsearch源码解析与优化实战



[Elasticsearch源码解析与优化实战 下载链接1](#)

著者:张超

出版者:电子工业出版社

出版时间:2019-1

装帧:平装

isbn:9787121352164

《Elasticsearch源码解析与优化实战》介绍了Elasticsearch的系统原理，旨在帮助读者了解其内部原理、设计思想，以及在生产环境中如何正确地部署、优化系统。系统原理分两方面介绍，一方面详细介绍主要流程，例如启动流程、选主流程、恢复流程；另一方面介绍各重要模块的实现，以及模块之间的关系，例如gateway模块、allocation模块等。本书的最后一部分介绍如何优化写入速度、搜索速度等大家关心的实际问题，并提供了一些诊断问题的方法和工具供读者参考。

《Elasticsearch源码解析与优化实战》适合对Elasticsearch进行改进的研发人员、平台运维人员，对分布式搜索感兴趣的朋友，以及在使用Elasticsearch过程中遇到问题的人

们。

作者介绍:

张超

长期从事服务端和基础架构等研发工作，对搜索、分布式系统、高性能网络服务有浓厚的兴趣，喜欢探究技术本质，喜欢分析有深度的问题。目前就职于360企业安全集团基础大数据团队，负责平台内核研发工作。

目录: 第1章 走进Elasticsearch

1.1 基本概念和原理

1.1.1 索引结构

1.1.2 分片 (shard)

1.1.3 动态更新索引

1.1.4 近实时搜索

1.1.5 段合并

1.2 集群内部原理

1.2.1 集群节点角色

1.2.2 集群健康状态

1.2.3 集群状态

1.2.4 集群扩容

1.3 客户端API

1.4 主要内部模块简介

1.4.1 模块结构

1.4.2 模块管理

第2章 准备编译和调试环境

2.1 编译源码

2.1.1 准备JDK和Gradle

2.1.2 下载源代码

2.1.3 编译项目，打包

2.1.4 将工程导入IntelliJ IDEA

2.2 调试Elasticsearch

2.2.1 本地运行、调试项目

2.2.2 远程调试

2.3 代码书签和断点组

第3章 集群启动流程

3.1 选举主节点

3.2 选举集群元信息

3.3 allocation过程

3.4 index recovery

3.5 集群启动日志

3.6 小结

第4章 节点的启动和关闭

4.1 启动流程做了什么

4.2 启动流程分析

4.2.1 启动脚本

4.2.2 解析命令行参数和配置文件

4.2.3 加载安全配置

4.2.4 检查内部环境

4.2.5 检测外部环境

4.2.6 启动内部模块

- 4.2.7 启动keepalive线程
- 4.3 节点关闭流程
- 4.4 关闭流程分析
- 4.5 分片读写过程中执行关闭
- 4.6 主节点被关闭
- 4.7 小结
- 第5章 选主流程
- 5.1 设计思想
- 5.2 为什么使用主从模式
- 5.3 选举算法
- 5.4 相关配置
- 5.5 流程概述
- 5.6 流程分析
- 5.6.1 选举临时Master
- 5.6.2 投票与得票的实现
- 5.6.3 确立Master或加入集群
- 5.7 节点失效检测
- 5.7.1 NodesFaultDetection事件处理
- 5.7.2 MasterFaultDetection事件处理
- 5.8 小结
- 第6章 数据模型
- 6.1 PacificA算法
- 6.1.1 数据副本策略
- 6.1.2 配置管理
- 6.1.3 错误检测
- 6.2 ES的数据副本模型
- 6.2.1 基本写入模型
- 6.2.2 写故障处理
- 6.2.3 基本读取模型
- 6.2.4 读故障处理
- 6.2.5 引申的含义
- 6.2.6 系统异常
- 6.3 Allocation IDs
- 6.3.1 安全地分配主分片
- 6.3.2 将分配标记为陈旧
- 6.2.3 一个例子
- 6.3.4 不会丢失全部
- 6.4 Sequence IDs
- 6.4.1 Primary Terms和Sequence Numbers
- 6.4.2 本地及全局检查点
- 6.4.3 用于快速恢复 (Recovery)
- 6.5 _version
- 第7章 写流程
- 7.1 文档操作的定义
- 7.2 可选参数
- 7.3 Index/Bulk基本流程
- 7.4 Index/Bulk详细流程
- 7.4.1 协调节点流程
- 7.4.2 主分片节点流程
- 7.4.3 副分片节点流程
- 7.5 I/O异常处理
- 7.5.1 Engine关闭过程
- 7.5.2 Master的对应处理
- 7.5.3 异常流程总结

- 7.6 系统特性
- 7.7 思考
- 第8章 GET流程
 - 8.1 可选参数
 - 8.2 GET基本流程
 - 8.3 GET详细分析
 - 8.3.1 协调节点
 - 8.3.2 数据节点
 - 8.4 MGET流程分析
 - 8.5 思考
- 第9章 Search流程
 - 9.1 索引和搜索
 - 9.1.1 建立索引
 - 9.1.2 执行搜索
 - 9.2 search type
 - 9.3 分布式搜索过程
 - 9.3.1 协调节点流程
 - 9.3.2 执行搜索的数据节点流程
 - 9.4 小结
- 第10章 索引恢复流程分析
 - 10.1 相关配置
 - 10.2 流程概述
 - 10.3 主分片恢复流程
 - 10.4 副分片恢复流程
 - 10.4.1 流程概述
 - 10.4.2 synced flush机制
 - 10.4.3 副分片节点处理过程
 - 10.4.4 主分片节点处理过程
 - 10.5 recovery速度优化
 - 10.6 如何保证副分片和主分片一致
 - 10.7 recovery相关监控命令
 - 10.8 小结
- 第11章 gateway模块分析
 - 11.1 元数据
 - 11.2 元数据的持久化
 - 11.3 元数据的恢复
 - 11.4 元数据恢复流程分析
 - 11.4.1 选举集群级和索引级别的元数据
 - 11.4.2 触发allocation
 - 11.5 思考
- 第12章 allocation模块分析
 - 12.1 什么是allocation
 - 12.2 触发时机
 - 12.3 allocation模块结构概述
 - 12.4 allocators
 - 12.5 deciders
 - 12.5.1 负载均衡类
 - 12.5.2 并发控制类
 - 12.5.3 条件限制类
 - 12.6 核心reroute实现
 - 12.6.1 集群启动时reroute的触发时机
 - 12.6.2 流程分析
 - 12.6.3 gatewayAllocator
 - 12.6.4 shardsAllocator

- 12.7 从gateway到allocation流程的转换
- 12.8 从allocation流程到recovery流程的转换
- 12.9 思考
- 第13章 Snapshot模块分析
 - 13.1 仓库
 - 13.2 快照
 - 13.2.1 创建快照
 - 13.2.2 获取快照信息
 - 13.2.3 快照status
 - 13.2.4 取消、删除快照和恢复操作
 - 13.3 从快照恢复
 - 13.3.1 部分恢复
 - 13.3.2 恢复过程中更改索引设置
 - 13.3.3 监控恢复进度
 - 13.4 创建快照的实现原理
 - 13.4.1 Lucene文件格式简介
 - 13.4.2 协调节点流程
 - 13.4.3 主节点流程
 - 13.4.4 数据节点流程
 - 13.5 删除快照实现原理
 - 13.5.1 协调节点流程
 - 13.5.2 主节点流程
 - 13.6 思考与总结
- 第14章 Cluster模块分析
 - 14.1 集群状态
 - 14.2 内部封装和实现
 - 14.2.1 MasterService
 - 14.2.2 ClusterApplierService
 - 14.2.3 线程池
 - 14.3 提交集群任务
 - 14.3.1 内部模块如何提交任务
 - 14.3.2 任务提交过程实现
 - 14.4 集群任务的执行过程
 - 14.5 集群状态的发布过程
 - 14.5.1 增量发布的实现原理
 - 14.5.2 二段提交总流程
 - 14.5.3 发布过程
 - 14.5.4 提交过程
 - 14.5.5 异常处理
 - 14.6 应用集群状态
 - 14.7 查看等待执行的集群任务
 - 14.8 任务管理API
 - 14.8.1 列出运行中的任务
 - 14.8.2 取消任务
 - 14.9 思考与总结
- 第15章 Transport模块分析
 - 15.1 配置信息
 - 15.1.1 传输模块配置
 - 15.1.2 通用网络配置
 - 15.2 Transport总体架构
 - 15.2.1 网络层
 - 15.2.2 服务层
 - 15.3 REST解析和处理
 - 15.4 RPC实现

- 15.4.1 RPC的注册和映射
- 15.4.2 根据Action获取处理类
- 15.5 思考与总结
- 第16章 ThreadPool模块分析
- 16.1 线程池类型
 - 16.1.1 fixed
 - 16.1.2 scaling
 - 16.1.3 direct
 - 16.1.4 fixed_auto_queue_size
- 16.2 处理器设置
- 16.3 查看线程池
 - 16.3.1 cat thread pool
 - 16.3.2 nodes info
 - 16.3.3 nodes stats
 - 16.3.4 nodes hot threads
 - 16.3.5 Java的线程池结构
- 16.4 ES的线程池实现
 - 16.4.1 ThreadPool类结构与初始化
 - 16.4.2 fixed类型线程池构建过程
 - 16.4.3 scaling类型线程池构建过程
 - 16.4.4 direct类型线程池构建过程
 - 16.4.5 fixed_auto_queue_size类型线程池构建过程
- 16.5 其他线程池
- 16.6 思考与总结
- 第17章 Shrink原理分析
 - 17.1 准备源索引
 - 17.2 缩小索引
 - 17.3 Shrink的工作原理
 - 17.3.1 创建新索引
 - 17.3.2 创建硬链接
 - 17.3.3 硬链接过程源码分析
- 第18章 写入速度优化
 - 18.1 translog flush间隔调整
 - 18.2 索引刷新间隔refresh_interval
 - 18.3 段合并优化
 - 18.4 indexing buffer
 - 18.5 使用bulk请求
 - 18.5.1 bulk线程池和队列
 - 18.5.2 并发执行bulk请求
 - 18.6 磁盘间的任务均衡
 - 18.7 节点间的任务均衡
 - 18.8 索引过程调整和优化
 - 18.8.1 自动生成doc ID
 - 18.8.2 调整字段Mappings
 - 18.8.3 调整_source字段
 - 18.8.4 禁用_all字段
 - 18.8.5 对Analyzed的字段禁用Norms
 - 18.8.6 index_options 设置
 - 18.9 参考配置
 - 18.10 思考与总结
- 第19章 搜索速度的优化
 - 19.1 为文件系统cache预留足够的内存
 - 19.2 使用更快的硬件
 - 19.3 文档模型

- 19.4 预索引数据
- 19.5 字段映射
- 19.6 避免使用脚本
- 19.7 优化日期搜索
- 19.8 为只读索引执行force-merge
- 19.9 预热全局序号 (global ordinals)
- 19.10 execution hint
- 19.11 预热文件系统cache
- 19.12 转换查询表达式
- 19.13 调节搜索请求中的batched_reduce_size
- 19.14 使用近似聚合
- 19.15 深度优先还是广度优先
- 19.16 限制搜索请求的分片数
- 19.17 利用自适应副本选择 (ARS) 提升ES响应速度
- 第20章 磁盘使用量优化
 - 20.1 预备知识
 - 20.1.1 元数据字段
 - 20.1.2 索引映射参数
 - 20.2 优化措施
 - 20.2.1 禁用对你来说不需要的特性
 - 20.2.2 禁用doc values
 - 20.2.3 不要使用默认的动态字符串映射
 - 20.2.4 观察分片大小
 - 20.2.5 禁用_source
 - 20.2.6 使用best_compression
 - 20.2.7 Force Merge
 - 20.2.8 Shrink Index
 - 20.2.9 数值类型长度够用就好
 - 20.2.10 使用索引排序来排列类似的文档
 - 20.2.11 在文档中以相同的顺序放置字段
 - 20.3 测试数据
- 第21章 综合应用实践
 - 21.1 集群层
 - 21.1.1 规划集群规模
 - 21.1.2 单节点还是多节点部署
 - 21.1.3 移除节点
 - 21.1.4 独立部署主节点
 - 21.2 节点层
 - 21.2.1 控制线程池的队列大小
 - 21.2.2 为系统cache保留一半物理内存
 - 21.3 系统层
 - 21.3.1 关闭swap
 - 21.3.2 配置Linux OOM Killer
 - 21.3.3 优化内核参数
 - 21.4 索引层
 - 21.4.1 使用全局模板
 - 21.4.2 索引轮转
 - 21.4.3 避免热索引分片不均
 - 21.4.4 副本数选择
 - 21.4.5 Force Merge
 - 21.4.6 Shrink Index
 - 21.4.7 close索引
 - 21.4.8 延迟分配分片
 - 21.4.9 小心地使用fielddata

- 21.5 客户端
 - 21.5.1 使用REST API而非Java API
 - 21.5.2 注意429状态码
 - 21.5.3 curl的HEAD请求
 - 21.5.4 了解你的搜索计划
 - 21.5.5 为读写请求设置比较长的超时时间
- 21.6 读写
 - 21.6.1 避免搜索操作返回巨大的结果集
 - 21.6.2 避免索引巨大的文档
 - 21.6.3 避免使用多个_type
 - 21.6.4 避免使用_all字段
 - 21.6.5 避免将请求发送到同一个协调节点
- 21.7 控制相关度
- 第22章 故障诊断
 - 22.1 使用Profile API定位慢查询
 - 22.2 使用Explain API分析未分配的分片 (Unassigned Shards)
 - 22.2.1 诊断未分配的主分片
 - 22.2.2 诊断未分配的副分片
 - 22.2.3 诊断已分配的分片
 - 22.3 节点CPU使用率高
 - 22.4 节点内存使用率高
 - 22.5 Slow Logs
 - 22.6 分析工具
 - 22.6.1 I/O信息
 - 22.6.2 内存
 - 22.6.3 CPU信息
 - 22.6.4 网络连接和流量
 - 22.7 小结
- 附录A 重大版本变化
 - • • • • [\(收起\)](#)

[Elasticsearch源码解析与优化实战 下载链接1](#)

标签

ElasticSearch

es源码解析

Elasticsearch源码解析与优化实战

计算机科学

搜索

数据_存储

编程

搜索引擎

评论

既不深入也不浅出。考虑到es的好书很少，网上原创博文也不多，给个3星鼓励下。

只是快速的过了第一遍，由于对elasticsearch的使用尚不是很熟悉，所以看的有点云里雾里，推荐看过elasticsearch实战以后，并且在使用对elasticsearch理解比较深入以后再来看这个本书效果会更好。
不推荐新手上来就看这本书，整体上书的内容还是不错的。

es源码20几万行呢，这本提供了很多帮助，对模块的拆解还有一些流程机制写的很清楚。当然设计思路这种没办法强求了，版本迭代太快，小哥哥也很拼了，加油~~

看了等于没看。。

有很多优化项，理由讲了很多，看起来很靠谱。

某种意义上说，elasticsearch可分为集群层、索引层、分片层和最后的存储引擎层（lucene）；集群层，一个节点作为master，采用bully算法选出，负责进行allocation、全局状态管理等；其他节点作为协调节点（gateway、query、route & merge）和数据节点；每个数据节点多个分片，分片间主从，采用PacificA、translog进行同步；一个比较奇怪的点是，es居然是partition by DocId（而非term）这导致了其搜索必须采用广播形式，因此无法做到很大规模；线程模型嘛，看起来是按照任务，分为不同独立的线程池和队列，底层数据共享；

当然，书里面有一些技巧是不错的（如：ARS & 预索引）

授人以鱼，不如授人以渔

偶尔翻一翻工具书都觉得枯燥乏味，技术热忱远不如从前了

还可以，不过感觉作者为了怕被骂贴代码，反而代码不部分过于少了，有些地方不太能联系起来

基于ES6.x版本写的，与ES7.x比有部分差异，比如选主算法换了。。。总体而言写的还是很细致的，推荐！

大佬 应该多写几章啊，写的真的不错

[Elasticsearch源码解析与优化实战_下载链接1](#)

书评

初看ES源码，有一本这样的源码解析的书，知道从哪个模块，更具体到哪个类入手开始看，所以还是有一些帮助的。其实发现书中好多东西也是从官方文档中借鉴过来的吧，结合书和英文文档看，理解起来会快一些。比如第六章数据副本模型：参考的链接就有很多：[docs-replication]、[e...

[Elasticsearch源码解析与优化实战_下载链接1](#)