第五次第十周小班讨论（个人资料）

搜索引擎问题

全文搜索

湖南大学 HUNAN UNIVERSITY

前言 PREFACE

全文搜索引擎、目录搜索引擎和元搜索引擎是互联网搜索领域的三大主流技术，它们各自具有独特的工作原理、优缺点和应用场景。

搜索引擎是对数据的检索，所以我们先从生活中的数据说起。我们生活中的数据总体分为两种：

•结构化数据

•非结构化数据

结构化数据： 也称作行数据，是由二维表结构来逻辑表达和实现的数据，严格地遵循数据格式与长度规范，主要通过关系型数据库进行存储和管理。指具有固定格式或有限长度的数据，如数据库，元数据等。

非结构化数据： 又可称为全文数据，不定长或无固定格式，不适于由数据库二维表来表现，包括所有格式的办公文档、XML、HTML、Word 文档，邮件，各类报表、图片和音频、视频信息等。

说明：如果要更细致的区分的话，XML、HTML 可划分为半结构化数据。因为它们也具有自己特定的标签格式，所以既可以根据需要按结构化数据来处理，也可抽取出纯文本按非结构化数据来处理。

根据两种数据分类，搜索也相应的分为两种：

•结构化数据搜索

•非结构化数据搜索

对于结构化数据，因为它们具有特定的结构，所以我们一般都是可以通过关系型数据库（MySQL，Oracle 等）的二维表（Table）的方式存储和搜索，也可以建立索引。

对于非结构化数据，也即对全文数据的搜索主要有两种方法：

•顺序扫描

•全文检索

顺序扫描： 通过文字名称也可了解到它的大概搜索方式，即按照顺序扫描的方式查询特定的关键字。

目录

湖南大学
HUNAN UNIVERSITY

第一部分

概述

搜索引擎

信息检索 (Information Retrieval 简称 IR) 和 搜索 (Search) 是有区别的，信息检索是一门学科，研究信息的获取、表示、存储、组织和访问，而搜索只是信息检索的一个分支，其他的如问答系统、信息抽取、信息过滤也可以是信息检索。

搜索引擎（Search Engine）是指根据一定的策略、运用特定的计算机程序从互联网上采集信息（主动采集或被动接收信息），在对信息进行组织和处理后，为用户提供检索服务，将检索的相关信息展示给用户的系统。 搜索引擎是工作于互联网上的一门检索技术，它指在提高人们获取搜集信息的速度，为人们提供更好的网络使用环境。从功能和原理上搜索引擎大致被分为全文搜索引擎、元搜索引擎、垂直搜索引擎和目录搜索引擎等四大类，目前一般通俗提到“搜索引擎”主要是指“全文搜索引擎”。

湖南大学
HUNAN UNIVERSITY

第二部分

实现原理

标题二

副标题

- Part1. 分词
- Part2.倒排索引
- Part 3.查询结果排序
- Part 4、空间索引
- Part 5、数值索引

正确性和粒度

分词就是对一段文本，通过规则或者算法分出多个词，每个词作为搜索的最细粒度一个个单字或者单词。**只有分词后有这个词，搜索才能搜到，分词的正确性非常重要。**分词粒度太大，搜索召回率就会偏低，分词粒度太小，准确率就会降低。如何恰到好处的分词，是搜索引擎需要做的第一步。

正确性&粒度

•分词正确性

- “他说的确实在理”，这句话如何分词？
- “他-说-的确-实在-理”[错误语义]
- “他-说-的-确实-在理”[正确语义]

分词的粒度

- “中华人民共和国宪法”，这句话如何分词？
- “中华人民共和国-宪法”，[搜索 中华、共和国 无结果]
- “中华-人民-共和国-宪法”，[搜索 共和 无结果]
- “中-华-人-民-共-和-国-宪-法”，[搜索其中任意字都有结果]

分词的粒度并不是越小越好，他会降低准确率，比如搜索 “中秋” 也会出现上条结果，而且粒度越小，索引词典越大，搜索效率也会下降，后面会细说。

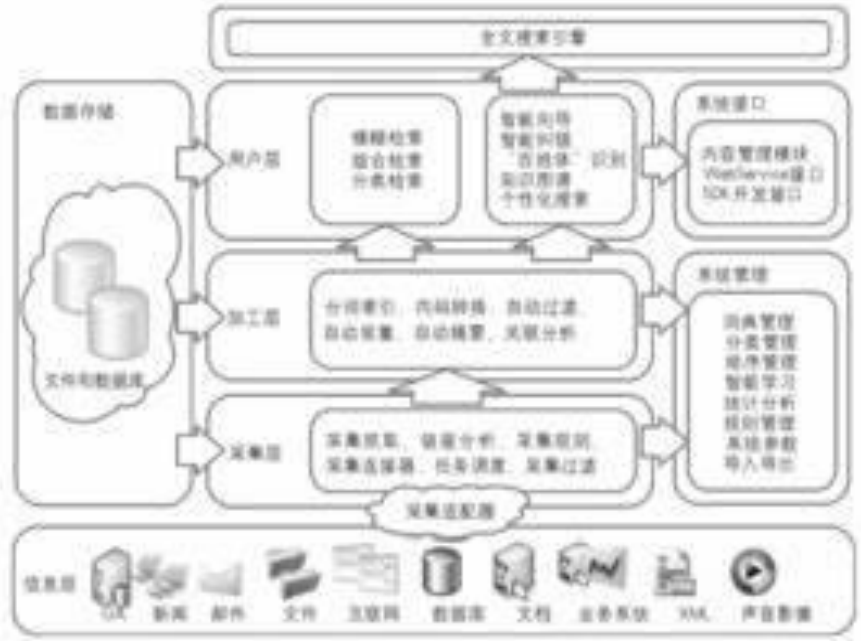
停 用 词 与 词 项 处 理

很多语句中的词都是没有意义的，比如“的”，“在”等副词、谓词，英文中的“a”，“an”，“the”，在搜索是无任何意义的，所以在分词构建索引时都会去除，降低不不要的索引空间，叫停用词(StopWord)。

通常可以通过文档集频率和维护停用词表的方式来判断停用词。

词项处理

词项处理，是指在原本的词项上在做一些额外的处理，比如归一化、词形归并、词干还原等操作，以提高搜索的效果。并不是所有的需求和业务都要词项处理，需要根据场景来判断。



词 项 处 理

1.归一化

- USA - U.S.A. [缩写]
- 7月30日 - 7/30 [中英文]
- color - colour [通假词]
- 开心 - 高兴 [同义词扩展范畴]

这样查询 U.S.A. 也能得到 USA 的结果，同义词可以算作归一化处理，不过同义词还可以有其他的处理方式。

2.词形归并（Lemmatization）

针对英语同一个词有不同的形态，可以做词形归并成一个，如：

- am, are, is -> be
- car, cars, car's, cars' -> car
- the boy's cars are different colors -> the boy car be different color

3.词干还原（Stemming）

通常指的就粗略的去除单词两端词缀的启发式过程

- automate(s), automatic, automation -> automat.
- 高高兴兴 -> 高兴 [中文重叠词还原]
- 明明白白 -> 明白

英文的常见词干还原算法，Porter算法。

Part2、倒排索引

正排索引

正排索引就是 MySQL 里的 B+ Tree，索引的结果是：

- “搜索引擎是信息检索系统” -> id2
- “搜索引擎提供检索服务” -> id1

表示对完整内容按字典序排序，得到一个有序的列表，以加快检索的速度。

倒排索引

第一步 分词

- “搜索引擎-提供-检索-服务” -> id1
- “搜索引擎-信息-检索-系统” -> id2

第二步 将分词项构建一个词典

- 搜索引擎
- 提供
- 检索
- 服务
- 信息
- 系统

第三步 构建倒排链

- 搜索引擎 -> id1, id2
- 提供 -> id1
- 检索 -> id1, id2
- 服务 -> id1
- 信息 -> id2
- 系统 -> id2

由此，一个倒排索引就完成了，搜索 “检索” 时，得到 id1, id2，说明这两条数据都有，搜索 “服务” 只有 id1 存在。但如果搜索 “检索系统”，此时会先建搜索词按照与构建同一种策略分词，得到 “检索-系统”，两个词项，分别搜索 检索 -> id1, id2 和 系统 -> id2，然后对其做一个交集，得到 id2。同理，通过求并集可以支持更复杂的查询。

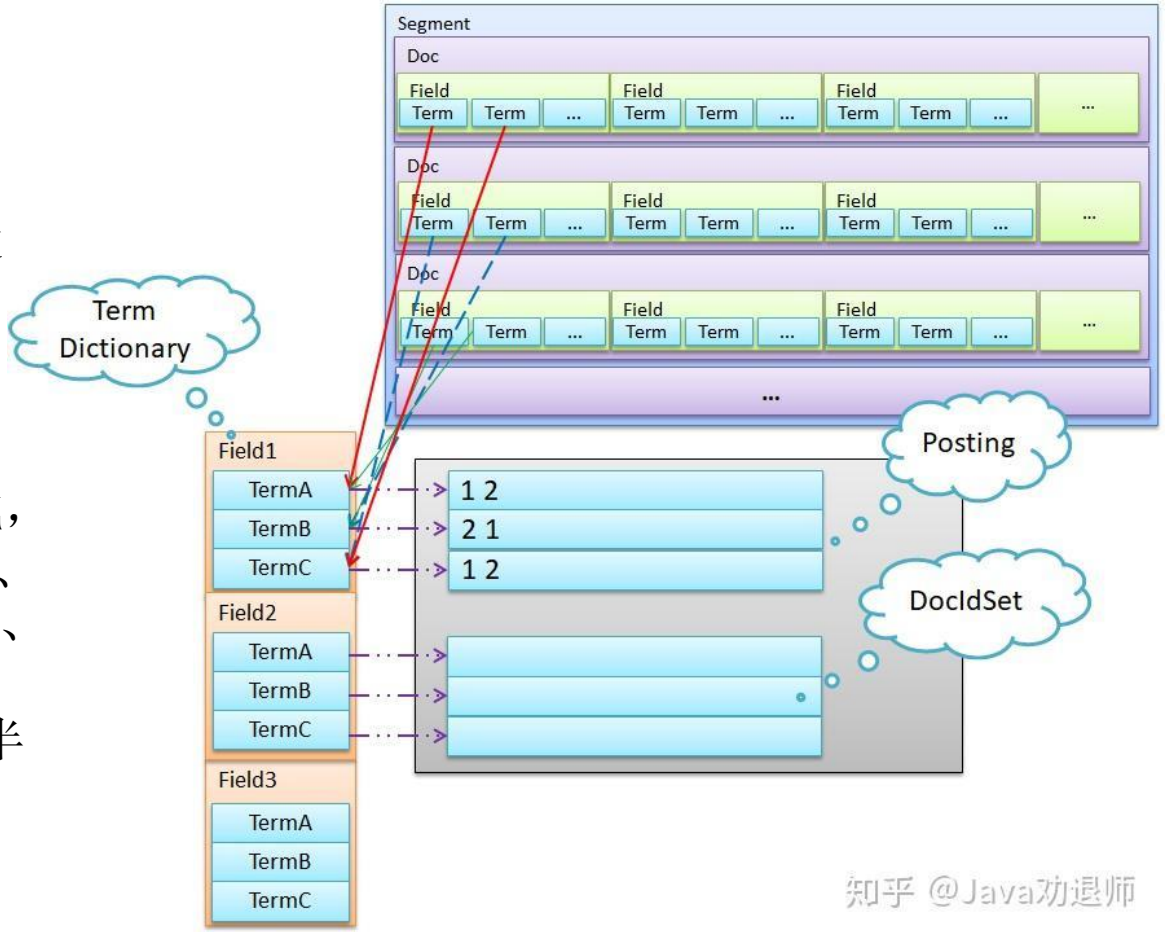
Part2、倒排索引

储存结构

结构化数据： 也称作行数据，是由二维表结构来逻辑表达和实现的数据，严格地遵循数据格式与长度规范，主要通过关系型数据库进行存储和管理。指具有固定格式或有限长度的数据，如数据库，元数据等。

非结构化数据： 又可称为全文数据，不定长或无固定格式，不适于由数据库二维表来表现，包括所有格式的办公文档、XML、HTML、Word 文档，邮件，各类报表、图片和音频、视频信息等。

说明：如果要更细致的区分的话，XML、HTML 可划分为半结构化数据。因为它们也具有自己特定的标签格式，所以既可以根据需要按结构化数据来处理，也可抽取出纯文本按非结构化数据来处理。



以 Lucene 为例，简单说明一下 Lucene 的存储结构。从大到小是Index -> Segment -> Doc -> Field -> Term，类比 MySQL 为 Database -> Table -> Record -> Field -> Value。

Part 3、查询结果排序

1 TF-IDF

2 BM25

搜索结果排序是根据 关键字 和 Document 的相关性得分排序，通常意义下，除了可以人工的设置权重 boost，也存在一套非常有用的相关性得分算法

Part 3、查询结果排序

TF-IDF

TF(词频)-IDF(逆文档频率) 在自动提取文章关键词上经常用到，通过它可以知道某个关键字在这篇文档里的重要程度。其中 TF 表示某个 Term 在 Document 里出现的频次，越高说明越重要；DF 表示在全部 Document 里，共有多少个 Document 出现了这个词，DF 越大，说明这个词很常见，并不重要，越小反而说明他越重要，IDF 是 DF 的倒数(取log)， IDF 越大，表示这个词越重要。

TF-IDF 怎么影响搜索排序，举一个实际例子来解释：

假定现在有一篇博客**《Blink 实战总结》**，我们要统计这篇文章的关键字，首先是对文章分词统计词频，出现次数最多的词是--"的"、"是"、"在"，这些是“停用词”，基本上在所有的文章里都会出现，他对找到结果毫无帮助，全部过滤掉。
只考虑剩下的有实际意义的词，如果文章中词频数关系： “Blink” > “词频” = “总结”，那么肯定是 Blink 是这篇文章更重要的关键字。

Part 3、查询结果排序

BM25

BM25算法，通常用来作搜索相关性平分。一句话概况其主要思想：对Query进行语素解析，生成语素qi；然后，对于每个搜索结果D，计算每个语素qi与D的相关性得分，最后，将qi相对于D的相关性得分进行加权求和，从而得到Query与D的相关性得分。

BM25算法的一般性公式如下：

$$Score(Q,d)=\sum_{i}^{n}W_i\cdot R(q_i,d)$$

其中，Q表示Query，qi表示Q解析之后的一个语素（对中文而言，我们可以把对Query的分词作为语素分析，每个词看成语素qi。）；d表示一个搜索结果文档；Wi表示语素qi的权重；R(qi，d)表示语素qi与文档d的相关性得分。

其中 Wi 通常使用 IDF 来表达，R 使用 TF 来表达；综上，BM25算法的相关性得分公式可总结为：

$$Score(Q,d)=\sum_{i}^{n}IDF(q_i)\cdot\frac{f_i\cdot(k_1+1)}{f_i+k_1\cdot(1-b+b\cdot\frac{dl}{avgdl})}$$

Part 4、空间索引

Geo Hash

1 Geo Hash 如何编码

2 Geo Hash 如何用于地理搜索

3 Geo Hash 依据的数学原理

编码

地球上任何一个位置都可以用经纬度表示，纬度的区间是 [-90, 90]，经度的区间 [-180, 180]。比如天安门的坐标是 39.908,116.397，整体编码过程如下

一、对纬度 39.908 的编码如下：

1.将纬度划分2个区间，左区间 [-90, 0) 用 0 表示，右区间 [0, 90] 用 1 表示， 39.908 处在右区间，故第一位编码是 1；

2.在将 [0, 90] 划分2个区间，左区间 [0, 45) 用 0 表示，右区间 [45, 90] 用 1 表示，39.908处在左区间， 故第二位编码是 0；

3.同1、2的计算步骤，39.908 的最后10位编码是 “10111 00011”

二、对经度 116.397 的编码如下：

1.将经度划分2个区间，左区间 [-180, 0) 用 0 表示，右区间 [0, 180] 用 1 表示，116.397处在右区间， 故第一位编码是 1；

2.在将 [0, 180] 划分2个区间，左区间 [0, 90) 用 0 表示，右区间 [90, 180] 用 1 表示，116.397处在右区间，故第二位编码是 1；

3.同1、2的计算步骤，116.397 的最后6位编码是 “11010 01011”

三、合并组码

1.将奇数位放经度，偶数位放纬度，把2串编码组合生成新串：“11100 11101 00100 01111”；

2.通过 Base32 编码，每5个二进制编码一个数，“28 29 04 15”

3.根据 Base32 表，得到 Geo Hash 为：“WX4G”

即最后天安门的4位 Geo Hash 为 “WX4G”，如果需要经度更准确，在对应的经纬度编码粒度再往下追溯即可。

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Base 32	0	1	2	3	4	5	6	7	8	9	b	c	d	e	f	g

地理搜索

举个例子，搜索天安门附近 200 米的景点，如下是天安门附近的Geo编码

1.首先确定天安门的Geo Hash为 WX4G0B，（6位区域码约0.34平分千米，约为长宽600米区域)

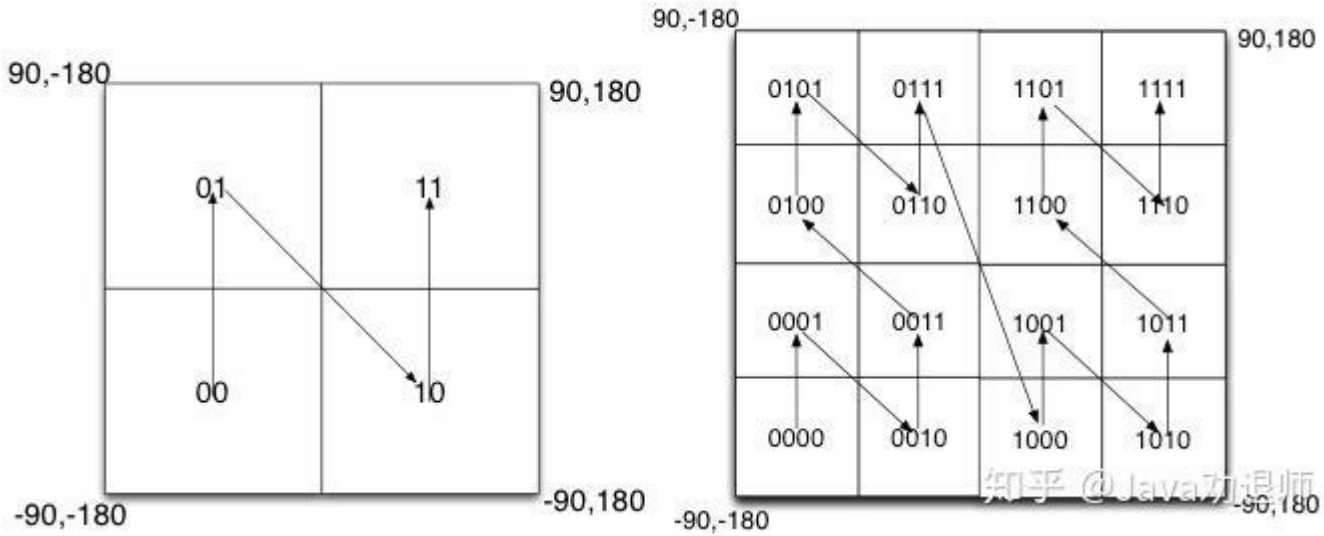
2.而6位编码表示 600 米，半径 300 米 > 要求的 200 米，搜索所有编码为 WX4G0B 的景点即可

3.但是由于天安门处于 WX4G0B 的边缘位置，并不一定处在正中心。这就需要将 WX4G0B 附近的8个区域同时纳入搜索，故搜索 WX4G0B、WX4G09、WX4G0C 一共9个编码的景点

4.第3步已经将范围缩小到很小的一个区间，但是得到的景点距离并不是准确的，需要在通过距离计算过滤出小于 200 米的景点，得到最终结果。

由上面步骤可以看出，Geo Hash 将原本大量的距离计算，变成一个字符串检索缩小范围后，再进行小范围的距离计算，及快速又准确的进行距离搜索。

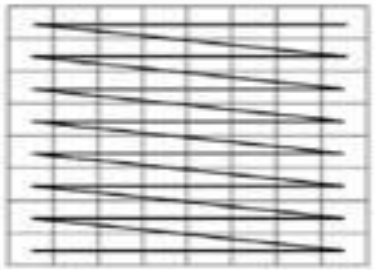
数学原理



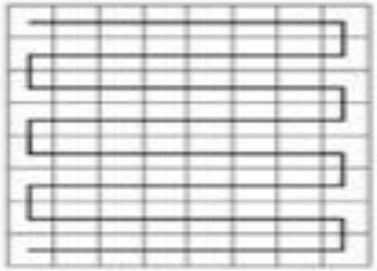
如图所示，我们将二进制编码的结果填写到空间中，当将空间划分为四块时候，编码的顺序分别是左下角00，左上角01，右下脚10，右上角11，也就是类似于Z的曲线。当我们递归的将各个块分解成更小的子块时，编码的顺序是自相似的（分形），每一个子快也形成Z曲线，这种类型的曲线被称为Peano空间填充曲线。

这种类型的空间填充曲线的优点是将二维空间转换成一维曲线（事实上是分形维），对大部分而言，编码相似的距离也相近， 但Peano空间填充曲线最大的缺点就是突变性，有些编码相邻但距离却相差很远，比如0111与1000，编码是相邻的，但距离相差很大。

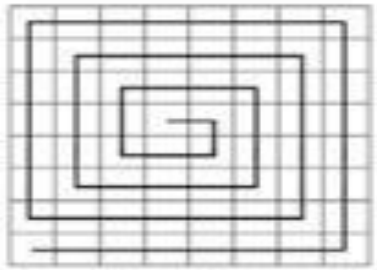
数学原理



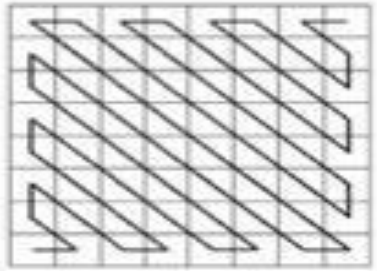
(a) row



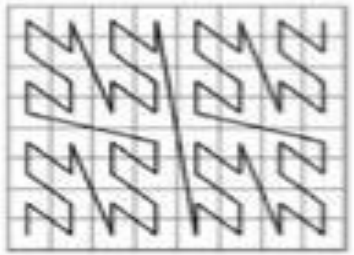
(b) row-prime



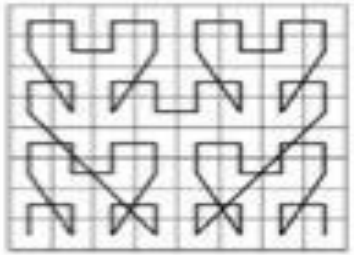
(c) spiral



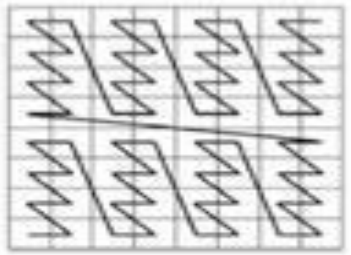
(d) Cantor



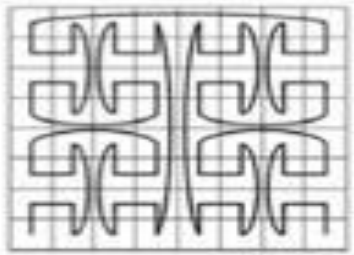
(e) Peano



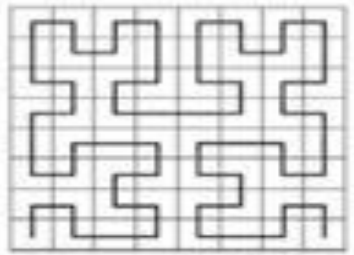
(f) U_Index



(g) Z_mirror



(h) Gray



(i) Hilbert

除Peano空间填充曲线外，还有很多空间填充曲线，如图所示，其中效果公认较好是Hilbert空间填充曲线，相较于Peano曲线而言，Hilbert曲线没有较大的突变。为什么GeoHash不选择Hilbert空间填充曲线呢？可能是Peano曲线思路以及计算上比较简单吧，事实上，Peano曲线就是一种四叉树线性编码方式。

数值索引

Lucene的倒排索引决定，索引内容是一个可排序的字符串，如果要查找一个数字，那么也需要将数字转成字符串。这样，检索一个数字是没问题的，如果需要搜索一个数值范围，怎么做呢？

要做范围查找，那么要求数字转成的字符串也是有序并单调的，但数字本身的位数是不一样的，最简单的版本就是前缀补0，比如 35, 234, 1 都补成 4 位，得到 0035, 0234, 0001，这样能保证：

```
数字(a) > 数字(b)    ===>    字符串(a) > 字符串(b)
```

这时候，查询应该用范围内的所有数值或查询，比如查询 [33, 36) 这个范围，对应的查询语法是：

```
33 || 34 || 35
```

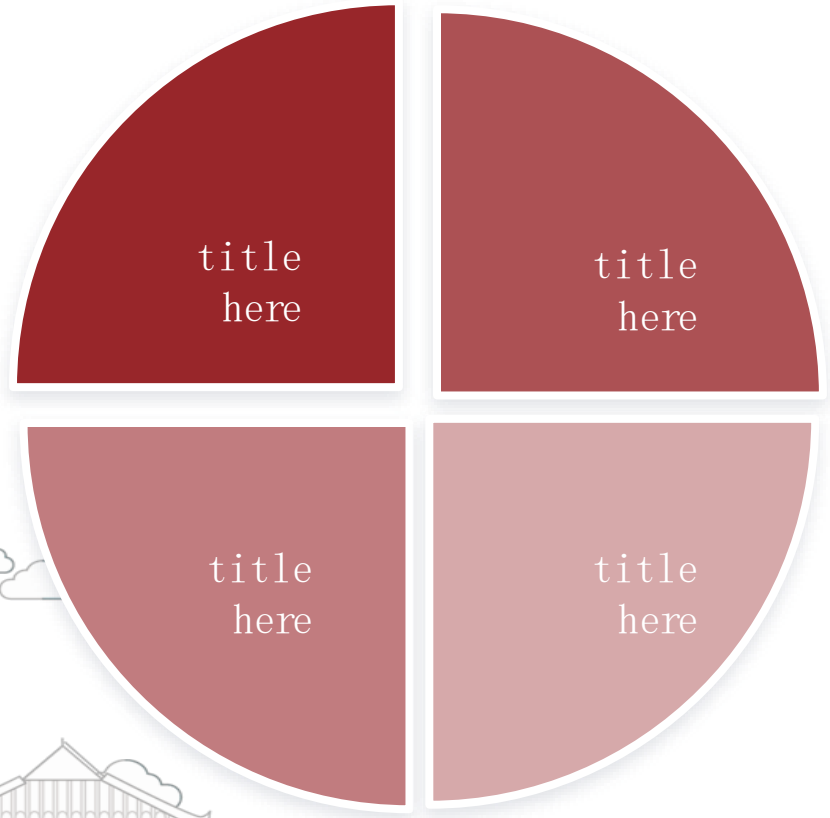
湖南大学
HUNAN UNIVERSITY

第三部分

优缺点

01

海量信息覆盖：搜索引擎通过爬取互联网上的内容，形成了庞大的索引库。无论是学术论文、新闻资讯还是日常生活中的疑问，用户只需在搜索框中输入关键词，即可获得海量信息。比如，当你想了解某个历史事件背后的故事时，搜索引擎能够快速找到相关资料并为你呈现。



02

个性化推荐：搜索引擎会根据用户的搜索历史、兴趣爱好等信息进行个性化推荐，提供更符合用户需求的内容。这种个性化服务使得用户能够更加高效地获取感兴趣的信息。

03

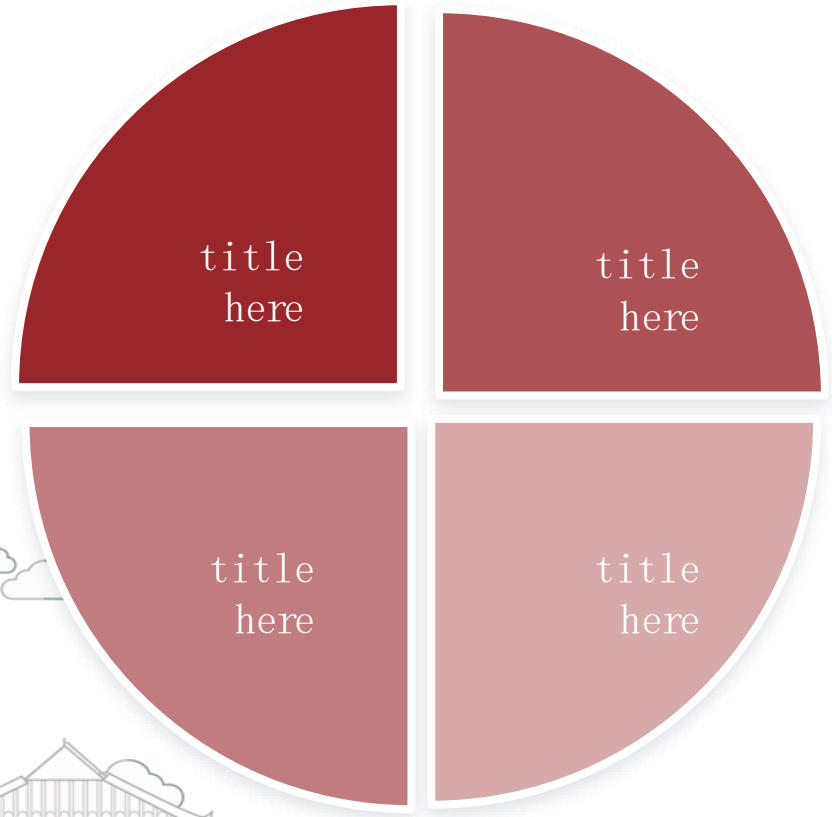
高效准确的搜索结果：搜索引擎通过复杂的算法对网页进行排序和匹配，能够根据用户输入的关键词提供相关性较高的搜索结果。

04

多样化的搜索方式：搜索引擎不仅支持文本搜索，还可以通过图片、视频等多种方式进行检索。用户可以通过上传图片或输入视频关键帧来寻找相关内容。

01

信息真实性难以保证：由于互联网上存在大量虚假、误导性的信息，搜索引擎无法完全保证检索结果的真实性。用户在使用搜索引擎时需要具备辨别能力，避免受到不准确或误导性信息的影响。



02

个人隐私泄露风险：搜索引擎在提供个性化推荐服务的同时，也会收集用户的搜索历史、地理位置等个人信息。

03

搜索结果过于广泛或局限：由于搜索引擎对关键词进行匹配时存在一定的主观性，有时会出现搜索结果过于广泛或局限的情况。

04

缺乏深度挖掘与综合分析能力：搜索引擎往往只能提供表面层次的信息，对于某些复杂问题或专业领域的深度挖掘与综合分析能力有限。

湖南大学
HUNAN UNIVERSITY

第四部分

优化

LSM思想

LSM (Log Structured Merge Tree)，最早是谷歌的 “BigTable” 提出来的，目标是保证写入性能，同时又能支持较高效率的检索，在很多 NoSQL 中都有使用，Lucene 也是使用 LSM 思想来写入。

为了保持磁盘的IO效率，lucene避免对索引文件的直接修改，所有的索引文件一旦生成，就是只读，不能被改变的。其操作过程如下：

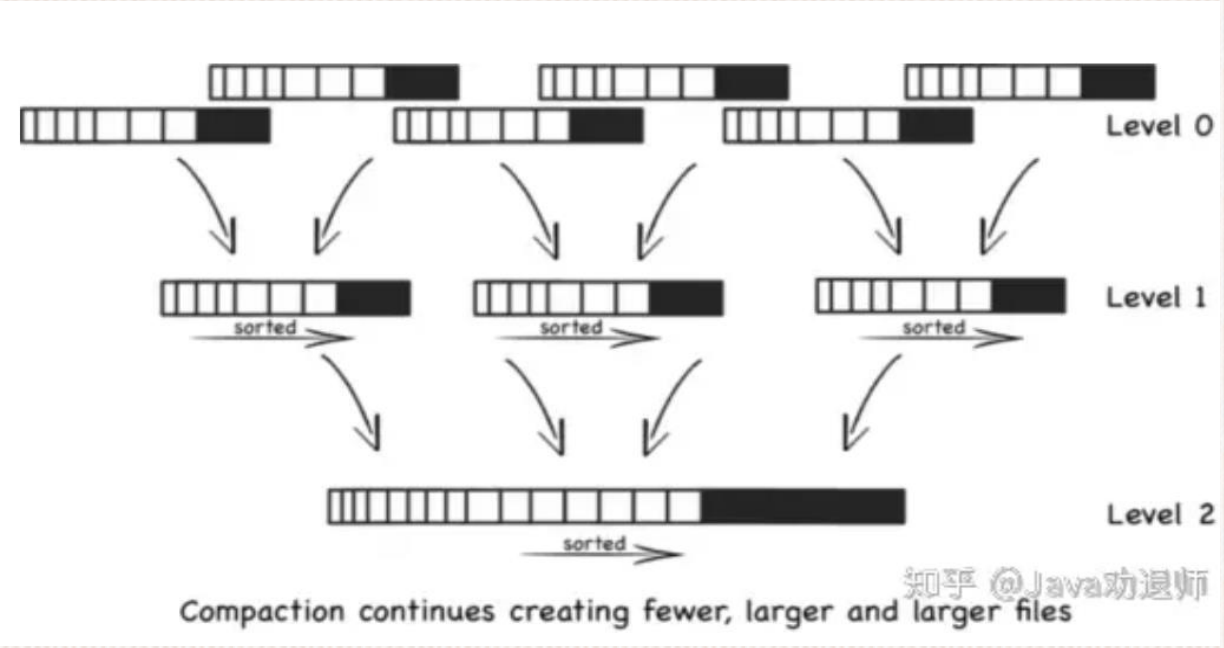
在内存中保存新增的索引, 内存缓存（也就是memtable）;
内存中的索引数量达到一定阈值时，触发写操作，将这部分数据批量写入新文件，我们称为segment；也就是 sstable文件
新增的segment生成后，不能被修改；
update操作和delete操作不会立即导致原有的数据被修改或者删除，会以append的方式存储update和delete标记;
最终得到大量的 segment，为了减少资源占用，也提高检索效率，会定期的将这些小的 segment 合并成大的 segment，由于map中的数据都是排好序的，所以合并也不会有随机写操作；
通过merge，还可以把update和delete操作真正生效，删除多余的数据，节省空间。

LSM思想

合并的过程：

Basic Compaction

每个文件固定N个数量，超过N，则新建一个sstable；当sstable数大于M，则合并一个大sstable；当大sstable的数量大于M，则合并一个更大的sstable文件，依次类推。



谢 谢 观 看

thank you for watching

参考文献

Csdn. MySQL全文索引及其优劣

郭心月. 运筹学(第四版)[M]. 清华大学出版社，2012.

知乎.深入理解搜索引擎原理,2020.