高效网页检索——倒排索引的原理与优化

郑铭涛-231250135

**引言**

网页检索技术是指通过搜索引擎等工具从互联网上的大量网页中快速、准确地查找所需信息的技术。随着互联网的快速发展，网络上的信息量急剧增加，如何有效地从海量信息中找到所需内容成为一个重要课题。网页检索技术的发展历程可以追溯到20世纪90年代初期，现代的网页检索技术已经发展得非常复杂和智能化。近年来，随着人工智能和自然语言处理技术的进步，网页检索技术也得到了进一步的发展。搜索引擎能够更好地理解用户的查询意图，提供更加精准和个性化的搜索结果。网页检索技术在信息获取、商业营销、科学研究等诸多领域发挥着重要作用，并随着技术的进步不断演进，未来有望变得更加智能和高效。本文我们将以搜索引擎为例，针对网络大数据资源，讨论网页检索的技术路线。

**正文**

1. **网页检索的基本原理**

索引本质上是基于目标内容而预先创建的一种数据结构，目的在于提高对目标内容的查询效率。例如，图书目录就是一种典型的索引形式。我们在翻看图书是，利用目录内容可以很容易地找到相应的章节及其页码。

支持网页检索的索引机制，利用的是一种称为“倒排索引”的技术手段。但是，要理解倒排索引，我们要先知道什么是“正排索引”。

1. **正向索引的基本原理**

正排索引即正向索引。从技术上而言，对索引的对象，首先会以文档的形式进行命名处理。每一个文档对应一个唯一的ID。文档的内容，则可以形式化地表达为一系列关键词的组合。经过这种形式上的统一处理，每一个文件就会分别对应一系列关键词的集合。在此基础上，还会对每一个关键词进行特定的属性描述，如该关键词在文档中出现的次数、在文档中每次出现的位置信息等。用一句话总结，就是“一个网页中对应多少关键词”。

正向索引的数据结构，是一种典型的键﹣值对存储形式的数据结构，这里文档ID 是键（key)，对应某一个文档的所有关键词及相关属性，就是文档的值（value)。

若我们要对某张网页中的内容建立正向索引，需要以下几个步骤：

第一，网页内容的预处理。

即过滤掉多余的HTML标签、代码等信息，仅仅保留有效的文本内容。经过预处理后的网页会变得更加方便后续操作。

第二，网络内容的分词。

对第一步预处理过的网页内容，使用分词算法，分隔出内容中的每个单词。这里我们需注意的是，我们是以英文网页为例，对于中文或其他的语种，都有相对应的分词算法。对于英文内容来说，可以以空格和标点符号作为分隔符，分隔出每个单词。当分词算法运行时，会顺序扫描整个文档，逐个分隔。为方便起见，我们将其都转为小写。在扫描的过程中，每分隔出一个单词，我们就将该单词出现的位置记录下来，将该单词出现次数的变量加一。

第三，针对单个网页的索引建立。

我们会使用MongoDB提供的接口，对分词后的结果进行统计，统计关键词的位置和出现次数。

第四，网页索引数据库的建立。

对于互联网环境下的众多网页，需要对每一个网页都进行上述处理，从而形成一个网页检索数据库。至此，我们已经完成了为一个网页建立正向索引的操作过程。

第五，基于正向索引的网页检索。

若搜索引擎使用的是正向索引机制，则搜索引擎需要对每一个被收录的网页，都分别构建一个正向索引。对于用户提交的基于关键词的检索要求，搜索引擎会在其管理的索引库中进行检索操作。

1. **倒排索引的基本原理**

搜索引擎的索引机制，是一种典型的"文档"到"单词"的映关系。基于这种数据结构的搜索引擎，需要遍历所有文档的索引文件，才能得出最后的检索结果。面对海量网页，正向索引耗时太长。

为了缩短检索的响应时间，我们对正向索引机制进行逻辑上的修订，即“单词”到“文档”的数据结构。这种机制被形象地称为“倒排索引”。一句话总结，就是“一个关键词在哪些网页中”。

如何理解倒排索引的速度优势？设文档总数为x，关键词的个数为y。在实际的互联网检索环境下，显然有x>>y，因为英语单词总数只有几十万，但互联网上的网页数以亿计，并且在不断增加。如果使用正向索引来检索某个关键词，必须遍历所有的文档，总时间是x的常数倍，这几乎是不可接受的。如果是倒排索引，因为单词是按序排好的，所以检索某个关键词的过程如同查字典，就相关文档检索而言，耗时相对小很多（一般前5个字母即可确定一个英语单词）。

倒排索引需要如下几个步骤：

1. 网页内容的预处理。

同正向索引。

1. 网页内容的分词。

同正向索引。

1. 针对单个网页的索引建立。

在倒排索引中，每一个单词都对应一个列表，记录了该词在哪些网页出现过，以及出现的次数和位置。对于本网页中的每一个词语，都执行一次上面的语句，以将词语的索引信息写入倒排索引中。

1. 网页索引数据库的建立。

对于每个网页，都执行上述步骤，将该网页的信息添加到倒排索引中。

1. 基于倒排索引的网页检索。

我们使用倒排索引来对关键词“南京大学 软件工程”进行检索。在索引过程中，首先对关键词“南京大学”和“软件工程”进行检索，然后合并检索到的结果，按照所含关键词的个数、每个关键词出现的次数、网站权重等信息，将其进行综合排序，以相关性强、重要性高的网页排在前面。

通过以上步骤，我们就完成了倒排索引。

1. **两种索引方式的时间复杂度比较**

正向索引：

假设我们有这样一个数据结构，它是一种“文档”到“单词”的映射：

DocID -> [term1, term2, term3, ..., termL]

根据上文介绍的过程，正向索引在最坏情况下，如果我们有N个文档，M个查询词，平均每个文档有L个词，那么时间复杂度为 O(N\*M\*L)，因为每个查询词都需要扫描所有文档，扫描文档要遍历所有的词。即，我们记：

Tforward=O(M\*N\*L)

倒排索引：

倒排索引的数据结构，它是“单词”到“文档”的映射：

termX -> [DocID1, DocID2, ..., DocIDK]

根据上文介绍的过程，查询时间复杂度主要取决于访问倒排列表的时间。如果我们有M个查询词，平均每个词映射了K个文件，那么时间复杂度为O(M\*K)，因为每个查询词都要遍历映射文档。即，我们记：

Tinverted=O(M\*K)

接下来我们用具体的数字来展现复杂度。我们取：

1. 文档数量 (N = 1,000,000)：

现代搜索引擎往往需要处理数百万甚至数十亿级别的文档。选择 1,000,000 个文档可以很好地模拟这种大规模场景。

1. 每个文档的词项数量 (L = 1000)：

假设每个文档包含 1000 个词项，是为了反映一般文档的长度，例如新闻文章或博客文章。这是一个较为常见的中等文档长度。

1. 每个词项的平均文档数 (A = 1000)：

这个值表示每个词项平均出现在 1000 个文档中。高频词会出现在大量文档中，低频词会出现在少数文档中，选择这个值反映了一个较为均匀的分布。

1. 查询词 （M=3）。

那么：

Tforward=3\*10^9

Tinvered=3\*10^3

倒排索引的查询效率比正向索引高了 10^6 倍。倒排索引通过将查询词直接映射到文档ID列表，避免了对所有文档的遍历，从而大大提高了查询效率。通过上述公式和数据可以清楚地看到，倒排索引在处理大规模数据时具有显著的性能优势。

1. **倒排索引的代价**

虽然倒排索引在查询效率上具有显著优势，但其实现和维护也存在一定的代价。主要的代价包括以下几个方面：

1. 储存空间大。

倒排索引需要额外的存储空间来保存每个词项对应的文档ID列表。假设有 N 个文档，每个文档平均有 L 个词，总共 T 个词项。存储倒排索引的空间复杂度大致为 O(T \* A)，其中 A 是每个词项平均包含的文档数量。假设 N = 1,000,000，每个文档 L = 1000 个词，T = 10,000,000 个不同词项，平均每个词项出现 A = 1000 个文档。则存储倒排索引需要的空间约为：

T\*A=10^10

如果每个文档ID占用 4 字节，则总存储空间为：

4\*10^10 bytes = 40 GB

第二，构建倒排索引的时间复杂度相对较高，特别是对于大规模数据集，索引构建可能需要很长时间。

第三，倒排索引的动态更新复杂度较高，特别是在面对频繁的文档插入、删除和修改操作时。

1. **针对倒排索引缺点的解决方法**

第一，针对储存空间大的缺点，我们引入三种技术：

1. Gamma 编码

Gamma 编码是一种用于整数编码的无损压缩技术，尤其适用于压缩倒排列表中的文档ID。Gamma 编码将一个正整数编码为两部分：前缀和后缀。

前缀码：首先将要编码的整数 n 转换为其二进制形式。如果这个二进制数有 k 位，前缀码就是 k−1 个 0 后跟一个 1。

尾码：尾码是整数n 的二进制表示，去掉最高位的 1。

例如：13d = 1011b。所以，EGC(13d) = 000 1011b。Gamma 编码的优点是适用于较小的正整数，能够压缩数据，减少存储空间；缺点是对于较大的整数，编码长度会增加，且编码和解码过程可能较复杂。

2. VByte 编码

VByte（Variable Byte）编码是一种变长字节编码方法，适用于存储整数。它通过使用多个字节来表示一个整数，最高位作为标志位，指示当前字节是否为整数的最后一个字节。

格式：每个字节的最高位（MSB）用作继续位（0表示这是最后一个字节，1表示后面还有字节），其余7位用来存储整数的值。

举个例子，假设要编码整数300：

300的二进制是“100101100”。用7位块表示：10 和 0010110。加上标志位：10000010（0x82）和01010110（0x56）。编码结果是两个字节：0x56 0x82。

优点是变长编码，适应数据的大小，节省空间，编码和解码过程相对简单，效率较高。缺点是对于小整数，可能会有一些空间浪费。

3. 布尔位图

布尔位图（Bitmap）是一种特殊的压缩技术，适用于高频词的表示。它使用位图表示法，即每个文档对应一个位，表示该文档是否包含特定词项。

举个例子，假设有5个文档，词项“example”在第1、3、4个文档中出现，则位图表示为：10110。

优点是高效存储和快速访问，查询某个词项是否在特定文档中非常快。缺点是对于低频词项，可能会浪费大量空间，适用于高频词项，低频词项使用位图可能效率不高。

1. 针对动态更新复杂度较高的缺点，我们引入三种方法：
2. 批量更新

批量更新是指定期批量更新索引，而不是实时更新。这样可以减少频繁更新带来的开销，优化系统性能。

2. 分段索引

分段索引是将倒排索引分为多个段，每个段独立更新和合并。独立更新，每个段独立更新，减少了整体索引的锁争用。并行处理，多个段可以并行处理，提高了更新和查询的效率。便于合并，独立的段可以定期合并，优化查询性能。

3. 增量索引

增量索引是维护一个小的增量索引，用于快速响应插入和删除操作，定期与主索引合并。快速响应，增量索引较小，插入和删除操作可以快速完成。减少主索引压力，主索引定期更新，减少了频繁更新带来的压力。灵活性高，增量索引可以灵活调整，适应不同的更新频率和查询需求。

**结论**

倒排索引作为网页检索技术中的核心技术，极大地提升了搜索引擎的效率和准确性。通过将文档中的每个词汇与其所在的文档位置进行索引，倒排索引能够快速定位和检索相关信息，满足用户的搜索需求。随着互联网信息量的不断增加，倒排索引在处理大规模数据方面展现了其强大的优势。未来，随着人工智能和大数据技术的发展，倒排索引技术将进一步优化和创新，提供更加精准和智能的检索服务，为用户带来更优质的搜索体验。

**参考文献**

1. 窦万春，杨剑，代飞. 大数据关键技术与引用创新（第二版） [M] 南京师范大学出版社，2023
2. Ian H. Witten, Alistair Moffat, Timothy C. Bell. 千兆字节管理：文档与图像的压缩和索引 [M]. Morgan Kaufmann, 1999.
3. David Salomon, Giovanni Motta, David Bryant. 数据压缩：完整指南 [M]. Springer, 2007.
4. W. Bruce Croft, Donald Metzler, Trevor Strohman. 搜索引擎：实践中的信息检索 [M]. Addison-Wesley, 2009.