基于 Hadoop 研发新药文献检索系统

梁永浩 陆 涛 赵鸿萍

（中国药科大学理学院 南京 211198）

〔摘要〕 为提高新药文献检索的效率，研发基于 Hadoop 的分布式新药文献检索系统。系统包括全文检索和化学结构式检索两大部分，其中全文检索基于关键技术 Lucene 和 Hadoop 实现；化学结构式检索则使用 Hbase 存储结构式的 SMILES 码和连接表，基于图同构算法 VF2 对结构式进行匹配。

〔关键词〕 Hadoop；Lucene；分布式检索；结构式检索

〔中图分类号〕R－056 〔文献标识码〕A 〔DOI〕10.3969/j.issn.1673－6036.2016.05.017

Developing New Medicine Literature Retrieval System Based on Hadoop *LIANG Yong－hao, LU Tao, ZHAO Hong－ping, College of Science, China Pharmaceutical University, Nanjing 211198, China*

〔**Abstract**〕 In order to improve the efficiency of retrieving literature on new medicine, a distributed new medicine literature retrieval system is developed based on Hadoop. This system contains two parts: full－text retrieval and chemical structural formula retrieval. The former is implemented based on the key technologies of Lucene and Hadoop. The latter uses Hbase to store structured SMILES and connection tables and matches the structural formula based on graph isomorphism algorithm VF2.

〔**Keywords**〕 Hadoop; Lucene; Distributed retrieval; Structure retrieval

1 引言

新药研发是一个高技术、高投入、高风险、竞争激烈的过程，每一个研发公司都需要及时了解市场同类药品研发的现状。但国内目前尚无专门进入药审平台前的在研新药文献数据库可以检索，同时，国内的各个文献检索平台均未提供根据化学结构式检索相关文献的功能。为帮助国内新药研发人员及时、准确地掌握国内竞争者的研究现状，本文尝试基于 Hadoop 和 Lucene，研发一个分布式新药文献检索系统。

〔修回日期〕 2016－02－25

〔作者简介〕 梁永浩，硕士研究生；通讯作者：陆涛，教授。

2 关键技术

2.1 Hadoop

Hadoop 是 Apache 基金会的一个用于运行大规模集群上的分布式应用的开源框架，它封装了分布式系统的实现细节，使得用户可以方便地开发分布式应用[2]，其两大核心组件是 Hadoop Distributed File System（HDFS）和 MapReduce[3]。HDFS 即 Hadoop 分布式文件系统，是整个系统的底层，负责存储和管理 Hadoop 集群中所有节点上的文件，隐藏下层的负载均衡、冗余复制等文件管理细节，为上层提供统一的文件系统应用接口。对客户机而言，HDFS 就像一个本地文件系统一样，可以直接进行读取、创建、删除等文件操作而不用关心文件所存储的具体物理位置。具有高容错性的特点，可

稳定部署在廉价的硬件集群上。MapReduce 是 Hadoop 的并行计算框架，负责进行任务调度和管理，它最大的优点是容易扩展到多个计算节点上处理数据。MapReduce 借鉴函数式编程的思想，将计算任务分为 Map 和 Reduce 两个主要步骤。Map 接受一组数据，将其转化为 key - value 列表装入 Mapper，Reduce 则处理来自 Mapper 的所有输出并给出最终结果。

2.2 Lucene

2.2.1 概述 Lucene 也是 Apache 基金会下的一个开源项目[4]，是高性能的全文搜索引擎工具包。它不是一个完整的应用程序，主要实现索引和检索部分的核心功能，为各种应用程序提供接口。包括 Wikipedia、Eclipse、AOL、Netflix 在内的很多著名公司和产品都使用 Lucene[5]。Lucene 的主要优势在于其功能强大，准确而高效的搜索算法：能处理超过 150GB/h 的数据，而本身只占用 1MB 堆内存；能返回与查询相关性较高的搜索结果，支持短语查询、通配符查询、邻近查询等多种查询类型；支持针对文档标题、作者、内容等具体域的搜索，按任何字段排序；支持插件化的排序模型，包括向量空间模型和 Okapi BM25 排序算法。

2.2.2 主要包及其组织关系 Lucene 源码包括 7 个子包，各包之间的组织关系，见图 1[6]。在核心组件中，包 Document 提供将要索引的文档封装起来所需要的类，包括 Document 和 Field 等，把每一个文档最终封装成一个 Document 对象，便于进行文档操作[7]。包 Analysis 用于对文档进行解析。Lucene 使用倒排索引作为其索引的数据结构，即使用单词作为索引，文档 ID 作为存储内容，以便于根据检索词迅速找到对应的文档，这就需要在索引之前，将文本信息切分为信息单元（即 token）。由于 Lucene 默认的 StandardAnalyzer 对中文分词效果不佳，系统使用 MMSeg4j[8] 进行分词。包 Index 提供一些类协助创建索引，或对已经创建好的索引进行更新。其中有两个重要的类：IndexWriter 和 IndexReader，其中 IndexWriter 用于创建索引并添加文档到索引中，IndexReader 用于读取索引中的文档。包 Search 提供在搜索的过程中所需要的类。如 IndexSearcher 和 Hits，IndexSearcher 定义在指定的索引上进行搜索的方法，Hits 用于保存搜索得到的结果。剩下的其他包中，QueryParser 用于进行查询语句解析，实现查询关键词运算。Store 用于对索引文件的存储管理。Util 则包含一些公共的工具类。

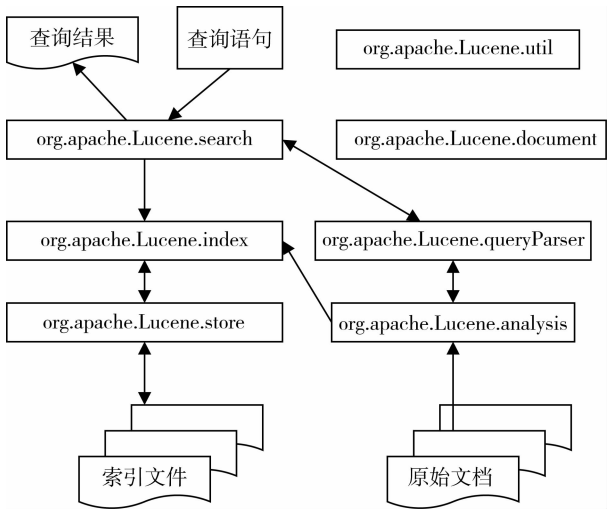

图 1 Lucene 主要包及其组织关系

3 全文检索实现

3.1 结构设计

由于硬件资源的限制，系统的实际开发只能在一台物理机上。为模拟分布式环境，使用虚拟机软件模拟 3 台服务器，分别名为 Master、Slaver1、Slaver2，其中 Master 用作 NameNode，其余两台用作 DataNode。由于 Hadoop 系统本身的高扩展性和可移植性，在条件允许时可轻易移植到真正的分布式集群中。由于 Lucene 库中，包 Store 里自带对索引文件的创建和存储的实现，不需要使用数据库。在化学结构式检索部分，需要使用 Hbase 存储结构式相关信息。整个系统的结构设计，见图 2。

3.2 开发环境

整个系统运行于 CentOS 操作系统，使用 Eclipse 开发并使用 Tomcat 作为 Web 服务器。具体软件环境，见表 1。

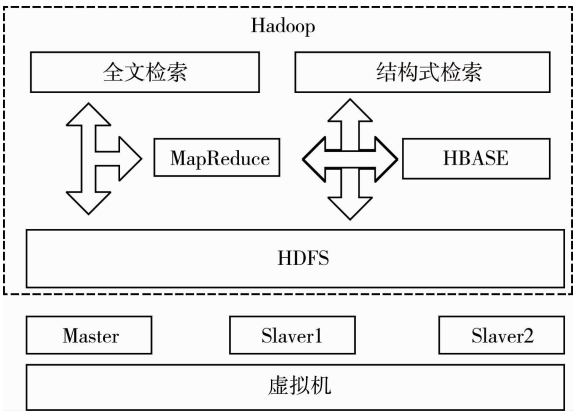

图2 系统结构设计

表1 软件开发环境

软件名称	版本
Hadoop	hadoop - 1. 2. 1
Hbase	hbase - 0. 98. 11 - hadoop1
Eclipse	eclipse - jee - luna - SR2 - linux - gtk - x86_ 64
Tomcat	apache - tomcat - 6. 0. 43
JDK	JDK 1. 7. 0_ 79 - b15
Lucene	lucene - 4. 10. 4

3.3 创建索引

在建立索引之前，首先要创建一个 Document 对象，确定 Document 中的标题、内容、作者、时间等各个域，然后使用 IndexWriter 接收新加入的文档，同时用 MMSegAnalyzer 对文档内容进行分词处理，创建倒排索引，写入索引文件中。在单机系统中创建索引较为简单，但在 Hadoop 中，Lucene 需要进行随机读写创建索引，而 HDFS 为了保证大规模数据吞吐时依然有良好的性能，只支持随机读，批量写。因此，直接将索引建立在 HDFS 上是不可行的[9]。系统采用与 Nutch 相同的解决方案，将索引文件先缓存于内存中，再批量上传至 HDFS。创建索引的部分关键代码如下[10]：

```
//前面部分代码为创建并配置相关的几个类
RAMDirectory ramdir = new RAMDirectory ( ); //创建缓存目录
IndexWriter indexWriter = new IndexWriter ( ramdir, new MMSegAnalyzer ( ), true); // 创建 IndexWriter 生成索引，指定使用 MMSegAnalyzer 对文档分词。
FSDataInputStream filereader = dfs. open ( new Path ( dfs. getWorkingDirectory ( ) + File_ DIR));
while ( (row = reader. readLine ( )) ! = null) {
String Arow [ ] = row. split ( " "); //读取用空格分隔的各个域
Document document = new Document ( );
document. add ( new Field ( " date", Arow [ 0 ], Field. Store. YES, Field. Index. ANALYZED));
……//将各个域依次添加到文档对象中。
indexWriter. addDocument ( document); }
```

至此，在内存中生成了索引，接着将索引写入 HDFS 中：

```
//前面代码为依次读取内存文件并转化为字节流。
FSDataOutputStream fs = dfs. create ( new Path ( dfs. getWorkingDirectory ( ) + Index_ DIR + fileList [ i] . trim ( )), true); }
```

3.4 对已索引的内容进行检索

HDFS 支持随机读，可直接读取 HDFS 上的索引进行检索。检索时先构造 Directory 对象，代表索引存储的位置。用 Directory 构造 IndexSearcher 打开索引文件。再用 Term 对象封装查询，提交给 IndexSearcher 的 Search 方法[11]，得到查询结果集 Hits。部分关键代码如下：

```
FileSystem dfs = FileSystem. get ( config); //获取 HDFS 配置。
Path pth = new Path ( dfs. getWorkingDirectory ( ) + Index_ DIR);
FileSystemDirectory directory = new FileSystemDirectory ( dfs, pth, false, config);
IndexSearcher searcher = new IndexSearcher ( directory);
Term term = new Term ( " contents", queryStr. toLowerCase ( ));
TermQuery luceneQuery = new TermQuery ( term);
Hits hits = searcher. search ( luceneQuery);
for ( int i = 0; i < hits. length ( ); i + + ) {
Document doc = hits. doc ( i); //依次读取 hits 中的内容。
```

4 结构式检索

4.1 化学结构式表示

4.1.1 SMILES 码与连接表 结构检索最基本的

问题就是化合物在计算机的表示方法。不同的表示方法可能存在着不能唯一表达一个结构式、丢失结构式立体信息、不便于进行检索处理等问题，需要根据检索方法选择合适的方案。化学结构式在计算机中编码描述的方式包括系统命名法、碎片码、线性码、连接表和拓扑指数码等几大类[12]。系统使用 SMILES 码和连接表。（1）SMILES 码。SMILES 码是一种线性码，是将原子按照一定的规则逐个编码的方法。SMILES 能唯一对应一个结构式，但同一个结构有可能生成不同的 SMILES，可以通过 Morgan 算法或制定规则保证一一对应。对于立体异构（Z，E 或者 R，S）的分子则会丢失立体信息，只能进行精确检索。（2）连接表。将化学结构视为计算机中的图数据结构，直接记录化合物中的原子性质及其拓扑关系，能完整记录化学物质的各方面信息，且便于进行各种运算，是目前最为常用的计算机存储化学结构的方式。连接表分为冗余连接表和非冗余连接表两种，在冗余连接表中，每个化学键连接两个原子，在连接表中被重复描述两次，为节省空间，可使用单向连接表将其压缩。

4.1.2 对原子进行编码 使用不同的编码方式描述化学结构前，必须对原子按照一定的顺序进行编码，不同的编号顺序可能导致同一个化学结构产生不同的编码。本研究使用 Morgan 算法[13]对原子进行编号，根据编号生成唯一编码。Morgan 算法是根据原子的连接度由大到小进行编号，连接度计算方法具体过程[14]：（1）计算每个原子与其他原子相连的化学键的数量，作为其初始连接度。（2）计算与每个原子相连的其他原子的初始连接度之和，作为该原子的拓展连接度。（3）重复步骤（2）使得化学结构式中不同连接度的原子数量等于结构式中原子的数量（即能将每个原子区分开来）或者该数量无法再增加为止。（4）以结构式中连接度最大的原子编号为 1，与其相连的原子中，连接度最大的为 2，以广度优先的方式遍历。权重相等时，按原子序数从大到小进行编号。

4.2 结构式匹配算法

结构检索时需要使用用户提交的查询结构与数据库中的化学结构进行图形匹配。图形匹配的主流算法是用回溯算法进行树搜索，使用一些启发式的方法进行剪枝缩小搜索空间。包括最早的 Ullmann[15]，以及之后的 SD[16]、VF[17]和 VF2[18-19]都是这类算法。其中，VF2 对各类型图的性能都是最好的[20]。VF2 算法的匹配过程可以使用状态空间表示法进行描述。匹配过程的每个状态 s 可以关联到一个局部映射 M（s），M（s）为映射函数 M 的一个子集。一个局部映射能唯一地标识 G_1 和 G_2 的两个子图，记为 G_1（s）和 G_2（s），表示 G_1 和 G_2 的节点在 M（s）中，有分支将节点相连。将 M（s）到 N_1 和 N_2 的投影分别记为 M_1（s）和 M_2（s），G_1（s）和 G_2（s）之中的分支分别记为 B_1（s）和 B_2（s）。VF2 匹配算法的伪代码描述如下：

```
Match（s）{
    INPUT：中间状态 s；初始状态 s0，M（s0）=ϕ
    OUTPUT：两图之间的映射
    IF M（s）包含 G2 的所有节点 {
        OUTPUT M（s）}
    ELSE {
        计算 P（s）作为 M（s）的候选配对
        FOREACH（n，m）IN P（s）{
            IF F（s，n，m）{
            将(n，m)添加至 M（s），计算状态 s′
            Match（s′）  }}
        重置数据结构
}}
```

其中 F（s，n，m）用于修剪搜索树。当它的值为真时，表示可以将（n，m）添加至 s 中变成 s′，s 属于局部同构时，s′也为同构。

4.3 结构绘制

用户进行结构式检索时，需要使用编辑器画出结构式提交给服务器，系统使用 JSME[21]实现结构绘制。用户使用 JSME 完成编辑后，可导出结构式的经过标准化的 SMILES 编码及 MOL 文件。MOL 文件是用于表示分子结构的文件，由头、块、连接表 3 部分组成，其中头和块包含一些用户信息以及分

子的原子数和化学键数、同位素、电荷数等信息。此外，开发过程还使用化学开发工具（Chemistry Development Kit，CDK）[22-23]。CDK 是 JAVA 开发的面向化学信息学、生物信息学的开源工具包，提供对原子、化学键、元素等各种化学对象的封装，以及对多种格式的化学文件解析、SMILES 的分析及产生、结构式图形的生成等各方面的支持，是开发许多化学相关开源软件的基础工具。

4.4 Hbase 数据库设计

Hbase 是一个基于 Hadoop 的高性能、高可靠性、便于横向扩展的分布式数据库。与传统的关系型数据库不同，Hbase 为提高性能，放弃了严格的关系模式束缚，仅能通过主键和主键的范围检索数据，不支持多表连接的复杂操作，使其达到一个表能包含上亿行数据的规模。Row Key 为自动编号的主键，Timestamp 为 Hbase 自动记录的每次操作的时间戳。列族 Structure 用于存储分子的结构信息，包括原子数、化学键数、SMILES 编码和连接表 4 个子列。列族 Info 用于存储其他信息，如该化学结构所对应的文档 Document。

4.5 MapReduce 实现

MapReduce 程序通过操作键/值对处理数据[24]，一般形式为：

map：(K1，V1) → list：(K2，V2)

reduce：(K2，list (V2)) → list：(K3，V3)

为保证键/值对能在集群中传递，为相关数据类型实现了 WritableComparable < T > 接口，使其能够序列化。序列化的类可以在集群中传输，需要使用时则反序列化读取。一个 MapReduce 程序至少包括一个 Mapper，一个 Reducer，以及调用并配置 Mapper 和 Reducer 的 MyJob。实现 Mapper 需要继承 Mapper 抽象类并实现 map（）方法。该方法接收键/值对（Text，Molecule），分别表示分子 ID 和对应 Molecule 对象，生成初步筛选后的（Text，Molecule）列表，OutputCollector 接受这个映射过程的输出。输出结果使用 HashPartitioner 分配给 Reducer。Mapper 类代码如下所示：

```
public static class Map extends MapteduceBase implements Mapper < Text, Molecule, Text, Molecule > {
public void map (Text moleculeId, Molecule molecule, OutputCollector < Text, Molecule > output, Reporter reporter) throws IOException {……} }
```

实现 Reducer 需要继承 Reducer 抽象类并实现 reduce（）方法。Reducer 任务接受来自 Mapper 的输出时，会自动按照键对输入数据进行排序，将相同键的值归并。然后调用 reduce（），使用 VF2 算法进行匹配，生成匹配完成的（Text，Molecule）列表。Reducer 类代码如下所示：

```
public static class Reduce extends MapteduceBase implements Reducer < Text, Molecule, Text, Molecule > {
public void reduce (Text moleculeId, Iterator < Molecule > molecules, OutputCollector < Text, Molecule > output, Reporter reporter ) throws IOException {……} }
```

MyJob 用于定义一个完整的 MapReduce 作业。首先实例化一个 JobConfig 对象，依次调用 set 方法配置作业名、Mapper、Reducer、输入格式、输出格式等，最后将该对象传递给 JobClient. runJob（）以启动 MapReduce 作业。部分代码如下：

```
Configuration conf = getConf ();
JobConf job = new JobConf (conf, MyJob. class);
job. setJobName ("MyJob");
……//进行配置
JobClient. runJob (job);
```

整个检索过程的实现简述如下：用户使用 JSME 编辑结构式，选择检索方式（精确匹配或子结构匹配）。若为精确匹配，直接使用 JSME 生成的 SMILES 码，对 Hbase 数据库中分子的 SMILES 码进行搜索。若为子结构匹配，则使用 CDK 解析 JSME 返回的 MOL 文件，将其封装为一个 Molecule 对象，使用 Morgan 算法对其中的连接表进行标准化。将 Molecule 序列化之后传入 Mapper，Mapper 读取 Molecule 中的原子种类、数目和化学键数目，扫描 Hbase，对结构式进行初步筛选，过滤明显无法匹配的结构式，生成可能匹配的结构式列表。将列表通过 HashPartitioner 划分，发送给各个 Reducer，Reducer 使用 VF2 算法对结构式进行匹配运算，将结果合并，返回。

5 结语

本文基于 Hadoop 和 Lucene 开发了一个分布式新药文献检索的原型系统，该系统除具备传统检索功能外，还支持化学结构式检索。使用系统检索时，可以在网页上用平台提供的工具画出结构式，进而检索出与该结构式相关的文献。系统开发于 Hadoop 平台之上，具有高效性、高可靠性和高扩展性的特点，与传统的单机检索方式相比，可提高检索效率。特别是结构式检索部分，由于子图同构为 NP 问题，数据量较大时，分布式检索比单机检索优势明显。论文的研究结果可以为文献数据库研发人员提供借鉴，随着数据库内容日趋完善，为新药研发人员及时了解竞争者情报提供帮助。

参考文献

1 阿丽塔，许培扬，孙灵芝. 药物研发过程中药学信息的利用［J］. 中国药房，2011，22（5）：466－648.

2 Apache. Hadoop［EB/OL］.［2015－09－02］. http：//wiki. apache. org/hadoop/.

3 王俊生，施运梅，张仰森. 基于 Hadoop 的分布式搜索引擎关键技术［J］. 北京信息科技大学学报（自然科学版），2011，26（4）：53－56.

4 Apache. Lucene Core［EB/OL］.［2015－09－02］. http：//lucene. apache. org/core/.

5 McCandless M，Hatcher E，Gospodnetic O. Lucene in Action：Covers Apache Lucene 3.0［M］. New Jersey：Manning Publications Co.，2010.

6 王振风. 基于 Lucene 的分布式全文检索技术的研究与应用［D］. 上海：东华大学，2015.

7 李永春，丁华福. Lucene 的全文检索的研究与应用［J］. 计算机技术与发展，2010，（2）：12－15.

8 Chenlb. MMSeg4j［EB/OL］.［2015－09－02］. http：//code. google. com/p/mmseg4j.

9 郭建荣. 基于分布式计算的全文检索关键技术研究［D］. 北京：北京邮电大学，2014.

10 Kashyap，Santoki. Indexing and Searching on a Hadoop Distributed File System［EB/OL］.［2015－09－02］. http：//www. drdobbs. com/parallel/indexing－and－searching－on－a－hadoop－distr/226300241.

11 Grant，Ingersoll. Next－generation Search and Analytics with Apache Lucene and Solr 4 Use search engine technology to build fast，efficient，and scalable data－driven applications［EB/OL］.［2015－09－02］. http：//www. ibm. com/developerworks/library/j－solr－lucene/index. html.

12 潘凯. ChemDataBase 数据库中化学分子子结构检索方法的设计与实现［D］. 兰州：兰州大学，2009.

13 Morgan H L. The Generation of a Unique Machine Description for Chemical Structures－A Technique Developed at Chemical Abstracts Service［J］. Journal of Chemical Documentation，1965，5（2）：107－113.

14 许禄，胡昌玉，许志宏. 应用化学图论［M］. 北京：科学出版社，2000.

15 Ullmann J R. An Algorithm for Subgraph Isomorphism［J］. Journal of the ACM，1976，23（1）：31－42.

16 Schmidt D C，Druffel L E. A Fast Backtracking Algorithm to Test Directed Graphs for Isomorphism Using Distance Matrices［J］. Journal of the ACM，1976，23（3）：433－445.

17 Cordella L P，Foggia P，i C，et al. Performance Evaluation of the VF Graph Matching Algorithm［C］. International Conference on Image Analysis and Processing，IEEE，1999：1172－1177.

18 Cordella L P，Foggia P，Sansone C，et al. An improved algorithm for matching large graphs［C］. Italy：3rd IAPR－TC15 workshop on graph－based representations in pattern recognition，2001：149－159.

19 Cordella L P，Foggia P，Sansone C，et al. A（sub）Graph isomorphism Algorithm for Matching Large Graphs［J］. IEEE Transactions on Pattern Analysis and Machine Intelligence，2004，26（10）：1367－1372.

20 Foggia P，Sansone C，Vento M. A Performance Comparison of Five Algorithms for Graph Isomorphism［C］. Proceedings of the 3rd IAPR TC－15 Workshop on Graph－based Representations in Pattern Recognition，2001：188－199.

21 Bienfait B，Ertl P. JSME：a free molecule editor in JavaScript［J］. J Cheminformatics，2013，（5）：24.

22 Steinbeck C，Han Y，Kuhn S，et al. The Chemistry Development Kit（CDK）：An open－source Java library for chemo－and bioinformatics［J］. Journal of Chemical Information and Computer Sciences，2003，43（2）：493－500.

23 Steinbeck C，Hoppe C，Kuhn S，et al. Recent Developments of the Chemistry Development Kit（CDK）－an open－source java library for chemo－and bioinformatics［J］. Current Pharmaceutical Design，2006，12（17）：2111－2120.

24 Lam C. Hadoop in Action［M］. New Jersey Virginia Manning Publications Co，2010.