分布式环境下的文档相似度研究与实现

赵华茗
(中国科学院国家科学图书馆　北京 100190)

【摘要】针对传统的相似度计算方法在海量信息处理过程中暴露出的数据处理规模限制和性能不足等方面的瓶颈问题,以非结构化文档为研究对象,提出一种基于 Hadoop 分布式环境,结合 Hive 数据处理平台和 PostgreSQL 关系型数据库的文档相似度计算方法,并给出关键技术思路、具体实现步骤和实证研究,通过研究证明 Hive SQL 语言可有效简化分布式数据处理的复杂性,但实时性有待改进。

【关键词】Hadoop　Hive　相似度　非结构化
【分类号】TP393

Research and Implementation of Textual Similarity in Distributed Environment

Zhao Huaming
(National Science Library, Chinese Academy of Sciences, Beijing 100190, China)

【Abstract】Aiming at the performance issue and limitation on data set size in the process of mass-data mining of traditional similarity algorithm, this paper takes unstructured textual data as research subject and introduces the method of Hadoop distributed textual similarity algorithm, which combines Hive data mining platform with PostgreSQL RMDB, and describes the basic technical ideas, implementations and the empirical research in details. The testing result shows that Hive SQL can effectively simplify the complexity of distributed data mining but its real-time performance should be improved.

【Keywords】Hadoop　Hive　Similarity　Unstructured

1　引　言

文档相似度计算是一种常见的信息分类和信息挖掘方法,在信息检索、数据挖掘、机器翻译、相似性检测等领域有着广泛的应用[1-3]。随着互联网的发展,数字信息成几何级数增长。在海量信息处理过程中,和很多数据挖掘算法一样,传统的相似度计算方法,暴露出数据处理规模及性能等方面的不足,这使得如何将传统的信息挖掘方法分解为可分布式并行的信息挖掘计算方法成为解决问题的关键。洪毅虹[4]提出了基于 MapReduce[5] 架构的文档相似度计算方法和思路,通过设计 Map 函数和 Reduce 函数,实现相似度计算的并行化,但 MapReduce 编程过程较复杂,也不利于算法的维护和复用。本文以非结构化文档为研究对象,利用 Hive[6] 数据处理平台和关系型数据库在数据分析挖掘过程中的优势,提出一种基于分布式环境的文档相似度计算方法,并结合开源软件及开发框架,详细阐述技术实现思路和关键点。

收稿日期: 2011-04-29
收修改稿日期: 2011-06-08

2　海量的文档相似度计算思路和关键技术

2.1　总体策略

分布式环境下，实现非结构化文档相似度计算的总体策略包括文档预处理和相似度计算两部分。具体实现步骤为：

(1) 定期将从互联网采集下来的非结构化文档根据计算需要做结构化预处理，形成具有特征词的结构化新文档；

(2) 导入分布式 Hive 平台，结合经典的相似性函数夹角余弦法和 Hive SQL 语言，与所有老文档依次并行执行相似度计算的遍历过程；

(3) 将计算结果保存到数据库当中。

2.2　关键技术

Hive 是本文在分布式环境下实现文档相似度计算的关键技术方法。Hive 平台是基于 Hadoop 构建的一套数据仓库分析系统，立足于 MapReduce 并行计算框架，提供类 SQL 功能的 Hive SQL 语言来分析存储在 Hadoop 分布式文件系统中的数据，有效简化分布式数据处理的复杂性，成为实用、高效的海量数据处理平台[7,8]。其系统架构[7]如图 1 所示：

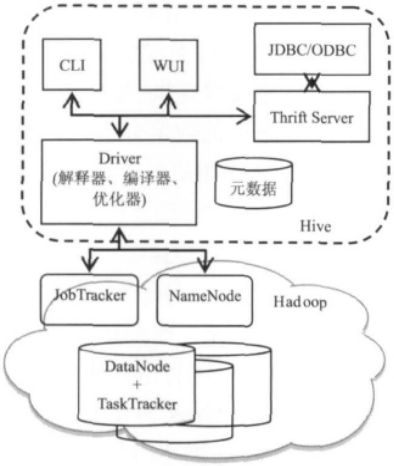

图 1　Hive 系统架构图[7]

Hive 将结构化的数据存储在数据仓库中，通过自己的 SQL 去查询分析需要的内容，这套 SQL 简称 Hive SQL。Hive 解释器、编译器、优化器完成 HQL (Hibernate Query Language) 查询语句从词法分析、语法分析、编译、优化以及查询计划的生成。生成的查询计划存储在 HDFS(Hadoop Distributed File System) 中，并在随后用 MapReduce 调用执行。但是包含“* ”的查询，Hive 不会生成 MapRedcue 任务，因此程序开发中应尽量避免“* ”的使用，以充分发挥分布式并行优势。可以看出，它与关系型数据库的 SQL 略有不同，但也支持了绝大多数的 SQL 语句如 DDL、DML 以及常见的聚合函数、连接查询、条件查询。

相比 HBase[9]、Pig[10]、MapReduce 等分布式数据处理工具，Hive 平台的主要优势是：

(1) 将复杂紧耦合的 MapReduce 数据处理过程变成简单可松耦合的 SQL 语句查询，使分布式数据处理更简捷易懂；

(2) 支持基于 X/Open 的 SQL 调用级接口 JDBC/ODBC 和 Thrift[11] 服务开发框架，这使得开发人员可以使用常用的方式访问分布式环境中的数据。

3　分布式的文档相似度开发运行环境

根据海量文档相似度计算总体策略和思路，充分利用 Hadoop 和关系型数据库在处理海量数据时的优缺点[12]，本文将需要大数据存储和计算的部分交给 Hive 平台完成，将需要较强表达能力的查询交互部分交给关系型数据库完成，形成可靠性高及数据处理能力强的大规模计算系统环境。整个系统主要包括 4 部分：开源分布式环境(Hadoop)、数据处理平台(Hive + PostgreSQL[13])、开发平台(Eclipse) 和 Web 应用平台(Tomcat)。

3.1　开源分布式环境(Hadoop)

Hadoop 是 Apache 软件基金会管理的一个项目，是 Google 开发的用来支持互联网级数据处理的 MapReduce 编程模型和底层文件系统 GFS(Google File System) 的开源实现[4,6]。在实际搭建 Hadoop 环境时要注意选择稳定性较高的版本，同时也要考虑和其他系统组成部分(如 Hive 平台) 的兼容性，本文选用 Hadoop0.20.2 版本。搭建 Hadoop 环境的简要步骤如下：

(1) 安装 Hadoop 之前，预安装 Java 环境，配置 SSH 服务，修改防火墙策略；

(2) 解压安装配置 Hadoop，参数配置包括：在所有节点上配置 conf/core - site.xml、conf/hdfs - site.xml、conf/mapred - site.xml、conf/hadoop - env.sh 及/etc/hosts 等文档，在主节点上配置 conf/master 和 slave 等

文档。同时,可考虑将 Hadoop 配置文件与其安装目录分离、配置适当的 Hadoop 数据冗余值和 PID(Process ID) 文件位置,构建更稳定的 Hadoop 分布式环境。

(3) 启动 Hadoop 分布式环境,通过"jps"命令查看 Hadoop 启动后所有进程,正常应看到 NameNode、SecondaryNameNode、JobTracker、DataNode 和 TaskTracker。

3.2 数据处理平台(Hive + PostgreSQL)

分布式并行计算环境中,对于大数据的处理,Hive 通过将 HQL 语句转换成 Hadoop 环境中可运行的 MapReduce 程序来实现数据的并行处理,同时 Hive 中的表结构信息需要保存在关系型数据库中,因此,Hive 服务启动前,除了必要的 Hadoop 分布式并行运行环境,还需安装配置关系型数据库系统环境作为 Hive 的元数据存储库。Hive 默认使用嵌入式的 Derby 数据库系统,本文使用 PostgreSQL 数据库系统保存 Hive 的表结构信息,包括表名字、属性、表的列和分区及其属性、表数据所在目录等,并通过配置 hive - site. xml 中数据库的 URL、用户名、密码将表结构信息和实际 Hive 数据连接到一起。而 Hive 表的实际数据内容保存在 Hadoop 的 HDFS 文件系统中,所以本文的相似度计算的数据处理系统环境由 PostgreSQL 和 Hive 两部分组成。主要实现步骤如下:

(1) 安装配置 PostgreSQL 数据库系统,创建用于保存 Hive 表结构信息的数据库,默认数据库名为"default"及相关的用户名和口令; 在/usr/java/jdk1. 6. 0_16/jre/lib/security/java. policy 文件中添加如下配置信息,开放 PostgreSQL 的 Socket 端口权限:

permission java. Net. SocketPermission " 127. 0. 0. 1: 5432" ," resolve ,connect";

(2) 安装配置 Hive,将 PostgreSQL 驱动放到 &HIVE_HOME/lib/目录下,在 hive - default. xml 中,配置 PostgreSQL 数据库的 URL、用户名、密码等信息。

(3) 启动 Hive Thrift Server 服务,CLI 命令为 $ HIVE_HOME/bin/hive - - service hiveserver 10000。启动成功后,程序员即可在任意客户端通过 JDBC 接口访问 Hive 数据,端口号为 10000。

3.3 开发平台(Eclipse)

Eclipse 是一个开放源代码的、基于 Java 的可扩展开发平台[14],应用广泛,版本也很多,考虑到与 Hadoop 环境的兼容性,本文使用 Galileo 版 Eclipse 作为文档相似度计算的开发平台,其安装布署过程与其他Eclipse 平台类似,主要配置重点是 Hive Connector 的配置,即将相关的 Jar 包 Path 到项目的 Lib 目录下,包括的 Jar 包有 hive/lib 目录下的所有与 Hive 平台相关的 Jar 包、postgresql - 9 * - * . jdbc4. jar 和 hadoop - * . * - core. jar。

3.4 Web 应用平台(Tomcat)

Tomcat 是 Apache 软件基金会 Jakarta 项目中的一个核心项目,因为技术先进、性能稳定,而且开源免费,成为目前比较流行的 Web 应用服务器[15]。本文选用 Tomcat6. 0 作为相似度计算的 Web 应用平台。与 Galileo 版 Eclipse 平台的整合包括"com. sysdeo. eclipse. tomcat_3. 2. 1"插件、Eclipse 中 Tomcat 服务器配置和新的性能优化组件"Tomcat - Native"安装。

3.5 文档相似度开发运行环境的主要环境变量

文档相似度开发运行环境涉及很多平台和参数,在 etc/profile 中,除了 Java 相关的环境变量外,还需要准确配置 Hadoop、Hive、PostgreSQL 环境变量,保证整个计算平台的正确稳定运行。

4 实现非结构化文档相似度计算的关键点

为了充分有效利用 Hive 平台对海量结构化数据分析处理功能强大的优势和存储优势,在海量非结构化文档相似度计算开发时,本文围绕 Hive 平台,仔细考虑 Hive 数据仓库的接口连接、数据的预处理、数据的导入及基于 Hive SQL 的数据分析处理等过程和实现细节,这也是整个文档相似度计算系统实现的关键点,文档相似度计算逻辑图如图 2 所示。

4.1 Hive 平台的 JDBC 接口连接

Facebook 公司提出 Hive 平台以来,Hive 因其大数据处理方面的优势被迅速广泛使用。Hive 平台通过 CLI、Client 和 WUI 三种方式为用户使用提供方便(见图 1) 。CLI 方式也就是 Linux 的控制台,每个连接都会启动一个不同的 Hive 副本,适合单元功能测试。WUI 方式是通过浏览器访问 Hive,功能有限,也不够灵活。Client 方式是 Hive 平台的客户端使用接口,用户可以通过 JDBC/ODBC 利用 Thrift 协议连接至 Hive Thrift Server,是系统开发合适的 Hive 服务连接接口,使用时指定连接驱动类型、Hiveserver 的 IP 地址、服务端口和 Hive 元数据库名即可。Hive JDBC 连接的通用代码如下。

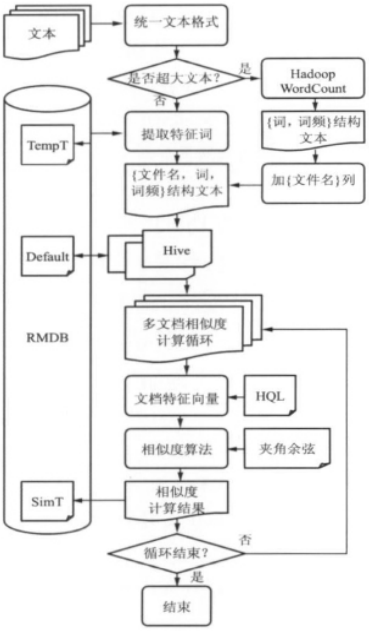

图 2　海量非结构化文档相似度计算逻辑图

```
import org. apache. hadoop. hive. *
Class. forName( " org. apache. hadoop. hive. jdbc. HiveDriver" ) ;
Connection con = DriverManager
getConnection( " jdbc: hive: //hiveserverip: 10000/defalt" ," hive 用
    户名" , " hive 密码" ) ;
Statement stmt = con. createStatement( ) ;
ResultSet res = stmt. executeQuery( " select * from tableName" ) ;
```

4.2　非结构化文档的预处理过程

为便于计算机处理，非结构化文档必须要有一种有效的表示方式。目前，在信息处理领域，文档的表示方法主要采用向量空间模型(VSM) [6]。基本思想是以向量来表示文档: < t1，w1; t2，w2; …; tn，wn > ，其中 ti 是词条项，wi 是 ti 在文档中的权值。因此，所有的 n 维词条向量组成一个文档向量空间。ti 可以是一个词、词组或短语，本文选择词为词条向量 ti，权值 wi 也表示 ti 在文档中的重要程度，选用词条在文档中出现的绝对频率来表示。使用特征词的向量空间来表示文档时，直接使用构成文档的词条作为向量空间的维度，会使相应的词条向量矩阵非常稀疏和巨大，而且存在着大量对文档的描述和区分不相关或影响很小的词条维度，这会造成对文档语义描述的混淆和模糊。为了提高算法的效率和准确度，有必要对构成文档的词条进行特征词的提取和筛选，即对词条向量空间进行降维处理。

非结构化文档的预处理过程是识别和提取非结构化文档中的特征词，通过数据清洗和规范化处理，形成具有明显结构化的中间格式，为数据导入 Hive 平台和后期数据处理做准备。非结构化文档的预处理的主要步骤如下:

(1) 数据源格式统一: 利用文档格式转换工具，将来源文件格式统一成 TXT 格式文档;

(2) 特征提取: 进行分词处理、去掉非特征词，包括文档的标点符号、停用词(如 a、the、of 等) ，在不影响计算精度的情况下有效降低文档向量的维度;

(3) 词频统计: 根据文档向量空间模型，选取词作为特征项，统计词频，将非结构化文档转换成格式为{ 文件名，词 ti，词频 wi} 的规范化文档。

超大容量的大文档可以通过 Hadoop 的 WordCount 算法首先处理为{ 词，词频} 格式的文档，再补充为{ 文件名，词，词频} 格式的文档。

4.3　数据导入

Hive 平台是结构化数据处理平台，数据是保存在 HDFS 文件系统中具有特定分割符的结构化文档，Hive 平台的数据分隔符默认使用 001(ctrl - A) 分隔列，012(\n) 分隔行，为了数据在本地和 HDFS 系统之间的顺利转换，本文将预处理后文档数据及 Hive 平台数据中的列分割符统一为制表符“\t”，行分割符默认。Hive 平台中列分隔符的指定是在创建表格时实现的，方法如下:

```
CREATE TABLE 表名( 文件名 STRING，词 STRING，词频 INT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' ; //创建一
个具有{文件名，词，词频} 字段的列分割符为制表符 "\t" 的 Hive 表
```

Hive 表创建后，即可将预处理后的结构化文档数据顺利导入 Hive，关键代码如下:

```
LOAD DATA [LOCAL] INPATH '预导入的文件名' [OVER-
WRITE] INTO TABLE 表名;
```

使用 LOCAL 选项将使数据导入本地文件系统，否则将导入 HDFS 文件系统。使用 OVERWRITE 选项将删除原来 Hive 表中的数据，否则将新数据添加到文件末尾，本文中文档相似度计算需要进行多文档的共词分析，需保留表中已导入的文档数据，因此不使用 OVERWRITE选项。

LOAD DATA 导入数据方式仅仅是将文件复制到 Hive 管理的目录下，并用 Hive 表的元数据去解释一个或多个文件，所以必须保证所有的数据文件的结构和 Hive 表的结构一致，否则可以 LOAD DATA 成功但是数

据解释不正确。特别注意预处理文档中的列、列分隔符和行分隔符要与创建的 Hive 表保持一致。

4.4 相似度计算中的向量参数的计算

目前存在多种基于 VSM 的文档相似度算法，如贝叶斯算法、神经网络算法等。本文采用“简单向量距离法”[17]，此算法将文档映射成向量空间中的点，点之间的距离用向量间的余弦夹角来度量，即表示文档间的相似程度。假设有 Di 和 Dj 两个不同的文档，经过特征向量选取后所得到的向量分别为：A = f(di)，B = f(dj)。这里将 A 和 B 分别记为：A = {A1，…，Ai，…，An}，B = {B1，…，Bi，…，Bn}，其文档相似度为标准向量点积除以两个向量的长度积。

$$sim(di,dj) = cos(di,dj) = \frac{\sum_{k=1}^{n} di_k \times dj_k}{\sqrt{\left(\sum_{k=1}^{n} di_k^2\right)\left(\sum_{k=1}^{n} dj_k^2\right)}} \quad (1)$$

其中，n 为向量维数，d_k 为词语在文档中的第 k 维权值。由式(1)可知，两个文档的词特征向量的夹角余弦值越接近 1，说明两个文档的向量间距离越短，相似度越大。

夹角余弦的几何意义是在由 N 个元素组成的 N 维空间中，表征两个向量之间夹角的余弦值。一般在使用前需要对向量中的各元素进行无量纲化处理，使各元素都为正，这时夹角余弦的取值范围为 [0，1]，取值越大表明两向量夹角越小，两者越接近，值为 1 时，两向量完全相同。另外，夹角余弦规范化了向量的长度，这意味着在计算相似度时，不会放大数据对象重要部分的作用[18]。“简单向量距离法”效率高，易于实现，相似度计算的精度能满足一般性的要求。

标准向量点本文通过文档 Di 的公共子向量集维度表示。向量的长度通过文档 Di 的特征向量集维度表示。夹角余弦相似度公式转化为：

$$Sim(Di,Dj) = \frac{\text{文档 Di 的公共子向量维度} \times \text{文档 Dj 的公共子向量维度}}{\text{文档 Di 的特征向量维度} \times \text{文档 Dj 的特征向量维度}} \quad (2)$$

在 Hive 平台中，通过 Hive SQL 语言实现文档 Di 的公共子向量集维度提取的关键代码如下：

```
ResultSet rst = stmt.executeQuery("select words from hiveTable where str_filename = '" + TarFileName.trim() + "'"); //提取源文件 Di 中的词条
while (rst.next()) { //遍历所提取的词条
ResultSet rstcount = stmtcount.executeQuery("select in_wordcount from hiveTable where (str_filename = '" + ResFileName.trim() + "' and str_words = '" + rst.getString("words").trim() + "')"); //查询目标文件 Dj 中是否有该词条
if (rstcount.next()) { //该词条存在为真
wordsum_of_tar_in_res = wordsum_of_tar_in_res + rst.getInt("wordcount"); //将词频累加到文件的词向量值中
}}
```

同理，本文提取文档 Dj 的公共子向量集维度和文档 Dj 的特征向量集维度，计算两个文档间的相似度。当多文档相似度计算时，文档通过两个循环，结合 Hive 平台和关系数据库平台实现新导入文档和老文档的相似度计算遍历过程，关键代码如下：

```
ResultSet rs = stmt.executeQuery("select * from postgresTable where sf_success = 'N' order by sf_id ASC"); //检索还没有完成相似度计算的新文档数据
while (rs.next()) { //遍历还没有完成相似度计算的新文档数据
ResultSet rs_secRes = stmt_secRes.executeQuery("select * from postgresTable where sf_success = 'Y' order by sf_id ASC"); //检索已完成相似度计算的老文档数据
while (rs_secRes.next()) { //遍历已完成相似度计算的老文档数据，并分别与新文档进行相似度计算
int wordsum_resTar = simRsToPostgre.resTargetFile(rs.getString("sf_filename").trim(), rs_secRes.getString("sf_filename").trim()); //文档 Di 的公共子向量集维度
int wordsum_Tar = rs_secRes.getInt("sf_wordsum"); //文档 Di 的特征向量集维度
int wordsum_tarRes = simRsToPostgre.tarResgetFile(rs_secRes.getString("sf_filename").trim(), rs.getString("sf_filename").trim()); //文档 Dj 的公共子向量集维度
int wordsum_Res = rs.getInt("sf_wordsum"); //文档 Dj 的特征向量集维度
int simrs = wordsum_tarRes* wordsum_resTar / wordsum_Tar* wordsum_Res; // 夹角余弦计算
stmt_insert.executeUpdate("insert into simresults values('" + rs.getString("sf_filename").trim() + "','" + rs_secRes.getString("sf_filename").trim() + "'," + simrs + ")"); //保存计算结果到关系数据库
stmt_insert.close();
}
rs_secRes.close();stmt_secRes.close();
stmt_rs_compared_update.executeUpdate("update postgresTable set sf_success = 'Y' where sf_id = " + rs.getInt("sf_id")); //计算后，将新文档标记为已完成相似度计算的老文档
stmt_rs_compared_update.close();
}
```

5　实证研究

结合以上算法，本文给出相应实验数据。文献集包括以计算(Computing)为主题的英文文献 10 篇，如高性能计算、软计算、可信计算等，总容量 23KB(因 Hive 平台的数据处理实时性和实验室分布式环境的规模问题，为了对比单机和分布环境的数据处理效果，这里只使用了小数据集进行试验)。实验环境，使用虚拟服务器方式，虚拟服务器配置为 Intel(R) Core(TM) 2 Duo CPU E8400 @ 3.00GHz，1GB 的内存，Centos Release 5.5(Final)系统。实验测试了单台虚机和 3 台虚机构建的分布式环境(1 台 Namenode，3 台 Datanode)两种情况。单台虚拟服务器时，文档预处理和数据导入共耗时 103s，平均每篇处理耗时约为 10s。相似度计算耗时超长，平均每对文档相似度计算耗时 200m。

在分布式环境下，文档预处理和数据导入共耗时 99s，平均每篇处理耗时约为 10s，与单服务器环境下相差无几，这与文档预处理过程没有做并行处理相关。相似度计算耗时还是很长，但平均每对文档相似度计算耗时缩短为 130m，较单服务器环境下执行效率有较大提高，说明 Hive SQL 语句有效调用 MapReduce 机制以并行方式执行数据处理，有效提高了文档相似度计算处理能力。

分析分布式环境下文档相似度计算的性能和效率，可以看出，目前 Hive 平台对数据处理的实时性效果还是没有明显的改进，Hive 更适合非实时性的信息查询分析与信息深度挖掘应用。当然，在不涉及 MapReduce 编程，而传统算法又无法实现对海量数据的分析处理时，Hive 平台是目前不错的可供选择的解决方案之一。

6　结　语

互联网信息资源因其载体和通信通道的便利性，逐步成为大众和科研工作者的主要信息来源。实现非结构化互联网文档的信息挖掘和分析是提升用户使用体验的重要基础，具有普适性。本文以非结构化文档为研究对象，基于分布式环境，结合 Hive 数据处理平台和 PostgreSQL 关系型数据库，详细说明传统文档相似度算法分解成可分布并行处理文档相似度计算的过程，并给出搭建海量非结构化文档相似度计算开发环境架构和关键步骤。使读者能够明确掌握 Hadoop 的 Hive 平台实质、HQL 语言的使用及如何实现传统算法向分布式并行算法的快速部署解决方案。当然，Hadoop 和 Hive 平台在实际应用中还需要面对很多方面的问题，如性能问题、Hadoop 平台带来系统稳定性问题和数据备份问题，及传统算法的可并行化问题等。

参考文献：

[1] Willett P. Recent Trends in Hierarchical Document Clustering: A Critical Review [J]. *Information Processing and Management*, 1988, 24(5): 577 – 597.

[2] Salton G, Buckley C. Term Weighting Approaches in Automatic Text Retrieval [J]. *Information Processing and Management*, 1988, 24(5): 513 – 523.

[3] Callan J P. Passage – level Evidence in Document Retrieval [C]. In: *Proceedings of the* 17*th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA. New York: Springer – Verlag. 1994: 302 – 310.

[4] 洪毅虹. 基于 MapReduce 架构的文档相似度计算方法 [J]. *网络与信息*, 2010(9): 36 – 37.

[5] MapReduce [EB/OL]. [2011 – 04 – 27]. http://hadoop.apache.org/mapreduce/.

[6] Hive [EB/OL]. [2011 – 04 – 27]. http://hive.apache.org/.

[7] Thusoo A, Sarma J S, Jain N, et al. Hive – A Petabyte Scale Data Warehouse Using Hadoop [C]. In: *Proceedings of the* 2010 *IEEE* 26*th International Conference on Data Engineering*(*ICDE*), Long Beach, California, USA. 2010: 996 – 1005.

[8] Hadoop开发者入门专刊 [EB/OL]. [2011 – 04 – 27]. http://ishare.iask.sina.com.cn/f/11493440.html.

[9] HBase [EB/OL]. [2011 – 04 – 27]. http://hbase.apache.org/.

[10] Pig [EB/OL]. [2011 – 04 – 27]. http://pig.apache.org/.

[11] Thrift [EB/OL]. [2011 – 04 – 27]. http://incubator.apache.org/thrift/.

[12] Pavlo A, Paulson E, Rasin A, et al. A Comparison of Approaches to Large – Scale Data Analysis [C]. In: *Proceedings of the* 35*th SIGMOD International Conference on Management of Data*, New York, NY, USA. 2009: 165 – 178.

[13] PostgreSQL [EB/OL]. [2011 – 04 – 27]. http://www.postgresql.org/.

[14] Eclipse [EB/OL]. [2011 – 04 – 27]. http://www.eclipse.org/.

[15] Tomcat [EB/OL]. [2011 – 04 – 27]. http://tomcat.apache.org/.

[16] Salton G, Wong A, Yang C S. A Vector Space Model for Automatic Indexing [J]. *Communications of the ACM*, 1975, 18(11): 613 – 620.

[17] Salton G. Automatic Text Processing: The Transformation Analysis and Retrieval of Information by Computer [M]. Boston, MA, USA: Addison - Wesley Longman Publishing Co., 1988.
[18] Tian R, Xie P. Study on the Standardization of Similarity Evaluation Method of Chromatographic Fingerprints(Part Ⅰ) [J]. *Traditional Chinese Drug Research & Clinical Pharmacology*, 2006, 17(1): 40 - 42.

(作者 E - mail: zhaohm@ mail. las. ac. cn)

Sloan 基金会资助 CLIR/DLF 研究如何培养数据保存技能

最近 Alfred P. Solan 基金会资助美国图书馆和信息资源委员会(以下简称 CLIR) 117 567 美元,以进行有关如何培养本学科数据保存能力的研究。该项目将由 CLIR 的数字图书馆联盟(DLF) 主管。

自然科学、社会科学和人文科学大多数的研究生课程都没有培养学生的数据管理能力,有时甚至都没有告诉他们为什么这些技能对他们学科的发展来说是重要的。在每一个学科,至少要有一些专业人士掌握有关研究数据的创造、获取、重用和保存这些复杂的需求。这是图书馆和信息技术专家的责任,也是学校图书馆、信息和计算机科学专家的责任。

CLIR 主席 Chuck Henry 说 "如果我们的教育机构要为今后的研究、教学和学习建立强大的、高效的、妥善整合的网络环境,那么每个学科的专家必须重视发展自己的数据保存技能。感谢 Sloan 基金会给我们机会深入地研究开发数字化保存实践和教育的大环境。"

该项目将包括三个相互关联的部分:

(1) 专业发展需求以及学术界关于数字化保存教育和培训机会的环境扫描。

(2) 使用人类学研究方法研究 5 个正在进行数字化保存活动的网站。

(3) 一份分析前两项研究工作的结果,然后根据研究发现提出有关修正 CLIR 的学术图书馆计划里博士后课程表(http://www.clir.org/fellowships/postdoc/postdoc.html) 建议的报告。

Chuck Herry,DLF 项目负责人 Rachel Frick,以及布尔茅尔学院首席信息官兼图书馆馆长 Elliott Shore 将成为项目的主要研究人员。此外,Shore 还是 CLIR 董事会成员和博士后指导教师。

CLIR(http://www.clir.org/) 是一个独立的、非营利性的组织,其使命是扩大信息的可获得范围,不论信息是如何记录和保存的。通过出版物、项目和方案,CLIR 努力使得在未来人们能更好地获得信息。在与其他机构的合作中,CLIR 帮助创造了扩大"图书馆"这一概念的各种服务,并支持了信息的提供者和保存者。

(编译自: http://www.clir.org/news/pressrelease/11sloanpr1.html)

(本刊讯)