基于 NoSQL 的海量航空物流小文件分布式多级存储方法*

丁建立[a,b]，郑峰弓[a]，李永华[a]，罗云生[a]，曹卫东[a,b]
(中国民航大学 a. 计算机科学与技术学院; b. 天津市智能信号与图像处理重点实验室，天津 300300)

摘 要： 为了解决航空物流领域海量小文件存储效率和访问效率不高的问题，提出一种基于 NoSQL 的海量小文件分布式多级存储方法。充分考虑到数据的时效性、本地性、操作的并发性以及文件之间的相关性，先根据相关性将文件合并，然后采用分布式多级存储，使用内存式 Redis 数据库做缓存，HDFS 做数据的持久化存储，其过程采用预取机制。实验结果表明，该方法有效提高了小文件的存取效率和磁盘的利用率，显著地降低了网络的带宽占用和集群 NameNode 的内存消耗，适合解决航空领域海量小文件存储问题。

关键词： 小文件; Redis; HDFS; 多级存储; 预取机制
中图分类号： TP311 **文献标志码：** A **文章编号：** 1001-3695(2017)05-1433-04
doi:10.3969/j.issn.1001-3695.2017.05.035

Method of distributed multi-level storage of massive small files of air logistics based on NoSQL

Ding Jianli[a,b], Zheng Fenggong[a], Li Yonghua[a], Luo Yunsheng[a], Cao Weidong[a,b]
(*a. College of Computer Science & Technology, b. Tianjin Key Laboratory for Advanced Signal Processing, Civil Aviation University of China, Tianjin* 300300, *China*)

Abstract: In order to solve the problem of inefficient storage and inefficient access of massive small files in the field of aviation logistics, this paper proposed a method of distributed multi-level storage of massive small files based on NoSQL, which fully considered the data timeliness, the data locality, the concurrency of operation and the correlation between files, according to the correlation, merging file, and then applied a distributed multi-level storage, and which applied memory-type Redis database to the data-cache and HDFS to permanently store the data with using prefetching mechanism. Experimental results show that, the method effectively improves the access efficiency of small files and the utilization of disk, and also significantly reduces the occupation of network bandwidth and the memory consumption of NameNode cluster, and is also suitable for solving problems of storage of massive small files in the field of aviation.

Key words: small file; Redis; HDFS; multi-level storage; prefetch mechanism

0 引言

针对航空物流行业，大多数信息的交换都是以报文的形式存在，种类繁多，而单个报文的大小基本上在 3 kB 左右。对于航空物流信息系统的 iLink 平台上每天的数据交换量为 10 GB 左右，然而对于 10 GB 大小的数据，大约有上千万个报文。随着民航信息产业的不断发展，导致海量数据信息也呈爆炸式增长。针对这种爆炸式的增长，数据存储技术也在日新月异。传统的数据存储方式已经不能应对当前产业的需求，原因主要存在两方面，即存储效率不高和响应时间过长。因此，各企业单位为了满足生产的需要，逐渐将数据接入云存储平台存储。

目前，最典型的云存储平台是 Hadoop 分布式文件系统(HDFS)。相对于传统的存储介质，Hadoop 分布式文件系统(HDFS)通过多台廉价的机器支持大规模的文件存储，伸缩性强，解决了存储空间限制的问题[1]。同时，HDFS 能提供高吞吐量的数据访问，非常适合大规模数据集上的应用，而且即使在出错的情况下也能保证数据存储的可靠性[2]。不幸的是，HDFS 是专门针对大文件存储的分布式系统，存储小文件效率极其低下。在 HDFS 存储小文件时，分布式集群的 NameNode 会消耗大量的内存空间去管理每个小文件的元数据，使得文件的访问效率大大降低；另外 HDFS 的存储是按照数据块存储的，大量的小文件存储就会极大地浪费磁盘空间，导致存储效率不高。为了解决海量小文件的存储效率和访问时间问题，本文提出一种高效存取海量小文件的方法。该方法首先采用数据在空间和时间的局部性原理以及小文件之间的相关性，将具有相关关系的小文件合并成大文件存储到 HDFS 上；然后采用多级存储方式，将经常用到的数据放于缓存级和活跃级存储，将长时间不用的数据放于永久级进行压缩式存储；再在缓存级与活跃级之间采用

收稿日期：2016-05-07；修回日期：2016-07-26 基金项目：民航局科技创新重大专项基金资助项目(MHRD20140106，MHRD20150107)；中央高校基金资助项目(3122014P004，3122016A001)；中国民航大学天津市智能信号与图像处理重点实验室开放基金资助项目(2015ASP02)

作者简介：丁建立(1963-)，男，河南洛阳人，教授，硕导，博士，主要研究方向为民航智能信息处理、航空物联网；郑峰弓(1990-)，男，河南鹤壁人，硕士研究生，主要研究方向为民航智能信息处理(macalzheng@163.com)；李永华(1974-)，女，天津人，讲师，硕士，主要研究方向为民航智能信息处理；罗云生(1990-)，男，四川达州人，硕士研究生，主要研究方向为民航智能信息处理；曹卫东(1964-)，女，天津人，副教授，博士，主要研究方向为数据库与数据挖掘、民航信息系统软件可靠性.

数据预取机制，根据文件的相关性，当一个文件被访问时，将与其相关的多个文件预取到缓存区，以减少磁盘 I/O 和提高读取效率；最后在缓存区采用 NoSQL 内存式数据库 Redis，利于缓存文件和元数据的管理及高效查询。

1 相关工作

航空物流产业的数据存储方式大多采用传统的文件转关系型数据库存储的方式，然而关系型数据应对大规模数据的存储和查询存在很大的瓶颈问题。同时，为了将数据接入云存储，不得不采取相应的解决方法将海量小文件高效地存储在云端，因为云端存储的基本上是大文件。目前，针对 HDFS 解决小文件高效存取的方法是将小文件合并为大文件存储，减少存储文件的数目，同时为小文件建立索引。

Hadoop 存在 HAR、SequeueFile 和 CombineFileInputFormat 几种机制来解决海量小文件的存储问题。HAR 机制是一种人工干预的方式，是在小文件合并后需要人工删除；另外 HAR 方式其实是在 Hadoop 集群上提交了一个作业，执行 MapReduce 过程，所以合并过程必须在 Hadoop 集群上提交；Sequicefile 方式由一系列的二进制 key/value 组成，如果为 key 小文件名，value 为文件内容，则可以将大批小文件合并成一个大文件，该方式适合一次性写入大量的小文件。CombineFileInputFormat 是一种新的 inputformat，用于将多个文件合并成一个单独的 split；另外，它会考虑数据的存储位置。

文献 [3] 在 BlueSky 中使用文件合并和缓存机制，减轻 NameNode 负荷，对 HDFS 存储效率进行了改善，提高了小文件的存取效率。文献 [4,5] 都采用了 NoSQL 内存式数据库 Redis 做缓存，以提高文件的检索速度和元数据的管理方便，而文献 [4] 采用 HDFS 的 SequeueFile 机制存储小文件。文献 [5] 针对海量 XML 文件使用合并和再合并的技术，提高海量小文件的访问效率，大大降低了 NameNode 的内存占用。上述方法虽然提高了小文件的存取效率，但是仍存在以下问题：

a) 没有考虑数据的时效性。对于一些数据随着时间的推移，数据的时效性逐渐降低，因为磁盘的空间是有限的，若长期不采取相应的措施，势必造成数据丢失。

b) 没有考虑数据的本地性和并发性操作。大多数小文件的合并都在集群之外进行，没有考虑数据的本地性，增加网络带宽和存储时间；另外合并主要是采取单机合并，没有考虑集群的并发能力，势必会造成存储拥堵，甚至产生数据丢失。

2 分布式多级存储架构

2.1 分布式多级存储架构

航空物流数据的基本特点有：单个文件的大小较小，文件的海量性、文件之间的相关性、数据的时效性（数据在一段时间以后就不再访问）、数据的访问量大以及数据访问要求的低延迟性等。根据航空物流数据的基本特点，利用 HDFS 的云存储能力解决数据存储的海量性，利用 Redis 的高速读写性提高数据访问的速度，利用分布式原理提高系统的并行性，从而解决大的访问量，采用多级存储是考虑到数据的时效性，将经常访问的数据放在缓存级和活跃级，提高访问速度，将不再访问的数据进行压缩式存储，有效节省磁盘空间。

图 1 所示为分布式多级存储的一个宏观架构，最外面是接入分布式存储的应用程序，方框里面是一个三级存储结构，包括数据缓存级、数据活跃级和数据永久级。

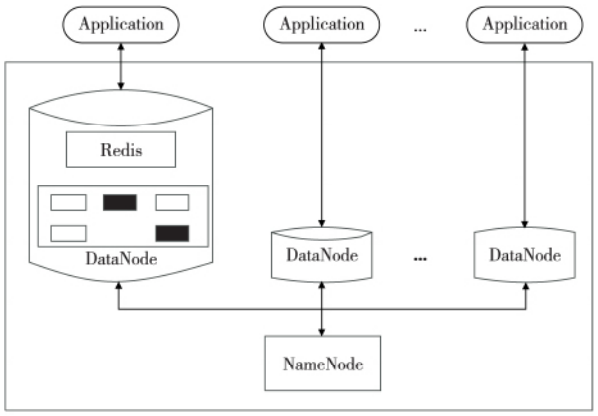

图1 分布式多级存储架构

1) 数据缓存级 该级位于每个 DataNode 的节点内存中，使用 Redis 内存数据库做缓存，且各节点 Redis 数据库之间自建集群，用于存储小文件的索引和缓存数据预取机制的文件。

2) 数据活跃级 该级属于 HDFS 的 DataNode 的数据存储区，图中用白色方格表示，用于存储在活跃期内的文件数据。

3) 数据永久级 该级也属于 HDFS 的 DataNode 的数据存储区，图 1 中用黑色方格表示。与活跃级不同的是，该级存储的是长久不被外界访问的数据，并采用数据块的压缩式存储。

2.2 缓存存储结构

本文使用 Redis 数据库作为多级存储的缓存级。Redis 是于 2009 年发布的开源 Key-Value 模型的内存数据库，经过数个版本的不断优化，它已经发展成为较成熟的内存数据库[7]。Redis 是一个速度非常快的非关系型数据库，它可以存储键（key）与六种不同类型的值（value）之间的映射，可以将存储在内存的键值对数据持久化到硬盘，可以使用复制特性来扩展读性能，还可以使用客户端分片来扩展写性能。本文使用 Redis 的其中的一种数据结构类型——HASH 来设计报文的元数据存储结构和数据缓存的存储结构。报文的元数据存储结构和数据缓存的存储结构分别如表 1、2 所示。

表 1 缓存级报文的元数据存储结构

类别	名称	类型	说明
Hash-key	FileName	字符串	报文名
Sub1	FileSize	数字值	报文大小
Sub2	MergeFileName	字符串	报文所属合并文件的文件名
Sub3	OffSet	数字值	报文在合并文件中的偏移量
Sub4	LastReadTime	数字值	上一次访问时间
Sub5	ImmortalizedFlag	数字值	文件永久化标志

表 2 缓存级报文的缓存数据存储结构

类别	名称	类型	说明
Hash-key	CD_FileName	字符串	报文名
Sub1	CD_FileSize	数字值	报文大小
Sub2	FlightNumber	字符串	报文所属的航班号
Sub3	CD_FileType	字符串	报文所属类型
Sub4	CD_Content	字符串	报文内容
Sub5	Read_count	数字值	在缓存中被访问的次数

3 分布式多级存储方案设计

3.1 小文件合并

当海量航空物流报文存储到云存储平台后，具有一定相关关系的报文文件将被合并成大文件，即一次航班的业务性报文

或非业务性报文分别放在连续空间，然后将多次航班的这样的文件合并成一个大小接近 128 MB 的大文件，同时建立数据块内部一个索引。图 2 所示是一个合并好的数据块。该数据块由 BlockInternalIndex（数据块内部索引）和 Blockdata（数据）组成。BlockInternalIndex 是由 offset 和 Length 组成，分别代表数据块内部的空间的偏移量和数据大小，也就是一次航班的业务性报文或非业务性报文在数据块中存储的偏移量和大小。

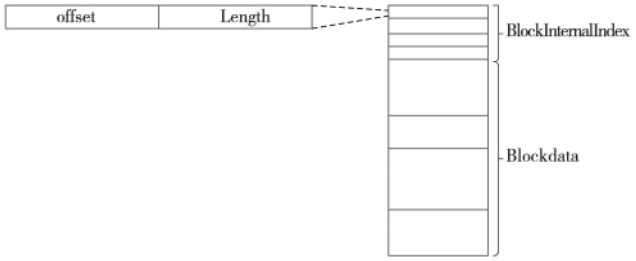

图2　数据块内部索引结构

3.2　文件存储

如图 3 所示，描述的仅是海量小文件的一个节点上的文件存储流程，包括文件的解析、缓存、建立索引、文件的合并和文件存储到 HDFS 上。同时，为了充分发挥分布式存储的本地性和并发性功能，首先要将数据拆分到不同的 DadaNode 节点。

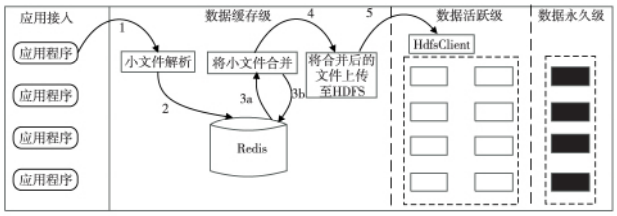

图3　文件存储流程

详细信息如下步骤：

a）将海量的小文件批量地读取到内存中，解析小文件，将文件的航班号和文件类型提取出来。

b）将提取出来的内容和文件批量地存储到 Redis 的缓存中，其中利用 Redis 的客户端分片功能来扩展写的性能。

c）将一次航班的业务性报文和非业务性报文分别聚集起来，并将其加入文件合并进程，在缓存中建立每个小文件的索引；在合并的文件中建立内部索引；将文件在大文件中的偏移量返回给 Redis。

d）将合并好的文件提交至 HDFS 客户端，将文件存储到 HDFS 上，该操作就是 HDFS 上传文件的基本操作。

3.3　文件访问

如图 4 所示，描述的仅是海量小文件的一个节点上的文件访问流程，主要包括文件缓存搜索、文件索引的映射、文件从 HDFS 上的读取、文件的预取机制、文件永久化的解除。

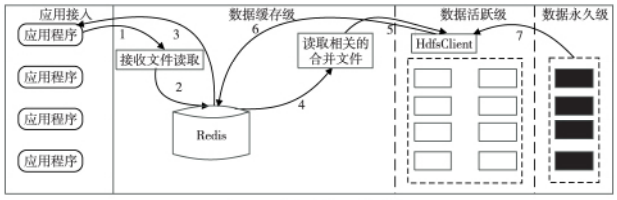

图4　文件访问流程

详细流程如以下步骤：

a）接收应用程序的文件访问请求，将请求提交给 Redis 客户端。

b）在缓存级的 Redis 数据库中搜索要读取的报文，若是在缓存中能找到，立即将报文返回给应用程序，Read_count 值加 1，访问结束；若是在缓存中找不到要读取的文件，转到步骤 c）。

c）在 Redis 中搜索文件索引表，找到文件对应的索引，根据索引找到在 HDFS 上存储的位置。

d）根据步骤 c）查找的文件索引，解析文件索引中读取的文件是否已经持久化。若没有持久化，生成一条文件读取命令提交给 HDFS 的客户端，从 HDFS 上读取想要的文件；反之进行步骤 e）。

e）向 HDFS 的客户端提交一个数据块解压缩命令，将要访问的数据块从数据永久级解压缩成数据活跃级。

f）从 HDFS 读取想要的文件，返回到数据缓存级存到 Redis 中，并将文件返回给应用程序。

g）数据预存机制被触发，根据上一时刻读取文件在数据块中的偏移量，通过数据块内部索引确定要预取的数据的偏移量和长度，将数据返回到缓存级并存储到 Redis。

步骤 d）根据缓存级文件缓存结构字段 Read_count 的值，将 Read_count 值比较低的文件从缓存中替换掉，并将这次缓存进来的文件的 Read_count 值赋值为 0。

3.4　文件永久化

如图 5 所示，描述的仅是海量小文件的一个节点上的文件永久化流程。具体步骤如下：

a）永久化监视器向 Redis 客户端提交一次统计查询，将属于同一数据块的文件作一次统计，判断这一数据块的文件的上一次访问时间是否都超过了设定的永久化时间，没有超过继续监视，若是超过了转到步骤 b）。

b）向 HDFS 的客户端提交一次数据块永久化任务。

c）HDFS 接到数据永久化任务，针对这些数据块进行一次压缩，将这些数据从活跃级移到永久区。

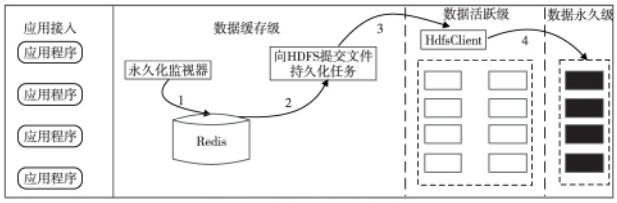

图5　文件永久化流程

3.5　预取机制

数据预取机制是一种高效的存取优化方式。数据预取指在处理器访问该数据进行计算之前，提前将数据从主存储器加载到缓存存储器上，以降低处理器访问数据的停顿时间，以提高处理器的性能。本文考虑报文之间的相关性和数据空间及时间的局部性，采用数据预取机制。

航空物流报文数据的基本特征有：一次物流航班，伴随着不同报文的产生，而这些报文之间存在很强的相关性。例如这些报文都属于同一航班，一个主单报文下面有很多分单报文，一个舱单报文下面又有很多主单报文，另外一次航班有很多状态报，而状态报又分很多种类等。

根据数据特征分析到：一次物流航班的报文不是很大，其中业务性报文约占 30%，非业务性报文约占 70%。非业务性报文主要是一次航班的状态报文。因此，主要思路是：a）在缓存级设置一个固定缓存区，用于缓存数据，内部采用简单的页面置换算法；b）将一次航班的报文分为业务性和非业务性两种，用于预取时取哪些数据；c）设定一次文件读取时，触发预取机制。例如，当读取一次航班的舱单时，将这次航班的主单、分单、委托书等业务性报文一起送到缓存；当读取一次航班的运抵报时，将这次航班的状态报文一起发送到缓存。

4 实验对比分析

实验采用 64 位 CentOS7 操作系统，Hadoop 版本为 2.6.4，Redis 版本为 3.0，搭建五个节点的集群环境，每个节点拥有一个 CPU 和 4 GB 内存，HDFS 的数据副本设置两份。实验数据使用某国内航空物流部门提供的某一周内的部分航空物流数据。实验内容包括：a) 比较本方案和原生的 HDFS 文件存储速度和 HDFS 的 NameNode 的内存占用量；b) 比较本方案和原生的 HDFS 文件的读取效率；c) 比较本方案和原生的 HDFS 磁盘使用效率。

4.1 小文件存储效率对比

实验中，为了对比原生的 HDFS 方法和分布式多级存储方法在文件存储效率和集群 NameNode 的内存占用量，分别进行五次实验，这五次实验分别存储 10、50、100、150、200 k 个小文件进行对比，文件大小分别为 18、94、196、282、384 MB，其中，单个文件大小均在 3 KB。实验结果如图 6 和 7 所示。

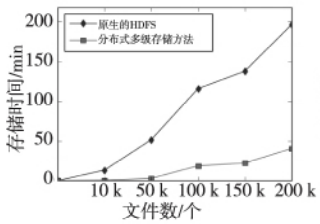

图6 文件存储时间比较

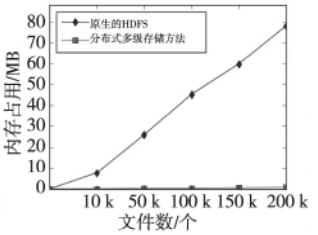

图7 NameNode内存占用情况

如图 6 所示，随着存储文件数据量的增长，原生的 HDFS 方法时间用得越来越多，而分布式多级存储方法上传文件的时间增长很缓慢，相比之下分布式多级存储方法存储文件要高效得多，实验得出在相同条件下，原生的 HDFS 方法文件的存储时间平均是目标优化方案的 8.8 倍。如图 7 所示，随着文件数量的增多，原生的 HDFS 的 NameNode 内存占用量呈直线增长趋势，而由于分布式多级存储方法采用合并后再存储文件的方式，从而极大地减少了 NameNode 的内存占用。这由于合并后的文件基本上接近 HDFS 的一个数据块的大小，集群只需管理这个大文件的元数据。而原生的 HDFS 之所以存储效率低下，主要是因为当存储每个小文件时，都需要进行一次存储空间的申请，并且当前一个文件存储备份完毕才能开始下一个文件的存储，没有做到存储的并行化和本地化；另外，每个文件的元数据都要存放在 NameNode 节点。

4.2 小文件读取效率对比

实验中，在存储了 15 万个小文件（文件大小为 282 MB，文件占用空间大小 671 MB）后进行实验，分别进行随机读取和顺序读取方式来对比两种方法在文件读取的效率，对比结果如图 8 和 9 所示。

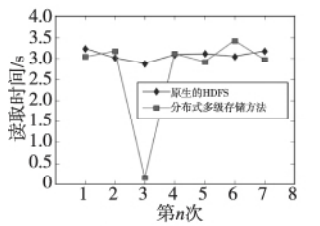

图8 文件随机读取时间对比

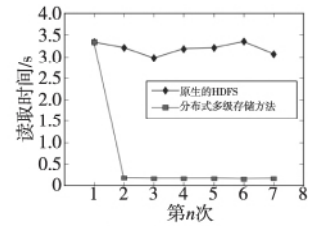

图9 文件顺序读取时间对比

如图 8 所示，当进行文件的随机读取时，两种存储方案的文件读取效率基本上是一致的，并且分布式多级存储方法文件的读取效率有时比原生的 HDFS 要高。如图 9 所示，当文件进行顺序读取时，分布式多级存储方法的文件读取效率明显地高于原生的 HDFS 方法，原因主要是分布式多级存储方法采用了缓存机制和数据预取机制。实验表明，采用缓存机制文件的读取时间基本上是 160 ms 左右；另外采用数据预取机制，极大地减少了多次磁盘 I/O 的时间，从而提高了文件读取效率。

4.3 磁盘利用率对比

实验中，每天存储 15 万个小文件（文件大小为 282 MB，文件占用空间大小 671 MB）进行实验，设置数据持久化的时间阈值为一天的时间（实际环境中则根据以往经验设置），即若是数据一天没有被访问，表明该数据要做持久化存储。图 10 记录的是连续五天两个存储方案的数据增长趋势。

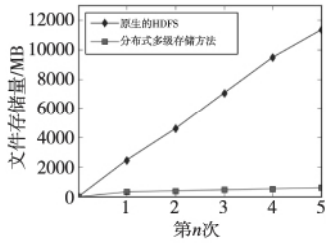

图10 文件存储量随时间的变化

如图 10 所示，随着时间的变化，原生的 HDFS 存储的数据增长速度很快，原因主要是原生的 HDFS 存储系统不能针对性和周期性地进行压缩式存储，而分布式多级存储方法的存储方式数据的增长速度非常缓慢，其中原因有两点：a) 分布式多级存储方法的存储是将向文件合并存储的；b) 分布式多级存储方法将不经常被访问的数据进行压缩式存储。注意的是，因为原因 a)，第一天没有进行永久化。实验中发现经压缩式存储可以节省 79% 左右的磁盘空间。同时注意到，经过持久化的数据再次被访问，要涉及到数据反压缩，会极大地增加访问时间。然而根据航空报文这种数据的特点，失效时间相对固定，这种情况发生概率很小；另外，也可以通过增大持久化时间阈值来减少这种情况的发生。综合来说，采用分布式多级存储的确可以极大地节省磁盘空间。

5 结束语

本文根据当前航空物流行业实际生产中面临的海量小文件存储的困境，提出一种基于 NoSQL 内存式数据库的分布式多级存储方案。该方案主要结合当前最流行的云存储平台 HDFS，又结合使用内存数据库 Redis，并根据航空物流数据的基于时间的有效性，设计多级存储机制，并采用文件的预取机制。该方案与原生的 HDFS 相比，有效提高了小文件的存取效率，显著地降低了集群 NameNode 的内存消耗，控制了磁盘数据的快速增长，有效提高了磁盘的利用率。需要说明的是，该方法只适用于那些具有周期性失效的数据存储，而且针对民航领域海量报文数据的存储能发挥很大的能力。

参考文献：

[1] 张守利，杨冬菊，韩燕波. 一种面向海量小文件的文件接收和存储优化方案[J]. 小型微型计算机系统，2015，36(8)：1747-1751.

[2] Dhruba B. HDFS architecture guide [EB/OL]. [2013-08-04]. http://hadoop.apache.org/docs/stable1/hdfs_design.html.

[3] Dong Bo，Qiu Jie，Zheng Qinghua，*et al*. A novel approach to improving the efficiency of storing and accessing small files on Hadoop：a case study by PowerPoint files [C]//Proc of IEEE International Conference on Services Computing. 2010：65-72.

[4] 刘高军，王帝澳. 基于 Redis 的海量小文件分布式存储方法研究[J]. 计算机工程与科学，2013，35(10)：58-64. （下转第 1441 页）

常庞大，存储空间占用较大。

b) 输入矩阵 Δ 和输出矩阵 Γ 均为 $n \times m$ 维矩阵，n 为所有命题的数量，m 为所有变迁的数量。由于健康饮食规则众多，知识库规模庞大，从而导致 Δ 和 Γ 维数过多，在一定程度上影响了计算效率。再者，由于推理得到的所有禁忌食物都需要保留，适宜食物也需保留较多项以备用户选择，所以数据库中存储的每日个性化健康饮食方案占用空间较大。

a) 由于系统的开发环境采用的是 Visual Studio 2013，没有自带的矩阵运算工具包，在矩阵运算上并没有独特的优势。而基于模糊 Petri 网的形式化并行推理算法采用矩阵运算，并且新定义了三个运算符，这就要求在开发过程中先自行编写矩阵类实现矩阵的表示与运算，增加了工作量，开发难度较大。

图4　个性化健康饮食方案推荐系统运行结果

3　结束语

健康饮食知识经过网络化变换，可以转换为确定性 Petri 网和模糊 Petri 网，而确定性 Petri 网又是一种特殊的模糊 Petri 网。本文提出的个性化健康饮食方案推理机制采用基于模糊 Petri 网的形式化正向并行推理。Petri 网的主要输入信息包括用户的基本信息、健康状态、当日外部环境信息，主要输出信息为适合当日状况的个性化健康饮食方案，包括适宜饮食和禁忌饮食。同时，推理算法会对 Petri 网中的冲突规则和矛盾命题进行自动识别与处理。目前，该方案生成机制已经应用到自主研发的个性化健康饮食方案推荐系统中，能准确地为用户推荐科学、安全、合理的个性化健康饮食方案，大大降低了普通用户在饮食养生保健上的搜索成本。

接下来的研究将着重关注两个方面：a) 对个性化健康饮食方案智能生成机制进行纵向扩展，研究如何对个性化健康饮食方案进行定量优化，确定方案中各种食物的最佳摄入量，确保能量摄入的合理性和膳食结构的平衡性；b) 对个性化健康饮食方案智能生成机制进行横向扩展，研究如何将这一机制应用到健康管理的其他领域，如个性化健康运动方案的推荐等，最终形成全方位的健康管理。

参考文献：

[1] Russell S，Norvig P. 人工智能——一种现代方法 [M]. 姜哲，译. 3 版. 北京：清华大学出版社，2013：165-168.

[2] Shortliffe. Computer-based medical consultations：MYCIN [M]. 3rd ed. [S. l.]：Elsevier，2012：78-85.

[3] 王婉婷. 故障诊断逻辑推理知识的在线更新方法初探 [D]. 武汉：华中科技大学，2012：22-24.

[4] 王炳和，相敬林. 基于神经网络方法的人体脉象识别研究 [J]. 西北工业大学学报，2002，20(3)：454-457.

[5] 韦玉科，汪仁煌，黎敬波. 一种亚健康诊断推理的新方法 [J]. 计算机应用研究，2006，23(3)：70-72.

[6] 韦玉科，汪仁煌，陈群，等. 基于竞争神经网络的中医智能诊断推理新方法 [J]. 计算机工程与应用，2006，42(7)：224-226.

[7] 韩崇昭，朱洪艳，段战胜，等. 多源信息融合 [M]. 2 版. 北京：清华大学出版社，2010：297-301.

[8] 罗昊. 模糊 Petri 网的形式化推理算法及其应用研究 [J]. 福建电脑，2016，32(4)：10-12.

[9] 鲍培明. 模糊 Petri 网在非结构化决策支持中的应用研究 [J]. 计算机工程，2001，27(12)：81-83.

[10] Gao Meimei，Wu Zhiming，Zhou Mengchu. A Petri net based formal reasoning algorithm for fuzzy production rule-based systems [C] //Proc of IEEE International Conference on Systems，Man and Cybernetics. [S. l.]：IEEE Press，2000：3093-3097.

[11] 贾立新，薛钧义，茹峰. 采用模糊 Petri 网的形式化推理算法及其应用 [J]. 西安交通大学学报，2003，37(12)：1263-1266.

[12] 莫太平，赵佩斯，段任航，等. 制造执行系统的 Petri 网建模及性能分析 [J]. 计算机集成制造系统，2015，21(8)：2064-2071.

[13] 刘妹琴，王毅星. 基于 Petri 网与 D-S 证据理论的电力系统故障诊断 [J]. 华中科技大学学报：自然科学版，2014，42(10)：88-92.

[14] 林涛. 基于模糊 Petri 网的知识推理与维护系统的设计 [D]. 长沙：长沙理工大学，2004：45-50.

[15] 吴荣海. 加权模糊 Petri 网在不精确知识表示和推理中的应用研究 [D]. 昆明：云南师范大学，2006：21-40.

[16] 徐欢. 基于模糊 Petri 网的专家系统知识推理及其应用 [D]. 天津：天津科技大学，2007：34-60.

[17] 沈开金. 中医养生调理与效验方 [M]. 合肥：安徽科学技术出版社，2016：128-200.

[18] 中国营养学会. 中国居民膳食指南 [M]. 3 版. 拉萨：西藏人民出版社，2016：120-125.

[19] 中华中医药学会. 中医体质分类与判定 [M]. 北京：中国中医药出版社，2009：35-46.

(上接第 1436 页)

[5] 刘俊龙，刘光明，张黛，等. 基于 Redis 的海量互联网小文件实时存储与索引策略研究 [J]. 计算机研究与发展，2015，52(S2)：148-154.

[6] Zhang Yin，Han Weili，Wang Wei，*et al*. Optimizing the storage of massive electronic pedigrees in HDFS [C] //Proc of the 3rd International Conference on Internet of Things. 2012：68-75.

[7] 刘小俊，徐正全，潘少明. 一种结合 RDBMS 和 Hadoop 的海量小文件存储方法 [J]. 武汉大学学报：信息科学版，2013，38(1)：113-115，120.

[8] 李洪奇，朱丽萍，孙国玉，等. 面向海量小文件的分布式存储系统设计与实现 [J]. 计算机工程与设计，2016，37(1)：86-92.

[9] 张春明，芮建武，何婷婷. 一种 Hadoop 小文件存储和读取的方法 [J]. 计算机应用与软件，2012，29(11)：95-100.

[10] 杜忠晖，何慧，王星. 一种 Hadoop 小文件存储优化策略研究 [J]. 智能计算机与应用，2015，5(3)：28-32，36.

[11] Chandrasekar S，Dakshinamurthy R，Seshakumar P G，*et al*. A novel indexing scheme for efficient handling of small files in Hadoop distributed file system [C] //Proc of International Conference on Computer Communication and Informatics. 2013：1-8.

[12] Zhou Xiaoqing，Sun Jiaxiu，Luo Xingxian. Design and development of the mass image storage platform based on Hadoop [J]. Applied Mechanics & Materials，2015，742(264)：721-725.