doi:10.3969/j.issn.1673-9833.2019.04.009

基于 Spark 平台并行化 Slope One 算法的设计与实现

黄 婕[1]，刘长生[1,2]，刘程莉[3]

（1. 空军航空维修技术学院 航空电子设备维修学院，湖南 长沙 410124；2. 湖南工业大学 智能信息感知及处理技术湖南省重点实验室，湖南 株洲 412007；3. 莱昂大学 工程、计算机与航空学院，西班牙 莱昂 24071）

摘 要：协同过滤算法是解决信息超载的关键技术之一，但仍存在预测不准确的问题。因此，在分析了 Spark 技术及框架并阐述了 Slope One 算法不足的基础上，针对项目与用户间的相似性提出了一种改进的 Slope One 算法，并在 Spark 平台上实现了该算法。实验证明，改进后的 Slope One 算法具有更高的预测准确性，且在 Spark 平台上实现了并行化操作，用 Speedup 和 Sizeup 方法证明了算法的并行性、扩展性良好，提高了算法的效率。

关键词：Spark；并行化；Slope One；预测准确性；扩展性

中图分类号：TP274 **文献标志码**：A **文章编号**：1673-9833(2019)04-0047-07

引文格式：黄 婕，刘长生，刘程莉 . 基于 Spark 平台并行化 Slope One 算法的设计与实现 [J]. 湖南工业大学学报，2019，33(4)：47-53.

Design and Implementation of Spark-Based Paralleled Slope One Algorithm

HUANG Jie[1]，LIU Changsheng[1,2]，LIU Chengli[3]

（1. Department of Aviation Electronic Equipment Maintenance，Airforce Aviation Repair Institute of Technology，Changsha 410124，China；2. Hunan Key Laboratory of Intelligent Information Perception & Processing Technology，Hunan University of Technology，Zhuzhou Hunan 412007，China；3. School of Engineering，Computer and Aviation，University of León，León 24071，Spain）

Abstract：As one of the key technologies to solve the information overload, the collaborative filtering algorithm exhibits the flaw of inaccuracy prediction. Therefore, based on the analysis of Spark technology as well as its framework and the elaboration of the flaw in Slope One algorithm，an improved Slope One algorithm has thus been proposed for the similarity between projects and users, followed by the implementation of the algorithm on Spark platform. Experimental results show that the improved Slope One algorithm has a higher accuracy of prediction with its paralleled implementation on Spark. The combined methods of Speedup and Sizeup prove that this algorithm is characterized with a good parallel effect and an excellent expansibility, thus helping to promote the efficiency.

Keywords：Spark；parallelization；Slope One；predictive accuracy；expansibility

1 研究背景

随着互联网的迅猛发展，数据量急剧增长。开放、共享的大数据既能给用户带来超量的信息和资讯，也使得数据量非常庞大而难以选择，用户得花大量时间去筛选有价值的信息。近几年，个性化推荐技术已

收稿日期：2018-11-23

作者简介：黄 婕（1979-），女，湖南长沙人，空军航空维修技术学院副教授，硕士，主要从事计算机应用与计算机软件方面的教学与研究，E-mail：haungjie918@163.com

通信作者：刘长生（1966-），男，湖南耒阳人，空军航空维修技术学院教授，博士，主要从事计算机应用技术与智能制造技术方面的教学与研究，E-mail：30623297@qq.com

经被广泛地应用在网络、媒体、社交等各个领域中，尤其是协同过滤的推荐技术，已经成为应用最广泛、最成功的一种[1]，但该算法在数据的预测准确性方面仍存在问题。

Spark 是一种开源、新兴的并行计算框架[2]，它具有基于分布式计算的 MapReduce 的优点[3]，而且 Spark 计算结果可以被保存在内存中，因而减少了 IO 开销，处理速度将大大提高。协同过滤算法存在一定的预测度不理想问题，为了达到更高的预测准确度，将改进后的算法在大数据 Spark 平台上做并行化操作，可提高算法的执行效率，满足推荐任务的即时、准确性要求，具有重要的应用价值。

随着大数据时代的到来，数据挖掘工具已然遭遇瓶颈，虽然离线计算框架 Hadoop Map Reduce 能进行大规模的数据计算[4]，但是需要频繁地输入输出，IO 开销增大，多任务、多作业的处理速度较慢，于是并行框架的 Spark 平台应运而生。

1.1 Spark 系统

Spark 系统是一个生态系统，可提供流运算、迭代运算、图运算等解决方案，如图 1 所示。

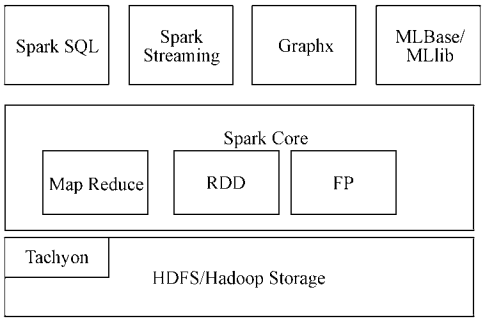

图 1 Spark 生态系统图

Fig. 1 Spark ecosystem diagram

Spark SQL：通过 SQL 表达式在 Spark 上做数据查询，进行大数据查询分析[5]，也可调用用户自定义函数将查询结果与分析结果合并，以重用数据，因而提高了查询效率。

Spark Streaming：这是一个流处理系统，能对实时数据进行 map、reduce、join 等复杂操作，并将结果保存到外部文件系统中。

GraphX：它是 Spark 平台提供图运算的 API，提供了弹性分布式属性图的概念及操作，利用其框架可实现基于图的各种运算。

MLBase/MLlib：它是一个能提供有关机器算法和实用程序的组件，具有分类算法、回归算法、聚类算法、降维和底层优化等程序，还可以进行测试和数据生成。MLBase 进行了 4 个边界的定义，分别是优化器（MLOptimizer）、特征抽取 API（MLI）、机器学习库（MLlib）和并行计算框架（MLRuntime）[6]。

Tachyon：该组件是一个分布式系统，能够提供高效、高容错性和高可靠性的计算[7]。

Spark Core：是生态系统的核心计算框架，既可单独运行又可在集群上运行。集群运行时，需要借助管理系统，将任务分布到各个节点上。

总之，Spark 生态系统是一个以弹性分布数据集（resilient distributed dataset，RDD）为基础、Spark 框架为核心的、基于内存计算的并行计算大数据平台，可利用系统中不同组件满足不同需求的计算。

1.2 Spark 运行模式和构架

Spark 可以在本地进行单机模式的运行计算，也可以在分布式集群上并行运算。Spark 的运行模式由环境变量 Spark Context 的 Master、本地单机和集群的部署模式共同决定。Spark 的常用运行模式如下：

1）Standalone。该模式可将 Spark 部署到 Master 和所有 Worker 的节点上。

2）Yarn-client。该模式中 Yarn 集群运行 Executor，本地运行 Driver。

3）Yarn-cluster。该模式中，Yarn 集群中同时运行 Executor 和 Driver。

这 3 种运行模式的不同之处在于有不同的资源分配方式，由不同的任务调度算法来执行计算任务。但是 3 种 Spark 运行模式有相同的工作流程，具体如图 2 所示。

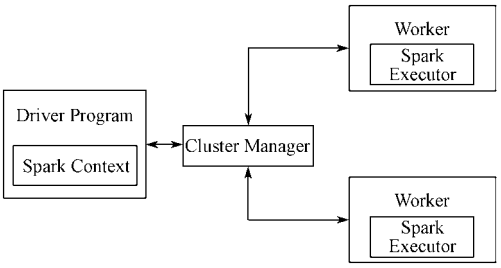

图 2 Spark 运行模式的工作流程图

Fig. 2 A workflow diagram of Spark operation mode

任意一个 Spark 程序都有对应的 Executor 进程，而每个 Executor 进程内部有多个 Task 线程与之对应。这种并行的资源分配、调度模式有利于不同 Spark 程序间的资源共享，因而大大提高了其执行效率。

图 3 为 Spark 运行构架图。如图 3 所示，Spark 程序运行中，通过动作触发 job，其中 job 基于 RDD 的依赖关系构建 DAG 图，构建好的 DAG 图再由 DAGScheduler 解析，构建成不同的 Stage，并且计算 Stage 间的依赖关系。然后由 TaskScheduler 调度器分配工作集 TaskSet，再将其划分成多个线程并行计算，同时将计算结果返回给 TaskScheduler，再返回给 DAGScheduler，计算结果全部完成后，将其返回

给驱动程序，或者保存在外部存储系统中，并且将资源全部释放。

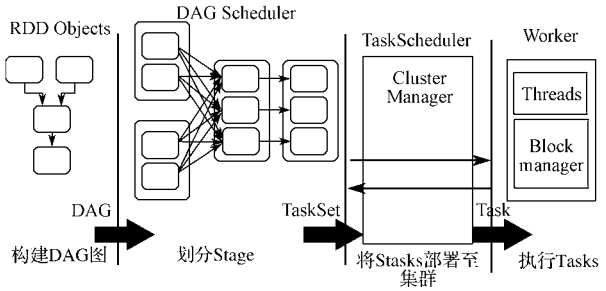

图 3　Spark 运行构架

Fig. 3　Spark operation framework

2　Slope One 算法及其改进

2.1　Slope One 算法

Slope One 算法是一种协同过滤算法 [8]，该算法简单且实用。若用户对 X、Y 两个项目之间存在某种线性关系 $Y=mX+b$（m、b 为常数），就可以通过 X 预测项目 Y 的偏好值。如果 m=1，则线性关系为

$$Y=X+b。\tag{1}$$

由式（1）可知，只需要确定两个项目间的平均差异，即 $b=Y-X$，就能进行过滤推算，具体的推算流程如图 4 所示。

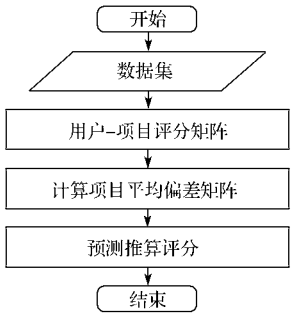

图 4　Slope One 推算流程图

Fig. 4　Slope One calculation workflow diagram

流程图实现 Slope One 算法，先主要计算项目评分偏差矩阵，再进行预测推算。

1）评分偏差矩阵

评分偏差值的计算式为：

$$dev_{ij}=\sum_{v\in U_{ij}}\left(R_{vi}-R_{vj}\right)\Big/\left|U_{ij}\right|。\tag{2}$$

式中：dev_{ij} 为两个项目 i、j 之间的评分偏差；

R_{vi}、R_{vj} 分别为用户 v 对项目 i、j 的评分；

U_{ij} 为评价项目 i 和项目 j 的共同用户集合，即 $U_{ij}=U_i \cap U_j$，其中 U_i、U_j 分别为对项目 i、j 评价的所有用户的集合；

$|U_{ij}|$ 为项目 i 和 j 评价的共同用户数目。

从式（2）可以看出，评分偏差是由用户对两个项目的评分计算而得到的，dev_{ij} 的值越小，则评分偏差越小，表明这两个项目评分差距越小。

先计算评分偏差值 dev_{ij}，再由 dev_{ij} 构成一个 m 行 n 列的评分矩阵 $\boldsymbol{D}$。

2）算法推荐结果

根据公式（1），b 是两项目间的平均评分偏差，X 是用户 u 对物品 i 的评分，利用其预测该用户对物品 j 的评分，即利用 $P_{uj}=R_{ui}+dev_{ij}$，P_{uj} 即 u 用户对物品 j 的预测评分。

$$P_{uj}=\sum_{i\in I_u}\left(R_{ui}+dev_{ij}\right)\Big/\left|I_u\right|。\tag{3}$$

上式可简化为

$$P_{uj}=\overline{R_u}+\sum_{i\in I_u}dev_{ij}\Big/\left|I_u\right|。\tag{4}$$

式（3）（4）中：I_u 是用户 u 的所有项目评分集合；

$|I_u|$ 是用户 u 的评分项目数量；

$\overline{R_u}$ 是用户 u 对项目的平均分。

以上算法可以计算出推荐预测值，但是其中没有考虑对同一项目有过评分的用户数量对推荐结果的影响因子，所以 Daniel Lemire 提出了一种基于权重的 Slope One 算法，即 WSO（Weighted Slope One）算法 [9]，其公式如下：

$$P_{uj}=\left[\sum_{i\in I_u}\left(R_{ui}+dev_{ij}\right)\times\left|U_{ij}\right|\right]\Big/\sum_{i\in I_u}\left|U_{ij}\right|。\tag{5}$$

WSO 算法的特点是原理简单且容易实现，预期差异值可以离线进行计算，这样既提高了算法的执行效率，也提高了预测的准确性。

2.2　Slope One 算法的改进

针对图 4 中 Slope One 算法的局限性，课题组考虑了项目与用户间的相似性而提出了一种改进的 Slope One 算法 [9]，其算法流程如图 5 所示。

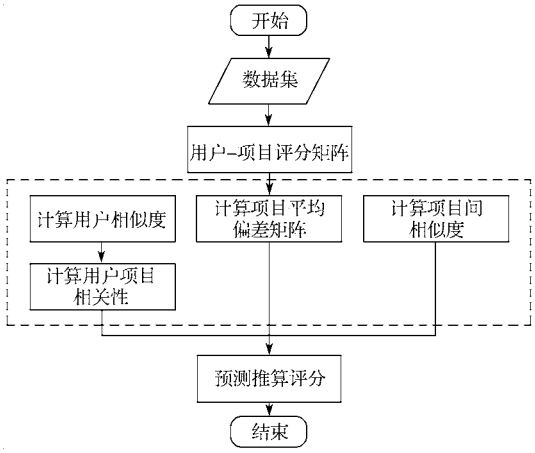

图 5　改进的 Slope One 算法流程图

Fig. 5　A modified Slope One algorithm workflow diagram

与图 4 对比，改进的 Slope One 算法新增了计算

用户相似度和项目相似度。

改进的 Slope One 算法计算过程如下：首先进行数据预处理，获取用户－项目评分矩阵；再进行用户相似度的计算和用户间平均相似度 S_{ui} 的计算（S_{ui} 即用户 u 对项目 i 进行评分的用户间的平均相似度）；然后对项目间评分偏差矩阵和项目间相似度进行计算；最后，利用用户相似度、项目相似度，以及评分偏差进行项目的预测推荐 [10]。

具体步骤如下：

1）数据预处理。将计算操作的数据类型转化成 m 行、n 列的二维空间矩阵 $\boldsymbol{R}$。

2）用户相似度 sim 的计算。利用公式（6）计算用户 u 和 t 之间的相似度 $sim(u, v)$。

$$sim(u,v)=\frac{\sum_{i\in I_{uv}}\left(R_{u,i}-\overline{R_u}\right)\times\left(R_{v,i}-\overline{R_v}\right)}{\sqrt{\sum_{i\in I_{uv}}\left(R_{u,i}-\overline{R_u}\right)^2}\times\sqrt{\sum_{i\in I_{uv}}\left(R_{v,i}-\overline{R_v}\right)^2}}。\quad(6)$$

3）用户项目相关性 S_{ui} 的计算。利用公式（7）计算用户 u 与对项目 i 所有评分用户的平均相似度。

$$S_{ui}=\begin{cases}\dfrac{\sum_{I\in U,\ i\neq u} sim(u,t)}{|U_i|-1}, & |U_i|\neq 1;\\ 1, & |U_i|=1。\end{cases}\quad(7)$$

4）评分偏差矩阵 $\boldsymbol{D}$ 的计算。利用前文中的式（2）计算项目间的评分偏差 dev_{ij}，再由 dev_{ij} 构成一个 m 行 n 列的矩阵 $\boldsymbol{D}$。

5）项目相似度的计算。项目 i 和项目 j 的相似度可以表示为如下公式（8）。

$$sim(i,j)=\begin{cases}sim_A(i,j), & sim_R(i,j)\leqslant 0;\\ \lambda sim_A(i,j)+(1-\lambda)sim_R(i,j), & sim_R(i,j)>0。\end{cases}\quad(8)$$

式中 λ 为可调节加权因子。允许动态调整，若有新项目加入，则无法计算此项目与其他项目的相似度，就将 λ 设置为 1。

6）推荐算法的评分预测。这是改进的 Slope One 算法的关键步骤，如公式（9）中 P_{uj} 即为用户 u 对项目 j 的预测评分。

$$P_{uj}=\frac{\sum_{i\in I_u}\left(R_{ui}+dev_{ij}\right)\times sim(i,j)\times s_{ui}}{\sum_{i\in I_u} sim(i,j)\times s_{ui}}。\quad(9)$$

通过以上 6 个步骤，可以对用户 u 没有评价过的项目做推荐预测，把合理项目推荐给用户。

3 Slope One 并行化算法实现

以增强 Slope One 算法处理数据的能力和计算数据的可扩展性为目的，将算法在 Spark 平台上做并行化实现，提出了一种基于 Spark 平台的并行化的 Slope One 算法。

1）评分偏差矩阵的计算

从式（2）可看出，评分偏差的计算得收集两个项目的共同用户，以〈*Uid*,（*lid*, *Rat*）〉格式输入，再利用 groupByKey( ) 归并评分过的项目，形成〈*Uid*, *List*（*lid*, *Rat*）, …〉的弹性分布数据集 [11]，即 Spark 创新的 RDD。*List*（*lid*, *Rat*）是先前已评价的用户列表。将 imForUrList 中用户的评分项目两两匹配，形成〈（lid_1, lid_2），*cha*〉的 RDD1。为了方便计算，使 $lid_1<lid_2$，对应式（2）中的 $R_{vi}-R_{vj}$。再利用函数归并元素，形成（lid_1, lid_2）元组；（s_1, s_2）是 value 的键值对，s_1 对应式（2）的分子，s_2 对应其分母。项目偏差矩阵计算示意图如图 6 所示。

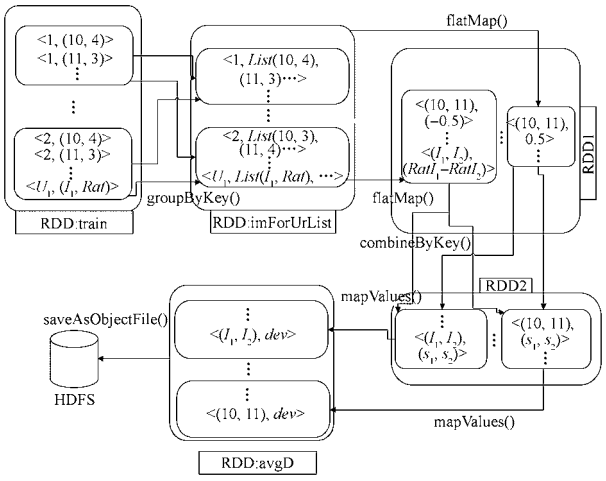

图 6 项目偏差矩阵计算示意图

Fig. 6 Project deviation matrix calculation diagram

combineByKey（）方法中包含 creComb（）、mergeValue（）、mergeComb（）方法的定义，3 种方法的定义如下：

1)creComb(v:Double):(Double , Int)={(v, I)}；

2)mergeValue(c:(Double, Int), v:Double):(Double, Int)= {(a=c._1+v, c._2+1)}；

3)mergeComb(c1:(Double, Int), c2:(Double, Int)): (Double, Int)= {(c._1+c2._1,c1._2+c2._2)}。

项目间的相似度用 mapValues（）方法求解，其元素格式为〈（lid_1, lid_2），*dev*〉，其中 *dev* 表示项目间的平均评分偏差，其值等于 s_2 除以 s_1。项目平均评分偏差可以离线计算，待后续在线测试时从 HDFS 文件系统中 I/O 读取，能有效增强算法的执行效率。

2）用户相似度的计算

计算用户相似度与项目相似度的方法类似，只是在 Spark 平台中有不同的输入方式和转换方式。具体

转换步骤如图 7 所示。

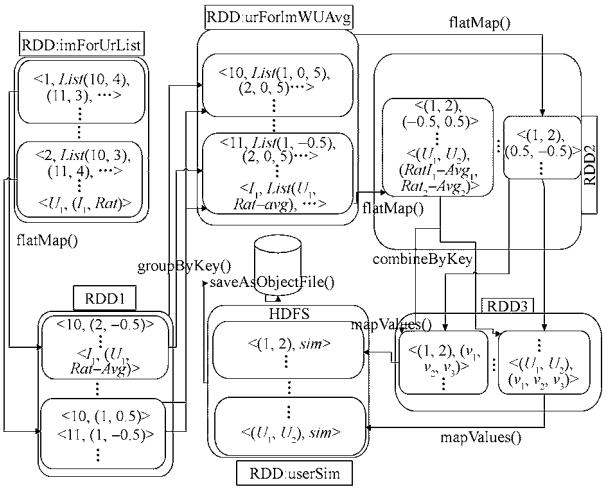

图 7　用户相似度计算示意图

Fig. 7　User similarity calculation diagram

首先，利用 ImForUrList 在 flatMap 中迭代 *List*(*lid*，*Rat*），计算出用户对所有项目的平均值，形成〈*lid*, *List*(*Uid*, *Rat*−*Avg*)〉的 RDD1。通过 groupByKey 归并〈*lid*, *List*(*Uid*, *Rat*−*Avg*), …〉的 urForImWUAge。再利用 flatMap 将 urForImWUAge 作为输入形成〈(*Uid*$_1$, *Uid*$_2$), (*Rat*$_1$−*Avg*$_1$, *Rat*$_2$−*Avg*$_2$)〉的 RDD2。其中，*Uid*$_1$、*Uid*$_2$ 是某个项目的用户，*Avg*$_1$ 是 *Uid*$_1$ 的评分平均值，*Rat*$_1$ 是 *Uid*$_1$ 对项目 *i* 的评分。最后，通过 mapValues，利用 *simU*（*simU* 是两两配对用户间的相似度）求解用户间的相似度，userSim 元素格式为〈(*Uid*$_1$, *Uid*)，*simU*〉。

3）用户与项目相关性矩阵的计算

计算用户项目相关性矩阵示意图如图 8 所示。

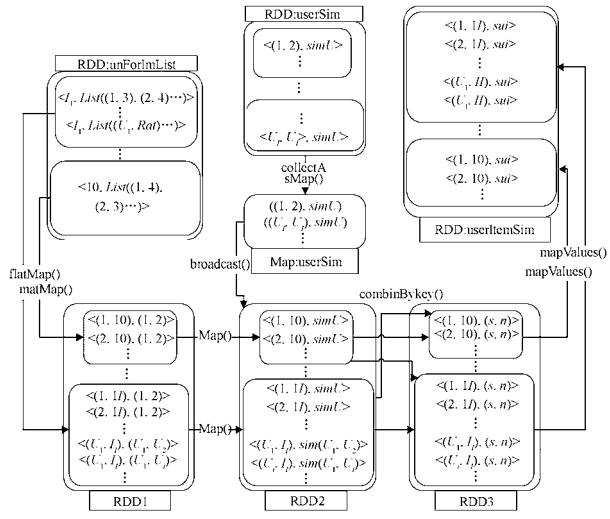

图 8　计算用户项目相关性矩阵示意图

Fig. 8　User project calculation correlation matrix diagram

首先利用 urFrolmList 为输入计算用户与项目 *i* 的其他用户间的平均相似度 *Sui*，在 flatMap 的转换中将 *value* 和 *key* 的键值对形成〈(*Uid*$_1$, *lid*), (*Uid*$_1$, *Uid*$_2$)〉的 RDD1。再对 RDD1 进行 map 操作，形成〈(*Uid*$_1$, *lid*), *simU*〉的 RDD2。在 combineByKey 中的 creCombSu、mergeSuValue 和 mergeSuComb 方法做聚合计算，形成〈(*Uid*$_1$, *lid*), (*s*, *n*)〉的键值对，构成 RDD3。最后，通过 mapValues 计算用户项目间的相关性。

4）预测评分的计算

Slope One 算法改进的核心是预测评分的计算，主要利用评分偏差、项目相似度和用户项目相关性矩阵进行评分预测的计算，计算预测值示意图见图 9。

利用用户 *u* 的项目评分表和本文中的公式（8），计算预测评分是基于用户和项目相似度的，计算完用户 *u* 和项目 *j* 之间的相似度及偏差值后，再计算预测评分值。基于 flatMap 项目配对，形成〈(*j*, *lid*), (*Rat*, *Sui*)〉的 RDD1；将 itemSUI 和 avgD 合并成〈(*j*, *lid*), (*Rat*, *Sui*), (*sim*, *dev*)〉的 RDD2；再利用 RDD2 中的 map、reduce 进行预测评分的计算。

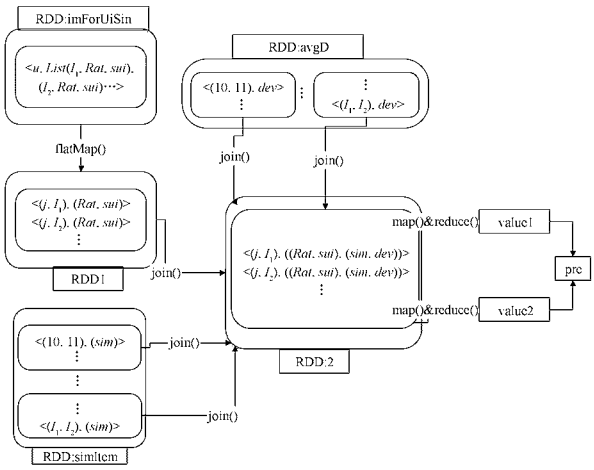

图 9　计算预测值示意图

Fig. 9　Predicted value calculation diagram

如图 9 所示，*value* 1 为公式（8）中的分子部分 $\sum_{i\in I_u}\left(R_{ui}+dev_{ij}\right)\times sim\left(i,j\right)\times s_{ui}$，*value* 2 为分母部分 $\sum_{i\in I_u} sim\left(i,j\right)\times s_{ui}$，*pre* 的值为 *value*1/*value*2，即为预测评分值。

4　实验结果分析

4.1　预测值的计算

将改进的协同过滤 Slope One 算法在 Spark 平台上做预测准确度的测评实验，为了验证其准确度，实验分为 4 组，在不同数据集上进行平均绝对误差（mean absolute error，MAE）的计算 [12]。MAE 是利用实际

评分和预测评分的偏差来衡量预测结果准确度的。

$$MAE=\sum_{i=1}^{n}\left|(p_i-r_i)\right|\Big/N。\qquad(10)$$

式中：p_i 为实际评分；

r_i 为预测评分；

N 为项目数量。

从式（10）可看出 MAE 值越小预测准确度越高。

表 1 中列出了并行 Slope One 算法在不同训练集比例上的 MAE 值。

表 1 不同训练集比例 MAE 值对比表

Table 1 MAE contrast table for different training sets

算法	训练集比例 /%			
	60	70	80	90
Slope One	0.749 752 4	0.745 429 1	0.742 784 3	0.741 418 4
改进的 Slope One	0.750 924 5	0.745 496 8	0.741 784 3	0.740 082 6

分析表 1 中的数据可以得知，随着训练集的增大，两种算法的 MAE 值均随之下降，但改进后的 Slope One 算法的预测值比 Slope One 的预测准确度更高。为便于观察，将两种算法的 MAE 值与训练集的关系绘制成图 10。

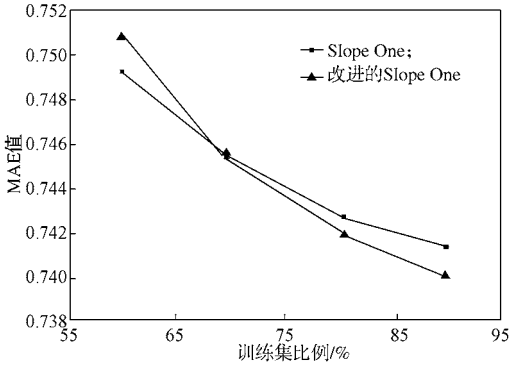

图 10 Slope One 和改进算法的 MAE 对比图

Fig. 10 MAE contrast diagram of Slope One and the modified algorithm

由图 10 可知，当数据训练集比例为 60% 时，Slope One 算法的 MAE 值低于改进算法的，随着训练集比例的增大，改进算法的 MAE 值低于 Slope One 算法的。即当数据越密集，改进的 Slope One 算法 MAE 值越高，预测准确度越高。

4.2 并行性的测评

算法并行性的测评中，有 *Speedup*、*Sizeup* 两个重要指标 [13]。

Speedup 是指一个任务在单处理器系统和并行处理系统的时间消耗比，即

$$Speedup(p)=T_l/T_p。\qquad(11)$$

式中：T_l 是任务在单处理器 l 上运行的时间；

T_p 是任务在 p 个并行节点中运行的时间。

Sizeup 是在数据里节点数不变时，扩大数据集倍数的时间消耗比，即

$$Sizeup(DS,\ p)=T_{sp}/T_{sl}。\qquad(12)$$

式中：T_{sl} 是集群处理数据集 DS 的时间；

T_{sp} 是集群处理 p 倍数据集的时间。

基于 Spark 平台的并行化 Slope One 算法，其性能通过 *Speedup*(p) 和 *Sizeup*(DS，p) 方法的曲线图反映出来。*Speedup*(p) 方法中，保持数据集不变，将集群的节点个数从 1 增加到 4，观察算法执行时间的变化，见图 11。

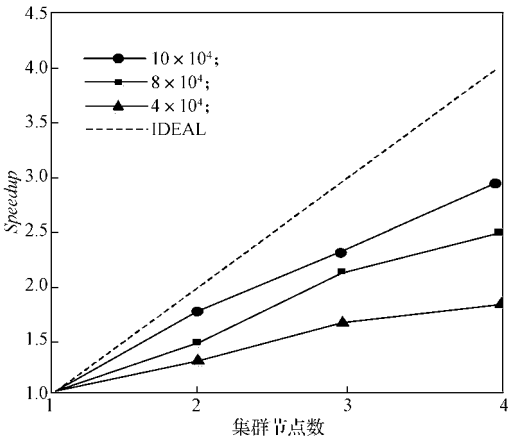

图 11 改进的 Slope One 算法 Speedup 曲线图

Fig. 11 Speedup curves of the modified Slope One algorithm

由图 11 可知，改进 Slope One 算法的 Speedup 曲线图基本是线性的，且随着数据集规模的增大，Speedup 值越趋于理想值。同时，对于同一数据集 3 节点时的加速比增长比 2 节点时的加速比增长快，主要是因为两个节点的 Spark 集群包括一个 Master 节点和一个 Work 节点。Master 节点除主要负责计算任务外，还需要负责集群的任务分配和资源调度。

为测评并行改进算法的 *Sizeup*(DS，p)，选取数据集中 2×10^4 条评分记录为基准数据集，将 2×10^4、4×10^4、10×10^4 条记录数据集作为 P 倍基础数据集的实验数据集。观察算法执行时间，见图 12。

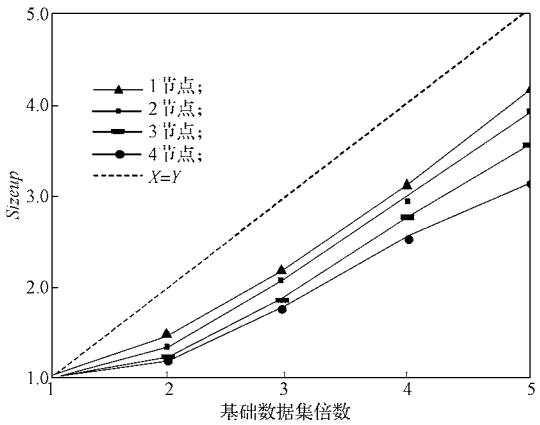

图 12 改进的 Slope One 算法的 *Sizeup* 曲线图

Fig. 12 *Sizeup* curves of the modified Slope One algorithm

由图 12 可看出，无论是 Spark 单机伪分布还是集群环境，并行算法的 *Sizeup* 值都在 *X*=*Y* 线下方。对于 *Sizeup* 指标，若指标曲线在 *X*=*Y* 线下方，说明并行算法的效率良好。且随着集群中 Worker 节点数增多，*Sizeup* 曲线越平滑，算法的并行效果越优。

以上两个实验中，*Speedup* 一直呈上升趋势，达到了线性效果。*Sizeup* 并没有随着数据集的增加而线性增长，说明改进 Slope One 算法的并行化操作性能优良，有效地解决了可扩展性问题。

5　结语

本文首先介绍了 Spark 的计算框架，阐述了 Slope One 算法过程，针对其预测不准确性做出了改进，并将改进的算法在 Spark 平台上做并行化操作，最后以实验证明基于 Spark 平台的并行化 Slope One 算法的预测准确度较高。同时，绘制了 Speedup 和 Sizeup 曲线图，证明该并行算法有较高的加速比，具有良好的扩展性，同时提高了算法的运行效率。

参考文献：

[1] 车晋强，谢红薇 . 基于 Spark 的分层协同过滤推荐算法 [J]. 电子技术应用，2015，41(9)：135-138.
CHE Jinqiang，XIE Hongwei. Hierarchical Collaborative Filtering Algorithm Based on Spark[J]. Application of Electronic Technique，2015，41(9)：135-138.

[2] 张明敏 . 基于 Spark 平台的协同过滤推荐算法的研究与实现 [D]. 南京：南京理工大学，2015.
ZHANG Mingmin. Research on and Implementation of Collaborative Filtering Recommendation Algorithm Based on Spark Platform[D]. Nanjing：Nanjing University of Science and Technology，2015.

[3] 中国 IDC 圈 . 腾讯在 Spark 上的应用与实践优化 [EB/OL]. （2016-03-04）[2018-03-24]. http://cloud.idcquan.com/yjs/ 73681.shtml.
IDC of China. The Application and Practice Optimization of Tencent on Spark[EB/OL]. （2016-03-04）[2018-03-24]. http://cloud. idcquan. com/yjs/73681.shtml.

[4] WANG Z，SUN L，ZHU W，et al. Joint Social and Content Recommendation for User-Generated Videos in Online Social Network[J]. IEEE Transactions on Multimedia，2013，15(3)：698-709.

[5] 陈业斌，刘　娜，徐　宏，等 . 基于 Spark 的空间范围查询索引研究 [J]. 计算机应用与软件，2018，35(2)：96-101.
CHEN Yebin，LIU Na，XU Hong，et al. Research on Range Queries in Spatial Index Based on Spark[J]. Computer Applications and Software，2018，35(2)：96-101.

[6] 孔维梁 . 协同过滤推荐系统关键问题研究 [D]. 武汉：华中师范大学，2013.
KONG Weiliang. Research on the Key Problems of Collaborative Filtering Recommendation System[D]. Wuhan：Central China Normal University，2013.

[7] 王　毅，楼恒越 . 一种改进的 Slope One 协同过滤算法 [J]. 计算机科学，2011，38(10A)：192-194.
WANG Yi，LOU Hengyue. An Improved Slope One Algorithm for Collaborative Filtering[J]. Computer Science，2011，38(10A)：192-194.

[8] 王　茜，王均波 . 一种改进的协同过滤推荐算法 [J]. 计算机科学，2010，37(6)：226-228，243.
WANG Qian，WANG Junbo. Improved Collaborative Filtering Recommendation Algorithm[J]. Computer Science，2010，37(6)：226-228，243.

[9] DESHPANDE M，KARYPIS G. Item-Based Top-N Recommendation Algorithms[J]. ACM Transactions on Information Systems，2014，22(1)：143-177.

[10] 孙金刚，艾丽蓉 . 基于项目属性和云填充的协同过滤推荐算法 [J]. 计算机应用，2012，32(3)：658-660，668.
SUN Jingang，AI Lirong. Collaborative Filtering Recommendation Algorithm Based on Item Attribute and Cloud Model Filling[J]. Journal of Computer Applications，2012，32(3)：658-660，668.

[11] ZAHARIA M，CHOWDHURY M，DAS T，et al. Resilient Distributed Datasets：A Fault-Tolerant Abstraction for In-Memory Cluster Computing[J]. In-Memory Cluster Computing.USENIX Symposium on Networked Systems Design and Implementation(NSDI)，2012，70(2)：141-146.

[12] TAKÁCS G，PILÁSZY I，NÉMETH B，et al. Major Components of the Gravity Recommendation System[J]. ACM SIGKDD Explorations Newsletter，2007，9(2)：80-83.

[13] NIGHTINGALE E B，CHEN P M，FLINN J. Speculative Execution in a Distributed File System[J]. ACM Transactions on Computer Systems，2006，24(4)：361-392.

（责任编辑：廖友媛）