doi:10.3772/j.issn.1002-0470.2015.03.011

# 一种基于标签和协同过滤的并行推荐算法①

祝晓斌② 蔡 强 白 璐 李海生

(北京工商大学计算机与信息工程学院 北京 100048)

**摘 要** 针对基于用户打分的传统协同过滤推荐算法存在准确率较低以及计算延时的问题,提出了一种基于标签与协同过滤的并行混合推荐算法。该算法通过计算标签的词频-逆文档频率(TF-IDF)值降低流行标签的权重,根据用户的历史行为预测用户对其他资源的偏好值,最后依据预测偏好值排序产生 Top-N 推荐结果。对该算法的计算效率与复杂度进行了理论分析,并且通过并行编程模型 MapReduce 使其得到了实现,最后在实验中进行了它与 Apache 软件基金会项目 Mahout 的协同过滤算法的对比分析。实验结果表明该算法有较高的准确性,能有效地提高推荐效率。

**关键词** 协同过滤, 推荐, 标签, TF-IDF, MapReduce

## 0 引 言

Web2.0 的发展使人类进入了信息爆炸的时代,用户在海量数据中挖掘符合自身个性化需求的资源需花费较大精力,个性化推荐系统的出现可有效地改变这种情况,个性化推荐系统可通过智能推荐算法预测用户的偏好和需求,向用户推荐个性化信息。推荐系统目前应用较多的算法是协同过滤算法,可以分为基于项目的协同过滤(item-based collaborative filtering, IBCF)算法和基于用户的协同过滤(user-based collaborative filtering, UBCF)。基本思路是先根据用户的历史行为计算资源或者用户之间的相似度。IBCF 算法则是向用户推荐与用户历史购买行为最相近的前 K 个资源,而 UBCF 算法则是找出与用户购买行为最相近的用户推荐该用户最频繁的 K 个资源。该算法由 Resnick 和 Lacovo 在 1994 年提出[1],并迅速受到学术界和工业界的关注,成为推荐系统的主流算法。随着互联网的发展,该算法由于资源的膨胀造成相似度矩阵过于稀疏,单节点的计算效率已经不能满足系统的需求,而且对新加入的资源无法完成准确的推荐,造成算法冷启动问题。针对以上问题,文献[2]提出了一种基于项目评分预测的协同过滤推荐算法,它根据资源间相似性初步预测对未评分资源的评分,并采用一种新颖的相似性度量方法计算目标用户的最近邻居。文献[3]提出了一种基于矩阵分解模型的协同过滤算法,它虽能提高推荐效果,但未能很好地解决数据稀疏性问。文献[4]针对单一评分相似度计算提出了一种基于用户间多相似度的协同过滤算法,它基于用户间不同项目类型的多个评分相似度来计算用户对未评分项目的预测评分。文献[5]通过对稀疏评分矩阵填充来提高用户相似度度量效果和系统推荐精度,算法使用最近邻算法进行推荐,分析传统相似度和基于云模型的相似度方法优化后的度量效果,分别为各填充方法选取最有效的相似度优化方案。

现有的推荐算法忽视了用户、资源自身的特征,存在冷启动问题,而且基于用户评分的协同过滤算法存在准确度不高、效率低等问题。为解决上述问

① 国家自然科学基金(61402023),北京市自然科学基金(4132025)和北京市教师队伍建设青年英才计划(YETP1448)资助项目。
② 男,1981 年生,博士,讲师;研究方向:模式识别,视频分析等;联系人,E-mail: buddysoft@sina.com
(收稿日期:2015-01-29)

题,本文提出了一种基于标签和协同过滤的并行推荐算法(Parallel recommendation algorithm based on Tagsing and Collaborative Filtering),简称 PTCF 算法。该算法利用标签可自由标注的特性[6],依据标签计算用户偏好程度和资源特征相似度,结合协同过滤算法推荐资源,并利用软件平台 Hadoop 实现推荐算法并行化以提高资源推荐效率。

## 1 基于标签的资源相似度计算

### 1.1 用户对资源的偏好

基于标签的推荐算法[7],利用标签的自由标注特性,通过用户使用标签标记资源的记录,从用户、资源两个角度挖掘对资源的喜爱程度,实现个性化资源推荐[8]。鉴于此,将标签作为用户偏好模型特征、资源特征,计算用户标签特征向量和资源标签特征向量,利用特征向量计算用户对资源偏好程度及资源相似度。并在计算用户标签特征和资源[9,10]标签特征的过程中借助了信息检索领域的词频-逆文档率(TF-IDF)思想,对较流行的标签和资源降低权重,以保证推荐质量。

定义用户集合 User = $\{u_1,u_2,\cdots,u_j,\cdots,u_N\}$,其中 $N$ 为用户总数,$j = 1,2,\cdots,N$; 所有资源的集合为 Item = $i_1,i_2,\cdots,i_j,\cdots,i_P$,其中 $P$ 为资源总数,$j = 1,2,\cdots,P$; 用户使用的标签集合为 Tag = $\{t_1,t_2,\cdots,t_j,\cdots,t_M\}$,其中 $M$ 为标签总数,$j = 1,2,\cdots,M$。用户的特征向量是用户点击的资源以及资源对应的 TF-IDF 值,用户的特征向量记为

$$\overrightarrow{\boldsymbol{V}_{ut_l}} = tf_{user_{tag}} \times idf_{user_{tag}} \left\{ \frac{n_{ut_1}}{n_{ut}} \log\left(\frac{N}{N_{ut_1}}\right) \cdots \frac{n_{ut_i}}{n_{ut}} \log\left(\frac{N}{N_n}\right) \cdots \frac{n_{ut_M}}{n_{ut}} \log\left(\frac{N}{N_{ut_M}}\right) \right\} \tag{1}$$

计算标签的频率和标签对用户的流行程度采用,$tf_{use\_tag} = \dfrac{n_{ut_i}}{n_{ut}}, idf_{uer\_tag} = \log\left(\dfrac{N}{N_{ut_M}}\right)$,$n_{ut_i}$ 表示某一个用户使用标签 $t_i$ 的次数,$n_{ut_i}$ 表示某一个用户使用的标签的个数,$N$ 表示数据集中用户总数,$tf_{user_{tag}} \times idf_{user_{tag}}$ 表示标签对用户的重要程度。

### 1.2 资源的标签特征向量

资源的标签特征向量是表示每个标签对资源的重要程度,值采用标签的 TD-IDF 表示,记为

$$\vec{\boldsymbol{V}}_{it_j} = tf_{item_{tag}} \times idf_{item_{tag}} \left\{ \frac{n_{it_1}}{n_{it}} \log\left(\frac{P}{n_{it_1}}\right) \cdots \frac{n_{it_j}}{n_{it}} \log\left(\frac{P}{N_{it_j}}\right) \cdots \frac{n_{it_M}}{n_{it}} \log\left(\frac{P}{N_{it_M}}\right) \right\} \tag{2}$$

标签相对资源被使用的频率和标签对资源的重要程度使用 $tf_{item_{tag}} = \dfrac{n_{it_j}}{n_{it}}$ 和 $idf_{item_{tag}} = \log\left(\dfrac{P}{N_{it_M}}\right)$ 表示,其中 $n_{it_j}$ 表示资源 $i$ 表示被标签 $t_j$ 标记的次数,$P$ 表示资源的总数,$n_{it}$ 表示资源 $i$ 的总的标签数。

用户对资源偏好矩阵可以用矩阵相乘计算得到:

$$\vec{\boldsymbol{V}}_{u_j i_k} = \sum_{t=1}^{M} \vec{V}_{u_j t} \times \vec{V}_{i_k t} \tag{3}$$

其中 $u_j \in U, j = 1,2,\cdots,N$; $i_k \in I, k = 1,2,\cdots,P$。用户 $u_j$ 的资源偏好特征向量表示为

$$\vec{\boldsymbol{V}}_{u_j} = (V_{u_j i_1}, V_{u_j i_2}, \cdots, V_{u_j i_k}, \cdots, V_{u_j i_N}) \tag{4}$$

其中 $V_{u_j i_k}$ 表示用户 $u_j$ 对资源 $i_k$ 的喜爱程度。

最后可以依据用户对资源的偏好向量,构造用户-资源偏好矩阵,记为

$$\boldsymbol{V}_{N\times P} = \begin{pmatrix} V_{u_1 i_1} & \cdots & V_{u_1 i_k} & \cdots & V_{u_1 i_P} \\ \vdots & \ddots & & & \vdots \\ V_{u_j i_1} & & V_{u_j i_k} & & V_{u_j i_P} \\ \vdots & & & \ddots & \vdots \\ V_{u_N i_1} & \cdots & V_{u_N i_k} & \cdots & V_{u_N i_P} \end{pmatrix} \tag{5}$$

其中,$u_j \in U, j = 1,2,\cdots,N; i_k \in I, k = 1,2,\cdots,P$; 该矩阵记录了用户的兴趣爱好向量,且能反映用户对各资源的喜爱程度。

### 1.3 资源相似度计算

资源相似度反映了两个不同资源之间的相似程度,协同过滤算法的传统相似度计算是根据计算的用户之间或者资源之间具有相同信息的个数,作为相似度的一个标准,这种方法忽略了标签本身自带的信息。本文利用标签信息计算资源相似度[11],并采用余弦相似度作为相似度度量方法,依据资源的历史信息也利于发现新的资源。

资源的特征信息可用基于标签的资源特征向量 $\overrightarrow{\boldsymbol{I}_k}$ 表示:

$$\overrightarrow{\boldsymbol{I}_k} = (n_{k1}, n_{k2}, \cdots, n_{ki}, \cdots, n_{kL}) \tag{6}$$

其中 $k = 1,2,\cdots,N$; $I = 1,2,\cdots,P$; $n_{ki}$ 表示 $t_i$ 被用

来标记资源 $i_k$ 数归一化后的值。

所有资源的特征信息可用资源特征向量矩阵 $\boldsymbol{I}_{k\times k}$ 表示:

$$\boldsymbol{I}_{k\times k}=\begin{bmatrix} n_{11} & \cdots & n_{1k} \\ \vdots & \ddots & \vdots \\ n_{k1} & \cdots & n_{kk} \end{bmatrix} \tag{7}$$

资源的相似度计算有多种方式,本文采用的是余弦相似度计算。通过资源特征向量计算资源间的余弦相似度:

$$\mathrm{sim}(i_j,\ i_k)=\cos(\overrightarrow{I_J},\ \overrightarrow{I_K})=\frac{\overrightarrow{I_J}\cdot\overrightarrow{I_K}}{|\overrightarrow{I_J}|\times|\overrightarrow{I_K}|} \tag{8}$$

通过计算资源间的相似度,可构造资源相似度矩阵 $\boldsymbol{S}_{P\times P}$,用以描述不同资源间的相似度:

$$\boldsymbol{S}_{P\times P}=\begin{pmatrix} 1 & \cdots & s_{1j} & \cdots & s_{1P} \\ \vdots & \ddots & & & \vdots \\ s_{j1} & & 1 & & s_{jP} \\ \vdots & & & \ddots & \vdots \\ s_{P1} & \cdots & s_{Pj} & \cdots & s_{PP} \end{pmatrix} \tag{9}$$

其中 $j=1,2,\cdots,P$;$s_{ij}$ 表示资源 $i_i$ 和 $i_j$ 的相似度。

### 1.4 预测偏好值计算

依据用户历史行为及资源相似度,可计算用户 $u$ 对未使用资源 $i_j$ 的偏好程度,并用预测偏好值表示:

$$pp_{u_{ij}}=\sum_{k=1}^{N}p_{u_ji_k}\times s_{i_ki_j} \tag{10}$$

其中 $p_{u_ji_k}$ 表示用户 $u_j$ 对历史使用资源 $i_k$ 的偏好程度,$s_{i_ki_j}$ 表示资源 $i_k$ 和 $i_j$ 的相似度。

预测偏好值通过用户使用的历史资源 $i_k$,计算各历史资源与资源 $i_j$ 的相似度,进而求得用户 $u$ 对 $i_j$ 的偏好程度。充分利用用户历史行为和资源相似度,提高了推荐准确度。

## 2 基于标签和协同过滤算法的设计

设 user _ tags$(u,t)$ 为用户-标签矩阵,表示用户 $u$ 打过标签 $t$ 的次数;tag _ item$(t,i)$ 为标签-资源矩阵,表示资源 $i$ 被标签 $t$ 标记的次数;ntu 为标签 $t$ 被不同 $u$ 使用的用户数,nit 为物品 $i$ 被不同的 $t$ 标记的次数。则基于标签的协同过滤算法表述如下:

第 1 步:通过数据集中的用户-标签关系统计 user _ tags 和 ntu,通过标签-资源关系统计 tag _ items 和 nit;

第 2 步:计算用户对资源的偏好矩阵。首先分别依据式(1)、(2)计算用户的标签特征向量和资源的标签特征向量,依据式(3)计算用户偏好向量,并构建用户-资源偏好矩阵;

第 3 步:根据式(8)计算资源相似度,并构造资源相似度矩阵;

第 4 步:基于用户对资源的历史记录,查询用户 $u$ 曾标记的资源与其他资源的相似度,并用式(10)依次计算用户与这些资源相似资源的预测偏好值;

第 5 步:按预测偏好值从大到小排序,并取前 $N$ 个资源组成 Top-N 推荐集输出。

并行推荐算法的思想是利用编程框架 MapReduce[11] 计算用户和资源的标签特征向量、用户对资源偏好及资源相似度,并对用户未访问过的资源预测偏好值,最后为用户生成符合其个人需求的推荐结果。MapReduce 编程框架可以分为 Map 和 Reduce 过程,分别是通过 map 和 reduce 函数实现,并且 Map 任务和 Reduce 任务的输入和输出都是(key,value)对。Map 过程分发数据到各个工作节点,每次调用都会产生一个(key,value)队列;当所有的 Map 任务完成后,MapReduce 主控进程会按 Map 任务输出的 key 进行分组,并输出到特定的 Reduce 任务,将所有结果组合并输出。算法设计如图 1所示。

依据算法流程,该算法包括如下 4 个 job 任务:

(1)job1:用户及资源的标签特征向量计算,包括两个子任务,分别为用户标签特征向量和资源标签特征向量,该步骤通过 6 个 mapreduce 程序实现;

(2)job2:资源相似度计算,该步骤通过一个 mapreduce 程序实现;

(3)job3:对用户未访问资源的偏好值预测,该步骤通过 3 个 mapreduce 程序实现;

(4)job4:生成 Top-N 推荐结果,该步骤通过一个 mapreduce 实现。并行推荐算法如图 2 所示。

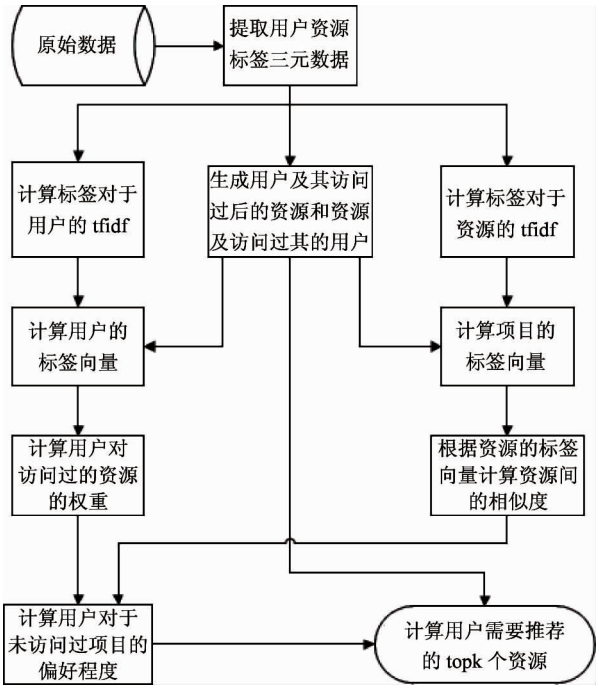


**图 1 并行推荐算法流程**

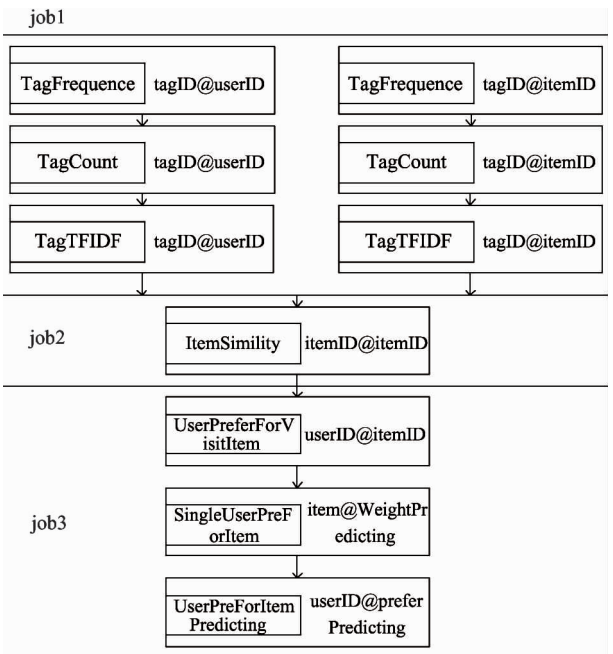


**图 2 并行推荐算法 mapreduce 图**

结合特点及 Hadoop 优势,基于 Hadoop 分布式计算平台离线计算用户、资源特征向量及资源间相似度产生推荐结果,以实现对大规模用户、资源数据计算的优化,确保推荐系统实效性,提高推荐效率。

## 3 实验结果与分析

### 3.1 推荐算法数据集与实验环境

实验采用 Delicious 数据集,该数据集有 1892 个用户,11946 个标签和 17632 个资源,共有 160 万行记录。本实验对数据进行预处理,依据各标签被使用过 10 次以上的原则选取记录作为数据集,数据集中随机随机选取 90% 作为训练样本,其余为测试集样本。实验采用 Hadoop 分布式集群,RedHat6.0 操作系统,1 个主节点,9 个从节点,节点硬件配置表 1所示。

**表 1 实验环境配置**

| 参数 | 主节点 | 从节点 |
|---|---|---|
| CPU | Xeon E5620 | Xeon E5620 |
| 内存 | 24Gb | 12Gb |
| 硬盘 | 1Tb | 300Gb |

### 3.2 推荐质量的评价标准

实验采用查全率(Recall)和查准率(Precision)作为评估准则。查准率表示算法推荐的结果中满足用户喜好的概率,查全率表示算法的所有推荐结果中与用户喜好相关的概率。查准率与查全率还将被进一步地合并为 F-measure[12] 指标,用来综合考虑两者的结果,以更加直观地反映算法的性能。设 $R(u)$ 是根据用户在训练集上的行为给用户作出的推荐列表, $T(u)$ 是用户在用户测试集上的行为列表。那么推荐查准率定义为

$$\text{Precision} = \frac{\sum_{u \in U} | R(u) \cap T(u) |}{\sum_{u \in U} | R(u) |} \tag{11}$$

推荐查全率定义为

$$\text{Recall} = \frac{\sum_{u \in U} | R(u) \cap T(u) |}{\sum_{u \in U} | T(u) |} \tag{12}$$

F-measure 定义为

$$\text{F-measure} = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{13}$$

### 3.3 结果对比

实现了协同过滤推荐算法,包括基于项目和基

于用户的协同过滤推荐算法。本文研究的是利用标签的并行推荐算法,因此本研究采用 Apache 软件基金会(Apache Software Foundation,ASF) 开发的开源项目 Mahout 实现的协同过滤推荐算法——MUT 算法和 MIT 算法与本文提出的基于标签和协同过滤的并行推荐算法即 PTCF 算法进行了比较。MUT 算法以标签对用户的重要程度代表将用户对资源的偏好项,MIT 算法以标签对资源的重要程度代表将用户对资源的偏好项,并利用 Mahout 基于资源的协同过滤算法进行推荐。用户行为仅在求项目相似度时使用标签的算法。并行推荐算法和 Mahout 对比实验结果见图 3 ~ 图 6。

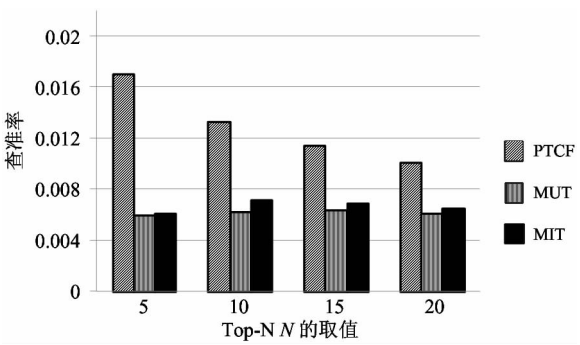


**图 3　不同 *N* 值下的查准率值**

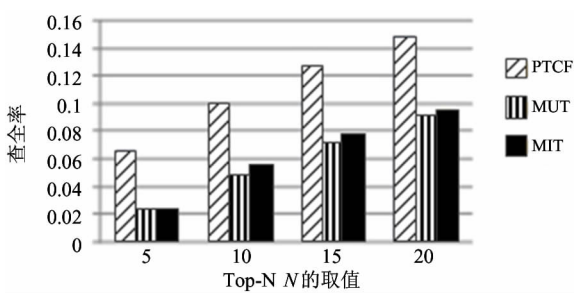


**图 4　不同 *N* 值下的查全率值**

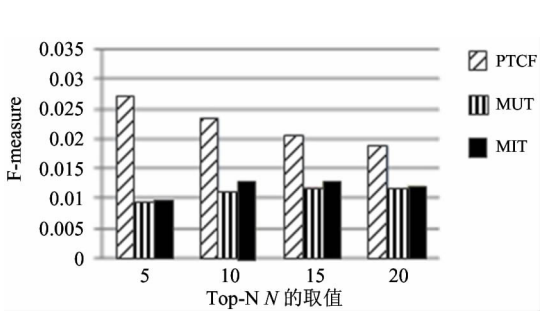


**图 5　不同 *N* 值下的 F-measure 值**

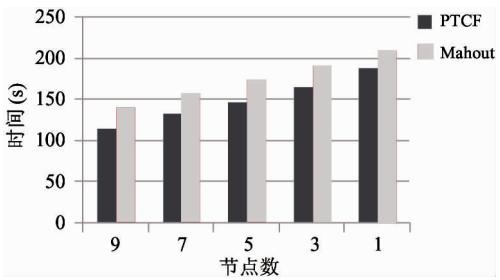


**图 6　不同 Hadoop 节点数的推荐效率**

从图 3、图 4 和图 5 可以看出,与基于 Mahout 的协同过滤推荐算法相比,本文提出的 PTCF 算法在准确度、查全率和 F-measure 都有较大提高。准确度随着 Top-N 中 *N* 值的逐步增大而减小,查全率随着 Top-N 中 *N* 值的逐步增大而增大,而 F-measuer[13] 同样是随着 *N* 值的增大而减小,可以看出准确度对推荐算法的质量影响较大。从实验结果可看出,处理大小不同的数据集文件推荐算法响应时间不同,一般文件越大,算法执行时间越长;对于同一数据集文件,增加 Hadoop 工作节点,可以有效提高推荐算法的效率,且相比较数据规模小的数据集,执行效率对大规模数据集的影响较小。该算法需要对数据集进行预处理清洗,耗费时间较长,但在推荐系统实际运用中,可离线计算相似度等数据,大大减少了实时推荐时间,提高系统的推荐效率和用户满意度。由实验结果可得出,借助标签的基于 Hadoop 的数字媒体分布式推荐算法的推荐质量有所提高,可扩展性也较强。

## 4　结　论

社会化标签具有自由标注的特性,本文结合基于资源的协同过滤思想,提出了一种满足用户个性化需求的并行推荐算法。该算法使用标签作为用户兴趣偏好及资源特征,既能提高推荐算法的质量,又能提供推荐解释,还能将算法应用到其他类似的标注模型中,具有一定的普适性;利用资源的标签特征向量计算资源相似度,可解决传统协同过滤算法的冷启动问题。利用 Hadoop 平台实现算法并行化,提高了算法的推荐效率,相比较基于图的推荐算法,该算法复杂度较低。但该方法在实际应用中仍有若干

问题有待解决,如标签的质量对推荐质量的影响等。

**参考文献**


[1] Resnick P, Iacovou N, Suchak M, et al. GroupLens: an open architecture for collaborative filtering of netnews. In: Proceedings of the ACM Conference on Computer Supported Cooperative Work, North Carolina, USA. 1994. 175-186

[2] 张斌,张引,高克宁等. 融合关系与内容分析的社会标签推荐. 软件学报, 2012, 23(3): 476-488

[3] Koren Y, Bell R, Volinsky C. Matrix Factorization techniques for recommender systems. *IEEE Computer Society*, 2009,42(8):30-37

[4] 范波, 程久军. 用户间多相似度协同过滤推荐算法. 计算机科学, 2012,39(1):23-26

[5] 罗辛,欧阳元新,熊璋等. 通过相似度支持度优化基于K近邻的协同过滤算法. 计算机学报,2010,33(8): 1437-1445

[6] Koren Y, Sill J. Collaborative filtering on ordinal user feedback. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, 2013. 3022-3026

[7] Wang J, Lin K, Li J. A collaborative filtering recommendation algorithm based on user clustering and Slope One scheme. In: Proceedings of the ICCSE, Colombo, Sri Lanka, 2013. 1473-1476

[8] Cai Y, Leung H, Li Q, et al. Typicality-Based Collaborative FilteringRecommendation. *IEEETransactions on Knowledge and Data Engineering*, 2013, 26(3):766-779

[9] Cechinel C, Sicilia M á, Sánchez-Alonso S, et al. Evaluating collaborative filtering recommendations inside large learning object repositories. *Information Processing & Management*, 2013, 49(1): 34-50

[10] Zhang X. Interactive patent classification based on multi-classifier fusion and active learning. *Neurocomputing*, 2014, 127(3):200-205

[11] Li L, Li C, Chen H, et al. MapReduce-based SimRank computation and its application in social recommender system. In: Proceedings of the International Congress on Big Data, Santa Clara, USA, 2013. 133-140

[12] 赵琴琴,鲁凯,王斌. SPCF:一种基于内存的传播式协同过滤推荐算法. 计算机学报,2013,36(3): 671-676

[13] Wang H, Chen B, Li W J. Collaborative topic regression withsocial regularization for tag recommendation. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, 2013. 2719-2725


# A parallel recommendation algorithm based on tagging and collaborative filtering


Zhu Xiaobin, Cai Qiang, Bai Lu, Li Haisheng

(School of Computer and Information, Beijing Technology and Business University, Beijing 100048)



**Abstract**

The study focused attention on the problems of lower precision and computing latency of traditional collaborative filtering recommendation algorithms, and proposed a parallel hybrid recommendation algorithm based on tagging and collaborative filtering. The algorithm reduces the weight of prevalent tags by calculating the TF-IDF (time frequency-inverse document frequency) value of tags on predicts user preference based on the user historical behaviors, and finally recommends the Top-N of the predictions. The algorithm's computation efficiency and complexity were theoretically analyzed, and it was implemented by using the parallel programing model of MapRedce. The analytical comparison of the algorithm with the collaborative filtering algorithm applied to the Mahout, an item of the Apache Software Foundation, was conducted, and the result showed its higher accuracy, so it can effectively improve the recommendation efficiency.



**Key words**: collaborative filtering, recommendation algorithm, tag, TF-IDF, MapReduce