云环境下 top-n 推荐算法

黄震华[1,2]
(1.同济大学计算机科学与工程系,上海 200092;2.同济大学嵌入式系统与服务计算教育部重点实验室,上海 200092)

摘 要: Top-n 推荐技术是近年来信息服务领域的一个研究重点和热点.针对云环境下的 top-n 推荐算法进行了深入研究,提出了适合 top-n 推荐的多层分布式存储架构 MDSA(Multilayer Distributed Storage Architecture),并从降低网络传输代价出发,设计了基于 MDSA 架构的数据编码模式,进而利用 map/reduce 分布式编程模型来快速实现 top-n 推荐.此外,为了满足实际的需求,给出了三种 top-n 推荐的应用扩展.理论分析和实验结果表明,本文所提的方法具有有效性和实用性.

关键词: 云数据; top-n 推荐; mapreduce; 信息服务
中图分类号: TP311.13 **文献标识码:** A **文章编号:** 0372-2112 (2015)01-0054-08
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2015.01.009

Top-n Recommendation Algorithms for Cloud Data

HUANG Zhen-hua[1,2]
(1. *Department of Computer and Technology, Tongji University, Shanghai* 200092, *China*;
2. *The Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai*, 200092, *China*)

Abstract: Top-n recommendation technology has recently received a lot of attention in information service community. In this paper, we study the problem of top-n recommendation under the cloud data. Firstly, we propose a multilayer distributed storage architecture(MDSA) which can fit the need of top-n recommendation. Secondly, based on the consideration of reducing the cost of network transmission, we present the data encoding pattern which is based on the MDSA architecture. Lastly, we fast realize top-n recommendation using the map/reduce distributed programming model. For satisfying the realistic requirement, we further present three application variants of top-n recommendation. Massive experiments demonstrate that our approach is efficient and has high overall performance.

Key words: cloud data; top-n recommendation; mapreduce; information service

1 引言

目前,top-n 推荐技术成为信息服务领域的一个研究重点和热点[1].给定有限项集合 M,其中每个项具有 k 个属性,每个属性衡量它的一个子特征(如价格、保质期等),top-n 推荐就是根据用户偏好函数 f,选择出 f 取值最高的 n 个项作为推荐集合.

近年来,国内外学者对 top-n 推荐技术进行了深入研究.文献[2]利用关联规则发现算法来完成 top-n 项的推荐.文献[3]通过分析不同项间的相似度,然后基于相似对象来识别需要推荐的 n 个项.文献[4]利用可信网络来提高 top-n 推荐的质量,基于可信网络的随机漫步算法来精确预测离散项间的相似度,从而有效提高 top-n 推荐的效果.文献[5]针对数据稀疏问题,提出了一种能够提高 top-n 推荐质量的有效方法,该方法首先识别先验预测的错误率,然后构造"用户-项"错误矩阵来进行 top-n 推荐.文献[6]考虑在维持 top-n 推荐质量的基础上,进行优化推荐对象的多样性,从而使得用户能够获得更为丰富的推荐集合.文献[7]将用户和项间的购买关系建模为一张双向图,并且利用信任传播来计算用户对购买项的偏好,从而提高 top-n 推荐的效果.针对新用户问题,文献[8]提出了一种基于 n 序访问解析逻辑的冷启动消除方法.该方法首先通过 web 日志来获取项序,进而定义了 n 序访问解析逻辑将其分解为用户访问子序集,并在此基础上设计了项序相似性计算框架来搜寻新用户的最近邻集合,进而生成面向新用户的 top-n 推荐.

收稿日期:2013-09-22;修回日期:2014-01-02;责任编辑:孙瑶
基金项目:国家自然科学基金(No.61272268);教育部新世纪优秀人才支持计划(No.NCET-12-0413);国家 973 基础研究发展计划(No.2014CB340404);霍英东教育基金会应用研究课题(No.142002);同济大学中央高校基本科研业务费专项资金

文献[2~8]主要考虑集中式处理 top-n 推荐.

随着云计算技术的深入应用,大多数电子商务企业将数据分布式存储于云架构下,这主要是因为云环境能使推荐系统具有高可用性、容错性和现场迁移能力[9].文献[10]基于 R-树和 KD-树混合索引来组织云数据,并通过索引树的局部遍历来获取满足用户偏好的前 n 个项.文献[11]在 R-树和 CAN(Content Addressable Network)路由协议的基础上,提出了 RT-CAN 多维索引来组织云数据,并通过 R-树节点和 CAN 节点的映射来有效支持 top-n 推荐.文献[12]将云架构中的计算机节点组织成一个结构化的 P2P 分布式网络 BATON(Balanced Tree Over-lay Network),并且基于 B^+-树索引来组织云数据,从而实现有效的 top-n 推荐.文献[13]提出了更新和查询平衡的 UQE(Update and Query Efficient)树来组织云环境下的键-值型数据,并通过索引树的局部遍历来进行 top-n 推荐.文献[14]给出了 KR-树来有效组织云数据,并设计了 CTC1 + 采样评估器来近似获取满足用户偏好的前 n 个项.同时,作者证明了 CTC1 + 采样评估器的近似下界为 $1-1/e\approx 63.2\%$.然而我们发现现有的分布式方法在处理 top-n 推荐时需要大量的网络传输代价、I/O 开销和 CPU 计算时间,特别当云数据容量较大时,这三方面的负荷更为突出,从而影响了云环境推荐系统的 top-n 推荐效率.

为了提高 top-n 推荐的效率,本文首先提出了一种全新的多层分布式存储架构 MDSA(Multilayer Distributed Storage Architecture),该架构将云环境下的网络节点组织成一棵层次索引树,并将数据按照特定的规则分割存储于层次索引树的各节点上.其次,基于 MDSA 架构,本文设计了适合 top-n 推荐的数据编码模式,来有效降低网络传输代价和 I/O 开销.最后,为了降低系统响应时间,利用目前成熟的 map/reduce 分布式编程模型[15]来快速获取满足用户偏好的前 n 个项.理论分析和实验结果表明,本文所提的方法具有有效性和实用性.

2　MDSA 分布式存储架构

在这一节中,我们给出用于 top-n 推荐的多层分布式存储架构 MDSA,如图 1 所示.

在 MDSA 架构中,我们将推荐系统的数据分割存储于 $u+1$ 层的索引树上.第 0 层为根节点 R,它存储推荐系统中用户偏好属性 $\mathrm{PA}=(\mathrm{PA}_1,\cdots,\mathrm{PA}_m)$ 上的数据集 $\Omega_{\mathrm{PA}}=\{r_1,\cdots,r_\lambda\}$,$\Omega_{\mathrm{PA}}$中的每条记录对应一个项的所有偏好属性;索引树上其余的 u 层对应 u 个特征属性抽象层次,第一层具有 k_1 个子节点,每个子节点 $N(1,R,i)$,$i\in[1,k_1]$,存储 t_i 个具有最高抽象层次的特征属性 $\mathrm{CA}_i=(\mathrm{CA}_{1R_i,1},\cdots,\mathrm{CA}_{1R_i,t_i})$ 上的数据 $\Omega_{\mathrm{CA}_i}=\{r_1,\cdots,r_\partial\}$;而第 u 层具有 k_u 个子节点,每个子节点 $N(1,P_{u-1},j)$,$j\in[1,k_u]$,存储 w_j 个具有最低抽象层次的特征属性 $\mathrm{CA}_j=(\mathrm{CA}_{uP_{u-1},1},\cdots,\mathrm{CA}_{uP_{u-1},w_j})$ 上的数据 $\Omega_{\mathrm{CA}_j}=\{r_1,\cdots,r_\eta\}$.为了便以描述,我们称该 $u+1$ 层索引树为 MDSA 索引树.

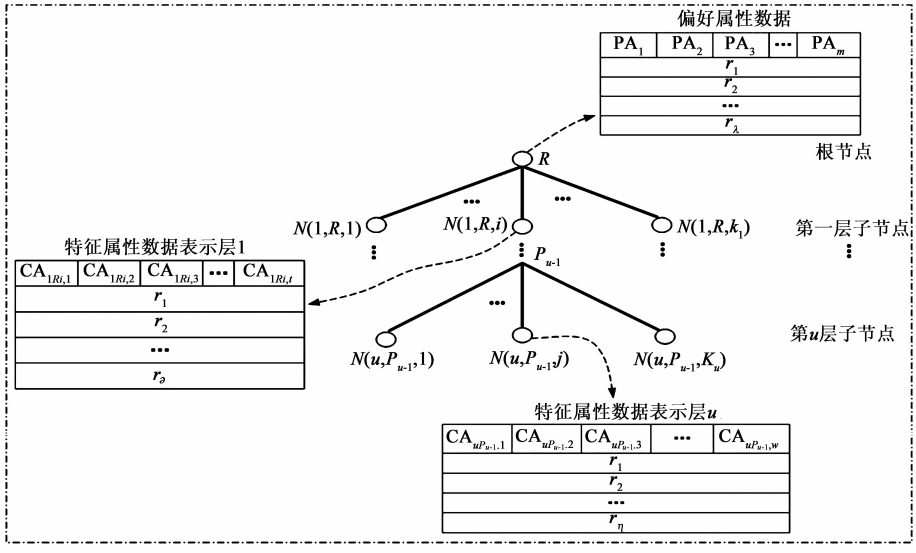


图1　MDSA多层分布式存储架构

假定用户对项的偏好属性 $\mathrm{PA}'\subseteq\mathrm{PA}$ 以及第 i 个抽象层次的特征属性 $\mathrm{CA}'\subseteq\mathrm{CA}_i$ 感兴趣,那么推荐系统首先从根节点 R 上抽取与偏好属性 PA′相关的数据;然后从第 i 层与特征属性 CA_i 对应的节点上抽取出与 CA′相关的数据,并将它传送到根节点 R 上;最后在 R 上,对这两类数据进行连接,以及按照用户偏好函数 $f(\bar{A})$ 进行聚合操作,并将取值最高的 n 个项作为推荐集合返回给用户.

值得注意的是,MDSA 索引树属性抽象与 deep learning 属性抽象的区别:MDSA 索引树的特征属性抽象

与偏好属性相对应,每个特征属性抽象表示推荐项的一个子特征(如价格、保质期等),这些子特征是已知的,并是准确的;而 deep learning 是包含多隐层感知器的一种深度学习结构,通过组合低层特征形成更加抽象的高层特征,主要是利用深度学习的方法来获取能够描述文本、图像等的特征,并最终进行正确的分类,这些特征是未知的,且是不准确的.

不难发现,在 MDSA 架构中,根节点 R 的 k_1 棵子树上的网络节点不必相互数据传输,他们只需与根节点交互即可,这样可以节省大量的网络传输代价.

3 数据编码模式

为了进一步缩减网络传输代价以及降低 I/O 和 CPU 开销,本节给出适合云数据 top-n 推荐的数据编码模式.

定义 1(项成员簇) 假定推荐系统项的集合为 $RS=\{i_1,\cdots,i_w\}$,其上的 MDSA 索引树 MDSA-Tree 具有 $u+1$ 个层(第 0 层为根节点),则它有 u 个特征属性抽象层次,与之对应的特征属性组为 $CAC=\{CA_1,\cdots,CA_u\}$. 如果 $IG=(ig_1,\cdots,ig_\varphi)$ 满足下列条件,那么我们称 IG 为 MDSA-Tree 的第 $h(1\leqslant h\leqslant u)$ 层特征属性 CA_h 的项成员簇:

(1) $\mathrm{depth}(ig_x)=h,(1\leqslant x\leqslant\varphi)$;

(2) $ig_x\ RS,(1\leqslant x\leqslant\varphi)$;

(3) $CA_h=\bigcup_{y=1}^{\varphi}(CA(ig_y))$;

(4) $\forall\ ig_x,ig_y\in IG, ig_x\bigcap ig_y=\varnothing(1\leqslant x,y\leqslant\varphi)$;

其中 $\mathrm{depth}(ig_x)$ 为 ig_x 所在的特征属性抽象层次,$CA(ig_x)$ 为 ig_x 的特征属性.

不失一般性,我们把第 h 特征属性层的第 x 项成员记为 ig_x^h. 不难看出,$RS=\bigcup_{h=1}^{u}\bigcup_{x=1}^{\phi}(ig_x^h)$. 另外,由定义 1 可知,处于同一特征属性层上的各项成员所表示的项集不相互重叠.

定义 2(项成员家族) 具有 u 个特征属性抽象层次的 MDSA 索引树 MDSA-Tree 上的项成员构成了 u 代家族,其中对于家族中第 h 代项成员 ig_x^h,它的子项成员集 $C(ig_x^h)$ 和父项成员集 $P(ig_x^h)$ 可分别表示为:

$$C(ig_x^h)=\begin{cases}\varnothing, & h=u\\ \{ig_x^{h+1}\subseteq ig_x^h\mid CA(ig_x^{h+1})\in CA_h\}, & h<u\end{cases},$$

$$P(ig_x^h)=\begin{cases}\varnothing, & h=1\\ \{ig_x^{h+1}\supseteq ig_x^h\mid CA_h\in CA(ig_x^{h-1})\}, & h>1\end{cases}$$

定义 3(子项成员映射) 假定项成员 ig_x^h 的子项成员集 $C(ig_x^h)$ 中包含 g_x^h 个子项成员,那么定义双射函数 $BF_x^h:C(ig_x^h)\to\{0,1,\cdots,g_x^h-1\}$. BF_x^h 为项成员 ig_x^h 的每个子项成员 $ig_x^{h+1}\in C(ig_x^h)$ 赋予一个互不相同的有序码位,从而定义了一种映射模式.

定理 1(映射时空间复杂度) 假定推荐系统的 MDSA 索引树 MDSA-Tree 具有 u 个特征属性抽象层次,第 $h(1\leqslant h\leqslant u)$ 层具有 m_h 个项成员 $ig_1^h,\cdots,ig_{m_h}^h$,而对于每个 $ig_x^h(1\leqslant x\leqslant m_h)$,它的子项成员集 $C(ig_x^h)$ 包含 g_x^h 个子项成员,那么 MDSA-Tree 完成映射的时间复杂度 T 和空间复杂度 S 分别为 $O(\sum_{h=1}^{u}\sum_{x=1}^{m_h}g_x^h)$ 和 $O\left(\sum_{h=1}^{u}\sum_{x=1}^{m_h}(g_x^h\cdot\log(g_x^h))\right)$,从而可知,定义 1 成立.

证明 从定义 3 可知,对于 MDSA-Tree 的一个项成员 ig_x^h,双射函数 BF_x^h 为 $C(ig_x^h)$ 中子项成员完成映射的时间为 $O(g_x^h)$,因此时间复杂度等于 $\sum_{h=1}^{u}\sum_{x=1}^{m_h}O(g_x^h)=O\left(\sum_{h=1}^{u}\sum_{x=1}^{m_h}g_x^h\right)$. 另一方面,由于 $C(ig_x^h)$ 包含 g_x^h 个子项成员,因此需 $O(\log(g_x^h))$ 个码位来完成映射,即需 $O(g_x^h\cdot\log(g_x^h))$ 的存储空间,所以空间复杂度等于 $\sum_{h=1}^{u}\sum_{x=1}^{m_h}O(g_x^h\cdot\log(g_x^h))=O\left(\sum_{h=1}^{u}\sum_{x=1}^{m_h}(g_x^h\cdot\log(g_x^h))\right)$.

定义 4(项成员编码) 假定推荐系统的 MDSA 索引树具有 u 个特征属性抽象层次,对于第 $h-1(2\leqslant h\leqslant u+1)$ 层的每个项成员 ig_x^{h-1},$C(ig_x^{h-1})$ 包含 g_x^{h-1} 个子项成员 $\{ig_1^h,\cdots,ig_{g_x^{h-1}}^h\}$,令 $C(ig_x^{h-1})\to\{0,1,\cdots,g_x^{h-1}-1\}$ 的双射函数等于 BF_x^{h-1},那么我们可以获得 MDSA-Tree 上每个项成员 $ig_j^h(1\leqslant j\leqslant g_x^h)$ 的编码值:

$$CV(ig_j^h)=\begin{cases}0, & h=1\\ BF_x^{h-1}(C(ig_x^{h-1}))\perp CV(ig_x^{h-1}), & h>1\end{cases}$$

其中,"$\perp$"为字符串连接符.

根据定义 4 可知,同一层的所有项成员编码长度是一样的,因此,我们可以用长度一致的二进制编码值来表示各项成员.

从定义 3 和 4 不难知道,MDSA 索引树上的各项成员均对应一个编码值. 接下来我们从理论上证明项成员编码值的唯一性.

定理 2(项成员编码值唯一性) MDSA 索引树上的各项成员编码值是唯一的.

证明 首先根据定义 3 可知,从第 1 层节点开始,每个项成员 ig_x^h 均赋予子项成员集 $C(ig_x^h)$ 一个唯一的双射函数 BF_x^h,而 BF_x^h 为 $C(ig_x^h)$ 中的每个子项成员 ig_x^{h+1} 指定一个互不相同的有序码位,因此对于同一层,项成员的编码是唯一的. 其次,根据定义 4 可知,从第 1 层节点开始,组合各层的编码而得到的全局编码值也是唯一的,从而可知,定理 2 成立.

定义 5(层辐射量) 假定推荐系统的 MDSA 索引树 MDSA-Tree 具有 u 个特征属性抽象层次,第 $h(1\leqslant h$

$\leqslant u$)层具有 m_h 个项成员 $ig_1^h,\cdots,ig_{m_h}^h$,对于项成员 ig_x^h($1\leqslant x\leqslant m_h$),$C(ig_x^h)$包含 g_x^h 个子项成员,那么第 h 层的层辐射量可表示为 $LRC^h=\log_2\max_{x=1}^{m_h}(g_x^h)$.

根据定义 5 可知,如果项成员 ig_x^h 和 ig_y^h 处以 MDSA 索引树的同一特征属性抽象层,那么它们所对应的层辐射量是相同的.

定义 6(MDSA 索引树路径编码) 假定推荐系统的 MDSA 索引树具有 u 个特征属性抽象层次,路径 PATH 遍历 MDSA 索引树 ζ 个层的项成员 $ig^1,\cdots,ig^\zeta$,对于 PATH 上的各节点 $ig^i(1\leqslant i\leqslant\zeta)$,它赋予子项成员集 $C(ig^i)$的双射函数为 BF^i,那么 PATH 的编码值为:

$$\begin{aligned}CV(PATH)&=BF^0(ig^1)+BF^1(ig^2)\cdot 2^{LRC^1}\\&\quad+BF^2(ig^3)\cdot 2^{LRC^1+LRC^2}+\\&\qquad\cdots\\&\quad+BF^{\zeta-1}(ig^\zeta)\cdot 2^{LRC^1+LRC^2+\cdots+LRC^{\zeta-1}}\\&=BF^0(ig^1)+\sum_{i=1}^{\zeta-1}BF^i(ig^{i+1})\cdot 2^{\sum_{j=1}^{i}LRC^j}.\end{aligned}$$

由定义 6 可知,MDSA 索引树上两条长度相同的路径,它们的编码长度也是一样的,因此,我们同样可以用长度一致的二进制编码值来表示所有长度相同的路径.

4 基于 map/reduce 的 top-n 推荐算法

在这一节中,我们在 MDSA 存储架构和数据编码模式的基础上,利用目前成熟的 map/reduce 分布式编程模型[15]来高效实施 top-n 推荐.

当用户提交偏好函数 $f(\bar{C},\bar{A},\mathfrak{D})$到 MDSA 索引树 MDSA-Tree 上的根节点 R 时,其中 $\bar{C}=(c_1,\cdots,c_e)$为 e 个项特征属性,$\bar{A}=(a_1,\cdots,a_w)$为 w 个项偏好属性,$\mathfrak{D}$ 为项推荐条件,根节点 R 将参数 $\bar{C}$ 和 $\mathfrak{D}$ 传递给与 $\bar{C}$ 相关的第一特征属性抽象层的 r 个节点 $N_1,\cdots,N_r$.此时,第 i 个($1\leqslant i\leqslant r$)节点抽取 $\mathfrak{D}$ 中与它相关的项推荐条件 $\mathfrak{D}_i$,并搜索项成员和路径编码文件来获取与 $\mathfrak{D}_i$ 对应的编码值 CV,然后将与 CV 相关的最大特征属性抽象层所包含的 x_i 个项编码值 $CV_1,\cdots,CV_{x_i}$传输给根节点 R.记这 x_i 个项编码值所构成的列表为 L_i.当 R 收到 r 个节点回送来的项编码值列表 $L_1,\cdots,L_r$ 后,R 在它的集群上调用 map/reduce 分布式编程模型来获取满足偏好的前 n 个项.具体过程如下:R 将 r 个项编码值列表分别分割成 M 份,然后每一台工作计算机分配 r 个子项编码值列表,map 函数的输入参数为项编码值构成的键值对,它将输出每个项和它在偏好属性数据上的标志位所构成的中间键值对;而 reduce 函数将依据用户的偏好对各项进行聚合操作,并返回前 n 个项给用户.

为了提高 map 函数的操作效率,我们提出了项知识格概念,定义 7 给出其形式化描述.

定义 7(项知识格) 假定与项特征属性 $\bar{C}=(c_1,\cdots,c_e)$相关的第一特征属性抽象层的 r 个节点为 $N_1,\cdots,N_r$,N_i 返回给根节点 R 的项编码值列表为 $L_i=\{CV_1,\cdots,CV_{x_i}\}$,而 x_i 个项编码值的标识符为 $F_1,\cdots,F_{x_i}$,令偏好属性数据表 ϖ 个项记录为 $PD_1,\cdots,PD_\varpi$,那么定义 N_i 与 R 间的项知识格为 ϖ 行和 x_i 列的二维矩阵 $\bar{\boldsymbol{Z}}[a,b]$,各分量计算如下:

$$\bar{\boldsymbol{Z}}[a,b]=\begin{cases}1, & \text{if } F_a\in PD_b\\0, & \text{otherwise}\end{cases}$$

不难看出,r 个第一特征属性抽象层节点对应 r 个项知识格 $\bar{Z}_1,\cdots,\bar{Z}_r$.我们可以将推荐条件 $\mathfrak{D}$ 在编码文件和路径编码文件上的查询操作,归约为两个及以上的项知识格的"位与操作"和"位或操作",来缩减获取每个满足推荐条件 $\mathfrak{D}$ 的项的时间开销.

接下来,我们具体给出 top-n 推荐算法 TRAMR(Top-n Recommendation Algorithm based on Map/Reduce),如算法 1 所示.

算法 1　TRAMR

输入:MDSA 索引树 MDSA-Tree;根节点 R 上的偏好属性数据表 PD;α 个第一特征属性抽象层节点的项成员和路径编码文件 $CF_1,\cdots,CF_\alpha$;用户偏好函数 $f(\bar{C},\bar{A},\mathfrak{D})$;
输出:满足用户偏好的前 n 个项;
Begin
1. R 接收用户偏好函数 $f(\bar{C},\bar{A},\mathfrak{D})$;
2. R 基于 MDSA 索引树 MDSA-Tree,将 $\bar{C}$ 和 $\mathfrak{D}$ 传输给与 $\bar{C}$ 相关的 r 个第一特征属性抽象层节点 $N_1,\cdots,N_r,r\leqslant\alpha$;
3. for $i=1$ to r do
4. $\quad\mathfrak{D}_i\leftarrow\mathfrak{D}$ 中与 N_i 相关的项推荐条件;
5. $\quad CV\leftarrow\mathfrak{D}_i$ 对应的项编码值;
6. $\quad len(CV)\leftarrow$项编码值 CV 的二进制长度;
7. $\quad len_i\leftarrow$项编码值文件 CF_i 最大编码长度;
8. $\quad CV'\leftarrow CV$;
9. $\quad CV''\leftarrow CV$;
10. $\quad$ for $j=1$ to len_i-$len(CV)$do
11. $\quad\quad CV'\leftarrow CV'\perp''0''$
12. $\quad\quad CV''\leftarrow CV'\perp''1''$
$\quad\quad$/* "$\perp$"为字符串连接符 */
13. $L_i\leftarrow\{CV_i\mid CV_i\in[CV',CV'']\}$;
14. R 接收 $N_1,\cdots,N_r$ 返回的 r 个项编码值列表 $L_1,\cdots,L_r$;
15. for $i=1$ to r do
16. $\quad$ 构造输入键值对集合 $IS_i=\{(F,CV)\mid F$ 是 CV 的标识符 and CV $\in L_i\}$;
17. $\quad$ 将 IS_i 分割成 M 份$\{IS_i^1,\cdots,IS_i^M\}$;
$\quad\quad$/* M 为用户参数 */
18. for $j=1$ to M do /* 并行处理 */
19. $\quad\{(I,P\_flag)\}\leftarrow map(\bigcup_{i=1}^{r}IS_i^j)$;
$\quad\quad$/* I 为项,P_flag 为 I 在偏好属性数据表 PD 上的标志位 */
20. for $t=1$ to V do

/* 并行处理,V 为执行 reduce 函数的工作计算机数量 */
21. $\{(I, P\_V)\} \leftarrow \text{reduce}(\{(I, P\_\text{flag})\})$;
/* I 为项,P_V 为根据用户偏好函数计算出的 I 的聚合值 */
22. 返回 P_V 排在前 n 为的项给用户.
End

TRAMR 算法中的 map 函数(第 19 行)和 reduce 函数(第 21 行)如算法 2 和算法 3 所示.

算法 2 map 函数

输入:(项编码值标识符 F,项编码值 CV)组成的 key-value 输入集;
输出:(项 I,偏好属性数据标志位 P_flag)组成的中间 key-value 集合;
Begin
1. for $i = 1$ to r do
2. 基于项编码值 CV_i 和偏好属性数据表 PD 构造项知识格 $\bar{Z}_i$;
3. 在 $\bar{Z}_i$ 中,对相同的项编码值所对应的列执行位或"∨"操作,得到新的知识格 $\bar{Z}_i$;
4. $a, b \leftarrow \bar{z}_1$ 的行数和列数;
5. $\bar{Z} \leftarrow \left[\begin{matrix} 1 & \cdots & 1 \\ \vdots & \vdots & \vdots \\ 1 & \cdots & 1 \end{matrix}\right]$ (b 列,a 行);
6. for $i = 1$ to r do
7. $\bar{Z} \leftarrow \bar{Z} \wedge \bar{Z}_i$;/* ∧为位与操作 */
8. for $j = 1$ to b do
9. for $y = 1$ to a do
10. if $\bar{Z}[j, y] = 1$ then
11. $I \leftarrow \bar{Z}$ 中列 j 所对应的项;
12. return(I, y);
End

算法 3 reduce 函数

输入:(项 I,偏好属性数据标志位 P_flag)组成的 key-value 输入集;
输出:(项 I,聚合值 P_V)组成的 key-value 集合;
Begin
1. for 输入集中所有相同的项 I do
2. $\text{PL}(I) \leftarrow \{P\_\text{flag}_1, \cdots, P\_\text{flag}_\sigma\}$;
/* $\text{PL}(I)$为项 I 所对应的所有偏好属性数据标志位构成的集合 */
3. $S(I) = \varnothing$;
4. for$\forall P_\text{flag}_i \in \text{PL}(I)$do
5. 基于 P_flag_i 获取用户偏好属性 $\bar{A}$ 上的取值 P_value;
6. $S(I) \leftarrow (I) \cup \{P\_\text{value}\}$;
7. 根据偏好函数计算出 I 的聚合值 P_V;
8. 返回(I, P_V);
End

5 top-n 推荐应用扩展

在实际应用中经常会碰到用户对 top-n 推荐集 NS 有如下要求:NS 中每个项 I 不会在所有 w 个偏好属性 $\bar{A} = (a_1, \cdots, a_w)$上的取值均差于某一个项属于 NS. 这主要是因为,如果项 I 不满足该条件,那么虽然 I 在 NS 中,但是 $T \in$ NS 在任何偏好属性上的取值均优于 I,即 T 完全可以取代 I,即没必要将 I 返回给用户.根据文献[16]的描述,NS 中的每个项必须是 skyline 项,其形式化定义如定义 8 和定义 9 所示.

定义 8(支配关系) 假定 I 和 T 是两个具有 w 个偏好属性的项,其偏好属性为 $\bar{A} = (a_1, \cdots, a_w)$,如果它们满足下列两个条件,那么我们称 T 支配 I:① $\forall a_i \in \bar{A}, I[a_i] \leqslant T[a_i]$,② $\exists a_j \in \bar{A}, I[a_j] < T[a_j]$,其中符号"≤"表示"不优于".

不失一般性,我们把 T 支配 I,记为 $I \prec T$;同时把 T 不支配 I,记为 $I \not\prec T$.

定义 9(skyline 项) 假项集和偏好属性分别为 IS 和 $\bar{A} = (a_1, \cdots, a_w)$,那么项 $I \in$ IS 是 IS 上的 skyline 项,那么它必须满足如下条件:$\neg \exists T \in \text{IS}, I \prec T$.

我们把项集 IS 上所有 skyline 项组成的集合称为 skyline 项集,记为 $\nabla(\text{IS})$.

根据定义 8 和定义 9,我们把 top-n 推荐的这类应用扩展称为 skyline top-n 推荐.为了返回 skyline top-n 推荐集,我们只需要在 reduce 函数(见算法 2)的第 6 行和第 7 行之间增加一条语句来过滤非 skyline 项即可.

我们把项集 IS 上 skyline top-n 推荐集,记为 $\text{TOP}(\nabla(\text{IS}), n)$.基于 skyline top-n 推荐,我们接下来给出两类带偏好约束的 skyline top-n 推荐.

当用户对某个特定偏好区域内的 skyline top-n 推荐项感兴趣时,我们只需返回该区域内推荐项即可.不难看出,根据用户的实际需求,带偏好约束的 skyline top-n 推荐有两种不同的形式,即全局约束 skyline top-n 推荐和局部约束 skyline top-n 推荐,我们把这两种推荐形式返回的项集分别称作全局约束 skyline top-n 推荐项集和局部约束 skyline top-n 推荐项集.

定义 10(全局约束 skyline top-n 推荐项集) 假定 IS 是偏好属性 $\bar{A} = (a_1, \cdots, a_w)$上的项集,用户对 $a_i (1 \leqslant i \leqslant w)$的区域$[B_i, E_i]$感兴趣,即约束条件 $\lambda = \forall i \in [1, w](B_i \leqslant p[a_i] \leqslant E_i)$,那么全局约束 skyline top-n 推荐项集 $\text{GC}(\text{IS}, \lambda, n)$可表示为:$\text{TOP}(\nabla(\text{IS}) \wedge \lambda, n)$.

根据定义 10 可知,$\text{GC}(\text{IS}, \lambda, n)$为满足约束条件 λ 的 skyline 项集上的 top-n 推荐集.

定义 11(局部约束 skyline top-n 推荐项集) 假定 IS 是偏好属性 $\bar{A} = (a_1, \cdots, a_w)$上的项集,用户对 $a_i (1 \leqslant i \leqslant w)$的区域$[B_i, E_i]$感兴趣,即约束条件 $\lambda = \forall i \in [1, w](B_i \leqslant p[a_i] \leqslant E_i)$,那么局部约束 skyline top-n 推荐项集 $\text{LC}(\text{IS}, \lambda, n)$可表示为:$\text{TOP}(\nabla(\text{IS} \wedge \lambda), n)$.

根据定义 11 可知,$\text{LC}(\text{IS}, \lambda, n)$为满足约束条件 λ 的项集 IS 上的 skyline top-n 推荐集.

值得注意的是,我们只需要稍微修改 reduce 函数即

可用于返回 GC(IS,λ,n)和 LC(IS,λ,n).对于 GC(IS,λ,n),我们只需在过滤非 skyline 项的基础上,进一步过滤掉不满足约束条件 λ 的 skyline 项即可;而 LC(IS,λ,n),我们只需在产生 skyline 项之前,过滤掉不满足约束条件 λ 的项即可.不难看出,通常情况下,GC(IS,λ,n)≠(IS,λ,n).然而,定理 3 表明当项集 IS 满足如下条件:∀ I,T∈IS,λ(I)∧I≺T⇒λ(T)时,这两个集合相等.

定理 3(集合相等条件性判定)　假定项集和约束条件分别为 IS 和 λ,那么下列命题成立:

∀ I,T∈IS,λ(I)∧I≺T⇒λ(T)⇔(IS,λ,n) = LC(IS,λ,n).

证明　定理 3 的正确性等价于如下命题的成立:∀ I,T∈IS,λ(I)∧I≺T⇒λ(T)⇔(∇(IS)∧λ) = ∇(IS∧λ).

(1)″⇒″成立:由 I∈∇(IS)∧λ,可得 I∈IS∧λ(I)∧¬ ∃T,(T∈IS∧λ(I)∧I≺T),从而有 I∈IS∧λ(I)∧¬ ∃T,(T∈IS∧λ(I)∧λ(I)∧I≺T),另外,因为λ(I)∧I≺T⇒λ(T)成立,所以有 I∈∇(IS)∧λ = I∈IS∧λ(I)∧¬ ∃T,(T∈IS∧λ(I)∧I≺T∧λ(T)) = ∇(IS∧λ).

(2)″⇐″成立:假设当(∇(IS)∧λ) = ∇(IS∧λ)成立时,∀ I,T∈IS,λ(I)∧I≺T⇒λ(T)不成立.因为λ(I)∧I≺T⇒λ(T)⇔¬ (λ(I)∧I≺T)∨λ(T),所以有∃ I,T,(λ(I)∧I≺T∧¬ λ(T))成立.此时,∃ I,(I∈∇(IS∧λ))⇔∃ I,T,(I∈∇(IS∧λ)∧λ(I)∧I≺T∧¬ λ(T))⇒∃ I,(I∈(IS∧λ)∧λ(I)∧∃ T,(T∈IS∧T∉(IS∧λ)∧I≺T))⇒∃ I,(I∉∇(IS)))⇒∃ I,(I∉(IS∧λ)),所以可得(∇(IS)∧λ)≠∇(IS∧λ),这与假设(∇(IS)∧λ) = ∇(IS∧λ)相矛盾,因此″⇐″成立.

综上可知,定理 3 成立.

6　实验评估

这一节通过具体的实验来评估本文方法的有效性.本文的实验环境由 20 台 PC 机组成三层分布式存储架构,每台 PC 机的配置为四核 i5-3450 CPU、4G 内存和 500G 硬盘,操作系统为 CentOS Linux 6.4.根节点 R 包含 8 台 PC 机组成的集群,其中 1 台 PC 机选为控制计算机(Master),这 8 台 PC 机构成 Hadoop 平台,其版本号为 1.0.3.而其余两层为特征属性抽象层对应 12 个分布式节点,共分配 12 台 PC 机.所有算法的代码编译采用 JDK 1.6.

在实验中,我们使用综合数据集,根节点 R 上的数据表包含 5×10^6 条记录和 10 个偏好属性,其类型均为 32 位 float 浮点型.第一特征属性抽象层的 3 个节点分别存储具有 5×10^5 条记录和 5 个特征属性的数据表,其类型均为字符串类型.第二特征属性抽象层的 9 个节点分别存储具有 1×10^6 条记录和 10 个特征属性的数据表,其类型均为字符串类型.

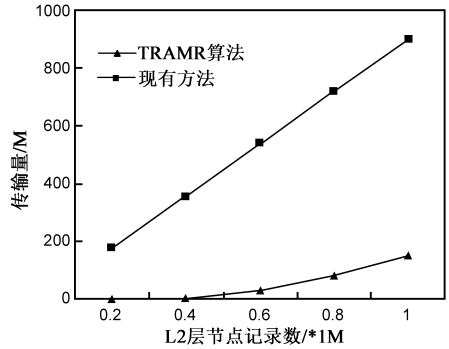
6.1　评估节点间的数据传输量

在第一组实验中,我们评估本文 TRAMR 算法在 MDSA 分布式架构节点间的数据传输量.现有方法[10~14]在处理 top-n 推荐时,没有考虑分布式节点间的数据传输代价,所采的传输方式基本一致,即直接对索引所指向的原始数据进行传输.在实验中,我们固定特征属性的个数,来考察数据传输量随特征属性数据表记录数的变化情况.图 2(a)给出了第二特征属性抽象层(L2)到第一特征属性抽象层(L1)的数据传输量,而图 2(b)给出了第一特征属性抽象层到根节点(R)的数据传输量.

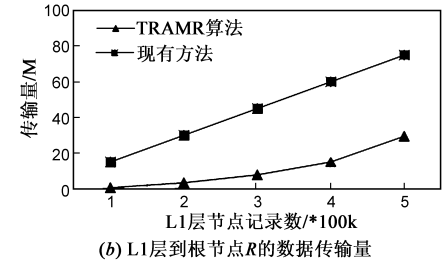
从图 2 可以看出,TRAMR 算法数据传输量比现有方法的数据传输量要小得多,这主要是因为 TRAMR 算法采取了有效的项成员编码和索引路径编码,从而不需要传输原始数据集,只需要将它们所对应的编码值传输给上一层节点即可.例如在图 2(a)中,当第二特征属性抽象层节点的记录数为 1×10^6 时,现有方法的传输量为 900M,而本文 TRAMR 算法的传输量仅为 152.8M.另外,在图 2(b)中,当第一特征属性抽象层节点的记录数为 5×10^5 时,现有方法的传输量为 75M,而本文 TRAMR 算法的传输量仅为 29.6M.

6.2　评估产生 top-n 推荐项的时间

在第二组实验中,我们评估本文 TRAMR 算法产生 top-n 推荐项的时间.与 TRAMR 算法比较的有目前常用的 RT-CAN 算法[11]、B+_topN 算法[12]和 KFSSI 算法[14].在实验中,不失一般性,我们固定 n 为 10,图 3 给出 top-n 推荐时间随记录数变化的实验结果.



(a) L2层到L1层的数据传输量



(b) L1层到根节点R的数据传输量

图2　算法节点间的数据传输量评估

从图 3 可以看出,TRAMR 算法产生 top-10 推荐的时间代价比 RT-CAN、B + _topN 和 KFSSI 这三个现有算法要低,这主要因为 TRAMR 算法利用项知识格上的位操作来缩减定位偏好属性数据表记录的时间.例如在图 3 中,当根节点记录数为 5×10^6 时,TRAMR 算法的时间开销为 34.6s,而 RT-CAN、B + _topN 和 KFSSI 这三个现有算法的时间开销分别为 55.2s、65.5s 和 47.9s.

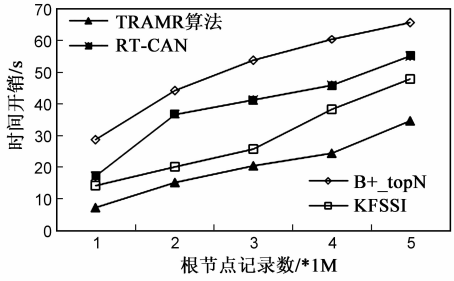


图3 top-n推荐时间随根节点记录数变化实验

6.3 评估 top-n 推荐应用扩展时间

在第三组实验中,我们评估本文给出的 top-n 推荐应用扩展时间开销,包括 skyline top-n 推荐、全局约束 skyline top-n 推荐和局部约束 skyline top-n 推荐三类.与第二组实验类似,不失一般性,我们固定 n 为 10,图 4 给出这三类应用扩展的时间随记录数变化的实验结果.为了便于对照,在实验中,我们也给出了 top-n 推荐的时间代价.

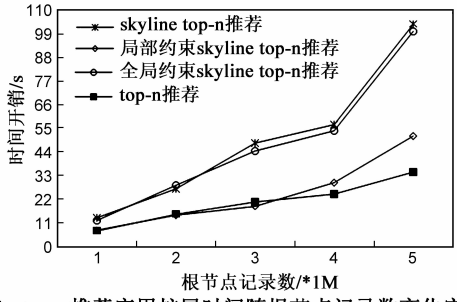


图4 top-n推荐应用扩展时间随根节点记录数变化实验

从图 4 可以看出,skyline top-10 推荐和全局约束 skyline top-10 推荐的时间开销差不多,这主要是因为这两类应用扩展需要对相同数据量的记录进行 skyline 操作.例如,在图 4 中,当根节点记录数为 5×10^6 时,skyline top-10 推荐和全局约束 skyline top-10 推荐的时间开销分别为 103.8s 和 100.4s.另外,由于 skyline 操作是 CPU 和 I/O 敏感的,因此 skyline top-10 推荐和全局约束 skyline top-10 推荐比 top-10 推荐需要更多的时间,例如,当根节点记录数为 5×10^6 时,top-10 推荐只需要 34.6s 的时间开销.此外,由于局部约束 skyline top-n 推荐在执行 skyline 操作之前先过滤掉不满足约束条件的项,因此这类应用扩展比 skyline top-10 推荐和全局约束 skyline top-10 推荐需要较少的 skyline 操作时间,例如,当根节点记录数为 5×10^6 时,局部约束 skyline top-n 推荐只需要 51.5s 的时间开销.

7 结论

目前,大都数电子商务企业的业务是基于云架构的,因此,研究云环境下的 top-n 推荐问题是一个很有意义的工作.本文在分析现有工作不足的基础上,设计了一种新颖的多层分布式存储架构 MDSA,在该架构中,网络节点被组织成一棵层次索引树 MDSA-Tree,而电子商务数据按照特定的规则分割存储于 MDSA-Tree 的各节点上.为了降低网络传输代价和 I/O 开销,本文提出了适合 top-n 推荐的数据编码模式,而为了缩减系统响应时间,本文利用目前成熟的 map/reduce 分布式编程模型来快速获取满足用户偏好的前 n 个项.此外,为了满足实际的需求,本文给出了三种 top-n 推荐的应用扩展:skyline top-n 推荐、全局约束 skyline top-n 推荐和局部约束 skyline top-n 推荐.最后,本文实施大量的实验评估,并从多个不同的角度来综合考察所提算法的实用性和可扩展性.

本文将来的后续工作主要包括:研究新的多层分布式存储架构和数据编码模式来进一步降低网络传输代价和 I/O 开销;根据实际的需要,设计其他形式的 top-n 推荐的应用扩展.

参考文献

[1] Y Ren, T Zhu, G Li. Top-n tecommendations by learning user preference dynamics[A]. Proceedings of PAKDD'13[C]. Gold Coast: Springer Verlag, 2013. 390 – 401.

[2] B Sarwar, G Karypis, J Konstan, J Riedl. Analysis of recommendation algorithms for e-commerce[A]. Proceedings of EC'00[C]. Minneapolis: ACM Press, 2000. 158 – 167.

[3] M Deshpande, G Karypis. Item-based top-N recommendation algorithms[J]. ACM Transactions on Information Systems, 2004, 22(1): 143 – 177.

[4] M Jamali, M Ester. Using a trust network to improve top-N recommendation[A]. Proceedings of RecSys'09[C]. New York: ACM Press, 2009. 181 – 188.

[5] HN Kim, AT Ji, HJ Kim, GS Jo. Error-based collaborative filtering algorithm for top-n recommendation[A]. Proceedings of APWeb'07[C]. Huang Shan: Springer Verlag, 2007. 594 – 605.

[6] N Hurley, M Zhang. Novelty and diversity in top-n recommendation-analysis and evaluation[J]. ACM Transactions on Internet Technology, 2011, 10(4): 161 – 192.

[7] J Ha, SH Kwon, SW Kim, C Faloutsos, S Park. Top-n recommendation through belief propagation [A]. Proceedings of

CIKM'12[C]. Maui: ACM Press, 2012. 2343 – 2346.

[8] 李聪,梁昌勇.基于 n 序访问解析逻辑的协同过滤冷启动消除方法[J].系统工程理论与实践,2012,32(7):1537 – 1545.
C Li, C Liang. Cold-start eliminating method of collaborative filtering based on n-sequence access analytic logic[J]. Systems Engineering-Theory & Practice, 2012, 32(7): 1537 – 1545. (in Chinese)

[9] Y Gong, Z Ying, M Lin. A survey of cloud computing[A]. Proceedings of GCN'12[C]. Gandia: Springer Verlag, 2012. 79 – 84.

[10] X Zhang, J Ai, Z Wang, J Lu, X Meng. An efficient multi-dimensional index for cloud data management[A]. Proceedings of CloudDB'09[C]. HongKong: ACM Press, 2009. 17 – 24.

[11] J Wang, S Wu, H Gao, J Li, B Ooi. Indexing multi-dimensional data in a cloud system[A]. Proceedings of SIGMOD'10[C]. Indianapolis: ACM Press, 2010. 591 – 602.

[12] S Wu, D Jiang, B Ooi, K Wu. Efficient B-tree based indexing for cloud data processing[A]. Proceedings of VLDB'10[C]. Singapore: VLDB Endowment, 2010. 1207 – 1218.

[13] Y Ma, J Rao, W Hu, X Meng, X Han, Y Zhang, Y Chai, C Liu. An efficient index for massive IOT data in cloud environment[A]. Proceedings of CIKM'12[C]. Maui: ACM Press, 2012. 2129 – 2133.

[14] Y Hsu, Y Pan, L Wei, W Peng, W Lee. Key formulation schemes for spatial index in cloud data managements[A]. Proceedings of MDM'12[C]. Bengaluru: IEEE Press, 2012. 21 – 26.

[15] A McClean, R Conceicao, M Halloran. A comparison of mapreduce and parallel database management systems[A]. Proceedings of ICONS'13[C]. Seville: IARIA Press, 2013. 64 – 68.

[16] S Borzsonyi, D Kossmann, K Stocker. The skyline operator[A]. Proceedings of IEEE ICDE'01[C]. Heidelberg: IEEE Press, 2001. 421 – 430.

作者简介

黄震华　男,1980 年生,博士,副教授,软件行业协会系统工程分会理事,CCF 会员.主要研究方向为云计算、信息推荐、数据挖掘与知识发现等.

E-mail: shtj08.zhh@gmail.com