采用协同过滤技术进行工作流活动推荐

陈广智,何 文,李 磊
(中山大学数据科学与计算机学院,广东广州 510006)

摘 要: 为解决企事业单位的流程变动问题,利用正常实例和异常实例信息向当前不完整实例推荐下一可能执行的活动.由于每个工作流实例是一个活动名称序列,它们不能直接参与数值运算,需首先将序列中每个活动出现的顺序以数值的形式表示出来,最终将实例库转换成矩阵形式,该矩阵类似于推荐系统中的User-Item矩阵,以便于实例间相似度计算.最后,从实例库中筛选出与当前不完整实例相似性高的完整实例,利用这些实例的信息构造出活动列表,作为推荐结果.实验结果及对比分析表明:我们的活动推荐算法是可行的和有效的.

关键词: 协同过滤;活动推荐;推荐系统;工作流活动;工作流实例
中图分类号: TP311 **文献标识码:** A **文章编号:** 0372-2112 (2017)04-0890-08
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2017.04.018

Workflow Activity Recommendation by Collaborative Filtering

CHEN Guang-zhi, HE Wen, LI Lei
(*School of Data and Computer Science, Sun Yat-sen University, Guangzhou, Guangdong* 510006, *China*)

Abstract: To address the problem of changes of business processes for an enterprise or organization, we utilize the normal and exceptional instances to recommend the next possible activity for the current incomplete workflow instance. Since every workflow instance is a sequence of activity names, it cannot be calculated numerically. we firstly extract the order of each activity in the sequence as a number value, and then get a matrix which is similar to User-Item matrix in traditional recommendation systems. This matrix can facilitate the calculation of similarity between two workflow instances. Finally, we choose these complete instances which are most similar to the current incomplete instance, construct the activity list as the recommendation result by these instances. Experimental results show that the proposed algorithm is effective and efficient.

Key words: collaborative filtering; activity recommendation; recommendation system; workflow activity; workflow instance

1 引言

工作流技术[1]已经在一些领域得到了应用,且应用领域越来越广泛,例如,汽车制造[2]、电子商务[3]、医疗[4]、电子政务[5,6]等.随着市场经济的快速发展及大数据、云计算等技术的普及,企业或组织的流程变动日益成为突出问题.采用智能化技术适应流程变动成为对工作流技术的一个必然要求[7].流程变动分为两种,一是模型的变动,另一个是实例的变动.针对模型变动,一些方法[8~11]利用先前保存的模型库对当前建模的模型进行建模任务推荐,从而重用先前建模信息,以加速建模过程;而实例变动研究较少,因而是本文的主要研究内容.

传统工作流管理系统假定工作流模型事先完全确定,当业务流程不发生变化的情况下它是可行的.事实上,掌控全部流程信息是困难的.信息管理系统则没有包含任何流程信息,不能从流程管理方面支持企业或组织高效率运转.上述两种系统的过渡地带就存在着这样的系统,其需要流程信息,但这些流程信息不太明确又经常变动;或者说,纵然预先定义的模型已经覆盖了系统大部分实例,但受各种情况的影响总会出现特殊实例.这类系统如采用传统方法管理流程,就会陷入"流程变动"的泥潭里.为了解决这类系统面临的流程变动问题,本文综合利用系统积累下来的正常实例和

收稿日期:2015-11-26;修回日期:2016-03-28;责任编辑:孙瑶
基金项目:国家自然科学基金(No. 61300095);广东省自然科学基金(No. S2012040011123);广东省教育厅高校优秀青年创新人才培育(No. 2012LYM_0065)

特殊实例信息,向当前不完整实例推荐下一个可能活动,而非强制规定.这既利用了模型信息,又融合了特殊实例信息,避免了工作流模型再设计.

当前,推荐技术在音乐[12]、电影[13]、图书[14]、购物[15,16]等方面已经得到了应用,自然地也可将其用于工作流活动推荐.我们将当前正在执行的工作流实例称为不完整实例,将已经执行结束的实例称为完整实例.为将协同过滤技术中的相似度量用于实例,本文先讨论如何将实例转化成数值形式;然后,在上述数值形式的基础上给出不完整实例和完整实例的相似度计算方法;最后,设计了两个工作流活动推荐算法 flowRec 和 flowRecK.实验结果及对比分析表明,本文提出的推荐算法是可行的和有效的.

2　相关工作

Agnes Koschmider 等人[9]利用元数据信息筛选出与当前模型块相似的业务模型或业务模型块,供用户选用.该方法推荐粒度较大,是模型级别上的推荐,以辅助模型建模.文献[8]讨论了如何更好地利用历史服务组合例子来帮助构建当前服务组合,采用离线创建的模式表向当前不完整服务组合推荐下一服务.本文虽然也用历史信息推荐,但却是实例级别上的推荐;更为不同的是本文借鉴了协同过滤技术[17~21],应用场景可以是电子政务中某项审批业务的活动推荐.文献[10,11]先用最小深度优先编码将图形表示的模型转化成字符串,接着用字符串匹配向不完整模型推荐下一任务;而本文是将活动序列构成的字符串转化成数值的集合,再用协同过滤方法推荐.

文献[22]也采用历史实例信息向不完整实例推荐,它采用三种不同的方法筛选与不完整实例相似的完整实例:前缀法、集合法和多重集合法.前缀法要求太苛刻且没有考虑当前活动之前的活动;后两种方法则完全忽略了活动的顺序信息.为克服上述方法的不足,本文方法不仅考虑了不完整实例的最后一个活动,而且也考虑了该活动之前的活动及它们的顺序信息.

3　问题描述

假定对某业务流程来说,所有可能执行活动的集合为 $\mathcal{A}=\{a_1,a_2\cdots,a_n\}$,一个工作流实例定义为有顺序的活动序列,记为 $e_i\equiv<a_{i_1},a_{i_2},\cdots,a_{i_k}>$,其中 $a_{i_j}\in\mathcal{A}$,$j=1,2,\cdots,k$,同一活动允许在实例中多次出现.进一步假定:所有可能执行的活动是确定的,即集合 $\mathcal{A}$ 是确定的,而工作流实例可能违反该业务模型.对一个完整实例来说,它的首活动和尾活动分别称为开始活动和结束活动.假定 E 是由 m 个完整工作流实例构成的实例库,记为 $E=\{e_1,e_2,\cdots,e_m\}$,它是多重集合,即某个完整实例可以在 E 中出现多次;$\hat{e}=<a_{i_1},a_{i_2},\cdots,a_{i_\delta}>$ 为一不完整实例,其中活动 a_{i_δ} 不满足模型结束活动定义,δ 是 $\hat{e}$ 的长度.我们的问题定义为:给定 E 和 $\hat{e}$,如何向 $\hat{e}$ 推荐其下一个可能执行的活动 $a_{i_{\delta+1}}$?

4　推荐方法

4.1　数据格式的转化

为利用协同过滤技术中数值形式的相似度计算方法,必须首先将 E 中完整实例信息转化成数值形式,本文将其转化成矩阵形式.矩阵行对应 E 中一个完整实例,矩阵列对应集合 $\mathcal{A}$ 中的一个活动,矩阵元素是某个完整实例中某个活动出现的顺序值.由于工作流模型可能包含循环分支,因此 E 中那些完整实例对应的活动序列中某些活动可能重复出现,进而导致矩阵元素不是一个数值,而是多个数值的集合.下面详细讨论如何将 E 转化成矩阵形式 $\boldsymbol{M}$.

首先扫描 E,取出所有可能出现的活动名称;对这些名称排序,把它们作为 $\boldsymbol{M}$ 的列属性,具体如何排序无所谓,不影响推荐.于是确定了矩阵的列和列数.然后,依次由 E 中每个完整实例确定 $\boldsymbol{M}$ 行元素,将实例中各个活动出现的先后顺序值作为该行对应活动列的元素值;如果某活动出现多次,则将该活动所有出现的顺序值都作为 $\boldsymbol{M}$ 中对应元素的值.一个数据格式转化的例子参见图 1.图中集合 $\mathcal{A}=\{a_1,a_2,\cdots,a_7\}$,并按顺序 $a_1,a_2,\cdots,a_7$ 作为 $\boldsymbol{M}$ 的列属性,活动 a_1、a_7 分别是开始、结束活动.由于 a_2 在第一个完整实例中分别以第 2、6 顺序出现,因此矩阵第一行第二列元素含两个数值 2 和 6.

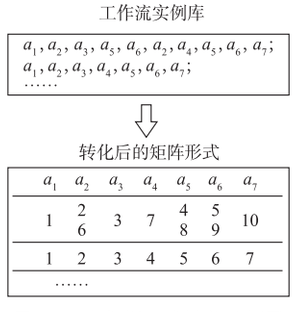

图1　一个数据格式转化的例子

按上述步骤处理后,得到 $\boldsymbol{M}$.严格说,$\boldsymbol{M}$ 不是通常意义下的矩阵,因其元素可能含多个数值,但为了方便,本文仍称之为矩阵.算法 1 描述了数据格式转化算法 dataTransfer(E).

算法 1　数据格式转化算法 dataTransfer(E)

Input:工作流实例库 E.

Output:矩阵 $\boldsymbol{M}$.

1 扫描 E,确定 $\mathcal{A}$;
2 对 $\mathcal{A}$ 中活动排序,得序列 $A=<a_1,a_2,\cdots,a_n>$;
3 for $i\leftarrow 1$ to $|E|$ do
4 for $j\leftarrow 1$ to n do
5 if 活动 a_j 在 E 的第 i 个实例中没有出现 then
6 $M_{i,j}\leftarrow 0$;
7 else
8 将活动 a_j 在第 i 个实例中出现的所有顺序值按照从小到大的顺序赋给元素 $M_{i,j}$;
9 Return $\boldsymbol{M}$;

算法 1 中,$M_{i,j}$表示 $\boldsymbol{M}$ 的第 i 行第 j 列元素,$|E|$ 表示 E 包含的实例数量. 如果 $E=\{e\}$,那么算法 dataTransfer($\{e\}$)返回的 $\boldsymbol{M}$ 就只有一行. 此情况下为方便,我们将 dataTransfer($\{e\}$)简写为 dataTransfer(e). 假定 $|E|=m$,$|\mathcal{A}|=n$,那么 $\boldsymbol{M}$ 的形式为:

$$\boldsymbol{M}=\begin{bmatrix} M_{1,1} & M_{1,2} & \cdots & M_{1,n} \\ M_{2,1} & M_{2,2} & \cdots & M_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ M_{m,1} & M_{m,2} & \cdots & M_{m,n} \end{bmatrix} \tag{1}$$

其中 $M_{i,j}$ 可能为一非负整数,也可能为非负整数的集合.

4.2 推荐算法

根据本文场景及将要在 5.1 节讨论的工作流模型,可知由 dataTransfer(E)产生的 $\boldsymbol{M}$ 不是稀疏的. 因为作为工作流活动推荐,所有正常实例都遵从同一模型,它们活动序列中包含的活动差别不会太大;另外按 5.1 节方法生成的异常实例,出现的数量较少,且包含的活动来自集合 $\mathcal{A}$,最终对 $\boldsymbol{M}$ 是否稀疏影响也不大. 于是当 $\boldsymbol{M}$ 不稀疏时,我们可以用 Pearson 相关系数计算两个实例 e_1 和 e_2 的相似度,而不采用稀疏矩阵时 Cosine 公式计算[23]. 因 $\boldsymbol{M}$ 中元素 $M_{i,j}$可能含多个数值,不能直接使用 Pearson 公式,为此需对 $\boldsymbol{M}$ 中与实例 e_1、e_2 对应的两行数据扁平化,即让行中各元素都含一个数值. 下面详细讨论扁平化. 注意,扁平化不针对 $\boldsymbol{M}$ 中某一行,只有当需计算相似度的两行都确定时,才能扁平化.

令 e_1 和 e_2 为两个工作流实例,A_1 和 A_2 分别为它们各自包含的活动名称集合,于是 $A_1\cap A_2$ 就是 e_1 和 e_2 共同出现的活动名称集合;用 $\text{dataTransfer}(e_1)|_{A_1\cap A_2}$表示从 e_1 转化后的那行数据中取出 $A_1\cap A_2$ 属性列对应的数值所构成的行数据,同理 $\text{dataTransfer}(e_2)|_{A_1\cap A_2}$. 上述转化后的行数据中的元素仍可能含多个数值,因此需扁平化. 假定:

$$\begin{aligned} &\text{dataTransfer}(e_1)|_{A_1\cap A_2}=(X_1,X_2,\cdots,X_i,\cdots X_k) \\ &\text{dataTransfer}(e_2)|_{A_1\cap A_2}=(Y_1,Y_2,\cdots,Y_i,\cdots Y_k) \end{aligned} \tag{2}$$

其中 X_i 和 Y_i 分别是 e_1 和 e_2 转化后的行数据中同一活动列对应的元素,$i=1,2,\cdots,k$.

我们分情况讨论如何扁平化(2)中数据:

(1)如果 X_i 仅含一个数值$\{x_{i1}\}$,Y_i 也仅含一个数值$\{y_{i1}\}$,那么取 $X_i\equiv x_{i1}$,$Y_i\equiv y_{i1}$.

(2)如果 X_i 含多个值$\{x_{i1},x_{i2},\cdots,x_{ip}\}$,$Y_i$ 仅含一个值$\{y_{i1}\}$,则从 X_i 中找出与 y_{i1}最接近的那个值,作为 X_i 扁平化后的值,Y_i 不需扁平化,即:$j^*=\arg\min\limits_{j=1,2,\cdots,p}|x_{i,j}-y_{i1}|$;取 $X_i\equiv x_{ij^*}$,$Y_i\equiv y_{i1}$.

(3)如果 Y_i 含多个值$\{y_{i1},y_{i2},\cdots,y_{iq}\}$,$X_i$ 仅含一个值$\{x_{i1}\}$,则按上述情况类似处理.

(4)如果 X_i 含多个值$\{x_{i1},x_{i2},\cdots,x_{ip}\}$,$Y_i$ 也含多个值$\{y_{i1},y_{i2},\cdots,y_{iq}\}$,则找出下标 $t\leqslant\min(p,q)$,使得 $\forall 1\leqslant j<t$,有 $x_{i,j}=y_{i,j}$,但 $x_{it}\neq y_{it}$,此时取 $X_i\equiv x_{it}$,$Y_i\equiv y_{it}$;如果不存在上述的 t,但 $\forall 1\leqslant j\leqslant\min(p,q)$,有 $x_{i,j}=y_{i,j}$,则取 $X_i\equiv x_{i\min(p,q)}$,$Y_i\equiv y_{i\min(p,q)}$.

对未扁平化的数据$(X_1,X_2,\cdots,X_i,\cdots,X_k)$,用符号 $\text{flat}((X_1,X_2,\cdots,X_i,\cdots,X_k))$表示扁平化的结果. 扁平化为用 Pearson 相关系数计算 e_1 和 e_2 的相似度创造了条件,二者的相似度按式(3)计算:

$$\text{sim}(e_1,e_2)=\text{Pearson}\begin{pmatrix}\text{flat}(\text{dataTransfer}(e_1)|_{A_1\cap A_2}),\\ \text{flat}(\text{dataTransfer}(e_2)|_{A_1\cap A_2})\end{pmatrix} \tag{3}$$

若用 $\boldsymbol{x}=(x_1^f,x_2^f,\cdots,x_k^f)$ 和 $\boldsymbol{y}=(y_1^f,y_2^f,\cdots,y_k^f)$ 表示两个扁平化数据,那么式(3)中 Pearson 函数如下:

$$\text{Pearson}(\boldsymbol{x},\boldsymbol{y})=\frac{\sum\limits_{i=1}^{k}(x_i^f-\bar{x}^f)(y_i^f-\bar{y}^f)}{\sqrt{\sum\limits_{i=1}^{k}(x_i^f-\bar{x}^f)^2}\cdot\sqrt{\sum\limits_{i=1}^{k}(y_i^f-\bar{y}^f)^2}} \tag{4}$$

例如,对图 1 中两个实例对应的矩阵行扁平化后分别得到(1,2,3,7,4,5,10)和(1,2,3,4,5,6,7),于是按式(3)二者的相似度为 0.8458.

在上述基础上,讨论如何向一不完整实例 $\hat{e}=<a_{i_1},a_{i_2},\cdots,a_{i_s}>$ 推荐下一可能活动. 借鉴 kNN 思想[24],首先利用式(3)从 E 中找出与 $\hat{e}$ 最相似的 k 个完整实例,记为 kNN($\hat{e}$);然后,从这些实例中找出相对于 $\hat{e}$ 的下一活动. 寻找方法为:对某完整实例 $e_i\in\text{kNN}(\hat{e})$,如果其长度大于 δ,则从其活动序列中取出排序在 $\delta+1$ 的那个活动,此活动是 $\hat{e}$ 的下一可能执行活动,记为 $na(e_i,\hat{e})$;对所有 $e_i\in\text{kNN}(\hat{e})$,都找出 $na(e_i,\hat{e})$,得到 $\hat{e}$ 的下一可能活动的集合,记为 $na(\text{kNN}(\hat{e}),\hat{e})$. 从该集合中找出出现频率最大的那个活动,将其推荐给 $\hat{e}$. 对 $a_i\in na(\text{kNN}(\hat{e}),\hat{e})$,如果用 $\text{Freq}(a_i)$ 表示它的频率,那么用 E 向 $\hat{e}$ 推荐的下一可能活动 $P_E(\hat{e})$ 为:

$$P_E(\hat{e})=\arg\max_{a_i\in na(\text{kNN}(\hat{e}),\hat{e})}\text{Freq}(a_i) \tag{5}$$

算法 2 描述了上述推荐过程. 它首先对 E 和 $\hat{e}$ 进行

数据格式转换;然后分别对 $\hat{m}$ 与 $M_i(i=1,2,\cdots,m)$ 做如下运算:取出它们公共活动列对应的元素值,将这些值按原顺序作为新的行数据,再扁平化,然后计算它们的 Pearson 相关系数 p;最后将 p、E 中与 M_i 对应的 e_i 一起作为二元组添加到 S 中. 将 p 和 e_i 一起添加到 S 中,可方便计算集合 $na(\mathrm{kNN}(\hat{e}),\hat{e})$. 如果 m 和 n 分别为 $\boldsymbol{M}$ 的行和列数量,且 $\boldsymbol{M}$ 中每个元素至多含 k 个数值,那么算法 2 在最坏情况下的时间复杂度为 $O(m\cdot n\cdot k)$.

算法 2　推荐算法 flowRec($E,\hat{e}$)

Input:实例库 E,不完整实例 $\hat{e}$.
Output:向 $\hat{e}$ 推荐的活动 $P_E(\hat{e})$.
1 $\boldsymbol{M}=\mathrm{dataTransfer}(E)$;
2 $\hat{m}=\mathrm{dataTransfer}(\hat{e})$;
3 令 $S=\{\}$;
4　for $i\leftarrow 1$ to m do
5　　取出 $\boldsymbol{M}$ 中的第 i 行数据 M_i;
6　　计算 $M_i\mid_{A_i\cap\hat{A}}$ 和 $\hat{m}\mid_{A_i\cap\hat{A}}$;
7　　计算 $M_i^f=\mathrm{flat}(M_i\mid_{A_i\cap\hat{A}})$ 和 $\hat{m}^f=\mathrm{flat}(\hat{m}\mid_{A_i\cap\hat{A}})$;
8　　计算 $p=\mathrm{Pearson}(M_i^f,\hat{m}^f)$;
9　　将 (p,e_i) 添加到 S;
10 利用 S 确定集合 kNN $(\hat{e})$;
11 计算 $na(\mathrm{kNN}(\hat{e}),\hat{e})$;
12 计算 $P_E(\hat{e})=\underset{a_i\in na(\mathrm{kNN}(\hat{e}),\hat{e})}{\arg\max}\mathrm{Freq}(a_i)$;
13　Return $P_E(\hat{e})$;

算法 2 的推荐结果只有一个,可是,推荐系统往往会推荐多个结果供用户选择. 类似地,可将算法 2 的第 12 行去掉,使它的返回值为 $na(\mathrm{kNN}(\hat{e}),\hat{e})$,亦即使算法返回 k 个活动. 这个改动后的算法记为 flowRecK($E,\hat{e}$).

5　实验结果及分析

5.1　实验数据

我们编写了一个 C++ 程序 DataGen,用于产生实验所需的仿真数据集:实例库 E. 产生过程如下:首先手工设计一个足够复杂的工作流模型(参见图 2);然后按该模型随机生成 $m-r$ 个正常实例,其中 m 是 E 中完整实例的数量,r 是 E 中异常实例的数量;最后,随机产生 r 个异常实例,作为图 2 工作流模型的“流程违反”,并将这 r 个实例随机插入到上述 $m-r$ 个正常实例中. 最终生成的 E 含 m 个完整实例. 我们分别生成了 $m=3000$、5000、7000、9000、11000、13000 的数据集,其中都设置 $r=m\times 5\%$,分别记为 *DS*3、*DS*5、*DS*7、*DS*9、*DS*11、*DS*13.

图 2 模型用文献[25,26]图表示方式,作为 DataGen 的输入;DataGen 根据该模型活动走向,随机确定下一走向,直到到达结束活动 a18,这就构造了一个完整实例.

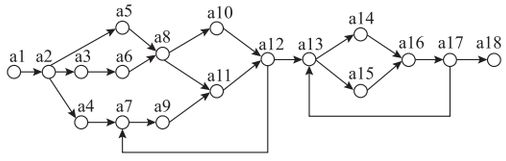

图2　一个用于生成工作流实例信息的工作流模型

5.2　实验方法及结果分析

为验证算法 flowRec 和 flowRecK 的有效性并评价二者的推荐效果,我们用 Python 语言实现了它们. 所有实验均在内存为 2G、Intel 双核主频为 2.8G 的 CPU、操作系统为 Ubuntu Kylin 14.04 的电脑上运行.

现给出训练数据和测试数据的划分及算法效果的评价. 从实例库 E 中随机选 $m/3$ 个实例作为测试数据 E_{Test},余下作为训练数据. 因 E_{Test} 中都是完整实例,它们不需推荐,所以我们需由 E_{Test} 构造不完整实例. 令 δ 表示一个不完整实例的活动序列长度,为避免推荐的冷启动问题(Cold-Start Problem)[27,28],本文规定 $\delta>3$. 由 E_{Test} 构造长度为 δ 的不完整实例的方法为:依次处理 E_{Test} 中每个实例,如果其长度大于 δ,则从它的开始活动开始按顺序截取长度为 δ 的子序列,将该子序列作为一个不完整实例添加到集合 $\hat{E}^{\delta}_{\mathrm{Test}}$. 图 3 是一个由完整实例构造不完整实例的例子,其中完整实例取自图 1,构造的两个不完整实例的长度分别为 6 和 4. 为更全面地评测算法,分别生成了 δ 为 4,5,6,…,17 的不完整实例测试集 $\hat{E}^{\delta}_{\mathrm{Test}}$.

算法评价指标采用推荐系统中的 hit ratio[29,30]. 具体到本文场景,令 $\hat{e}^{\delta}$ 为一长度为 δ 的不完整实例,e 是生成 $\hat{e}^{\delta}$ 的完整实例,根据上述构造方法,可以通过 e 确定 $\hat{e}^{\delta}$ 的真正下一活动,记其为 $tna(\hat{e}^{\delta})$. 对 $\hat{e}^{\delta}$ 来说,如果 $P_E(\hat{e}^{\delta})=tna(\hat{e}^{\delta})$,则称 $P_E(\hat{e}^{\delta})$ 为向 $\hat{e}^{\delta}$ 推荐的一次 hit. 于是评价算法 flowRec 的 hit ratio 为:

$$hr_{\mathrm{flowRec}}(k,\delta)=\frac{\left|\{\hat{e}^{\delta}\in\hat{E}^{\delta}_{\mathrm{Test}}\mid P_E(\hat{e}^{\delta})=tna(\hat{e}^{\delta})\}\right|}{\left|\hat{E}^{\delta}_{\mathrm{Test}}\right|}\quad(6)$$

由于算法 flowRecK 推荐的是一个活动集合,因此其 hit ratio 的计算方法与式(6)稍微不同. 如果 flowRecK 返回的集合 $na(\mathrm{kNN}(\hat{e}^{\delta}),\hat{e}^{\delta})$ 中含 $\hat{e}^{\delta}$ 真正要执行的下一活动 $tna(\hat{e}^{\delta})$,那么就称该集合为向 $\hat{e}^{\delta}$ 推荐的一次 hit,于是评价算法 flowRecK 的 hit ratio 为:

$$hr_{\mathrm{flowRecK}}(k,\delta)=\frac{\left|\{\hat{e}^{\delta}\in\hat{E}^{\delta}_{\mathrm{Test}}\mid tna(\hat{e}^{\delta})\in na(\mathrm{kNN}(\hat{e}^{\delta}),\hat{e}^{\delta})\}\right|}{\left|\hat{E}^{\delta}_{\mathrm{Test}}\right|}\quad(7)$$

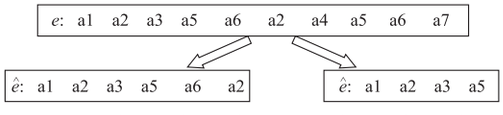

图3　由一个完整工作流实例 e 生成不完整工作流实例举例

为评价算法 flowRec 和 flowRecK 在不同数据集下的推荐效果，固定 $k=20$，比较了二者在数据集 *DS*3、*DS*5、*DS*7、*DS*9、*DS*11、*DS*13 上推荐的 hit ratio，如图 4. 从图中可看出，当 $\delta<11$，二者的推荐效果对数据集不敏感，而当 $\delta\geqslant 11$，它们 hit ratio 才波动. 当 $\delta>11$，flowRec 的推荐效果随着数据规模的增大而改善；而 flowRecK 则相反，其推荐效果随着数据规模的增大而变差. 这是因为 flowRecK 受 k 的影响较大，而此时 k 固定，从而导致数据规模大时，式(7)分母变大，分子由于 k 无变化而保持相对稳定，推荐效果变差. 由于二者推荐效果受数据规模变化不同方向的影响，我们进行折衷，在后面实验中采用数据集 *DS*7.

再来看算法 flowRec 和 flowRecK 在不同 k 值下的 hit ratio 对比，如图 5. 与前述类似，当 $\delta<11$，二者的 hit ratio 不随 k 变化；当 $\delta\geqslant 11$，它们的 hit ratio 才随 k 变化. 变化趋势仍是反向的：flowRec 的 hit ratio 随 k 增加而降低，而 flowRecK 的随 k 增加而增加. 因 flowRec 用集合 $na(\mathrm{kNN}(\hat{e}^{\delta}),\hat{e}^{\delta})$ 中出现频率最大的活动推荐，当 k 增加时，该集合的非真正要执行的下一活动的频率就可能超过真正执行的下一活动的频率，进而引起式(6)分子变小，所以 flowRec 的 hit ratio 随 k 增加而降低. 算法 flowRecK 的 hit ratio 的变化较容易理解，k 增大，$na(\mathrm{kNN}(\hat{e}^{\delta}),\hat{e}^{\delta})$ 会包含更多的真正要执行的下一活动，进而促进 hit 增加.

接着比较算法 flowRec 和 flowRecK 的运行时间，同样固定 $k=20$，令 δ 遍取集合 $\{4,5,\cdots,17\}$ 中的值. 图 6 给出了二者在不同数据集上所耗时间对比. 曲线上点的纵坐标表示：flowRec 采用横坐标上某数据集、运行完测试数据分别为 $\hat{E}^{\delta}_{\mathrm{Test}}(\delta=4,5,\cdots,17)$ 所耗时间减去 flowRecK 在同情况下所耗时间得到的差. 从图中可看出，flowRec 仅在 *DS*7、*DS*13 上用的时间比 flowRecK 少，其余都比 flowRecK 多. 由此表明大部分情况下 flowRec 所耗时间多于 flowRecK，这是因为 flowRec 多了一步计算 $na(\mathrm{kNN}(\hat{e}^{\delta}),\hat{e}^{\delta})$ 中活动频率的操作.

最后，对比算法 flowRec、flowRecK 与文献[8]算法 FlowRecommender 的推荐效果，评价指标仍是 hit ratio. 基于前面讨论，数据集用 *DS*7，算法 flowRec、flowRecK 的参数 k 设置为 100. 为保证对比公平，算法 FlowRecommender 用 Python 语言实现，并在同平台上运行. 由于文献[8]没给出算法 FlowRecommender 的最优参数设置，这里基于我们对该算法实验中参数的优化调整，将其相关参数设置如下：置信阈值(Confidence threshold) $\sigma_{\mathrm{conf}}=0.7$，子序列匹配的最大后向位置(Maximum backward location) $K=3$，距离阈值(Distance threshold) $\sigma_d=0.3$. 因 FlowRecommender 用模式表推荐，每次推荐结果列表中元素数量不定，所以它不受本文算法参数 k 的影响. 又因其推荐结果为多个，所以其 hit ratio 按式(7)计算.

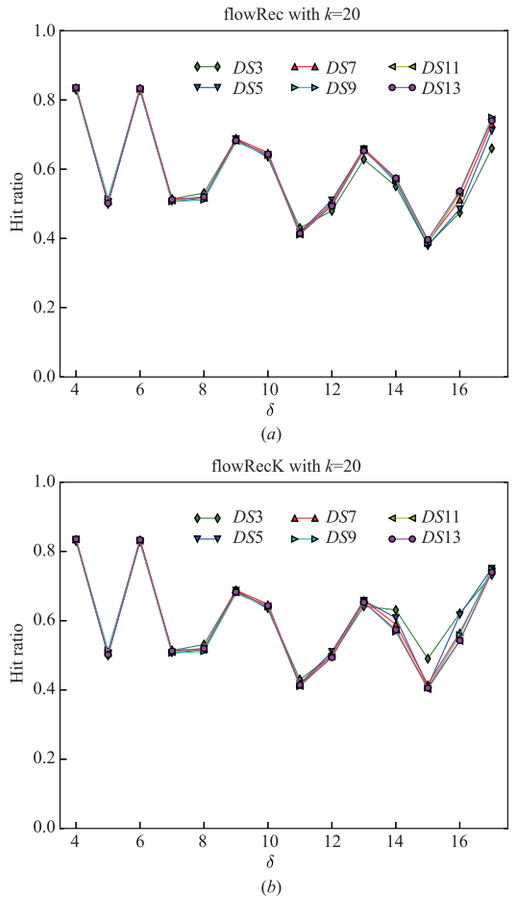

图4 固定k=20，算法flowRec和flowRecK在不同数据集下hit ratio对比

此外，为更细致地查验本文算法的推荐效果，我们又实现了随机推荐算法 ranRec：从 E 中随机选一完整实例 e；如果 e 的长度大于 $\hat{e}^{\delta}$ 的长度 δ，则将 e 的活动序列中第 $\delta+1$ 个活动添加到集合 $na(\hat{e}^{\delta})$ 中；重复上述过程，直到 $na(\hat{e}^{\delta})$ 中含有 k 个活动为止；然后从 $na(\hat{e}^{\delta})$ 中随机选一活动，将其推荐给 $\hat{e}^{\delta}$. 算法 ranRec 的参数 k 同样设置为 100. 因 ranRec 的推荐结果只有一个，所以其 hit ratio 按式(6)计算. 因它的随机性，在此对每个 $\hat{E}^{\delta}_{\mathrm{Test}}$ 运行该算法 100 次，算出 hit ratio 的平均值. 图 7 给出了这四种算法的 hit ratio 对比.

从图 7 可看出，算法 flowRec、flowRecK 和 FlowRecommender 的 hit ratio 明显好于算法 ranRec. 当 $\delta<11$，flowRec 和 flowRecK 的推荐效果基本一样，说明 δ 较小时采用本文相似度计算方法不能区分二者的推荐效果，宜用其他更好方法，这将是我们下一步工作；当 $\delta\geqslant 11$，flowRecK 的推荐效果好于 flowRec，说明当 δ 较大，亦即不完整实例与完整实例的公共活动较多时采用本

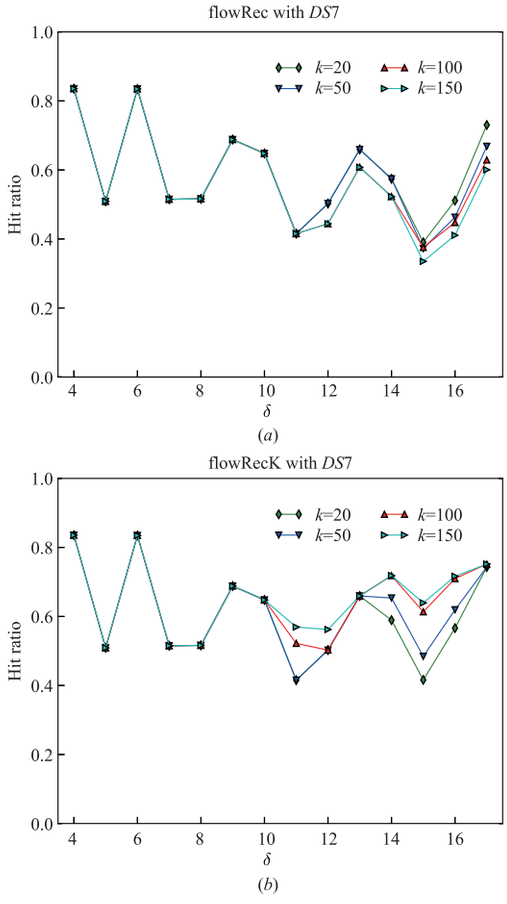

图5　数据集*DS*7下算法flowRec和flowRecK在不同k下hit ratio对比

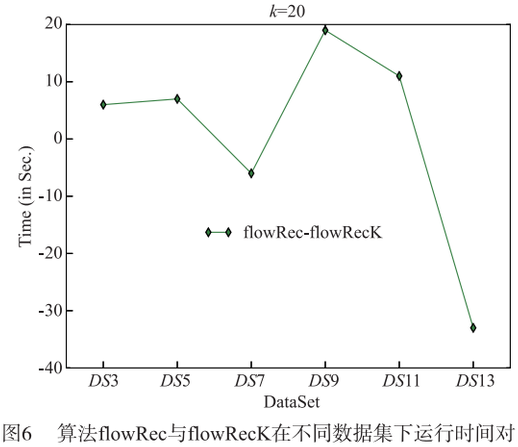

图6　算法flowRec与flowRecK在不同数据集下运行时间对比

文相似度计算方法能区分二者的推荐效果. 因 flowRecK 与 FlowRecommender 的推荐结果同为多个,在此对二者的推荐效果进行分析. 从图中可看出,当 $\delta=4$、5、6、9、13、17 时 flowRecK 的推荐效果好于 FlowRecommender. 这是因为:当 $\delta=4$、5、6,由 *DS*7 构造的 $\hat{E}^{\delta}_{\text{Test}}$ 具有较多的下一执行活动的可能性,而 FlowRecommender 受限于只为每个候选推荐活动建立一个模式表;当 $\delta=9$、13、17,

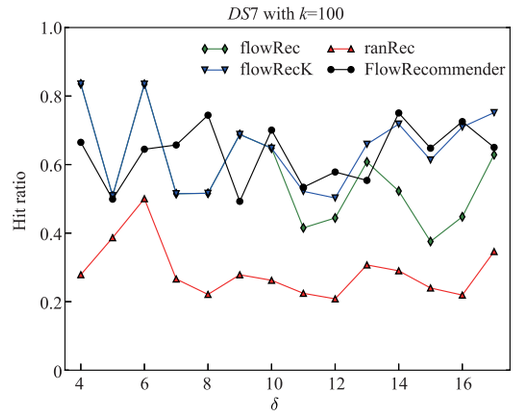

图7　算法flowRec、flowRecK、ranRec及FlowRecommender在数据集*DS*7下、k=100时的推荐效果对比

$\hat{E}^{\delta}_{\text{Test}}$ 中每个不完整实例包含了一些重复出现的子活动序列,这对算法 FlowRecommender 建立准确的模式表产生干扰. 算法 flowRecK 不受上述两情况的影响,能利用 $\hat{e}^{\delta}$ 各个活动的位置信息推荐. 此外,从图 7 还看出,当 $\delta=7$、8、10、12、14 时 flowRecK 的推荐效果差于 FlowRecommender. 这是因为此情况下 $\hat{E}^{\delta}_{\text{Test}}$ 中每个不完整实例具有较少的下一执行活动的可能性,FlowRecommender 利用统计方法建立的模式表就比较准确. 当 $\delta=7$、8,不完整实例的长度较短,flowRecK 的 hit ratio 与 FlowRecommender 的差别比较大;但当 $\delta=10$、12、14,flowRecK 利用了不完整实例的更多活动信息,其推荐准确性随之提高,因此其 hit ratio 与 FlowRecommender 的差别较小,只稍微低一点.

可见,算法 flowRecK 与 FlowRecommender 各有优劣,且 flowRecK 相比于 FlowRecommender 劣势的地方其 hit ratio 仅仅低一点,不超过 0.04. 这说明本文算法是有效的和可行的. 上述本文算法的劣势是我们下一步努力改进的方向.

6　结束语

本文给出了一种将实例库转化成矩阵的方法,讨论了不完整实例与完整实例的相似度计算;并提出了推荐算法 flowRec 和 flowRecK,对比分析了它们在仿真数据集上的推荐效果. 实验结果表明:大部分情况下算法 flowRecK 的推荐效果好于 FlowRecommender,且推荐效果不如 FlowRecommender 的地方,其 hit ratio 仅低一点,不超过 0.04. 从而证实本文算法是可行的和有效的. 通过实验还知当不完整实例的长度较短时,本文相似度计算方法不恰当,需进一步改进. 将推荐系统领域的协同过滤方法用于工作流活动推荐是本文的新意.

未来的工作包括:设计当不完整实例较短时更好的相似度计算方法;改进当不完整实例和完整实例都

含有较多重复的子序列时的推荐方法;在工作流实例中加入更多的信息,如用户信息、资源使用信息、及时间信息等,以使活动推荐更符合实际.

参考文献

[1] MARLON D, WIL M P Aalst, ARTHUR H. Process-Aware Information Systems: Bridging People and Software Through Process Technology[M]. New Jersey, NJ, USA: John Wiley & Sons, 2005. 21 – 34.

[2] DOMINIC M, JOACHIM H, MARKUS H, et al. IT support for release management processes in the automotive industry[A]. Business Process Management[C]. Heidelberg: Springer Verlag, 2006. 368 – 377.

[3] JAAP G, J M AKKERMANS. Value-based requirements engineering: exploring innovative E-commerce ideas[J]. Requirements Engineering, 2003, 8(2): 114 – 134.

[4] RICHARD L, MANFRED R. IT support for healthcare processes-premises, challenges, perspectives[J]. Data & Knowledge Engineering, 2007, 61(1): 39 – 58.

[5] GIANNIS V, DIMITRIS G, GREGORIS M. Modeling e-government service workflows through recurring patterns [A]. Electronic Government[C]. Heidelberg: Springer Verlag, 2004. 483 – 488.

[6] JOSÉFÉLIX M, CARLOS B, FRANCISCO S. Modeling administrative procedures to improve information to the public[A]. Advancing Democracy, Government and Governance[C]. Heidelberg: Springer Verlag, 2012. 155 – 169.

[7] BELA M, MANFRED R, JOHANNES B. Unleashing the effectiveness of process-oriented information systems: problem analysis, critical success factors, and implications[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 2008, 38(3): 280 – 291.

[8] JI Z, QING L, KAI X. Flowrecommender: a workflow recommendation technique for process provenance[A]. Proceedings of the 8th Australasian Data Mining Conference[C]. Melbourne, Australia: ACS, 2009. 55 – 61.

[9] AGNES K, THOMAS H, ANDREAS O. Recommendation-based editor for business process modeling[J]. Data & Knowledge Engineering, 2011, 70(6): 483 – 503.

[10] 曹斌,尹建伟,邓永光,等. 一种基于近距离最大子图优先的业务流程推荐技术[J]. 计算机学报, 2013, 36(2): 263 – 274.
CAO Bin, YIN Jian-wei, DENG Yong-guang, et al. A near neighbour and maximal subgraph first based business process recommendation technique[J]. Chinese Journal of Computers, 2013, 36(2): 263 – 274. (in Chinese)

[11] YING L, BIN C, LIDA X, et al. An efficient recommendation method for improving business process modeling[J]. IEEE Transactions on Industrial Informatics, 2014, 10(1): 502 – 513.

[12] KUANG M, LIDAN S, JU F, et al. Competence-based song recommendation: matching songs to one's singing skill[J]. IEEE Transactions on Multimedia, 2015, 17(3): 396 – 408.

[13] AMOS A, AVINATAN H, SARIT K, et al. Movie recommender system for profit maximization[A]. Proceedings of the 7th ACM Conference on Recommender Systems[C]. New York, NY, USA: ACM, 2013. 121 – 128.

[14] RAYMOND J M, LORIENE R. Content-based book recommending using learning for text categorization[A]. Proceedings of the 5th ACM Conference on Digital Libraries[C]. New York, NY, USA: ACM, 2000. 195 – 204.

[15] 黄震华. 云环境下 top-n 推荐算法[J]. 电子学报, 2015, 43(1): 54 – 61.
HUANG Zhen-hua. Top-n recommendation algorithms for cloud data[J]. Acta Electronica Sinica, 2015, 43(1): 54 – 61. (in Chinese)

[16] 黄震华,张波,方强,等. 一种社交网络群组间信息推荐的有效方法[J]. 电子学报, 2015, 43(6): 1090 – 1093.
HUANG Zhen-hua, ZHANG Bo, FANG Qiang, et al. An efficient algorithm of information recommendation between groups in social networks[J]. Acta Electronica Sinica, 2015, 43(6): 1090 – 1093. (in Chinese)

[17] MUKUND D, GEORGE K. Item-based top-n recommendation algorithms[J]. ACM Transactions on Information Systems, 2004, 22(1): 143 – 177.

[18] Greg L, BRENT S, JEREMY Y. Amazon. com recommendations: item-to-item collaborative filtering[J]. IEEE Internet Computing, 2003, 7(1): 76 – 80.

[19] BADRUL S, GEORGE K, JOSEPH K, et al. Item-based collaborative filtering recommendation algorithms[A]. Proceedings of the 10th International Conference on World Wide Web[C]. New York, NY, USA: ACM, 2001. 285 – 295.

[20] JOSEPH A K, BRADLEY N M, DAVID M, et al. Grouplens: applying collaborative filtering to usenet news[J]. ACM Communications, 1997, 40(3): 77 – 87.

[21] CHRISTIAN D, GEORGE K. A comprehensive survey of neighborhood-based recommendation methods[A]. Recommender Systems Handbook[C]. USA: Springer US, 2011. 207 – 144.

[22] HELEN S, BARBARA W, BOUDEWIJN D, et al. Supporting flexible processes through recommendations based on history[A]. Business Process Management[C]. Heidelberg: Springer Verlag, 2008. 51 – 66.

[23] RON Z. A programmer's guide to Data Mining: The Ancient Art of the Numerati[OL]. http://guidetodata-

mining. com,2016 - 03 - 10.
[24] KOEN V,BART G. Unifying nearest neighbors collaborative filtering[A]. Proceedings of the 8th ACM Conference on Recommender Systems[C]. New York,USA:ACM,2014. 177 - 184.
[25] JAMES C,YIPING K,WILFRED N. Graphgen:A Graph Synthetic Generator[OL]. http://www. cse. ust. hk/graphgen,2016 - 03 - 10.
[26] WOOK-SHIN H, JINSOO L, MINH-DUC P, et al. iGraph: a framework for comparisions of disk-based graph indexing techniques[J]. Proceedings of the VLDB Endowment,2010,3(1):449 - 459.
[27] 印桂生,张亚楠,董宇欣,等. 基于受限信任关系和概率分解矩阵的推荐[J]. 电子学报,2014,42(5):904 - 911. YIN Gui-sheng,ZHANG Ya-nan,DONG Yu-xin,et al. A constrained trust recommendation using probabilistic matrix facctorization[J]. Acta Electronica Sinica,2014,42(5):904 - 911. (in Chinese)
[28] JINGWEI X,YUAN Y,HANGHANG T,et al. Ice-breaking:mitigating cold-start recommendation problem by rating comparison[A]. Proceedings of the 24th International Conference on Artificail Intelligence[C]. USA: AAAI Press,2015. 3981 - 3987.
[29] GEORGE K. Evaluation of item-based top-n recommendation algorithms[A]. Proceedings of the 10th International Conference on Information and Knowledge Management[C]. New York,NY,USA:ACM,2001. 247 - 254.
[30] HORATIU D,MAREK G,NEGAR H,et al. On-demand feature recommendations derived from mining public product descriptions[A]. Proceedings of the 33rd International Conference on Software Engineering[C]. New York,NY,USA:ACM,2011. 181 - 190.

作者简介

陈广智　男,1981 年 10 月出生,河南太康人. 现为中山大学博士研究生,从事工作流推荐、类比推理及迁移学习方面的有关研究.
E-mail:chgzhi@ mail2. sysu. edu. cn

何　文　男,1990 年 8 月出生,广东茂名人. 于 2014 年获中山大学软件工程专业学士学位,现于中山大学数据科学与计算机学院攻读硕士学位. 研究方向为数据挖掘和推荐系统.