文章编号:1671-4598(2020)03-0235-07 DOI:10.16526/j.cnki.11—4762/tp.2020.03.049 中图分类号:TP301.4 文献标识码:A

基于模型结构和事件日志的流程相似度计算

张智慧[1], 吴 珏[1], 杨福军[2]
(1. 西南科技大学 计算机科学与技术学院, 四川 绵阳 621000;
2. 中国空气动力研究与发展中心 计算空气动力研究所, 四川 绵阳 621000)

摘要: 流程相似度的计算在企业业务流程管理中具有重要作用; 目前相似度的计算主要存在两个问题: 1) 大多数相似度计算方法只考虑模型结构或事件日志, 导致算法不够精确; 2) 综合考虑了模型结构和事件日志的算法复杂度高且效率低; 因此, 提出了一种改进的流程模型结构和事件日志相结合的方法; 首先将流程模型结构中的紧邻活动转化为邻接矩阵, 然后根据事件日志中的行为信息对邻接矩阵进行加权得到加权邻接矩阵, 最后采用符合距离度量特性的矩阵间距离的算法来度量流程间相似度; 通过实验与 MDS、GED 以及 WBPG 等算法进行对比, 所提方法的准确率更高, 为 99.51%, 计算效率也更高。

关键词: 流程相似度; 模型结构; 事件日志; 加权邻接矩阵; 矩阵间距离

Calculating Similarity Between Business Process Based on Model Structure and Event Log

Zhang Zhihui[1], Wu Jue[1], Yang Fujun[2]
(1. College of Computer Science and Technology, Southwest University of Science and Technology, Mianyang 621000, China;
2. Institute of Computational Aerodynamics, China Aerodynamics Research and Development Center, Mianyang 621000, China)

Abstract: The calculating of business process similarity plays an important role in enterprise business process management. At present, there are two main problems in the calculation of similarity: one is that most methods only consider the model structure or event log, which results in inaccurate algorithm, the other is that the algorithm considering the model structure and event log has high complexity and low efficiency. An improved approach is proposed to calculate similarity by combining model structure and event log. Firstly, constructs the adjacency matrix based on the adjacent activities of process model structure. Then the weight adjacent matrix is obtained by weighting the adjacent activities according to the behavior information in the event log. Finally, the inter—matrix distance algorithm conforming to the distance metric is used to measure the business process similarity. By comparing with the algorithms such as matrix distance similarity (MDS), graph edit distance (GED) and weight business process graph (WBPG), the accuracy of the proposed approach is 99.51%, and the calculation efficiency is higher.

Keywords: business process similarity; process model structure; event log; weight adjacent matrix; distance between matrix

0 引言

作为企业业务的一种描述, 业务流程模型在企业的发展过程中扮演着重要的角色, 能够有效提高公司的效率, 对企业的业务运营具有重要的指导意义。随着企业业务的增多, 相应的业务流程也越来越多, 大型企业往往保存着成千上万的业务流程。构建新的流程复杂且耗时, 若能为相关人员推荐流程库中的相似流程, 并在已有流程的基础上进行修改将大幅度缩短工作时间, 减少重复工作, 而流程推荐的核心是流程相似度的计算, 因此流程相似度计算成为一个研究热点[1-3]。

收稿日期:2019-08-23; **修回日期**:2019-09-03。
基金项目: 国家重点基础研究发展计划基金项目(2014CB744100),西南科技大学博士基金(13zx7102)。
作者简介:张智慧(1993-),男,河南周口人,硕士研究生,主要从事前端框架、工作流管理方向的研究。
吴 珏(1978-),女,四川绵阳人,副教授,博士研究生,CCF 会员,主要从事大数据技术、数据挖掘、人工智能方向的研究。

1 相关工作

近年来流程相似度计算的问题取得了较好的成果, 现有的方法主要包含基于文本内容相似度[4-5]、基于模型结构相似度[6-16]和基于流程行为相似度[17-19]三种。

基于文本内容的相似度主要是根据流程对应节点的文本内容相似度来度量, 包括字符串编辑距离、语义相似度等, 这种方法不考虑模型结构仅比较对应节点的文本内容, 比较简单, 然而当流程模型结构发生改变而文本内容不变的时候, 该方法无法识别, 导致准确率较低。

作为流程的重要属性, 模型结构能够很好地表示活动间的逻辑关系, 决定数据以及控制流的前进方向。目前基于模型结构相似度的方法有三种, 包括矩阵间距离、树的编辑距离和图的编辑距离。文献 [6-10] 根据变迁紧邻关系, 将流程模型结构转化邻接矩阵, 定义矩阵间的距离来计算流程结构相似度。文献 [11-12] 将流程模型结构转化成对应的结构树, 通过计算树的编辑距离得到流程相似度。文献 [13-16] 的相关工作是将流程模型转化为标准的业务

流程图，通过图之间的匹配度来衡量流程之间的相似度。这些算法以更直观的方式表示模型结构，从而更方便的计算流程相似度，相比文本内容相似度，准确率更高。不足之处在于将模型结构转化为邻接矩阵、树或图的过程中会损失能够区分不同流程行为的语义信息，导致无法区分流程活动之间的复杂关系，如选择、并行关系和循环关系，因此需要结合流程结构和行为信息来更准确的表示业务流程，使得计算结果更加精确。

基于流程行为相似度的计算方法[17-19]主要是根据流程的模型结构，模拟流程中活动间的逻辑关系得到活动执行序列，从而获取流程行为信息。由于流程的执行过程与相关人员关系紧密，流程的行为信息往往会包含参与者的行为习惯，因此通过模拟的形式获取行为信息的方式在大多数实际应用中是无效的，需要一种更加客观的方法记录行为信息。事件日志是流程执行过程的客观观察，包含流程的语义、轨迹以及参与者的执行习惯，反映了流程的真实执行情况，具有极高的参考意义。

针对上述问题，周长红等人[20-21]提出了一种将模型结构和行为信息结合的方法，该方法根据行为信息对图进行加权得到加权业务流程图，基于加权图的编辑距离来计算流程相似度，提高了准确率。但是加权图的编辑操作需要对节点和边进行处理，包括节点和边的的插入、删除和替换，这些操作的定义困难，复杂度高且时间效率较低。

本文在此基础上提出一种改进的将模型结构和事件日志结合的相似度计算方法。首先用邻接矩阵表示流程模型结构，然后根据事件日志中的轨迹频次对邻接矩阵进行加权，得到加权邻接矩阵，最后给出矩阵间距离算法的定义，计算流程相似度。本文基于矩阵间距离的计算只需要对差异矩阵求和，比加权图的编辑距离计算更简单，效率更高。

2 流程相似度计算方法

本文改进的基于模型结构和事件日志的流程相似度计算方法的流程如图 1 所示，主要包含三部分：1）首先对流程模型文件中的流程进行预处理，即将原始流程转化为仅包含紧邻活动关系的邻接矩阵；2）获取流程的事件日志，对预处理后的邻接矩阵进行加权；3）根据矩阵间距离算法对加权矩阵进行计算，得到矩阵间距离，返回相似度。

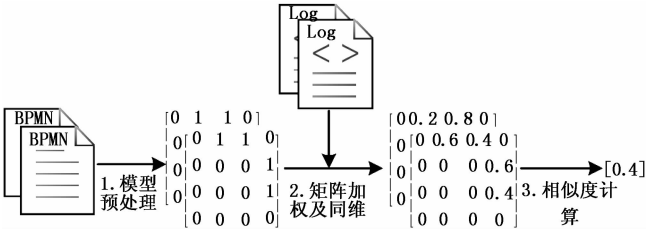

图 1 流程相似度计算流程

2.1 模型预处理

模型预处理是将模型转化为只包含紧邻活动关系的邻接矩阵。本文用 Petri 网模型表示流程的模型结构信息，因此先给出相关定义。

定义 1（Petri 网）：一个流程的 Petri 网是一个三元组 $PN=(P, A, F)$，其中：

a）P 是一个有限的库所集（place）；

b）A 是一个有限的变迁集（transition），$P \cup A \neq \varnothing$ 且 $P \cap A = \varnothing$；

c）$F \subseteq (P \times A) \cup (A \times P)$ 是一个表示流程关系的有向弧集，包含库所和变迁；

定义 2（紧邻活动）：对于任意流程 PN，设活动集合为 A，有向弧集合 $F=\{f_1, f_2, \cdots, f_n\}$，活动 a，b A，判断活动 a，b 是否为紧邻活动的准则是当且仅当在 PN 中存在有向弧 f_i，f_{i+1}，使得活动 a 经过有向弧 f_i，f_{i+1}可以到达活动 b，且 a 和 b 之间不存在其他活动，其中 i（$\{1, 2, \cdots, n-1\}$。

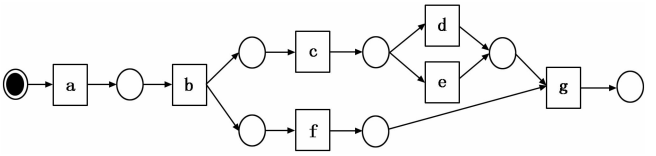

图 2 某流程 BP_1 的 Petri 网模型

图 2 是某流程 BP_1 的 Petri 网模型，本文不考虑模型的库所，只是为了区分复杂结构。流程 BP_1 的活动集合 $A=\{a, b, c, d, e, f, g\}$，紧邻活动有 ab，bc，cd，ce，dg，eg，bf，fg。

定义 3（邻接矩阵）：以流程模型 PN 中的紧邻活动关系为元素的矩阵称为邻接矩阵（Adjacent Matrix，AM），流程的活动集为 A，矩阵元素的值表示如下：

$$AM_{i,j}=\begin{cases}1 & \text{若 } a_i, a_j \text{ 为紧邻活动}\\ 0 & \text{其他}\end{cases} \tag{1}$$

其中：a_i，a_j A。

对于任意给定流程，可以根据如下方法构建邻接矩阵：设流程有 n 个不同活动，首先初始化一个 $n \times n$ 的矩阵，如图 2 所示流程共包含 7 个不同的活动 a、b、c、d、e、f、g，因此图 2 对应一个 7×7 的邻接矩阵。对于流程模型中的任意两个活动 a_i，a_j，若活动 a_i，a_j 为紧邻活动，邻接矩阵 $\boldsymbol{AM}_{i,j}$对应位置填 1，即 $\boldsymbol{AM}_{i,j}=1$，否则 $\boldsymbol{AM}_{i,j}=0$。根据流程 BP_1 构建邻接矩阵 $\boldsymbol{AM}$，

$$\boldsymbol{AM}=\begin{bmatrix}0&1&0&0&0&0&0\\0&0&1&0&0&1&0\\0&0&0&1&1&0&0\\0&0&0&0&0&0&1\\0&0&0&0&0&0&1\\0&0&0&0&0&0&1\\0&0&0&0&0&0&0\end{bmatrix}$$

2.2 矩阵加权及同维

邻接矩阵仅考虑流程模型结构的紧邻活动关系，未考虑紧邻活动关系的重要性，因此本文提出加权邻接矩阵的概念。加权邻接矩阵是在邻接矩阵的基础上，根据事件日志中的行为信息对每一组紧邻活动进行加权，从而将日志

行为信息添加到业务流程模型的邻接矩阵。在流程实际运行过程中，可能会出现流程执行异常或日志记录错误的情况，这种异常或错误通常被称为噪声，为了减小噪声的影响，从而有效地提取与分析行为信息，本文根据文献[10]提出的降噪方法对事件日志进行处理，删除所占比例小于给定阈值 δ 的轨迹，本文设置阈值 $\delta=0.01$，活动已经按照执行时间先后排序。经过处理，流程 BP_1 的简单事件日志如表 1 所示。

表 1　某流程 BP_1 的日志信息

轨迹	频次
$\{a,b,c,d,f,g\}$	64
$\{a,b,c,e,f,g\}$	16
$\{a,b,c,f,d,g\}$	56
$\{a,b,c,f,e,g\}$	14
$\{a,b,f,c,d,g\}$	12
$\{a,b,f,c,e,g\}$	3

假设该流程的简单事件日志为 L，轨迹 τ 在事件日志 L 中的执行频次为 $\#_{frequency}(\tau, L)$。流程 BP_1 共包含 6 个不同轨迹，第 i 个轨迹表示为 τ_i（$1\leqslant\tau\leqslant6$），则 6 个轨迹的执行频次为 $\#_{frequency}(\tau_1, L)=64$，$\#_{frequency}(\tau_2, L)=16$，$\#_{frequency}(\tau_3, L)=56$，$\#_{frequency}(\tau_4, L)=14$，$\#_{frequency}(\tau_5, L)=12$，$\#_{frequency}(\tau_6, L)=3$。

2.2.1　构建加权邻接矩阵

定义 4（加权邻接矩阵）：以邻接矩阵 AM 为基础，每个元素对应的紧邻活动在简单事件日志 L 中出现的次数除以轨迹总数得到新的元素值，构成加权邻接矩阵（Weight Adjacent Matrix，WAM），即：

$$WAM_{i,j}=\begin{cases}\dfrac{\#_{frequency}(\{a_i,a_j\},L)}{w} & AM_{i,j}=1\\ 0 & AM_{i,j}=0\end{cases}\tag{2}$$

其中：w 表示轨迹总数。

以活动 c，d 为例，在轨迹 τ_1、τ_3、τ_5 中，活动 c 之后的确有活动 d 出现，频次分别为 64、56 和 12，因此 $\#_{frequency}(\{c,d\},\tau_1)=64$，$\#_{frequency}(\{c,d\},\tau_3)=56$，$\#_{frequency}(\{c,d\},\tau_5)=12$。活动 c，d 对应活动集的下标分别为 2 和 3，对应矩阵的第 2 行第 2 列，因此活动 c，d 对应的权值 $\boldsymbol{WAM}_{2,3}=(64+56+12)/(64+16+56+14+12+3)=0.8$，根据上述方法可得，流程 BP_1 对应的加权邻接矩阵 $\boldsymbol{WAM}=\begin{bmatrix}0&1&0&0&0&0&0\\0&0&1&0&0&1&0\\0&0&0&0.8&0.2&0&0\\0&0&0&0&0&0&0.8\end{bmatrix}$

$\begin{bmatrix}0&0&0&0&0&0&0.2\\0&0&0&0&0&0&1\\0&0&0&0&0&0&0\end{bmatrix}$。

对流程模型而言，将模型结构转化为邻接矩阵的问题是出现分支时无法区分分支中的活动之间是选择关系还是并行关系，而加权邻接矩阵可以通过紧邻活动的不同权重来区分不同的分支结构。具体来说，当两个活动为并行关系时，这两个活动与其最近的公共直接前驱活动组成的紧邻活动对应的权值均相同。如图 2 所示活动 c，f 为并行关系，活动 b 为两者的公共直接前驱活动，由 ***WAM*** 可知，活动 b，c 和活动 b，f 对应的权值 $\boldsymbol{WAM}_{2,3}$、$\boldsymbol{WAM}_{2,6}$ 均为 1，且并行关系开始前的活动 a，b 紧邻关系的权值 $\boldsymbol{WAM}_{1,2}$ 也为 1。当两个活动为选择关系时，则这两个活动与其公共直接前驱活动组成的紧邻活动对应的权值一般不同，且权值一定小于选择关系开始前的紧邻关系的权值。如图 2 所示，活动 d，e 为选择关系，活动 c 为两者的公共直接前驱活动，由 ***WAM*** 可知，活动 c，d 对应的权值 $\boldsymbol{WAM}_{3,4}$ 为 0.8，活动 c，e 对应的权值 $\boldsymbol{WAM}_{3,5}$ 为 0.2，均小于选择关系开始前的活动 b，c 对应权值为 1 的 $\boldsymbol{WAM}_{2,3}$。此外，本文将流程模型转换成加权矩阵后还可以区分循环结构。若加权邻接矩阵的对角线的元素全为 0，说明该流程不存在自循环结构。若加权邻接矩阵中的某个元素的值大于 1，说明对应的紧邻活动之间存在循环，从而推断出该流程存在循环结构。

2.2.2　同维加权邻接矩阵

由于流程活动集的大小可能不同，导致矩阵的行列数不一定相等，想要获取差异矩阵，需要将加权矩阵同维化，转换成行列相等的矩阵。假设 ***WAM***、***WAM*** 分别代表两个流程的加权邻接矩阵，可将两个矩阵的对应元素相减得到差异矩阵 ***DM***，用 ***DM*** 度量流程之间的差异。图 3 是流程 BP_2 的 Petri 网模型，共包含 6 个活动，设流程 BP_2 的加权邻接矩阵为 $\boldsymbol{WAM}=\begin{bmatrix}0&1&0&0&0&0\\0&0&1&0&0&0\\0&0&0&0.6&0.4&0\\0&0&0&0&0&0.6\\0&0&0&0&0&0.4\\0&0&0&0&0&0\end{bmatrix}$。

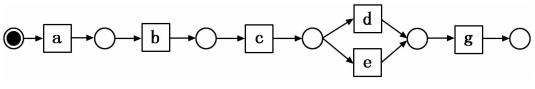

图 3　某流程 BP_2 的 Petri 网模型

定义 5（同维加权邻接矩阵）：对于任意两个流程 P 和 P，A、A 分别是两个流程的活动集合，***AM*** 和 ***AM***（为两个流程所对应的加权邻接矩阵，WAM 和 WAM 表示同维加权邻接矩阵，其维度为 $n=|A\cup A|$。设总活动集 $S=A\cup A(=\{a_1,a_2,\cdots,a_n\}$，生成两个函数 $f(x)$ 和 $f((x)$，分别将活动集 S 的活动下标和活动集 A，A 的活动下标形成映射关系，使得 S 中的下标为 x 的活动，A 中下标为 $f(x)$、A 中下标为 $f(x)$ 的活动表示的是同一个活动。$\boldsymbol{WAM}_{i,j}$ 和 $\boldsymbol{WAM}_{i,j}$ 分别表示 ***WAM*** 和 ***WAM*** 中第 i 行，第 j 列的元素，它们的元素值可以表示为如下形式：

$$WAM_{i,j}=\begin{cases}AM_{f(i),f(j)} & AM_{f(i),f(j)}\neq\phi\\ 0 & \text{其他}\end{cases}\tag{3}$$

$$WAM'_{i,j}=\begin{cases}AM'_{f'(i),f'(j)} & AM'_{f'(i),f'(j)}\neq\phi\\ 0 & \text{其他}\end{cases} \tag{4}$$

由流程 BP_1、BP_2 以及定义 5 可知，处理后的总活动集 $S=\{a,b,c,d,e,f,g\}$，和流程 BP_1 的活动集 A 相同，因此流程 BP_1 的同维加权邻接矩阵 ***WAM*** 不改变。流程 BP_2 的活动集 A 与 S 不同且 $A<S$，因此矩阵大小由 6×6 扩大为 7×7，同维加权邻接矩阵 $\boldsymbol{WAM}=\begin{bmatrix}0&1&0&0&0&0&0\\0&0&1&0&0&0&0\\0&0&0&0.6&0.4&0&0\\0&0&0&0&0&0&0.6\\0&0&0&0&0&0&0.4\\0&0&0&0&0&0&0\\0&0&0&0&0&0&0\end{bmatrix}$。由图 3 可知流程 BP_2 不存在活动 f，其中 f 对应矩阵的第六行和第六列，所以 ***WAM*** 对应行列的元素全为 0。

2.3 相似度计算

本文用流程矩阵间距离表示流程相似度，为了更直观的表示矩阵间的距离，参照矩阵间距离相似度（Matrix Distance Similarity，MDS）算法[9]的定义，对公式修改如下。

定义 6（矩阵间距离）：设 ***WAM***，***WAM*** 分别为流程 P、P'对应的同维加权邻接矩阵，C_0 为 $\boldsymbol{WAM}_{i,j}\neq0$ 且 $\boldsymbol{WAM}_{i,j}\neq0$ 的个数，C_1、C_2 分别代表 $\boldsymbol{WAM}_{i,j}$、$\boldsymbol{WAM}_{i,j}$中非零元素的个数。将矩阵 ***WAM*** 和 ***WAM*** 对应位置做减法，得到差异矩阵 ***DM***，对 ***DM*** 的每一个元素取绝对值得到 $|\boldsymbol{DM}|$，则流程间相似度可以表示为：

$$D(P,P')=\frac{\sum_{i,j}^{n}|DM(i,j)|}{C_1+C_2-C_0} \tag{5}$$

相似度计算的核心是总活动集 S 的构建以及差异矩阵 ***DM*** 的计算。设 m 是流程 P、P 的活动集个数较大值，构建活动集 S 的算法的时间复杂度是 $O(m^2)$，计算差异矩阵是通过两个同维矩阵对应位置相减，矩阵的大小等于活动集 S 的大小 n，该部分的时间复杂度是 $O(n^2)$，由于 $m\leqslant n$，所以总的时间复杂度为 $O(n^2)$。

计算 BP_1，BP_2的相似度 $D(BP_1,BP_2)=0.35$，因此流程 BP_1、BP_2 之间的相似度为 0.35。

3 验证实验

3.1 实验数据

目前相关算法的实验结果是在不同数据集下得到的，不能直接用来比较算法。针对该问题，周长红等人[20-21]选取了一个真实的保险理赔申请流程（表 2，P_0），通过人为的增加或删除一些分支，以及改变不同分支结构的权重，构造了 13 个流程变体（表 2，P_1-P_{13}），并且记录所有变体的日志信息，最后获得的数据集共包含 5 445 个轨迹，26 825个活动。该数据集能更好的满足模型结构和事件日志结合的相关算法测试，可以更公平的进行算法对比。

表 2 流程基本信息

流程	组别	变化	轨迹数量	活动数量
P_0		—	320	1600
P_1	A 组	删除一个分支	480	2400
P_2		删除一个分支	280	1400
P_3		删除两个分支	405	2025
P_4		删除两个分支	440	2640
P_5		删除两个分支	560	2240
P_6		删除两个分支	360	1440
P_8		添加选择分支	160	640
P_9		添加选择分支	340	1700
P_7	B 组	删除并行分支	240	1440
P_{10}		添加并行分支	560	2800
P_{11}	C 组	选择变并行	520	2600
P_{12}		并行变选择	428	2140
P_{13}		并行变选择	352	1760

本文将上述流程转换为对应的加权邻接矩阵，以进行相似度的计算，转化结果如图 4 所示。

3.2 实验结果及分析

实验选取相关典型算法，包括模型结构相似度、模型行为相似度、模型结构和行为结合的相似度算法，与本文方法进行比较，相关算法的信息如表 3 所示。

表 3 对比算法相关信息

算法序号	算法名称	计算目标
A_1	MDS[9]	模型结构
A_2	GED[15]	模型结构
A_3	PTS++[22]	模型行为
A_4	WBPG[20-21]	模型结构和行为
A_5	本文方法	模型结构和行为

实验选取 P_0 为参考模型，采用上述算法计算 13 个变体流程与 P_0 的相似度，结果如表 4 所示。

表 4 不同算法的实验结果

流程	组别	A_1	A_2	A_3	A_4	A_5
P_1	A 组	0.75	0.89	0.35	0.80	0.58
P_2		0.75	0.89	0.35	0.94	0.92
P_3		0.62	0.80	0.59	0.90	0.89
P_4		0.62	0.80	0.59	0.83	0.71
P_5		0.62	0.80	0.59	0.76	0.54
P_6		0.62	0.80	0.59	0.69	0.36
P_8		0.80	0.91	0.60	0.96	0.96
P_9		0.80	0.91	0.60	0.81	0.64
P_7	B 组	0.75	0.89	0.24	0.83	0.75
P_{10}		0.80	0.91	0.23	0.88	0.80
P_{11}	C 组	1.00	1.00	0.11	0.71	0.75
P_{12}		1.00	1.00	0.57	0.77	0.75
P_{13}		1.00	1.00	0.57	0.64	0.75

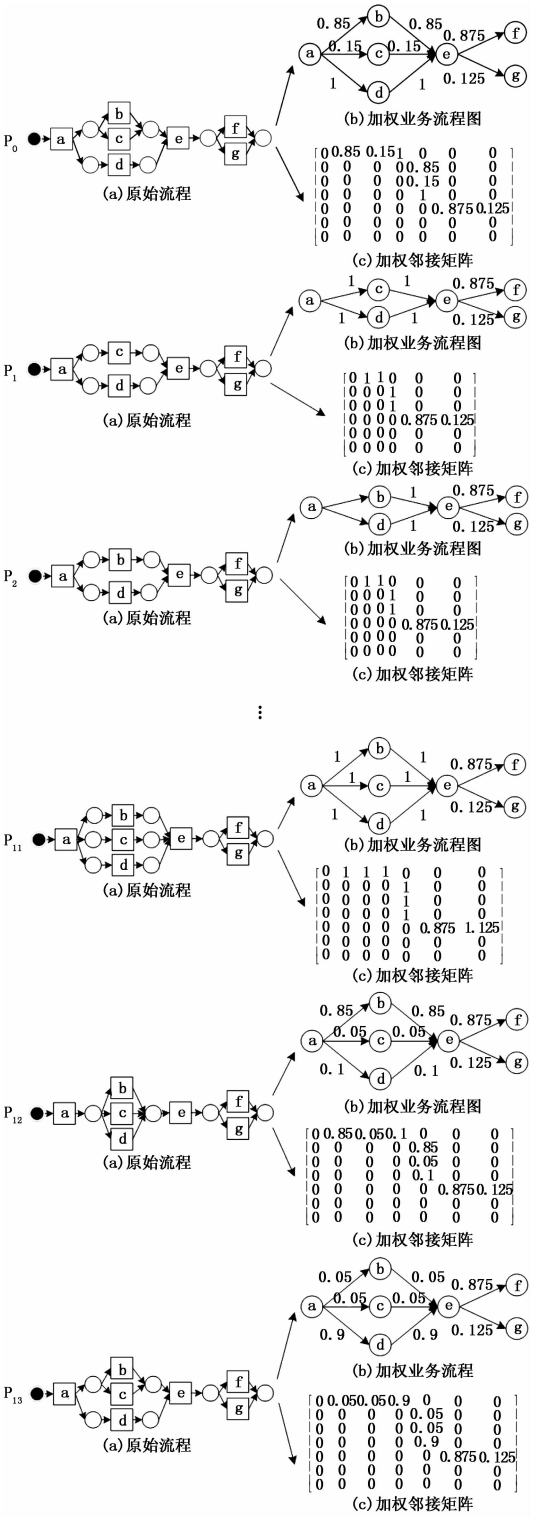

图 4　流程转化图

如表 4 所示，在 A 组中对选择分支进行删除或添加操作，分支的重要性发生的变化在算法 A_1～A_3 中无法体现。如 P_1 和 P_2 的结构相同（图 4），但 P_1 删除了分支 b，而 P_2 删除的是分支 c，两者与 P_0 的相似度不为 1，说明算法可以检测出分支结构的差异性。但 P_1 与 P_0 的相似度值和 P_2 与 P_0 的相似度值相同，说明以上算法没有检测到分支重要性的变化。而算法 A_4 和本文方法 A_5 既能检测出分支结构的变化，又能检测分支重要程度发生的变化。因为两种算法都考虑了模型结构，又根据事件日志中轨迹的执行次数对不同分支进行加权。另外，在 B 组中对并行分支进行删除或添加后对模型结构影响较大，但原有分支的重要性不受影响，因此本文方法计算的结果更接近基于结构相似度的算法 A_1 和 A_2 计算的结果，而与基于行为相似度的算法 A_3 计算的结果相差较大。

在 C 组中，P_{11} 中的选择分支变为并行分支，P_{12} 和 P_{13} 中的并行分支变为选择分支（图 4），而这两种变化均未引起模型结构的改变，但会引起事件日志的执行轨迹发生变化，因此，算法 A_1 和 A_2 只考虑模型结构无法区分流程中活动之间的选择关系和并行关系，而考虑了行为信息的 A_3－A_5 可以区分行为变化带来的影响。

综上所述，仅考虑模型结构无法检测分支重要程度的变化，也无法区分选择关系和并行关系，融入事件日志中的行为信息后能更准确的区分流程间的相似度。在上述算法中，只有本文方法和 A_4 同时考虑了模型结构和行为信息，效果更理想，但本文是将原始流程转化为加权邻接矩阵，而算法 A_4 是转化为加权业务流程图（见图 3），两者相似度的计算方式明显不同，所得相似度值也不同。

3.3　准确率评估

在上节中，本文详细分析了各种算法区分分支结构变化和重要性变化的能力。总的来讲，能检测各种分支变化的算法要优于只能检测部分分支变化的算法。但对都能检测出分支变化的算法而言，却无法判断哪种算法得到的相似度更准确。为了更好的对实验结果进行评估，文献［20］给出了由 15 位领域专家按 P_1～P_{13} 与 P_0 的相似性大小排序的结果以及对应流程变体的权重，并将该排序结果作为基准排序。本文将不同算法中计算的 13 个流程变体与 P_0 的相似度大小排序和基准排序结果进行比较，见表 5，并计算归一化折损累积增益（normalized discount cumulative gain，NDCG）[23] 评估算法的准确率。NDCG 的计算公式如下：

表 5　相似度大小排序结果

位次权重	基准排序	A_1	A_2	A_3	A_4	A_5
0.9	P_8	P_8	P_{12}	P_8	P_8	P_8
0.85	P_2	P_{12}	P_{11}	P_9	P_2	P_2
0.8	P_3	P_{11}	P_{13}	P_3	P_3	P_3
0.75	P_{10}	P_{13}	P_8	P_5	P_{10}	P_{10}
0.7	P_{12}	P_2	P_{10}	P_4	P_7	P_{12}
0.65	P_7	P_3	P_9	P_6	P_4	P_7
0.6	P_{11}	P_{10}	P_2	P_{12}	P_9	P_{11}
0.55	P_1	P_7	P_7	P_{13}	P_1	P_{13}
0.5	P_9	P_9	P_1	P_2	P_{12}	P_4
0.45	P_5	P_1	P_3	P_1	P_5	P_9
0.4	P_4	P_5	P_5	P_7	P_{11}	P_1
0.35	P_6	P_4	P_4	P_{10}	P_6	P_5
0.3	P_{13}	P_6	P_6	P_{11}	P_{13}	P_6

$$D_n = \begin{cases} r(n) & n = 1 \\ r(1) + \sum_{n=2}^{n} \frac{r(n)}{\log_2 n} & n > 1 \end{cases} \quad (6)$$

$$NDCG_n = \frac{D_n}{I_n} \times 100\% \quad (7)$$

其中：$r(n)$ 表示排在第 n 个位置的流程的权重；D_n 表示某个给定流程模型与其 n 个相关流程的折损累积增益，I_n 是基准排序中某个给定流程模型与其 n 个相关流程的理想折损累计增益（IDCG）。从公式（7）可知，DCG 与 IDCG 越接近，算法的 NDCG 值越高，准确率越高。

本文计算得到基准排序的 IDCG 值为 4.167，分别对不同算法的实验结果进行排序，计算相应的 DCG 值，最后得到每种算法的 NDCG 值，实验结果如表 6 所示。

表 6 DCG 与 NDCG

算法序号	DCG	NDCG
A_1	3.911	93.84
A_2	3.706	88.92
A_3	3.772	90.51
A_4	4.134	99.21
A_5	4.147	99.51

由表 6 可知，上述方法准确率都比较高，原因是所有变体流程分支权重的差异变化比较小，但是仍可以看出结合模型结构和行为信息的算法（A_4 和 A_5）要比仅考虑单一因素的算法准确度要高。在实际应用中，用户更关心较为靠前的结果，分析表 5 可得，本文方法前 7 个推荐结果和专家的排序结果一致，说明本文方法更加符合专家的判定，准确率相比算法 A_4 更高。

3.4 复杂度分析

对于企业而言，用户关心的不仅仅是流程推荐的准确度，还要求流程推荐具有高效性，从而节省公司的支出。本文方法与其他算法的时间复杂度如表 7 所示。

表 7 时间复杂度对比

算法	A_1	A_2	A_3	A_4	A_5
时间复杂度	$O(n^2)$	$O(n^3)$	$O(n^3)$	$O(n^4)$	$O(n^2)$

本文方法和算法 A_1 都是通过将流程模型转化为邻接矩阵，求矩阵间的距离得到流程相似度，时间复杂度均为 $O(n^2)$，但是本文方法能区别更多不同结构的流程且准确率更高。算法 A_2～A_4 都是基于图的编辑距离计算流程相似度。图编辑距离指一个图变形到另一个图所需要的最小消耗。其中变形由一系列的变形操作完成，包括节点的插入/删除操作、节点的替换操作、边的插入/删除操作和边的替换操作，而每个操作均有一定的时间消耗。因此算法的时间复杂度非常高，通常与涉及到的两个图的节点数目呈指数关系，这样的时间消耗在业务流程库增加到更大规模时，检索时间也将成倍增加，严重影响了大规模流程检索的用户体验，也在一定程度上阻碍了企业业务的发展。综上，本文的算法在准确度及时间效率上都具有非常大的优势。

3.5 实验说明

本文的实验环境是：Intel（R）Core（TM）i5－4460 CPU @ 3.20 GHz 8 GB 内存。实验程序用 Python 实现，版本是 3.7.0，运行在 64 位的 Windows 10 系统上。实验数据中 14 个流程的事件日志，是使用事件日志生成工具（PLG）生成的，数据集已上传到 GitHub，地址为 https://github.com/swustzzh/static/tree/master/process。

4 结束语

现有的大部分流程相似度计算方法主要根据流程模型结构计算相似度，或者根据事件日志中的行为信息计算相似度。本文尝试以模型结构信息为主，融入一些日志行为信息的方法计算流程相似度，从而能够更好的表示企业业务流程的行为习惯，更加贴合企业的需求。首先根据流程模型结构中的紧邻活动关系将其转化为邻接矩阵，然后通过事件日志的轨迹执行频次的对紧邻活动进行加权。最后定义了加权邻接矩阵的同维操作，并在此基础上给出矩阵间距离的计算公式。实验结果表明本文的算法能够有效的结合流程模型结构和行为日志，相比周长红等人的算法，计算更简便，时间复杂度更低。本文提出的方法有利于帮助企业推荐相似的流程，可以在大规模的流程库中进行检索。下一步工作将会进一步优化流程间距离算法，使其能够更好地区分差异较小的流程模型。

参考文献：

[1] Thaler T, Schoknecht A, Fettke P, et al. A comparative analysis of business process model similarity measures [A]. International Conference on Business Process Management [C]. Springer, Cham, 2016: 310－322.

[2] Rahmawati D, Aini L, Sarno R, et al. Comparison of behavioral similarity use TARs and naive algorithm for calculating similarity in business process model [A]. International Conference Science in Information Technology [C]. Bandung, 2017: 115－120.

[3] Shafer S, Smith H, Linder J. The power of business models [J]. Business Horizons, 2005, 48 (3): 199－207.

[4] Han V, Henrik L, Hajo A. Detecting inconsistencies between process models and textual descriptions [A]. International Conference on Business Process Management [C]. 2016: 90－105.

[5] Riefer M, Ternis S, Thaler T. Mining process models from natural language text: a state－of－the－art analysis [A]. Multikonferenz Wirtschaftsinformatik [C]. Illmenau, Germany, 2016.

[6] Zha H, Wang J, Wen L, et al. A workflow net similarity measure based on transition adjacency relations [J]. Computers in Industry, 2010, 61 (5): 463－471.

[7] Ma Y, Zhang X, Lu K. A graph distance based metric for data oriented workflow retrieval with variable time constraints [J]. Expert Systems with Applications, 2014, 41 (4): 1377－1388.

[8] 殷 明，闻立杰，王建民，等. 基于变迁紧邻关系重要性的流

程相似性算法 [J]. 计算机集成制造系统，2015，21 (2)：344－358.
[9] 胡　华，乔　静，胡海洋. 基于概率时间 Petri 网的流程推荐方法 [J]. 计算机应用研究，2018，35 (1)：62－68.
[10] 吴亚锋，谭文安. 基于邻接矩阵的业务流程间距离计算方法 [J]. 计算机工程，2018，44 (4)：52－58.
[11] Kunze M，Weske M. Metric trees for efficient similarity search in large process model repositories [J]. Lecture Notes in Business Information Processing，2010，66：535－546.
[12] 贾　楠，付晓东，黄　袁，等. 基于树编辑距离的工作流距离度量方法 [J]. 计算机应用，2012，32 (12)：3529－3533.
[13] Dijkman R，Dumas M，Garcia L. Graph matching algorithms for business process model similarity search [J]. Lecture Notes in Computer Science，2009，5701：48－63.
[14] Yan Z，Dijkman R，Grefen P. Fast business process similarity search with feature－based similarity estimation [M]. On the Move to Meaningful Internet Systems，2010.
[15] Gao X，Xiao B，Tao D，et al. A survey of graph edit distance [J]. Pattern Analysis and Applications，2010，13 (1)：113－129.
[16] 徐周波，张　鹃，宁黎华，等. 图编辑距离概述 [J]. 计算机科学，2018，45 (4)：11－18.
[17] Nejati S，Sabetzadeh M，Chechik M，et al. Matching and merging of Statecharts specifications [A]. International Conference on Software Engineering [C]. IEEE Computer Society，2007.
[18] Ehrig M，Koschmider A，Oberweis A. Measuring similarity between semantic business process models [A]. Conference on Conceptual Modelling. DBLP [C]. 2007.
[19] Dijkman R，Dumas M，Dongen B，et al. Similarity of business process models：metrics and evaluation [J]. Information Systems，2011，36 (2)：498－516.
[20] 周长红，曾庆田，刘　聪，等. 基于模型结构与日志行为的流程相似度计算 [J]. 计算机集成制造系统，2018，24 (7)：1793－1805.
[21] Zhou C，Liu C，Zeng Q，et al. A comprehensive process similarity measure based on models and logs [J]. IEEE Access，2019 (7)：69257－69273.
[22] 董子禾，闻立杰，黄浩未，等. 基于触发序列集合的过程模型行为相似性算法 [J]. 软件学报，2015，26 (3)：449－459.
[23] 曹　斌，王佳星，安卫士，等. ProBench：一种评估流程相似性查询算法的基准数据集 [J]. 计算机集成制造系统，2017，23 (5)：1069－1079.

(上接第 217 页)

生物的智能优化算法——北极熊算法，基本按照原文献的描述还原了算法的背景、生物特性、局部搜索算子、全局搜索算法和种群控制算子。通过详细介绍算法的执行步骤，为领域内的相关学者提供了有价值的参考和借鉴。文章后半部分描述了算法的应用领域，呈现了算法在实际生产生活中的重要作用。目前算法尚处于研究初期，可以围绕北极熊算法在其他工程领域、混合算法领域（北极熊算法混合帝国竞争算法和狮群算法等[14-15]）和多目标问题研究领域进行进一步的深入探讨和挖掘。

参考文献：

[1] David P，Marcin W. Polar bear optimization algorithm：Meta－heuristic with fast population movement and dynamic birth and death mechanism [J]. Symmetry，2017，9 (10)：203－222.
[2] Marcin W，Kamil K，Jakub M，et al. Heat production optimization using bio－inspired algorithms [J]. Engineering Application of Artificial Intelligence，2018，76 (11)：185－201.
[3] Van Laarhoven P J，Aarts E H. Simulated Annealing：Theory and Applications [M]. Berlin：Springer，Germany，1987：7－15.
[4] Holland J H. Genetic algorithms [J]. Scientific American，1992，267 (1)：66－73.
[5] Kennedy J，Eberhart R. Particle swarm optimization [A]. Proceedings of the IEEE International Conference on Neural Networks [C]. Perth，Australia，1995：1942－1948.
[6] Toksari M D. Ant colony optimization for finding the global minimum [J]. Applied Mathematics and Computation，2006，176 (1)：308－316.
[7] Yang X S，Deb S. Cuckoo search via levy flights [A]. Proceedings of the World Conference on Nature & Biologically Inspired Computing [C]. Coimbatore，India，2009，Piscataway：IEEE，USA，2009：210－214.
[8] Yang X S. A new metaheuristic bat－inspired algorithm [A]. Nature Inspired Cooperative Strategy for Optimization [C]. Berlin：Springer，Germany，2010：65－74.
[9] Yang X S. Firefly algorithm，stochastic test functions and design optimisation [J]. International Journal of Bio－inspired Computing. 2010，2 (2)：78－84.
[10] Yang X S. Flower pollination algorithm for global optimization [A]. Proceedings of the International Conference on Unconditional Computing and Natural Computation [C]. Orleans，France，2012，Berlin：Germany，Springer，2012：240－249.
[11] Zheng Y J. Water wave optimization：A new nature－inspired metaheuristic [J]. Computers and Operations Research，2015，55 (3)：1－11.
[12] Mirjalili S. Moth－flame optimization algorithm：A novel nature－inspired heuristic paradigm [J]. Knowlege－based systems，2015，89 (3)：228－249.
[13] Mirjalili S. Dragonfly algorithm：A new meta－heuristic optimization technique for solving single－objective，discrete，and multi－objective problems [J]. Neural Computing and Applications，2016，27 (4)：1053－1073.
[14] Atashpaz E，Lucas C. Imperialist competitive algorithm：an algorithm for optimization inspired by imperialist competition [A]. IEEE Congress on Evolutionary Computation [C]. 2007，21 (1)：4661－4667.
[15] Rajakumar B R. The Lion’s Algorithm：A new nature－inspired search algorithm [C]. Procedia Technology，2012，6：126－135.