基于智能规划的工作流任务识别算法

曾 霖[1,2],卓汉逵[1],李 磊[1]
(1. 中山大学数据科学与计算机学院,广东广州 510006;2. 贺州学院数学与计算机学院,广西贺州 542899)

摘 要: 针对传统算法将活动看成是彼此之间相互独立的事件,无法准确识别的问题,提出一种基于智能规划的工作流任务识别算法,利用工作流与智能规划在执行序列和操作规则方面存在的共性. 通过高层次的抽象描述,自动推导出活动之间的内在逻辑联系,且能从外部信息充分挖掘潜在的知识,将工作流任务识别问题转变为对应的智能规划识别问题来进行求解,有效地解决了传统算法识别困难和对噪声数据敏感的问题. 实验结果表明,本文提出的算法是可行的.

关键词: 工作流活动; 任务识别; 规划算法; 智能监控
中图分类号: TP311 **文献标识码:** A **文章编号:** 0372-2112 (2018)04-0871-07
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2018.04.015

Workflow Task Recognition Algorithm Based on Intelligent Planning

ZENG Lin[1,2], ZHUO Han-kui[1], LI Lei[1]
(1. *School of Data and Computer Science, Sun Yat-sen University, Guangzhou, Guangdong* 510006, *China*;
2. *School of Mathematics and Computer Science, Hezhou University, Hezhou, Guangxi* 542899, *China*)

Abstract: For traditional algorithms, activities are treated as events that are independent from each other and cannot be accurately identified, we propose a workflow task recognition algorithm based on intelligent programming, which utilizes the commonality of workflow and intelligent planning in the implementation of sequence and operation rules. Through the high-level abstract description, we can deduce the intrinsic logic relation between them automatically, and also explore potential knowledge from outside information, and transform the workflow task recognition problem into the corresponding planning recognition problem, and solve the problem that the traditional algorithm is difficult in recognition and sensitive to the noise data. Experimental results show that our algorithm is feasible.

Key words: workflow activity; activity recognition; planning; intelligent monitoring

1 引言

随着社会和科技的不断进步,工业生产正朝着自动化、智能化的方向快速发展. 工作流相关技术的研究也越来越受到人们的重视[1],并被广泛地应用于电子业务[2]、制造业[3]、软件开发[4]、生物医学[5]和银行业务[6]等各个领域. 工作流不但能够自动化处理相关的活动或任务,减少人机交互处理过程中带来的潜在错误,而且能够精确化每一处理步骤,最大化地提高企业的生产效率. 这就要求工作流能够支持动态可变的、灵活的应用场景[7].

由于安全生产、产品质量及其成本控制等方面的需求,工作流任务识别已成为一个重要的研究问题. 例如制造业生产装配线上的智能监控、医院病人护理和手术流程中的自动管理等方面. 近年来,在大数据、"互联网+"和工业自动化背景下,工作流中的业务流程日趋复杂多变. 由需求分析引起的业务过程重新建模或由维护升级引起的过程模式变更和改进,也将变得越来越频繁. 如何在这样动态多变的复杂环境下快速准确地识别出任务,就成为目前工作流任务识别研究亟需解决的关键问题.

当前对工作流任务识别的研究方法都要依赖于精确的流程模式,且将活动或任务看成是相互独立的事件,而在现实世界中,活动与活动之间通常是存在逻辑

收稿日期:2016-12-06;修回日期:2017-04-18;责任编辑:李勇锋
基金项目:国家自然科学基金(No. 61309011, No. U1611262);广州市珠江科技新星;广东省大数据分析与处理重点实验室

因果联系的. 值得注意的是,这里的活动指的是完成工作流任务的具体操作步骤,可以逻辑表示成智能规划中对应的动作. 另外,现代企业组织的业务流程常发生变动,导致工作流模式也随之重新设计和实现[8];且当活动的种类和类型繁多时,即使是在工作流模式确定的情况下,活动之间组成的排列组合数量级别也是非常庞大的,因此很容易陷入"任务识别大爆炸"泥潭中. 如果再加上背景遮挡、相似环境等复杂情况的影响,传统识别算法将无法有效地对所有活动或任务进行识别.

本文提出了一种基于智能规划的工作流任务识别算法(Workflow Task Recognition,WTR),利用工作流与智能规划在执行序列和操作规则方面存在的共性,将工作流任务识别问题转变成智能规划识别问题. WTR 算法从工作流实例库和外部信息中挖掘出活动之间潜在的逻辑关系,只需根据部分已观察到的状态信息,同时结合问题结构模型,以及当前隐动作等相关辅助标注信息,就能推出其他未能观察到的隐动作,从而得到任务识别问题的完整动作序列及其目标条件,有效地解决了传统算法识别困难和对噪声数据敏感等问题. 实验结果表明,本文提出的算法是可行有效的.

2 相关工作

目前国内外学者对工作流任务识别采用的主要研究方法有聚类分析、密度估计、目标轨迹法、特征提取技术和隐马尔可夫模型(Hidden Markov Models,HMM)等. 例如 Yan 等人提出了一种对日常生活活动中的视觉数据进行分析的多任务聚类框架[9]. Boiman 和 Irani 从过去视频实例中提取出一些特定的区域,再聚类产生出新的观察部分,解决了视觉数据中的不规则问题[10]. Khalid 等人针对行为识别和异常检测,提出了一种监督特征提取和多变量建模方法,且利用分层索引技术来提高分类器的效率[11]. Geib 和 Goldman 提出一种基于树结构解析的识别算法[12]. Avrahami 等人提出一个自动生成匹配决策树的方法,能将多特征观察结果匹配到规划库中的动作[13]. Ramlrez 和 Geffner 提出一种可直接利用规划器求解的规划识别算法[14].

3 问题描述及其智能规划定义

对于一个长度为 T 的工作流任务识别问题实例 $\mathcal{L}_i = \{(a_t, s_t, y_t)_1^T\}_i$,其中 $t(1 \leqslant t \leqslant T)$ 表示动作执行的时刻点,a_t,s_t 和 y_t 分别表示业务流程未知的隐动作、观察到的状态和隐动作辅助标注信息. a_t 与 s_t 密切相关,y_t 是传感器等检测设备间接或局部感知到与 a_t 相关的输出结果. y_t 的值可分为缺失和有值两种情况,其中缺失表示感知设备未能输出 a_t 的结果,有值表示能得到 a_t 的结果,但不保证其是正确的. 特别地,用 α_0 表示辅助信息 $y = <y_1, y_2, \cdots, y_T>$ 中未能标注到的隐动作.

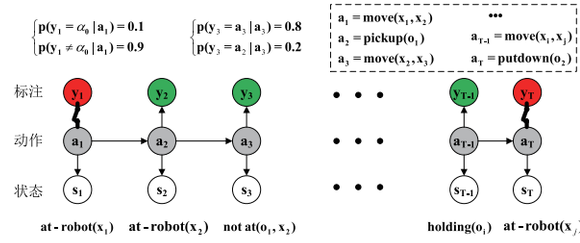

图1 工作流任务识别问题实例

如图 1 所示,红色、绿色的圆圈分别表示标注 y 缺失和有值的两种情况,图 1 右上角表示实例中的动作集 $\{(a_t)_1^T\}$. 在实例 $\mathcal{L}_i$ 中,存在一个机器人(robot),其可执行的动作有移动(move)、拿起(pickup)和放下(putdown)等,则 t_1 时刻点的可观察状态 s_1 = at-robot(x_1)表示机器人在地点 x_1 的位置,s_1 对应的隐动作 a_1 = move(x_1,x_2)表示机器人从地点 x_1 移动到 x_2. t_2 时刻点的 s_2 = at-robot(x_2)表示机器人在地点 x_2 上,s_2 对应的隐动作 a_2 = pickup(o_1)表示机器人在地点 x_2 上拿起物体 o_1. 类似地,可以得到其他时刻点 t_i 中 a_i、y_i 和 s_i 的情况. 从图 1 中不难看出,获取到标注 y_1 值的概率为 $p(y_1 \neq \alpha_0 | a_1) = 0.9$,缺失的概率为 $p(y_1 = \alpha_0 | a_1) = 0.1$. 注意,这里标注 y_1、y_T 的值是缺失的,表示未能感知到任何信息. 标注 y_3 能够得到结果,且正确、错误判断 a_3 的概率分别为 0.8 和 0.2. 因此标注 y_3 可能会错误判断 a_3 为 a_2.

在工作流实例 $\mathcal{L}_i$ 中,(a_t, s_t, y_t) 会存在未知的随机变量,特别指出本文中 $\mathcal{L}_i$ 指的都是变量 a_t 未知的情况. 如果 $\mathcal{L}_i$ 可以分成若干长度不等的小段,且可能缺少其中的部分小段,则我们称之为不完整实例,记为 $\hat{\mathcal{L}}_i$,且 $|\hat{\mathcal{L}}_i| < |\mathcal{L}_i|$. 可以进一步将 $\hat{\mathcal{L}}_i$ 表示成 n 个小段 $\hat{\mathcal{L}}_i = \{\hat{\mathcal{L}}_{i1}, \hat{\mathcal{L}}_{i2}, \cdots, \hat{\mathcal{L}}_{in}\}$. 实例库 $\mathcal{L} = \{\mathcal{L}_1, \mathcal{L}_2, \cdots, \mathcal{L}_n\}$ 是由 n 个完整实例 $\mathcal{L}_i$ 组成的. 在实际业务流程中,很容易获取到一系列可观察的状态 s_t 和关于隐动作 a_t 的辅助标注信息 y_t. 这里,我们采用智能规划领域中的定义语言 PDDL 对工作流任务识别问题进行正式的定义和描述:

定义 1 一个工作流任务识别问题可以表示为四元组形式 $\Gamma = <P, \mathcal{O}, \mathcal{G}, \mathcal{L}>$,其中 $P = <F, I, \mathcal{G}, \mathcal{A}>$ 是工作流问题的领域,F 是业务流程中所有的有限状态集合,$I \subseteq F$ 是业务流程的初始状态,$G \subseteq F$ 是要识别任务的目标条件,$\mathcal{G}$ 是候选目标集,即 $G \in \mathcal{G}$,$\mathcal{A} = \{\alpha_i\}_{i=0}^{N_a}$ 是执行具体任务的动作集,$\hat{\mathcal{L}}_i$ 是不完整观察实例,$\mathcal{O} = (o_1, \cdots, o_m)$ 是 $\hat{\mathcal{L}}_i$ 中部分标注的隐动作序列,$o_i \in \mathcal{A}$,$i \in [1, m]$,$\mathcal{L}$ 是给定的实例库.

在满足正确标注隐动作序列 $\mathcal{O}$ 的前提下,工作流

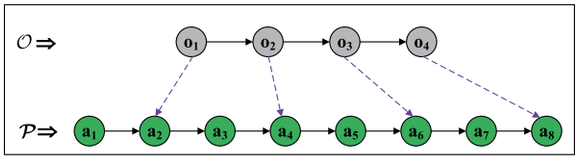

图 2　隐动作序列$\mathcal{O}$在规划解$\mathcal{P}$中的对应位置

任务识别问题 Γ 的规划解 $\mathcal{P}[G]$是一个动作序列 $\vec{a}=<a_1,\cdots,a_n>$. 执行这一动作序列 $\vec{a}$,能够使问题 Γ 从初始状态 I 达到目标条件 G. 如果 $\vec{a}$ 满足 $\mathcal{O}$,则它们存在这样的一个单调映射函数 f,使得正确标注隐动作 o_j 在 $\mathcal{O}$ 中的位置与动作序列 $\vec{a}$ 中的位置一一对应,即 $a_{f(j)}=o_j, j\in[1,n]$. 比如在图 2 中,隐动作序列 $\mathcal{O}$ 中的 o_1、o_2、o_3、o_4 分别对应规划解 $\mathcal{P}$ 中的 a_2、a_4、a_6 和 a_8.

4　工作流任务识别算法(WTR)

4.1　规划问题的结构模型

给定一个长度为 T 的问题实例 $\mathcal{L}_i$,观察状态 $\vec{s}=<s_1,s_2,\cdots,s_T>\in\mathcal{L}_i$ 是一个可观察状态序列,$\vec{a}=<a_1,a_2,\cdots,a_T>\in\mathcal{L}_i$ 是其对应的隐动作序列,$\vec{y}=<y_1,y_2,\cdots,y_T>$ 是 $\vec{a}$ 中对应部分的、含噪声的辅助标注信息,具体对应关系如图 1 所示. 如果工作流中 t 时刻点的隐动作 a_t 具有马尔可夫的特性,即 $p(a_t|a_1^{t-1})=p(a_t|a_{t-1})$ $(1\leqslant t\leqslant T)$,则在前面 $t-1$ 时刻点条件下,产生状态 s_t 的概率等于在 a_t 条件下产生 s_t 的概率,即 $p(s_t|a_1^{t-1},s_1^{t-1})=p(s_t|a_t)$. 当 a_t 被标注,但是出现标注错误的情况,则 $p(y_t\neq a_t|a_t)=1-p(y_t=a_t|a_t)=1-\eta$,其中参数 η 表示隐动作 a_t 被正确标注的概率. 特别地,用参数 ζ 表示获取到辅助标注信息 y_t 的概率,则当参数 $\zeta=0$ 时,表示当前时刻点 t 未检测到任何与隐动作 a_t 相关的标注信息. 如果已知隐动作 a_t 的标注是错误的,则 a_t 被标注成 a_j 的概率为 $p(y_t=a_j|a_t)=1/(N_\alpha-1)$,其中 N_α 表示隐动作的个数.

构建问题结构模型 $\mathcal{M}(\theta,\Delta)$ 的具体过程如下:首先初始化参数 $\theta=<\pi,\Lambda,\Gamma>$ 和 Δ. θ 的初始值要符合概率统计模型,π 是初始观察状态的概率,可以简单初始化为 $\pi_i=1/N_\alpha$. Λ、Γ 分别表示初始的动作—状态转移概率矩阵和动作—动作转移概率矩阵,其对应的转移函数分别表示成 $\psi_{jk}=p(s_t=\xi_k|a_t=\alpha_j)$ $\chi_{ij}=p(a_t=\alpha_j|a_{t-1}=\alpha_i)$. 其中 ψ_{jk} 表示隐动作 α_j 出现时观察到状态 ξ_k 的概率,χ_{ij} 表示隐动作 α_i 在时刻点 $t-1$ 出现时,在下一时刻点 t 动作 α_j 出现的概率,Δ 表示工作流规划领域知识. 然后通过对问题实例库 $\mathcal{L}$ 中的每一实例进行训练和学习,利用最大似然估计计算出 $\mathcal{M}(\theta,\Delta)$ 的具体参数 θ,即计算 $\arg\max_\theta \log(p(\vec{s},\vec{y}|\theta))$ 的值. 这一过程需要通过 EM 步骤进行不断迭代更新来实现. 值得注意的是,在工作流实例库 $\mathcal{L}$ 中,由于事先并不完全知道 $\mathcal{L}$ 中的隐动作序列 $\vec{a}$,因此需要通过实例来计算出 $\log(p(\vec{s},\vec{y}|\theta))$ 的最大值,即计算 $\arg\max_\theta \log(p(\vec{s},\vec{y}|\theta))$. 但 $\log(p(\vec{s},\vec{y}|\theta))$ 值也无法直接计算,这里,我们通过下面不等式(1)中的 $E_{\vec{a}|\vec{s},\vec{y};\theta}[\log(p(\vec{a},\vec{s},\vec{y}|\theta))]$ 来间接计算其最大似然估计值,且在当前最大值的基础上不断调整和更新参数 θ.

$$\begin{aligned}\log(p(\vec{s},\vec{y}|\theta))&\geqslant\sum_{\vec{a}}Q(\vec{a};\theta)\log\left(\frac{p(\vec{a},\vec{s},\vec{y}|\theta)}{Q(\vec{a};\theta)}\right)\\&=E_{\vec{a}|\vec{s},\vec{y};\theta}[\log(p(\vec{a},\vec{s},\vec{y}|\theta))]\end{aligned}\tag{1}$$

式(1)中的 $Q(\vec{a};\theta)=p(\vec{a}|\vec{s},\vec{y};\theta)$,这个值可通过 EM 算法中的 E 步骤来求解. 那么计算 $\log(p(\vec{s},\vec{y}|\theta))$ 的最大值,只需计算 $\log(p(\vec{a},\vec{s},\vec{y}|\theta))$ 关于 $\vec{a}$ 最大化期望值. 利用条件概率公式逐步分解出式(1)中的 $E_{\vec{a}|\vec{s},\vec{y};\theta}[\log(p(\vec{a},\vec{s},\vec{y}|\theta))]$,可得:

$$\begin{aligned}&\arg\max_\theta E_{\vec{a}|\vec{s},\vec{y};\theta}[\log(p(\vec{a},\vec{s},\vec{y}|\theta)]\\&=\arg\max_\theta\sum_{\vec{a}}Q(\vec{a};\theta)\log(p(\vec{s},\vec{a},\vec{y}|\theta))\\&=\arg\max_\theta\sum_{\vec{a}}Q(\vec{a};\theta)\log(p(\vec{s}|\vec{a},y,\theta))p(\vec{y}|\vec{a},\theta)p(\vec{a}|\theta)\end{aligned}\tag{2}$$

上式(2)中的 $\vec{s}$ 在条件$(\vec{a},\theta)$下独立于 $\vec{y}$,而 $\vec{y}$ 在条件 $\vec{a}$ 下独立于 θ. 因此,可继续得到如式(3):

$$\begin{aligned}&\arg\max_\theta E_{\vec{a}|\vec{s},\vec{y};\theta}[\log(p(\vec{a},\vec{s},\vec{y}|\theta)]\\&=\arg\max_\theta\sum_{\vec{a}}Q(\vec{a};\theta)\log(p(\vec{s}|\vec{a},\theta))p(\vec{a}|\theta)\end{aligned}\tag{3}$$

值得注意的是,在最大化式(2)时,除了 $Q(\vec{a};\theta)$ 需要 $\vec{y}$ 外,其它因子的计算并未涉及到 $\vec{y}$. 由于 $Q(\vec{a};\theta)$ 的值可通过 EM 算法中的 E 步骤计算获得,因此在式(2)中,其为常量,且在后续 EM 算法的 M 步骤中迭代计算和更新参数 θ 时也不涉及到 $\vec{y}$. 一旦在迭代计算和更新过程中需要确定参数 θ 值,就可以根据拉格朗日乘子法得到 χ_{ij} 和 ψ_{jk} 的如下公式:

$$\begin{aligned}\chi_{ij}&=\frac{\sum_{\vec{a}}Q(\vec{a};\theta)\sum_{t=1}^{T-1}\ell(a_t=\alpha_i\wedge a_{t+1}=\alpha_j)}{\sum_{\vec{a}}Q(\vec{a};\theta)\sum_{t=1}^{T-1}\ell(a_t=\alpha_i)}\\&=\frac{\sum_{t=1}^{T-1}p(a_t=\alpha_i,a_{t+1}=\alpha_j|\vec{s},\vec{y};\theta)}{\sum_{t=1}^{T-1}p(a_t=\alpha_i|\vec{s},\vec{y};\theta)}\end{aligned}\tag{4}$$

$$\begin{aligned}\psi_{jk}&=\frac{\sum_{\vec{a}}Q(\vec{a};\theta)\sum_{t=1}^{T}\ell(a_t=\alpha_j\wedge s_t=\xi_k)}{\sum_{\vec{a}}Q(\vec{a};\theta)\sum_{t=1}^{T}\ell(a_t=\alpha_j)}\\&=\frac{\sum_{t=1}^{T}p(a_t=\alpha_j,s_t=\xi_k|\vec{s},\vec{y};\theta)}{\sum_{t=1}^{T}p(a_t=\alpha_j|\vec{s},\vec{y};\theta)}\end{aligned}\tag{5}$$

上式的χ_{ij}和ψ_{jk}在求解过程中涉及大量的重复性计算,因此我们采用向前和向后递归的方法进行优化. 用$\varpi_t(i,j)=p(a_t=\alpha_i,a_{t+1}=\alpha_j|\vec{s},\vec{y};\theta)$表示在给定$\vec{s},\vec{y}$和参数$\theta$的前提下,隐动作从$t$时刻点$\alpha_i$转变成$t+1$时刻点$\alpha_j$时的概率,用$\zeta_t(i)=p(a_t=\alpha_i|\vec{s},\theta)$表示在给定$\vec{s}$和参数$\theta$的前提下,隐动作$a_t$在$t$时刻点为$\alpha_i$的概率. 因此式(4)、(5)可以进一步表示成如下的形式:

$$\chi_{ij}=\frac{\sum_{t=1}^{T-1}\varpi_t(i,j)}{\sum_{t=1}^{T-1}\zeta_t(i)}\quad \psi_{jk}=\frac{\sum_{t=1,s_t=\xi_k}^{T}\zeta_t(j)}{\sum_{t=1}^{T}\zeta_t(j)} \tag{6}$$

另外,用$\mu_t(i)=p(s_1^t,y_1^t,a_t=\alpha_i|\theta)$表示$s_1^t$和$y_1^t$出现的同时,隐动作$a_t$在$t$时刻点为$\alpha_i$的概率,用$\nu_t(i)=p(s_{t+1}^T,y_{t+1}^T|a_t=\alpha_i;\theta)$表示隐动作$a_t$在$t$时刻点为$\alpha_i$的情况下$s_{t+1}^T$和$y_{t+1}^T$出现的概率,那么$\mu_t(i)$和$\nu_t(i)$可以用下面的命题1中的递推公式给出,$\varpi_t(i,j)$由命题2给出.

命题1 在进行迭代计算时,公式$\mu_t(i)$和$\nu_t(i)$具有如下的递推形式:

$$\mu_t(i)=\hbar(y_t,\alpha_i)\psi_{is_t}\sum_{i=1}^{N_\alpha}\chi_{ji}\mu_{t-1}(j),2\leqslant t\leqslant T \tag{7}$$

$$\nu_t(i)=\sum_{j=1}^{N_\alpha}\hbar(y_{t+1},\alpha_j)\nu_{t+1}(j)\chi_{ij}\psi_{js_{t+1}},1\leqslant t\leqslant T-1 \tag{8}$$

其中$\hbar(y_t,\alpha_i)=\ell(y_t=\alpha_0)(1-\zeta)+\ell(y_t=\alpha_i)\zeta\eta+\ell(y_t\neq\alpha_i\wedge y_t\neq\alpha_0)\zeta(1-\eta)/(N_\alpha-1)$,$\ell(x)$是指示函数,当$x$满足时,它的值为1,反之为0. 另外,将其分别初始化为$\mu_1(i)=\hbar(y_1,\alpha_i)\pi_i\psi_{is_1}$和$\nu_T(i)=1$.

命题2 假定已经给定了命题1中公式$\mu_t(i)$和$\nu_t(i)$的递推形式,则$\varpi_t(i,j)$的值可以用如下公式计算:

$$\begin{aligned}\varpi_t(i,j)&=p(a_t=\alpha_i,a_{t+1}=\alpha_j|s,y,\theta)\\&=\frac{\hbar(y_{t+1},\alpha_j)\psi_{js_{t+1}}\chi_{ij}\mu_t(i)\nu_{t+1}(j)}{p(\vec{s},\vec{y}|\theta)}\end{aligned} \tag{9}$$

其中$\hbar(y_t,\alpha_i)$的定义同前. $\ell(x)$是指示函数,当x满足时,它的值为1,反之为0.

另外,用下式(10)表示观察状态序列$\vec{s}$和动作a_t在t时刻点为α_i同时出现的概率:

$$p(\vec{s},a_t=\alpha_i|\theta)=\mu_t(i)\nu_t(i) \tag{10}$$

还可进一步得到$\vec{s}$出现的概率$p(\vec{s}|\theta)$公式:

$$p(\vec{s}|\theta)=\sum_{i=1}^{N_\alpha}p(\vec{s},a_t=\alpha_i|\theta)=\sum_{i=1}^{N_\alpha}\mu_t(i)\nu_t(i) \tag{11}$$

如前所述,$\zeta(i)$表示在$\vec{s},\vec{y}$前提下,动作a_t在t时刻点为α_i的概率,因此可用下式(12)表示$\zeta(i)$:

$$\zeta_t(i)=p(a_t=\alpha_i|\vec{s},\vec{y},\theta)=\frac{\mu_t(i)\nu_t(i)}{\sum_{i=1}^{N_\alpha}\mu_t(i)\nu_t(i)} \tag{12}$$

根据命题1、2和相关公式推导,最后可以用下式(13)、(14)计算和更新$\mathcal{M}(\theta,\mathcal{D})$的参数$\theta$:

$$\hat{\chi}_{ij}=\frac{\sum_{t=1}^{T-1}\varpi_t(i,j)}{\sum_{t=1}^{T-1}\zeta_t(i)},1\leqslant i\leqslant N_\alpha,1\leqslant j\leqslant N_\alpha \tag{13}$$

$$\hat{\psi}_{jk}=\frac{\sum_{t=1,s_t=\xi_k}^{T}\zeta_t(j)}{\sum_{t=1}^{T}\zeta_t(j)},1\leqslant j\leqslant N_\alpha,1\leqslant k\leqslant N_s \tag{14}$$

4.2 工作流活动的表示及其转换规则

工作流模型所产生的一个业务流程,主要是由一系列被执行的任务及决定其次序的前提条件组成. 如果用STRIPS模型形式化描述和定义业务流程,则一个业务问题可以用$P=<F,I,G,\mathcal{A}>$表示,其中F是业务流程中的有限状态集合,$I\subseteq F$是业务流程的初始状态,$G\subseteq F$是业务流程结束时的目标条件,$a=<pre(a),add(a),del(a)>\in\mathcal{A}$三元组形式表示完成这些任务的具体活动,$pre(a)$、$add(a)$和$del(a)$分别表示活动被执行时必须满足的前提条件、增加和删除效果. 特别地,引入一个与活动执行后密切相关的状态s,得到与之相关的状态转移函数$\gamma:F\times\mathcal{A}\rightarrow F$. 值得注意的是,在构建工作流的动作模型时,要满足动作两个最基本规则约束,即$pre(a)\cap add(a)=\varnothing$和$del(a)\subseteq pre(a)$.

利用规划算法来求解任务识别问题,需进一步保证$\mathcal{O}=<o_1,\cdots,o_m>$中的动作$o_i$出现在$\mathcal{P}[G]$的对应位置. 因此,需将$\mathcal{O}$中动作之间序列信息直接转并融合到问题的求解过程中,并将这一转换规则方法称之为generate_new_plan(). 具体如下:将原问题$\Gamma=<P,\mathcal{G},\mathcal{O},\mathcal{L}>$转换成新问题$\Gamma'=<P',\mathcal{G}',\mathcal{O}',\mathcal{L}>$,其中$P'$中命题集$F'=F\cup F_o$,$F_o=\{f_a|a\in\mathcal{O}\}$,动作集$\mathcal{A}'=\mathcal{A}\cup\mathcal{A}_o$,新增动作集$\mathcal{A}_o=\{o_a|a\in\mathcal{O}\}$,目标条件$G'=G\cup F_o$. 在$\mathcal{O}$中,如果动作$a$的直接前驱动作是$b$,则在$a$的前提条件$pre(a)$中新增一个命题$f_b\in add(o_b)$.

4.3 工作流任务识别算法实现

算法1是基于智能规划的工作流任务识别算法框架. 首先初始化$\mathcal{M}(\theta,\mathcal{D})$中的相关参数,其中参数$\theta$中的值需满足概率统计的分布条件(步骤1). 然后逐步扫描实例库$\mathcal{L}$中的每行数据,根据实例库中已知或观察到的辅助标注隐动作信息,用EM算法迭代计算和更新参数θ(步骤2~7),其具体过程是E步骤根据每个可能出现的隐动作序列$\vec{a}$确定其$Q(\vec{a};\theta)=p(\vec{a}|\vec{s},\vec{y};\theta)$值

(步骤 4);接着 M 步在此基础上计算和更新 θ 参数值(步骤 5),并使得 $E_{\vec{a}|\hat{s},\hat{y};\theta}[\log(p(\vec{a},\vec{s},\vec{y}|\theta))]$ 的值最大化(步骤 6),重复执行这一过程,直到其值不再收敛为止. 其次通过修改 ARMS 算法[15],并利用 $\mathcal{M}(\theta,\mathbb{A})$ 计算结果和规则约束来获取动作模型$\widehat{\mathbb{A}}$(步骤 9). 根据问题实例$\widehat{\mathcal{L}_i}$ 关于动作的辅助标注信息,利用式 (15) 计算出其对应的隐动作序列 $\mathcal{O}=(o_1,\cdots,o_m)$(步骤 11). 最后利用 FF(Fast-Forward)[16] 规划器求解出问题的解,并返回最优结果(步骤 12 ~ 15).

$$\arg\max_i \mu_t(i) = \begin{cases} \arg\max_i \hbar(y_t,\alpha_i)\psi_{is_t}\sum_{i=1}^{N_a}\chi_{ji}\mu_{t-1}(j), \\ \qquad\qquad 2 \leqslant t \leqslant T; \\ \arg\max_i \hbar(y_1,\alpha_i)\pi_i\psi_{is_1}, t=1 \end{cases} \tag{15}$$

算法 1　工作流任务识别算法(WTR)

输入:工作流实例库 $\mathcal{L}$,观察实例$\widehat{\mathcal{L}_i}$,相关参数$(\theta,\mathbb{A})$.
输出:问题的规划解 $\mathcal{P}[G]$.
步骤:
1:　初始化相关参数$(\theta,\mathbb{A})\leftarrow(\theta^0,\mathbb{A}^0)$;
2:　for each $\mathcal{L}_i\in\mathcal{L}$ do
3:　　while ! done do
4:　　E 步骤确定 $Q(\vec{a};\theta)$;
5:　　M 步骤计算 $\arg\max_\theta E_{\vec{a}|\hat{s},\hat{y};\theta}[\log(p(\vec{a},\vec{s},\vec{y}|\theta))]$;
6:　　递推:利用式(13)、(14)迭代计算并更新$\hat{\chi}_{ij}$和$\hat{\psi}_{jk}$;
7:　　终止:得到模型参数$\hat{\theta}=<\hat{\pi},\hat{\chi}_{ij},\hat{\psi}_{jk}>$;
8:　　end while
9:　　根据 $\mathcal{M}(\theta,\mathbb{A})$ 计算结果和规则约束来修改动作模型$\widehat{\mathbb{A}}$;
10:　end for
11:　利用式(15)计算$\widehat{\mathcal{L}_i}$ 中的隐动作序列 $\mathcal{O}=(o_1,\cdots,o_m)$;
12:　产生出任务识别问题新规则过程 generate_new_plan();
13:　依次对 $\mathcal{G}$ 求解并能满足 $\mathcal{O}$ 的规划解;
14:　$\mathcal{P}[G]\leftarrow\mathcal{P}[G^*]$;
15:　return $\mathcal{P}[G]$

5　实验结果及分析

本文采用的工作流任务识别问题背景源自一汽车制造生产线上视频监控[17]. 我们用 Python 编写一个程序 PlanData,用来模拟仿真出工作流实例库 $\mathcal{L}$ 及其测试集$\widehat{\mathcal{L}}$. 采用文献[18]的图表示方式来设计出足够复杂的业务流程模型,首先从开始任务出发,随机选择一个目标条件结点;接着根据任务走向,以随机方式确定下一个任务节点,直到达到问题的目标条件结点,从而产生出一个完整的业务流程. 然后将具体执行任务的活动构建出其对应的隐动作 a、状态 s 以及辅助标注信息 y. 设置 $\mathcal{L}$ 和$\widehat{\mathcal{L}_i}$ 中的 y 具有隐动作信息的概率范围为 ζ $(0.6\leqslant\zeta\leqslant0.95)$,且具有一定噪声水平 $1-\eta(0.8\leqslant\eta\leqslant0.95)$. 最后将流程实例转换成对应规划问题 Γ 后,再利用规划器 FF 求解出规划解 $\mathcal{P}[G]$,并依此产生出 s 和 y 组成一个完整实例 $\mathcal{L}_i$. 假定已得到 Γ 的规划解 $\mathcal{P}[G]$,从 $\mathcal{P}[G]$ 中按其规划长度的比率 ξ 随机移除动作及其相关信息,得到不同的$\widehat{\mathcal{L}_i}$. 对 m 个不同问题重复这一过程,就可以产生出大小为 m 的测试集$\widehat{\mathcal{L}}$.

算法识别准确率定义成 $Acc(\widehat{\mathcal{L}})=\frac{1}{N}\sum_{i=1}^{N}\frac{K_i}{|\mathcal{O}_i|}$,其中 K_i、$|\mathcal{O}_i|$ 分别表示问题$\widehat{\mathcal{L}_i}$ 中正确识别的隐动作数和要识别的隐动作数,N 为测试集$\widehat{\mathcal{L}}$的大小. 本文所有实验硬件条件均在内存大小为 4G,CPU 为 Intel Core 2.0G,操作系统为 64 位的 Ubuntu 14.04 的计算机平台上运行,且算法单次求解时间限制在 2400 秒之内.

5.1　算法识别准确率比较

实验选择最新的识别算法(Plan Recognition as Planning,PRP)[14]作为算法比较基准. 由于 PRP 算法预先需要其规划领域完整的动作模型,因此我们根据文献[15]的算法思想,实现出能自动学习动作模型的识别算法 PRP_a^*. 为了验证 y 的作用,给定了理想条件下的 PRP_g 算法,即具有完整动作模型,且其观察的隐动作序列都是正确的. 另外,还实现了基于 HMM 模型的算法 PRP_h^* 作为算法性能另一比较基准,且设置不同 ζ 和 η 参数来验证算法的识别性能,具体实验结果和分析如下:

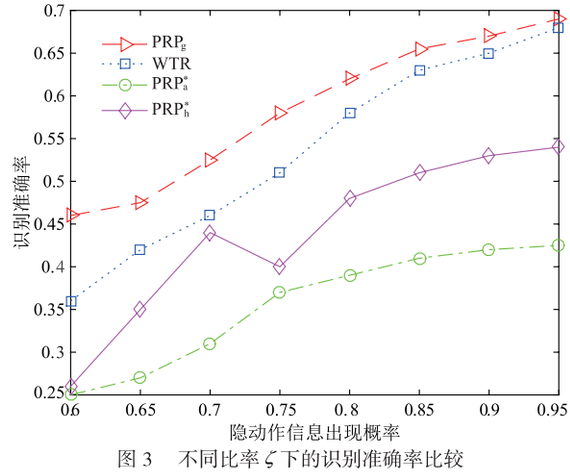

图 3　不同比率 ζ 下的识别准确率比较

首先通过固定 $\eta=0.95$,同时不断调整 ζ. 实验设置每一问题都有$|\mathcal{G}|=5$ 个可能的候选目标集. 实验结果如图 3 所示,不同算法的准确率都会随 ζ 增加而提高.

这主要原因是在辅助标注信息 y 比较可靠的情况下,算法的识别效果都依赖于隐动作这一关键信息. 获取到这类信息越多,算法的识别效果也就越明显. 当 y 中的 η 较小时,WTR 算法仍能保持较好的识别效果,而其他两种算法相比对噪声数据会比较敏感. 从图 3 中不难看出,WTR 算法识别准确率要明显好于其他的两种识别算法. 总的算法性能表现为 WTR 算法平均识别准确率要比算法 PRP_a^* 提高 53%,比算法 PRP_h^* 提高 22%. 通过与图 3 和图 4 中的 PRP_g 对比,可以发现 WTR 算法性能非常接近于理想条件下的 PRP_g. 这是因为 WTR 算法不仅将过去实例的隐含知识和动作模型的操作规则结合在一起,且在此基础上充分考虑了隐动作的辅助标注信息.

图 4 是在不同隐动作信息准确率 η 下的对比图. 通过固定参数 $\zeta=0.8$,同时不断调整 η 参数. 从图 4 可以看出,在提高 η 的情况下,与其他两种算法相比,WTR 算法的识别效果表现地同样明显. 具体表现为 WTR 算法平均识别准确率分别比 PRP_a^*、PRP_h^* 算法提高 42% 和 18%. 另外,我们还发现训练集和测试集的参数 ζ 和 η 选择也是有一定影响的,具体表现为:如果它们设置的值比较一致时,WTR 算法识别效果表现得最为理想. 因此,如何设置好参数 ζ 和 η 是一个比较困难的选择问题,尤其是事先并未知 ζ 和 η 值的情况下. 一个比较直观的解决思路是自动调整 ζ 和 η 值,但这会增加算法的计算量. 从总的实验情况来看,隐动作信息的噪声水平比较高(此时参数 η 值比较小)时,WTR 算法仍能随着 ζ 的增加而显著提升算法的识别性能. 这说明 WTR 算法受噪声数据影响较小.

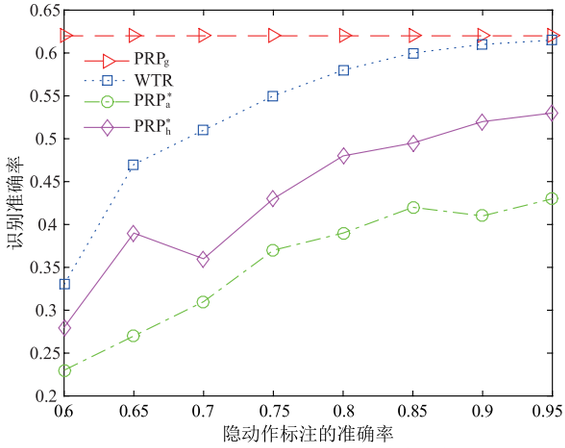

图 4 不同比率η下的识别准确率比较

5.2 算法识别效率比较

为了进一步验证算法的识别效率,我们随机产生出大小范围为 50 ~ 400 的测试集,且以每次递增 50 的方式来改变测试集大小. 实验设置所需的参数同前. 从图 5 不难看出,WTR 算法求解效率要优于其他识别算法,特别是当要问题个数达到 300 以上时表现得更为明显. 具体表现为 WTR 算法平均求解效率分别要比 PRP_g、PRP_a^* 和 PRP_h^* 算法提升 9%、13% 和 19%. WTR 算法的求解效率要优于其他算法归功于本文在实现 WTR 算法时,采用一种基于“路标”(Landmarks)[19] 的启发式搜索策略. 给定一个规划识别问题 Γ,动作“路标”都会出现在 Γ 的每一规划解中. 这就非常有利于指导规划的搜索方向,特别是能通过迭代局部搜索来实现“路标”的子目标,从而快速地求解出问题 Γ 的全局目标. 值得注意的是,PRP_g 算法并未引入这种“路标”的启发式搜索来求解问题. 因此,WTR 算法可以充分利用已有的辅助标注信息,挖掘出这种隐动作“路标”,且能够与 FF 启发函数相结合来减少搜索范围,有效地提高算法的搜索效率.

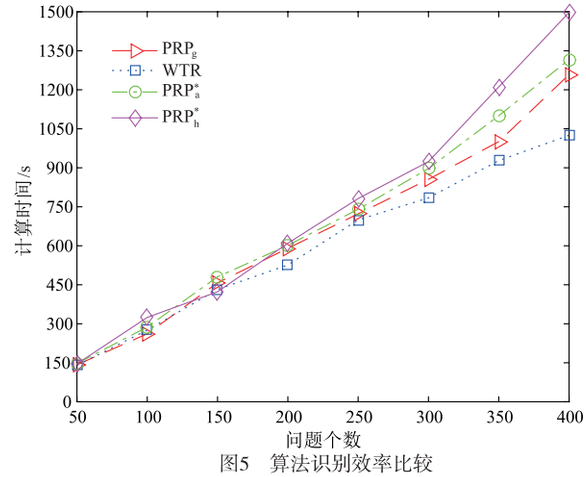

图5 算法识别效率比较

6 结论

本文提出了一种基于智能规划的工作流任务识别算法 WTR,将低层次细节方面的识别问题自动转换成高层次描述的智能规划识别问题,且能从实例库和外部辅助标注信息中,充分挖掘出潜在的知识,解决了传统算法识别困难和对噪声数据敏感等问题 . 实验结果表明,本文提出的算法是可行的.

参考文献

[1] Dumas, Marlon, et al. Process-Aware Information Systems: Bridging People and Software Through Process Technology [M]. New Jersey: John Wiley & Sons, 2005.

[2] Fan Shaokun, Lele Kang, Leon Zhao. Workflow-aware attention tracking to enhance collaboration management [J]. Information Systems Frontiers, 2015, 17(6): 1253 – 1264.

[3] Qanbari Soheil, Fei Li, Schahram Dustdar. Toward portable cloud manufacturing services [J]. IEEE Internet Compu-

ting,2014,18(6):77 – 80.
[4] Völter Markus,et al. Model-Driven Software Development: Technology, Engineering, Management[M]. New Jersey: John Wiley & Sons,2013.
[5] Eliceiri Kevin,et al. Biological imaging software tools[J]. Nature methods,2012,9(7):697 – 710.
[6] Branco Moisés Castelo,et al. A case study on consistency management of business and IT process models in banking [J]. Software & Systems Modeling, 2014, 13(3): 913 – 940.
[7] 孙瑞志,史美林. 一个支持动态变化的工作流元模型[J]. 电子学报,2002,30(S1):2052 – 2056.
SUN Rui-zhi, SHI Mei-lin. A meta-model supporting dynamic changing workflow[J]. Acta Electronica Sinica, 2002,30(S1):2052 – 2056. (in Chinese)
[8] Costa Lauro Beltrão, et al. The case for workflow-aware storage: An opportunity study[J]. Journal of Grid Computing,2015,13(1):95 – 113.
[9] Yan Yan, et al. Egocentric daily activity recognition via multitask clustering [J]. IEEE Transactions on Image Processing,2015,24(10):2984 – 2995.
[10] Boiman Oren, Michal Irani. Detecting irregularities in images and in video[J]. International Journal of Computer Vision,2007,74(1):17 – 31.
[11] Khalid Shehzad, Usman Akram, Shahid Razzaq. Behaviour recognition using multivariate m-mediod based modelling of motion trajectories[J]. Multimedia Systems, 2015, 21(5):485 – 505.
[12] Geib Christopher, Robert P Goldman. A probabilistic plan recognition algorithm based on plan tree grammars[J]. Artificial Intelligence,2009,173(11):1101 – 1132.
[13] Avrahami-Zilberbrand Kaminka. Fast and Complete Symbolic Plan Recognition[A]. Proceedings of the 19th International Joint Conference on Artificial Intelligence[C]. Burlington: Morgan Kaufmann,2005. 653 – 658.
[14] Ramirez Geffner. Plan recognition as planning[A]. Proceedings of the 21st International Joint Conference on Artificial Intelligence[C]. Burlington: Morgan Kaufmann, 2009. 1778 – 1783.
[15] Yang Qiang, Kangheng Wu, Yunfei Jiang. Learning action models from plan examples using weighted MAX-SAT [J]. Artificial Intelligence,2007,171(2 – 3):107 – 143.
[16] Hoffmann Jörg. FF: The fast-forward planning system[J]. AI magazine,2001,22(3):57 – 62.
[17] Voulodimos Athanasios, et al. A dataset for workflow recognition in industrial scenes[A]. Proceedings of the 18th IEEE International Conference on Image Processing [C]. Piscataway: IEEE,2011. 3249 – 3252.
[18] Han Wook-Shin, et al. iGraph: a framework for comparisons of disk-based graph indexing techniques[A]. Proceedings of the VLDB Endowment[C]. California: PVLDB,2010. 449 – 459.
[19] Karpas Erez, Carmel Domshlak. Cost-optimal planning with landmarks[A]. Proceedings of the 21st International Joint Conference on Artificial Intelligence[C]. Burlington: Morgan Kaufmann,2009. 1728 – 1733.

作者简介

曾　霖　男,1981 年生,中山大学数据科学与计算机学院博士研究生. 研究方向为智能规划、机器学习、数据挖掘.
E-mail: zsuzenglin@163.com

卓汉逵　男,1982 年生,中山大学数据科学与计算学院副教授,博士. 研究方向为智能规划、机器学习.

李　磊　男,1951 年生,中山大学数据科学与计算学院教授,博士生导师. 研究方向为数据库、数据挖掘、人工智能.