软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2016,27(7):1685−1699 [doi: 10.13328/j.cnki.jos.005041]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于可能世界模型的关系数据不一致性的修复*

徐耀丽, 李战怀, 陈 群, 钟 评

(西北工业大学 计算机学院,陕西 西安 710129)

通讯作者: 李战怀, E-mail: lizhh@nwpu.edu.cn

摘 要: 针对关系数据的不一致性虽然已有各种修复方法被提出,但这些修复策略在构建最终修复方案的过程中只分析函数依赖包含属性的信息(即,数据集的部分信息),且偏向于修复代价最小的方案,而忽略了数据集的其他属性以及这些属性与函数依赖包含属性之间的相关性.为此,提出一种基于可能世界模型的不一致性修复方法.它首先构造可能的修复方案,然后从修复代价和属性值相关性两个方面量化各个候选修复方案的可信性程度,并最后找出最优的修复方案.实验结果验证了所提出的修复方法取得了比现有基于代价的修复方法更好的修复效果.同时也分析了错误率和不同类型概率量化对所提出的修复方法的影响.

关键词: 不一致性;函数依赖;修复代价;可能世界;修复质量

中图法分类号: TP311

中文引用格式: 徐耀丽,李战怀,陈群,钟评.基于可能世界模型的关系数据不一致性的修复.软件学报,2016,27(7):1685−1699. http://www.jos.org.cn/1000-9825/5041.htm

英文引用格式: Xu YL, Li ZH, Chen Q, Zhong P. Repairing inconsistent relational data based on possible world model. Ruan Jian Xue Bao/Journal of Software, 2016,27(7):1685−1699 (in Chinese). http://www.jos.org.cn/1000-9825/5041.htm

Repairing Inconsistent Relational Data Based on Possible World Model

XU Yao-Li, LI Zhan-Huai, CHEN Qun, ZHONG Ping

(School of Computer Science and Technology, Northwestern Polytechnical University, Xi'an 710129, China)

Abstract: Various techniques have been proposed to repair inconsistent relational data that violate functional dependencies by optimizing the repair plan by the metric of repair cost. However, they may fall short in the circumstances where the erroneous data occurs in the left-hand side of a functional dependency or repair cost is not a reliable optimization indicator. In this paper, a novel repairing approach based on possible world model is proposed. It first constructs candidate repair plans and then estimates their possible world probabilities. The possible world probabilities are measured by quantifying both repair cost and candidate value appropriateness with regard to other related attribute values presented in relational data. Finally, extensive experiments on synthetic datasets show that the proposed approach performs considerably better than the cost-based approach on repair quality.

Key words: inconsistency; functional dependency; repair cost; possible world; repair quality

随着信息技术的普及,各行各业逐渐建立起各自的信息化系统,并在实践中积累大量的业务数据.互联网时代的到来,数据的种类和来源呈多样化发展,更新周期大为缩短,数据量大幅度飙升.如何有效而可靠地分析处理这些数据,为各行各业的经营决策服务,成为亟待解决的问题.然而在收集这些数据的过程中,由于人为因素,

* 基金项目: 国家重点基础研究发展计划(973)(2012CB316203); 国家自然科学基金(61332006, 61472321, 61502390); 西北工业大学基础研究基金(3102014JSJ0013, 3102014JSJ0005)

Foundation item: National Basic Research Program of China (973) (2012CB316203); National Natural Science Foundation of China (61332006, 61472321, 61502390); Northwestern Polytechnical University Foundation for Fundamental Research (3102014JSJ0013, 3102014JSJ0005)

收稿时间: 2015-10-14; 修改时间: 2016-01-12; 采用时间: 2016-02-22; jos 在线出版时间: 2016-03-22

CNKI 网络优先出版: 2016-03-22 13:23:37, http://www.cnki.net/kcms/detail/11.2560.TP.20160322.1323.009.html

如拼写错误、录入格式不相同和数据更新不及时等原因,导致数据集存在数据缺失、不一致等数据质量问题.这些问题的存在,使得数据的价值大打折扣,相关部门或企业在使用这些数据前需要花费大量的时间和人力审查并纠正存在的错误.数据质量专家指出,40%~50%的工程预算用于耗时费力地纠正错误数据[1].

数据的不一致性是指给定的数据集不满足预先给定的数据约束(如函数依赖、条件函数依赖等).不一致性修复是指通过某些数据操作,如插入、删除或更新,使得修复后的数据集满足给定的数据约束.不一致性修复问题面临两大挑战:一是满足给定数据约束的候选修复方案有很多,也就是说,候选修复方案的数目随着数据集的增加呈爆炸式增长;二是数据约束数目的增多以及它们之间的相互制约关系增加了不一致性修复问题的难度,可能会导致同一条元组的属性在上一轮策略中得到修改,在下一轮又被改回.这些挑战要求修复策略一定要高效且合理.

现有的数据修复方法[2,3]采用最小代价原则,也就是修复后的数据集与修复前的数据集的修改代价最小的方案,但文献[2]仅仅考虑出现在函数依赖的右边的属性.然而现实场景中,错误的出现位置是随机的,可能是函数依赖左边的属性,也可能是右边的属性.例如,函数依赖f_3:CK→NN 由于数字之间的差别不大,相比用户所属国家(NN),疲惫的录入人员更可能在输入用户编号(CK)时因走神而录入错误.虽然文献[3]提供含有变量的修改方案 V-repair,可能将函数依赖左边的属性值替换为变量 v,但该修复方案给数据集应用人员带来很大的不便,因为变量 v 的正确值是什么,还需要专家费神分析才能确定.

为了叙述上方便起见,ORDERKEY,O_ORDERSTATUS,R_NAME,P_MFGR,P_BRAND,O_SHIPPRIORITY,CUSTKEY 和 N_NAME 属性名分别代表订单编号、订单状态、地区名、供应商名、供应商标、订单船舶优先权、用户编号和用户所属国家,依次简记为 OR,OO,RN,PM,PB,OS,CK 和 NN.

另一方面,一个修复代价小的修复方案并不一定是一个修复质量高的修复方案.现有的修复策略仅仅考虑了数据集中与数据依赖相关的属性信息,并未考虑数据依赖以外的其他属性以及这些属性之间的相互关系.例如,在修复如图 1 所示的数据依赖 f_1 时,只关注与数据依赖相关的属性,也就是订单编号(OR)和订单状态(OO),而没有考虑数据依赖以外的属性,如地区名(RN)、供应商名(PM)和供应商标(PB)等,以及这些属性与订单编号(OR)、订单状态(OO)的相关信息.数据集中,元组属性列之间的相关性信息反映了数据的某些内在规律,使用机器学习方法可以抽取出这部分有用的信息,为修复方案的确定提供更坚实的推理支撑.然而,目前的修复研究中没有涉及相关性信息.本文的修复框架能够分析属性列之间的相关性并尝试修复错误出现在数据依赖左边的场景,且修复后数据集不包含变量.

TID	OR	OO	RN	PM	PB	OS	CK	NN
t_1	96	F	AMERICA	Manufacturer#5	Brand#53	0	217	UNITED STATES
t_2	96	P	ASIA	Manufacturer#5	Brand#52	0	217	INDIA
t_3	131	F	AMERICA	Manufacturer#5	Brand#53	0	187	PERU
t_4	64	F	AMERICA	Manufacturer#4	Brand#53	0	65	CANADA
t_5	99	F	AMERICA	Manufacturer#5	Brand#52	0	178	CANADA
t_6	32	O	ASIA	Manufacturer#5	Brand#52	0	262	CHINA
t_7	65	P	AMERICA	Manufacturer#3	Brand#31	0	34	CANADA
t_8	36	O	AMERICA	Manufacturer#3	Brand#32	0	232	PERU
t_9	70	F	ASIA	Manufacturer#2	Brand#22	0	130	CHINA

(I) 数据集 I;

(II) 函数依赖集 F={f_1:OR→OO,f_2:PB→PM,f_3:CK→NN}

Fig.1 Dataset and functional dependencies

图 1 数据集和函数依赖集

随着数据采集和处理技术在各行各业的广泛应用,数据的不确定性普遍存在.针对这些应用场景,研究者提供了多种描述数据不确定性的数据模型,这些模型的核心思想是可能世界模型[4].可能世界模型能够有效地刻画数据集合的不确定性.而概率数据库作为可能世界模型的表述方式,用概率值量化每个可能世界实例存在的可能性,是一种不确定性的数据库.本文借助概率数据库表述可能世界的思想,把满足函数依赖的某个候选修复方案视为一个可能世界实例,并用概率值量化该候选修复方案正确的可能性.其中,概率值计算是综合分析属性

列之间的相关性、修复代价而得到的.本文设计并实现基于可能世界模型的不一致性修复算法,以解决给定函数依赖下关系数据的不一致性修复问题.本文的主要贡献如下:

(1) 提出了一种基于可能世界模型的不一致性修复框架.该框架综合分析数据集的有用信息,如相关信息、统计信息等等,并为候选修复方案的选择提供信息支撑;

(2) 设计并实现了一种融合了修复代价和基于相关性概率量化的关系数据不一致性修复算法.该算法将候选修复方案类比为可能世界实例,将候选修复方案正确的可能性量化为可能世界实例的概率,搜索概率值较高的可能世界实例作为修复方案.该算法具有较好的修复效果;

(3) 在模拟数据集上进行大量实验,对本文提出的算法进行验证和分析.

本文第 1 节综述数据修复相关工作.第 2 节介绍相关基础知识并定义不一致性修复问题.第 3 节介绍基于可能世界的不一致修复框架,详细讲述如何构建候选属性记录,如何筛选候选属性值,如何量化候选属性值的概率(用于表示该属性值正确的可能程度).第 4 节设计启发式贪心算法,用于在满足给定数据依赖的可能世界实例集合中搜索概率值较高的作为修复方案.第 5 节验证算法的修复效果,并分析各个参数对算法的影响.最后总结全文工作并对未来工作加以展望.

1 相关工作

在信息化时代,数据成为企事业单位的重要资源之一.数据修复问题作为数据质量问题的重要组成部分,一直被学术界广泛关注.早期的数据修复系统或 ETL 商业工具包的主要目的是进行数据格式的转换[5],或者标记差异[6],或者由用户定制数据转换操作[7].这些修复方法并不能处理由函数依赖检测出的冲突情况.早期的数据修复方面的研究[5]主要关注使用插入和删除操作,但这样可能使得修复后的数据库实例信息流失,特别地,对于重要的信息,这是无法容忍的.大数据时代的到来,数据修复问题等数据质量问题重新得到了企事业单位和专家学者的广泛重视,并进行了广泛的研究,涌现出各式各样的修复算法和框架[1−3,8−11].

这些修复算法的核心思想是:计算出一个数据集 D',使得 D'满足函数依赖,并且从源数据集 D 转换成 D'的修改量最小.若这个转换操作采用修改元组属性值的方式,那么检测是否存在这样一个修复后数据集 D'已被证明是 NP 难问题[2].文献[2]提出了一个以修改量最小为优化目标的启发式修复模型,该模型借助等价类技术将冲突的检测阶段和修复属性值的确定阶段在一定程度上解耦,这样,修复属性值的确定阶段便能汇集更多有效信息,使得修复属性值的确定是全局最优的.文献[3]提出了一种近似最优修复算法,该算法利用超图可以产生一个与最优修复的差异在常数范围内的 V-repair.文献[8]针对概率数据库的不一致性问题,将概率数据库描述的可能世界划分成互不重叠的组,然后在每个组中使用文献[2]的启发式修复算法解决数据不一致性问题.这些修复方法在代价最小的假设下能够得到较好的实验效果,但是代价最小并不一定就是最合理的,而且数据集的信息很多,不仅有构成函数依赖的那些属性,还有其他的属性.而上述基于代价最小的修复框架和算法仅仅使用了函数依赖相关的属性列,其他属性列的信息被忽略.若函数依赖包含的属性和函数依赖未包含的属性之间有一定的关系,深入分析这部分信息有助于改进修复方案.

部分文献[1,9−11]将数据挖掘或机器学习的信息挖掘和学习技术引入数据修复领域.文献[9]融合了置信传播和关系依赖网络的技术,提出了一个以数据库为中心的数据清洗框架,该框架主要是清洗缺失的数据值.文献[10]针对互联网上数据错误的形式多样化现象,模拟各类错误的产生过程,提出了一个彻头彻尾的概率清洗框架,该框架包括一个基于贝叶斯的生成模块和采用最大熵综合各个错误产生过程的错误模块.生成模块主要是计算整个 Internet 中每个元组 t 的候选元组 t^*出现的概率,错误模块主要是综合各类错误产生过程中每个候选元组 t^*变成数据集中元组 t 的概率.文献[1]统计分析数据集的概率分布,综合各类机器学习方法为脏元组提供多个局部候选修复,将全局修复的预测问题转换为图优化问题,试图找到具有最大可能性并且修改量最小化的修复方案.文献[11]提出了一个最大限度地减少用户参与的交互式修复框架,主要包括排序模块和学习模块.排序模块主要是使用决策理论的信息价值,记为 VOI,来为待咨询的一系列修复更新组排序.学习模块主要是综合用户的反馈确定修复更新的不确定性,使用了主动学习技术.上述文献在进行数据清洗或修复时都使用了机器学

习的相关技术,从数据集中抽取出了有用信息来提升修复的质量或者减少用户的参与,但这些信息大都分散到数据修复过程的各个模块中,而且有些文献[1,9,10]完全忽略了数据依赖.本文的修复框架考虑了数据依赖对修复的有用信息,并将学习得到的有用信息以概率的信息形式统一整合在一起,使得信息的效用得到最大程度的使用.

2 不一致性修复简介

在数据质量领域中,现有的数据约束形式有很多种,如函数依赖(FDs)、条件函数依赖(CFDs)、匹配依赖(MDs)和拒绝约束(DCs)等等.数据的不一致性是指给定数据集 D 和数据约束集 C,至少存在某些元组 $T=\{t|t\in D\}$ 不满足某个 $c\in C$.本文的数据约束形式是函数依赖,主要是修复函数依赖检测出的不一致数据.本节首先介绍函数依赖,然后介绍数据不一致,并引出不一致性修复问题.

2.1 函数依赖

函数依赖表示数据库中两个属性集合之间的约束关系.每个函数依赖均有如下形式:

$$X\to Y,X\subset Attr(R),Y\subset Attr(R),$$

其中,$Attr(R)$是指 R 中所有属性名构成的集合.平凡的函数依赖在数据库实例中始终是成立的,它的定义为

$$X\to Y,Y\subseteq X,X\subset Attr(R),Y\subset Attr(R).$$

数据库中任意元组 t_i 和 t_j,如果$\forall x\in X,t_i[x]=t_j[x]$成立,那么由于 $Y\subseteq X$,必然有$\forall y\in Y,t_i[y]=t_j[y]$.也就是说,$I$ 始终满足平凡函数依赖.所以,我们仅仅考虑非平凡的函数依赖,其定义为

$$X\to Y,Y\not\subseteq X,X\subset Attr(R),Y\subset Attr(R),$$

任意函数依赖 $f:X\to Y$ 总能使用一元形式的函数依赖集合来等价表示.如果某函数依赖 $f:X\to Y$ 的右边属性集 Y 的秩为 1,则称 f 是一元的.由于一般的函数依赖集总能够直接转换成一元的函数依赖集,为了简化讨论,本文假设给定的函数依赖集 F 中每个函数依赖 $f\in F$ 都是一元的.

2.2 不一致性修复问题描述

所谓数据不一致是指给定数据库实例 I 不满足函数依赖集 F.若给定数据库实例 I 不满足函数依赖 $f:X\to A$,记为 $I\not\models f$.它的语义是指 I 中存在两个元组 t_1 和 t_2,它们的 X 属性值一一对应相同,但是 A 的属性值不相同,其数学描述为:$\exists t_1,t_2\in I,t_1[X]=t_2[X]$且 $t_1[A]\neq t_2[A]$.类似地,如果数据库实例 I 不满足函数依赖集 F 中某个函数依赖 $f\in F$,则称数据库实例 I 不满足 F,记为 $I\not\models F$.也就是说,I 相对于 F 是不一致的.若给定数据库实例 I 满足某个函数依赖 $f:X\to A$,记为 $I\models f$.它的语义是指:I 中任意两个元组 t_1 和 t_2,如果它们的 X 属性值一一对应相同,那么它们的 A 的属性值也相同.其数学描述为$\forall t_1,t_2\in I$,如果 $t_1[X]=t_2[X]$,那么 $t_1[A]=t_2[A]$.如果对于函数依赖集 F 中任意函数依赖 $f\in F$,I 均满足 f,则称 I 满足 F,记为 $I\models F$.也就是说,I 相对于 F 是一致的.

所谓不一致性修复问题是指:给定函数依赖集 F 和不一致的数据库实例 I,通过某些操作,如插入、删除或更新,使得修复后的数据库实例 I_R 满足函数依赖集,记为 $I_R\models F$.考虑到删除操作会导致有用信息的丢失,插入操作会在数据库中引入意义不大的记录信息,本文仅采用更新操作完成修复任务.

3 不一致性修复框架

在本节中,我们提出了一个基于可能世界模型的不一致性修复框架,如图 2 所示.该框架在借鉴前人的基于修复代价思路的基础上,融合机器学习方法和信息论中基于互信息的相关性分析,将数据中对修复有用的信息量化为候选修复值的正确的可能性,具体描述形式是概率值,取值范围是[0,1],使得从候选修复方案中筛选出更具合理性且修复质量更高的修复方案.

该框架以给定的不一致数据库实例 I 和函数依赖集合 F 作为输入,输出的是满足函数依赖集 F,即 $I_R\models F$,且概率值较高的可能世界实例 I_R.如图 2 所示,该框架主要包括 3 个阶段,依次是冲突元组候选属性记录和属性值的构建阶段(Gcarav)、候选属性值的概率量化阶段(Ccavp)以及满足给定函数依赖集 F 且概率值较高的可能世

界实例的搜索阶段(Sqi).

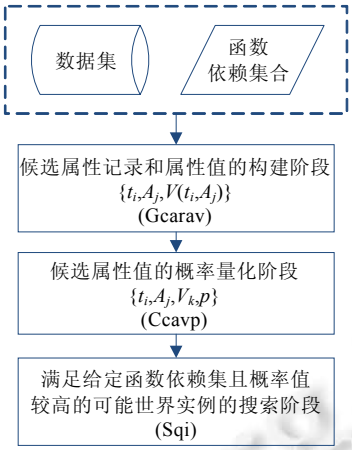


Fig.2 Framework for repairing inconsistent relational data based on possible world model

图 2 基于可能世界的不一致性修复框架图

冲突元组候选属性记录和属性值的构建阶段(Gcarav)主要是解决两个问题.

第一,候选属性记录的确定,就是错误出现的位置.

所谓冲突元组,是指那些导致数据库实例 I 不满足函数依赖集 F 的那些元组.例如图 1 中,对于 f_1:OR→OO,t_1 和 t_2 的 OR 属性值一一对应相等,但 OO 的属性值不相同,也就是说,t_1 和 t_2 不满足 f_1,所以 t_1 和 t_2 是冲突元组.对于 t_1 而言,可能是订单编号(OR)也可能是订单状态(OO).由于我们认为错误的位置是随机的,所以产生两条候选属性记录,分别是$\langle t_1$,OR,$Vs(t_1$,OR)$\rangle$和$\langle t_1$,OO,$Vs(t_1$,OO)$\rangle$.候选属性记录的形式是三元组$\langle t_i,A_k,Vs(t_i,A_k)\rangle$,其中,$t_i$ 是指 I 中第 i 个元组的唯一标识,A_k 是指 t_i 的第 k 个属性,$Vs(t_i,A_k)$是 t_i 的 A_k 属性的候选属性值构成的集合.

第二,候选属性值的选择.

对于某个候选属性记录,有哪些可能的候选属性值,也就是 $Vs(t_i,A_k)$的定义,具体详见第 3.1 节.

候选属性值的概率量化阶段(Ccavp)主要是从数据角度分析候选属性记录的每个候选属性值作为修复属性值的正确的可能性,并量化为概率值.简而言之,就是候选属性值的概率量化计算.我们综合考虑修复代价和相关性两方面信息来量化候选属性值的可信程度.直观上,假如一个候选属性值作为修复属性值的修复代价较小,那么该候选属性值是正确值的可能性就比较大,概率值比较高.直观上,如果函数依赖 f:$X\rightarrow A$ 不包含的 B 属性值,与候选属性值有比较大的正相关性,其中,$B\notin X\cup A$,那么该候选属性值正确的可能性比较大.基于相关性的概率量化就是借助这些函数依赖以外的属性信息,分析候选属性值的正确的可能性,具体详见第 3.2 节.

在可能世界模型中,一个元组的某属性可能有多个取值.将某元组的所有属性进行笛卡尔乘积可得到该元组的取值的所有可能情况.所谓一个可能世界实例就是从数据集中选择某些元组,然后对每个元组,选择它取值的一种可能情况,所有被选中元组的取值汇聚在一起就构成了该数据集的一个可能世界实例.考虑到修复问题需要求解所有元组的取值,对可能世界实例进行了约束.一个可能世界实例,不是针对数据集中的部分元组,而是针对所有元组.也就是说,对于数据集中每个元组选择它的一种可能情况,所有元组的可能情况就构成一个可能世界实例.例如图 1 所示,假设数据集为 I,函数依赖集为 F={f_1:OR→OO},可得冲突元组集 $I_{\bar{F}}=\{t_1,t_2\}$,进而可得候选属性记录集 *CARs*.所谓候选属性记录集,是由所有冲突元组的所有候选属性记录构成的集合.对候选属性记录集 *CARs* 中每个元素的候选属性值进行过滤操作.订单状态(OO)出现在函数依赖的右边,所以它的候选属性值有{P,F},不包含 O,而订单编号(OR)出现在函数依赖的左边,对于 t_1 而言,与 t_1[OO]取值相同的元组有{t_1,t_3,t_4,t_5, t_9},所以它的订单编号(OR)的候选属性值是{96,131,64,99,70}.类似地,对于 t_2 而言,它的订单编号(OR)的候选属性值是{96,65}.对候选属性记录集中属性值的取值进行过滤操作后,可得 *CARs*,它由 4 个候选属性记

录构成,分别是⟨t_1,OR,{96,131,64,99,70}⟩,⟨t_1,OO,{*P*,*F*}⟩,⟨t_2,OR,{96,65}⟩和⟨t_2,OO,{*P*,*F*}⟩.为了减少搜索概率值较高可能世界实例的计算量,直接忽略候选属性记录不包括的属性,如 t_1 的地区名(RN)、供应商名(PM)等属性.可能世界实例的构建过程是任取两个候选属性记录,对这两个候选属性值集合进行笛卡尔乘积,得到一个超级候选属性记录.将已经处理的候选属性记录从 *CARs* 中移除,并将得到的超级候选属性记录加入 *CARs*,循环执行,直到只剩下一个超级候选属性记录.该超级候选记录的属性值集合的每一个元素就是一个可能世界实例.所谓超级候选记录是扩展了候选属性记录的定义后所得,是由形式为⟨*TIDs*,*As*,*avs*⟩的三元组构成的集合.其中,*TIDs* 是一个 *n* 元组,即⟨tid_1,tid_2,…,tid_n⟩,该元组的元素由元组标识组成,与属性值集合 *avs* 中元素的属性值列表建立一一对应关系,表示属性值的唯一元组标识.类似地,*As* 也是一个 *n* 元组,即⟨A_1,A_2,…,A_n⟩,该元组的元素由元组属性名组成,与 *avs* 中元素的属性值列表建立一一对应关系,表示属性值的唯一属性标识.*TIDs* 和 *As* 元组的维度 *n* 由候选属性记录的数目决定.*avs* 是 *As* 取值情况构成的集合,该集合的元素是一个 *n* 元组,即⟨v_1,v_2,…,v_n⟩.如将⟨t_1,OR,{96,131,64,99,70}⟩和⟨t_1,OO,{*P*,*F*}⟩经过笛卡尔乘积操作后,可得 *TIDs*=⟨t_1,t_1⟩,*As*=⟨OR,OO⟩,*avs*={⟨96,*F*⟩,⟨96,*P*⟩,…}.

概率数据库可以很好地描述一个可能世界.例如,冲突元组集 $I_{\bar{F}}=\{t_1,t_2\}$ 描述的可能世界可表示成如图 3 所示,其中,每个属性值都附加一个概率值,该概率值是融合候选属性值的概率量化阶段(Ccavp)得到的候选属性值的基于相关性的概率和基于修复代价的概率而得.

TID	OR	OO
t_1	{(96,0.36),(99,0.24),(131,0.2),(64,0.15),(70,0.05)}	{(*P*,0.37),(*F*,0.63)}
t_2	{(96,0.66),(65,0.34)}	{(*P*,0.32),(*F*,0.68)}

Fig.3 Probabilistic database

图 3 概率数据库

一个可能世界实例的概率值计算分如下 3 步来计算.

- 首先,从候选属性值的量化阶段(Ccavp)得到每个属性值的两个概率并融合为一个;
- 然后,计算每个元组的概率,也就是对该元组所有属性值的概率进行累加和操作,记为 $p(t_i)$,其计算公式为

$$p(t_i)=\sum p(t_i,A_j,v_k),$$

其中,$p(t_i,A_j,v_k)$是指元组 t_i 的 A_j 属性值为 v_k 的概率值;

- 最后,将可能世界实例的所有元组的概率进行累加和操作,记为 $p(I(PD))$,它的计算公式如下所示:

$$p(I(PD))=\sum_{t_i\in I_{\bar{F}}} p(t_i),$$

其中,*PD* 是指表述可能世界的概率数据库,*I*(*PD*)指的是该概率数据库所描述的某个可能世界实例.

满足给定函数依赖且概率值较高的可能世界实例的搜索阶段(Sqi)主要是以可能世界的所有可能世界实例集合作为搜索空间,从中选择出满足 *F* 且实例的概率值较高的实例 I_R 作为修复方案.选择概率值较高的实例是因为可能世界实例的概率值是由候选属性值的概率值表示的,该概率值越高,那么表示构成实例的属性值的正确性越高、越可信.然而,采用这种直观的先构建再判断的搜索策略是很低效的.特别是当 $I_{\bar{F}}$ 中元组数目、候选属性数目和候选属性值的取值增大时,可能世界实例的数目呈爆炸式增长.假设 $I_{\bar{F}}$ 中的元组数目为 *n*,属性列的数目为 *k*,且某个元组 $t_i(i\in[1,2,\ldots,n])$、某个属性 $A_j(j\in[1,2,\ldots,k])$的取值都有 *m* 种,那么该概率数据库包含的可能世界实例数目为 $N=n\wedge m\wedge k$.鉴于先构建再判断的搜索策略是很低效的,为此,在第 4 节设计并实现了贪婪的搜索算法.

3.1 候选属性记录的构建

候选属性记录的构建主要用来确定哪些冲突元组是有错误的、错误元组的哪个属性含有错误、错误属性的可能取值有哪些,它们分别对应于候选属性记录三元组形式中的 *t*,*A* 和 *Vs*(*t*,*A*)的取值.冲突元组的候选属性记录的数目和候选属性值的数目直接决定了可能世界的实例数目.减少不必要的候选属性记录数目和过滤完全

不可能的候选属性值,可以减少错误值的干扰,同时减少概率量化阶段的计算量.

直观上,错误总是出现在小部分里面.对于违反某函数依赖 $f:X\to A$ 的冲突元组集 I_f 而言,我们假设小部分元组是有错误的,而大部分的元组是正确的,因此只需为小部分元组构建候选属性记录.所谓小部分元组是指 I 中与该元组 t 的 X 和 A 属性取值相同的元组数目,记为 $Num(t[XA])$.小于与该元组的 X 属性取值相同的元组数目,记为 $Num(t[X])$的一半.如图 1 中 f_2:PB→PM 的冲突元组集 $\{t_1,t_3,t_4\}$中,t_4 就属于小部分元组.因为供应商标(PB)取值为 Brand#53 的元组数目为 3,而供应商标(PB)取值为 Brand#53 且供应商名(PM)取值为 Manufacturer#4 的元组数目为 1,小于 3 的一半,所以属于小部分元组,而 t_1 和 t_3 则属于大部分元组,不需要为它们构建候选属性记录.如果 $Num(t[XA])$等于 $Num(t[X])$的一半,那么我们为所有的冲突元组构建候选属性记录,因为这些元组虽然不属于小部分元组,但是有 0.5 的概率是含有错误的.如图 1 中 f_1:OR→OO 的冲突元组集 $\{t_1,t_2\}$,订单编号(OR)为 96 的元组数目为 2,订单编号(OR)为 96 且订单状态(OO)为 F 的元组数目为 1,所以 t_1 虽然不属于小部分元组,但需要为 t_1 构建候选属性记录.

对于某些含有错误的元组,可以对该元组违反的函数依赖集进行分析,事先辨别该错误元组哪部分属性含有错误.这样就可以只为含有错误的属性构建候选属性记录,而不必为不含错误的属性构建候选属性记录,减少了不必要的计算和干扰.假设 t 为违反 $f:X\to A$ 的小部分元组,且 $|F_X|$大于 3,有如下两种情况可以事先辨别 t 的哪部分属性含有错误.

- 情况 1:t 的 X 中含有错误.

假设该元组 t 违反了 f,同时违反了给定函数依赖集 F_X 中所有的函数依赖,那么我们认为错误出现在函数依赖 X 属性中.其中,F_X 是指给定函数依赖集 F 中所有左边为 X 的函数依赖.

反证法:假设对于该元组错误出现在 A 中,那么它必定违反 f,但不一定同时违反所有左边为 X 的函数依赖.

- 情况 2:t 的 A 中含有错误.

假设该元组 t 违反了 f,同时不违反给定函数依赖集 F_X 中除去 f 以外的所有函数依赖,那我们认为 t 的 A 中含有错误.

反证法:假设 t 的 X 中含有错误,那么 t 至少违反 F_X 中两个函数依赖.

如果属于上述情况,那么只需为含有错误的属性构建候选属性记录;反之不属于上述两种情况,那么,我们假设错误元组的 X 属性和 A 属性均有可能含有错误,为 A 属性构建一个候选属性记录,即$\langle t,A,Vs(t,A)\rangle$,为 X 中每个属性 $X_i\in X$ 构建一个候选属性记录$\langle t,X_i,Vs(t,X_i)\rangle$.

候选属性值 $Vs(t,A)$的过滤原则是:只添加可能的候选值,以排除完全不可能的候选值的干扰.假设 F_A 是给定函数依赖集 F 中包含 A 的那些函数依赖,过滤的方法是:找出 t 为满足 F_A 需要保持一致的那些相关联元组,将这些相关联元组的 A 属性值作为候选属性值.对于 F_A 中某函数依赖 $f:X\to B$,若 A 与 B 相同,也就是在函数依赖的右边,那么计算 t 为满足 f 需要保持一致的所有相关元组集合,记为 $refTs(f)=\{t'|t'[X]=t[X]\}$.若 A 与 B 不相同,也就是在函数依赖的左边,那么相关元组集 $refTs(f)$是指与 t 的 C 属性值相同的那些元组构成的集合的并集,即:

$$refTs(f)=\bigcup\{t'\mid t'[C]=t[C]\},$$

其中,C 是指除去 A 以外的函数依赖属性 $C\in X\cup B\backslash A$.然后,将 F_A 中所有相关元组集的并集作为 t 的 A 属性相关的元组集,这些元组的所有 A 属性值就构成了候选属性值 $Vs(t,A)$,其数学定义如公式(1)所示.

$$Vs(t,A)=\left\{t'[A]\middle|t'\in\bigcup_{f\in F_A}refTs(f)\right\} \tag{1}$$

如图 1 所示,t_1 的订单状态(OO)在 f_1 的右边,所以 t_1 为满足 F_{OO} 的相关联元组集合 $refTs(f_1)$为 $\{t_1,t_2\}$,那么候选属性值 $Vs(t_1,OO)$为 $\{F,P\}$.而 t_1 的订单编号(OR)在 f_1 的左边,所以 t_1 为满足 F_{OR} 的相关联元组集合 $refTs(f_1)$为 $\{t_1,t_3,t_4,t_5,t_9\}$,也就是所有订单状态(OO)值为 F 的那些元组.

3.2 候选属性值的概率量化

候选属性值的概率值量化包括基于相关性的概率量化和基于修复代价的概率量化两部分,那么,候选属性

值的总概率是两者的平均值,以表示该候选属性值正确的可能程度.

3.2.1 基于相关性的概率量化

数据集中属性列之间的相关性暗含的推理信息,可以协助确定更合理的修复方案.鉴于本文在处理数据时把所有的属性都视为标称属性来处理,而信息处理的互信息在量化标称属性的相关性较好,采用互信息来进行独立性判断和相关程度度量.

相关性分析的核心思想是对某函数依赖$f:X\rightarrow A$,使用互信息量化函数依赖的属性A与数据集中属性B之间的相关程度,其中,$A\in U_f,B\notin U_f,U_f=X\cup A,U_f$是函数依赖的属性集.所谓函数依赖的属性集是指出现在函数依赖的属性构成的集合.若把属性 A 和 B 视为随机变量,那么 A 和 B 的互信息记为 $I(A;B)$,其取值范围是[0,min($H(A)$,$H(B)$)],其数学定义为 $I(A;B)=H(A)-H(A|B)$.熵是随机变量不确定性的度量,熵越大,变量的不确定性越高.对于随机变量 A,它的数学定义如公式(2)所示.

$$H(A)=-\sum_{a}p(a)\times\log p(a) \tag{2}$$

其中,$p(a)$是指变量 A 的取值为 a 事件发生的概率值.给定 B 后,A 的条件熵 $H(A|B)$是指观察到变量 B 后 A 的不确定性,它的数学定义如公式(3)所示.

$$H(A\mid B)=\sum_{b}H(A\mid B=b)\times p(B=b) \tag{3}$$

将 B 的取值为 b 条件下变量 A 视为随机变量,那么 $H(A|B=b)$表示该随机变量的熵;$p(B=b)$是指变量 B 的取值为 b 事件发生的概率值.由于 A 的熵 $H(A)$量化了观察到变量 B 之前 A 的不确定程度,而条件熵 $H(A|B)$是观测到 B 后 A 尚存的不确定性,所以,互信息 $I(A;B)$表示变量 B 包含多少 A 的信息.当 $I(A;B)$为 0 时,表明 B 与 A 是相互独立的;当 $I(A;B)$不为 0 时,表明 B 与 A 是相关的,并且两个变量的互信息越大,那么它们之间的相关程度越高.所谓无冲突数据集 I_F 是指 I 中去掉那些冲突元组后,得到的元组集合,即 $I_F\models F$.如图 1 所示,假设数据集为 I,函数依赖集为 $F=\{f_1:\mathrm{OR}\rightarrow\mathrm{OO}\}$,可得无冲突元组集:

$$I_F=\{t_3,t_4,t_5,t_6,t_7,t_8,t_9\},U_{f_1}=\{\mathrm{OR},\mathrm{OO}\},\mathrm{OO}\in U_{f_1},\mathrm{PM}\notin U_{f_1}.$$

依据 I_F,可得 OO 变量的值域为$\{F,O,P\}$,$p(F)=4/7$,$p(P)=1/7$,$p(O)=2/7$.进而,依据公式(2)可得 OO 的熵 $H(\mathrm{OO})$约为 0.96.类似地,PM 变量取值为 Manufacturer#5,Manufacturer#4,Manufacturer#3,Manufacturer#2 发生的概率分别为 3/7,1/7,2/7,1/7.OO 变量在 PM=Manufacturer#5 的条件下取值为 F,O,P 发生的概率分别为 2/3,1/3,0.可得 $H(\mathrm{OO}|\mathrm{PM}=\mathrm{Manufacturer\#5})\approx0.63$,依据公式(2),可得:

$H(\mathrm{OO}|\mathrm{PM}=\mathrm{Manufacturer\#4})=0$,$H(\mathrm{OO}|\mathrm{PM}=\mathrm{Manufacturer\#3})\approx0.69$,$H(\mathrm{OO}|\mathrm{PM}=\mathrm{Manufacturer\#2})=0$.

将这些值代入公式(3),可得给定 PM 后 OO 的条件熵 $H(\mathrm{OO}|\mathrm{PM})\approx0.47$.最后,依据互信息的定义,可得 $I(\mathrm{OO};\mathrm{PM})\approx0.49$.类似地,可得 $I(\mathrm{OO};\mathrm{OS})=0$,表明订单船舶优先权(OS)与订单状态是独立的.$I(\mathrm{OO};\mathrm{RN})\approx0.07$ 比 0.47 小,表明供应商名(PM)与订单状态(OO)相关程度比地区名(RN)与订单状态(OO)相关程度要高.

相关性分析的处理流程以无冲突数据集 I_F 和函数依赖集 F 为输入,输出相关信息矩阵 $array_c$,如过程 1 所示.L3 是将 I_F 的属性分为两类 U_F 和 $U_{\bar{F}}$,其中,U_F 是函数依赖的属性集,$U_{\bar{F}}$ 是非函数依赖的属性集.L5~L12 用来判断 U_F 中每个属性 A 与其他属性之间是否独立:若不独立,则量化它们之间的相关性程度.L7 是依据定义计算属性 A 和 B 的互信息 $I(A;B)$.L8~L11 则是依据互信息的性质,如果 $I(A;B)$为 0,那么将 B 添加到 A 的独立属性列表(*indANs*)中;否则,将 B 添加到 A 的相关属性列表(*dANs*)中,并将 $I(A;B)$作为两者相关程度的量化值.L12 是将计算得到的相关信息存储在 $array_c$ 中.

过程 1. 数据相关性分析.

输入:无冲突数据集 I_F,函数依赖集 F;

输出:相关信息矩阵 $array_c$.

1. BEGIN
2. $array_c\leftarrow\varnothing$
3. $[U_F,U_{\bar{F}}]\leftarrow splitAttributes(I_F,F)$

4. $U \leftarrow U_F \cup U_{\bar{F}}$
5. FOR EACH $A \in U_F$ DO
6. FOR EACH $B \in U \backslash A$ DO
7. $I(A;B) \leftarrow computeMI(A,B)$
8. IF $I(A;B)==0$ THEN $A.indANs.add(B)$
9. ELSE
10. $A.dANs.add(B)$
11. $A.cc.add(I(A;B))$
12. $array_c.add(A)$
13. RETURN $array_c$
14. END

基于相关性的概率量化是使用机器学习算法和相关性信息矩阵 $array_c$,为每个候选值的正确程度进行概率量化.一方面,由于本文关注标称属性数据的修复问题,而 k 最近邻算法简单且易于改进,在大多数情况下表现良好;另一方面,由于 k 最近邻分类算法在预测阶段使用投票的方式,这样可以把投票比例转换为候选值的可信程度的概率值,所以考虑使用 k 最近邻算法结合相关信息矩阵分析并量化每个候选值正确的可能性.相关性信息矩阵 $array_c$ 可减少机器学习方法的计算量,同时提高候选值可信程度量化的准确率.例如在计算候选属性记录 $\langle t_1,OO,\{P,F\}\rangle$ 中每一个候选属性值的概率时,机器学习方法需要考虑 t_1 的其他属性列的信息,如订单编号(OR)、供应商名(PM)等.由于订单状态(OO)与订单船舶优先权(OS)的互信息 $I(OO;OS)$ 为 0,那么它们独立.在计算 t_1 的订单状态(OO)候选属性值的概率时,可以把订单船舶优先权(OS)排除掉,减少计算量,同时不影响概率量化的准确性.另一方面,如果两个变量的互信息越大,表明它们之间的相关程度越高,那么在候选属性值的概率量化时,相关属性信息按照互信息的大小进行加权处理,使得候选属性值的概率计算更准确.具体就是在 k 最近邻算法的加权距离测量公式中,如公式(4)所示,将相关属性 A_k 的属性值与 C 属性的互信息 $I(C;A_k)$ 作为它们编辑距离的权重.

$$d(t,x)=\sqrt{\sum_{A_k \in dANs_c} I(C;A_k)\times diff(t[A_k],x[A_k])} \tag{4}$$

其中,C 为候选属性记录的属性,k 最近邻算法把 C 作为类别属性来处理.

由于本文处理的是符号数据,也就是标称属性,所以属性值 $t[A_k]$ 与 $x[A_k]$ 之间的距离,如公式(4)的 $diff(t[A_k],x[A_k])$ 所示,采用字符串的编辑距离,而非欧式距离.如图 1 所示,假设测试元组 t 为 t_1,训练数据集中某元组 x 为 t_3,候选属性记录的属性 C 为 OO,与 OO 相关的属性集 $dANs_C$={OR,RN,PM,PB,CK,NN},它们与 OO 的互信息依次为{0.95,0.07,0.48,0.76,0.95,0.29},按照公式(4)计算可得 $d(t_1,t_3)\approx 14.25$.假设 k 近邻的训练集为 $I_F=\{t_3,t_4,t_5,t_6,t_7,t_8,t_9\}$,且 k 取值为 2,那么计算训练集中所有元素与 t_1 的距离,并选择距离最近的前 2 个元组{t_4,t_5,},由于它们的 OO 属性的取值均为 F,所以候选属性记录$\langle t_1,OO,\{P,F\}\rangle$中候选属性值 F 的相关性概率值为 1.

3.2.2 基于修复代价的概率量化

修复代价概率量化主要是预估每个候选属性值 $v' \in Vs(t,A)$作为正确值更新到数据库实例后,为保证修改后数据库实例满足 F_A,需要进行的修改操作数目或者更新操作前后冲突元组数目变化量,然后使用指数函数将修改操作数目或者冲突元组数目变化量转换为[0,1]的概率值.

假设 t 违反 $f:X\rightarrow B$.如果 $A\in X$,假如要把 I 中元组 t 的属性 A 的值改为 v',修改后数据库实例记为 I',那么修复代价记为 $RCost(t,A,v,v',f)$,其数学表示为公式(5).

$$RCost(t,A,v,v',f)=VT(I',f)-VT(I,f) \tag{5}$$

$VT(I,f)$是指 I 中不满足函数依赖 f 的元组数目.类似地,$VT(I',f)$是指修改后 I'中不满足函数依赖 f 的元组数目,其中,$f\in F_A$.如果把元组 t 的 A 属性的正确值 v 改为 v',会导致修复代价变为负数,不便于使用指数函数 $y=e^{-RCost(t,A,v,v',f)}$将修复代价转换为取值为[0,1]的概率值.为此,对公式(5)进行了一次坐标平移变换.坐标平移变

换的思路是:在元组 t 的属性 A 的候选属性值的修复代价为负值时,找到最小的修复代价,记为 min$RCost(t,A,f)$,其数学表示为公式(6).

$$\min RCost(t,A,f)=\min\{RCost(t,A,v,v',f)|v'\in Vs(t,A)\} \tag{6}$$

将该候选属性记录中所有候选属性值的修复代价加上最小的修复代价的绝对值,这样就可以保证候选属性值的修复代价概率量化均为 0~1 之间的取值.

如果 A 与 B 相同,那么对于 $Vs(t,A)$中某候选属性值 v'的修复代价是指:为保证 t 满足 f,需要将 I 中所有 t'的 A 属性值由 v 改为 v'的编辑距离值之和,其中,t'是指 X 取值为 $t[X]$的元组,我们使用编辑距离作为修改操作数目.这里未使用更新操作前后冲突元组数目的变化量,是由于修改操作数目量化在某些情况下与直观感觉相符合.如图 1 所示,直观上,t_2 的订单状态(OO)候选属性值{P,F}的概率都应该是 0.5,确实无法分辨.按照修改操作数目度量 t_2 的候选属性值 P 和 F 的修复代价相同,可得与直观相符合的概率值.而使用冲突元组数目变化量则会使得 t_2 的候选属性值 P 和 F 的修复代价不相同,因为若更新为 P,更新前与 f 冲突的元组数目为 2,更新后还是 2,所以冲突变化量为 0.若更新为 F,更新前与 f 冲突的元组数目为 2,更新后变为 0,所以冲突变化量为−2.在经过坐标变化后,得到的概率值是不相同的,与直观不符.

由于 A 属性值的修改会影响到所有与属性 A 有关系的函数依赖集 F_A,所以需要对 F_A 中每个函数依赖计算修复代价,然后累加求和后,使用指数函数 $y=e^{-RCost}$ 转换为[0,1]的概率值.

4 不一致性修复算法

本节设计并实现了一种贪心修复算法,该算法以冲突元组候选属性记录和属性值的构建阶段和候选属性值的概率量化阶段得到的候选属性记录集合 CRs 为输入,输出是满足 F 的数据集 I_R.已有研究[2]证明:计算满足函数依赖集,且修复代价最小的数据修复问题是一个 NP 难问题.假如只考虑基于修复代价的概率值,不考虑基于相关性的概率值,然后使用指数函数的逆运算——对数函数,可将修复代价的概率值变换为修复代价,那么我们的问题就可以转化为文献所述的 NP 难问题.这说明,寻找满足函数依赖集并且概率值较高的可能世界实例也是一个 NP 难问题.这表明了算法的启发式特点.本节的贪心搜索算法是使用贪心技术的启发式算法.本节还证明了算法的可终止性,并分析了算法的复杂度.

该贪心算法一方面避免了计算量惊人的可能世界实例枚举操作,另一方面能够在满足函数依赖集 F 的所有可能世界实例中快速地找到概率值之和较高的实例,也就是修复解决方案.该算法的处理思路是:

首先,将候选属性记录由原来的三元组扩展为六元组,该六元组的形式为$\langle t_i,V_j,Vs,RPs,CPs,Ps\rangle$.其中,$RPs$ 用来存放修复代价的概率量化值,CPs 用来存放基于相关性的概率量化值,Ps 用来存放 RPs 和 CPs 融合后的概率值以表示该候选属性值正确的可能性.这 3 个概率值均与 Vs 建立一一对应关系.接着,依照候选属性记录集合 CRs 中每个候选属性记录$\langle t_i,V_j,Vs,Ps\rangle$将数据集 I 中元组 t_i 的 A_j 属性的属性值设置为 uncertain,表示未确定,此时,数据集记为 I_U.然后,从所有的候选属性值中抽取出满足函数依赖 F 的所有候选属性值,并选择概率值最高的候选属性值,将原来设置为 uncertain 的位置更新为该候选属性值.此时,数据集由原来的 I_U 变为 I'_U .不断迭代这个过程,直到所有的 uncertain 都标记为被更新过.假如该贪心算法可以找到满足函数依赖且概率值较高的可能世界实例,那么 uncertain 标记位的属性值就组成了该可能世界实例.此时,数据集由原来的 I_U 变为 I_R.假如执行若干次迭代后数据集变为了 I'_U ,找不到满足函数依赖 F 的候选属性值,但还存在 uncertain 标记的位置,那么将 uncertain 标记的元组 t 属性值必定与 I'_U 中某个元组 t'一起对某函数依赖 f 构成了冲突,那么将元组 t 的属性值改为与元组 t'相同,使得函数依赖不冲突.当所有的 uncertain 都处理完毕后,该贪心算法就找到了一个满足函数依赖且概率值较高的可能世界实例.

该贪心算法的输入是数据集 I、函数依赖集 F 和冲突元组的候选属性记录集合 CRs,其元素 $cr\in CRs$ 是冲突元组的那些具有候选属性值的属性,具体形式是一个六元组$\langle t_i,V_j,Vs,RPs,CPs,Ps\rangle$.输出满足 F 的数据集 I_R.其伪代码如算法 1 所示.L2~L10 是将候选属性集 CRs 中每个候选属性值的修复代价概率量化 RP 和基于相关性的概率量化 CP 中概率值融合成 P.L8 是将所有的候选属性值添加到 $sortedAllAVs$ 中.L9 将数据集 I 中元组 $cr.tid$

的 *cr.AN* 属性的属性值设置为 uncertain.L11~L28 是贪心搜索满足函数依赖且概率值较高的可能世界实例的过程.L14 是将所有的属性值按照其概率值降序排列.L16 是判断当前概率值最高的属性值 *sortedAllAVs.get*[*j*]是否满足函数依赖:如果满足,那么将 *isFind* 设置为 true,并将该属性值所属的候选属性记录的元组唯一标识、属性标识等信息存放到 *selectAVInfo*,跳出本次循环.L23 是当找到满足函数依赖并且概率值最高的值后,使用 *selectAVInfo* 的信息更新 I_U 为 I'_U,并将候选属性的候选属性值从 *sortedAllAVs* 中移除.L26 就是处理还存在 uncertain 标记的位置,但是未找到满足函数依赖 *F* 的候选属性值情况.L29 是当所有的不确定标记都被覆盖后,得到的数据集 I_U 就是修复方案 I_R.

算法 1. 搜索满足函数依赖集且概率值较高的可能世界实例贪心算法,记为 PWM 算法.

输入:数据集 *I*,函数依赖集 *F*,候选属性记录集合 *CRs*;

输出:满足函数依赖集 *F* 的数据集 I_R.

```
1.  BEGIN
2.    FOR EACH cr∈CRs DO
3.        FOR i←0 TO CRs.Avs.length DO
4.            P←(cr.getRP(i)+cr.getCP(i))*0.5
5.            add P into Ps
6.        END FOR
7.        cr.Ps←Ps
8.        sortedAllAVs.addAll(cr.getVs())
9.        setUncertain(I,cr.tid,cr.an)
10.   END FOR
11.   I_U←I
12.   FOR i←0 TO |CRs| DO
13.       isFind←false
14.       Sort sortedAllAVs by av.p in descending order
15.       FOR j←0 TO |sortedAllAVs| DO
16.           isFind←isConsistent(I_U,F,sortedAllAVs.get[j])
17.           IF (isFind)
18.              selectAVInfo←getCrAndAV(sortedAllAVs.get[j])
19.              BREAK
20.           END IF
21.       END FOR
22.       IF (isFind)
23.          I'_U ← update(I_U,selectAVInfo)
24.          I_U ← I'_U
25.       IF (!isFind)
26.          I'_U ← processExceptionCase(I_U,selectAVInfo)
27.          I_U ← I'_U
28.   END FOR
29.   Return I_R=I_U
30. END
```

定理 1. 给定任意数据集 *I* 和函数依赖集合 *F*,算法 PWM 总能终止,并且返回一个修复方案 I_R,使得 $I_R \models F$.

证明:候选属性元组集的数目为 *N*,也就是数据集 I_U 中不确定性标记的属性值数目,每一步都能处理掉一个

不确定性标记,得到 I'_U,且 I'_U 满足函数依赖集合 F.所以,算法 PWM 是可终止的,并且返回一个修复方案 I_R. □

- 复杂性分析

该算法的关键操作是找到一个能够更新不确定性标记的候选属性值.由伪代码可知,该算法的复杂度是 $O(|CRs|\times|sortedAllAVs|)$.假设数据集 I 中元组的数目是 N,属性名的数目为 M,每个属性名的候选属性值的数目为 L,那么最坏情况下,$|CRs|$的取值为 $N\times M$,而$|sortedAllAVs|$的取值为 $N\times M\times L$,所以最坏情况下,算法复杂度为 $O(N^2\times M^2\times L)$.然而,实际情况没有这么差,因为冲突元组的数目远远小于 N,经过过滤操作后候选属性值的数目 L 会减少很多,而且每当确定一个候选属性值,那么下一次循环中 *sortedAllAVs* 集合中与该属性值具有相同元组标识和属性名的属性值就会被移除,使得 *sortedAllAVs* 中元素数目很快减少.

5 实 验

本节在模拟数据集上评估并描述基于可能世界模型的不一致性修复方法的实验结果.实验运行环境、数据集以及数据质量的评估标准,如类似信息检索研究领域的查全率、查准率和 *F*-measure,会在第 5.1 节详细介绍.第 5.2 节则在算法的有效性方面进行详细分析.

5.1 实验配置

所有实验的运行环境配置为 Intel(R) Core(TM) i7-4710MQ 2.50GHz 处理器,8GB 内存,Windows 8.1 中文版 64 位操作系统.数据集使用 MySQL Workbench 5.2.47 CE 存储,编程语言是 Java.

实验使用的数据集有 TPCH 数据集——它是事务处理性能委员会提供的 TPC-H 基准测试集.其中,TPCH 数据集是模拟数据集.为了便于评价,假设源数据集是正确的,我们采用的插错策略是随机地选择数据集的一个子集,然后以概率 *right*%向某函数依赖 $f:X\rightarrow A$ 的右边属性 A 插入错误,该错误是从属性 A 的值域 $DOM(A)$中选择与 A 的原属性值不同的值 a'替换掉 A 的属性值 a;反之,则以概率 1–*right*%在函数依赖的左边属性名 $B\in X$ 插入错误,最终保证插入的错误率达到 *noi*%.其中,*noi*%是指错误的属性值数目与整个数据集大小的比值,且所有含错误的元组均是冲突元组,即,不满足函数依赖.本实验使用了 22 个函数依赖.

- 评价基准

评价基准扩展信息检索中常用的查准率、查全率和 *F*-measure.查准率是指修复算法正确更新的属性值数目与更新的属性值数目的比值,记为 Precision.查全率是指修复算法更新的属性值数目与数据集中错误的属性值数目的比值,记为 Recall.*F*-measure 由查全率和查准率计算得到,定义如下:

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

5.2 实验结果分析

本节在算法的有效性方面详细分析实验效果,其中,本文的方法用标签 PWM 标记,文献[2]中基于修复代价的修复方法用标签 ECR 标记.

图 4 是固定错误率 *noi*%为 0.06 和 *right*%为 0.8,数据集的元组数目从 200 增长到 1 600 时,查全率、查准率和 *F*-measure 的变化情况.

由图 4 可知,该方法在查全率、查准率和 *F*-measure 的度量上均高于基于修复代价的修复方法.

图 5 是固定 *right*%为 0.8,数据集的元组数目为 1 000,错误率 *noi*%从 0.02 变化到 0.1 时,查全率、查准率和 *F*-measure 的变化情况.

由图 5 可知,基于修复代价的修复方法和基于可能世界模型的修复方法随着错误率的增长,修复质量变化不大,但我们的方法在查全率、查准率和 *F*-measure 的度量上依然高于基于修复代价的修复方法.

图 6 是固定 *right*%为 0.8,数据集的元组数目为 1 800,错误率 *noi*%从 0.01 变化到 0.1 时,查全率和查准率的变化情况.

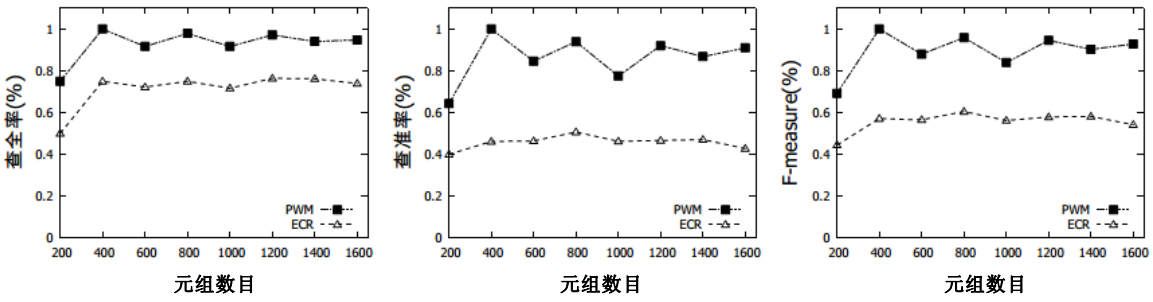


Fig.4 Recall, precision and *F*-measure on tuples number

图 4 数据集的元组数目变化情况下查全率、查准率和 *F*-measure 的对比

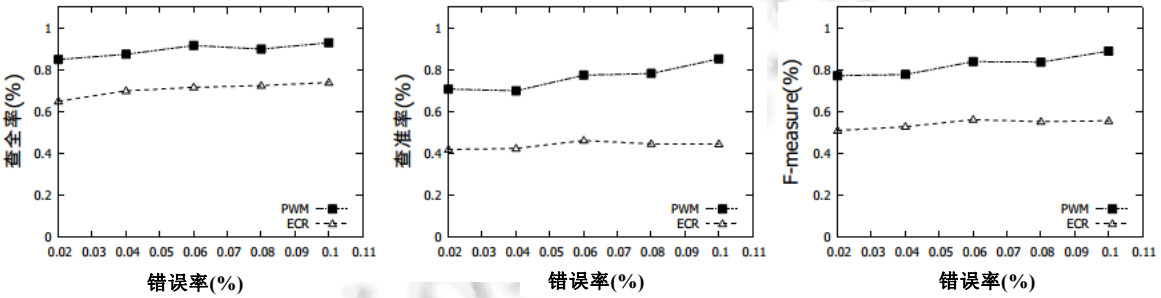


Fig.5 Recall, precision and *F*-measure on error rates

图 5 错误率变化情况下查全率、查准率和 *F*-measure 的对比

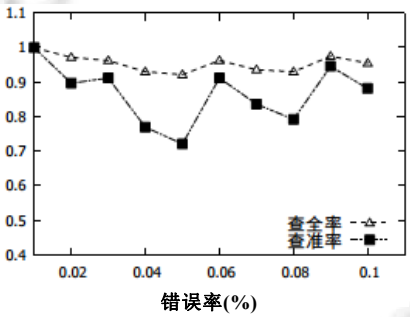


Fig.6 Recall and precision of PWM on error rates

图 6 不同错误率对 PWM 方法的影响

由图 6 可知,查全率随着插入错误的增多,变化幅度较小,而查准率则会有较大的波动,这主要是因为错误出现形式多样,特别是在函数依赖的左边出现错误,且左边的属性值分布比较混乱、程度比较高时,基于相关性的概率和修复代价的概率量化均无法有效分析真值的概率值,使得原有错误未改正确,而新的错误却被引入.

图 7 是对比了在固定 *right*%为 0.8,*noi*%为 0.06,数据集的元组数目从 800 变化到 2 000 的情况下,使用不同概率量化方法在查全率、查准率和 *F-measure* 的变化情况.其中,候选属性值的概率值仅使用基于相关性的概率量化,即 Only Correlation 所标记,仅仅使用修复代价的概率量化,即 Only Repair Cost 所标记,以及同时使用两者的概率量化,即 Both 所标记.

由图 7 可知,在仅使用修复代价的概率下,查准率和 *F*-measure 有比较大的波动,表现不够稳定,而使用基于相关性的概率量化则修复效果比较稳定.

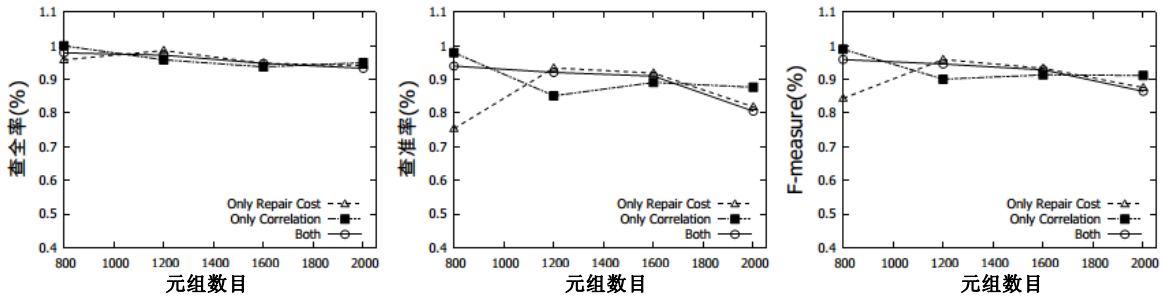


Fig.7 Recall, precision and *F*-measure of PWM on different probability metrics

图 7 不同概率值组合对 PWM 方法的影响

6 总结与展望

本文提出了一种基于可能世界模型的不一致性修复框架,设计并实现了一种融合了修复代价、属性值相关性的概率量化的不一致性修复算法,并在模拟数据上验证了算法的有效性.

在数据修复领域还有很多公开问题,比如:

- 在不一致性修复问题上,比较多的研究是关于文本数据的修复,而数值型数据的相关研究比较少;
- 大多数修复是假定数据依赖是存在的,然后以此为数据依赖提出近似算法,若数据依赖不存在,那么如何进行修复的研究相对较少.

今后,我们将对上述问题进行探索性研究.

References:

[1] Yakout M, Berti-Equille L, Elmagarmid AK. Don't be scared: Use scalable automatic repairing with maximal likelihood and bounded changes. In: Proc. of the ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD 2013). New York: ACM Press, 2013. 553−564. [doi: 10.1145/2463676.2463706]

[2] Bohannon P, Flaster M, Fan WF, Rastogi R. A cost-based model and effective heuristic for repairing constraints by value modification. In: Proc. of the ACM SIGMOD Int'l Conf. on Management of Data. Baltimore: ACM Press, 2005. 143−154. [doi: 10.1145/1066157.1066175]

[3] Kolahi S, Lakshmanan LVS. On approximating optimum repairs for functional dependency violations. In: Proc. of the 12th Int'l Conf. on Database Theory (ICDT 2009). St. Petersburg: ACM Press, 2009. 53−62. [doi: 10.1145/1514894.1514901]

[4] Zhou AY, Jin CQ, Wang GR, Li JZ. A survey on the management of uncertain data. Chinese Journal of Computers, 2009,32(1): 1−16 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2009.00001]

[5] Galhardas H, Florescu D, Shasha D, Simon E, Saita CA. Declarative data cleaning: Language, model, and algorithms. In: Apers PMG, Atzeni P, Ceri S, Paraboschi S, Ramamohanarao K, Snodgrass RT, eds. Proc. of the 27th Int'l Conf. on Very Large Data Bases (VLDB 2001). Roma: Morgan Kaufmann Publishers, 2001. 371−380.

[6] Raman V, Hellerstein JM. Potter's wheel: An interactive data cleaning system. In: Apers PMG, Atzeni P, Ceri S, Paraboschi S, Ramamohanarao K, Snodgrass RT, eds. Proc. of the 27th Int'l Conf. on Very Large Data Bases (VLDB 2001). Roma: Morgan Kaufmann Publishers, 2001. 381−390.

[7] Rahm E, Do HH. Data cleaning: Problems and current approaches. IEEE Data Engineering Bulletin, 2000,23(4):3−13.

[8] Lian X, Lin YC, Chen L. Cost-Efficient repair in inconsistent probabilistic databases. In: Proc. of the 20th ACM Conf. on Information and Knowledge Management (CIKM 2011). Glasgow: ACM Press, 2011. 1731−1736. [doi: 10.1145/2063576.2063826]

[9] Mayfield C, Neville J, Prabhakar S. ERACER: A database approach for statistical inference and data cleaning. In: Proc. of the ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD 2010). Indianapolis: ACM Press, 2010. 75−86. [doi: 10.1145/1807167.1807178]

[10] Hu YH, De S, Chen Y, Kambhampati S. Bayesian data cleaning for Web data. arXiv: 1204.3677, 2012.

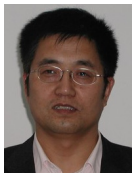
[11] Yakout M, Elmagarmid AK, Neville J, Ouzzani M, Ilyas IF. Guided data repair. PVLDB, 2011,4(5):279−289. [doi: 10.14778/1952376.1952378]

附中文参考文献:

[4] 周傲英,金澈清,王国仁,李建中.不确定性数据管理技术研究综述.计算机学报,2009,32(1):1−16. [doi: 10.3724/SP.J.1016.2009.00001]

徐耀丽(1987－),女,河南安阳人,硕士,CCF 学生会员,主要研究领域为数据质量.

陈群(1976－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为大数据管理,物联网信息管理.



李战怀(1961－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为数据库理论与技术.

钟评(1985－),男,硕士,CCF 学生会员,主要研究领域为数据质量.