软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, 2023,34(3):1065−1086 [doi: 10.13328/j.cnki.jos.006793]

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

兼顾行列的时序数据质量规则发现*

丁小欧[1], 李映泽[1], 王 晨[2], 王宏志[1], 李昊轩[1]

[1](哈尔滨工业大学 计算机科学与技术学院, 黑龙江 哈尔滨 150001)

[2](大数据系统软件国家工程研究中心(清华大学), 北京 100084)

通信作者: 王宏志, E-mail: wangzh@hit.edu.cn

摘 要: 智能装置设备产生的时序数据增长迅速, 存在严重的数据质量问题. 劣质时序数据质量管理和数据质量提升技术需求日益迫切. 时序数据的有序时窗、行列关联等特点, 为时序数据质量语义表达提出了挑战. 提出了一种同时考虑时序数据在行与列上的数据依赖信息的数据质量规则, 即时序否定约束 TDC. 研究了 TDC 的定义与构建方法, 从时窗与多阶表达式运算这两个方面, 对已有的数据质量规则体系进行表达力的扩展, 并提出针对兼顾行列的时序数据质量规则挖掘方法. 在真实时序数据集上开展大量实验, 实验结果验证了该方法能够有效且高效地挖掘时序数据中隐藏的数据质量规则. 对比实验的结果表明, 该方法能够有效地对行与列上的关联信息进行谓词构造; 在质量规则挖掘效果上优于单纯的行上约束挖掘方法以及单纯的列上约束挖掘方法.

关键词: 数据质量管理; 数据质量规则; 时序数据管理; 工业大数据

中图法分类号: TP311

中文引用格式: 丁小欧, 李映泽, 王晨, 王宏志, 李昊轩. 兼顾行列的时序数据质量规则发现. 软件学报, 2023, 34(3): 1065–1086. http://www.jos.org.cn/1000-9825/6793.htm

英文引用格式: Ding XO, Li YZ, Wang C, Wang HZ, Li HX. Time Series Data Quality Rules Discovery with Both Row and Column Dependencies. Ruan Jian Xue Bao/Journal of Software, 2023, 34(3): 1065−1086 (in Chinese). http://www.jos.org.cn/1000-9825/6793.htm

Time Series Data Quality Rules Discovery with Both Row and Column Dependencies

DING Xiao-Ou[1], LI Ying-Ze[1], WANG Chen[2], WANG Hong-Zhi[1], LI Hao-Xuan[1]

[1](School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China)

[2](National Engineering Research Center for Big Data Software (Tsinghua University), Beijing 100084, China)

Abstract: Time series data generated by intelligent devices are growing rapidly and faced with serious data quality problems. The demand for time series data quality management and data quality improvement based on data repairing techniques is increasingly urgent. Time series data has the obvious characteristics about the ordered time window and strong associations between rows and columns. This brings much more challenges for the research of the data quality semantic expression of time series data. This study proposes the definition and the construction of time series data quality rules, which takes into account the association on both rows and columns. It extends the expression of the existing data quality rule systems in terms of time window and multi-order expression operation. In addition, the discovery method is proposed for time series data quality rules. Experiment results on real time series data sets verify that the proposed method can effectively and efficiently discover hidden data quality rules from time series data, showing that the proposed method has higher performance with the predicate construction of associated information on row and column, compared with the existing data quality rule discovery method.

Key words: data quality management, data quality constraint, time series data management; industrial big data

* 基金项目: 国家自然科学基金(62232005, 62202126); 国家重点研发计划(2021YFB3300502); CCF-华为胡杨林基金数据库专项(CCF-HuaweiDB202204); 黑龙江省博士后资助项目(LBH-Z21137)

本文由“大数据治理的理论与技术”专题特约编辑杜小勇教授、杨晓春教授和童咏昕教授推荐.

收稿时间: 2022-05-16; 修改时间: 2022-07-29; 采用时间: 2022-09-23; jos 在线出版时间: 2022-10-27

随着数据采集技术的发展和数据存储能力的提升, 在工业、医疗、智慧城市等多领域积累了大量的数据. 这种被测量对象按特定时间(通常固定周期)持续测量记录的数值序列被称为时序数据. 在以工业互联网应用领域为典型代表的时序数据管理系统中, 由于传感设备采集数据的可靠性低、数据库系统入库标准不完善等原因, 时序数据中存在大量的数据质量问题. 这不仅导致数据预处理环节消耗大量人力成本, 也使得知识挖掘的计算结果发生严重偏差和错误, 最终影响依据数据分析与决策的精准性. 因此, 对时序数据建立完整、有效的质量管理体系, 是保证时序数据可用性的必要条件. 为了避免在“不完整”“不完美”“不精准”的劣质数据条件下进行数据活动, 需要在数据分析之前展开有效的数据质量管理工作.

相比于传统关系型的数据, 时序数据具有鲜明的数据特点, 总结为以下两个方面.

(1) 有序时窗. 时序数据具有时间标记, 数据记录之间存在显著的时序特性, 这种时间有序性是时序数据建模中重要因素之一. 在对时序数据的分析处理中, 相比于零散时刻上的数据, 更需要以时窗数据作为基本单位, 设计谓词进行有序窗口的集合计算.

(2) 时空关联性. 时序数据在时间段内以及多属性之间的数值均有较强的关联性, 相比于关系型数据, 时序数据在每条元组记录(即行)和每个属性(即列)的关联特性更为明显, 体现了物理对象之间的关联及函数运算关系以及数据在时间维度上的关联影响关系等.

时序数据具有的特点使其在时窗及时空关联性的质量规则语义表达需求更为复杂. 在构建时序数据质量表达机理时, 需充分考虑时窗和行列组合的约束特征, 这对已有的数据质量规则语义的表达力提出了更高的要求. 目前, 以函数依赖和否定约束为代表的行上约束难以实现对具有一定长度时窗内数据的质量要求表达; 另一方面, 以速度约束为代表的列上约束仅针对单条序列而设计, 难以描述多属性上的时序数据的质量要求.

基于此, 面对应用领域对高质量时序数据的迫切需求以及时序数据质量提升技术存在的挑战, 本文面向时序数据, 研究了能够兼顾行与列上关联的数据质量规则表达方法及规则挖掘问题, 主要贡献总结如下.

(1) 本文对以否定约束为核心的数据质量规则定义进行拓展, 提出了一种能够兼顾行与列上数据依赖关系的时序数据质量表达规则: 时序否定约束(temporal denial constraints, TDC), 实现了对时序窗口和谓词运算要素的表达, 并分析了时序否定约束推理系统的公理体系, 以解决目前时序数据质量规则表述不完备的问题.

(2) 本文提出并研究了 TDC 挖掘问题, 分析了 TDC 搜索空间规模, 并提出了 TDC 挖掘算法, 以数据驱动和简洁性为构建原则, 有效且高效地挖掘时序数据中行与列中潜在的数据质量规则模式, 以解决目前时序数据质量规则获取难、人工标注成本高的问题.

在两个真实的时序数据集上开展大量实验, 本文验证了所提出的方法能够有效地挖掘出时序数据中的质量规则. 与领先的数据质量规则挖掘方法的对比实验表明, 通过同时从行与列依赖信息计算表达式模式, 本文提出的TDC挖掘算法在时序数据上具有更优的规则挖掘效果, 实验结果和规则示例分析结果证实了兼顾行与列上依赖关系进行数据质量规则构建的必要性.

本文第 1 节为本文相关研究综述. 第 2 节介绍研究问题的预备知识和基本概念, 并简述本文内容的研究脉络. 第 3 节介绍本文提出的时序否定约束的表述方法及理论性质分析. 第 4 节介绍时序数据质量规则挖掘方法及示例分析. 第 5 节介绍本文方法的实验结果. 第 6 节进行全文总结及未来研究展望.

1 相关研究综述

规则违反问题是数据质量管理与数据清洗领域的核心难点问题之一, 目前形成了以数据可用性表达机理、数据质量判定和错误数据检测与修复的研究脉络[1]. 2019 年, Ilyas 等人在文献[2]中总结了数据清洗的典型流程, 介绍了数据转换、质量规则发现、错误数据检测、错误数据修复问题的技术脉络和前沿成果.

1.1 数据质量规则的发现与探究

由领域专家制定或者由领域业务流程总结得出一组能够准确而全面描述数据领域语义的完整性约束, 称为数据质量规则[1]. 基于规则的数据清洗技术采用领域特有知识, 充分挖掘数据的规律和关系来提升数据质

量, 其研究要点包括质量规则的发现与推理问题、基于规则的错误记录检测与修复问题等[2]. 早期, 研究人员提出了函数依赖(functional dependencies, FD)、条件函数依赖(conditional functional dependencies, CFD)等[3,4]语义规则, 以实现对数据一致性的有效表达. 规则挖掘方法旨在实现对数据中隐藏的有价值的规则模式进行发现提取[5,6], 主要分为模式驱动和实例驱动等策略. 较早被提出的 TANE 方法[7]即为一个典型的模式驱动的FD挖掘方法, 随后, 研究人员继续拓展了在CFD挖掘上的研究, 并相继提出了CTANE、CFDMiner、FASTCFD等方法[6,8−10].

2009 年, Golab 等人提出了顺序依赖(sequential dependencies, SD)[11]的概念, 以描述序列数据中连续数据点之间的数值差别要求, 但顺序依赖未考虑对时间戳的语义表达. 考虑到数据在时间属性上的质量要求, 针对数据库中的时间戳不完整的问题, Fan 等人提出了具有一阶逻辑语句的时效约束(currency constraints, CC)的概念[11,12], 在不依靠时间戳的情况下, 实现对数据库中属性、元组记录的时效性度量. 针对有时间戳的时序数据, 2016 年, Song 等人提出了速度约束(speed constraints, SC)[9]等概念, 用于发现单时间序列上不同变化幅度的单点错误数据. 2013 年起, 研究人员提出了否定约束(denial constraints, DC)[5]这一通用量化的一阶逻辑形式的规则, DC 既兼容 FD、CFD 模式, 又对其进行拓展, 实现了属性值之间的大于、小于的比对判断. DC 在表达力上提升也导致其挖掘问题计算空间的增长, 因此, DC 挖掘问题比 CFD 挖掘问题的难解性更高. Chu 等人提出了 FASTDC 的挖掘算法[5], Bleifub 等人提出了启发式的 Hydra 挖掘算法[13]. 2019 年起, 文献[14]提出了近似否定约束概念及其挖掘问题, 以避免规则挖掘中过拟合问题的发生, 并提出了挖掘算法 ADCMiner.

1.2 基于规则的时序数据清洗

2019 年, 综述文献[15]归纳并介绍了目前时序数据清洗研究进展, 基于统计模型的数据平滑和数据修复, 是时序数据清洗的主要方法. 相比于关系型数据, 时序数据尚未完全构建质量规则表达体系. 2016 年, 文献[16]聚焦时序数据质量管理, 从“行”与“列”的角度, 提出了单实体单属性、单实体多属性、多实体单属性、多实体多属性这 4 种规范化的数据质量规则类型, 并提出了时序数据质量管理通用模型. CFD 和 DC 都是单实体多属性类型的典型代表, 但单实体多属性的规则未能表达时窗数据的质量要求. 考虑到对时序的表达, 2015 年起, 文献[17]提出并解决了基于速度约束的时序数据清洗问题, 实现了对单序列上大幅度错误数据的修复, 这也是较早开展的基于规则的时序数据清洗研究. 在此基础上, Gao 等人提出了多区间速度约束的修复方法等[18], 提高了多实体单属性规则的实用性. 由于语义表达力上的局限, 目前, 基于多实体多属性规则的修复方法开展较少. 丁小欧等人和 Liang 等人提出了基于相关性的多维时序数据错误检测方法[19,20], 并针对 4 类数据质量规则类型提出了多规则的错误识别与诊断方法框架, 并提出了基于加权集合覆盖的求解方法.

1.3 小 结

目前, 在数据质量管理领域, 形成了以数据质量表达机理、数据质量判定、数据错误检测与修复为主线的理论和技术成果. 但已积累的基础理论和技术方法在时序数据质量管理上具有较大的局限性, 兼顾行列组合依赖的时序数据质量规则语义尚未完全建立. 已经建立的以条件函数依赖和否定约束为核心的数据质量规则体系对有序时窗和多属性数据间函数运算的表述具有一定的局限性. 而以顺序依赖、速度约束为例的质量规则仅针对单序列而设计, 时序数据质量规则语义尚未完全建立, 仍需对时序数据的特点以及时序数据质量要求进行分析, 深入、系统地研究时序数据质量管理问题.

2 研究问题介绍

2.1 基本概念

定义 1(时间序列). 时间序列是由传感器采样和捕获的一系列连续的数据点. 一条长度为 N 的时间序列表示为 $S=\langle s_1,s_2,\ldots,s_N\rangle$, 其中, 每个序列点表示为一个二元组 $s_i=\langle x_i,t_i\rangle$, x_i 是一个实数值, t_i 是时间记录点. 对于任意的整数 i, j, 若 $i<j$, 则有 $t_i<t_j$. $T=\{t_i\}_{i=1}^{N}$ 记作时间序列 S 的时间点集合. $\boldsymbol{S}$ 是一个包含 K 条具有相同时间点集合

T 的时间序列集合，记为 $\boldsymbol{S}=\langle s_1,s_2,\ldots,s_K\rangle$. $\boldsymbol{S}$ 称为 K 维时间序列.

正如第 1 节中提到: 否定约束是单实体多属性质量规则的典型代表，顺序依赖、速度约束则是多实体单属性规则的代表. 下面给出这 3 种约束的定义.

定义 2(否定约束). 给定关系 $R(A_1,A_2,\ldots,A_n)$的实例 I, t, t'是 I 上的两条元组，否定约束的表达式如下:

$$\forall t,t':\neg(P_1\wedge\ldots\wedge P_n),$$

其中，每个谓词 P_x 表示为 $v_1\theta v_2$ 或者 $v_1\theta c$, $v_1,v_2\in t.attr(R)\cup t'.attr(R)$, c 是取值在 $dom(A)$中的一个常量，运算符 $\theta\in\{=,\neq,>,\geqslant,<,\leqslant\}$. 称 R 的实例 I 满足一条否定约束φ, 如果至少一个谓词 P_x 不为真，记为 $I\vDash R$.

否定约束是一种形式为否定多个谓词同时存在的约束，表述了数据集中元组不能满足否定约束中的所有谓词. 例如，一条 DC 表示为$\forall t_i,t_j\in R$, $\neg((t_i[Comp]=t_j[Comp])\wedge(t_i[Sal]<t_j[Sal])\wedge t_i[Tax]>t_j[Tax]))$, 描述了两个在同一家公司就职的员工，收入更高者纳税更多.

顺序依赖(SD)[11]、速度约束(SC)[9]则是多实体单属性规则的典型代表，下面介绍顺序依赖和速度约束的定义.

定义 3(顺序依赖). $time$ 为时间轴, X 是数据的属性列. 如果数据 D 中$\forall i\in time$, $X_{i+1}-X_i\in g$, 即按照时间轴为序，相邻两个项的差值均落在 g 中，则称数据 D 符合顺序约束 SD: $time\rightarrow_g X$.

定义 4(速度约束). $s=(s_{\min},s_{\max})$: 已知窗口大小 w，最大速度 $s_{\max}$ 与最小速度 $s_{\min}$，称一个序列 x 满足 s，记为 $x\vDash s$，当在窗口内的 x_i, x_j，即 $0<t_i-t_j<w$，均有 $s_{\min}\leqslant\dfrac{x_j-x_i}{t_j-t_i}\leqslant s_{\max}$.

- 示例介绍

图 1 显示了某工厂水位传感器装置采集的一段数据，包含 4 个传感器，记为属性 A, B, C, D. 对历史数据进行分析，并结合专家经验，该工况下传感器的数据质量应满足如下要求: (1) 传感器 A 的水位不能超过 25 cm; (2) 传感器 B 在相邻时刻上水位变化差距在 4 cm 以内; (3) 传感器 C 在相邻时刻的水位变化速度不能超过 2 cm/s; (4) 传感器 D 的水位上涨或下降的增幅不能超过 2 cm/s^2; (5) 传感器 A 的水位不能比传感器 B 的水位高; (6) 传感器 C 的水位高度应低于传感器 A 与传感器 B 的水位总和; (7) 传感器 D 的变化速度应该小于传感器 C 的变化速度.

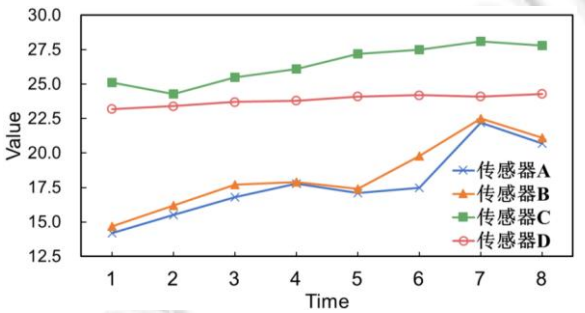


图 1 水温传感器数据片段

表 1 总结了利用已有的数据质量规则对上述要求进行形式化表述的情况.

表 1 数据质量规则形式化表述

数据质量要求	数据质量规则形式化表述	数据质量要求	数据质量规则形式化表述
描述(1)	值域约束: $A\leqslant 25$	描述(4)	加速度约束: $\left(\dfrac{\dfrac{D_{i+2}-D_{i+1}}{t_{i+2}-t_{i+1}}-\dfrac{D_{i+1}-D_i}{t_{i+1}-t_i}}{t_{i+2}-t_i}\right)\leqslant 1$
描述(2)	顺序依赖: $Time\rightarrow_{(-\infty,4)}B$	描述(5)	否定约束: $\forall t_i:\neg(t_i.A>t_i.B)$
描述(3)	速度约束: $\dfrac{C_{i+1}-C_i}{t_{i+1}-t_i}\leqslant 2$	描述(6)、描述(7)	无

可以看出, 分别利用值域约束、顺序依赖、速度约束和加速度约束对前 5 条描述进行表示, 但对数据质量表述(6)和表述(7)未能有效地表达. 观察发现, 表述(6)涉及同一条记录中不同属性之间的函数关系, 而表述(7)同时涉及行与列上数据取值要求. 面对这种情况, 本文的研究旨在找到一种能够兼顾行列的数据质量定义方法, 从而满足多维时间序列数据上的质量要求表达.

2.2 研究方法框架

针对时序数据的特点及时序数据质量表达方面研究的不充分性, 本文提出了一种能够兼顾行列语义表述的时序数据质量规则定义, 并研究了针对时序数据质量规则的挖掘算法. 研究内容如图2所示, 主要包括规则语义构建和自动挖掘算法两部分.

- 在时序数据质量规则语义构建部分, 本文要解决的问题是: 针对时序数据存在的有序时窗、数据持续、高速大量、行列关联等特点, 如何以已有的数据质量规则为基础, 拓展否定约束在时序数据上的表达力, 并保证时序否定约束体系的正确性和推理的完备性. 本部分在第 3 节进行介绍.
- 在时序数据质量规则自动挖掘算法部分, 本文要解决的问题是: 提出一种实现时序数据质量规则自动挖掘的算法, 并给出问题解空间规模的分析, 实现从大规模历史数据中发现并提取有效的质量规则知识, 从而有效地降低领域专家制定质量规则的人工成本. 本部分在第 4 节进行介绍.

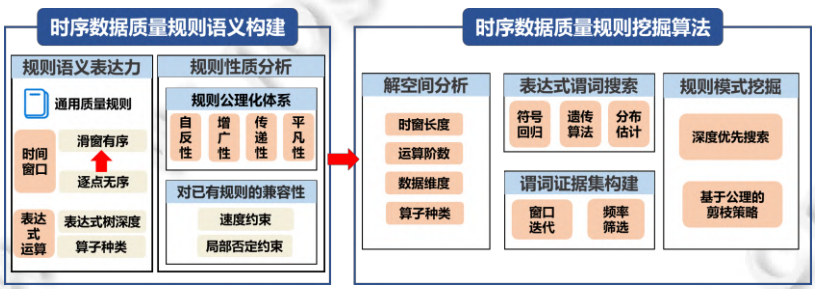


图 2 本文研究内容框架

3 时序数据质量表达规则语义构建

3.1 对已有关系数据质量表达机理的扩展

针对时序数据具有特点, 本文以关系型数据上否定约束和顺序依赖等已有的数据质量规则语义为基础, 提出可表达时窗内行列组合依赖的时序数据质量规则定义, 从时序窗口和谓词运算两方面对质量规则的表达力进行扩充完善.

定义 5(时序数据库中的表模式). 时序数据模式 $schema=(\boldsymbol{S}_0,B,p)$, 其中, $\boldsymbol{S}_0$ 为 K 维时间序列组成的时序数据分布集合, B:{=,≠,>,≥,<,≤}是一组有限的比较符. op 是一组运算算子, 定义为 op:{+,−,×,÷}. 如果 B 是自闭的, 则对 B 中任何比较符取反, 取反后的符号仍然在 B 中, 即 $\forall cmp\in B, \overline{cmp}\in B$.

(1) 时间窗口

对于时序数据的分析, 通常需利用窗口来衡量数据的局部特性. 因此, 本文对通用的数据质量规则引入时间窗口向量, 以此将关系表中任意两条元组(即时刻点上数据)的数据质量要求的表达扩展到任意两个时段之间(即时窗内数据), 并利用时间窗口向量, 保证数据间运算的有序性(即多列). 考虑由 N 条记录、维度为 K 的时序数据实例 $\boldsymbol{S}$、长度为 $WinSize$ 的滑动窗口, 我们定义 0 阶窗口表达式为

$$Exp^0\in\{M_t[i][j]|i\in\{0,1,\ldots,WinSize-1\},j\in\{0,1,\ldots,n_0\}\}.$$

(2) 表达式运算

由于时序数据的属性之间有很强的关联性, 仅用 B 中的 6 种比较符难以准确表述属性取值之间的关系. 因此, 本文对通用的数据质量规则引入运算阶数, 选取合适的有序集合运算操作, 将通用数据质量规则的谓词中属性取值的简单比对扩展到时间窗口内属性元素项的按阶运算后的比对. 首先, 通过选取若干 0 阶窗口

表达式 Exp^0 作为表达式树的叶节点，递归地构造深度为 k 的表达式树作为 k 阶窗口表达式 Exp^k. 对于算子 $\Delta \in op$, 实数集中常数 $const$、最大深度小于 k 的表达式树 v: $v \in \{Exp^m, const\}$ $(m \leqslant k-1)$, $j \in \{0,1,\ldots,n_0\}$, 定义 k 阶窗口表达式为: $Exp^k = Exp^{k-1} \Delta v (k \geqslant 1)$. Exp^k 即为最大深度为 k 的表达式树.

在时间窗口滑动计算的过程中，存在某些表达式结构一致且表达式的行下标的距离相同的情况. 例如，表达式 $Exp_i^1 : M_t[0][0] - M_t[0][1]$ 与表达式 $Exp_j^1 : M_t[1][0] - M_t[1][1]$ 都是用某一行的第 0 列的值减去第 1 列的值. 本文将这种现象定义为表达式的平移等价，见定义 6.

定义 6(表达式平移δ等价). 对于两个 k 阶表达式 Exp_i^k 与 Exp_j^k 以及给定的固定常数 δ, 当且仅当如下条件满足时，称这两个表达式平移 δ 等价，简写 $Exp_i^k =_\delta Exp_j^k$: (1) 如果 $k=0$, 则 Exp_i^k 与 Exp_j^k 要么是相等的常数，要么从滑动窗口矩阵 M_t 中选取出的元素应该满足行下标差距等于 δ, 即 $Exp_i^k.rowNum - Exp_j^k.rowNum = \delta$; (2) 如果 $k \neq 0$, Exp_i^k 与 Exp_j^k 这两棵表达式树需满足左子树和右子树均相互平移 δ 等价，即

$$Exp_i^k.l =_\delta Exp_j^k.l \text{且} Exp_i^k.r =_\delta Exp_j^k.r.$$

如果存在 δ, 使得两个表达式平移 δ 等价，则它们平移等价.

如图 3 所示，通过以滑动窗口中的元素作为叶子节点、多维时序表模式 S 中的运算符作为内节点，可以构造出图 3 右侧的 3 棵表达式树. 其中，表达式 $A_i - B_i$ 与表达式 $A_{i+1} - B_{i+1}$ 属于 Exp^1，并且它们是平移 δ 等价的 $(\delta=1)$; 而表达式 $(C_{i+1}-C_i)/(t_{i+1}-t_i)$ 是属于 Exp^2 的，这是因为其左子树和右子树均为 Exp^1 的表达式. 根据 k 阶表达式的定义，我们可以得出 $(C_{i+1}-C_i)/(t_{i+1}-t_i)$ 是 Exp^2，即 2 阶窗口表达式.

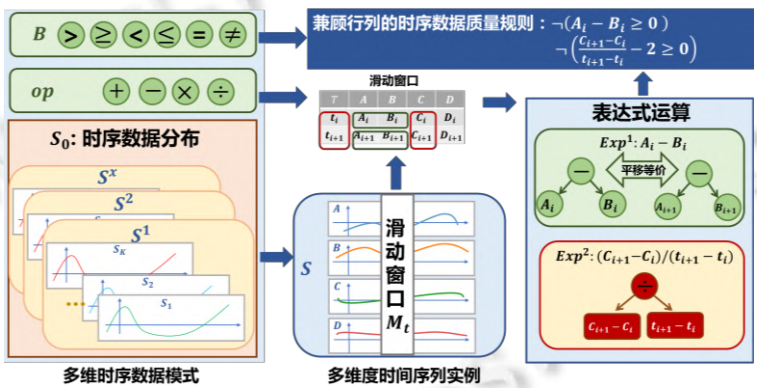


图 3 兼顾行列的时序数据质量规则构建过程

根据上述分析，本文在定义 7 中提出一种能够兼顾行列的时序数据质量规则表达方法.

定义 7(兼顾行列的时序数据质量规则 TDC). 对于给定的长度为 $WinSize$ 的时窗，一条兼顾行列的时序数据质量规则 φ 的形式为: $\neg(P_1^k \wedge P_2^k \wedge P_3^k \wedge \ldots \wedge P_n^k)$，其中，$P_i^k$ 为使用 k 阶运算下的窗口谓词. 时序数据库实例 D 如果满足时序数据质量规则 φ, 那么对于任意时刻 t 的滑窗 M_t, 在 φ 的所有谓词构成的集合 $\varphi.Pres$ 中，至少有一个谓词 P_i^k，满足 $P_i^k(M_t) = \text{False}$，记为 $D \vDash \varphi$; 同时，将 φ 称作为相对于 D 的一条有效的时序否定约束(简称为 TDC).

在定义 7 中，窗口谓词 P_i^k 是由滑动时窗矩阵 M_t 生成的 k 阶表达式 Exp^k. 对于比较运算符 $cmp \in B$, 组成的结构如 Exp^k cmp 0 的谓词，记为 $P_i^k = Exp^k$ $cmp\,0$. 从 B 中取出 cmp 的反 $\overline{cmp}$，可以构造出 P_i^k 的取反形式: $\overline{P_i^k} = Exp^k$ $\overline{cmp}$ 0. 显然，对于任何一个固定的滑窗实例 M_t, 由于 Exp^k 可以算得一个确定值，$P_i^k : Exp^k$ cmp 0 与 $\overline{P_i^k} = Exp^k$ $\overline{cmp}$ 0 不能同时成立. 对于谓词 P_i^k，存在一种谓词级别的蕴含关系. 即如果 P_i^k 成立，可以推理出一系列的谓词也成立. 对于这类蕴含性质，将 P_i^k 能够蕴含的谓词集合记为 $Imp(P_i^k)$. 如果 P_j^k 可以由 P_i^k 推理得出，则称 $P_j^k \in Imp(P_i^k)$. 由于引入了表达式结构，这类关系的判断无法通过简单的比较符号的种类来决

定, 本文将在第 4.2 节中详细介绍相关内容.

(3) 示例介绍

如图 3 右上所示, 利用 $\frac{C_{i+1}-C_i}{t_{i+1}-t_i}-2$ 以及 A_i–B_i 这两个表达式, 以及关系模式 S 中的比较运算符集合 B 构造出 $\frac{C_{i+1}-C_i}{t_{i+1}-t_i}-2\geqslant 0$ 以及 A_i–$B_i\geqslant 0$ 这两个窗口谓词, 可分别组成$\neg(A_i$–$B_i\geqslant 0)$以及 $\neg\left(\frac{C_{i+1}-C_i}{t_{i+1}-t_i}-2\geqslant 0\right)$这两条时序数据质量规则. 这正是在第 3.2 节中提到的数据质量描述(3)和描述(5).

表 2 中展示了利用本文提出的 TDC 实现对上述 7 条数据质量要求的描述. 可以看出, TDC 对已有规则难以描述的要求(6)和要求(7), 均实现了有效的表述.

表 2 兼顾行列的时序数据质量规则与已有质量规则表述的对比

数据质量要求	现有的数据质量规则	兼顾行列的时序数据质量规则 TDC
描述(1)	值域约束: $A\leqslant 25$	$\neg(A_i$–$25>0)$
描述(2)	序列依赖: $Time\rightarrow_{(-\infty,4)}B$	$\neg((B_{i+1}$–$B_i)$–$4\geqslant 0)$
描述(3)	速度约束: $\frac{C_{i+1}-C_i}{t_{i+1}-t_i}\leqslant 2$	$\neg\left(\frac{C_{i+1}-C_i}{t_{i+1}-t_i}-2>0\right)$
描述(4)	加速度约束: $\left(\frac{\frac{D_{i+2}-D_{i+1}}{t_{i+2}-t_{i+1}}-\frac{D_{i+1}-D_i}{t_{i+1}-t_i}}{t_{i+2}-t_i}\right)\leqslant 1$	$\neg\left(\frac{\frac{D_{i+2}-D_{i+1}}{t_{i+2}-t_{i+1}}-\frac{D_{i+1}-D_i}{t_{i+1}-t_i}}{t_{i+2}-t_i}-1>0\right)$
描述(5)	否定约束: $\forall t_i{:}\neg(t_i.A>t_i.B)$	$\neg(A_i$–$B_i>0)$
描述(6)	无	$\neg(C_i$–$(A_i$–$B_i)\geqslant 0)$
描述(7)	无	$\neg((D_{i+1}$–$D_i)$–$(C_{i+1}$–$C_i)<0)$

3.2 兼顾行列的时序数据质量规则的性质

本部分从理论上对所提出的TDC的公理化体系进行分析, 并类比已有的数据质量规则, 介绍TDC具有的性质.

3.2.1 规则公理化体系

推理系统 I 是一种在语法结构上检测一组规则能否逻辑蕴含另一组规则的有效方法. 如果系统是正确且完备的, 那么相对较为复杂的规则集逻辑蕴含判定问题, 即可转化为简单的规则集的闭包构造以及子集判定问题. 类比函数依赖和否定约束采用 Armstrong 公理体系进行蕴含分析[21,22], 本节对 TDC 推理系统的公理系统进行介绍. 首先, 在定义 8 中给出规则中逻辑蕴含的概念.

定义 8(TDC 的逻辑蕴含). 给定(时序数据)关系模式 $R(attr,\Sigma)$, $attr$ 为属性集, Σ为时序数据质量规则集. 若在 R 的任意关系实例 r 中, 一条 TDC φ始终成立, 则称Σ逻辑蕴含φ, 记作$\varphi\sqsubset\Sigma$.

概括地说, 本文提出的 TDC 以否定约束为基础, 在时窗和表达式运算两方面进行了表达力的拓展. 因此, TDC 与否定约束具有相似的总体结构及谓词形式. 但不同的是, TDC 能够实现对时序窗口中的任意 p 条记录(p 由谓词的阶数决定)进行比对, 并且每个谓词的单元变量 v 不再简单地采用某一行列上的属性值, 而是窗口中, 通过四则运算得到的最大深度为 k 的表达式结构 $v_i=Exp^k$. 接下来, 我们将介绍兼顾行列的时序数据质量规则 TDC 推理系统的 4 条公理.

平凡性. $\forall P_i,P_j$, 如果 $\overline{P_i}\in Imp(P_j)$, 那么$\neg(P_i\wedge P_j)$是平凡的 TDC.

增广性. 如果$\neg(P_1\wedge\ldots\wedge P_n)$是有效的 TDC, 那么$\neg(P_1\wedge\ldots\wedge P_n\wedge Q)$也是有效的 TDC.

传递性. 如果$\neg(P_1\wedge\ldots\wedge P_n\wedge Q_1)$与$\neg(R_1\wedge\ldots\wedge R_m\wedge Q_2)$都是有效的 TDC, 且 $Q_2\in Imp(\overline{Q_1})$, 那么$\neg(P_1\wedge\ldots\wedge P_n\wedge R_1\wedge\ldots\wedge R_m)$也是有效的 TDC.

自反性. 对于两条 TDC φ_1,φ_2, 存在一个固定的距离δ, $\forall P_m\in\varphi_1.Pres$, $\exists P_n\in\varphi_2.Pres$, 它们的表达式平移$\delta$等价, $P_m.Exp=_\delta P_n.Exp$. 那么, 若φ_1有效, 可推得φ_2有效.

对于本文提出的 TDC, 需从正确性和完备性两方面论证这种数据质量规则推理系统在理论上的严谨性.

TDC 推理系统的正确性应表明: 对于一个 TDC 集合Σ, 如果能够通过 I 的推理可以推导出一条 TDC φ成立, 那么Σ逻辑蕴含φ.完备性表明: 任何被Σ逻辑蕴含的φ都能通过推理系统 I 遵循一定的系统的步骤推理得出. 类比文献[5]对否定约束的推理系统的构建, 定理 1 说明了本文提出的 TDC 的推理系统的正确性和完备性.

定理 1. 本文提出的兼顾行列的时序数据质量规则公理体系是有效的, 并且对于满足结构 $\neg(P_1^k \wedge P_2^k \wedge P_3^k \wedge ... \wedge P_m^k \wedge Q)$ 的 TDC, 公理体系是完备的. 其中, $P_i^k = M_t[0][j] - M_t[0][j]$ cmp 0, $\forall i \in [1,m]$ 并且 $Q = M_t[0][k] - M_t[0][k]$ $\overline{cmp}$ 0, $cmp, \overline{cmp} \in B$.

证明: 受篇幅所限, 下面简述定理 1 的证明思路. 对于平凡性, 如果一个 TDC 存在两个不能同时为真的谓词, 那么对于任何滑动窗口实例, 它都将得到满足. 因此, 仅需证这两个谓词中必有一个为假, 满足平凡性的 TDC 将永远为真. 对于增广性, 通过证明增广后的谓词不影响行列依赖中合取式取假的情况, 进而增广后的 TDC 将对给定实例 r 中任意窗口实例都为真, 进而实现证明. 对于传递性, 如果存在两个 TDC 以及两个不能同时为真的谓词(每一个行列依赖各一个), 则可以通过去掉这两个谓词并合并这两个 TDC, 并得到一个新的有效的 TDC. 因此, 可推出传递后的规则中的谓词中一定有一个为假, 进而使整个规则为真, 进而完成证明. 在自反性方面, 对于两个 TDC φ_1, φ_2, 其中, φ_1 中所有的表达式都能通过对φ_2 的表达式在行上面平移相等的距离进而得到, 则它们是等价的. 其正确性的证明如下: 对于给定时窗 M_t, 利用表达式平移等价在为真的规则φ_1 中, 寻找使 $M_{t-\delta}$为假的谓词 P_{1i}, 代入φ_2 中对应平移等价的 P_{2j}, 根据平移等价的真值特性, $P_{2j}(M_t)$= $P_{1i}(M_{t-\delta})$=False, 进而使φ_2 整个规则为真实而得到证明.

而在完备性上, 可利用否定约束公理完备性进行证明[5]. 对于一条 TDC 在窗口为 1 时(即只有一行元组), TDC 和否定约束的谓词一一对应, 可根据否定约束完备性推理的路径, 构造出一个对应的由前 3 条公理产生的推理路径, 进而得到由$\Sigma \vdash_I \varphi$的推理路径. □

3.2.2 与已有规则的兼容性分析

下面从列和行两方面介绍所提出的时序数据质量规则对已有规则的兼容性.

(1) 对列上的约束兼容性

首先, 在定义 9 中形式化地归纳时序数据上的列关系约束.

定义 9(列关系约束). 给定窗口大小 *WinSize* 以及多维时序数据在某条序列投影得到的序列 S, k 阶时序列关系约束 $C_{col}(WinSize,k)$定义为: 如果在窗口 *WinSize* 中任取 m 个元素进行不超过 k 次四则运算后的值, 按照某一固定的模式计算特征后落在区间 $U=(u_{\min},u_{\max})$中, 则称序列 S 满足 C_{col}, 记为 $S \vDash C_{col}$.

根据定义 9, 我们在定理 2 中给出 TDC 与列关系约束存在的理论关系.

定理 2. TDC 可以兼容表达时序-列关系约束 C_{col}.

证明: 即证明: 给定任意一个固定的列关系约束 $C_{col}(WinSize,k)$, 存在由两条 TDC 组成的规则集Σ, 使得对任意时序实例 D: 如果 $D \vDash C_{col}$, 那么对于任意$\varphi \in \Sigma$, $D \vDash \varphi$; 如果 $D \nvDash C_{col}$, 那么一定存在$\varphi_0 \in \Sigma$, 使得 $D \nvDash \varphi_0$.

对于给定的列关系约束 $C_{col}(WinSize,k)$, 可构造一个和其对应的表达式结构 $Exp^m(m \leqslant k)$, 由这个特定的表达式结构派生出 P_{left}: $Exp^k - u_{\min} \leqslant 0$, P_{right}: $Exp^k - u_{\max} \geqslant 0$, 进而构造出两个 TDC 组成的$\Sigma$:$\{\neg P_{right}, \neg P_{left}\}$. 对于任何满足 $C_{col}(WinSize,k)$的数据, 需要同时满足按照给定模式计算的数据, 落在区间 $U=(u_{\min},u_{\max})$中, 即不可小于 $u_{\min}$, 也不可大于 $u_{\max}$, 那么$\neg P_{right}$,$\neg P_{left}$ 均成立. 即如果 $D \vDash C_{col}$, 那么对于任意$\varphi \in \Sigma$, $D \vDash \varphi$. 如果数据不满足 $C_{col}(WinSize,k)$, 可能的情况有以下两种: 数据大于 $u_{\max}$, 或者小于 $u_{\min}$, 那么 P_{left} 与 P_{right} 必有一个为真, 即 $\neg P_{right}$, $\neg P_{left}$ 必有一个为假. 即如果 $D \nvDash C_{col}$, 那么一定存在$\varphi_0 \in \Sigma$, $D \nvDash \varphi_0$.

综上, 对于任意给定的列关系约束, 均存在与之对应的兼顾行列的时序数据质量规则.

根据定义 9, 顺序依赖可表示为 *WinSize*=2、运算阶数 k 为 1 的列关系约束; 速度约束记为 *WinSize*=2、运算阶数为 3 的列关系约束; 加速度约束记为 *WinSize*=3、运算阶数为 9 的列关系约束(加速度的计算用了 6 次减法 3 次除法).

根据定理 1 知, 本文提出的 TDC 能够兼容表达以顺序依赖和速度约束为典型代表的列关系约束. □

(2) 对行上的约束兼容性

本节介绍TDC与行上约束的理论关系, 以否定约束为例, 本文提出的TDC能够兼容表达否定约束的一种局部特例, 见定义 10. 而这种局部否定约束的特性, 使得 TDC 的挖掘问题具有线性复杂度的优势, 进而更适用于时序数据的质量管理场景.

定义 10(局部否定约束). 给定关系 $R(A_1,A_2,\ldots,A_n)$的实例、距离常数θ, 局部否定约束ϕ的结构如下:

$$\forall t,t':\neg((t^1[A^1]cmp_1u^1[B^1])\wedge\ldots\wedge(t^n[A^n]cmp_nu^n[B^n])),$$

其中, $t^i,u^i\in\{t,t'\}$, $A^i,B^i\in\{A_1,A_2,\ldots,A_n\}$, $cmp^i\in\{=,\neq,>,\geqslant,<,\leqslant\}$. 元组间的最大距离小于$\theta$, 即 $dis(t,t')<\theta$. 实例 D 满足局部否定约束ϕ, 当且仅当至少有一个谓词 $t^k[A^k]cmp_ku^k[B^k]$不为真, 记作: $D\vDash\phi$.

定义 10 中, $dis(t,t')$是对数据实例按照某一个属性 A_i 进行排序后得到 t,t'的下标的差值. 显然, 本文示例中的数据质量描述(5)即可用局部否定约束所表示. 在验证一个长度为 N 的关系是否符合一条否定约束时, 往常需要逐一比对关系数据的表项, 即计算复杂度为 $O(N^2)$; 而局部否定约束规定了其对应的最大表项距离常数θ, 将比较表项对的次数减少至 $O(\theta N)$. 下面介绍本文提出的 TDC 能够兼容表达局部否定约束.

定理 3. 时序行列依赖可以兼容局部否定约束.

证明: 若要证明定理 3, 只需证明给定任意一个固定距离参数θ的局部否定约束ϕ, 存在由θ个对应的 TDC 组成的规则集Σ, 使得对任意时序数据实例 D: 如果 $D\vDash\phi$, 那么对于任意$\varphi\in\Sigma$, $D\vDash\varphi$; 如果 $D\nvDash\phi$, 那么一定有$\varphi\in\Sigma$, $D\nvDash\varphi$.

任取给定的局部否定约束ϕ, 其固定常数θ限定了元组搜索的范围, 故可取 $WinSize=\theta$, $k=1$, 构建Σ. 首先构造谓词, 对ϕ中每一个谓词 $t^m[A^m]cmp_mu^m[B^m]$, 在滑窗 M_t 中构造θ个对应的表达式谓词: $P_{mi}=(A^m[0]-B^m[i])$ cmp_m 0 (i=1,2,…,C;m=1,2,…,n)标识在 A^m 对应的列下, 滑窗的第 0 行元素. 由这些谓词构造和ϕ对应的由θ个对应的 TDC 组成的行列依赖集Σ:$\{\varphi_i|\varphi_i=\neg(P_{1i}\wedge P_{2i}\wedge\ldots\wedge P_{ni}),i\in\{1,2,\ldots,C\}\}$.

下面我们证明构造出的集合Σ与局部否定约束ϕ在表达能力上等价, 以下两种情况讨论.

- 如果 $D\vDash\phi$, 则ϕ中至少存在一个谓词 $t^m[A^m]$ cmp_m $u^m[B^m]$, 对于 D 中两个相距小于θ的表项, 均满足 $P_{\phi m}$:$t^m[A^m]$ cmp_m $u^m[B^m]$=False, 那么$\forall\varphi_i\in\Sigma$, 对于谓词: $P_{mi}=(A^m[0]-B^m[i])$ cmp_m 0, 根据ϕ的条件以及窗口大小可知, $i\leqslant\theta$, 根据 $P_{\phi m}$ 的条件知 P_{mi}=False. 即对于在 D 上的任意的滑动窗口, 该谓词都会返回 False. 将这个分析结果带入φ中, 得到 $D\vDash\varphi$, 即可得到: 任何满足局部否定约束的关系实例, 必然满足集合Σ.
- 如果 $D\nvDash\phi$, 则$\exists t_1,t_2\in D$, $\forall P_{\phi m}\in\phi$, $P_{\phi m}(t_1,t_2)$=True. 令 $x=dis(t_1,t_2)$, $(0\leqslant x<\theta)$. 对$\varphi_x\in\Sigma$, $\varphi_x=\neg(P_{1x}\wedge P_{2x}\wedge\ldots\wedge P_{nx})$, 利用$\varphi_x$ 对 D 进行检测, 扫描到包含 t_1, t_2, 且由 t_1, t_2 中的靠前者构成的滑动时窗 M_0 时, 由 $dis(t_1,t_2)=x$, M_0 的第 0 行与第 x 行即 t_1, t_2 所在的行. 由于$\forall P_{\phi m}\in\phi$, $P_{\phi m}(t_1,t_2)$=True, 知道由$(A^m[0]-B^m[x])$ cmp_m 0 作用在 M_0 也将会是 True. 因此, φ_x中的任意一个谓词 P_{mx}在这个特定的滑窗 M_0上都将是 True. 取非, $\varphi_x(M_0)$=False, 即对于 D 存在特定的时窗 M_0, 使得$\varphi_x(M_0)$=False. 即 $D\nvDash\varphi_x$, 进而证明: 如果 $D\nvDash\phi$, 那么一定存在$\varphi_0\in\Sigma$, $D\nvDash\varphi_0$.

综上, 我们构造出的Σ和ϕ的表达能力等价, 即任意一个固定常数θ局部否定约束ϕ, 均存在一个与之对应的 TDC. 进而证明了 TDC 可以兼容局部否定约束.

根据定理 3 可知, 可利用θ条 TDC 组成的规则集达到和局部否定约束等价的表达力. 随着 TDC 数目逐渐增多, 与之等价的局部否定约束的范围限制也更松弛. 当所限制的比对距离的参数趋于所有时序数据实例 D 的总长度 N 时, 局部否定约束将变回为通用否定约束. 因此, 如果允许 TDC 通过增大规则数到表的长度, 本文提出的 TDC 将完全和通用的否定约束具有等价的表达力, 即 TDC 完全兼容通用否定约束. □

4 时序数据质量规则自动挖掘方法

4.1 问题定义

参考条件函数依赖、否定约束挖掘问题的任务, 下面给出时序数据中 TDC 挖掘问题的定义.

定义 11(TDC 挖掘问题). 对于多维时序数据实例 D, 找到 D 上若干有效、最小且非平凡的 TDC 所组成的

规则集Σ.

这里对有效、非平凡和最小进行解释: 对于给定的数据实例 D 和滑窗 M_t, 如果$\forall M_t \in D$, $\varphi(M_t)$=True, 称φ是有效的 TDC, 记作 $D \vDash \varphi$; 如果对于任意数据 D'都有 $D' \vDash \varphi$, 则称φ是平凡的; 对于两条 TDC φ_1, φ_2, 如果不存在 $D \vDash \varphi_1$, $D \vDash \varphi_2$ 且$\varphi_2.Pres \subset \varphi_1.Pres$ 的情况, 则称φ_1是最小的, 其中, $\varphi.Pres$ 表示行列依赖φ的谓词集合.

4.2 规则解空间分析

在求解规则发现问题之前, 先对 TDC 的谓词空间规模进行分析. 设原数据为 K 维, 窗口大小为 $WinSize$, 表达式为 k 阶. k 阶的表达式树可以视为深度为 k+1 的二叉树, 其中, M_t 中的元素构成叶子节点, b_0 种算子构成非叶节点. 记 $f(k)$表示 k 阶表达式的种类数, 当 k=0 时, 有 $f(0)=|\{e|e$ 为零阶表达式 $Exp^0\}|=WinSize \times m$; 当 $k=k_0+1$ ($k_0 \geqslant 0$)时, 此时的 k 阶表达式树可被拆分为左右子树, 其种类数决定于左右子树的种类数. 将根节点左子树的数目 $f(k_0)$乘上右子树的数目 $f(k_0)$, 然后乘上中间节点算子的可能数目 b_0, 得到了 $f(k_0+1)=b_0 \times f(k_0)^2$. 综合 k=0 的情况, 向下递推即得到 $f(k)=b_0^{2^k-1} \times (WinSize \times K)^{2^k}$.

可以看出, $f(k)$是一个指数级的表达式空间. 在具体实例中, 对于一个窗口大小 $WinSize$=10、数据维度 K=6 的窗口, 假设算子为加减乘除 4 种, 则在 0 阶时, 所有可能的表达式一共有 $K \times WinSize$=6×10=60 种. 而在表达式为 4 阶时, 表达式空间的基数就达到了 $f(4)=b_0^{2^k-1} \times (WinSize \times K)^{2^k}=4^{2^4-1} \times (10 \times 6)^{2^4}=3.02 \times 10^{37}$ 种! 根据每个谓词对应一个表达式可以推出, 谓词空间的规模也为同一量级, 即 $\#(P)=O(b_0^{2^k-1} \times (WinSize \times n_0)^{2^k})$. 考虑到一个 TDC 通常由多个谓词组合而成, 因此, TDC 可能的解空间将是 $2^P : O(b_0^{2^k-1} \times (WinSize \times n_0)^{2^k})$. 根据上述分析的结果, 如何在如此庞大的解空间中进行表达式规则搜索和规则构建, 是一个有挑战的问题.

4.3 挖掘算法设计

如图 4 所示, 本文提出了兼顾行列的时序数据质量规则挖掘方法, 主要包括表达式机构搜索、谓词空间构造、证据集构造与计算、规则冗余消除以及规则评分等步骤.

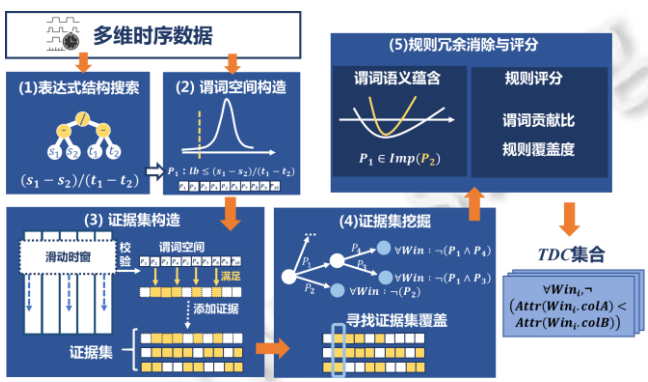


图 4 TDC 挖掘算法流程图

TDC 挖掘算法首先针对多维时序数据构造表达式谓词, 考虑到时序质量规则解空间的庞大, 不同于传统方法的暴力穷举所有可能的谓词结构, 本文利用遗传算法进行符号回归并生成表达式结构, 进而构造表达式结构所对应的谓词空间. 然后, 在训练数据上进行顺序滑窗扫描, 对每一个时窗向量, 计算其对应的谓词空间的证据, 并加入证据集合中. 和传统的规则挖掘方法的逐条比对来检验谓词不同的是, TDC 挖掘方法充分利用了滑动窗口的特性, 对历史数据进行一趟扫描即得到所需的证据集. 然后, 类比否定约束挖掘[5]方法, 通过深度优先搜索寻找证据集的最小覆盖, 进而得到候选的规则. 最后, 通过冗余消除算法及规则评分函数, 得到简洁且重要的高质量 TDC 集合, 整体算法流程如算法 1 所示.

算法 1. 自动化规则挖掘算法.

输入: 时序数据实例 *Data*.

输出: TDC 集合Σ.

1. $ExpSet \leftarrow$ ***ExpSearch***$(Data)$ //搜索表达式结构
2. $P^{space} \leftarrow$ ***PredicateSpaceFit***$(ExpSet,Data)$ //构造谓词空间
3. $EviSet \leftarrow$ ***EviMake***$(Data,P^{space})$ //构造证据集
4. $Q \leftarrow \phi$, $MC \leftarrow \phi$
5. ***RuleFind***(Q,Evi,P,MC) //搜索规则
6. ***Reduce***(MC) //对规则进行去重
7. $\Sigma \leftarrow$ ***Score***(MC) //对规则打分
8. **return** Σ

下面将详细介绍 TDC 挖掘算法的主要步骤.

4.3.1 表达式结构搜索

由于TDC挖掘问题面临的表达式空间的庞大, 因此需要一种高效的方法从指数级别扩张的表达式空间中自动地搜索出适用于时序数据的表达式. 这里, 我们提出表达式特征的自动化搜索需要遵循的两个基本原则:

(1) 数据驱动: 从时序数据中挖掘的规则模式应与数据所属领域的数据特点和性质相符. 例如, 对于电力数据场景, 表达式 $R=\frac{U}{I}$ 比速度 $V=\frac{s_i-s_{i+1}}{t_i-t_{i+1}}$ 要更适用.

(2) 简洁性: 挖掘出的表达式应具有结构上的简洁性. 比如, 表达式Δ_x=X_1−X_2 比 $f(x)=X_1-\frac{X_2}{X_1^5-X_1^5+1}$ 要易于理解, 尽管其都是反映了差分关系, 前者更具简洁特性.

基于以上两点原则, 我们采取了基于符号回归[23−25]的监督学习策略, 从历史数据中对表达式空间进行搜索, 根据数据的特点, 对不同表达式结构计算其适应度函数, 并对适应度函数引入裁剪系数, 以惩罚过长的表达式, 进而保证表达式结构的简洁性. 方法首先随机初始化若干个表达式树作为一个种群, 其中每一个表达式树作为种群的一个个体, 在给定的历史数据下, 每一个表达式树都有其自身的适应度. 迭代地计算表达式在给定样本上的适应度, 并选取出适应度较高的表达式进行交叉变异(交换子树(交叉)、改变内部运算节点的符号(变异)). 具体地, 将表达式结构按适应度得分从高到低排序, 选取前 n 个表达式结构继续执行搜索. 当迭代达到一定数目以后停止搜索. 本文的表达式搜索部分采取的策略是: 迭代地选取滑动窗口中的一个位置作为因变量 y, 而其他位置作为自变量$(x_1,x_2,\ldots,x_n)$. 构造自变量的表达式树以拟合表达式 $\tilde{y}=f(x_1,x_2,...,x_n)$. 在迭代完成以后, 将 $y-f(x_1,x_2,\ldots,x_n)$将作为表达式输出, 过程如图 5 所示.

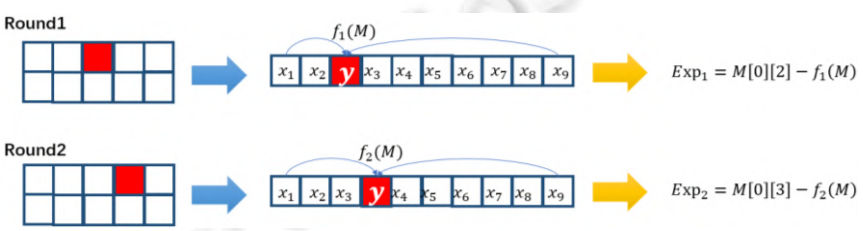


图 5 表达式搜索过程

综上, 表达式搜索过程如算法 2 所示.

算法 2. 表达式搜索算法 *ExpSearch*.

输入: 时序数据 D_{n_1*m}、窗口大小 *WinSize*、表达式阶数 k、采样个数 N_0、种群数 P_0、符号回归迭代轮数 r、优胜者数目 n_2.

输出: 表达式集合 $ExpSet\{\cdot\}$.

1. $ExpSet \leftarrow \phi$ //搜索表达式结构
2. $WinSample \leftarrow Sample(D,N_0)$ //从时序数据抽样

3. **for** i in *range*($Winsize$):
4. **for** j in *range*(m):
5. $y_0 \leftarrow \langle i,j \rangle$
6. $SymExp \leftarrow SymReg(WinSample, y_0, P_0, r, n_2, k)$ //符号回归求解表达式集合
7. $ExpSet \leftarrow ExpSet \cup SymExp$

在对表达式集合初始化后，使用水库抽样算法实现在线性时间内对于历史数据进行抽样进行均匀采样，得到一个历史数据的窗口样本. 在第 3–6 行，通过迭代对滑动窗口的每一个元素进行预测表达式结构,然后将预测出的表达式结构和因变量组合添加入表达式集合中. $SymReg(\cdot)$为基本的符号回归算法，其首先随机初始化一个最大深度为 k 的表达式树种群，然后在给定的样本上计算种群中个体的适应度 $fitness(Ind)$，然后不断地利用选择与变异操作使表达式种群尽可能地贴近数据. 这里，选用的适应度函数如公式(1)所示.

$$fitness(Ind) = -\frac{1}{m}\sum_{i=1}^{m}(Sample_i[r,c] - Ind(Sample_i))^2 - \lambda(length(Ind)) \tag{1}$$

其中，$Sample$ 为抽样数据，$\langle r,c \rangle$为目标位，λ是给定的裁剪系数，$length(Ind)$表示表达式的长度. 适应度函数由均方误差以及表达式长度惩罚项这两部分构成：均方误差函数使得表达式树的形状最终会越来越适应原始数据；后者通过乘上裁剪系数来惩罚过长的表达式结构，进而实现简洁性原则. 以均方误差作为适应度函数，表达式树的形状最终会越来越适应原始数据，并产生 $WinSize \times n_2 \times m$ 条表达式特征. 其中，n_2 为优胜者个数.

- 示例分析

以传感器 A, B 为例，下面展示在得到 t_9 传来的数据后，表达式的构造情况. 需要注意的是，考虑到真实的时序数据中可能存在噪声，因此在 t_9 时刻添加一组噪声数据，以展示本文方法在表达式计算及证据集构造方法所具有的鲁棒性. 具体见表 3.

表 3 传感器 A 和传感器 B 的序列片段

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
A	14.2	15.5	16.8	17.8	17.1	17.5	22.2	20.7	25.5
B	14.7	16.2	17.7	17.9	17.4	19.8	22.5	21.1	15.2

设窗口大小选择 2，并采样了 3 个滑动窗口，即 t_i={1,4,7}，t_{i+1}={2,5,8}. 算法将首先从历史数据中构造得到抽样后的历史窗口数据，然后分别以 t_i, A_i, B_i, t_{i+1}, A_{i+1}, B_{i+1} 为因变量，调用符号回归算法，以采样后的时窗作为训练集，得到对应的表达式结构. 以 A_i 作为因变量为例，在裁剪系数λ=0.1 的情况下，算法 2 将首先初始化若干表达式树的结构作为种群，可产生随机种群 $\left\{B_i - t_i, B_i \cdot A_{i+1}, \frac{t_i}{A_{i+1}}, t_{i+1}\right\}$；然后，对种群中的每一个体，将在采样后的滑动窗口中计算对应的适应度函数. 以 B_i–t_i 为例，其长度为 3，选取每轮迭代的 2 个优胜者进行交叉变异，其对应的适应度函数是 $-\frac{1}{3}\cdot(((14.7-1)-14.2)^2+((17.9-4)-17.8)^2+((22.5-7)-22.2)^2)-0.1\cdot 3=$ –20.417. 类似地，$B_i \cdot A_{i+1}$, $\frac{t_i}{A_{i+1}}$, t_{i+1} 这 3 个表达式的适应度分别为–10091.2, –310.78, –171.53. 然后，选取适应度高的前 2 个表达式 B_i–t_i 与 t_{i+1} 进行交叉变异，其适应度分别为–20.417 与–171.53，得到了新的种群 $\{B_i, B_i+t_i, B_i-t_i, t_{i+1}\}$,并继续计算适应度，选取优胜者 B_i 和因变量 A_i 共同组合成新的表达式：B_i–A_i. 重复上述步骤，可以得到遗传算法中的优胜者，如 B_{i+1}–B_i, t_{i+1}–t_i. 综上，我们搜索得到的表达式结构集合为：$\{B_{i+1}-B_i, t_{i+1}-t_i, B_i-A_i\}$.

- 表达式搜索的计算复杂度

对于一共有 N 条记录的 K 维数据 D，算法 2 的采样复杂度为 O(N). 第 3–7 行的二层循环中调用了符号回归 $Winsize \times K$ 次，其中，占据主要复杂度的是利用 $fitness$ 公式对种群中每一个个体，在采样数据中进行适应度的计算，其复杂度为 O($|P_0 \times S_0|$)，其中，S_0 为采样数目. 因此，根据迭代的轮数 r，符号回归的复杂度为

$O(r\times P_0\times S_0)$. 代入可得, 搜索的总复杂度为抽样复杂度和符号回归的复杂度的较大者: $\text{Max}\{O(Winsize\times K\times r\times P_0\times S_0), O(N)\}$. 对于小规模的数据, 符号回归占总运行时间的主要部分, 且运行时间和迭代轮数、种群数量、时窗大小成正比; 而对于大规模数据, 搜索的复杂性与数据规模成正比.

4.3.2 谓词空间构造

时序数据中数值型数据居多, 存在一定的概率分布, 因此需要借助统计学方法进行规则发掘. 例如, 文献[12]对速度的分布进行了统计, 利用非参估计的手段获得速度的近似分布, 然后对统计后的速度取 95%为限, 确定速度的边界. 本文将否定约束的谓词空间搜索和统计学方法相结合, 得到行列组合的谓词空间. 类比 SCREEN 方法, 采取非参估计(核密度估计)的手段得到窗口向量在某一维度上的分布, 然后根据给定置信度阈值, 得到在该维度上面的规则谓词区间.

- 示例分析

利用历史数据对 B_i-A_i 的表达式结构进行核密度估计. 首先, 由核密度估计得到其近似概率分布函数, 并对其进行积分转累计密度函数, 得到在 95%置信度下的边界约束阈值为: $-0.117\leqslant B_i-A_i\leqslant 2.474$. 因此构造出谓词: P_1: $(B_i-A_i+0.117)\leqslant 0$, P_2: $(B_i-A_i+0.117)>0$, P_3: $(B_i-A_i-2.474)\leqslant 0$, P_4: $(B_i-A_i-2.474)>0$, 如图 6 所示.

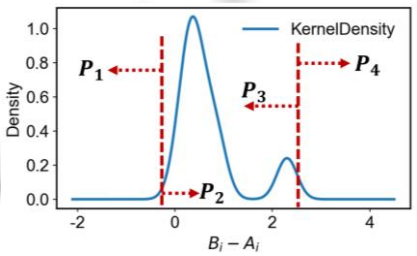


图 6 对表达式进行核密度估计

类似地, 对已得到的表达式结构集合$\{B_{i+1}-B_i, t_{i+1}-t_i, B_i-A_i\}$进行谓词空间构建, 得到:

P_1: $(B_i-A_i+0.117)\leqslant 0$ P_2: $(B_i-A_i+0.117)>0$ P_3: $(B_i-A_i-2.474)\leqslant 0$

P_4: $(B_i-A_i-2.474)>0$ P_5: $(t_{i+1}-t_i-1.0001)\leqslant 0$ P_6: $(t_{i+1}-t_i-0.9999)\leqslant 0$

P_7: $(t_{i+1}-t_i-1.0001)>0$ P_6: $(t_{i+1}-t_i-0.9999)>0$ P_9: $(B_{i+1}-B_i-4.3921)\leqslant 0$

P_{10}: $(B_{i+1}-B_i+2.834)\leqslant 0$ P_{11}: $(B_{i+1}-B_i-4.3921)>0$ P_{12}: $(B_{i+1}-B_i+2.834)>0$

4.3.3 谓词证据集构建

首先介绍证据集定义. 首先, 将时窗 M_t 上的数据的满足集记为: $SAT(M_t)=\{P|P\in \boldsymbol{P}, P(M_t)=\text{True}\}$, 其中, $\boldsymbol{P}$ 是谓词空间, $P(M_t)$=True 表示将 M_t 中数据代入 P 的结果为真. 基于此, 将时序数据 D 所有的滑窗对应的满足集组成的集合称为 D 的证据集, 即 $Evi_D=\{SAT(M_t)|\forall M_t\in D\}$.

在证据集构造过程中, 对给定长度为 N 的时序数据的每个时窗 M_{t0} 计算满足集 $SAT(M_{t0})$: 遍历谓词空间中的每一个谓词 P_i, 逐一验证时窗 M_{t0} 是否符合谓词 P_i, 如果符合, 则将谓词 P_i 加入 $SAT(M_{t0})$中. 在滑动窗口滑动一趟后, 得到 $N-WinSize+1$ 组满足集. 将所有满足集加入证据集中进行去重, 得到一个证据集的构造. 分析知: 对于给定长度为 N 的时序数据, 证据集构造复杂度为 $O(|P|\times(N-WinSize+1))$. 在实际中, N 通常远大于 $WinSize$, 因此上式可简化为 $O(N\times|P|)$, 反映了证据集构造的用时随数据量呈线性增长的趋势. 值得注意的是, 对于长 N 的数据以及谓词空间 $\boldsymbol{P}$ 下, 传统的否定约束证据集构造具有的 $O(N^2\times|P|)$复杂度; 相比而言, TDC 的证据集构造方法更适用于时序数据. 构造出的证据集可用于对谓词蕴含测试中正确但不完备的判定, 如果谓词 P_x 蕴含了谓词 P_y, 那么在 P_x 出现的满足集中 P_y 也会相应出现; 反之亦然. 因此, 在判定谓词 P_x 与 P_y 之间的蕴含关系时, 可以通过检查所有的满足集是否满足两个性质: (1) 所有包含 P_x 的满足集一定包含 P_y; (2) 所有不包含 P_y 的满足集一定不包含 P_x. 通过谓词蕴含测试, 我们能够对证据集中冗余的规则模式进行有效的去除.

- 示例分析

在得到谓词空间后, 通过滑窗计算得出证据集 $\boldsymbol{Evi}=\{\{\boldsymbol{P_2},\boldsymbol{P_3},\boldsymbol{P_5},\boldsymbol{P_8},\boldsymbol{P_9},\boldsymbol{P_{12}}\},\{\boldsymbol{P_1},\boldsymbol{P_3},\boldsymbol{P_5},\boldsymbol{P_8},\boldsymbol{P_9},\boldsymbol{P_{10}}\}\}$. 在设置证据集频率需至少大于 1 后, 得到过滤后的证据集: $\boldsymbol{Evi}_{\delta>1}=\{\{\boldsymbol{P_2},\boldsymbol{P_3},\boldsymbol{P_5},\boldsymbol{P_8},\boldsymbol{P_9},\boldsymbol{P_{12}}\}\}$.

4.3.4 证据集挖掘

本文提出一种启发式的深度优先搜索算法来执行证据集挖掘过程, 如算法 3 所示. 其主要步骤为通过基于证据集的覆盖度对节点排序, 并对每个节点维护 4 个关键变量: 当前搜索树中的路径 $Q(Q\subseteq P)$、未被覆盖的证据集 *Evi*、可以在后续搜索过程中包含的谓词集合 *P* 以及已经搜索到的覆盖集合 *MC*. 在谓词搜索过程中, 利用谓词在证据集上面的覆盖度对谓词进行排序, 形成一个优先队列, 然后逐一进入搜索(第 9、10 行). 覆盖度描述了该谓词出现在多少证据集中, 即 $Cov(P,Evi)=|P\in E|E\in Evi|$. 而 $P.order>Q.order$ 如果 $Cov(P,Evi)>Cov(Q,Evi)$. 如果没有可以覆盖的证据集, 且不存在比当前路径更短的路径能够完全覆盖证据集, 则终止搜索, 将当前搜索到的路径 *Q* 加入已经搜索到的覆盖集合 *MC* 当中. 如果目前的搜索深度超过阈值或找不到可用的谓词, 算法将直接返回. 算法 3 的搜索过程采用了一些剪枝策略. 对于证据集 *Evi* 中的每一个证据 e_i, 其在全时序窗口矩阵数据中出现的次数 $count(e_i)$取决于在证据集中满足的谓词数目的多少[5,14]. 不同的证据给 TDC φ的得分的贡献程度是不同的, 概括来说, 证据集中满足的谓词数目越多, 它对φ的贡献度就越大. 因此, 给定 TDC φ、证据 e_i, 若φ满足 e_i 中 *k* 个谓词, 那么则称 e_i 为 *k-evidence*(*k*-E), 其存在着贡献比ω_i, 如公式(2)所示.

$$\omega_i=\frac{k+1}{\#\varphi.Pres} \tag{2}$$

根据谓词的贡献比以及每一个证据在数据中出现的次数, 给定 TDC φ, 按照如下的公式计算规则φ的得分:

$$Score(\varphi)=\frac{\sum_{e_i} count(e_i)\times\omega_i}{\sum_{e_i} count(e_i)} \tag{3}$$

上述评分公式的直观理解即遍历证据集中的每一个证据, 根据贡献比作为证据的权重, 计算 TDC φ在多大程度上满足给定的训练数据. 算法第 12 行即根据公式(3), 根据规则φ对应的评分, 以确定在搜索树里面所采用的剪枝策略, 即如果一条路径的评分函数小于某一阈值, 对该路径进行减枝操作.

算法 3. TDC 证据集挖掘算法 *RuleFind*(*Q*,*Evi*,***P***,*MC*).

输入: 证据集 *Evi*、谓词空间 ***P***、目前的搜索路径 *Q*、最小覆盖集合 *MC*.

```
1.  if |Q|>threshold
2.      return
3.  if Evi 为空
4.      if 不存在大小为|Q−1|的 Q 覆盖 Evi:
5.          MC+=Q
6.  else if P 为空
7.      return
8.  else
9.      根据 Evi 上的覆盖度, 将 P 排序
10.     for P_add∈P:
11.         Q+=P_add
12.         if Score(Q)≤Bound:
13.             Q−=P_add
14.             continue
```

15. $E_{add}\leftarrow\{e|e\in Evi\wedge P_{add}\notin Evi\}$

16. $P_{add}\leftarrow\{P_0|P_0\in\boldsymbol{P}\wedge P_0\notin Q\}$

17. ***RuleFind***(Q,E_{add},P_{add},MC)

值得注意的是, 算法 3 的证据集挖掘方法与传统的否定约束的证据集挖掘方法有所不同. 算法 3 通过限定 DFS 搜索的深度来限制 TDC 的长度, 进而限制了搜索的范围. 这样做的原因在于: 一方面, 考虑到 TDC 的阶数 k 和窗口大小 *WinSize* 二者共同作用导致了谓词空间$|P|$的爆炸式增长, 因此需要更加注重算法的高效性; 另一方面, 考虑到过于长的 TDC 容易存在过拟合现象, 由奥卡姆剃刀原则, 本文更倾向于选择较小长度的 TDC 模式.

- 示例分析

若使用第 4.3.3 节中未筛选的证据集: $\boldsymbol{Evi}=\{\{\boldsymbol{P_2},\boldsymbol{P_3},\boldsymbol{P_5},\boldsymbol{P_8},\boldsymbol{P_9},\boldsymbol{P_{12}}\},\{\boldsymbol{P_1},\boldsymbol{P_3},\boldsymbol{P_5},\boldsymbol{P_8},\boldsymbol{P_9},\boldsymbol{P_{10}}\}\}$, 通过执行算法 3 计算证据集中的覆盖度, 如 $Cov(P_2,Evi)=2$, $Cov(P_3,Evi)=2$, 得到一个序关系: $P_3\geqslant P_5\geqslant P_8\geqslant P_9\geqslant P_{12}\geqslant P_{10}\geqslant P_1\geqslant P_2$. 然后, 基于此逐一遍历搜索谓词. 以 P_3 为例, E_{add} 置为空, 而 MC 中并没有比$\{P_3\}$更短的能够覆盖证据集, 因此将得到第 1 个候选规则 P_3. 类似地, 通过遍历证据集, 可以得到下面的最小覆盖 $MC=\{\{P_3\},\{P_5\},\{P_8\},\{P_9\},\{P_{12},P_1\},\{P_{10},P_2\}\}$. 然后对每一个覆盖中的谓词逐一去取非, 并用合取符号连接, 得到以下 6 条候选规则.

φ_1:$\neg((B_i-A_i-2.474)\leqslant 0)$ φ_2:$\neg((t_{i+1}-t_i-1.0001)>0)$

φ_3:$\neg((t_{i+1}-t_i-0.9999)\leqslant 0)$ φ_4:$\neg((B_{i+1}-B_i-4.3921)>0)$

φ_5:$\neg(((B_{i+1}-B_i+2.834)\leqslant 0)\wedge((B_i-A_i+0.117)>0))$ φ_6:$\neg(((B_{i+1}-B_i+2.834)>0)\wedge((B_i-A_i+0.117)\leqslant 0))$

而如果采用频率筛选之后的证据集: $\boldsymbol{Evi}_{\delta>1}=\{\{\boldsymbol{P_2},\boldsymbol{P_3},\boldsymbol{P_5},\boldsymbol{P_8},\boldsymbol{P_9},\boldsymbol{P_{12}}\}\}$, 则将会得到下面 6 条更精准的规则.

φ_1:$\neg((B_i-A_i-2.474)\leqslant 0)$ φ_2:$\neg((t_{i+1}-t_i-1.0001)>0)$

φ_3:$\neg((t_{i+1}-t_i-0.9999)\leqslant 0)$ φ_4:$\neg((B_{i+1}-B_i-4.3921)>0)$

φ_5:$\neg((B_{i+1}-B_i+2.834)\leqslant 0)$ φ_6:$\neg((B_i-A_i+0.117)>0)$

不难发现, φ_6 即印证了示例里面的领域专家总结出的规则: 传感器 A 的水位高度不超过传感器 B 的高度. 而φ_4 与φ_5 表明, 传感器 B 的相邻两时刻之间的水位变化差距在(2.8 cm,4.39 cm)之间.

4.4 对规则冗余去除的思考

时序数据质量规则在时间窗口和谓词运算方面对规则的表达力进行了扩展, 导致质量规则的模式空间变得复杂、庞大. 因此, 在执行挖掘算法的过程中, 需要从时窗长度、属性数量、运算阶数等方面对规则集进行合理的精简, 消除冗余规则, 进而提高挖掘结果的质量. 受篇幅限制, 本节简要概述我们在规则冗余去除方面的思考.

- 在语法层面, 基于上述已建立的公理化体系, 类比 FASTDC[7,8]中的蕴含测试算法, 设计了语法层面的蕴含分析算法对规则集的冗余性进行检测, 并利用规则摘要性指标减少质量规则的规模.
- 在语义层面, 主要解决窗口表达式谓词的蕴含测试问题.

例如, 对于以下谓词: $P_1:A_i^4-10\leqslant 0$, $P_2:A_i^2-20\leqslant 0$ 以及 $P_3:A_i^4-10<0$, 如图 7 所示, 符号<蕴含≤, 则有: 如果 P_3 成立, 则 P_1 必然成立, 即 $P_1\in Imp(P_3)$. 该蕴含关系可由第 3 节的规则公理推导得到. 而对于 P_2 和 P_3, 从图 7 中不难看出, 如果 f_3 成立, 那么对应的 f_2 也必然成立, 即推导出 $P_2\in Imp(P_3)$. 但是, 这种由表达式结构带来的语义蕴含问题难以通过简单的比较运算符进行确定, 此类语义冗余消除难题可通过两类方法进行解决.

1) 基于训练的近似冗余消除. 考虑到有蕴含关系的谓词往往具有相似的表达式结构, 因此可结合训练数据, 利用神经网络对规则中谓词的表达式结构进行嵌入表示. 由于相似的嵌入向量可能存在蕴含和被蕴含关系, 可训练一个分类器对规则间的蕴含性关系实现判定.
2) 基于规划方程的完备冗余消除. 利用规划方程, 将不同表达式结构的谓词的蕴含关系转换为在不同函数约束下的优化问题, 进而实现完备的蕴含判定.

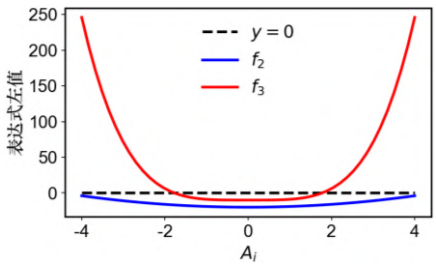


图 7 蕴含情况示例

5 实验分析

5.1 实验设定

(1) 数据集

本文实验主要选取了两个时序数据集.

- NASDAQ 数据集[26]: 从 1984-10-11 到 2022-02-11 的美国纳斯达克指数数据, 包含 6 个属性.
- WIND 数据集[19,27]: 某电厂引风机机组运行数据, 实验中分析了 40 个属性上 50 余万行数据记录.

数据集的划分策略采用留出法, 以数据的前 70%为训练集, 用于训练数据, 并产生分布; 并使用数据的后 30%作为测试集, 进行实验测试.

(2) 对比算法

本文实现了所提出的 TDC 自动挖掘方法的全部算法. 为客观验证本文方法的性能, 本文也实现了两种经典方法进行对比实验.

- 为了验证在列上质量规则发现效果, 本文复现了 SCREEN 中的速度约束[12], 其主要思想是: 通过统计时序数据速度分布, 并以速度分布的 95%为界, 建立速度的上下界.
- 为了验证在行上质量规则发现效果, 采用 NADEEF 开源项目中的挖掘算法 Hydra[13], 实现对数据中否定约束的挖掘.

(3) 计算指标

本文采用了错误数据检测任务的评价计算指标, 以能否有效地构建时序数据质量规则作为评测目标: 首先,利用 TDC 挖掘算法在训练集中挖掘得到 TDC 规则集; 然后, 将此规则集用于含有劣质数据的测试集上进行错误数据检测. 通过检测结果, 衡量 TDC 挖掘算法的语义构建能力. 如果检测结果与真实结果相符, 则将成功进行了语义构建记为真(True), 未成功进行语义构建记为假(False), 正常数据作为阴性(negatives), 错误数据则作为阳性(positives). 对于传统的否定约束, 选择数据集中的任意两条元组对作为一个实例单位, 对于速度约束, 选择适合计算速度的间隔为 2 的时间窗口作为一个实例单位. 对于本文提出的时序否定约束, 选择时窗中规则选用的元组对作为实例. 根据实例检测的结果, 可通过计算准确率、召回率、准确率、$F1$ 值来评价算法的性能.

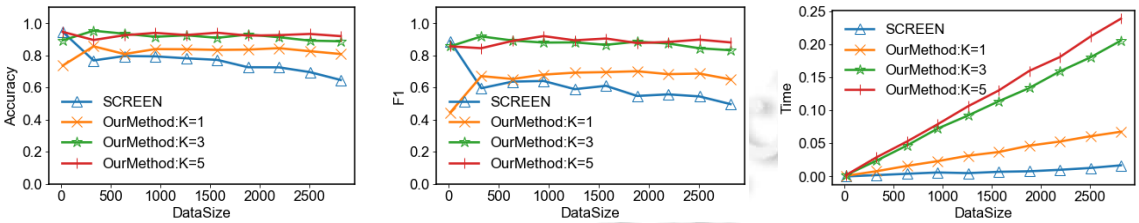
- 正确率 $Acc=(TP+TN)/(TP+TN+FP+FN)$.
- 准确率 $P=TP/(TP+FP)$.
- 召回率 $R=TP/(TP+FN)$.
- $F1$ 值: $F1=2PR/(P+R)$.

5.2 时序数据质量规则语义构建实验结果分析

本节首先在第 5.2.1 节和第 5.2.2 节中分别从列、行两方面介绍本文方法与已有方法对时序数据质量规则的构建效果, 然后在第 5.2.3 节介绍几组本文方法从两个数据集中挖掘的质量规则实例.

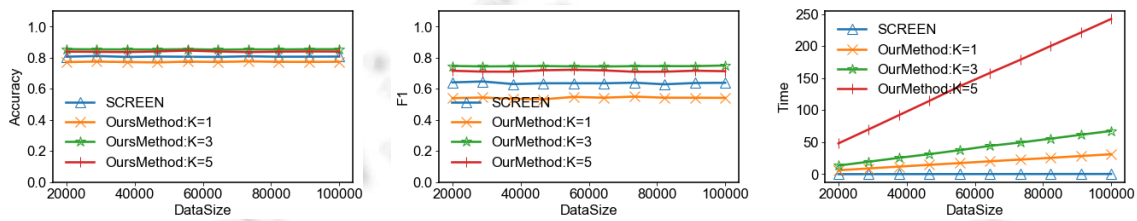
5.2.1 列上的语义构建

在本组实验中，我们将 TDC 挖掘算法的参数设置为：种群数量为 512，迭代两轮，并选取最后的 4 个优胜者. 本文方法与 SCREEN 方法采取的计算速度的时窗长度均为 2. K 指的是训练数据产生规则时，利用序列数据中的数据的维度数目. 图 8、图 9 分别介绍了在两个数据集上，这两种方法在正确率、$F1$ 值和运行时间上的结果对比.



(a) 正确率对比 (b) $F1$ 值对比 (c) 运行时间对比

图 8 NASDAQ 数据集上的结果



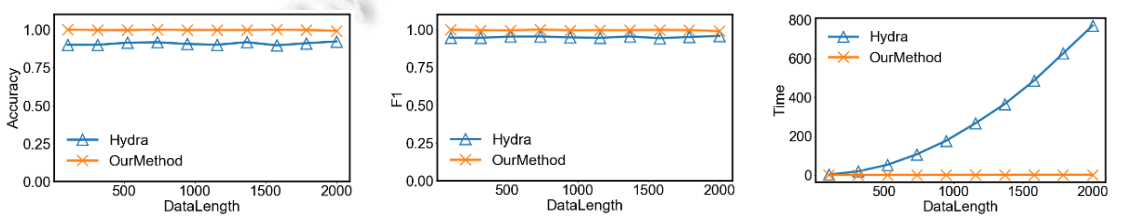
(a) 正确率对比 (b) $F1$ 值对比 (c) 运行时间对比

图 9 Wind 数据集上的结果

整体上，本文方法在 $F1$ 和正确率指标上取得了比 SCREEN 方法更有的效果. 这是由于相较于 SCREEN 方法仅使用了单一的表达式谓词结构，本文提出的 TDC 同时考虑了行与列上数据的依赖关系，采用更丰富的表达式谓词结构，并在证据集构建时，对不合理的表达式结构进行了筛选排序，因此得到了更可靠的规则构建效果. 相比而言，两种方法在 Wind 数据集上规则构建的效果比在 NASDAQ 数据集上更为稳定. 这是由于 Wind 数据集的测试集规模有几十万条数据，而股票类数据的测试集规模较小，总测试条目数仅有两千余条，受随机因素影响较大. 综合的来看，本文方法对时序数据列上的质量规则语义构建效果高于 SCREEN 方法 30%左右，但在运行时间上比 SCREEN 方法用时更长. 这是由于本文方法不仅只构建列上的规则，而是计算得到兼顾行列的谓词表达式，因此需要更多的用时.

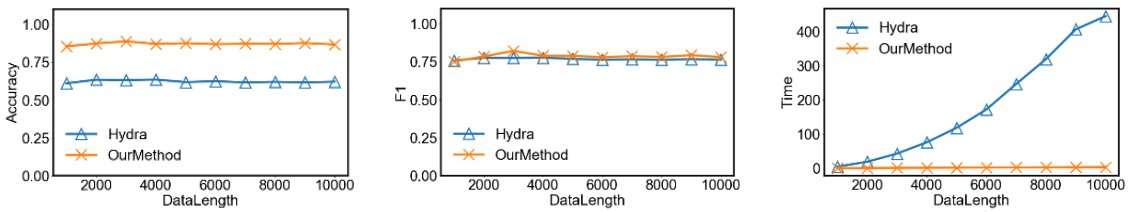
5.2.2 行上的语义构建

下面介绍本文方法与 Hydra 算法在行上质量规则的构建效果. 在两个数据集上，均选取 6 个属性进行测试，并控制数据长度作为自变量. 本文方法中，固定种群数量为 512，迭代两轮，并选取遗传算法中的最后 2 个优胜者. 实验结果如图 10、图 11 所示.



(a) 正确率对比 (b) $F1$ 值对比 (c) 运行时间对比

图 10 NASDAQ 数据集上的结果



(a) 正确率对比 (b) $F1$ 值对比 (c) 运行时间对比

图 11 Wind 数据集上的结果

在语义构建效果方面，两种方法对数据规模的增长均表现出稳定的效果，且本文方法比 Hydra 算法的效果更优. 这是由于与 Hydra 算法相比，TDC 挖掘算法能够利用局部时窗的信息，并设置了证据集的频率筛选条件，其抗干扰能力较强，不易受到数据集中的扰动的影响. 因此，TDC 挖掘算法得到的规则可信度和质量更高. 和第 5.2.1 节的构建效果相比，行上规则的构建效果更高. 通过分析发现，本文方法在多维时序数据上能够利用不同属性间的行上的规则信息，能够辅助更精准地构建列上规则，体现了“1+1>2”的效果，进一步验证了兼顾行列构造数据质量规则具有的重要价值.

在运行时间方面，本文的方法效率要明显高于 Hydra 算法. 随着数据规模的增长，Hydra 算法呈平方级的增长趋势. 这是由于该方法挖掘否定约束时需两两进行比对表项，而本文方法只需检查时窗的局部信息，具有线性时间复杂度的优势. 实验结果印证了第 4.2 节中的理论介绍，也证实了本文提出的 TDC 比朴素的否定约束在语义上更适用于对时序数据的质量表达.

5.2.3 规则语义构建实例介绍

表 4 展示了本文提出的 TDC 挖掘算法针对 Wind 与 NASDAQ 数据集挖掘出的规则实例.

表 4 挖掘得到的数据质量规则示例

数据集	TDC 实例	描述
NASDAQ	$\neg(High_i-Close_i+0.027\leqslant 0)$	股市当日最高价格一般高于当日收盘价
	$\neg(Open_{i+1}-Open_i-338.95\geqslant 0)$	相邻两天的开盘价变化一般不超过 338.95
	$\neg(AdjClose_i-Close_i-0.22\geqslant 0)$	调整后的收盘价格和收盘价满足一定的差分关系
Wind	$\neg(2\cdot Force_i^{out}-Force_i^{in}-8.27\leqslant 0)$	风机的输入风压近似大于输出风压的 2 倍
	$\neg((Force_i^{out}-Force_{i+1}^{out}-0.64\geqslant 0)\wedge (Force_i^{in}-Force_{i+1}^{in}-0.074\geqslant 0))$	输入和输出风压变化幅度有一定的范围要求
	$\neg((Power_i-(Force_i^{out}-Force_{i+1}^{in})^2\leqslant 1526.3)\wedge (Power_i\geqslant 0))$	在发电机输出功率大于 0 时，其输出公率与输入输出风力差值的平方相关

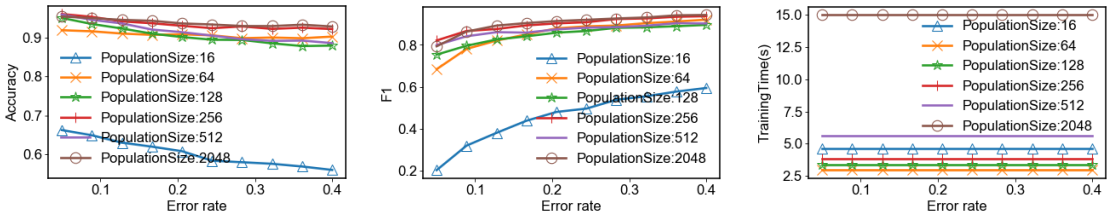
不难发现，本文提出的 TDC 挖掘算法能够有效地找到时序数据中隐藏的质量规则；同时，TDC 具有较丰富的表达力，能够有效地兼容表达序列依赖和部分否定约束. 例如，对于否定约束$\forall t_\alpha\in D$, $\neg(t_\alpha.Open>t_\alpha.High)$的质量规则语义，实际上已经被 TDC: $\neg(High_i-Close_i+0.027\leqslant 0)$所兼容，其中的 0.027 是由于分布密度估计取 95%所带来的残余. 此外，我们发现，TDC 算法更倾向于挖掘形如差分关系的表达式. 后续可通过对 TDC 挖掘过程采用的遗传算法进行改进，以提升其对更复杂结构表达式的有效挖掘.

5.3 时序数据质量规则自动挖掘实验结果分析

本部分探究了多个关键参数对自动化挖掘算法结果的影响. 为了确保结果的可靠性，采取重复 10 次取各项指标的平均值的手段，以消除符号回归中遗传算法随机性带来的影响.

5.3.1 种群数量对自动化挖掘算法的影响

我们在这一节探究种群数量对规则自动发现质量的影响. 我们选取 NASDAQ 数据集的 6 个属性，采取控制变量的原则，固定窗口大小为 2，并设定迭代轮数为 4，得到实验结果如图 12 所示.



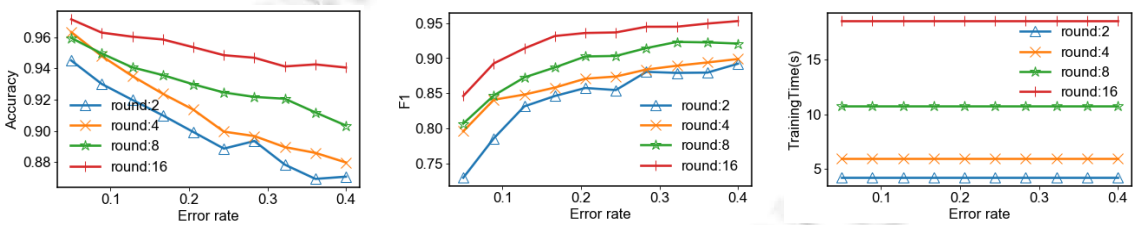
(a) 正确率 (b) $F1$ 值 (c) 运行时间

图 12 种群数量对算法性能的影响

从实验中可以看出，增加种群的数量，在给定的迭代轮数下，模型的性能得到提升. 这反映出种群数量大小和规则构建能力的强弱直接相关. 在种群规模较小时，其多样性较低，所产生的表达式结构难以贴近数据本身的结构，只能由几个在训练集中适应较好的表达式个体去不断地交叉进化，表达式结构容易在训练集中过拟合. 而在种群规模增大以后，随着种群的多样性丰富，更贴近数据本身特点的个体将迅速“脱颖而出”，主导整个种群的进化方向，进而得到适合数据领域特点的表达式结构. 不容忽视的是，更大的种群数目导致所需的计算代价增大，因此需要更多的预算来迭代寻找表达式结构. 因此，种群数量和预算这两个因素,将在方法部署时需进行权衡考虑.

5.3.2 迭代轮数对自动化挖掘算法的影响

本节介绍迭代轮数对规则自动挖掘效果的影响. 我们采取控制变量的原则，选取 NASDAQ 数据集的 6 个属性，固定窗口大小为 2，并设定种群数量为 128，得到实验结果如图 13 所示.



(a) 正确率 (b) $F1$ 值 (c) 运行时间

图 13 迭代轮数对算法性能的影响

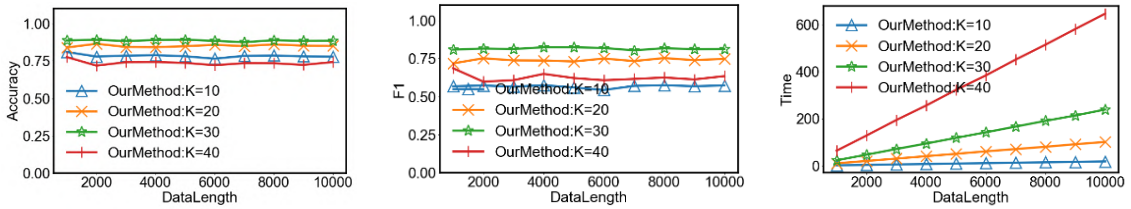
从实验中可以看出，固定种群数目并增加迭代轮数，模型逐步提升. 这是由于迭代轮数越多，符号回归算法趋于收敛，挖掘到的规则的质量也相对较高. 图 13 展示了当迭代轮数从 4 增长到 16 时，正确率和 $F1$ 值均有大幅度提升. 而另一方面，在相同的预算下，增加迭代轮数带来的挖掘质量上的收益并没有增加种群规模的收益大. 但是增大迭代轮数可以使算法趋于收敛，挖掘到更多高质量的规则. 对于这种现象，我们认为，这是由于时序数据的表达式搜索空间庞大，如果小种群没有在初期覆盖适合数据的表达式结构，方法则需要不断地遗传进化去搜索表达式结构，每一轮的进化需要遍历所有训练数据计算适应度，从而增加了时间消耗. 而对于大种群，合适的表达式结构出现较早，因此在很短的时间内即可得到符合原始数据的表达式结构.

5.3.3 属性个数对自动化挖掘算法的影响

我们在 Wind 数据集上测试属性个数对挖掘算法的影响. 固定迭代轮数为 4，并选取种群大小为 512，窗口大小为 2，实验结果如图 14 所示.

从实验中可以看出，挖掘算法的正确率和 $F1$ 值出现随着属性个数增加而先增而后下降的趋势. 当属性个数从 10 增长到 30 时，本文的挖掘方法能够利用更多的维度间的信息进行错误数据的检测，进而能够达到更好的检测结果. 而当属性个数从 30 增加到 40, $F1$ 值有所下降. 这是由于此时表达式空间已经较大，而固定的种群个数无法实现对整个表达式空间的有效搜索. 因此，本方法的语义构建的能力出现了一定程度的下降. 本文提出的 TDC 能够有效地捕捉多属性维度之间的关联关系，因此随着属性个数的增加，对潜在的表达式搜

索次数增多，进而导致了方法整体运行时间的增长.

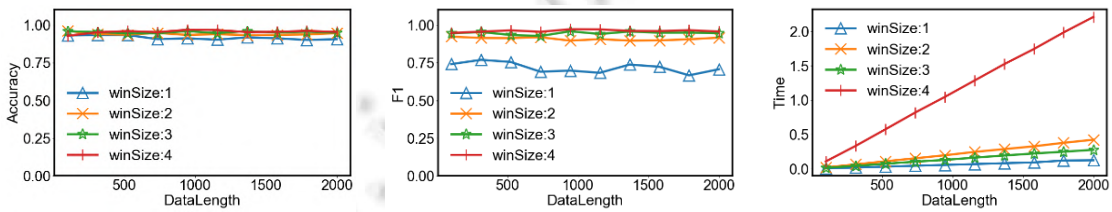


(a) 正确率 (b) $F1$ 值 (c) 运行时间

图 14 属性个数对算法性能的影响

5.3.4 时窗大小对自动化挖掘算法的影响

固定迭代轮数为 4，并选取种群大小为 128，本节介绍 NASDAQ 数据集上窗口大小对规则自动挖掘效果的影响，如图 15 所示.



(a) 正确率 (b) $F1$ 值 (c) 运行时间

图 15 时窗大小对算法性能的影响

从实验中可以看出，从窗口大小为 1 增大到 2，给方法带来的性能增益要大于从 2 增大到 4 的性能增益. 因此，对于 NASDAQ 数据集来说，时窗设置在 2–4 比较合适. 在窗口大小为 1 时，方法仅能构建单一行上的规则，例如$\neg(High_i-Close_i+0.027\leqslant 0)$，此时能够用于产生表达式的信息较少，极易出现表达式的过拟合现象，导致了较差的构建效果. 而当时窗大小设置为 2 及以上时，方法即可同时利用行上和列上的关系信息,进而产生更多有意义的规则，也极大地缓解了过拟合现象的发生. 从实验数据中发现，当时窗达到一定阈值时，时窗规模将不再给方法带来明显增益. 这是由于随着时窗增大，规则之间更易出现平移等价性的情况，不再提升规则构建的贡献度，导致了冗余的出现. 这些冗余模式会被算法 3 所去除，进而得到更精炼的 TDC 集合. 表 5 介绍了在不同窗口大小下，通过执行算法 3 所去除的具有平移等价的冗余规则数量. 需要说明的是，不同数据集所具有的最合适的时窗大小存在很大差异，如何针对给定的时序数据集，找出最优的时窗大小，是未来需要继续探索的问题.

表 5 不同时窗下的冗余消除情况

窗口大小	被冗余消除的平移等价规则数目(平均)
1	0
2	13
4	95
8	142

6 总结和展望

本文研究了一种兼顾行列关系的时序数据质量规则 TDC 的定义方法，从时窗与表达式多阶运算两方面，对已有的数据质量规则体系进行补充，并论证了 TDC 推理系统的正确性和完备性以及 TDC 对已有的数据质量规则表达力上的兼容性；提出了时序数据质量规则挖掘方法，对于给定的时窗数据，有效地搜索表达式结构,并构造谓词空间，通过对证据集的计算得出高质量的 TDC 集合. 真实的时序数据集上的实验表明，本文提

出的同时考虑行与列上数据关联性的时序数据质量规则, 比单纯的行上约束或单纯的列上约束具备更有效的表达力. 所提出的 TDC 挖掘算法能够有效且高效地发现时序数据中潜在的数据质量要求模式. 未来的研究方向包括: (1) 近似 TDC 的定义及挖掘问题; (2) TDC 的谓词空间冗余消除问题; (3) 时窗规模自动估计问题等.

References:

[1] Li JZ, Wang HZ, Gao H. State-of-the-art of research on big data usability. Ruan Jian Xue Bao/Journal of Software, 2016, 27(7): 1605−1625 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5038.htm [doi: 10.13328/j.cnki.jos.005038]

[2] Ilyas IF, Chu X. Data Cleaning. ACM, 2019. 150−194.

[3] Ilyas IF, Chu X. Trends in cleaning relational data: Consistency and deduplication. Foundations and Trends® in Databases, 2015, 5(4): 281−393.

[4] Fan WF, Geerts F. Foundations of Data Quality Management. Morgan & Claypool Publishers, 2012. 13−68.

[5] Chu X, Ilyas IF, Papotti P. Discovering denial constraints. Proc. of the VLDB Endowment, 2013, 6(13): 1498−1509.

[6] Pena EHM, de Almeida EC, Naumann F. Discovery of approximate (and exact) denial constraints. Proc. of the VLDB Endowment, 2019, 13(3): 266−278.

[7] Huhtala Y, Kärkkäinen J, Porkka P, Toivonen H. TANE: An efficient algorithm for discovering functional and approximate dependencies. The Computer Journal, 1999, 42(2): 100−111.

[8] Chiang F, Miller RJ. Discovering data quality rules. Proc. of the VLDB Endowment, 2008, 1(1): 1166−1177.

[9] Song SX, Zhang AQ, Wang JM, Yu PS. SCREEN: Stream data cleaning under speed constraints. In: Proc. of the ACM SIGMOD Int'l Conf. on Management of Data. ACM, 2015. 827−841.

[10] Fan WF, Geerts F, Li JZ, Xiong M. Discovering conditional functional dependencies. IEEE Trans. on Knowledge and Data Engineering, 2010, 23(5): 683−698.

[11] Golab L, Karloff H, Korn F, *et al*. Sequential dependencies. Proc. of the VLDB Endowment, 2009, 2(1): 574−585.

[12] Fan WF, Geerts F, Tang N, *et al*. Conflict resolution with data currency and consistency. Journal of Data and Information Quality (JDIQ), 2014, 5(1-2): 1−37.

[13] Bleifuß T, Kruse S, Naumann F. Efficient denial constraint discovery with hydra. Proc. of the VLDB Endowment, 2017, 11(3): 311−323.

[14] Livshits E, Heidari A, Ilyas IF, *et al*. Approximate denial constraints. Proc. of the VLDB Endowment, 2020, 13(10): 1682−1695.

[15] Wang X, Wang C. Time series data cleaning: A survey. IEEE Access, 2019, 8: 1866−1881.

[16] Dasu T, Duan R, Srivastava D. Data quality for temporal streams. IEEE Data Engineering. Bulletin, 2016, 39(2): 78−92.

[17] Zhang AQ, Song SX, Wang JM, *et al*. Time series data cleaning: From anomaly detection to anomaly repairing. Proc. of the VLDB Endowment, 2017, 10(10): 1046−1057.

[18] Gao F, Song SX, Wang JM. Time series data cleaning under multi-speed constraints. Ruan Jian Xue Bao/Journal of Software, 2021, 32(3): 689−711 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6176.htm [doi: 10.13328/j.cnki.jos.006176]

[19] Ding XO, Yu SJ, Wang MX, Wang HZ, Gao H, Yang DH. Anomaly detection on industrial time series based on correlation analysis. Ruan Jian Xue Bao/Journal of Software, 2020, 31(3): 726−747 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5907.htm [doi: 10.13328/j.cnki.jos.005907]

[20] Liang Z, Wang HZ, Ding XO, Mu TY. Industrial time series determinative anomaly detection based on constraint hypergraph. Knowledge-based Systems, 2021, 233: Article No.107548.

[21] Baudinet M, Chomicki J, Wolper P. Constraint-generating dependencies. Journal of Computer and System Sciences, 1999, 59(1): 94−115.

[22] Abiteboul S, Hull R, Vianu V. Foundations of Databases. Addison-Wesley, 1995.

[23] Schmidt M, Lipson H. Distilling free-form natural laws from experimental data. Science, 2009, 324(5923): 81−85.

[24] La Cava W, Orzechowski P, Burlacu B, de França FO, Virgolin M, Jin Y, Kommenda M, Moore JH. Contemporary symbolic regression methods and their relative performance. In: Proc. of the 35th Conf. on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1). 2021. https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/file/c0c7c76d30bd3dcaefc96f40275bdc0a-Paper-round1.pdf

[25] Virgolin M, Alderliesten T, Witteveen C, *et al*. Improving model-based genetic programming for symbolic regression of small expressions. Evolutionary Computation, 2021, 29(2): 211–237.

[26] https://finance.yahoo.com/quote/%5EIXIC?p=^IXIC&.tsrc=fin-srch

[27] Li ZJ, Ding XO, Wang HZ. An effective constraint-based anomaly detection approach on multivariate time series. In: Wang X, Zhang R, Lee YK, Sun L, Moon YS, eds. Proc. of the 4th APWeb-WAIM Joint Int'l Conf. on Web and Big Data. LNCS Vol.12318, Cham: Springer, 2020. 61–69. [doi: 10.1007/978-3-030-60290-1_5]

附中文参考文献:

[1] 李建中, 王宏志, 高宏. 大数据可用性的研究进展. 软件学报, 2016, 27(7): 1605−1625. http://www.jos.org.cn/1000-9825/5038.htm [doi: 10.13328/j.cnki.jos.005038]

[18] 高菲, 宋韶旭, 王建民. 多区间速度约束下的时序数据清洗方法. 软件学报, 2021, 32(3): 689−711. http://www.jos.org.cn/1000-9825/6176.htm [doi: 10.13328/j.cnki.jos.006176]

[19] 丁小欧, 于晟健, 王沐贤, 王宏志, 高宏, 杨东华. 基于相关性分析的工业时序数据异常检测. 软件学报, 2020, 31(3): 726−747. http://www.jos.org.cn/1000-9825/5907.htm [doi: 10.13328/j.cnki.jos.005907]

丁小欧(1993－), 女, 博士, 助理教授, CCF 专业会员, 主要研究领域为数据质量, 数据清洗, 时序数据管理.

李映泽(2001－), 男, 本科生, 主要研究领域为时序数据质量管理, 数据库.

王晨(1981－), 男, 副研究员, CCF 专业会员, 主要研究领域为数据库, 工业大数据, 工业化联网.

王宏志(1978－), 男, 博士, 教授, 博士生导师, CCF 杰出会员, 主要研究领域为数据库管理系统, 大数据分析与治理.

李昊轩(2001－), 男, 本科生, 主要研究领域为数据清洗, 异常检测, 时序数据挖掘.