

中

TP391

论 : 10006ZY1806105

北京航空航天大學

专

时

作

专 计

指

培 计

**Research and Application of Temporal Graph
Association Rule**

A Dissertation Submitted for the Degree of Master

Candidate : Liu Chengyang

Supervisor : Prof. Fan Wenfei

School of Computer Science and Engineering

Beihang University, Beijing, China

中 TP391

论 10006ZY1806105

硕

时

作

指

称 教

专

图

学

论

学

月 日

关

本

人

郑

重

声

明

:

所

呈

交

的
论
文
是
本
人
在
指
导
教
师
指
导
下
独
立
进
行
研
究
工
作
所
取
得
的
成
果

，论文中有关资料和数据是实事求是的。

。尽我所知，除文中已经

加以标注和致谢外，本文不包含其他人已经发表或撰写的研究所

成
果
,
也
不
包
含
本
人
或
他
人
为
获
得
北
京
航
空
航
天
大
学
或
其
它
教
育

机
构
的
学
位
或
学
历
证
书

若

学

期

学

本

人
完
全
同
意
北
京
航

空
航
天
大
学
有
权
使
用
本
学
位
论
文
(
包
括
但
不
限
于
其
印
刷
版
和
电
子

版
)
,
使
用
方
式
包
括
但
不
限
于
:
保
留
学
位
论
文
,

按
规
定
向
国
家
有

关
部
门
(
机
构
)
送
交
学
位
论
文
,

以
学
术
交
流
为
目
的
赠
送
和
交
换
学

位
论
文
,
允
许
学
位
论
文
被
查
阅

、
借
阅
和
复
印
,
将
学
位
论

保

学
指

題
題

摘要

随着大数据产业的发展，图理论在现实世界中有着越来越多的应用场景。图研究中的一个热门方向是关联规则研究，图关联规则被广泛应用于事件预测、模式识别和用户行为研究等领域。时态信息总是伴随着数据产生，并对事件的发生起到举足轻重的作用。本文以时序图的关联规则为研究方向，旨在进一步提高图关联规则的表达能力，以解决时态问题，为事件预测问题提供新的尝试思路。

本文的主要工作包括：

1. 提出了一种结合时序和时间窗的时序图关联规则，该规则相比其它规则具有更强的时序表达能力。
2. 给出了基于支持度和置信度的时序图关联规则发现算法，并分析了其时间复杂度。针对发现算法给出了一系列优化策略，包括扩边引导、多线程加速、匹配时间阈值、时间窗口和动态支持度剪枝。
3. 利用时序图关联规则可以进行事件预测。先给出时序图匹配算法，并分析其时间复杂度，然后给出可多机加速的并行应用算法，并证明了并行可扩展性。
4. 在真实的图数据上，通过实验证明了时序图关联规则和发现算法的准确性，并验证了发现算法和应用算法的可扩展性。
5. 本文设计并实现了可视化的图平台，该平台集成了时序图关联规则的发现和应用算法，图数据分析人员在该平台可以方便快捷地完成任务。

关键词：图数据，时序图，关联规则，事件预测

Abstract

With the development of the big data industry, graph theory has more and more application scenarios in the real world. A popular direction in graph research is association rules. Graph association rules are widely used in the fields of event prediction, pattern identification, and user behavior research. Temporal information is always generated with data and plays a decisive role in the occurrence of events. This dissertation takes the temporal graph association rules as the research direction, and aims to further improve the expressive ability of the graph association rules, so as to solve temporal problems and provide new ideas for event prediction.

The main work of this dissertation includes:

1. Propose temporal graph association rules combining time order and time window, which has stronger temporal expression ability than other rules.
2. The algorithm for discovering temporal graph association rules based on support and confidence is given, and its time complexity is analyzed. A series of optimization strategies are given, including edge extension guidance, multithreading acceleration, matching time threshold, time window and dynamic support pruning.
3. The temporal graph association rules can be used for event prediction. First, the temporal graph matching algorithm is given, and its time complexity is analyzed. Then the parallel application algorithm that can be speeded up by multiple machines is given, and the parallel scalability is proved.
4. On the real graph data, the accuracy of the temporal graph association rules and discovering algorithm is verified through experiments, and the scalability of the discovering algorithm and application algorithm is verified.
5. A visual graph platform is designed and implemented. The platform integrates the discovering and application algorithms of the temporal graph association rules. Graph data analysts can quickly and easily complete tasks on this platform.

Key words: Graph Data, Temporal Graph, Association Rules, Event Prediction

目 录

第一章 绪论.....	1
1.1 研究背景及意义.....	1
1.2 研究现状.....	2
1.3 研究目标和研究内容.....	6
1.3.1 研究目标.....	6
1.3.2 研究内容.....	6
1.4 本文组织结构.....	6
第二章 相关理论和技术.....	8
2.1 基于图的关联规则.....	8
2.1.1 图.....	8
2.1.2 图模式.....	9
2.1.3 图模式匹配.....	10
2.1.4 图关联规则模型与发现算法.....	10
2.2 时态网络上的规则研究.....	12
2.2.1 图时态关联规则.....	12
2.2.2 时态网络主题.....	13
2.2.3 时态网络用户行为.....	14
2.2.4 时态网络的时间窗口.....	15
2.3 图算法并行化技术.....	16
2.4 本章小结.....	17
第三章 时序图关联规则和发现方法.....	18
3.1 时序图问题研究.....	18
3.1.1 时序图定义.....	18
3.1.2 时序图模式定义.....	19

3.1.3 时序图模式匹配定义.....	21
3.2 时序图关联规则.....	21
3.2.1 时间窗口.....	21
3.2.2 时序图关联规则模型.....	23
3.2.3 模型的优势.....	25
3.3 时序图关联规则的发现.....	25
3.3.1 支持度和置信度的定义.....	25
3.3.2 发现问题描述.....	28
3.3.3 发现算法.....	28
3.3.4 优化策略.....	33
3.3.5 复杂度分析.....	37
3.4 本章小结.....	38

第四章 基于时序图关联规则的事件预测.....40

4.1 问题描述.....	40
4.2 时序图的匹配研究.....	41
4.2.1 匹配算法.....	41
4.2.2 复杂度分析.....	44
4.3 并行化方案.....	45
4.3.1 时序图划分方法.....	45
4.3.2 事件预测的并行算法.....	47
4.3.3 复杂度分析.....	48
4.4 本章小结.....	49

第五章 实验与分析.....50

5.1 数据集准备.....	50
5.2 实验设置.....	51
5.2.1 算法设置.....	51
5.2.2 实验环境设置.....	52

5.3 时序图关联规则发现实验及分析.....	52
5.3.1.时序图关联规则和发现算法的准确性.....	52
5.3.2 发现算法的可扩展性.....	54
5.3.3 发现结果示例.....	57
5.4 时序图关联规则应用实验及分析.....	58
5.4.1 时序图匹配算法的效率.....	58
5.4.2 事件预测算法的可扩展性.....	59
5.5 本章小结.....	61
第六章 时序图关联规则计算平台的设计与实现.....	62
6.1 系统架构设计.....	62
6.2 数据格式设计.....	62
6.3 业务层逻辑设计.....	63
6.4 用户交互层设计及界面展示.....	65
6.5 本章小结.....	67
总结与展望.....	68
论文总结.....	68
工作展望.....	69
参考文献.....	70
致谢.....	75

图目录

图 1 图数据示例.....	9
图 2 图模式示例.....	9
图 3 图模式关联规则示例.....	11
图 4 关联规则处理流程图.....	11
图 5 图时态关联规则示例 ^[28]	13
图 6 时态网络主题示例 ^[22]	14
图 7 行为图模式示例 ^[24]	14
图 8 并行加速方案图.....	16
图 9 通话网络示例.....	18
图 10 通话网络模式示例.....	20
图 11 时间窗划分的通话时序子图.....	22
图 12 时序图关联规则.....	24
图 13 时序图关联规则发现过程示意图.....	29
图 14 边时序扩展示意图.....	33
图 15 可扩展边信息哈希结构图.....	34
图 16 时序信息数据结构图.....	42
图 17 次序匹配处理流程示意图.....	43
图 18 以中心点跳数对图划分示意图.....	46
图 19 以时间窗口对时序图划分示意图.....	47
图 20 发现算法效率和准确率实验结果.....	54
图 21 线程数 n 对发现算法效率影响实验结果.....	55
图 22 剪枝阈值 σ 对发现效率影响实验结果.....	56
图 23 图规模 G 对发现效率影响实验结果.....	57
图 24 真实时序图关联规则示例.....	57
图 25 时序图匹配算法的效率对比实验结果.....	58
图 26 计算节点数 n 对事件预测效率影响实验结果.....	59

图 27 规则集规模 Σ 对事件预测效率影响实验结果.....	60
图 28 时序图规模 G 对事件预测效率影响实验结果.....	61
图 29 系统架构图.....	62
图 30 时序图模式展示界面.....	65
图 31 时序图关联规则发现工具界面.....	66
图 32 时序图匹配查询工具界面.....	66
图 33 任务管理界面.....	67

表目录

表 1 现有关联规则研究的各类方法对比	5
表 2 时序图关联规则发现算法主流程	30
表 3 边扩展函数	31
表 4 次序扩展函数	32
表 5 次序匹配算法	42
表 6 时间窗划分时序子图算法	46
表 7 基于时序图关联规则的事件预测并行算法	48
表 8 实验选取数据集	50
表 9 发现算法设置表	51
表 10 事件预测算法设置表	52
表 11 时序图关联规则和发现算法准确性实验结果	53
表 12 发现算法可扩展性实验项目表	55
表 13 事件预测算法可扩展性实验项目表	59
表 14 点文件 CSV 格式	63
表 15 边文件 CSV 格式	63
表 16 时序图关联规则发现服务的配置文件字段表	64
表 17 时序图匹配服务的配置文件字段表	64

第一章 绪论

本章首先从研究背景和意义出发，阐述论文针对的问题和研究价值；随后分析国内外的研究现状，介绍目前为止的关系型数据的时态规则、图关联规则和依赖、时态网络规则和基于神经网络的规则等相关研究；接着提出本文的研究目标和内容；最后给出论文的整体组织结构。

1.1 研究背景及意义

随着基于社交网络模式的一系列新型应用的快速发展，人类获取数据的规模正以前所未有的速度爆炸式增长，与大数据相关的技术变革成为当今世界的热点话题。针对大数据的规则挖掘和分析等研究正在如火如荼地开展，具有巨大的商业价值和应用前景。而时态信息是数据中不可忽视的一个维度，无论数据的存储结构如何变化，它总是伴随着数据而产生，揭示着规则之间的因果关联。

由于数据库理论的发展较为成熟和关系型数据简洁的对应形式，关系型数据结构与时态信息的结合已经屡见不鲜，如对商品的价格预测^[1]和商品关联研究^[2]，对规则发生和时态之间的关系研究^[3,4]、规则的周期^[5]和间隔研究^[6]等等，从而强化关联规则，更好地为顾客推荐商品。但随着数据规模和丰富程度的增大，基于关系型数据结构的规则无法挖掘数据间深层次联系的缺点也愈发明显。

图结构具有天然描述数据拓扑关系的能力，现实世界中的许多应用场景都需要图结构表示，与图相关的应用几乎无所不在。比如，传统应用中的最短运输线路的确定、疾病爆发路径的预测、科技文献的引用关系、生物信息网络分析等，以及新兴应用中的社交网络分析、知识图谱、数据万维网、人脑网络等，都可以看作大图数据的应用^[7]。

图关联规则和依赖是近年来针对图数据库提出的理论体系。从理论意义上讲，图关联规则和依赖研究可以进一步完善图的理论体系，帮助人们更好地理解大图数据，提高大图算法效率；从实际应用上讲，基于图的关联规则^[8,9]可以用于规则预测，挖掘潜在顾客，进行社交网络推销，图依赖理论^[10-15]可以用于图的一致性检测，实体识别等领域。

现实世界中的实体和事件组成的图更为复杂，并且往往与时间相关，从而形成了时态网络。在时态网络上使用关联规则进行事件预测具有更大挑战，但更加贴合真实场景，具有更高的应用价值。时态网络的研究在数据融合^[16,17]、安全领域的异常检测^[18-21]、网络主题分析^[22,23]、用户行为识别^[24-26]、事件匹配^[27]和预测^[28,29]等方面具有重要意义。

基于图的关联规则具有更好的预测能力，但是目前规则中对时态关系的结合还可以进一步增强，从而在时态网络中发挥更大作用。如何使用关联规则刻画事件之间的时态信息是一个值得研究的问题。一组事件之间的时序可以表示事件发生的先后关系，作为事件产生过程中所具有的天然属性，在一定程度上能够揭示事件的发生规律，结合时序的关联规则能够具有更好的表达能力。

基于以上背景，本文选题为基于时序图的关联规则研究，旨在进一步提高现有的图关联规则的表达能力，以解决时序图的关联规则问题。其意义在于，将时序特性引入到图关联规则中，一方面提高图关联规则的表达能力，完善图数据库的相关理论体系，另一方面为时序图上的关联规则预测问题提供新的尝试思路。

1.2 研究现状

随着大数据产业的发展，图结构被广泛应用于各个领域，基于图的关联规则有着越来越多的应用。时态信息总是伴随着数据的产生，无论数据的存储结构和研究方法如何变化，数据的时态信息永远是值得被考虑的一个维度。接下来将主要从关系型数据的时态规则、图的关联规则和依赖、时态网络的规则研究和神经网络方法的规则预测等方面来介绍现阶段的关联规则研究现状。

关系型数据的时态规则。基于时间序列模型的相关研究可以用于各类单一数据维度的预测，例如商品价格的预测^[1]。在传统的事务数据库中，关联规则及其在时间维度上的扩展已经有了较为充分的研究，如可以在商品购买数据中发现关联规则^[2]。这类规则通常通过统计方式将时态属性加入到关联规则中，进而研究事务集的发生和时态之间的关系^[3,4]。Cyclic ARs^[5]主要研究了随时间周期性发生的规则。已经有工作研究了规则的时态特性^[30]，其中包括规则拥有的时间间隔等，在规则的发现过程中可以将其视作规则存在的生命周期^[6]。较为复杂的组合的类模式也有相关研究，这类模式由与时态

配对的一组事件所组成^[3]。

关系型数据库从诞生伊始就迅速发展，现如今已经相对成熟。基于关系型数据库开展的各类规则研究和数据挖掘也较为充分，在关系数据结构中对时序信息有较为全面的考虑和利用。但是基于关系型结构开展的关联规则研究大多采用统计类的方法，数据形式和维度比较单一，忽略了数据中不同实体之间的关联性，并且对于规则的直观解释较为模糊。

图的关联规则和依赖。随着关系数据库的理论体系日趋完善和近些年图数据库的广泛应用，研究的关注点开始逐渐向图数据库转移。图数据上的关联规则和依赖理论，填补了图数据库相关理论领域的空白，为图数据的规则应用和数据质量提升提供了新的思路和依据。

图模式关联规则(GPAR)^[8]定义了一类基础的图关联规则，对于一个模式 Q ，图 G 中任何满足该模式的匹配在 x 和 y 之间都有一条标签为 q 的边。在社交网络和用户商品购买的图数据中使用该关联规则可以进行预测，挖掘潜在的顾客。量化图关联规则(QGAR)^[9]是在 GPAR 的基础上，增加了量词约束，进一步丰富了图关联规则的表达能力，使得图关联规则能够表达除了等于之外的多种逻辑关系。

随后应用于实体识别和图一致性检测的基于图属性的规则约束被逐步提出，该依赖主要借鉴于关系数据库的主键、函数依赖和匹配依赖等依赖关系^[31-34]，扩充了图数据库的依赖理论体系。

图主键(graph keys)^[9]提出将图模式与条件约束相结合，通过定义图模式匹配的语义将关系数据库上的主键的概念扩展到图数据上，从而将图主键作为识别图中唯一实体的依据。图函数依赖(GFD)^[11]主要对图属性值进行约束，它在图的拓扑结构基础上，增加了语义检查，作为函数依赖在图数据上的扩展，图函数依赖为识别图数据中的不一致现象和限制图数据中的逻辑错误提供了方法和依据。图实体依赖(GED)^[12]通过采用图同态的匹配模式，从理论上将图主键和图函数依赖进行了统一。图数值依赖(NGD)^[13]是在图依赖的基础上提出的，考虑到图的属性之间的数值关系而提出的一种依赖，它将原有的语义集合中的等值判断扩充成了线性操作，即两个语义之间不再仅仅是“等于”的关系。为了满足图数据的实体识别问题的离线模式，Kwashie 等人提出了图差别依赖(GDD)^[14]，Ma 等人提出了本体图依赖(OGK)^[15]。

图关联规则可以通过对图模式的匹配来挖掘潜在顾客。图依赖着眼于图节点确定和图属性研究，被广泛应用于实体识别和图一致性检测上。图关联规则和图依赖相辅相成，将图的关联规则和依赖理论体系扩充完善，在边预测、实体识别和图一致性检测上有着广泛应用。

基于图的关联规则能够充分地刻画不同实体之间的关联性，并且具有良好的可解释性，但是该类方法不着重于对时态信息的捕捉。针对不同的应用问题，其表达能力还有待进一步提升，另外由于匹配问题的复杂性，如何加速基于图的关联规则的发现过程也是一个非常值得研究的问题。

时态网络上的规则研究。不同于静态图结构，时态网络中的实体、事件以及相关属性会随时间发生连续的变化。时态信息非常关键，最直观的便是可以通过时态信息对不同来源的数据进行融合^[16,17]。在实际生活中，数据因其本身的含义可能具有大量的时态特点，而在工业生产中，数据的产生本身就带有时间戳属性。

对于时态网络的相关研究多是在异常检测上，因此主要应用于安全领域。通常做法是设计一系列基于图模式或统计学方法的算法对时态网络进行监测，从而探测到异常的事件变化^[18]。例如可以将计算机系统的监控数据作为一个时态网络，利用文件访问日志、防火墙日志^[19]和网络监控日志^[20]展开特定的应用研究，还有通过系统调用日志捕获系统活动的综合方法研究^[21]，系统调用日志相比部分日志更能涵盖整个系统实体。

时态网络中对事件的检测也类似于静态图上的匹配查询，目前有大量基于时态网络的时态查询研究，用于检测动态网络上的事件。这些查询基于不同的图匹配方法，例如有基于子图同构^[35,36]、密集子图^[37]、准斜度^[38]、最大化边缘权重的重子图^[39]、在演进图上查询子图同构的连续图模式^[40]和同构匹配时间持续最长的持久查询^[36]等，采用何种匹配语义与研究内容和应用场景密切相关。

对于时态模式的发现也有相关研究工作，该研究通过构造树结构来枚举和验证频繁的模式^[41]。另外在时态网络中已经有发现时态网络的主题^[22,23]和用户行为^[24-26]以及基于图数据流的事件匹配^[27]的相关研究，此类研究可以用于发现时态网络中有意义的事件主题和通过用户行为来识别风险信息等。

此类时态模式在生成时会结合时间属性，因此对于关联规则如何结合时序很有借

鉴意义。但基于此类主题和事件进行的关联规则研究较少，规则上所结合的匹配模式较为基础，表达能力有限。图时态关联规则^[28]通过扩展图模式关联规则，将时间加入图中作为图的基础属性，研究了时态网络中的事件规则挖掘，并设计实现了可以挖掘 top-k 网络事件的系统 BEAMS^[29]，给出了直观的结合时态进行图规则发现的一种方法，但该方法的搜索空间较大，适用于检索时间窗口较小的图模式规则挖掘，并且该方法仅仅将时态信息作为预测下一次事件发生的间隔时间，本质上依然没有考虑实体之间事件发生的时态关系，对于时间关系的刻画能力仍有提高的空间。

神经网络方法的图规则预测。神经网络在各个领域的问题研究中扮演着越来越重要的角色，通过大规模网络训练，许多问题都迎刃而解。通过机器学习和神经网络的方法来处理图数据，从中探寻有意义的规则是一个值得研究的方向。

图神经网络(GNN)更是专门为处理图结构数据而设计。已经有一些使用 GNN 进行规则预测的相关研究工作，例如使用深度学习方式对社会影响力进行预测^[42]，使用应用程序内的用户行为图来表征和预测用户的参与度^[43]，使用双向图卷积网络进行社交媒体谣言检测^[44]，一种基于 γ 衰减理论的利用 GNN 学习的启发式方法进行边预测^[45]。但是使用神经网络方法获得的规则可解释性差，并且存在着过拟合的风险。

根据对上述相关研究的调研，汇总得到如下对比表格。

表 1 现有关联规则研究的各类方法对比

方法类型	代表模型	优势	不足
关系型时序关联规则	ARIMA, Cyclic ARs ^[5]	表示简单，使用传统的统计概率方法，结果可靠。	忽略了数据之间的拓扑关系，无法挖掘数据深层次的关联。
图模式关联规则和依赖	GPAR ^[8] , QGAR ^[9] , GFD ^[11]	天然具有描述数据拓扑关系的能力，结果有较强的可解释性。	相关研究对数据时态信息结合的程度较低。
时态网络上的规则研究	TMotifs ^[22] , GTAR ^[28] , TGMiner ^[24]	在预测中考虑时态属性对于事件发展的影响。	多数应用在异常检测、主题和行为识别方面。
神经网络方法的时序规则预测	GNN	能处理多维度数据，预测结果具有较高的准确性。	可解释性弱，容易出现过拟合现象。

在图数据结构上设计关联规则，结合数据的时序特性，在具有时间特性的图上进行更精准和高效的关联规则发现和应用，将在保证规则关联性和解释性的前提下，进一步提升关联规则的准确性和表达能力，这是一个可以进一步尝试的研究方向。

1.3 研究目标和研究内容

1.3.1 研究目标

论文工作将围绕时序图关联规则的研究、规则发现方法研究、规则应用研究和系统实现这四个部分展开。在图数据结构上，考虑时序信息，研究时序图关联规则的发现和应用，从而为事件预测问题提供新的思路和方法，并最终集成为一个基于时序图关联规则的发现和应用系统。

1.3.2 研究内容

本文的研究内容分为以下四个部分：

(1) 时序图关联规则的研究。主要包括时序图关联规则的形式化表示，难点在于如何在图模式中引入时序关系，从而确定时序图关联规则的形式化表示，以保证较强的表达能力和时间复杂度之间的平衡。

(2) 时序图关联规则的发现方法研究。主要包括发现算法的设计和时间复杂度分析，主要难点在于考虑了时序之后，如何进行边扩展生成候选规则，如何快速选取有效的规则。需要通过一系列实验来验证发现算法的效率和扩展性，并设计对比实验验证扩展得到的规则的准确率。

(3) 时序图关联规则的应用研究。应用研究主要包括给出具体的时序图匹配算法，并根据具体应用场景来设计基于时序图关联规则的事件预测方法，给出可扩展的多机并行方案，从而在时序图中预测潜在事件或在风控领域中识别风险用户，设计实验验证算法的可扩展性。

(4) 时序图关联规则发现和应用系统的设计与实现。主要包括采用软件工程的设计方法，设计合理的架构，实现一个基于时序图的关联规则发现和应用系统，为图数据分析人员提供良好的用户交互界面和便捷的操作体验。

1.4 本文组织结构

本文各章节的组织结构如下：

第一章为绪论，首先介绍本文的研究背景和研究意义，从背景和应用前景阐述了研究时序图关联规则的意义。并对关联规则领域的相关工作进行了综述，包括传统关系型数据的时序规则、图关联规则和依赖、时态网络的规则研究和神经网络方法的规则预测等。最后介绍了本文的研究目标，并根据该研究目标划分了相应的内容。

第二章为相关理论和技术。介绍了本文研究所需的相关图理论基础和算法，以及所使用的关键技术。主要包括基于图的关联规则的相关理论基础、时态网络上的规则研究和图算法的并行化技术等。

第三章为时序图关联规则和发现问题研究。首先给出了时序图、时序图模式和时序图模式匹配的定义，然后提出了适用于时序图的关联规则。在发现问题研究中，首先给出了支持度和置信度的定义，然后给出了基于支持度剪枝的发现算法，并分析了该算法的时间复杂度，最后针对发现算法提出了一系列优化策略。

第四章为基于时序图关联规则的事件预测问题研究。首先给出了该问题的形式化表达，然后给出了时序图匹配算法以及复杂度分析，接着给出了按时间窗划分时序图的方法，并设计了基于时序图关联规则的事件预测的并行算法，并证明该算法具有良好的并行可扩展性。

第五章为实验与分析。首先描述了实验使用的数据集、算法配置和实验环境配置。然后设计了发现算法实验，主要包括与其他规则和发现算法对比的准确性实验和改变参数验证可扩展性的实验。最后设计了应用算法实验，主要包括时序图匹配算法的效率对比和事件预测并行算法的可扩展性实验。

第六章为时序图关联规则的发现和应用系统的设计与实现。从系统架构、数据格式、业务逻辑和交互设计等系统组成部分详细介绍了该系统。

最后一章是总结和展望，对本文工作进行了总结，并对未来可以开展的研究工作提出了可能的方向。

第二章 相关理论和技术

本章将介绍论文中使用到的相关理论和技术。首先介绍基于图的关联规则的研究，这是本文进一步研究的理论基础；然后介绍与论文研究相关的时态网络上的规则研究；最后介绍图算法并行化技术，本文算法会采用相关技术，以提高运行效率。

2.1 基于图的关联规则

本节将介绍图数据的基础定义和理论，包括图、图模式和图模式匹配，并介绍基于图的关联规则及其发现和应用。

2.1.1 图

图是表示事物与事物之间关系的数学对象，图 G 可以被表示为：

$$G = (V, E, L) \quad (2.1)$$

- (a) V 是点的有限集合；
- (b) $E \subseteq V \times V$ ，是一组边的集合，其中 (u, v) 表示从点 u 到点 v 的边；
- (c) V 中的每个点 u 都具有 $L(u)$ ，表示该点的标签或内容；
- (d) E 中的每条边 e 都具有 $L(e)$ ，表示该边的标签或内容。

例 1：图数据示例。图 1 是一个通话网络的图数据示例，记作 $G_1 = (V_1, E_1, L_1)$ ，

其中 (1) $V_1 = [u_1, \dots, u_4, p_1, \dots, p_4, r]$ ，代表 4 个通话用户和 4 个手机号码，以及 1 个风险用户标识，一共包含 3 种类型的点标签，分别是 *user*、*phonenumber* 和 *risk*；(2) 总共存在 13 条边， $|E_1| = 13$ ，一共包含 4 种类型的边标签，分别是用户和手机号码之间的通话 (*call*) 和短信 (*msg*)，以及是否为风险用户 (*is/is_{not}*)。

例如，边 (p_1, u_1) 代表手机号码 p_1 向用户 u_1 拨打了一个电话，边 (u_4, p_4) 代表

用户 u_4 向手机号码 p_4 发送了一条短信，边 (u_2, r) 代表用户 u_2 是一个风险用户。

该图数据示例选取自联通的用户通话网络^[46]，数据中提供了 45 个连续自然日期间，抽样模拟的 4999 个用户每天的通话和短信的脱敏数据，用户分为正常用户和风险用户两类，其中风险用户可能参与电信诈骗、拨打骚扰电话等违法活动。该图定义无法表示两点之间的重复边，即无法表示一个手机号码给同一个用户拨打多次电话的情况，此类情形在该图定义下只会被表示为一条边。

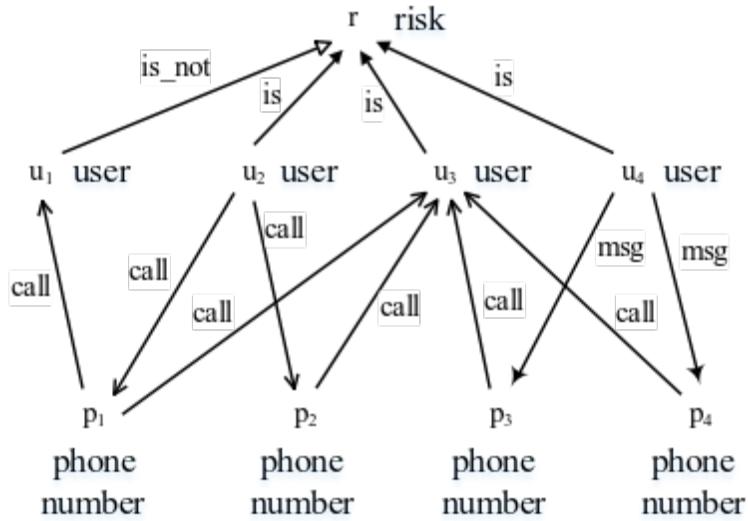


图 1 图数据示例

2.1.2 图模式

一个图模式是一个有向图 Q ，可以表示为：

$$Q = (V_Q, E_Q, L_Q) \quad (2.2)$$

- (a) V_Q 是有向图 Q 中的点集合；
- (b) E_Q 是有向图 Q 中的边集合；
- (c) V_Q 中的每个点 u 具有标签 $L_Q(u)$ ， E_Q 中的每条边 e 都具有标签 $L_Q(e)$ ，该标签用于指定搜索的条件。

例 2：图模式示例。图 2 是一组来自用户通话网络的图模式示例：(1) Q_1 表示

一个用户给一个手机号码拨打电话； (2) Q_2 表示一个手机号码接到一个用户呼入的电话和给一个用户呼出一个电话； (3) Q_3 表示一个用户是风险用户，该用户给两个手机号码发送短信。

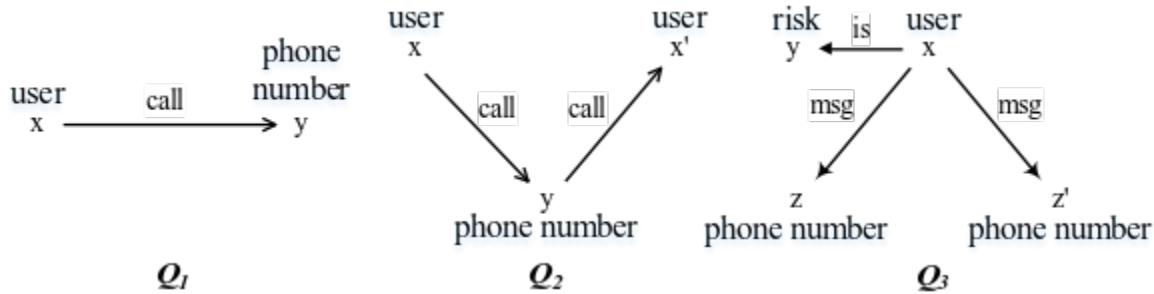


图 2 图模式示例

2.1.3 图模式匹配

图模式匹配根据匹配语义的不同，可以有多种匹配方式，最常用的匹配语义为图同构和图同态语义。

本文采用图同构的匹配语义，图模式 Q 在图 G 中的匹配是从 Q 中的点到 G 的子图 G' 的点的双射函数 h ，因此对于每个点 $u \in V_Q$ ，满足

$L_Q(u) = L(h(u))$ ，并且 (u, v) 是 Q 中的一条边，当且仅当 $(h(u), h(v))$ 是 G' 中

的一条边并且满足 $L_Q(u, v) = L(h(u), h(v))$ 。

例 3：图模式匹配示例。考虑图 2 中的 Q_2 在图 1 中的图 G_1 上的匹配，可以得到 3 对匹配结果。使用图模式 Q_2 的边到图 G_1 上的边的映射关系来表示匹配结果，

则有 $M_1 = \{(x, y) \mapsto (u_2, p_1), (y, x') \mapsto (p_1, u_1)\}$ ， $M_2 = \{(x, y) \mapsto (u_2, p_1), (y, x') \mapsto (p_1, u_3)\}$ 和

$M_3 = \{(x, y) \mapsto (u_2, p_2), (y, x') \mapsto (p_2, u_3)\}$ 。

采用图同构或同态语义的图匹配问题是一个 NPC 问题^[48]，针对该问题的主流算法均是采取了搜索和剪枝的策略，来查找给定子图在目标图上的所有匹配^[49-53]。匹配算法所采用的主要思想均是根据已知信息，在目标图中寻找与子图中的点可能匹配的点的

候选集，然后由该候选集进行每一轮的扩展，在扩展过程中通过剪枝策略加速搜索过程，直到最终扩展出与子图匹配的图结构或者无匹配，搜索完所有情况为止。根据图的种类不同，剪枝策略的效果也不尽相同，不断提出的新算法由于采用了更有效的剪枝策略，所以比之前的算法有较大的性能提升，但是该问题依然是 NP 难的问题。

2.1.4 图关联规则模型与发现算法

图模式关联规则(GPAR)^[8]定义了一类基础的图关联规则，对于一个模式 Q ，图 G 中任何满足该模式的匹配在 x 和 y 之间都有一条标签为 q 的边，形式化表示为：

$$R(x, y): Q(x, y) \Rightarrow q(x, y) \quad (2.3)$$

- (a) $Q(x, y)$ 表示包含点 x 和 y 的图模式；
- (b) $q(x, y)$ 表示点 x 和 y 之间存在一条标签为 q 的边。

例 4：图模式关联规则示例。图 3 是一个来自通话网络的图模式关联规则示例。该规则的含义为：如果一个用户 (x') 是 (is) 风险用户 (y)，该用户拨打了电话 ($call$) 给一个手机号码 (z)，随后该电话号码向另一个用户 (x) 拨打了电话，那么预测另一个用户也可能是风险用户。

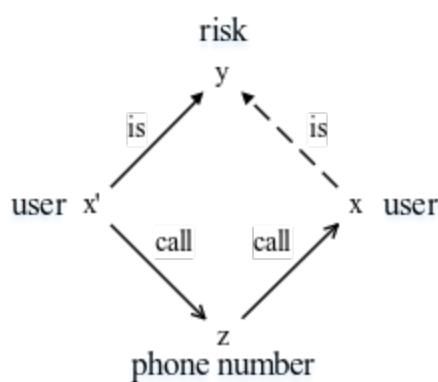


图 3 图模式关联规则示例

由以上示例可以看出 GPAR 在图数据上对于关联规则具有较好的刻画能力，但在具有时态关系的事件网络中，缺少对因果关系的表达能力，因此还有进一步的提升空间。接下来将介绍图关联规则的一般发现方法和应用方法。

图模式关联规则^[8]中研究了关联规则的发现问题，发现采取的主要思想是搜索和剪枝。整个发现过程可以分为扩展规则、为规则打分更新 top-k 队列和对规则进行剪枝三个部分。最开始的候选集中只有预测边 $q(x, y)$ ，在此基础上进行加边扩展，对扩展得到的每一条规则的处理流程如下图所示。

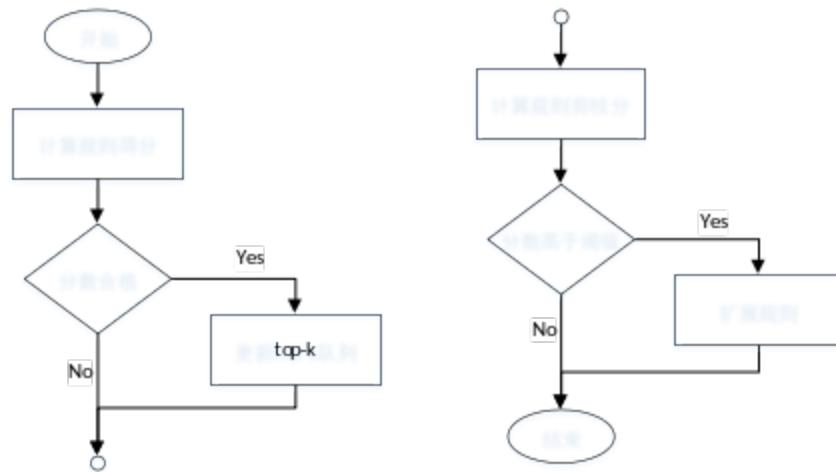


图 4 关联规则处理流程图

首先计算该规则的得分，图模式最简单的得分计算方式就是计算其在图中的匹配次数，因此需要进行子图匹配计算。但是在实际发现过程中单单考虑正确匹配是不够的，图模式关联规则中定义了支持度和置信度的概念，通过置信度来对规则进行打分。对该规则的分数是否合格进行判断，如果合格则将其更新到 top-k 队列中。

然后判断该规则是否能够进行下一轮扩展，一般通过计算一个剪枝分数，看是否满足最小的剪枝阈值。在图模式中最典型和好用的策略就是使用有效的匹配次数作为剪枝的衡量标准，当然该剪枝策略可以是多角度方法的组合，共同剪枝，从而更有效的减少候选规则的数量，提升整个发现算法的效率。如果该规则的剪枝分数高于阈值，则对该规则进行加边扩展，生成一系列新的候选规则，这些候选规则将在下一轮计算置信度判断是否放入 top-k 队列中，并且用于扩展新的规则直到循环轮数结束。

图模式关联规则^[8]中将规则的应用描述为实体标识问题（Entity Identification Problem），通过关联规则来标识某个实体所属的种类，进而在社交网络上实现对顾客购买商品的行为预测。图模式关联规则没有使用时序信息，无法刻画事件发生之间的时序关系，如在通话网络数据中无法表示两个人之间不同时间发生的多次通话事件，因此需要进一步结合时序关系，对该关联规则进行扩展。

2.2 时态网络上的规则研究

本节将介绍与本文研究相关的一些时态网络上的规则研究，主要包括图时态关联规则、时态网络的主题、用户行为和时间窗口研究。

2.2.1 图时态关联规则

图时态关联规则(GTAR)^[28]在图模式关联规则的基础上，将关联规则的应用范围从图扩展到了时态图上，使用图时态关联规则能够通过既定的事件 P_1 预测 Δt 时间后的事件 P_2 ，对时态图的定义如下：

$$G_T = (V, E, L, T) \quad (2.4)$$

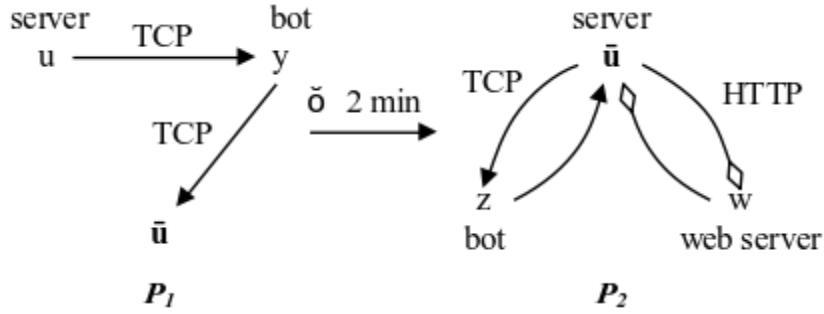
- (a) T 是一个时间窗口（一系列连续的时间戳）；
- (b) V 是点的集合， $E = V \times V \times T$ ，是与 T 中的时间戳相关联的有向边的集合；
- (c) 每个点 $v \in V$ 都有函数 L 分配标签 $L(v)$ ，每条边 $e \in E$ 都有函数 L 分配标签 $L(e)$ ， $e = (u, v, t), t \in T$ ，表示在 u 和 v 之间存在一条带有标签 $L(e)$ 时间戳为 t 的边。

对于图时态关联规则的定义如下：

$$\Phi: (P_1 \Rightarrow P_2, u, \Delta t) \quad (2.5)$$

- (a) $P_1 = (V_1, E_1, L_1, u)$ 和 $P_2 = (V_2, E_2, L_2, u)$ 是两个共享公共点 u 的事件，将 P_1 和 P_2 分别称为 Φ 的前因事件和后果事件；
- (b) Δt 是指定时间间隔的常数。

例 5：DDoS 攻击的图时态关联规则^[28]。图 5 描述了一个网络攻击和信息泄露场景，如果某个主机在某个时候参与了 DDoS 攻击 (P_1) ，则该主机很可能在两分钟之内成为信息泄露 (P_2) 的受害者。

图 5 图时态关联规则示例^[28]

从图时态关联规则的定义可以看出，时态图中的时间信息主要被用于预测下一个事件的发生时间距离当前事件的时间。而对于图模式或者事件本身，并没有考虑时间信息。而在引入时态的图结构中，已经有考虑事件之间时序关系的相关研究工作，但是还尚未用于关联规则预测。

2.2.2 时态网络主题

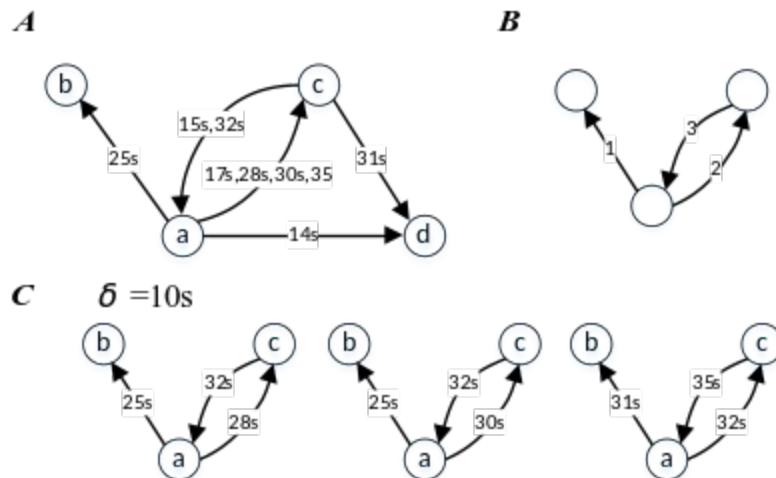
在将图模式作为主题检测^[22]的时态网络研究工作中，点集合 V 上的时态图 T 是元组 (u_i, v_i, t_i) 的集合， $i=1, \dots, m$ ，其中每个 u_i 和 v_i 是 V 中的元素，每个 t_i 是 R 中的时间戳。将特定的 (u_i, v_i, t_i) 元组称为时间边。从 u 到 v 可能有许多时间边。假设时间戳 t_i 是唯一的，因此可以严格地对元组进行排序。最后通过忽略时间戳和重复边，可以从时态图中产生一个标准有向图，称其为在时态图 T 中具有静态边的静态图 G ，即 (u, v) 是 G 中的边当且仅当 T 中存在时间边 (u, v, t) 。

文中将 δ 时间间隔内的时态主题定义为：

$$M = (u_1, v_1, t_1), (u_2, v_2, t_2), \dots, (u_l, v_l, t_l), t_1 < t_2 < \dots < t_l, t_l - t_1 \leq \delta \quad (2.6)$$

表示由 k 个点和 l 条边组成的在 δ 时间内的时态主题，这 l 条边满足时序顺序，并且它们的时序跨度在 δ 时间内。

例 6：时态网络主题示例^[22]。如图 6 所示 A 是一个时态图， B 是一个时态主题。当 $\delta=10s$ 时，该时态主题 B 在时态图 A 上有 3 个匹配，如 C 所示。

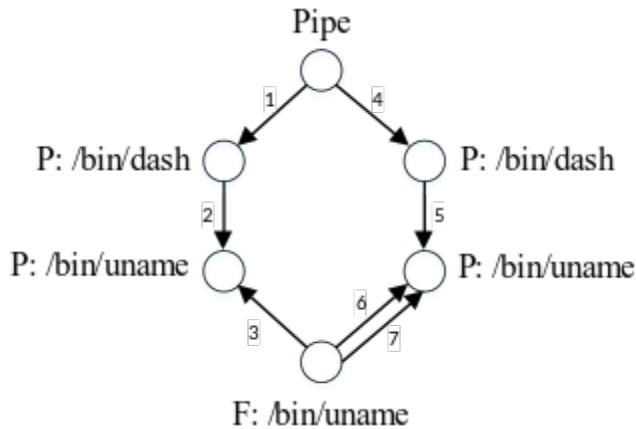
图 6 时态网络主题示例^[22]

该文章在给定了时态图和时态主题定义后，针对几种常见的时态主题分别设计了算法来统计该主题在时态图中的出现频次，从而探究时态网络中的时态主题分布规律，结果表明不同的时态网络中会有不同类型的时态主题分布，这也是符合直觉的。因此，本文考虑将该时态主题中的时序关系概念扩充到图模式匹配中，并设计通用的时序关系匹配算法，使得其在时态图中能够表达更强的时序信息。

2.2.3 时态网络用户行为

时态网络中的用户行为研究的相关工作^[24]，将带有时间顺序的图模式作为行为模式进行研究，并且给出了行为图模式的扩展方式和剪枝策略。

例 7：远程登录用户的行为图模式示例^[24]。图 7 的行为图模式是一个有区别的远程登录用户的行为模式，表示了该计算机用户的异常操作行为。对此类监控日志进行分析可以发现有意义的用户行为模式，从而对用户行为进行研究。

图 7 行为图模式示例^[24]

该模式是在选定一个用户后对其行为进行研究，通过行为模式判断用户的类别，这就意味着无法预测用户接下来的行为，并且该模式也不支持不同节点间具有相同顺序的事件发生，所以为了适用于关联规则预测，该结合时序的图模式依然需要改进。并且行为图模式发现算法的相关研究中，所用的数据规模较小，对较大规模数据的支持有限。所以本文在研究时序规则的扩展时需要考虑更强有力的扩展策略和剪枝策略以提高此类具有时态属性的图模式的发现速度。

2.2.4 时态网络的时间窗口

在涉及用户行为的社交网络数据中，以实体和事件组成的规则通常具有有效时间范围和阶段性的特点，因此在以关系型结构作为规则的相关研究中已经有诸如时间窗口的设置，用于限定规则的作用时间范围，例如使用时间约束的时间函数规则^[54]和具备时间动态的分阶段商品推荐^[55]。因为在时态数据中相同类型的事件会多次发生，使用此类数据构造的图具有点少边多、点的平均度数高的特点。所以多数针对时态网络的研究，都会采用诸如时间窗的方式划定一个时间范围，研究该时间跨度内的图模式。这样既能够满足数据特点，又可以降低计算的复杂度。

在时态网络主题检测^[22]的相关研究中，根据公式 (2.6) 中的主题规则的定义可知，除了给定了图模式的边组成之外，还设置了一个窗口 δ ，边上的最大时间戳 t_l 和最小时时间戳 t_1 需要满足 $t_l - t_1 \leq \delta$ ，用于限定该主题的作用时间范围。

在时态网络用户行为^[24]的相关研究中，定义了一个 T 连通时态图的概念：如果

在一个时态图 $G=(V,E,L,T)$ 中，使用任意的小于时间戳 t 的边 (u,v,t) ，都可以组成一个连通的图，则说明该时态图是 T 连通时态图。如图 7 所示的行为图模式就是一个 $T=7$ 的连通时态图模式。因为当 $t=1,\dots,7$ 时，选择时间戳小于 t 的边所组成的图都是连通的。该定义限定了事件发生在 T 时间以内，同时以连通图的方式限制行为必须连续。

时间窗口的选择也有相关研究，大多数是研究在特定情形下如何确定最优的时间窗口，从而对时态网络进行分割。

在多活动识别的最优时间窗^[56]的研究中，针对放置在城市不同位置的多个传感器在同一段时间内收集的多组数据选择时间窗口，文中设计了启发式的搜索算法，当给定初始时间、结束时间以及搜索时窗口的移动步长后，可以得到多组数据的最优时间窗划分。该时间窗划分适用于相同时间内的多组数据集的时间窗口选择。

在覆盖时态数据的最优时间窗^[57]研究中，则是以最大化用户参与为目标进行时间窗划分，划分后的数据在每个时间窗内都具有尽可能大的用户参与量。该种时间窗划分适用于单个实体的事件发生于一段短周期内的情形，如果该实体产生的事件存在于整个网络过程中，就无法得到合理的划分结果。

本文对于时序图的研究也将采用时间窗口的概念，对于时态周期明显的数据，可以根据数据的时态特点依照人工经验来确定时间窗口，从而更好地约束规则的作用范围和降低匹配计算的时间复杂度。因为用户行为往往受自然周期的影响，所以最简单的划分方式是根据数据的自然周期来划定时间窗口。

2.3 图算法并行化技术

并行化方案有单机多核并行和多机分布式并行两种，如下图所示。

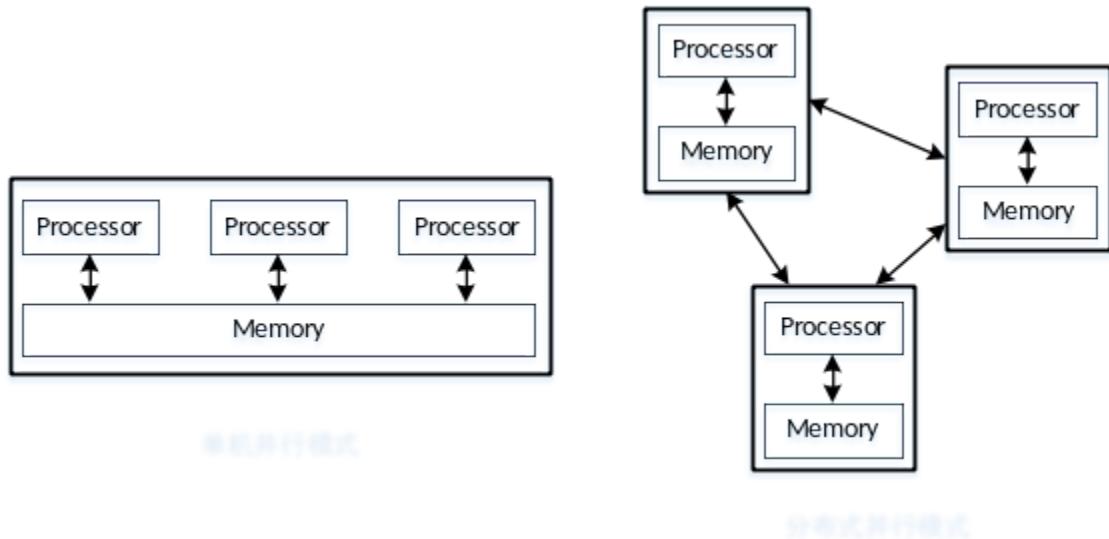


图 8 并行加速方案图

在图计算领域，有大量可以实现并行化方案的技术工具。多线程任务可以采用 OpenMP 的加速方案^[58]，分布式编程可以通过 MPI 实现^[59]，这二者都是高性能计算研究领域常用的技术工具。完成度更高的分布式框架如 Hadoop、Spark 等，除了可以用于传统数据挖掘计算，也可以应用在图类算法的计算上。另外，一些底层数据结构专门为图结构设计的特定通信模式的分布式图计算框架也层出不穷。例如，以点为中心进行通信的 Pregel^[60]、GraphLab^[61]，以图为中心的 Giraph^[62]，以块为中心的 Blogel^[63]，通过 PIE 接口支持串行算法并行化的 GRAPE^[64,65]等计算框架。

规则发现过程中对关联规则的打分判断需要频繁调用匹配算法，而不同关联规则之间是没有数据依赖的，因此本文考虑通过 OpenMP 技术实现多线程对关联规则进行打分，从而提高整个发现算法的效率。挖掘过程采用的图规模一般单机即可存储，因此使用 OpenMP 可以有效避免通信开销对并行性能的影响。

关联规则的匹配考虑到具体应用场景，一组规则可能会用于多组图数据中，并且为了保证应用过程的分布式可扩展性，所以本文考虑采用 MPI 的分布式并行方案来实现时序图关联规则应用的并行化算法。

2.4 本章小结

本章介绍了论文中用到的或者与论文相关的理论和技术。首先介绍了图的基本概念、图模式和图模式匹配等理论基础，进而介绍了基于图模式的关联规则和发现算法；

然后介绍了时态网络上的相关研究，重点介绍了图时态关联规则，时态网络的主题和用户行为研究，以及时间窗口选择；最后介绍了图算法常用的并行化技术方案。

第三章 时序图关联规则和发现方法

本章将介绍一种结合时序和时间窗的图关联规则，并研究其发现方法。首先从图上的时态属性入手，结合实际场景给出了时序图、时序图模式和时序图模式匹配的定义；然后明确了时间窗的概念，给出了时序图关联规则 TGR（Temporal Graph association Rules），并分析了该模型的优势；最后给出了一种基于支持度和置信度的时序图关联规则的发现算法 TGRD（Temporal Graph association Rules Discovering algorithm），并给出了一系列优化策略和算法复杂度的理论分析。

3.1 时序图问题研究

本节通过具体的真实场景来说明时序图、时序图模式以及时序图模式匹配问题，拥有时序的图模式在进行匹配时需要考虑时序的匹配，因此复杂度会有所增加，但所表达的信息更加丰富，刻画更加准确。

3.1.1 时序图定义

在真实的事件网络中，事件的发生一般都带有时间，时间可以描述事件之间的发生次序，从而表达更丰富的关联性。为了更好地研究事件之间的时序关系，选取时间跨度为 2 天的通话网络中的一组场景，将实体表示为点，事件表示为边，事件发生的时间对应到边上的时间戳，得到如图 9 所示的通话网络。

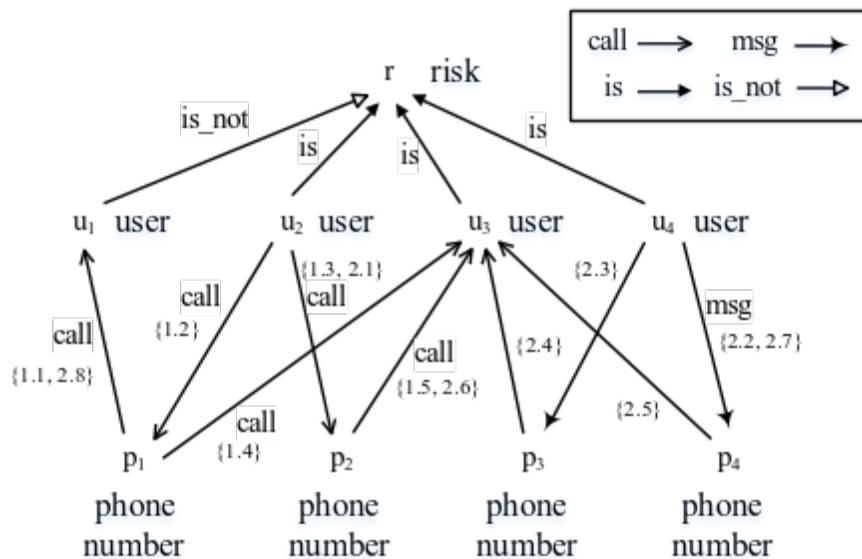


图 9 通话网络示例

该图描述了4个用户与4个手机号码在以天为周期的2天时间内的通话和短信情况，其中包括一些风险用户，如 u_2 、 u_3 和 u_4 ，风险用户是电信诈骗的主要参与者。

通话和短信的事件具有发生时间，该通话数据的原有发生时间格式为“天\时\分\秒”，可以直接使用该格式的时间数据作为时间戳。但是考虑到直接使用该格式比较繁琐，并且为了更好地研究事件发生的时序关系，此处将事件发生时间进行一个映射，得到边上的时间戳。映射后的时间戳格式由“天”和“发生次序”组成，表示为 $x.y$ ， x 代表事件发生在第几天， y 代表该事件是当天发生的第几个事件。这样表示的时间戳在图上呈现的信息比较直观，并且保留了事件之间的时序关系，不会影响后续研究。诸如 is 和 is_not 这类判定作用的边则不具有时间戳，因为如果一个用户被认定为风险用户，那么这个指向将是永久的。

本文将时序图定义为：

$$G = (V, E, L, T) \quad (3.1)$$

- (a) V 表示一个有限点集，每个点 $u \in V$ ，具有一个标签 $L(u)$ ，点的标签用于描述该点所属的实体类别；
- (b) $E \subseteq V \times V \times T$ ，表示与时间相关的边集合，对于每条边 $e = (u, v, t)$ 表示从点 u 到点 v 的标签为 $L(e)$ 的时间戳为 t 的一条边，其中 $t \in T$ ， T 是时间戳集合。边的标签用于描述该边所属的事件类别，边的时间戳用于描述事件的发生

时间，若边上不存在时间戳，则 $t=0$ ，表示该条边在图中一直存在。

例 8：根据定义，图 9 可以表示为一个通话网络时序图 $G_d = (V_d, E_d, L_d, T_d)$ 。该时序图描述了用户和电话号码之间的通话和短信关系，并通过边关系对用户是否属于风险用户进行了标记。该时序图定义可以表示两个实体在不同时刻发生的多次同类型的事件，如边 $(p_1, u_1, 1.1)$ 和 $(p_1, u_1, 2.8)$ 分别表示手机号码 1 在 1.1 时刻和 2.8 时刻给用户 1 拨打了电话，根据时间戳的映射关系，1.1 时刻意味着这是在第 1 天发生的第 1 个事件，2.8 时刻意味着这是在第 2 天发生的第 8 个事件。不具有时间戳的边使用 0 来表示该条边永久存在，如边 $(u_2, r, 0)$ 表示用户 2 是一个风险用户。

3.1.2 时序图模式定义

一般的图模式中没有考虑事件发生次序，因此如果使用图模式在时序图中进行匹配，将得到大量的匹配。在用户的通话网络中，可以找到大量的如图 10 所示的通话网络模式，边上的数字用于标识该事件的发生次序。

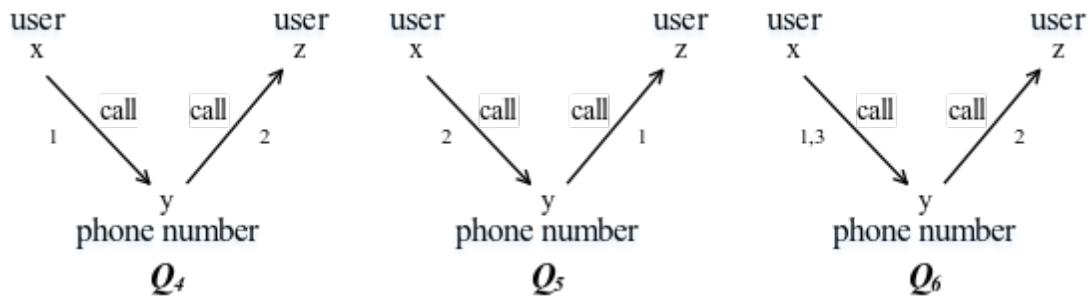


图 10 通话网络模式示例

如果不考虑事件发生时间，则图中的 3 个模式没有任何区别，但是在实际场景中，事件发生的时间也包含了十分重要的信息。不具有时序关系的图模式忽略了事件发生之间的因果关联，并且无法刻画两个节点间的多重同类事件，而时序图模式可以很好地解决这一问题。

本文将时序图模式定义为：

$$Q = (V_Q, E_Q, L_Q, O_Q) \quad (3.2)$$

- (a) V_Q 表示时序图模式中点的集合；
- (b) E_Q 表示时序图模式中边的集合；
- (c) 对于任意点 $u \in V_Q$ 或边 $e \in E_Q$ ，都有一个标签 $L_Q(u)$ 或 $L_Q(e)$ 描述它的类型；
- (d) O_Q 表示边的次序值的集合，对于任意一条边 $e=(u,v,o)$ ，都有且只有唯一的次序值 $o \in O_Q$ 表示它在整个模式中的发生次序，次序值连续且可以重复，若 $o=0$ ，则表示该条边的发生次序不存在，即不考虑时序。

例 9：根据时序图模式的定义，图 10 就列举了 3 个不同的通话网络时序图模式：

- (1) Q_4 表示一个手机号码先接到一个用户电话，又给另一个用户打电话； (2) Q_5 表示一个手机号码先给一个用户打电话，又接到另一个用户打的电话，该模式与 Q_4 模式的事件发生顺序是不同的； (3) Q_6 表示在 Q_4 模式发生后，该号码又接到最初打入的电话。这 3 个时序图模式所描述的场景是不同的。因此相比一般的图模式，时序图模式具有更丰富的语义。

时序图模式允许不同实体间的事件发生次序相同，也允许两个实体间不同类型的事件的发生次序相同。但是不允许两个实体间相同类型的事件发生次序相同，这是因为在现实场景中两个实体之间相同类型的事件在同一时刻只可能发生一次。

时序图模式中边上的次序允许为 0，代表该事件的发生不考虑次序。当所有事件均不考虑发生次序时，时序图模式就退化为普通的图模式，匹配方式也与普通图模式匹配相同，因此时序图模式可以包含一般的图模式。

3.1.3 时序图模式匹配定义

采用子图同构的匹配语义，将时序图模式 Q 在时序图 G 上的匹配表示为从 Q 到 G 存在一个双射函数 h ，满足如下条件：

- (a) 对于 V_Q 中的每一个点 u ，都有 $L_Q(u)=L(h(u))$ ；
- (b) 对于 E_Q 中的每一条边 $e=(u,v,o)$ ，在图 G 中都有 $e'=(h(u),h(v),t)$ ，满足 $L_Q(e)=L(e')$ ；

(c) 对于 E_Q 中的任意两条边 e_i 和 e_j ，它们的次序 o_i 和 o_j 之间的大小关系为 $o_i \oplus o_j$ ，在图 G 中对应的边 e'_i 和 e'_j 的时间戳为 t_i 和 t_j ，满足 $t_i \oplus t_j$ ，其中 \oplus 表示数值的大于、小于或等于这三种关系。

例 10：时序图模式匹配的示例。根据时序图模式匹配的定义，将图 10 中的通话时序图模式 Q_5 在图 9 的通话时序图 G_d 中进行匹配，可以得到 2 个匹配结果。因为时序图中两个点之间存在时间戳不同的多条边，所以使用时序图模式 Q_5 的边到时序图 G_d 上的边的映射关系来表示匹配结果，则有

$$M_4 = \{(y, z, 1) \mapsto (p_1, u_1, 1.1), (x, y, 2) \mapsto (u_2, p_1, 1.2)\} \text{ 和}$$

$$M_5 = \{(y, z, 1) \mapsto (p_2, u_3, 1.5), (x, y, 2) \mapsto (u_2, p_2, 2.1)\} \text{ 。}$$

在本文接下来的章节中，如果没有歧义，则图匹配即指代图模式匹配，时序图匹配即指代时序图模式匹配。

3.2 时序图关联规则

本节介绍时序图关联规则，该关联规则以时序图模式为基础，并增加了时间窗的限定，从而能够描述事件发生的时序关系并且消除过时信息的影响。相比一般关联规则，时序图关联规则能够更好地在时序图上进行事件预测。

3.2.1 时间窗口

在与时间相关的事件网络中，事件的发展往往具有时间的周期性，例如通话网络、交通负载网络、购物网络等。在通话网络中，风险用户对于真实用户的电信诈骗通话通常会在一天之内完成。在真实的通话数据中，将规则作用范围限定在一定时间内，该规则的准确率有所上升，这与预期相符。

时序图中事件发生具有先后性和周期性，因此给出时间窗口的概念：时间窗口是一个大小为 τ 滑动步长为 s 的时间区间，可以将时序图划分为若干个时序子图，从而将时序图模式 Q 在时序图上的匹配问题转变为在若干个时序子图上的匹配问题。

给定一个时序图 G 和时间窗的大小 τ 和滑动步长 s ，可以简单描述根据时

间窗划分时序图的过程：将时序图中的事件按照发生时间升序排列后，以事件的最小时间戳的值作为时间窗口的起点，即时间窗区间的左值，该区间每次按 s 跨度进行滑动，即每次滑动后，时间窗区间的左值和右值均增加 s ，直到区间的左值大于事件的最大时间戳为止。这样可以选出多组事件，每组事件的发生时间均在单次时间窗的时间区间内，一组事件和对应的实体可以组成一个时序子图。

时序图模式 Q 从在整个时序图上进行匹配转化为在划分后的时序子图上分别进行匹配。因此时间窗口的设定不仅可以将过期的冗余信息排除在外，提高规则准确率，同时还可以减少图的规模，降低图匹配的复杂度。

时间窗口的参数一般根据时序图的自然周期通过人工经验选取的方式确定，例如通话和交通网络一般以天为周期，则可以选取 $\tau=1$ ， $s=1$ （其中 1 指代一天，具体的值与时间戳的数据格式有关），而对于可能会存在的事件跨过时间周期的场景，则可以适当设置 $s < \tau$ 。

因为图 9 的通话网络具有自然的天周期，所以可以选择时间窗口大小为 1 天，滑动步长也为 1 天，这样就可以将其划分为 2 个时序子图，划分结果如图 11 所示。

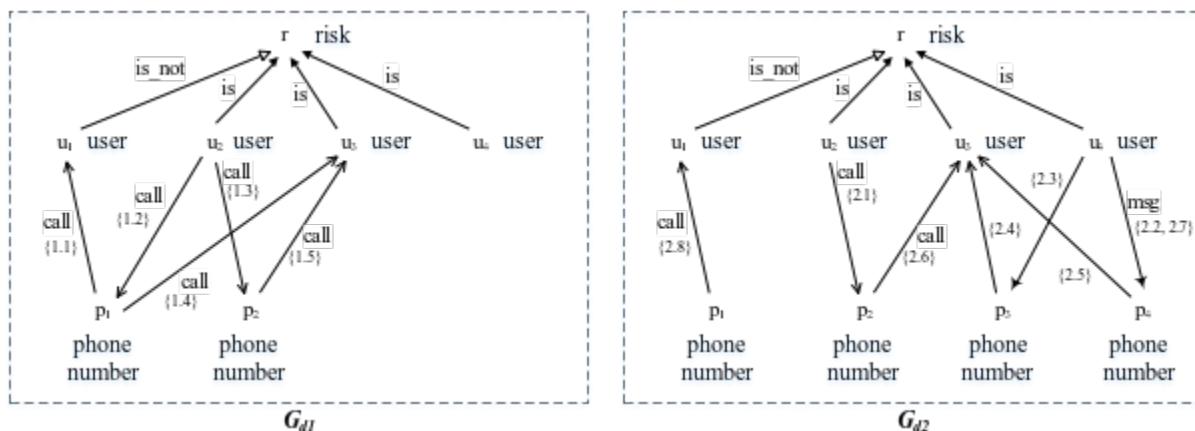


图 11 时间窗划分的通话时序图

因为电话号码 p_3 和 p_4 在第一天内没有通话和短信行为，按照时间窗口划分的方式，划分后的时序图由当前周期内发生的事件和对应的实体组成，因此时序子图 G_{d1} 中没有 p_3 和 p_4 这两个电话号码实体。

例 11：时序图模式在时间窗划分的时序子图上的匹配示例。时序图 G_d 被划分为时序子图 G_{d1} 和 G_{d2} ，分别只包含发生在自己时间窗范围内的事件，此时图 10 中

的通话时序图模式 Q_5 在图 11 时间窗口划分后的时序子图上只有 1 次匹配 M_4 。相比于没有时间窗划分的例 10 中的 2 次匹配减少了 1 次，将横跨时间窗口的事件排除在外，限制了规则的作用范围，起到了排除冗余信息的作用。

3.2.2 时序图关联规则模型

将时序图关联规则 TGR (Temporal Graph association Rules) 定义为：

$$Q(x, \tau, s) \rightarrow q(x, y) \quad (3.3)$$

(a) $Q(x, \tau, s)$ 表示作用在时间窗口范围内的时序图模式， x 为匹配的中心点；
 (b) τ 为有效的时间窗口大小，表示当 Q 在时序图 G 中的匹配的边，按时间戳升序排列后为 $e'_1, \dots, e'_n \in E$ ，对应的时间戳为 $t'_1, \dots, t'_n \in T$ 时，需满足 $t'_n - t'_1 < \tau$ ；

(c) s 为滑动步长，表示时间窗口每次向后滑动的时间跨度，滑动的起始点为时序图 G 中边的最长时间戳值；
 (d) $q(x, y)$ 表示预测边，即在给定标签的匹配点 x 和 y 之间存在一条标签为 q 的边。

按照时序图模式定义，规则中的预测边 $q(x, y)$ 可以表示为 $e=(x, y, o)$ ，
 $L_Q(e)=q$ ，次序值 o 取值为 0 或者 $o_{max}+1$ ，其中 $o_{max} \in O_Q$ ，是时序图模式 Q 的边上的最大次序值。 $o=0$ 代表该预测边不考虑时序关系，如果边存在则会一直存在； $o=o_{max}+1$ 代表该预测边在时序图模式 Q 中的事件之后发生。

当 τ 为无穷大时，表示该时序图关联规则的作用范围在整个时序图中。当 τ 和 s 分别取一个正整数时，表示会按时间窗大小对时序图进行划分，划分后的图中边的最大时间戳与最长时间戳的差小于该时间窗口大小。

每次时间窗口滑动的步长由 s 决定：当 $s=\tau$ 时，代表对时序图进行了一次均等的划分；当 $s < \tau$ 时，意味着划分后的时序子图存在部分事件的重叠，从而可以更全面地考虑部分跨周期的规则的影响，但同时会增加匹配时间，因此选取参数时应尽量避免 $s \ll \tau$ 。在时序图关联规则发现过程中采用以中心点计算支持度的方式，因此

事件的重叠不会出现重复计算规则占比权重的情况，具体发现方法见下一节相关内容。

一组时序图关联规则如图 12 所示，该组规则是从用户通话网络数据中选取。接下来举例说明如何使用时序图关联规则进行风险用户的识别。

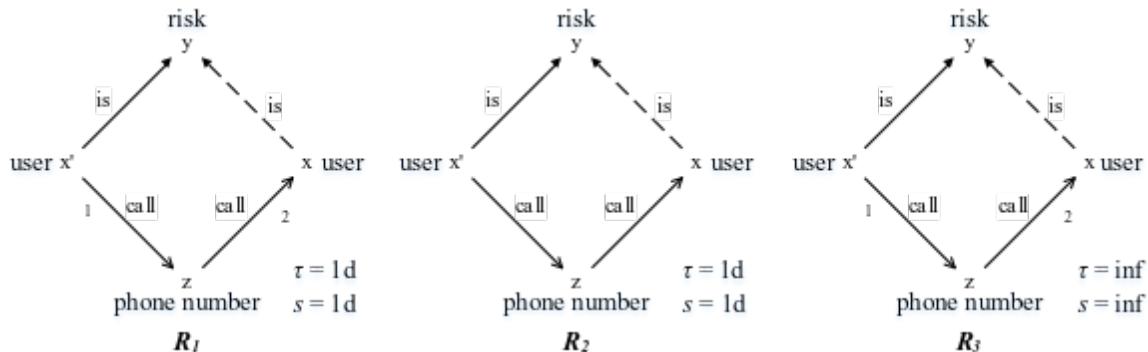


图 12 时序图关联规则

例 12：通话网络时序图关联规则示例。如图 12 所示，时序图关联规则 $R_1=Q_7(x, 1, 1) \rightarrow is(x, y)$ ，该时序图关联规则表示在给定时间窗内满足“一个电话号码接到来自风险用户的电话后又打给另一个用户”的匹配模式时，预测打给的另一个用户也是风险用户。使用 R_1 在图 9 所示通话网络中进行预测，根据 R_1 的时间窗口和滑动步长，该通话网络时序图的划分如图 11 所示。时序图模式 Q_7 在时序子图 G_{d1} 中可以被匹配 2 次，为

$$M_6 = \{(x', z, 1) \mapsto (u_2, p_1, 1.2), (z, x, 2) \mapsto (p_1, u_3, 1.4), (x', y, 0) \mapsto (u_2, r, 0)\} \text{ 和}$$

$$M_7 = \{(x', z, 1) \mapsto (u_2, p_2, 1.3), (z, x, 2) \mapsto (p_2, u_3, 1.5), (x', y, 0) \mapsto (u_2, r, 0)\} ; \text{ 在图 } G_{d2} \text{ 中}$$

被匹配 1 次，为 $M_8 = \{(x', z, 1) \mapsto (u_2, p_2, 2.1), (z, x, 2) \mapsto (p_2, u_3, 2.6), (x', y, 0) \mapsto (u_2, r, 0)\}$ ，

都可以推出 u_3 是风险用户，与实际情况相符。

如果不考虑规则的时序关系，使用 $R_2=Q_8(x, 1, 1) \rightarrow is(x, y)$ 进行预测，此时将会新增匹配 $M_9 = \{(x', z, 0) \mapsto (u_2, p_1, 0), (z, x, 0) \mapsto (p_1, u_1, 0), (x', y, 0) \mapsto (u_2, r, 0)\}$ ，此时根据

规则 R_2 会将 u_1 判定为风险用户，但是实际图中的情况并非如此。在现实中，当诈骗电话呼入一个正常手机号码后，该用户如果上当则可能会马上拨打另外一个诈骗电话，此类电话诈骗很常见。但是在接到诈骗电话前，该正常用户可能会给朋友同事拨打电话，所以如果图模式中没有考虑时序的话，就会造成误判。

如果不考虑使用时间窗对时序图划分，将得到规则 $R_3 = Q_9(x, \infty, \infty) \rightarrow is(x, y)$ ，使用 R_3 进行预测。此时相当于 Q_9 在图 9 的 G_d 上进行匹配，匹配结果除了 M_6 、 M_7 和 M_8 之外，还会有所谓的“溢出”结果。

$$M_{10} = \{(x', z, 1) \mapsto (u_2, p_2, 1.3), (z, x, 2) \mapsto (p_2, u_3, 2.6), (x', y, 0) \mapsto (u_2, r, 0)\}$$

$$M_{11} = \{(x', z, 1) \mapsto (u_2, p_1, 1.2), (z, x, 2) \mapsto (p_1, u_1, 2.8), (x', y, 0) \mapsto (u_2, r, 0)\}$$

则同样会误判 u_1 为风险用户。这是因为电信诈骗的模式只在一个较短的时间周期内有效，如果考虑较长一段时间的通话网络，则一个用户会存在大量的通话行为，额外引入的信息反而会成为干扰项，从而导致误判。带有时序关系的图一般比较稠密，而且同类型事件的发生概率高，因此图模式匹配上的可能性与时间跨度成正比，但是匹配上的事件的相关性与时间跨度成反比。所以使用时间窗既可以减少干扰信息，进而提高规则准确率，还可以减少匹配时的图规模，从而提高匹配速度。

3.2.3 模型的优势

描述事件发生的时序关系。时序图关联规则具有更加丰富的表达能力，在时序图 G 中边的发生具有时效性，如果使用基于传统的图模式匹配的关联规则，将会产生非常多的匹配结果，这些匹配结果并没有考虑图的时序特性。

消除过时的冗余数据信息的影响。通过增加时间窗口，将规则的应用范围限制在一个时间窗口内。事件发生的过程中具有关联的事件往往在相近的时间段内发生，而不会跨越很长时间，因此时间窗口的设置可以避免过时的数据信息的影响。

有界的计算复杂度。在图模式上增强了对时序信息的表达能力，但是并没有改变规则在发现和应用问题上的复杂度。而在实际运用中，时间窗口的设计也能够有效的降低计算复杂度。

降低计算所需的时间复杂度。

3.3 时序图关联规则的发现

本节主要介绍一种基于支持度和置信度的时序图关联规则发现算法。首先对支持度的定义进行扩充，使其在时序图上满足反单调性，基于此给出置信度的表示方式；然后给出时序图关联规则的发现问题的具体描述；接下来给出具体的发现算法以及一系列优化策略；最后给出了发现算法复杂度的理论分析。

3.3.1 支持度和置信度的定义

图模式 Q 在图 G 中的支持度表示为 $supp(Q, G)$ ，代表图模式 Q 在图 G 中的可接受出现频次。图关联规则的支持度的衡量应该满足反单调性，即对于图模式 Q 和 Q' ，如果 $Q' \sqsubseteq Q$ ，在任何的图 G 中，都满足 $supp(Q', G) \geq supp(Q, G)$ 。

支持度的反单调性概念对于时序图依然适用，并且显然适用于时序图的任意时间窗范围内的子图。

对于非时序图，最直接的支持度的定义方式是将图模式 Q 在图 G 中的匹配次数 $\|Q(G)\|$ 作为支持度，但是易知该定义不满足反单调性。在时序图中，容易将该支持度定义扩充为图模式 Q 在时序图 G 的每个时间窗内的出现频次之和，即 $\sum_{G_i \sqsubseteq G} \|Q(G_i)\|$ ，但是该支持度的定义不满足反单调性并且有违直觉，接下来将通过一个具体的例子来说明这种不合常理的情形。

例 13：直接匹配次数的支持度定义违背反单调性。一个通过时间窗划分的部分通话网络时序图如图 11 所示。 Q'_d 是标签为 *phonenumber* 的单个点的图模式，它在时序图 G_d 的第一个时间窗 G_{d1} 中的匹配次数为 2，分别匹配了点 p_1 和 p_2 ，因此在子图 G_{d1} 中的支持度是 2； Q_d 是标签为 *phonenumber* 的一个点和从该点出发的标签为 *call* 的一条边组成的图模式，它在时序图 G_d 的第一个时间窗 G_{d1} 中

的匹配次数为 3，分别匹配了边 $(p_1, u_1, 1.1)$ 、 $(p_1, u_3, 1.4)$ 和 $(p_2, u_3, 1.5)$ ，因此在

子图 G_{d1} 中的支持度是 3。此时尽管有 $Q'_d \subseteq Q_d$ ，但是

$$supp(Q'_d, G_{d1}) < supp(Q_d, G_{d1})$$

导致这种情况发生的原因是图模式 Q 与图中包含一个实体点的部分图结构发生了多次匹配，这在一般的图中出现频率较低，但是在以事件作为边的时序网络中则非常频繁，因为一个实体往往会有许多条同类事件发生。除此之外，该定义模式对于只包含单个点的 Q' 来说也是有问题的，这些点在每个时间窗内都会被重复计算，这是有违直觉的。这相当于选择某个用户为中心点后，对用户进行了加权，如果一个用户在多个时间窗内都满足了匹配，那么这个用户的权重就增大了。然而这只是一个用户，可能该用户的行为具有规律性，但是并不能以一个样本的规律来描述整个群体的规律，因此在多个时间窗内对于中心点的匹配次数应当按照一次计算，而不是多次。这样可以避免单个数据样本对于整个群体规律的影响。

参考[8]中给出的满足反单调性的支持度定义方式，将其扩充到时序图上，使时序图中的支持度定义满足反单调性。即图模式 Q 在时序图 G 中的支持度为它在所有时间窗内所有匹配 $Q(G)$ 中不同中心点 x 的数量。对于一个查询 Q ，在使用时间窗口划分的时序图 G 中的支持度表示为：

$$supp(Q, G) = \left\| \{x \mid \exists x \in Q(x, G_i) \wedge G_i \subseteq G\} \right\| \quad (3.4)$$

在该定义方式下，每个中心点 x 所在的图模式 Q 无论在一个时序子图中匹配到多少次，也无论在多少个时序子图上拥有匹配，最终该中心点 x 只会被计算一次，所以该支持度的定义满足反单调性。该支持度的计算方式与在不使用时间窗口划分的时序图上计算支持度的区别是，可能存在匹配跨不同的时序子图，这些匹配会被排除。即通过时间窗口划分的时序图的支持度计算结果会小于等于在未划分的时序图上的支持度计算结果，这是使用时间窗口过滤掉冗余匹配的结果。

例 14：在该支持度定义下，标签为 $phonenum$ 的单个点的图模式 Q_d' 在时序子图 G_{d1} 中的支持度为 2，标签为 $phonenum$ 的一个点和从该点出发的标签为 $call$ 的一条边组成的图模式 Q_d 在时序子图 G_{d1} 中的支持度为 2，满足反单调性原则。同理，它们在整个时序图 G_d 上也满足反单调性原则。

对于关联规则 $R: Q(x, \tau, s) \rightarrow q(x, y)$ ，可以将其视作带有给定 x 和 y 的图模式 $P_R(x, y)$ ，其在时序图 G 中的支持度表示为：

$$supp(R, G) = \left\| \left\{ x \mid \exists x \in P_R(x, G_i) \wedge G_i \subseteq G \right\} \right\| \quad (3.5)$$

关联规则 R 在时序图 G 中的置信度表示为 $conf(R, G)$ ，代表该关联规则能够被接受的可信度，置信度越高，代表该关联规则越可靠。

关联规则的置信度可以从“正例”、“反例”和“未知”三个方面考虑。最直观的置信度定义为关联规则 R 的支持度与图模式 Q 的支持度的比值，该定义方式仅考虑了“正例”对置信度的贡献，形式化表示如下：

$$conf(R, G) = \frac{supp(R, G)}{supp(Q, G)} \quad (3.6)$$

在时序图中依然可以引入局部世界假设^[66]，引入后增加额外的几个支持度定义。

对给定预测 q 的支持度，用点 x 在时序图中的匹配次数来表示， x 代表满足 $q(x, y)$ 的点。其定义如下：

$$supp(q, G) = \left\| \left\{ x \mid \exists x \in P_q(x, G_i) \wedge G_i \subseteq G \right\} \right\| \quad (3.7)$$

对不满足预测 q 的支持度，用点 u 在时序图中的匹配次数来表示， u 代表与 x 有相同标签的点，并且具有 q 这条边，但是在任意时间窗的子图内都不属于

$P_q(x, G_i)$ 。其定义如下：

$$supp(\dot{q}, G) = \left\| \left[u \mid u.label = x.label \wedge u.q = x.q \wedge u \notin \forall P_q(x, G_i) \right] \right\| \quad (3.8)$$

对满足查询时序图模式 Q 但不满足预测 q 的支持度，用点 u 在时序图中的匹配次数来表示， u 是不满足预测 q 的支持度中的点并且存在 u 属于 $Q(x, G_i)$ 。其定义如下：

$$supp(Q \dot{q}, G) = \left\| \left[u \mid u.label = x.label \wedge u.q = x.q \wedge u \notin \forall P_q(x, G_i) \wedge \exists u \in Q(x, G_i) \right] \right\| \quad (3.9)$$

根据[8]可以定义基于贝叶斯因子的置信度表示：

$$BFconf(R, G) = \frac{supp(R, G) \times supp(\dot{q}, G)}{supp(Q \dot{q}, G) \times supp(q, G)} \quad (3.10)$$

当 $supp(Q \dot{q}, G) = 0$ 时，表明该关联规则在时序图 G 上对于任意 x 都适用；当 $supp(q, G) = 0$ 时，表明该关联规则没有满足 $q(x, y)$ 的 x 点。这两种情况都表明该关联规则是平凡的，没有实际价值，因此在挖掘过程中需要舍弃掉此类关联规则。

置信度的定义方式会影响挖掘算法得到的最终关联规则的准确度，而直观的置信度表示因为仅考虑“正例”的影响，所以准确度相对较差。 $BFconf(R, G)$ 对于“正例”、“反例”和“未知”情况都会考虑，可靠性更高，所以本文将采用基于贝叶斯因子的置信度表示方法来衡量时序图关联规则的有效性。

3.3.2 发现问题描述

时序图关联规则的发现问题与一般的图关联规则发现问题的目标一致，都是需要找到在给定图 G 上准确率最高的一组关联规则。所以可以将时序图关联规则的发现问题描述为如下形式：

- (1) 输入：给定时序图 G ，需要预测的事件 $q(x, y)$ ，时间窗的大小 τ 和每次滑动的步长 s ，支持度剪枝阈值 σ ，规则包含的最大边数 d ，最终选出规则的个数 k ；
- (2) 输出：包含置信度最高的 k 个关联规则的优先队列 L_k 。对于每一个规

则 $R \in L_k$ ，都满足边数 $|E_R| \leq d$ ，都能够预测出事件 $q(x, y)$ ，并且规则的有效作用时间范围为时间窗大小 τ 。

针对该问题，最容易想到的方法便是枚举所有包含事件 $q(x, y)$ 的时序图模式，依次计算这些时序图关联规则在时序图 G 上的置信度，选择置信度最高的前 k 个规则。但是这样做会扩展生成大量的候选关联规则，由于时序图匹配的时间复杂度较高，计算如此多的关联规则的支持度将导致发现算法无法完成，所以需要考虑一系列更加有效的剪枝和搜索策略来解决这个问题。

3.3.3 发现算法

时序图关联规则的发现算法 TGRD (Temporal Graph association Rules Discovering algorithm) 采用搜索和剪枝的设计思想，通过给定的最大扩展边数 d ，进行总共 d 轮的时序图关联规则发现。发现框架主要由计算有效规则和扩展候选规则两部分组成。在该发现框架下，关联规则按照树形结构进行扩展，每轮的候选关联规则都由上一轮满足支持度阈值的规则扩展得到，具有较高搜索效率。对于每一轮候选集中的任意关联规则的发现步骤如图 13 所示。

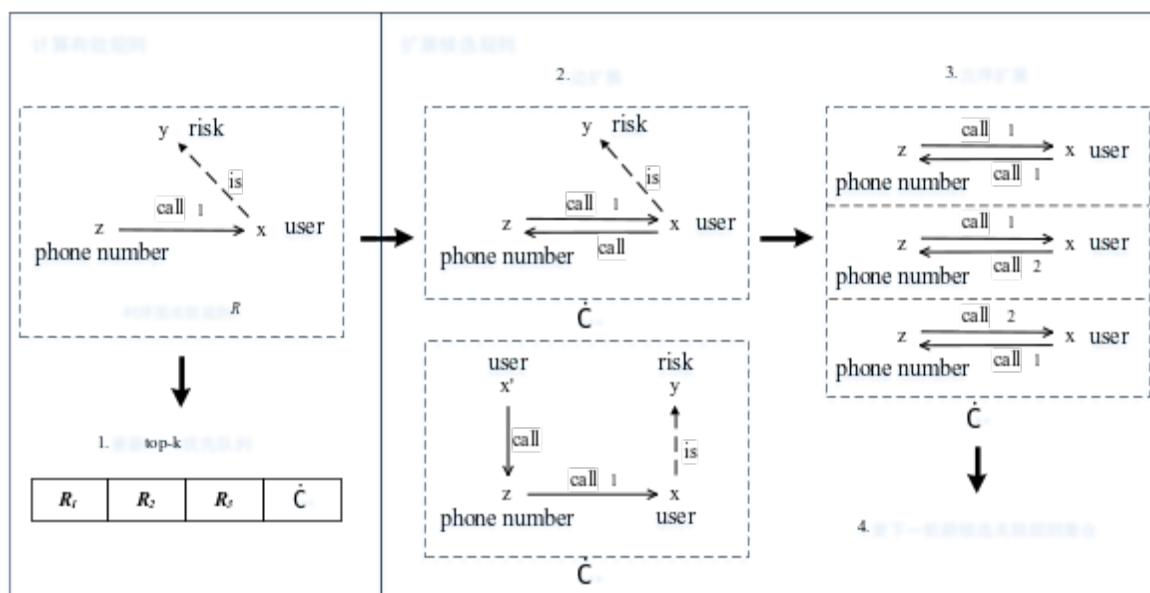


图 13 时序图关联规则发现过程示意图

计算有效规则。 规则的有效性通过置信度来衡量，有效性高的规则放入 top-k 优先队列中存储。top-k 优先队列是以置信度作为优先级进行排序的一个大小为 k 的队列，

用于记录发现过程中置信度最高的 k 个时序图关联规则。对当前规则 R 会计算它的置信度，以此作为是否将其放入 top- k 优先队列的衡量标准。

扩展候选规则。以时序图关联规则 R 为基础，通过每轮增加一条边的方式扩展生成下一轮所需候选关联规则。为了保证搜索的高效性，使用支持度进行剪枝，只有当前规则的支持度不小于设定的支持度阈值时，才会对该规则进行扩展。扩展的过程由边扩展和次序扩展两部分组成。边扩展会尝试以规则 R 中的点集合中的点作为起点或者终点进行扩展，扩展所需另一个点可以是新增点也可以是点集合中已有的点。如图 13 中的边扩展：一个示例是以点 x 作为起点，集合中已有点 z 作为终点，扩展新增加一条边；另一个示例是以点 z 作为终点，新增一个点 x' 作为起点，扩展它们之间的一条边。次序扩展会确定新增加的边与规则 R 中原有边的次序关系，如图 13 中的次序扩展确定了拥有次序的边之间可能的次序关系。经过边扩展和次序扩展后，得到扩展后的时序图关联规则，放入候选集中，用于下一轮的计算。

根据时序图关联规则的发现框架，设计并给出时序图关联规则发现的算法，伪代码表示如下表所示。

表 2 时序图关联规则发现算法主流程

算法：TGRD

输入：时序图 G ，预测边 $q(x, y)$ ，时间窗 τ 和步长 s ，剪枝阈值 σ ，选出数 k ，扩展边数 d

输出：时序图关联规则优先队列 L_k

1. $r := 1$; $L_k := \emptyset$; $M := [q(x, y)]$;
2. $Partition(G, \tau, s)$; // 使用时间窗口对时序图进行划分
3. **while** $r \leq d$ **do**
4. $r := r + 1$;
5. $R_{Set} := M$;
6. $M := \emptyset$;
7. **for each** $R \in R_{Set}$ **do**
8. 计算 $supp(R, G)$ 和 $conf(R, G)$
9. **if** $L_k.\text{len} < k$
10. $\text{if } conf(R, G) > \sigma$
11. $L_k.checkAndPush(R)$;

```

11. else if  $conf(R, G) > L_k.top().conf$  and  $L_k, < R, conf(R, G) > \textcolor{red}{i}$   

   !existInL $\textcolor{red}{i}$ 
12. then  $L_k.pop()$  ;  $\textcolor{red}{i} R, conf(R, G) > \textcolor{red}{i}$   

    $L_k.push \textcolor{red}{i}$  ;
13. if  $supp(R, G) \geq \sigma$  then  $M := ExpandEdge(M, R)$  ;
14. return  $L_k$  ;

```

初始化相应的数据结构，关联规则候选集中最初只有预测边 $q(x, y)$ （算法第 1 行）。对时序图 G 按照给定的时间窗大小 τ 和滑动步长 s 进行划分，时间窗口的参数可以根据数据的时间周期特性依照经验选取（算法第 2 行）。时序图关联规则的发现过程按轮数进行，一共有 d 轮，每轮会从关联规则候选集中遍历每个规则 R ，通过规则 R 更新 top-k 优先队列和扩展下一轮的候选关联规则。

更新 top-k 优先队列（算法 8-12 行）。首先计算当前关联规则 R 的支持度和置信度，如果 top-k 优先队列的大小未达到设定的 k 值，则调用 $checkAndPush$ 函数，该函数的作用是如果给定的关联规则在优先队列中不存在，则将其放入优先队列中，新的关联规则放入后，优先队列会自动按照置信度的大小进行升序排序；如果队列已满，只有当新的关联规则的置信度大于该优先队列中头部的关联规则的置信度，并且新的关联规则在队列中不存在时，才将优先队列头部的关联规则弹出，把新的关联规则放入优先队列中。

扩展下一轮的候选关联规则（算法第 13 行）。如果当前关联规则 R 的支持度低于给定的支持度阈值 σ ，则过滤掉该关联规则，否则对该关联规则进行扩展，边扩展过程通过 $ExpandEdge$ 函数完成，接下来将对边扩展的具体做法进行详细描述。

边扩展函数 $ExpandEdge$ 主要包括加边操作和次序扩展操作两个部分，前者用于为一个时序图关联规则增加一条边，后者用于为加边后的时序关联规则确定边上的次序。边扩展函数的伪代码描述如下表所示。

表 3 边扩展函数

函数：ExpandEdge

输入：候选集合 M ，待扩展规则 R

输出：扩展后的规则集 M

```

1. for each  $v \in V_R$  do
2.   if  $v.id = \textcolor{red}{i} R.link.y$  then continue;
3.    $new\_label := outerHash(v.label)$  ;

```

```

4.    $eLabel := v2eHash(v.label, new_{label})$  ;
5.    $Q := R$  ;
6.    $uid := Q.addVertex(new_{label})$  ;
7.    $eid := Q.addEdge(v.id, uid, eLabel)$  ;
8.   if  $E_R[eid].hasOrder()$  then  $M := expandOrder(M, Q, eid)$  ;
9.   else  $M.checkAndPush(Q)$  ;
10.  for each  $u \in V_R$  do
11.    if  $u.label \neq new_{label}$  then continue;
12.     $Q := R$  ;
13.     $eid := Q.addEdge(v.id, u.id, eLabel)$  ;
14.    if  $E_R[eid].hasOrder()$  then  $M := expandOrder(M, Q, eid)$  ;
15.    else  $M.checkAndPush(Q)$  ;
16.  同理, 将  $v$  作为终点, 使用  $innerHash$  获取起始点标签, 进行加边扩展
17.  return  $M$  ;

```

边扩展函数会遍历规则 R 的点集合 V_R , 对于任意 $v \in V_R$, 当该点不是预测边 $q(x,y)$ 的 y 点时, 分别以该点 v 作为起点和终点来进行加边扩展 (算法 1-2 行)。首先将 v 作为起点进行加边扩展, 通过 $outerHash$ 获取终点的标签 new_{label} (如果将 v 作为终点, 则通过 $innerHash$ 获取起始点标签), 通过 $v2eHash$ 获取边的标签 $eLabel$ (算法 3-4 行)。映射关系 $outerHash$ 、 $innerHash$ 和 $v2eHash$ 可以通过预处理扫描全图信息来构造, 从而过滤掉无效的标签组合, 详细优化方案见下一小节优化策略部分。加边所需的另一个点可以是新增的点, 也可以是当前规则的点集合中已经存在的点, 这两种情况都会进行扩展。将另一个点 u 作为新增的点, 先增加新的标签为 new_{label} 的点 u , 再增加 v 和 u 之间的标签为 $eLabel$ 的边, 如果新增加的边带有时序关系, 则调用次序扩展函数 $expandOrder$ 进行次序扩展, 否则直接调用 $checkAndPush$ 函数, 检查该规则在候选集中是否存在, 如果没有重复的话, 则将其加入到候选集中 (算法 5-9 行)。将另一个点 u 作为该规则点集合中已经存在的点, 去遍历点集合 V_R 找到标签相同的点, 在 v 和 u 之间加一条边, 同理如果新增的边有时序关系, 则进行次序扩展, 否则直接将新规则加入候选集 (算法 10-15 行)。将 v 作为终点进行加边的过程与作为起始点时的过程相同 (算法第 16 行)。

次序扩展函数 $expandOrder$ 主要采用保证次序的思想，因为关联规则 R 中的各边已经具有连续的次序，且该规则具有较高的支持度，因此新扩展的边在保证已有的边的次序情况下确定自己的位置，便可覆盖所有的满足支持度的情况。具体描述如下表所示。

表 4 次序扩展函数

函数： $expandOrder$

输入：候选集合 M ，待扩展次序的规则 Q ，扩展的边索引 eid

输出：扩展后的规则集 M

```

1.    $x := 0$  ;
2.   计算  $Q$  中非预测的边具有的最大次序  $max_{order}$  ;
3.   while  $x < max_{order}$  do
4.      $x := x + 1$  ;
5.      $E_Q[eid].order := x$  ;
6.      $M.checkAndPush(Q)$  ;
7.      $Q_m := Q$  ;
8.     for  $e \in E_{Q_m}$  do
9.       if  $e.order \geq x$  then  $e.order +\textcolor{red}{1}$  ;
10.       $E_{Q_m}[eid].order := x$  ;
11.       $M.checkAndPush(Q_m)$  ;
12.       $E_Q[eid].order := max_{order} + 1$  ;
13.      if  $Q.link.hasOrder()$  then  $Q.link.order +\textcolor{red}{1}$  ;
14.       $M.checkAndPush(Q)$  ;
15.    return  $M$  ;

```

首先计算规则 Q 中的除了预测边之外的所有边所具有的最大次序值 max_{order} （算法第 2 行）。 x 的取值从 1 开始直到最大次序值 max_{order} 结束，接下来确定新增加的边索引为 eid 的边的可能次序值，并调整已有边的次序值，从而得到次序扩展后的新的候选关联规则：新增加的边可能与已有边的次序相同，因此直接将新增加的边的次序设置为 x 后，将该规则进行重复检查后加入关联规则的候选集（算法 5-6 行）；新增加的边的次序可能为 x ，但没有与它次序相同的边，因此将规则的边集合中的次序值大于等于 x 的边的次序都增加 1，新增加的边的次序设置为 x ，将该

关联规则进行重复检查后加入候选集（算法 7-11 行）；新增加的边的可能是最后发生的，则将新加边的次序设置为 $\max_{order} + 1$ ，如果预测边具有次序的话，则将预测边的次序增加 1，因为它要发生在新增加的边之后，将扩展后的规则进行重复检查后加入候选集（算法 12-14 行）。

例 15：边的次序扩展示意图。如图 14 左侧所示时序图模式，已经新扩展了一条边 e_5 ，现在对其次序进行扩展，图 14 右侧展示了进行次序扩展前的边的次序排列情况。根据已有边的次序信息，容易计算得到 $\max_{order} = 3$ 。在保证原排列的有序性的前提下，确定新扩展的边的次序值。首先考虑新扩展边与已有边的次序相同的情况，则新扩展边的 $order$ 可能取值为 1、2、3，分别扩展后放入候选规则集合。然后考虑新扩展边在已有边的次序间隔中的情况，则当 $order=1$ 时，其它次序值不小于 $order$ 值的边的次序都增加 1，将该扩展规则放入候选规则集合。 $order$ 可以取不大于 \max_{order} 的任意值，对于 $order$ 的其它取值情况，采用相同的方式调整边的次序。最后 $order$ 取值为 4，同时将预测边 $link$ 的次序值调整为 5，放入候选规则集合。

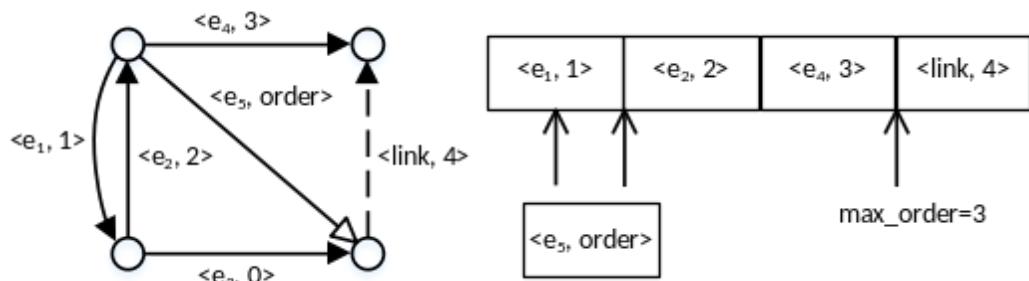


图 14 边时序扩展示意图

3.3.4 优化策略

为了加速时序图的关联规则的发现过程，结合实际情况，提出了一些优化的策略。这些策略主要从减少候选集的数量和加速匹配过程等方面考虑，使用这些策略可以提高关联规则发现算法的速度。

(1) 扩边引导。在一个具有 N 种类型的实体和 M 种类型的事件的时序图中，并非每种类型的实体都拥有 M 种类型的事件与 N 种类型的实体相连，即图中存在的（实体标签，实体标签，事件标签）的组合情况上限是 $N \times N \times M$ ，但是在实际的时序图中，可能存在的组合数量要远远小于该上限。因为一种类型的实体与另一种

类型实体间的标签往往只有少数的 1-2 种。如通话网络中用户对电话号码的打电话和发短信事件，社交网络中用户对用户推文的转发和评论，对于将属性抽出作为实体的图，这类实体和属性之间的边的类型是唯一的，如 $(user, 29, age_{is})$ 、 $(user, Peter, name_{is})$ 等。

因此在读取时序图之后，可以通过预处理的方式，对实体类型与实体类型和它们之间对应的事件类型关系做一个哈希映射，从而在发现过程中根据起始点和终点的类型来增加相应的边类型，更好地指导边扩展，避免对所有组合的枚举导致大量无用候选关联规则的产生。存储的哈希结构如图 15 所示。

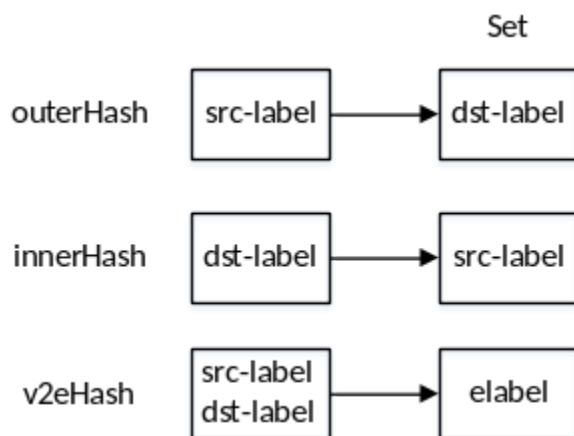


图 15 可扩展边信息哈希结构图

其中 $v2eHash$ 是利用初始点标签和终点标签共同作为哈希索引来查找对应的边标签集合。 $outer Hash$ 是通过起始点标签查找对应的终点标签集合， $inner Hash$ 是通过终点标签查找对应的起始点标签集合。因为对边进行扩展时，可以增加一条出边或者入边，所以这两个结构都是必需的。

通过该映射结构还可以去除低频率的边标签扩展。使用该映射结构可以计算每种边标签 c 经由点标签 a 到点标签 b 的出现频次，该频次在总频次中的占比就是该边标签扩展 (a, b, c) 的频率，通过设定 $frequency$ 阈值，可以对低于该阈值的边标签进行剪枝。出现频率较低的边标签意味着所占权重较小，相比频率高的边标签而言，对于图模式的匹配结果影响小。而挖掘过程中本身就会使用支持度进行剪枝，因

此去除这部分边标签的扩展可以有效的加速发现过程，并对发现结果几乎不会产生影响。

(2) **OpenMP 多线程加速。**发现算法主要耗费时间的地方在于支持度的计算，因为子图同构本身是较为复杂的 NP 问题，并没有多项式时间内的解法，虽然单个支持度的计算无法加速，但是可以考虑其他的加速方案来提高整个发现过程的效率。在同一轮的时序图关联规则的发现过程中，关联规则支持度的计算和候选关联规则的生成过程是相互独立的，没有资源互斥的限制。因此该部分算法可以通过多线程加速，从而提高计算效率，使用 OpenMP 可以便捷地实现多线程加速策略。只有在将候选规则放入候选集合和将满足的新规则放入 top-k 队列时会出现资源互斥的情况，可以使用 OpenMP 的锁来实现临界区资源的访问，此处对并发度影响不大。

(3) **设置匹配时间阈值。**如果一个图模式在图中的匹配时间超过阈值，则停止匹配返回已经匹配到的结果。因为挖掘过程中会通过置信度来衡量一个关联规则是否可靠，而置信度很大程度上要依赖该关联规则在图中的支持度。当一个图模式在图中迟迟无法返回匹配结果时，说明子图同构算法在不停地枚举可能的尝试，但是匹配结果较少，即使等待超长时间完成匹配，匹配的结果几乎为零或者拥有极低的支持度，这对于整个挖掘过程得不偿失。

根据图模式 Q 在时序图 G 中的置信度定义可知，会在对时序图划分后的子图中计算匹配到的所有不同的 x 的数量。这意味着在当前子图匹配上的 x 不会参与到下一个子图中匹配，如果未匹配上，则会参与到下一个子图继续匹配。据实验观察，如果一个图模式在每个子图中匹配未超时，但是匹配速度都很慢，最终匹配得到的结果为零或者极低的支持度。因此除了设置图模式在单个子图中的匹配超时阈值外，还可以通过设置在整个时序图时间窗内的匹配超时阈值，进行有效的剪枝。

(4) **时间窗口划分。**时序图关联规则给出了滑动窗口的概念，整个时序图可以被滑动窗口分为多个时序子图，这些时序子图是原时序图的一个覆盖，时序子图之间可以有重叠的部分。时间变量虽然是线性的，但是人为产生的事件在时间作用下往往具有周期性，因此时间窗口可以很好地描述这个现象，但重叠部分意味着计算量的增加和部分的重复计算。

在一些标志性明显且具有人为因素影响的事件网络中，对图的划分可以由人为产

生的因素来决定，例如一组商品在一段时间内经由不同活动折扣而呈现的购买行为所组成的网络，依据不同场次活动的开始时间和结束时间就可以对整个购买网络进行一个合理划分，从而研究在不同活动条件下用户的购买行为。

而对于一些没有明显标志性开端和结束的事件组成的网络图，按照天、周、月、年等自然的时间单位划分往往是一种效果较好且时间成本较低的做法，划分之后的各个子图之间没有重叠，例如通话网络、交通网络，都具有明显的天周期特性。划分之后，过早（在上一个子图中）或过晚（在下一个子图中）的冗余事件信息都不会对现阶段（在当前子图中）的匹配结果产生影响。

(5) 动态支持度剪枝。在关联规则的发现过程中，生成的候选规则会随着扩展边数而越来越多，使用固定的支持度阈值进行剪枝，可以有效地过滤掉部分支持度较低的关联规则，从而加速发现过程。但是在实际应用中，最终得到的所需关联规则往往是按置信度排序的 top- k 个规则，其中 k 的取值不会很大。过多的候选关联规则会导致发现过程缓慢，但对于最终发现的关联规则影响最大的是支持度较高的规则，因此可以通过动态支持度的方式对候选关联规则进一步剪枝。

按照原有的边扩展规则，每一轮进行边扩展时，会检查当前关联规则的支持度是否大于设定的支持度阈值，如果满足则进行加边扩展，生成新的候选关联规则，否则不进行扩展。假设 top- k 队列的容量为 K ，扩展后得到新的候选关联规则集合 M 。动态支持度剪枝策略会根据候选关联规则集合的大小 $|M|$ 进行筛选：如果其数量不大于 C 倍的 K ，则集合内的候选关联规则全部进入下一轮迭代；如果其数量超出 C 倍的 K ，则将候选集合按照加边扩展前的支持度进行降序排序，选择前 CK 个规则以及后面的一定比例的规则进入下一轮的迭代。选取规则的数量 N 的表示如下：

$$N = CK + \lfloor P(|M| - CK) \rfloor \quad (3.11)$$

其中， C 代表倍数，在正数范围取值， P 代表超出部分的选择比例，取值范围是在 0~1 之间的有理数，对于 P 与超出部分的乘积进行向下取整。这样相当于对首次支持度剪枝后扩展得到关联规则又进行了一次动态支持度剪枝，从而避免在使用固定支持度剪枝阈值时随着扩展边数增大而出现大量的候选关联规则。

动态支持度并不是直接改变支持度的阈值，而是通过给定参数筛选一定比例的候选规则进入下一轮扩展，从而间接确定了一个支持度阈值，将低于该支持度阈值的候选规则进行了剪枝，从而起到了高效过滤的效果。

3.3.5 复杂度分析

边次序扩展的复杂度。 给定一个边的规模为 $|E|$ 的时序图模式，已经为其扩展了一条边，现在考虑对这条边进行次序扩展。因为时序图模式的规模很小，调整边的次序的过程消耗时间极少，所以此处不再考虑次序扩展的时间复杂度，而是考虑扩展出的时序图关联规则的数量。因为时序图匹配问题复杂度较高，所以扩展出的候选关联规则的数量将影响发现算法的效率。

如果采用枚举的方式进行次序扩展，这些边的发生顺序可以任意组合，即使不考虑同时发生的情况，任意组合的情形是算上新扩展边的边的次序的全排列，产生的候选关联规则的个数为 $|E+1|!$ ，即图模式的边的数量的阶乘。如果考虑边同时发生的情况，将会产生更多的关联规则，这无疑会导致发现过程缓慢。采用边次序扩展方法将具有更好的扩展效果，每轮新增的边的次序或者与现有边的次序相同（ $|E|$ 种扩展），或者在现有次序之间（ $|E+1|$ 种扩展），因此产生的候选关联规则的个数为 $|2E+1|$ 种。因为关联规则本身所具有的边的数量有限，在扩展边数 $d \leq 5$ 的情况下，该方法产生的候选关联规则数量是可计算的，并且相比枚举而言能大幅度提高发现算法的效率。当 $d=4$ 时，该扩展方法会产生 9 个关联规则，而枚举方法在不考虑次序相同情况下会产生多达 120 个关联规则。边次序扩展方法将候选关联规则的个数从排列组合数变为线性数，从而有效地提高了发现算法的效率。

发现算法的时间复杂度。 从每轮来看，发现所需的时间由计算支持度和置信度、更新队列和生成候选集的时间组成。其中消耗时间最多的是计算支持度的时间，因为采用子图同构的匹配语义，本身就具有较高的时间复杂度。

假设关联规则 R 在时序图 G 上的匹配时间为 $T_1(R, G)$ ，因为采用了子图同构语义，所以该 T_1 函数具有指数的时间复杂度。通过对时序图进行划分，当划分均

等并采用多线程计算时，可以将每次匹配的计算时间变为 $NT_1(R, G/N)$ 。如果第 i 轮的候选关联规则数量为 A_i ，则计算支持度所需的时间为 $A_i NT_1(R, G/N)$ 。置信度的计算通过支持度的数值来完成，只需要 $O(1)$ 的时间复杂度。

更新队列需要进行规则的重复检测，判断新增规则与队列中规则是否一致，最多消耗时间为 $2kT_2(R_m, R_n)$ ， k 为队列的大小，因为基于子图同构语义，所以 T_2 函数具有指数的时间复杂度，因为判断两个规则是否一致只需要判断两个规则是否互为对方的子图匹配，采用子图同构的算法进行两遍计算即可。由于图模式规模较小，因此该过程不会消耗太多时间。最坏情况下第 i 轮内的每个规则都需要更新队列，此时更新所花费的时间不超过 $2A_i k T_2(R_m, R_n)$ 。

对候选关联规则集合的更新也需要进行规则的重复检测，如果支持度不小于剪枝阈值的关联规则扩展出的下一轮新规则的数量为 A_{i+1} ，因为采用固定的边扩展策略和剪枝策略，所以每轮 A_{i+1} 增加的值有限，可看作与前一轮呈线性关系，即 $A_{i+1} = aA_i$ 。当产生的所有关联规则都不相同时，更新规则集合所进行的比较次数是一个首项为 0 公差为 1 数量为 aA_i 的等差数列，取得最大消耗时间不大于 $(aA_i)^2 T_2(R_m, R_n)/2$ 。

因为采用多线程加速，在更新队列和候选集时会有资源互斥情况发生，并且发生概率随着线程数增加而增大，冲突次数也与本轮需要访问的 top-k 队列的关联规则数量 A_i 和生成的下一轮候选关联规则数量 A_{i+1} 呈正比，将该部分时间消耗表示为 $T_3(N, A_i)$ 。

综上所示，时序图关联规则发现算法的第 i 轮扩展的时间复杂度上界可以表示为 $O(A_i NT_1 + (2A_i k + (aA_i)^2/2)T_2 + T_3)$ ，并且由于 $|G/N| \gg |R|$ ， T_1 是与 $|G/N|$ 有

关的指数函数， T_2 是与 $|R|$ 有关的指数函数， T_3 是与 N 和 A_i 有关的线性函数，所以 $A_i N T_1 \gg (2A_i k + (aA_i)^2/2)T_2 \gg T_3$ 。当给定发现算法的扩展轮数 d 后，将各轮时间消耗求和就是发现算法总的时间复杂度。从中可以看出， $A_i N T_1(R, G/N)$ 消耗的时间远远大于其他部分，降低该部分时间消耗可以通过降低时序图 G 的规模、增加线程 N 的数量和减少候选规则 A_i 的个数来实现。

3.4 本章小结

本章作为论文的主体部分之一，主要研究了时序图关联规则及其发现问题。

针对时序图关联规则：首先明确了时序图、时序图模式和时序图模式匹配的定义，在一般图的边上增加了时间戳，用于描述事件发生的时间信息；然后在此基础上，提出了一种考虑时序关系和时效性的时序图关联规则，该规则以时序图模式为基础，考虑时序网络中的事件发生次序，具有更丰富的表达能力，并且通过时间窗口的限制，约束规则的作用时间范围，保证规则的时效性。

针对规则的发现问题：首先对支持度和置信度的定义进行了扩充，结合时序图的特性，使其适用于时序图；然后提出了时序图关联规则的发现算法，算法主要步骤包括候选规则的生成、top-k 队列的更新等，并且针对发现算法提出了一系列优化策略，包括扩展边时根据标签组合情况和频率进行合理引导，使用多线程加速支持度计算过程，设置时序图模式匹配的时间阈值，合理使用时间窗口对时序图进行划分，以及设置动态的支持度进行剪枝等优化策略，使用这些优化策略可以极大提高发现算法的效率；最后对发现算法的复杂度进行了理论分析。

第四章 基于时序图关联规则的事件预测

本章将研究基于时序图关联规则的事件预测问题。首先给出了该问题的形式化表达；然后给出了时序图匹配算法 TGPM (Temporal Graph Pattern Matching algorithm)，并进行了复杂度的理论分析，证明时序图匹配的复杂度并不显著高于图匹配；接下来研究了事件预测的并行化解决方案，给出了基于时间窗的图划分方法，以及基于时序图关联规则的事件预测并行算法 TGREP (a parallel algorithm for Event Prediction based on Temporal Graph association Rules)，并证明该算法具有良好的并行可扩展性。

4.1 问题描述

给定一个时序图关联规则集合 Σ ，它是由一组具有相同预测 $q(x, y)$ 的关联规则组成，通过这些关联规则都可以做出对给定事件的预测。将 Σ 中的关联规则依次在时序图 G 上计算匹配结果，将该匹配的实体结果存入 $\Sigma(x, G)$ 中，形式化表示如下：

$$\Sigma(x, G) = \{ u_x \vee u_x \in Q(x, G_i) \wedge Q(x, G_i) \Rightarrow q(x, y) \in \Sigma \} \quad (4.1)$$

其中， $G_i \subseteq G$ ，是使用时间窗口对时序图进行划分得到的时序子图。

所以可以将基于时序图关联规则的事件预测问题描述为如下形式：

- (1) 输入：给定时序图 G ，一组预测事件 $q(x, y)$ 的时序图关联规则集合 Σ ，时间窗的大小 τ 和每次滑动的步长 s ；
- (2) 输出： $\Sigma(x, G)$ ，该匹配结果集合中的中心点 x 均满足预测事件 $q(x, y)$ 。

针对该问题，可以很自然地给出对应的串行算法：依次从集合 Σ 中获取关联规则 R ，对每个规则中的时序图模式 Q 计算其在时序图 G 上的匹配结果。若存在匹配结果则说明关联规则 R 在时序图 G 上找到一组预测，将匹配到的大图信息存储到结果集中。

但是在实际应用中，许多任务具有时效性，要求在较短时间内能够在一定规模的

图数据上完成任务。而串行算法无论从速度上还是对于大图规模的支持上都存在局限性，因此设计算法的并行化方案是非常有必要的。

并行可扩展性。为了更好地衡量并行算法的效率问题，按照[67]中的研究给出并行可扩展性的形式化表示。考虑图 G 上的一个问题 A ，将使用串行算法解决该问题的最坏时间复杂度表示为 $t(|A|, |G|)$ ，对于一个并行算法，将使用 n 个处理器解决该问题的时间复杂度表示为 $T(|A|, |G|, n)$ ，因为图的规模很大，所以有 $n \ll |G|$ 。

一个算法是并行可扩展的，如果它的时间复杂度满足如下形式：

$$T(|A|, |G|, n) = O(t(|A|, |G|)/n + (n|A|)^{O(1)}) \quad (4.2)$$

该公式说明并行算法能够实现串行算法运行时间的多项式缩减，额外增加的运行时间是与图的规模 $|G|$ 无关的量。因此如果一个算法满足并行可扩展性，则它在图 G 上解决问题 A 的运行时间一定会随着处理器的个数增加而减少。

所以满足并行可扩展性的算法能够更好地解决基于时序图关联规则的事件预测问题，具有更好的实际应用效果。

4.2 时序图的匹配研究

因为使用时序图关联规则进行事件预测需要在时序图上进行匹配，所以本节给出了一种时序图匹配算法，在子图同构的匹配过程中考虑边之间的时序关系，确定时序图中边的匹配，该算法通过时序剪枝策略可以提前过滤掉不符合的匹配，具有较好的匹配效率。最后给出了该算法复杂度的理论分析。

4.2.1 匹配算法

因为时序图匹配时需要考虑边上的时序信息，所以基于子图同构匹配进行改进，设计时序图匹配算法 TGPM (Temporal Graph Pattern Matching algorithm)。在时序图模式匹配的定义中，进行匹配的图模式的边的次序 (o) 与匹配子图上边的时间戳 (t) 顺序相同，将其表述为时序剪枝策略，形式化表示如下：

$$\forall e_i, e_j \in E_Q \wedge O(e_i) \oplus O(e_j) \leftrightarrow \forall e'_i, e'_j \in E \wedge T(e'_i) \oplus T(e'_j) \quad (4.3)$$

时序图匹配算法在子图同构算法的基础上增加了该条时序剪枝策略。在每次将满足的候选点对加入到部分匹配结果后，会对部分匹配结果进行时序匹配，从而验证该部分匹配结果是否满足边之间的时序关系，若不满足则提前剪枝。接下来将给出时序剪枝策略的具体实现，称为次序匹配算法。

使用图 16 所示的结构来存放时序图模式和时序图之间的时序信息。

OrderQueue 是一个优先队列，存储着 $\text{pair}(order, TimeQueue)$ ，其中 *order* 值为查询 Q 中每条边的次序值，该优先队列按次序值升序排列。*TimeQueue* 也是一个优先队列，存储着查询 Q 中的边所对应的图 G 中的边的时间戳，按照时间戳的值升序排列。查询 Q 中的一条边在图 G 中可能会与多条边相匹配，因此它们的时序对应关系可以是一对多。默认时序 0 为无时序状态，当查询 Q 中的边次序为 0 时表明其不考虑时序，即不放入 *OrderQueue* 中。

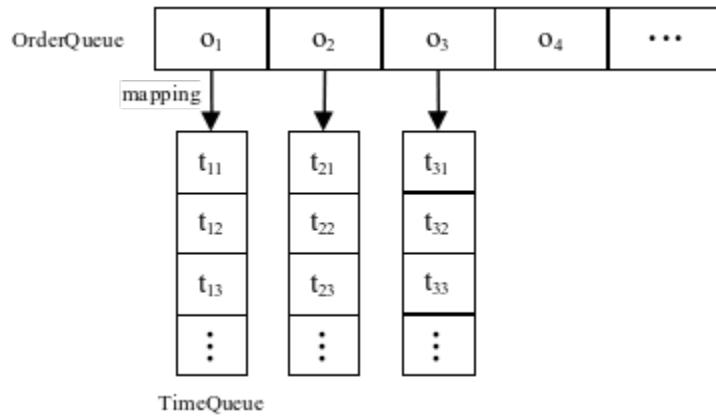


图 16 时序信息数据结构图

次序匹配算法如下表所示，输入为查询 Q 和时序图 G 中点的映射关系，输出为一个布尔值，如果为 *True* 代表次序匹配成功，如果为 *False* 代表不匹配。

表 5 次序匹配算法

算法：Order Match

输入： $\text{Match}(Q, G)$

输出： True/False

1. 根据 $\text{Match}(Q, G)$ 构造 *OrderQueue* 和 *TimeQueue*

```

2. while !OrderQueue.empty() do
3.   OrderList:=PopSameOrder(OrderQueue) ;
4.   interSet:=U ;
5.   for each TimeQueue ∈ OrderList do
6.     interSet:=interSet ∩ TimeQueue ;
7.   if interSet.empty() then return False;
8.   timelimit:=interSet[0] ;
9.   for each TimeQueue ∈ OrderQueue do
10.    while !TimeQueue.empty() && TimeQueue.top()≤timelimit do
11.      TimeQueue.pop() ;
12.    if TimeQueue.empty() then return False;
13. return True;

```

首先根据匹配 $\text{Match}(Q, G)$ 构造如图 15 所示的时序信息数据结构图（算法第 1 行）。然后根据查询 Q 构造的 OrderQueue 队列中包含升序的次序值，当 OrderQueue 队列不为空时，每次从中弹出相同次序值所对应的 TimeQueue 组成的 OrderList （算法 2-3 行）。假设全集为 U ，将 U 与 OrderList 中的所有 TimeQueue 取交集，得到所有 TimeQueue 相交的时间戳（算法 4-6 行）。如果该交集为空，则表明次序匹配失败，返回 False （算法第 7 行）。取交集中最小的时间戳作为 $\text{time}_{\text{limit}}$ ，对于 OrderQueue 对应的 TimeQueue 队列进行遍历，弹出所有不大于 $\text{time}_{\text{limit}}$ 的时间戳，如果弹出后 TimeQueue 队列为空，则表明没有剩余的时间戳用于之后的查询 Q 边的次序匹配，次序匹配失败，返回 False （算法 8-12 行）。当 OrderQueue 队列完整遍历结束后，表明查询图模式 Q 中的每条边的次序在图 G 中都有相应的时间戳匹配，次序匹配成功，返回 True （算法第 13 行）。

接下来通过一个具体的示例描述次序匹配的算法过程，该示例的时序图模式和构建的时序信息数据结构如图 17 所示。

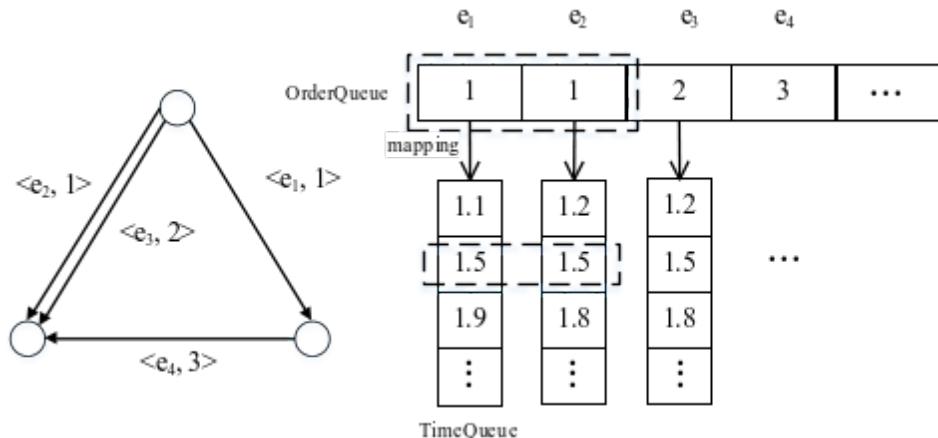


图 17 次序匹配处理流程示意图

例 16：次序匹配算法示例。给定如图 17 左侧所示的图模式，使用 $\langle e, \text{order} \rangle$ 表示边和对应的次序，假定这些边都有相同的标签。假设根据它的边在时序图上的映射关系构造的时序信息结构如图 17 右侧所示。 TimeQueue 中存放时序图上对应的时间戳，按升序排列，并且根据时序图的定义易知任意一个 TimeQueue 中的时间戳均不相等。因为边 e_2 和 e_3 具有相同的起点和终点并具有相同的标签，所以这两条边对应的 TimeQueue 队列是相同的。首先从 OrderQueue 中选择相同次序值所对应的 TimeQueue 弹出组成 OrderList ，因为 $O(e_1)=O(e_2)=1$ ，所以弹出它们所对应的 TimeQueue 组成 OrderList 。求 OrderList 中的 TimeQueue 的时间戳交集，从时间戳的对应关系，可以得到交集 $[1.5]$ ，选取最小的值，即 $\text{time}_{\text{limit}}=1.5$ 。然后遍历检查 OrderQueue 中剩余部分，对于 e_3 对应的 TimeQueue ，弹出小于等于 1.5 的时间戳，即弹出 1.2 和 1.5，对 OrderQueue 中的 e_4 也进行同样的处理，若该过程中出现弹出时间戳后队列为空，则说明剩余部分没有合适的时间戳用于后续匹配，次序匹配失败。接下来进行下一轮，从 OrderQueue 中选择相同次序值所对应的 TimeQueue 弹出组成 OrderList ，从 OrderQueue 中弹出 e_3 对应的 TimeQueue ，取得 $\text{time}_{\text{limit}}=1.8$ ，循环操作直到遍历完 OrderQueue 中所有的次序值或中途有不满足次序匹配的情况发生为止。

4.2.2 复杂度分析

时序图匹配算法是在子图同构算法的匹配过程中增加了一条时序剪枝策略，以此来满足时序图匹配，并加速匹配的过程，该剪枝策略具体实现为次序匹配算法。接下来将分别分析次序匹配算法的时间复杂度和时序图匹配算法的时间复杂度。

次序匹配算法的时间复杂度。 次序匹配算法分为构造数据结构和计算次序匹配两个阶段，将分别分析这两个阶段的耗时。

对于构造数据结构，假设当前查询 Q 的边集合为 E ，则构造按边的顺序升序排列的队列 $OrderQueue$ 所需要的平均时间为排序所需时间 $O(|E|\log|E|)$ ，但是其实构造的时间可以缩短为线性时间 $O(|E|)$ ，因为查询 Q 中的边的次序在生成关联规则时就已经确定了，因此可以在生成时将边按照次序存储，从而线性构造 $OrderQueue$ 队列。然后构造查询图中每条边映射到大图中的边组成的队列 $TimeQueue$ ，也是按照升序排列。该映射关系可以在图模式匹配时建立，构造 $TimeQueue$ 需要进行排序，最大耗时为 $O(|E|N\log N)$ ， N 是映射到大图中的具有相同标签的边的最大数量。因为查询图模式中的边不一定都具有次序，只需要构建具有次序的边所对应的 $TimeQueue$ 队列。

对于计算阶段，求 $TimeQueue$ 队列中元素的交集的最坏的情况为将所有队列中的元素遍历一遍，最大时间复杂度为 $O(|E|N)$ 。因为队列中元素有序，所以可以 $O(1)$ 时间拿到最小的顺序作为阈值 $time_{limit}$ 。然后对所有 $TimeQueue$ 队列遍历，弹出小于该阈值的元素。使用固定长度和索引构造的 $TimeQueue$ 队列可以实现 $O(1)$ 时间的弹出元素操作，只需要移动索引标记即可，因此该遍历队列弹出操作最多遍历所有队列，耗时 $O(|E|N)$ 。查询图模式最多具有 $|E|$ 条边进行计算，因此计算阶段所需的时间复杂度最大为 $O(2N|E|^2)$ 。而实际上平均复杂度并没有这么高，因为如果在弹出操作时遍历了所有队列中的元素，那么就意味着元素队列已经为空，此

时计算阶段就会结束。

综上所述，一轮次序匹配的时间复杂度可以表示为 $O(|E|) + O(|E|N \log N) + O(2N|E|^2)$ ，合并后表示为 $O(|E|(N \log N + 2N|E| + 1))$ 。在实际应用中，虽然时序图 G 中的两个实体之间会有多条具有相同标签不同发生时间的事件，但是在时序图关联规则中采用了时间窗的策略，即使时序图 G 很大，但是因为其时间周期性的缘故，经过时间窗划分后的时序子图中两个实体之间此类边的数量有限，即 N 很小且随着时间跨度增加变化不大。由此可知时序匹配算法具有多项式时间的复杂度，主要与图查询 Q 的具有次序的边的数量有关。

时序图匹配算法的时间复杂度。采用子图同构语义的时序图匹配，尽管增加了时序剪枝策略，并且基于该策略的次序匹配算法具有较低的多项式时间复杂度，但是时序图匹配问题依然属于 NPC 问题，其匹配过程仍旧是通过搜索候选点对和剪枝过滤来完成，所以时序图匹配算法具有指数的时间复杂度。

4.3 并行化方案

本节主要讨论基于时序图关联规则的事件预测的并行化方案。首先讨论并行方案中所需的图数据组织形式，在分析了以中心点跳数划分时序图的不足之处后，给出根据时间窗口划分时序图的方法；然后给出基于时序图关联规则的事件预测并行算法；最后对算法的并行可扩展性进行了理论分析。

4.3.1 时序图划分方法

以中心点跳数划分图。图模式关联规则中对图的划分通过跳数计算来实现^[8]，这是基于现实中 98% 的图模式的半径为 1 和 1.8% 的图模式半径为 2^[68]，以及社交网络图的平均点度数为 14.3^[69]。已知给定的关联规则集合 Σ ，集合中的关联规则的任意点均在中心点 x 的 d 跳以内。对图 G 按照中心点 x 进行划分，将与 x 标签相同的点平均分到不同的子图中，确保每个子图中的中心点 x 的 d 跳以内的点和边都划分在一起。使用划分后的子图进行并行计算则不会遗漏任何可能满足的匹配。

例 17：给定 $d=2$ 的图划分。使用中心点 d 跳以内的划分方式对图进行划分的

示意图如图 18 所示，划分后的各个子图中会有多个与中心点 x 标签相同的点，该子图中包含的其它点都是该点 2 跳以内可以到达的点。这样只需要保证给定的关联规则的 x 所能到达的最大跳数也是 2，即可保证匹配结果不会跨图分区，从而只需要在每个图分区上做图匹配即可，避免了复杂的跨分区的图匹配问题。

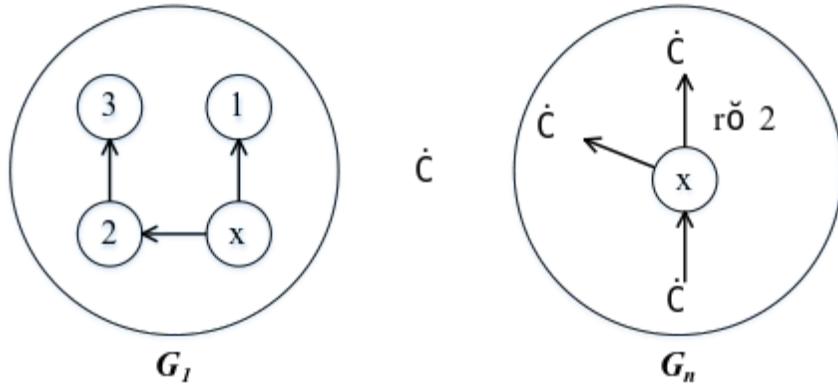


图 18 以中心点跳数对图划分示意图

但是许多大型网络的一个共同特点是点的连通度服从无标度的幂律分布^[70]。对于点的度数倾斜较为严重的幂律图和平均半径较大的图而言，一个子图划分可能会包含接近于整个图，该划分方法存在局限。

以时间窗口划分时序图。时序图所刻画的事件网络更倾向于此种情形，具有点的度数不均匀、点少边多、半径比一般图大的特点。因此不能很好地适用以跳数计算的图划分方法。考虑时序图具有天然的时间周期性，根据时序图关联规则，可以通过时间窗口和滑动步长对时序图进行划分。给出划分方法如下表所示。

表 6 时间窗划分时序子图算法

算法：Partition

输入：时序图 G ，时间窗 τ 和滑动步长 s

输出：划分后的时序图 $GSet$

1. $GSet := \emptyset$;
2. 对边集合按照时间戳升序排序， $E_s := \langle e_1, t_1 \rangle \langle e_2, t_2 \rangle, \dots, \langle e_n, t_n \rangle$;
3. **while** $i := 0, 1, 2, \dots$ **do**
4. $t_s := t_1 + s * i$;
5. **if** $t_s > t_n$ **then break;**
6. $t_e := t_s + \tau$;
7. $G := \emptyset$;

```

8.   for  $\langle e_m, t_m \rangle \in E_s$  do
9.     if  $t_s \leq t_m < t_e$  then  $G.Add(\langle e_m, t_m \rangle)$  ;
10.     $GSet := GSet \cup G$  ;
11.  return  $GSet$  ;

```

首先初始化子图集合，将时序图的边按时间戳升序排序（算法 1-2 行）。设置时间窗的左端 t_s 为时序图中时间戳最小的边的时间戳 t_1 加上当前轮数乘以步长，时间窗的右端 t_e 为左端 t_s 加上时间窗大小 τ （算法 4-6 行）。一个时间窗大小覆盖的边集合和对应点集合为一个时序子图，遍历时序图，将满足条件的边和对应点加入到当前子图中，得到一个子图划分（算法 7-10 行）。随后按照滑动步长对时间窗进行移动，计算下一个时序子图，直到完成对整个时序图的划分。如果时间窗左端 t_s 大于时序图中时间戳最大的边的时间戳 t_n ，则表明所有边已经按照时序划分完成，算法结束（算法第 5 行）。

例 18：给定时间窗个数 n 对时序图进行划分。输入时间窗大小和滑动步长容易计算出划分的时间窗个数。图 19 是以时间窗口对时序图进行划分的示意图，其中 t 是一个标志，表示该条边的时间戳在该时序子图的时间窗范围内，随后的 x 、 y 、 z 代表在该时间窗内的具体时间值。

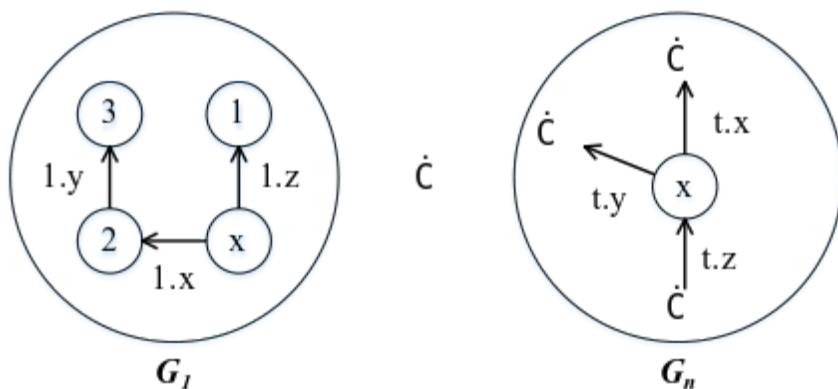


图 19 以时间窗口对时序图划分示意图

使用时间窗的时序图划分方法能够避免出现图数据组织不均的情况。对于发展平稳的事件网络，即网络中的事件的数量不存在随着时间发展而暴增或锐减，即使网络服从无标度的幂律分布或者图的半径较大，但图中的边会根据发生时间被平均分布到每个时间周期内，从而时序子图的划分是较为均匀的。

4.3.2 事件预测的并行算法

根据时序图应用问题和时序图的划分方式，给出一种基于时序图关联规则的事件预测的并行算法 TGREP (a parallel algorithm for Event Prediction based on Temporal Graph association Rules)。算法的主要思想是采用时间窗口对时序图进行划分，通过多个计算节点担任 worker 在划分后的每个时序子图上计算匹配，选择一个节点担任 coordinator 收集部分结果进行汇总。伪代码描述如下表所示。

表 7 基于时序图关联规则的事件预测并行算法

算法：TGREP
输入： 时序图 G ，预测事件 $q(x, y)$ 关联规则集 Σ ，时间窗 τ 和滑动步长 s
输出： 匹配结果 $\Sigma(x, G)$

```

/*Coordinator*/
step1: Partition( $G, \tau, s$ ) ; 将切分后的时序子图平均划分为多个分区  $F_i$ 
step2: 接收 worker 发送的部分结果，计算匹配的实体集  $\Sigma(x, G)$ 
return  $\Sigma(x, G)$  ;

/*Worker*/
step1: workeri 在分区  $F_i$  上对  $\Sigma$  中的每条规则计算匹配到的实体  $x$ 
step2: 将部分计算结果发送给 coordinator

首先，coordinator 对时序图进行划分得到时序子图，将时序子图随机打乱后平均分配给各个 worker。然后，worker 在自己的分区  $F_i$  上进行计算，一个分区可能包含多个时序子图。对于关联规则集合  $\Sigma$  中的每一条规则  $R$ ，计算其在时序图上对应的匹配实体，得到部分匹配结果，发送给 coordinator，为了最大化计算资源，可以选取 0 号 worker 作为 coordinator。最后，coordinator 将每条规则的匹配实体进行汇总（相同的匹配实体在整个时序图上只计算一次匹配），得到该规则在整个时序图上的匹配结果，将匹配结果放入  $\Sigma(x, G)$ 。对所有关联规则处理结束后，返回所有的匹配结果  $\Sigma(x, G)$ 。

```

4.3.3 复杂度分析

时间复杂度。 算法的过程包括对时序图的划分、每个 worker 计算自己分区的匹配

结果和将部分结果汇总由 coordinator 计算最终结果。首先，coordinator 对时序图进行切分，只需扫描一遍图 G ，根据时间戳的范围和时间窗大小和步长即可完成分图，所需要的时间复杂度为 $O(|G|)$ ，是一个线性时间。然后，每个 worker 计算关联规则在自己分区的匹配情况，假设中心点集合为 X ，在一个时序子图上处理一个中心点匹配所需的最坏时间为 $t(|G/N|, |\Sigma|)$ ，其中 N 是分图数量， Σ 是关联规则集合。

因为有 n 个 worker 同时处理，所以所需时间复杂度为 $O(Nt(|G/N|, |\Sigma|)|X|/n)$ 。

worker 将匹配结果发送给 coordinator 进行汇总，需要花费时间为 $O(|X||\Sigma|/n)$ 。因为中心点的规模不大于图的点规模，所以有 $|X| \leq |V| < |G|$ ，关联规则集合的规模 $|\Sigma|$ 很小，因此 $O(|X||\Sigma|/n)$ 的时间可以忽略不计。同样时序图划分的时间 $O(|G|)$ 也可以忽略不计。

因此应用算法的主要时间复杂度为 $O(Nt(|G/N|, |\Sigma|)|X|/n)$ 。从子图同构问题的复杂度可知函数 $t()$ 是指数级别，除非 $N = NP$ 被证明，并且该指数函数消耗的时间随着 $|G/N|$ 规模变化，且 $|G|$ 很大， $|G/N| \ll |G|$ ，所以 N 和 $|X|$ 对函数 $t()$ 的影响非常小，从而有 $Nt(|G/N|, |\Sigma|)|X| \ll t(|G|, |\Sigma|)$ 。所以该并行算法的时间复杂度上界为 $O(t(|G|, |\Sigma|)/n)$ ，随着计算节点的数量增加，运行时间降低，具有良好的并行可扩展性。

并行通信量。并行计算过程中存在通信时间开销，但该通信时间开销相比每个分区的匹配计算的时间开销而言非常小，这从通信量的规模上能够衡量。

因为采用了中心点的支持度定义，所以一个关联规则在一个时序子图上能够满足的最大匹配的中心点数量为 $|X|$ ，则整个关联规则集合在全部时序子图上的产生的

最大消息数量为 $N|\Sigma||X|$ 。因为时序子图数量和 worker 数量不一定相等，它们的关系满足 $n \leq N$ ，如果有些时序子图属于同一个 worker，那么这部分匹配的中心点可以在该 worker 汇总后再发给 coordinator，因此 n 个 worker 向 coordinator 发送消息的最大数量为 $n|\Sigma||X| + n|M|$ ，其中 M 是额外的信息，用于标识消息所属 worker 和中心点数量等信息，是个固定常量。 $n|M|$ 对于通信消息量影响非常小，主要通信量来源于 $n|\Sigma||X|$ ，这是一个与 worker 数量相关的线性函数，可见随着计算分区数的增加，并行通信的消息量线性增加。不过中心点 $|X|$ 的规模有限，因此该通信量消耗的时间在整个应用算法中的占比很小。

4.4 本章小结

本章作为论文主体工作的一部分，重点研究了时序图关联规则的应用问题。首先给出了基于时序图关联规则的事件预测问题的形式化表达；然后给出了在子图同构的匹配过程中增加时序剪枝条件的时序图匹配算法，以满足在时序图上的匹配需要；接下来结合时序图的时间周期特性，给出了按时间窗划分时序图的方法，并给出了基于时序图关联规则的事件预测的并行算法，并理论证明了该算法具有良好的并行可扩展性。

第五章 实验与分析

本章将对时序图关联规则的相关算法进行一系列实验，并对实验结果进行分析。首先，介绍了数据集和实验的相关设置。然后，进行了时序图关联规则发现实验和分析，主要从规则和发现算法的准确性以及发现算法的可扩展性等方面进行，并列举了部分发现结果。接下来，进行了时序图关联规则的应用实验和分析，主要从时序图匹配效率和基于时序图关联规则的事件预测的可扩展性等方面进行。

5.1 数据集准备

实验中使用 2 个来自真实世界中的图数据和 1 个合成图数据，如表 8 所示。表中描述了数据的基本信息，包括场景、点的数量 $|V|$ 、边的数量 $|E|$ 以及所跨时间长度 T 等。

表 8 实验选取数据集

数据集	场景	$ V $	$ E $	T
LianTong ^[46]	用户通话网络	248K	1.5M	45 days
JingDong ^[47]	用户购买商品网络	198K	8M	365 days
Synthetic	合成网络	1M-5M	2M-10M	45 days

LianTong，是一个联通用户的真实通话网络数据集，包含 45 个连续自然日期间，抽样的 4999 个用户每天的通话、短信、访问网站/应用记录的脱敏数据，有 248K 实体和 6.2M 边。实体中包含用户、手机号码、App 名称和风险用户等类型，事件中包含打电话行为、短信行为、访问 App 行为和属于某类用户等类型。通话、短信和访问等行为具有时间标签，例如用户 user 在 t 时刻给电话号码 phone 拨打电话的事件可以描述为一条边 $e=(user, phone, t)$ ， $L(e)=call$ ；用户 user 是一个风险用户 fake 可以描述为一条静态边，该边的时间戳为零，即 $e=(user, is, 0)$ ， $L(e)=fake$ 。

JingDong，是一个用户的真实购买行为数据集，包含 365 个连续自然日期间，抽样的 98924 个用户对一组具有关联性的商品的操作行为，数据已经过脱敏处理。有 198K 实体，8M 边，包括浏览、关注、购买和评价等带有时间戳的行为事件。例如用户 user 在 t_1 时刻浏览了某件商品 sku，并在 t_2 时刻对其下单购买，这两个行为可以分别描

述为 $e_1 = (user, sku, t_1)$, $L(e_1) = browse$ 和 $e_2 = (user, sku, t_2)$, $L(e_2) = buy$ 。

Synthetic, 是一个合成图数据, 以联通的通话网络数据为基础, 通过扩充构造的一系列合成图数据, 扩充后的数据中相同类型的图模式随着数据规模成比例增加, 用于检测算法的可扩展性。合成数据集的规模 (点数量, 边数量) 分别为(1M, 2M), (2M, 4M), (3M, 6M), (4M, 8M)和(5M, 10M)。

基于以上数据描述, 将采用这 3 组数据集进行相关实验研究。

5.2 实验设置

实验设置包括算法设置和实验环境设置两个方面: 在算法设置中描述了实验中采用的常规参数和算法实现方式; 在实验环境设置中描述了运行实验的机器的相关配置。

5.2.1 算法设置

在接下来的实验中, 如果没有对参数进行额外的说明, 则一律采用如下描述的参数设置: 发现算法的 top-k 队列的大小为 30, 扩展边数为 5; 对于采用时间窗的算法, 设置时间窗参数为 $\tau = s$, 其中, 对 LianTong 数据按照自然日划分为 45 个时序子图, 对 JingDong 数据按照自然月划分为 12 个时序子图; 对于采用动态支持度阈值的剪枝策略的发现算法, 设置参数为 $C=2$, $P=0.1$; 关联规则集 Σ 中的规则均是在相应数据集上发现的有意义规则, 规则大小均为 $|R|=(5,8)$, 表示该规则由 5 个点和 8 条边组成。

为了更好地验证采用优化策略和时间窗对于时序图关联规则的发现算法的影响, 实验中将采用多组不同的算法实现, 分别命名为 $TGRD_n$ 、 $TGRD_f$ 、 $TGRD_w$ 和 $TGRD_a$, 各个算法的具体策略设置如下表所示。

表 9 发现算法设置表

发现算法名称	具体含义
$TGRD_n$	使用多线程加速;
$TGRD_f$	使用多线程加速、扩边引导和匹配时间阈值的优化策略;
$TGRD_w$	使用多线程加速、扩边引导和匹配时间阈值的优化策略; 使用合适时间窗;

$TGRD_a$	使用多线程加速、扩边引导和匹配时间阈值的优化策略； 使用合适时间窗； 使用动态支持度剪枝的优化策略；
----------	--

为了更好地验证时序图匹配算法的效率和基于时序图关联规则的事件预测算法的并行化效率，实验中将采用多组算法实现，分别命名为 $TGREP_n$ 、 $TGREP_s$ 、 $TGREP$ ，各个算法的具体策略设置如下表所示。

表 10 事件预测算法设置表

事件预测算法名称	具体含义
$TGREP_n$	基于图关联规则的事件预测串行算法，模式匹配采用子图同构算法
$TGREP_s$	基于时序图关联规则的事件预测串行算法，模式匹配采用 TGPM 算法
$TGREP$	基于时序图关联规则的事件预测并行算法，模式匹配采用 TGPM 算法

5.2.2 实验环境设置

实验所使用的 CPU 计算集群由北京市大数据与脑机智能高精尖中心提供，单个计算节点的配置为：sockets = 2，cores=12，threads=2，CPU 频率 2.80GHz，内存大小 256GB。时序图关联规则的发现算法采用多线程加速的单机并行方案，实验中使用单个计算节点，最多运行 48 个线程。基于时序图关联规则的事件预测实验采用分布式扩展的多机并行方案，实验中最多使用 8 个计算节点，每个节点运行单个进程，内含单个线程。

5.3 时序图关联规则发现实验及分析

本节将从算法的准确性和可扩展性两个方面进行实验和分析。准确性用于验证时序图关联规则和发现算法选出规则的准确程度。可扩展性用于验证发现算法在不同参数条件下的效率是否符合预期。

5.3.1. 时序图关联规则和发现算法的准确性

时序图关联规则旨在更好地解决时序图上的事件预测问题，因此发现的关联规则的准确性十分重要。准确性实验将从两方面开展：一方面是对比时序图关联规则和发现算法与其他规则和算法之间的准确性；另一方面是通过实验研究发现算法的准确性

和效率之间的关系，看是否存在一个平衡点，能够在保证准确率的情况下最大化发现效率。

准确性对比实验。这里选择的对比对象为 GPAR（基于图匹配的未考虑时间属性的关联规则），时序图关联规则的发现算法选用 $TGRD_w$ 进行实验，分别对比发现 top-10、top-30 和 top-60 关联规则的准确率。

对于实验数据的划分，在进行 GPAR 实验时，将数据集以原有的自然天数为单位进行随机划分，保证训练集 F_1 的数据量为 80%，测试集 F_2 的数据量为 20%。在进行时序图关联规则的实验时，按照符合该数据类型的时间窗口 τ 和滑动步长 s 对时序图 G 进行切分，将 80% 的时序子图作为训练集 F_1 ，将剩余 20% 的时序子图作为测试集 F_2 。

将关联规则在测试集上的准确率定义为关联规则 R 的中心点匹配数与查询 Q 的中心点匹配数的比值，形式化表示如下：

$$Prec(R) = \frac{supp(R, F_2)}{supp(Q, F_2)} \quad (5.1)$$

分别在真实世界的数据集 LianTong 和 JingDong 上进行实验。对于 GPAR，设置从中心点开始的最大扩展跳数为 2。对于 $TGRD_w$ ，设置最大扩展边数 $d=5$ 。设置 LianTong 数据上的剪枝阈值 $\sigma=600$ ，JingDong 数据上的剪枝阈值 $\sigma=2k$ 。每组实验通过重新划分训练集和数据集，重复三次，对实验结果取平均值，得到如下表所示实验结果。

表 11 时序图关联规则和发现算法准确性实验结果

算法设置	LianTong			JingDong		
	Top-10	Top-30	Top-60	Top-10	Top-30	Top-60
GPAR	0.865	0.854	0.833	0.512	0.469	0.432
$TGRD_w$	0.898	0.891	0.872	0.532	0.495	0.451

从表中数据可以看出，时序图关联规则发现算法 $TGRD_w$ 挖掘出的时序图关联规则在测试集上的准确率要高于没有考虑时序关系的图关联规则 GPAR，由此可见时序图关联规则确实可以提高关联规则的准确性。还可以看出随着规则数量增多，预测精度呈下降趋势，说明高质量的规则有限。在 LianTong 数据上进行的是风险用户的预测，

使用发现的规则进行预测的准确率整体较高，说明风险用户具有一定比较明显的行为模式，而在 JingDong 数据上对用户购买商品的预测整体准确率相比之下要低得多。

发现算法效率和准确率的平衡。因为发现过程中会生成大量的候选关联规则，因此通过支持度阈值进行剪枝是非常必要的，剪枝效果越好，发现的效率越高，但因为支持度较小的关联规则也有可能有较大置信度，所以阈值设置的大小对于最终结果准确性的影响是一个值得关注的问题。

将使用 $TGRD_a$ 算法来研究发现效率和准确性的平衡问题。因为 $TGRD_a$ 的剪枝策略是动态剪枝，在给定初始阈值之后，每轮新增的候选关联规则的数量与设置的参数 C 和 P 有关，可以控制每轮进入下一轮迭代的候选规则数量，从而起到类似于每轮更新剪枝阈值的效果，从而大幅度提高发现算法的效率。可以通过设置初始阈值 σ 和参数 C 与 P 来探究发现算法效率和准确性之间的关系。

在 LianTong 和 JingDong 数据集上分别设置初始阈值 $\sigma=600$ 和 $\sigma=2k$ ，使用线程数 $n=24$ ，设置 $C=0$ ， P 的取值范围为 0.1 到 0.5，最终发现个数 $K=30$ ，选择 80% 的数据作为训练集，20% 的数据作为测试集。分别进行实验得到如下结果。

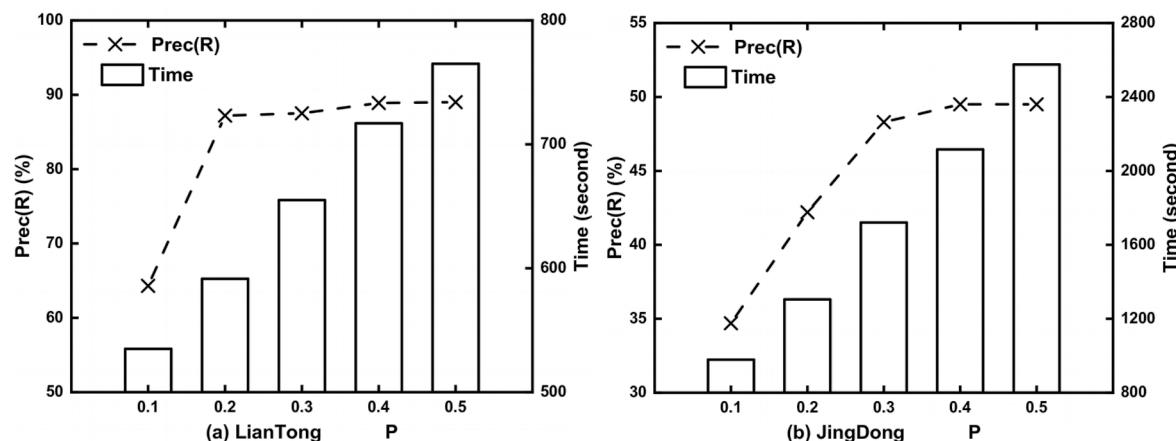


图 20 发现算法效率和准确率实验结果

从图中可以看出，在一定范围内，增加候选关联规则的数量，可以提高最后选出的关联规则在测试集上的准确率。但是过多的候选关联规则并不会对最终的准确率产生较大影响。在图(a)的 LianTong 数据中，较好的平衡点是 $P=0.2$ ，此时准确率为 87.2%，挖掘时间为 591.5s，意味着每轮选择前 20% 的候选关联规则进行扩展，即可以得到较高的精确度和较快的发现速度；在图(b)的 JingDong 数据中，较好的平衡点是

$P=0.3$ ，此时准确率为 48.3%，挖掘时间为 1711.6s，意味着每轮选择前 30% 的候选关联规则进行扩展，即可以得到较高准确度和较快发现速度。所以只需要选择一定数量的较高支持度的关联规则进行下一轮的扩展，就可以达到可接受的准确率，并且有一个较快的发现过程，这是发现算法在效率和准确率之间的一个平衡。

5.3.2 发现算法的可扩展性

算法的可扩展性非常重要，将决定算法在不同的参数和数据规模下的效率是否符合预期。发现算法可扩展性的相关实验如下表所示。

表 12 发现算法可扩展性实验项目表

自变量	取值
线程数量 n	8, 16, 24, 32, 40, 48
剪枝阈值 σ	实验中根据不同数据集选取不同值
图的规模 $ G $	合成图，从(1M, 2M)开始增加到(5M, 10M)

通过改变所用线程的数量 n 、剪枝阈值 σ 和图的规模 $|G|$ ，观察花费时间变化，来验证发现算法在改变参数和数据规模情况下的可扩展性。接下来将依次进行相关实验并对实验结果进行分析。

线程数量对发现效率的影响。通过改变使用的线程数量，实验分析发现算法的效率。实验过程中设置挖掘边数 $d=5$ ，对于 LianTong 数据设置剪枝阈值 $\sigma=600$ ，对于 JingDong 数据设置剪枝阈值 $\sigma=2k$ ，改变线程数 n 的大小，变化范围为 8 到 48，使用发现算法 $TGRD_f$ 、 $TGRD_w$ 和 $TGRD_a$ 分别进行实验，实验结果如下图所示。

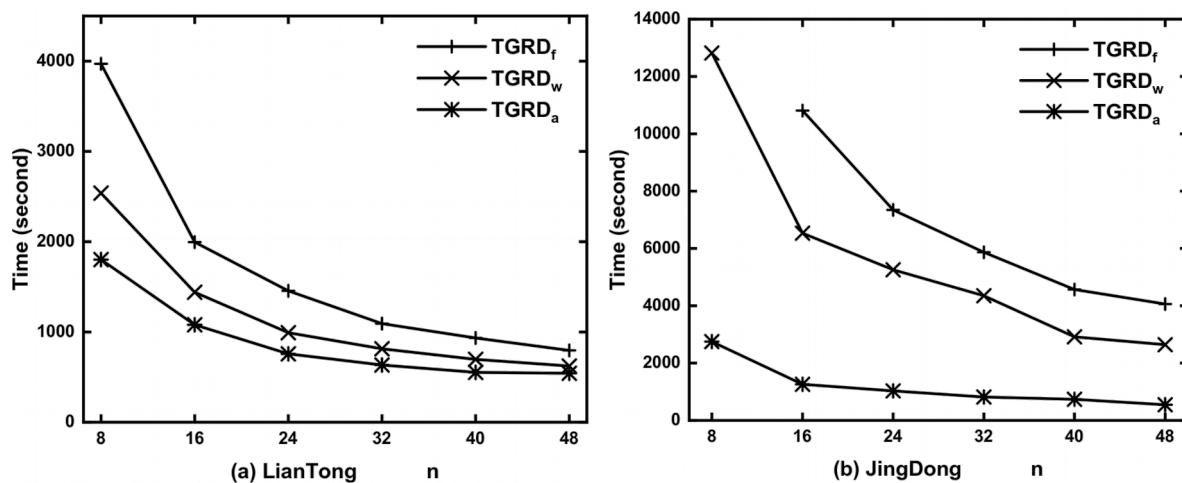


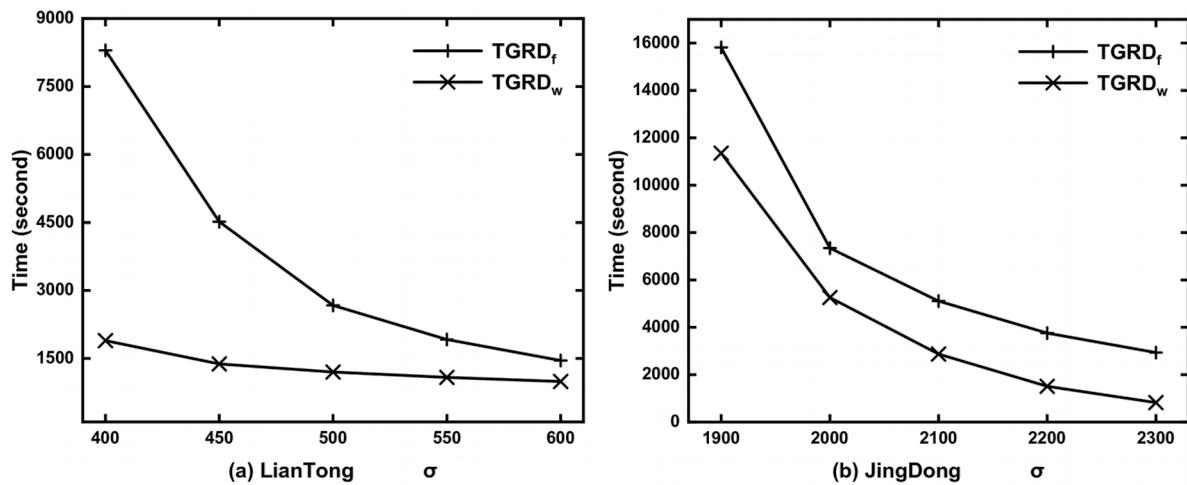
图 21 线程数 n 对发现算法效率影响实验结果

从图中可以看出，采用不同优化策略的发现算法的消耗时间均随着线程数增大而降低，说明它们都具有良好的并行可扩展性。在没有使用动态支持度阈值的发现算法中，在 LianTong 数据集中一共扩展了多达 1800 个时序图关联规则，在 JingDong 数据集中一共扩展了多达 3000 个时序图关联规则。图中未绘制 $TGRD_n$ ，因为不采用任何优化策略的发现算法耗时过久，但其消耗时间同样满足随着线程数增加而降低的趋势。未绘制 $TGRD_f$ 在线程数 8 时的数据点，此处耗时超过 2 万秒。

在图(a)中，线程数 n 从 8 变化到 48 之后， $TGRD_f$ 的后者与前者的运行速度比是 4.99 倍， $TGRD_w$ 的后者与前者的运行速度比是 4.07 倍， $TGRD_a$ 的后者与前者的运行速度比是 3.31 倍。在图(b)中，线程数 n 从 8 变化到 48 之后， $TGRD_f$ 的后者与前者的运行速度比是 4.95 倍， $TGRD_w$ 的后者与前者的运行速度比是 4.84 倍， $TGRD_a$ 的后者与前者的运行速度比是 5.02 倍。

发现算法 $TGRD_w$ 相比 $TGRD_f$ 具有更快的发现速度，这是因为时间窗的设置相当于对时序图进行了划分，加快了时序图模式在每一个时序子图上的匹配速度。发现算法 $TGRD_a$ 具有最快的速度，这是因为它采用了动态支持度阈值进行剪枝，可以通过支持度过滤大量的候选关联规则，从而加速每一轮的发现速度。

剪枝阈值对发现效率的影响。通过改变剪枝阈值 σ 的大小，研究发现算法效率的变化。设置扩展边数 $d=5$ ，使用线程数 $n=24$ ，在 LianTong 数据集上设置剪枝阈值 σ 的变化范围为 400 到 600，变化幅度为 50，在 JingDong 数据集上设置剪枝阈值 σ 的变化范围为 1.9k 到 2.3k，变化幅度为 100。使用发现算法 $TGRD_n$ 、 $TGRD_f$ 和 $TGRD_w$ 分别进行实验，实验结果如下图所示。

图 22 剪枝阈值 σ 对发现效率影响实验结果

从图中可以看出，所有算法均满足剪枝阈值 σ 越小运行时间越久的特点，这是因为剪枝阈值越小，就会有越多的候选关联规则需要进行验证，从而花费更多的时间。图中未绘制 $TGRD_n$ ，因为其消耗时间与其它算法相比过长，但同样满足上述特点。

可以看出增加了时间窗的发现算法 $TGRD_w$ 相比 $TGRD_f$ 在相同剪枝阈值下具有更快的速度，这是因为通过时间窗对图进行划分，相当于降低了每次匹配所在图的规模，加速了时序图匹配的过程。

图规模对发现效率的影响。通过合成图来研究图的规模大小对发现效率的影响，设置扩展边数 $d=5$ ，使用线程数 $n=24$ ，剪枝阈值 $\sigma=600$ ，合成图的规模从小到大依次选取(1M, 2M), (2M, 4M), (3M, 6M), (4M, 8M)和(5M, 10M)，使用发现算法 $TGRD_n$ 、 $TGRD_f$ 、 $TGRD_w$ 和 $TGRD_a$ 分别进行实验，实验结果如下图所示。

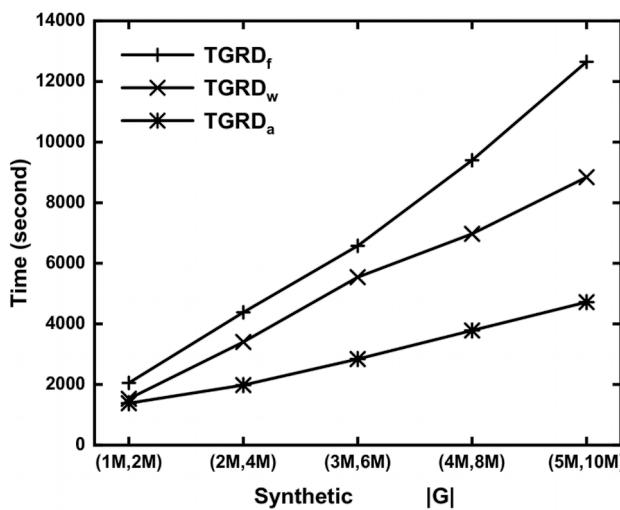


图 23 图规模 $|G|$ 对发现效率影响实验结果

从图中可以看出，随着图规模增加，所有策略下的发现算法的消耗时间都在增加，而且增加的趋势较为明显，这是由于子图同构问题的复杂性导致的。图中未绘制 $TGRD_n$ ，因为其消耗时间与其它算法相比过长。可以看出采用了时间窗的 $TGRD_w$ 在不同图规模下均比 $TGRD_f$ 具有更高的效率，采用动态支持度阈值的 $TGRD_a$ 在不同图规模下均具有最快发现速度，这说明了优化策略的有效性。

5.3.3 发现结果示例

在真实的数据集 LianTong 和 JingDong 中进行时序图关联规则发现，得到如下的高置信度规则，该规则具有较好的可解释性。

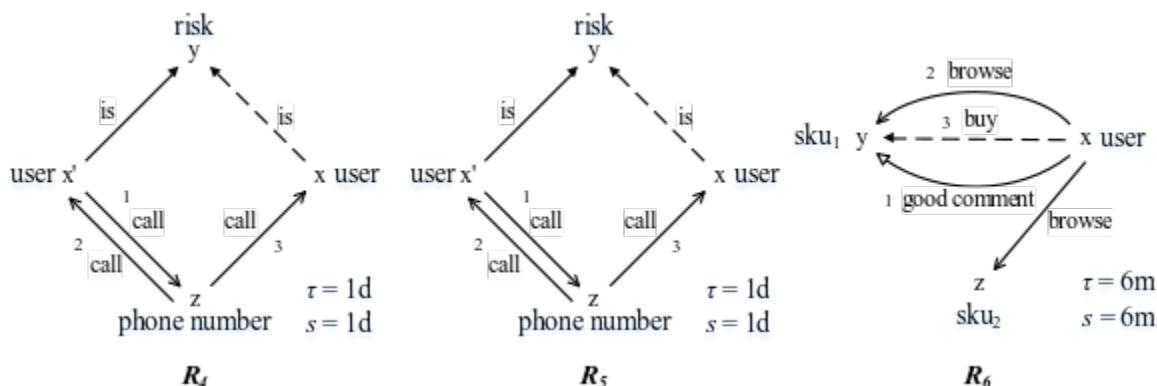


图 24 真实时序图关联规则示例

如图所示，关联规则 R_4 和 R_5 的时间窗口大小为 1 天， R_4 描述了当一个风险用户给一个电话号码打电话之后，该电话号码进行了回拨，随后又在时间窗口限制的时间内给另一个号码打了电话。该情况下，另一个号码极有可能也是风险用户。关联规则 R_5 描述了风险用户给正常用户发短信，人们回拨之后又去给另一个用户打电话的场景，这也符合一般的电信诈骗套路。关联规则 R_6 的时间窗口为 6 个月，描述了当一个用户给过一个商品好评之后，再次浏览该商品，并且有浏览过同类型其它商品，则该用户随后继续下单购买该商品的概率较高。

5.4 时序图关联规则应用实验及分析

本节将从时序图匹配算法的效率和事件预测算法的扩展性两个方面进行实验和分

析。由于在原有的子图同构基础上额外增加的时序匹配会增加匹配的时间消耗，所以需要通过实验证时序图匹配算法的效率。可扩展性实验可以验证基于时序图的事件预测算法在不同参数条件下的效率是否符合预期。

5.4.1 时序图匹配算法的效率

因为时序图模式增加了时序关系，相比于采用子图同构算法的图模式匹配而言，时序图模式匹配一定会消耗更多的时间。单次匹配的时间不易计算，因此通过在一组关联规则集 Σ 上的匹配时间来衡量它们之间的效率差别。在单机情况下，选择不考虑时序的事件预测串行算法 $TGREP_n$ 和考虑时序的事件预测串行算法 $TGREP_s$ 进行实验研究，规则集的规模 $|\Sigma|$ 的取值为 8 到 48，得到如下实验结果。

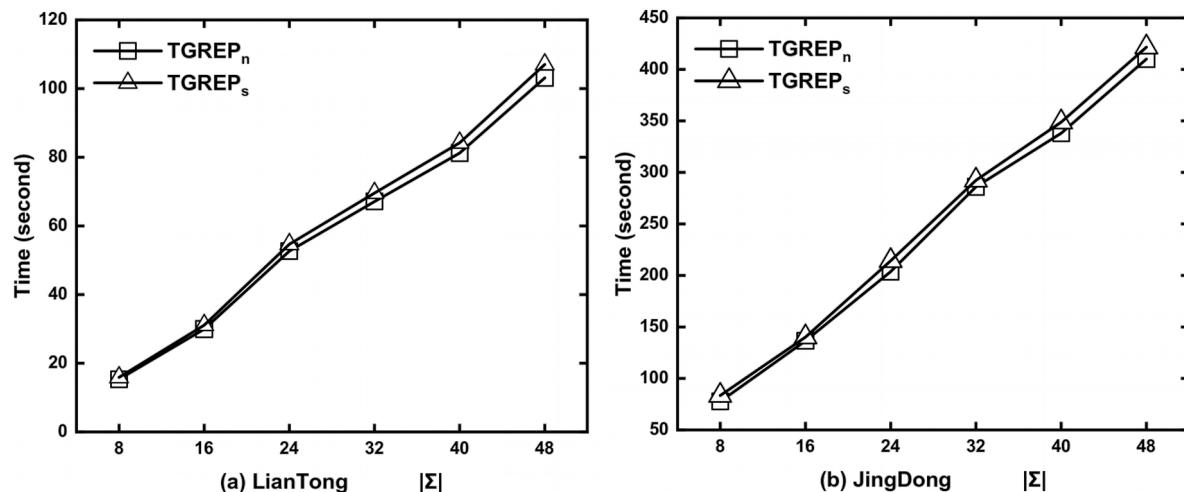


图 25 时序图匹配算法的效率对比实验结果

从图中可以看出，基于时序图匹配的事件预测算法比基于图匹配的事件预测算法要消耗更多的时间，这是合理且符合预期的。从所消耗的时间来看，增加了时序匹配之后所需的匹配时间是可以接受的。并且在实际的事件预测应用中，可以通过多机扩展的方式进行并行加速，因此时序图匹配算法的效率可以接受。

5.4.2 事件预测算法的可扩展性

事件预测算法可扩展性的相关实验如下表所示。

表 13 事件预测算法可扩展性实验项目表

横坐标参数	取值
计算节点数量 n	2, 4, 6, 8
规则集的规模 $ \Sigma $	8, 16, 24, 32, 40, 48
时序图的规模 $ G $	合成图, 从(1M, 2M)开始增加到(5M, 10M)

改变计算节点的数量 n 来验证算法的并行可扩展性, 改变关联规则集的规模 $|\Sigma|$ 和时序图的规模 $|G|$ 来验证算法在不同参数和数据规模情况下的可扩展性。

计算节点数量对事件预测效率的影响。通过改变使用的计算节点数量, 每个节点上运行一个 worker, 验证算法的并行可扩展性。设置时序图关联规则集的规模 $|\Sigma|=24$ 。改变计算节点的数量 n , 取值范围为 2 到 8, 得到如下实验结果。

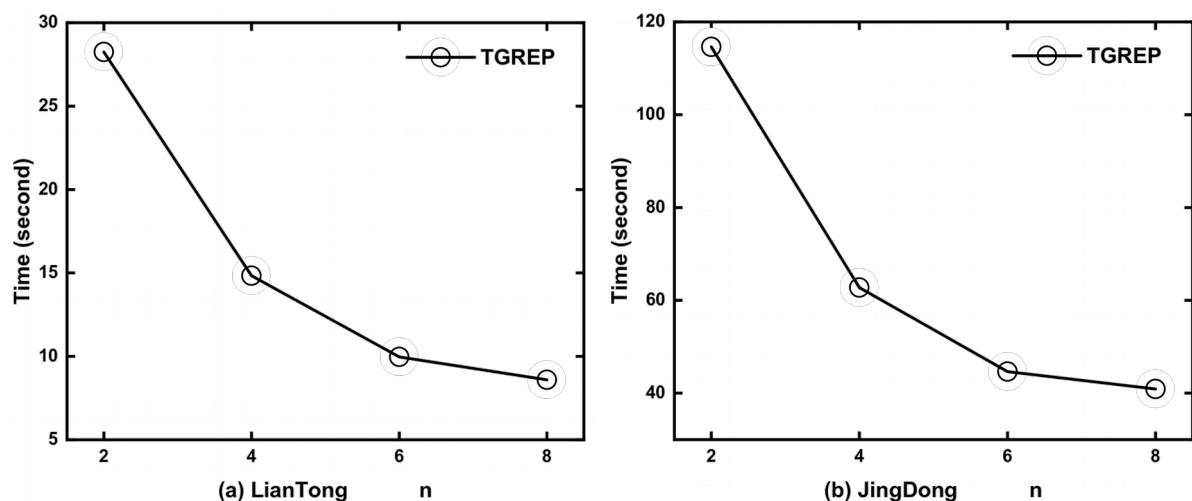


图 26 计算节点数 n 对事件预测效率影响实验结果

从图中可以看出, 随着节点数的增加, 基于时序图关联规则的事件预测算法具有较好的并行可扩展性。在图(a)的 LianTong 数据中, 当节点数从 2 增加到 8 时, 后者与前者的运行速度比是 3.28 倍, 在图(b)的 JingDong 数据中, 当节点数从 2 增加到 8 时, 后者与前者的运行速度比是 2.8 倍。

使用时间窗对时序图进行子图划分, 从而采用多计算节点的加速方案提高效率, 在一定程度上具有很好的效果, 但是仍然具有一定的局限性。按照时间窗分图, 有可能出现分图不均情况, 虽然可以通过负载均衡策略将每个 worker 所计算的分区由总规模上相近的一组时序子图组成, 但计算较慢的 worker 总会拖慢整个事件预测算法的效率, 这也是随着节点数增加, 影响加速比的重要原因。

规则集规模对事件预测效率的影响。通过改变规则集规模的大小, 研究对算法的效率影响。设置规则集中的关联规则大小为 $|R|=(5,8)$, 计算节点数 $n=8$, 规则集

的规模 $|\Sigma|$ 的变化范围为 8 到 48，实验结果如下图所示。

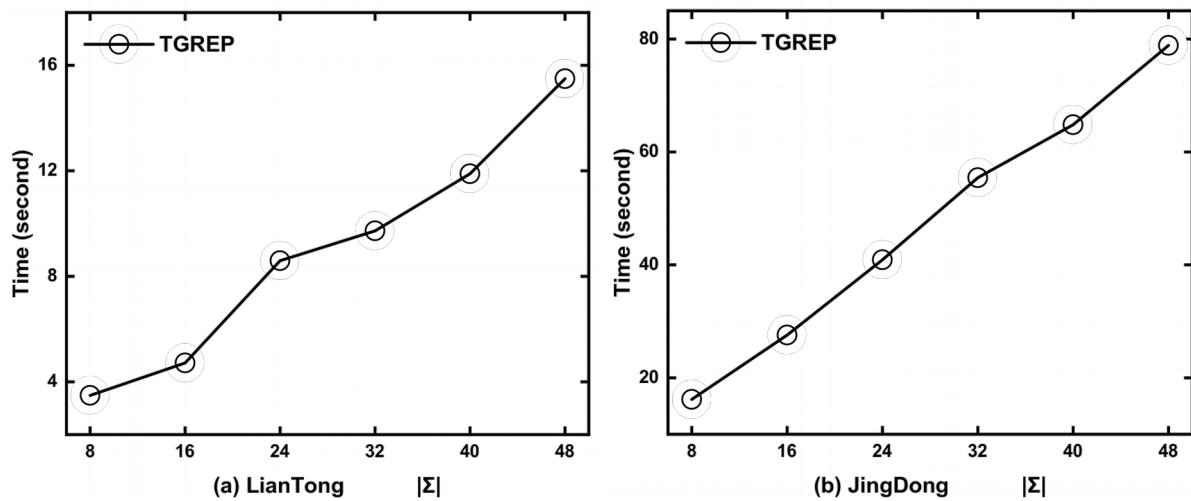


图 27 规则集规模 $|\Sigma|$ 对事件预测效率影响实验结果

从图中可以看出，TGREP 算法的消耗时间随着规则集规模增大而增加，这是符合预期的。当规则集规模从 8 增大到 48 时，在图(a)中后者与前者所消耗时间的比值为 4.44，在图(b)中后者与前者所消耗时间的比值为 4.88，可见消耗时间随着规则集规模增大呈线性增长的趋势。

时序图的规模对事件预测效率的影响。通过合成图来研究图的规模大小对算法效率的影响，设置时序图关联规则集的规模 $|\Sigma|=24$ ，其中包含规则的大小 $|R|=(5,8)$ 。首先选择图规模较大的合成图 $|G|=(5M,10M)$ 在不同节点数下进行了实验，验证在较大规模数据上算法的并行可扩展性。接着设置节点数 $n=8$ ，在不同规模的合成图上进行实验研究算法的效率，实验结果如下图所示。

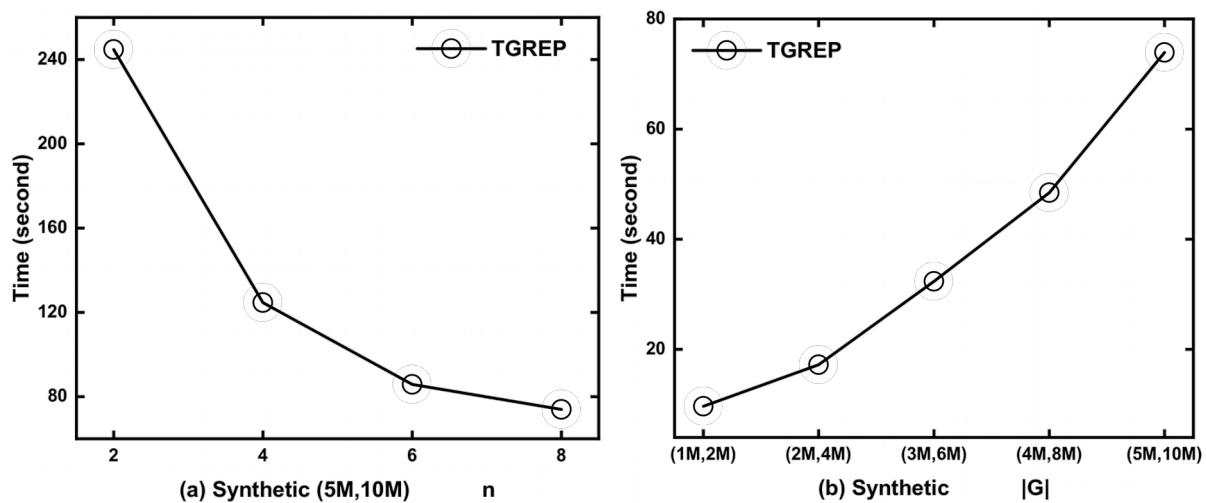


图 28 时序图规模 $|G|$ 对事件预测效率影响实验结果

从图(a)中可以看出，*TGREP* 算法在较大的时序图上依然具有很好的并行可扩展性，节点数从 2 增加到 8，后者与前者的速度比为 3.31。在图(b)中，随着合成图的规模增加，*TGREP* 算法的消耗时间也在增加，但增加趋势并不非常陡峭，这意味着可以通过增加并行节点的数量来进一步加速事件预测的过程。

5.5 本章小结

本章为实验研究和分析。首先介绍了实验所选用的数据集和相关实验设置，包括采取不同策略的发现算法和事件预测算法的设置以及实验环境的配置。接下来对于时序图关联规则和发现算法的准确性进行了对比实验，验证了相比其它规则，时序图关联规则在时序数据上具有更好的预测能力。然后通过改变线程数、剪枝阈值和时序图的规模验证了发现算法具有良好的可扩展性。接下来针对应用问题，通过实验验证了时序图匹配算法的效率完全可以接受，然后通过改变计算节点数量、关联规则集规模和时序图的规模验证了基于时序图关联规则的事件预测算法具有良好的可扩展性。

第六章 时序图关联规则计算平台的设计与实现

本章将介绍一个时序图关联规则发现和应用平台的设计与实现。平台整合了时序图关联规则发现算法和时序图匹配算法，并提供了可视化的 Web 操作界面以及图模式可视化展示。接下来将从系统架构、数据格式、业务层逻辑、用户交互层和界面展示等方面对系统进行介绍。

6.1 系统架构设计

因为系统所涉及的业务逻辑并不复杂，主要是算法实现，因此选用经典的三层架构即可满足需求，架构图如下所示。

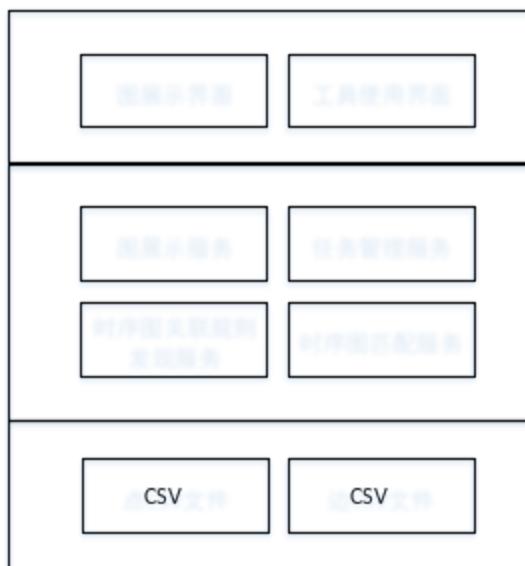


图 29 系统架构图

第一层为用户交互层，主要功能界面为图展示界面和工具使用界面；第二层为业务逻辑层，实现了用于满足交互层的请求调用服务，主要包括图展示服务、时序图关联规则发现服务、时序图匹配服务和任务管理服务；第三层为数据层，用于持久化表示。数据和服务调用在上下两层之间流动，不能跨层。

6.2 数据格式设计

系统内部交互所使用的数据格式为 CSV 格式，即逗号分隔值文件格式。选用该种

文件格式的原因主要有以下几个方面：一是该格式能够清晰准确且简洁地表示图数据，相比 XML、JSON 等数据格式而言，CSV 格式更加轻量；二是主流的图数据库和图计算工具都支持 CSV 格式或某种变体，考虑兼容性，采用该格式能够更好地实现与其它工具之间的数据互通。

对于图的点文件的 CSV 描述，点的 ID 类型采用 int64 类型（因为目前发现算法和应用算法中点的规模不会过于庞大，该数据规模足够表示），点的标签采用 int32 类型，额外的点属性信息可以通过增加属性名和对应类型的方式添加，具体格式见下表。

表 14 点文件 CSV 格式

点 ID	点标签	点属性
vertex_id:int64	label_id:int32	attribute:type

对于图的边文件的 CSV 描述，边 ID 用于边的索引，类型为 int64，边上的起始点和终点的 ID 采用 int64 类型，与点保持一致，边上的标签采用 int32 类型，额外的边属性信息可以通过增加属性名和对应类型的方式添加，具体格式见下表。

表 15 边文件 CSV 格式

边 ID	起始点 ID	终点 ID	边标签	边属性
edge_id:int64	source_id:int64	target_id:int64	label_id:int32	attribute:type

6.3 业务层逻辑设计

业务层主要包括图展示服务、时序图关联规则发现服务、时序图匹配服务和任务管理服务等功能模块。本节将依次介绍各个业务功能的设计和现实方式，因为算法设计在前面章节已经充分描述，因此本节将主要着重于工程实践方面的设计。

图展示服务。该服务主要提供在线的图展示功能，用于将时序图关联规则和匹配结果进行可视化展示。前端系统处理用户的操作并通过 NodeJS 对图数据进行渲染，绘制可视化的图。数据在 Web 前后端通过 JSON 形式传输，用户保存的图模式会被储存为 CSV 格式，以兼容平台内各类算法的计算。

时序图关联规则发现和时序图匹配服务。该部分集成到系统的任务服务中，用户通过工具界面使用相应工具即可调用对应服务。具体的算法设计在前面的章节部分已经详细描述，此处重点介绍工程上的设置。为了方便用户对于服务的配置和调用，系

统后台使用 YAML 文件对算法所需的参数进行描述。前端界面根据各类算法 YAML 描述文件的不同，自动展示不同种类的可视化参数输入界面，供用户选择。

时序图关联规则的发现服务的 YAML 配置文件的主要内容如下表所示。

表 16 时序图关联规则发现服务的配置文件字段表

一级字段	二级字段	类型	详情描述
Graph	Dir	字符串	图的位置
	Name	字符串	图的名称
	HasAttributes	布尔型	表示图是否具有属性
Discovery	K	整型	表示发现算法中的 top-k 队列大小
	SuppThreshold	整型	表示发现算法中的 support 剪枝阈值
	D	整型	表示发现算法中扩展的最大边数
	F	浮点型	表示发现算法中设置的边频率
	N	整型	表示发现算法中使用的线程数
OutPutGraph	Dir	字符串	发现的 k 条规则的存储位置
	Name	字符串	规则的存储名称

时序图匹配服务的 YAML 配置文件的主要内容如下表所示。

表 17 时序图匹配服务的配置文件字段表

一级字段	二级字段	类型	详情描述
Graph	Dir	字符串	图的位置
	Name	字符串	图的名称
	HasAttributes	布尔型	表示图是否具有属性
TGR	Dir	字符串	表示时序图关联规则的文件所在位置
	Name	字符串	表示时序图关联规则的文件名称
OutPutGraph	Dir	字符串	匹配结果的存储位置
	Name	字符串	匹配结果存储文件的名称

用户在设置好相关输入参数后，通过操作界面即可直接运行一个所需服务的任务，并通过任务管理服务查看任务的调度情况和最终结果。

任务管理服务。该服务用于任务的管理，通过轮询任务队列的方式实现。用户选择需要使用的服务，并填写所需的配置信息后，开始运行服务，就会触发任务管理服务新建一个对应服务的任务。新建的任务会进入到任务队列，任务管理服务根据后台机器资源的具体情况，为算法任务分发计算节点，从而实现负载均衡。

任务管理服务会去轮询执行任务队列，对于超时或执行失败的任务进行信息更新。对于执行成功的任务，会在任务管理界面显示相应的任务完成信息，包括任务执行的

开始时间、结束时间和结果详情等。

6.4 用户交互层设计及界面展示

系统通过 Web 页面的形式向用户提供可视化的操作，以便于图数据分析人员更好地进行图数据分析的相关工作。

时序图模式展示界面。该页面提供时序图数据的可视化展示功能，前端界面可以处理系统内图数据，获取实体和事件的标签，分析其组成关系，并将整个时序图模式可视化展示。

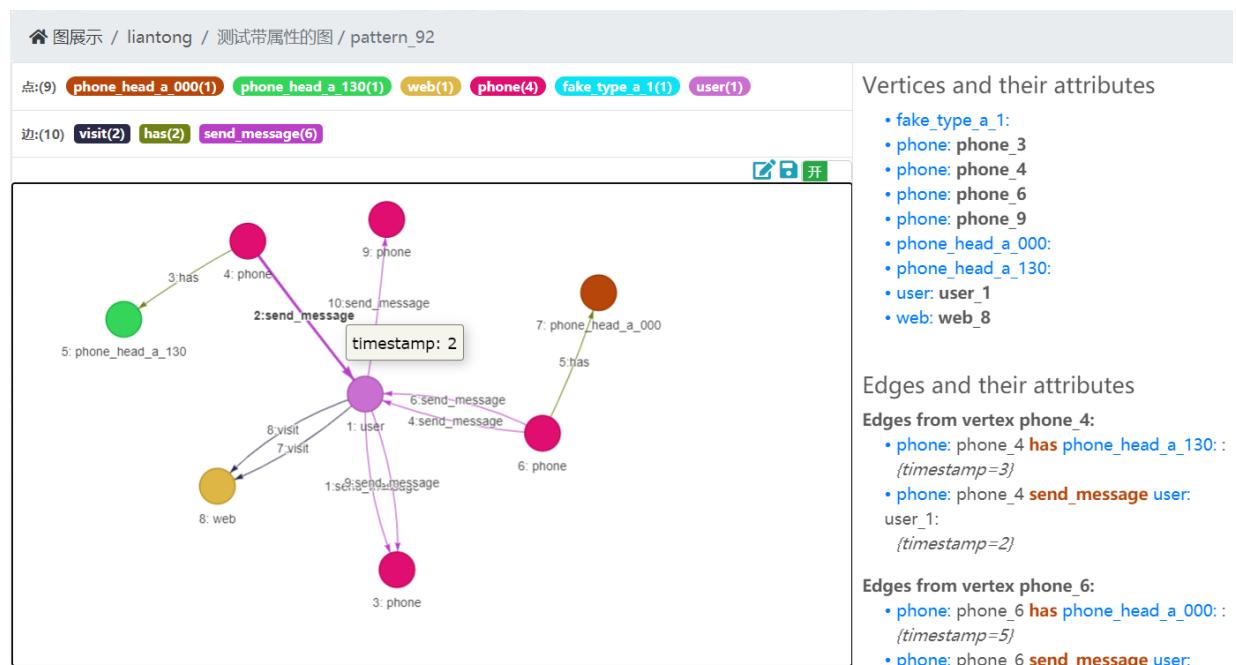


图 30 时序图模式展示界面

时序图关联规则发现工具界面。在系统中集成了时序图关联规则的发现算法，通过前端页面设定相关的参数，可以进行时序图关联规则的发现。



图 31 时序图关联规则发现工具界面

时序图匹配查询工具界面。在系统内集成了时序图匹配算法，通过前端页面设定相关参数，可以进行时序图模式的匹配查询。



图 32 时序图匹配查询工具界面

任务管理界面。通过图算法工具提交的任务会在系统后台执行，执行结束的具体信息在任务管理界面可以查询。如果是时序图关联规则发现任务则会显示任务的详情和发现规则的输出位置，如果是时序图查询任务则会显示查询所使用的时序图关联规则和在大图上查询到的匹配结果。操作和显示界面对于图数据分析人员非常友好。

序号	图集	输入图	耗时	创建时间	状态	链接
1	liantong	4999Unfold	75s	2020-10-10 12:58:10.089	执行完毕	» 浏览
2	liantong	4999Unfold	77s	2020-10-10 12:52:26.043	执行完毕	» 浏览
3	liantong	4999Unfold	82s	2020-10-10 12:13:42.672	执行完毕	» 浏览
4	liantong	-	-	2020-09-29 19:20:57.964	执行超时	» 浏览
5	liantong	-	-	2020-09-29 19:17:08.542	执行超时	» 浏览

显示第 1 到第 5 条记录, 总共 5 条记录

图 33 任务管理界面

6.5 本章小结

本章从系统工程角度描述了时序图关联规则发现和应用系统的设计与实现。系统架构设计详细介绍了该系统的架构模式，采用经典的三层架构，从下到上分别是数据层、业务逻辑层和用户交互层。数据层采用 CSV 的数据格式，以实现系统内外的数据传输；业务逻辑层主要包括图展示服务、时序图关联规则发现服务、时序图匹配服务以及任务管理服务等主要功能；用户交互层主要是系统运行时与用户交互的 Web 界面，主要包括时序图关联规则发现和时序图匹配的工具界面、时序图模式可视化展示界面和任务管理界面。

总结与展望

论文总结

本文以具有时态属性的图中的关联规则为研究方向，依次研究了时序图关联规则的形式化表示、时序图关联规则的发现算法、时序图匹配算法和基于此规则的事件预测的并行化方案，以及设计并搭建了集成时序图相关算法的系统平台。

首先，本文提出了一种结合时序和时间窗的时序图关联规则，并研究了该规则的发现问题。该规则基于时序图模式匹配，同时增加了时间窗概念，通过时间窗对整个时序图进行划分，时序图模式的匹配发生在多个时序子图上，从而降低了匹配的时间复杂度。针对时序图关联规则的发现问题：本文将支持度的定义扩充到时序图上，使其在时序图上能够满足反单调性，并考虑多种置信度的表示方式；形式化描述了发现问题并给出了相应的发现算法 TGRD，根据时序图特性在发现算法中重点给出了加边方法和边上次序的扩展方法；对发现算法给出了一系列优化策略，包括扩边引导、多线程加速、匹配时间阈值、划分时间窗口和动态支持度剪枝，此类优化策略在实验阶段证明对于发现算法的加速非常有效；并对发现算法进行了复杂度的理论分析。

然后，本文研究了时序图关联规则的应用。形式化描述了基于时序图关联规则的事件预测问题；给出了时序图模式匹配算法 TGPM，该算法在子图同构算法的匹配过程中考虑时序匹配，从而提前剪枝，具有较好的匹配效率，并给出了算法的理论复杂度分析；并给出了相应的并行算法 TGREP，并理论证明了该算法具有良好的并行可扩展性；针对多机并行所需解决的分图问题，给出了根据时间窗的图划分方法，相比按跳数的划分方法具有更低的点边重复度。

接下来，进行了实验研究。通过实验验证了时序图关联规则和发现算法 TGRD 的准确性和可扩展性，并探究了候选关联规则数量对发现准确率的影响；通过实验验证了时序图模式匹配算法 TGPM 的效率和基于时序图关联规则的事件预测并行算法 TGREP 的可扩展性。

最后，本文设计并实现了可视化的系统平台，将时序图关联规则的发现和应用算法集成于该系统中，并支持图模式的可视化展示。

工作展望

时态网络上的关联规则研究还有很大的发展空间，本文的研究内容只是提供了一种可能的思路，接下来还有很多新的工作可以开展。

无论是图上的规则挖掘还是结合了时间因素的规则挖掘，如果想要使用图的拓扑结构信息作为规则的一部分，那么必须要考虑子图同构问题的较高的复杂度，因此采用各类方法对候选关联规则进行剪枝处理是非常重要的。本文使用了支持度剪枝的策略，并尝试采用动态支持度阈值进行剪枝，具有不错的效果。对于搜索和剪枝策略，可以考虑采用一些智能算法策略和机器学习的方法。

虽然在具体应用场景中，时序图数据的规模往往会很大，但是在事件预测过程中并非所有的图数据都对结果至关重要，近似解在某些场景下是可以接受的，因此可以探索如何减少查询所需的时序图规模，给出近似解法，从而加快预测的速度。另外如何提供在线实时的事件预测也是值得探索的一个方向。

参考文献

- [1] Shenjia Ji, Hongyan Yu, et al. Research on sales forecasting based on ARIMA and BP neural network combined model[A]. ICIIP[C], 2016: 1–6.
- [2] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases[J]. SIGMOD Record, 1993, 22(2): 207-216.
- [3] P. Kam and A. Fu. Discovering temporal patterns for interval-based events[A]. DaWaK[C], 2000: 317-326.
- [4] C. Rainsford and J. Roddick. Adding temporal semantics to association rules[A]. PKDD[C], 1999: 504-509.
- [5] B. Özden, S. Ramaswamy, et al. Cyclic association rules[A]. ICDE[C], 1998: 412-421.
- [6] J. M. Ale and G. H. Rossi. An approach to discovering temporal association rules[A]. SIGAPP[C], 2000: 294-300.
- [7] Fan W, Hu C. Big Graph Analyses: From Queries to Dependencies and Association Rules[J]. Data Science and Engineering, 2017, 2(1): 36–55.
- [8] W. Fan, X. Wang, Y. Wu, et al. Association rules with graph patterns[J]. PVLDB, 2015, 8(12): 1502-1513.
- [9] W. Fan, Y. Wu, and J. Xu. Adding Counting Quantifiers to Graph Patterns[A]. SIGMOD[C], 2016: 1215-1230.
- [10] W. Fan, Z. Fan, C. Tian, et al. Keys for Graphs[J]. PVLDB, 2015, 8(12): 1590-1601.
- [11] W. Fan, et al. Functional Dependencies for Graphs[A]. SIGMOD[C], 2016: 1843-1857.
- [12] W. Fan, P. Lu. Dependencies for Graphs[A]. PODS[C], 2017: 403-416.
- [13] W. Fan, X. Liu, P. Lu. Catching Numeric Inconsistencies in Graphs[A]. SIGMOD[C], 2018: 381-393.
- [14] Kwashie S, Liu L, and Liu J. Certus: an effective entity resolution approach with graph differential dependencies [J]. Proceedings of the VLDB Endowment, 2019, 12(6): 653-666.
- [15] Ma H, Alipourlangouri M, and Wu Y. Ontology-based entity matching in attributed graphs[J]. Proceedings of the VLDB Endowment, 2019, 12(10): 1195-1207.

- [16] Y. Zheng, H. Zhang, and Y. Yu. Detecting collective anomalies from multiple spatio-temporal datasets across different domains[A]. In SIGSPATIAL[C], 2015: 1-10.
- [17] B. Zong, Y. Wu, J. Song, et al. Towards scalable critical alert mining[A]. In KDD[C], 2014: 1057-1066.
- [18] Ranshous, S., Shen, S., Koutra, D., et al. Anomaly detection in dynamic networks: A survey[J]. Wiley Interdisciplinary Reviews Computational Stats, 2015, 7(3): 223-247.
- [19] S. T. King, Z. M. Mao, D. G. Lucchetti, and et al. Enriching intrusion alerts through multi-host causality[A]. In NDSS[C], 2005.
- [20] W. R. Cheswick, S. M. Bellovin, and A. D. Rubin. Firewalls and Internet Security[M]. Addison-Wesley Professional, 2003.
- [21] W. Venema. TCP wrapper: Network monitoring, access control, and booby traps[J]. In USENIX Security, 1992.
- [22] A. Paranjape, et al. Motifs in Temporal Networks[A]. WSDM[C], 2017: 601-610.
- [23] S. Gurukar, S. Ranu, and B. Ravindran. Commit: A scalable approach to mining communication motifs from dynamic networks[A]. In SIGMOD[C], 2015: 475-489.
- [24] B. Zong, X. Xiao, Z. Li, et al. Behavior query discovery in system-generated temporal graphs[J]. VLDB, 2015, 9(4): 240-251.
- [25] V. Srinivasan, S. Moghaddam, A. Mukherji, et al. Mobileminer: Mining your frequent patterns on your phone[A]. In UbiComp[C], 2014: 389-400.
- [26] Q. Yuan, G. Cong, and A. Sun. Graph-based point-of-interest recommendation with geographical and temporal influences[A]. In CIKM[C], 2014: 659-668.
- [27] C. Song, T. Ge, C. Chen, et al. Event Pattern Matching over Graph Streams[J]. Proceedings of the VLDB Endowment, 2014, 8(4): 413-424.
- [28] MH Namaki, Y. Wu, Q. Song, et al. Discovering Graph Temporal Association Rules[A]. CIKM[C], 2017: 1697-1706.
- [29] MH Namaki, K. Sasani, Y. Wu. BEAMS: Bounded Event Detection in Graph Streams[A]. ICDE[C], 2017: 1387-1388.
- [30] X. Chen and I. Petrounias. Mining temporal features in association rules[A]. PKDD[C], 1999: 295-300.

- [31] Codd F. A Relational Model of Data for Large Shared Data Banks[J]. Communications of the ACM, 1970, 13(6): 377-387.
- [32] Bernstein P A. Synthesizing third normal form relations from functional dependencies[J]. ACM Transactions on Database Systems, 1976, 1(4): 277-298.
- [33] W. Fan, H. Gao, X. Jia. Dynamic constraints for record matching[J]. VLDB, 2011, 20(4): 495-520.
- [34] W. Fan. Dependencies Revisited for Improving Data Quality[A]. PODS[C], 2008: 159-170.
- [35] H. P. Hsieh and C. T. Li. Mining temporal subgraph patterns in heterogeneous information networks[A]. In SocialCom[C], 2010: 282-287.
- [36] K. Semertzidis and E. Pitoura. Durable graph pattern queries on historical graphs[A]. In ICDE[C], 2016: 541-552.
- [37] C. C. Aggarwal, Y. Li, P. S. Yu, and et al. On dense pattern mining in graph streams[A]. VLDB[C], 2010: 975-984.
- [38] A. Silva, W. Meira Jr, and M. J. Zaki. Mining attribute-structure correlated patterns in large attributed graphs[A]. VLDB[C], 2012: 466-477.
- [39] P. Bogdanov, M. Mongiovi, and A. K. Singh. Mining heavy subgraphs in time evolving networks[A]. In ICDM[C], 2011: 81-90.
- [40] J. Gao, C. Zhou, and J. X. Yu. Toward continuous pattern detection over evolving large graph with snapshot isolation[A]. VLDB[C], 2015: 1-22.
- [41] K. M. Borgwardt, H. P. Kriegel, and P. Wackersreuther. Pattern mining in frequent dynamic subgraphs[A]. In ICDM[C], 2006: 818-822.
- [42] Jiezhong Qiu, Jian Tang, et al. DeepInf: Social Influence Prediction with Deep Learning[A]. KDD[C], 2018: 2110-2119.
- [43] Yozen Liu, Xiaolin Shi, et al. Characterizing and Forecasting User Engagement with In-app Action Graph: A Case Study of Snapchat[A]. KDD[C], 2019: 2023-2031.
- [44] Tian Bian, Xi Xiao, et al. Rumor Detection on Social Media with Bi-Directional Graph Convolutional Networks[A]. AAAI[C], 2020.
- [45] Muhan Zhang, Yixin Chen. Link Prediction Based on Graph Neural Networks[A]. NeurIPS[C], 2018: 5171-5181.
- [46] LianTong dataset[EB/OL]. <https://jdata.jd.com/html/detail.html?id=3>, 2018-06-11/2020-

- 11-15.
- [47] JingDong dataset[EB/OL]. <https://jdata.jd.com/html/detail.html?id=2>, 2018-07-10/2020-11-15.
- [48] Karp R M. Reducibility among Combinatorial Problems[J]. Complexity of Computer Computations, 1972.
- [49] Ullmann J.R. An algorithm for subgraph isomorphism[J]. ACM, 1976, 23(1): 31-42.
- [50] B.D. McKay. Practical Graph Isomorphism[J]. Congressus Numerantium, 1981, 30: 45-87.
- [51] L.P. Cordella, P. Foggia, C. Sansone, et al. A (sub)graph isomorphism algorithm for matching large graphs[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004, 26(10): 1367-1372.
- [52] W.S. Han, J. Lee, J.H. Lee. Turbo ISO: towards ultrafast and robust subgraph isomorphism search in large graph databases[A]. SIGMOD[C], 2013: 337-348.
- [53] Ren X., Wang J. Exploiting vertex relationships in speeding up subgraph isomorphism over large graphs[J]. VLDB Endow, 2015, 8(5): 617-628.
- [54] Abedjan Ziawasch, Akcora Cuneyt G., et al. Temporal rules discovery for web data cleaning[J]. PVLDB, 2015, 9(4): 336-347.
- [55] Wenxing Hong, Lei Li, Tao Li. Product recommendation with temporal dynamics[J]. Expert Systems with Applications, 2012, 39(16): 12398-12406.
- [56] Lioni J, Qin A K, Salim F D. Optimal time window for temporal segmentation of sensor streams in multi-activity recognition[A]. MobiQuitous[C], 2016: 10-19.
- [57] Cao B, Hou C, Fan J. Covering the Optimal Time Window Over Temporal Data[A]. CIKM[C], 2017: 687-696.
- [58] Alejandro Duran, Julita Corbalan, and Eduard Ayguade. Evaluation of OpenMP task scheduling strategies[A]. International Workshop on OpenMP[C], 2008: 100-110.
- [59] William Gropp, Ewing Lusk, and Anthony Skjellum. Using MPI: Portable Parallel Programming with the Message Passing Interface[M]. MIT Press, 1994.
- [60] G. Malewicz, M. H. Austern, A. J. C. Bik, et al. Pregel: a system for large-scale graph processing[A]. In SIGMOD[C], 2010: 135-146.
- [61] Y. Low, J. Gonzalez, A. Kyrola, et al. Distributed GraphLab: a framework for machine learning and data mining in the cloud[J]. PVLDB, 2012, 5(8): 716-727.
- [62] Y. Tian, A. Balmin, S. A. Corsten, et al. From “think like a vertex” to “think like a graph”[J]. PVLDB, 2013, 7(7): 193-204.
- [63] D. Yan, J. Cheng, et al. Blogel: A block-centric framework for distributed computation on

- real-world graphs[J]. PVLDB, 2014, 7(14): 1981-1992.
- [64] W. Fan, J. Xu, Y. Wu, et al. Parallelizing sequential graph computations[A]. SIGMOD[C], 2017: 495-510.
- [65] W. Fan, J. Xu, Y. Wu, et al. GRAPE: Parallelizing sequential graph computations[J]. PVLDB, 2017, 10(12): 1889-1892.
- [66] X. Dong et al. Knowledge vault: A web-scale approach to probabilistic knowledge fusion[A]. KDD[C], 2014: 601-610.
- [67] C. P. Kruskal, L. Rudolph, and M. Snir. A complexity theory of efficient parallel algorithms[J]. Theoretical Computer Science, 1988, 71(1): 95-132.
- [68] M. A. Gallego, J. D. Fernandez, et al. An empirical study of real-world SPARQL queries[A]. In USEWOD workshop[C], 2011.
- [69] P. Burkhardt and C. Waring. An NSA big graph experiment[R]. Technical Report NSA-RD, 2013.
- [70] Albert-László Barabási, Réka Albert. Emergence of Scaling in Random Networks[J]. Science, 1999, 286(5439): 509-512.

致谢

一个人可以选择度过一天的方式，或忙碌，或慵懒，或开心，或悲伤，但是却无法拒绝落日与朝阳；一个人可以选择度过一生的方式，或踌躇满志，或默默无闻，但是却无法逃避获得与失去。

时光总是在不知不觉间溜走，在北航的学习生活也临近尾声，我常常思考每天的朝阳与落日有何不同，反思自己的得与失。但 2020 年注定是艰难的一年，大到国家与社会，小至家庭与个人。冬天过去了，春天还会远吗？我像每一个年轻人一样用充满希望的眼眸领略四季的更迭，但总有许多人和事永远留在了上一个冬日。在这里，感谢冬日里的每一缕温暖阳光。

首先，感谢我的父母，无论在任何时候，你们都始终理解和支持我，尊重我做的决定，给予我充足的物质保障和关怀，让我不必为生计而苦恼，可以自由地从事自己喜欢的工作，过自己想要的生活。

然后，感谢我的老师、同学和朋友们。感谢导师樊文飞老师、邓婷老师、陆平老师、黄贝宁老师、纪寿庆老师在整个研究生阶段对我的悉心教导和帮助。感谢韩军老师、李建欣老师、张日崇老师对我毕设方向和内容的建议。感谢于文渊老师、徐静波学长、罗小简学长对我科研和技术上的帮助和指导。感谢孙广志同学平日的陪伴与帮助，一起吃饭吐槽缓解生活的压力。感谢黄婧婧同学的帮助，感谢王云杰同学、赵阳艳同学的帮助。还有实验室曾经的同学们，感谢李东泽学长、曾维彬学长、吴丁禾学长、侯蕾学长的帮助，感谢张思远同学、刘沐阳同学、孟子中同学、姜周希雨同学的帮助，一起玩耍的时光很快乐。感谢大学和高中的室友们，一起分享快乐与压力。感谢工作和生活中结交的朋友们，无论是专业技术还是为人处世，你们都教会我很多，并给予了我很多帮助。感谢北京市大数据与脑机智能高精尖中心提供的计算集群。无论大家身处何方，都愿前程似锦，平安喜乐。感谢冬日中给予我帮助的人们，无论曾经，现在，还是未来。

接下来，感谢安澜一直以来的支持与陪伴，分享我的快乐，分担我的忧愁。往后的时光，希望我们继续分担寒潮、风雷、霹雳，共享雾霭、流岚、虹霓。

最后，感谢我自己，感谢我拥有一颗乐观而强大的内心和执着坚持的勇气。愿我

致谢

走出半生，归来仍是少年。