ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.12, December 2006, pp.2416−2424
DOI: 10.1360/jos172416

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

一种主动容错的序列流并行分析算法*

赵 峰[1,2+], 李庆华[1,2], 金 莉[1]

[1](华中科技大学 计算机科学与技术学院,湖北 武汉 430074)

[2](国家高性能计算中心(武汉),湖北 武汉 430074)

A Parallel Analysis Algorithm for Sequence Stream Based on Proactive Fault Tolerance

ZHAO Feng[1,2+], LI Qing-Hua[1,2], JIN Li[1]

[1](School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China)

[2](National High Performance Computing Center (Wuhan), Wuhan 430074, China)

+ Corresponding author: Phn: +86-27-63264292, Fax: +86-27-87544471, E-mail: jimmyzf@gmail.com, http://www.hust.edu.cn

Zhao F, Li QH, Jin L. A parallel analysis algorithm for sequence stream based on proactive fault tolerance. *Journal of Software*, 2006,17(12):2416−2424. http://www.jos.org.cn/1000-9825/17/2416.htm

Abstract: A parallel sequence stream analysis algorithm named FTPSA (proactive fault-tolerant parallel sequence stream analysis algorithm) is proposed in order to deal with sequence stream's adaptive analysis in noisy environment, which is based on proactive fault-tolerant knowledge learning. The algorithm utilizes learning network to describe sequence stream and stores those in 0-1 matrix, delaminates the low-proportion and the high-proportion noisy data and utilizes fault-tolerant and structure-optimize learning methods, utilizes global filtration to depress memory cost and communication cost. The experimental results on real stream show that FTPSA algorithm is more fault-tolerant, scaleable, accurate, and less memory.

Key words: sequence stream; proactive fault tolerance; knowledge learning; parallel algorithm

摘 要: 提出一种主动容错的序列流并行分析算法——FTPSA 算法(proactive fault-tolerant parallel sequence stream analysis algorithm),以解决噪声环境下大规模序列流的自适应分析问题.算法利用学习网络描述流序列,并存于 0-1 矩阵中;将低比例和高比例不良数据分层考虑,分别采用基于容错和基于结构优化的学习方法;同时,经过全局筛选,有效地减少了中间结果集合,降低了内存和通信消耗.真实数据集上的实验结果表明,FTPSA 算法准确率高,占用的存储空间小,并有良好的容错性和扩展性.

关键词: 序列流;主动容错;知识学习;并行算法
中图法分类号: TP391 文献标识码: A

序列流分析是数据流挖掘的一个重要分支,序列数据之间的关联关系是我们认识世界的主要方法之一,它帮助我们解释序列之间的关系并预测事件发展的规律,在商业决策、信息安全和科学计算等多个领域有着重要的应用.人类社会电子化的日益增强,流数据的数量、种类和规模都在不断增大.同时,数据流中也存在大量的不完备数据、溢出数据、噪声数据等不良数据.如何在小运行时间和低存储空间内快速、高效地从大规模数据流

* Supported by the National Natural Science Foundation of China under Grant Nos.60503048, 60273075 (国家自然科学基金)
Received 2005-10-08; Accepted 2006-01-20

中分析出序列相关性,并主动排除不良数据对分析过程的干扰,是序列流分析面临的最主要的挑战.本文设计了一种主动容错的序列流并行分析算法——FTPSA 算法(proactive fault-tolerant parallel sequence stream analysis algorithm),用贝叶斯学习发现噪声环境中序列流数据之间的相关性.真实数据集上的实验结果表明该算法是可行和有效的.

1 相关研究

许多学者对序列分析进行了广泛研究,提出了一些分析算法:文献[1]提出了 GSP,WINEPI,CSPADE,MSDD 等泛化的序列分析的算法,这些算法计算中间结果的时间复杂度较高;文献[2–4]提出了 Count 分布算法、Data 分布算法、Candidate 分布算法、EVE 算法等并行序列分析算法,用于降低序列计算的时间复杂度,但这些算法仅适用于离散形式的序列数据库,且准确率不高.

文献[5–10]将机器学习思想引入序列分析算法中,以提高算法的准确度:文献[5]提出了一种基于 3 阶段机制的学习算法 Algorithm A,该算法的优点是准确度高,但其 CI(conditional independence)计算上的高效是以空间上的巨大开销作为代价的,并且其计算 d-separate 集的时间开销仍然很大;文献[6]中提出了采用 $n\times n$ 的矩阵存储节点的 HEA 算法,该算法大幅度降低了空间开销,但该算法过分依赖 CI 测试结果的正确性,并且采用了功效搜索来矫正 CI 测试的错误结果.虽然其 CI 计算的时间复杂度为 $O(n^3)$,但其功效搜索的时间复杂度仍为 $O(2^n)$;文献[7,8]提出用于数据挖掘的演化算法,但其是以巨大的计算时间为代价来换取分析结果的高准确性的.

文献[10–12]提出了几种面向数据流的挖掘算法:文献[10]提出了面向多维数据流的贝叶斯在线学习算法;文献[11]提出用线形回归方法分析高维时序数据流;文献[12]提出了基于不同时间粒度的数据流频繁模式挖掘的方法.这些算法可以有效地从数据流中提取知识,但其分析过程占用的存储空间太大,且无法排除噪声数据的干扰.

2 基本概念

定义 **1**. 非空集合 $I=\{i_1,i_2,i_3,\ldots,i_m\}$称为项集,其中 i_k 称为项.

定义 **2**. 序列是项集的有序表,记为 $s=s_1\rightarrow s_2\rightarrow\ldots\rightarrow s_n$,其中,$s_k\subset I(k=1,\ldots,n)$.含有 k 个项的序列长度为 k,称为 k 序列($k=\sum|s_i|$).

序列之间的相关性称为序列关联,可表示为 $L=\langle s_1s_2s_3\ldots s_n\rangle$,其中,$s_i$ 为事件或事件集,若 s_i 在 s_j 之前发生,则 s_i 为前序事件,s_j 为后序事件.长度为 k 的序列模式记为 L_k.如果序列出现得足够频繁,我们就称它为频繁序列.决定模式是否有效的参数有:支持度、可信度、重要性和覆盖度[1].本文算法中,我们定义了一种加权的阈值ξ的计算方式,若决定序列是否频繁的参数记为支持度 sp、可信度 cn、重要性 sg 和覆盖度 cv,它们的权值分别为 $w_{sp},w_{cn},w_{sg},w_{cv}$,则判断序列是否频繁的阈值为 $\xi=\sum_{i=\{sp,cn,sg,cv\}}v_iw_i$.

定义 **3**. 贝叶斯网络是一种将贝叶斯概率和有向无环图的网络拓扑结构有机结合的表示模型[5],描述了元组数据项及元组和元组之间的相互依赖关系.与一个贝叶斯网络结构相同的无向无环图称为该贝叶斯网络的网架.

在本文的算法中,一个贝叶斯网络可表示为(G,p),其中:$G=(V,E)$是有向无环图,V 是 G 的节点集,表示序列项集的集合;E 是 G 的边,表示 V 中节点之间的概率联系.边 $E=\langle XY\rangle$表示 $X\rightarrow Y$,其中:X 是父节点;Y 是子节点,节点 Y 的所有父节点的集合记为 $\prod_Y$.

定义 **4**. V 中节点 x_1,x_2,则 x_1 和 x_2 的互信息[10]可表示为 $I(x_1,x_2)=\sum_{x_1,x_2}p(x_1,x_2)\log\frac{p(x_1,x_2)}{p(x_1)p(x_2)}$.若 $I(x_1,x_2)$小于某一指定的阈值ξ,则称 x_1 和 x_2 边缘独立.

定义 **5**. V 中节点 x_1,x_2,则 x_1 和 x_2 的条件互信息[10]可表示为 $I(x_1,x_2\mid S)=\sum_{x_1,x_2}p(x_1,x_2,S)\log\frac{p(x_1,x_2\mid S)}{p(x_1\mid S)p(x_2\mid S)}$,其中,$S\subset V$.若 $I(x_1,x_2)$小于某一指定的阈值ξ,则称 x_1 和 x_2 条件独立.

3 FTPSA 算法

在基于知识学习的序列分析算法中,常用的是基于统计、神经网络、贝叶斯学习的分析方法.基于统计的方法速度快,但准确率低,且易受噪声数据影响;基于神经网络的方法准确率高,但速度慢,可扩展性差;基于贝叶斯学习的方法,其速度快,准确率高,并且有较好的容错性和扩展性,可适应大规模数据流对序列分析准确性、容错性及实时性的要求.

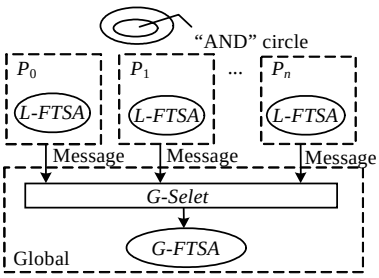

Fig.1 FTPSA algorithm

图 1 FTPSA 算法

FTPSA 的基本思想是:将输入数据和候选贝叶斯网络分别分布和复制到所有处理器 $P_0,P_1,\ldots,P_n$ 上,序列关联候选集在各个处理器上线性生成.候选集存在于分布的贝叶斯网络列表中,贝叶斯网分布本着让所有处理器上的候选集数目尽可能相等的原则.FTPSA 算法中处理器需要处理输入数据和序列候选集,当序列持续的时间值较小时,可将候选集静态置于处理器上,建立本地贝叶斯网络结构,而数据流采取"与"循环的方式输入;当序列持续的时间值较大时,将本地贝叶斯网络传到集中处理器上处理,如图 1 所示.

3.1 序列的贝叶斯网络模型

序列可分为连续序列、并行序列及混合序列.连续序列中,数据呈线性输入,任何数据之间有严格的先后关系;并行序列中,任何一组数据中的数据呈并列方式,数据和数据之间没有先后之分,出现在相同时刻;在混合序列中,既有连续序列也有并行序列,是两者的混合.鉴于序列的不同种类的多发性、序列的有序性、序列不循环的特性,我们可以用贝叶斯网络来描述不同类型的序列,如图 2 所示.

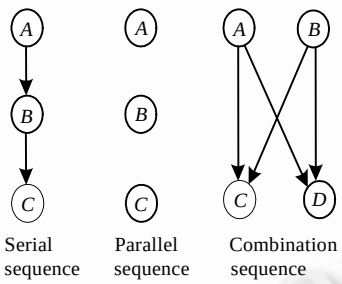

Fig.2 Bayesian network for sequence

图 2 序列贝叶斯网络模型

3.2 数据结构

FTPSA 算法采用 Larrange 在文献[7]中提出的 $n\times n$ 的矩阵 M 来存储有 n 个节点的贝叶斯网络,矩阵元素 M_{ij} 定义为

$$M_{ij}=\begin{cases}1, & \text{如果节点 } j \text{ 是节点 } i \text{ 的父节点}\\ 0, & \text{其他}\end{cases},$$

其中,第 i 行表示节点 i 的父节点的集合 $\prod_i$.图 3 是一个贝叶斯网矩阵存储的图例.

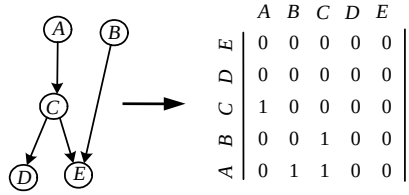

Fig.3 Matrix structure for Bayesian network

图 3 贝叶斯网络矩阵存储图例

3.3 算法步骤

① 设有处理器 $P_1,P_2,\ldots,P_n$,各处理器在本地采集系统数据,再在这些数据集上通过 L-FTSA 算法构建本地贝叶斯网络(local Bayesian networks,简称 LBN)序列.

② 在每个处理器上,判断生成的 LBN 序列是否频繁:如果不频繁,则丢弃;如果频繁,则保留,并将 LBN 序列传送到全局处理器(global)中作进一步判断.

③ G-Select 选择性地接受 $P_1,P_2,\ldots,P_n$ 传送来的数据,形成 Global 数据集.

④ 在 Global 数据集上,通过 G-FTSA 算法构建全局贝叶斯网络 GBN(global Bayesian networks)序列.

⑤ 判断生成的 GBN 序列是否频繁,如果频繁,则保留.

4 基于容错的 L-FTSA 算法

基于条件独立测试的学习算法是构建贝叶斯网络的主要方法之一,国内外众多学者对此作了广泛的研究.但这些算法存在如下缺陷:① 基于 CI 测试的算法其时间复杂度都很高,最坏情况下,每个 CI 测试至少需要指数时间;② 算法的准确性依赖于条件集的规模,条件集越大,算法的准确性越低;③ 由于 CI 测试是逐步计算的,早期的噪声数据会给后期的结果造成较大的影响,算法的容错性很差.针对以上缺陷,结合序列数据特性,本文设计了一种基于容错的贝叶斯学习算法——L-FTSA(local-FTSA)算法.该算法侧重于提高运行速度,降低存储空间及主动降低噪声数据的干扰.

在序列流中,对于数据量少的数据种类,其 CI 测试的条件集是大规模的,我们称其为“高序”;反之,对于数量大的数据种类,其 CI 测试的条件集是小规模的,我们称其为“低序”.对于(G,p),有$\forall G=(V,E),X,Y\in V$ 且 $X\neq Y$.令 $C\subset V\backslash\{X,Y\}$,若 C 中不存在 X,Y 的可连接路径,则称 C 是 X,Y 的 CI 测试集.对于序列流 $x_1,x_2,\ldots,x_n$ 网络模型,有 $C=V\backslash\{I|I\in\{x_1,x_2,\ldots,x_n\}\}$,对于$\forall E=\langle I_xI_y\rangle,I_x\in x_i,I_y\in x_j$,有 $C=V\backslash\{\{I|I\subset E,E=\langle I_yI_x\rangle\}\cup\{I|I\subseteq E,E=\langle I_xI_y\rangle\}\}$.

鉴于序列的先后次序及无循环特性,必然有 $i<j$,也即 $C=V\backslash\{I|I\subseteq E,E=\langle I_xI_y\rangle\}(x<y)$.显然,$\{X,Y\}$越大,集合 $\{I|I\subseteq E,E=\langle I_xI_y\rangle\}$越大,$C$ 越小.故而在基于贝叶斯网络学习的序列流分析中,所有不良的数据都有一个共同的特点:不良数据比正常数据的数量要少,故源于它们的 CI 测试的条件集必然是“高序”的.

4.1 算法步骤

步骤 1. 贝叶斯网络矩阵初始化.

设有 n 个项(集)1,2,…,n,每个项(集)是一个节点,对于任意节点 i,令 $\prod_i=\{1,2,\ldots,i-1,i+1,\ldots,n\}$ $(1\leq i\leq n)$.

步骤 2. 构建冗余贝叶斯网络.

以最早出现的序列项(项集)作为根节点,寻找所有可能存在的节点连接,构造冗余的贝叶斯网架;以序列项发生的先后次序为准,确定冗余贝叶斯网架中边的指向,调整 E 中节点对,使$\langle node1,node2\rangle$表示从节点 1 指向节点 2 的边,以便生成冗余贝叶斯网络;

① 初始化:令 $G=(V,E)$,其中 $V=${序列项/项集};$E=\varnothing$;$n=$序列项集个数;临时线性表 $L=\varnothing$;

② for (i=1;i<(n+1)/2;i++)

{ *Root*=V[i]; //指定根节点

③ while ($i\neq j$) and ($v[j]\in V$)

{计算 $I(v[i],v[j])$,其中,将所有 $I(v[i],v[j])>\xi$的节点对按递减顺序存入 L;}

④ while ($L\neq\varnothing$)

{从 L 中依次取出节点对,连接两节点,将边添加到 E 中;}

}

步骤 3. 精简贝叶斯网络,删除多余的边.

需要删除的边包括两种类型:一是由错误数据生成的边;二是本身不具有序列相关性的边.设有节点 X,Y,若 X 和 Y 条件独立,则从 $\prod_X$ 中删除 Y,并从 $\prod_Y$ 中删除 X,同时标记要删除边 $X\rightarrow Y$ 和边 $Y\rightarrow X$.

步骤 4. 恢复错误删除的边,构建完整的贝叶斯网络.步骤 3 有可能删除了那些不应删除的边,需要将错误删除的边重新恢复.

⑤ for (已标记删除的每条边$\langle node1,node2\rangle$) {

⑥ 对于$\langle node1,node2\rangle$,计算 S,其中

$S=\{node\in V|node$ 可以 d-separate[5]节点 $node1$ 和 $node2\}$;

⑦ 计算 $I(node1,node2|S)$;

if ($I(node1,node2)|S)<\xi$)

将边$\langle node1,node2\rangle$从 E 中永久删除;

else

将去掉边$\langle node1,node2\rangle$的删除标记;}

4.2 算法分析

L-FTSA 算法采用 $n\times n$ 的 0-1 矩阵存储节点,空间开销为 $O(n^2)$,对于值域 U 上的 n 个节点的贝叶斯网,其任意一对节点 X,Y 之间 CI 测试的计算最坏情况下,只要计算 $I(X,\varnothing,Y)(M_{ij}=0)$和 $I(X,Z,Y),Z\in U\backslash\{X,Y\}(M_{ij}=1)$.故其计算互信息 $I(X,Y)$需要 $O(n^2)$时间,则 CI 测试的计算时间复杂度为 $O(n^2)$.L-FTSA 算法和几个经典的贝叶斯学习算法的 CI 计算开销的比较见表 1.

Table 1 The comparison of CI computing consume for different algorithms

表 1 不同算法 CI 计算开销对比

Algorithm	Time use	Space use
A algorithm[5]	$O(n^2)$	$O(2^n)$
HEA algorithm[6]	$O(n^3)$	$O(n^2)$
DMEL algorithm[8]	$O(2^n)$	$O(2^n)$
MDLEP algorithm[10]	$O(2^n)$	$O(n^3)$
L-FTSA algorithm	$O(n^2)$	$O(n^2)$

5 Global 处理机制

当序列的跨距时间很长时,单个处理器无法找到其所有频繁序列,需要在 Global 中进一步处理.我们在采用 FTPSA 算法的 Global 处理机制时,从以下 3 个方面来提高算法效率:① 处理器向 Global 中传送的不是原始数据集而是 LBN,这样,Global 便可利用 L-FTSA 算法的计算结果,简化分析过程;② Global 并不是接受所有的 LBN,而是通过 G-Select 选择性地接受,它只接收与其他 LBN 可信关联的 LBN,这样便可屏蔽无关的数据集,减小 Global 处理的数据量;③ Global 在处理跨距时间较长的序列时,一方面要寻找完整的频繁序列,另一方面也要寻找序列的最优贝叶斯模型,以便后续学习.为了与 L-FTSA 算法无缝结合,本文设计了基于最优选择的 G-FTSA (global-FTSA)算法.

5.1 G-Select机制

为了降低通信消费、减少数据传输量,提高算法效率,处理器 $P_1,P_2,\ldots,P_n$ 在向 Global 提交数据时,传送的并不是原始数据,而是本地构建的贝叶斯网络(LBN).G-Select 机制的作用在于判断一个 LBN 是否与其他处理器相关联,以便决定该 LBN 是否有效,再选择性地接收有效的 LBN.在 FTPSA 算法中,G-Select 仅接收那些与其他处理器有关联的 LBN,也即仅接受关联概率 $P(G)$大于某一指定的阈值ξ的 LBN.

不失一般性,设有处理器 P_1,P_2 及 P_1,P_2 构建的局部贝叶斯网 G_1,G_2,S_1,S_2 是 G_1,G_2 的节点集合,$S=S_1\cap S_2$,则 G_1,G_2 的关联概率为

$$P(G_1)=P(S_1\mid cn(G_1))=\prod p(x,x\in S_1\mid cn(G_1)),\ P(G_2)=P(S_2\mid cn(G_2))=\prod p(x,x\in S_2\mid cn(G_2)).$$

其中,$cn(G_1)=\{x|x\in(S_2-S)$ and x 和 S 关联$\}$,$cn(G_2)=\{x|x\in(S_1-S)$ and x 和 S 关联$\}$.

如图 4 所示,G_1,G_2 分别位于不同的处理器 P_1,P_2 上,则 S_1={A,E,B,C,D,G},S_2={F,H,B,C,D,G},S={B,C,D,G},$cn(G_1)$={F,H},$cn(G_2)$={A}.故有 $P(G_1)=p(A)p(E|A)p(C|B)p(B|A,F)p(D|C)p(G|C,H)$,$P(G_2)=p(F)p(H|F)p(C|B)p(B|A,F)p(D|C)p(G|C,H)$.若 $P(G_1)>\xi$且 $P(G_2)>\xi$,则 Global 接收;否则丢弃.

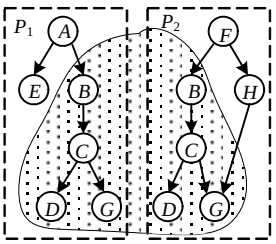

Fig.4 G-Select mechanism
图 4 G-Select 机制图例

G-Select 机制在收集数据时首先计算各处理器传来的每两个 LBN 的 $P(G)$,保留 $P(G)>\xi$的 LBN;然后再计算每 3 个 LBN 的 $P(G)$,保留 $P(G)>\xi$的 LBN;依此类推,直到没有 $P(G)>\xi$的 LBN 为止.若处理器数目为 n,每个处理器向 Global 传送的 LBN 平均数量为 m,则 G-Select 计算 $P(G)$的总次数为 $O((m\times n)^2)$.

5.2 基于结构优化的G-FTSA算法

由于原始数据集中含有噪声数据,G-Select 选择后的 LBN 有可能是正常数据的 LBN,也有可能是不良数据的 LBN.虽然 L-FTSA 算法剔除了“高序”CI 测试集的 LBN,但有一种情况是 L-FTSA 算法没有考虑到的,当不良数据的数量过大时,其 CI 测试集也有可能是“低序”的,L-FTSA 算法并没有完全剔除不良数据的影响,也即 L-FTSA 算法构建的 LBN 并不是最优的,我们称这些 LBN 为 F-LBN(fault-LBN).如果 F-LBN 与其他 LBN 没有关联,那么,通过 G-Select 就可以排除;若 F-LBN 与其他 LBN 有关联,它进入 Global 数据集,就需要对其作进一步处理.

G-FTSA 算法的基本思想是:若两个 LBN(G_1,G_2)的贝叶斯网架相似,并且其节点集相同,则有可能是一序列及其受不良数据影响的序列,G-Select 算法试图以 G_2 为参考优化 G_1,以得到最优的 LBN 结构.如果可以优化,则返回优化的结果;否则返回 G_1.然后,再在最优结构的基础上与其他 LBN 连接,形成 GBN(global Bayesian network),一般得到完整的频繁序列.算法如下所示:

输入:*LBN*(*G*(*V*,*E*)),处理器数量*n*,处理器LBN最大数量*m*.
输出:存放全部GBN的*List*.
List=∅; //存放GBN的容器
for (*k*=2;*k*≤*n*;*k*++) {
  Vs=∅; //关联LBN的节点集合
  for (*i*=1;*i*≤*m*;*i*++) {
    计算$G_1[i],\ldots,G_k[i]$的$P(G)$;
  if ($P(G_1),\ldots,P(G_k)>\xi$) {
    Vs=*Merge*($V(G_1)\cup\ldots\cup V(G_k)$);
  List[*k*]=*L-FTSA*(*Vs*)算法生成GBN;}
}
}

经过 L-FTSA 算法处理后的网架相似、节点集相同的 LBN 的数量一般是 2 个,最多是 log($m\times n$)个(处理器数目为 n,每个处理器向 Global 传送的 LBN 平均数量为 m).

Merge 过程首先采用两两比较进行结构优化,然后在中间优化结果的基础上再两两比较优化结构,直到产生最优的 LBN 结构为止,最坏情况下要比较 log(log($m\times n$))次.

设有 G_1,G_2,其节点集合分别为 U_1,U_2,任意节点 $N_i\in\{U_1\cap U_2\}$,令其分别在 G_1,G_2 中的贝叶斯概率为 P_i^1,P_i^2,父节点集合为 $\prod_i^1$,$\prod_i^2$.Merge 过程两两比较、优化 LBN 结构的算法如下:

① 对于 $N_i\in\{U_1\cap U_2\}$ (0<i<log($m\times n$)),计算 $\Delta P_i=P_i^1-P_i^2$,并按ΔP_i从大到小排序将节点存入到列表 L 中;
② 设置优化节点集合 S=∅;
③ while ($L\neq\varnothing$) {
  设置临时节点集合 S_1=∅;
  若 L_i 不在 S_1 中,将 L_i 存到 S_1 中;

④ for (S_1中的每个节点) {
将其与 G_2 中的节点 N_i^2 比较:若它们之间在 G_1 中存在直接或间接的边,则在 S_1 中保留该节点;否则删除;
}
⑤ 对于$\Delta S=S_1-S$中的每个节点ΔS_i,计算它的ΔP_i,若$\Delta P_i>0$,则用 $\prod_i^2$ 替换 $\prod_i^1$.
⑥ $S=S\cup S_1$;
⑦ 取 L 的下一个节点;
}
⑧ 返回修改的 G_1 的矩阵存储结构 M.

Merge 过程的计算时间主要集中在第③步的 while 循环,最坏情况下,循环时间复杂度为 $O(\log(m\times n))$,也即 $O(\log n^2)$;第④步的节点比较查询是否有链路存在需要 $O(n)$时间;第⑤步ΔS 中节点最大个数不会超过 L 中节点个数.计算ΔP_i需 $O(n)$时间,替换最坏情况下需要 $O(n)$时间;故而,Merge 过程计算的时间复杂度为 $O(n^2\log n)$.

6 实验结果及分析

为验证 FTPSA 算法性能,我们在真实的数据集 DARPA98 和 KDDCUP99 上进行了实验,DARPA98 是 MIT Lincoln Labs.用于入侵检测的数据集,作为检测系统来使用.KDDCUP99 数据集是 DARPA98 数据集的一种扩展,增加了一些构造特征,这个数据集主要是为数据挖掘算法应用而构造的,我们仅使用了一个 10%的子集(包含 494 020 记录)来测试算法的性能.实验环境为 4 台 P2.0G/256M 的 PC 机和 DAWNING3000A,编程基于 AIX/C++,Linux/C++, LibPcap 和 MSBNX1.4,4 台 PC 用于获取的不完整网络记录作为不良数据.

我们将从两个方面验证 FTPSA 算法的性能:一是检测分析出的频繁序列的准确率及其效率;二是检测 FTPSA 算法的容错能力.

图5、图6给出了在不同的阈值ξ(0.1,0.2,0.3,0.4,0.6,0.8)下,FTPSA算法与HEA算法、EVE算法在DARPA98和KDDCUP99数据集上分析出的模式数量的对比.实验结果表明,本文提出 FTPSA 算法分析序列时,挖掘出的模式数量受主观阈值ξ取值的影响要小得多.

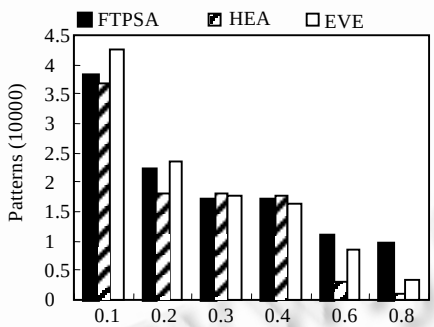

Fig.5 Pattern amount comparison on DARPA

图 5 DARPA 数据集上模式数量的比较

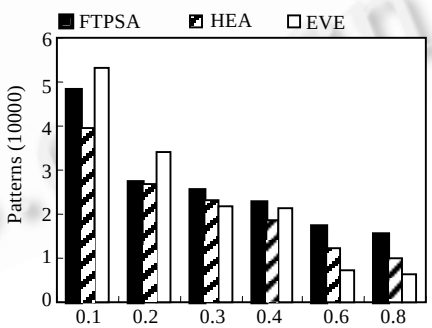

Fig.6 Pattern amount comparison on KDDCUP99

图 6 KDDCUP99 数据集上模式数量的比较

图 7、图 8 给出了在噪声数据占总体数据不同比例(0.01,0.05,0.1,0.15,0.2,0.25,0.3)情况下,FTPSA 算法与 HEA 算法、EVE 算法在 DARPA98 和 KDDCUP99 数据集上算法执行时间的对比.实验结果表明:① FTPSA 算法的执行时间受不良数据的影响要小得多;② 当含有噪声数据时,FTPSA 算法的执行时间要优于其他算法,不良数据占的比例越大,这个优势就越明显.

图9、图10给出了在噪声数据占总体数据不同比例(0.03,0.1,0.15,0.2,0.25,0.3)的情况下,FTPSA算法与HEA算法、EVE 算法在 DARPA98 和 KDDCUP99 数据集上分析出的模式准确率的对比.实验结果表明:① 当不良数据的比例低于 30%时,FTPSA 算法有很好的容错性,分析模式的准确率受不良数据的影响很小;② FTPSA 算法分析模式的准确性要优于其他算法.

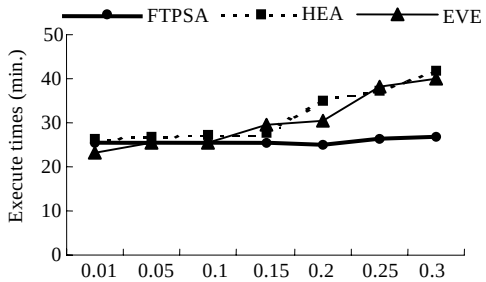

Fig.7 Execute time comparison on DARPA

图 7 DARPA 数据集上执行时间的比较

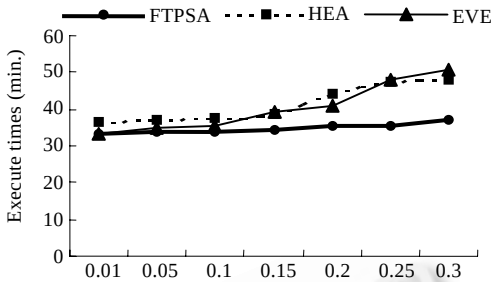

Fig.8 Execute time comparison on KDDCUP99

图 8 KCCCUP99 数据集上执行时间的比较

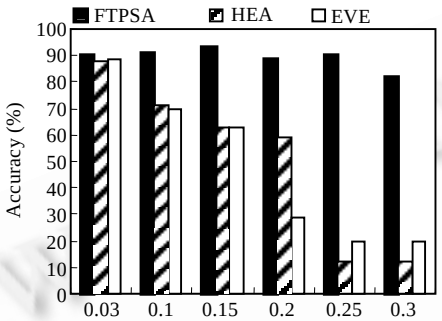

Fig.9 Accuracy comparison on DARPA

图 9 DARPA 数据集上模式准确率的对比

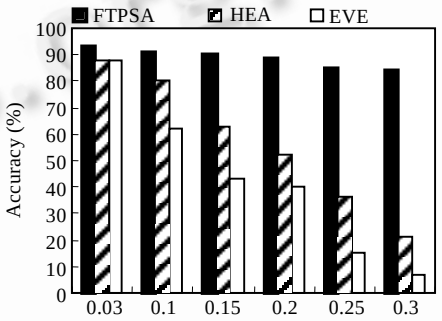

Fig.10 Accuracy comparison on KDDCUP99

图 10 KDDCUP99 数据集上模式准确率的对比

我们随机选取了 DARPA98 和 KDDCUP99 中的 10 000 条记录,通过改变不良数据的比例(0.03,0.1,0.15,0.2,0.25,0.3)对算法进行了跟踪,图 11、图 12 给出了执行 FTPSA 算法与 HEA 算法、EVE 算法时 CPU 使用情况的对比.实验结果表明:当不良数据的比例低于 30%时,FTPSA 算法的稳定性更好,所占用的 CPU 资源基本相同;而 HEA 和 EVE 算法占用的 CPU 资源情况则出现较大的波动.

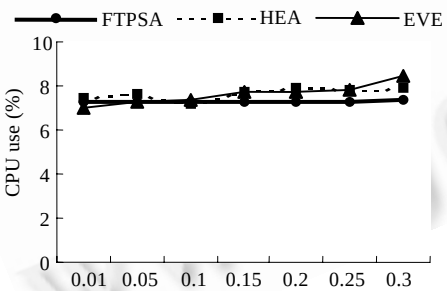

Fig.11 CPU use comparison on DARPA98

图 11 DARPA98 数据集上 CPU 占用情况的对比

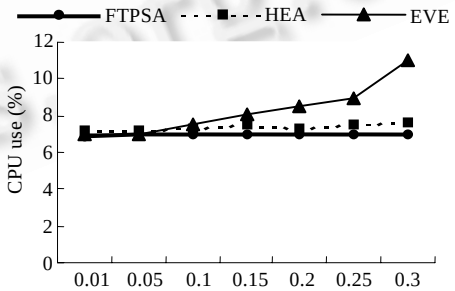

Fig.12 CPU use comparison on KDDCUP99

图 12 KDDCUP99 数据集上 CPU 占用情况的对比

7 结束语

本文提出一种基于容错的贝叶斯改进学习算法,在此基础上设计了主动容错的流序列分析并行算法——FTPSA 算法.该算法特别适用于含有不良数据的大规模数据流的序列分析.FTPSA 算法的优点在于:① FTPSA 算法在单个处理器和集中处理器上分别采用 L-FTSA 和 G-FTSA 算法剔除不良数据产生的中间结果,考虑了不同比例不良数据的影响,使得算法可以有效地容错;② 本文提出的 L-FTSA 算法是对文献[5]中 Algorithm A 学

习算法的改进.在保持 CI 测试计算时间为 $O(n^2)$的情况下,采用 $n\times n$ 的 0-1 矩阵存储贝叶斯网,降低了空间复杂度,使算法的存储空间从 $O(2^n)$降至 $O(n^2)$;③ 传统序列分析算法挖掘的中间结果都很大,分析这些数据需要大量的时间,本文提出的 G-Select 选择性接收中间结果集降低了通信消耗,减少了数据传输量,提高了算法效率.

本文的今后研究工作将着眼于进一步提高序列流分析的自适应性及算法在安全免疫系统、入侵容忍系统中的实际应用.

致谢 感谢评审专家给我们提出了宝贵的意见.

References:

[1] Jussi A. Ming sequential patterns. Technical Report, No.TTE1-2001-10, Finland: VTT Information Technology, 2001.

[2] Mahesh VJ, George K, Vipin K. Parallel algorithms for mining sequential associations: Issues and challenges. Technical Report, No.00-002, Minneapolis: University of Minnesota, 2000.

[3] Agrawal R, Imielinski T, Swami A. Mining association rules between sets of items in large databases. In: Buneman P, ed. Proc. of the '93 ACM SIGMOD Int'l Conf. on Management of Data. Washington: ACM Press, 1993. 207−216.

[4] Agrawal R, Shafer JC. Parallel mining of association rules. IEEE Trans. on Knowledge and Data Engineering, 1996,8(6):866−883.

[5] Chen J, Bell D, Liu W. Learning Bayesian networks from data: An efficient approach based on information theory. Artificial Intelligence, 2002,137(1-2):43−90.

[6] Wong ML, Leung KS. An efficient data mining method for learning Bayesian networks using an evolutionary algorithm-based hybird approach. IEEE Trans. on Evolutionary Computation, 2004,8(4):378−404.

[7] Wong ML, Lam W, Leung KS, Ngan PS, Cheng JCY. Discovering knowledge from medical databases using evolutionory algorithms. IEEE Trans. on Pattern Anal. IEEE Engineering in Medicine and Biology Magazine, 2000,4(19):45−55.

[8] Au WH, Chan KC, Yao X. A novel evolutionary data mining algorithm with applications to churn prediction. IEEE Trans. on Evol. Comput, 2003,12(7):532−545.

[9] Chen R, Sivakumar K, Kargupta H. An approach to online Bayesian learning from multiple data streams. In: Hargupta H, Sivakumar K, Wirth R, eds. Proc. of the Workshop on Mobile and Distributed Data Mining. Freiburg: IEEE Press, 2001. 31−45.

[10] Wong ML, Lam W, Leung KS. Using evolutionary programming and minimum description length Princple for data mining of Bayesian network. IEEE Trans. on Pattern Analysis and Machine Intelligence, 1999,2(21):174−178.

[11] Chen YX, Dong G, Han J. Multi-Dimensional regression analysis of time-series data streams. In: Bernstein PA, Loannidis YE, Ramakrishnan R, eds. Proc. of the 28th Int'l Conf. on Very Large Data Bases. Hong Kong: IEEE Press, 2002. 323−334.

[12] Teng WG, Chen MS, Yu PS. A regression-based temporal pattern mining scheme for data streams. In: Freytag JC, Lockemann PC, eds. Proc. of the 29th VLDB Conf. Berlin: IEEE Press, 2003. 93−104.

赵峰(1976－),男,湖北安陆人,博士,主要研究领域为机器学习,知识发现,网络安全.

金莉(1978－),女,博士生,主要研究领域为知识提取,信息安全.

李庆华(1940－),男,教授,博士生导师,主要研究领域为并行计算,智能软件,网络安全.