# 基于不变网络模型和故障注入的分布式信息系统故障溯源方法

贾庆轩[1†], 艾冠群[1], 高 欣[1], 李新鹏[1,2], 阎 博[3],
陈春旭[1], 李军良[4], 徐建航[4], 刘震宇[5], 庞 博[5]

(1. 北京邮电大学 自动化学院，北京 100876；2. 国家电网有限公司，北京 100031；
3. 国网冀北电力有限公司，北京 100054；4. 南瑞集团(国网电力科学研究院)有限公司，北京 100192；
5. 承德供电公司，河北 承德 067000)

**摘 要:** 针对传统分布式信息系统故障溯源算法对于先验知识依赖严重的问题,提出一种基于不变网络与故障注入相结合的故障溯源方法. 首先,利用系统日志中收集到的系统组件运行数据,构建系统的不变网络模型,在此基础上进行节点或组件故障注入及扩散建模,建立故障网络集;然后,根据原始时间序列取值情况,制定数据质量评价规则以甄别数据是否发生突变;最后,利用实际故障网络与故障网络集中故障网络局部拟合的方式进行故障溯源,并利用数据质量评价规则对该结果进行修正,实现对系统故障源的精确定位. 在仿真数据集、某开源系统数据集和某电网调度系统实采数据上的实验结果表明,所提出方法具有更高的准确率.

**关键词:** 信息系统；故障溯源；时间序列；不变网络；故障注入；局部拟合

**中图分类号:** TP307 **文献标志码:** A

**DOI:** 10.13195/j.kzyjc.2019.0214 **开放科学(资源服务)标识码(OSID):**

**引用格式:** 贾庆轩,艾冠群,高欣,等. 基于不变网络模型和故障注入的分布式信息系统故障溯源方法[J]. 控制与决策, 2020 , 35 (11 ): 2723-2732.

# Fault source location algorithm for distributed information system based on invariant network and fault injection

*JIA Qing-xuan*[1†], *AI Guan-qun*[1], *GAO Xin*[1], *LI Xin-peng*[1,2], *YAN Bo*[3], *CHEN Chun-xu*[1], *LI Jun-liang*[4], *XU Jian-hang*[4], *LIU Zhen-yu*[5], *PANG Bo*[5]

(1. School of Automation，Beijing University of Posts and Telecommunications，Beijing 100876，China；2. State Grid Corporation of China，Beijing 100031，China；3. State Grid Jibei Electric Power Co., Ltd.，Beijing 100054，China；4. NARI Group (State Grid Electric Power Research Institute) Co., Ltd.，Beijing 100192，China；5. Chengde Power Supply Company，Chengdu 067000，China)

**Abstract:** The normal fault source location algorithm depends too much on the prior knowledge of distributed information systems, therefore, this paper presents a fault source location algorithm based on the invariant network and fault injection. Firstly, the invariant network model of the system is established by using the running data of the system components collected in the system log and in order to obtain the fault network model and form a set, the fault injection and diffusion modeling of each component are carried out on the model. Then, according to the value of the original time series, the data quality evaluation rules are formulated to judge whether the data has changed violently. Finally, the source of the system fault is determined using the method of local fitting between the actual fault network and the centralized fault network, the data quality evaluation rules are used as result revise to realize the accurate location of system fault source. The results on the synthetic data set, the data set collected by an open source distributed information system and the actual data set of a power grid dispatching system show that the proposed method has higher accuracy.

**Keywords:** information systems；fault source location；time series；invariant network；fault injection；local fitting

## 0 引 言

分布式信息系统是指采用分布式软件系统构建的支持分布式运算的信息系统,通常由服务器、存储器、网络设备、操作系统、应用软件等成千上万个组件构成[1]. 一个组件故障可能造成极为严重的后果[2-3],而在系统中组件间的关联关系错综复杂,使得

**收稿日期:** 2019-02-27；**修回日期:** 2019-06-03.
**责任编委:** 高会军.
†通讯作者. E-mail: spacerobot@163.com.

故障溯源变得极为困难. 因此,及时发现并排除根源性故障具有实际意义和学术价值.

目前,常用的故障溯源方法主要有基于规则的故障溯源方法[4-8]和基于建模的故障溯源方法[9-13]. 基于规则的故障溯源方法利用故障与其预兆之间的联系生成一定的规则进行故障溯源;基于建模的方法通过建立系统的故障关联模型确定故障的根本原因. 上述方法虽然对信息物理系统故障溯源问题进行了多方面的讨论,但对逻辑拓扑结构、故障案例等先验知识依赖较大.

为解决上述问题,文献[14-16]提出并发展了基于ARX模型的不变网络建模方法,以数据驱动形式实现系统建模. 文献[17-21]基于不变网络提出了一系列故障溯源方法,但仍存在部分节点异常度漂移、靠近故障源的非故障节点异常度偏高、无向图模型对于故障的单向传播考虑不足等问题. 鉴于此,本文提出一种基于不变网络与故障注入相结合的故障溯源方法,创新和贡献如下:

1) 利用系统日志中收集到的数据信息建立系统的不变网络模型,并在该模型上进行各组件的故障注入及扩散建模,从而得到各组件为故障源的故障网络模型,并由此组成故障网络集,使得对于根源性故障部件的回溯有据可循;

2) 综合考虑异常度最高局部以及故障传播方向,通过故障网络局部拟合和突变点检测解决部分节点存在的异常度漂移问题,实现分布式信息系统故障源组件的准确回溯;

3) 充分利用数据采集部位所提供的先验信息,综合考虑时间序列级与组件级的故障溯源,设计相应规则,提高算法的灵活性和可操作性.

## 1 故障溯源方法

如图1所示,基于不变网络和故障注入相结合的故障溯源方法主要由制定故障溯源规则和故障溯源两部分构成.

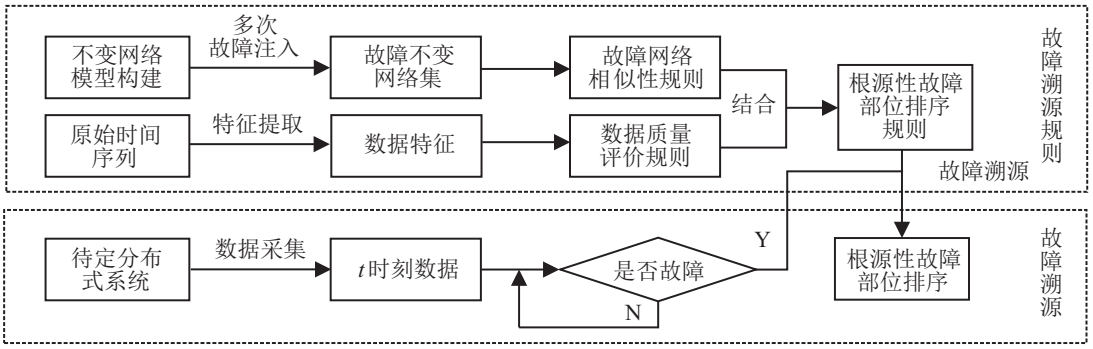



**图1 算法流程**

故障溯源规则的制定过程由两条主线构成:

1) 建立该系统的不变网络模型,对不同节点进行故障注入及扩散建模,并由此组成故障网络集,制定故障网络相似性规则;

2) 利用数据的幅值等特征制定数据质量评价规则,结合故障网络相似性规则与数据评价规则制定故障溯源规则.

利用由系统$t$时刻采集的数据,对系统故障与否进行判断,当系统发生故障时,利用故障溯源规则列出可能的故障源.

### 1.1 不变网络建模

计算机相关领域中,“网络”通常指系统中为各个组件提供信息交互的虚拟平台[22],本文所采用的不变网络实质是一种无向图模型. 图2详细描述了信息系统不变网络建模的过程,利用ARX模型[15]对运行数据中的时间序列进行两两拟合,建立系统的不变网络模型,其中时间序列被映射为不变网络的一个节点,组件被映射为一个节点组,这里的组件指的是系统中实现一定功能的单元,如进程、软件或相关硬件,后续内容将这些功能单元统称为组件.

文献[14]对不变网络的搭建过程进行了详细的描述,根据其提出的方法,假设有$A$、$B$两个流强度,根据ARX模型,它们的关系可以表示为

$$\begin{aligned} &y(t)=\varphi(t)^{\mathrm{T}}\theta=\\ &a_1y(t-1)+\ldots+a_ny(t-n)-\\ &[b_0x(t-k)+b_1x(t-k-1)+\ldots+\\ &b_mx(t-k-m)]. \end{aligned} \tag{1}$$

其中:$\theta=[a_1,a_2,\ldots,a_n,b_0,b_1,\ldots,b_m]^{\mathrm{T}}$,$\varphi(t)=[-y(t-1),\ldots,-y(t-n),x(t-k),\ldots,x(t-k-m)]^{\mathrm{T}}$,$x(t)$和$y(t)$分别为流强度$A$和流强度$B$在$t$时刻的数值,$m$、$n$分别为$t$时刻之前对流强度$B$在$t$时刻的数值产生影响的$A$、$B$两个时间序列的时刻数,$k$为流强度$A$相对于流强度$B$的延时,$a_i$、$b_i$为ARX模型系数.

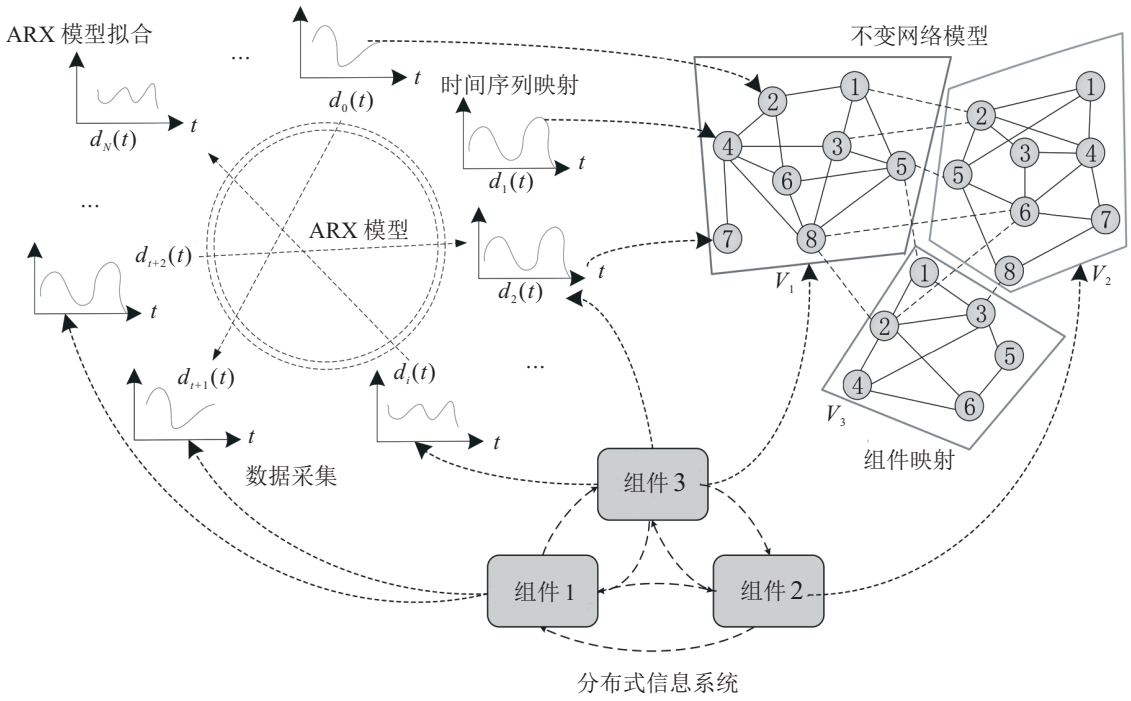



**图 2　不变网络建模示意图**

利用最小二乘法可以得到 $\hat{\theta}$, 利用拟合分数 $F\_\mathrm{net}(\theta)$ 表示模型预测值 $\hat{y}(t|\theta)$ 与实际值 $y(t)$ 之间的误差, 以判定该模型与实际情况的拟合程度, 其定义如下所示:

$$F\_\mathrm{net}(\theta)=\left[1-\sqrt{\frac{\sum_{t=1}^{N}|y(t)-\hat{y}(t|\theta)|^2}{\sum_{t=1}^{N}|y(t)-\bar{y}|^2}}\right]\times 100. \quad (2)$$

其中: $\bar{y}$ 为序列 $Y_N=\{y(1),y(2),\ldots,y(N)\}$ 的平均值; $\hat{y}(t\,|\,\theta)$ 为由 ARX 模型预测的 $t$ 时刻 $Y_N$ 序列的取值; $F\_\mathrm{net}(\theta)$ 表示模型的拟合程度, $F\_\mathrm{net}(\theta)\in[0,100]$, 值越大表示 ARX 模型与实际情况的拟合程度越好. 本文参照文献 [19-20] 中耦合分数阈值及 ARX 模型参数的设置, 假定 $F\_\mathrm{net}(\theta)\geqslant 70$ 的两个时间序列存在不变链接. 另外, 根据最大的网络规模确定 $m$、$n$、$k$ 的取值, 一般取 $0\leqslant m,n,k\leqslant 2$. 不变链接消失的检测方法[16] 为

$$R(t)=\frac{|y(t)-\hat{y}(t|\theta)|}{\varepsilon_{\max}}. \quad (3)$$

其中: $\varepsilon_{\max}=|y(t)-\hat{y}(t|\theta)|_{\max}$, $\varepsilon_{\max}$ 为模型搭建过程中预测值 $\hat{y}(t|\theta)$ 与实际值 $y(t)$ 最大误差. 当 $R(t)>1$ 时, 两个时间序列之间的不变链接消失.

### 1.2　故障注入及扩散建模

分布式信息系统中某个组件的故障会沿着耦合路径传播, 导致大量组件报警; 在不变网络中, 故障沿着不变链接传播, 造成大量节点的异常度升高. 本文采用半监督标签传播算法[23-24] 对系统进行故障注入和传播建模, 假设不变网络中共有 $M$ 个节点, $X=\{x_1,x_2,\ldots,x_M\}$.

节点异常程度 $F_i\in[0,1]$, 0 表示节点正常, 1 表示节点异常, $F_i$ 越接近于 1 表示节点异常程度越大.

故障扩散算法的具体实现步骤如下.

step 1: 根据不变网络中两个节点间不变链接的存在情况生成邻接矩阵 $W$, 有:

1) 当 $x_i$ 和 $x_j$ 间存在不变链接时, $W_{ij}=1$;

2) 当 $x_i$ 和 $x_j$ 间不存在不变链接时, $W_{ij}=0$;

3) 令 $W_{ii}=0$, 以避免标签在自身不断传播.

step 2: 建立矩阵 $W$ 的度标准化矩阵

$$\tilde{W}=D^{-\frac{1}{2}}WD^{-\frac{1}{2}}.$$

其中: $D$ 为对角矩阵, $D_{ii}=\sum_{j=1}^{N}W_{ij}$, 矩阵 $W$ 为对称矩阵. 本文认为故障会在具有不变链接的两个节点间相互传播符合实际情况.

step 3: 进行故障注入建模, 令 $F(t)$ 为时刻不变网络中各个节点的异常度, 令 0 时刻不变网络中节点的异常程度为

$$F(0)=\{x_a,x_n|x_a=1,a\in[1,l],$$
$$x_n=\mathrm{random}(0,0.4),b\in[l+1,N]\},$$

即令选定为异常的节点的异常度为 1, 非异常节点的异常程度在 [0, 0.4] 之间随机取值.

step 4: 进行故障扩散建模, 不变网络中节点初始

异常程度与最终异常程度的约束关系为

$$\min \alpha \sum_{i,j=1}^{n} W_{ij}\left\| \frac{1}{\sqrt{W_{ii}}}F(0)_i - \frac{1}{\sqrt{W_{jj}}}F(0)_j \right\|^2 + (1-\alpha)\sum_{i=1}^{n}\|F(t)-F(0)\|^2. \tag{4}$$

其中: $F(0)$ 为0时刻不变网络各个节点的异常程度, $F^*$ 为故障扩散完成后各个节点的异常程度.

step 5: 得到的极限为

$$F^* = \lim_{t\to\infty} F(t) = (1-\alpha)(I-\alpha W)^{-1}F(0), \tag{5}$$

其中 $F_i^*$ 为 $x_i$ 在故障扩散完成后的异常程度. 由于信息系统中故障扩散速度较快,其注入和扩散的仿真过程被认为是瞬间完成的.

### 1.3 局部异常程度的判定

不变网络由节点和不变链接构成[15],不变链接的消失情况标志着节点的异常程度[17]. 为综合考虑时间序列级与组件级故障溯源,使方法在应用过程中具有更高灵活性和可操作性,分别对节点和代表组件的节点簇的异常程度进行如下定义:

$$\mathrm{Dam}_{\mathrm{of\_}x_i} = \frac{\text{节点}x_i\text{消失的不变链接数}}{\text{节点}x_i\text{所有的不变链接数}}, \tag{6}$$

$$\mathrm{Dam}_{\mathrm{of\_}V_i} = \frac{\text{组件}V_i\text{消失的不变链接数}}{\text{组件}V_i\text{所有的不变链接数}}, \tag{7}$$

$$\mathrm{Dam}_{x_i} = \frac{\text{节点}x_i\text{相关节点消失不变链接数}}{\text{节点}x_i\text{相关节点所有不变链接数}}, \tag{8}$$

$$\mathrm{Dam}_{V_i} = \frac{\text{节点}V_i\text{相关节点消失不变链接数}}{\text{节点}V_i\text{相关节点所有不变链接数}}. \tag{9}$$

其中: $\mathrm{Dam}_{\mathrm{of\_}x_i}$ 为节点 $x_i$ 的异常程度, $\mathrm{Dam}_{\mathrm{of\_}V_i}$ 为组件 $V_i$ 的异常程度, $\mathrm{Dam}_{\mathrm{of\_}x_i}$ 和 $\mathrm{Dam}_{\mathrm{of\_}V_i}$ 均在 $[0,1]$ 之间取值,越接近1表示该组件的异常程度越高; $\mathrm{Dam}_{x_i}$ 表示以节点 $x_i$ 为中心的不变网络的局部异常程度, $\mathrm{Dam}_{V_i}$ 表示以 $V_i$ 为中心的不变网络的局部异常程度, $\mathrm{Dam}_{x_i}$ 和 $\mathrm{Dam}_{V_i}$ 均在 $[0,1]$ 之间取值,越接近1表示该组件所在局部的异常程度越高.

### 1.4 数据质量评价规则

在不变网络的建模过程中,考虑了两个时间序列存在长期联系的情况[25],但对时间序列的非规则性变化考虑不足. 某一条不变链接消失,其两端节点的异常程度都会上升,无法判断是哪一个时间序列的突变引起了不变链接的中断. 本文对发生突变的时间序列进行甄别,以进一步确定故障源:

1) 时间序列中出现向上、下突变的异常值,定义该异常值为

$$x_i > Q_3 + 1.5\,\mathrm{IQR}, \tag{10}$$

$$x_i < Q_1 - 1.5\,\mathrm{IQR}. \tag{11}$$

其中: $Q_1$、$Q_3$ 分别为时间序列取值的第一、三分位数, IQR为四分位距[24].

2) 当时间序列出现连续为零的情况时,有

$$\mathrm{len}\,0 > \mathrm{len}\,0_{\max}. \tag{12}$$

其中: $\mathrm{len}\,0$ 为当前时刻该时间序列出现连续零值的个数, $\mathrm{len}\,0_{\max}$ 为训练数据中该时间序列出现的最大连续零值的长度.

### 1.5 故障网络的局部拟合

由第1.3节可知,节点或组件的特征提取由周围节点及节点自身不变链接消失的情况确定,根据故障溯源精度不同其局部定义也不同: 当故障溯源精度为时间序列级时,局部指不变网络中该时间序列所对应节点以及与其存在不变链接的各个节点; 当故障溯源精度为组件时,局部指不变网络中该组件所对应节点簇以及与其存在不变链接的节点簇所组成的节点集.

由式(8)和(9)得到局部故障程度,从而得到局部异常程度排序. 当局部以节点为中心时,按照节点的程度排名逐一增加,与故障网络集中故障网络相应节点的异常度进行拟合,按照拟合的效果进行排序,得到时间序列级故障溯源结果; 当局部以组件为中心时,按照组件异常程度排名逐一增加,将组件的异常度与故障网络集中故障网络相应组件的异常度进行拟合,得到组件级故障溯源结果. 采用欧氏距离和夹角余弦表示两个网络的拟合程度,有

$$D(N,\widehat{N}) = \sqrt{\sum_{i=1}^{n}\left(\mathrm{Dam}_{x_i} - \widehat{\mathrm{Dam}}_{\mathrm{of\_}x_i}\right)^2}, \quad x_i \in V_D; \tag{13}$$

$$\cos(N,\widehat{N}) = \frac{\sum_{i=1}^{n}\mathrm{Dam}_{x_i}\cdot\widehat{\mathrm{Dam}}_{\mathrm{of\_}x_i}}{\sqrt{\sum_{i=1}^{n}\mathrm{Dam}_{x_i}^2}\cdot\sqrt{\sum_{i=1}^{n}\widehat{\mathrm{Dam}}_{\mathrm{of\_}x_i}^2}}, \quad x_i \in V_D. \tag{14}$$

其中: $V_D$ 为以局部异常程度最高的 $n$ 个节点或者局部异常程度最高的 $m$ 个组件所包含的所有节点; $N$ 和 $\widehat{N}$ 分别为实际情况和完成故障注入及扩散建模后故障不变网络中 $V_D$ 包含各个节点的异常程度; $\mathrm{of\_}x_i$ 表示对 $x_i$ 节点进行故障注入; $D(N,\widehat{N})$ 和 $\cos(N,\widehat{N})$ 分别为两个向量的欧氏距离和余弦距离,即故障局部中各点故障注入模拟出的异常程度与实际异常程度之间的拟合程度. 利用 $\mathrm{fitscore}_{x_i}$ 表示该局部的实际异常程度与对节点 $x_i$ 进行故障注入及扩散建模后得到的故障网络中对应各个节点异常程度的拟合分数,

有

$$\text{fitscore}_{x_i} = [1 - \text{Norm}[\cos(N, \widehat{N}_{x_i})]] + \text{Norm}[D(N, \widehat{N}_{x_i})],\ x_i \in V_D. \quad (15)$$

其中: $\text{Norm}(\cdot)$ 为归一化函数, 表示对所有 $D(N, \widehat{N})$ 和 $\cos(N, \widehat{N})$ 分别进行归一化; $N$ 为 $V_D$ 中各故障节点的异常程度; $\widehat{N}_{x_i}$ 为对不变网络中节点 $x_i$ 进行故障注入及扩散建模后得到的故障不变网络中 $V_D$ 所包含各个节点的异常程度. $\text{fitscore}_{x_i}$ 越低, 拟合程度越好, $x_i$ 是根源性故障时间序列的可能性越大. 同理有

$$\text{fitscore}_{V_i} = [1 - \text{Norm}[\cos(N, \widehat{N}_{V_i})]] + \text{Norm}[D(N, \widehat{N}_{V_i})]. \quad (16)$$

其中: $N$ 为 $V_D$ 中各个故障组件的异常程度; $\widehat{N}_{V_i}$ 为对不变网络中组件 $V_i$ 进行故障注入及扩散建模后得到的故障不变网络中 $V_D$ 所包含各个组件的异常程度; $\text{fitscore}_{V_i}$ 为该局部的实际异常程度与对组件 $V_i$ 进行故障注入及扩散建模后得到的故障网络中对应各个组件异常程度的拟合分数, 该分数越低, 拟合程度越高, $V_i$ 是故障源的可能性越大.

### 1.6　局部异常程度的判定

采用不变网络局部拟合与突变点检测相结合的方法, 消除异常度漂移、故障传播方向考虑不足等问题带来的误差. 由第 1.2 节得到故障不变网络中各个节点或组件的异常度, 由第 1.3 节得到故障发生时不变网络模型中各个节点或组件的局部异常度. 首先, 将第 1.3 节中得到的局部异常度对节点或组件进行从大到小排序, 按照该顺序利用第 1.4 节所提出的拟合分数对故障网络与故障网络集中故障网络相应各点的异常度进行拟合, 按照拟合分数越低排名越靠前的原则, 得到大致的根源性故障部位的可能性排序; 然后, 对时间序列的质量进行判定, 若故障溯源级别为时间序列, 则利用时间序列的质量判定结果对排序结果进行修正, 将出现异常的时间序列排名按照原有顺序提到最前面, 若故障溯源的级别为组件, 则将组件时间序列异常比例超过 30 % 的组件按照原有顺序提到最前; 最后, 得到最终的排序结果.

## 2　实验分析

### 2.1　实验数据

实验数据主要包含电网调度系统 SCADA 应用数据集、某开源信息系统数据集和按一定规则生成的时间序列数据集[14], 电网调度系统 SCADA 应用数据集由 D5000 系统 SCADA 应用各进程中采集到的时间序列组成, 共包含 7 420 个时间序列, 每个时间序列包含 15 天的数据, 每 30 s 采集一次; 开源信息系统所采集到的数据集共包含 718 个时间序列, 每个时间序列包含 90 min 的数据, 每秒采集一次; 仿真数据为按照一定规则生成的数据, 共有 1 300 个时间序列, 每个时间序列包含 200 个点.

### 2.2　故障注入

#### 2.2.1　仿真数据

对于仿真数据的故障注入, 通过改变几个时间序列的数值来实现, 使其某点数据满足

$$R(t) > 1. \quad (17)$$

将时间序列的值人为改变, 使之最终满足式 (12) 和如下两式, 从而实现故障注入:

$$y(t) > \varepsilon_{\max} + \hat{y}(t \mid \theta), \quad (18)$$

$$y(t) < \hat{y}(t \mid \theta) - \varepsilon_{\max}. \quad (19)$$

#### 2.2.2　开源信息系统

对真实系统注入的故障包括网络故障、进程故障和系统故障.

1) 网络故障分为网络延时和网络中断. 网络延时可以通过 Linux 中的 TC 工具完成[26], 利用相关命令实现网络延时由 23 ms 逐渐增加到 3 s; 网络中断采用 service network-manager stop 命令关闭系统中某节点的网络服务来实现.

2) 采用 kill 命令杀死某些关键进程以达到进程闪退故障注入的目的.

3) 对于系统故障, 采用 shutdown-h now 命令关闭某节点操作系统实现宕机故障的注入.

### 2.3　排序验证基准

由于文中数据分为仿真数据和实际数据, 对于故障溯源结果的验证标准是不同的, 根据实际情况制定如下基准:

1) D5000 系统数据集.

D5000 系统作为耦合复杂的信息系统, 往往一个组件发生故障时会有多个组件同时报警, 系统专家通常不去寻找故障根源, 而是采用重启等方式排除故障, 因此要求他们给出故障发生及扩散的路径是几乎不可能的[17,27]. 本文采用时间序列的变化率作为基准, 在流强度时间序列数据中, 振幅的变化是表示时间序列异常程度的重要标志[28], 按照时间序列变化率由高到低的顺序对时间序列进行排序, 并以此作为验证各个算法的基准. $t$ 时刻时间序列由 $y(t-1)$ 到 $y(t)$ 的变化率计算为

$$r(t) = \frac{|y(t) - y(t-1)|}{y(t-1)}. \quad (20)$$

D5000 系统采集时间序列的间隔为 30 s, 这段时

间对于故障的发生和扩散而言是足够的,因此可以认为D5000系统中的故障发生及扩散是瞬间完成的. 当系统发生故障时,相关时间序列的振幅一定会有所变化,因此利用两点之间振幅的变化率作为基准是符合实际情况的,而组件级的基准是该基准前$K$个时间序列中各组件所占比例的排序.

2) 开源信息系统数据集.

开源信息系统结构较为简单,容易得到系统中故障的传播情况,为确定基准提供了依据,即进行故障注入的组件是根源性故障部位.

### 2.4 实验结果

本文参考文献[19]在进行算法评价时所采用的实验过程和评价指标,采用召回率、准确率、误警率和增益值(nDCG)对算法进行评价. 仿真数据集共有1 300个时间序列,38 277条不变链接,随机对30个时间序列进行故障注入,故障溯源结果如图3所示.

由图3可见,所提出方法的前50个排名在召回率、准确率和误警率等指标上均略优于其他算法,仅当$K=2$时系统检测到异常情况,体现了数据质量检测规则的有效性.

开源信息系统的不变网络包含41 657条不变链接,对开源信息系统进行4种故障注入,分别为:宕机、关键进程意外中断、网络延时和网络中断. 时间序列级故障溯源结果如图4和图5所示.

由图4可见:对于网络延时这类故障,所提出方法的前20个排名在各个指标上均略优于其他算法;准确率、误警率曲线相互印证,仅当$K=1$时系统命中相关时间序列. 由图5可见,对于网络中断这类

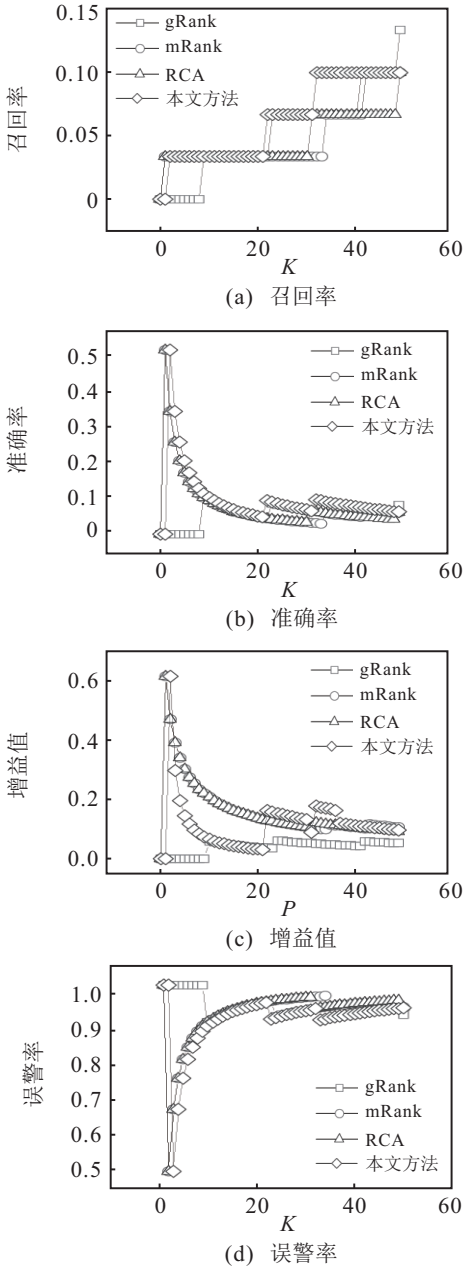



图3 仿真时间序列故障溯源结果

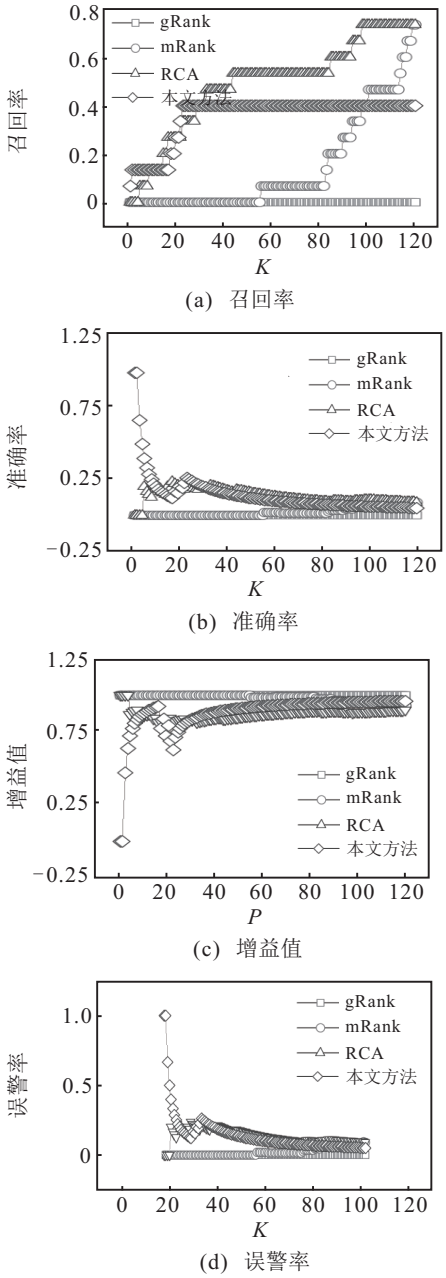
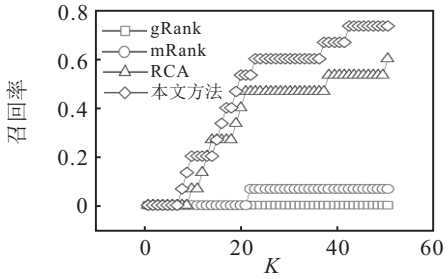
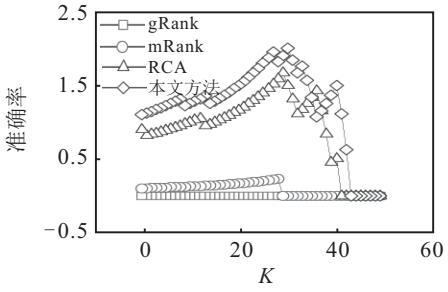


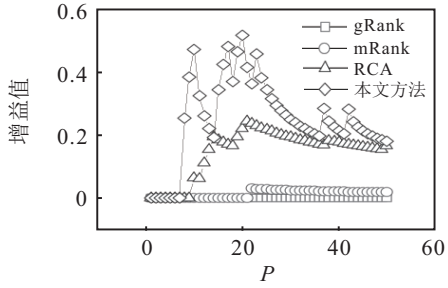

图4 网络延时时间序列级故障溯源结果



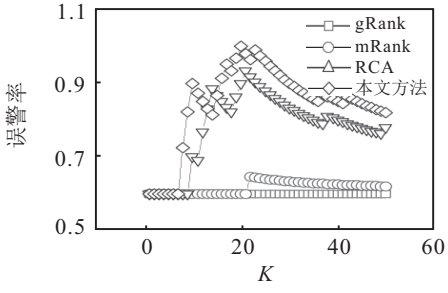

(a) 召回率




(b) 准确率

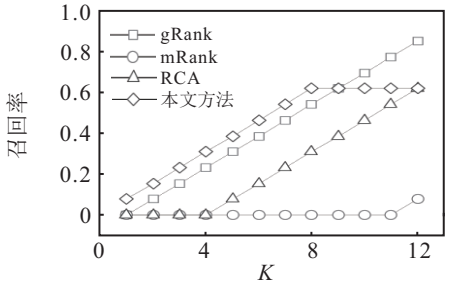



(c) 增益值

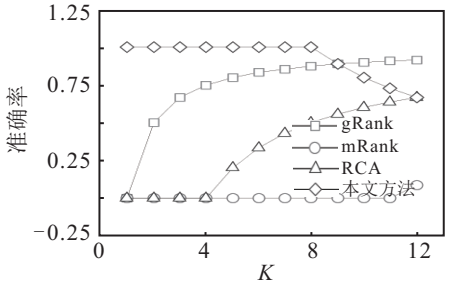



(d) 误警率

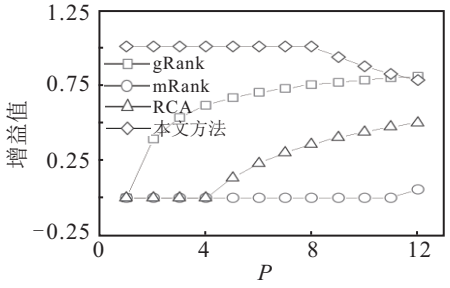
**图 5　网络中断时间序列级故障溯源结果**



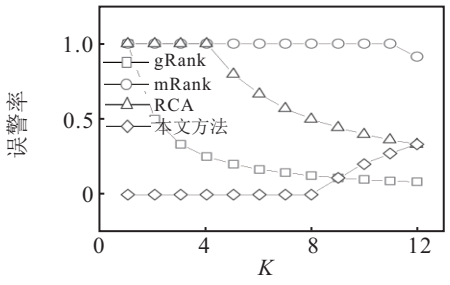

(a) 召回率




(b) 准确率




(c) 增益值




(d) 误警值

**图 6　宕机组件级故障溯源结果**

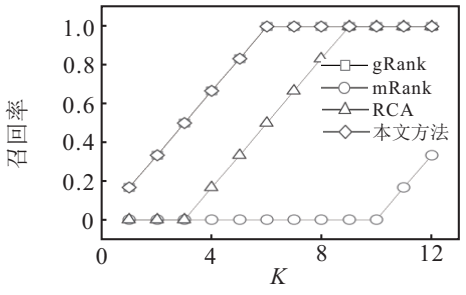
故障,所提出方法在召回率、准确率和误警率等指标上均略优于其他算法.

本文以宕机和关键进程闪退为例对组件级故障进行定位,实验结果如图6和图7所示.

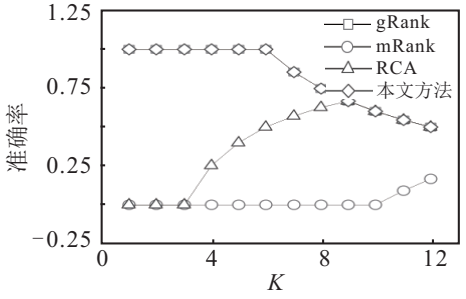
由图6可见,所提出方法的前8位结果在召回率、准确率等指标上均略优于其他算法,在 $K = 1$ 时,所提出方法命中相关组件,且该方法前8个怀疑对象全部命中. 由图7可见,所提出方法表现最好,在宕机、进程闪退这类故障溯源问题上均有较好表现.

以网络中断为例,详细描述组件级故障溯源结果. 当开源信息系统某一节点由于某种原因出现网络故障时,与其存在信息交互的计算机与该计算机的网络连接也会中断,这种故障会最直接反映到网络延时相关时间序列上,在基准中,表示网络延时的组件排名会靠前. 系统中第63、64、65、66号组件分别表示1、2、3号从机和主机的网络模块,从机3网卡失效,其他几台计算机网络相关的时间序列也会受到影响.

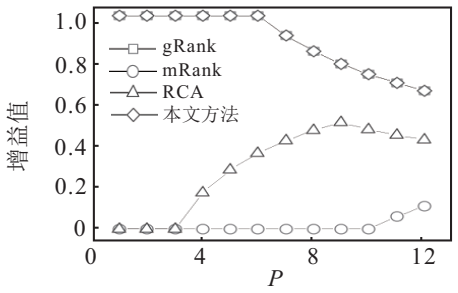
如表1所示,所提出方法第1个怀疑对象即命中故障源,且在第2、3位分别命中受影响组件,可见所提出方法在该故障上具有较好的效果. 电力调度系统数据集共包含59台计算机的数据,742个组件,收集到7 420个时间序列,利用该数据构建的不变网络包含242 341条不变链接. 对于某时刻出现的66个报警信息进行故障溯源,结果如图8和图9所示.



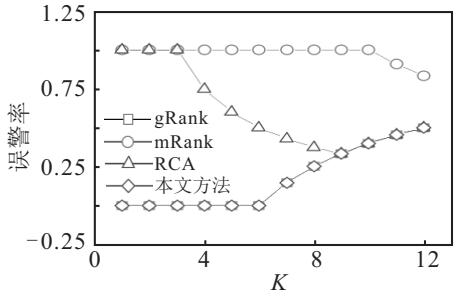

(a) 召回率




(b) 准确率

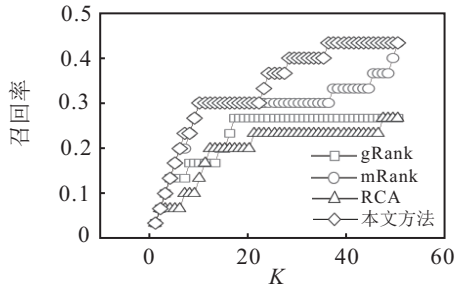



(c) 增益值

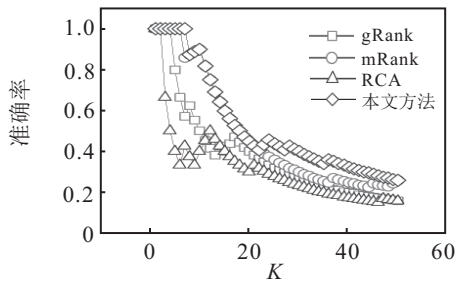



(d) 误警值

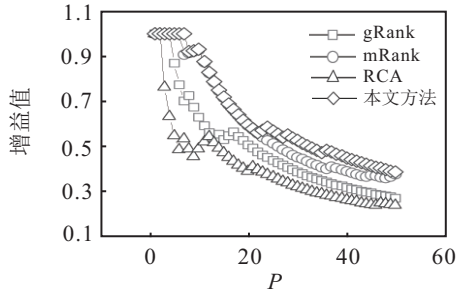
**图 7 关键进程闪退进程级故障溯源结果**



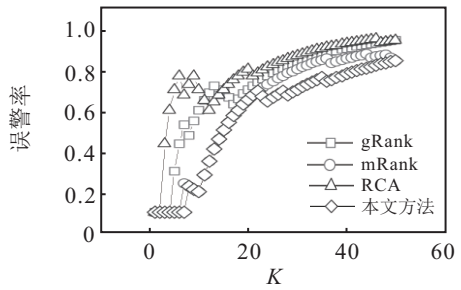

(a) 召回率




(b) 准确率




(c) 增益值



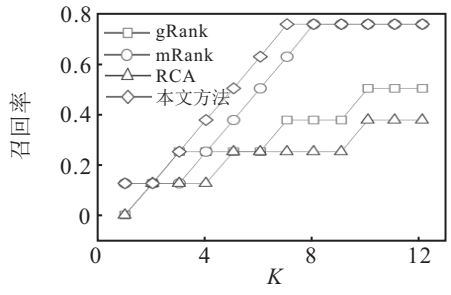

(d) 误警值

**图 8 D5000系统报警信息时间序列级故障溯源结果**

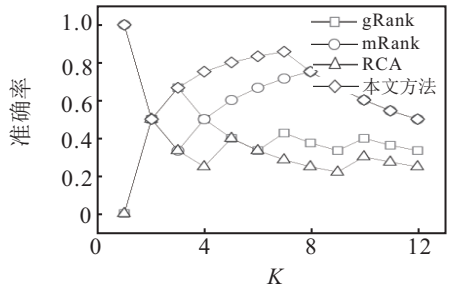
**表 1 网络中断组件级故障溯源结果**

| | mRank | gRank | RCA | 本文方法 |
|---|---|---|---|---|
| 1 | 43 | 55 | 27 | 65 |
| 2 | 46 | 56 | 64 | 63 |
| 3 | 66 | 39 | 63 | 64 |
| 4 | 32 | 62 | 65 | 20 |
| 5 | 64 | 57 | 20 | 6 |
| 6 | 30 | 38 | 35 | 27 |
| 7 | 12 | 37 | 6 | 7 |
| 8 | 6 | 40 | 7 | 3 |
| 9 | 42 | 17 | 4 | 5 |
| 10 | 44 | 61 | 14 | 14 |
| 11 | 63 | 64 | 21 | 4 |
| 12 | 65 | 63 | 5 | 21 |

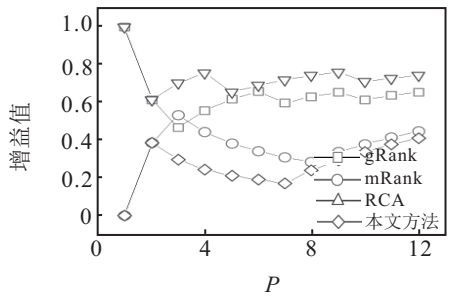
由图8可见,所提出方法在召回率、 准确率、nDCG等指标上均略优于其他算法,准确率与误警率相互印证,且在$K = 1$时怀疑对象命中故障源,体现了所提出方法的有效性. 以时间序列变化率排序前$N$个组件所占的比例进行排序,得到变化最大的8个组件,给出各算法在组件级故障溯源结果中排序的前12位如图9所示. 对于大面积报警的组件级故障溯源问题,所提出方法在召回率、准确率、nDCG等指标上均略优于其他算法,召回率和准确率曲线可以相互印证,当$K \leqslant 7$时,除第2个点外,其余点均命中,展现了所提出方法故障溯源排序中靠前部分性能突出.



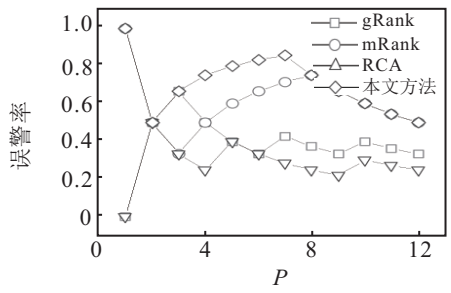

(a) 召回率




(b) 准确率




(c) 增益值




(d) 误警值

**图 9　D5000 系统报警信息组件级故障溯源结果**

## 3　结　论

本文针对分布式信息系统组件间耦合关系复杂、逻辑拓扑图不易描述等问题,提出了一种基于不变网络与故障注入相结合的故障溯源方法,利用不变网络局部拟合以及突变点检测等方法,解决了现有基于不变网络故障溯源算法中网络边界点异常度漂移、故障传播方向考虑不足等问题,有效解决了逻辑拓扑结构不明确条件下分布式信息系统故障溯源问题. 通过数学推导和实验两部分对所提出算法的先进性和可行性进行验证,充分证明了所提出方法可以实现有效的故障溯源,能够为运维人员查找故障源头提供可靠的参考和指导. 另外,所提出方法也可应用于在时间序列间可进行流动力学建模的其他信息物理系统,但在提取不变网络节点特征时将所有节点看作具有相同地位,并未对网络中节点的密度、节点重要度等进行分析. 结合节点密度、单个节点不变链接数对节点的重要程度进行区分以及大数据条件下结合故障频率对易故障节点进行建模分析将是未来研究的方向.

## 参考文献(References)


[1] Mehdi S, Jorge P, Michel L. Information system architectures: Where we are?[C]. International Conference on Information & Communication Technologies: From Theory to Applications. Damascus: IEEE, 2004: 509-510.

[2] Alireza Fazlirad, Robert W Brennan. Multiagent manufacturing scheduling: An updated state of the art review[C]. The 14th International Conference on Automation Science and Engineering (CASE). Munich: IEEE, 2018: 722-729.

[3] Dragan Djurdjanovic, Jay Lee, Jun Ni. Watchdog agent — An infotronics-based prognostics approach for product performance degradation assessment and prediction[J]. Advanced Engineering Informatics, 2003, 17(3/4): 109-125.

[4] Damri G, Pant G, Jain A K. Correlating multiple events and data in an ethernet network [C]. The 7th International Conference on Communication Systems and Network Technologies (CSNT). Nagpur: IEEE, 2017: 56-61.

[5] Liang X, Wallace S A, Nguyen D. Rule-based data-driven analytics for Wide-Area fault detection using synchrophasor data[J]. IEEE Transactions on Industry Applications, 2016, 53(3): 1789-1798.

[6] Zhao Y, Zhang P, Jin Y. Netography: Troubleshoot your network with packet behavior in SDN[C]. Network Operations & Management Symposium. Istanbul: IEEE, 2016: 878-882.

[7] 江雪晨, 王大志, 宁一, 等. 基于关联规则的电网故障诊断解析方法[J]. 控制与决策, 2016, 31(6): 1138-1142.
(Jiang X C, Wang D Z, Ning Y, et al. Analytic method for fault diagnosis of power systems based on association rules[J]. Control and Decision, 2016, 31(6): 1138-1142.)

[8] 陈墨, 金磊, 龚向阳, 等. 面向5G海量网管数据的故障溯源技术[J]. 北京邮电大学学报, 2018, 41(5): 131-136.
(Chen M, Jin L, Gong X Y, et al. Research on fault tracing technology for 5G mass network management data[J]. Journal of Beijing University of Posts and Telecommunications, 2018, 41(5): 131-136.)

[9] Eberle W, Holder L, Cook D. Identifying threats using graph-based anomaly detection[M]. Boston: Springer, 2009: 73-108.

[10] Jia T, Chen P, Yang L, et al. An approach for anomaly diagnosis based on hybrid graph model with logs for distributed services[C]. 2017 IEEE International

Conference on Web Services (ICWS). Hondulu: IEEE, 2017: 25-32.

[11] Zasadzinski M, Muntes-Mulero V, Simo M S. Actor based root cause analysis in a distributed environment[C]. IEEE ACM International Workshop on Software Engineering for Smart Cyber-physical Systems. Buenos Aires: IEEE, 2017: 14-17.

[12] 王梦园, 张雄, 马亮, 等. 基于因果拓扑图的工业过程故障诊断[J]. 山东大学学报: 工学版, 2017, 47(5): 192-199.
(Wang M Y, Zhang X, Ma L, et al. Fault diagnosis for industrial processes based on causal topological graph[J]. Journal of Shandong University: Engineering Science, 2017, 47(5): 192-199.)

[13] 李振兴, 孟晓星, 李振华, 等. 应用等效网络原理的新型配电网故障定位技术[J]. 电力系统及其自动化学报, 2019, 31(1): 31-39.
(Li Z X, Meng X X, Li Z H, et al. Novel fault location technology for distribution network based on equivalent network Principle[J]. Journal of Power System and Automation, 2019, 31(1): 31-39.)

[14] Jiang G F, Chen H F, Yoshihira K. Discovering likely invariants of distributed transaction systems for autonomic system management[J]. Cluster Computing, 2006, 9(4): 385-399.

[15] Jiang G F, Chen H F, Yoshihira K. Modeling and tracking of transaction flow dynamics for fault detection in complex systems[J]. IEEE Transactions on Dependable and Secure Computing, 2006, 3(4): 312-326.

[16] Sharma A B, Chen H, Ding M, et al. Fault detection and localization in distributed systems using invariant relationships[C]. IEEE/IFIP International Conference on Dependable Systems & Networks. Budapest: IEEE, 2013: 1-8.

[17] Ge Y, Jiang G F, Ding M, et al. Ranking metric anomaly in invariant networks[J]. Acm Transactions on Knowledge Discovery from Data, 2014, 8(2): 1-30.

[18] Tao C, Ge Y, Song Q, et al. Metric ranking of invariant networks with belief propagation[C]. 2014 IEEE International Conference on Data Mining (ICDM). Shenzhen: IEEE Computer Society, 2014: 1001-1006.

[19] Cheng W, Zhang K, Chen H, et al. Ranking causal anomalies via temporal and dynamical analysis on vanishing correlations[C]. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2016: 805-814.

[20] Cheng W, Zhang K, Chen H, et al. Ranking causal anomalies via temporal and dynamical analysis on vanishing correlations[P]. U.S.: 15/420,949. 2017-08-10.

[21] Luo C, Chen Z Z, Tang L A, et al. TINET: Learning invariant networks via knowledge transfer[C]. Proceedings of the 24th Acm SIGKDD International Conference on Knowledge Discovery & Data Mining. New York: ACM, 2018: 1890-1899.

[22] 谢希仁. 计算机网络[M]. 北京: 电子工业出版社, 2008: 1890-1899.
(Xie X R. Computer network[M]. Beijing: Electronic Industry Press, 2008: 1890-1899.)

[23] Zhou D, Bousquet O, Lal T N, et al. Learning with local and global consistency[C]. Advances in Neural Information Processing Systems. Cambridge: ACM, 2004: 321-328.

[24] Wang H, Wang S B, Li Y F. Instance selection method for improving graph-based semi-supervised learning[J]. Frontiers of Computer Science, 2018, 12(4): 725-735.

[25] Xie K, Li X C, Wang X, et al. On-line anomaly detection with high accuracy[J]. ACM Transactions on Networking, 2018: 26(3): 1222-1235.

[26] Cannon J. Command line kung fu: Bash scripting tricks, linux shell programming tips, and bash one-liners[M]. Charleston: Create Space Independent Publishing Platform, 2014: 163-165.

[27] Ghanbari S, Amza C. Semantic-driven model composition for accurate anomaly diagnosis[C]. International Conference on Autonomic Computing. Chicago: IEEE, 2008: 35-44.

[28] Lakshminarayan C, Alvarado A S, Principe J C, et al. Anomaly detection in streaming data[P]. U.S.: 9,218,527. 2015-12-22.

## 作者简介

贾庆轩 (1964−), 男, 教授, 博士生导师, 从事空间机器人、人工智能等研究, E-mail: spacerobot@163.com;

艾冠群 (1995−), 男, 硕士生, 从事电力系统自动化、数据挖掘的研究, E-mail: andygqun@gmail.com;

高欣 (1974−), 男, 副教授, 博士, 从事电力系统自动化、数据挖掘等研究, E-mail: xlhhh74@bupt.edu.cn;

李新鹏 (1989−), 男, 高级工程师, 博士生, 从事电力系统自动化的研究, E-mail: xinpengli@126.com;

阎博 (1985−), 男, 高级工程师, 博士, 从事电力系统自动化等研究, E-mail:yan.bo.c@jibei. sgcc.com.cn;

陈春旭 (1996−), 男, 硕士生, 从事电力系统自动化、数据挖掘的研究, E-mail: xiaoguangchenmeng@163.com;

李军良 (1981−), 男, 高级工程师, 硕士, 从事电力系统自动化等研究, E-mail: lijunliangcn@163.com;

徐建航 (1988−), 男, 工程师, 从事电力系统自动化等研究, E-mail: 18210039762@126.com;

刘震宇 (1976−), 男, 硕士, 从事电力系统自动化等研究, E-mail: Liuzhenyu2000@sina.com.cn;

庞博 (1975−), 男, 硕士, 从事电力系统自动化等研究, E-mail: lli517@163.com.

(责任编辑: 郑晓蕾)