基于生成式超图聚类的网络告警日志归并框架

任泽华
(西安交通大学系统工程研究所 西安 710049)

摘 要 面对日益严峻的网络攻击，入侵检测系统（IDS）发挥着越来越重要的作用。由于真实网络环境中告警数量巨大、业务场景复杂、攻击方式多元，很难仅靠人工排查的方式进行处置归并。然而，大多数传统IDS告警关联方案仅针对单一攻击场景且依赖专家经验，大多数时候可解释性不强。因此本文提出了一种基于生成式超图聚类的网络告警归并框架：GHCAM，通过将多条告警按一定规则建模为超图，保留了日志之间的高阶关联并且具有高度的可扩展性。使用生成式聚类方法，使得归并结果能够从概率角度得到解释。此框架提供可修改的分辨率参数以生成不同层次的聚类结果，比传统方法更加灵活。为衡量此方法的有效性，我们定义了特定应用场景下不同维度的聚类评估指标。实验表明，此框架相较传统图相关方法速度提升了近一倍，将原始告警数量降低了近2个数量级，面对告警数量爆炸的突发情况也具有较好的鲁棒性。

关键词 入侵检测；告警关联；数据挖掘；超图聚类；社区发现；生成式模型
中图法分类号 TP18 DOI号: *投稿时不提供DOI号

Network Alarm Merge Framework Based on Generative Hypergraph Clustering

REN Ze-Hua
(Institute of Systems Engineering, Xi'an Jiaotong University, Xi'an 710049)

Abstract In the face of increasingly severe network attacks, intrusion detection systems (IDS) play an increasingly important role. Due to the huge number of alarms, complex business scenarios, and diverse attack methods in the real network environment, it is difficult to handle and merge them manually. However, most traditional IDS alarm correlation methods only target a single attack scenario, rely on expert experience and are not interpretable. This paper proposes a network alarm merging framework based on generative hypergraph clustering: GHCAM. By modeling multiple alarms as hypergraphs according to certain rules, the high-order correlation between logs is preserved and is scalable. Using generative clustering methods, the pooled results are explained probabilistically.This framework provides modifiable resolution parameters to generate different levels of clustering results, which is more flexible than traditional methods. To measure the effectiveness of this method, we define clustering evaluation metrics for different dimensions in this scenario. Experiments show that this framework doubles the speed of traditional graph correlation methods, reduces the number of original alarms by 2 orders of magnitude, and has good robustness in the face of emergencies when the number of alarms explodes.

Keywords intrusion detection; alert correlation; data mining; hypergraph clustering; community discovery; generative model

收稿日期：2022-08-22；最终修改稿收到日期：2022-08-23.
任泽华，硕士研究生，主要研究领域为入侵检测、电网安全、高级图论. E-mail: renzehua@stu.xjtu.edu.cn.

1 引 言

随着支持接入互联网设备数量的增长，使得利用大量主机进行协同攻击成为可能。由于大部分攻击行为涉及多个设备，面对同一安全事件，IDS系统会产生海量告警。安全运维人员被越来越多的告警所淹没，从而难以准确定位问题、把真实攻击行为和低危误报进行区分[1]。

利用告警日志的时空关联性进行归并的方法统称为告警聚类算法（Alarm clustering algorithms），对告警进行聚类的动机来源于相同根本原因——即相同安全事件产生的系列告警通常是“相似的”，90% 的警报可以归因于少数根本原因[2]。传统的告警聚类算法包括K. Julisch等[2]基于告警属性信息的层次聚类算法；F. Cuppens等[3]使用逻辑描述语言对告警进行了关联聚合；K. Julisch等[4]进一步提出了广义告警的概念，这是本文中安全事件的雏形；G. Giacinto等[5]引入“元告警”(Meta Alarm)的概念：由于某种攻击而产生的基本告警，将它们融合产生更高级别的告警消息，称为元告警，进一步丰富和完善了安全事件的内涵；近些年此领域仍有进展，M. Raman等[6]使用遗传算法结合支持向量机的方式对告警进行建模和归并。传统聚类框架大多针对单一攻击场景（如僵尸网络、DDos等），在面对海量异构告警数据时往往力不从心，无法提供一种可扩展的聚类方案。

近些年兴起了一系列以图论为基础的告警聚类框架，其中最具代表性的是S. Haas等提出的GAC框架[1]，通过将海量日志建模为告警相似度图，描述各个告警之间的相似相关程度。进而在相似度图上运行以CPM[7]为基础的网络社区发现算法，找到相应的告警集群。然后再建立告警流图，反映告警涉及网络设备间的拓扑关系，在此基础上修正告警集群划分。最后依据生成的告警集群作为发现的安全事件进行下一步关联分析。由于第一步构建的告警相似图连接非常紧密，一旦告警数量增大，对于传统图聚类算法来说运行时间都是无法接受的，而且此方法无法对聚类结果进行解释。

为此，我们参考S. Philip [8]等人的最新工作，提出了基于生成式超图聚类的网络告警日志归并框架：GHCAM（Generative Hypergraph Clustering Alarm Merging）。超图是一般图的泛化，超图里的边可以连接任意数量的顶点，是描述多个节点之间相互作用的网络结构。我们使用超图对海量日志进行建模，描述了告警之间的高阶关联结构，使用不同的超边构建准则可以按照实际工程需要对算法进行扩展。基于生成式的聚类算法对当前构建的告警态势超图进行建模，以最大似然的方式对超图结构进行估计，使得聚类结果具有概率意义上的可解释性。同时此方法支持不同分辨率[9]的初始化参数，可以按照自己的需要进行分辨率选择，避免了传统方法分辨率的不可预测性[10]。

本文其余部分结构如下：第二节描述我们的关联超图模型和两种超图扩展方案、第三节介绍问题的定义和评价指标、第四节总结传统的社区检测算法并介绍我们的超图算法框架、第五节展示实验结果、第六节总结了本文并提出了未来的工作方向。

2 关联建模

2.1 告警超图模型

我们通过将相似或相关的告警之间连接一条超边来建立超图模型，图1给出了告警超图建模的一种方式：将同属于一个源设备的告警或同属于一个目的设备的告警划入一条超边。左图是上文提到的告警拓扑流图，节点表示网络设备、边表示网络告警；而右图中网络告警用节点表示，原始拓扑图中同源或同目的的告警用圆圈圈出，代表这些节点间存在一条超边。这种方式是基于告警流图拓扑特征的推广。同时，我们也可以定义其他不同的规则：如同厂商同属性同威胁等级的告警划入一条超边。由于超边的定义宽松，我们可以根据真实工程需求

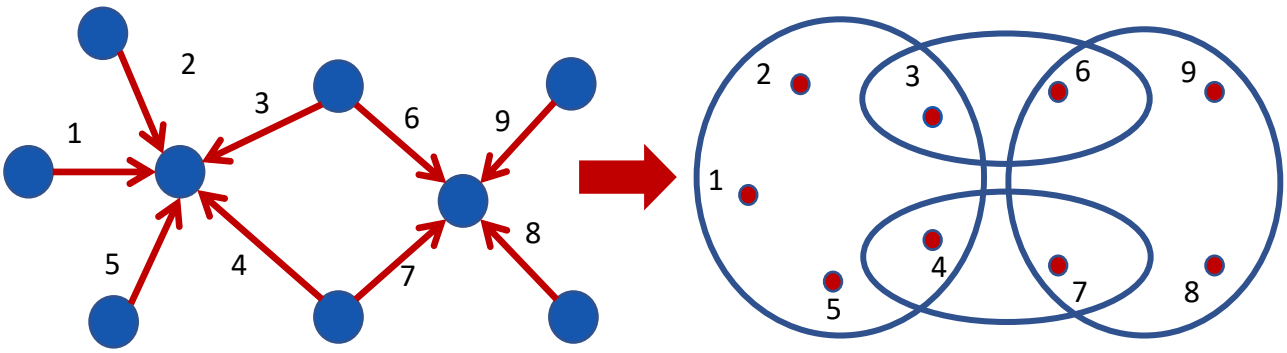
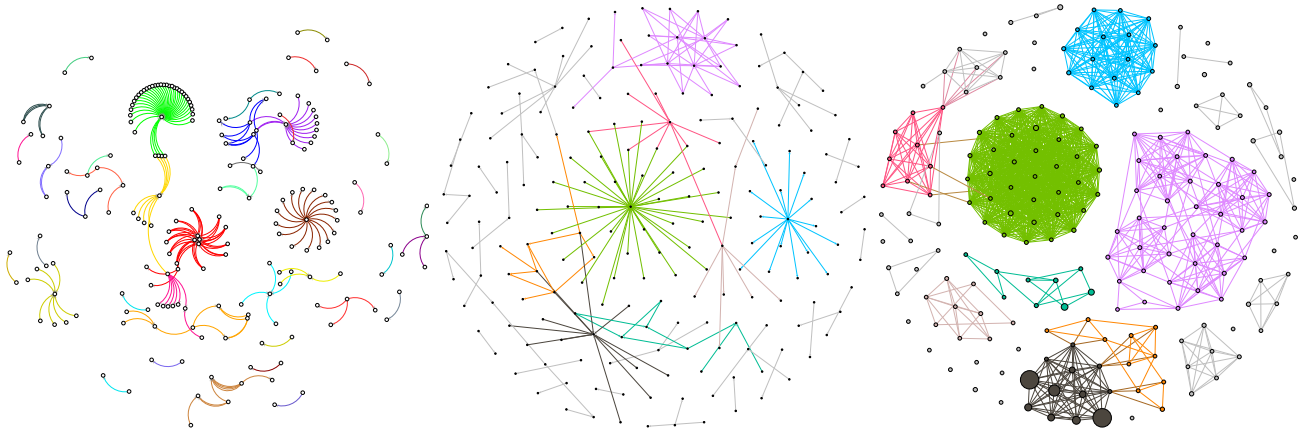


图1 告警拓扑建模为超图



(a) 原始拓扑　　(b) 抽象拓扑　　(c) 超边扩展

图2　超图模型的效果

指定相应的超边生成规则以适应不同的场景。

真实网络告警拓扑流图如图2（a）所示，告警之间直观连接较稀疏，使得更高阶的关联模式难以发现。为便于比较超图模型和普通拓扑图模型的区别，我们将原始拓扑图点边进行抽象得到了图2（b）。图2（c）则绘制了对超图进行扩展后的完全图，其中全连接的边由同一条超边转换而来，节点仍然代表告警（超图扩展方式将在下一部分进行介绍）。从图中我们可以看出，使用告警超图建模后，告警之间的关联变得更加紧密，便于发现普通图无法描述的高阶结构。

2.2　超图扩展方案[11]

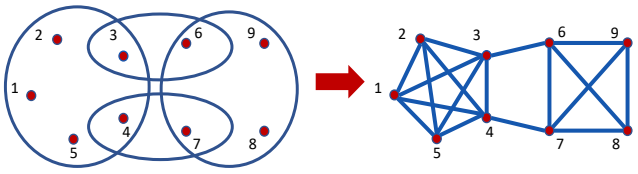


图3　超图的团扩展

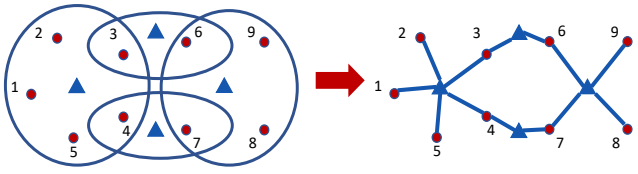


图4　超图的星形扩展

团扩展：

团扩展又叫连通分量扩展，如图3所示，将超边中所有顶点都连接在一起，比如有3个顶点的超边，扩展成普通图时两两相连就会有3条边。以此类推，连接和n 个顶点的超边拓展后有C_n^2 条边。同一个超边转换成的边具有跟以前边同样的权重。

星形扩展：

如图4所示，星形扩展在每个超边中加入一个“星星”，连接上超边中其他的点，所以这种方式会在原来的节点上增加额外的节点，也就是有加点的操作，而团扩展是没有的。加上的点归入一个集合，原来的点归入另一个集合，这个拓展后的普通图是一个二部图（bipartite graph）又称作二分图，是图论中的一种特殊模型。星图边的权重变成对应超边的权重除以超边的度。

3　形式化描述

3.1　问题定义

符号设置：

设n 是超图中的节点数，每个节点i 被分配到$\bar{l}$ 个组其中之一，$z_i \in [\bar{l}] = 1, 2, ..., \bar{l}$ 表示节点i 的组分配，放在向量$\mathbf{z} \in [\bar{l}]^n$ 中。每个节点i 都有一个参数θ_i 来控制其度数，放在向量$\theta \in \mathbb{R}^n$中。令$\mathcal{R}$ 表示无序节点元组的集合，因此每个$R \in \mathcal{R}$ 是一组表示可能超边位置的节点（我们允许$\mathcal{R}$ 包含具有重复节点的节点元组）.令$\mathbf{z}_R$ 表示给定元组R 中节点的组标签向量，而θ_R 表示度参数向量。

超图生成模型：

我们使用亲和函数Ω 来控制在给定节点元组R 上放置超边的概率，这取决于R 中节点的组分类情况。形式上，Ω将组分配$\mathbf{z}_R$ 映射为非负数。如果$\Omega(\mathbf{z}_R)$很大，则在R 中的节点之间形成超边的概率更高。在我们的模型中，在节点元组$R \in \mathcal{R}$ 处存在的超边数分布为$a_R \sim Poisson(b_R \pi(\theta_R)\Omega(\mathbf{z}_R))$，其中$b_R$ 表示对R 中的节点排序方式的数量，$\pi(\theta_R) = \prod_{i \in R} \theta_i$ 是度数参数的乘积。实现给定值a_R 的概率为：

$$\mathrm{P}\left(a_R \mid \mathbf{z}, \Omega, \boldsymbol{\theta}\right)=\frac{e^{-b_R \pi\left(\boldsymbol{\theta}_R\right) \Omega\left(\mathbf{z}_R\right)}\left(b_R \pi\left(\boldsymbol{\theta}_R\right) \Omega\left(\mathbf{z}_R\right)\right)^{a_R}}{a_R !} \tag{1}$$

参数估计：

我们通过坐标上升法执行近似最大似然推断，以学习节点标签的估计$\hat{\boldsymbol{z}}$、亲和函数的估计$\hat{\boldsymbol{\Omega}}$和度参数的估计$\hat{\boldsymbol{\theta}}$：

$$\widehat{\mathbf{z}}, \widehat{\Omega}, \hat{\boldsymbol{\theta}} \equiv \underset{\mathbf{z}, \Omega, \boldsymbol{\theta}}{\operatorname{argmax}} \mathrm{P}(\mathbf{A} \mid \mathbf{z}, \Omega, \boldsymbol{\theta}) \tag{2}$$

其中A是由（整数加权）超边集合表示的给定数据集。将公式（2）转变为对数似然函数：

$$\begin{aligned}\mathcal{L}(\mathbf{z}, \Omega, \boldsymbol{\theta}) &= \sum_{R \in \mathcal{R}} \log \mathrm{P}\left(a_R \mid \mathbf{z}, \Omega, \boldsymbol{\theta}\right) \\ &= Q(\mathbf{z}, \Omega, \boldsymbol{\theta})+K(\boldsymbol{\theta})+C\end{aligned} \tag{3}$$

其中：

$$Q(\mathbf{z}, \Omega, \boldsymbol{\theta}) \equiv \sum_{R \in \mathcal{R}}\left[a_R \log \Omega\left(\mathbf{z}_R\right)-b_R \pi\left(\boldsymbol{\theta}_R\right) \Omega\left(\mathbf{z}_R\right)\right] \tag{4}$$

$$K(\boldsymbol{\theta}) \equiv \sum_{R \in \mathcal{R}} a_R \log \pi\left(\boldsymbol{\theta}_R\right) \tag{5}$$

$$C \equiv \sum_{R \in \mathcal{R}}\left[a_R \log b_R-\log a_R !\right] \tag{6}$$

第一项$Q(\mathbf{z}, \Omega, \boldsymbol{\theta})$是对数似然取决于组分配$\mathbf{z}$和亲和函数$\Omega$的唯一部分。第二项取决于$\boldsymbol{\theta}$，第三项仅取决于数据$A$，可以出于推理目的忽略。在未知分组标签（即安全事件分组）的时候，我们通过简单图方案初始化分组标签，执行两阶段交替迭代：

（1）在第一阶段，我们假设$\hat{\boldsymbol{z}}$并通过求解下式得到新的Ω和$\boldsymbol{\theta}$估计：

$$\widehat{\Omega}, \widehat{\boldsymbol{\theta}}=\underset{\Omega, \theta}{\operatorname{argmax}} \mathcal{L}(\widehat{\mathbf{z}}, \Omega, \boldsymbol{\theta}) \tag{7}$$

（2）在第二阶段，我们假设$\widehat{\Omega}$和$\widehat{\boldsymbol{\theta}}$并通过求解得到新的$\mathbf{z}$估计：

$$\widehat{\mathbf{z}}=\underset{\mathbf{z}}{\operatorname{argmax}} \mathcal{L}(\mathbf{z}, \widehat{\Omega}, \hat{\boldsymbol{\theta}}) \tag{8}$$

我们在这两个阶段之间交替，直到最终收敛。

推广模块度：

我们通过规定Ω关于节点标签的排列对称来获得一类重要的目标函数。在这种情况下，$\Omega(z_R)$不取决于给定节点元组R中的特定标签z_R，而仅取决于每个标签的重复次数。在统计上，相应的生成超图中所有组在统计上是相同的（对称的），仅取决于其组成节点的度数。因此，我们使用对称亲和函数将种植分区SBM算法[12]（planted partition SBM）灵活地推广到超图上。

定义函数$\phi(\mathbf{z})=\mathbf{p}$，其中$p_j$是在$z$中的第$j$号最大分组在本次标签向量中出现的条目数，可以任意断掉连接。例如，如果$\mathbf{z}=(1,1,4,1,2,3,2)$，则$\mathbf{p}=(3,2,1,1)$。我们称$\mathbf{p}$为分区向量。对称假设意味着$\Omega$是$z_R$的函数，仅通过$\mathbf{p}=\phi(\mathbf{z}_R)$和它建立连接。因此，当$\mathbf{p}=\phi(\mathbf{z})$时，我们通过定义$\Omega(\mathbf{p}) \equiv \Omega(\mathbf{z})$来符号滥用[13]。

我们现在为k个节点的元组定义对应于可能的分区向量$\mathbf{p}$的广义割（generalized cuts）和广义体积（generalized volumes）：

$$\mathbf{cut}_{\mathbf{p}}(\mathbf{z}) \equiv \sum_{R \in \mathcal{R}^k} a_R \delta\left(\mathbf{p}, \phi\left(\mathbf{z}_R\right)\right) \tag{9}$$

$$\mathbf{vol}_{\mathbf{p}}(\mathbf{z}) \equiv \sum_{\mathbf{y} \in[\bar{l}]^k} \delta(\mathbf{p}, \phi(\mathbf{y})) \prod_{y \in \mathbf{y}} \mathbf{vol}(y) \tag{10}$$

其中$\mathcal{R}^k$是$\mathcal{R}$中由k个节点组成的元组子集。函数$\mathbf{cut}_{\mathbf{p}}(\mathbf{z})$计算被$\mathbf{z}$分割到指定分区$\mathbf{p}$中的边数，而函数$\mathbf{vol}_{\mathbf{p}}(\mathbf{z})$是诱导出分区$\mathbf{p}$的所有分组向量$\mathbf{y}$上的体积的和积。令$\mathcal{P}$为在大小为$\overline{k}$的集合上的划分向量集，即超边的最大大小。对称模块化目标可以写为：

$$Q(\mathbf{z}, \Omega, \mathbf{d})=\sum_{\mathbf{p} \in \mathcal{P}}\left[\mathbf{cut}_{\mathbf{p}}(\mathbf{z}) \log \Omega(\mathbf{p})-\mathbf{vol}_{\mathbf{p}}(\mathbf{z}) \Omega(\mathbf{p})\right] \tag{11}$$

对称亲和函数有不同的表达形式，见表1：

表1 对称亲和函数种类

亲和函数类型	公式
全有或全无(AON)	$\Omega(\mathbf{p})=\begin{cases}\omega_{\mathrm{k1}} & \|\|\mathbf{p}\|\|_0=1 \\ \omega_{k0} & \text{otherwise.}\end{cases}$
分组数量(GN)	$\Omega(\mathbf{p})=f(\|\|\mathbf{p}\|\|_0, k)$
相对复数(RP)	$\Omega(\mathbf{p})=g\left(p_1-p_2, k\right)$
节点对(pair)	$\Omega(\mathbf{p})=h\left(\sum_{i \neq j} p_i p_j, k\right)$

我们使用第一种亲和函数进行建模。将表1中的AON亲和函数插入公式（11），经过一些代数计算得到目标函数：

$$Q(\mathbf{z}, \Omega, \mathbf{d})=-\sum_{k=1}^{\bar{k}} \beta_k\left[\mathbf{cut}_k(\mathbf{z})+\gamma_k \sum_{\ell=1}^{\bar{\ell}} \mathbf{vol}(\ell)^k\right]+J(\boldsymbol{\omega}) \tag{12}$$

$\beta_k = \log\omega_{k1} - \log\omega_{k0}, \gamma_k = \beta_k^{-1}(\omega_{k1} - \omega_{k0})$，$J(\boldsymbol{\omega})$ 集合了不依赖于分区$\mathbf{z}$ 的项。我们将$\{\beta_k\}$和$\{\gamma_k\}$ 收集到向量$\boldsymbol{\beta}, \boldsymbol{\gamma} \in \mathbb{R}^{\bar{k}}$ 中。我们还定义了：

$$\mathbf{cut}_k(\mathbf{z}) \equiv m_k - \sum_{R\in\mathcal{R}^k} a_R \delta(\mathbf{z}_R) \tag{13}$$

在这个表达式中，m_k 是大小为k 的超边的（加权）个数，即$m_k = \sum_{R\in\mathcal{R}^k} a_R$。因此，切割项$\mathbf{cut_k}(\mathbf{z})$ 计算大小为k 的包含两个或多个不同簇中的节点的超边的数量。该计算是最近提出的图模块化方法的直接推广。我们称公式（12）是AON超图模块度。

3.2　评价指标

符号设置：

告警$a \in A$ 由一个固定的属性向量$a = (a^1, a^2, ...a^n)$组成。在我们的场景中为：源目的IP、端口、告警厂商、告警类型、源目的设备类型、攻击结果、威胁程度等。真实的攻击或业务行为i 引发一组告警S_i，所有关联告警集合由$S = \{S_0, S_1, ..., S_{n-1}\}$ 给出。这些由同一种原因引起的系列告警，具有如下特征：

（1）告警属性特征相同或大部分相似（同源、同目的、同类型、同威胁程度、同设备类型等）。

（2）构成简单，一般为单中心型，或者是典型二分图，源和目的设备类型相同。

（3）具有原子性、不可再分性，仅涉及入侵过程的一步。

告警聚类就是在A中使用各种方法，得到一个告警聚类$C = \{C_0, C_1, ..., C_{n-1}\}$，使得$C$尽可能地接近$S$。

指标定义：

a. 轮廓系数

单条告警k的轮廓系数定义为：

$$S'(k) = \frac{b(k) - a(k)}{max\{a(k), b(k)\}} \tag{14}$$

使用$a(k)$ 描述k 向量到同一簇内其他点x不相似程度$d(k,x)$的平均值。使用$b(k)$ 描述k 向量到其他簇节点y的平均不相似程度$d(k,x)$的最小值。

$$a(k) = mean(d(k,x)) \quad x \in C_k, x \neq k \tag{15}$$

$$b(k) = min(d(k,y)) \quad y \in C_j \quad C_j \in C, C_j \neq C_k \tag{16}$$

使用$d(x,y)$描述两条告警x和y的不相似程度，使用汉明距离来衡量，根据需要，不看具体IP值，而是综合考察厂商、类型、IP属性、攻击结果、威胁程度。

$$d(x,y) = \sum_{i=1}^{|a|} x[a^i] \oplus y[a^i] \tag{17}$$

归一化轮廓系数其区间范围为$[0,1]$：

$$S(k) = \frac{S'(k)+1}{2} \tag{18}$$

对聚类结果使用总平均轮廓系数来衡量：

$$Sil(C) = \frac{1}{|A|} \sum_{k\in C_i \ i\in[0,n)} S(k) \tag{19}$$

b. 原子性

因为多中心型可以看作多个单中心型的复合，所以单中心型和简单型更具备原子性。对每个告警集群，使用三种拓扑线性化指标：

$$\begin{aligned}
\delta_{\text{OtO}} &= \frac{1}{3}\cdot\left(\frac{|V|-|O|}{|V|-1} + \frac{|V|-|T|}{|V|-1} + \frac{V-|O-T|}{|V|}\right) \\
\delta_{\text{OtM}} &= \frac{1}{3}\cdot\left(\frac{|V|-|O|}{|V|-1} + \frac{|T|}{|V|-1} + \frac{||O|-|T||}{|V|-2}\right) \\
\delta_{\text{MtO}} &= \frac{1}{3}\cdot\left(\frac{|O|}{|V|-1} + \frac{|V|-|T|}{|V|-1} + \frac{||O|-|T||}{|V|-2}\right)
\end{aligned} \tag{20}$$

其中V是集群中所有节点，O是集群中所有源节点（origin），T是集群中所有目的节点（target）。δ_{OtO}是简单型，δ_{OtM}是单中心发散型，δ_{MtO}是单中心汇聚型。这三个指标哪个最大（最接近于1），就说明集群更倾向于哪个拓扑分类。其中δ_{OtO}不用计算，如果集群内只有一条告警，那么一定就是$\delta_{\text{OtO}} = 1$ 每个告警集群的原子性指标为：

$$\delta(C_i) = max(_{OtO}, _{OtM}, _{MtO}) \tag{21}$$

对聚类结果的总原子性指标为每个告警集群拓扑原子性指标的平均值：

$$\delta(C) = \frac{1}{|C|}\sum_{C_i\in C}\delta(C_i) \tag{22}$$

c. 二分性

定义聚类结果的二分性指标。其中V是集群中所以节点，O是集群中所有源节点（origin），T是集群中所有目的节点（target）。

$$Bip(C) = \frac{2|V|}{|O|+|T|} - 1 \tag{23}$$

表2 经典社区发现算法一览

算法名称	时间	类别	描述	时间复杂度
KL	1969	静态、独立	划分为已知大小的两个社区的二分贪婪方法	$O(n^2 * logn)$
GN/FN	2004	静态、独立	基于分裂思想，使用边介数作为相似度的度量方法	$O(m(m+n))$
CPM	2005	静态、重叠	寻找k-派系的连通子图，使用派系连接矩阵计算	$O(n^2)$
Walktrap	2006	静态、独立	基于随机游走和层次聚类，无法处理有向图	$O(n^2 * logn)$
LPA/COPRA	2007	静态、独立	基于标签传播的局部社区划分	$O(m)$
louvain	2008	静态、独立	基于模块度最大化的社区发现算法	$O(n * logn)$
InfoMap	2008	静态、独立	构造转移概率，随机游走层次编码，最小化L(M)	$O(n * logn)$
LFM	2009	静态、重叠	基于局部优化，可同时发现重叠社区和分层结构	$O(n^2)$
OSLOM	2011	动态、重叠	第一种能检测动态网络社区的方法	$O(n^2)$
Letden	2019	静态、独立	近几年经典社区检测的SOTA算法之一	$O(n * logn)$
谱聚类	-	静态、独立	标准化拉普拉斯矩阵+KNN（与深度学习结合）	-

d. 聚类率

为衡量算法的归并的程度，我们定义聚类率：

$$Clus(C) = 1 - \frac{|C|}{|A|} \tag{24}$$

4 聚类算法

4.1 传统社区发现算法

社区结构是指网络拓扑结构中表现出来的社团化特征，整个网络由若干个社团构成，并保证每个社团内的节点之间的连接相对非常紧密，但是各个社团之间的连接相对比较稀疏。寻找网络中存在的不相交社区结构的过程称为社区发现。

经典社区发现算法的概况见表2，这些算法主要分为以下几类：

（1）基于图划分：通过将节点或边进行直接划分以寻找可能的社区，包括最早的二分贪婪划分算法KL和经典的FN/GN算法。此方法复杂度不高但经常难以发现深层结构。

（2）基于模块度：最早由Newman定义，存在许多模块度的扩展变种，最经典的当属louvain[14]（Fast Unfolding），近些年提出的Leiden算法[15]被认为是此领域的SOTA算法。但是此方法存在不稳定、过拟合、分辨率限制等问题。

（3）基于动力学模型：复杂网络具有动态性，节点具有随机性，但社区内节点间的连接相对随机变化来说又显得很紧密。基于上述网络动态特征，衍生了一系列基于动力学的社区发现算法，如LPA、Walktrap、InfoMap等。这些算法大部分基于随机游走，可能在社区内部会停留很长的时间。

（4）基于网络局部优化：从宏观上看，网络社区结构本身就是属于一种局部特征，它的组成只与其社区内部的节点和边连接有关，而与网络拓扑结构的其它部分区域无关。所以，使用基于局部网络信息的社区发现算法在理论上应该更加符合社区结构的定义常见算法有LFM和CPM，此类算法往往时间复杂度高。

（5）基于谱聚类：该方法一般采用求解网络节点邻接矩阵的特征向量(也可是邻接矩阵的拉普拉斯矩阵)，其节点相似性可通过向量间的夹角或者距离来进行判定，最后按照某种节点聚类算法来获得网络结构的社区划分。

（6）基于概率生成模型：此类方法可看作是以统计推理为特征的一类特定算法，通过假设连接节点对的边的出现概率来构造算法模型用以对原始网络进行模拟，然后推理出与客观事实相符的且能代表该网络特征的社区结构。常见算法有OSLOM，此思想也是本文所采用的。

本文所提的方法为基于扩展模块度的超图聚类，同时借鉴概率生成模型的思想，使用生成式建模的方法对告警进行可解释性归并。我们先来回顾一下经典模块度社区发现算法louvain，如图5所示主要分为两步：

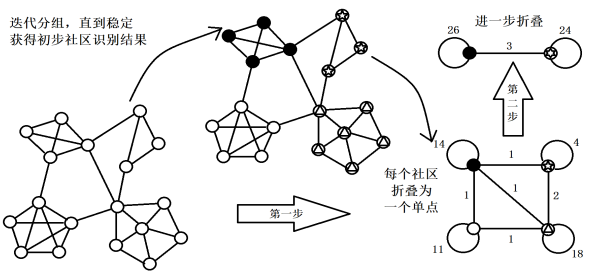


图5 经典louvain算法原理

（1）模块度优化阶段：每个节点将自己作为自己社区标签。每个节点遍历自己的所有邻居节点，尝

试将自己的社区标签更新成邻居节点的社区标签，选择模块度增量最大(贪婪思想)的社区标签，直到所有节点都不能通过改变社区标签来增加模块度。（2）网络凝聚阶段：每个社区合并为一个新的超级节点，超级节点的边权重为原始社区内所有节点的边权重之和，形成一个新的网络。

4.2 生成式超图聚类算法

当Ω 是AON 亲和函数时，对每条边我们不需要计算完整的分区向量$\mathbf{p}$，而只需检查$||\mathbf{p}||_0 = 1$是否成立。我们不给出一般的亲和函数Ω，而是提供出现在公式(12)中的参数向量$\boldsymbol{\beta}$ 和$\boldsymbol{\gamma}$。这使我们能够在相当简化的数据结构上进行计算。特别是，我们能够遵循经典的Louvain策略，将集群折叠成单个合并的超级节点，并将注意力限制在跨越多个超级节点的超边上。因为我们不需要跟踪超边跨越多个超节点的精确方式，所以我们可以忘记大部分原始邻接数据$\mathcal{A}$，而是简单地存储超图的边大小。这些简化既可以节省大量内存，又可以非常快速地评估目标更新函数ΔQ。

算法1 描述了AON GHCAM的外循环。整体结构与内循环算法2 中的大致相同。在外循环的每次迭代中，我们使用函数$Collapse(H, \mathbf{z})$ 将初始聚类$\mathbf{z}$ 折叠成简化的超图$\bar{H}$，这种方式仍然保留了特殊的AON 结构。我们在简化超图上运行Louvain 步骤，然后使用函数$Expand(H, \mathbf{z}, \bar{\mathbf{z}})$ 将简化超图上的聚类$\bar{\mathbf{z}}$ 转换为原始超图上的聚类$\mathbf{z}'$。如果$\mathbf{z}$ 和$\mathbf{z}'$重合，则Louvain 步骤没有发现任何改进，我们终止。否则，我们开始一个新的迭代，首先在运行Louvain 步骤之前将更新的聚类折叠成一个简化的超图。在构造简化表示$\bar{H}$ 时，有必要仅将折叠的超图与每个边的原始大小的向量$\bar{\mathbf{s}}$ 一起存储。每个折叠节点l 的度数$\bar{d}_l$ 只是相应集群l 中节点的度数之和。

算法1　AllOrNothingGHCAM(H, β, γ)

Data: 超图H, 参数向量β 和γ
Result: 更新后的标签向量$\mathbf{z}$

$\mathbf{z}' \leftarrow \mathbf{z} \leftarrow [n]$ // 初始化将每个节点视为一个集群
do:
　$\mathbf{z}' \leftarrow \mathbf{z}$
　$\overline{H}, \overline{s} \leftarrow Collapse(H, \mathbf{z})$
　$\overline{\mathbf{z}}' \leftarrow AONLouvainStep(\overline{H}, \overline{s}, \beta, \gamma)$
　$\mathbf{z}' \leftarrow Expand(H, \mathbf{z}, \overline{\mathbf{z}}')$
while: $\mathbf{z} \neq \overline{\mathbf{z}}'$
return: $\mathbf{z}$

Louvain 步骤本身由**算法2** 给出。在每个阶段，我们检查折叠节点i 并计算与将$\bar{\mathbf{z}}$ 更改为$\bar{\mathbf{z}}^{i\mapsto A}$时相应目标函数的变化，其中：

$$\overline{\mathbf{z}}_j^{i\mapsto A} = \begin{cases} A & i = j \\ \overline{\mathbf{z}}_j & \text{otherwise} \end{cases} \tag{25}$$

此式是通过设置$z_i \mapsto A$ 而其他保持不变获得的折叠标签向量。该计算包含在子程序**算法3** ΔQ_{AON} 中。

由于公式（12）中体积项的局部变化只需要更新集群体积的幂和，由分辨率参数 和进行放缩。更新公式（12）的割项需要对与要更新的节点i 相邻的每个超边的切割状态的变化求和，所以对于推广模块度的计算非常快。

在真实计算的过程中，我们需要指定初始的参数向量β和γ，因此需要使用传统社区发现算法生成初始化的集群向量$\mathbf{z}$——这一步需要给定初始化的聚类分辨参数。按照公式（7）和公式（8）迭代地使用最大似然估计法，此处亲和函数Ω对应AON亲和函数的参数向量β和γ，将得到的参数向量代入GHCAM算法（算法1）即可得到使用生成式超图归并框架计算出的聚类结果。

算法2　AONLouvainStep($\overline{H}, \overline{s}, \beta, \gamma_l$)

Data: 折叠超图$\overline{H} = (\overline{V}, \overline{E})$，边尺寸向量$\overline{s}$，
　　参数向量β 和γ
Result: 在$\overline{V}$上的标签向量$\overline{\mathbf{z}}$

$\overline{\mathbf{z}} \leftarrow [\overline{n}]$ // $\overline{n}$是超图$\overline{H}$中的节点数
$improving \leftarrow true$
while $improving$ **do:**
　$improving \leftarrow false$
　for $i \in \overline{V}$ **do:**
　　$\mathcal{E}_i \leftarrow \{e \in \bar{E} \mid i \in e\}$ // 包含节点i的超边
　　$\mathcal{A}_i \leftarrow \{\ell \mid \exists j \in \mathcal{E}_i : z_j = \ell\}$ // 与i 相邻集群
　　// 把节点i 移动到集群$A' \in Ai$ 时
　　Q的最大改变Δ 和取最大值时的集群A
　　$(\Delta, A) \leftarrow \operatorname{argmax}_{A' \in \mathcal{A}_i}$
　　　　$\Delta Q_{\text{AON}}\left(\bar{H}, \overline{\mathbf{z}}, \bar{\mathbf{s}}, A', i, \mathcal{E}_i, \boldsymbol{\beta}, \gamma\right)$
　　if $\Delta > 0$ **then:**
　　　$\bar{z}_i \leftarrow A, improving \leftarrow true$
　　end
　end
end

算法3 $\Delta Q_{\mathrm{AON}}\left(\bar{H},\overline{\mathbf{z}},\overline{\mathbf{s}},A',i,\mathcal{E}_i,\boldsymbol{\beta},\gamma\right)$

Data: 折叠超图$\overline{H}=(\overline{V},\overline{E})$，当前聚类向量$\overline{\mathbf{z}}$，边尺寸向量$\bar{s}$，候选新集群$A'$，移动节点$i$与i 相关的边集$\mathcal{E}_i$，参数向量$\beta$ 和γ

Result: 将节点i 移到新集群A 相关模块度的变化

$v_A \leftarrow \mathrm{vol}(A), v_i \leftarrow \mathrm{vol}(\overline{\mathbf{z}}[i])$

$\Delta v =$

$\sum_{k=1}^{\bar{k}} \beta_k\gamma_k\left[v_i^k-\left(v_i-\bar{d}_i\right)^k+v_A^k-\left(v_A+\bar{d}_i\right)^k\right]$

$\Delta c=\sum_{e\in\mathcal{E}_i}\beta_{\bar{s}_i}\left[\delta\left(\overline{\mathbf{z}}_e^{i\mapsto A}\right)-\delta\left(\overline{\mathbf{z}}_e\right)\right]$

return: $\Delta c+\Delta v$

5 实验验证

5.1 与传统方法比较

为对比本文使用的生成式超图聚类算法和传统社区发现算法，我们选取了常规的企业内网环境每小时内的告警数据并计算平均。此数据集包含6个不同厂商的IDS设备产生的告警，平均每天的告警数量达数十万条。我们选取的这段时间属于常规运行维护期，一小时告警数量在5000条左右。图6展示了不同算法对于同一数据集运行时间的对比。算法使用科学计算语言Julia进行实现。生成式超图算法的运行时间为右一，可以看出生成式超图算法在时间上比传统算法更快，相比号称“改进稳定模块度”的ecg[16]算法（左三）效率有了极大提升。

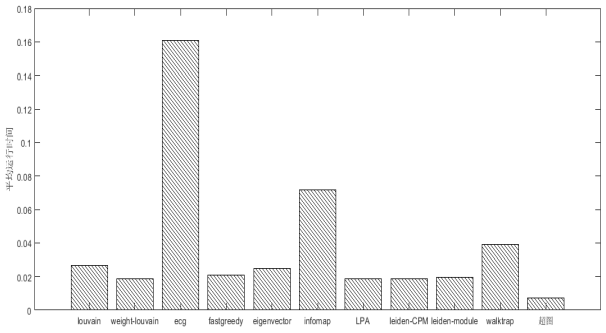


图6 超图聚类与传统方法运行时间对比

对于第三节提出的四种归一化评价指标，越接近于1就说明此算法在此数据集下该指标更优。由于数据集限制，在当前实验使用的仅考虑拓扑的建模方法下，各种算法的二分性指标都为1，也就是获得的告警集群都为标准二分图，不存在告警链路和环。图7展示了上节提到的各种经典算法和本文生成式超图算法在其它三种评价指标上的表现。图中从左到右依次是：聚类率、轮廓系数和原子性，分别描述了聚类结果的归并程度、簇间相似性和集群独立性。可以看出本文中的超图算法（最右）在各项指标上都能够和传统算法看齐，同时在某些指标上显示出了不小的优越性。

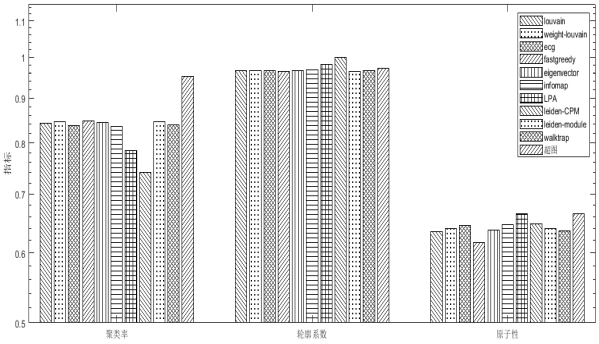


图7 超图聚类与传统方法评估指标对比

5.2 两种扩展方式对比

三种典型场景：

为进一步探究生成式超图归并框架采用不同初始化方案（团扩展、星形扩展）的区别，以及不同初始化生成分辨率参数对最终结果的影响，我们选取了三种典型网络攻击情形对不同初始化方案和初始化参数进行比较。图9展示了三种典型的测试场景，从左往右依次是：（1）端口扫描行为，此模式表现为大量同目的告警，在超图模型中表现为节点数量庞大的巨型超边；（2）渗透测试行为，此模式为内部护网行动产生的模拟进攻告警，在超图上表现为大量节点数差不多的超边；（3）常规网络行为，此模式为正常系统业务触发的日常告警，超边数量和节点数量都相对较小。

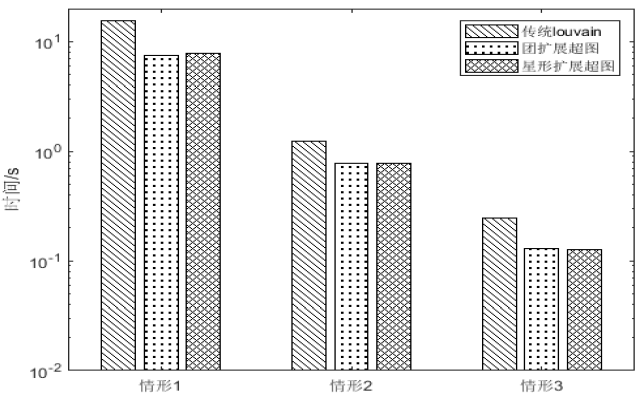


图8 三种典型场景平均时间对比

(a) 场景1 扫描　　(b) 场景2 渗透测试　　(c) 场景3 常规行为

图9　三种典型测试场景

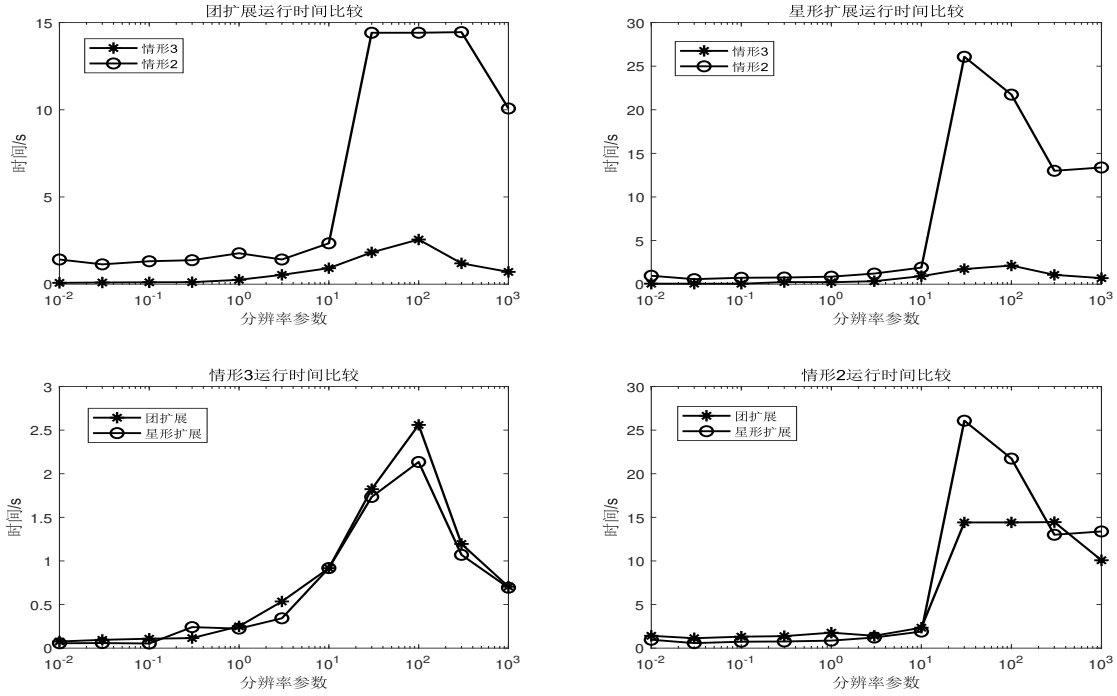


图10　两种扩展方式平均时间对比

运行时间对比：

图8展示了经典louvain算法和生成式超图框架的两种扩展方式应对三种不同场景的运行时间比较。可以发现扫描行为情形下，三种算法都比较耗时，都需要10秒左右。在渗透测试情形下，三种算法都能在1秒内获得结果，基本满足实时性需求。在常规情形下，超图算法运行速度可以达到0.1秒，说明了本文超图框架的可行性。超图方法比传统方法稍快一些，这是因为超图算法避免了对扩展后高密度图的大量重复计算。同时我们注意到两种超图初始化扩展方式在时间上差别不大，对于扫描行为情形，由于大度数节点的存在，使得超图模型中超边的度分布也相当不均匀，甚至可能影响到细粒度的告警归并。设置预先的扫描行为检测与归并算法是解决此问题的有效方案。

我们对初始化分辨率参数对运行时间的影响曲线在不同情形上进行了纵向比较，同时在两种不同超图扩展方式上进行了横向比较。图10上一行展示了情形2和情形3，即渗透测试和常规行为情形下的分辨率-时间曲线对比。由于情形1扫描行为运行太过耗时，所以此处没有进行展示。我们可以看出常规行为下运行时间都在1-2秒左右，但渗透测试情

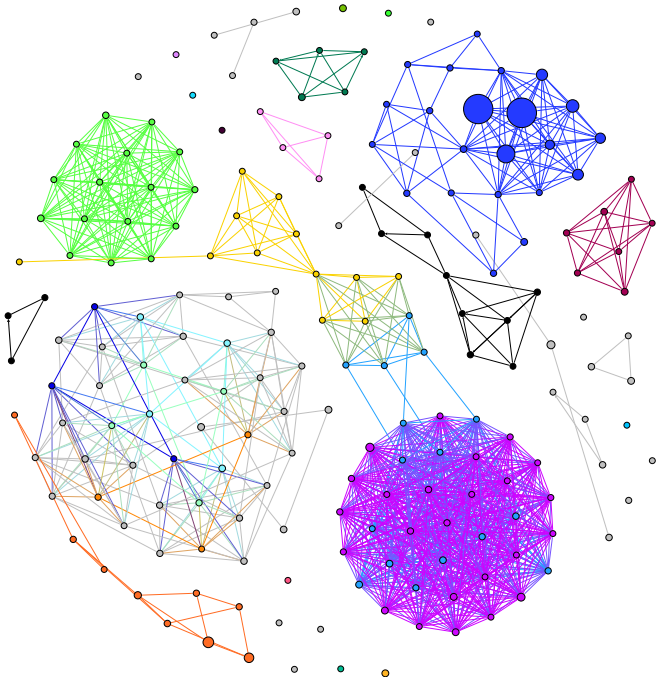


图11 团扩展划分效果

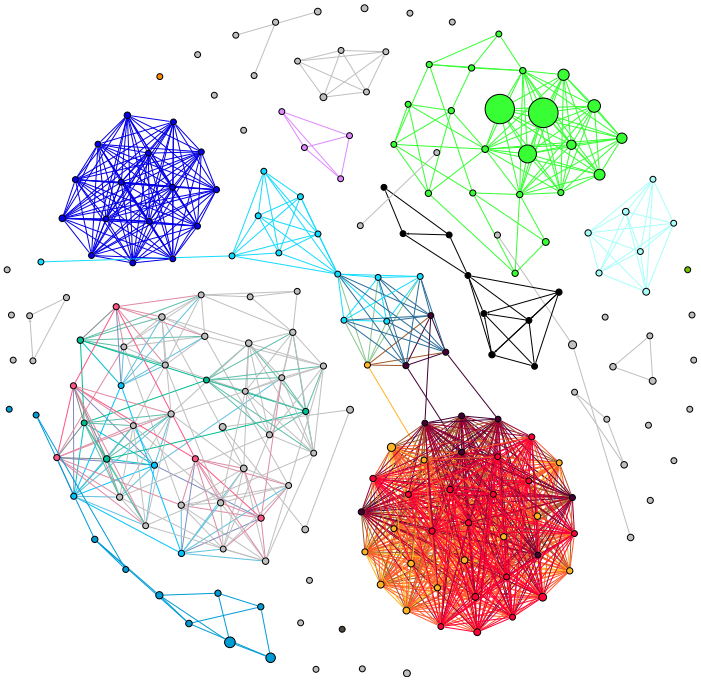


图12 星形扩展划分效果

形下当分辨率达到10以上数量级时，运行时间会有突变，最终时间会稳定在10秒量级左右。图10下一行展示了团扩展和星形扩展方式的分辨率-时间曲线对比。可以看出相同网络攻击情形下，两种扩展方式对曲线变化趋势影响不大，分辨率达到10以上数量级时，所有情形都会产生一定的阶跃，阶跃变化的幅度取决于当前的安全态势。

评估指标对比：

为了直观展示不同初始化扩展方式对聚类效果的影响，我们选取分辨率为3.0，分别对超图聚类结果进行了可视化。图11和图12分别是团扩展和星形扩展的划分结果，可以看出，忽略随机分配的集群颜色，两种方法的归并结果无肉眼可见的不同。由于分辨率参数大于1，所以生成的划分簇数量比第二节图2（c）更多，粒度也更细。

针对不同安全情形和不同初始化方式，我们对上小节提到的三种评估指标与三者的和随分辨率参数的变化趋势进行了绘制，图13展示了这种趋势变化。上行是团扩展，下行是星形扩展；左列是情形3常规行为，右列是情形2渗透测试。从图上我们可以看出，不同扩展方式对分辨率-评估指标的变化趋势影响不大。对于常规行为来说，分辨率小时划分效果较好，而对于渗透测试的复杂情形，划分效果存在一定的起伏，在分辨率参数为1左右时效果最好。这是因为渗透测试时存在大量分散的攻击行为，必须使用较大的分辨率参数以发现更细粒度的告警集群。而当分辨率参数过大时，所有算法都倾向于把告警按单条进行划分，这就失去了告警归并的意义。

观察三个评估指标的变化趋势我们可以得到：原子性随着分辨率参数的增大先减小后增大；聚类率随分辨率的增大而减小；针对常规情况，轮廓系数随分辨率增大先减小后增大，而渗透测试情形下轮廓系数先增大后减小，这说明轮廓系数收网络安全态势影响较大。即，不同情形下评估指标的不同变化趋势的主要影响因素是轮廓系数，说明告警的相互作用和安全态势息息相关。

针对不同的安全情形，我们可以有针对性地选择不同的初始化分辨率参数以获得最佳效果，同时要合理选取分辨率参数以避免过拟合的情况发生。总而言之，生成式超图聚类算法可以快速、便捷地发现高质量的安全事件告警集群，同时为发现的结果提供概率意义上的解释。

5.3 告警归并效果

图14展示了使用超图聚类算法的告警归并效果，其中最上方的折线代表原始告警数量，中间的折线代表使用传统告警归并方案（同属性优先归并）后剩余的告警数量，下方的折线代表使用生成式超图归并框架得到的安全事件数量。上下两幅图分别展示了不同安全环境下（平时和护网期间）两种框架的归并效率。可以看出本文使用的框架可以有效地将告警数量降低1-2个数量级，并且可以从数量上直观地反映出安全事件和工作日休息日的

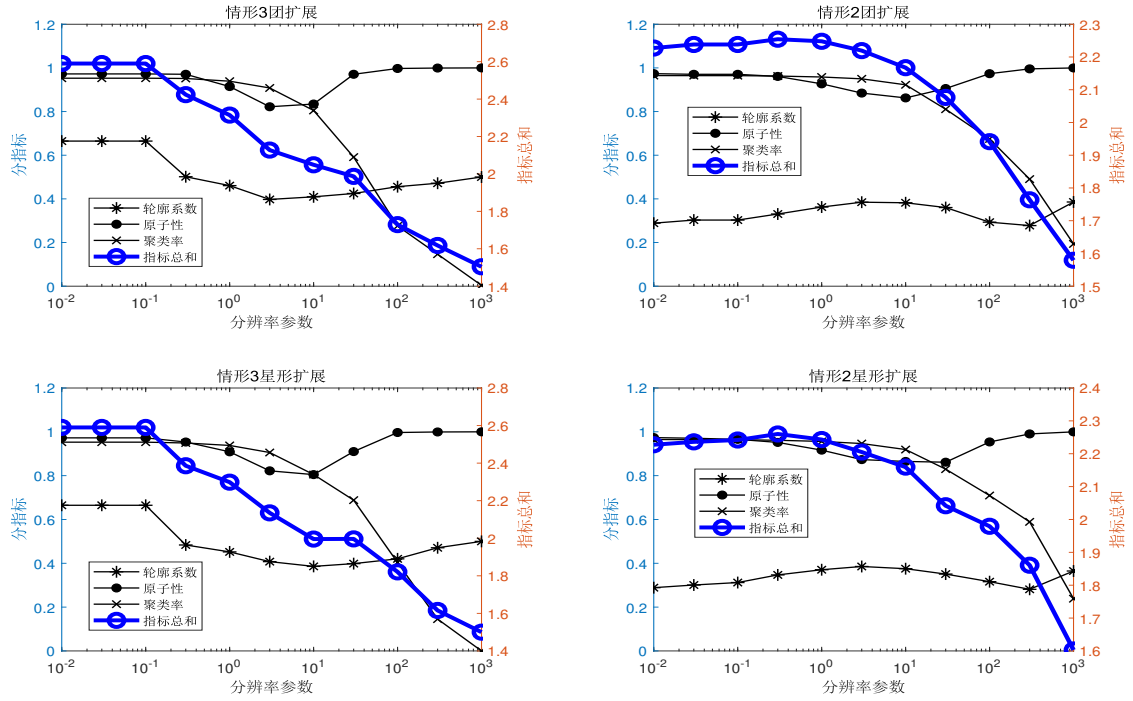


图13　两种扩展方式评估指标对比

关联：安全事件数量每七天减小一次，意味着到周末网络中的业务减少，相应触发的安全事件也变少了，这在原始告警数量图中是完全无法体现的。

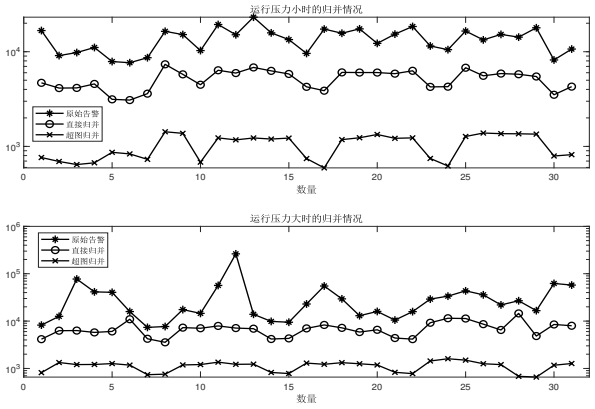


图14　使用超图聚类算法的告警归并效果

6　结论

本文针对海量网络告警日志，使用生成式超图归并框架对关联程度高的告警进行可解释性聚类。根据定义的归并规则使用超边对告警进行建模，保证了算法的可扩展性，同时保留了告警之间的高阶关联。实验表明，此框架比传统社区发现算法运行时间缩短50%，在自己定义的归并效果评估指标上有更好的表现。算法初始化时支持选择两种不同的超图扩展模式，研究显示扩展方式对运行时间和归并效果影响不大。使用最大似然准则对当前告警相似度超图进行参数估计以指导聚类过程，使得最终的归并结果可以从概率角度进行解释。我们为算法提供了可选的分辨率参数，可以根据安全情况进行动态选取。实验表明常规安全形势下使用最小分辨率参数即可获得最佳归并效果和最短运行时间；在安全形势复杂的情况下需要适当提高分辨率参数；对于扫描行为来说，由于不平衡大度数超边的存在，此算法运行较慢，可以考虑加入预先归并过程对大量重复扫描行为进行预归并。研究表明，生成式超图归并框架可以将告警数量减少1-2个量级，同时可以发现许多传统归并方案难以发现的时域特征。

未来工作：

目前对告警关联超图的建模准则还比较初步，仅停留在拓扑属性上，这就使得分散式攻击行为难以被发现。对告警生成原理的建模能够帮助我们更好地理解安全事件，并制定更好的规则。当前生成式超图归并算法本质上是基于louvain的扩展算法，对于稳定性的指标尚待完善。实验部分使用的数据集比较单一，许多攻击场景没有得到完全复现，未

来计划扩展数据集以测试更多安全情形下本框架的性能。如何更快速地发现层次性的安全事件模式、如何对挖掘出的安全事件进行深层次关联分析将会是令人感兴趣的研究方向。

参 考 文 献

[1] S. Haas and M. Fischer, "Gac: graph-based alert correlation for the detection of distributed multi-step attacks," in *the 33rd Annual ACM Symposium*, 2018.

[2] K. Julisch, "Mining alarm clusters to improve alarm handling efficiency," in *Computer Security Applications Conference, 2001. ACSAC 2001. Proceedings 17th Annual*, 2002.

[3] F. Cuppens and A. Miège, "Alert correlation in a cooperative intrusion detection framework," *IEEE Computer Society*, 2002.

[4] K. Julisch, "Clustering intrusion detection alarms to support root cause analysis," *ACM Transactions on Information and System Security*, vol. 6, no. 4, pp. 443–471, 2003.

[5] G. Giacinto, R. Perdisci, and F. Roli, "Alarm clustering for intrusion detection systems in computer networks," in *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, 2005.

[6] M. Raman, N. Somu, K. Kirthivasan, R. Liscano, and V. Sriram, "An efficient intrusion detection system based on hypergraph - genetic algorithm for parameter optimization and feature selection in support vector machine," *Knowledge-Based Systems*, vol. 134, no. oct.15, pp. 1–12, 2017.

[7] Imre, Derényi, Gergely, Palla, Tamás, and Vicsek, "Clique percolation in random networks," *Physical Review Letters*, vol. 94, no. 16, pp. 160 202–160 202, 2005.

[8] P. S. Chodrow, N. Veldt, and A. R. Benson, "Generative hypergraph clustering: From blockmodels to modularity," *Science Advances*, vol. 7, no. 28, p. eabh1303, 2021.

[9] R. Lambiotte, J. C. Delvenne, and M. Barahona, "Laplacian dynamics and multiscale modular structure in networks," *Physics*, 2012.

[10] V. A. Traag, P. V. Dooren, and Y. Nesterov, "Narrow scope for resolution-limit-free community detection," *Physical Review E*, vol. 84, no. 1 Pt 2, p. 016114, 2011.

[11] Y. Dong, W. Sawin, and Y. Bengio, "Hnhn: Hypergraph networks with hyperedge neurons," 2020.

[12] C. Lee and D. J. Wilkinson, "A review of stochastic block models and extensions for graph clustering," *Applied Network Science*, vol. 4, no. 1, 2019.

[13] A. Fb, B. Gc, D. Iic, G. Vlcef, J. Mlhi, K. Ap, L. Jgy, and N. Gpm, "Networks beyond pairwise interactions: Structure and dynamics," *Physics Reports*, vol. 874, pp. 1–92, 2020.

[14] V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics Theory & Experiment*, 2008.

[15] V, A, Traag, L, Waltman, N, J, van, and Eck, "From louvain to leiden: guaranteeing well-connected communities." *Scientific Reports*, 2019.

[16] V. Poulin and F. Théberge, "Ensemble clustering for graphs: Comparisons and applications," *Applied Network Science*, 2019.