OPEN ACCESS

University of BRISTOL

Zakrzewski, R., Martin, T. P., & Oikonomou, G. (2022). Graph modelled system change detection in WSNs. In *2022 11th Mediterranean Conference on Embedded Computing, MECO 2022* (2022 11th Mediterranean Conference on Embedded Computing, MECO 2022). Institute of Electrical and Electronics Engineers (IEEE). https://doi.org/10.1109/MECO55406.2022.9797209

Peer reviewed version

Link to published version (if available):
10.1109/MECO55406.2022.9797209

Link to publication record in Explore Bristol Research
PDF-document

This is the accepted author manuscript (AAM). The final published version (version of record) is available online via IEEE at 10.1109/MECO55406.2022.9797209. Please refer to any applicable terms of use of the publisher.

University of Bristol - Explore Bristol Research
General rights

This document is made available in accordance with publisher policies. Please cite only the published version using the reference above. Full terms of use are available:
http://www.bristol.ac.uk/red/research-policy/pure/user-guides/ebr-terms/

Graph modelled system change detection in WSNs

Robert Zakrzewski
Communication Systems and Networks
University of Bristol
Bristol, UK
robert.zakrzewski@bristol.ac.uk

Trevor Martin
Artificial Intelligence Group
University of Bristol
Bristol, UK
trevor.martin@bristol.ac.uk

George Oikonomou
Communication Systems and Networks
University of Bristol
Bristol, UK
g.oikonomou@bristol.ac.uk

***Abstract*—Graphs are suitable to model topology and data patterns in systems such as WSNs. To detect change, there is a need for graph comparison, a computationally demanding task difficult to run on constrained devices. For monitoring, the definition of normal patterns, and deviation from normal are required. In this contribution a flexible graph comparison method allowing monitoring of normal patterns and metrics providing measures of deviation from normal are proposed. In this manuscript, we apply the method to the system modelled by synthetic and random graphs. We demonstrate that the fingerprints of normal topology and data patterns can be acquired with the measures of deviation from normal. We discuss applicability of the method at the edge of WSNs.**

***Index Terms*—Wireless sensor networks, Data pattern, Topology, Graph comparison, Monitoring,**

I. Introduction

Wireless Sensor Networks (WSNs) are vulnerable to attacks due to limited resources available for protection and deployment which by nature is ubiquitous. Some vulnerabilities can be reduced by careful hardware and software design but due to diverse market with many manufactures it is difficult if not impossible to assess security of devices. For critical devices, security certification process may be used, usually costly and time consuming which is clearly in contrast to low-cost nature and ubiquity of sensing devices [1].

Typically, security is addressed by threat modelling. This multi-step process is expensive and requires definition of use cases and entities to protect, threats and possible attackers, security objectives, and the way to address them [2]. The modelling suffers from limited information about devices, diverse nature of Eco-system to protect and inability to predict all attack vectors. To circumvent this limitation, a "black box" approach can be used to analyze what is observable externally (e.g. at the edge) to infer about something unusual occurring in the system. This typically involves data (also encrypted) and network topology which are subject to attacks such as denial of service, and malicious topology change to facilitate data eavesdropping. By defining topology and data patterns it becomes possible to monitor these normal patterns and flag up any deviations.

In this contribution, topology and data patterns are modelled as graphs. To allow comparison, graph similarity measures are proposed allowing acquiring normal data and topology patterns and detection of deviation from normal. The method is flexible as graphs can be scaled to reflect computational resources available and can be deployed at edge devices.

II. Related work

Machine learning and neural networks have been used to analyse graphs. An example of which are deep graph kernels in [3] and graph convolution networks [4], [5] which use convolution operation and neural networks for graph analysis.

Nodes embedding techniques in [6], [7] help perform graph mining for vertex and edge comparison for the tasks such as vertex labelling and link predictions. They also use neural networks and require model parameters optimisation by minimizing the loss function during the training. The loss function compares the statistics acquired from random walks performed on a graph with the statistics encoded by the embeddings. Stochastic gradient descent is used to find a solution to the optimisation with some approximation. The loss function quantifies errors in the approximation, and does not define or impose any restriction on relation between the embeddings in the latent space. As a result, the similarity measure becomes inaccurate with the dissimilarity increase. The technique is not suitable for comparison between graphs as each run of the optimisation algorithm may yield different results. For weighted graphs, the approximation in results impacts detection sensitivity. In [8] it was proven the optimisation process, finds a solution (many may exist) to a matrix factorisation problem for the random walk statistics. Matrix factorisation may be seen as finding latent representations of target statistics which reside on a hyper-plane.

As attack vectors are often scarce and may not reflect the latest threats, a system capturing robust representation of the normal is of value. In [9], [10] the latent representation of the normal is found by varational autonecoders, which can also be used for anomaly detection. As deep models are used, the problem of latent space irregularity is solved by encoding latent space following standard (Gaussian) distributions, and by defining the loss function minimising reconstruction errors and penalising irregularity. This achieves regularisation of the latent space in the high dimensional space. The normal data is used for training and to facilitate classification and anomaly detection. However, these methods are computationally expensive, requiring considerable data-sets for training.

In this contribution, a graph comparison method for directed, weighted graphs is poposed. The method allows capturing normal topology and data patterns fingerprints which can be used for further processing and change/anomaly detection in the systems such as WSNs.

III. Normal modelling

Graphs are naturally suited to model relations. A relation is an abstract term and is associated with passing or possessing information. Information may be related to control, user data or metrics measured and associated with entities. Topology is often defined as a geometric form that remain the same after continuous (smooth) transformations. To model information passing, besides defining information, the paths need to be

considered. That implies geometric structure, the topology with additional description about information being passed. The description can be defined for vertices and edges and contained in labels. Labels attached to vertices usually reflect permanent state, whereas labels attached to edges (e.g. weights) may represent volatile state. Weights may contain measured information. In communication systems such as WSNs, topology is used to describe connectivity between nodes. A WSN may use part of topology, a sub-topology to pass information (data), and routing protocols make sure that the currently used sub-topology gives the highest probability of successful delivery (e.g. Routing Protocol for Low-Power and Lossy Networks [11]). As the current sub-topology may change over time, the normal topology patterns consists of all legitimate sub-topologies. Likewise normal data patterns consist of a set of sub-topologies and measured egress data distributions from vertices. In this contribution systems modeled as graphs are compared using the proposed graph comparison method. Normal topology patterns and normal data patterns are defined as sets of numbers, which can be processed further as demonstrated in [12].

A. Data and topology pattern change

Difficulty to define topology pattern stems from verbosity of a graph which is further exacerbated when graphs are compared. A topology pattern is defined as an ordered pair of directed graphs (graph A, graph B) each modelling connectivity between sensors in a WSN network. Graph A is the selected baseline and graph B is the assessed graph. The topology pattern measure is a comparative measure defined by metrics which describe pattern change. A single n-vertex directed graph has $(2^n-1)^n$ different topology patterns excluding the patterns which have one or more vertices with no egress edges. If a vertex has no egress edge, it is modelled as having a self loop. For example, a 4-vertex graph excluding vertices with no edges, has $(2^4-1)^4 = 50625$ topology patterns. A data pattern is defined by distributions of weights assigned to edges. For a given topology, data patterns are unbounded, as each vertex may have many different continuous distributions of weights. The data pattern depends on the current sub-topology. In this contribution a method measuring topology and data pattern change in relation to the selected base graph is discussed.

B. System view (WSN)

WSNs are enablers for sensors to send data to the main infrastructure modelled as a cloud with the edge device, a border router interconnecting the cloud with the sensors. Due to distributed nature of WSNs, a monitoring system at the edge capable of detecting topology and data pattern change is of great benefits. Sensors push data to the cloud, receive control data, or are polled by server processes. Topologies representing data path connectivity are formed in WSNs based on 6LoWPAN/IPv6 which are deployed in a multi-hop mesh scenario. In this setup, routing protocol needs to be used to form paths for data. Often RPL (Routing Protocol for Low-Power and Lossy Networks) is used which perceives the network as a Destination Oriented Directed Acyclic Graph (DODAG). DODAG is not always a tree, but in practice the existing RPL implementations result in the routing topology being a tree with the root at the gateway. This means that traffic flowing up the tree towards the root causes a funneling effect as sensors relay data to/from other sensors to facilitate the distribution of WSNs. This makes the system vulnerable to attacks and miss-configuration with the impact exacerbated as the impaired devices get closer to the edge router. For these reasons detection of topology and data patterns change with locality information is valuable to implement countermeasures. The graph is built to reflect topology of the system. Edges represent data flows; vertices represent devices. Vertices are annotated with labels reflecting permanent state which in WSN is a unique IPv6 address. As a measured and volatile state, the normalized data volumes passed between vertices is used to label edges. Sensor data traffic is often MQTT over TCP and MQTT and TCP protocols are used to push data to the cloud. The method presented in this paper was applied to WSNs in [12]. However, it can also be applied to other systems, and any protocol layer (e.g. a bespoke network stack), and to measure protocol data and/or traffic volumes, which makes the method flexible.

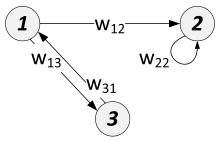

Fig. 1. Directed, weighted, annotated graph (example)

C. Graph comparison method

The method proposed in this paper is a matrix-based method which combines concepts of obtaining graph feature vectors in [7], and the latent space regularisation in [9] for anomaly detection. Vertices of a weighted directed graph are transformed to the latent space which is a bounded hyper-plane as in [7]. The latent space is regularised requiring same distance between the embeddings (similarity to [9]) but the dimensionality of the latent space is not reduced. With these assumptions, it is possible to find matrix factorisations analytically, in section III-D, and later perform calculation of vertex metrics for lower dimensionalal representation, and further processing.

Unlike in [9], the latent space is flat and bounded, varational inference is not required ($\boldsymbol{N}(\overline{x}, s^2)$ parameters are not estimated), and the latent space vectors are not sampled from the target distributions. However, the latent space representations of the normal graph vertices are regularised conditional probability distributions of data over egress flows given each vertex assuming $s^2 = 0$. In [10], the Kullback–Leibler (KL) distance measure of conditional posterior distribution of the latent variable given a new sample, and the prior is used as the measure of anomaly. The KL measure is not used in this contribution, but d1 metrics in section III-D is calculated instead as a new measure of topology/distribution change. If data-sets are rich, the statistics of d1 are also calculated, or otherwise thresholds are used for anomaly detection as discussed in [12]. Metrics d3 in section III-D is used to provide additional discrimination when d1 on its own does not suffice, and is used for patterns fingerprinting. Topology hash is also calculated as discussed in section IV to further differentiate scenarios when topology differs but some edges have small weights leading to vertex vector representations proximity.

As n-vertex graph requires a hyper plane in n-dimensions (R^n) for the latent space, n vertices suffice to uniquely define the hyper-lane. If some vertices have equal distributions or are linearly dependant, the least square approximation is used to find the plane. Each assessed vertex is compared with baseline n-vertices due to regularisation. N-vertices define the

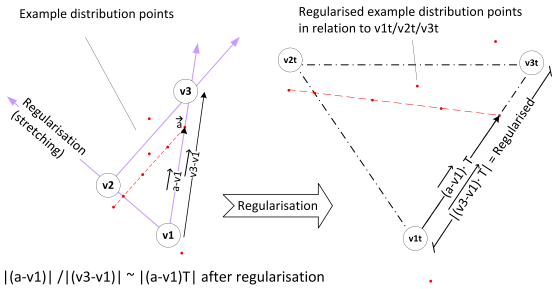

Fig. 2. Regularisation and relation to normal baseline vertices

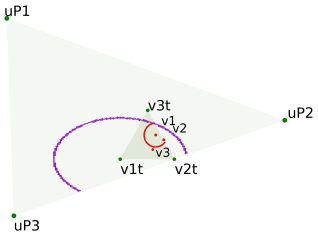

Fig. 3. Domain space transformation after regularisation

partitioning of R^n and the comparison scale. A distribution also defines topology as absent edges are marked by weights equal to zero. As a hyper-plane is smooth, a group of points having different distributions but the same topology can be identified. Two groups are in neighbourhood if their topology differs by 1 edge which enforces a topological layout.

An example in 3d space is given in Fig.3. The points (v1,v2,v3) represent probability distributions of egress flows for vertices of the baseline graph in Fig.4 before regularisation. These points after regularisation (i.e. applying the transformation T in section III-D) are plotted as (v1t,v2t,v3t). Regularisation is further exemplified by the red circle with the same radius from the point p(v1). The circle represents hypothetical distributions an assessed vertex can take. Their location only depends on their distributions disregarding similarity to other baseline vertices. By regularisation (i.e. applying the same transformation T), the red circle, becomes the purple ellipse on the triangle P(uP1, uP2, uP3). The triangle P represents the latent space. The purple ellipse takes baseline vertex distributions (and also topology) into account (see also Fig.2) as each point is expressed in the latent space as a linear combination of baseline vertices. The same procedure is applied to any assessed graph for its vertices i.e. the transformation T is applied, and the distance from the labeled normal vertex serves as a change measure. Having the same baseline, allows comparison also between assessed graphs.

The model benefits from the diversity of the normal baseline graph. The transformation T modifies R^n space in order to relate each assessed vertex to the normal baseline expressing its latent representation as a linear combination of the baseline vertices.

The regularisation of normal vertex representations aims to achieve the best in the least square sense orthonormal vectors in R^n also making them reside on the same bounded hyper-plane. Likewise, the assessed vertices reside on the same hyper-plane, and their distributions (also carrying topological information) are expressed in relation to the regularised normal vertex representations either as a relative change (latent representations in the area of the triangle N(v1t, v2t,v3t)) or projected (the area on the triangle P(uP1, uP2, uP3) excluding the triangle N).

Normalisation flattens the latent space as (1) defines a hyper-plane. However, adding edges (i.e increasing egress vertex degree) decreases probabilities due to normalisation, also reducing the distance between vectors, and the scale. To help, regularisation is used for scale adjustment, the level of which depends on vertices' degree difference and weights, and to measure relative change as presented in Fig.2. The closed form solution to matrix factorisations derived in III-D removes approximation errors which impact similarity measure for large dissimilarities in [7]. Regularisation also helps to incorporate exact relation to all the baseline vertices with their topology and weights distribution in embeddings helping the measure finer changes, which is absent in [7]. The metrics used for comparison are discussed in section III-D.

D. Mathematical overview

The weighted, annotated (labelled), directed graphs an example of which is presented in Fig.1 can be described by weighted adjacency matrix. Labelled vertices are ordered and the same order is applied when constructing adjacency matrices. The weights as distribution of data volumes represent conditional probability of transition to vertex n, given vertex k and traffic type t.

$$w_{kn} = P(v_n|_{v_k,t}), for\ all\ k \sum_n w_{kn} = 1 \tag{1}$$

The problem addressed is how to compare systems represented by graphs each described by the corresponding adjacency matrix $\mathbf{A}$ and $\mathbf{B}$ and perform vertex comparison with respect to incident edges from vertices. Typically to compare matrices a norm (distance measure) needs to be defined. However, it is hard for a norm to reflect graphs' verbosity when it transforms the graph space to another space (for example Hilbert space). This leads to distortion of graph properties in the target space. A desirable property of a norm is to be bounded, to make their use by machines easier with ability to localize anomaly. Frobenius norm is often used to compare matrices, however it is not suitable for graphs described by adjacency matrices, because of its local (vertex) scope, indifference to permutation of rows or poor discrimination of edge permutations for scarcely connected vertices to mention a few. To address this a norm with a graph scope is desirable. In order to achieve it, it is proposed to factorise graph adjacency matrices to make graph comparison feasible. There are many matrix pairs **(V, Z)** which can factorize an adjacency matrix. When comparing graph A and graph B, there are four unknown matrices $\mathbf{V_A}, \mathbf{Z_A}, \mathbf{V_B}, \mathbf{Z_B}$ which factorise two adjacency matrices $\mathbf{A}$ and $\mathbf{B}$ as presented in (2).

$$\begin{aligned} \mathbf{V_A Z_A} &= \mathbf{A} \\ \mathbf{V_B Z_B} &= \mathbf{B} \end{aligned} \tag{2}$$

In the prior art, the problem of finding matrix $\mathbf{Z}$ (a single matrix factorization problem) has multiple realisations such as principal component analysis (PCA) which changes coordinates base to achieve alignment with the largest variance components. Another approach is to train a neural network auto-encoder which does not impose any restrictions on $\mathbf{Z}$ if the outputs match the inputs. Some other variants use the regularisation

term in the loss function for the neural network training to obtain desirable features in the latent space. However, neither PCA, nor matrix factorization based on neural networks are suitable due to either computational requirements or their applicability to the graph comparison problem where two matrices needs to be factorised and compared. As matrix multiplication can be seen as rotations and re-scaling of each input vector, the adjacency matrix change for graphs A and graph B can be reflected in matrix V_B by making assumption that $\mathbf{Z_A} = \mathbf{Z_B}$. Lets define metric d1 in (3) where $\|\cdot\|_F$ denotes Frobenius norm, and use it for comparison.

$$\mathbf{d1} = \|\mathbf{V_A} - \mathbf{V_B}\|_\mathbf{F} \tag{3}$$

Metric d1 simplifies when $\mathbf{V_A}$ is orthogonal i.e. d1 becomes only dependant on $\mathbf{A}$, and $\mathbf{B}$. It is further assumed that $\mathbf{V_A}$ is orthonormal with 1s on the diagonal. Matrix $\mathbf{T}$ proposed in this document transforms matrices $\mathbf{A}$ and $\mathbf{B}$ to make $\mathbf{V_A}$ orthonormal so that matrix $\mathbf{V_B}$ reflects graph B in relation to graph A. The matrices $\mathbf{V_A}$ and $\mathbf{V_B}$ are approximated in the least square sense by using Moore-Penrose pseudo inverse when calculating $\mathbf{T} = \mathbf{A}^+$(this is to be solvable when matrix $\mathbf{A}$ is singular). Having found matrix $\mathbf{T}$, the same transformation can be applied to matrix $\mathbf{B}$, and the norm **d1** is further modified in (4).

$$\begin{aligned}\mathbf{d1} &= \|\mathbf{AT} - \mathbf{BT}\|_\mathbf{F} \\ &= \|\mathbf{A_T} - \mathbf{B_T}\|_\mathbf{F}\end{aligned} \tag{4}$$

After this operation, n$^{\text{th}}$ row vector of matrix $\mathbf{B_T}$, $\mathbf{b_{Tn}}$ is comparable with the corresponding row vector from the orthonormal matrix $\mathbf{A_T}$, and graph space is converted to vector space which is measurable. Metric $\mathrm{d1_n}$ is defined for n$^{\text{th}}$ peer vertices in (5).

$$\begin{aligned}&\mathbf{d1_n} = \|\mathbf{a_{Tn}} - \mathbf{b_{Tn}}\|_\mathbf{F} \\ &a_{Tn}, b_{Tn} - \text{ n}^{\text{th}} \text{ row vectors of } \mathrm{A_T},\ \mathrm{B_T}\end{aligned} \tag{5}$$

For matrix $\mathbf{B}$, row vectors $\mathbf{b_{Tn}}$ of $\mathbf{B_T}$ yield alignment such as distance measure d1 is bounded by the interval [0..U1] with 0 indicating perfect alignment (similarity). U1 is the upper bound defined in (6), and its value depends on graph A.

$$\begin{aligned}&u1_n = max(\{elem | elem = \|\overrightarrow{a_{Tn}} - \overrightarrow{v_k}T\|,\ for\, all\, k\}) \\ &\overrightarrow{v_k} - \text{ zero vector with 1 at component k} \\ &U1 = \sum_n u1_n\end{aligned} \tag{6}$$

d2 is defined in (7) as cosine similarity measure, where $\mathbf{a_{Tn}}$, $\mathbf{b_{Tn}}$ are n$^{\text{th}}$ row vectors from matrix $\mathbf{A_T}$, $\mathbf{B_T}$ respectively.

$$\mathbf{d2} = \sum_n \mathbf{d2_n}, \mathbf{d2_n}(\overrightarrow{a_{Tn}}, \overrightarrow{b_{Tn}}) = \frac{\overrightarrow{a_{Tn}} \cdot \overrightarrow{b_{Tn}}}{\|\overrightarrow{a_{Tn}}\| \cdot \|\overrightarrow{b_{Tn}}\|} \tag{7}$$

d2 is bounded by the interval [U2..N] with N indicating perfect alignment (similarity), where N is the number of vertices. U2 is the lower bound defined in (8) and its value depends on graph A.

$$\begin{aligned}&u2_n = min(\{elem | elem = \frac{\overrightarrow{a_{Tn}} \cdot \overrightarrow{v_k}T}{\|\overrightarrow{a_{Tn}}\| \cdot \|\overrightarrow{v_k}T\|},\ for\, all\, k\}) \\ &\overrightarrow{v_k} - \text{ zero vector with 1 at component k} \\ &U2 = \sum_n u2_n\end{aligned} \tag{8}$$

For data pattern also metric **d3** is defined in (9) bounded by the interval [0..U3].

$$\begin{aligned}\mathbf{d3} &= \sum_n \mathbf{d3_n}, \mathbf{d3_n} = \arccos(d2_n) \\ \mathbf{U3} &= \sum_n \arccos(u2_n)\end{aligned} \tag{9}$$

Given a base graph, metric d1 has only one extremum (minimum), equal to zero when matrices A and B are the same and increases with the anomaly increase. Likewise d3 metric is zero when matrices A and B are the same and increases with anomaly increase. Local similarity information can be obtained when the corresponding metrics for peer row vectors are compared individually and not included as part of the aggregated measure.

The transformation $\mathbf{T}$ can also be described as a regularisation process in the Hilbert space to produce the orthonormal constellation of vertices of the system with deviation expressed in relation to that system. The regularisation attempts to make the distance between the labelled normal vertices the same. Vertices with the same representation in the Hilbert space, and yet different label are transformed to the same vector due to the least square solution obtained (Moore-Penrose inverse is used). Another way to see this transformation is to separate the vector representations for vertices by moving them to the latent space. Having a predefined shape of the normal pattern represented as a constellation of orthonormal vectors in the latent space, it enables comparisons of the normal graph with the assessed graph. Geometrically, each row vector of $\mathbf{A_T} - \mathbf{B_T}$ resides on the same bounded hyper-plane, as the regularisation transforms graph independent but vertex specific hyper-planes each bounded by (1) to one bounded hyper-plane for the graphs.

1) Relative data change estimate: For a given topology, and the normal data pattern point p(d1,d3), the upper bounds of u1(d1), u3(d3) for metrics d1, d3 can be calculated. This is similar to the calculations of U1, U3 in (6),(9) except the bounds are expressed as the maximum distance from the normal data point p, and not the point (0,0). Vector $\overrightarrow{v_k}$ is defined if edge k exists for vertex n in the assessed graph. Given a topology, the measure showing relative change of d1 and d3 is defined in (10).

$$\begin{aligned}\mathbf{d1_{rel}} &= (d1 - d1_{normal})/u1(d1_{normal}) \\ \mathbf{d3_{rel}} &= (d3 - d3_{normal})/u3(d3_{normal})\end{aligned} \tag{10}$$

E. Implementation and scalability

The graph comparison method requires calculation of the Moore-Penrose pseudo inverse matrix $\mathbf{A}^+$, and vector multiplications by $\mathbf{A}^+$. The vectors are row vectors from the adjacency matrix which are typically sparse. $\mathbf{A_T}$ has non zero elements on the diagonal for non-singular matrices, and has small memory and storage footprint. For singular matrices, only rows which are not linearly independent have more than one component greater than zero. For Moore-Penrose pseudo inverse calculation, the main computational effort is in singular value decomposition. A^+ is calculated once, which can be done offline and stored locally. However, if the calculation needs to be done at the edge, matrix multiplication and singular value decomposition can be scaled depending on computational resources available and efficiently computed in general purpose hardware with moderate computational resources [13], [14].

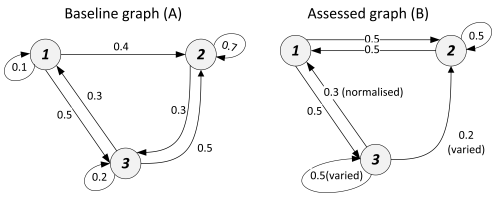

Fig. 4. Graphs for comparison (example)

TABLE I
TOPOLOGY MEASURE (EXAMPLE)

Vertex n	1	2	3
$d1_n$	1.4142	1.4576	0
$d2_n$	0	-0.0163	1

The computation of normal patterns involves multiplication of typically sparse vectors by $\mathbf{A}^+$ and metrics calculation which also can be done offline if needed, and stored locally. An assessment process requires only calculation of the graph's hash H, metrics d1, d3 and normal pattern retrieval from the local storage for comparison. The algorithm has $\mathcal{O}(n^2)$ space complexity. In terms of time complexity, it is is proportional to $\mathcal{O}(n^2)$. This can be improved by breaking down WSN modelled as an l-vertex graph to sub-networks (sub-graphs) with k-vertices which reduces space and time complexity by the factor $(k/l)^2$ for each sub-network (not done in this contribution).

IV. SYNTHETIC GRAPHS EXAMPLE

A system state modelled by two synthetic graphs is presented in Fig.4. The baseline graph A and the assessed graph B are compared as discussed in section III-D.

For topology pattern change detection, incident edges from vertex are ordered and labeled with the values obtained from a uniform pseudo random generator defined for (0..1] (the generator is re-initialised with the same seed for each vertex). The edge ordering follows vertex ordering in the adjacency matrix. The values are normalised per vertex to satisfy (1). Metrics $d1_n$ and $d2_n$ are calculated for vector representation of peer 'n' vertices (i.e. the vertices with the same label 'n') from the baseline and assessed graphs. For vertex *n* topology discrimination, a hash value is calculated using SHA256 algorithm for the sequence of ($d1_n$, $d2_n$). To produce a graph level topology hash, a sequence of vertex hash values are hashed again using SHA256 algorithm. The sequence is ordered based on vertex ordering in the adjacency matrix. For the graphs in Fig.4, metrics $d1_n$, $d2_n$ for the peer vertices used for topology hash calulations are presented in Table I.

To measure data pattern change, the weighs of the assessed graph were varied for the edge (3,3) and (3,2) and compared against the normal (baseline) graph. This is visualized for vertex 3 in Fig. 5,6 where $\|\overrightarrow{a_{T3}} - \overrightarrow{b_{T3}}\|$ and the angle Θ [rad] between vector representation $\overrightarrow{a_{T3}}$ and $\overrightarrow{b_{T3}}$ are plotted. Zero indicates perfect similarity. Metrics $d1, d3$ for vertex 1 and vertex 2 remain unchanged.

For the baseline graph in Fig.4 the metrics bounds for topology and data are presented in Table II.

Acquiring fingerprints of the normal system operation in sets {d1,d3,H}, allows further processing to detect anomaly and measure relative data change.

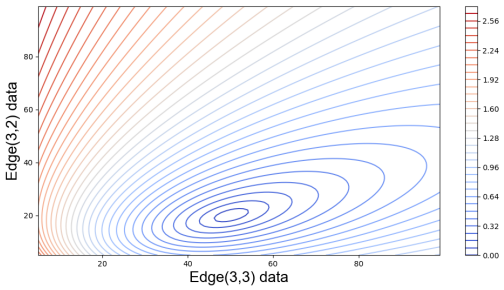

Fig. 5. Vectors length difference (d1) for vertex 3

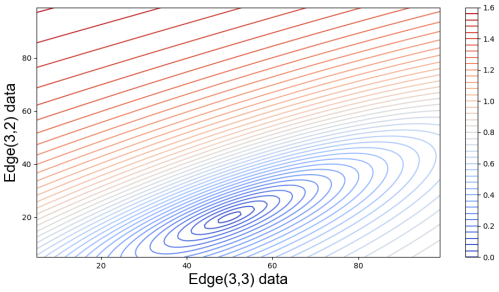

Fig. 6. Angle θ (d3) for vertex 3

A. Relative data pattern change

To demonstrate relative data pattern change, a hypothetical scenario is considered where data patterns are measured at time t_1, and t_2. Only weights w_{33} and w_{32} changed at vertex 3 of the assessed graph in Fig.4. The data pattern and relative change at t2, t1 are presented in Table III and in (11) respectively.

$$\begin{aligned} \Delta d1 &= (d1(t2) - d1(t1))/u1(d1(t1)) = 0.041 \\ \Delta d3 &= (d3(t2) - d3(t1))/u3(d3(t1)) = 0.0226 \end{aligned} \tag{11}$$

Metrics d1, d3 carry topology and data pattern information. To measure data patterns change given assessed and normal topology, the topology component needs to be discriminated i.e. assessed topology at t_1, $T(t_1)$ is the same as at t_2. Otherwise the metrics contain the aggregated topology and data measure, and the change may be caused also by the topology component. Topology hash values for $T(t_1)$ and $T(t_2)$ for the assessed graph given the baseline graph are the same.

TABLE II
METRIC BOUNDS FOR TOPOLOGY AND DATA (EXAMPLE)

Vertex n	1	2	3
$Td1_n$	[0..9.971]	[0..10.361]	[0..11.317]
$Td2_n$	[-0.708..1]	[-0.018..1]	[-0.675..1]
$Dd1_n$	[0..4.175]	[0..4.751]	[0..3.742]
$Dd3_n$	[0..2.165]	[0..2.165]	[0..1.878]

TABLE III
DATA PATTERN CHANGE (EXAMPLE)

Vertex	Time	Edge weights
3	t1	[0.3,0.2,0.5]
3	t2	[0.3,0.1,0.6]

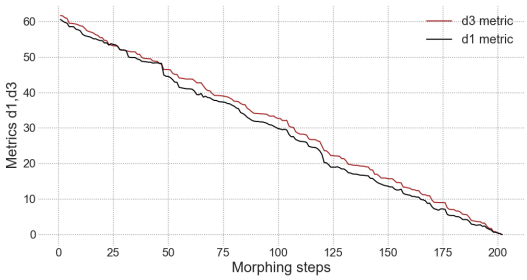

Fig. 7. Graph morphing steps- d1, d3 metrics

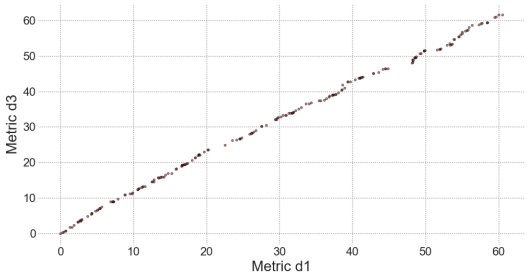

Fig. 8. Graph morphing - (d1, d3) points

B. Topology patterns discrimination

For topology discrimination, the baseline graph in Fig.4 was used to discriminate assessed graph topology change. All possible target topology patterns excluding the patterns containing vertices with no egress edge were evaluated. All $(2^3 - 1)^3 = 343$ target topology patterns given the baseline graph in Fig.4 produced different topology hash H values. This was further extended to change the base graph which produced 3894 collisions. There were 343*342 = 117306 different (normal,assessed) topology pairs excluding the pairs when graphs are the same. The collisions occurred only when a 3-vertex base graph had one edge for two or more vertices, or the graph was symmetrical. Larger graphs provide more diversity.

V. Random graphs example

For further demonstration, two random directed graphs were generated, each containing 40 vertices, with random topology pattern limited to 100 edges. Edge weights were randomly generated from [1..2000] and normalised. The graph A was morphed to become the base graph B. The morphing step was to remove a randomly selected edge from graph A and add randomly selected edge from graph B. After each morphing step, the morphed graph A is compared with the base graph B and metrics d1, d3 are calculated. d1, d3 change at each morphing step as plotted in Fig.7,8. When d1, d3 become zero, graphs A and B are the same. d1 and d3 are correlated with pearson correlation coefficient r = 0.99888 in this example. However, metrics d1, d3 allow finer data pattern detection when d1 or d3 alone does not allow sufficient discrimination (e.g. in Fig.5, there are different target graphs for which d1 metric defined as a distance measure is the same). Anomaly detection in WSNs for data and topology patterns change is demonstrated in [12].

VI. Conclusion

The method allows comparing the systems modelled as graphs to detect topology and data patterns change. The change is represented as a set of numbers, providing fingerprints of topology and data patterns which can be further analysed. Metrics were proposed to quantify the change. The size of the graph can be varied depending on computational resources available such as memory and processing power helping scalability. The method is better suited for sparse networks. As the network becomes densely connected, relative traffic distribution variations become smaller to the point when it is comparable to data acquisition uncertainty. However, it can be circumvented by defining sub-graphs with a subset of vertices and edges also helping scalability. The sub-graphs can be selected based on criteria such as geographical/topological proximity, traffic volumes, or probability which is left for future study.

References

[1] S. N. Matheu, J. L. Hernandez-Ramos, and A. F. Skarmeta, "Toward a cybersecurity certification framework for the internet of things," *IEEE Security Privacy*, vol. 17, no. 3, pp. 66–76, 2019.

[2] J.-Y. Yu, E. Lee, S.-R. Oh *et al.*, "A survey on security requirements for WSNs: Focusing on the characteristics related to security," *IEEE Access*, vol. 8, pp. 45 304–45 324, 2020.

[3] P. Yanardag and S. V. Vishwanathan, "Deep graph kernels," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, vol. 2015-August. Association for Computing Machinery, aug 2015, pp. 1365–1374.

[4] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in *Proceedings of the 5th International Conference on Learning Representations*, ser. ICLR '17, 2017.

[5] Z. Tong, Y. Liang, C. Sun *et al.*, "Directed graph convolutional network," *CoRR*, vol. abs/2004.13970, 2020. [Online]. Available: https://arxiv.org/abs/2004.13970

[6] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 701–710.

[7] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 855–864.

[8] Y. Zhao, Z. Liu, and M. Sun, "Representation learning for measuring entity relatedness with rich information," in *Proceedings of the 24th International Conference on Artificial Intelligence*, ser. IJCAI'15. AAAI Press, 2015, p. 1412–1418.

[9] A. Jinwon and C. Sungzoon, " Variational autoencoder based anomaly detection using reconstruction probability," SNU Data Mining Cente, Tech. Rep., 2015.

[10] A. A. Pol, V. Berger, C. Germain *et al.*, "Anomaly detection with conditional variational autoencoders," in *18th IEEE International Conference On Machine Learning And Applications, ICMLA 2019, Boca Raton, FL, USA, December 16-19, 2019*. IEEE, 2019, pp. 1651–1657.

[11] A. Brandt, J. Hui, R. Kelsey *et al.*, "7/17/2015 RFC 6550 RPL: IPv6 Routing Protocol for LowPower and Lossy Networks," pp. 1–314, 2015.

[12] R. Zakrzewski, T. Martin, and G. Oikonomou, "Anomaly detection of data and topology patterns in WSNs," in *Proc. 17th International Conference on Distributed Computing in Sensor Systems*, jul 2021.

[13] V. Demchik, M. Bačák, and S. Bordag, "Out-of-core singular value decomposition," jul 2019. [Online]. Available: http://arxiv.org/abs/1907.06470

[14] P. Blanchard, M. Zounon, J. Dongarra, and N. Higham, "D2.9 Novel SVD Algorithms," pp. 1–29, 2019. [Online]. Available: https://www.nlafet.eu/wp-content/uploads/2019/04/D2-9-SVDalgorithms-final.pdf