CASPITA: Mining Statistically Significant Paths in Time Series Data from an Unknown Network

Andrea Tonon
Department of Information Engineering
University of Padova
Padova, Italy
andrea.tonon@dei.unipd.it

Fabio Vandin
Department of Information Engineering
University of Padova
Padova, Italy
fabio.vandin@unipd.it

Caspita: italian exclamation indicating surprise,
e.g., "Caspita! Such significant paths are really surprising."

***Abstract*—The mining of time series data has applications in several domains, and in many cases the data are generated by networks, with time series representing paths on such networks. In this work, we consider the scenario in which the dataset, i.e., a collection of time series, is generated by an *unknown* underlying network, and we study the problem of *mining statistically significant paths*, which are paths whose number of observed occurrences in the dataset is unexpected given the distribution defined by some features of the underlying network. A major challenge in such a problem is that the underlying network is unknown, and, thus, one cannot directly identify such paths. We then propose CASPITA, an algorithm to mine statistically significant paths in time series data generated by an unknown and underlying network that considers a generative null model based on meaningful characteristics of the observed dataset, while providing guarantees in terms of false discoveries. Our extensive evaluation on pseudo-artificial and real data shows that CASPITA is able to efficiently mine large sets of significant paths, while providing guarantees on the false positives.**

***Index Terms*—Pattern Mining, Statistically-Sound Pattern Mining, Time Series, Graph Mining**

I. INTRODUCTION

Time series data mining [1]–[3] is a fundamental data mining task that covers a wide range of real-life problems in various fields of research. Even if the common purpose is to extract meaningful knowledge from the data, many different problems and approaches have been proposed over the years, ranging from anomaly detection [4], [5] to motif discovery [6], from clustering [7] to classification [8]. However, in many real life scenarios, time series data are generated by networks, and thus represent paths constrained by the structures and the distributions that define such networks. Very often, one has access to a collection of time series but does not know the distribution on the network that generated them, or the structure of such network. As an example, consider a survey on the paths traveled by people with the underground service of a given city. In such a scenario, one has a dataset that represents a limited number of paths from a network, defined by the underground structure, but does not know the distribution defined by the entire population that uses such service.

In this work, we study the problem of mining statistically significant paths from an unknown network. We assume to have a time series dataset, defined as a collection of time series, and that such time series are paths generated from an unknown network. In such a scenario, we are interested in mining unexpected paths from the dataset. Standard techniques usually use the frequency or the number of occurrences as extraction criteria, with the aim of finding interesting paths, but, in many real applications, such metrics are not enough to find paths that provide useful knowledge. For example, paths that appear only few times in a dataset may be over represented if we consider the distribution of the network underlying the data, or vice-versa, paths that appear a lot of times may be under represented. Thus, techniques based on such metrics may led to several spurious discoveries. In addition, since we do not know the network underlying the data, we can not directly find over or under represented paths.

We then introduce CASPITA, an algorithm to find statistically significant paths over (or under) represented from time series data considering a generative null model based on meaningful characteristics of the observed dataset, while providing guarantees in terms of the false positives employing the Westfall-Young (WY) method. Our generative null model is based on the observed number of occurrences of paths of a given length, with the idea that such paths represent well-known substructures of the underlying network. In the simplest case, they are paths of length one, that are edges of the underlying network. Then, such null model is used to test the significance of paths of a given, higher, length, which are the paths of interest mined from the observed dataset. The intuition is to create a generative null model that is able to explain the number of occurrences of shorter paths, and to check whether such generative null model is able to also explain the observed number of occurrences of the paths of interest. Otherwise, such paths can be considered significant, in the sense that they appear more (or less) times than expected under such generative null model. Let us consider, as example, a network composed by the web-pages of a website, and suppose that we want to find sequences of web-pages visited more (or less) than expected with respect to the underlying distribution of the network, defined by the navigation of the users on the website. Given the application or the structure

of the network, there may be some well-know substructures, defined as short sequences of web-pages, that are traversed by the users that visit such website with a particular distribution. Again, let us note that in the simplest case, the substructures are paths of length one, that represent a direct link between two web-pages. Thus, in such a scenario, one may be interested in finding if such substructures also explain the number of observed occurrences of longer paths, or if such longer paths are significant due to some external factors causing their number of occurrences.

A. Our Contributions

In this work, we introduce the problem of mining *statistically significant paths* in time series data from an unknown network. In this regard, our contributions are:

- We introduce the problem of mining statistically significant paths in time series data from an unknown network, defining a generative null model based on meaningful characteristics of the observed dataset;
- We introduce CASPITA, an algorithm to mine statistically significant paths (over or under represented) from a time series dataset, while providing guarantees on the probability of reporting at least one false positive;
- We introduce an alternative interesting scenario in which CASPITA can be applied, which consists in mining paths that are significant with respect to a null model based on data from a different dataset;
- We perform an extensive suite of experiments that demonstrates that CASPITA is able to efficiently mine statistically significant paths in real datasets while providing guarantees on the false positives.

Let us note that throughout the paper, we only describe the scenario in which one is interested in mining statistically significant paths that occur more times than expected under the null hypothesis (*over represented paths*), for clarity of presentation. However, all the reasoning are still valid to mine paths that occur less times than expected (*under represented paths*). In particular, our open source implementation of CASPITA mines over or under represented paths, and results for both scenarios are shown in the experimental evaluation.

B. Related Works

We now discuss the relation of our work to prior art on significant pattern mining, anomaly detection in sequential data, and temporal anomaly detection in graphs, which are the areas most related to our work. Since the nature of our work, we only consider unsupervised approaches.

In significant pattern mining, the dataset is seen as a collection of samples from an unknown distribution, and one is interested in finding patterns significantly deviating from an assumed *null hypothesis*, i.e., *distribution*. Many variants and algorithms have been proposed for the problem. We point interested reader to the survey [9], and recent works that employ permutation testing [10]–[12]. Even if our work falls within the framework of significant pattern mining, such approaches are orthogonal to our work, which focuses on finding significant paths, i.e., patterns, from time series that are constrained by a network structure.

Many works have been proposed to detect anomalies in sequential data [4], [13], employing several definition of anomalies, and considering different types of patterns. For example, [5] defines a pattern as *surprising* if its frequency differs substantially from that expected by chance, given some previously seen data. Lemmerich et al. [14], instead, considers the mining of subgroups, defined by subsets of attributes, that exhibit exceptional transition behavior, i.e., induce different transition models compared to the ones of the attributes that describe the entire dataset. Although our approach adopts a definition of significant pattern based on how the number of its occurrences differs from the one expected under an appropriate model, similarly to other works, we consider the setting in which the data represent paths from a weighted and directed graph, which results in a different problem. In fact, this aspect makes our work closer to the task of detecting anomalies in temporal graph [15], [16], i.e., graphs that evolve over time. However, even if our work considers data generated by a network and aims to find paths whose number of occurrences is significant with respect to the network's distribution, we consider the scenario in which we do not know the network, and we have only access to a sample.

The only work that considers the problem of finding anomalous paths in time series data from an unknown network is [17]. In this work, the authors propose an algorithm, HYPA, to find anomalous length k paths using a null model based on length $k-1$ paths. In particular, they aim to find length k paths whose number of occurrences in a dataset is anomalous with respect to a null model based on the number of occurrences of length $k-1$ paths in the same dataset. Reducing the difficult problem of detecting anomalous length k paths to the easier problem of detecting anomalous edges in a k-th order De Bruijn graph, they describe a strategy based on the hypergeometric distribution to compute a score for each length k path, where the score describes the level of anomaly of such a path. Even if our approach is inspired by [17], our work differs from it in many key aspects. First of all, we aim to find length k paths whose number of occurrences in a dataset is significant with respect to a null model based on the number of occurrences of length h paths, with $h \in \{1, \dots, k-1\}$ provided in input by the user, and not only with $h = k-1$ as in [17]. In such a direction, it is not clear if HYPA can be modified to consider a more general length $h \in \{1, \dots, k-1\}$. Finally, while our approach employs the WY method to correct for multiple hypothesis testing providing guarantees in terms of false positives, [17] uses fixed thresholds to define interesting patterns, which does not provide any guarantee.

To the best of our knowledge, our work is the first approach that employs the statistical hypothesis testing framework to mine paths, i.e., patterns, from time series constrained by the structure and the distribution of an unknown network, while providing rigorous guarantees on the probability of reporting at least one false positive, i.e., the FWER.

II. Preliminaries

We now provide the definitions and concepts used in the paper. First, in Section II-A, we describe the task of mining paths in time series data from a network. Then, in Section II-B, we define, similarly to [18], the concept of k-th order De Bruijn graph used in the paper to define our generative null model. Finally, in Section II-C, we describe concepts of hypothesis and multiple hypothesis testing for paths.

A. Mining Paths in Time Series Data from a Network

Let us define a *network* $N = (G, \omega)$ as a *directed graph* $G = (V, E)$ and a *weight function* $\omega : E \to [0, 1]$. $V = \{v_1, v_2, \ldots, v_{|V|}\}$ is the *vertices set*, where each $v \in V$ is called *vertex*, and $E = \{(u, v) : u, v \in V\}$ is the *edges set*, where each (u, v) is an *ordered* pair of vertices, called *edge*. An edge (u, v) is an *incoming edge* of the vertex v and an *outgoing edge* of the vertex u. Denoting with $(u, :)$ an outgoing edge of u, for each vertex $u \in V$, we have $\sum_{(u,:)\in E} \omega((u, :)) = 1$, that is, the weights of the edges from u represent a probability distribution. Fig. 1 (left) shows an example of network.

A *path*, or *walk*, $w = \{v_{i_0}, v_{i_1}, \ldots, v_{i_{|w|}}\}$ of length $|w|$ on the network N is an ordered sequence of $|w| + 1$ vertices such that $(v_{i_j}, v_{i_{j+1}}) \in E \ \forall j \in \{0, \ldots, |w| - 1\}$. Note that a vertex $v \in V$ is a path $w = \{v\}$ of length $|w| = 0$. A path $w = \{w_0, w_1, \ldots, w_{|w|}\}$ *occurs* in a path $q = \{q_0, q_1, \ldots, q_{|q|}\}$ starting from position $s \in \{0, \ldots, |q| - |w|\}$, denoted by $w \subset q^{(s)}$, if and only if $w_0 = q_s, w_1 = q_{s+1}, \ldots, w_{|w|} = q_{s+|w|}$. We say that the path w is a *sub-path* of the path q. The number of *occurrences* $Occ_q(w)$ of w in q is the number of times that w occurs in q, that is, $Occ_q(w) = |\{s \in \{0, \ldots, |q| - |w|\} : w \subset q^{(s)}\}|$.

A *time series dataset* $\mathcal{D} = \{\tau_1, \tau_2, \ldots, \tau_{|\mathcal{D}|}\}$ from a network N is a bag of $|\mathcal{D}|$ *transactions*, which are paths on N. Given a path w on N, the number of occurrences $Occ_{\mathcal{D}}(w)$ of w in $\mathcal{D}$ is the sum of the number of occurrences $Occ_\tau(w)$ of w in τ, $\forall \tau \in \mathcal{D}$, that is, $Occ_{\mathcal{D}}(w) = \sum_{\tau\in\mathcal{D}} Occ_\tau(w)$.

Given a positive integer ℓ, the task of *mining paths* of length ℓ from a time series dataset $\mathcal{D}$ from a network N is the task of mining the set $\mathcal{W}_{\mathcal{D}}(\ell)$ of all paths of length ℓ that occur at least once in $\mathcal{D}$ and the number of their occurrences, that is,

$$\mathcal{W}_{\mathcal{D}}(\ell) = \{(w, Occ_{\mathcal{D}}(w)) : |w| = \ell \wedge Occ_{\mathcal{D}}(w) > 0\}.$$

With an abuse of notation, in the following we use $w \in \mathcal{W}_{\mathcal{D}}(\ell)$ to indicate that $\exists (w, Occ_{\mathcal{D}}(w)) \in \mathcal{W}_{\mathcal{D}}(\ell)$.

B. k-th Order De Bruijn Graph

Given a directed graph $G = (V, E)$ and an integer $k > 0$, the *k-th order De Bruijn graph* $G^k = (V^k, E^k)$ of G is a directed graph where each vertex $v^k \in V^k$ is a path of length $k - 1$ on G, i.e., $v^k = \{v_{i_0}, v_{i_1}, \ldots, v_{i_{k-1}}\}$, and an ordered pair (v^k, u^k), with $v^k = \{v_{i_0}, v_{i_1}, \ldots, v_{i_{k-1}}\}$, $u^k = \{u_{j_0}, u_{j_1}, \ldots, u_{j_{k-1}}\} \in V^k$, it is an edge of G^k if and only if $v_{i_t} = u_{j_{t-1}} \ \forall t \in \{1, \ldots, k - 1\}$. Thus, each edge $(v^k, u^k) \in E^k$ is a path of length k on G, since $(v^k, u^k) = \{v_{i_0}, v_{i_1} = u_{j_0}, v_{i_2} = u_{j_1}, \ldots, v_{i_{k-1}} = u_{j_{k-2}}, u_{j_{k-1}}\}$. Let us note that G itself is a 1-st order De Bruijn graph of G. Fig. 1 (right) shows an example of k-th order De Bruijn graph.

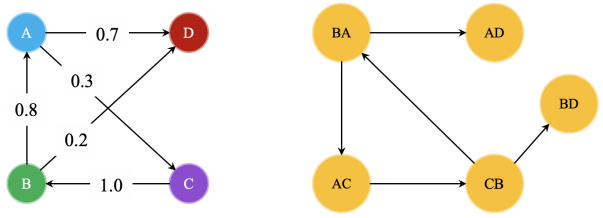

Fig. 1. Example of network and of De Bruijn graph. It shows the network $N = (G, \omega)$ (left) and the 2-nd order De Bruijn graph $G^2 = (V^2, E^2)$ of G (right). The network N is composed by the directed graph $G = (V, E)$, with $V = \{A, B, C, D\}$ and $E = \{(A, C), (A, D), (B, A), (B, D), (C, B)\}$, and by the weight function ω, such that $\omega((A, C)) = 0.3, \omega((A, D)) = 0.7$, $\omega((B, A)) = 0.8$, $\omega((B, D)) = 0.2$ and $\omega((C, B)) = 1.0$. The path $w = CBAC$ is an example of path of length $|w| = 3$ on N. The 2-nd order De Bruijn graph G^2 is composed by $V^2 = \{AC, AD, BA, BD, CB\}$, where each $v^2 \in V^2$ is a path of length 1 on G, and by $E^2 = \{(AC, CB), (BA, AC), (BA, AD), (CB, BA), (CB, BD)\}$, where each $(v^2, u^2) \in E^2$ is a path of length 2 on G.

C. Multiple Hypothesis Testing for Paths

The task of mining statistically significant paths is to identify paths whose number of occurrences in a dataset $\mathcal{D}$ is *significant*, or unexpected, with respect to the distribution of the weight function of the network that generated such data. To assess the significance of a path, we employ the framework of statistical hypothesis testing. For each path w, let H_w be the null hypothesis that the number of occurrences $Occ_{\mathcal{D}}(w)$ of w on $\mathcal{D}$ well conforms to the number of its occurrences in *random* time series data generated from the network $N = (G, \omega)$. That is, data generated from the graph G in accordance with the weight function ω while preserving the starting vertices observed in the original data and that have the same number of paths of interest of $\mathcal{D}$.

Under the null hypothesis, the number of occurrences of w is described by a random variable X_w, and in order to assess the significance of w, a p-value p_w is commonly computed. The p-value p_w of w is the probability of observing a number of occurrences, under the null hypothesis, at least as large as the number of occurrences $Occ_{\mathcal{D}}(w)$ of w in $\mathcal{D}$, that is,

$$p_w = \Pr\left[X_w \geq Occ_{\mathcal{D}}(w) | H_w\right].$$

For complex null hypotheses, the p-values can not be computed analytically, since there is not a closed form for X_w. However, when one can generate random data from the distribution described by the null hypothesis, the p-values can be estimated by a simple Monte Carlo (MC) procedure as follows: to generate M random time series datasets $\tilde{\mathcal{D}}_i$, with $i \in \{1, \ldots, M\}$, from the distribution described by the null hypothesis. Then, the p-value p_w is estimated as

$$p_w = \frac{1}{M+1}\left(1 + \sum_{i=1}^{M} \mathbb{1}\left[Occ_{\tilde{\mathcal{D}}_i}(w) \geq Occ_{\mathcal{D}}(w)\right]\right), \quad (1)$$

where $\mathbb{1}[\cdot]$ is the indicator function of value 1 if the argument is true, and 0 otherwise.

The statistical hypothesis testing framework is commonly used to provide guarantees on the false discoveries, i.e., paths flagged as significant while they are not. When a single path w is tested for significance, flagging w as significant, i.e., rejecting the null hypothesis, when $p_w \leq \alpha$, where $\alpha \in [0,1]$ is the *significance threshold* fixed by the user, guarantees that the probability that w corresponds to a false discovery $\leq \alpha$.

The situation is completely different when several paths are tested simultaneously, as in the case of path mining. If d paths are tested with the approach used for a single path, i.e., each path is flagged as significant if its p-value is $\leq \alpha$, then the expected number of false discoveries can be as large as αd. To solve this issue, one identifies a *corrected significance threshold* $\delta \in [0,1]$ such that all paths with p-value $\leq \delta$ can be reported as significant while providing some guarantees on the number of false discoveries. A common approach is to identify δ that provides guarantees on the Family-Wise Error Rate (FWER), defined as the probability of reporting at least one false positive, that is, if FP is the number of false positives, then FWER $= \Pr[\text{FP} > 0]$. For a given value δ, let FWER(δ) be the FWER obtained when δ is used as corrected significance threshold, that is, by reporting as significant all paths with p-value $\leq \delta$. Often FWER(δ) can not be evaluated in closed form, and thus approaches, as the Bonferroni correction or based on permutation testing described below, must be employed.

The Westfall-Young (WY) method [19] is a multiple hypothesis testing procedure based on permutation testing that results in high statistical power and that has been successfully applied in other pattern mining scenarios [10]–[12]. The WY method directly estimates the joint distribution of null hypotheses using permuted datasets, i.e., datasets obtained from the distribution described by the null hypothesis. In detail, the WY method considers P random datasets $\tilde{\mathcal{D}}_i$, with $i \in \{1,\dots,P\}$, generated from the distribution described by the null hypothesis. Then, for every dataset $\tilde{\mathcal{D}}_i$, with $i \in \{1,\dots,P\}$, it computes the minimum p-value $p^{(i)}_{min}$ over all paths of interest in $\tilde{\mathcal{D}}_i$. The FWER FWER(δ) obtained using δ as corrected significance threshold can then be estimated as

$$\text{FWER}(\delta) = \frac{1}{P}\sum_{i=1}^{P} \mathbb{1}\left[p^{(i)}_{min} \leq \delta\right]. \tag{2}$$

Thus, given a *FWER threshold* $\alpha \in [0,1]$, the corrected significance threshold δ^* is obtained as

$$\delta^* = \max\{\delta : \text{FWER}(\delta) \leq \alpha\}. \tag{3}$$

III. caSPiTa: Mining Statistically Significant Paths

In this section, we describe our method caSPiTa, mining statistiCAlly Significant Paths In Time series dAta, to mine *statistically significant paths* in time series data generated by a network, while controlling the probability of having at least one false discovery, i.e., the FWER. Given a time series dataset $\mathcal{D}$ from an *unknown* network N, we aim to mine statistically significant paths, that are paths that have a number of occurrences on $\mathcal{D}$ that is surprising, i.e., higher than the expected number of their occurrences under the null hypothesis. In particular, given two natural values $k, h \in \mathbb{N}^+$, with $k > h$, we aim to mine length k paths from $\mathcal{D}$ whose number of occurrences are not due to the number of occurrences of length h paths observed in $\mathcal{D}$, with the idea that such paths of length h represent some well-know substructures in the underlying and unknown network.

The idea behind caSPiTa is the following. First, we mine all the paths $\mathcal{W}_{\mathcal{D}}(k)$ of length k from the time series dataset $\mathcal{D}$. Since we do not know the network N from which $\mathcal{D}$ has been generated, we can not directly infer the statistical significance of such paths with respect to N, and thus we need to construct a new generative null model from the dataset $\mathcal{D}$. Such generative null model is then used to estimate the p-values and to compute the corrected significance threshold δ^* using the WY method. We now describe the generative null model employed by caSPiTa.

A. Generative Null Model

In this work, we aim to find length k paths whose number of occurrences in $\mathcal{D}$ are not due to the number of occurrences of length h paths in $\mathcal{D}$. Thus, we create a generative null model in accordance with the number of occurrences of the paths of length h in $\mathcal{D}$, and then we test the significance of the paths of length k using such model. Given a time series dataset $\mathcal{D}$, generated by an unknown network $N = (G, \omega)$, and $h \in \mathbb{N}^+$, we define the *h-th order generative model* $N^h(\mathcal{D})$ of the time series dataset $\mathcal{D}$ as a network $N^h(\mathcal{D}) = (G^h, \omega^h)$, where G^h is the h-th order De Bruijn graph of G (based on $\mathcal{D}$, since the entire structure of G is unknown), and ω^h is a weight function. The h-th order De Bruijn graph $G^h = (V^h, E^h)$ is composed as follows: $V^h = \{w \in \mathcal{W}_{\mathcal{D}}(h-1)\}$, while E^h is constructed as defined in Definition II-B. Thus, each vertex $v^h \in V^h$ is a path of length $h-1$ in $\mathcal{D}$ (and thus on G), while each edge $(u^h, v^h) \in E^h$ represents a path of length h in $\mathcal{D}$ (and thus on G). With an abuse of notation, in the following we use (u^h, v^h) to indicate both the edge in G^h and the corresponding path on G. Finally, the weight function ω^h is defined as follows: $\forall (u^h, v^h) \in E^h$,

$$\omega^h\left((u^h, v^h)\right) = \frac{Occ_{\mathcal{D}}\left((u^h, v^h)\right)}{\sum_{(u^h,:)\in E^h} Occ_{\mathcal{D}}\left((u^h,:)\right)}.$$

Let us note that the weight function ω^h of the h-th order generative null model $N^h(\mathcal{D})$ is defined using the observed number of occurrences of length h paths, and that each edge of $N^h(\mathcal{D})$ represents a path of length h. Thus, $N^h(\mathcal{D})$ correctly represents the distribution of the number of occurrences of the paths of length h in the dataset $\mathcal{D}$. An example of generative null models is shown in Fig. 2.

As explained in Section II-C, to compute the p-values p_w of the paths $w \in \mathcal{W}_{\mathcal{D}}(k)$ and to estimate the corrected significance threshold δ^*, we require random data generated from the generative null model. In the following two sections, we introduce two different strategies to generate random datasets

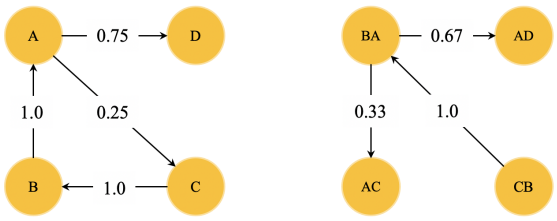

Fig. 2. Example of generative null models. It shows $N^1(\mathcal{D})$ (left) and $N^2(\mathcal{D})$ (right), respectively the 1-st and the 2-nd order generative null models of $\mathcal{D} = \{\tau_1 = BAD, \tau_2 = CBAC, \tau_3 = CBAD, \tau_4 = AD\}$, which is a possible time series dataset from the network N shown in Fig. 1. Let us note that the two generative models represent different probability distributions for the paths of length $> h$ and that they are based on the dataset $\mathcal{D}$ and not on the network N that generated $\mathcal{D}$. Indeed, they have some missing edges with respect to the network N (that is a 1-st order De Bruijn graph of itself) and its 2-nd order De Bruijn graph G^2, respectively, both shown in Fig. 1.

$\tilde{\mathcal{D}}$ that contain the same total number T of paths of length k of the original dataset $\mathcal{D}$, that is,

$$T = \sum_{w \in \mathcal{W}_{\mathcal{D}}(k)} Occ_{\mathcal{D}}(w) = \sum_{w \in \mathcal{W}_{\tilde{\mathcal{D}}(k)}} Occ_{\tilde{\mathcal{D}}}(w). \qquad (4)$$

First, we describe the *transactions oriented generation* (TOG) strategy, a natural way to generate random datasets performing a series of random walks that generate random transactions with characteristics similar to the ones of the transactions in $\mathcal{D}$. To overcome some issues of this strategy when it is applied to large generative null model, we then introduce the *paths oriented generation* (POG) strategy, an alternative approach that directly generates random length k paths. For this second strategy, we also introduce an approximation based on the binomial distribution that allows to estimate the p-values avoiding expensive MC procedures.

B. Transactions Oriented Generation (TOG) Strategy

In this section, we explain how to generate random datasets $\tilde{\mathcal{D}}$ from the generative null model $N^h(\mathcal{D})$ defined above using the TOG strategy. The idea is to perform a series of random walks that generate random transactions $\tilde{\tau}$ with characteristics similar to the ones of the transactions $\tau \in \mathcal{D}$. Characteristics that are natural to consider and that we want to preserve are: i) the dataset $\tilde{\mathcal{D}}$ has the same number of transactions of $\mathcal{D}$; ii) each transaction $\tilde{\tau} \in \tilde{\mathcal{D}}$ has the same length of the corresponding transaction $\tau \in \mathcal{D}$; iii) each transaction $\tilde{\tau} \in \tilde{\mathcal{D}}$ starts from the same vertex (of the generative null model) of the corresponding transaction $\tau \in \mathcal{D}$. Let us note that to preserve such characteristics guarantees to preserve also the total number T of length k paths in the dataset. As a motivation to consider such characteristics, let us consider the case in which the dataset $\mathcal{D}$ contains transactions that represent visits of some users in a website. In such a scenario, we are interested in preserving the web-pages from which the visits start, since they probably are homepages (or web-pages from which the users typically start their navigation). In addition, by preserving the length of the transactions, we preserve the number of web-pages that the users visit in a single navigation on the website.

Let s_τ be a path of length $|s_\tau| = h-1$ such that $s_\tau \subset \tau^{(0)}$, that is, s_τ is the vertex of $N^h(\mathcal{D})$ from which the transaction τ starts. The TOG strategy is the following. For each $\tau_i \in \mathcal{D}$, we perform a random walk on $N^h(\mathcal{D})$ of $|\tau_i| - (h-1)$ steps, starting from the vertex s_{τ_i}. At each step, the random walk moves from a vertex v^h to a vertex u^h with probability $\omega^h\left((v^h, u^h)\right)$. The path generated from such random walk is then the transaction $\tilde{\tau}_i \in \tilde{\mathcal{D}}$ that corresponds to the transaction $\tau_i \in \mathcal{D}$, with $|\tau_i| = |\tilde{\tau}_i|$. Performing all the $|\mathcal{D}|$ random walks, we generate the random dataset $\tilde{\mathcal{D}}$. Let us note that a random walk may reach a vertex without outgoing edges before performing the desired number of steps, generating a shorter transaction, and thus not preserving the second characteristic (and neither T). In such a case, we discard the transaction and repeat the random walk until we generate a transaction $\tilde{\tau}_i$ with $|\tilde{\tau}_i| = |\tau_i|$.

The TOG strategy is the most natural way to generate random data from the generative null model. However, when the generative null model is large, as happens in many real applications, the number of paths contained in a dataset is only a small fraction of the gargantuan number of paths that can be generated as sub-paths of such long transactions. Thus, the corrected significance threshold δ^* obtained with the WY method could be very small, resulting in few or even zero reported significant paths. In addition, depending on the structure of the generative null model, to generate such long transactions may be computationally expensive for the high number of transactions that we need to generate and discard before reaching the desired lengths.

C. Paths Oriented Generation (POG) Strategy

To overcome the issue of the TOG strategy, we now describe an alternative approach to generate random data. In the POG strategy, instead of generating long transactions, we generate single random paths w of length $|w| = k$. In particular, from the generative null model $N^h(\mathcal{D})$ defined above, we generate random datasets $\tilde{\mathcal{D}}$, which are bags of paths of length k, where the number of paths w of length $|w| = k$ that start in each vertex (of the generative null model) is the same in the two datasets $\mathcal{D}$ and $\tilde{\mathcal{D}}$. Let us note that to preserve such characteristic guarantees to also preserve the total number T of length k paths in the dataset.

Let us remember that s_w is a path of length $|s_w| = h-1$ such that $s_w \subset w^{(0)}$, that is, s_w is the vertex of $N^h(\mathcal{D})$ from which the path w starts, and let $\mathcal{S} = \{s_w : w \in \mathcal{W}_{\mathcal{D}}(k)\}$ be the set of vertices of $N^h(\mathcal{D})$ from which starts at least one path $w \in \mathcal{W}_{\mathcal{D}}(k)$. To generate random paths, for each vertex $s \in \mathcal{S}$, we perform a series of random walks of $k-(h-1)$ steps on $N^h(\mathcal{D})$, until we generate

$$n_s = \sum_{w \in \mathcal{W}_{\mathcal{D}}(k): s_w = s} Occ_{\mathcal{D}}(w) \qquad (5)$$

random paths w of length $|w| = k$ that start from such vertex s. Then, the bag of all the paths of length k generated from all the vertices $s \in \mathcal{S}$ is the random dataset $\tilde{\mathcal{D}}$. Let us note that $|\tilde{\mathcal{D}}| = \sum_{s \in \mathcal{S}} n_s = T$. As explained above, let us remember

that at each step the random walk moves from a vertex v^h to a vertex u^h with probability $\omega^h((v^h, u^h))$. Thus, each random walk generates a path of length k or a path of length $< k$ that ends in a vertex without outgoing edges. Since we are interested in paths of length k, we discard all generated paths of length shorter than k.

While the POG strategy overcomes the issue of the TOG strategy explained above reducing the space of paths that can be generated from the generative null model, it still requires expensive MC procedures to estimate the p-values, and such procedures could be computationally prohibitive for large datasets. In the following section, we introduce a method to approximate the p-values for the POG strategy avoiding the MC procedure.

1) Binomial Approximation for the p-values: In this section, we illustrate an approach to approximate the p-values p_w of paths w of length $|w| = k$ when the POG strategy is used to generate random data. First, we compute with which probabilities such paths are generated under the POG strategy. Let us consider a random walk that starts from a vertex $s \in \mathcal{S}$ and that performs $k - (h-1)$ steps on $N^h(\mathcal{D})$, and let $\mathcal{W}_s$ be the set of all paths that can be generated by such random walk. As explained above, the set $\mathcal{W}_s$ contains paths of length $|w| = k$ and, eventually, paths of length $|w| < k$ that end in a vertex without outgoing edges. Let $RW(w)$ be the set of edges of $N^h(\mathcal{D})$ that the random walk traverses to generate the path $w \in \mathcal{W}_s$. From the definition of random walk, the probability $\Pr(w)$ that the random walk generates $w \in \mathcal{W}_s$ starting from s is

$$\Pr(w) = \prod_{(u^h, v^h) \in RW(w)} \omega^h\left((u^h, v^h)\right).$$

Let us note that $\sum_{w \in \mathcal{W}_s} \Pr(w) = 1$. Let E_k be the event that the random walk generates a path of length exactly k and let $\mathcal{W}_s^k \subseteq \mathcal{W}_s$ be the set of paths $w \in \mathcal{W}_s$ with $|w| = k$. Since in the POG strategy we discard paths shorter than k which could be generated by the series of random walks, then the probability of generating the path $w \in \mathcal{W}_s^k$ is

$$\Pr(w \mid E_k) = \frac{\Pr(w \cap E_k)}{\Pr(E_k)}, \tag{6}$$

where $\Pr(w \cap E_k) = \Pr(w)$ for all $w \in \mathcal{W}_s^k$ and 0 otherwise, and $\Pr(E_k) = \sum_{w \in \mathcal{W}_s^k} \Pr(w)$. Again, let us note that $\sum_{w \in \mathcal{W}_s^k} \Pr(w \mid E_k) = 1$, and that if $\mathcal{W}_s \backslash \mathcal{W}_s^k = \emptyset$, then $\Pr(w \mid E_k) = \Pr(w)$. An example of these probabilities is shown in Fig 3.

Since from a given vertex $s \in \mathcal{S}$, we generate exactly n_s (see Equation 5) length k paths, then the number of occurrences $Occ_{\tilde{\mathcal{D}}}(w)$ of a path $w \in \mathcal{W}_s^k$ in the random dataset $\tilde{\mathcal{D}}$ follows a binomial distribution, that is, $Occ_{\tilde{\mathcal{D}}}(w) \sim \text{Bin}(n_s, \Pr(w \mid E_k))$. For a fixed vertex $s \in \mathcal{S}$, this is true for all the paths $w \in \mathcal{W}_s^k$, but the binomial distributions corresponding to these paths are not independent, and thus, the computation of the p-values p_w as

$$p_w = \Pr\left[\text{Bin}(n_s, \Pr(w \mid E_k)) \geq Occ_{\mathcal{D}}(w)\right] \tag{7}$$

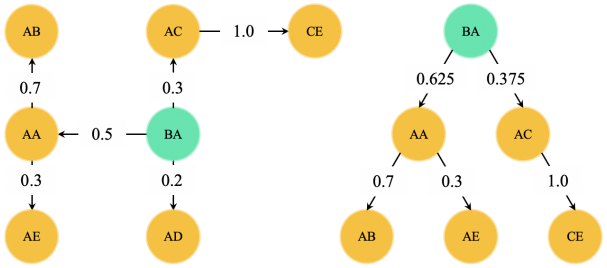

Fig. 3. Example of paths oriented generation. It shows the 2-nd order generative null model $N^2(\mathcal{D})$ and the starting vertex BA (left), and the probabilities of all paths of length 3 under the POG strategy (right). For $k = 3$, starting from the vertex BA and performing $k - (h-1) = 2$ steps, a random walk can generate the paths of length 3 $BAAE$, $BAAB$ and $BACE$, or can reach the vertex AD just after one step, generating the path of length 2 BAD. The probabilities of all these paths are: $\Pr(BAAE) = 0.15$, $\Pr(BAAB) = 0.35$, $\Pr(BACE) = 0.3$, and $\Pr(BAD) = 0.2$. Instead, their probabilities under the POG strategy are: $\Pr(BAAE \mid E_3) = 0.1875$, $\Pr(BAAB \mid E_3) = 0.4375$, and $\Pr(BACE \mid E_3) = 0.375$.

considers a number of paths that is in *expectation* n_s, and not exactly n_s as for the original POG strategy. (Note that, as a consequence, the *total* number of considered paths of length k is T *in expectation*.) However, in our experimental evaluation, we empirically show that the p-values for the binomial approximation are within one order of magnitude of the corresponding MC p-values, and, thus, that the binomial approximation is a valid approach to approximate the p-values for the POG strategy, avoiding expensive MC procedures.

Let us note that while this approximation does not require the generation of M random datasets $\tilde{\mathcal{D}}$ to estimate the p-values, CASPITA still requires the generation of P random datasets $\tilde{\mathcal{D}}$ for the WY method. However, the binomial approximation can also be used to approximate the minimum p-value in the P random datasets. Thus, given the observed dataset $\mathcal{D}$, we compute the p-values for all paths $w \in \mathcal{W}_{\mathcal{D}}(k)$ using Equation 7. Then, we generate a series of P random datasets $\tilde{\mathcal{D}}$ required by the WY method using the POG strategy. For all the P random datasets $\tilde{\mathcal{D}}$, we compute the minimum p-value over all the paths $w \in \mathcal{W}_{\tilde{\mathcal{D}}}(k)$, where the p-value p_w of w is computed with Equation 7 replacing $Occ_{\mathcal{D}}(w)$ with $Occ_{\tilde{\mathcal{D}}}(w)$.

D. Analysis

In this section, we describe in detail our algorithm CASPITA and formally prove its false positives guarantees. Algorithm 1 shows the pseudo-code of CASPITA. Its inputs are the time series dataset $\mathcal{D}$, the FWER threshold $\alpha \in [0, 1]$, the order $h > 0$ of the the generative null model, and the paths length $k > h$. For a given generation strategy, (i.e., TOG or POG), CASPITA first mines the set $\mathcal{W}_{\mathcal{D}}(k)$, of all paths w of length $|w| = k$ that occur at least once in $\mathcal{D}$. Then, it constructs the generative null model $N^h(\mathcal{D})$ as explained in Section III-A, and it uses $N^h(\mathcal{D})$ to compute the p-values of the paths $w \in \mathcal{W}_{\mathcal{D}}(k)$. The p-values can be computed with a MC procedure using Equation 1 (for both generation strategies), and thus generating M random datasets, where M is a parameter set by the user, or with the binomial approximation

using Equation 7 (for the POG strategy). To compute the corrected significance threshold δ^*, it then employs the WY method, which requires the generation of P random datasets, where P is a parameter set by the user. For each random dataset $\tilde{\mathcal{D}}_i$, with $i \in \{1, \ldots, P\}$, it mines the set $\mathcal{W}_{\tilde{\mathcal{D}}_i}(k)$ and then it computes the minimum p-value $p^{(i)}_{min}$ over all paths $\tilde{w} \in \mathcal{W}_{\tilde{\mathcal{D}}_i}(k)$. For the computation of such p-values, the considerations made above are still valid. The corrected significance threshold δ^* is then computed using Equation 3. If $\delta^* > \alpha$, then we set $\delta^* = \alpha$, corresponding to an uncorrected threshold. Finally, the output is the set of paths $w \in \mathcal{W}_{\mathcal{D}}(k)$ such that $p_w < \delta^*$. Theorem 1 proves that the output of CASPITA has FWER $\leq \alpha$.

Algorithm 1: CASPITA

Data: Time Series Dataset $\mathcal{D}$, FWER Threshold $\alpha \in [0, 1]$, Order $h > 0$ of the Generative Null Model, Paths Length $k > h$.
Result: Set $\mathcal{SW}$ with FWER $\leq \alpha$.

$\mathcal{W} \leftarrow$ `MinePaths`$(\mathcal{D}, k)$;
$N^h \leftarrow$ `GenerativeNullModel`$(\mathcal{D}, h)$;
foreach $w \in \mathcal{W}$ **do**
  $p_w \leftarrow$ `PValue`$\big(N^h, w, Occ_{\mathcal{D}}(w)\big)$;
for $i \leftarrow 1$ **to** P **do**
  $\tilde{\mathcal{D}}_i \leftarrow$ `RandomDataset`(N^h, k, h);
  $\mathcal{W}_i \leftarrow$ `MinePaths`$(\tilde{\mathcal{D}}_i, k)$;
  foreach $\tilde{w} \in \mathcal{W}_i$ **do**
    $p_{\tilde{w}} \leftarrow$ `PValue`$\big(N^h, \tilde{w}, Occ_{\tilde{\mathcal{D}}_i}(\tilde{w})\big)$;
  $p^{(i)}_{min} \leftarrow \min\{p_{\tilde{w}} : \tilde{w} \in \mathcal{W}_i\}$;
$\delta^* \leftarrow \max\left\{\delta : \sum_{i=1}^{P}\left(\mathbb{1}[p^{(i)}_{\min} \leq \delta]\right) \leq \alpha P\right\}$;
$\mathcal{SW} \leftarrow \{(w, Occ_{\mathcal{D}}(w), p_w) : w \in \mathcal{W} \wedge p_w < \delta^*\}$;
return $\mathcal{SW}$;

Theorem 1. *The output of* CASPITA *has FWER* $\leq \alpha$.

Proof. Let us consider the P random datasets $\tilde{\mathcal{D}}_i$, with $i \in \{1, \ldots, P\}$, generated by CASPITA for the WY method. Let us note that they do not contain any significant paths of length k, since they are generated from the generative null model N^h, and thus from the distribution described by the null hypothesis. Given $\delta \in [0, 1]$, the FWER FWER(δ) obtained using δ as significance threshold can be estimated using Equation 2. That is, estimated as the fraction, over P, of the number of datasets $\tilde{\mathcal{D}}_i$ that contain at least one path with p-value $\leq \delta$, and thus a path that would be reported as significant while it is not when δ is used as significance threshold. Since CASPITA uses the corrected significance threshold $\delta^* = \max\{\delta : \text{FWER}(\delta) \leq \alpha\}$, then its output has FWER $\leq \alpha$, which concludes our proof. □

We now provide a brief analysis of the time complexity of CASPITA. The time complexity t_W to mine $\mathcal{W}_{\mathcal{D}}(k)$ and t_N to construct $N^h(\mathcal{D})$ are $t_W = t_N = \mathcal{O}(D)$, with $D = \sum_{\tau \in \mathcal{D}} |\tau|$, since they can be done with a single scan of the entire dataset. The time complexity t^{MC}_P to estimate the p-values of all paths $w \in \mathcal{W}_{\mathcal{D}}(k)$ using MC procedures is $t^{MC}_P = \mathcal{O}(M \cdot t_{\tilde{\mathcal{D}}} + |\mathcal{W}(\mathcal{D}(k)|)$, where $t_{\tilde{\mathcal{D}}}$ is the time complexity to generate a random dataset $\tilde{\mathcal{D}}$ and M is the number of random datasets to create. Let us note that $t_{\tilde{\mathcal{D}}}$ depends on the generation strategy, and that it is not trivial to bound it since we do not know in advance the number of random walks that we need to generate $\tilde{\mathcal{D}}$. However, our experimental evaluation empirically proves that such random datasets can be generated with feasible computational time. In addition, the MC procedure is well-suited to parallelization: when C cores are used to compute the p-values considering M random datasets, each core computes the p-values on M/C random datasets, and the results are then aggregated at the end. The time complexity t^B_P to compute the p-values of all paths $w \in \mathcal{W}_{\mathcal{D}}(k)$ using the binomial approximation is instead $t^B_P = \mathcal{O}(|\mathcal{W}(\mathcal{D}(k)|)$, but it first requires the computation of the probabilities of Equation 6. Such computation has time complexity $\mathcal{O}(|\mathcal{S}| \cdot R^{MAX})$, where $|\mathcal{S}|$ is the number of starting nodes and R^{MAX} is the maximum, over all $s \in \mathcal{S}$, number of vertices that can be reached in $k - h$ steps on $N^h(\mathcal{D})$, starting from each vertex s. Finally, the time complexity of the WY method is $t_{WY} = \mathcal{O}(P \cdot (t_{\tilde{\mathcal{D}}} + t_p))$, with P the number of random datasets to generate and t_p one of the two time complexity described above to compute the p-values.

As previously stated, while here we consider the mining of over represented paths, all our reasoning can be easily adapted to the mining of under represented paths.

IV. Mining Statistically Significant Paths from Different Datasets

In this section, we illustrate another interesting scenario in which our algorithm CASPITA can be applied. Let us suppose to have two datasets, $\mathcal{D}_1$ and $\mathcal{D}_2$, and that such two datasets are taken from the same network N, but in different circumstances, e.g., in different temporal points, or maybe that they represent data generated from two different populations, e.g., men and women. In such a scenario, one may be interested in finding paths from one of the two datasets that are statistically significant considering the distribution represented by the other dataset. Thus, it is possible to use a slightly modified version of CASPITA, considering the dataset $\mathcal{D}_1$ to generate the h-th order generative null model $N^h(\mathcal{D}_1)$ and then, to consider the paths mined from the other dataset $\mathcal{W}_{\mathcal{D}_2}(k)$, and to compute their significance using $N^h(\mathcal{D}_1)$. Differently from the scenario described above, in this setting it is also possible to mine statistically significant paths of length k considering the h-th order generative null model with $k = h$.

V. Experimental Evaluation

In this section, we report the results of our experimental evaluation on multiple pseudo-artificial and real datasets to assess the performance of CASPITA for mining statistically significant paths from an unknown network.

The goals of the evaluation are the following: i) to prove that for small datasets, CASPITA is able to find statistically significant paths with both generation strategies, i.e., TOG and

POG, using the MC procedure, while for larger datasets the binomial approximation is necessary to provide useful results; ii) to prove that the binomial approximation is a valid approach to approximate the p-values for the POG strategy; iii) focusing on the POG strategy with the binomial approximation, to prove that CASPITA is able to find large sets of statistically significant paths in real large datasets, while avoiding false positives, and compare CASPITA with HYPA [17]; iv) to prove that CASPITA is able to find statistically significant paths in the scenario in which the generative null model is constructed considering data from an other dataset (see Section IV).

A. Environment and Datasets

We implemented CASPITA in Java. We performed all the experiments on the same machine with 512 GB of RAM and 2 Intel(R) Xeon(R) CPU E5-2698 v3 @ 2.3GHz, using Java 1.8.0_201. To parallelize the MC procedures, we used Apache Spark Java API version 3.1.1. Our open-source implementation of CASPITA and the code developed for the tests and to generate the datasets are available in [20]. In all the experiments, we fixed the FWER threshold to the commonly used value $\alpha = 0.05$. To compare with HYPA [17], we used their implementation available online.[1]

In the following, we describe the datasets used in the evaluation, and how we generated them. Their characteristics are shown in Table I.

1) BIKE: data on the bike sharing service of Los Angeles. Each vertex is a bike station, while each transaction represents the sequence of bike stations that a given bike visits. We considered the 2019 data of the "Los Angeles Metro Bike Share trip data",[2] containing single trips in the format of starting station, ending station, and an unique numerical identifier of the bike, among other information. We collected the temporal ordered sequence of bike stations that each bike visited. Such sequence is a transaction in our dataset. In the case of data anomalies, i.e., an ending station of a trip does not correspond to the starting station of the following trip, we split the sequence where the gap happens, creating two transactions.

2) BIKE10 and BIKE20: smaller versions of the BIKE dataset. From BIKE, we only considered the 10 or 20 vertices, respectively, that occur most frequently times, and collect all the transactions that only contain such vertices.

3) FLIGHT: data of the commercial flights in the USA. Each vertex is an airport, while each transaction represents the sequence of airports visited in a single itinerary by a passenger. We considered the 2019 data of the "Origin and Destination Survey: DB1BCoupon".[3] Such data contains single flights in the format of origin and destination airports, an unique numerical identifier of the itinerary that contains the flight, and the sequence number of the flight inside the itinerary. We collected the temporal ordered sequence of airports that each passenger visited in a single itinerary, sorting the airports using the sequence numbers. Such sequence is a transaction in our dataset.

4) WIKI: it contains human navigation paths on Wikipedia, collected through the human-computation game Wikispeedia [21]. Each vertex is a Wikipedia web-page, while each transaction is a sequence of web-pages visited by an user during a game. We considered the data "paths finished",[4] that represent finished games.

TABLE I
DATASETS CHARACTERISTICS. $|\mathcal{D}|$: NUMBER OF TRANSACTIONS; AVG $|\tau|$: AVERAGE TRANSACTION LENGTH; MAX $|\tau|$: MAXIMUM TRANSACTION LENGTH; FOR THE 1-ST GENERATIVE NULL MODEL $N^1(\mathcal{D})$, $|V^1|$: NUMBER OF VERTICES; $|E^1|$: NUMBER OF EDGES.

Dataset $\mathcal{D}$	$\lvert\mathcal{D}\rvert$	Avg $\lvert\tau\rvert$	Max $\lvert\tau\rvert$	$N^1(\mathcal{D})$	
				$\lvert V^1\rvert$	$\lvert E^1\rvert$
BIKE10	3025	1.54	11	10	76
BIKE20	5080	1.90	21	20	279
BIKE	38651	7.51	232	237	10269
FLIGHT	17447803	1.63	15	455	69234
WIKI	51307	5.76	434	4169	59530

B. Generation Strategies Comparison

In this section, we compare the results obtained by CASPITA with the TOG or POG strategies that employ MC procedures, and the POG strategy that uses the binomial approximation, on BIKE10 and BIKE20.

The experiments have been performed with $P = 1000$, $M = 10^5$, $k \in \{2, \ldots, 5\}$, and $h \in \{1, \ldots, k-1\}$. The results are reported in Table II. For BIKE10, the smallest dataset, the number of significant paths obtained with the TOG and POG strategies with MC procedures differs from at most 1, for all combinations of parameters. The same is true when the POG strategy with the binomial approximation is used. In all the cases, CASPITA reported at most 3 statistically significant paths, which is not surprising since BIKE10 only contains few distinct paths. For BIKE20, the situation is different. For some combinations of parameters (shown in bold in Table II), CASPITA with the MC procedures did not report any significant (over represented) paths, while it reported some paths (from 1 to 8) when the binomial approximation is used. In all such cases, the MC estimates resulted in a corrected threshold $\delta^* = 1/(M+1)$, corresponding to the minimum achievable p-value considering M random datasets. Thus, to be able to mine paths, one has to consider a larger value of M, which is infeasible with larger datasets, or to resort to the binomial approximation. This phenomenon appeared with $k > h - 1$, that is, when a large number of distinct paths can be generated, even for a small dataset such as BIKE20. This emphasizes the issue of the TOG strategy described above, that is, the gargantuan number of paths that must be considered with the generation of long transactions.

We then compared the p-values from the POG strategy obtained with the MC procedure and the binomial approximation.

[1] https://github.com/tlarock/hypa

[2] https://bikeshare.metro.net/about/data/

[3] https://www.transtats.bts.gov/Fields.asp?gnoyr_VQ=FLM

[4] https://snap.stanford.edu/data/wikispeedia.html

Note that while in the MC procedure the total number of length k paths starting from a vertex is fixed to the value observed in the data, using the binomial approximation such property holds only in expectation. Thus, the p-values from the two approaches will be different. However, by comparing the p-values[5] for all paths (over and under represented) of BIKE10 and BIKE20 with $k \in \{2, \ldots, 5\}$ and $h \in \{1, \ldots, k-1\}$, and considering $M \in \{10^4, 10^5, 10^6\}$ random datasets for the MC estimates, we observed that the p-values for the binomial approximation are within one order of magnitude of the corresponding MC p-values, and that the difference between binomial p-values and MC p-values is lower than the standard deviation of the MC estimates (obtained from 5 estimates of the MC p-values). Furthermore, the binomial approximation is several order of magnitude faster than the MC procedure (few milliseconds against over 40 seconds, considering the maximum execution time for both strategies and using 8 cores to parallelize the MC estimates).

C. Results for POG Strategy with Binomial Approximation

Since the results of the previous section demonstrated that the POG strategy with the binomial approximation is necessary to mine statistically significant paths from large datasets, and that the p-values for the binomial approximation are within one order of magnitude of the corresponding MC p-values, in this section we focus on such version of CASPITA.

First, we investigated the false positives guarantees of CASPITA on pseudo-artificial datasets. Starting from a real dataset, we created its h-th order generative null model, which we used to generate random datasets using the POG strategy. Thus, each random dataset is a bag of paths of a given length $k > h$ that does not contain any significant path of length k (since they have been generated in accordance with the generative null model). We then executed CASPITA on each random dataset, with parameters h and k corresponding to the ones used to generate the random dataset, and we checked whether CASPITA reported some paths, which would be false positives by construction. We considered BIKE10 and BIKE20 as starting real datasets, $k \in \{2, \ldots, 5\}$, and $h \in \{1, \ldots, k-1\}$, mining under and over represented paths. Given a real dataset, we generated 20 random datasets for each combinations of h and k, obtaining a total number of 400 runs for each real dataset. We then estimated the FWER as the fraction of runs with at least one false positive. For BIKE10, the estimated FWER is 0.75% with $P = 100$, 1.25% with $P = 1000$, and 0.5% with $P = 10000$. Instead, for BIKE20, the estimated FWER is 2.25% with $P = 100$, 1.75% with $P = 1000$, and 3.25% with $P = 10000$. These results show that the false positives guarantees of CASPITA are even better than the theoretical ones, which are $\leq 5\%$ using $\alpha = 0.05$, and that $P = 100$ is enough to obtain such guarantees. Using these random datasets, we also made a comparison with HYPA. Let us remember that HYPA employs a fixed threshold β to flag as anomalous a path, without any theoretical guarantees. We used $\beta \in \{0.00001, 0.001, 0.05\}$, which are the minimum, the most commonly used, and the maximum value used in [17], and $k \in \{2, \ldots, 5\}$. (For h, HYPA always considers $h = k-1$, thus we only used this value.) For BIKE10, it obtained an estimated FWER of 40.63% with $\beta = 0.05$, 15.63% with $\beta = 0.001$, and 0.0% with $\beta = 0.00001$. Instead, for BIKE20, it obtained an estimated FWER of 60.63% with $\beta = 0.05$, 32.50% with $\beta = 0.001$, and 1.25% with $\beta = 0.00001$. These results show that HYPA is able to return anomalous paths achieving low FWER with the correct threshold, but also emphasize the importance of having a strategy, as the one that we employ, to compute such threshold in an automatic way, since the usage of fixed thresholds may lead to many spurious discoveries, or to a low statistical power.

We then executed CASPITA on some real datasets, i.e., BIKE, FLIGHT, and WIKI. Table III reports the results obtained with $P = 100$, $k \in \{2, \ldots, 5\}$, and $h \in \{1, \ldots, k-1\}$. For all the datasets, and for almost all combinations of parameters, CASPITA reported some significant paths. It is interesting to notice that the number of over represented paths is almost always (some order of magnitude) greater than the number of under represented paths. In addition, for a fixed value of k, the number of over represented paths always decreases considering higher values of h. The number of under represented paths, instead, always decreases for BIKE, while increases and then decreases for FLIGHT and WIKI, highlighting different substructures in the three underlying networks. The computational time ranges from under 3 minutes (BIKE with $k = 2$, $h = 1$) to over 12 hours (FLIGHT with $k = 5$, $h = 1$). Let us note that FLIGHT has over $17M$ transactions, but we are still able to analyze it with a reasonable running time. The combination $k = 5$ and $h = 1$ is always the most expensive from a computational point of view, since it requires to generate longer paths and also to analyze a large number of vertices to compute the probabilities of Equation 7. Overall, these results show that CASPITA is able to efficiently mine significant paths from real datasets, with feasible computational time even in huge datasets.

D. Analysis of BIKE

In this section, we provide a brief analysis of some paths returned by CASPITA from BIKE. The over represented path of length 2 with the lowest p-value and highest number of occurrences is a path which starts and ends in "Ocean Front Walk & Navy" located in Venice Beach. The fact that this path is over represented indicates that people tend to leave and then come back to this place, instead of moving to other parts of the city. For example, such pattern may capture the fact that people leave the beach to buy some food and then immediately come back. Instead, the under represented path of length 2 with the lowest p-value is a path which starts from "Union Station West Portal", goes to "Main & 1st", and then comes back. "Main & 1st" is located near the Los Angeles City Hall, the center of the government of the city, while "Union Station West Portal" is near the Union Station, the main railway station of the city.

[5]We only considered p-values $\geq 1/(M+1)$, since lower p-values require larger M to be correctly estimated with the MC procedure.

TABLE II

CASPITA RESULTS WITH BIKE10 AND BIKE20. k: PATHS LENGTH; h: ORDER OF THE NULL MODEL; FOR EACH DATASET, BIKE10 AND BIKE20, $|\mathcal{W}|$: NUMBER OF DISTINCT PATHS OF LENGTH k; T: NUMBER OF TOTAL PATHS OF LENGTH k; FOR EACH GENERATION STRATEGY, TOG (T), POG (P), AND POG WITH BINOMIAL APPROXIMATION (B), $|\mathcal{SW}|$: NUMBER OF SIGNIFICANT PATHS REPORTED, OVER $(+)$ AND UNDER $(-)$ REPRESENTED.

k	h	BIKE10								BIKE20							
		$\|\mathcal{W}\|$	T	$\|\mathcal{SW}^T\|$		$\|\mathcal{SW}^P\|$		$\|\mathcal{SW}^B\|$		$\|\mathcal{W}\|$	T	$\|\mathcal{SW}^T\|$		$\|\mathcal{SW}^P\|$		$\|\mathcal{SW}^B\|$	
				$+$	$-$	$+$	$-$	$+$	$-$			$+$	$-$	$+$	$-$	$+$	$-$
2	1	164	1630	3	3	3	3	3	3	978	4553	11	12	9	8	11	8
3	1	163	575	1	2	2	2	2	2	997	2320	0	9	0	7	**8**	7
	2			1	2	1	2	1	2			2	8	1	6	1	5
4	1	104	210	0	2	0	2	1	2	713	1220	0	4	0	5	**4**	5
	2			0	1	1	1	1	1			0	7	0	4	0	4
	3			0	1	0	0	0	0			0	4	0	2	0	2
5	1	57	83	0	2	0	2	0	2	450	661	0	2	0	2	**2**	2
	2			0	1	1	0	1	0			0	3	0	2	**1**	2
	3			0	0	0	0	0	0			0	1	0	1	0	1
	4			0	0	0	0	0	0			0	0	0	1	0	1

TABLE III

CASPITA RESULTS WITH REAL DATASETS. k: PATHS LENGTH; h: ORDER OF THE NULL MODEL; FOR EACH DATASET, BIKE, FLIGHT, AND WIKI, $|\mathcal{W}|$: NUMBER OF DISTINCT PATHS OF LENGTH k; T: NUMBER OF TOTAL PATHS OF LENGTH k; NUMBER OF SIGNIFICANT PATHS REPORTED, OVER $(+)$ AND UNDER $(-)$ REPRESENTED; TIME (S): EXECUTION TIME IN SECONDS.

k	h	BIKE					FLIGHT					WIKI				
		$\|\mathcal{W}\|$	T	$+$	$-$	Time (s)	$\|\mathcal{W}\|$	T	$+$	$-$	Time (s)	$\|\mathcal{W}\|$	T	$+$	$-$	Time (s)
2	1	90.4K	252K	197	28	150	574K	11.1M	96.5K	16.4k	5.73K	155K	244K	160	53	264
3	1	172K	221K	118	40	350	849K	5.16M	132K	36	10.0K	169K	194K	219	6	384
	2			1	8	375			128K	1.63K	13.3K			8	12	495
4	1	179K	195K	80	15	662	406K	530K	19.2K	0	2.85K	139K	147K	193	1	765
	2			10	7	639			10.4K	30	1.99K			16	6	747
	3			0	3	713			3.11K	19	1.52K			2	2	594
5	1	166K	174K	71	6	855	127K	155K	6.66K	0	43.9K	106K	108K	113	0	1.03K
	2			17	3	514			4.28K	0	1.31K			7	0	371
	3			0	1	458			1.80K	5	723			0	1	267
	4			0	1	365			884	2	638			0	0	211

The fact that this path is under represented is probably due to the fact that a lot of people move from the station to the city hall, and vice-versa, but in particular moments of the day, i.e., in the morning and in the evening. Thus, even if the two direct links are very popular, it is uncommon to see this entire path. These are only two example of paths mined by CASPITA, but they highlight its capability in detecting real life trends.

Finally, we investigated the capability of CASPITA in mining significant paths in the scenario in which the generative null model is created considering a different dataset (see Section IV). Using the procedure to generate BIKE (described in [20]), we generated a new dataset, NEWBIKE, considering the 2020 data from the same website. We then used the original BIKE dataset to create the h-th order generative null model and we tested on it the significance of length k paths mined from NEWBIKE. Given the pandemic situation that involved the world, one may be interested in finding changes in the habits of the people defining a model based on paths traveled in 2019 to test paths traveled in 2020. Table IV reports the results obtained with $k = h \in \{1, \ldots, 5\}$ and $P = 100$. (For NEWBIKE, we only considered full transactions that can be generated by the generative null model obtained from BIKE.) Again, it is possible to notice that CASPITA returned paths for almost all combinations of parameters, and that the number of over represented paths is always higher than the number of under represented paths. Overall, these results demonstrate that CASPITA is able to mine paths also in such a scenario.

TABLE IV

CASPITA RESULTS ON BIKE CONSIDERING DIFFERENT DATASETS. SEE TABLE III FOR THE MEANING OF THE VALUES.

$k = h$	$\|\mathcal{W}\|$	T	$+$	$-$
1	5.79K	68.0K	256	120
2	5.51K	12.2K	30	9
3	1.19K	2.45K	14	1
4	436	786	4	0
5	124	203	0	0

VI. Conclusions

In this work, we introduced the problem of mining *statistically significant paths* in time series data from an unknown network, which naturally arises in several applications. We described CASPITA, an algorithm to mine statistically significant paths (over and under represented) while controlling the probability of reporting at least one false positive, i.e., the FWER, employing the Westfall-Young method. We also introduced an alternative scenario which considers a different dataset to construct the generative null model. Our extensive experimental evaluation shows that CASPITA is able to efficiently mine large sets of significant paths from real datasets, while correctly controlling the FWER. Interesting future directions are the extension of CASPITA to mine statistically significant paths while controlling the *false discovery rate* (FDR) and to identify the k most significant paths.

Acknowledgment

Part of this work was supported by the MIUR, the Italian Ministry of Education, University and Research, under the initiative "Departments of Excellence" (Law 232/2016) and PRIN Project n. 20174LF3T8 "AHeAD", and by the University of Padova project "SID 2020: RATED-X".

References

[1] E. Keogh and S. Kasetty, "On the need for time series data mining benchmarks: a survey and empirical demonstration," *Data Min. Knowl. Discov.*, 2003.
[2] J. F. Roddick et al., "An updated bibliography of temporal, spatial, and spatio-temporal data mining research," *Int. Workshop on Temporal, Spatial, and Spatio-Temporal Data Min.*, 2000.
[3] P. Esling and C. Agon, "Time-series data mining," *ACM Computing Surveys*, 2012.
[4] G. M. Weiss, "Mining with rarity: a unifying framework," *ACM SIGKDD Explorations Newsletter*, 2004.
[5] E. Keogh et al., "Finding surprising patterns in a time series database in linear time and space," *ACM SIGKDD*, 2002.
[6] J. Lin et al., "Visually mining and monitoring massive time series," *ACM SIGKDD*, 2004.
[7] E. Keogh and J. Lin, "Clustering of time-series subsequences is meaningless: implications for previous and future research," *Knowl. Info. Syst.*, 2005.
[8] L. Wei and E. Keogh, "Semi-supervised time series classification," *ACM SIGKDD*, 2006.
[9] W. Hämäläinen and G. I. Webb, "A tutorial on statistically sound pattern discovery," *Data Min. Knowl. Discov.*, 2019.
[10] F. Llinares-López et al., "Fast and memory-efficient significant pattern mining via permutation testing," *ACM SIGKDD*, 2015.
[11] L. Pellegrina and F. Vandin, "Efficient mining of the most significant patterns with permutation testing," *Data Min. Knowl. Discov.*, 2020.
[12] A. Tonon and F. Vandin, "Permutation strategies for mining significant sequential patterns," *IEEE ICDM*, 2019.
[13] M. Gupta et al., "Outlier detection for temporal data: A survey," *Trans. Knowl. Data Eng.*, 2013.
[14] F. Lemmerich et al., "Mining subgroups with exceptional transition behavior," *ACM SIGKDD*, 2016.
[15] C. C. Noble and D. J. Cook, "Graph-based anomaly detection," *ACM SIGKDD*, 2003.
[16] L. Akoglu et al., "Graph based anomaly detection and description: a survey," *Data Min. Knowl. Discov.*, 2015.
[17] T. LaRock et al., "Hypa: Efficient detection of path anomalies in time series data on networks," *SIAM SDM*, 2020.
[18] I. Scholtes, "When is a network a network? multi-order graphical model selection in pathways and temporal networks," *ACM SIGKDD*, 2017.
[19] P. H. Westfall and S. S. Young, "Resampling-based multiple testing: Examples and methods for p-value adjustment," *John Wiley & Sons*, 1993.
[20] https://github.com/VandinLab/CASPITA.
[21] R. West and J. Leskovec, "Human wayfinding in information networks," *WWW*, 2012.