Mining Statistically Significant Connected Subgraphs in Vertex Labeled Graphs

Akhil Arora
aarora@cse.iitk.ac.in

Mayank Sachan
mayasac@cse.iitk.ac.in

Arnab Bhattacharya
arnabb@iitk.ac.in

Dept. of Computer Science and Engineering,
Indian Institute of Technology, Kanpur,
India

ABSTRACT

The steady growth of graph data in various applications has resulted in wide-spread research in finding significant sub-structures in a graph. In this paper, we address the problem of finding statistically significant connected subgraphs where the nodes of the graph are labeled. The labels may be either discrete where they assume values from a pre-defined set, or continuous where they assume values from a real domain and can be multi-dimensional. We motivate the problem citing applications in spatial co-location rule mining and outlier detection. We use the chi-square statistic as a measure for quantifying the statistical significance. Since the number of connected subgraphs in a general graph is exponential, the naïve algorithm is impractical. We introduce the notion of contracting edges that merge vertices together to form a super-graph. We show that if the graph is dense enough to start with, the number of super-vertices is quite low, and therefore, running the naïve algorithm on the super-graph is feasible. If the graph is not dense, we provide an algorithm to reduce the number of super-vertices further, thereby providing a trade-off between accuracy and time. Empirically, the chi-square value obtained by this reduction is always within 96% of the optimal value, while the time spent is only a fraction of that for the optimal. In addition, we also show that our algorithm is scalable and it significantly enhances the ability to analyze real datasets.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data Mining*

Keywords

Graph mining; Statistical significance; Chi-square; Vertex labels; Connected subgraphs; Significant subgraphs

1. MOTIVATION

The significant growth of graph data in a wide range of commercial and scientific applications including bioinformatics, chemo-informatics and social networks has encouraged research in studying various patterns within graphs. Many of these applications rely on extracting significant substructures in the form of paths [26], trees [4], cliques [19], or subgraphs [11, 21, 33].

The problem has been mostly studied with reference to frequent subgraphs [21, 33] where a subgraph is considered to be significant when its frequency exceeds a particular threshold. However, in many applications, this characterization is not always sufficient. Rather, identifying the *statistically significant subgraphs* where the pattern within the subgraph deviates from the expected can potentially unearth interesting properties that merit further thorough analyses. The idea that a frequent subgraph might not be significant was explored in [11] and a new definition of statistical significance was proposed by transforming graphs into feature vectors. In [21], a scalable approach to mine significant patterns was proposed, and in [33], they were searched using dissimilar graph patterns.

There are a broad range of applications where the nature of the graph is such that every vertex of the graph is assigned a label. In biological and chemical networks, vertices are usually labeled from a discrete set of biochemical entities ranging from molecules to genes [35]. The continuous numerical labeling scheme is used in quantifying the amount of traffic at a particular node in communication networks [20] or in assigning scores to nodes in the web networks [32]. Many spatial datasets are represented in the form of such vertex labeled graphs [10, 14, 15, 16, 27].

Depending on the application, the vertex labels can be discrete [14, 27] or continuous [10, 16]. In a spatial dataset pertaining to ecology, the discrete vertex labels can be chosen from a set of boolean features enlisting different vegetation types (an example is the North-East dataset mentioned in Section 5.1) while in a geographical dataset, the vertices may be assigned a continuous numerical attribute such as the density of infected cases in that particular location (the WNV dataset explained in Section 5.2). Moreover, the continuous labels can be multi-dimensional such as rainfall and temperature [15]. Although mapping a continuous range to discrete intervals is sometimes possible, such a reduction may lose significant information and is, therefore, not always desirable.

Consequently, in this paper, we consider the two cases as being inherently different and study them separately for the purpose of mining *statistically significant connected subgraphs*. The condition of connectivity is necessary to guarantee a contiguous region in the space which ensures a meaningful relationship among the nodes. Statistical significance ascertains whether a given outcome is due to some extraneous factors or can be ascribed solely due to chance. It quantifies the deviation of the observed behavior from what is expected according to an underlying *null hypothesis*. The null hypothesis specifies the expected distribution of labels for both discrete and continuous cases. We consider the basic null model in

SIGMOD'14, June 22–27, 2014, Snowbird, UT, USA.

ACM 978-1-4503-2376-5/14/06 ...$15.00.
http://dx.doi.org/10.1145/2588555.2588574.

which the labels of the vertices are assumed to be independently and randomly drawn from a fixed probability distribution. The deviation of the actual labels of the vertices in a subgraph from the expected labels imposed by the null model is used as a measure of statistical significance.

The *p-value* [22] is one of the most widely used metrics for quantifying statistical significance. It denotes the probability of obtaining a test statistic at least as extreme as the one that is actually observed, assuming the underlying null hypothesis to be true. Hence, the lower the p-value, the higher is the statistical significance.

Since p-value computation may require exponential number of steps, we use the *chi-square statistic* [22] to make it practical. The chi-square statistic is the most commonly used estimate [5, 31, 34] of the p-value (the details about this approximation are mentioned in [22]). Higher the chi-square value, lower the p-value, and, thus, more statistically significant is the subgraph.

Our main contribution in this paper is to handle the general problem of identifying such statistically significant connected subgraphs efficiently. To the best of our knowledge, no prior research has been done that uses vertex labels for mining statistically significant connected subgraphs for both discrete and continuous labeled graphs.

Our proposed algorithm finds the most statistically significant connected subgraphs in expected linear time for "dense" graphs (we later formally define what is meant by a dense graph).

The rest of the paper is organized as follows. In Section 2, we explain in detail one application for each of the discrete and continuous labeling cases before the problem statements are presented. Section 3 highlights the related work while Section 4 describes our algorithm and its analysis. In Section 5, experimental results are presented before Section 6 concludes.

2. PROBLEM STATEMENT

Although our problem statement is much more generic and can be applied to a variety of settings, we motivate the discrete case specifically through co-location rule mining and the continuous case through outlier detection problem.

2.1 Discrete Labeling

Mining statistically significant subgraphs is applicable in finding co-location patterns from spatial databases that aim to reveal features that are likely to be closely located with each other in the spatial dimension [14, 27]. Given a set of n locations in a spatial database endowed with a neighborhood relationship $\mathcal{N}$, suppose $F = \{f_1, \ldots, f_l\}$ denotes a set of l boolean spatial features on each node. The objective of the *co-location rule* mining is to find rules of the form $X \Rightarrow Y$, where X and Y are subsets of spatial features. A particular instance of the features is said to be *co-located* if they occur within the neighborhood of each other. For example, a rule *{traffic jam, police} ⇒ {accident} (80%)* signifies that whenever there are traffic jam and policemen, there is a 80% probability of an accident in the neighborhood. Various algorithms exist for mining such co-location rules [14, 27].

Given any such co-location rule, it may be interesting to find the contiguous regions in the space where this co-location rule is statistically significant. The statistical significance may imply that over that region, instances of X contain an exceptionally large or an exceptionally small ratio of instances of Y in their neighborhood. Once such regions have been identified, it may be useful to study the external forces acting over those regions to establish the causality of such co-location rules.

The space can be modeled as an undirected, un-weighted, labeled graph G where each vertex represents a spatial location. The connectivity of a pair of vertices through an edge is guided by some criterion such as a Euclidean distance threshold or a common border. Each vertex is labeled with a symbol from an alphabet Σ of size l. The boolean set for a co-location rule $X \Rightarrow Y$, therefore, is a set of two symbols c_0 and c_1 where c_0 represents the presence of Y in the neighborhood of X and c_1 represents the absence. The underlying assumption is that these symbols are randomly and independently drawn from a fixed binomial probability distribution $P = \{p_0, p_1\}$ where $p_0 + p_1 = 1$. The choice of p_0 can be either provided by the co-location rule (0.80 in the above example) or can be empirically calculated as the fraction of number of occurrences over the whole space. The null hypothesis states that the co-located patterns are distributed uniformly throughout the space. With respect to the given null hypothesis, the objective is to mine the statistically significant connected subgraphs.

The above analysis can be applied to a general graph as well. Consider a spatially connected graph where the nodes are labeled from a discrete set. Some of the labels can be rare, either due to its semantics, or due to prior empirical expectations. For example, in an ecologically important region where the nodes are marked with biodiversity and disturbance levels, it is intuitive that high disturbance should not allow high biodiversity. Thus, a label that encapsulates the high values together is expected to be rare. Hence, a spatial region that contains a large proportion of such labels will be a statistically significant connected subgraph. Such a finding is extremely important as it highlights an ecologically valuable region under the threat of human disturbance such as poaching.

Statistical significance is quantified by the *chi-square* statistic [22], which measures the deviation of the observed frequencies (O_i) from their expected values (E_i) given by the null model:

$$X^2 = \sum_{\forall \text{labels}} \frac{(O_i - E_i)^2}{E_i}. \quad (1)$$

Consider a subgraph with n vertices and l possible labels having an observed frequency vector $Y = \{Y_1, \ldots, Y_l\}$ where $\sum_{i=1}^{l} Y_i = n$. Since each vertex label is drawn randomly and independently from a fixed probability distribution $P = \{p_1, \ldots, p_l\}$, the expected frequencies are $E_i = n.p_i$. The chi-square, therefore, is

$$X^2 = \sum_{i=1}^{l} \frac{(Y_i - n.p_i)^2}{n.p_i} = \sum_{i=1}^{l} \frac{Y_i^2}{n.p_i} - n. \quad (2)$$

It has been shown that under the assumed null hypothesis, the chi-square statistic follows the chi-square distribution with $l-1$ degrees of freedom, denoted by $\chi^2(l-1)$ [22].

If all the outcomes are independent, the *p-value* can be computed by using the cumulative distribution function (cdf) $F(\cdot)$ of the $\chi^2(l-1)$ distribution. If z is the X^2 value of an observed outcome, then its p-value is $1 - F(z)$.

Although the X^2 statistic for a particular subgraph follows the χ^2 distribution, since the subgraphs share vertices, the individual X^2 values of all the subgraphs are *not* mutually independent. Hence, it is not possible to directly map the X^2_{max} statistic to a *p-value*. Nevertheless, a higher X^2 value will always necessarily point to a smaller p-value. Moreover, as the focus of the paper is not quantifying the degree of statistical significance directly but designing scalable algorithms to find the *most statistically significant connected subgraphs*, i.e, the ones with the *highest* chi-square values, this requirement is not crucial.

DEFINITION 1 (MOST SIGNIFICANT SUBGRAPH). *The* most significant connected subgraph (MSCS) *of a graph is the connected subgraph having the highest chi-square value among all possible connected subgraphs.*

DEFINITION 2 (TOP-t SIGNIFICANT SUBGRAPHS). *The* top-t set of significant subgraphs (TSSS) *of a graph is the set of t connected subgraphs having the highest chi-square values with the constraint that none of them share any vertex.*

The problem we tackle in this paper is next stated formally.

PROBLEM 1. *Given an undirected un-weighted graph having every vertex of it labeled with one of l possible symbols from Σ, find the MSCS. More generally, find the TSSS.*

One iterative way of obtaining the TSSS set is to first find the MSCS and then delete it from the graph to obtain the MSCS for the next step, and so on.

In order to find these subgraphs, a very useful concept is that of a *local maximally significant connected subgraph*.

DEFINITION 3 (LOCAL MAXIMALLY SIGNIFICANT). *A connected subgraph H of a graph is said to be a* local maximally significant connected subgraph (LMCS) *if it is not possible to increase the chi-square value of H by either adding to or removing from it a vertex such that H still remains connected.*

By definition, a MSCS and all members of TSSS are LMCS.

Several other interesting problems can also be conceived of, e.g., finding the connected subgraphs whose significance is *greater than a threshold* or finding the most significant connected subgraph that *exceeds a particular size*, etc. The TSSS algorithm can be utilized for solving these cases. A sufficiently large t can be chosen such that all subgraphs in the answer set exceed the particular significance threshold. Similarly, a large enough t will allow finding a subgraph whose size is more than the queried threshold size. Thus, although such problems are interesting in their own right, we stick to the more basic MSCS and TSSS problems in this paper.

2.2 Continuous Labeling

Mining statistically significant subgraphs where node labels assume real values can be applied to detecting *spatial outliers*, which are defined as observations that are inconsistent with their neighborhood [10, 16, 28]. Most outlier detection techniques assume that the attribute values (i.e., labels) are dependent on just the neighbors, and accordingly, classify a node as an outlier based only on that.

One standard technique for outlier detection is using *z-scores* [23]. In this technique, first, the value x_i of a node i is scaled according to its neighborhood by subtracting the weighted average of the attribute values of the neighbor set $\mathcal{N}(i)$:[1]

$$y_i = x_i - \sum_{j \in \mathcal{N}(i)} w_j x_j. \tag{3}$$

Scaling helps in eliminating the dependence of x_i on its neighbors, i.e., the y_i values become independent and identically distributed (i.i.d.) under the null hypothesis. The z-score of a node is then calculated using the sample mean $\widehat{\mu}$ and sample standard deviation $\widehat{\sigma}$ of the scaled values (these are representatives of the true values)

$$z_i = (y_i - \widehat{\mu})/\widehat{\sigma}. \tag{4}$$

Since y_i's are i.i.d. random variables, due to the central limit theorem [12], the z-score follows the standard normal distribution $N(0,1)$. Moreover, similar to y_i's, under the null hypothesis, z_i's are also independent. Hence, the nodes exhibiting large magnitudes of z-scores (either positive or negative) are deemed as outliers.

[1]Some useful weighting schemes have been proposed in [16].

Instead of just finding single outlier nodes, we extend the notion to *outlier spatial regions* as well. The *combined z-score* of a region Z_S containing the set of nodes, S, is defined as the (scaled) sum of the individual z-scores:

$$Z_S = \sum_{i \in S} z_i / \sqrt{|S|}. \tag{5}$$

The sum is scaled by a factor of $1/\sqrt{|S|}$ to ensure that under the null hypothesis the distribution of the combined z-scores remains the same as those of individual nodes. In this way, the statistical significances of regions and single nodes are directly comparable.

The z-score of a set S, composed of a mutually exclusive set of nodes S_1 and S_2 with z-scores Z_1 and Z_2 respectively, is

$$Z_S = \frac{\sqrt{|S_1|}.Z_1 + \sqrt{|S_2|}.Z_2}{\sqrt{|S_1 + S_2|}}. \tag{6}$$

Since all z_i's are i.i.d. $N(0,1)$ variables, Z_S also follows $N(0,1)$.

We further extend the notion of z-score to multiple dimensions. Given a set of k independent multiple attributes of a node, the *multi-dimensional z-score* of a set of nodes, S, is a tuple $(Z_S^1, \ldots, Z_S^k)$ where Z_S^j represents the z-score of the j^{th} attribute of S. Since the attributes are independent of each other, the z-scores in each dimension are also independent of each other. Further, since each Z_S^j follows $N(0,1)$, the multi-dimensional z-score $(Z_S^1, \ldots, Z_S^k)$ follows a multi-variate standard normal distribution whose *pdf* is

$$f(Z_S^1, \ldots, Z_S^k) = \frac{1}{(2\pi)^{k/2}} e^{-\sum_{j=1}^{k} (Z_S^j)^2/2}. \tag{7}$$

The *chi-square* statistic, X^2, of the multi-dimensional z-score is defined as the sum of squares of the z-scores in each dimension:

$$X_S^2 = \sum_{j=1}^{k} (Z_S^j)^2. \tag{8}$$

Since the X^2 statistic appears as a negative power of exponential in the pdf (Eq. (7)), outcomes having higher chi-square statistic values have lesser probability densities and are, therefore, less likely to occur. Since X^2 is the sum of k individual i.i.d. $N(0,1)$ variables, it follows the $\chi^2(k)$ distribution under the null hypothesis.

The objective, thus, is to find contiguous regions in the space as outliers. These outlier regions are classified as statistically significant with respect to the given null hypothesis.

We reduce this problem to the generic graph problem where the nodes represent the individual locations and edges represent the spatial connectivity or the neighborhood relationships between these locations. A contiguous spatial region corresponds to a connected subgraph of this spatial graph. The problem, therefore, is to find the connected subgraphs exhibiting large chi-square values.

PROBLEM 2. *Given an undirected un-weighted graph with every vertex containing a multi-dimensional z-score value, find the MSCS. More generally, find the TSSS.*

3. RELATED WORK

Randomization tests, also called permutation tests [7], is a common method for testing significance of graphs. Given an input graph, the tests sample data from the class of graphs that share certain structural properties with the input graph. The basic idea in randomization tests is to perturb the original graph by swapping certain edges but still maintaining the structural properties and then carry out experiments with the randomized versions of the graph

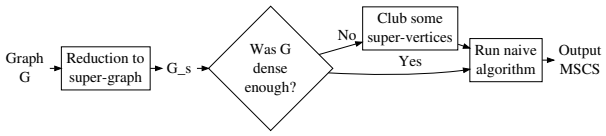


Figure 1: Overview of our algorithm.

thus obtained. Though the objective is similar, our problem is inherently different as the randomization is not in the structure of the graph but in the vertex labels. Moreover, the randomization tests are used to generate the distribution for the test statistic that do not follow any standard distribution, while for us, the null hypothesis assumes the well-known chi-square distribution for vertex labels.

The existing literature has witnessed a host of random graph models, the oldest of which is the Erdős-Rényi (ER) model [9]. Its major drawback is that it does not model real-life phenomena as it assumes independent and equally probable edges. Watts et al. [30] introduced the Watts-Strogatz model with "small-world" properties (high clustering, triadic closures, etc.). This model, however, has unrealistic degree distributions unlike that in standard "scale-free" networks, for which Newman et al. [18] proposed an improvement. Many real world graphs including the world wide web, biological networks, social networks etc. are believed to be scale-free and schemes such as *preferential attachment* have been proposed to explain such networks. The preferential attachment paradigm states that the more popular a node is, the higher is the chance of a new node getting attached to it. The Barabási-Albert (BA) model [1] incorporated this notion with the dynamic growth of a graph as well. Holme et al. [13] extended the BA model by including a triad formation step to improve the clustering coefficient. The importance of triads in real graphs has been highlighted by Durak et al. [6].

Mining statistically significant co-location patterns was studied in [2] where instead of using a threshold, subsets of spatial features that are co-located due to some form of spatial dependency were found. It was extended in [3] where segregation patterns were mined by taking into account the auto-correlation of the co-location rules. Statistical significance techniques were used to locate the most informative set of patterns in [17]. The p-value and z-score computation approaches were used for hotspot detection in [29].

The problem of identifying statistically significant substrings with multinomial distribution as the underlying null hypothesis has been recently studied [25]. The string can be seen as an over-simplified version of a graph with substrings representing the connected subgraphs. Hence, this can be considered a special case of our study.

4. ALGORITHM

4.1 Naïve Algorithm

The naïve algorithm for both the discrete and the continuous case simply examines all the possible connected subgraphs. Since the number of different connected subgraphs for a graph can be exponential in the number of vertices, irrespective of the number of edges, the running time is impractical unless the graph is small.

4.2 Outline of our Algorithm

To solve the problem of finding statistically significant subgraphs efficiently, we propose an algorithm as outlined in Figure 1. We first reduce the input graph $G(V, E)$ to a super-graph $G_s(V_s, E_s)$ by merging together some of the vertices and removing the edges between them. We prove that if the original graph G is dense enough (we will specify exactly what is meant by "dense enough" later), then the size of this super-graph G_s is so small that running the naïve algorithm on it is feasible. The reduction to super-graph for the discrete case has the desirable property that the MSCS and TSSS in the original graph are exactly the same as the MSCS and TSSS in the super-graph. The continuous case, however, may not always exhibit such properties.

If, however, G was not sufficiently dense to start with, the size of G_s is not sufficiently small either. Hence, we need to reduce its size by further clubbing together some of the super-vertices. This step does not guarantee that the final MSCS obtained is the actual MSCS. This, thus, provides a trade-off between running time and accuracy. Through detailed analysis and experiments we show that the obtained solution does not deviate much from the optimal.

Next, we describe the super-graph creation and reduction algorithms in detail. The algorithms are fundamentally similar for discrete and continuous labels.

Algorithm 1 Construction of Super-graph (Discrete Labels)

Input: Graph $G = (V, E)$
Output: Super-graph $G_s = (V_s, E_s)$
1: Copy $G_c(V_c, E_c) \leftarrow G(V, E)$
2: Delete all non-contracting edges from E
3: $V_s \leftarrow$ Connected components of G_c by performing BFS
4: **for** all edges $e = (u, v) \in E$ **do**
5: **if** e is not contracting **then**
6: Super-vertices u_s contains u and v_s contains v
7: Add super-edge $e_s = (u_s, v_s)$ to G_s
8: **end if**
9: **end for**
10: **return** $G_s(V_s, E_s)$

4.3 Construction of the Super-graph

The *super-graph* $G_s = (V_s, E_s)$ is obtained by clubbing together neighboring vertices of the original graph $G = (V, E)$. The vertices of the super-graph are called *super-vertices*. The set of super-vertices are *mutually exclusive* and *exhaustive*. There exists a *super-edge* between two super-vertices v_{s_1} and v_{s_2} if and only if there exists an edge $(v_1, v_2) \in E$ with $v_1 \in v_{s_1}$ and $v_2 \in v_{s_2}$.

We first explain the notion of a *contracting edge*. An edge is contracting if the two vertices that it joins become part of the same super-vertex; otherwise it is non-contracting. The exact definition of when two vertices can become part of the same super-vertex differs from the discrete to the continuous case, and will be specified when the super-graph construction algorithms are explained.

We start with the discrete labeled graphs and then follow it up with the continuous labeled case.

4.3.1 Discrete Labels

The pseudo-code for the super-graph construction for a discrete labeled graph is shown in Algorithm 1. For discrete labels, an edge $e = (v_1, v_2)$ is *contracting* if and only if the labels of v_1 and v_2 are the same. The algorithm first deletes all non-contracting edges from a copy G_c of the original graph G (lines 1-2). It then finds all the *connected components* in G_c by running a BFS (line 3). Each connected component forms a super-vertex. Two super-vertices are connected if and only if two component vertices of them are connected in the original graph (lines 4-9).

The algorithm requires $O(m + n)$ time for n vertices and m edges. It also leads to the following conclusion.

CONCLUSION 1. *Given any connected subgraph H_s of the super-graph G_s, there exists a corresponding connected subgraph $H \subseteq G$ such that the vertex set of H is the union of the vertices in each super-vertex of H_s.*

The next result is important to understand why a classification of edges into contracting and non-contracting is useful.

LEMMA 1. *For a given connected subgraph H, if addition of one vertex with label c_r to H does not decrease the X^2 value of H, then addition of one more vertex of the same label c_r necessarily increases the X^2 value of H.*

PROOF. Suppose H_1 and H_2 are the subgraphs obtained by adding respectively one and two vertices of label c_r to H. Also, suppose Z_0, Z_1 and Z_2 are the corresponding X^2 values of H, H_1 and H_2 respectively. Further, assume that the sizes of the three subgraphs H, H_1 and H_2 are $t-1$, t and $t+1$ respectively, and their count vectors (i.e., the vectors denoting the number of vertices of each label) are $\{X_1, \ldots, X_r - 1, \ldots, X_l\}$, $\{X_1, \ldots, X_r, \ldots, X_l\}$ and $\{X_1, \ldots, X_r + 1, \ldots, X_l\}$ respectively where $\sum_{i=1}^{l} X_i = t$. Using the definition of X^2 from Eq. (2), we get

$$Z_1 = \sum_{i=1}^{l} \frac{X_i^2}{tp_i} - t. \tag{9}$$

$$\begin{aligned} Z_0 &= \sum_{i=1, i\neq r}^{l} \frac{X_i^2}{(t-1)p_i} + \frac{(X_r-1)^2}{(t-1)p_r} - (t-1) \\ &= \sum_{i=1}^{l} \frac{X_i^2}{(t-1)p_i} - \frac{2X_r - 1}{(t-1)p_r} - (t-1) \\ &= \frac{1}{t-1}\left[t(Z_1 + t) - \frac{(2x_r - 1)}{p_r} - (t-1)^2\right] \\ &= \frac{1}{t-1}\left[tZ_1 - \frac{(2x_r - 1)}{p_r} + 2t - 1\right]. \end{aligned} \tag{10}$$

$$\begin{aligned} Z_2 &= \sum_{i=1, i\neq r}^{l} \frac{X_i^2}{(t+1)p_i} + \frac{(X_r+1)^2}{(t+1)p_r} - (t+1) \\ &= \sum_{i=1}^{l} \frac{X_i^2}{(t+1)p_i} + \frac{2X_r + 1}{(t+1)p_r} - (t+1) \\ &= \frac{1}{t+1}\left[t(Z_1 + t) + \frac{(2X_r + 1)}{p_r} - (t+1)^2\right] \\ &= \frac{1}{t+1}\left[tZ_1 + \frac{(2X_r + 1)}{p_r} - 2t - 1\right]. \end{aligned} \tag{11}$$

Since it is given that $Z_1 \geq Z_0$, we have

$$\begin{aligned} & Z_1 \geq \frac{1}{t-1}\left[tZ_1 - \frac{(2X_r - 1)}{p_r} + 2t - 1\right] \\ \Rightarrow\ & (t-1)Z_1 \geq tZ_1 - \frac{(2X_r - 1)}{p_r} + 2t - 1 \\ \Rightarrow\ & \frac{(2X_r - 1)}{p_r} \geq Z_1 + 2t - 1 \\ \Rightarrow\ & \frac{(2X_r + 1)}{p_r} \geq Z_1 + 2t - 1 + \frac{2}{p_r}. \end{aligned} \tag{12}$$

Plugging Eq. (12) in Eq. (11), and using $0 < p_r < 1$, we get

$$\begin{aligned} & Z_2 \geq \frac{1}{t+1}\left[tZ_1 + Z_1 + 2t - 1 + \frac{2}{p_r} - 2t - 1\right] \\ \Rightarrow\ & Z_2 \geq Z_1 + \frac{1}{t+1}\left(\frac{2}{p_r} - 2\right) > Z_1. \end{aligned} \tag{13}$$

□

Algorithm 2 Construction of Super-graph (Continuous Labels)

Input: Graph $G = (V, E)$
Output: Super-graph $G_s = (V_s, E_s)$

```
V_s ← ∅; E_s ← E
for all vertices v ∈ V do
    v_s ← v; v.super ← v_s
    Add v_s to V_s
end for
for all edges e = (u, v) ∈ E do
    combz ← Combined z-score of u_s and v_s
    if X^2_combz > max{X^2_{u_s}, X^2_{v_s}} then
        Super-vertex w = Merge(u_s, v_s)
        X^2_w ← X^2_combz
        Add all edges of form (u_s, x) and (v_s, x) as (w_s, x)
        V_s ← (V_s ∪ {w_s}) − {u_s, v_s}; E_s ← E_s − {e}
    end if
end for
return G_s(V_s, E_s)
```

The next result uses Lemma 1 to prove that for all bi-connected LMCS of G (and hence, also the MSCS and the TSSS of G), there exists an equivalent connected subgraph in the super-graph G_s.[2]

LEMMA 2. *Given any bi-connected LMCS $H \subseteq G$, there exists a connected subgraph $H_s \subseteq G_s$ such that every super-vertex of H_s is composed of vertices from H and every vertex of H is also a part of some super-vertex of H_s. Alternatively, $H_s \subseteq G_s$ is equivalent to $H \subseteq G$ as X^2 of H is the same as X^2 of H_s.*

PROOF. We prove this by contradiction. Suppose, the vertices of H_s are chosen as the union of all super-vertices that contain some vertex from H. Now, if the lemma does not hold, then there must exist some super-vertex $v_s \in H_s$ containing vertices v_1 and v_2 with $(v_1, v_2) \in E$ such that $v_1 \in H$ and $v_2 \notin H$. Clearly, as v_1 and v_2 are part of the same super-vertex v_s, they should have the same labels. If H_1 is the subgraph obtained by removing v_1 from H, H_1 is connected since H is bi-connected. Similarly, if H_2 is the subgraph obtained by adding v_2 to H, H_2 is also connected as $(v_1, v_2) \in E$. Further, since H is a LMCS, X^2 value of H_1 or H_2 cannot be more than that of H. Applying Lemma 1 on H_1, H (obtained by adding v_1 to H_1) and H_2 (obtained by adding v_2 to H), we get that X^2 value of H_2 is greater than H, which is a contradiction. □

This leads to the following conclusion.

CONCLUSION 2. *All the bi-connected LMCS of G, and hence, also the MSCS and the TSSS of G have equivalent connected subgraphs H_s in G_s. Thus, they can be extracted by running the naïve algorithm over G_s instead of G.*

The result emphasizes the fact that the super-graph construction technique ensures the *correctness* of the results within the sufficient condition of bi-connectivity. However, the condition is not necessary; a LMCS of G which is not bi-connected can still have an equivalent connected subgraph in G_s.

4.3.2 Continuous Labels

The algorithm for constructing the super-graph for the continuous labels (Algorithm 2) is similar to the discrete case. The main

[2]A connected subgraph is *bi-connected* if it remains connected even after the removal of any vertex from it.

Algorithm 3 Construction of a Erdős-Rényi Random Graph

Input: Number of vertices n of a random graph
Output: Random graph $G = (V, E)$

```
V ← {1, ..., n}; E ← ∅; m ← 0
while G = (V, E) is not connected do
    Choose randomly i, j from {1, ..., n}, i ≠ j
    if (i, j) ∉ E then
        Add edge (i, j) to E
        m ← m + 1
    end if
end while
return G
```

difference is in the way a contracting edge is defined, which depends on the X^2 values of the corresponding vertices. If the X^2 of the combined vertex set is greater than the individual vertices, then the edge is *contracting*. In other words, edge $e = (u, v)$ is contracting if and only if $X^2_{(u,v)} > X^2_u, X^2_v$.

After the two vertices along a contracting edge are clubbed, the algorithm takes into account the changed X^2 value of the merged vertex. The classification of an edge as contracting, being dependent on this updated X^2 value, thus, depends on the order in which the edges are processed. A very simple example is clubbing u with either of its neighbors v or w, where, with both additions, the X^2 value is increased so much that it prevents clubbing of the other neighbor. Thus, only one of the edges of u—the one which is processed before the other—is contracting. Thus, the super-graph constructed may not be unique. Hence, unlike Conclusion 1, there is no strong result for the continuous case. There may be cases where a LMCS of G may not have a corresponding connected subgraph in G_s. Nevertheless, since combining the two vertices always necessarily increases the X^2 value, in general, the X^2 values of super-vertices are likely to be much more than the individual connected components of the original graph.

Next, we analyze the size of super-graphs for both the cases.

4.4 Analysis of the Size of Super-graph

In this section, we analyze how the number of vertices n_s of the super-graph G_s depends on the number of edges m and the number of vertices n of G. We consider two random graph models.

In the Erdős-Rényi (ER) model, each edge is independent and equally likely. Algorithm 3 shows the construction of a random ER graph. It starts with n vertices and no edge, and keeps adding edges randomly to the graph till it becomes connected.

Algorithm 4 outlines how a random graph is constructed for the basic Barabási-Albert (BA) model. Besides the total number of vertices in the graph, it requires another parameter d. Each newly added vertex attaches itself to d old vertices. The initialization step requires d number of disconnected vertices. All the remaining $n-d$ vertices are then added one at a time, thus requiring $n-d$ iterations. Each new vertex prefers attachment to an old vertex that is more heavily connected, i.e., the probability of attaching to an old vertex is proportional to its degree.

The next lemma characterizes the expected number of edges in a *connected* random Erdős-Rényi graph.

LEMMA 3. *The expected number of edges required to make a random ER graph having n vertices connected, is less than $n \ln n$.*

PROOF. At each addition of a random edge, the number of components decrease by at most 1. Suppose the random variable X denotes the total number of edges added and X_c denotes the number of edges added to reduce the number of connected components from $c + 1$ to c. Therefore,

$$X = \sum_{c=1}^{n-1} X_c. \tag{14}$$

A random edge $e = (v_1, v_2)$ reduces the number of components by 1 if v_1 and v_2 lie in different connected components. The probability p_c of choosing an edge such that it reduces the number of components from $c + 1$ to c is at least $p_c \geq c/(n-1)$ as for any choice of v_1, each of the c other components must contain at least one vertex in them. Thus, at least c out of $n-1$ different possible choices of v_2 will reduce the number of components to c.

We keep drawing edges randomly as part of X_c till one such edge is obtained. Thus, X_c follows a geometric distribution whose expectation is $1/p_c$. Using the linearity of expectations,

$$E[X] = \sum_{c=1}^{n-1} E[X_c] = \sum_{c=1}^{n-1} \frac{1}{p_c} \leq \sum_{c=1}^{n-1} \frac{n-1}{c} < n \ln n. \tag{15}$$

□

A BA random graph, on the other hand, is always connected, as can be seen from its construction.

We next obtain the probability of a random edge being contracting and analyze the two cases of labeling separately.

4.4.1 Discrete Labels

LEMMA 4. *In a random ER graph with discrete labels, the probability of drawing a contracting edge is more than $1/l$ where l is the number of labels.*

PROOF. Suppose the count vector for the random graph is $\{Y_1, \ldots, Y_l\}$ where $\sum_{i=1}^{l} Y_i = n$. A contracting edge between two vertices having label i can be drawn in $\binom{Y_i}{2}$ ways. Therefore, the total number of ways a contracting edge can be drawn is obtained by summing over all possible labels 1 to l. Since the total number of ways of drawing any edge is $\binom{n}{2}$, the probability of drawing a contracting edge is

$$p_c = \sum_{i=1}^{l} \frac{Y_i(Y_i-1)/2}{n(n-1)/2} > \sum_{i=1}^{l} \frac{Y_i(Y_i-1)}{n^2} > \sum_{i=1}^{l} \frac{Y_i^2}{n^2} - \frac{1}{n}. \tag{16}$$

Since Y_i^2 is a convex function with $\sum_{i=1}^{l} Y_i = n$, using Jensen's inequality, we get $\sum_{i=1}^{l} Y_i^2 \geq \frac{n^2}{l}$. Putting it back in Eq. (16),

$$p_c > \frac{1}{l} - \frac{1}{n} \approx \frac{1}{l}. \tag{17}$$

□

The above result states that if the random graph contains more than $l.n \ln n$ edges, the expected number of contracting edges among these will be more than $n \ln n$. Since all these edges can take part in clubbing together of vertices, using Lemma 3, on average these are sufficient to reduce the number of super-vertices to l. Hence, we conclude the following.

CONCLUSION 3. *For graphs with discrete labels, the number of super-vertices obtained from a sufficiently dense graph, i.e., where the number of edges m is greater than $l.n \ln n$ for n vertices and l labels, is very small and is close to l.*

Since l is a small constant in our problem setting, running the naïve algorithm on the obtained super-graph G_s for graphs with

Algorithm 4 Construction of a Barabási-Albert Random Graph

Input: Number of vertices n of the graph and parameter d
Output: Random graph $G = (V, E)$

```
Initialize V = {V_1, ..., V_d} with d disconnected vertices
for i = d + 1 to n do
    S ← Choose d unique vertices from {V_1, ..., V_{i-1}} with
      probability of choosing a vertex proportional to its degree
    for ∀v in S do
        Add edge (v, V_i) to E and vertex V_i to V
    end for
end for
return G
```

$m > l.n \ln n$ is quite feasible. Moreover, as stated in Conclusion 2, all the bi-connected subgraphs of G have a equivalent subgraph in G_s. Further, the following direct result from [8] for ER model states that if $m = \omega(n \log n)$, then the subgraphs are bi-connected.

LEMMA 5. *In a random ER graph G, if $m = \frac{1}{2}n \ln n + n \ln \ln n + \alpha n + o(n)$ where α is a real constant then G is bi-connected with high probability. Hence, if $m = \omega(n \ln n)$, with high probability, any subgraph H of G is bi-connected.*

PROOF. Please see [8]. □

This implies that for graphs dense enough such that $m = \omega(l.n \ln n)$, our algorithm finds all the MSCS and TSSS without any approximation in linear time.

The next lemma shows that a large enough graph generated by the basic BA model is bi-connected with a high probability and, hence, all the previous results on the ER model follow as well.

LEMMA 6. *A random graph G that follows the basic BA model is bi-connected with high probability.*

PROOF. Assume that the parameters of the random BA graph are n and $d > 1$. Initially, the graph contains d disconnected vertices. In the first iteration, when the $(d+1)^{\text{th}}$ vertex is added, the graph assumes a star topology with the new vertex at the center. When the $(d+2)^{\text{th}}$ vertex is added, two cases may occur. In the first, the new vertex attaches itself to the old d vertices. In this case, the graph becomes bi-connected. In the second, one of the original vertices (say v) is not attached to. Then, the center of the star becomes a cut vertex with v on one side and the rest on the other.

Now consider the case after i iterations, i.e., when the $(d+i)^{\text{th}}$ vertex is added. The graph is *not* bi-connected if and only if it was not bi-connected till the previous iteration and the new vertex did not get attached to it, or in other words, none of the previous vertices from $d+1$ to $d+i$ got attached to v. In every iteration, the number of edges increase by $2d$. Thus, the probability that the i^{th} vertex did not get attached to v after d attachments, is

$$P_i = (1 - 1/(2id))^d \tag{18}$$

For any constant α,

$$(1 - 1/(2\alpha d))^d \leq \lim_{d \to +\infty} (1 - 1/(2\alpha d))^d \leq e^{-\frac{1}{2\alpha}} \tag{19}$$

Hence, the probability that the graph is not bi-connected after the i^{th} iteration is

$$P_{\text{not-bi}}(i) = \prod_{j=1}^{i} P_j \leq \prod_{j=1}^{i} e^{-\frac{1}{2j}} = e^{-\sum_{j=1}^{i} \frac{1}{2j}} < e^{-\frac{1}{2} \cdot \ln i} = \frac{1}{\sqrt{i}} \tag{20}$$

Since the iterations run from 1 to $i = n - d$, $P_{\text{not-bi}} = 1/\sqrt{n-d}$. For large n and a constant $d \ll n$, $P_{\text{not-bi}}$ is quite small. □

Algorithm 5 Reducing the Size of Super-graph

Input: Super-graph $G_s = (V_s, E_s)$, threshold N_θ
Output: Reduced super-graph $G_s = (V_s, E_s)$

```
while G_s.num_vertices > N_θ do
    Choose edge e = (u, v) such that X_u^2 + X_v^2 is minimum
    Add vertex w = Merge(u, v) to V_s, and compute X_w^2
    Add all edges of form (u, x) and (v, x) as (w, x) to E_s
    Remove u, v and e from G_s
end while
return G_s(V_s, E_s)
```

4.4.2 Continuous Labels

For continuous labels, we show that the probability of drawing a contracting edge in a random graph is, again, greater than a constant. As the algorithm is similar except for the definition of a contracting edge, the rest of the analysis is exactly the same.

LEMMA 7. *In a random graph with each edge equally likely and the k-dimensional z-scores following the null hypothesis, the probability of drawing a contracting edge is $1/4$.*

PROOF. Please see Appendix A. □

The algorithm for continuous labels is different from the discrete case as the z-scores and X^2 values keep increasing. Thus, the assumption of z-scores being drawn from a normal distribution is not completely valid. However, as shown later in the experimental section, such an increase does not adversely affect the size of the super-graph obtained and the effect is negligibly small for large k. Thus, the following conclusion still approximately holds.

CONCLUSION 4. *For graphs with continuous labels, the number of super-vertices obtained from a sufficiently dense graph, i.e., where the number of edges $m > 4n \ln n$ for n vertices and k-dimensional z-scores, is small.*

Thus, for sufficiently dense graphs, running the naïve algorithm on the obtained super-graph G_s is feasible.

4.5 Reduction of the Super-graph

For sparse graphs, reducing the super-graph size is necessary. We first state a result that provides the central idea for the reduction.

LEMMA 8. *Suppose X_1^2 and X_2^2 represent the X^2 values of super-vertices v_{s_1} and v_{s_2} in G_s. The X^2 value of the new super-vertex X_c^2 formed by clubbing v_{s_1} and v_{s_2} is upper bounded by $X_1^2 + X_2^2$, and trivially lower bounded by 0.*

PROOF. Suppose the sizes of super vertices v_{s_1} and v_{s_2} are n_1 and n_2 respectively and their count vectors are $\{Y_1, \ldots, Y_k\}$ and $\{Z_1, \ldots, Z_k\}$ respectively.

From the definition of X^2 for discrete labels in Eq. (2),

$$\begin{aligned} X_1^2 + X_2^2 &= \left(\sum_{i=1}^{k} \frac{Y_i^2}{n_1 p_i} - n_1 \right) + \left(\sum_{i=1}^{k} \frac{Z_i^2}{n_2 p_i} - n_2 \right) \\ &\geq \sum_{i=1}^{k} \frac{(Y_i + Z_i)^2}{(n_1 + n_2) p_i} - (n_1 + n_2) = X_c^2. \end{aligned} \tag{21}$$

For continuous labels, suppose $(A_1, \ldots, A_k)$ and $(B_1, \ldots, B_k)$ denote the multi-dimensional z-score values of v_{s_1} and v_{s_2} respectively. The multi-dimensional z-score for the combined vertex

	Low	Moderate	High	Very High
Bio-diversity Richness	A	B	C	D
Disturbance	E	F	G	H
Medicinal	I	J	K	
Economic	L	M	N	

Table 1: Quantization information for the North-East dataset.

along the i^{th} dimension is $\sqrt{\frac{n_1}{n_1+n_2}}A_i + \sqrt{\frac{n_2}{n_1+n_2}}B_i$. From the definition of X^2 for continuous labels in Eq. (8),

$$X_1^2 + X_2^2 = \sum_{i=1}^{k} A_i^2 + \sum_{i=1}^{k} B_i^2 \geq \sum_{i=1}^{k} \left(\sqrt{\frac{n_1}{n_1+n_2}}A_i + \sqrt{\frac{n_2}{n_1+n_2}}B_i \right)^2 = X_c^2. \quad (22)$$

The lower bound for any X^2 value is trivially 0. □

The algorithm for reducing the size of the super-graph (Algorithm 5) is based on Lemma 8. In each iteration of the algorithm, the edge connecting the two vertices that have the least sum of X^2 values is contracted, thereby reducing the number of super-vertices by 1. The corresponding vertices are merged and all their neighboring vertices become neighbors of this new vertex. As a result, in the final solution, the MSCS or TSSS either contains both these vertices or none of them. This may introduce non-optimality as only one of them may have been part of the optimal solution. However, since the vertices that are merged have low X^2 values to start with, they are unlikely to be part of the optimal solution. Even if they are, Lemma 8 ensures that the maximum error is $X_1^2 + X_2^2$. Hence, if X_1^2 and X_2^2 are small, the solution returned by our algorithm is unlikely to deviate a lot from the optimal. This is confirmed by the experiments as well (Section 5).

The contraction of edges is conducted till the number of super-vertices in the super-graph falls below a threshold N_θ. The naïve algorithm is then run on this reduced super-graph. Thus, the threshold N_θ acts as a trade-off between speed and accuracy. The lower the threshold, the lesser is the size of the reduced graph, and the faster the naïve algorithm finishes, but the farther away from the optimal solution it returns.

4.6 Time Complexity of the Algorithm

The first step of super-graph construction requires an iteration over all the edges. If the edge is non-contracting, the total time spent in that iteration is $O(1)$; otherwise, for contracting edges, every iteration in Algorithm 1 and Algorithm 2 may take $O(deg(v_i))$ time. As the vertex v_i is deleted at the end of such an iteration, the number of contracting edges is at most 1 for every vertex. Thus, the time complexity of this step is $O(m + \sum_i deg(v_i)) = O(m)$. If the graph is dense enough such that the number of super-vertices obtained is effectively a constant, the time complexity of the further steps can be ignored. Thus, the total time complexity of the complete algorithm is linear.

Otherwise, there may be at most n_s iterations in the super-graph reduction algorithm (where n_s is the number of vertices in the super-graph) and every step of it requires extracting the minimum X^2 edge. This can be done in $O(\log m_s)$ time by using heaps, where m_s is the number of edges in the super-graph. Thus, in the worst case, the total time complexity is $O(n_s \log m_s)$. The time complexity of the final step of running the naïve algorithm is exponential on the size of the reduced super-graph, which is a user-controlled parameter.

Co-location Rule	Rule Prob.	MSCS (Top-1) Region		
		Ratio (of 1)	Sizes	Labels
I ⇒ H	0.54	0.00	{98}	{0}
C ⇒ H	0.65	0.00	{92}	{0}
I ⇒ D	0.70	1.00	{75}	{1}
I ⇒ A	0.35	0.03	{48, 3, 42}	{0, 1, 0}
L ⇒ A	0.40	0.01	{50, 1, 29}	{0, 1, 0}
E ⇒ B	0.62	0.90	{15, 2, 4}	{1, 0, 1}

Table 2: Inferences from the North-East dataset.

5. EXPERIMENTS

All the simulations were done using the Boost graph library (http://www.boost.org/) in C++ on an Intel(R) Xeon(R) 24-core machine with 2.4 GHz CPU and 194 GB RAM running Linux Debian 6.0.7. We present results on both real (large) graphs and synthetic datasets (averaged over 10 different runs).

5.1 Real Dataset: Discrete

To test the applicability of our method on a real dataset, we used the North-East Biodiversity data formed as the result of a survey by the Indian Space Research Organisation (ISRO) [24]. The area is one of the most bio-diverse regions of India and the survey was done to understand the trends of depletion of natural reserves and bio-reserves due to poaching or other causes. The dataset comprises of 1202 spatial points along with four types of information: (i) Bio-diversity richness index, (ii) Disturbance index, (iii) Medicinal property, and (iv) Economical property. The first two indices, i.e., bio-diversity richness index and disturbance index, were already quantized by domain experts into four labels each, namely, Low, Moderate, High and Very High. We quantized the medicinal properties to three labels based on the (normalized) ranges $[0, 0.4]$, $(0.4, 0.8]$, $(0.8, 1]$. The economical properties were quantized into three labels $[0, 0.65]$, $(0.65, 0.9]$, $(0.9, 1]$. Thus, finally we have a total of 14 quantized values labeled A to N (Table 1).

This information can be used to infer many hidden characteristics, such as bio-diversity hotspot detection, relationships between a highly bio-diverse region and the type of vegetation found there, etc. We extend these inferences to a deeper level by unraveling certain characteristics that cannot be deciphered by just using co-location rule mining. We incorporate the co-location rule as input and mine the contiguous regions in space where the rule is statistically significant, and not just frequent. We only consider rules of size 2, i.e., co-locations including 2 features $\{X, Y\}$ and the rule $X \Rightarrow Y$, since that provides the most basic understanding.

For a rule $X \Rightarrow Y$, the subgraph inducing only those nodes which has a label X is first extracted. A node in this graph is then given a label of 1 if it exhibits the label Y; otherwise it is labeled as 0. Thus, the number of discrete labels here is $l = 2$ (Y present or absent) and the probability of the 1 event is the same as that of the spatial co-location rule $X \Rightarrow Y$. The probability of the 0 event is, thus, one minus this quantity, i.e., the residual value.

When the rule is quite likely (resp. unlikely), our method finds the largest connected contiguous region of 1 (resp. 0) as the subgraph of interest. This is quite intuitive and equivalent to hot-spot detection. More importantly, the connected subgraphs can display some additional interesting structures such as two large regions of 1 (resp. 0) connected by a bridge of 0 (resp. 1) of a smaller size. Without this bridge, the two 1 (resp. 0) regions would not have been connected. This bigger region can have a larger chi-square than any of the individual 1 (resp. 0) regions, thereby attracting attention to the bridge. A simple hotspot detection method would have failed in this scenario.

Table 2 shows some representative results. For each rule, we mention the probability and the characteristics of the top-1 sub-

graph mined using it including the ratio of 1 nodes (i.e., proportion of presence), the number of super-vertices along with the number of original nodes and the labels. The first set of results are contiguous regions of either 1 or 0. For example, the first result consisting of 98 nodes ($I \Rightarrow H$) highlights a region (in the Indian state of Manipur) where there is very high disturbance when medicinal plants are low. The second set of results are more interesting as they highlight the bridges mentioned earlier. For example, the top result for the rule $I \Rightarrow A$ is a bridge of 3 label-1 nodes connecting two regions of label-0 nodes of sizes 48 and 42 (in Manipur again). This unearths two regions of low bio-diversity richness connected by a region of higher bio-diversity when all the regions have low medicinal plants. This provides the domain scientists some target areas to try and unravel interesting actions of nature or external factors. We emphasize the fact that a simple co-location rule mining would not have been able to identify such regions.

The total time required to find the top-5 regions for the largest graph (containing 883 nodes and 6066 edges) is only 0.25 s. Since the graph is not dense, the reduced super-graph contains as many as 96 nodes. This is further reduced to 15 nodes. The two reduction steps require only 0.03 s as the original graph is quite small. Consequently, the time to run the naïve algorithm on the 15-vertex super-graph (0.22 s) is the dominating factor.

We also highlight some interesting results from mining the entire spatial graph considering only two labels at a time. A region of 32 nodes was found in Mizoram where even though the bio-diversity is low, the medicinal plants are found in high quantity. The probability of this combined label (low bio-diversity and high medicinal, i.e., AK) is only 5%. Such low probability regions are extremely hard to find using apriori or co-location rule mining as the search space explodes [21]. Interestingly, the almost same region (except one node) was found to be extremely statistically significant when the attributes considered were bio-diversity (low) and economic plants (high). The probability (4%) of this event AN is even lesser. Similarly, another region consisting of 30 nodes was found in Manipur where the bio-diversity is high in spite of high disturbance (label CG). The probability (6%) is again low. These results showcase the importance of our mining algorithm as it reveals spatial regions that can be extremely valuable to domain scientists for unearthing new processes and findings. Discussion with the authors of [24] indeed confirmed the utility of the method; in particular, the last example highlighted an area that requires more ground-level studies on the causes of high disturbance (such as Jhum cultivation) that may cause depletion of bio-diversity in such a highly biodiverse patch. It is extremely important for the government officials to identify such cases at the earliest, and analyze and restrict such harmful practices. Currently, identifying such areas are done by conducting extensive surveys by teams of scientists. Our method can act as a very nice filter to help them.

5.2 Real Dataset: Continuous

The continuously labeled real dataset is the West Nile virus dataset (WNV) provided by the U.S. Center for Disease Control and Prevention (http://www.cdc.gov/ncidod/dvbid/westnile/). The 3109 different counties in USA act as vertices of the graph. Two counties are connected by an edge if they share a border. The label on a vertex is the density of human cases infected by WNV in 2011, i.e., number of cases per unit area. We use the *Weighted Z-value* and *Average Difference* algorithms [16] that take into account the difference in densities between the county and its neighbors to assign the z-score values. The Weighted Z-value approach normalizes the difference of the density of a county from the weighted average of the density of its neighbors while the Avg Diff approach weights

	County	Z-score	X^2	Density	Avg. Dens. Neighbors
1	Dist. of Columbia	+47.05	2213.53	0.0776	0.0051
2	Prince George's	-11.47	131.53	0.0007	0.0110
3	St. Louis City	+10.05	100.91	0.0173	0.0007
4	Alexandria	-9.76	95.27	0.0000	0.0232

Table 3: Top counties based on Weighted Z-value approach.

	County	Z-score	X^2	Density	Avg. Dens. Neighbors
1	Dist. of Columbia	+44.43	1974.04	0.0776	0.0051
2	Arlington City	+13.03	169.67	0.0136	0.0132
3	Prince George's	-11.06	122.34	0.0007	0.0110
4	Montgomery	-10.52	110.58	0.0063	0.0132

Table 4: Top counties based on Avg Diff approach.

sum of differences between the density of a county with its corresponding neighbor to obtain the raw scores which are normalized using the standard technique of Eq. (4) to obtain z-scores. The weights in both the cases are assigned using the inverse centroid distance and the length of common border between a county and its neighbor [16].

The top results based on the Weighted Z-value and Avg Diff methods are shown in Table 3 and Table 4 respectively. The District of Columbia (DC) ranks the highest in both. The summary of the corresponding top results for our algorithm are shown in Table 5 and Table 6 respectively. Not very surprisingly, the DC county is the most significant for both the cases. The second results are regions composed of the top counties as well. The third result for Avg Diff is more interesting as it is composed of 7 nodes, none of which is very high ranking by itself, but together form a significant region. We emphasize the fact that without the ability to mine connected subgraphs, this region could never have been found.

The total time required to find the top-10 regions is 66 s. Since there are 3109 nodes and only 8871 edges, the graph is not dense, and the reduced super-graph contains as many as 373 nodes. This is further reduced to 20 nodes. The entire reduction requires only 4 s as the original graph is quite small. Consequently, the time to run the naïve algorithm on the 20-vertex super-graph (62 s) dominates.

5.3 Real Large Datasets

To test the scalability of our algorithm on real graphs that are *not* dense, we used 4 large graphs from the SNAP database at https://snap.stanford.edu/data/. The details are mentioned in Table 7. Continuous labeling was used. The degree of a node was normalized by subtracting the average degree of the graph and scaled by the standard deviation to get the z-score for that node.

The running times are shown in Figure 2. The reduction of the super-graph requires the largest amount of time, especially when the graph is very sparse. The threshold of reduction was kept to 20 nodes for all the datasets. Although the Orkut graph is larger than the LiveJournal one (comparable number of nodes but almost 4 times more number of edges) and, hence, requires a larger time for super-graph conversion (30 minutes against 4 minutes), the overall running time is much better (4 hours against 16 hours) due to its higher density which produces a much smaller super-graph. This shows that our algorithms are able to handle million-scale nodes with close to billion-scale edges.

5.4 Synthetic Datasets

The first set of experiments on synthetic data calibrates the size of the super-graphs after the reduction. The graphs are generated synthetically according to the ER and BA models. For discretely labeled graphs, the labels are drawn uniformly randomly from the total number of possibilities. The multi-dimensional z-scores for

	Counties	Size	Z-score	X^2
1	Dist. of Columbia	1	+47.05	2213.53
2	Prince George's, Alexandria, Montgomery	3	-15.07	226.96
3	St. Louis City	1	+10.05	100.91

Table 5: Significant subgraphs for Weighted Z-value.

	Counties	Size	Z-score	X^2
1	Dist. of Columbia	1	+44.43	1974.04
2	Prince George's, Alexandria, Montgomery, Arlington City	4	-21.84	477.39
3	New York, Hudson, Richmond, Kings, Bronx, Nassau, Queens	7	+10.05	100.91

Table 6: Significant subgraphs for Avg Diff.

continuous labels are drawn from the $N(0, 1)$ distribution. The results for synthetic graphs on the two models are very similar. Therefore, only one of them is shown as a representative.

As shown in Figure 3a, the number of super-vertices reduce drastically with the number of edges for discrete labels. When the number of edges is large (around $(l/2)n \ln n$), the number of super-vertices obtained drops to exactly l as predicted by the theory. For graphs that are not sufficiently dense, the number of super-vertices can be large, although in most cases, it falls below 20, thereby making it feasible to run the naïve algorithm.

Figure 3b shows the amount of time required for reduction to the super-graph. For a fixed n, it grows linearly with m. We have omitted showing the time to obtain the significant subgraphs from this reduced super-graph as that is in the order of $2^l (= 32)$ which is a negligible constant (in milliseconds).

The experiments for the continuous case (graph omitted) show the same trend. Around $4n \ln n$ edges, the number of super-vertices decrease and saturate to a small constant. The absolute running times are larger.

Figure 4 profiles the same behavior across different number of discrete labels. The curves tally nicely with the theoretical prediction of the super-graph being reduced to l nodes. The running times grow linearly with m. There is little difference among different l's.

Figure 5 plots the number of super-vertices against different values of k for the continuous case. For values of $k > 1$, there is little difference in the curves. For $k = 1$, the convergence happens when the graph becomes sufficiently dense, i.e., the number of edges goes past $4n \ln n$. This empirically confirms the result to be invariant of k, as shown in Lemma 7. There is little difference in running times for different k's (graph omitted). Once more, we have omitted the times required for running the naïve algorithm on the reduced super-graphs (which for 15 super-vertices is ≈ 2 s).

5.5 Quality Results for Sparse Graphs

The final set of experiments shows the quality of results when the original graph is *not* sufficiently dense, thereby introducing non-optimality in the subsequent reduction.

Figure 6 shows how the chi-square and the time changes when the super-graph is reduced further. The x-axis shows the reduction in the amount of vertices *from* the super-graph. For example, in Figure 6a, the original super-graph starts with 22 super-vertices (indicated by $n_{rg} = 22$), and we reduce it progressively to 2 super-vertices. Thus, the last reduction value is marked by 20 which translates to $n_{rg} = 2$. The y-axis shows the *ratios* of the chi-square and the time obtained as a factor of the optimal values. The optimal values are obtained when no reduction of the super-graph is done, and hence, the ratios start with 1.

Both figures show that the reduction in time is substantial when the number of super-vertices are reduced. This is in tune with the fact that the number of connected subgraphs is exponential with the number of vertices. However, the corresponding decrease in the chi-square value is almost negligible – it is less than 1% for the discrete case and 4% for the continuous case. (In most runs, the chi-square value does not drop at all. We are omitting those results.) This, therefore, justifies our method of reducing the super-graph further without sacrificing much on the accuracy.

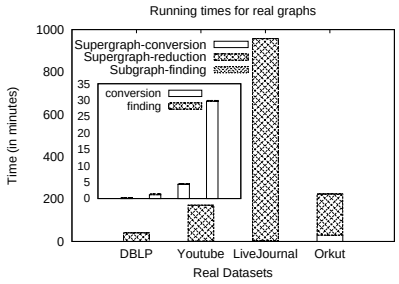


Figure 2: Running time for large real graphs.

Dataset	Nodes	Edges	Avg. Degree
com-DBLP	317,080	1,049,866	3.31
com-Youtube	1,134,890	2,987,624	2.63
com-LiveJournal	3,997,962	34,681,189	8.67
com-Orkut	3,072,441	117,185,083	38.14

Table 7: Large real graphs.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we addressed the problem of finding statistically significant connected subgraphs in a vertex-labeled graph where the labels are either discrete or continuous. Since the number of connected subgraphs is exponential in the number of vertices, the naïve algorithm on the entire graph is impractical. Consequently, we designed an efficient algorithm that converts the graph into a super-graph, and if necessary, reduces the size further. We highlighted in detail one real application for each of the cases and showed that our algorithms scale well and are practical for large real graphs as well.

In future, other sub-structures or directed and edge-weighted graphs can be focused upon. Further, our method may also be successfully applied on other important applications including community detection and dense subgraph mining. Finally, the naïve method of exploring all possible subgraphs can be targeted for improvement.

7. REFERENCES

[1] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[2] S. Barua and J. Sander. SSCP: Mining statistically significant co-location patterns. In *STD*, pages 2–20, 2011.

[3] S. Barua and J. Sander. Mining statistically significant co-location and segregation patterns. *TKDE*, 99(pre):1, 2013.

[4] Y. Chi, Y. Yang, and R. Muntz. Indexing and mining free trees. In *ICDM*, pages 509–512, 2003.

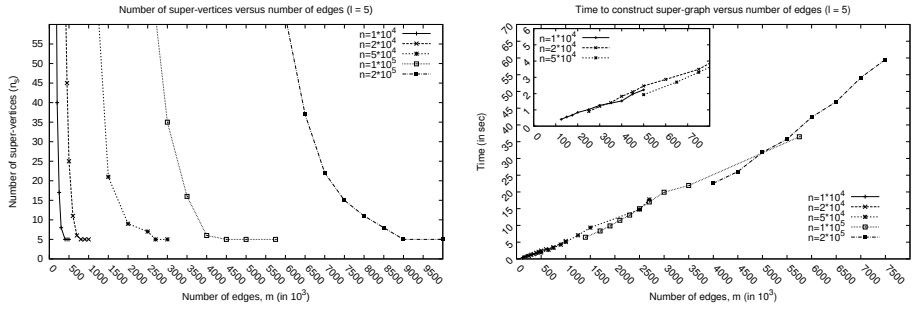
[5] A. Denise, M. Régnier, and M. Vandenbogaert. Assessing the statistical significance of overrepresented oligonucleotides. In *WABI*, pages 537–552, 2001.

[6] N. Durak, A. Pinar, T. G. Kolda, and C. Seshadhri. Degree relations of triangles in real-world networks and graph models. In *CIKM*, pages 1712–1716, 2012.

[7] E. Edgington and P. Onghena. *Randomization Tests*. Marcel Dekker, 1995.

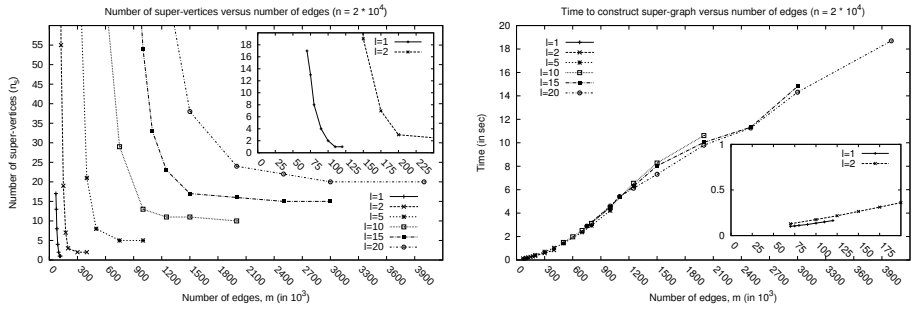
[8] P. Erdős and A. Rényi. On the strength of connectedness of a random graph. *Acta Mathematica Scientia Hungary*, 12:261–267, 1961.

[9] P. Erdös and A. Rényi. On random graphs, I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.



(a) Super-vertices (b) Time

Figure 3: Varying number of edges against number of vertices for discrete labels (Barabási-Albert).



(a) Super-vertices (b) Time

Figure 4: Varying number of edges against number of discrete labels (Erdős-Rényi).

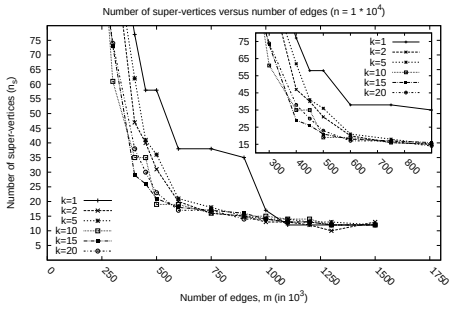


Figure 5: Varying number of edges against dimension of continuous labels (Barabási-Albert).

[10] R. Frank, W. Jin, and M. Ester. Efficiently mining regional outliers in spatial data. In *SSTD*, pages 112–129, 2007.

[11] H. He and A. Singh. Graphrank: Statistical modeling and mining of significant subgraphs in the feature space. In *ICDM*, pages 885–890, 2006.

[12] R. Hogg, A. Craig, and J. McKean. *Introduction to Mathematical Statistics*. Pearson Education, 2004.

[13] P. Holme and B. J. Kim. Growing scale-free networks with tunable clustering. *Physical Review E*, 65(2):026107, 2002.

[14] Y. Huang, J. Pei, and H. Xiong. Mining co-location patterns with rare events from spatial data sets. *GeoInformatica*, 10(3):239–260, 2006.

[15] H. Jiang, J. Cheng, D. Wang, C. Wang, and G. Tan. A general framework for efficient continuous multidimensional top-k query processing in sensor networks. *IEEE Trans. Parallel Distrib. Syst.*, 23(9):1668–1680, 2012.

[16] Y. Kou, C.-T. Lu, and D. Chen. Spatial weighted outlier detection. In *SDM*, pages 613–617, 2006.

[17] J. Lijffijt, P. Papapetrou, and K. Puolamäki. A statistical significance testing approach to mining the most informative set of patterns. In *DMKD*, pages 1–26, 2012.

[18] M. E. J. Newman, S. H. Strogatz, and D. J. Watts. Random graphs with arbitrary degree distributions and their applications. *Physical Review E*, 64(2):026118, 2001.

[19] J. Pei, D. Jiang, and A. Zhang. Mining cross-graph quasi-cliques in gene expression and protein interaction data. In *ICDE*, pages 353–354, 2005.

[20] L. Popa, A. Rostamizadeh, R. Karp, C. Papadimitriou, and I. Stoica. Balancing traffic load in wireless networks with curveball routing. In *MobiHoc*, pages 170–179, 2007.

[21] S. Ranu and A. Singh. Graphsig: A scalable approach to mining significant subgraphs in large graph databases. In *ICDE*, pages 844–855, 2009.

[22] T. Read and N. Cressie. *Goodness-of-Fit Statistics for Discrete Multivariate Data*. Springer, 1988.

[23] M. Régnier and M. Vandenbogaert. Comparison of statistical significance criteria. *J. Bioinf. & Comp. Bio.*, 4:85–97, 2006.

[24] P. Roy and S. Tomar. Biodiversity characterization at landscape level using geospatial modelling technique. *Biological Conservation*, 95(1):95–109, 2000.

[25] M. Sachan and A. Bhattacharya. Mining statistically significant substrings using the chi-square statistic. *PVLDB*, 5(10):1052–1063, 2012.

[26] J. Scott, T. Ideker, R. M. Karp, and R. Sharan. Efficient algorithms for detecting signaling pathways in protein interaction networks. *J. Comp. Bio.*, 13(2):133–144, 2006.

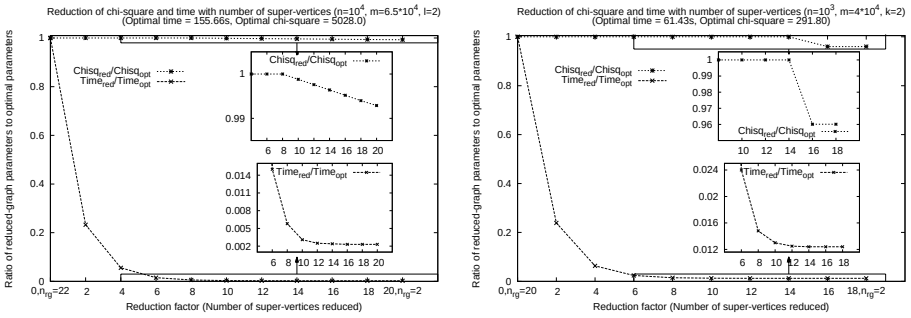
[27] S. Shekhar and Y. Huang. Discovering spatial co-location patterns: A summary of results. In *SSTD*, pages 236–256, 2001.

[28] S. Shekhar, C.-T. Lu, and P. Zhang. Detecting graph-based spatial outliers: algorithms and applications (a summary of results). In *KDD*, pages 371–376, 2001.

[29] D. Wang, W. Ding, H. Z. Lo, T. F. Stepinski, J. Salazar, and M. Morabito. Crime hotspot mapping using the crime related factors – a spatial data mining approach. *Appl. Intell.*, 39(4):772–781, 2013.

[30] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):409–10, 1998.

[31] K. Wongpanya, K. Sripimanwat, and K. Jenjerapongvej. Simplification of frequency test for random number



(a) Discrete case (b) Continuous case

Figure 6: Reduced parameters against number of vertices in reduced super-graph (Erdős-Rényi).

generation based on chi-square. In *AICT*, pages 305–308, 2008.

[32] W. Xing and A. A. Ghorbani. Weighted pagerank algorithm. In *CNSR*, pages 305–314, 2004.

[33] X. Yan, H. Cheng, J. Han, and P. Yu. Mining significant graph patterns by leap search. In *SIGMOD*, pages 433–444, 2008.

[34] N. Ye and Q. Chen. An anomaly detection technique based on a chi-square statistic for detecting intrusions into information systems. *Quality and Reliability Engineering International*, 23, 2001.

[35] C. H. You, L. B. Holder, and D. J. Cook. Temporal and structural analysis of biological networks in combination with microarray data. In *CIBCB*, pages 62–69, 2008.

APPENDIX

A. PROOF OF LEMMA 7

PROOF. We first prove it explicitly for $k = 1$ and then extend the argument for any general k using the principle of symmetry.

Suppose the multi-dimensional z-score values of two adjacent vertices v_1 and v_2 of sizes s_1 and s_2 respectively are $(Z_{11}, \ldots, Z_{1k})$ and $(Z_{21}, \ldots, Z_{2k})$ respectively. Using Eq. (8), the chi-square values of the two vertices, $X^2_{v_1}$ and $X^2_{v_2}$, are

$$X^2_{v_1} = \sum_{i=1}^{k} (Z_{1i})^2 \text{ and } X^2_{v_2} = \sum_{i=1}^{k} (Z_{2i})^2. \tag{23}$$

Using Eq. (6), the combined chi-square value, $X^2_{(v_1,v_2)}$, is

$$X^2_{(v_1,v_2)} = \sum_{i=1}^{k} \frac{(\sqrt{s_1}Z_{1i} + \sqrt{s_2}Z_{2i})^2}{s_1 + s_2}. \tag{24}$$

For an edge to be contracting, this combined z-score value should be greater than the individual z-scores, i.e.,

$$X^2_{(v_1,v_2)} \geq X^2_{v_2} \Rightarrow \sum_{i=1}^{k} \frac{(\sqrt{s_1}Z_{1i} + \sqrt{s_2}Z_{2i})^2}{s_1 + s_2} \geq \sum_{i=1}^{k} (Z_{2i})^2$$

$$\Rightarrow \sum_{i=1}^{k} (s_1 Z^2_{1i} - s_1 Z^2_{2i} + 2\sqrt{s_1 s_2} Z_{1i} Z_{2i}) \geq 0 \Rightarrow \sum_{i=1}^{k} A_i \geq 0 \tag{25}$$

$$X^2_{(v_1,v_2)} \geq X^2_{v_1} \Rightarrow \sum_{i=1}^{k} \frac{(\sqrt{s_1}Z_{1i} + \sqrt{s_2}Z_{2i})^2}{s_1 + s_2} \geq \sum_{i=1}^{k} (Z_{1i})^2$$

$$\Rightarrow \sum_{i=1}^{k} (s_2 Z^2_{2i} - s_2 Z^2_{1i} + 2\sqrt{s_1 s_2} Z_{1i} Z_{2i}) \geq 0 \Rightarrow \sum_{i=1}^{k} B_i \geq 0 \tag{26}$$

where A_i and B_i stand for the longer expressions. Now, a given edge is contracting if and only if both the conditions $\sum_{i=1}^{k} A_i \geq 0$ and $\sum_{i=1}^{k} B_i \geq 0$ hold simultaneously.

We first show that for $k = 1$, Eq. (25) and Eq. (26) are together satisfied with probability $1/4$. We simplify the conditions further by substituting $R = Z_{11}/Z_{21}$ and $s = \sqrt{s_2/s_1}$. Using these substitutions,

$$A_1 \geq 0 \Rightarrow s_1 Z^2_{11} - s_1 Z^2_{21} + 2\sqrt{s_1 s_2} Z_{11} Z_{21} \geq 0$$
$$\Rightarrow s_1 Z^2_{21}(R^2 - 1 + 2sR) \geq 0 \tag{27}$$
$$B_1 \geq 0 \Rightarrow s_2 Z^2_{21} - s_2 Z^2_{11} + 2\sqrt{s_1 s_2} Z_{11} Z_{21} \geq 0$$
$$\Rightarrow s_1 s Z^2_{21}(s - sR^2 + 2R) \geq 0 \tag{28}$$

Since Z_{11} and Z_{12} follow a continuous distribution, they are non-zero with probability 1. Thus, satisfying $A_1 \geq 0$ and $B_1 \geq 0$ is equivalent to finding the solutions of the quadratic equations $R^2 + 2sR - 1 \geq 0$ and $sR^2 - 2R - s \leq 0$. It can be shown that the range of values of R satisfying both the quadratic equations is

$$(\sqrt{s^2 + 1} - s) \leq R \leq (\sqrt{s^2 + 1} + 1)/s. \tag{29}$$

Since both Z_{11} and Z_{21} are i.i.d. random variables and follow $N(0, 1)$, their ratio R follows the Cauchy $(0, 1)$ distribution [12] whose c.d.f. is $F(x) = \frac{1}{\pi} \arctan(x) + \frac{1}{2}$. Hence, the probability of R being in the given range is

$$\frac{1}{\pi} \left[\arctan\left(\frac{\sqrt{s^2 + 1} + 1}{s} \right) - \arctan(\sqrt{s^2 + 1} - s) \right]. \tag{30}$$

Since $s > 0$, substituting $s = \tan\phi$ with $\phi \in (0, \pi/2)$, we get

$$\begin{aligned} &P(A_1 > 0 \text{ and } B_1 > 0) \\ &= [\arctan((1 + \sec\phi)/\tan\phi) - \arctan(\sec\phi - \tan\phi)]/\pi \\ &= [\arctan(\cot(\phi/2)) - \arctan(\tan(\pi/4 - \phi/2))]/\pi \\ &= [(\pi/2 - \phi/2) - (\pi/4 - \phi/2)]/\pi = \frac{1}{4}. \end{aligned} \tag{31}$$

Thus, for $k = 1$, we have shown that the probability is $1/4$.

For any general k, the problem reduces to finding the probability of $\sum A_i \geq 0$ and $\sum B_i \geq 0$ simultaneously. Given the null hypothesis, since each Z_{1i} and Z_{2i} are i.i.d. standard normal random variables $N(0, 1)$, the random variables A_i and B_i are individually (but not together) i.i.d. as well. Also, we observe that the p.d.f. of each A_i and B_i is an even function as substituting $Z_{1i} \rightarrow -Z_{2i}$ and $Z_{2i} \rightarrow Z_{1i}$ reverses their signs. Further, the probability of the outcomes remains the same due to the p.d.f. of $N(0, 1)$ being an even function as well. Thus, for $k = 1$, we can say that $P(A_i \geq 0)$ and $P(B_i \geq 0)$ are both $1/2$ individually. As the p.d.f. of both A_i and B_i are even functions, given any outcome, the probability of the new outcome by replacing all $A_i \rightarrow -A_i$ and $B_i \rightarrow -B_i$ remains the same. However, this reverses the sign of $\sum A_i$ and $\sum B_i$. So, $P(\sum A_i \geq 0) = P(\sum B_i \geq 0) = 1/2$. Hence, the probability of these two conditions remain the same for any k.

Thus, although we have proved the theorem formally for $k = 1$, for other cases, we have intuitively shown that the probability of an edge being contracting is invariant of k and is, hence, $1/4$. □

The experiments in Section 5.4 show that the cases for $k > 1$ are better than $k = 1$ or are equal. The algorithm remains scalable as long as this is greater than or equal to a constant.