# Mining Uncertain Data with Probabilistic Guarantees

Akhil R, Mani vannan S, Kathiravan C, Nishanthi S
*Department of Computer Science and Engineering*
*K.S.R. College of Engineering, Anna University, Chennai.*
akhilrajan5593@gmail.com
mani.cser@gmail.com
kathiravanck1993@gmail.com
nishasaravanan.s@gmail.com

***Abstract-*** **We study the problem of clustering probabilistic graphs. Similar to the problem of clustering standard graphs, probabilistic graph clustering has numerous applications, such as finding complexes in probabilistic protein-protein interaction (PPI) networks and discovering groups of users in affiliation networks. We extend the edit-distance-based definition of graph clustering to probabilistic graphs. We establish a connection between our objective function and correlation clustering to propose practical approximation algorithms for our problem. A benefit of our approach is that our objective function is parameter-free. Therefore, the number of clusters is part of the output. We also develop methods for testing the statistical significance of the output clustering and study the case of noisy clustering. Using a real protein-protein interaction network and ground-truth data, we show that our methods discover the correct number of clusters and identify established protein relationships. Finally, we show the practicality of our techniques using a large social network of Yahoo! users consisting of one billion edges.**

## I. INTRODUCTION

We focus on the problem of partitioning a probabilistic graph into clusters. This is a fundamental problem for probabilistic graphs, just as it is for deterministic graphs. Partitioning a probabilistic graph into clusters has many applications such as finding complexes in protein-protein interaction networks and communities of users in social networks. A straightforward approach to clustering probabilistic graphs is to heuristically cast the probability of every edge into a weight and apply existing graph- clustering algorithms [4] on this weighted graph. This approach is problematic; not only there is no meaningful way to perform such a casting, but also there is no easy way to present a principled approach to probabilistic graph clustering. Motivated by the possible- world semantics of probabilistic databases [5], [6] and probabilistic graphs [7], [8], we treat every probabilistic graph G as a generative model for deterministic graphs. Each such deterministic graph is a possible world of G and is associated with a probability to be generated.

Consider a deterministic graph G and a partitioning, C, of the nodes in V.A clustering objective function quantifies the cost of the clustering C with respect to G. The possible world semantics dictate that the cost of a clustering C for a probabilistic graph G is the expected value.

Although this generalization is natural, it raises computational concerns. For instance, evaluating using the definition of expectation requires considering all, exponentially many, possible worlds of G. Further, the expectation of well-established clustering objective functions (e.g., the maximum cluster diameter), is infinite since, typically, there exist possible worlds where parts of the graph are disconnected. Therefore, new definitions of the clustering problem in probabilistic graphs are necessary.

We view clustering C as a cluster graph, i.e., a graph consisting of disconnected cliques. Our optimization function is the edit distance between G and cluster graph C. In other words, it is the number of edges that we need to add and remove from G to get C. Given a probabilistic graph G, we define PCLUSTEREDIT as the problem of finding the cluster graph C that has the minimum expected edit distance from G. Our problem is a generalization of the CLUSTEREDIT problem introduced by Shamir et al. [9] for deterministic graphs.

Our framework for clustering probabilistic graphs has many desirable features. First, our objective function can be computed in polynomial time. That is, we avoid its evaluation for every possible world of G. Further, the value of our objective function is never infinity; it is bounded by the number of node pairs in the graph. Also, the variance of our objective to any clustering depends solely on the graph and not on the specific clustering. This indicates that the edit distance is a robust objective for clustering probabilistic graphs. Finally, our objective is parameter free, hence, the number of clusters is part of the output

## II. TRANSACTIONS on KNOWLEDGE and DATA ENGINEERING

In our contributions we give a new definition of clustering in probabilistic graphs based on graph edit distance and we establish a connection between our problem and correlation clustering [10]. We exploit this connection in order to design efficient algorithms for clustering large probabilistic graphs. Our algorithms also provide approximation guarantees. Further,

we establish a framework to compute deviations of a random world from the discovered clustering and to test the statistical significance of the resulting clustering. Also, we study versions of our problem where the target clustering is itself noisy. Our experimental evaluation on a probabilistic protein-protein interaction network shows that our algorithms discover the correct number of clusters and identify known co-complex relationships among proteins. We also show that they can efficiently cluster a probabilistic social network from Yahoo! Groups with one billion edges.

Discussion: Our framework outputs a partition of the nodes of the probabilistic graph into groups. That is, it does not support probabilistic membership of nodes to clusters. Extending the proposed framework to find probabilistic assignments to clusters or identify overlapping clusters is material for future work.

We define the basic version of the PCLUSTEREDIT problem and give algorithms for solving it. Some extensions to the basic PCLUSTEREDIT problem are presented. We present a thorough experimental evaluation on real data sets. To the best of our knowledge, we are the first to define and study the problem of clustering probabilistic graphs using the possible-worlds semantics. However, uncertain data management and graph mining has motivated many studies in the data mining and database community. We highlight some of this work here.

## III. GRAPH and PROBABILISTIC-GRAPH MINING

Clustering and partitioning of deterministic graphs has been an active area of research [11], [12], [13]. For an extensive survey on the topic see [4] and the references there in. Most of these algorithms can be used to handle probabilistic graphs, either by considering the edge probabilities as weight, or by setting a threshold value to the probabilities of the edges and ignoring any edge with probability below this thresh- old. The disadvantage of the first approach is that once probabilities are interpreted as weights, then no other weights can be taken into consideration (unless the probabilities are multiplied with edge weights—in which case this composite weight has no interpretation). The disadvantage of the second approach is that there is no principled way of deciding what the right value of the threshold is. Although both the above methodologies would result in an algorithm that would output some node clustering, this algorithm, contrary to ours, would not optimize an objective defined over all possible worlds of the input probabilistic graph.

Further, various graph mining problems have been studied recently assuming uncertain graphs [5], [12], [15], [10], [7], [8], [9]. For example, Hintsanen and Toivonen [15] looked at the problem of finding the most reliable sub graph, and Zou [3] considered the problem of finding frequent sub graphs of an input probabilistic graph. More recently, Potamias [7] proposed new robust distance functions between nodes in probabilistic graphs that extend shortest path distances from deterministic graphs and proposed methods to compute them efficiently. The problem of finding shortest paths in probabilistic graphs based on transportation networks has also been considered [7], [10]. The intersection between the above methods and ours is that all of them deal with probabilistic graphs. However, the graph-clustering task under the possible- world's semantics has not yet been addressed by researchers in probabilistic graph mining.

## IV.DATA MINING on UNCERTAIN DATA

Data mining over uncertain data has also received a lot of attention. Several classical data-mining problems have been revisited in the context of uncertainty. Examples include clustering of relational data, frequent-pattern mining and evaluating spatial queries. All these works are tailored to model uncertain multidimensional data where uncertainty is associated either with the location of the data points in the space or with the actual existence of the data point in the data set. One could think of defining probabilistic-graph clustering using the same ideas as those used for clustering uncertain multidimensional data. After all, the main idea there is to consider as an objective the expectation of standard clustering optimization criteria across all possible worlds [7]. It may be tempting to try and use the same definitions for probabilistic graphs, particularly since standard clustering objectives (e.g., k-centre or k-median) can be optimized in deterministic graphs. However, there is a fundamental difficulty with such clustering definitions in the probabilistic-graph setting: since there are many worlds where parts of the graph are disconnected, the distance (or proximity) of a node to any of the existing clustering centres can be infinity. Indeed, for nontrivial probabilistic graphs, there is always a nonzero probability of having a node with infinite distance to all the cluster centres. In that case, the optimization function becomes infinity. Therefore, new definitions of the clustering problem in probabilistic graphs are necessary. This paper addresses this challenge.

## V.PROBABILISTIC DATABASES

Probabilistic databases is an- other active research area, mostly focusing on the development of methods for storing, managing, and querying probabilistic data [12]. There exists fundamental work on the complexity of query evaluation on such data [6], on the computation of approximate answers to queries [11], [13] and on efficient evaluation of top-k queries [5], [6], [7], [8], [9]. Although we borrow the possible-world semantics pioneered by the probabilistic-database community, the computational problems we address here are different and require the development of new methodologies.

## V1.THE PROBABILISTIC GRAPH MODEL

Similar to deterministic graphs, probabilistic graphs may be undirected or directed and carry additional labels on the edges (such as weights). We focus on undirected

Probabilistic graphs. Our model assumes independence among edges.

We represent a probabilistic graph G using tuple corresponds to the set of nodes in G, and we assume that. P maps every pair of nodes to a real number in uv represents the probability that edge exists. For weighted graphs we also use W to denote the weight associated with every edge. In this paper we focus on unweighted probabilistic graphs. In this case, we

 

represent the probabilistic graph as G. For a probabilistic graph G we define its complement to be the probabilistic graph G0.

One can think of a probabilistic graph as a generative model for deterministic graphs.

## VII.DETERMINISTIC CLUSTER GRAPHS

In this section, we formulate the problem of clustering in probabilistic graphs as an optimization problem. First, we define the edit distance between two graphs. Then, we generalize this definition for probabilistic graphs and use it to define our graph-clustering problem.

A central notion for the remainder of our analysis is the cluster graph. A cluster graph is a special deterministic graph that consists of vertex-disjoint disconnected cliques.

## VIII.ALGORITHMS

One could solve the PCLUSTEREDIT problem using the following heuristic. First, decide on a threshold and ignore all edges that exist with probabilities below this threshold. Then, leverage an existing algorithm for clustering deterministic graphs to partition the points into clusters. Such an algorithm would have the following disadvantages: first, the choice of threshold seems arbitrary. Second, it will be very hard to prove that such an algorithm is an approximation algorithm for the PCLUSTEREDIT problem. In fact, it is not clear what is the objective that such an algorithm optimizes neither is it clear what the expectation of this objective over all possible worlds is.

Our algorithms for the PCLUSTEREDIT problem exploit the connection between our problem and the problems of correlation clustering [10] and clustering aggregation [12], [13]. Therefore, before describing the algorithms themselves we give a brief introduction CORRELATION CLUSTERING and highlight its connection to PCLUSTEREDIT.

We can compute the value of E by creating random instances and counting the fraction of instances in which the cost of the original cluster graph C is less than the cost of its randomized versions. The E of a clustering takes values in the closer this value is to 1 the larger the confidence that clustering C is statistically significant.

## IX.ALTERNATIVE CLUSTERING DEFINITIONS

We briefly discuss some alternative clustering optimization functions; CLUSTEREDIT for the most-probable world. This problem can be mapped to an unweighted instance of the Correlation Clustering. In particular, the most probable world can be easily constructed

## X.EXPERIMENTS

In this section, we present an experimental evaluation of our methodology. Our experiments verify that our algorithms output clustering that is meaningful and statistically significant. We also demonstrate that our approximation algorithm pKwikCluster scales well and can handle real-world social networks with more than one billion edges. Finally, we show that pKwikCluster scales linearly in synthetic scale-free graphs. We ran all experiments on a Linux server with eight AMD Opteron processors with 64 GB memory. All methods have been implemented in C++.

## X.PROTEIN-PROTEIN INTERACTION NETWORK: THE CORE DATA SET

For this experiment, we used the core interaction network provided by Krogan et al. The network consists of 2,708 nodes that correspond to proteins. There are 7,123 protein interactions that correspond to the network edges. The edge probabilities encode the confidence of the interaction between nodes. We refer to this network as CORE and we denote the underlying probabilistic graph as G.

CORE Data Set: Summary of Results in Terms of Edit Distance and Wallclock Time of all algorithms edge in the network with probability less than 0.27. Also, about 20 percent of the edges have probability greater than 0.98. The edge probabilities are uniformly spread in the remaining range l⁄20:27; 0:98Š. The data set exhibits power- law degree distribution, short paths, and high clustering coefficient.

## XI. QUANTITATIVE PERFORMANCE of ALGORITHMS

The goal of this experiment is to compare the performance of our algorithms (pKwikCluster, Agglomerative, and Furthest) with respect to the quality of the solutions they produce as well as their running times. The quality of a solution is measured in terms of our objective function, i.e., the expected edit distance of the output cluster graphs from the input probabilistic graph.

Apart from our algorithms, we also report results for three other methods: pCast, Balls, and MCL. The past algorithm is a straightforward adaptation of the Cast heuristic [14] for probabilistic graphs. The Balls algorithm is a variant of pKwikCluster, proposed by Gionis et al [13] for solving the clustering-aggregation problem. Finally, the MCL procedure is a two-parameter heuristic clustering technique based on random walks and matrix multiplications. Krogan et al. [14] report that they have manually tuned the two parameters of MCL to optimize the utility of the algorithm's results. We use this optimized output provided by Krogan et al, which is available in the online supplemental material. We refer to this cluster- ing as Reference.

We implemented pKwikCluster and pCast. For Agglomerative, Furthest, Balls we used the code provided by Gionis et al. [13], which is available online.2 Both pCast and Balls require the setting of one parameter. For pCast, we set the value of the parameter to 0.7 after experimentally observing that it minimizes the expected edit distance. We set the parameter of Balls to its default value 0.17.

The running time of these algorithms for our implementations are also reported. Since pKwikCluster is a randomized algorithm, we ran it for 100 times and we report the best value of our objective achieved over these runs. In terms of edit distance, pKwikCluster and Agglomerative out-perform the rest of the algorithms yielding solutions with edit distance 4,194 and 3,420 respectively. In this small data set, pKwikCluster takes just 5ms for a single run. Even though Agglomerative produces the highest quality clustering in CORE, its running time is 10 s. For the Reference clustering we

do not report the running time since we utilize the reported results of MCL directly [3], without running the algorithm ourselves. However, MCL involves matrix multiplications so its complexity is at least quadratic to the number of nodes.

## XII.QUALITATIVE PERFORMANCE of ALGORITHMS

In this experiment, we validate the output of our methods with respect to a known ground truth. Our results indicate that our techniques not only produce meaningful clustering but also discover the correct number of clusters.

We use the MIPS database of protein complexes as ground truth [15]. MIPS complexes define co-complex relationships among proteins; a co-complex relationship is a pair of proteins that both belong to the same complex. Among the co-complex relationships present in MIPS, we only keep the ones that occur in G.Thus, we end up with a ground truth of 5,380 pairs of proteins. We emphasize that the complexes from MIPS correspond to overlapping sets of proteins; i.e., proteins may belong to more than one complex. On the other hand, the output clustering of the methods reported here are partitions, i.e., every protein participates in exactly one cluster.

The confusion matrix constructed using the output of each algorithm. In particular, the third, fourth, and fifth columns report, respectively, the number of True Positives (TP), False Positives (FP), and False Negatives (FN) attained by the different algorithms. All these calculations are done considering the MIPS database as the ground truth. The results indicate that our methods settle for a different trade off than Reference. For instance, the numbers of TPs are 838 for pKwikCluster, while Reference discovers twice as many. On the other hand, the FPs of pKwikCluster is six times less than the ones of Reference.

To sum up, our techniques produce different clustering compared to Reference. Yet, in terms of quality the results are comparable even though each clustering achieves

## XIII.PERFORMANCE of pKwikCluster

Both previous experiments indicate that pKwikCluster, which is a provably approximation algorithm, performs very well in practice. The quality of the solutions it produces in terms of both their edit distance and their structural characteristics compare favourably to the other algorithms. Further, pKwikCluster scales well and is thus appropriate for large data sets. Therefore, we focus on pKwikCluster for the remainder of this paper. The goal of this experiment is to illustrate that in practice, the probability of sampling a random world with edit distance significantly larger than the expected edit distance optimized by pKwikCluster is negligible.

## XIV.SCALABILITY EXPERIMENT on POWER-LAW GRAPHS

Since most of the real-world graphs are scale-free, we test the scalability of our algorithm (pKwikCluster) using synthetically generated scale-free graphs.

We generate synthetic graphs using the Baraga´si-Albert (BA) model [11]. The BA graph-generation process adds nodes to the graph one at a time. Each new node is connected to k existing nodes with a probability that is proportional to the number of links that the existing nodes already have. We make these graphs probabilistic by generating a probability value uniformly at random in [0, 1] for each edge. We call these graphs Probabilistic BA graphs.

The execution time of pKwikCluster in seconds as a function of the number of edges of the generated graph. For each data point shown in Fig. 3, we create 20 probabilistic graphs and we run pKwikCluster 20 times on each graph. Thus, each point is the average of 400 executions. The execution time of pKwikCluster in seconds is shown on the y-axis. All graphs have 50,000 nodes and the x-axis is the total number of edges in the graph. We vary the number of edges by choosing parameter k from f1; 2; ...; 10g. It confirms that the running time of pKwikCluster scales linearly to the number of edges.

## XV.CONCLUSION

In this paper, we presented a thorough study of clustering probabilistic graphs using the edit distance metric. We focused on the problem of finding the cluster graph that minimizes the expected edit distance from the input probabilistic graph. Our formulation adheres to the possible-worlds semantics. Also, our objective function does not require the number of clusters as input; the optimal number of clusters is determined algorithmically.

We showed that our problem can be efficiently approximated, by establishing a connection with correlation clustering. In addition, we proposed various intuitive.

## REFERENCES


***[1]*** *C.C. Aggarwal and P.S. Yu, "A Framework for Clustering Uncertain Data Streams," Proc. IEEE 24th Int'l Conf. Data Eng.(ICDE), pp. 150-159, 2008.*

*[2] G. Cormode and A. McGregor, "Approximation Algorithms for Clustering Uncertain Data," Proc. 27th ACM SIGMOD-SIGACTSIGART Symp. Principles of Database Systems (PODS), pp. 191-200, 2008.*

*[3] S. Gu¨nnemann, H. Kremer, and T. Seidl,"Subspace Clustering for Uncertain Data," Proc. SIAM Int'l Conf. Data Mining (SDM),pp. 385-396, 2010.*

*[4] S. Guha and K. Munagala, "Exceeding Expectations and Clustering Uncertain Data," Proc. 28th ACM SIGMOD-SIGACT SIGART Symp. Principles of Database Systems (PODS), pp. 269-278, 2009.*

***[5]*** *N. Bansal, A. Blum, and S. Chawla, "Correlation Clustering,"Machine Learning, vol. 56, nos. 1-3, pp. 89-113, 2004.*

*[6] U. Brandes, M. Gaertler, and D. Wagner, "Engineering Graph Clustering: Models and Experimental Evaluation," ACM J.Experimental Algorithmics, vol. 12, article 1.1, pp. 1-26, 2007.*

*[7] B. Kao, S.D. Lee, F.K.F. Lee, D.W.-L. Cheung and W.S. Ho,"Clustering Uncertain Data Using Voronoi Diagrams and R-Tree Index," IEEE Trans. Knowledge Data Eng., vol. 22, no. 9, pp. 1219-1233, Sept. 2010.*

*[8] H.-P. Kriegel and M. Pfeifle, "Density-Based Clustering of Uncertain Data," Proc. 11th ACM SIGKDD Int'l Conf. Knowledge Discovery in Data Mining (KDD), pp. 672-677, 2005.*

*[9]W.K. Ngai, B. Kao, C.K. Chui, R. Cheng, M. Chau, and K.Y. Yip, "Efficient Clustering of Uncertain Data," Proc. Sixth Int'l Conf. Data Mining (ICDM), pp. 436-445, 2006.*

*[10] T. Bernecker, H.-P. Kriegel, M. Renz, F. Verhein, and A. Zuefle, "Probabilistic Frequent Item set Mining in Uncertain Databases," Proc. 15th ACM SIGKDD Int'l Conf. Knowledge Discovery in DataMining (KDD), 2009.*

*[11] L. Sun, R. Cheng, D.W. Cheung, and J. Cheng, "Mining Uncertain Data with Probabilistic Guarantees," Proc. 16th ACM SIGKDD Int'l Conf. Knowledge Discovery in Data Mining (KDD), pp. 273-282, 2010.*

*[12] Q. Zhang, F. Li, and K. Yi, "Finding Frequent Items in Probabilistic Data," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD), pp. 819-832, 2008*

*[13] H. Mewes et al. "Mips: Analysis and Annotation of Proteins from Whole Genomes," Nucleic Acids Research, vol. 32, pp. D41-D44, 2004*

*[14] N.J. Krogan et al., "Global Landscape of Protein Complexes in the Yeast Saccharomyces Cerevisiae," Nature, vol. 440, no. 7084, pp. 637-643, http://dx.doi.org/10.1038/nature04670, Mar.2006.*

*[15] S.E. Schaeffer, "Graph Clustering," Computer Science Rev., vol. 1,no. 1, pp. 27-64, 2007.*

 