Efficient Ad-Hoc Graph Inference and Matching in Biological Databases

Xiang Lian
Department of Computer Science
Kent State University
1300 Lefton Esplanade
Kent, OH 44242, USA
xlian@kent.edu

Dongchul Kim
Department of Computer Science
The University of Texas Rio Grande Valley
1201 West University Drive
Edinburg, TX 78539, USA
dongchul.kim@utrgv.edu

ABSTRACT

In many real applications such as bioinformatics and biological network analysis, it has always been an important, yet challenging, topic to accurately infer/reconstruct *gene regulatory networks* (GRNs) from microarray data, and efficiently identify those matching GRNs with similar interaction structures for potential disease analysis and treatment tasks. Motivated by this, in this paper, we formalize the problem of *ad-hoc inference and matching over gene regulatory networks* (IM-GRN), which deciphers ad-hoc GRN graph structures online from gene feature databases (without full GRN materializations), and retrieves the inferred GRNs that are subgraph-isomorphic to a query GRN graph with high confidences. Specifically, we propose a novel probabilistic score to measure the possible interaction between any two genes (inferred from gene feature vectors), and thus model GRNs by probabilistic graphs, containing edge existence probabilities. In order to efficiently process IM-GRN queries, we propose effective reduction, pruning, and embedding strategies to significantly reduce the search space of GRN inference and matching, without materializing all GRNs. We also present an effective indexing mechanism and an efficient IM-GRN query processing algorithm by the index traversal. Finally, extensive experiments have been conducted to verify the efficiency and effectiveness of our proposed IM-GRN query answering approaches over real/synthetic GRN data sets.

Keywords

Ad-hoc graph inference and matching; gene regulatory networks; IM-GRN

1. INTRODUCTION

In many real applications such as bioinformatics [25, 14, 7] and medical/health databases [8], it has been increasingly important, yet challenging, to investigate microarray data and better understand gene functions and cellular dynamics. Specifically, since genes in living organisms do not function alone (i.e., they may interact with each other), one important biological problem is to accurately infer *gene regulatory networks* (GRNs) from microarray data (via "reverse engineer" techniques [3, 4, 23, 13]), and use them for

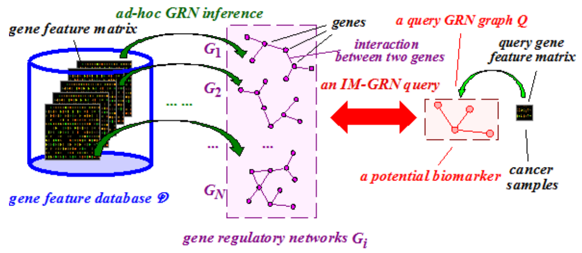


Figure 1: An example of finding a potential biomarker for cancer over GRNs inferred from gene feature databases.

important tasks, such as disease analysis and treatment (e.g., cancer [12]), derivation of new biological hypothesis about molecular interactions [4], guide of novel biological perturbation/intervention experiments [7], design of medicines or drugs [8], and so on.

In this paper, we will tackle an important problem of ad-hoc matching over the inferred GRN graphs, which has the following motivation examples in biological databases.

Example 1 (Identification of Diagnostic Biomarkers [6]) *Recent studies showed that GRNs could be used as biomarkers of diseases (e.g., cancer [12]) for diagnostic and predictive purposes. In particular, hallmarks of cancer are represented by pathways, on which genes actively interact with each other. Therefore, in order to confirm a potential biomarker, as shown in Figure 1, we can first infer a query GRN graph Q (i.e., a possible cancer biomarker) from cancer samples (in a query gene feature matrix), and then retrieve those matching (subgraphs of) GRNs (i.e., similar to Q with high confidences), inferred from the existing gene feature database $\mathcal{D}$ (which is collected from experiments of the biological literature, public databases, medical centers, or research institutions). This way, the retrieved matching GRNs can be used as the supporting evidence of the cancer biomarker, case studies about physiological and disease conditions for the cancer biomarker, and references for treatment methodologies. Similarly, we can also identify diagnostic biomarkers for other diseases as well.* ■

Example 2 (Disease Clustering and Classifications) *GRN graph structures often change dynamically for different phases of the disease or under different physiological/disease conditions. Thus, with microarray data from heterogeneous data sources (e.g., biomedical experiments in the literature or from different institutions), it is important to identify clusters of the inferred GRNs (which might be under distinct phases or conditions) for comparative network analysis [5]. During the graph clustering, one classical problem is the pattern matching, that is, given a query GRN pattern (a representative pattern) Q, obtaining those inferred GRNs G_i (in the same cluster) that subgraph-match with Q with high confidences.*

Moreover, given a newly emerging (or unknown) disease, it is crucial to find some existing (labeled) diseases that are similar to

SIGMOD'17, May 14-19, 2017, Chicago, Illinois, USA
© 2017 ACM. ISBN 978-1-4503-4197-4/17/05...$15.00
DOI: http://dx.doi.org/10.1145/3035918.3035929

the unknown disease, in terms of gene interactions. In this case, we can first infer a query GRN graph Q from gene features of the new disease (obtained from partial biological experiments due to time/budget limitations). Then, we can retrieve those inferred GRNs, G_i, from existing diseases (in the literature or experiments from institutions), which have the same regulatory structures as Q with high confidence level. Intuitively, we can classify the new disease into categories of those retrieved (existing) diseases (via GRNs), which may potentially reveal valuable information for treating the new disease (e.g., by using treatment strategies similar to existing diseases). ■

Inspired by the examples above, in this paper, we will formalize an important and useful problem, namely *ad-hoc inference and matching over gene regulatory networks* (IM-GRN), which aims to efficiently obtain ad-hocly inferred GRNs that match with a given query GRN Q with high confidences.

In the literature of the GRN inference, a classical scoring-based method [4] utilizes the *Pearson's correlation score* between any two gene feature vectors, and infers a gene interaction (i.e., an edge) between two genes in GRN (if this score is above a user-specified threshold). However, in the case of noisy feature samples and/or small sample size, the resulting correlation scores may often deviate from their actual correlation values, which thus lead to inaccurate edge inference, w.r.t. the score threshold, in GRNs.

Therefore, in this paper, we will propose a novel and robust variant of the GRN inference measure (based on the Pearson's correlation). Specifically, this measure can accurately describe the *interaction probability* between any two genes, derived from gene feature vectors via *randomization techniques*. As will be confirmed later in Section 6.2, our proposed inference measure is more robust than the classical one over real gene data, in terms of the *receiver operating characteristic* (ROC). Accordingly, we model the inferred GRN by a probabilistic GRN graph, in which each edge is associated with an existence probability (i.e., our proposed novel measure).

With this GRN graph model, given an ad-hoc inference threshold, our IM-GRN problem is to online reconstruct probabilistic GRN graphs from database $\mathcal{D}$ (by inferring edges w.r.t. the ad-hoc inference threshold), and meanwhile retrieve those inferred GRNs, G_i, from biological database $\mathcal{D}$, such that G_i contain an inferred query GRN Q with high confidences.

Obstacles. To efficiently and effectively process the IM-GRN query, we need to overcome several major obstacles. First, in the IM-GRN problem, users (e.g., biologists) can flexibly specify an ad-hoc inference threshold for online GRN inference. Due to arbitrary possible values of the inference threshold, it is both time- and space- inefficient to offline materialize all possible GRN graphs G_i (against all possible inference thresholds) from database $\mathcal{D}$. Moreover, the offline pre-computations of interaction probabilities for all possible edges in GRNs (i.e., complete graphs) are neither space-efficient for storage nor time-efficient for the search, especially in the case of large-scale biological databases. Thus, we need to design efficient query processing algorithms to tackle the IM-GRN problem without offline GRN materialization, which is rather challenging. To the best of our knowledge, there is no prior work that solves such an IM-GRN problem on ad-hocly inferred graphs.

Second, due to the uncertainty and graph properties in GRNs, it is also challenging for the IM-GRN problem to efficiently obtain matching probabilistic subgraphs that are isomorphic to the query graph Q with high confidences. Note that, the isomorphism checking for graphs is NP-hard. Moreover, in the context of probabilistic graph [19], we usually consider *possible worlds* semantics for probabilistic graphs (i.e., GRN), where each possible world is a materialized instance of the probabilistic graph, G_i, with deterministic edges. However, considering that each edge may or may not exist in the real world, there are an exponential number of possible worlds, that is, $O(2^{|E(G_i)|})$, where $|E(G_i)|$ is the number edges in graph G_i. Thus, it is inefficient, or even infeasible, to tackle the IM-GRN problem over possible worlds of GRNs. Therefore, in this paper, we will propose effective pruning mechanisms to greatly reduce the search space of IM-GRN query answering.

Third, the IM-GRN problem involves a large number of gene feature vectors of different sample sizes (dimensions). While prior works usually considered encoding or indexing on objects with the same dimensionality (e.g., R*-tree [1]), it is challenging to effectively encode and index gene features of different dimensions. As a result, in the sequel, we will propose novel encoding/indexing methods to facilitate efficient IM-GRN query processing over the large-scale gene feature database $\mathcal{D}$.

In this paper, we make the following contributions.

1. We formally define the problem of ad-hoc inference and matching over gene regulatory networks (IM-GRN) in Section 2.
2. We reduce our IM-GRN problem over graph possible worlds with the newly proposed inference measure to the one in the Euclidean space, and design effective pruning strategies to filter out false alarms of IM-GRN answers in Section 3.
3. We propose an effective pivot-based embedding approach (guided by a cost model) to encode gene feature vectors of different dimensions in Section 4.
4. We design an effective indexing mechanism over gene feature databases, and illustrate a novel IM-GRN query processing algorithm in Section 5.
5. We demonstrate through extensive experiments the effectiveness and efficiency of our proposed inference approach and IM-GRN query answering algorithms in Section 6.

In addition, Section 7 reviews previous works on inference of gene regulatory networks and probabilistic graph databases. Finally, Section 8 concludes this paper.

2. PROBLEM DEFINITION

Section 2.1 presents the data model for gene feature databases. Section 2.2 defines the inference of gene regulatory networks (GRNs) from gene feature databases. Section 2.3 formalizes our problem of *ad-hoc inference and matching over gene regulatory networks*, namely IM-GRN.

2.1 Data Model for Gene Feature Databases

A gene feature database, $\mathcal{D}$, contains gene expression features collected from N data sources (e.g., experimental results from the biological literature, public databases, research institutions, or medical centers). For the i-th data source ($1 \leq i \leq N$) in database $\mathcal{D}$, we store its corresponding gene features, represented by an $l_i \times n_i$ matrix, M_i, which are obtained from biological experiments.

DEFINITION 1. **(Gene Feature Databases)** *A gene feature database, $\mathcal{D}$, contains N $l_i \times n_i$ gene feature matrices, denoted as M_i, in which each element $M_i[j][k]$ records the feature value of the k-th gene evaluated from the j-th individual (e.g., patient), where $1 \leq i \leq N$, $1 \leq j \leq l_i$, and $1 \leq k \leq n_i$.*

From Definition 1, gene features from each data source can be represented by an $l_i \times n_i$ matrix, M_i. Specifically, the j-th row of matrix M_i stores n_i gene feature values, which are collected from the j-th patient's sample; the k-th column of matrix M_i is a gene feature vector, which contains l_i feature samples of the k-th gene, collected from l_i patients, respectively.

Note that, in the gene feature database $\mathcal{D}$, N gene feature matrices (for N data sources, respectively), may be of different sizes (i.e., with distinct numbers of patients (rows) and genes (columns)).

2.2 Inference of Gene Regulatory Networks

In this subsection, we give the definition of inferring an interaction graph (i.e., GRN), denoted as G_i, from each gene feature matrix, M_i. In this paper, we consider a variant of a classical scoring-based measure, *absolute Pearson's correlation coefficient* [4], to decide if two genes interact with each other in GRNs.

Specifically, in [4], the scoring-based method first computes the absolute Pearson's correlation coefficient between any two feature vectors, X_s and X_t, of length l_i, from the s-th and t-th genes, respectively. Then, two genes (vertices) have a connecting edge in GRNs, if the resulting coefficient is above a fixed inference threshold. Practically, different thresholds may result in different GRNs. Thus, it is difficult, and lacking of guidelines, for non-expert users to set the inference threshold and obtain an effective/accurate GRN. What is worse, due to small sample sizes or noisy gene feature samples, the previous correlation measure [4] cannot achieve good inference accuracy (which will be later confirmed by experiments in Section 6.2).

Inspired by the problem with the scoring-based method, in this paper, we will design a more robust measure, which is defined as the confidence level (probability) that the (Pearson's) correlation between two gene feature vectors is higher than that of independent (randomized) feature vectors. With this probabilistic measure, users can specify an ad-hoc inference threshold to guarantee the confidences of the inferred edges in GRNs.

Formally, we give the inference of an edge between two genes (vertices) in a GRN from gene feature vectors below.

DEFINITION 2. **(Inference of Edges in GRNs)** *Given two $l_i \times 1$ feature vectors, X_s and X_t, of two genes (corresponding to vertices v_s and v_t, respectively), and an inference threshold $\gamma \in [0, 1)$, vertices v_s and v_t have a connecting edge, $e_{s,t}$, if it holds that:*

$$e_{s,t}.p = Pr\{r(X_s, X_t) > r(X_s, X_t^R)\} > \gamma, \tag{1}$$

where X_t^R is a randomized vector of the original vector X_t, and $r(X_s, X_t)$ is the absolute Pearson's correlation coefficient [4] between vectors X_s and X_t, given as follows:

$$r(X_s, X_t) = \left| \frac{\sum_i (X_s[i] - \overline{X_s}) \cdot (X_t[i] - \overline{X_t})}{\sqrt{\sum_i (X_s[i] - \overline{X_s})^2} \cdot \sqrt{\sum_i (X_t[i] - \overline{X_t})^2}} \right|. \tag{2}$$

Intuitively, Definition 2 computes the probability that two vectors, X_s and X_t, has higher correlation score than that of two randomized (independent) vectors, X_s and X_t^R (as given in Eq. (1)). If this probability, $e_{s,t}.p$, is greater than an ad-hoc confidence level γ, then there exists an edge, $e_{s,t}$, between two vertices, v_s and v_t, corresponding to vectors X_s and X_t, respectively.

DEFINITION 3. **(Gene Regulatory Networks, GRNs)** *Given an $l_i \times n_i$ gene feature matrix M_i, a gene regulatory network (GRN), G_i, is a probabilistic graph in the form of a triple $(V(G_i)$, $E(G_i)$, $\Phi(G_i))$. Here, we have:*

- *$V(G_i)$ is a set of vertices v_s with gene labels $l(v_s)$;*
- *$E(G_i)$ is a set of edges $e_{s,t}$ associated with existence probabilities $e_{s,t}.p \in [0, 1)$;*
- *$\Phi(G_i)$ is a mapping function: $\Phi(G_i) : V(G_i) \times V(G_i) \mapsto E(G_i)$.*

where the existence probability, $e_{s,t}.p$, of edge $e_{s,t}$ is from Eq. (1).

Note that, different users may specify ad-hoc and distinct inference thresholds γ, and the resulting GRNs can be different. One possible method to infer GRNs is to offline enumerate all possible GRNs, under various γ values, which is however not time- and space- efficient. In contrast, in this paper, we will only organize gene feature matrices M_i in the database $\mathcal{D}$, without offline materializing them into all possible GRN structures G_i (w.r.t. different inference thresholds γ).

In the literature of the GRN inference, there are some other measures/approaches to reconstruct GRN graphs, such as *mutual information* [23, 3], *regression* [13, 9], and so on. In this paper, we will only focus on the novel/robust variant of the correlation score [4] for the GRN inference. We would like to leave the topics of using similar idea of the randomized vectors for other interesting inference measures/approaches as our future work.

2.3 Ad-Hoc Inference and Matching Problem in GRNs

The IM-GRN problem. Next, we give the problem definition of *ad-hoc inference and matching over gene regulatory networks* (namely, IM-GRN). Specifically, given a query gene feature matrix M_Q, our IM-GRN problem is to retrieve those gene feature matrix data, M_i, from database $\mathcal{D}$, such that the query graph Q (inferred from M_Q) is matching with (i.e., subgraph isomorphic to) subgraphs, G, of the inferred GRNs, G_i, with high confidences.

Specifically, we denote M_Q as an $l_Q \times n_Q$ query gene feature matrix, which can be obtained from biological experiments, where each element $M_Q[j][k]$ (for $1 \leq j \leq l_Q$ and $1 \leq k \leq n_Q$) stores the feature value of the k-th gene collected from the sample of the j-th patient. Similar to the inference of GRN graphs mentioned in Section 2.2, we can also infer from the query matrix M_Q, and obtain a query GRN graph, Q (representing a graph biomarker, a representative graph pattern in a cluster, or a GRN of a new disease), which contains n_Q vertices (i.e., genes), q_s, and edges, $qe_{s,t}$ that connect vertices q_s and q_t (indicating that gene q_s interacts with q_t), where each edge $qe_{s,t}$ is associated with an existence probability, $qe_{s,t}.p$.

DEFINITION 4. **(Ad-Hoc Inference and Matching Over Gene Regulatory Networks, IM-GRN)** *Given an $l_Q \times n_Q$ query gene feature matrix, M_Q, a gene feature database $\mathcal{D}$ with N gene feature matrices M_i ($1 \leq i \leq N$), an inference threshold $\gamma \in [0, 1)$, and a probabilistic threshold $\alpha \in [0, 1)$, an ad-hoc inference and matching query over gene regulatory networks (IM-GRN) retrieves matrices $M_i \in \mathcal{D}$ such that the inferred GRNs G_i and query GRN Q (via the inference in Definition 2) satisfy the conditions that:*

- *the query graph Q is isomorphic to a subgraph, G, of the inferred GRN G_i (denoted as $Q \equiv G$); and*
- *the appearance probability of subgraph G is greater than α, that is, $Pr\{G\} > \alpha$.*

Here, it holds that:

$$\begin{aligned} Pr\{G\} &= \prod_{\forall qe_{s,t} \in E(Q)} Pr\{\text{edge } e_{s,t} \text{ exists in } E(G)\} \\ &= \prod_{\forall qe_{s,t} \in E(Q)} e_{s,t}.p \end{aligned} \tag{3}$$

where $e_{s,t}.p$ is given by Eq. (1).

Intuitively, in Definition 4, the IM-GRN problem obtains those GRN graphs G_i (inferred from gene feature matrices M_i) that contain some subgraphs G, such that (1) subgraphs G are isomorphic to the query graph Q (inferred from the given query matrix M_Q), and (2) subgraphs G ($\subseteq G_i$) have high existence confidence (i.e., $Pr\{G\}>\alpha$). In particular, as given in Eq. (3), the appearance probability, $Pr\{G\}$, of G is given by the multiplication of (non-) existence probabilities of edges $e_{s,t}$ in G, if their corresponding edges $qe_{s,t}$ in query graph Q exist.

Challenges. To efficiently and effectively solve the IM-GRN problem, there are several major challenges below. First, as given in

Symbols	Descriptions
$\mathcal{D}$	a gene feature database from N data sources
G_i	a gene regulatory network (GRN) with n_i genes (vertices)
M_i	an $l_i \times n_i$ matrix, corresponding to the data graph G_i
Q	a query gene regulatory network with n_Q genes (vertices)
M_Q	an $l_Q \times n_Q$ matrix, corresponding to the query graph Q
l_i	the number of patient samples in gene feature matrix M_i
n_i	the number of genes in gene feature matrix M_i
$V(G_i)$	a set of vertices, v_j ($1 \leq j \leq n_i$), in the GRN graph G_i
$E(G_i)$	a set of edges in the GRN graph G_i
v_s (or v_t)	a vertex in $V(G_i)$
q_s (or q_t)	a vertex in $V(Q)$
$e_{s,t}$ (or $qe_{s,t}$)	an edge connecting vertices v_s and v_t (or vertices q_s and q_t)
$e_{s,t}.p$ (or $qe_{s,t}.p$)	the existence probability of edge $e_{s,t}$ (or $qe_{s,t}$)
$r(X_s, X_t)$	the absolute Pearson's correlation coefficient between vectors X_s and X_t
$dist(X_s, X_t)$	the Euclidean distance between vectors X_s and X_t

Table 1: Notations and their descriptions.

Definition 2, to obtain the existence probability of each edge $e_{s,t}$ (as shown in Eq. (1)), in turn, we need to compute the absolute Pearson's correlation coefficient, $r(X_s, X_t^R)$, between two randomized vectors, which requires applying the Monte Carlo sampling method, and thus incurs high computation cost. Thus, it is challenging to efficiently calculate the probability $e_{s,t}.p$. In this paper, we will propose a novel approach to reduce our IM-GRN problem over an exponential number of possible worlds over probabilistic graphs (with edge existence probabilities in Eq. (1)) to the one in the Euclidean space.

Second, given an ad-hoc inference threshold γ, it is not time- and space- efficient to offline materialize all GRNs G_i from N matrices $M_i \in \mathcal{D}$. Typically, given N $l_i \times n_i$ matrices, the space complexity of offline pre-computing existence probabilities of all pairwise vertices (genes) in (complete) GRN graphs is given by $O(\frac{n_i \cdot (n_i - 1)}{2} \cdot N)$. Assuming that $n_i = 300$ and $N = 100K$, the required storage is 17.94 Gigabytes with single-precision floating numbers (or 35.88 Gigabytes with double-precision floating numbers), which is not space-efficient. Moreover, to retrieve matching GRN subgraphs, we also need to sequentially scan all these pre-computed data with $O(\frac{n_Q \cdot (n_Q - 1)}{2} \cdot N)$ time cost and high I/O cost of reading 17.94 GB (or 35.88 GB) data from the disk, which is not time- and I/O-efficient, where n_Q is the number of genes in the query GRN graph Q. Thus, we need to design effective pruning methods to reduce the search space of online IM-GRN query processing, without materializing all GRNs.

Third, each matrix M_i is collected from different numbers of patients and different sets of genes, which are also distinct from that of query matrix M_Q. As a result, it is non-trivial how to organize these matrices of different sizes, and efficiently conduct the subgraph matching over a large number of online inferred GRNs. Inspired by this, in this paper, we propose an embedding approach to map matrices into a unified multidimensional data space, design an effective index mechanism, and present an efficient IM-GRN query processing approach by traversing the index.

Discussions on a class of problems generalized from IM-GRN and their applications/solutions. Please refer to a general class of problems over ad-hocly inferred graphs (similar to IM-GRN) in real applications like social network analysis and video copyright-violation detection in Appendix A. We would like to leave topics of ad-hoc graph problems in these domains as our future work.

Table 1 depicts the commonly used symbols and their descriptions in this paper.

3. IM-GRN PROBLEM REDUCTION

3.1 The Edge Existence Probability Reduction

As mentioned in Section 2.3, it is time-consuming to compute existence probabilities of edges (as given in Eq. (1)), since we need to consider randomized vectors, X_t^R (e.g., via the Monte Carlo sampling method [17, 15]). Due to the complexity of computing absolute Pearson's correlation coefficient and, in turn, the edge existence probability, in this subsection, we will reduce the computation of edge probability $e_{s,t}.p$ via Pearson's correlations to the one in a Euclidean space.

LEMMA 1. **(Reduction of the Edge Existence Probability)** *Assume that two feature vectors X_s and X_t are two standardized vectors of length $l_i > 1$ (corresponding to vertices v_s and v_t in GRN G_i, respectively). The existence probability, $e_{s,t}.p$, of edge $e_{s,t}$ between two vertices v_s and v_t in Eq. (1) can be reduced to:*

$$e_{s,t}.p = Pr\{dist(X_s, X_t^R) \geq dist(X_s, X_t)\}, \tag{4}$$

where $dist(X, Y)$ computes the Euclidean distance between vectors X and Y, that is, $dist(X, Y) = \sqrt{\sum_{k=1}^{l_i}(X[k] - Y[k])^2}$.

PROOF. Please refer to the detailed proof in Appendix B. □

Lemma 1 indicates that the edge existence probability $e_{s,t}.p$ can be reduced to the probability that involves the Euclidean distance (instead of absolute Pearson's correlation coefficient), which is given by Eq. (4).

Monte Carlo sampling. In order to estimate the edge existence probability $e_{s,t}.p$ (given in Eq. (4)), we can apply the Monte Carlo sampling method [17] as follows.

Denote $dist(X_s, X_t^R)$ as a random variable Z. Then, Eq. (4) can be written as: $e_{s,t}.p = Pr\{Z > dist(X_s, X_t)\}$. Here, random variable Z has the population of size $O(l_i!)$, which corresponds to distances, $dist(X_s, X_t^R)$, with respect to $l_i!$ possible randomized vectors X_t^R. The Monte Carlo sampling method obtains S independent (uniformly distributed) samples from the population, and estimate the edge existence probability $e_{s,t}.p$ as $\widehat{\rho}$, which is given by the ratio of times that $Z > dist(X_s, X_t)$.

Let ρ be the actual value of $e_{s,t}.p$. We have the following lemma [15] about the required number, S, of samples (i.e., randomized vectors X_t^R), used for estimating $e_{s,t}.p$ with the confidence guarantee.

LEMMA 2. **(Determining the Sample Size [15])** *Given a sample size $S \geq \frac{3}{\epsilon^2} ln \frac{2}{\delta}$, the estimated edge existence probability $\widehat{\rho}$ is an ϵ-approximation of actual edge existence probability ρ ($= e_{s,t}.p$), such that:*

$$Pr\{(1 - \epsilon)\rho \leq \widehat{\rho} \leq (1 + \epsilon)\rho\} \geq 1 - \delta. \tag{5}$$

3.2 Pruning Strategies

So far, we have reduced the GRN graph inference problem that involves absolute Pearson's coefficient to the one in the Euclidean distance. As mentioned in Section 2, given a user-specified ad-hoc inference threshold γ, it is still very costly to compute exact edge existence probabilities $e_{s,t}.p$ (by using the Monte Carlo sampling method), and reconstruct edges $e_{s,t}$ (satisfying $e_{s,t}.p > \gamma$) for all the N GRNs G_i. Thus, in this subsection, we will propose effective pruning methods, namely *edge inference pruning* and *graph existence pruning*, to filter out those false alarms of GRN graphs, G_i, that do not contain the query graph Q.

3.2.1 Edge Inference Pruning

Our basic idea of the *edge inference pruning* method is to derive an upper bound, $ub\_P(e_{s,t})$, of the edge existence probability $e_{s,t}.p$, which can be computed at low cost (compared with the costly computation of exact probability). Then, if this upper bound $ub\_P(e_{s,t})$ is smaller than or equal to inference threshold γ, we can safely say that there is no edge $e_{s,t}$ between vertices v_s and v_t.

LEMMA 3. **(Edge Inference Pruning in GRNs)** *Given an inference threshold γ, and a potential edge $e_{s,t}$ between two vertices v_s and v_t, if it holds that $ub\_P(e_{s,t}) \leq \gamma$, then edge $e_{s,t}$ does not exist in G_i.*

PROOF. Please refer to the detailed proof in Appendix C. □

The computation of probability upper bound $ub\_P(e_{s,t})$. Next, we discuss how to obtain an upper bound, $ub\_P(e_{s,t})$, of the edge existence probability $e_{s,t}.p$. Recall from Section 3.1 that $e_{s,t}.p = Pr\{Z > dist(X_s, X_t)\}$, where Z is a random variable corresponding to distance $dist(X_s, X_t^R)$ (w.r.t. randomized vectors X_t^R). Then, with the *Markov's inequality*[1], we can derive the probability upper bound $ub\_P(e_{s,t})$ in the lemma below.

LEMMA 4. **(Derivation of Probability Upper Bound $ub\_P(e_{s,t})$)** *An upper bound, $ub\_P(e_{s,t})$, of the edge existence probability $e_{s,t}.p$ can be given by:*

$$ub\_P(e_{s,t}) = \frac{E(Z)}{dist(X_s, X_t)}. \quad (6)$$

where $E(Z)$ is the expectation of variable Z.

PROOF. Please refer to the detailed proof in Appendix D. □

3.2.2 Graph Existence Pruning

As mentioned in Definition 4, the IM-GRN problem aims to retrieve subgraphs, G, of the inferred GRNs, G_i, that are matching with the query graph Q, and with appearance probabilities $Pr\{G\}$ greater than α. Thus, we propose a graph existence pruning method, which obtains upper bounds, $UB\_Pr\{G\}$, of appearance probabilities $Pr\{G\}$, and filters out those false alarms of GRN subgraphs G satisfying $UB\_Pr\{G\} < \alpha$.

We summarize the graph existence pruning in the lemma below.

LEMMA 5. **(Graph Existence Pruning)** *Let $UB\_Pr\{G\}$ be an upper bound of appearance probability $Pr\{G\}$, where G is a candidate subgraph of an inferred GRN G_i that may match with Q. If it holds that $UB\_Pr\{G\} \leq \alpha$, then subgraph G cannot be the IM-GRN answer.*

PROOF. Please refer to the detailed proof in Appendix E. □

Intuitively, Lemma 5 computes upper bounds, $UB\_Pr\{G\}$, of probabilities $Pr\{G\}$ (for $G \subseteq G_i$ and $G \equiv Q$), and we can safely prune GRN subgraphs G such that their probability upper bounds are low (i.e., $\leq \alpha$).

Based on Eq. (3), we can compute an upper bound $UB\_Pr\{G\}$ of probability $Pr\{G\}$ by overestimating edge existence probabilities $e_{s,t}.p$ by $ub\_P(e_{s,t})$ (as given in Eq. (6)). That is, we have $UB\_Pr\{G\} = \prod_{\forall qe_{s,t} \in E(Q)} ub\_P(e_{s,t})$.

4. PIVOT-BASED MATRIX EMBEDDING

4.1 Motivation

Although Section 3.2 provides effective pruning strategies for inferring GRN graph edges and filtering out GRN false alarms with low appearance probabilities, they are only limited to the pruning over a pair of GRN graphs (G, Q). In other words, we still need to sequentially scan all the (inferred) GRNs from the database, and compare the query graph Q with each of GRNs G_i by using our proposed pruning methods. The time complexity of this *linear scan* method is given by $O(N \cdot cmp\_cost)$, where N is the number of gene feature matrices M_i (corresponding to GRNs G_i) in the database $\mathcal{D}$, and cmp_cost is the average time cost of checking whether GRN G_i is the IM-GRN query answer (including subgraph isomorphism checking and the appearance probability calculation). Clearly, this is not time-efficient and not scalable for a large-scale database $\mathcal{D}$ (i.e., for a large N value).

Inspired by the inefficiency of the linear scan method above, in the sequel, we will design a novel and effective matrix embedding mechanism, which can be used for preparing the index construction. With the index over the embedded data, we can avoid accessing a large portion of the inferred GRNs G_i via the index pruning, and only need to perform the refinement over a small candidate set of GRNs (by applying pruning methods discussed in Section 3.2).

Outline. In this section, we will organize gene feature matrices M_i in database $\mathcal{D}$ by designing a *pivot-based matrix embedding* approach which maps gene feature vectors into a $2d$-dimensional Euclidean space via pivots. Then, in Section 5, we will illustrate how to store the embedded data in an index, and use a novel *pivot pruning* method to facilitate the IM-GRN query answering.

4.2 Pivot-Based Matrix Embedding

In gene feature database $\mathcal{D}$, different matrices M_i may contain gene feature vectors (e.g., X_s or X_t) of different dimensions l_i. While prior works on indexing usually built multidimensional indices (e.g., R*-tree [1]) on data of the same dimension, they cannot directly deal with indexing over data of distinct dimensions. Thus, it is non-trivial and challenging to construct an index over matrices of different sizes in $\mathcal{D}$ (i.e., feature vectors of distinct arities).

Pivot-based embedding. In order to index matrx data of diverse dimensions, below, we will design an index over gene feature matrices via *pivots*. Specifically, for each $l_i \times n_i$ matrix $M_i \in \mathcal{D}$, we will select d pivots (vectors) piv_1, piv_2, ..., and piv_d, with the same dimension l_i. Then, for each feature vector (column) X_s ($1 \leq s \leq n_i$) of size l_i in matrix M_i, we can offline pre-compute d Euclidean distances between X_s and d pivots, piv_w ($1 \leq w \leq n_i$). Similarly, for X_s, we can also offline calculate d expected distances between X_s^R and d pivots, where X_s^R is a randomized vector.

As a result, we can embed each gene feature vector X_s from matrix M_i into a $2d$-dimensional vector, $g_{i,s}$, as follows:

$$\begin{aligned} g_{i,s} = (&dist(X_s, piv_1), E(dist(X_s^R, piv_1)),\\ &dist(X_s, piv_2), E(dist(X_s^R, piv_2)),\\ &...,\\ &dist(X_s, piv_d), E(dist(X_s^R, piv_d))). \end{aligned}$$

For simplicity, we denote $g_{i,s}$ as follows:

$$\begin{aligned} g_{i,s} = (&x_s[1], y_s[1];\\ &x_s[2], y_s[2];\\ &...;\\ &x_s[d], y_s[d]). \end{aligned}$$

Derivation of pruning conditions via pivots. Next, we will utilize the embedded $2d$-dimensional vectors (points), and derive a *pivot pruning* method. The basic idea is as follows. Similar to the edge inference pruning (given in Lemma 3), in the embedded $2d$-dimensional space, our goal is to obtain an upper bound, $ub\_P(e_{s,t}, piv_w)$, of the edge existence probability, $e_{s,t}.p$, w.r.t. pivots piv_w ($1 \leq w \leq d$).

Specifically, since the triangle inequality holds in the Euclidean space, we have:

$$|dist(a,b) - dist(b,c)| \leq dist(a,c) \leq dist(a,b) + dist(b,c),$$

for vectors (points) a, b, and c, where $dist(\cdot,\cdot)$ is a Euclidean distance function. Then, we can derive a probability upper bound from Eq. (4) by relaxing the distance bounds:

[1] *http://en.wikipedia.org/wiki/Markov's_inequality.*

$$
\begin{aligned}
e_{s,t}.p &= Pr\{dist(X_s, X_t^R) > dist(X_s, X_t)\} \quad (7)\\
&\leq Pr\{dist(X_s, piv_w) + dist(X_t^R, piv_w) > dist(X_s, X_t)\\
&\quad \underline{\textit{// since } dist(X_s, piv_w) + dist(X_t^R, piv_w) \geq dist(X_s, X_t^R)}\\
&\leq Pr\{dist(X_s, piv_w) + dist(X_t^R, piv_w)\\
&\qquad > \max_{r=1}^{d} |dist(X_s, piv_r) - dist(X_t, piv_r)|\}\\
&\quad \underline{\textit{// since } dist(X_s, X_t) \geq |dist(X_s, piv_r) - dist(X_t, piv_r)|}\\
&= Pr\{dist(X_t^R, piv_w) > C\},
\end{aligned}
$$

where constant $C = \max_{r=1}^{d} |dist(X_s, piv_r) - dist(X_t, piv_r)| - dist(X_s, piv_w)$.

From Eq. (7), we have two cases:

- **(Case 1)** If $C \leq 0$ holds, we have: $ub\_P(e_{s,t}, piv_w) = 1$;
- **(Case 2)** If $C > 0$ (i.e., $\max_{r=1}^{d} |dist(X_s, piv_r) - dist(X_t, piv_r)| > dist(X_s, piv_w)$) holds, we have:

$$
\begin{aligned}
e_{s,t}.p \leq\ & Pr\{dist(X_t^R, piv_w) > \max_{r=1}^{d} |dist(X_s, piv_r)\\
& -dist(X_t, piv_r)| - dist(X_s, piv_w)\}.
\end{aligned}
$$

Let W be a random variable of $dist(X_t^R, piv_w)$. By applying the Markov's inequality, we can further rewrite the inequality above as:

$e_{s,t}.p \leq \frac{E(W)}{\max_{r=1}^{d} |dist(X_s, piv_r) - dist(X_t, piv_r)| - dist(X_s, piv_w)}$.

Thus, we obtain a probability upper bound $ub\_P(e_{s,t}, piv_w)$

$= \frac{E(W)}{\max_{r=1}^{d} |dist(X_s, piv_r) - dist(X_t, piv_r)| - dist(X_s, piv_w)}$

$= \frac{E(W)}{\max_{r=1}^{d} (dist(X_t, piv_r) - dist(X_s, piv_r)) - dist(X_s, piv_w)}$.

By considering d pivots, we can obtain a tighter upper bound:

$$ub\_P(e_{s,t}) = \min_{w=1}^{d} ub\_P(e_{s,t}, piv_w).$$

Pivot-based pruning. Given a probability upper bound, $ub\_P(e_{s,t}, piv_w)$, w.r.t. pivot piv_w, the basic idea of our pivot-based pruning is as follows: if the pivot-based upper bound holds that $ub\_P(e_{s,t}, piv_w) \leq \gamma$, then we can safely prune the edge $e_{s,t}$.

Note that, for Case 1, since $\gamma \in [0, 1)$, we cannot prune edge $e_{s,t}$ (i.e., as $ub\_P(e_{s,t}, piv_w) = 1$ is always greater than γ). Moreover, in Case 2, we can obtain the pivot pruning condition that:

$$\frac{E(dist(X_t^R, piv_w))}{\max_{r=1}^{d}\{dist(X_t, piv_r) - dist(X_s, piv_r)\} - dist(X_s, piv_w)} \leq \gamma, \quad (8)$$

where $\max_{r=1}^{d} |dist(X_s, piv_r) - dist(X_t, piv_r)| > dist(X_s, piv_w)$.

Eq. (8) is equivalent to:

$$
\begin{aligned}
E(dist(X_t^R, piv_w)) \leq \max_{r=1}^{d}\{&\gamma \cdot dist(X_t, piv_r) \quad (9)\\
&-\gamma \cdot dist(X_s, piv_r) - \gamma \cdot dist(X_s, piv_w)\}.
\end{aligned}
$$

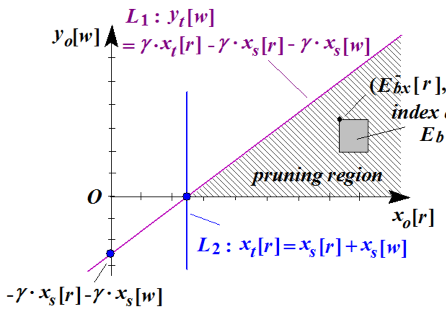
Denote $y_t[w] = E(dist(X_t^R, piv_w))$, $x_t[r] = dist(X_t, piv_r)$, $x_s[r] = dist(X_s, piv_r)$ and $x_s[w] = dist(X_s, piv_w)$. The pruning condition in Eq. (9) can be transformed to:

$$y_t[w] \leq \gamma \cdot x_t[r] - (\gamma \cdot x_s[r] + \gamma \cdot x_s[w]),$$

for some $1 \leq r \leq d$. Moreover, the condition of Case 2 (i.e., $C > 0$) can be transformed to $x_t[r] \geq x_s[r] + x_s[w]$.

Pivot-based pruning condition: Consider two embedded points, $g_{i,s} = (x_s[1], y_s[1]; ...; x_s[d], y_s[d])$ and $g_{i,t} = (x_t[1], y_t[1]; ...; x_t[d], y_t[d])$, obtained from genes, X_s and X_t, respectively. Figure 2 visualizes the condition of pruning the edge between genes X_s and X_t, in a 2-dimensional $y_t[w]$-and-$x_t[r]$ space (for gene X_t).

Specifically, given the value of $x_s[r]$ ($= dist(X_s, piv_r)$), we define a *pivot-based pruning region* (PPR), namely $PPR(X_s, piv_w)$,



$L_1 : y_t[w]$
$= \gamma \cdot x_t[r] - \gamma \cdot x_s[r] - \gamma \cdot x_s[w]$
$y_o[w]$
$(E^-_{bx}[r], E^+_{by}[w])$
index entry
E_b
pruning region
O
$x_o[r]$
$L_2 : x_t[r] = x_s[r] + x_s[w]$
$-\gamma \cdot x_s[r] - \gamma \cdot x_s[w]$

Figure 2: Illustration of the Pivot-Based Pruning Region (PPR) and the Index Pruning.

in the $y_t[w]$-and-$x_t[r]$ data space, which is denoted by the shaded area (with the sloped lines). According to the pruning conditions discussed above, PPR is given by the intersection of the halfplane below line $L_1 : y_t[w] = \gamma \cdot x_t[r] - (\gamma \cdot x_s[r] + \gamma \cdot x_s[w])$, that to the right of line $L_2 : x_t[r] = x_s[r] + x_s[w]$, and the area above the horizontal axis (i.e., $x_o[r]$-axis).

As a result, we can convert the conditions for the pivot-based pruning to the one related to PPR. Intuitively, given a transformed point $g_{i,s}$ (via pivots), as long as there exists an r-th dimension such that point $(x_t[r], y_t[w])$ falls into PPR, $PPR(X_s, piv_w)$, then we say that genes (vertices) X_s and X_t do not have a connecting edge, $e_{s,t}$, between them in the GRN G_i (i.e., edge $e_{s,t}$ can be pruned).

4.3 Cost-Model-Based Pivot Selection

Up to now, we have discussed how to utilize pivots to enable the pruning by deriving the pruning condition. Since different pivots may lead to different pruning power, one remaining, yet important, issue is how to select "good" pivots in order to maximize the pruning power. Thus, in this subsection, we will design a cost model to formalize the pruning power of a pivot selection strategy, and then propose a cost-model-based pivot selection approach to choose pivots with high pruning power.

Specifically, we illustrate the rationale behind our cost model by using the example shown in Figure 2. Intuitively, if the pivot-based pruning region (shaded with the sloped lines), $PPR(X_s, piv_w)$, in Figure 2 is large, then this PPR is expected to achieve high pruning power (i.e., ruling out all points in PPR). Thus, our goal of finding good pivots, piv_r ($1 \leq r \leq d$), is to maximize the area of PPR.

Alternatively, from Figure 2, the area of PPR is decided by the position of line $L_1 : y_t[w] = \gamma \cdot x_t[r] - (\gamma \cdot x_s[r] + \gamma \cdot x_s[w])$ (note: γ is a constant, given by online IM-GRN queries). In other words, the maximization of the PPR area is equivalent to minimizing the term: $(x_s[r] + x_s[w])$ ($= dist(X_s, piv_r) + dist(X_s, piv_w)$).

Therefore, given d pivots $piv_1 \sim piv_d$, we can model the "goodness" of the selected pivots in matrix M_i by the cost formula below:

$$T_i = \sum_{\forall X_s \in M_i} (\min_{r=1}^{d}\{\min_{w=1}^{d}\{dist(X_s, piv_r) + dist(X_s, piv_w)\}\}).$$

Intuitively, small T_i value implies large (expected) pivot-based pruning region, and in turn leads to high pruning power. Thus, during our pivot selection algorithm, we aim to choose d out of n_i feature vectors (e.g., X_s) from M_i as pivots ($d \leq n_i$), such that the cost T_i is minimized.

Cost-model-based pivot selection algorithm. Next, we present the algorithm for the pivot selection, based on our proposed cost model. In particular, given an $l_i \times n_i$ matrix M_i, we have n_i feature vectors X_s of size l_i. Initially, we randomly select d out of n_i feature vectors from M_i as pivots piv_1, piv_2, ..., and piv_d. Then, each time we randomly swap one pivot piv_r with a non-pivot, X_s, in order to minimize the cost function T_i. In order to avoid the local optimum solution, we run such a pivot selection process multiple times by selecting different initial pivots. The detailed pseudo code of this algorithm are depicted in procedure Pivot_Selection of Figure 3.

```
Procedure Pivot_Selection {
  Input: an $l_i \times n_i$ gene feature matrix $M_i$ (containing $n_i$ feature vectors)
         and an integer $d$
  Output: $d$ pivots $piv_1, piv_2, ...,$ and $piv_d$ in $S_{PIV}$ with the minimum cost $T_i$
  (1)  $global\_cost = +\infty$; $S_{PIV} = \emptyset$
  (2)  for $a = 1$ to $global\_iter$
  (3)      randomly select $d$ pivots from $n_i$ feature vectors and form a pivot set $PIV$
  (4)      evaluate the cost function $T_i$ w.r.t. $PIV$ and let $local\_cost = T_i$
  (5)      for $b = 1$ to $swap\_iter$
  (6)          select a random pivot $piv_r \in PIV$
  (7)          randomly choose a non-pivot $X_t \in (M_i - PIV)$
  (8)          $PIV^{new} = PIV - \{piv_r\} + \{X_t\}$
  (9)          evaluate the cost function $T_i^{new}$ w.r.t. $PIV^{new}$
  (10)         if $T_i^{new} < local\_cost$
  (11)             $local\_cost = T_i^{new}$
  (12)             $PIV = PIV^{new}$
  (13)     if $local\_cost < global\_cost$
  (14)         $S_{PIV} = PIV$
  (15)         $global\_cost = local\_cost$
  (16) return $S_{PIV}$
}
```

Figure 3: Algorithm for Pivot Selection.

The Complexity Analysis. Algorithm Pivot_Selection in Figure 3 contains a nested loop, that is, an outer loop (i.e., $global\_iter$ iterations in lines 2-15) and an inner loop (i.e., $swap\_iter$ iterations in lines 5-12). Specifically, in lines 4 and 9, the complexity of evaluating the cost function T_i (or T_i^{new}) w.r.t. a pivot set PIV (or PIV^{new}) is given by $O(l_i \cdot d^2)$. Moreover, the complexity of line 3 is $O(d)$, and the remaining lines have $O(1)$ time costs. Therefore, the total time complexity of the algorithm is given by $O(global\_iter \cdot (d + l_i \cdot d^2 + swap\_iter \cdot l_i \cdot d^2))$.

Discussions on the pivot selection. Figure 3 aims to minimize the cost T_i, or equivalently maximize the pruning power, of our IM-GRN query processing algorithm. From the biological perspective, these selected pivots can represent a group (cluster) of genes in the GRN graph, which might be useful for clustering genes in GRNs. However, this is out of the scope of this paper. We would like to leave this interesting topic on identifying the biological perspective of pivots as our future works.

5. IM-GRN QUERY PROCESSING

5.1 The Index Construction

Indices on gene feature matrices. In this subsection, we will illustrate how to build indices for gene feature matrices, M_i, in the database $\mathcal{D}$.

Multidimensional index. Specifically, for each vector X_s from matrix M_i, we can convert it into a $(2d+1)$-dimensional point $\widehat{g_{i,s}} = (x_s[1], y_s[1]; x_s[2], y_s[2]; ...; x_s[d], y_s[d]; g_s)$, which consists of $2d$-dimensional embedded vector (as mentioned in Section 4) and 1D gene name/ID, where $x_s[r] = dist(X_s, piv_r)$, $y_s[r] = E(dist(X_s^R, piv_r))$, and g_s is the gene name/ID of the s-th gene in matrix M_i (represented by an integer). The intuition that we include g_s as one dimension is that, we want to group those genes with the same gene names/IDs (from distinct matrices or data sources) together in the index, in order to reduce the search cost.

For each embedded point $g_{i,s}$, we maintain a bit vector, $V_f(g_s)$, of size B, into which the gene feature ID, g_s, is hashed. That is, the $H_f(g_s)$-th position in the bit vector $V_f(g_s)$ is set to 1 (while others are set to 0), where $H_f(\cdot)$ is a hashing function. Moreover, we keep another bit vector $V_d(i)$, which encodes the data source ID, i, with respect to matrix M_i, using another hash function $H_d(\cdot)$.

To build an index over the embedded vectors, we insert each $(2d+1)$-dimensional point $g_{i,s}$ into a multidimensional index. In this paper, we use the R*-tree index [1], denoted as $\mathcal{I}$. For intermediate nodes E in the index $\mathcal{I}$, each entry E_k contains a *minimum bounding rectangle* (MBR) that minimally bounds data points under E_k. In particular, E_k is in the form:

$$\begin{aligned}E_k = (&E_{kx}^-[1], E_{kx}^+[1]; E_{ky}^-[1], E_{ky}^+[1];\\ &E_{kx}^-[2], E_{kx}^+[2]; E_{ky}^-[2], E_{ky}^+[2];\\ &...;\\ &E_{kx}^-[d], E_{kx}^+[d]; E_{ky}^-[d], E_{ky}^+[d];\\ &E_k^-[2d+1], E_k^+[2d+1]),\end{aligned}$$

where $E_{kx}^-[r] = \min_{\forall g_{i,s} \in E_k} x_s[r]$, $E_{kx}^+[r] = \max_{\forall g_{i,s} \in E_k} x_s[r]$, $E_{ky}^-[r] = \min_{\forall g_{i,s} \in E_k} y_s[r]$, $E_{ky}^+[r] = \max_{\forall g_{i,s} \in E_k} y_s[r]$, $E_k^-[2d+1] = \min_{\forall g_{i,s} \in E_k} g_s$, and $E_k^+[2d+1] = \max_{\forall g_{i,s} \in E_k} g_s$.

Furthermore, each entry E_k is also associated with a gene feature ID bit vector, $V_f(E_k)$, of size B, where $V_f(E_k)$ is a bit-OR of all bit vectors, $V_f(g_s)$, for gene IDs g_s under entry E_k, that is, $V_f(E_k)[r] = \bigvee_{\forall g_{i,s} \in E_k} V_f(g_s)[r]$ $(1 \le r \le B)$. Similarly, we can also obtain data source ID bit vector $V_d(E_k)$, where $V_d(E_k)[r] = \bigvee_{\forall g_{i,s} \in E_k} V_d(i)[r]$ $(1 \le r \le B)$.

Inverted bit-vector file. In order to facilitate the IM-GRN query processing, we also maintain an inverted bit-vector file, IF, which can help determine possible data source IDs (e.g., i) for each given gene name g_s. Specifically, in the inverted bit-vector file IF, each entry corresponds to a gene name g_s. For all matrices M_i that contain gene name g_s, we hash their corresponding data source IDs, i, into a bit vector, $IF[g_s]$, of size B. In other words, we have: $IF[g_s] = \bigvee_{\exists g_{i,s} \in M_i} V_d(g_s)$.

The index pruning. With the multidimensional tree index $\mathcal{I}$, we can apply the pivot-based pruning (as mentioned in Section 4.2) to filter out false alarms on the level of tree nodes (instead of genes).

Assume that E_a and E_b are two intermediate tree nodes, that may potentially contain genes X_s and X_t (connecting with an edge). We have the following lemma to prune, E_k, of the tree index $\mathcal{I}$.

LEMMA 6. **(Index Pruning)** *Given two node entries E_a and E_b from index $\mathcal{I}$, entry E_b can be safely pruned, if there exists one dimension w, such that:*

$$E_{by}^+[w] \le \max_{r=1}^{d}\{\gamma \cdot E_{bx}^-[r] - \gamma \cdot E_{ax}^+[r] - \gamma \cdot E_{ax}^+[w]\}, \qquad (10)$$

where $E_{by}^+[w] = \max_{\forall X_t \in E_b} E(dist(X_t^R, piv_w))$, $E_{bx}^-[r] = \min_{\forall X_t \in E_b} dist(X_t, piv_r)$, $E_{ax}^+[r] = \max_{\forall X_s \in E_a} dist(X_s, piv_r)$, and $E_{ax}^+[w] = \max_{\forall X_s \in E_a} dist(X_s, piv_w)$.

PROOF. Please refer to the detailed proof in Appendix F. □

Intuitively, as illustrated in Figure 2, with respect to any embedded point x_s in E_a, if node entry E_b is completely in the pruning region (i.e., PPR), then node E_b can be safely pruned. Note that, the condition that E_b is in PPR is equivalent to that top-left point, $(E_{bx}^-[r], E_{by}^+[w])$, is below line L_1 $(\forall x_s \in E_a)$, which is exactly given by Inequality (10).

5.2 The IM-GRN Algorithm

In this subsection, we illustrate the IM-GRN query processing algorithm, namely IM-GRN_Processing, in Figure 4, which retrieves the matching (inferred) GRNs that are similar to an inferred query graph Q with high confidences, by using multidimensional index $\mathcal{I}$ and inverted file IF.

Specifically, given the query gene feature matrix M_Q, we can first perform the edge inference pruning method (in Lemma 3), and then obtain an exact query graph Q in which edges, $qe_{s,t}$, have existence probabilities, $qe_{s,t}.p$, greater than edge inference threshold γ (line 1). Next, we will start from one gene g_s with the highest degree in Q, and identify its neighbor set $NS(g_s)$ in Q (line 2). Intuitively, the vertex with the highest degree can achieve higher pruning power. Then, we can generate bit vectors $qV_f(s)$ and

```
Procedure IM-GRN_Processing {
  Input: a gene feature database $\mathcal{D}$, a query gene feature matrix $M_Q$, an R$^*$-tree
         index $\mathcal{I}$ over embedded points from matrices $M_i$ in $\mathcal{D}$, an inverted bit-
         vector file, $IF$, an edge inference threshold $\gamma$, and a probabilistic threshold $\alpha$
  Output: IM-GRN query answers $A$
  (1) compute the query GRN graph $Q$ from matrix $M_Q$ using edge inference and
        graph existence pruning methods    // Lemma 3 in Section 3.2
  (2) obtain a gene (vertex), $g_s$, from GRN $Q$ with the highest degree,
        as well as its neighbor gene name set $NS(g_s)$ in $Q$
  (3) generate bit vectors $qV_f(s)$ and $qV_f(t)$ for query gene name $g_s$ and its
        neighbors in $NS(g_s)$, respectively
  (4) let $qV_d(s) = IF[g_s]$ and $qV_d(t) = O$
  (5) for each gene name $g_t \in NT(g_s)$
  (6)     $qV_d(t) = qV_d(t) \vee IF[g_t]$
  (7) $S_{cand} = \emptyset$;
  (8) initialize an empty priority queue $\mathcal{Q}$
  (9) for each entry $E_a \in root(\mathcal{I})$
  (10)    for each entry $E_b \in root(\mathcal{I})$
  (11)        if $qV_f(s) \wedge V_f(E_a) \neq 0$ and $qV_f(t) \wedge V_f(E_b) \neq 0$
                and $qV_d(s) \wedge V_d(E_a) \wedge qV_d(t) \wedge V_d(E_b) \neq 0$
  (12)            if $E_a$ and $E_b$ cannot be pruned by the index pruning    // Lemma 6
  (13)                insert $(height(\mathcal{I}) - 1, E_a, E_b)$
  (14) while $\mathcal{Q}$ is not empty
  (15)    $(key, E_s, E_t)$ = de-queue($\mathcal{Q}$)
  (16)    if $E_s$ (or $E_t$) is a leaf node
  (17)        for each gene $g_{i,s} \in E_s$
  (18)            for each gene $g_{i',t} \in E_t$
  (19)                if $qV_f(s) \wedge V_f(g_s) \neq 0$ and $qV_f(t) \wedge V_f(g_t) \neq 0$
                        and $qV_d(s) \wedge V_d(i) \wedge qV_d(t) \wedge V_d(i') \neq 0$ and $i = i'$
  (20)                    if $(g_{i,s}, g_{i',t})$ pair cannot be pruned by pivot-based
                          pruning and edge inference pruning
  (21)                        add gene pair $(g_{i,s}, g_{i',t})$ to $S_{cand}$
  (22)    else    // $E_s$ is an intermediate node
  (23)        for each child $c_s \in E_s$
  (24)            for each child $c_t \in E_t$
  (25)                if $qV_f(s) \wedge V_f(c_s) \neq 0$ and $qV_f(t) \wedge V_f(c_t) \neq 0$
                        and $qV_d(s) \wedge V_d(c_s) \wedge qV_d(t) \wedge V_d(c_t) \neq 0$
  (26)                    if $(c_s, c_t)$ cannot be pruned by the index pruning in Lemma 6
  (27)                        insert $(key - 1, c_s, c_t)$ into $\mathcal{Q}$
  (28) apply graph existence pruning on edges of $G_i$ in $S_{cand}$
  (29) refine the remaining candidate GRNs $G_i$ inferred from $G_i \in S_{cand}$
  (30) return actual IM-GRN query answers
}
```

Figure 4: Algorithm for IM-GRN Query Answering.

$qV_f(t)$ for gene names g_s and its neighbors $g_t \in NS(g_s)$, respectively (line 3). Similarly, we also initialize bit vectors $qV_d(s)$ and $qV_d(t)$ for possible query data source IDs, via inverted bit vector file IF (lines 4-6).

To traverse index $\mathcal{I}$, we use an initially empty priority queue, $\mathcal{Q}$ (line 8). Each element in $\mathcal{Q}$ is in the form (key, E_s, E_t), where key indicates the priority of element in the queue (smaller key has higher priority), and E_s and E_t are index nodes that may contain two interacting genes $g_{i,s}$ and $g_{i,t}$, respectively. Here, we use the level of the node E_s (or E_t) as key, which can traverse the tree index in a depth-first manner.

First, we consider pairwise entries E_a and E_b in the root, $root(\mathcal{I})$, of the index $\mathcal{I}$. If E_a (E_b) contains common gene names with the query gene names g_s ($g_t \in NS(g_s)$) (i.e., $qV_f(s) \wedge V_f(g_s) \neq 0$ and $qV_f(t) \wedge V_f(g_t) \neq 0$), E_a and E_b have common data source IDs (i.e., $qV_d(s) \wedge V_d(E_a) \wedge qV_d(t) \wedge V_d(E_b) \neq 0$), and this node pair cannot be pruned by the index pruning (Lemma 6), then we insert elements $(height(\mathcal{I}) - 1, E_a, E_b)$ into the queue $\mathcal{Q}$, where $height(\mathcal{I})$ is the height of the root (lines 9-13).

Each time we pop out an element (key, E_s, E_t) with the minimum key, key, from queue $\mathcal{Q}$ (lines 14-15). When entry E_s is a leaf node, we consider each pair of embedded points $g_{i,s}$ from E_s and $g_{i',t}$ from E_t (lines 16-18). If they do not have the same gene names or data source IDs as the query graph (verified by bit-AND operations over bit vectors), then we can safely prune this pair (line 19). Otherwise, we apply pivot-based pruning (in Section 4.2) and edge inference pruning (in Lemma 3) to filter out false alarm of the gene pair $(g_{i,s}, g_{i',t})$ (line 20). In the case that we cannot prune this pair, we add the gene pair $(g_{i,s}, g_{i',t})$ to a candidate set S_{cand} (line 21).

Parameters	Settings
γ	0.2, 0.3, **0.5**, 0.8, 0.9
α	0.2, 0.3, **0.5**, 0.8, 0.9
d	1, **2**, 3, 4
n_Q	2, 3, **5**, 8, 10
$[n_{min}, n_{max}]$	[10, 20], [20, 50], **[50, 100]**, [100, 200], [200, 300]
N	10K, 20K, **30K**, 40K, 50K, 100K

Table 2: The experimental settings.

When the index entry E_s (or E_t) is an intermediate node, we also check each child entry c_s in E_s and child c_t in E_t (lines 22-27). In particular, for each pair of child entries, (c_s, c_t), we check whether they have the same gene names and data source IDs as the query graph Q (line 25). If the answer is yes and the child pair cannot be further pruned by the index pruning method (given in Lemma 6), we need to add element $(key - 1, c_s, c_t)$ to queue $\mathcal{Q}$ for later checking, where $(key - 1)$ is the level of nodes c_s and c_t (line 26).

The index traversal terminates when the queue $\mathcal{Q}$ becomes empty (line 14). After that, we can obtain a number of candidate pairs in set S_{cand} that might correspond to subgraphs (of G_i) matching with Q with high confidences. We will then apply the graph existence pruning (in Lemma 5) to rule out false alarms. Finally, we refine candidate matrices M_i in S_{cand} by inferring their subgraphs G of G_i from M_i and checking the two matching conditions in Definition 4 (line 29), and return actual IM-GRN answers after the refinement (line 30).

The Complexity Analysis. Algorithm IM-GRN_Processing in Figure 4 first infers the query GRN graph Q from query gene feature matrix M_Q (line 1), which has the $O(l_Q \cdot n_Q^2)$ time complexity. Moreover, for lines 2-6, we construct synopses, w.r.t. query genes (and their neighbors) in the query graph Q, with a $O(n_Q)$ time complexity. Moreover, lines 7-27 traverse the index via a queue $\mathcal{Q}$. Given any two index nodes E_a and E_b (or E_s and E_t), the algorithm performs the pairwise checking for all entries in nodes (lines 9-10, 17-18, and 23-24) with the $O(F^2)$ cost, where F is the average fanout of index nodes. Therefore, given N $l_i \times n_i$ matrices, the average number of leaf nodes is given by $\frac{N \cdot n_i}{F}$, and the height of the tree index is $\lceil log_F\left(\frac{N \cdot n_i}{F}\right)\rceil$. As a result, the total number of index nodes is given by $O(\frac{N \cdot n_i}{F})$. Due to the pairwise checking on each node level, the time complexity of the index traversal (lines 8-27) is given by $O(N^2 \cdot n_i^2)$ in the worst case (i.e., without any pruning). Nevertheless, from our experimental results, the CPU time of traversing the index is quite efficient on average (e.g., less than 0.07 second in Figure 7(a)), which indicates the effectiveness of our (index) pruning methods. Furthermore, the cost of applying graph existence pruning and candidate refinement is given by $O(|S_{cand}| \cdot n_Q^2)$ (line 28), where $|S_{cand}|$ is the number of candidates after pruning. Finally, the remaining candidates are refined by performing the isomorphism checking and the probability calculation in Eq. (3).

6. EXPERIMENTAL EVALUATION

6.1 Experimental Settings

Real/synthetic data sets. In our experiments, we tested both real and synthetic data for gene feature databases. Specifically, for real data, we used three real gene feature data sets [22], $E.coli$, $S.aureus$, and $S.cerevisiae$, which include gene microarrays (matrices) and the network structure from organisms. Specifically, $E.coli$ contains a gene feature matrix with 4,511 genes (columns) and 805 observations/samples (rows) for each gene; $S.aureus$ has a 160 × 2810 gene feature matrix, and; $S.cerevisiae$ contains a 536 × 5950 matrix. As a gold standard structure [22], we know the ground truth of the actual GRN graphs for these real data sets (e.g., the actual GRN for $E.coli$ has 2,066 edges), which will be used for our experimental evaluation on the inference accuracy. We also randomly extract a combination of N small $l_i \times n_i$ matrices (which can be considered as sets of patient samples from distinct data sources) from these 3 real data sets for the efficiency test.

For synthetic data, we applied the classic GRN data generator in the biological literature [22], and produced N synthetic $l_i \times n_i$ gene feature matrices M_i for gene feature databases $\mathcal{D}$. In particular, we generate each $l_i \times n_i$ matrix M_i (for random $l_i \in [l_{min}, l_{max}]$ and $n_i \in [n_{min}, n_{max}]$), based on a linear model: $M_i = M_i \cdot B_i + E_i$, where B_i is an $n_i \times n_i$ adjacency matrix that specifies a GRN graph structure (i.e., each element is either non-zero or zero to indicate with or without edge, respectively), and E_i is an $l_i \times n_i$ error matrix. For error matrix E_i, each element is a random noise drawn from the Gaussian distribution $\mathcal{N}(0, 0.01)$. For adjacency matrix B_i, we first initialize it to a zero matrix, and then set each element in B_i to a random nonzero value e with probability $\frac{n_i \cdot deg(G)}{n_i \cdot (n_i - 1)}$, where $deg(G)$ (set to 1 by default) is the average (expected) indegree of each vertex in GRN. Here, the nonzero value e in B_i follows a variant of the Uniform or Gaussian (i.e., $\mathcal{N}(1, 0.01)$) distribution over two ranges $[-1, -0.5]$ and $[0.5, 1]$. Particularly, for the Gaussian distribution of e, we first produce a random value, e', following the normal distribution $\mathcal{N}(1, 0.01)$, and then let e be equal to e' if $e' \leq 1$ (otherwise, let $e = e' - 2$). Given matrices E_i and B_i, we can generate a synthetic gene feature matrix M_i by: $M_i = E_i \cdot (I - B_i)^{-1}$, where I is an $n_i \times n_i$ diagonal matrix.

This way, with different Uniform and Gaussian distributions of element $e \in B_i$ (in the linear model), we can obtain two types of synthetic data sets, denoted as Uni and Gau.

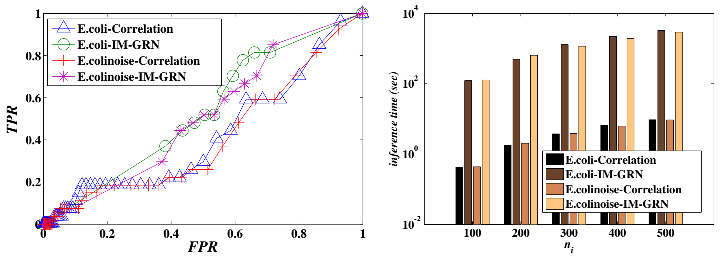
In order to evaluate the IM-GRN query performance, we randomly extract 20 query gene feature matrices, M_Q, from matrices M_i in database $\mathcal{D}$. That is, for each query matrix M_Q, we first select a random matrix $M_i \in \mathcal{D}$, and then extract n_Q gene feature vectors (columns) from M_i to form an $l_Q \times n_Q$ query gene feature matrix M_Q (note: during the extraction, we choose genes from matrix M_Q, such that their corresponding query GRN graph, Q, is connected), where l_Q has the same number of rows as that of matrix M_i (i.e., l_i) in $\mathcal{D}$, and n_Q is set to 5 by default.

Evaluation measures. Following the bioinformatics literature [22], we measure the accuracy of our IM-GRN inference approach by the *receiver operating characteristic* (ROC) curve, where x-axis of the curve is the *false positive rate* (FPR), and y-axis is the *true positive rate* (TPR) (a.k.a. *recall*). Here, TPR is given by the number of actual edges in the inferred GRNs divided by the total number of actual edges in GRNs, and FPR is defined as the percentage of incorrectly inferred edges. Intuitively, high TPR (recall ratio) and low FPR indicate good accuracy of the inference measure.

Regarding the query efficiency, we will report the performance of our proposed IM-GRN query processing approaches, in terms of the CPU time, the I/O cost, and the number of candidates. In particular, the CPU time is the time cost of retrieving candidates of IM-GRN query answers; the I/O cost is the number of page accesses during the IM-GRN query answering; the number of candidates is given by the number of the candidate genes after the index traversal and applying the pruning methods.

Competitors. Regarding the inference accuracy, in this paper, we will compare our IM-GRN inference approach (given in Definition 2) with the one (denoted as $Correlation$) that uses absolute Pearson's correlation coefficients (a.k.a. relevance networks [4]).

To the best of our knowledge, no prior works studied the IM-GRN problem that conducts the subgraph matching over online inferred GRNs (via an ad-hoc inference threshold) from gene feature matrices. A baseline method, $Baseline$, is to offline pre-compute



(a) ROC curve (b) inference time

Figure 5: Effectiveness and efficiency of $Correlation$ vs. IM-GRN inference over real data sets, $E.coli$, with and without noises $\mathcal{N}(0, 0.3)$.

and store existence probabilities of all possible edges between pairwise vertices in GRNs. Then, for an IM-GRN query, $Baseline$ online computes all GRN graphs G_i w.r.t. an ad-hoc inference threshold γ, and for each G_i, conduct the subgraph matching between G_i and a query GRN graph Q. Clearly, this baseline method needs to materialize N complete graphs offline, which is space-inefficient for the storage and time-inefficient for scanning pre-computed data during online IM-GRN query processing. In contrast, our proposed IM-GRN query answering algorithm utilizes effective pruning strategies that can significantly reduce the IM-GRN search space, and thus outperform the baseline method by orders of magnitude.

Parameter settings. Table 2 depicts the parameter settings in our experiments, where the default values of parameters are in bold font. For each set of experiments, we will vary one parameter at a time, while other parameters are set to their default values. All our experiments are conducted on a PC with Intel Core(TM)2 Duo 3.5GHz CPU with 32G memory.

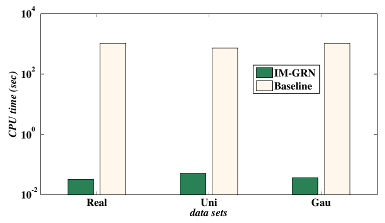
6.2 Effectiveness of the IM-GRN Inference

In the first set of experiments, we report the effectiveness of our IM-GRN inference approach (denoted as $IM{-}GRN$), compared with $Correlation$ (which uses the absolute Pearson's correlation coefficients to infer GRNs), over real data sets, $E.coli$, $S.aureus$, and $S.cerevisiae$. In order to test the robustness of our proposed IM-GRN inference approach over noisy data, we also use noisy real data sets, $E.coli + noise$, $S.aureus + noise$, and $S.cerevisiae + noise$ in which we added Gaussian noises, $\mathcal{N}(0, 0.3)$, to each element of matrices.

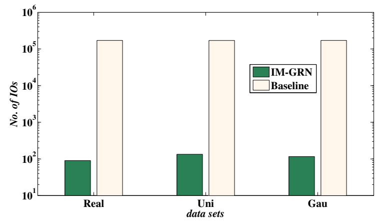
Figure 5(a) shows the *receiver operating characteristic* (ROC) of our $IM{-}GRN$ inference approach over $E.coli$ and $E.coli + noise$ data sets, compared with $Correlation$, where points (FPR, TPR) are plotted with respect to different inference thresholds γ from 0 to 1 (with an increment of 0.01), and $n_i = 200$. Intuitively, high TPR and low FPR values indicate good accuracy of the inference approach. In the figure, for either $E.coli$ or $E.coli + noise$, the ROC curve of our IM-GRN approach is above that of $Correlation$ in most cases, which indicates that $IM{-}GRN$ is more effective than $Correlation$ for the GRN inference. Moreover, our $IM{-}GRN$ approach over $E.coli$ with and without noises has similar ROC curves, which confirms the robustness of our $IM{-}GRN$ inference approach against noises. Please also refer to similar ROC curves of $IM{-}GRN$ and $Correlation$ for the other two real data sets, $S.aureus$ and $S.cerevisiae$, in Appendix G.

In addition, we also compare our $IM{-}GRN$ inference approach with another inference measure, *partial correlation*, denoted as $pCorr$. Similar experimental results are given in Appendix H.

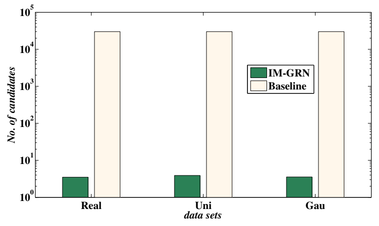
Figure 5(b) illustrates the efficiency of the two inference approaches on $E.coli$ data sets with different graph sizes $n_i = |V(G_i)|$ from 100 to 500. In this figure, our $IM{-}GRN$ inference approach requires higher time cost than $Correlation$. This is reasonable, since our $IM{-}GRN$ inference measure (given in Eq. (1)) needs to



(a) CPU time

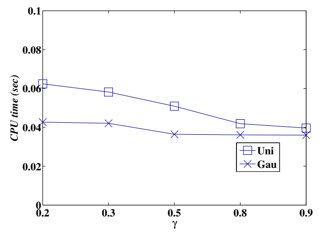


(b) I/O cost

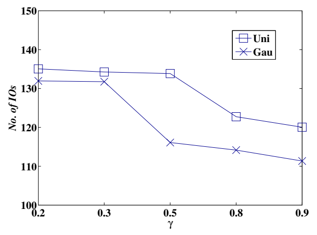


(c) No. of candidates

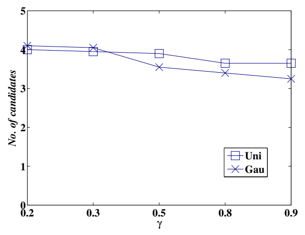
Figure 6: IM-GRN vs. Baseline over real/synthetic data sets.



(a) CPU time

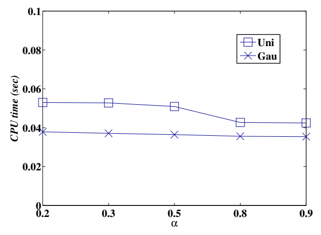


(b) I/O cost

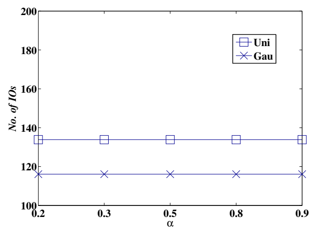


(c) No. of candidates

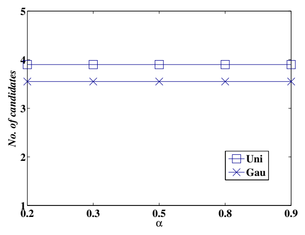
Figure 7: IM-GRN query performance vs. the ad-hoc inference threshold γ.



(a) CPU time

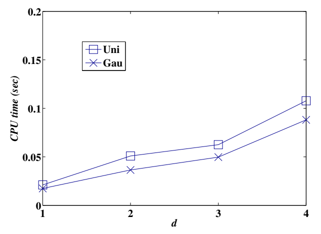


(b) I/O cost

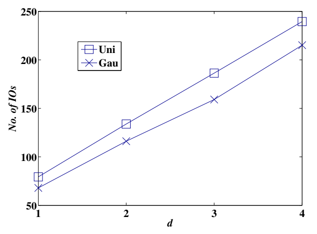


(c) No. of candidates

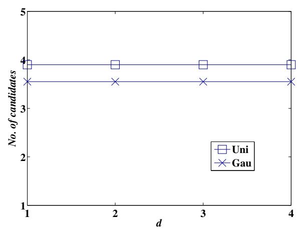
Figure 8: IM-GRN query performance vs. the probabilistic threshold α.



(a) CPU time



(b) I/O cost



(c) No. of candidates

Figure 9: IM-GRN query performance vs. the number of pivots d (or the dimensionality, $(2d+1)$, of the index).

compute the Pearson's correlation coefficients for multiple randomized vectors X_t^R, whereas $Correlation$ only needs to calculate the coefficient once.

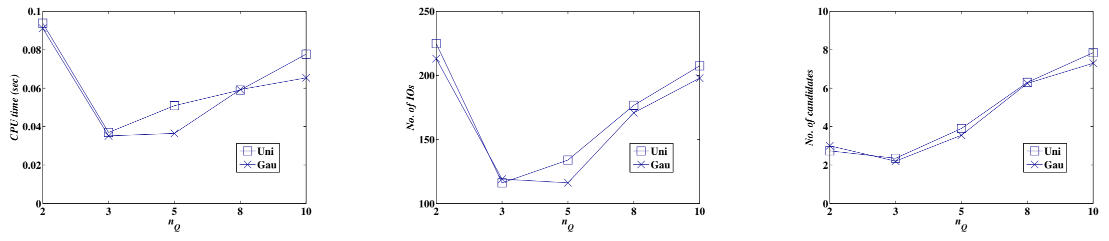
From the experiments above, compared with $Correlation$, our $IM-GRN$ inference approach trades the efficiency for the inference accuracy. Due to high time cost of $IM-GRN$ inference, it is not efficient to infer/materialize all GRNs during the IM-GRN search, which inspires us to propose encoding/pruning techniques. In the next subsection, we will exactly test the efficiency of our proposed IM-GRN query processing approaches.

6.3 Efficiency of IM-GRN Query Answering

IM-GRN vs. $Baseline$ **over real/synthetic data sets.** First, we compare the efficiency of our proposed IM-GRN approach with that of the baseline method, $Baseline$, over real/synthetic data sets, $Real$, Uni, and Gau. For real data set, $Real$, we use a combination of 3 real-life data, $S.aureus$, $E.coli$, and $S.cerevisiae$, which contains N matrices ($N/3$ matrices from each type of data). Taking $E.coli$ as an example, we randomly extract $\frac{N}{3}\ l_i \times n_i$ matrices from the original $4,511 \times 805$ $E.coli$ matrix, where other parameters are set to their default values (the same as synthetic data). In Figures 6(a) and 6(b), we can see that our IM-GRN approach over both real and synthetic data sets can achieve low CPU time (below 0.16 second) and I/O cost (around 100 page accesses), which outperforms the baseline method by 2-3 orders of magnitude. Moreover, as shown in Figure 6(c), the number of the remaining candidates after pruning in our IM-GRN approach is only around $3 \sim 4$, which is much smaller than that of $Baseline$. Thus, the experimental results confirm the effectiveness of our proposed pruning methods, and efficiency of our IM-GRN query answering approach, compared with the $Baseline$ method.

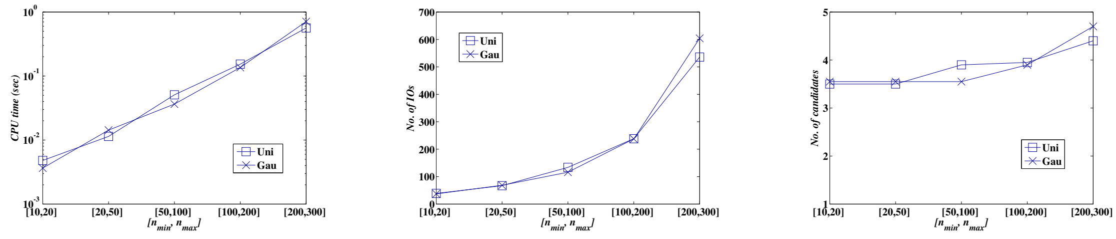
To clearly illustrate the performance trends, in subsequent experiments, we will only report the efficiency of our IM-GRN approach on synthetic data, Uni and Gau, by varying different parameters.

The IM-GRN performance vs. the ad-hoc inference threshold, γ**.** We next show the effect of the ad-hoc inference threshold γ on the IM-GRN query performance, by varying γ from 0.2 to 0.9, where other parameters are set to their default values. When the user-specified inference threshold γ increases, the number of potential candidate genes that may match with query genes in the



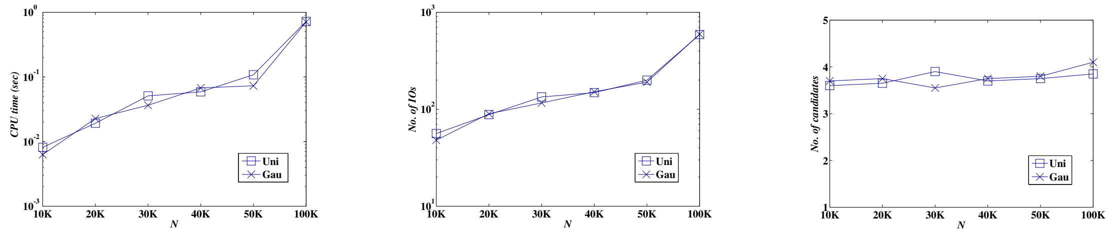
(a) CPU time (b) I/O cost (c) No. of candidates

Figure 10: IM-GRN query performance vs. the number, n_Q, of genes in query gene feature matrix M_Q.



(a) CPU time (b) I/O cost (c) No. of candidates

Figure 11: IM-GRN query performance vs. the range, $[n_{min}, n_{max}]$, of the number n_i of genes per matrix.



(a) CPU time (b) I/O cost (c) No. of candidates

Figure 12: IM-GRN query performance vs. the total number, N, of gene feature matrices.

query graph Q decreases (as shown in Figure 7(c)). Thus, as illustrated in Figures 7(a) and Figure 7(b), for larger γ value, it requires less time and I/O costs to process fewer candidates during the IM-GRN query answering over the index.

For both Uni and Gau synthetic data sets, with different γ values, the CPU times are low, that is, around $0.04 \sim 0.07$ second. Moreover, the number of page accesses (i.e., I/O cost) is about 110 $\sim$ 134. Furthermore, only $3 \sim 4$ remaining candidates for refinement are left after the pruning, which indicates high pruning power of our proposed pruning and indexing strategies.

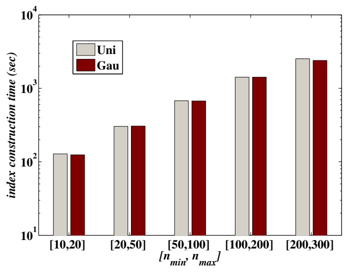
The IM-GRN performance vs. the probabilistic threshold, α. Figure 8 presents the query performance of our proposed IM-GRN approach for different given probabilistic thresholds α, where $\alpha = 0.2, 0.3, 0.5, 0.8, 0.9$, and default values are used for other parameters. As given by Definition 4, larger α threshold will filter out more probabilistic subgraphs G with low appearance probabilities. Therefore, when the probabilistic threshold α becomes larger, more subgraph candidates can be quickly ruled out by our pruning methods, which thus leads to fewer CPU times (as given in Figure 8(a)). In Figure 8(b), the I/O cost during the index traversal is not very sensitive to different α values, and remains low (around 116 $\sim$ 134). Moreover, as illustrated in Figure 8(c), after the pruning, the number of remaining candidates is also low, that is $3.5 \sim 3.9$, which shows the good pruning ability of our IM-GRN approach.

The IM-GRN performance vs. the number of pivots, d. Figure 9 shows the effect of the number, d, of the used pivots for embedding/indexing on the IM-GRN query performance, where d varies from 1 to 4, and other parameters are set to their default values. Note that, with the number of pivots d, the dimensionality of embedded points in the index is given by $(2d + 1)$. From figures, we can see that, when d becomes larger, the CPU time and the I/O cost also increase. This is due to the problem of the "dimensionality curse" [2, 18, 21], that is, the query performance degrades dramatically for high dimensionality. This can be confirmed by the increase of time and I/O costs in Figures 9(a) and 9(b), respectively. Due to the same query set over indexes of different reduced dimensions, the number of candidates remains the same.

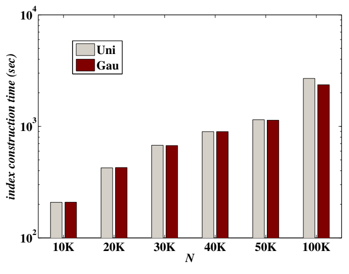
Nonetheless, for both data sets Uni and Gau, the CPU time is low (i.e., between 0.017 and 0.1 second) for different reduced dimensions. The I/O cost is around $68 \sim 240$. Furthermore, the number of the remaining candidates is also small (i.e., $3.55 \sim 3.9$).

The IM-GRN performance vs. the number of query genes, n_Q. Figure 10 reports the experimental results of our IM-GRN query processing approach over Uni and Gau synthetic data sets, by varying the number of genes, n_Q, in the query GRN graph from 2 to 10, where other parameters are set to default values. As shown in Figures 10(a) and 10(b), when n_Q becomes larger, both CPU time and I/O cost first decrease, and then increase. This is reasonable, since more (i.e., larger n_Q) query genes can filter out more candidates (which reduces the computation and I/O costs), but introduce higher costs of processing more query genes through the index. Thus, these two factors influence the CPU time and I/O cost at the same, and lead to "U" curves. Similarly, as illustrated in Figure 10(c), the number of matching gene candidates first decreases (for higher pruning power, with larger n_Q), and then increases (due to more candidate genes matching with more n_Q query genes).

Similar to previous results, the CPU time is low (i.e., around $0.03 \sim 0.1$ second), the I/O cost is $116 \sim 225$, and the number of candidates is between 2 and 8. This indicates the effectiveness of our proposed pruning strategies, and the efficiency of the IM-GRN query answering approach.



(a) indexing time vs. $[n_{min}, n_{max}]$



(b) indexing time vs. N

Figure 13: The time cost of the index construction.

The IM-GRN performance vs. the range of the number of genes per matrix, $[n_{min}, n_{max}]$**.** Figure 11 varies the range, $[n_{min}, n_{max}]$, for the number, n_i, of genes (columns) in matrix M_i from $[10, 20]$ to $[200, 300]$, and illustrates the IM-GRN query performance over Uni and Gau data sets, where default values for other parameters are used. From figures, with the increase of the range for n_i, the CPU time and the I/O cost both increase. This is because more genes (i.e., larger n_i) are involved in each matrix, which results in more gene candidates that may match with the query genes. Nonetheless, the CPU time is low (i.e., $0.003 \sim 0.71$ second), and the I/O cost is around $37 \sim 604$. Furthermore, the number of candidates is small (i.e., between 3.5 and 4.7). This indicates the scalability of our proposed IM-GRN approach, against parameter $[n_{min}, n_{max}]$.

The IM-GRN performance vs. the number of gene feature matrices, N**.** Figure 12 tests the scalability of our proposed IM-GRN query processing approach, by changing the number, N, of gene feature matrices in database $\mathcal{D}$ from $10K$ to $100K$, where other parameters are assigned with default values. When the size, N, of database $\mathcal{D}$ becomes larger, both time and I/O costs are smoothly increasing over Uni and Gau data sets. Nevertheless, as depicted in Figures 12(a) and 12(b), the CPU time remains low (i.e., 0.006 $\sim$ 0.72 second), and the I/O cost is about $48 \sim 595$, which indicates the good scalability of our IM-GRN approach against large number of matrices, N, in the database.

As illustrated in Figure 12(c), the number of the remaining candidates is only around $3 \sim 4$, which implies that our pruning methods can achieve high pruning power against large database size N.

The index construction time. Finally, we also report the index construction time over Uni and Gau data sets in Figure 13, with respect to different ranges of gene (column) numbers per matrix, $[n_{min}, n_{max}]$, and different numbers of matrices, N, in the database $\mathcal{D}$. From figures, we can see that, when the values within the range $[n_{min}, n_{max}]$ become larger or the number of matrices N increases, we need to insert more transformed points (from gene feature vectors) into the R*-tree index, which thus leads to higher index construction time.

7. RELATED WORK

In this section, we overview previous works on biological network inference and probabilistic graph databases.

Biological network inference. Previous works on the GRN inference can be classified into two categories, the learning-based [26, 9] and scoring-based [4, 23] methods. Specifically, the learning-based approaches learn GRN graph structures via *Bayesian network* [26] or L_1 *regularized linear regression* (Lasso) [9]. The scoring-based approaches infer edges via scores between any two gene features, such as *Correlation* [4] or *Mutual Information* [23, 3]. These two types of inferences have their own disadvantages. For example, the learning-based approaches, like Bayesian network or Lasso methods, need to tune parameters, and require high computation costs over samples. On the other hand, the scoring-based approaches are fast to calculate the regulatory score, however, not so robust, in the presence of small sample sizes and noisy samples. In contrast, in this work, we proposed a robust probabilistic measure, that is, the probability that two gene feature vectors are correlated with each other, based on vector randomization.

Probabilistic graph databases. In real-world applications such as biological data analysis [13], Semantic Web [19], social networks [11], workflow graphs [28], and transportation system [15, 20], many application data can be inherently modeled by probabilistic graphs. The uncertainties in probabilistic graphs can usually be classified into two categories, vertex/edge label uncertainty and edge existence uncertainty.

For the vertex/edge label uncertainty, prior works studied probabilistic graphs, in which each vertex or edge has multiple possible labels with existence probabilities. For example, road networks [15, 20] can be modeled by probabilistic graphs with uncertain traffic conditions (i.e., velocity samples associated with probabilities). Regarding edge existence uncertainty in probabilistic graphs, prior works [16, 24, 27] modeled probabilistic graphs that contain edges associated with existence probabilities. In this paper, our GRN graph model falls into this category.

However, in contrast to prior works, GRNs are not offline materialized as a static graph database, but, instead, online inferred (via different ad-hoc inference thresholds γ) from gene feature matrices (i.e., microarray data). Thus, our IM-GRN problem needs to infer ad-hoc GRNs online, and meanwhile efficiently retrieve IM-GRN query answers, which has not been studied before. Furthermore, since gene feature matrices are of different sizes, we have to design novel approaches to embed matrices of distinct sizes, and propose indexing or query processing algorithms specific for IM-GRN. Therefore, previous techniques on static probabilistic graph databases cannot be directly used for online inferred GRNs.

8. CONCLUSIONS

In this paper, we study an important problem, namely IM-GRN, for ad-hocly inferring and matching GRN graphs. To efficiently and effectively answer IM-GRN queries, we model GRNs by probabilistic graphs, and propose effective reduction and pruning methods to greatly filter out GRN false alarms. Moreover, we design novel cost-model-based embedding/indexing mechanisms which can facilitate IM-GRN query answering without materializing all GRNs, and present an efficient IM-GRN query procedure. Extensive experiments have demonstrated the efficiency and effectiveness of our IM-GRN approach on real and synthetic GRN data.

As a future work, we would like to develop a real prototype system, which integrates our proposed technologies (e.g., probabilistic GRN graph model, embedding, pruning, indexing, and algorithms). In particular, the system organizes gene feature data from various data sources (e.g., public biological databases and experimental data from institutions) and provides users with an interface to conduct ad-hoc IM-GRN queries over gene feature databases. Users can specify as the input those samples of gene features or a query GRN graph Q (e.g., a cancer biomarker, a representative GRN pattern in a disease cluster, or experimental data from a newly emerging and unclassified disease), and the system will use our proposed technologies to quickly return the matching GRNs inferred from gene feature databases as the output.

9. ACKNOWLEDGMENTS

This work is supported by Lian Start Up No. 220981, Kent State University.

Appendix

10. REFERENCES

[1] N. Beckmann, H. Kriegel, R. Schneider, and B. Seeger. The R*-tree: an efficient and robust access method for points and rectangles. In *SIGMOD*, 1990.

[2] S. Berchtold, D. A. Keim, and H. P. Kriegel. The X-tree: An index structure for high-dimensional data. In *VLDB*, 1996.

[3] A. J. Butte, I. S. Kohane, and I. S. Kohane. Mutual information relevance networks: Functional genomic clustering using pairwise entropy measurements. In *Biocomput.*, 2000.

[4] A. J. Butte, P. Tamayo, D. Slonim, T. R. Golub, and I. S. Kohane. Discovering functional relationships between rna expression and chemotherapeutic susceptibility using relevance networks. In *Proceedings of the National Academy of Sciences*, 2000.

[5] M. Dehmer, M. Grabner, A. Mowshowitz, and F. Emmert-Streib. An efficient heuristic approach to detecting graph isomorphism based on combinations of highly discriminating invariants. *Adv. Comput. Math.*, 39, 2013.

[6] M. Dehmer, L. Mueller, and F. Emmert-Streib. Quantitative network measures as biomarkers for classifying prostate cancer disease states: a systems approach to diagnostic biomarkers. *PLoS ONE*, 8(11), 2013.

[7] F. Emmert-Streib, M. Dehmer, and B. Haibe-Kains. Gene regulatory networks and their applications: understanding biological and medical problems in terms of networks. *Frontiers in Cell and Developmental Biology*, 2(38), 2014.

[8] K. Fortney, W. Xie, M. Kotlyar, J. Griesman, Y. Kotseruba, and I. Jurisica. Networx: connecting drugs to networks and phenotypes in saccharomyces cerevisiae. *Nucleic Acids Res.*, 41, 2013.

[9] G. Geeven, R. E. van Kesteren, A. B. Smit, and M. C. de Gunst. Identification of context-specific gene regulatory networks with gemula - gene expression modeling using lasso. In *Bioinformatics*, 2012.

[10] R. Gentleman, B. Ding, S. Dudoit, and J. Ibrahim. Distance measures in dna microarray data analysis. In *Bioinformatics and Computational Biology Solutions Using R and Bioconductor, Statistics for Biology and Health, Chapter 12*. Springer, 2005.

[11] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. A data-based approach to social influence maximization. *PVLDB*, 5(1), 2011.

[12] D. Hanahan and R. A. Weinberg. Hallmarks of cancer: the next generation. *Cell*, 144, 2011.

[13] A.-C. Haury, F. Mordelet, P. Vera-Licona, and J.-P. Vert. TIGRESS: trustful inference of gene regulation using stability selection. *BMC Systems Biology*, 6:145, 2012.

[14] J. D. Hoheisel. Microarray technology: Beyond transcript profiling and genotype analysis. In *Nature Reviews Genetics*, 2006.

[15] M. Hua and J. Pei. Probabilistic path queries in road networks: traffic uncertainty aware path selection. In *EDBT*, 2010.

[16] H. Huang and C. Liu. Query evaluation on probabilistic RDF databases. In *WISE*, 2009.

[17] R. Jampani, F. Xu, M. Wu, L. L. Perez, C. Jermaine, and P. J. Haas. MCDB: a monte carlo approach to managing uncertain data. In *SIGMOD*, 2008.

[18] N. Katayama and S. Satoh. The SR-tree: an index structure for high-dimensional nearest neighbor queries. In *SIGMOD*, 1997.

[19] X. Lian and L. Chen. Efficient query answering in probabilistic RDF graphs. In *SIGMOD*, 2011.

[20] X. Lian and L. Chen. Trip planner over probabilistic time-dependent road networks. *TKDE*, 26(8), 2014.

[21] K.-I. Lin, H. V. Jagadish, and C. Faloutsos. The TV-tree: An index structure for high-dimensional data. *VLDB J.*, 3(4), 1994.

[22] D. Marbach, J. C. Costello, R. Küffner, N. M. Vega, R. J. Prill, D. M. Camacho, K. R. Allison, M. Kellis, J. J. Collins, G. Stolovitzky, et al. Wisdom of crowds for robust gene network inference. *Nature methods*, 9(8), 2012.

[23] A. A. Margolin, I. Nemenman, K. Basso, C. Wiggins, G. Stolovitzky, R. D. Favera, and A. Califano. ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, 7(S-1), 2006.

[24] M. Potamias, F. Bonchi, A. Gionis, and G. Kollios. k-nearest neighbors in uncertain graphs. *PVLDB*, 3, 2010.

[25] R. De Smet and K. Marchal. Advantages and limitations of current network inference methods. In *Nature Reviews Microbiology*, 2010.

[26] N. Xuan, M. Chetty, R. Coppel, and P. P. Wangikar. Gene regulatory network modeling via global optimization of high-order dynamic bayesian network. *BMC bioinformatics*, 13(1), 2012.

[27] Y. Yuan, G. Wang, L. Chen, and H. Wang. Graph similarity search on large uncertain graph databases. *VLDB J.*, 24(2), 2015.

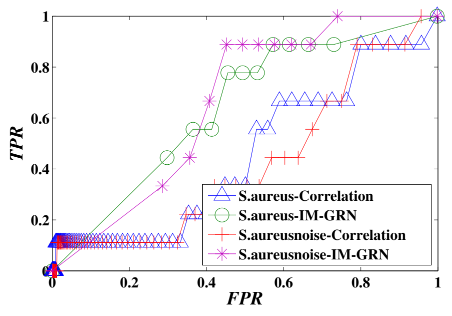
[28] X. Zhu, S. Song, J. Wang, P. S. Yu, and J. Sun. Matching heterogeneous events with patterns. In *ICDE*, 2014.

A. Discussions on a Class of Problems Generalized from IM-GRN and Their Applications/Solutions.

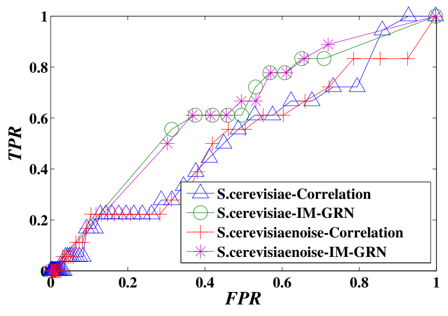
Our IM-GRN problem can be generalized to a general class of problems, which are related to queries over ad-hocly inferred graphs, where graph structures (or graph edges) are not deterministic, but are ad-hocly inferred from contents of vertices or other relevant data sources. This class of problems has the applications in social network analysis, near-duplicate video detection, and so on.

For example, in the case of social networks, the influences of one person to another with respect to specific ad-hoc keywords/topics (e.g., of a given advertisement type) can be ad-hocly inferred by their profiles, friendship, interactions (e.g., tweets and retweets), similarity of their posted messages, and so on, Thus, the entire influence networks in social networks cannot be materialized offline (against ad-hoc online keywords/topics) in advance, and a number of query types (e.g., pattern matching) over such an ad-hocly inferred influence networks are important and worthy to study in social media.

Similarly, in real applications like video copyright-violation detection, it is important to perform the near-duplicate video retrieval over multimedia databases, where near-duplicate videos may be modified, scaled, and/or rotated from the original videos. In particular, each keyframe of videos may contain features such as color histograms, texture, and some local interest points (or keypoints corresponding to some objects). One interesting direction is to model each potentially duplicate video by a graph, in which vertices are keyframes in the video, and edges indicate the similarity between two keyframes inferred from image features (i.e., based



(a) $S.aureus$



(b) $S.cerevisiae$

Figure 14: ROC curves of $Correlation$ vs. IM-GRN inference over real data sets, $S.aureus$ and $S.cerevisiae$, with and without noises $\mathcal{N}(0, 0.3)$.

on color histograms, textures, interest points, and so on). Given a query video (with copyright) and an ad-hoc similarity threshold (i.e., an inferred query graph), we can conduct an ad-hoc search over video databases (online inferred graphs) to identify near-duplicate videos.

To tackle this general class of problems, we need to design specific techniques (e.g., embedding, pruning, indexing, and query processing algorithms). In particular, similar to IM-GRN, with respect to edge existence measures between two vertices, we need to devise effective pruning methods to reduce the search space. For example, with *partial correlation* or *mutual information*, we can devise specific pruning conditions for them. The rationale behind the pruning is similar, that is, we want to quickly discard those false alarms of edges $e_{i,j}$ with low existence probabilities ($< \gamma$). One possible direction is to start from definitions of probabilities (defined w.r.t. partial correlations or mutual information), and derive a lower bound of the edge existence probability $e_{i,j}.p$. If this lower bound is smaller than γ, then this edge $e_{i,j}$ does not exist in the GRN graph. Next, with the pruning conditions, we can further derive space-efficient synopses to embed the matrix data at vertices (e.g., via pivots) for indexing and query answering. We would like to leave topics of designing accurate and efficient techniques/algorithms over ad-hoc probabilistic graphs for specific measures or inference approaches (e.g., partial correlation, mutual information, Fisher's transform, or Student's t-test) as our future work.

B. Proof of Lemma 1.

PROOF. From [10], the relationship between Pearson's correlation coefficient and Euclidean distance between two standardized vectors can be given as follows:

$$dist(X_s, X_t) = \sqrt{2 \cdot l_i \cdot (1 - cor(X_s, X_t))}, \qquad (11)$$

where $cor(\cdot, \cdot)$ is the Pearson's correlation coefficient.

We can rewrite Eq. (11) as:

$$r(X_s, X_t) = |cor(X_s, X_t)| = \left|1 - \frac{dist^2(X_s, X_t)}{2 \cdot l_i}\right|. \qquad (12)$$

By substituting Eq. (12) into Eq. (1), we can obtain:

$$\begin{aligned} e_{s,t}.p &= Pr\{r(X_s, X_t) > r(X_s, X_t^R)\} \qquad (13)\\ &= Pr\left\{\left|1 - \frac{dist^2(X_s, X_t)}{2 \cdot l_i}\right|^2 > \left|1 - \frac{dist^2(X_s, X_t^R)}{2 \cdot l_i}\right|^2\right\}\\ &= Pr\left\{\left(2 - \frac{dist^2(X_s, X_t) + dist^2(X_s, X_t^R)}{2 \cdot l_i}\right)\right.\\ &\qquad \left.\cdot \frac{dist^2(X_s, X_t^R) - dist^2(X_s, X_t)}{2 \cdot l_i} > 0\right\} \end{aligned}$$

Note that, in Eq. (13), since X_s and X_t are standardized unit vectors, we have: $dist^2(X_s, X_t) = \sum_{k=1}^{l_i}(X_s[k] - X_t[k])^2 \leq 4$. Similarly, we have $dist^2(X_s, X_t^R) \leq 4$. Moreover, since $l_i \geq 2$, the first term within $Pr\{\cdot\}$ is positive, Eq. (13) can be rewritten as:

$$\begin{aligned} e_{s,t}.p &= Pr\{r(X_s, X_t) > r(X_s, X_t^R)\}\\ &= Pr\{dist^2(X_s, X_t^R) > dist^2(X_s, X_t)\}\\ &= Pr\{dist(X_s, X_t^R) > dist(X_s, X_t)\}, \end{aligned}$$

which is exactly equivalent to Eq. (4). Hence, the lemma holds. □

C. Proof of Lemma 3.

PROOF. From the lemma assumption, we have $ub\_P(e_{s,t}) \leq \gamma$. Moreover, since $ub\_P(e_{s,t})$ is an upper bound of the edge existence probability $e_{s,t}.p$ (i.e., $e_{s,t}.p \leq ub\_P(e_{s,t})$), we can apply the inequality transition, and obtain $e_{s,t}.p \leq \gamma$. According to Definition 2, edge $e_{s,t}$ exists, only if $e_{s,t}.p > \gamma$ holds. Hence, we can conclude that edge $e_{s,t}$ does not exist in the inferred GRN graph G_i. □

D. Proof of Lemma 4.

PROOF. Based on the Markov's inequality, since $dist(X_s, X_t)$ is a positive value, we have:

$$e_{s,t}.p = Pr\{Z > dist(X_s, X_t)\} \leq \frac{E(Z)}{dist(X_s, X_t)}.$$

Thus, $\frac{E(Z)}{dist(X_s, X_t)}$ is an upper bound of probability $e_{s,t}.p$. Hence, the lemma holds. □

E. Proof of Lemma 5.

PROOF. Since $UB\_Pr\{G\}$ is an upper bound of probability $Pr\{G\}$, we have $Pr\{G\} \leq UB\_Pr\{G\}$. From the lemma assumption that $UB\_Pr\{G\} \leq \alpha$, we can apply the inequality transition and obtain $Pr\{G\} \leq \alpha$. Based on Definition 4, since the subgraph G of GRN G_i has low appearance probability $Pr\{G\}$ (i.e., less than or equal to α), G cannot be our IM-GRN query answers. □

F. Proof of Lemma 6.

PROOF. From Inequality (9), we have the condition of pruning gene X_t that: $y_t[w] \leq \gamma \cdot x_t[r] - (\gamma \cdot x_s[r] + \gamma \cdot x_s[w])$. Our goal is to prove that, if Inequality (9) holds for any two genes X_s and X_t under nodes E_a and E_b, respectively, then node E_b can be safely pruned (since any gene X_t under E_b can be pruned due to $X_s \in E_a$. Therefore, we can derive Inequality (10) from Inequality (9), by overestimating LHS and underestimating RHS of Inequality (9). That is, in Inequality (9), we replace $y_t[w]$ ($= E(dist(X_t^R, piv_w))$) with $E_{by}^+[w]$ and $x_t[r]$ ($= dist(X_t, piv_r)$) with $E_{bx}^-[r]$, for a group of points in node E_b, and relax $x_s[w]$ (or $x_s[r]$) with $E_{ax}^+[w]$ (or $E_{ax}^+[r]$), which exactly corresponds to Inequality (10). □

G. Experimental Results on ROC Curves Over Real Data Sets, $S.aureus$ and $S.cerevisiae$.

In Figure 14, we report the ROC curves of $IM\!-\!GRN$ and $Correlation$ over real data sets, $S.aureus$ and $S.cerevisiae$, with and without noises (Gaussian noises $\mathcal{N}(0, 0.3)$). Similar to the results of $E.coli$, our $IM\!-\!GRN$ approach has the ROC curve above that of $Correlation$ in most cases for both data sets (with or without noises), which indicates the effectiveness and robustness of our $IM\!-\!GRN$ inference approach.

H. Experimental Results on ROC Curves of $IM\!-\!GRN$ vs. Partial Correlation Over Real Data Set, $E.coli$.

We compare our $IM\!-\!GRN$ inference approach with *partial correlation*[2], denoted as $pCorr$, over real data set $E.coli$ and $E.coli+noise$, in terms of the ROC curve. The experimental results are reported in Figure 15. Similarly, our IM-GRN inference approach can achieve high TPR and low FPR, compared with $pCorr$.

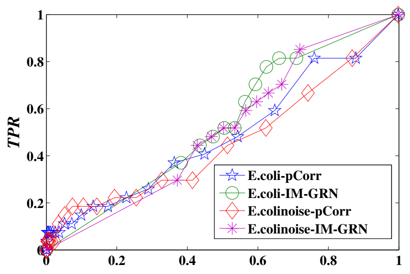


Figure 15: ROC curves of $pCorr$ vs. IM-GRN inference over real data set, $E.coli$, with and without noises $\mathcal{N}(0, 0.3)$.

[2]https://en.wikipedia.org/wiki/Partial_correlation