Integrating and Ranking Uncertain Scientific Data

University of Washington Technical Report, UW-CSE-08-06-03

Landon Detwiler [1], Wolfgang Gatterbauer [2], Brent Louie [1], Dan Suciu [2], Peter Tarczy-Hornoch [1]

[1]Biomedical and Health Informatics, University of Washington, Seattle, USA

{det,brlouie,pth}@u.washington.edu

[2]Computer Science and Engineering, University of Washington, Seattle, USA

{gatter,suciu}@cs.washington.edu

Abstract

The *BioRank* project investigates formalisms for modeling uncertainties of scientific data in a mediator-based data integration system. Our motivating application is determining previously not known functions of proteins. Much can be learned in biology by integrating known information with biological similarity functions and confidence values of experimental results. In this paper, we evaluate the role that probabilities can play in such a scenario with the help of real-world data. In particular, we show that: (i) explicit modeling of uncertainties increases the quality of our functional predictions for *less-* and *unknown* functions, not however for *well-known* functions. This suggests exploratory search and new knowledge discovery as ideal application domains for probabilistic data integration; (ii) slight perturbations of the input probabilities do not severely affect the quality of our predictions. This suggests that probabilistic information integration is actually robust against slight variations in the way uncertainties are transformed into probabilities by domain experts; and (iii) our probabilistic scoring functions can be evaluated efficiently with the help of several techniques. This suggests that probabilistic query evaluation is not as hard for real-world problems as theory indicates.

1 Introduction

Explicit management of uncertainties is becoming imperative in the scientific domain. Data in the life sciences and especially in molecular biology is highly dispersed and inherently uncertain. New insights can be gained by integrating and analyzing existing information from separate sources, but current database techniques are not suited for integrating data with many kinds of uncertainties [1, 34]. Most of the recent work on managing data with uncertainties has postulated that uncertainties are quantified with probabilities and then mostly focused on efficient evaluation techniques on probabilistic databases [12, 46, 36, 38, 13, 16, 2]. Comparatively little has been done to examine the actual validity of the fundamental assumption, that *probabilities are the right model for representing uncertainties in real-world applications.*

In this paper, we study the validity of this assumption in the context of an important problem in molecular biology: assigning functions to proteins, also known as *functional protein annotation.* Thousands of proteins have no known function or their full complement of functions is not known. Having a complete list of known and possible further functions is essential to e.g. biologists that study the pathogenic pathways of protein complexes. The two major approaches for protein

function prediction are actually modeling the physical folding process and structure of proteins [6], or integrating existing knowledge with biological similarity functions and manually handling experimental and computational evidence in the process [34, 26]. The role of a data integration system is to help in the second approach: in this paper we study how the explicit management of uncertainties in scientific data integration helps during protein predication.

Motivating example. We give a simple example of how scientists deal with uncertainties during manual data integration. Suppose a researcher is searching for some new, possible yet *unknown* functions of the protein corresponding to gene ABCC8. Some previously studied proteins have *well-known* functions, which are recorded somewhere in the data sources, and our researcher definitely wants to retrieve these. But she is especially interested in *unknown yet likely* functions, which she can later validate. There are several data sources where she can search for ABCC8. One is EntrezGene, a curated database, which returns 12 GO functions for ABCC8. The Gene Ontology (GO) is a shared vocabulary of biological functions. Each of those functions is annotated with a status code like "reviewed" or "predicted" that represents the certainty that this function is correct: she can use these annotations to rank the 12 functions. She can also get the protein's sequence from EntrezProtein and then paste it into TigrFam, a Hidden Markov Model matching algorithm, which returns 53 GO functions: 3 are included in the previous 12 from EntrezGene, the rest are new. The functions have a new kind of uncertainty: the expectation values (e-values) that represent the certainty of the result on an exponential scale. There are more such sources, NCBIBlast returns protein sequences that are similar to that of ABCC8 (there are 100 such), whose GO functions (41) are further obtained from EntrezGene; and Pfam (another sequence matching database) returns more functions. Altogether she ends up with a list of 97 candidate functions for her protein. At this point her data integration task has ended, and she would have to continue to evaluate these candidate functions manually, giving preference to those that are more "certain" or likely to be correct.

The only way to validate the functions of a protein is to experimentally test it for that function. Such experiments are always function-specific and costly. Researchers, therefore, have to decide which functions to focus on and test for. Any tool that allows biologists to pre-sort and rank candidate functions given available but heterogeneous evidence is valuable. The goal of an automatic data integration system is to help her rank these results according to the degree of uncertainty, and this is not an easy task. ABCC8 is a well studied gene, and 13 of the 97 candidate functions are *well-known*: they are confirmed in iProClass, a reference curated database. Clearly we want the system to rank these functions high. But this turns out to be the easier part, since for confirmed functions there is usually enough redundant evidence in the data sources which can be used to rank them highly. The question is whether the system can also rank highly the unknown functions, which are likely functions but for which there is not a lot of evidence in the data. For ABCC8 we have manually examined the other 84 (97-13) functions and searched for them in PubMed: we have been able to find 3 recently discovered functions that are reported in recent publications but that had not yet been inserted in iProClass. Unlike the well-known functions, the unknown functions did not have lots of redundant evidence, but had a small number of supporting evidence with high confidence score. Our goal is to enable the data integration system to also highly rank the *unknown*, but likely functions.

Contributions. We introduce BioRank, a system that integrates scientific data from various sources, takes into account the uncertainty of each result and ranks information by the level of combined evidence. We address the following important questions: (i) what ranking method is best suited in this setting, (ii) how robust are those rankings to errors in the input probabilities,

and (iii) how can rankings be computed efficiently. To answer (i), we examine two classes of ranking methods: one uses only deterministic information [27], s.a. the number of paths leading to an answer (e.g. GO functions found through several paths are ranked higher), and the other that explicitly takes into account a probabilistic score of the degree of uncertainty in the data. Our main finding is that the deterministic ranking methods are as good as, or slightly better than the best probabilistic ones (reliability and propagation) for ranking *well-known* functions. However, the probabilistic ranking methods are better for ranking yet *unknown* functions; since researchers are likely to use the system to find unknown functions, we believe that our finding offer proof of the main postulate in uncertain data: that uncertainties have to be quantified probabilistically. Question (ii) is especially important in our system because we have determined the input probabilities manually, after extensive consultation with experts. This raises the question of how robust these probabilities are, and for that we have examined the influence of perturbations in the input probabilities on the quality of final ranking results. Our finding is the rankings were very robust, which suggests that the probabilistic semantics is robust against slightly subjective and varying estimates by domain experts. To answer (iii), we propose three heuristics for speeding up the computation of a network reliability score, which is used in the main probabilistic ranking function.

Outline. Section 2 presents our formal model of uncertain information integration for exploratory search. Section 3 describes the three probabilistic and two deterministic ranking methods, as well as several optimization techniques for speeding up the ranking function. In Sect. 4 present several experiments designed to answer the three questions mentioned above. We discuss our main findings in Sect. 5, related work in Sect. 6 and conclude in Sect. 7.

2 A Model for Integrating Uncertain Data

In this section, we describe the data and the query models that allow biologists to explore multiple biological databases in the presence of uncertainties in the data. We describe here only from the mediated schema, since the rest of the integration system (mappings, wrappers, query translations, connection to sources) is based on our previous work [32, 39, 15, 40].

Schema integration. We use an Entity-Relationship (E/R) schema as the mediated schema. An entity set has a schema $P(\underline{id}, a_1, a_2, \ldots)$, where id is the key, and a relationship has a schema $Q(\underline{id, id'}, b_1, b_2, \ldots)$ where id, id' are foreign keys to two entity sets P, P' that Q relates. Here $a_1, a_2, \ldots, b_1, b_2, \ldots$ denote attributes. Every data source that we integrate exports one ore more entity sets. Examples of such entity sets are:

$$\text{EntrezProtein}(\textit{name}, \underline{\textit{seq}})$$
$$\text{EntrezGene}(\underline{id_{EG}}, \textit{StatusCode}, id_{GO})$$

Our system computes a number of relationships between the sources to achieve the actual integration, e.g. by following foreign keys, looking up aliases, or even matching keywords. When necessary, we translated ternary relationships to binary ones. For example, given a sequence *seq1*, NCBIBlast computes a set of similar sequences *seq2* together with an *e-value* for the similarity score of the two sequences. It also returns a foreign key into EntrezGene:

$$\text{NCBIBlast}(\textit{seq1}, \textit{seq2}, id_{EG}, \textit{e-value})$$

Here *seq1*, *seq2* are foreign keys in *EntrezProtein*, and id_{EG} is a foreign key to EntrezGene. In the

mediated schema we split this ternary relationship into two binary ones:

$$\text{NCBIBlast1}(\underline{\textit{seq1}, \textit{seq2}}, \textit{e-value})$$
$$\text{NCBIBlast2}(\underline{\textit{seq2}, \textit{id}_{EG}})$$

Our system currently connects to the following 11 data sources. #E and #R stand for the number of entities and relationships that they expose. Some relationships are not in the sources, but are instead computed during integration.

Source	*#E*	*#R*
AmiGO	1	4
NCBIBlast	2	3
CDD	3	1
EntrezGene	2	3
EntrezProtein	1	11
PDB	1	0
Pfam	2	2
PIRSF	2	2
UniProt	2	2
SuperFamily	3	1
TIGRFAM	2	2

Transforming uncertainties into probabilities. Our method allows to populate 4 probabilistic metrics. Let $P(\underline{id}, a_1, a_2, \ldots)$ be an entity set and $Q(\underline{id, id'}, b_1, b_2, \ldots)$ be a relationship. We define a probabilistic score p_s for the entire set P, and a probabilistic score q_s for the set Q. In addition, we define transformation functions $p_r(a_1, a_2, \ldots)$ and $q_r(b_1, b_2, \ldots)$ on the attributes of P and of Q, respectively, which return a probabilistic score for each record in P and Q.

	Sets	Records
Entity	$p_s \in [0,1]$	$p_r(a_1, a_2, \ldots) \in [0,1]$
Relationship	$q_s \in [0,1]$	$q_r(b_1, b_2, \ldots) \in [0,1]$

More detail on our four probabilistic metrics:

- First, p_s represents the degree of confidence in a data source as a whole. Biologists generally have more confidence in some sources than others. This can depend on their problem or the type of query. We provide p_s as a user-tunable parameter for every entity set. For example, our collaborators have evidence that results from PIRSF are more accurate than Pfam, two well known functional annotation databases.

- Analogously, q_s refers to the degree of confidence in a relationship as a whole. For instance, Pfam uses a sequence comparison algorithm which takes adjacency of amino acids into account. The BLAST algorithm, used by NCBIBlast does not. Algorithms like those in Pfam are believed to be more accurate in general.

- The p_r score indicates the degree of confidence in a single record and is computed from its attributes. Typically, there is one attribute that indicates the status of that entry, how it was curated, or some numerical attribute for the degree of confidence in that record. A transformation function converts the value of that attribute into a probability. E.g. for EntrezGene, the p_r score is computed from its *StatusCode* attribute; for AmiGO from the *EvidenceCode* attribute. Example evidence codes are IDA ("Inferred from Direct Assay", very reliable) or IEA ("Inferred from Electronic Annotation", less reliable).

EntrezGene

StatusCode	p_r
Reviewed	1.0
Validated	0.8
Provisional	0.7
Predicted	0.4
Model	0.3
Inferred	0.2

AmiGo

EvidenceCode	p_r
IDA / TAS	1.0
IGI / IMP / IPI	0.9
IEP / ISS / RCA	0.7
IC	0.6
NAS	0.5
IEA	0.3
ND / NR	0.2

- Similarly, the q_r metric is computed from the attributes of a relationship. Some cross-references use unique identifiers (foreign keys), and then we set $q_r = 1$. E.g. NCBIBlast has a foreign key id_{EG} to EntrezGene. Then we set $q_r = 1$. But NCBIBlast also returns an *e-value* measuring the likelihood that these 2 sequences are similar by chance. We translate this *e-value* into a probability as follows:

$$q_r = -\frac{1}{300}\log(\textit{e-value}) \ .$$

For BioRank, these probabilities were determined after extensive discussions with our collaborators from Seattle Children's Hospital Research Institute. This raises the question of how robust the system is to errors in the input probabilities: we address this in Sect. 4. Thus, our data model is a probabilistic database, where each schema component and each data record has a probabilistic score representing the confidence in that item.

Graph representation of the integrated data. Conceptually, we represent the entire integrated data as a *probabilistic entity graph*, which is defined as follow:

Definition 2.1 (Probabilistic Entity Graph)**.** *A probabilistic entity graph is a labeled, directed graph $G = (N, E, p, q)$, where N is the set of nodes, $E \subseteq N \times N$ the set of edges, and $p : N \to [0, 1]$ and $q : E \to [0, 1]$ are probability labels for each node and edge, respecitvely.*

The mapping from the E/R data to the probability entity graph is straightforward: data records become nodes, and relationships become edges. The node and edge probabilities $p(i)$, and $q(i, j)$ are defined:

$$p(i) = p_s(i) \cdot p_r(i)$$
$$q(i, j) = q_s(i, j) \cdot q_r(i, j)$$

where $p_s(i)$, $p_r(i)$ are the set and record probabilities, of the entity i, and similarly $q_s(i, j)$, $q_r(i, j)$ are the set and record probability of the relationship (i, j).

Exploratory Queries. The precursor of BioRank had a standard interface to the integrated schema, based on conjunctive queries [32]. However, biologists were not using such an interface effectively, because they needed the data integration system mostly for exploratory tasks, and not for data retrieval tasks. BioRank supports a simpler, yet more powerful class of queries, which we call *exploratory queries*. A user selects an input entity set P, one of its attributes *attr*, a value, and a set of output entity sets $P_1, \ldots, P_n$. The system retrieves all records in P whose attribute matches the value, then follows all links recursively to find all reachable records and returns those entities that are in $P_1, \ldots, P_n$. The result is a ranked answer set of records which can be reached from the query node. Note that ranking according to decreasing certainty is critical in exploratory queries: without ranking, users get flooded with irrelevant answers [28].

Definition 2.2 (Exploratory Query)**.** *An exploratory query is*

$$(P.attr = \text{“value”}, \{P_1, \ldots, P_n\}) \ .$$

The result to such a query is a ranked answer set with

$$A = \{v \mid \exists x \in P, \text{“value”} \in x.attr, x \to v, \exists i.v \in P_i\} \ ,$$

were $x \to v$ means that there exists a path from node x to node v in the entity graph and ranking is determined by some defined relevance function on the probabilistic query graph.

Definition 2.3 (Probabilistic Query Graph)**.** *A probabilistic query graph is a probabilistic entity graph $G = (N, E, p, q, s, A)$ where $s \in N$ represents the query and $A \subset N$ the answer set to the query.*

Definition 2.4 (Relevance Function)**.** *Assume a probabilistic query graph $G = (N, E, p, q, s, A)$. A relevance function $r : A \to R$ assigns each node in the answer set $t \in A$ a relevance score $r \in R$, where R stands for the range. This relevance function imposes a partial order on the answer set which can be used for ranking.*

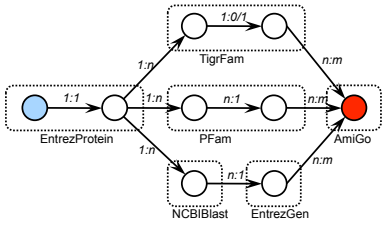


Figure 1: The subset of the E/R schema relevant to the exploratory query (EntrezProtein.name = “ABCC8”, AmiGO).

Motivating example (continued). We illustrate the working of our system within the context of our example from the introduction and Fig. 1. The system creates a new node, called *query node* (blue node on the left), then links to all records in EntrezProtein that match the keyword ABCC8. From here the system continues in three separate sources, searching for all possible paths to some record in AmiGo (the red node on the righ). Note that the figure illustrates the graph only at the schema level: the real graph has 684 nodes, 977 edges, and the answer set consists of 97 nodes (all represented by the single red node). This is a typical case for all exploratory queries: the system will return many incorrect answers because of uncertainty and imprecision in the data. Therefore, BioRank evaluates the level of certainty in each function to assigns a *relevance scores r* to each function and presents the user with a ranked list of protein functions:

#	*Function (abbr.)*	*GO term*	*r score*
1	sulphonylurea receptor activity	GO:0008281	0.7000
2	potassium ion conductance	GO:0006813	0.7000
3	Interacting selectively with ATP	GO:0005524	0.6999
4	cytoplasmic membrane	GO:0005886	0.6996
5	small-molecule carrier or transporter	GO:0005215	0.6977
...	...	...	...

3 Methods for Ranking Integrated Data

Previous work [27] on ranking integrated biological data proposed to use simple graph metrics like link cardinality and length of paths from a query to a target node to rank connected information in a data integration system. In contrast, we next introduce 3 probabilistic ranking methods that explicitly take into account the uncertainties at each integration step and give efficient methods to evaluate such queries. As benchmark, we also define 2 non-probabilistic ranking methods which we then text against each other in the later experimental section.

3.1 Reliability

Our first semantics calculates relevance scores by interpreting the query graph as a *network reliability problem* [11]. More specifically, we use the generalized source-target reliability problem with node failures that can be reduced to the standard network reliability problem by removing node failures and reifying the graph [11, Sect. 1.4]). Given a probabilistic query graph $G = (N, E, p, q, s, A)$ where $p : N \to [0, 1]$ and $q : E \to [0, 1]$ are the probabilities $p(n)$ and $q(e)$ that the node or edge are present[1], and $s \in N$ and $A \subset N$ are the source node and the answer set, respectively. For each target node $t \in A$, the *reliability score* $r(t)$ is then the probability that t is connected to s and active. We use these relevance scores to rank the nodes in the answer set.

Our rationale for the reliability semantics is that it is equivalent to the *possible worlds* semantics in probabilistic databases [13]: each subgraph of the network graph is a *world*, and the reliability score of an answer is the probability over all worlds that the answer is reachable from the query node. However, query evaluation is difficult under this semantics. The network reliability problem is #P-hard in general [43]. While it has a long history in Computer Science [11], most of the earlier algorithms are based on some form of state enumeration. A polynomial time approximation algorithm was given recently [23], which however relies on enumerating cut sets, and therefore does not work well in our setting (multiple parallel paths with a common source and target have exponentially many cut sets). We, therefore, developed three techniques to make query evaluation tractable in our setting.

1. Monte Carlo simulations. A naive Monte Carlo simulation computes the source-target network reliabilities for all target nodes $t \in A$ by repeating the following step: randomly choose a subgraph by including each node $i \in N$ with probability $p(i)$ and each edge $e \in E$ with probability $q(e)$; check if there exists a path from s to each t in the random subgraph and t is present. If m out of n trials resulted in subgraphs with nodes s and t connected, then $r(t)$ is approximated by $\hat{r}(t) = m/n$. Such Monte Carlo methods have been previously proposed [24], and applied e.g. for the purpose of inferring protein complex membership [3].

In practice, we implemented an improved version as shown in Algorithm 3.1. The key observation for the improvement is that it is not necessary to simulate every node and every edge in each trial. After simulating a subset of the nodes and/or edges we will have already chosen some to exclude. Such exclusions may leave entire subgraphs disconnected from the source node. We need not simulate further, for this trial, any nodes or edges in the disconnected portion. Our resulting idea is to perform a depth first search from the source node s. For a node x to be reachable, there must be a contingent path from s to x. If no such path exists in a trial, we do not visit x nor its downstream subgraph, unless it is reachable by another path. In this manner we don't simulate any nodes or edges only to later discover that they are disconnected. Whereas such improvements do

[1]The traditional network reliability problem uses the complement, i.e. the probability that the edge *fails*.

not change the theoretical complexity of the algorithm, they yield significant runtime performance gains, the extent of which is determined by the characteristics of the underlying graph. The greatest benefit occurs when (i) the node and/or edge weights are very low (i.e. we exclude them often), particularly if these low weights occur close to the source node, (ii) when paths are long (i.e. greater chance of excluding a node or edge somewhere along a path), and (iii) there are fewer converging paths (i.e. less chance that a node is reachable via some other path). In the experimental section, we show an average speed-up of factor 3.4 (-70%) on our test set.

Algorithm 3.1: Reliability Traversal Monte Carlo Simulation

Input : Probabilistic query graph $G = (N, E, p, q, s)$, #Iterations n
Output: Vector $\hat{\mathbf{r}}$ of approximate relevance scores $\hat{r}(x), \forall x \in N$

$\forall x \in N : x.lastSim \leftarrow 0 \wedge x.reachCount \leftarrow 0$
for iteration $t = 1$ to n **do**
  `Traverse`(G, s, t)
forall nodes $x \in N$ **do**
  $\hat{r}(x) \leftarrow x.reachCount/n$
return $\hat{\mathbf{r}}$

`Traverse`(G, x, t):
if $x.lastSim = t$ **then**
  return
$x.lastSim \leftarrow t$
if `Random`$([0, 1]) \leq p(x)$ **then**
  increment $x.reachCount$
  forall edges $(x, y) \in x.outEdges$ **do**
    if `Random`$([0, 1]) \leq q(x, y)$ **then**
      `Traverse`(G, y, t)

We use bounds for estimating the necessary number of trials under assumption of independent coin flips. We prove the following theorem in Appendix A and test those bounds in the experimental section.

Theorem 3.1. *Assume the scores of two nodes i and j are $r(i)$ and $r(j)$, with $r(i) = r(j)+\varepsilon$ ($\varepsilon > 0$). Running n independent random trials for each node suffices to guarantee that the simulated scores are not incorrectly ranked with probability at least $1 - \delta$, where*

$$n \geq \frac{(1+\varepsilon)^3}{\varepsilon^2(1+\frac{\varepsilon}{3})} \ln\left(\frac{1}{\delta}\right) .$$

The motivation for this bound is that the predicted relevance scores $\hat{r}(i)$ and $\hat{r}(j)$ can be incorrectly ranked when ε and n are small. If our ranking semantics is a good one, very close ties between relevant and non-relevant nodes should happen infrequently, and if they do, then we do not have enough evidence to distinguish them. Choosing a 95% confidence and separable difference between two scores $\varepsilon = 0.02$, we learn that 10,000 trials should be enough. We will test these bounds in the experimental section.

2. Graph Reductions. Consider the following graph transformation rules:

- *Delete inaccessible nodes.* Remove a sink node x (i.e. without outgoing edges), if x is not a target node.
- *Collapse serial paths.* If x has a single incoming edge (y, x) and a single outgoing edge (x, z), remove x and add an edge (y, z) with $q(y, z) = q(y, x) \cdot p(x) \cdot q(x, z)$.
- *Collapse parallel paths.* If there are two or more edges $e_1, e_2, \ldots$ from x to y (e.g. introduced by a serial reduction) then replace them with one edge with probability $1 - \prod_i (1 - q(e_i))$.

These transformations can be applied repeatedly until no rule changes the graph anymore. In the best case, the graph then consists of single edges from the source s to each target node n_i and the reliability is $r(i) = q(s, i) \cdot p(i)$. But that is usually not the case. Such, or similar transformations have been considered for a long time in network reliability [11, Ch. 2.2]. We apply them repeatedly to any graph to reduce it before running the more expensive Monte Carlo simulation algorithm. In general these transformations are known not to reduce significantly the complexity of the reliability problem: they get stuck, for example, on the Wheatstone Bridge graph (Fig. 2c). However, we found them to work surprisingly well in scientific workflow-type data integration with uncertainties (-78% in edges and nodes in our experiments), because the query graphs of *scientific workflows* [14] have a certain regularity.

3. Tractable closed solution. We found that the above transformation rules often allow us to calculate reliability in a tractable closed form by applying them not to the whole graph, but individually to each subgraph connecting the source and each target node. To discern when such reductions are possible, we have to study the possible E/R schemas of graphs. If Q is a relationship in the E/R schema, we denote its type with [1:n], or [n:1], or [m:n]. We are interested in characterizing an E/R Schema such that the following holds: for any data graph instance of that schema, the graph transformation rules will completely reduce the graph. We call such schemas *reducible*.

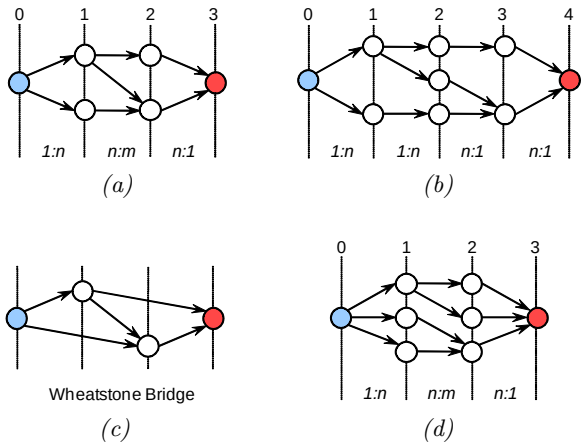


Figure 2: The concatenations in the probabilistic source graph partly determine the query complexity of our probabilistic entity graph.

In general, [n:m] relations lead to irreducible graphs (Fig. 2a), because instances of such schemas may contain Wheatstone bridges (Fig. 2c). Even if all relations [1:n] or [n:1], the graph may still be irreducible (Fig. 2b), and some [n:m] can actually be reduced (Fig. 2d). To check if an E/R schema

is reducible we need to examine compositions of relationships: $Q \circ Q'$. Note that $[1{:}n] \circ [1{:}n] = [1{:}n]$ and $[n{:}1] \circ [n{:}1] = [n{:}1]$. In general $[1{:}n] \circ [n{:}1]$ can be either of $[m{:}n]$, $[n{:}1]$, or $[1{:}n]$ (we include [1:1] into one of the latter two), but with domain knowledge we can often determine the type of the composed relationship. Then:

Theorem 3.2. *Let S be an E/R schema.*

A) If S is a tree consisting only of [1:n] relationships, then the schema S is reducible.

B) Suppose that there is an entity set P that has exactly one incoming relationship Q of type $[1{:}n]$ and exactly one outgoing relationship Q' of type $[n{:}1]$. If:

a) the composition $Q \circ Q'$ is either $[1{:}n]$ or $[n{:}1]$ (but not $[m{:}n]$), and

b) the schema S' obtained by removing P and replacing Q and Q' with $Q \circ Q'$ is reducible.

Then the schema S is reducible.

They key insight of the theorem is the order in which one has to compose the relationships. Two relationships of types $[1{:}n]$ and $[1{:}n]$ can be composed at the schema level and result in a $[1{:}n]$ relationship, but at the data level no *serial-path* reduction is possible in general. By contrast, given a sequence of two relationships $[1{:}n]$ and $[n{:}1]$, all intermediate nodes can be removed using the *serial-path* rule, followed by the *parallel-path* rule to remove duplicate edges (Fig. 3). We prove Theorem 3.2 in Appendix B.

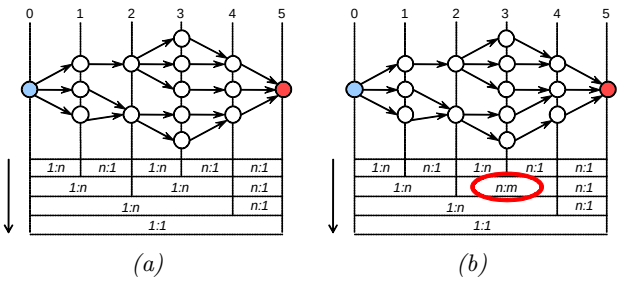


Figure 3: Illustration for Theorem 3.2. The schema on the left is reducible, because there is a sequence of composing the innermost $[1{:}n]$ and $[n{:}1]$ relationships s.t. all intermediate relationships are either $[1{:}n]$ or $[n{:}1]$. The schema on the right is not reducible, because the first composition results in an $[m{:}n]$ relationship.

3.2 Propagation

The motivation for this model is to break the complexity of network reliability and introduce a 'local' view on calculating relevance scores. In this model, relevance 'propagates' along the edges from the query node to all other nodes, similarly to PageRank [9]. In this semantics, each node $y \in N$ has a *propagation score* $r(y)$ that depends only on its parents and that ignore mutual dependencies or correlations between parents:

$$r(y) \quad = \quad \left(1 - \prod_{(x,y)\in E} \Big(1 - r(x) \cdot q(x,y)\Big)\right) \cdot p(y)$$

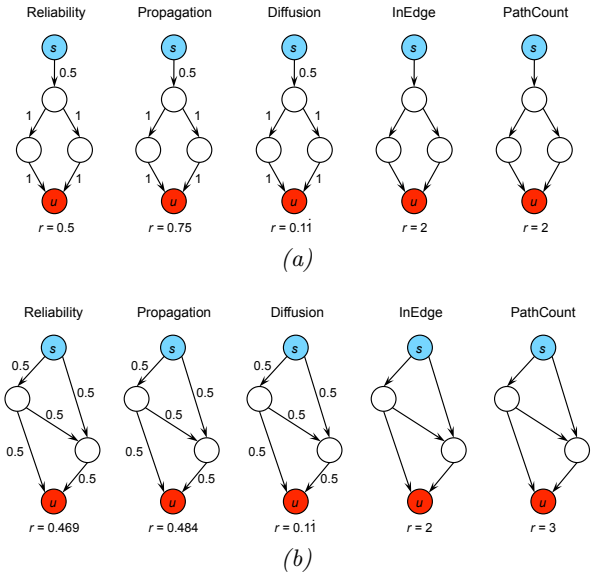


Figure 4: Illustration of the 5 relevance scoring functions on a serial-parallel graph (a) and a Wheatstone bridge (b).

This semantics also admits a probabilistic interpretation, but here the score of each node or edge depends only on its local neighborhood. In fact, the following proposition is easy to verify:

Proposition 3.1. *Let $G = (N, E, q, s)$ be a graph with probabilities on edges. If the graph is a tree with root s, then for every node x the reliability and the propagation scores coincide.*

When the graph is not a tree, however, then reliability and propagation differ. We show with the help of Fig. 4a. There are two paths from source to target, each with probability $0.5 \cdot 1 \cdot 1$. The reliability score is 0.5, because it accounts for the fact that the 0.5 edge is common to both paths. The propagation score is 0.75 because it considers the two paths independent. In general, the propagation scores will always be bigger or equal to reliability scores.

The main advantage of propagation is that it is efficient to compute. We compute it using the straightforward iterative Algorithm 3.2. In contrast to the Monte carlo simulation, this iterative approximation converges faster. For DAGs like in our scientific workflow scenarios, this algorithm actually reaches equilibrium already after the maximum pathlength. The disadvantage is that propagation considers all paths leading to an answer node as independent, as illustrated in Fig. 4. This can lead to artificially high scores for certain graph topologies. A particularly troublesome case is when the graph has a cycle: in this case the propagation semantics unfolds the cycle into an infinite sequences of independent paths, and boosts considerably (and incorrectly) the scores for some answer nodes.

3.3 Diffusion

In this model, we keep the locality aspect, but change the semantics of probabilistic evidence accumulation. Its main intuition is that the relevance $r(x)$ of a node x is allowed to "flow" to a neighbor y only if $r(x) > \bar{r}(y)$, where $\bar{r}(y)$ is a new intermediate variable used for the incoming diffusion along all edges into node y. Hence, in this model, diffusion along an edge depends on the

Algorithm 3.2: Relevance Propagation

Input : Probabilistic query graph (N, E, p, q, s), #Iterations n
Output: Vector $\mathbf{r}$ of relevance scores $r(x), \forall x \in N$

1: $r(s) \leftarrow 1; \forall x \in N \setminus \{s\} : r(x) \leftarrow 0$
2: **for** iteration $t = 1$ to n **do**
3: **forall** nodes $y \in N \setminus \{s\}$ **do**
4: $r^*(y) \leftarrow \Big(1 - \prod_{(x,y)\in E} \big(1 - r(x) \cdot q(x,y)\big)\Big) \cdot p(y)$
5: $\mathbf{r} \leftarrow \mathbf{r}^*$
6: **return r**

relatedness scores at both ends. Also, in this model, the total incoming diffusion at a node is we use addition, not inverse multiplication. Formally, the relevance $r(x)$ of a node x in the diffusion semantics is defined by the following recursive equations:

$$\bar{r}(y) = \sum_{(x,y)\in E} \max\big[\big(r(x) - \bar{r}(y)\big) \cdot q(x,y), 0\big]$$
$$r(y) = \bar{r}(y) \cdot p(y)$$

An interesting aspect of the diffusion semantics is that it tends to favor nodes that have fewer stronger paths over nodes that with more but weaker paths, which should help discover less-known information. However, the influence of the shortest pathlength between query and target node is far stronger than in reliability and propagation. Also, the requirement of introducing a direction into diffusion (which is, in nature, an undirected, dissipative process) makes this semantics less interpretable and the quality of ranking a priori unpredictable, a priori. In addition, calculation is more intricate than propagation. Diffusion defines recursively two quantities $\bar{r}(x)$ and $r(x)$, which requires us to split the main iteration into two loops. The outer loop computes new values $r^*(x)$ from the results of the previous iteration $r(x)$. The inner loop, shown as function `solve` in Algorithm 3.3, computes the value $\bar{r}(x)$. As $\bar{r}(x)$ is defined implicitly, we iterate until we find a good approximation. This process is guaranteed to converge, however it requires $O(nm)$ iterations, where m is the number of iterations in the inner loop; by contrast the propagation algorithm requires $O(n)$ iterations. In practice, however, diffusion is only moderately slower to evaluate than propagation overall as the inner recursive loop is not evaluated in each step.

3.4 Incoming Edges

The simple topological measure *InEdge*, proposed in [27] as "cardinality", assigns the total number of incoming edges to each target node as its relevance score. The big advantage is that it is very fast to calculate. The disadvantages are: (i) it completely ignores the evidence supporting each entity or relation in the query graph; (ii) it only considers the part of the query graph adjacent to a target node; and (iii) as InEdge relevance scores are natural numbers instead of real numbers, we expect more ties between nodes and, therefore, less discrimination between nodes with little evidence.

Algorithm 3.3: Relevance Diffusion

Input : Probabilistic query graph (N, E, p, q, s), #Iterations n
Output: Vector $\mathbf{r}$ of relevance scores $r(x), \forall x \in N$

$r(s) \leftarrow 1; \forall x \in N \backslash \{s\} : r(x) \leftarrow 0$
for iteration $t = 1$ to n **do**
  forall nodes $y \in N \backslash \{s\}$ **do**
    $\texttt{solve}\Big(\bar{r}(y) = \sum_{(x,y)\in E} \texttt{max}\big[\big(r(x) - \bar{r}(y)\big) \cdot q(x,y), 0\big] \Big)$
    $r^*(y) \leftarrow \bar{r}(y) \cdot p(y)$
  $\mathbf{r} \leftarrow \mathbf{r}^*$
return r

3.5 Counting Paths

For *PathCount*, we count the total number of paths that connect the query and a target node. Our rationale is that, in contrast to InEdge, PathCount measures connectivity in the whole subgraph between query and target node (compare Fig. 4). A possible disadvantage of PathCount is that it can only work on DAGs and, therefore, workflow-type graphs. Cycles lead to infinite PathCounts.

4 Experiments

In this section, we evaluate our framework in the context of an important problem in molecular biology: *functional protein annotation*, which is determining the complete set of functions for proteins [37]. Determining the function of a single protein in isolation is a very difficult biological problem. However, much in biology can be learned from already known information by using similarity functions. Our collaborators at Seattle Children's Hospital Research Institute (SCHRI) have surmised that integrating similarity functions from multiple sources has the potential to increase the accuracy of function predictions [29]. This rationale underlines our information integration approach. We are especially interested in comparing the performance of our ranking methods under decreasing levels of certainty in the data, as the aim of BioRank is to allow scientists to discover previously not known information. For this purpose, we created different scenarios with varying degrees of difficulty in finding the information. This required considerable experimental effort on our part.

Measuring Ranking Performance. BioRank assigns a relevance score to all functions connected to the query protein and ranks them. The output to the user is an ordered list of predicted functions for a query protein. To compare ranking performance across different scenarios, we use the measure *average precision* (AP) as uniform method to assess ranking quality. AP is a standard evaluation metric in information retrieval [5] and rewards rankings that place relevant items earlier in rankings. Our biological rationale is that users prefer that the system suggest which functions have the strongest evidence. Given an ordered list of n items of which k are relevant. Let rel_i denote the relevance of the item at rank i, which is 1 if it is relevant according to a predefined golden standard or 0 otherwise. Further write P@i for the precision or the ratio of retrieved and relevant items over the total retrieved at item i. Representing the ranking by a binary vector $\mathbf{rel} = (rel_i)$, we can write

$$\mathrm{AP}(\mathbf{rel}) = \frac{\sum_{i=1}^{n} \mathrm{P@}i \cdot rel_i}{k} \ .$$

AP can be calculated at a specified number of results (e.g. AP@20), at a certain percentage of recall (e.g. AP@80%), or at predefined recall points (e.g. 11-point interpolated average precision). We calculate AP at 100% recall for two reasons: (i) we want to evaluate ranking on the complete set of functions for each protein and (ii) the total number of functions returned varies considerably for different queries. Averaging over all retrieved relevant functions allows a consistent performance comparison across different scenarios.

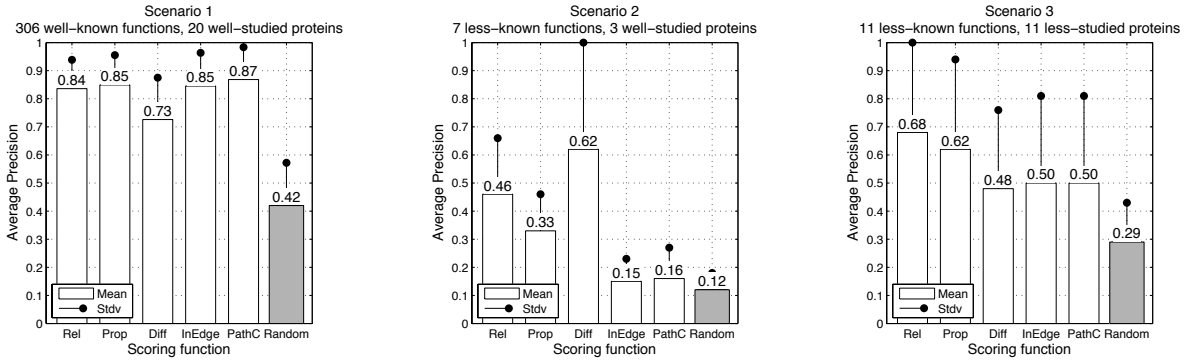
Scoring functions sometimes lead to ties between functions and, therefore, only partial orderings in the result list. We use the analytic method recently proposed in [31] to calculate AP in the presence of ties. The idea behind the method is to calculate the mean AP over all possible permutations. We refer to their paper for details of the method. The alternative to ranking is presenting the list of n results in arbitrary order. We, therefore, use the expected average precision of an arbitrarily ranked result list as benchmark. The following equation can be derived analytically following ideas proposed in [31].

Definition 4.1 (Random average precision)**.** AP_{rand} *is the expected* AP *when a list of* k *relevant and* n *total items is randomly sorted. It is given by*

$$\mathrm{AP}_{rand}(k,n) = \sum_{i=1}^{n} \frac{(k-1)(i-1)+(n-1)}{i(n-1)n}$$

Scenario 1: Well-known functions for well-studied proteins. In our first experimental setup, we chose 20 well-studied proteins from the iProClass database [47] to serve as a test set for our function assignments. These were chosen because they serve as a reference standard in that they have highly reliable experimental evidence for their functions. Experimental function assignment is more reliable (and expensive) than computational prediction, which is the most common method. Over 46% of functions in our test set are experimental versus 5% in general [42]. The biological data sources which provide function predictions which were considered for this study were Pfam [18], TIGRFAM [19], BLAST at NCBI [48], and Entrez [30]. The iProClass database was not considered because it was the source of the test set. The others were chosen because of their familiarity to biologists and their use of Gene Ontology (GO) terms, a shared vocabulary for protein functions [4]. The databases were accessed in June 2007. Table 1 shows the 20 proteins, the number of functions returned by iProClass, the number of functions returned in BioRank's anwer set, and their relative ratio. Fig. 5a compares the performance of our 5 ranking approaches on these 20 proteins. All 5 ranking approaches perform significantly better than randomly sorting functions. The two deterministic ones, InEdge and PathCount, perform slightly better than reliability and propagation. Diffusion performs worst.

Scenario 2: Less-known functions for well-studied proteins. All previous 306 functions in scenario 1 are contained in iProClass and, as such, are *well-known*. For scenario 2, we tried to find new, yet *unknown* GO functions for any of the 20 iProClass proteins which were not contained in iProClass at the time we accessed the database (June 2007). Newly discovered protein functions are usually first described in publications before they are entered into curated databases such as iProClass and become, de facto well-known. We manually searched through PubMed for recently published papers that described any functions for the 20 proteins that were not already contained in iProClass. We found 7 new functions for 3 of the 20 proteins. Table 2 lists those proteins, their newly found functions together with the PubMed ID of the publication describing this function. It also contrasts their respective ranks for our 5 ranking methods. Ties are marked by intervals. E.g., InEdge and PathCount rank GO:0006855 between 35 and 97 out of 97 possible ranks, as



(a) *(b)* *(c)*

Figure 5: Scenario 1: InEdge and PathCount perform slightly better than probabilistic rankings (a). Scenario 2: Probabilistic rankings perform better than InEdge and PathCount on less-known functions for well-studied genes (b). Scenario 3: Reliability and propagation perform best on less studied genes (c).

Table 1: Scenario 1: 20 chosen golden standard proteins

Protein	*# iProClass Functions*	*# BioRank Functions*	*%*
ABCC8	13	97	13%
ABCD1	15	79	19%
AGPAT2	10	16	63%
ATP1A2	31	108	29%
ATP7A	35	130	27%
CFTR	19	90	20%
CNTS	8	15	53%
DARE	18	39	46%
EIF2B1	15	35	43%
EYA1	12	38	32%
FGFR3	16	65	25%
GALT	8	15	53%
GCH1	10	21	48%
GLDC	7	17	41%
GNE	13	24	54%
LPL	13	36	36%
MLH1	19	52	37%
MUTL	13	28	46%
RYR2	18	66	27%
SLC17A5	13	66	20%
Sum	306	1036	37%

only one single, but strong path connects that function to the protein. The result is that the deterministic measures do not significantly rank those new functions better than random, whereas all 3 probabilistic ranking methods successfully differentiate those functions. Fig. 5b lists AP just on the 3 genes and just for the newly found functions for contrast with scenario 1.

Scenario 3: Unknown functions for less-studied proteins. This scenario models a real-world problem in biology: assigning function to proteins of unknown function, otherwise known as *hypothetical proteins*. For this scenario, and to evaluate the various ranking functions, we created a reference set of hypothetical proteins for which functions were determined independently by biological experts. The method used to assign function is an innovative approach developed with

Table 2: Scenario 2: Additional 7 functions for 3 proteins

Protein	*Function*	*PubMedID (year)*	*Method & assigned rank*					
			Rel	*Prop*	*Diff*	*InEdge*	*PathC*	*Random*
ABCC8	GO:0006855	18025464 (2007)	21-22	24-25	1-3	34-97	35-97	1-97
	GO:0015559	18025464 (2007)	21-22	24-25	1-3	34-97	35-97	1-97
	GO:0042493	18025464 (2007)	17	20	1-3	22-33	25-34	1-97
Cftr	GO:0030321	17869070 (2007)	1-2	8	1-2	17-31	22-23	1-90
	GO:0042493	18045536 (2007)	24	24	16-18	17-31	22-23	1-90
EYA1	GO:0007501	17637804 (2007)	4	10	15	32-38	27	1-38
	GO:0042472	17637804 (2007)	14	6	6	13-16	18	1-38
		Mean	14.8	16.7	6.5	36.6	35.9	39.6
		Stdv	8.5	8.1	6.5	26.9	26.5	44.2

our collaborators at SCHRI. This method allows assignment of function to hypothetical proteins by comparing them to experimentally characterized functions manually and following carefully designed protocols. The approach is described in more detail in [29]. However, this procedure is very costly in terms of human effort. Therefore our reference set for less-studied proteins is very small with only 11 proteins. Table 3 lists the 11 bacterial proteins, their functions (generally only one in bacteria), and the respective ranks assigned by our 5 ranking functions. Fig. 5c shows these results in terms of average precision. Clearly, reliability and propagation perform better than deterministic rankings.

Table 3: Scenario 3: 11 hypothetical proteins and their functions

Protein	*Function*	*Method & assigned rank*					
		Rel	*Prop*	*Diff*	*InEdge*	*PathC*	*Random*
DP0843	GO:0003973	5	5	3-20	4-5	4-5	1-47
DP1954	GO:0019175	2	2	1-2	2-3	2-3	1-18
NMC0498	GO:0016226	1	1	1-2	1-3	1-3	1-5
NMC1442	GO:0050518	1	1	1-2	1	1	1-17
NMC1815	GO:0019143	1-2	1-2	1-2	1-3	1-3	1-14
SO_0025	GO:0004729	4-5	4-5	4-5	4-5	4-5	1-5
SO_0599	GO:0005524	1-10	1-10	1-10	4-19	4-19	1-19
SO_0828	GO:0008990	1	1	4	1-4	1-4	1-4
SO_0887	GO:0047632	1-2	1-2	4-6	4-6	4-6	1-6
SO_1523	GO:0003951	1-2	1-2	1-2	1-2	1-2	1-24
WGLp528	GO:0004017	1	3	3-4	1	1	1-9
	Mean	2.3	2.5	3.8	3.5	3.5	15.3
	Stdv	2.2	2.2	4.2	3.8	3.8	12.5

Sensitivity analysis. In Sect. 2, we described the translation of uncertainties in the data into probabilities. The remaining question is how we can make sure that we use correct probability estimates. One way is machine learning, but this has a high cost due to hard-to get training data and possible lack of generality due to overfitting. [17]. In contrast, we use biological experts to estimate the *default parameters* for BioRank [28]. This is easier but problematic in that human estimates can be imprecise [41]. But how exactly are those parameters determined, and how do we know they are correct? We address these concerns with a reformulated question: how robust are the predictions of BioRank to variations in its parameter estimates? If the average precision of the system is good under default parameters (i.e. ranks correct functions highly) and under systematic variations, then the performance of the approach is good as well as stable (i.e. robust) and the

estimation method sufficient.

To answer this question, we perform a sensitivity analysis of the quality of ranking results with respect to changes in the input probabilities. Specifically, we compare AP in our 3 scenarios under default parameters and under systematic variations. For that purpose, we use a multi-way sensitivity analysis approach which perturbs all parameters simultaneously and is representative of our situation where all parameters may be imprecise. For the systematic variations, we use a method proposed by [20]: normally distributed random noise is added to a log-odds probability then converted back to a probability. This approach avoids the need for range checks and enables control over amount of noise added. Given the original probability $p \in (0, 1)$, a chosen standard deviation σ, a function Lo that converts a probability to its log-odds, and Lo^{-1} its inverse, we create a randomly perturbed probability p' as

$$p' = \mathrm{Lo}^{-1}\big(\mathrm{Lo}(p) + e\big) \ ,$$

where $e = \mathrm{Normal}(0, \sigma)$ is random Gaussian noise. Figure 6 shows the effects on AP for each of the

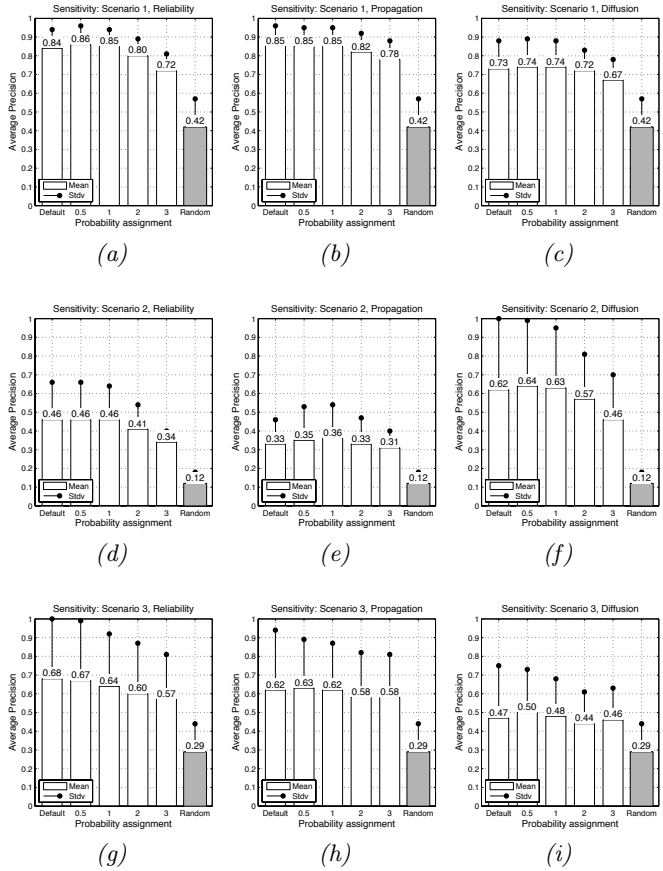


Figure 6: Effect of random perturbations of the input probabilities on the 3 probabilistic ranking methods. Performance remains robust over a wide range.

3 probabilistic ranking methods and for each of the 3 scenarios after perturbing the probabilistic default parameters of UII for nodes (p) and edges (q) with noise at $\sigma = 0.5, 1, 2, 3$ and averaging

over $m = 100$ repeated experiments. Confidence intervals (95%) were very narrow (0.001 to 0.022 over all noise perturbations), indicating a sufficient number of simulations.

The interesting result is that the quality of ranking does not significantly decrease before adding 3 standard deviations of noise and is still higher than the deterministic alternatives for less known information. In some instances, AP even increases with slight noise. Therefore, a slightly different choice of default probabilities could have improved our rankings, but overall, the probabilistic ranking methods are very robust in their quality against slightly subjective and varying estimates of default probabilities by domain experts. Our results are consistent with observations made in artificial intelligence that probabilistic belief networks often show a similar robustness to imprecise input probabilities [35].

Efficiency of query evaluation. We evaluate the time requirements of our 5 different ranking approaches on the 20 query graphs for for scenarios 1 and 2 as those are the largest. We treat reliability in more depth as it is the one that follows our desired possible world semantics and that turned out to be the most predictive for less-known information.

(1) *Closed solution*: Theorem 3.2 developed an indicator when reliability rankings can be calculated efficiently in closed form. Applying the theorem to our setup (Fig. 1), we see: (i) the total graph is not reducible due to the last $[n{:}m]$ relation; (ii) the individual queries, however, can be solved in a closed solution. The reason is that the last $[n:m]$ relationship becomes $[n:1]$ from the point of view of each node in the answer set. And the theory predicts that the queries can be reduced by just evaluating the subtrees to each answer node sequentially by themselves. Our theory proves to be right and useful.

(2) *Monte Carlo*: We previously estimated that 10000 runs of Monte Carlo simulations should be enough to achieve the correct reliability rankings with high confidence. To verify that, we repeated Monte Carlo simulations ($m = 100$) for varying number of simulations ($n = 1, 3, 10, \ldots, 10000$) and calculated mean and standard deviation. Figure 4 shows that already 1000 trials achieve high average accuracy.

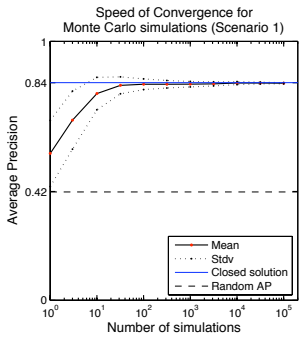
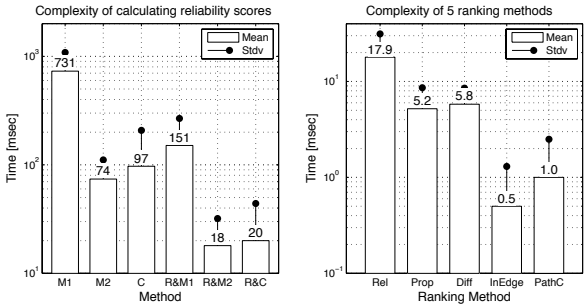


Figure 7: Effect of number of trials in the Monte Carlo algorithm on reliability ranking performance. 1000 trials already deliver very reliable results.

(3) *Graph reductions*: Our 20 original graphs have on average 520 nodes and 695 edges. Applying our graph reductions, we can reduce their number on average by 78%. Those reductions can help speed up both, the closed solution and the Monte Carlo calculation.

(4) *Results*: Fig. 8a compares the different approaches for calculating reliability. M1, M2 stand for Monte Carlo simulations on the original graphs with 10000 and 1000 trials, respecively, and C for the closed solution. For each of these 3 approaches, we contrast with the time it takes for first

performing graph reductions (R&). In order to account for the effects of disk latency, we first loaded the vector of ranks and scores for our example queries into main memory. The result is that the graph reduction together with 1000 Monte Carlo trials is actually the fastest method for calculating reliability, even faster than the closed solution. To put that result in context, we compared our Monte Carlo implementation to the naive method: we got an average speed-up of 3.4 (-70%) for our Monte Carlo method and 13.4 (-93%) for reduction and Monte Carlo. Given that Monte Carlo can be run on all graphs, even such where no efficient closed solution exists, we use that method as our benchmark. Fig. 8b then contrasts this method with the less expensive alternative semantics. It is well-known that probabilistic approaches are more expensive than deterministic ones. The fact that our system can evaluate probabilistic queries of a possible world semantics, however, in considerably less than a 100 msec and within 1-2 orders of magnitude of deterministic ones, is a positive result.



(a) *(b)*

Figure 8: (M1: Monte Carlo 10000 simulations, M2: Monte Carlo 1000 simulations, R: Reduction of graph, C: Closed solution)

5 Discussion

The value of retaining probabilities. Our experiments showed a good ranking performance of the non-probabilistic redundancy measures, InEdge and PathCount, for ranking *well-known* information. The intuitive explanation for that result is that, commonly, many different ways lead to the same well-known conclusion, and simply counting the number of such different paths seems a good approach to rank information (Fig. 9a). In contrast, we built BioRank to provide scientists a means to discover *less-known* or less-disseminated information. Finding new functions for proteins is an arms race against time for biologists. Intuitively, redundancy cannot be used for ranking in that scenario. Instead, the strength of each piece of evidence has to be taken into account (Fig. 9b).

Indeed, our experiments showed that ranking methods that take into account uncertainties perform increasingly better for less-known information. Providing examples of such less-known information bascially required us standing at the forefront of scientific discovery and having access to information which is not yet widely disseminated. In scenario 2, we were able to assign new functions to several proteins, which are extensively studied and curated by multiple organizations (PIR, GeneTests, etc.). Given that, discovering 7 new functions is an accomplishment. We say "new" functions as database records for these genes do not list these functions at this time. In scenario 3, we were able to better predict functions for hypothetical proteins, whose proteins are

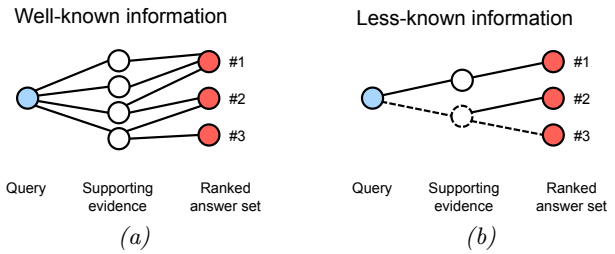


Figure 9: Ranking with simple redundancy measures is successful for well-known information *(a). For* less-known information*, the uncertainty of each data item and each connection (shown by dotted line) has to be considered for discriminating more from less relevant information (b).*

not yet studied.

Our main conclusion is that ignoring both (i) probabilities and (ii) dependencies between data sources works fine for well-known information (e.g. highly ranked in search results). However, for yet not widely known information, retaining probabilities and taking dependencies explicitly into account allows to better predict relevant scientific information (Fig. 10).

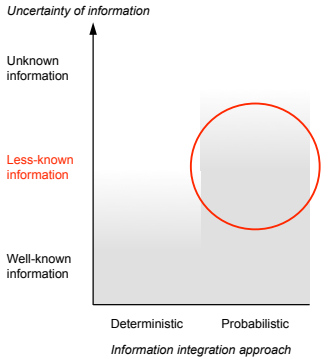


Figure 10: The value of uncertain information integration: Probabilistic approaches perform about as well as deterministic ones on well-known *information. However, they outperform them on* less-known *information. This makes them better suited for discovery-based scientific applications such as exploratory search.*

The semantics of accumulating probabilistic evidence. We conceived of the diffusion method to find out if the method by which probabilistic evidence is accumulated affects the results, or if the mere fact of probabilistically adding evidence suffices. For this purpose, we created the diffusion method in which, at each node in the graph, probabilities are aggregated in a additive instead of a standard multiplicative fashion. This change of semantics, however, comes at the cost of being highly dependent of path length. In our experiments, diffusion turns out to be worse than all other measures and highly unpredictable. In the one case that diffusion works considerably better than all other scoring functions (new functions for ABCC8 in Table 2), the single but strong evidence comes from one database which has a shorter connection to the query node. For its overall bad performance, we do not suggest using the diffusion semantics for uncertain information integration and take the results as evidence that those changes in probabilistic semantics actually affect the outcome considerably and the standard possible world semantics better models reality.

Divergent and non-workflow schemas. Our experiments were performed on *convergent*, scientific workflow graphs [14]. Those are characterized by (i) directed edges, and (ii) alternative paths leading to the same information. We have also conceived of scenarios where entries from different databases cannot be linked together and the result is a *divergent* star schema. This might arise when common vocabularies like the Gene Ontology do not exist and linking between different databases is difficult and unreliable. In such a scenario, InEdge and PathCount cannot be used as each piece of evidence has only exactly one path and taking into account the strength of each individual path is the only way to rank results.

6 Related Work

Probabilistic databases. There has been recent interest in developing general-purpose query evaluation methods on probabilistic databases [12, 46, 36, 38, 13, 2]. Probabilities are typically associated at the tuple level, and the query language is a subset of SQL. Probabilistic data is used in data integration in [33, 44, 16]. These studies start from the assumption that the data is probabilistic, or assume that the data is uncertain but that it can be modeled as probabilistic data.

Ranking of integrated biological data. Approaches have been proposed that exploit the global link structure between integrated data items for ranking biological entities. Lacroix et al. [27] for example, propose path-length and cardinality for ranking data. Hussels et al. [22] add to the above by highly ranking "surprising" results, such which are less connected among sources. Biozon [8] uses a ranking algorithm based on the PageRank model. However, uncertainty of individual data points are not taken into account and play no role in determining ranks. BioRank is unique in that it models uncertainty inherent in data entities, links between entities, and at the data source level. In addition, our model implicitly accounts for the link structure between data items. Whereas our approach is computationally more expensive than simple non-probabilistic ranking methods, our experiments indicate that ranking by link structure alone is insufficient to find new and less-known, but valuable information.

Reliability and Propagation for biological inference. Network Reliability and Propagation algorithms have previously been proposed for inferring protein complex membership [3, 45]. These studies have shown promise in terms of efficient calculation of reliability and propagation as well as biological inference. They differ from our approach however in that they are creating and evaluating a specific model. They are not concerned with general-purpose data integration and information retrieval uncertainty semantics.

Robustness of ranking results. Work related to our robustness results can be found in the AI literature. There is a long work discussing if and when Bayesian belief networks are sensitive to their input parameters (see e.g. [25, 10]). Studies which conclude that belief networks can be sensitive are concerned with (i) rank-order "swaps" in result lists, and (ii) large changes in output probabilities. We use average precision as an evaluation metric for UII because our primary goal is to separate the good results from the poor. So long as we cluster the most relevant results towards the top of our ranked list, we are not overly concerned with individual rank-order swaps. Additionally we do not attach meaning to the probabilities (relevance scores) beyond their relative ordering. In this regard, we are similar to findings of robustness in belief networks [20]. The important implication of our results is that also for data integration, retaining probabilities in the process of integrating data can increase the ranking quality even though the actual "correctness" of the individual probabilities may not be guaranteed.

7 Conclusions

Much recent work has investigated techniques for efficient processing of probabilistic data. Surprisingly little has been done to compare probabilistic methods to their alternative deterministic counterparts and examine the actual trade-off between (i) the added complexity of probabilistic query processing and (ii) the assumed added value in the quality of results. In this paper, we investigated this fundamental trade-off in the context of a concrete scientific research problem. Our experiments led to the following observations:

- Introducing a probabilistic framework into an existing biological data integration application allowed us to consistently higher rank *less-known* information, which is such information that is true, but not widely known. In contrast, probabilities did not improve our results for highly ranking already *well-known* information.
- Slight perturbations in our uncertainty-to-probability transformations do not significantly affect the ranking quality. Our rankings were very robust, which suggests that our method is robust against slightly subjective and varying estimates of domain experts.
- We could find several methods that allowed us to perform probabilistic query evaluations on real data graphs in consistently short time (< 0.1s). We could also provide a new theorem when closed solutions are actually tractable for our applications. This results suggests that, whereas probabilistic query evaluation is more expensive and, in theory often prohibitive, practical problems of real importance seem to actually allow efficient methods.

The high-level take-away from our work is that while probabilistic approaches are not necessary and valuable for all data integration problems, they are ideally suited for highly uncertain domains such as new scientific knowledge discovery.

8 Acknowledgment

We thank Dr. Eugene Kolker and Dr. Peter Myler for their ongoing collaboration, and Nilesh Dalvi for his useful comments. This work was partially supported by NSF IIS-0513877, NSF IIS- 0454425, NSF IIS-0713576, NIH NLM T15 LM07442, and a gift from Microsoft.

References

[1] S. Abiteboul et al. The lowell database research self-assessment. *Commun. ACM*, 48(5):111–118, 2005.

[2] L. Antova, T. Jansen, C. Koch, and D. Olteanu. Fast and simple relational processing of uncertain data. In *ICDE*, pages 983–992. IEEE, 2008.

[3] S. Asthana, O. D. King, F. D. Gibbons, and F. P. Roth. Predicting Protein Complex Membership Using Probabilistic Network Reliability. *Genome Res.*, 14(6):1170–1175, 2004.

[4] T. Attwood. The quest to deduce protein function from sequence: the role of pattern databases. *The International Journal of Biochemistry and Cell Biology*, 32:139–155, 2000.

[5] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.

[6] D. Baker and A. Sali. Protein structure prediction and structural genomics. *Science*, 294(5540):93–96, 2001.

[7] G. Bennett. Probability inequalities for the sum of independent random variables. *Journal of the American Statistical Association*, 57(297):33–45, March 1962.

[8] A. Birkland and G. Yona. BIOZON: a system for unification, management and analysis of heterogeneous biological data. *BMC Bioinformatics*, 7:70, 2006.

[9] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[10] H. Chan and A. Darwiche. When do numbers really matter? *Journal of Artificial Intelligence Research*, 17:265–287, 2002.

[11] C. Colbourn. *The Combinatorics of Network Reliability*. Oxford University Press, Inc., New York, NY, USA, 1987.

[12] N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. In *VLDB*, 2004.

[13] N. Dalvi and D. Suciu. Management of probabilistic data: foundations and challenges. In *PODS*, 2007.

[14] S. B. Davidson, S. C. Boulakia, A. Eyal, B. Ludäscher, T. M. McPhillips, S. Bowers, M. K. Anand, and J. Freire. Provenance in scientific workflow systems. *IEEE Data Eng. Bull.*, 30(4):44–50, 2007.

[15] L. Donelson, P. Tarczy-Hornoch, P. Mork, C. Dolan, J. Mitchell, M. Barrier, and H. Mei. The biomediator system as a data integration tool to answer diverse biologic queries. In *Medinfo*, pages 768–72, 2003.

[16] X. Dong, A. Halevy, and C. Yu. Data integration with uncertainty. In *VLDB*, 2007.

[17] M. Druzdzel and L. Van Der Gaag. Building probabilistic networks: 'where do the numbers come from?'. *IEEE Transactions on Knowledge and Data Engineering*, 12(4):481–486, 2000.

[18] R. Finn, J. Mistry, B. Bockler-Shuster, S. Griffiths-Jones, V. Hollich, T. Lassmann, S. Moxon, M. Marshall, A. Khanna, R. Durbin, S. Eddy, E. Sonhammer, and A. Bateman. Pfam: clans, web tools and services. *Nucleic Acids Research*, 34(Database Issue):D247–D251, 2006.

[19] D. Haft, J. Selengut, and O. White. The TIGRFAMs database of protein families. *Nucleic Acids Research*, 31:371–373, 2003.

[20] M. Henrion, M. Pradhan, B. Favero, K. Huang, G. Provan, and P. O'Rourke. Why is diagnosis using belief networks insensitive to imprecision in probabilities? In *UAI*, 1996.

[21] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, March 1963.

[22] P. Hussels, S. Trißl, and U. Leser. What's new? what's certain? - scoring search results in the presence of overlapping data sources. In *Proc. 4th DILS*. Springer, 2007.

[23] D. Karger. A randomized fully polynomial time approximation scheme for the all-terminal network reliability problem. *SIAM Review*, 43(3):499–422, 2001.

[24] R. M. Karp and M. Luby. Monte-carlo algorithms for enumeration and reliability problems. In *FOCS*, pages 56–64, 1983.

[25] O. Kiersztok and H. Wang. Another look at sensitivity of bayesian networks to imprecise probabilities. In *AISTAT*, 2001.

[26] E. Kolker, A. Piccone, M. Galperin, R. Smith, C. Giometti, K. Nealson, J. Fredrickson, and J. Tiedje. Global profiling of shewanella oneidensis mr-1: Expression of hypothetical genes and improved functional annotations. *PNAS*, 102(6):2099–2104, 2005.

[27] Z. Lacroix, L. Raschid, and M.-E. Vidal. Efficient techniques to explore and rank paths in life science data sources. In *Proc. 1st DILS*, 2004.

[28] B. Louie, T. Detwiler, N. Dalvi, R. Shaker, P. Tarczy-Hornoch, and D. Suciu. Incorporating uncertainty metrics into a general-purpose data integration system. In *SSDBM*, 2007.

[29] B. Louie, P. Tarczy-Hornoch, R. Higdon, and E. Kolker. Validating annotations for uncharacterized proteins in Shewanella oneidensis, 2008. (manuscript in preparation).

[30] D. Maglott, J. Ostell, K. Pruitt, and T. Tatusova. Entrez gene: gene-centered information at ncbi. *Nucleic Acids Research*, 33(Database Issue):D54–D58, 2005.

[31] F. McSherry and M. Najork. Computing information retrieval performance measures efficiently in the presence of tied scores. In *ECIR*, pages 414–421, 2008.

[32] P. Mork, A. Y. Halevy, and P. Tarczy-Hornoch. A model for data integration systems of biomedical data applied to online genetic databases. In *AMIA Annual Fall Symposium*, pages 473–77, 2001.

[33] A. Nierman and H. Jagadish. Protdb: Probabilistic data in xml. In *VLDB*, 2002.

[34] S. Philippi and J. Kohler. Addressing the problems with life-science databases for traditional uses and systems biology. *Nature Reviews Genetics*, 7:481–488, June 2006.

[35] M. Pradhan, M. Henrion, G. Provan, B. Del Favero, and K. Huang. The sensitivity of belief networks to imprecise probabilities: an experimental investigation. *Artificial Intelligence*, 85(1-2):363–397, 1996.

[36] C. Re, N. Dalvi, and D. Suciu. Efficient Top-k query evaluation on probabilistic data. In *ICDE*, 2007.

[37] R. J. Roberts. Identifying protein function–a call for community action. *PLoS Biol*, 2(3):E42, 2004.

[38] P. Sen and A. Deshpande. Representing and querying correlated tuples in probabilistic databases. In *ICDE*, 2007.

[39] R. Shaker, P. Mork, M. Barclay, and P. Tarczy-Hornoch. A rule driven bi-directional translation system remapping queries and result sets between a mediated schema and heterogeneous data sources. *Jour Amer Med Inform Assoc, Fall Symposium Suppl*, pages 692–696, 2002.

[40] R. Shaker, P. Mork, J. Brockenbrough, L. Donelson, and P. Tarczy-Hornoch. The biomediator system as a tool for integrating databases on the web. In *IIWeb*, 2004.

[41] A. Tversky and D. Kahneman. Judgment under uncertainty: heuristics and biases. *Science*, 185:1124–31, 1974.

[42] A. Valencia. Automatic annotation of protein function. *Current Opinion in Structural Biology*, 15:267–274, 2005.

[43] L. Valiant. The complexity of enumeration and reliability problems. *SIAM J. Comput.*, 8:410–421, 1979.

[44] M. van Keulen, A. de Keijzer, and W. Alink. A probabilistic xml approach to data integration. In *ICDE*, 2005.

[45] J. Weston, A. Elisseeff, D. Zhou, C. S. Leslie, and W. S. Noble. Protein ranking: from local to global structure in the protein similarity network. *PNAS*, 101(17):6559–6563, Apr. 27 2004.

[46] J. Widom. Trio: A system for integrated management of data, accuracy, and lineage. In *CIDR*, 2005.

[47] C. Wu, H. Huang, A. Nikolskaya, Z. Hu, and W. Barker. The iproclass integrated database for protein functional analysis. *Computational Biology and Chemistry*, 28:87–96, 2004.

[48] J. Ye, S. McGinnis, and T. Madden. BLAST: improvements for better sequence analysis. *Nucleic Acids Research*, 34(Web Server issue):W6–W9, 2006.

A Number of Monte Carlo iterations

Proof of Theorem 3.1

Proof. We introduce a random variable X_i to model the outcome of the i-th trial according to the following rule: X_i becomes "1" if node 1 was false and node 2 true, "-1" if node 1 was true and node 2 was false and "0" if either both were true or false (probability $1 - r_2(1 - r_1) - r_1(1 - r_2)$). Hence, X_i has the probability mass function

$$p_{X_i}(x_i) = \begin{cases} r_2(1 - r_1) & \text{if } X_i = 1 \\ 1 + 2r_1r_2 - r_1 - r_2 & \text{if } X_i = 0 \\ r_1(1 - r_2) & \text{if } X_i = -1 \ . \end{cases}$$

We now consider the random variable $X = S/n$ with $S = \sum_{i=1}^{n} X_i$. The expected value $\mathbf{E}(X)$ of X can easily be verified to be $r_2 - r_1 = -\varepsilon$. This means that X is a fair or unbiased estimator of $-\varepsilon$. The probability $\mathbf{P}[\hat{r}_1 \leq \hat{r}_2]$ and, therefore, the probability that we do not necessarily rank the nodes in the correct order is equal to the probability that $X \geq 0$. Hence, we want to bound the probability $\mathbf{P}[X \geq 0] \leq \delta$. We can get rather tight bounds by using both the first and second moment in a formula by Bennet [7] in the form stated by Hoeffding [21][Inequality (2.9)].

Formula 1 (Bennet)**.** *Let $Y_1, Y_2, \ldots, Y_n$ be independent random variables with $Y = \frac{1}{n}\sum_{i=1}^{n} Y_i$, $\mathbf{E}(Y) = 0$, $\sigma^2 = n\mathbf{Var}(Y)$, $Y_i \leq b$, and $i \in \{1, \ldots, n\}$. Then, for $0 < c < b$,*

$$\mathbf{P}[Y \geq c] \leq e^{-\frac{nc}{b}\left(\frac{1+\sigma^2}{bc}\ln\frac{1+bc}{\sigma^2} - 1\right)} \ . \tag{1}$$

We can transform our original random variable to suit the new formula by using $Y_i = X_i + \varepsilon$. Then $Y_i \leq 1 + \varepsilon = b$ and $c = \varepsilon$. We can now use the above inequality to write

$$\mathbf{P}\left[Y \geq \varepsilon\right] \leq \left(e^{-\frac{\varepsilon}{1+\varepsilon}\gamma}\right)^n \ , \tag{2}$$

with

$$\gamma = \left(1 + \frac{1}{\lambda}\right)\ln(1+\lambda) - 1 \ , \quad (\gamma > 0) \ ,$$

and

$$\lambda = \frac{bc}{\sigma^2} = \frac{\varepsilon(1+\varepsilon)}{\sigma^2}, \quad (\lambda > 0) \ .$$

We can now bound γ by

$$\gamma > \frac{\lambda}{2} - \frac{\lambda^2}{6} = \frac{\varepsilon(1+\varepsilon)}{2\sigma^2}\Big(1 - \frac{\varepsilon(1+\varepsilon)}{3\sigma^2}\Big) \geq \frac{\varepsilon(1+\varepsilon)}{2\sigma^2_{max}}\Big(1 - \frac{\varepsilon(1+\varepsilon)}{3\sigma^2_{max}}\Big) \ .$$

Calculating σ^2 as

$$\sigma^2 = r_2(1-r_1) + r_1(1-r_2) - (r_2 - r_1)^2 \ ,$$

then using $r_1 = r_2 + \varepsilon$ and determining σ^2_{max} by setting $\frac{d\sigma^2}{dp_1} = 0$, we get that σ^2_{max} for $p_1 = \frac{1+\varepsilon}{2}$, which approaches 0.5 for decreasing ε. Plugging back into σ^2, we get

$$\sigma^2_{max} = \frac{1}{2}(1+\varepsilon)^2 \ ,$$

and for $\gamma > \frac{\varepsilon(3+\varepsilon)}{3(1+\varepsilon)^2}$. Hence, we know that

$$\mathbf{P}\left[Y \geq \varepsilon\right] \leq \left(e^{-\frac{\varepsilon}{1+\varepsilon}\gamma}\right)^n \ < \left(e^{-\frac{\varepsilon^2(3+\varepsilon)}{3(1+\varepsilon)^3}}\right)^n \ .$$

Requiring now $\mathbf{P}\left[Y \geq \varepsilon\right] \leq \delta$, we get as bounds,

$$n_{\text{simulations}} = \left\lceil \frac{(1+\varepsilon)^3}{\varepsilon^2\big(1+\frac{\varepsilon}{3}\big)} \ln\left(\frac{1}{\delta}\right) \right\rceil \ . \tag{3}$$

□

B Reducable Schemas

Proof of Theorem 3.2

Proof. Part A: In this case the data graph is a tree with s the root node and t one of the leaf nodes. The reduction rules will first eliminate all branches other than $s \to t$, then collapse the path $s \to t$ to a single edge.

Part B: Consider every node $x \in P$. There is exactly one incoming edge (from Q) and exactly one outgoing edge (from Q'). Compose the two edges (the *serial-path* reduction), and remove the node x, and repeat for all x'. This will remove all nodes x, and the newly introduced edges are all independent. Moreover the new (smaller) graph is an instance of schema S', hence we can apply induction: since the smaller graph is reducible, so is the original graph. □