Answering Top K Queries Efficiently with Overlap in Sources and Source Paths

Louiqa Raschid
University of Maryland
louiqa@umiacs.umd.edu

María Esther Vidal
Universidad Simón Bolívar
mvidal@ldc.usb.ve

Yao Wu
University of Maryland
yaowu@cs.umd.edu

Felix Naumann **Jens Bleiholder**
Hasso-Plattner-Institut, Potsdam
Felix.Naumann@hpi.uni-potsdam.de
Jens.Bleiholder@hpi.uni-potsdam.de

Abstract

Challenges in answering queries over Web-accessible sources are selecting the sources that must be accessed and computing answers efficiently. Both tasks become more difficult when there is overlap among sources and when sources may return answers of varying quality. The objective is to obtain the best answers while minimizing the costs or delay in computing these answers and is similar to solving a Top K problem efficiently. We motivate these problems and discuss solution approaches.

Introduction

When querying Web sources, there are often multiple sources that can answer a query. A challenge in answering queries over Web-accessible sources is the appropriate selection of those sources to provide sufficient query results in an efficient manner. Selecting a subset of sources may be difficult because a characteristic of Web sources is that they overlap in the content they provide. In the simple case, information about some object may be available from multiple sources. In a more complex case, answering queries requires following navigational paths (labeled source paths) through multiple sources. Both the data objects and the edges between the objects in these source paths may overlap. This is similar to the overlap of multiple graphs as will be discussed. A further characteristic is that answers from Web sources vary in their quality or relevance, and efficient query answering requires choosing sources so that the best answers will be retrieved.

Suppose that there is a cost metric associated with accessing each source or source path, and a benefit metric associated with each target object that answers the query and that can be reached by a source or source path. Choosing less sources may retrieve objects in the set of target objects (TO) with lower benefit while choosing more sources will increase execution cost and delay in returning answers. We define an optimization problem to obtain the Top K answers (based on the benefit metric) while minimizing the cost of accessing the sources or source paths. There is a related problem of accurately estimating the benefit associated with each target object in TO so as to efficiently identify the Top K answers. We use two examples to motivate the research and then address these two problems.

Motivating Examples

Consider a set of bibliographic sources. Typically a query can be answered by accessing multiple sources and there may be an overlap of the target objects (TO) for the query in these sources. Visiting a source will retrieve all (or some) of the objects in TO. There typically is an access cost or delay associated with visiting each source and collecting the TOs. For simplicity, we assume that when there is overlap, then retrieving the object from any one source is sufficient. There can be a more general case where the information about the object varies across sources, so additional information can be obtained from visiting multiple sources. Another possibility is that information is updated over time and some sources may contain more accurate information.

In (Bleiholder *et al.* 2006; Raschid *et al.* 2006), we described an example of navigational queries traversing source paths in the life science domain. Consider the following query: *"Return all publications of* PUBMED *that are linked to a gene entry or an* OMIM *entry about diseases related to the keyword (gene name)* `tnf`*."* A set of source paths that can answer this query is shown in Figure 1. Figure 2 lists these source paths, e.g., a path (OMIM → NCBI_Protein → PUBMED) that goes from OMIM through NCBI_Protein to reach objects in PUBMED.

Each of these source paths corresponds to a *result graph (RG)* of objects and edges between the objects. There is overlap of objects and edges in the different RGs. There is also overlap of the *target objects (TO)* of the query; in this case the TO are publications in PUBMED.

A navigational query is answered by evaluating one or more source paths to reach the set of target objects TO. Evaluating a source path may require visiting multiple sources. Intermediate objects may have to be downloaded (to validate search criteria). An object in TO may also be reached by different source paths. These issues complicate the task of choosing a subset of source paths to answer the query efficiently.

Each object in TO is associated with a benefit score (rank). The exact value of the score depends on the metric. For a document, this score can be an IR style/word based score reflecting relevance. Other options are a qual-

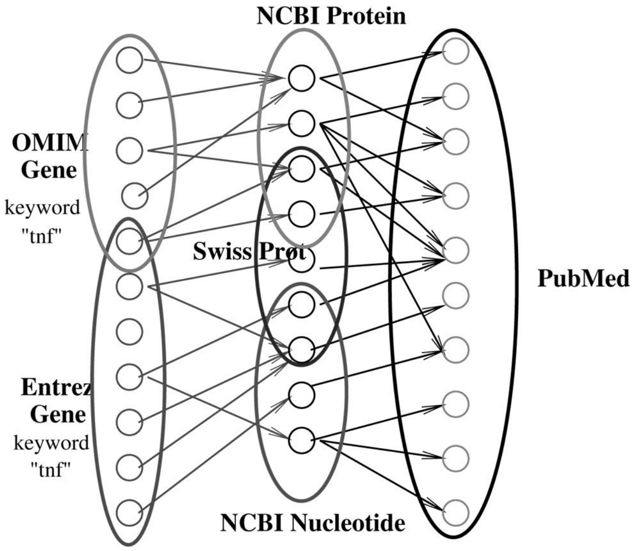

Figure 1: An example of multiple source paths and overlap of TO

(P1) OMIM → NCBI_Protein → PUBMED
(P2) OMIM → SwissProt → PUBMED
(P3) OMIM → NCBI_Nucleotide → PUBMED
(P4) EntrezGene → SwissProt → PUBMED
(P5) EntrezGene → NCBI_Nucleotide → PUBMED

Figure 2: Several paths from OMIM or EntrezGene to PUBMED

ity score provided by users or a score reflecting importance such as PageRank or ObjectRank (Balmin, Hristidis, & Papakonstantinou 2004; Page *et al.* 1998). We make the assumption the score is known a priori. If the score is local to each source, and the object appears in multiple sources, then one may need to compute some aggregate score. For simplicity, we assume that the score (rank) is global and not query-dependent.

However, the above assumption that the local (or global) score is known a priori is probably not realistic since this is an expensive computation that has to be evaluated for all possible queries. Consider the situation of multiple source paths and the corresponding result graphs (RGs) for some query. Suppose the objects and links of all the RGs are stored locally. Computing the exact ranking for objects in TO may require visiting all source paths (RGs) and it may be computationally expensive. This becomes even more expensive if the objects and edges of the RGs are not locally stored. One solution to this problem is to estimate an approximate benefit score and compute an approximate ranking. However, we must guarantee a high correlation between the exact and approximate ranking.

Problem Definition

Based on our motivating examples and discussion, we present two related problems as follows:

Problem Cover(TopK): Find a minimal set of sources or source paths that must be visited so that the K objects with the highest benefit score in TO are reached. The minimal set of sources or source paths can either be based on the cardinality of selected sources (paths) or based on the least cost.

Consider a collection of sources or source paths $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$, and a world of target objects $Z = \{z_1, z_2, \ldots, z_n\}$.
There is a mapping to indicate if element z_j occurs as a TO of s_i.
There is an associated cost $c(s_i)$ for each s_i.
Further, there is a benefit metric that determines a score b_j for object z_j.
For each TO there is a set of K objects with the highest benefit score $TO_K = o_1, o_2, \ldots, o_K$ that answers query Q.
The goal of **Cover(TopK)** is to find a subset $\mathcal{S}'$ of $\mathcal{S}$ to cover all objects in TO_K while minimizing the cost of visiting $\mathcal{S}'$.

We note that in **Cover(TopK)**, the identity of the top K target objects TO_K is known a priori. This allows us to model the Top K problem as a set cover problem as will be discussed. This is in contrast to most versions of the Top K problem where some mechanism is used to probe sources and compute the aggregate score of each object and to determine the Top K.

Determining the identity of the top K target objects TO_K a priori may be expensive as discussed earlier. This leads us to the definition of the next problem:

Problem EstimateRank: Efficiently estimate the benefit score for some object in TO so that the relative error of estimating the benefit score, for some metric M, is within some confidence level.

Consider a result graph RG, a metric M, a benefit score b_j for object z_j, and a sampled graph $\overline{RG}$ which is a sampled subset of nodes and edges of RG.
The estimated benefit of z_j in $\overline{RG}$ is $est\_ben(z_j, \overline{RG}, M)$.
The objective of **EstimateRank** is to determine the minimum sampling cost of constructing $\overline{RG}$ or an upper bound on the sample size, so that the confidence level in the relative error of estimating the benefit metric, $est\_ben(z_j, \overline{RG}, M)$, is at least α.

Related Research

There is a rich body of research on efficient Top K techniques (Bruno, Chaudhuri, & Gravano 2002; Chang & won Hwang 2002; Das *et al.* 2006; Fagin, Lotem, & Naor 2001). Given a monotonic rank aggregation function (on several attributes), the Top K problem computes the K objects with the highest combined score. Two classical algorithms TA and NRA are proposed in (Fagin, Lotem, & Naor 2001) to achieve early termination of query processing. By exploiting statistics, (Bruno, Chaudhuri, & Gravano 2002) proposes a way to determine a range query to evaluate a Top K query. (Chang & won Hwang 2002) explores the problem when some predicates in the query are expensive to evaluate; their solution will minimize the number of expensive probes by identifying a set of necessary probes. (Das *et al.* 2006) addresses the problem using materialized views to reduce the number of retrieved tuples. The view selection problem is solved for two attribute relations and then extended to multi-attribute relations. We note that **Cover(TopK)** identifies a novel variation of Top K since it assumes that the K target objects with the highest benefit TO_K are known. Thus, it addresses a set cover problem in selecting sources or source paths. There is some related research in (Bender *et al.* 2005; Michel *et al.* 2006), where a quality estimation method is presented for selecting and adding a new peer in a P2P system. **Cover(TopK)** takes a different approach since we first estimate the benefit for each object and then solve an optimization problem. In prior work (Bleiholder *et al.* 2006), we considered query evaluation for navigational queries, but we did not consider the quality of the answers returned or try to estimate the benefit of the target objects or try to solve a Top K problem; hence this paper is a significant extension of our prior research.

Solutions to the **EstimateRank** problem will estimate the score of the objects in the TO efficiently, so as to avoid visiting a large number of objects in the RG. We summarize related work. In the context of query optimization, different sampling-based algorithms have been proposed to estimate the cardinality of a query efficiently (Haas & Swami 1992; Hou, Ossoyoglu, & Doglu 1991; Ling & Sun 1992; Lipton & Naughton 1990; Lipton, Naughton, & Schneider 1990; Ruckhaus, Ruiz, & Vidal 2006). The challenge of these methods is to reach estimates that satisfy the required confidence levels while the size of the sample remains small. A key decision involves when to stop sampling the population and this is determined by the mean and variance of the sample in comparison to the target population. The different techniques use different methods to reach this decision. In (Lipton & Naughton 1990; Lipton, Naughton, & Schneider 1990), mean and variance are approximated using some upper bounds which are defined in terms of the cardinality constraints that characterize the relationships between the objects in the population to be sampled. (Haas & Swami 1992; Hou, Ossoyoglu, & Doglu 1991) do not define an upper bound for these statistics; in contrast, they approximate them on-the-fly. In (Hou, Ossoyoglu, & Doglu 1991), mean and variance are computed from a small portion of the data, which is sampled in the first stage. In (Haas & Swami 1992), mean and variance are recomputed during each iteration of the sampling. Thus, the last two techniques are able to reach better estimates but require, in general, more time for sampling.

Proposed Solution

Solution to Cover(TopK)

We first formalize this problem as an Integer Programming/Linear Programming (IP/LP) as follows:
Let t_j indicate if z_j is in TO_K as given.
We set integer variables $x_i = 1$ iff source/source path s_i is chosen in the solution.
We set integer variables $y_j = 1$ iff z_j is covered in the solution.

Minimize $\sum_{i=1}^{m} c(s_i) \cdot x_i$

subject to

$$\begin{array}{lll} \sum_{j=1}^{n} y_j \cdot t_j & \geq & K \\ y_j & \leq & \sum_{\{l \mid z_j \in s_l\}} x_l \quad \text{for all } j \\ x_i \in \{0,1\} & & \text{for all } i \\ y_j \in \{0,1\} & & \text{for all } j \end{array}$$

This IP formulation can be relaxed to obtain an LP formulation for an approximate solution.

Next, we consider a greedy solution to **Cover(TopK)**. We assume that the exact benefit of each object in each source or source path is computed a priori. As a result, the Top K target objects TO_K is known, as well as the source or source path in which they occur. We can exploit this knowledge to choose paths efficiently to cover TO_K.

```
Algorithm GreedyCover(TopK)
▷ Let $M_1$ be a matrix where a value of 1 at entry $M_1(i,j)$
  indicates if element $z_j$ occurs in source or path $s_i$.
▷ Let $M_2$ be a vector where a value of 1 at entry $M_2(j)$
  indicates if element $z_j$ occurs in the top K objects $TO_K$
  for some given metric.
▷ Let $U$ be a vector where a value of 1 at entry $U(j)$
  indicates if element $z_j$ occurs in the subset of sources
  picked so far. Initially, $U$ contains all 0s.
1. Rank source or source paths $s_i$ based on $R_i = Count_i/c(s_i)$
   in descending order.
   $Count_i$ is the count of the number of objects in $s_i$ that
   occur in $TO_K$, and can be determined using $M_1$ and $M_2$.
   $Count_i = \sum_{l=1}^{n}(M_1(i,l) \cdot M_2(l))$
2. Pick source or source path $s_t$ with the largest ratio $R_t$.
   – Update vector $U$ to determine the union of objects covered
     so far. $U(j) = U(j) \vee M_1(t,j)$.
   – Adjust the ratio of the remaining paths $s_k$ to discount
     $Count_k$ by those objects in $s_t$ that occur in $TO_K$
     and occur in the overlap of $s_t$ and $s_k$.
     $Count_k = \sum_{l=1}^{n}(M_1(k,l) \cdot (1 - U(l)) \cdot M_2(l))$
3. Continue choosing sources or paths until $TO_K$ is covered.
```

As discussed earlier, computing the exact benefit b_j can be expensive. Suppose we cannot assume that we know TO_K or $M_2(j)$ a priori. Now the greedy algorithm will not be able to determine $Count_i$ for each source or source path s_i. One solution to this problem is to estimate an approximate benefit score and this leads us to the related problem **EstimateRank**. The approximate ranking, however, must guarantee a high correlation between the exact and approximate ranking.

A variant of the greedy algorithm can then rank the paths using the ratio ($Sum\_Est\_Benefit_i/cost_i$), where $Sum\_Est\_Benefit_i$ is the sum of estimated benefits and can be obtained using the estimated value of b_j and $M_1(i,j)$. Methods to estimate the value of b_j are discussed next.

Solution to EstimateRank

Recall that the objective of **EstimateRank** is to sample the result graph and estimate the benefit for some object z_j in

TO. Consider a result graph RG, a metric M, a benefit score b_j for object z_j, and a sampled graph $\overline{RG}$ which is a sampled subset of nodes and edges of RG. The objective of **EstimateRank** is to determine the minimum sampling cost of constructing $\overline{RG}$ or an upper bound on the sample size, so that the confidence level in the relative error of estimating the benefit metric, $est\_ben(z_j, \overline{RG}, M)$, is at least α.

Consider the case where the benefit associated with a target object is a quality score that is assigned to the object by a set of users. Alternately, we may consider a score that is determined by some automated technique based on the content of the target object, e.g., an IR-style score that is based on the bag of words in the object. Such a benefit score will be independent of the link structure or other properties of the result graph. To model this situation, we first present a *metric independent* solution to **EstimateRank**. However, many search engines on the Web rely on metrics such as PageRank or ObjectRank (Balmin, Hristidis, & Papakonstantinou 2004; Page *et al.* 1998) where the score reflects importance of an object and depends on the link structure of the result graph. To address this situation, we then present a *metric dependent* solution to **EstimateRank** that exploits properties of the metric. For simplicity, we explain the latter case using a metric *PathCount* whose computation is simpler compared to PageRank.

Consider the RG in Figure 3; it is a layered DAG. A layered DAG can partition its nodes into multiple layers, and any edge in the layered DAG can only occur between nodes of adjacent layers. See (Raschid *et al.* 2006) for details of navigational queries and how we construct the result graph RG as a layered DAG. The figure shows a layered graph with 4 layers and the object paths from objects in the first layer to the 3 target objects in the fourth layer.

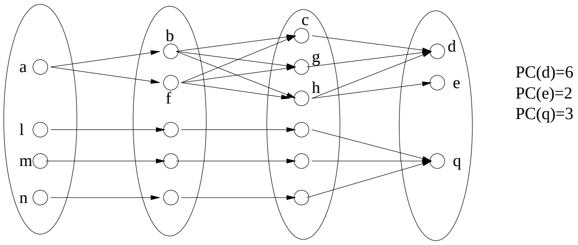

Figure 3: PathCount for an object in TO

Consider a target object z_j in layer L_n of RG. Let $BL_{j,(n-1)}$ be the set of *backlink*s from layer $L_{(n-1)}$ to L_n that reach z_j. In Figure 3, $BL_{d,3}$ is comprised by the backlinks of the object *d* in the third layer.

Metric Independent Solution to EstimateRank A metric independent solution to the **EstimateRank** problem is defined as follows:

Let $\overline{TO}$={$\overline{z_1}, \overline{z_2}, ..., \overline{z_m}$} be a sample of TO such that each $\overline{z_j}, 1 \leq j \leq m$, is randomly chosen from TO with replacement.

A sampled graph $\overline{RG}$ corresponds to the minimal sub-graph of RG that contains only the objects in $\overline{TO}$ in the last layer, i.e., $\overline{L_n}$ is equal to $\overline{TO}$. In addition, any layer $\overline{L_i}$ of $\overline{RG}$ is only composed of the objects in L_i of RG that are *backlinks* of the objects in layer $\overline{L_{i+1}}$.

To ensure that the estimation of the ranking is between certain convergence bounds, the size m of the sample is defined using the Chernoff Bound (Scholkopf & Smola 2001) as follows:

Let $X_v(\overline{z_j})$ be an independent identically distributed (i.i.d.) binary random variable that has value 1 if the benefit score of the sample target object $\overline{z_j}$ is v, and 0 otherwise. Let S_v be another random variable that averages the variables $X_v(\overline{z_j})$ for the objects in $\overline{TO}$, i.e.,

$$S_v = \frac{1}{m} \sum_{j=1}^{m} (X_v(\overline{z_j}))$$

Let p be the probability of the benefit score for an object $\overline{z_j}$ being v, i.e., $Pr(X_v(\overline{z_j}) = 1) = p$. Since the sequence $X_v(\overline{z_1}), X_v(\overline{z_2}), \ldots, X_v(\overline{z_n})$ represents a sequence of Bernoulli trials, p corresponds to the probability of success of the trials or the expection of S_v denoted by $E(S_v)$.

Then by using the Chernoff bound, the size m of the sample has to satisfy the following formula to ensure that the relative error of the estimation of $E(S_v)$ is greater than some given constant ϵ with some probability α:

$$Pr(|S_v - E(S_v)| \geq \epsilon) \leq 2exp(^{-2m\epsilon^2})$$

Sufficient objects in layers L_1 to L_{n-1} need to be sampled in order to reach m target objects in $\overline{TO}$, in layer $\overline{L_n}$ of the sampled graph $\overline{RG}$.

Metric Dependent Solution to EstimateRank Given very large samples (large result graphs and large cardinality of the target objects in TO), it is well known that a metric-independent sampling performs well. However, domain information, such as the properties of the result graph or target objects or properties of the metric, can be used to determine a more precise sample size, and to improve the efficiency of the estimation method. For illustration, we consider a simple example metric, `PathCount` or PC, to show how domain information can be exploited. The path count measure was introduced in (Katz 1953) in the context of social networks. The PC benefit score represents the number of object paths through the RG that reach a target object z_j in TO. Thus, in Figure 3, the value of the benefit score for the metric `PathCount` (PC) for the target objects d, e, and q, is 6, 2, and 3, respectively.

Consider a target object z_j in layer L_n of RG. Let $BL_{j,(n-1)}$ be the set of *backlink*s from layer $L_{(n-1)}$ to L_n that reach z_j. Then, the benefit b_j or the PC score for z_j is the sum of the PC score for each of the objects in layer $L_{(n-1)}$ that is reached by each backlink in $BL_{j,(n-1)}$. Thus,

$$b_j = PC(z_j) = \sum_{z_i:(z_i,z_j) \in BL_{j,(n-1)}} PC(z_i),$$

where z_i is an object in layer $L_{(n-1)}$ such that edge (z_i, z_j) is in $BL_{j,(n-1)}$.

Our sampling strategy to estimate the benefit score b_j is as follows:

- Edges in $BL_{j,(n-1)}$ are sampled with replacement to estimate the PC scores of each object z_j using the sampled links $Sample\_BL_{j,(n-1)}$.
- The estimated PC for z_j is

  $$est\_ben(z_j, \overline{RG}, PC) = (s/m) * Card(BL_{j,(n-1)})$$

  where s is the sum of the PC values of the sampled objects in $Sample\_BL_{j,(n-1)}$, m is the size of the sample, i.e., the cardinality of $Sample\_BL_{j,(n-1)}$, and $Card(BL_{j,(n-1)})$ is the cardinality of $BL_{j,(n-1)}$.
- Different sampling-based techniques are used to decide when to stop sampling $BL_{j,(n-1)}$, i.e., to determine m. These methods are based on the estimation of mean (Y) and variance (S) of the PC scores of the objects in $BL_{j,(n-1)}$. Our objective is to reach (through sampling) $Sample\_BL_{j,(n-1)}$ with mean $\overline{Y}$ such that the probability that the relative error of the estimation is greater than some given constant r is α; α is the confidence level. The expression is as follows:

  $$P(|\frac{Y - \overline{Y}}{Y}| \geq r) = \alpha$$

- Recall that the sum of PC scores of the sampled objects is s. We stop sampling when s exceeds an upper bound b. We consider the following three methods, based on the estimation of mean (Y) and variance (S) of the PC scores of the objects in $BL_{j,(n-1)}$ to determine b:
  - **Adaptive Sampling (Lipton & Naughton 1990; Lipton, Naughton, & Schneider 1990)**: The upper bound b is defined as an approximation of $\frac{S}{Y}$. Following this approach, we define b as the product of the maximal indegree of the nodes (objects) in all layers L_1, L_2, $\cdots$, L_{n-2} preceding layer L_{n-1}, i.e.,

    $$b = \prod_{i=1}^{n-2} Max(Indegree(L_i))$$

    Thus, b corresponds to an upper bound of the PC score of any object in layer L_{n-1}. The stop condition of the sampling is as follows:

    $$s > k_1 \times b \times d \times (d+1)$$

    where, k_1 and d are values associated with the desired relative error r and the confidence level of the sampling α. We refer the reader to (Lipton & Naughton 1990; Lipton, Naughton, & Schneider 1990) for details.
  - **Double Sampling (Hou, Ossoyoglu, & Doglu 1991)**: This approach involves two phases of sampling. In the first phase, t% of the objects in $BL_{j,(n-1)}$ are sampled to estimate S and Y. Then, these estimated values are used to compute m, the desired cardinality of the sample $Sample\_BL_{j,(n-1)}$, using the following formula:

    $$m = \frac{(S \times t_\alpha)^2}{(r \times Y)^2}(1 + \frac{8r}{t_\alpha} + \frac{S^2}{m_1 Y^2}) + \frac{2}{m_1}$$

    where, m_1 is the size of the first sample, α is the confidence level, r is the relative error, and t_α is defined based on α and an standardized normal random variable. This new equation is considered because some certain corrections has to be made if S and Y are estimated. We refer the reader to (Hou, Ossoyoglu, & Doglu 1991) for details.
  - **Sequential Sampling (Haas & Swami 1992)**: Following this technique, S and Y are estimated at each sampling step using all the current observations. Let S_t and Y_t be the estimators of S and Y after t objects in $BL_{j,(n-1)}$ have been sampled with replacement, and let s be the sum of observed PC scores and let b be the upper bound defined for the adaptive sampling method (Lipton & Naughton 1990; Lipton, Naughton, & Schneider 1990). Then, the stop condition is as follows:

    $$r \times max(s, b) \geq t_\alpha (t \times S_t)^{1/2}$$

    Thus, the termination condition is determined at every sampling step, and sampling will terminate when the desired accuracy and confidence level are reached.

We plan to extend our current sampling techniques with additional domain information about the conditional probability of accessing an object o from an object p that is in the backlink set for o. This will allow us to estimate the probability of visiting a node in a subgraph. These estimated subgraphs can be exploited to solve the *EstimateRank* problem for metrics such as *ObjectRank* (Balmin, Hristidis, & Papakonstantinou 2004) and *lgOR* (Raschid *et al.* 2006), where the structure of the subgraph (result graph) plays an important role in the benefit score of a target object.

Proposed Evaluation

We are in the process of implementing and evaluating our solutions to the two problems **Cover(TopK)** and **EstimateRank**. The evaluation will be on a biological database from NCBI/NIH, the gatekeeper for biological data produced using federal funds in the US[1]. We have constructed a graph of 10 data sources and 46 links. We used several hundred keywords to sample data from these sources (the EFetch utility) and followed links to the other sources (the ELink utility). We created a data graph of approximately 28.5 million objects and 19.4 million links.

For **Cover(TopK)**, we will compare the coverage and cost trade-off of our greedy solution compared to the optimal solution. For **EstimateRank**, we will experiment with the `PathCount` metric discussed in this paper as well as other well known metrics, such as PageRank and ObjectRank (Balmin, Hristidis, & Papakonstantinou 2004; Page *et al.* 1998). We will determine the effectiveness of the different sampling techniques for these metrics.

Acknowledgements:

This research has been partially supported by the National Science Foundation under grants IIS0222847 and

[1] www.ncbi.nlm.nih.gov

IIS0430915, and German Research Society (DFG grant no. NA 432). We thank Samir Khuller for his feedback on maximum cover problems, Blai Bonet for his comments on graph sampling techniques, and Woei-Jyh Lee, Hector Rodriguez and Luis Ibañez for their participation in creating the datasets and running experiments.

References

Balmin, A.; Hristidis, V.; and Papakonstantinou, Y. 2004. Objectrank: Authority-based keyword search in databases. In *VLDB*, 564–575.

Bender, M.; Michel, S.; Triantafillou, P.; Weikum, G.; and Zimmer, C. 2005. Improving collection selection with overlap awareness in p2p search engines. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, 67–74. New York, NY, USA: ACM Press.

Bleiholder, J.; Khuller, S.; Naumann, F.; Raschid, L.; and Wu, Y. 2006. Query planning in the presence of overlapping sources. In *EDBT*.

Bruno, N.; Chaudhuri, S.; and Gravano, L. 2002. Top-k selection queries over relational databases: Mapping strategies and performance evaluation. *ACM Trans. Database Syst.* 27(2):153–187.

Chang, K. C.-C., and won Hwang, S. 2002. Minimal probing: supporting expensive predicates for top-k queries. In *SIGMOD '02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, 346–357. New York, NY, USA: ACM Press.

Das, G.; Gunopulos, D.; Koudas, N.; and Tsirogiannis, D. 2006. Answering top-k queries using views. In *VLDB'2006: Proceedings of the 32nd international conference on Very large data bases*, 451–462. VLDB Endowment.

Fagin, R.; Lotem, A.; and Naor, M. 2001. Optimal aggregation algorithms for middleware. In *PODS '01: Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 102–113. New York, NY, USA: ACM Press.

Haas, P., and Swami, A. 1992. Sequential sampling procedures for query estimation. In *In Proceedings of VLDB*.

Hou, W.; Ossoyoglu, G.; and Doglu. 1991. Error-constrained count query evaluation in relational databases. In *In Proceedings of SIGMOD*.

Katz, L. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18(1):39–43.

Ling, Y., and Sun, W. 1992. A supplement to sampling-based methods for query size estimation in a database system. *SIGMOD Record* 21(4):12–15.

Lipton, R. J., and Naughton, J. F. 1990. Query size estimation by adaptive sampling (extended abstract). In *PODS '90: Proceedings of the ninth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, 40–46. New York, NY, USA: ACM Press.

Lipton, R. J.; Naughton, J. F.; and Schneider, D. A. 1990. Practical selectivity estimation through adaptive sampling. In *SIGMOD '90: Proceedings of the 1990 ACM SIGMOD international conference on Management of data*, 1–11. New York, NY, USA: ACM Press.

Michel, S.; Bender, M.; Triantafillou, P.; and Weikum, G. 2006. IQN routing: Integrating quality and novelty in P2P querying and ranking. In Ioannidis, Y.; Scholl, M. H.; Schmidt, J. W.; Matthes, F.; Hatzopoulos, M.; Boehm, K.; Kemper, A.; Grust, T.; and Boehm, C., eds., *Advances in Database Technology - EDBT 2006: 10th International Conference on Extending Database Technology*, volume 3896 of *Lecture Notes in Computer Science*, 149–166. Munich, Germany: Springer.

Page, L.; Brin, S.; Motwani, R.; and d, T. W. 1998. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project.

Raschid, L.; Wu, Y.; Lee, W.-J.; Vidal, M. E.; Tsaparas, P.; Srinivasan, P.; and Sehgal, A. K. 2006. Ranking target objects of navigational queries. In *WIDM '06: Proceedings of the eighth ACM international workshop on Web information and data management*, 27–34. New York, NY, USA: ACM Press.

Ruckhaus, E.; Ruiz, E.; and Vidal, M. 2006. Query optimization in the semantic web. *Theory and Practice of Logic Programming. Special issue on Logic Programming and the Web*.

Scholkopf, B., and Smola, A. 2001. *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond (Adaptive Computation and Machine Learning)*. The MIT Press.