1. **LITERATURE SURVEY**
2. A Survey on **“An Adaptive Crawler for Locating Hidden-Web Entry Points”**

**Abstract**

In this paper we describe new adaptive crawling strategies to efficiently locate the entry points to hidden-Web sources. The fact that hidden-Web sources are very sparsely distributed makes the problem of locating them especially challenging. We deal with this problem by using the contents of pages to focus the crawl on a topic; by prioritizing promising links within the topic; and by also following links that may not lead to immediate benefit. We propose a new framework whereby crawlers automatically learn patterns of promising links and adapt their focus as the crawl progresses, thus greatly reducing the amount of required manual setup and tuning. Our experiments over real Web pages in a representative set of domains indicate that online learning leads to significant gains in harvest rates—the adaptive crawlers retrieve up to three times as many forms as crawlers that use a fixed focus strategy.

**Conclusion**

We have presented a new adaptive focused crawling strategy for efficiently locating hidden-Web entry points. This strategy effectively balances the exploitation of acquired knowledge with the exploration of links with previously unknown patterns, making it robust and able to correct biases introduced in the learning process. We have shown, through a detailed experimental evaluation, that substantial increases in harvest rates are obtained as crawlers learn from new experiences. Since crawlers that learn from scratch are able to obtain harvest rates that are comparable to, and sometimes higher than manually configured crawlers, this framework can greatly reduce the effort to configure a crawler. In addition, by using the form classifier, ACHE produces highquality results that are crucial for a number information integration tasks.

There are several important directions we intend to pursue in future work. As discussed in Section 5, we would like to integrate the apprentice of into the ACHE framework. To accelerate the learning process and better handle very sparse domains, we will investigate the effectiveness and trade-offs involved in using back-crawling during the learning iterations to increase the number of sample paths. Finally, to further reduce the effort of crawler configuration, we are currently exploring strategies to simplify the creation of the domain-specific form classifiers. In particular, the use of form clusters obtained by the online-database clustering technique described in as the training set for the classi- fier.

1. A Survey on **“Understanding the Deep Web”**

**Abstract**

The most coveted commodity of the information age is indeed information. Information has become a basic need after food, shelter, and clothing. Due to technological advancements, a large amount of information is available on the Web, which has become a complex entity containing information from a variety of sources. Information is found using search engines. A searcher has access to a large amount of information, but it still far from the huge treasury of information lying beneath the Web, a vast store of information beyond the reach of conventional search engines: the “Deep Web” or “Invisible Web.”

The contents of the Deep Web are not included up in the search results of conventional search engines. The crawlers of conventional search engines identify only static pages and cannot access the dynamic Web pages of Deep Web databases. Hence, the Deep Web is alternatively termed the “Hidden” or “Invisible Web.” The term Invisible Web was coined by Dr. Jill Ellsworth to refer to information inaccessible to conventional search engines. But using the term Invisible Web to describe recorded information that is available but not easily accessible, is not accurate.

**Conclusion**

The advent of Internet and access to global information was a great benefit, even though information managers had the difficult task of organizing, retrieving, and providing access to precise information. Users depend on the popular search engines and portals, which cannot provide access to the hidden store of valuable information available in the Deep Web. To access the information available on these databases, users will have to become familiar with the structure of the Deep Web. Any information created should be shared and used, since that alone leads to the creation of more information. When a specific database is created, information regarding its existence should published so that users will be aware and make maximum use of available information.

1. A Survey on **“Relevance and Trust Assessment for Deep Web Sources Based on Inter-Source Agreement”**

**Abstract**

One immediate challenge in searching the deep web databases is source selection—i.e. selecting the most relevant web databases for answering a given query. The existing database selection methods (both text and relational) assess the source quality based on the query-similarity-based relevance assessment. When applied to the deep web these methods have two deficiencies. First is that the methods are agnostic to the correctness (trustworthiness) of the sources. Secondly, the query based relevance does not consider the importance of the results. These two considerations are essential for the open collections like the deep web. Since a number of sources provide answers to any query, we conjuncture that the agreements between these answers are likely to be helpful in assessing the importance and the trustworthiness of the sources. We compute the agreement between the sources as the agreement of the answers returned. While computing the agreement, we also measure and compensate for possible collusion between the sources. This adjusted agreement is modeled as a graph with sources at the vertices. On this agreement graph, a quality score of a source that we call SourceRank, is calculated as the stationary visit probability of a random walk. We evaluate SourceRank in multiple domains, including sources in Google Base, with sizes up to 675 sources. We demonstrate that the SourceRank tracks source corruption. Further, our relevance evaluations show that SourceRank improves precision by 22-60% over the Google Base and other baseline methods. SourceRank has been implemented in a system called Factal.

**Conclusion**

A compelling holy grail for the information retrieval research is to integrate and search the structured deep web sources. An immediate problem posed by this quest is source selection, i.e. selecting relevant and trustworthy sources to answer a query. Past approaches to this problem depended on purely query based measures to assess the relevance of a source. The relevance assessment based solely on query similarity is easily tampered by the content owner, as the measure is insensitive to the popularity and trustworthiness of the results. The sheer number and uncontrolled nature of the sources in the deep web leads to significant variability among the sources, and necessitates a more robust measure of relevance sensitive to source popularity and trustworthiness. To this end, we proposed SourceRank, a global measure derived solely from the degree of agreement between the results returned by individual sources. SourceRank plays a role akin to PageRank but for data sources. Unlike PageRank however, it is derived from implicit endorsement (measured in terms of agreement) rather than from explicit hyperlinks. For added robustness of the ranking, we assess and compensate for the source collusion while computing the agreements. Our comprehensive empirical evaluation shows that SourceRank improves relevance sources selected compared to existing methods and effectively removes corrupted sources. We also demonstrated that combining SourceRank with Google Product search ranking significantly improves the quality of the results.

1. A Survey on **“MODEL-BASED RICH INTERNET APPLICATIONS CRAWLING: “MENU” AND “PROBABILITY” MODELS”**

**Abstract**

Strategies for “crawling” Web sites efficiently have been described more than a decade ago. Since then, Web applications have come a long way both in terms of adoption to provide information and services and in terms of technologies to develop them. With the emergence of richer and more advanced technologies such as AJAX, “Rich Internet Applications” (RIAs) have become more interactive, more responsive and generally more user friendly. Unfortunately, we have also lost our ability to crawl them.

Building models of applications automatically is important not only for indexing content, but also to do automated testing, automated security assessments, automated accessibility assessment and in general to use software engineering tools. We must regain our ability to efficiently construct models for these RIAs. In this paper, we present two methods, based on “Model-Based Crawling” (MBC) first introduced in [1]: the “menu” model and the “probability” model. These two methods are shown to be more effective at extracting models than previously published methods, and are much simpler to implement than previous models for MBC. A distributed implementation of the probability model is also discussed. We compare these methods and others against a set of experimental and “real” RIAs, showing that in our experiments, these methods find the set of client states faster than other approaches, and often finish the crawl faster as well.

**Conclusion**

Two new model-based crawling algorithms were introduced: the Menu model and the Probability model. We models RIAs as a graph, based on the JavaScript events in each state of the DOM. The approach consists of making assumptions about the category of events in order to derive a strategy, then learn, and adapt these categories as the crawling proceeds. In the Menu model, the events are assumed to either always lead to the same state (menu events), or to always lead to new states. In the probability strategy, an actual probability of leading to a new state is calculated for each event, based on how successful it was in the past at finding new states. A concurrent implementation of the probability model was also described. A prototype implementation for these strategies was built and the performance of these strategies is compared with several other algorithms. We have shown empirically that our new strategies are better than other known strategies when it comes to finding all the states of the application being modeled. We have also evidences showing that the concurrent implementation of the probability strategy provides an good additional speed up, opening the door to fast crawling of RIA using concurrent crawlers.

Some directions for future studies are:

* Better definition of states: Learning about independent widgets, as suggested by [24], can substantially improve performance of both Menu and Probabilistic model by reducing the number of events to execute.
  + Adaptive Crawling: starting from a generic meta-model and recognizing the meta model of a web-application on-the-fly is another area of improvement.
  + Relaxing assumptions: In future work, it will be important to relax or waive some of the assumptions made in regard to this work. In particular the assumptions that the applications is deterministic from the viewpoint of the model builder.

1. A Survey on **“Optimal Algorithms for Crawling a Hidden Database in the Web”**

**Abstract**

A hidden database refers to a dataset that an organization makes accessible on the web by allowing users to issue queries through a search interface. In other words, data acquisition from such a source is not by following static hyper-links. Instead, data are obtained by querying the interface, and reading the result page dynamically generated. This, with other facts such as the interface may answer a query only partially, has prevented hidden databases from being crawled effectively by existing search engines.

This paper remedies the problem by giving algorithms to extract all the tuples from a hidden database. Our algorithms are provably efficient, namely, they accomplish the task by performing only a small number of queries, even in the worst case. We also establish theoretical results indicating that these algorithms are asymptotically optimal – i.e., it is impossible to improve their efficiency by more than a constant factor. The derivation of our upper and lower bound results reveals significant insight into the characteristics of the underlying problem. Extensive experiments confirm the proposed techniques work very well on all the real datasets examined.

**Conclusion**

Currently, search engines cannot effectively index hidden databases, and are thus unable to direct queries to the relevant data in those repositories. With the rapid growth in the amount of such hidden data, this problem has severely limited the scope of information accessible to ordinary Internet users. In this paper, we attacked an issue that lies at the heart of the problem, namely, how to crawl a hidden database in its entirety with the smallest cost. We have developed algorithms for solving the problem when the underlying dataset has only numeric attributes, only categorical attributes, or both. All our algorithms are asymptotically optimal, i.e., none of them can be improved by more than constant times in the worst case. Our theoretical analysis has also revealed the factors that determine the hardness of the problem, as well as how much influence each of those factors has on the hardness.