An Automatic Approach to Construct Domain-Specific Web Portals

Ismail Sengor Altingovde[1], Rifat Ozcan[1], Suleyman Cetintas[2], Hakan Yilmaz[2], Özgür Ulusoy[1]
Department of Computer Engineering, Bilkent University, 06800, Ankara, Turkey
[1]{ismaila, rozcan, oulusoy}@cs.bilkent.edu.tr, [2]{sule, yilmazh}@ug.bilkent.edu.tr

ABSTRACT
We describe the architecture of an automatic domain-specific Web portal construction system. The system has three major components: i) a focused crawler that collects the domain-specific pages on the Web, ii) an information extraction engine that extracts useful fields from these Web pages, and iii) a query engine that allows both typical keyword based queries on the pages and advanced queries on the extracted data fields. We present a prototype system that works for the course homepages domain on the Web. A user study with the prototype system shows that our approach produces high quality results and achieves better precision figures than the typical keyword based search.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *search process, query formulation*

General Terms
Experimentation, Design.

Keywords
Focused Crawling, Information Extraction, Querying.

1. INTRODUCTION
With the very fast growth of WWW, the quest for locating the most relevant answers to users' information needs becomes more challenging. In addition to general purpose Web directories and search engines, several domain specific Web portals or search engines also exist, which essentially aim to cover a specific domain/topic (e.g., education), product/material (e.g., product search for shopping), geographic region (e.g., transportation, hotels etc. at a particular country [2]) or media/file type (e.g., mp3 files or personal homepages [9]).

Such specialized search tools may be constructed manually (by also benefiting from possible assistance of the domain experts) or automatically. Some examples of the automatic approaches simply rely on intelligent combination and ranking of results obtained from traditional search tools (just like meta search engines), whereas some others first attempt to gather the domain specific portion of the Web using focused crawling techniques and then apply several post processing operations (i.e., information extraction, integration, etc.) on this collection (e.g., [7]). In [10], focused crawling is used for obtaining high quality pages on a mental health topic (depression) and constructing a Web portal. In [7], a prototype system is constructed that achieves focused crawling and multilingual information extraction on the laptop and job offers domains.

In this paper, we describe the architecture of a domain-specific Web portal construction system. This system will enlarge and refresh its collection by periodic focused crawls, and allow simple keyword based search on the raw Web pages as well as the advanced queries on the structured data obtained by using information extraction techniques.

As a prototype, we present Course Homepage Finder which is a Web portal for course homepages. To our knowledge, there is no automatically built portal that aims to maintain and query course homepages for various departments/majors, although several manually maintained pointers to course related resources may exist. We believe that such a system would be of great value for academicians, instructors and teaching assistants who want to survey similar course contents while preparing the course materials as well as the students who want to reach additional material for studying. We conduct a user study with the prototype system and show that the advanced queries supported by our approach produce higher quality results than the typical keyword based queries.

2. SYSTEM OVERVIEW
The overall architecture of the system is depicted in Figure 1. In what follows, we briefly describe each stage of the proposed system.

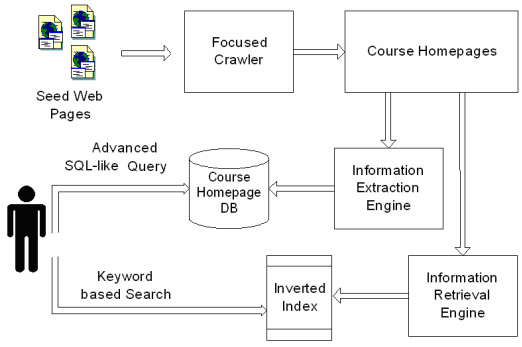

Figure 1. System architecture.

Focused Crawling. In this study, we first implemented the baseline focused crawler described in [3]. In particular, the crawler involves a document classifier which is trained with

CIKM'07, November 6-8, 2007, Lisboa, Portugal.

example Web pages for each topic in the system. The target topic is given by the user as one or more of these training topics. During the crawling, the crawler maintains a priority queue of URLs to be visited. A URL score is computed according to the classifier score indicating the relevancy of the target topic to the page from which the URL is extracted.

In addition to the above approach, two recent methods that exploit link context information are also explored [6], as well as a third one that we propose. In the first approach, so called text-window, only a number of words around each hyperlink are used for determining the priority of that link. The second one, tag-tree heuristic, uses the words that are in the DOM (Document Object Model) tree immediately in the node that a link appears, or its parents, until a threshold is satisfied [6]. Finally, we propose a similar but slightly different technique, so-called page segmentation method, which fragments a Web page according to the use of HTML tags. More specifically, for each tag, we define rules to decide whenever it constitutes to a semantic group. For instance, each (,) tag pair including more than, say 50 words, is a group. The list of some rules to determine the page segments in a Web page is given in Table 1. The segments are obtained in a recursive manner starting from the inner-most segment. Our intuition is that segmentation of a page would yield a set of coarse structural groups that are probably related to a single topic. The coarseness is intended to provide the classifier adequate evidence to decide a segment's class, in contrast to the fine grain approaches, which provide only some words around each link. Figure 2 shows an example faculty web page that is segmented by our approach. Each page segment is shown by rectangular blocks.

Table 1. Use of HTML tags for page segmentation method.

Explanation & usage	Semantic group
Unordered list: <UL> <LI>* </UL>.	If the total length of list items is less than a predefined threshold, the entire text between the <UL> (or <OL>) tags define a segment; otherwise each list item defines a segment on its own.
Ordered list: <OL> <LI>* </OL>.	
Paragraph: <P></P>	If the total length of the text between <P> tags is greater than a predefined threshold, the entire text in-between defines a segment. Otherwise, the text between these tags is combined with the first succeeding segment in the page.
Table: <TABLE><TBODY> (<TR><TD></TD> </TR>)* </TBODY></TABLE>	A table can be used for organizing the content or providing a neat presentation for the Web page. Therefore, if the length of the text in rows and columns is less than a predefined threshold and the cells do not include any of the above tags, the entire table defines a segment. Otherwise, each row, column or even cell may define a segment.
<FRAME>	Each frame is considered as a separate web page and further segmented by using the above rules.

Information Extraction. In our IE engine, we employ two different approaches and merge their results. The SRV (Sequence Rules with Validation) algorithm [4] based on a top-down machine learning technique, and Sanner's IE software [8] based on the Hidden Markov Model are adapted to our framework.

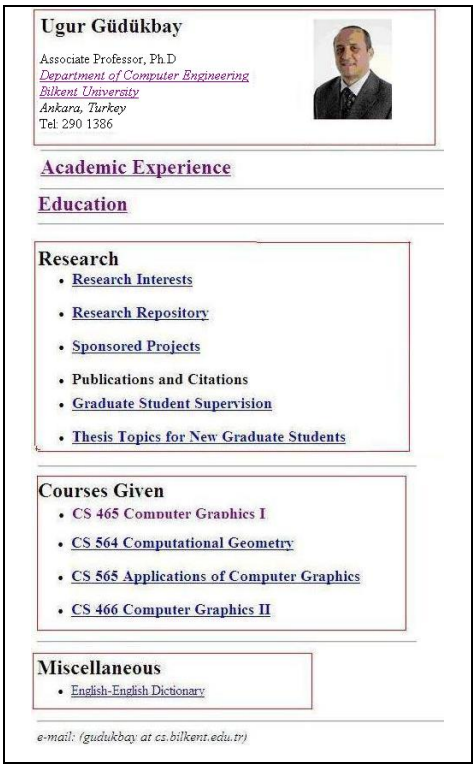

Figure 2. Segmentation of an example web page

SRV is a machine learning algorithm that learns first order logic extraction pattern rules from training examples. Training set consists of annotated HTML pages. Each learned extraction rule consists of predicates. Rule learning phase starts with an empty rule and it greedily adds predicates to the rule which achieves the most information gain by covering as many positive examples as possible and at the same time as few negative examples as possible. Each learned rule covers different positive examples in the training set.

Sanner's IE software is based on the algorithm given in [5]. The software learns three different HMMs using the same training data. Each of these HMMs has different state transition structures. These HMMs are combined by a weighted sum of their probability values.

Searching and Querying. The system provides two basic methods to retrieve relevant information to a user query.

- Inverted index based keyword searching: An inverted index of terms included in the Web pages is used to answer the keyword based searches. The Information Retrieval (IR) engine assigns term weights by using the vector space model and TF-IDF weighting scheme (see [12] for details). The similarity comparisons between the pages and user queries are computed by the cosine measure [12].
- SQL-like advanced querying: In this case, the users can specify search values for one or more of the fields that are

populated during the information extraction stage. The final query is constructed as a typical SQL query and sent to the underlying database system. We envision that advanced queries over the structured fields would significantly facilitate reaching relevant information with respect to the keyword based search on the entire crawl data. The validity of this expectation is evaluated by a user study as discussed in the next section.

3. PROTOTYPE SYSTEM

Dataset. For the prototype system, we only concentrate on the Computer Science (CS) course homepages and use a collection of 25,614 pages obtained from 4 Turkish university Web sites (namely Bilkent, METU, Bogazici and Koc). For this dataset, freely available crawlers are seeded with the university homepages and the crawls are restricted to .edu domain. Note that, we prefer to evaluate crawler strategies by using offline datasets, to avoid fluctuations of the online crawling with several parallel threads.

Crawling Stage. For the classifier component of the focused crawler, we use an SVM package, namely libSVM (http://www.csie.ntu.edu.tw/~cjlin/libsvm/), as discussed in [6]. The classifier is trained with the WebKB dataset (www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/). This dataset has been manually constructed from the CS departments of various universities and includes seven classes (topics), namely, course, student, staff, project, faculty, department and other, which constitute to 8,282 pages in total. As mentioned before, "course" topic is set to be the target topic.

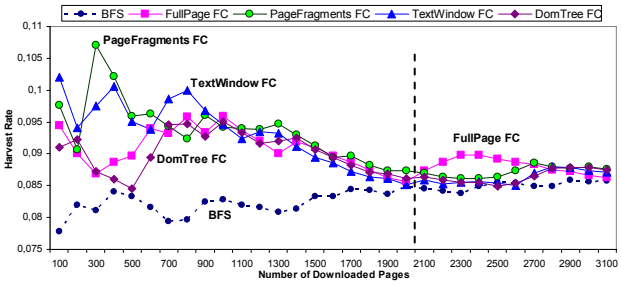

Figure 3. Harvest ratios for crawling strategies.

In Figure 3, we evaluate several crawlers on the pages collected from Bilkent University Web site. In particular, we compare the performance of breadth-first crawler (BFS), and four different focused crawler (FC) strategies, which assign URL scores according to either the full page content, or the context defined by the text-window, tag-tree or page segments. For the context-sensitive approaches, the final score of a URL is computed as a linear combination of the full page score and context score as computed by the corresponding strategy. Following the practice in [6], we use 25% of the full page score and 75% of the context score as the final score of a URL. The plot shows the harvest rate, i.e., the ratio of the number of relevant pages to the number of crawled pages at each point. Note that, as all strategies will eventually converge to the same harvest rate for the entire dataset, the performance at the beginning of the crawl (shown with dots) would be more informative for comparison purposes.

The results reveal that, BFS, the baseline strategy, is inferior to all focused approaches. Furthermore, the findings observed in earlier works are confirmed in that combining context information usually improves focused crawler performance and especially the text-window strategy may improve the harvest rate [6]. Finally, the page segmentation based approach proposed in this paper is also shown to be a worthwhile strategy as it is better than or comparable to the text window approach.

During the focused crawling stage, the pages with the classifier score 1.0 are decided to be the course homepages and stored to be fed to the IE component. We also manually determined CS related course pages and used them to evaluate the precision and recall for the data passed to the IE engine. For our dataset, 1084 pages are assigned the score 1.0 by the classifier, which constitutes the 4.2% of the entire collection. Our findings show that, for this dataset, the average precision and recall are 0.37 and 0.54, respectively.

Information Extraction Stage. The IE-engine is trained with a subset of WebKB dataset (including 900 annotated course pages) to extract the following fields: course id's, course names, semester, instructors' names and emails. The extracted data is stored in a relational DBMS, as shown in Figure 1. We manually evaluated the success of the extraction process for CS related course pages. Since user study experiments are performed by searching for course homepages using course name field, the performance of IE-engine is evaluated only for course name extraction. While SRV system achieves 0.28 precision and 0.43 recall on course name field, Sanner's IE software has 0.48 precision and 0.71 recall figures for the same field.

Querying stage. The prototype system (available at http://139.179.21.106/~ismaila/seniorTR/index.php) supports both keyword-based searches and SQL-like advanced queries on the extracted fields. In particular, the keyword-based search uses an inverted index over all crawled dataset including 25K pages, whereas the SQL-like system uses the fields extracted from the 1084 pages that are decided to be course pages, as described above. The user interface for Course Homepage Finder is shown in Figure 4.

User study. A user study including a group of senior and graduate CS students is conducted to measure the user satisfaction for the advanced and keyword-based querying approaches. In particular, a random set of 60 course names are determined from a disjoint dataset and each user is assigned 5 of them. Each user is required to enter the given query and evaluate the resulting records as relevant or irrelevant. Figure 5 shows the user interface that is used for the relevance judgment part of the user study. The results from the keyword-based and advanced querying approaches are shuffled, to prevent any bias. Next, for each query, we compute the precision score for the number of documents that are returned by the advanced approach. This is necessary, since the SQL-like advanced approach may return a restricted number of results per query whereas keyword-based query may return much more pages (due to keyword similarity based ranking). For this case, the precision of the advanced system is 66%, whereas the keyword-based system achieves 57%. Note that, comparing recall is not possible since it is impossible to know the all relevant pages in the entire dataset. (The Web site for the user study is available at http://139.179.21.106/~ismaila/SENIOR-1-TR/login.html)

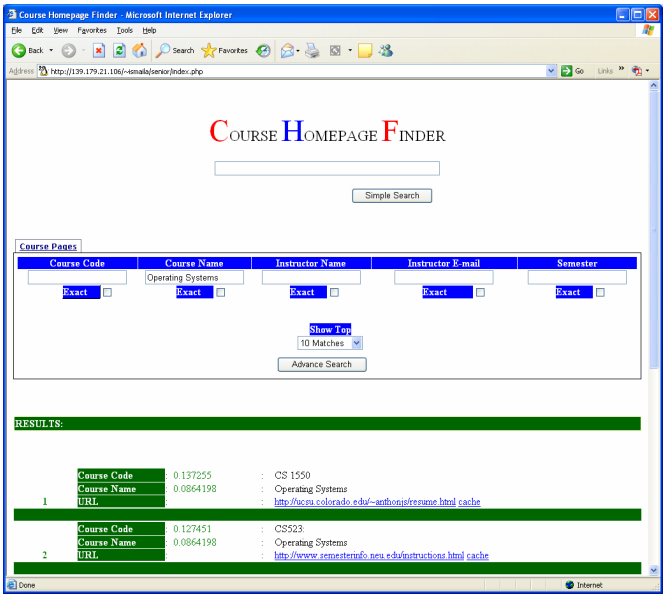

Figure 4. Course Homepage Finder user interface.

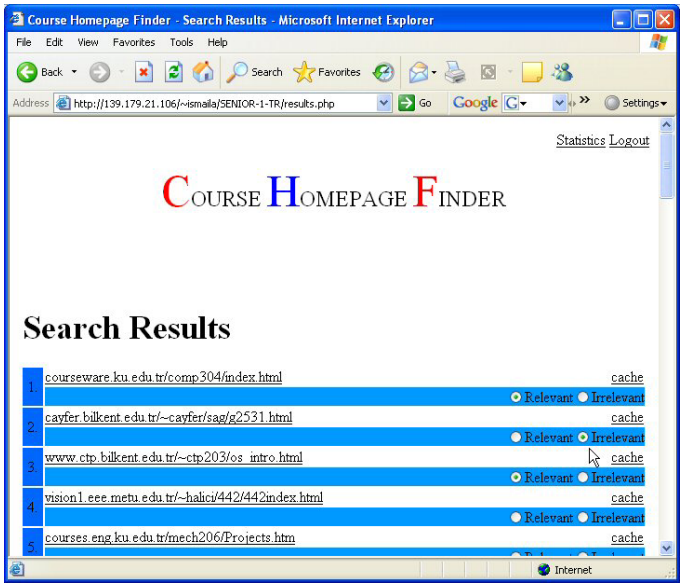

Figure 5. Relevance judgment interface for the user study.

4. CONCLUSION AND FUTURE WORK

In this study, we present an automatic approach to create domain-specific Web-portals and show its usability by conducting a user study on the prototype system constructed for course homepages.

Future work involves extensive experimentation with the proposed focused crawling methods, as well as adapting some other Web page segmentation methods in the literature (e.g., the VIPS algorithm, which segments Web pages based on the visual clues [1], or the method proposed in [11]). The effectiveness of IE component will also be further investigated, giving special emphasis on adaptive methods where only the most similar instances to a test instance in the train set are used for IE purposes. Finally, we plan to combine results from keyword-based and advanced-querying components, to further increase user satisfaction.

5. ACKNOWLEDGMENTS

This work is supported by The Scientific and Technical Research Council of Turkey (TÜBİTAK) under the grant no 105E024. We also would like to thank P. Angin, L. Ak, B. Atikoglu, A. Boynuegri, Ö. R. Atay, O. Ö. Dolu, İ. Durmaz, E. Karaca, T. Yıldız, E. Kucukoguz, A. Türk and E. Karaca for their help during the implementation.

6. REFERENCES

[1] Cai, D., Yu, S., Wen, J.-R., and Ma, W.-Y. Extracting Content Structure for Web Pages based on Visual Representation. In *Proc. of the Fifth Asia Pacific Web Conference (APWeb2003)*, 2003.

[2] Cambazoglu, B. B., Karaca, E., Kucukyilmaz, T., Turk, A. and Aykanat, C. Architecture of a Grid-Enabled Search Engine. *Information Processing & Management, 43*, 3 (May 2007), 609–623.

[3] Chakrabarti, S. *Mining the Web Discovering Knowledge from Hypertext Data*. MK. Publishers, San Francisco, 2003.

[4] Freitag, D. *Machine Learning for Information Extraction in Informal Domains*. Ph.D. Thesis, Carnegie Mellon University, PA, 1998.

[5] Freitag, D. and McCallum, A. Information Extraction with HMM's and Shrinkage. In *Proc. of AAAI '99 Workshop on Machine Learning for Information Extraction*, 1999.

[6] Pant, G. and Srinivasan, P. Link Contexts in Classifier-Guided Topical Crawlers. *IEEE TKDE, 18,* 1 (Jan. 2006), 107-122.

[7] Pazienza, M. T., Stellato A., and Vindigni, M. Purchasing the Web: an Agent based E-retail System with Multilingual Knowledge. In *Proc. of WI2003 Workshop on Applications, Products and Services of Web-based Support Systems*, Halifax, Canada, 2003.

[8] Sanner's HMM-based Text Mining and Extraction Tool. Available at www.cs.toronto.edu/~ssanner/software.html

[9] Shakes, J., Langheinrich, M. and Etzioni, O. Dynamic Reference Sifting: A Case Study in the Homepage Domain. In *Proc. of WWW*6, Santa Clara, CA, 1997, 189-200.

[10] Tang, T., Hawking, D., Craswell, N., Griffiths, K. Focused crawling for both relevance and quality of medical information. In *Proc. of CIKM 2005*, 2005, 147-154.

[11] Vadrevu, S., Gelgi F., Davulcu, H. Semantic Partitioning Web Pages. *World Wide Web, Internet and Web Information Systems (WWWJ)*, Springer, 2006.

[12] Witten, I. H., Moffat, A. and Bell, T. C. *Managing Gigabytes Compressing and Indexing Documents and Images*, Van Nostrand Reinhold, New York, 1994.