Knowledge Agents on the Web

Yariv Aridor[1], David Carmel[1], Ronny Lempel[2], Aya Soffer[1] and Yoelle S. Maarek[1]

[1] IBM Haifa Research Laboratory
MATAM, Haifa 31905, Israel
{yariv,carmel,ayas,yoelle}@il.ibm.com
[2] Computer Science Department, Technion, Haifa, Israel
rlempel@cs.technion.ac.il

Abstract. This paper introduces and evaluates a new paradigm, called Knowledge Agents, that incorporates agent technology into the process of domain-specific Web search. An agent is situated between the user and a search engine. It specializes in a specific domain by extracting characteristic information from search results. Domains are thus user-defined and can be of any granularity and specialty. This information is saved in a knowledge base and used in future searches. Queries are refined by the agent based on its domain-specific knowledge and the refined queries are sent to general purpose search engines. The search results are ranked based on the agent's domain specific knowledge, thus filtering out pages which match the query but are irrelevant to the domain. A topological search of the Web for additional relevant sites is conducted from a domain-specific perspective. The combination of a broad search of the entire Web with domain-specific textual and topological scoring of results, enables the knowledge agent to find the most relevant documents for a given query within a domain of interest. The knowledge acquired by the agent is continuously updated and persistently stored thus users can benefit from search results of others in common domains.

1 Introduction

The amount of information available on the World Wide Web is increasing on a daily basis. General purpose search engines and browsers provide valuable assistance to users in locating general information relevant to their needs. However, finding information for a narrow query in a specific domain has become more and more difficult with the growth of the Web, and thus frequently resembles a search for a needle in a haystack.

Individuals spend more and more time filtering out irrelevant information returned from general purpose search engines while searching the Web. It is not uncommon for a user to obtain thousands of hits which match her query but belong to irrelevant domains. This is termed the low precision problem, as defined in information retrieval [15].

Many approaches have been suggested to tackle the low precision problem on the Web [3, 7, 1, 5]. One such approach is to restrict the search into a pre-defined domain. Several search services, most notably Yahoo! [8], allow users to specify the domain in which to evaluate the query. Such a restriction narrows the search space, hence increases the precision. However, the domain hierarchy is manually (or at best semi manually) crafted as a taxonomy of predefined categories and users cannot request personal domains of interest. While this approach guarantees more quality, browsing is often time consuming, and coverage is extremely limited. Major search engines, such as AltaVista, index about two orders of magnitude more Web pages than Yahoo! [16].

This paper presents a new paradigm, termed Knowledge Agents, which provides domain-specific search in the context of dynamic domains. With knowledge agents, domains are defined by the users and can thus be of any granularity and specialty. In essence, it is an architecture which enables knowledge acquisition from search results which automatically characterizes the domain in which the search was applied. This knowledge is persistently saved by the agent and can then be utilized to automatically narrow future searches within that domain.

The rest of the paper is organized as follows. Section 2 highlights the knowledge agent approach. Section 3 describes the system architecture. Section 4 provides some examples and experimental results. Section 5 discusses related work. Section 6 concludes the paper, highlighting future work and challenges.

2 Knowledge Agent Main Approach

Knowledge agents improve search precision by mimicking the steps that users might perform to find what they are really looking for. These steps include

- Choose a search engine and submit a query.
- Traverse the list of retrieved pages to find the relevant ones.
- Apply shallow browsing based on outgoing hyperlinks from the set of retrieved pages.
- Provide relevance feedback for “more like this” services.
- Refine the query repeatedly and resubmit it (possibly to other search engines).

The key to the knowledge agent (KA) approach is that it specializes in a domain by extracting relevant information every time it performs a search and uses this knowledge to improve the precision of subsequent search efforts. To this end, the KA maintains a knowledge base (KB) that stores this information persistently. The KB consists of a set of leading sites in its domain and a repository of frequent terms in these sites. Each term is associated with a list of lexical affinities – closely related terms frequently found in its proximity [12]. The KB is adapted continuously by the agent during search. New highly relevant pages found by the agent are entered into the KB, possibly taking the place of old pages with lower utility. The KB can be initialized by providing a set of sites relevant to the domain of interest. For example, this set could be extracted from the user’s bookmark file or from any other existing pre-defined categorization of Web sites.

The first role the KA performs on behalf of the user is query refinement, which has long been recognized as an efficient tool for improving search results [17]. Query refinement is usually performed by adding terms related to the user’s terms using a thesaurus or a synonym table. The KA, on the other hand, expands the query by adding to each of the terms its most notable lexical affinities as found in the KB. The advantage of this approach is that the agent’s local thesaurus characterizes its domain-specific ontology and thus relations between terms are domain dependent. For example, consider a search for the query “knowledge”. An “artificial intelligence” agent would likely expand the query to include the terms “acquisition”, “reasoning”, “discovery”, “representation”, while a “cryptographic” agent would likely expand the query using the terms “zero”, “private”, etc.

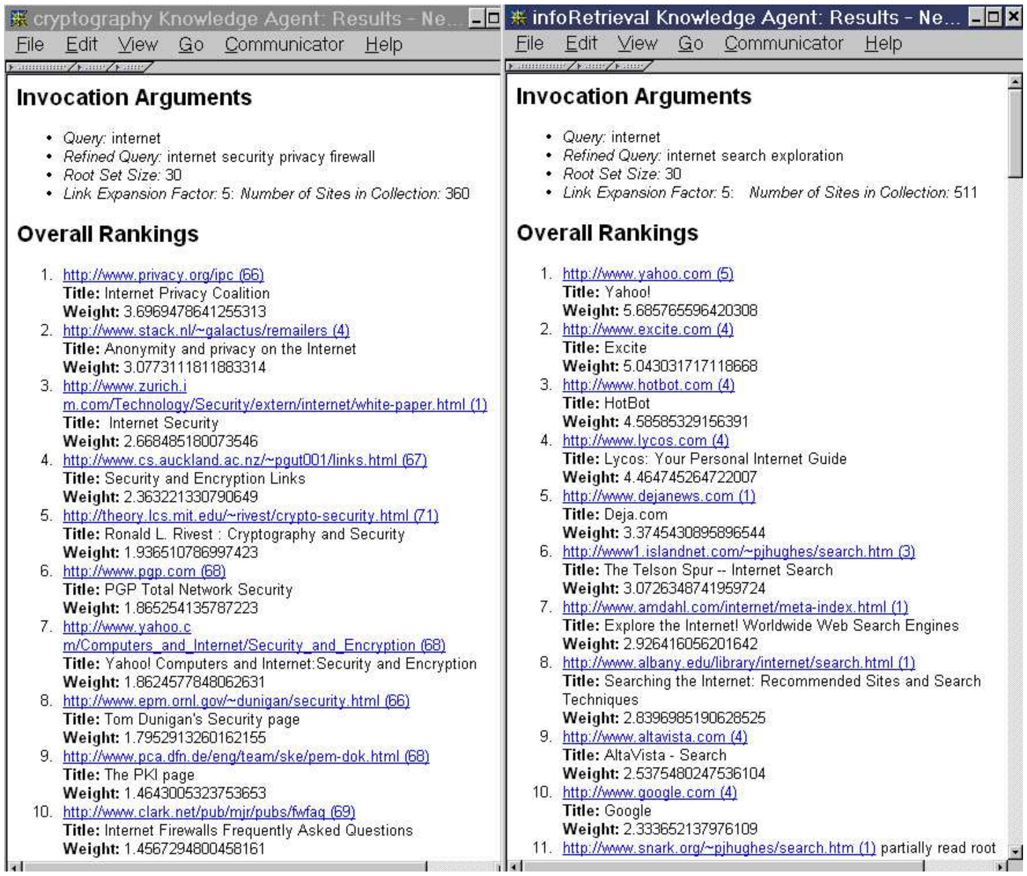
The second role that the KA performs on behalf of the user is shallow Web crawling. The agent applies a topological search mechanism similar to the one applied by Clever [3]. It first compiles a list of candidate result pages (root set) by sending the refined query to one or several search engines. This basic set is extended to include pages that are pointed to by pages in this set based on their potential relevance to the query. Pages that point to pages in the root set are also added to the retrieved set in an attempt to find higher level pages about the topic of interest. Finally, the pages saved in the KB, which are assumed to be the most authoritative information sources for the given domain, are added to the retrieved set.

The third role of the KA is traversing the retrieved pages and ranking them such that the most relevant pages will be listed first in the result. Ranking is performed based on both textual and topological aspects, utilizing information stored in its KB. The textual similarity measures the relevance of the pages retrieved to the specific query as well as to the agent’s domain. The link topology score is computed using a combination of Kleinberg’s mutual reinforcement algorithm [10] as well as stochastic link analysis [11].

It follows that a knowledge agent retrieves the most relevant sites according to its personal point of view of its domain of specialization. Figure 1 is an example of this behavior. It shows the result of submitting the query “internet” to a cryptography (Crypto) agent and an information retrieval (IR) agent. The Crypto agent refined the query to include the terms “security privacy firewall”, while the IR agent added the terms “search exploration”. Each agent viewed the term “internet” in the context of its domain of expertise. The IR agent retrieved the main sites of the leading Internet search engines, while the Crypto agent returned sites dealing with Internet privacy and security.

To summarize, the main highlights of the knowledge agent approach are:

– Personalization: the user creates and maintains knowledge agents for his private domains of interest rather than being dependent on a fixed set of domains. These domains of interest can be of any granularity.
– Persistent knowledge: knowledge agents make it possible to utilize knowledge gained through search to improve search precision of future queries in the same domain.
– Global search: the knowledge agent searches the entire Web rather than a subset as in the case of domain-specific search engines.
– Portability: users can easily import knowledge agents created by others to assist in their search in any common domains of interests since the KB is implemented as a plug-in component of the knowledge agent software.
– Easy deployment: agents can be implemented as a front end to any search engine. They do not impose any extra functional overhead on the user compared to available search engines.


cryptography Knowledge Agent: Results - Ne...
File Edit View Go Communicator Help
infoRetrieval Knowledge Agent: Results - Ne...
File Edit View Go Communicator Help

Invocation Arguments

- *Query:* internet
- *Refined Query:* internet security privacy firewall
- *Root Set Size:* 30
- *Link Expansion Factor:* 5: *Number of Sites in Collection:* 360

Overall Rankings

1. http://www.privacy.org/ipc (66)
   Title: Internet Privacy Coalition
   Weight: 3.6969478641255313
2. http://www.stack.nl/~galactus/remailers (4)
   Title: Anonymity and privacy on the Internet
   Weight: 3.0773111811883314
3. http://www.zurich.i
   m.com/Technology/Security/extern/internet/white-paper.html (1)
   Title: Internet Security
   Weight: 2.668485180073546
4. http://www.cs.auckland.ac.nz/~pgut001/links.html (67)
   Title: Security and Encryption Links
   Weight: 2.363221330790649
5. http://theory.lcs.mit.edu/~rivest/crypto-security.html (71)
   Title: Ronald L. Rivest : Cryptography and Security
   Weight: 1.936510786997423
6. http://www.pgp.com (68)
   Title: PGP Total Network Security
   Weight: 1.865254135787223
7. http://www.yahoo.c
   m/Computers_and_Internet/Security_and_Encryption (68)
   Title: Yahoo! Computers and Internet:Security and Encryption
   Weight: 1.8624577848062631
8. http://www.epm.ornl.gov/~dunigan/security.html (66)
   Title: Tom Dunigan's Security page
   Weight: 1.7952913260162155
9. http://www.pca.dfn.de/eng/team/ske/pem-dok.html (68)
   Title: The PKI page
   Weight: 1.4643005323753653
10. http://www.clark.net/pub/mjr/pubs/fwfaq (69)
   Title: Internet Firewalls Frequently Asked Questions
   Weight: 1.4567294800458161

Invocation Arguments

- *Query:* internet
- *Refined Query:* internet search exploration
- *Root Set Size:* 30
- *Link Expansion Factor:* 5: *Number of Sites in Collection:* 511

Overall Rankings

1. http://www.yahoo.com (5)
   Title: Yahoo!
   Weight: 5.685765596420308
2. http://www.excite.com (4)
   Title: Excite
   Weight: 5.043031717118668
3. http://www.hotbot.com (4)
   Title: HotBot
   Weight: 4.58585329156391
4. http://www.lycos.com (4)
   Title: Lycos: Your Personal Internet Guide
   Weight: 4.464745264722007
5. http://www.dejanews.com (1)
   Title: Deja.com
   Weight: 3.3745430895896544
6. http://www1.islandnet.com/~pjhughes/search.htm (3)
   Title: The Telson Spur -- Internet Search
   Weight: 3.0726348741959724
7. http://www.amdahl.com/internet/meta-index.html (1)
   Title: Explore the Internet! Worldwide Web Search Engines
   Weight: 2.926416056201642
8. http://www.albany.edu/library/internet/search.html (1)
   Title: Searching the Internet: Recommended Sites and Search Techniques
   Weight: 2.8396985190628525
9. http://www.altavista.com (4)
   Title: AltaVista - Search
   Weight: 2.5375480247536104
10. http://www.google.com (4)
   Title: Google
   Weight: 2.333652137976109
11. http://www.snark.org/~pjhughes/search.htm (1) partially read root

Fig. 1. The results for the query "internet" using two different knowledge agents.

3 System Architecture

The knowledge agent architecture contains the following components (Figure 2):

1. An agent manager, that sets up new knowledge agents and restarts existing agents. It is responsible for managing multiple concurrent connections for reading from the Web and serve multiple agents.
2. One or more Knowledge Agents which perform domain-specific Web search.
3. Each knowledge agent has an associated knowledge base which contains domain-specific information to be used for searching. The KB is updated continuously throughout the use of the agent.

3.1 The Knowledge Base

The knowledge base contains a bounded collection of ranked sites and an aggregate profile of the textual content of these sites. Sites in the knowledge base are those which

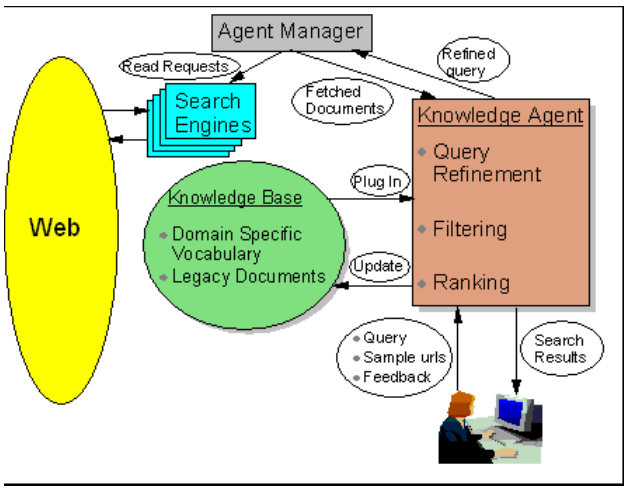


Fig. 2. The knowledge agent architecture.

have proven to be highly relevant to a majority of the queries submitted to the agent. The rationale for this is that sites which are consistently relevant to the users' queries are deemed as central to the domain.

The textual profile contains all of the words which appear in the sites, after deletion of stop words and a mild stemming process, along with their number of appearances. Each word in the profile is associated with a list of its lexical affinities. The flow of sites into and out of the KB is regulated using an evolutionary adaptation mechanism. Sites fight for the right to be included in the agent's KB. Each site is assigned a history score reflecting its relevance to the domain through the life of the agent. A combination of the history score and the relevance score for a specific query determines which sites are inserted and removed from the KB.

The KB is a pluggable component. It can be saved to a file and restored from one, and thus knowledge can easily be transferred from one user to another.

3.2 The Search Process

The search process (Figure 3) starts with the user entering a query and ends with the agent returning a ranked set of (hopefully highly relevant) sites to the user.

The system supports two kinds of queries, text queries and sample-url queries. A text query is a keyword based query such as those typically submitted to general purpose Web search engines. The user's query is automatically refined in the context of the agent's domain by adding to each of the keywords in the query its most notable lexical affinities as found in the profile of the KB. The refined query is submitted to the user's choice of one or more search engines. The results returned by the search engine(s) to the refined query are called the root set of sites.

A sample-url Query is a query which specifies a few (typically 1-5) seed urls, and whose purpose is to find a community of sites which are closely related to the seeds.

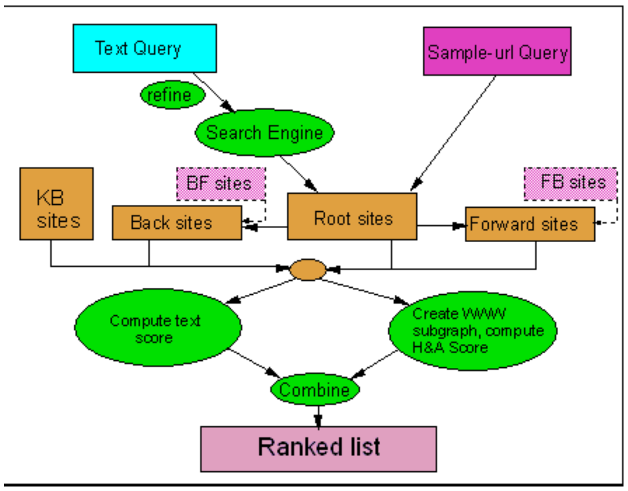


Fig. 3. The search process applied by the agent.

Similar services are offered by Excite's "More like this" [6] and by Google's "GoogleScout" [7]. Both services receive as input a single site, while we allow an arbitrary number of seeds. In sample-url queries, the user-supplied seed sites assume the role of the root set of sites, as if they were returned by a search engine in response to some textual query. The seed sites are read, and their combined content serves as a pseudo query for the purpose of evaluating the textual content of other sites in the search process.

The collection of root sites is expanded by following the hyperlinks surrounding them, and by adding the KB sites. The breadth of the expansion is controlled by a user-defined link expansion factor that specifies how many pointed/pointing sites will be added in each expansion stage. We denote the entire collection of sites by C.

3.3 Ranking the Web Pages

Computing the textual score of a web page. The agent receives a set of sites from the meta-search, from which it creates a ranked set of sites to be returned to the user. The agent computes a textual similarity score, $T_q(s)$, for each page s with respect to the query using a tf-idf formula originally used in the Guru search engine [12]. The textual profiles of the pages saved in the KB serve as the set of documents from which the terms' document frequencies are taken.

The agent also computes a textual similarity score, $T_d(s)$, of each site to the domain. $T_d(s)$ is set to the dot product of the vectors of lexical affinities representing s and the domain. Term weights are assigned in relation to their frequency in the domain. The rationale for this is that pages that have many lexical affinities in common with the domain are most related to the domain. $T_q(s)$ and $T_d(s)$ are normalized and combined to create the overall textual similarity score $T(s)$.

Computing the link topology score of a web page. The agent builds a Web subgraph, induced by the collection of sites C, on which connectivity analysis is performed in order to find authoritative Web sites. The idea behind connectivity analysis is that a hyperlink from a site s to a site t indicates that these two sites share a common topic of interest, and that s conveys a positive assessment on t's contents by recommending that surfers who visit s also visit t. Weights are first assigned to the edges of the Web subgraph. Each link receives a positive weight according to the anchor text associated with it. The weight is boosted if either the source site or the target site belong to the KB. This weighted Web subgraph is used to assign the hub and authority scores to each site from which a link topology score is derived.

Computing the overall score of a Web page. The textual score and the link topology score are combined to yield the overall score of each site s in C:

$$S(s) = \alpha_C T(s) + (1 - \alpha_C) L(s)$$

Link topology scores are reliable only for collections in which many neighboring sites have been added around the meta-search results. We therefore set the value of α_C according to the ratio between the size of the compiled collection C and the size of the root set. The larger that ratio, the more confidence we have in the link-based score, and the lower we set α_C. When the ratio is low, meaning that the link expansion phase did not add many sites, we raise the influence of the text-based scores by raising α_C.

4 Examples and Experiments

We have developed a prototype system that implements the KA architecture. We created several agents using our system: a palm pilot agent, a cryptography agent (Crypto), an artificial intelligence agent (AI), a Geographic Information System agent (GIS), an Information Retrieval agent (IR), and a Star Wars agent. We trained each agent by submitting several textual queries relevant to its domain. For example the IR agent was trained using the following queries: "information retrieval", "text mining", "information extraction", "query refinement", "vector space model", "probabilistic model", "recall precision". Each agent's KB includes 50 to 100 urls (depending on the agent setup) as well as a textual profile of these pages. In this section, we present some examples and experimental results using these agents. The experiments were conducted as a proof of concept for the KA main functionality. Clearly, these experiments do not replace a future more exhaustive study of the KA performance.

4.1 Query Refinement

To examine the query refinement capabilities of the agents, we selected a few queries and examined the terms added by each KA while refining these queries. We made the following observations. The agents can complete names of people in their fields. For example, the Crypto agent adds "Ron" and "MIT" when asked about "Rivest", while the AI agent adds "Stuart", "Norvig", and "aima", when asked about "Russel" (aima is an abbreviation for a famous introductory AI book written by Stuart Russel and Peter Norvig). They know about algorithms and theoretical aspects of their domains (e.g., the Crypto agent adds “Interactive” and “Proof” to “Zero Knowledge”, and adds “Stream” and “Block” to “Ciphers”). They even know the marketplace (given the term “Checkpoint”, the Crypto agent adds “firewall” and “vendor”). The agents are particularly good with acronyms. For example, the GIS agent expands “dlg” to “digital line graph data” and “eos” to “nasa earth observation system”.

Table 1 shows how different agents refine the same terms according to their domain-specific ontology. For example, the query “science” is expanded by the Crypto agent to include “computer” and “cryptography”, by the GIS agent to include “earth” and “information”, and by the Star Wars agent to include “fiction”.

	Palm Pilot	Cryptography	Geographic Information System (GIS)	Star Wars
software	development,download	encryption,security	esri (large software company)	game
knowledge		zero	opto systems	star war encyclopedia
public	license,gnu domain	key,x509,certificate	domain, data	free site
science		cryptography, computer	earth,information	fiction
Microsoft	windows,operating system	crypto api	terraserver (ms geographic data server)	
unit	battery,modem	rsa certificate trusted	map boundary, remote sensing	hyperdrive
video	driver	stream	gis library remote sensing	clip picture sound

Table 1. : query refinement performed by different agents.

We examined the precision of the agent’s textual analysis without link expansion. We compared the precision of the top results using KA with the precision of general purpose search engines for several queries. Each query was submitted to AltaVista and Google, as well as to a knowledge agent specializing in the domain of the query. The KA used AltaVista for meta-search in the first run and Google in the second. The relevance of each site in the top results was examined in the context of the domain of interest by an expert in the domain. Table 2 summarizes the results of these experiments.

From these results, it is apparent that performing textual analysis in the context of the domain of the specific KA improves the precision of the search significantly. The KA usually succeeds in sifting up relevant sites from the bottom of the meta-search results. For example, when submitting the query “landsat 7 sample image” to the GIS KA using AltaVista as the meta-search engine, the four most relevant sites, ranked 2, 5, 14, and 19 by AltaVista, were ranked 1- 4 by the KA. Two additional sites with numerous links to satellite images were extracted from the agent’s knowledge base and returned in the top 10 hits. The top 10 sites returned by Google were related to landsat 7, however only six of these sites had sample images or links to such images. When

		top@10				top@20			
Query	Agent	AV	KA+ AV	Google	KA+ Google	AV	KA+ AV	Google	KA+ Google
landsat 7 sample image	GIS	0.2	0.6	0.6	1	0.2	0.4	0.55	0.65
zero knowledge	Crypto	0.1	0.6	0.3	0.5	0.35	0.35	0.35	0.4
relevance feedback	IR	0.6	0.6	0.4	0.7	0.6	0.5	0.5	0.65

Table 2. Precision at 10 and 20 using various search engines and a knowledge agent with zero link expansion.

submitting the same query to the GIS agent using Google as the meta-search engine, all ten sites in the top 10 where relevant. The four non-relevant sites in Google's top 10 were replaced by four relevant sites from further down the list.

It is important to note that the agent's success in these examples is attributed to the fact that it was able to rank the sites more relevant to its particular domain higher. It may be the case, that by expanding the query "zero knowledge" to include the term "cryptography", the general purpose search engines would be more successful. However, it has been shown in [1] that for queries containing both broad domain terms and narrow terms general search engines tend to return sites relevant to the broad domain rather than sites dealing with the narrow topic within the context of the domain.

4.2 Link Expansion

In order to test the effect of link expansion we executed the same queries with a link expansion of 5 (every site in the root set contributes 5 sites which it points to and 5 sites which point to it) using AltaVista for the meta-search. We compare these results to executing the same queries with zero link expansion. Table 3 summarizes the results.

		top@10		top@20	
Query	Agent	No Exp.	Exp.	No Exp.	Exp.
landsat 7 sample image	GIS	0.4	0.8	0.2	0.55
zero knowledge	Crypto	0.6	0.4	0.35	0.5
relevance feedback	IR	0.6	0.7	0.5	0.6

Table 3. Precision at 10 and 20 using knowledge agents with zero link expansion and with link expansion 5.

In general, link expansion improved the results by finding additional relevant sites that were not returned by the meta-search but were either pointed to or pointed by these sites. For example, for the query "landsat 7 sample images", the GIS agent was able to retrieve several additional relevant sites compared to search with zero link expansion. Similar results were observed in the top 20. The improvement for the "relevance feedback" query was not as significant, since we already had good precision with zero link expansion. Nevertheless, we were able to add one more relevant site to the top 10 and two relevant sites to the top 20. Note that in the case of the query "zero knowledge" the precision at 10 decreased using link expansion since the agent retrieved two additional non-relevant sites and mistakenly ranked them in the top 10.

We tried to compare our results to those of Clever which also uses link expansion for retrieval. Clever was basically unable to find good sites for these queries. It returned some good IR hubs and authorities for the "relevance feedback" query, but was unable to locate even one site that contains this term. Similarly for the query "zero knowledge", it returned cryptography sites but none pertaining to zero knowledge. These results are not surprising since Clever is not meant for these type of narrow queries [4].

While link expansion is a powerful tool for improving precision, its drawback is that time complexity increases with the size of the expansion. The KA system lets the user choose the desired link expansion, thus giving them control over the tradeoff between response time and quality of results.

5 Related Approaches

Many approaches have been suggested to tackle the low precision problem involved in searching the Web. Hierarchical categorization of Web sites is the most common solution. It allows users to navigate through a hierarchy of categories and specify explicitly the domain in which to evaluate their query. This reduces the number of irrelevant results, hence increases the precision significantly. However, as mentioned above, Web categorization requires significant human effort and coverage is extremely limited.

Another alternative is using domain-specific search engines. Such engines allow users to perform a focused search on a given topic within the scope of the specific domain covered by the search engine. For example, MRQE [14] allows users to search only for movie reviews. CampSearch [2] enables complex queries over summer camps by age, size, location and cost. Performing these searches with a general purpose search engine would be extremely tedious and most likely not yield to such accurate results. However, as in the case of manual categorization, building such search engines is a labor intensive process.

The search broker [13] is a meta-search tool which utilizes existing domain-specific search engines. Users specify both the domain and the query, making it possible to apply a domain-specific search engine for their narrow query. Each search engine offered by the search broker covers a certain domain, and each domain is associated with a list of terms (aliases) related to it. The search broker chooses the most proper search engine for the narrow query according to the domain specified by the user. The collection of search engines and the assignment of aliases that describe each engine are maintained by a human librarian.

Focused crawling [5] attempts to automatically build domain-specific search engines by selectively seeking out pages that are relevant to a predefined domain. Rather than collecting and indexing all accessible Web documents, a focused crawler finds the links that are likely to be most relevant for the specified domain, and thus avoids irrelevant regions of the Web. Jcentral [9] is such a search engine which allows Java developers to search for Java sources on the Web.

The common feature of the aforementioned methods is focusing the search into a specific domain in order to improve search precision. The knowledge agents described in this work apply a similar strategy, however, they can automatically specialize in any domain as defined by their user and are therefore much more suitable for personal assistance. Furthermore, the knowledge acquired during the agent's activity is persistently kept by the agent and continually updated to enable improved search services in the future. Finally, the domain-specific textual profile maintained by the knowledge agent is a very powerful tool both for query refinement and for textual ranking of the relevancy of documents to a specific query within the domain.

6 Concluding Remarks

The knowledge agent approach presented in this paper suggests a new paradigm which provides domain-specific search in the context of dynamic domains. These domains are defined by the users and can thus be of any granularity and specialty. Queries are refined by the agent based on its domain-specific knowledge. The refined queries are sent to general purpose search engines and the results are ranked from the viewpoint of the specific knowledge agent, thus filtering out documents which match the query but are irrelevant to the domain of interest. A topological search of the web for additional relevant documents is conducted from a domain-specific perspective as well. The knowledge stored by the agent, enables it to enjoy the benefit of topological search, while traversing a relatively small search space. The combination of a broad search of the entire Web, using general purpose search engines, with domain-specific textual and topological scoring of results, enables knowledge agents to find the most relevant documents at search time for a given query within the realm of the domain of interest.

Knowledge agents is work in progress. We have implemented a prototype and conducted some initial experiments that show the potential benefits of this approach. There are however still several open questions. First and foremost is the question of how to best characterize a domain. Currently the domain is characterized by a list of leading sites and the entire textual content of these sites. Ideally, we would like to use only those terms that in fact distinguish this domain from others while filtering out irrelevant terms. Furthermore, sites found as most relevant to a particular query, might be entered automatically into the KB. While these sites should best characterize the domain, this is not always the case. A feedback mechanism whereby users could indicate the relevance of results to the domain as a whole can assist the learning process and improve the domain characterization. Finally, since the Web is constantly changing, letting the agent work autonomously off-line, looking for new sites similar to those already in its knowledge base, will probably be needed in order to keep the domain characterization up to-date. These are all subjects for future work.

In this paper we have described how knowledge agents can be used for Web search. We envision knowledge agents as light components that can be plugged into several other Web applications such as browsing, filtering and routing systems. The knowledge acquired by the agent while searching for information pertaining to a user's domain of interest, can assist in analyzing information acquired from other Internet sources from the point of view of this particular agent's domain.

Acknowledgments

We thank Dan Pelleg for useful discussions concerning the knowledge agent approach.

References

1. I. Ben-Shaul, M. Herscovici, M. Jacovi, Y. S. Maarek, D. Pelleg, M. Shtalhaim, V. Soroka, and S. Ur. Adding support for dynamic and focused search with fetuccino. In *Proceedings of the Eighth International WWW Conference*, pages 575–587. Elsevier, 1999.
2. CampSearch. The search engine for camps. http://www.campsearch.com.
3. IBM Almaden Research Center. Clever. http://www.almaden.ibm.com/cs/k53.clever.html.
4. S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, S.R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Mining the web's link structure. *IEEE Computer*, 32(8):60–67, August 1999.
5. S. Chakrabarti, B. Dom, and M. ven den Berg. Focused crawling: A new approach to topic-specific web resource discovery. In *Proceedings of the Eighth International WWW Conference*, pages 545–562. Elsevier, 1999.
6. Excite Inc. Excite search. http://www.excite.com/.
7. Google Inc. Google search engine. http://www.google.com/.
8. Yahoo Inc. Yahoo! http://www.yahoo.com.
9. IBM Jcentral. Search the web for java. http://www.jcentral.com.
10. J. M. Kleinberg. Authoritaive sources in a hyperlinked environment. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, volume 25-27, pages 668 – 677, January 1998.
11. Ronny Lempel. Finding authoritative sites on the WWW (and other hyperlinked media) by analyzing the web's link-structure. Master's thesis, Technion, Israel Institute of Technology, July 1999.
12. Y. Maarek and F. Smadja. Full text indexing based on lexical relations, an application: Software libraries. In N. Belkin and C. van Rijsbergen, editors, *Proceedings of SIGIR89*, pages 198 – 206. Cambridge MA, ACM press, 1989.
13. U. Manber and P. A. Bigot. The search broker. In *The First Usenix Symposium on Internet Technologies and Systems*, pages 231–240, Monterey CA, December 1997.
14. MRQE. Movie review query engine. http://www.mrqe.com.
15. G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* Computer Series, McGraw-Hill, New York, 1983.
16. Search Engine Watch. Search engine watch. http://www.searchenginewatch.com.
17. J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th annual international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 4 –11, 1996.