Large Scale Distributed Foraging, Gathering, and Matching for Information Retrieval: Assisting the Geospatial Intelligence Analyst

Eugene Santos Jr.[a], Eunice E. Santos[b], Hien Nguyen[a], Long Pan[b] and John Korah[b]

[a] Computer Science and Engineering Department, University of Connecticut
{eugene,hien}@cse.uconn.edu
[b] Department of Computer Science, Virginia Polytechnic Institute & State University
santos@cs.vt.edu,{panl,jkorah}@vt.edu

ABSTRACT

With the proliferation of online resources, there is an increasing need to effectively and efficiently retrieve data and knowledge from distributed geospatial databases. One of the key challenges of this problem is the fact that geospatial databases are usually large and dynamic. In this paper, we address this problem by developing a large scale distributed intelligent foraging, gathering and matching (I-FGM) framework for massive and dynamic information spaces. We assess the effectiveness of our approach by comparing a prototype I-FGM against two simple controls systems (randomized selection and partially intelligent systems). We designed and employed a medium-sized testbed to get an accurate measure of retrieval precision and recall for each system. The results obtained show that I-FGM retrieves relevant information more quickly than the two other control approaches.

Keywords: Geospatial information retrieval, dynamic information space, distributed information retrieval, parallel information retrieval, multi-agent systems, retrieval performance

1. INTRODUCTION

Retrieving geospatial information from a large, dynamic geospatial database is an important field of research because of the growing needs for accessing geospatial information by users such as intelligence analysts, especially in emergency situations which are critically hampered by the limitations of existing commercial search engine technologies. Geospatial information contains not only geometrics information such as shape, distance, measurements, geophysical structure, and images about a specific place, but also social information such as population, hotels, school rankings, and medical services. The ultimate goal of any geospatial information retrieval system is to provide *all* relevant information to a user in a timely and efficiently manner. One of the challenges is that current technology, such as general purpose search engines, cannot be used as it would return a lot of irrelevant information which then overwhelms the user. Another challenge is that the dynamic nature of information requires a real-time solution which further complicates matters. Lastly, the huge amount of information we need to process would pose another difficulty in getting the relevant information in a timely fashion. We need a mechanism to provide large scale real time geospatial information retrieval [4]. For intelligence analysts, this task becomes very important because they need to retrieve the relevant geospatial information quickly from the massive amount of data. The ability to retrieve relevant information early is critical to the decision making process that every intelligence analyst has to make and thus eventually affects the outcomes of any missions.

In this paper, we focus on addressing the basic question of retrieving relevant information in an effective and efficient manner from a massive and dynamic geospatial information space, by designing a large-scale distributed real-time solution based on anytime intelligent foraging, gathering, and matching (I-FGM). I-FGM is designed as a multi-agent system. When a query is received by the system, the system creates a number of parallel processes that scour through the geospatial space looking for relevant nuggets of information. Each geospatial information nugget or *knowledge nugget* is represented by a directed acyclic graph containing incomplete or complete geospatial information. Since the search space is large and dynamic, using the traditional method of completely processing a nugget for determining its relevancy is computationally intensive and not very effective since providing real time results may not be possible with

this approach. I-FGM solves this problem by partially processing a nugget and calculating its partial similarity measure. Based on this partial appraisal, the system makes a decision whether to allocate more resources to this nugget. So it is the search space that directs how the finite resources are allocated. As the knowledge nuggets are incrementally processed by the system, the most relevant ones can be displayed to the user. This is useful in two ways: it ensures that users receive real time results and the updated relevancy measures of the dynamic knowledge nuggets are displayed. An important characteristic of IFGM is its modular architecture which allows easy 'plug-in' of IR technologies. This ensures that integration of new technologies into the system is easy. Since I-FGM is a multi-agent system, it is highly scalable and a large number of processes can be deployed in a parallel fashion. This helps in the quick and easy reallocation of computational resources and consequently helps in load balancing. It is clear that similarity measures play a pivotal role in the system though formulating the similarity function is a challenge as the similarity measure calculated using the partial knowledge nuggets must reflect its final relevancy value. In short, the key strength of this approach lies in the dynamic nature of gathering information, intelligently choosing the promising knowledge to process, partially and incrementally processing knowledge, and quickly presenting results to the user.

We assess the efficiency and effectiveness of our approach by implementing I-FGM and two other control systems which are the baseline and the partially intelligent foraging, gathering and matching system. The commonly used metrics in the information retrieval community – precision and recall – are used to evaluate the effectiveness of the system. We compute the efficiency in terms of work involved and in terms of running time for the systems.

This paper is organized as follows: We start with a brief overview of related work. Next, we present the architecture of I-FGM. This is followed by the description of our experimental set up and presentation results and analyses. We will then present our conclusions and future work.

2. RELATED WORK

In recent years, researchers in information retrieval and image processing have focused on the development of distributed geospatial information retrieval systems in which a user can retrieve relevant information about a given place. There have been many popular traditional applications from industry, such as Mapquest (available at http://www.mapquest.com) which allows a user to find geospatial information for a specific address from a static database. In addition, content-based information retrieval techniques and text-based information retrieval techniques have been explored to retrieve images, as in [2, 3, 5, and 11]. However, all the existing technologies assume that the databases are static and therefore, the complete information is available and unchanged during the time of retrieval. However, with the continuous growth of the Internet, information gets created and updated dynamically. For example, if we want to find an after-game report for a sports event related to the Final Four Tournament at New Orleans in 2004, at five minutes after the end of the game, we may get the final score, but obtain very little information about the game itself. However, at 15 minutes after the game, we will get the complete game report which includes the score, the players, and post-games quotes from couches and players. Even worse, during emergency situations such as September 11, 2001, the information is constantly changing with new critical information being created at an incredible rate Simply consider the various levels/hierarchies of intelligence analysts from single source analysts to all-source/national level analysts and the work-products they are creating. Starting at the lowest levels, these single-source analysts are generating enormous amounts of geospatial information which are necessary inputs to the increasingly higher levels of analysts and who must be made aware of them. Current searching and retrieval technologies, with their static assumptions, may attempt to remedy this through massive parallelism. However, when confronted with the attendant massive amounts of irrelevant information, such a straightforward solution fails and is a waste of resources.

Our approach assumes we must dynamically gather information from a dynamic information space. One early related work is the hypertext metacrawler from the University of Washington [10, 12, and 13]. The relevancy of such work to our approach is the methodology they use for combining the best results from leading search engines. In this work, they have performed experiments with different search engines to explore the returned documents for the same query as well as with the ranking algorithms. This is related to our work in determining the reliability of each search engine for improving retrieval performance. Another effort from Microsoft research on text-based image retrieval [1] gathers information by crawling the web once a day and constructing and updating an offline database for retrieval. The

relevancy of this work to our work is the creation of a database by crawling through the information space on a regular basis and the combination between text-based image retrieval and content-based image retrieval. We would like to look into this combination in the future.

3. SYSTEM ARCHITECTURE

3.1. Overview

Intuitively, in a general framework of a system to retrieve documents from a large, dynamic geospatial information space, we would need to select data from a geospatial information space, and then choose the most promising candidates for matching against a user's query while continuously updating the results for the user over time. Therefore, we design our I-FGM system with the following components:

1. I-Forager
2. gIG-Soup
3. gIG-Builder
4. I-Matcher
5. Blackboard

We note that a general I-FGM framework was discussed in [9]. In this paper, we focus our discussion on I-FGM for geospatial data. Specifically, we first focus on geospatial data represented as natural language text and take advantage of the current search engines to create our database.

3.2 Components

I-Forager: This component starts with receiving the query from a user, followed by gathering "would-be" relevant information from the geospatial information space, and then placing them in a repository called the gIG-Soup. I-Foragers use current search technologies to provide the first level of filtering. It is known that different search technologies work better with certain queries and not so well with others. Since the ranking provided by the search engines provide an initial approximation of the relevancy of the knowledge nugget, it is essential that we know the reliability of each search technology. A parameter that quantifies this is called the *reliability* of the I-Forager and is used to measure the first-order similarity of a document. The expected first-order similarity of a new document is defined as the initial priority assigned to the new document which is downloaded by the I-Foragers. For each downloaded document, it is a summation of the product of scaled reliability of the I-Forager and I-Forager measurement, which is the inverse of its rank in the search result that the I-Forager returned, over all I-Foragers.

The downloaded documents are placed in gIG-Soup (below) in order to extract knowledge nuggets. A knowledge nugget is defined as a directed graph (also referred to as document graph), which represents the concepts and relations between concepts in a geospatial domain. An example of a concept is represented as noun phrases such as "time square", and "wall street". We capture two relations in our knowledge nuggets: set-subset and related to. An example of relations in a knowledge nugget is: "time square" related to "New York city". For more information about this representation, please refer to [7, 8].

In the implementation, we use 5 machines as I-Foragers in our initial setting since we chose 5 popular search engines. Each I-Forager node uses Lynx-2.8.5 to retrieve documents from the WWW. Lynx is a text based browser. When a query is received from the user, the I-Forager uses Lynx to retrieve the URLs of relevant documents from a given search engine. Each I-Forger then downloads the documents using the URLs. The I-Foragers each uses a different search engine. The 5 search engines used in our experiment are Google, Yahoo, MSN, Teoma, and Looksmart. HTML tags and other useless data are removed from the downloaded documents and placed in the gIG-Soup. Since it is difficult to accurately define a function for the reliability of the I-Foragers, we make a simplification by giving all the I-Foragers equal reliability in order to better analyze our initial prototype.

gIG-Soup: This component is a repository that contains documents that have been foraged and whose document graphs are under construction by the gIG-Builder. The gIG-Soup is implemented as a Network File System (NFS) directory that is shared by all the nodes in the system. A sub-component of the gIG-Soup is a database (implemented using

MySQL) that contains important information about the documents in gIG-Soup such as file name, URL, identification for a query, number of sentences parsed and so forth. It also stores the partial and full relevancy measure of the documents in the gIG-Soup.

gIG-Builder: The gIG-Builder is the process that goes through the gIG-Soup, selects the most promising documents and works on building the document graph for a period of time. The most promising documents are documents with highest priority. Priority is a parameter which tells the system how potentially relevant this document is to the given query. Priority is determined by the chosen similarity measure, the history of the similarities, the history of parsing time, and the expected first-order similarity of the document. The priority of a newly downloaded document depends primarily on the expected first-order similarity. As the document is processed by the gIG-builders, the priority is refined by the system according to the similarity measures. It is the refinement process that helps to guide the allocation of computational resources. The period of time (also called parsing time) that the gIG builder processes a document is determined by its priority and the similarity measure used.

In our implementation, 5 nodes are used as gIG-Builders for our initial setting making it equal to the number of I-Foragers in order to see how this component works with other components before planning our expansion. The gIG-Builders continuously query the gIG-Soup, specifically its MySQL database, for documents whose document graphs have to be built. The documents with the highest priority are more likely to be selected. The information about document priority and the file containing the document are obtained from the MySQL database. The gIG-builders retrieve a document from the gIG-Soup and construct its document graph for a period of time. After this process is over, the document graph is placed back into the gIG-Soup. The MySQL database is updated with information about the documents, such as the number of lines parsed. Since there are multiple gIG-Builders working on the gIG-Soup, appropriate flags are set to ensure that they do not interfere with each other, while accessing the database and the documents from gIG-Soup.

I-Matcher: Each I-Matcher generates a query graph from a user's query. Next, I-Matcher processes the nuggets in the gIG-Soup and calculates/updates their relevance measure. I-Matcher queries the MySQL database continuously for documents that have been processed by the gIG-Builder and not yet matched. The documents with highest priorities will be more likely selected by the I-Matcher and the similarity values for them will be calculated. A match between a query graph q and a document graph di is defined as

$$\text{sim}(q,d_i)= \frac{n}{2*N} + \frac{m}{2*M}$$

where n and m are the number of concepts and relation nodes of the query graph found in the document graph and, N and M are the total numbers of concept and relation nodes of the query graph. Note that two relation nodes are matched if and only if at least their parent and their child are matched. In this formula, we treat relation nodes and concept nodes equally. The priority of the documents is then updated based on the similarity measure. We note that the similarity or match function need not be fixed to the definition above.

Blackboard: The blackboard continuously queries the MySQL database for any new results in the top ten most relevant documents. If there are any new results, they are displayed.

4. CONTROL SYSTEMS

In order to test and verify the effectiveness of our I-FGM system, we have implemented two control systems. They are the baseline system and partially intelligent system. We consider I-FGM to be the fully intelligent system. The main system components for these two systems such as I-Forager, I-Matcher, gIG-Soup, gIG-Builder, and Blackboard are similar to I-FGM system. However, the functionality of each component and the interaction between components are different and much simplified for the control systems. Also, to accurately compare performance, these three systems use the same information resources for the same query. The goal of designing the control systems are two folds. First, we would like to compare the performance of the I-FGM system with other systems and the control systems are implemented to serve this purpose. Secondly, in I-FGM system, we make use of knowledge about an information

nugget to guide the process of retrieving information. Therefore, by designing the baseline and partially intelligent systems, we would like to see how the knowledge of information nugget affects the retrieval process. Specifically, in baseline system, we have no knowledge about any information nugget while in the partially intelligent system, we only have initial knowledge about information nuggets.

4.1 Baseline System

The baseline system is a simple system without any intelligence. In this system, I-Foragers download the documents for a given query and directly place them into the gIG-Soup. The gIG-Builders choose documents randomly and construct document graphs for a fixed period of time. I-Matchers randomly pick a partially or completely parsed document graph to compare with the query graph.

4.2 Partially intelligent system

The partially intelligent system uses expected first-order similarity as the priority of the document during the whole query process. In other words, the I-Matcher will not update the document priority. The priority of the document is determined only based on the reliability of each I-Forager. Since the function for reliability will be refined as more geospatial information becomes available during the query process, this system is dynamic. Using the expected first-order similarity, it, to some extent, can dynamically and intelligently guide the finite computational resource in locating the target information. The gIG-Builders arrange the documents in the gIG-Soup according to priority and then randomly choose a document from top *n* documents. The parsing time is computed as the product of maximum allowed time and the priority.

4.3 I-FGM

In this system, after the I-Forager puts the document into gIG-Soup, the expected first-order similarity is determined and is used to calculate the initial priority of the document. The gIG-Builders randomly choose a document from a group of documents with highest priority to parse. In this implementation, we have randomly selected a document from a group of top 33% (1/3) of documents highly ranked according to priority. We chose this value from our initial experiments with different values (top 25%, top 33% and top 50%) conducted for the partially intelligent system.. The parsing time is computed as the product of maximum allowed time and priority. The difference between I-FGM and partially intelligent system is that priority is updated as the retrieval process progresses. I-Matcher compares the partial/complete document graph with the query graph and updates the relevance measure of the corresponding document. It also updates the document record in the Blackboard. When the I-Matcher finishes the comparison of a partial/complete parsed document, it updates the priority of the document.

The priority of a document *d* is computed as follows:

$$P_k = k_1 p_1 + k_2 p_2$$
$$p_1 = \delta P_{k-1} + (1-\delta) Sim_k$$
$$p_2 = \begin{cases} \Delta_k & \Delta_k > 0 \\ -\Delta_0 \dfrac{t_k}{t_0} & \Delta_k = 0 \end{cases}$$

where P_k represents the priority of a document *d* at time *k*. It is computed from the sum of the history of priority for this particular document, P_{k-1} and the current values of similarity Sim_k. δ is the weight put on the previous similarity. Δ_0 is the average expected similarity for the maximum parsing time. In our experiment, δ is assigned the value of 0.8 and Δ_0 is assigned the value of 0.6. We choose these values from our initial experiments conducted on the baseline and partially intelligent systems. We have placed more emphasis on the history of partial similarity, in computing priority for the I-FGM system. The flexible architecture of I-FGM allows us to modify priority, reliability and similarity functions easily.

Comparing partially intelligent control system with I-FGM, we find that the I-FGM combines the I-Forager's property with the document characteristics to dynamically guide the system action. On the other hand, the partially intelligent system only uses the characteristics of the I-Forager to guide the system action.

All these three systems are implemented on a set of 12 machines. Each machine or node has 1GHz Pentium III processor with 256 MB RAM. The machines are interconnected with 100 MBPS Ethernet network. Of the 12 nodes, 5 nodes have been implemented as I-Foragers, 5 nodes as gIG builders, 1 node as I-Matcher, and the remaining 1 node as the server that receives the query from the users. This experimental setup of 12 machines will be enlarged for later experiments. We also plan to upgrade the RAM capability for each node. The results from this evaluation will be used for computing the needed resources and expected performance in our future work.

5. EVALUATION

We evaluate I-FGM by comparing it with the control systems described above. A set of five queries are given to each of the three systems and results analyzed. In order to use the recall and precision performance metrics, we have to create a test bed. The procedure for creating a test bed is given below.

5.1 Testbed

The test bed is created for evaluating the retrieval performance of the given I-FGM systems. The topic of this evaluation is "Terrorism alerts for financial institutions, military compounds in New York city, New Jersey, and Washington DC". We have a set of five queries:
1. Financial buildings in New York City
2. Tall buildings in New Jersey
3. Military compounds in Washington DC
4. Financial institutions
5. Terrorism alerts for New York City

The set of targeted documents for each query is created by a pooling approach which includes the following steps:
Step 1: for each query in the query set, send the query to all available searching engines
Step 2: get top *50 results* from each search engine
Step 3: combine results for each query into a pool
Step 4: convert completely each document in the pool to document graph
Step 5: match each completely parsed document graph to the query graph
Step 6: create the list of documents with non-zero similarity measure
Step 7: consider the top 10 of these documents in this set as relevant documents

This approach to creating the testbed finds most of the relevant documents without having to assess every single document in the collection which is potentially huge in this case. Therefore, the set of targeted documents can be created with reasonable effort. We note that the set of targeted documents is highly dependent on the quality of search engines used.

5.2 Procedure for evaluating retrieval performance.

For each query:
- Run all three systems until all documents have been completely processed.
For each run, record the following data:
- Blackboard content: Define an interval amount of time t_i (t_i= 30 seconds in this experiment). We record the contents of the blackboard after each interval time t_i.
- Data used for computing reliability for each I-Forager: Number of documents, retrieved by a particular I-Forger that appeared in the final blackboard content.

The control systems were carefully chosen to bring out the salient features of I-FGM. By comparing with the baseline system, we are answering the question whether the methodology and similarity measures used in I-FGM is really necessary to designing an efficient IR system. By comparing with the partially intelligent system, we seek to prove that the superior performance of I-FGM cannot be emulated by just simply using the technology found in the search engines.

5.3. Metrics

We employ the commonly used metrics from information retrieval: precision and recall [6]. Precision is the ratio between the number of retrieved relevant documents over the number of retrieved documents. Recall is the ratio between the number of retrieved relevant documents over the number of relevant documents. Since we are reporting results as content to the blackboard, which is the top 10 documents at any given time, the precision and recall will be the same for this case.

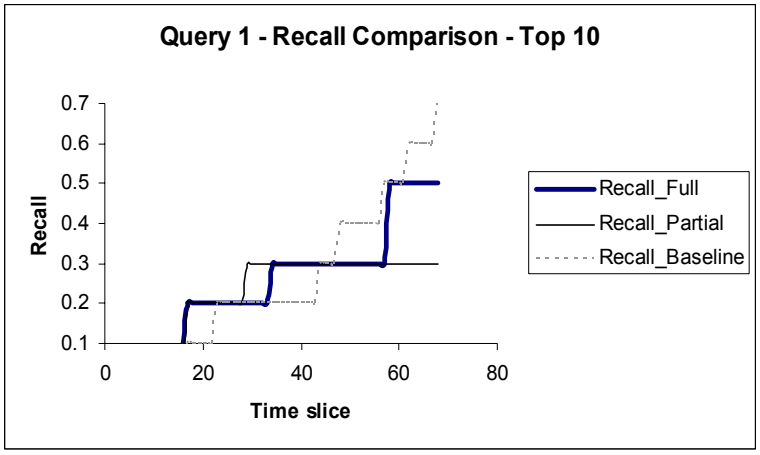
6. RESULTS AND ANALYSIS

The procedure described in Section 5.2 is used to conduct the experiments. The results were collected and analyzed. In Fig.1 (a-e), we show the recall values of the top 10 relevant documents for all five queries in our scenario over time. For each query, the plots show only the range of recall from 0 to 0.7 to highlight the early performance differences in the recalls for all three systems, which is critical to fast relevant retrieval. We plot the data to the point when at least one system met the recall level of at least 0.7. The y-axis is the recall value which is a number from 0 to 1 while the x-axis represents time-slices starting from 0. Two time slices are half a minute apart from each other. In general, the I-FGM system maintains higher recall than the partially intelligent system and baseline for some time, before the partially intelligent system and baseline systems catches up with the I-FGM. It is shown clearly by the results for queries 3, 4 and 5. Take for instance query 4; the baseline system did not retrieve any quality relevant documents in the top 10 for the first 7 minutes, while it takes only 1.5 minutes for the I-FGM to retrieve 30% of the top 10. It takes 2.5 minutes for the partially intelligent system to retrieve 30% of the total quality relevant documents in the top 10 for this query. Similarly, for query 3, it takes the baseline and partially intelligent system 13 minutes to retrieve a document from top 10 whereas it takes only 3 minutes for I-FGM to retrieve a document from top 10. For query 5, the I-FGM system is clearly better than both baseline and partially intelligent system. It takes 9 minutes for I-FGM to retrieve 50% of top 10 documents while it takes the same amount of time for the baseline system to retrieve a document from top 10. The partially intelligent system did not do well in this query, either. The strength of the IFGM system is that it is able to direct the resources to promising candidates. By doing so, we should be able to get the top 10 documents in the blackboard significantly quicker than the baseline and partially intelligent systems. This is validated by the results of queries 3, 4 and 5.

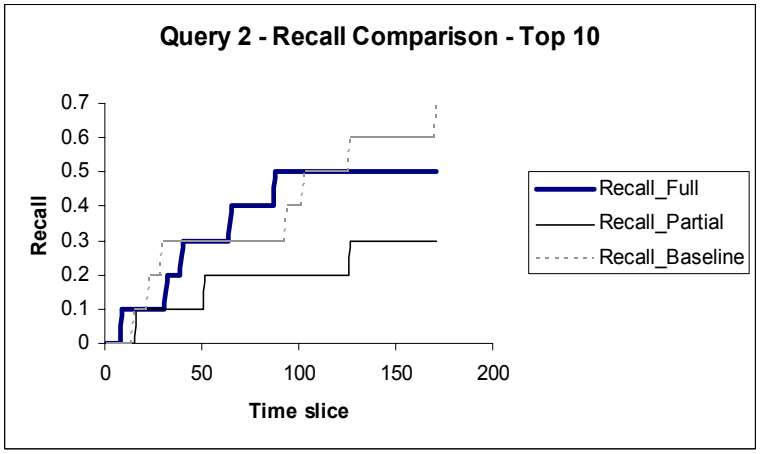
The performance difference in terms of recall for queries 1 and 2 of the I-FGM system compared to the baseline system is less marked. IFGM continues to attain lower levels of recall very quickly. For query 1, the baseline and partially intelligent systems need an equal amount of time to retrieve a quality document from the top 10 while IFGM retrieves a document from top 10 list 30 seconds earlier. I-FGM performed better than the baseline system for 3.5 minutes during which time, recall of the I-FGM system is 0.2 versus 0.1 of baseline system. For the remaining recall levels, it trailed the baseline system. For query 2, I-FGM needed 5.5 minutes to retrieve a document from top 10 while it took 7.5 and 8 minutes for the baseline and partially intelligent systems to retrieve a document from top 10, respectively. However, the baseline system gradually achieved higher recall compared to I-FGM and partially intelligent system. The partially intelligent system did not perform well for queries 1 and 2.

The reason that the I-FGM system did not work as well for queries 1 and 2 compared to queries 3, 4 and 5 is that some documents ranked highly by reliable search engines contain very little information related to the query and only contained a lot of links, relevant to the current query. The search engines ranked these documents highly because they take the popularity of the webpage into account. When such a document is selected, the I-FGM system will assign a longer parsing time to process it. However, these documents turn out to contain very little relevant information. The baseline on the other hand, inadvertently avoids this situation since it gives each document an equal amount of parsing time and randomly picks a document to process. Although there is no intelligence in this resource allocation strategy, it works in such situations. Another important observation from the results that we noticed is that baseline achieves better

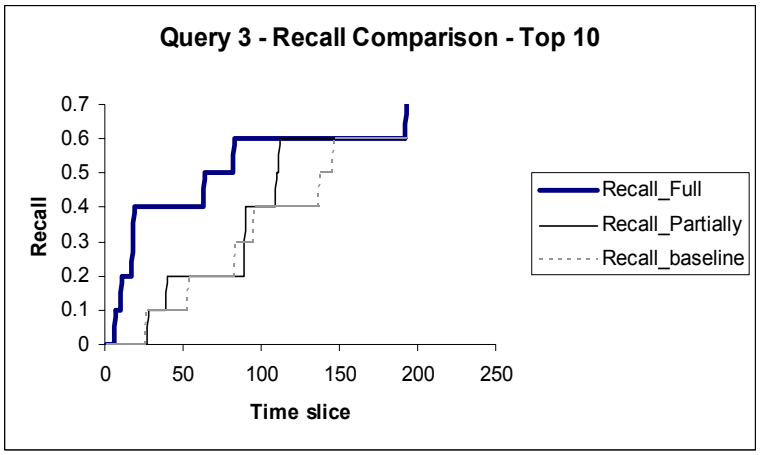
recall as time progresses. This is because after I-FGM has identified around 25-40% of the relevant documents, it will still be allocating parsing time for these documents in the subsequent iterations. Hence, the time given for any fresh candidate is small. Since baseline gives constant time to all the documents in the search space, it will identify the rest of the relevant documents quicker than I-FGM. We can address this problem by continuing to develop an effective reliability function. As discussed earlier, we have given equal reliability to all the I-Foragers. Past performance of the I-Forger in terms of recall, will be one of the factors used to calculate reliability. We believe that with more experiments, larger testbeds, and improved priority and reliability functions, we will be able to address this problem.



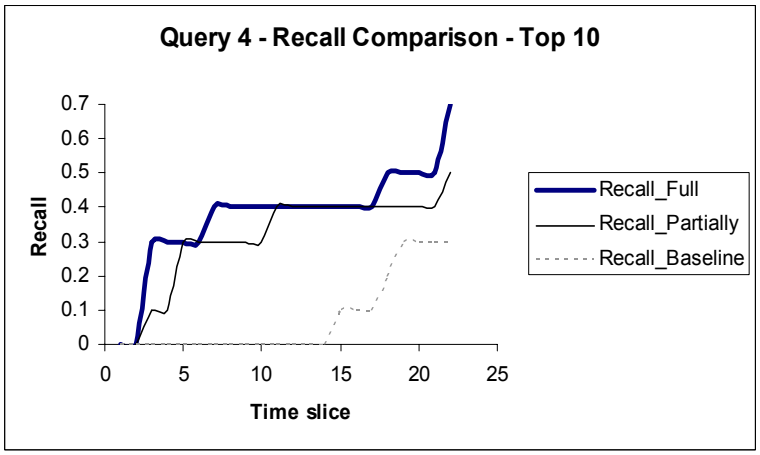
(a)



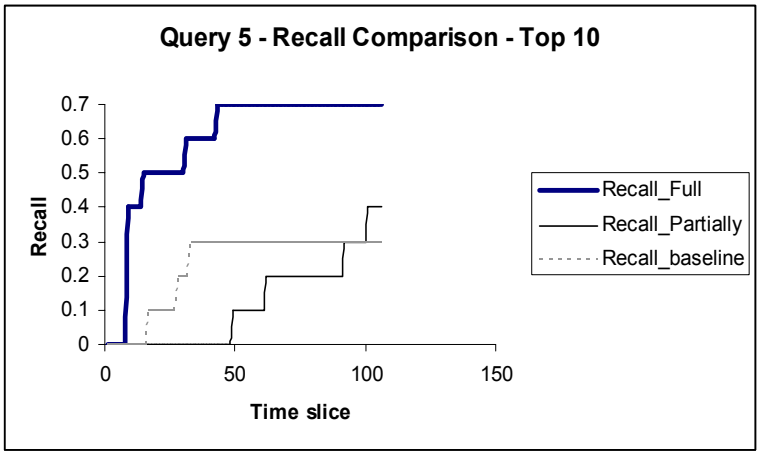
(b)



(c)



(d)



(e)

Figure 1: Recall for (a) query 1, (b) query 2, (c) query 3, (d) query 4 and (e) query 5

We have also listed the time required for three systems to reach a certain recall level. We choose 0.2, 0.5, and 0.7 to correspond to low, medium, and high recall levels, respectively. Each value in Table 1 represents the time taken for a system to reach a certain recall level. For example, for query 4, I-FGM met the recall level 0.5 at 9 minute mark. As shown in Table 1, I-FGM system achieves high recalls early compared to baseline system for queries 3, 4 and 5, but it only works well at low and middle recall points for query 1. It did not beat the baseline system for query 2. The partially intelligent system only achieves high recalls early for query 4.

In Table 2, we assess the quality of our blackboard by computing the difference between the final similarity measure of documents in the final top 10 relevant documents and its partial similarity during the first 10 minutes of the system run time. We choose this particular period of time because for some of the control systems, it takes a long period of time to retrieve the top 10 documents into the blackboard. The values in Table 2 represent the average of similarity differences for each system. As we can see, the average difference for I-FGM is the smallest one for four queries except query 2. The partially intelligent system achieves the second smallest difference in 4 queries except query 1. This means that I-FGM produces better quality blackboard content compared to the two control systems. In summary, our I-FGM system can achieve higher recall level earlier than the partially intelligent system and the baseline system and also provides more quality blackboard compared to these two control systems.

Recall for top 10 relevant documents	Time (minutes)		
	I-FGM	Partially Intelligent	Baseline
Query 1			
>=0.2 & < 0.5	8.5	8.5	11.5
>=0.5 & < 0.7	29	54	28.5
>=0.7	60.5	168	34
Query 2			
>=0.2 & < 0.5	16	26	11.5
>=0.5 & < 0.7	44	118.5	51.5
>=0.7	130	208	85.5
Query 3			
>=0.2 & < 0.5	5.5	20	27
>=0.5 & < 0.7	32	55	69
>=0.7	96.5	100	104.5
Query 4			
>=0.2 & < 0.5	1.5	2.5	9
>=0.5 & < 0.7	9	11	12.5
>=0.7	11	12	15
Query 5			
>=0.2 & < 0.5	5.5	32	15
>=0.5 & < 0.7	8.5	94	56
>=0.7	22.5	121	61.5

Table 1: Summary of the time for reaching 3 point recall for 5 queries for all three systems.

Average difference in similarity	Query 1	Query 2	Query 3	Query 4	Query 5
I-FGM	0.566	0.77	0.27	0.181	0.73
Partially Intelligent System	0.88	0.594	0.35	0.194	0.87
Baseline	0.598	0.628	0.36	0.207	0.89

Table 2: Average difference of similarity for Blackboard content for the first 10 minutes of system runtime.

7. CONCLUSION

In this paper, we reported our preliminary results of the design, implementation and evaluation of the Intelligent Foraging, Gathering and Matching system (I-FGM) to retrieve geospatial information. We have shown through the evaluation of our short scenarios that I-FGM system always retrieved quality documents faster than the control systems. In essence, we have proved that the methodology of partially evaluating the information in the search space and incrementally updating its relevance measure works and provides an effective way to use the computational resources efficiently and consequently, retrieve results quickly. We also show that by using a simple heuristic of choosing which documents to process next, the retrieval performance of our system can be increased substantially.

There were some issues brought to our attention through our evaluation of the I-FGM system. One is the issue of how much computational resource is required by the system without being excessive. In our experiments, we used 1 machine for matching and 5 machine for building document graphs. We would be interested in observing how the number of machines used for matching and building document graphs affects the results of the recall for the top 10 documents and also the proportion of the various components needed to prevent resource bottlenecks. These studies are also important as we will be modifying some of the I-FGM components to be inter-operable. Thus, an I-Matcher can act as a gIG-Builder when the need arises. Secondly, the priority function that we are currently using, takes into account only the time spent previously and the current similarity of this document. The size of the document remaining to be processed is not taken into account in the priority function. Doing so could provide a means for more equitable use of computational resources potentially leading to more efficient processing. New documents will then have a better chance to be allocated processing time. We will also combine machine learning techniques with certain empirical values such as number of parsed sentences and the closeness of the sentences to the current query, to determine the priority for each document. Third, any information retrieval algorithm is task and collection dependent. We will explore experimenting with different search domains and larger testbeds to gain further insights and continue validation of I-FGM. Lastly, we want to find out how dynamic the information space is and how it affects our retrieval performance by performing experiments at different times and measuring the differences in terms of relevant documents and retrieved documents. The use of crawlers to further explore the information space in real-time will also help address this issue.

In conclusion, the Intelligent Foraging, Gathering and Matching (I-FGM) system has the potential to be a new innovation in the field of geospatial information retrieval. The preliminary experimental results validate the success of our I-FGM system. The strengths of its architecture and similarity measures can be further harnessed for better efficiency and accuracy.

ACKNOWLEDGEMENTS

The work presented in this paper was supported in part by the National Geospatial Intelligence Agency Grant No. HM1582-04-1-2027.

REFERENCES

[1] Chen Z., Wenyin L., Zhang F., Li M. and Zhang H. *Web Mining for Web Image Retrieval.* Journal of the American Society for Information Science. Vol. 52(10). Pages 831-839, 2001.

[2] Gudivada V. N. and Raghavan V. V. *Modeling and Retrieving Images by Content.* Information Processing & Management. Vol 33 (4). Pages 427-452. 1997

[3] Jeon J., Lavrenko V. and Manmatha R. *Automatic Image Annotation and Retrieval using Cross-Media Relevance Models*. In Proceedings of the 26th Annual International ACM SIGIR conference on Research and Development in Information Retrieval. Pages 119-126. 2003.

[4] National Research Council. *Distributed Geolibraries Spatial Information Resources.* National Academy Press. Washington DC. 1999.
[5] Rui Y. and Huang S. T. *Image Retrieval: Current Techniques, Promising Directions, and Open Issues*. Journal of Visual Communication and Image Representation. Vol 10. Pages 39-62. 1999.
[6] Salton G. and McGill M. Introduction to Modern Information Retrieval. McGraw-Hill. 1983.
[7] Santos Jr. E., Nguyen H., and Brown M. S. *Kavanah: An Active User Interface information Retrieval Agent Technology.* Maebashi, Japan, October 2001. Pages 412-423. 2001.
[8] Santos Jr. E., Nguyen H., Zhao Q and Pukinskis E. *Empirical Evaluation of Adaptive User Modeling in a Medical Information Retrieval Application.* In Proceedings of the 9th User Modeling Conference, Pages 292-296. 2003
[9] Santos Jr. E., Santos E.E. and Santos E.S. *Distributed Real-Time Large-Scale Information Processing and Analysis* In Proceedings of the SPIE Defense & Security Symposium, Vol. 5421. Pages 161-171. 2004.
[10] Selberg E. and Etzioni O. *The MetaCrawler Architecture for Resource Aggregation on the Web.* IEEE Expert. January-February issue. Pages 11-14. 1997.
[11] Weigand N. and Zhou N. Ontology-Based Geospatial Web Query System. Next Generation Geospatial Information – 2003 (NG2I 03) Conference, 2003.
[12] Zamir O. and Etzioni O. *Grouper: A Dynamic Clustering Interface to Web Search Results.* Computer Networks (Amsterdam, Netherlands). Vol 31 (11-16). Pages 1361-1374. 1999
[13] Zamir O. and Etzioni O. *Web Document Clustering: A Feasibility Demonstration.* In Proceedings of the 21st annual international ACM SIGIR conference on Research and Development in Information Retrieval. Pages 46-54. 1998.