CARROT II and the TREC 11 Web Track

R. Scott Cost, Srikanth Kallurkar, Hemali Majithia, Charles Nicholas, and Yongmei Shi

University of Maryland, Baltimore County
Baltimore, MD USA
{cost,skallu1,hema1,nicholas,yshi1}@csee.umbc.edu

Abstract. We describe CARROT II, an agent-based architecture for distributed information retrieval and document collection management. CARROT II consists of an arbitrary number of agents, distributed across a variety of platforms and locations. CARROT II agents provide search services over local document collections or information sources. They advertise content-derived metadata that describes their local document store. This metadata is sent to other CARROT II agents which agree to act as brokers for that collection, and every agent in the system has the ability to serve as such a broker. A query can be sent to any CARROT II agent, which can decide to answer the query itself from its local collection, or to send the query on to other agents whose metadata indicate that they would be able to answer the query, or send the query on further. Search results from multiple agents are merged and returned to the user. CARROT II differs from similar systems in that metadata takes the form of an automatically generated, unstructured feature vector, and that any agent in the system can act as a broker, so there is no centralized control. We present experimental results of retrieval performance and effectiveness in a distributed environment.
We have evaluated CARROT II in the context of the Web Track of NIST's annual Text Retrieval Conference. Our methodology is described, and results are presented. [1]

1 Introduction

We have developed a scalable, distributed query routing and information retrieval system, CARROT II. It is the successor of an earlier project (Collaborative Agent-based Routing and Retrieval of Text) [8] (originally CAFE [7]). CARROT II is composed of a flexible hierarchy of query routing agents. These agents communicate with one another using KQML [9] and the Jackal platform [5], and may be distributed across the Internet. While all agents in the system are alike, they can each control widely varying information systems. Agents interact with information sources via a well-defined interface. Queries presented to any agent in the system are routed, based on the content of the query and metadata about the contents of the servers, to the appropriate destination. Agents themselves are uniform and extremely simple.

CARROT II contains wrappers that extend other well-known IR systems (e.g. MG [19], Telltale [15]), as well as CARROT II's own, modest IR system. These wrappers present

[1] This document is an expanded version of a paper presented at the Workshop on Cooperative Information Agents Workshop (CIA '02) [6].

a basic interface to the CARROT II system for operating on documents and metadata. Agents are addressable via commands that are communicated in KQML. This means that a CARROT II system can be created, configured, and accessed by another information system, and so can be employed to extend the search capabilities of an existing project. We use the Jackal platform to support communication among agents in CARROT II and to provide an interface to the outside world.

Section 2 discusses the problems areas facing DIR and the efforts so far by the research community, Section 3 describes the CARROT II architecture and Section 4 describes the operation of the system. In Section 5, we describe our experience with TREC data.

2 Related Work

In the past there have been attempts to introduce the concepts of agent-based information retrieval. Systems like SavvySearch [13] demonstrated a simple approach to querying web search engines and combining their results in a single ranked order.

Historically, Harvest was the first system to demonstrate the use of broker agents in distributed search. The Harvest system [2] is a distributed, brokered system originally designed for searching Web, FTP, and Gopher servers. In Harvest, "gatherer" agents collect metadata from information providers and deliver it to one or more brokers. Metadata objects are represented in Summary Object Interchange Format (SOIF), an extensible attribute-value-based description record. Harvest pioneered the ideas of brokering, metasearch, replication, and caching on the Internet.

2.1 Distributed Information Retrieval

Information Retrieval in a distributed environment normally follows three steps [3]:

1. Information Source Selection: Select the best information source(s) per query
2. Query Processing: Send the query to the source(s) and return ranked list(s) of documents
3. Results Fusion: Create a single ranked list from ranked lists returned from the sources.

For retrieval from text, one of the methods for information source selection is use of automatically generated metadata from the content. Comparing the query to metadata about the sources can reveal the possible relevance of each source to the query. CORI [4] and gGloss [12] are examples of such metadata in information source selection. The CORI model is based on inference networks. CORI creates a virtual document containing Document Frequency (DF) and Inverse Collection Frequency (ICF). The ICF indicates the importance of the term across the collections and is analogous to the Inverse Document Frequency (IDF), which is a measure of term importance in a single collection. gGloss creates a virtual document containing DF(s) and Term Frequency (TF), i.e. number of occurrences per document of unique terms of the collection. French et al. [10] showed that CORI performed better than gGloss in terms of

retrieval effectiveness, however they could not provide a reason for CORI's superior performance.

Gibbins and Hall [11] modeled query routing topologies for Resource Discovery in mediator based distributed information systems. Liu [14] demonstrated query routing using processes of query refinement and source selection, which interleaved query and database source profiles to obtain a subset of good databases.

The final step in answering a query is fusing the ranked lists from the queried sources to obtain a single ranked list. Voorhees et al. [18] used query training to determine the number of documents that should be selected from each source, then used a die based approach to determine the next source from which a document should be selected to form the final ranked list. Aslam and Montague [1] showed that results fusion based on ranks alone can be as good as regular fusion techniques and that relevance scores are not required.

In general, there is a performance gain by distributing information, but distributed retrieval lags behind centralized retrieval in terms of retrieval effectiveness, i.e. percentage of relevant documents returned for a query. However Powell et al. [16] showed that a distributed search can outperform a centralized search under certain conditions.

3 CARROT II Architecture

A CARROT II system is a collection of coordinated, distributed agents managing a set of possibly heterogeneous IR resources. These agents each perform the basic tasks of collection selection, query routing, query processing, and data fusion. In order to effect this coordination, some amount of underlying structure is required.

There are three components to the CARROT II architecture. The central work of CARROT II is performed by a distributed collection of CARROT II agents. There is also a network of infrastructure agents which facilitate communication and control of the system. Finally, a small set of support agents facilitates access to the system, and coordinates its activities. Each of these components is described in detail below.

3.1 CARROT II Agents

The CARROT II agent is the cornerstone of the CARROT II system. Its role is to manage a certain corpus, accept queries, and either answer them itself, or forward them to other CARROT II agents in the system. In order to do this, each CARROT II agent creates and distributes metadata describing its own corpus to other CARROT II agents. All CARROT II agents are identical, although the information systems they manage may vary.

A standard interface provides methods for manipulating documents, metadata, and handling queries. The agent maintains a catalog of metadata which contains information about the documents stored by peer agents.

3.2 Information Integration

A CARROT II agent interfaces with an information source which may be an ordinary IR package, or a database manager.

The CARROT II system currently has a wrapper that interfaces with the WONDIR[2] IR engine. It can however be extended to support other types of Information sources. As mentioned earlier, metadata is derived from the document collection. The metadata takes the form of a vector of the unique "N-grams" of the collection and a sum of their number of occurrences across all documents in the collection. Hence, unlike Harvest, CARROT II metadata describes the agent's collection of documents, not a single document. CARROT II uses such metadata for source selection. The motivation for such a form of metadata comes from relative ease of use, low cost of generation, and the ability to aggregate metadata, such that a single vector may contain metadata about multiple agents.

The same query operation can now be performed on both documents and metadata. A query operation returns a similarity score using $TF * IDF$ based cosine similarity [17]. Querying a collection returns a ranked list of the documents sorted by their similarity scores. For querying metadata collection IDF is replaced by the ICF (see Section 2).

3.3 CARROT II Infrastructure Agents

In order to support the successful inter-operation of potentially very many CARROT II agents, we have constructed a hierarchical infrastructure. The infrastructure is largely dormant while CARROT II is in operation, serving primarily to facilitate the orderly startup and shutdown of the system, and provide communications support.

The infrastructure is controlled by a single Master Agent, which may be located anywhere on the network. At startup, the Master Agent is instructed as to the number of agents required, and some factors regarding their desired distribution. These include the number of physical nodes to be used, as well as the degree of resource sharing at various levels.

The Master Agent starts a Node Agent on each participating machine, and delegates to it the task of creating a subset of the required agents. The node will be divided into Platforms, or independent Java Virtual Machines, each governed by a Platform Agent. The Node Agent creates an appropriate number of Platforms, and again delegates the creation of a set of agents.

Within each Platform, the Platform Agent creates a set of Cluster agents. The purpose of the Cluster Agent is to consolidate some of the 'heavier' resources that will be used by the CARROT II agents. Primarily, this means communication resources. A Cluster Agent maintains a single instance of Jackal. Each Cluster Agent creates a series of CARROT II agents; these run as subthreads of the Cluster Agent. Because most agents will be dormant at any one time, we allow a CARROT II agent to be assigned more than one collection, creating a set of 'virtual' agents. Thus the virtual agents are the agents visible to all entities external to the system.

[2] Word or N-gram based Dynamic Information Retrieval; an in-house system, developed as part of the CARROT II project

3.4 CARROT II Support Agents

While the agents in the CARROT II system work largely independently, a small set of support agents serve to coordinate the system's activities. These are the Agent Name Server, the Logger Agent, and the Collection Manager.

An Agent Name Server provides basic communication facilitation among the agents. Through the use of Jackal, CARROT II employs a hierarchical naming scheme, and its operation is distributed through a hierarchy of name servers.

A Logger Agent monitors log traffic, and allows the system to assemble information about the details of operation. This information can then be used to feed monitors or visualization tools.

Finally, a Collection Manager facilitates the distribution of data and metadata. It determines which collection of documents or information source will be assigned to each agent, how each agent will distribute its metadata, and what set of agents will be visible outside the system.

4 CARROT II Operation

Metadata distribution and query processing are the two main functions of the CARROT II agents. Recall from Section 3.2 that the CARROT II metadata is an automatically generated feature vector derived from the content itself.

4.1 Metadata Distribution

CARROT II uses a vector-based representation of metadata which describes the contents of the local corpus (see [8]). Metadata, as well as corpus documents, are managed by the IR engine (for example, WONDIR). The distribution of metadata has a profound impact on the system's ability to route queries effectively, and determines the "shape" of the CARROT II system. Agents receive instructions on metadata distribution from the Collection Manager. There are three metadata distribution modes that can be used by CARROT II:

1. Flood: Each agent broadcasts it's metadata to every other agent in the system. Under this scheme, any agent receiving a query would have complete (and identical) knowledge of the system, and be able in theory to find the optimal target for that query.
2. Global: As in the original CARROT architecture a designated broker agent has knowledge of the entire system. All agents share their metadata with only this agent.
3. Group: Once the system reaches a certain size, however, both of the above schemes are impractical; the system would become susceptible to bottlenecking. The group scheme is based on mathematical, quorum-based distributions, where a group of agents is represented by a chosen (or elected) agent. Metadata sharing would occur inside such groups and amongst the group leaders.

Metadata distribution can be effected by transferring the entire vector, a "difference" vector in case of changes in the agent's corpus, or just a URL pointing to the location of the metadata, enclosed in a KQML message.

4.2 Query Processing

Once the system is running, the Collection Manager becomes the primary or initial interface for outside clients. A client first contacts the Collection Manager to get the name or names of CARROT II agents that it may query. The names in this set will be determined by the metadata distribution policy. For example, in the case of group-based distribution, the set will contain the group leader agent names. The client then sends queries to randomly selected agents from the given set. It is also possible to model more restricted or brokered architectures by limiting the list to only one or a few agents, which would then feed queries to the remainder of the system.

In response to an incoming query, an agent decides whether the query should be answered locally, forwarded to other agents, or both. The agent compares the query to its local metadata collection and determines the best destinations. Based on the results, it may send the query to the single best source of information, or it may choose to send it to several. One of those sources may be its own local IR engine. Once answers are computed and received, the results are forwarded back to the originator of the query. If more than one information source is targeted, the agent faces the problem of fusing the information it receives into one coherent response.

Queries may be routed through a number of different agents before finally being resolved; this depends on the scheme used for metadata distribution and the routing algorithm. For example, the simplest scheme is to have each agent broadcast its metadata to every other agent in the system. The corresponding routing algorithm would be to route to the best information source. Since all agents have the same metadata collection and employ the same algorithm, a query will be forwarded at most once before being resolved. For schemes which employ a more efficient distribution of metadata, or possibly higher order metadata, queries may pass through many CARROT II agents before finally being resolved.

4.3 Implementation

The current implementation of CARROT II uses the flood mode of metadata distribution. This implies that a query given to any agent in the system will return the same answer. The query can be routed either based on a metadata similarity cutoff or to the best N agents, based on the metadata similarity scores. Therefore, as stated in Section 4.2, the query needs to be *forwarded* only once. The results fusion is based on Voorhees's query clustering approach [18]. But unlike their method, the importance of the collection is measured by the metadata similarity score generated. The metadata score is simply applied as a factor to each of the individual document similarity scores of each collection's ranked list. The results are then sorted based on this new similarity score and the top N documents returned as results.

5 TREC 11

We have evaluated CARROT II in the context of the TREC 11 Web Track. This track of NIST's annual Text Retrieval Conference focuses on retrieval tasks in web data; in

this case, an 18 gigabyte crawl of the .gov domain. The task we chose to attempt was 'topic distillation'. In this task, the goal is to find the best resource page for a given query. Such a page may not necessarily be the page which best matches the content of the query. Other factors, such as links to other relevant pages, may also play a role.

For this task, we used a CARROT II system of 65 agents for indexing and retrieval. Table 1 shows the average document and index sizes. A total of five runs were submitted. The experiments were conducted in two phases. In the first phase the sub-collections were indexed and 1000 results for the 50 topics query generated, under normal CARROT II operation. Two different approaches to query dispersion were used, resulting in two 'base' runs. In the first, queries received were forwarded to all agents in the system. In the second, queries were forwarded to only the ten best agents for that query, based on sub-collection metadata. The results of these base runs were then used as initial seeds and used along with the link topology information provided with the collection. A description of the link topology usage is provided in below:

5.1 Link Topology

The link topology information was used in the following way:

1. Using the link topology file provided by TREC, for each document, a list of in-links (documents which point to this document) and outlinks (documents which are pointed to by this document) was compiled. The nodes of the file are regarded as nodes of a supergraph G.
2. The 1000 top-ranked documents (seeds), generated from the base runs, are expanded on their inlinks, and a subgraph V is obtained.
3. A directed graph DG on V from G is created.
4. The similarity values are propagated from the seeds on the DG, using one of the 3 formulae (for different runs):
   (a) Constant c = 1.1
   if (at least half of the children have greater similarity than itself)
   *sim = c * avg(best-children)*
   else
   sim = avg(sim, avg (worst-children))
   (b) Constant c = 1.05
   if (at least half of the children have greater similarity than itself)
   sim = $c^{(number-of-good-children)}$ ** avg(best-children),*
   sim = min(1+ sim(best-child))/2 , sim)
   else
   sim = avg(sim, avg (worst-children))
   (c) Constant c = 1.0
   if (at least quarter of the children have greater similarity than itself)
   sim = $c^{(number-of-good-children}$ *) * avg(best-children)*
   else
   sim = min(1+ sim(best-child))/2 , sim),
   sim = avg(sim, avg (worst-children))
5. The 1000 nodes in DG with the largest sim-value are returned.

Collection	.gov
Average Number of Documents per Agent	19000
Average Document Collection Size	275MB
Average Document Size	6.3KB
Average Index Size per Agent	93MB
Average Metadata Size per Agent	899MB

Table 1. .gov TREC Web Track Collection Statistics

Mean Average Precision	Run1	Run3
At 5 Documents	0.15	0.08
At 10 Documents	0.11	0.05
At 100 Documents	0.03	0.02

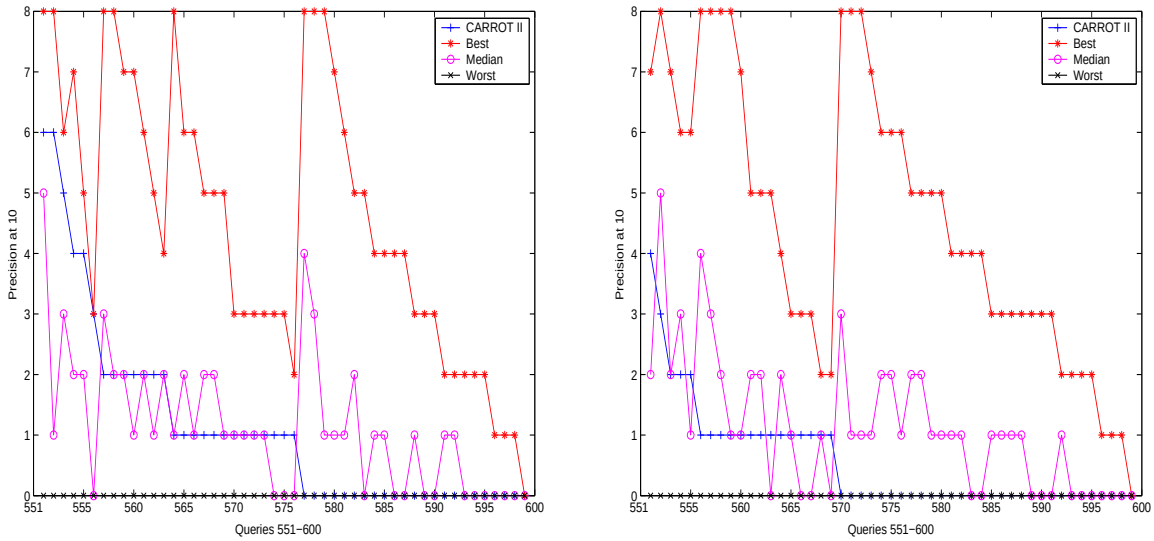
Table 2. *CARROT II* Performance on Topic Distillation Task

5.2 Runs

A description of the five runs submitted is given below:

1. RUN1 (CARROT2A): Up to 65 collections were queried and results fused.
2. RUN2 (CARROT2C): Based on results from RUN1, use link structure and formula a.
3. RUN3 (CARROT2B): Up to 10 collections were queried and results merged.
4. RUN4 (CARROT2D): Based on results from RUN1, use link structure and formula b.
5. RUN5 (CARROT2E): Based on results from RUN1, use link structure and formula c.

Table 2 shows the precision values (provided by TREC) of RUN1 and RUN3 for the 50 queries 551 to 600. The exception was query 582, for which precision values were not provided. We also compared RUN1 and RUN3 precision values with the best, median and worst precision values for each query (see Figure 1). The graph was made by sorting the CARROT II values and using them as reference for the best, median and worst values (The queries represented on the X-asis were sorted in order of decreasing precision values of the CARROT II system). The comparsion showed that for approximately half the number of queries, the CARROT II retrieval effectiveness was close to median. Thus, the simple cosine similarity metric based distributed retrieval approach was able to perform close to the median without enhancements such as query expansion. We believe that these results are encouraging for further improvements in database selection and results fusion techniques for improvements in effectiveness. However, in general, advanced retrieval tasks such as topic distillation on large amounts data maybe outside the scope of purely statistical retrieval techniques. The trial runs involving usage of link structure information did not produce noticably different results from those of RUNS 1 and 3.



(RUN1) (RUN3)

Fig. 1. Comparison of Precsion Results for RUN1 and RUN3

6 Conclusions

We presented an initial prototype of a Distributed Information Retrieval system on an agent-based architecture. We have built a system that demonstrates that a distributed information retrieval approach may perform at a level comparable to or slightly less than, that of a centralized system with a corpus of the same size. Our approach to topic distilation was based on using the cosine similarity metric with the data divided amongst agents. The observations derived from our participation at the Web Track are:

1. There is a scope of improvement in terms of using just term statistic based cosine similarity metrics, augmented with link topology information and,
2. The potential benefits of distributed retrieval in open environments like the web does make a case further research.

References

1. J. A. Aslam and M. Montague. Models for metasearch. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 276–284, 2001.
2. C. M. Bowman, P. B. Danzig, D. R. Hardy, U. Manber, and M. F. Schwartz. The Harvest information discovery and access system. *Computer Networks and ISDN Systems*, 28(1–2):119–125, 1995.

3. J. Callan. *Advances in Information Retrieval*, chapter 6: Distributed Information Retrieval, pages 127–150. Kluwer Academic Publishers, 2000.
4. J. P. Callan, Z. Lu, and W. B. Croft. Searching distributed collections with inference networks. In E. A. Fox, P. Ingwersen, and R. Fidel, editors, *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–28, Seattle, Washington, 1995. ACM Press.
5. R. S. Cost, T. Finin, Y. Labrou, X. Luan, Y. Peng, I. Soboroff, J. Mayfield, and A. Boughannam. Jackal: A Java-based tool for agent development. In J. Baxter and C. Brian Logan, editors, *Working Notes of the Workshop on Tools for Developing Agents, AAAI '98*, number WS-98-10 in AAAI Technical Reports, pages 73–82, Minneapolis, Minnesota, July 1998. AAAI, AAAI Press.
6. R. S. Cost, S. Kallurkar, H. Majithia, C. Nicholas, and Y. Shi. Integrating Distributed Information Sources with CARROT II. In *Proceedings of the Sixth International Workshop on Cooperative Information Agents, CIA 02*, September 2002.
7. G. Crowder and C. Nicholas. Resource selection in CAFE: An architecture for network information retrieval. In *Proceedings of the Network Information Retrieval Workshop, of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval 96*, August 1996.
8. G. Crowder and C. Nicholas. Metadata for distributed text retrieval. In *Proceedings of the Network Information Retrieval Workshop, SIGIR 97*, 1997.
9. T. Finin, Y. Labrou, and J. Mayfield. KQML as an agent communication language. In J. Bradshaw, editor, *Software Agents*. MIT Press, 1997.
10. J. C. French, A. L. Powell, J. P. Callan, C. L. Viles, T. Emmitt, K. J. Prey, and Y. Mou. Comparing the performance of database selection algorithms. In *Proceedings of the 22th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 238–245, 1999.
11. N. Gibbins and W. Hall Scalability issues for query routing service discovery. In *Second Workshop on Infrastructure for Agents, MAS and Scalable MAS at the Fourth International Conference on Autonomous Agents*, pages 209–217, 2001.
12. L. Gravano and H. Garcia-Molina. Generalizing gloss to vector-space databases and broker hierarchies. In *Proceedings of the 21st VLDB Conference*, Zurich, Switzerland, 1995.
13. A. E. Howe and D. Dreilinger. SAVVYSEARCH: A metasearch engine that learns which search engines to query. *AI Magazine*, 18(2):19–25, 1997.
14. L. Liu. Query Routing in Large Scale Digital Library Systems. *ICDE, IEEE Press*, 1997.
15. C. Pearce and C. Nicholas. TELLTALE: Experiments in a dynamic hypertext environment for degraded and multilingual data. *Journal of the American Society for Information Science*, June 1994.
16. A. L. Powell, J. C. French, J. Callan, M. Connell, and C. L. Viles. The impact of database selection on distributed searching. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 232–239, 2000.
17. G. Salton, C. Yang, and A. Wong. A vector space model for automatic indexing. *Communication of the ACM*, pages 613–620, 1975.
18. E. M. Voorhees, N. K. Gupta, and B. Johnson-Laird. Learning collection fusion strategies. In *SIGIR*, pages 172–179, 1995.
19. I. H. Witten, A. Moffat, and T. C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Van Nostrand Reinhold, 1994.