A Multi-Agent System for Distributed Information Retrieval on the World Wide Web

Keith L. Clark and Vasilios S. Lazarou
Logic Programming Section, Department of Computing
Imperial College, 180 Queen's Gate, London SW7 2BZ
{K.L.Clark, V.Lazarou}@ic.ac.uk

Abstract

In this paper a novel approach concerned with the general framework of Information Management, is presented. We use a Multi-Agent System to cope with the problem of Distributed Information Retrieval. The Distributed Information Retrieval task deals with the collection of information from multiple and usually heterogeneous information sources that exist in a distributed environment, which in our case is the World Wide Web.

1 Overview of the system's architecture

The advent of large wide-area networks, Internet is the most characteristic example, has caused a vast increase both in the information availability and in the number of the information sources. This evolution offers great promise for obtaining and sharing diverse information conveniently. However, the multitude, diversity and the dynamic nature of on-line information sources make accessing any specific piece of information an extremely difficult task.

One way to address these issues is to use information agents. These Distributed Information Retrieval agents should be able to:

- accept a request from a human or agent client,
- translate this request into a language understood by the information sources,
- identify the information sources that contain information relevant to the request,
- pose the request to these sources,
- collect the corresponding results,
- process the returned results and
- present the results to the client.

We have followed this approach in developing our information retrieval system for the WWW. The overall agent architecture is as follows (see Figure 1). The inter-agent communication is based on standard Knowledge Query Manipulation Language (KQML) performatives [Patil 94]. Our system supports a collection of information sites. The notion of an information site is used to describe a logical entity that contains a set of information sources. It is a logical clustering of actual-physical WWW sites.

In each information site, we find the ***extractor agent*** and the ***information source agent***. The extractor periodically scans through all the information sources, represented as URLs. These can be URLs of the top-level web pages of various research groups, for example. The extractor traverses through all the *local* documents (e.g. documents belonging to that research group) that are accessible via a chain of links from the top-level page. It classifies each such page as 'interesting' or not and extracts from each 'interesting' web page the key features and represents these features in a relational/attribute-based form. For example, it will describe an identified research paper in terms of attributes like authors, title, topics, keywords, document location (URL), abstract of document location (URL) and referenced authors.

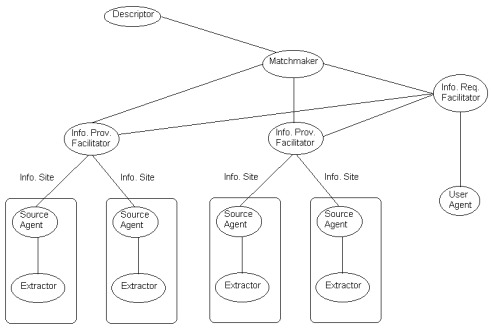

Figure 1

Finally, these important features are passed to the information source agent. The information source

agent handles the query answering process. It accepts retrieval enquiries and attempts to evaluate them against the attribute-based information. It acts as an information gateway to the information sources it manages.

In most cases, we envision a structure where the extractor and information source agents are located in the same local network as the information sources they manage. However, this is not an architectural requirement but an efficiency consideration.

The information source agent also registers an abstract of the attribute-based information it contains with an ***information supply facilitator agent***, for the authors and topics it covers. This facilitator manages semantically similar information as sent from the various source agents. For example, an information supply facilitator agent that manages the general area of AI will generally receive queries to do with any subtopic of AI and will route it to the appropriate information source agents that have registered with it. Finally, each information supply facilitator, in turn, advertises its capabilities with a ***matchmaker***, the corner stone of the distributed retrieval system.

In contrast to the above, the ***user agent*** is the one that the end user interacts with. It formulates the user's query, entered via a web browser form, translates into an appropriate query message format and displays the answers.

The user agent makes use of the services of a corresponding ***information request facilitator agent***. This facilitator accepts requests from user agents. It has the role to identify which information supply facilitators have the potential to satisfy this request through the information source agents that have registered with them. The request facilitators initially find out about information supply facilitators via queries to the matchmaker. Thereafter, they maintain direct information about these supply facilitators and the individual information sources that they manage. This is based on queries that have been successfully answered by them. Metadata caching and query planning are among other activities of an information request facilitator.

Concluding with the overall agent architecture, there are two other agents: the ***matchmaker agent*** and the ***descriptor agent***. The matchmaker serves as an advisor agent that facilitates the diffusion of the requests to agents that have expressed an ability to handle them. This is performed by accepting advertisements from supply facilitators and recommendation requests from request facilitators. The descriptor contains terminological knowledge that is exploited both by the extractor during the elicitation of the key attributes and the information request facilitator for potential query reformulation in order to overcome ontological differences between the various agents. This knowledge is provided by an external thesaurus (like WordNet).

2 The roles of the agents

The behaviour of the various agents will be illustrated below in a more detailed way. The extractor will be described using example(s) that concern the initialisation phase of our system. The remaining agents will be presented using example(s) that concern the query-processing phase.

2.1 Extractor Agent

The tasks to be performed by the extractor include detecting which web pages contain relevant information, extracting this information and representing it in an attribute-based format.

The input of this agent includes primarily a set of top-level terms that outline the ontology of the agent and a set of 'top' URLs. These URLs identify the site (web page(s) of a research group for example). For our technical report application, this set of terms describes some particular scientific subjects that are expected to cover the area(s) of the various publications to be found at the site. This set is not meant to be exhaustive. Upon initialisation, the extractor asks the descriptor to supply it with synonym topic names and the subtopic names.

The above URLs included in the given set, are noted as 'top' since we require that all the relevant information for this specific site be locatable in URLs accessible from the Web page(s) corresponding to the URL(s). Local private documents not generally accessible via the WWW can also be taken into account. In any case, SGML, HTML and text pages/files will be examined by the extractor. Postscript and other formats will also be examined after being converted to text.

The extractor elicits the links-references of each page, discards the ones that do not match the name or type preconditions named above and fetches the next page. The process stops when all the pages have been traversed. One heuristic that the extractor uses during the classification of interesting pages is that a heading like publications, reports, papers and bibliography are very good indications of an 'interesting' page. Similarly, if the 'reports' term appears in a link, this is also an adequate indication. Another heuristic used is that terms like proceedings, conference, workshop indicate a publication type. If the nearby text contains strings that are names and titles, the page is also approved.

The extractor goes through the pages of interest to spot any textual portions that are used to describe some technical paper. First, the page is partitioned into smaller textual parts. In our example page, list tags are used to separate conceptually different portions of text. Other HTML formatting information like paragraph tags, horizontal rules or empty lines can also be used. Note, however that for plain text pages, the formatting information is limited so the heuristics to be applied are constrained.

Then each small textual part is examined. The appearance of a term like publication, bibliography, papers and reports usually guarantees that the portion to come will have paper-related information. Other portions are initially tested to see if there is a chance to include such information (names, title-like strings or terms like workshop, conference and proceedings are used as hints). The attributes that we seek to extract are the authors, the title, the type and the location of the report and its abstract

Finally, the detection of the above attributes is performed with the assistance of other formatting information. For instance, authors are often enclosed in address tags; the titles in named anchors while the locations are always in reference links. Other possibilities include citations and emphasis physical formatting for titles, name-like strings for authors and URL-like strings for locations.

The topic of a report is extracted from the title. It is done by looking for terms inside the title that match the given topics. For example after the extractor has elicited the title "A novel deductive query processing technique" then the inferred topic is "Deductive Databases" assuming that additional information about this topic is provided by the descriptor. The information about referred authors, titles etc., is classified in a similar manner looking into the bibliography or references section of an actual document. After all the pages are examined, this attribute-based, relational information is passed to the information source agent (for the format of this information, see the section about the source agent).

2.2 User Agent

The user enters the query in a Prolog-like form and sets various arguments that determine a customised processing. The predicates that the user may include in the query belong to a predefined, supported set and they correspond to the most significant attributes of a technical report. This set contains the authors, the title, the topic, the document_type, keywords and referred authors and titles. An example query is:

```
document(?D)
and document_type(paper, ?D)
and {author(["KLC"],?D)
and (topic(["CSCW"],?D)
or topic(["AL"], ?D))
and referred_author(["SG"], ?D)
or author(["NS"], ?D)
and topic(["CIS"], ?D) }
```

This a query requesting URLs of paper documents which are either authored by "K.L.Clark" (KLC) and are related to the topic CSCW or the topic Agent Languages (AL), and include "S.Gregory" (SG) as a reference. Alternatively, they can be authored by "N.Skarmeas" (NS) and related to the topic Cooperative Information Systems (CIS). The various values are abbreviated in the above query (and the following ones) just for illustration purposes. During system operation, the values have to be given in full.

The various options for the retrieval process are the following. One is used to specify the depth of the query processing by the information source agents that will eventually receive the query. The first option is that the source agents need only to access their attribute database when attempting to answer the query. The other two search level options indicate that additional search may be performed. Hence, whenever a query cannot be answered by the information contained in the attribute database, any keyword terms mentioned in the query will be used to perform a keyword search on the text of the abstracts and documents correspondingly.

Another option indicates if the results are to be presented back to the user "all-together" or "one at a time". There is also an upper bound on the number of answers that the user desires to see.

Another option turns the current query into a persistent request. For a persistent request, the user will be automatically notified of any new documents that satisfy the query. This is performed by sending a mobile agent, fed with the current query, to the sites of the source agents that answered the query successfully. This mobile agent has a monitoring role. It periodically interrogates the source agents for any new documents, collects any that satisfy the query and propagates them back to the user agent.

A direct addressing choice enables the user to specify the identities of certain source agents that will be able to help in the answering of the current query. This can happen if for a previously posed query (similar to the current one), some source agent returned highly relevant results.

After submission of the query, the user agent transforms it into a KQML-style message structure to allow easy inspection and transformation by other agents. The transformed query is then sent to the corresponding information request facilitator that typically resides physically close to the user agent.

The results are displayed through a web interface via an HTML page.

2.3 Information Request Facilitator

The information request facilitator agent is in charge of organising the query answering process. It accepts queries from user agents and attempts to find which information supply facilitators have the potential to help answer part of or the entire query; in other words, it acts as a query planner. Some of the activities it performs include query reformulation, consulting and subscribing to the matchmaker, and metadata caching. These roles will be explained through the use of our example.

Suppose the query given above has arrived at the information request facilitator to be processed. If this mentions authors or topics that the facilitator has not previously encountered, it will ask the matchmaker to match the attribute values that are contained in the query with corresponding advertisements by supply facilitators. The recommend request sent to the matchmaker would contain the following query:

```
info_supply_facilitator(?SF)
and {author(["NS"],?SF)
or topic(["CSCW"],?SF)
or topic(["AL"], ?SF)
or referred_author(["SG"], ?SF))
or author(["KLC"],?SF)
or topic(["CIS"], ?SF)}
```

This request indicates that any match on the attribute values existing in the original query is sufficient to make it useful to forward the original query to any supply facilitators that have advertised that attribute value. Recall that the matchmaker holds advertisements that are abstracted forms of the information processing capabilities of the information supply facilitators. Therefore, if we insisted on a request identical or equivalent to the original query, we would be in great danger having a very low success ratio because of some information requests that can be handled but are not advertised. Additionally, the supply facilitators (and after them the source agents) will actually receive and process the original query (or an equivalent one). Thus, there is no danger of obtaining answers irrelevant to what the query expresses because of the above looser form. Alternatively, the request facilitator attempts to satisfy the query using information previously supplied from the matchmaker.

Note that during the processing of a match between author names, we do not insist in exact match, a "fuzzy" match is sufficient. The notion of fuzziness in this case declares that a name like "K.L.Clark" as exists in our example query, will successfully match any names, like "Keith L. Clark", "Keith Clark", "Keith Lee Clark", "K. Lee Clark" etc. In other words, the surname has to be identical while the first name(s) have to be identical only if given in full. If only initials are provided, the first name(s) that begin with these initials are considered as successful matches. In conclusion, if the matchmaker succeeds then the query is passed to the supply facilitators detected. However, the query that is passed does not have to be identical to the original one, as will be explained below in the matchmaker section.

2.4 Matchmaker & Descriptor

Let us now consider matching of topics as performed by the matchmaker. Topics as scientific subjects can be classified in a conceptual hierarchy in contrast to the name values. A topic can have synonym topics, several subtopics (topics used to describe a subject more specific than the current one) and supertopics (topics used to describe a subject more general than the current one). For example, the topic of Multi-Agent Systems can be referred also as Intelligent Distributed Systems and Agent Software, has as immediate subtopics WWW agents and Interface agents while it belongs to the immediate supertopic of Distributed AI. It should be obvious that these other topics can have synonyms and sub or super topics of their own.

The matchmaker should consider this conceptual hierarchy when it receives a recommend message by a request facilitator. As mentioned in the general system overview, the descriptor agent is the one that holds terminological knowledge capable of identifying synonyms, hypernyms and hyponyms of a specific term. Therefore, when a recommendation request arrives to the matchmaker containing attribute values that concern topics, the matchmaker has in turn to consult the descriptor about the conceptual placement of this topic in the topic hierarchy.

If for example the recommendation request involves arriving at the matchmaker the topic of Multi-Agent Systems all its synonym topics and subtopics will be identified by querying the descriptor. For the supertopics only the ones that have a maximum difference of two levels will be returned, so only the "parent" topic Distributed AI and the "grandparent" topic AI will be returned. Note however, that the above querying will take place only for topics that have not already been encountered. This is because the matchmaker caches information that comes from the descriptor in order to avoid consulting it continuously.

The matchmaker, during the process of matching incoming requests with advertisements of supply facilitators, will use all the related topics. Moreover,

when it answers the recommendation enquiry it indicates which topics are supported from the corresponding supply facilitators as well as the relation of these topics to the ones mentioned in the recommendation enquiry. Therefore, if for example the matchmaker finds one supply facilitator that supports the "Mobile Agent Languages" topic (which is a subtopic of "Agent Languages") and another one that supports the "AI Languages" topic (which is a supertopic of "Agent Languages"), it will indicate this in its response. Consequently, when the request facilitator receives this response it will reformulate the original query sending two different variations of it. To the first supply facilitator, it will send a query which is the same as the original one except that instead of having the "Agent Languages" topic it will have the "Mobile Agent Languages" one. Similarly for the second the topic, "AI Languages" will replace the "Agent Languages" one.

The relationship (synonym, hyponym, hypernym) between any requested topic and the actually supported ones indicated in the matchmaker response, will be forwarded to the user agent. The user agent will take into account this information when presenting the final results to the user. Results for identical or synonym topics will be presented first as they match the original query most closely. Results for subtopics will follow by placing the immediate subtopics first and then subtopics of subtopics and etc. Finally, results for supertopics will be left at the end since they match approximately the original query. Again, immediate supertopics will have greater display priority.

2.5 Information Supply Facilitator

Let us see now what happens when an information supply facilitator receives a query from the information request facilitator. Prior to any request processing, during the set-up of the system, the information supply facilitator accepts registrations from source agents. When they register, the source agents send an abstracted form of the information they possess. This summary information includes the attribute values that occur in a repeated manner in their information bases. Furthermore, the most frequent part of that information will be advertised to the matchmaker that covers a more general scientific area.

Now when a query arrives to the information supply facilitator, the content of its summary information has to be examined in order to identify which of the registered source agents contain information to potentially answer the query. This operation will be performed using a simplified form of the original query where only the attributes that can have repeated values will be incorporated (authors and topics in our case) since these are the ones included in the summary information. So for our example this simplified form will look like:

```
document(?D)
and {author(["KLC"], ?D)
and {topic(["CSCW"],?D)
or topic(["AL"], ?D)}
and referred_author(["SG"], ?D)}
or {author(["NS"], ?D)
and topic(["CIS"], ?D)}.
```

If this query can be answered against the summary information of the supply facilitator, then the original query will be routed to the source agents that are responsible for the successful outcome. Otherwise, the information request facilitator will be notified of the failure.

2.6 Information Source Agent

The final destination of our example query is the information source agent (or briefly source agent). It now has to be evaluated against the content of its information base. After the extractor finishes its classification processing, it feeds the source agent with a structured representation of the underlying information site contents. This representation is in an attribute-based form. For the technical reports application the schematic representation is composed by the following relations:

```
document_location(doc_id, URL)
abstract_location(doc_id, URL)
author(doc_id, authors)
title(doc_id, title)
topic(doc_id, topics)
document_type(doc_id,doc_type)
keyword(doc_id, keywords)
referred_author(doc_id,authors)
referred_title(doc_id, title)
```

The document descriptions that provided a successful query evaluation will be returned to the user agent. If however the evaluation does not produce any successful results, then the supply facilitator will be notified of this failure. If in addition, the supply facilitator receives failures from all its source agents selected for this query, it will report a more general failure back to the request facilitator. Concluding, if the request facilitator receives general failures from all the supply facilitators, then the user agent is notified of a complete failure.

3 Extensions and related work

Very briefly, the important directions for future extensions are:

- Multimedia information support, multimedia data modelling and retrieval methods
- More sophisticated use of mobile agents
- User modelling and elaborated learning methods
- Ecology of agents as an architectural complement
- Advanced use of constraints

The existing Distributed Information Retrieval systems have a significant influence in the architectural as well as the implementation features of our system. The concept of some of the agent classes with a role similar to ours exists in UMDL (such as the collection interface agents) [Birmingham 95] and [Vidal 95], IRA (such as userbots, corpusbots) [Voorhees 94], TSIMMIS (such as classifiers, translators) [Garcia-Molina 95], Information Brokers [Fikes 95], SHADE-COINS [Kuokka 95] and Knowledge Navigator (such as advisory agents) [Burke 95] among others. Other research work on multimedia information [Gudivada 95], [Marcus 95a], [Marcus 95b] and [Sistla 95] and for Information Retrieval techniques [Crowder 95] is directing some future aspects. For aspects specific to our agents' functionality, work from [Borghoff 96], [Espinoza 96], [Goldman 96], [Moukas 96] and [Turpeinen 96] has an important role.

All the above systems tend to focus on specific aspects of both the system's architecture and the agents' responsibilities. The UMDL, IRA systems focus on the mid-level activities providing only agents similar to our source, user and facilitator agents. The TSIMMIS one examines mainly the retrieval procedure; it does not incorporate for example the basic user interface agent class. The SHADE-COINS is the only one that provides the high level class of the matchmakers but leaves an architectural gap caused by the lack of facilitators. The Knowledge Navigator has a different orientation since it adopts the browsing paradigm while the Information Brokers system appears inflexible by assigning all the activities into one agent class.

4 Summary

- Sites managed as local deductive databases and WWW as a distributed deductive one.
- Prolog-like query language provides expressiveness and accuracy concerning the user needs.
- The predicate set for query formulation corresponds to characteristic document properties.
- Completely precise answers.
- The semantics behind the used terms is captured; polysemy and synonymy are tackled.
- Fully distributed, scalable and modular system; the information providers are not passive request servers.

Bibliography

[Birmingham 95] Birmingham, W. P., Durfee, E.H. The Distributed Agent Architecture of the University of Michigan Library. In AAAI95 Symposium: Information Gathering from Heterogeneous, Distributed Environments.

[Borghoff 96] Borghoff, U. M., Karch, H., Schlichter, J. H. Constraint-based Information Gathering for a Network Publication System. In PAAM 96.

[Burke 95] Burke, R., Hammond, K.J. Combining Databases and Knowledge Bases for Assisted Browsing. In AAAI95 Symposium: Information Gathering from Heterogeneous, Distributed Environments.

[Crowder 95] Crowder, G., Nicholas, C. An Approach to Large Scale Distributed Information Systems Using Statistical Properties of Text to Guide Agent Search. In CIKM95 IIA Workshop.

[Espinoza 96] Espinoza, F., Hook, K. A WWW Interface to Adaptive Hypermedia System. In PAAM 96.

[Fikes 95] Fikes, R., Engelmore, R. Network-Based Information Brokers. In AAAI95 Symposium: Information Gathering from Heterogeneous, Distributed Environments.

[Garcia-Molina 95] Garcia-Molina, H., Hammer, J. Integrating and Accessing Heterogeneous Information Sources TSIMMIS. In AAAI95 Symposium: Information Gathering from Heterogeneous, Distributed Environments.

[Goldman 96] Goldman, C. V., Langer, A., Rosenschein, J. S. Musag: agent that learns what you mean. In PAAM 96.

[Gudivada 95] Gudivada, V. N., Raghavan, V. V., Vanapipat, K. A Unified Approach to Data Modelling and Retrieval for a Class of Image Database Applications. In Multimedia Database Systems, Issues and Research Directions, Springer-Verlag.

[Kuokka 95] Kuokka, D., Harada, L. Supporting Information Retrieval via Matchmaking. In AAAI95 Symposium: Information Gathering from Heterogeneous, Distributed Environments.

[Marcus 95a] Marcus, S., Subrahmanian, V. S. Towards a Theory of Multimedia Database Systems. In Multimedia Database Systems, Issues and Research Directions, Springer-Verlag.

[Marcus 95b] Marcus, S. Querying Multimedia Databases in SQL. In Multimedia Database Systems, Issues and Research Directions, Springer-Verlag.

[Moukas 96] Moukas, A. Amalthea: Information Discovery and Filtering using a Multi-agent Ecosystem. In PAAM 96.

[Sistla 95] Sistla, A. P., Yu, C. Retrieval of Pictures Using Approximate Matching. In Multimedia Database Systems, Issues and Research Directions, Springer Verlag.

[Turpeinen 96] Turpeinen, M., Saarela, J., Puskala, T. Architecture for Agent Mediated Personalised News Services. In PAAM 96.

[Vidal 95] Vidal, J. M., Durfee, E. H. Task Planning Agents in the UMDL. In CIKM95 IIA Workshop.

[Voorhees 94] Voorhees, E. Information Agents. In AAAI 94 Spring Symposium: Software Agents.