Handling Information Overload: A MAS Architecture For Distributed Information Brokering

Klaus Stein, Christoph Schlieder
Laboratory for Semantic Information Processing
Otto-Friedrich-Universität Bamberg
{klaus.stein,christoph.schlieder}@wiai.uni-bamberg.de

1. INTRODUCTION

The gathering of information, exploration of information resources etc. is a longstanding problem and a well known agent task (from personal search agents [4] over distributed information retrieval systems [5] to autonomous agents exploring dynamic document networks [3]). An information retrieving agent not only faces the problem of acquiring information but also of selecting the interesting documents.

m agents retrieving information in a large set of n documents (messages on a discussion blackboard, websites in an intranet, documents in a CMS, ...) on their own gives far to many $(m \cdot n)$ read requests. We propose to reduce this complexity we introduce an intermediate level of preprocessing information broker agents for efficently selecting all documents matching a given search query from a document network (documents referencing each other, e. g webpages with links or papers with citations) and ranking these documents using the document network structure (i. e. sorting them by relevance).

2. SYSTEM ARCHITECTURE

Fig. 1 shows two types of cooperating information broker agents being helpful by answering search queries: the **network information broker agent** (NIBA) managing a (continuously updated) index of all the documents contents and meta information of a given repository for fast access by search terms and a preprocessed representation of the network structure with fast access to the sets of documents referring (R_a) and being referred (C_a) by any document p_a, and the **rank information broker agent** (RIBA) handling queries from an information searching agent by using the structured and preprocessed information provided by the NIBAs to calculate the structural ranks r_i of the documents. The common way to compute these ranks is to start with some initial values $r_{i,0}$ for each document and then iterating several (j) times distributing r_i along the edges in each iteration (see e. g. the journal ranking by Pinski and Narin (later adapted to google pagerank [1]), Kleinbergs hub-authority-approach [2] etc. As there is not one *true* ranking of documents but different rankings depending on the needs of the searching agents, different RIBAs with different algorithms are provided.

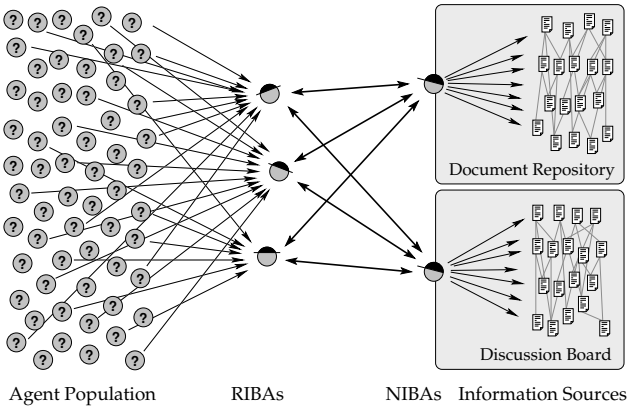

Figure 1: multi agent information selection

A *preprocessing* RIBA does a structural ranking of all documents which is cached and updated any time the RIBA is informed about changes within an information source by the associated NIBA. Due to the preprocessing searching is fast and cheap: the searching agent sends its query to the RIBA which passes it to the NIBAs. The NIBAs return a list of matching documents, the RIBA adds the corresponding ranks from its cache returning this enhanced list.

An *on-demand processing* RIBA works by selecting a subnet consisting of the documents $D' = D \cup (\bigcup_{p_k \in D} R_k) \cup (\bigcup_{p_k \in D} C_k)$ with D the subset of documents returned by the NIBA (the subset of documents matched by the search query). By computing the structural ranking on this subnet only documents related to the search term contribute giving more specific results. This increases search costs for the ranking has to be computed for each query. This also allows to incorporate thematic information (e. g. in setting the $r_{i,0}$) in the ranking by further increased costs because the documents contents have to get processed.

Altogether a complete search runs as follows: the NIBAs return a list of matching documents, the RIBA adds the corresponding ranks from its cache (or computes the structural ranking on the subnet D') and returns this enhanced list to A. A can customize its search by choosing one from many RIBAs computing rankings with different algorithms

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
AAMAS'05, July 25-29, 2005, Utrecht, Netherlands.
Copyright 2005 ACM 1-59593-094-9/05/0007 ...$5.00.

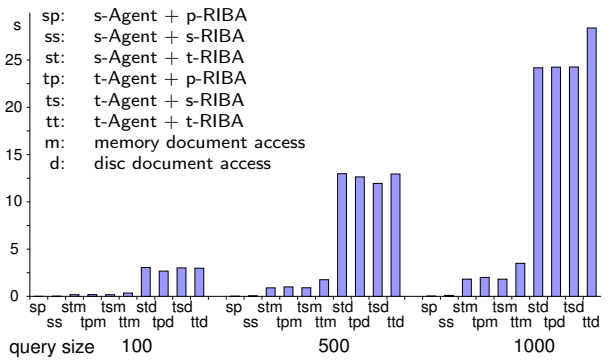

Figure 2: Average query time of different agent-RIBA-combinations

or different parameter sets. The thematic ranking is done by the searching agent A itself.

On a document reference network with n documents and a search query returning k documents the costs for each query depend on the setting: an agent A searching by itself without using a search agent gives $O(n)$. A NIBA with a preprocessed index can answer a search query in $O(k)$ (perfect hash) to $O(k + \log n)$ (search tree/hierarchical hash). A preprocessing **p-RIBA** adds structural information in $O(1)$ and A does thematic ranking in $O(k)$ and sorts the documents in $O(k \log k)$, so we get $O(\log n + k \log k)$ for a complete search query.

A query specific on-demand ranking (**s-RIBA**) with a small number of iterations j' on a subnet D' gives $O(j'|D'|)$. We estimate $|D'| = |D|\bar{l} = k\bar{l}$ (with $\bar{l}$ being the average number of references per document) and get $O(j'k\bar{l})$ at search time (with constant j' and $\bar{l}$: $O(k)$).

3. SIMULATION

While the described scenarios do not differ much with respect to complexity, they have big differences in run time. We implemented a prototypical MAS with different kinds of seaching agents using different kinds of RIBAs on a document network with $n = 1000000$ documents (files on the local harddisk) with an average size of 700 bytes and an average number of references $\bar{l} = 10$, which is handled by a NIBA answering a search query by a list of URIs (filenames) of matching documents. The whole MAS was implemented in Ruby and run on one single computer in Linux single user mode (no other processes/daemons running).

Searching is done by using three different kinds of RIBAs: the preprocessing **p-RIBA** keeping a list of the preprocessed ranks of all documents, the structural on-demand ranking **s-RIBA** creating and ranking the subnet D' for each new search query, and the thematic-structural on-demand ranking **t-RIBA**, working similar to the s-RIBA but addionally ranking the documents in D by content. s-RIBA and t-RIBA do $j' = 4$ iterations for each ranking. The queries are done by two different kinds of search agents: the **s-Agent** sort the documents by the ranks given from the RIBA and the **t-Agent** agent combining the given ranking with an own thematic ranking to sort the documents.

Now for any combination of search agents and RIBAs the time for a query returning 100, 500 or 1000 documents is measured (see Fig. 2; the given timings are averages over several queries). Any thematic ranking needs to read the documents contents with disk IO dominating all other timings, therefor we run our tests in two modes: memory mode (m), where all documents are kept in memory and no disk IO occours, and the disk mode (d), where all documents are read from disk any time needed.

The data (Fig. 2) collected from these simulations shows the increase in cost from preprocessed (sp) over on-demand structural (ss) to on-demand thematic ranking (stm, tpm, tsm and ttm), with the most expencive cases being where disk access is involved (std, tpd, tsd and ttd). The worst case here is the combination t-Agent and t-RIBA, because here all documents have to be read twice, nevertheless ttd does not take twice the time of tpd (while ttm needs twice the time of tpm). This is caused by the operating system disk cache. We additionaly did tests with cache switched off and got doubled runtime for ttd (not shown in the graph).

As expected the thematic ranking is expencive, even if no disk access is involved, and even if the query size (the number of documents to be parsed to do the thematic ranking) is small ($k = 100$).

Generally search time increases linearly in query size within the given range (we did some tests with $k = 10000$ which seem to hold this). The structural on-demand-ranking takes five times longer than using preprocessed ranks (ss vs. sp), thematic ranking causes 80 to 100 times longer runtime (stm, tpm and tsm to sp), double thematic ranking doubles this (ttm) and any disk access involved gives an additional factor of 15, so the slowest combination t-Agent and t-RIBA with disk access is more than 1000 times slower than the fastest one (s-Agent and p-RIBA). And all this is magnitudes better than a single agent scanning the document network by itself.

4. CONCLUSION

The MAS architecture for distributed information brokering provides a flexible approach to enable search agents to handle the problem of information overload. Using the distributed structure of the information brokering system the searching agent benefits from the speed-up gained by using preprocessing and search engine technology for information retrieval provided by the NIBA while keeping flexibility and control over the ranking process by choosing one of several RIBAs and doing part of the ranking by itself.

5. REFERENCES

[1] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[2] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[3] F. Menczer and R. K. Belew. Adaptive information agents in distributed textual environments. In *Proceedings of the second international conference on Autonomous agents*, pages 157–164. ACM Press, 1998.

[4] G. Somlo and A. E. Howe. Querytracker: An agent for tracking persistent information needs. AAMAS 2004, pages 488–495.

[5] H. Zhang, W. B. Croft, B. Levine, and V. Lesser. A multi-agent approach for peer-to-peer based information retrieval system. AAMAS 2004, pages 456–463.