A SCALEABLE AND FAULT-TOLERANT ARCHITECTURE FOR DISTRIBUTED WEB RESOURCE DISCOVERY

Anders Fongen
Norwegian School of Information Technology, Hans Burums v.30,
1357 Bekkestua, Norway
Frank Eliassen
Simula Research Laboratory, P.O.Box.134,
1325 Lysaker, Norway

Ian Ferguson
Univ. of Strathclyde, Livingstone Tower, 26
Richmond Street, Glasgow, G1 1XH, UK

Simon Stobart, John Tait
Univ. of Sunderland, St. Peter's Campus,
Sunderland, SR6 0DD, UK

Keywords:

Large scale distribution, distributed resource discovery, peer-to-peer networking

Abstract

Most Internet search engines are built on a centralised design, and will therefore not cope with the expected future growth in information and query volume.

A distributed approach to Internet resource discovery seems necessary. Many distributed designs have been proposed, but their scaleability is largely unknown.

We propose a distributed design based on the use of classified metadata, which can be proven to be extremely scaleable, and has interesting fault-tolerant properties. We will show the principles of this design, called the Content-Sensitive Infrastructure (CSI), and demonstrate its properties through formal analysis and simulation.

Introduction

The use of Internet search engines is increasing steadily at an exponential rate [1]. As Internet appliances start to surf the Semantic Web [2], the number of queries and volume of resource information will grow beyond what can be processed by monolithic search engine sites: The networking resources needed to process queries and to keep indices up-to-date grows beyond what is realistic to find at a central site [3, 4]. Current search engines seldom index more than 15% of the total web content [5], which can be an indication of a shortage of network resources.

A partitioning of the indices associated with a search engine across several sites can contribute to a distribution of the network traffic generated from queries. In order to route queries to the sites that are holding relevant information, *forward knowledge* is required. Forward knowledge is information describing the content of an index in a condensed form. The entities acting on this information are often called "Query Routers" or “Brokers” [6, 7], and the volume of the forward knowledge that needs to be transported to the Query Routers is critical to the system's ability to scale. It has been shown [8] that the volume of forward knowledge will grow exponentially over time if all the indices hold "general" information (not linked to a specific topic).

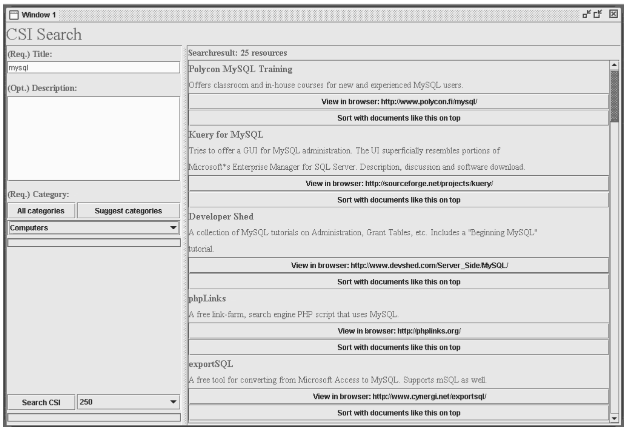
To cope with this problem, the CSI is based on the idea that search engine indices can be partitioned and distributed according to the *topic* of the resources that they describe, and that queries can be routed towards the relevant sites based on their topical property. Since the topical property of an index partition does not change, the forward knowledge becomes *static*, which improves the scaleability of the architecture. The design relies firmly on the fact that both information and queries can be usefully located in a *classification system*, i.e. a set of related topics. The CSI is based on the use of *metadata resource descriptions*, since full-text indices are hard to distribute using a classification system.

The remainder of this paper is organised as follows: The first part presents the design and protocols of the CSI. The second part presents an analysis of the CSI’s properties. The paper then moves to an empricial review of the classification process and of related work. Finally possible further work is proposed

The CSI in use

Using a User Agent, one can pass queries into the CSI and have the response presented as a ranked list of resources, like many other search engines. However the query is analysed and classified, and the user can accept that suggestion or choose another topic.

The User Agent is not the focus of this paper, but included here only for illustration.


Window 1
All categories
Suggest categories
Search CSI
View in browser: http://www.polycon.fi/mysql/
Sort with documents like this on top
View in browser: http://sourceforge.net/projects/kuery/
Sort with documents like this on top
View in browser: http://www.devshed.com/Server_Side/MySQL/
Sort with documents like this on top
View in browser: http://phplinks.org/
Sort with documents like this on top
View in browser: http://www.cynergi.net/exportsql/
Sort with documents like this on top

CSI Search

(Req.) Title: mysql

(Opt.) Description:

(Req.) Category: Computers

250

Searchresult: 25 resources

Polycon MySQL Training

Offers classroom and in-house courses for new and experienced MySQL users.

Kuery for MySQL

Tries to offer a GUI for MySQL administration. The UI superficially resembles portions of Microsoft*s Enterprise Manager for SQL Server. Description, discussion and software download.

Developer Shed

A collection of MySQL tutorials on Administration, Grant Tables, etc. Includes a "Beginning MySQL" tutorial.

phpLinks

A free link-farm, search engine PHP script that uses MySQL.

exportSQL

A free tool for converting from Microsoft Access to MySQL. Supports mSQL as well.

Figure 1: Screen shot from the User Agent

CSI Members

The building block of the CSI is the *Member*, which is a forwarding and processing entity able to handle resource information (as metadata) and queries. The only externally visible properties of a member are its network endpoint and the set of information topics it is willing to observe (i.e. store information and process queries). The topics in question are described by a hierarchical classification system, and the member chooses a branch of the category tree by declaring its *Category of Interest* (COI).

Through a network protocol where the members inform each other about their presence and their COI, the members form a network where "areas" of this network map to "topics" and vice versa. In other words, a given topic “belongs” to a coherent part of the network. Resource information on the subject of a certain topic will be stored inside the area of this topic, where the members reside that have this branch of topics covered by their COI. The network of members is completely self-configuring, and members can enter or leave the network at any time without disrupting the operation of the system.

When members exchange metadata the information flows towards the areas of the network where there exist members interested in this particular topic. A member that receives metadata on a topic that is relevant to it will store this piece of metadata in permanent storage for later query processing.

Queries are also associated with topics, and will be forwarded towards the members covering this topic. A member that receives a query "of interest" will process the query and return a set of metadata matching the requirements in the query.

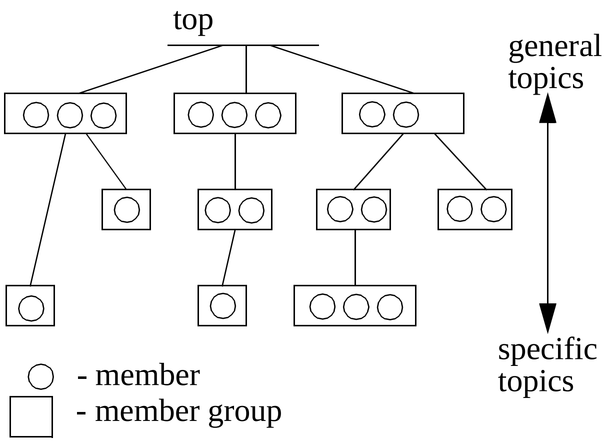


Figure 2: Members organised in the context of a classification system

Member Groups

Members having the same COI form *member groups* as shown in figure 2. Members of a group do *not* hold exactly the same metadata information, despite their identical COIs. The metadata are distributed so that each item is replicated in no more than *MAXR* members. This property is important for the scaleability of the system (see later). The formation of member groups also contributes to the dependability of the CSI, since groups with two or more members form alternative forwarding paths through the network.

The different member groups form a *tree*, where the relations between their respective COIs determine the parent-child relationships between them.

A query must be forwarded to every member of a group, and to accomplish this in a scaleable manner a *redundant multicast algorithm* is employed: The members of a group are sequentially ordered, and the member with ordinal number *m* will forward the query to the members numbered *(2m-1)* to *(2m+4)*, as shown in figure 3.

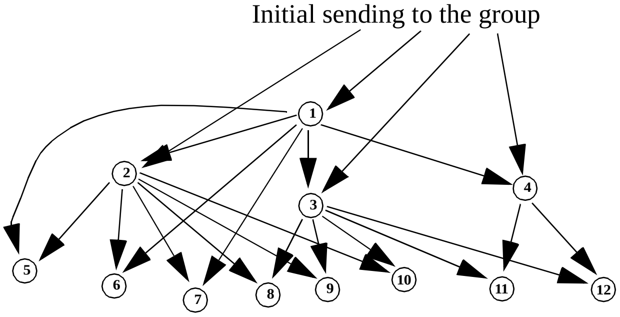


Figure 3: Multicast distribution of queries in a member group

Using this algorithm, the member group is organised as a tree of degree 3, and any member receives at most 2 copies of a query, and forwards at most 6 copies, regardless of the size of the group.

The forwarding protocols

The rules governing the forwarding of metadata and queries are straightforward:

1. Any query or item of metadata is forwarded at most once. Duplicates are recognised and discarded.
2. A query having the same COI as the member's COI is forwarded to members of the same group using the redundant multicast algorithm described above.
3. A query having a COI that is a sub-category of the member's COI is forwarded down the tree of member groups, towards the member group having the same COI as the query. When passing a query to a member group, the four members in the group with the lowest order are addressed, the multicast operation of (2) does the rest.
4. Metadata are forwarded down the tree of member groups towards that group with the same COI as the metadata, but not further. When forwarding metadata to a member group, *MAXR* members of the group are randomly chosen and addressed.

These rules enforce the following policy:

For a query with a given COI, all metadata having the same COI (or any sub-category COI) is of interest.

The optimal *migration path* for an item of metadata is from the "top" of the tree of member groups, i.e. the group of members with interest (COI) in a top category, down to the member group having the same COI as the metadata. The optimal destination for a query is the single member group having the same COI as the query. The migration paths may not be optimal in practice, due to the following reasons:

- There are no members having exactly the same COI as the metadata or the query. In this case, the parent member acts on behalf of the non-existent child member.
- The members do not have complete information about the infrastructure. Partial information may cause a query to start its migration higher up in the tree than necessary.

The Helper Member. Sending queries and metadata to the starting point of their migration path is the responsibility of a *helper member.* The User Agent needs to operate in the CSI through a helper member, and this role is evenly distributed among the CSI members through a randomised approach.

The scaleability of the CSI

An important characteristic of the CSI is how the number of network messages sent and received by any single member is affected by the number of members present in the infrastructure. In the following discussion we will refer to this effect as the *message complexity* of the member.

Since the processing and forwarding of metadata and queries are atomic and unrelated events, it is a trivial observation that the message complexities are *O(Q)* and *O(M)*, where *Q* and *M* denotes the volume (or rate) of queries and metadata, respectively.

Distribution of members: In the following analysis the following simplifications are made:

- A member being created will have a *depth* in the topic hierarchy, between 1 and *D*. The depth is subject to a probability distribution: $P_d(d)$ *which* denotes the probability that the member is of depth *d*. Obviously,

$$\sum_{d=1}^{D} P_d(d) = 1$$

- A member can have up to *A* child member groups. At depth *d* there are A^d possible member groups. Members created at depth *d* have the same probability of occupying any position at that depth.

Therefore, the probability that a new member has a specific position in the topic hierarchy at depth *d* is:

$$\frac{P_d(d)}{A^d}$$

In a CSI with *n* members, the average size of the member groups at depth *d* is:

$$\frac{P_d(d)}{A^d} \cdot n$$

The processing of metadata: The processing of metadata involves these transport operations:

- A helper member receives one item of metadata (from a User Agent) and forwards it to a maximum number of *MAXR* members inside the target member group.
- Each member that receives an item of metadata will forward it to a maximum number of *MAXR* members in a child member group.

We now introduce a new probability function $P_m(d)$, which denotes the probability that the “depth” of the metadata (referring to its position in the topic hierarchy) is larger than or equal to *d*. Note that $P_m(1) = 1$. The use of this function is to express the fraction of *M* (metadata volume) that will be processed by one of the member groups at a given depth. At depth *d*, metadata is evenly distributed across A^d subcategories.

For a member group at depth *d*, the metadata volume that needs to be processed is:

$$\frac{M \cdot P_m(d)}{A^d}$$

Given that *MAXR* of the members need to process this volume, the fraction of *M* that needs to be processed by one single member is:

$$\frac{M \cdot P_m(d)}{A^d} \cdot \frac{\text{MAXR} \cdot A^d}{n \cdot P_d(d)} = O(M/n)$$

The metadata volume that a member needs to transport as a *helper member* is simply its equal share of the total volume M:

$$\frac{M}{n} = O(M/n)$$

The helper member receives one metadata item from the UA and needs to send the metadata to a maximum number of *MAXR* members. The member involved in the forwarding of a metadata item receives a maximum number of *MAXR* copies of an item and forwards it to a maximum number of *MAXR* child members. These constant factors do not alter the complexity expression, so we conclude that the *message complexity* in processing the metadata is $O(M/n)$, **and that the CSI is able to process any volume of metadata by adding a sufficient number of members.**

The processing of queries: The processing of queries involves these transport operations:

- A helper member receives one query from a UA, and forwards it to a maximum number of *four* members inside one member group (described previously).
- Each member that receives a query will possibly forward it to a maximum number of *six* members in the same member group, or to *four* members in a child member group.

Note that these operations involve a constant communication cost regardless of information volume (*Q* or *M*) or CSI size (*n*). It can be shown that the communication cost of processing the queries is distributed among members and that the cost for one single member decreases as the number of member groups increases:

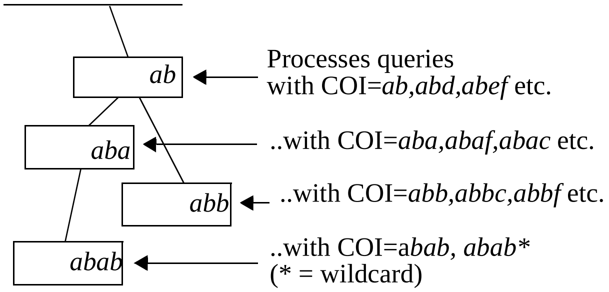

ab
Processes queries with COI=*ab,abd,abef* etc.
aba
..with COI=*aba,abaf,abac* etc.
abb
..with COI=*abb,abbc,abbf* etc.
abab
..with COI=a*bab, abab**
(* = wildcard)

Figure 4: The delegation of query processing to child members. The COI of the members are represented as a "pathname".

We introduce the probability function *$P_q(d)$,* which denotes the probability that any query has "depth" equal[1] to *d*.

A query is sent to the member group with the same COI or (in case it does not exist) its closest parent. The range of queries that a member must process is therefore dependent on the presence or absence of child member groups. Figure 4 shows an example on how the processing of queries is "delegated" to child members as they arrive in the CSI.

To determine the range of queries that must be processed, we must find the probability that a "child COI" (subtopic) has no closer parent than this member.

A member at depth *d* has the same COI for this fraction of *Q*:

$$\frac{P_q(d)}{A^d} \qquad \text{(eq.1)}$$

The fraction that should (ideally) be handled by child members is:

$$\sum_{x=d+1}^{D} \frac{P_q(x)}{A^d}$$

The probability that among *n* members, *none of them* has a given COI at height *g*, is:

$$\left(1 - \frac{P_d(g)}{A^g}\right)^n$$

The probability that one member (at depth *d*) has *no* child members that can process queries with a given subtopic at depth *g*:

$$\prod_{x=d+1}^{g} \left(1 - \frac{P_d(x)}{A^x}\right)^n = P_c(d, g, n)$$

(This function is assumed to return 1 if *(d==g)*)

The fraction of *Q* that should be processed by child members, but has to be processed by this member (at depth *d*) is:

$$\sum_{x=d+1}^{D} \left(\frac{P_q(x)}{A^d} \cdot P_c(d, x, n)\right)$$

[1] Unlike *$P_m(x)$,* this function is not "cumulative".

(Eq.1) gives the fraction of Q for which this member has “direct responsibility”, and since $P_c(d,d,n)$=1, the total fraction of Q being processed by this member is:

$$\sum_{x=d}^{D}\left(\frac{P_q(x)}{A^d}\cdot P_c(d,x,n)\right) = h(d,n)$$

A graphical representation of *h(d,n)* with *d* held constant reveals a monotonically decreasing function with a horizontal asymptote at a very small positive number. The fraction of *Q* that needs to be processed by one member is therefore decreasing with the size of *n* (number of members in the CSI):

$$O(Q\cdot h(d,n))$$

The transport operation involved in the role of a UA helper member (inserting queries into the CSI) follows the same analysis as for metadata: The work is evenly distributed between the members and each operation has a constant communication cost:

$$\frac{Q}{n} = O(Q/n)$$

We can therefore conclude that the communication cost involved in processing of queries decreases for a single member as the number of members[1] increases, **and that the CSI is able to process any volume of queries by adding a sufficient number of members.**

Simulation experiment: A simulation of the CSI has enabled the study of message complexity in an experimental context. A simulation run has consisted of the following steps:

1. Create a number of members and assign to them COIs according to a given probability distribution *P(d)*. The members inform each other about their existence through the self-configuration protocol.
2. Create a number of resource descriptions and inject them into the infrastructure, as how a User Agent would do. The number of messages sent and received in every member is counted during this phase.
3. Create a number of queries in the same manner. Messages are counted in every member.

The average number of messages were calculated for different values of *d*, and the highest averages were plotted as a function of member count shown in figure 5.

In the figure, the result of several runs are shown to illustrate the effect of varying number of members. **The plot confirms the analytical results on the scaleability of the CSI**.

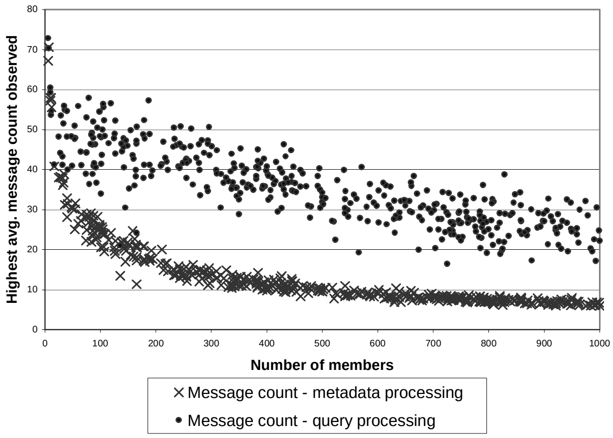


Figure 5: Number of messages in members during simulated operation of the CSI

Fault-tolerant properties

The fault-tolerant properties of the CSI have been studied by simulation. The effect of errors during message exchange can be measured by the *retrieval ratio*, which is the ratio of actual to optimal result set sizes from a query. By keeping track of the metadata injected in the CSI we can calculate the optimal result set from a query “outside” the simulation model.

Intuitively, the CSI will become more robust to errors as it grows in size: Metadata will be stored in more members, and there will be more alternative forwarding paths between members as they become more numerous. Since the design is completely without any single point of failure, we also expect the retrieval ratio to degrade gracefully as the transport errors become more frequent. The effect of crashed members has not been studied in this experiment.

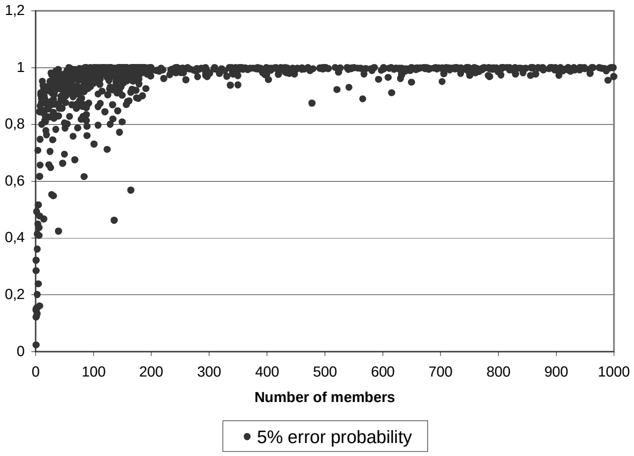


Figure 6: The retrieval ratio as a function of member count

Figure 6 shows the retrieval ratio for several simulation runs with different number of members, but with constant probability for errors during message exchange.

[1] As the analysis indicates, it is actually the number of *member groups* that contributes to the distribution of query processing.

There seems to be a “critical mass” above which the CSI operates very reliably.

In figure 7 the member count is held constant and the error probability is varied. The figure shows how the retrieval ratio drops gracefully as the errors during message exchange occur more frequently. There is no “point of collapse” to be seen, which confirms the robustness of the CSI design.

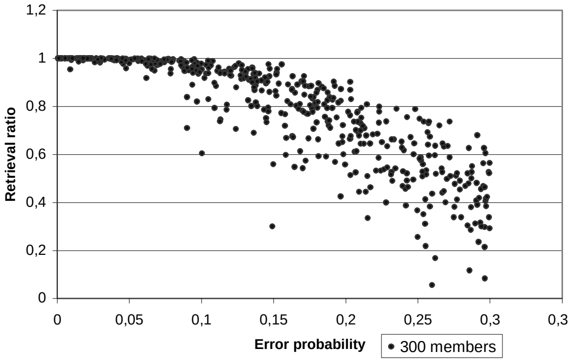


Figure 7: Retrieval ratio as a function of error probability

The classification process

The scaleability of distributed lookup services in a peer-to-peer environment has been discussed in recent papers [9, 10], where the use of hashing techniques have been utilised on order to locate a key/value pair in the network.

The CSI is not able to use hashing techniques, but must use classification techniques to locate the requested information inside a *zone* of network nodes[1]. This also enables us to store the information redundantly.

The classification experiment

Automatic classification is extensively researched, but automatic classification in the general domain (e.g. the Internet) is considered a difficult problem. Automatic classification of queries is not well investigated.

The CSI requires that there exists a hierarchical classification system that is possible to use both in manual classification (it must make sense as a taxonomy of human knowledge) and automatic classification. Ideally, the entire result set expected from a query should be located in the same zone of the network, and the query should be directed to this zone.

The chosen classification system is the ODP Web Directory [11], which consists of approx. 380 000 topics.

We have employed an automatic classification of the TREC WT10g document collection [12] in order to study the *recall*[2] of a retrieval system based on the CSI design.

The WT10g collection consists of 1.7 million documents fetched from the World Wide Web, 100 topics (information queries expressed in normal language) and relevance judgements which list the documents regarded as the result sets of the different topics[3]. It is thus possible to test our choice of classification system and the classification algorithms by checking how many of the relevant documents are classified as the same topic or sub-topic as the query.

The classification experiments on the WT10g collection and TREC topics have used a straightforward automatic classifier using well-known methods of lexical analysis. The results show recall values in the same range as observed with other simple, baseline methods: 20-40% of the metadata can be recalled if items of metadata are replicated in a small number (<10) of members with different topic values. By doing this, the same item of metadata will exist in different places in the network, improving the chances that a query will find it. This increases the network traffic with a constant factor only, and does not affect our complexity analysis of the CSI’s scaleability. These numbers are acceptable recall values for a resource discovery system in the general domain.

Related work

The CSI is an application in the field of Information Retrieval. Several papers suggest various design alternatives for a distributed search engine. The best known of these projects are:

- Harvest [14] - A distributed retrieval system based on the separation of *gatherers* (responsible for collecting information) and *brokers* (responsible for index generation and information dissemination).
- Whois++ [15] - A distributed information system where the servers inform each other about their content by exchanging *centroids*. Centroids enables the servers to offer *query routing*.
- MIDS [6] - A project that builds on Harvest, but provides a more detailed architectural framework for distributed retrieval services. Also based on the use of Centroids.

The traffic volume generated by the Centroids and their effect on the distribution of the query volume is not analysed in these projects. Our own evaluation framework for this type of design [8] indicates that the volume of

[1] This zone is equivalent to a branch of the topic hierarchy

[2] The recall of a retrieval operation is the volume of relevant data in a result set, divided by the volume of relevant data in the entire system [13 p.5]

[3] More topics and relevance judgements do exist, but do not apply to this retrieval experiment

Centroids grows in the same manner as the information volume, i.e. exponentially.

The CSI design has a proven performance in terms of scaleability, but assumes that it is possible to classify queries and metadata without too much loss of information during retrieval operations. *Manual classification* can be done for all kinds of resources (e.g. services and multimedia) if necessary. Textual resources can also be *automatically classified* through inspection of their information content by a *classifier.*

Remaining problems for future research

The tight coupling between classification and distribution is a novel approach to distributed information retrieval, and several questions need to be addressed:

- What will be the observed response time when querying the CSI using actual data?
- How can ranking techniques (e.g. relevance feedback) improve the usefulness of the result set?
- How will the actual distribution of topic values affect the scaleability analysis?
- Will other classification models and more advanced classification techniques improve the recall of the result set?
- How will member crashes affect the recall of the result set (parts of the metadata storage becomes unavailable)?

To address some of these problems, a large distribution experiment has recently been conducted, where the CSI design has been evaluated in a real networked environment consisting of 130 computers.

Conclusion

Using classification as a basis for distribution in a large-scale resource discovery system appears to give a scaleability not seen in earlier research projects. The CSI also shows a robust operation and a graceful degradation when networking errors occur more frequently. Experiments with the classification process and a reference collection indicates that it is possible to obtain acceptable quality of retrieval operations using this approach.

References

1. Kobayashi, M. and Takeda, K., Information Retrieval on the Web, *ACM Computing Surveys*, 32 (2), June 2000, pp. 144-173
2. Berners-Lee, T., J. Hendler, and O. Lassila, The Semantic Web, *Scientific American*. May 2001.
3. Grey, D.J., P. Dunne, and R.I. Ferguson, WebSeeker: a means of efficiently locating resources on the World Wide Web using mobile, collaborative agents. *Workshop2000 on agent-based simulation*. Passau, Germany, 2000, pp. 57-62
4. Kosmynin, A., From Bookmark Managers to Distributed Indexing: An Evolutionary Way to the Next Generation of Search Engines. *IEEE Communications Magazine*, 35 (6), 1997, pp. 146-151.
5. Sherman, C., What's new with Web Search, *Online*, 24 (3), 2000, pp. 27-34.
6. Helm, D.J., R.J. D'Amore, and P.-F. Yan, MIDS: a framework for information organization and discovery. *Journal of Network and Computer Applications*, 19 (4), 1996, pp. 381-394.
7. Weider, C., J. Fullton, and S. Spero, Architecture of the Whois++ Index Service, *RFC1913*. 1996, IETF.
8. Fongen, A., et al., Distributed Resource Discovery Using a Content-Sensitive Infrastructure, *International Conference on Information Integration and Web-based Applications & Services*, Linz, Austria, 2001, pp. 205-215.
9. Ratnasamy, S., et al. A Scalable Content-Adressable Network, *SIGCOMM 2001*. San Diego, California, 2001, pp. 161-172.
10. Stocia, I., et al. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications, *SIGCOMM 2001*. 2001. San Diego, California.pp. 149-160.
11. ODP, The Open Directory Project *[online]*, http://www.dmoz.org/.
12. Hawking, D., et al., Results and Challenges in Web Search Evaluation, *Computer Network*, 31,1999, pp.1321-1330.
13. Kowalski, G.J. and M.T. Maybury, *Information Storage and Retrieval Systems, theory and implementation. 2nd ed.* 2 ed. 2000: Kluwer Academic Publisher.
14. Bowman, M., et al., The Harvest Information Discovery and Access System, *Computer Networks and ISDN Systems*, 28, 1995: p. 119-125.
15. Wang, B., et al., Standards in the CHIC-Pilot distributed indexing architecture, *Computer Networks and ISDN Systems*, 30 (16), 1998, pp.1571-1578.