Information Processing and Management 49 (2013) 51–66

Contents lists available at SciVerse ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Document replication strategies for geographically distributed web search engines

Enver Kayaaslan [a,1], B. Barla Cambazoglu [b,*], Cevdet Aykanat [a]

[a] Computer Engineering Department, Bilkent University, Ankara, Turkey
[b] Yahoo! Research, Barcelona, Spain

ARTICLE INFO

Article history:
Received 16 September 2010
Received in revised form 7 September 2011
Accepted 2 January 2012
Available online 2 February 2012

Keywords:
Web search
Distributed information retrieval
Document replication
Query processing
Query forwarding
Result caching

ABSTRACT

Large-scale web search engines are composed of multiple data centers that are geographically distant to each other. Typically, a user query is processed in a data center that is geographically close to the origin of the query, over a replica of the entire web index. Compared to a centralized, single-center search engine, this architecture offers lower query response times as the network latencies between the users and data centers are reduced. However, it does not scale well with increasing index sizes and query traffic volumes because queries are evaluated on the entire web index, which has to be replicated and maintained in all data centers. As a remedy to this scalability problem, we propose a document replication framework in which documents are selectively replicated on data centers based on regional user interests. Within this framework, we propose three different document replication strategies, each optimizing a different objective: reducing the potential search quality loss, the average query response time, or the total query workload of the search system. For all three strategies, we consider two alternative types of capacity constraints on index sizes of data centers. Moreover, we investigate the performance impact of query forwarding and result caching. We evaluate our strategies via detailed simulations, using a large query log and a document collection obtained from the Yahoo! web search engine.

1. Introduction

We consider a large-scale web search engine architecture with multiple, geographically distributed data centers (Baeza-Yates, Gionis, Junqueira, Plachouras, & Telloli, 2009; Cambazoglu, Plachouras, & Baeza-Yates, 2009). In this architecture, each data center crawls and maintains the documents that are served by the web sites in its geographical region (Cambazoglu, Plachouras, Junqueira, & Telloli, 2008). User queries are routed to data centers according to the regions they originate from. For example, a data center in Madrid crawls the web sites in Spain and processes the queries submitted from Spain. As we will discuss next, this architecture leads to two extremes for the placement of the web index and query processing.

At one extreme, a global index is built over the entire web collection, and this index is replicated on all data centers. Queries are processed on the entire web index, and hence search result qualities are identical to those of a centralized search architecture. However, this approach does not scale well since the global web index needs to be constructed from a distributed document collection and periodically maintained. Moreover, this approach requires major hardware investments and results in high power consumption, which is an important issue for commercial search engines. Finally, processing queries

* Corresponding author. Address: Yahoo! Research, Avda. Diagonal 177, 8th Floor, 08018 Barcelona, Spain. Tel.: +34 93 183 8830; fax: +34 93 183 8901.
E-mail addresses: enver@cs.bilkent.edu.tr (E. Kayaaslan), barla@yahoo-inc.com (B.B. Cambazoglu), aykanat@cs.bilkent.edu.tr (C. Aykanat).
[1] This work is conducted during the author's internship at Yahoo! Research Barcelona.

0306-4573/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2012.01.002

over an entire web index may be too costly to satisfy the tight response time constraints of large-scale web search engines (Cambazoglu, Zaragoza, et al., 2010).

At the other extreme, each data center builds a regional web index on its local crawl and processes its queries over this partial (local) index. This approach is highly scalable because partial indexes are locally maintained and less resources are needed for query processing (alternatively, queries can be processed faster). However, as processing of queries is limited to a partial index, some high-quality or best matching documents that are indexed by non-local data centers may be missing in search results. This may lead to not affordable losses in search result qualities, negatively impacting the user satisfaction and potentially the revenues of the search engine.

A search engine architecture based on selective replication of documents on data centers emerges as a feasible mid-ground between these two extremes. The main idea in selective replication is to identify the documents that are of interest to the users of each geographical region and replicate the documents on the data centers according to the user interest. If this can be wisely done, queries can be locally processed in regional data centers, reducing the search quality loss relative to the second extreme and providing better scalability compared to the first extreme. Selective replication can be further coupled with selective forwarding of queries between data centers so that documents that are missing in the local top k results (with respect to the global top k results) can be retrieved from non-local data centers, preventing any search quality loss (Baeza-Yates et al., 2009; Cambazoglu, Varol, Kayaaslan, Aykanat, & Baeza-Yates, 2010).

In this paper, we propose strategies for selectively replicating documents in a geographically distributed search engine setting. Our strategies identify the documents that are of interest to the users of certain geographical regions, based on the occurrence frequencies of documents in past search results. The identified documents are then replicated and indexed on non-local data centers so that future queries can be efficiently and effectively processed.

The outline of the paper is as follows. In Section 2, we provide the details of the search engine architecture that we consider in this work and provide formal definitions for two variants of the document replication problem we aim to solve. Section 3 describes the datasets used in our work and the setup of our simulations. In Sections 4–6, we propose various replication algorithms, each optimizing different performance metrics under different constraints and assumptions. We report the experimental results about the performance of our algorithms in the associated sections. In Section 7, we investigate the impact of query forwarding on the performance. We survey the related work in Section 8. The paper is concluded in Section 9.

2. Preliminaries

2.1. Architecture

We consider a search engine architecture composed of multiple data centers. In this architecture, each data center crawls and stores documents belonging to a disjoint subset of the Web. Each data center then builds a local web index over its crawled documents, independent of the other data centers. We assume that IP addresses (or countries at a higher granularity) are statically assigned to data centers according to their geographical proximity. Each data center is responsible for processing queries that originate from its subset of IPs and is said to be the local data center for those queries.

In our architecture, certain documents are replicated on non-local data centers. Hence, in addition to its local index, each data center maintains a replicated index, built over its non-local documents. The replication pattern of documents is periodically determined based on the frequencies with which documents appear in the search results generated by individual data centers.

Queries are evaluated as follows.[2] A user query is first processed in the local data center associated with the user, over both local and replicated indexes. A local top k result set is formed based on the estimated relevance scores of documents (Cambazoglu & Aykanat, 2006). At this point, this result set may be immediately returned to the user. Alternatively, the query may be forwarded to a set of non-local data centers, hoping to retrieve some documents whose scores are higher than that of the lowest scoring document in the locally computed top k set. Forwarded queries are concurrently processed over the local indexes of non-local data centers, whose top k results are returned to the local data center. These results are then merged in decreasing order of scores and the top k results are returned to the user.

Fig. 1 illustrates the process. In the figure, each data center is represented by a large box. The patterns indicate the original assignment of documents to data centers. The box in the top row represents the local documents of a data center. The boxes in the bottom row represent non-local documents that are locally replicated. The directed arcs show contributions of different document collections to the final search results.

In this architecture, if a query is only locally processed, the search result quality may deteriorate as some of the documents that appear in the global top k result set may not be available in the local data center. On the other hand, if the query is forwarded to non-local data centers, the query response time increases due to the network latency between the local and non-local data centers (the workload may also increase). Readers may refer to (Cambazoglu, Varol, et al., 2010) for more details about the architecture.

[2] We assume that the global collection statistics are made available to all data centers so that the scores generated by different data centers are comparable.

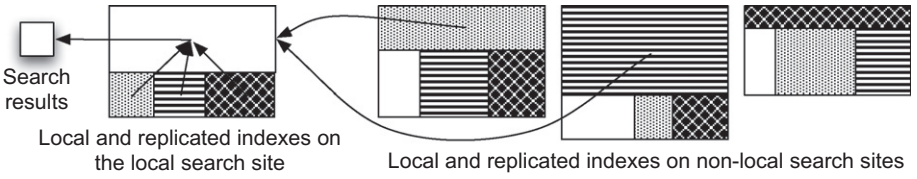

Fig. 1. Replication and query processing on the proposed web search architecture.

We note that document replication and query forwarding are not competing strategies, but rather query forwarding is a technique that is complementary to partial document replication. There are query forwarding algorithms (Baeza-Yates et al., 2009; Cambazoglu, Varol, et al., 2010) that can obtain the result quality of the global ranking by forwarding queries to a subset of non-local data centers instead of all.[3] Query forwarding algorithms are beyond the scope of this paper. In this work, we assume either no query forwarding or an oracle query forwarding algorithm that correctly predicts the non-local data centers that will contribute to the global top k set and forwards queries to only those data centers.

2.2. Document replication framework

Our focus in this work is specifically on document replication, which may lead to efficiency and effectiveness improvements in the above-mentioned search architecture. In our replication framework, the documents that are frequently requested by the queries originating from a particular region are replicated on the data center responsible for that region. This approach improves the search quality attained by the local data centers, assuming a scenario where queries are not forwarded between data centers. This is because the overlap between the local and global top k sets is likely to increase as more non-local documents are replicated on the local data centers. The replication of documents also leads to improvements in the query processing efficiency, assuming a scenario where queries are forwarded between the data centers. This is because fewer queries need to be forwarded and hence savings can be achieved in average query response times and the query workload of the search engine. The former benefit is because the network latency overhead incurred by query forwarding is eliminated for some queries. The latter benefit is since queries are processed over a smaller portion of the index.

Obviously, increasing the replication amount has a negative impact on query processing times of local data centers due to the increase in their index sizes. One of the main goals of this paper is to observe the trade-off between replication and search performance to identify the best replication strategies. In Sections 4–6, we investigate the impact of replication on the search quality, average response time, and query workload, respectively.

In our document replication framework, we assume that the amount of compute resources made available to the search engine remains the same after replicating documents.[4] We also assume that the resources are distributed among the data centers in proportion to their index sizes, as an attempt to preserve the relative query processing performances of data centers. These assumptions are necessary to draw sound conclusions about the query processing efficiency under document replication since query processing times depend on both index sizes and computing powers of data centers.

We constrain the allowed replication amount in two different ways. In the first approach, a global capacity constraint bounds the total size of the index obtained after document replication, over the entire search system. This approach requires redistribution of the available hardware, after replication, among the data centers to preserve the ratio between the index size and the amount of compute resources of each data center. In the second approach, a local capacity constraint bounds the replicated index size of each data center relative to its local index size. This approach does not require redistribution of the hardware because the ratio between the local and replicated index sizes is the same for all data centers.

In Fig. 2, we illustrate these two types of constraints by an example. Fig. 2a shows a distributed search engine with four data centers, each initially having a local index of varying size and a fixed amount of hardware. Figs. 2b and 2c show the indexes created after replication. In Fig. 2b, the total index size in the system doubles after replication. In Fig. 2c, the index size of each data center individually doubles. We note that the distribution of compute resources among the data centers changes in Fig. 2b, whereas it remains the same in Fig. 2c.

2.3. Formal problem definition

We are given a set $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$ of N documents, a set $\mathcal{Q} = \{q_1, q_2, \ldots, q_M\}$ of M queries, and a set $\mathcal{C} = \{C_1, C_2, \ldots, C_K\}$ of K data centers. Every data center $C_\ell \in \mathcal{C}$ initially stores a disjoint set $\mathcal{D}_\ell \subseteq \mathcal{D}$ of documents and serves to a disjoint set $\mathcal{Q}_\ell \subseteq \mathcal{Q}$ of queries. Moreover, each document $d_j \in \mathcal{D}$ is associated with a space overhead s_j (e.g., the number of postings the document contributes to the index) and the frequency f_j with which the document appears in the relevant results of queries in $\mathcal{Q}$.

[3] However, these algorithms cannot restrict the set of contacted non-local sites to only those that are guaranteed to contribute to the global top k result set.

[4] If additional hardware is used in order to compensate the increase in the index size due to replication, the financial implications of this should be taken into account.

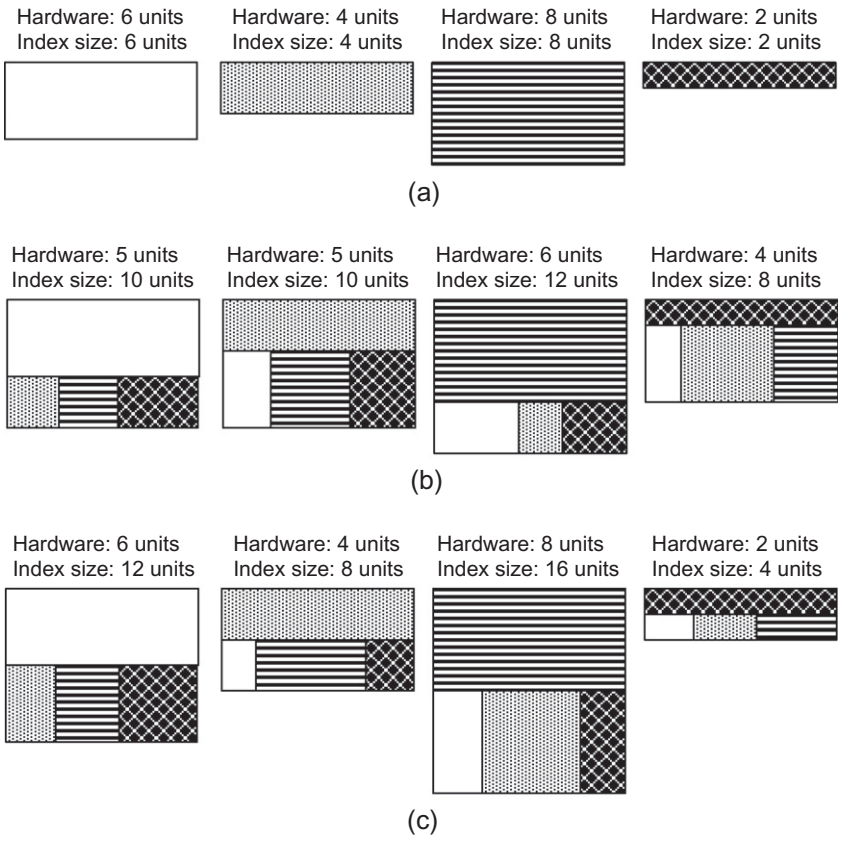

Fig. 2. A search engine with four data centers. Indexes: (a) before replication, (b) after replication under a constraint on the total index size (doubles the total index size), (c) after replication under a constraint on individual index sizes of data centers (doubles the index size of each data center).

Definition 1 (*Document replication*). A document replication Φ is a mapping from a document $d_j \in \mathcal{D}_\ell$ to a set of data centers in $\mathcal{C} - \{C_\ell\}$, i.e., $\Phi(d_j)$ denotes the set of non-local data centers on which document d_j is replicated.

For a given document replication Φ, let $\mathcal{D}_\ell^\Phi$ denote the set of documents that are either initially stored or later replicated on C_ℓ, i.e.,

$$\mathcal{D}_\ell^\Phi = \mathcal{D}_\ell \cup \{d_j \in \mathcal{D} : C_\ell \in \Phi(d_j)\}. \tag{1}$$

Let ϕ_j be the number of non-local data centers on which d_j is replicated, i.e., $\phi_j = |\Phi(d_j)|$. Given this notation, we define the following two document replication problems. The first problem assumes a global capacity constraint on the replication amount while the second assumes that each data center has a local capacity constraint.

Problem 1 (*Replication under global capacity constraint (DR-G)*). Given sets $\mathcal{C}$, $\mathcal{D}$, $\mathcal{Q}$, and a global replication capacity $G \geqslant 0$, find a document replication Φ such that the total replication amount does not exceed G, i.e.,

$$\sum_{d_j \in \mathcal{D}} s_j \phi_j \leqslant G, \tag{2}$$

while a given performance objective is optimized.

Problem 2 (*Replication under local capacity constraints (DR-L)*). Given sets $\mathcal{C}, \mathcal{D}, \mathcal{Q}$, and a local replication capacity $L_\ell \geqslant 0$ for each data center $C_\ell \in \mathcal{C}$, find a document replication Φ such that the replication amount on each $C_\ell \in \mathcal{C}$ does not exceed its capacity L_ℓ, i.e.,

$$\sum_{d_j \in \mathcal{D}_\ell^\Phi - \mathcal{D}_\ell} s_j \leqslant L_\ell, \quad \forall C_\ell \in \mathcal{C}, \tag{3}$$

while a given performance objective is optimized.

2.4. Baseline solutions

As a simple baseline solution for the DR-G problem, we use a modified version of the document replication heuristic used in (Cambazoglu, Varol, et al., 2010). In this heuristic, each document d_j is assigned a profit value p_j, estimated by the ratio between the past access frequency and the space overhead of d_j, i.e., $p_j = f_j/s_j$. The heuristic then iterates over all documents in decreasing order of their profits. At each iteration, a document d_j is inserted into an initially empty set $\mathcal{G}$ if the global capacity check $\sum_{d_j \in \mathcal{G}} s_j(K-1) \leqslant G$ does not fail. After the algorithm iterates on all documents, the documents in $\mathcal{G}$ are replicated on all non-local data centers. We note that this heuristic slightly improves over a heuristic that terminates when the capacity check first fails (Cambazoglu, Varol, et al., 2010). The complexity of this algorithm is $O(N\lg N + NK)$, where the former term is the cost of sorting the documents and the latter is the cost of decoding the document replication solution.

For the DR-L problem, we use a baseline similar to the above-mentioned baseline. The sorted list of documents is traversed once, performing for each document K separate checks on local capacities of data centers. Given a data center C_ℓ, only the documents that are non-local for C_ℓ (i.e., those in $\mathcal{D} - \mathcal{D}_\ell$) are considered for replication on C_ℓ. Documents are inserted into a separate set $\mathcal{L}_\ell$ if the local capacity check $\sum_{d_j \in \mathcal{L}_\ell} s_j \leqslant L_\ell$ does not fail for data center C_ℓ. Once the traversal terminates, documents in each set $\mathcal{L}_\ell$ are replicated on the respective data center C_ℓ. The complexity of this algorithm is $O(N\lg N + NK)$, similar to the DR-G problem.

3. Setup

For simulations, we create two different setups, referred to as `Europe` and `World`. Both setups simulate a geographically distributed search engine with five data centers. In `Europe`, data centers are located in Germany, Spain, France, Italy, and UK. In `World`, they are located in Australia, Brazil, Canada, Germany, and Mexico. The former simulates an architecture with low network latencies between data centers while the data centers in the latter setup have high network latencies. We assume that each data center is located in the capital city of the respective country. We also assume that queries are issued from the five most populated cities in each country.

We predict the network latencies between the data centers (also, between the data centers and their users), using the speed of light on copper wire (200,000 km/s) and the great-circle distance between data center locations. We then project the predicted latencies to more accurate values by a formula obtained through regression between predicted and real-life latency measurements (Cambazoglu, Varol, et al., 2010). The real-life latency values are obtained over several geographically distant computers available to us.

The document collection contains about 200 million web pages crawled from the Web in 2009. This is a high-quality collection obtained after various cleansing (e.g., spam filtering). Using a proprietary classifier, we determine an initial local country assignment for every document. We limit our study to only the documents that are assigned to one of the selected countries. We built separate local and replicated indexes, using Terrier (Ounis, Amati, Plachouras, He, Macdonald, & Johnson, 2005).

For each data center, we extract samples with about 8.5 and 7 million queries from the query logs of Yahoo!, for the `Europe` and `World` setups, respectively. We preprocess queries in four steps: query terms are case-folded, stop-words are eliminated, duplicate terms are uniqued, and query terms are alphabetically sorted.[5] Finally, the queries are sorted in increasing order of arrival timestamps and split into four pieces. The first three pieces form the training set. The final piece forms the test set.

For both the training and test sets, we assume that the top k result sets obtained over the entire index form the relevant documents. We evaluate the queries using a modified version of Terrier.[6] The top k result sets of training queries are used as inputs to the replication algorithms given in later sections. The top k result sets of test queries are used for evaluation.

According to Fig. 3, the occurrence frequencies of the documents in the top k search results follow a power-law distribution for both setups. In the `Europe` setup, the fraction of documents that appear in the top k result set of at least one query is 0.53 for the training query set and 0.32 for the test query set. For the `World` setup, the fractions are 0.61 and 0.40, respectively.

In our simulations, we assume a setting where each node in a search cluster builds an index on three million documents. We estimate the total number of processors available to the entire search system based on the same assumption. We assign each data center a number of processors proportional to its index size. Hence, query processing times are comparable for data centers. During our simulations, we assume that the indexes are maintained in the main memories of the search nodes. The simulator assumes that the query processing cost is linearly proportional with the total number of postings associated with the query terms. We set the time cost of processing a single posting to 200 ns, which is an empirical value obtained from Terrier. We also assume a 20 ms preprocessing overhead per query. We omit all other costs as they are relatively less important, especially for low k and K values.

Regarding caching of previous search results, we conduct our experiments under two different scenarios: no result cache or an infinite result cache (Cambazoglu, Junqueira, et al., 2010). In the latter scenario, we assume that each data center

[5] We use only the queries requesting the first result page.

[6] We modified Terrier to support the true AND logic in document matching.

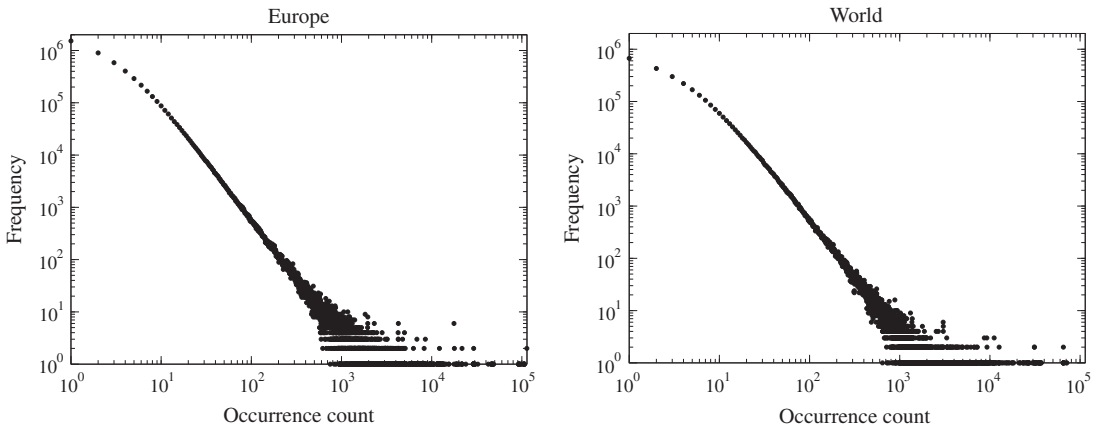

Fig. 3. Number of documents that appear in search results with a given occurrence count (both axes are in log scale).

maintains a result cache that stores the results of all queries that are previously issued from its region. In either scenario, the results are reported over the entire (both hit and miss) set of test queries. The result cache hit ratios are 0.56 (0.54) for the training query set and 0.51 (0.47) for the test query set in case of the `Europe` (`World`) setup.

In all plots, we use the tags `DR-G` and `DR-L` to denote the proposed replication algorithms, which use a global capacity constraint or local capacity constraints, respectively. The baselines given in Section 2.4 are similarly named as `B-G` and `B-L`. The default case with no replication is denoted by `NR`. We denote the scenario where the entire index is replicated on all data centers by `FR`. The replication amounts reported in the plots represent the percent replication on the entire system with respect to the replication amount in the `FR` scenario, i.e., $(K-1)\sum_{d_j \in \mathcal{D}} s_j$. In case of `DR-G`, 100% replication corresponds to the `FR` scenario. We note that, in case of `DR-L`, the final replication amount on a data center may be lower than the local replication capacity if the index becomes fully replicated on that data center before the local replication capacity is reached. A single simulation run takes under an hour with the parameters in our experiments.

4. Optimizing search quality

4.1. Objective

Herein, we focus on replication algorithms for a specific search scenario where queries are processed only in their local data centers without any forwarding to non-local data centers. In this scenario, search result qualities may degrade as queries are processed over a subset of the entire index. The main idea behind our replication algorithms is to replicate, in a particular data center, the non-local documents that are frequently accessed by the users of that data center. As we replicate more documents, the search quality achieved by processing queries only within the local data centers is expected to be closer to that of a centralized search architecture.

We denote by $\mathcal{R}_i$ the set of relevant documents obtained by evaluating query q_i on an index built over the entire document collection $\mathcal{D}$. Also, we denote by $\widehat{C}_i$ the data center that serves query q_i and by $\widehat{\mathcal{D}}_i^{\Phi}$ the document collection on $\widehat{C}_i$ after documents are replicated via some Φ. We measure the result quality of a query by its precision, defined as follows.

Definition 2 (*Precision of a query*). For a given document replication Φ, the precision $\rho(q_i, \Phi)$ of a query q_i is defined as the fraction of the relevant documents on $\widehat{C}_i$ to all relevant documents, i.e.,

$$\rho(q_i, \Phi) = \left|\widehat{\mathcal{D}}_i^{\Phi} \cap \mathcal{R}_i\right| / |\mathcal{R}_i|. \tag{4}$$

The total precision $\mathrm{P}(\mathcal{Q}, \Phi)$ of a document replication Φ is defined as a sum over the precisions of individual queries in $\mathcal{Q}$, i.e.,

$$\mathrm{P}(\mathcal{Q}, \Phi) = \sum_{q_i \in \mathcal{Q}} \rho(q_i, \Phi). \tag{5}$$

Given this definition, the problem is to find a feasible document replication Φ that maximizes $\mathrm{P}(\mathcal{Q}, \Phi)$ as the objective of the DR-G and DR-L problems.

4.2. Solution

Our solution to the DR-G problem is based on a combinatorial reduction to the well-known 0–1 knapsack problem (Cormen, Leiserson, Rivest, & Stein, 2009), where we are given a capacity W and a set $\mathcal{T}$ of n items, each associated with

a positive weight and a positive value. The goal is to find a subset $\mathcal{T}^* \subseteq \mathcal{T}$ such that the total weight of items in $\mathcal{T}^*$ does not exceed W and their total value is maximized. For every pair of document $d_j \in \mathcal{D}$ and data center $C_\ell \in \mathcal{C}$ such that $d_j \notin \mathcal{D}_\ell$, we introduce an item $t_{j\ell}$ into the item set $\mathcal{T}$ with an associated weight $w_{j\ell} = s_j$ and value $v_{j\ell} = \sum_{q_i \in \mathcal{Q}_\ell, d_j \in \mathcal{R}_i} (1/|\mathcal{R}_i|)$. The knapsack capacity is set to the global replication capacity G. Based on the solution set $\mathcal{T}^*$ of the knapsack problem, we form the set $\Phi(d_j)$ of data centers where document d_j will be replicated as

$$\Phi(d_j) = \{C_\ell : t_{j\ell} \in \mathcal{T}^*\}. \tag{6}$$

Here, each item $t_{j\ell} \in \mathcal{T}^*$ represents the replication of document d_j on data center C_ℓ. This replication increases the total precision by $1/|\mathcal{R}_i|$ for each query $q_i \in \mathcal{Q}_\ell$ such that $d_j \in \mathcal{R}_i$, i.e., by $v_{j\ell}$. Replicating d_j on C_ℓ consumes a space of s_j from the available space, bounded by the global capacity G, i.e., we pick an item of weight $w_{j\ell}$ without exceeding the knapsack capacity W. Hence, the proposed reduction correctly maximizes $\mathrm{P}(\mathcal{Q}, \Phi)$.

For the DR-L problem, we use a slight variation of the above solution. Since each data center has its own local replication capacity, we solve a different knapsack problem instance for each data center C_ℓ as follows. For each document $d_j \in \mathcal{D} - \mathcal{D}_\ell$, we introduce an item $t_{j\ell}$ into the item set $\mathcal{T}_\ell$ with the same weight and value used in the previous formulation. The knapsack capacity is set to the local replication capacity L_ℓ of data center C_ℓ. We then solve the knapsack problem instance associated with each data center C_ℓ and obtain a solution set $\mathcal{T}^*_\ell$. After we obtain all K solution sets, we form a document replication Φ as

$$\Phi(d_j) = \{C_\ell : t_{j\ell} \in \mathcal{T}^*_\ell\}. \tag{7}$$

This reduction correctly maximizes $\mathrm{P}(\mathcal{Q}, \Phi)$ since replicating d_j on C_ℓ consumes a space of s_j from C_ℓ's available space, bounded by the local capacity L_ℓ, i.e., we pick an item of weight $w_{j\ell}$ without exceeding the knapsack capacity W_ℓ in the knapsack problem instance associated with C_ℓ.

Since the 0–1 knapsack problem is NP-hard, we use a greedy approximation algorithm (Cormen et al., 2009) in our solution. The algorithm has a complexity of $O(n\lg n)$, where n refers to the number of items in the knapsack instance. In the algorithm, each item is assigned a profit, set to the ratio between the item's value and weight. The heuristic iterates over all items in decreasing order of profits. At each iteration, an item is placed in the solution set if its placement does not violate the capacity constraint.

Our solutions to the DR-G and DR-L problems have time complexities $O(MK + NK\lg(NK))$ and $O(MK + NK\lg N)$, respectively. In these complexities, the first term is the total cost of encoding the document replication instance as knapsack instance(s). The second term is the cost of the greedy approximation algorithm as we solve one knapsack instance with $O(NK)$ items in the DR-G problem and K knapsack instances each with $O(N)$ items in the DR-L problem. Since the cost of decoding the knapsack solution(s) as a document replication is $O(NK)$, it is not shown in the complexities.

4.3. Performance evaluation

Fig. 4 shows the average precision values observed as the replication amount increases. The average precision over a test query log $\mathcal{Q}'$ is computed as $\mathrm{P}(\mathcal{Q}', \Phi)/|\mathcal{Q}'|$. For a better visibility of the curves, the average precision values for `NR` (0.61 and 0.55, for `Europe` and `World`, respectively) are not displayed in the figure. Relative to these values, even with only 1% replication, considerable improvement in precision is observed (up to about 0.26 precision increase in either setup). When 16% of the index is replicated, the loss in the average precision is at most 10% with respect to the `FR` scenario, which naturally achieves a precision of one. This behavior is mainly due to the power-law distribution in the occurrence frequencies of documents in search results (previously shown in Fig. 3).

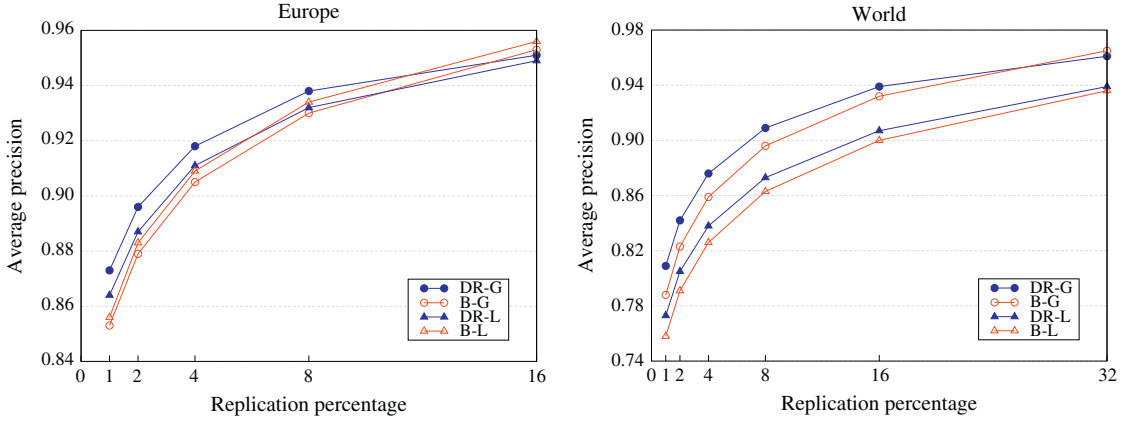

Fig. 4. Average precision values as the replication amount increases.

According to Fig. 4, both DR-G and DR-L achieve slightly better precision values than their respective baselines. As expected, the improvements over the baselines are less pronounced at high replication amounts. In general, DR-G has a relatively better performance than DR-L since DR-G has a larger solution space due to the flexibility of reassigning the hardware.

The precision values in the World setup are lower than those in the Europe setup since the search results in the former setup contain relatively more unique documents. Hence, the World setup requires larger amounts of replication to achieve the same performance with the Europe setup. For example, achieving a precision of 0.92 requires more than 8% replication in World, whereas it requires only 4% replication in Europe (assuming DR-G).

5. Optimizing response time

5.1. Objective

In this section, we consider a search architecture where queries are forwarded between the data centers to retrieve all relevant documents so that there is no loss in the search quality, i.e., the precision is always one. We assume an oracle algorithm that forwards the queries to only the non-local data centers that contain relevant documents. As the performance objective in replicating documents, we try to reduce the average query response time.

In this scenario, the average query response time is determined by the time needed to compute the query results on different indexes and the network latencies between the data centers. The main idea in the proposed replication algorithm is to maximize the number of queries that can be entirely processed by the local data centers without any forwarding, thus eliminating the network latency and the overhead of query processing on non-local data centers. If all relevant results of a query are found in the local data center, the query is dubbed local. The locality of a query is defined as follows.

Definition 3 (*Locality of a query*). For a given document replication Φ, the locality $\gamma(q_i, \Phi)$ of a query q_i is defined as

$$\gamma(q_i, \Phi) = \begin{cases} 1, & \text{if } \mathcal{R}_i \subseteq \widehat{\mathcal{D}}_i^{\Phi} \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

The total locality $\Gamma(\mathcal{Q}, \Phi)$ of a document replication Φ is defined as a sum over the locality values of queries in $\mathcal{Q}$, after the documents are replicated via Φ, i.e.,

$$\Gamma(\mathcal{Q}, \Phi) = \sum_{q_i \in \mathcal{Q}} \gamma(q_i, \Phi). \tag{9}$$

The goal is now to find a feasible document replication Φ that maximizes $\Gamma(\mathcal{Q}, \Phi)$ as the objective of the DR-G and DR-L problems.

5.2. Solution

We formulate the problem as a variation[7] of the set union knapsack problem (Kellerer, Pferschy, & Pisinger, 2004). In our problem, we are given a main set $\mathcal{T}$ of n items, where each item is associated with a positive weight. We are also given a family $\mathcal{F} = \{S_1, S_2, \ldots, S_m\}$ of m sets, where each set S_i is a subset of the main set, i.e., $S_i \subseteq \mathcal{T}$. The objective of the problem is to find a subset $\mathcal{T}^*$ of the main set $\mathcal{T}$ such that $|\{S_i \in \mathcal{F} : S_i \subseteq \mathcal{T}^*\}|$ is maximized while the total weight of the items in $\mathcal{T}^*$ does not exceed a given capacity W.

Since our problem is NP-hard, we employ a heuristic solution (Ntoulas & Cho, 2007). This heuristic follows the maximum benefit per unit cost policy (Cormen et al., 2009) and has no guarantee of optimality. In this heuristic, each item in $\mathcal{T}$ is assigned a profit value which is set to the ratio between the number of sets that contain the item and the item's weight. The heuristic then iterates over all items in decreasing order of their profits. At each iteration, an item is placed in $\mathcal{T}^*$ if the capacity constraint is not violated by the placement of the item. The heuristic has a complexity of $O(m + v + n\lg n)$, where v denotes the sum of the sizes of the sets in $\mathcal{F}$, i.e., $v = \sum_{S_i \in \mathcal{F}} |S_i|$. The first two complexity terms refer to the cost of computing the profit values and the last term refers to the cost of sorting the items.

The solution we propose for the DR-G problem is based on a combinatorial reduction to our variation of the set union knapsack problem. Let us consider every data center $C_\ell \in \mathcal{C}$. For each document $d_j \in \mathcal{D} - \mathcal{D}_\ell$, we introduce an item $t_{j\ell}$ into the main set $\mathcal{T}$ with an associated weight $w_{j\ell} = s_j$. Moreover, for each query $q_i \in \mathcal{Q}_\ell$, we introduce a set $S_i = \{t_{j\ell} : d_j \in \mathcal{R}_i - \mathcal{D}_\ell\}$ into family $\mathcal{F}$. The capacity W is set to the global replication capacity G. After obtaining solution $\mathcal{T}^*$, we form the document replication Φ as

$$\Phi(d_j) = \{C_\ell : t_{j\ell} \in \mathcal{T}^*\}. \tag{10}$$

Here, each item $t_{j\ell} \in \mathcal{T}^*$ represents the replication of document d_j on data center C_ℓ. Thus, each set $S_i \subseteq \mathcal{T}^*$ implies $\mathcal{R}_i \subseteq \widehat{D}_i^{\Phi}$ for the corresponding query $q_i \in \mathcal{Q}$. Due to (8) and (9), the objective of the reduced problem corresponds to maximizing the total

[7] The same formulation is adopted by (Ntoulas & Cho, 2007) for static index pruning.

locality $\Gamma(\mathcal{Q}, \Phi)$. Replicating d_j on data center C_ℓ consumes a space of s_j from the available space, bounded by the global capacity G, i.e., we pick an item of weight $w_{j\ell}$ without exceeding the capacity W. Hence, the proposed formulation is correct.

For the DR-L problem, we use a minor variant of the above solution. Since each data center has its own local replication capacity, we solve a separate problem instance for each data center $C_\ell \in \mathcal{C}$. For each document $d_j \in \mathcal{D} - \mathcal{D}_\ell$, we introduce an item $t_{j\ell}$ into the main set $\mathcal{T}_\ell$ with the same weight as before. For each query $q_i \in \mathcal{Q}_\ell$, we introduce a set $S_i^\ell = \{t_{j\ell} : d_j \in \mathcal{R}_i - \mathcal{D}_\ell\}$ into family $\mathcal{F}_\ell$. The capacity W_ℓ is set to the local replication capacity L_ℓ. We then solve the problem instance associated with each data center $\mathcal{T}_\ell$ and obtain a solution set $\mathcal{T}_\ell^*$. After we obtain all K solution sets, we form a document replication Φ as

$$\Phi(d_j) = \{C_\ell : t_{j\ell} \in \mathcal{T}_\ell^*\}. \tag{11}$$

We omit a discussion on the correctness of this solution as it is very similar to the above discussion on the correctness of our solution for the DR-G problem.

For both the DR-G and DR-L problems, the cost of encoding the document replication instance is $O(NK + Mk)$. For the DR-G problem, the complexity of the employed heuristic is $O(Mk + NK\lg(NK))$ while, for the DR-L problem, the complexity of solving the K problem instances is $O(Mk + NK\lg N)$. In both problems, the decoding of the obtained solution(s) has $O(NK)$-time complexity. Hence, our solutions to the DR-G and DR-L problems have overall time complexities of $O(Mk + NK\lg(NK))$ and $O(Mk + NK\lg N)$, respectively.

5.3. Performance evaluation

In Fig. 5, the average response times are shown for varying replication amounts, assuming that the previous search results are not cached in the data centers. These response time values are obtained using the simulation setup described in Section 3. They include the cost of processing queries over the indexes in the local and non-local data centers as well as the round-trip network latencies between the data centers and the network latencies between users and their local data centers (Cambazoglu, Varol, et al., 2010).

According to the figure, the best-performing replication strategy is `DR-G`, especially at low replication rates. In the `World` setup, at 16% replication, it outperforms its respective baseline `B-G` by about 5%. The performance gap between `DR-L` and `B-L` is relatively small.

The fraction of queries that can be fully answered by the local data centers increases as the replication amount increases (see Fig. 6). Hence, replicating more documents implies savings in network latencies. On the other hand, replication leads to an increase in local index sizes, implying an increase in query processing times. Consequently, the lowest average response times are observed with replication amounts between no replication and full replication. The optimum replication amount depends on the setup. In `Europe`, the lowest average response time is attained by `DR-G` with replication amounts in the 2–4% range, achieving about 17% reduction over the average response time attained by `NR` (117 ms versus 141 ms). However, as the replication amount further increases, the average response times start to increase. Beyond 32% replication, all strategies result in average response times higher than that of `NR`. For the `World` setup, the lowest response times are achieved by `DR-G` at relatively larger replication amounts. At 16% replication, the average response time saving relative to `FR` is about 19%. Interestingly, the performance of the replication strategies relative to `FR` deteriorates if the replication amount goes below 1%.

It is interesting to observe that, in the `Europe` setup, `NR` achieves a considerably lower average response time than `FR` (141 ms versus 206 ms), whereas `FR` performs better than `NR` in the `World` setup (209 ms versus 270 ms). This is because, in the `Europe` setup, query response times are dominated by the cost of evaluating queries over the index, whereas network latencies between the data centers form the main overhead in the `World` setup. Also, we observe that both the baseline and proposed strategies can outperform the `NR` and `FR` strategies when the replication amounts are low.

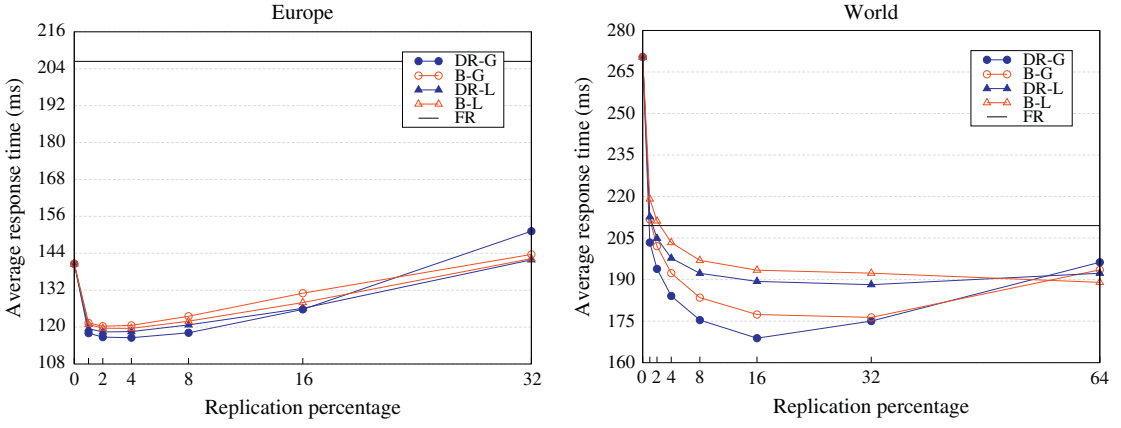

Fig. 5. Average response time as the replication amount increases (no result cache).

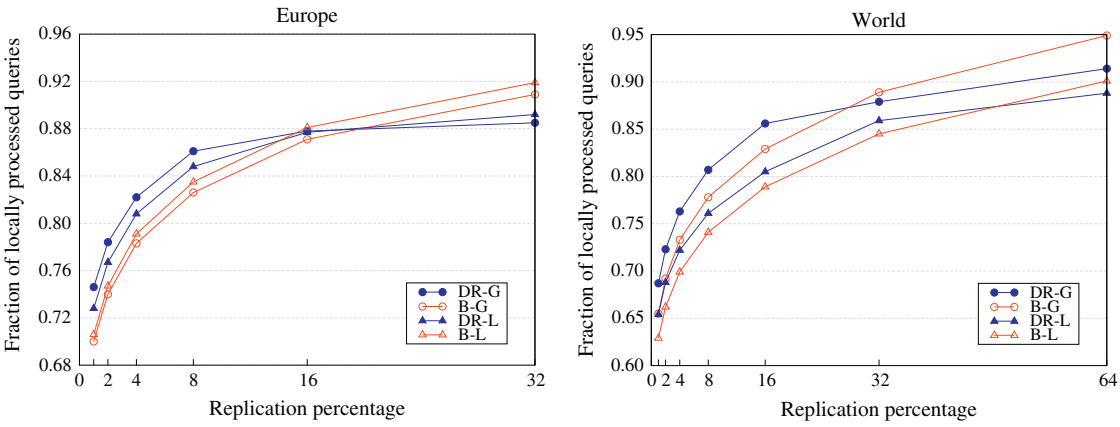

Fig. 6. Fraction of locally processed queries (no result cache).

Fig. 7 shows the impact of using an infinite result cache on the average response times. As expected, there is a considerable reduction in response times as many queries can be readily served by the cache, without incurring any processing or network latency overheads. We observe that, in the `Europe` setup, the lowest response time is now achieved at slightly higher replication amounts. Similarly, in the `World` setup, more documents need to be replicated to perform better than the `FR` strategy.

6. Optimizing query workload

6.1. Objective

In this section, we continue to focus on a search architecture similar to the one described in Section 5, i.e., certain queries are processed on non-local data centers that are determined by an oracle query forwarding algorithm. The performance objective we consider is to reduce the query processing workload of the system. In particular, as a closely related optimization objective, we aim to minimize the average number of non-local data centers that participate in processing of queries by selectively replicating documents on data centers. We first provide some notation before presenting our solution. The remote load incurred by a query is defined as follows.

Definition 4 (*Remote load of a query*). For a given document replication Φ, the remote load $\omega(q_i, \Phi)$ of a query q_i is defined as the number of non-local data centers $C_r \in \mathcal{C} - \{\widehat{C}_i\}$ that contain at least one relevant document not available on local data center $\widehat{C}_i$, i.e.,

$$\omega(q_i, \Phi) = \left| \left\{ C_r \in \mathcal{C} : (\mathcal{D}_r \cap \mathcal{R}_i) \not\subseteq \widehat{\mathcal{D}}_i^{\Phi} \right\} \right|. \tag{12}$$

The total remote load of a document replication Φ is defined as a sum over the remote loads of queries in a $\mathcal{Q}$, after the documents are replicated via Φ, i.e.,

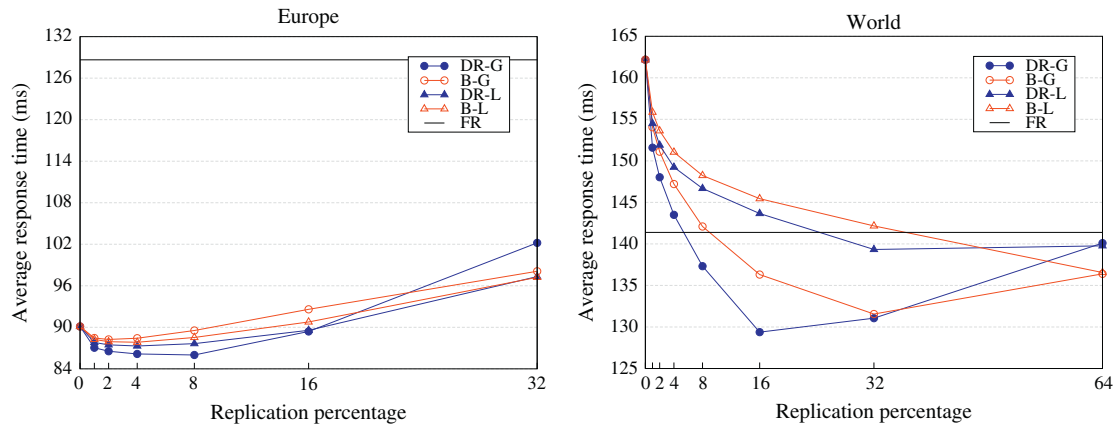

Fig. 7. Average response time as the replication amount increases (infinite result cache).

$$\Omega(\mathcal{Q}, \Phi) = \sum_{q_i \in \mathcal{Q}} \omega(q_i, \Phi). \tag{13}$$

The goal now becomes to find a feasible document replication Φ that minimizes $\Omega(\mathcal{Q}, \Phi)$ as the objective of the DR-G and DR-L problems.

6.2. Solution

Our solution to the DR-G problem is based on a combinatorial reduction to the adapted version of the set union knapsack problem (see Section 5.2). For each pair of document $d_j \in \mathcal{D}$ and data center $C_\ell \in \mathcal{C}$ such that $d_j \in \mathcal{D} - \mathcal{D}_\ell$, we introduce an item $t_{j\ell}$ in the main set $\mathcal{T}$ with an associated weight $w_{j\ell} = s_j$. Moreover, for each pair of query $q_i \in \mathcal{Q}$ and non-local data center $C_r \in \mathcal{C} - \{C_\ell\}$, we introduce a set $S_{ir} = \{t_{j\ell} : d_j \in \mathcal{D}_r \cap \mathcal{R}_i, C_\ell = \widehat{C}_i\}$ into family $\mathcal{F}$. The capacity W is set to the global replication capacity G. After having a solution $\mathcal{T}^*$ to the reduced problem, we form the document replication Φ as

$$\Phi(d_j) = \{C_\ell : t_{j\ell} \in \mathcal{T}^*\}. \tag{14}$$

In this formulation, each item $t_{j\ell} \in \mathcal{T}^*$ represents the replication of document d_j on data center C_ℓ. Thus, each set $S_{ir} \subseteq \mathcal{T}^*$ implies $(\mathcal{D}_r \cap \mathcal{R}_i) \subseteq \widehat{\mathcal{D}}_i^{\Phi}$ for query $q_i \in \mathcal{Q}$. This, in turn, implies that q_i is not forwarded to C_r, decreasing by 1 the remote workload $w(q_i, \Phi)$ incurred by q_i. Due to (13), the objective of the reduced problem correctly captures the minimization of the total remote load $\Omega(\mathcal{Q}, \Phi)$. We omit a discussion on the capacity constraint since it is similar to the formulations in Sections 4 and 5.

For the DR-L problem, we use a modified version of the above solution. Because each data center has its own local replication capacity, we solve a separate problem for each data center C_ℓ. For each document $d_j \in \mathcal{D} - \mathcal{D}_\ell$, we introduce an item $t_{j\ell}$ in $\mathcal{T}_\ell$ with an associated weight $w_{j\ell} = s_j$. For each pair of query $q_i \in \mathcal{Q}_\ell$ and non-local data center $C_r \in \mathcal{C} - \{C_\ell\}$, we introduce a set $S_{ir}^{\ell} = \{t_{j\ell} : d_j \in \mathcal{D}_r \cap \mathcal{R}_i\}$. The capacity W_ℓ is set to the local replication capacity L_ℓ. After having every solution set $\mathcal{T}_\ell^*$ associated with each data center C_ℓ, we form the document replication Φ as

$$\Phi(d_j) = \{C_\ell : t_{j\ell} \in \mathcal{T}_\ell^*\}. \tag{15}$$

For both the DR-G and DR-L problems, the encoding of the solution(s) has a complexity of $O(NK + MK + Mk)$. The complexity of the heuristic(s) for DR-G and DR-L are $O(MK + Mk + NK\lg(NK))$ and $O(MK + Mk + NK\lg N)$, respectively. In both problems, the decoding of the obtained solution(s) has $O(NK)$-time complexity. Hence, overall, the solutions to the DR-G and DR-L problems have time complexities of $O(MK + Mk + NK\lg(NK))$ and $O(MK + Mk + NK\lg N)$, respectively.

We note that although the constructed problem instances are different in case of optimizing the average response time and optimizing the workload, the solutions yield exactly the same replication pattern. This is because an item appears the same number of times among the sets and the heuristic takes into account only this number to decide on the replication pattern.

6.3. Performance evaluation

The workload values reported in this section are obtained by assuming that the workload incurred by a query on a data center is proportional to the size of the posting lists that need to be processed in that data center. The values are normalized by the workload estimated for query evaluation over the entire index. Hence, the normalized workload is 1.0 for the FR strategy.

Fig. 8 shows the normalized query workloads as the replication amount varies. According to the figure, with very little replication (e.g., 1%), it is possible to obtain large savings in the query workload. These large savings are mainly due to

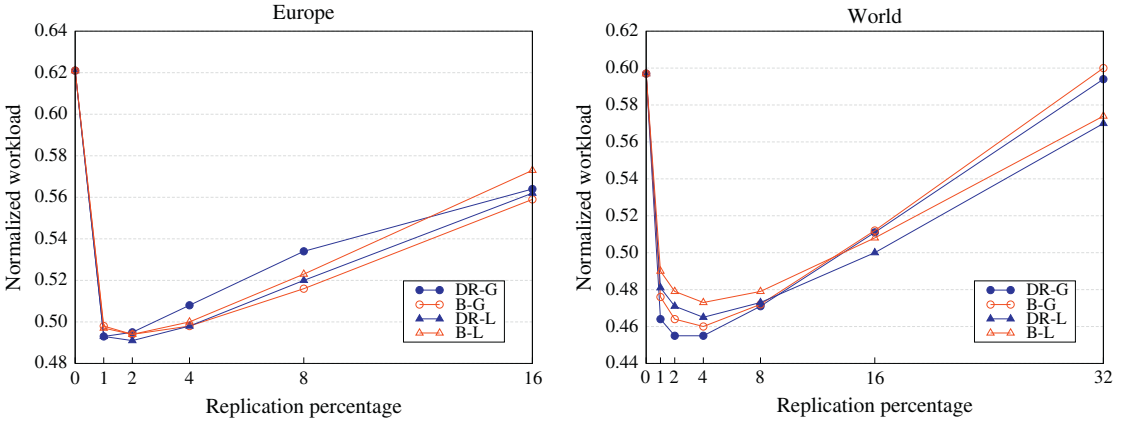

Fig. 8. Normalized workload as the replication amount increases (no result cache).

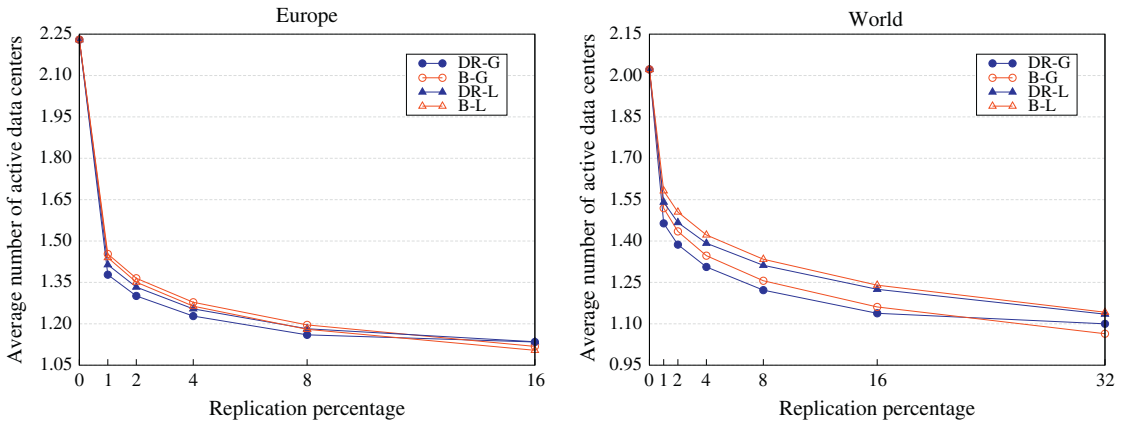

Fig. 9. Average number of active data centers (no result cache).

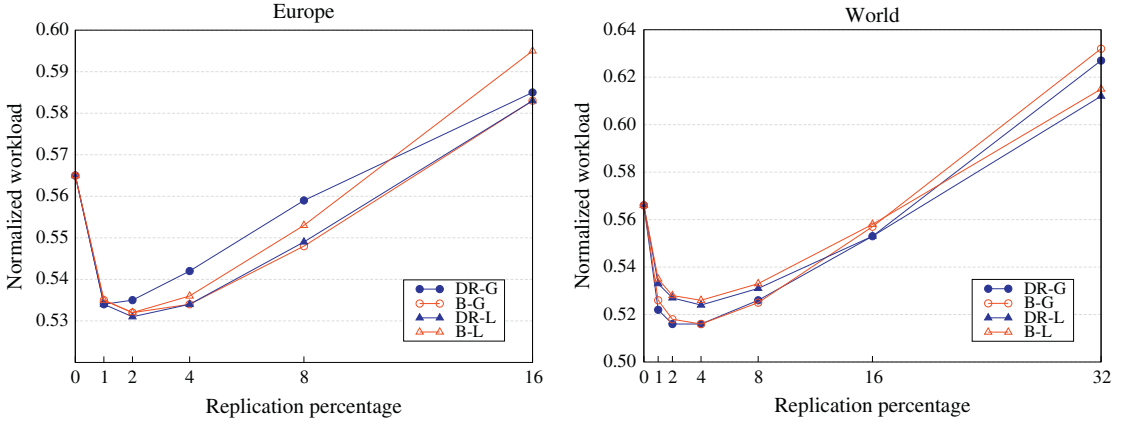

Fig. 10. Normalized workload as the replication amount increases (infinite result cache).

the sharp decrease in the average number of data centers that are active in query processing (see Fig. 9). However, as we replicate more documents, the local index sizes start to increase, diminishing the gains obtained by avoiding non-local computation.

In general, the proposed algorithms perform similar to the baseline algorithms. In the `Europe` setup, we observe that it is possible to decrease the workload by 21% (with 2% replication) relative to `NR`. In the `World` setup, the workload gain relative to `NR` goes up to 24% (with 4% replication). In both setups, it is possible to reduce the workload relative to `FR` by more than a half, replicating only a small fraction of the documents (e.g., 1%).

Fig. 10 shows the normalized query workload values assuming that an infinite result cache is deployed in data centers. In general, the performance behavior is not affected much by the presence of the result cache. On the other hand, the result cache has a major impact on the query workload. In case of the `FR` strategy, result caching leads to workload reductions of about 48% and 41% in the `Europe` and `World` setups, respectively (these values are not displayed in the plots).

7. Impact of query forwarding

In Sections 5 and 6, where we tried to optimize the average query response time and the query workload, respectively, we assumed that certain queries are forwarded between the data centers via an oracle query forwarding algorithm. However, in Section 4, where we tried to optimize the search quality, we assumed that queries are not forwarded between the data centers. Hence, in this section, we separately investigate the impact of query forwarding on the trade-off between the search quality and the average query response time.

Fig. 11 demonstrates the relative performance in precision and query response time values if we add query forwarding capability to a search system where the queries are originally not forwarded. In the figure, the values on the *x* axes of the two plots show the reciprocal of the average precision values attained by the search system where no queries are

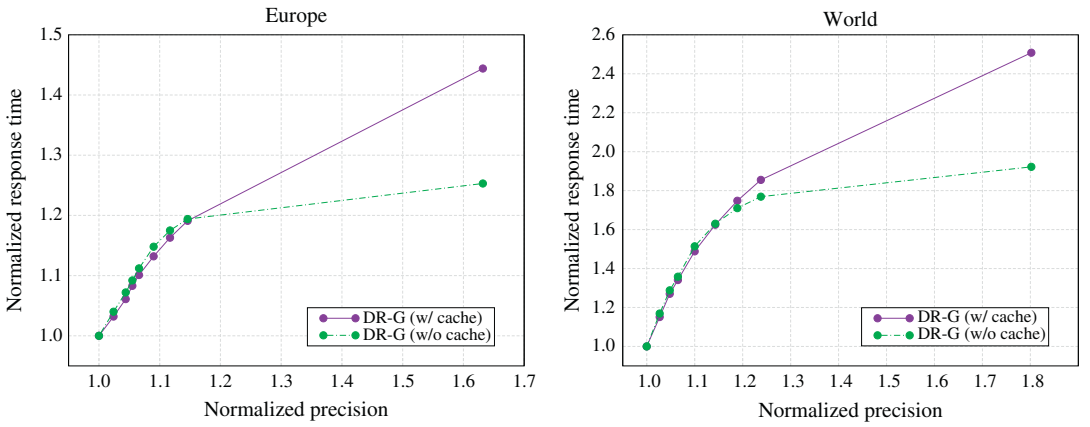

Fig. 11. The impact of query forwarding on the precision and query response time as the replication amount varies.

forwarded.[8] The values on the *y* axes show the response times for the system with oracle query forwarding capability. These values are normalized by the respective response times observed on the system that has no query forwarding capability. For document replication, we use the respective heuristics proposed for the DR-G problem. The data points in the plots correspond to particular replication amounts (left to right: 100%, 64%, 32%, 16%, 8%, 4%, 2%, 1%, and 0%).

As expected, when queries are forwarded, we observe gains in precision in exchange of an increase in average query response time. In particular, in the `World` setup, when the replication amount is 1% (the second data point from the right), the average response time almost doubles in exchange of about 25% increase in precision. In the same setup, achieving precision improvements similar to those in the `Europe` setup leads to relatively higher response time increase and higher document replication amounts. In general, result caching does not significantly affect the trade-off between search quality and response time, except for very low replication amounts.

8. Related work

There has been much research on distributed web search engines (Baeza-Yates, Castillo, Junqueira, Plachouras, & Silvestri, 2007; Barroso, Dean, & Hölzle, 2003; Cacheda, Carneiro, Plachouras, & Ounis, 2007a, Cacheda, Carneiro, Plachouras, & Ounis, 2007b; Orlando, Perego, & Silvestri, 2001). However, the feasibility of multi-site, geographically distributed search engines has not been studied until recently (Baeza-Yates et al., 2009; Cambazoglu, et al., 2008, 2009; Cambazoglu, Varol, et al., 2010). Cambazoglu, et al. (2008) investigated the feasibility of geographically distributed web crawling. Reduced network latencies between data centers and web sites are shown to bring significant performance benefits for geographically distributed web crawling, compared to a centralized web crawling architecture. Baeza-Yates et al. (2009) conducted a study to demonstrate the performance gains in multi-site web search engine architectures. A cost model is developed and some simulation results are provided, illustrating the feasibility of multi-site web search. The same work also proposed a simple query forwarding algorithm that guarantees the correctness of the top *k* search results, relative to the results obtained by a centralized architecture. Cambazoglu et al. (2009) investigated the trade-off in search efficiency and effectiveness in multi-site web search architectures. An analytical cost model is developed to quantify the impact of replication and query forwarding on search efficiency. The same work also reported some results about the impact of geographically distributed web crawling on search quality, in particular, on the impact of web coverage on relevance. Using a setup similar to ours, Cambazoglu, Varol, et al. (2010) proposed a linear-programming-based query forwarding algorithm that improves the performance of the query forwarding algorithm proposed in (Baeza-Yates et al., 2009).

Both (Baeza-Yates et al., 2009) and (Cambazoglu, Varol, et al., 2010) employed simple replication algorithms in their performance evaluations. The former work prioritizes documents for replication according to their popularity in past search results. The latter work improves over the former work by also considering the space overheads of documents. We used a slightly modified version of the algorithm in (Cambazoglu, Varol, et al., 2010) as our baseline.

To our knowledge, so far, no work has proposed sophisticated document replication algorithms for multi-site search engines. However, replication is previously considered in many other contexts: content delivery networks (Kangasharju, Roberts, & Ross, 2002; Rabinovich, Rabinovich, Rajaraman, & Aggarwal, 1999), distributed database systems (Apers, 1998; Wolfson, Jajodia, & Huang, 1997; Deris, Abawajy, & Mamat, 2008; Loukopoulos & Ahmad, 2000; Plattner & Alonso, 2004),

[8] Recall that the precision is always one under the oracle query forwarding assumption.

multimedia databases (Kwok, Karlapalem, Ahmed, & Pun, 1996), the Grid (Ranganathan & Foster, 2001, 2003; Lamehamedi, Shentu, Szymanski, & Deelman, 2003; Tang, Lee, Tang, & Yeo, 2006; Stockinger et al., 2002; Chang, Chang, & Lin, 2007; Tang, Lee, Yeo, & Tang, 2005; Abiteboul, Bonifati, Cobéna, Manolescu, & Milo, 2003), wireless networks (Jin & Wang, 2005), P2P (Sozio, Neumann, & Weikum, 2008; Cohen & Shenker, 2002; Yamamoto, Maruta, & Oie, 2005), and distributed web servers (Zhuo, Wang, & Lau, 2003; Tenzakhti, Day, & Ould-Khaoua, 2004; Tse, 2005). There is also a body of literature on replication for achieving fault-tolerance and consistency in distributed systems (Wolfson et al., 1997; Loukopoulos & Ahmad, 2000; Khan & Ahmad, 2004) and on replica placement (i.e., mirroring) (Korupolu, Plaxton, & Rajaraman, 1998; Radoslavov, Govindan, & Estrin, 2002; Qiu, Padmanabhan, & Voelker, 2001; Cronin et al., 2002). We omit these works here as they are not directly relevant to our context. Interested reader may refer to (Helal, Bhargava, & Heddaya, 1996) for an overview of data replication in distributed systems and to (Rabinovich & Spatschek, 2002) for document replication in the Web. Karlsson, Karamanolis, and Mahalingam (2002) provide a good classification of papers on document replication. There are also several surveys on the topic (Loukopoulos, Ahmad, & Papadias, 2002; Saito & Shapiro, 2005; Rabinovich, 1998; Khan & Ahmad, 2008).

In the rest of the section, we summarize only the works that are most relevant to ours. We specifically focus on the algorithmic aspects of previous work on data replication. Lu and McKinley (1999) investigate the problem of selecting relevant partial replicas in an information retrieval (IR) system using the inference network. They also study how to improve the performance of an IR system performance using partial replication and caching (Lu & McKinley, 2000). Baev and Rajaraman (2001) propose an approximation algorithm for the object replication problem in parallel databases. Kangasharju et al. (2002) provide several greedy document replication heuristics to minimize data transfer costs in CDNs. Zhuo et al. (2003) present four different algorithms, which make use of past document access patterns, to replicate documents in a geographically distributed web server. Among the presented algorithms, the proximity-aware algorithm is similar to ours in that documents are replicated according to their popularities in different Internet regions. Tenzakhti et al. (2004) provide distributed replication heuristics for dynamically changing web collections. Tse (2005) proposes a replication algorithm with an approximation guarantee to balance the workload as its objective. Brefeld, Cambazoglu, and Junqueira (2011) develop a machine-learned classification technique to replicate newly discovered web documents on data centers. Their problem is different than ours because document access frequencies are not available in their scenario. Hence, the replication decisions are made solely based on the features extracted from the documents (e.g., region and language). In a recent study, Blanco, Cambazoglu, Junqueira, Kelly, and Leroy (2011) propose document assignment strategies for a multi-site search engine, aiming to assign each document to single a data center responsible for indexing the document. That work, however, does not consider replication of documents.

The closest application of document replication to ours is that on geographically distributed web servers, where the main objective is to prevent possible hot spots in accessing documents on a server and to provide a good load balance across all servers. However, our problem setting differs in several ways. First, geographically distributed search engines are likely to have their own private high-speed networks, like CDNs (Qiu et al., 2001). This implies that the communication overhead incurred by data replication is not likely to be an issue. Therefore, in the replication algorithms, it is not vital to take this overhead into account. Second, the objectives we try to optimize (e.g., search quality) are quite different than the objectives in other works (e.g., balancing the workload (Zhuo et al., 2003), reducing the volume of data transfer (Apers, 1998; Kwok et al., 1996; Zhuo et al., 2003), increasing replica availability Tewari & Adam, 1992). Third, in case of geographically distributed search engines, processing of a query may be performed on multiple sites, each contributing some documents to the final result set, whereas other works assume that a single document is requested from a single site. Finally, in our case, every user issues his queries to a local data center, i.e., a content-aware request dispatcher is not used (Zhuo et al., 2003).

9. Conclusions

In this work, we investigated the problem of replicating documents on a multi-site, geographically distributed web search engine. We devised different heuristics aiming to improve various important performance criteria, such as search quality, average query response time, and query workload. The proposed heuristics are shown to slightly improve over the baseline replication algorithms used in previous works.

The main finding of our work is that replication can be a feasible alternative to fully partitioned or fully replicated multi-site search engines. We demonstrated that the optimum performance is achieved at certain replication amounts. We experimentally identified these optimum replication rates for different performance metrics and different search engine settings: the best replication rates are between 4% and 16% for minimizing the average response time and between 2% and 4% for minimizing the query workload. We also confirmed the performance benefits of result caching in multi-site search architectures.

A possible extension is to investigate the performance of our replication heuristics on related search architectures (e.g., P2P). Another research direction is to combine the replication problem with the user-to-site assignment problem, where the objective is to find a good static assignment between users and local data centers. Finally, the oracle query forwarding algorithm used in our work can be replaced with a practical query forwarding algorithm and the impact on the optimum replication rates can be observed.

Acknowledgments

This publication is based on work performed in the framework of the Project COAST-ICT-248036, funded by the European Community, and partially supported by the Scientific and Technological Research Council of Turkey under grant EEEAG-109E019 and COST Action IC080 ComplexHPC.

References

Abiteboul, S., Bonifati, A., Cobéna, G., Manolescu, I., & Milo, T. (2003). Dynamic XML documents with distribution and replication. In *Proceedings of the 2003 ACM SIGMOD internationall conference on the management of data* (pp. 527–538).

Apers, P. M. G. (1998). Data allocation in distributed database systems. *ACM Transactions on Database Systems, 13*(3), 263–304.

Baev, I. D., & Rajaraman, R. (2001). Approximation algorithms for data placement in arbitrary networks. In *Proceedings of the 12th annual ACM-SIAM symposium discrete algorithms* (pp. 661–670).

Baeza-Yates, R., Castillo, C., Junqueira, F., Plachouras, V., & Silvestri, F. (2007). Challenges on distributed web retrieval. In *Proceedings of the 23rd international conference on data engineering* (pp. 6–20).

Baeza-Yates, R., Gionis, A., Junqueira, F., Plachouras, V., & Telloli, L. (2009). On the feasibility of multi-site web search engines. In *Proceedings of the 18th ACM conference on the information and knowledge management* (pp. 425–434).

Barroso, L. A., Dean, J., & Hölzle, U. (2003). Web search for a planet: The Google cluster architecture. *IEEE Micro, 23*(2), 22–28.

Blanco, R., Cambazoglu, B. B., Junqueira, F. P., Kelly, I., & Leroy, V. (2011). Assigning documents to master sites in distributed search. In *Proceedings of the 20th ACM conference information and knowledge management* (pp. 67–76).

Brefeld, U., Cambazoglu, B.B., Junqueira, F.P., 2011. Document assignment in multi-site search engines. In *Proceedings of the 4th ACM international conference web search and data mining* (pp. 575–584).

Cacheda, F., Carneiro, V., Plachouras, V., & Ounis, I. (2007a). Performance analysis of distributed information retrieval architectures using an improved network simulation model. *Information Processing and Management, 43*(1), 204–224.

Cacheda, F., Carneiro, V., Plachouras, V., & Ounis, I. (2007b). Performance comparison of clustered and replicated information retrieval systems. In *Proceedings of the 29th European conference information retrieval* (pp. 124–135).

Cambazoglu, B. B., & Aykanat, C. (2006). Performance of query processing implementations in ranking-based text retrieval systems using inverted indices. *Information Processing and Management, 42*(4), 875–898.

Cambazoglu, B. B., Junqueira, F. P., Plachouras, V., Banachowski, S., Cui, B., Lim, S., et al. (2010). A refreshing perspective of search engine caching. In *Proceedings of the 19th international conference World Wide Web* (pp. 181–190).

Cambazoglu, B. B., Plachouras, V., & Baeza-Yates, R. (2009). Quantifying performance and quality gains in distributed web search engines. In *Proceedings of the 32nd international ACM conference research and development in information retrieval* (pp. 411–418).

Cambazoglu, B. B., Plachouras, V., Junqueira, F., & Telloli, L. (2008). On the feasibility of geographically distributed web crawling. In *Proceedings of the 3rd international conference scalable information systems* (31:1–31:10).

Cambazoglu, B. B., Varol, E., Kayaaslan, E., Aykanat, C., & Baeza-Yates, R. (2010). Query forwarding in geographically distributed search engines. In *Proceedings of the 33rd international ACM conference research and development in information retrieval* (pp. 90–97).

Cambazoglu, B. B., Zaragoza, H., Chapelle, O., Chen, J., Liao, C., Zheng, Z., et al. (2010). Early exit optimizations for additive machine learned ranking systems. In *Proceedings of the 3rd ACM international conference web search and data mining* (pp. 411–420).

Chang, R.-S., Chang, J.-S., & Lin, S.-Y. (2007). Job scheduling and data replication on data grids. *Future Generation Computer Systems, 23*(7), 846–860.

Cohen, E., & Shenker, S. (2002). Replication strategies in unstructured peer-to-peer networks. In *Proceedings of the 2002 conference applications, technologies, architectures, and protocols for computer communications* (pp. 177–190).

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to algorithms* (3rd ed.). The MIT Press.

Cronin, E., Jamin, S., Jin, C., Kurc, A. R., Raz, D., & Shavitt, Y. (2002). Constrained mirror placement on the Internet. *IEEE Journal of Selected Areas in Communications, 20*(7), 1369–1382.

Deris, M. M., Abawajy, J. H., & Mamat, A. (2008). An efficient replicated data access approach for large-scale distributed systems. *Future Generation Computer Systems, 24*(1), 1–9.

Helal, A. A., Bhargava, B. K., & Heddaya, A. A. (1996). *Replication techniques in distributed systems*. Norwell, MA, USA: Kluwer Academic Publishers.

Jin, S., & Wang, L. (2005). Content and service replication strategies in multi-hop wireless mesh networks. In *Proceedings of the 8th ACM international symposium on modeling, analysis and simulation of wireless and mobile systems* (pp. 79–86).

Kangasharju, J., Roberts, J., & Ross, K. W. (2002). Object replication strategies in content distribution networks. *Computer Communications, 25*(4), 376–383.

Karlsson, M., Karamanolis, C., & Mahalingam, M. (2002). A framework for evaluating replica placement algorithms. Tech. rep. HPL-2002-219, HP Labs.

Kellerer, H., Pferschy, U., & Pisinger, D. (2004). *Knapsack problems*. Springer Verlag.

Khan, S. U., & Ahmad, I. (2004). Heuristic-based replication schemas for fast information retrieval over the Internet. In *Proceedings of the 17th international conference parallel and distributed computing systems* (pp. 278–283).

Khan, S. U., & Ahmad, I. (2008). Comparison and analysis of ten static heuristics-based Internet data replication techniques. *Journal of Parallel and Distributed Computing, 68*(2), 113–136.

Korupolu, M. R., Plaxton, C. G., & Rajaraman, R. (1998). Analysis of a local search heuristic for facility location problems. In *Proceedings of the 9th annual ACM-SIAM symposium on discrete algorithms*. pp. 1–10.

Kwok, Y. K., Karlapalem, K., Ahmed, I., & Pun, N. M. (1996). Design and evaluation of data allocation algorithms for distributed multi-media database systems. *IEEE Journal Selected Areas in Communications, 14*(7), 1332–1348.

Lamehamedi, H., Shentu, Z., Szymanski, B., & Deelman (2003). Simulation of dynamic data replication strategies in data grids. In *Proceedings of the 17th international symposium on parallel and distributed processing* (p. 100b).

Loukopoulos, T., & Ahmad, I. (2000). Static and adaptive data replication algorithms for fast information access in large distributed systems. In *Proceedings of the 20th international conference distributed computing systems* (pp. 385–392).

Loukopoulos, T., Ahmad, I., & Papadias, D. (2002). An overview of data replication on the Internet. In *Proceedings of the 2002 international symposium on parallel architectures, algorithms and networks* (pp. 27–32).

Lu, Z., & McKinley, K. S. (1999). Partial replica selection based on relevance for information retrieval. In *Proceedings of the 22nd international ACM conference research and development in information retrieval* (pp. 97–104).

Lu, Z., & McKinley, K. S. (2000). Partial collection replication versus caching for information retrieval systems. In *Proceedings of the 23rd international ACM conference research and development in information retrieval* (pp. 248–255).

Ntoulas, A., & Cho, J. (2007). Pruning policies for two-tiered inverted index with correctness guarantee. In *Proceedings of the 30th international ACM conference research and development in information retrieval* (pp. 191–198).

Orlando, S., Perego, R., & Silvestri, F. (2001). Design of a parallel and distributed web search engine. In *Proceedings of the 2001 parallel computing conference* (pp. 197–204).

Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., & Johnson, D. (2005). Terrier information retrieval platform. In D. E. Losada & J. M. Fernández-Luna (Eds.). *Advances in information retrieval. Lecture notes in computer science* (Vol. 3408, pp. 517–519). Berlin/Heidelberg: Springer.

Plattner, C., & Alonso, G. (2004). Ganymed: Scalable replication for transactional web applications. In *Proceedings of the 5th ACM/IFIP/USENIX international conference middleware* (pp. 155–174).
Qiu, L., Padmanabhan, V. N., & Voelker, G. M. (2001). On the placement of web server replicas. In *Proceedings of the 20th annual joint conference IEEE computer and communications societies* (pp. 1587–1596).
Rabinovich, M. (1998). Issues in web content replication. *Data Engineering Bulletin, 21*(4), 21–29.
Rabinovich, M., Rabinovich, I., Rajaraman, R., & Aggarwal, A. (1999). A dynamic object replication and migration protocol for an Internet hosting service. In *Proceedings of the IEEE international conference distributed computing systems* (pp. 101–113).
Rabinovich, M., & Spatschek, O. (2002). *Web caching and replication*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.
Radoslavov, P., Govindan, R., & Estrin, D. (2002). Topology-informed Internet replica placement. *Computer Communications, 25*(4), 384–392.
Ranganathan, K., & Foster, I. (2001). Identifying dynamic replication strategies for high performance data grids. In *Proceedings of the 2nd international workshop on grid computing* (pp. 75–86).
Ranganathan, K., & Foster, I. (2003). Simulation studies of computation and data scheduling algorithms for data grids. *Journal of Grid Computing, 1*(1), 53–62.
Saito, Y., & Shapiro, M. (2005). Optimistic replication. *ACM Computer Surveys, 37*(1), 42–81.
Sozio, M., Neumann, T., & Weikum, G. (2008). Near-optimal dynamic replication in unstructured peer-to-peer networks. In *Proceedings of the 27th ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems* (pp. 281–290).
Stockinger, H., Samar, A., Holtman, K., Allcock, B., Foster, I., & Tierney, B. (2002). File and object replication in data grids. *Cluster Computing, 5*(3), 305–314.
Tang, M., Lee, B., Tang, X., & Yeo, C. (2006). The impact of data replication on job scheduling performance in the data grid. *Future Generation Computer Systems, 22*(3), 254–268.
Tang, M., Lee, B.-S., Yeo, C.-K., & Tang, X. (2005). Dynamic replication algorithms for the multi-tier data grid. *Future Generation Computer Systems, 21*(5), 775–790.
Tenzakhti, F., Day, K., & Ould-Khaoua, M. (2004). Replication algorithms for the World-Wide Web. *Journal of Systems Architecture, 50*(10), 591–605.
Tewari, R., & Adam, N. R. (1992). Distributed file allocation with consistency constraints. In *Proceedings of the 12th international conference distributed computing systems* (pp. 408–415).
Tse, S. S. H. (2005). Approximate algorithms for document placement in distributed web servers. *IEEE Transactions on Parallel and Distributed Systems*, 489–496.
Wolfson, O., Jajodia, S., & Huang, Y. (1997). An adaptive data replication algorithm. *ACM Transactions on Database Systems, 22*(2), 255–314.
Yamamoto, H., Maruta, D., & Oie, Y. (2005). Replication methods for load balancing on distributed storages in P2P networks. In *Proceedings of the 2005 Symposium on Applications and the Internet* (pp. 264–271).
Zhuo, L., Wang, C.-L., & Lau, F. C. M. (2003). Document replication and distribution in extensible geographically distributed web servers. *Journal of Parallel and Distributed Computing, 63*(10), 927–944.