Distributed Search Efficiency and Robustness in Service-oriented Multi-agent Networks

Weimao Ke
Drexel University
3141 Chestnut St
Philadelphia, Pennsylvania 19104
wk@drexel.edu

ABSTRACT

We study decentralized searches in a large service-oriented agent network and investigate the influences of multiple factors on search efficiency. In this study we focus on overall system robustness and examine search performance in unstable environments where individual agents may fail or a system-wide attack may occur. Experimental results show that searches continue to be efficient when a large number of service agents become unavailable. Surprisingly, overall system performance in terms of a search path length metric improves with an increasing number of unavailable agents. Service unavailability also has an impact on the load balance of service agents. We plan to conduct further research to verify observed patterns and to understand related implications on system architecture design.

CCS CONCEPTS

•Information systems → Peer-to-peer retrieval; •Computing methodologies → *Self-organization;*

KEYWORDS

information network, service agents, distributed search, robustness

ACM Reference format:
Weimao Ke. 2017. Distributed Search Efficiency and Robustness in Service-oriented Multi-agent Networks. In *Proceedings of ICMSS '17, Wuhan, China, January 14-16, 2017,* 10 pages.
DOI: 10.1145/3034950.3034975

1 INTRODUCTION

Digital information is distributed in many large networked environments, in which it is increasingly difficult to collect all information in advance for centralized retrieval operations. The growing magnitude, dynamics, and heterogeneity of today's digital environments such as the Web pose great challenges for finding information in them. While classic information retrieval systems provide search operations by collecting and indexing information in advance, this centralized model has suffered from the increasing decentralization of information and systems [6].

We live in a distributed networked space, where information and intelligence are highly distributed. In reality, people have different expertise, share information with one another, and ask trusted peers for advice/opinions on various issues. The World Wide Web is a good example of information distribution, where web sites serve narrow information topics and tend to form communities through hyperlink connections [19, 20, 37]. Likewise, individual digital libraries maintain independent document collections and none claims to be all encompassing or comprehensive. There is no single global information repository.

Because of the distributed nature of information and its size, dynamics, and heterogeneity, it is extremely challenging, if not impossible, to collect, store, and process all information in one place for retrieval purposes. Centralized solutions will suffer from its vulnerability to scalability demands [6]. It has become critical to investigate alternative models beyond the state-of-the-art retrieval systems, particularly for searching a large, highly decentralized environment such as the Web. A potential candidate is to take advantage of existing distributed computing powers and design a new search architecture in which all systems can participate to help one another find information.

Research in areas such as distributed IR, peer-to-peer search, cloud/parallel computing, and multi- agent systems have addressed some related problems. Nonetheless, basic principles and fundamental theories about the findability of information and scalability of decentralized search in these environments remain to be studied and understood. Without a better understanding, solutions for the next generation web searching and browsing will remain ad hoc. The proposed research aims to investigate guiding theories and alternative models related to effective and efficient searches in large-scale, decentralized, and dynamic environments. The ultimate goal is to be able to connect people and desired information in a timely fashion regardless of their digital locality.

Our research studies the general problem of search and retrieval in a fully distributed/decentralized environment, where no global information is available nor can a centralized index be built. Specifically, it focuses on query routing for decentralized search and addresses the scalability challenge by integrating perspectives from information retrieval as well as complex network research. In this study, we aim to better understand how distributed service agents can continue to function efficiently in unstable environments. In particular, we are interested to know whether the system as a whole can stand and how well it performs when a significant number of members become unavailable and unresponsive due to individual failures or attacks.

2 RELATED WORK

State-of-the-art search engines, relying on information collection and data pre-processing in advance, face great scalability challenges because of the dynamics (too difficult to collect information) and the tremendous, growing amount (too expensive to pre-process data) of distributed information. Related challenges for distributed search have been studied in areas of distributed (federated) information retrieval, peer-to-peer networks, multi-agent systems, and complex networks [12, 16, 29, 49].

2.1 Distributed P2P Search

Recent distributed IR research has focused on distributed database content (and characteristics) discovery [42], database selection [23, 41], and result fusion [5, 32, 43]. Research has studied the effectiveness of database selection and result fusion given a relatively small number of distributed, persistent information collections [40]. Their scalability to larger, unstable environments remains an important question.

A peer-to-peer network often involves more than thousands, sometimes millions, of distributed peers who dynamically join and leave the community. Distributed hashing tables (DHTs) have been used in structured P2P environments for unique identifier lookup. Some studies applied DHTs for partitioning an indexing space across redundant peers for efficient location of popular information items [36, 46]. Others proposed the use of this technique for search in the presence of information overlap [10].

While DHT-based techniques are applicable in structured P2P environments, their resilience to transient populations and adaptability to content and topology changes remain open questions [35]. More important, when diverse information needs are to be served, it is extremely challenging for such techniques to create and maintain (update) a distributed index structure in a space- and traffic-efficient manner. Flooding is often the technique employed for maintaining indexing currency, which has received critiques for its computational costs.

DHTs, based on document-key-level index partitioning, are not the ideal technique for information retrieval in distributed environments, particularly in unstructured peer-to-peer networks. Alternative methods based on peer-level segmentation can be used to support search efficiency and effectiveness [8, 34]. Reorganization of collections around content clusters/topics was proposed to improve distributed retrieval effectiveness.

Semantic overlay networks (SONs), based on peer segmentation, have been widely used for distributed IR operations in unstructured networks [16, 18]. In SONs, peers with semantically similar content are clustered together, which in turn form a global (hierarchical) structure for efficient query routing [16]. These techniques were shown in experiments to improve retrieval performance.

One popular approach to SONs was to form a hierarchical network structure through re-organization of distributed systems/peers, in which super-peers assumed greater responsibilities for bridging/mediating across segments. However, some questioned the reliability of such an architecture as attacks on super peers (nodes or agents) can lead to a large disconnected structure [3, 35]. In addition, as Lu and Callan [34] observed, updating super-peers for changes in distributed collections is traffic intensive and may cause problems in environments where bandwidth is limited.

2.2 Search in Complex Networks

Regardless of various approaches to peer re-organization and segmentation, the underlying network structure appears to play an important role in conducting effective and efficient retrieval operations in distributed settings. As individual systems interconnect to form a global structure, finding relevant information in decentralized environments transforms into a problem concerning not only information retrieval but also complex networks. Understanding network structure or system interconnectivity will provide guidance on how decentralized search and retrieval methods can function in these information spaces.

In a variety of large interconnected environments, it is well known that any pair of individual nodes are separated by a very small number of others. In other words, small diameters are a common feature of many naturally, socially, or technically developed communities – a phenomenon known as *small world* or *six degrees of separation* [38, 47]. The small world phenomenon also appears in various types of large-scale digital information networks such as the World Wide Web [3, 4] and the network for email communications [17]. The small degree of separation shows promises on efficient traversal of such a network to reach any desired targets. Nonetheless, it remains challenging to identify shortcuts to *relevant* targets in the information retrieval context, where relevant as well as non-relevant information collections are all within short distances.

Network clustering represents one approach to understanding how network characteristics can be taken advantage of for efficient traversal of relevant paths. One level of clustering, in P2P research, is the identification of similar peers and segmentation of them based on topical relevance. As we discussed, semantic overlay networks (SONs) have been widely used for retrieval effectiveness and efficiency [8, 16, 18, 33]. Clustering enables similar peers to connect to each other and sometimes allows super peers to coordinate local reconstruction and to update remote connections for efficient query routing. Query propagation in local segments often leads to improved recall [8, 33].

Research on complex networks studies the problem in its basic form and shows promises as well. Particularly, studies showed that, with local intelligence and basic information about targets, members of a very large network are able to find very short paths (if not the shortest) to destinations collectively [11, 17, 30, 31, 38, 48]. The implication in IR is that relevant information, in various networked environments, is not only a few degrees (connections) away from the one who needs it but potentially findable. This provides the potential for distributed algorithms to traverse such a network to find relevant information efficiently.

In this respect, Kleinberg [30] conducted one of the key studies on decentralized search in small world networks. The research, based on an abstract lattice model and a clustering exponent α to control network clustering, discovered that some critical value of α enables optimal search efficiency. Particularly, in a d dimensional space, search time (or the number of hops required to reach a target) is bounded by $c \log^2 N$ only when $\alpha = d$, where N is network size. When α becomes either larger or smaller, search performance is greatly degraded. In other words, neither weak clustering nor strong clustering is desirable. A specific, balanced level of clustering must be maintained for search efficiency. Related studies on complex network search provided results consistent to this finding [11, 17, 29, 31, 48].

3 SEARCH SCALABILITY & ROBUSTNESS

Finding relevant information in distributed environments is a problem concerning complex networks and information retrieval. We know from the small world phenomenon, common in many real networks, that every piece of information is within a short radius from any location in a network. However, relevant information is only a tiny fraction of all densely packed information in the "small world."

If we allow queries to traverse the edges of a network to find relevant information, there has to be some association between the network space and the relevance space in order to orient searches. Random networks could never provide such guidance because edges are so independent of content that they have little semantic meaning. Fortunately, research has discovered that development of a wide range of networks follows not a random process but some preferential mechanism that captures "meanings."

3.1 Network Clustering and Searches

Distributed information retrieval, particularly unstructured peer-to-peer IR, relied on peer-level clustering for better decentralized search efficiency. Topical segmentation based techniques such as semantic overlay networks (SONs) have been widely used for efficient query propagation and high recall [8, 16, 18, 34]. Hence, overall, clustering was often regarded as beneficial whereas the potential *negative* impact of clustering (or over-clustering) on retrieval has often been overlooked.

Research on complex networks has found that a proper level of network clustering with some presence of remote connections has to be maintained for efficient searches [11, 30, 31, 44, 48]. Clustering reduces the number of "irrelevant" links and aids in creating topical segments useful for orienting searches. Without sufficient clustering, the network has too much randomness to guide efficient traversals because *weak ties* dominate. While searches may jump quickly from one place to another (hops) in the network space, there is no "gradient" to lead them toward targets. With very strong clustering, on the other hand, a network tends to be fragmented into local communities with abundant *strong ties* but few *weak ties* to bridge remote parts [21, 45]. Although searches might be able to move gradually toward targets, necessary "hops" become unavailable.

In other words, trade-off is required between *strong ties* for search orientation and *weak ties* for efficient traversal. In Granovetter's terms, whereas *strong ties* deal with local connections within small, well-defined groups, *weak ties* capture between-group relations and serve as bridges of social segments [21].

One key parameter widely used in complex network research for studying the impact of clustering is the *clustering exponent* α. Kleinberg [30] studied decentralized search in small world using a two dimensional model, in which peers had rich connections with immediate neighbors and sparse associations with remote ones. The probability p_r of connecting to a neighbor beyond the immediate neighborhood was proportional to $r^{-\alpha}$, where r was the search distance between the two in the dimensional space and α a constant called *clustering exponent*[1]. It was shown that only when *clustering exponent* $\alpha = 2$, search time (i.e., search path length) was optimal and bounded by $c(\log N)^2$, where N was the network size and c was some constant [29]. More generally, when $\alpha = d$ on a d-dimension space, decentralized search is optimal. Further studies conducted on small world networks as well as in distributed IR have shown consistent results [11, 25–27, 31, 44, 48].

3.2 Verified Scalability Model

Previous studies in distributed IR have found significant impacts of network clustering on search efficiency, that optimal search performance was supported by a specific, balanced level of network clustering (e.g., $\alpha = 2$ in several experiments). Further investigation is needed to understand the influences of other network structure variables and the scalability of searches in related settings.

Our research has focused on search efficiency and scalability with growing network sizes N and varied (distributed system) neighborhood size d distributions (degree distributions). We have proposed and validated a scalability function which we discuss below.

Let L denote search path length, the number of hops (distributed systems) a search has to traverse the network to reach a desired target. According to [29] and several studies in distributed IR, when network clustering is optimal, a reasonable relationship between $\hat{L}$ (expected value of L based on relevance/similarity searches) and network size N (the number of distributed systems in the network) is:

$$\hat{L} = \beta' \cdot (\log_b N)^{\lambda} \tag{1}$$

where β' is a constant and b is the logarithmic base. λ is an exponent parameter to be identified with empirical data.

Assume the majority of distributed systems (hops) have a neighborhood size (number of interconnected systems) d_m. Let L_g denotes the ideal search path length given a (imagined) perfect, global index of all distributed systems[2]. It can be shown that $L_g \propto \log_b N$ as well as $N \approx d_m^{L_g}$:

$$\begin{aligned} L_g &\propto \log_b N \qquad (2)\\ &\approx \log_{d_m} N \qquad (3) \end{aligned}$$

Hence, we can replace $\log_b N$ with $\log_{d_m} N$ (i.e. using d_m as the logarithmic base) in Equation 1, which becomes:

$$\begin{aligned} \hat{L} &= \beta \cdot (\log_{d_m} N)^{\lambda} \qquad (4)\\ &= \beta \cdot (\log N / \log d_m)^{\lambda} \qquad (5) \end{aligned}$$

where β is a constant and d_m the neighborhood size (degree) of majority distributed systems. To simulate real networks, a power-law function will be used for degree distribution $d \in [d_m, d_x]$, where d_m is the min degree (which the vast majority have in a

[1]The *clustering exponent* α is also known as the *homophily exponent* [44, 48].
[2]For example, when degree $d_m = 2$, L_g can be seen as the number of steps needed to perform a binary search (traversal of a binary tree) on N nodes.

power law) and d_x is the maximum value (which only a small number of nodes have).

3.3 Research Focus on Robustness

In a previous study, we validated the scalability model with large-scale experiments, identified the exponent λ, estimated the β coefficient with varied N and d_m settings, and predicted potential search efficiency in real-scale environments with millions to billions distributed systems [28].

In this study, we aim to better understand how distributed service agents can continue to function efficiently in unstable environments. In particular, we are interested to know whether the system as a whole can stand and how well it performs when a significant number of members become unavailable and unresponsive due to individual failures or attacks.

4 SIMULATION FRAMEWORK

We use multi-agent systems for a bottom-up investigation of decentralized IR functions [24]. We have developed a decentralized search platform for finding relevant information in distributed settings. Each agent represents an IR system, which has its document collection and can connect to others to route queries. We emphasize the societal view of agents who have local intelligence and can collaborate with one another to perform global search tasks. We illustrate the conceptual model in Figure 1 and elaborate on major components shown in Figure 2.

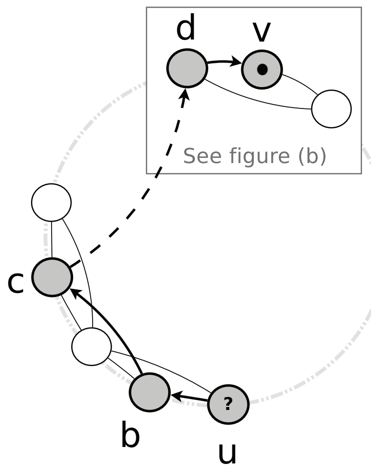

Figure 1: (a) Global view of the conceptual framework showing agents working together to route a query in the network space.

Figure 1 visualizes a 2D circle (1-sphere) representation of the information space. Let agent A_u be the one who has an information need whereas agent A_v has the relevant information. The problem becomes how agents in the connected society, without global information, can collectively construct a short path to A_v. When an agent receives a query, it first runs a local search operation to identify potential relevant information from its individual document collection. If local results are unsatisfactory, the agent will send

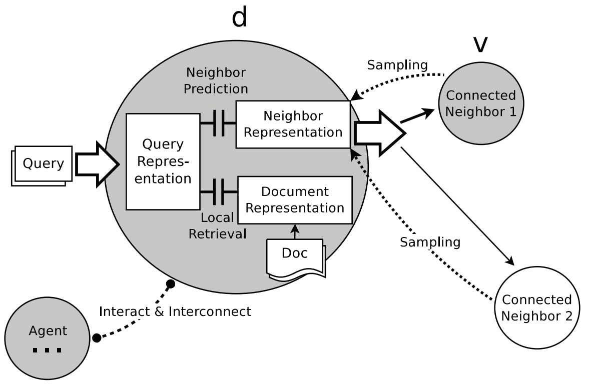

Figure 2: (b) Local view of the conceptual framework showing how components function within an agent's neighborhood.

the query to neighbors based on a predicting function using the query representation and information about neighbors. In Figure 1, the query traverses a search path $A_u \rightarrow A_b \rightarrow A_c \rightarrow A_d \rightarrow A_v$ to reach the target. While agents A_b and A_d help move the query toward the target gradually (through strong ties), agent A_c has a remote connection (weak tie) for the query to "jump." The entire network topology is self-organized by agents using an interconnectivity probability function supervised by clustering exponent α, which we discuss in Section 5.

4.1 Service-oriented Prototype

In addition to the simulation framework, we also plan to develop a prototype system for distributed web searching and browsing, by integrating major components illustrated in Figure 3. Essentially the software will be a set of modules and RESTful web services which can be deployed to different web sites to provide the following functions: indexing (updated) local documents, parsing query requests (through http), searching the local document collection (site pages), communicating with other agents (sites), and providing search results, etc. Once the prototype has been installed on multiple web servers, web sites hosted by these servers can work together in the described decentralized manner for information retrieval.

4.1.1 System Design, Modules and APIs. The prototype will first be implemented in node.js, which has gained increased support on the web. The implementation can certainly be replicated in other languages and web frameworks such as Ruby on Rails, Play framework, Python, PHP, and Perl. All exposed APIs will be implemented as RESTful web services with JSON. As shown in Figure 3, here are major modules and APIs:

- **Indexing module** which builds the inverted index for pages on the file system and in database;
- **Searching API** which receives a query from a user or directed from another node/site and identifies relevant information/documents from the local site;

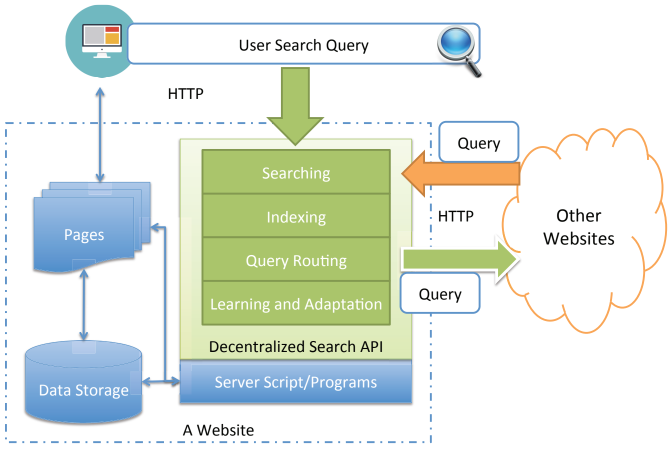

Figure 3: Service-oriented Architecture Prototype

- **Query routing module** which forwards queries to neighbors for further searching, with a **call-back API** to be contacted when results are ready.
- **Learning and adaptation module** which analyzes the site's interaction with other sites (e.g. from the history of responses and associated topics) and learns how to proceed with queries in the future.

The prototype system will have a built-in mechanism for self organization through distributed discovery of other nodes. When one node/site installs related APIs locally and joins the search network, the node which takes in the new one will introduce a list of other nodes to connect. The new node then decides which else to connect based on the build-in selection algorithm (see notes in section 5.3).

5 ALGORITHMS

The simulation framework for experiments was implemented in Java, based on two well-known open-source platforms: 1) JADE, a multi-agent system/middle-ware that complies with the FIPA (the Foundation for Intelligent Physical Agents) specifications [9], and 2) Lucene, a high-performance library for full-text search [22]. This section elaborates on specific algorithms implemented in the framework and used in the research.

5.1 Basic Functions

*5.1.1 TF*IDF Information Representation.* We use the Vector-Space Model (VSM) for information (document and query) representation [7]. Given that information is highly distributed, a global term space is not assumed. Instead, each agent processes information it individually has and produces a local term space, which is used to represent each information item using the TF*IDF (Term Frequency * Inverse Document Frequency) weighting scheme. An information item is then converted to a numerical vector where an item t is computed by:

$$W(t) = tf(t) \cdot log(\frac{N}{df(t)}) \tag{6}$$

where $tf(t)$ is the frequency of the term t of the term space in the information item, N is the total number of information items (e.g., documents) in an agent's local collection, and $df(t)$ is the number of information items in the set containing the term t of the term space. IDF values ($log(\frac{N}{df(t)})$) are computed within the information space of an agent given no global information.

*5.1.2 DF*INF Agent Representation.* For neighbor (agent) representation, we use a similar mechanism. Specifically, we assume agents are able to collect their direct neighbors' document frequency (DF) information and use it to represent each neighbor using a meta-document of terms. Distributed IR research has shown DF information useful for collection selection [13, 14, 39]. Treating each meta-document as a normal document, it becomes straightforward to calculate *neighbor frequency* (NF) values of terms, i.e., the number of meta-documents (neighbors) that contains a particular term. A meta-document (neighbor) is then represented as a vector where term t is computed by:

$$W'(t) = df'(t) \cdot log(\frac{N'}{nf'(t)}) \tag{7}$$

where $df'(t)$ is the frequency of the term t of the term space in the meta-document, N' is the total number of an agent's neighbors (meta-documents), and $nf'(t)$ is the number of neighbors containing the term t. We refer to this function as *DF*INF*, or document frequency * inverse neighbor frequency.

5.1.3 Similarity Scoring Function. Based on the term vectors produced by the *TF*IDF* (or *DF*INF*) representation scheme described above, pair-wise similarity values can be computed. Given a query q, the similarity score of a document d matching the query is computed by :

$$\sum_{t \in q} W(t) \cdot coord(q,d) \cdot queryNorm(q) \tag{8}$$

where $W(t)$ is the weight of term t given by equation 6 or 7, $coord(q,d)$ a coordination factor based on the number of terms shared by q and d, and $queryNorm(q)$ a normalization value for query q given the sum of squared weights of query terms. The function is a variation of the well-known cosine similarity measure adopted in Lucene [7, 22]. Given a query, an agent will use this scoring function to rank its local documents and determine whether it has relevant information. In addition, when an agent has to forward the query to a neighbor, similarity-based neighbor selection methods will use this to evaluate the relevance of each neighbor.

5.2 Neighbor Selection (Search) Methods

The similarity scoring function in Equation 8 can produce output about each neighbor's similarity/relevance to a query. Based on this output, we further propose the following strategies to decide which neighbors should be contacted for the query. Each search will keep track of all agents on the search path. All strategies below will ignore neighbors who have been contacted for a query to avoid loops. These strategies will be tested and compared in experiments.

5.2.1 SIM Search: Similarity-based Greedy Routing. Let k be the number of neighbors an agent has and $S = [s_1, .., s_k]$ be the similarity vector about each neighbor's similarity/relevance to a query. The *SIM* method sorts the vector and forwards the query to the neighbor with the highest score. With greedy routing, only one instance of the query will be forwarded from one agent to another until relevant information is found or some predefined conditions are met (e.g., the maximum search path length or Time to Live (TTL) is reached).

5.2.2 DEG Search: Degree-based greedy routing. In the degree-based strategy, we further assume that information about neighbors' degrees, i.e., their numbers of neighbors, is known to the current agent. Let $D = [d_1, .., d_k]$ denote degrees of an agent's neighbors. The *DEG* method sorts the D vector and forwards the query to the neighbor with the highest degree, regardless of what a query is about. Related degree-based methods were found to be useful for decentralized search in power-law networks [1, 2].

*5.2.3 SimDeg: Similarity*Degree Greedy Routing.* The *SimDeg* method is to combine information about neighbors' relevance to a query and their degrees. Simsek and Jensen [44] reasoned that a navigation decision relies on the estimate of a neighbor's distance from the target, or the probability that the neighbor links to the target directly, and proposed a measure based on the product of a degree term (d) and a similarity term (s) to approximate the expected distance. Following the same formulation, the *SimDeg* method uses a combined measure $SD = [s_1 \cdot d_1, .., s_k \cdot d_k]$ to rank neighbors, given neighbor relevance vector $S = [s_1, .., s_k]$ and neighbor degree vector $D = [d_1, .., d_k]$. A query will be forwarded to the neighbor with the highest $s \cdot d$ value. It was shown in studies that this combined method is sensitive to the ratio of values between two neighbors, not the actual values that might not be accurately measured [44].

5.3 Interconnectivity and Network Clustering

For network clustering, the first step is to determine how many links (degree d_u) each distributed system u should have. Once the degree is determined, the system will interact with a large number of other systems (from a random pool) and select only d_u systems as neighbors based on a connectivity probability function guided by the clustering exponent α.

In experiments on the ClueWeb09B collection, we collect information about web sitess (treated as distributed agents/systems) incoming hyperlinks and normalize the in-degrees as their d_u values. We control the range of degree distribution $[d_m, d_x]$ and study its impact on search performance with varied degree ranges. Given the number of incoming hyperlinks d'_u of system u, the normalized degree is computed by:

$$d_u = d_m + \frac{(d_x - d_m) \cdot (d'_u - d'_m)}{d'_x - d'_m} \tag{9}$$

where d'_x is the maximum degree value in the hyperlink in-degree distribution and d'_m the minimum value in the same distribution. Once degree d_u is determined from the degree distribution, a number of random systems/agents will be added to its neighborhood pool such that the total number of neighbors $\hat{d}_u \gg d_u$ ($\hat{d}_u = 1,000$ in this study). Then, the agent in question (u) queries each of the $\hat{d}_u$ neighbors (v) to determine their topical distance r_{uv}. Finally, the following connection probability function is used by system u to decide who should remain as neighbors (to build the interconnectivity overlay):

$$p_{uv} \propto r_{uv}^{-\alpha} \tag{10}$$

where α is the *clustering exponent* and r_{uv} the pairwise topical (search) distance. The finalized neighborhood size will be the expected number of neighbors, i.e., d_u. With a positive α value, the larger the topical distance, the less likely two systems/agents will connect. Large α values lead to highly clustered networks while small values produce random networks with many topically remote connections.

6 EXPERIMENTAL DESIGN

6.1 Data Collection

We rely on the ClueWeb09 Category B collection created by the Language Technologies Institute at Carnegie Mellon University for IR experiments. The ClueWeb09 collection contains roughly 1 billion web pages and 8 billion outlinks crawled during January - February 2009. The Category B is a smaller subset containing the first crawl of 50 million English pages from 3 million sites with 454 million outlinks. The ClueWeb09 dataset has been adopted by several TREC tracks including Web track and Million Query track [15]. Additional details about the ClueWeb09 collection can be found at http://boston.lti.cs.cmu.edu/Data/clueweb09/.

A hyperlink graph is provided for the entire collection and the Category B subset. In the Category B subset, there are 428,136,613 nodes and 454,075,604 edges (hyperlinks). Nodes include the first crawl of 50 million pages and additional pages that were linked to. Only 18,607,029 nodes are the sources (starting pages) of the edges (average 24 outlinks per node) whereas 409,529,584 nodes do not have outgoing links captured in the subset.

6.2 Network Model and Sizes

Each agent represents an IR system serving a collection of pages (documents). We assume that there is no global information about all document collections. Nor is there centralized control over individual agents. Agents have to represent themselves using local information they have and evaluate relevance based on that. Using the ClueWeb09 collection, we treat a web site/domain as a distributed system/agent and use hyperlinks between sites to construct the initial network.

We first construct a list of all web domains in the category B subset with at least one in-link in the provided web graph. We take the first 1000 web domains/sites to construct the initial network and extend it to 10000 systems. Network clustering is performed using the method described in Section 5.3 to establish individual system neighborhoods. We use an observed optimal clustering exponent $\alpha = 2$ in the experiments.

6.3 Search Task - Rare Item Search

Given the size of the web (and likewise the ClueWeb09 collection), it is nearly impossible to manually judge the relevance of every document and establish a complete relevance base. Hence, we primarily rely on existing evidence in data to do automatic relevance judgment. We use documents (with title and content) as queries for decentralized searches. From the first 1000 web domains constructed above, we select as queries 12 random web pages with at least 3 in-links. The final set of query documents include (all trecids with prefix *clueweb09-en000*): 1-42-03978, 1-73-04287, 1-90-26216, 2-73-04700, 2-91-14776, 3-27-30577, 3-30-28328, 3-51-10345, 3-55-31539, 4-61-19060, 4-72-24215, 4-92-04942.

The search task is to find the exact copy of a given document (query). Specifically, when a query document is assigned to an agent, the task involves finding the site or author who created it and therefore hosts it. In other words, in order to satisfy a query, an agent must have the *exact* document in its local collection. The strength of this task is that relevance judgment is well established provided the relative objectiveness and unambiguity of creatorship or a "hosting" relationship. The extreme rarity will pose a great challenge on the proposed decentralized search methods.

6.4 Degree Distribution: $[d_m, d_x]$

We will use the degree (in-degree) distribution of the ClueWeb09B hyperlink graph and normalize the distribution to fall within a range $[d_m, d_x]$. With different d_m and d_x values, the degree distribution will continue to follow a power-law pattern in which the majority of nodes have the degree of d_m. We use various degree ranges $d_u \in [4, 8]$, $[16, 64]$, $[64, 128]$, and $[256, 512]$, to examine the impact of degree distribution (neighborhood size) on decentralized searches.

6.5 Maximum Search Path Length L_{max}

The maximum search path length L_{max} specifies the longest path each search (TTL) allowed for query traversal. If a search reaches the maximum value, even when the query has not been answered, the task will be terminated and returned to its originator. With our focus on search efficiency/scalability, we set L_{max} as the total number of systems N in experiments to achieve best possible effectiveness.

6.6 Evaluation

6.6.1 Effectiveness. We use a classic IR effectiveness metric based on precision and recall: $F_1 = 2PR/(P + R)$. Given that the task is to find an exact match for each (document) query, we only retrieve a document when an exact copy of the document is identified; otherwise, no document is retrieved. By definition, precision is 1 in either case whereas recall (and hence F_1) is either 0 (none retrieved) or 1 (1 retrieved and relevant) for a query. Reported F_1 scores are based on the average score (of 0s and 1s) of all queries.

6.6.2 Efficiency. For efficiency, actual search path length are recorded in experiments. The average search length of all tasks can therefore be calculated to measure efficiency: $\bar{L} = \sum_{i=1}^{N_q} L_i / N_q$, where L_i is the search path length of the i_{th} query and N_q the total number of queries. With shorter path lengths, the entire distributed system is considered more efficient given fewer systems/agents involved in searches.

6.6.3 Robustness. To understand overall system robustness, we simulate the number of agents that become unavailable and cannot provide routing services to help route any query. Combined with different system configurations, we measure search performance (efficiency in terms of search path length) and individual agent loads (the frequency of query visits). In the experiments, we randomly select a number of service agents (from 10% to 80% of the entire network) and make them unresponsive. However, we keep those particular agents that are relevant to search queries so that all searches will ultimately be able to identify relevant information.

6.7 Parameter Settings

The list below summarizes major variables discussed above. We use full combinations of these parameters in experiments, i.e., 1 (network size) × 2 (degree ranges) × 2 (search methods) × 5 (simulations of agent unavailability).

- Network sizes $N = 10,000$ and max search path lengths $L_{max} = N$;
- Degree ranges $d \in$: $[16, 64]$ and $[64, 128]$;
- Search methods: Similarity (SIM) search and similarity*degree (SimDeg) search.
- Fraction of unavailable agents: we vary the portion of unavailable agents from 0 (all available), to 10%, 20%, 40%, and 80%.

6.8 Hardwre and Software

Experiments are conducted on a Linux cluster of 5 PC nodes, each has Dual Intel Xeon E5620 4C (2.4 Ghz) Quad Core Processors (8 processors), 24 GB fully buffered system memory, and a REHL 6 installation. The nodes are connected internally through a dedicated 1Gb network switch. The Java Runtime Environment version for this study is OpenJDK 1.6.0_22.

7 RESULTS

Experimental results show that searches are efficient and scalable in large networks, especially with large neighborhood sizes (degrees). The distributed network of service agents shows a high degree of robustness. When a significant portion of agents become unavailable to provide routing services, search queries are able to take alternative paths to find relevant information.

7.1 Search Effectiveness

While our focus in this study is not on search effectiveness, we shall briefly report that SIM and SIM*DEG methods achieve perfect or nearly perfect effectiveness scores $F_1 = 1.0$. The superior effectiveness is due to the fact that maximum search path length (TTL) is set to be the total number of service agents in the network, allowing searches to traverse the network more thoroughly.

7.2 Search Efficiency

Figures 4 and 5 show the impact of agent unavailability (failure) on search efficiency where x denotes the percentage of unavailable service agents and y is the average search path length for all queries. Surprisingly with an increase in the number of failed or unavailable agents, searches become more efficient in the number of hops they have to traverse.

Closer examination reveals that when certain neighbor agents become unavailable, the other agents are able to find alternative routes to reach the target. When many agents are down and other agents are forced to change their connectivity, this in effect causes

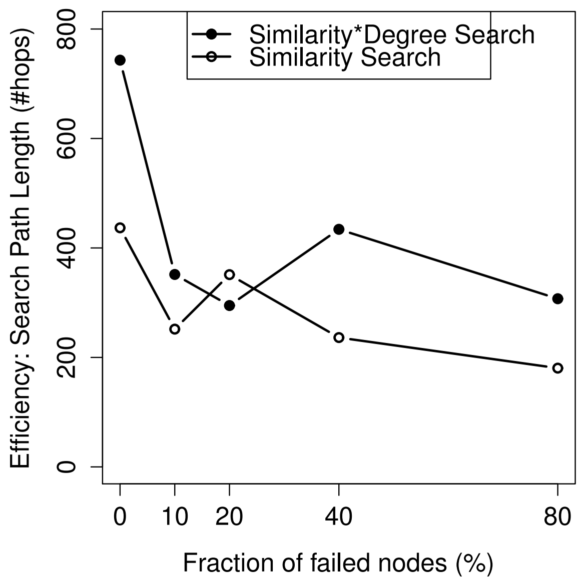

Figure 4: Search efficiency vs. percentage of unavailable service agents (with neighborhood size range [16, 32])

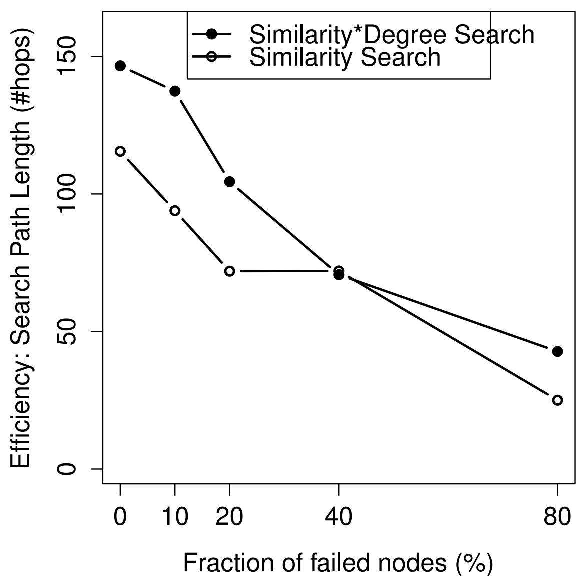

Figure 5: Search efficiency vs. percentage of unavailable service agents (with neighborhood size range [64, 128])

available agents to form a smaller network – a smaller community where searches can be done more quickly.

Note that in this study we assume the target agents with relevant information are always available and searches are always possible to reach their targets. The experiment is setup this way so that searches can be completed and measured. However in real networks there is no such guarantee and other factors should be considered as well.

7.3 Load Balance

In figures 6 7, we measure the maximum load (most highly loaded agents in search paths) and examine how this is impacted by other agents that failed. It appears that, except in the case of SIM search in the 16-32 neighborhood configuration, there is a inflecting pattern where the max agent load peaks at a certain unavailability level around 20%. Either decreasing the unavailability percentage or increasing it reduces the maximum load (of the most frequently contacted service agents). When the network is very stable (e.g. with zero unavailability), the overall system is healthy with a reduced load; when the network becomes more fragile, for those who remain in the network, they seem to be able to find a way to better balance their loads.

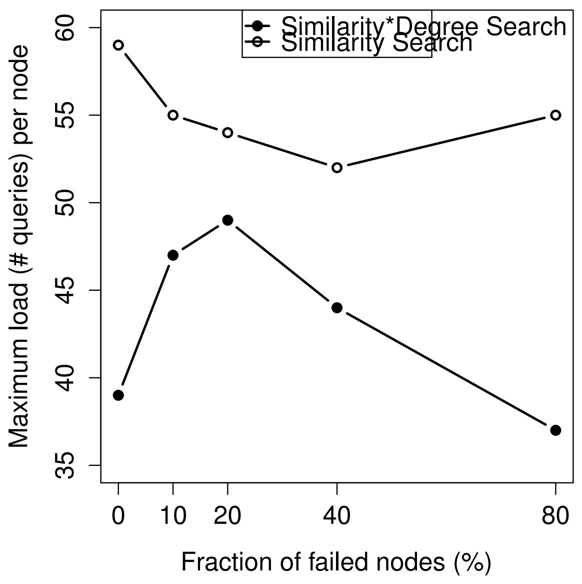

Figure 6: Max agent load (# queries per agent) vs. percentage of unavailable service agents (with neighborhood size range [16, 32])

In order to understand this and the load balance among service agents, we examine the gap (difference) between the load of the most highly loaded agent and the average. Figures 8 and 9 show the impact of unavailable agents on the gap load (i.e. max load - average load). From the figures, we continue to observe the inflecting pattern we discussed early, that the greatest gap (greatest difference between the most highly loaded and the average) occurs with a particular percentage of agent unavailability.

8 CONCLUSION

Our research studies decentralized searches in a large service-oriented agent network and investigate the influences of multiple factors on search efficiency. We focus on overall system robustness and examine search performance in unstable environments where individual agents may fail or a system-wide attack may occur.

Experimental results show that searches continue to be efficient when a large number of service agents become unavailable. Surprisingly, overall system performance in terms of a search path length

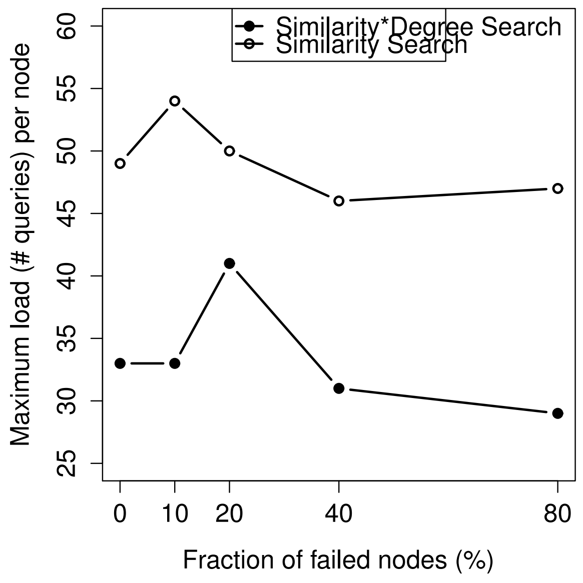

Figure 7: Max agent load (# queries per agent) vs. percentage of unavailable service agents (with neighborhood size range [64, 128])

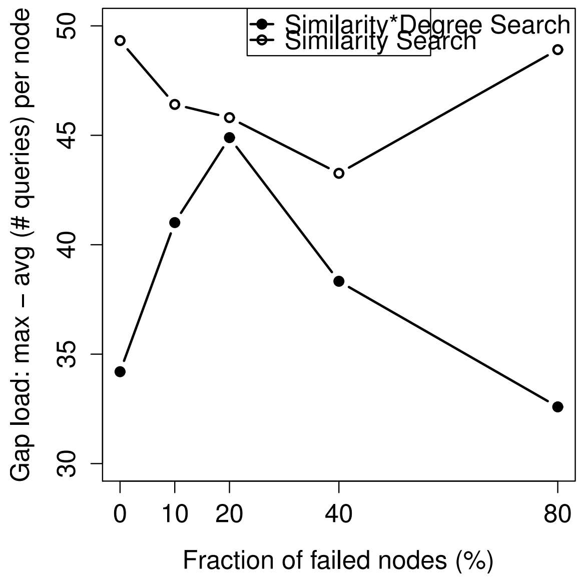

Figure 8: Gap agent load (# queries per agent) vs. percentage of unavailable service agents (with neighborhood size range [16, 32])

metric improves with an increasing number of unavailable agents. Analysis of agent load balance reveals a inflecting pattern where the greatest imbalance occurs at a certain level of agent unavailability. Interestingly, decreasing or increasing the number of unavailable agents leads to reduced imbalance (better load balance).

While our findings are to be confirmed in other settings, the result does suggests fault tolerance of distributed searches based

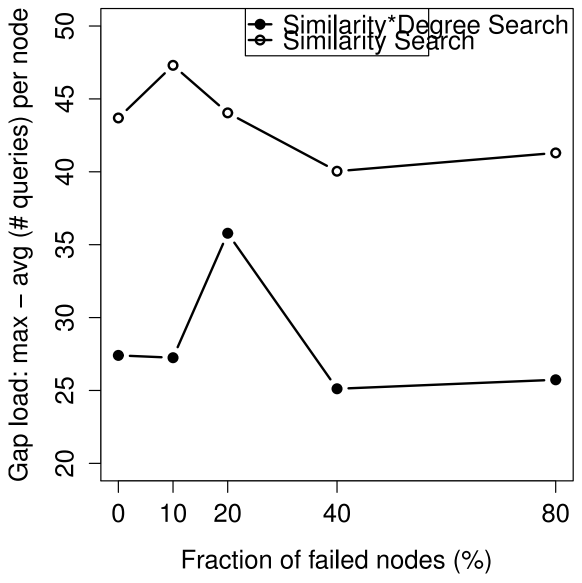

Figure 9: Gap agent load (# queries per agent) vs. percentage of unavailable service agents (with neighborhood size range [64, 128])

on service agents. It remains unclear why the inflection exists and how we should interpret it for the design of a service-oriented architecture for better load balance. We plan to conduct further research experiments to verify whether what we have observed here is something accidental in our specific experimental settings or of broader generalizability.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under grant no. 1646955.

REFERENCES

[1] Lada Adamic and Eytan Adar. 2005. How to search a social network. *Social Networks* 27, 3 (2005), 187 – 203. DOI:http://dx.doi.org/DOI:10.1016/j.socnet.2005.01.007

[2] Lada A. Adamic, Rajan M. Lukose, Amit R. Puniyani, and Bernardo A. Huberman. 2001. Search in power-law networks. *Physical Review E* 64, 4 (Sep 2001), 046135. DOI:http://dx.doi.org/10.1103/PhysRevE.64.046135

[3] Réka Albert and Albert-Lászlo Barabási. 2002. Statistical mechanics of complex networks. *Reviews of Modern Physics* 74, 1 (2002), 47–97. DOI:http://dx.doi.org/abs/cond-mat/0106096

[4] Réka Albert, Hawoong Jeong, and Albert-Laszlo Barabási. 1999. Internet: Diameter of the World-Wide Web. *Nature* 401, 6749 (Sep 1999), 130–131. http://dx.doi.org/10.1038/43601

[5] Javed A. Aslam and Mark Montague. 2001. Models for metasearch. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval.* ACM, New York, NY, USA, 276–284. DOI:http://dx.doi.org/10.1145/383952.384007

[6] R. Baeza-Yates, C. Castillo, F. Junqueira, V. Plachouras, and F. Silvestri. 2007. Challenges on Distributed Web Retrieval. *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on* (April 2007), 6–20. DOI:http://dx.doi.org/10.1109/ICDE.2007.367846

[7] R. Baeza-Yates and B. Ribeiro-Neto. 2004. *Modern Information Retrieval.* Addison Wesley Longman Publishing.

[8] Mayank Bawa, Gurmeet Singh Manku, and Prabhakar Raghavan. 2003. SETS: search enhanced by topic segmentation. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval.* ACM, New York, NY, USA, 306–313. DOI:http://dx.doi.org/10.

1145/860435.860491

[9] Fabio Luigi Bellifemine, Giovanni Caire, and Dominic Greenwood. 2007. *Developing Multi-Agent Systems with JADE (Wiley Series in Agent Technology)*. John Wiley & Sons.

[10] Matthias Bender, Sebastian Michel, Peter Triantafillou, Gerhard Weikum, and Christian Zimmer. 2005. Improving collection selection with overlap awareness in P2P search engines. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, New York, NY, USA, 67–74. DOI:http://dx.doi.org/10.1145/1076034.1076049

[11] Marián Boguñá, Dmitri Krioukov, and K. C. Claffy. 2009. Navigability of complex networks. *Nature Physics* 5, 1 (2009), 74 –80. DOI:http://dx.doi.org/10.1038/nphys1130

[12] Jamie Callan. 2000. *Advances in Information Retrieval*. Springer US, Chapter Distributed Information Retrieval, 127–150. http://dx.doi.org/10.1007/0-306-47019-5_5

[13] Jamie Callan and Margaret Connell. 2001. Query-based sampling of text databases. *ACM Transactions on Information Systems* 19, 2 (2001), 97–130. DOI:http://dx.doi.org/10.1145/382979.383040

[14] James P. Callan, Zhihong Lu, and W. Bruce Croft. 1995. Searching distributed collections with inference networks. In *SIGIR '95: Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, New York, NY, USA, 21–28. DOI:http://dx.doi.org/10.1145/215206.215328

[15] Charles L. A. Clarke, Nick Craswell, and Ian Soboroff. 2009. Overview of the TREC 2009 Web Track. In *Proc. of TREC-2009*. http://trec.nist.gov/pubs/trec18/papers/WEB09.OVERVIEW.pdf

[16] Arturo Crespo and Hector Garcia-Molina. 2005. Semantic Overlay Networks for P2P Systems, In Agents and Peer-to-Peer Computing. *Agents and Peer-to-Peer Computing* (2005), 1–13. http://dx.doi.org/10.1007/11574781_1

[17] Peter Sheridan Dodds, Roby Muhamad, and Duncan J. Watts. 2003. An Experimental Study of Search in Global Social Networks. *Science* 301, 5634 (2003), 827–829. DOI:http://dx.doi.org/10.1126/science.1081058 arXiv:http://www.sciencemag.org/cgi/reprint/301/5634/827.pdf

[18] Christos Doulkeridis, Kjetil Norvag, and Michalis Vazirgiannis. 2008. Peer-to-peer similarity search over widely distributed document collections. In *LSDS-IR '08: Proceeding of the 2008 ACM workshop on Large-Scale distributed systems for information retrieval*. ACM, New York, NY, USA, 35–42. DOI:http://dx.doi.org/10.1145/1458469.1458477

[19] Gary William Flake, Steve Lawrence, C. Lee Giles, and Frans M. Coetzee. 2002. Self-Organization and Identification of Web Communities. *IEEE Computer* 35, 3 (2002), 66–71.

[20] David Gibson, Jon Kleinberg, and Prabhakar Raghavan. 1998. Inferring Web communities from link topology. In *HYPERTEXT '98: Proceedings of the ninth ACM conference on Hypertext and hypermedia : links, objects, time and space—structure in hypermedia systems*. ACM, New York, NY, USA, 225–234. DOI:http://dx.doi.org/10.1145/276627.276652

[21] Mark S. Granovetter. 1973. The Strength of Weak Ties. *Amer. J. Sociology* 78, 6 (May 1973), 1360–1380. DOI:http://dx.doi.org/10.1086/225469

[22] Erik Hatcher, Otis Gospodnetić, , and Michael McCandless. 2010. *Lucene in Action* (second edition ed.). Manning Publications. 475 pages. DOI:http://dx.doi.org/1933988177

[23] David Hawking and Paul Thomas. 2005. Server selection methods in hybrid portal search. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, New York, NY, USA, 75–82. DOI:http://dx.doi.org/10.1145/1076034.1076050

[24] Nicholas R. Jennings. 2001. An agent-based approach for building complex software systems. *Commun. ACM* 44, 4 (2001), 35–41. DOI:http://dx.doi.org/10.1145/367211.367250

[25] Weimao Ke and Javed Mostafa. 2009. Strong Ties vs. Weak Ties: Studying the Clustering Paradox for Decentralized Search. In *Proceedings of the 7th Workshop on Large-Scale Distributed Systems for Information Retrieval, co-located with ACM SIGIR 2009*. Boston, USA, 49–56.

[26] Weimao Ke and Javed Mostafa. 2010. Scalability of findability: effective and efficient IR operations in large information networks. In *SIGIR'10: Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 74–81.

[27] Weimao Ke and Javed Mostafa. 2013. Studying the Clustering Paradox and Scalability of Search in Highly Distributed Environments. *ACM Transactions on Information Systems* 31, 2, Article 8 (May 2013), 36 pages. DOI:http://dx.doi.org/10.1145/2457465.2457468

[28] Weimao Ke and Javed Mostafa. 2016. Scalability Analysis of Distributed Search in Large Peer-to-peer Networks. In *2016 IEEE International Conference on Big Data (Big Data)*. Washington, DC, 909–914.

[29] Jon Kleinberg. 2006. Complex networks and decentralized search algorithms. In *In Proceedings of the International Congress of Mathematicians (ICM)*.

[30] Jon M. Kleinberg. 2000. Navigation in a small world. *Nature* 406, 6798 (August 2000). http://dx.doi.org/10.1038/35022643

[31] David Liben-Nowell, Jasmine Novak, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. 2005. Geographic routing in social networks. *Proceedings of the National Academy of Sciences of the United States of America* 102, 33 (2005), 11623–11628. DOI:http://dx.doi.org/10.1073/pnas.0503018102 arXiv:http://www.pnas.org/content/102/33/11623.full.pdf+html

[32] David Lillis, Fergus Toolan, Rem Collier, and John Dunnion. 2006. ProbFuse: a probabilistic approach to data fusion. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, New York, NY, USA, 139–146. DOI:http://dx.doi.org/10.1145/1148170.1148197

[33] Jie Lu. 2007. Full-text federated search in peer-to-peer networks. *SIGIR Forum* 41, 1 (2007), 121–121. DOI:http://dx.doi.org/10.1145/1273221.1273233

[34] Jie Lu and Jamie Callan. 2006. User modeling for full-text federated search in peer-to-peer networks. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, New York, NY, USA, 332–339. DOI:http://dx.doi.org/10.1145/1148170.1148229

[35] Eng Keong Lua, Jon Crowcroft, Marcelo Pias, Ravi Sharma, and Steven Lim. 2005. A Survey and Comparison of Peer-to-Peer Overlay Network Schemes. *IEEE Communications Surveys and Tutorials* 7 (2005), 72–93.

[36] Toan Luu, Fabius Klemm, Ivana Podnar, Martin Rajman, and Karl Aberer. 2006. ALVIS peers: a scalable full-text peer-to-peer retrieval engine. In *P2PIR '06: Proceedings of the international workshop on Information retrieval in peer-to-peer networks*. ACM, New York, NY, USA, 41–48. DOI:http://dx.doi.org/10.1145/1183579.1183588

[37] Filippo Menczer. 2004. Lexical and semantic clustering by web links. *Journal of the American Society for Information Science and Technology* 55, 14 (2004), 1261–1269. DOI:http://dx.doi.org/10.1002/asi.20081

[38] Stanley Milgram. 1967. Small-world Problem. *Psychology Today* 1, 1 (1967), 61–67.

[39] Allison L. Powell and James C. French. 2003. Comparing the performance of collection selection algorithms. *ACM Transactions on Information Systems* 21, 4 (2003), 412–456. DOI:http://dx.doi.org/10.1145/944012.944016

[40] Milad Shokouhi and Luo Si. 2011. Federated Search. *Foundations and Trends in Information Retrieval* 5, 1 (2011), 1–102.

[41] Milad Shokouhi and Justin Zobel. 2007. Federated text retrieval from uncooperative overlapped collections. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, New York, NY, USA, 495–502. DOI:http://dx.doi.org/10.1145/1277741.1277827

[42] Luo Si and Jamie Callan. 2003. Relevant document distribution estimation method for resource selection. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM, New York, NY, USA, 298–305. DOI:http://dx.doi.org/10.1145/860435.860490

[43] Luo Si and Jamie Callan. 2005. Modeling search engine effectiveness for federated search. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, New York, NY, USA, 83–90. DOI:http://dx.doi.org/10.1145/1076034.1076051

[44] Özgürand Simsek and David Jensen. 2008. Navigating networks by using homophily and degree. *Proceedings of the National Academy of Sciences* 105, 35 (2008), 12758–12762. DOI:http://dx.doi.org/10.1073/pnas.0800497105 arXiv:http://www.pnas.org/content/105/35/12758.full.pdf+html

[45] Munindar P. Singh, Bin Yu, and Mahadevan Venkatraman. 2001. Community-based service location. *Commun. ACM* 44, 4 (2001), 49–54. DOI:http://dx.doi.org/10.1145/367211.367255

[46] Gleb Skobeltsyn, Toan Luu, Ivana Podnar Zarko, Martin Rajman, and Karl Aberer. 2007. Web text retrieval with a P2P query-driven index. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, New York, NY, USA, 679–686. DOI:http://dx.doi.org/10.1145/1277741.1277857

[47] Duncan Watts. 2003. *Six Degrees: The Science of a Connected Age*. W.W. Norton, New York.

[48] Duncan J. Watts, Peter Sheridan Dodds, and M. E. J. Newman. 2002. Identity and Search in Social Networks. *Science* 296, 5571 (2002), 1302–1305. DOI:http://dx.doi.org/10.1126/science.1070120 arXiv:http://www.sciencemag.org/cgi/reprint/296/5571/1302.pdf

[49] Bin Yu and Munindar P. Singh. 2003. Searching social networks. In *AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multiagent systems*. ACM, New York, NY, USA, 65–72. DOI:http://dx.doi.org/10.1145/860575.860587