SoJa: Collaborative Reference Management Using A Decentralized Social Information System

Anwitaman Datta
School of Computer Engineering
Nanyang Technological University
Singapore
Email: anwitaman@ntu.edu.sg

***Abstract*—In this (invited) paper, we present a work in progress social library and reference management system called *SoJa* (Social Jabref), which is realized on top of a decentralized (peer-to-peer) social information system. The contribution of the work is multi-fold. It provides a platform to collaborate and socialize to carry out a specific task (managing and sharing bibliographic meta-information). From systems design perspective, it is an effort to realize social software on a peer-to-peer infrastructure, as well as make such a peer-to-peer system robust and reliable by leveraging on the social network. Particularly, we discuss how (we think) social networks can be leveraged to build reliable indexing, routing and storage services. We elaborate on the *SocialCircle* DHT which exclusively uses social links, and hence is expected to be naturally robust against various kinds of attacks. We also discuss several open challenges currently under investigation, which need to be addressed to build mature systems that can be deployed at large-scale. Furthermore, while not the principal focus of this specific work, the experiences in realizing SoJa are also directly relevant to the recent spate of work on realizing decentralized online social networks (DOSNs).**

Keywords: Social software, reference management, digital library, peer-to-peer (P2P), decentralized online social network

I. Introduction

The landscape of Internet usage has changed dramatically in recent years, both in the way the computers connected to the network interact as well as the way the end-users using these computers interact - with the Internet and with each other.

On the *networking plane*, infused by the (somewhat infamous) success of P2P file-swapping software, the last decade has witnessed an increased emphasis of using resources available at the edge to perform tasks which would otherwise have heavily burdened any centralized infrastructure. Thus to say, there is an increased proliferation of peer-to-peer mechanisms to either replace, or more often supplement, the client-server paradigm.

At the *application plane*, with the advent of Web 2.0 and social networks, we witness end users participating not only as passive consumers of content provided by the websites (client/server), but also as a contributor creating content collaboratively with fellow users. Thus at a logical level, many of these Web 2.0 applications are inherently peer-to-peer in nature.

Nevertheless, somewhat ironically, all current Web 2.0 applications rely on an underlying infrastructure based on the traditional client-server model. When the user interactions are peer-to-peer in nature, and while there is such a proliferation of unrelated P2P systems and applications, it is natural to ask if and how to realize a peer-to-peer underlying networking infrastructure for social and collaborative applications. End user privacy and autonomy from service providers, system scalability and reduced operational costs for service providers, and usability in ad-hoc and delay tolerant networking environments are some of the important motivations for realizing social software in quasi-decentralized (serverless) manner.

From an *application perspective* this paper describes a social software that allows users to maintain a personal digital library and manage references by annotating the content with reviews, rating or tagging the content based on their personal discretion and need. It is realized as a plug-in integrating collaborative and social networking features for a third party open source stand alone reference management software called JabRef [6]. We call it *SoJa* in brief to stand for social JabRef.

SoJa allows users to share their resources like papers' reviews or ratings with others in a selected manner, or collaborate in groups to create such content. Each user can maintain her own social contacts (buddy list), and can decide which of its local content should (or not) be shared with which specific buddies, or whether to make it accessible openly to the larger community. Users can also explore their buddies' social network subject to access rights granted by these buddies. These features are analogous to the very many online social networking sites, including several web based social libraries such as [3], [7], [24].

By focusing specifically on sharing scientific papers and personal reviews of such papers, SoJa serves as a collaboration tool. A group of researchers working on a project together can share their personal collection of research papers, or summaries of others' works, as well as collaboratively build a knowledge base for their project(s). Thus, purely from an application perspective, SoJa binds social networking and collaboration mechanisms together for a niche application, that of reference management.

From a *systems perspective*, by leveraging on a underlying peer-to-peer infrastructure, SoJa, though designed for a niche application, is another step to realize decentralized online social networks (DOSN). Several outstanding challenges to realize DOSNs had previously been identified [12]. Some of these have since been (partly) addressed by the research

COLLABORATECOM 2010, October 9-12, Chicago, USA

DOI 10.4108/icst.collaboratecom.2010.32

community as a whole (refer to [16] for a survey). In implementing SoJa, we borrow ideas developed in the community at large, but also carry out a few innovations which are expected to help the general research on DOSNs. SoJa thus acts as a vehicle which we use to innovate and validate ideas relevant to decentralized online social networks, and is closely related to sister projects carried out in the SANDS research group at NTU[1] on PeerSoN [13] which is aimed to build a general purpose DOSN,[2] and COBS [35], another niche DOSN application aimed to facilitate collaborative online browsing and search.

We provide an overview of SoJa's functionalities in Section II. While SoJa is expected to be a useful reference management application in its own right, we developed it as a vehicle to showcase the usability of decentralized information systems for social and collaborative applications. We provide a top-down overview of SoJa's implementation, exposing in Section III the enabling underlying peer-to-peer technologies to realize such decentralized social software in general. This includes how existing ideas such as that of a *self-referential directory* may be used to support log-in in a decentralized setting, which we describe in Section III-A, as well as novel ideas such a dichotomy of DHT-based global and social friends based local, decentralized storage as described in III-B, and a DHT realized using social links which we call *SocialCircle* introduced in Section III-C. We discuss related works in Section IV, comparing SoJa with other related reference management applications, as well as putting the enabling techniques in context with existing results from a decade of peer-to-peer research. We conclude in Section V.

SoJa is a work in progress, with some of the enabling ideas to realize a completely decentralized social software already implemented, while others still to undergo integration even if they have been separately tested, while some other issues that remain open. This paper tries to provide the details of the envisioned system architecture. A working and usable implementation of SoJa[3] is currently available. The current implementation is partially decentralized. It uses an auxiliary dedicated server, as shown in Figure 1. Such a dedicated infrastructure is vital particularly when the user community is small, since there may not be a critical mass to sustain 24/7 storage resource needs purely using end-user resources. Some other practical issues for large-scale deployment, such as NAT traversal and dealing with firewalls are also missing in the current implementation. SoJa has been tested to work within the NTU campus, and at a scale of the order of a dozen users. The current implementation is expected to adequately cater to usage needs within an individual organization in isolation, but is still a few steps away from being ready for a global scale deployment.

[1]http://sands.sce.ntu.edu.sg/
[2]In collaboration with Sonja Buchegger from KTH Sweden.
[3]http://code.google.com/p/socialjabref/

II. SoJa overview

SoJa has been implemented as a plug-in for JabRef [6], which is an open-source Java based reference manager and uses BibTeX as its native data format, and has functionalities to search and import data from several online bibliographic databases. This section provides an overview of the social networking and collaboration features supported by SoJa. The underlying enabling techniques are explained in the next section.

Figure 1 provides a high level overview of SoJa. A separate peer data management layer on top of the underlying social graph which determines the direct user-to-user connections is realized following the principle of network data independence [23]. The social network of end-users create the underlying basic overlay network, which also has support for Gnutella style search. Any sophisticated search algorithm for unstructured overlays may be used in this layer, but the current SoJa implementation uses a basic flooding based search. On top of such a social links induced unstructured overlay, we realize a DHT using exclusively social links. This non-trivial problem will be discussed in detail later in Section III-C, because traditional DHTs assume a fully connected underlying network, while a social network does not induce such a fully connected graph. The DHT is used for indexing and routing, as well as for self-referential directory [11] and global storage services. Without a critical mass of users it may not be possible to guarantee 24/7 storage, and hence an auxiliary dedicated server is additionally deployed, which also provides the same key-value store interfaces as the DHT.

We recognize that not all data to be stored and shared in a social networking collaborative application needs a global storage system. Instead, we leverage the fact that access to most such data is local in the social network, and hence the storage too can be confined to socially local nodes. Thus, we have a dichotomy of global (using DHT) and local (using friends) storage systems. We will elaborate later in Section III-B how and when each of these two different storage primitives are used in SoJa.

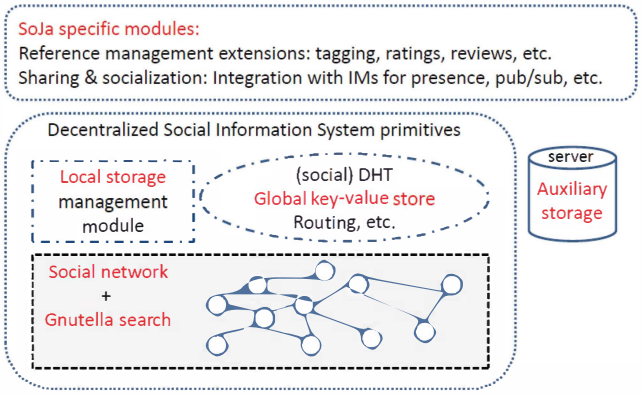

Fig. 1. SoJa architecture using a general purpose decentralized (p2p) social information systems back-end

SoJa has been designed to be minimally intrusive to the rest of JabRef. Thus, in terms of reference meta-information, we introduce only the additional features of a (i) review of a paper (which can also be used by a user to keep a personal note or summary of the paper), and a (ii) rating scheme (using 5-stars). JabRef's existing keywords attribute is used to tag content. We next describe the social networking and collaboration features of SoJa.

Sign-up and login: When a user first installs and starts SoJa, a private/public key is created, which is used to uniquely identify a node in the network. In subsequent sessions, the user thus continues to retain a persistent identity. Furthermore, copying this necessary public/private key information in another computer before starting an instance of SoJa allows portability.

A user can add buddies to her contact list by sharing this public identifier using out of channel mechanisms such as email, or posting on a (social networking) website.[4] SoJa provides support to use such out-of-band mechanisms from within SoJa itself, though this can easily be done by otherwise copy pasting the invite message and sharing it using any alternate user preferred channel. Such out of band invitation is useful to help a user bootstrap and get integrated with SoJa's social network layer.

Afterwards, when a user logs back in, the public key of buddies can be searched using SoJa's DHT network to reestablish connection. This is done by using an approach we call a self-referential directory [11]. We summarize the mechanism later in Section III-A. Users can thus determine the presence status of their buddies.

Navigation: A user can browse the profile and (shared) libraries of their buddies, as well as see the list of buddies of these buddies, and navigate the social network accordingly.

Search: A user can also carry out search for bibliographic items (for example, based on the name of authors, conference, part of paper title, tags, etc.), and specify to search over the social network using a Gnutella like flooding based approach using few hops, or initiate a global search using the DHT-based index. The rational for support of these two kinds of searches is as follows.

The Gnutella search based on the social network confines the search over a (multi-hop) egocentric view of the social network, and a web-of-trust can be established to determine the quality of results (particularly of reviews and ratings). A global search on the contrary is useful to see what potentially random people think about a paper, based on the availability of publicly available reviews. Global search is also useful to find new (presumably good) papers on a topic, which people in the immediate social network do not have in their local libraries either.

Inbox: Users can request to be notified about changes in specific items or a whole library of any of their buddies, or alternatively run a continuous query (subscription) to be notified about any information based on publication to the global index realized using a DHT. Alternatively, users can explicitly push specific content to other specific users. Such notifications are aggregated in an 'inbox' of SoJa. We describe later in Section III-A how the inbox is maintained for offline users, when we describe the log-in process.

Groups & collaboration: Users can organize the contact list of buddies into (intersecting) groups, and members of a group can collaborate to build together a shared library. Alternatively a user can share her review of an article with specific groups, or openly to the whole SoJa network in lines with public reviews; while other users can in turn rate these reviews - all enriching academic collaboration and enhancing shared knowledge either within closed groups as well as openly.

Access control: Individual users should be able to determine the individuals with whom they want to share each of its content. Users can define the granularity of sharing restricted to specific subsets of immediate friends or to the whole network. Users can also decide which of their social contacts should (not be) be visible to which of their other contacts. Users can accordingly search or browse other users' public profile and shared library. Note that since a user can locally decide which of her contact will (not be) visible to which other contacts, it is not only these other contacts from whom this information will stay secret, but in fact there is also no service provider who will be privy to such confidential information.

III. Decentralized social information system

A. Logging in, in a decentralized system

Traditional file-sharing systems do not have or even need a persistent peer identifier across sessions. Some systems use the IP address as the identifier. However, because of node mobility, dynamic IP address assignments, as well as need of portability across devices, IP address is unsuitable for providing a user ID for social networking applications.

A simple solution to this issue can be to use a logically centralized directory service storing the up-to-date peer-to-address mappings. However, given that peer-to-peer (DHT) networks themselves work as decentralized directory services, one can also imagine using the peer-to-peer network itself as a self-referential self-contained directory service to store meta-information about the participants, including their current physical address. This basic idea has been proposed independently (and varying in details) in several academic as well as commercial peer-to-peer networks, including P-Grid [11] structured overlay, Microsoft's Peer Name Resolution Protocol [25] and Skype [31]. Notice that the login service is thus supported using a global storage component, which we describe and distinguish from a complementary local storage component afterwards in Section III-B.

The basic idea is to use a self-referential directory [11] based on a DHT formed by the peers themselves. We assume that a peer's public key P_{pub} is known to its social contacts from previous interactions (exchanged using out-of-channel

[4]Integration with extrinsic instant messaging networks like xmpp is straightforward, and are being considered, but is missing in current implementation.

mechanisms such as email, IM, etc.). Whenever a peer returns online it inserts its latest address signed with its' private key in the DHT corresponding to a DHT key derived from a globally known hash of its public key. Likewise, any peer looking for a specific contact can search for the contact's public key, and discover its latest physical address.

A continuous query for the same can be left at the responsible DHT node, so that when a peer comes back online and reinserts its latest address, other peers interested in this peer (who have this peer in their list of buddies) are notified without them having to query again. This is essential to support presence.

The DHT is also used to store offline messages for the node, which it can retrieve when it comes back online. People sending offline messages need to encrypt it with the target's public key to preserve privacy, and store it corresponding to the hash of the target's public key. Consequently, once the peer logs back in, it can retrieve messages sent to it while it was offline.

Thus to summarize: when a peer comes online, it reinserts its latest address corresponding to its public key and signed with its private key, and also issues queries to locate the latest address of all nodes in its buddy list, as well as retrieve back any offline messages for itself. Furthermore nodes who have this peer in their buddy list are notified of its latest address once such an insertion is carried out.

Note that discovering the last address inserted by a buddy may however not sufficient, since the buddy may have in the meanwhile gone offline, and some other peer may be using the same address. Since buddies know each others' public keys, its easy to verify each other's identity once an address is found.

Notice that persistence of the necessary (inserted) information - such as the ID-to-IP mapping, or subscription information - in the underlying DHT, as well as other performance issues like load-balancing need to be taken care of at the DHT layer. Later in Section 2 we describe the design of a DHT, which, unlike traditional DHTs, does not require a fully connected underlying graph, and instead embeds the DHT on the social graph.

B. Storage

Since not all nodes are always online, providing persistent storage becomes important. Despite a decade long research on p2p storage systems, there is no perfect and ready to use solution for this problem. Noting that in social networking applications, there are many kind of data which are user specific and are of little interest beyond a user's ego-centric social network, while there is other information which can be of broader interest, we leverage on two different categories of peer-to-peer storage.

Socially local replication: User centric content such as profile information, as well as user generated content is replicated at a subset of the user's friends. Such a replication scheme conforms reasonably with access-control constraints because social contacts often form triads, and thus reduces the overheads of maintaining extra information (such as public keys) of the people who should have access to a specific object. The degree of replication is adapted dynamically to increase availability, while replicas of least recently accessed objects are garbage collected to manage with limited overall storage capacity in the system. The current choices are rather ad-hoc and we realize that some sort of optimization, taking into account the diversity in friends' uptime can be utilized to maximize the coverage. Replicas are synchronized using *google-diff-match-patch*.[5] Note that such a replica placement scheme is analogous to recent p2p back-up systems such as Friendstore [34] which also use only friends.

Such localized replication readily supports coarse grained access control as used in SoJa, where a user can decide to share her data with either her friends, or with the whole network. Note that a general purpose, finer grained access control, particularly where multiple writers may be involved, is neither needed nor addressed in our current work.

Global storage service: A global DHT is used to store offline messages (encrypted with recipients' public keys, and corresponding to the hash of such public keys). The DHT's native replication mechanism is used to ensure availability. The specific details of the SocialCircle DHT are provided in Section III-C. Similar to Wuala[6], a dedicated storage server is also used currently to augment the DHT, since in absence of a critical mass of users (and corresponding resources), an always available storage service simply cannot be realized. Such a design is also analogous to other DOSN architectures including PeerSon, but in contrast to those systems, our usage of the global storage is limited in nature, and the bulk of the storage load is socially localized, which is both easier to rationalize (in terms of dis/incentives, trust and enforcement) and also more efficient because of the localized nature of content access. The global storage service is also used to store the indices necessary to carry out queries over the network, as well as to realize a self-referential directory service which facilitates persistent user identity across multiple sessions, as summarized in Section III-A.

C. SocialCircle DHT

Another aspect of identity in decentralized settings is that users can create bogus identifiers. A major security threat in such systems is if a resource rich adversary creates many bogus identifiers - popularly known as Sybil attack [18] - then it can disrupt the functionalities of the system (denial-of-service), as well as more actively hurt the other genuine users. A practical approach to thwart Sybil attacks in decentralized systems is to exploit social relationships which exist between real people. We next explain how distributed hash tables may be embedded using only social links [36]. Alternative approaches of leveraging social links to mitigate Sybil attacks in DHTs have also recently been proposed.[7]

[5] http://code.google.com/p/google-diff-match-patch/

[6] www.wuala.com

[7] We note that besides Sybil attack resistance, use of social links may allow easier use of (web-of-)trust relations, or discourage free-riding, etc. We are yet to explore such additional benefits.

The ring topology is arguably the simplest and most popular structure used in various DHTs. In a ring based overlay network like Chord [32] nodes are assigned distinct points over a circular key-space, and the ring invariant is said to hold if each node correctly knows the currently online node which succeeds it (and the one which precedes it) in the ring. The ring is both a blessing and a curse. On the one hand, an intact ring is sufficient to guarantee correct routing. Hence, historically, all existing structured overlays over circular key space have considered it necessary de facto.

Previous attempts have used social network links to bolster DHTs, e.g., Sprout [27], preferring social links whenever possible, nevertheless, also requiring links to random nodes. Such an approach still relies on using the untrusted links most of the time, but was arguably as good as it could get under the older paradigm of DHT designs, where a completely connected underlying graph and ring invariance were considered necessary.

In the recent years several radical DHT designs have been proposed, for example VRR [14] proposed for ad-hoc environments and Fuzzynet [20] designed specifically to avoid ring maintenance. Neither of these two rely on sanctity of a ring or fully connected underlying graph. We design the SocialCircle DHT by adapting and hybridizing ideas from these two DHTs. Inlined in the description of SocialCircle below, we also point out which of the features are derived from which of VRR or Fuzzynet respectively.

Virtual ring routing (VRR) is a DHT style overlay layer approach used to define the underlying network's routing mechanism. It is implemented directly on top of the link layer and provides both traditional point-to-point network routing and DHT routing to the node responsible for a hashed key, without either flooding the network or using location dependent addresses. While traditional DHTs take for granted point-to-point communication between any pair of participating nodes, VRR extends the idea, using only link layer connectivity. Essentially this means that the VRR scheme relaxes the traditional DHT assumption of a completely connected underlying graph. Each node in VRR has an unique address and location independent fixed identifier, organized in a virtual ring, emulating Chord style network. Each node keeps a list of $r/2$ closest clockwise and counter-clockwise neighbors for the node on the virtual ring. Such a set of neighbors is called the node's virtual neighbor set (*vset*).

Typically, members in a node's *vset* won't be directly accessible to it through the link layer. Thus each node also maintains a second set called the physical neighbor set (*pset*), comprising nodes physically reachable to it through the link layer. In SocialCircle, we exploit this idea, and replace VRR's *pset* with the set of friends a node has - its social set *sset*.

Thus, instead of exploiting the physical layer connectivity as VRR does, in SocialCircle we try to build the overlay over the *social plane* exploiting people's social connections. In figure 2 the lower plane shows the social graph, while the upper plane shows the SocialCircle DHT. Adaption of VRR to exploit social links rather than physical neighbors provides

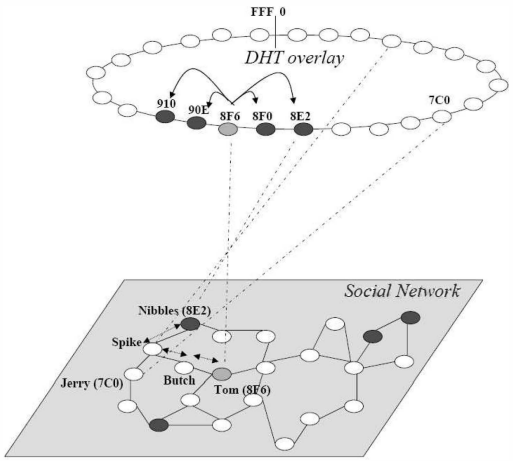

Fig. 2. Sybil attack resistant *SocialCircle* DHT exploiting social connections. This example of DHT over Tom & Jerry's social graph is adapted from the virtual ring figure in [14] for routing in ad-hoc networks.

a good abstraction, enabling us to realize a DHT where end-to-end routing can be achieved following a web or trust of friends-of-friends.

Finally, each peer maintains a routing table, which comprises of routes to its $vset$ neighbors using its $sset$. These routes can be established and maintained using different strategies typically inspired by mobile ad-hoc routing protocols. Like in VRR, nodes in SocialCircle also keep track of the routes that pass through them. The advantage of using the DHT abstraction to do the routing over social graph is same as the use of DHTs instead of using flooding based search in a typical peer-to-peer system. The DHT abstraction ensures efficiency and certainty of routing to the appropriate target.

Thus, in the example from Figure 2, Tom with logical identifier $8F6$ on the SocialCircle has $8F0$, $8E2$, $90E$ and 910 in its $vset$. $Spike$ has $Jerry$, $Nibbles$ and $Butch$ in its $sset$ since they are his direct social connections.

Tom needs to maintain routes to all its $vset$ nodes, and thus, for $8E2$, he will have a route through his $sset$ entry $Butch$, who will route through his *sset* entry $Spike$.

So when Tom needs to route a message to $7C0$, then it will try to forward the message closest to the target on the SocialCircle, which happens to be $8E2$. While the message is being routed to $8E2$ following the $sset$ nodes at each peer, $Spike$ will observe that the ultimate destination is $7C0$, for which it may already have a route passing through it, and will thus forward the message to $Jerry$, instead of sending it to the intermediate destination $Nibbles$. $Jerry$ processes the routing request, and forwards it to the final destination $Quacker$, who happens to have the identifier $7C0$ on SocialCircle.

VRR works on such an opportunistic manner, where the route is forwarded along the virtual ring, but discovers shortcuts, so that the search is still efficient. SocialCircle preserves the same benefits by routing over the social links. Each hop on the social link involves IP level routing, which may need several hops, just like any logical overlay hop of traditional

DHTs.

While the routing in SocialCircle follows the ideas from VRR, we use Fuzzynet's data-management ideas [20] for storing and retrieving key-value pairs in SocialCircle. In contrast to traditional DHTs where data is necessarily stored over the consecutive nodes on the ring identifier space, Fuzzynet routes a "write" request on the DHT to arrive as close to the target as it can, and then gossips using the nodes' links to store the data replicated at multiple nodes in the neighborhood, but not necessarily at the consecutive nodes. Lookup is done likewise. Such a non-deterministic (hence the name "fuzzy") placement basically achieves the benefits of structured overlays - efficient search and of unstructured overlays - resilience, and has proven to be resilient against churn in comparison to traditional DHTs by orders of magnitude.

Preliminary simulation based evaluation of SocialCircle [36] has shown the feasibility of such an overlay. A more thorough evaluation, including resistance to (Sybil) attacks, incorporation of web-of-trust mechanisms, etc. are important and interesting open issues in their own right, and are part of our ongoing investigation.

IV. Related works

In the discussion of works related to SoJa, we naturally need to discuss other works related to it as an end-application (collaborative reference management), as well as the enabling decentralized information systems back-end. A more focused discussion of related works follows. Additionally, in Table I we summarize how SoJa is placed with respect to representative related works from the domains of reference management and peer-to-peer systems, as well as how it compares with other applications such as web-based online social networks.

A. Digital libraries

The peer-to-peer paradigm has been proposed to distribute the workload of citeseer [33] or for archival storage of digital documents [26]. These systems do not have any social component in their realization.

Reference management: There are numerous reference management software[8]. At an application level SoJa differs from them primarily in the social and collaborative functionalities by supporting sharing and searching of references as well as personal comments or other meta-information related to these references. The collaboration capabilities of SoJa are on the lines of systems like Bibster [22]. Bibster supports such collaboration among end users by leveraging on implicitly formed semantic communities, but unlike SoJa lacks the notion of explicit (declarative) social networks. In that respect SoJa bears stronger similarity to web based social libraries [3], [7], [24].

Open peer review: GPeerReview [5] focuses specifically on sharing reviews publicly in the form of endorsements to form a web of trust based reputation of scientific publications. SoJa implicitly supports such functionality.

[8] http://en.wikipedia.org/wiki/Comparison_of_reference_management_software

B. Decentralized information systems back-end

Our main focus was to demonstrate the feasibility of social applications using a decentralized information systems infrastructure. Thus, the presented work composes together several ideas developed in the past decade of peer-to-peer research.

DHTs: Numerous DHTs have been proposed in the last decade. SocialCircle differs from most of them in that it uses only social connections to establish the DHT network, and is heavily motivated by recent innovations of DHT designs which do not require completely connected underlying graphs to establish a DHT [19], [14]. The closest existing work to our approach was a different approach where Sprout DHT [27] was realized with a mix of social links as well as links to other (not directly linked at social layer) nodes.

Social P2P systems: Recent years have witnessed strong interest in building decentralized online social networks (DOSNs). A more exhaustive survey of such initiatives can be found in [16]. To the best of our knowledge, existing DOSNs typically use either a stand-alone generic DHT (disentangled from the social relations of nodes) to provide a global storage service, for example this is the approach in PeerSoN [13] or LifeSocial [21], or alternatively use strictly localized storage at (friends-of-)friends, as in SafeBook [15] or FriendStore [34]. In SoJa, the advantages of each kind of placement policies, local as well as global, are leveraged. Furthermore, the deployed DHT itself is custom built and embedded in the social network.

Parts of SoJa resemble P2P storage based applications like the ePost P2P mail system [28] and the Friendstore [34] backup system, which both store user specific content - emails and backed up data respectively - by pooling resources in a peer-to-peer manner from the participating users. However, data in such applications is accessed by only the owner, and there are no application level collaboration or social interactions in these systems. Nevertheless these systems, particularly ePost, is a pre-existing proof of concept that a peer-to-peer infrastructure can be used for asynchronous communication among users. Support for off-line messages in SoJa is analogous.

While not a DOSN in the strict sense, Tribler [30] is another interesting related project. It bridges the gap between peer-to-peer video streaming and Web 2.0 applications like YouTube. Tribler uses social context in various ways including allowing users of similar tastes to form ad-hoc communities of *taste buddies* in order to enhance the chances of discovering content of common interests, as well as allowing users to cooperate and coordinate with friendly peers in sharing bandwidth which is used as currency in the system in order to enhance the performance of the download process itself.

DOSNs have a strong notion of identity of individual users and the social bonds among these users. These require persistence of identifiers across multiple sessions in an address independent way. Such strong notion of identifier is missing from traditional peer-to-peer systems. This is achieved using techniques from our previous work [11]. Other analogous

System	Application	P2P or Web based	Identity persistence	Address independence	Persistent storage	DHT or Unstructured	User generated	Search/Browse sharing	Collaboration	Recommend
SoJa	Ref. Mgmnt.	P2P (social)	Strong	Yes	Partial	Both	Yes	Yes	Yes	Not yet
Bibster [22]	Ref. Mgmnt.	P2P (random)	No	-	-	Unstructured	-	Yes	-	-
LibraryThing [7]	Ref. Mgmnt.	Web based	Strong	Yes	Yes	-	Yes	Yes	Yes	Yes
BibSonomy [24]	Ref. Mgmnt.	Web based	Strong	Yes	Yes	-	Yes	Yes	Yes	Yes
CiteULike [3]	Ref. Mgmnt.	Web based	Strong	Yes	Yes	-	Yes	Yes	Yes	Yes
CiteSeerx [2]	Ref. Mgmnt.	Web based	Strong	Yes	Yes	-	Yes	Yes	Yes	Yes
Edutella [29]	Ref. Mgmnt.	P2P/ Federation	No	-	-	Unstructured	Partly	Yes	-	-
OverCite [33]	Distributed Server & Crawler	Web based	-	-	Yes	DHT	-	-	-	-
LOCKSS [26]	Archival Storage	Distributed	-	-	-	-	-	-	-	-
Skype [31]	VoIP	P2P (both)	Strong	Yes	-	Unstructured	-	Partial	Partial	-
eMule [4]	File sharing	P2P (random)	Weak	-	No	Both	No	Yes	No	No
BitTorrent [1]	File sharing	P2P (random)	Weak	-	No	Tracker	No	Yes	No	No
Tribler [30]	Video sharing	P2P (both)	Weak	Not yet	Partial	Unstructured	Partial	Yes	No	Taste buddy
YouTube [10]	Video sharing	Web based	Strong	Yes	Yes	-	Yes	Yes	Yes	Yes
Orkut [9]	Social networking	Web based	Strong	Yes	Yes	-	Yes	Yes	Yes	-
Friendstore [34]	Backup	P2P (social)	Strong	Multiple machines	Yes	-	-	-	-	-
ePost [28]	Mail/Storage	P2P (arbitrary)	Strong	Yes	Yes	DHT	-	-	-	-

TABLE I

OVERVIEW OF AND COMPARISON OF SOJA WITH RELATED WORKS: VARIOUS KINDS OF POINT-TO-POINT RELATIONS MAY EXIST IN PEER-TO-PEER SYSTEMS. THE CONNECTIONS MAY BE WITH *random (arbitrary) peers* OR WITH *social friends*, OR ALTERNATIVELY HAVE *both kinds of links*. BY *user generated content* WE MEAN INFORMATION WHICH CAN UNIQUELY BE GENERATED BY THE END USER HERSELF, FOR INSTANCE RATINGS OR REVIEWS. THUS, UNIVERSALLY KNOWN META-INFORMATION ABOUT A CONTENT, EVEN IF FILLED IN BY AN END USER, IS NOT CONSIDERED. WE USE '-' FOR ATTRIBUTES THAT ARE *not applicable*. IF ONE COMPARES SOJA'S FEATURES WITH A SOCIAL NETWORKING WEBSITE LIKE ORKUT OR LIBRARYTHING'S FEATURES, IT CAN BE SEEN THAT THERE IS A STRONG MATCH FOR THE ASPECTS IMPORTANT FOR SOCIAL NETWORKING, SUCH AS: USER GENERATED DATA, SEARCH, BROWSE, COLLABORATE, AS WELL AS KIND OF LINKS EACH USER HAS.

approaches from Microsoft [25] and Skype [31] also exist. The Tribler website[9] indicates that similar mechanism using unstructured search is currently under investigation.

V. Future work and conclusion

Given the strong sense of identity in both the application layer at SoJa and the underlying P2P infrastructure SocialCircle, other social mechanisms like reputation can be used to enforce or judge the contribution of peers in the P2P infrastructure resources, as well as the quality of the content contributed by the users. Thus at the networking layer, malicious behaviors like free riding, etc. can be thwarted, making the system robust. Likewise at the application layer, users have an incentive to establish credibility by providing good content, helping build sustainable communities and knowledge base, and collaboratively filtering spam or spurious content. Such mechanisms are part of our future work.

Optimal utilization of local storage resources available at friends is another interesting open question, particularly when the global impact (load-balance, fairness) of such local decisions are taken into consideration. Support for finer granularity of access control, particularly for mutable content is another interesting open challenge.

Evaluation of the individual components, such as the SocialCircle DHT, as well as performance of continuous queries (pub/sub) on such a DHT are open questions currently under investigation. Practical implementation issues such as NAT traversal to make SoJa usable out in the open in large-scale deployment also need attention.

Besides the outstanding systems research issues, there are several interesting features that can be incorporated in SoJa. This includes recommendation of relevant content to users, for instance on the lines of taste buddies as used in Tribler, or alternatively using context aware selective gossiping mechanisms [17]. Finally, while SoJa currently provides rich features for carrying out collaboration, it is desirable to be able to use SoJa's social network to explore and find such potential collaboration partners/experts.

The current work demonstrates the feasibility of deploying social networking applications on a decentralized social information system infrastructure. We argue that such a design is both natural as well as essential to meet privacy and data ownership needs of individuals in the era of online social networks. The presented application facilitates sharing and collaborative management of bibliographic reference, and is expected to serve as an useful tool at workplace.

Acknowledgement

This research is (in part) supported by A*Star SERC Grant 072 134 0055 for the mTeam project [8]. The author will like to thank Hoon Thien Rong and Adrian Iskandar, who implemented major components of SoJa as parts of their respective final year undergraduate projects, and thank Krzysztof Rzadca and Jackson Tan for their help in mentoring these undergraduate students. Finally, the author will like to thank Krzysztof Rzadca and Sonja Buchegger for their feedback and help in improving the manuscript's presentation.

[9]http://www.tribler.org/trac/wiki/SocialOverlay

References

[1] BitTorrent. http://www.bittorrent.com/.
[2] CiteSeerx: Scientific Literature Digital Library and Search Engine. http://citeseerx.ist.psu.edu/.
[3] CiteULike: Everyone's library. http://www.citeulike.org/.
[4] eMule project homepage. http://www.emule-project.net.
[5] GPeerReview. http://code.google.com/p/gpeerreview/.
[6] JabRef reference manager. http://jabref.sourceforge.net/.
[7] LibraryThing: Catalog your books online. http://www.librarything.com/.
[8] mTeam: A Creative Environment for Mobile Knowledge Workers. http://sands.sce.ntu.edu.sg/mTeam/.
[9] Orkut. http://www.orkut.com/.
[10] YouTube. http://www.youtube.com/.
[11] K. Aberer, A. Datta, and M. Hauswirth. Efficient, self-contained handling of identity in peer-to-peer systems. *IEEE Transactions on Knowledge and Data Engineering*, 16(7):858–869, July 2004.
[12] S. Buchegger and A. Datta. A case for P2P infrastructure for social networks - opportunities and challenges. In *The Sixth International Conference on Wireless On-demand Network Systems and Services (IFIP/IEEE WONS 2009) special session on 'Social Networks'*, 2009.
[13] S. Buchegger, D. Schiöberg, L-H. Vu, and A. Datta. PeerSoN: P2P social networking - early experiences and insights. In *Proceedings of the Second ACM Workshop on Social Network Systems Social Network Systems 2009, co-located with Eurosys 2009*.
[14] M. Caesar, M. Castro, E.B. Nightingale, G. O'Shea, and A. Rowstron. Virtual ring routing: network routing inspired by dhts. In *SIGCOMM, Proceedings*, 2006.
[15] L. A. Cutillo, R. Molva, and T. Strufe. Safebook: A privacy-preserving online social network leveraging on real-life trust. In *IEEE Communications Magazine*, 2009.
[16] A. Datta, S. Buchegger, L-H Vu, T. Strufe, and K. Rzadca. Decentralized online social networks. In Borko Furht, editor, *Handbook of Social Network Technologies and Applications*. Springer, 2010.
[17] A. Datta and R. Sharma. GoDisco: Selective Gossip based Dissemination of Information in Social Community based Overlays. Technical report, NTU Singapore, 2010.
[18] J.R. Douceur. The sybil attack. In *Peer-To-Peer Systems: First International Workshop, IPTPS, Revised Papers*. Springer, 2002.
[19] S. Girdzijauskas, W. Galuba, V. Darlagiannis, A. Datta, and K. Aberer. Fuzzynet: Zero-maintenance Ringless Overlay. Technical report, 2008.
[20] S. Girdzijauskas, W. Galuba, V. Darlagiannis, A. Datta, and K. Aberer. Fuzzynet: Ringless routing in a ring-like structured overlay. *Peer-to-Peer Networking and Applications Journal*, 2010.
[21] K. Graffi, S. Podrajanski, P. Mukherjee, A. Kovacevic, and R. Steinmetz. A distributed platform for multimedia communities. In *IEEE International Symposium on Multimedia*, 2008.
[22] P. Haase, J. Broekstra, M.Ehrig, M. Menken, P.Mika, M. Plechawski, P. Pyszlak, B. Schnizler, R. Siebes, S. Staab, and C. Tempich. Bibster - A semantics-based bibliographic peer-to-peer system. In *ISWC 2004*.
[23] J. M. Hellerstein. Toward network data independence. *SIGMOD Rec.*, 32(3), 2003.
[24] A. Hotho, R. Jschke, C. Schmitz, and G. Stumme. BibSonomy: A social bookmark and publication sharing system. In *Proceedings of the Conceptual Structures Tool Interoperability Workshop at the 14th International Conference on Conceptual Structures*, 2006.
[25] C. Huitema and J.L. Miller. Peer-to-peer name resolution protocol (PNRP) and multilevel cache for use therewith. United States Patent 7,065,587.
[26] P. Maniatis, M. Roussopoulos, T.J. Giuli, D.S.H. Rosenthal, M. Baker, and Y. Muliadi. LOCKSS: A Peer-to-Peer Digital Preservation System. *ACM Transactions on Computer Systems (TOCS)*, 2005.
[27] S. Marti, P. Ganesan, and H. Garcia-Molina. Dht routing using social links. In *The 3rd International Workshop on Peer-to-Peer Systems*. Springer, 2004.
[28] A. Mislove, A. Post, A. Haeberlen, and P. Druschel. Experiences in building and operating epost, a reliable peer-to-peer application. In *EuroSys '06: Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006*, 2006.

[29] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmer, and T. Risch. Edutella: A p2p networking infrastructure based on rdf. In *WWW 2002*.

[30] J. A. Pouwelse, P. Garbacki, J.Wang, A. Bakker, J. Yang, A. Iosup, D. H. J. Epema, O. M. Reinders, M. R. van Steen, and H. J. Sips1a. Tribler: a social-based peer-to-peer system. *Concurrency and Computatation: Practice and Experience*.

[31] Skype.com. Skype P2P telephony explained, 2004. http://www.skype.com/intl/en/download/explained.html.

[32] I. Stoica, R. Morris, D. Liben-Nowell, DR Karger, MF Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *Networking, IEEE/ACM Transactions on*, 11(1):17–32, 2003.

[33] J. Stribling, J. Li, I.G. Council, M. F. Kaashoek, and R. Morris. OverCite: A distributed, cooperative CiteSeer. In *3rd Symposium on Networked System Design and Implementation (NSDI'06)*, 2006.

[34] D. N. Tran, F. Chiang, and J. Li. Friendstore: Cooperative online backup using trusted nodes. In *SocialNets '08: Proceedings of the 1st workshop on Social network systems*, 2008.

[35] C. von der Weth, A. Datta, and S. Ang. COBS: A Tool for Collaborative Browsing and Search on the Web. In *IEEE International Conference on Multimedia & Expo (ICME 2010) Demo*. http://code.google.com/p/socialcobs/.

[36] L. Zaczek and A. Datta. Mapping social networks into p2p directory service. In *SocInfo 2009, International Conference on Social Informatics*.