A Social Content Delivery Network for e-Science

Kyle Chard1, Simon Caton2,3, Kai Kugler2, Omer Rana4, and Daniel S. Katz1,5

1 Computation Institute, University of Chicago & Argonne National Laboratory, Chicago, IL, USA
2 Karlsruhe Service Research Institute, Karlsruhe Institute of Technology, Germany
3 National College of Ireland, Dublin, Ireland
4 School of Computer Science and Informatics, Cardiff University, UK
5 National Center for Supercomputing Applications, University of Illinois Urbana-Champaign, Urbana, IL, USA

Abstract

We are in the midst of a scientific data explosion in which the rate of data growth is rapidly increasing. While large scale research projects have developed sophisticated data distribution networks to share their data with researchers globally, there is no such support for the many millions of research projects generating data of interest to much smaller audiences (as exemplified by the long tail scientist). In data-oriented research every aspect of the research process is influenced by data access. However, sharing and accessing data efficiently as well as lowering access barriers is difficult. In the absence of dedicated large scale storage, many have noted that there is enormous storage capacity available via connected peers, none more so than the storage resources of many research groups. With widespread usage of the CDN model for disseminating web content, we believe a similar model can be applied to distributing, sharing and accessing long tail research data in an e-Science context. We describe the vision and architecture of a Social Content Delivery Network (S-CDN) – a model that leverages the social networks of researchers to automatically share and replicate data on peers' resources based upon shared interests and trust. Using this model, we describe a simulator and investigate how aspects such as user activity, geographic distribution, trust and replica selection algorithms affect data access and storage performance. From these results we show that socially-informed replication strategies are comparable with more general strategies in terms of availability, and outperform them in terms of spatial efficiency.

1. Introduction

The effective and meaningful sharing of scientific data has long been a central challenge in collaborative research. Its complexity stems from data attributes such as size, format, and version; computational constraints with regard to how and where data may be stored; economic constraints such as budget and willingness to pay for shared (storage) resources; and social issues such as data ownership and user willingness to share data. Nowadays, researchers have many potential storage options available to them (e.g., local, institutional, and cloud storage); however, their data storage needs often exceed available capacity and providing reliable and accessible storage incurs significant cost. Moreover, as data is acquired, curated, studied, and analyzed in different

places by different researchers, efficient data sharing methods are required to ease the burden of collaboration. This requirement has led to a large number of commercial solutions for data sharing, such as Dropbox, Copy, and SugarSync amongst others. Such approaches are generally efficient and easy to use and set up. However, they do not fully take into account trust issues between participants sharing data. Many such systems rely on one individual (the data owner) identifying who can view/update data.

We propose a model that leverages the social structures inherent within scientific communities to enable data sharing in a socially-derived trusted context, thus addressing some of the challenges associated with research data sharing and distribution. In most scientific collaborations there are different dissemination levels that could be associated with the (perceived) maturity of a dataset as well as aspects of the community in which it will be shared. To address this need we propose to base sharing and distribution upon the nuances of a (social) scientific network, allowing users to select and evolve differentiated dissemination levels using socially-derived sharing policies. For example, researchers could first share data within a network, i.e., with colleagues, before doing so in a wider community.

Building on the Content Delivery Network (CDN) model that underlies many of the world's largest websites, we propose a Social CDN (S-CDN) for sharing and disseminating (research) data [1]. A S-CDN is a platform that facilitates data-based scientific exchanges. While it applies caching and replica placement techniques common to CDNs, it is differentiated by its use of relationships expressed as social networks to restrict data access, to determine replica placement, and as a means of managing use of user-contributed resources (similar in nature to the resource contribution model applied in volunteer computing projects [2]). The S-CDN enables scalable and efficient distribution of data using a pool of user-contributed storage resources. The S-CDN model builds on the concept of *Data Followers*. Like Twitter followers, these are users who follow actions (e.g., data sharing or use) of a given user. Unlike Twitter, users curate their follower lists, explicitly deciding who has access to their data. This is an important distinction as it explicitly expresses an element of trust in the follower. To support data sharing, users contribute storage resources for hosting datasets, replicating shared datasets, and forming the S-CDN backbone.

In this paper, which is an extension of our previous work [1, 3, 4], we describe our S-CDN model, present our prototype implementation, and investigate via simulation how social sharing can improve data access performance, increase data availability and reduce data access latency via user-contributed infrastructure. We present and analyze socially-derived replication algorithms to show that a social sharing model is feasible and beneficial.

The remainder of this paper is structured as follows: In Section 2 we discuss scenarios and challenges related to social data sharing for collaborative research. Section 3 presents our architecture and implementation of a prototype S-CDN. Section 4 presents a simulator to facilitate the evaluation in Section 5. Section 6 discusses related work. Finally, Section 7 summarizes the paper, noting future work.

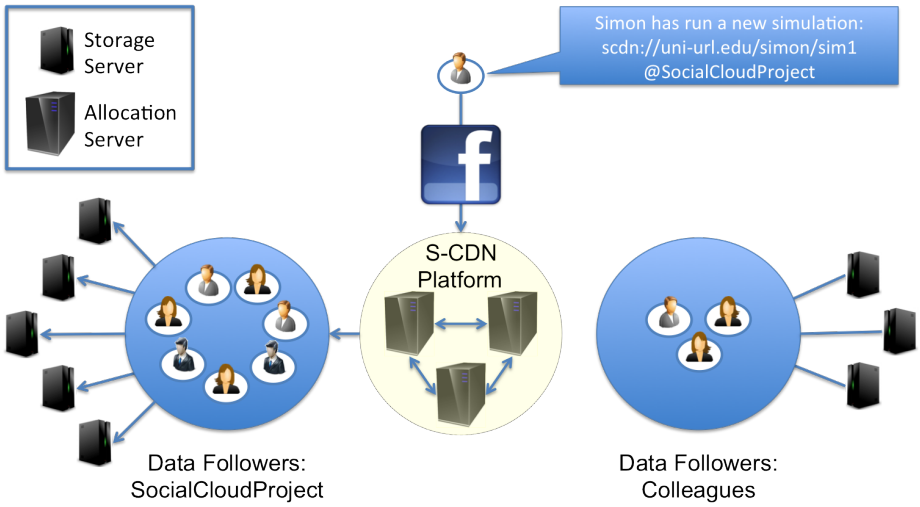


Figure 1: S-CDN Overview. Data followers receive updates when a user interacts with the S-CDN (e.g., sharing or accessing a dataset). S-CDN allocation servers use social network analysis to determine suitable replica placement.

2. Social Data Sharing

The concept of a S-CDN builds upon the idea of a Social Cloud [5]: "*a resource and service sharing framework utilizing relationships established between members of a social network*." In a Social Cloud, users connect with their peers to exchange resources and offer access to these resources in a cloud-like manner. Unlike a Social Cloud, in which users share resources directly with one another, a S-CDN is designed to transparently utilize contributed resources for the good of a community. A community could represent a particular research domain or research group, a single experiment, a funded proposal, or any other group in which its members are linked by an implicit or explicit social network. Thus, resource allocation (and therefore data replication) decisions are made by the S-CDN itself rather than by its users.

Figure 1 illustrates the high level S-CDN model. Here a user (Simon) has run an experiment as part of a collaborative project and he now wants to share the results with his collaborators. In a Social Cloud context this action would constitute a backup action using the user's peers' storage resources. However, in a S-CDN, the action is motivated by sharing and distributing data amongst collaborators. Figure 1 depicts four key aspects that differentiate the S-CDN use case from other forms of CDNs, Peer-to-peer (P2P) networks, file sharing, and collaboration approaches:

Data Follower Circles: A user has one or more sets of "Data Followers," denoted here as a Data Follower Circle. A data follower is a user who follows the actions of a given user. However, in contrast to Twitter and ResearchGate, a data follower must be authorized by a user before being able to follow that user (i.e., given access to that user's data). In some cases, policy approaches could be used as a proxy for explicit authorization, such as an existing digital relationship within a social network platform

and a given relationship type (e.g., colleague, close friend etc.), which can be partially represented as membership within a specified friend list (in Facebook) or circle (in Google+). It is important to note that in a S-CDN users do not automatically reciprocally follow their data followers. That is, a follower-followee edge is not bilateral, although the relationship must be authorized by both parties.

Data Status Update: To share data a user posts a status message (analogous to a Tweet) to the S-CDN. The status message includes a reference to the data to be shared. The S-CDN is then able to determine what data should be shared and with which data followers. The S-CDN's data management algorithms are then applied to distribute the data across the S-CDN. Similarly, when users access a dataset, status messages can be published to one's followers, thereby enabling social interactions around data usage.

Data Recommendations: The social basis of the S-CDN offers the potential to learn from other users' actions to provide recommendations for other users. In this way, the S-CDN could recommend datasets of interest or users to follow in the same way that ResearchGate recommends publications, LinkedIn recommends connections, and Amazon recommends products. Providing accurate recommendations requires learning from a large community of users' actions. However, there is a large body of work that could be built upon to construct S-CDN specific recommendation systems.

Resource Servers: A S-CDN is differentiated from a traditional CDN at its resource layer. In a S-CDN, the storage resources used for temporary storage and dissemination are located on the edge devices contributed by members of data follower circles. These resources are used both to share and access data, as well as providing storage for temporary replicas of other users' data within the S-CDN.

Allocation Servers: An integral part of the CDN platform is its overlay: the allocation servers, which perform all data management actions. These actions include instantiating data movement to/from followees and followers as back-ups, replicas, caches, and repositories. The infrastructure to support the platform may either be performed by a trusted third party, or through a co-operative collaboration infrastructure [6].

2.1. Social Sharing Challenges

To realize the seamless sharing of data, several challenges must be addressed:

Effective use of infrastructure resources: Data will not be shared universally, i.e., users will control who may access their data. Not only will this impact where data can be hosted, it also makes it difficult to accurately predict the potential popularity of datasets, and by extension where they (or their replicas) are best located in a socially-connected storage network. Where "best" is with respect to both potential users as well as other aspects of user activity, such as storage availability.

Accessing valuable social structures: To meaningfully use social structures, they first need to be provided by users. While in the era of platforms like Facebook these structures are always well represented and encouraging users to provide them is not always straightforward. Moreover, there are few social networks that model scientific collaboration, and those that do are sparse, incomplete and/or lack suitable APIs to retrieve collaborator networks. However, we expect that as research platforms like ResearchGate continue to grow, access and representation of collaborator networks will become increasingly useful.

Replication and quality of service: there are a myriad of replication methods that can manage and distribute data in a S-CDN; these choices affect availability, quality of service, and the ease of data access. Whilst replication strategies within traditional CDNs provide good quality of service with respect to availability and other quality factors, they may conflict with the idiosyncrasies and challenges of social data sharing. Here, one pressing issue is with respect to sharing preferences identified by a user. Whilst some users may be more open to sharing data and resources, others may be more frugal with their resources under certain sharing contexts. Such decisions may be grounded in the social standing of other users or other more nuanced socio-political issues of scientific collaboration.

Diverse (resource) availability: In leveraging user contributed resources for the S-CDN infrastructure, data availability is linked to the availability of users' resources. This is fundamentally different to a user's willingness to share, as noted above. Instead, we refer here to the physical accessibility and/or availability of a resource. For example, personal computers (laptops and desktops) will often have transient availability, while servers have continued availability. Similarly, personal computers are perhaps more convenient for smaller data, those currently needed for collaboration, or those that undergo frequent versioning. However, laptop availability will certainly change over time, for example when the owner is traveling, behind proxies or firewalls. In contrast, servers can be expected to have increased availability and accessibility. For a S-CDN and the replication of data, heterogeneous resource types present significant challenges with respect to replica placement strategies. Similar availability concerns also arise depending on the type of network infrastructure connecting the user resource – such as DSL vs. high speed network connectivity (each subject to bandwidth thresholds/throttling and other fair share allocation issues).

Economic factors and incentives: Irrespective of whether users are aware of it or not, sharing comes at a cost in terms of both power and time as well as effort. Consequently, users may need tangible incentives to foster participation, share data and provide resources. There is not necessarily an overlap between the incentives necessary to achieve all three. Similarly, not all users will react to the same incentive schemes in the same manner. For a more detailed discussion of incentives, their effectiveness and considerations, we refer to [6, 7].

Approximating and quantifying relationships: In designing an S-CDN, interpreting the underlying relationships is critical for key decisions, for example with respect to replica placement, group formation, data access etc. Ultimately, we leverage data from social networks (or proxies thereof) and previous interactions to approximate notions of trust. However, a S-CDN could also leverage other approaches such as user-based preferences for social ranking [8], trust derived from social interactions or other Granovetter [9] -like methods [10, 11, 12] or P2P systems [13, 14]. For a more in-depth discussion on trust within the context of a Social Cloud, and by extension a S-CDN, see [15].

2.2. Social Sharing Scenarios

Social sharing policies allow data owners to dynamically modify who can access individual datasets, with minimal effect on the underlying S-CDN. We can identify several sharing scenarios that span the spectrum from localized sharing to open data:

Localized sharing: sharing data within a highly localized (either geographically or homomorphically) (sub)network, for example, a team based at a single institution, or set of (geographically dispersed) researchers consistently working together. This sharing scenario is the most tight knit, with datasets shared between a small set of users. Here, we can imagine sharing policies individualized to datasets, collaborators and closed groups. In a localized sharing scenario, data related to early stage (low maturity) or unpublished work is likely to be common.

A **bounded community**: sharing data in a specific community that manages its membership, for example, a project consortium, where every community member is at least acquainted with every other member. In contrast to localized sharing, this community is bound by some underpinning collaboration agreement and/or funding mechanism. Membership of such a community is by invitation and data sharing is unilateral within the community, but not beyond. In a bounded community the sharing scenario is similar to that of a localized sharing setting. However, data related to (partially) completed work is likely to also be shared.

An **unbounded community**: sharing data in an open scientific community (after requiring users to explicitly join the community and subsequently passing through an authentication stage) with other members, such as occurs in nanoHUB.org [16] and myExperiment.org [17]. In this scenario, researchers may opt into the community to benefit from its wider reach and engagement from other researchers and stakeholders. In such a setting new collaborations may arise through the act of sharing data. In an unbounded community, we would expect data stemming from mature(r) research to be shared. Here, researchers may share data as a component of a publication in production in order to improve the publication's reach within the community (i.e. where data is not for public consumption but for initial feedback) or engage other researchers to further the initial work.

Open (access) data: sharing data openly, where users do not require permission to follow another user or access their data. Data is shared without any specific concern with respect to who will use it, or who is actually following the originator. We envisage only mature (potentially curated and sampled) data to be shared in this scenario probably in connection to some publication or other research dissemination activity.

Associated with these sharing scenarios there may also be issues of: (i) provenance – how data sources are derived from others; (ii) licensing issues – what licenses are associated with shared data – or do some data sources reference commercially provisioned data; (iii) attribution/credit issues – who owns which data and has the right to share it with others. These issues are outside the scope of this work.

3. Architecture and Implementation

The high-level architecture of the S-CDN and the process of registering and sharing data is shown in Figure 2. Our S-CDN implementation leverages Globus [18] to provide access to distributed storage and as a backbone for high performance data transfer between S-CDN nodes; a Django [19] based web application to store and manage users, determine replica placement, and allocate storage across the network; and social network adapters to extract relationships between users. The major interactions in a S-CDN are as follows:

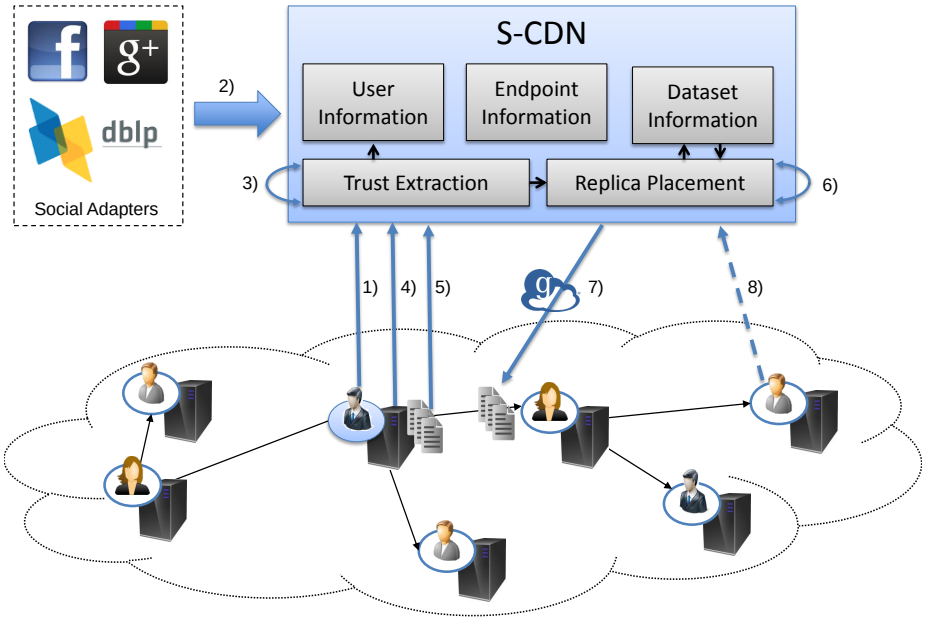


Figure 2: S-CDN usage workflow showing the interactions between users, the S-CDN and social adapters.

Signup and Login: When registering with the S-CDN, a user signs in using their social network identity and is asked to grant access to their friend lists (1). The S-CDN application extracts the user's profile information and social network(s) (2). Measures of trust are extracted from the network. For example, social network metrics, such as the strength of individual relationships are calculated (3). Steps (2) and (3) are repeated every time a user logs in. All information about users, their relationships, and social network metrics is stored in the S-CDN database.

Endpoint Registration: As a requirement of joining the S-CDN, every user must provide access to storage resources via a Globus endpoint: a logical mapping to a lightweight data access and transfer agent installed on a storage resource. The endpoint provides the means by which the S-CDN can access storage, methods to reference and access datasets, and an interface for transferring data between endpoints. Users must create a local account on their storage system, in order to establish a sandbox for data access. This local account is used by the S-CDN to access and transfer data on the user's behalf. Users register their endpoints with the S-CDN application (4). Endpoint information is stored in the S-CDN database.

Dataset Registration: Users may register datasets with the S-CDN by selecting a directory on their endpoint representing the dataset (5). A replica placement algorithm is applied to determine appropriate endpoints on which to replicate the dataset (6). Dataset information and replica locations are stored in the database. Globus transfer jobs are submitted to transfer (copy) the dataset to the selected replica endpoints (7).

Dataset Access Requests: Datasets may be accessed and downloaded (8) by searching for a dataset (via dataset and/or owner name) or via a unique URI. The S-CDN selects the most appropriate replica to service the request, and the dataset is transferred asynchronously using Globus to the user's endpoint.

S-CDN Activity: Given the underlying social theme of the S-CDN, an "activity" page shows the recent actions of friends, including registration of endpoints and datasets as well as downloads of datasets. This page allows followers to monitor the actions of their followees, and it also serves as a record of a user's actions within the S-CDN, providing a means for other users to also access and view *interesting* datasets that have been registered, used, or accessed by their friends.

3.1. Storage Repositories and Data Transfer

The S-CDN model is based upon a distributed data fabric composed of storage resources contributed by users. Upon joining the S-CDN, users register their storage resources. The data fabric then provides access to, and transfer between, registered storage resources using Globus. Globus is a provider of reliable and high performance data transfer, synchronization and sharing. Through its standard authentication, data access, and data management interfaces, users (and in this case the S-CDN) can securely manage data across a federated collection of storage endpoints.

S-CDN storage services hosted on participant resources are exposed as Globus endpoints. Globus endpoints are logical representations of servers that implement Globus' data access interfaces. Globus Connect, the software installed on local resources that implements Globus data access interfaces, can be used on a variety of storage resources including Mac, Linux, and Windows PCs as well as Linux-based servers. Globus endpoints can also be mapped to cloud object storage such as Amazon S3 buckets. Globus endpoints support authorized data access and asynchronous movement of data via GridFTP [20]. When transferring data between endpoints Globus implements a "fire-and-forget" model through which users can start a transfer asynchronously between two remote endpoints. Globus subsequently manages the transfer and notifies users on completion. Throughout the transfer, Globus ensures that bandwidth usage is optimized, file integrity is maintained, and users are notified of events that occur.

The typical Globus access model allows authenticated access to an endpoint based on local user accounts or pre-defined authentication providers. Globus' in-place data sharing model [21] allows data access to be delegated to other Globus users without requiring user accounts on the physical server. The S-CDN makes use of these data sharing capabilities to control contributed storage resources without requiring local accounts to be established. All S-CDN endpoints must be configured with sharing enabled and delegated management of a selected directory to a special S-CDN management (Globus) user ('scdnagent'). This allows the S-CDN to manage transfers to the shared local S-CDN directory, therefore enabling the asynchronous upload and download of datasets as well as the creation of temporary replicas on contributed resources.

While the S-CDN model may resemble a P2P network, there are many significant differences. First, access to the endpoints, and therefore data transfer, is possible from anywhere, not only from a node in the network. Second, replication is carried out by the S-CDN rather than by sharing of datasets that have been explicitly downloaded

by others in the network. Finally, S-CDN management is performed by a centralized service rather than by a decentralized overlay network.

3.2. S-CDN Allocation Server

Management of the S-CDN is provided by a centralized allocation server. The allocation server is responsible for managing registered users, datasets and replicas. It manages mappings between users, their social network identities, social network graphs, and datasets. It also orchestrates the transfer of, and access to, S-CDN data via Globus.

To provide reliability and availability, the allocation server is implemented as a service and can be hosted on cloud resources. All capabilities are exposed via a programmatic REST interface and a web-based user interface. The architecture follows a typical web approach, building upon a MySQL database and Python's Django framework deployed to an Apache HTTP sever. We currently operate a single instance of the allocation server on a virtual machine hosted at the University of Chicago. The allocation server is designed to be elastic and scalable, that is, the stateless allocation server can replicated for improved availability and performance. It also supports the use of an elastic and reliable cloud-based database, such as Amazon Relational Database Service (RDS). While this model represents a single point of failure, in practice highly available services can be operated using appropriate deployment and hosting models. For example Globus utilizes a similar deployment model and achieved 99.96% availability in 2013 [22]. Moreover, many such allocation servers can be hosted for different communities.

3.3. Social Adapters

To support the extraction of social networks and the ability to authenticate using a social network identity, we have developed a social adapter model through which different social networks can be integrated. These capabilities are based on Django's `social auth` module.

In our prototype implementation we use Facebook as the social network provider. We chose to use Facebook for several reasons: 1) it is the largest and most ubiquitous social network available; 2) it has advanced interaction features such as dynamic news feeds; and 3) it provides well documented APIs for retrieving social network structures. Other social networks can also be easily integrated into our model by providing the URLs used for OAuth-based authentication and ensuring that the APIs for accessing social connectivity are supported. For example, research social networks such as ResearchGate could be integrated. In addition, implicit social networks such as co-authorships extracted from a publication database can be used. In Section 4 we describe how co-authorship networks are used as a basis for establishing a scientific social network.

When using the Facebook social adapter, users first associate their social network identity with their S-CDN account using the OAuth protocol. This action creates a delegated access token that allows the S-CDN application to access various parts of the user's online profile. The S-CDN can the retrieve the user's profile information, their friends, and any friend lists through the Facebook Graph API. These lists are

compared against existing S-CDN users and any matching relationships are stored in the databases. Social information is periodically updated when users log into the S-CDN.

3.4. Data and State Storage

The S-CDN allocation server stores user data and session state in a MySQL database. We have developed data models that represent users, social networks, storage endpoints, datasets, replica locations, social relationships, and events in the S-CDN such as data registration and access:

User data: username, first name, last name, email, registration data, last login.
SocialAuth: social network type, social network id, oauth token.
Endpoint data: id, owner, endpoint name.
Datasets: id, owner, dataset name, endpoint, relative path.
Dataset versioning: owner, date, type <"endpoint", "registration", "update", "download">.
Dataset replicas: dataset id, owner, endpoint, relative path.
Download metrics: dataset id, downloader, download time.
Relationship metrics: Owner, follower, score.

3.5. Metadata Management

As the number of datasets available in the S-CDN grows methods are required for easily discovering and understanding shared data. Our current model supports storage of only basic file metadata (e.g., owner, name, and location), however the ability to store more semantically meaningful metadata will improve discovery and use. Such metadata is particularly important in scientific domains as metadata describes attributes such as the sample or specimen, the experimental or simulation conditions, and provenance or data lineage. It is this rich metadata that both facilitates understanding of data and also enables more flexible discovery. In related work, we have developed a scalable scientific metadata cataloging system [23, 24]. This catalog implements an efficient decomposed data model in which arbitrary typed key-value pairs of metadata can be associated with files or collections of files. The catalog stores this data in a relational database using a data model that is designed to efficiently store and query sparse scientific metadata. We envision that a similar model could be implemented in the S-CDN database to enable association of rich file, user, and scientific metadata with S-CDN datasets.

3.6. User Interfaces

The S-CDN web interface provides access to capabilities such as user and endpoint registration, dataset registration, definition of social sharing restrictions (e.g., users and groups), access to activity feeds, and the ability to download datasets. The interface is built on a lightweight HTML/CSS/JS model that uses AJAX to dynamically render and update pages. Requests are made directly to the allocation server using the REST API.

3.7. Calculating Social Network Metrics

Our S-CDN prototype includes functionality to extract social network metrics that can be used when making allocation and replica selection decisions. When users authenticate with the S-CDN we store profile and friend information from their social network. We also record users' actions with the S-CDN, such as datasets shared, datasets accessed, etc. These sources of information provide the ability to calculate metrics that may be used to improve S-CDN operations. For example, for each user that is linked in the S-CDN we calculate a weight for the relationship between the two as follows. We subsequently use this weight to determine replica placements.

$$RelationshipScore = Interactions * GroupFactor \quad (1)$$

Interactions: Represents the number of times two users have interacted with one another. The number of interactions between users can be used as a crude measure of relationship strength. As users are often reluctant to allow third party applications to access personal Facebook information we place a higher emphasis on interactions in the context of the S-CDN. For example, we track exchange reciprocity, i.e., how often user A downloads a dataset from B in relation to how often B downloads a dataset from A, and use this metric to represent interactions.

Group Factor: Represents user classifications of the strength of relationships based on social network groups. If users permit, we extract their social network groups, as these may indicate comparative social consideration. Here, we only consider groups that are predefined by Facebook, and assign a strength factor that reflects the relative strength of each group type (e.g., Acquaintances: 1, Family: 2, Close Friends: 3). While these groupings and their weightings are subjective they provide a simple model for users to characterize and group connections. Although this grouping may lead to some users ranking other users more highly than others this approach initially mitigates cold start issues common to recommender systems. In previous work we have explored a complex two-sided preference matching model in which user preferences with respect to their propensity or willingness to share with other users are applied when determining resource allocations [8]. These same approaches could be applied here.

3.8. Replica Selection

Replica selection is a crucial feature of the S-CDN model as it underlies the access and availability of data. There are a number of potential metrics that could be used to determine the 'best' node on which to place a replica and to select the 'best' replica from which to download a dataset. It is also possible to determine an optimal number of replicas, depending on the capacity and capability of the underlying infrastructure. Examples of potential metrics include traditional metrics such as the type of storage resource, the availability of that resource, the location of the resource, and the bandwidth and latency between resources. In a S-CDN setting, social relationships and trust can be used to select replicas by determining appropriate placements based on similarities between users. For example, similarity metrics could be calculated based on previous data sharing, access to similar datasets, or social network (follower-followee) proximity. We investigate various approaches for replica selection in Section 5.

At present, the S-CDN prototype places complete replicas of each dataset on randomly selected available endpoints of a dataset owner's friends (followers), where the replication factor (the number of nodes selected for replication) is a configurable system parameter. While this is a reasonable first approach, as in small communities only one replica of a desired dataset has to be online for a request to be satisfied, more sophisticated approaches that take account of infrastructure properties (e.g. bandwidth and network latency) may be used to determine number of replicas. These approaches are investigated in Section 5.

We do not yet consider the length of time that replicas are maintained nor do we consider reassignment of replicas after initial allocation. Both, would conceivably improve data availability in a S-CDN. We do however, track usage information, such as data registrations, data access, replica allocation, and replica selection. This information could be used in the future to determine which replicas should be moved to meet requirements for social, geographical or temporal distribution, when replicas should be removed if they are not used frequently, and which replicas should be removed to crate space for replication of new data. Such decisions would need to be made globally and consider the tradeoffs between storage space and availability while also considering dataset popularity.

4. Simulating a S-CDN

To investigate the performance of a S-CDN under different scenarios we have developed a Python-based S-CDN simulator. The simulator can be configured to model: 1) the social network; 2) various social metrics; 3) user contributed resources and activity; 4) simulation workload; 5) geographic distribution; 6) data access permissions (based on trust); and 7) replica selection algorithms. These aspects of the simulator are explained further in this section.

4.1. Social Network

The simulator is designed to model arbitrary social networks, which may be randomly generated or derived from an existing network. The simulator could be configured to use a social network extracted from Facebook (or any other online social network) or an implicit network extracted from other sources such as a co-authorship network. Due to privacy concerns and lack of access to large Facebook networks we instead focus on research social networks by using the DBLP [25] database as a proxy for scientific collaboration. DBLP is an online database that includes over 2.3 million computer science articles published over the last 30 years. The database relates authors to one another via co-authorship, thus forming a network of authors with shared publications.

To initialize the simulator with the DBLP graph, we construct a user graph by parsing the DBLP database for co-authored publications over a period of time. This graph centers around an initial node (seed author) and creates a subgraph of configurable depth. Every author is represented as a node and every co-authorship is represented as an edge between nodes. Each co-authored paper is considered an interaction and can be used to bootstrap social metrics between co-authors (e.g., frequency of collaboration). The simulator includes methods to further prune the social network based

on the extraction of trust metrics. For instance, social networks can be pruned based on the strength of relationships by placing a limit on the number of edges (in DBLP, co-authorship) between nodes.

4.2. Calculation of Social Metrics

The simulator is able to apply weightings to edges in the graph based on user-defined factors. These weightings can be subsequently used when making replica placement decisions and access decisions. One example of a common weighting aims to quantify the relationship strength between two users. In the prototype S-CDN this measure is based on a relationship score. In the simulator we have implemented similar metrics by counting the number of edges (in the DBLP graph, co-authorships) between nodes over a given period of time.

4.3. User Contributed Resources and User Activity

Modeling user contributed resources (in terms of their availability) as well as user activity (with respect to sharing data and requesting shared data) has a significant impact on system behavior. Resource availability is central to determining how many replicas are available, and it is dependent on the types of resources users commit to the S-CDN. Some users—especially in scientific communities—will have servers available that they use to decouple the availability of their data from their own activity. Others will use personal computers that may be shut down when they are not in use.

To investigate the impact of different resources, the simulator models two types: those that shut down (*laptops*) and those that do not (*servers*). Modeling user patterns of potential S-CDN users is complicated. However, there is significant research related to the working patters of academics [26, 27]. We use the data outlined by Begole et al. [26] to model user behavior over the course of one working day (see Figure 3). This provides an activity function that determines the probability with which a user is online at a certain time of the day. To assess the sensitivity of user activity, we have developed two activity functions based on this model: *normal* and *reduced*, a more pessimistic variant. For simplicity, we assume laptop availability is identical to user activity. In order to model availability, the simulator is designed to operate in time steps (e.g., hourly). Individual simulation events (registration, data access, etc.) occur periodically in accordance with the configured time step.

4.4. Simulation Workload

The simulator is designed to model arbitrary workloads representing individual data registration and access requests at particular times. Again, we have used the DBLP database to generate sample workloads for the simulator. Each new publication is interpreted as the registration of a dataset in the S-CDN by the first author. Registration starts by determining a random hour of the day in accordance to the user's activity distribution. After this time, a co-author of that publication is chosen at random to follow the first author and requests the dataset according to their activity distribution. If an endpoint that hosts either a replica or the original dataset is available, the download commences, otherwise the simulator will retry later. Note: if the requesting endpoint already hosts a replica of the dataset, this replica is ignored.

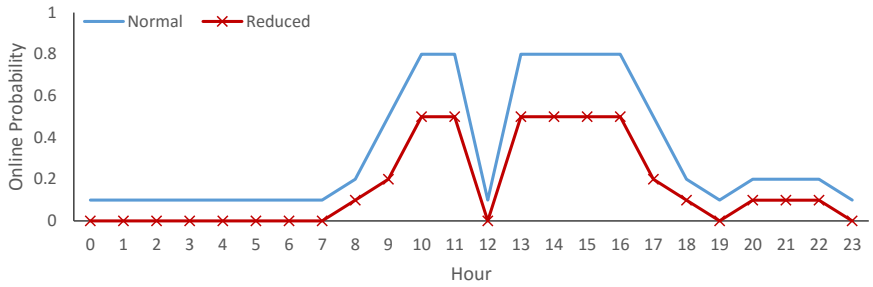


Figure 3: Probability distribution for user activity over the day.

4.5. Geographic Distribution

Geographic distribution of S-CDN users affects temporal shifts in user activity, for example, a user in the United States may have a similar working pattern to a user in Europe, but offset by several time zones. In other words, where Figure 3 defines a probability distribution for user availability, geographic distribution creates offsets between the probability distributions of users. To capture this notion, the simulator is able to assign users to time zones and then to subsequently use this information when determining resource availability based on the availability distributions discussed above.

Unfortunately, time zones cannot be easily inferred from social networks or DBLP. In our DBLP dataset most entries do not provide a location. While there are 26 time zones worldwide, for simplicity we reduce these to six: -8, -4, 0, +4, +8 and +12. In assigning users to time zones, we can either assume a randomized distribution across all time zones, or (more likely) assume that (sub)communities are co-located, with some probability. The simulator can be configured with either strategy. In Section 5 we chose the latter, adjusting user time zones to the one that most of their peers (based on co-authorship graphs) have with a probability of 0.7[1]. For an example, see Figure 4, which is based on the authors' co-authorship network. It reflects a network where sub-communities mostly but not completely share time zones.

4.6. Data Access

An important capability provided by a S-CDN is the ability to share data with a restricted group of socially connected individuals. The S-CDN enables the overlay of various socially-derived sharing polices to coordinate authorized access to datasets (and replicas). To manage replica access via inferred trust, we consider three sharing policies: 1) **unrestricted:** allows any user access to an endpoint, 2) **restricted:** prevents users from accessing a replica placed on an endpoint not owned by an immediate follower, and 3) **abstracted:** permits users to access a replica if the associated dataset is owned by a follower or someone they follow.

[1]However, we note that this parameter can be tuned to vary the probability of co-located (sub-)communities forming.

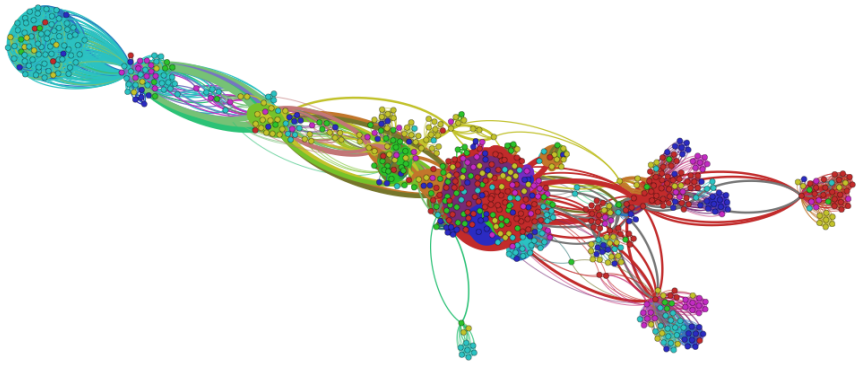


Figure 4: Authors' co-authorship graph with colored time zones.

4.7. Replica Placement

The simulator is capable of implementing arbitrary replica placement algorithms. These algorithms can use any of the information available to the simulator to make their placement decisions, for example social network information, geographic information, or the type of resources available.

We have implemented six replica placement algorithms: *universe*, *node degree*, *community node degree*, *known*, *best known*, and *random known*. Each algorithm builds an ordered list of suitable endpoints, used by the S-CDN to create the required number of replicas (k). Where k is a configurable parameter offered by the simulator. Each algorithm produces a different pattern of replica placement, as illustrated in Figure 5. Note: we do not use techniques like erasure codes [28]. Rather, we replicate datasets in their entirety. Thus, only one replica has to be available at any given time to satisfy a request.

Universe (Figure 5a) serves as a baseline algorithm in which endpoints are randomly chosen across the whole network irrespective of relationships between users, and k available endpoints are then selected as replicas. Only available endpoints are considered at the time of replica creation.

Node Degree (Figure 5b) uses the topology of the graph as well as the degree of the nodes, i.e., the number of relationships for a particular user is used to determine replica placement. First, all followers are ranked by their node degree, then unavailable endpoints are removed from the list, and finally the top k replicas are selected.

Community Node Degree (Figure 5c) uses the topology of the graph as well as the degree of the nodes, to determine replica placement. Unlike *Node Degree* this algorithm will not allow two replicas to be placed on adjacent nodes. First, all followers are ranked by their node degree, then unavailable endpoints are removed from the list, for each of the nodes in the list a replica is placed if a directly adjacent node does not already contain a replica, this process is continued until k replicas are selected.

Known (Figure 5d) uses the topology of the graph to determine appropriate replica placement. Here every user's followers' endpoints available at the time of upload are

selected as replicas. This represents a gready approach to replica selection and does not consider the limitation of k replicas.

Random Known (Figure 5e) uses a subset of *known* replicas. Here k randomly chosen followers' endpoints are selected as replicas. When considering a maximum number of replicas or endpoint quality (e.g., based on uptime, availability, endpoint bandwidth, relationship score etc.) replicas may cluster on a small number of "super endpoints". By randomly choosing followers, replicas should be more uniformly distributed from a statistical point of view.

Best Known (Figure 5f) uses the topology of the graph as well as the weight of the edges, i.e., the relationship score, to determine replica placement. First, all followers are ranked by their relationship score, then unavailable endpoints are removed from the list, and finally the top k replicas are selected.

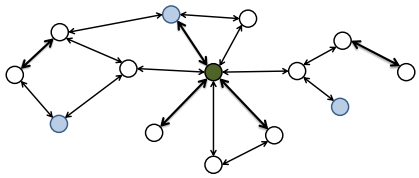
With respect to the replication algorithms presented here, we can also envisage the social sharing scenarios to which they could be most applicable. In this sense, we see them as proxies for establishing social sharing policies, as well as mechanisms to distribute and replicate data in the S-CDN. *Universe*, *Node Degree*, and *Community Degree* are closest to the concept of open data: unless access restrictions are placed on the endpoints hosting the dataset, they would be accessible to anyone. This could also represent an open scientific community where datasets are shared openly with other members, for example, nanoHUB.org [16] and myExperiment.org [17]. *Known* and *Random Known* are similar to the concept of a bounded community where the sharing of data is performed within a specific community that manages its membership, for example, a project consortium, where every community member is at least acquainted with every other member. *Best known* can be seen as localized sharing where sharing is performed within a highly localized (either geographically or homomorphically) (sub)network, for example, a team based at a single institution, or set of (geographically dispersed) co-authors consistently working together. It is important to reiterate that users manage their followers, and in doing so, enforce social sharing policies. We also envision users having multiple follower circles allowing for a differentiated enactment of sharing policies at the point of sharing a dataset.

5. Evaluation

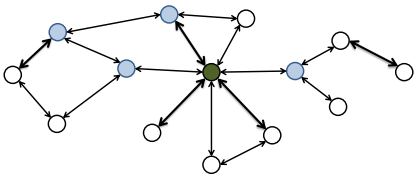
Our evaluation focuses on three aspects: 1) the benefits that can be obtained when building upon social networks with trusted relationships; 2) the performance of various data replication algorithms on the availability and spatial efficiency of the S-CDN under different scenarios; 3) the potential cost benefits of using a S-CDN over alternative mechanisms.

5.1. Extracting trust from social networks

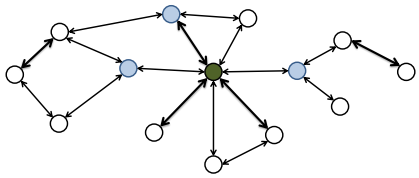
We first investigate how data availability is impacted by the choice of replica placement using different trust (sub)graphs [1]. This investigation is based on the assumption that each author is able to offer data storage resources which can be accessed by other members of the trust graph. We extract the publications history of one author (Kyle Chard) from DBLP for the time span of 2009 – 2011 and explode his authorship network to a maximum distance of 3 hops (coauthors of Kyle's coauthors' coauthors).



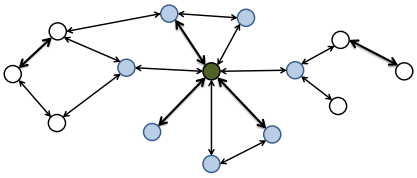
(a) *Universe*: replication to k random endpoints.



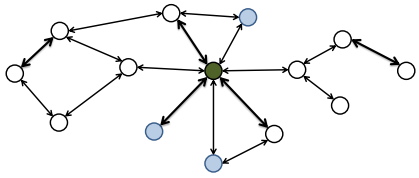
(b) *Node Degree*: replication to k endpoints ranked by degree.



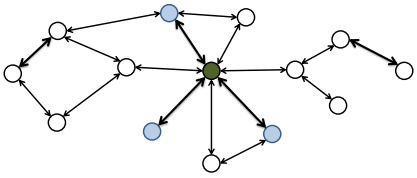
(c) *Community Node Degree*: replication to k endpoints ranked by degree avoiding adjacent nodes.



(d) *Known*: replication to all followers.



(e) *Random known*: replication to k random followers



(f) *Best known*: replication to k *closest* followers.

Figure 5: Replication algorithms: Dark green marks the originating endpoint, light blue the replicas, and white unused endpoints. Arrows represent relationships; thicker arrows represents higher scoring ones.

Note that we consider unbounded publications from the entire network, and not just from the graph seed.

Having extracted the authorship graph, we use the years 2009 and 2010 as a training set. We focus on how different replica placement algorithms affect data availability using three trust graphs composed from different trust heuristics. Once a distribution of replicas has been assigned, we then use publications from 2011 from the unbounded network (any author in the subgraph) to determine dataset availability, given new collaborations as well as new collaborators.

5.1.1. Trust subgraphs

In order to capture different notions of trust within the graph we prune the graph using different trust heuristics based on extracted social network metrics: 1) the initial graph with no trust threshold; 2) a subgraph composed of nodes and edges where authors have coauthored more than one publication together in the time period considered; and 3) a subgraph that includes only publications with fewer than six authors. This investigation is based on the hypothesis that multiple co-authorship (i.e. multiple publications with the same two authors) can be indicative of a closer working relationship and therefore a better predictor of future collaboration and conversely, publications with many co-authors (i.e. a larger number of co-authors on a publication) are less useful for predicting collaborative relationships as there is not typically a strong link between all authors. The resulting trust graphs are summarized in Table 1 and depicted in Figure 6. While the graph topologies are increasingly sparse with fewer nodes and edges, it is important to note that the maximum span remains six hops between nodes in each graph. Another observation is that, unlike the other topologies, Figure 6b includes isolated islands formed due to the pruning algorithm requiring at least two co-authorships between nodes. This can have a significant affect on the allocation of replicas as they may be disconnected from the rest of the network. However, it also serves to identify communities of trusted researchers.

Graph	Nodes	Publications	Edges
Baseline	2335	1163	17973
Double-Author	811	881	5123
Number of Authors	604	435	1988

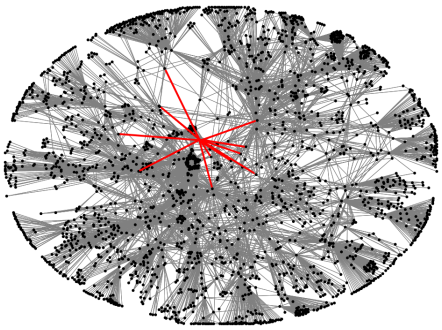
Table 1: The number of nodes and edges in each of the trust graphs.

5.1.2. Trust graph replica allocation

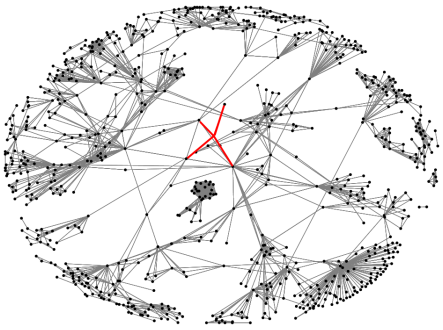
To compare the effect of these different trust heuristics we compare three unbounded replica placement algorithms on each trust graph. We use the following allocation algorithms: *Universe*, *Node Degree*, and *Community Node Degree* as described in Section 4.7.

Figure 7 shows the percentage of replica "hits" based on 2011 publications coauthored by at least one author in the trust graph. We define a "hit" as an author with a direct link to a replica (hop=1) without considering the availability of that replica based on any other factor. We consider a "miss" as an author without a direct link to a replica. We report misses only when the author exists in the subgraph; misses for authors that are not in the subgraph are constant (as they are not in any subgraph) and therefore do not affect the results except to reduce the overall hit ratio. Each of the experiments presented has been run 100 times to account for randomness.

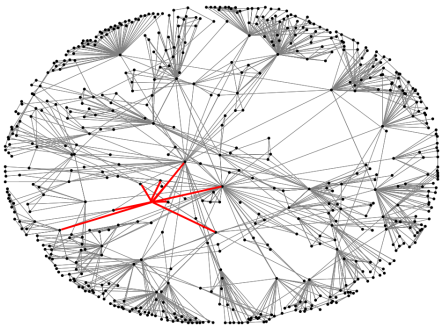
Figure 7 shows an increase in overall hit rate for each trust graph. While it is not surprising that smaller graphs have a higher hit ratio, this is not the only factor in the improvement. The increased hit ratio in trusted networks suggests a better indicator of future co-authorship. In all cases using a community "elected" replica (*Community Node Degree*) outperforms the other replica placement algorithms because it ensures



(a) Baseline Graph



(b) Double Co-authorship



(c) Number of Authors.

Figure 6: The baseline, double co-authorship and number of authors trust graphs. Authors are depicted as nodes and edges represent co-authorship on one or more publications. The node marked in red is the initial seed node for the graph and the red edges indicate the first degree relationships from this seed node.

replicas are distributed across the network rather than grouped together. When using the number of authors to create the trust graph the hit ratio of *Community Node Degree* and *Node Degree* are similar. This is because the nodes with higher degree are more evenly spread across the network in this graph. The baseline trust graph shows a near flat increase in hit ratio for *Node Degree* with more than two replicas. In fact, the rate increases by only 0.17% when 10 replicas are used. Investigation shows this is caused by a group of authors extracted from a single publication: [29]. This publication has 86 authors, which has the effect of creating an artificially high node degree for many of these edge authors, the result of which, when allocating replicas, is that the subsequent replicas added are also authors in this cluster, which only minimally increases the hit ratio as these nodes are already, at most, one hop away from a replica. This is an interesting observation, as in some domains, it is not uncommon to have a large numbers of authors and large project-based collaborations. However, this also serves to support our assumption that publications with many authors is not indicative of a close relationship between authors.

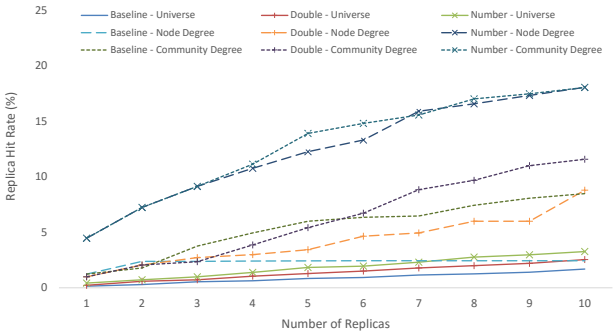


Figure 7: Replica hit rate for each trust graph and algorithm combination.

5.2. Replica Selection Algorithm Performance

To evaluate the Quality of Service (QoS) of the S-CDN under different conditions we use two performance metrics:

1) **Availability** is the number of time periods needed to commence a transfer. One purpose of a S-CDN is to provide data at the time that it is requested. However, due to the potential for transient resources, requesting a dataset and its delivery is not guaranteed to occur without some delay.

2) **Spatial efficiency** is the average number of replicas that are used per dataset. While replication can improve availability enormously, this is at the cost of increased storage space used throughout the network. Users might not participate in a S-CDN if they feel the cost (providing storage capacity) is more than their perceived gain (sharing and accessing shared data).

In considering these two performance metrics, we assess the tradeoff between availability and the level of replication needed.

We explore the use of different replication placement algorithms (Section 4.7) and consider the effect of various configurations as supported by the simulator (Section 4):

1) the ratio of *laptops* to *servers*, to observe the effect of transient resource availability; 2) *restricted* and *abstracted* modes of access to replicas, to investigate the effect of trust (in)transitivity; 3) time zones, to investigate the effects of geographic distribution and transient resource availability; 4) different seed authors, i.e. a *junior* researcher (Simon Caton) vs. a *senior* researcher (Omer Rana), to investigate the effect of topological structure; and 5) endpoint capacity, to investigate fairness of replica placement in the network.

To initialize the simulator, we construct a user graph by parsing the DBLP database for coauthored publications in 2007. This graph centers around an initial node (seed author) and creates a subgraph of depth three. Our base scenario is a graph of approximately 1,700 nodes with 20,800 edges, an average degree of 12, and an undirected network diameter of 6, with default parameter settings: 75% *laptops*, correlated time zones, a *junior* researcher, no cap on endpoint capacity, and a replication rate (k) of 5.

The workload consists of 859 uploads and as many downloads.

5.2.1. Resource Availability

Data availability in a S-CDN is dependent on the availability of the resources on which it is stored. To understand data availability, we first investigate a simple scenario with no replication ($k = 0$). Figure 8 shows the availability of a dataset at the time of request with both *normal* and *reduced* user activity. Timezones are not considered. Increasing the ratio of *laptops* results in a linear decrease in availability in both cases. This implies that changing user activity evenly will not change the behavior of the resulting availabilities, but only their absolute values, if the shape of the activity function is preserved.

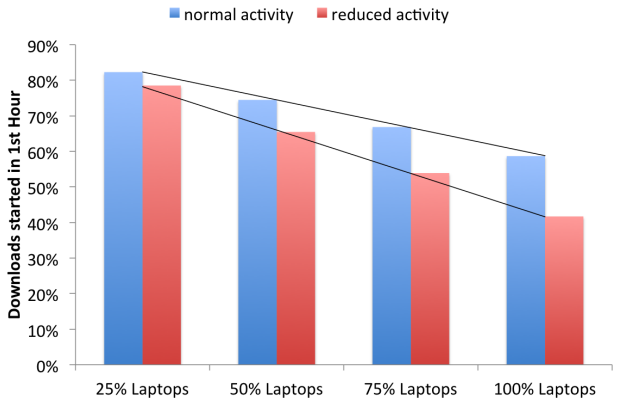
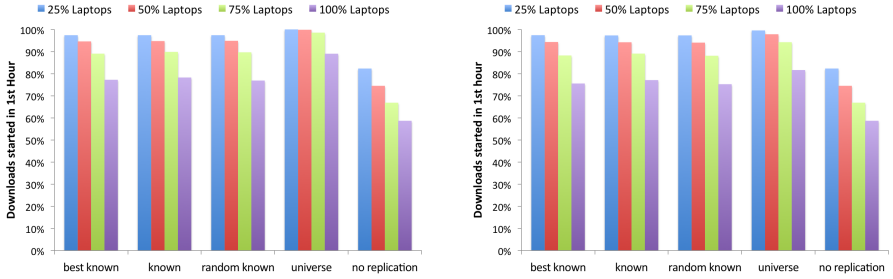


Figure 8: Chance to retrieve a dataset at the time of request with no replication.

Figure 9a and Figure 9b show the effect of varied resource availability when using 5 replicas ($k = 5$), *abstracted* access control, and various replica allocation algorithms. In this setting *Universe* performs best, and all socially-based algorithms perform similarly. The effect of resource availability does not impact the results significantly; the worst case is a reduction of 18% data availability with 100% laptops. In Figure 9b we apply *restricted* access control. Somewhat counter-intuitively, *Universe* still performs well, however we believe this is due to the (small) graph size.

These results show that high data availability can be achieved under "normal" circumstances. The mathematical basis is simple: a replica is available with probability $p_a(x)$, where a is the availability function and x is the time of request. Therefore the probability that a dataset is available for download is $p_{ak}(x) = 1 - (1 - p_a)^{k+1}$ for a k replicas with the same online probability. The worst case availability is k *laptops* in $[0, 6]$ (10%); if $k = 2$, $p_{a2}(x) = 1 - (1 - 0.1)^3 = 0.271$. However, a user only has a 10% probability of requesting a dataset in this time window. Based on our activity distribution, between 9h and 17h, an 80% chance of activity ensures not only that more requests are generated, but also that replica availability will also be more than 80%; $p_{a2}(x) = 1 - (1 - 0.8)^3 = 0.992$. Obviously, this is influenced by k, whether end-

points are capped, and other network attributes, but it is useful to illustrate as a baseline explanation of availability.



(a) Comparison of replication strategies with five replicas and *abstracted* access.

(b) Comparison of replication strategies with five replicas and *restricted* access.

Figure 9: A baseline experiment to understand the behavior of the replication algorithms in terms of availability under default parameters without time zones.

In Figure 10 we observe the general rate of availability degradation with an increase in *laptops*. As illustrated in FriendBox [30], it is important to understand the limitations of the S-CDN with respect to the percentage of *servers* needed to ensure a minimum level of availability. Regardless of the parameters chosen in the simulator, we observe the trend shown in Figure 10, i.e., a polynomial decrease in data availability with a decrease in resource availability (more *laptops*).

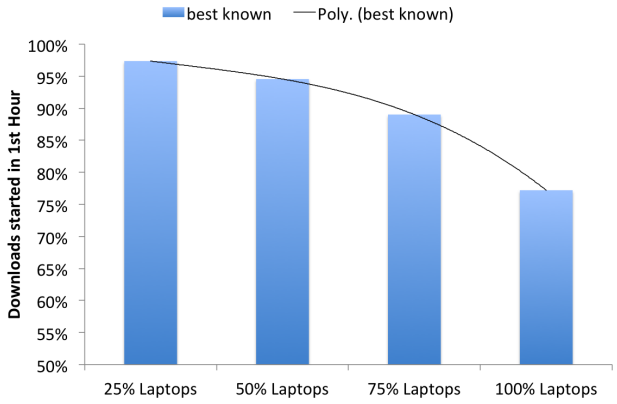


Figure 10: Availability for different laptop rates, here presented for *best known*.

5.2.2. Effect of Time Zones

We explore three scenarios for time zones: 1) assigning all users the same time zone, the best case scenario; 2) assigning users random time zones, the worst case scenario, as adjacent nodes will in most cases be in different time zones; and 3) correlating

user time zones, as described in Section 4. The effects of these scenarios on data availability is shown in Figure 11.As expected there is a small decrease in availability when time zones are correlated and a large decrease when time zones are randomly assigned. Note: the results are based on our chosen algorithm for correlating time zones and are only to be seen as guidelines.

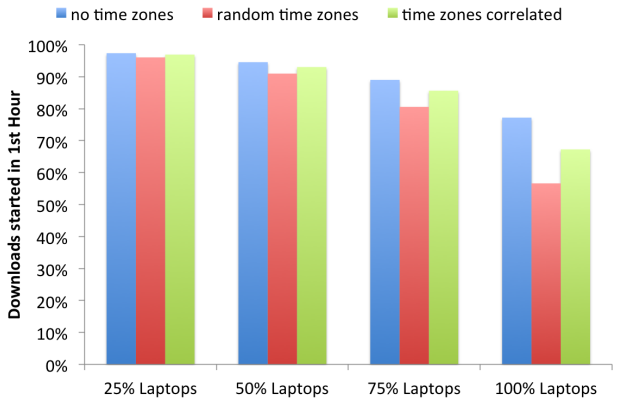
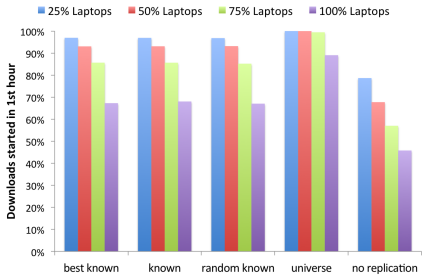
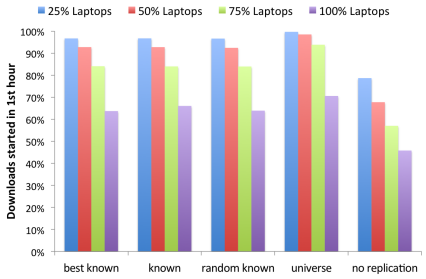


Figure 11: Effect of timezones on data availability when applying different resource availability rates.

Figure 12a and Figure 12b show the performance of different replica selection algorithms with *abstracted* and *restricted* access respectively and using correlated time zones. In all cases, the higher the laptop rate (the fraction of resources that are laptops), the more influential the restriction. This is especially the case for *Universe*.



(a) *Abstracted* and correlated time zones.



(b) *Restricted* and correlated time zones.

Figure 12: Behavior of availability with the introduction of time zones and otherwise default conditions.

5.2.3. Level of Replication

Figure 13 shows how different numbers of replicas effects performance. Here we explore different values for k for *best known*. As anticipated, there is an increase in

availability with more replicas. This effect is more pronounced as the percentage of *laptops* increases. However, the approximately 2% increase per replica is not a significant improvement.

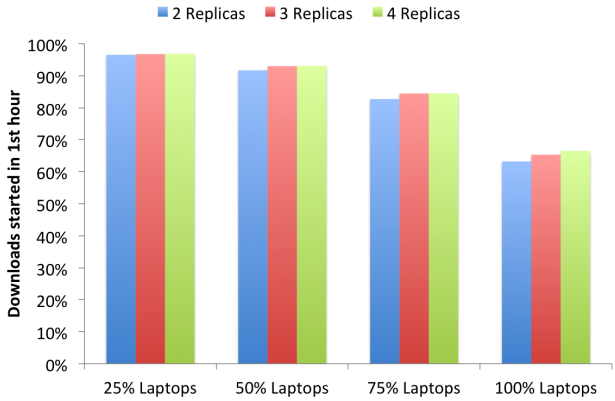


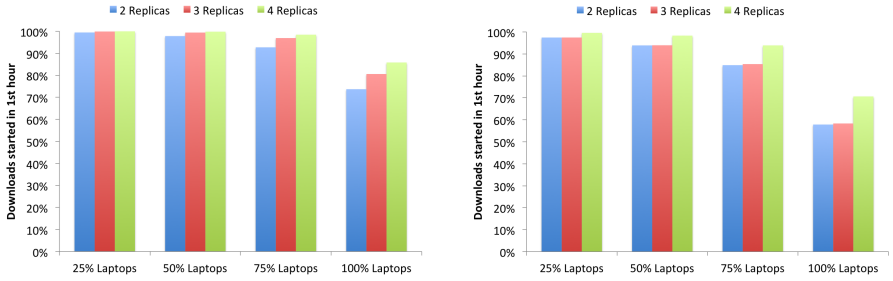
Figure 13: Data availability when varying the number of replicas. *best known* (abstracted).

We do not include results for *Known* as there is no performance difference, that is, the number of replicas is dependent on the topology. In the abstracted setting when using *Universe* (Figure 14a) data availability is high. This setting benefits more than any other from additional replicas, which, as discussed in [31], is because the algorithm circumvents the correlated online behavior of followers. That is, by distributing replicas across the network, they are less likely to be based in correlated time zones. In the restricted case (Figure 14b) there is only a small difference in data availability between using two and three replicas, however there is a significant increase in data availability when using four replicas. This is due to the small graph size (with maximum hop of three), making it more likely that at least one replica is placed in the immediate vicinity of a user. In a bigger graph, significantly more replicas would be needed to achieve the same effect.

5.2.4. Limited Replicas per Endpoint

Figure 15 shows the spatial efficiency (number of replicas) for each algorithm. While data availability is comparable for each replication algorithm, the number of replicas (which is proportional to storage space) for each algorithm differs. *Universe* and *Known* use the most space. *Universe* uses k replicas irrespective of the topology, while *Known* places replicas on all followers' resources. *Best known* and *random known*, however, use significantly less space without sacrificing availability; both place no more than k replicas.

While Figure 15 highlights the spatial efficiency of the various algorithms it does not consider the number of replicas placed on a single endpoint. This fact is highlighted in Figure 16 which shows replica placement fairness; red marks an endpoint with more than five replicas (max 39).



(a) *Universe* (abstracted). (b) *Universe* (restricted).

Figure 14: Data availability when varying the number of replicas (k), otherwise default parameters.

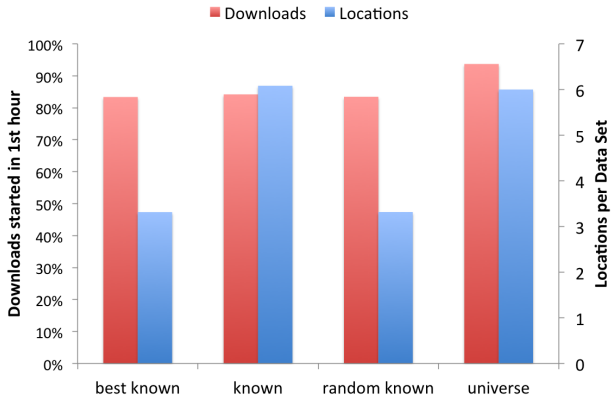


Figure 15: Availability and spatial efficiency: 75% laptop rate, 5 replicas, restricted access

A more realistic scenario is shown in Figure 17. Here each endpoint imposes a limit on the number of replicas that can be hosted. The graph shows data availability with a limit of 3, 5 or 7 replicas per endpoint using the *Best known* replication algorithm. The results show that fairer allocation across endpoints does not significantly effect availability.

5.2.5. Comparison of Topologies

The graph size and topology varies significantly depending on the selected seed author. We now consider two types of seed authors: a *junior* and a *senior* researcher. The junior researcher graph features 54 nodes with 32 datasets and 139 relationships while the senior researcher graph features 964 nodes with 454 datasets and 3,507 relationships. Figure 18 shows the data availability with these two seed authors when using *abstracted* access to replicas. The results show that while the difference in topology effects data availability, the difference is small.

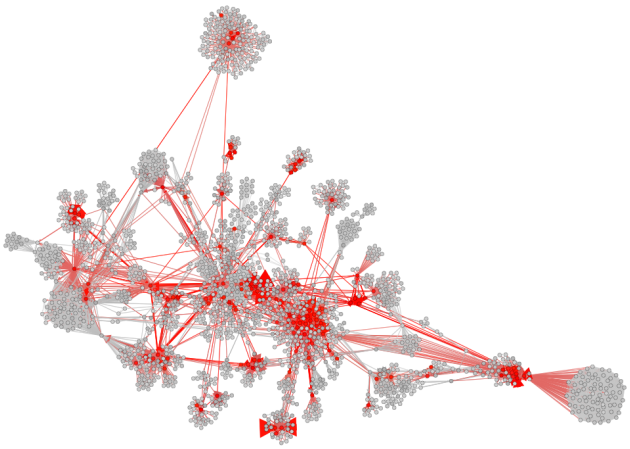


Figure 16: Distribution of replicas, red endpoints contain more than five replicas.

5.3. Data Sharing Cost Comparison

One of the motivations for an S-CDN is that the cost of using commercial cloud solutions for large datasets may be cost prohibitive. To investigate the potential advantages of the S-CDN model, we explore the costs associated with hosting a scientific dataset on several commercial platforms. Table 2 shows the cost of storage and data transfer using each of these approaches. All values are obtained from the respective published websites.

An enterprise account for Dropbox costs $750 per year, for 5 users and up to 5GB. A personal Dropbox account allows a single user access to 1 TB for $99 per year. Backup and version history are included. Hosting 1 TB of data on Amazon S3 costs $30 ($0.03 per GB) per month and $360 per year. Additional charges for PUT and GET requests apply, although they are small. Data transfers outside of AWS also incur additional charges, for example copying the entire 1-TB dataset out of S3 would cost an additional $90 ($0.09 per GB).

While it is difficult to quantify the cost of using a S-CDN as users contribute partial resources from existing systems we can establish a baseline cost for a dedicated S-CDN endpoint created using commodity hardware. The typical price for a low-end PC is approximately $500 and a Western Digital 4-TB hard disk for storage costs approximately $150. This amounts to up front costs of $650. A PC based system can run for as low as 90 W, which amounts to 788.400 kWh per year. At the US average price of $0.12 per kWh we expect this cost to be $94.68. This would result in a total cost of $745 for the first year and $95 for subsequent years. These calculations do not consider additional network costs.

After two years of operation the dedicated PC-based system would cost less than any of the cloud systems when storing 4TB. Subsequent operation would result in further savings as upfront costs are amortized over several years. Moreover, the premise of the S-CDN relies upon resource contribution and therefore the perceived cost to users

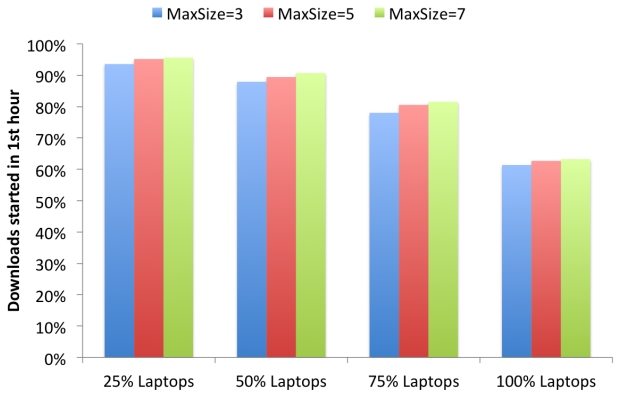


Figure 17: Performance of *best known* when varying the number of replicas stored per endpoint.

	Dropbox Enterprise	Dropbox Personal	Amazon S3	S-CDN PC
Number of users	5	1	unlimited	unlimited
Storage	5	1	1	4
Up-front cost	-	-	-	650
Cost per year (1 TB)	750	99	360	-
Cost per year (energy)	-	-	-	95
Transfer cost	-	-	90	-
Annual cost	750	99	380	745

Table 2: Comparison of annual costs (US$) for different storage technologies.

would be significantly lower than operating a dedicated endpoint.

5.4. Discussion

Importantly, our results confirm existing literature that shows that the availability of participating users has a huge impact on data availability in a collaborative storage environment [30]. While others conclude that transient storage must be augmented with persistent storage (e.g., Cloud storage) to provide satisfactory quality of service. We have shown that in situations where there is a mixture of both transient (laptops) and persistent (user-owned servers) storage, a S-CDN can provide suitable quality of service even with a relatively small number of servers randomly distributed throughout the network. We also have shown that a S-CDN benefits from its inherent structure. For example, as peers are typically geographically co-located, overall dataset availability is increased with respect to data access patterns when compared with random distributions.

Our results show that the basis of trust that can be extracted from social networks can provide improved data availability. By pruning a social network to create trusted

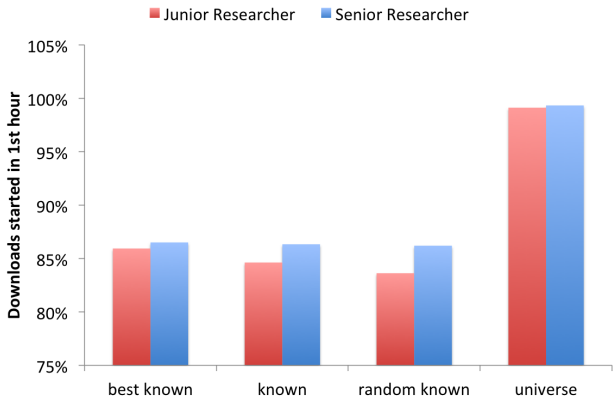


Figure 18: Replica availability for *junior* and *senior* researchers' topologies (75% laptop rate).

overlays we are able to improve replica allocation–in essence by predicting future co-authorships. When combined with social network-based replica selection algorithms, especially those that ensure replica distribution, we showed that data availability can be significantly improved.

Interestingly, our replication algorithms improved dataset availability, but not in the form that we had anticipated. Our results show that strictly using the network topology often does not facilitate the desired level of replication. While this factor improves the spatial efficiency of *Best known* it does so at almost no expense of data availability. *Universe* performs well in terms of availability if *abstracted* access to replicas is assumed. Random allocation of replicas also provides significant value if one is willing to compromise trust and instead share data publicly. While privacy and security can be derived in other manners (e.g., encryption), incentives for participation may be weaker in this scenario.

Finally, we showed that the cost of using a S-CDN can be significantly lower than that of commercial cloud storage. In a realistic setting, in which existing resources are contributed, these costs could be considered negligible and therefore provide a low-cost alternative to using cloud storage.

6. Related Work

The S-CDN builds upon a number of areas of related work including friend-based storage, CDNs, decentralized social online networks, and (social) data replication algorithms.

6.1. Friend to Friend storage

Friend to Friend (F2F) storage systems provide capabilities similar in purpose to the S-CDN, enabling users to contribute storage resources which may be subsequently used by their friends. F2F storage is based upon the establishment of a private P2P

network utilizing direct connections between peers. The most well known examples of F2F storage are FriendStore [32] and FriendBox [30].

FriendStore is designed to support cooperative backup of data. It allows users to backup their data to a subset of their connected peers—essentially creating a replica. Unlike the S-CDN, here the process of replication is manually driven, users must actively select the nodes on which they wish to store (replicate) data. FriendStore establishes storage contracts, a form of socially-based contract, that, via trust derived from the social relationship provides some level of guarantee over data storage.

FriendBox is a more automated approach for collaborative storage and sharing of data. Like FriendStore, users operate storage clients which may be used by their friends to store data on their storage resources. FriendBox includes a social user interface that serves as an access point to the service. Unlike the S-CDN model, FriendBox is designed to leverage commercial cloud storage to overcome decreased availability caused by interpersonal correlations. In addition, to increase availability, FriendBox also places data replicas on other sites or, in a more sophisticated approach, use Reed-Solomon erasure codes to split data into fragments.

The most significant differences between the S-CDN and F2F storage is related to the the motivation for data storage and the methods for replication. In F2F storage the primary goal is for users to store their own data. The S-CDN on the other hand focuses both on storage and dissemination to other users. Moreover, the S-CDN is designed to handle large research datasets and therefore includes methods by which data is moved asynchronously and automatically. In the case of FriendBox, replicas are created on commercial providers when friend-based storage is insufficient whereas the S-CDN relies entirely on contributed infrastructure. Finally, the S-CDN leverages automated replication decisions based on analysis of social constructs that are derived via social adapters.

6.2. Content Delivery Networks

There are a number of commercial CDNs available including Akamai [33], Amazon Cloud Front, and Limelight. Commercial CDNs are built upon a global distributed network of dedicated servers that provide high performance access to data. While such networks could be used to share scientific data, they are not typically designed to handle large datasets and usage is typically expensive. Open source CDNs, such as Coral CDN, can be used to create a P2P-based CDN, however without considerable resource contributions it is unlikely that such networks can scale to meet the needs of scientific data sharing.

Traditional CDNs focus on the problems associated with distribution of popular data. However, recently, the focus has shifted towards the distribution of less popular, user generated content derived from platforms such as Facebook and Youtube. Like the research data supported by the S-CDN, user generated content is considered long-tail. It therefore lacks the scale to warrant significant investment in distribution. To address these unique needs researchers have developed hybrid CDN models that leverage dynamically provisioned cloud storage. For example, MetaCDN [34] and Content-delivery-as-a-service (CoDaaS) [35]

MetaCDN is a cloud-based CDN model in which the CDN relies on disparate cloud storage providers (e.g., Amazon S3). MetaCDN has been shown to be a low cost

alternative to existing commercial options, while providing similar functionality such as replication, fail-over, geographical load redirection, and load balancing. CoDaaS is a cloud-based approach for distributing user generated content. It enables on-demand virtual content delivery via an elastic service built upon a hybrid media cloud.

6.3. Decentralized Social Online Networks

Decentralized social online networks (DSONs) like Safebook [36], Peerson [37], My3 [38], Gemstone [39], and SODESSON [40], and the information distribution networks that support them such as the Social CDN [41], also put major importance on sharing of data and while leveraging social relationships for trust and security.

Peerson and My3 replicate data across peers so that a user's content is only available to themselves and their friends, independent of online status. Updates are spread across peers until the profiles are consistent. Requests are satisfied by choosing replicas based on availability, performance, and geographical location of a user's friends.

Safebook implements a *Matryoshka* structure, where every request to access a dataset is recursive in nature and the number of hops between nodes determines users' access rights. Thus, restrictive access policies can be defined that ensure only close nodes have access to data.

Gemstone presents a unique approach using a DHT as the foundation and Gemstone itself as a social overlay. Encrypted replicas are distributed according to an algorithm that utilizes online time, social relationships, and user experience to determine which nodes are used. Gemstone's sophisticated replication algorithms are shown to reach availabilities up to 99% in a small world graph.

SODESSON proposes a publish/subscribe system not unlike the the S-CDN to deliver messages over rendezvous-points in either offline or online models. SODESSON uses a locality-based DHT and social routing tables which contain identifiers of a user's friends' devices to allow messages to be stored on socially connected peers' resources.

The Social CDN [41] consists of social caches [42] that enable data distribution between friends using a variety of socially-aware distribution algorithms. Like our S-CDN, the Social CDN uses social networks to form the CDN. Unlike our approach, the Social CDN focuses on techniques to manage small datasets (e.g., user profile information), and thus different tradeoffs are considered. However, these approaches demonstrate that such techniques are viable for distributing data across a pool of collaborative users.

The major differences between DSONs and the S-CDN are the type of data exchanged and the inherent networks considered. DSONs focus on small messages that are aimed at bounded groups of followers and which can be downloaded by many of these followers in their entirety. S-CDN datasets are often large, may be accessed by many, and represent significant investments to be downloaded in their entirety. Thus, the replication approaches that can be applied are significantly different as the S-CDN must consider storage efficiency and cost. Moreover, most DOSN approaches consider Facebook-like networks that feature nodes with an average of hundreds of friends. From our analysis of co-authorship networks this may not be true for research networks as many researchers publish with only few co-authors over a long period of time.

6.4. Data Replication Algorithms

There has been only limited investigations into social replication algorithms. Rzadca et al. [43] examine the problem of peers' greediness by analyzing replica placement in both centralized and decentralized environments. Using analysis and simulation to evaluate the consequences of peers' selfishness on the performance of the network they prove that thee problem is NP-hard and propose a series of replication heuristics that adopt the average case.

In the context of DSONs several replication algorithms have been proposed. For example, Gemstone uses a P2P overlay that applies learning mechanisms to data replication using social relationships and online behavior of peers to increase availability. SODESSON uses a P2P network that, in addition to leveraging social context, allows data transfers to temporarily offline devices, while considering optimal latencies for data transfer and scalability. While these approaches may be of use in the S-CDN researchers focus solely on the availability of datasets in the network and do not consider spatial efficiency.

S-Clone [44] aims to determine if, and where, data should be replicated in order to maintain an efficient and scalable network. To this end, it places replicas of a user's data on servers that are primary access points of the communities' members. The approach deals both with a static scenario in which the graph never changes and also a dynamic scenario. The authors show that S-Clone can improve replication efficiency, load balancing, migration cost, and the consistency of replication efficiency.

7. Summary and Future Work

The S-CDN applies a socially-based model for enabling collaborative storage and dissemination of data. It is designed to alleviate the challenges faced by researchers who lack sophisticated distribution networks for sharing their data. By adopting a proven CDN-based approach and combining it with the use of user-contributed "edge nodes" the S-CDN provides a low-cost alternative model to purchasing dedicated infrastructure or utilizing commercial cloud storage. Our prototype S-CDN, which builds upon Globus as a reliable data access and transfer middleware and leverages social network relationships to establish trusted sharing scenarios, demonstrates both the feasibility and value of a S-CDN.

In this paper we have shown, via simulation, that replicating data based on social structures provides high availability and spatial efficiency, even with transient resources. We showed that the use of social network and trust metrics can be used to improve data availability. We also observed that correlated work patterns mitigate times of low resource availability, that our algorithms scale with network size, and that fairer replica placement does not decrease availability. To make our simulations as realistic as possible, we considered co-authorship-based social networks, academic work patterns, the inherent geographic distribution of users, the size of the social network, and multiple seed authors.

There are several areas that warrant further investigation such as social network metrics, security and redundancy measures, adaptive replication techniques, and associating metadata and provenance. We aim to focus on the definition, extraction and

usage of rich social network metrics to improve replica allocation and dataset retrieval. We have shown that such approaches can provide considerable value; however, significant work is required to further optimize these approaches. Our current implementation is also unable to support partial replicas, which may be able to improve storage of large datasets as well as increase spatial efficiency and availability, as more endpoints would be used to service a request. Over time it is inevitable that dataset popularity and requirements will change. The S-CDN requires adaptive replication techniques to analyze and respond to dynamic usage patterns, for instance, being able to migrate a replica to improve availability, adding replicas to service increased demand or overcome availability issues, and optimizing replica placement across a large storage network. As the number of datasets grows users will require more sophisticated search methods to locate datasets. As such, we expect requirements to arise that warrant association of more complex metadata with datasets. Similarly, we expect that users may wish to track the provenance of datasets. This presents an interesting opportunity as the social basis of the S-CDN could enable more accountable tracing of data access than other methods.

Acknowledgments

Work by Katz was supported by the National Science Foundation while working at the Foundation; any opinion, finding, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

[1] K. Chard, S. Caton, O. Rana, D. S. Katz, A social content delivery network for scientific cooperation: Vision, design, and architecture, in: Proceedings of the SC Companion: High Performance Computing, Networking, Storage and Analysis (SCC), 2012, pp. 1058–1067. doi:10.1109/SC.Companion.2012.128.

[2] D. P. Anderson, BOINC: A system for public-resource computing and storage, in: Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing (GRID '04), IEEE Computer Society, Washington, DC, USA, 2004, pp. 4–10. doi:10.1109/GRID.2004.14.

[3] K. Kugler, K. Chard, S. Caton, O. Rana, D. S. Katz, Constructing a social content delivery network for eScience, in: Proceedings of the 9th IEEE International Conference on e-Science (e-Science), 2013, pp. 350–356.

[4] K. Kugler, S. Caton, K. Chard, D. S. Katz, On replica placement in a social CDN for e-Science, in: Proceedings of the 10th IEEE International Conference on e-Science (e-Science), 2014, pp. 13–20.

[5] K. Chard, K. Bubendorfer, S. Caton, O. Rana, Social cloud computing: A vision for socially motivated resource sharing, IEEE Transactions on Services Computing 5 (4) (2012) 551–563.

[6] C. Haas, S. Caton, K. Chard, C. Weinhardt, Co-operative infrastructures: An economic model for providing infrastructures for social cloud computing, in: Proceedings of the 46th Hawaii International Conference on System Sciences (HICSS), 2013, pp. 729–738.

[7] C. Haas, Incentives and two-sided matching-engineering coordination mechanisms for social clouds, Ph.D. thesis, Karlsruhe Institute of Technology (KIT) (2014).

[8] S. Caton, C. Haas, K. Chard, K. Bubendorfer, O. F. Rana, A social compute cloud: Allocating and sharing infrastructure resources via social networks, IEEE Transactions on Services Computing 7 (3) (2014) 359–372.

[9] M. S. Granovetter, The strength of weak ties, The American Journal of Sociology 78 (6) (1973) 1360–1380.

[10] G. Bachi, M. Coscia, A. Monreale, F. Giannotti, Classifying trust/distrust relationships in online social networks, in: Proceedings of the International Conference on Privacy, Security, Risk and Trust (PASSAT) and the International Conference on Social Computing (SocialCom), 2012, pp. 552–557.

[11] B. Carminati, E. Ferrari, M. Viviani, A multi-dimensional and event-based model for trust computation in the social web, in: K. Aberer, A. Flache, W. Jager, L. Liu, J. Tang, C. Guret (Eds.), Social Informatics, Vol. 7710 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012, pp. 323–336. doi:10.1007/978-3-642-35386-4_24.

[12] T. DuBois, J. Golbeck, A. Srinivasan, Predicting trust and distrust in social networks, in: Proceedings of the International Conference onPrivacy, Security, Risk and Trust (PASSAT) and the Inernational Conference on Social Computing (SocialCom), 2011, pp. 418–424.

[13] A. Rahbar, O. Yang, Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing, IEEE Transactions on Parallel and Distributed Systems.

[14] L. Xiong, L. Liu, Peertrust: Supporting reputation-based trust for peer-to- peer electronic communities., IEEE Transacitons on Knowledge and Data Engineering.

[15] S. Caton, C. Dukat, T. Grenz, C. Haas, M. Pfadenhauer, C. Weinhardt, Foundations of trust: Contextualising trust in social clouds, in: Cloud and Green Computing (CGC), 2012 Second International Conference on, IEEE, 2012, pp. 424–429.

[16] G. Klimeck, M. McLennan, S. P. Brophy, G. B. Adams, M. S. Lundstrom, nanoHUB.org: Advancing education and research in nanotechnology, Computing in Science and Engineering 10 (2008) 17–23.

[17] D. De Roure, C. Goble, R. Stevens, The design and realisation of the myExperiment virtual research environment for social sharing of workflows, Future Generation Computer Systems 25 (5) (2009) 561–567.

[18] I. Foster, Globus Online: Accelerating and democratizing science through cloud-based services, IEEE Internet Computing 15 (3) (2011) 70–73.

[19] Django Project, `https://www.djangoproject.com/`.

[20] W. Allcock, J. Bresnahan, R. Kettimuthu, M. Link, C. Dumitrescu, I. Raicu, I. Foster, The globus striped gridftp framework and server, in: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing (SC'05), 2005, p. 54.

[21] K. Chard, S. Tuecke, I. Foster, Efficient and secure transfer, synchronization, and sharing of big data, Cloud Computing, IEEE 1 (3) (2014) 46–55.

[22] K. Chard, M. Lidman, J. Bryan, T. Howe, B. McCollam, R. Ananthakrishnan, S. Tuecke, I. Foster, Globus nexus: Research identity, profile, and group management as a service, in: Proceedings of the 10th IEEE International Conference on e-Science (e-Science), Vol. 1, 2014, pp. 31–38.

[23] I. Foster, R. Ananthakrishnan, B. Blaiszik, K. Chard, R. Osborn, S. Tuecke, M. Wilde, J. Wozniak, Networking materials data: Accelerating discovery at an experimental facility, in: Volume 26: Big Data and High Performance Computing, IOS Press, 2015, pp. 117–132.

[24] J. Wozniak, K. Chard, B. Blaiszik, R. Osborn, M. Wilde, I. Foster, Big data remote access interfaces for light source science, in: Proceedings of the 2nd IEEE/ACM International Symposium on Big Data Computing (BDC), 2015, pp. –.

[25] M. Ley, The DBLP computer science bibliography: Evolution, research issues, perspectives, in: A. Laender, A. Oliveira (Eds.), String Processing and Information Retrieval, Vol. 2476 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2002, pp. 1–10.

[26] J. B. Begole, J. C. Tang, R. B. Smith, N. Yankelovich, Work rhythms: Analyzing visualizations of awareness histories of distributed groups, in: Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW), ACM, 2002, pp. 334–343.

[27] N. Eagle, A. Pentland, Eigenbehaviors: identifying structure in routine, Behavioral Ecology and Sociobiology 63 (7) (2009) 1057–1066.

[28] A. Dimakis, P. Godfrey, Y. Wu, M. Wainwright, K. Ramchandran, Network coding for distributed storage systems, IEEE Transacitons on Information Theory 56 (9) (2010) 4539–4551.

[29] M. Riedel, E. Laure, T. Soddemann, L. Field, J. P. Navarro, J. Casey, M. Litmaath, J. P. Baud, B. Koblitz, C. Catlett, D. Skow, C. Zheng, P. M. Papadopoulos, M. Katz, N. Sharma, O. Smirnova, B. Kónya, P. Arzberger, F. Würthwein,

A. S. Rana, T. Martin, M. Wan, V. Welch, T. Rimovsky, S. Newhouse, A. Vanni, Y. Tanaka, Y. Tanimura, T. Ikegami, D. Abramson, C. Enticott, G. Jenkins, R. Pordes, N. Sharma, S. Timm, N. Sharma, G. Moont, M. Aggarwal, D. Colling, O. van der Aa, A. Sim, V. Natarajan, A. Shoshani, J. Gu, S. Chen, G. Galang, R. Zappi, L. Magnoni, V. Ciaschini, M. Pace, V. Venturi, M. Marzolla, P. Andreetto, B. Cowles, S. Wang, Y. Saeki, H. Sato, S. Matsuoka, P. Uthayopas, S. Sriprayoonsakul, O. Koeroo, M. Viljoen, L. Pearlman, S. Pickles, D. Wallom, G. Moloney, J. Lauret, J. Marsteller, P. Sheldon, S. Pathak, S. De Witt, J. Mencák, J. Jensen, M. Hodges, D. Ross, S. Phatanapherom, G. Netzer, A. R. Gregersen, M. Jones, S. Chen, P. Kacsuk, A. Streit, D. Mallmann, F. Wolf, T. Lippert, T. Delaitre, E. Huedo, N. Geddes, Interoperation of world-wide production e-science infrastructures, Concurrency and Computation: Practice & Experience 21 (8) (2009) 961–990.

[30] R. Gracia-Tinedo, M. Sanchez-Artigas, A. Moreno-Martinez, P. García-López, FriendBox: A hybrid F2F personal storage application, in: Proceedings of the 5th IEEE International Conference on Cloud Computing (CLOUD 2012), 2012, pp. 131–138.

[31] R. Gracia-Tinedo, M. Sanchez-Artigas, P. Garca-Lpez, F2Box: cloudifying F2F storage systems with high availability correlation, in: Proceedings of the 5th IEEE International Conference on Cloud Computing (CLOUD 2012), 2012, pp. 123–130.

[32] D. N. Tran, F. Chiang, J. Li, Friendstore: Cooperative online backup using trusted nodes, in: Proceedings of the 1st Workshop on Social Network Systems (SocialNets), ACM, New York, NY, USA, 2008, pp. 37–42.

[33] E. Nygren, R. K. Sitaraman, J. Sun., The akamai network: A platform for high-performance internet applications, ACM SIGOPS Operating Systems Review 44 (3) (2010) 2–19.

[34] J. Broberg, R. Buyya, Z. Tari, MetaCDN: Harnessing 'storage clouds' for high performance content delivery., Journal of Network and Computer Applications 32 (5) (2009) 1012–1022.

[35] Y. Jin, Y. Wen, G. Shi, G. Wang, A. Vasilakos, CoDaaS: An experimental cloud-centric content delivery platform for user-generated contents, in: Proceedings of the International Conference on Computing, Networking and Communications (ICNC), 2012, pp. 934–938.

[36] L. Cutillo, R. Molva, T. Strufe, Safebook: A privacy-preserving online social network leveraging on real-life trust, Communications Magazine, IEEE 47 (12) (2009) 94–101.

[37] S. Buchegger, D. Schiöberg, L.-H. Vu, A. Datta, PeerSoN: P2P social networking: early experiences and insights, in: Proceedings of the Second ACM EuroSys Workshop on Social Network Systems (SNS), ACM, New York, NY, USA, 2009, pp. 46–52.

[38] R. Narendula, T. G. Papaioannou, K. Aberer, My3: A highly-available P2P-based online social network, in: Proceedings of the IEEE International Conference on Peer-to-Peer Computing (P2P), 2011, pp. 166–167.

[39] F. Tegeler, D. Koll, X. Fu, Gemstone: Empowering decentralized social networking with high data availability, in: Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM), 2011, pp. 1–6.

[40] M. Florian, F. Hartmann, I. Baumgart, A socio- and locality-aware overlay for user-centric networking, in: Proceedings of the International Conference on Computing, Networking and Communications (ICNC), 2014, pp. 327–333.

[41] L. Han, M. Punceva, B. Nath, S. Muthukrishnan, L. Iftode, SocialCDN: Caching techniques for distributed social networks, in: Proceedings of the 12th IEEE International Conference on Peer-to-Peer Computing (P2P), 2012, pp. 191–202.

[42] L. Han, B. Nath, L. Iftode, S. Muthukrishnan, Social butterfly: Social caches for distributed social networks, in: Proceedings of the 3rd IEEE Inernational Conference on Social Computing (SocialCom) and Privacy, Security, Risk and Trust (PASSAT), 2011, pp. 81–86.

[43] K. Rzadca, A. Datta, S. Buchegger, Replica placement in P2P storage: Complexity and game theoretic analyses, in: Proeedings of the 30th IEEE International Conference on Distributed Computing Systems (ICDCS), 2010, pp. 599–609.

[44] D. A. Tran, K. Nguyen, C. Pham, S-CLONE: Socially-aware data replication for social networks, Computer Networks 56 (7) (2012) 2001 – 2013.