Enabling Mobile Distributed Social Networking on Smartphones

Kanchana Thilakarathna, Henrik Petander
NICTA & UNSW, Australia
[kanchana.thilakarathna; henrik.petander]
@nicta.com.au

Julián Mestre
USYD, Australia
mestre
@it.usyd.edu.au

Aruna Seneviratne
NICTA & UNSW, Australia
aruna.seneviratne
@nicta.com.au

ABSTRACT

Distributed social networking services show promise to solve data ownership and privacy problems associated with centralised approaches. Smartphones could be used for hosting and sharing users data in a distributed manner, if the associated high communication costs and battery usage issues of the distributed systems could be mitigated. We propose a novel mechanism for reducing these costs to a level comparable with centralised systems by using a connectivity aware replication strategy. To this end, we develop an algorithm based on a combination of bipartite b-matching and a greedy heuristics for grouping devices into tribes among intended content consumers. The tribes replicate content and serve it using low-cost network connections by exploiting time elasticity of user generated content sharing. The performance is evaluated using three real world trace data sets. The results show that a persistent low-cost network availability can be achieved with an average of two replicas per content. Additionally, a content creator can reduce 3G traffic by up to 43% and device energy use by up to 41% on average compared to content sharing in non-mobile-optimised distributed social networking approaches. Moreover, the results show that the proposed mechanism can provide the benefits of a distributed content sharing system for monetary and energy costs comparable to those of a centralised server based system.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Store and forward networks*

Keywords

User Generated Content Sharing, Cellular Data Traffic Offloading, Mobile Social Networking

MSWiM'12, October 21–25, 2012, Paphos, Cyprus.

1. INTRODUCTION

The increasing capabilities of mobile devices are driving rapid growth in the use of these devices in social networking applications/services. Majority of current social network services, such as Facebook, Twitter, YouTube and Google+, use the mobile device as a thin client with all the data associated with at the applications being centrally hosted on the servers of the service provider. This centralised approach has several benefits, for both users and the service provider. For the service provider the benefits of the centralised architecture are the ease of management and the ability for application/service provider to easily mine user data. For the user the advantages are the ability to access the service and the data from any device and perception that the data is safe. The primary disadvantage for the service provider is the cost associated with storage and distribution of the data. However, the benefits that can be obtained by mining the user data outweighs all the disadvantages of hosting the application/service centrally. The disadvantages for the user are twofold. Firstly, perceived advantage of the data being safe comes at the cost of the users loosing the ownership of their data. In some cases, they may not even be able to export it out of the service, thus locking them in to a single service. Secondly, as the data is mined by the application/service provider, the users loose their privacy. In contrast to the service provider, there are no over arching benefits for the user that will outweigh these disadvantages.

This has lead to the development of distributed decentralised social networking architectures, which are being proposed to be used for the next-generation of social networking services [9, 5, 16, 13]. The distributed architectures provide users with control of their data by enabling an individual user or a community of users to host their data, thus overcoming the disadvantages associated with loosing control and leakage of privacy due to mining of their data by the service provider. In general, this requires the users need to provide "always on" devices which will host the data or rent resources from a hosting service. The hosting service can be provided by the service provider themselves or a third party. If the hosting service is provided by the service provider, the users may still loose control or and more importantly the providers may mine the data. If provided by a third party, inconvenience and the costs of hosting will limits its appeal and usability.

Today most users have devices such as smartphones and tablets which have significant computing power, storage capacities and are connected to a network. Its is likely that these devices in the future will have even more computing

power, and storage. Although there have been numerous proposals for peer-to-peer services which potentially utilise these capabilities, they have not been harnessed to date for providing distributed social network services. This is primarily due to the communications costs and battery usage. For example, with mobile data plans with capped limits, if Safebook [5] and MyZone [13] were to be provided using mobile device resources, it would be prohibitively expensive as they replicate content on a number of devices without consideration of communications costs or power usage. To this end, we showed that if it is possible to exploit the time elasticity of social networking content and the fact that the user will have access to high speed low-cost network such as WiFi networks, having spare capacity at some time periods of the day, it is possible to provide social networking services for sharing user generated content [19]. The proposed MobiTribe architecture provides user the necessary control of their data, and also protect their privacy.

This paper provides a deeper analysis of the MobiTribe architecture described in [19] to show that it is possible to provide a persistent content availability through low-cost networks by replicating the content only twice. We show that peer-to-peer storage schemes such as MobiTribe, can provide the same functionality as centralised architectures with the same or lower distribution costs and energy usage and it is always cost efficient than content sharing via non-mobile-optimised distributed social networking. The contributions of this paper are:

- Show that the content replication problem in distributed peer-to-peer architectures such as MobiTribe is *NP-Hard* and then present a new algorithm based on a combination of a bipartite b-matching and a greedy heuristic, which minimises content replication and maximises content availability whilst ensuring fairness;
- Models for determining the communication cost and the device energy consumption; and
- Using real data traces, we show the viability of the proposed algorithm, namely that it is possible to reduce the network load created by content sharing without compromising the availability of the content. For the particular tracks, MobiTribe content creator can save up to 43% of 3G bandwidth and up to 41% of device energy compared to non-mobile-optimised distributed social networking architecture.

The remainder of the paper is organised as follows: The next section presents related work. This is followed by an overview of the MobiTribe system architecture in Section 3. In Section 4, we formally define the content replication problem in peer-to-peer architectures such as MobiTribe and Section 5 presents our device grouping algorithm. The evaluation of the device grouping algorithm is presented in Section 6. Finally, Section 7 concludes the paper.

2. RELATED WORK

Diaspora [9] can be considered as the only widely used distributed social networking architecture. It is fully funded by user donations and estimated to have over 1.5 million users in March 2012 indicating a high demand for privacy-aware social networking. In order to host data, Diaspora users need to set up their own server for hosting their content. PrPl [17] is another proposed service which enables a user to run the service on a home server, or rent a vendor to

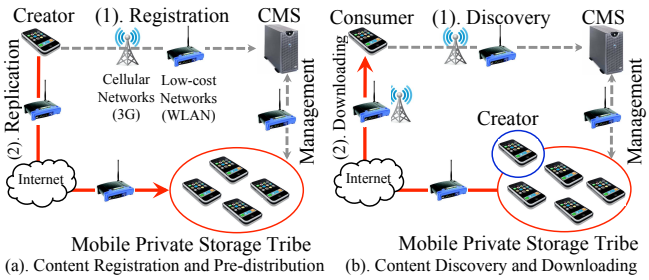


Figure 1: MobiTribe System Overview

host own data. If a mobile device is used to host the content as proposed in these services, either the availability has to be compromised or incur increased communications costs.

Safebook [5] is based on the concept of decentralisation and collaboration among friends and creates a secure social network. Similar to MobiTribe, friends are assumed to be cooperative and friends devices are used for replication to increase the availability. MyZone [13] also deploys user profile replicas on the devices of trusted friends to increase the availability of the profile. However, either system does not explicitly consider mobile devices, and the availability is dependent on the number of replicas. Hence, similar to Diaspora, if used with mobile devices, it would result in increased communication costs and energy consumption of devices. Contrail [18] is a cloud-based architecture to enable distributed social networking on smartphones which is conceptually similar to MobiTribe. The cloud relays content between users and stores content for offline recipients to increase the reliability. Though, it is not completely centralised, the content is stored in a third-party location, whereas MobiTribe propose to store content only among intended consumers.

Tribler [16] is a peer-to-peer file sharing system, which takes advantage of social relationships. Peers replicate their contextual information related to interests to enable local searching capability for relevant content. Also, peers are clustered into social groups based on similar interests. Again although there are similarities to MobiTribe, Tribler does not consider methods to increase the availability of the content or minimise communication costs.

In ActiveCast [12], the content is pre-fetched and cached in different locations including mobile phone via low-cost connectivity. An operator managed memory partition inside the mobile phone is used to cache content. ActiveCast does not address uploading or hosting of user generated content. Han et al. [8] have proposed a target set selection for propagation of the content with opportunistic communication. This mechanism will only work well for popular content, whereas popularity distribution of user generated content has a long tail [4]. Moreover, the focus of all this work is dissemination of data from a "centralised" data store and as a result does not address the privacy and censorship issues which MobiTribe addresses.

3. THE MOBITRIBE ARCHITECTURE

The primary objectives of MobiTribe are to provide users control of their data and to protect their privacy, without negatively impacting the usability and cost. The proposed MobiTribe architecture is shown in Figure 1. A *Creator*

distributes the content to be shared to a group of devices, *Mobile Private Storage Tribe*, which would be selected as described in Section 3.2. The pre-distribution of content is carried out as described in Section 3.1. A *Consumer* would then access the shared content through the mobile private storage tribe. The architecture relies on a centralised entity *(Content Management Server - CMS)* to track the addresses and current connectivity of devices hosting content and to mange the peer-to-peer content distribution.

3.1 Sharing and accessing of content

A creator could share content via a social networking application similar to usual social networking applications/services. Then, the underneath MobiTribe platform initiates the content sharing process in two steps: *1). Registration:* The content would be registered with the CMS immediately after content creation. The registration can be done through any available network as shown in Figure 1a. Then, a link to the content, pointing to the CMS, is posted to the users who are supposed to view the content. *2). Replication:* Devices that take part in MobiTribe are grouped into tribes, as described below in Section 3.2. The content to be shared is replicated on the devices of the tribe, thus improving its availability to all users. The replication of the content is done when the creator and the devices in the storage tribe have access to low-cost networks. The low-cost network connections can be spare cellular network capacity, WLAN access or direct connectivity. The CMS would coordinate the content replication process acting as a tracker. After this pre-distribution process has been completed, the content will have very high availability over spare network capacity as a result of the device grouping strategy described in Section 4. The strategy ensures that the devices in the tribe have complementary low-cost network access patterns, thus making it highly probable that one of them can upload the content over a low-cost network connection. The pre-distribution is not instantaneous, and time required will depend on the network connectivity available to the creator and devices in the tribe. If during this time another user wanted to access the content, the content creator has the option of delivering the content over the least congested data connection available to the creator or make it appear as unavailable until the pre-distribution process has been completed.

Figure 1b shows the content access mechanism in MobiTribe. A consumer could initiate the content downloading process by clicking on a shared notification appearing in the social networking application. The notification feed contains the link to the CMS and trigger the content downloading process. *1). Discovery:* When the CMS receives a query for content, it directs requests to the devices in the mobile private storage tribe that the device should connect to get the requested content. The choice of the devices is based on the network connectivity and the load of available devices in the tribe. *2). Downloading:* A content consumer has the option of downloading the content over whatever network connection available or could be scheduled to download through low-cost networks. In contrast, the upload part of the path, i.e. from tribe to fixed network edge, would always be over spare network capacity. In addition, during the download and after it, the consumer would also act as a seeder in BitTorrent to share the clip temporarily helping the system to scale with demand.

To perform the content sharing between the content creator, devices in the tribe and content consumers, a peer-to-peer protocol, such as BitTorrent, could be used with the help of the CMS acting as a tracker to balance the traffic load and increase the scalability and data rates of transfers. We consider the details of the peer-to-peer mechanisms to be outside the scope of this paper due to lack of space.

3.2 Making a tribe for replicating content

A mobile device, if used as a content store, cannot provide access to the content continuously for three primary reasons, namely the communications costs, battery usage, and possibly not being "on" all the time. However, if devices which have complementary connectivity, battery power and being "on" can be found, and if the content can be replicated on these devices, it will be possible to provide similar availability to content that is hosted in a centralised server. Moreover, through this process, if the traffic can be routed through a low-cost and/or less congested network, it will result in significant benefits to both the service providers and the users. This can be achieved by having a dedicated encrypted storage partition in the devices in the tribe, which is controlled by the service and the content to be only accessible through an interface provided by the service. The down sides of the content replication are the overheads of replication such as the additional use of cellular bandwidth, energy and storage capacity of the mobile devices. Therefore, device grouping is crucial for the viability and also efficiency and performance of MobiTribe.

The device grouping is performed by the CMS, as it has a global view of the system state in terms of both mobile devices and network infrastructure. CMS carries out the grouping periodically or in response to a significant degradation to the availability of a tribe's content. To perform device grouping, all devices need to update the CMS with their context information, namely the low-cost network availability, spare storage capacity of the device and the remaining battery life of the device periodically. This context information is used to create tribes as described in Section 5. Although storing context information on a centralised server may result in some loss of privacy, this is insignificant in comparison to exposing all user information.

4. DEVICE GROUPING PROBLEM

The primary objective of the device grouping is to locate a tribe of devices for content replication such that the tribe maximises the availability of a content via low-cost networks. To enable that at least one device in the tribe needs to be always available at a low-cost connection with sufficient battery resources. Hence, a device could only be used to host content at a given time instance, if it has low-cost connectivity, sufficient battery and spare storage capacity. To this end, we name the behavioural pattern of capability to host content on the device as "Device Availability" pattern. The device availability patterns can be generated based on the device context information such as available network types, the battery level, AC power availability and the amount of spare storage capacity on the mobile device. The communication cost can be further optimised by analysing network infrastructure context information available at the central management entity such as connected cell ids, network load and relative importance of the traffic.

In our previous work [19], we showed that the history of device availability patterns can be effectively used to infer

Table 1: Contingency Table

		Device 2		
		p	q	r
Device 1	p:available	n_{pp}	n_{pq}	n_{pr}
	q:unavailable	n_{qp}	n_{qq}	n_{qr}
	r:unknown	n_{rp}	n_{rq}	n_{rr}

future availability patterns of a device and groups of devices. Therefore, we aim to group devices into tribes based on history of device availability patterns of devices. Suppose that there are two devices having device availability patterns for M time instances. The first step to determine the potential for pairing is to derive a *Contingency Table*, shown in Table 1, that can be considered as the summary of behaviour of the availability patterns. The table contains elements n_{ij} indicating the number of time instances that one device in state i and the other in state j, where i, j stand for the availability states of the each device, available: p, unavailable: q and unknown: r.

If at least one device in the tribe has availability, the content can be considered as available, as it enables the delivery of the content to the consumer. Hence, the probability of availability of a tribe can be defined as $P_a = \frac{n_{pp}+n_{qp}+n_{pq}+n_{pr}+n_{rp}}{M}$. In addition, pre-distribution of content is performed only when both end devices are connected to low-cost networks. To this end, we define the device grouping metric $P_a^{W_o}$, a modified probability of availability;

$$P_a^{W_o} = \frac{(n_{pp} + n_{qp} + n_{pq} + n_{pr} + n_{rp}) + W_o \times n_{pp}}{M + W_o \times n_{pp}}$$

The overlapping weight, W_o is used to control the impact of overlapping low-cost availability during the group selection, which is evaluated in Section 6. If $P_a^{W_o}$ is higher than a certain threshold - the "Threshold of Pairing", $P_{threshold}$, we consider that the tribe satisfies the availability constraint. However, we cannot implicitly select tribes based on threshold of pairing because replication consumes scarce resources in a mobile environment such as network bandwidth, storage and energy of devices. To this end, we consider two additional constraints in device grouping to manage the resource limitations in mobile networks.

Minimum Replication: It is obvious that the availability of the content increases with the level of replication. However, a high level of replication will disrupt the system performance by creating additional traffic flows and overheads, wasting battery life and spare storage of mobile devices. Hence, the device grouping should be able to mange the trade-off between replication and availability. Here after, the minimum replication is confined by the "Limit of Replication", i.e. the maximum allowed size of a tribe.

Fairness: The system should not place an extra burden on the resources of devices with good low-cost network availability. For instance, if such a device is selected by many users to host data, it would drain energy and storage of that device. Therefore, the replication mechanism should make sure to replicate content fairly among the devices. In the remainder of the paper, the fairness of the system is quantified by the "Limit of Hosting", i.e. the maximum number of tribes that a single device can belong to.

Each device in the system should be able to locate a tribe of devices while satisfying the threshold of pairing, limit of replication and limit of hosting. After considering all these constraints when a device is able to find its replication tribe, we say that the device is *covered*. Thus, the content replication problem becomes *"how to find a maximum cover while ensuring minimum replication and fairness constraints"*.

PROBLEM DEFINITION

A hypergraph is a pair (V, E), where V is a ground set of elements and E is a collection of subsets of V. The rank of a hypergraph $H = (V, E)$ is defined as $\Delta_H = \max_{e \in E} |E|$; we will drop the subscript H when the hypergraph is clear from the context. For a subset C of E and a vertex $u \in V$, we denote the *degree of* u *in* C by $\deg_C(u) = |\{e \in C : u \in e\}|$.

In the DEVICE GROUPING (DG) problem we are given a hypergraph $H = (V, E)$ and a capacity function $c : V \to \mathbb{Z}^+$. The objective is to find a subset $C \subseteq E$ that maximises the number of vertices spanned by C, namely;

$$\begin{array}{ll} \text{Maximise} & |\{u \in V : \deg_C(u) \geq 1\}| \\ \text{subject to} & \deg_C(u) \leq c(u) \text{ for all } u \in V \end{array}$$

The relation between DG and our application is as follows. The vertex set V represents the set of devices. The hyperedge set E is the collection of subsets of V satisfying the availability constraint, $P_{threshold}$. The parameter Δ is the maximum allowed group size, i.e. the limit of replication. Finally, the capacity function c is used to upper bound the maximum number of tribes that a single device can belong to, i.e. the limit of hosting. The objective is to perform a grouping that covers (spans) as many devices as possible. Our first result is to show that DC is computationally hard even for hypergraph of rank 3.

THEOREM 1. DG *is NP-Hard even when* $\Delta = 3$ *and* $c(u) = 1$ *for all* $u \in V$.

PROOF. Let (V, H) be a hypergraph. A matching is a subset C of E, where the sets in C are pairwise disjoint. The matching is perfect if every vertex is covered by some edge in C; namely, $\cup_{e \in C} e = V$. When $\Delta = 3$, testing if such a matching exists is known as the 3D-matching problem and is one of Karp's original 21 NP-Hard problems [10].

The hardness of DG follows immediately since the case specified in the theorem statement is the problem of finding the largest subset $C \subseteq E$ such that sets in C are pairwise disjoint. □

5. DEVICE GROUPING ALGORITHM

5.1 A tractable special case of DG

Even though the problem is hard when $\Delta \geq 3$, we can show that the problem is solvable in polynomial time when $\Delta = 2$, i.e. when (V, E) is a graph. This shows that the hardness proof in the previous section cannot be pushed to an even simpler version of the problem.

First, we reduce DG to the MINIMUM WEIGHT DEGREE CONSTRAINED SUBGRAPH (MWDCS) problem for $\Delta = 2$. Let, a graph (V, E), a weight function $w : E \to R^+$, and lower and upper degree constraints $L(u) \leq U(u)$ for each $u \in V$ as inputs to MWDCS. The objective is to find a minimum weight subset $F \subseteq E$ such that $L(u) \leq \deg_F(u) \leq U(u)$ for all $u \in V$. Gabow [6] gave an $O(U(V)|E| \log |V|)$ time algorithm for this problem, where $U(V) = \sum_{v \in V} U(v)$.

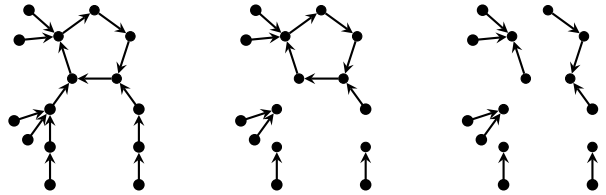


Figure 2: An example of how F is pruned in the proof of Theorem 3. First edges outside the cycle are pruned, and then edges inside the cycle are pruned.

THEOREM 2. DG *can be solved in* $O(c(V)|E|\log|V|)$ *time when* $\Delta = 2$.

PROOF. Given an instance (V, E, c) of DG, we construct an instance of WMDCS (V', E', U, L, w). The graph (V', E') is obtained by taking (V, E) and adding an edge (u, u') for each $u \in V$; in other words, $V' = \{u, u' : u \in V\}$ and $E' = E \cup \{(u, u') : u \in V\}$. We set $w(e) = 0$ for all $e \in E$ and $w(e) = 1$ for all $e \in E' \setminus E$. Finally we set $L(v) = 1$ and $U(v) = c(v)$ for all $v \in V$, and $L(v') = 0$ and $U(v') = 1$ for all $v \in V$. It is trivial to check that a solution for the DG problem covering X vertices induces a solution for the degree constrained problem with weight $|V| - X$, and vice versa. Therefore, a minimum weight solution for the degree constrained problem corresponds to a maximum coverage solution for the DG problem. □

While this settles the complexity of DG when $\Delta = 2$, this result does not constitute a satisfactory solution for our problem because Gabow's algorithm [6] involves complex data structures and there are no implementations of it available. Instead, we show that in most cases our problem can be reduced to a single bipartite b-matching computation for which highly tuned implementations are available (the problem reduces in a straightforward way to maximum flow). An instance of the b-matching problem is specified by a bipartite graph (A, B, E) and a function $b : A \cup B \to \mathbb{Z}^+$. The objective is to find a maximum cardinality subset M of E such that $\deg_M(u) \leq b(u)$ for all $u \in A \cup B$.

THEOREM 3. *When* $\Delta = 2$ *and* $c(u) > 1$ *for all* $u \in V$*, we can reduce* DG *to a single bipartite* b*-matching computation in linear time.*

PROOF. Let (V, E, c) be our DG instance. We construct a bipartite graph (A, B, E) where $A = \{v : v \in V\}$ and $B = \{v' : v \in V\}$; for each $(u, v) \in E$ we include two edges (u, v') and (v, u') in E. In addition, we set $c(u) = 1$ and $c(u') = b(u)$ for each $u \in V$.

We claim that any solution to the DG instance induces a solution to the b-matching problem with the same value, and vice versa. To prove the forward direction, let $C \subseteq E$ be a solution to DG. For each vertex u covered by C, we choose an arbitrary edge $(u, v) \in C$ incident on it, and add (u, v') to the matching. It is trivial to show that the cardinality of the matching equals the number of devices covered by C and that the matching is feasible. The other direction takes more effort.

Let M be some b-matching. We construct a directed graph (V, F) such that if $(u, v') \in M$, we add the directed edge (u, v) to F. The meaning that we want to attach to edges is that device u is *being covered* by v. Our ultimate goal is to find a solution for DG that covers the same number of devices as M. We use $\deg^+_F(u)$ to denote the number of edges in F going out of u, and $\deg^-_F(u)$, the number of edges coming into u. Notice that for every vertex $u \in V$, we have $\deg^+_F(u) \leq 1$ and $\deg^-_F(u) \leq c(u)$; this follows immediately from b-matching constraints. This means that in the underlying undirected graph induced by F, each connected component will be made up of one-trees (a tree plus one edge, or equivalently, a graph with a single cycle).

It is tempting to think that we could disregard the directions of the edges in F and use this as our DG solution. This, however, will not work because the combined in and out degree of a vertex u can be $c(u) + 1$. Ultimately, we will ignore the direction of edges in F, but first we need to prune some unnecessary edges. We say a node is a *leaf* in F if $\deg^-_F(u) = 0$. We repeatedly perform the following operation: If there is a node u such that $\deg^+_F(u) = 1$ and $\deg^-_F(u) \geq 1$ and its in-neighbourhood is made up of leaves, then delete from F the edge going out of u. Once we reach a point where we cannot delete any more edges, we must be left with a collection of disjoint stars and directed cycles with some additional edges pointing to vertices in the cycles as in the second graph in Figure 2. If a vertex u in cycle has an incoming edge from outside the cycle, then we remove from F its outgoing edge as in the third graph in Figure 2.

At this point it is not difficult to show that if a vertex u still has its out-going edge, it must be the case that u is a leaf or it was a cycle node with a single incoming edge from other vertex in a cycle. Notice that in either case $\deg^-_F(u) + \deg^+_F(u) \leq 2$. For all other vertices with no out-going edges we have $\deg^-_F(u) + \deg^+_F \leq c(u)$. Therefore, if we just disregard the directions of edges in F, the solution is now feasible for DG because the $c(u) > 1$ for all $u \in V$ by the assumption made in the theorem statement.

The only thing left to show is that every vertex covered by the b-matching is also covered by the DG solution. Notice that every time we removed an edge (u, v) from F we always had another edge (x, u) with $\deg^-_F(x) = 0$; thus, we are guaranteed that (x, u) will survive further pruning and remain in F until the end. Therefore, vertex u will be covered by the (undirected) edge (x, v) in our DG solution. □

5.2 A greedy approximation algorithm for general instances

For general hypergraphs (V, E), it is not possible to solve the problem in polynomial time since it is NP-Hard. However, we will be able to approximate the objective efficiently. This section shows that the following greedy heuristic has an approximation ratio bounded by Δ, the size of the largest set in E. The algorithm works in iterations by building a solution C. In each iteration, it finds an edge e that can safely added to C without violating the capacity constraints so as to maximise the number of newly covered elements.

Algorithm 1 GREEDY(V, E, b)

1. $C \leftarrow D \leftarrow \emptyset$
2. **while** $\exists\, e \in E : deg_{C+e}(u) \leq c(u)$ for all $u \in e$ **do**
3. Let e an edge in E maximising $|e \setminus D|$ such that $deg_{C+e}(u) \leq c(u)$ for all $u \in e$
4. $C \leftarrow C + e$
5. $D \leftarrow D \cup e$ {vertices in e are now covered}
6. **return** C

We note in passing that the set systems defined by the feasible solutions of the DG was shown to be Δ-extendible by Mestre [14] and the maximization objective is submodular. For such a problem, Chekuri et al. [3] showed that greedy is a $\Delta + 1$ approximation. However, for our special objective function, a slightly better approximation is possible.

THEOREM 4. *Let (V, E, b) be an instance of* DG*, then* GREEDY *is a Δ-approximation algorithm.*

PROOF. Let O be an optimal solution. The idea is to assign devices covered by O to devices covered by C so no device covered by C is assigned more than Δ devices covered by O. First though, we will assign sets in O to sets in C.

Suppose that GREEDY picks a set e in a given iteration. For each $u \in e$ we find some set f_u in O, if any, such that $u \in f_u$; we assign f_u to e and we remove f_u from O. We claim that at the end of the algorithm all sets in O must be assigned. Indeed, let f be some set in O. If $\deg_C(u) = c(u)$ for some $u \in f$, then clearly f must have been assigned. Otherwise, $\deg_C(u) < c(u)$ for all $u \in f$, but this means that GREEDY left the while-loop prematurely because f can be safely added to C, a contradiction.

Because of our greedy choice, when adding e to C the number of newly covered devices cannot be larger if we had added f_u to C. More specifically, $|e \setminus D| \geq |f_u \setminus D|$ for the set D just before adding e to C. Thus, we can distribute the devices covered by $\{f_u\}$ to devices covered by e so that each newly-covered device in e receives at most Δ devices from $\{f_u\}$. Therefore, the overall number of devices covered by O is no more than Δ times the overall number of devices covered by C. $\square$

5.3 Device Grouping in practice

DEVICE GROUPING will be carried out at a central entity. Each device periodically updates the central entity with its recent device availability patterns. From the device availability patterns, an undirected graph among all pairs of devices will be generated such that there exists an edge between two devices u and v if $P_a^{W_o} > P_{threshold}$. We use this undirected graph to solve DG for the case of $\Delta = 2$, i.e. a maximum tribe size of two. Next, we construct a bipartite graph to implement the single bipartite b-matching reduction as in Theorem 3. The maximum flow computation is used to find b-matching by introducing two extra nodes s and t. All incoming edges to node t from a device u have the capacity equal to $c(u)$, i.e. the limit of hosting. All the other edges have the capacity of one. Thus, the limit of hosting ensures that the node u only takes part in a maximum of $c(u)$ tribes. Once we compute the maximum flow from s to t, we prune extra edges from the resulted directed graph as illustrated in Figure 2. Finally, we ignore the direction of the edges and consider it as the solution of DG problem for the case of $\Delta = 2$. As the next step, if there are ungrouped devices, the GREEDY approximation algorithm proved in Theorem 4 is used to find tribes for the remaining devices.

6. PERFORMANCE EVALUATION

We evaluate the effectiveness of the DEVICE GROUPING algorithm in the context of device grouping using WiFi connectivity patterns of three data sets. Effectiveness of DEVICE GROUPING can be quantified by the low-cost network availability of a selected tribe, required replication for a content and the maximum number of tribes that a single device belongs to. Since MobiTribe depends on user behaviour, an analysis on real mobile users connectivity patterns is the best method to perform a comprehensive and realistic evaluation. To this end, we were able to find three data sets which are applicable to our particular problem, but with limited context information. Therefore, in this evaluation, the context information considered for low-cost network selection is limited to WiFi connectivity patterns of the devices.

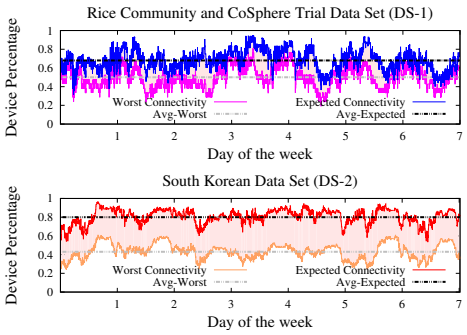
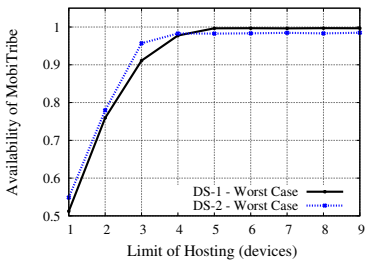


Figure 3: Percentage of devices which have WLAN connectivity

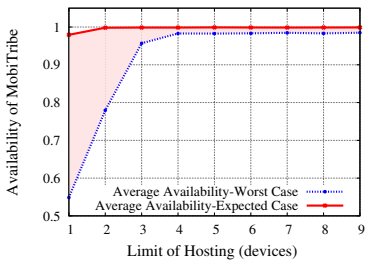
6.1 Dynamics of Data Sets

The Rice Community [1] trace data set has WiFi connectivity information with a one minute granularity from 14 users in Houston for 6 weeks in Feb. 2007. The CoSphere trial [1] contains WiFi connectivity patterns of 12 users for 6 weeks in Mar. 2007. We refer to these two sets as DS1. Further, we were able to use WiFi connectivity patterns of 97 iPhone users from a recent study conducted by Lee et al. [11] in South Korea in Feb. 2010 which spanned over 18 days with a 3 minute granularity. We refer to this data set as DS2. However, neither data set has consistent connectivity patterns, i.e. there are time instances for which the exact state of WiFi connectivity is unknown. In cases where there is only one time stamp gap between data points, we filled it to match the surrounding state if the surrounding states are the same. Otherwise, the gap was left as "unknown". However, we had to remove traces of 18 devices due to a high portion of unknown states. If a device has WiFi connectivity, we consider that the device is available via a low-cost network and vice versa. If we consider all "unknown" states as "unavailable", that gives the worst possible low-cost network availability for a particular device. On the other hand, we can remove the "unknown" states from the evaluation, which then gives the realistic low-cost network availability of the devices. Hence, these two cases are used to obtain bounds for the evaluation metrics as the worst and the expected cases where necessary.

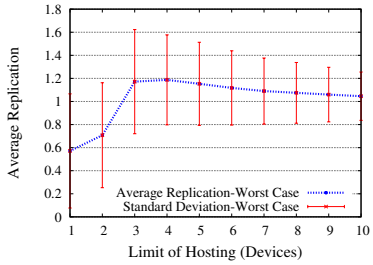
The portion of devices having WiFi connectivity at a given time is shown in Figure 3 for both expected and worst cases. The worst case average WiFi availability for DS1 data set is 50%, i.e. each device has on the average WiFi available for 50% of the time, and for DS2 43%. The difference between expected and worst case is higher in DS2 suggesting that it has more unknown states compared to DS1. Altogether, the data sets contain 1470 days of connectivity patterns from 105



(a) For DS1, $P_a = 0.99$,$W_o = 0.5$. For DS2, $P_a = 0.95$,$W_o = 0.5$



(b) Expected and worst case availability of DS2



(c) Expected and worst case replication of DS2

Figure 4: Availability and replication of MobiTribe vs limit of hosting

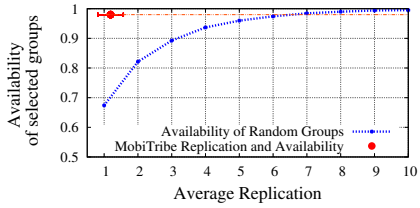


Figure 5: Availability of randomly selected groups

users in three different cites at three different times. Thus, the combination of DS1 and DS2 contains a wide range of WiFi connectivity patterns of real mobile users which can be considered as large enough and diverse enough for the evaluation of the proposed DEVICE GROUPING algorithm.

6.2 Availability, Replication and Fairness

Availability in MobiTribe is the probability of low-cost network availability (P_a) of content hosted in a selected tribe. We performed DEVICE GROUPING over a seven day training period[1], calculated P_a for all selected tribes and averaged it to obtain the availability of MobiTribe for the entire system. This procedure was repeated over the entire trace data period to obtain average system evaluation metrics. It was observed that the fairness constraint described in Section 4, limit of hosting, i.e. the maximum number of tribes that a single device belongs to, affects the efficiency and performance of grouping. The main reason for this is that the devices with high availability would be preferred by many others to host content. Hence, we measured the availability of MobiTribe against the limit of hosting.

Figure 4a shows the availability variation of DS1 and DS2 with respect to limit of hosting. Note that both data sets give similar availability behaviour irrespective of their different connectivity characteristics. This indicates that the DEVICE GROUPING algorithm have the potential to adjust to different types of mobile environments. Figure 4b depicts that in the expected case, we can achieve nearly 100% availability irrespective of the limit of hosting. In the worst case, limit of hosting has to be more than 4 for the best performance. Hence, the availability of tribes will fall on to the shaded area in Figure 4b. In the remainder of this paper, only DS2 is used since it has a larger device pool and the worst case analysis is used in every case to benchmark the performance of MobiTribe with alternative architectures.

Replication of the system is the average number of replicas per content in addition to the original content (R_a). When the limit of hosting is lower, some devices may not be able to find a tribe, which lowers the average replication as shown in Figure 4c. When the limit of hosting is higher, devices are allowed to select a single device with high availability instead of several devices with low availability, which again lowers the average replication. The maximum average replication required to satisfy the availability requirement of $P_a = 0.95$ is $R_a = 1.19$ with standard deviation of 0.39. That is, in this particular social network, each user only needs to group with one or two other users to increase the low-cost network availability of its own content.

This analysis shows that a very high low-cost network availability of content (nearly 100%), can be achieved with a very low number of replicas (approximately 2), if the devices are carefully grouped into tribes. Although the result is subjective to these trace data sets, we argue that the DEVICE GROUPING algorithm has the capability to group devices appropriately when there is such diversity in the WiFi connectivity patterns. This minimises the overhead of replication in terms of network bandwidth, storage and battery consumption. Earlier proposed systems, such as Safebook [5] or MyZone [13], do not select devices for replication based on the connectivity patterns. Instead, they use random replication, which requires a higher number of replicas than MobiTribe to reach a similar level of availability. We performed a random group selection in order to compare the performance with MobiTribe strategy as shown in Figure 5. Random selection requires 7 replicas to reach the same availability level that is provided by MobiTribe with 1.19 replicas. Hence, the random replication would not be suitable for mobile environment, since it consumes more scarce resources in mobile networks. For content sharing through distributed architectures such as Diaspora [9] which do not replicate the content, the availability would depend on whether the user's device has low-cost network availability at the time when another user wants to access the content. In the case of such systems if the device has no network access or is switched off, the content will not be accessible at all.

6.3 Delay Tolerance

After a device has registered its content with the CMS, it has to wait until one of the devices in the tribe has overlapping low-cost network availability to pre-distribute the content, namely the pre-distribution delay. As described in

[1]The effect of training period selection is analysed in [19] with a heuristic device grouping algorithm.

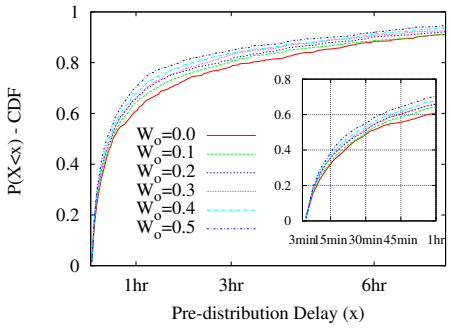


Figure 6: The CDF of the pre-distribution delay

Table 2: Network Usage Models for MobiTribe. The probability of successful pre-distribution (M_a), the average replication per content (R_a), the probability of availability of a selected tribe (P_a), the number of downloads (N), and the probability of WLAN availability of a device (W_a), size of the content (F_s)

MobiTribe WLAN Usage Model	
Uplink	$[M_a R_a + (1 - M_a) W_a N + M_a P_a N] F_s$
Downlink	$[M_a R_a + W_a N] F_s$
MobiTribe 3G Usage Model	
Uplink	$[M_a(1 - P_a) + (1 - M_a)(1 - W_a)] N F_s$
Downlink	$(1 - W_a) N F_s$

Section 3.1, if the content is accessed before this time, it will be delivered from the originating device over the available lowest cost network. Hence, this pre-distribution delay is critical for the performance of MobiTribe. We calculated the pre-distribution delay of selected tribes repeatedly for seven days. The CDF of the delay for different overlapping factors are shown in Figure 6. As expected, the higher the overlapping factor W_o the lower the delay. However, the effect of W_o seems not very significant compared to the added complexity to DEVICE GROUPING. The probability of the delay being less than 1 hour is over 0.6 irrespective of W_o. There is 90% probability for the pre-distribution delay being less than 6 hours. In [20], it has been observed that 55% of Flicker content is uploaded after a lag of more than one day. Hence, we believe that the results of more than 6 hours delay tolerance would be practically significant.

6.4 Time Complexity of Device Grouping

We have noticed that we can cover the majority of devices by using bipartite b-matching algorithm for all three data sets. Hence, the time complexity of the DEVICE GROUPING is dominated by the bipartite b-matching algorithm. Since bipartite b-matching is directly reduced to the maximum flow, the complexity is bounded by the maximum flow computation. There are many fine tuned algorithms available for maximum flow computation in the literature [7]. In this evaluation, there were 2905 sets which satisfied the $P_{threshold} = 0.95$ for 84 devices in DS2. As per our notation in Section 4, $|V| = 84$ and $|H| = 2905$. In general, if a system has n devices, there can be a maximum of $\frac{n(n-1)}{2}$ number of pairs or edges in the hypergraph (V, H) for the case of $\Delta = 2$. Then, the bipartite flow network (A, B, E) consists of $|A \cup B| = 2|V| + 2 = 2n + 2 \approx 2n$ vertices and $|E| = 2(\frac{n(n-1)}{2})$ edges. Hence, if we use Goldberg's algorithm [7], the upper bound for the bipartite b-matching would be $O\left(|A \cup B||E| \log \frac{|A \cup B|^2}{|E|}\right) \approx O(n^3)$ in general. This can be considered as a practical time complexity for our initial NP-Hard DEVICE GROUPING problem.

6.5 3G Bandwidth Saving Comparison

Cellular network capacity is typically scarce and minimizing its usage is one of the key goals of MobiTribe. To evaluate its efficiency in this regard, we benchmark the performance of MobiTribe with a centralised server architecture, which is currently the common method of content sharing in social networks such as Facebook. Further, we compare the performance of MobiTribe with a simple Diaspora [9] like non-mobile-optimised distributed architecture.

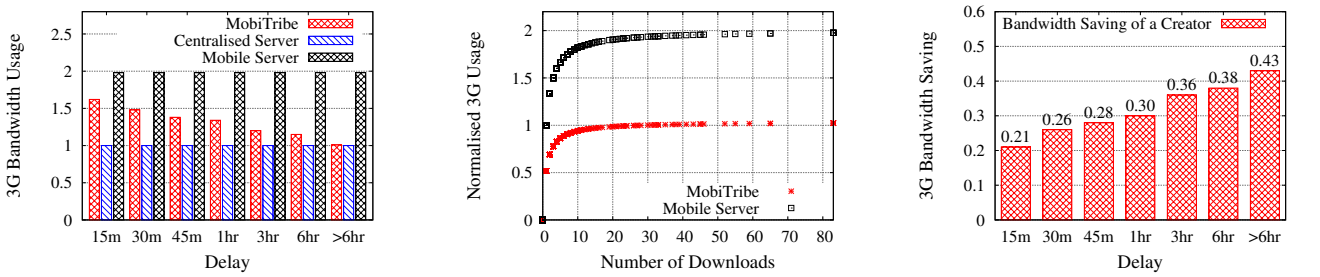
MobiTribe: The network usage models for MobiTribe is presented in Table 2. From previous analysis, we observed that $P_a = 0.98$, $R_a = 1.19$ and $W_a = 0.43$ for DS2 in the worst case. Since M_a is directly related to the delay tolerance as in Figure 6, we use the delay to represent M_a. We used a simple assumption that all downloads occur exactly after the delay and all consumers are selfish, i.e. the consumers will not share the downloaded content with others.

Centralised server: The content will be uploaded to a centralised server at the time of sharing and all consumers access the content through the server. From the same set of parameters used in the MobiTribe model, we can derive the 3G bandwidth usage of the centralised server system as $C^{3g}_{server} = (1 + N)(1 - W_a)F_s$.

Mobile server: Diaspora [9] allows users to host data on their own servers. The content will not be uploaded to a centralised server at the time of sharing. Instead, the content consumers access the shared content via creator's own server. Then, the 3G bandwidth consumption model would be $M^{3g}_{server} = 2(1 - W_a)NF_s$.

In all architectures, we consider that on-the-spot data offloading via WLAN is possible since majority of the modern smartphones are equipped with WiFi prioritization over cellular network. The high replication overheads of the non-mobile-optimised replicating systems such as SafeBook [5] and MyZone [13], would almost certainly lead to higher bandwidth consumption. This is due to the large number of replicas that would be needed to provide consistent availability of the content via low-cost networks when using a replication algorithm that is not aware of network availability patterns. Therefore, we do not compare MobiTribe against those architectures in this comparison.

Figure 7a shows that the normalised 3G bandwidth usage of MobiTribe decreases with content access delay because a higher delay allows more time to pre-distribute the content to the tribe via low-cost network connections. Note that we considered worst case content popularity model in terms of bandwidth usage, i.e. everyone in the group accessing all generated content. Even in the worst case situation, MobiTribe performs better than the mobile server system irrespective of the delay tolerance. When the delay is larger than 6 hours, i.e. $M_a \simeq 1$, MobiTribe 3G bandwidth consumption is almost equal to the centralised server system, whereas the mobile server system consumes twice as much bandwidth. To this end, in this particular trace environment, MobiTribe justifies its key design goal of cellular bandwidth saving, and stands out as a mobile opti-



(a) 3G bandwidth usage normalised by centralised server architecture. N=83-worst case content popularity

(b) Normalised 3G bandwidth usage against number of downloads. N ∼Pareto-(80-20 Rule). $M_a \simeq 1$

(c) 3G bandwidth saving of a content creator. N ∼Pareto-(80-20 Rule)

Figure 7: Comparison of 3G bandwidth usage of MobiTribe with other alternative architectures

mised distributed content sharing platform for online social networking from its other contemporary architectures.

However, user generated content popularity has a "long-tail" distribution as observed in the literature [4], i.e. the majority of the content is accessed by a very few consumers. Figure 7b shows normalised 3G bandwidth usage with a long-tail content popularity model, where N is distributed as a Pareto variable. MobiTribe consumes less bandwidth than the centralised server system in the majority of the instances, whereas the mobile server system performs better only when $N = 0$. To this end, MobiTribe consumes even less 3G bandwidth than a centralised server system in real situations by providing monetary cost benefits for the user.

Usually, in distributed content sharing the content is hosted by the creator or a group of users including the creator, which could possibly lead the content creators to consume more bandwidth than others. Therefore, we compare the 3G bandwidth consumption of content creators in MobiTribe and mobile server architectures. Figure 7c shows that a MobiTribe content creator can achieve 21%-43% of bandwidth saving compared to a Diaspora like content creator. Since more than 6 hours access delay could be a realistic assumption to be made [20], savings would be more than 43%. This would incentivise mobile users to use MobiTribe for distributed social networking, who would otherwise have been reluctant to do so because of the high monetary cost involved in the distributed architectures.

In all three architectures, 3G downlink usage $(1-W_a)NF_s$ is similar because it only depends on network availability of the consumers at the time of downloading. In future work, we intend to extend MobiTribe to replicate content where it is likely to be consumed. However, this approach requires modelling of user consumption and was left out of the scope of this paper. In addition, we are planning on extending MobiTribe to pre-distribute content created out side the MobiTribe system. The algorithm presented in this paper could be directly applied to this extension as well. This would allow us to further reduce the network utilization by a significant amount from downlink bandwidth usage.

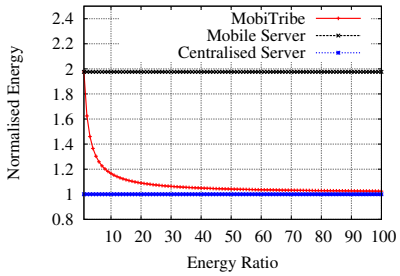
6.6 Energy Saving Comparison

We analyse MobiTribe energy consumption to compare it with alternative architectures. The energy consumption of WLAN, E_w, is significantly lower than of the 3G networks, E_{3g}, due to the heavy traffic load in cellular networks leading to lower speeds and therefore to longer durations of data transfers as observed by [2, 15]. The experimental study conducted by Petander [15] shows that E_{3g}/E_w can be varied up to 100 based on the network status at the time of transfer. Hence, a comprehensive energy evaluation would only be possible by an experimental implementation of MobiTribe. At this stage, we evaluate the energy consumption of MobiTribe and other alternatives with respect to the energy ratio E_{3g}/E_w and network usage models described in Section 6.5. We consider that the energy ratio is calculated after considering all the factors which affect energy consumption such as WLAN association energy, signal strength and data rate.

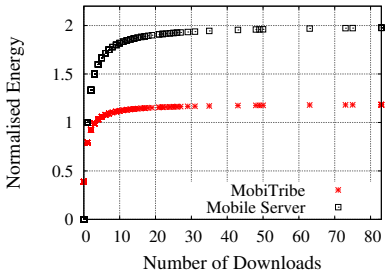
Figure 8a shows the energy consumption of each system, normalised by the energy consumption of a centralised server system for the worst case content popularity model. A mobile server system always consumes approximately 2 times more energy than a centralised server system, whereas MobiTribe energy consumption exponentially decays with E_{3g}/E_w and consumes almost the same energy as the centralised server system for higher ratios, even with the worst case content popularity model. The high overheads of the non-mobile-optimised replicating systems, such as SafeBook [5] and MyZone [13], would lead to higher energy consumption. To this end, MobiTribe does not consume extra energy, unlike other distributed systems in general.

Similar to 3G bandwidth comparison, we evaluated the two systems against realistic content popularity distribution. Figure 8b shows that MobiTribe consumes less energy when the number of downloads is smaller, even for $E_{3g}/E_w = 10$. Although, MobiTirbe does not consume any 3G bandwidth when $N = 0$, the creator still uploads the created content to the tribe via low-cost networks. Thus, energy consumption of MobiTribe is 0.4 times the centralised server system when $N = 0$, while mobile server based creators consume zero energy at the same time. However, even with this extra energy consumption associated with pre-distribution, a MobiTribe content creator will be able to gain 18%-41% energy saving over a mobile server creator as shown in Figure 8c. To make it more realistic, the average saving is obtained by varying E_{3g}/E_w from 1 to 100 and N as Pareto distribution.

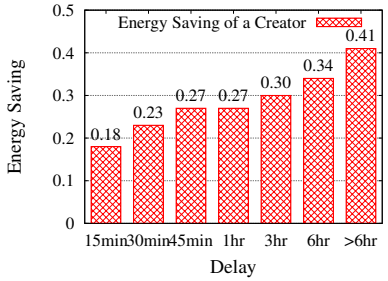
The energy consumption of a MobiTribe creator does not improve with the number of downloads since we used the simple assumption of selfish consumers. However, in reality the performance would keep on improving with the number of downloads spanning a longer period of time because there can be others who wish to share the content. Moreover, the proposed extension with predictive pre-fetching would reduce the energy consumption of consumers.



(a) Energy consumption normalised by centralised server architecture. N=83-worst case content popularity



(b) Normalised energy consumption against number of downloads. N ∼Pareto, $M_a \simeq 1$, $E_{3g}/E_w = 10$



(c) Energy saving of a content creator. N ∼Pareto-(80-20 Rule) and $E_{3g}/E_w = (1:100)$

Figure 8: Comparison of energy consumption of MobiTribe with other alternative architectures

7. CONCLUSION

Data ownership and privacy problems inherent in centralised approaches could be solved by using distributed social networking services. Smart phones could contribute to these services by hosting and sharing distributed social network data, if the associated high communication costs and battery usage were mitigated. We propose a novel mechanism to address these challenges by taking advantage of network availability patterns among groups of users to provide continuous availability of content over low-cost network connections. We show that the content replication problem in distributed peer-to-peer architectures such as MobiTribe is NP-Hard, but in case of our test data sets, it can be reduced to a P-hard problem. A new algorithm based on a combination of a bipartite b-matching and a greedy heuristic is developed, which minimises content replication and maximises content availability whilst ensuring fairness. The performance of the algorithm was evaluated using three empirical data sets and the results show that a persistent low-cost network availability of content can be achieved with only two replicas. In addition, the analysis of the system showed that a content creator can save up to 43% of 3G bandwidth and up to 41% of device energy on average compared to non-mobile-optimised distributed social networking approaches.

8. REFERENCES

[1] Crawdad:. In *http://crawdad.cs.dartmouth.edu*.

[2] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: a measurement study and implications for network applications. In *Proc. of the 9th ACM SIGCOMM IMC '09*, pages 280–293, Chicago, 2009.

[3] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput*, 2009.

[4] M. Cha, H. Kwak, P. Rodriguez, Y. Ahn, and S. Moon. Analyzing the video popularity characteristics of large-scale user generated content systems. *IEEE/ACM Transactions on Networking (TON)*, 17(5):1357–1370, 2009.

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

[6] H. N. Gabow. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In *Proc. of the 15th ACM Symposium on Theory of Computing*, pages 448–456, 1983.

[7] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum-flow problem. *J. ACM*, 35:921–940, 1988.

[8] B. Han, P. Hui, V. Kumar, M. Marathe, J. Shao, and A. Srinivasan. Mobile data offloading through opportunistic communications and social participation. *Mobile Computing, IEEE Transactions on*, (99):1–1, 2011.

[9] http://joindiaspora.org.

[10] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[11] K. Lee, I. Rhee, J. Lee, S. Chong, and Y. Yi. Mobile data offloading: how much can wifi deliver? In *Proc. of the Co-NEXT '10*, pages 1–12, Philadelphia, 2010.

[12] P. Lungaro, Z. Segall, and J. Zander. Activecast-a network and user aware mobile content delivery system. In *Proc. of ICUFN*, pages 309–313, June 2010.

[13] A. Mahdian, J. Black, R. Han, and S. Mishra. Myzone: A next-generation online social network. In *Tech Report: Dpt. of Computer Science, Uni. of Colorado, Boulder*, 2011.

[14] J. Mestre. Greedy in approximation algorithms. In *Proceedings of the 14th Annual European Symposium on Algorithms*, pages 528–539, 2006.

[15] H. Petander. Energy-aware network selection using traffic estimation. In *Proc. of the 1st ACM workshop MICNET '09*, pages 55–60, Beijing, 2009.

[16] J. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D. Epema, M. Reinders, M. Van Steen, and H. Sips. Tribler: a social-based peer-to-peer system. *Concurrency and Computation: Practice and Experience*, 20(2):127–138, 2008.

[17] S. Seong, J. Seo, M. Nasielski, D. Sengupta, S. Hangal, S. Teh, R. Chu, B. Dodson, and M. Lam. Prpl: a decentralized social networking infrastructure. In *Proc. of the ACM Workshop on MCS: Social Networks and Beyond*, page 8. ACM, 2010.

[18] P. Stuedi, I. Mohomed, M. Balakrishnan, Z. Mao, V. Ramasubramanian, D. Terry, and T. Wobber. Contrail: Enabling decentralized social networks on smartphones. *Middleware 2011*, pages 41–60, 2011.

[19] K. Thilakarathna, H. Petander, and A. Seneviratne. Performance of content replication in mobitribe:a distributed architecture for mobile ugc sharing. In *Proc. of IEEE LCN*, pages 558–566, oct 2011.

[20] I. Trestian, S. Ranjan, A. Kuzmanovic, and A. Nucci. Taming user-generated-content in mobile networks via drop zones. In *INFOCOM'11*, pages 2840–2848, 2011.