"Moving from Clouds to Mobile Clouds": **Cost-Efficient Privacy-Aware Mobile Content Delivery Mechanisms**

Dinesh Kanchana Thilakarathna

A thesis in fulfilment of the requirements for the degree of

Doctor of Philosophy

UNSW
AUSTRALIA

School of Electrical Engineering and Telecommunications
Faculty of Engineering

January 2015

Abstract

Smart mobile devices are being used to create, consume and share a variety of user data with others. This has led to a significant growth in the use of social networking services and services that support the distribution of user generated content. Service providers are making available more free services, as they are able to monetise user data. As a result, mobile data traffic is growing exponentially and placing heavy demands on mobile networks. This has a two-fold impact on users. Firstly, the communication costs of users are increasing. Secondly, the monetisation activities of service providers pose threats to user privacy and control of their data. Therefore, it has become vital to develop content delivery mechanisms that minimise communication costs, improve user privacy and provide more user control without breaking the existing digital services eco-system.

This thesis introduces three novel mechanisms referred to as User-Stash, MobiTribe and Yalut, which address the three issues: cost, privacy and control of data. It demonstrates the viability of those mechanisms through analytical and experimental evaluations. User-Stash demonstrates that it is possible to reduce mobile network traffic by exploiting the transient co-location of mobile users and spatio-temporal correlation of content popularity. User-Stash selects a set of devices to become crowd-sourced stashes that cache popular content for the benefit of others nearby. MobiTribe enables decentralised social networking services on smart mobile devices exploiting the time elasticity of social networking content and the fact that users will have access to high-speed low-cost networks such as WiFi, to effectively replicate user data on the devices of their trusted friends whilst minimising the communication costs as well as providing privacy and user control. Yalut augments MobiTribe to provide further cost benefits to users using opportunistic communication where the users are geographically clustered into communities. The novel solutions are based on a set of dynamic time-aware centrality metrics that identify most influential users to propagate content with minimum content delivery delay. In addition to analytical evaluations of the above mechanisms, the thesis demonstrates the practical feasibility of these mechanisms by developing mobile applications that can be used on commodity smart mobile devices.

Acknowledgements

The road towards writing this thesis would not be a reality without the direct and indirect support, guidance and encouragement of a number of people.

Above all, I offer my profound gratitude to my supervisor, Prof. Aruna Seneviratne, for giving me the opportunity to purse my PhD with him and for his boundless support from the first day I walked into NICTA. He always finds time to discuss my challenges, read and correct my work and guide me towards the right direction. This thesis would not have been possible without his support and encouragement.

My sincere gratitude extends to many others whom I worked with throughout my candidature. Specially, Dr. Henrik Petander and Dr. Dali Kaafar who supervised me during my stay at NICTA. I express my heartfelt gratitude to Dr. Aline Viana who was my supervisor during my internship at INRIA, France for her countless advices and support since then to-date. My gratitude extends to Dr. Julián Mestre from University of Sydney and Prof. Prasant Mohapatra from University of California, Davis for their support and guidance during the collaborative work throughout my candidature. I would also like to thank all researchers at NICTA for internally reviewing my papers and for their valuable advices. I also take this opportunity to thank all of my teachers from pre-school to university for their invaluable guidance and dedication.

I extend my gratitude to my fellow students, Suranga Seneviratne, Kamal Gupta and Fangzhou Jiang for their great support in discussing challenges and writing papers together. I highly appreciate the support given by Xinlong Guan, Abdul Karim, Sirine Marbet and Ji Zhao in implementing my proposals on real devices and the support given in experimental evaluations of the proposed methods. Special thanks go to the Network Research Group co-ordinator Prashanthi Jayawardhane for her endless support from travel arrangements to research equipment purchases. Moreover, I thank all students at the Network Research Group and NICTA staff who helped me in numerous ways. I also would like to extend my gratitude to all of my friends for their support and friendship.

The journey from a small town Anuradhapura in Sri Lanka, to finishing a PhD in Sydney, Australia would have been only a dream without the love and dedication

of my parents, who dedicated their entire lives to make my life a better one. I am forever grateful to them. My heartfelt gratitude extends to my sister for her loving support and my parents-in-law for their kind support and encouragement. I also thank my sister and brother-in-law for being with us for the last three years.

Finally and most importantly, I thank my loving wife Madhuka, who was with me every step of the way towards finishing my PhD, for her support, understanding and endless love and care.

Contents

Abbreviations

Ad	Advertisement appearing on mobile apps or web browsers.
AP	Wireless Access Points.
API	Application Programming Interface.
app	Mobile Application.
C-OSN	Centralised Online Social Network.
CAD	Content Access Delay.
CAD_{peak}	Mode of the Gamma distributed Content Access Delay.
CDF	Cumulative Probability Density Function.
CDN	Content Delivery Network.
CMS	Content Management Server.
Consumer	A mobile user who accesses and consumes, e.g. watch, any content.
Content	Digital objects, e.g. snapshots and videos.
Creator	A mobile user who initiates sharing and assumes to be the owner of the content.
D-OSN	Decentralised Online Social Network.
DHT	Distributed Hash Tables.
DL	Downlink - Service provider to mobile device.
EP	Evaluation Period.
FPR	False Positive Rate of content access prediction.
GSM	Global System for Mobile Communications.

HCMM	Home Cell community based Mobility Model.
HD	High-definition.
HDR	High-dynamic-range.
HTTP	Hypertext Transfer Protocol.
ID	Identifier.
ISP	Internet Service Provider.
LTE	Long Term Evolution.
MLE	Maximum Likelihood Estimation.
MP	Mega pixel.
MSN	Mobile Social Network.
mTribe	Mobile Private Storage Tribe.
NFC	Near Field Communications.
OSN	Online Social Network.
P2P	Peer-to-Peer.
PDF	Probability Density Function.
Propagator	A mobile user who disseminate content to other users nearby.
QoE	Quality of Experience.
RAM	Random Access Memory.
Replicator	A mobile user who keeps a copy of a content for the purpose of distributing it to others.
ROM	Read Only Memory.
SDK	Software Development Kit.
SSID	Service Set Identifier - “Network Name” of the WLAN.
THR_{host}	Limit of Hosting.
THR_{rep}	Limit of Replication.
TP	Training Period.

TPR	True Positive Rate of content access prediction.
UGC	User Generated Content.
UI	User Interface.
UL	Uplink - Mobile device to service provider.
UMTS	Universal Mobile Telecommunications System.
US	User-Stash.
US-app	User-Stash Mobile Client Application.
US-LAN	User-Stash Wireless Local Access Network.
US-server	User-Stash Mobile Server Application.
UX	User Experience.
VoD	Video on Demand.
WLAN	Wireless Local Area Network.

List of Figures

List of Tables

Publications

Journals

1. **K. Thilakarathna**, H. Petander, J. Mestre, and A. Seneviratne, "MobiTribe: Cost Efficient Distributed User Generated Content Sharing on Smartphones," *IEEE Transactions on Mobile Computing*, vol. 13, no. 9, pp. 2058–2070, Sep. 2014. (Impact factor $\sim$ 2.912)

2. **K. Thilakarathna**, A. Seneviratne, A. C. Viana, and H. Petander, "User Generated Content Dissemination in Mobile Social Networks through Infrastructure Supported Content Replication," *Special Issue of Mobile Social Networks in Elsevier Pervasive and Mobile Computing*, vol. 11, pp. 132–147, Apr. 2014. (Impact factor $\sim$ 1.677)

Conferences

4. **K. Thilakarathna**, X. Guan, A. Seneviratne, "Yalut: User-centric Social Networking Overlay," in *Proc. ACM/SIGMOBILE MobiSys'14 Demos*, Bretton Woods, USA, 2014, pp. 369–361.

5. **K. Thilakarathna**, F. Jiang, S. Mrabet, M. Ali Kaafar, A. Seneviratne, P. Mohapatra "Crowd-Cache: Popular Content for Free," in *Proc. ACM/SIGMOBILE MobiSys'14 Demos*, Bretton Woods, USA, 2014, pp. 358–359.

6. **K. Thilakarathna**, A. C. Viana, A. Seneviratne, and H. Petander, "Mobile Social Networking through Friend-to-Friend Opportunistic Content Dissemination," in *Proc. ACM/SIGMOBILE MobiHoc'13*, Bangalore, India, Aug. 2013, pp. 263–266. (Acceptance rate $\sim$ 15%)

7. **K. Thilakarathna**, A. Karim, H. Petander, and A. Seneviratne, "MobiTribe: Enabling Device Centric Social Networking on Smart Mobile Devices," in *Proc. IEEE SECON'13 Demos*, New Orleans, Jun. 2013, pp. 230–232.

8. **K. Thilakarathna**, "MobiTribe: User-centric Content Sharing Overlay for Online Social Networking," presented at the *IEEE iToF'13 Demos*, Sydney, Australia, 2013. (Best Demo Award)

9. X. Guan, **K. Thilakarathna**, S. Seneviratne, "ContextInfo: Configurable Data Collection Platform for Android Devices ," presented at the *IEEE iToF'13 Demos*, Sydney, Australia, 2013.

10. A. Seneviratne, **K. Thilakarathna**, S. Seneviratne, M. Ali Kaafar, P. Mohapatra, "Reconciling Bitter Rivals: Towards Privacy-aware and Bandwidth Efficient Mobile Ads Delivery Networks," in *Proc. IEEE COMSNETS'13*, Bangalore, India, Jan. 2013, pp. 1–10.

11. **K. Thilakarathna**, H. Petander, J. Mestre, and A. Seneviratne, "Enabling Mobile Distributed Social Networking on Smartphones," in *Proc. IEEE/ACM MSWiM'12*, Cyprus, Oct. 2012, pp. 357–366. (Acceptance rate $\sim$ 25%)

12. A. Seneviratne, **K. Thilakarathna**, H. Petander, D. Wasalathilake, "Moving from Clouds to Mobile Clouds to Satisfy the Demands of Mobile User Generated Content," in *Proc. IEEE ANTS'11*, Bangalore, India, Dec. 2011, pp. 1–4.

13. **K. Thilakarathna**, H. Petander, and A. Seneviratne, "Performance of Content Replication in MobiTribe: a Distributed Architecture for Mobile UGC Sharing," in *Proc. IEEE LCN'11*, Bonn, Germany, Oct. 2011, pp. 558–566. (Acceptance rate $\sim$ 29%)

Technical Reports

16. **K. Thilakarathna**, A. C. Viana, A. Seneviratne, and H. Petander, "The Power of Hood Friendship for Opportunistic Content Dissemination in Mobile Social Networks," INRIA-Saclay, France, Tech. Rep. TR7002, Aug. 2012.

Chapter 1

Introduction

Mobile phones are not anymore used for mere voice calls as it has been in the past. Today, they are *smart computing devices* equipped with advanced capabilities and come in different forms such as smartphones, tablets, notebooks and laptops. By the end of year 2013, it was estimated that there were 7 billions of mobile-connected devices globally and it is expected to surpass the number of people on earth by the end of 2014 [1]. These smart devices have been revolutionising human life style in many aspects and taking life experience way beyond expectations as a result of the various attractive services provided through smart mobile devices.

Pervasive use of smart mobile devices.

The technology has evolved not only in the power of processing, storage and connectivity, but also in sensing and capturing the behaviour of surrounding environment and the user. Table 1.1 shows the advanced capabilities of one of the modern smartphones. Apart from communication, modern smartphones are capable of providing services such as a camera, a music player, an internet browser, a navigator, a health monitor and many more services through millions of mobile applications, made available via app stores such as Apple App Store[1] and Google Play Store[2].

In essence, these state-of-the-art capabilities of smartphones completely changed the way people have been using mobile devices in the past. Today, it is possible to create ultra high-definition (HD) videos or capture 16MP HDR (High-Dynamic-Range) snapshots whenever and wherever you go, as a smartphone is always with you. Then, it is possible to share these with your friends through local short-range connectivity options such as Bluetooth, WiFi direct or usual internet connections via cellular or WiFi networks. Moreover, it is also possible to view these content on

[1] `http://www.apple.com/au/itunes/`
[2] `https://play.google.com/store`

Table 1.1: Technical specifications of a modern smartphone - Samsung Galaxy S5 [2]

Category	Capabilities
Processing	2.5GHz Quad-Core
Storage	2GB RAM, 16GB ROM and up to 128GB of external storage
Connectivity	Long-range: 4G LTE, 3G UMTS, 2G GSM , WiFi Short-range: Bluetooth, WiFi Direct, NFC
Sensing	Accelerometer, Gyro, Proximity, Compass, Barometer, Hall, RGB ambient light, Gesture, Fingerprint, Heart Rate Monitor
Creation	Main camera: 16 MP, auto-focus and flash, Front camera: 2MP Ultra HD (3840×2160 at 30fps) video
Consumption	Display resolution full HD (1920×1080), 16M colour depth Playback of various audio and video formats

full HD display whenever you want as these devices are practically *always-on* and *always-connected* to at least one of the available networks.

The ability to *create* and *consume* a variety of content, e.g. snapshots and videos, and the seamless ability to *share* content with other online users along with other contextual information such as location and activity of the users paved the way for the development and popularity of mobile apps for online social networking. There are a number of social networking applications available to-date, Facebook[3], Google+[4], Twitter[5], Snapchat[6] to name a few. There has been a significant growth in the use of social networking services, such as Facebook, and as well as services that support the distribution of user generated content (UGC), such as YouTube. In fact, the number of social network users has increased by 54% from 2011 to 2014 and all the predictions show that it will continue to grow [3]. More often than not, users have another life within these virtual social communities in the online world as they communicate and share every moment of life with online friends using these always-on and always-connected smartphones.

Consequences of explosive growth in the use of mobile social networking.

(1). As a result of the increasing usage of online content sharing services, users are loosing control over their own personal data because majority of the service are provided centrally. For instance, the current popular content sharing services, such as Facebook, Twitter, YouTube and Google+, use smartphone as a thin client with

[3] `https://www.facebook.com`
[4] `https://plus.google.com`
[5] `https://twitter.com`
[6] `http://www.snapchat.com`

all the data associated with the applications being centrally hosted on the servers of the service providers. Even though the cost of storage and distribution is high when using centralised architectures, the service providers prefer to host the service centrally as it allows the mining of user data to monetise user profiles from other sources, e.g. advertisements. Advantages for the user are; being able to access the content from any device and not having to worry about maintenance. However, the perceived advantages of free services come at the cost of losing control of their data and potential loss of their privacy. Majority of online users do not understand the consequences of the leakage of privacy and ownership of their data. Increasing usage of smartphones makes the problems associated with privacy and data ownership even more acute, because with all these sensing and capturing capabilities (Table 1.1), smartphone provides a significantly close representation of the personal behaviour of the owner.

(2). Mobile data traffic is growing faster than ever before. The data intensive social networking services such as UGC sharing and distribution is one of the primary drives of this exponential growth. In particular, mobile data traffic grew 81% during the year 2013 and mobile video accounted for 53% of the global mobile data traffic [1]. Moreover, Cisco forecasts 11-fold increase in mobile data traffic between 2013-2018 [4]. This massive surge of data traffic is placing heavy demands on the cellular networks. In addition, at peak times, this high demand introduces high latencies which reduces users quality of experience (QoE) [5].

Mobile operators are adapting to the higher traffic levels by both reducing the cell sizes to increase the capacity of current technologies and by upgrading their infrastructure with higher bandwidth technologies, such as 4G LTE. In the particular case of UGC, both the number of users uploading and the volume of content is growing faster than the ability of the operators to increase the capacity only using these traditional methods because they essentially attempt to provision their networks for the peak hour load which leads to having excessive resources during off peak hours. The problem is exacerbated because the peak hour traffic is growing faster than average traffic [4]. These additional expenses are being passed on to the users in the form of data usage based capped price plans. More often than not, users end up paying for extra data usage above the capped value at the end of the month.

Existing work on cost-efficient privacy-aware mechanisms.

There have been numerous alternative proposals for dealing with the increasing mobile data traffic by taking advantage of short-range communication among users

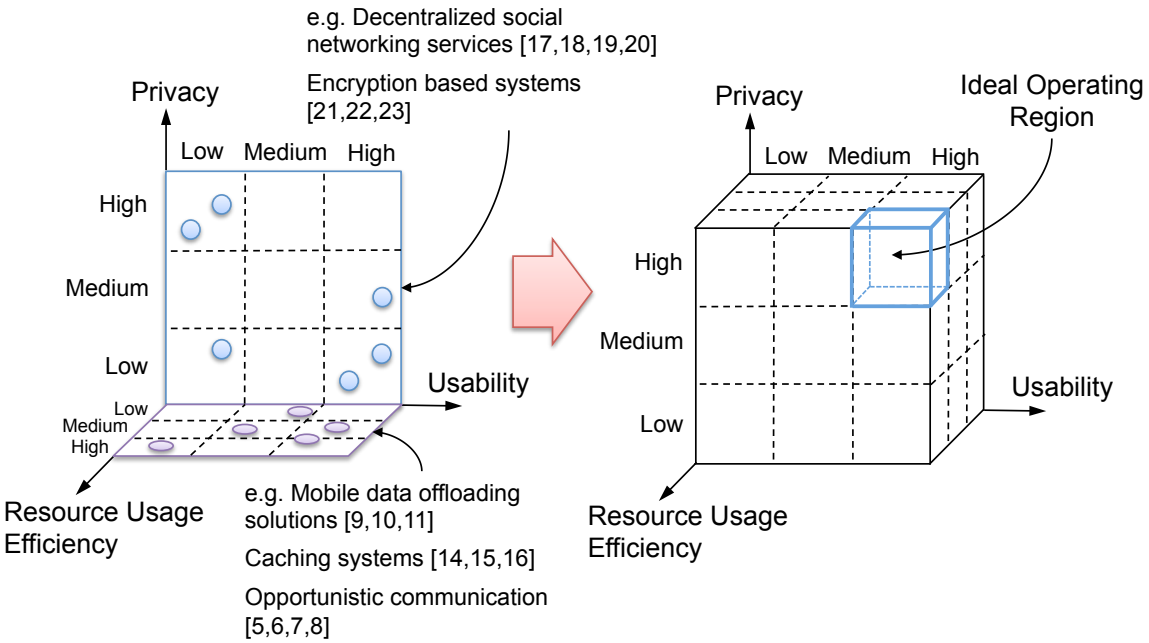


Figure 1.1: Categorisation of existing work and the solution space for new mobile content delivery mechanisms.

when they meet each other to exchange data (i.e. *opportunistic communication*) [6]–[9] or the availability of different type of networking infrastructure, such as WLANs [10]–[12] to transfer data through these alternative networks (i.e. *data traffic offloading*). Despite offering many advantages for mobile users, such as providing connectivity when there is no direct access to a network [13], opportunistic communication solutions have not seen wide spread adoption primarily for two reasons. Firstly, there is an inherent reluctance by the users to interact with strangers or third parties, due to security and privacy concerns despite it being partially addressed by the use of secure communication and storage methods through encryption. Secondly, due to the unbounded high delivery latency, i.e. being unable to access content when none of the couriers of content are in the vicinity when a user wants to access or share content. Traditional in-network *content caching* systems on the other hand are not effective as the delay and cost bottlenecks quite often occur on the last hop wireless link in mobile networks [14]. Likewise, local caching schemes including content pre-fetching [15]–[17] rely on the ability to predict user interests, which is difficult, prone to failure and can lead to loss of privacy.

Opportunistic communication and data traffic offloading schemes however do not address the user related issues of loss of control of user data or the loss of privacy of the user. On the other hand, the proposals aimed at addressing the issues associated

with privacy and the user losing control of their data, *decentralised social networking architectures* [18]–[21], disregard the mobile network related issues and lead to an increase in data traffic, if the devices being used are mobile. None of the previously proposed decentralised social networking services have seen widespread use mainly for three reasons. Firstly, because of most users have already subscribed to the popular centralised services. Secondly, because decentralised social networking services are too complicated and expensive for a typical user. Thirdly, because users are not able to migrate from their current services, due to personal data being locked in with the subscribed centralised services. In addition, data encryption and ephemeralness are two other mechanism that have been used in the literature to realise the privacy-aware social networking on top of centralised servers [22]–[24]. Similar to other privacy-aware methods, these also do not address any issues associated with costs of communications.

It is possible to categorise all proposed systems and mechanisms based on the level of usability, resource usage efficiency and privacy provided as shown in Figure 1.1. Majority of the work to date focused on either privacy-usability plane or resource usage efficiency-usability plane. For instance, the proposed decentralised social networking services belongs to the privacy-usability plane because they are not optimised for resource constraints in mobile networks as discussed in Section 2.3. In contrast, majority of data offloading and content caching solutions do not mitigate any privacy related issues associated with mobile content delivery systems. As a combination of these factors, we have not yet seen wide spread adaptation of any solution in real practical mobile content distribution networks, except for on-the-spot data traffic offloading to WiFi networks. On-the-spot offloading is essentially a resource efficient scheme which does not pose additional privacy and usability issues.

To this end, the primary objectives of this thesis are to propose, evaluate and validate novel mechanisms considering all three factors of usability, resource usage efficiency and privacy whilst taking the solutions as close as to the ideal operating region as shown in Figure 1.1. The idea is to harness the advanced capabilities of mobile devices and pervasive availability of different networks. Smartphones and tablets of today have sufficient computing power, storage capacities and networking capabilities to provide services as a networking entity and a content store. Furthermore, users of these devices are likely to be the biggest consumers and producers of content. Therefore if the capabilities of these device could be leveraged, they could form the platform for providing data intensive rich media services such as social networking and UGC sharing without the drawbacks of communication costs and loss of privacy.

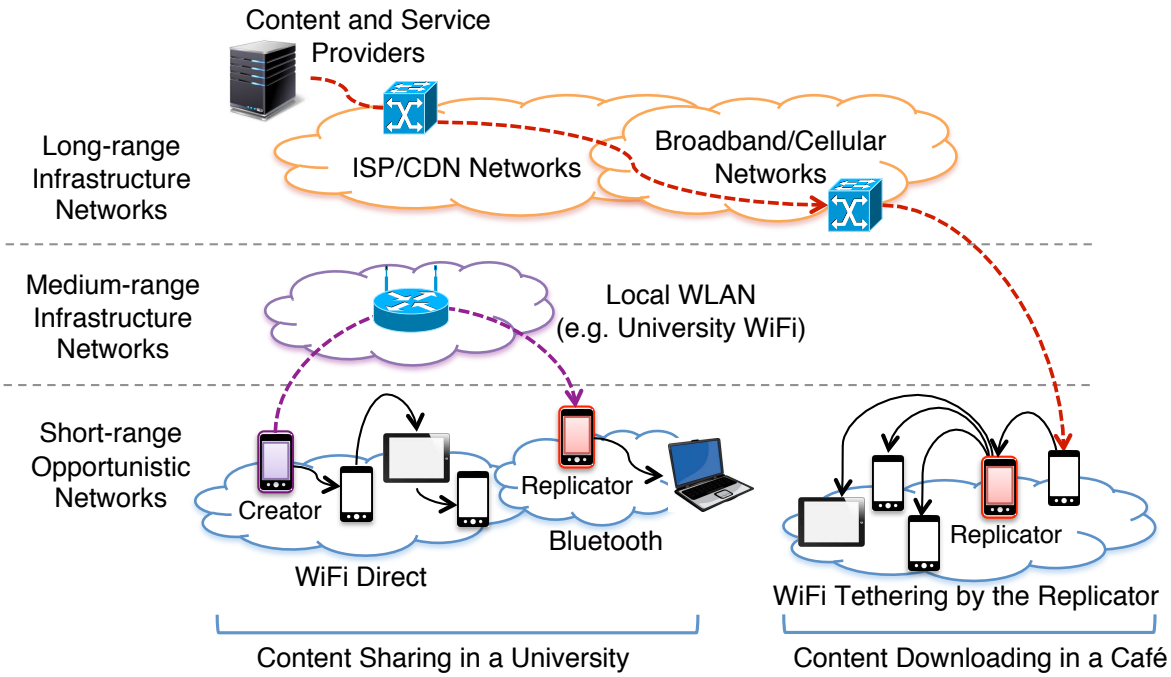


Figure 1.2: Taking advantage of heterogeneous access networking technologies and advanced capabilities of mobile devices.

1.1 Our Approach - Scenarios

Even during peak hours there are many locations and other networks such as WLANs that have unused network capacity. Moreover, there are scenarios where users in the same location will have similar interests that are location specific, e.g. shoppers in the same mall, spectators in a sports event, students in a university, etc. In these scenarios, it is possible to exchange content using short-range networking technologies such as WiFi Direct, Bluetooth and WiFi Tethering, which more often incur lower cost than cellular networks. Therefore, if it is possible to shift some of the traffic from peak hours of the cellular networks to less loaded networks, or delay the transmissions until off peak times, the peak load can be significantly reduced [10]. A smart mobile device, if used as an intelligent networking entity combined with applications exploiting the availability of heterogeneous networks, it is possible to enable this shifting of traffic either to off-peak times if the traffic is delay tolerant or to lower cost networks if low-cost networks are available. Moreover, if a smart mobile device is used as a content store, it is also possible to share content directly from one friend's device to another friend's device without storing the content in any centralised location providing the user more control of users' private data and preserving users' privacy.

Content downloading. Assume a mobile user who is sitting in a cafe downloads a popular music video clip from one of the current centralised content sharing services, e.g. YouTube. Traditionally, even if another user, who is in the same cafe requests the same video clip, the second user is asked to fetch it from the centralised servers of the service, e.g. YouTube servers, without taking advantage of the video clip which is already fetched to a device in the same location. If it is possible to make the downloaded video clip available to the users in the same geographical area either by forwarding the content to others or pushing the content to a local storage cache, it will be possible to reduce the cost of capped data communication for other users with similar interest. The idea is to disseminate the content fetched to mobile devices via lower cost short-range networks, e.g. WiFi Tethering, wherever and whenever possible as shown in Figure 1.2. Due to the higher capacity of the local networks, latencies will be minimised and as a result, the user QoE will be improved in addition to the monetary gains of not using the cellular network. Some mobile users may cache the content for the benefit of others acting as content *Replicators* taking advantage of advanced processing, storage and networking capabilities of smart mobile devices.

Content sharing. Even in the case of sharing personal content with a perviously known set of friends, e.g. social networking friends, it is possible to take advantage of storage and networking capabilities of the mobile device rather than uploading the content to a centralised UGC sharing service. Consider a university student, a content creator, who took an interesting video clip from his smartphone and wants to share it with his friends. Imagine that a set of creators's friends are already with her/him in the same location. Even in that case, the current sharing services request the creator to upload the video to their servers without taking advantage of other available local networking options. Assume there is a new mobile app that empowers the smart mobile device to identify the availability of the friends' devices in the communication range of each other and enables device-to-device content transfer using short-range networking technology such as WiFi Direct as shown in Figure 1.2. In this scenario, the content creator and the friends in the vicinity reduce the use of capped cellular network data. Moreover, the creator keeps more control over his video clip as it is not stored in any third party service provider preserving user's privacy as well. In the case of sharing with other friends who are not in the same geographical area, it is possible to use the lowest cost network available to the user to transfer the content. In the scenario of Figure 1.2, the creator's mobile device takes advantage of university WLAN network to transfer the video clip to another friend. Furthermore, it is possible to pre-distribute the content to mobile devices of some

friends, who could deliver the content locally to other friends acting as *Replicators*.

In this thesis, we investigate how to realise these advantages by exploiting the advanced capabilities of mobile devices and mobile user behavioural patterns. We hypothesise that;

- Smart mobile devices have sufficient spare storage capacity or the capacity can be upgraded in selected devices to cache or replicate content and also there is a sustainable incentive scheme to motivate mobile users to collaborate with each other and such incentive schemes can be developed.

- There is a transient colocation of mobile users and the spatial temporal correlation of content popularity, where users in a particular location and at specific times would likely be interested in similar content. Mechanisms can be developed to effectively identify the existence of required content if it is cached in the local network or download it from the custodian if the content is not cached, without affecting the usability of the service.

- Majority of social networking friends are clustered into geographical communities and these communities can be predicted in advance for the purpose of content replication. Then, there is a content replication algorithm which maximises the availability of content via low-cost networks without exhaustively using the resources of mobile devices. Finally, it is possible to develop mechanisms that provides the benefits of the decentralised content sharing without negatively impacting the communications and energy costs.

- Content sharing methods that co-exists with the already available social networking eco-system can be developed combining the advantages of decentralised content sharing and opportunistic communication such that the content will be delivered to all users in timely manner via friend-to-friend opportunistic dissemination.

1.2 Main contributions of the thesis

To validate the above hypotheses, this thesis introduces three mobile content delivery methods which harness the advanced capabilities and heterogenous connectivity of mobile devices for efficient mobile content delivery. The viability of the methods are demonstrated in three common application scenarios. In essence, the proposed mechanisms exploit the capability of content caching at the end user mobile de-

vices thereby creating groups of mobile devices as mobile clouds for each particular application. The main contributions of this thesis are concisely presented below.

- **User-Stash** enables mobile users to access popular content without using their cellular data plan at locations where there are no low-cost networks such as WiFi. User-Stash is an alternative approach that exploits the transient co-location of mobile users and potential spatio-temporal correlation of content popularity. The rationale of the proposed system is inspired by the delivery of information via the free newspapers in public transportation systems in major cities around the world, where users consume content (reading the paper) whilst travelling but leave the content (the paper) when they leave. The development of User-Stash system makes the following contributions:
  - The use of crowd to stash and distribute geographically popular content locally to the users in the vicinity, exploiting the transient co-location and spatio-temporal correlation of content consumption patterns of mobile users.
  - Eliminates the need for prediction of user demand as well as low-cost networks, while exploiting the spare storage capacity on mobile devices and the capabilities to host a local WiFi network to create a new low-cost network for all the users in the vicinity.
  - The method of downloading the content via one network (cellular) and uploading to a stash via a different network (WiFi), which is fundamentally different from an in-network cache system.
  - Presents an in-app advertisement based incentive scheme to encourage users to become User-Stashes.
  - Models the video content access and the corresponding content consumption patterns of mobile users followed by the behaviour of User-Stash system in a public transportation system using a unique large real-world data set from a popular video content provider.
  - Shows that User-Stash reduces the cellular network usage of passengers during an average city bus commute using the proposed models.
  - Demonstrates the feasibility and practicality of the proposed system in terms of energy consumption and data transfer latencies from the measurements of an implementation on real devices.

- **MobiTribe** provides decentralised social networking services using smartphones without negatively impacting the communication costs and battery usage of mobile devices. MobiTribe exploits the time elasticity of social networking content and the fact that the user will have access to high speed low-cost networks such as WiFi to effectively cache user data on mobile devices. The main contributions of MobiTirbe are as follows:
  - The formation of Mobile Private Storage Tribe (*mTribe*) using the mobile devices of groups of friends, which stores and distributes user data whilst improving user privacy and providing the user more control over their own data.
  - Proves the content replication problem in distributed peer-to-peer architectures to be NP-Hard and presents a novel scalable algorithm for content replication based on a combination of a bipartite b-matching and a greedy heuristic, which minimises content replication and maximises content availability whilst ensuring fairness.
  - Shows the viability of the proposed algorithm using real-world data traces of low-cost network connectivity of mobile users, namely that it is possible to achieve persistent low-cost network availability with only two replicas.
  - Benchmarks the performance of MobiTribe against alternative social networking approaches and show that MobiTribe reduces both uplink and downlink cellular bandwidth usage compared to current widely used centralised server based architectures.

- **Yalut** augments MobiTribe to provide further cost benefits to users of location-based mobile social networks (MSN) where the users are geographically clustered into communities by combining the advantages of distributed decentralised storage and opportunistic communications. Such approaches by themselves are not new, however, the proposed hybrid time-aware method of combining them, considering the initial encounter time and duration of users' encounters is novel. Yalut makes the following contributions:
  - Shows the inefficiency in opportunistic content dissemination in MSNs through real-world and synthetic trace driven simulation study.
  - Provides a formal definition of content replication which maximises content delivery success rate and minimises replication in opportunistic content dissemination and shows this to be NP-hard.

- Presents a community based greedy algorithm for efficient content replication which takes advantage of routine behavioural patterns of mobile users. It uses a set of dynamic time-aware centrality metrics to identify most influential users within a community, based on a dynamic weighted contact graph composed of opportunistic user encounters with contact duration and initial contact time information.

- Shows that random content replication provides comparable results and that it can be exploited to develop a component based random content replication algorithm.

- Shows that Yalut reduces cellular network usage and increases opportunistic content delivery success rate using a human mobility simulator and through extensive trace driven simulations.

- Moreover, the practical feasibility of User-Stash, MobiTribe and Yalut mobile content delivery mechanisms were demonstrated through prototype implementations on real mobile devices.

  - User-Stash client and server side applications were developed on Android smartphones. Although User-Stash system concepts are valid for any popular content type, this implementation focuses on the distribution of video content, as mobile video would account for the majority of mobile data traffic in future. The app is integrated with YouTube and Dailymotion video content distribution services. The app will be made available for general public in future via the website `http://www.userstash.com`.

  - MobiTribe and Yalut have been integrated together as a unified mobile app, which is integrated with popular social networking services Facebook, Google+ and Twitter. Seamless integration of Yalut with existing centralised social networks overcomes usability drawbacks of previously proposed decentralised social networking services. For connivence of the user, Mac and Windows software are also developed and all the Yalut client applications have been released for public use through the website `http://www.yalut.com` and Google Play Store[7].

[7] `https://play.google.com/store/apps/details?id=com.yalut&hl=en`

1.3 Organisation of this thesis

This thesis is divided into chapters based on contributions. The organisation of the remainder of the thesis and the contents of each chapter are briefly described below:

Chapter 2 presents a comprehensive review of the background and the related areas of research. Previously proposed mechanisms are categorised into *content caching*, *data traffic offloading* and *privacy-aware social networking*. Then, we compare and contrast these existing mechanisms with the mobile content delivery mechanisms proposed in this thesis providing the context for the three mechanisms presented in the following chapters.

Chapter 3 introduces the three mechanisms, User-Stash, MobiTribe and Yalut and paves the way for describing the content delivery mechanisms that are presented in the following three chapters.

Chapter 4 provides the details of the User-Stash content delivery mechanisms. It models the operation of User-Stash in a public transportation system and evaluates the performance in terms of resource usage efficiency and usability.

Chapter 5 describes the mechanisms of MobiTribe and introduces a novel scalable content replication algorithm. Then it evaluates the performance of MobiTribe using real-world data sets and realistic content creation and consumption modelling.

Chapter 6 presents the content delivery mechanisms of Yalut by introducing infrastructure based content replication algorithms to improve the opportunistic content dissemination. Then it evaluates the performance using both synthetic and real-world traces with realistic content creation and consumption modelling.

Chapter 7 shows how the proposed mechanisms can be realised on real mobile devices by implementing two mobile systems, namely, User-Stash and Yalut.

Chapter 8 concludes the thesis summarising the investigations carried out throughout the thesis. The main contributions of the thesis are recaptured and recommendations for future research are also presented.

Chapter 2

Related Work

Advanced capabilities of smart mobile devices such as the seamless ability to create, share and consume rich media content have led to the exponential growth in the use of social networking services and UGC sharing services. Service providers are making available more free services, as they are able to monetise user data. Therefore, mobile users are compromising their privacy and loosing control over their own data by using these free centralised social networking services. As a result of these data incentive usage of smart mobile devices, mobile data traffic is increasing dramatically and placing virtually unsustainable demands on the mobile operators [1]. At peak times, these high demands introduce high latencies which in turn reduces users QoE [5]. The predictions are that the demand will outpace the increases in capacity that will be provided by new technologies such as LTE [25]. Therefore, data usage based capped price plans are already adopted by the operators to shape the demand.

This has led to significant research efforts to develop techniques for minimising the cellular network data traffic and improving user QoE. This work broadly falls into two areas, namely *content caching* and *data traffic offloading*. As a separate body of work, there have been a number of proposals for *privacy-aware social networking* aimed at addressing data ownership and privacy problems associated with centralised approaches. This chapter presents a comprehensive review of related work in these areas to provide the context for novel mobile content delivery mechanisms presented in this thesis.

2.1 Content Caching Schemes

Content caching is a well studied and established technique, which is being used by many online content delivery services. Since all popular social networking and UGC sharing services use a centralised architecture, all mobile users are required

to download all content that they are interested in from the servers of the service providers. These servers are usually connected to tier-1 Internet Service Provider (ISP) networks. More often than not these services use Content Delivery Networks (CDN), e.g. Akamai[1], to bring the content as close as possible to the user using in-network caching. Even though content caching can be done at any location of the networking infrastructure starting from tier-1 ISP network to a personal end user terminal, traditional CDNs caching occurs at the backhaul networks. However, traditional in-network caching systems are not effective as the delay and cost bottlenecks are quite often occur on the last hop wireless link in mobile networks [14]. Therefore, there is no benefit apart from the potential improvement in QoE due to the reduction in latency as the data being closer.

This has led to moving the locations of caching further down the network hierarchy as close as to the end user terminal by taking advantage of the advanced capabilities of new user devices and availability of heterogenous access network technologies. This work broadly falls into three categories: caching at the edge of the network, caching for personal use and cooperative caching at mobile devices.

2.1.1 Caching at the edge of the network

There are many proposals for caching at the edge of the network, i.e. at Access Points (APs) and/or Base Stations [26], [27]. Mashhadi et al. [26] propose opportunistic proactive pushing of content to the mobile device through dedicated APs that do proactive caching. This requires however the availability of APs with internet access. VideoFountain [27] deploys kiosks at popular venues to store and locally distribute content. The primary aim is to cache messages that users generate at different locations where the kiosk is located, which are then made available to users in the vicinity of a kiosk. Thus the system, similarly to opportunistic networks, relies on human mobility to carry content between kiosks. Erman et al. [28] propose a cost-benefit trade-off model to investigate the caching benefits at different levels of a cellular network. However as mentioned earlier, caching at the base stations is not effective as the bottlenecks quite often occur on the last hop wireless link in mobile networks.

None of these strategies are easily deployable due to resource constraints such as power, storage, installation and security concerns when compared to in-network caching at core network servers. In addition, most of these proposals, if not all, require hardware changes or support from additional infrastructure to cache con-

[1] `http://www.akamai.com`

tent. In contrast, the mechanisms proposed in this thesis such as User-Stash do not necessarily require fixed APs or additional infrastructure to cache content, neither do they require internet connectivity from the APs, nor the users explicitly carrying information. Instead, User-Stash rely on the transient colocation of mobile users and the spatio-temporal correlation of content popularity.

2.1.2 Caching for personal use

Similar to in-network caching, caching at the end-user devices (e.g., smartphones) has also been studied previously [15], [29]. Qian et al. [15] show that there are 17-20% redundant data transfers on cellular networks as a majority of current mobile web applications under-utilise the caching capabilities. In [29], the authors focus on the QoE improvement of web browsers and show that 60% of the requests can be served by a browser cache of only 6MB.

IncomingTV[2] Android app that predicts the user interest based on previous usage and pre-fetches the videos to the mobile device when it is connected to WiFi networks, has been able to attract 1-5 million users highlighting the need for such apps that provides cost benefits for mobile users. Predicting user consumption of content and pre-fetching the content by caching it on the user's device to minimise the network congestion and cost, has been also evaluated in [16], [30], [31]. The effectiveness of content pre-fetching heavily relies on accurate prediction of future demand and the ability to find uncongested and lower cost networks to pre-fetch the data.

However, it has been shown that accurately predicting user behaviour and the network availability is challenging [32], which is even reflected in the user comments of the IncomingTV app. In addition, it also raises numerous privacy issues due to the prediction of future user behaviour as well as the amount of information these services require to perform the prediction [33]. In contrast, all mechanisms proposed in this thesis cache content and make it available to other nearby users, which is in line with the studies that argue in favour of exploiting redundant data transfers ([15], [29]).

2.1.3 Cooperative caching at mobile devices

All three mobile content delivery mechanisms proposed in this thesis, cooperatively cache content at mobile devices for the purpose of disseminating the content to

[2] `https://play.google.com/store/apps/details?id=com.incoming.au&hl=en`

others in the vicinity locally or to others not in the vicinity via via low-cost networks. There have been proposals of hybrid content dissemination systems content is replicated in a selected set of users. Han et al. [6] proposed a target set selection for content replication using cellular networks followed by propagation of the content with opportunistic communication. The focus of their work is dissemination of data to and from a centralised data storage. Ioannidis et al. [8] proposed a distributed caching mechanism for the purpose of social welfare where users cache content downloaded through the networking infrastructure. Similarly, Whitebeck et al. [9] proposed a hybrid content delivery system with a control loop through which users send acknowledgements of delivery to the central service provider. VIP delegation [7] replicates data using networking infrastructure on a few "socially important" (VIP) users in a mobile network. VIPs in turn distribute the content to other users opportunistically. However, the content delivery delay is quite large, which is unacceptable in MSNs, because of the large geographical area and only one-hop opportunistic propagation. Furthermore, in [7] the metrics used to select devices for replication do not consider the dynamic aspects of contact time and duration of users. Taghizadeh et al. [34] present a social community based cooperative caching system. It is aimed at minimising the cost of content distribution to users with common interests that are physically co-located. All users cache content and distribute via an ad-hoc network. In [35], the authors have analysed the effectiveness of temporal communities in dissemination of content opportunistically, i.e. at a given time users gather together forming communities. The results show that the users with high contact rates that are truly mobile are mostly responsible for opportunistic content dissemination. Similarly, Contentplace [36] proposes to use as "content transporters", the users who in the future will be in touch with the majority of the users, according to their social behaviour.

However there is no proposal to effectively use these temporal communities for content dissemination which is addressed in this thesis. Moreover, only using such users with high contact rates may not be effective due to very limited time windows and geographical space of content delivery in MSNs. The effectiveness of such opportunistic content dissemination is further evaluated in Section 6.1, which had led to the content replication based content delivery mechanisms proposed in this thesis. All these cooperative caching methods suffer from the variable delays, higher energy consumption and the trust, security and privacy issues of opportunistic networks, due to the required interactions with previously unknown people who are in the vicinity. Neither of the proposals discuss the impact of collaboration and willingness of users to replicate content, nor to access content using networking in-

frastructure. In contrast, the mechanisms proposed in this thesis provide the user of the device control as to whether to replicate others' content based on availability of storage, battery power, etc. There are number of opportunistic routing protocols which propagate messages from a source user to a destination user using cooperative caching at mobile devices [37], [38]. Whereas we make UGC available to users who are interested in the content locally. User-Stash users only need to trust the User-Stash server and MobiTribe and Yalut users only exchange information with other users who are previously known as friends. Furthermore, the mechanisms proposed in this thesis do not introduce any significant energy costs or delays. In fact, we show the proposed mechanisms improve the user QoE due to the higher data transferring rates of short-range networks.

2.2 Data Traffic Offloading

2.2.1 Communication cost optimisation

In ActiveCast [39], the content is pre-fetched and cached in different locations including end user terminals via low-cost networks. ActiveCast does not address uploading or hosting of UGC, and relies on accurate prediction of future demand [39]. Lee et al. [10] presents a quantitative analysis of offloading 3G data traffic to WLANs. Using data traces from 96 iPhones, the authors show that it is possible to offload 65% of mobile data traffic to WLANs and the gain could be increased by up to 30%, if the data transfer could be delayed for more than one hour. The authors further illustrate in their work that 55% of battery power can be saved by offloading via WLAN. Wiffler [11] augments mobile 3G network capacity, taking advantage of delay-tolerant applications. It predicts availability of WLAN in the near future and delays the transmission for the purpose of offloading data transfers to the potential WLANs that will become available in the future. The limitation of their work is that the predictions are done only for up to 100 seconds and the delay tolerance is only 60 seconds in most cases. Predicting near future WLAN connectivity and quality is addressed also in Breadcrumbs [12] to improve cost efficient scheduling of the data transfers. However, in this thesis, the focus is on UGC sharing and distribution, which can usually accommodate larger delay tolerance. Further, none of these mobile data offloading schemes address the issues of loss of control of data or user privacy.

Predicting user consumption of content allows pre-fetching of content to the device, when the user's device has good connectivity, i.e. WiFi or an unloaded cellular

network [40]. This reduces the monetary and energy costs of delivering the content and improves the user experience compared to streaming of content over loaded networks. Even though it is difficult to predict the content consumption of a user with a high degree of accuracy, relatively low levels of accuracy can lower the cost of the content delivery compared to streaming over cellular networks. For example, the energy consumed to download content will dependent mostly on the time spent downloading it. Thus, by using unloaded cellular or WiFi networks, the time spent downloading the content could be reduced by a factor between 10 and 100, when compared to heavily loaded cellular networks [41]. Further, the user experience of the content viewing would improve dramatically in terms of both the startup delay[3] and the re-buffering, which caused the stalling of 5-40% of videos, depending in on the operator and time of day [42].

2.2.2 Device energy optimisation

Balasubramanian et al. [43] derived an energy consumption model for data transferring over each networking interface (3G, GSM and WiFi) through experimental studies and proposed an energy efficient data transferring protocol named TailEnder. In 3G and GSM networks, there are three energy states: 1) Ramp energy: energy required to switch to high power state, 2) Transmission energy: energy consumed during data transfer, and 3) Tail energy: energy consumed just after data transfer in high power state. In WiFi, the energy is quantified as: 1) Association energy: energy required to scan and associate with an access point, and 2) Transmission energy. The authors conducted an experimental study for different data sizes (1 to 1000KB) with varying intervals in different locations and days. Table 2.1 tabulates the energy consumption model derived by Balasubramanian et al. where R(x) denotes the sum of Ramp and Transfer energy. According to the results, nearly 60% of 3G energy is wasted as Tail energy after the data transfer. GSM is the most energy efficient data transferring method for small sized transfers. The authors show that WiFi is more energy efficient than 3G even with the association overhead of WiFi. However, these results are dependant on the data rates of the networks. The energy model Table 2.1 can be used to investigate energy consumption patterns of mobile content delivery systems similar to the proposed systems in this thesis.

In energy efficient data transferring protocol TailEnder, the data transfer is delayed for delay-tolerant applications such as emails and aggressively pre-fetches the

[3]We compared playback of pre-fetched video on an Android device from local storage with streaming over loaded and unloaded 3G cellular, and the startup delays were 0.5s, 7s and 25s

Table 2.1: Energy model for downloading x bytes of data over 3G, GSM and WiFi networks. All values except the Maintenance values for 3G and GSM, are averaged over more than 50 trials [43].

	3G	GSM	WiFi
Transfer Energy - R(x)	0.025(x)+ 3.5	0.036(x)+ 1.7	0.007(x)+ 5.9
Tail energy - E	0.62 J/sec	0.25 J/sec	NA
Maintenance - M	0.02 J/sec	0.03 J/sec	0.05 J/sec
Tail time - T	12.5 seconds	6 seconds	NA
Energy per 50KB transfer	12.5 J	5.0 J	7.6 J

predicted content for applications such as web browsing. For delay-tolerant applications, content is delayed up to a user specified period and transferred in bulk, which reduces the tail time compared to several individual transfers. The authors suggest that this strategy saves half of the total energy consumption. Pre-fetching has to be managed carefully to reduce the energy wastage of unwanted pre-fetching. Since WiFi interface is energy efficient than 3G interface, the device is switched from 3G to WiFi whenever a WiFi network is available. Therefore, TailEnder enabled mobile device uses three times less energy compared to a device that use only 3G networks. Although the primary goal of TailEnder is mobile energy saving, its results justify the rationale behind the proposed mechanisms in this thesis. However, it is not possible to fully evaluate the effectiveness of data traffic offloading as there is no experimental evaluation of 3G traffic offloading of TailEnder users.

Rahmati et al. [44] proposed a ubiquitous energy-efficient wireless connectivity scheme based on the device context information. The authors propose to accurately estimate the future WiFi network conditions based on context information before powering up WiFi interface. The proposed prediction algorithm is claimed to able to predict WiFi connectivity patterns more than 10 hours ahead, with 95-90% accuracy. The model considers context information such as time, history, cellular network conditions, and device motion to find the option for transferring the data for a given user, which minimises the energy consumption. Initially, acceleration estimation is used to determine the validity of the previous condition and if it is not valid, the cellular network ID or fingerprinting estimation is used to predict future network conditions. Fingerprinting estimation uses an ordered set of visible cell towers at each location, i.e. prior training at each location. This limits its practicality despite its accurate predictions. However, this study proves that if there are enough information available, it is possible to predict connectivity patterns over a long time frame. The experimental study similar to TailEnder [43], shows a 30-40% saving

in battery life. An energy-delay trade-off in UGC sharing is analysed by Ra et al. [45]. They proposed an online link selection algorithm called SALSA based on the Lyapunov optimisation framework, which essentially delays the data transmission decision based on the factors: data backlog, power cost and channel quality. A trace driven simulation results show that SALSA enabled smartphone can save 10-40% battery capacity compared to a scheme that does not trade-off increased delay.

All these approaches are optimised for energy conservation and indirectly offload cellular network data traffic to other networks. Similar to mobile data traffic offloading approaches, none of these systems address the problems associated with user privacy. However, these systems could be incorporated with the proposed content delivery mechanisms in this thesis to get similar or even greater energy savings in content dissemination using commodity smart mobile devices.

2.3 Privacy-Aware Social Networking

There have been a number of proposals aimed at addressing issues of privacy and the user losing ownership of data in online social networking [18]–[20], [46]. These proposals either provide the users with a capability to host data on their own personal containers or exploit the restricted access capabilities of data encryption. This has led to the development of decentralised online social networking (D-OSN) architectures. The location of the personalised containers can vary from being stored in users' personal devices to the cloud.

2.3.1 Peer-to-peer systems

Peer-to-peer protocols. Peer-to-peer content sharing systems have been proposed more than a decade ago [47]–[50], a well before the concept of online social networking. BitTrorrent [49] has become the most widely used peer-to-peer protocol. It has been adapted to suit a number of different application scenarios[4]. All peer-to-peer systems employ some form of storage redundancy technique to provide sufficient availability of the content. In systems such as Napster [47], Kadmelia [48] and BitTorrent [49], data replication occurs implicitly as each file downloaded by a user is replicated at the user's device. PAST [50] replicates at the closest locations according to node-IDs. However, these techniques do not explicitly manage replication, and the required redundancy (number of replicas) is not closely coupled

[4]BitTorrent varieties: `http://www.vuze.com`, `http://www.tranmissionbt.com`, `http://www.libtorrent.org`, etc.

with the required level of availability which could lead to overestimating the needed redundancy. Moreover, at the time of BitTorrent was designed and deployed, mobile devices were not as popular as today. Thus the system mechanisms were not optimised for resource constraints in the mobile systems.

Distributed storage and sharing systems. With the increasing popularity of social networking services and thereby the discussions of associated privacy and security concerns have led the development of distributed and decentralised services primarily using peer-to-peer content sharing systems as building blocks. OceanStore [51] is one of the oldest systems that archive user objects using end users' resources. All OceanStore nodes take part in the storage system and all objects are encrypted. However, OceanStore is not available for mobile users and also it is not designed considering the resource constraints in the mobile networks. Tribler [20] is a peer-to-peer file sharing system, where peers are clustered into social groups and replicate their contextual information. Tribler does not consider methods to increase the availability of the content or minimise communication costs. Sharma et al. [52] proposed a friend-to-friend content replication strategy to ensure minimal replication and maximal availability. Erasure coding based friend-to-friend storage system is proposed in [53]. These coding based redundancy techniques are generally not suitable for social networking content due to their relatively small size and frequency of access as discussed in [21]. BlockParty [54] and FriendStore [55] are also peer-to-peer content backup systems that use only real-world friends' devices to store content. The authors argue that friendship based replication may have less permanent node departure compared to random replication. However, friendship based replication may provide lower content availability despite the higher trust and security. In general, these proposed methods do not ensure fairness of resource usage of devices. Also, they are not designed for mobile content dissemination and as a results does not take into consideration the resource constraints of mobile systems, especially bandwidth and battery usage.

Decentralised online social networks. LifeSocial [56] is a distributed social networking service that is developed on top of FreePastry[5] and PAST [50] content replication strategy. Similar to other peer-to-peer systems, LifeSocial is also not optimised for mobile devices. Buchegger et al. [57] has discussed the need and the challenges in designing D-OSNs and then proposed a D-OSN named PeerSoN [58]. PeerSoN employs the resources of global peer-to-peer architecture of Distributed Hash Tables (DHT) such as Kadmelia [48] coupled with data encryption to mitigate the privacy issues of C-OSNs. PeerSoN also enable direct data exchange when there

[5] `http://www.freepastry.org`

is no Internet connectivity using local short-range opportunistic networks and DHT lookup tables, which is an added advantage for mobile users. However, encryption techniques in PeerSoN has not been implemented and the efficiency has not been evaluated to date on mobile devices. Since then there has been a number of academic proposals for decentralised social networking services, which takes advantage of friend-to-friends content replication and encryption.

Safebook [19] is based on the concept of decentralisation and collaboration among friends to create a secure social network. Safebook consists of three components, trusted identity server, multi-ring node grouping structure called *Matryoshkas* and a peer-to-peer distribution based on Kadmelia. Friends are assumed to be co-operative and their devices are used for content and profile replication to increase the availability of user profiles. Trusted identity server is a central entity for managing identification and certificates of users which does not store any information. SuperNova [21] is another decentralised social networking system that uses content/profile replication on friends devices. The idea is to increase the online availability of content of the users who do not have enough friends for sufficient content availability, by using a super-peer based network of volunteer agents. Super-peers can provide any service such as storage, high bandwidth or management services. However, the users need to trust the services offered by super-peers although the content is encrypted. Safebook and SuperNova both attempt to provide as many functionalities that C-OSN services provide, including status updates, messaging and content sharing. MyZone [46] also deploys user profile replicas on the devices of trusted friends to increase the availability of the profile. The idea of all these systems is to mitigate the problems associated with online availability of content in distributed systems. However, none of these systems attempt to lower the communication cost for mobile devices. In particular, none of the systems explicitly consider mobile devices, and the availability of content/profile is dependent on the number of replicas. Hence, if used with mobile devices, it would result in increased communication costs and energy consumption.

Ephemeral content sharing. Geambasu et al. [59] propose a system called Vanish that provides self-destructive content even if the content is copied, transmitted or stored externally. The content becomes unreadable after a predefined period of time. Vanish leverages the services provided by global DHT, which essentially provides an index-value database on top of peer-to-peer devices. Vanish encrypts data with a random key not known to the user and then breaks the key into parts and sprinkle it to random indices in the DHT using *threshold secret sharing* [60]. The threshold determines how many pieces of the key is required to reconstruct the

original key. At the same time, Vanish destroys the local copy of the key on the creators device. If it is possible to reconstruct the key, the original content can be recovered. Due to inherent time-out feature in DHTs and churn of nodes over the time, some pieces of the key will not be available after certain period of time and the reconstruction will not be possible. For instance, VuzeDHT[6] has fixed 8-hour time out. However, the expiry time cannot be guaranteed and also the reconstruction. Another disadvantage is the delay in gathering enough pieces stored in various DHT nodes to reconstruct the key. This would not only dependent on the internet connectivity of the consumer, but also the internet connectivity of the specific DHT nodes.

2.3.2 Cloud storage based systems

Cloud storage based privacy-aware social networking services provide efficient use of resources and high content availability. However the cloud service providers have access to the user data and can perform data mining on user data as a means of monetising. Therefore, additional mechanisms are required to prevent unauthorised access of content by other parties. Data encryption and ephemeralness are two mechanism that have been used in the literature to realise the privacy-aware social networking on top of centralised servers.

Encryption based systems. Contrail [22] is a cloud-based distributed social networking architecture specifically designed for mobile users to address problems associated with connectivity, bandwidth and energy usage of smart mobile devices, which is conceptually similar to the proposed systems in this thesis. There are sender side content filters that are similar to those used in Publish/Subscribe systems [61], that enable users to selectively receive a subset of data published by another user. A simple cloud-based messaging layer is used to enable basic communication between Contrail enabled devices including encrypted data sharing. The cloud relays content between users and stores content for recipients who are offline to increase the reachability. Contrail entirely depends on the data encryption as the user data is stored in a cloud service provider. The larger the content item, the heavier the encryption and decryption in terms of computation and thereby higher the energy consumption.

In Vis-a-Vis [23], each user host their content on personal Virtual Individual Servers (VISs), which is provided by a cloud service. Vis-a-Vis supports scalable location based group abstractions which are controlled by users. However, Vis-a-Vis

[6] Azureus. `http://www.vuze.com/`

exposes unencrypted data to the cloud service provider and therefore, the privacy and security of user data are entirely at the hands of the service provider, while reducing the appeal of enhanced privacy and user control in D-OSNs. In all these cloud based systems, users have to provision their own cloud storage partitions. Therefore, the possible monetary cost of renting resources and the cost of management reduce the usability of such systems.

Ephemeral content sharing. Wickr[7] is an ephemeral content sharing service, which let users to choose from one second to five days. In addition, Wickr provides strong end-to-end encryption of communication as well as encryption of each message. However, it does not address any issues associated with communication or energy costs. Snapchat[8] also provides the functionality of self destructive content, where the users are given the option to specify the number of seconds (up to ten) the consumers are allowed to view the content. Snapchat app first uploads the content to be shared to their servers and then the access to that content will be removed after the expiry period [24]. Therefore, all privacy and trust related issues with C-OSNs are not addressed as it leaves a copy of the content, which is under the control of Snapchat. In addition, there are a number of security vulnerabilities in the way Snapchat implements ephemeralness. Snapchat monitors and report back to the creator if someone take a screenshot of the content. The users have found alternative ways to take screenshots without informing Snapchat app [62]. Moreover, Snapchat does not delete the content from the users device right after the expiry time, instead renames the content and makes it inaccessible via the Snapchat app [63], [64]. Despite these limitations, both Snapchat and Wickr have drawn a significant user attention due to its superior usability compared to other similar apps, highlighting the need for alternative services that offers better privacy.

2.3.3 Hybrid systems

Diaspora [18] can be considered as the only widely used D-OSN service. It is fully funded by user donations and owned by users, not by founders or an entity. It is estimated to have over 1 million users as of March 2014 indicating the user demand for privacy-aware social networking services. Diaspora provides the users freedom to select the location where their data is stored and hosted, called “Pods”. The users can choose from Diaspora’s open Pods hosted by different communities or they can set up their own Pod on their own personal device. PrPl [65] is another proposed

[7]https://www.wickr.com
[8]https://www.snapchat.com

service which enables a user to run the service on a home server, or rent a service to host their own data. The latter provides the best in terms of privacy and user control of data. However, the users need the necessary technical expertise to set up their own servers[9]. Diaspora and PrPl have not been originally designed to use mobile devices to store and host data. If a user's mobile device is used to host the content, i.e. a mobile server, either the availability of the content has to be compromised or the user will have to incur increased communications costs. Even if a cloud based hosting solution is used, the creator will incur the cost of uploading all content to the cloud storage irrespective of the popularity of the content among the creator's friends. In addition, there will be the cost of hosting.

Persona [66] takes advantage of attribute based encryption (ABE) [67] which allows users to share encrypted content with overlapping groups of users. Persona generates an ABE secret key (ASK) for each social group, e.g. co-workers. When a user wants to share content with "co-workers" and "family", an ABE encrypted key is generated using these attributes, public/secret key pair for the group and the user's public ABE key. Keys are exchanged through an encrypted peer-to-peer communication protocol. In particular, this prevents the C-OSNs from accessing content or meta data of the content for user profiling. Only one encryption is required to send content to a group of users, compared to many that is required when using traditional cryptography. Despite great flexibility, ABE encryption can still be computationally heavy on a mobile device. This can be overcome to some extent by giving the option to enable or disable encryption based on the importance of the shared content.

2.4 Summary

In this chapter, we reviewed the existing solutions in the literature that address the issues related to mobile network capacity limitations, user privacy and loss of control over user's own data. In the case of reducing mobile data traffic, we focused on systems and mechanisms that offload data traffic to alternative low-cost networks and content caching systems taking advantage of the capabilities of smart mobile devices and availability of different networks. Then, we reviewed decentralised social networking services and other attempts that aim to provide more control to the owners of content in social networking eco-systems.

As briefly discussed in Chapter 1, it is possible to categorise all existing proposals

[9] `https://wiki.diasporafoundation.org/Choosing_a_pod`

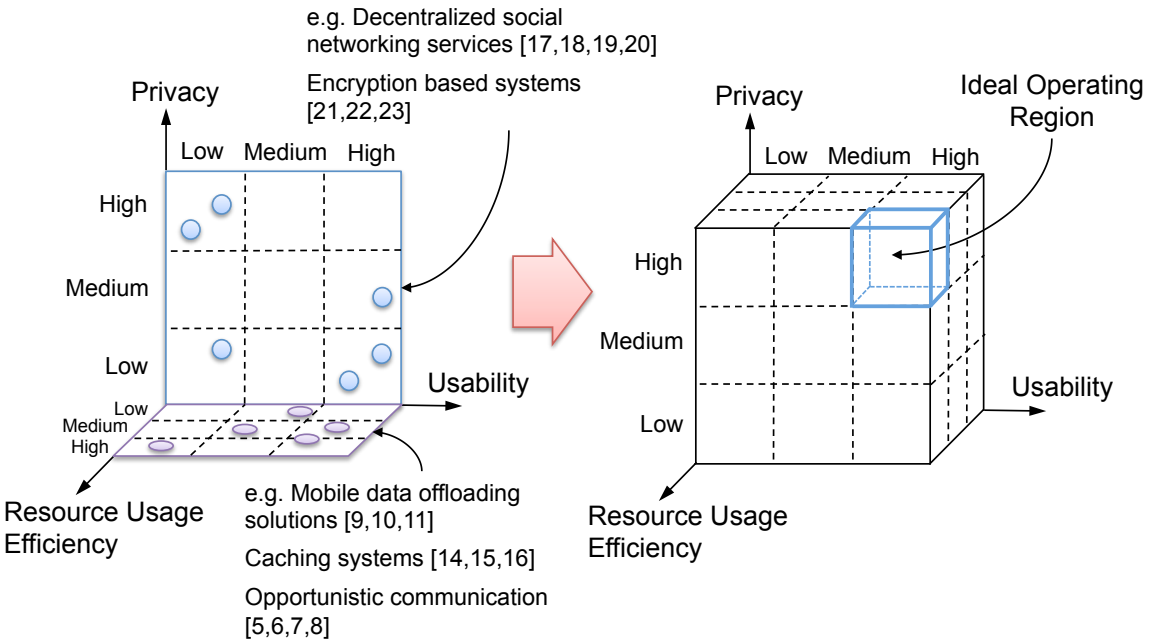


Figure 2.1: Categorisation of existing literature and the solution space for new mobile content delivery mechanisms.

based on the level of *usability*, *resource-usage-efficiency*, the level of *privacy* and user control provided, as shown in Figure 2.1. Majority of the academic work to-date focus on either privacy – usability plane or resource-usage-efficiency – usability plane. For instance, the proposed decentralised social networking services belong to the privacy – usability plane because they are not optimised for resource constraints in mobile networks as discussed in Section 2.3. In addition, majority of the content caching and data traffic offloading mechanisms discussed in Section 2.1 and 2.2 do not address privacy related issues associated with mobile content delivery systems. As a combination of these factors, we have not yet seen wide spread adaptation of any of these academic solutions on real practical mobile networks.

Therefore, the primary objective of this thesis is to develop mobile content delivery mechanisms that address all these three factors: privacy, resource usage efficiency and usability, which makes the solutions operate closer to the ideal operating region as shown in Figure 2.1. Smartphones and tablets of today have significant computing power, storage capacities and are connected to at least one of the available heterogeneous networks such as 3G, LTE, WLAN, WiFi-direct, Bluetooth, etc. It is likely that these devices will have even more computing power and storage in the future. However, current mobile systems do not explicitly harness these capabilities for data communication. Moreover, there are scenarios where users in the same

location will have similar interests. Therefore, if the capabilities of these devices could be leveraged rather than simply using them as thin clients, they could form the platform for providing solutions which address both the network and user related issues associated with existing centralised mobile content delivery systems such as: 1) communications cost, 2) privacy and 3) control over users' own data, without negatively impacting the usability and existing eco-systems.

The next chapter introduces three novel mechanisms referred to as User-Stash, MobiTribe and Yalut, exploiting the advanced capabilities of mobile devices and pervasive availability of different networks for the purpose of addressing the cost, privacy and user control associated issues whilst improving the usability. Then, the following chapters demonstrate the viability of each mechanism through analytical and experimental evaluations and finally the practicality is demonstrated by implementing them on commodity smart mobile devices.

Chapter 3

Novel content delivery mechanisms using distributed smart mobile devices

Smart mobile devices are being used to *create*, *consume* and *share* a variety of user data with other users, which has led to a significant growth in the use of social networking services as well as services that support the distribution of user generated content. Service providers are making available more free services, as they are able to monetise users' data, which is fuelling the demand for mobile data. As a result, mobile data traffic is growing faster than ever before and placing heavy demands on mobile networks. This has a two-fold impact on users. Firstly, the communication costs of users are increasing. Secondly, the monetisation activities of service providers pose threats to user privacy and control of their data. Therefore, the primary objective of this thesis is to develop mobile content delivery mechanisms that minimise the communication cost, improve user privacy and provide more user control, without negatively impacting the usability and breaking the existing digital services eco-system. As described in Section 1.1, the rationale of this thesis is to exploit the following three characteristics:

- Firstly, the fact that networks are under utilised in many locations even during peak hours and also the availability of alternative high-speed low-cost networks such as WiFi.

- Secondly, the mobile users have spatio-temporal locality of content creation and consumption patterns, i.e there are scenarios where users in the same location will have similar interests that are location specific, e.g. shoppers in the same mall, spectators in a sports event, students in a university, etc.

- Thirdly, the modern mobile devices have large storage capacities, considerable processing power and a number of connectivity options to connect with at least one of the available heterogeneous networks such as 3G, LTE, WLAN, WiFi-direct, Bluetooth, etc.

Therefore, if it is possible to shift some of the traffic from peak hours of the cellular networks to less loaded networks, or delay the transmissions until off peak times, the peak load can be significantly reduced. A smart mobile device, if used as an intelligent networking entity combined with applications exploiting the availability of heterogeneous networks, it is possible to enable this shifting of traffic either to off-peak times if the traffic is delay tolerant or to lower cost networks if low-cost networks are available. Moreover, if a smart mobile device is used as a content store, it is also possible to share content directly from one friend's device to another friend's device without storing the content in any centralised location providing the user more control of users' private data and preserving users' privacy. Since short-range networking technologies are often provide higher data transfer rates and lower latency, it is possible to improve QoE and thereby the usability of the mechanisms can be improved.

In essence, these scenarios exploit the capability of mobile devices to create groups of mobile devices that act as mobile clouds, which could be used for caching content and decentralised distributed content delivery. This chapter introduces three novel content delivery mechanisms which realise these hypotheses in real networks addressing a number of fundamental and practical challenges such as:

1. How to reduce the cost of content accessing when there is no low-cost network in the vicinity?
2. How to provide incentives for users to take part in co-operative content delivery mechanisms?
3. How to ensure privacy and user control of the content creator in online content sharing?
4. How to increase the content availability without a dedicated centralised storage?

User-Stash addresses the first two challenges by enabling mobile users to access popular content without using their cellular data plan at locations where there are no low-cost networks such as WiFi. The novelty of User-Stash stems from:

- The use of crowd to store the geographically popular content in a local store, exploiting the transient co-location and the spatio-temporal correlation of content consumption patterns of mobile users.

- Eliminating the need for prediction of user demand as well as low-cost networks, while exploiting the spare storage capacity on mobile devices and the capabilities to host a local WiFi network to create a new low-cost network for all users in the vicinity.

- The method of downloading the content via one network (cellular) and uploading to a stash via another network (WiFi), which is fundamentally different from in-network caching system.

- An advertisements based incentive scheme to encourage users to become User-Stash devices.

Section 3.1 provides the detailed descriptions of the User-Stash content delivery mechanisms.

MobiTribe addresses the third challenge of providing user privacy and user control of their own data by keeping the data away from centralised service providers and addresses the fourth challenge of increasing content availability without a dedicated centralised storage by effectively replicating content on the devices of their trusted friends. The novelty of MobiTribe are:

- Formation of Mobile Private Storage Tribe using the mobile devices of groups of friends, which stores and distribute user data whilst improving user privacy and providing the user more control over their own data.

- Scalable content replication algorithm that is being developed to increase the availability of content for the originally NP-Hard content replication problem, exploiting low-cost network availability, battery usage and storage capacity of mobile users.

- Seamless ability to interact with users of existing centralised social networking services whiles improving the usability of the proposed mechanisms.

Section 3.2 provides the detailed descriptions of the MobiTribe content delivery mechanisms.

Yalut augments MobiTribe to provide further cost benefits to users of location-based mobile social networks where the users are geographically clustered into communities whilst addressing all the above mentioned challenges. Yalut is novel in the sense of:

- Hybrid time-aware method of combining distributed storage and opportunistic friend-to-friend content dissemination exploiting the regular behavioural patterns of mobile users.
- Proposal of dynamic centrality metrics considering encounter time of users and duration of encounters to carefully select users to replicate content such that it minmises content delivery delay through opportunistic content propagation.
- Community based content replication algorithms and the use of available lowest cost networking infrastructure to transfer the content to be disseminated to geographically disconnected user communities.

Section 3.3 provides the detailed descriptions of the Yalut content delivery mechanisms.

3.1 User-Stash Architecture

The proposed User-Stash system makes use of the heterogeneous network and storage capabilities of the modern mobile devices. The system exploits both transient colocation of devices and the epidemic nature of content popularity, where users in a particular location and at specific times are likely to be interested in the same content. The key idea of the User-Stash system is to gather and store content of interest in a localised store, so that further requests can be served locally via local network. Thus, User-Stash avoids the use of expensive cellular bandwidth. An overview of the system's operations is described in the following section.

3.1.1 User-Stash: An Overview

The rationale of User-Stash is inspired by the delivery of information via the free news papers in public transportation systems in major cities around the world such as Metro in NYC[1] and and 20minutes in France[2], where users consume content (reading the paper) whilst travelling and leave the content (the paper) when they leave. User-Stash enables a similar service by providing storage facilities and a free local network in public contained places such as transportation systems. The users can then stash the content they have downloaded via other networks prior to coming to the area or downloaded whilst in the area. The users who contribute to the stash

[1] http://www.readmetro.com/en/usa/new-york/
[2] http://www.20minutes.fr

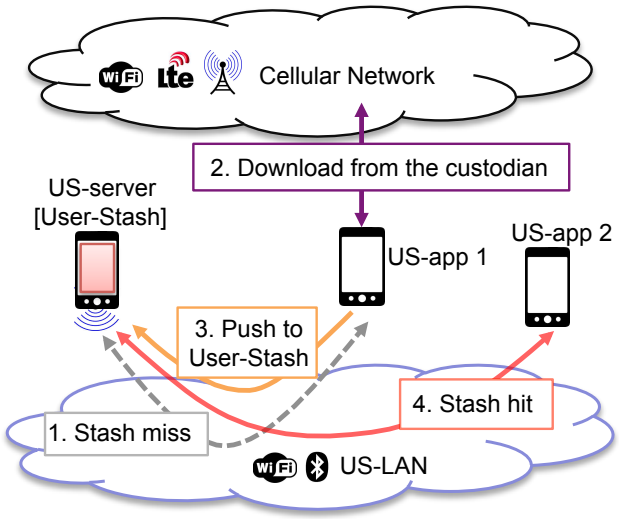


Figure 3.1: Operations of the User-Stash System

also gets access to the content in the stash. Thus, as the stash gets populated with content from contributors, users can access it locally. This is fundamentally different from an in network cache system, as the stash is not in the data path and content is pushed to the stash via a different network to which it was downloaded.

There is no cost for the users or for using the User-Stash system, except from a small energy penalty for pushing some content to the stash. There is no need to predict users' interests, as what is available in the stash will be what other users of the system consumed. The appropriateness, and the content integrity and authenticity can be adequately addressed through well known techniques that are used in the user generated content hosting platforms as discussed later in this thesis.

The architecture of the proposed system is illustrated in Figure 3.1. Typically, users install an app on their devices, *US-app*, or an extension (plug-in) to mobile web-browsers. The US-app enables the users to contribute the content that they have downloaded and consumed to a local store (*US-server*) via a local network (*US-LAN*), and making content on the US-server available locally. The US-server is provided by either another mobile device or a dedicated device, acting as a local server. In practice, users are required to get permissions to install the US-server mode from the User-Stash system provider. Prior to authorisation of the US-server mode, one is required to satisfy certain conditions. In particular, installer needs to ensure the US-server candidate device has adequate system capabilities (hardware, memory, etc.) to provide the required (caching and delivery) services. Devices of users who have registered as US-server, and have been accepted as such by the User-Stash system provider, will act as US-LAN access-points, in addition to acting as a (remote) stash. In Section 3.1.3, we provide the incentives for users to act as

US-servers. The US-app users can simply download the app from the app stores.

The US-server devices advertise their availability via specific SSIDs broadcast (e.g. us-SSID), which is operated by simply activating the WiFi Hotspot mode on the devices. All mobile devices running a US-app can then associate to the US-LAN when available and access the US-server. Associations to the US-LAN are handled by the US-app and is secured using e.g. existing standard WLAN security mechanisms such as a WPA authentication mechanism[3].

When a user requests content via the US-app, the device associates with US-LAN. The corresponding US-server then checks the stash for the requested content. In case of a stash hit, the content is delivered from the US-server via the US-LAN. In case of a miss, US-app is informed which triggers a switch of the network connection from US-LAN to the their own *private network* (e.g. 4G LTE, 3G network), and attempts to download the content from the content custodian as shown in step 2 in Figure 3.1.

Contribution to the User-Stash happens whenever a user downloads content via their own *private network connection* due to a stash miss or the user has downloaded content via different network elsewhere. When first connecting to a US-LAN, US-app pushes the content in the device local cache to US-server it is connected to (step 3 in Figure 3.1). The US-server makes a local decision as to whether it is useful or not to update its cache with the new content[4]. Then, whenever another US-app user requests the same content, a stash hit happens as illustrated by the US-app 2 case shown in Figure 3.1.

Note that more than one US-servers can be available on a given location at any given time. The US-app devices then connect to the US-LAN with the strongest signal strength (e.g. closest). However, in such a scenario, US-servers require to synchronise their stashes by pushing their cache content to each other for an optimal performance of the User-Stash system[5].

3.1.2 Application Scenario

We envisage several deployment scenarios of the User-Stash system. In the following, we describe a real-life deployment which considers a public transport scenario (e.g. Bus), where users commute to work. For simplicity, assume that the bus driver is a

[3] Authentication keys can be hard-coded in the app, and periodically updated if needed

[4] This depends on the content replacement strategy adopted by the US-server, e.g. replacing least recently used content (LRU), or removing least frequently requested (LFU) content, etc.

[5] For simplicity, in the remainder of the thesis we consider that there is only one US-server that is available at a given time.

registered US-server. Thus, the bus driver's mobile device will host a US-LAN within the bus. Initially, the stash of the bus driver's US-server only contains data that has been downloaded via the driver's own private network connection for his/her own use. When a passenger who has subscribed to the User-Stash service gets on the bus, the US-app discovers the availability of a US-LAN via the active discovery mode searching for specific US-LAN AP beacons. The passenger's mobile device associates and authenticates with the US-LAN in the bus.

First, US-app pushes locally cached content on the passengers mobile device to the US-server on the bus driver's device. When a new a passenger joins the US-LAN and requests to access some internet content (e.g. YouTube videoclips, hot trending news, etc.), US-app first checks whether the content is available in the stash. In case of a stash hit, the content is obtained from the US-server via the US-LAN. Otherwise, the user request is served through his own private network connection if available (the user is notified that the connection is redirected to outside the US-LAN).

More generally, each passenger who gets on the bus and has subscribed to the User-Stash system, transfers its own cached data to the bus driver's US-server stash. In such a semi transient environment, after a while the stash of the US-server will most likely contain the popular content, which will result in an increase in the stash hit rates. Thus, the passengers of the bus will be able to reduce the usage of cellular data, while the bus driver will be provided with incentives as described in the next Section for hosting the US-Server and US-LAN within the bus.

3.1.3 User Incentives/Disincentives

The incentive for the users of US-app is that using the User-Stash system provides the opportunity to minimise the amount of data that they will have to download via their own private network connections (e.g. LTE, 3G). Furthermore, their download latencies will be minimised when data is served from a US-server thus reducing the device's energy consumption.

There will be two versions of US-app app, 1) paid premium app and 2) Ad supported free app. In the case of an Ad supported free US-app, whenever it receives data from a US-server, it also displays an Ad as schematically shown in Figure 3.2. However, as these Ads are locally served from a US-server on the US-LAN, such data will not be accounted on the users private network quotas. The US-servers and US-apps will upload their Ads impression counts whenever they have access to an Internet connection via a *private network connection*, e.g when they have access to

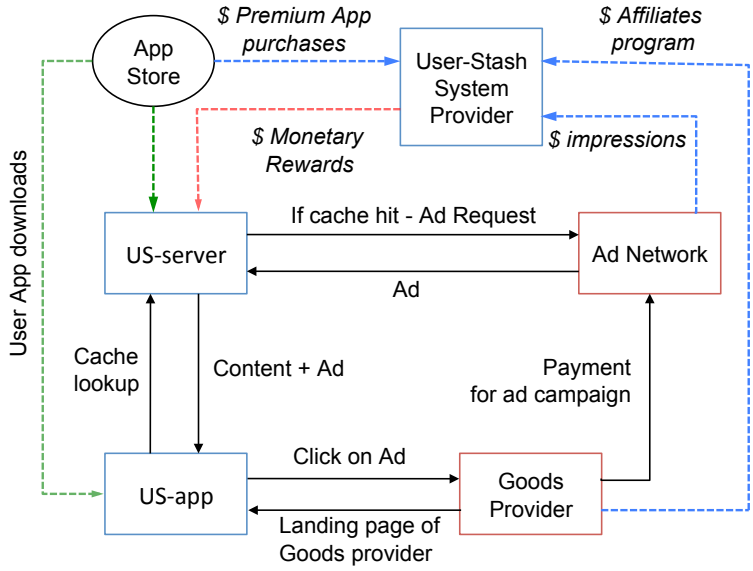


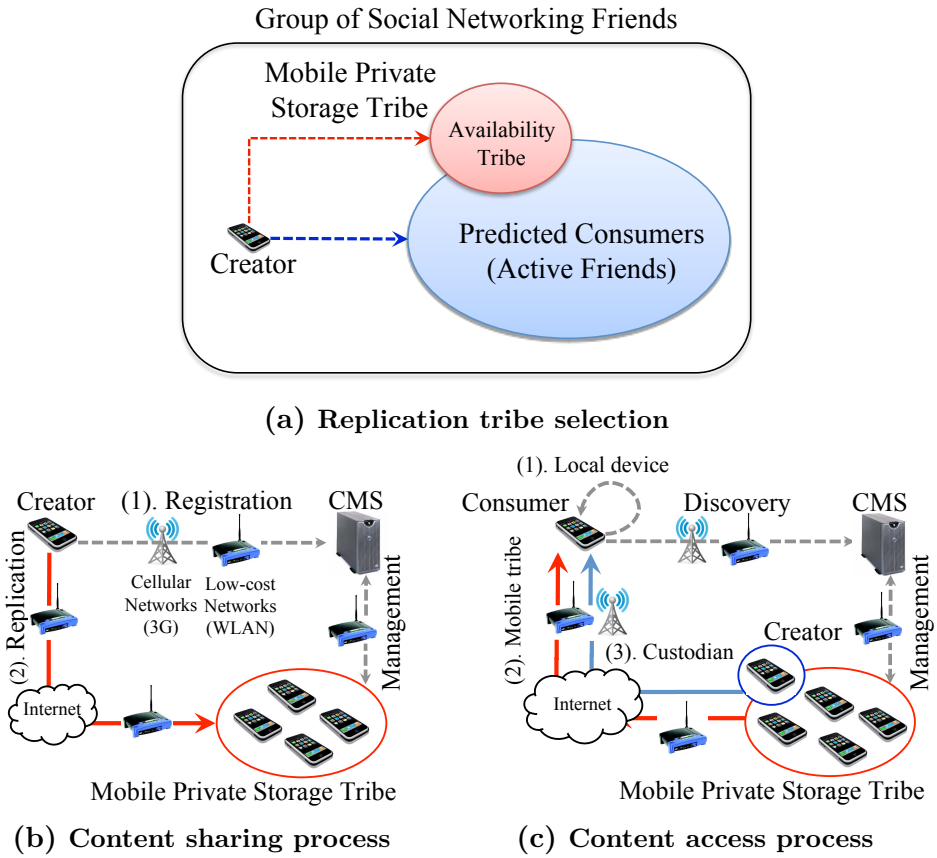
Figure 3.2: Incentive scheme for US-server users and cash flow of the system

a low cost network at home. This enables the User-Stash system to do the necessary accounting and detect fraudulent activities. In the case of a premium US-app, no Ads will be served.

From the US-server's perspective, the incentives for participating are two-fold. Firstly, US-server will get the same benefits as any US-app user as the content downloaded to the US-server would be available locally and thus reducing their download costs from their private network. Secondly, several reward-based schemes encourage users to act as US-servers. The US-server users will be rewarded based on the number of Ads impressions, as a portion of the Ads monetary revenue generated by the Ads clicks as shown in Figure 3.2. In the above model, the more a US-server supplies content via the US-LAN, the more it will get rewarded. This represents a clear incentives for US-servers to participate and operate as a cache contributor whenever they can.

Another important aspect to consider is the potential drain of battery from the content pushing process to the US-server, i.e. content sharing within the US-LAN. However, the extra energy consumption of uploading will be offset by the reduction in energy consumption when US-app downloads data from the US-server[6]. We assume that the US-servers will be connected to a power source, e.g. in the case of a bus drivers, US-server device will be connected to a power source on the bus.

[6]It has been shown that the energy consumed by WiFi is significantly lower than those of cellular networks mainly due to the heavy traffic load in cellular networks leading to lower speeds and thus taking longer time to complete data transfers [41].



(a) Replication tribe selection

(b) Content sharing process

(c) Content access process

Figure 3.3: MobiTirbe Architecture

3.2 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. MobiTribe achieves its cost objective by delaying uploads and prefetching of content, using low-cost network connections ahead of time. MobiTribe protects user privacy by keeping user data on the users' own device or on a trusted friends' devices and away from third party service providers.

The MobiTribe architecture is shown in Figure 3.3. A *Creator* distributes the content to be shared to two groups of devices from the creators' friends as shown in Figure 3.3a. The group of friends devices form the *Mobile Private Storage Tribe - mTribe*, as described in Section 3.2.1. The devices of the friends that are predicted to consume the content becomes the *Predicted Consumers*. Each user device identifies potential shared content to consume, based on a content recommendation algorithm similar to that described in [68]. This is achieved by each device maintaining and periodically updating three matrices for content recommendation, 1) user-user social relationship matrix, 2) content-content similarity matrix based on keywords and tags

of contents, 3) user-content matrix, which shows the relationship between content access and users. Then the content shared by friends are ranked according to the relevance to the user for pre-fetching as described in [68].

3.2.1 Sharing of content

The architecture relies on a centralised entity *(Content Management Server - CMS)* to track the addresses and current connectivity of devices hosting content and to manage the peer-to-peer content sharing and downloading processes. A creator sharing content via a social networking application such as Facebook would result in MobiTribe initiating the content sharing as a two step process.

1) Registration: The content is registered with the CMS immediately after creation. The registration is done through any available network as shown in Figure 3.3b. Then, a link to the content, pointing to the CMS, is advertised to the users who subscribed to receive updates from the content creator through a social networking service such as Facebook or Twitter.

2) Replication: Each subscribed device then decides whether to pre-fetch the content, based on predicted demand and informs the CMS of their intention of downloading the content forming the group of *Predicted Consumers*. The CMS decides whether these *Predicted Consumers*, provide sufficient availability of the content for the other (on-demand) consumers based on the availability patterns of all users. If not, an extra set of devices, namely an *mTribe* is formed to replicate the content by the CSM as it has a global view of the system state in terms of both mobile devices and network infrastructure. The CMS carries out the grouping periodically or in response to a significant degradation of the availability of a tribe's content. The device grouping is done using the context information of devices and network infrastructure as described in Section 5.1.

An *mTribe* device, may not be able to provide access to the content continuously due to three primary reasons, namely the communications costs, battery usage, and not being "on" all the time. This is overcome by replicating content on devices that have complementary connectivity, sufficient battery power and being "on". Moreover, through this process, if the traffic is routed through a low-cost and/or less congested network, it will result in significant benefits to both the service providers and the users. The down-sides of the content replication are the overheads associated with replication. These disadvantages can be minimised as the *mTribe* members, where possible, are chosen from the potential consumers, i.e. subscribers to receive updates from the creator. Then, there is no overhead as they are likely to consume

the content.

The replication of the content takes place when the creator and the devices in the *mTribe* have access to low-cost networks. The low-cost network connections are considered to be networks with spare capacity, WLANs or device to device connections. The CMS coordinates the replication process acting as a tracker. The storage on *mTribe* devices is managed by removing the old replicated content using standard cache replacement strategy, namely LRU. The creator has the option to remove/archive or keep its own content. If old content becomes popular, it will be treated as a newly generated content. The replication strategy ensures that the devices in the *mTribe* have complementary low-cost network access patterns, thus making it highly probable that one of the devices in the *mTribe* can serve the content over a low-cost network connection.

3.2.2 Downloading of content

A consumer initiates the content downloading process by clicking on an advertised notification appearing in the social networking application as shown in Figure 3.3c. This triggers the content downloading process through:

1) Local device (Cache hit): If the consumer's device has successfully prefetched the content, it would be served from the device itself. If not, i.e a local cache miss, the request is delivered to the CMS using the link which is included in the notification feed.

2) Mobile tribe (Cache miss): When the CMS receives a request for content, it directs this request to the devices in the *mTribe* that could potentially serve the requested content. The choice of the *mTribe* devices to forward the request is based on the real-time network connectivity, the communication load and the remaining battery capacity at the time of request.

3) Download from custodian: When a request for content is received by a CMS, if the content is not available in the *mTribe*, the request is forwarded to the content creator. The content creator has the option of delivering the content over the lowest cost data connection available to the creator or make it unavailable until the pre-distribution of content to *mTribe* has been completed.

A content consumer has the option of downloading the content over whatever network connection available or the content can be scheduled to be downloaded through a low-cost network when it becomes available. In contrast, the content uploading path, i.e. from *mTribe* to fixed network edge, always uses low-cost networks as *mTribe* devices are selected such that at least one device is connected to a low-cost

network at a any given time.

Upon downloading the content, the consumer also shares the content for a defined period of time, thus helping the system to scale with demand. To perform the content sharing between the content creator, devices in the *mTribe* and content consumers, an augmented BitTorrent peer-to-peer protocol, with the CMS acting as a tracker to balance the traffic load, is used[7].

3.2.3 Privacy and user control of data

In online content sharing services, when a user accesses the service, there is some loss of privacy through direct and indirect channels as described in [69]. Thus, it is impossible to completely prevent leakage of privacy. Our aim is to minimise the loss of privacy compared to the existing centralised content sharing architectures.

This is done in MobiTribe by keeping user data away from third party service providers thereby limiting the ability to mine user data or profile users. The CMS only facilitates the interactions among friends and does not have access to any user data stored on *mTribe* devices. The users can retrieve any content hosted on the *mTribe* back through the MobiTribe service. As a result the privacy of the user and ownership of the data are preserved. Although the content is replicated by specifically selecting the devices of friends of the creator, the risk of privacy leakage due to replication is minimised.

Further, we enable ephemeral content sharing in MobiTribe. The content creator is given the ability to specify the duration of time the content is to be made available for consumers. When a consumer opens the content, MobiTribe app will start a count down timer for the duration specified by the creator and when the time expires the content will be removed from all devices. To make it reliable, all downloaded content are cocooned in the MobiTribe app and can only be accessed through the MobiTribe app. Once removed from the device, the content availability will not be visible and the content will be deleted from all devices. Moreover, even if the smartphone is lost or is hacked, there will not be any security threat as the content is no longer available. In addition, the users can retrieve any content hosted on the friends' devices back through the MobiTribe service.

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 are required to determine the capability of a device to take part in *mTribe*. To avoid leakage of privacy, these user context information is aggregated to a single

[7]Description of the practical implementation is provided in Chapter 7.

bit which indicates the availability of the device to host data, *Device Availability*, as described in Section 5.1. Although storing of the aggregated context information on a centralised server may result in some loss of privacy, the design allows users to significantly reduce the communication cost of content sharing.

3.2.4 User Incentives/Disincentives

User privacy and the user control of data is inherent in the system design which is the primary incentive for users to use the service. In our opinion, one of the major reasons behind the low acceptance of the previous proposed decentralised social networking services is their sole focus on privacy preservation, rather than complete systems that address cost of use, energy consumption and ease of use. In contrast, MobiTribe addresses all these issues. More importantly, MobiTribe allows users to use their centralised social networking services, which they are familiar with, to interact with their friends.

Predicting the consumption of content for pre-fetching reduces the monetary and energy costs of delivering the content as shown in Section 5.3. Then, it also improves the user experience both in terms of the startup delay[8] and for the re-buffering [42]. Finally, due to the operator managed data offloading and traffic localisation capabilities, MobiTribe could be used a telecom operator to provide price incentives such as family and friends data plans as in the case of voice communication.

3.3 Yalut Architecture

Mobile social networks consist of several physically disconnected components/communities due to the social behaviour of mobile users. These subgraphs in mobile social networks are composed by the users with specific similar interests or real life social interactions. For examples, the community created by the shoppers in a specific food store in a shopping mall, a group of friends sitting together in a sport event, etc. can be considered as instances of creating these subgraphs. This temporal community structure has been observed in the literature in other mobile social networks as well [35]. In these scenarios, the lowest cost network for distributing content will dependent on the location of the source and destination devices. If they are within communication range of each other, short-range opportunistic networks such as WiFi Direct, Bluetooth or WiFi Tethering could be the lowest cost network.

[8] We compared playback of pre-fetched video on an Android device from local storage with streaming over loaded and unloaded 3G cellular, and the startup delays were 0.5s, 7s and 25s.

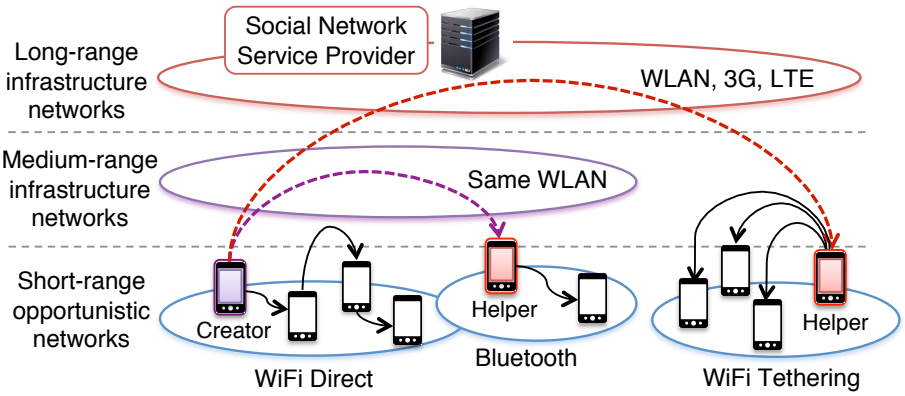


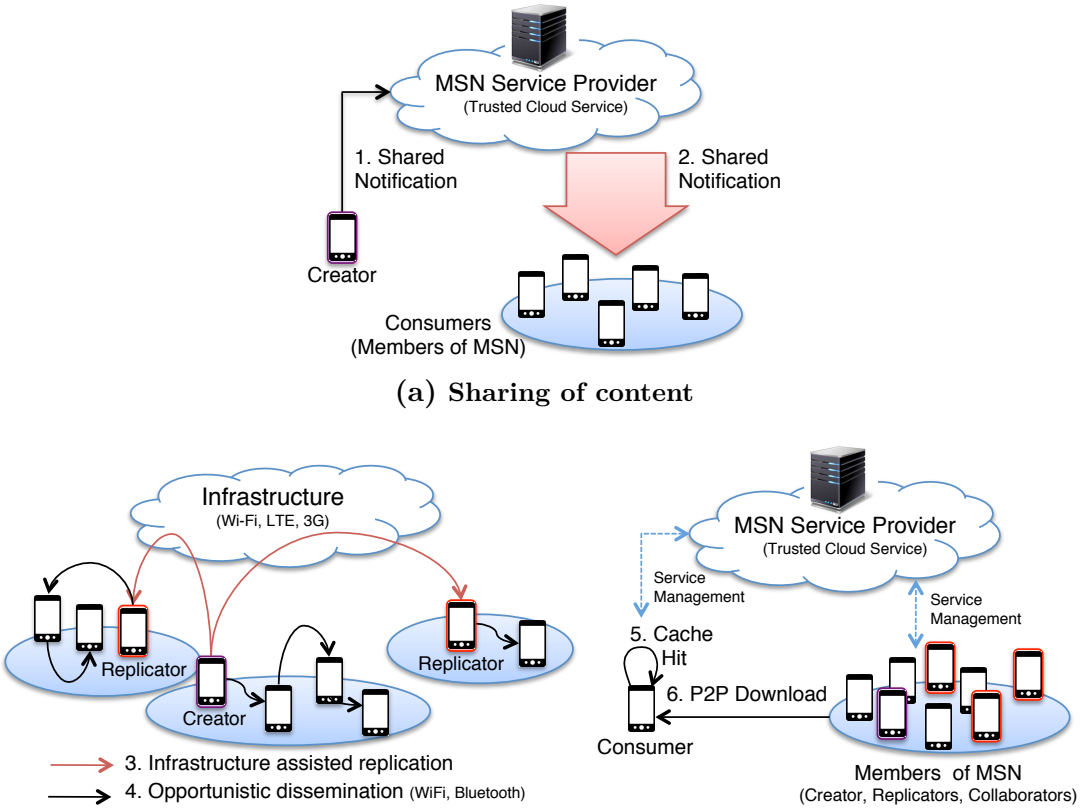
Figure 3.4: Overview of the proposed concept of hybrid mobile content dissemination.

If not, WiFi or off-peak cellular could be the lowest cost network. The proposed system, Yalut exploits the availability of these hierarchical heterogenous networks to disseminate content among social network friends depending on their location as shown in Figure 3.4. In these scenarios, it is possible to use opportunistic direct wireless connectivity among the devices due to the geographical proximity.

This section augments the previously proposed MobiTribe architecture for even more effective content dissemination. The objective is to provide cost-efficient timely distribution of UGC in mobile social networks. We assume that users search and download content using a mobile app provided by Mobile Social Network (MSN) service providers such as Foursquare or Facebook. For example, when a user enters a shopping mall, he or she subscribes to the location-based MSN related to that particular shopping mall. The idea is to use the mobile app and the MSN service as the front end to advertise the content and use device-to-device communication for content dissemination.

3.3.1 Sharing and downloading of content

To illustrate the operation, consider the case where a particular user, a creator, wants to share content with a set of users who have previously been identified as friends through a social networking service (e.g. Facebook). Once the content creator shares content, a notification is delivered to all users through the MSN service as shown in Figure 3.5a. We assume that the member list of the MSN is available to all users through the MSN app. Then, a set of users namely *Helpers*, preferably from different communities in the MSN, are selected. Helpers then pre-fetch and replicate the shared content using available low cost networking infrastructure such as WLANs or 3G as shown in Figure 3.5b. Finally, these helpers propagate the



(a) Sharing of content

(b) Disseminating the shared content

(c) Downloading of content

Figure 3.5: Content sharing process in the proposed hybrid content dissemination strategy for MSNs

content to other users in the community using opportunistic direct device-to-device communication technologies such as WiFi Direct, Bluetooth and WiFi Tethering as shown in Figure 3.4. Figure 3.5c shows the content downloading process. When a consumer click on the shared notification, the mobile app first checks for the content on the local device, if not available, the content is downloaded from the members of the MSN using a peer-to-peer communication protocol, similarly to MobiTribe.

Assume that potential consumers among these friends can be predicted based on their history of content viewing patterns. Then, we can propagate the content only to these subset of friends who have been identified as potential consumers and leave the option for other friends to fetch the content from the creator or one of the consumers who have already downloaded the content. If this pre-fetching can be scheduled to use low-cost networks depending on the location of the users (Figure 3.4), it will minimise the communication costs, energy usage and storage requirements of mobile devices. We contend that even though it is difficult to predict the content consumption of a user with a high degree of accuracy, it will still possible to have a significant impact on reducing the communications costs.

3.3.2 Replicating of content

Even though the replication improves delivery performance, the networking infrastructure usage incurs cost and therefore its usage needs to be minimised. The energy and monetary cost of networking infrastructure usage to the user vary with the time and the location of the user. For instance, a 3G cellular network at night can have a lower energy cost to the user compared to congested WiFi network [41]. Short-range opportunistic networks can be considered as the lowest cost network. If short-range opportunistic networks are to be used for disseminating content, the main challenge is guaranteeing the timeliness of the delivery as it is not possible to ensure a sufficient number of consumers being present in the vicinity of the creator. This is overcome by replicating the content on some carefully selected consumers, namely helpers, and using these helpers to propagate the content to other consumers through opportunistic communication, as shown in Figure 3.4. Each user in the MSN can potentially become a helper. The decision as to which devices replicate content and become helpers can be based on the device context information, such as available network types, the battery level, AC power availability and the amount of spare storage capacity as before.

The viability of the proposed system thus depends on the selection of helpers, and the minimisation of replication. We consider that a central entity similar to CMS performs the helper selection based on the users' connectivity information, as described in Section 6.2.2. As shown in Figure 3.4, initial content replication is carried out by either a pre-existing networking infrastructure such as WLANs and cellular networks, or short-range opportunistic communication, if the helpers are in the vicinity of the creator. At the same time, the creator initiates content dissemination to the consumers in the vicinity via short-range opportunistic communication.

Similar to MobiTribe, to perform content dissemination among the users, a modified version of BitTorrent peer-to-peer protocol, is used. In particular, a consumer becomes a *propagator* or a seeder only after the consumer has downloaded all pieces of the content. This enables the prioritisation of the energy consumption of a mobile device, by first downloading the full content for its own use and then helps others.

3.3.3 User Privacy and Incentives/Disincentives

As discussed in the previous section, content sharing with online friends inherently reduces the privacy of the user through direct and indirect channels [69].Similar to MobiTribe, our aim is to minimise loss of privacy and confidentiality threats compared to existing social networking systems.

Yalut achieves the privacy objectives by design as it keeps user data away from the centralised service providers by using trusted social networking friends for storage. Moreover, the proposed system does not propagate content through strangers compared to other opportunistic content dissemination systems. The improved user privacy and user control over data can be considered as one of the incentives to take part in the proposed system. The other incentive is that the users can reduce their cellular data usage as the system enables opportunistic content delivery. Therefore, when friends are in the communication range of each other, Yalut provides further cost benefits than MobiTribe due to usage of lowest cost option of device to device communication. In addition, it is important to consider incentives for helpers as they leverage their resources for the benefit of the community. When acting as a helper, the system enables content pre-fetching and thereby providing another incentive for the helpers. We believe that these incentives will be sufficient for users to collaborate with their friends for the benefit of the community of friends as a whole. In addition, if necessary, it is possible to incorporate a credit scheme where helpers accumulates credits in return for propagating others content as proposed in [70].

3.4 Summary

This Chapter has introduced three mobile systems User-Stash, MobiTribe and Yalut to mitigate mobile data traffic growth whilst providing privacy and usability benefits for the users.

User-Stash provides cheap access to geographically popular content using a novel mechanism of uploading the content downloaded by US-app users via cellular network to US-servers via a local network US-LAN. User-Stash eliminates the need of predicting low-cost network availability as it exploit the capability of smart mobile devices to create a local low-cost network. Moreover, we proposed a sustainable in-app advertisement based incentive scheme for users to become US-server users.

MobiTribe enables decentralised social networking on smart mobile devices by making the content available through low cost networks irrespective of the location of the users. The availability of content via low-cost networks is increased by replicating content on friends of the content creator forming mobile private storage tribe. It is also possible to integrate MobiTirbe with existing centralised service providers provide over the top services with improved privacy and user control of data.

Yalut extends MobiTribe architecture for further benefits in terms of communication costs using a hybrid time-aware method of combining distributed storage

and opportunistic friend-to-friend content dissemination. Yalut exploit the available lowest cost networking infrastructure to transfer the content to be disseminated to geographically disconnected user communities.

The next three Chapters 4, 5 and 6 provide content delivery mechanisms including respective content caching and replicating methods of the three architectures. Finally, Chapter 7 shows the practicality and feasibility of all three architectures through implementations on Android mobile devices.

Chapter 4

User-Stash: Dissemination of popular content

User-Stash addresses the challenges of lowering the cellular bandwidth usage and improving the QoE for users. As proposed in Section 3.1, it consists of a novel content caching mechanism that was inspired by the delivery of information via the free news papers in public transportation systems in major cities around the world, where users consume content (reading the paper) whilst travelling but leave the content (the paper) when they leave. The User-Stash mechanisms proposed in Section 3.1 relies on the transient colocation of devices and the epidemic nature of content popularity and exploits the capabilities of modern mobile devices. In essence, the observation that users in a particular location, at a given time period are likely to be interested in similar content with a high probability. The system allows users to contribute the content that they have downloaded and consumed to a local store (*US-server*) via a local network (*US-LAN*), and then allows other users to retrieve items of interest to them using a mobile app (*US-app*) from the same local store US-server. If the content could be obtained from the US-server via the US-LAN, the usage of users' *private cellular network* connection will be reduced, and users' QoE will be improved due to lower latency and higher throughput of short-range local content delivery.

In this chapter, we analytically and experimentally validates the hypotheses used in Section 3.1 such as spatial-temporal correlation of content access and capabilities of smart mobile devices to host and distribute content locally.This Chapter makes the following contributions:

- Models the video content access and the corresponding content consumption patterns of mobile users in public transportation systems using a unique large real-life dataset from a popular video content provider.

- Shows that more than 60% cache hit rate can be achieved during an hour bus ride regardless of the content popularity distribution using the developed models of the behaviour of the User-Stash users. Moreover, it is possible to reduce the cellular bandwidth usage and also reduce the latencies. In particular, more than 80% of the passengers can save at least 40% of their cellular data usage during a typical average city bus commute of 10 minutes.

- Demonstrates the feasibility and practicality of the proposed content delivery mechanisms through the development of a prototype Android application. The experimental results shows that the US-app users lowers the device energy consumption compared to accessing the content through cellular networks if the proposed system provides at least 10% cache hit rate and throughput when accessing content locally via US-LAN can be as high as 6Mbps.

The remainder of this chapter is organised as follows: The operation of User-Stash in public transportation system is modelled using a real-world dataset as described in Section 4.1. In Section 4.2, the performance evaluation of User-Stash content delivery mechanisms in terms of stash hit rates and the use of cellular bandwidth through a simulation study are presented. Then in Section 4.3, the performance in terms of latencies and energy consumption is evaluated using measurements of the prototype implementation on Android mobile devices. Finally, Section 4.4 concludes the chapter.

4.1 User-Stash System Model

In the following, we model the User-Stash content delivery mechanisms using a real-world dataset of video content access information of mobile users to derive probabilistic models for user behaviour and online content access patterns.

4.1.1 Dataset in use

We use logs of video content access of mobile users of PPTV[1], one of the largest VoD service providers in China. The dataset spans over one week in December 2011 and consists of content requests from three major cities - Shanghai, Beijing and Tianjin. It consists over 9.5 million content requests by more than 500K users. There are 22 categories of videos including news, movies, TV series and animations. Table 4.1 summarises the statistics of the dataset. In the remainder of this section, we

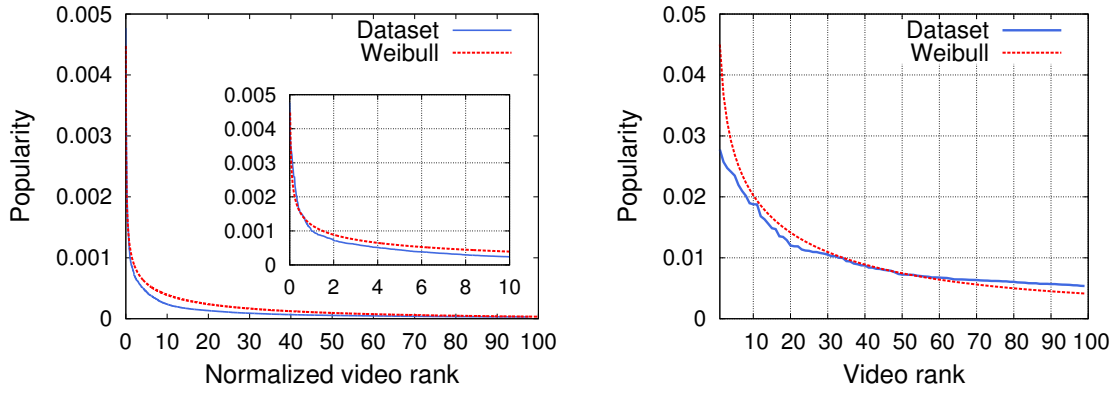
[1] `http://www.pptv.com`

Table 4.1: Summary of PPTV dataset

Field	Statistics
Duration	7 days - Dec. 2011
# of users	516, 149
# of content requests	9, 579, 576
Video categories	22 (news, movies, etc.)
% of WiFi requests	76.15%
# of requests/user/day	2.65
Avg. length of a video	50 min
Avg. length of view time	18 min

analyse the popularity and the size characteristics of videos as well as the transient aspects of content access and consumption patterns of this data, for the purpose of modelling User-Stash mechanisms.

4.1.2 Popularity distribution of video content



(a) For all videos (Global popularity). **(b) For top 100 popular videos.**

Figure 4.1: Content popularity of PPTV dataset.

It has been previously reported that the popularity of online video content follows a Weibull distribution [71]. Figure 4.1a shows the maximum likelihood estimation (MLE) fit of a Weibull distribution for the PPTV dataset using the SciPy Library[2]. The videos are ranked based on the number of requests during the one week period. We obtain an R^2 goodness-of-fit value of 0.9382 with the MLE parameters for the Weibull distribution equal to a shape α of 0.68 and a scale λ of 1600. Even though the linear regression of the goodness-of-fit provides a very high estimate of the empirical distribution, the Weibull distribution does not fit well for the entire range as seen in Figure 4.1a. Home pages of online video websites such as YouTube

[2] SciPyLib-http://docs.scipy.org/doc/

contain the most popular videos for a particular geo-location. Similarly, User-Stash operations are mostly concerned with the most popular content. Therefore, we then focus on the popularity distribution for only the top 100 most popular videos of the PPTV dataset in Figure 4.1b. The shape parameter of the MLE fit for Weibull distribution is similar to the global popularity ($\alpha = 0.75$), while the scale ($\lambda = 41.7$) is drastically reduced. For the top 100 videos, the Weibull distribution represents a good approximation of the actual popularity distribution with an R^2 goodness-of-fit test value of 0.984.

The popularity of content i (P_i) is then considered as a Weibull probability function such that $P_i = \frac{\alpha}{\lambda}\left(\frac{i}{\lambda}\right)^{(\alpha-1)} e^{-(i/\lambda)^{\alpha}}$ where $i > 0$ and $\alpha, \lambda > 0$. However, the popularity distribution in Figure 4.1 has a considerable long tail due to the large variety of content access in a large geographical area. Since User-Stash is mainly designed to operate in a confined geographical area for a limited period of time, the distribution of the content popularity should be narrower than the PPTV dataset. Hence, we further investigate the effects of content popularity in Section 4.2 by varying the shape parameter of the Weibull distribution whilst considering different content popularity scenarios.

4.1.3 Size distribution and video categories

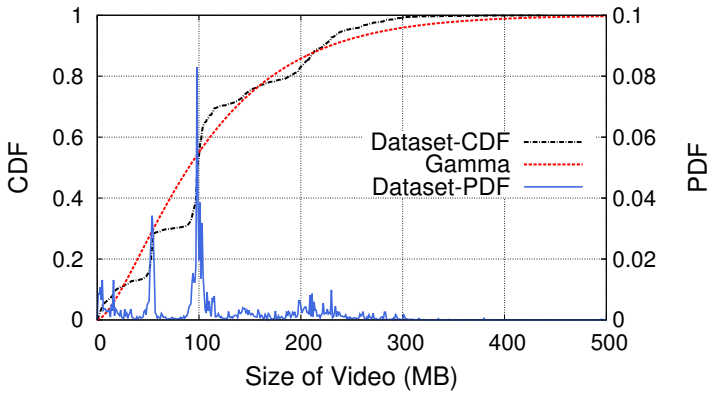


Figure 4.2: Video size distribution of all content

The video size (in Bytes) directly affects the storage capacity of the US-server devices. As our dataset only contains the length of a video in seconds, we calculate the size of the videos assuming a bit rate of 330Kbps for this model (as the average content bit rate in YouTube [72]). The video size follows a Gamma distribution similar to YouTube content size distribution [71] where k and θ denote the shape and scale parameters of the distribution respectively. The cumulative probability distribution function for the content size of the dataset is depicted in Figure 4.2.

The overall MLE fit for the size distribution is found to be a Gamma distribution with $k = 1.63$ and $\theta = 68.74$.

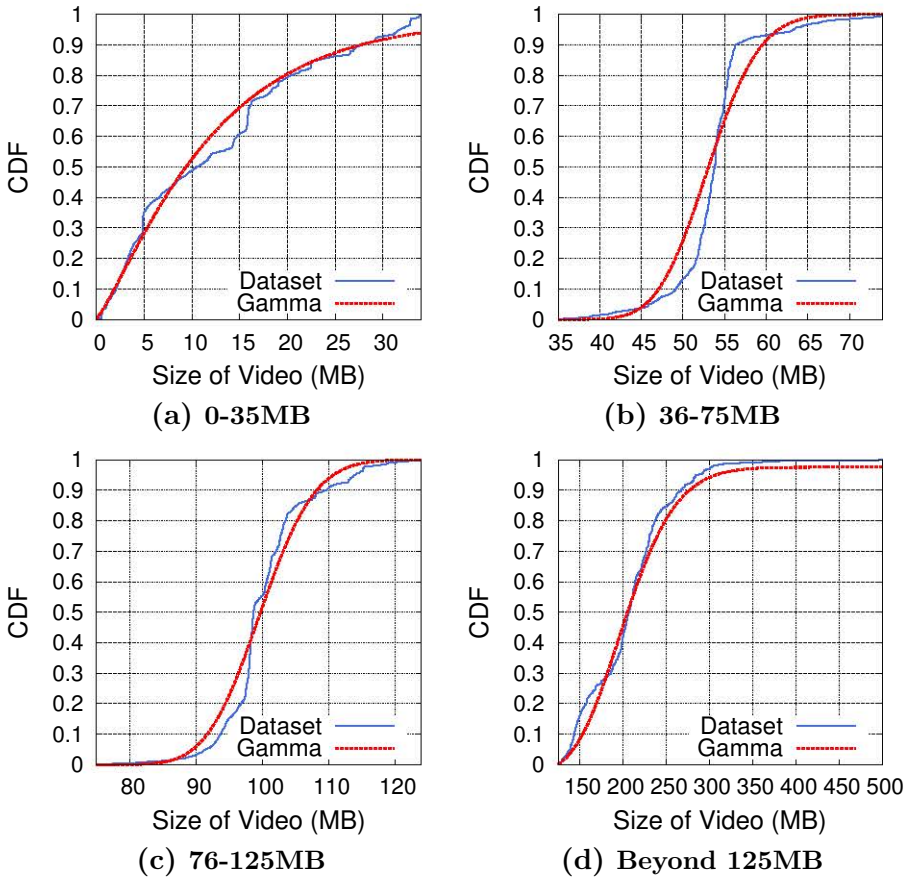
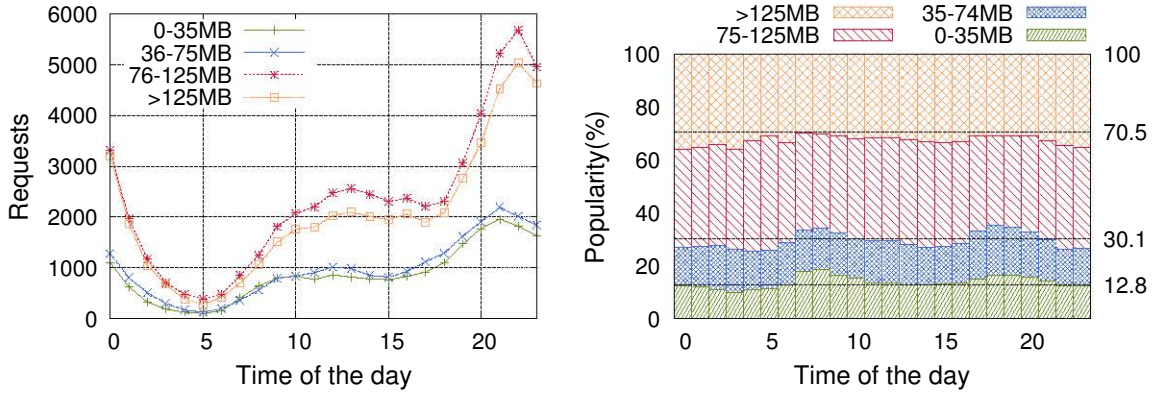


Figure 4.3: Size distribution of four categories of video content

Table 4.2: Models for video categories

Size (MB)	Type	Gamma distribution	Pop. (%)
c1- 0-35	Trailers	$k = 1.527$, $\theta = 6.271$	12.8
c2- 36–75	Variety show	$k = 31.474$, $\theta = 1.637$	17.3
c3- 76-125	Animation	$k = 242.589$, $\theta = 0.414$	40.34
c4- >125	TV shows	$k = 19.273$, $\theta = 10.704$	29.57

However, the overall model does not fit well with the actual size distribution (R^2= 0.31) due to the high percentage of certain video sizes as can be seen in the PDF distribution, e.g. peaks at approximately 50MB and 100MB. The PDF illustrates that there are four separate categories of sizes: 0-35MB, 36-75MB, 76-125MB and beyond 125MB. These broad categories correspond to 4 out of the 22 categories [73], namely movie trailers, variety shows, animation and TV shows. We model the size



(a) Number of requests **(b) Popularity of categories**

Figure 4.4: Popularity of video categories during a day

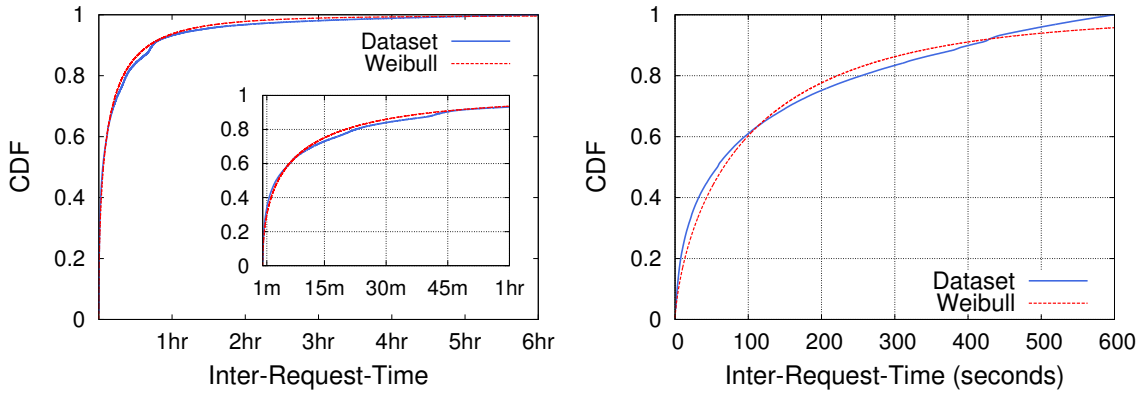
of these four categories separately, as shown in Figure 4.3. Table 4.2 summarises the Gamma model parameters (R^2= 0.99 for all models) as well as the popularity of each of these categories. In this dataset, more than ∼70% of the content are of size greater than 75 MB. In contrast, the average content size in online video distribution services such as YouTube are much lower in the order of 10MB [71]. However, the size distribution in both services follow Gamma distribution. Therefore, we model the video size distribution as a Gamma distribution such that the size of content i, corresponds to a probability function $S_i = \frac{i^{(k-1)}e^{-(i/\theta)}}{\theta^k \Gamma(k)}$ where $i, k, \theta > 0$ and $\Gamma(k)$ is the Gamma function evaluated at k.

4.1.4 Transient aspects of content request and consumption

First, we study the behaviour of content requests with respect to the time of day. Figure 4.4a shows the average number of content requests of videos of four size categories during each hour of the day. During the working hours (∼ from 9am to 6pm), the number of requests are relatively stable across all the categories. After that, the number of requests increase rapidly peaking between 9pm and 10pm. In particular, night time peaks for larger video categories of animation and TV shows (∼5500 requests per hour) are larger than for the two smaller categories (∼2000 requests per hour). However, if we normalise the popularity of each category from the total number of requests for the particular hour, all categories show approximately steady behaviour throughout the day as illustrated in Figure 4.4b. For instance, the portion of 0-35MB video requests out of all requests in an hour is ∼12.8% at any given hour of the day. Therefore, we consider that the popularity of each video category to be quasi-stationary and does not change with the time of the day even

though the total number of content requests is much greater during night time.

We extract inter-request-time for individual users from the dataset and then model the inter-request-time for content i, I_i as a Weibull distributed variable with γ and β as the shape and scale parameters respectively. Figure 4.5a shows the CDF of inter-request-time of all users in PPTV and the Weibull distribution with MLE fit parameters of $\gamma = 0.5$ and $\beta = 456.14$. In particular, approximately 50% of users request a video at least every 5 minutes and more than 80% every 20 minutes. Since the dataset we consider here consists of videos of larger view length, the inter-request-time can be expected to be lower than this representation in other online video sharing services. In addition, it is expected that in the application scenario of User-Stash, users would tend to request content more frequently as mobile devices are generally heavily used by passengers in transit [74]. Therefore, we investigate the distribution of inter-request-time being less than 10 minutes reflecting the fact that an average duration between bus stoppings is approximately 10 minutes. Figure 4.5b shows that again it follows the shape of a Weibull distribution with $\gamma = 0.7$ and $\beta = 110$ and R^2 value of 0.9909, where 80% of users request new content at least every 3-4 minutes.

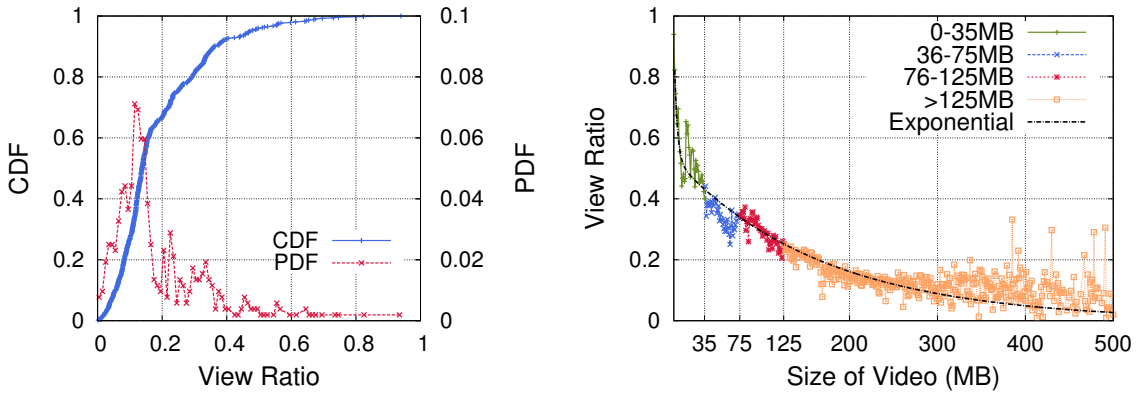


(a) Global IRT **(b) Distribution of IRT<10min**

Figure 4.5: Inter-Request-Time (IRT) time distribution of individual users

Figure 4.6a shows the probability distributions of the view ratio (defined as the view time normalised by the length of the video) for all videos. It can be seen that more than 80% of the videos have less than 0.3 view ratio. Even though the average length of a video is as large as 50 minutes, the users rarely watch a full video leading to an average actual view time of just 18 minutes (Table 4.1). Moreover, the median view time is as low as 1 minute. This is an intriguing factor to consider when designing content caching mechanisms such as User-Stash, as it determines whether it is reasonable to stash the full content. However, the average view ratio

is considerably higher (> 0.4) for shorter videos (0-35MB), as shown in Figure 4.6b. Then, as expected, the view ratio gradually decreases with the size of the video. We model this behaviour of view ratios as a linear combination of exponential functions such that, the view ratio of a video of size $s > 0$ as $V_s = a \times \exp(\lambda_1) + b \times \exp(\lambda_2)$ where $a = 0.4$, $b = 0.53$, $\lambda_1 = -0.3$, $\lambda_2 = -0.006$ and $\exp(\lambda) = \lambda e^{-\lambda s}$. The exponential model provides a close representation of the content view ratio of PPTV users with a 0.995 R^2 goodness-of-fit test value.



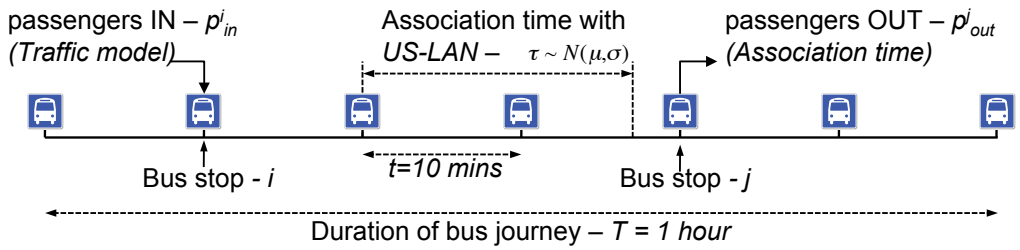
(a) Probabilities of view ratio **(b) View ratio vs size of the video**

Figure 4.6: Distributions of view ratio of videos

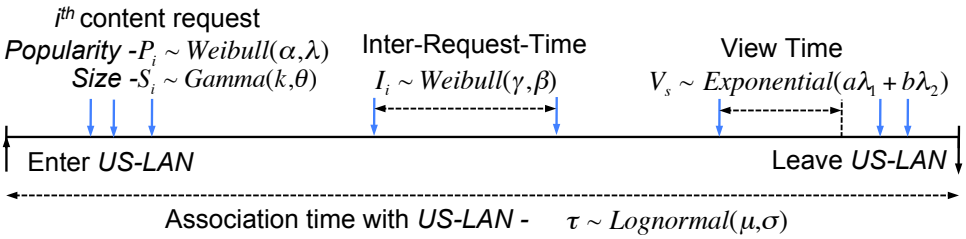
4.1.5 Transient aspects of passengers on a bus

We consider a scenario of hosting the US-LAN in a bus as described in Section 3.1, where there is one US-server device and N number of US-app users who are within the communication range of the US-LAN during a given time period of T. The only US-server device would belong to the bus driver. N is limited by the maximum number of passengers for a given bus which we consider to be equal to a constant 50 for the sake of simplicity. Then, the time period T is bounded by the duration of a bus journey. Since User-Stash is a passive content storage mechanism, the transient aspects of US-app users such as the number of users in the bus at a given time, duration of the bus journey of each user and content access and consumption behaviours of users determines the effectiveness of the cache. We model the transient aspects of the bus journey as illustrated in Figure 4.7a.

Specifically, due to the mobility of users, the number of US-app users connected to the US-server device varies with time. We consider that the bus journey is limited to one hour. The association time of each US-app device (τ) with the US-LAN is considered as a Log-normal distribution similar to the passenger travel time in a bus



(a) Model of the passengers of the bus



(b) Model of the content access

Figure 4.7: Overview of the User-Stash system model

[75]. That is, on average after $\tau = e^{\mu+\sigma^2/2}$ time, US-app users leave the US-LAN, where the mean $\mu = 0.6197$ and the standard deviation $\sigma = 10.48$ for the log normal distribution of τ. Since the average time of association is around 10 minutes, we consider that there are bus stops at every 10 minutes as shown in Figure 4.7a.

The number of passengers that get onboard at each bus stop is dependent on a number of factors such as the time of the day, day of the week, weather, etc. To emulate this, we assume three traffic models to evaluate different scenarios as follows;

- *Peak:* The bus is full all the time, i.e. $N = 50$. At each bus stop, the same number of passengers get on and off the bus.
- *Off-peak:* Ten passengers get onboard at every bus stop if there is enough room for 10 more people in the bus. If there is not enough space, the bus will be filled up to the maximum capacity.
- *Random:* At the beginning of a journey, a random number of passengers between zero to 50 get onboard. At each bus stop, a random number of passengers between zero to the remaining capacity of the bus get onboard.

In addition, we assume that all passengers onboard at any given time are users of the US-app and new US-app users get onboard at each bus stop (p^i_{in} in Figure 4.7a) based on either one of the proposed three traffic models. As soon as a passenger gets onboard, the device associates with the US-LAN. At each bus stop, passengers get

out of the bus (p^j_{out} in Figure 4.7a) if their association time (τ) has expired. During the association with US-LAN, each US-app user generated content requests based on observed content access models, which are graphically shown in Figure 4.7b. Table 4.3 summarises all the parameters of these models. These observed models and characteristics of video content access, namely - content popularity, size, inter-request-time, view ratio and passenger model - are further exploited in the next section to evaluate the effectiveness and benefits of the User-Stash content delivery mechanisms.

Table 4.3: Summary of User-Stash system model

Transients of content access behaviour	
Content popularity	Weibull - α=0.75, λ=41.7
Size	Gamma - k=1.527, 31.474, 242.589, 19.273 θ=6.271, 1.637, 0.414, 10.704
Inter-Request-Time	Weibull - γ=0.7, β=110
View time	Exponential - a=0.4, λ_1=-0.3, b=0.3, λ_2=-0.006
Transients of bus scenario	
Duration of bus ride	T=1 hour
Bus stops	Every 10 mins
Capacity of a bus	N=50
No of Passengers	Peak, Off-peak, Random
Association time	Log-normal (μ=0.6197, σ=10.48)

4.2 Performance Evaluation

We developed a discrete-event simulator in Python due to its flexibility and rich statistical libraries, which is developed to evaluate the performance of User-Stash mechanisms under different scenarios of content access and consumption. The simulator takes the system model parameters summarised in Table 4.3 as inputs and calculates the stash hit rate and the bandwidth saving statistics. Later in Section 4.3, we consider a real-life scenario when User-Stash is implemented on commodity smart mobile devices and analyse the effects of network conditions on the system performance.

Simulation setup

Figure 4.8 schematically illustrates the components and the complete process of the discrete-event simulator in Python. *User Generator* component first creates

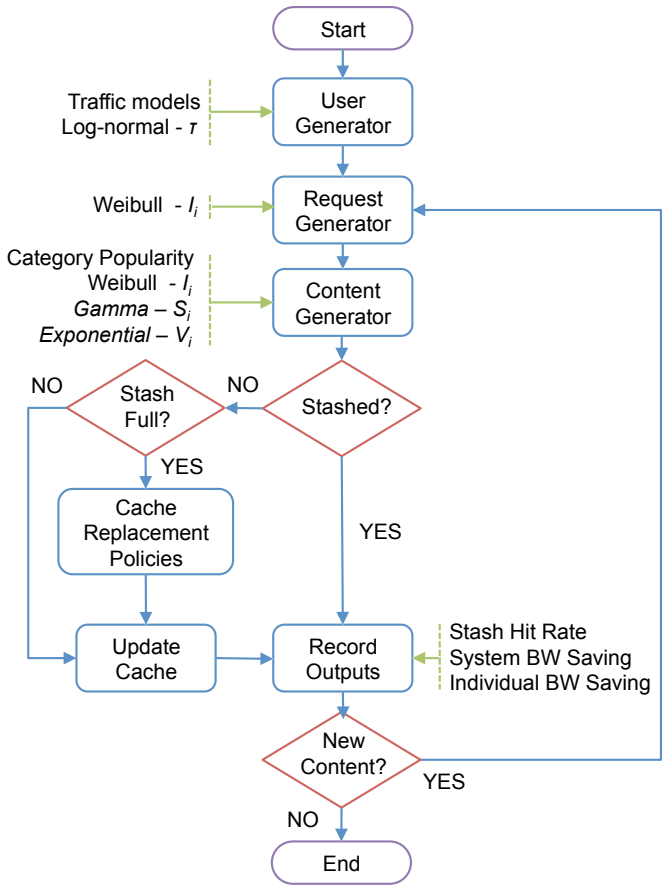


Figure 4.8: Illustration of the Python Simulator

a set of users with entry times and leave times for a bus trip based on the three passenger traffic models described in Section 4.1.5. The leave time is determined by the log-normally distributed passenger travel time (τ). We also assume that all passengers in this scenario are US-app users. *Request Generator* appends each user dictionary by assigning time instances to generate new content request events using the observed Weibull distributed inter-request-time (I_i) model. Then, the simulator progresses through the list of new content requests by all assigned users with one second granularity from the beginning of a bus trip to the end. When there is a content request, *Content Generator* draws attributes for that particular content request (i), namely content popularity (P_i), view time (V_i), content category type (c) and content size (S_i) from the analytical models summarised in Table 4.3. SciPy[3] statistical library is used to draw values from probabilistic distributions. A new content will not be generated for the same user until the current content is watched to the expected length. A unique ID number is given to each content, hashing the content category and the content popularity. All generated content attributes are

[3] `SciPyLib-http://docs.scipy.org/doc/`

saved under this unique ID and it is also used to identify the stash hits and misses.

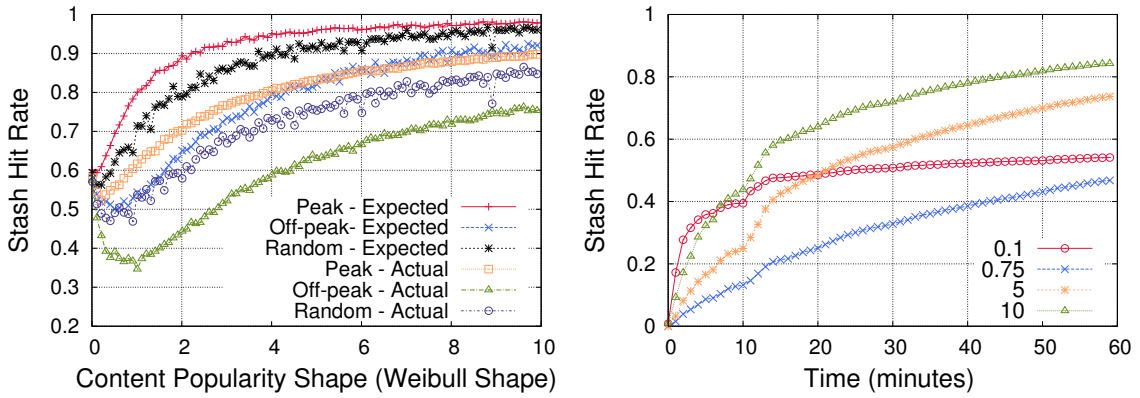
For this evaluation, the communication between all entities, i.e. a user device, the US-server and the content provider, is considered to be via a quasi-perfect communication channel where there is unlimited bandwidth, a non significant transfer delay and zero bit error rate. The US-server is represented by a dictionary holding all the stashed video content. A requested content is served from the US-server, if it is a stash hit. Otherwise it is considered to be downloaded and pushed to the US-server. At this stage of the simulation, the statistical results such as stash hit rate, total bandwidth saving and individual bandwidth saving are recorded with reference to the simulation time.

The effectiveness of User-Stash is mainly driven by the popularity of the content items stored in the US-server device. In addition to the user behavioural factors, the storage capacity of the US-server device and the cache replacement policy constitute two additional factors that may undermine the efficiency of the system. One of the cache replacement policies then determine whether or not to stash the content if the local stash is full. Based on the policy, certain contents in the stash will be evicted to make room for the new content. First, we consider an unlimited storage capacity at the US-server device. Then, we examine the system performance under practical resource constraints along with different cache replacement policies. Next, the simulator moves forward to the next content request until the end of the bus journey. For each evaluation metric, we performed 50 simulations to increase the confidence of the results. We validated the correctness of the simulator from the desired outputs for a certain inputs in different scenarios.

4.2.1 Performance with an unlimited stash size

The expected stash hit rate, referred here as ($E[H_s]$), is the stash hit rate that can be expected for the content stored in the US-server device at a given time. $E[H_s]$ is then the probability of finding a requested content item in the stash, i.e. $E[H_s] = \sum_{\forall i} P_i$ where P_i is the popularity of content i. The actual stash hit rate is the ratio between the number of hits and the total number of requests for a considered time duration. If the content popularity distribution does not vary in time, the actual stash hit rate should converge to $E[H_s]$ asymptotically.

Figure 4.9a shows the stash hit rate values after a 1-hour bus ride for the three traffic models described in Section 4.1.5 under various content popularity distributions (i.e. varying the Weibull α parameter). The scale of the content popularity distribution is considered to be equal to that of the empirical PPTV dataset ($\lambda = 41.7$).



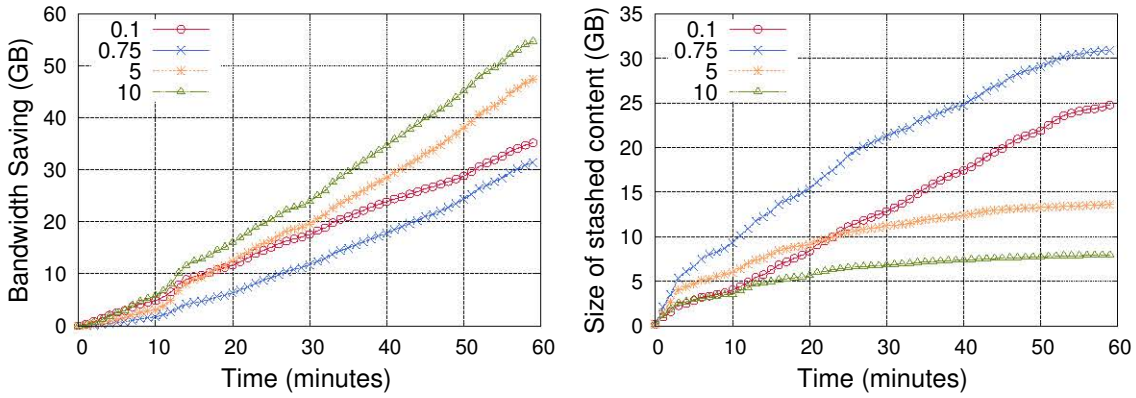
(a) As a function of the shape of content popularity, and when $\lambda = 41.7$.

(b) As a function of time for $\alpha = 0.75$ for the PPTV dataset and $\alpha = 0.1, 5, 10$ for extreme scenarios.

Figure 4.9: The stash hit rate performance during a one hour bus ride.

The shape parameter changes the long-tail nature of the popularity distribution. When $0 < \alpha < 1$, the smaller the shape, the higher the popularity of the most popular content. When $\alpha \geq 1$, the larger the shape, the narrower the probability distribution of the content. As a result, the stash hit rate increases with α in general for $\alpha \geq 1$. As expected, the peak traffic model gives the highest stash hit rates due to the larger number of contributors to the User-Stash. In particular, the expected stash hit rate is higher than 60% regardless of the type of popularity distribution in consideration. Notably, in the off-peak scenario, the minimum stash hit rate is still greater than 35%. The random traffic model results in up to 67% hit rate for the particular shape value ($\alpha = 0.75$) of the PPTV dataset, which suggests a considerable potential of higher cellular bandwidth saving for users with similar behavioural patterns to PPTV customers when using User-Stash.

Figure 4.9b depicts the variation of the actual stash hit rates during a one hour bus ride, starting from empty stashes in all the devices and considering a random traffic model. It shows the results for five different content popularity distributions: $\alpha = 0.1$ and $\alpha = 5$ or 10, which represent two extreme cases of very large and very short long-tail popularity distributions respectively. $\alpha = 0.75$ represents the popularity distribution for PPTV dataset. For the two extreme cases of $\alpha = 0.1$ and 10, the probability of requests of the most popular items is higher than other distributions. As a result, we observe a rapid increase in the stash hit rate early during the bus journey, i.e. immediately after the stash is populated by the first set of passengers bringing high popular content to the User-Stash. In the case of a very long-tail distribution of the content popularity, e.g. $\alpha = 0.1$, the evolution of the stash hit rate stabilises almost immediately after the peak observed at the first



(a) Cellular bandwidth saving of all users. **(b) Total amount of stayed content.**

Figure 4.10: Bandwidth saving and amount of the stashed content during an hour.

stop of the bus (the 10^{th} minute of the simulation) reaching a 50% hit rate. This is due to the disparity of the content requests as an effect of the long-tail distribution. In general, we observe that the actual stash hit rate monotonically increases with time, as the new users joining the system tend to request popular content that is already stashed. User-Stash reaches a significantly high hit rate between 70-80% in the case of $\alpha = 5$ or 10 due to the narrowness of the content popularity distributions at higher α values.

The cellular bandwidth saving is one of the primary objectives of User-Stash. Figure 4.10a shows the total savings of cellular bandwidth as a function of time in our particular scenario of a unique US-server device serving the bus passengers (according to different content popularity distributions). We observe that the system gradually off-loads a significant amount of data, which results in total savings ranging between ∼35GB and 55GB. This illustrates the potential of the proposed system as an alternative solution for reducing cellular network bandwidth usage. To estimate the required storage capacity, we calculate the total amount of stashed content, under different content popularity distributions. For the lower distribution shape values (e.g. $\alpha = 0.1, 0.75$), the stashed amount is significantly larger as shown in Figure 4.10b, despite it being shown earlier that the stash hit rates are comparatively lower in Figure 4.9b. The total size of the stashed content is approximately 30GB after one hour trip. We expect that for an online content sharing service such as YouTube, the total size of the stash in use is rather lower mainly due to the lower average content size (∼10MB as opposed to ∼ 100MB in PPTV). Next, we examine the impact of limiting the storage capacity at the US-server device.

4.2.2 Performance with a limited stash size

Due to the ever increasing storage capacities of mobile devices which can support up to 160GB as of today[4], we believe that assuming a 10-30GB of storage stash in a mobile device is very realistic. However, for the best performance, the stashes need to be populated continuously. Therefore, it is necessary to have an appropriate stash replacement strategy.

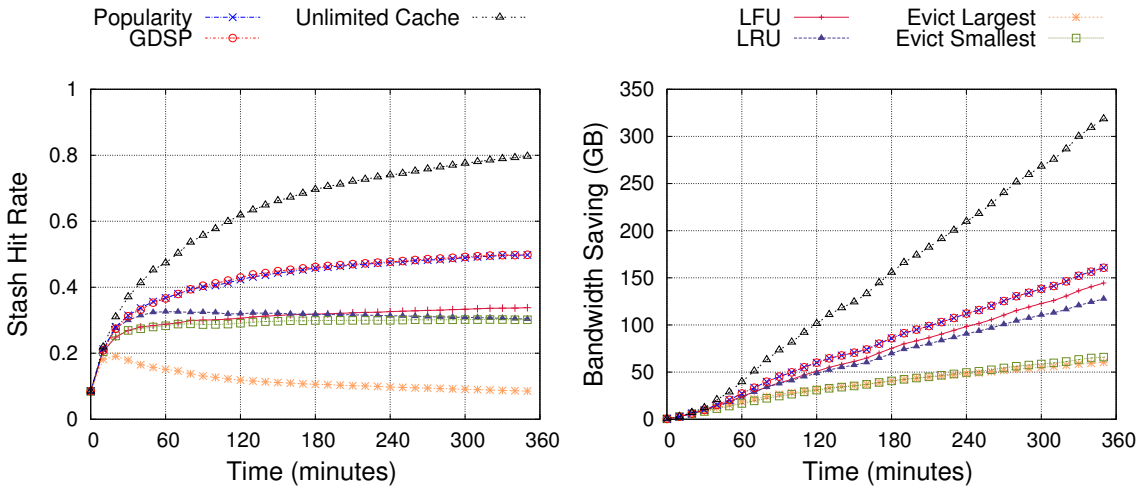
Stash Replacement Policies

To represent an extreme case, where the stash is provided by a low end smartphone, we restricted the stash size to 10GB. For this evaluation, the content popularity is considered to be similar to the PPTV content (i.e., the distribution parameters are $\alpha = 0.75, \lambda = 41.2$). We consider a number of cache replacement policies, namely:

- *LRU (Least-Recently-Used):* Keeps the most recently requested content in the cache without considering the popularity of the content.
- *LFU (Least-Frequently-Used):* Keeps the most popular content items in the cache without considering the time of request.
- *Evict Smallest (and respectively Largest):* Removes the smallest (and respectively the largest) content in size which is in the cache to make room for the new content.
- *Popularity:* Replaces the content that has the lowest popularity value (P_c) per unit size (S_c) with new content using the utility metric (P_c/S_c).
- *GDSP (Greedy-Dual-Size-Popularity [76]):* Captures both popularity value per unit size and recency of request including a dynamic ageing factor L. The content with the lowest utility metric $U(c) = L + M_c P_c/S_c$ is replaced such that, $L \leftarrow \min\{U(c) : \text{for every c in the stash}\}$. The cache miss penalty M_c is equal to one, considering the retrieval cost for stash misses are equal for all content items.

Figure 4.11a shows the stash hit rates observed for different cache replacement policies. All the policies except Evict-largest, results in a stash hit rate higher than 5% after a 1 hour trip. Notably, the two content popularity-based policies show relatively higher stash hit rates. Although LRU and LFU are two of the most

[4]Samsung Galaxy S5-128GB external and 32GB internal storage: `http://www.samsung.com/global/microsite/galaxys5/specs.html`.



(a) Stash hit rate **(b) Bandwidth saving of all users**

Figure 4.11: Performance with limited stash at US-server device, stash size=10GB.

popular cache replacement policies, their performance is not superior in the case of User-Stash because the content access is heavily dependent on content popularity. GDSP and Popularity show similar results. However, GDSP is expected to perform better than Popularity when the content popularity change with time, since GDSP takes the time of content access into account in its objective function.

Higher stash hit rate does not always lead to the highest bandwidth saving as can be seen in Figure 4.11b. For instance, even though stash hit rate for Evict-smallest is higher than Evict-largest by 20%, bandwidth saving for both these strategies are almost similar. Furthermore, the difference between LRU and GDSP are comparatively low in bandwidth saving, despite GDSP saves more cellular bandwidth for users because GDSP tends to keep high popular smaller content in the cache compared to LRU. Overall, if we employ a cache replacement policy which takes time of access, popularity and the size of the content, User-Stash performs better even under extreme conditions such as only having a 10GB stash.

Transient aspects of passengers

Figure 4.12 illustrates the effects of travel time of bus commuters which is determined by the log-normally distributed association time with US-server (τ) and the duration between two consecutive bus stops. Since the same user is not going to request the same content multiple times, the User-Stash receives a diverse content requests (from the long-tail part of the popularity distribution) when the same content commuters travel for longer periods. As a result, User-Stash achieves higher stash hit rate for

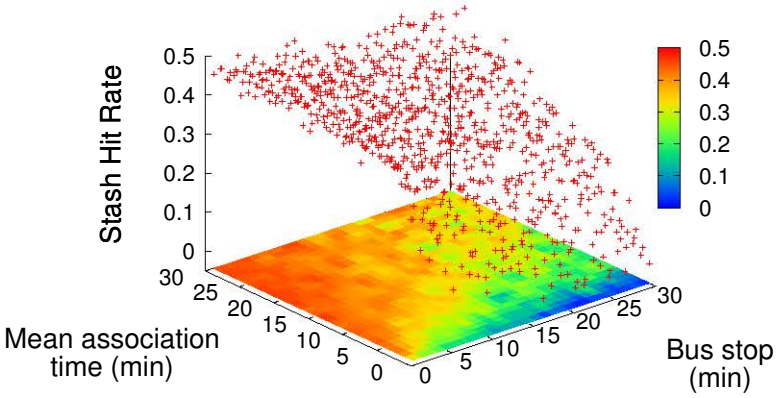


Figure 4.12: Stash hit rate against transient aspects of passengers, stash size=10GB and the random traffic model.

short travel distances where there is frequent arrival and departure of new commuters as shown in Figure 4.12. In addition, mean association time is the deceive factor as the stash hit rate does not vary significantly with the duration between two consecutive bust stops for a given association time. Therefore, User-Stash performs better in metro type transport scenarios. Overall, the heat map depicts that for majority of the cases, stash hit rate reaches more than 30% after one hour, even for a 10GB stash.

4.2.3 Benefits for User-Stash users

For a US-app user, one of the benefits would be the reduction of the monthly capped cellular network data usage. Therefore, we evaluate the cellular bandwidth saving for individual users during a bus ride. Each user associates with US-LAN for only 12 minutes on average according to the log-normally distributed association time. Since US-server device starts with an empty stash at time=0, the passengers during the first hour would be the users that receive least benefit. In the first hour when using a 10GB stash, nearly 19% of the users do not save any bandwidth as shown in Figure 4.13, while that reduces to ∼11% for the passengers who get onboard in the third hour. The savings are expected to be much higher in real-world systems as the storage of the US-server device can be easily upgradable to 128GB. Figure 4.13 illustrates that there are only 2% of the passengers with zero bandwidth saving benefits who enter the bus in the third hour, when the storage is increased 128GB. The mean bandwidth saving is also shifted to 40-60%. Moreover, 80% of passengers save more than 40% of cellular bandwidth.

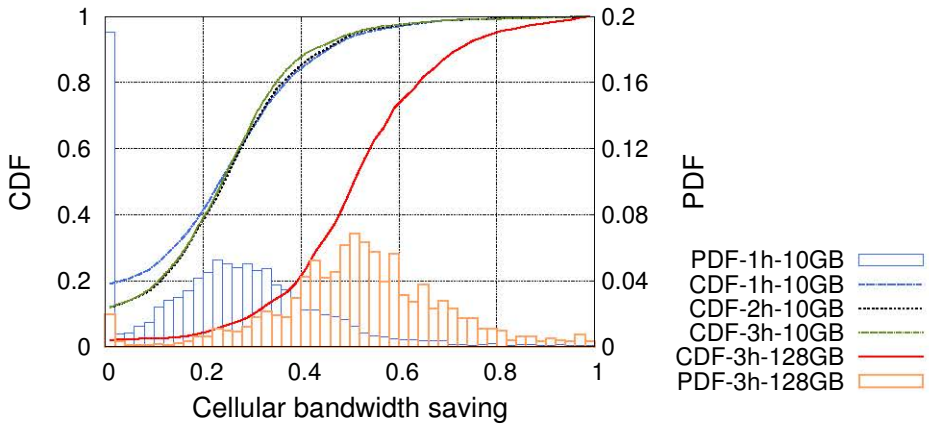


Figure 4.13: Individual bandwidth saving for $\alpha = 0.75$.

4.3 Experimental Evaluation

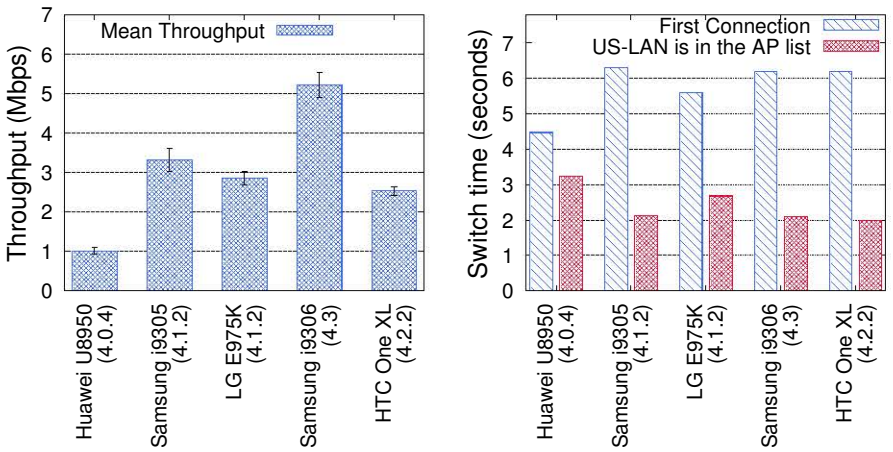
To demonstrate the viability of the User-Stash content delivery mechanisms, we implemented US-app and US-server as Android apps. Although the proposed concepts are valid for any popular content type, in our implementation we focus only on video content as described in Section 3.1. The details of the implementation are described in Chapter 7.

In this section, we present and discuss the measurements obtained using this implementation with the US-server on a Samsung Galaxy S4 (i9306), and the US-app on a variety of smartphones from different manufacturers and with various capabilities.

4.3.1 Throughput and latency

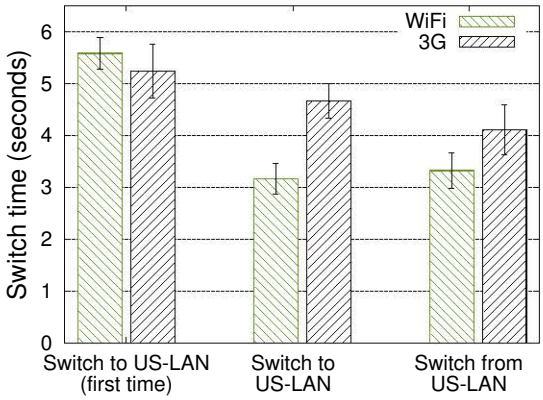
Figure 4.14a shows that the throughput obtained from devices from different manufacturers with different Android versions when only one US-app is in use. The throughput ranges from 1Mbps to 6Mbps. The maximum rate of ∼6Mbps was achieved when using a Samsung i9306 and the lowest rate of ∼1Mbps was achieved when using a Huawei U8950. While this is comparable to cellular throughput in many parts of the world, it is much greater than the cellular data rates that could be achieved when using the same equipment, which was approximately 400Kbps. In essence, although we observed that throughputs are dependant of the device type, as expected User-Stash still achieves significantly faster data rates when compared to using cellular networks.

With current Android smartphones, it is not possible to simultaneously use the cellular network and WiFi network interfaces. Therefore US-app manages the switching between the cellular and the WiFi network interfaces as required. Figure 4.14b il-



(a) Throughput over US-LAN for different device types.

(b) Switch time to US-LAN for different device types.



(c) Switch time to/from US-LAN to/from WiFi and cellular networks.

Figure 4.14: Practical measurements of throughput and latencies.

lustrates this switch time between the cellular and the WiFI networks. The results show that there is a considerable difference in switch time when 1) a device attempts to connect to a US-LAN for the first time ("First connection" in Figure 4.14b), and 2) the authentication parameters for US-LAN are stored under the previously connected AP list ("US-LAN is in the AP list" in Figure 4.14b). For the latter case, it only takes 2 to 3 seconds for all the tested devices to switch to US-LAN from another WiFi network. The switching time is not significant and barely noticeable as the switching process occurs in the background whilst the users are scrolling through the search results or the most popular content. However, when a user launches the app for the first time, and since the US-LAN is not yet in the list of preferred networks of the user device, it takes between 5 and 7 seconds to associate and connect. This is acceptable as it only happens once. Figure 4.14c shows the switch time between US-LAN and cellular networks as an average over two cellular network service providers

we experimented with. As expected, connections to/from cellular networks takes longer than switching to WiFi, due to the additional cellular network signalling. The users experience this switch time delay only when they access content that are not stashed in the US-server and when they search for a particular content in the online content providers. More importantly, these switching overheads are only temporary, since smartphones already start enabling the simultaneous use of multiple network interfaces[5].

4.3.2 Client device energy consumption

The device that hosts US-server is expected to be connected to a power source, and therefore US-server will not be sensitive to the energy usage (e.g., the bus driver plugs the US-server to a power source). In contrast, the US-app needs to be energy efficient. We consider two use case scenarios: (1) downloading content form the US-server, i.e. a stash hit, and (2) the content request results in a stash miss, and US-app downloads the content via the cellular connection.

In order to measure the practical energy consumption, the battery of the device was hijacked to connect to a shunt resistor ($R_s = 15m\Omega$) and then the voltage across the shunt resistor (V_s) was measured using a National Instruments USB-6008, a multifunction DAQ (NI-DAQ)[6]. We performed a measurement per every millisecond and exported the results using NI-LabView. US-app was also configured to log time stamps for the start and the end of event categories shown in Figure 4.15. During the measurements, special caution has been taken not to introduce concurrent background activities. The energy consumption for the duration T is calculated as $E = V_s/R_s \times V_b \times T$, where $V_b = 3.8V$, the battery voltage.

Here, we show energy measurements assuming that the US-app is in the foreground of the device, and that the device is first connected to the US-LAN. We also measure the energy consumption when downloading the same content through a cellular network (3G case). Figure 4.15 depicts the energy consumption for the previously mentioned two use cases normalised by the energy consumption of the 3G case. For the 3G case, we eliminate any streaming protocol and/or foreground user interface related power consumption discrepancies by using the US-app to access videos in the two experiments.

As expected, when using User-Stash, the stash miss scenario results in more power consumption than 3G downloads due to the extra operational steps of checking the

[5]e.g Samsung Galaxy S5 enables to boost access speed through simultaneous downloads through WiFi and cellular networks.

[6]`http://sine.ni.com/nips/cds/view/p/lang/en/nid/201986.`

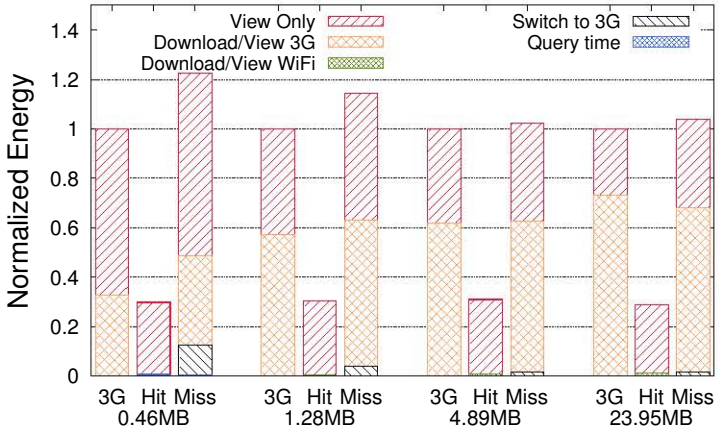


Figure 4.15: Client device energy consumption measurements for the two application scenarios of stash hit and stash miss compared to direct content access through cellular network.

US-server (Query time), and then switching from US-LAN to the cellular network. However, the larger the size of the requested file, the lower the relative energy consumption of the stash miss case and the extra overhead is amortised over the longer download time. On the other hand, a stash hit allows significant energy savings compared to the 3G case regardless of the size of the video. There is a saving of ∼70%, primarily due to higher throughput over the US-LAN (Figure 4.14a). The energy consumption for both download/view over US-LAN and query time are small and is almost indistinguishable in Figure 4.15. Moreover, the amount of energy consumed in the cases of downloading via 3G networks is always larger than that of local User-Stash download although the view only time is comparatively similar in all three cases. This is due to the fact that the mobile device is still at the high power state even after downloading the content for the 3G case.

The results suggest then that the US-app's energy consumption is dependant on the stash hit ratio. If we consider that there are four content size categories (denoted by c), with a popularity likelihood of p_c and a normalised energy consumption of e_c^{hit} and e_c^{miss} for a stash hit and stash miss respectively, the expected normalised energy consumption E can be represented as a function of the stash hit ratio h as follows;

$$E(h) = h \sum_{\forall c} p_c e_c^{hit} + (1-h) \sum_{\forall c} p_c e_c^{miss} \tag{4.1}$$

We notice that $E(h)$ is a linear function of h with a gradient of $\sum_{\forall c} p_c(e_c^{hit} - e_c^{miss})$ and a y-intercept of $\sum_{\forall c} p_c e_c^{miss}$. If we assume that the content popularity of the four content sizes in Figure 4.15 are similar to the PPTV dataset (Table 4.2), $E(h)$ linearly decreases at a rate of 0.77 along with h. Moreover for $h > 0.0968$, the

normalized energy consumption $E(h)$ is less than one. Therefore, if the system is able to provide at least a 10% stash hit rate, users are very likely to save on the device energy consumption. This is a compelling case of the practicality and viability of the proposed system while improving the user QoE and reducing the mobile data traffic.

4.4 Summary

Mobile video traffic has been driving an explosive growth in the mobile data traffic, with users of smart mobile devices struggling to limit their usage to monthly capped data plans. In Section 3.1, we proposed a novel crowd-sourced mobile content delivery mechanism - User-Stash, that enables users to access popular content for free in areas such as public transport where there are no cheap networks such as WiFi. The User-Stash provider selects a set of users to deploy crowd-sourced User-Stash devices (US-server) in public places. The smartphone users can access the US-server's content storage via the mobile app (US-app). The content downloaded by US-app users via cellular network will be uploaded and stashed in to US-servers via a local network hosted by US-server. An advertisement based incentive scheme has been developed for users to become US-server users.

In this chapter, we validate a hypotheses of spatio-temporal correlation of content access using a real-world data set of a video content provider. The content access patterns and characteristics of content are probabilistically modelled for the purpose of evaluating the performance of User-Stash under different operational environments. Combing with the analytical models of user behaviour in public transportation systems, we showed that more than 60% cache hit rate can be achieved during a one hour bus ride regardless of the content popularity distribution and User-Stash achieves higher stash hit rate for short travel distances - e.g. metro type transport services, where there is frequent arrivals and departures of new commuters.

Then, the performance was evaluated against various system constraints such as the available storage capacity in the stash, device manufacturer and operating system dependencies. Popularity based cache replacement strategies such as GDSP performed better than other widely used schemes such as LRU under strict content storage constraints at the US-server device. Moreover, with a 128GB of User-Stash storage, more than 80% of the users reduced their cellular network usage at least by 40%. Finally, throughput, latency and device energy consumption of the US-app was evaluated using the measurements obtained from the experimental implementation on Android mobile devices. The throughput when accessing content locally via

the US-LAN from the US-server varied from 1Mbps to 6Mbps depending on the mobile device manufacturers and capabilities. The results also showed that US-app users lower the device energy consumption compared to accessing the content through cellular networks if the proposed system provides at least 10% stash hit rate. Further details of the implementation of the US-app and US-server components on real smart mobile devices are presented in Chapter 7.

Chapter 5

MobiTribe: Content delivery over low-cost networks

In Chapter 4, we proposed User-Stash that enables the users to access popular content for free reducing their cellular network usage and improving the user QoE. However, User-Stash does not address the issues associated with privacy and user control of data. MobiTribe architecture described in Section 3.2 overcomes the challenge of providing users more control of their data and to protect their privacy by keeping the data away from centralised service providers. 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 traditional distributed systems could be mitigated.

In this chapter, we exploit the time elasticity of social networking content, harness the advanced capabilities of mobile devices and the fact that the user will have access to high speed low-cost network such as WiFi networks to provide cost efficient content sharing mechanisms. The idea is to use a connectivity aware content replication on the users' devices who are highly likely to view the same content there by reducing the costs of distributing the content to a level comparable with centralised systems. Further, this chapter provides a performance analysis of MobiTribe using realistic content creation and consumption models. The results show that peer-to-peer store and forward architectures can provide the same functionality as centralised architectures, but with lower distribution costs and energy consumption. Moreover, it is always more cost efficient than content sharing via non-mobile-optimised distributed social networking platforms. This chapter shows that:

- The content replication problem in distributed peer-to-peer architectures is *NP-Hard* and then present a novel algorithm for device grouping in content replication based on a combination of a bipartite b-matching and a greedy

heuristic, which minimises content replication and maximises content availability whilst ensuring fairness.

- The computational feasibility of the proposed algorithm by theoretical worst case time complexity analysis for scalability.
- The viability of the proposed algorithm such that it is possible to achieve persistent low-cost network availability with only two replicas, using a real-world data traces.
- When compared to alternative social networking approaches, MobiTribe can save more than 75% of the uplink and 45% of the downlink cellular bandwidth, using real-world data sets and analytical modelling of content creation and consumption behaviour.
- MobiTribe mitigates the high device energy consumption associated with peer-to-peer architectures as it increases the usage of high speed low-cost networks.

The remainder of this chapter is organised as follows: Section 5.1 presents the intuition and constraints for device grouping and then, we formally define the content replication problem in peer-to-peer architectures. Section 5.2 presents three device grouping algorithms followed by the evaluation of the performance of content replication in Section 5.3. Then, in Section 5.4 MobiTribe system performance is evaluated in terms of communications cost and energy efficiency through analytically modelling content creation and consumption. Finally, Section 5.5 summarises the contributions of this chapter.

5.1 Content Replication in MobiTribe

5.1.1 Device availability to host content for others

The primary objective of the device grouping is to determine a tribe of devices for content replication such that the tribe maximises the availability of a content via low-cost networks. As described in Section 3.2.1, device grouping for content replication is performed by the CMS periodically or in response to a significant degradation of the availability of a tribe's content. The ability to host content on a device, namely "Device Availability" can be represented as a binary pattern as shown in Figure 5.1. If a set of carefully selected devices combined together to form a storage tribe (mTribe), then as long as at least one device in the tribe is available to distribute

the content to other users, as depicted in Figure 5.1, the content becomes always available.

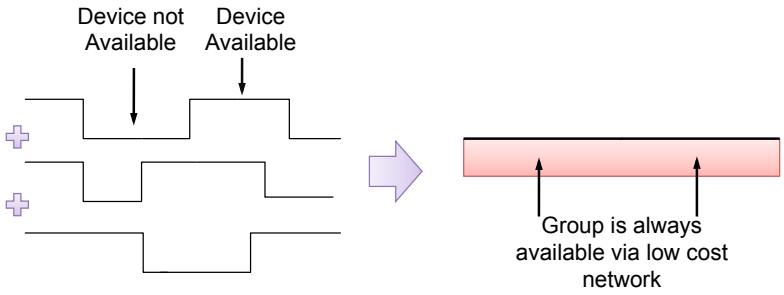


Figure 5.1: Intuition for device grouping based on device availability

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. Each device records user context information and uploads collected data periodically to the CMS. In order to preserve user privacy, all these information is aggregated to a single bit to indicate whether it can to host content for other users. The communication cost can be further optimised by analysing network infrastructure context information available at the CMS such as connected cell ids, network load and relative importance of the traffic. The idea is to use the history of device availability patterns to group users, because users have the regular behavioural patterns.

Let the availability of a device i be A_i^t, where $A_i^t = 1$, if the device is capable of hosting others' content during the time slot t and $A_i^t = 0$ otherwise. Then, the probability of low-cost network availability of a tribe of devices (G) can be defined as follows, where $|T|$ represents the number of time slots considered to infer future availability, namely the "Training Period".

$$P_a(G) = \frac{\sum_{t \in T} \bigcup_{i \in G} A_i^t}{|T|}, \quad A_i^t = \begin{cases} 1 & \text{Available} \\ 0 & \text{Not available} \end{cases}$$

If users have regular behavioural patterns, it is possible to use $P_a(G)$ as a metric to predict future availability because $P_a(G)$ gives the probability of at least one device having device availability at a randomly selected instance to co-host content for other users during the Training Period. Therefore, if $P_a(G)$ is higher than a threshold, the *"Threshold of Pairing" (P_{th})*, we consider that tribe G satisfies the availability constraint and can be used to host others' content in the future. However, in real systems there can be time instances that the exact status of device availability is unknown. Table 5.1 summarises the device availability patterns of two devices

Table 5.1: Contingency Table

		Device i		
		1	**0**	**2**
Device j	Available:**1**	n_{11}	n_{10}	n_{12}
	Not available:**0**	n_{01}	n_{00}	n_{02}
	Unknown:**2**	n_{21}	n_{20}	n_{22}

considering all three states; 1) device available: 1, 2) device not available: 0 and 3) availability unknown: 2. The table contains elements n_{ij} indicating the number of time instances that one device in state i and the other in state j. Hence, the probability of availability of a tribe can be defined as;

$$P'_a(G) = \frac{n_{11} + n_{01} + n_{10} + n_{12} + n_{21}}{|T|}$$

In addition, the content can only be distributed when both end devices are connected to low-cost networks. Thus, we redefine the device grouping metric $P_a^{W_o}$, a modified probability of availability;

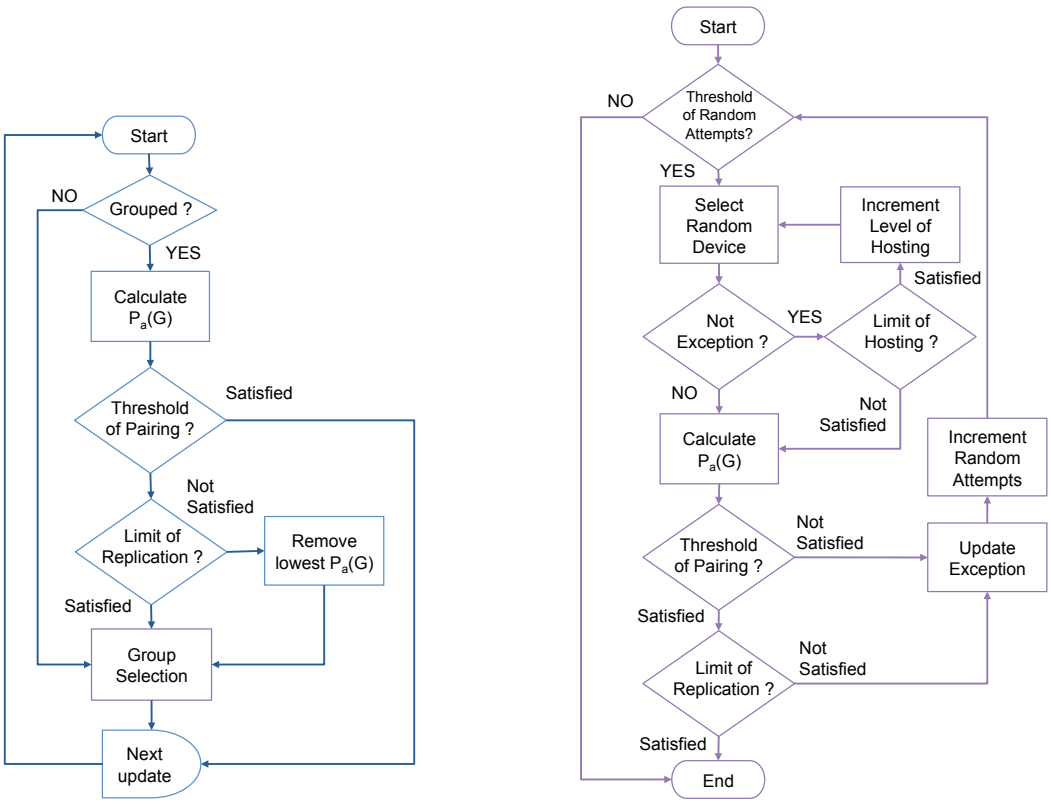
$$P_a^{W_o}(G) = \frac{(n_{11} + n_{01} + n_{10} + n_{12} + n_{21}) + W_o \times n_{11}}{|T| + W_o \times n_{11}}$$

The probability of having low-cost connectivity in more than one device increases with the overlapping weight W_o, which is used to control the impact of overlapping low-cost availability during the group selection.

However, we cannot implicitly select tribes based on threshold of pairing because replication consumes scarce resources. Therefore, we consider two additional constraints in device grouping to ensure the resource limitations are taken into consideration.

1) Minimum Replication: It is obvious that, the availability of the content increases with the level of replication. A high level of replication will disrupt the system performance by creating additional traffic flows and overheads, wasting battery life and storage of mobile devices. Hence, the device grouping trades-off replication to availability. The minimum replication is ensured by defining a *Limit of Replication*, i.e. the maximum allowed size of a tribe.

2) 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



(a) Content replication **(b) Group selection procedure**

Figure 5.2: Heuristic content replication algorithm.

fairly among the devices. The fairness of the system is ensured by defining 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. When these three constraints are satisfied, a device included in the mTribe is said to be *covered*. Thus, the content replication problem becomes *"how to find a maximum cover while satisfying minimum replication and fairness constraints"*.

5.1.2 A heuristic content replication algorithm

First, we study the feasibility of using history of device availability patterns for device grouping while satisfying the above mentioned design requirements. The group selection is carried out using the device availability patterns stored in the CMS. Figure 5.2a shows the flow diagram of a heuristic device grouping algorithm and the following sub sections describe the key decision points with reference to the flow of the algorithm.

If the device is already grouped, the algorithm re-calculates $P_a(G)$ for the group

by matching the most recent availability patterns. If the new P_a is greater than P_{th}, the algorithm move to the next device and if not, the device is moved to the group selection process. As shown in the Figure 5.2a, the algorithm always checks the threshold of replication before the group selection. If the group size reaches the threshold of replication, un-pairing will occur one at a time, by removing the least suitable device to pair with.

Group selection

After checking both of the above thresholds, the algorithm starts the group selection procedure, which is shown in Figure 5.2b. The group selection operates until it finds a group of matching nodes to host the content or until it reaches *Threshold of Random Attempts*. If the initiating device has very low connectivity, it may be hard to find a device to pair it with. When there is large number of devices involved, the algorithm will keep searching for a matching device for a long time. The Threshold of Random Attempts is introduced to reduce the search times. The value for the Threshold of Random Attempts can be decided depending on the available computational resources.

The group selection procedure is designed to select a device or group of devices randomly with the lowest possible level of hosting while ensuring the maximum possible fairness. When selecting a random device, the algorithm ignores the devices in the current group and in the *Exception* category. The devices that have already failed to satisfy the threshold of pairing are included in the Exception category as shown in Figure 5.2b. The selected device or devices calculated P_a followed by its validation with the Threshold of Pairing. If it is not satisfied, the algorithm moves back to select another device after updating the Exception category and the Threshold of Random Attempts.

The replication was designed to be done symmetrically. That is, if one device selects another device to host data, then the reverse is also allowed with out any evaluation. This may increase the convergence time of the system as well as overheads. However, this procedure creates unnecessary replication for devices which have very good connectivity, because that devices will be selected by many other devices to host data. On the other hand, these devices do not need to replicate in many other devices to provide the required availability. Hence, we removed the symmetric replication to reduce the average level of replication in the system.

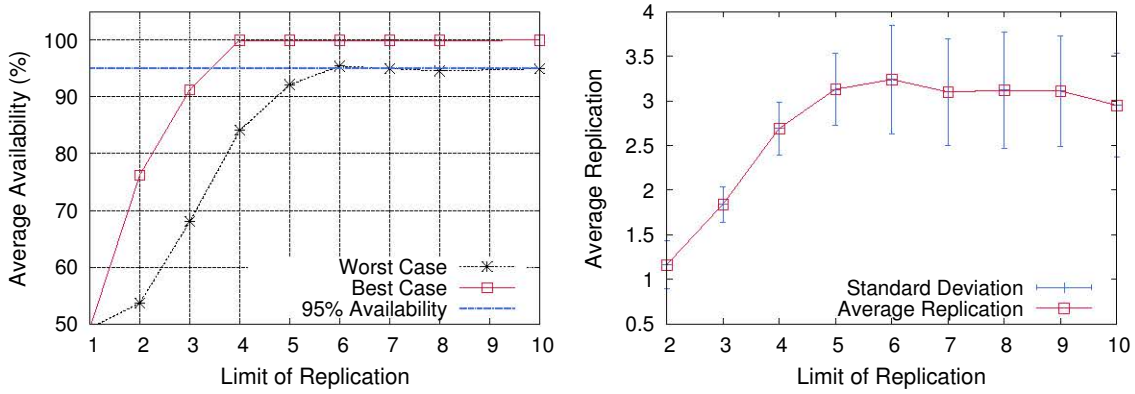
5.1.3 Evaluation of heuristic content replication

For this analysis, the only context information considered is availability of WLAN of the devices. Although there are several traces available at trace data repositories such as CRAWDAD [77], WiFi connectivity traces obtained from normal use of smart phones are rarely publicly available. We used a Rice Community trace data set at [77] and it has WiFi connectivity information with a one minute granularity from 14 HTC Wizard PDA phones. We found another data set from CoSphere trial at [77]. The data set contains information from three different wireless network interfaces cellular, WiFi and Bluetooth. We combined these two data sets to get a larger data set of 26 devices. However, only 21 devices had consistent data through out the period. We identified three states: WLAN available, WLAN not available and trace record not available. We extracted a transition vector for each user from the raw data as a representation of device availability patterns. Although the time period of two traces need not have been be the same to combine them, but it happened to be the same in this case. The granularity of the data was not equal and we had to transform the Cosphere data into one minute granularity to obtain a similar transition vector such that if a device is available for the majority of the time for a given minute, the device is considered as available for that minute.

Simulation setup

The trace data was used to evaluate the performance of the algorithm by dividing the data into two categories: *Training Period (TP)* and *Evaluation Period (EP)*. The TP data block was used to perform the group selection. The performance of the selected group was then evaluated during the EP data block. We considered the minimum TP as one day i.e. it is required to be at least $1440min$ length transition vectors in the CMS to start the replication algorithm. The TP will keep growing with each periodic update of context information. To achieve best performance there should be enough amount of past context information. Hence, we selected the values TP=7 and EP=3 to evaluate the performance of the algorithm. The impact of the TP and EP on the results is discussed at the end of this section. In addition, without loss of generality, Limit of Replication (THR_{rep}) and Limit of Hosting (THR_{host}) were considered as equal during the whole evaluation.

Since P_a is defined by considering all the undetermined instances as "Unavailable", the P_a gives the worst possible availability during the EP, namely *"Worst Case"*. On the other hand, it is possible to remove all undetermined instances from the evaluation and we name it as the *"Best Case"*. Hence, we evaluated the perfor-



(a) Average system availability vs replication for best and worst case connectivity patterns

(b) Average replication vs limit of replication

Figure 5.3: Average availability and level of replication of the selected groups

mance of the algorithm with respect to both best and worst case scenarios. After each execution of the algorithms illustrated in Figures 5.2a and 5.2b, all evaluation metrics were calculated for each selected group and averaged over the groups to obtain the average system evaluation metric. The same procedure was repeated over the entire trace data period by sliding the TP and EP by one day and once again averaged to get the final metric. The repetition procedure ensures the adaptation of the algorithm for different sets of training data sets. Hence, the average metric values can be considered as more generic and realistic.

Results and Discussion

Figure 5.3a shows that almost 100% of average system availability can be achieved for a limit of replication (THR_{rep}) of four or more in the best case scenario. Even the worst case leads to more than 95% availability, if more than six devices per group is allowed. Moreover, the average of replication is always less than the limit of replication, as shown in Figure 5.3b. For the best case, 99.98% availability can be achieved for an average replication of 2.69 with the limit of 4 replicas per user. The system average replication, i.e. average group size, is used as the metric to check the performance for minimal replication. Figure 5.3b shows that the system average replication reaches its maximum at limit of replication of six and never goes beyond that even if the limit of replication increase up to 10. This indicates that the algorithm identifies the best devices to group with and the design goal of minimal replication is achieved. The standard deviation increases with limit of replication as it allows all devices in the cloud to achieve high availability. But, the

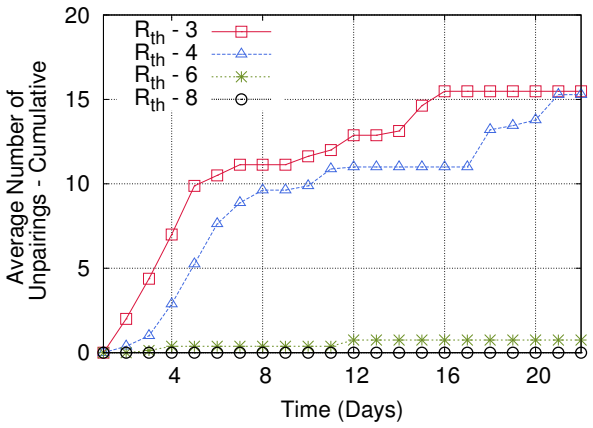


Figure 5.4: Cumulative Un-pairing for different values of Limit of Replication

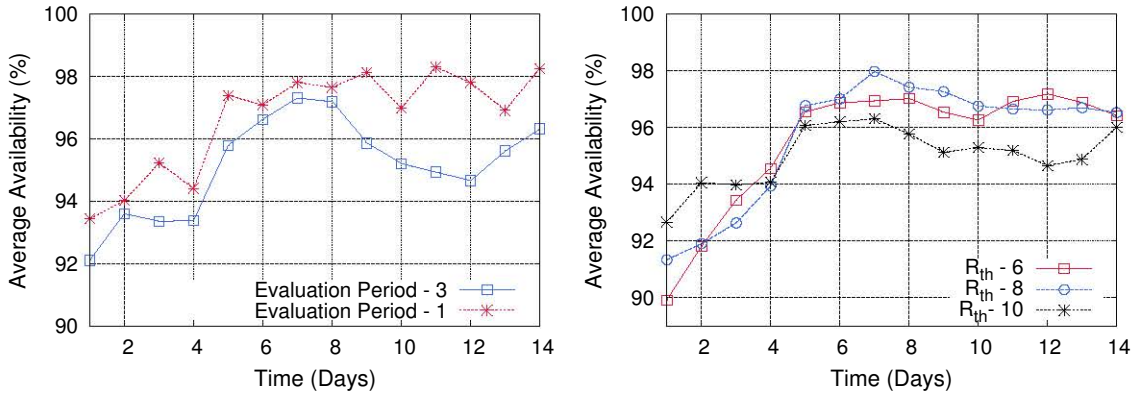
maximum value is 0.5 which is sufficiently low to accept the fact that any device in this particular data set requires only 2-3 replicas.

The availability increases with threshold of pairing (higher the P_{th} tighter the constraint) and reaches a maximum value when $P_{th} = 0.9999$. On the other hand, $P_{th} = 0.9999$ has the highest value of replication. Hence, the trade-off between maximal availability and minimal replication has to be managed appropriately with respect to available resources and behaviour of specific groups of users. In general, if the device pool is large enough, the trade-off will be reduced due to the availability of different connectivity patterns. It is worth noting that the average availability of a single device in this particular data set is 49.21%. The device availability however has been improving. A recent study by Lee et al. [10] suggests that WLAN availability in metropolitan areas is now approximately 70%. That means the algorithm can achieve 100% availability with even less replications, which clearly enhances the feasibility of MobiTribe.

Minimal reshuffling

Every pairing, un-pairing and reshuffling of groups consumes network resources and device resources such as battery power. Therefore, the distribution algorithm should be capable of minimising the number of pairing, un-pairing and reshuffling activities. We evaluate the number of un-pairing activities per day for the particular data traces. An un-pairing activity takes place only if the existing group does not satisfy the threshold of pairing requirement.

The cumulative count of un-pairing activities increases considerably during the system convergence period and remains at that level as in Figure 5.4. That is, there is almost zero un-pairing in the settled system. In addition, there is almost



(a) Impact of EP **(b) Impact of TP**

Figure 5.5: Impact of Training Period (TP) and Evaluation Period (EP)

zero un-pairing activities when THR_{rep} is high enough. Hence, it suggests that it is possible to find stable groups for content replication. The limiting factor would be the frequency of the total system reshuffle, i.e. regrouping all devices as the level of replication may grow up to limit of replication for all groups eventually. Because of the long-tail nature of UGC, older content is less likely to be accessed and do not need to be reshuffled to gain more availability. We propose to regroup devices after certain time period only to replicate newly created content, i.e. older content will not be reshuffled which ensure minimal additional overheads due to replication.

Impact of EP and TP

Since periodic updates of context information suppose to be received daily, it is possible to re-evaluated the groups every day. Therefore, the Evaluation Period should ideally be one day. If one device is unable to send the periodic update, current group should be capable of maintaining the availability for an extra time period until the next periodic update. Therefore, we set the EP as three days for evaluation purposes. Since the EP=1 results shows better performance than EP=3 as shown in Figure 5.5a, it is worth noting that all performance metrics will be much improved in the actual scenario than the evaluation.

The TP will keep growing with each periodic update of context information, but in practice the CMS can not keep large amounts of historic data and it is also not necessary. We evaluated the system performance, percentage of availability for the worst case scenario, by increasing TP=1 to TP=14. As shown is Figure 5.5b, the system takes at least seven days to converge to its optimal performance. After that, increasing the TP does not show big difference of availability where as it increases

computational complexity of the algorithm and storage. We can assume that this is due to weekly life patterns of mobile users. Hence, we selected TP=7 for all evaluation purposes. Also, we trained the system considering a sliding window of seven days and identified that it is possible to maintain the near-best performance with less complexity. In practice, the specific TP value depends upon several factors such as storage, computational power and behaviour of social network.

This analysis shows that it is possible to exploit history of device availability patterns for device grouping. The heuristic algorithm proposed in this section is capable of providing considerable increase in content availability with in a selected tribe.

5.2 Device Grouping Algorithm

In this section we first formally define the device grouping problem subjected to three constraints, 1) Threshold of Pairing, 2) Limit of Replication and 3) Limit of Hosting . Then, a novel scalable algorithm to perform device grouping is proposed.

5.2.1 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 $c : V \to Z^+$. 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, *Threshold of Pairing* (P_{th}). The parameter Δ is the maximum allowed group size, i.e. the *Limit of Replication*. Finally, the capacity function c represents the *Limit of Hosting*, the fairness constraint. The objective is to perform a grouping that covers (spans) as many devices as possible.

Theorem 5.1. 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 [78].

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. □

5.2.2 A tractable special case of device grouping

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, via a reduction to the MINIMUM WEIGHT DEGREE CONSTRAINED SUBGRAPH (MWDCS) problem. This shows that the hardness proof in the previous section cannot be represented as a simpler problem.

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 [79] gave an $O(U(V)|E| \log |V|)$ time algorithm for this problem, where $U(V) = \sum_{v \in V} U(v)$.

Theorem 5.2. 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 [79] involves complex data structures and there are no implementations of it available. However,

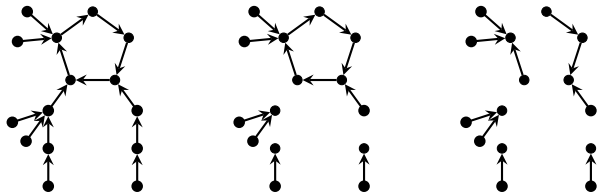


Figure 5.6: Edge pruning in the proof of Theorem 5.3

it can be shown 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 directly to maximum flow). An instance of the b-matching problem is specified by a bipartite graph (A, B, E) and a function $b : A \cup B \to Z^+$. The objective is to find a maximum cardinality subset M of E such that $\deg_M(u) \leq b(u) \; \forall \; u \in A \cup B$.

Theorem 5.3. *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 $c(u') = b(u) \; \forall \; u \in V$.

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, choose an arbitrary edge $(u, v) \in C$ incident on it, and add (u, v') to the matching. Then 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.

In the reverse direction, let M be some b-matching. Construct a directed graph (V, F) such that if $(u, v') \in M$ by adding the directed edge (u, v) to F. These edges represent that device u is *being covered* by v. The goal is to find a solution for DG that covers the same number of devices as M. Let $\deg^+_F(u)$ denote the number of edges in F going out of u, and $\deg^-_F(u)$ denote the number of edges coming into u. Note that for every vertex $u \in V$, $\deg^+_F(u) \leq 1$ and $\deg^-_F(u) \leq c(u)$; this follows immediately from b-matching constraints. Therefore 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 not possible to disregard the directions of the edges in F and use this as our DG solution, because the combined in and out degree of a vertex u, can be $c(u) + 1$. First, we prune unnecessary edges. A node is a *leaf* in F, if $\deg^-_F(u) = 0$. Then, 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, delete the edge going out of u from F iteratively. When it is not possible to delete any more edges, the remainder is a collection of disjoint stars and directed cycles with some additional edges pointing to vertices in the cycles as shown in the second graph in Figure 5.6. If a vertex u in a cycle has an incoming edge from outside the cycle, then remove its outgoing edge from F as shown in the third graph in Figure 5.6.

It can be easily shown that if a vertex u still has its out-going edge, u is a leaf or it was a cycle node with a single incoming edge from another vertex in a cycle. Note that in either case $\deg_F^-(u) + \deg_F^+(u) \leq 2$. For all other vertices with no out-going edges, $\deg_F^-(u) + \deg_F^+ \leq c(u)$. Therefore, if the directions of edges in F are disregarded, the solution is feasible for DG because the $c(u) > 1 \; \forall \; u \in V$ by the assumption made in the theorem statement.

Now the only requirement is to show that every vertex covered by the b-matching is also covered by the DG solution. Note that every time an edge (u, v) is removed from F, there is always another edge (x, u) with $\deg_F^-(x) = 0$; thus, it guarantees 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 the proposed DG solution. □

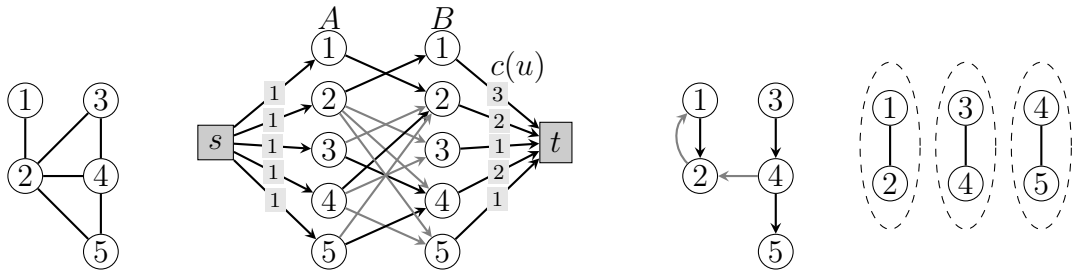
5.2.3 A greedy algorithm for general instances

For general hypergraphs (V, E), the problem cannot be solved in polynomial time, since it is NP-Hard. However, it is possible to approximate the problem of covering the device efficiently using a greedy heuristic with 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 be safely added to C without violating the capacity constraints so as to maximise the number of newly covered elements.

Algorithm 1 GREEDY(V, E, C, D, c)

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

Note that the set systems defined by the feasible solutions of the DG was shown to be Δ-extendible [80] and the maximisation objective is submodular. For such a



(a) Example graph **(b) Maximum flow computation** **(c) Edge pruning** **(d) DG solution**

Figure 5.7: An instance of the DEVICE GROUPING algorithm in practice step by step

problem, Chekuri et al. [81] showed that greedy is a $\Delta+1$ approximation. However, for our special objective function, a slightly better approximation is possible.

Theorem 5.4. *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, assign sets in O to sets in C.

Suppose that GREEDY picks a set e in a given iteration. For each $u \in e$, find some set f_u in O, if any, such that $u \in f_u$; assign f_u to e and remove f_u from O. 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 the greedy choice, when adding e to C the number of newly covered devices cannot be larger than if f_u is added to C. More specifically, $|e \setminus D| \geq |f_u \setminus D|$ for the set D just before adding e to C. Thus, it is possible to 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. □

5.2.4 Summary of device grouping

DG will be carried out at a central entity, namely the CMS as shown in Algorithm 2. Let $H = (V, E)$ is a hypergraph consists of V set of devices. As defined earlier,

Δ is the Limit of Replication and the capacity function $c : V \rightarrow Z^+$ represents the Limit of Hosting, the fairness constraint. The function $b : A \cup B \rightarrow Z^+$ represents the difference between the limit of hosting and the number of that a device belongs to, i.e. $b(u) \leq c(u)$. The set C represents the set of hyper edges $e \in E$ that are considered to be covered and the set D contains the vertices of the covered edges, i.e. the set of devices that are covered, the solution of the device grouping algorithm.

Algorithm 2 MobiTribe CMS$(V, E, C, D, c, b, \Delta, P_a)$

1. $E' \leftarrow \{\text{COMBINATIONS}(V, \Delta = 2)\}$
2. **for** each $e \in E'$ **do**
3. **if** $P_a(e) > P_{th}$ **then**
4. $E \leftarrow E + e$
5. **else**
6. $E \leftarrow E - e$
7. **if** $e \in C$ **then**
8. $D \leftarrow \{D \setminus e\}$
9. $C \leftarrow C - e$
10. $C, D =$Device Grouping(V, E, D, c, b)
11. **if** new context information received **then**
12. MobiTribe CMS$(V, E, C, D, c, b, \Delta, P_a)$

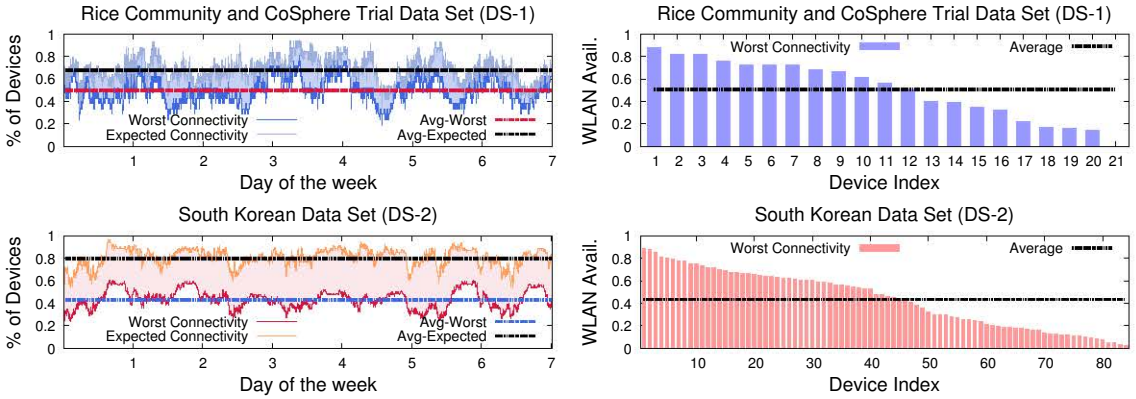
Each device periodically updates the central entity with its recent device availability patterns (A_i^T). From these availability patterns, an undirected graph among all pairs of devices is generated to solve DG for the case of $\Delta = 2$, i.e. the maximum tribe size of two. Figure 5.7a shows an instance of a simple graph created from availability patterns of five devices. Then, a hyper edges between two devices u and v is added, if $P_a(e) > P_{th}$ as shown in step 3 and 4. If the edge does not satisfy P_{th} and already considered as a group previously, it is removed from C and D. Then, the device grouping algorithm is carried out in order to cover remaining devices.

As the first step, a bipartite graph is constructed to implement a single bipartite b-matching reduction as explained in Theorem 5.3. For each edge (u, v) in the original graph, edges (u, v') and (v, u') are added in the bipartite graph as shown in step 2 to 4 in Algorithm 3. The set A consists of devices that are not already grouped and the set B consists of the devices with $b(u) \geq 1$, i.e. the devices that can pair with at least one more group. Since the b-matching can be reduced to a maximum flow computation, a flow network is generated by introducing two extra nodes s and t as in Figure 5.7b. Each incoming edge to node t from a device u is assigned the

capacity equal to $b(u)$, i.e. the limit of hosting. All the other edges are assigned the capacity value of one. In the example network in Figure 5.7b, though device 2 is connected to all other devices, it can only host the content of two others. Thus, the limit of hosting ensures that device 2 only takes part in a maximum of $c(2) = 2$ tribes. Once the maximum flow from s to t is computed, we prune extra edges from the resulted directed graph as illustrated in Figure 5.6 and step 9 to 12 in Algorithm 3. In this particular example, edges $(4, 2)$ and $(2, 1)$ are pruned from the resulting graph as shown in Figure 5.7c. Finally, the direction of the edges are ignored and the resulted graph is considered as the solution for the DG problem for the case of $\Delta = 2$. $b(u)$ is updated for all devices that have assigned new groups as shown in the step 15. As the next step, if there are any ungrouped devices, the GREEDY approximation algorithm (Algorithm 1) proved in Theorem 5.4 is used to find tribes for the remaining devices.

Algorithm 3 DEVICE GROUPING(V, H, C, D, c, b)

1. Let s, t be dummy vertices
2. $A \leftarrow \{v : v \in \{V \setminus D\}\}$
3. $B \leftarrow \{v' : v \in V \cup b(v) \geq 1\}$
4. $E \leftarrow \{(s, u) : u \in \{V \setminus D\}\} \cup \{(u, v') : (u, v) \in H\} \cup \{(v, u') : (u, v) \in H\} \cup \{(u, t) : u \in B\}$
5. $c(s) \leftarrow c(u) \leftarrow 1$
6. $c(u') \leftarrow b(u)$
7. M=MAXIMUM FLOW(E, s, t)
8. $F \leftarrow \{(u, v) : (u, v') \in M\}$
9. **while** $\exists\,(u, v) \in F$ such that $deg_F^-(u) = 0$, $deg_F^-(v) \geq 1$ and $deg_F^+(v) = 1$ **do**
10. prune outgoing edge from v
11. **while** $\exists\, cycle \in F$ **do**
12. prune $\lfloor |cycle|/2 \rfloor$ edges
13. $C \leftarrow \{e : e \in F\}$
14. $D \leftarrow \{u : u \in \bigcup F\}$
15. **for** each $u \in F$ **do**
16. $b(u) \leftarrow (b(u) - deg_F(u))$
17. C, D=GREEDY(V, H, b, C, D)
18. **for** each $u \in F$ **do**
19. $b(u) \leftarrow (b(u) - deg_F(u))$
20. **return** C, D



(a) Percentage of devices which have WiFi connectivity

(b) Percentage of WiFi availability of each device

Figure 5.8: Percentage of devices which have WiFi connectivity

5.3 Evaluation of Content Replication Algorithm

Effectiveness of device grouping can be determined by 1) the aggregated low-cost network availability of the selected groups, namely the mTribes, 2) the amount of replicated content and 3) the number of tribes that a single device belongs to. The higher the low-cost network availability, the lower the replication, the smaller the tribe membership of devices, the more effective the device grouping. Since these three parameters depend on user behaviour, time and location, the analysis needs to consider real mobile users' connectivity patterns. We use three data sets as described in the next section to evaluate the effectiveness of the proposed DG algorithm. The context information used for low-cost network selection was limited to WiFi connectivity.

5.3.1 Data sets

The Rice Community [77] trace data set has WiFi connectivity information with a one minute granularity for 14 users in Houston for 6 weeks. The CoSphere trial [77] also contains WiFi connectivity patterns of 12 users for 6 weeks. Since these data sets are individually too small to evaluate device grouping, we combined these two data sets and created one data set (DS1). We assume that these users are social networking friends in the same time zone for this evaluation. The data set from South Korea (DS2) has WiFi connectivity patterns of 97 iPhone users spanning over 18 days with a 3 minute granularity [10].

When a device has WiFi connectivity, the device is considered to be available

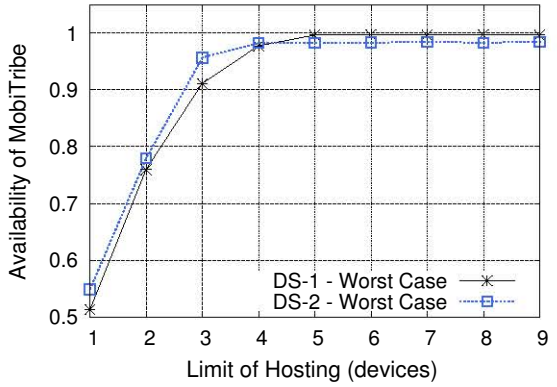
via a low-cost network. Neither data set has consistent connectivity patterns, i.e. there are time instances when the exact state of WiFi connectivity is "unknown". We consider all unknown states as representing WiFi unavailability to evaluate the *"Worst Case"* performance. On the other hand, we can remove the "unknown" states from the evaluation, which then gives the realistic low-cost network availability of the devices, namely *"Expected Case"*. 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 5.8a for both expected and worst cases. The difference between expected and worst case is higher in DS2 suggesting that it has more unknown states compared to DS1. 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% as shown in Figure 5.8b. Altogether, the data sets contain 1470 days of connectivity patterns from 105 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 to be a representation of a real operating environments for the evaluation of the proposed DEVICE GROUPING algorithm.

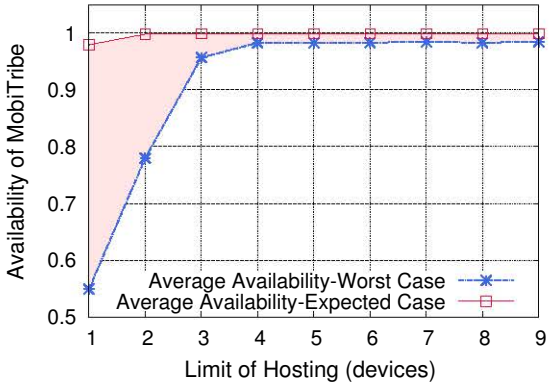
The trace data sets are divided into *training* and *evaluation* periods. The training period is used to perform the tribe calculation and the performance of the selected tribes are evaluated during the evaluation period. The device grouping is carried out daily. If one device in the selected tribe is available via WiFi, the content stored in the tribe is considered to be available. The availability of all selected tribes are calculated and the same procedure is repeated over the entire trace data period to obtain average system availability, *Availability of MobiTribe*. It has been observed that the fairness constraint described in Section 5.1, i.e. the maximum number of tribes that a single device belongs to, namely the *Limit of Hosting*, affects the efficiency and the performance of grouping. The main reason for this is that the devices with high availability would be preferred by many others to host their content. Hence, we measured the availability of a tribe against the limit of hosting.

5.3.2 Availability-Replication-Fairness trade-off

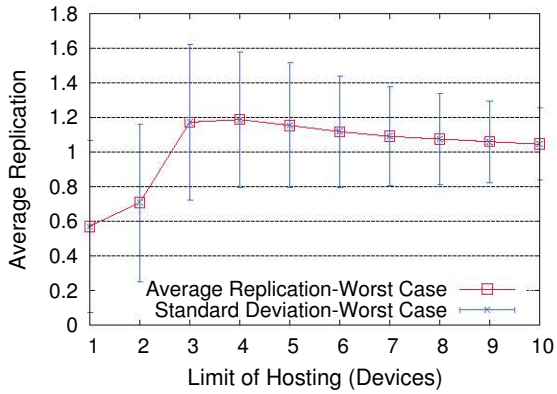
Figure 5.9a shows the availability variation of DS1 and DS2 with respect to limit of hosting. Note that both data sets give similar availability behaviour irrespective of the different connectivity characteristics of the data sets. This indicates that the DEVICE GROUPING algorithm have the potential to adjust to different types of



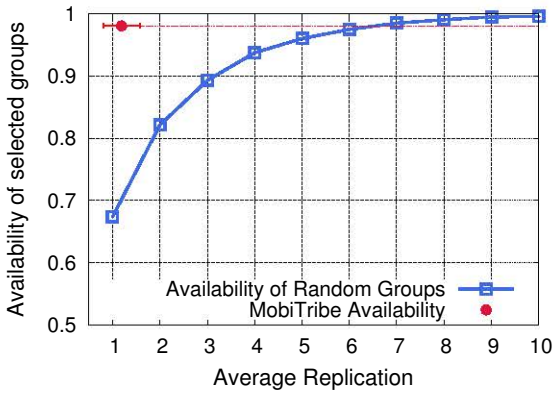
(a) DS1 and DS2 comparison



(b) Expected and worst case availability



(c) Average Replication



(d) Availability of randomly selected groups

Figure 5.9: Availability and replication of MobiTribe vs limit of hosting

mobile environments. Figure 5.9b 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 5.9b. In the remainder of this section, 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.

Average Replication of the system is the average number of replicas of 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 R_a as shown in Figure 5.9c. 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 R_a. The maximum average replication required to satisfy the availability requirement of $P_a = 0.95$ is $R_a = 1.19$ with the standard deviation of 0.39. Thus, in this particular social network, each

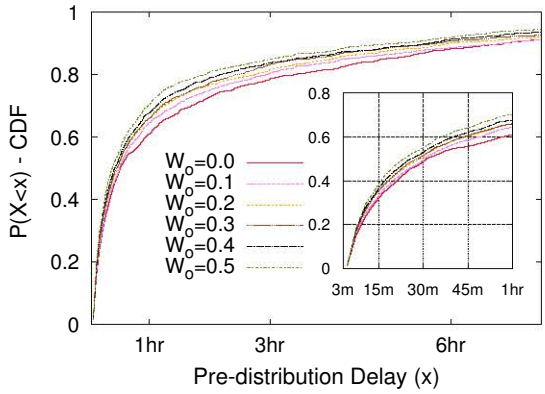


Figure 5.10: The CDF of the pre-distribution delay

user only needs to group with one or two other users to increase the availability via a low-cost network.

This analysis shows that nearly 100% availability of content over low-cost networks can be achieved with approximately 2 replicas, if the devices are grouped into tribes as described in Section 5.2. This minimises the overhead of replication in terms of network bandwidth, storage and battery consumption. Other systems, such as Safebook [19] or MyZone [46], do not select devices for replication based on the connectivity patterns. Instead, they use random replication, which results in a higher number of replicas than MobiTribe to provide a similar level of availability. To validate this we performed a random group selection. As shown in Figure 5.9d, random selection requires 7 replicas to provide the same availability level that is provided by MobiTribe with 1.19 replicas. Although the results are for the small scale trace data sets used, the number of replicas will be the same or even lower for large scale real environments because the probability of matching complementary connectivity patterns increase with the number of users due to the availability of variety of connectivity patterns of devices. For content sharing through distributed architectures such as Diaspora [18], which do not replicate the content, the availability would depend on whether the user's device has low-cost network availability at the time another user wants to access the content. Therefore, if the device has no network access or is switched off, the content will not be accessible at all.

5.3.3 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 Section 3.2.1, 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 assuming ideal communication channels with unlimited bandwidth. The CDF of the delay for different overlapping factors are shown in Figure 5.10. 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 DG algorithm. The probability of the delay being less than 1 hour is over 60% irrespective of W_o. In [82], it has been observed that 55% of Flicker content is uploaded after a lag of more than one day. This indicates that majority of content creators naturally tolerate content delivery delays greater than 6 hours. Hence, we believe that the content delivery delays presented in Figure 5.10 would be practically significant as there is 90% probability for the pre-distribution delay being less than 6 hours.

5.3.4 Scalability of content replication algorithm

As summarised in Section 5.2.4, the device grouping process consists of three phases. We first evaluate the asymptotic worst case time complexity for each phase.

1) Hypergraph creation: Similar to the notation in Section 5.1, let a hypergraph $H = (V, E)$ where the vertex set V represents the set of devices. The hyperedge set E is the collection of subsets of V satisfying the availability constraint, $P_a > P_{th}$. Figure 5.9c shows that we can cover the majority of devices by using bipartite b-matching algorithm, i.e. the maximum allowed group size $\Delta = 2$. Hence, we limit the rank of the hypergraph to 2 in the initial graph creation phase, which makes it a simple undirected graph. In general, if a system has n devices, there can be a maximum of $\frac{n(n-1)}{2}$ number of pairs that we need to calculate P_a for the creation of edge set E. P_a calculation does not require heavy computation as it only requires the calculation of the union of two binary signals and one division for each pair of users. Therefore, the upper bound for hypergraph creation would be $O(n^2)$ in general. In the data sets considered, there were $|E| = 2905$ sets out of 3528 pairs, which satisfied the $P_{th} = 0.95$ for $|V| = 84$ devices.

2) Bipartite b-matching algorithm: The scalability of the DG is dominated by the calculation of the bipartite b-matching algorithm. Bipartite b-matching is directly reduced to the maximum flow, thus the complexity is bounded by the maximum flow computation. There are many fine tuned algorithms available for maximum flow computation in the literature [83]. 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 [83], the upper bound for the bipartitie b-matching would be $O\left(|A \cup B||E| \log \frac{|A \cup B|^2}{|E|}\right) \approx O(n^3)$.

3) Greedy replication: We perform greedy replication selection only for the devices which are unable to group together when $\Delta = 2$. Let there be m remaining devices. Then, for each m there are $\binom{n-1}{\Delta-1}$ combinations to evaluate the threshold of pairing (P_{th}) and threshold of hosting ($c(u)$). Hence, the upper bound for the greedy algorithm would be $O(m\binom{n-1}{\Delta-1}) \approx O(mn^{(\Delta-1)})$. We perform greedy selection until $m \to 0$ while incrementing Δ.

Altogether, the scalability of the device grouping would be $O(n^2) + O(n^3) + O(mn^{(\Delta-1)})$ for all three phases. For the case of $\Delta \leq 3$, the asymptotic time complexity is upper bounded by $O(n^3)$ because the fastest growing component is the bipartite b-matching. For the case of $\Delta > 3$, the time complexity of device grouping is dominated by greedy replication and is given by $O(mn^{(\Delta-1)})$. Since it is possible to cover majority of devices in phase two, m would be considerably low compared to n and drastically decrease with each increment of Δ. In addition, Δ does not need to increase more than 4 in real-life networks. However, m can be as large as n in theory. Therefore, the theoretical worst case time complexity increases to $O(n^{\Delta})$. This is highly unlikely in real-life networks because of high WiFi availability as observed in this evaluation.

It has been shown that the active relationships a person can maintain is about 150 (Dunbar's number [84]) and validated for many online social networks. Even in large systems like Facebook, the median friend size is approximately 100 [85], i.e. n would be around 100. Since the device grouping is performed at the CMS, where there are no processing power and energy limitations compared to mobile devices, $O(n^{\Delta})$ and $O(n^3)$ can be considered as practical asymptotic complexities for the NP-Hard DG problem. Furthermore, it is expected that the selected tribes in MobiTribe will become quasi static over the time due to the regular behaviour of people [86]. Hence, the tribe calculation will be carried out only for users who have changed their behavioural patterns such that the tribe attributes go below the required thresholds. In practice, the number of users of the service increases gradually and as a result tribes will only be calculated for newly added users. Therefore, it is evident that the device grouping algorithm will scale up with the increasing number of users.

Content popularity is another factor which could impact scalability during content sharing. As the size of friends will be limited in human social networks (Dunbar's number [84]), the content popularity is also limited by the number of friends, if not shared as public content. The user population of data sets used for the evaluation of

Table 5.2: Content creation and consumption model

Content creation model	
Creator	Everyone
Amount per week	142MB [89][88]
Inter-arrival-time	Exponential (mean=2 hours) [10]
File size	Gamma (scale=2, mean=2MB)[71]
Content consumption model	
Consumer	Random
Content popularity	Pareto Type II (80-10 rule) [90]
Content access delay	Gamma (scale=2, mode=15min to 1 day)

MobiTribe is similar to the current number of friends of Facebook users. Moreover, only 20% of them accounts for 70% of interactions [87]. Therefore the content popularity among groups of friends will not have any scalability issues for MobiTribe. When the same content is popular among a number of groups, it is considered as separate content for each group. In addition, the group of friends in MobiTribe are assumed to be collaborative, i.e. once someone downloaded content she will be willing to share it with others participating in the peer-to-peer content distribution process (seeding). For highly popular content and for creators with a large number of friends, there will be a higher number of copies available on mobile devices. Therefore, the peer-to-peer protocol scales with demand similar to BitTorrent.

In addition, the number of different content items generated by users within a group could impact scalability because of storage capacity. Although the usage of spare storage capacity in a mobile device has very low relative cost for both the users and operators, it is obviously not comparable to the storage available via centralised services. Therefore, to avoid accumulation of content, the old replicated content will be removed from user caches using a standard cache replacement scheme such as Least Recently Used (LRU).

5.4 Performance of MobiTribe Architecture

The overall performance of MobiTribe content delivery mechanisms can be quantified by the savings of cellular bandwidth and the energy consumption of devices. To analyse the system performance in the future, we use the projected mobile network data traffic figures of Cisco [88] to model the content creation and consumption behaviour of mobile users.

5.4.1 Content creation and consumption model

Table 5.2 summarises the content creation and consumption model parameters. It is predicted that an average smartphone will consume 2.6GB of data per month by 2016. Since the uplink (UL) traffic percentage is around the 23-25% mark [89], we assume that the average smartphone user generates 612MB (23% of 2.6GB) per month and the generated content are Gamma distributed [71] with the mean content size of 2MB. It has been shown that the internet access time of mobile users of DS2 are exponentially distributed in [10]. Therefore, the inter-arrival-time for the content generation is obtained from an exponential distribution such that it is distributed throughout the whole trace duration.

It has been observed that 10% of the top popular videos in YouTube, account for 80% of views [90]. Hence, we use the Pareto type II distribution for content popularity such that each generated content is accessed by a number of randomly selected consumers from a Pareto distribution. Then, these selected consumers will access the content after a content access delay which again has a Gamma distribution. We evaluate the performance by varying the mode value of the Gamma distribution, namely *"Peak Content Access Delay"* (CAD_{peak}) to understand the performance under different content access delay conditions. Note that there can be many instances that the content is accessed right after sharing, even if the CAD_{peak}=1 day, because CAD_{peak} is only the mode value of the Gamma distributed content access delay. We evaluate the performance against the *true positive rate* (TPR) of prediction which is the portion of correctly predicted consumers out of the actual consumers and we assume the TPR=0.7.

5.4.2 Simulation setup

We developed a custom simulator in Python to simulate a content sharing network among the mobile users in DS2 (WiFi connectivity patterns of 97 iPhones as described in Section 5.3.1). We consider WiFi data rate as 1.97Mbps, which is the mean data rate observed in the experiment [10] and 3G data rate as 200Kbps, which is the same data rate used in the simulation for this environment in [10]. For this evaluation, we assume that both content creators and consumers are mobile devices, and the mobile devices are always-on and always-accessible via a cellular network. We use this as the communication model for this simulation along with the workload in Table 5.2 to evaluate the effectiveness of each content delivery mechanism.

Figure 5.11 illustrates the main components of the simulator. *User Generator* maintains a dictionary of users with their device availability patterns and *mTribe*

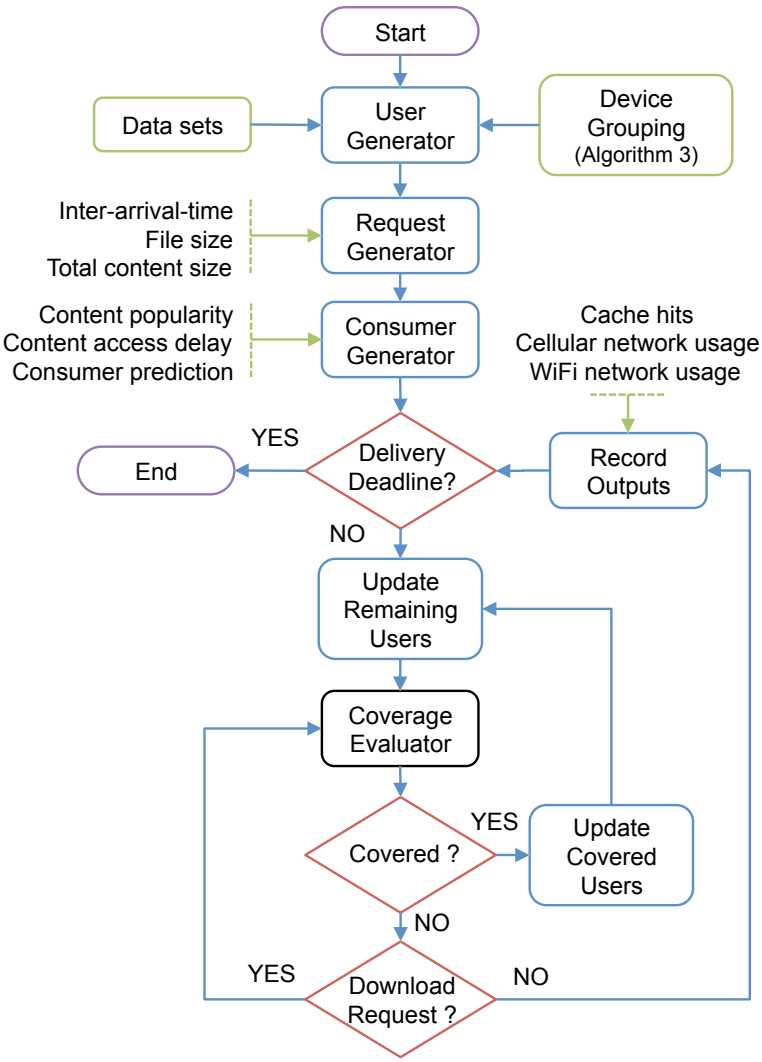


Figure 5.11: Illustration of the Python simulator for MobiTribe.

devices as values using the data set and output of *Device Grouping* algorithm (Algorithm 3). Then, *Request Generator* assign content request events to each user based on the content creation model summarised in Table 5.2. For each request, a set of consumers is assigned based on the content consumption models. Next, the simulator loops through time until it reaches the content delivery deadline, which is considered to be one day. For each generated content, the simulator maintains two dictionaries, namely *Covered Users* and *Remaining Users*. At the outset, only the content creator is added to the *Covered Users*. All users in the *mTribe* and *Predicted Consumers* based on the given TPR are added to the *Remaining Users*. A user in the *Remaining Users* is added to the *Covered Users*, if the user receives the complete content to be downloaded, which is evaluated by the *Coverage Evaluator*. In the case of MobiTribe content delivery mechanisms, a user is considered to be covered, if the

user had aggregated overlapping WiFi network availability for (file size in bits/1.97) seconds with users in the set of *Covered Users*. Then, the simulator checks whether there is a content download request from a consumer. If the consumer is originally a predicted consumer, there is a high possibility that the consumer is already moved to *Covered Users*, i.e. the content is prefetched to the device. If not, the all relevant outputs such as cache hit count and cellular bandwidth usage are calculated and recorded based on the network availability of the consumer device and also the *mTribe* devices. Then, the simulator moves to the next time instance.

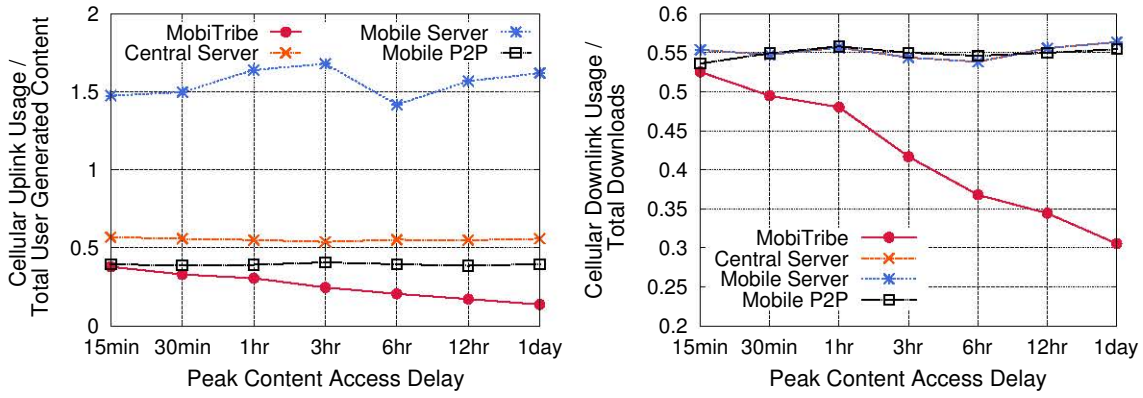
Furthermore, to compare the performance of MobiTribe with other alternative content delivery mechanisms, the *Coverage Evaluator* is extended to evaluate other content delivery mechanisms such as;

1) Central Server: where the content is uploaded to a centralised server at the time of sharing. With a centralised server, the shared content will be immediately available to the consumers. Majority of the current content sharing schemes via social networking services such as Facebook, YouTube and Google+ fall under this category.

2) Mobile Server: where the content will not be uploaded to a centralised server. With a mobile server, the content consumers access the shared content via creator's own server on one of their own devices. The distributed social networking architectures such as Diaspora [18] and PrPl [65] fall under this category.

3) Mobile Peer-to-Peer: where all content consumers will also host the downloaded content to help others to access the content (seeding). Many services which make use of protocols like BitTorrent, fall under this category.

In addition, we consider that on-the-spot data traffic offloading via WiFi networks is possible in all architectures. Since all of the above architectures support immediate content availability, we do not consider delayed offloading to a centralised server. Note that even in MobiTribe, although the content pre-fetching takes time, the shared content is immediately available via the creator's device. Therefore, delayed offloading is not directly comparable with MobiTribe. On the other hand, the high replication overheads of the non-mobile-optimised replicating systems, such as SafeBook [19] and MyZone [46] would almost certainly lead to higher bandwidth and energy 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 which is not aware of network availability patterns. Hence, we do not consider those architectures for this evaluation.



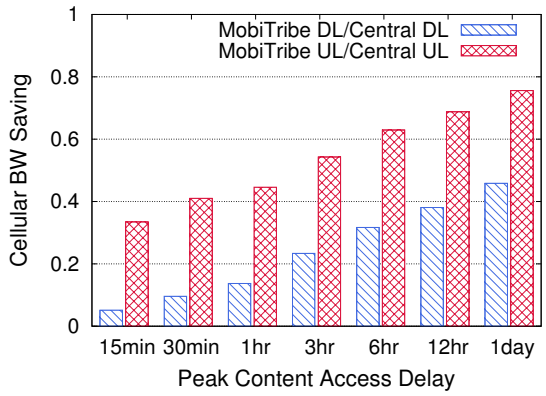
(a) Cellular UL **(b) Cellular DL**

Figure 5.12: Cellular UL and DL bandwidth usage

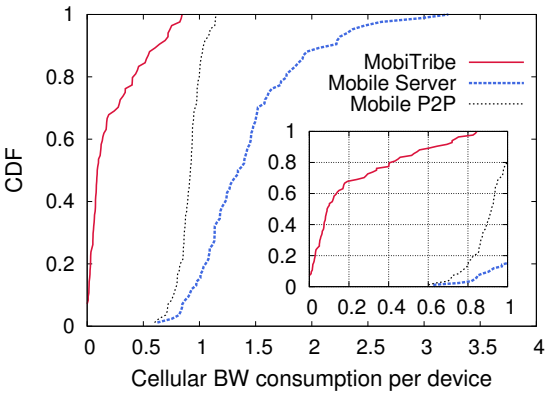
5.4.3 Cellular bandwidth consumption

In this section, cellular bandwidth consumption by MobiTribe will be compared with alternative systems. In Figure 5.12, alternative architectures do not show a significant change in uplink or downlink cellular usage with increasing content access delay. In contrast, MobiTribe provides considerable reduction of cellular network usage when there is higher content access delay. This is because a large content access delay allows content creators to pre-distribute the content to the mobile devices in a *mTribe* via low-cost networks. The mobile server architectures consume the highest uplink bandwidth as shown in Figure 5.12a, which results in approximately 150% increase in transferred data. This is because of that the creator is responsible for serving content requests. Similarly, downlink bandwidth usage of alternative schemes do not vary with content access delay as shown in Figure 5.12b. This is because none of the alternative schemes perform predictive pre-fetching and thus entirely depend on the connectivity pattern of the downloading device. Content access prediction in MobiTribe leads to significant downlink bandwidth saving due to successful pre-distribution of content to the consumers via low-cost networks. When the peak content access delay is one day, 30% of the total downlink traffic is transferred via cellular networks in MobiTribe, compared to approximately 55% with the alternative architectures. In [82], it has been observed that 55% of Flickr content is uploaded after a lag of more than one day. Hence, we believe that MobiTribe will result in significant bandwidth savings.

We compare MobiTribe cellular bandwidth saving against the central server architecture which is the most widely used architecture. Figure 5.13a shows that MobiTribe saves 76% of the uplink cellular bandwidth compared to a central server

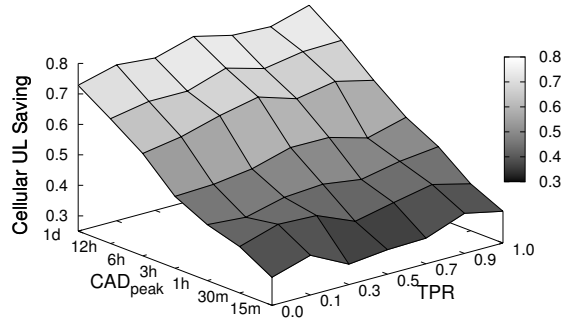


(a) Bandwidth saving

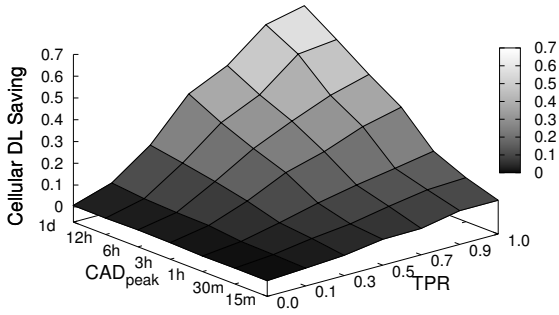


(b) Cellular network usage of devices, CAD_{peak}=1day

Figure 5.13: Bandwidth saving and cellular network usage of devices compared to central server



(a) Cellular UL saving



(b) Cellular DL saving

Figure 5.14: Cellular bandwidth saving against CAD_{peak} and true positive rate of prediction (TPR)

architecture, if the CAD_{peak}=1 day. Though the percentage saving for the downlink is 46%, the amount of bandwidth saved by the downlink is higher than the uplink because of the larger downlink data volume. In total, MobiTribe saves 54% of the cellular bandwidth compared to a centralised server architecture with on-the-spot data traffic offloading, when the peak content access delay is one day.

Figure 5.13b illustrates the cellular bandwidth consumption of each device in MobiTribe normalised by cellular bandwidth consumption of a central server. In this particular social network, 90% of the devices consume only 60% of the cellular bandwidth compared to the central server, i.e. 90% of these devices can save at least 40% of the cost of cellular communications. On the other hand, in both mobile server

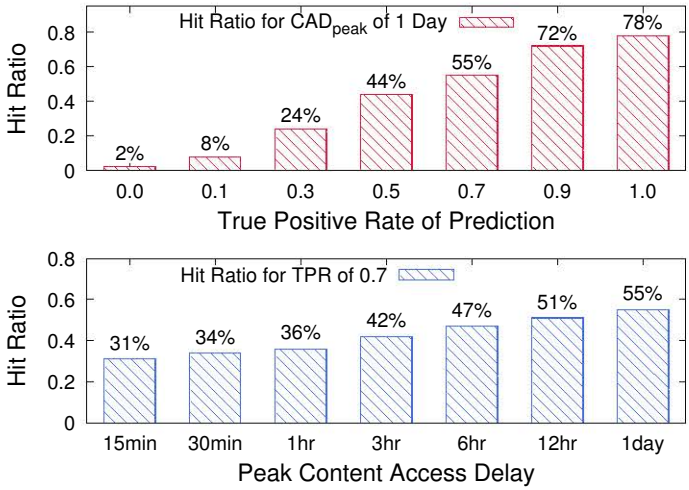


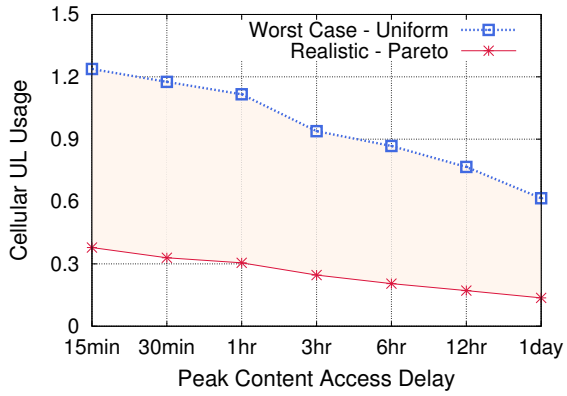
Figure 5.15: Success rate of Cache Hits

and mobile peer-to-peer architectures there are devices that will see an increase in cellular bandwidth consumption. In systems like Diaspora, 85% of the devices will consume more cellular bandwidth compared to the central server architecture, which affects both monetary cost and device energy consumption. Thus, MobiTribe satisfies its key design goal of cellular bandwidth saving. In the following sections, we evaluate how the cellular bandwidth saving of MobiTribe varies with algorithmic and content consumption model parameters.

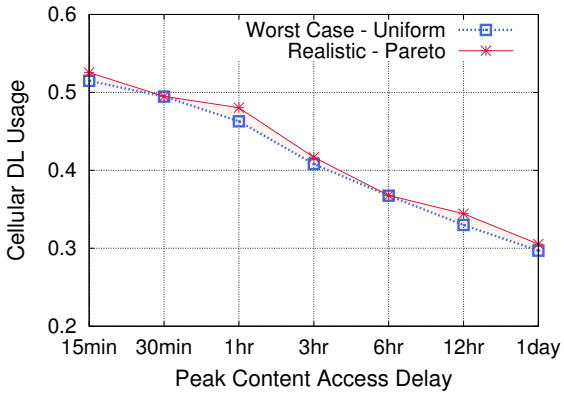
Effect of the content access prediction

Since replication of content is done via low-cost networks, the false predictions do not affect cellular bandwidth consumption. Hence, we evaluate the performance against the true positive rate (TPR) of content access prediction. Figure 5.14a shows that cellular uplink bandwidth saving is not heavily dependent on TPR, whereas bandwidth saving significantly increases with the content access delay. In contrast, downlink shows significant saving only when both the TPR and the content access delay are higher as shown in Figure 5.14b. However, the TPR has more impact on downlink saving than the content access delay.

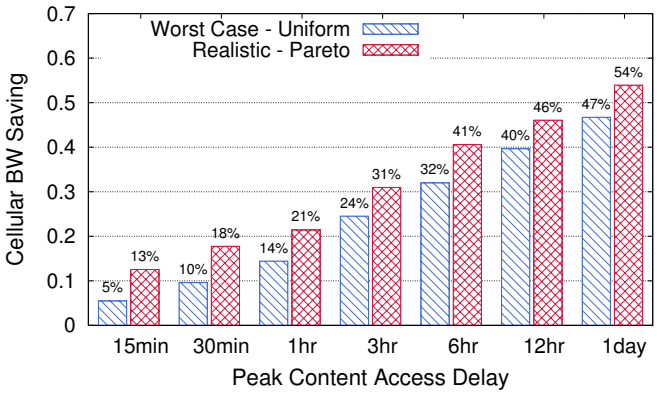
Since the content is to be pre-distributed to devices of users who are also likely consumers, users can access the content instantly. Figure 5.15 depicts the variation of cache hit ratio with respect to the two main limiting factors - TPR and content access delay. According to the Figure 5.15, the TPR significantly improves the cache hit ratio, whereas the impact of content access delay is relatively low. When the TPR is zero, the cache hit ratio is as low as 2% and increases up to 78% for



(a) Cellular UL bandwidth consumption



(b) Cellular DL bandwidth consumption



(c) Bandwidth saving against centralised systems

Figure 5.16: Effect of content access popularity

completely accurate prediction. Note that even with the 15min content access delay, 31% of users are able to access the content instantly for a TPR of 70%. Overall, a high average cache hit ratio, higher than 50%, can be achieved by MobiTribe, which can be considered as an extra benefit to the users on top of the cost savings.

Effect of the content popularity model

The worst possible content popularity model in terms of bandwidth consumption is the one where everyone in the group is accessing all generated content. In Figure 5.16a and 5.16b, cellular bandwidth consumption for the worst case content popularity model and the Pareto model is presented. The uplink shows significant increase in bandwidth usage for the worse case model, downlink follows similar usage for both models. In contrast, the cellular bandwidth usage of centralised server architectures do not change with the content popularity model since it uploads the content to a central server regardless of the demand. Figure 5.16c depicts the cellular bandwidth

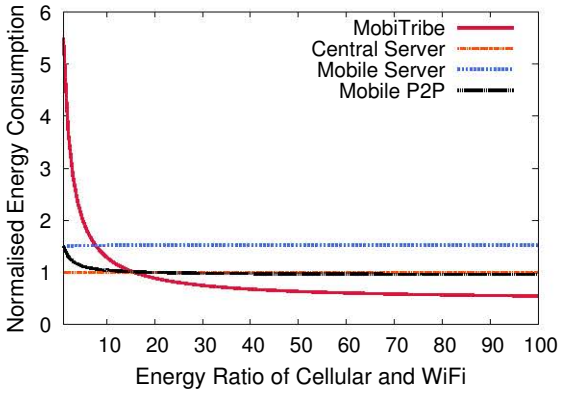


Figure 5.17: Architecture comparison, FPR=0.1, CAD$_{peak}$=1day

saving of MobiTribe for the worst case content popularity normalised by a centralised server architecture. The total bandwidth saving does not drop significantly for the worst case. When the CAD$_{peak}$ is 1 day, the cellular bandwidth saving for the worst case is 47% compared to the 54% for a realistic Pareto distribution. Thus, in this particular social environment, MobiTribe will be able to save approximately 50% of the cellular bandwidth irrespective of the content popularity distribution. Since the data trace contains 1470 days of connectivity patterns from 105 users in three different cites in three different times, we believe that the performance of MobiTribe improves independent of the social environment.

5.4.4 Energy consumption

Content replication incurs more data transfers among devices than a centralised architecture. However, this overhead traffic is mainly transferred through low-cost networks in MobiTribe. Hence, the energy gain in MobiTribe depends on the energy efficiency of WiFi over cellular networks. It has been shown that the energy consumed by WiFi, (E_w) is lower than those of 3G networks (E_{3g}) in majority of practical networks [41], [43]. However, the the energy ratio E_{3g}/E_w can vary between 1 to 100 depending on status of the cellular network at the time of the transfer [41]. This is mainly due to the heavy traffic load in cellular networks leading to lower speeds and thus taking longer time to complete data transfers. To overcome these, the analysis assumes that the energy consumption is proportional to the time spent on each network and evaluates the performance with respect to the energy ratio E_{3g}/E_w.

Figure 5.17 shows the energy consumption of each architecture, normalised by the energy consumption of on-the-spot offloading to a centralised server. The mobile

server architecture always consumes approximately 1.5 times more energy than a centralised server. The mobile peer-to-peer architecture performs slightly better when $E_{3g}/E_w > 20$. In contrast, normalised energy consumption of MobiTribe has an exponential decay and has a 50% lower consumption than a central server architecture for higher E_{3g}/E_w.

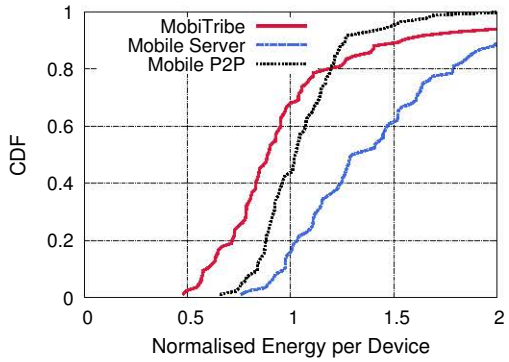
Even though MobiTribe consumes almost the same energy as a centralised server architecture when $E_{3g}/E_w = 20$, approximately 70% of devices still consume lower energy than a centralised server architecture as shown in Figure 5.18a. Less than 10% of devices consume more than 1.5 times energy. Though the mobile peer-to-peer architecture has a similar energy consumption pattern as a central server architecture, 80% of the devices consume more energy than the MobiTribe when $E_{3g}/E_w = 20$. In the mobile server architecture, 80% of the devices consume more energy than the centralised server architecture. Thus, it is clear that mobile server architectures such as Diaspora and fully distributed peer-to-peer architectures are not suitable for mobile environments. In summary, MobiTribe has an energy efficiency similar to the centralised server architectures. However, it is more efficient than all alternative distributed content sharing architectures.

Effect of the content popularity model

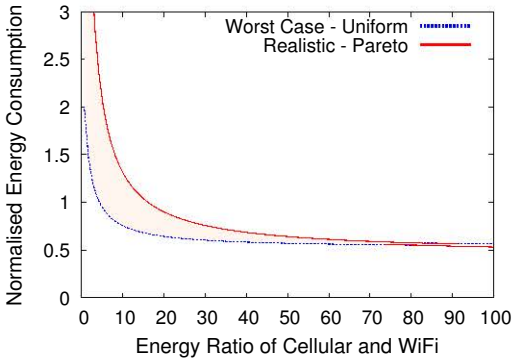
The worst case content popularity model turns out to be more energy efficient than the realistic Pareto model as shown in Figure 5.18b. In the worst case, when everyone in the group is accessing all the generated content, the false positive rate (FPR) becomes zero, which is the portion of incorrectly predicted consumers out of the actual consumers. Hence, there will not be any redundant data transfers or any replication overhead. The energy consumption becomes steady after $E_{3g}/E_w > 20$. Thus, the content popularity distribution stands out as the controlling factor for the trade-off between cellular bandwidth and energy consumption.

Effect of the false positive rate of prediction

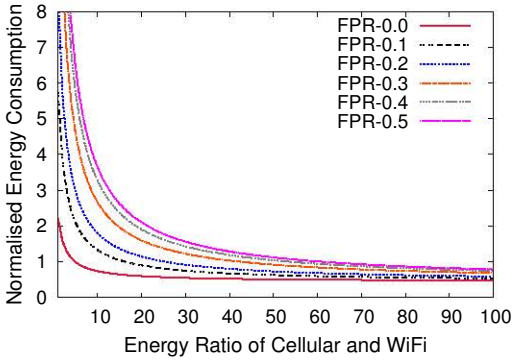
False positives in content access prediction creates redundant data transfers and results in extra energy consumption. Figure 5.18c shows the variation of energy consumption of MobiTribe with the FPR of the prediction algorithm, which could be as much as 2.1 times higher than the centralised server when $E_{3g}/E_w = 20$. Hence, the FPR can be considered as a critical factor for energy consumption. However, a higher FPR increases the availability of the content via low-cost networks. Also, amount of false positives can be reduced at the cost of increasing false negatives.



(a) **Comparison of energy usage per device, $E_{3g}/E_w = 20$, FPR=0.1**



(b) **Effect of content popularity, FPR=0.1**



(c) **Effect of false positive rate.**

Figure 5.18: Evaluation of normalised energy consumption. TPR=0.7, CAD_{peak}=1day

It is possible to extend MobiTribe to pre-fetch the content when devices are connected to a power source, which will reduce the impact of replication energy. In fact, mobile device charging analysis presented in [91] illustrates that majority of users charge devices during the night. Also, the majority of the users have WiFi access during the night because of the availability of home WLANs.

5.5 Summary

Smartphones could be used for UGC sharing and host/share distributed social network data, if the challenges of high communication costs and battery usage were solved. A novel mechanism was proposed for addressing these challenges by providing continuous availability of content over low-cost network connections. The content is replicated in a set of mobile devices forming a mobile private storage tribe such that at least one device is always available to server the content via a

low-cost network connection. The devices for storage tribe are selected from the friends of the content creator, i.e. the users who are supposed to consume the content, while reducing the replication overhead. We showed the content replication problem in distributed peer-to-peer architectures such as MobiTribe is NP-Hard. We then developed a new scalable algorithm which minimises content replication and maximises content availability whilst ensuring fairness.

The performance of the algorithm was evaluated using three real world data sets and realistic content creation and consumption models. Even with these small scale data sets, it was shown that persistent availability of content via low-cost network can be achieved with approximately two replicas, which will become lower with larger variety in real scale environments. The theoretical worst case asymptotic time complexity of the proposed content replication algorithm is shown to be $O(n^{\Delta})$, where n is the number of friends of the content creator and Δ is the limit of replication per content. In addition, the results from 97 iPhones showed that MobiTribe saves 76% of the uplink and 46% of the downlink cellular bandwidth compared to current widely used centralised server based content sharing architectures with on-the-spot data traffic offloading when the peak content access delay is one day. Moreover, 90% of mobile users can save at least 40% of the cost of their cellular communications. Finally, we showed that device energy consumption decays exponentially with energy consumption ratio between low-cost networks and cellular networks. Approximately 70% of the devices consume lower energy than a centralised server architecture when WiFi networks are 20 times energy efficient than cellular networks.Therefore, we conclude that MobiTribe can provide a mobile optimised distributed content sharing platform for mobile social networking.

In Chapter 7, we show the details of the practical realisation of MobiTribe on Android devices, Mac and Windows desktop platforms. The app is integrated with the popular social networking services such as Facebook, Twitter and Google+ enabling the users to interact with friends using existing services whilst preserving and protecting user privacy and user control over own data.

Chapter 6

Yalut: Content delivery over opportunistic networks

The primary objectives of this thesis are to provide cost-efficient, privacy-aware and timely distribution of mobile content without negatively impacting the usability and online services eco-systems. In Chapter 4, we proposed and evaluated mechanisms, namely User-Stash, to access popular content without using their cellular data plan at locations where there are no low-cost networks. Then, in Chapter 5, we provided mechanisms, namely MobiTribe, to enable mobile decentralised social networking on smart mobile devices to provide more control of their data and to protect their privacy. In MobiTribe, the cost of content distribution is mitigated by exploiting the low-cost networks (e.g. WLANs) connectivity patterns of mobile devices, irrespective of the location of the users.

This Chapter provides mobile content delivery mechanisms to further reduce the cost of content distribution by exploiting the geographical locations of the users. As described in Section 3.3, there are many MSN applications where users are clustered into interest based communities that are location specific, e.g. shoppers in the same mall, spectators in a sports event, students in a university, etc. In these scenarios, it is possible to use opportunistic direct wireless connectivity among the devices due to the geographical proximity. Despite offering many advantages for mobile users, such as by providing connectivity when there is no direct access to a network [13], opportunistic content delivery in MSNs have not seen wide spread adoption primarily for two reasons. Firstly, there is an inherent reluctance by the users to interact with strangers or third parties, due to security and privacy concerns. Secondly, due to the unbounded high delivery latency, i.e. being unable to access content when none of the couriers of content are in the vicinity, when a user wants to access or share content. In Section 3.3, we propose a novel content delivery architecture, Yalut that addresses both privacy and latency concerns of opportunistic

communications systems, and enables the provision of efficient distributed MSN services. In particular, we propose to exploit content replication to bridge the gap between the user and the social networking *friends*, i.e. the friends who are not in the vicinity of the user, thereby reducing the content delivery latency and increasing the delivery success rate.

The proposal uses a set of devices, *replicators*, which are selected from the members of the MSN to replicate the shared content using the available lowest cost network. The replicators are then used to propagate the content to other users opportunistically. Compared to transferring via networking infrastructure which typically has a higher monetary cost, the proposed scheme significantly lowers the overall cost of content delivery in MSNs. In addition, it enables content pre-fetching which enhances quality of experience of the users by shortening startup delays and minimising the need for re-buffering. On the other hand, content replication consumes scarce storage capacity on mobile devices and increase redundant data transfers. Therefore, the trade-off of content availability and cost of storage and distribution has to be effectively managed when replicating content. This chapter makes the following contributions;

- Shows the inefficiencies of opportunistic content dissemination in MSNs even under unlimited networking and storage resources through real-world and synthetic trace driven simulation study.

- Provides a formal definition of content replication, which maximises content availability and minimises replication and shows this to be NP-hard.

- Presents a community based greedy algorithm for efficient content replication by taking advantage of routine behavioural patterns of mobile users. A set of dynamic time-aware centrality metrics are proposed to identify most influential users within a community. Such metrics are based on a dynamic weighted contact graph, composed of opportunistic user encounters with contact duration and initial contact time information.

- Shows that it is possible to provide delivery rate of 80% with less than 10% replication under variety of practical MSN environments, through extensive trace driven simulations using both real world and synthetic trace data sets. Despite infrastructure supported replication, the proposed hybrid content dissemination ultimately saves approximately 60% of infrastructure bandwidth usage compared to pure opportunistic dissemination.

- Shows that random content replication provide considerable results and it has been further exploited to develop a component based random content replication algorithm.

- Using a human mobility simulator, the evaluation shows that just 10% of consumers significantly increases the content delivery to almost all consumers. Moreover, content pre-fetching via opportunistic contacts provides a 80-90% success rate, increasing the user QoE. Moreover, the robustness of proposed content replication strategies under different user collaboration and knowledge of the contact patterns of the MSN users are also evaluated.

In the remainder of the chapter, we first present a simulation study of the effectiveness of opportunistic content dissemination in MSN in Section 6.1. Then the formalisation of the problem of content replication in opportunistic environments such as Yalut is presented in Section 6.2. Section 6.3 presents the dynamic centrality metrics and community based content replication algorithm. Section 6.4 evaluates the performance of proposed metrics and content replication algorithm. Based on the results of the first evaluation, a new component based random replication algorithm is presented in the next section (Section 6.5) followed by a simulation based performance evaluation in Section 6.6. Finally, Section 6.7 concludes the chapter.

6.1 Effectiveness of opportunistic content dissemination in MSNs

In this chapter, we focus on location-based MSNs where the members are clustered in nearby physical locations. The idea is to improve the timeliness of the opportunistic content dissemination by taking advantage from always-on networking infrastructure. As the first step, to investigate the effectiveness of opportunistic dissemination in MSNs where the users are clustered in location specific communities (location-based MSNs), we consider a scenario where a content *creator* wants to share content with all other users, *consumers* in the location-based MSN. The content is then distributed to others in store-carry-and-forward manner emulating a peer-to-peer distribution. We evaluate this scenario using human contact patterns from both real-world trace data sets and synthetic mobility models with realistic content creation and consumption patterns.

Table 6.1: Content creation and access model

Content creation model	
Amount per week	142MB [4]
File Size	Gamma(scale=2,mean=4MB)[92]
Inter-arrival time	Exponential (mean=3.5 hrs)[10]
Content access model	
No. of consumers	Pareto Type II (80-10 rule)[85]
Consumer location	Random distribution
Transfer rate	2 Mbps [10]
Delivery deadline	3 days

6.1.1 Content creation and access model

We consider that each user in the Dartmouth trace data set generates content over two months as indicated in Table 6.1. Cisco has predicted that an average smartphone will consume 2.6GB of data per month by 2016 [4]. Though the amount of UGC is predicted to increase, the ratio between upload and download data has remained between 23-25% for the last few years [4]. Therefore, we assumed that the average smartphone user generates 142MB per week, considering 25% of 2.6GB is evenly distributed among 30 days. The inter-content generation-time, or the inter-arrival-time of content is reported to be exponentially distributed [10] and therefore these 142MB of content items are assumed to be distributed throughout the week with a mean inter-arrival-time of 3.5 hours. The size of each generated content is characterised by a Gamma distribution similar to [92] with a mean content size of 4MB.

The number of consumers is selected based on a Pareto distribution since both content popularity in YouTube [92] and degree distribution in Facebook follow power-law models [85]. In general, the popularity of content in Facebook is limited by the number of friends of the creator. If one of the friends is allowed to re-share it will again be considered as a newly generated content. The consumers are randomly selected among the total user population and thereby the locations of the consumers are also randomly distributed. Since WLAN is the most common opportunistic network type, we consider a practical data transfer rate of 2Mbps [10]. Further, we take content delivery deadline to be 3 days, to evaluate the best case scenario in terms of delivery success rate for this evaluation. In fact, 50% of Facebook users login every day[1] indicating that delivery delay should be less than 1 day for at least

[1]http://blog.kissmetrics.com/facebook-statistics/

50% of the users.

Let a dynamic contact graph generated by these data sets be $G_t = (V, E_t)$ that changes its topology over time $t \in (1, 2, \cdots, \Delta)$, where Δ is the content delivery deadline or the lifetime of the content. V is a set of consumers and an edge $e \in E_t$ exists among two consumers if they are within communication range of each other at time t. Consider a content *creator* c wanting to share content via a MSN application/service with all consumers $u \in V$. A consumer is said to be *covered* if he/she has received the full content before the content delivery deadline of Δ.

It was further assumed that (1) no initial content replication is performed and (2) the consumers are collaborative, i.e. once content is downloaded by a consumer, she/he will be willing to share it with other consumers. We assume that the content is disseminated when a user carrying content (an infected user) meets a user who does not have the content (an uninfected user), similar to epidemic dissemination, with unlimited resources, i.e. unlimited bandwidth and infinite buffers. This presents the best case scenario in terms of *coverage* for this evaluation[2]. Further, we assume all users are collaborative and interested in downloading the content. Therefore, there are no relay users who only help to disseminate the content. Then the *coverage* is measured as the portion of successful deliveries from the total consumers. We repeat this experiment considering each user as a creator and iterate over throughout the trace data sets.

6.1.2 Data sets

We use two real-world human mobility traces from a university environment and various other environments are emulated using synthetic contact patterns, generated by a human mobility simulator for opportunistic environments. The details of the traces are the following:

Dartmouth: The contact traces are generated from Dartmouth campus data sets [93]. We consider that if two users are connected to the same WiFi access point, these two users can exchange information as described in [94]. We use two months of data, collected from January to March 2004 that has contact patterns of 1146 users. This university based contact trace is a representation of a sparse MSN due to the large geographical area.

Sigcomm: This data set contains the contact traces from 76 smartphone users, running Mobiclique [95] application during the conference Sigcomm 2009. The application run opportunistic device discovery at every 120 ± 10 seconds using Blue-

[2]Effects of these factors are further evaluated in Section 6.4

Table 6.2: Simulation parameters used in the HCMM

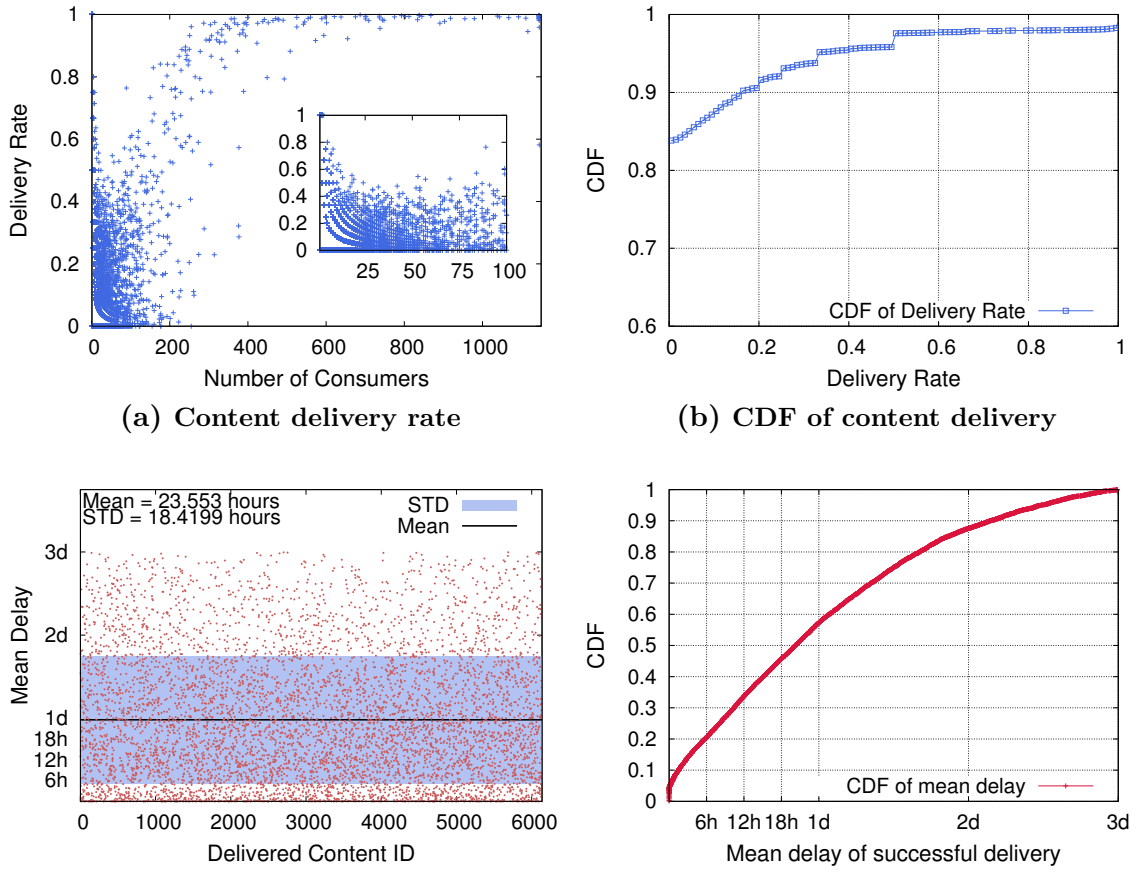
HCMM simulator setup	
No. of users	10-500
No. of groups	10
Rewiring Probability	0.1
Remaining Probability	0
Simulation time	1 hour
Simulation Area	2000×2000 units
Cells	20×20
Radius of a user	250 units
Speed of a user	10 units/step

tooth. We consider connectivity patterns only among these volunteered 76 smartphone users. In contrast to Dartmouth, the Sigcomm scenario represents a dense MSN due to the smaller geographical area of the conference venue and higher social interaction.

HCMM: Home-cell community based mobility model generates contact patterns by considering social and location attractiveness of human mobility where all users have a home-cell and they are attracted towards the home-cell while they are mobile [96]. In real-world, the home-cell can be the work place or home of individuals. Each user is assigned to a home-cell at the beginning and it can be changed at each reconfiguration time. We do not change the home-cell of users during the simulation time, i.e. reconfiguration time > simulation time. The simulation parameters are summarised in Table 6.2. Network scenarios with different number of users are considered. At each scenario, the users are evenly divided into 10 groups. The users in the same group are considered to have a social relationship and also users can make relationship with users from other groups based on the value of the rewiring probability, which is assigned to be 0.1 for this evaluation. These relationships among users determine the user mobility. We consider that the probability of remaining in a non-home-cell is zero assuming all users would come back to their home-cells. We use the HCMM synthetic simulator with these parameters to generate different MSN environments for evaluation purposes in the remainder of the paper.

6.1.3 Coverage without initial content replication

We consider two scenarios of longer delivery deadline of 3 days and shorter delivery deadline of 1 hour to study the effectiveness of opportunistic content delivery rate and delay in different application scenarios.



(a) Content delivery rate

(b) CDF of content delivery

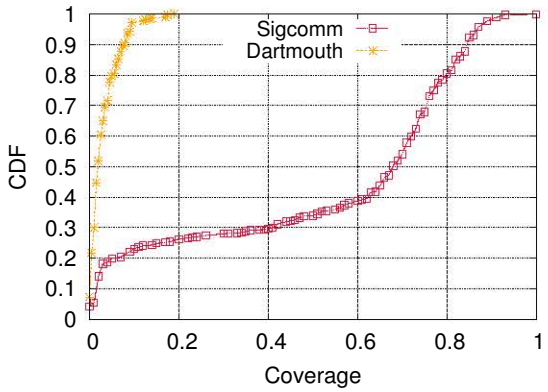
(c) Mean delay of delivered content

(d) CDF of mean delay

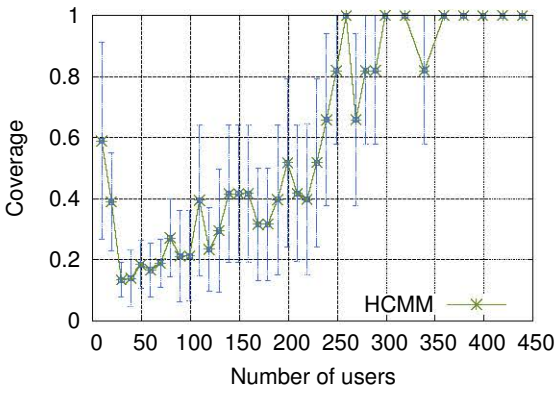
Figure 6.1: Effectiveness of opportunistic communication in content dissemination, $\Delta = 3$ days.

Longer delivery deadline of 3 days:

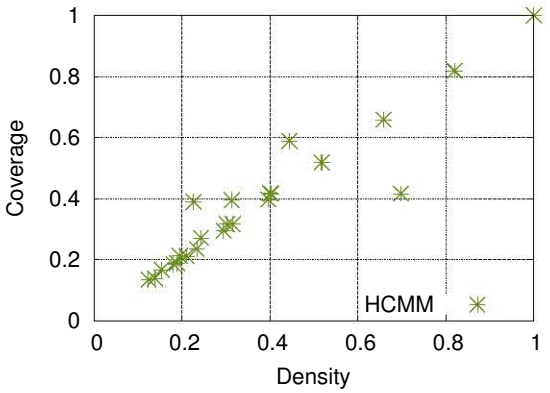
For longer delivery deadline, we use Dartmouth data set due to its larger geographical area. If content can be opportunistically disseminated from a creator to a consumer through one-hop wireless communication before the delivery deadline of 3 days, we consider it as a successful delivery and otherwise a failure. The main factor that affects the delivery rate is the number of consumers as shown in Figure 6.1a. For a large number of consumers, the delivery rate is almost 100% as there are enough consumers to collaborate in content dissemination. However, the number of consumers are often lower in UGC sharing and social networking due to power-law distribution [85], which makes the delivery rate very low for a majority of the instances. As shown in the cumulative distribution function of delivery rate in Figure 6.1b, the delivery rate is zero for approximately 84% of content. For each successful delivery, the content delivery delay from the content generation time is illustrated in Figure 6.1c. It shows that the mean delivery delay is approximately 1



(a) Coverage of opportunistic dissemination.



(b) Opportunistic coverage variation with number of users in HCMM generated contact patterns.



(c) Opportunistic coverage as a function of density of the connectivity network. Density$= 2|\bigcup_{t=1}^{\Delta} E_t|/(|V|(|V|-1))$

Figure 6.2: Effectiveness of opportunistic communication with shorter deadline of delivery delay, $\Delta = 60min$**.**

day and Figure 6.1d shows the probability of mean delay being less than one day is approximately 60% among the successful deliveries, which is 16% out of all content generated.

Shorter delivery deadline of 1 hour:

First, we compare the best case scenario for Dartmouth and Sigcomm users, where all users are considered as consumers. Figure 6.2a shows that even in this best case distribution, it is not possible to cover all users within this limited time window. For Dartmouth, the probability of the coverage being less than 10% is 0.9 and the maximum achieved coverage is less than 20%. For Sigcomm, the coverage is much higher, but 50% of users have less than 70% of coverage. This illustrates the difference between these two extreme (dense and sparse) MSN environments.

To evaluate the variation of delivery rate with respect to sparseness of the commu-

nity, we created different simulation environments by varying the number of users in the HCMM simulator[3]. The higher the number of users, the more dense the connectivity network as well as the coverage as shown in Figure 6.2b. When the number of users are very low (less than 30), they naturally stick together and the coverage is slightly higher. When the number of users are very high (more than 250), the coverage is more than 80% of the network. At 350 users, the coverage becomes 100% and the standard deviation of the coverage approaches zero, i.e. every single user can reach all others through opportunistic communication.

In Figure 6.2c, we plot the same results from HCMM generated traces with respect to the density of the aggregated connectivity graph. We consider that there is an edge between the two users, if they are within the communication range of each other at least once during the one-hour time window. Intuitively, the higher the density, the higher the *coverage*, as shown in Figure 6.2c. We can cover all users only when the network is a complete graph (density=1), which is a highly unlikely scenario in real-world MSNs. The trace data sets from Dartmouth and Sigcomm can be considered as two extremes of real-world densities of 0.014 and 0.22 on average.

Despite many advantages, the results show that the opportunistic dissemination is not effective in propagating content with short lifetimes in location-based MSNs when the number of consumers is low. This is the most likely case due to the power-low distribution of the number of consumers. Similarly, majority of the research work in opportunistic content dissemination in MSN are based on networks with less than 100 users [70], [95]. In such cases, it is possible to use content replication to improve the performance of the content delivery. However, since content replication on helpers consumes more network resources due to the use of infrastructure based communication, there is an obvious trade-off between delivery performance and content replication overhead. Thus, the content replication challenge is to *maximise the content delivery rate with limited content replication.*

6.2 Content Replication

6.2.1 Formal definition of content replication

Table 6.3 summarises all the symbols that are used in this section. Consider a dynamic contact graph $G_t = (V, E_t)$, where V is a set of users and E_t is an edge set at time slot $t \in (1, 2, \cdots, n)$; an edge $e \in E_t$ exists among two users if they are in the communication range of each other at time t. Without loss of generality the

[3]It can be done by varying any parameter such as area, speed of a user, radius of a user, etc.

Table 6.3: Summary of symbols for content replication

Symbol	Description
c	content creator
u, v	users
V	set of users
G_t	dynamic contact graph at time t
E_t	set of edges at time t
e	an edge in E_t
Δ	delivery deadline
P_t	set of propagators at time t
$\alpha(u)$	minimum contact duration required for user u to receive full content from content propagators
$\sigma(c)$	consumers covered only by the creator c
$H(c)$	set of selected helpers for the creator c
$\lambda(c)$	limit of helpers for the creator c

length of each time slot is considered as one unit, which represents the minimum duration in which there is no change in the topology. Suppose a *creator* $c \in V$ wants to share content via a mobile social networking application/service with other users, i.e. *consumers*, in $V \setminus c$. A consumer is called *covered* if it receives the full content within the content delivery deadline of Δ time slots. Consumers are assumed to be collaborative and they become content *propagators* only after being covered. Let $\alpha(u)$ be the minimum total contact duration required by a consumer u to receive the full content from content propagators. We denote $P_t \in V$ as the set of content propagators at time t. When there is no initial replication, $P_1 = c$. Then, if a user u has to receive the full content, the aggregated contact duration of u with propagators should be greater than $\alpha(u)$,

$$\text{i.e.} \sum_{t=1}^{\Delta} \mathbb{I}((u,v) \in E_t \text{ for some } v \in P_t) \geq \alpha(u), \tag{6.1}$$

where the indicator function,

$$\mathbb{I}(\textit{statement}) = \begin{cases} 1 & \text{if } \textit{statement} = \text{true} \\ 0 & \text{otherwise} \end{cases} \tag{6.2}$$

Hence, the set of consumers covered by a creator c is:

$$\sigma(c) = \left\{ u \in V : \sum_{t=1}^{\Delta} \mathbb{I}((u,v) \in E_t \text{ for some } v \in P_t) \geq \alpha(u) \right\} \tag{6.3}$$

Consider a set of helpers $H(c) \in V$ for a creator c. Thus, the creator and the helpers are the initial set of propagators, $P_1 = H(c) \cup c$. The objective is to cover the all consumers with minimum number $\lambda(c)$ of helpers. Then, our CONTENT REPLICATION (CR) problem is to minimise the cardinality of the set P_1 such that it covers all consumers in V, formally;

$$\begin{aligned} &\text{Minimise} && |P_1| \\ &\text{subject to} && \sigma(c) = V \setminus P_1 \end{aligned} \tag{6.4}$$

Here, we show that the CR problem is computationally NP-Hard even for a simple instance of a static social graph, where $E_t = E \ \forall \ t$ and $\alpha(u) = 1 \ \forall \ u \in V$, i.e. the full content can be transferred in a single contact. This is similar to best case scenario where there is unlimited bandwidth and zero bit error rates.

Theorem 6.1. CR *is NP-Hard even when* $E_t = E \ \forall \ t$ *and* $\alpha(u) = 1 \ \forall \ u \in V$.

Proof. We show that *minimum dominating set* is polynomial time reducible to CR problem. Let $G'' = (V'', E'')$ be an undirected graph. A *dominating set* of the graph G'' is a $D \subseteq V''$ such that every vertex $u'' \notin D$ is adjacent to at least one member of D. The dominating number $\gamma(G'')$ is the cardinality of the smallest dominating set. For a given positive integer k, the decision problem of whether there exist a $\gamma(G'') \leq k$ is one of the well-known NP-Complete problems [97].

Recall the dynamic contact graph $G_t = (V, E_t)$. Since we assume that $E_t = E \ \forall \ t$, G_t becomes a static undirected contact graph $G = (V, E)$. In addition, $\alpha(u) = 1 \ \forall \ u \in V$ makes that a vertex u can be covered if there is at least one edge to the set of initial propagators in P_1. Then the decision problem of CR is to find whether there is a set of initial propagators P_1 of size at most $|P_1| = k$ which covers maximum number of consumers $\sigma(c)$. The coverage is maximised only when $|\sigma(c)| = |V \setminus P_1|$, i.e P_1 has to be a minimum dominating set of size k.

This follows immediately that if there is a solution to the decision problem of CR, there should be a solution to the minimum dominating set problem. Since the decision problem of the minimum dominating set is NP-Complete, the hardness of the optimisation problem of CR becomes NP-Hard. □

Since the content replication problem is NP-Hard, it can not be solved in polynomial time. Nevertheless, the objective can be approximated efficiently using heuristics. In the remainder of this section, we present two helper selection algorithms that can be used for different types of environment and under different resource constraints. Table 6.4 summarises the symbols that are used in the following section.

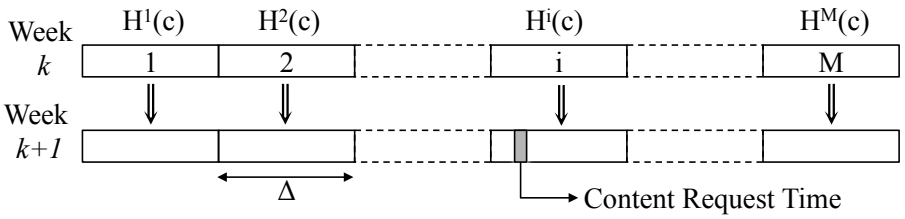


Figure 6.3: Periodic weekly helper selection

6.2.2 Greedy helper selection algorithm

Opportunistic encounters among devices are highly dependent on user mobility patterns, which essentially demonstrates social behaviour of users. Hence, there is a diurnal correlation of opportunistic encounters among users. These patterns have been extensively analysed in the areas of context-aware services and mobile social networking [7]. Usually, social behaviour of the majority of users have weekly routines. Further, there is higher probability that a user meets the same people at the same time every week. We take advantage of this predictive regularity of encounter patterns of users for the purpose of content replication. In order to allow instant content dissemination, helpers have to be selected in advance: i.e., helpers for the week $k+1$ have to be selected during the week k, as shown in Figure 6.3.

Consider the week (Δ_k) is divided into Δ time slots, where the Δ is the content delivery deadline. Since we do not know when a creator u_c is going to generate a content during the week $k+1$, we select several sets of helpers $H_{k+1}(c) = \left\{H_k^j(c) : u \in V \text{ and } j \in [1 : \Delta_k/\Delta]\right\}$ during week k. At the end of every week, the central management entity performs helper selection and informs all creators the respective sets of helpers. A creator will be assigned a new set of helpers only if the creator or previous helpers change its behavioural patterns.

First, we present a greedy algorithm GREEDY-HELPERS for content replication (CR). The most influential user in the network is the one who has contacts with (covers) the maximum number of consumers, i.e. has $\max|\sigma(\cdot)|$, which can be intuitively used as a greedy choice property.

Algorithm 4 presents the naive greedy algorithm to select $H(c)$ for the content creator c. D is the set of consumers covered by the creator and the selected helpers. After calculating $\sigma(u)$ for all $u \in V$, i.e. line 3, D will be equaled to the set of consumers covered by the creator $\sigma(c)$ (i.e. line 4) and the helper set $H(c)$ will be equaled to the creator c (i.e. line 5). Then, we loop through until we cover all devices or reach the threshold of replication $\lambda(c)$ while selecting the consumer with highest $|\sigma(u)|$ from the remaining consumers.

Algorithm 4 GREEDY-HELPERS$(G_t, \Delta, \lambda, c)$

1. $D \leftarrow H(c) \leftarrow \emptyset$
2. **for** all $u \in V$ **do**
3. Find $\sigma(u)$
4. $D \leftarrow \sigma(c)$
5. $H(c) \leftarrow c$
6. **while** $|H(c)| \leq \lambda(c)$ or $D \neq V$ **do**
7. Let $u \in (V \setminus (H(c) \cup D))$ maximising $|\sigma(u)|$
8. $H(c) \leftarrow u$
9. $D \leftarrow D \cup \sigma(u)$
10. **return** $H(c)$

This set-covering based solution has considerably high level of approximation factor. Kempe et al. [98] shows that this type of greedy algorithm is $(1 - 1/e)$ approximation, where e is the base of the natural algorithm. Even though, this provides an acceptable approximation algorithm for CR problem, finding $\sigma(u)$ for all $u \in V$ is computationally too complex in a dynamic network under resource constraints. To this end, we propose, in the next section, computationally simple dynamic centrality metrics for greedy choice that exploit the temporal and spatial regularity of social wireless connectivity patterns.

6.3 Community based Content Replication

6.3.1 Dynamic centrality metrics

As the first step, we aggregate every contact for a single graph without loosing any temporal information. Let an aggregated weighted graph $G = (V, E)$ consists of all edges in G_t, $\forall t \in (1, 2, \cdots, n)$ such that $G = G_1 \cup G_2 \cup \cdots \cup G_n$ and $\alpha^t_{u,v}$ be the edge weights at time t of each $e \in E_t$. $\alpha^t_{u,v}$ is the contact duration between the two users u and v at time t. For instance, if $\exists\ (u, v, \alpha^1_{u,v} = 20) \in E_1$ and $(u, v, \alpha^2_{u,v} = 30) \in E_2$, there are two edges in E connecting u and v with the contact duration of 20 and 30 seconds and happening at $t = 1$ and $t = 2$. Then, we focus on centrality metrics in G which provides better approximations for $\sigma(\cdot)$, i.e. expected number of covered consumers.

Hereafter, we propose two kinds of centrality metrics: local and global. Local metrics consider the information available locally (i.e. one-hop away) to decide the influence of the user. In addition to its simplicity and distributed calculation, the

Table 6.4: Summary of symbols for dynamic centrality metrics

Symbol	Description
D	set of consumers covered by c and the selected helpers $H(c)$
$\alpha_{u,v}^t$	contact duration between u, v at time t
G	aggregated weighted graph of $G_t \forall t$ with $\alpha_{u,v}^t$ as edge weights
E	total set of edges $\forall t$
$N(u)$	set of neighbours of u
$C_{LD}(u)$	local metric: $\|N(u)\|$
$I(u,v)$	initial contact time for u, v for a given Δ
$D(u,v)$	aggregated contact duration for u, v over Δ
$w(u,v)$	$I(u,v) + (1/D(u,v))$
$C_{LID}(u)$	local improved metric: $\|N(u)\| + \frac{\|N(u)\|}{\sum_{v \in N(u)} w_{u,v}}$
$p(u,v)$	binary parameter for the existence of path between u and v
$G'(V', E')$	directed aggregated contact graph
$C_{GP}(u)$	global metric: $\sum_{v \in V} p(u,v)$
$sp(u,v)$	shortest path between u and v in terms of $w(u,v)$
$C_{GIP}(u)$	global improved metric: $\sum_{v \in V'} p(u,v) + \frac{\sum_{v \in V'} p(u,v)}{\sum_{v \in V'} sp(u,v)}$
$com(u)$	set of users in u's community

privacy of the users can be well preserved: The users do not need to send any personal information, but only the aggregated value of the centrality metric, to the central entity. On the other hand, global metrics consider the whole network topology in order to decide the centrality value of the user, which is more complex and needs to be carried out in a central location.

Local metrics

One of the simplest centrality metric that implies the capability of neighbourhood coverage is the degree centrality $C_{LD}(u) = |N(u)|$ where $N(u)$ is the set of neighbours of u in the aggregated graph G. Degree centrality identifies popular nodes in the network and thus, has higher influence on content propagation.

Nevertheless, simple degree centrality does not guarantee that all counted encounters are useful for propagations of content due to the lack of consideration of temporal information in dynamic networks. Further, the contacts that happen early are important in propagation than those that happen later. Hence, a centrality metric which captures temporal information could be more realistic

to be considered in dynamic networks. To this end, we define the initial contact time as $I(u,v) = min\{t\} : \alpha^t_{u,v} > 0$ for all $t \leq \Delta$ and the total contact duration $D(u,v) = \sum_{t=1}^{\Delta} \alpha^t_{u,v}$ for an edge (u,v). We calculate the weight $w_{u,v} = I(u,v) + (1/D(u,v))$ for all $(u,v) \in E$. $w_{u,v}$ has the meaning of earliness and solidity of the contact (u,v). In practice, each mobile device can calculate w locally for all other devices it encounters for a given period. To this end, we define an improved dynamic degree centrality metric:

$$C_{LID}(u) = |N(u)| + \frac{|N(u)|}{\sum_{v \in N(u)} w_{u,v}}$$

$N(u)$ is the set of neighbours of u. C_{LID} describes how early and how independently the user makes other users into content propagators. We aim to use C_{LID} in the greedy choice for CR problem.

Global metrics

Even though centralised systems have disadvantages in terms of privacy and scalability, we make use of the global information to perform more accurate heuristics. Here, we define two path-based centrality metrics for node ranking. We first define a naive simple metric $C_{GP}(u) = \sum_{v \in V} p(u,v)$ for all $t \leq \Delta$, where $p(u,v) = 1$ if there is a path between u and v, and $p(u,v) = 0$ otherwise. This can be viewed as an extended degree for node u giving heuristics about the popularity and the availability of the node. Simplicity of the metric is the main advantage, which requires only information about existence of a path.

On the other hand, simplicity does not provide accurate heuristics. Therefore we define an improved dynamic centrality metric by considering temporal information such as the contact duration $D(u,v)$ and the initial contact time $I(u,v)$. We construct a directed aggregated graph $G' = (V', E')$ by directing all edges in G for both directions with same weights. Next, we prune all unrealisable edges in the network, i.e. if content is to be propagated via a node, its outgoing contact has to take place after at least its first incoming contact. At this point, we have an aggregated graph and at each node there is a guarantee that content will be propagated to other nodes if the content has arrived at the node. We calculate shortest-path $sp(u,v)$ for all node pairs $(u,v) \in V'$ in terms of edge weights $w_{u,v} = I(u,v) + (1/D(u,v))$. We define a path-based dynamic centrality metric C_{GIP}, similar to C_{LID}, such that it implies how early and how independently the user makes other content propagators as,

$$C_{GIP}(u) = \sum_{v \in V'} p(u,v) + \frac{\sum_{v \in V'} p(u,v)}{\sum_{v \in V'} sp(u,v)}$$

6.3.2 Community based greedy algorithm

In this section, we present our content replication algorithm which combines social sub-structural properties such as communities with previously defined dynamic centrality metrics. In order to distribute helpers within the network, we extract social sub-structures present in the contact graph.

Algorithm 5 COMMUNITY-GREEDY$(G, G', \Delta, \lambda, c)$

1. $D \leftarrow H(c) \leftarrow \emptyset$
2. **for** all $u \in V$ **do**
3. Find centrality metric $C_{LD}(u), C_{LID}(u), C_{GP}(u), C_{GIP}(u)$
4. communities $\leftarrow$ k-clique-algorithm$(G', 3)$
5. Let a community $com(u)$ be the u's community
6. $D \leftarrow com(c)$
7. $H(c) \leftarrow c$
8. **while** $|H(c)| \leq \lambda(c)$ or $D \neq V$ **do**
9. Let $u \in (V \setminus (D \cup H(c)))$ maximising $C_{LD}(u), C_{LID}(u), C_{GP}(u), C_{GIP}(u)$
10. $H(c) \leftarrow u$
11. $D \leftarrow D \cup com(u)$
12. **return** $H(c)$

For this we detect communities using *k-clique* community algorithm. Then, we distribute helpers among communities based on their ranking given by the proposed dynamic centrality metrics as in Algorithm 5. First, the consumer with highest centrality value is selected as a helper and rely on that helper to propagate the content with in the community. Then, the next highest consumer from a different community is selected, i.e. line 9 of the Algorithm 5. If the threshold of replication $\lambda(c)$ is lower than the number of communities, initial content propagators will not be selected from the creator's community, assuming that the creator is capable to propagate the content within its community. There can be a scenario where the majority of the consumers do not belong to communities. Then, the selection is purely based on the centrality value of consumers.

Furthermore, it is expected that the selected helpers will become quasi static over the time due to the regular behaviour of people [86]. Hence, the helper calculation will be carried out only for users who have changed their behavioural pattern. In practice, the number of users of the service increases incrementally and as a result

Table 6.5: Summary of mobility trace data sets

Data set	Duration (weeks)	Active devices (/week)	Contacts (/device/week)	Network type
Dartmouth	8	1138	12.456	WiFi
USC	8	1846	60.92	WiFi
SWIM	8	500	26.53	Bluetooth
D-SWIM	8	1500	29.85	Bluetooth
A-SWIM	8	1500	79.61	Bluetooth

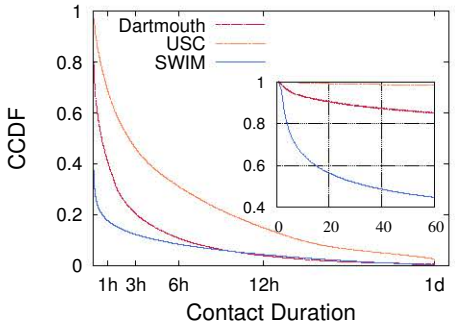
helpers will only be calculated for newly added users. To this end, we believe that the helper selection algorithm will scale up with the increasing number of users. In the next section, we evaluate the performance of this approach with respect to different centrality metrics.

6.4 Performance Evaluation of Community based Replication

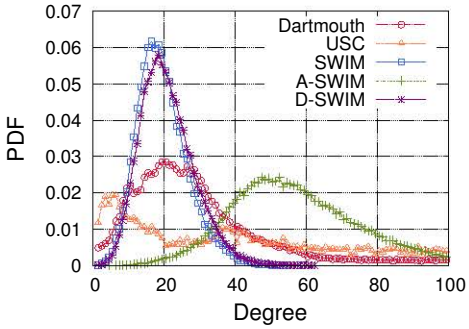
First, we present the dynamics of mobility traces and simulation setup that we use for performance evaluation. Then, we compare the performance of different centrality metrics for selecting influential consumers in terms of delivery success rate and latency. Finally, we compare the performance of content replication with *no replication* and *random replication* strategies and show that the proposed mechanism can incentivise users to use mobile social networks for UGC sharing.

6.4.1 Data sets

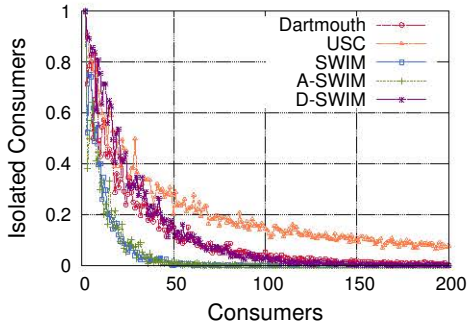
Two real-world data sets were used in the evaluation: Dartmouth data set [93] described in Section 6.2 and USC [99] data set, which also contains wireless connectivity patterns of users in a campus environment with 1846 users on average per week. Moreover, we use the synthetic traces that are generated using the SWIM simulator [100] by extending the Cambridge campus data set [101]. SWIM is the 500-nodes extended version of the original Cambridge data set of 36 Bluetooth enabled iMotes. Then, the trace data is further scaled up to generate 1500-nodes versions: 1) D-SWIM by keeping the density constant and 2) A-SWIM by keeping the area constant. We use these extended versions to understand the performance variations of the content replication in different environmental conditions. A summary of basic information of the data sets are presented in Table 6.5.



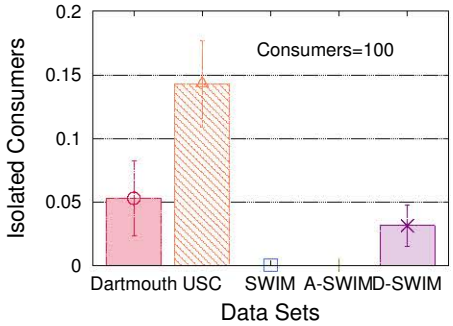
(a) **Complementary cumulative distribution function of contact duration**



(b) **Degree distribution**



(c) **Isolated consumers**



(d) **Isolated consumers in group of 100 consumers**

Figure 6.4: Dynamics of trace data sets

For both Dartmouth and USC, we consider that two users are in contact when they are connected to the same WiFi access point as described in [94]. For SWIM, when two users are within the Bluetooth communication range of each other, we consider those two devices are in contact. As per the complementary cumulative distribution function of contact duration in Figure 6.4a, more than 50% of users in USC data set have more than 3 hours of contact duration per day. This can be considered as a very high value which can be used to transfer any mount of data between two users, if the allowed delivery latency is one day. Even in Dartmouth more than 80% of users have more than 60 seconds contact duration per day. In contrast, SWIM has much lower aggregated contact duration. The degree distributions of all data sets are shown in Figure 6.4b. SWIM and A-SWIM have highly skewed degree distributions, i.e. majority of the users have similar number of contacts. In contrast, USC has a distributed degree while the degree distribution of the Dartmouth lies in between those two extremes. Further, we analysed the amount of isolated consumers when we randomly select a set of consumers. Figure 6.4c illustrates that USC trace

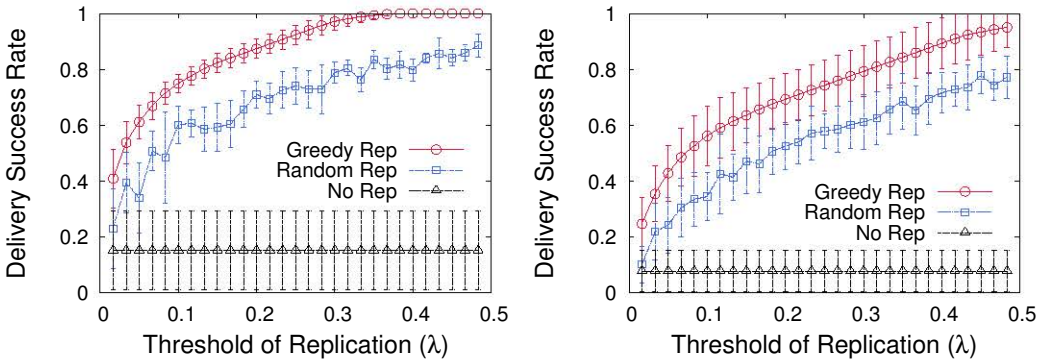
contains a large number of isolated users compared to others. For the case of 100 consumers (Figure 6.4d), nearly 15 of them are isolated in USC and only 5 of them are isolated in Dartmouth. In contrast, in all trials there are no isolated consumers in SWIM, it is less than 5 even in D-SWIM. To this end, the five trace data sets are not similar and cover various aspects that affect the performance of content replication. Hence, the performance evaluation resulted from these trace data sets would be applicable to wide variety of social environments.

6.4.2 Simulation setup

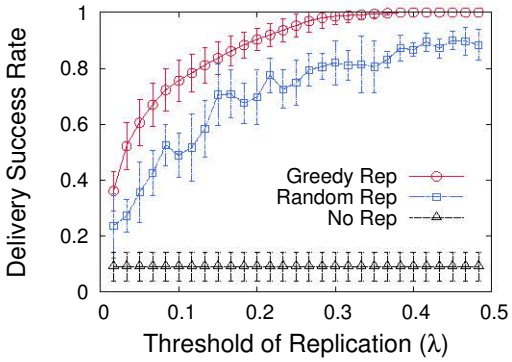
The simulator takes contact traces, content creation/access workloads, replication algorithm and communication protocol as inputs, and outputs the content delivery time for each user in the trace. We customise the discrete-event and discrete time simulator developed in Section 5.4.2 (Figure 5.11) for this particular content dissemination environment.

The trace data sets were divided into weeks as shown in Figure 6.3. We select the set of helpers according to proposed algorithms during week k (training week) and evaluate the performance in terms of delivery success rate and delivery latency during week $k+1$ (evaluation week). *Device Grouping* component in the simulator (Figure 5.11) performs the helper selection during the training week and then the coverage is evaluated by *Coverage Evaluator* component during the evaluation week.

The creator and consumers are selected randomly by *User Generator* and *Consumer Generator* modules of the simulator. In content propagation, we only use opportunistic contacts among the consumers. We consider that the number of consumers for a creator is 100 as it is the average size of group of friends in the popular social networking service Facebook [85]. We select consumers randomly from the users in the trace to evaluate the worst case scenario where the connectivity patterns of the users within the group have a minimum level of correlation. *Request Generator* assigns a content of size 8.4MB for each creator, which is the median content size in YouTube [92]. Then, *Coverage Evaluator* perform the evaluation of opportunistic content delivery performance. The transfer rate among consumers are considered as uniform and 2Mbps [10]. Hence, a consumer has to have aggregated contact duration $\alpha(u)$ of 33.6 seconds with the creator or any of the helpers or the propagators to completely download the content. The content delivery deadline Δ is considered as 3 days considering 3 days could be the largest tolerable delay for content sharing in social networking. If the content is delivered to a consumer before the delivery deadline, it is considered as a successful delivery. Thereby the **Delivery Success**



(a) Dartmouth Data Set **(b) USC Data Set**



(c) SWIM-500 Data Set

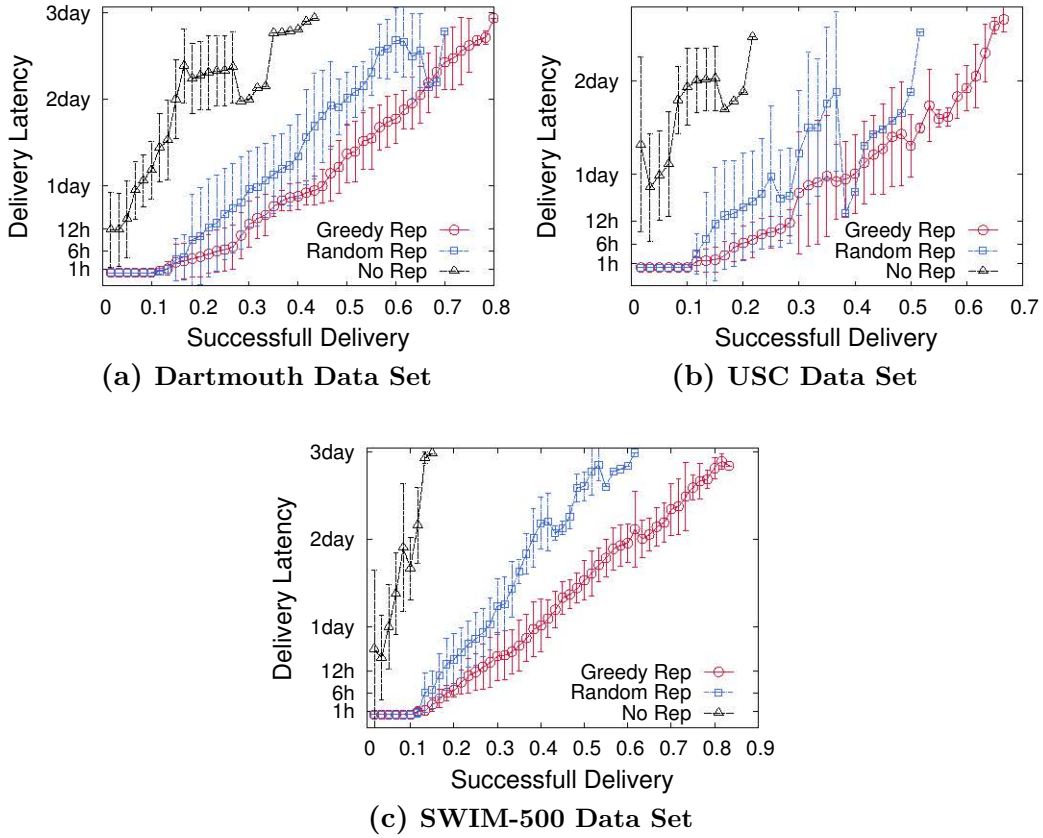
Figure 6.5: Delivery success rate against the threshold of replication (λ)

Rate is calculated as the ratio between number of successful deliveries and total number of consumers. The time lag between the content sharing and the content receiving time is considered as the **Delivery Latency** for a particular device. All simulations are carried out varying the monitoring and evaluation periods through out the duration of the data sets.

We evaluate and compare the content delivery success rate and delivery latency of the proposed system against the cases where **1) No replication** is performed and **2) Random replication**, which is the simplest way of selecting helpers without any knowledge about the contact patterns among consumers. Moreover, the increment in the percentage of one-hop opportunistic transmission is also analysed, which is proportional to energy and communication cost savings.

6.4.3 Evaluation of delivery performance

For all data sets in Figure 6.5, it is evident that there is a significant gain in content delivery success rate and delivery latency for the GREEDY-HELPER selection (Algorithm 1) compared to no replication approach. In fact, the greedy replication is



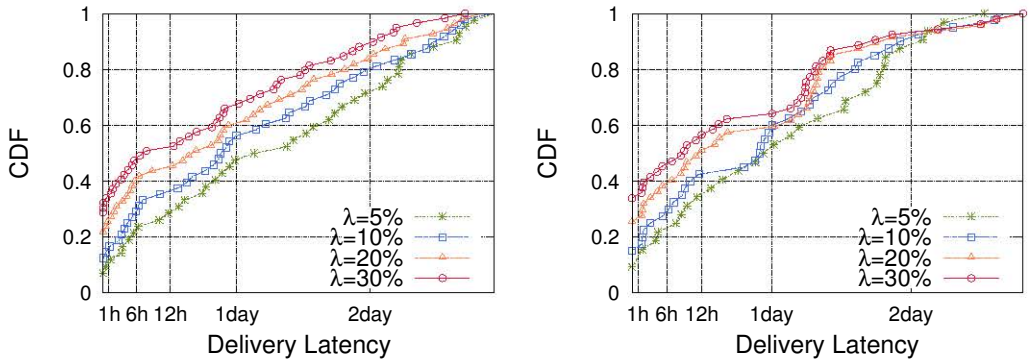
(a) Dartmouth Data Set **(b) USC Data Set**

(c) SWIM-500 Data Set

Figure 6.6: Delivery latency for specific delivery success rate when $\lambda = 0.1$

always better than the random replication. Note that this greedy replication is the worst case scenario due to random selection of consumers. In real environments, consumers of the same content would have highly correlated connectivity patterns and the performance can be expected to be improve. In Dartmouth (Figure 6.5a), the delivery success rate is approximately 80% for 10% of content replication ($\lambda = 0.1$), while USC has a comparatively lower success rate of approximately 60%. After a certain level of replication (i.e. approximately $\lambda = 0.1$), the delivery rate shows linear increment, i.e. further replication will not deliver content to any other consumer via opportunistic communication because there will be only isolated consumers to be selected for content replication. This threshold of replication can be considered as an effective upper bound for λ.

Even though we are dealing with delay tolerant content dissemination applications, delivery latency is still one of the prime factor to consider in mobile social networking. Figure 6.6 shows the time taken by greedy replication, no replication, and random replication approaches against the percentage of successful delivery. In Dartmouth (Figure 6.6a), content replication delivers content to 40% of the consumers in less than 1 day, whereas the latency is almost 3 days if there is no initial



(a) Dartmouth Data Set

(b) USC Data Set

(c) SWIM-500 Data Set

Figure 6.7: The variation of CDF of delivery latency with λ values

content replication. All three data sets show similar behaviour in terms of delivery latency. For instance, the time taken to cover 40% and 60% of the consumers is approximately 1 day and 2 days respectively, in all three cases. In random replication, sometimes the delivery latency is lower than the greedy replication. However, in general, the delivery latency for greedy replication is lower than the random replication.

Even though the proposed algorithms perform better than random helper selection, the results of random selection can not be negligible as it does have comparable results in some cases without any prior knowledge of contact patterns. This behaviour has been observed in the literature as well [6], [9]. For some group of friends, one may not need an intelligent helper selection strategy. Mainly because the connectivity graph is either too dense or too sparse.

The cumulative distribution function of the delivery latency for successful deliveries are shown in Figure 6.7. In all data sets, the probability of the delivery latency being less than 1 day is approximately 60% for $\lambda = 10\%$. In [82], it has been observed that 55% of Flicker content is uploaded after a lag of more than 1 day. Thus,

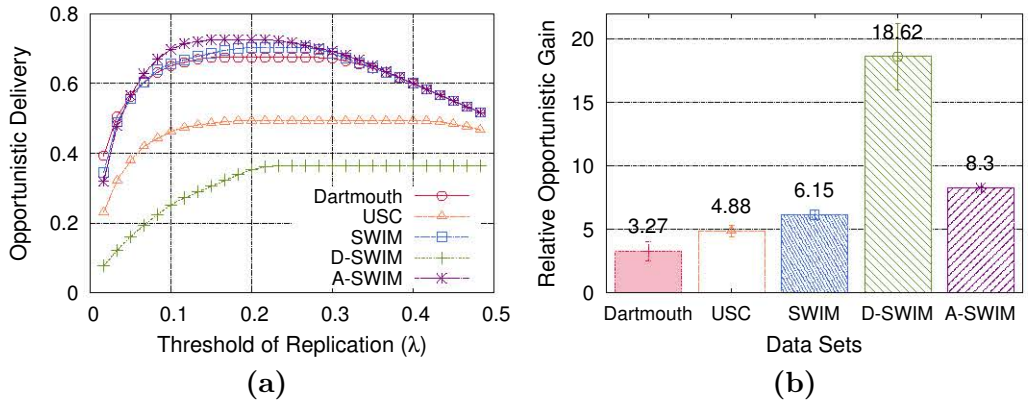


Figure 6.8: Analysis of amount of opportunistic content delivery. $\lambda = 10\%$

we believe that the delivery latency resulted from the opportunistic approach is practical in such content dissemination applications. For applications/services that require a lower delay, it is possible to increase the threshold of content replication as shown in Figure 6.7. The common pattern is that the delivery latency reduces with increasing λ values. In Dartmouth and USC, there is a 20% improvement in the delivery latency being less than 1 day when we increase λ from 5% to 30% while it is a 30% improvement in SWIM. However, the difference between two consecutive λ values becomes smaller when λ increases.

6.4.4 Opportunistic delivery gain

The primary objective of the content replication is to increase the opportunistic content delivery. If the content is not received by the delivery deadline through low-cost networks, we consider that those consumers will download the content through cellular network. Based on that, the portion of *Opportunistic Delivery* is calculated out of all content deliveries. Figure 6.8a shows that the portion of opportunistic deliveries when we use greedy replication selection algorithm. The amount of opportunistic deliveries increases with λ only for low λ values. In Dartmouth, it is possible to deliver content for approximately 70% of the consumers via opportunistic communication after 10% of replication which is below 40% when there is no replication. We extended our simulations to the two extended versions of the SWIM data set to understand the behaviour of opportunistic delivery percentage. The results are closely related to the degree distribution of the data sets as shown in Figure 6.4b. D-SWIM has the lowest performance because it was extended by increasing the area and number of users while keeping the density of the network constant and equal to SWIM. This makes the network more spread and increases the number

of appearing communities and consequently requires a high level of replication to cover the same number of consumers as in SWIM. In contrast, when we increase the density as in A-SWIM, it improves the opportunistic delivery percentage. Similarly, in USC, there is a large number of isolated users as shown in Figure 6.4d, which decreases the overall density. Hence, the density of the contact graph has a considerable impact on the opportunistic delivery performance.

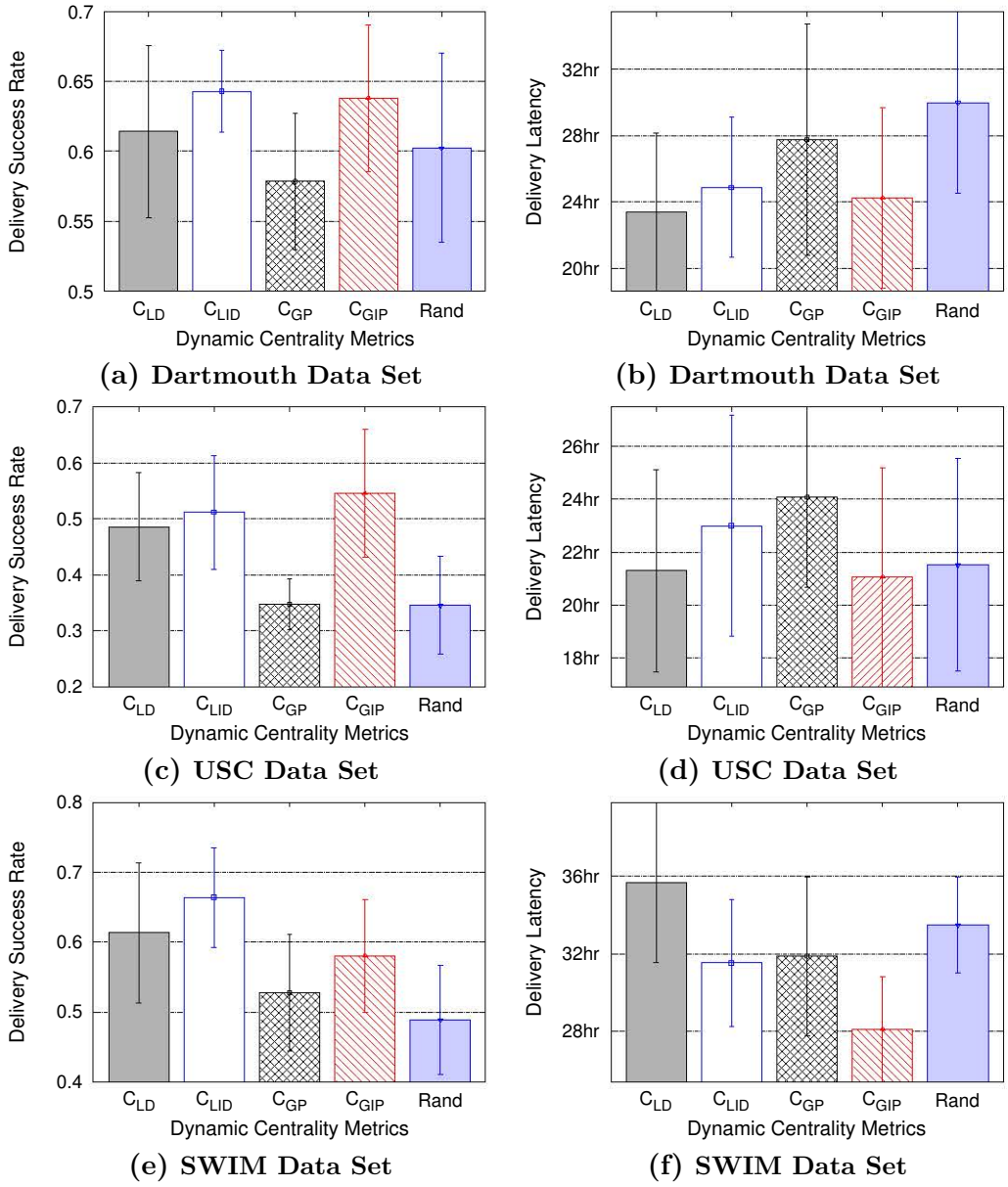
Figure 6.8b summarises the *Relative Opportunistic Gain* as the portion of opportunistic delivery with content replication and with no replication approaches for all data sets. Even though D-SWIM has the lowest percentage of opportunistic delivery, it has the highest relative gain of 18.62 times because in D-SWIM, the percentage of opportunistic delivery when there is no replication is as low as 1.3%. Dartmouth data set shows the lowest gain of 3.27 times, since it consists of well connected users compared to other environments. Thus, the results show that it is possible to significantly increase the one-hop opportunistic delivery with a low number of initial content replication, i.e. approximately less than 10% of the consumers. In addition, improvement in opportunistic transmission is proportional to the energy and communication cost and thereby incentivise mobile users to take part in distributed mobile social networks for content dissemination.

6.4.5 Evaluation of dynamic centrality metrics

In this section, we compare the influence of different dynamic centrality metrics in content replication. Further, we compare the results with *Random* selection of helpers, which is the simplest way of selecting helpers without any knowledge about the contact patterns among consumers.

Figure 6.9 shows the content delivery success rate and the mean content delivery latency when the helpers are selected based on different centrality metrics according to the Algorithm 5. When we compare the two local centrality metrics, C_{LID} has a slightly better delivery success rate with lower standard deviation than C_{LD} in all three data sets. Similarly, the improved global centrality metric C_{GIP} has better performance in terms of both delivery rate and latency compared to the naive C_{GP}. This is due to the fact that each improved metrics, C_{LID} and C_{GIP}, consider the time dependency in connectivity patterns, which improves the content propagation. On the other hand, although the random selection has a bit worse performance than the improved metrics, random selection is not negligible due to the heavy reduction in resource requirements.

However, in some cases there is no significant difference between the performance



(a) Dartmouth Data Set **(b) Dartmouth Data Set**

(c) USC Data Set **(d) USC Data Set**

(e) SWIM Data Set **(f) SWIM Data Set**

Figure 6.9: Comparison of different dynamic centrality metrics. $\lambda = 10\%$

of local and global metrics. All these general similarities are related to dynamics of the contact patterns among consumers in these environments. For instance, when the degree of the majority of the consumers are similar, i.e. skewed degree distribution as shown in Figure 6.4b for SWIM, the performance of a simple degree based local centrality metric becomes significant as depicted in Figure 6.9e for the same data set. In contrast, when the degree distribution is not skewed, the intelligent path based selection will perform better, similar to C_{GIP} performance for Dartmouth and USC. In particular, USC has the largest improvement of approximately 20% in coverage for C_{GIP} compared to C_{GP} because USC has the most distributed

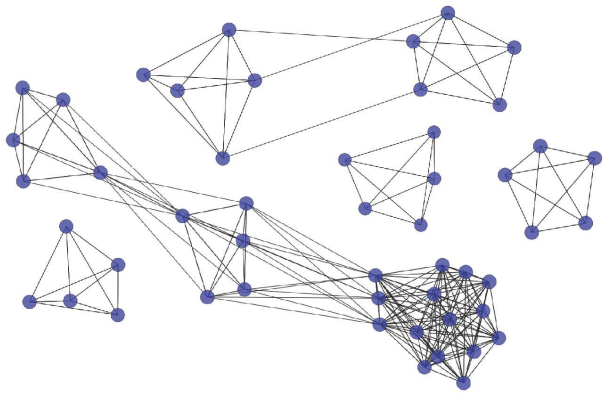


Figure 6.10: HCMM generated aggregated contact graph of 50 users for one hour

degree distribution. On the other hand, due to the large number of isolated consumers (Figure 6.4c), USC does not have much gain in delivery latency. In SWIM, C_{GIP} has much lower delivery latency because it has the lowest number of isolated users. Hence, the selection of the appropriate centrality metric to identify the most influential users in content dissemination is highly environment dependent and the appropriate centrality metric can be identified by analysing the behaviour of the users of the particular community.

As can be seen in Figure 6.9 in some cases, the random selection even without any knowledge about contact patterns would perform better or similar to intelligent selection. Han et al. [6] also observed similar behaviour in their target set selection for content dissemination, where the random selection performs similar to the greedy solution. Therefore, we further investigate on random selection of helpers to propagate the content within a given community.

6.5 Component based Random Replication

As observed in the previous section, the effectiveness of opportunistic content dissemination is highly dependent on the characteristics of the contact patterns among users. Figure 6.10 shows a snapshot of a HCMM generated aggregated contact graph of 50 users after one hour. It can be seen that the network consists of several physically disconnected components due to the social behaviour of mobile users. These subgraphs in a MSN are composed by the users with specific similar interests or real life social interactions. E.g. the shoppers in a specific food store in a MSN created by all MSN users in the shopping mall, a group of friends sitting together in a sport event, etc. can be considered as instances of creating these subgraphs in MSNs. This temporal community structure has been observed in the literature in

other MSN data sets as well [35].

However, the users in MSNs have very little knowledge about other users in the network. Majority of MSNs are formed among a set of random users who happen to be at the same location at the same time. Therefore, in most cases, there are no history of connectivity patterns to exploit regular behavioural patterns to infer future contacts as proposed in the previous section. Furthermore, the centralised replication management is limited by the requirement of having an Internet connection. Thus we propose two simple distributed replication strategies.

1) Random Replication: is the simplest strategy and does not require any knowledge of the contact patterns. Replicators are randomly selected among the users in the MSN. We consider that the maximum replication is constrained by λ - *Threshold of Replication*;

$$\lambda = \frac{\text{number of replicators}}{\text{number of all MSN users}}$$

i.e. the fraction of users used for content replication (replicators) out of all MSN users. The number of all MSN users can be obtained through the MSN service.

2) Component Replication: assumes that the creator has some knowledge of the contact graph $G_{t=0} = (V, E_{t=0})$ at the time of content creation. This can be obtained through a protocol for sharing contact patterns among the users or through the MSN service. We consider that two users are connected if they are in the direct wireless communication range of each other based on the geographical location of the users. The service provider can maintain a contact graph for each MSN and update members of the MSN periodically. Let D be the set of consumers covered by the creator u_c and the replicators $R(c)$. The creator identifies disconnected components in the contact graph as in Figure 6.10. Then the creator selects one replicator from each component, as shown in Algorithm 6. To increase the coverage with minimum replication, the creator greedily selects replicators from largest components until it reaches the threshold of replication as in step 5 and 6.

However, all contact patterns may not be available to all creators. Therefore, we evaluate the performance of component replication by varying the initial knowledge at the time of replication selection. The connected components in a graph can be computed in linear time with respect to number of vertices and edges. When the contact graph is too complex due to large number of users and contacts, the use of the random replication strategy is more indicated. In fact, the random replicators and the creator itself would provide enough coverage due to high density of the contact graph as shown in Figure 6.2c. Finally, in Section 6.6, both replication

strategies are compared against the optimal replication obtained when assuming the perfect future knowledge of contact patterns.

Algorithm 6 COMPONENT-GREEDY($G_{t=0}, \lambda, c$)

1. $D \leftarrow R(c) \leftarrow \emptyset$
2. components $\leftarrow$ components(G)
3. Let a component $com(u)$ be the u's component
4. $D \leftarrow com(c)$
5. **while** $|R(c)| \leq \lambda|V|$ or $D \neq V$ **do**
6. Let $u \in (V \setminus (D \cup R(c)))$ maximising $|com(u)|$ randomly
7. $R(c) \leftarrow u$
8. $D \leftarrow D \cup com(u)$
9. **return** $R(c)$

6.6 Performance Evaluation of Random Replication

In this section, we first evaluate the performance of the proposed content replication strategies in terms of content delivery time and coverage. Then, the effects of other influential factors such as content pre-fetching and collaboration among users are analysed and discussed. Finally, we quantify the networking infrastructure bandwidth savings that can be achieved with the proposed hybrid content dissemination strategies.

6.6.1 Simulation Setup

Similar to the previous evaluation in Section 6.4, we customise the simulator developed in Section 5.4.2 (Figure 5.11) for this particular content dissemination environment.

Evaluation metrics and benchmarks

A randomly selected creator initiates content dissemination by replicating the content using infrastructure network while distributing the content to the devices in the communication range. *Coverage Evaluator* (Figure 5.11) component measures the **coverage** as the portion of successful deliveries from the total consumers. The content **delivery time** is the time lag until a user receives the last piece of the

content from the content creation time. These metrics are calculated for each user in the MSN and then the sample mean and standard deviation are presented. We evaluate the trade-off between replication and performance by varying the **threshold of replication**, i.e. the maximum replication per content. If the content does not arrive via opportunistic communication at the end of the delivery deadline, we assume that the user accesses the content using infrastructure.

We compare the coverage and content delivery time of the two replication strategies described in Section 6.5. *Device Grouping* component in the simulator (Figure 5.11) is customised to select replicators for each user such that;

- **Random Replication:** replicators are selected randomly from the consumers of the content.
- **Component Replication:** replicators are selected according to the Algorithm 6. First, we consider perfect knowledge of the network at the time of content creation and replication selection. Then the knowledge of the network is varied later in this section to study the impact of low collaboration of users on the delivery performance.

Then, we benchmark these strategies with;

- **Oracle Replication:** replicators are selected to maximise coverage assuming future knowledge of contact patterns of all users. Even if we have the future knowledge, optimal replication selection is not possible, since it is a NP-Hard problem as shown in [102]. Therefore we use optimal greedy selection for this evaluation by selecting users who can cover maximum remaining users as proposed in [102]. This can be considered as an upper bound for performance metrics.
- **No Replication:** opportunistic dissemination initiates only from the creator, which gives the lower bound for performance metrics.

Data sets and content creation/access models

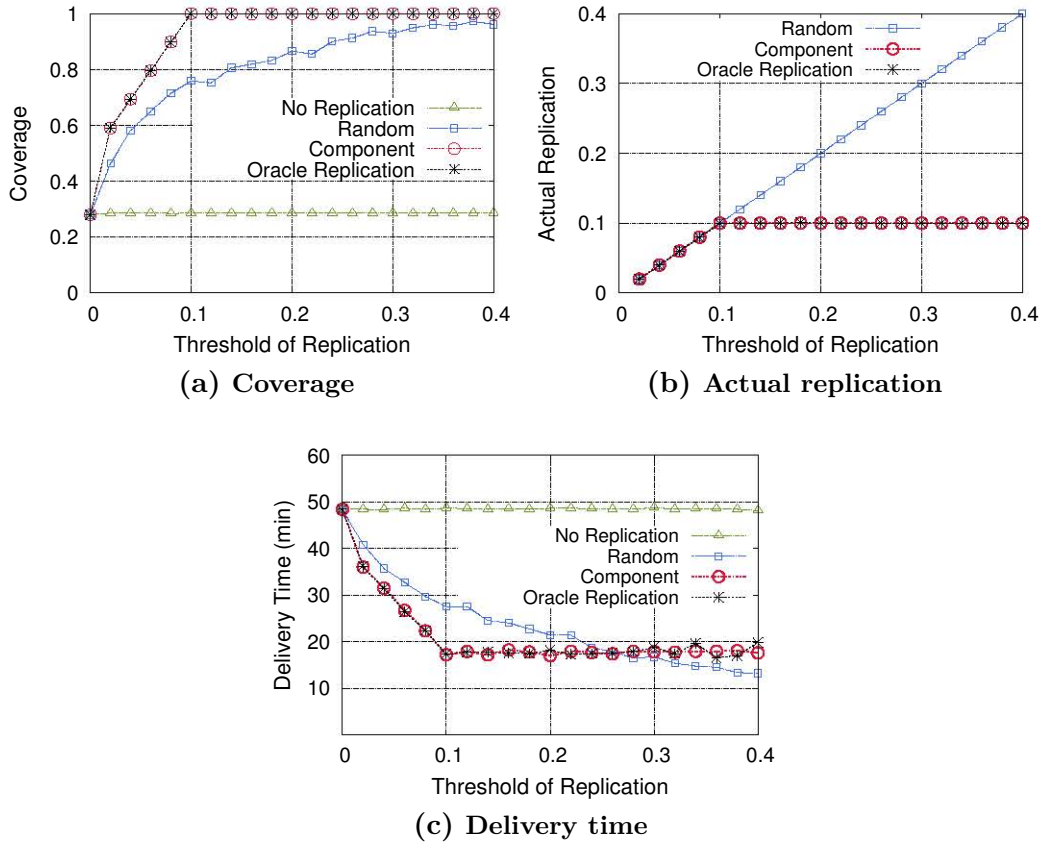
Sigcomm [95], Dartmouth [93] and HCMM [96] data sets described in Section 6.1.2 are again used for this evaluation. The content creation/access workload and the data communication are modelled according to Table 6.6. The number of users is limited to 50. **Content access delay** (CAD) is the time lag between the content creation by a creator and content access by a consumer, i.e. the time when the

Table 6.6: Content creation/access workload and content dissemination model for emulating MSNs

Content creation and access model	
No. of users	50
Content access delay	Gamma (mean=20min)
Size of a content	8MB
Delivery deadline (Δ)	60 minutes
Content dissemination model	
Transfer rate	2 Mbps
Dissemination protocol	Epidemic P2P
Piece size	265KB

consumer starts downloading the content. We model CAD as Gamma distributed among the 49 consumers and then vary the mean value to evaluate the impact of CAD. It is worth noting that even when the mean access delay is 60min, there are users who access the content immediately after the content creation. The size of the generated content is considered as 8MB which is the mean content size on popular UGC sharing service YouTube [92]. Since local WiFi is the most common network type for MSNs, we consider a practical data transfer rate of 2Mbps as observed in [10]. For this evaluation, we consider that one piece of content can be shared in one second contact duration, i.e. the piece size is 256KB. We emulate Epidemic P2P(peer-to-peer) content dissemination model with these parameters. The users exchange pieces of content that they do not have when they are in the communication range of each other. In practice, this can be achieved by using any peer-to-peer data transferring protocol such as BitTorrent. We have demonstrated the feasibility of such an implementation in Android smartphones in [103]. The users may not be able to download all pieces in a single contact duration with one user. Therefore, when a user completes an aggregated contact duration of 32 seconds with others who have the required pieces of the content, it is considered as a successfully downloaded assuming a zero bit error rate.

We assume that all users are collaborative and propagate the content to nearby devices. Moreover, we consider that the networking infrastructure is reliable and always available to all users. Since the main purpose of the evaluation is to measure the performance gains in the opportunistic delivery after the content replication, all evaluations are carried out considering that the content replication through the infrastructure has been already finished.



(a) Coverage

(b) Actual replication

(c) Delivery time

Figure 6.11: Performance of replication strategies vs threshold of replication for HCMM generate contact patterns

6.6.2 Coverage, delivery delay and replication trade-off

First, we study the behaviour of the performance metrics, coverage and delivery time, in a controlled simulation environment for HCMM generated contact patterns. Then, we evaluate them on real-world contact traces of Dartmouth and Sigcomm.

As expected, opportunistic coverage for all replication strategies increases with the number of replicas as shown in Figure 6.11a. For this evaluation, we assume complete knowledge of the network at the time of content creation to perform component-based replication. Component replication achieves 100% coverage when we allow replicas of 10% of users (5 out of 50 users in this case) and performs as good as oracle replication. Note that even the random replication has a relatively high performance compared to no replication. In fact 75% of coverage is achieved with 10% of replication. Compared to no replication even random replication achieves approximately 50% of more coverage with just 10% of replication.

Actual replication is the fraction of replicators out of all MSN users, which is limited by the algorithm parameter - the threshold of replication (λ). The actual repli-

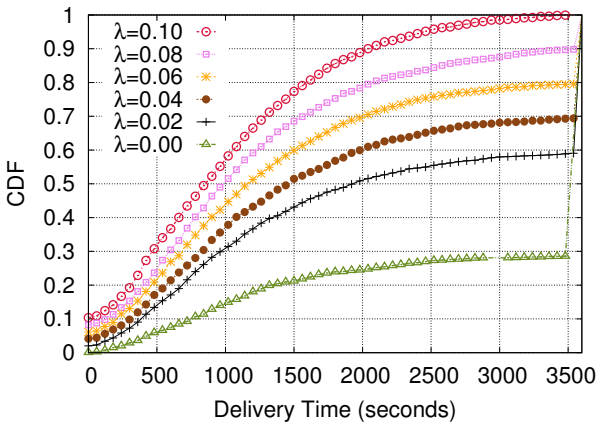
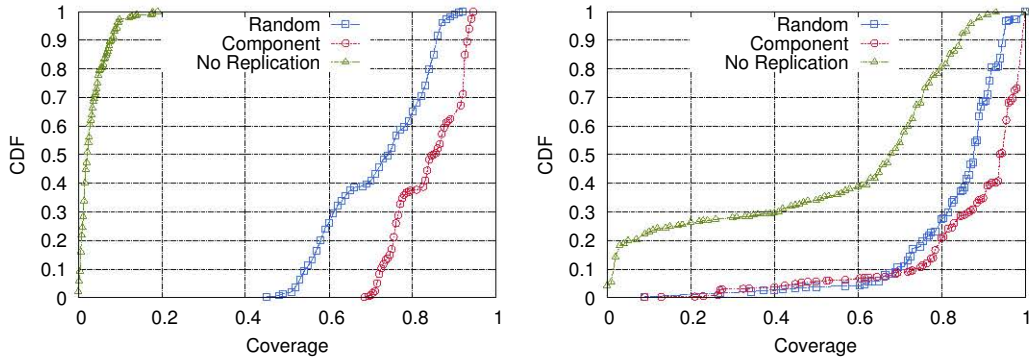


Figure 6.12: Cumulative distribution function (CDF) for content delivery time of individual users for HCMM.

cation in component-based strategy does not increase beyond 10% (Figure 6.11b) similar to oracle replication, even though λ is increased beyond 10%. Whereas the actual replication in random selection linearly increases with λ. The reason behind this is that there are only 5 disconnected components in this particular MSN, which limits the component replication to 10% of users. This network characteristic based limit of replication is quite useful when there is a large number of users in the MSN.

Figure 6.11c shows the decrease in content delivery time with respect to the number of replicas. Similar to coverage, delivery time for component replication converges to around 20min (mean access delay) after 10% of replication. Delivery time for random replication continuously goes down as the replication linearly grows with λ because replicators always pre-fetch the content using infrastructure. The cumulative distributions of the delivery time for λ up to 10% are shown in Figure 6.12. As can be observed in Figure 6.12, even a single replica significantly increases the probability of receiving the content with lower delivery delay. For instance, the probability of delivery time less than 2000 seconds increases by 0.25 with just one replica ($\lambda = 0.02$) and reaches 0.9 when replication increases to 5 ($\lambda = 0.1$).

In Figures 6.13a and 6.13b, we compare the coverage gain from content replication compared to the no replication approach in real-world environments. In Sigcomm (Figure 6.13b), there is a considerable coverage gain compared to no replication irrespective of the replication strategy. In fact, the probability of the coverage being higher than 80% is 0.8 for component replication and 0.1 for no replication. Due to the high connectivity in the conference scenario, the difference between the random selection and component-based selection is not very large in Sigcomm. On the other hand, due to the low density of the connectivity network in Dartmouth as



(a) Dartmouth **(b) Sigcomm**

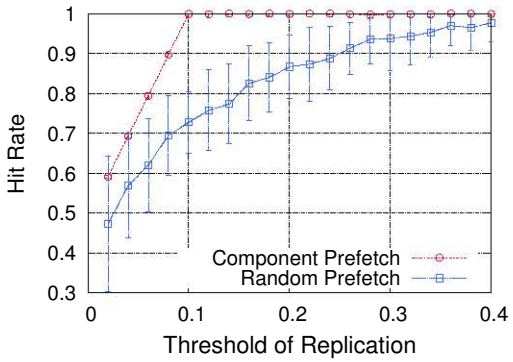
Figure 6.13: Cumulative distribution function (CDF) of coverage, $\Delta = 60min,\ \lambda = 0.1$.

discussed in Section 6.1, the component-based replication improves the coverage a lot in Dartmouth (Figure 6.13a). For instance, the coverage is higher than 50% and 70% of consumers approximately with random and component replication respectively, for all creators in the MSN. Whereas the maximum coverage for no replication is only 20%. Thus, the content replication has shown promise in increasing content dissemination in both simulated and real-world MSN environments. It is worth noting that the surprisingly high coverage of random replication has been observed in some other studies in the literature as well [6], [9], [104]. In some environments like Sigcomm, we may not need an intelligent replication strategy because majority of the users are anyway connected.

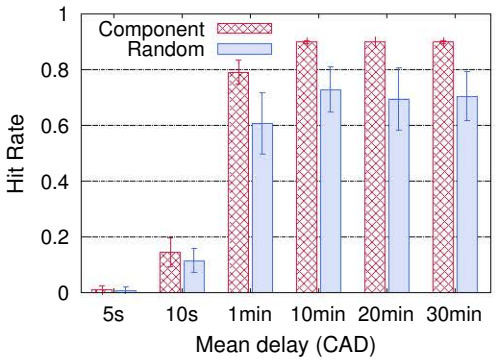
In summary, infrastructure supported content replication significantly increases the opportunistic coverage and decreases the content delivery time with small number of replicas, approximately 10% of the total consumers, irrespective of the replication strategy.

6.6.3 Effects of content pre-fetching

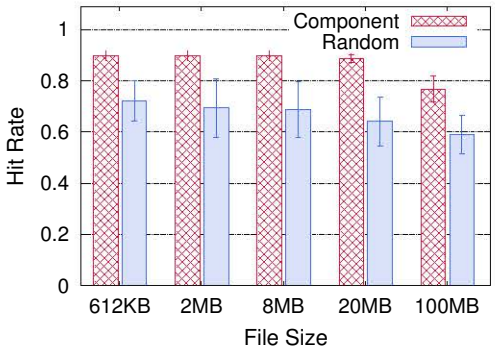
The main advantage of content pre-fetching is the improved user satisfaction where there is no startup delay. Furthermore, it saves cellular bandwidth usage, if the content is requested when the user is connected to cellular network as the pre-fetching is performed through other networks. MSN application installed in the mobile device monitors content sharing feeds in the background and starts searching for the pieces of the shared content in nearby devices immediately after the shared notification. If all the pieces of the content is fetched before the content access delay, we consider it as a **cache hit**.



(a) hit rate vs λ, mean CAD=20min,File size=8MB.



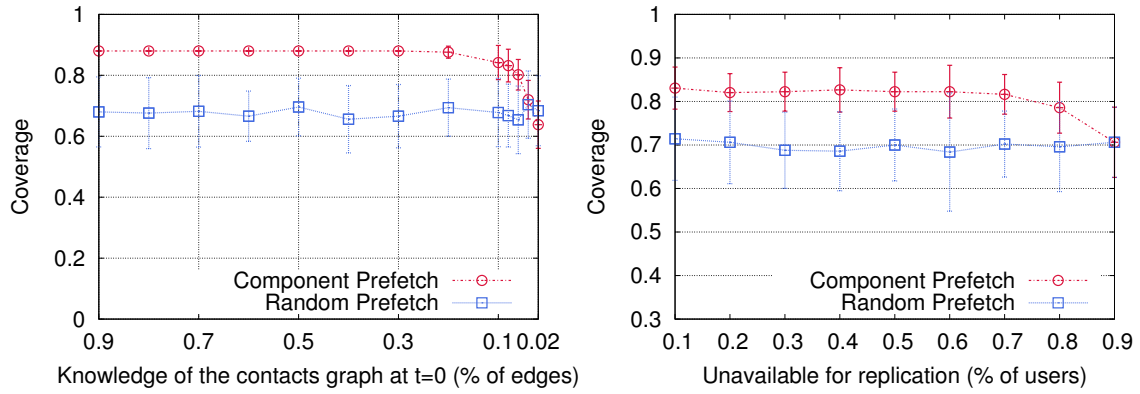
(b) hit rate vs mean CAD, $\lambda = 0.08$, File size=8MB.



(c) hit rate vs file size, $\lambda = 0.08$, mean CAD=20min.

Figure 6.14: Performance of cache hit rate for HCMM (mean and standard deviation).

Figure 6.14a shows that the cache hit rate increases with the number of replicas for HCMM. Even with random replication, it is possible to achieve 70% cache hit rate for just 10% of replication when the mean content access delay is 20 minutes (delay is Gamma distributed). However, the standard deviation is high for random replication when there is only a small number of replicas, and it decreases gradually as the number of replicas increase. Furthermore, we evaluate cache hit rate by varying content access delay and the size of the created content. Cache hit rate is very low for very short access delays (less than 1 minute) as shown in Figure 6.14b. However, it increases rapidly to about 80-90% when the mean access delay is larger than one minute. In practice, it is not unreasonable to assume one minute content access delay in MSN environments. Thus access delay does not significantly affect the opportunistic delivery performance in these particular environments. Similarly, cache hit rate does not drop significantly with the size of the created content even if it is as large as 100MB as shown in Figure 6.14c. This is a result of very high aggregated



(a) Coverage vs knowledge of the connectivity network at the time of content creation.

(b) Coverage vs availability of nodes to replicate others content.

Figure 6.15: Coverage (mean and standard deviation) vs knowledge of the network and availability of nodes to replicate others content for HCMM, $\lambda = 0.08$.

contact duration with other users in these environments. In general, the standard deviation is higher in random replication because of the intrinsic nature of that selection, whereas intelligent component replication always has a small deviation. To this end, we argue that content pre-fetching provides a fairly high cache hit rate under all circumstances with infrastructure supported content replication. This can be used to incentivise users to take part in MSN service and moreover to become replicators, because all replicators pre-fetch content.

6.6.4 Effects of collaboration of users

When selecting component-based replicators (Algorithm 6), we considered that the creator has complete knowledge of the connectivity network at the time of content creation. However, it is not reasonable to assume that all creators have this complete knowledge because of a number of practical issues such as some users may not agree to disclose their contact patterns to other users or to the service provider due to privacy and security concerns. If users do not get in contact with many users they may have partial knowledge of the network until they receive an update of the contact graph from the MSN service. To understand the effect of low user collaboration, we vary the knowledge of the connectivity network at the time of content creation by randomly removing known edges from the connectivity network of HCMM generated contact patterns. In Figure 6.15a, it can be seen that random selection is not affected by the knowledge of the network and thereby can be con-

sidered as an advantage over other strategies. It is interesting to see that even if we reduce the known edges to 10% from all existing edges, there is no significant drop in coverage for component-based replication. The coverage decreases and becomes similar to random replication only when there is 10% or low knowledge of the connectivity network at the time of content creation. The main reason for this is the small-world phenomena in MSN environments, which is robust to random perturbations. Therefore, even if we remove majority of the social links, the basic structure of the network is preserved and thus the components to select replications. Thus, if the content creator or the contact pattern sharing protocol has a little knowledge (at least 10% of the edges) of the network, it can achieve high coverage through the proposed component-based greedy replicator selection.

As described in Section 6.5, users have the option to become replicators based on the device or network context such as spare storage, battery and available infrastructure network type. Also, it can be purely based on social context such as privacy, security and personal preference. We evaluate the coverage along with the availability of devices to replicate others' content in Figure 6.15b. We assume all users are willing to opportunistically propagate content to others in the communication range, even if they refuse to become an initial replicator. Similar to partial knowledge in Figure 6.15a, random replication is not affected by reduction in potential replicator set. Moreover, coverage from the component-based replication does not drop significantly until 80% of the users are unavailable for replication. Usually, MSNs are formed among users who are willing to collaborate and thus it is not unreasonable to assume that at least 20% of them agree to replicate content. Thus, the proposed content replication strategies are robust enough to maintain high coverage even under very low user collaboration within the particular MSN.

6.6.5 Infrastructure bandwidth usage

Infrastructure bandwidth typically poses higher cost to the user and minimising its usage is one of the key goals of the proposed system. If the content is not received at the end of the delivery deadline through opportunistic dissemination, we assume that these consumers fetch the content via the existing networking infrastructure (WiFi or Cellular). In Figure 6.16, infrastructure bandwidth consumption is compared with no replication strategies. It is natural to assume that infrastructure supported content replication consumes more infrastructure bandwidth. However the coverage increment achieved through replication reduces the infrastructure usage of all consumers at the end of the delivery deadline. Figure 6.16 depicts the normalised

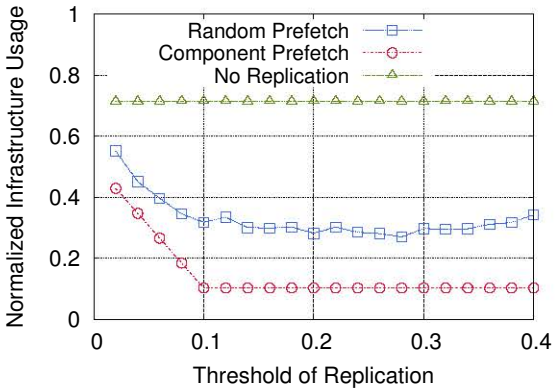
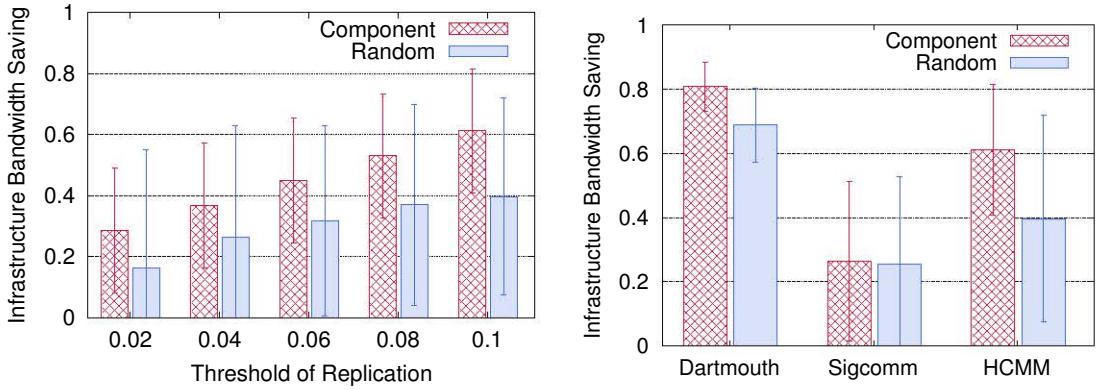


Figure 6.16: Infrastructure bandwidth usage of all users vs threshold of replication for HCMM generated contact patterns.



(a) Infrastructure bandwidth saving of HCMM.

(b) Infrastructure bandwidth saving for different data sets.

Figure 6.17: Infrastructure bandwidth saving (mean and standard deviation) for different data sets compared to no replication.

infrastructure usage where it equals one, when all consumers access the content using infrastructure. The infrastructure usage of no replication is not equal to one because consumers around the creator receive the content through opportunistic propagation. Although the number of replicas for random replication increases linearly with threshold of replication, the infrastructure usage converges to around 0.3. Since component-based replication achieves high coverage after 10% of replication, the infrastructure is only used to replicate content which reflects 0.1 normalised infrastructure usage.

In Figure 6.17a, we show the infrastructure bandwidth saving of content replication compared to no replication. When there is only one replica ($\lambda = 0.02$), sometimes random replication does not perform effectively, whereas component-based replication saves around 30% of infrastructure bandwidth. In addition, the stan-

dard deviation in random replication is always higher than component replication because sometimes random replication selects isolated users for content replication. However, if we increase the threshold of replication, the standard deviation would be reduced as observed in Figure 6.14a. When the replication is about 10% of total consumers ($\lambda = 0.1$), component replication saves approximately 60% of infrastructure usage.

Then we compare the results with two different real-world MSN environments in Figure 6.17b. Dartmouth is a sparsely connected environment of 1146 users in a university and Sigcomm is a densely connected network of 76 users in a conference. Due to the high coverage gain from content replication in Dartmouth (Figure 6.13a), the saving in infrastructure usage is considerably high, i.e. 80% saving for component replication. Moreover, the standard deviation in Dartmouth is smaller reflecting the high confidence of achieving high coverage by content replication. Since the users in Sigcomm trace are well connected, its infrastructure saving compared to no replication is not that high (approximately 20%) and for some users there is no infrastructure saving for random replication due to their high opportunistic contact rate with others. For HCMM, the infrastructure saving is in-between Sigcomm and Dartmouth. This reflects the fact that we generate the HCMM contacts patterns for a moderately dense MSN. Similar to other analysis, random replication shows high variation of the results. In summary, the hybrid content replication effectively reduces the total infrastructure usage at the end, compared to simple opportunistic content dissemination.

6.7 Summary

In this chapter, we first showed that the traditional opportunistic content dissemination schemes are not suitable for disseminating content in mobile social networking environments due to the relatively low number of content consumers and short content delivery deadlines. Starting from this, we formally defined the content replication problem in mobile social networks and showed that this is NP-hard. Then, we developed a community based greedy algorithm for efficient content replication by taking advantage of routine behavioural patterns of users. Using both real world and synthetic traces, we showed that content replication can attain a delivery success rate of 80% with less than 10% replication and approximately 60% of the content can be delivered in less than one day. Further, different dynamic centrality metrics were proposed to identify the most influential users within a community and to show that the performance are highly environment dependent.

Since random replication results have shown significant results even without any knowledge of contact patterns of users, it has been further analysed using a human mobility simulator for opportunistic environments. Then, a component based greedy algorithm has been presented and the results show that replicating on just 10% of consumers significantly increases the content delivery to almost all consumers. Moreover, content pre-fetching provides a 80-90% cache hit rate, increasing user satisfaction of the MSN service. Although we propose using networking infrastructure supported replication, the proposed hybrid content dissemination ultimately reduces the networking infrastructure bandwidth usages by approximately 60% when compared to using no replication or a standard dissemination method. In addition, the proposed replication strategies are robust and are not sensitive to the level of collaboration of consumers for supporting replication of content on their devices.

In Chapter 7, we demonstrate the feasibility of the proposed system, namely Yalut, by implementing it on Android smartphones, Mac and Windows desktop platforms. Even though the communication level operations are quite different from the existing centralised content sharing mechanisms, the users do not need to change their behaviour considerably compared to other decentralised content sharing systems. In particular, Yalut leverages of the users' preferred existing content sharing services to interact with their friends. We believe that Yalut will motivate mobile users to take part in decentralised social networking systems as it provides improved privacy, user control and minimises communications cost for the users.

Chapter 7

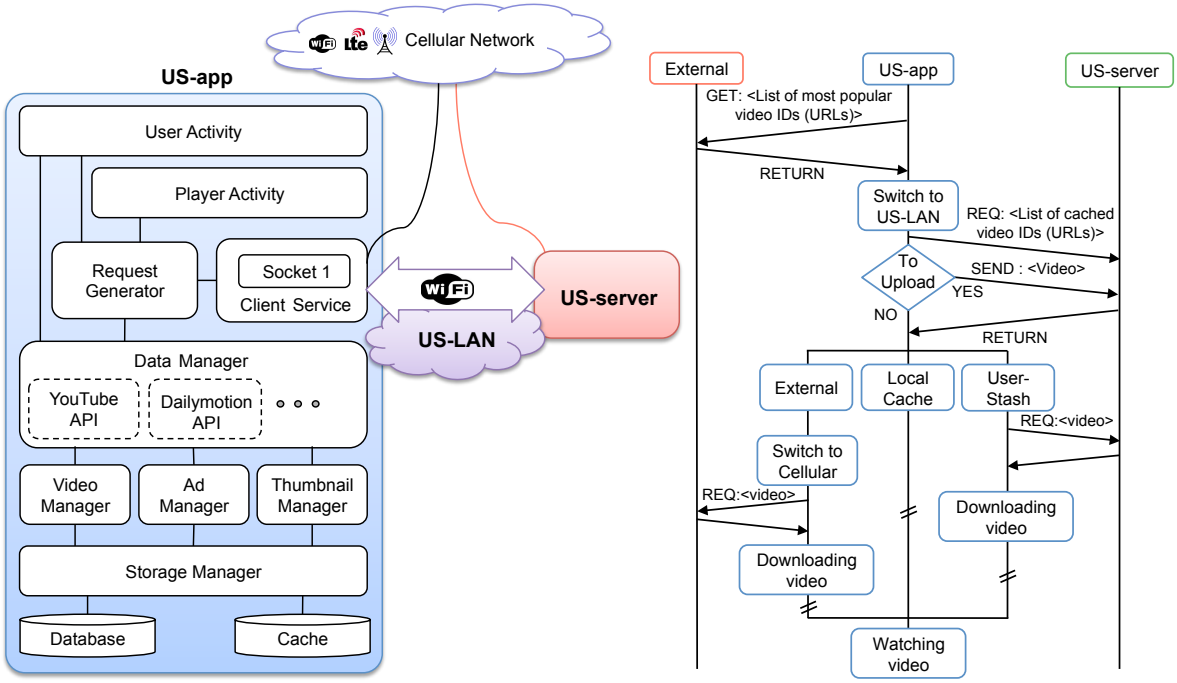
Realisation on Smart Mobile Devices

In previous chapters, we proposed three novel mobile content delivery mechanisms, namely User-Stash, MobiTribe and Yalut, which address cost and privacy associated problems in current mobile content delivery architectures. In this chapter, we demonstrate the feasibility of implementing the proposed mechanisms on commodity mobile devices.

The User-Stash content delivery mechanisms are implemented on Android smartphones as a mobile app. Section 7.1 presents the design decisions and methods of implementation that were used to optimise the scarce resource usage and enhance the usability of the system as an attractive alternative solution for general mobile users. We are in the process of releasing User-Stash for public use and especially on busses from the Central Station, Sydney to the University of New South Wales[1]

The two mechanism of decentralised content sharing, MobiTribe and Yalut, have been combined together for a mobile app, which we named as Yalut. In the process of developing the Android application, we realised that it is vital to have desktop companion applications to enable accessing and sharing content on Yalut using desktop computers. Therefore, Mac and Windows desktop applications were also been developed. Yalut has been released for public use on the Android app store[2] and on the Yalut landing webpage[3]. The details of the implementations of Yalut is described in Section 7.2.

[1] `http://www.userstash.com`
[2] `https://play.google.com/store/apps/details?id=com.yalut&hl=en`
[3] `http://www.yalut.com`



(a) US-app architecture **(b) Communication protocol**

Figure 7.1: US-app implementation and the communication protocol.

7.1 Implementation of User-Stash

User-Stash consists of two main components US-app and US-server as described in Sections 3.1. The US-app enables users to consume and contribute the content that they have downloaded and consumed to a local store, US-server. The US-server hosts the local store and makes it available to US-app users by enabling a local network US-LAN. Although the system concepts are valid for any popular content type, e.g. videos, photos, music, in this implementation we consider the viability of the User-Stash system by focusing only on video content.

7.1.1 US-app client application

US-app can be developed as a standalone mobile app or an extension (plug-in) to mobile web-browsers. Further, it is also possible to provide US-app as an SDK to be used with any app due to the applicability of User-Stash concepts for various other apps. We develop US-app as a standalone mobile app for Android smartphones evaluating the performance of User-Stash as it provides much greater flexibility and allows seamless distribution via mobile app stores.

The basic components of the US-app client application are illustrated in Figure 7.1 and the communication flow of the US-app with other entities (US-server and

external servers) are illustrated in Figure 7.1b. *User Activity* and *Player Activity* are the main activities that interact with the user, which initiates the relevant flow of steps based on the user inputs. When the US-app is launched, *User Activity* initiates a request to get the unique IDs (URLs) of the most popular videos from the video service providers as shown in Figure 7.1b. This version of the US-app was integrated with YouTube and Dailymotion video service providers. *Data Manager* component forwards these requests to *Request Generator* through relevant APIs, e.g. YouTube Data API[4] and Dailymotion Graph API[5]. *Data Manager* can be easily extended to other local video service providers such as PPTV[6]. The *Request Generator* directs these requests via the user's private cellular or WiFi network connection and then, first get the video IDs of the requested content. *User Activity* compares the received video IDs with the video IDs that are already cached in the device via a *Database* query. Then, if there are new video IDs, a new request is generated to fetch the related meta data for those videos such as keywords and thumbnails of the videos. In addition to the most popular content, US-app enables the user to search for a particular video or a set of related videos for a particular keyword on the service provider (currently YouTube and Dailymotion). In this case, the same process will be repeated to receive the video IDs and relevant meta data.

Once the *User Activity* receives all relevant information for the most popular videos or user requested videos, it invokes switching of networking interface to the available US-LAN to obtain the list of videos that are stashed in the US-server as shown in Figure 7.1b. Then, the results are shown to the user with an indication of whether the video is stashed in the US-server or not, as described in Section 7.1.3. At the same time, if there is any new content in the local cache of the US-app that is not in the stash, the app pushes it to the US-server in the background while the user is scrolling through search results. Then the user can select a video clip that is either, (1) cached in the device (*Local Cache*), (2) stashed in the US-server (*User-Stash*), or (3) need to be downloaded from the custodian (*External*). *Player Activity* implements all the services related to video playback (e.g. play, pause, scrolling and full screen) including the content downloading protocol. If a video that can be obtained locally from US-server is chosen by the user, the *User Activity* fetches the content from the US-server via the US-LAN to which, the device is already connected. If a video that needs to be obtained externally is selected, the network interface is switched back to the cellular network, and the

[4]YouTube Data API - `https://developers.google.com/youtube/v3/`
[5]Dailymotion Graph API - `http://www.dailymotion.com/doc/api/graph-api.html`
[6]`http://www.pptv.com`

video is downloaded via the cellular network as shown in the flow diagram in Figure 7.1b. In accessing content either from the US-server or the external website, the content is downloaded to the device instead of streaming the content as it allows US-app to cache the content locally. We developed a custom HTTP client for this purpose using Apache HttpCore [105] libraries. If the user wishes to watch the same video again, it will be played from the local cache. In the case of external websites, then it is possible to upload the locally cached copy to the US-server when the user next connects to a US-LAN. Switching network interfaces is required because the cellular network interface is not active while the device is connected to US-LAN as the majority of current smartphones does not allow simultaneous use of cellular and WiFi networking interfaces. However, this will be a temporary issue, as the most recent smartphones are enabling the simultaneous use of multiple network interfaces[7].

In Section 3.1.3, we proposed an advertisement based incentive scheme for the User-Stash system. US-app has registered with Google AdMob[8] to receive Ads. *Ad Manager* integrates the ad library provided by the registered ad network (Google AdMob) to the US-app. In this prototype implementation, Google Ads are displayed at the bottom of the UI as shown in Figure 7.3 when the user connects to Internet via the user's cellular network. When the user connected to US-LAN, *Ad Manager* displays the Ads provided by the US-server. These local Ads will be stored in the local cache once downloaded. However, we do not cache Google Ads as it is not allowed by the AdMob library. Moreover, *Ad Manager* keeps records of the number of Ads impressions and clicks count of local Ads for accounting purposes.

The local *Cache* and *Database* entries are managed by the *Storage Manager*. The *Database* contains all meta data information related to a particular video file and it also contains the mapping between related thumbnails and Ads for the particular video. External storage (SD-Card) is used to store US-app data, where it is available. *Storage Manager* employs LRU cache replacement algorithm and the user is given the option to specify the maximum allocated storage for US-app *Cache* in the external SD-Card of the device.

7.1.2 US-server application

Similar to US-app, there are multiple ways to implement US-server. It can be developed either on a off-the-shelf mobile device or on a dedicated device, e.g. a single

[7] e.g. Samsung Galaxy S5 enables to boost access speed through simultaneous downloads through WiFi and cellular networks.

[8] `http://www.google.com/ads/admob/`

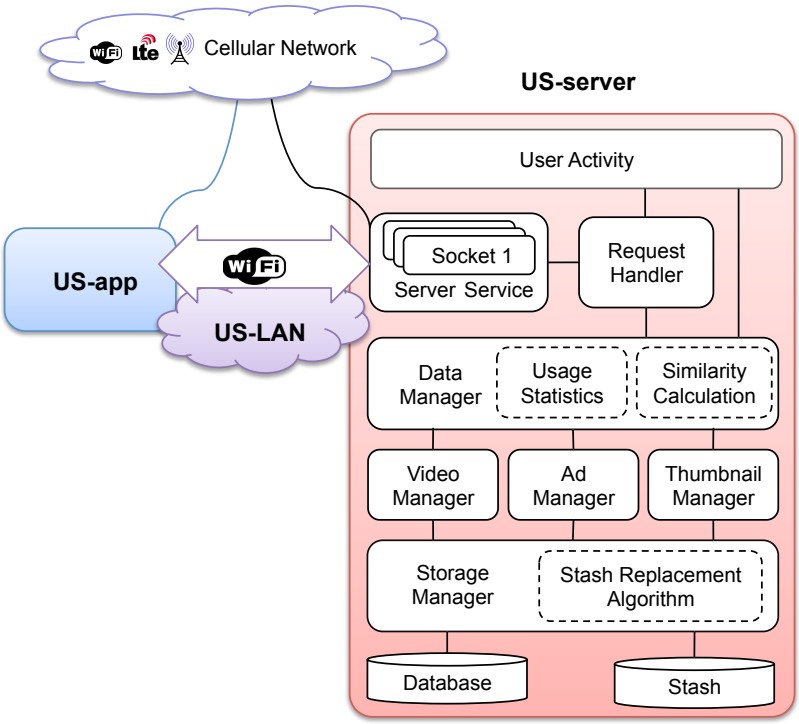


Figure 7.2: US-server app architecture.

board linux computer [106], [107]. Mobile device implementation is an easily deployable solution compared to installation and distribution of a dedicated device, despite advantages of processing and storage of dedicated devices. Therefore to demonstrate the feasibility of the US-server, it was implemented on Android smartphones. Moreover, if the User-Stash service provider distributes US-server devices, a smartphone would be a significant incentive for users to become US-servers. In particular, the User-Stash service provider ensures the US-server candidate device has adequate system capabilities (hardware, memory, etc.) to provide the required (caching and delivery) services.

As majority of the smartphones supports 64GB of external storage and latest phones such as Samsung S5 supports 128GB storage (Table 1.1), 128GB is a sufficient storage for a US-server based on our simulation results presented in Section 4.2. US-server is essentially the same as US-app with a set of extra features to support distribution and storage of content as can be seen in Figure 7.2. The users with appropriate credentials to access US-server features can toggle between US-app and US-server modes through the settings menu of the app. *User Activity* initiates the US-server functionalities by activating WiFi hotspot on the device. Apache HttpCore [105] libraries are used to host HTTP server to communicate with the US-app devices. The *Request Handler* decodes the requests (read or write from/to US-server cache) generated by US-apps and forwards them to the *Data Manager*.

When the user is connected to the US-LAN, US-app will display similar videos that are stashed in the US-server as suggestions to watch next. *Data Manager* computes the Jaccard Similarity Index[9] between the keywords of the given video and all other videos in the *Database* and returns the meta data information of the videos with highest similarity. To reduce redundant data communication, only the 10 most similar videos are displayed. If the user requests more (pressing the "More" button), more information will be send to US-app. *Data Manager* records the usage statistics of videos stashed in the *Stash* and updates the *Database* following each request. In addition, MD5-Hash of each content is computed and stored in the *Database* for the purpose of detecting duplicates and also to perform hash filtering to identify and remove inappropriate content from the US-server.

Ad Manager pushes Ads locally to US-apps via US-LAN and then records Ads statistics (impressions and clicks) for accounting purposes. In this version of the US-server, we focused only on the Ads that can be stored in the US-server without any ad network involvement such as an Ad campaign that directly deals with the User-Stash service provider. *Storage Manager* is responsible for maintaining the mapping of database entries and the relevant videos, Ads and thumbnails. Then, *Storage Manager* takes the stash replacement decision for write requests by applying LRU policy. Section 4.2 presented the performance of different cache replacement policies. Although GDSP showed the highest stash hit rate, the difference between LRU and GDSP is smaller for bandwidth saving. Therefore, LRU was chosen for this implementation considering the trade-off between the simplicity of implementation and performance.

7.1.3 UX-UI design

The interface of the US-app is illustrated in Figure 7.3. We have given special attention in designing the user interface (UI) as that is what matters the most finally[10]. As stated earlier, the user can search the selected video service provider for a particular content. If it is not found, the app will display the most popular content for the particular geographical region. Depending on the user request, the US-app displays the search results or shows the most popular videos with an indication whether the content can be obtained locally, i.e. from within the User-Stash network (the green arrow) or needs to be downloaded via the cellular network connection (red arrow) as shown in Figure 7.3. The US-app also enables the user to view the most

[9] Jaccard Similarity Index of video v_1 and $v_2 = \frac{\|\text{keywords}(v_1) \cap \text{keywords}(v_2)\|}{\|\text{keywords}(v_1) \cup \text{keywords}(v_2)\|}$

[10] We acknowledge the support given by UX designer Phil Grimmett.

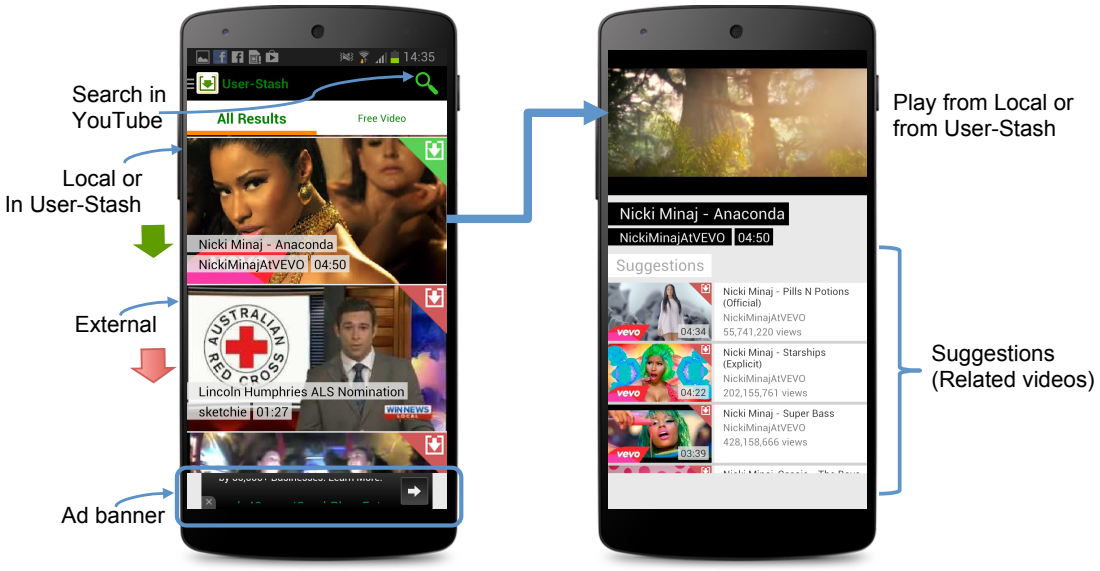


Figure 7.3: US-app user interface - the representation of locally stashed and external videos.

popular videos stored in the US-server in a separate tab (the "Free video" tab in Figure 7.3). Since these videos can be obtained locally, some users may tend to browse and view them to find out *"similar popular content"* and *"what is free for today"*.

When a user clicks on a video, it will trigger the downloading of the particular video from the relevant source. On this screen, the app also suggests a set of related videos to the user. If the user has switched to the US-LAN network, the related videos that are stashed in the User-Stash will be displayed as it allows the user to obtain the content locally, thus reducing the usage of cellular data and the network switching overhead. Otherwise, the suggestions will be similar to YouTube and Dailymotion content service providers. In this case, the app displays whether those suggestions are stashed in the US-server. Moreover, it enables the user to filter the content based on usual video categories such as sports, music, news etc. For example, if the user selects "sports" category, the US-app will fetch the most popular "sports" videos from the particular service provider. Thus, US-app provides services to the user similar to traditional centralised content providers, but for lower cost and improved QoE due to localised content delivery.

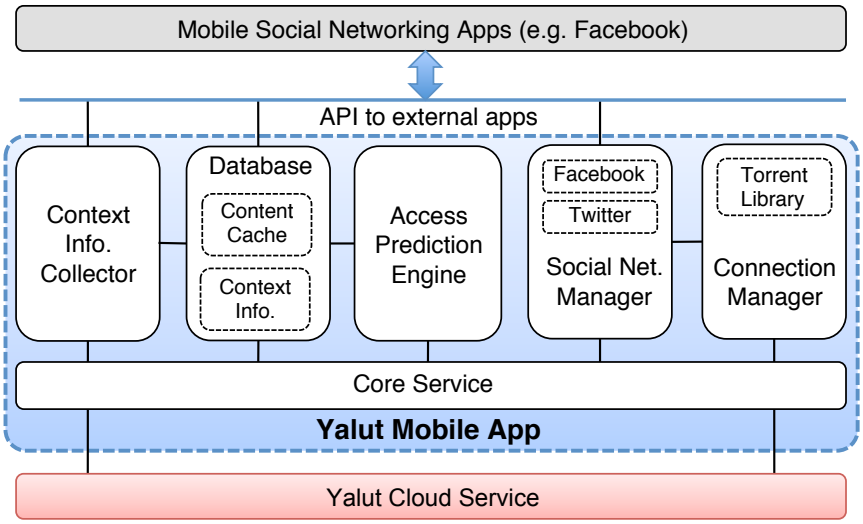


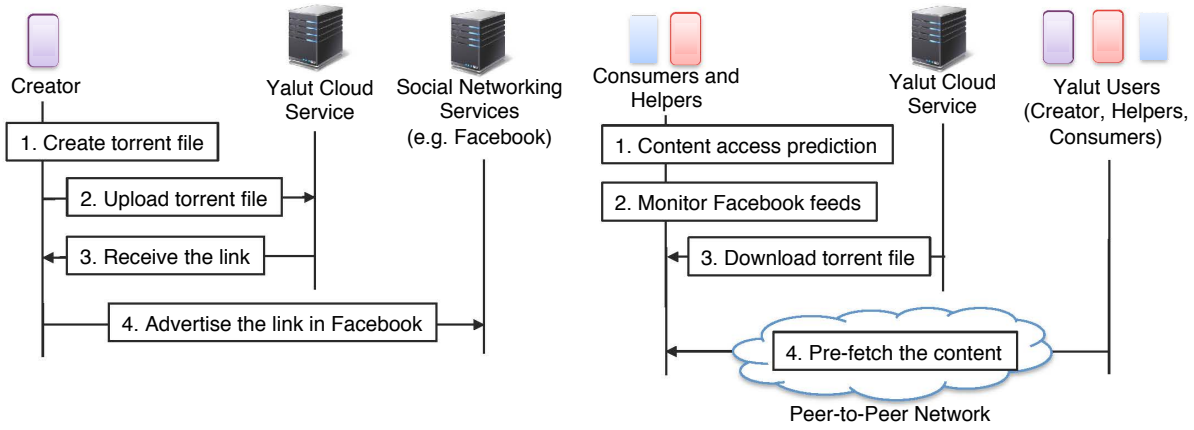
Figure 7.4: Components of Yalut mobile app

7.2 Implementation of Yalut

User-Stash demonstrates the advantages of local content distribution. However, it does not address the issues associated with privacy and user control of data. As discussed thoroughly in Chapter 3, MobiTribe and Yalut content delivery mechanisms are designed to overcome the above shortcomings by acting as an overlay on top of the widely used centralised online social networking services and provides highly efficient content sharing and storage methods, ensures the same level of content availability, reduces the cost of communication and battery usage. In the following, we demonstrate the feasibility of realising these mechanisms proposed in Chapter 5 and 6 on real mobile devices. MobiTribe and Yalut mechanisms are combined to develop a unified mobile system, namely Yalut. Yalut consists of two main entities 1) *Yalut mobile app* and (2) *Yalut cloud service*.

7.2.1 Yalut mobile app

The basic components of the mobile application are shown in Figure 7.4 and the message flow to/from the mobile app to other entities (Yalut cloud service and third party services) are illustrated in Figure 7.5. The *Core Service* component is the main service, which receives *Intents* from Android OS such as the action of sharing a photo or accessing a link, and then it orchestrates the appropriate flows and components. In the case of sharing a photo, *Core Service* calls *Connection Manager*, which manages peer-to-peer communication in Yalut using modified BitTorrent as the communication protocol. tTorrent [109] and libTorrent [111] libraries are used



(a) Content sharing **(b) Content downloading**

Figure 7.5: Yalut message flows in content sharing and downloading.

in this implementation. Even though there are some other open source torrent libraries available, those were not chosen due to license incompatibility and Android support issues. For instance, libraries with GNU GPL version 2.0 license are not compatible with Apache License version 2.0[11], which is the base licence for most of Android libraries. All the external libraries used are summered in Table 7.1, which are compatible with each other.

Connection Manager first creates a torrent file for the content to be shared and then uploads the torrent file to the Yalut cloud service requesting a link to that particular torrent file as shown in Figure 7.5a. With this resulting link, *Core Service* requests *Social Network Manager* to send the "shared notification" through the selected social networking service. In the current version of the app, we have integrated Yalut with three major social networking services, namely Facebook, Google+ and Twitter. Further, we enable the users to send shared notifications through Email and text messages (SMS). It is possible to provide an API to other external services if they wish to link Yalut to disseminate content in order to send this notification through Yalut app. The users have to sign in to the social networking service through the app if they want to send shared notifications using a particular social networking service. This is managed by *Social Network Manager*, which makes use of the Google+ [113], Facebook SDK for Android [117] and the Twitter API [119]. Use of these external libraries makes the system dependent to their terms and conditions of usage as summarised in Table 7.1. For instance, Google APIs Terms of Service [113] forces to have a "Data Breach Policy". This is overcome by promptly

[11]Fedora Licensing Guidelines: `https://fedoraproject.org/wiki/Licensing:Main?rd=Licensing`

Table 7.1: Summary of APIs, Libraries, SDKs used in Yalut development

APIs/Libs/SDKs	Purpose	Terms and Conditions
Android 4.4 SDK	Android development	Android Developer Terms [108]
tTorrent [109]	P2P content distribution	Apache License 2.0 [110]
libTorrent [111]	P2P content distribution	BSD-2 Clause license [112]
Google+ SDK	Integrate with Google+	Google APIs Terms of Service [113] User Content and Conduct Policy [114] Platform Terms of Service [115] Platform Developer Policy [116]
Facebook SDK	Integrate with Facebook	Apache License 2.0 [110] Facebook Developer Terms [117] Statement of Rights and Responsibilities [118]
Twitter API	Integrate with Twitter	Twitter API Terms [119]
Apache Commons	Filename operations	Apache License 2.0 [110]
SQL Lite [120]	Database operations	Apache License 2.0 [110]
Thumbnailator [121]	Thumbnails for images	MIT License [122]
Xuggle-xuggler [123]	Thumbnails for videos	GNU GPL Version 3.0 [124]
Apache HttpClient and HttpCore [105]	Http server at the CMS	Apache License 2.0 [110]

notifying all Yalut users if there is a security breach in the system[12].

After advertising the link to Yalut cloud service, the *Connection Manager* starts seeding the content, i.e. sending periodic messages to the tracker updating the location and contact information of the device. To minimise the increase in energy consumption of the device due to seeding, the maximum number of concurrent seeds that a device can handle is limited to 50 and the default time that a user seeds content after sharing or downloading it, is limited to 1 day. All these parameters can be configured in the advanced settings of the app.

Access Prediction Engine monitors social networking feeds in the background and predicts the content that is likely to be consumed by the user based on the history of content access patterns. For this experimental implementation, the app periodically pre-fetches all content shared through the Yalut service. The content pre-fetching interval is set to 10 minutes by default and can also be set to any value through app settings. Once a shared notification is identified by the *Access Prediction Engine* or by the user, the *Core Service* initiates the content downloading process first by fetching the related torrent file for that content from the Yalut cloud service. The torrent file contains the *announce-URL* of the tracker (Table 7.3) and the *Connection Manager* triggers the peer-to-peer content downloading process contacting the tracker as shown in Figure 7.5b. The content creator or other online users who have

[12]The data breach policy is documented in the Terms of Services of Yalut in Appendix A.1.

Table 7.2: Android permission table for Yalut

Permission	Usage	Purpose
RECEIVE_BOOT_COMPLETED	Receive broadcast message after booting	Automatically restart Yalut at system reboot
WRITE_EXTERNAL_STORAGE	Write to external storage	Write relevant info to external storage
INTERNET	Open network sockets	Upload/download content
ACCESS_NETWORK_STATE	Get connected network info	Upload/download content
ACCESS_WIFI_STATE	Get WiFi network info	Upload/download when only connected to WiFi networks
WAKE_LOCK	Keep processor from sleeping	Run Yalut even when the phone screen is off
GET_ACCOUNTS	Use by Google+	Integrate with Google+
USE_CREDENTIALS	Use by Google+	Integrate with Google+

the same torrent file will take part in forming the peer-to-peer network. The content sharing is set to perform only when devices are connected via WiFi networks to reduce the cost of usage. However, the user is given the ability to change that to any network through the settings menu.

Besides that, *Context Info Collector* records context information such as information of the WiFi access points, remaining battery capacity and spare storage capacity to compute the *Device Availability* to host others' content described in Section 5.1. Device availability pattern will be periodically (once per day via WiFi networks) uploaded to the Yalut cloud server for the purpose of replicator selection. In addition, only if the user permits, *Context Info Collector* records the Yalut app usage statistics for the purpose of Yalut app improvements and future research. Both collected context information and downloaded content are stored in an SQL Lite database [120] on the external storage of the device.

In Android, some operations such as automatic restart of an app after reboot requires user permission at the time of installation. Table 7.2 summarises the user permissions required to install Yalut, the usage and the purpose of the particular permission block. Since Yalut is a privacy preserving app, the app requires the minimum number of user permissions to provide the desired functionalities. As the Android app is to be distributed via the Google Play Store, there are a number of other policies and terms that were considered during the development of Yalut[13]. The Yalut Terms of Services (Appendix A.1) and Privacy Policy (Appendix A.2) were developed by carefully analysing all these policies.

[13]Google Play Terms and Conditions: Developer Distribution Agreement [125], Developer Program Policies [126], Google Content Rating Guidelines [127], Google Privacy Policy [128], Google Play Terms of Service [115] and Google Terms of Service [129]

Security related features

The security threats for Yalut emanate from two sources. 1) Internally, through the trusted social networking friends and through the integrated centralised service providers as they may attempt to access data stored in user devices to gather more information about the users. 2) Externally, through numerous parties such as crawlers and hackers. For this version of the app, we utilise some techniques to improve the security aspects although it can be further improved in the future.

We provide ephemeral content sharing capability in Yalut. Similar to apps such as Snapchat, the content creator will be given the ability to specify the duration that the content is to be made available for consumers. This information is integrated to the torrent file and uploaded to the Yalut cloud service. At the consumers device, this expiry time information is extracted from the torrent file and stored in the local app database. Once the content is downloaded, *Core Service* starts a count down timer for the duration of the expiry time. After the specified expiry time, the content will be removed from all devices as well as the "share notifications" from the social networking services. To make it reliable, all downloaded content are stored in a hidden directory within the Yalut app directory and can only be accessed through the Yalut app. In addition, the content can be encrypted, if necessary. Therefore, even if the smartphone is lost or is hacked, there will not be any security threat as the content is no longer available. The creator is also given the ability to remove any content hosted on the friends' devices through the Yalut service. If the creator deletes any content, Yalut app puts a notification in the Yalut cloud service associated with the particular torrent file. Then, the devices of all users who have downloaded this particular torrent file will be informed to delete the content when they connect with the Yalut cloud service next time. The local Yalut app of the devices takes care of executing these delete requests without any requirement of user interaction. However, it is inevitable to restrict someone copying the content as there are many undetectable ways to copy content such as taking a photo of the content using another device. The idea is to reduce the leakage of private information to third parties such as the centralised social networking services.

In order to enable *blacklisting* as a way of excluding malicious users from the system and also to restrict sharing copyright violated content, we introduce a "Signup" process at the time of installation. Then, the Yalut cloud server uses the Yalut user ID assigned at Signup in all communications with users to keep records of the identities of the Yalut enabled devices. In addition to that, we enable an additional security step by allowing the users enter a user name and password to open the app in Android. Then, even if someone forgot to lock the device, an intruder will not be

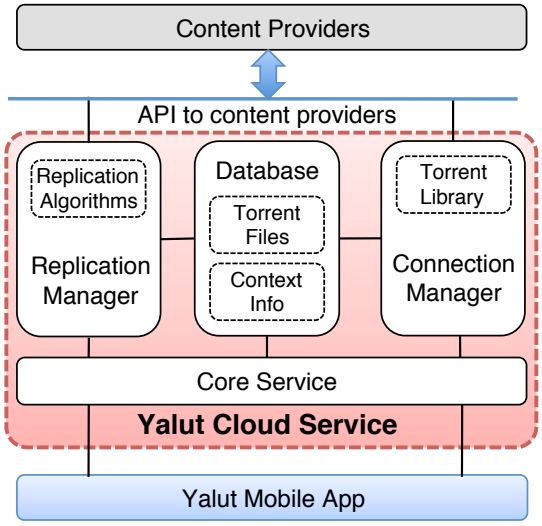


Figure 7.6: Components of Yalut cloud service

able to open the Yalut app without these credentials.

Moreover, Yalut inherits all the security measures imposed by centralised service providers. At the time of sharing, the creator can select which friends or friend circle that the content is to be shared with. At the consumer device, when the user clicks on a Yalut notification to initiate the peer-to-peer download, the *Social Network Manager* ensures whether that particular link appears on the social networking feeds of the user. If it is not in the feeds, Yalut app prevents the downloading of the particular content. This is an extra step of authentication provided by Yalut to restrict downloading of content by stealing the link.

Desktop applications

MobiTribe and Yalut have been initially proposed to be implemented on smart mobile devices. However, it was realised that a desktop companion application is also an essential requirement during our development and initial user testing processes, as people still access content using desktop and laptop computers. Therefore, dedicated Mac and Windows desktop applications were developed using JAVA as the baseline programming language. It is also possible to develop a web-browser plug-in for Yalut. However, a dedicated application provides the option to create Yalut groups to share content in future and to keep consistency in terms of features and UX-UI designs. All functionalities of desktop app are made similar to the Android app, except the following two additions: (1) The users can initiate content sharing by dragging and dropping files into the Yalut directory inside the user's home directory. (2) The users can copy the link to be shared and then distribute the link using any service available to the user such as chat applications.

Table 7.3: Essential user information stored in the CMS

Item	Description
Torrent file	Torrent file contains - announce URL of the tracker, creation date, file length, file name
Hash of the social network IDs Encrypted Yalut user ID Encrypted Yalut password	This is used to uniquely identify the owner of the content to rectify copyright violations
Hash of the content	In order to comply with legal terms, hash of the content is used to filter illegal content. This is used to stop further sharing of content and notify the creators if there is a complaint.
Expiry time	This information is transferred to all downloaders along with the torrent file
Delete Flag	Yalut app checks this flag periodically for the downloaded content
Thumbnail	In order to display thumbnail in social network feeds, it is required to store thumbnail.
Public Flag	If not public, the app checks that the creator is a social networking friend of the downloader.
Device availability	Replication manager uses this information for replicator selection.

7.2.2 Yalut cloud service - Connection Management Server

Yalut central entity is the Connection Management Server (CMS) that manages the communication between Yalut enabled devices. *Replication Manager* (Figure 7.6) determines the replicators to be used for each user periodically, using the device availability patterns stored in the *Database* of the CMS. This is done by the replication algorithms proposed in Sections 3.2 and 3.3. *Replication Manager* notifies all users when there is a change in the set of helper devices for a particular user. The *Connection Manager* component works as a torrent tracker and an indexer. Moreover, it is possible to provide an API to external content providers such as advertisers if they wish to use Yalut content sharing platform to disseminate content among its users.

All the information that are required for the functionalities of Yalut is summarised in Table 7.3. Thumbnail of the content is stored in the CMS to display it in social networking feeds. The creator has the option to stop uploading the thumbnail in the settings menu of the app. As described earlier, Yalut has to keep records of the owners of the shared content for the purpose of blacklisting malicious users and to stop sharing copyright violated content. Therefore, CMS stores hashed social networking IDs, encrypted Yalut username and password for all shared content. In addition, MD5-Hash of each content is also stored to comply with legal terms and

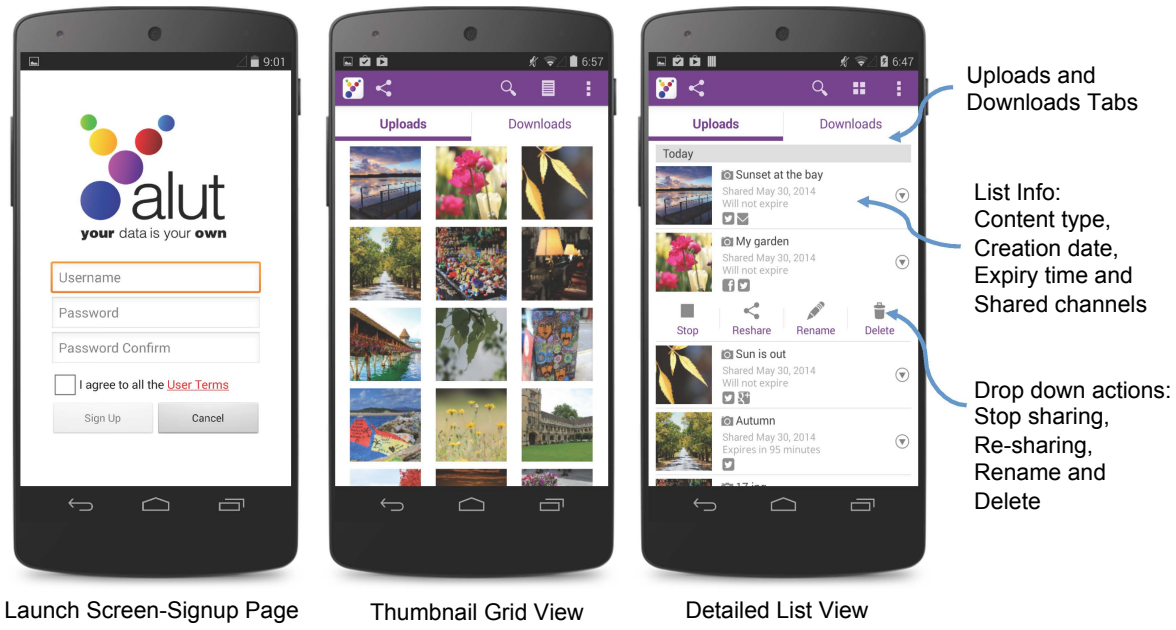


Figure 7.7: Yalut Android app interfaces.

to take down the content if there is a complaint of copyright violations.

In addition, the collected user statistics from Yalut users will also be stored in Yalut cloud service for the purpose of Yalut system improvements and also for research and development of privacy preserving social networking applications. We seek user consent before collecting this information. Further, the users can disable uploading these information via settings menu of the app.

7.2.3 UX-UI design

In real-life implementations, ultimately what matters is the user experience. Therefore, extra effort was taken to design the user interface of the app and to design the process of content sharing and downloading from the user's perspective[14]. To be consistent among different platforms, Android, Mac and Windows UIs are designed to be exactly the same.

At the time of installation, all users are asked to create a Yalut user account, which is shown in Figure 7.7. The users are required to accept the Terms of Services (Appendix A.1) and Privacy Policy (Appendix A.2). These terms and policies are developed by consulting professional lawyers. The next two screens in Figure 7.7 are

[14]The design of Yalut UI was given as an assignment to UX-UI design students at General Assembly, Sydney (`https://generalassemb.ly/sydney`). We acknowledge their support in making user experience more natural and their suggestions has been considered in the final design of the UI.

Figure 7.8: Yalut content sharing steps.

the thumbnail *grid view* and detailed *list view* of the content shared (Uploads tab) and content downloaded (Downloads tab) of each user. The list view shows content creation time, expiry time, file name, file type and the channel that the content is being shared. Then, the drop down menu provides additional control actions of stopping, re-sharing, re-naming and deleting a shared content. For instance, if the user deleted any content shared by him which is under Uploads, the particular content will be deleted from all other devices as soon as those devices connect with the Yalut Cloud Service. On the other hand, if the user deleted any downloaded content, it will only be deleted from the local cache of the device.

Figure 7.8 shows the flow of views that the user goes through when sharing content. When the sharing icon is invoked, the app asks to select the type of content that the user wishes to share. Then, the user is directed to either the photo, video or audio gallery or to the file explorer based on the user input. After that, the app shows Yalut content sharing page, where the user can select the content expiry time and the service through which to send the shared notification, which was designed with the help of a UX-UI design expert. This is shown in Figure 7.8. Once a service is selected, the user is directed to that service. For example, if Facebook is selected, the user is directed to the Facebook sharing page, in which the process will be similar to any usual content sharing in Facebook. Yalut sharing notification looks visually similar to other feeds in the social networks where the link to the CMS is embedded inside. Figure 7.9 depicts Yalut shared notifications in Facebook

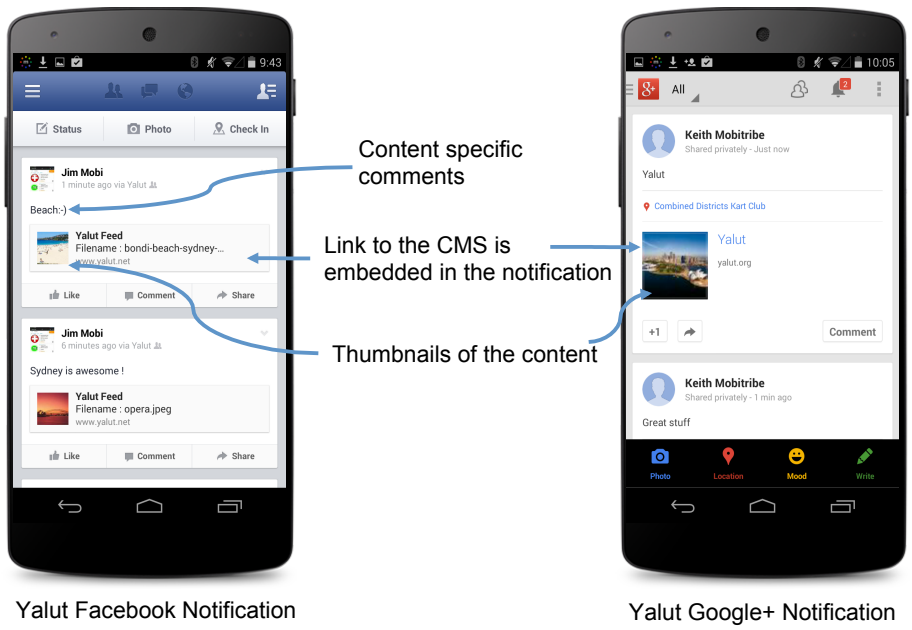


Figure 7.9: Yalut shared notifications in Facebook and Google+.

and Google+. Furthermore, the Mac and Windows application provides the option to copy the link to the clipboard, which can then be copied to any service to send the shared notification, e.g. e-mail and iChat.

7.3 Summary

In this chapter, we presented an overview of prototype implementations of two mobile systems: User-Stash and Yalut, which were developed based on the content delivery mechanisms proposed in Chapters 4, 5 and 6. In User-Stash, US-app and US-server applications have been successfully realised on Android smartphones and the feasibility of providing cost-efficient content access was demonstrated. Yalut and MobiTribe mechanisms are implemented on Android, Mac and Windows platforms as a privacy-aware overlay service for social networking services. Since Yalut enables users to share content from user-to-user without a centralised storage, special measures have been taken to mitigate the malicious usage of the service such as the capability to take down and blacklist rouge users. In addition, terms of services, take down policy and privacy policy have been developed for Yalut by taking legal advice, as this experimental Yalut app is now available for public users via the Google Play Store and Yalut website. Both User-Stash and Yalut implementations have been demonstrated in ACM SIGMOBILE MobiSys'14 conference in July 2014 highlighting the successful realisation of both systems on real devices. From

the lessons learnt during this prototype implementations, we aim to further develop these two systems for the purpose of taking the research outcomes of this thesis for the benefit of general mobile users.

Chapter 8

Thesis Conclusion and Future Work

The primary objective of this thesis was to develop novel mobile content delivery mechanisms to mitigate the problems associated with user privacy and control over user's own data and to reduce the use of cellular network data while improving the user quality of experience and usability. As discussed in Chapter 1, the aim was to take the proposed solutions as close to the ideal performance as possible (Figure 1.1), in the sense high performance in *privacy*, *usability* and *resource-usage-efficiency*.

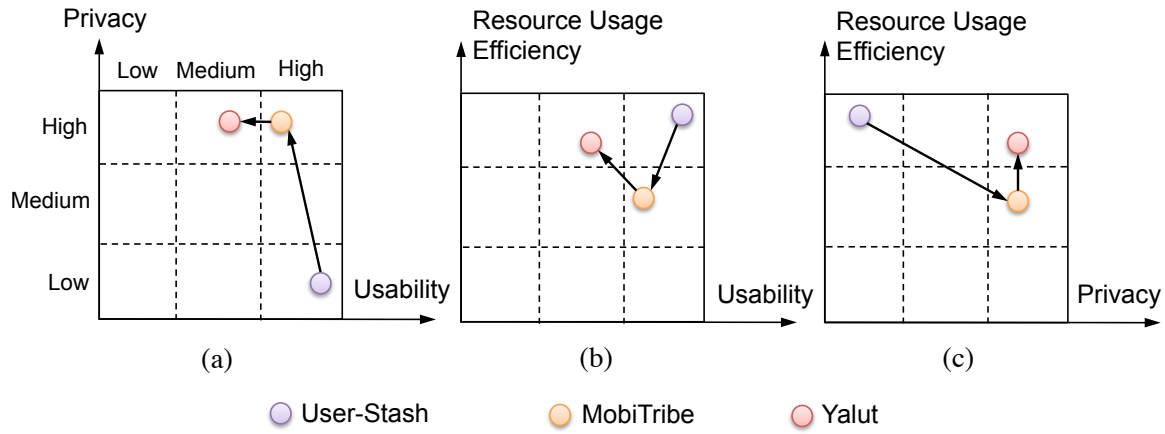


Figure 8.1: Projection of three proposed mechanisms on to the 2D plains of three primary objectives.

The success of achieving the ideal operating conditions with respect to *privacy*, *usability* and *resource-usage-efficiency* is illustrated in Figure 8.1. It schematically illustrates the level of benefits provided by each mechanism - namely User-Stash, MobiTribe and Yalut, in the 2-dimensional (2D) planes of the above three objectives. Firstly, User-Stash presented in Chapter 4 provides high *resource-usage-efficiency*

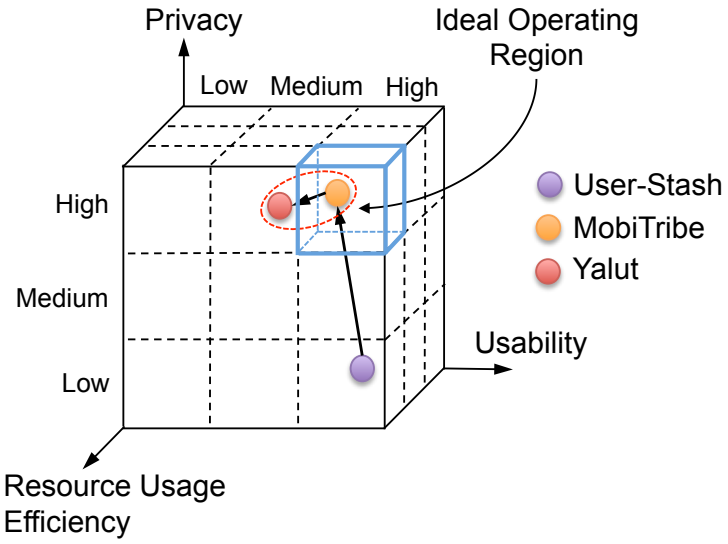


Figure 8.2: Operating regions of proposed mechanisms in 3D solution space.

and high *usability* as shown in Figure 8.1(b). However, it does not solve issues associated with user privacy and user control of data and thus the level of privacy preservation is low (Figure 8.1(a)). Secondly, MobiTribe presented in Chapter 5 primarily focused on protecting user privacy and improving user control of data. Thus, MobiTribe operates in the ideal operating region in *privacy – usability* plane as shown in Figure 8.1(a). The level of *resource-usage-efficiency* of MobiTribe can be considered as medium compared to User-Stash as MobiTribe users need to use more cellular bandwidth in general. Thirdly, Yalut presented in Chapter 6 further improved the *resource-usage-efficiency* of MobiTribe by exploiting the locality and opportunistic low-cost networks. This takes Yalut to the ideal operating region in *resource-usage-efficiency – privacy* plane as shown in Figure 8.1(c). Yalut provides the same level of privacy as MobiTribe, but it reduces the usability to medium level (Figure 8.1(a)) due to the additional content delivery delays associated with opportunistic content dissemination.

The progress of the operating regions of User-Stash, MobiTribe and Yalut in the 3D solution space is illustrated in Figure 8.2. Starting from the bottom of the cube (User-Stash), the proposed mobile content delivery mechanisms move MobiTribe and Yalut solutions adjacent to the ideal operating region. The combination of MobiTribe and Yalut thus provides the best solution as shown in Figure 8.2 and achieves the aims of this thesis that was described in Chapter 1di. In the remainder of this section, we succinctly summarise the novel contributions and subsequent findings realised in the process of developing these mechanisms.

8.1 Summary and Conclusion

User-Stash is a novel crowd-sourced content delivery mechanism that enables users to consume popular content for free in locations such as public transportation systems where there is no access to cheap networks such as WiFi.

The novelty of User-Stash stems from:

- The use of crowd to cache and distribute geographically popular content locally, exploiting the transient co-location and spatio-temporal correlation of content consumption patterns of mobile users.

- Eliminating the need for prediction of user demand as well as low-cost networks, while exploiting the spare storage capacity on mobile devices and the capabilities to host a local WiFi network to create a new low-cost network for all users in the vicinity.

- The method of downloading the content via one network (cellular) and uploading to a stash via another network (WiFi) to which it was download.

- An advertisement based incentive scheme to encourage users to become User-Stash devices.

Using a real-world dataset of video content access patterns and probabilistically modelling the behaviour of users in public transportation systems, we evaluated the performance of User-Stash in terms of stash hit rate and cellular bandwidth usage. We showed that:

- More than 60% stash hit rate can be achieved during a one hour bus ride regardless of the content popularity distribution and User-Stash achieves higher stash hit rate for short travel distances - e.g. metro type transport services, where there is frequent arrival and departure of new commuters.

- More than 80% of the users reduce their cellular network usage at least by 40% when there is a 128GB of storage at the stash in US-server.

- Using the measurements obtained from the experimental implementation on Android mobile devices, the throughput when accessing content locally via the US-LAN from the US-server varies from 1Mbps to 6Mbps depending on the mobile device manufacturers and capabilities.

- The larger the size of the requested content, the lower the relative energy consumption of US-app to download from cellular networks and if the system is able to provide at least a 10% stash hit rate, users are very likely to save on the device energy consumption.

MobiTribe is a novel content sharing mechanism that preserves user privacy and improves user control over their own data by keeping user data away from centralised service providers. MobiTribe addresses the challenges of enabling decentralised social networking services on smart mobile devices by providing continuous availability of content over low-cost network connections. The novelty of MobiTribe can be summarised as follows:

- The formation of Mobile Private Storage Tribe (mTribe) using the mobile devices of groups of friends, which stores and distributes user data whilst improving user privacy and providing the user more control over their own data.

- Formalisation of content replication problem in distributed peer-to-peer architectures and proving that it is NP-Hard.

- Scalable content replication algorithm based on a combination of bipartite b-matching and greedy heuristics that was developed to increase the availability of content exploiting low-cost network availability, battery usage and storage capacity of mobile users.

- Seamless ability to interact with users of existing centralised social networking services improving the usability of the proposed mechanism and the deployability as a service.

MobiTribe content delivery performance in terms of content availability via a low-cost networks and cellular bandwidth saving were evaluated using real-world WiFi connectivity patterns of mobile users and with content creation and consumption modelling. We showed that:

- Persistent availability of content via low-cost networks can be achieved with approximately two to three replicas per content for the considered data sets.

- The theoretical worst case asymptotic time complexity of the proposed content replication algorithm is $O(n^{\Delta})$, where n is the number of friends of the content creator and Δ is the limit of replication per content.

- MobiTribe saves 76% of the uplink and 46% of the downlink cellular bandwidth compared to current widely used centralised server based content sharing architectures with on-the-spot data traffic offloading when the peak content access delay is one day. Moreover, 90% of mobile users can save at least 40% of the cost of their cellular communications.

- Device energy consumption decays exponentially with energy consumption ratio between low-cost networks and cellular networks. Approximately 70% of the devices consume lower energy than a centralised server architecture when WiFi networks are 20 times energy efficient than cellular networks.

Yalut further reduces the cost of content delivery combining the advantages of distributed decentralised storage and opportunistic communications. Yalut's novelty can be highlighted as:

- Hybrid time-aware method of combining distributed storage and opportunistic friend-to-friend content dissemination exploiting the regular behavioural patterns of mobile users.

- Proposal of dynamic centrality metrics considering encounter time of users and duration of encounters to carefully select users to replicate content such that content delivery delay through opportunistic content propagation is minimised.

- Community based greedy content replication algorithm and the use of available lowest cost networking infrastructure to transfer the content to be disseminated to geographically disconnected user communities.

The performance of content delivery in Yalut using opportunistic device-to-device contacts among groups of friends were evaluated using both real-world and synthetic traces. We showed that:

- Content replication selection problem for maximising content delivery success rate while minimising the number of replicas to be NP-Hard.

- Replicating on just 10% of consumers significantly increases the content delivery to almost all consumers for all considered contact patterns of users.

- The proposed hybrid content dissemination ultimately reduces the networking infrastructure bandwidth usage by approximately 60% when compared to using no replication, although the content replication is carried out using networking infrastructure.

- Content prefetching using Yalut provides an 80-90% cache hit rate when the mean access delay is larger than one minute.
- The proposed replication strategies are robust and are not sensitive to the level of collaboration of consumers for supporting replication of content on their devices.

8.2 Future Work

The future work to be done can be categorised into two sections, 1). Research: where we identify mechanisms to further improve the benefits of proposed content delivery mechanisms, and 2). Implementation: where we identify the future work required to develop usable mobile systems for real users extending the current proof-of-concept implementations.

Research: We have not considered the existence of multiple User-Stashes at the same location. In such a scenario, US-servers require to synchronise their stashes by pushing their cache content to each other for optimal performance. In the public transport use case, it is also possible to synchronise the User-Stashes at least once a day at the bus depot. Therefore, there can be multiple ways to take advantage of the coexistence of multiple US-servers. In addition, we focused only on video content delivery in this thesis. However, it is possible to extend the current work for any type of content such as dissemination of news which has very high geographical relevance.

In MobiTribe, we highlighted the possibility of using modified content recommendation algorithms available in the literature. Content recommendation has recently gained much attention due to the significant privacy threats posed by existing content recommendation algorithms. Therefore, alternative decentralised content recommendation algorithms that keep user data on their devices for the purpose of content pre-fetching in MobiTribe would be an interesting research question to be considered in future.

Implementation: In the case of User-Stash, the proof-of-concept Android implementation of US-app and US-server presented here are expected to be developed further to enhance the practical feasibility of User-Stash in terms of user QoE and appropriateness, integrity, and authenticity of content. Through activating WiFi hotspot mode on the mobile device, US-server manages multiple socket connections to all the clients within the communication rage. However, the WiFi chipsets on mobile devices can only handle 8-10 simultaneous connections. This can be over-

come by implementing US-server on single board computers such as Gumstix [106] and Raspberry Pi [107].

Standard practices such as hash filtering (registering with content identification databases[1]) need to be carried out at the US-server to identify and remove inappropriate content from the User-Stash similar to any other user-generated-content distribution service. In addition, adequate take down policies, terms of services and privacy policy need to be developed to address practical deployment issues of integrity and content authenticity.

We consider that the US-server only stashes content that are cacheable as some of the video content providers may not allow users to cache content. However, when User-Stash becomes a popular service, such issues can be mitigated by negotiating with content providers since User-Stash enhances the distribution of their content. Moreover, it will be possible to negotiate with local content providers such as PPTV with subscribed users because their requirement is to deliver the content to all users as cheaply as possible. The US-app can be easily extended to support such a scenario where there are separate channels for specific content providers, e.g. PPTV subscribed users to access PPTV content stored in a User-Stash.

Throughout the thesis, device-to-device communication has been exploited for the purpose of privacy-aware content dissemination. Even though mobile devices are equipped with advanced capabilities to do so, the existing data communication protocols still pose limitations on device-to-device communication. For instance, in Yalut experimental implementation, the device-to-device communication is realised using modified BitTorrent protocol libraries of tTorrent and libTorrent. In order to build a device-to-device connection, this implementation requires UPnP protocol[2] support from the connected network. However, due to the adverse usage of peer-to-peer protocols in some other applications, some networks do not support peer-to-peer traffic. In our initial testing, we found out that most of the cellular networks allow peer-to-peer traffic to pass through. However, most of the free or visitor type WiFi networks do not provide UPnP support. This remains as a potential limitation of the current experimental implementation. Therefore, an investigation of seamless device-to-device communication protocols would be highly beneficial in future not only for decentralised systems such as Yalut, but also for other systems such as increasingly popular Internet-of-Things applications.

We have linked MobiTribe and Yalut with existing social networking service providers because users are not able to or unwilling to migrate from their current

[1] e.g. https://www.audiblemagic.com/content-databases/

[2] `http://upnp.org/about/what-is-upnp/`

services due to personal data being locked in with these services and all their friends already using these services. Therefore, the system is being developed as an overlay service to all other centralised services. However, these centralised service providers may consider Yalut as a threat to their service as they do not get to store user content. For instance, if Facebook thinks Yalut is a threat to their service, it is possible for them to block their API access anytime. This can be overcome by extending Yalut to enable users to create Yalut groups to share content without using these centralised providers.

We had not thoroughly investigated the concerns of data storage and communication security vulnerabilities. Alternative and more secure methods to enable ephemeral content sharing can be investigated such as self-destructive content regardless of the file systems and operating platform. In addition, Yalut cloud service has to be extended to utilise other standard techniques to detect attacks such as denial-of-service (DoS), spamming or Sybil attacks. Further, mobile optimised encryption techniques can be developed to strengthen the content integrity and authenticity of both User-Stash and Yalut systems.

Bibliography

[1] Cisco, "Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013–2018," 2013.

[2] Samsung galaxy s5 specifications, [Online]. Available: `http://www.samsung.com/au/consumer/mobile-phone/mobile-phone/smartphone/SM-G900IZKAXSA-spec`.

[3] eMarketer. (Feb 2012), [Online]. Available: `https://www.emarketer.com/coverage/socialmedia.aspx`.

[4] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2011–2016," 2012.

[5] A. Balachandran, V. Aggarwal, E. Halepovic, J. Pang, S. Seshan, S. Venkataraman, and H. Yan, "Modeling web quality-of-experience on cellular networks," in *Proc. of the 20th Annual International Conference on Mobile Computing and Networking*, Maui, Hawaii, USA, 2014, pp. 213–224.

[6] 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*, vol. 11, no. 5, pp. 821–834, 2012.

[7] M. Barbera, J. Stefa, A. Viana, M. de Amorim, and M. Boc, "VIP delegation: Enabling VIPs to offload data in wireless social mobile networks," in *DCOSS'11*, IEEE, 2011, pp. 1–8.

[8] S. Ioannidis, L. Massoulie, and A. Chaintreau, "Distributed caching over heterogeneous mobile networks," *SIGMETRICS Perform. Eval. Rev.*, vol. 38, no. 1, pp. 311–322, Jun. 2010.

[9] J. Whitbeck, Y. Lopez, J. Leguay, V. Conan, and M. D. De Amorim, "Push-and-track: saving infrastructure bandwidth through opportunistic forwarding," *Pervasive and Mobile Computing*, vol. 8, no. 5, pp. 682–697, 2012.

[10] 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*, Philadelphia, USA, 2010, 26:1–26:12.

[11] A. Balasubramanian, R. Mahajan, and A. Venkataramani, "Augmenting mobile 3G using WiFi," in *ACM MobiSys '10*, 2010, pp. 209–222.

[12] A. J. Nicholson and B. D. Noble, "Breadcrumbs: forecasting mobile connectivity," in *ACM MobiCom'08*, California, 2008, pp. 46–57.

[13] A. Khan, V. Subbaraju, A. Misra, and S. Seshan, "Mitigating the true cost of advertisement-supported free mobile applications," in *ACM HotMobile'12*, California, 2012.

[14] F. Qian, Z. Wang, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, "Characterizing radio resource allocation for 3G networks," in *ACM IMC'10*, 2010, pp. 137–150.

[15] F. Qian, K. S. Quah, J. Huang, J. Erman, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck, "Web caching on smartphones: ideal vs. reality," in *ACM MobiSys'12*, 2012, pp. 127–140.

[16] B. D. Higgins, J. Flinn, T. Giuli, B. Noble, C. Peplin, and D. Watson, "Informed mobile prefetching," in *ACM MobiSys'12*, 2012, pp. 155–168.

[17] L. Robert, "Data prefetching algorithm in mobile environments," *European Journal of Scientific Research*, vol. 28, no. 3, pp. 478–491, 2009.

[18] Diaspora. [Online]. Available: `http://joindiaspora.org`.

[19] L. Cutillo, R. Molva, and T. Strufe, "Safebook: a privacy-preserving online social network leveraging on real-life trust," *Communications Magazine, IEEE*, vol. 47, no. 12, pp. 94–101, 2009.

[20] 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*, vol. 20, no. 2, pp. 127–138, 2008.

[21] R. Sharma and A. Datta, "Supernova: super-peers based architecture for decentralized online social networks," in *COMSNETS'12*, IEEE, 2012, pp. 1–10.

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

[23] A. Shakimov, H. Lim, R. Cáceres, L. P. Cox, K. Li, D. Liu, and A. Varshavsky, "Vis-a-vis: privacy-preserving online social networking via virtual individual servers," in *Communication Systems and Networks (COMSNETS), 2011 Third International Conference on*, IEEE, 2011, pp. 1–10.

[24] Snapchat. (2013). How snaps are stored and deleted, [Online]. Available: `http://blog.snapchat.com/post/50060403002/how-snaps-are-stored-and-deleted`.

[25] J. A. Racoma, *Attention smartphone users: your internet connection is just about to crawl to a halt*, 2013. [Online]. Available: `http://http://www.androidauthority.com/smartphone-data-use-outpacing-capacity-149489/`.

[26] A. J. Mashhadi and P. Hui, "Proactive caching for hybrid urban mobile networks," *University College London, Tech. Rep*, 2010.

[27] G. M. Lee, S. Rallapalli, W. Dong, Y.-C. Chen, L. Qiu, and Y. Zhang, "Mobile video delivery via human movement," in *IEEE SECON'13*, IEEE, 2013, pp. 406–414.

[28] J. Erman, A. Gerber, M. Hajiaghayi, D. Pei, S. Sen, and O. Spatscheck, "To cache or not to cache: the 3g case," *Internet Computing, IEEE*, vol. 15, no. 2, pp. 27–34, 2011.

[29] Z. Wang, F. X. Lin, L. Zhong, and M. Chishtie, "How far can client-only solutions go for mobile browser speed?" In *ACM WWW'12*, 2012, pp. 31–40.

[30] A. Finamore, M. Mellia, Z. Gilani, K. Papagiannaki, V. Erramilli, and Y. Grunenberger, "Is there a case for mobile phone content pre-staging?" In *ACM CoNEXT'13*, 2013, pp. 321–326.

[31] N. Gautam, H. Petander, and J. Noel, "A comparison of the cost and energy efficiency of prefetching and streaming of mobile video," in *Proc. of the 5th Workshop on Mobile Video*, ACM, 2013, pp. 7–12.

[32] U. Rathnayake, M. Ott, and A. Seneviratne, "Network availability prediction with hidden context," *Perform. Eval.*, vol. 68, no. 9, pp. 916–926, 2011.

[33] J. B. Gomes, C. Phua, and S. Krishnaswamy, "Where will you go? mobile data mining for next place prediction," in *Data Warehousing and Knowledge Discovery*, Springer, 2013, pp. 146–158.

[34] M. Taghizadeh, K. Micinski, and S. Biswas, "Distributed cooperative caching in social wireless networks," *IEEE Transactions on Mobile Computing*, vol. 12, no. 6, pp. 1037–1053, 2013.

[35] A.-K. Pietiläinen and C. Diot, "Dissemination in opportunistic social networks: the role of temporal communities," in *Proceedings of the thirteenth ACM international symposium on Mobile Ad Hoc Networking and Computing*, ACM, 2012, pp. 165–174.

[36] C. Boldrini, M. Conti, and A. Passarella, "Contentplace: social-aware data dissemination in opportunistic networks," in *Proceedings of the 11th international symposium on Modeling, analysis and simulation of wireless and mobile systems*, ACM, 2008, pp. 203–210.

[37] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay-tolerant networks," *Mobile Computing, IEEE Transactions on*, vol. 10, no. 11, pp. 1576–1589, 2011.

[38] A. Mtibaa, M. May, C. Diot, and M. Ammar, "Peoplerank: social opportunistic forwarding," in *INFOCOM, 2010 Proceedings IEEE*, 2010, pp. 1–5.

[39] P. Lungaro, Z. Segall, and J. Zander, "Activecast-a network and user aware mobile content delivery system," in *Proc. of ICUFN*, Jun. 2010, pp. 309–313.

[40] T. Ye, H. Jacobsen, R. Katz, *et al.*, "Mobile awareness in a wide area wireless network of info-stations," in *Proceedings of the 4th ACM/IEEE MobiCom*, ACM, 1998, pp. 109–120.

[41] H. Petander, "Energy-aware network selection using traffic estimation," in *Proc. of the 1st ACM MICNET '09*, Beijing, 2009, pp. 55–60.

[42] ByteMobile, "Mobile analytics report," in *http://www.bytemobile.com*, 2011.

[43] 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*, Chicago, Illinois, USA, 2009, pp. 280–293.

[44] A. Rahmati and L. Zhong, "Context-based network estimation for energy-efficient ubiquitous wireless connectivity," *IEEE Transactions on Mobile Computing*, vol. 10, pp. 54–66, 2011.

[45] M.-R. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely, "Energy-delay tradeoffs in smartphone applications," in *MobiSys '10*, San Francisco, California, USA, 2010, pp. 255–270.

[46] A. Mahdian, J. Black, R. Han, and S. Mishra, "Myzone: a next-generation online social network," in *Tech Report: Department of Computer Science, University of Colorado at Boulder*, 2011.

[47] Napster, [Online]. Available: `http://opennap.sourceforge.net/`.

[48] P. Maymounkov and D. Mazieres, "Kademlia: a peer-to-peer information system based on the xor metric," *Peer-to-Peer Systems*, pp. 53–65, Feb. 2002.

[49] B. Cohen. (2008). The bittorrent protocol specification, [Online]. Available: `http://www.bittorrent.org/beps/bep0003.html`.

[50] A. Rowstron and P. Druschel, "Storage management and caching in past, a large-scale, persistent peer-to-peer storage utility," *SIGOPS Oper. Syst. Rev.*, vol. 35, pp. 188–201, 5 2001.

[51] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, *et al.*, "Oceanstore: an architecture for global-scale persistent storage," *ACM Sigplan Notices*, vol. 35, no. 11, pp. 190–201, 2000.

[52] R. Sharma, A. Datta, M. DeH'Amico, and P. Michiardi, "An empirical study of availability in friend-to-friend storage systems," in *IEEE P2P'11*, IEEE, 2011, pp. 348–351.

[53] R. Gracia-Tinedo, M. S. Artigas, and P. Garcia Lopez, "Analysis of data availability in F2F storage systems: When correlations matter," in *IEEE P2P'12*, 2012, pp. 225 –236.

[54] J. Li and F. Dabek, "F2F: Reliable Storage in Open Networks," in *Proc. of IPTPS*, 2006.

[55] D. N. Tran, F. Chiang, and J. Li, "Friendstore: cooperative online backup using trusted nodes," in *Proceedings of the 1st Workshop on Social Network Systems*, ACM, 2008, pp. 37–42.

[56] K. Graffi, S. Podrajanski, P. Mukherjee, A. Kovacevic, and R. Steinmetz, "A distributed platform for multimedia communities," in *Multimedia, 2008. ISM 2008. Tenth IEEE International Symposium on*, IEEE, 2008, pp. 208–213.

[57] S. Buchegger and A. Datta, "A case for p2p infrastructure for social networks-opportunities andchallenges," in *Wireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on*, IEEE, 2009, pp. 161–168.

[58] S. Buchegger, D. Schioberg, L.-H. Vu, and A. Datta, "Peerson: p2p social networking: early experiences and insights," in *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, ACM, 2009, pp. 46–52.

[59] R. Geambasu, T. Kohno, A. A. Levy, and H. M. Levy, "Vanish: increasing data privacy with self-destructing data.," in *USENIX Security Symposium*, 2009, pp. 299–316.

[60] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[61] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Comput. Surv.*, vol. 35, no. 2, pp. 114–131, Jun. 2003.

[62] R. Empson. (2013). Not-so-ephemeral messaging: new snapchat "hack" lets users save photos forever, [Online]. Available: `http://techcrunch.com/2013/01/22/not-so-ephemeral-messaging-new-snapchat-hack-lets-users-save-photos-forever/`.

[63] P. Ducklin. (2013). Snapchat images that have "disappeared forever" stay right on your phone, [Online]. Available: `http://nakedsecurity.sophos.com/2013/05/10/snapchat-images-that-have-disappeared-forever-stay-right-on-your-phone/`.

[64] G. Dunn. (2013). Yet another way to retrieve deleted snapchat photos, [Online]. Available: `http://www.salon.com/2013/06/04/yet_another_way_to_retrieve_deleted_snapchat_photos_partner/`.

[65] 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*, ACM, 2010, p. 8.

[66] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: an online social network with user-defined privacy," in *ACM SIGCOMM Computer Communication Review*, ACM, vol. 39, 2009, pp. 135–146.

[67] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute based encryption," in *Security and Privacy, 2007. SP07. IEEE Symposium on*, IEEE, 2007, pp. 321–334.

[68] Z. Wang, L. Sun, W. Zhu, S. Yang, H. Li, and D. Wu, "Joint social and content recommendation for user-generated videos in online social network," *Multimedia, IEEE Transactions on*, vol. 15, no. 3, pp. 698–709, 2013.

[69] M. Cunche, M.-A. Kaafar, and R. Boreli, "I know who you will meet this evening! linking wireless devices using wi-fi probe requests," in *IEEE WoWMoM'12*, 2012, pp. 1–9.

[70] U. Sadiq, M. Kumar, and M. Wright, "Crisp: collusion-resistant incentive-compatible routing and forwarding in opportunistic networks," in *Proceedings of the 15th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, ACM, 2012, pp. 69–78.

[71] A. Abhari and M. Soraya, "Workload generation for youtube," *Multimedia Tools and Applications*, vol. 46, no. 1, pp. 91–118, 2010.

[72] X. Cheng, C. Dale, and J. Liu, "Statistics and social network of youtube videos," in *Proc. of 16th International Workshop on Quality of Service*, Enschede, Jun. 2008, pp. 229–238.

[73] Z. Li, J. Lin, M.-I. Akodjenou, G. Xie, M. A. Kaafar, Y. Jin, and G. Peng, "Watching videos from everywhere: a study of the pptv mobile vod system," in *ACM IMC'12*, 2012, pp. 185–198.

[74] M. Berry and M. Hamilton, "Changing urban spaces: mobile phones on trains," *Mobilities*, vol. 5, no. 1, pp. 111–129, 2010.

[75] D. A. Hensher, "Measurement of the valuation of travel time savings," *Journal of Transport Economics and Policy (JTEP)*, vol. 35, no. 1, pp. 71–98, 2001.

[76] S. Jin and A. Bestavros, "Popularity-aware greedy dual-size web proxy caching algorithms," in *Distributed computing systems, 2000. Proceedings. 20th international conference on*, 2000, pp. 254–261.

[77] "Crawdad:" in *http://crawdad.cs.dartmouth.edu*.

[78] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, Plenum Press, 1972, pp. 85–103.

[79] 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*, 1983, pp. 448–456.

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

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

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

[83] A. V. Goldberg and R. E. Tarjan, "A new approach to the maximum-flow problem," *J. ACM*, vol. 35, pp. 921–940, 4 1988.

[84] R. Dunbar, "Social networks: electronic networking," *New Scientist*, vol. 214, no. 2859, pp. vi –vii, 2012. [Online]. Available: `http://www.sciencedirect.com/science/article/pii/S0262407912608574`.

[85] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow, "The anatomy of the facebook social graph," *Arxiv preprint arXiv:1111.4503*, 2011.

[86] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin, "Diversity in smartphone usage," in *MobiSys '10*, California, 2010, pp. 179–194.

[87] C. Wilson, B. Boe, A. Sala, K. Puttaswamy, and B. Zhao, "User interactions in social networks and their implications," in *Pro. of the ACM European conference on Computer systems*, Acm, 2009, pp. 205–218.

[88] Cisco, "Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2012–2017," 2013.

[89] ——, "Cisco Visual Networking Index: Usage," in *http://www.cisco.com*, 2010.

[90] 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)*, vol. 17, no. 5, pp. 1357–1370, 2009.

[91] D. Ferreira, A. Dey, and V. Kostakos, "Understanding human-smartphone concerns: a study of battery life," *Pervasive Computing*, pp. 19–33, 2011.

[92] A. Abhari and M. Soraya, "Workload generation for youtube," *Multimedia Tools and Applications*, vol. 46, no. 1, pp. 91–118, 2010.

[93] D. Kotz, T. Henderson, I. Abyzov, and J. Yeo, *CRAWDAD data set dartmouth campus*, Sep. 2009. [Online]. Available: `http://crawdad.cs.dartmouth.edu/dartmouth/campus`.

[94] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on opportunistic forwarding algorithms," *Mobile Computing, IEEE Transactions on*, vol. 6, pp. 606–620, 2007.

[95] A.-K. Pietiläinen, E. Oliver, J. LeBrun, G. Varghese, and C. Diot, "Mobiclique: middleware for mobile social networking," in *Proceedings of the 2nd ACM workshop on Online social networks*, ACM, 2009, pp. 49–54.

[96] C. Boldrini, M. Conti, and A. Passarella, "Users mobility models for opportunistic networks: the role of physical locations," *Proc. of IEEE WRECOM*, p. 23, 2007.

[97] V. Kann, "On the approximability of np-complete optimization problems," PhD thesis, Royal Institute of Technology Stockholm, 1992.

[98] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proc. of the ACM SIGKDD'03*, ACM, 2003, pp. 137–146.

[99] W. jen Hsu and A. Helmy, *CRAWDAD data set usc/mobilib (v. 2008-07-24)*, Jul. 2008.

[100] A. Mei and J. Stefa, "SWIM: A simple model to generate small mobile worlds," in *INFOCOM 2009, IEEE*, 2009, pp. 2106–2113.

[101] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, *CRAWDAD data set cambridge/haggle (v. 2009-05-29)*, May 2009.

[102] K. Thilakarathna, A. C. Viana, A. Seneviratne, and H. Petander, "The Power of Hood Friendship for Opportunistic Content Dissemination in Mobile Social Networks," INRIA, France, Tech. Rep. TR7002, 2012.

[103] K. Thilakarathna, A. Karim, H. Petander, and A. Seneviratne, "MobiTribe: Enabling Device Centric Social Networking on Smart Mobile Devices," in *Proc. of IEEE SECON'13 Demostrations*, New Orleans, 2013, pp. 230–232.

[104] K. Thilakarathna, A. C. Viana, A. Seneviratne, and H. Petander, "Mobile social networking through friend-to-friend opportunistic content dissemination," in *ACM MobiHoc'13*, Bangalore, India, 2013, pp. 263–266.

[105] Apache HttpClinet and HttpCore, [Online]. Available: `http://hc.apache.org/downloads.cgi`.

[106] Gumstix, [Online]. Available: `https://www.gumstix.com`.

[107] Raspberrypi, [Online]. Available: `http://www.raspberrypi.org`.

[108] Android Developer Terms, [Online]. Available: `http://developer.android.com/sdk/terms.html`.

[109] tTorrent Library, [Online]. Available: `https://github.com/turn/ttorrent`.

[110] The-Apache-Software-Foundation. (2004). Apache License Version 2.0, [Online]. Available: `http://www.apache.org/licenses/LICENSE-2.0.html`.

[111] libTorrent Library, [Online]. Available: `http://www.libtorrent.org`.

[112] Open-Source-Initiative. (1998). BSD-2 Clause License, [Online]. Available: `http://opensource.org/licenses/BSD-2-Clause`.

[113] Google. (2014). Google+ sdk: google apis terms of service, [Online]. Available: `https://developers.google.com/terms/`.

[114] Google+ SDK: User Content and Conduct Policy, [Online]. Available: `http://www.google.com/intl/en/+/policy/content.html`.

[115] Google+ SDK: Google+ Platform Terms of Service, [Online]. Available: `https://developers.google.com/+/terms`.

[116] Google+: Platform Developer Policy, [Online]. Available: `https://developers.google.com/+/policies`.

[117] Facebook Developer Terms, [Online]. Available: `https://developers.facebook.com/policy`.

[118] Facebook: Statement of Rights and Responsibilities, [Online]. Available: `https://www.facebook.com/legal/terms`.

[119] Twitter API Terms, [Online]. Available: https://dev.twitter.com/terms/api-terms.

[120] Atlassian. (2014). SQL Lite JSBC Library, [Online]. Available: https://bitbucket.org/xerial/sqlite-jdbc#markdown-header-usage.

[121] Thumbnailator Library, [Online]. Available: http://code.google.com/p/thumbnailator/.

[122] MIT License, [Online]. Available: http://opensource.org/licenses/mit-license.php.

[123] Xuggle-Xuggler Library, [Online]. Available: http://www.xuggle.com.

[124] GNU General Public License Version 3.0, [Online]. Available: https://www.gnu.org/copyleft/gpl.html.

[125] Google: Developer Distribution Agreement, [Online]. Available: https://play.google.com/intl/ALL_au/about/developer-distribution-agreement.html.

[126] Google: Developer Program Policies, [Online]. Available: http://play.google.com/about/developer-content-policy.html.

[127] Google Content Rating Guidelines, [Online]. Available: https://support.google.com/googleplay/android-developer/answer/188189.

[128] Google Privacy Policy, [Online]. Available: http://www.google.com/intl/en/policies/privacy/.

[129] Google Terms of Service, [Online]. Available: http://www.google.com/intl/en/policies/terms/.

Appendix A

Yalut – Policies

A.1 Terms of Services

Last updated 15 May 2014.

In this document, the expressions "we", "us", "our" and "NICTA" are a reference to National ICT Australia Limited, ABN 62 102 206 173 of Level 5, 13 Garden Street, Eveleigh NSW 2015, Australia.

These terms of service ("Terms of Service") apply to your access to and use of our Yalut social networking service (the "Services" or "Yalut Social Network"), including to: downloading and sharing of music, photos, videos or other material (the "Content"); use of our associated software applications (the "Software"), such as our Android app on Google Play; and our website, http://yalut.com (the "Website"). If you do not accept these Terms of Service, you must refrain from using our Services.

1. Amendments
These Terms of Service and all information appearing on the Yalut Social Network are subject to change. Amendments to these Terms of Service will be effective immediately upon notification via our Software or Website. Your continued use of the Services following notification will represent an agreement by you to be bound by the terms and conditions as amended.

2. General Obligations

- You must not engage in unlawful activities.
- You must comply with these Terms of Service.
- You must be at least 16 years old.
- You may only register for one user account.
- You must not pretend to be anyone else.
- You may not share your account with others.
- You must not use or access anyone else's account.

- You must cease using our Services if we terminate your account.
- You may only use your account for personal, non-commercial purposes.
- You must not use your account for financial benefit of any kind.
- You may only use our Software to access any portion of the Yalut Social Network other than our Website.

3. Additional obligations relating to Content

Our Software allows you to share Content using peer-to-peer technology. If you share, or offer to share, any Content you must ensure that it:

- does not contain any sexually explicit material;
- does not infringe copyright or other intellectual property rights (including moral rights) of any person;
- does not infringe the confidentiality or privacy rights of any person;
- does not discriminate against any person;
- does not expose any person to viruses, malicious computer code or other forms of interference that may damage a computer system; and is not defamatory, illegal or otherwise prohibited by laws which apply to you or to us.

You also acknowledge and agree that:

- You are responsible for the activity that occurs under your account.
- Any use or reliance on Content you obtain via the Yalut Social Network is at your own risk.
- You must procure on behalf of yourself and your recipients, and on behalf of us, all proper licences, clearances, permissions and releases in writing in respect of any material (including copyright material) included in your Content before you distribute it via the Yalut Social Network.

4. Access to third party social networks

Our Software may allow you to access some of your third party social networking accounts, so that you can notify your friends on other social networks about Content that you wish to share with them via the Yalut Social Network. For example, you may be able to access your Facebook, Google+ or Twitter accounts.

If you access another social networking service through our Software, your access to and use of the third party service will be subject to its terms of service. To the extent of any inconsistency, the terms of service of the third party service will prevail over these Terms of Service with respect to your access to and use of it through our Software.

You acknowledge and agree that third party social networks are not under our control, and that it is solely your responsibility to ensure that your access to and use of any third party service complies with its terms of service. We are not responsible for any loss arising from your use of any third party service, such as any termination of your account on a third party social network due to your failure to comply with its terms.

You acknowledge that our Software facilitates your access to certain third party social networks as a convenience only. We are not affiliated with, and facilitating such access does not constitute an endorsement of, any such third party social network, or a representation that use of the Yalut Social Network in conjunction with a third party social network will comply with its terms of service.

5. Privacy
These Terms of Service are to be read in conjunction with our Supplemental Privacy Policy.

6. Notifying us about alleged copyright infringement
These Terms of Service require users of our Services to respect the intellectual property rights (including copyright) of others. If you wish to notify us that our Services have been used in connection with an activity that you allege constitutes copyright infringement, please refer to the "Copyright Infringement Notifications" page on our Website.

7. Registration
You must register in order to use the our Services.
To register, complete your registration details in the manner described in our FAQ. Registration is free but non-transferable. When registering, you must choose a user name and password. Your password must be kept secure and may not be shared. You may choose to use a pseudonym as your user name, but you must not pretend to be or otherwise impersonate another person.

By registering, you acknowledge that you have read, understood and accept these Terms of Service. These Terms of Service, as well as our Supplemental Privacy Policy, will be displayed to you at the time of your registration, and will not be able to complete your registration without checking a box to affirm your assent to these Terms of Service and our Supplemental Privacy Policy. If you do not understand and agree to these terms, you are not permitted to register.

8. Disclaimer
OUR SOFTWARE AND THE SERVICES ARE PROVIDED "AS IS" WITHOUT EXPRESS OR IMPLIED WARRANTY OF ANY KIND, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW. THE APPEARANCE OF A HYPERLINK OR CONTENT FOR ANY DURATION DOES NOT CONSTITUTE AN APPROVAL, ADOPTION, PROMOTION, RATIFICATION OR ENDORSEMENT OF ITS CONTENTS BY US.

TO THE MAXIMUM EXTENT PERMITTED BY LAW, WE DISCLAIM ALL LIABILITY FOR ANY LOSS OR DAMAGE, HOWEVER CAUSED (INCLUDING THROUGH NEGLIGENCE OR UNAVAILABILITY OR PERFORMANCE OF THE SERVICES), WHICH YOU MAY DIRECTLY OR INDIRECTLY SUFFER IN CONNECTION WITH YOUR USE OF OUR SERVICES AND SOFTWARE, INCLUDING LOSS OR DAMAGE SUFFERED BY DIRECTLY OR INDIRECTLY RELYING ON ANY INFORMATION APPEARING ON OUR WEBSITE OR THE SOFTWARE.

TO THE MAXIMUM EXTENT PERMITTED BY LAW, ANY CONDITION OR WARRANTY WHICH WOULD OTHERWISE BE IMPLIED INTO THESE TERMS AND CONDITIONS IS HEREBY EXCLUDED. WHERE LEGISLATION IMPLIES ANY CONDITION OR WARRANTY, AND THAT LEGISLATION PROHIBITS US FROM EXCLUDING OR MODIFYING THE APPLICATION OF, OR OUR LIABILITY UNDER, ANY SUCH CONDITION OR WARRANTY, THAT CONDITION OR WARRANTY WILL BE DEEMED INCLUDED BUT OUR LIABILITY WILL BE LIMITED FOR A BREACH OF THAT CONDITION OR WARRANTY TO ONE OR MORE OF THE FOLLOWING: (A) IF THE BREACH RELATES TO GOODS, (I) THE REPLACEMENT OF THE GOODS OR THE SUPPLY OF EQUIVALENT GOODS, (II) THE REPAIR OF SUCH GOODS, (III) THE PAYMENT OF THE COST OF REPLACING THE GOODS OR OF ACQUIRING EQUIVALENT GOODS OR (IV) THE PAYMENT OF THE COST OF HAVING THE GOODS REPAIRED; AND (B) IF THE BREACH RELATES TO SERVICES, (I) THE SUPPLYING OF THE SERVICES AGAIN OR (II) THE PAYMENT OF THE COST OF HAVING THE SERVICES SUPPLIED AGAIN.

9. Exception to disclaimer

This disclaimer set out in these terms and conditions does not attempt or purport to exclude liability arising under statute if, and to the extent, such liability cannot be lawfully excluded.

10. Specific warnings

You must ensure that your access to and use of our Services, and any Content that you access via our Services, is not illegal or prohibited by laws which apply to you.

You may incur additional cost in connection with your use of our Software and our Services, such as additional data charges from your mobile phone carrier or internet service provider. You are solely responsible for all such costs.

A failure to comply with these Terms of Service may result in termination of your user account.

If we become aware of Content with child sexual abuse imagery, we will report it to the appropriate authorities and delete the accounts of those involved with the distribution. You must take your own precautions to ensure that the processes that you employ for accessing our Services do not expose you to risk of viruses, malicious computer code or other forms of interference which may

damage your own computer system. For the removal of doubt, we do not accept responsibility for any interference or damage to your own computer system which arises in connection with your accessing of our Website, the Software, or any third party social network. You also acknowledge that the Software may not be error-free.

11. Indemnity

You release and indemnify us, our servants and agents against all actions, claims and demands (including the cost of defending or settling any action, claim or demand) which may be instituted against us arising out of a breach by you of these Terms of Service or arising as a result of your negligent or wilful misconduct in connection with the downloading or sharing of Content pursuant to the Terms of Service, including without limitiation against all actions, clause and demands (including the cost of defending or settling any action, claim or demand) relating to infringement of intellectual property rights.

Our facilitation of access to your third party social network accounts should not be construed as an endorsement, approval or recommendation by us of the owners or operators of those third party social networks, or of any information, graphics, materials, products or services referred to or contained on those third party social networks, unless and to the extent stipulated to the contrary.

In respect of any claim between the parties under or in connection with this agreement, the parties agree that to the maximum extent permitted by law, the operations of Part 4 of the Civil Liability Act 2002 (NSW) and of any laws having a similar effect in the Commonwealth and other States and Territories of Australia are excluded and have no application or effect insofar as any of them would apportion liability to us which would not have been so apportioned but for such laws.

12. Termination of access

Your access to the Services, including the Software, may be terminated at any time by us without notice. Our disclaimer will nevertheless survive any such termination.

13. Governing law

These terms and conditions are governed by the laws in force in New South Wales, Australia. You agree to submit to the exclusive jurisdiction of the courts of that jurisdiction.

14. General

If any of these terms and conditions are held to be invalid, unenforceable or illegal for any reason, the remaining terms and conditions shall nevertheless continue in full force.

A.2 Privacy Policy

Last updated 15 May 2014.

In this Supplemental Privacy Policy, "NICTA" refers to National ICT Australia Limited (ABN 62 102 206 173).

NICTA's standard Privacy Policy is available for review at www.nicta.com.au/privacy ("Standard Privacy Policy"). It provides you with general information about how your "personal information" may be collected, accessed, used, stored, disclosed or otherwise handled by NICTA.

This Supplemental Privacy Policy incorporates and supplements our Standard Privacy Policy. It provides you with information about additional information handling practices of NICTA that are specific to the Yalut social networking service (the "Yalut Service").

In the event of an inconsistency between this Supplemental Privacy Policy and our Standard Privacy Policy, this Supplemental Privacy Policy will take precedence in relation to your access to and use of the Yalut Service.

How we collect and use information relating to the Yalut Service

We may collect your personal information to provide, administer, and continually improve the Yalut Service, including personal information about your use of the Yalut Service.

You are required to register for a user name and password in order to use the Yalut Service. However, if you do not wish to directly disclose your identity, you may choose to use a pseudonym as your user name.

We will use and disclose the information we collect in accordance with applicable law and our Standard Privacy Policy. For example, we may use or disclose such information:

- to protect ourselves from liability, such as to defend against a legal claim; or
- to respond to a court order or other legal process; or
- to investigate, prevent or assist in the investigation or prevention of a suspected breach of our Terms of Service, including any suspected unlawful activity; or
- for our research purposes.
  Our research purposes may include, for example, investigating and developing efficient content delivery methods, publishing details of such research outcomes in journal articles or other academic papers, and preparing and discussing Yalut Service usage patterns at academic conferences. We may also share

anonymised usage information with researchers located in Australia or other countries who conduct similar research.

We currently do not plan to use the information we collect for advertising or direct marketing purposes.

Collection of essential usage information

We will track your usage of the Yalut Service. The following are some examples of the kinds of information that we may collect and associate with your Yalut Service user name:

- Your Internet Protocol (IP) address.
- A unique alphanumeric value (called a "hash value") that we calculate based on a media access control address (MAC address) or other unique identifier relating to your mobile phone or any other device which you use to access the Yalut Service. A hash value that we will calculate based on content that you share with other users of the Yalut Service (but we will not ourselves store or be recipients of such content).
- Meta-data about the content you share, including dates/times, file names and sizes.
- If you use the Yalut Service to login to a third party social networking service such as Facebook, Google+ or Twitter, such as for the purpose of sending a "share notification", we will also store a hash value that we will calculate based on your username and/or user ID for the third party service.

Collection of optional usage information

We would also like to collect certain additional usage information, but only with your opt-in consent.

When you register, you will be asked to consent to the automated, periodic, and ongoing collection of certain additional information about your use of the Yalut Service, including:

- crash log information (which may help us to rectify bugs); and
- information about your content transfers, including file names, file sizes, transfer dates and expiry times, and details about content seeding, pre-fetching and re-sharing.

If you do not give your consent to collect this additional usage information, you will still be able to fully access and use the Yalut Service.

However, if you do provide your consent, this information may assist us to optimize content sharing strategies and algorithms, improve integration with third party

social networks, understand the importance and popularity of ephemeral content sharing, and improve bandwidth consumption, and conduct and further our research.

If you do provide your consent, your continued participation will always be completely voluntary – you may disable the collection of this additional information by yourself at any time.

Please also be aware that the collection, if any, of such additional information will increase your WiFi data usage (which may attract usage fees from your other service providers depending on your WiFi data allowances). See our FAQ for further information.

Our policy toward obtaining consents

We do not wish to collect non-essential information about you without first obtaining your opt-in consent. If we become aware that a person does not have the capacity to consent, we will take steps to terminate that person's Yalut Service registration.

As Australia's Privacy Act does not specify an age after which individuals can make their own privacy decisions, our current policy is to presume that any individual aged 16 or over has the capacity to consent to the collection of their personal information unless we have reason to believe otherwise. If you become aware that a person under your supervision has registered to use the Yalut Service, and that person is under 16 or does not otherwise have the capacity to consent, please contact us using the details provided.

Deleting your account or information associated with your account

If you cease using the Yalut Service, and would like us to permanently delete your account, please contact us using the details provided. After receiving your request, we will take reasonable steps to destroy the information or to ensure that it is de-identified, although we may (for example) retain a record to indicate that your user name is not available for re-use by any other user.

Use of third party social networking services

If you use the Yalut Service to login to a third party social networking service such as Facebook, Google+ or Twitter, for the purpose of sending a "share notification", your use of that social networking service will be subject to the data practices of the third party service provider.

How to contact us

For any matter relating to this Supplemental Privacy Policy or your personal infor-

mation, including if you wish to complain about a breach of the Australian Privacy Principles, please contact:
Privacy Officer
Postal Address: Locked Bag 9013 Alexandria NSW 1435 Australia
Phone: +61 2 9209 4755
Email: privacy@nicta.com

We will respond to you, and we will try to resolve any complaints, as soon as possible.

A.3 Notification of Copyright Infringements

We have designated a representative and implemented other procedures to receive and act on notices of claimed infringements of copyright from owners and agents of owners of copyright material.

Notifications of copyright infringement must be written and signed, and should contain the following information:

1. Your contact details.
Please provide your name, address, telephone number, fax number and email address.

2. Description of copyright material.
Please provide a description that is sufficient to enable us to identify the copyright material in respect of which the infringement is claimed.

3. Location of copyright material.
Please provide sufficient information for us to locate the copyright material in respect of which infringement is claimed. For example, the IP address of a user who you allege has engaged in peer-to-peer copying of your copyright material.

4. If you are the owner of the copyright owner, please include the following statements:
"I am owner of the copyright in the copyright material, being copyright material residing on your system or network. I believe, in good faith, that the storage of the specified copyright material on your system or network is not authorised by me or a licensee, or the Copyright Act 1968, and is therefore an infringement of the copyright in that material. I have taken reasonable steps to ensure that the information and statements in this notice are accurate."

5. If you are the agent of the copyright owner, please include the following statements:
"I am the agent of the owner of the copyright in the copyright material, being copyright material residing on your system or network. I believe, in good faith, that the storage of the specified copyright material on your system or network

is not authorised by the copyright owner or a licensee of the copyright owner, or the Copyright Act 1968, and is therefore an infringement of the copyright in that material. I have taken reasonable steps to ensure that the information and statements in this notice are accurate."

If we receive such a notice alleging copyright infringement, our policy is to remove or disable access to the infringing material. If we cannot remove or disable access to infringing materials using technological measures, or in the case of repeated infringements or other appropriate circumstances, we will prevent access to the material by suspending use of any relevant user account that we can identify. We will also send a copy of your notice of claimed infringement to such user account, so that the user can get in touch with you to resolve your complaint.

If you wish to notify us of a claimed infringement of copyright, please send your signed notice our designated representative using the details below:

Attention: Copyright Manager
National ICT Australia Limited
Locked Bag 9013
Alexandria NSW 1435
Australia
copyright.manager@nicta.com.au

If you are providing your notice by email, you can sign your notice by providing a scanned physical signature or a valid electronic signature.