國立清華大學電機資訊學院資訊工程研究所
碩士論文
Department of Computer Science
College of Electrical Engineering and Computer Science
National Tsing Hua University
Master Thesis

挑戰性網路下針對行動裝置的新聞影片配送

Distribution of News Videos to Mobile Devices over Challenged Networks

王書挺
Shu-Ting Wang

指導教授：徐正炘 博士
Advisor: Cheng-Hsin Hsu, Ph.D.

中華民國 104 年 06 月
June, 2015

國立清華大學
資訊工程研究所

碩士論文

挑戰性網路下針對行動裝置的新聞影片配送

王書挺 撰

104
06

中文摘要

本論文針對高速通訊網路下非對稱的頻寬和資源問題，提出了一個在有限資源下的非對稱冗餘消除演算法 (RCARE) ，利用多餘的下載頻寬和接收端的資源，以加速上傳的數據傳輸。該系統可以部署於客戶端或代理伺服器上。RCARE 與現有的非對稱演算法不同，它使用更加靈活的匹配機制來識別冗餘資料，並使用一個傳送端的暫存器吸收過高的下載流量。和現有的冗餘消除演算法相比，它提供了一個可根據資源與效能調整的傳送端暫存器。我們從多個伺服器和校園網路記錄了真實的流量資料，並利用這些資料評估 RCARE 的效能。由我們的模擬結果顯示， RCARE 可比目前的非對稱式通訊演算法達到更高的上傳增益，以及更低的下載流量。我們也為有限資源的傳送端設計了動態調整演算法。此演算法可根據目前的樣本資料，預測並分配資源給目前的數據流，以達到最大的上傳增益。與平均分配資源的基準演算法相比較，動態調整演算法提高了高達 87% 的上傳增益。在前 10% 的實驗結果中 (以最佳的上傳增益排序) ， RCARE 平均達到了高達 40.5% 的上傳增益。

Abstract

Mobile devices are getting increasingly popular all over the world, including developing countries, where mobile users rarely have the Internet access. In this paper, we propose a Challenged Content Delivery Network (CCDN) to opportunistically distribute news reports to mobile users with intermittent Internet access. In particular, we formulate an optimization problem to compute the distribution plans for individual mobile users, so as to maximize the overall user experience under various resource constraints. Our formulation jointly considers the characteristics of news reports, mobile users, and intermittent networks. We present a distribution planning algorithm based on the multidimensional knapsack problem, and we develop several online heuristics to adapt to the system and network dynamics. We conduct extensive trace-driven simulations to evaluate our proposed CCDN, which demonstrate that our algorithm: (i) outperforms the baseline algorithms by 55% to 10 times in terms of user experience, (ii) achieves higher system efficiency than the baseline algorithm–by 37% to 20 times, and (iii) terminates in 12 minutes for a medium-size network of 150 users. We envision that our CCDN will allow news providers to reach out to more mobile users, and mobile users to watch news reports without always-on Internet access.

Contents

List of Figures

Chapter 1

Introduction

1.1 Introduction

All over the world, the speed of urbanization largely exceeds the deployment of network infrastructure. Although increasingly more people own modern mobile devices, such as smartphones and tablets, they may not have always-on Internet access. In particular, people living in developing cities suffer from weak (or non-existing) network infrastructure, while people living in developed cities face over-crowded and expensive mobile Internet access. For example, it is reported that, in India [**?**], poorly educated people who don't even have a SIM card but just use their phones as music players." Even if we include the well-developed cities like Cairo, Mumbai, and Shanghai, only 15%, 21%, and 30% of mobile users in Africa, India, and China have cellular dataplan [7–9]. That is, many mobile users have no access to online multimedia content, such as news reports, notification messages, targeted advertisements, movie trailers, and TV shows. Such digital divide may occur due to other reasons, e.g., in 2014, crowds in Taiwanese Student Movement adopt mobile apps for communications due to overloaded cellular infrastructure, while protesters in Hong Kong use FireChat [5] to avoid ubiquitous government censorship. Efficient content distribution in these situations remains very challenging and urgently needs a creative solution.

Let's consider one day of Amy, who owns a smartphone but couldn't afford of the data plan. Amy lives on a farm without the Internet access, and thus Amy can not watch online news reports in the evenings, when she finally gets some leisure time. Online news providers, therefore, lose an opportunity to deliver news reports to Amy, and suffer from degraded impacts to the society. One way to cope with such limitation is to hire Content Delivery Networks (CDNs) to push news reports to the servers close to the mobile users. Unfortunately, traditional CDNs do not help Amy, because Amy has no Internet access at home. On the other hand, Amy's smartphone is not *isolated* all day long, as illustrated

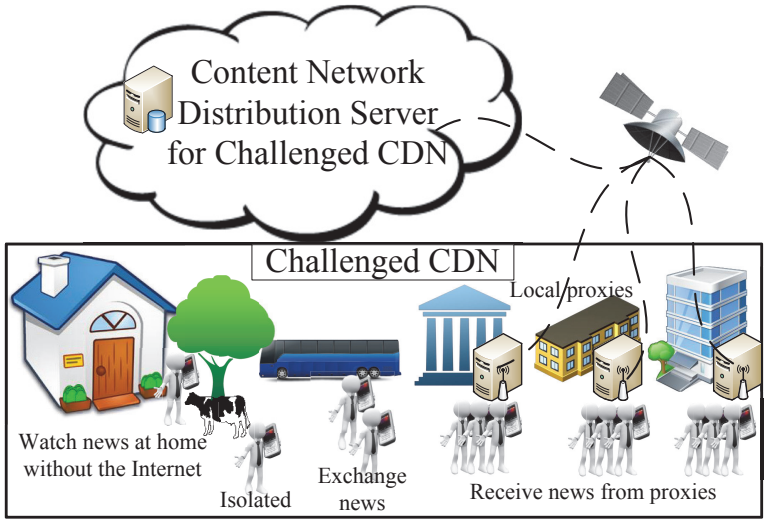


Figure 1.1: A usage scenario of the proposed CCDN.

in Fig. 1.1. In particular, Amy drops off and picks up her kids at the school, works at the city hall, and buys food at the market, where WiFi access is available. By deploying a *local proxy* next to each WiFi access point, the proxy can send some news reports to near-by mobile users, including Amy. Moreover, while Amy commutes, she runs into other citizens, who may previously have downloaded some news reports and can send the news reports to Amy as well. Hence, Amy, in fact, has *intermittent* network access via a non-traditional network called a *challenged network* [32], which suffers from high round-trip time, frequent link downs, long queueing delays, high dynamics, and scarce resources [19].

In this paper, we propose a Challenged Content Delivery Network (CCDN) to help news providers reach out to citizens living in rural areas without the Internet access, such as Amy. The proposed CCDN leverages Amy's *contacts* with local proxies and other mobile users, so as to distribute news reports to Amy. Therefore, Amy can enjoy the news reports in the evenings, although she does not have the Internet access at home. The crux of the proposed CCDN is to create a *distribution plan*, in order to intelligently distribute news reports: (i) at the best time, (ii) to the right mobile users, and (iii) at the highest possible quality. The best time refers to the contact with the best channel condition, which in turn results in faster data transfer and lower energy consumption. The right mobile users are the mobile users who are likely to watch a given news report, otherwise the transfer energy is simply wasted. The highest possible quality is measured as the average user experience across all the watched news reports among all mobile users. Take Amy as an example, a good distribution plan may: (i) instruct her to get news reports from the local proxy at the city hall where the number of WiFi users is lower, (ii)

suggest her to get entertainment news since Amy loves movies, and (iii) recommend her to get high-resolution videos for entertainment news reports, but only articles for other news reports. Computing the best distribution plan is extremely challenging due to the complex nature of the news reports, mobile users, and intermittent networks in CCDNs. For example, a news report may include several representations, such as article (including titles, descriptions, and images), audio, and layered video [37]; mobile users have diverse commute trajectories and news interests; and cellular networks are vulnerable to fading, shadowing, and interference that may result in disconnections. Throughout this paper, we focus on the distribution planning problem, but we also cover other crucial system's aspects of the CCDNs.

One way to solve the distribution planning problem is to compute the plan on each mobile device, which however is suboptimal by nature because each mobile device only has limited local view and suffers from insufficient computation and storage resources. A better way to solve the problem is to compute the distribution plan on a *distribution server* in the content network of a news provider, as illustrated in Fig. 1.1. More specifically, the distribution server periodically collects historical traces of news reports, mobile users, and intermittent networks from all the local proxies, where the local proxies gather the traces from mobile users. With the historical traces, the distribution server computes the optimal distribution plan once a day (different frequencies are also possible) for individual mobile users, based on the global view on the CCDN. Each distribution server manages a set of local proxies in a geographic region, as mobile users only interact with local proxies and mobile users in proximity. We allow the mobile users to specify their energy and storage budgets to prevent depleted batteries and full disks. We envision that the news providers would deploy low-cost local proxies in the field to boost the penetration rate of their news, while mobile users will help each other to break the digital divide.

This paper makes the following contributions:

- We propose the Challenged Content Delivery Networks (CCDNs) to distribute news reports to mobile users with intermittent Internet access, and increase the impacts of news providers. Prior challenged network studies [32, 34] often consider short messages, such as hazard and criminal alarms.

- We rigorously formulate and solve the distribution planning problem, which is the core optimization problem in CCDNs. We consider detailed characteristics of news reports, mobile users, and intermittent networks, while prior studies only solve the problem from a single aspect, such as networking [18, 26].

- We conduct extensive simulations using real datasets of news reports, mobile user trajectories, and user interests to quantify the performance of our proposed CCDNs

and algorithms. The simulation results show the merits of our CCDNs, e.g., our algorithm: (i) outperforms the baseline algorithms by 55% to 10 times in terms of user experience, (ii) outperforms the baseline algorithms by 37% to 20 times in terms of system efficiency, and (iii) terminates in 12 mins for a network with 150 users.

Chapter 2

Related Work

2.1 Opportunistic Networks

The solutions of opportunistic networks can be categorized based on assumptions regarding specific applications and environments. Either naive flooding [41, 42] or controlled flooding [22] are inefficient forwarding protocols under minimal assumptions. Epidemic forwarding [42] disseminates messages like spreading a disease, which is the most basic flooding strategy. Spray-and-wait [41] first spreads many copies of a message on several mobile devices. These mobile devices carry the message and forward it to the destination. Controlled flooding [22] considers two metrics: the total number of messages sent by nodes in the network and the total time that a node wants to a message until the destination node receives the message. The flooding is controlled by simple probabilistic heuristics, Time-to-live (TTL), passive drop of forwarding packets, and beacon intervals.

More efficient forwarding techniques require more sophisticated assumptions. The device mobility [13,27], the controlled mobility of some nodes [12,47], and contact histories [**?**, 29] are also used for more efficient opportunistic networks. Message ferrying [47] leverages a set of special mobile nodes with non-random, controlled mobility to deliver and communicate messages. Message ferrying With improves the efficiency of message delivery and energy consumption based on the knowledge of ferry routes and predictable mobility. Considering single node mobility, or device mobility in our usage scenario, Leguay et al. [27] propose a position-based routing mechanism for similar mobility pattern in high-dimensional Euclidean space. Cheng et al. [13] propose a geographic routing solution which exploits vehicular mobility and navigation systems, and it outperforms greedy based solutions in terms of packet delivery, such as Greedy Perimeter Stateless Routing and Greedy Perimeter Coordinator Routing. MV routing [12] is based on the observed mobility pattern and ensure robust message delivery for nodes on different geographic locations.

Capturing intrinsic behavior on social networks, researchers are able to design social-aware forwarding algorithms [16, 23, 33] that considers ranking or centrality information of mobile devices. Daly et al. [16] exploit the exchange of betweenness centrality metrics and local similarity information on social networks for the routing. Mtibaa et al. [33] propose PeopleRank with the inspiration from PageRank. PeopleRank exploit the most popular nodes in the social network to forward message, since they are mostly likely to contact with other nodes. Hui et al. [23] design a social-aware forwarding algorithm that considers communities and centrality of mobile devices. Similarly, Gao et al. [20] formulate multicast relay selection problem as a knapsack problem using social networks. Different from these studies, our CCDN systematically distributes news reports with multiple representations to mobile users who rarely have the Internet access.

Delay-tolerant networking, an architecture of opportunistic networks implementation, has been studied in the literature [4, 6, 19]. For example, Fall [19] proposes a network architecture composing of resource-constrained mobile devices, which is essentially an overlay network above the transport layer. However, These studies focus on shorter messages, whereas our CCDN system supports multi-layer multimedia content.

2.2 Caching

Several studies consider caching Web contents on mobile devices. Qian et al. [36] reveal the inefficient cache implementation using HTTP libraries and mobile browsers. Their results show that the redundant transfers contribute about 20% of the total traffic and 7% of the radio energy consumption. Zhang et al. [46] provide a real implementation of the Web caching service at the OS kernel level. Their implementation saves 42% traffic under real user browsing behavior and doubles the Web accessing speed. Lymberopoulos et al. [30] download Web contents before user requests using a machine learning model, which predicts future Web accesses and caches 60% of the URL for about 80% users. These studies are designed for the traditional Internet.

Cooperative caching improves performance of Web applications in opportunistic networks. The technique proposed in [45] caches data in a set of easily accessible mobile devices and exercises the tradeoff between data accessibility and caching overhead. Wang et al. [44] leverage the popularity ranks to support cooperative caching under opportunistic networks via Bluetooth or WiFi. Besides, a cooperative caching system [24] is proposed for interactive Web applications over challenged networks. While our distribution planning algorithm jointly considers news reports, mobile users, and intermittent networks to compute the best distribution plan, the prior studies only consider some aspects, e.g., [24] does not take the characteristics of news reports into consideration.

2.3 Intermittent Clouds

Intermittent clouds are the cloud with some cloud servers that are not always connected to the Internet, which are similar to our local proxy servers. Shi et al. [39] design and implement a system that enables computation offloading from mobile devices to intermittent cloud servers. CirrusCloud [38] investigate the best time for mobile devices to connect to a central cloud, and offload computations to the cloud through other encountered mobile devices. IC-Cloud [15] focuses on lightweight prediction for mobile devices, such as connectivity prediction, execution prediction, and usage prediction to make offloading decisions. We consider the media dissemination problem, which is complementary to the computation offloading problem.

Chapter 3

News Video Distribution System

3.1 System Architecture

3.1.1 Proposed Network Model: Challenged CDNs

The system architecture is presented in Fig. 3.1. The proposed CCDN is an extension of CDN, and consists of two parts: (i) a content network that distributes news reports over the Internet to the users all over the world, and (ii) challenged networks where mobile users rarely have Internet access. Mobile users in challenged networks do not have direct access to the content network. Therefore, we propose to set up local proxies at popular locations, such as coffee shops, city halls, schools, and markets, to distribute the news reports to mobile users whenever they have contacts with local proxies. These mobile users, therefore, may enjoy the news reports even when Internet access is unavailable. The contacts among mobile users are also leveraged for exchanging news reports to speed up the distribution. The distribution planning problem is solved on the distribution server in the content network. The resulting distribution plans are sent to mobile users via local proxies. The distribution planning algorithm runs periodically, say once a day, and the frequency can be adjusted by system administrators.

3.1.2 News Report Model

We consider the online news reports from news providers, such as CNN, BBC, CBC, and Al Jazeera. Each news report has different representations that are suitable under different circumstances. For example, for mobile users with a few short contacts, distributing news videos to them is unrealistic. In contrast, a well-connected tablet computer user may allocate more energy and disk budgets for high-resolution news videos. In particular, each news report can be rendered in the following representations: articles, audio, and layered video. Each representation contains one or multiple *layers*, e.g., an audio representation

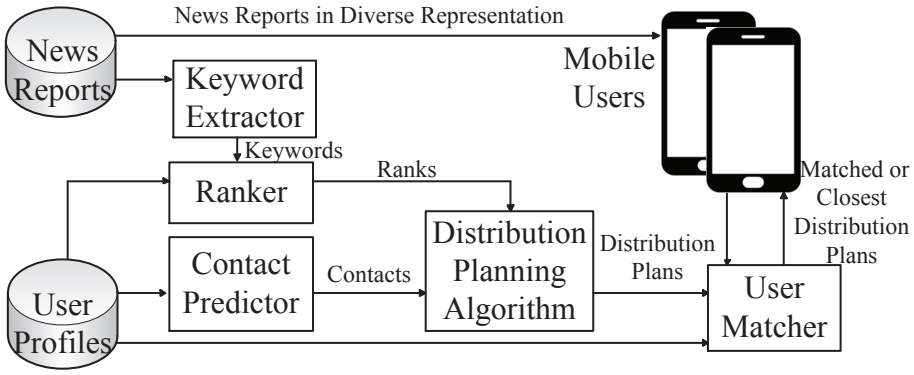


Figure 3.1: The system architecture of our proposed CCDN.

contains the article and audio layers, and the high-resolution video representation contains all layers. That is, there exists a linear dependency among the layers, and the sizes of different layers are quite diverse. Last, mobile users perceive different levels of experience when watching news reports in different representations. The user experience improvement typically follows a saturated function with more layers, e.g., moving from nothing to an article is a huge jump, while moving from medium- to high-resolution is less dramatic.

The distribution planning problem has to determine which news report to send to which mobile user. Each news report has one or multiple topics, which may be used to match against mobile user interests. We take a step further and extract keywords from individual news articles. Upon extracting the keywords from all news reports, we compare the keywords and individual user interests to get a personalized rank on news reports, which is then mapped into viewing probability following the viewing statistics. The viewing probability of each news report and mobile user is an input to the distribution planning algorithm. The keyword extraction is done in the *keyword extractor* and the ranking is done in the *ranker*.

3.1.3 Mobile User Model

We keep track of the user profile, which consists of various user-specific historical data. First, each mobile user is interested in different keywords, which may be manually specified by mobile users, or derived via collecting all the keywords of recently watched news reports. The most current user interests are used to rank future news reports. Second, each mobile user moves along different trajectories, which are essentially series of timestamped locations. Such trajectories may be generated using GPS readers on smartphones. Once we get the trajectories of all mobile devices, we compute the user contacts based on the WiFi range, where each contact includes the duration and network condition. That is,

using the trajectories of all mobile users, we compute the contact sequences of individual mobile users. Our distribution planning algorithm takes the user contacts as inputs to generate the distribution plan. However, the user contacts may be different each day, and we rely on the *contact predictor* to estimate the future user contact. Fortunately, human mobility is highly predictable and 85% of time a mobile user stays at his/her top 5 favorite locations [21, 40].

Our distribution planning algorithm computes the distribution plans for all known users. The distribution server then pushes the distribution plan and user profile of a mobile user to the local proxies that are on the user's contacts. The mobile user fetches the distribution plan when getting into the proximity of any of these local proxies. Some mobile users may fail to find their own distribution plans, because they are new to the CCDN or dramatically changed their daily trajectories. We use the *user matcher* to locate the plan for the closest, ideally exact, mobile user, so that even if the exact distribution plan is not available at a local proxy, the most suitable one can be sent to the mobile user. Specifically, the user matcher computes the similarity between the subject user's profile and all known user profiles, to find the exact or closet mobile user.

3.2 Distribution Planning Problem and Solution

3.2.1 Notations

We consider a CCDN that delivers news reports to U mobile users. The mobile users communicate with the local proxies, and we let S denote the number of local proxies. Let N be the total number of news reports and L be the number of layers of each news report. The layer l ($1 \leq l \leq L$) is only decodable if all layers $l' \leq l$ have also been received. We define the delivery *unit* as a layer of a news report, which is the basic allocation unit. In particular, unit $i = nL + l$ is a unique identifier pointing to layer l of news report n. We let ρ_i ($1 \leq i \leq NL$) be the user experience improvement when receiving unit i in addition to all the layers beneath it. We let b_i be the size of the unit i, and $\psi_{u,n}$ be the probability for mobile user u to watch news report n. We let $\bar{\psi}$ be the minimal viewing probability: a mobile user would not request a news report from another mobile user who is unlikely to watch it.

We let T be the number of time slots that are considered in our formulation, and $t = 0$ be the starting time slot. We assume that mobile users' trajectories are given, i.e., each user's location at every time slot is provided by some localization techniques. With the mobile users' trajectories and local proxies' locations, the sequence of contacts during time $[0, T]$ is determined. We let C be the maximal number of contacts across all mobile

devices. Without loss of generality, we assume each contact happens between two parties, which can be either mobile users or local proxies. If a mobile user concurrently runs into $\dot{u}$ other parties, it equally divides the contact into $\dot{u}$ disjoint contacts along the time domain, where each contact has exactly two parties. That is, simple time-division multiplex is done to avoid interference due to concurrent transfers. We let $p_{u,c}$ be the other party of contact c ($1 \leq c \leq C$) of user u ($1 \leq u \leq U$), where $1 \leq p_{u,c} \leq U + S$. When $p_{u,c} \leq U$, it points to mobile user $p_{u,c}$, while $p_{u,c} \geq U$, it points to local proxy $p_{u,c} - U$. Last, we write the duration of contact c of user u as $\kappa_{u,c}$.

Combining the contacts with trajectories, we can estimate the throughput and energy consumption of each contact. In particular, we write the receiving throughput of contact c of user u as $r_{u,c}$, the transmitting per-unit energy consumption as $\hat{e}_{u,c}$ and the receiving per-unit energy consumption as $\check{e}_{u,c}$. Last, we use q_u and d_u to represent the energy and disk budgets of mobile user u during $t \in [1, T]$. q_u and d_u are user-specified parameters.

3.2.2 Problem Formulation

With the notations developed above, we formulate a distribution planning problem to determine which units to request during which contacts in order to maximize the overall user experience. The decision variables are the distribution plans: $x_{u,n,l,c}$, where $1 \leq u \leq U$, $1 \leq n \leq N$, $1 \leq l \leq L$, and $1 \leq c \leq C$. $x_{u,n,l,c} = 1$ if mobile user u requests for unit $nL + l$ during contact c; $x_{u,n,l,c} = 0$ otherwise. The formulation is written as:

$$\max \sum_{u=1}^{U} \sum_{n=1}^{N} \sum_{l=1}^{L} \sum_{c=1}^{C} x_{u,n,l,c}\, \rho_{nL+l}\, \psi_{u,n} \tag{3.1a}$$

$$st: \psi_{p_{u',c'},n'} \geq \bar{\psi} x_{u',n',l',c'} \; \forall 1 \leq u' \leq U, 1 \leq n' \leq N, 1 \leq l' \leq L, 1 \leq c' \leq C \tag{3.1b}$$

$$\sum_{n=1}^{N} \sum_{l=1}^{L} b_{nL+l} x_{u',n,l,c'} \leq r_{u',c'} \kappa_{u',c'} \forall 1 \leq c' \leq C, 1 \leq u' \leq U \tag{3.1c}$$

$$\sum_{c=1}^{C} x_{u',n',l',c} \geq \sum_{c=1}^{C} x_{u',n',l'',c} \; \forall 1 \leq u' \leq U, 1 \leq n' \leq N, 1 \leq l' < l'' \leq L \tag{3.1d}$$

$$\sum_{n=1}^{N} \sum_{l=1}^{L} \sum_{c=1}^{C} b_{nL+l} x_{u',n,l,c} \leq d_{u'} \; \forall 1 \leq u' \leq U \tag{3.1e}$$

$$\sum_{u=1}^{U} \sum_{n=1}^{N} \sum_{l=1}^{L} \sum_{c=1}^{C} \{1 - \min[1, \max(p_{u,c}, u') - \min(p_{u,c}, u')]\}\, b_{nL+l}$$
$$x_{u,n,l,c} \hat{e}_{u',c} + \sum_{n=1}^{N} \sum_{l=1}^{L} \sum_{c=1}^{C} b_{nL+l} x_{u',n,l,c} \check{e}_{u',c} \leq q_{u'} \; \forall 1 \leq u' \leq U \tag{3.1f}$$

$$\sum_{c=1}^{C} x_{u',n',l',c} \leq 1 \; \forall 1 \leq u' \leq U, 1 \leq n' \leq N, l \leq l' \leq L \tag{3.1g}$$

$$x_{u,n,l,c} \in \{0, 1\} \; \forall u, n, l, c. \tag{3.1h}$$

The objective function in Eq. (3.1a) maximizes the expected overall user experience across all mobile users. Eq. (3.1b) makes sure that mobile users never request a news report from someone who is unlikely to watch it. Eq. (3.1c) ensures that each contact duration is long enough to complete the planned unit transfer under the given transmission throughput. Eq. (3.1d) captures the layer dependency, i.e., higher layer l'' is only decodable when all its lower layers l' are received. Eq. (3.1e) makes sure that the total size of planned news videos does not exceed the user's disk budget. Eq. (3.1f) sums up the transmitting and receiving energy and ensures that it does not exceed the energy budget. The first part of this constraint accounts for the transmission energy, and the term in the brace is 1 iff $p_{u,c} = u'$, which means that mobile user u plans to receive news videos from

mobile user u'. The second part accounts for the receiving energy. Eq. (3.1g) ensures that each user does not receive the same unit multiple times, which results in wasted resources.

3.2.3 Distribution Planning Algorithm

Our problem formulation in Eq. (3.1) is a 0-1 Integer Linear Programming (ILP) problem. A closer look reveals that our formulation can be converted into a 0-1 Multidimensional Knapsack Problem (MKP). The MKP problem is written as:

$$\max \sum_{j=1}^{J} r_j x_j \tag{3.2a}$$

$$st : \sum_{j=1}^{J} w_{k,j} x_j \leq y_k \; \forall k = 1, 2, \ldots, K \tag{3.2b}$$

$$x_j \in \{0, 1\} \; \forall j = 1, 2, \ldots J. \tag{3.2c}$$

The boolean decision variable x_j indicates whether we want to put object j in our knapsack, while r_j represents the profit of having object j. There are K constraints, where each constraint k has a resource limit y_k, and each object j consumes a given amount of resource $w_{k,j}$. The MKP problem strives to pick a subset of objects so that the total profit is maximized, while none of the constraints are violated. The transformation from our distribution planning problem to an MKP problem is done as follows. We first let $J = UNLC$, and $r_j = \rho_{nL+l}\psi_{u,n}$, where $j = uNLC + nLC + lC + c \; \forall u, n, l, c$, which turns our objective function into MKP's objective function. Next, for each constraint of our formulation, we move all the decision variables (along with their coefficients) to the left hand side, and other constants to the right hand side. We let $w_{k,j}$ be the corresponding coefficient, and y_k be the constant. We can do this because our constraints (Eqs. (3.1b)–(3.1g)) are all in the form of linear combinations of decision variables. The transformation yields a corresponding MKP problem.

The MKP problem is solved by several exact and heuristic algorithms proposed in the literature [43]. Finding the best algorithm for MKP problems is not our focus, and thus we leverage on a commercial solver [1]. That is, we solve our distribution planning problem in three steps: (i) transforming it into an MKP problem, (ii) solving the MKP problem, and (iii) converting the MKP solution into a distribution plan. Our initial tests show that the running time is quite short for medium-size problems. For larger problems, we may set a *time limit* for our problem to trade optimality for running time. While a time limit may lead to non-optimal distribution plans, we design several online adaptation strategies (Sec. 3.3.1) to compensate this situation.

3.3 Practical Concerns

3.3.1 Adaptive Communication Strategies

The distribution server is connected to the local proxies over potentially unreliable or intermittent links, such as low Earth orbit satellites. However, the distribution planning algorithm is infrequently invoked, and there is plenty of time for the distribution server and local proxies to synchronize up in terms of news reports, user profiles, and distribution plans. In contrast, the contacts in challenged networks are very short, and thus require adaptive communication strategies. For a mobile device having a contact with a local proxy, the mobile device first checks if it has a valid distribution plan. If not, the mobile device requests for a new distribution plan. Upon having a distribution plan, the mobile device checks if it should request for any outstanding units from the local proxy based on the distribution plan. Upon the units planned for this contact are all finished, the units planned for other contacts are requested. Once all the units on the distribution plan are downloaded, the local proxy generates on-the-fly recommendations by matching mobile user interests and the news report keywords. The recommendations might be first presented to mobile users for validation and augmentation. If the contact terminates, the mobile device aborts the ongoing transfer and waits for the next contact.

For a mobile device having a contact with another one, the mobile devices first exchange the outstanding units that are planned to request from each other. Once these units are done, the mobile devices exchange the outstanding units that are in their plans. When there are no outstanding units in the plans, the mobile devices show the available units to mobile users, and allow them to select the news reports to request. If the storage and energy budgets are used up, the mobile devices abort and wait for the next contacts.

When requesting multiple units, the order is crucial, e.g., requesting a higher layer earlier is vulnerable to broken layer dependency. Moreover, it is preferred to devote resources to those units that result in the higher user experience improvement normalized to unit size. More precisely, when a contact is available, each mobile device computes ρ_u/b_u of the next outstanding layer of each news report. The mobile device requests the unit u^* with the highest ρ_{u^*}/b_{u^*}. This heuristic is repeated until the contact is over or resources (e.g., energy and disk) are used up.

3.3.2 Machine Learning on Distribution Servers

Several machine learning algorithms are adopted in our distribution server, which have more resources and better security, compared to local proxies. Running them on mobile devices takes long time and consumes excessive energy, while running them on local

Table 3.1: Sample News Reports

Report	1	2	3	4	5	6	7	8	9	10
Duration (s)	145	1500	1060	132	104	134	144	145	146	195
Article (KB)	4.2	1.0	1.6	1.8	2.9	1.8	0.3	2.1	0.5	1.6
Audio (MB)	0.74	10.4	7.2	1.35	1.09	1.17	1.05	0.74	0.86	1.58
Low-Res Video (MB)	4.6	47	36	4.4	3.4	4.7	4.8	4.6	4.9	6.3
Med-Res Video (MB)	17	178	124	16	13	16	17	17	17	24
High-Res Video (MB)	31	316	221	29	23	29	31	31	31	42

proxies is risky because the proxies are not trustworthy, e.g., they can be stolen from these public areas. Our keyword extractor derives keywords using topical model techniques, such as latent semantic analysis [17] and latent Dirichlet allocation [10]. The ranker then computes user centric ranking using, e.g., RankingSVM [25] and LambdaMART [11]. Last, the contact predictor estimates the future contact locations using either trajectory pattern [31], social networks [14], or frequency-based approach. Upon the contact locations are determined, the contact durations can be predicted using the techniques proposed in, e.g., Li et al. [28]. The cited machine learning techniques are by no mean an exhausted list, as many techniques are constantly proposed by the research community. In our simulator, we implement and evaluate a few machine learning algorithms; see Sec. **??** for details.

3.3.3 Segmenting Video Layers

We have analyzed the layer sizes of typical Al Jazeera news reports. The article and audio layers are merely 1+ MB at most, but the video layers are quite large[1]. Therefore, we define a maximal segment size Z, which is a system parameter. For unit with size b larger than Z, we divide the unit into $\lceil b/Z \rceil$ segments, where the first $\lceil b/Z \rceil - 1$ segments are in the size of Z. The segments in the same units are assigned different time-based priorities, e.g., the news room may manually tag the first one-third of a report as the most important part and the middle one-third as the least important part. Then, we define the dependency among the segments by their priorities. By doing so, we ensure that the more important segments are distributed earlier. For brevity, the user experience improvement of a unit is equally split among all segments in that unit; more comprehensive approaches are also possible. The segmentations are done after the distribution plans are computed, because incorporating the concept of segments in the distribution planning problem increases the

[1]Strictly speaking, the Al Jazeera videos are not encoded as layered videos [37], as layered codecs are not widely available yet. We use their sizes to approximate the layered video sizes.

problem size, which leads to staggering computational overhead.

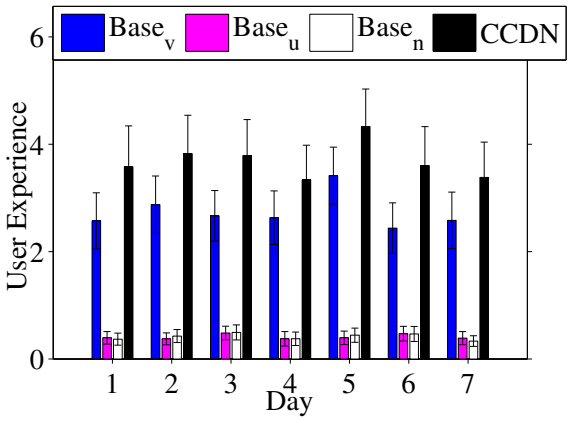
Chapter 4

Trace-driven Simulations

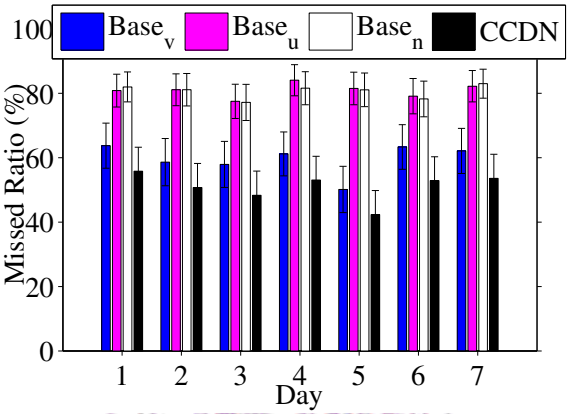
4.1 Datasets

We employ three datasets: (i) user contacts, (ii) video reports, and (iii) user interests to drive our simulator. We adopt two user contact datasets: GeoLife [48] and SIGCOMM [35]. The GeoLife dataset contains the GPS trajectories of about 180 mobile users over 4 years. Using the locations of mobile users, we estimate the contacts (including duration and throughput) by assuming a WiFi range of 100 m and considering various modulation and coding schemes. The SIGCOMM dataset contains 76 mobile users' BlueTooth contacts for slightly over 3 days. We increase the throughput by 10 times, as the Bluetooth has much lower data rate than WiFi. Since the dataset does not include trajectories, for any contacts that did not actually happen, we use the average throughput of both parties to approximate it.

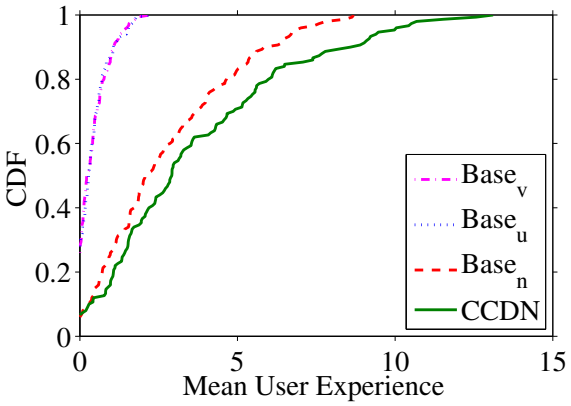
For the news video dataset, we collect 27 news reports from Al Jazeera in mid-July 2014, and divide each news report into five layers: article, audio, low-, medium-, and high-resolution videos. The sizes of each layer is calculated. The shortest and longest news reports last for 1.73 and 26 minutes, respectively. We adopt Latent Dirichlet allocation [10] to extract keywords from the articles. We get at least 5 keywords per news report. The resulting keywords are used by the ranker and user matcher. Last, we derive the user interests by leveraging the user queries in the LETOR [2] dataset. In particular, we randomly pick a user query, and take the keywords in it to mimic a user's interests. The keywords in LETOR dataset are different from our Al Jazeera dataset, and we map the keywords using their orders of appearance.


Base_v
Base_u
Base_n
CCDN
User Experience
Day
0
2
4
6
1
2
3
4
5
6
7

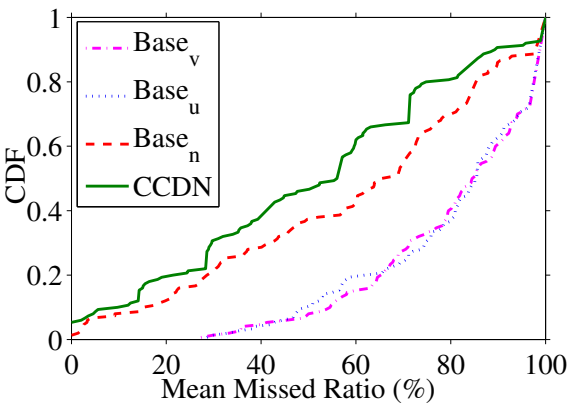
(a)


Base_v
Base_u
Base_n
CCDN
Missed Ratio (%)
Day
0
20
40
60
80
100
1
2
3
4
5
6
7

(b)


CDF
Mean User Experience
Base_v
Base_u
Base_n
CCDN
0
0.2
0.4
0.6
0.8
1
0
5
10
15

(c)


CDF
Mean Missed Ratio (%)
Base_v
Base_u
Base_n
CCDN
0
0.2
0.4
0.6
0.8
1
0
20
40
60
80
100

(d)

4.2 Simulator Implementation

We have implemented a detailed CCDN simulator using a mixture of Python, C, and CPLEX to evaluate the performance of our system. Our focus is on the proposed distribution planning algorithm and online adaptive strategies. The distribution planning algorithm is executed once a day in our simulations. Upon the distribution plans are computed, our simulator carries out the simulations following the ground truth given in the real datasets. For comparison, we have also designed three baseline algorithms: (i) the experience-driven baseline (Base_v) that greedily requests the unit with the highest user experience improvement in each contact, (ii) the centrality-driven baseline (Base_u) that greedily sends the units to the mobile device with the highest number of contacts, and (iii) the throughput-driven baseline (Base_n) that greedily sends the units to the mobile device with the best channel condition. The baseline algorithms represent the prior arts, and each of them considers an aspect of CCDNs: news reports, mobile users, or intermittent networks. In contrast, our algorithm jointly considers all three aspects for more intelligent distribution plans.

We repeat each simulation scenario with the four algorithms. We consider the following performance metrics:

- **User experience**: the summation of the user experience of all the watched news reports.
- **Energy consumption**: the energy consumption of mobile devices.
- **System efficiency**: the ratio of user experience and energy consumption.
- **Used disk space**: the amount of used disk space.
- **Missed ratio**: the fraction of unavailable news reports among all the user demanded ones.
- **Viewing ratio**: the fraction of viewed news reports among all the downloaded ones.

We report the average performance with 95% confidence intervals whenever applicable.

Last, we have also implemented some machine learning algorithms. The contact predictor keeps track of the historical contacts, and predicts the future contacts based on frequencies. The ranker employs LamdaMART [11] from RankLib [3] to compute the ranks based on news report keywords and user interests, and the ranks are mapped to view probabilities proportional to the discounted cumulative gains. The user matcher adopts the cosine similarity to compute the distance between any two user profiles.

4.3 Simulation Scenarios

We conduct two sets of simulations to investigate the performance of: (i) the proposed CCDNs and (ii) the machine learning algorithms. The first set include 7- and 3-day simulations using GeoLife and SIGCOMM datasets, respectively. The GeoLife dataset is very sparse: (i) only 3.33% of user-day GPS trajectories are non-empty and (ii) the dataset spans over the greater Beijing area. Therefore, we focus on a 92 km^2 downtown area, and create a 7-day trace by choosing the most active day of each mobile user. We remove the mobile users who never get into the downtown area, which yields a trace with 150 mobile users. Last, 30 local proxies are deployed in the crowded locations. For the SIGCOMM dataset, we promote 10 mobile users with the most contacts to be the local proxies. In both cases, we randomly pick 15 news reports every day from the Al Jazeera dataset. We consider the disk budget in $\{60, 125, 250, 500\}$ MB, and the energy budget in $\{500, 1000, 2000, 4000\}$ J. By default, we set disk (energy) budget to be 250 MB (2000 J). Moreover, WiFi power consumption is 600 mW, and the maximal segment size is 25 MB. Last, we set the user experience improvements of layers 1–3 to be 0.26, and the medium- and high-resolution videos to be 0.12 and 0.10. The second set of simulations evaluates the implications of the implemented machine learning algorithms: the contact predictor and ranker. We conduct the simulations with and without the machine learning algorithms, and study their performance.

4.4 Results

The proposed CCDN algorithm improves the service quality. Fig. 4.1 reports the service quality of our CCDN in user experience and missed ratio. We first present sample results from GeoLife. Fig. 4.1(a) shows that our algorithm outperforms all three baseline algorithms. The gaps of and Base_u and Base_n to our algorithm are large, because these two algorithms do not take the characteristics of news reports into considerations. In contrast, Base_v considers the user experience improvement of each unit when requesting for news reports, and thus achieves better user experience: only about 30% worse than our algorithm. Fig. 4.1(b) gives the missed ratio, which shows that the mobile users miss about 80% of news reports if Base_u or Base_n are employed. In contrast, our algorithm leads to only about 50% missed ratio in general, while Base_v is at most 10% worse than our algorithm. Next, for each mobile user, we compute the mean user experience and missed ratio over 7 days. We plot the results in Figs. 4.1(c) and 4.1(d), which clearly show that our algorithm results in higher user experience and lower missed ratio. Last, we report sample results from the SIGCOMM dataset in Figs. 4.1(e) and 4.1(f). These two

figures reveal that our algorithm outperforms Base_u and Base_n in terms of service quality. However, Base_v achieves similar service quality than our CCDN algorithm. A closer look indicates that the SIGCOMM dataset is well connected, and thus all the mobile users receive at least medium-resolution news reports for all news reports. This in turn leads to virtually no optimization room to our proposed CCDN algorithm.

The proposed CCDN algorithm is resource efficient. We report the resource efficiency of our proposed CCDN in Fig. 4.2. Fig. 4.2(a) presents the daily system efficiency of our CCDN, which is the ratio between user experience and energy consumption. This figure clearly shows that our CCDN algorithm outperforms all three baseline algorithms. In particular, more than 10 times of improvement is achieved compared to Base_u and Base_n, while up to 25% of improvement is achieved compared to Base_v. This indicates that our CCDN algorithm achieves good service quality in an energy-efficient manner. Fig. 4.2(b) gives the daily used disk space, which shows that our algorithm uses at most 1/3 of disk space, compared to all three baseline algorithms. This shows that our algorithm is conservative in terms of disk usage as well. Next, we plot the viewing ratio in Fig. 4.2(c), which reveals that Base_u and Base_n suffer from less than half of the viewing ratio achieved by our algorithm, while Base_v has a viewing ratio 5% lower than our algorithm. Therefore, our CCDN algorithm wastes the least downloaded news reports. We also compute the per-user mean performance results in Figs. 4.2(d)–4.2(f), which also demonstrate that our CCDN algorithm leads to higher system efficiency, less disk space usage, and higher viewing ratio.

Implications of diverse energy and disk budgets. Next, we vary the energy and disk budgets and compare the performance of our CCDN algorithm against the baseline algorithms in Fig. 4.3. Figs. 4.3(a) and 4.3(b) present the service quality under different energy budgets, which show that more energy budgets lead to higher user experience and lower missed ratio. We observe that, compared to the baseline algorithms, our CCDN algorithm enjoys the highest user experience boost (> 5 times) and missed ratio reduction (by half). Figs. 4.3(c) and 4.3(d) show the service quality under different disk budgets. We make two observations: (i) more disk budgets result in improved service quality for the three baseline algorithms, and (ii) the CCDN algorithm achieves roughly constant service quality under different disk budget. The latter observation indicates that our CCDN algorithm uses the disk budget in an efficient way, e.g., with only 60 MB disk budget, our algorithm achieves $< 50\%$ missed ratio.

Running time of our CCDN algorithm. In Fig. 4.4, we report the daily running time of: ranker, problem constructor, and ILP solver. The figures show that the ILP solver runs fast, but the ranker may take some time. Overall, the daily running times are 12 mins and 100 mins for GeoLife and SIGCOMM datasets. Since we solve the distribution planning

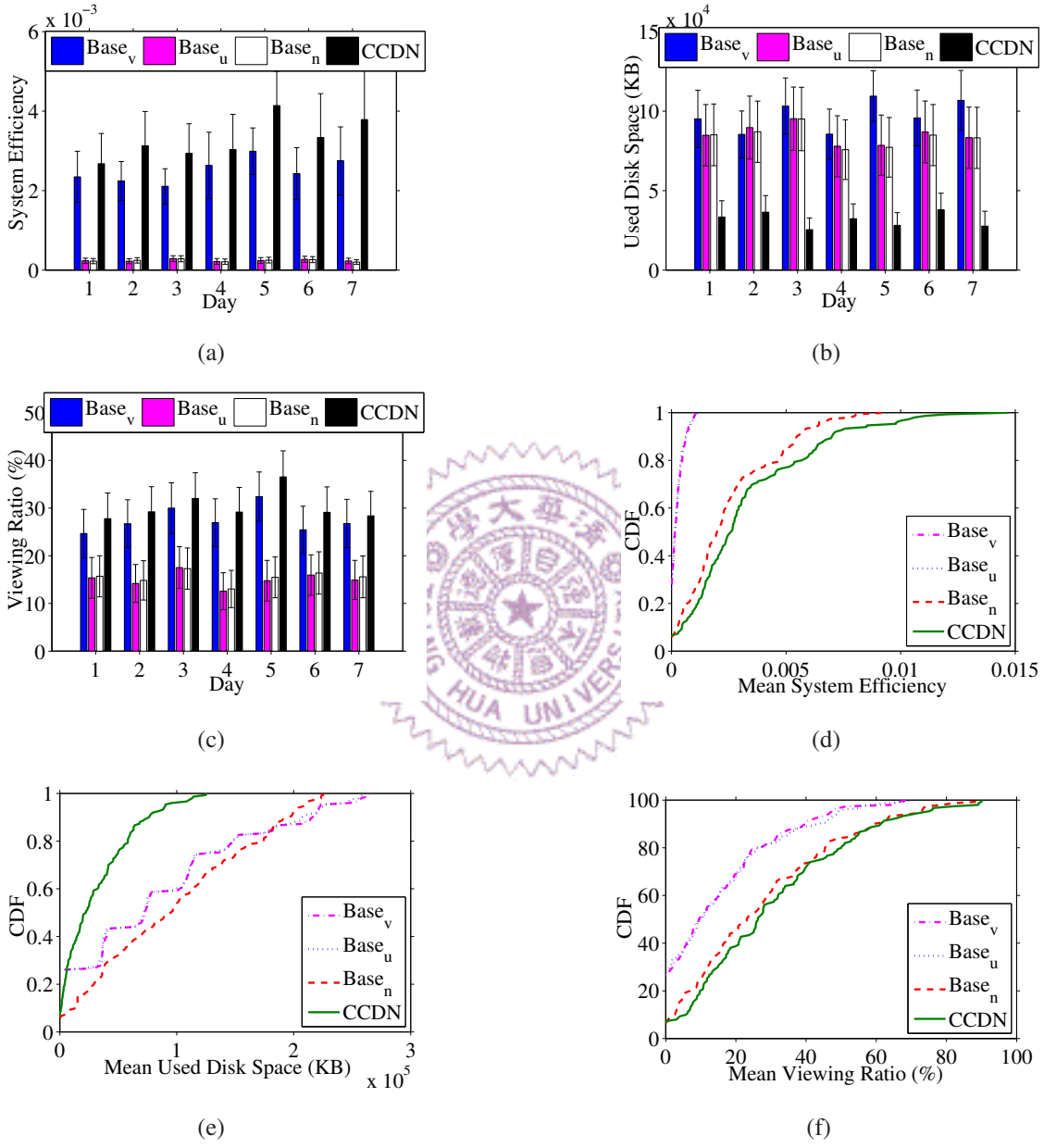


Figure 4.2: Resource efficiency of our CCDNs: (a) daily system efficiency, (b) daily used disk space, (c) daily viewing ratio, (d) system efficiency CDF, (e) used disk space CDF, and (f) viewing ratio CDF. Sample results from GeoLife dataset.

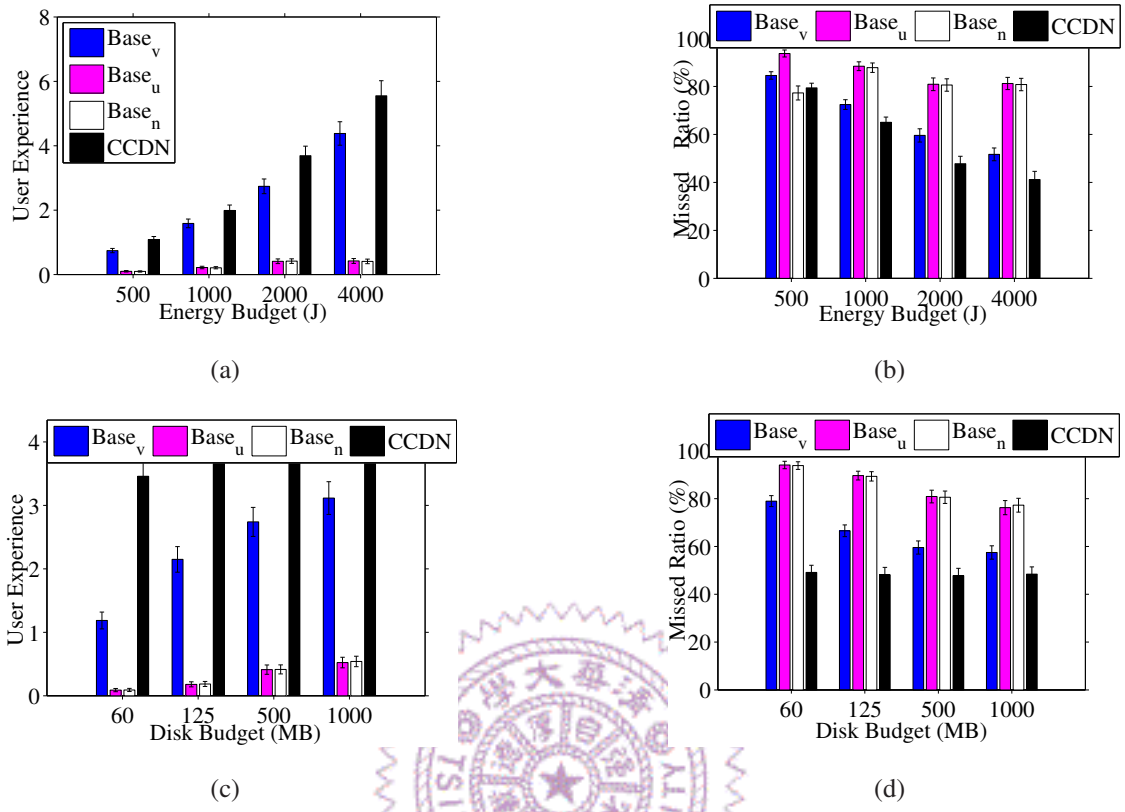


Figure 4.3: Implications of diverse budgets on the CCDNs: (a) energy budget on user experience, (b) energy budget on missed ratio, (c) disk budget on user experience, and (d) disk budget on missed ratio. Sample results from GeoLife dataset.

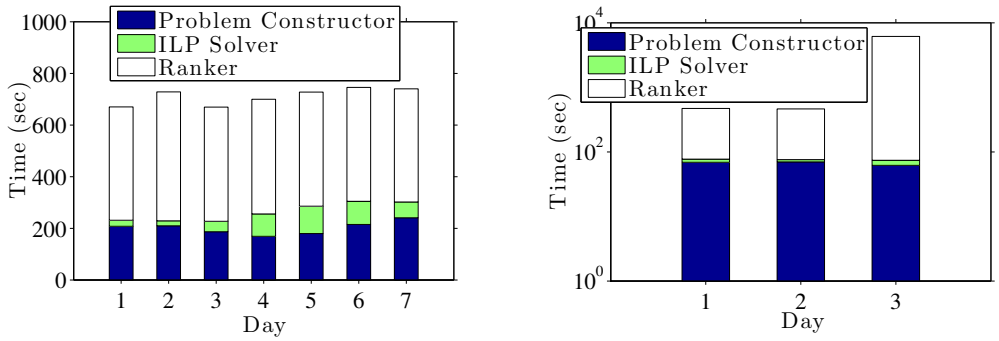
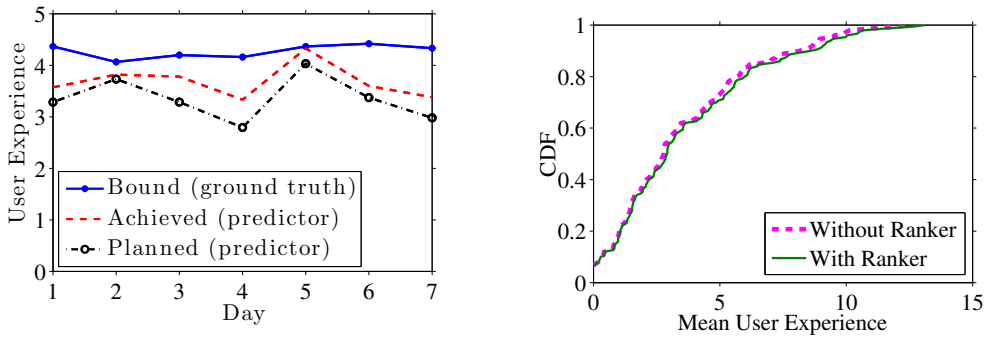


Figure 4.4: Daily running time: (a) GeoLife and (b) SIGCOMM datasets.



(a) (b)

Figure 4.5: Daily user experience CDF: (a) with and without the contact predictor and (b) with and without the ranker.

problem once a day, the running-time is reasonable. If time is a concern, a simpler ranker may be adopted at the expense of less accurate viewing probabilities.

The machine learning algorithms. We plot the daily user experience with and without the contact predictor in Fig. 4.5(a). The top line indicates the upper bound of the user experience, which is derived using the ground truth (without the contact predictor). The bottom line is the expected user experience with the contact predictor, which is computed using our CCDN algorithm. The middle line is the actual user experience, which is achieved by applying both the distribution plan and the adaptive communication strategies. This figure shows that our adaptive communication strategies effectively increase user experience to better approach the upper bound. Fig. 4.5(b) presents the user experience CDF with and without the ranker. This figure shows that our ranker slightly improves the user experience.

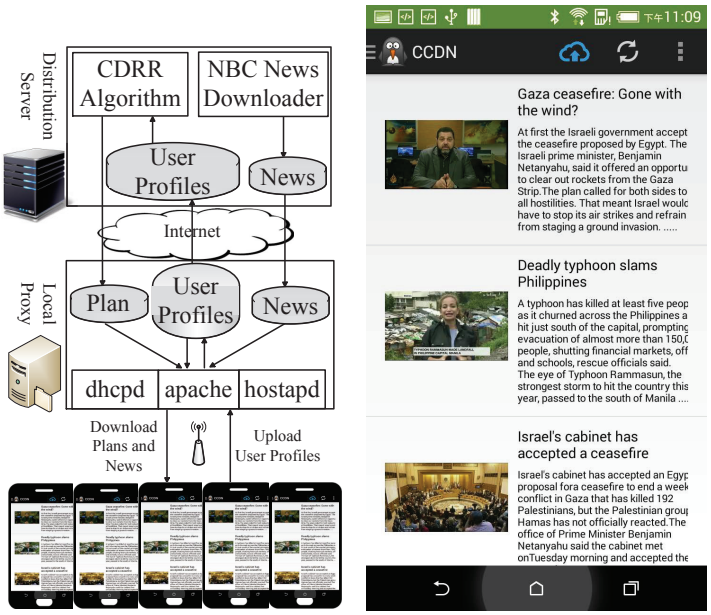
Chapter 5

System Prototype

We have implemented a proof-of-concept testbed of the proposed CCDN system shown in Fig. 5.1. The distribution server and local proxies are built on Linux boxes. We realize our proposed algorithm on the distribution sever, and `hostapd`, `dhcpd`, and `apache` daemons on local proxies. The local proxies are equipped with 802.11b wireless cards, and are connected to the distribution server via Internet. We implement a CCDN mobile app on Android as shown in Fig. 5.1(b), which follows the distribution plan to download multimedia content whenever it runs into local proxies. It also records timestamped events and sends them via local proxies to the distribution server as profiles. The distribution server analyzes the profiles for various inputs, such as contacts, and computes distribution plans. The *News Downloader* retrieves news reports from NBC at 5 a.m. everyday as a `cron` job. The computed plans and downloaded news reports are sent from distribution server to mobile users via local proxies.

We set up a distribution server and two local proxies on our campus, and install our mobile app on 5 Android phones. The distribution server and a local proxy are put in a lab room (in EECS building), and another local proxy is put in a kitchen (in another general purpose building). We set the client disk budget to be 2 GB (25% of the total disk) and energy budget to be 6000 J (20% of the total energy). Every morning, the distribution server sends the latest 300 news reports to local proxies. Each news report has 5 layers, including text, audio, and videos transcoded to three different resolutions using FFmpeg leading to 1500 units every day. Mobile users watch the news reports every evening and we clean up their disks at 5 a.m. everyday. For any user, we use his/her profile on day $a-1$ to predict his/her behavior on day a. We then use the predicted behaviors to compute the distribution plans and send the plans to mobile users when they run into local proxies.

Fig. 5.2 plots the number of remaining units of users' distribution plans, remaining disk budget, and remaining energy budget on a sample day. It reveals that: (i) the numbers of planed units are diverse, e.g., user_5 plans to download 1401 units and user_2 plans to



(a) (b)

Figure 5.1: CCDN prototype: (a) system components and (b) interface showing downloaded multimedia content.

download 1121 units, (ii) smartphones finish different percentages of their plans, e.g., user_3 achieves 50% and user_4 achieves $> 99\%$, and (iii) shorter contact duration leads to fewer downloaded units, e.g., user_3 only gets 502 units due to the shortest contact duration.

The current prototype system has some limitations: (i) smartphones cannot download units from peer smartphones, (ii) better machine learning algorithms shall be adopted to better leverage mobile user profiles, (iii) large video files shall be segmented into chunks to efficiently support download resumes, and (iv) the distribution server and local proxies may be in different geographic locations with diverse network conditions. Some of the limitations are addressed in Sec. 3.3, and we are continue enhancing the prototype system.

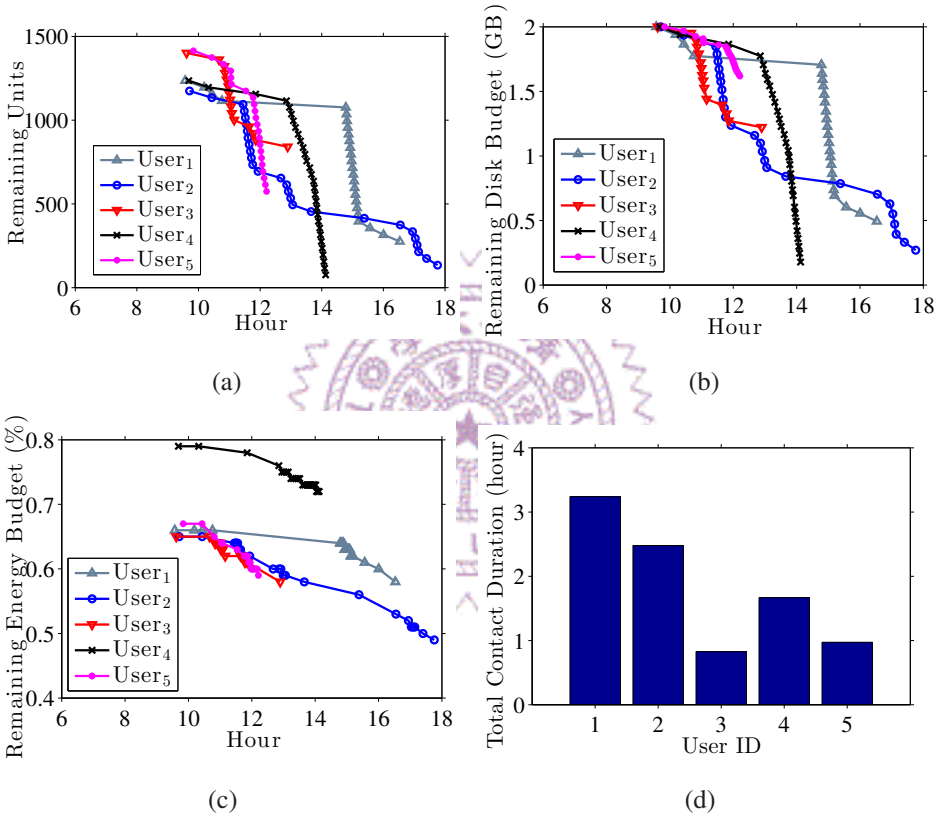


Figure 5.2: Sample results on March 30, 2015. The remaining: (a) number of units to be downloaded, (b) disk budget, and (c) energy budget; (d) per-user total contact duration.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this paper, we studied the problem of distributing news reports over challenged networks to mobile devices. We proposed a Challenged Content Delivery Network (CCDN), which carefully plans the distribution of news reports to mobile users. The crux of the proposed CCDN is the distribution planning algorithm, which intelligently distributes news reports: (i) at the best time, (ii) to the interested mobile users, and (iii) at the highest possible quality. We formulated this distribution planning problem into a mathematical optimization problem, and we solve the problem utilizing an optimal ILP solver. Furthermore, we developed several heuristics to adapt to system and network dynamics without recomputing the distribution plans. We conducted extensive simulations using real datasets. The simulation results indicate that our proposed CCDN algorithm results in: (i) better user experience, (ii) higher system efficiency, (iii) reduced disk usage, (iv) lower missed ratio, and (v) higher viewing ratio. This work can be extended in several directions. For example, we are developing and deploying CCDN prototype systems for developing countries to demonstrate its practicality and efficiency.

Bibliography

[1] IBM CPLEX optimizer. http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/.

[2] Letor 4.0 dataset. http://research.microsoft.com/en-us/um/beijing/projects/letor/letor4dataset.aspx, 2009.

[3] Ranklib. http://sourceforge.net/p/lemur/wiki/RankLib/, 2009.

[4] Dtn2. http://sourceforge.net/projects/dtn/files/DTN2/dtn-2.9.0/, 2012.

[5] FireChat shows the triumph of technology over repression. http://tinyurl.com/p8eblx7, 2012.

[6] Ibr-dtn. http://trac.ibr.cs.tu-bs.de/project-cm-2012-ibrdtn, 2012.

[7] How do we accelerate Internet access in Africa? http://ventureburn.com/2014/01/how-do-we-accelerate-internet-access-in-africa/, 2013.

[8] It's time to take a closer look at China's mobile industry. http://www.businessinsider.com/the-key-china-mobile-industry-statistics-2013-12?op=1, 2014.

[9] Social, digital and mobile in India 2014. http://wearesocial.net/blog/2014/07/social-digital-mobile-india-2014/, 2014.

[10] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3:993–1022, 2003.

[11] C. Burges. From ranknet to lambdarank to lambdamart: An overview. Technical report, Microsoft Research, 2010.

[12] B. Burns, O. Brock, and B. Levine. Mv routing and capacity building in disruption tolerant networks. In *Proc. of IEEE INFOCOM*, 2005.

[13] P. Cheng, K. Lee, M. Gerla, and J. Härri. GeoDTN+Nav: Geographic DTN routing with navigator prediction for urban vehicular environments. *Mobile Networks and Applications*, 15(1):61–82, 2010.

[14] E. Cho, S. M. A., and J. Leskovec. Friendship and mobility: User movement in location-based social networks. In *Proc. of ACM KDD*, 2011.

[15] S. Cong, P. Pranesh, N. Kangqi, Y. Juyuan, A. Mostafa, N. Mayur, and Z. Ellen. Ic-cloud: Computation offloading to an intermittently-connected cloud. Technical report, January 2013.

[16] E. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *Proc. of ACM MobiHoc*, 2007.

[17] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[18] N. Do, C. Hsu, and N. Venkatasubramanian. Hybcast: Rich content dissemination in hybrid cellular and 802.11 ad hoc networks. In *Proc. of IEEE SRDS*, 2012.

[19] K. Fall. A delay-tolerant network architecture for challenged Internets. In *Proc. of ACM SIGCOMM*, 2003.

[20] W. Gao, Q. Li, B. Zhao, and G. Cao. Multicasting in delay tolerant networks: A social network perspective. In *Proc. of ACM MobiHoc*, 2009.

[21] M. Gonzalez, C. Hidalgo, and A. Barabasi. Understanding individual human mobility patterns. *Nature*, 453:779–782, 2008.

[22] K. Harras and K. Almeroth. Controlled flooding in disconnected sparse mobile networks. *Wireless Communication Mobile Computing*, 9(1):21–33, 2009.

[23] P. Hui, J. Crowcroft, and E. Yoneki. Bubble rap: Social-based forwarding in delay-tolerant networks. *IEEE Transactions on Mobile Computing*, 10(11):1576–1589, 2011.

[24] S. Isaacman and M. Martonosi. Low-infrastructure methods to improve Internet access for mobile users in emerging regions. In *Proc. of ACM WWW*, 2011.

[25] T. Joachims. Optimizing search engines using clickthrough data. In *Proc. of ACM KDD*, 2002.

[26] L. Keller, A. Lec, B. Cici, H. Seferoglu, C.Fragouli, and A. Markopoulou. Microcast: Cooperative video streaming on smartphones. In *Proc. of ACM MobiSys*, 2012.

[27] J. Leguay, T. Friedman, and V. Conan. DTN routing in a mobility pattern space. In *Proc. of ACM SIGCOMM WDTN*, 2005.

[28] Z. Li, B. Ding, J. Han, R. Kays, and P. Nye. Mining periodic behaviors for moving objects. In *Proc. of ACM KDD*, 2010.

[29] A. Lindgren, A. Doria, and O. Schelén. Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE mobile computing and communications review*, 7(3):19–20, 2003.

[30] D. Lymberopoulos, O. Riva, K. Strauss, A. Mittal, and A. Ntoulas. Pocketweb: Instant web browsing for mobile devices. In *Proc. of ACM ASPLOS*, 2012.

[31] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti. Wherenext: A location predictor on trajectory pattern mining. In *Proc. of ACM KDD*, 2009.

[32] V. Mota, F. Cunha, D. Macedo, J. Nogueira, and A. Loureiro. Protocols, mobility models and tools in opportunistic networks: A survey. *Computer Communications*, 48(0):5 – 19, 2014.

[33] A. Mtibaa, M. May, C. Diot, and M. Ammar. Peoplerank: Social opportunistic forwarding. In *Proc. of IEEE INFOCOM*, 2010.

[34] H. Ntareme, M. Zennaro, and B. Pehrson. Delay tolerant network on smartphones: Applications for communication challenged areas. In *Proc. of ExtremeCom*, 2011.

[35] A. Pietilainen, E. Oliver, J. Lebrun, G. Varghese, and C. Diot. MobiClique: Middleware for mobile social networking. In *Proc. of ACM WOSN*, 2009.

[36] F. Qian, K. Quah, J. Huang, J. Erman, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck. Web caching on smartphones: Ideal vs. reality. In *Proc. of ACM MobiSys*, 2012.

[37] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the H.264/AVC standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(9):1103–1120, 2007.

[38] C. Shi, M. Ammar, E. Zegura, and M. Naik. Computing in cirrus clouds: The challenge of intermittent connectivity. In *Proc. of ACM SIGCOMM MCC*, 2012.

[39] C. Shi, V. Lakafosis, M. Ammar, and E. Zegura. Serendipity: Enabling remote computing among intermittently connected mobile devices. In *Proc. of ACM MobiHoc*, 2012.

[40] C. Song, Z. Qu, N. Blumm, and A. Barabási. Limits of predictability in human mobility. *Science*, 327:1018–1021, 2010.

[41] T. Spyropoulos, K. Psounis, and C. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proc. of ACM SIGCOMM WDTN*, 2005.

[42] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical report, Duke University, 2000.

[43] M. Varnamkhasti. Overview of the algorithms for solving the multidimensional knapsack problems. *Advanced Studies in Biology*, 4(1):37–47, 2012.

[44] T. Wang, P. Hui, S. Kulkarni, and P. Cuff. Cooperative caching based on file popularity ranking in delay tolerant networks. In *Proc. of ExtremeCom*, 2012.

[45] G. Wei, C. Guohong, A. Iyengar, and M. Srivatsa. Cooperative caching for efficient data access in disruption tolerant networks. *IEEE Transactions on Mobile Computing*, 13(3):611–625, 2014.

[46] Y. Zhang, C. Tan, and L. Qun. Cachekeeper: A system-wide web caching service for smartphones. In *Proc. of ACM UbiComp*, 2013.

[47] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Proc. of ACM MobiHoc*, 2004.

[48] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W. Ma. Understanding mobility based on gps data. In *Proc. of the UbiComp*, 2008.