Collaborative Content Distribution in Mobile Networks with Caching and D2D-Assistance

by

Haoru Xing

Bachelor of Engineering,
Beijing University of Posts and Telecommunications, China, 2017

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Computer Science

In the Graduate Academic Unit of Computer Science

This thesis is accepted by the

Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

November, 2019

Abstract

While the mobile networks are evolving to a content-centric paradigm, the surging traffic demands keep stressing the ever-increasing network capacities. In-network caching offers an effective means to alleviate the traffic pressure by saving bandwidth utilization and balancing traffic loads. In this thesis, we first introduced an integrated content distribution framework that integrates universal in-network caching and enables collaboration across domains. This framework takes advantage of appealing design principles in device-to-device (D2D) communications, mobile edge computing (MEC), network function virtualization (NFV), and software-defined networking (SDN).

Leveraging this framework, we studied the request screening problem, which aims to appropriately select the video requests offloaded to energy-efficient D2D communications. We redirected content requests to maximize the coverage of requests that can be fulfilled through D2D communications. Given the constraints of individual transmission and caching capacities, the number of available D2D channels, and information privacy with social-awareness, we can decouple the screening problem into two subproblems, i.e., the device caching and matching problem, and the D2D channel allocation problem. As we proved that both problems are NP-hard, we proposed social-aware heuristic algorithms that iteratively make the best offloading decision in each step. Simulation results show that the proposed algorithms perform fairly closely to optimal solutions in small-scale instances and outperform the reference schemes under various situations.

Then, we studied the request routing problem, which selects sources and redirects video streams appropriately to optimize in-network flows. We proposed a context-aware approach for request routing through the integrated edge-core. The results demonstrate that the proposed solution achieves significant performance gain over the reference schemes in exploiting network dynamics and user context to relieve network congestion.

Acknowledgements

Throughout the writing of this dissertation I have received a great deal of support and assistance. I would first like to thank my supervisor, Dr. Wei Song, whose expertise was invaluable in formulating the research topic and methodology in particular. The door to Professor Song's office was always open whenever I ran into a trouble spot or had a question about my research or writing. She consistently encouraged me to work on the research problems on my own, but steered me in the right direction whenever she thought I needed it.

I would also like to acknowledge Professor Huajie Zhang and Professor Paul Cook from the Faculty of Computer Science of the University of New Brunswick, and Professor Maryhelen Stevenson from the Department of Electrical and Computer Engineering of the University of New Brunswick, as the examining committee members of this thesis, and I am gratefully indebted to their very valuable comments on this thesis.

In addition, I would like to thank my families for their wise counsel and sympathetic ear. You are always there for me. Finally, there are my friends, who were of great support in deliberating over our problems and findings, as well as providing happy distraction to rest my mind outside of my research.

Table of Contents

Vita

List of Tables

List of Figures

List of Abbreviations

3GPP	The Third Generation Partnership Project
5G	The fifth generation mobile networks
AP	Access point
BBU	Baseband unit
BS	Base station
CCN	Content-centric network
CDN	Content delivery network
C-RAN	Cloud-based radio access network
D2D	Device-to-device
EPC	Evolved packet core
ILP	Integer linear program
IoT	Internet-of-Things
IXP	Internet exchange point
LP	Linear program
LTE	Long Term Evolution
MCP	Maximum coverage problem
MEC	Mobile edge computing
MIP	Mixed integer program
NFV	Network function virtualization
PDN	Packet data network

P-GW	Packet data network gateway
QoE	Quality of experience
QoS	Quality-of-service
RAN	Radio access network
RRH	Radio remote head
SDN	Software defined network
SNR	Signal-to-noise ratio
UE	User equipment
VCP	Vertex coloring problem

List of Symbols

$\mathcal{N}$	Set of all network nodes
$\mathcal{E}$	Set of in-network links
$\mathcal{V}$	Set of video clips
$\mathcal{U}$	Set of all UEs
$\mathcal{U}_j$	Set of UEs within BS j
$\mathcal{U}_t$	Set of D2D groups
$\mathcal{A}$	Set of arches in social relationship graph
$\mathcal{A}_c$	Set of arches in conflict graph for D2D channel allocation
$\mathcal{R}$	Set of all video requests
$\mathcal{R}'$	Set of requests unfulfilled by device caching
$\mathcal{R}_j$	Set of video requests to BS j
$\mathcal{W}$	Set of cellular users
m_i	Video requested by device i
$c(e)$	Capacity of in-network link $e \in \mathcal{E}$
B_i	Bandwidth of wireless link to device i
d_{ij}	Distance between device i and BS j
ψ_i	Achievable data rate at device i
Γ_i	Received SNR at device i
P_j	Transmit power of BS j
L_i	Log-distance path loss at device i

ζ_c	Coefficient of Zipf distribution for device caching
ζ_r	Coefficient of Zipf distribution for video requests
$\vartheta_{j,k}^{(c)}$	Probability of edge server j caching video k
$\vartheta_{i,k}^{(r)}$	Probability of device i requesting video k
l_{ij}	Content sharing availability between devices i and j
β_i	Individual budget of device i
ϖ	Collaboration distance
$\mathcal{E}_b$	Social relationship graph among devices
γ	Maximum number of caching sources allowed in a cell
τ	Duration of time interval
λ	Maximum link utilization
$\lambda*$	Target threshold for maximum link utilization
d_i^k	Actual demand rate for request (i, k)
q_i^k	Target transmission rate for request (i, k)
p_i^k	Playout time for request (i, k)
s_i^k	Data received at device i before current interval
T_i^k	Buffer time for request (i, k)
$\mathcal{P}$	Set of all in-network paths
$\mathcal{P}_i^k$	Set of available paths for request (i, k)
$x(P)$	Flow amount over path P

Chapter 1

Introduction

1.1 Motivation

Nowadays, mobile data traffic reaches almost 50% of traffic in mobile networks. Since the use of laptops, smartphones and many other new emerging entertainment applications is quickly increasing, mobile video streaming growth is expected to be dramatic. While the fifth-generation (5G) mobile networks are evolving to a content-centric paradigm, the surging traffic demands keep stressing the ever-increasing network capacities. The 5G mobile networks are expected to support a plethora of emerging services and applications, such as multimedia content distribution, augmented and virtual reality, and Internet-of-Things (IoT). Hence, the battle between ever-growing traffic demands and out-paced network capacities will continue and thus require more efficient solutions for content distribution.

Traditionally, various video sources, such as dedicated content distribution networks (CDNs) and cloud service points, can be deployed widely to expedite mobile access to video services. Fig. 1.1 depicts a typical use case of video delivery in the Long Term Evolution (LTE) systems of the Third Generation Partnership Project (3GPP). As seen from the dashed line, the content delivery needs to flow through the entire

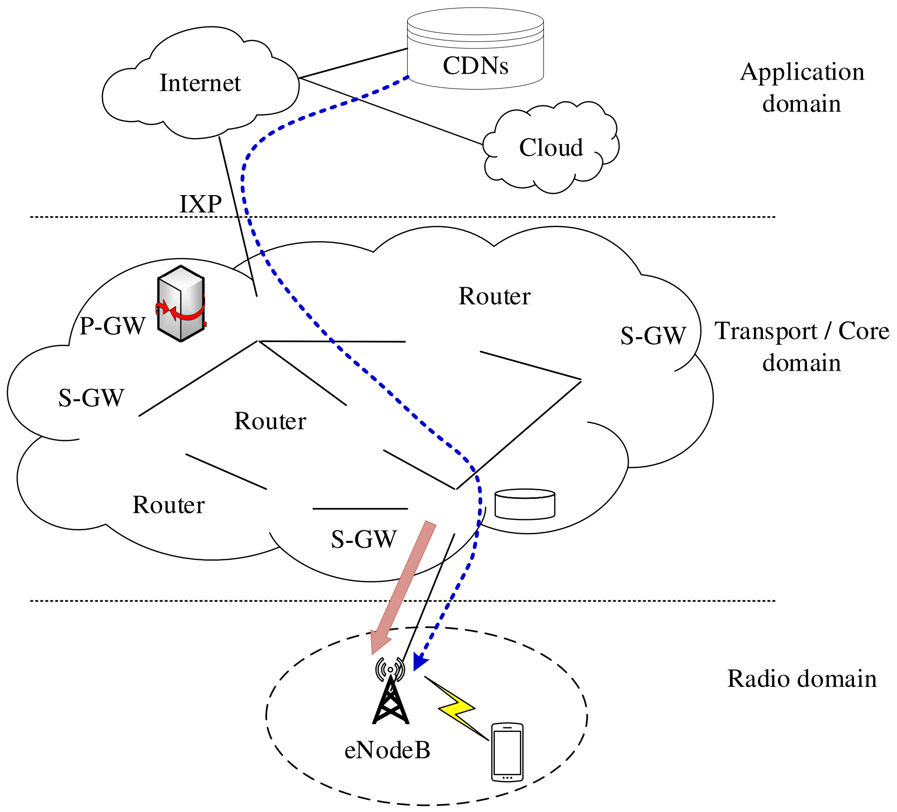


Fig. 1.1: Example of video content distribution through an LTE network.

hierarchical network infrastructure. In addition, due to data frame encapsulation, the mobile network is completely oblivious to traffic details and engages little control of the video flow for network optimization or quality-of-service (QoS) enhancement. To further reduce end-to-end latency and redundant network traffic, in-network caching has been proposed to place cache components, *e.g.*, between the mobile core (*i.e.*, transport/core domain) and the Internet at the Internet exchange point (IXP) or inside the mobile core network, which is also called Evolved Packet Core (EPC) in LTE systems at the packet data network (PDN) gateways (P-GWs) or the serving gateways (S-GWs). With built-in caches inside the mobile core network, the flow path of video delivery can be reduced significantly.

In recent years, there are many studies that consider moving data closer to end users toward the network edge. That is, content caching can be enabled inside the radio access networks (RANs) at base stations (BSs), namely, eNodeBs in LTE systems, and even in mobile devices. In particular, the rapidly developing device-to-device (D2D) communication technologies, such as LTE Direct and Wi-Fi Direct, can facilitate the sharing of the content cached in mobile devices among nearby users. D2D-assisted content distribution not only can offload traffic from the BSs, but also can exploit the close proximity to improve user-perceived quality [2, 3].

1.2 Research Objectives

In this thesis, we consider an integrated framework for video content distribution in mobile networks, which incorporates caching in the core, mobile edge and user devices, and enables collaboration among virtualized source points across domains. Based on this integrated framework, we focus on two research problems.

First, we study a request screening problem, which selects certain mobile devices to cache contents and serve other requests via D2D communications. As such, a portion of content requests can be fulfillled by local devices without going deep to the caching framework. Taking into account social-awareness, device locations, individual and overall resource constraints, this problem simultaneously addresses cache deployment in devices and device matching to fulfill content requests. A heuristic scheme is proposed due to the high complexity of large-scale instances.

Second, for the remaining requests unfulfilled by device caching, we address them by solving a request routing problem, which fetches contents from multiple collaborative caches in the core and edge. It needs to determine the best content sources and routing paths for the so-generated traffic flows. This idea is similar to smart routing studied in [4], but we further extend it to a more orchestrated framework with an end-to-end scope and accommodate more implementation constraints. With the assistance of the framework, our solution exploits user context, such as the wireless link capacity, video client's buffer status, and dynamic request information, in order to steer traffic and utilize diverse available paths.

1.2.1 Request Screening

In this work, video requests are initiated by mobile devices and first submitted to the associated BS, which acts as a transceiver connecting a number of other devices to one another and/or to a wider area. If two devices request the same video and fall within each others transmission range, it is more efficient for the BS to satisfy only one request and let the other request be fulfilled by D2D communications. Considering that some popular video is demanded by multiple neighboring users and thus can be served by D2D multicast, the BS needs to appropriately select the video requests offloaded to D2D communications. This is the *request screening problem*, which splits the requests into those served by local device caching or others to be fulfilled by in-network caching. The number of feasible D2D links in a cell should be restricted to allow tolerable interference to regular cellular users. Therefore, we can limit the maximum number of offloaded video requests in a single cell. The goal of the request screening problem is to maximize the number of requests fulfilled by local caching while satisfying the resource constraint on D2D links.

Towards this end, we first consider the impact of collaboration distance between devices. Given the limitation of D2D links, only request devices located in the transmission range of a source device can be fulfilled. Similarly, a source device can fulfill no more than a certain number of requests from other devices due to the stint of the device energy and electricity. Beyond these, social relationships among D2D users should be considered to exploit social-awareness and opportunistic contacts with user mobility [5].

Since mobile devices (*e.g.*, smart phones, smart vehicles and laptops) are used by people, it is necessary to consider and explore social behaviors of human users when solving the request screening problem. In order to protect private user information, contents cannot be sent between users who have no strong social relationships between each other. The request screening approach takes two main steps. First, the edge server selects some user devices to become seeds (also known as sources) and delivers contents to these seeds. Then other user devices can fetch contents from those selected seed devices through D2D communications. Since both the properties of the physical network and social relationships are important factors, a social-physical graph model needs to be built to combine the mobile network with the social network based on certain social network models or real datasets of social networks. The request screening problem is presented in detail in Chapter 4.

1.2.2 Request Routing

After screening all device requests, the remaining requests unfulfilled by local caching are further directed to edge servers. For each video request, the user context information is gathered at the controller and utilized for subsequent content delivery. While the edge servers deliver video content to the device, the received data are first buffered at the video client for an initial buffering period. This involves a short start-up delay in the order of a few seconds. Then, the video player at the client begins to fetch the buffered data for playout.

As external and internal content caches are virtualized as source servers, a requested video can be retrieved from one or more sources, and the so-generated flows can be routed over multiple paths toward the target device. Therefore, a *joint source redirection and flow routing problem* (termed the request routing problem in short) needs to be solved by the controller. Specifically, each video request can be fulfilled by more than one source points so the data flow is split into several fractions and delivered through different paths. The duty of the controller is to determine what sources should be used to fetch the data and make routing decisions for the corresponding video streams to optimize in-network flows.

On one hand, since more buffered video data achieve lower playout latency and smoothness, the controller aims to route more data of requested videos in each time period so that there are enough buffered data at the video client to guarantee the playout performance and the user experience. On the other hand, the capacity of all network links should be taken into account. Here, we use the maximum link utilization to measure the congestion in all network links so that the traffic congestion can be avoided by bounding the maximum link utilization. Based on the above requirements, we consider the request routing problem as a joint problem by taking both playout continuity and traffic balancing into account. The objective of this joint problem is to maximize the minimum buffer time among all active video sessions, while avoiding the traffic congestion in all network links, *i.e.*, minimizing the maximum link utilization. Chapter 5 provides analysis and the solution for the request routing problem.

1.3 Thesis Organization

This thesis is organized as follows. Chapter 2 introduces the background knowledge and reviews the related works. Chapter 3 describes the integrated content distribution framework with collaborative caching, then introduces models of request screening and request routing. Chapter 4 introduces two sub-problems of request screening, and presents a heuristic social-aware algorithm for request screening and evaluation results. Chapter 5 provides the formulation of the request routing problem and introduces a context-aware solution for multi-source request routing and gives experimental results. Chapter 6 concludes this thesis and provides the directions of future work.

Chapter 2

Background and Related Work

In this chapter, we introduce the background knowledge related to MEC, D2D communications, and mobile content distribution. Then the related research works on content distribution in mobile networks and content distribution with D2D-assistance are presented.

2.1 Mobile Edge Computing

Nowadays our daily lives have been flooded with diverse mobile applications, ranging from entertainment and education, to business, health, and online social platforms. As a consequence, mobile data traffic is expected to double continuously every year. Since thriving and emerging mobile services are becoming an integral part of daily and social life of all mobile device users, users are definitely expecting higher Quality of Experience (QoE) when enjoying these new services provided by different service providers. To meet the fast increasing demands from mobile users, the mobile network providers are designing efficient solutions to improve the user experience while continuing to maintain healthy revenue growth. MEC has been successfully proposed as an effective concept to defeat the current limitations in the Radio Access Network (RAN). The key point of MEC is to explore the use of mobile edge and push contents close to mobile users. For example, an MEC server can be deployed in the BS, gateways, and even mobile devices to allow computing and processing near mobile users. With the assistance of MEC, the stringent low latency requirements of 5G networks can be achieved.

MEC has become popular and widely studied in industries. Intel and Nokia Siemens Networks introduced a very first MEC platform which can be applied in real world [6]. In [7], IBM discussed the possibilities to use mobile edge virtualization and described a smart city scenario, where the near-real time manner information of people who are crossing a city can be collected by the mobile network. Furthermore, in [8], Saguna proposed a virtualized MEC architecture, Open-RAN, which provides a public platform to run different kinds of third-party applications. The major breakthrough in the MEC field was the MEC standardization, which was initiated by the European Telecommunications Standards Institute (ETSI) [9]. They introduced the full architecture, key techniques, and diverse practical use cases of MEC by connecting cloud and edge, and launched the MEC Industry Specification Group (ISG). The goal of this standardization efforts is to provide an open and crossing-domain environment, which can be accessed by different content providers and third-party applications at the mobile edge network (*i.e.*, the edge of the RAN), in order to overcome the limitations of cloud computing and guarantee a high-quality and low-latency experience. By integrating and utilizing the computing, communication, and storage resources at mobile network edge effectively, heavy computing can be offloaded and traffic flow congestions can be relieved from the core and backhaul network. It is envisioned that this decentralized cloud architecture forms the backbone of emerging 5G systems. In this case, traditional mobile base stations are converted to cloud-like servers, which have capabilities of cloud computing and provide services directly to end users at the mobile edge.

In the MEC architecture, communications can happen both horizontally and vertically. First of all, the MEC servers can be co-located with wireless Access Points (APs) as small data centers. The APs not only offer interfaces for network operators to deploy MEC servers, but also facilitate access to the backhaul network and service clouds. After deploying MEC servers, the APs are enabled to serve mobile devices directly. Then for the remaining mobile devices located far away from MEC servers or those that cannot fetch contents from MEC servers directly, D2D communications provide a capability of sharing and sending sources in an end-to-end manner between devices within a certain range of areas.

2.1.1 Key Technical Enablers

Cloud Computing — Cloud computing has achieved rapid development and earth-shaking changes. It is another great change after the big computer-to-client-server transformation in the 1980s. It aims to solve the problem of how to make full use of the original computing resources to reduce unnecessary loss. Cloud computing materializes the vision of utility computing and makes a central logical large-scale console possible. The Internet revolution triggered by cloud computing has centralization of computing and storage, and SDN conforms to this trend. Instead of using local servers, cloud computing provides access to data or applications on remote servers through the Internet. Applications or services provided by cloud computing can be categorized into different levels. Infrastructure-as-a-Service (IaaS) provides hardware related services such as servers and switches. Platform-as-a-Service (PaaS) providers offer a variety of solutions for developing and distributing applications, such as virtual servers and operating systems, as well as database systems. Software-as-a-Service (SaaS) is a layer that users can directly access and use the service (*e.g.*, send email, order products, view flight information, *etc.*).

Software Defined Networks — In existing networks, the control and forwarding of traffic depend on the implementation of the network device, while the operating system and dedicated hardware tightly coupled with the service characteristics are integrated in the device. These operating systems and dedicated hardware are developed and designed by each manufacturer. As a new type of network architecture, the goal of SDN is to separate the network's control plane from the data forwarding plane, so as to realize the programmable control of the underlying hardware through the software platform in the centralized controller, and implement flexible network resources. The typical architecture of SDN is divided into three layers. The upper layer is the application layer, including various services and applications. The middle control layer is mainly responsible for processing the data-plane resources, maintaining the network topology, state information, and so on. The lowest infrastructure layer is responsible for data processing, forwarding, and state collection based on flow tables.

SDN separates the underlying hardware from the services, which means that the hardware is only responsible for data forwarding and storage, so network infrastructure can be built with relatively general inexpensive equipment. This separation facilitates centralized network control so that the control layer can obtain the global information of the network resources, thereby performs global resource allocation and optimization according to the service requirements (such as traffic engineering and load balancing). Additionally, centralized control also makes the entire network logically regarded as a single entity for operating and maintaining, without the need for on-site configuration of physical devices, thereby improving the convenience of network control.

In [10], a hierarchical mobile SDN architecture with cross-domain orchestration is introduced, which follows the SDN principles and addresses the specific challenges for programmability and flexibility in 5G networks. In this architecture, a hierarchical, modular, and programmable control and orchestration plane across the domains is designed for resource coordination across radio and transport networks in the context of 5G networks. On one hand, an SDN-enabled mobile network can gather fine-grained real-time information from end users and different network domains [11]. On the other hand, various video caches can be virtualized as internal source points with a uniform interface to an SDN controller. With a global network view and timely information, the controller can dynamically adjust content delivery to improve resource utilization and user-perceived quality.

Network Function Virtualization (NFV) — As a more realistic SDN application scenario, NFV aims to implement dedicated network service functions in software and run these functions on a virtualized infrastructure, using controllers to coordinate the scheduling of network resources and computing resources. The goal of NFV is to replace private and closed network elements in communication networks to reach an open architecture for unified general-purpose hardware platforms and business logic software. By hardware decoupling and function abstraction, NFV enables network device functions to no longer depend on dedicated hardware. Resources can be fully and flexibly shared, which enables rapid development and deployment of new services, thus achieving automatic deployment, elastic scaling, fault isolation, and self-healing based on actual business needs. The combination of NFV and SDN will have a major impact on the future development of communication networks, but will bring new problems and challenges as well. In [12], Bradai *et al.* proposed a novel and simplified architecture for cellular networks using SDN and NFV, and exploited the capability of the mobile edge networks to gather the user information, which can be shared with third parties.

2.1.2 Applications and Use Cases

Augmented Reality (AR) — With the gradual popularity of smart devices, augmented reality applications are becoming more prevalent. For example, soccer fans can immerse themselves in the high-quality football game like they are watching the game at the front rows, by using AR equipment and 5G real-time video delivery. However, these applications require not only the devices themselves, but also a high level of computing power in addition to the basic equipment to cope with the overloaded real-time computing tasks. In the beginning, cloud computing can provide the computing power that is needed for AR applications. However, sensitive delays can dramatically reduce the user's experience. Towards this end, MEC implements computing, processing, and content delivery at the edge, thus offers a peer-to-peer approach for the AR environment and successfully solves the bandwidth and latency issues.

Video Streaming and Analysis — When mobile video traffic is exploding, network latency has greatly reduced the perception of mobile video audiences. Mobile video stagnation and buffering remain a big issue for operators and their customers. MEC is an effective solution when network congestion seriously affects the experience of mobile video watching.

First, since the mobile edge computing server can act as a memory close to the wireless side, the content can be cached into the MEC server in advance. When there is a demand for watching mobile video (*i.e.*, the user initiates a content request), the mobile edge computing server will immediately check whether the content requested by the user is already stored in the local cache. If no such content is found in the local cache, the MEC server needs to fetch and cache the requested video from the content provider. When other users generate the same demand next time, their requests can be fulfilled directly from the MEC server.

Second, the cross-layer interactive feedback can be utilized for video optimization. The MEC server senses the throughput rate of the underlying wireless physical layer, and the server (*i.e.*, the upper layer) decides to send video with different quality and definition to the user, which improves the user experience while reducing network congestion and improving link utilization.

Third, the MEC server can explore user perception. Due to the service and user-aware features of MEC, customers with different demands can be distinguished. In addition, MEC determines service levels to differentiate wireless resource allocation and data packet delay for users. In this case, network resources can be allocated reasonably to improve the overall user experience.

Internet of Things (IoT) Applications — The core of the IoT is to connect everything to the Internet, so that every object can communicate and be operated intelligently. Edge computing enables IoT to better sense, interact, and control objects through data analysis and processing capabilities closer to the edge. As an architecture that extends computing, networking, and storage capabilities from the cloud to the edge of the IoT network, MEC follows the mode that runs applications at the edge while managing them in the cloud. Currently, various smart devices with sensors are connecting to the Internet. On one hand, the rapid growth of the number of connections will generate massive amounts of data traffic. On the other hand, such connected devices often require intelligent computing. For the massive and rapid-growth issues of IoT data, directly building more and larger data centers will greatly increase management costs and reduce system reliability. MEC, as a small information center that is very close to the terminal information source, has capabilities to push applications, processing and storage to the mobile boundary, so that massive data can be processed efficiently without having to build additional data centers. In [9], an MEC server can work as a serving gateway to provide major services such as decision logic, database logging, remote provisioning and access control to the devices.

2.2 D2D Communications

D2D communications enables direct communication between user equipments (UEs) through techniques such as Wi-Fi Direct, Bluetooth, and LTE Direct. It provides attractive paradigms for content distribution in the 5G networks. Specifically, it enables direct communication including content sharing between user devices and communication is controlled by the mobile edge server, which is often co-located at the BS.

In the future 5G networks, UEs are distributed in a D2D-assisted network, where users not only have message exchange with the BS, but also are equipped with the function of message forwarding. Moreover, all users located in the D2D-assisted network can share a portion of their resources, such as stored content, information, and even network connection. In this case, the resources and services shared by user devices form a D2D network, which can be easily accessed by other mobile devices to avoid going through intermediate nodes. As one of the most important techniques in the 5G networks, the D2D features and capabilities will provide new business opportunities for operator-managed networks by supporting new use cases and service scenarios.

Communications between mobile devices in the D2D network can improve spectral efficiency. Simulations in [13] demonstrate that the network throughput performs better with the assistance of D2D cooperation. Besides direct connection between two UEs, the mobile device can also act as a relay to offload traffic in the RAN and the core network. In [14], the authors discussed the performance of the D2D network with consideration of mobile relays. The evaluation experiments indicate that both the coverage and edge throughput of the cell achieve better performance by exploring some mobile users as relays in the D2D network. Beyond these, many researchers intended to analyze the performance of D2D networks using different communication approaches. For example, the evaluation results in [15] show that, when the D2D links have power limitations, the outage probability can be reduced by considering hop-by-hop relaying instead of single relay selection. The capacity of D2D communications increases even under the influence of strong cellular signals.

2.3 Mobile Content Distribution

As the global mobile market is speeding up the migration to the 5G, it is found that video content distribution is a haunting service that overburdens mobile networks due to its overwhelming bandwidth consumption and stringent QoS requirements. To alleviate traffic load and improve QoS assurance, a widely studied technique is caching, which deploys replicas of popular video contents at geographically distributed locations to provide cost-effective content access. Caching has been quite successful in traditional CDNs coupled with the cloud for video services in wired networks. In recent years, researchers have shown that caching is also promising to support mobile content distribution and to cope with the unabated mobile traffic. With caching, content distribution mainly consists of two phases, *i.e.*, content placement and content delivery. In the following, we will review some state-of-the-art in the two aspects for mobile content distribution.

2.3.1 Content Placement

The key questions to be answered for content placement are *where to cache*, *what to cache*, and *how to cache* [16]. Here, how to cache focuses on the granularity of cached contents, which can be the file level, chunk level, packet level, or byte level. For example, real-time video services can employ finer-grained caching for more flexible content sharing, whereas this often involves deeper traffic inspection and larger overhead. What to cache determines the contents fetched for caching points while meeting certain constraints on storage space, transmission bandwidth, energy consumption, and so on. It has been observed that user requests exhibit a long-tailed feature in that a significant portion of the requests concentrate on a small portion of most popular contents. This feature of content popularity can be exploited to ensure sufficient caching for popular contents, while diverse contents are cached in neighbouring locations to complement each other.

Core Caching — Where to cache concerns the content storage locations, whose specific attributes further influence the selection of caching contents (*i.e.*, what to cache) and deploying methods (*i.e.*, how to cache). Built-in caching in mobile networks can be located inside the mobile core, RANs, or mobile devices. The most widely deployed place is the mobile core, which can benefit from the large potential user population and large storage space. Take EPC in LTE systems as an example. In EPC, P-GW connects the mobile network to the Internet or other data networks, while S-GW serves as a mobility management anchor to keep users' IP addresses unchanged when users' movement causes switching of attachment points [12]. Though P-GW has been considered as the in-network caching place, most mobile operators only have few P-GWs, *e.g.*, one of the largest mobile operators in the US only has 4-6 P-GWs nationwide [4]. If content caches are placed in P-GWs, they may be quickly overloaded by the tremendous access traffic. Another potential cache location in EPC is S-GWs. A large number of S-GWs are geographically distributed over wide areas and can better balance the traffic.

Edge Caching — Edge caching is a recent trend that aims to push content closer to end users by deploying caches in the RANs. Compared to caching in the mobile core, edge caching can save the traversal time through the core, and make it possible to improve content delivery by leveraging dynamic user demands and network information, such as traffic load, channel condition, interference, and resource availability. In [17], the authors studied reactive and proactive caching at the BSs in the RAN. The caching policies utilize user preference profiles and work together with the video-aware backhaul and wireless channel scheduling techniques to maximize the number of concurrent video sessions while satisfying their requirements on initial delay and stalling.

On the other hand, the performance gain with RAN caching can be limited by the small user population within a cell and small storage space. As a consequence, it is essential to enable collaborative caching among the BSs, so that user requests can be redirected or complementary contents can be shared [18]. In LTE systems, the X2 interface is defined to interconnect eNodeBs. Although the X2 interface is being increasingly used by mobile operators, it is mainly designed for exchanging control information to enhance handover and coordinate transmission in neighbouring cells. In [19], Tran *et al.* considered a cloud-based RAN (C-RAN), and proposed to add a *cloud cache* at the baseband unit (BBU) in addition to *edge caches* at the radio remote heads (RRHs). Motivated by the design principles of SDN and NFV for reducing operational costs and improving management flexibility, C-RAN centralizes the processing functionalities at the BBU in the cloud, while distributing lightweight RRHs at cell sites. As RRHs are connected to a common BBU by high-bandwidth low-latency fronthaul links, the cooperation between the cloud cache and edge caches is made much easier and smoother.

Device Caching — Caching can be further pushed into mobile devices. In particular, social-aware device caching assisted with D2D communications is attracting much research attention. It intends to leverage users' properties in both the physical network and the social network for content caching and sharing. On the physical side, important parameters include users' mobility characteristics (*e.g.*, contact frequency and contact duration), interference strengths of D2D links, spectrum resource availability, power budgets, device storage limits, and so on. On the social side, the social structure can be exploited to partition users into communities and identify most influential users of high centrality measures as caching candidates. The social structure can be derived by various means, *e.g.*, by mining data analytics on social ties, common interests, shared activities, social interactions, personal attributes, and so on.

Though there have been various studies on in-network caching, they mostly pay attention to caching in one domain, either in mobile core, mobile edge, or UEs. However, a holistic solution is more appealing as it can make best use of all caching resources to optimize content distribution. Fortunately, the fast-growing SDN and NFV technologies can offer perfect support for such a collaborative paradigm. A hierarchical mobile SDN architecture with cross-domain orchestration is introduced in [10]. In this architecture, a hierarchical, modular, and programmable control and orchestration plane across the domains is designed for resource coordination across radio and transport networks. We make good use of this collaborative paradigm provided by SDN architecture so that caching capabilities can be fully exploited in all domains. In our work, caching can be deployed at source points in mobile core, BSs in mobile edge, and mobile devices. With universal in-network caching, a request can be fulfilled by local device caching through D2D transmission, caching at the BS, or caching at the neighboring BSs with assistance of collaboration between mobile edge servers.

2.3.2 Content Delivery

For mobile content distribution, most work focuses on content placement. However, to make the best use of cached contents, another key aspect is content delivery, which directs a content request to an optimal cache location and routes the so-generated flow through the network. On one hand, cached contents can be updated during the delivery phase instead of waiting for the regular periodical replacement. The refreshed cache contents will further affect the fulfillment of subsequent requests. On the other hand, the cache selection and corresponding flow routing should not throttle other traffic. For example, a popular cache point may easily become a hot element of the network and interfere with regular flow routing if the content access traffic is not spread appropriately.

As discussed in Section 2.3.1, edge caching places contents in the RANs or mobile devices and often involves D2D communications to facilitate content delivery and offload traffic. In [20], Wang *et al.* considered the impact of D2D offloading on the caching strategy at the BSs. Content requests that are not satisfied by nearby devices with feasible D2D links have to be forwarded to the BSs and thus affect the content caching at the BSs. Taking into account the impact of D2D content delivery, this work further develops a Markov decision process and a Q-learning based strategy for the BS caching placement.

In [4], the request routing problem is studied with built-in caches at S-GWs in EPC. Since S-GWs are actually interconnected in a flat topology, now with built-in sources, a video request can be served directly at an S-GW, or the S-GW is allowed to fetch the video from other S-GWs or source points. The proposed fast algorithms can obtain a near-optimal routing solution that minimizes the maximum link utilization (*i.e.*, the congestion level) or the total link cost. A similar request routing problem is studied in [21], which assumes that caches are deployed in EPC and RANs. It considers a hierarchical topology from an origin server to an EPC cache and then to a chain of RAN caches at eNodeBs. A RAN cache is directly connected to at most two neighbouring RAN caches via the X2 interface. The proposed solution transforms two decomposed subproblems for content placement and request routing into the problems of maximizing a submodular function subject to a matroid constraint.

2.4 D2D-Assisted Content Distribution

2.4.1 Content Caching and Delivery in D2D Networks

Expedited by the popularity of social media, diverse online video contents are most likely to be repeatedly demanded by a huge number of users simultaneously. These demands can cause congestion in intensive hotspots and during peak hours of the day. Edge caching proposes to store these contents closer to the end users on the edge. Content caching can be deployed further beyond the edge to offload some requests for the same content toward the user plane. D2D caching perfectly supports this requirement through placing popular video contents at some preselected devices temporarily, and these cache devices can fulfill other requests in its close proximity. Some research works have studied cache placement strategies in D2D-assisted networks. However, three challenges need to be conquered. The first one is to obtain a solution that maximizes request offloading, *i.e.*, to maximize content sharing with the minimum content caching in devices. The second challenge is to address the limitations of individual devices when placing and forwarding contents in the D2D network. The third one is to take into account the social relationships among all users, which would affect preference of matching and forwarding between UEs.

Some studies focus on the content caching strategies in D2D networks. For instance, the maximization problem for offloading probability is solved in [22] through an optimal content caching approach with effective transmission accessibility between devices. In [23], a probabilistic caching placement strategy is studied for stochastic D2D caching networks, and the optimal caching probabilities that maximize the requests successfully served by local device caches are obtained via numerical optimization. Moreover, in [24], the authors optimized the strategies for cache device distribution for the purpose of maximizing the average density of successful receptions considering interference and noise. These studies do not consider the user preferences for content caching and forwarding. In other words, some users may unwilling to cache a video that is not requested by themselves. Moreover, content sharing hardly happens under a circumstance that two users have no social relationship between each other.

There are also some researches that study device pairing and content forwarding with assistance of device caching and D2D communications. Different from the studies on content caching strategies, they paid attention to the matching of request devices with devices that have already cached videos. For example, in [25], the authors exploited interference-aware collaborations among devices and proposed an approximation approach that aims to offload requests by pairing request devices with cache devices in close proximity. In [26], the approximation algorithm is further improved to a three-step approach for the device paring problem to maximize the request coverage through D2D communications. Moreover, a heuristic device pairing approach with channel-awareness is introduced in [27]. In [28], the authors studied the optimization of multicast message allocation in D2D-assisted content distribution to relieve traffic from the BS and minimize transmission cost.

However, many previous studies on cache device selection and device pairing consider strategies of either caching placement or device pairing and content forwarding. In other words, they consider simplified constraints for individual devices and the D2D channel limitation of the BS. However, caching placement and forwarding should be considered jointly due to the interaction between each other, which can affect the performance of request screening. For example, individual serving budget and social relationship can affect caching placement to achieve maximum request coverage. Moreover, each selection of device caching will affect subsequent device matching decisions. Therefore, to ensure successful content delivery between devices, joint optimization of content caching and delivery is critical to improve the performance of cache-enabled D2D networks. In our work, we jointly consider device caching placement and device pairing when formulating the optimization problem of request screening, instead of solving them separately. Specifically, we take user preferences for content forwarding into account when placing content caching in devices.

2.4.2 Social-Awareness for D2D-Assisted Content Distribution

Social-aware device caching with D2D-assistance is attracting more and more research attentions. It intends to leverage users' properties in both the physical network and the social network for content caching and sharing. On the physical side, important parameters include users' mobility characteristics (*e.g.*, contact frequency and contact duration), interference strengths of D2D links, spectrum resource availability, power budgets, device storage limits, and so on. On the social side, the social structure can be exploited to partition users into communities and identify most influential users of high centrality measures as caching candidates. The social structure can be derived by various means, *e.g.*, by mining data analytics on social ties, common interests, shared activities, social interactions, personal attributes, *etc.* In [29], the authors proposed a D2D-assisted solution that profits from social relationships for source selection and data forwarding. In [30], a hypergraph model is proposed to incorporate multidimensional information for social-aware D2D caching. Accordingly, the hypergraph techniques such as colouring and matching can be used to optimize D2D spectrum management and cache placement. Moreover, due to direct engagement of end users in the service chain, it is inevitable to address the incentive constraint in device caching. In [31], a non-cooperative game is proposed for incentivized social-aware caching and its pure strategy Nash equilibrium is derived. As content caching and sharing cost non-negligible bandwidth, energy, and storage resources, this solution mainly concerns the selfish human nature and takes into account data placement and access costs.

Chapter 3

Framework and System Model

As described in Chapter 2, though there have been various studies on in-network caching, how to enable collaboration from the core, mobile edge to user devices is still underexplored. A holistic solution is appealing as it can make best use of all caching resources to optimize content distribution. In this chapter, we consider an integrated framework, which incorporates caching in the core, mobile edge and user devices, and enables collaboration among virtualized source points across domains. Then, based on the proposed framework, we first describe our content distribution model of the whole network, then we focus on the request screening model in an edge server and the request routing model.

3.1 Integrated Content Distribution Framework

There has been considerable research progress on mobile content distribution with in-network caching, especially on what to cache and how to cache at specific storage locations. Nevertheless, how to orchestrate caches in different network segments and enable effective collaboration among them remains under-explored. In Fig. 3.1, we consider an integrated framework that addresses this question by efficiently integrating the caching capabilities at the mobile core, BSs and mobile devices. This

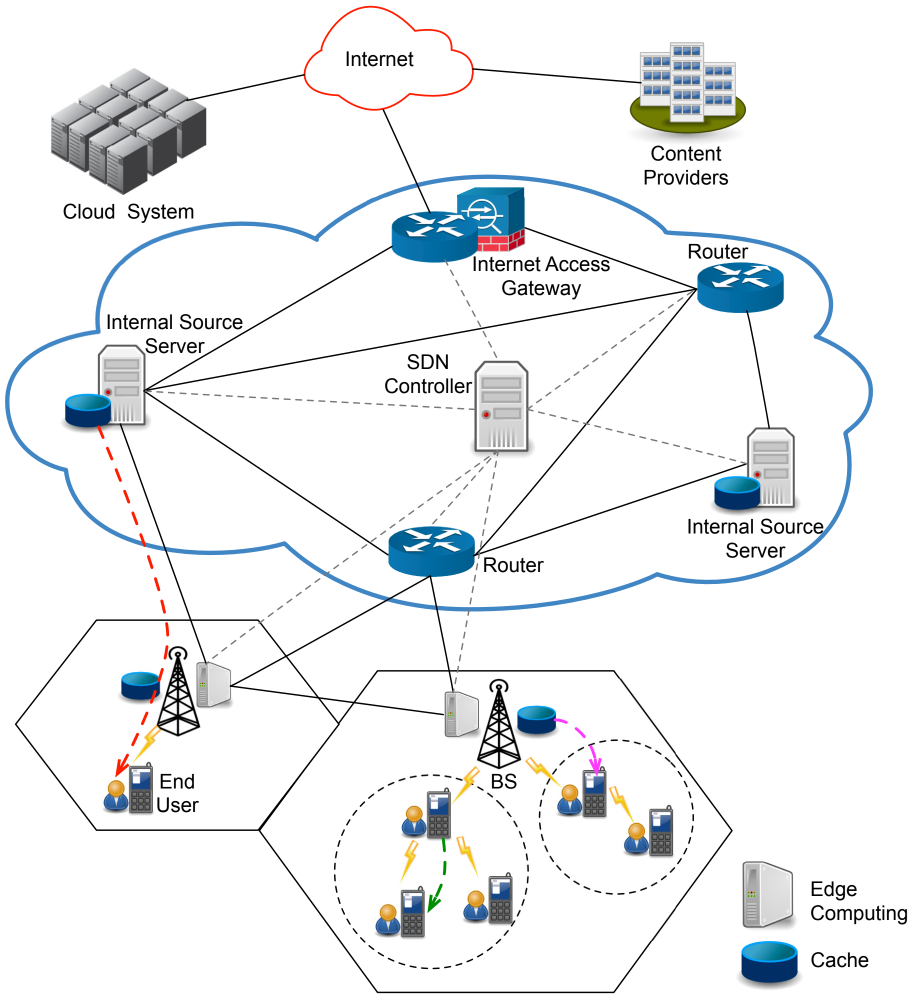


Fig. 3.1: An integrated framework for collaborative mobile content distribution with universal in-network caching.

framework takes advantage of evolving technologies such as MEC, NFV, and SDN and features in three aspects. First, it involves more extensive cooperation between edge caching and core caching. In particular, the deployment of MEC can offer an ideal support for this framework. Second, the framework considers virtualized sources so that in-network caching can be more efficiently integrated for best utilization. Third, the framework incorporates an end-to-end (E2E) paradigm to exploit fine-grained dynamic context information for optimizing content distribution.

Assistance with MEC and Edge-Core Collaboration – To tackle the surging traffic arising from content distribution, the 5G mobile networks are expected to employ innovative technologies to achieve substantial advancement in terms of network capacities and data rates. For instance, it is envisioned that 5G networks would be ultra-dense and heterogeneous with small cells nested within macro-cells. Though 5G networks demonstrate these promising prospects, additional technologies are required to enable rapid deployment of emerging applications and timely upgrade of existing services. Similar to concepts such as mobile cloud computing, cloudlet, and fog computing, MEC is such a key technology for 5G networks, which intends to move contents, network functions and services close to end users [32]. On one hand, it aims to integrate and utilize the abundant computing, communication, and storage resources at the network edge. On the other hand, MEC can leverage close proximity to support applications with low latency, high data rates, and low power consumption.

As MEC servers are often located near the BSs and thus spatially distributed, it is vital to engage *horizontal cooperation* among MEC servers to compensate for the limited resources. On the other hand, it is known that content caches have been widely deployed in the mobile core as well as the traditional CDNs that are usually hosted in the centralized cloud. Therefore, *vertical cooperation* between MEC and the mobile core or the cloud can further complement the limitations of each other.

Universal In-Network Caching as Virtualized Sources – As discussed in Section 2.3.1, content caches can be placed in remote CDN servers in the cloud, interconnecting gateways in the mobile core, and/or BSs in the RANs. As device caching is more transitory, we will consider it separately. The external caches in the CDNs and those in the mobile core can be virtualized as *internal source servers* in a converged infrastructure, while the caches in the RANs can be considered as virtual sources at the *edge servers*. As seen in Fig. 3.1, these source servers can be considered as overlay functions at the existing network components. For instance, the internal source servers can run on top of serving gateways in the mobile core, while the edge servers can sit at the MEC nodes. In addition, the intermediate nodes can be the interconnecting routers in the transport network. As such, universal in-network caching is integrated logically for best utilization.

Dynamic Content Delivery with an E2E SDN paradigm – Furthermore, SDN-enabled mobile networks are becoming the mainstream design for 5G, which offers a scalable paradigm by decoupling the control plane and data plane and lifting up intelligent control functionalities to a logically centralized controller. Originally, the SDN design for 5G networks focuses on the mobile core, and it is now extending to the RANs and even mobile terminals. Ultimately, an end-to-end (E2E) SDN solution across different network segments would provide better control flexibility and agility [23]. Incorporating such an SDN-enabled mobile network into the framework, the controller can efficiently manage the virtualized caching sources via a unified and programmable interface, which can even be opened to third-party service providers. On one hand, the controller can collect fine-grained network and user information across various domains in a timely manner. The collected information can range from device locations, video request patterns, flow-level dynamics, buffered data amounts and playing time of individual users, to available link capacities and wireless channel conditions. The collected information is not only of great breadth and fine granularity, but also updated in a time scale comparable to the dynamics of user engagement, typically a few seconds. On the other hand, the controller can compile the gathered information into “decisions” or “policies” and deploy them to network components via the control plane to optimize network performance. It is known that a typical video playout lasts 10-300 seconds, while content placement is usually updated on a weekly basis. As seen, the fine-grained information can be better utilized to adapt content delivery and unleash the potential benefits of in-network caching. Particularly, this information can be exploited to coordinate different domains and steer the video flows through the network with minimum congestion. The rapid adaptation is also helpful to address dynamics posed by user mobility that is usually in a larger time scale.

3.2 System Model

3.2.1 Content Distribution Model

Consider the integrated content distribution framework with universal in-network caching in Fig. 3.1, the network nodes include the intermediate servers, internal source servers, or edge servers, which are virtualized as source service points. Specifically, internal source servers are located in the core network and equipped with caching capabilities. All edge servers are co-located with BSs, and a portion of them enable content caching. Intermediate servers act as routers in the core network which enable content forwarding between internal source servers, or between the mobile edge and the core network. Here, we can model the network as a directed graph $G = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of interconnected source nodes, and $\mathcal{E}$ is the set of links and each link $e \in \mathcal{E}$ has a capacity $c(e)$.
Assuming that an edge server can be concretized as a BS that caches content and provides services for a number of UEs, which are uniformly located in the coverage of the BS, *i.e.*, a circular area of radius L with the BS at the center. Let $\mathcal{U}_j$ denote the set of UEs that are located within the coverage area of BS j. The wireless communication link between the edge server and UEs can be modelled by taking into account log-distance path loss and Rayleigh fading. Then, the maximum achievable data rate at device $i \in \mathcal{U}_j$ is given by

$$\psi_i = B_i \cdot \log_2(1 + \Gamma_i) \tag{3.1}$$

where B_i is the bandwidth of the wireless link allocated to device i, and Γ_i is the received signal-to-noise ratio (SNR) at device i. Based on the wireless channel model, we have

$$\Gamma_i = \frac{\left(\frac{P_j}{L_i}\right) \cdot ||f_i||^2}{\sigma^2} \tag{3.2}$$

where P_j is the transmit power of BS j, L_i is the path loss at device i, f_i is a random variable that captures the small-scale fading of the wireless link, and σ^2 is the power of additive white Gaussian noise at device i. Considering the log-distance path loss model, we have $L_i \propto \ell_{ij}^{\alpha}$, where ℓ_{ij} is the separation distance between device i and BS j, and α is the path loss exponent. Assuming the small-scale channel variations are modelled by Rayleigh fading, $||f_i||^2$ follows an exponential distribution of unit mean.

Each device only requests one file from a video library (denoted by $\mathcal{V}$) and all requests follow a Zipf popularity distribution. Therefore, the probability that device i requests video $k \in \mathcal{V}$ is given by

$$\vartheta_{i,k}^{(r)} = \frac{\frac{1}{k^{\zeta_r}}}{\sum_{j=1}^{|\mathcal{V}|} \frac{1}{j^{\zeta_r}}}, \quad \zeta_r > 0 \tag{3.3}$$

where the exponent ζ_r characterizes the relative popularity of videos. A higher value of ζ_r indicates that all users' requests are concentrated on fewer videos. We denote the video requested by device i as m_i. The set of requests submitted to BS j is denoted by $\mathcal{R}_j$.

For a cache server in the core network, as it is often hosted in the cloud without a capacity constraint, we assume that the cache stores the entire video library. In contrast, an edge server can only cache most popular videos according to its storage limit. Considering the random caching policy studied in [33], we assume that edge server j caches file k according to a Zipf distribution with exponent ζ_c with probability

$$\vartheta_{j,k}^{(c)} = \frac{\frac{1}{k^{\zeta_c}}}{\sum_{i=1}^{|\mathcal{V}|} \frac{1}{i^{\zeta_c}}}, \quad \zeta_c > 0. \tag{3.4}$$

Video requests are initiated by mobile devices and first submitted to the associated BS, which acts as a transceiver connecting a number of other devices to one another and/or to a wider area. If two devices request the same video, fall within each other's transmission range and have a strong social relationship, it is more efficient that the edge server only satisfies one request and the receiving device caches the video to fulfill the other request by D2D communications. Considering that some popular video is demanded by multiple neighbouring users and thus can be served by local device caching, the BS needs to appropriately select the video requests offloaded to D2D communications. This is the *request screening problem*, which is roughly introduced in Section 1.2.1 and will be described in detail in Chapter 4. The request screening model is presented in Section 3.2.2.

Once certain requests are screened and directed to D2D communications, the remaining requests unfulfilled by local caching have to be further satisfied by other cache sources. There, the controller needs to solve the request routing problem, which is introduced in Section 1.2.2. The request routing model is introduced in Section 3.2.3, and the problem will be investigated in Chapter 5.

Leveraging the SDN-enabled framework, the controller can collect complete information from the control plane, such as static configurations of the network, dynamic states of wireless channels, and dynamic traffic loads from video requests. Taking advantage of the collected information, the controller can optimize content delivery from interval to interval. As user engagement with one video request is usually in the order of 10s [34], we consider an interval duration in a time scale comparable to the dynamics of user engagement. Within each interval (denoted by τ), the controller can collect or estimate real-time user context, including the data rate of the wireless link to a UE, the requested video information (*e.g.*, video coding rate, video clip

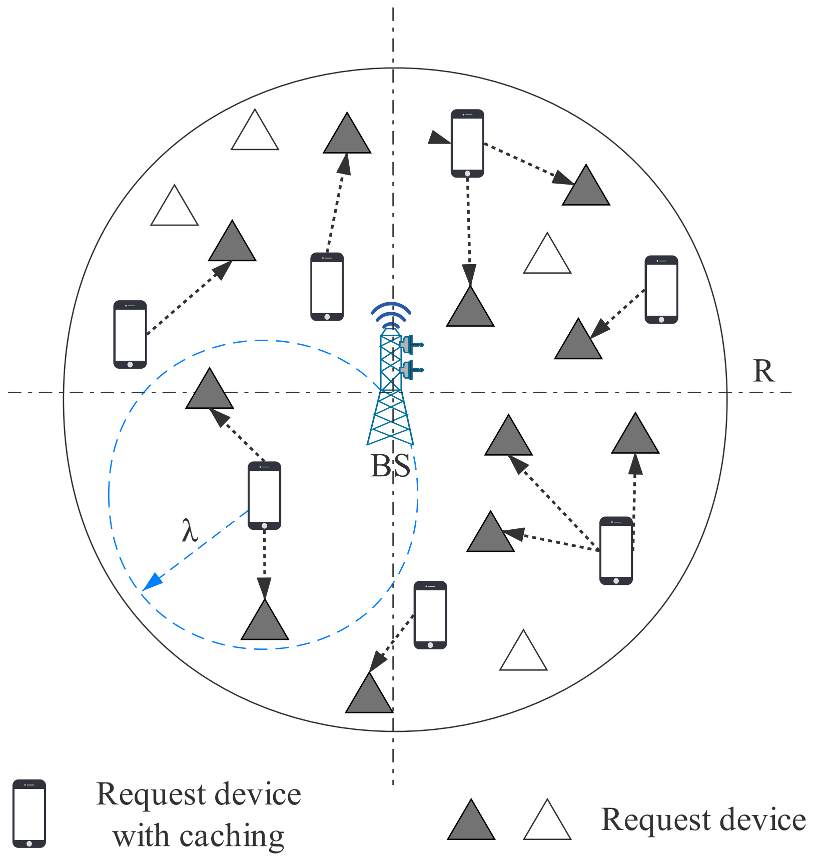


Fig. 3.2: Content distribution scenario with request screening.

duration, and total data size), the current playing time and buffered data amount at the UE.

3.2.2 Request Screening Model

Based on the content distribution model in Section 3.2, we further focus on a request screening model at the mobile edge as depicted in Fig. 3.2. As described in Section 3.2.1, the BS can appropriately screen the video requests and offload some to be fulfilled by cache devices via D2D communications. This D2D-assisted content distribution not only can offload traffic from the mobile edge but also is energy-efficient in view of the close proximity. Since each edge server $j \in \mathcal{N}$ needs to solve the request screening problem, we skip subscript j and use $\mathcal{U}$ for any device set.

A set of request user devices, denoted by $\mathcal{U} = \{1, 2, \cdots, n\}$, are requesting contents from a video library $\mathcal{V}$ that contains m videos in total. All devices are located in the coverage of a BS, a circular area of radius L with the BS at the center. The distance between two devices i and j $(i, j \in \mathcal{U})$ is denoted by d_{ij}. The collaboration distance between any two devices is denoted by ϖ, which means that device i and j cannot share content between each other if d_{ij} is bigger than ϖ. Besides the physical attributes, we also consider a social network among all users in $\mathcal{U}$, which abstracts social relationships among devices in $\mathcal{U}$ and is denoted by graph $G_s(\mathcal{U}, \mathcal{A})$. An edge exists between two users with a social relationship. Each device only requests one file from video library $\mathcal{V}$ and all requests follow a Zipf popularity distribution with coefficient ζ_r.

Once requests are offloaded and the matching result is obtained, the BS can identify each device as a source, a receiver, or a cellular user. The source has responsibility to receive contents from BS and transmit them to its receiver(s) through D2D link. Also, some devices whose requests cannot be fulfilled by device caching thus must fetch contents directly from the BS acting as regular cellular users. In this case, all devices in $\mathcal{U}$ can be categorized into source devices, receivers, and cellular users. We denote the set including all D2D groups (*i.e.*, source devices) as $\mathcal{U}_t$, and the set of all cellular users is denoted by $\mathcal{W}$. In order to fulfill requests of cellular users, first, the BS allocates independent channels for each cellular users in $\mathcal{W}$. Then, for resource efficiency, every source device can reuse one of cellular users' channels as its D2D link to finish content propagation. However, the negative effect due to co-channel interference should be taken into account to avoid transmission conflicts.

Based on the D2D-assisted content distribution scenario illustrated in Fig. 3.2, we raise the request screening problem, which splits the requests into those served by local device caches or others to be fulfilled by in-network caches. The goal of the request screening problem is to maximize the number of requests fulfilled by local caching while satisfying certain resource and feasibility constraints. We first consider the *collaboration distance* ϖ between devices to ensure QoS of D2D communications. That is, only request devices located within the collaboration range of the selected cache device can be fulfilled. Similarly, a source device can only fulfill no more than a certain number of requests from other devices due to the stint of battery life, network bandwidth, and private information security. As most users are unwilling to share contents with others they do not trust, we require that the users of selected source devices should have social relationships with their matched receivers to protect information privacy.

Moreover, considering battery life, memory size, and privacy protection, any device i has an individual budget β_i for content caching and sharing. In other words, device i can serve no more than β_i user requests if it is chosen to cache the received video and act as a source device. Besides, the total number of selected source devices for video caching is limited by γ for each BS. Although more video caches lead to better screening performance, it is infeasible to unrestrictedly allocate the mobile edge resources for D2D transmissions. In practice, the BSs need to reserve sufficient channel resources for other uses, such as for the regular cellular links. Therefore, in addition to considering the resource constraint of each individual device, we limit the total number of selected sources for content caching since each source needs at least one D2D channel to serve the matched request(s). As such, we restrict the total number of D2D channels used to serve the requests within the cell. In a nutshell, *the device caching and matching problem*, which aims to select at most γ appropriate devices in $\mathcal{U}$ to cache video contents and match them to other request devices in $\mathcal{U}$. Rather than defining a one-on-one matching, we pursue a one-to-many matching solution that each transmitter can share its content to multiple receivers.

Solving the device caching and matching problem, we can obtain at most γ D2D pairs or groups (with one D2D transmitter and one or multiple D2D receivers), which require at most γ distinct resource channels for the D2D links. Assume that co-channel interference is negligible beyond the transmission range ϖ. Considering the spatial distribution of the D2D users and underlaying cellular users in set $\mathcal{W}$ that share their channels, we need to properly allocate the D2D channels to avoid co-channel interference and minimize channel occupation time, *i.e.*, *the D2D channel allocation problem.* As such, these channels can be released as early as possible for other uses.

3.2.3 Request Routing Model

Once all requests $\mathcal{R} = \cup_j \mathcal{R}_j$ are screened, the remaining requests unfulfilled by local caching (denoted by set $\mathcal{R}'$) are further directed to edge servers. For a video request $(i, k) \in \mathcal{R}'$ at device i for video k, the user context information is gathered at the controller and utilized for subsequent content delivery. After screened and obtained the remaining requests set, the demand rate for each remaining request is calculated in the controller so that the wireless channel source can be reasonably allocated. The demand rate for a video request $(i, k) \in \mathcal{R}'$ at device i for video k is given by

$$d_i^k = \min(\psi_i, q_i^k) \tag{3.5}$$

where ψ_i is the achievable data rate at device i through the wireless link as defined in (3.1),and q_i^k is the required transmission rate for video k to meet desired playing smoothness. The transmission rate $\tilde{d_i^k}$ requested by device i for video k depends on the video coding rate, client buffer state, and other video playout information. As seen in (3.5), the demand rate of each unfulfilled request takes the minimum of the achievable data rate limited by the wireless channel conditions and the video-dependent transmission rate.

In (3.5), q_i^k depends on the video delivery and playout process. While the edge servers deliver video content to the device, the received data are first buffered at the video client for an initial buffering period. This involves a short start-up delay in the order of a few seconds. Then, the video player at the client begins to fetch the buffered data for playout. To ensure smooth playout, it is important to keep a sufficient amount of data in the buffer that are continuously consumed for playout. Specifically, q_i^k is determined according to a target buffer time T_i^k, which gives the extra time that video playout can sustain using only remaining data in the buffer. At the beginning of each interval, we can record the already downloaded data amount s_i^k and current playing time p_i^k since playout begins after a short start-up delay. Then, assuming that $\psi_i \geq q_i^k$ and thus $d_i^k = q_i^k$, we have the expected T_i^k at the end of the interval given by

$$T_i^k = \frac{s_i^k + q_i^k \cdot \tau}{\mu_i^k} - (p_i^k + \tau) = \left(\frac{s_i^k}{\mu_i^k} - p_i^k - \tau \right) + \frac{q_i^k \cdot \tau}{\mu_i^k} \tag{3.6}$$

where μ_i^k is the video coding rate. Here, the first term in the final expression of (3.6) is the remaining buffer time at the end of the interval (denoted by b_i^k), while the second term is the increase of buffer time with newly received data. Then, we can obtain the required transmission rate for video k

$$q_i^k = \frac{\left[T_i^k - \left(\frac{s_i^k}{\mu_i^k} - p_i^k - \tau \right) \right] \cdot \mu_i^k}{\tau} = \frac{(T_i^k - b_i^k) \cdot \mu_i^k}{\tau}. \tag{3.7}$$

When selecting sources and flow paths for each video request, the controller intends to maximize the remaining playout time of each video in the buffer to guarantee the playout smoothness and reduce latency. Meanwhile, since each link has its flow capacity, the controller needs to appropriately route data flows to balance traffic and avoid congestion in network links.

Chapter 4

Request Screening

Based on the integrated content distribution model in Chapter 3, this chapter studies the request screening problem, which is introduced in Section 1.2.1 and Section 3.2.2. We first formulate two subproblems for request screening, *i.e.*, the device caching and matching problem and the D2D channel allocation problem and analyze their computational complexity. Then, we provide two heuristic algorithms for the two subproblems respectively.

4.1 Problem Formulation

4.1.1 Device Caching and Matching Problem

Given the device matching in our content distribution scenario, we select both sources for content caching and their receivers from device set $\mathcal{U}$. Since every matching between sources and receivers must be feasible, we can abstract the relationship between them with a bipartite graph, which is illustrated in Fig. 4.1. An edge in Fig. 4.1 indicates the feasibility of sharing video contents between two devices. We use a binary variable l_{ij} to represent whether D2D communications and content sharing can happen between device i and j. There exists an edge between devices i

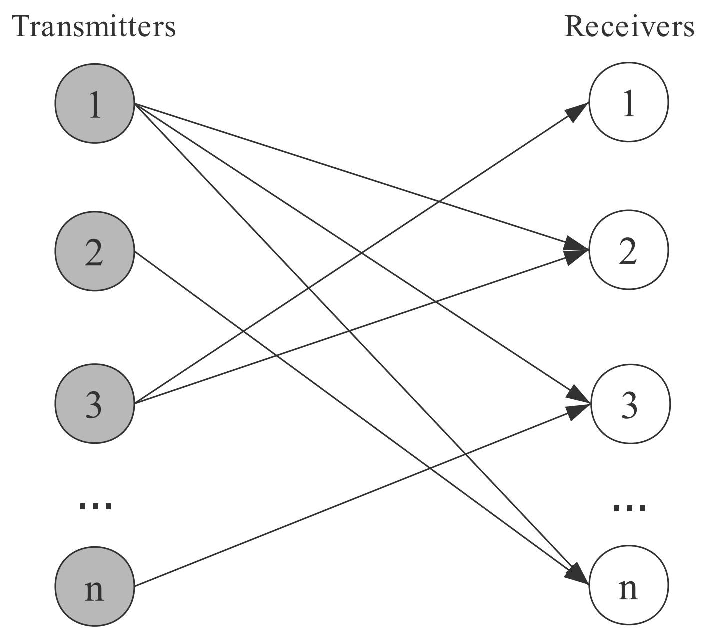


Fig. 4.1: A bipartite graph model for transmitters and receivers.

and j if $l_{ij} = 1$. Here, l_{ij} is defined as follows:

$$l_{ij} = \begin{cases} 1, & \text{if } m_i = m_j, d_{ij} < \varpi, \text{ and } e_{ij} \in \mathcal{E}_b \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

where ϖ is the radius of the transmission range, also termed as the collaboration distance, and $\mathcal{E}_b$ is the social relationship graph among all devices. When device i is selected as a cache device (*i.e.*, a source), it means that it can choose to fulfill the request from device j only if $l_{ij} = 1$. After the edge server receives all video requests from user devices, we filter requests aiming to maximize the total number of requests that can be fulfilled by local device caching, while satisfying the constraints of collaboration distance, budget of serving, social relationship, and the number of simultaneous D2D transmissions. According to the above requirements, we can formulate the device caching and matching problem as follows:

$$(P_1) \quad \max. \quad \sum_{i \in \mathcal{U}} \sum_{j \in \mathcal{U}} x_{ij} \tag{4.2a}$$

$$\text{s.t.} \quad \sum_{i \in \mathcal{U}} x_{ij} \leq 1, \forall j \in \mathcal{U} \tag{4.2b}$$

$$\sum_{j \in \mathcal{U}} x_{ij} \leq \beta_i, \forall i \in \mathcal{U} \tag{4.2c}$$

$$\sum_{i \in \mathcal{U}} z_i \leq \gamma \tag{4.2d}$$

$$x_{ij} = 0, \text{ if } l_{ij} = 0, \forall i, j \in \mathcal{U} \tag{4.2e}$$

$$\sum_{j \in \mathcal{U}} x_{ij} = \begin{cases} 0, & \text{if } z_i = 0, \forall i \in \mathcal{U} \\ \geq 1, & \text{if } z_i = 1, \forall i \in \mathcal{U} \end{cases} \tag{4.2f}$$

$$\sum_{i \in \mathcal{U}} x_{ij} \leq 1 - z_j, \forall j \in \mathcal{U} \tag{4.2g}$$

$$x_{ij} \in \{0, 1\}, \forall i, j \in \mathcal{U} \tag{4.2h}$$

$$z_i \in \{0, 1\}, \forall i \in \mathcal{U}. \tag{4.2i}$$

Here, the decision variable x_{ij} is a binary variable that indicates whether device i is selected to serve the request from device j. Hence, the objective function in (4.2a). maximizes the total number of requests to be fulfilled by selected sources. The first constraint requires that each request should not be fulfilled by more than one source, the second constraint limits the serving bound of each device, and the third constraint indicates that the number of selected sources cannot exceed the maximum number of D2D links allowed in the cell (*i.e.*, γ). To make sure that the matching between sources and receivers is valid, the fourth constraint limits the matching among the feasible edges in the bipartite graph in Fig. 4.1, which makes it consistent with our request screening model in Section 3.2.2.

In addition to $\{x_{ij}\}$, this problem also involves binary decision variables $\{z_i\}$, where $z_i = 1$ indicates that device i is selected as a source to serve other request(s). Clearly, $\{x_{ij}\}$ and $\{z_i\}$ are not independent but related by the conditional constraint in (4.2f). For formulation convenience, we can replace this conditional constraint by the following inequality constraints:

$$\begin{cases} \sum_{j\in\mathcal{U}} x_{ij} \geq 1 - M\cdot(1-z_i), & \forall i \in \mathcal{U} \\ \sum_{j\in\mathcal{U}} x_{ij} \leq M\cdot z_i, & \forall i \in \mathcal{U} \end{cases} \tag{4.3}$$

where M is a sufficiently large constant that meets the condition $M \geq \max_i \beta_i$. It is verifiable that the inequality constraints in (4.3) are equivalent to the conditional constraint in (4.2f).

Constraint (4.2g) further narrows the relationship between x_{ij} and z_i so that when a device is selected as a source it cannot be further served by any other device. In Fig. 4.1, the set of nodes on the left is in fact the same set of nodes on the right. According to constraint (4.2g), if a left node j is selected as a source, i.e., $z_j = 1$, the final matching should not include any edge incident on the corresponding right node j, i.e., $\sum_{i\in\mathcal{U}} x_{ij} = 0$.

In summary, we can reformulate the device caching and matching problem in (4.2) as an integer linear program (ILP) in the following:

$$(P_2) \quad \max. \sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{U}} x_{ij} \tag{4.4a}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{U}} x_{ij} \leq 1, \qquad \forall j \in \mathcal{U} \tag{4.4b}$$

$$\sum_{j\in\mathcal{U}} x_{ij} \leq \beta_i, \qquad \forall i \in \mathcal{U} \tag{4.4c}$$

$$\sum_{i\in\mathcal{U}} z_i \leq \gamma \tag{4.4d}$$

$$x_{ij} \leq l_{ij}, \qquad \forall i,j \in \mathcal{U} \tag{4.4e}$$

$$\sum_{j\in\mathcal{U}} (-x_{ij}) \leq M\cdot(1-z_i) - 1, \qquad \forall i \in \mathcal{U} \tag{4.4f}$$

$$\sum_{j\in\mathcal{U}} x_{ij} \leq M\cdot z_i, \qquad \forall i \in \mathcal{U} \tag{4.4g}$$

$$\sum_{i \in \mathcal{U}} x_{ij} \leq 1 - z_j, \qquad \forall j \in \mathcal{U} \tag{4.4h}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i, j \in \mathcal{U} \tag{4.4i}$$

$$z_i \in \{0, 1\}, \qquad \forall i \in \mathcal{U}. \tag{4.4j}$$

4.1.2 D2D Channel Allocation Problem

After solving the device caching and matching problem, we can obtain at most γ D2D groups. Then, the BS needs to allocate at most γ channels for these D2D transmitters to forward the cached contents. This brings about the D2D channel allocation problem. To improve spectrum efficiency, these D2D users can reuse the channels allocated to certain underlaying cellular users, which thus causes interferences among D2D users and cellular users. An important factor that influences interference strength is the distance between a transmitter and the interfered receivers. Depending on their spatial locations, we can obtain a conflict graph $G_c = (\mathcal{W} \cup \mathcal{U}_t, \mathcal{A}_c)$, where the vertices are the cellular users in $\mathcal{W}$ and D2D groups in $\mathcal{U}_t$ (one vertex for each D2D group), and an edge exists between any two vertices if they fall within transmission range ϖ.

Fig. 4.2 shows an example of the conflict graph. Here, the cellular users are labelled by distinct colors, which represent that they have been allocated orthogonal channels. Fig. 4.3 shows an example for interferences between D2D groups, where a D2D group with source device i and receiver set $\mathcal{U}_{r,i}$ conflicts with another D2D group with source device j and receiver set $\mathcal{U}_{r,j}$. If at least one receiver device in $\mathcal{U}_{r,j}$ is located within the transmission range of source device i, we assume that the interference between these two D2D groups is unacceptable. Since D2D groups i and j are in conflict, they cannot reuse the same cellular channel. Hence, Fig. 4.2 uses distinct colors for the corresponding vertices. Moreover, the selection of the reused cellular channels should also take into account the interference between D2D users and cellular users,

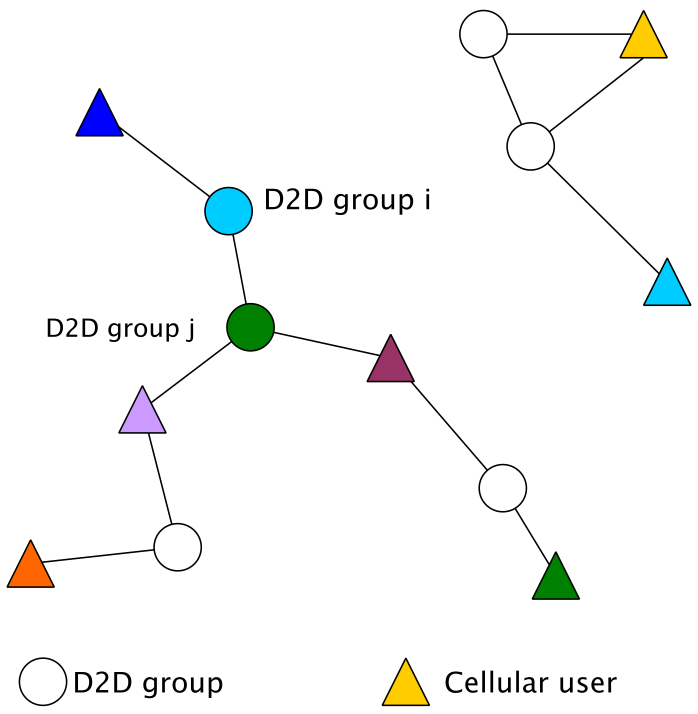


Fig. 4.2: Conflict graph for D2D channel allocation.

which can be characterized by another conflict graph similar to Fig. 4.3.

Based on the constructed conflict graphs, the D2D groups in Fig. 4.2 should be colored properly to meet the interference constraints. Given that the data transmission for D2D group i lasts for a duration τ_i, the D2D group will occupy the allocated channel for time τ_i. To minimize the total channel occupation time, we formulate the D2D channel allocation problem as follows:

$$(P_2) \quad \min. \sum_{k\in\mathcal{W}} z_k \tag{4.5a}$$

$$\text{s.t.} \quad \sum_{k\in\mathcal{W}} x_{ki} = 1, \qquad \forall i \in \mathcal{U}_t \tag{4.5b}$$

$$x_{ki} + \zeta_{ij} x_{kj} \le 1, \qquad \forall k \in \mathcal{W}, \forall i, j \in \mathcal{U}_t \tag{4.5c}$$

$$x_{ki} \le 1 - \rho_{ki}, \qquad \forall k \in \mathcal{W}, \forall i \in \mathcal{U}_t \tag{4.5d}$$

$$x_{ki}\tau_i \le z_k, \qquad \forall k \in \mathcal{W}, \forall i \in \mathcal{U}_t \tag{4.5e}$$

$$x_{ki} \in \{0, 1\}, \qquad \forall k \in \mathcal{W}, \forall i \in \mathcal{U}_t \tag{4.5f}$$

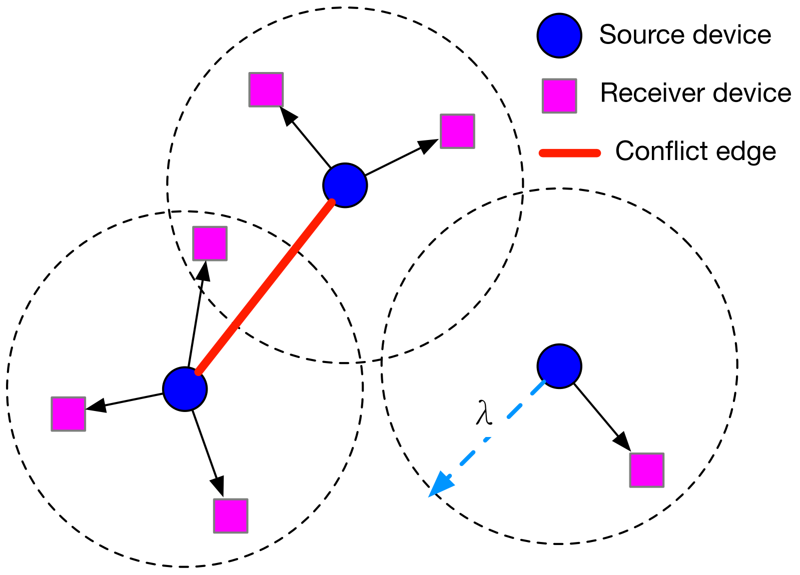


Fig. 4.3: Example of edges in the conflict graph.

$$z_k \in \{0, 1\}, \qquad \forall k \in \mathcal{W}. \tag{4.5g}$$

Here, decision variable x_{ki} indicates whether the channel of cellular user k is allocated to D2D group i. Hence, constraint (4.5b) requires that each D2D group be allocated one cellular channel. To avoid interference among D2D groups, constraint (4.5c) ensures that x_{ki} and x_{kj} for D2D groups i and j with a conflict edge (*i.e.*, $\zeta_{ij} = 1$) cannot be set to one at the same time. To distinguish co-channel interference with cellular users, we use ρ_{ki} to represent the conflict between cellular user k and D2D group i. Then, constraint (4.5d) means x_{ki} must be zero when there exists conflict between cellular user k and D2D group i with $\rho_{ki} = 1$, which limits valid cellular channel candidates. Last, (4.5e) defines variable z_k, which is the maximum transmission duration of all D2D groups that are allocated to reuse channel k. In other words, if channel k is allocated to multiple D2D groups, we should consider the longest time that channel k is occupied by any of these D2D groups, *i.e.*, the makespan of the channel occupation time, defined as

$$z_k = \max\{x_{k1}\tau_1, \cdots, x_{ki}\tau_i, \cdots, x_{k\gamma}\tau_\gamma\}. \tag{4.6}$$

Clearly, (4.6) can be represented by inequality constraint (4.5e). Therefore, objective function (4.5a) aims to minimize the total time span that the reused cellular channels are occupied.

4.2 Problem Analysis and Hardness Results

As shown in Section 4.1.1 and Section 4.1.2, we formulate two ILP problems for request screening. In this section, we analyze their computational hardness and prove both problems are NP-hard.

Theorem 1. *The device caching and matching problem in* (4.4) *is NP-hard.*

Proof. To prove that problem (4.4) is NP-hard, we can reduce the well-known NP-hard maximum coverage problem (MCP) to a special instance of problem (4.4). For completeness, the MCP is defined as follows:

$$(MCP) \quad \max. \sum_{j \in \mathcal{U}} y_j \tag{4.7a}$$

$$\text{s.t.} \quad \sum_{S_i \in \mathcal{S}} \varphi_i \leq \gamma \tag{4.7b}$$

$$y_j \leq \sum_{j \in S_i} \varphi_i, \forall j \in \mathcal{U} \tag{4.7c}$$

$$y_j \in \{0, 1\}, \forall j \in \mathcal{U} \tag{4.7d}$$

$$\varphi_i \in \{0, 1\}, \forall S_i \in \mathcal{S}. \tag{4.7e}$$

Here, $\mathcal{U}$ is a universe of elements to be covered, where $|\mathcal{U}| = u$, and $\mathcal{S} = \{S_1, S_2, \cdots, S_g\}$ is a group of subsets of these elements, where $|\mathcal{S}| = g$. The variable φ_i indicates whether subset S_i is selected. If $\varphi_i = 1$, then subset S_i is selected for the cover. The variable y_j represents whether element j is covered by the selected subsets. As seen

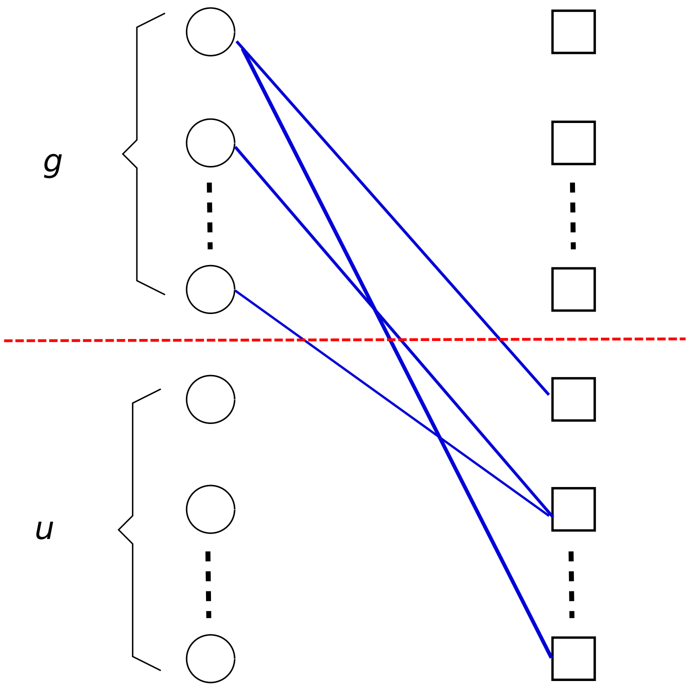


Fig. 4.4: A bipartite graph model for an instance of problem (4.4).

in constraint (4.7c), the relationship between y_j and φ_i can be expressed as

$$y_j = \begin{cases} 0, & \text{if } \sum_{j \in S_i} \varphi_i = 0 \\ 1, & \text{if } \sum_{j \in S_i} \varphi_i > 0. \end{cases} \tag{4.8}$$

The MCP is to determine the decision variables $\{\varphi_i, \forall i \in \mathcal{S}\}$ so that the number of elements covered by the selected subsets is maximized.

Next, we map the MCP to the device caching and matching problem in (4.4) to construct an instance:

- First, for the bipartite graph defined by $\{l_{ij}, \forall i, j \in \mathcal{N}\}$ for problem (4.4), we consider an $n \times n$ bipartite graph with $n = (g + u)$, as shown in Fig. 4.4. Note that edges only exist between the first g left nodes and the last u right nodes. For each subset S_i in the MCP, each left node i is only connected to the subset of the right nodes included in S_i. As such, all subsets in $\mathcal{S}$ are incorporated by constraint (4.4e).

- Then, we map y_j in the MCP to $\sum_{i\in\mathcal{N}} x_{ij}$ in problem (4.4). Hence, constraint (4.4b) naturally implies $y_j \in \{0,1\}$ in (4.7d). The objective function in the MCP becomes equivalent to that of problem (4.4).

- In addition, we set $\beta_i = u$ so that constraint (4.4c) is always satisfied. This is because a left node in the bipartite graph in Fig. 4.4 can be matched to at most u nodes on the right.

- Next, we map φ_i in the MCP to z_i in problem (4.4). That is, selecting subset i with $\varphi_i = 1$ implies that left node i is selected as a source with $z_i = 1$. Then, all the right nodes connected to left node i (contained in subset S_i) are covered. Also, constraint (4.7b) becomes equivalent to constraint (4.4d).

- Last, variables y_j and φ_i in the MCP are related according to (4.7c), which implies the conditional constraint in (4.8). This means that, when element j is covered, at least one subset that contains this element must have been selected. Constraints (4.4f) and (4.4g) define the same relationship between $\sum_{i\in\mathcal{N}} x_{ij}$ and z_i. Furthermore, since we define a bipartite graph in Fig. 4.4, constraint (4.4h) is always satisfied. That is, when $z_j = 1$, it is not possible that the corresponding right node j is covered by any selected source since there is no edge incident on node j. On the other hand, if a right node is covered in the resulting matching, the corresponding left node should never be selected as a source because it does not have any outgoing edge.

Based on the above steps, we can reduce the MCP to an instance of problem (4.4). Since the MCP is known to be NP-hard, problem (4.4) at least has the same complexity as the MCP. Therefore, we conclude that the device caching and matching problem in (4.4) is also NP-hard.

$\square$

Theorem 2. *The D2D channel allocation problem in* (4.5) *is NP-hard.*

Proof. To prove problem (4.5) is NP-hard, we reduce the NP-hard vertex coloring problem (VCP) to a special case of (4.5). The VCP aims to use the minimum number of colors to mark a graph's vertices such that no two adjacent vertices are of the same color. Mathematically, the VCP can be formulated as

$$(VCP) \quad \min . \sum_{k \in \mathcal{W}} y_k \tag{4.9a}$$

$$\text{s.t.} \quad \sum_{k \in \mathcal{W}} x_{ki} = 1, \qquad \forall i \in \mathcal{V} \tag{4.9b}$$

$$x_{ki} + \zeta_{ij} x_{kj} \leq 1, \qquad \forall k \in \mathcal{W}, \forall i, j \in \mathcal{V} \tag{4.9c}$$

$$x_{ki} \in \{0, 1\}, \qquad \forall k \in \mathcal{W}, \forall i \in \mathcal{V} \tag{4.9d}$$

$$y_k \in \{0, 1\}, \qquad \forall k \in \mathcal{W}. \tag{4.9e}$$

Here, decision variable y_k indicates whether color $k \in \mathcal{W}$ is used, variable x_{ki} indicates whether color k marks vertex $i \in \mathcal{V}$, and parameter ζ_{ij} represents whether vertices i and j are adjacent in the graph.

Comparing problem (4.9) with the VCP in (4.5), we can map vertex set $\mathcal{V}$ in (4.9) to D2D group set $\mathcal{U}_t$ in (4.5), and decision variable y_k in (4.9) to z_k in (4.5). Then, it is easily seen that (4.5a) and (4.5b) are the same as (4.9a) and (4.9b), respectively. In addition, considering the special case with $\tau_i = 1, \forall i$, we have constraints (4.5c) and (4.5e) merged to one constraint as defined in (4.9c). Also, assume that all cellular users are so far away from D2D groups that no interference occurs between cellular users and D2D users. In other words, all cellular channels in $\mathcal{W}$ are valid, so constraint (4.5d) is naturally satisfied. As such, we construct an instance of problem (4.5) that is equivalent to the VCP in (4.5). Since the VCP is known to be NP-hard, problem (4.5) at least has the same complexity as the VCP and thus is also NP-hard. $\square$

4.3 Problem Solutions

In this section, we propose efficient algorithms for the two NP-hard problems formulated in Section 4.2. We also give an upper-bound solution to device caching and matching based on Lagrangian relaxation.

4.3.1 Upper-Bound Solution to Device Caching and Matching

As proved in Section 4.2, the device caching and matching problem is NP-hard. Although some small-scale instances can be solved according to the ILP formulation by solvers such as SCPSolver [35], it is computationally infeasible when a large number of user devices are associated with each edge server. Therefore, a heuristic algorithm is more appropriate for large-scale instances. Moreover, for comparison purpose, we also derive an upper bound by Lagrangian relaxation. When the optimal solution is intractable, we can refer to this upper bound to evaluate the performance of the heuristic algorithm.

Consider relaxing the difficult constraints (4.4f), (4.4g), and (4.4h) in problem (4.4). Then, the *Lagrangian dual* is written as follows:

$$
\begin{aligned}
(D) \quad \min_{\mu,\nu,\eta}. \quad C(\mu,\nu,\eta) \triangleq \max_{x_{ij},z_i}. & \textstyle\sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{U}} x_{ij} \\
& + \textstyle\sum_{i\in\mathcal{U}} \mu_i \left(\sum_{j\in\mathcal{U}} x_{ij} - 1 + M\cdot(1-z_i)\right) \\
& + \textstyle\sum_{i\in\mathcal{U}} \nu_i \left(M\cdot z_i - \sum_{j\in\mathcal{U}} x_{ij}\right) \\
& + \textstyle\sum_{j\in\mathcal{U}} \eta_j \left(1 - z_j - \sum_{i\in\mathcal{U}} x_{ij}\right)
\end{aligned} \tag{4.10a}
$$

$$
\text{s.t.} \quad \sum_{i\in\mathcal{U}} x_{ij} \le 1, \forall j \in \mathcal{U} \tag{4.10b}
$$

$$
\sum_{j\in\mathcal{U}} x_{ij} \le \beta_i, \forall i \in \mathcal{U} \tag{4.10c}
$$

$$\sum_{i \in \mathcal{U}} z_i \leq \gamma \tag{4.10d}$$

$$x_{ij} \leq l_{ij}, \forall i, j \in \mathcal{U} \tag{4.10e}$$

$$x_{ij} \in \{0, 1\}, \forall i, j \in \mathcal{U} \tag{4.10f}$$

where the *Lagrange multipliers* are $\mu = \{\mu_1, \ldots, \mu_n\} \geq 0$, $\nu = \{\nu_1, \ldots, \nu_n\} \geq 0$, and $\eta = \{\eta_1, \ldots, \eta_n\} \geq 0$.

The *Lagrangian subproblem* $C(\mu, \nu, \eta)$ can be decomposed into two independent subproblems:

$$\begin{aligned} (S_1) \quad \max_{x_{ij}} . \quad & \sum_{i \in \mathcal{U}} \sum_{j \in \mathcal{U}} x_{ij}(1 + \mu_i - \nu_i - \eta_j) \\ \text{s.t.} \quad & \sum_{i \in \mathcal{U}} x_{ij} \leq 1, \forall j \in \mathcal{U} \\ & \sum_{j \in \mathcal{U}} x_{ij} \leq \beta_i, \forall i \in \mathcal{U} \\ & x_{ij} \leq l_{ij}, \forall i, j \in \mathcal{U} \\ & x_{ij} \in \{0, 1\}, \forall i, j \in \mathcal{U} \end{aligned} \tag{4.11}$$

$$\begin{aligned} (S_2) \quad \max_{z_i} . \quad & \sum_{i \in \mathcal{U}} z_i(M \cdot \nu_i - M \cdot \mu_i - \eta_i) \\ \text{s.t.} \quad & \sum_{i \in \mathcal{U}} z_i \leq \gamma \\ & z_i \in \{0, 1\}, \forall i \in \mathcal{U}. \end{aligned} \tag{4.12}$$

Here, subproblem S_1 can be viewed as an instance of the well-known transportation problem. Specifically, the unit gain over each edge e_{ij} of the bipartite graph in Fig. 4.1 is defined by weight $(1 + \mu_i - \nu_i - \eta_j)$, the demand of each left node i is limited by β_i, and the capacity of each right node is just one. As an instance of the transportation problem, S_1 can be solved in polynomial time with the north-west corner method or ILP solvers. On the other hand, subproblem S_2 can be easily solved by ranking n weights $(M \cdot \nu_i - M \cdot \mu_i - \eta_i)$ in a non-descending order and setting z_i to one only if the corresponding weight is positive, until there are no more positive weights or γ of the n variables of z_i have been set to one. After solving S_1 and S_2, we have the optimal solution, denoted by $\{x^*_{ij}\}$ and $\{z^*_i\}$, to the Lagrangian subproblem $C(\mu, \nu, \eta)$ with given Lagrange multipliers μ, ν, and η. The optimal objective value is given by

$$\begin{aligned} C^*(\mu, \nu, \eta) = & \sum_{i \in \mathcal{U}} \sum_{j \in \mathcal{U}} x^*_{ij}(1 + \mu_i - \nu_i - \eta_j) && \text{(4.13a)} \\ & + \sum_{i \in \mathcal{U}} z^*_i (M \cdot \nu_i - M \cdot \mu_i - \eta_i) && \text{(4.13b)} \\ & + \sum_{i \in \mathcal{U}} ((M - 1) \cdot \mu_i + \eta_i). && \text{(4.13c)} \end{aligned}$$

Next, in order to solve the Lagrangian dual in (4.10), we apply the subgradient method to iteratively search for the best Lagrange multipliers that minimizes the objective value $C^*(\mu, \nu, \eta)$. As such, we can obtain the tightest upper bound to the optimal value of primal problem P_1 before Lagrangian relaxation. Based on the subgradient method, the values of all Lagrangian multipliers μ, ν, and η are first randomly initialized, and then updated after the Lagrangian subproblem with the given Lagrangian multipliers is solved in each iteration t as follows:

$$\begin{aligned} \mu_i^{(t+1)} &= \max\left\{0, \mu_i^{(t)} - \alpha^{(t)} \left(\textstyle\sum_{j \in \mathcal{U}} x^*_{ij} - 1 + M \cdot (1 - z^*_i)\right)\right\} \\ \nu_i^{(t+1)} &= \max\left\{0, \nu_i^{(t)} - \alpha^{(t)} \left(M \cdot z^*_i - \textstyle\sum_{j \in \mathcal{U}} x^*_{ij}\right)\right\} \\ \eta_j^{(t+1)} &= \max\left\{0, \eta_j^{(t)} - \alpha^{(t)} \left(1 - z^*_j - \textstyle\sum_{i \in \mathcal{U}} x^*_{ij}\right)\right\} \end{aligned}$$

where $\alpha^{(t)}$ is the step size for iteration t, given by $\alpha^{(t)} = \frac{1}{2^t}$. The iteration is ter-

Algorithm 1: An upper bound for device caching and matching with Lagrangian relaxation.

```
   Input: Device set $\mathcal{U}$, social relationship graph $G_s = (\mathcal{U}, \mathcal{A})$, request set $\mathcal{R}$,
          $\{\beta_i : i \in \mathcal{U}\}$, $\varpi$, $\gamma$, $\epsilon$
   Output: $\tilde{x} = \{\tilde{x}_{ij} : \forall i, j \in \mathcal{U}\}$, total number of covered requests $\psi$
1  Initialize $\{l_{ij} : \forall i, j \in \mathcal{U}\}$ with $G_s$, $\mathcal{R}$, and $\varpi$
2  $C^* \leftarrow +\infty$
3  $t \leftarrow 1, \alpha^{(t)} \leftarrow 1$
4  $\mu^{(t)} \leftarrow \{0, \dots, 0\}, \nu^{(t)} \leftarrow \{0, \dots, 0\}, \eta^{(t)} \leftarrow \{0, \dots, 0\}$
5  while true do
6      Use SCPSolver to obtain an optimal solution $\{x^*_{ij}\}$ and $\{z^*_i\}$ to
       subproblems $S_1$ and $S_2$
7      Compute objective value $C^*(\mu^{(t)}, \nu^{(t)}, \eta^{(t)})$
8      if $C^*(\mu^{(t)}, \nu^{(t)}, \eta^{(t)}) < C^*$ then
9          $C^* \leftarrow C^*(\mu^{(t)}, \nu^{(t)}, \eta^{(t)})$
10         $\tilde{x}_{ij} \leftarrow x^*_{ij}, \forall i, j \in \mathcal{U}$
11     end
12     if $|C^*(\mu^{(t)}, \nu^{(t)}, \eta^{(t)}) - C^*(\mu^{(t-1)}, \nu^{(t-1)}, \eta^{(t-1)})| \leq \epsilon$ then
13         break
14     end
15     Update Lagrange multipliers $\mu^{(t+1)}$, $\nu^{(t+1)}$, and $\eta^{(t+1)}$ according to the
       subgradient method
16     Update step size $\alpha^{(t+1)} \leftarrow \frac{1}{2^(t+1)}$
17     $t \leftarrow t + 1$;
18 end
19 Compute objective value to original problem $P_1$: $\psi \leftarrow \sum_{i \in \mathcal{U}} \sum_{j \in \mathcal{U}} \tilde{x}_{ij}$
20 Return $\tilde{x}, \psi$
```

minated when the gap between $C^*(\mu^{(t)}, \nu^{(t)}, \eta^{(t)})$ and $C^*(\mu^{(t-1)}, \nu^{(t-1)}, \eta^{(t-1)})$ is not more than an accuracy threshold ϵ. The solution with the minimum objective value for the Lagrangian dual provides an upper bound for problem (4.4). Alg. 1 shows the details of solving the Lagrangian dual with the subgradient method.

4.3.2 Heuristic Algorithm for Device Caching and Matching

In Section 4.3.1, since we relax the constraints that limit the relationships between x and z, the upper bound obtained by Lagrangian relaxation may not be feasible with respect to the original problem. Considering that problem (4.4) is NP-hard, we cannot obtain a feasible optimal solution in polynomial time. Hence, we propose a heuristic algorithm in Alg. 2 to find a feasible solution close to the optimum.

The goal of problem (4.4) is to maximize the request coverage through source selection and device matching. In other words, we need to select at most γ sources among n devices and match them with other intended receivers so that we can maximize the total number of video requests that can be fulfilled by device caching. As seen in Alg. 2, we rank all devices based on a heuristic metric h_i, which depends on the resource budget of an individual device i, *i.e.*, β_i, and the number of potential receivers (denoted by set S_i) that each device i can serve. Specifically, $S_i = \{j \in \mathcal{U} : l_{ij} = 1\}$. That is, device j belongs to set S_i only if $l_{ij} = 1$. Accordingly, we define the heuristic metric h_i of device i as

$$h_i = \min\left\{\beta_i, |S_i|\right\}, \forall i \in \mathcal{U}. \tag{4.15}$$

Here, by choosing the minimum value between β_i and $|S_i|$, we obtain the maximum number of requests that device i can serve if it is selected for video caching. Even if device i has a large value of $|S_i|$, it may not be able to share contents with other devices because of very limited transmission power (*i.e.*, $\beta_i = 0$). Similarly, device i cannot fulfill any request if β_i is large but $S_i = \varnothing$. Therefore, we consider both the individual resource budget and the number of available candidates.

In Alg. 2, we divide the solution into two main decision steps, *i.e.*, caching source selection and device matching. First, in Lines 5-16, we select the sources according to their coverage capability (*i.e.*, the heuristic value). Since our goal is to select sources to fulfill as many requests as possible through device content sharing, we prefer to choose the devices that can contribute more to request screening. Hence, the device with the maximum heuristic value is first selected as a source. In contrast, when choosing the receivers, we tend to prioritize the receivers with the fewest options, *i.e.*, to match a source with the receiver that has the minimal capability of content caching and transmitting (see Lines 9-12). In each round, a new source is selected

Algorithm 2: A heuristic social-aware algorithm for device caching and matching.

```
    Input: Device set $\mathcal{U}$, social relationship graph $G_s = (\mathcal{U}, \mathcal{A})$, request set $\mathcal{R}$,
           $\{\beta_i : i \in \mathcal{U}\}$, $\varpi$, $\gamma$
    Output: $\tilde{x} = \{\tilde{x}_{ij} : \forall i, j \in \mathcal{N}\}$, total number of covered requests $\psi$
 1  Initialize $\{l_{ij} : i, j \in \mathcal{U}\}$ with $G_s$, $\mathcal{R}$, and $\varpi$
 2  Initialize $x_{ij} \leftarrow 0, z_i \leftarrow 0, \forall i, j \in \mathcal{U}$
 3  Initialize potential receiver sets $\mathcal{S} = \{S_1, S_2, \ldots, S_n\}$, where $S_i$ includes the
    subset of devices that are incident on the left node $i$ in the bipartite graph
 4  Calculate heuristic value $h_i$ for each device $i \in \mathcal{U}$ and rank all devices in $\mathcal{N}$
    with $|S_i| \geq 1$ to set $\mathbb{U}$ in a non-descending order of $h_i$
 5  begin Select sources for local device caching
 6      while $\sum_{i \in \mathcal{U}} z_i < \gamma$ or $\mathbb{U} \neq \varnothing$ do
 7          Select device $i$ in $\mathbb{U}$ with maximum heuristic value as source, and set
            $z_i \leftarrow 1$
 8          Remove device $i$ from $\mathbb{U}$
 9          while $\sum_{j \in \mathcal{U}} x_{ij} < \beta_i$ or $S_i \neq \varnothing$ do
10              Select device $j$ from potential receiver set $S_i$ that has minimum
                heuristic value as receiver, and set $x_{ij} \leftarrow 1$
                // Device $j$ cannot be a source any more
11              Remove device $j$ from $\mathbb{U}$
12              Remove device $j$ from all sets in $\mathcal{S}$
13          end
14          Remove $S_i$ from $\mathcal{S}$
15          Update heuristic value $h_i$ for each device $i \in \mathbb{U}$
16      end
17  end
18  begin Optimize device matching with selected sources
        // Modify bipartite graph based on selected sources
19      Set $l_{i'j} \leftarrow 0, \forall i', i, j \in \mathcal{U}$, $i' \neq i$ and $z_i = 1$
20      Set $l_{ij} \leftarrow 0, \forall i, j \in \mathcal{N}$, and $z_j = 1$
21      Formulate a transportation problem with the modified bipartite graph,
        where the objective function is $\max. \sum_{i \in \mathcal{U}} \sum_{j \in \mathcal{U}} x_{ij}$ s.t. the constraints
        in subproblem $S_1$
22      Solve the transportation problem to find the optimal matching solution
        $\{\tilde{x}_{ij} : \forall i, j \in \mathcal{U}\}$
23      Compute objective value to original problem $P_1$: $\psi \leftarrow \sum_{i \in \mathcal{U}} \sum_{j \in \mathcal{U}} \tilde{x}_{ij}$
24  end
25  Return $\tilde{x}, \psi$
```

while at most h_i receivers are tentatively matched to the selected source i. It is worth noting that both sources and receivers are chosen from set $\mathbb{U}$, and a device should be removed from $\mathbb{U}$ if it has been selected as a cache device or a receiver. The heuristic values of the devices that remain in set $\mathbb{U}$ are dynamically updated after each iteration.

Second, in Lines 18-23, the selected sources are matched to the potential receivers in an optimal manner. Though the procedure in Lines 5-16 tentatively assigns certain receivers to the selected sources, this matching may not be optimal due to the iterative procedure. Once the sources are determined, we can further obtain the optimal matching by formulating a transportation problem. Here, we modify the original bipartite graph by keeping only the edges incident on the selected sources on the left with $z_i = 1$, and also removing the edges incident on the right nodes corresponding to the selected sources. As such, we can ensure that the resulting matching can only be from a source device to a non-source receiver. Based on the modified bipartite graph, the transportation problem aims to maximize the number of covered requests (*i.e.*, $\sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{U}} x_{ij}$) subject to the constraints of subproblem S_1 in (4.11). Then, the optimal matching with the sources can be obtained by solving the transportation problem optimally.

4.3.3 Heuristic Algorithm for D2D Channel Allocation

As proved in Section 4.2, the D2D channel allocation problem can be viewed as a generalized VCP and thus is NP-hard. Since there does not exist a polynomial-time algorithm to find the optimal solution, we propose a heuristic algorithm in Alg. 3 to obtain a feasible and efficient solution. According to problem (4.5), since we aim at minimizing the total channel occupation time, a source device with a long service duration can be prioritized to select channel first. Otherwise, if a source device with a really long data transmission time is assigned a channel very late, it is likely that there would be fewer channel options and the total time span may be prolonged significantly. Furthermore, when selecting channels for source devices, the channel

Algorithm 3: A heuristic algorithm for D2D channel allocation.

```
Input: Set of D2D groups $\mathcal{U}_t$, set of cellular users $\mathcal{W}$, $\varpi$, transmission
       durations $\{\tau_i : i \in \mathcal{U}_t\}$
Output: $x = \{x_{ki} : \forall k \in \mathcal{W}, \forall i \in \mathcal{U}_t\}$, total channel occupation time $\xi$
1  Initialize conflict graph $G_c$ among D2D groups and cellular users, configure
   $\{\zeta_{ij} : \forall i, j \in \mathcal{U}_t\}$ and $\{\rho_{ki} : \forall k \in \mathcal{W}, \forall i \in \mathcal{U}_t\}$
2  Rank all source devices in $\mathcal{U}_t$ to set $\mathbb{U}_t$ in a descending order of $\{\tau_i : i \in \mathcal{U}_t\}$;
   and if $\tau_i = \tau_j$, then rank them in a non-descending order of their vertex
   degrees in conflict graph $G_c$ with only D2D groups; Rank all cellular users in
   $\mathcal{W}$ to set $\mathbb{W}$ in a non-descending order of their vertex degrees in conflict
   graph $G_c$
3  begin Allocate channels for all source devices
4      while $\mathbb{U}_t \neq \varnothing$ do
5          Select top-ranked source device $i$ in $\mathbb{U}_t$
6          Assign to source device $i$ the top-ranked cellular channel $k$ without
           interference with $i$
7          $x_{ki} \leftarrow 1$
8          Remove source device $i$ from set $\mathbb{U}_t$: $\mathbb{U}_t \leftarrow \mathbb{U}_t \setminus \{i\}$
9          Initialize set $\mathcal{F} \leftarrow \{i\}$            // Source device $i$ is fulfilled
10         for All unfulfilled source devices in $\mathbb{U}_t$ do
11             Select next sorted source device $j$ in $\mathbb{U}_t$
12             if Device $j$ has no interference with cellular user $k$ and all source
               devices in $\mathbb{F}$ then
13                 $x_{kj} \leftarrow 1$      // Source device $j$ is assigned channel $k$ as well
14                 Remove source device $j$ from set $\mathbb{U}_t$: $\mathbb{U}_t \leftarrow \mathbb{U}_t \setminus \{j\}$
15                 Add source device $j$ to set $\mathbb{F}$: $\mathcal{F} \leftarrow \mathcal{F} \cup \{j\}$
16             end
17         end
18         Calculate $z_k$: $z_k = \max\{x_{ki} \cdot \tau_i, \forall i \in \mathbb{F}\}$
19         Remove cellular channel $k$ from $\mathbb{W}$: $\mathbb{W} \leftarrow \mathbb{W} \setminus \{k\}$
20     end
21 end
22 Calculate objective value $\xi$: $\xi \leftarrow \sum_{k \in \mathcal{W}} z_k$
23 Return $x, \xi$
```

of the cellular user who has the lowest vertex degree in the conflict graph has priority because it has the smallest probability of having interferences with source devices. Therefore, at the beginning of Alg. 3 in Lines 1-2, we rank all source devices based on their required data transmission time, and rank all cellular users according to their vertex degrees in the conflict graph.

After that, the channel allocation procedure is divided into two main steps. First, in Lines 3-20, we consider the source device that has not been assigned to any cellular channel and has the maximum required occupation time. Then, we iterate the ranked cellular user set, find the first cellular user without interference with the source device, and assign this cellular channel to the source. This source device is then removed from ranked set $\mathbb{U}_t$, but recorded in another temporary set $\mathbb{F}$, which maintains the devices fulfilled in the current iteration by a selected cellular channel. Since we have started to reuse a new cellular channel for a source device, it would be efficient to assign more remaining source devices to share the same channel if possible. In Lines 9-17, we iterate the remaining source devices in set $\mathbb{U}_t$. As long as an unfulfilled source device has no interference with the currently selected cellular user and the source devices already in temporary set $\mathbb{F}$, the unfulfilled source device is assigned to share the same channel as others in $\mathbb{F}$. Meanwhile, $\mathbb{F}$ is updated to include the newly assigned source device, while this fulfilled device is removed from $\mathbb{U}_t$. As seen, through the two steps in each iteration, we attempt to make most use of a selected cellular channel and accommodate as many D2D devices as possible with it. After that, this cellular channel is removed from the candidate set $\mathbb{W}$. This channel allocation procedure repeats until all source devices are fulfilled and set $\mathbb{U}_t$ is cleared.

4.4 Simulation Results and Discussions

This section evaluates the performance of the heuristic algorithms described in Section 4.3.2 and Section 4.3.3 under various problem scales, collaboration distances, budgets of edge sever and user devices, and social relationship densities.

4.4.1 Simulation Settings

We develop a simulator in Java to evaluate the performance of the proposed request screening solution. In our experiments, we compare the proposed heuristic approach in Alg. 2 with a greedy algorithm and the optimal solution in small-scale cases. For a large-scale scenario where the optimal solution is intractable, we consider the upper bound derived from Lagrangian relaxation as described in Section 4.3.1. Although the upper-bound solution is not always feasible since certain constraints are relaxed, we can still include it as a benchmark in the absence of the optimal solution. The greedy approach makes a local optimal decision at each step as it attempts to solve the entire problem. The greedy approach ranks all user devices according to the number of potential receivers. Then, it always picks the device that covers the maximum number of uncovered requests as a cache until γ devices are selected. Different from the proposed heuristic approach, the greedy approach randomly assigns at most β_i receivers to a newly selected cache device i to fulfill their video requests.

For the heuristic approach in Alg. 3, we also compare it with a greedy algorithm and the optimal solution. The greedy algorithm is based on the idea of coloring vertices one by one iteratively [36]. In each round, it picks an uncolored vertex according to some ordering and assigns to it the lowest numbered color that has not been used by any previously colored vertices adjacent to the vertex. If all previously used colors appear on vertices adjacent to the vertex, a new unused color is assigned.

4.4.2 Performance of Device Caching and Matching

4.4.2.1 Impact of Different Schemes

In order to assess the performance of our proposed heuristic algorithm for the request screening problem, it is necessary to compare it with the optimal solution. Although existing LP solvers is not able to provide the optimal solution with a large number of variables, the optimal solution can be obtained when the number of UEs is relatively small. We first consider a small-scale scenario, where 30 UEs are randomly located around the BS. The collaboration distance ϖ between two devices is set to 100m, and the maximum number of user requests that each device can serve is set between 0 and 5 (*i.e.*, $\beta_i \in [0, 5]$, $\forall i \in \mathcal{U}$). The total number of selected sources is set to 7, which means that at most 7 devices can be selected to cache video contents. Each user has social relationships with all other users. We run 50 rounds of experiments and the locations of all UEs are randomly adapted for each round.

The total number of video requests that can be fulfilled by device caching with different solutions is shown in Fig. 4.5. As observed in the results, the heuristic scheme always outperforms the greedy approach in all 50 rounds of experiments. On the other hand, compared to the optimal solution, the heuristic algorithm achieves the optimal result in 43 rounds totally, while its objective values are only 1 less than those of the optimal solution in the other 7 rounds. Based on the above results, we can see that our proposed solution performs fairly closely to the optimal solution in the small-scale cases.

We further consider a larger-scale network where there are 100 UEs with the same collaboration distance as above. The maximum number of user requests served by each device is set between 0 and 10, and at most 10 source devices can be selected. As shown in Fig. 4.6, the gap between the proposed heuristic scheme and the greedy approach becomes much larger in each round. On average, 68% of video requests can be fulfilled through device caching under the heuristic scheme, while the greedy approach only offloads 46% of the video requests to D2D transmissions. Moreover, since the Lagrange relaxed solution provides an upper bound for the intractable optimal solution, we can see that the heuristic scheme achieves at least 70% of the performance of the optimal solution.

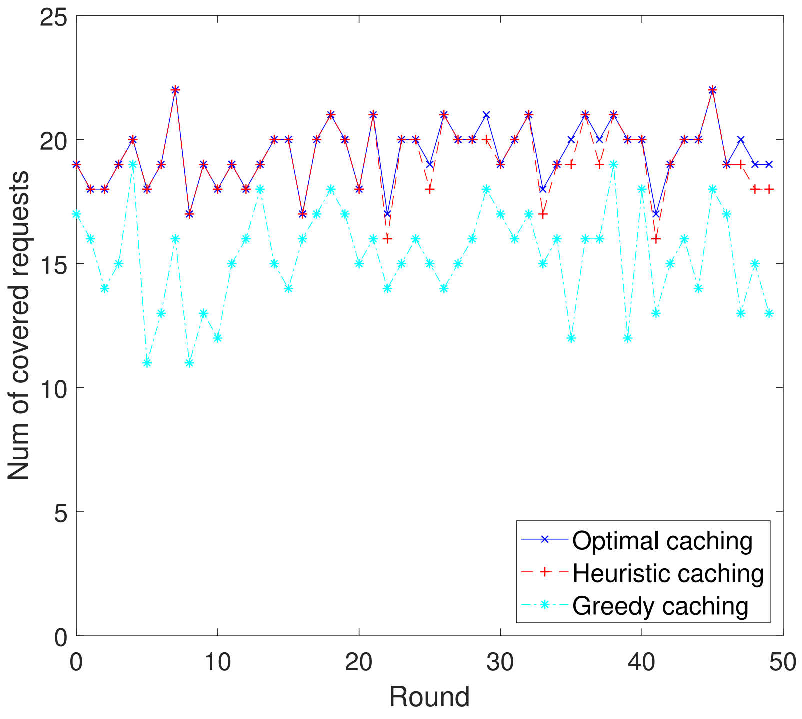


Fig. 4.5: Performance of device caching and matching schemes in a small-scale network.

4.4.2.2 Impact of Collaboration Distance

For the simulation of this section, we assume all parameters follow the same setting as given in Section 4.4.2.1. Fig. 4.7 illustrates the performance of request screening with different collaboration distances ϖ. We can see that our proposed scheme always outperforms the greedy approach. With the increase of the collaboration distance ϖ, the gap between the two schemes becomes larger, especially when $\varpi \geq$ 43m. This is because, when we have a larger ϖ, the edge server has more different options for cache device selection, *i.e.*, as for which devices should be chosen to cache video contents and deliver them to other users. On the contrary, when there is a smaller collaboration distance, each candidate device has a lower capability of sharing contents with other devices, since they can only transmit videos to other users located in the near vicinity. Hence, the results show little difference under short

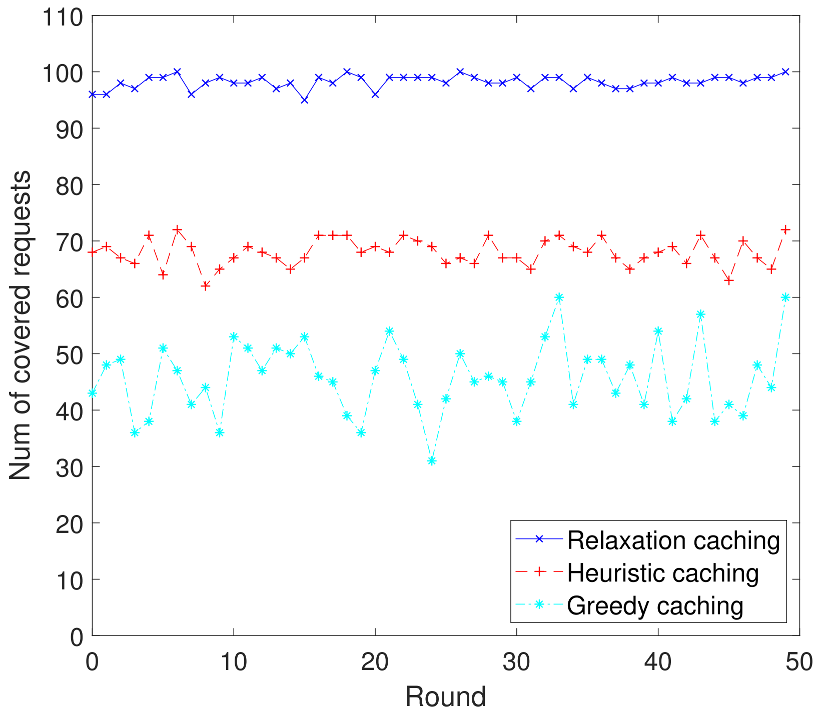


Fig. 4.6: Performance of device caching and matching schemes in a large-scale network.

collaboration distances. In contrast, when it is affordable to share contents between distant users, the proposed algorithm can provide a more intelligent strategy of device caching selection. Moreover, it is observed that the proposed scheme follows the trend of the relaxed upper-bound solution, whereas the greedy approach shows more fluctuations. This demonstrates that the heuristic scheme is more reliable and stable than the greedy approach.

4.4.2.3 Impact of Resource Budget of Individual Device

In practice, not all UEs have the same capability of sharing video contents with others due to various concerns (*e.g.*, battery life, private information security, and device caching capacity). Hence, we take into account the factor $\{\beta_i, \forall i \in \mathcal{U}\}$ in order to represent the resource constraint of each individual device for content caching and D2D sharing. Assume that the resource budgets of all devices follow a uniform

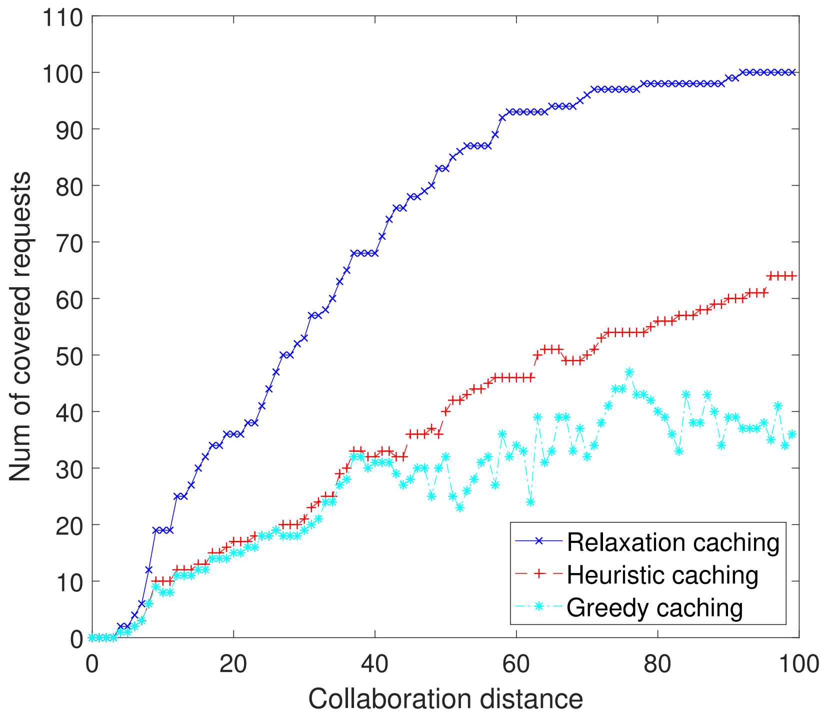


Fig. 4.7: Performance variation with collaboration distance ϖ.

distribution within a range of $[0, b]$, where b is the maximum value of β_i. In this experiment, we vary b from 0 to 20 while keeping the other parameters for the large-scale network described in Section 4.4.2.1. As shown in Fig. 4.8, the proposed algorithm always outperforms the greedy method in terms of the number of covered requests. In other words, our heuristic solution can offload more traffic to the user plane whether the UEs have rich resources or not, since it is always able to identify stronger devices to cache the video contents and share them with weaker devices.

4.4.2.4 Impact of Transmitter Density

To show how transmitter density affects requests coverage, we consider the scenario with 100 UEs as described in Section 4.4.2.1 . Fig. 4.9 shows how the performance of the two schemes varies with different values of γ. As seen, compared to the greedy approach, the proposed scheme can cover nearly 17 more video requests on average

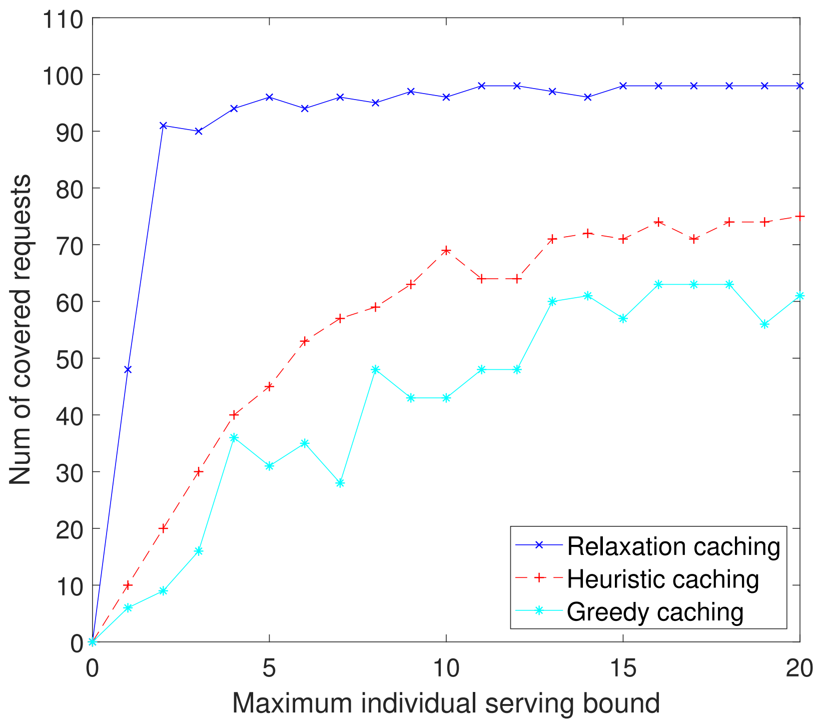


Fig. 4.8: Performance variation with maximum budget β_i.

while selecting the same number of transmitters. With the heuristic scheme, the number of covered requests increases significantly at the beginning, while after that the growth becomes flattened gradually. Furthermore, we can observe in Fig. 4.9 that the performance growth of the proposed algorithm is much smoother than that of the greedy solution. In particular, the performance increase is not continuous with the greedy solution, which indicates that selecting more transmitters does not always guarantee a better coverage result. For example, when $\gamma = 6$, the greedy solution can cover 31 video requests, while it can only cover 27 requests even when $\gamma = 7$.

4.4.2.5 Impact of Social Relationship Density

To examine the impact of social relationships on the performance, we consider an Erdos-Renyi model when simulating the social network graph among UEs. According

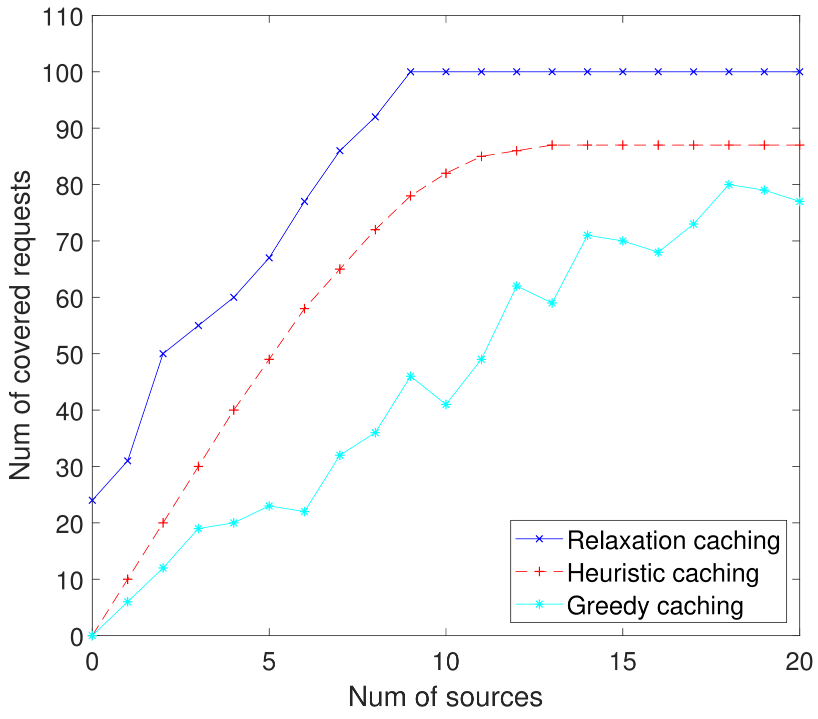


Fig. 4.9: Performance variation with D2D capacity limit γ.

to the Erdos-Renyi model, a social relationship graph is constructed by connecting users randomly. Each edge is included in the graph with probability p independent of any other edge. In this test, we vary the connection probability p following a discrete set $p \in [0.0, 0.1, 0.2, \ldots, 1.0]$. Thus, at each simulation round, a new social network with a different density is used to examine the performance of the two schemes. As shown in Fig. 4.10, when the social connection probability is not more than 0.2, the gaps between the two schemes are negligible. However, when the social network density further increases, the greedy method can no longer keep up with the performance of the proposed scheme. For instance, the greedy approach can only fulfill 47 video requests through device caching when every group of users has a social connection, while the proposed algorithm achieves a much larger objective value of 70. Moreover, Fig. 4.10 also shows that the greedy scheme fluctuates more dramatically when the social network density is higher.

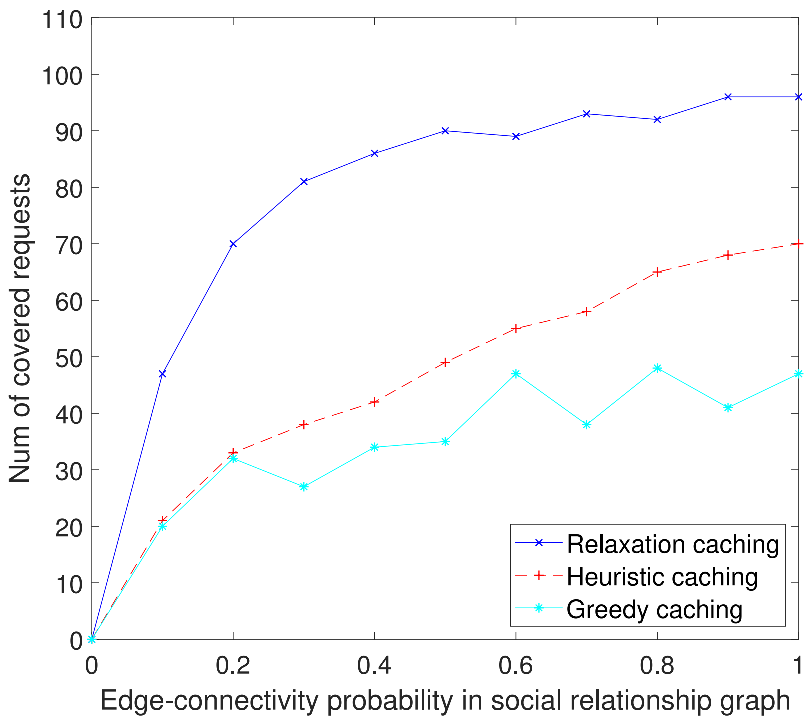


Fig. 4.10: Performance variation with social relationship density.

4.4.2.6 Comparison of Computation Time

Considering the different complexities of the proposed heuristic scheme and greedy approach, we also set an experiment to compare the computation time. In this test, we follow all the parameter settings in Section 4.4.2.1, and record the computation time of each round with the heuristic algorithm and greedy scheme. As shown in Fig. 4.11, the gaps between the two schemes are considerable. With the proposed scheme, after round 7, the running time fluctuates between 55ms and 77ms. In contrast, the computation time of the greedy scheme drastically decreases after the first round and maintains at 3ms and below after the 7th round. The proposed algorithm is more complex than the greedy method and costs more time to obtain the screening result, since it simultaneously considers the transmitter side and the receiver side. In addition, solving the transportation problem to augment the device matching in the post-processing step needs more computation time, especially when the number of

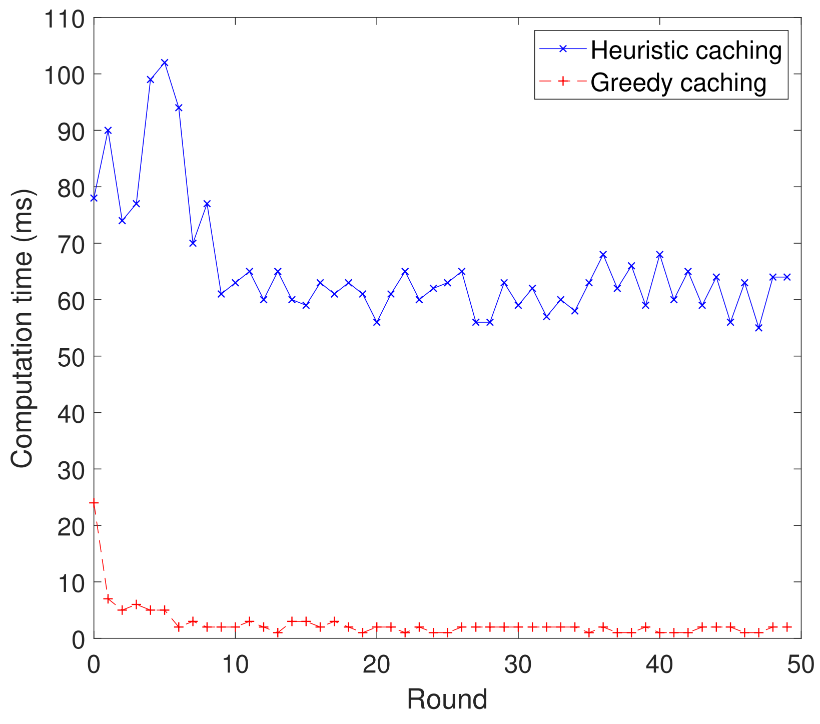


Fig. 4.11: Computation time with different schemes.

variables grow. In view of the effectiveness of the heuristic algorithm shown above, it is reasonable to sacrifice some computation time to offload a considerable amount of traffic from the core network to device caching and D2D communications.

4.4.3 Performance of D2D Channel Allocation

4.4.3.1 Small-Scale Scenario

By device caching and matching, selective requests can be offloaded from the BS and fulfilled by D2D communications. To improve spectrum efficiency, the D2D links can reuse the channels of regular cellular users. We propose Alg. 3 for D2D channel allocation, aiming at minimizing the total channel occupation time. This heuristic algorithm is compared with the optimal solution and the greedy algorithm for a small-scale network, where 30 UEs are randomly located around the BS within a circular area of radius 100m. The transmission range of each device is set to 30m.

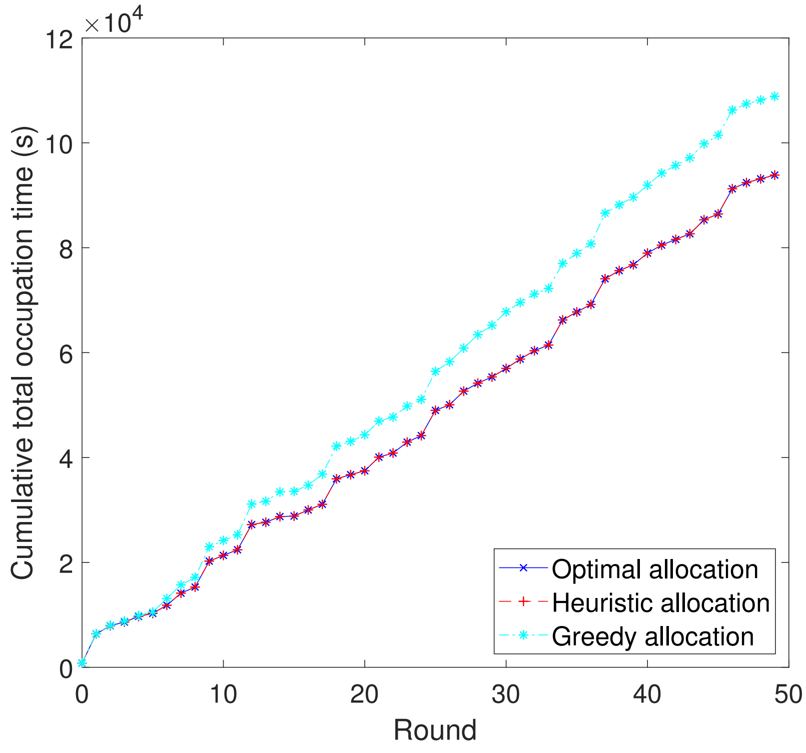


Fig. 4.12: Cumulative total occupation time of D2D channel allocation schemes in a small-scale network with random spatial distribution of UEs.

The D2D transmission time required for each request ($i.e.$, $\tau_i, \forall i$) follows a Lévy distribution with a location parameter 0.0 and a scale parameter 50.0. We run 50 rounds of simulations and the locations of all UEs are randomly assigned for each round.

Fig. 4.12 shows the cumulative total channel occupation time with different channel allocation schemes. As seen, the proposed heuristic algorithm achieves the optimal solution in all 50 simulation rounds, which indicates the efficiency of the proposed scheme in the small-scale network. In contrast, the greedy scheme ends up with 16% more total occupation time. Fig. 4.13 further shows the number of channels allocated for D2D transmission in each round. We can observe that the heuristic scheme uses the same number of channels for D2D communications as the optimal solution in each round, whereas the greedy scheme uses 1 more channel in 6 rounds. Since the goal of channel allocation is to minimize the total channel occupation time,

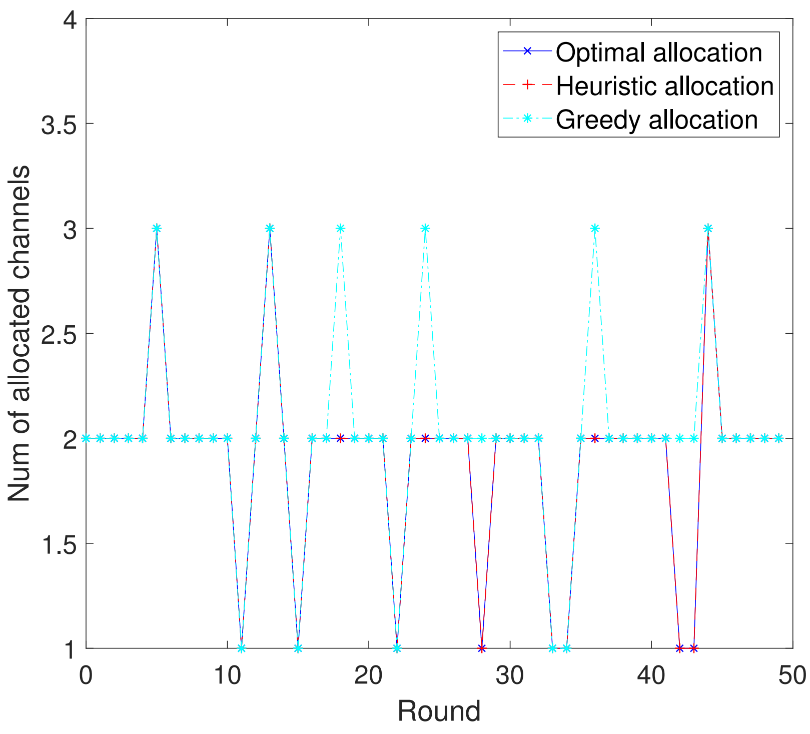


Fig. 4.13: Number of allocated channels of D2D channel allocation schemes in a small-scale network with random spatial distribution of UEs.

allocating fewer channels generally helps lower the total occupation time.

4.4.3.2 Large-Scale Random Scenario

We further evaluate the proposed channel allocation solution in a large-scale network, where 500 UEs are randomly located around the BS within a circular area of radius 200m. The D2D transmission duration of each request follows a Lévy distribution with a location parameter 0.0 and a scale parameter 30.0.

Fig. 4.14 shows the total channel occupation time in each simulation round. The results show that the proposed heuristic algorithm outperforms the greedy scheme in 49 rounds and performs same as the greedy scheme in 1 round. Specifically, the heuristic scheme achieves approximately 27% less total channel occupation time in each round than the greedy scheme on average. The results in Fig. 4.15 show the cumulative channel occupation time, which can clearly indicate the effectiveness of the

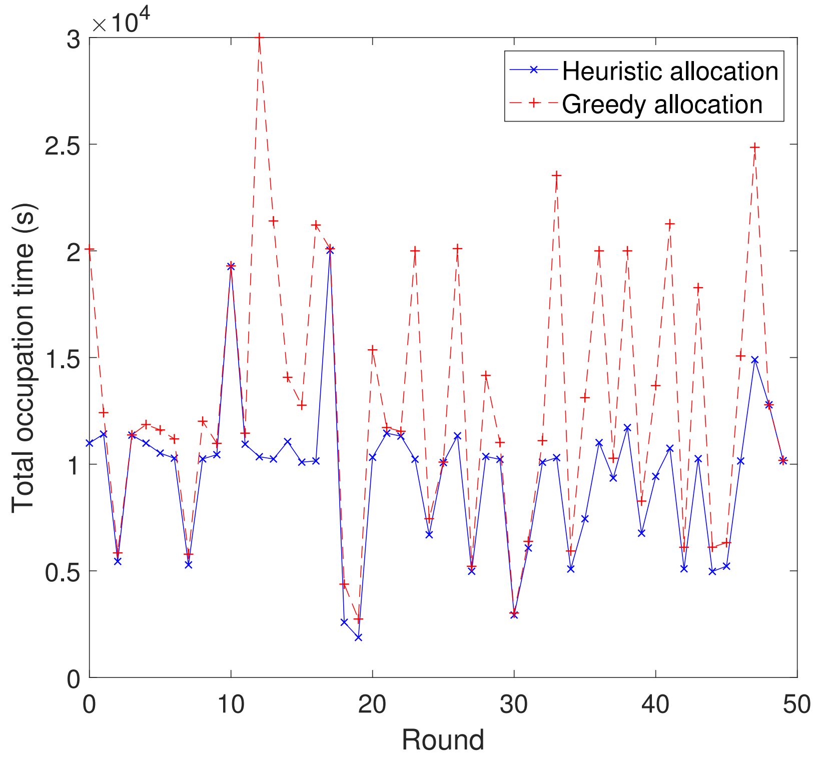


Fig. 4.14: Total channel occupation time of each round in a large-scale network with random spatial distribution of UEs.

heuristic channel allocation solution. According to the results, the heuristic allocation scheme reduces more than 24% channel occupation time for D2D transmission, which significantly saves spectrum resources for other wireless services while meeting the content distribution requirements among devices.

Fig. 4.16 shows the number of channels allocated for D2D transmission in each round. We can observe that the heuristic scheme uses the same number of or fewer channels for D2D communications compared to the greedy scheme except the 21st round. The number of D2D channels allocated per round with the heuristic scheme is 0.7 less than that with the greedy scheme. Generally, fewer channels can achieve lower total occupation time. However, the heuristic solution may use more channels if a lower total channel occupation time can be achieved in that way.

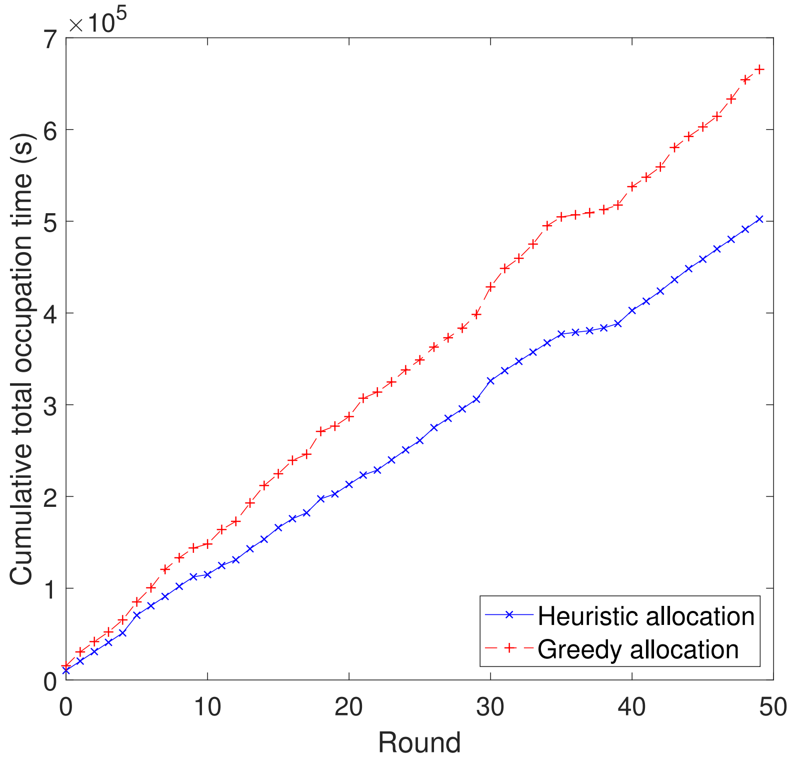


Fig. 4.15: Cumulative of total occupation time in a large-scale network with random spatial distribution of UEs.

4.4.3.3 Large-Scale Clustered Scenario

We also compare the channel allocation schemes in another large-scale network, where 500 UEs follow a Matérn cluster point spatial distribution, with 10 clusters and 50 devices in each cluster. The radius of the circular area is 200m, and the radius of each cluster is set to 30m, which is same as the device transmission range. The D2D transmission duration of each request follows the same Lévy distribution that used in Section 4.4.3.2.

Fig. 4.17 shows the total channel occupation time in each simulation round. As seen, the proposed heuristic algorithm outperforms the greedy scheme in all 50 rounds. Specifically, the heuristic scheme occupies the allocated channels for approximately 36% less time than the greedy scheme on average. Fig. 4.18 further shows the cumulative channel occupation time. A larger gap is seen clearly between two schemes

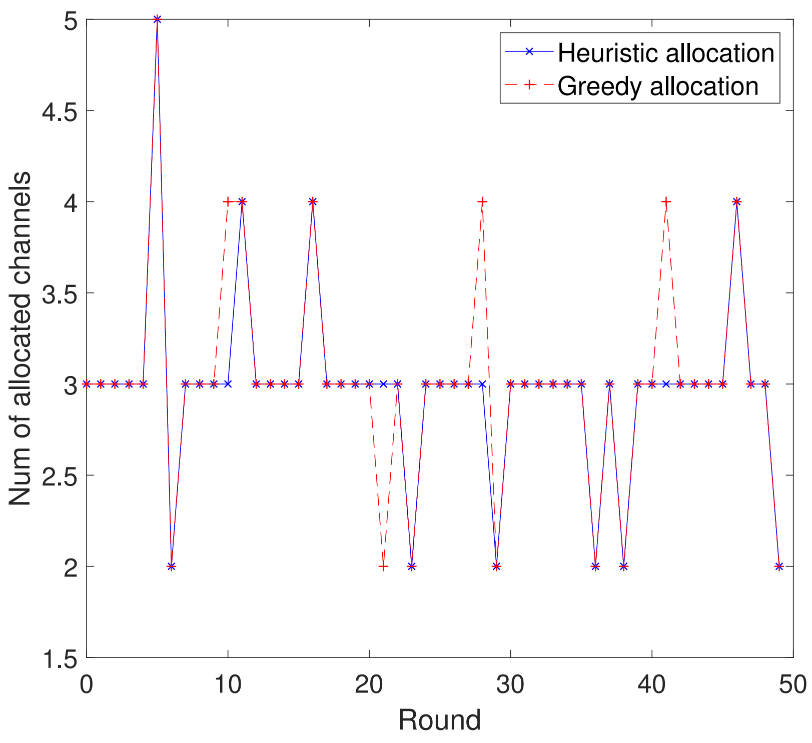


Fig. 4.16: Number of allocated channels of D2D channel allocation schemes in a large-scale network with random spatial distribution of UEs.

in the large-scale clustered network in comparison with the small-scale randomly distributed network. In total, the heuristic allocation scheme reduces nearly 38% channel occupation time for D2D transmission.

Fig. 4.19 shows the number of channels allocated for D2D transmission in each round. In accordance with the results, the heuristic scheme uses the same number of channels for D2D communications as the greedy scheme in 46 rounds in total. In 3 rounds, the heuristic scheme uses 1 less channel than the greedy method, while allocates 1 more channel in 1 rounds. Compared to the results with random spatial distribution in Section 4.4.3.2, the heuristic scheme achieves better performance and saves more channel occupation time than the greedy scheme when the geographical locations of the UEs follow a clustered spatial distribution.

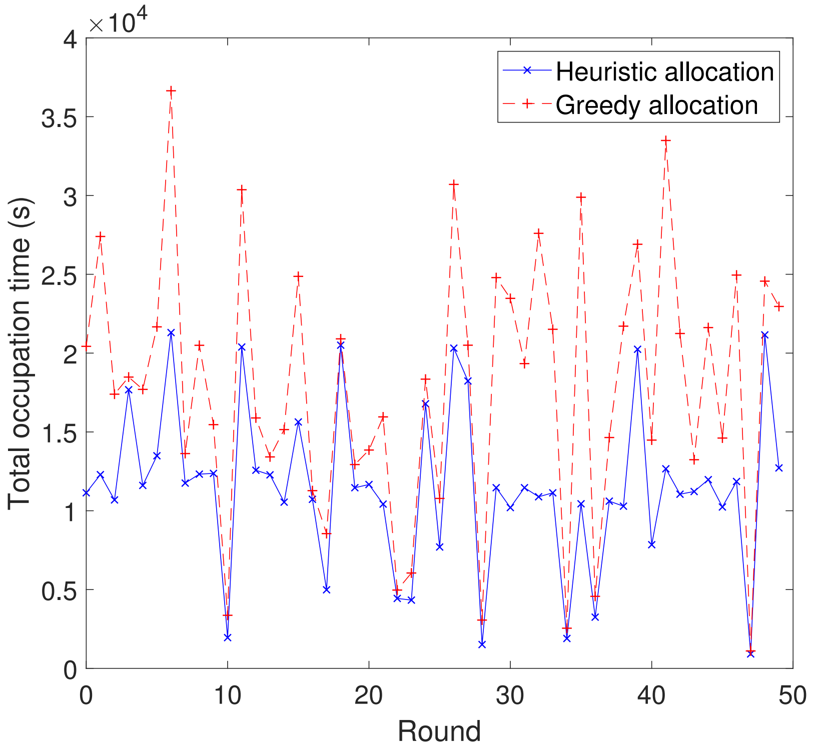


Fig. 4.17: Total channel occupation time of each round in a large-scale network with clustered spatial distribution of UEs.

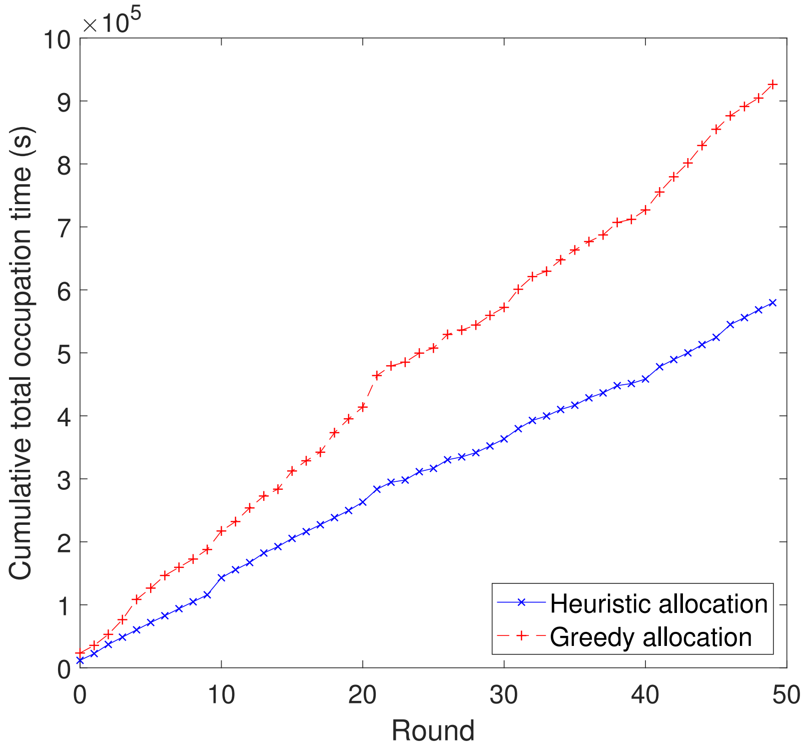


Fig. 4.18: Cumulative of total occupation time in a large-scale network with clustered spatial distribution of UEs.

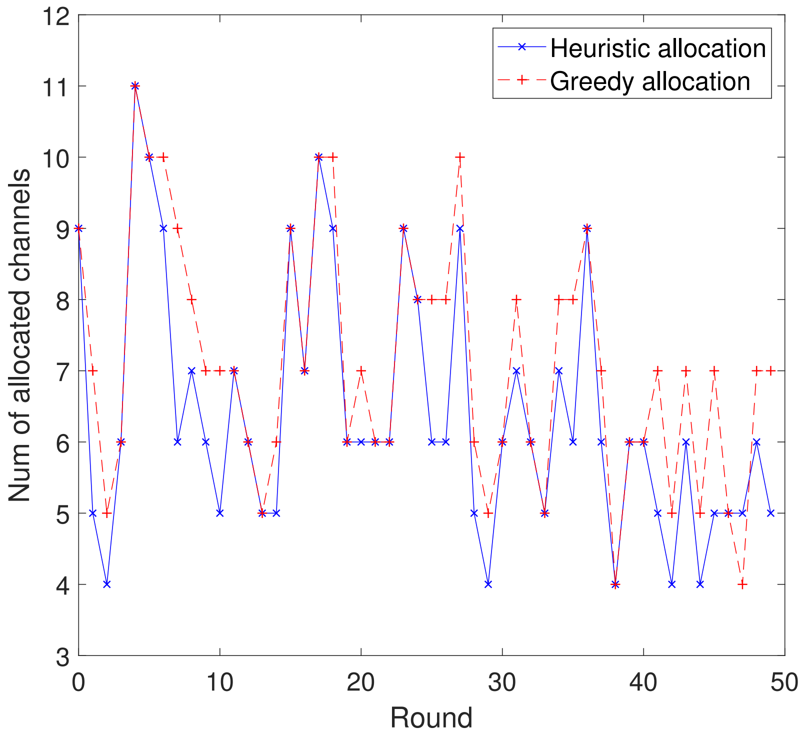


Fig. 4.19: Number of allocated channels of D2D channel allocation schemes in a large-scale network with clustered spatial distribution of UEs.

Chapter 5

Joint Source Selection and Flow Routing

After the request screening results are obtained, all requests that are unfulfilled by local device caching should fetch contents from in-network caching, such as the BS, the neighboring BSs, or even the source points in the core network. This chapter investigates the joint source selection and flow routing problem described in Section 1.2.2, and then proposes a context-aware solution.

5.1 Problem Formulation

As described in Section 3.2.3, the controller selects sources for all remaining requests that cannot be fulfilled by local device caching, and makes appropriate flow routing decisions to maximize the minimum buffer time, while satisfying link capacities. Taking into account user context information, we formulate this joint problem as

$$\max. \quad \min_{(i,k)\in\mathcal{R}'} T_i^k \tag{5.1a}$$

$$\text{s.t.} \quad \sum_{P:e\in P} x(P) \le \lambda^* \cdot c(e), \quad \forall e \in \mathcal{E} \tag{5.1b}$$

$$\sum_{P \in \mathcal{P}_i^k} x(P) \geq d_i^k, \quad \forall (i,k) \in \mathcal{R}' \tag{5.1c}$$

$$x(P) \geq 0, \quad \forall P \in \mathcal{P} \tag{5.1d}$$

where $\mathcal{P}$ is the set of all paths for every pair of nodes in the network, $\mathcal{P}_i^k$ is the set of available paths from a source point that caches the requested video k to serve request (i,k), and $x(P)$ is the flow amount over path P. To bound the congestion level, the first constraint restricts that each link e can be limited to use up to λ^* fraction of its capacity for video delivery. The second constraint requires that the demand rate for each request (i,k) be fulfilled. The design objective is to maximize the minimum buffer time among all active video sessions. Note that this problem can also be presented by an equivalent polynomial-size edge-flow formulation without path enumeration for $\mathcal{P}$.

Problem (5.1) is a linear program (LP) problem. However, the problem size can be extremely large if there are a large number of video requests. Here, we solve it based on a fast approximation algorithm for a related traffic engineering problem as follows:

$$\min. \quad \lambda \tag{5.2a}$$

$$\text{s.t.} \quad \sum_{P: e \in P} x(P) \leq \lambda \cdot c(e), \quad \forall e \in \mathcal{E} \tag{5.2b}$$

$$\sum_{P \in \mathcal{P}_i^k} x(P) \geq d_i^k, \quad \forall (i,k) \in \mathcal{R}' \tag{5.2c}$$

$$x(P) \geq 0, \quad \forall P \in \mathcal{P}. \tag{5.2d}$$

Problem (5.2) aims to minimize the maximum link utilization λ, *i.e.*, the ratio of traffic load to link capacity. Since the optimal value of λ can exceed 1, this implies that the video demands $\{d_i^k\}$ must be scaled by λ to be feasible and thus may not be satisfied at full rates.

5.2 Context-Aware Solution

In [4], we solved problem (5.2) with a fully polynomial-time approximation algorithm, which achieves an approximation ratio of $(1 + \omega)$ for an arbitrary $\omega > 0$. Here, we extend this algorithm in several aspects to make it more practical. The original approximation algorithm intends to fetch contents from as many source servers as possible, since more sources can potentially balance the traffic load more effectively. However, a higher overhead is also involved in aggregating the contents fetched from different servers. Hence, we further limit the number of sources engaged for each request to reduce coordination overhead. Moreover, we set a timer to ensure that the engaged video sources remain unchanged until the timer expires. Only when the timer is reset can the selected video sources be reevaluated and updated. This mechanism can further improve stability and reduce control overhead for switching among sources.

Alg. 4 shows the details of the context-aware algorithm to solve the joint source selection and flow routing problem in (5.1). After Alg. 2 described in Section 4.3.2 is applied to filter out certain video requests to be served by local device caching, Alg. 4 is used to further process the remaining uncovered requests in set $\mathcal{R}'$. The main procedure aims to determine the maximum target buffer time Δ^*, *i.e.*, the optimal value of problem (5.1), such that estimated maximum link utilization does not exceed given upper bound λ^*. Here, we combine the secant method and the bisection method to speed up the search process. The secant method starts with two initial values to construct the first secant line, and updates the next search point based on the line. Because the initial values are critical for the search efficiency, we first find reasonable bounds by a method similar to bisection search and use them as the initial values. This is much faster than using some randomly chosen initial values. As seen in Lines 2 and 3, search bounds $[\Delta^-, \Delta^+]$ are initialized and the first trial value for the target buffer time is set to the middle point.

Algorithm 4: A context-aware solution for multi-source request routing.

```
   Input: Network graph $G = (\mathcal{N}, \mathcal{E})$, link capacities $\{c(e) : e \in \mathcal{E}\}$, unfulfilled
          request set $\mathcal{R}'$, $\gamma$, interval duration $\tau$, accuracy parameters $\delta$ and $\omega$
   Output: Flows $\{x^*(P) : P \in \mathcal{P}\}$, target buffer time $\Delta^*$
1  $t \leftarrow 0$                                   // Initialize iteration round
   // Initialize search bounds for target buffer time
2  $\Delta^- \leftarrow 0$, $\Delta^+ \leftarrow$ max buffer time
3  $\Delta_t \leftarrow \lfloor (\Delta^+ + \Delta^-)/2 \rfloor$               // Initialize first search point
4  while true do
5      if $t = 1$ then
           // Determine 2nd search point $\Delta_1$ according to Alg. 5, and also
           update upper bound $\Delta^+$
6      end
7      else if $t \geq 2$ then
           // Set new search point with secant method
8          $\Delta_t \leftarrow \max\left(0, \left\lfloor \Delta_{t-1} - \lambda_{t-1} \cdot \frac{\Delta_{t-1} - \Delta_{t-2}}{\lambda_{t-1} - \lambda_{t-2}} \right\rfloor\right)$
9      end
10     Set target buffer time for next interval: $T_i^k = \min(\Delta_t$, playout time of
       unreceived data for request $(i, k))$
11     Compute remaining buffer time: $b_i^k \leftarrow s_i^k / \mu_i^k - p_i^k - \tau$
12     Compute target rate: $q_i^k \leftarrow (T_i^k - b_i^k) \cdot \mu_i^k / \tau$
13     Compute data rate $\psi_i$ over wireless link by (3.1)
14     Set request rate: $d_i^k \leftarrow \min\{\psi_i, q_i^k\}$
15     For requests in $\mathcal{R}'$ with above demands, compute max link utilization $\lambda_t$
       and flow solution $x_t(P)$ with extended algorithm for (5.2)
16     if $\Delta_t = 0$ and $\lambda_t > \lambda^*$ then
17         $\Delta^* \leftarrow \Delta_t, x^*(P) \leftarrow x_t(P)$
18         break
19     end
20     if $t \geq 1$ and $|\Delta_t - \Delta_{t-1}| \leq \delta$ then
21         $\Delta^* \leftarrow \Delta_t, x^*(P) \leftarrow x_t(P)$
22         break
23     end
24     $t \leftarrow t + 1$
25 end
26 Return $x^*(P)$ and $\Delta^*$
```

Then, Alg. 4 goes through iterative rounds in Lines 4-25. For each tentative solution Δ_t for buffer time, the demand rate for every request $(i, k) \in \mathcal{R}'$ is set according to updated user context. First, target buffer time T_i^k for request (i, k) is restricted to be not longer than the playout time of unreceived video data. Then, it estimates

Algorithm 5: A fast algorithm to determine initial search points for Alg. 4.

```
    Input: $\Delta_0$, $\lambda^*$, $\lambda_0$, $\Delta^+$, $\Delta^-$
    Output: 2nd search point $\Delta_1$, and updated upper bound $\Delta^+$
 1  if $\lambda_0 > 2 \cdot \lambda^*$ then
 2      $\Delta_1 \leftarrow 0$
 3  end
 4  else if $\lambda_0 > \lambda^*$ then
 5      $\Delta^+ \leftarrow \Delta_0$, $\Delta_1 \leftarrow \left\lfloor \frac{\Delta^- + \Delta^+}{2} \right\rfloor$
 6  end
 7  else // $\lambda_0 \le \lambda^*$
 8      $\Delta_1 \leftarrow \Delta^+$
 9  end
10  Return $\Delta_1$ and $\Delta^+$
```

remaining buffer time b_i^k, computes desired transmission rate q_i^k and achievable data rate ψ_i over the wireless channel, and sets request rate to $\min\{\psi_i, q_i^k\}$. After setting the request demands, we use the extended algorithm for (5.2) to find maximum link utilization λ_t. Lines 16-23 check the termination conditions. If λ_t exceeds target threshold λ^* even with target buffer time Δ_t set to 0, the while-loop exits since it is impossible to find a target buffer time that meets threshold λ^*. If threshold λ^* is not violated and the last two search values are sufficiently close, the search process also terminates.

Moreover, Lines 5-9 of Alg. 4 choose the next trial solution for the target buffer time. For the second round, the trial value is set according to the method in Alg. 5. If the link utilization obtained in the first round exceeds two times of threshold λ^*, Δ_t is set to 0; if the link utilization is lower but still greater than λ^*, search upper bound Δ^+ is reduced to the previous test point and Δ_t is set to the middle point of updated search range $[\Delta^-, \Delta^+]$; and otherwise Δ_t is just set to search upper bound Δ^+ since the link utilization has not reached threshold λ^*. For the subsequent rounds, as there are already at least two searched points, the next trial value is set according to the secant method (Line 9 of Alg. 4).

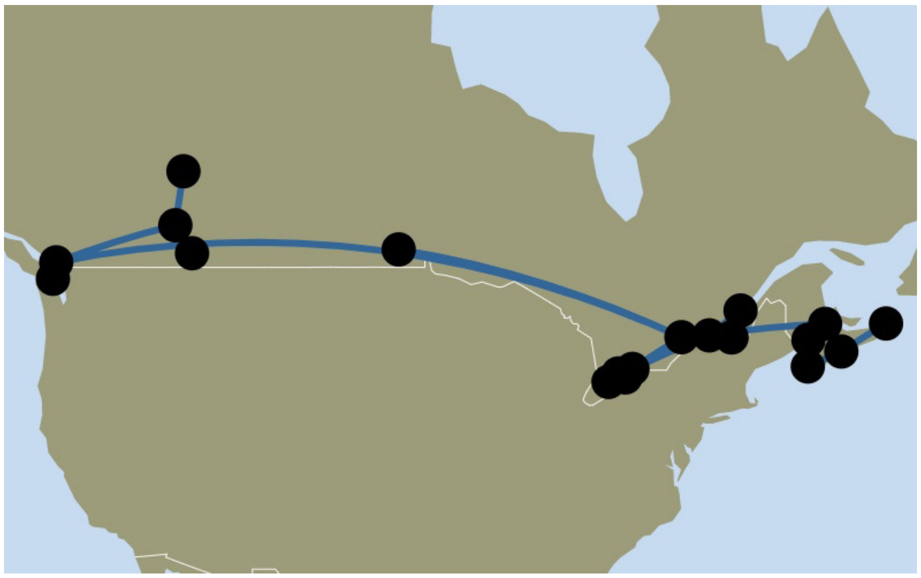
5.3 Simulation Results and Discussions

5.3.1 Simulation Settings

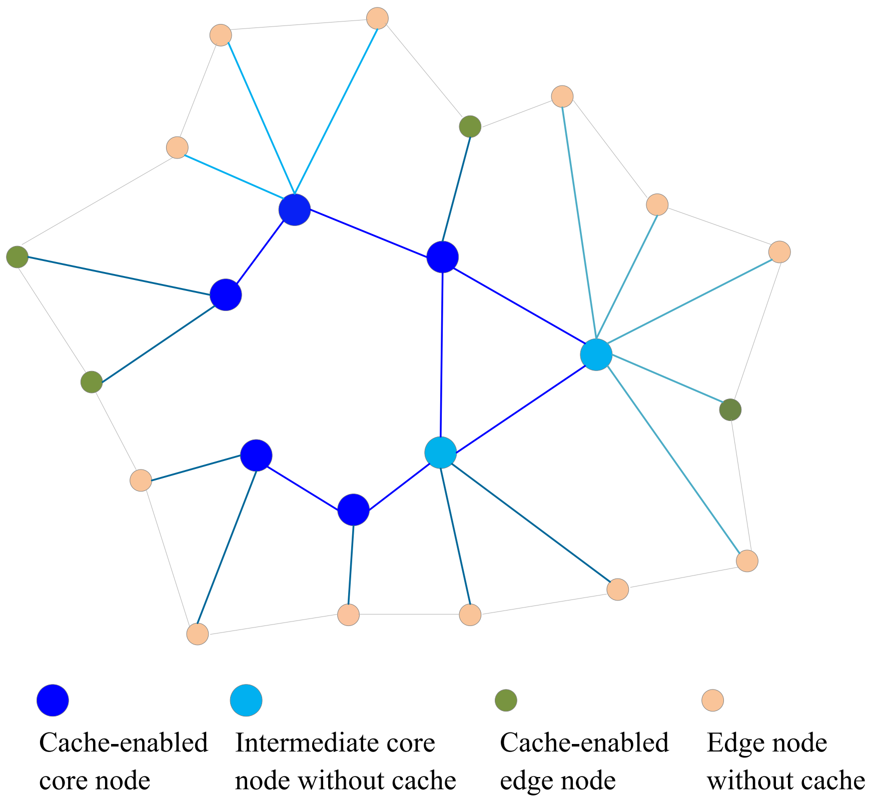
We develop a simulator in Java to evaluate the performance of the proposed solution for collaborative video distribution. Table 5.1 summarizes the key simulation parameters. In the simulations, we refer to the iSTAR network topology with a size of 23 nodes, which is cited from the Internet Topology Zoo collection [1]. The visualization of the iSTAR network created in 2011 is shown in Fig. 5.1(a). Two intermediate nodes are not shown in Fig. 5.1(a) from the Internet Topology Zoo gallery. Based on the given iSTAR network, we consider a network topology illustrated in Fig. 5.1(b), where there are 7 core nodes (including 5 core nodes with caches), 2 intermediate nodes, and 16 edge nodes (including 4 cache-enabled edge nodes). We further add links between each pair of neighbouring edge nodes to enable their collaborations. All links are assumed to be bi-directional with a capacity of 1 Gbps for each direction. Each edge server is associated with a BS and 500 UEs that are randomly distributed within a circular area of radius 250m. The social relationships among the UEs are simulated by following the Erdos-Renyi model.

To simulate YouTube-like user-provided videos, we choose 200 videos from the dataset given in [37]. The duration of all videos is in the range of [50, 150] seconds, and the coding rate of videos is in the range of [300, 350] kbps. Since core caches and edge caches differ in storage capacity, we place the whole video library among 5 core caches, while each edge cache holds 50 videos. Video clips are cached among edge nodes according to a Zipf distribution. Video requests arrive at each edge node as a Poisson process. We control the traffic load by a factor named *traffic density*, which denotes the average number of requests collected at each edge node at the beginning of an interval.

For comparison purpose, we also implement several reference schemes, which are



(a)



(b)

Fig. 5.1: The network topology for simulations. (a) Original iSTAR network visualization [1]. (b) Network topology graph based on iSTAR.

Table 5.1: Simulation Parameters.

Parameter	**Value**
Simulation interval	6s
Start-up delay	6s
Wired link capacity	1 Gbps or within [0.5, 1.5] Gbps
Video duration	[50, 150] seconds
Video coding rate	512 kbps or within [300, 350] kbps
Wireless channel bandwidth	1.4 MHz
Path loss at distance ℓ in km	$128.1 + (37.6 \log_{10}(\ell))$ (dB)
Noise power	-164 dBm
BS transmit power	32.8 dBm
Coverage radius of BS	250m
Number of UEs per cell	500

described as follows.

- The static scheme potentially configures each client to fetch content from one best source. Such static configuration scheme is used in some existing video systems. In our implementation, the static scheme always fulfills a request by the nearest source and optimizes the so-generated traffic by the shortest path first algorithm.

- The random scheme always randomly selects a source to fetch the video when a request arrives. It can potentially balance the traffic load among the available source nodes.

- The flexible hierarchical request forwarding scheme has been proposed in [21]. For the iSTAR-based network topology, we consider the core network and the edge servers as two levels in a hierarchical structure. If a request cannot be fulfilled by a neighbouring edge server, the requested content will be fetched from the core network. The source with the highest marginal value is selected, and we consider the available bandwidth over a flow path as the value here.

As seen, this scheme is limited to some specific caching topologies, whereas our solution is topology-independent.

5.3.2 Performance of Source Selection and Flow Routing Schemes

Fig. 5.2 shows the maximum link utilization of the static scheme, the random scheme, the hierarchical scheme, and the content-aware scheme with the iSTAR-based topology. A congestion line is shown to indicate over-congestion of the link, i.e., the maximum link utilization equals to 1. As the fast algorithm used in Alg. 4 guarantees to achieve a maximum link utilization not more than $(1+\omega)$ times the optimum, we can obtain a lower bound for the optimal maximum link utilization, which is also included in Fig. 5.2 as a benchmark.

As seen in Fig. 5.2, the context-aware scheme still performs the best and stays below the congestion line. In contrast, the maximum link utilization of the reference schemes increases rapidly when more video requests arrive and exceeds the congestion line quickly. That means some links are overloaded and congestion is occurring. It is worth mentioning that, when running the context-aware scheme, we restrict that each video request is served by no more than 3 cache nodes during each 10 intervals. This is to limit the switching and coordination overhead. The performance of the context-aware scheme is expected to be even better if this constraint is relaxed.

5.3.3 Performance of Request Screening Schemes

To evaluate the performance of the request screening schemes, we assess the integrated performance with request screening and routing, considering a situation where some popular videos have reached the peak of user views, and social relationship density is quite high. Here, the traffic density is set to 400 per edge node, and the Zipf coefficient for caching is $\zeta_c = 0.9$. The collaboration distance ϖ is 150m,

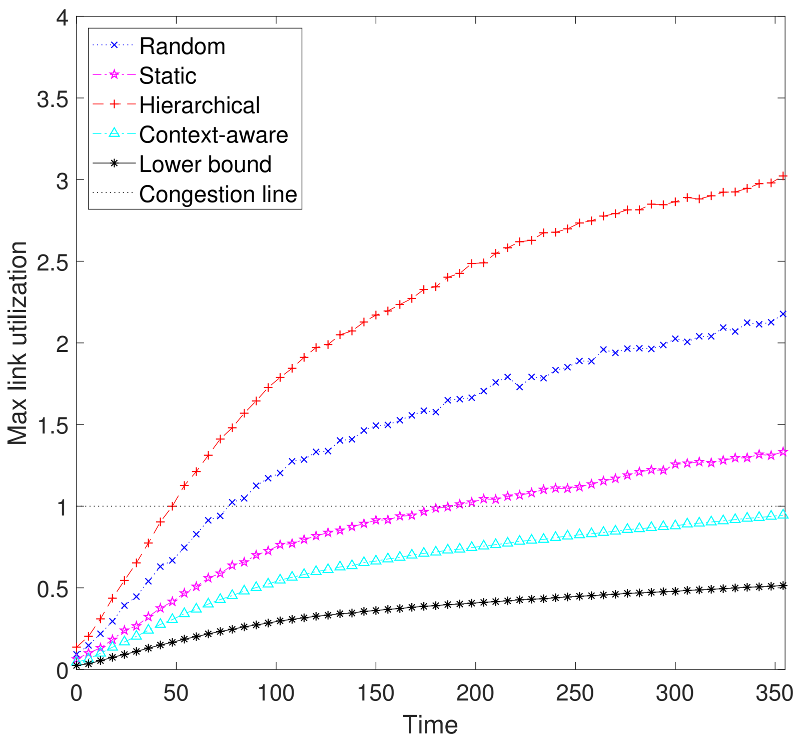


Fig. 5.2: Maximum link utilization with different source selection and flow routing schemes.

and the maximum serving bound for individual devices is set between 0 and 20 (*i.e.*, $\beta_i \in [0, 20]$, $\forall i \in \mathcal{U}$). Every device is assumed to be socially connected with each other. The maximum number of allowed cache devices is $\gamma = 20$. Fig. 5.3 shows the maximum link utilization with different solutions. As seen, the heuristic request screening scheme always outperforms the greedy scheme. In the final interval, the maximum link utilization of the greedy scheme reaches 0.7078, while the heuristic scheme only stays at 0.5368. Therefore, when some popular videos are widely requested and the social network graph is very dense, the heuristic algorithm can effectively offload traffic to cache devices and mobile edge.

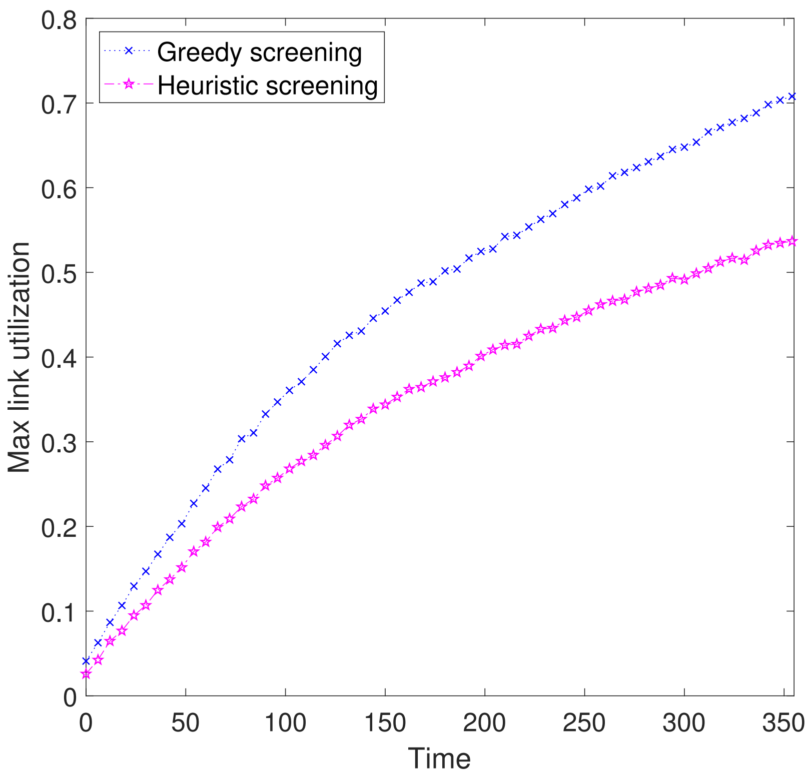


Fig. 5.3: Maximum link utilization with different request screening schemes.

5.3.4 Impact of Transmission Rate Adaptation

For the context-aware scheme, we further assess the impact of transmission rate adaptation on traffic control. As given in Alg. 4, the transmission rate for a video request can be adapted according to the wireless channel condition, client buffer status, and target buffer time. To demonstrate the effect of rate adaptation, we compare it with a scheme that associates a fixed transmission rate with a video request according to the video coding rate and wireless channel condition. We run the simulation for 60 intervals (360s totally). The traffic density is set to 180 per edge node and the Zipf coefficient for caching is $\zeta_c = 0.5$. The target buffer time of all requested videos is 5s, so the video demand rate can be adjusted at the beginning of each interval. To highlight the effect of rate adaptation, we skip request screening and set 80% devices located closer to the BS, while the rest of 20% devices are farther

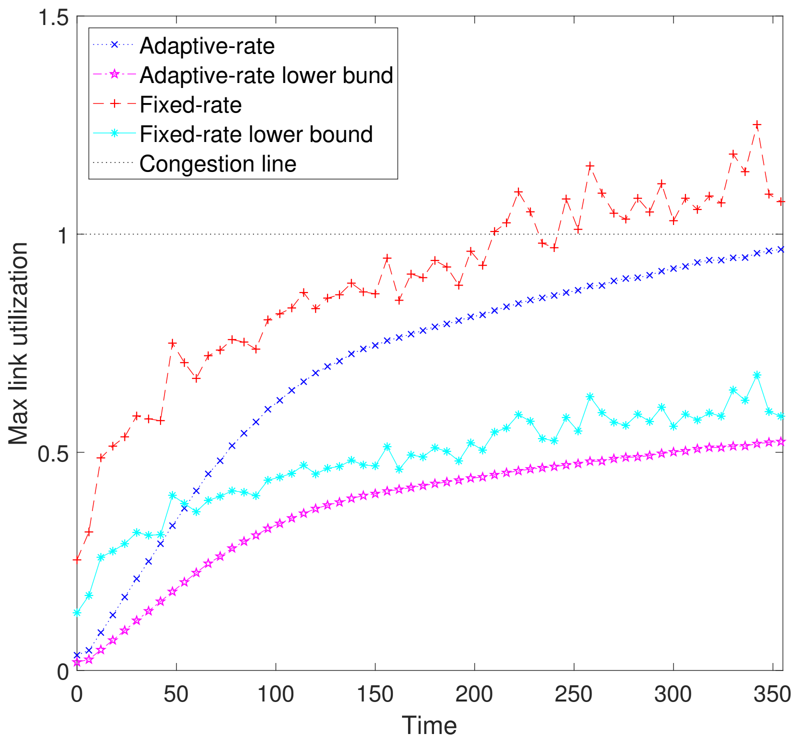


Fig. 5.4: Maximum link utilization with fixed-rate and adaptive-rate schemes.

away.

Fig. 5.4 shows the maximum link utilization and the theoretical lower bound with the fixed-rate scheme and the adaptive-rate scheme. As seen, the adaptive-rate scheme outperforms the fixed-rate scheme. For instance, with the fixed-rate scheme, some links become over-congested after time 210s, while the adaptive-rate scheme maintains link utilization below the congestion line. Moreover, it is seen that the traffic grows relatively slowly after applying the adaptive-rate scheme when more requests arrive. In contrast, the fixed-rate scheme exhibits larger fluctuations in link utilization.

5.3.5 Impact of Device Caching

When the traffic load is so heavy that network links are still over-congested even with the context-aware routing scheme, we can extend content caching to user devices to further relieve congestion. Here, we consider a scenario similar to that for Fig. 5.3, except that the collaboration distance ϖ is set to 100m and the number of cache devices is varied. Fig. 5.5 shows the performance with different values for γ, which restricts the number of D2D links allowed in a cell. As seen, the maximum link utilization with $\gamma = 10$ is reduced by 46% compared to that with $\gamma = 0$. This means that nearly half of the traffic load over the backhaul links is relieved by deploying 10 cache devices within each cell and sharing their received contents with other request devices. Parameter γ depends on the available channel resources in the radio domain. When the radio domain experiences lower interference and more cellular channels can be reused for D2D links, a larger value can be set to γ. Accordingly, the content-aware scheme redistributes the traffic in the transport domain and achieves a lower maximum link utilization. As such, we can make the best use of the available resources across domains to reduce network congestion, especially when traffic density is high and social connectivity among users is dense.

Though Fig. 5.5 shows that the content-aware scheme can efficiently fulfill video requests by exploiting collaborative video caches, it needs fine-grained context information (*e.g.*, video buffer states and link capacities) and this may involve a large overhead. Therefore, it is vital for the integrated content distribution framework to incorporate an SDN-enabled mobile network to assist context collection. In addition, it is worth noting that Alg. 4 needs to solve problem (5.2) with a $(1+\omega)$-approximation algorithm. The computing time increases proportionally to ω^{-2}. Hence, the proposed solution takes longer computing time than the reference schemes.

5.3.6 Target Buffer Time

Fig. 5.6 shows how the target buffer time is adapted to control video demand rates. Here, we consider a traffic density of 400 per edge node without request screening, and the Zipf coefficient for caching is $\zeta_c = 0.5$. In addition, we set an upper bound

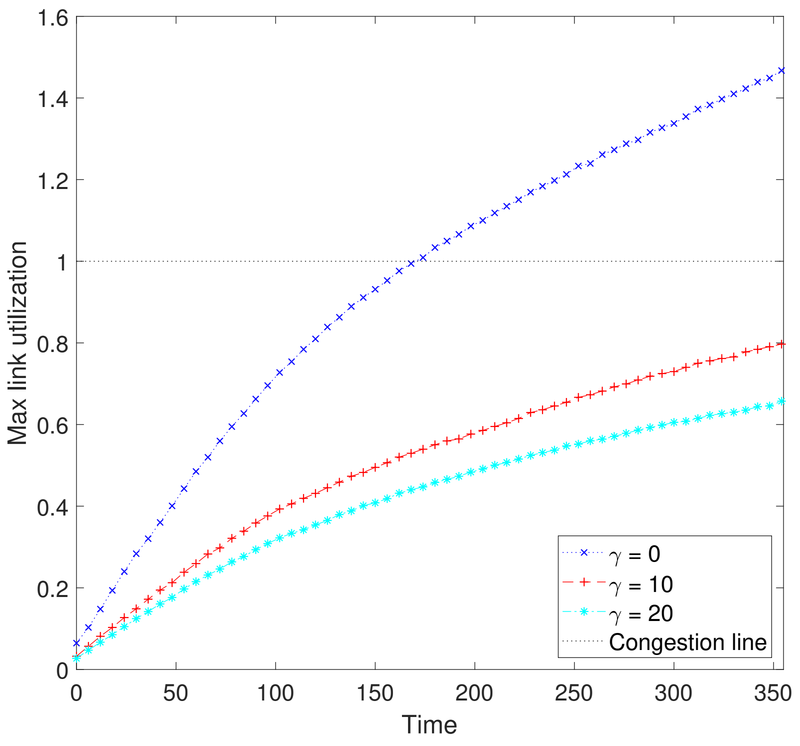


Fig. 5.5: Maximum link utilization with different numbers of cache devices at each BS.

for the target buffer time at 150s, which means the buffer time cannot exceed 150s even if physical links allow.

As seen, with the progress of time, when more video streams become active, the context-aware scheme downgrades the same amount of buffer time later than other reference schemes. For the simulation until 138s, the context-aware scheme maintains the buffer time at 150s. In contrast, the buffer time of the reference schemes is significantly lower than that. For example, some links are saturated quickly at time 24s and 42s with the hierarchical scheme and the random scheme, respectively. For the static scheme, the buffer time drastically degrades after 96s, and the performance stays between that of the context-aware scheme and that of the other two reference schemes. As a consequence, some video sessions start to experience stalls and resort to re-buffering to recover playback. This result shows that the context-aware scheme

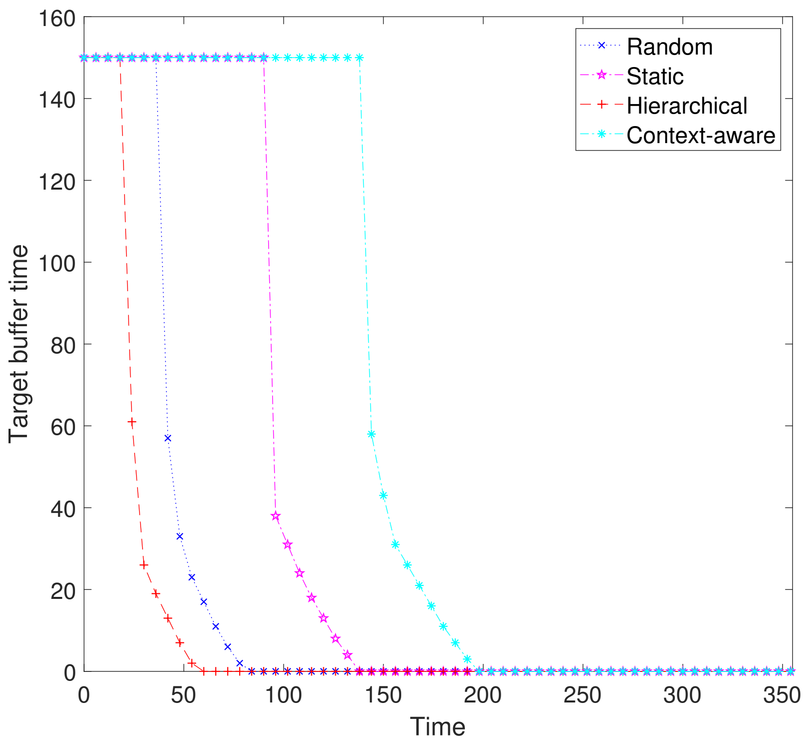


Fig. 5.6: Variation of target buffer time Δ.

can effectively exploit network and user dynamics to balance traffic load over multiple sources and multiple paths and thus relieve network congestion.

5.3.7 Comparison of Search Methods

As discussed in Section 5.1, the demand rates of requests can be determined by the target buffer time. Conversely, the target buffer time is also adjustable to adapt to backhaul link utilization and avoid over-congestion. In Alg. 4, we combine the secant method with bisection search to find the maximum buffer time between lower bound Δ^- and upper bound Δ^+ according to the threshold of maximum link utilization λ^*. Fig. 5.7 compares the number of search rounds of the combined method and classic bisection search. As seen, the combined method can effectively accelerate the search procedure with fewer rounds, especially when the target buffer time stays at the upper bound or decreases to the lower bound. For example, when the target buffer

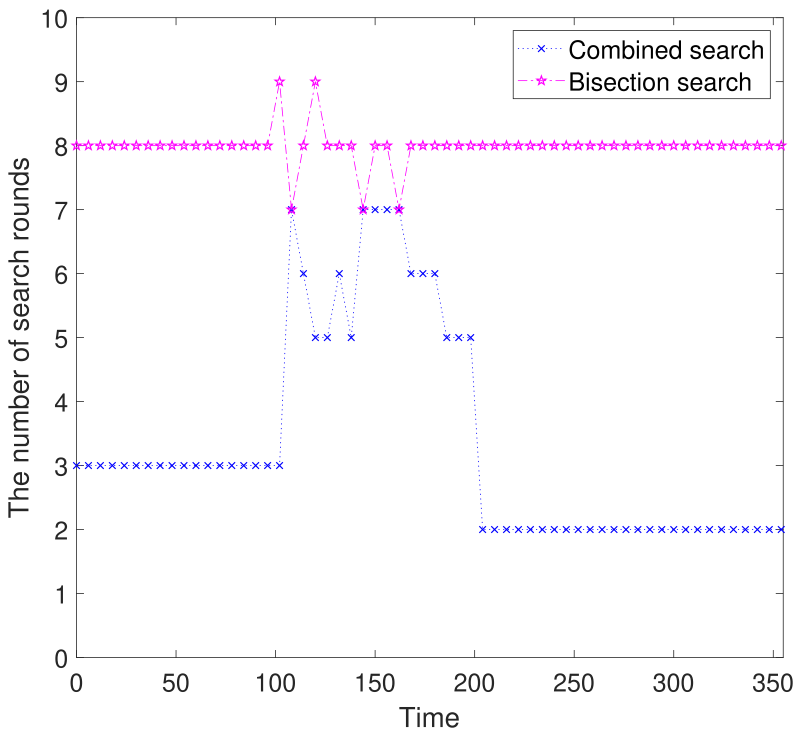


Fig. 5.7: Number of search rounds with different methods.

time equals to the upper bound, the combined method only takes 3 rounds for searching, while bisection search needs 8 rounds. Similarly, after the physical links are over-congested, bisection search still takes 8 rounds to obtain the target buffer time, while the combined method only takes 2 rounds. With bisection search, if the target buffer time reaches the upper bound or lower bound, it takes at least $\log_2(\Delta^+ - \Delta^-)$ rounds to finish search. In contrast, the combined method performs better in such cases since it involves linear interpolation during the iterative procedure.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

As the unremitting traffic from content distribution is straining the mobile networks, it becomes imperative to equip the future mobile networks with effective means to accommodate the surging demands. Emerging techniques such as mobile edge computing and D2D communications can help relieve traffic at the mobile edge and accommodate surging traffic demands from various content-centric services. In this thesis, we considered an integrated content distribution framework that leverages cutting-edge technologies such as MEC, NFV, and SDN to utilize universal in-network caching and enable collaboration across domains. Based on this framework, we introduced a video delivery solution and studied two key problems for video distribution optimization, *i.e.*, the request screening problem and the request routing problem.

First, we investigated a request screening problem for collaborative content distribution, which filters out a certain portion of the content requests to be fulfilled by the cached contents in mobile devices via D2D communications. To accommodate various resource and capacity constraints, we need to address several critical challenges while maximizing the request coverage by D2D transmissions, including the properties of content requests, the social relationships among users, the resource limitations of individual devices, as well as the interferences among cellular and D2D users. To mitigate the complexity, we decomposed the request screening problem into two subproblems. The first subproblem for device caching and matching selects certain devices as sources to temporarily store the received contents, and each source forms a group to fulfill the nearby members' content requests via D2D forwarding. The second subproblem allocates reused cellular channels for these D2D groups to provide their required resources while causing minimum interferences. As both subproblems are proved to be NP-hard, we developed heuristic algorithms to solve them efficiently.

We conducted simulations to evaluate the two proposed heuristic algorithms in both a small-scale scenario and a large-scale scenario. For the source selection and device matching problem, we compared our heuristic algorithm with the optimal solution and the greedy solution in a small-scale network. In a large-scale network where the optimal solution is unachievable, the proposed heuristic algorithm is compared to the greedy solution and the relaxation solution which is considered as an upper bound. The results show that the proposed algorithm can approximate the optimal solutions in a small-scale network and significantly outperforms the reference schemes in a large-scale network. Furthermore, we evaluated the impact of the collaboration distance, the maximum individual serve budget, the maximum number of D2D links per cell, and the social relationship probability. Although the proposed algorithm is relatively more time consuming, the results show that the proposed heuristic algorithm can screen more requests compared to the reference schemes. In addition, we compared our proposed heuristic algorithm for D2D channel allocation with the optimal solution and a greedy algorithm in a small-scale network and with only the greedy scheme in a large-scale network. The results show that the proposed algorithm performs close to the optimum and outperforms the greedy scheme in both scenarios.

Second, we further studied the joint source selection and flow routing problem. For the remaining requests that cannot be fulfilled by device caching, we determined the demand rates and target buffer playback time, so as to meet a threshold for the maximum allowed link utilization, which reflects the tolerable congestion level. Accordingly, the content requests are directed to potentially multiple sources and routed through collaborative nodes to balance the traffic load. With the assistance of the framework, our solution exploits user context, such as the wireless link capacity, video client's buffer status, and dynamic request information, in order to steer traffic and utilize diverse available paths. We developed a context-aware solution over the framework to provide collaborative video content distribution.

The computer simulations were conducted with a real-world network topology to evaluate the proposed context-aware routing solution and the combination of request screening and routing. We compared our context-aware solution with the static scheme, random scheme, and flexible hierarchical request forwarding scheme. The simulation results demonstrate significant performance gain over the reference schemes in reducing network congestion in terms of the maximum link utilization, and the target buffer time. By improving the secant search method, the search rounds for the target buffer time are reduced on average.

6.2 Future Work

Here we highlight several research directions that can extend this work in the future. First, the solutions for two subproblems of request screening can be further improved. As we used heuristic algorithms for both device matching and channel allocating, more advanced solutions can be explored to guarantee the performance stability while consuming acceptable computation time.

Second, the context-aware request routing solution can be evaluated with other real-world network topologies. In this thesis, we only conduct evaluations on the iSTAR network topology. However, some routing solutions are limited to some specific caching topologies. For example, the flexible hierarchical request forwarding solution is more effective with the network topology where coordination is easy between network nodes at the same level of the hierarchical structure. The nearest source forwarding solution can outperform the hierarchical solution with some topologies. Therefore, in order to prove the topology-independence of the context-aware solution, it can be further tested with more topologies in the future.

Third, content replacement in the mobile edge can further improve the efficiency of caching resources. There have been extensive studies on content placement in various network segments, ranging from traditional CDNs in the cloud to the mobile network edge. Nevertheless, more in-depth studies are needed on how to coordinate content placement and delivery. For instance, content delivery via D2D communications may change the dominant caching paradigm that pre-fetches contents in advance and updates the contents periodically. Instead, *caching on demand* may prevail for device caching, considering the resource and storage constraints of mobile devices. The coordination between content placement and delivery can balance the resource costs in the two stages and reduce resource waste on unnecessary caching. Due to the storage constraint, each BS can only cache a limited portion of video library. In a short time scale, the popularity of all videos in the library is stable. However, during a long period, users may change their preference so the popularity distribution of video contents experiences dynamic changes. In order to achieve a higher hit ratio, the contents cached in BSs can be updated after an appropriate time period. For example, a BS can update its storage following some classic cache replacement policies such as the least frequently used and the least frequent recently used.

Bibliography

[1] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, “The Internet topology zoo,” *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, 2011. [Online]. Available: http://www.topology-zoo.org/

[2] W. Song, Y. Zhao, and W. Zhuang, “Stable device pairing for collaborative data dissemination with device-to-device communications,” *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1251–1264, 2018.

[3] J. Xie and W. Song, “Channel-aware device-to-device pairing for collaborative content distribution,” in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, 2017.

[4] J. He and W. Song, “Optimizing video request routing in mobile networks with built-in content caching,” *IEEE Transactions on Mobile Computing*, vol. 15, no. 7, pp. 1714–1727, 2016.

[5] Y. Zhao, W. Song, and Z. Han, “Social-aware data dissemination via device-to-device communications: Fusing social and mobile networks with incentive constraints,” *IEEE Transactions on Services Computing*, pp. 1–14, 2016.

[6] Intel and Nokia Siemens Networks, “Increasing mobile operators value proposition with edge computing,” *Technical Brief*, 2013.

[7] IBM Corporation, “Smarter wireless networks: Add intelligence to the mobile network edge,” *Thought Leadership White Paper*, 2013.

[8] Saguna and Intel, “Using mobile edge computing to improve mobile network performance and profitability,” *White Paper*, 2016.

[9] Y. C. H, M. Patel, D. Sabella, N. Sprecher, and V. Young, “Mobile edge computing - A key technology towards 5G,” *ETSI White Paper*, no. 11, 2015.

[10] A. Rostami, P. Ohlen, K. Wang, Z. Ghebretensae, B. Skubic, M. Santos, and A. Vidal, “Orchestration of RAN and transport networks for 5G: An SDN approach,” *IEEE Communications Magazine*, vol. 55, no. 4, pp. 64–70, 2017.

[11] P. Yang, N. Zhang, Y. Bi, L. Yu, and X. Shen, “Catalyzing cloud-fog interoperation in 5G wireless networks: An SDN approach,” *IEEE Network*, vol. 31, no. 5, pp. 14–20, 2017.

[12] A. Bradai, K. Singh, T. Ahmed, and T. Rasheed, “Cellular software defined networking: A framework,” *IEEE Communications Magazine*, vol. 53, no. 6, pp. 36–43, 2015.

[13] Z. Liu, T. Peng, B. Peng, and W. Wang, “Sum-capacity of D2D and cellular hybrid networks over cooperation and non-cooperation,” in *Proceedings of International Conference on Communications and Networking in China (CHINACOM)*, 2012, pp. 707–711.

[14] K. Vanganuru, S. Ferrante, and G. Sternberg, “System capacity and coverage of a cellular network with D2D mobile relays,” in *Proceedings of IEEE Military Communications Conference*, 2012, pp. 1–6.

[15] D. Lee, S. I. Kim, J. Lee, and J. Heo, “Performance of multihop decode-and-forward relaying assisted device-to-device communication underlaying cellular

networks," in *Proceedings of International Symposium on Information Theory and its Applications (ISITA)*, 2012, pp. 455–459.

[16] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. M. Leung, "Cache in the air: Exploiting content caching and delivery techniques for 5G systems," *IEEE Communications Magazine*, vol. 52, no. 3, pp. 131–139, 2014.

[17] H. Ahlehagh and S. Dey, "Video-aware scheduling and caching in the radio access network," *IEEE/ACM Transactions on Networking*, vol. 22, no. 5, pp. 1444–1462, 2014.

[18] S. Zhang, N. Zhang, P. Yang, and X. Shen, "Cost-effective cache deployment in mobile heterogeneous networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 12, pp. 11 264–11 276, 2017.

[19] T. X. Tran, A. Hajisami, and D. Pompili, "Cooperative hierarchical caching in 5G cloud radio access networks (C-RANs)," *IEEE Network*, vol. 31, no. 4, pp. 35–41, 2017.

[20] W. Wang, R. Lan, J. Gu, A. Huang, H. Shan, and Z. Zhang, "Edge caching at base stations with device-to-device offloading," *IEEE Access*, vol. 5, pp. 6399–6410, 2017.

[21] S. Ren, T. Lin, W. An, G. Zhang, D. Wu, L. N. Bhuyan, and Z. Xu, "Design and analysis of collaborative EPC and RAN caching for LTE mobile networks," *Computer Networks*, vol. 33, no. 1, pp. 80–95, 2015.

[22] S. Soleimani and X. Tao, "Cooperative crossing cache placement in cache-enabled device to device-aided cellular networks," *Applied Sciences*, vol. 8, no. 9, p. 1578, 2018.

[23] T. Chen, M. Matinmikko, X. Chen, X. Zhou, and P. Ahokangas, "Software defined mobile networks: Concept, survey, and research directions," *IEEE Communications Magazine*, vol. 53, no. 11, pp. 126–133, 2015.

[24] D. Malak, M. Al-Shalash, and J. G. Andrews, "Optimizing content caching to maximize the density of successful receptions in device-to-device networking," *IEEE Transactions on Communications*, vol. 64, no. 10, pp. 4365–4380, 2016.

[25] W. Song and Y. Zhao, "Efficient interference-aware D2D pairing for collaborative data dissemination," in *Proceedings of IEEE Conference on Communications (ICC)*, 2018, pp. 1–6.

[26] M. A. Habibi, B. Han, and H. D. Schotten, "Stable device pairing for collaborative data dissemination with device-to-device communications," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1251–1264, 2018.

[27] J. Xie and W. Song, "Channel-aware device-to-device pairing for collaborative content distribution," in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, 2017, pp. 1–6.

[28] J. Xie, X. Tao, and W. Song, "A study of multicast message allocation for content distribution with device-to-device communications," in *Proceedings of IEEE Vehicular Technology Conference (VTC Fall)*, 2017, pp. 1–6.

[29] Y. Zhao and W. Song, "Energy-efficient incentivized data dissemination via wireless D2D communications with weighted social communities," *IEEE Transactions on Green Communications and Networking*, vol. 2, no. 4, pp. 945–957, 2018.

[30] B. Bai, L. Wang, Z. Han, W. Chen, and T. Svensson, "Caching based socially-aware D2D communications in wireless content delivery networks: A hypergraph framework," *IEEE Wireless Communications*, vol. 23, no. 4, pp. 74–81, 2016.

[31] K. Zhu, W. Zhi, L. Zhang, X. Chen, and X. Fu, "Social-aware incentivized caching for D2D communications," *IEEE Access*, vol. 4, pp. 7585–7593, 2016.

[32] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017.

[33] N. Golrezaei, P. Mansourifard, A. F. Molisch, and A. G. Dimakis, "Base-station assisted device-to-device communications for high-throughput wireless video networks," *IEEE Transactions on Wireless Communications*, vol. 13, no. 7, pp. 3665–3676, 2014.

[34] F. Dobrian, A. Awan, D. Joseph, A. Ganjam, J. Zhan, V. Sekar, I. Stoica, and H. Zhang, "Understanding the impact of video quality on user engagement," *SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 362–373, 2011.

[35] H. Planatscher and M. Schober. Scpsolver - An easy to use Java linear programming interface. [Online]. Available: http://scpsolver.org/

[36] R. M. R. Lewis, *A Guide to Graph Colouring Algorithms and Applications.* Springer, 2016, ch. Bounds and Constructive Algorithms.

[37] X. Cheng, C. Dale, and J. Liu., "Dataset for statistics and social network of YouTube videos," 2008. [Online]. Available: http://netsg.cs.sfu.ca/youtubedata/

Vita

Candidate's full name: Haoru Xing

University attended:

September 2013 - June 2017
Bachelor of Engineering
School of Digital Media and Design Arts
Beijing University of Posts and Telecommunications
Beijing, China

September 2017 - November 2019
Master of Computer Science
Faculty of Computer Science
University of New Brunswick
Fredericton, New Brunswick, Canada

Publications:

1. H. Xing and W. Song, "Collaborative Content Distribution in 5G Mobile Networks with Edge Caching," *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pp. 1-6, Shanghai, China, 2019.

2. H. Xing and W. Song, "Collaborative Content Distribution in 5G Mobile Networks with Edge Caching," *Poster in Faculty of Computer Science Annual Research Expo*, University of New Brunswick, Fredericton, NB, Canada, April 2019.

3. H. Xing and W. Song, "Request Offloading with Social-Aware Device Caching for D2D-Assisted Content Distribution" submitted to *IEEE Transactions on Green Communications and Networking*, 2019.