I, Minzhe Guo, hereby submit this original work as part of the requirements for the degree of Doctor of Philosophy in Computer Science & Engineering.

It is entitled:
Algorithmic Mechanism Design for Data Replication Problems

Student's name: Minzhe Guo

This work and its defense approved by:

Committee chair: Prabir Bhattacharya, Ph.D.

Committee member: Ki Jung Lee, Ph.D.

Committee member: Dharma Agrawal, D.Sc.

Committee member: Raj Bhatnagar, Ph.D.

Committee member: John Franco, Ph.D.
Algorithmic Mechanism Design for Data Replication Problems

A dissertation submitted to the Graduate School of the University of Cincinnati in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Electrical Engineering and Computing Systems of the College of Engineering and Applied Science by

Minzhe Guo

M.S. Southern Polytechnic State University, 2010

June, 2016

Committee Chair: Prabir Bhattacharya, Ph.D.
Abstract

Data replication is an important technique in modern storage-capable distributed systems, such as content delivery networks (CDNs), peer-to-peer networks (P2Ps), and mobile networks, for improving system availability, reliability, and fault-tolerance. Most existing studies on data replication problems assume that all participants in the system fully comply with the designed protocols. Nevertheless, in real-world data replication applications, entities, e.g., servers, data providers, or data consumers, can belong to different stakeholders or administrative domains with different preferences and objectives, exhibiting heterogeneous behaviors that may not be consistent with the expected behavior of the designed protocols. This dissertation studies the problems of data replica placement (DRP), a key component in data replication applications, and utilizes algorithmic mechanism design theory to design algorithms for DRP problems in the settings with heterogeneous behavior models.

We first study the DRP problem in CDN in a strategic setting where multiple self-interested players with private preferences own data objects for replication. We design quantitative metrics to measure the content delivery cost associated with specific replica placements and investigate the super-modularity and monotonicity of the cost metrics. We then design DRPMECH, an incentive compatible mechanism that approximates a socially efficient solution to the
problem. A detailed set of experiments validates the properties of DRPMECH and shows that it outperforms a state-of-the-art game-theoretical algorithm.

Next, we study the DRP problem in peer-assisted content delivery networks (PCDNs) with self-interested seeders. Recently, PCDNs have been proposed for simultaneously obtaining the scalability advantage of P2Ps and the reliability and manageability advantages of CDNs. However, the benefits of peer assistance can be severely affected by the self-interested nature of peers, who in general wish to download more and upload less, unless otherwise motivated. We present DPRP-IC, a decentralized algorithm for DRP in PCDNs, which considers peer contributions and incentives self-interested seeders. We investigate the incentive compatibility of the algorithm and design experiments to evaluate its performance. Results suggest that replica placement algorithms that consider peer contributions have better performance in PCDN; in addition, our approach incentivizes the contribution of self-interested seeders and further improves the performance of replica placement in PCDN.

Finally, we study the impact of malicious behaviors on the mechanism design. We identify two types of adversaries in the system, design quantitative metrics to measure the magnitude of malice, and experimentally evaluate the impact of malicious behaviors on the DPRP-IC algorithm. We then integrate the probability of malice of each agent into mechanism design and extend DPRP-IC to a security aware solution, DPRP-IC-SA, which is more resilient to malicious attacks, as demonstrated with a detailed set of experiments.
Acknowledgments

There are many without whom this dissertation would not have been finished. First of all, I would like to thank my advisor Dr. Prabir Bhattacharya, who has been a gentle, patient, far-sighted, and knowledgeable advisor ever since the first day that I came to Cincinnati. Dr. Bhattacharya always provides invaluable guidance and prompt feedback throughout the course of my graduate study. Even during the times when my attention was attracted by other things, he kept encouraging me and helping me to overcome those difficulties.

Next, I would like to thank Dr. Dharma Agrawal, Dr. John Franco, Dr. Ki Jung Lee, and Dr. Raj Bhatnagar for their time to serve as my dissertation committee members. I am grateful to them for their invaluable suggestions on my graduate study and dissertation.

I would also like to express my sincere gratitude to Dr. Yan Xu for employing me as a research assistant in her lab at Cincinnati Children’s Hospital Medical Center, providing me the opportunities to conduct research on cutting-edge challenges in biomedical data analysis.

My parents have provided a solid and loving foundation on which I could easily pursue my studies. Along the way, I have been fortunate to have good friends to help me and encourage me, including Huaiyu, Hailong, and Yina.
Content

List of Figures ............................................................................................................. VIII
List of Tables ............................................................................................................. X

1 Introduction ............................................................................................................. 1
  1.1 Research Problems and Contributions ......................................................... 4
  1.2 Thesis Outline ................................................................................................. 8

2 Background ............................................................................................................. 9
  2.1 Content Delivery Systems ........................................................................... 10
  2.2 Data Replica Placement Problem and Related Work ................................. 12
  2.3 Behavioral Models ......................................................................................... 15
  2.4 Algorithmic Mechanism Design ................................................................. 16

3 Mechanism Design for Replica Placement in Content Delivery Networks with Self-interested Content Providers ........................................... 19
  3.1 Introduction ................................................................................................... 20
  3.2 System Modeling and Problem Formulation .............................................. 22
    3.2.1 The DRP Problem ............................................................................... 22
    3.2.2 Cost Model ......................................................................................... 25
    3.2.3 The DRP Problem in Strategic Setting ............................................. 31
  3.3 The Mechanism ............................................................................................. 33
3.4 Evaluation ................................................................. 42
   3.4.1 Environment Simulation and Data Generation .............. 42
   3.4.2 Validation of Incentive Compatibility ..................... 44
   3.4.3 Validation of Local Efficiency ............................ 46
   3.4.4 Global Efficiency Evaluation .............................. 49
3.5 Conclusions .................................................................. 55

4 Mechanism Design for Replica Placement in Peer-assisted Content Delivery with Self-interested Seeders ........................................ 56
   4.1 Introduction ................................................................ 57
   4.2 System Model and Problem Formulation ..................... 61
   4.3 Replica Placement with Incentive Compatible Seeding ....... 72
   4.4 Evaluation ................................................................ 78
      4.4.1 Environment Setup ......................................... 78
      4.4.2 Results .......................................................... 81
   4.5 Conclusions ............................................................ 88

5 Mechanism Design for Replica Placement in Peer-assisted Content Delivery with Heterogeneous Behavior Models ....................... 89
   5.1 Introduction ................................................................ 90
   5.2 System Model .......................................................... 92
   5.3 Input and Execution Adversaries ............................... 94
      5.3.1 Input Adversary ................................................. 95
      5.3.2 Execution Adversary ......................................... 97
      5.3.3 Impact of Malicious Behaviors on DPRP-IC ......... 98
5.4 Replica Placement with Security-Aware Incentive-Compatible Seeding

5.5 Conclusions

6 Conclusions and Future Work

6.1 Conclusions

6.2 Future Work

Bibliography

Appendices

A. Proofs of the Results in Chapter 3

B. Proofs of the Results in Chapter 4

C. Proofs of the Results in Chapter 5
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>An example content delivery network</td>
<td>11</td>
</tr>
<tr>
<td>3.1</td>
<td>An example network topology for data replica placement</td>
<td>23</td>
</tr>
<tr>
<td>3.2</td>
<td>Truthful bidding in DRPMECH optimizes the utility of player 1 when other players are also truthful bidding</td>
<td>45</td>
</tr>
<tr>
<td>3.3</td>
<td>Truthful bidding in DRPMECH optimizes the utility of player 1 when other players are randomized bidding</td>
<td>46</td>
</tr>
<tr>
<td>3.4</td>
<td>One of the six networks for local efficiency validation</td>
<td>48</td>
</tr>
<tr>
<td>3.5</td>
<td>The impact of capacity ratio on the performance of DRPMECH and MECH</td>
<td>51</td>
</tr>
<tr>
<td>3.6</td>
<td>The impact of the number of data objects on the performance of DRPMECH and MECH</td>
<td>52</td>
</tr>
<tr>
<td>3.7</td>
<td>The impact of the number of replication sites on the performance of DRPMECH and MECH</td>
<td>53</td>
</tr>
<tr>
<td>3.8</td>
<td>The impact of the write-read ratio on the performance of DRPMECH and MECH</td>
<td>54</td>
</tr>
<tr>
<td>4.1</td>
<td>An abstract architecture of peer-assisted content delivery network</td>
<td>61</td>
</tr>
<tr>
<td>4.2</td>
<td>Types of peers in a downloading swarm of a server in PCDN</td>
<td>65</td>
</tr>
<tr>
<td>4.3</td>
<td>An economic model of replica placement in PCDN</td>
<td>66</td>
</tr>
<tr>
<td>4.4</td>
<td>The decentralized PCDN replica placement scheme</td>
<td>76</td>
</tr>
<tr>
<td>4.5</td>
<td>The decentralized PCDN replica placement with incentive compatible seeding</td>
<td>77</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>4.6</td>
<td>The impact of the maintenance cost on the performance of replica placement algorithms</td>
<td>84</td>
</tr>
<tr>
<td>4.7</td>
<td>The impact of the seeding prices on the performance of replica placement algorithms</td>
<td>85</td>
</tr>
<tr>
<td>4.8</td>
<td>The impact of the seeding capacity on the performance of replica placement algorithms</td>
<td>86</td>
</tr>
<tr>
<td>4.9</td>
<td>The impact of the number of seeders on the performance of replica placement algorithms</td>
<td>87</td>
</tr>
<tr>
<td>5.1</td>
<td>Types of peers in a downloading swarm of a server in PCDN</td>
<td>91</td>
</tr>
<tr>
<td>5.2</td>
<td>The characteristics of the trace T4’05 data</td>
<td>100</td>
</tr>
<tr>
<td>5.3</td>
<td>The impact of input adversary on DPRP-IC</td>
<td>102</td>
</tr>
<tr>
<td>5.4</td>
<td>The impact of execution adversary on DPRP-IC</td>
<td>103</td>
</tr>
<tr>
<td>5.5</td>
<td>The decentralized PCDN replica placement with security-aware incentive-compatible seeding</td>
<td>106</td>
</tr>
<tr>
<td>5.6</td>
<td>DPRP-IC-SA is more resilient to malicious attacks</td>
<td>107</td>
</tr>
</tbody>
</table>
List of Tables

Table 3.1  Parameters for DRP instances for local efficiency validation  48
Table 3.2  Validation of local approximation ratio  49
Chapter 1

Introduction

A basic assumption on designing protocols and algorithms in computer science is that all entities are fully faithful, i.e., they will honestly comply with the designed operations and protocols. This assumption was concise and effective in the early stage of computing environment where the networks were relatively small and rather isolated within each stakeholder. However, with the enormous complexity and applicability being evolved in today's pervading computing environment, this model would seem rather simple. Research efforts have been invested on studying more advanced approaches to model the real-world cyber environment from different perspectives and with the interactions among multiple disciplines, e.g. psychology, sociology, and economics. The prevalence of Internet as a standard platform for distributed computation and the proliferation of economic activities in Internet have rendered the issues like ownership, cooperation, incentive, or competition essential for Internet applications. Therefore, in one direction, researchers have started to look at the problems from economic/social perspectives and introduce self-interested agents into the model, i.e., agents may belong to different stakeholders with different profit functions.
and they behave with the objective to maximize their own profits. Game theory and mechanism design theory are the important tools to analyze and solve the problems in the cyber environment from the economic perspectives. Game theory [1-4] is a long established branch of economics; mechanism design can be thought of as the “engineering” side of economic theory [5] and it has profound applications in auction theory [6-10]. There is a growing interest in the interactions of game theory and mechanism design with computer science [11-13]. In another direction, the research on cybersecurity introduces the malicious agents into the model, i.e., the agents can behave maliciously to do harm to other agents due to various reasons. The research in this direction tends to solve the problems from technological perspectives, using technical solutions (e.g., encryption, intrusion detection, firewall, etc.) to countermeasure the security and privacy problems. Nevertheless, technology and behavior are intrinsically linked in the world of cybersecurity. It is increasingly acknowledged that no solution for cybersecurity is complete without examining the behaviors of the agents in the system. The primary goal of our research is to design robust and practical protocols for Internet applications in realistic settings with complex and heterogeneous behavioral models, including altruistic, self-interested, or malicious behaviors.

This dissertation attempts to approach to this goal through the study of data replication problems in strategic and adversarial settings. Data replication is an important technique in many modern storage-capable distributed systems, such
as content delivery networks (CDNs), peer-to-peer networks (P2Ps), wireless sensor networks, or mobile networks, for improving system availability, reliability, and fault-tolerance. Most related efforts on data replication problems assume that all participants in the system fully comply with the designed protocols. Nevertheless, in realistic environments, entities (e.g., servers or data content) in a data replication application can belong to different administrative domains with different objectives. They would have obvious incentives to perform the replications with the objective to maximize their own benefits; or they may behave maliciously to compromise the security properties (confidentiality, integrity, and availability) of other entities or the entire system. This calls for protocols that can provide incentives for self-interested entities and limit the impact of malicious entities on the system. Our approach towards designing such protocols is mainly based on algorithmic mechanism design (AMD) theory [11], a new field of study that lies on the border of mechanism design and computer science that deals with mechanism design in algorithmically-complex scenarios. Traditional AMD research assumes that all agents are selfish and aims to design incentive compatible mechanisms with socially and computationally efficient implementations. The impact of malicious behavior on the incentive compatibility and social efficiency of AMDs has not been fully understood. Our research will contribute to the understanding of mechanism design in the settings with malicious behavior.
Though the research presented in this thesis focuses on a few specific topics, it is part of a larger effort on understanding the economic and social dimensions of entities in complicated systems and their relations with cybersecurity. The results from this research not only contribute to improving our understanding of data replication problems in content delivery systems but also benefit the research on the design of robust algorithms for problems with the co-existence of altruistic, selfish, and malicious agents. In addition, we provide insights into the impact of malicious behavior on the incentive compatible mechanisms and the design of security aware mechanisms.

1.1 Research Problems and Contributions

This dissertation studies the problems of data replica placement (DRP), a type of data replication problems that aims to find an optimal placement of replicas of data objects/content within a set of replication servers such that the demands for the data can be satisfied with minimal cost. Though there have been an extensive number of studies on DRP problems, most assume that the agents in the system have a homogeneous behavior model, such as all altruistic or all self-interested; in addition, most of the studies focus on DRP problem in CDN or P2P environments separately; the DRP algorithms for the hybrid CDN-P2P systems with heterogeneous types of agents are sparse. In this research, we study the DRP problems in both CDN and PCDN systems and consider heterogeneous behaviors models. Specific research problems and our contributions are presented in the following.
First, we study the DRP problem in CDN in a strategic setting inspired by practical market-based data replication applications. Multiple self-interested players with private preferences own data objects for replication. Players compete for storage space among replication servers with the objective to optimize their own profits. In traditional mechanism design, the above modeling of the DRP problem in the strategic setting can be naturally formulated as a combinatorial auction. Nevertheless, the combinatorial auction modeling presents challenges to the application of DRP in real-world large-scale CDN systems. The simultaneity in combinatorial auctions may involve complex coordination among a potentially large number of replication sites and may cause service disruption due to the large amount of data to be transmitted. In addition, combinatorial auctions require solving computationally hard problems. Even though truthful polynomial-time approximation schemes do exist [14], the running time is still expensive in practice. In fact, no common auction environment is running simultaneous auctions for large set of items [15].

Our contributions to this problem include: (1) we present an alternative model that formulates the problem as a sequential composition of knapsack auctions, in which each auction sells the space at a replication site; (2) we define a cost saving valuation to represent the preference of self-interested data owners for different placement schemes, and prove the sub-modularity and monotonicity of the cost saving valuation; (3) an algorithmic mechanism, DRPMECH, is designed to elicit the preferences of self-interested data owners; and (4) we analyze both
the economic and computational properties of DRPMECH, validate the properties using experiments, and compare its performance against a related game-theoretical solution.

Next, we study the DRP problem in peer-assisted content delivery networks (PCDNs) with altruistic and selfish seeders. Recently, PCDN systems have been proposed as a type of hybrid CDN-P2P system that aims to incorporate the best of both approaches and offset each other’s disadvantages. However, the benefits of peer assistance can be severely affected by the selfish or malicious behaviors of peers. While an extensive number of DRP algorithms have been proposed for traditional CDNs, they may not work efficiently in PCDNs where P2P technology is incorporated. Only a few algorithms have been proposed for DRP in PCDN; and DRP algorithms that consider self-interested behaviors in PCDNs are sparse.

Our contributions to this problem include: (1) we introduce an economic model to consolidate the delivery cost of PCDN and the incentives of peer participation; (2) we formulate the DRP problem with seeding incentives in PCDN; (3) a decentralized algorithm, named DPRP-IC, is developed to incentivize the upload contribution of self-interested seeders while controlling the payments for seeding incentives, and ultimately reducing the cost of content delivery in the system; (4) we design a set of experiments to comparatively evaluate DPRP-IC against POP and Greedy-CDN (two DRP algorithms without considering peer contributions), as well as DPRP (a decentralized DRP algorithm that considers
peer contributions). Results show that the DRP algorithms that consider peer contributions have better performance in PCDN; in addition, DPRP-IC incentivizes the contribution of selfish seeders and further improves the performance of DRP in PCDN.

Finally, we study the DRP in PCDN systems with the co-existence of altruistic, selfish, and malicious seeders. In real-world PCDN systems, entities may behave maliciously due to various reasons. Traditional mechanisms are vulnerable to malicious behaviors. The impact of malicious behavior on the properties of the mechanisms is not fully understood. In addition, designing attack resilient mechanisms is also an open challenge in AMD research.

Our main contributions to this problem include: (1) two types of adversaries are identified, including input adversary – seeders who report information to the mechanism with the objective to reduce the utility of other agents, and execution adversary – seeders who win in the mechanism but refuse to provide the upload bandwidth to the system, causing a denial of service like attack to the system; (2) we design two metrics to measure the magnitude of malice, including malicious volume – the percentage of upload bandwidth being compromised; and malicious size – the number of malicious seeders; (3) the impact of malicious behaviors is quantified by price of malice, which is defined as the percentage of performance being reduced due to malicious attacks; (4) we then integrate the probability of malice of each agent into mechanism design and extend DPRP-IC to a security aware algorithm, named DPRP-IC-SA; (5) we
design experiments to evaluate the impact of malicious behaviors on the algorithms and show that DPRP-IC-SA is more resilient to malicious attacks.

1.2 Thesis Outline

The rest of this dissertation is organized as follows. Chapter 2 presents background information on algorithmic mechanism design theory and data replication problems in our study. Chapter 3 elaborates our efforts on developing an incentive compatible mechanism for the DRP problem in CDN with self-interested data owners. In Chapter 4, we formulate the DRP problem in PCDN systems with self-interested seeders and present a decentralized PCDN replica placement scheme with incentive compatible seeding to reduce the cost of content delivery in PCDN. Chapter 5 studies the DRP in PCDN with the co-existence of altruistic, self-interested, and malicious seeders, analyzes the impact of malicious behaviors on the algorithm, and presents an approach to design a security aware algorithm that is more resilient to security attacks. Chapter 6 summarizes our work and presents possible extensions of current work.
Chapter 2

Background

The work in this dissertation builds upon the existing body of literature in several research areas ranging from content delivery systems to algorithmic mechanism design theory. Since the content delivery systems, including traditional content delivery networks and peer-assisted content delivery networks, are the main application domains of our research, the first section in this chapter provides a brief description of the content delivery systems. The second section introduces the data replica placement (DRP) problem and discusses the related literature on both non-game-theoretical and game-theoretical approaches to DRP problems. The third section defines the types of entities (behavior models) in our study of DRP problems in content delivery systems. We close the chapter by briefly introducing the algorithmic mechanism design theory, the main research tool that we utilized to design solutions to DRP problems in the settings with heterogeneous behavior models.
2.1 Content Delivery Systems

Content delivery networks (CDNs) and peer-to-peer networks (P2Ps) are two major alternative technologies for delivering Internet content. Both have their advantages and disadvantages. With huge sets of servers deployed around the world, CDN companies, such as Limelight (https://www.limelight.com) and Akamai (https://www.akamai.com), are serving geo-distributed data consumers with a higher throughput and better quality of service. Figure 2.1 shows a typical content delivery environment where the storage-capable servers (replication servers or surrogate servers) are located at the edge of the network to which the end-users are connected. A content provider can sign up with a CDN provider for service and have its content placed on the servers. A user (content consumer) is served with the content from the nearby server. Thus, the user ends up unknowingly communicating with a replicated server close to her and retrieves files from that server. Despite of the advantages of CDN systems, the maintenance and management of a large number of server clusters distributed around the globe is challenging and expensive.

On the other hand, P2P networks can achieve high scalability by leveraging the resources of the participating peers and keep the server requirements low. For example, PPLive used less than 10Mbps of server bandwidth to simultaneously serve a 400 kbps video stream to roughly 1.5 million end users [16]. Nevertheless, the decentralized and uncoordinated operation in
P2Ps implies that this scaling comes with undesirable side effects, such as the high startup delay [17] and potential asynchronicity in peer arrival times [18].

Figure 2.1 An example content delivery network.

Giving the complementary advantages of both approaches, researchers have started to proposed hybrid CDN-P2P systems [19, 20], aiming to incorporate the best of both approaches and offset each other’s disadvantages. The hybrid CDN-P2P delivery model is becoming increasingly popular largely due to the rapid growth in the demand for online video and due to both academic interests and industrial efforts. One of the early work was proposed in [20], in which the authors analyzed the service handoff issue between CDN and P2P network, and demonstrated the benefits through simulations. Various applications with hybrid
architecture were then proposed and studied. A detailed survey can be found in [21]. Besides the efforts on designing specific hybrid CDN-P2P systems, a line of work [16, 18, 20, 22-24] has been devoted to investigating the benefits and risks of peer-assistance. Their results suggest that P2P support can greatly decrease the cost of content delivery. In addition to the above academic efforts, a number of commercial or non-commercial PCDNs have been constructed, such as CoralCDN [25], Antfarm [26], Akamai Netsession [22], LiveSky [17, 27], Conviva [28], Xunlei Kankan [29]. VeriSign, CacheLogic, Grid Networks, Internap, and Joost have also announced their own CDN-P2P services as well [24].

2.2 Data Replica Placement Problem and Related Work

Data replication is an important technique in content delivery systems to reduce distributed data access latency, network traffic, and server load in the system, improving system availability, fault-tolerance, and reliability. A typical abstract model for studying the problems in data replication in the existing literature is the distributed replication group model, which was initially proposed in Leff et al. [30] and was later extended to the strategic settings by Laoutaris et al. [31]. The model considers the computing nodes in a distributed network as two groups: the client group - the set of nodes, named clients, that have demands for accessing certain data objects; and the replication group - the set of nodes, named servers (or sites), that have storage capacity for placing the replicas of data objects. To access a data object in this model, a read request from a node in the client group is first received by a local server node in the replication group. If the object
is stored locally, the request is processed in the local server and the object is returned to the requested client immediately, thereby incurring a minimal access cost; otherwise, the requested object is fetched from a remote server in the replication group with a higher cost that is proportional to the distance between the remote server and the local server [31]. To ensure the consistency, an update of a data object would invoke the transfer of the updated object to all of its replicators, thereby causing the maintenance costs associated with a replica placement scheme. The servers store data replicas to supply clients' demands better through the optimization of the network performance, e.g. minimizing the overall access costs and maintenance costs.

Based on the above modeling, the objective of data replica placement (DRP) problem is to find an optimal placement of data replicas within the replication group such that clients' demands can be satisfied with minimal access and maintenance costs. DRP problems have been proven to be NP-Hard [32], so researchers have proposed various techniques to approximate the optimal solution to the problem.

There are a number of previous efforts [30, 32-57] on non-game theoretical versions of the DRP problem. Greedy [42] iteratively determines the replica location for each data object by greedily picking the server that would generate the highest impact on reducing the cost for an object. GRA [51] is a genetic algorithm that exploits mix-and-match technique to solve the DRP problem. It uses chromosomes to encode various replication schemas, and
performs a series of selection, crossover, and mutation processes to find the best chromosome. In [32], the authors abstract the problem with a slightly different model and propose a LP-rounding-based 10-approximation algorithm for the problem with identical object length in arbitrary networks. In [55] the authors propose a distributed 2-approximation algorithm to solve a simplified version of DRP problem, in which the object size is ignored, the capacity constraints bound the number of objects, and only the read requests are considered.

DRP algorithms for PCDNs remain sparse. Jiang et al. [58] introduce the DRP problem in hybrid CDN-P2P and propose a centralized algorithm that takes the peer contribution into consideration; then Wang et al. [59] propose a decentralized replica placement algorithm for the ring based PCDNs. Recently, Garmehi et al. [60] introduce an economic model of PCDN and propose an algorithm that optimizes the number and places of replicas for P2P service in PCDN. All the above efforts assume that peers fully comply with designed protocols and honestly contribute their upload bandwidth. Incentives for peer contribution have not been considered in existing DRP algorithms for content delivery in hybrid CDN-P2P systems. Since peer participation has large impact on the performance of replica placement and content delivery in PCDN systems, in this dissertation, we study the DRP problems in PCDNs with simultaneous consideration of peer incentives.

While game theory has emerged as a popular tool to tackle optimization problems, especially in the field of distributed computing, not much attention has
been received in the game-theoretic aspects of data replication [61]. The earliest work that studies the data replication problem in strategic settings where servers act selfishly is possibly [62], wherein the authors propose a basic game and a payment game for the problem, prove the existence of pure Nash equilibrium in both games, and analyze the price of anarchy and the optimistic price of anarchy of the game. However, this work does not consider storage capacity limits on the servers. In [31], the distributed replication group model is extended to the strategic settings where servers act selfishly. The authors consider the capacity constraints on the servers, and derive equilibrium placement strategies, which are obtainable by two computationally efficient algorithms that are realizable through a distributed protocol. In [63], the authors consider the object replication problem in strategic settings where objects act selfishly and propose a discriminatory algorithmic mechanism for WWW content replication. In [61], the authors also use game-theoretical techniques to study the problem in an environment where the servers behave in a self-interested manner and derive a pure Nash equilibrium and pure strategies for the players. In this dissertation, we study the mechanism-design-based solutions to DRP problems in CDNs and PCDNs with heterogeneous behavior models.

2.3 Behavioral Models

In our study of data replication problems, we consider the entities in the systems with heterogeneous behavioral models, including altruistic, self-interested, and malicious behaviors. They are informally defined in the following:
- Altruistic/obedient behavior. Entities with altruistic/obedient behavior have no independent strategic goals and simply do what they are programmed to do.

- Self-interested behavior. Entities with self-interested behavior aim to maximize their own benefit defined by a known utility function. Self-interested nodes will deviate from the suggested protocol if and only if doing so increases their utility from participating in the system.

- Malicious behavior. Entities with malicious behavior are the various sorts of adversaries found in cybersecurity, ranging from honest but curious enemies to Byzantine enemies [64].

2.4 Algorithmic Mechanism Design

Mechanism design is a subfield of microeconomics and game theory [3]. It allows a game designer to use game theory [65] tools to implement desired goals, normally in the form of social choice functions, in problems that involve self-interested players. A mechanism design model consists of $m$ players. Each player $j \in \{1, \ldots, m\}$ has a private information $t_j \in T_j$, known as the player's type. Moreover, it defines a set of strategies $\Theta_j$ for each player $j$. The player can choose any strategy $\theta_j \in \Theta_j$ to input in the mechanism. According to the strategy profile $\theta_1, \ldots, \theta_m$ of all the players, the mechanism calculates an output vector $o = (o_1, \ldots, o_m)$ and a payment vector $p = (p_1, \ldots, p_m)$. The preference of each player from the output is calculated by a valuation function $v_j(T_j, o)$. This is
quantification in terms of a real number to evaluate the output for a player \( j \). The
quasilinear utility of a player is calculated as \( u_j = v_j(T_j, o) - p_j \). This means that
the utility is the combination of output measured by valuation function and the
payment calculated by the mechanism. The mechanism provides a global output
from the input vector and computes a payment for each player. An overview of
mechanism design can be found in [66].

Largely motivated by the thriving of the Internet, there is a growing
research interest in the interactions of game theory and mechanism design with
computer science (e.g., [12, 67]). This line of research has promoted the field of
algorithmic mechanism design (AMD) [11], which combines ideas from
economics, game theory and computer science to design games to satisfy both
economic goals (e.g., utility maximizations) and algorithmic considerations (e.g.,
worst-case polynomial time). A general overview of algorithmic mechanism
design can be found in [68].

The research scope of both MD and AMD are quite large and varied. In
this research, we limit our research on AMD with the following assumptions and
objectives:

- **Quasilinear utility.** We adopt a common assumption in mechanism design
  research that players’ utility functions are quasi-linear.

- **Risk neutrality.** We assume that all participants are risk neutral, which
  means that participants are indifferent between choices with equal
  expected payoffs even if one choice is riskier.
• Social efficiency. Revenue and social welfare (efficiency) maximization are the two central goals of mechanism design. In this project, we focus on implementing a social choice function that maximizes social efficiency.

• Incentive compatibility. We focus on designing incentive compatible mechanisms. A mechanism is said to be incentive compatible if all players profit best when they are truth telling, i.e., truthfully reveal any private information asked for by the mechanism. The incentive compatibility has several degrees, each of which is associated with a solution concept, e.g., dominant strategy equilibrium, ex-post Nash equilibrium, or Bayes-Nash equilibrium.

• Computational efficiency. The allocation problem and payment function in the mechanism must be computable with polynomial-time computations.

In summary, the focus of AMD in our research is to design computationally efficient mechanisms for DRP problems in content delivery systems such that they implement a social choice function defined as the maximization of social welfare, and the mechanisms satisfy incentive compatibility.
Chapter 3

Mechanism Design for Replica Placement in Content Delivery Networks with Self-interested Content Providers

In this chapter, we study the data replica placement (DRP) problem in a strategic setting inspired by practical market-based data replication applications, e.g., content delivery networks (CDNs). Multiple self-interested players with private preferences own data objects for replication. Players compete for storage space among CDN servers for placing replicas with the objective to optimize their own profits. Using mechanism design approach, we consider the problem as a sequential composition of knapsack auctions and design an algorithmic mechanism DRPMECH to aggregate players’ preferences and approximate a social-efficient allocation for the problem. We analyze both the economic and computational properties of DRPMECH, validate the properties using experiments, and compare its performance against a related game-theoretical solution.
3.1 Introduction

The objective of data replica placement (DRP) problem is to find an optimal placement of replicas within the replication group such that clients' demands can be satisfied with minimal access and maintenance costs. DRP problems have been proven to be NP-Hard [32], so researchers have proposed various techniques (e.g., [32, 55, 69]) to approximate the optimal solution to place replicas. Most related efforts assume all participants in the system fully comply with the designed protocols. For a better understanding of DRP problems in realistic distributed systems and services, a number of researchers have extended the problem into strategic settings, i.e., introducing self-interested agents (players) into the system, and have proposed approaches (e.g., [31, 61, 62]) to solve the problem in such environments. In most of existing strategic settings, the replication servers are owned and operated by multiple rational but selfish parties with different goals and preferences. This can be reasonable since parties in different administrative domains may utilize their servers to better support clients in their own domains, and they have obvious incentives to replicate objects such that the benefits of their own domains are maximized.

In this chapter, we model the data objects, another important component in DRP problems, to be owned by different parties with different private preferences. Each party (data owners) would act rationally but selfishly in a game theoretic sense in order to maximize her own profit, which is associated with whether a
replica of her object is placed in a particular replication site. This modeling is inspired by real-world applications, in particular data replication in content delivery networks (CDNs) [70-74]. The CDN architecture may consist of multiple surrogate servers (replication group in our model) to store and deliver data copies that are owned by content providers (data owners in our model). Each content provider has her original server (will be modeled as primary site in Section 3.2) to store her data objects and will pay for space and services in surrogate servers for replicating her content and serving the demands from content consumers (clients in our model). Existing CDNs are proprietary in nature and their objectives consist of revenue maximization and efficient allocation (e.g., social welfare maximization). On the other hand, the self-interested content providers aim at maximizing the benefits from the replication and minimizing their payment for the storage space, network bandwidth, or computing services incurred by the replication.

The main contributions in this chapter include: (1) the DRP problem is formulated and extended to a strategic setting where multiple rational but selfish players own the data objects for replication with different preferences for replica placements; a cost model is designed to measure the profits of players and the social welfare of a placement; and the monotonicity and supermodularity of the cost model are analyzed; (2) using mechanism design approaches, we model the problem as a sequential composition of knapsack auctions and propose a mechanism DRPMECH, which designs the allocation rule, pricing rule, and
bidding strategy, for social welfare maximization; both the economic and computational properties of DRPMECH are analyzed; and (3) elaborate experiments were designed and conducted to validate the properties of the mechanism and compare its performance with related works.

The rest of this chapter is organized as follows. Section 3.2 formulates the problem, defines the cost model, and introduces the strategic settings for the problem. In Section 3.3, we present our mechanism design solution to the problem and analyze its properties; and in Section 3.4, detailed experiments were conducted to evaluate the properties and performance of the mechanism. Section 3.5 concludes this chapter.

3.2 System Modeling and Problem Formulation

In this section, we formulate the problem based on the distributed replication group model [30], design cost metrics, analyze the properties of the cost metrics, and introduce the strategic setting for the problem.

3.2.1 The DRP Problem

The underlying network topology for DRP problem can be modeled as a weighted undirected graph $G = \langle V, E \rangle$, where $V$ is the set of computation nodes in the system and $E$ is the set of links between any two nodes when a communication path exists. A unary weighting function is defined to assign weight to each link representing the distance of communication path between two
end-points. The distances are assumed symmetric in this work. Based on the distributed replication group model [30], a subset $S$ of $V$ is the replication group, consisting of $n$ sites, each of which is storage-capable to hold replicas of data objects and to serve the demands from the client group. The client group contains the remaining set of nodes in the set $V$, each of which is a client that requests data from the replication group. Figure 3.1 shows an example network topology, in which $\{S_0, S_1, S_2, S_3\}$ is the replication group and other nodes, labeled with “c”, are the clients.

![Figure 3.1 An example network topology for data replica placement.](image)

Now we formulate the DRP problem in Definition 3.1, which is based on the previous formulation efforts in [31, 61, 63].

**Definition 3.1** A DRP problem is given by the following ingredients:

- $S$ is a replication group of $n$ sites: $\{1, \ldots, n\}$;
- $C$ is a storage capacity vector for the replication group: $\{C_1, \ldots, C_n\}$;
• **O** is a set of *m* data objects, \{1,\ldots,m\}, to be replicated within the replication group; each data object *j* has a size \(o_j\); it is reasonable to assume that each object's size is no larger than the minimum capacity in \(C\), i.e., \(0 < o_j \leq \min\{C_1,\ldots,C_n\}\), \(\forall j \in \{1,\ldots,m\}\);

• **Pr** is a vector storing the primary sites of objects: \(Pr_j = i, j \in \{1,\ldots,m\}\), \(i \in \{1,\ldots,n\}\), means that site *i* holds the primary copy of data object *j*;

• **R** is an \(n \times m\) matrix representing objects' read demands; \(R_{ij} \in \mathbb{R}^+\) is the read demands of object *j* addressed to site *i*;

• **U** is an \(n \times m\) matrix representing objects' update (write) demands; \(U_{ij} \in \mathbb{R}^+\) is the update demands of object *j* addressed to site *i*;

• **X** is an \(n \times m\) matrix encoding a placement of *m* objects in *n* sites: \(X_{ij} = 1\) means site *i* has a replica of data object *j*, otherwise \(X_{ij} = 0\);

• **\(\sigma\)** is a cost model measuring the cost associated with a specific instance of the DRP problem: \(\sigma : (S, C, O, Pr, R, U, X) \rightarrow \mathbb{R}\);

• **SCS** is the storage capacity constraint that \(\sum_{j=1}^{m} X_{ij}o_j \leq C_i, i = 1,\ldots,n\);

• **PCS** is the primary site constraint that there exists a primary copy of each object and it cannot be removed from its primary site specified in **Pr**.
Under the constraints of SCS and PCS, given $S$, $C$, $O$, $Pr$, $R$, and $U$, the goal of a DRP problem is to find an optimal $X$ such that $\sigma$ is minimized, i.e.,

$$\arg\min_X \sigma : (S,C,O,Pr,R,U,X)$$

s.t.  \(X_{ij} \in \{0,1\}, i = 1,\ldots,n, j = 1,\ldots,m\)  \(\sum_{j=1}^{m} X_{ij} o_j \leq C_i, i = 1,\ldots,n\)  \(Pr_j \in \{1,\ldots,n\}, j = 1,\ldots,m\)  \(X_{Pr_j,j} = 1, j = 1,\ldots,m\)

Equation (3.2) specifies the storage capacity constraint to ensure that the placement matrix is feasible with regard to the storage capacities of the replication group; Equations (3.3) and (3.4) specify the primary site constraint such that each object has a primary site and an object's primary copy shall not be removed from its primary site. It is to be noted that not all of the related DRP formulations specify the primary site constraint. This work considers the primary site constraint according to [61].

### 3.2.2 Cost Model

In DRP problem, a cost model $\sigma$ is defined to measure the cost associated with a replica placement to satisfy read and write demands. In the distributed replication group model, we consider the set of $n$ sites $S$ to provide storage space
for placing replicas. To measure the distance between replication sites, we consider a weighted complete graph of $S$ derived from the original network by assigning the shortest path distance between two sites in the original graph as the distance of the link between the two sites in the derived complete graph. The read and write demands of a data object in a site are obtained by aggregating the demands of the object from its clients.

Given $S$, $C$, $O$, $Pr$, $R$, $U$, and $X$, the cost model $\sigma$ calculates the total cost associated with the replica placement $X$. In general, $\sigma$ includes: read cost $\sigma_r$, which is the cost incurred by cache miss, and write cost $\sigma_w$, which is the cost caused by maintaining the consistency of the replication. A number of related works (e.g., [57, 75]) only consider read cost, while some others (e.g., [43, 61]) consider both the read and write cost.

Following the settings in related literature, we assume that the read cost is negligible if a request can be fulfilled locally by a replication site that contains the content being requested. Therefore the metric of read cost $\sigma_r$ measures the cost incurred by cache miss and it is closely related with the strategies employed by the servers in the replication group to handle cache miss. In CDN, these strategies usually refer to content outsourcing strategies [76]. There are generally three types of content outsourcing strategies:

- **Non-cooperative Pull-Based (NPLB) approach**: client requests are directed to their closest surrogate servers; and if there is a cache miss, surrogate servers pull content from the origin server (primary site).
• **Cooperative Pull-Based (CPLB)** approach: client requests are directed to their closest surrogate servers; and if there is a cache miss, surrogate servers pull content from a nearest server that holds a replica of the requested content. If the original server (primary site) is the closest one, then the content will be pulled from the original server.

• **Cooperative Push-Based (CPSB)** approach: content is pushed to the surrogate servers from the origin and each request is directed to the closest surrogate server or otherwise the request is directed to the origin server. If there is a cache miss, surrogate servers pull content from the origin server.

Many commercial CDN providers, e.g., Akamai [72], use NPLB approach [76], while an academic CDN Coral [77] has implemented the CPLB approach. CPSB approach is still considered as a theoretical approach since it is still at the experiment stage and has not been widely used [76]. Therefore, this work focuses on using NPLB and CPLB to handle cache miss.

We define a metric for $\sigma_r$. Let $X$ denote a replica placement for $m$ objects across $n$ sites and $\sigma_r(X)$ is the total read cost associated with the placement $X$, which can be computed by Equation (3.5).

$$\sigma_r(X) = \sum_{j \in \mathcal{O}} \sigma'_r(S_j)$$  \hspace{1cm} (3.5)

In Equation (3.5), $\sigma'_r : 2^\mathcal{O} \to \mathbb{R}$ measures the cost incurred by the cache miss of $j$’s data object, and $S_j = \{i \mid X_{ij} = 1, i \in S\}$ is a subset of replication servers
that hold object \( j \)'s replicas. When using CPLB, \( \sigma'_r(S_j) \) can be defined as in Equation (3.6), in which \( \text{Pr}_j \) denotes the primary site of \( j \), \( r(i,S_j) \) find a site in \( S_j \) that is the nearest to \( i \), \( d_{i,\text{Pr}_j} \) measures the distance between site \( i \) and site \( \text{Pr}_j \), \( d_{i,r(S_j)} \) measures the distance between site \( i \) and \( r(i,S_j) \), and finally \( \phi_{ij} \geq 0 \) measures \( j \)'s unit-distance cost due to the cache miss at \( i \). When using the NPLB approach, \( \sigma_r \) is equivalent to a special case of Equation (3.6), as shown in Equation (3.7).

\[
\sigma'_r(S_j) = \sum_{i \in S \setminus S_j} \phi_{ij} \min(d_{i,\text{Pr}_j}, d_{i,r(S_j)}) \quad (3.6)
\]

\[
\sigma'_r(S_j) = \sum_{i \in S \setminus S_j} \phi_{ij} d_{i,\text{Pr}_j} \quad (3.7)
\]

The definition of \( \sigma_r \) is general. When using the end-user access time [41], data availability [34], or object transfer cost [51] to detail the design of \( \phi_{ij} \), \( d_{i,\text{Pr}_j} \), and \( d_{i,r(S_j)} \), \( \sigma_r \) can easily be transformed to the cost associated with cache miss in many cost models of related works, e.g., [43, 57, 75]. In Section 3.4, we present the comparative evaluation of our mechanism with an existing mechanism [61] that use object transfer cost. The object transfer cost metric is defined as the cumulative cost of data objects' movements within the replication group for satisfying the read and update demands from the client group. In the following, we use the concept and cost model described in [61] to adjust \( \sigma_r \) to object transfer cost in Equation (3.8).
\[ \sigma_r = \sum_{i=1}^{n} \sum_{j=1}^{m} \left( (1 - X_{ij}) R_{ij} o_j d_{r(i, X_j)} \right) \] (3.8)

\[ X_{ij} = 1 \] if site \( i \) has a replica of object \( j \), and the object transfer cost of a read request to object \( j \) at site \( i \) is the minimum, say 0. \( X_{ij} = 0 \) if site \( i \) doesn’t hold a replica of object \( j \) and \( i \) will request object \( j \) from site \( r(i, X_j) \), the nearest replicator of object \( j \). In such case, the object transfer cost is proportional to \( o_j \), the object size, \( R_{ij} \), the read demands of \( j \) at \( i \), and \( d_{r(i, X_j)} \), the shortest path distance between site \( i \) and site \( r(i, X_j) \).

The write cost \( \sigma_w \) depends on how the updates to data objects are accepted and how the updates are propagated to replicas. Here we assume that every replication server can accept write (update) requests. To ensure data consistency, a write request to an object will be propagated to all replicators of the object. For example, a site \( i \) may have \( U_{ij} \) write requests of data object \( j \), then the object transfer cost incurred by these \( U_{ij} \) write demands would be calculated as follows: if \( X_{ij} = 1 \), the update will be processed locally and transferred to the primary site of object \( j \), then the primary site will transfer the updated object \( j \) to all its replicators; otherwise, if \( X_{ij} = 0 \), the write requests will be passed to the primary site of data object \( j \), then the updated object \( j \) will be transferred to all its replicators. Using the object transfer cost described in [61], the overall write cost, \( \sigma_w \), can be calculated by Equation (3.9) in the following.
\[
\sigma_w = \sum_{j=1}^{m} \sum_{i=1}^{n} \left( (1 - X_{ij}) U_{ij} d_{ij} o_j + X_{ij} \left( \sum_{r=1}^{n} U_{ij} \phi_r d_{ij} \right) \right)
\]  

(3.9)

Monotonicity and supermodularity are useful properties in combinatorial optimization problems, since they enable the development of polynomial-time algorithms with good approximation. To prove these properties for \( \sigma^j_r \), we first introduce the definition of static conditions on the unit-distance cost and distance measurements.

**Definition 3.2** \( \phi_j \), the unit-distance cost of \( j \) at \( i \), is called static if it remains unchanged during the course of data replica placement.

**Definition 3.3** \( d_{ij} \), the distance measurement between any two sites \( i \in S \) and \( r \in S \), is called static if it is independent of \( j \)’s placement and remain unchanged during the course of data replica placement.

Static unit-distance cost is not uncommon in related literature, e.g., in [57, 61, 75]. One example is that \( \phi_j \) can be defined as the product of the size of the object \( j \) and a pre-estimated cache miss probability of \( j \) at \( i \) based on history and certain prediction model. With static unit-distance cost and distance measurements, we present the monotone supermodularity of \( \sigma^j_r \) when using CPLB to handle cache miss and its modularity when using NPLB.

**Definition 3.4** (Monotonicity) Let \( S \) be a finite set, and a function \( f : 2^S \rightarrow \mathbb{R} \) is monotonically decreasing if for any \( A \subseteq B \subseteq S \) we have that \( f(A) \geq f(B) \). For
monotonically increasing, the inequality is reversed.

**Definition 3.5** Let \( S \) be a finite set, and a function \( f : 2^S \rightarrow \mathbb{R} \) is supermodular if for any \( A \subseteq B \subseteq S \) and \( x \in S \setminus B \), \( f(A \cup \{x\}) - f(A) \leq f(B \cup \{x\}) - f(B) \). For submodularity, the inequality is reversed, and it is modular (additive) if it is both supermodular and submodular.

**Proposition 3.1** \( \sigma_r^j : 2^S \rightarrow \mathbb{R} \) is supermodular and monotonically decreasing for each \( j \in O \) under the conditions that (1) CPLB is used to handle cache miss, (2) unit-distance cost and distance measurements are static, and (3) unit-distance cost and distance measurements are non-negative real values.

**Proposition 3.2** \( \sigma_r^j : 2^S \rightarrow \mathbb{R} \) is modular (additive) and monotonically decreasing for each \( j \in O \) under the conditions that (1) NPLB is used to handle cache miss, (2) unit-distance cost and distance measurements are static, and (3) unit-distance cost and distance measurements are non-negative real values.

The proofs of Proposition 3.1 and 3.2 are given in Appendix A.

### 3.2.3 The DRP Problem in Strategic Setting

By "strategic setting", we mean that there exist players in the system who have different preferences and behave rationally but selfishly. The terms “self-interested” and “rational but selfish”, as well as the terms “players” and “agents”, are used interchangeably in this work. Informally, the term "rational" means that a player always takes actions that maximize her own utility, and the term "selfish"
means that a player only cares about how her actions affect her own utility. Traditional DRP research in the discipline of computer science assumes that all players involved in the model faithfully comply with the designed protocols, while most existing research on the DRP problem in strategic settings assumes that each replication site is owned and operated by multiple rational but selfish players with different preferences. This work studies the DRP problem in another strategic setting, in which multiple rational but selfish content providers own the data objects for replication with private preferences for various replica placements.

The goal of a DRP problem in Definition 3.1 is to find an optimal placement such that it minimizes the cost measured by $\sigma$. If given the current placement matrix $X^c$ and $Costsaving = \sigma(X^c) - \sigma(X)$, $X$ is an optimal placement when the $Costsaving$ is maximized. In this way, we can convert the cost minimization problem in Definition 3.1 to an equivalent $Costsaving$ maximization problem under the same constraints. Using this conversion, we can connect the preferences of self-interested players for a placement with the cost savings incurred by their decisions on the placement. The preference of a player for placing a replica at a particular site depends on how much the cost would be reduced from the replication; and the profit (utility) of a player comes from the difference between the cost savings obtained by the player's replication and her payment for occupying the space for the replication. If there is no decrease in the cost or the payment is higher than the cost savings, there is no benefit for the player to place the replica. Players compete for space among the replication sites.
for placing replicas to maximize their own profits. A new placement of replicas is a result of such a competition; and the social welfare (an aggregation of players' preferences) of the new placement is equivalent to the Costsaving between the new placement and the old one.

Instances of DRP problem in the above-mentioned strategic setting can be viewed as instances of social choice problem in strategic settings. In Section 3.3, we propose a mechanism design approach to aggregating the players' preferences and to implement a social choice function that approximates social welfare maximization, which is equivalent to the minimization of the cost.

3.3 The Mechanism

This section presents our algorithmic mechanism, DRPMECH, for the DRP problem in the strategic setting and the analysis of its properties.

In mechanism design, our modeling of the DRP problem in the above-mentioned strategic setting can be naturally formulated as a combinatorial auction as follows. An auctioneer (e.g., the operator of a CDN) is auctioning off the storage space in a replication group $S$ of $n$ sites (e.g., surrogate servers in CDN) simultaneously to $m$ self-interested players $O$ (e.g., content providers in CDN). Each replication site $i \in S$ has $C_i$ units of storage resource. Each player $j \in O$ owns a data object that requires $o_j$ units of storage space at each site. Player $j$ places exactly one copy of her object at each replication site $i \in S$. Each subset of $S$ constitutes a possible placement of $j$'s replicas. Player $j$ values a placement
\( S_j \in 2^S \) at \( v_j(S_j) \). \( v_j(\cdot) \)'s are the private values of the players. The auctioneer determines the allocation of storage space to players based on their bids. If \( j \) gets allocation for placing her object, she pays \( p_j \geq 0 \); otherwise she pays zero. \( j \)'s objective is to maximize her utility, e.g., \( v_j - p_j \) if quasi-linear utility is used.

The goal of mechanism design is to maximize the social welfare \( \sum_j v_j(S_j) \).

Unfortunately, the above combinatorial auction modeling presents challenges to the application of DRP in real-world large-scale scenarios. The simultaneity in combinatorial auctions may involve complex coordination among a potentially large number of replication sites and may cause service disruption due to the large amount of data to be transmitted. In addition, combinatorial auctions require solving computationally hard problems. Even though truthful polynomial-time approximation schemes do exist [14], the running time is still expensive in practice. In fact, no common auction environment is running simultaneous auctions for large set of items [15].

In this work, we consider the sequential composition of auctions, in which each auction sells the space at a replication site. It turns out that, in a sequential composition, the auction at each replication site is a knapsack auction defined in Definition 3.6.

**Definition 3.6** [78] A *knapsack auction* is private value version of knapsack problem. In the auction, an auctioneer is auctioning off space in a knapsack \( i \) of fixed capacity \( C_i \), each player \( j \) would like to place exactly one object in the
knapsack that takes up \( o_j \) amount of space in the knapsack. Player \( j \) values the placement of her object in the knapsack at \( v_j \) and bids at \( b_j \). \( v_j \) is the private value of \( j \). The auctioneer determines the allocation of \( C_i \) to players based on their bids. If \( j \) gets enough allocation for placing her object, she pays \( p_j \geq 0 \); otherwise she pays zero. \( j \)'s objective is to maximize her utility, e.g., \( v_j - p_j \) if quasi-linear utility is used.

Mu’alem and Nisan [79] study the knapsack auction with the objective of social welfare maximization, while Aggarwal and Hartline [78] study the problem with the objective to maximize auctioneer’s revenue. In the present work, we study the sequential composition of knapsack auctions with the objective of social welfare maximization.

**Definition 3.7** A knapsack auction at a replication site \( i \) is comprised of the following ingredients:

- \( C_i \) units of storage resource for auctioning;
- for every player \( j \), a set of types \( T^j_i \), each value \( t^j_i \in T^j_i \) is the player \( j \)'s private information;
- an alternative set \( A^i \), each alternative \( a^i \in A^i \) is an \( m \) binary vector encoding the allocation of objects at a particular site: \( a^i_j = 1 \) means that player \( j \) is allocated with enough space (i.e., \( j \)'s allocated space is greater than or equal to \( j \)'s storage requirement) to place a replica of his object at
Each individual knapsack auction shall induce a game with independent private values and strict incomplete information. Informally, the term "independent private values" means that the utility of a player only depends on his private information and not on other players' information, and the term "strict incomplete information" means that there is no probabilistic information in the model [71]. We define a game induced by an individual knapsack auction as a local game, and a game induced by a sequential composition of knapsack auctions as a global game. Our objective in the mechanism design for DRP is to design an
allocation and payment rule for the auctioneer and a bidding strategy for the players so that they can reach equilibrium in local and/or global games.

In the following, we present the design of our proposed mechanism DRPMECH for the DRP problem in strategic settings. DRPMECH is comprised of the following identical allocation rule, pricing rule, and bidding strategy for each individual knapsack auction \( i \in S \):

- The allocation rule maps a bidding profile \( b^i \) to an alternative \( a^i \in A^i \) by allocating space to players based on a greedy allocation technique proposed in [79]. It first ranks players in descending order of bidding value \( b^i_j \) and bidding density \( b^i_j / o_j \), respectively. Then for each ranking, the allocation rule greedily allocates space to players with the highest rankings until the storage space is not enough or all players are allocated with space. Let \( a^i_v \) denote the alternative selected based on the ranking of bidding values, and \( a^i_d \) be the alternative selected based on the ranking of bidding densities. The allocation rule picks up from \( a^i_v \) and \( a^i_d \) the one with higher social welfare (breaking ties by selecting \( a^i_v \)) as the final alternative \( a^i \) for site \( i \), in which \( a^i_j = 1 \) means player \( j \) can place a replica at site \( i \), and \( a^i_j = 0 \) means player \( j \) has no replica at site \( i \).

- The payment rule determines the payment for each player based on their bids and allocations. If player \( j \) loses in the competition (i.e., \( a^i_j = 0 \)), she
pays 0; otherwise, if player $j$ wins (i.e., $a_j^i = 1$), her payment is calculated by the following two cases: (1) if $a^i$ is the winning allocation, then player $j$ pays the maximum bidding value among the players who lose in $a^i$; and (2) if $a^{d,i}$ is the winning allocation, then player $j$ pays the multiplication of the object size and the maximum bidding density among the players who lose in $a^{d,i}$.

- The players’ bidding strategy is truth-telling, i.e., each player $j$ declares its type $t_j^i$ as his bid $b^i_j$ for space at site $i$, i.e., $b^i_j = s^i_j(t_j^i) = t_j^i$, therefore, the action (bidding) profile is $b^i = t^i$.

The resulting allocation of DRPMECH in a global game is $a = \{a^i | i \in S \}$, the global social welfare is $F(a) = \sum_{i \in S} F^i(a^i)$, and a player’s utility is an aggregation of her utility in each local game $u^i_j = \sum_{s \in S} u^i_j$.

In the following, we analyze the properties of DRPMECH. The analysis is performed with the following assumptions.

- The following information is publicly known, including the replication group $S$ of $n$ sites, the sites' storage capacity vector $C$, the size of each data object, the constraints $SCS$ and $PCS$, the full sequence of the auctions, and current placement matrix $X^c$.

- The following parameters remain unchanged during a global game: the set
of data owners, the unit-distance cost for each data object at each site, and the distances among computational nodes.

- There is no overlapping between any two consecutive knapsack auctions, and after each knapsack auction, the winning allocation and prices become public knowledge.

With the above assumptions, we first show the properties of bidders with cost saving valuation when only considering read cost in their valuation functions. To facilitate the analysis, we define the cost saving valuation in the following.

**Definition 3.8** (Cost saving valuation) Suppose $S^c_j \subseteq S$ is the current placement for $j$, $j$’s cost saving valuation for any placement $S'_j \in 2^S$ is given by

$$v_j(S'_j | S^c_j) = \max \{ \sigma^j(S'_j) - \sigma^j(S^c_j) \}.$$ 

**Definition 3.9** (Read-cost saving valuation) A cost saving valuation is a read-cost saving valuation if $\sigma^j(\cdot) = \sigma'_j(\cdot)$, where $\sigma'_j(\cdot)$ is defined in Equations (3.6) or (3.7).

The following two propositions present the monotonicity and modularity of the read-cost saving valuation.

**Proposition 3.3** For each player $j$, read-cost saving valuation is monotone submodular under the conditions that (1) CPLB is used to handle cache miss, (2) unit-distance cost and distance measurements are static, and (3) unit-distance
distance cost and distance measurements are non-negative real values.

**Proposition 3.4** For each $j$, read-cost saving valuation is monotone increasing and modular (additive) under the conditions that (1) NPLB is used to handle cache miss, (2) unit-distance cost and distance measurements are static, and (3) unit-distance cost and distance measurements are non-negative real values.

Based on the properties of the valuation function, the incentive compatibility of DRPMECH is presented as follows.

**Lemma 3.1** DRPMECH is incentive compatible in each local knapsack auction under the conditions that (1) either NPLB or CPLB is used to handle cache miss, (2) all players have cost saving valuation, (3) unit-distance cost and distance measurements are static, and (4) unit-distance cost and distance measurements are non-negative real values.

**Theorem 3.1** DRPMECH is incentive compatible in the sequential composition of knapsack auctions under the conditions that (1) NPLB is used to handle cache miss, (2) all players have read-cost saving valuation, (3) unit-distance cost and distance measurements are static, and (4) unit-distance cost and distance measurements are non-negative real values.

The social efficiency of DRPMECH is analyzed in the following.

**Proposition 3.5** DRPMECH achieves a $\frac{1}{2}$-approximation ratio to the optimal social welfare of an individual knapsack auction under the conditions that (1)
either NPLB or CPLB is used to handle cache miss, (2) all players have cost saving valuation, (3) unit-distance cost and distance measurements are static, and (4) unit-distance cost and distance measurements are non-negative real values.

For the global approximation, Baev et al. [32] studies a similar problem and proves that the general replica placement problem in arbitrary network is APX-Hard. Here, we present the global efficiency in a specific case in Proposition 3.6.

**Proposition 3.6** DRPMECH achieves a ½-approximation ratio to the optimal social welfare of a sequential composition of knapsack auctions under the conditions that (1) NPLB is used to handle cache miss, (2) all players have read-cost saving valuation, (3) unit-distance cost and distance measurements are static, and (4) unit-distance cost and distance measurements are non-negative real values.

The computational complexity of the algorithmic implementation of DRPMECH is presented as follows:

**Proposition 3.7** DRPMECH can be implemented using an algorithm with complexity $O(nm \log m)$.

The proofs of the results in this section are deferred to Appendix A.
3.4 Evaluation

This section presents our evaluation of DRPMECH. We first present the methods for DRP instance generation; then validate the incentive compatibility of the mechanism, and finally, evaluate the social efficiency of the mechanism.

3.4.1 Environment Simulation and Data Generation

The DRP environment was simulated on a 64-bit 2.67GHz dual core machine with 6GB memory, and the algorithms were implemented in Java 1.6.

- Network topology. To establish the diversity in network topologies, four categories of topologies were used in the experiment: user-defined topology, random topology, power-law topology, and transit-stub topology. GT-ITM [80] and Inet [81] were the topology generators in our experiments. The generation of different topologies is illustrated in the following. (1) User-defined Network Topology. The parameters of the networks in this category were manually defined for specific evaluation purpose. (2) Random Network Topology. A random graph $G = \langle V, E, p \rangle$ is a graph where $\forall e = (u, v) \in E$, $u, v \in V$ is chosen independently with probability $p \in [0,1]$. The edge length was uniformly generated from a user-defined range. (3) Power-law Network Topology. For obtaining the power-law Internet topology [82], we used Inet [81] to generate Autonomous System level topologies, which have similar characteristics of the Internet from November 1997 to June 2000. (4) Transit-stub
Network Topology. We used GT-ITM [80] to generate transit-stub networks, in which nodes (e.g., routers on the Internet) are organized into logical domains, and the nodes within a domain tend to be fairly interconnected within the domain, but rarely connect to nodes outside of the domain.

- Data objects and read/write demands. The number of data objects \( m = |O| \) and the size of data objects were uniformly chosen from user-defined ranges. For each object \( j \in O \), each site \( i \in S \) has a read demand \( R_{ij} \) and a write (update) demand \( U_{ij} \). In our experiments, \( R_{ij} \) is a uniform random variable in the range \((0, 653656)\), i.e., \( R_{ij} \sim \text{Uniform}(0, 653656) \), and \( U_{ij} \) is proportional to \( R_{ij} \), i.e., \( U_{ij} = f_{w/r} \times R_{ij} \). We call \( f_{w/r} \) the write-read ratio and evaluate its impact on the DRP solutions.

- Site capacity. The capacity \( C_i \) for each site \( i \in S \) is proportional to the total size of objects, i.e., \( C_i = f_c \times \sum_j o_j, \forall i \in S, j \in O \). We call \( f_c \) the site capacity ratio and define it as a uniform random variable in the range \((0.1, 1.5)\), i.e., \( f_c \sim \text{Uniform}(0.1, 1.5) \).

- Primary site. The primary site for each object \( j \in O \) is a site that is uniformly chosen from the replication group \( S \) and that has enough capacity for the placement of the object \( j \).
3.4.2 Validation of Incentive Compatibility

The first set of experiments aims to validate the incentive compatibility of DRPMECH and study the impact of bidding strategies on players’ utilities. The DRP environment for the experiments consists of three replication sites \( S = \{S_1, S_2, S_3\} \) and three self-interested players competing for storage space in \( S \) for placing the replicas of their objects with the objective to minimize the cost to supply the read demands. NPLB is used to handle cache miss. To validate the incentive compatibility of DRPMECH, we need to show that the truthful bidding strategy always maximizes a player’s utility regardless of the strategies of other players. To show this, we designed four bidding strategies, including: (1) “TB” - truthful bidding strategy, which means that players bid their true valuations; (2) “LB” – low bidding strategy, which means that players bid a value lower than their true valuations; (3) “HB” – high bidding strategy, which means that players bid a value higher than their true valuations; and (4) “RB” – random bidding strategy, which means that players bid a random value. In Figure 3.2, we show that when player 2 and player 3 are truthful bidding, player 1 maximizes her utility in individual auctions, as well as her aggregated utility in the sequential composition of auctions; while in Figure 3.3, we show that when player 2 and player 3 randomize their bids, player 1 also maximizes her utility in individual auctions, as well as her aggregated utility. The results support the incentive compatibility of DRPMECH as stated in Lemma 3.1 and Theorem 3.1.
Figure 3.2 Truthful bidding in DRPMECH optimizes the utility of player 1 when other players are also truthful bidding. “TB” - truthful bidding strategy, which means that players bid their true valuations; “LB” – low bidding strategy, which means that players bid a value lower than their true valuations; “HB” – high bidding strategy, which means that players bid a value higher than their true valuations; and “RB” – random bidding strategy, which means that players bid a random value. The “Aggregation” group encodes player 1’s aggregated utility in the sequential composition of auctions at three sites using different bidding strategies.
Figure 3.3 Truthful bidding in DRPMECH optimizes the utility of player 1 when other players are randomized bidding. “TB” - truthful bidding strategy, which means that players bid their true valuations; “LB” – low bidding strategy, which means that players bid a value lower than their true valuations; “HB” – high bidding strategy, which means that players bid a value higher than their true valuations; and “RB” – random bidding strategy, which means that players bid a random value. The “Aggregation” group encodes player 1’s aggregated utility in the sequential composition of auctions at three sites using different bidding strategies.

3.4.3 Validation of Local Efficiency

The second set of experiments aims to validate the approximation ratio to the optimal social welfare in each local knapsack achieved by our DRPMECH via
the comparison with $OPT_L$, an algorithm that exhaustively finds the optimal solution at each site. The complexity of $OPT_L$ is $O(2^m)$, where $m$ is the number of data objects. Since $OPT_L$ is time consuming, we applied the random model to generate six small network topologies for this local efficiency validation. Figure 3.4 shows one of the six random networks. The weights between any two nodes in the graph representation of the networks are not the physical distances, but the time to communicate unit size of data objects. For example, in Figure 3.4, the physical distance between node V0 and V1 is 3.2km, the propagation speed on this link (V0, V1) is $2.8 \times 10^8$ m/s (copper wire), the bandwidth on this link is 100 Mbps, the unit size of data is 1 Kbyte, then the weight of link (V0, V1) is calculated as $(3\text{km}/2.8 \times 10^8\text{m/s})+(1\text{ Kbyte}/100\text{ Mbps}) \approx 0.02114\text{ms}$ (for better representation, instead of using $21.14 \times 10^3$, we label the weight of the link (V0, V1) as 21.14 in Figure 3.4). For each network, 10 DRP instances were generated and parameters for generating instances are shown in Table 3.1. Table 3.2 shows the validation results. We can see that DRPMECH achieves near optimal social welfare in each local auction in all the sixty DRP instances; and thereby the Proposition 3.4 is supported.
Figure 3.4 One of the six networks for local efficiency validation.

Table 3.1 Parameters for DRP instances for local efficiency validation.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sites ( n =</td>
<td>S</td>
</tr>
<tr>
<td>Objects ( m =</td>
<td>O</td>
</tr>
<tr>
<td>Site capacity ( C_i )</td>
<td>( C_i = f_c \sum o_j, \forall i \in S, j \in O ); ( f_c \sim \text{uniform}(0.1,1.5) )</td>
</tr>
<tr>
<td>Object size ( o_j )</td>
<td>( 64 \times 1024 \leq o_j \leq 4 \times 1024 \times 1024, \forall j \in O )</td>
</tr>
<tr>
<td>Read demand ( R_{ij} )</td>
<td>( 0 \leq R_{ij} \leq 653656, \forall i \in S, j \in O )</td>
</tr>
<tr>
<td>Write demand ( U_{ij} )</td>
<td>( 0 \leq U_{ij} \leq 653656, \forall i \in S, j \in O )</td>
</tr>
</tbody>
</table>
Table 3.2 Validation of local approximation ratio.

<table>
<thead>
<tr>
<th>Approx. Ratio</th>
<th>V0</th>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
<th>V6</th>
<th>V7</th>
<th>V8</th>
</tr>
</thead>
<tbody>
<tr>
<td>g1</td>
<td>1</td>
<td>0.994</td>
<td>1</td>
<td>1</td>
<td>0.998</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>g2</td>
<td>0.999</td>
<td>0.998</td>
<td>0.999</td>
<td>1</td>
<td>0.998</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>g3</td>
<td>0.999</td>
<td>1</td>
<td>1</td>
<td>0.998</td>
<td>0.999</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.996</td>
</tr>
<tr>
<td>g4</td>
<td>0.997</td>
<td>0.997</td>
<td>0.985</td>
<td>0.982</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>g5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.997</td>
<td>1</td>
<td>0.999</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>g6</td>
<td>1</td>
<td>0.999</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.998</td>
<td>1</td>
<td>-</td>
</tr>
</tbody>
</table>

Approximation ratio is the ratio between the social welfare of DRPMECH and the social welfare of $OPT_L$. g1-g6 are the six randomly-generated network topologies and g1 is presented in Figure 3.4. V0-V8 are the replication sites. In each cell, '-' means the network does not include the site, otherwise it represents the approximation ratio, e.g., 1 in (g1, V0) means that, on average, DRPMECH achieves optimal social welfare in the local game at site V0 in network g1.

3.4.4 Global Efficiency Evaluation

The third set of experiments aims to study of the global efficiency of DRPMECH in a sequential composition of auctions. Several important parameters of DRP were experimented in this section, including the number of sites, the site
capacities, the number of objects, and the data objects' write/read demands. Similar to [61], a performance metric \( \tau \) has been defined for all global performance evaluations. \( \tau \) measures the average percentage of cost savings and is calculated as in Equation (3.10), wherein \( N \) is the number of experiments under the same topology and parameters and \( X_k^0, X_k^1 \), is the placement matrix before, after, the game, respectively, in the \( k \)\(^{th} \) experiment.

\[
\tau = \frac{1}{N} \sum_{k=1}^{N} \left[ \frac{\sigma(X_k^0) - \sigma(X_k^1)}{\sigma(X_k^0)} \times 100\% \right]
\] (3.10)

To examine the global performance of DRPMECH, we conduct experiments to compare DRPMECH against related solutions. Among the existing literatures on the DRP problem, Greedy [42], A\(_e\)-Star [52], GRA [51], MECH [63], and nRAG [61] are the approaches that are similar to ours. We selected MECH for comparative analysis since it is the only mechanism design based approach to DRP problem that we are aware of that models data objects as self-interested players. The rest of this section discusses the results of our comparative experiments of the algorithmic implementation of DRPMECH and MECH (in Figure 2 of [63]). The assumptions and system parameters were adjusted to be the same for the algorithmic implementations of both mechanisms and MECH was executed in sequential mode. GT-ITM [80] and Inet [81] were used to generate network topologies for DRP instances.

First, we analyze the effect of increasing the sites' storage capacities in the
system. The storage capacity for each site is represented as a ratio to the total size of all data objects. The higher the site capacity ratio, the more data objects can be considered for replication during the game. Figure 3.5 shows the impact of the sites storage capacities on the average cost savings $\tau$ under the condition that the number of data objects is $m=100$, the number of sites is $n=50$, and the data objects' write-read ratio is 0.005. The results show that both the average cost savings increases as the increase of site capacities. Compared to MECH, DRPMECH obtains better cost savings. Especially when the site capacity ratio is lower than a value (e.g., 0.7 in Figure 3.5), DRPMECH achieves much more savings than MECH does.

![Figure 3.5](image_url)  
**Figure 3.5** The impact of capacity ratio on the performance of DRPMECH and MECH.
Second, we examine the impact of $m$, the number of data objects, on the performance. When the sites storage capacities are fixed, increasing the number of data objects is actually decreasing the site capacity ratio, whose effect has already been shown in Figure 3.5. To exclude this effect from the experiments of the number of objects, we fixed the site capacity ratio to 75%. Figure 3.6 shows the impact of the number of data objects on the average cost savings $\tau$ under the condition that the number of sites is $n=50$, the write-read ratio of data objects is 0.005, and the capacity ratio of each site is 75%. The results show that both DRPMECH and MECH are scalable in $m$ and have stable performance when the number of data objects increases. In the case of small $m$, both mechanisms show high variation in their performance.

![Figure 3.6](image)

**Figure 3.6** The impact of the number of data objects on the performance of DRPMECH and MECH.
Third, we evaluate the effect of $n$, the number of sites, on the performance. Figure 3.7 shows the impact of the number of sites on the average cost savings $\tau$ under the condition that the number of data objects is $m=100$, the write-read ratio of data objects is 0.005, and the storage capacity ratio of each site is 75%. When the number of replication sites is small, the competition for space can be fierce and increasing the number of replication sites can increase the space for placing replicas and help distribute read demands to close replication sites; and therefore, the cost saving improves. Nevertheless, after a critical number, the cost for maintaining the consistency among replicas becomes greater than the benefit from placing replicas to better serve read demands, resulting that fewer objects have benefits for replication; and therefore the cost saving reduces.

![Graph showing the impact of the number of replication sites on the performance of DRPMECH and MECH.](image)

**Figure 3.7** The impact of the number of replication sites on the performance of DRPMECH and MECH.
Finally, we investigate the influence of write-read ratio of data objects in the system. The average read demands of data objects is usually higher than the average update demands in DRP instances, since the data replica placement techniques are often used to reduce the access time of users’ read demands. In particular, when the data replication techniques are used for caching, the write-read ratio is extremely low. Figure 3.8 shows the impact of the write-read ratio on the average cost savings $\tau$ under the condition that the number of data objects is $m=100$, the number of sites is $n=50$ and the storage capacity ratio of each site is 75%. When the write-read ratio increases, the maintenance cost increases, and fewer data objects have benefits to place replicas, therefore the cost savings decreases.

![Graph showing the impact of write-read ratio on cost savings for DRPMECH and MECH.](image)

**Figure 3.8** The impact of the write-read ratio on the performance of DRPMECH and MECH.

54
In summary, the global efficiency of DRPMECH can be comparable to the efficiency of MECH and DRPMECH achieves better cost savings than MECH does in most our simulated DRP instances.

3.5 Conclusions

This chapter studies the DRP problem in a strategic setting where multiple rational but selfish players own data objects for replication, and proposes a mechanism DRPMECH to design a game for playing the replication in such an environment. We analyze the incentive compatibility, social efficiency, and computational complexity of the mechanism. Elaborate experiments have been conducted to validate the theoretical properties of the mechanism and show that the performance of DRPMECH is comparable with an existing game-theoretical DRP solution. In the future, we shall continue to study the impact of the underlying network topology, especially the networks in real-world data replication applications, on the social efficiency of our mechanism. We shall further investigate both sequential and simultaneous decompositions and implementations of the mechanism for players with more general valuation functions in practical DRP applications.
Chapter 4

Mechanism Design for Replica Placement in Peer-assisted Content Delivery with Self-interested Seeders

In this chapter, we study the data replica placement (DRP) problem in peer-assisted content delivery networks (PCDNs) with the presence of self-interested seeders. We introduce an economic model to consolidate the cost of content delivery, the replica placement, and the incentives for peer participation, and formulate the DRP problem with seeding incentives in PCDNs. A decentralized DRP algorithm, DPRP-IC, is developed to derive DRP schemes that reduce the cost of content delivery in PCDN systems with the presence of self-interested seeders. We design a set of experiments to comparatively evaluate DPRP-IC and the state-of-the-art algorithms. Results show that the DRP algorithms that consider peer contributions have better performance in PCDN; in addition, DPRP-IC incentivizes the contribution of self-interested seeders and further improves the performance of content delivery in PCDN.
4.1 Introduction

In this chapter, we study the DRP problem in hybrid CDN-P2P systems. Based on the coupling model, the hybrid CDN-P2P systems can be classified into two categories [21]: peer-assisted CDN (PCDN) and CDN-assisted P2P. Most of existing approaches [17, 19, 20, 22-24] fall into the first category, in which the P2P network is designed as a complement component for the regular CDN based content delivery. In such systems, user requests are mainly redirected and served by CDN, while the P2P network is applied to improve user experience or alleviate the load stress of the CDN servers. Recent large-scale measurements revealed that substantial savings could be obtained from such approaches, e.g., a reported 70% of server traffic offload in Akamai’s NetSession [22], and over 87% savings for a video-on-demand workload from Conviva [28], suggesting that peer assistance can greatly decrease the cost of content delivery. Nevertheless, the gains from peer assistance can be limited by a number of obstacle factors. Karamshuk et al. [18] identified three obstacle factors, including ISP-friendliness – requiring peers to be within the same Internet service provider (ISP), bit rate stratification – the need to match peers with others needing similar bit rate, and partial participation – some peers choosing not to redistribute content. They investigated the impact of the three obstacles on the potential gains from peer assistance using a month-long trace data and showed that, among the three factors, partial participation affected the gain from peer assistance the most. This suggests that to obtain the benefits of peer assistance, incentives of peers need to be aligned in real-world PCDNs.
In content delivery systems, DRP algorithms are used to decide where to replicate specific content in order to achieve efficient and effective content delivery. While an extensive number of DRP algorithms have been proposed for traditional CDNs [41, 43, 44, 46-51, 53, 54, 56, 57, 70, 83-88], they may not work efficiently in PCDN environment where P2P technology is incorporated to achieve cost efficiency in a two-level hybrid architecture. For example, in traditional CDNs, when the number of user requests from one region increases, there will be a higher preference to place a replica on a nearby PCDN server to reduce delivery cost; however, in PCDNs, the increase in user requests also means a higher probability that the requests can be served by nearby peers [58], reducing the traffic overload in the nearby CDN server. DRP algorithms for PCDN need to take peer contribution into consideration. The research in this area remains sparse. In 2009, Jiang et al. [58] introduced the replica placement problem in hybrid CDN-P2P and proposed a centralized algorithm that takes the peer contribution into consideration; in 2010, Wang et al. [59] proposed a decentralized replica placement algorithm for ring based PCDNs; and in 2014, Garmehi et al. [60] introduced an economic model of PCDN and proposed an algorithm that optimizes the number and places of replicas for P2P service in terms of the profit of the PCDN operator. All the above efforts assume that peers fully comply with designed protocols and honestly contribute their upload bandwidth. Incentives for peer contribution have not been considered in existing DRP algorithms for content delivery in hybrid CDN-P2P systems. While mechanism design was used
in [60], they studied the problem at the PCDN server level, i.e., modeling the PCDN servers as self-interested agents, deciding for a specific server whether to provide CDN service or P2P service. In this work, we propose to study the DRP problems in PCDNs with consideration of peer incentives.

Improving peer contribution has long been a challenge in real-world P2P systems due to the self-interested nature of real-world peers, who in general wish to download more and upload less, unless otherwise motivated. Following the design of PCDN architecture in [19], we considered the use of BitTorrent-like protocols for P2P level content distribution in PCDN. BitTorrent is a file-sharing protocol designed by Cohen [89], and it has turned into one of the most popular P2P file sharing protocols used for various purposes such as video on demand and media streaming. The protocol involves three basic entities: trackers (the central authority that tracks and coordinates content distribution among peers), leechers (peers that have obtained parts of a file and are in the process of acquiring the rest of the parts), and seeders (a special type of peers that possess an entire content and are distributing content to leechers). A tit-for-tat (TFT) mechanism is utilized in BitTorrent to incentivize content contribution among leechers (in which, leechers provide blocks of files to those who have provided them blocks in the past); nevertheless, BitTorrent does not include a mechanism to incentivize a peer who already possesses an entire content to become a seeder. Seeding is of major importance in the performance of BitTorrent-like systems. Carlsson et al. showed that seeding can significantly reduce the delivery cost of content distribution in
PCDN systems [90]. Therefore, we focus on improving peer contribution in PCDN by designing an algorithm that incentivizes the contribution of self-interested seeders. The main contributions are summarized in the following.

- We present a system model for DRP in PCDNs, introduce an economic model that consolidates the delivery cost of PCDN and the incentives of peer participation, and formulate the DRP problem with seeding incentives in PCDN systems.

- We propose DPRP-IC, a decentralized algorithm for replica placement in PCDN, to incentive the upload contribution of self-interested seeders while controlling the payments for seeding incentives, and ultimately reduce the cost of content delivery in the system.

- By applying techniques from algorithmic mechanism design, we prove the seeding incentive compatibility and computational efficiency of the proposed algorithm.

- Experiments were conducted to validate the properties of the proposed method and compared its performance with the state-of-the-art DRP algorithms. Results suggest that DRP algorithms that consider peer contributions have better performance in PCDN; moreover, our approach incentivizes the contribution of self-interested seeders and further improves the performance of replica placement.
The remainder of this chapter is organized as follows. In Section 4.2, we present the system model for DRP in PCDN, introduce an economic model for measuring the quality of DRP schemes, formulate the DRP problem, and underlie necessary system related assumptions. We present our replica placement algorithm and the analysis of its properties in Section 4.3. Comparative evaluations are presented and discussed in Section 4.4. Finally, we conclude this chapter in Section 4.5.

4.2 System Model and Problem Formulation

In this section, we present the architecture and technical assumptions of PCDN systems, introduce an economic model that consolidates the cost of content delivery, replica placement, and incentives of peer participation in PCDN, and formulate the DRP problem with peer contributions and incentives.

Figure 4.1 An abstract architecture of peer-assisted content delivery network.
4.2.1 The Architecture of PCDN

The architecture of PCDN can be abstracted to a two-level hierarchical hybrid model (Figure 4.1), involving three major types of entities - content providers, PCDN servers, and content consumers. Content providers such as media companies and e-commerce vendors produce and maintain a primary copy of the content and upload the content to PCDN for distribution, aiming to obtain the benefits such as server traffic offloading, access latency reduction, reduced geographical barriers, and improved reliability and availability. The PCDN system replicates the content to her servers, pushing the content close to consumers. Consumers generate the demands for accessing the content. In PCDN, consumers are assumed to have upload capability in addition to their download bandwidth, though specific upload decisions are made by individual consumers. Content consumers locating a file to the same PCDN server can form a P2P network, making part of their upload bandwidth available to the system and exchanging the content with each other. To form a PCDN, we assume a tightly-coupled hybrid model [21] of PCDN servers and P2P networks, in which servers and peers closely collaborate to execute content delivery function, including the server acting as a tracker to involve in the construction process of the P2P overlay network and manage P2P nodes to realize P2P delivery. A number of PCDN systems such as LiveSky [17, 27], Netsession [22], PeerCDN [91], and HCDN [19], can be categorized into this tightly-coupled hybrid model [21]. This model also benefits the utilization of chunk-based delivery protocols, such as BitTorrent
[89], which is one of the most popular P2P file sharing protocols used for various purpose such as video on demand and media streaming [91, 92].

The operational model is as follows. Each content provider is assumed to have a primary server that hold her contents and she selects and uploads content to the PCDN for delivery. The process of content delivery in PCDN is divided into two stages: CDN-level distribution and P2P-level distribution. At the CDN-level, servers are geographically deployed, aiming to serve access demands locally. With content pushed towards the network edges, client-perceived latency can be reduced. Following the model in [90], we assume that each server is associated with a unique locality region and the local regions of PCDN servers are non-overlapping. At the P2P level, requests from data consumers (peers) are redirected to a nearby PCDN server and they can concurrently retrieve content from the server, as well as from other peers in the same swarm, which is formed by peers downloading the same content from the same server. For each file, we assume that each peer is local to exactly one region (i.e., one PCDN server), and is considered as remote to all other PCDN servers. In BitTorrent-like protocols, the data are split in smaller parts, called chunks; and thus, to obtain a complete file a user has to obtain all the chunks of the file. In such systems, a leecher is a peer who still needs chunks, and a seeder is a peer who has the complete file. The contribution of upload bandwidth by the leechers and seeders in BitTorrent can help localize network traffic, offload server traffic, and alleviate congestions in the backbone. Nevertheless, in real-world environment, peers are self-interested.
by nature, who in general wishes to download more and upload less, unless otherwise motivated. Without proper incentives, peers may be freeriding (i.e., downloading the content without making upload contributions to the system) or hit-and-run (i.e., leaving the system after obtaining the complete file), resulting in the partial participation of peers, which can greatly affect the gain from peer assistance [18]. We assume the behavior of leechers follows the behavior model in BitTorrent protocols, in which the upload contribution of leechers shall be enforced by a tit-for-tat mechanism. Regarding the behavior model of seeders, we consider the co-existence of two types of seeders (Figure 4.2):

- Altruistic seeders - seeders who always follow the protocols and contribute upload bandwidth whenever they are eligible and able for seeding;
- Self-interested seeders – seeders who strategically contribute their upload bandwidth with the objective to maximize their own benefits.

Self-interested seeders need proper incentives to encourage their upload contribution. Nevertheless, there is no built-in incentive mechanism for seeding in BitTorrent. Therefore, this work focuses on designing incentives for the upload contribution of self-interested seeders.
In the above content delivery process, two main decision making problems are involved. Deciding which PCDN server to place which content is the problem of replica placement, while deciding which servers should serve which content consumers is the problem of request redirection. While some studies propose to solve the two problems jointly [93, 94], most of the related work address the two problems separately. In this work, we treat the two problems separately, and only focus on the replica placement problem.

4.2.2 The Economic Model

With the above modeling of PCDN system, we introduce an economic model to consolidate the cost of content delivery, replica placement, and incentives of peer participation. Figure 4.3 shows the model defining the payments associated with data uploads in PCDN. Briefly, content providers pay to
the ISP for the bandwidth usage of their primary servers and pay to the PCDN operator for their bandwidth usage in PCDN. On the other hand, the PCDN operator pays to the ISP for her bandwidth usage of content delivery; and she also pays the cost for maintaining the content replicas (e.g., the storage cost or the bandwidth cost associated with content updates). Replica placement techniques can be utilized by a PCDN operator to optimize network bandwidth and storage usage, improving her revenue. To improve peer participation, a PCDN operator can use parts of her revenue to incentive the contribution of self-interested seeders.

**Figure 4.3** An economic model of replica placement in PCDN.

The detailed definition of the economic model is described as follows. We consider a PCDN system with \( m = |M| \) servers to deliver a set of \( n = |N| \) data objects (e.g., documents, videos, or software), where \( M \) and \( N \) represents the set
of PCDN servers and data objects, respectively. Each data objects \( i \in N \) is owned by a content provider, who has a primary server to hold the primary copy of \( i \), as well as serving user requests for \( i \) (data upload path (a) in Figure 4.3). To achieve more efficient and reliable delivery, the content provider can choose to replicate her data to PCDN (data upload path (b) in Figure 4.3). The PCDN then replicates the data to servers, redirect user requests to nearby servers, and try to serve the demand locally. We use \( x_{ij} = 1 \) to represent that PCDN server \( j \in M \) stores a replica of data object \( i \) using \( o_i \) amount of storage (\( o_i \in \mathbb{R}^+ \) and \( o_i \leq C_j, \forall i, j \)), where \( C_j \) is the total amount of storage space available in PCDN server \( j \); then \( X = \{x_{ij} | i \in N, j \in M \} \) represents a replica placement scheme of the set of contents \( N \) in the set of servers \( M \).

Let \( K_i \) be the set of data consumers requesting \( i \) and \( d_{ik} \) be the download bit rate of data object \( i \) requested by the consumer \( k \in K_i \), then \( D_i = \sum_{k \in K_i} d_{ik} \) constitutes the demand for content \( i \) in the system. In this work, we aim to study the replica placement problem in PCDN; therefore, it is assumed that the redirection of user requests to the primary servers and each specific PCDN server is determined by an existing request redirection mechanism in PCDN and is considered as input to the problem. Therefore, we have Equation (4.1) for each content \( i \in N \).

\[
D_i = D_i^{PServer} + D_i^{PCDN} = D_i^{PServer} + \sum_{j \in M} D_{ij}^{PCDN} \tag{4.1}
\]
Where $D_i^{PServer}$ represents the total demand of $i$ on the primary server, $D_i^{PCDN}$ represents the total demand of $i$ on the entire PCDN system, and $D_{ij}^{PCDN}$ is the demand of $i$ on each PCDN server $j$. There are three sources that can provide the upload bandwidth to fulfill $D_i$, including the upload bandwidth of the primary server of $i$ (data upload path (a) in Figure 4.3), the upload bandwidth of the PCDN servers (data upload path (c) in Figure 4.3), and the upload bandwidth of peers in the downloading swarms of $i$ (data upload path (d) in Figure 4.3). To satisfy the demand $D_i$, we have Equation (4.2) for each content $i \in N$.

\[
D_i^{PServer} = B_i^{PServer}
\]

\[
D_{ij}^{PCDN} = B_{ij}^{EServer} + B_{ij}^{Peers} \quad (4.2)
\]

\[
B_{ij}^{Peers} = \sum_{l=1}^{L_i} \alpha_{ijl} L_i + \sum_{a=1}^{A_i} \alpha_{ijal} A_i + \sum_{s=1}^{S_i} \alpha_{ijsl} S_i
\]

where $B_i^{PServer}$ represents the upload bandwidth provided by the primary server of $i$, $B_{ij}^{EServer}$ is the bandwidth used by a specific PCDN server $j$ to deliver $i$, and $B_{ij}^{Peers}$ is the upload bandwidth contributed by the peers $K_{ij}$ in the downloading swarm of $i$ at PCDN server $j$. With the use of BitTorrent-like protocols, $K_{ij}$ is composed of leechers $L_{ij}$, altruistic seeders $A_{ij}$, and self-interested seeders $S_{ij}$. Leechers $L_{ij}$ generate the demand for $i$ at server $j$; and during downloading, they also contribute part of their upload bandwidth to the
swarm. Let $\mu_{ijl}^L$ denote the available upload bandwidth of leecher $l \in L_{ij}$, and $\alpha_{ij}^L \in [0,1]$ is the percentage of her upload contribution, then $\sum_{l \in L_{ij}} \alpha_{ij}^L \mu_{ij}^L$ represents the total upload bandwidth contributed by the set of leechers $L_{ij}$.

Seeders are the set of peers who have downloaded all the chunks of the data. As described in the previous section, we consider the existence of two types of seeders, the set of altruistic seeders $A_{ij}$ and the set of self-interested seeders $S_{ij}$.

Let $\mu_{ija}^A$ and $\mu_{ij}^S$ denote the available upload bandwidth of an altruistic seeder $a \in A_{ij}$ and a self-interested seeder $s \in S_{ij}$, respectively, and $\alpha_{ija}^A \in [0,1]$ and $\alpha_{ij}^S \in [0,1]$ is the percentage of the upload contribution of $a \in A_{ij}$ and $s \in S_{ij}$, respectively, then $\sum_{a \in A_{ij}} \alpha_{ija}^A \mu_{ija}^A$ and $\sum_{s \in S_{ij}} \alpha_{ij}^S \mu_{ij}^S$ represents the total upload bandwidth contributed by the altruistic and self-interested seeders, respectively.

The content provider of data object $i$ pays to the ISP for the bandwidth usage of the primary server (payment path (1) in Figure 4.3) and pays to the PCDN for the PCDN server bandwidth (payment path (2) in Figure 4.3). Note that we didn’t consider storage cost here since most CDN today does not charge it. Therefore, the cost associated with the content provider of $i$ is defined as in Equation (4.3), where $w_i^{PS\text{Server}}$ is the unit price that the content provider paid for the bandwidth usage $B_i^{PS\text{Server}}$, and $w_i^{CE}$ is the unit price that the content provider pays to the PCDN operator for the PCDN server bandwidth.
By providing the delivery service for data object $i$, the PCDN operator receives $w_i^{CE} \sum_{j \in M} B_{ij}^{EServer}$ from the content provider of $i$, pays to the ISP for the PCDN server bandwidth usage (payment path (3) in Figure 4.3), and pays the maintenance cost for storing the replicas. Consequently, the cost associated with the PCDN for providing the delivery service of $i$ with regard to a specific replica placement $X_i$ are defined in Equations (4.4).

$$
\sigma_i^{PCDN}(X_i) = \sum_{j \in M} \left\{ x_{ij} w_{ij}^{PCDN} + \left( 1 - x_{ij} \right) \left( w_{ij}^{PCDN} + g_{ij}^{PCDN} \right) B_{ij}^{EServer} + x_{ij} o_z z_{ij}^{PCDN} \right\} (4.4)
$$

If $x_{ij} = 1$, server $j$ can serve the request for $i$ locally, and incur a bandwidth cost $B_{ij}^{EServer} w_{ij}^{PCDN}$ and maintenance cost $o_z z_{ij}^{PCDN}$, where $w_{ij}^{PCDN}$ represents the unit cost of $B_{ij}^{EServer}$, and $z_{ij}^{PCDN}$ represents the unit replica maintenance cost, which can include the storage cost and update cost. When $x_{ij} = 0$, PCDN server $j$ cannot fulfill the demand locally; and thus the demand will be fulfilled remotely according to a data outsourcing policy of PCDN and incur an additional remote access bandwidth cost $g_{ij}^{PCDN}$. So the total cost is $\left( w_{ij}^{PCDN} + g_{ij}^{PCDN} \right) B_{ij}^{EServer}$. By replacing Equations (4.1)-(4.3) in Equation (4.4), we have the full definition of the delivery cost without seeding incentives in Equation (4.5).
\[
\sigma_i^{PCDN}(X_i) = \sum_{j=1}^{j=M} \left( D_{ij}^{PCDN} - B_{ij}^{Peers} w_{ij}^{PCDN} + g_{ij}^{PCDN} (1-x_{ij}) + x_{ij} o_{ij} z_{ij}^{PCDN} + p_{ij} \right) \tag{4.5}
\]

From Equation (4.5), we can see that the higher the peer contribution, the lower the delivery cost. To encourage the upload contribution of self-interested seeders, proper incentives can be provided (payment path (4) in Figure 4.3). Let \( p_{ij} \) represent the amount that the PCDN operator pays to the self-interested seeders in the download swarm of file \( i \) at server \( j \) to incentivize their contribution, then the delivery cost with seeding incentives is defined in Equation (4.6).

\[
\sigma_i^{PCDN}(X_i) = \sum_{j=1}^{j=M} \left( D_{ij}^{PCDN} - B_{ij}^{Peers} w_{ij}^{PCDN} + g_{ij}^{PCDN} (1-x_{ij}) + x_{ij} o_{ij} z_{ij}^{PCDN} + p_{ij} \right) \tag{4.6}
\]

### 4.2.3 Data Replica Placement Problem in PCDN

With the defined economic model in the previous section, one can optimize certain model variables, e.g., the cost of the content providers or the cost of the PCDN operator. In this work, we focus on the latter, and therefore the problem of replica placement in PCDN is to find an optimal placement such that it minimizes the delivery cost of the PCDN with respect to storage constraints.

\[
\arg\min_{X \in 2^{[s,n]}} \sum_{i=1}^{i=s} \sum_{j \in N} \sigma_i^{PCDN}(X_i) \quad \text{s.t.} \quad \sum_{i \in N} o_{ij} \leq C_j, \forall j \in M \tag{4.7}
\]

As discussed in Section 4.1, seeding is important to the performance of P2P level content distribution. We assume the co-existence of altruistic and self-interested seeders. Altruistic seeders will not be affected by the incentive mechanisms. Therefore, our goal is to design a replica placement scheme that
incentives upload contribution of self-interested seeders (i.e., incentive compatible seeding) while minimizing the overall content delivery cost in PCDN.

4.3 Replica Placement with Incentive Compatible Seeding

In order to design the replica placement scheme with incentive compatible seeding for PCDN, algorithmic mechanism design theory is employed. From Equations (4.6) and (4.7), we can see that increasing peer contribution $B_{ij}^{Peers}$ can reduce the delivery cost. From Equation (4.2), $B_{ij}^{Peers}$ is composed of the upload contribution of leechers, altruistic seeders, and self-interested seeders. Based on the system assumptions in Section 4.2, the upload contribution of leechers and altruistic seeders are considered as the input to the problem, and thus we focus on improving the upload contribution of self-interested seeders. Since the set of seeders in each downloading swarm varies and a seeder’s strategy in different swarms may be inconsistent, a system-wide centralized mechanism that properly incentivizes all seeders in all downloading swarms can be highly complex and inefficient given the scale of the PCDN systems. Therefore, in this work, we sought to a decentralized solution, i.e., incentivizing the seeding contribution in the swarm of each file in each server separately.

For the downloading swarm of file $i$ in PCDN server $j$, we consider the following strategic setting. The PCDN server $j$ would like to buy the upload bandwidth from a set of self-interested seeders $S_{ij}$ who already downloaded the
complete chunks of file $i$. Each seeder $k \in S_{ij}$ has $\mu_{ijk}$ amount of upload resources and she has a value $v_{ijk}$ for the contribution of $\mu_{ijk}$ to the system. The value $v_{ijk}$ may include any costs associated with the uploading. The server asks the seeders to report their upload capabilities and the amount of payments that they desire for.

Each seeder reports $(\mu_{ijk}', v_{ijk}')$ to the server; the server receives the reports, determines a subset of seeders $S_{ij}' \subseteq S_{ij}$ as winners, buys the resources from them, and pays $p_{ijk}$ to each winner. The PCDN server’s goal is to maximize the seeding contribution, i.e., $\max_{S_{ij}' \subseteq S_{ij}} \sum_{k \in S_{ij}'} \mu_{ijk}'$, while the objective of each seeder is to maximize her own utility $p_{ijk} - v_{ijk}$ (by assuming quasilinear utility). This setting is essentially a reverse auction (or procurement) with one buyer and multiple self-interested sellers. In traditional distributed computing settings (full information settings), all the $\mu_{ijk}'$s and $v_{ijk}'$s are publicly known, while in the strategic settings (incomplete information settings), $\mu_{ijk}$ and $v_{ijk}$ are considered as private information and are only known to individual seeder $k$. They are called the type of seeder $k$. Seeders are self-interested. They may not be willingly to reveal their true types, and thus their reports may not be consistent with their true types. How to report constitutes the strategy spaces of the seeders. When a seeder reports their true types, the strategy is called truth-telling. Truth-telling is a desirable property that enables to the server to determine the outcomes based on the correct inputs.

In this work, we aim at designing mechanisms with the following properties:
• Incentive compatibility, that is, all self-interested seeders profit best (i.e., maximize their own utilities) when they are truth-telling. The incentive compatibility has several degrees, each of which is associated with a solution concept, such as dominant strategy equilibrium, ex-post-Nash equilibrium, or Bayes-Nash equilibrium. A mechanism is said to be strategyproof if it is incentive compatible in dominant strategy equilibrium, i.e., the dominant strategy of every agent is to truth-telling, regardless of the strategies of other agents. In this work, we aim at designing mechanisms incentive compatible in dominant strategy.

• Computational efficiency. The allocation function and the payment function can be computed in polynomial time.

The seminal work of Vickrey [95] and Clarke [96] pioneered research on strategyproof mechanism designs, which culminated in the general Vickrey-Clarke-Groves (VCG) mechanisms as presented by Groves [97]. Nevertheless, VCG mechanisms requires the computation of an optimal allocation that maximizes the social welfare, which can be inefficient in PCDN systems that are typically consisting of a large number of servers and data objects. In addition, the VCG mechanisms do not have an upper bound on the payment; while in our problem, to minimize the delivery cost, the total payments shall be kept as small as possible. In order to maximize seeding contribution while controlling the payment, we sought to a new type of mechanisms, called budget feasible mechanisms, which were initially studied by Singer [98]. More specifically, we
model the seeding procurement in each swarm as a budget feasible knapsack procurement auction: given a budget and a set of self-interested seeders \( S_y \), each with upload capacity \( \mu_{ijk} \) and valuation \( v_{ijk} \), find a subset \( S'_y \subseteq S_y \) that maximizes \( \sum_{k \in S'_y} \mu_{ijk} \) while the total amount of payments is under the budget constraints. In [98], Singer proposed a proportional share allocation rule for solving the problem and proved that it can be used to design budget-feasible incentive-compatible mechanisms with good approximation guarantees. Nevertheless, in [98], the problem setting is the single-parameter setting, i.e., the resources of each agent is known to the mechanism and only the valuation is the private information. Here, we extend their results to a more general setting, i.e., both the resources and valuations are private information. This is the setting with unknown self-interested seeders (based on the definition of “unknown agents” in [14]) and it is more close to real-world situations where peers are heterogeneous.

In the following, we first present a decentralized scheme for replica placement in PCDN systems, and then we integrate a mechanism into the algorithm to make the integrated algorithm seeding incentive compatible in each file downloading swarm in order to encourage the upload contributions of self-interested seeders.

The decentralized PCDN replica placement (DPRP) is a decentralized variant of the greedy algorithm [49, 88] proposed for replica placement in traditional CDNs. DPRP takes the upload contributions of leechers and altruistic
seeders into consideration when calculating the contribution of replicating an object at a server to the delivery cost reduction (Figure 4.4).

**DPRP: Decentralized PCDN Replica Placement Scheme**

At PCDN server $j \in M$

1. For each data object $i \in N$, calculate the contribution of replicating $i$ to the delivery cost reduction:

   $$R_{ij} = \max \left\{ 0, \left( D_{ij}^{PCDN} - \sum_{l \in L_i} \alpha_{ijl}^L \mu_{ijl}^L - \sum_{a \in A_i} \alpha_{ija}^A \mu_{ija}^A \right) g_{ij}^E - z_{ij}^E o_{ij} \right\}$$

2. Let $N' = \{ i \in N, R_{ij} > 0 \}$

3. Order all objects in $N'$ such that $R_{1,j} \geq R_{2,j} \geq \cdots \geq R_{|N'|,j}$

4. Let $y = 1$, $C'_j = 0$, and $Y = \emptyset$

5. While $y \leq |N|$ and $(C'_j + o_y) \leq C_j$
   a. $Y = Y \cup \{ y \}$, $C'_j = C'_j + o_y$, and $y = y + 1$

6. Let $Y'$ be the original identities (i.e., their indices in $N$) of data objects in $Y$

7. For each $i \in N$, $x_{ij} = 1$ if $i \in Y'$, otherwise, $x_{ij} = 0$

Return: replica placement scheme in server $j$: $X_j = \{ x_{ij} \mid i \in N' \}$

*Figure 4.4* The decentralized PCDN replica placement scheme.
DPRP-IC: DPRP with Incentive Compatible Seeding
At PCDN server $j \in M$

1. In each downloading swarm $i \in N$
   1.1. $j$ calculates the delivery cost reduction contribution of replicating $i$:
   \[
   R_y = \max \left\{ 0, D_y^{PCDN} - \sum_{a \in E_y} \alpha^L_y \mu_{yj}^L - \sum_{a \in A_y} \alpha^A_y \mu_{yja}^A \right\} 
   \]

1.2. $j$ runs the following procurement with budget $f_R R_y$ ($f_R \in [0,1]$) to buy upload bandwidth from $S_y$, the set of self-interested seeders in the downloading swarm
   1.2.1. $j$ sends messages to $S_y$, asking for reporting upload capacities and valuations
   1.2.2. each peer $k \in S_y$ reports $(\mu^r_{yjk}, v^r_{yjk})$ to $j$
   1.2.3. $j$ selects $S'_y = \{ k \mid k \in S_y, v^r_{yjk} \leq R_y \}$, and orders seeders based on their per cost upload capacity, i.e.,
   \[
   u^r_{yjk}/v^r_{yjk} \geq u^r_{yjk}/v^r_{yjk} \geq \cdots \geq u^r_{yjk}/v^r_{yjk} \]
   1.2.4. let $q = 1$, and $Q = \emptyset$
   1.2.5. While $q \leq |S'_y|$ and $v_{yjk} \leq R_y \left( \frac{\mu^r_{yjk}}{\sum_{k \in Q} \mu^r_{yjk}} \right)$
   \[
   Q = Q \cup \{q\}; \quad q = q + 1
   \]
   1.2.6. $j$ buys $\mu^r_{yjl}$ from each winning seeder $l \in Q$
   1.2.7. $j$ pays to $l \in Q$: $p_{yjl} = \min \left\{ R_y \mu^r_{yjl}/\sum_{k \in Q} \mu^r_{yjk}, (\mu^r_{yjl}/v_{yjl}/\mu^r_{yjl}) \right\}$

2. Calculate remaining demands:
   \[
   \Delta D_y^{PCDN} = \max \left\{ 0, D_y^{PCDN} - \sum_{l \in L_y} \alpha^L_y \mu^L_y l - \sum_{a \in A_y} \alpha^A_y \mu^A_y a - \sum_{l \in Q} \mu^r_{yjl} \right\}
   \]

3. Execute DPRP at $j$ with $\{ \Delta D_y^{PCDN} \mid i \in N \}$ and obtain $X_j$, the placement of objects in $j$
   Return: replica placement scheme in server $j$: $X_j = \{ x_{ij} \mid i \in N \}$

**Figure 4.5** The decentralized PCDN replica placement with incentive compatible seeding.
To motivate the upload contribution of self-interested seeders, servers can use payment as incentives, however, it will also increase the delivery cost. To control the payment, we estimate the cost saving attributed to placement and use it as the budget for procuring upload bandwidth from self-interested seeders. The decentralized PCDN replica placement scheme with incentive compatible seeding (DPRP-IC) is presented in Figure 4.5. By applying the characterization of budget-feasible truthful mechanisms in [98, 99] and the characterization of truthful mechanisms with unknown agents in [14], we have the following results (proofs are deferred to Appendix B).

**Theorem 4.1** DPRP-IC is computational efficient and incentive compatible in dominant strategy in each download swarm with unknown self-interested seeders.

### 4.4 Evaluation

This section presents the evaluation of our proposed decentralized replica placement schemes for reducing the content delivery cost in PCDN systems.

#### 4.4.1 Environment Setup

The PCDN system was simulated on a 64-bit 2.67GHz dual core machine with 6GB memory, and the algorithms were implemented in R 3.2.2 (x64). In addition to evaluating the performance of the two new algorithms, DPRP and DPRP-IC, which we proposed in Section 4.3, we also evaluated and compared the performance of POP [49] and Greedy-CDN (the Greedy-Global algorithm in [49]), two algorithms originally proposed for DRP in traditional CDNs:
• POP: Each server stores the most popular objects among its content consumers. The popularity can be measured in terms of estimated demands or accumulated demands. The server sorts the objects in decreasing order of popularity and stores as many objects in this order as the storage constraint allows.

• Greedy-CDN: The PCDN operator calculation the cost reduction contribution for all server \( i \) and objects \( j \). Then the operator picks the server-object-pair that has the highest contribution to the delivery cost reduction and stores that object in that server. This results in a new placement. Then the operator re-calculates the contribution under the new placement, picks the server-object-pair that has the highest contribution, and stores that object in that server and obtain another new placement. This process is iterated until no server-object-pairs have positive contribution or until no servers have enough storage space for placement.

Among the four algorithms, DPRP, DPRP-IC, and POP are decentralized schemes, while Greedy-CDN is a centralized scheme. POP and Greedy-CDN are oblivious of peer contributions, DPRP considers peer contributions, and DPRP-IC incentivizes peer contributions.

The impacts of several parameters on the performance of the algorithms were evaluated. If not explicitly specified, the generation of the key parameters is described in the following.
• Data objects (content providers). The number of data objects \( n \) and the size of data objects were uniformly chosen from user-defined ranges.

• PCDN servers. The number of servers \( m \) was uniformly chosen from a user-defined range. The capacity \( C_j \) for each server \( j \in M \) is proportional to the total size of objects, i.e., \( C_j = f_c \sum_{i \in N} o_i \), \( f_c \in \mathbb{R}^+ \).

• Data consumers. We assumed the popularity of data objects in each PCDN server followed the Zipf–Mandelbrot distribution, which is shown to be more realistic for modeling the popularity of P2P files [58]. The number of leechers \( |L_{ij}| \) in the downloading swarm of file \( i \) at server \( j \) was proportional to the popularity of the file \( i \) in server \( j \). The number of altruistic seeders and self-interested seeders were proportional to the number of leechers. We evaluated the impact of the number of self-interested seeders on the performance of the algorithm. We assume \( d_{ijl} = 1 \) Mbps, which means leechers have homogeneous downloading speed. The upload bandwidth of leechers, altruistic seeders and self-interested seeders were randomly generated as follows: \( \mu_{ijl}^L \sim \text{normal}(f_id_{ijl},0.1) \), \( \mu_{ija}^A \sim \text{normal}(f_ad_{ija},0.1) \), and \( \mu_{ijx}^S \sim \text{normal}(f_sd_{ijx},0.1) \), where \( f_i \in [0,1] \), \( f_a \in [0,1] \), and \( f_s \in [0,1] \) are parameters of evaluation.

• Price. The unit price of server bandwidth was generated following a
normal distribution: $w_{ij}^{PCDN} \sim \text{normal}(10,2)$. All the other prices were normalized as a ratio to $w_{ij}^{PCDN}$. We evaluated the impact of the bandwidth price on the performance of the algorithms.

### 4.4.2 Results

Three performance metrics were used in the evaluation, including delivery cost, upload contribution, and payment ratio. The calculation of delivery cost was defined in Equation (4.6). Since the generation of problem instances involved randomness, to make the delivery cost comparable, we normalized the cost within each problem instance, i.e., for each problem instance, dividing the delivery cost of each algorithm by the maximum of the costs obtained by all algorithms. The upload contribution measures the relative upload contributions by the PCDN servers, the altruistic peers, and the self-interested seeders. The payment ratio measures the ratio of the payment for seeding to the total delivery cost. By performing replica placement in PCDN, we aim to reduce the delivery cost, increase upload contribution of self-interested seeders while keeping a low payment ratio. The performance of the algorithms was evaluated against several parameters. In each parameterization, we repeated the problem instance generation and algorithm execution 100 times and took the averages as the performances of the algorithms under the specific parameterization.

Figure 4.6 presents the impact of the maintenance cost on the performance of the algorithms. With the increase of the maintenance cost, the benefits of
replica placement reduced. DPRP, DPRP-IC, and Greedy-CDN were aware of the increase in the maintenance cost and placed fewer replicas in order to balance the benefits of replication and the cost of its maintenance; while on the other hand, POP was oblivious of the maintenance cost, so it may choose a placement scheme with a high maintenance cost. Therefore, the difference in delivery cost between POP and the other algorithms increased with the increase of the maintenance cost (Figure 4.6A). With the increase of maintenance cost, the budget for incentive decreased, fewer self-interested seeders had incentives to contribute upload bandwidth to the system, and thus the percentage of upload contribution by self-interested seeders decreased (Figure 4.6B). Since the payment was controlled by the budget, the payment ratio also decreased (Figure 4.6C).

Figure 4.7 presents the impact of the seeding price on the performance of the algorithms. The seeding price was defined as the average valuation of self-interested seeders for contributing their upload bandwidth. In the experiments, we only increased the seeding price and fixed all the other parameters. DPRP, POP, and Greedy-CDN were oblivious of the seeding price, so their performance was stable in the experiments (Figure 4.7A). While for DPRP-IC, the price increased, but the budget remained the same, so fewer upload bandwidths were purchased; and therefore, the benefits of DPRP-IC reduced along with the increase of seeding price (Figure 4.7A). The upload contribution of self-interested seeders also reduced (Figure 4.7B). Attributed to the fact that the payments in DPRP-IC were
controlled by the budgets (which remained unchanged), the payment ratio was maintained at a low level (Figure 4.7C).

Figure 4.8 presents the impact of the upload capacity of self-interested seeders on the performance of the algorithms. In the experiments, we fixed all the other parameters and increased the average upload bandwidth of the seeders. This has the effect that the budget remained the same, but the unit price reduced, so more upload bandwidth could be purchased without increasing the cost. Therefore, we observed the decrease of the delivery cost achieved by DPRP-IC (Figure 4.8A), the increase of the upload contribution by self-interested seeders (Figure 4.8B), and the maintenance of the payment ratio at a lower level (Figure 4.8C). DPRP, POP, and Greedy-CDN are oblivious of the seeding capacity, so their performance was stable in the experiments (Figure 4.8A).

Figure 4.9 presents the impact of the number of self-interested seeders on the performance of the algorithms. This number was the average of the number of self-interested seeders in each swarm normalized by the swarm size. The increase of the number of self-interested seeders had similar effect as the increase of the seeding capacity did on the delivery cost (Figure 4.9A), the upload contribution (Figure 4.9B), and the payment ratio (Figure 4.9C).
Figure 4.6 The impact of the maintenance cost on the performance of replica placement algorithms.
Figure 4.7 The impact of the seeding prices on the performance of replica placement algorithms.
**Figure 4.8** The impact of the seeding capacity on the performance of replica placement algorithms.
Figure 4.9 The impact of the number of seeders on the performance of replica placement algorithms.
In summary, using the experiments, we showed that the proposed decentralized replica placement scheme improved seeding contribution when seeders are self-interested, controlled the incentives for the seeding contribution, and thus reduced the total cost of content delivery in PCDN.

4.5 Conclusions

In this chapter, we study the DRP problems in PCDNs with simultaneous consideration of peer incentives. We present a system model for DRP in PCDNs, introduce an economic model that consolidates the delivery cost of PCDN and the incentives of peer participation, and formulate the DRP problem with seeding incentives in PCDN systems. We propose DPRP-IC, a decentralized algorithm for replica placement in PCDN, to incentivize the upload contribution of self-interested seeders while controlling the payments for seeding incentives, and ultimately reduce the cost of content delivery in the system. By applying techniques from algorithmic mechanism design, we prove the seeding incentive compatibility and computational efficiency of the proposed algorithm. Experiments are conducted to validate the properties of the proposed method and compare its results with state-of-the-art DRP algorithms. Results suggest that DRP algorithms that consider peer contributions have better performance in PCDN; moreover, our approach incentivizes the contribution of self-interested peers and further improves the performance of replica placement.
Chapter 5

Mechanism Design for Replica Placement in Peer-assisted Content Delivery with Heterogeneous Behavior Models

In this chapter, we study the data replica placement (DRP) problem in peer-assisted content delivery networks (PCDNs) with the co-existence of altruistic, self-interested, and malicious seeders. Two types of adversaries are identified in the system, including input adversary and execution adversary. We design metrics to quantitatively measure the magnitude of malice. The impact of malicious behavior is quantified using the price of malice metric. We then integrate the probability of malice of each seeder into mechanism design and extend DPRP-IC, the algorithm that we proposed in Chapter 4, to a security aware algorithm, named DPRP-IC-SA. We design experiments to evaluate the impact of malicious behavior on the algorithms and show that DPRP-IC-SA is more resilient to malicious attacks.
5.1 Introduction

In the previous chapters, we design mechanisms for the DRP problems in systems with altruistic and self-interested agents. Nevertheless, the pragmatic situation on content delivery systems is that all the three types of agents (altruistic, self-interested, and malicious agents) might co-exist. Mechanisms designed without the consideration of malicious agents might be affected by their malicious behaviors and might no longer be incentive compatible for self-interested agents. In this chapter, we extend the study of DRP in PCDN to a setting with all three types of agents, i.e., in each downloading swarm, we consider the co-existence of three types of seeders (Figure 5.1), including:

- Altruistic seeders - seeders who always follow the protocols and contribute upload bandwidth whenever they are eligible and able for seeding;
- Self-interested seeders – seeders who strategically contribute their upload bandwidth with the objective to maximize their own benefits;
- Malicious seeders – seeders who behave to do harm to others or the whole system, e.g., reducing the utility of other seeders, or performing a denial of service attack.

This work identifies malicious behaviors in the system and investigates how they affect the performance and properties of the algorithms designed
without considering the malicious behavior. We also study the design of security aware mechanism to reduce the impact of malicious behaviors.

![Figure 5.1 Types of peers in a downloading swarm of a server in PCDN.](image)

The remainder of this chapter is organized as follows. In Section 5.2, we present the system model and the notations used in this chapter. In Section 5.3, we identify the adversaries, design quantitative metrics to measure the magnitude of malice and the impact of malice, and experimentally evaluate the impact of malicious behaviors on DPRP-IC, the algorithm that we developed in Chapter 4 for DRP in PCDN with self-interested but without malicious seeders. In Section 5.4, we present DPRP-IC-SA, a security aware decentralized replica placement algorithm for DRP in PCDN, analyze its properties, and comparatively evaluate the impact of malicious behavior on DPRP-IC-SA and DPRP-IC. Finally, we conclude this chapter in Section 5.5.
5.2 System Model

The system model and technical assumptions for the problem in this chapter are almost the same as the ones that we presented in Section 4.2 of Chapter 4. The only difference is that we are now considering three types of seeders: altruistic, self-interested, and malicious seeders. In the following, we summarize the key components in the system.

We consider a PCDN system with \( m = |M| \) servers to deliver a set of \( n = |N| \) data objects (e.g., documents, videos, or software), where \( M \) and \( N \) represents the set of PCDN servers and data objects, respectively. The PCDN replicate the data to servers, redirect user requests to their nearby servers, and try to serve the demand locally. We use \( x_{ij} = 1 \) to represent that PCDN server \( j \in M \) stores a replica of data object \( i \) using \( o_i \) amount of storage (\( o_i \in \mathbb{R}^+ \) and \( o_i \leq C_j, \forall i,j \), where \( C_j \) is the total amount of storage space available in PCDN server \( j \)); then \( X = \{ X_{ij} | i \in N, j \in M \} \) represents a replica placement scheme of the set of contents \( N \) in the set of PCDN servers \( M \).

Let \( K_{ij} \) be the set of data consumers requesting \( i \) from server \( j \). In this work, we aim to study the replica placement problem in PCDN; therefore, it is assumed that the redirection of user requests to the primary servers and each specific PCDN server is determined by an existing request redirection mechanism in PCDN and considered as input to the problem. With the use of BitTorrent-like
protocols in the P2P level content distribution, $K_{ij}$ is composed of leechers and seeders. Leechers $L_{ij}$ generate the demand for $i$ at PCDN server $j$; and during downloading, they also contribute part of their upload bandwidth to the swarm. Let $\mu_{ijl}^l$ denote the available upload bandwidth of leecher $l \in L_{ij}$, and $\alpha_{ijl}^l \in [0,1]$ is the percentage of her upload contribution, then $\sum_{l \in L_{ij}} \alpha_{ijl}^l \mu_{ijl}^l$ represents the total upload bandwidth contributed by the set of leechers $L_{ij}$. Seeders are the set of peers who have downloaded all the chunks of the data. As described in the previous section, we consider the existence of three types of seeders, the set of altruistic seeders $A_{ij}$, the set of self-interested seeders $S_{ij}$, and the set of malicious seeders $W_{ij}$. Let $\mu_{ija}^A$, $\mu_{ijs}^S$, and $\mu_{ijw}^W$ denote the available upload bandwidth of an altruistic seeder $a \in A_{ij}$, a self-interested seeder $s \in S_{ij}$, and a malicious seeder $w \in W_{ij}$, respectively; and $\alpha_{ija}^A \in [0,1]$, $\alpha_{ijs}^S \in [0,1]$, and $\alpha_{ijw}^W \in [0,1]$ is the percentage of the upload contribution of $a \in A_{ij}$, $s \in S_{ij}$, and $w \in W_{ij}$, respectively, then $\sum_{a \in A_{ij}} \alpha_{ija}^A \mu_{ija}^A$, $\sum_{s \in S_{ij}} \alpha_{ijs}^S \mu_{ijs}^S$, and $\sum_{w \in W_{ij}} \alpha_{ijw}^W \mu_{ijw}^W$ represents the total upload bandwidth contributed by the altruistic, self-interested, and malicious seeders, respectively.

The DRP problem in this chapter is defined as in Equation (5.1), in which $\sigma_i^{PCDN}(X_i)$ measures the content delivery cost associated with $X_i$, a placement of data object $i$ in the PCDN servers.
\[
\begin{align*}
\arg\min_{X \in \{0,1\}^{N}} \sum_{i \in N} \sigma_{i}^{PCDN}(X) & \quad \text{s.t.} \quad \sum_{i \in N} o_{ij} \leq C_j, \forall j \in M \\
\end{align*}
\]

\[\sigma_{i}^{PCDN}(X)\] is defined by the Equations (4.1) – (4.6). The only difference is that the \(B^{\text{Peers}}_{ij}\) in Equation (4.2), which measures the upload bandwidth contributed by the peers, is now calculated as defined in Equation (5.2).

\[B^{\text{Peers}}_{ij} = \sum_{a \in L} \alpha_{ij}^{L} \mu_{ij}^{L} + \sum_{a \in A} \alpha_{ij}^{A} \mu_{ij}^{A} + \sum_{a \in S} \alpha_{ij}^{S} \mu_{ij}^{S} + \sum_{a \in W} \alpha_{ij}^{W} \mu_{ij}^{W}\]

5.3 Input and Execution Adversaries

DPRP-IC is the mechanism that we proposed in Chapter 4 for DRP in PCDN with altruistic and self-interested seeders but without malicious seeders. The execution of DPRP-IC mechanism involves two phases, including (1) the input phase – seeders report their upload bandwidth and valuation to the mechanism; and (2) the execution phase – winning seeders obtain their payments and contribute upload bandwidth to the system to provide the seeding service. In traditional mechanism design, we assume that (1) in the input phase, all seeders are self-interested, i.e., they behave with the objective to maximize their own benefits; and (2) in the execution phase, the winning seeders actually contribute upload bandwidth to the system as they reported in the input phase. In this section, we identify malicious behaviors in both phases and analyze their impact on the performance of DPRP-IC.
5.3.1 Input Adversary

In the input phase, a seeder can behave maliciously by reporting information to the mechanism with the objective to reduce the utility of other seeders. This means that the valuation functions of those malicious seeders are not independent, but instead, depend on the valuations of other seeders. We use the approach proposed by Chorppath and Alpcan [100] to design a new valuation function that consolidate the valuations of all the three types of seeders. The new function represents the valuation of a seeder as a convex combination of the valuations of other seeders, as shown in Equation (5.3).

$$y_{jk} = (1 - \eta_{ijk}) y_{jk} + \eta_{ijk} \sum_{t \in K_j, t \neq k} y_{it}$$

(5.3)

$y_{jk}$ represents the valuation function of seeder $k \in K_{ij}$ in the downloading swarm of file $i$ in server $j$, measuring her valuation of contributing her uploading bandwidth to the system. $\eta_{ijk}$ is the parameter that specifies the type of seeder $k$.

$$\eta_{ijk} = \begin{cases} 1, & \text{altrustic} \\ 0, & \text{self-interested} \\ -1, & \text{malicious} \end{cases}$$

(5.4)

We assume that the valuation is single-minded, as defined in Equation (5.4), where $(\mu_{ijk}, v_{ijk})$ is the true type of seeder $k$, and $\Pi_{ijk}$ represents the allocation result. $\Pi_{ijk} = \mu_{ijk}$ means that server $j$ will buy $\mu_{ijk}$ amounts of upload resources from seeder $k$. 
Based on Equations (5.3) and (5.4), the malicious input behavior is equivalent to making the other seeders not be able to get allocated. To achieve this, malicious seeders can overbid to improve their probability of winning in the mechanism. More specifically, we define the following three types of malicious overbidding behaviors for analysis.

- **Proportional overbidding attack (PO):** let \((\mu_{ijk}, v_{ijk})\) be the true type of seeder \(k \in K_{ij}\); a proportionally overbidding attacker \(k\) provides \((\mu_{ijk} + \max_{r \in K_j}(\mu_r), v_{ijk})\) to the mechanism. If the attacker wins, she still contributes upload bandwidth to the system.

- **Random overbidding attack (RO):** let \((\mu_{ijk}, v_{ijk})\) be the true type of seeder \(k \in K_{ij}\); a random overbidding attacker \(k\) provides \((\varepsilon_{ijk} + \max_{r \in K_j}(\mu_r), v_{ijk})\) to the mechanism, where \(\varepsilon_{ijk}\) is uniformly sampled from the range between 0 and \(\max_{r \in K_j}(v_r)\). If the attacker wins, she still contributes upload bandwidth to the system.

- **Proportional overbidding and denial of service attack (PO+DoS):** the attacker performs a proportional overbidding attack and if she wins, she does not contribute the upload bandwidth to the system.
The magnitude of malice is quantified using a metric, named malicious size (MS), as defined in Equation (5.5), where \( W_{ij} \) and \( K_{ij} \) is the set of malicious and all seeders in the downloading swarm of file \( i \) in server \( j \), respectively.

\[
MS = \frac{\sum_{i \in N, j \in M} W_{ij}}{\sum_{i \in N, j \in M} K_{ij}}
\]  
(5.5)

### 5.3.2 Execution Adversary

In the execution phase, a winning seeder can behave maliciously by leaving the system without contributing their declared amount of upload resources to the PCDN system, causing a denial of service like attack to the mechanism. To evaluation the specific impact of the malicious execution behavior on the mechanism, we assume that the execution phase attackers perform truthful telling in the input phase. We use two metrics, malicious size (MS) and malicious volume (MV), to quantify the magnitude of malice in the execution phase. MS is already defined in Equation (5.5). MV is defined in Equation (5.6),

\[
MV = \frac{\sum_{i \in N, j \in M} B_{ij}^W}{\sum_{i \in N, j \in M} B_{ij}^{SUW}}
\]  
(5.6)

where \( \sum_{i \in N, j \in M} B_{ij}^{SUW} \) represents the total upload bandwidth procured from both self-interested and malicious seeders, and \( \sum_{i \in N, j \in M} B_{ij}^W \) represents the total upload bandwidth procured from malicious seeders.
5.3.3 Impact of Malicious Behaviors on DPRP-IC

In this section, we investigate the impact of malicious behaviors on the DPRP-IC mechanism. The PCDN system was simulated on a 64-bit 2.67GHz dual core machine with 6 GB memory, and the algorithms were implemented in R 3.2.2 (x64). We generated DRP problem instances from a real-world trace dataset, named Trace T4’05, downloaded from the Peer-2-Peer Trace Archive [101] (http://p2pta.ewi.tudelft.nl). The trace T4’05 [102] was collected from the LegalTorrents.com during the period between March 22, 2005 and July 17, 2005 with 5 minute sampling interval, containing the traced data of 41 file downloading swarms, including the number of leechers and seeders, total number of completed downloads and traffic of each swarm; it also contained the descriptive information of measured torrents including file name, added time, file size, number of files in each torrent and description. We processed the trace T4’05 data and present its characteristics in Figure 5.2. We randomly distributed the 41 swarms to 10 servers and set the capacity of each server to 50% of the total size of files. To quantify the impact of malicious behavior on the mechanism, we define two metrics:

- The average cost saving ($\tau$) metric measures the average percentage of cost savings and is calculated according to Equation (5.7), wherein $\sigma(\cdot)$ is defined in Equation (5.2), $E$ is the number of experiments under the same topology and parameters, $X^0_k = \{X_{ij} = 0 | i \in N, j \in M\}$ is an empty placement in the $k^{th}$ experiment, and $X^1_k$ is the placement matrix obtained
by an DRP algorithm in the $k^{th}$ experiment. The higher the average cost saving, the better the DRP algorithm.

$$
\tau = \frac{1}{E} \sum_{e=1}^{E} \left[ \frac{\sigma(X^0_e) - \sigma(X^1_e)}{\sigma(X^0_e)} \times 100\% \right]
$$

(5.7)

- The price of malice (PoM) metric measures the reduction in the average cost saving of a DRP algorithm caused by the malicious behaviors, as defined in Equation (5.8), where $\Psi$ represents a DRP algorithm, $\tau_{\psi}$ is the average cost saving of the algorithm $\Psi$, and $\tau'_{\psi}$ is the average cost saving of $\Psi$ with the presence of malicious behavior. The lower the price of malice, the more resilient to malicious attacks the DRP algorithm.

$$
PoM(\Psi) = \frac{\tau_{\psi} - \tau'_{\psi}}{\tau_{\psi}}
$$

(5.8)

The DRP algorithms under evaluation include DPRP and DPRP-IC, two algorithms that we developed in Chapter 4. The DPRP algorithm (Figure 4.4) considers altruistic peer contributions, but it does not consider self-interested and malicious behaviors. The DPRP-IC algorithm considers both altruistic and self-interested behaviors, but it does not consider malicious behavior. At each magnitude of malice, the experiments were repeated 30 times to obtain the average cost saving.
Figure 5.2 The characteristics of the Trace T4’05 data. (A) The popularity of the files. The popularity of a file was defined as the ratio between the demands for the file and the total demands. (B) The leecher/seeder ratio of files. (C) The size of files. In (A), (B), and (C), files were all ranked in the decreasing order of their popularity.
We first evaluate the impact of input adversaries on the DRP algorithms (Figure 5.3). The magnitude of malice was measured by malicious size as defined in Equation (5.5). Zero magnitude means that there are no malicious seeders; with the increase of the magnitude, the proportion of malicious seeders increases; and when magnitude equals to 1, it means that all seeders are malicious. All the three types of malicious behaviors (PO, RO, and PO+DoS) have negative impacts on the performance of DPRP-IC. For all magnitude, the PO+DoS behavior has the most severe impact, followed by the RO behavior, and then the PO behavior. Both the proportional and random overbidding reduce the winning probability of self-interested seeders with high per-cost contributions, causing the servers to pay more but obtain less, and thus the performance reduces. The PO+DoS behavior causes the servers to pay more but obtain nothing, so it has the most severe impact on the performance. It is interesting to see that starting from 0.35, with the increase of the magnitude, the negative impact of RO behavior stops increasing, while the negative impact of PO behavior decreases. At low magnitudes, the competition is mainly among self-interested seeders; with the increase of magnitudes, the competition is between malicious seeders and self-interested seeders; at high magnitudes, the competition is mainly among malicious seeders.

We then evaluate the impact of execution adversaries on the DRP algorithms (Figure 5.4). The magnitude of malice was measured by malicious size (MS) and malicious volume (MV). With the increase of the magnitude, the
performance reduces. MS has a relatively low impact since it measures the number of malicious seeders, who might only have small amounts of resources.

**Figure 5.3** The impact of input adversary on DPRP-IC. (A) The impact was quantified using average cost saving. (B) The impact was measured using price of malice. DPRP-IC (PO), DPRP-IC (RO), and DPRP-IC (PO+DoS) represents the performance of DPRP-IC under proportional overbidding attacks (PO), random overbidding attacks (RO), and proportional overbidding and denial of service attacks (PO+DoS), respectively.
Figure 5.4 The impact of execution adversary on DPRP-IC. (A) The impact was quantified using average cost saving. (B) The impact was measured using price of malice. DPRP-IC (MV) and DPRP-IC (MS) represents the performance of DPRP-IC at different magnitudes of malice measured by malicious volume (MV) and malicious size (MS), respectively.
5.4 Replica Placement with Security-Aware Incentive-Compatible Seeding

From the definition of adversaries in Section 5.3, we observe that for both of the input adversary and execution adversary, a critical step in their attacks is to win in the mechanism. A seeder wins in the DPRP-IC mechanism means that the mechanism will buy the upload resource from the seeder and pay according to the payment rule. The input adversaries overbid to increase their winning probability, causing the system to pay more but obtain less peer contributions, affecting the performance of replica placement and content delivery in PCDN. The execution adversaries also need to win so that they can refuse to contribute the upload resources and cause a denial of service attack to the system. Our approach to mechanism design with the presence of malicious behavior is to make the mechanism be aware of the security risks and reduce the winning probability of malicious seeders. To achieve this, we introduce the probability of malice vector \( \Theta = \{ \theta_t \mid \theta_t \in [0,1], t \in S \cup W \} \), where \( S = \{ S_{ij} \mid i \in N, j \in M \} \) and \( W = \{ W_{ij} \mid i \in N, j \in M \} \) represents the set of self-interested seeders and malicious seeders, respectively, in the whole PCDN system. \( \theta_t \) measures the probability of malice of seeder \( t \). Given \( \Theta \), the security aware mechanism will prefer seeders with both low probability of malice and high per-cost upload contribution. In the following, we first present the design of our security-aware algorithm, then comparatively evaluate the algorithms under malicious attacks, and finally discuss the probability of malice.
The decentralized PCDN replica placement with security-aware incentive-compatible seeding algorithm (DPRP-IC-SA) is extended from the DPRP-IC algorithm that we proposed in Chapter 4. As in DPRP-IC, DPRP-IC-SA models the seeding procurement in each downloading swarm as a budget feasible knapsack procurement auction: given a budget and a set of self-interested seeders \( S_y \), each with upload capacity \( \mu_{ijk} \) and valuation \( v_{ijk} \), find a subset \( S_{ij}' \subseteq S_y \) that maximizes \( \sum_{k \in S_{ij}'} \mu_{ijk} \) while the total amount of payments is under the budget constraints. The difference is that in DPRP-IC-SA, each seeder will also associated with a probability of malice \( \theta_{ijk} \in [0,1] \). In DPRP-IC, a server buys resources from seeders purely based on their declared per-cost contribution \( (\mu_{ijk} / v_{ijk}) \); thus it can be severely affected to malicious behaviors of the winning seeders. To tolerate malicious behaviors, DPRP-IC-SA integrates the probability of malice into the allocation rule, generates two rankings of seeders, one in the decreasing order of their per-cost contribution \( \mu_{ijk} / v_{ijk} \) and the other in the decreasing order of \( ((1 - \theta_{ijk}) \mu_{ijk}) / v_{ijk} \), and then buys resources from seeders who win in both rankings under the budget constraints. The DPRP-IC-SA is presented in Figure 5.5. We have the following result for the algorithm (proofs are deferred to Appendix C).

**Theorem 5.1** DPRP-IC-SA is computational efficient and incentive compatible in dominant strategy in each download swarm with unknown self-interested seeders.
DPRP-IC-SA: Security Aware DPRP-IC

At PCDN server $j \in M$,

1. In each downloading swarm $i \in N$
   1.1. $j$ calculates the cost reduction contribution of replicating $i$:
   \[
   R_{ij} = \max \left\{ 0, D_{ij}^{PCDN} - \sum_{t \in T_{ij}} \alpha_{ij}^t \mu_{ij}^t - \sum_{a \in A_{ij}} \alpha_{ija}^A \mu_{ija}^A \right\} 
   \]
   \[
   z_{ij}^E - z_{ij}^o 
   \]
   \[
   \sum_{t \in T_{ij}} \alpha_{ij}^t \mu_{ij}^t - \sum_{a \in A_{ij}} \alpha_{ija}^A \mu_{ija}^A \right\} \]

1.2. $j$ identifies $S_{ij}$, the set of seeders in the downloading swarm and $
   \Theta_{ij} = \left\{ \theta_{ijk} \mid k \in S_{ij} \right\}$, the probability of malice of the seeders.

1.3. $j$ runs the following procurement with budget $f_R R_{ij}$ ($f_R \in [0,1]$):

   1.3.1. $j$ sends messages to $S_{ij}$, asking for bidding; then each peer $k \in S_{ij}$ reports $(u_{ijk}', v_{ijk}')$ to $j$.

   1.3.2. $j$ selects $S_{ij}' = \left\{ k \mid k \in S_{ij}, v_{ijk}' \leq R_{ij} \right\}$, and generates two rankings, $\Pi^1$ and $\Pi^2$, of seeders in $S_{ij}'$. $\Pi^1$ orders seeders in decreasing order of $\mu_{ijk}' / v_{ijk}'$, while $\Pi^2$ orders seeders in decreasing order of $\left( 1 - \theta_{ijk} \right) \mu_{ijk}' / v_{ijk}'$.

   1.3.3. Using the winner determination procedure in DPRP-IC (Steps 1.2.4 and 1.2.5 in Figure 4.5), $j$ determines a set of winning seeders $Q^1$ based on $\Pi^1$ and a set of winners $Q^2$ based on $\Pi^2$. Let $Q^* = Q^1 \cap Q^2$ and $q^1$ be the largest index of seeders in $Q^1$ according to the ranking $\Pi^1$.

   1.3.4. $j$ buys $\mu_{ijl}'$ from each winning seeder $l \in Q^*$ and pays
   \[
   p_{il} = \min \left\{ R_{ij} \frac{\mu_{ijl}'}{\sum_{l' \in Q} \mu_{ijl}'}, \left( \frac{\mu_{ijl}'}{v_{ijl}'(q^1+1)}, \frac{\mu_{ijl}'}{v_{ijl}'(q^2+1)} \right) \right\}
   \]

2. Calculating remaining demands:
   \[
   \Delta D_{ij}^{PCDN} = \max \left\{ 0, D_{ij}^{PCDN} - \sum_{t \in T_{ij}} \alpha_{ij}^t \mu_{ij}^t - \sum_{a \in A_{ij}} \alpha_{ija}^A \mu_{ija}^A - \sum_{l \in Q^*} \mu_{ijl}' \right\}
   \]

3. Execute DPRP at $j$ with $\left\{ \Delta D_{ij}^{PCDN} \mid i \in N \right\}$ and obtain $X_j$.

Return: replica placement scheme in server $j$: $X_j = \left\{ x_{ij} \mid i \in N \right\}$

Figure 5.5 The decentralized PCDN replica placement with security-aware incentive-compatible seeding.
**Figure 5.6** DPRP-IC-SA is more resilient to malicious attacks. (A) The distribution of the probability of malice. (B) The average cost of the DRP algorithms with and without malicious behavior. (C) The price of malice of DPRP-IC and DPRP-IC-SA under malicious attacks. (D) The percentage of upload resources contributed by self-interested seeders under malicious attacks. In the experiments, the malicious behavior is the denial of service like attack at the execution phase. The magnitude of malice measures the malicious size.
We also use the T4’05 data to generate DRP instance for comparative evaluation of DPRP-IC and DPRP-IC-SA under malicious attacks. The environment setup and the metrics are the same as the ones described in Section 5.3.3. The distribution of the probability of malice of seeders is shown in Figure 5.6A, which means that about 1.3% of seeders are perfectly non-malicious (probability of malice=0), about 0.5% of seeders are completely malicious (probability of malice=1), and the remaining seeders have certain probability of malice. From the results, we can see that (1) without malicious behavior, DPRP-IC obtained better performance (higher average cost saving) than DPRP-IC-SA did (Figure 5.6B); (2) the price of malice of DPRP-IC increased quickly even at small magnitudes of malice (Figure 5.6C); (3) DPRP-IC-SA was able to maintain the price of malice at a low level (10% ~ 15%) at all magnitudes of malice (Figure 5.6C); and (4) DPRP-IC-SA obtained higher upload contribution from self-interested seeders than DPRP-IC did at middle to high magnitudes of malice (0.5 ~ 1) and DPRP-IC-SA achieved about 77.0% of the social welfare obtained by DPRP even in the settings with a high magnitude of malice (Figure 5.6D). The above results suggested that DPRP-IC-SA is more resilient to malicious attacks.

Now we have shown that by taking the probability of malice into consideration, the security aware scheme, DPRP-IC-SA, is more resilient to malicious attacks. At the end of this section, we discuss the approaches to obtain the price of malice. In general, there are two approaches: one is a centralized method, and the other is a decentralized method.
• The centralized method is based on reputation system. In this approach, the PCDN system shall maintain a global reputation system and define the probability of malice of a peer as a monotone function of her reputation which can be calculated based on her previous behaviors in the system. To properly maintain the reputation value of a seeder, the system needs a verifier to confirm that the claimed transfers of content by the seeder actually occurred. A number of verifiers have been proposed in P2P systems and they can be utilized for constructing the reputation system.

• The decentralized method is based on peer trust. In this approach, each agent can have a certain level of trust in another agent based on prior experiences or beliefs. Instead of calculating the probability of malice based on a global reputation value maintained by the PCDN system, this method calculates the probability of malice of a peer by aggregating the trust of other peers in this peer. One major challenge in aggregating the trust information from peers is to encourage the peers with heterogeneous behaviors to truthfully report their trust in other peers. One approach to addressing this challenge is through the use of trust-based mechanism design proposed by Ramchurn et al.[103].

5.5 Conclusions

In this chapter, we study the DRP problem in PCDN systems with the co-existence of altruistic, self-interested, and malicious seeders. We identify two
types of adversaries in the system, including input adversary and execution adversary, and experimentally evaluate the impact of the malicious behaviors on the performance of DPRP-IC, an algorithm designed without considering the malicious behaviors. By integrating the probability of malice of seeders into mechanism design, we extend DPRP-IC to DPRP-IC-SA, a security aware, incentive compatible, and computational efficient replica placement scheme. Experiment results show that DPRP-IC-SA is more resilient to malicious attacks. It maintains a stable price of malice at all magnitudes of malice and achieves about 77.0% of the social welfare obtained by DPRP-IC even in the settings with high magnitude of malice.
Chapter 6

Conclusions and Future Work

6.1 Conclusions

This dissertation studies the problem of data replica placement (DRP), an important technique used in storage-capable distributed networks to improve system availability, reliability, and fault-tolerance. We study the DRP problems in the settings with complex and heterogeneous behavior models.

We first study the DRP problem in traditional content delivery networks (CDNs) with self-interested data owners who compete for the storage space in the CDN servers. Traditional mechanism design models the problem as a combinatorial auction, which can have high computation and communication overhead due to the large size of the problem. Here, we present an alternative model that considers the problem as a sequential composition of knapsack auctions and design an algorithmic mechanism DRPMECH to aggregate players’ preferences and approximate a social efficient allocation for the problem. We
analyze both the economic and computational properties of DRPMECH, validate the properties using experiments, and show that it outperforms a related game-theoretical solutions.

Next, we study the DRP problem in peer-assisted content delivery networks (PCDNs). Few algorithms have been proposed for DRP in PCDNs. Here, we present the first replica placement algorithm for content delivery in PCDN with the presence of altruistic and self-interested seeders. We introduce an economic model to consolidate the delivery cost of PCDN and the incentives of peer participation, and formulate the DRP problem with seeding incentives in PCDN systems. A decentralized algorithm, named DPRP-IC, is developed to derive the DRP schemes to improve the upload contribution of self-interested seeders while controlling the payments for seeding incentives, and ultimately reducing the cost of content delivery in the system. We design a set of experiments to comparatively evaluate DPRP-IC against POP and Greedy-CDN (two RP algorithms without considering peer contribution), as well as DPRP (a decentralized RP algorithm that considers peer contributions). Results show that the DRP algorithm that consider peer contributions have better performance in PCDN; in addition, DPRP-IC incentivizes the contribution of selfish seeders and further improves the performance of DRP in PCDN.

Finally, we study the DRP in PCDN system with altruistic, selfish, and malicious seeders. Two types of adversaries are identified in the system, including input adversary – agents who report information to the mechanism with the
objective to reduce the utility of other agents, and execution adversary – seeders who win in the mechanism but refuse to provide the upload bandwidth to the system, causing a denial of service to the system. We design two metrics to measure the magnitude of malice: malicious volume – the percentage of upload bandwidth being compromised; and malicious size – the number of malicious seeders. The impact of malicious behavior is quantified by price of malice, defined as the percentage of performance being reduced due to malicious behavior. We then integrate the probability of malice of each agent into mechanism design and extend DPRP-IC to a security aware algorithm, named DPRP-IC-SA. We design experiments to evaluate the impact of malicious behaviors on the algorithms and show that DPRP-IC-SA is more resilient to malicious attacks.

6.2 Future Work

Some of the future directions for the approaches presented in this thesis are as follows:

- Online DRP problem and dynamic mechanism design. In this dissertation, we assume that the DRP algorithm is executed periodically at a relatively long time interval, such as one day, or one week; and between two executions, the system collects the information that is needed to estimate the parameters for the computation of optimal placement. This assumption is reasonable due to the high overhead of placement updating operations.
Nevertheless, in current research, we also assume that the entities (e.g., the file objects, the leechers, the seeders, etc.) in a DRP execution are static and remain constant until the next execution. A more practical model will allow existing files to be removed, new files to be added, and peers to leave or join the system during the DRP execution. This is essentially the online variant of the DRP problem. Mechanisms shall also incorporate the dynamicity of the system into consideration. In the future, we shall study the online DRP problems in the content delivery systems with the co-existence of altruistic, self-interested, and malicious behaviors and sought to extend the online mechanism design proposed by Parkes [104] to develop robust algorithms for the online DRP problems.

- Mechanisms with Collusion Attack Resilience. In current research, we identify two types of adversaries in the system, including input adversary and execution adversary, investigate their impact on the properties and performance of the mechanisms, and present an approach to design a security aware solution that is shown to be more resilient to the two types of malicious attackers. Nevertheless, in current research, we assume that attackers are independent, while in real world environments, attackers may collude to compromise the mechanism collaboratively. Traditional mechanisms are vulnerable to collusion behaviors. For example, in Vickrey-Clarke-Groves (VCG) mechanism, bidders can reveal their private information to each other and collude to lower their valuations,
reducing the social efficiency of the mechanism. In the future, we shall extend the DRP problem to the settings with the presence of collusion attacks and investigate the design of collusion-proof mechanisms for the problem.
Bibliography


Z. Wang, H. Jiang, Y. Sun, J. Li, J. Liu, and E. Dutkiewicz, “A k-coordinated decentralized replica placement algorithm for the ring-based


Appendices

A. Proofs of the Results in Chapter 3

Here we present the proofs of the results stated in Chapters 3.

**Proposition 3.1** \( \sigma_r^j : 2^S \rightarrow \mathbb{R} \) is supermodular and monotonically decreasing for each \( j \in O \) under the conditions that (1) CPLB is used to handle cache miss, (2) unit-distance cost and distance measurements are static, and (3) unit-distance cost and distance measurements are non-negative real values.

**Proof.** Fix \( j \). Let \( S_j = \{ i | X_{ij} = 1, i \in S \} \subseteq S \) be the set of sites that hold object \( j \)'s replicas and \( \sigma_r^{j,i}(S_j) \) denote the read cost incurred by the cache miss of \( j \)'s object at site \( i \).

\[
\sigma_r^{j,i}(S_j) = \begin{cases} 
\phi_{ij} \min(d_{p_i}, d_{r(i,S_j)}), & i \in S \setminus S_j \\
0, & i \in S_j
\end{cases}
\]

Therefore \( \sigma_r^j(S_j) = \sum_{i \in S} \sigma_r^{j,i}(S_j) \).

We first prove the monotonicity.

For any \( \phi \subseteq S_j^{(1)} \subseteq S_j^{(2)} \subseteq S \) and \( S_j^{(3)} \in S_j^{(2)} \setminus S_j^{(1)} \), we have that:

\[
\sigma_r^j(S_j^{(1)}) = \sum_{i \in S} \sigma_r^{j,i}(S_j^{(1)})
\]
\[ \sigma_r^j(S_j^{(2)}) = \sum_{i \in S} \sigma_r^{j,i}(S_j^{(2)}) \]

\[ \sigma_r^j(S_j^{(2)}) - \sigma_r^j(S_j^{(1)}) = \sum_{i \in S} \sigma_r^{j,i}(S_j^{(2)}) - \sum_{i \in S} \sigma_r^{j,i}(S_j^{(1)}) \]

\[ = \sum_{i \in S : S_i^{(2)}} \sigma_r^{j,i}(S_j^{(2)}) - \sum_{i \in S : S_i^{(2)}} \sigma_r^{j,i}(S_j^{(1)}) - \sum_{i \in S : S_i^{(2)}} \sigma_r^{j,i}(S_j^{(1)}) \]

\[ = \sum_{i \in S : S_i^{(2)}} \left[ \phi_j \min(d_{ij}, d_{ij(S_i^{(2)})}) - \varphi_j \min(d_{ij}, d_{ij(S_i^{(1)})}) \right] \]

\[ - \sum_{i \in S : S_i^{(1)}} \sigma_r^{j,i}(S_j^{(1)}) \]

Since \( \mu_j \) and distances are non-negative real values, \( \sigma_r^j(S_j^{(2)}) - \sigma_r^j(S_j^{(1)}) \leq 0 \).

Therefore \( \sigma_r^j \) under the defined conditions is monotone decreasing.

Now we prove the supermodularity of \( \sigma_r^j \).

For any \( S_j^{(1)} \subset S \), \( S_j^{(2)} \subset S \), \( S_j^{(1)} \cap S_j^{(2)} = \emptyset \)

and \( z \in S \setminus (S_j^{(1)} \cup S_j^{(2)}) \), we have that:

\[ \sigma_r^j(S_j^{(1)}) = \sum_{i \in S} \sigma_r^{j,i}(S_j^{(1)}) \]

\[ \sigma_r^j(S_j^{(2)}) = \sum_{i \in S} \sigma_r^{j,i}(S_j^{(2)}) \]

\[ \sigma_r^j(\{z\}) = \sum_{i \in S} \sigma_r^{j,i}(\{z\}) \]
\[ \sigma_r^j(S_j(1) \cup S_j(2)) = \sum_{i \in S} \min(\sigma_r^{j,i}(S_j(1)) \sigma_r^{j,i}(S_j(2))) \]

\[ \sigma_r^j(S_j(1) \cup \{z\}) = \sum_{i \in S} \min(\sigma_r^{j,i}(S_j(1)) \sigma_r^{j,i}(\{z\})) \]

\[ \sigma_r^j(S_j(1) \cup S_j(2) \cup \{z\}) = \sum_{i \in S} \min(\sigma_r^{j,i}(S_j(1) \cup S_j(2)) \sigma_r^{j,i}(\{z\})) \]

Then,

\[ \sigma_r^j(S_j(1) \cup \{z\}) - \sigma_r^j(S_j(1)) \]

\[ = \sum_{i \in S} [\min(\sigma_r^{j,i}(S_j(1)) \sigma_r^{j,i}(\{z\})) - \sigma_r^{j,i}(S_j(1))] \]

\[ \sigma_r^j(S_j(1) \cup S_j(2) \cup \{z\}) - \sigma_r^j(S_j(1) \cup S_j(2)) \]

\[ = \sum_{i \in S} [\min(\sigma_r^{j,i}(S_j(1) \cup S_j(2)) \sigma_r^{j,i}(\{z\}) - \min(\sigma_r^{j,i}(S_j(1) \cup S_j(2)) \sigma_r^{j,i}(\{z\})] \]

For each \( i \), let’s consider \( \min(\sigma_r^{j,i}(S_j(1) \cup S_j(2)), \sigma_r^{j,i}(\{z\})) \).

If \( \min(\sigma_r^{j,i}(S_j(1) \cup S_j(2)), \sigma_r^{j,i}(\{z\})) = \sigma_r^{j,i}(\{z\}) \),

then \( \min(\sigma_r^{j,i}(S_j(1)), \sigma_r^{j,i}(\{z\})) = \sigma_r^{j,i}(\{z\}) \);

since \( \sigma_r^j(S_j(1) \cup S_j(2)) \leq \sigma_r^j(S_j(1)) \),

we have that \( \sigma_r^j(S_j(1) \cup S_j(2) \cup \{z\}) - \sigma_r^j(S_j(1) \cup S_j(2)) \).
\[ \geq \sigma'_r(S_j^{(1)} \cup \{z\}) - \sigma'_r(S_j^{(1)}) . \]

If \( \min(\sigma'_r(S_j^{(1)} \cup S_j^{(2)}), \sigma'_r(\{z\})) = \sigma'_r(S_j^{(1)} \cup S_j^{(2)}) \),

then \[ \sigma'_r(S_j^{(1)} \cup S_j^{(2)} \cup \{z\}) - \sigma'_r(S_j^{(1)} \cup S_j^{(2)}) = 0 , \]

since \[ \sigma'_r(S_j^{(1)} \cup \{z\}) - \sigma'_r(S_j^{(1)}) \leq 0 , \]

we have that \[ \sigma'_r(S_j^{(1)} \cup S_j^{(2)} \cup \{z\}) - \sigma'_r(S_j^{(1)} \cup S_j^{(2)}) \geq \sigma'_r(S_j^{(1)} \cup \{z\}) - \sigma'_r(S_j^{(1)}) . \]

Therefore, \( \sigma'_r \) under the defined conditions is monotone decreasing and supermodular.

**Proposition 3.2** \( \sigma'_j : 2^S \rightarrow \mathbb{R} \) is modular (additive) and monotonically decreasing for each \( j \in O \) under the conditions that (1) NPLB is used to handle cache miss, (2) unit-distance cost and distance measurements are static, and (3) unit-distance cost and distance measurements are non-negative real values.

**Proof.** The proof of this proposition follows the Definition 3.4, Definition 3.5, and Proposition 3.1.
**Proposition 3.3** For each player $j$, read-cost saving valuation is monotone submodular under the conditions that (1) CPLB is used to handle cache miss, (2) unit-distance cost and distance measurements are static, and (3) unit-distance cost and distance measurements are non-negative real values.

**Proof.** Fix $j$, Let $S_j^{(1)} \subseteq S$ be the set of sites that hold object $j$’s replicas and $S_j^{(2)} \subseteq S$ be a new placement. Based on Definition 3.8 and 3.9, $j$’s read-cost saving valuation for $S_j^{(2)}$ is:

$$v_j(S_j^{(2)}) = v_j(S_j^{(2)} | S_j^{(1)}) = \max \{0, \sigma_j(S_j^{(1)}) - \sigma_j(S_j^{(2)})\}$$

$\sigma_j(S_j^{(1)})$ can be replaced by a constant, then $v_j(S_j^{(2)})$ is the negation of $\sigma_j(S_j^{(2)})$. Based on Proposition 3.1, $v_j(S_j^{(2)})$ is a monotone increasing and submodular function. Therefore, the read-cost saving valuation under the defined conditions is monotone submodular.

**Proposition 3.4** For each $j$, read-cost saving valuation is monotone increasing and modular (additive) under the conditions that (1) NPLB is used to handle cache miss, (2) unit-distance cost and distance measurements are static, and (3) unit-distance cost and distance measurements are non-negative real values.

**Proof.** The proof of this proposition follows the Definition 3.4, Definition 3.5,
Definition 3.8, Definition 3.9, Proposition 3.1, and Proposition 3.3.

Lemma 3.1 DRPMECH is incentive compatible in each local knapsack auction under the conditions that (1) either NPLB or CPLB is used to handle cache miss, (2) all players have cost saving valuation, (3) unit-distance cost and distance measurements are static, and (4) unit-distance cost and distance measurements are non-negative real values.

Proof. Fix a local knapsack auction at replication site \( i \). Based on the definition of cost saving valuation, all players have non-negative valuations for competing the space at site \( i \). Based on the definition of monotone allocation and critical-value based payment rule in [79], it is easy to show that the allocation rule in DRPMECH for each knapsack auction is a monotone allocation algorithm and the payment rule in DRPMECH is a critical-value based payment function. Then based on Theorem 1 in [79], DRPMECH is incentive compatible for the local knapsack auction at \( i \).

Theorem 3.1 DRPMECH is incentive compatible in the sequential composition of knapsack auctions under the conditions that (1) NPLB is used to handle cache miss, (2) all players have read-cost saving valuation, (3) unit-distance cost and distance measurements are static, and (4) unit-distance cost and distance
measurements are non-negative real values.

**Proof.** Based on Proposition 3.4, players under the defined conditions have monotone increasing and additive valuation. So a player’s utility at an individual auction is independent from her utility in other auctions and the allocation results of other auctions. Therefore, the incentive compatibility of DRPMECH in the sequential composition of knapsack auctions follows from the incentive compatibility of DRPMECH in each individual auction as proved in Lemma 3.1.

**Proposition 3.5** DRPMECH achieves a ½-approximation ratio to the optimal social welfare of an individual knapsack auction under the conditions that (1) either NPLB or CPLB is used to handle cache miss, (2) all players have cost saving valuation, (3) unit-distance cost and distance measurements are static, and (4) unit-distance cost and distance measurements are non-negative real values.

**Proof.** The proof is similar to that in [79]. In an individual knapsack auction at a replication site, let $SW^*$ be the optimal social welfare of this auction. Reorder players in the non-increasing order of their bidding densities, i.e.,

$$b_1/o_1 \geq b_2/o_2 \geq \cdots \geq b_m/o_m,$$

and allocate space to players in this order. Let $k \in \{1, \ldots, m\}$, $\sum_{j=1}^{k} o_j \leq C$ and $\sum_{j=1}^{k} o_j > C$, where $C$ is the storage capacity of the...
site \( i \) that runs this subgame. The upper bound \( SW^* \leq \sum_{j=1}^{k} t'_j \) holds, where \( t'_j \) is the type of player \( j \). Then \( SW^* \leq \sum_{j=1}^{k} t'_j \leq \sum_{j=1}^{k-1} t'_j + \max_j t'_j \leq 2 * DRPMECH \).

**Proposition 3.6** DRPMECH achieves a \( \frac{1}{2} \)-approximation ratio to the optimal social welfare of a sequential composition of knapsack auctions under the conditions that (1) NPLB is used to handle cache miss, (2) all players have read-cost saving valuation, (3) unit-distance cost and distance measurements are static, and (4) unit-distance cost and distance measurements are non-negative real values.

**Proof.** Based on Proposition 3.4, players under the defined conditions have monotone increasing and additive valuation. So the optimal social welfare of an individual auction is independent from the allocation results in other auctions. Therefore, the global \( \frac{1}{2} \)-approximation ratio of DRPMECH in the sequential composition of knapsack auctions follows from the local \( \frac{1}{2} \)-approximation ratio of DRPMECH in each individual auction as proved in Proposition 3.5.

**Proposition 3.7** DRPMECH can be implemented using an algorithm with complexity \( O(nm \log m) \).

**Proof.** In each individual auction, DRPMECH can be implemented using an
algorithm with complexity $O(m \log m)$ since the ranking operations in the
allocation rule of DRPMECH can be implemented in $O(m \log m)$, where $m$ is the
number of data objects. Therefore, the time complexity of DRPMECH in a
sequential composition of $n$ auctions can be implemented in $O(nm \log m)$.

B. Proofs of the Results in Chapter 4

Here we present the proof of Theorem 4.1 in Chapter 4.

**Theorem 4.1** DPRP-IC is computational efficient and incentive compatible in
dominant strategy in each download swarm with unknown self-interested
seeders.

**Proof.** The proof of Theorem 4.1 follows the proofs of the Lemma 4.1 and
Lemma 4.2.

**Lemma 4.1** DPRP-IC is incentive compatible in dominant strategy (i.e.,
strategyproof) in each download swarm where the upload bandwidth and
valuation of the self-interested seeders are unknown.

**Proof.** Fix a downloading swarm of file $i$ in server $j$ and let $S_j$ be the set of self-
interested seeders. The proof of the incentive compatibility of DPRP-IC is
based on the Proposition 1 in [14], which states that the mechanism $M = (f, p^f)$ is strategyproof if the agents are single-minded, $f$ is a monotone and exact allocation algorithm for social welfare maximization problem, and $p^f$ is a critical payment scheme associated with $f$. An allocation rule $f$ is monotone if $i \in Q(f((\mu_k, v_k), (\mu_{-k}, v_{-k}))) \Rightarrow i \in Q(f((\mu'_k, v'_k), (\mu_{-k}, v_{-k})))$ for any $\mu'_k \geq \mu_k$ and $v'_k \leq v_k$, where $Q(f(\mu, v))$ represents the set of winners determined by $f$ given the declarations of the agents $(\mu, v)$. Intuitively, the monotonicity states that a winner remains winning if she can make more contributions with lower costs. The critical value payment scheme associated with the allocation rule is then defined $p^f_k(\mu, v) = \theta^f_k$, if $k \in Q(f(\mu, v))$, otherwise $p^f_k(\mu, v) = 0$. $\theta^f_k$ represents the minimum value that makes $k$ win when she declares $(\mu_k, v_k)$, i.e., $k \in Q(f((\mu_k, v_k), (\mu_{-k}, v_{-k})))$ for any $v_k < \theta^f_k$ and $k \not\in Q(f((\mu_k, v_k), (\mu_{-k}, v_{-k})))$ for any $v_k > \theta^f_k$. The allocation rule $f$ is exact if, for declarations $(\mu, v)$, we have $f(\mu, v)_k = \mu_k$ or $f(\mu, v)_k = \phi$.

In DPRP-IC, we consider unknown agents, that is both their contributions $\mu$ and costs $v$ are unknown to the mechanism; therefore, to show the incentive compatibility of DPRP-IC, we need to show that the allocation rule in DPRP-IC is monotone as defined above, i.e., a winner remains winning if she can make more contributions with lower costs, and the payment rule in DPRP-IC is a critical payment scheme. In DPRP-IC, the allocation rule first selects agents
with costs no greater than the budget, so a selected agent remains being selected in this step if she declares a lower cost or higher contribution. Then the allocation rule in DPRP-IC determines winners in the decreasing order of per-cost contribution $\mu_k/v_k$. Declaring a higher contribution or a lower cost will not decrease the rank of an agent when the declarations of other agents are fixed, making a winning agent remain winning. In addition, to meet budget feasibility [98], a winner declaration $(\mu_k, v_k)$ shall satisfy the condition that $v_k \leq R\mu_k / \left( \mu_k + \sum_{q \in Q^*} \mu_q \right)$, where $R$ is the budget and $Q^*$ is the set of winners ranked in front of $k$. Let $\mu'_k > \mu_k > 0$, then we have that $\mu_k / \left( \mu_k + \sum_{q \in Q} \mu_q \right) < \mu'_k / \left( \mu'_k + \sum_{q \in Q} \mu_q \right)$, and thus $v_k \leq R\mu_k / \left( \mu_k + \sum_{q \in Q} \mu_q \right) < R\mu'_k / \left( \mu'_k + \sum_{q \in Q} \mu_q \right)$, so the budget constraint still holds. Therefore, the allocation rule in DPRP-IC is monotone even when the agents are unknown. The associated payment rule in DPRP-IC has been shown to be a critical payment scheme [98]. The allocation rule in DPRP-IC is exact since the server buys $\mu_k$ from seeder $k$ if $k \in Q(f((\mu_k, v_k),(\mu_{-k}, v_{-k})))$ and buys nothing from $k$ if $k \not\in Q(f((\mu_k, v_k),(\mu_{-k}, v_{-k})))$. In DPRP-IC, each seeder $k$ values the contribution $\mu_k$ at $v_k$ when the server buys all $\mu_k$; otherwise, her valuation is 0; therefore, based on the definition introduced by [105], the seeders in DPRP-IC are single-minded.
In summary, the allocation rule in DPRP-IC is monotone and exact, the payment rule is a critical payment scheme, and the agents in DPRP-IC are unknown and single-minded. Based on the Proposition 1 in [14], DPRP-IC is incentive compatible in dominant strategy in the downloading swarm of file $i$ in server $j$ with unknown self-interested seeders.

**Lemma 4.2** The time complexity of DPRP-IC is $O(n \log n) + O(k \log k)$.

**Proof.** The allocation rule and payment rule in each downloading swarm can be executed in $O(k \log k)$, where $k$ is the maximum number of self-interested seeders in a downloading swarm. The DPRP algorithm can be computed in $O(n \log n)$ at each server, where $n$ is the number of data objects. Since the DPRP-IC is decentralized executed, its total time complexity is $O(n \log n) + O(k \log k)$.

**C. Proofs of the Results in Chapter 5**

Here we present the proof of Theorem 5.1 in Chapter 5.

**Theorem 5.1** DPRP-IC-SA is computational efficient and incentive compatible in dominant strategy in each download swarm with unknown self-interested seeders.
**Proof.** The proof of Theorem 5.1 follows the proofs of the Lemma 5.1 and Lemma 5.2.

**Lemma 5.1** DPRP-IC-SA is incentive compatible in dominant strategy in each download swarm where the upload bandwidth and valuation of the self-interested seeders are unknown.

**Proof.** From Theorem 4.1, DPRP-IC mechanism is incentive compatible in dominant strategy in each download swarm for unknown self-interested seeders. The allocation rule and payment rule in DPRP-IC-SA are the same as the ones in DPRP-IC except that the allocation rule in DPRP-IC-SA includes a second ranking $\Pi^2$ into consideration, which orders seeders in the decreasing order of $(1-\theta_k)\mu_k/v_k$. Since the probability of malice $\theta_k$ is independent of the bidding of the seeder $k$, if seeder $k$ wins based on $\Pi^2$ when bidding $(\mu_k,v_k)$, she will remain to win when she bids $\mu'_k \geq \mu_k$ and/or $v'_k \leq v_k$; therefore the ranking $\Pi^2$ is also monotone and exact. So allocation rule in DPRP-IC-SA is monotone and exact. The payment rule in DPRP-IC-SA remains a critical payment scheme associated with its allocation rule. In summary, the allocation rule in DPRP-IC-SA is monotone and exact, the payment rule is a critical payment scheme, and agents in DPRP-IC-SA are unknown and single-minded. Based on Proposition 1 in [14], DPRP-IC-SA is
incentive compatible in dominant strategy in the swarm of file \( i \) in server \( j \) with unknown self-interested seeders.

**Lemma 5.2** The time complexity of DPRP-IC-SA is \( O(n \log n) + O(k \log k) \).

**Proof.** The allocation rule and payment rule in each downloading swarm can be executed in \( O(k \log k) \), where \( k \) is the maximum number of seeders in a downloading swarm. The DPRP algorithm can be computed in \( O(n \log n) \) at each server, where \( n \) is the number of data objects. Since the DPRP-IC-SA is decentralized executed, its total time complexity is \( O(n \log n) + O(k \log k) \).