# Opportunistic Mobile Data Offloading with Deadline Constraints

Guoju Gao, *IEEE Student Member,* Mingjun Xiao, *IEEE Member,* Jie Wu, *IEEE Fellow,* Kai Han, *IEEE Member,* Liusheng Huang, *IEEE Member,* and Zhenhua Zhao

**Abstract**—Due to the explosive proliferation of mobile cloud computing applications, much data needs to be transmitted between mobile users and clouds, incurring a huge traffic demand on cellular networks. Mobile offloading is a promising approach to address this challenge. In this paper, we focus on the problem of offloading many deadline-sensitive data items to some WiFi networks with capacity constraints; that is, how to schedule each data item to the WiFi networks, so that we can offload as many data items before their deadlines as possible, while taking the constraints of transmission capacity into consideration. This problem involves a probabilistic combination of multiple 0-1 knapsack constraints, which differs from existing problems. To solve this problem, we propose a greedy oFfline Data Offloading (FDO) algorithm, achieving an approximation ratio of 2. Also, we propose an oNline Data Offloading (NDO) algorithm, which has a competitive ratio of 2. Additionally, we extend our problem to a more general scenario where WiFi transmission costs are heterogeneous. We design a Heterogeneous Data Offloading (HDO) algorithm to solve the extended problem, and give its performance analysis. Finally, we demonstrate the significant performances of our algorithms through extensive simulations based on some real-world and synthetic WiFi datasets.

**Index Terms**—deadline-sensitive data offloading, mobile data offloading, opportunistic WiFi offloading.

## 1 Introduction

With the explosive growth of user population and their demands for bandwidth-eager multimedia content in recent years, a big challenge is raised regarding the cellular network. The Cisco VNI [8] report predicts that mobile data traffic will grow at a compound annual growth rate (CAGR) of 53 percent from 2015 to 2020, reaching 30.6 exabytes per month by 2020. Furthermore, the aggregate smartphone traffic will be 8.8 times greater than it is today, with a CAGR of 54 percent by 2020. To cope with the unprecedented traffic load, mobile network operators need to increase their cellular network capacities significantly. However, this is expensive and inefficient. One promising solution to this problem is to offload part of traffic to other coexisting networks, while leaving the capacities of cellular networks unchanged. Some recent research efforts have been focused on offloading cellular traffic to other forms of networks, such as WiFi networks [9, 12, 18–20, 36] and Delay Tolerant Networks (DTNs) [14, 22, 28, 34, 38].

In this paper, we focus on the mobile data offloading based on WiFi networks in mobile cloud computing [15]. Consider the scenario in which a mobile user is performing some mobile cloud computing applications and needs to upload some data items to the cloud side. In order to ensure the quality of the mobile cloud computing applications, each data item needs to be uploaded before a deadline. On the other hand, when the user conducts the mobile cloud computing applications, it can access cellular networks at any time, anywhere. Meanwhile, the user also might pass by some WiFi APs. Hence, the user can transmit the data items through cellular networks directly, or offload some data to WiFi networks, when it visits a WiFi AP, as shown in Fig. 1. In general, the data transmission via cellular networks has the advantage of instantaneity, but it will lead to a large monetary cost. In contrast, data being offloaded to WiFi networks can save a significant monetary cost, but the instantaneity cannot be ensured. There is a trade-off between the two transmission modes, especially when the transmission capacity of WiFi APs is taken into consideration. Our concern is how to schedule data items between the two transmission modes, so that we can minimize the total monetary cost, while ensuring that each data item be uploaded before its deadline.

The proposed data offloading is different from existing offloading problems [6, 12–14, 20, 22, 23, 28, 34, 35, 38]. These works in [14, 22, 28, 34] mainly focus on offloading data from cellular networks to DTNs, which is formulated as a target-set selection problem. Zhuo *et al.* [38] provides an incentive framework based on the reverse auction to leverage the delay tolerance for data offloading based on DTNs. In addition, the works in [12, 20] study the economic benefits and load balance problem of traffic offloading between cellular networks and WiFi networks from the perspective of Network Services Providers (NSPs). In contrast, we consider the data offloading problem from the user's side. Moreover, our problem can be deduced as an optimization

- *G. Gao, M. Xiao*, K. Han, L. Huang and Z. Zhao are with the School of Computer Science and Technology / Suzhou Institute for Advanced Study, University of Science and Technology of China, Hefei, P. R. China. *Correspondence to: xiaomj@ustc.edu.cn*
- *J. Wu is with the Department of Computer and Information Sciences, Temple University, 1805 N. Broad Street, Philadelphia, PA 19122. E-mail: jiewu@temple.edu*

*This paper is an extended version of the conference paper [11] published in IEEE SECON 2016. This research was supported in part by the National Natural Science Foundation of China (NSFC) (Grant No. 61572457, 61379132, U1301256, 61502261, 61303206, 61572342), NSF grants CNS 1629746, CNS 1564128, CNS 1449860, CNS 1461932, CNS 1460971, CNS 1439672, CNS 1301774, ECCS 1231461, and the Natural Science Foundation of Jiangsu Province in China (Grant No. BK20131174, BK2009150).*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2017.2720741, IEEE Transactions on Parallel and Distributed Systems



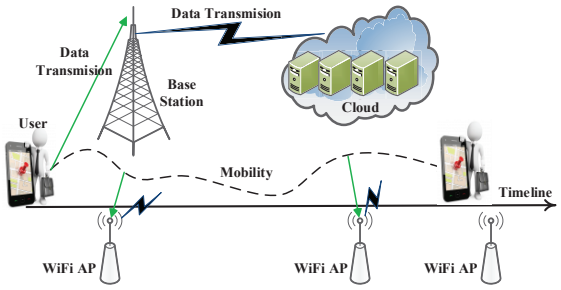



Fig. 1. Data offloading scenario: the mobile user uploads data items onto the cloud side through WiFi networks when it visits WiFi APs during Time-To-Lives (TTLs) of data items, or via cellular networks when the TTLs of data items expire, respectively.

problem with multiple 0-1 knapsack constraints, in which each knapsack is related to a WiFi AP. Adding a data item into a knapsack means offloading this data item via the corresponding WiFi AP. Since the accessibility of each WiFi AP is uncertain, it is a probabilistic event to add a data item into a knapsack. Furthermore, each data item is allowed to be added into multiple knapsacks. Hence, these data items share a combinatorially probabilistic optimization objective. Meanwhile, each data item also needs to be subject to a different deadline constraint. It is because of these features that our problem differs from the existing trivial Multiple Knapsack Problems (MKP) [5, 26], and the existing algorithms (e.g., Shortest Remaining Time First scheduling algorithm) are not applicable in our problem.

To this end, we design an offloading utility function according to the combinatorially probabilistic optimization objective. Based on this utility function, we propose a greedy offline data offloading algorithm to solve the aforementioned problem. Furthermore, we also propose an online data offloading algorithm. The offline algorithm indicates that the mobile user makes the data offloading decisions before it visits any WiFi AP, while the online algorithm means that the mobile user dynamically makes the immediate data offloading decisions at each time when it visits a WiFi AP. Also, we extend our problem and solution to a more general scenario where the transmission costs per unit data traffic via different WiFi APs are heterogeneous. More specifically, our major contributions are summarized as follows:

- We introduce a problem of offloading many deadline-sensitive data items to some WiFi APs with capacity constraints. We formalize it as an optimization problem with multiple 0-1 knapsack constraints, sharing a combinatorially probabilistic optimization objective. Moreover, we prove the NP-hardness of this problem.
- We propose an offline data offloading algorithm, i.e., FDO, to solve the above problem. A greedy strategy is adopted in this algorithm. We prove that this greedy strategy can achieve the approximation ratio of 2.
- We also propose an online data offloading algorithm, i.e., NDO. It is composed of a series of greedy offloading decisions, each of which is made when the mobile user visits a WiFi AP. Furthermore, we derive that this algorithm has the competitive ratio of 2.
- We further extend our problem to a more general scenario where the transmission costs per unit data traffic via WiFi networks are different. Accordingly, we propose a heterogeneous data offloading algorithm, i.e., HDO, to solve it. We also analyze the performance of HDO.
- We conduct extensive simulations to evaluate the performances of the proposed algorithms, based on a real WiFi dataset and some synthetic datasets. The results show that our algorithms can achieve better performances, compared with other algorithms.

The remainder of the paper is organized as follows. We describe the network model, and formulate the optimization problem in Section 2. The offline and online algorithms are proposed in Sections 3 and 4, respectively. In Section 5, we introduce the extended problem and new algorithm. In Section 6, we evaluate the performances of our algorithms through extensive simulations. After reviewing related work in Section 7, we conclude the paper in Section 8.

## 2 Model and Problem Formulation

### 2.1 Offloading Model

We consider that a mobile user is conducting some mobile cloud computing applications, in which the user needs to upload some data to the cloud side. The data can be denoted by a set $\boldsymbol{D} = \{d_1, \cdots, d_i, \cdots, d_n\}$, where $d_i = \langle s_i, t_i \rangle$ $(1 \leq i \leq n)$, in which $s_i$ and $t_i$ denote the size and Time-To-Live (TTL) of the $i$-th data item, respectively. Without loss of generality, we assume that these data items are organized in the ascending order of their TTLs, that is, $t_1 \leq t_2 \leq \cdots \leq t_n$. At the same time, each data item is assumed to be indivisible. Moreover, the data item needs to be uploaded successfully before the time when its TTL expires, called the transmission *deadline* of this data item. Here, the deadline of each data item has taken the transmission time into consideration. Concretely, the deadline of a data item in our model is the latest time from which the data item can be successfully uploaded via cellular network. Its value is actually the completion time of the offloading minus the transmission time.

On the other hand, the mobile user is assumed to move around in an urban area, so that it can upload these data items to the cloud side, by using cellular networks at any time, anywhere. However, if the mobile user transmits all of these data items through cellular networks, it generally needs to pay many fees for these data transmissions. In this paper, we assume that there are many WiFi APs distributed in the urban area, and the NSP is willing to provide the WiFi-based offloading service, so as to alleviate the load of cellular networks. Hence, in order to reduce the monetary costs, the mobile user can offload some data items via WiFi networks. Since most WiFi APs cannot be accessed for free, the traffic offloading will also produce some costs, but they will be much lower than the cost via cellular networks. We use $C$ and $c$ to denote the transmission costs per unit data traffic via cellular networks and WiFi networks, respectively.

In real scenarios, not all WiFi APs can provide the offloading service. It is subject to many factors, such as when the mobile user enters the communication range of a WiFi AP, whether the WiFi AP is accessible, and so on. Moreover, since the transmission rate and the time that the user stays in the communication range of a WiFi AP are restricted, the data items that the user can transmit via this WiFi AP are generally limited. That is to say, the transmission capacity

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2017.2720741, IEEE Transactions on Parallel and Distributed Systems



is also limited. To this end, we use a triple $w = \langle \tau, p, q \rangle$ to describe the *offloading opportunity* from a WiFi AP, where $\tau$ ($> 0$) is the time of the user visiting the WiFi AP, $p$ ($\in (0, 1]$) is the probability of the WiFi AP providing the offloading service, and $q$ ($> 0$) is the transmission capacity of this WiFi AP. In this paper, we assume that NSP has recorded the historical offloading transactions, including the offloading time, transmission rate, and so on. This is reasonable since all offloading operations are conducted via NSP. Based on these historical offloading records and the mobile behavior, each mobile user can derive the offloading opportunity $w = \langle \tau, p, q \rangle$ (from NSP) for each given WiFi AP. More specifically, the probability $p$ can be estimated by the corresponding frequency of historical offloading transactions. The transmission capacity $q$ can be calculated by using the transmission rate and the time that the user stays in the communication range of each WiFi AP. Moreover, we use $\boldsymbol{W} = \{w_1, w_2, \cdots, w_m\}$ to denote all offloading opportunities, where $w_j = \langle \tau_j, p_j, q_j \rangle$ ($1 \leq j \leq m$), and $\tau_1 < \tau_2 < \cdots < \tau_m$. Here, if the user visits a WiFi AP more than one time, it can offload data items multiple times, each of which is seen as an offloading opportunity in $\boldsymbol{W}$.

In addition, since the mobile user can connect cellular network at any time, anywhere, the time that the user stays in the communication range of cellular network is long enough. As a result, the total amount of data items which can be uploaded via cellular network is large enough. Moreover, compared to the data items that the user needs to upload, the processing capacity of cloud side is generally powerful enough. Thus, we did not take the capacity constraint of cellular network and the processing capacity of cloud side into account in our data offloading model.

### 2.2 Problem Formulation

Then, we focus on the data items scheduling problem in the above offloading model, that is, how to schedule the data items in $\boldsymbol{D}$ to the offloading opportunities in $\boldsymbol{W}$, so as to minimize the total transmission cost, while ensuring that each data item is uploaded before its deadline.

Before the problem formulation, we define two terms for the simplicity of the following descriptions:

***Definition 1.*** [*Data Offloading Operation*] A data offloading operation, denoted by $(d_i, w_j)$, indicates that $d_i$ will be offloaded to the $j$-th offloading opportunity $w_j$.

***Definition 2.*** [*Data Offloading Solution*] A data offloading solution, denoted by $\boldsymbol{\Phi}$, is defined as a set of data offloading operations, i.e.,

$$\boldsymbol{\Phi} = \{(d_i, w_j) | (d_i, w_j) \in \boldsymbol{D} \times \boldsymbol{W}\}. \tag{1}$$

In light of the uncertainty of each offloading opportunity, it is important to note that we allow each data item to be scheduled to multiple offloading opportunities, as shown in Fig. 2, so as to improve the probabilities of being offloaded. If the data item still fails to be uploaded after these offloading opportunities, it will have to be transmitted by using cellular networks, to ensure it be uploaded to the cloud side before its deadline.

In our model, an offloading opportunity is not equivalent to a WiFi AP. The mobile user encountering an offloading opportunity means that it visits the related WiFi AP and

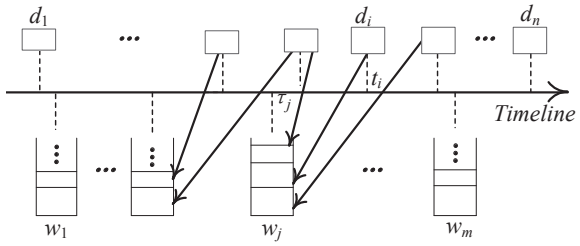



Fig. 2. Data items might be offloaded to multiple WiFi APs.

at the same time the AP can provide the offloading service. Moreover, if a data item has been offloaded via an offloading opportunity successfully, it will not be uploaded via the remaining offloading opportunities. Thus, for a given data offloading solution $\boldsymbol{\Phi}$, we can derive the successful probability of a data item $d_i$ being offloaded to WiFi networks. It is the probability of the data item $d_i$ being offloaded via any one offloading opportunity in $\boldsymbol{\Phi}$, defined as follows:

***Definition 3.*** [*Successful Offloading Probability*] For a given data offloading solution $\boldsymbol{\Phi}$, the successful offloading probability of data item $d_i$, denoted by $\rho_i(\boldsymbol{\Phi})$, satisfies:

$$\rho_i(\boldsymbol{\Phi}) = 1 - \prod_{j:(d_i, w_j) \in \boldsymbol{\Phi}} (1 - p_j). \tag{2}$$

Then, according to the probabilities, we can derive the total expected transmission cost of all data items being uploaded, defined as follows:

***Definition 4.*** [*Total Transmission Cost*] The total expected transmission cost is the sum of the expected costs of all data items in $\boldsymbol{D}$ being uploaded for a given data scheduling solution, denoted by $f_{cost}(\boldsymbol{\Phi})$, which satisfies:

$$f_{cost}(\boldsymbol{\Phi}) = \sum_{i=1}^{n} s_i \big(c\rho_i(\boldsymbol{\Phi}) + C(1 - \rho_i(\boldsymbol{\Phi}))\big). \tag{3}$$

Now, we can formalize our problem as follows:

$$\begin{aligned} Minimize: \quad & f_{cost}(\boldsymbol{\Phi}) \\ Subject\ to: \quad & \sum_{i:(d_i, w_j) \in \boldsymbol{\Phi}} s_i \leq q_j,\ 1 \leq j \leq m; \\ & t_i \geq \tau_j,\ \text{for } \forall (d_i, w_j) \in \boldsymbol{\Phi} \subseteq \boldsymbol{D} \times \boldsymbol{W}. \end{aligned} \tag{P1}$$

Here, $\sum_{i:(d_i, w_j) \in \boldsymbol{\Phi}} s_i \leq q_j$, called the *capacity constraint*, means that the total size of data items that are offloaded to the $j$-th WiFi AP should be no larger than the capacity of the WiFi AP; and, $t_i \geq \tau_j$, called the *deadline constraint*, indicates that each data item $d_i$ can be offloaded via the offloading opportunity $w_j$, only when the TTL of this data item is no less than the time of the offloading opportunity $w_j$.

By analyzing Eq. 3, we obtain

$$f_{cost}(\boldsymbol{\Phi}) = C \sum_{i=1}^{n} s_i - (C - c) \sum_{i=1}^{n} s_i \rho_i(\boldsymbol{\Phi}), \tag{4}$$

where $C \sum_{i=1}^{n} s_i$ and $(C - c)$ are known fixed values. Based on this, we define an offloading utility function as follows:

***Definition 5.*** [*Offloading Utility Function*] The offloading utility function of a data offloading solution $\boldsymbol{\Phi}$, denoted by $\mathcal{U}(\boldsymbol{\Phi})$, is the expected total size of data items that will be offloaded to WiFi networks under this data offloading solution. Then, $\mathcal{U}(\boldsymbol{\Phi})$ satisfies:

$$\mathcal{U}(\boldsymbol{\Phi}) = \sum_{i=1}^{n} s_i \rho_i(\boldsymbol{\Phi}). \tag{5}$$

TABLE 1
Description of major notations.

| Variable | Description |
|---|---|
| $n$, $m$ | the numbers of data items and offloading opportunities, respectively. |
| $\boldsymbol{D}, \boldsymbol{W}$ | the sets of data items and offloading opportunities, respectively. |
| $i$, $j$ | the indexes for data items and offloading opportunities, respectively . |
| $d_i = \langle s_i, t_i \rangle$ | the size and TTL of $i$-th data item $d_i$, respectively. |
| $w_j = \langle \tau_j, p_j, q_j \rangle$ | the time, probability and capacity of $j$-th offloading opportunity $w_j$, respectively. |
| $C$, $c$ | transmission costs per unit data traffic via cellular networks and WiFi networks, respectively. |
| $(d_i, w_j)$, $\boldsymbol{\Phi}$ | a data offloading operation (Definition 1) and a data offloading solution (Definition 2). |
| $\rho_i(\boldsymbol{\Phi})$ | the successful offloading probability of $d_i$ for a given solution $\boldsymbol{\Phi}$ (Definition 3). |
| $\Delta\rho_{ij}(\boldsymbol{\Phi})$ | the contribution of $(d_i, w_j) \in \boldsymbol{\Phi}$ to the successful offloading probability of the data item $d_i$. |
| $\varrho_{ij}(\boldsymbol{\Phi})$, $\varrho_{i0}(\boldsymbol{\Phi})$ | the expected probability of $d_i$ being transmitted via $w_j$ and cellular networks, respectively. |
| $\boldsymbol{\Omega}_{d_i}$ | the set of deadline-satisfying offloading operations for the data item $d_i$. |

Since $f_{cost}(\boldsymbol{\Phi}) = C\sum_{i=1}^{n} s_i - (C-c)\mathcal{U}(\boldsymbol{\Phi})$, Problem (P1) can be equivalently re-formalized as follows:

$$\begin{aligned} Maximize: &\quad \mathcal{U}(\boldsymbol{\Phi}) \\ Subject\ to: &\quad \sum_{i:(d_i,w_j)\in\boldsymbol{\Phi}} s_i \le q_j,\ 1 \le j \le m; \qquad (P2) \\ &\quad t_i \ge \tau_j,\ \text{for } \forall (d_i, w_j) \in \boldsymbol{\Phi} \subseteq \boldsymbol{D}\times\boldsymbol{W}. \end{aligned}$$

Unlike existing MKP [26], (P2) is an optimization problem with multiple 0-1 knapsack constraints, where each data item might be added into multiple knapsacks, and these data items in all knapsacks must share a combinatorially probabilistic optimization objective. In the subsequent section, we consider the following cases in order to solve this problem: the offline data offloading and the online data offloading. For the ease of reference, we summarize the commonly used notations throughout the paper in Table 1.

## 3 OFFLINE DATA OFFLOADING

In this section, we analyze the hardness of our problem, and then, propose an offline data offloading algorithm, followed by the performance analysis.

### 3.1 Problem Hardness Analysis

First, we prove that Problem (P2) cannot be solved in polynomial time unless $P = NP$. More specifically, we have the following theorem:

***Theorem 1.*** Problem (P2) is NP-hard.

*Proof*: To prove the NP-hardness of Problem (P2), we first consider the following special 0-1 knapsack problem:

$$\begin{aligned} Maximize: &\quad s_1x_1 + s_2x_2 + \cdots + s_nx_n \\ Subject\ to: &\quad s_1x_1 + s_2x_2 + \cdots + s_nx_n \le S, \qquad (P3) \\ &\quad x_1, x_2, \cdots, x_n \in \{0, 1\}. \end{aligned}$$

Here, $s_i$ is the size of the $i$-th item, $S$ is the size of the knapsack, and $x_i$ is a variable which indicates whether the $i$-th item is added into the knapsack. The special 0-1 knapsack problem (P3) is NP-hard [24].

Second, we consider a special case of Problem (P2), in which there is only one WiFi AP, i.e., $\boldsymbol{W} = \{\langle \tau_1, p_1, q_1 \rangle\}$, and $\tau_1 \le t_1$. Such a data offloading problem can be expressed as:

$$\begin{aligned} Maximize: &\quad \sum_{i:(d_i,w_1)\in\boldsymbol{\Phi}} s_i \qquad (P4) \\ Subject\ to: &\quad \sum_{i:(d_i,w_1)\in\boldsymbol{\Phi}} s_i \le q_1. \end{aligned}$$

Mapping $S$ in Problem (P3) to $q_1$ in Problem (P4), we can get the two problems to be equivalent. That is to say, Problem (P4), i.e., the special case of Problem (P2), is a special 0-1 knapsack problem, which is NP-hard. Thus, Problem (P2) is also NP-hard. ■

### 3.2 The Basic Solution

Since Problem (P2) has both deadline constraints and capacity constraints, we divide our solution into two phases. We take the deadline and capacity constraints into consideration in the two phases, respectively.

In the first phase, we focus on the deadline constraints of data items. That is, we first determine the priority of data offloading operations according to the TTLs of data items, and then remove the deadline constraints. More specifically, the data items with smallest TTLs will be offloaded first, since they have fewest offloading opportunities. Thus, the data items are handled (i.e., determining the corresponding offloading operations) in the ascending order of their TTLs, i.e., $d_1, d_2, \cdots, d_n$. To remove the deadline constraints, we determine a set of deadline-satisfying data offloading operations for each data item $d_i$, denoted as $\boldsymbol{\Omega}_{d_i}$:

$$\boldsymbol{\Omega}_{d_i} = \{(d_i, w_j) \mid \forall (d_i, w_j) \in \boldsymbol{D}\times\boldsymbol{W} : t_i \ge \tau_j\}. \qquad (6)$$

When we let all data offloading operations in $\boldsymbol{\Phi}$ be selected only from $\boldsymbol{\Omega}_{d_i}$ ($d_i \in \boldsymbol{D}$), the data offloading solution $\boldsymbol{\Phi}$ will be deadline-satisfying, and will not miss any feasible data offloading operations.

In the second phase, we focus on the optimization problem with the capacity constraints. Since the problem is NP-hard due to the capacity constraints, we adopt a greedy strategy to approximately solve the problem. Each iteration of the second phase consists of two main steps: (1) we select the data offloading operation in $\boldsymbol{\Omega}_{d_i}$ for each data item $d_i$, which can increase the offloading utility function value most quickly; (2) if the selected offloading operation incurs the failure of capacity constraint, we use it to replace some offloading operations in $\boldsymbol{\Phi}$ for ensuring the capacity constraint of the related offloading opportunity.

More specifically, in the first step, we find the data offloading operation, which can increase the offloading utility most quickly. This step can be formulated as follows:

$$(d_i, w_{j^*}) = \underset{(d_i,w_j)\in\boldsymbol{\Omega}_{d_i}}{\operatorname{argmax}} \mathcal{U}(\boldsymbol{\Phi}\cup\{(d_i, w_j)\}) - \mathcal{U}(\boldsymbol{\Phi}). \qquad (7)$$

In the second step, if the offloading operation $(d_i, w_{j^*})$ can satisfy the capacity constraint of $w_{j^*}$, it will be added

into the offloading solution $\mathbf{\Phi}$ directly. Otherwise, we will conduct the replacement procedure. To better describe the procedure, we denote the contribution of the offloading operation $(d_i, w_j) \in \mathbf{\Phi}$ to the successful offloading probability of the data item $d_i$ as $\Delta\rho_{ij}(\mathbf{\Phi})$, that is,

$$\Delta\rho_{ij}(\mathbf{\Phi}) = \rho_i(\mathbf{\Phi}) - \rho_i(\mathbf{\Phi} \backslash \{(d_i, w_j)\}). \quad (8)$$

In the replacement procedure, we first find a set $\mathbf{\Gamma} \subseteq \mathbf{\Phi}$, satisfying $s_i \leq q_{j^*} + \sum_{(d_x, w_{j^*}) \in \mathbf{\Gamma}} s_x$. This means that when we use $(d_i, w_{j^*})$ to replace $\mathbf{\Gamma}$, the capacity constraint of $w_{j^*}$ can be ensured. Here, each data offloading operation in $\mathbf{\Gamma}$ is selected as follows. First, we let the offloading operations in $\mathbf{\Phi}$ corresponding to $w_{j^*}$, i.e., $\{(d_x, w_{j^*}) | (d_x, w_{j^*}) \in \mathbf{\Phi}\}$, be organized in the ascending order of the $s_x \Delta\rho_{xj^*}(\mathbf{\Phi})$ value, where $s_x \Delta\rho_{xj^*}(\mathbf{\Phi})$ is the incremental offloading utility of $(d_x, w_{j^*})$. According to this order, we add the corresponding offloading operations into $\mathbf{\Gamma}$ one by one, until $s_i \leq q_{j^*} + \sum_{(d_x, w_{j^*}) \in \mathbf{\Gamma}} s_x$ is satisfied. After determining the set $\mathbf{\Gamma}$, we compare the incremental offloading utility values of $(d_i, w_{j^*})$ and $\mathbf{\Gamma}$, i.e., $s_i \Delta\rho_{ij^*}(\mathbf{\Phi} \cup \{(d_i, w_{j^*})\})$ and $\sum_{(d_x, w_{j^*}) \in \mathbf{\Gamma}} s_x \Delta\rho_{xj^*}(\mathbf{\Phi})$. If the former is larger than the later, we will use $(d_i, w_{j^*})$ to replace $\mathbf{\Gamma}$. Otherwise, we will not conduct the replacement.

### 3.3 The Detailed Algorithm

Based on the above strategy, we design the greedy algorithm to approximately solve the optimization problem (P2), as shown in Algorithm 1. In Step 1, the data offloading solution $\mathbf{\Phi}$ and the sets of deadline-satisfying offloading operations for each data item $d_i \in \boldsymbol{D}$ (i.e., $\mathbf{\Omega}_{d_i}$) are initialized to be empty. In Steps 2-5, we add all deadline-satisfying data offloading operations corresponding to $d_i$ into the set $\mathbf{\Omega}_{d_i}$ ($d_i \in \boldsymbol{D}$). Then, the data offloading operation in $\mathbf{\Omega}_{d_i}$, which can increase the offloading utility function most quickly (e.g., $(d_i, w_{j^*})$), will be considered first, as shown in Steps 6-7. If the capacity constraint of $w_{j^*}$ is satisfied, $(d_i, w_{j^*})$ will be added into $\mathbf{\Phi}$ directly, and at the same time the remaining transmission capacity of $w_{j^*}$ is updated in Steps 8-10.

Otherwise, we first find a set $\mathbf{\Gamma} = \{(d_x, w_{j^*}) | (d_x, w_{j^*}) \in \mathbf{\Phi}\}$, in which each offloading operation is selected in the ascending order of $s_x \Delta\rho_{xj^*}(\mathbf{\Phi})$, to ensure the capacity constraint of $w_{j^*}$ while replacing $\mathbf{\Gamma}$ by $(d_i, w_{j^*})$, i.e., $s_i \leq q_{j^*} + \sum_{(d_x, w_{j^*}) \in \mathbf{\Gamma}} s_x$, in Steps 11-12. Then, we compute the incremental offloading utility values of $(d_i, w_{j^*})$ and $\mathbf{\Gamma}$, i.e., $s_i \Delta\rho_{ij^*}(\mathbf{\Phi} \cup \{(d_i, w_{j^*})\})$ and $\sum_{(d_x, w_{j^*}) \in \mathbf{\Gamma}} s_x \Delta\rho_{xj^*}(\mathbf{\Phi})$, respectively. If the former is larger than the later, we will replace $\mathbf{\Gamma}$ by $(d_i, w_{j^*})$, and update the remaining capacity of $w_{j^*}$ in Steps 13-15. Else, we will not conduct the replacement. Then, we update the set $\mathbf{\Omega}_{d_i}$ by deleting $(d_i, w_{j^*})$ from it, in Step 16. After conducting the offloading procedure for the last data item $d_n$, the algorithm terminates and outputs the data offloading solution $\mathbf{\Phi}$, in Step 17.

By analyzing Algorithm 1, we show that the algorithmic procedures are polynomial-time, and the computational overhead of Algorithm 1 is $O(m^2 n^2)$. Moreover, we can straightforwardly demonstrate correctness of the algorithm in the following theorem:

***Theorem 2.*** Algorithm 1 is correct. It will terminate for sure, and will produce a feasible data offloading solution.

**Algorithm 1** The FDO Algorithm

**Require:** $\boldsymbol{D}, \boldsymbol{W}$.
**Ensure:** $\mathbf{\Phi}$.
1: Initialize $\mathbf{\Phi} = \phi$ and $\mathbf{\Omega}_{d_i} = \phi$ ($d_i \in \boldsymbol{D}$);
2: **for** $d_i$ from $d_1$ to $d_n$ **do**
3: **for** $w_j$ from $w_1$ to $w_m$ **do**
4: **if** $\tau_j \leq t_i$ **then**
5: $\mathbf{\Omega}_{d_i} = \mathbf{\Omega}_{d_i} \cup \{(d_i, w_j)\}$;
6: **while** $(\exists\ (d_i, w_j) \in \mathbf{\Omega}_{d_i})$ **do**
7: $(d_i, w_{j^*}) = \underset{(d_i, w_j) \in \mathbf{\Omega}_{d_i}}{\operatorname{argmax}} \mathcal{U}(\mathbf{\Phi} \cup \{(d_i, w_j)\}) - \mathcal{U}(\mathbf{\Phi})$;
8: **if** $s_i \leq q_{j^*}$ **then**
9: $\mathbf{\Phi} = \mathbf{\Phi} \cup \{(d_i, w_{j^*})\}$;
10: $q_{j^*} = q_{j^*} - s_i$, where $s_i$ is the size of $d_i$;
11: **else**
12: Find a set $\mathbf{\Gamma} \subseteq \mathbf{\Phi}$, s.t., $s_i \leq q_{j^*} + \sum_{(d_x, w_{j^*}) \in \mathbf{\Gamma}} s_x$;
13: **if** $s_i \Delta\rho_{ij^*}(\mathbf{\Phi} \cup \{(d_i, w_{j^*})\}) > \sum_{(d_x, w_{j^*}) \in \mathbf{\Gamma}} s_x \Delta\rho_{xj^*}(\mathbf{\Phi})$ **then**
14: $\mathbf{\Phi} = \mathbf{\Phi} \cup \{(d_i, w_{j^*})\} \backslash \mathbf{\Gamma}$;
15: $q_{j^*} = q_{j^*} - s_i + \sum_{(d_x, w_{j^*}) \in \mathbf{\Gamma}} s_x$;
16: $\mathbf{\Omega}_{d_i} = \mathbf{\Omega}_{d_i} \backslash \{(d_i, w_{j^*})\}$;
17: **return** $\mathbf{\Phi}$;

*Proof*: Since each data offloading operation is selected from the sets of deadline-satisfying data offloading operations, some limited sets, the algorithm will terminate for sure, and the results will satisfy the deadline constraints. On the other hand, at each round of selection in Algorithm 1, the capacity constraints are ensured. Thus, the produced data offloading solution must be feasible. ■

### 3.4 Examples

To better understand Algorithm 1, we present an example to show the data offloading procedure, in which the mobile user has four data items $\boldsymbol{D} = \{d_i = \langle s_i, t_i \rangle | 1 \leq i \leq 4\}$, where $s_1 = 8, t_1 = 11, s_2 = 6, t_2 = 13, s_3 = 5, t_3 = 17, s_4 = 10, t_4 = 18$, and it wishes to offload the data items to two offloading opportunities $w_1 = \langle \tau_1, p_1, q_1 \rangle$ and $w_2 = \langle \tau_2, p_2, q_2 \rangle$, where $\tau_1 = 10, p_1 = 0.6, q_1 = 15, \tau_2 = 15, p_2 = 0.9, q_2 = 10$. Since the deadline constraints $\tau_1 < t_1 < t_2 < \tau_2 < t_3 < t_4$ are satisfied, $\mathbf{\Omega}_{d_i}$ ($d_i \in \boldsymbol{D}$) is first determined in Fig. 3(a). Then, Algorithm 1 greedily selects data offloading operations as follows:

In the first round, $\mathbf{\Phi} = \phi$. For the first data item $d_1$, we have $\mathbf{\Omega}_{d_1} = \{(d_1, w_1)\}$. Since $\mathcal{U}(\mathbf{\Phi} \cup \{(d_1, w_1)\}) - \mathcal{U}(\mathbf{\Phi}) = 4.8$ and $s_1 \leq q_1$, we add $(d_1, w_1)$ into $\mathbf{\Phi}$ and delete $(d_1, w_1)$ from $\mathbf{\Omega}_{d_1}$, as shown in Fig. 3(a). Moreover, we update $q_1 = 7$. Now, since $\mathbf{\Omega}_{d_1}$ is empty, we consider the next data item $d_2$.

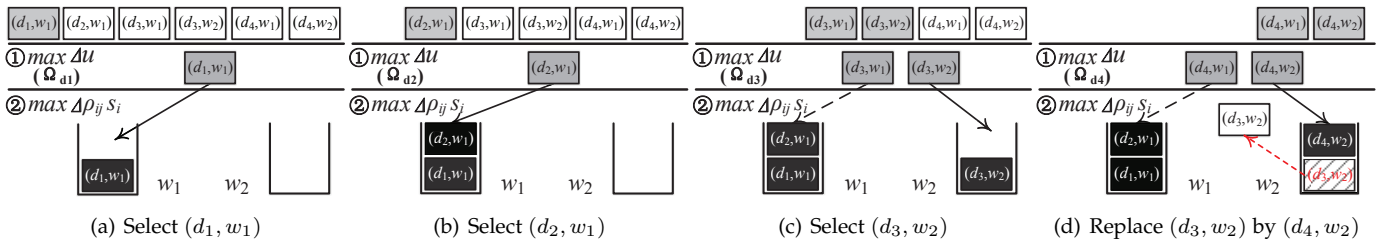
In the second round, $\mathbf{\Phi} = \{(d_1, w_1)\}$. We consider the data item $d_2$ and get the corresponding set of deadline-satisfying offloading operations $\mathbf{\Omega}_{d_2} = \{(d_2, w_1)\}$. Similarly, since $\mathcal{U}(\mathbf{\Phi} \cup \{(d_2, w_1)\}) - \mathcal{U}(\mathbf{\Phi}) = 3.6$ and $s_2 \leq q_1 = 7$, we add $(d_2, w_1)$ into $\mathbf{\Phi}$ and delete $(d_2, w_1)$ from $\mathbf{\Omega}_{d_2}$, as shown in Fig. 3(b). We also update $q_1 = 1$.

In the third round, we focus on the data item $d_3$. Now, we have $\mathbf{\Phi} = \{(d_1, w_1), (d_2, w_1)\}$ and $\mathbf{\Omega}_{d_3} = \{(d_3, w_1), (d_3, w_2)\}$. Algorithm 1 computes the increased offloading utility function values for each offloading operation $(d_3, w_j) \in \mathbf{\Omega}_{d_3}$. The results are listed as follows:

$$\mathcal{U}(\mathbf{\Phi} \cup \{(d_3, w_1)\}) - \mathcal{U}(\mathbf{\Phi}) = 3.0, \mathcal{U}(\mathbf{\Phi} \cup \{(d_3, w_2)\}) - \mathcal{U}(\mathbf{\Phi}) = 4.5.$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2017.2720741, IEEE Transactions on Parallel and Distributed Systems






(a) Select $(d_1, w_1)$ (b) Select $(d_2, w_1)$ (c) Select $(d_3, w_2)$ (d) Replace $(d_3, w_2)$ by $(d_4, w_2)$

Fig. 3. Example: greedily schedule the data items $d_1, d_2, d_3, d_4$ to offloading opportunities $w_1, w_2$, and the obtained data offloading solution is $\mathbf{\Phi} = \{(d_1, w_1), (d_2, w_1), (d_4, w_2)\}$. The selected offloading operations are according to the greedy criterion max $\Delta\mathcal{U} = \mathcal{U}(\mathbf{\Phi} \cup \{(d_i, w_j)\}) - \mathcal{U}(\mathbf{\Phi})$ and max $\Delta\rho_{ij}(\mathbf{\Phi})s_i$, respectively.

According to the results, we select $(d_3, w_2)$. Due to $s_3 \leq q_2$, we add $(d_3, w_2)$ into the set $\mathbf{\Phi}$ and delete it from $\mathbf{\Omega}_{d_3}$. We update $q_2 = 5$. Next, we compute $\mathcal{U}(\mathbf{\Phi} \cup \{(d_3, w_1)\}) - \mathcal{U}(\mathbf{\Phi}) = 0.3$. At the moment, we have $s_3 = 5 > q_1 = 1$. Therefore, according to the ascending order of $s_i\Delta\rho_{ij}(\mathbf{\Phi})$, we have $\mathbf{\Gamma} = \{(d_2, w_1)\}$ so that $q_1 + s_2 = 7 > s_3$. Then, we get $s_2\Delta\rho_{21}(\mathbf{\Phi}) = 3.6 > s_3\Delta\rho_{31}(\mathbf{\Phi} \cup \{(d_3, w_1)\}) = 0.3$. This means that we will not conduct the replacement, as shown in Fig. 3(c). After deleting $(d_3, w_1)$ from $\mathbf{\Omega}_{d_3}$, we get $\mathbf{\Omega}_{d_3} = \phi$. Thus, we will consider the next data item.

In the fourth round, we have $\mathbf{\Omega}_{d_4} = \{(d_4, w_1), (d_4, w_2)\}$ and $\mathbf{\Phi} = \{(d_1, w_1), (d_2, w_1), (d_3, w_2)\}$. Similar to the computation in the third round, we have the following results:

$$\mathcal{U}(\mathbf{\Phi} \cup \{(d_4, w_1)\}) - \mathcal{U}(\mathbf{\Phi}) = 6, \mathcal{U}(\mathbf{\Phi} \cup \{(d_4, w_2)\}) - \mathcal{U}(\mathbf{\Phi}) = 9.$$

We first consider the offloading operation $(d_4, w_2)$. Since $s_4 = 10 > q_2 = 5$, we have $\mathbf{\Gamma} = \{(d_3, w_2)\}$ according to the $s_i\Delta\rho_{ij}(\mathbf{\Phi})$ value. Due to $s_3\Delta\rho_{32}(\mathbf{\Phi}) = 4.5 < s_4\Delta\rho_{42}(\mathbf{\Phi} \cup \{(d_4, w_2)\}) = 9$, we replace $(d_3, w_2)$ by $(d_4, w_2)$ in $\mathbf{\Phi}$, and get $\mathbf{\Phi} = \{(d_1, w_1), (d_2, w_1), (d_4, w_2)\}$. After deleting $(d_4, w_2)$ from $\mathbf{\Omega}_{d_4}$, we have $\mathbf{\Omega}_{d_4} = \{(d_4, w_1)\}$ and $\mathcal{U}(\mathbf{\Phi} \cup \{(d_4, w_1)\}) - \mathcal{U}(\mathbf{\Phi}) = 0.6$. Similarly, we obtain $\mathbf{\Gamma} = \{(d_1, w_1), (d_2, w_1)\}$, and further get $s_1\Delta\rho_{11}(\mathbf{\Phi}) + s_2\Delta\rho_{21}(\mathbf{\Phi}) = 8.4 > s_4\Delta\rho_{41}(\mathbf{\Phi} \cup \{(d_4, w_1)\}) = 0.6$. This means that we will not conduct the replacement. Now, since $\mathbf{\Omega}_{d_4}$ is empty, Algorithm 1 terminates and outputs the final offloading solution $\mathbf{\Phi} = \{(d_1, w_1), (d_2, w_1), (d_4, w_2)\}$, as shown in Fig. 3(d).

### 3.5 Performance Analysis

In this subsection, we analyze the approximation ratio of Algorithm 1. First, we use $\boldsymbol{opt}_F$ to denote the optimal offline offloading solution of optimization problem (P2). Then, we have the following theorem:

***Theorem 3.*** FDO has an approximation ratio of 2. That is,

$$\frac{\mathcal{U}(\boldsymbol{opt}_F)}{\mathcal{U}(\mathbf{\Phi})} < 2. \tag{9}$$

*Proof*: First, we consider a special solution. For this solution, we assume that all data items can be divided, and let each data item $d_i = \langle s_i, t_i \rangle$ be divided as $d_{i_1} = \langle 1, t_i \rangle$, $\cdots$, $d_{i_{s_i}} = \langle 1, t_i \rangle$. Then, we conduct our Algorithm 1 to get a solution, denoted by $\boldsymbol{opt}_F^*$. When all data items are divisible, the greedy strategy in Algorithm 1 can achieve the optimal result. This is because the problem has the property of optimal substructure, the best offloading operation is selected in each round, and the transmission capacity of each offloading opportunity is fully utilized. Since $\boldsymbol{opt}_F$ is the optimal solution where data items are indivisible, it cannot fully utilize the transmission capacity of each offloading opportunity in most cases. Hence, we have:

$$\mathcal{U}(\boldsymbol{opt}_F^*) \geq \mathcal{U}(\boldsymbol{opt}_F). \tag{10}$$

Second, we consider another special solution for the case where data items are indivisible, but the capacity constraint of each offloading opportunity can be broken once. Denote this solution as $\boldsymbol{opt}_F^+$. Since $\boldsymbol{opt}_F^+$ and $\boldsymbol{opt}_F^*$ are produced by using the same greedy criterion, while $\boldsymbol{opt}_F^+$ can offload data items beyond each capacity constraint once, we have:

$$\mathcal{U}(\boldsymbol{opt}_F^+) \geq \mathcal{U}(\boldsymbol{opt}_F^*). \tag{11}$$

Now, we compare $\boldsymbol{opt}_F^+$ and $\mathbf{\Phi}$. Without loss of generality, we assume that there are $g$ data offloading operations corresponding to $w_j$, which have been selected into $\mathbf{\Phi}$, denoted as $\{(d_{i_1}, w_j), \cdots, (d_{i_x}, w_j), \cdots, (d_{i_g}, w_j)\}$, in which $d_{i_x} = \langle s_{i_x}, t_{i_x} \rangle$. Now, we consider that the current offloading operation $(d_i, w_j)$, and assume that $s_i > q_j$ where $s_i$ and $q_j$ denote the data size of $d_i$ and the remaining capacity of $w_j$, respectively. According to the replacement strategy in Algorithm 1, we find a set $\mathbf{\Gamma} \subseteq \mathbf{\Phi}$ so that $s_i \leq q_j + \sum_{(d_{i_x}, w_j) \in \mathbf{\Gamma}} s_{i_x}$. In $\boldsymbol{opt}_F^+$, $(d_i, w_j)$ will be added directly since each offloading opportunity can be broken once. In contrast, we replace $\mathbf{\Gamma}$ by $(d_i, w_j)$ if $s_i\Delta\rho_{ij}(\mathbf{\Phi} \cup \{(d_i, w_j)\}) > \sum_{(d_{i_x}, w_j) \in \mathbf{\Gamma}} s_{i_x}\Delta\rho_{i_x j}(\mathbf{\Phi})$. For convenience, we use $\Delta\mathcal{U}_j(\mathbf{\Phi})$ to denote the incremental offloading utility corresponding to $w_j$ based on $\mathbf{\Phi}$. Therefore, we have

$$\Delta\mathcal{U}_j(\mathbf{\Phi}) = \sum_{(d_{i_x}, w_j) \in (\mathbf{\Phi} \setminus \mathbf{\Gamma})} s_{i_x}\Delta\rho_{i_x j}(\mathbf{\Phi}) + \max\{\sum_{(d_{i_x}, w_j) \in \mathbf{\Gamma}} s_{i_x}\Delta\rho_{i_x j}(\mathbf{\Phi}), s_i\Delta\rho_{ij}(\mathbf{\Phi} \cup \{(d_i, w_j)\})\}; \tag{12}$$

$$\Delta\mathcal{U}_j(\boldsymbol{opt}_F^+) = \sum_{(d_{i_x}, w_j) \in \mathbf{\Phi}} s_{i_x}\Delta\rho_{i_x j}(\mathbf{\Phi}) + s_i\Delta\rho_{ij}(\mathbf{\Phi} \cup \{(d_i, w_j)\}). \tag{13}$$

Then, for $\forall j \in [1, m]$, we have

$$2\Delta\mathcal{U}_j(\mathbf{\Phi}) \geq \Delta\mathcal{U}_j(\boldsymbol{opt}_F^+) + \sum_{(d_{i_x}, w_j) \in (\mathbf{\Phi} \setminus \mathbf{\Gamma})} s_{i_x}\Delta\rho_{i_x j}(\mathbf{\Phi}). \tag{14}$$

Furthermore, based on Eqs. 5, 8 and 14, we get

$$2\mathcal{U}(\mathbf{\Phi}) > \mathcal{U}(\boldsymbol{opt}_F^+) \geq \mathcal{U}(\boldsymbol{opt}_F). \tag{15}$$

Thus, the theorem is correct. ■

## 4 ONLINE DATA OFFLOADING

In this section, we propose the online data offloading algorithm, in which the data offloading decision is made only when the user encounters the offloading opportunities.

### 4.1 The Basic Idea

The basic idea is that the mobile user makes the online data offloading decisions only when it *encounters* an offloading opportunity. Here, the "encounter" means that the user enters the communication range of the related WiFi AP and at the same time this AP can provide offloading service. When the user encounters the offloading opportunity $w_j$, the estimated probability $p_j$ for this encounter is replaced by 1. For convenience, we directly let $p_j = 1$. Otherwise, if the WiFi AP cannot provide offloading service when the user visits it, we say that the user does not encounter the offloading opportunity and let $p_j$ be replaced by 0.

Different from the offline case, we just focus on the data offloading based on the encountered offloading opportunity $w_j$ in the online case. That is, we will offload some data items via $w_j$ in real time, while ignoring other offloading opportunities. Moreover, once the offloading operation $(d_i, w_j)$ is determined, the data item $d_i$ is offloaded via $w_j$ for sure, and it will not be considered for the later offloading opportunities. This means that each data item is scheduled only once in the online case. By extending the offline strategy, we divide the online solution into two phases: (1) we first determine the priority of data offloading operations and remove the deadline constraints; (2) we select the data offloading operations, which can increase the offloading utility most quickly and at the same time satisfy the capacity constraints.

In the first phase, we determine the priority of offloading operations and remove the deadline constraints. Similar to the offline case, the data items with smallest TTLs will be offloaded first in the online case, since they have fewest offloading opportunities. Also, we use $\mathbf{\Omega}_{d_i}$ to denote the set of deadline-satisfying offloading operations for the data item $d_i$ ($\in \boldsymbol{D}$). Here, since we just focus on the encountered offloading opportunity $w_j$, $\mathbf{\Omega}_{d_i}$ only contains one offloading operation, i.e., $\mathbf{\Omega}_{d_i} = \{(d_i, w_j)\}$, for the data item $d_i$.

In the second phase, we consider the capacity constraint of the encountered offloading opportunity. For convenience, we use $\mathbf{\Phi}_j$ to denote the offloading solution corresponding to the encountered offloading opportunity $w_j$. For the data item $d_i$, if the offloading operation $(d_i, w_j)$ in $\mathbf{\Omega}_{d_i}$ satisfies the capacity constraint of $w_j$, it will be added into the offloading solution $\mathbf{\Phi}_j$ directly. Otherwise, we will conduct the replacement procedure. Concretely, we first find a set $\mathbf{\Gamma} = \{(d_x, w_j) | (d_x, w_j) \in \mathbf{\Phi}_j\}$, which satisfies the capacity constraint of $w_j$ when replacing $\mathbf{\Gamma}$ by $(d_i, w_j)$, i.e., $s_i \leq q_j + \sum_{(d_x,w_j)\in\mathbf{\Gamma}} s_x$. Due to $p_j = 1$, the incremental offloading utility of a data offloading operation $(d_x, w_j) \in \mathbf{\Phi}_j$ is actually the data size, i.e., $s_x$. Based on this, we add the offloading operations into $\mathbf{\Gamma}$ in the ascending order of data sizes. By comparing the incremental offloading utility values of $(d_i, w_j)$ and $\mathbf{\Gamma}$, i.e., $s_i$ and $\sum_{(d_x,w_j)\in\mathbf{\Gamma}} s_x$, we replace $\mathbf{\Gamma}$ by $(d_i, w_j)$ if $s_i > \sum_{(d_x,w_j)\in\mathbf{\Gamma}} s_x$.

### 4.2 The Detailed Algorithm

The detailed algorithm is presented in Algorithm 2. First, the offloading solution $\mathbf{\Phi}^*$ and $\mathbf{\Phi}_j$ ($1 \leq j \leq m$) are initialized to be empty in Step 1. Then, for each offloading opportunity in $\boldsymbol{W}$, if the mobile user encounters the $j$-th offloading opportunity $w_j$, the corresponding probability $p_j$ is replaced by 1 and Algorithm 2 makes the online offloading decisions in Steps 3-15, otherwise the algorithm skips $w_j$ and continues in Steps 16-17.

**Algorithm 2** The NDO Algorithm

**Require:** $\boldsymbol{D}$, $\boldsymbol{W}$.
**Ensure:** $\mathbf{\Phi}^*$.
1: $\mathbf{\Phi}^* = \mathbf{\Phi}_j = \phi$ $(1 \leq j \leq m)$;
2: **for** $w_j$ in $\boldsymbol{W}$ **do**
3: **if** the user encounters $w_j$ **then**
4: $\boldsymbol{D} = \{d_i | d_i \in \boldsymbol{D}, t_i \geq \tau_j\} \setminus \{d_i | (d_i, w_j) \in \cup_{j=1}^{m} \mathbf{\Phi}_j\}$;
5: **for** $d_i$ in $\boldsymbol{D}$ **do**
6: $\mathbf{\Omega}_{d_i} = \{(d_i, w_j)\}$;
7: **while** $(\exists\ (d_i, w_j) \in \mathbf{\Omega}_{d_i})$ **do**
8: **if** $s_i \leq q_j$ **then**
9: $\mathbf{\Phi}_j = \mathbf{\Phi}_j \cup \{(d_i, w_j)\}$, $q_j = q_j - s_i$;
10: **else**
11: Find a set $\mathbf{\Gamma} \subseteq \mathbf{\Phi}_j$, s.t., $s_i \leq q_j + \sum_{(d_x,w_j)\in\mathbf{\Gamma}} s_x$;
12: **if** $s_i > \sum_{(d_x,w_j)\in\mathbf{\Gamma}} s_x$ **then**
13: $\mathbf{\Phi}_j = \mathbf{\Phi}_j \cup \{(d_i, w_j)\} \setminus \mathbf{\Gamma}$;
14: $q_j = q_j - s_i + \sum_{(d_x,w_j)\in\mathbf{\Gamma}} s_x$;
15: $\mathbf{\Omega}_{d_i} = \mathbf{\Omega}_{d_i} \setminus \{(d_i, w_j)\}$;
16: **else**
17: Continue; //the user does not meet $w_j$, i.e., $p_j = 0$;
18: $\mathbf{\Phi}^* = \cup_{j=1}^{m} \mathbf{\Phi}_j$;
19: **return** $\mathbf{\Phi}^*$;

More specifically, when the user encounters the offloading opportunity $w_j$, the set of data items, which have not been offloaded and the corresponding TTLs have not expired, is determined in Step 4. Then, the offloading operation corresponding to $w_j$ for each data item $d_i$ is determined in Steps 5-6. If the offloading operation $(d_i, w_j)$ in $\mathbf{\Omega}_{d_i}$ satisfies the capacity constraint of $w_j$, it will be added into $\mathbf{\Phi}_j$ directly, and at the same time the remaining capacity of $w_j$ is updated, in Steps 8-9. Otherwise, we determine a set $\mathbf{\Gamma} \subseteq \mathbf{\Phi}_j$ satisfying $s_i \leq q_j + \sum_{(d_x,w_j)\in\mathbf{\Gamma}} s_x$, in Step 11. If the incremental offloading utility of $(d_i, w_j)$ is larger than that of $\mathbf{\Gamma}$, the offloading operations in $\mathbf{\Gamma}$ are replaced with $(d_i, w_j)$, and the remaining transmission capacity of $w_j$ is updated in Steps 12-14. Then, $(d_i, w_j)$ will be deleted from $\mathbf{\Omega}_{d_i}$ in Step 15. When the user does not encounter $w_j$, Algorithm 2 will skip $w_j$ and continue, in Steps 16-17. At last, by combining the offloading solution for each encountered offloading opportunity (i.e., $\mathbf{\Phi}_j$), Algorithm 2 terminates and outputs the final offloading solution $\mathbf{\Phi}^*$ in Steps 18-19.

In addition, the computational overhead of Algorithm 2 is $O(mn)$.

### 4.3 Performance Analysis

We use competitive ratio to evaluate the online approximation performance of NDO. Assume that there is a god, who can foresee whether the mobile user will encounter each offloading opportunity. Based on this knowledge, the god can give an optimal online offloading solution, denoted by $\boldsymbol{opt}_N$. The competitive ratio is defined as the ratio of $\boldsymbol{opt}_N$ and our online solution $\mathbf{\Phi}^*$, i.e., $\frac{\mathcal{U}(\boldsymbol{opt}_N)}{\mathcal{U}(\mathbf{\Phi}^*)}$. This metric is different from the approximation ratio adopted in the offline case. Note that the approximation ratio is the ratio of the optimal offline solution $\boldsymbol{opt}_F$ and our offline solution $\mathbf{\Phi}$, i.e., $\frac{\mathcal{U}(\boldsymbol{opt}_F)}{\mathcal{U}(\mathbf{\Phi})}$. Since $\boldsymbol{opt}_F$ is not optimal in the online case,

$\boldsymbol{opt}_N$ is better than $\boldsymbol{opt}_F$. As a result, the competitive ratio is more accurate than the approximation ratio. Then, we have:

***Theorem 4.*** The competitive ratio of NDO satisfies

$$\frac{\mathcal{U}(\boldsymbol{opt}_N)}{\mathcal{U}(\boldsymbol{\Phi}^*)} < 2. \tag{16}$$

*Proof*: We use mathematical induction to prove the correctness. Like the offline case, we consider a special solution, where each item can be divisible. Then, the algorithm can produce an online offloading solution for this case, denoted by $\boldsymbol{opt}^*_N$. Also, we consider another special solution in which each offloading opportunity can be broken once, and use $\boldsymbol{opt}^+_N$ to denote the online solution. Similar to the analysis in the offline case, we straightforwardly have

$$\mathcal{U}(\boldsymbol{opt}^+_N) \geq \mathcal{U}(\boldsymbol{opt}^*_N) \geq \mathcal{U}(\boldsymbol{opt}_N). \tag{17}$$

Then, we focus on $\boldsymbol{opt}^+_N$ and $\boldsymbol{\Phi}^*$ hereinafter.

(1) Without loss of generality, we assume that the first encountered offloading opportunity is $w_{j_1}$, and the total $g$ data offloading operations have been selected for $w_{j_1}$, i.e., $\boldsymbol{\Phi}_{j_1} = \{(d_{i_x}, w_{j_1}) | 1 \leq x \leq g\}$, in which $d_{i_x} = \langle s_{i_x}, t_{i_x} \rangle$. When considering the next offloading operation $(d_i, w_{j_1})$, we find that $s_i > q_{j_1}$ where $q_{j_1}$ is the remaining transmission capacity of $w_{j_1}$. Then, we determine a set $\boldsymbol{\Gamma} \subseteq \boldsymbol{\Phi}_{j_1}$ so that $s_i \leq q_{j_1} + \sum_{(d_{i_x}, w_{j_1}) \in \boldsymbol{\Gamma}} s_{i_x}$. Based on this, we have $\mathcal{U}(\boldsymbol{\Phi}^*) = \mathcal{U}(\boldsymbol{\Phi}_{j_1}) = \sum_{(d_{i_x}, w_{j_1}) \in (\boldsymbol{\Phi}_{j_1} \setminus \boldsymbol{\Gamma})} s_{i_x} + \max\{s_i, \sum_{(d_{i_x}, w_{j_1}) \in \boldsymbol{\Gamma}} s_{i_x}\}$ and $\mathcal{U}(\boldsymbol{opt}^+_N) = \mathcal{U}(\boldsymbol{opt}^+_{N_{j_1}}) = \sum_{(d_{i_x}, w_{j_1}) \in \boldsymbol{\Phi}_{j_1}} s_{i_x} + s_i$, where $\mathcal{U}(\boldsymbol{opt}^+_{N_j})$ denote the offloading utility corresponding to $w_j$ based on the online solution $\boldsymbol{opt}^+_N$. Thus, for $w_{j_1}$, we get

$$2\mathcal{U}(\boldsymbol{\Phi}^*) \geq \mathcal{U}(\boldsymbol{opt}^+_N). \tag{18}$$

(2) Then, we consider that Eq. 18 holds for the $h$-th encountered offloading opportunity $w_{j_h}$, i.e., $2\mathcal{U}(\cup_{k=1}^{h} \boldsymbol{\Phi}_{j_k}) \geq \mathcal{U}(\cup_{k=1}^{h} \boldsymbol{opt}^+_{N_{j_k}})$, and now we take $w_{j_{h+1}}$ into account. When the user encounters $w_{j_{h+1}}$, we divide the situation into two cases. In the first case, we consider that $\sum_{d_i \in \boldsymbol{D}} s_i \geq \frac{1}{2} q_{j_{h+1}}$. Hence, similar to the analysis in (1), we can directly get

$$2\mathcal{U}(\cup_{k=1}^{h+1} \boldsymbol{\Phi}_{j_k}) \geq \mathcal{U}(\cup_{k=1}^{h+1} \boldsymbol{opt}^+_{N_{j_k}}). \tag{19}$$

In the second case, we consider $\sum_{d_i \in \boldsymbol{D}} s_i < \frac{1}{2} q_{j_{h+1}}$. This may be caused by the abandonment of offloading operations for the encountered offloading opportunities. Without loss of generality, we assume that only some data offloading operations, whose corresponding TTLs of the data items are between $\tau_{j_h}$ and $\tau_{j_{h+1}}$, are abandoned. This is because the offloading operation (e.g., $(d_{i^*}, w_{j_h})$) with larger data size where $t_{i^*} \geq \tau_{j_{h+1}}$ is added into $\boldsymbol{\Phi}_{j_h}$. For convenience, we use $\boldsymbol{D}_j$ to denote the set of data items whose TTLs are between $\tau_j$ and $\tau_{j+1}$. Then, in the worst case where $\boldsymbol{D}_{j_{h+1}} = \{d_{i^*}\}$ and $s_{i^*} < \frac{1}{2} q_{j_{h+1}}$, we have

$$\mathcal{U}(\boldsymbol{\Phi}_{j_h}) + \mathcal{U}(\boldsymbol{\Phi}_{j_{h+1}}) = s_{i^*}; \tag{20}$$

$$\mathcal{U}(\boldsymbol{opt}^+_{N_{j_h}}) + \mathcal{U}(\boldsymbol{opt}^+_{N_{j_{h+1}}}) = s_{i^*} + \sum\nolimits_{d_i \in \boldsymbol{D}_{j_h}} s_i. \tag{21}$$

According to $\sum_{d_i \in \boldsymbol{D}_{j_h}} s_i < \frac{1}{2} q_{j_h}$ and $s_{i^*} \geq \frac{1}{2} q_{j_h}$ in the second case, we can also get

$$2\mathcal{U}(\cup_{k=1}^{h+1} \boldsymbol{\Phi}_{j_k}) \geq \mathcal{U}(\cup_{k=1}^{h+1} \boldsymbol{opt}^+_{N_{j_k}}). \tag{22}$$

Based on the nature of mathematical induction, we have

$$2\mathcal{U}(\boldsymbol{\Phi}^*) \geq \mathcal{U}(\boldsymbol{opt}^+_N) \geq \mathcal{U}(\boldsymbol{opt}_N). \tag{23}$$

As a result, the theorem holds. ■

## 5 Extension

In this section, we extend our problem to a more practical scenario, where the transmission costs per unit data traffic via WiFi networks are heterogeneous. We first introduce the extended problem, and then propose a Heterogeneous Data Offloading algorithm, called HDO, to solve this problem, followed by the performance analysis.

### 5.1 The Extended Problem

In our initial problem, we consider that all WiFi AP owners have come to an agreement with NSP, and made the transmission costs per unit data traffic via all WiFi networks be uniform. In a more general case, the WiFi APs are distributed in different locations in the city. Different locations mean different difficulty degrees of accessing WiFi networks, resulting in different transmission costs per unit data traffic. Hence, the initial triple $w = \langle \tau, p, q \rangle$ to describe the *offloading opportunity* from a WiFi AP, is replaced by $w = \langle \tau, p, q, c \rangle$, where $c$ denotes the transmission cost per unit data traffic via this WiFi AP. Moreover, we consider that $\forall c_j (1 \leq j \leq m)$ is much smaller than the cost via cellular networks $C$. We also use $\boldsymbol{\Phi}$ to denote the data offloading solution in the extended problem. According to the defined *offloading utility function* $\mathcal{U}(\boldsymbol{\Phi}) = \sum_{i=1}^{n} s_i \rho_i(\boldsymbol{\Phi})$ in the original problem, minimizing the total transmission cost $f_{cost}(\boldsymbol{\Phi})$ is equivalent to maximizing the utility function $\mathcal{U}(\boldsymbol{\Phi})$. However, in the extended case, the transmission costs per unit data traffic via WiFi networks are different. That is to say, here, minimizing the total transmission cost $f_{cost}(\boldsymbol{\Phi})$ is not equivalent to maximizing the utility function $\mathcal{U}(\boldsymbol{\Phi})$. Therefore, the greedy strategy used in the FDO and NDO algorithms is not applicable. To solve the extended problem, we propose another concept of *offloading cost function*, based on which we design another greedy algorithm, called Heterogeneous Data Offloading (HDO) algorithm.

For the simplicity of following descriptions, we use $\varrho_{ij}(\boldsymbol{\Phi})$ to denote the expected probability that $d_i$ is scheduled via WiFi AP $w_j$ with a given data scheduling solution $\boldsymbol{\Phi}$. $\varrho_{ij}(\boldsymbol{\Phi})$ is calculated in the following form:

$$\varrho_{ij}(\boldsymbol{\Phi}) = \prod_{k:k<j \wedge (d_i, w_k) \in \boldsymbol{\Phi}} (1 - p_k) \times p_j. \tag{24}$$

Also, we use $\varrho_{i0}(\boldsymbol{\Phi})$ to denote the expected probability of transmitting data item $d_i$ via cellular networks, and we have the following definition.

***Definition 6.*** [*Expected Transmission Probability via Cellular Networks*] For a given data offloading solution $\boldsymbol{\Phi}$, the expected transmission probability via cellular networks for the data item $d_i$, i.e., $\varrho_{i0}(\boldsymbol{\Phi})$, satisfies:

$$\varrho_{i0}(\boldsymbol{\Phi}) = 1 - \sum_{j=1}^{m} \varrho_{ij}(\boldsymbol{\Phi}). \tag{25}$$

Then, the total expected transmission cost for a given data scheduling solution $\mathbf{\Phi}$ is expressed as follows:

$$f_{cost}(\mathbf{\Phi}) = \sum_{i=1}^{n} s_i \big( \sum_{j=1}^{m} \varrho_{ij}(\mathbf{\Phi}) c_j + \varrho_{i0}(\mathbf{\Phi}) \times C \big). \quad (26)$$

Here, $\sum_{j=1}^{m} \varrho_{ij}(\mathbf{\Phi}) c_j$ denotes the expected transmission cost per unit data traffic via all WiFi networks, while $\varrho_{i0}(\mathbf{\Phi}) \times C$ denotes the expected cost through cellular networks. The total cost of all data items that are transmitted through cellular networks, is denoted as $C \times \sum_{i=1}^{n} s_i$, which is fixed. Hence, for a given data offloading solution $\mathbf{\Phi}$, the total transmission cost via all WiFi networks can be computed. Different from the concept of *Offloading Utility Function* $\mathcal{U}(\mathbf{\Phi})$ in Definition 5, we define an offloading cost function as follows:

***Definition 7.*** [*Offloading Cost Function*] The offloading cost function of a data offloading solution $\mathbf{\Phi}$, denoted by $\mathcal{C}(\mathbf{\Phi})$, is the expected total cost of data items that will be offloaded to WiFi networks under this data offloading solution, which satisfies:

$$\mathcal{C}(\mathbf{\Phi}) = C \sum_{i=1}^{n} s_i - f_{cost}(\mathbf{\Phi}) = \sum_{i=1}^{n} s_i \big( \sum_{j=1}^{m} (C - c_j) \varrho_{ij}(\mathbf{\Phi}) \big). \quad (27)$$

Then, we formalize the extended problem as follows:

$$\begin{aligned} Maximize: \quad & \mathcal{C}(\mathbf{\Phi}) \quad (Extension) \\ Subject\ to: \quad & \sum_{i:(d_i,w_j)\in\mathbf{\Phi}} s_i \le q_j,\ 1 \le j \le m; \\ & t_i \ge \tau_j,\ \text{for } \forall (d_i, w_j) \in \mathbf{\Phi} \subseteq \boldsymbol{D} \times \boldsymbol{W}. \end{aligned}$$

Note that when $c_j = c (1 \le j \le m)$, $\mathcal{C}(\mathbf{\Phi}) = (C - c)\mathcal{U}(\mathbf{\Phi})$, indicating that $\mathcal{U}(\mathbf{\Phi})$ is actually a special form of *Offloading Cost Function* $\mathcal{C}(\mathbf{\Phi})$. Also, we use $\Delta\mathcal{C}_{\mathbf{\Phi}}((d_i, w_j))$ to denote the increment of $\mathcal{C}(\mathbf{\Phi})$ after adding a new offloading operation $(d_i, w_j)$, based on a given data offloading solution $\mathbf{\Phi}$. Thus, we have the following expression:

$$\Delta\mathcal{C}_{\mathbf{\Phi}}((d_i, w_j)) = \mathcal{C}(\mathbf{\Phi} \cup \{(d_i, w_j)\}) - \mathcal{C}(\mathbf{\Phi}). \quad (28)$$

Similarly, we use $\Delta\mathcal{C}_{\mathbf{\Phi}}(\mathbf{\Gamma})$ to denote the incremental offloading cost function value by adding a set of offloading operations $\mathbf{\Gamma}$ into the data offloading solution $\mathbf{\Phi}$.

### 5.2 The HDO Algorithm

To solve the extended problem in which the transmission costs via different WiFi APs are heterogeneous, we propose an extended algorithm, i.e., Heterogeneous Data Offloading (HDO) algorithm. Since the extended problem also has capacity constraints and deadline constraints, we adopt the similar strategy used in the offline algorithm to remove them. That is, we determine the priority of data offloading operations and further remove the deadline constraints in the first phase. Then, we select the offloading operations that satisfy the capacity constraints in the second phase.

In the first phase, we determine the priority of offloading operations and remove the deadline constraints. Concretely, the data items with smallest TTLs will be considered first. We also use $\mathbf{\Omega}_{d_i}$ to denote the set of deadline-satisfying offloading operations for the data item $d_i$. After we have derived the set $\mathbf{\Omega}_{d_i}$ for $d_i \in \boldsymbol{D}$, we always select the data offloading operations from $\mathbf{\Omega}_{d_i}$ ($d_i \in \boldsymbol{D}$). Based on this, the data offloading solution $\mathbf{\Phi}$ is deadline-satisfying.

**Algorithm 3** The HDO Algorithm

**Require:** $\boldsymbol{D}, \boldsymbol{W}$ where $w_j (\in \boldsymbol{W}) = \langle \tau_j, p_j, q_j, c_j \rangle$.
**Ensure:** $\mathbf{\Phi}$.

```
1: Initialize Φ = φ and Ω_{d_i} = φ (d_i ∈ D);
2: for d_i from d_1 to d_n do
3:   for w_j from w_1 to w_m do
4:     if τ_j ≤ t_i then
5:       Ω_{d_i} = Ω_{d_i} ∪ {(d_i, w_j)};
6:   while (∃ (d_i, w_j) ∈ Ω_{d_i}) do
7:     (d_i, w_{j*}) = argmax_{(d_i,w_j)∈Ω_{d_i}} ΔC_Φ((d_i, w_j));
8:     if s_i ≤ q_{j*} then
9:       Φ = Φ ∪ {(d_i, w_{j*})}, q_{j*} = q_{j*} − s_i;
10:    else
11:      Find a set Γ ⊆ Φ, s.t., s_i ≤ q_{j*} + Σ_{(d_x,w_{j*})∈Γ} s_x;
12:      if ΔC_Φ((d_i, w_{j*})) > ΔC_{Φ\Γ}(Γ) then
13:        Φ = Φ ∪ {(d_i, w_{j*})} \ Γ;
14:        q_{j*} = q_{j*} − s_i + Σ_{(d_x,w_{j*})∈Γ} s_x;
15:    Ω_{d_i} = Ω_{d_i} \ {(d_i, w_{j*})};
16: return Φ;
```

In the second phase, we consider the capacity constraints. More specifically, we first select the data offloading operation $(d_i, w_{j^*})$, which can increase the defined offloading cost function value most quickly. The greedy criterion of selection in each round is formulated as follows:

$$(d_i, w_{j^*}) = \underset{(d_i,w_j)\in\mathbf{\Omega}_{d_i}}{\operatorname{argmax}} \Delta\mathcal{C}_{\mathbf{\Phi}}((d_i, w_j)). \quad (29)$$

Then, if the data offloading operation $(d_i, w_{j^*})$ can satisfy the capacity constraint of the offloading opportunity $w_{j^*}$, it will be added into $\mathbf{\Phi}$ directly. Else, we will conduct the replacement procedure as follows. we first find a set $\mathbf{\Gamma} \subseteq \mathbf{\Phi}$ to ensure the capacity constraint of $w_{j^*}$ when replacing $\mathbf{\Gamma}$ by $(d_i, w_{j^*})$ (i.e., $s_i \le q_{j^*} + \sum_{(d_x,w_{j^*})\in\mathbf{\Gamma}} s_x$). Here, each data offloading operation $(d_x, w_{j^*}) \in \mathbf{\Gamma}$ is selected based on the minimum increment of offloading cost function value. Concretely, we organize the offloading operations in $\mathbf{\Phi}$ corresponding to $w_{j^*}$, i.e., $\{(d_x, w_{j^*}) | (d_x, w_{j^*}) \in \mathbf{\Phi}\}$, in the ascending order of the incremental offloading cost function value, i.e., $\Delta\mathcal{C}_{\mathbf{\Phi}\setminus\{(d_x,w_{j^*})\}}((d_x, w_{j^*}))$. According to this order, we add offloading operations into $\mathbf{\Gamma}$ one by one until $s_i \le q_{j^*} + \sum_{(d_x,w_{j^*})\in\mathbf{\Gamma}} s_x$ is satisfied. After comparing the incremental offloading cost function values of $(d_i, w_*)$ and $\mathbf{\Gamma}$, i.e., $\Delta\mathcal{C}_{\mathbf{\Phi}}((d_i, w_{j^*}))$ and $\Delta\mathcal{C}_{\mathbf{\Phi}\setminus\mathbf{\Gamma}}(\mathbf{\Gamma})$, we will use $(d_i, w_{j^*})$ to replace $\mathbf{\Gamma}$ if the former is larger than the later.

Based on this greedy strategy, we present the Heterogeneous Data Offloading (HDO) algorithm, as shown in Algorithm 3. The HDO algorithm also starts by initializing the offloading solution set $\mathbf{\Phi}$ and the sets of deadline-satisfying offloading operations $\mathbf{\Omega}_{d_i}$ ($d_i \in \boldsymbol{D}$), in Step 1. Then, the set $\mathbf{\Omega}_{d_i}$ for each data item $d_i \in \boldsymbol{D}$ is determined in Steps 2-5. We select the data offloading operation (e.g., $(d_i, w_{j^*})$) which can increase the offloading cost function value most quickly in Steps 6-7. If $(d_i, w_{j^*})$ satisfies the capacity constraint of $w_{j^*}$, it will be added into $\mathbf{\Phi}$ directly in Steps 8-9. Else, we will conduct the replacement procedure in detail. We first determine a set $\mathbf{\Gamma} \subseteq \mathbf{\Phi}$ satisfying $s_i \le q_{j^*} + \sum_{(d_x,w_{j^*})\in\mathbf{\Gamma}} s_x$

in Step 11. After comparing the incremental offloading cost function values of $(d_i, w_{j^*})$ and $\mathbf{\Gamma}$, we choose the better one for the data offloading, in Steps 12-14. Then, each considered offloading operation $(d_i, w_{j^*})$ will be deleted from $\mathbf{\Omega}_{d_i}$ in Step 15. At last, Algorithm 3 terminates and outputs the offloading solution in Step 16.

Additionally, the computation overhead of Algorithm 3 is also $O(m^2n^2)$.

### 5.3 Performance Analysis

In this subsection, we analyze the performance of Algorithm 3. After denoting the optimal solution to the extended problem as $\boldsymbol{opt}_T$, we have the following theorem:

***Theorem 5.*** The offloading cost function for the data offloading solution $\mathbf{\Phi}$ produced by HDO satisfies:

$$\frac{\mathcal{C}(\boldsymbol{opt}_T)}{\mathcal{C}(\mathbf{\Phi})} < 2. \tag{30}$$

*Proof*: Similar to the analysis of the FDO algorithm, we first consider two special solutions $\boldsymbol{opt}_T^*$ and $\boldsymbol{opt}_T^+$. More specifically, $\boldsymbol{opt}_T^*$ and $\boldsymbol{opt}_T^+$ denotes the solutions obtained by Algorithm 3, in which all data items are assumed to divisible and the capacity constraints of offloading opportunities can be broken once, respectively. Since $\boldsymbol{opt}_T$, $\boldsymbol{opt}_T^*$ and $\boldsymbol{opt}_T^+$ are produced by the same strategy, we have

$$\mathcal{C}(\boldsymbol{opt}_T^+) \geq \mathcal{C}(\boldsymbol{opt}_T^*) \geq \mathcal{C}(\boldsymbol{opt}_T). \tag{31}$$

When comparing $\boldsymbol{opt}_T^+$ and $\mathbf{\Phi}$, we assume that there are $g$ data offloading operations corresponding to $w_j$ in $\mathbf{\Phi}$, denoted as $\{(d_{i_x}, w_j)|1 \leq x \leq g\}$. For the offloading operation $(d_i, w_j)$, we assume $s_i > q_j$. According to the replacement strategy in Algorithm 3, we determine a set $\mathbf{\Gamma} \subseteq \mathbf{\Phi}$ so that $s_i \leq q_j + \sum_{(d_{i_x}, w_j) \in \mathbf{\Gamma}} s_{i_x}$. The offloading operations in $\mathbf{\Gamma}$ are selected based on the minimum incremental offloading cost function value, i.e., $\Delta\mathcal{C}_{\mathbf{\Phi}\setminus\{(d_{i_x}, w_j)\}}((d_{i_x}, w_j))$. In $\boldsymbol{opt}_T^+$, $(d_i, w_j)$ will be added directly since each offloading opportunity can be broken once. In contrast, we select $(d_i, w_j)$ if $\Delta\mathcal{C}_{\mathbf{\Phi}}((d_i, w_j)) > \Delta\mathcal{C}_{\mathbf{\Phi}\setminus\mathbf{\Gamma}}(\mathbf{\Gamma})$. Also, we use $\Delta\mathcal{C}_j(\mathbf{\Phi})$ to denote the incremental offloading cost function value corresponding to $w_j$ based on the solution $\mathbf{\Phi}$. Since $\sum_{j=1}^{m}(\varrho_{ij}(\mathbf{\Phi}) - \varrho_{ij}(\mathbf{\Phi} \setminus \{(d_i, w_j)\}))$ is actually the value of $\Delta\rho_{ij}(\mathbf{\Phi})$ defined in the offline case, we have

$$\frac{\Delta\mathcal{C}_j(\mathbf{\Phi})}{C - c_j} = \sum_{(d_{i_x}, w_j) \in (\mathbf{\Phi}\setminus\mathbf{\Gamma})} s_{i_x} \Delta\rho_{i_x j}(\mathbf{\Phi}) + \max\{\sum_{(d_{i_x}, w_j) \in \mathbf{\Gamma}} s_{i_x} \Delta\rho_{i_x j}, s_i \Delta\rho_{ij}(\mathbf{\Phi} \cup \{(d_i, w_j)\})\}; \tag{32}$$

$$\frac{\Delta\mathcal{C}_j(\boldsymbol{opt}_T^+)}{C - c_j} = \sum_{(d_{i_x}, w_j) \in \mathbf{\Phi}} s_{i_x} \Delta\rho_{i_x j} + s_i \Delta\rho_{ij}(\mathbf{\Phi} \cup \{(d_i, w_j)\}). \tag{33}$$

Then, we get

$$2\Delta\mathcal{C}_j(\mathbf{\Phi}) \geq \Delta\mathcal{C}_j(\boldsymbol{opt}_T^+) > \Delta\mathcal{C}_j(\boldsymbol{opt}_T), \text{ for } \forall j \in [1, m]. \tag{34}$$

Furthermore, we have

$$2\,\mathcal{C}(\mathbf{\Phi}) \geq \mathcal{C}(\boldsymbol{opt}_T^+) > \mathcal{C}(\boldsymbol{opt}_T). \tag{35}$$

Thus, this theorem holds. ■

## 6 PERFORMANCE EVALUATION

We conduct extensive simulations to evaluate the performances of our algorithms. Note that the FDO and NDO algorithms are designed for the scenario where transmission costs per unit data traffic via all WiFi networks are uniform, while the HDO algorithm is designed for the case in which transmission costs per unit data traffic via WiFi networks are different. Hence, the simulations are divided into two parts. The FDO, NDO algorithms and two compared algorithms are conducted in the same simulation settings, and the HDO algorithm is conducted with the compared algorithms in other simulation settings. More specifically, we first introduce the compared algorithms used in our simulations. Then, we present the real trace that we used and the corresponding settings. We also describe the synthetic traces and the relevant simulation settings. Finally, we present and analyze the obtained experimental results.

### 6.1 Algorithms in Comparison

As we discussed in Section 1, our problem is different from the existing works. Previous offloading algorithms cannot be applied in our problem directly. Hence, we implement two other scheduling algorithms for comparison: RS (Random Selection) and SRTF (Shortest Remaining Time First). In the RS algorithm, all data offloading operations are randomly selected from $\mathbf{\Omega}_{d_i}$ ($d_i \in \boldsymbol{D}$), while satisfying capacity constraints of offloading opportunities and deadline constraints of offloaded data items simultaneously. In the SRTF algorithm, all data items are first sorted in the ascending order of their TTLs (Time-To-Live). Then, SRTF selects the data items which satisfy the deadline constraints to each offloading opportunity one by one according to this order, until the total size of selected data items exceeds the capacity of the corresponding offloading opportunity. In other words, the data items with smallest TTLs will be offloaded first.

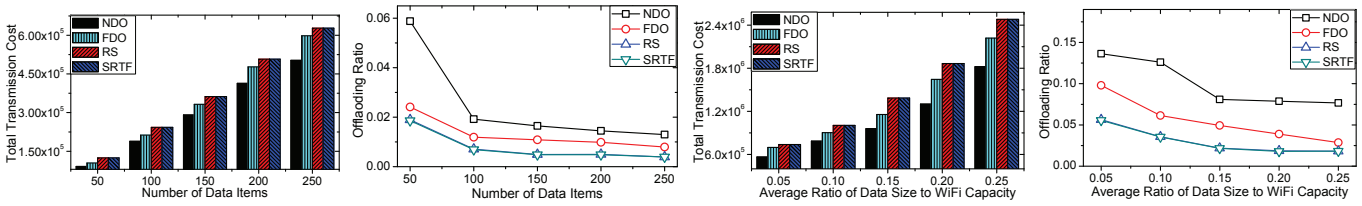
### 6.2 Real-trace Used and Simulation Settings

We adopt the newest real dataset [31] collected from two smartphone testbeds deployed in University at Buffalo (UB) and University of Notre Dame (ND): 5-month scans from PhoneLab at UB, and 32-month scans from NetSense at ND. Smartphones perform WiFi scans to adapt to the changing wireless environments caused by mobility, and WiFi scan results data, together with other WiFi related logs, is collected using the PhoneLab smartphone testbed over 5 months. Throughout the paper we mainly use the dataset called **WifiRSSIChange**, which contains 274 contents and one content corresponds to a specific WiFi AP. Moreover, every content includes multiple files, each of which means the logs of WiFi Received Signal Strength Indicator (RSSI) change for a day. For the simplicity of following description, these files are called LogDate. Each LogDate contains lots of entries, and each one for the log of WiFi RSSI state for a specific moment is called LogMoment. One LogMoment includes (1) WiFi SSID and BSSID, (2) a log timestamp, (3) WiFi linkspeed and (4) RSSI values.

We first select an arbitrary content (i.e., a WiFi AP) in **WifiRSSIChange**, and then choose a file (i.e., LogDate) in the content randomly. Next, we filtrate the **WifiRSSIChange** and find all WiFi APs which contain the LogDate with the

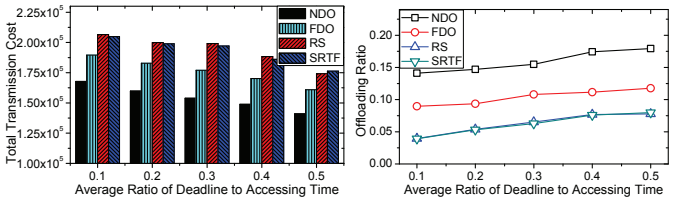
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2017.2720741, IEEE Transactions on Parallel and Distributed Systems






(a) Cost vs. Number of Data Items (b) Ratio vs. Number of Data Items (c) Cost vs. Data Size (d) Ratio vs. Data Size

Fig. 4. Performance comparisons: total transmission cost and offloading ratio vs. the numbers of data items and the average sizes of data items.




(a) Cost vs. TTLs of Data Items (b) Ratio vs. TTLs of Data Items

Fig. 5. Performance comparisons: the total transmission cost and offloading ratio vs. the different TTLs of data items.

same name as the selected LogDate. We let $m$ be the number of selected WiFi APs, and use $\boldsymbol{W}=\{w_1,\cdots,w_m\}$ to denote them. Since the WiFi linkspeed is changing over time in a LogDate, we select an arbitrary moment in the LogDate to denote the timestamp as the time that the user enters the communication range of one WiFi AP, i.e., $\tau_j$. Furthermore, we use the time period, during which the linkspeed of WiFi APs remains unchanged, to denote the time that the user stays in the communication range of one WiFi AP. Then, we calculate the capacity of each WiFi AP, i.e., $q_j$, according to the linkspeed and time period. In addition, we denote the probability $p_j$ of accessing the WiFi AP $w_j$ by using some random values which are generated from $(0,1]$. By that analogy, we can get the all parameters of satisfied WiFi APs ($\tau_j, q_j, p_j, 1\leq j\leq m$). Note that $\tau_1\leq\tau_2\leq\cdots\leq\tau_m$. Additionally, we use $\overline{\tau}$ and $\overline{q}$ to denote the average appearing time and average capacity of all WiFi APs, respectively.

Note that the simulation settings in the initial and extended problems are same, except for the transmission costs via all WiFi networks. Therefore, we first introduce the same simulation settings in both scenarios, and then present the different settings. Since there is no information about mobile user in the dataset **WifiRSSIChange**, we generate a fictitious mobile user and randomly produce the deadline-sensitive data items for it. More specifically, the number of data items is selected from $\{50,100,\cdots,250\}$. The size and Time-To-Lives (TTLs) of all data items are randomly produced in $[0,2l]$ and $[0,2t]$, where $l$ and $t$ denote the average size and TTL of data, respectively. Moreover, $l$ and $t$ are selected from $\{0.05\overline{q},0.1\overline{q},0.15\overline{q},0.2\overline{q},0.25\overline{q}\}$ and $\{0.1\overline{\tau},0.2\overline{\tau},0.3\overline{\tau},0.4\overline{\tau},0.5\overline{\tau}\}$, respectively.

For the simplicity of descriptions, the simulation settings where the transmission cost via WiFi APs is uniform, based on the real trace **WifiRSSIChange**, are called *settings1-1*. The settings in which the transmission costs are different, are called *settings1-2*. Hence, the different settings in initial and extended problems are presented as follows.

1) *settings1-1:* We let the transmission costs per unit data traffic via cellular network and WiFi networks be $C=0.1$ and $c=0.01$, respectively.

2) *settings1-2:* We let the transmission costs via WiFi networks be generated from $[0,2\delta]$ randomly, where $\delta$ is selected from $\{0.005,0.01,0.015,0.02,0.025\}$. Additionally, we still let the transmission cost via cellular network be $0.1$.

### 6.3 Synthetic Traces and Simulation Settings

In order to evaluate the performances of our algorithms with different attributes of WiFi APs, we also conduct a series of simulations on synthetic datasets. Similar to **WifiRSSIChange**, the simulation settings in the synthetic datasets are also classified into two parts. The simulation settings in which the transmission cost via WiFi networks is uniform based on synthetic trace, are called *settings2-1*, while the settings which consider the differences of transmission costs via WiFi networks, are called *settings2-2*. We first introduce the common settings in *settings2-1* and *settings2-2*. To evaluate the performance of our algorithms with different numbers of WiFi APs, we let the numbers of WiFi APs be selected from $\{5,10,\cdots,25\}$. More specifically, we take another two attributes of WiFi APs into consideration as follows. The capacities of WiFi APs are randomly generated in $[0,2L]$, where $L$ is selected from the set $\{1000,2000,\cdots,5000\}$. The probabilities of contacting WiFi networks are produced in $[0,2p]$ randomly, and $p$ is selected from the set $\{0.1,0.15,\cdots,0.3\}$, which is used to generate contact events. Note that the different settings in *settings2-1* and *settings2-2* are the same as *settings1-1* and *settings1-2*, respectively.

### 6.4 Evaluation Metrics

In a generic WiFi-based offloading model, the most important performance metrics include the amount of offloaded data and the offloading delay. However, in our mobile data offloading model, the offloading delay is used as the deadline constraints. Our primitive optimization problem is to minimize the total data transmission cost. Hence, the total transmission cost of all data items is used as a most leading metric in our simulation. In addition to the Total Transmission Cost (TTC), we also evaluated the data offloading ratio (OR) which is defined as Eq. 36, based on the initial problem and the extended problem.

$$OR=\begin{cases}\dfrac{\sum_{i=1}^{n}s_i\rho_i(\boldsymbol{\Phi})}{\sum_{i=1}^{n}s_i}; & (P_2)\\ \dfrac{\sum_{i=1}^{n}s_i(\sum_{j=1}^{m}\varrho_{ij}(\boldsymbol{\Phi}))}{\sum_{i=1}^{n}s_i}; & (Extension)\end{cases}\tag{36}$$

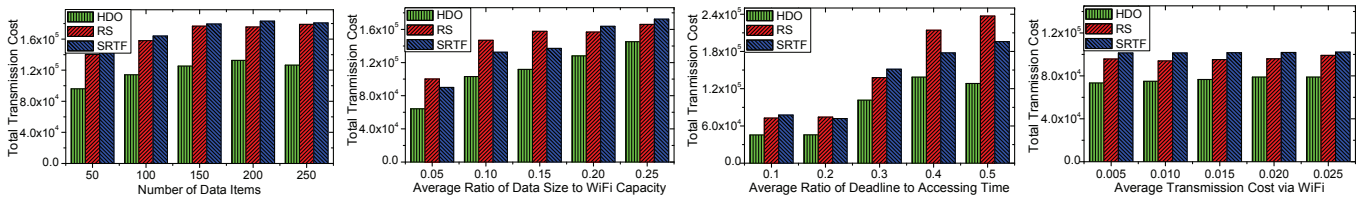
### 6.5 Evaluation Results

#### 6.5.1 Results of the Real Trace

1) *settings1-1:* We first present the results obtained by the four algorithms, FDO, NDO and two compared algorithms, according to the dataset **WifiRSSIChange**, in

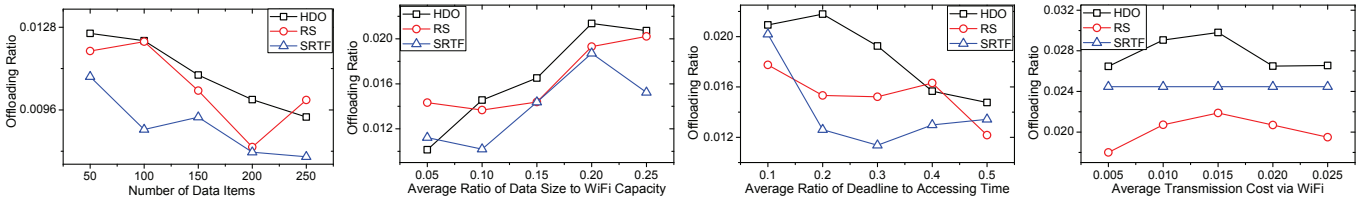
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2017.2720741, IEEE Transactions on Parallel and Distributed Systems






(a) Cost vs. Number of Data Items (b) Cost vs. Average Data Size (c) Cost vs. Average TTL of Data (d) Cost vs. Average Cost via WiFi

Fig. 6. Performance comparisons on the total transmission cost with the different numbers of data items, the average sizes of data items, the average TTLs of data items, and the average transmission costs per unit data traffic via WiFi APs.

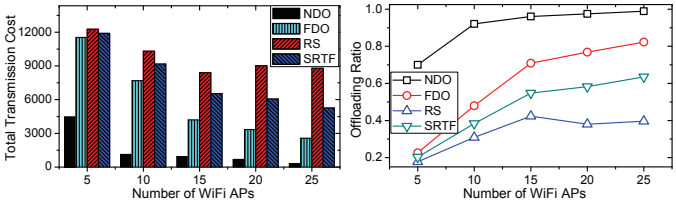



(a) Ratio vs. Number of Data (b) Ratio vs. Average Data Size (c) Ratio vs. Average TTL of Data (d) Ratio vs. Average Cost via WiFi

Fig. 7. Performance comparisons on the offloading ratio with the different numbers of data items, the average sizes of data items, the average TTLs of data items, and the average transmission costs per unit data traffic via WiFi APs.

*settings1-1*. The performance comparisons in terms of the number of data items $n$, average size $l$ and average TTL $t$ of data items, are presented in Figs. 4 and 5. In the simulations, when we conduct the four algorithms by changing one of the parameters $n$, $l$, and $t$, we keep others fixed. The results of the total transmission cost (TTC) and offloading ratio (OR) while changing the number of data items $n$ are shown in Figs. 4(a) and 4(b). Similarly, the results of TTC and OR by changing $l$ or $t$ are shown in Figs. 4(c), 4(d), 5(a), and 5(b), respectively. By analyzing the results, we conclude that NDO and FDO achieve about 26.6% and 10.9% smaller total transmission costs than the two compared algorithms as a whole, respectively. Additionally, we get that when the number of data items or the average size of data items increases, the TTCs of all algorithms increase, and the ORs decrease; when the average TTL of data items increases, the TTCs increase, while the ORs decrease. These simulations validate our theoretical analysis results.

2) *settings1-2:* Then, we show the results obtained from the HDO algorithm and two compared algorithms on **WifiRSSIChange**, according to *settings1-2*. The performance comparisons on TTCs of all data items and ORs with different numbers of data items $n$, average sizes of data items $l$, average TTLs of data items $t$ or average transmission cost via all WiFi networks $c$, are shown in Figs. 6 and 7. Here, the default values of $n$, $l$, $t$, and $\delta$ are set to 100, $0.1\overline{q}$, $0.1\overline{\tau}$ and 0.01, respectively. We see that when we change $n$, $l$, or $t$, HDO always achieves the best performance, and gets about 67.7%, 23.3%, 35.1% smaller total transmission costs than those of the compared algorithms, respectively. Moreover, when we change the parameters of $n$ or $l$, the TTCs and ORs almost have the same change trend as the *settings1-1*. We see that the ORs derived in HDO may be less than that obtained in the compared algorithms. This is because that the expected probability of offloading data to WiFi networks in HDO may be less than that of RS and SRTF, resulting in the lower ORs. However, the expected transmission cost through WiFi networks in HDO, considering the various transmission costs per unit data traffic via WiFi APs, may be much smaller than that in RS and SRTF. In addition, the performance comparisons on TTCs and ORs with the parameter $\delta$, are shown in Figs. 6(d) and 7(d). HDO gets about 22.5% smaller total transmission costs than those of the compared algorithms. Along with the increase of $\delta$, the TTCs obtained in all three algorithms increase correspondingly. This is because the average transmission cost per unit data size via all WiFi networks increases, resulting in the higher total transmission costs. These simulation results are still consistent with our theoretical analysis.




(a) Cost vs. Number of WiFi APs (b) Ratio vs. Number of WiFi APs

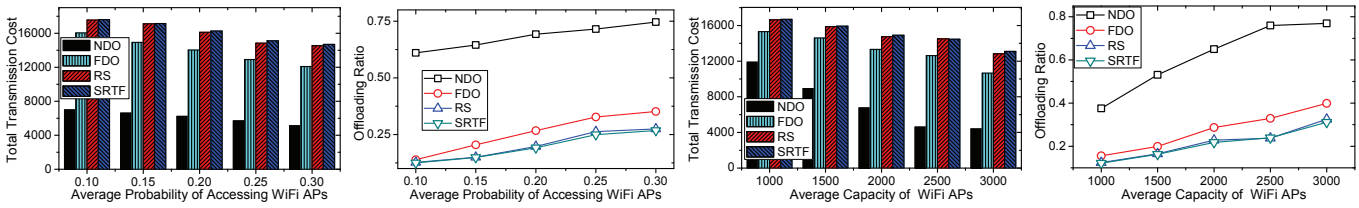
Fig. 8. Performance comparisons: the total transmission cost and offloading ratio vs. the number of WiFi APs.

### 6.5.2 Results of Synthetic Traces

1) *settings2-1:* Next, we present the simulation results of the four algorithms by using synthetic traces, based on *settings2-1*. We evaluate the performances of the four algorithms, taking the number of WiFi APs $m$, the accessible probability $p$ and capacity $L$ of each WiFi AP into consideration. Also, when we change one of the parameters for evaluation, we keep the other parameters fixed. The performances of TTCs and ORs by changing $m$ are presented in Figs. 8(a) and 8(b). As we expected, NDO achieves the best performance, and FDO follows. The total transmission costs of FDO and NDO are about 36.7% and 85.0% smaller than those of the compared algorithms, respectively. The offloading ratio of NDO achieves the best result; the FDO and the two compared algorithms decrease stepwise. Then, we evaluate the performances of the four algorithms by changing the parameter $p$ or $L$. The results of TTCs and ORs are shown in Fig. 9. FDO and NDO achieve about 12.3% and 57.2% smaller total transmission costs than the two compared algorithms, respectively. Additionally, NDO has the biggest offloading ratio, and FDO follows. When $p$ or $L$ increases, the total transmission costs decrease. The offloading ratios of all algorithms increase simultaneously.

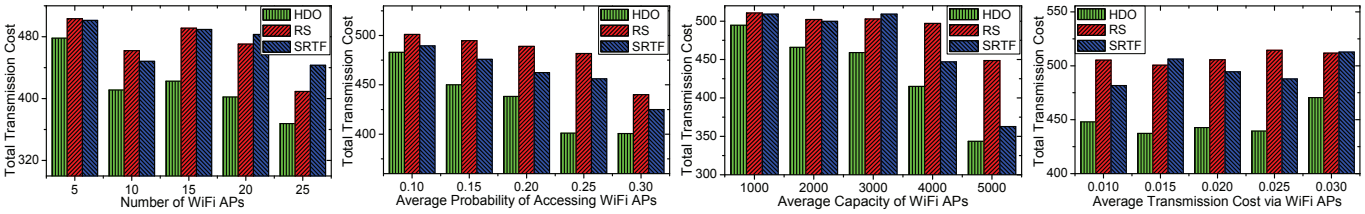
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPDS.2017.2720741, IEEE Transactions on Parallel and Distributed Systems





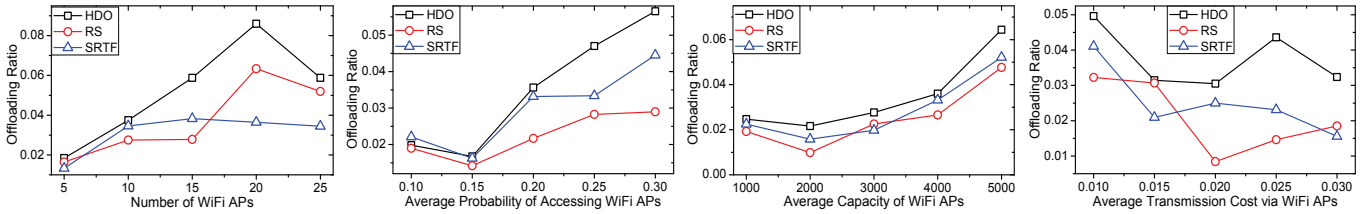

(a) Cost vs. Average Probability (b) Ratio vs. Average Probability (c) Cost vs. Average Capacity (d) Ratio vs. Average Capacity

Fig. 9. Performance comparisons: the total transmission cost and offloading ratio vs. the average accessing probability and capacity of WiFi APs.




(a) Cost vs. Number of WiFi APs (b) Cost vs. Average Probability (c) Cost vs. Average Capacity (d) Cost vs. Transmission Cost

Fig. 10. Performance comparisons on the total transmission cost with the number of WiFi APs, the average probability of accessing WiFi APs, the average capacity of WiFi APs, and the average transmission cost via WiFi APs.




(a) Ratio vs. Number of WiFi APs (b) Ratio vs. Average Probability (c) Ratio vs. Average Capacity (d) Ratio vs. Transmission Cost

Fig. 11. Performance comparisons on the offloading ratio with the different numbers of WiFi APs, average probabilities of accessing WiFi APs, average capacities, and average transmission cost per unit data traffic of WiFi APs.

2) *settings2-2:* Lastly, we evaluate the performances of three algorithms: HDO, RS, and SRTF, by considering the number of WiFi APs, average probability of accessing WiFi APs, average capacity of all WiFi APs, and average transmission cost via WiFi networks. Likewise, when we change one of the parameters for evaluation, we keep the other parameters fixed. The simulation results are shown in Figs. 10 and 11. The results demonstrate that HDO achieves better performances than RS and SRTF, about $10.8\%$ smaller total transmission costs than the two compared algorithms. In addition, along with the increase of the number of WiFi APs, average capacity of WiFi APs, and average probability of mobile user visiting WiFi APs, the TTCs decrease. The ORs of all algorithms increase simultaneously. In contrast, along with the increase of the transmission costs via all WiFi networks, the TTCs increase. These simulations remain consistent with our theoretical analysis results.

## 7 RELATED WORK

In this paper, we focus on the data transmission problem in mobile cloud computing applications, in which these offloading data items must share a combinatorially probabilistic optimization objective. By far, the latest works [2–4, 7, 10, 17, 33, 37] concentrate on offloading traffic from cellular networks to other coexisting networks to provide better service. In a broad sense, offloading cellular traffic can be mainly classified into two categories: WiFi-based offloading [1, 12, 20, 21, 25, 27, 32, 33] and DTNs-based offloading [14, 22, 28, 29, 34, 38].

Generally, data offloading through third party WiFi APs or femtocell APs requires the cooperation and agreement of both the mobile cellular network operators (MNOs) and AP owners (APOs). Gao *et al.* [12] developed a model to analyze the interaction among one MNO and multiple APOs by using Nash bargaining theory. Lee *et al.* [20] studied the economic benefits generated due to delayed WiFi offloading, by analyzing the traffic load balance between cellular networks and WiFi networks. In the work [18], the heterogeneous network is responsible for collecting the network information, and decides the specific portion of traffic to be transmitted via WiFi networks, to maximize the per-user throughput. Wang *et al.* [33] proposed an auction-based algorithm to achieve both load balancing among base stations and fairness among mobile users, which optimally solves the global proportional fairness problem in polynomial time by transforming it into an equivalent matching problem. Additionally, Mehmeti *et al.* [25] proposed a queueing analytic model for delayed WiFi offloading, and derive the mean delay, offloading efficiency, and other metrics of interest, as a function of the user's "patience". The authors in the work [10] proposed and evaluated an integrated architecture exploiting the opportunistic networking paradigm to migrate data traffic from cellular networks to metropolitan WiFi APs. Different from the aforementioned works, our purpose is to minimize the total transmission cost of all data items, from the perspective of mobile users. Additionally, we take the deadline constraints and the capacity constraints into consideration simultaneously.

Furthermore, our work is also different from the offloading using DTNs. For example, Zhuo *et al.* [38] mainly investigated the trade-off between the amount of traffic being offloaded and the users' satisfaction. Then, they proposed a novel incentive offloading target where users with

high delay tolerance and large offloading potential will be prioritized for traffic offloading. Li *et al.* in [22] established a mathematical framework to study the problem of multiple-type mobile data offloading under realistic assumptions, where (1) mobile data is heterogeneous in terms of size and lifetime; (2) mobile users have different data subscribing interests; and (3) the storages of offloading helpers are limited. Then they formulated the objective of achieving maximum mobile data offloading as a submodular function maximization problem with multiple linear constraints of limited storage, and proposed three algorithms to solve this challenging optimization problem. The authors of work [34] proposed the framework of traffic offloading assisted by Social Network Services (SNS) via opportunistic sharing, to offload SNS-based cellular traffic by user-to-user sharing, which is formulated as a special target-set selection problem. Han *et al.* [14] exploited opportunistic communications to facilitate information dissemination in the emerging Mobile Social Networks (MSNs) and thus reduce the amount of mobile data traffic. The work [14] investigated the target-set selection problem for information delivery to minimize the cellular data traffic. Different from the existing problems, we formulate the objective of achieving the minimum of data transmission cost from a mobile device to the cloud side.

We deduce the problem as an optimization problem with a probabilistic combination of multiple 0-1 knapsack constraints, which also differs from the existing MKP [5, 26]. The closest to our problem is the Multiple 0-1 Knapsack problem with Assignment Restrictions and Capacity Constraints (MK-AR-CC) [26], in which multiple knapsacks is independent. By contrast, the multiple 0-1 knapsack constraints in our optimization problem involves a probabilistic combination, and each item, which is allowed to be assigned to multiple knapsacks, shares a combinatorially probabilistic optimization objective in our model. Thus, our problem is more complicated than MK-AR-CC. The method used in MK-AR-CC [26] cannot solve our problem. Since dynamic programming cannot solve MK-AR-CC [26] to get an optimal result, it cannot solve our problem.

Besides, some recent research efforts have been focused on other aspects while alleviating the traffic load over cellular networks. For example, Saad *et al.* [30] considered the problem of uplink user association in small cell networks, and then proposed a distributed algorithm to solve it. Barbera *et al.* [2] designed and built a working implementation of CDroid, a system that tightly couples the device OS to its cloud counterpart, where the cloud-side handles data traffic through the device efficiently and caches code or data optimally for possible future offloading. Higgins *et al.* [16] designed a useful mobile prefetching system, where they used a cost-benefit analysis to decide when to prefetch data, and employed goal-directed adaptation to minimize application response time while meeting budgets for battery lifetime and cellular data usage.

## 8 CONCLUSION

We have studied the problem of how to offload multiple mobile data items from cellular networks to WiFi networks to minimize the total transmission cost from the perspective of mobile users. These data items are heterogeneous in data sizes and TTLs, and the capacities of WiFi networks are limited. We first prove the NP-hardness of the offloading problem. Then, we design the offline algorithm (FDO) and the online algorithm (NDO) to solve the optimization problem. We prove that FDO achieves the approximation ratio of 2, and NDO achieves the competitive ratio of 2. In addition, we extend our problem and solution to a more general scenario where the transmission costs per unit data traffic via WiFi networks are heterogeneous. We further propose the heterogeneous data offloading algorithm (HDO), and analyze the performance of HDO. At last, extensive simulations based on real and synthetic traces are conducted to verify the significant performances of our algorithms.

## REFERENCES

[1] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting mobile 3g using wifi. In *ACM MobiSys*, 2010.
[2] M. V. Barbera, S. Kosta, A. Mei, V. C. Perta, and J. Stefa. Mobile offloading in the wild: Findings and lessons learned through a real-life experiment with a new cloud-aware system. In *IEEE INFOCOM*, 2014.
[3] M. Bennis, M. Simsek, A. Czylwik, W. Saad, S. Valentin, and M. Debbah. When cellular meets wifi in wireless small cell networks. *Communications Magazine*, 51(6):44–50, 2013.
[4] K. Berg and M. Katsigiannis. Optimal cost-based strategies in mobile network offloading. In *ICST CROWNCOM*, 2012.
[5] C. Chekuri and S. Khanna. A ptas for the multiple knapsack problem. 2005.
[6] N. Cheng, N. Lu, N. Zhang, X. Shen, and J. Mark. Vehicular wifi offloading: Challenges and solutions. *Vehicular Communications*, 1(1):13–21, 2014.
[7] M. H. Cheung and J. Huang. Optimal delayed wi-fi offloading. In *Modeling Optimization in Mobile, Ad Hoc Wireless Networks (WiOpt), International Symposium on*, 2013.
[8] I. Cisco. Cisco visual networking index: Global mobile data traffic forecast update. *2015-2020 White paper*, 2015.
[9] H. Deng and I.-H. Hou. Online scheduling for delayed mobile offloading. In *IEEE INFOCOM*, 2015.
[10] S. Dimatteo, P. Hui, B. Han, and V. O. K. Li. Cellular traffic offloading through wifi networks. In *IEEE MASS*, 2011.
[11] G. Gao, M. Xiao, J. Wu, K. Han, and L. Huang. Deadline-sensitive mobile data offloading via opportunistic communications. In *IEEE SECON*, 2016.
[12] L. Gao, G. Iosifidis, J. Huang, L. Tassiulas, and D. Li. Bargaining-based mobile data offloading. *IEEE Journal on SAC*, 32(6):1114–1125, 2014.
[13] S. Guo, B. Xiao, Y. Yang, and Y. Yang. Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing. In *IEEE INFOCOM*, 2016.
[14] B. Han, P. Hui, V. Kumar, M. Marathe, J. Shao, and A. Srinivasan. Mobile data offloading through opportunistic communications and social participation. *IEEE Transactions on Mobile Computing*, 11(5):821–834, 2012.
[15] Z. Hao, Y. Tang, Y. Zhang, E. Novak, N. Carter, and Q. Li. Smoc: a secure mobile cloud computing platform. In *IEEE INFOCOM*, 2015.
[16] B. D. Higgins, J. Flinn, T. J. Giuli, B. Noble, C. Peplin, and D. Watson. Informed mobile prefetching. In *ACM MobiSys*, 2012.
[17] X. Hou, P. Deshpande, and S. R. Das. Moving bits from 3g to metro-scale wifi for vehicular network access: An integrated transport layer solution. In *IEEE ICNP*, 2011.
[18] B. H. Jung, N.-O. Song, and D. K. Sung. A network-assisted user-centric wifi-offloading model for maximizing peruser throughput in a heterogeneous network. *IEEE Transactions on Vehicular Technology*, 63(4):1940–1945, 2014.
[19] X. Kang, Y.-K. Chia, and S. Sun. Mobile data offloading through a third-party wifi access point: An operator's

perspective. *IEEE Transactions on Wireless Communications*, 13(10):5340–5351, 2014.
[20] J. Lee, Y. Yi, S. Chong, and Y. Jin. Economics of wifi offloading: trading delay for cellular capacity. *IEEE Transactions on Wireless Communications*, 13(3):1540–1554, 2014.
[21] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong. Mobile data offloading: how much can wifi deliver? *IEEE Transactions on Wireless Communications*, 21(2):536–550, 2010.
[22] Y. Li, M. Qian, D. Jin, P. Hui, Z. Wang, and S. Chen. Multiple mobile data offloading through disruption tolerant networks. *IEEE Transactions on Mobile Computing*, 13(7):1579–1596, 2014.
[23] Z. Lu, X. Sun, and T. L. Porta. Cooperative data offloading in opportunistic mobile networks. In *INFOCOM*, 2016.
[24] S. Martello, D. Pisinger, , and P. Toth. Dynamic programming and strong bounds for the 0-1 knapsack problem. In *Management Science*, 1999.
[25] F. Mehmeti and T. Spyropoulos. Is it worth to be patient? analysis and optimization of delayed mobile data offloading. In *IEEE INFOCOM*, 2014.
[26] S. Miyazaki, N. Morimoto, and Y. Okabe. Approximability of two variants of multiple knapsack problems. In *Algorithms and Complexity*. Springer, 2015.
[27] V. F. S. Mota, D. F. Macedo, Y. Ghamri-Doudane, and J. M. S. Nogueira. On the feasibility of wifi offloading in urban areas: The paris case study. In *Wireless Days*, 2013.
[28] W. Peng, F. Li, X. Zou, and J. Wu. The virtue of patience: Offloading topical cellular content through opportunistic links. In *IEEE MASS*, 2013.
[29] K. K. Rachuri, C. Mascolo, M. Musolesi, and P. J. Rentfrow. Sociablesense: exploring the trade-offs of adaptive sampling and computation offloading for social sensing. In *ACM MOBICOM*, 2011.
[30] W. Saad, Z. Han, R. Zheng, M. Debbah, and H. V. Poor. A college admissions game for uplink user association in wireless small cell networks. In *IEEE INFOCOM*, 2014.
[31] J. Shi, C. Qiao, D. Koutsonikolas, and G. Challen. CRAWDAD dataset buffalo/phonelab-wifi. Downloaded from http://crawdad.org/buffalo/phonelab-wifi/20160309.
[32] S. Singh, H. S. Dhillon, and J. G. Andrews. Offloading in heterogeneous networks: Modeling, analysis, and design insights. *IEEE TWC*, 12(5):2484–2497, 2013.
[33] W. Wang, X. Wu, L. Xie, and S. Lu. Femto-matching: Efficient traffic offloading in heterogeneous cellular networks. In *IEEE INFOCOM*, 2015.
[34] X. Wang, M. Chen, Z. Han, D. Wu, and T. Kwon. Toss: traffic offloading by social network service-based opportunistic sharing in mobile social networks. In *IEEE INFOCOM*, 2014.
[35] L. Xiang, S. Ye, Y. Feng, B. Li, and B. Li. Ready, set, go: Coalesced offloading from mobile devices to the cloud. In *IEEE INFOCOM*, 2014.
[36] O. B. Yetim and M. Martonosi. Dynamic adaptive techniques for learning application delay tolerance for mobile data offloading. In *IEEE INFOCOM*, 2015.
[37] D. Zhang and C. K. Yeo. Optimal handing-back point in mobile data offloading. In *IEEE Vehicular Networking Conference*, 2012.
[38] X. Zhuo, W. Gao, G. Cao, and S. Hua. An incentive framework for cellular traffic offloading. *IEEE Transactions on Mobile Computing*, 13(3):541–555, 2014.

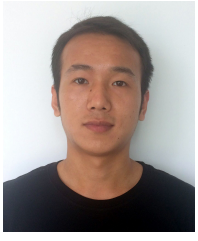
**Guoju Gao** received his B.S. degree in information security from the University of Science and Technology of Beijing, Beijing, China, in 2014. He is currently working toward a PhD degree on computer science and technology at the University of Science and Technology of China, Hefei, China. His research interests are in the areas of networking and communications, including mobile social networks, device-to-device communication, and vehicle Ad Hoc networks.

**Mingjun Xiao** is an associate professor in the School of Computer Science and Technology at the University of Science and Technology of China (USTC). He received his Ph.D. from USTC in 2004. In 2012, he was a visiting scholar at Temple University, under the supervision of Dr. Jie Wu. He is a TPC member of ICDCS 2015, Mobihoc 2014, and has served as a reviewer for many journal papers. His main research interests include mobile crowdsensing, mobile social networks, and vehicular ad hoc networks. He has published over 50 papers in refereed journals and conferences, including TC, TMC, TON, TPDS, INFOCOM, ICNP, etc.

**Jie Wu** is the Associate Vice Provost for International Affairs at Temple University. He also serves as Director of Center for Networked Computing and Laura H. Carnell professor. He served as Chair of Computer and Information Sciences from 2009 to 2016. Prior to joining Temple University, he was a program director at the National Science Foundation and was a distinguished professor at Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. Dr. Wu regularly publishes in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE Transactions on Service Computing and the Journal of Parallel and Distributed Computing. Dr. Wu was general co-chair for IEEE MASS 2006, IEEE IPDPS 2008, IEEE ICDCS 2013, ACM MobiHoc 2014, ICPP 2016, and IEEE CNS 2016, as well as program co-chair for IEEE INFOCOM 2011 and CCF CNCC 2013. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). Dr. Wu is a CCF Distinguished Speaker and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.

**Kai Han** received his B.S. and Ph.D degrees in computer science from the University of Science and Technology of China, Hefei, China, in 1997 and 2004, respectively. He is currently a professor at the School of Computer Science and Technology, University of Science and Technology of China, China. His research interests include wireless ad hoc and sensor networks, mobile and cloud computing, combinatorial and stochastic optimization, algorithmic game theory, as well as machine learning.

**Liusheng Huang** received his MS degree in computer science from University of Science and Technology of China, Anhui, in 1988. He is a professor at the School of Computer Science and Technology, University of Science and Technology of China. His main research interests include delay tolerant networks and Internet of things. He serves on the editorial board of many journals. He has published 6 books and more than 200 papers.



**Zhenhua Zhao** received his B.S. degree in the School of Computer Science and Technology at the University of Science and Technology of China, Hefei, China, in 2015. He is currently a master student of computer science at University of Science and Technology of China, Hefei, China. His research interests include deep learning, neural network, and Internet of Things.