An Incentive Framework for Cellular Traffic Offloading

Xuejun Zhuo, Wei Gao, *Member, IEEE*, Guohong Cao, *Fellow, IEEE*, and Sha Hua

Abstract—Cellular networks (e.g., 3G) are currently facing severe traffic overload problems caused by excessive traffic demands. Offloading part of the cellular traffic through other forms of networks, such as Delay Tolerant Networks (DTNs) and WiFi hotspots, is a promising solution. However, since these networks can only provide intermittent connectivity to mobile users, utilizing them for cellular traffic offloading may result in a non-negligible delay. As the delay increases, the users' satisfaction decreases. In this paper, we investigate the tradeoff between the amount of traffic being offloaded and the users' satisfaction. We provide a novel incentive framework to motivate users to leverage their delay tolerance for cellular traffic offloading. To minimize the incentive cost given an offloading target, users with high delay tolerance and large offloading potential should be prioritized for traffic offloading. To effectively capture the dynamic characteristics of users' delay tolerance, our incentive framework is based on reverse auction to let users proactively express their delay tolerance by submitting bids. We further illustrate how to predict the offloading potential of the users by using stochastic analysis for both DTN and WiFi cases. Extensive trace-driven simulations verify the efficiency of our incentive framework for cellular traffic offloading.

Index Terms—Cellular Traffic Offloading, Auction, Delay Tolerant Networks, WiFi Hotspots.

1 Introduction

The recent popularization of cellular networks (e.g., 3G) provide mobile users with ubiquitous Internet access. However, the explosive growth of user population and their demands for bandwidth-eager multimedia content raise big challenges to the cellular networks. A huge amount of cellular data traffic has been generated by mobile users, which exceeds the capacity of cellular network and hence deteriorates the network quality [1]. To address such challenges, the most straightforward solution is to increase the capacity of cellular networks, which however is expensive and inefficient. Some researchers studied on how to select a small part of key locations to realize capacity upgrade, and shift traffic to them by exploiting user delay tolerance [2]. Remaining the capacity of cellular networks unchanged, offloading part of cellular traffic to other coexisting networks would be another desirable and promising approach to solve the overload problem.

Some recent research efforts have been focusing on offloading cellular traffic to other forms of networks, such as DTNs and WiFi hotspots [3] [4] [5], and they generally focus on maximizing the amount of cellular traffic that can be offloaded. In most cases, due to user mobility, these networks available for cellular traffic offloading only provide intermittent and opportunistic network connectivity to the users, and the traffic offloading hence results in non-negligible data downloading delay. In general, more offloading opportunities may appear by requesting the mobile users to wait for a longer time before actually downloading the data from the cellular networks, but this will also make the users become more impatient and hence reduce their satisfaction.

In this paper, we focus on investigating the tradeoff between the amount of traffic being offloaded and the users' satisfaction, and propose a novel incentive framework to motivate users to leverage their delay tolerance for traffic offloading. Users are provided with incentives; i.e., receiving discount for their service charge if they are willing to wait longer for data downloading. During the delay, part of the cellular data traffic may be opportunistically offloaded to other networks mentioned above, and the user is assured to receive the remaining part of the data via cellular network when the delay period ends.

The major challenge of designing such an incentive framework is to minimize the incentive cost of cellular network operator which includes the total discount provided to the mobile users, subject to an expected amount of traffic being offloaded. To achieve this goal, two important factors should be taken into account; i.e., the *delay tolerance* and *offloading potential* of the users. The users with high delay tolerance and large offloading potential should be prioritized in cellular traffic offloading.

First, with the same period of delay, the users with higher delay tolerance require less discount to compensate their satisfaction loss. To effectively capture the dynamic characteristics of the users' delay tolerance, we propose an incentive mechanism based on reverse auc-

X. Zhuo is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084 China. E-mail: xzhuo1220@gmail.com.
W. Gao is with the Department of Electrical Engineering and Computer Science, University of Tennessee at Knoxville, 302 Min Kao Building, 1520 Middle Drive, Knoxville, TN 37919 USA. E-mail: weigao@utk.edu.
G. Cao is with the Department of Computer Science and Engineering, 354G Information Sciences and Technology Building, The Pennsylvania State University, University Park, PA 16802 USA. E-mail: gcao@cse.psu.edu.
S. Hua is with the Department of Electrical and Computer Engineering, Polytechnic Institute of New York University, 6 MetroTech Center, NY 11201 USA. E-mail: shua01@students.poly.edu.

Digital Object Indentifier 10.1109/TMC.2013.15

tion which is proved to conduct a justified pricing. In our mechanism, the users act as sellers to send bids, which include the delay that they are willing to experience and the discount that they want to obtain for this delay. Such discount requested by users is called "*coupon*" in the rest of the paper. The network operator then acts as the buyer to buy the delay tolerance from the users.

Second, with the same period of delay, users with larger offloading potential are able to offload more data traffic. For example, the offloading potential of a user who requests popular data is large, because it can easily retrieve the data pieces from other contacted peer users during the delay period. Also, if a user has high probability to pass by some WiFi hotspots, its offloading potential is large. To effectively capture the offloading potential of the users, we propose two accurate prediction models for DTN and WiFi case respectively.

The optimal auction outcome is determined by considering both the delay tolerance and offloading potential of the users to achieve the minimum incentive cost, given an offloading target. The auction winners set up contracts with the network operator for the delay they wait and the coupon they earn, and other users directly download data via cellular network at the original price. More specifically, the contribution of the paper is three-fold:

- We propose a novel incentive framework, *Win-Coupon*, based on reverse auction, to motivate users leveraging their delay tolerance for cellular traffic offloading, which have three desirable properties: 1) truthfulness, 2) individual rationality, 3) low computational complexity.
- We provide an accurate model using stochastic analysis to predict users' offloading potential based on their data access and mobility patterns in the DTN case.
- We provide an accurate Semi Markov based prediction model to predict users' offloading potential based on their mobility patterns and the geographical distribution of WiFi hotspots in the WiFi case.

The rest of the paper is organized as follows. In Section 2 we briefly review the existing work. Section 3 provides an overview of our approach and the related background. Section 4 describes the details of our incentive framework, and proves its desirable properties. Section 5 evaluates the performance of Win-Coupon through trace-driven simulations and Section 6 discusses further research issues. Section 7 concludes the paper.

2 Related Work

To deal with the problem of cellular traffic overload, some studies propose to utilize DTNs to conduct offloading. Ristanovic et al. [6] propose a simple algorithm, Mix-Zones, to let the operator notify users to switch their interfaces for data fetching from other peers when the opportunistic DTN connections occur. Whitbeck et al. [7] design a framework, called Push-and-Track, which includes multiple strategies to determine how many copies should be injected by cellular network and to whom, and then leverages DTNs to offload the traffic. Han et al. [3] provide three simple algorithms to exploit DTNs to facilitate data dissemination among mobile users, in order to reduce the overall cellular traffic. Many research efforts have focused on how to improve the performance of data access in DTNs. In [8], the authors provide theoretical analysis to the stationary and transient regimes of data dissemination. Some later works [9] [10] disseminate data among mobile users by exploiting their social relations. Being orthogonal with how to improve the performance of data access in DTNs, in this paper, we propose an accurate model to capture the expected traffic that can be offloaded to DTNs to facilitate our framework design.

Public WiFi can also be utilized for cellular traffic offloading. In [6], the authors design HotZones to enable users turning on WiFi interfaces when a WiFi connection is expected to occur based on the user mobility profile and location information of hot zones covered by WiFi. In [5], the authors measure the availability and the offloading performance of public WiFi based on vehicular traces. Lee et al. [4] consider a more general mobile scenario, and present a quantitative study on delayed and on-the-spot offloading by using WiFi. The prediction of future WiFi availability is important to the offloading scheme design, and has been studied in [11] [12]. In [11], the authors propose to enable mobile users to schedule their data transfers when higher WiFi transmission rate can be achieved based on the prediction. In [12], a Lyapunov framework based algorithm, called SALSA, is proposed to optimize the energy-delay tradeoff of the mobile devices with both cellular network and WiFi interfaces. Different from the existing work, in this paper, we propose an accurate model to predict how much traffic that can be offloaded via WiFi hotspots if a mobile user is willing to wait for certain delay time.

All the existing offloading studies have not considered the satisfaction loss of the users when a longer delay is caused by traffic offloading. To motivate users to leverage their delay tolerance for cellular traffic offloading, we propose an auction based incentive framework. Auction has been widely used in network design. Applying auction in the spectrum leasing is one of the most practical applications. Federal Communications Commission (FCC) has already auctioned the unused spectrum in the past decade [13], and there are a large amount of works on wireless spectrum auctions [14] [15]. Moreover, auction has also been applied for designing incentive mechanism to motivate selfish nodes to forward data for others [16] [17]. However, none of them has applied auction techniques to cellular traffic offloading.

This paper substantially extends the preliminary version of our results appeared in [18]. In [18], we mainly focused on how to stimulate users to offload cellular traffic via DTNs. In this paper, we propose a more general framework which considers both DTNs and WiFi case. We provide an accurate model to predict users' offloading potential in the WiFi case and perform trace-driven simulations to evaluate its performance. In addition, we

change the data query model in [18] to more realistic Zipf-like distribution to evaluate our framework.

3 Overview

3.1 The Big Picture

In this section, we give an overview of the Win-Coupon framework. By considering the users' delay tolerance and offloading potential, Win-Coupon uses a reverse auction based incentive mechanism to motivate users to help cellular traffic offloading. Figure 1 illustrates the main idea. The network operator acts as the buyer, who offers coupons to users in exchange for them to wait for some time and opportunistically offload the traffic. When users request data, they are motivated to send bids along with their request messages to the network operator. Each bid includes the information of how long the user is willing to wait and how much coupon he wants to obtain as a return for the extra delay. Then, the network operator infers users' delay tolerance. In addition, users' offloading potential should also be considered when deciding the auction outcome. Based on the historical system parameters collected, such as users' data access and mobility patterns, their future value can be predicted by conducting network modeling, and then based on the information, users' offloading potential can be predicted.

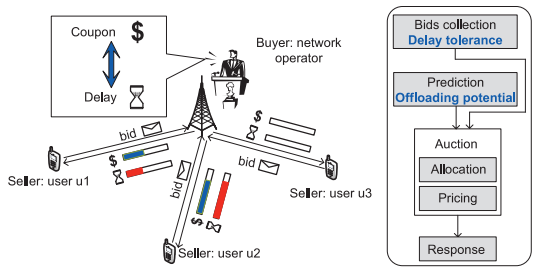


Fig. 1. The main idea of Win-Coupon

The optimal auction outcome is to minimize the network operator's incentive cost subject to a given offloading target according to the bidders' delay tolerance and offloading potential. The auction contains two main steps: *allocation* and *pricing*. In the allocation step, the network operator decides which bidders are the winners and how long they need to wait. In the pricing step, the network operator decides how much to pay for each winner. Finally, the network operator returns the bidders with the auction outcome which includes the assigned delay and the value of coupon for each bidder. The winning bidders (e.g. user $u1$ and $u2$ shown in Figure 1) obtain the coupon, and are assured to receive the data via cellular network when their promised delay is reached. For example, suppose p is the original data service charge, if user $u1$ obtains the coupon with value c in return for delay t, it only needs to pay $p-c$ for the data service. During the delay period, $u1$ may retrieve some data pieces from other intermittently available networks, e.g., by contacting other peers which cache the data or moves into the wireless range of APs. Once delay t passes, the cellular network pushes the remaining data pieces to $u1$ to assure the promised delay. The losing bidders (e.g. user $u3$ shown in Figure 1) immediately download data via cellular network at the original price.

3.2 User Delay Tolerance

With the increase of downloading delay, the user's satisfaction decreases accordingly, the rate of which reflects the user's delay tolerance. To flexibly model users' delay tolerance, we introduce a *satisfaction function* $S(t)$, which is a monotonically decreasing function of delay t, and represents the price that the user is willing to pay for the data service with the delay. The satisfaction function is determined by the user himself, his requested data, and various environmental factors. We assume that each user has an upper bound of delay tolerance for each data. Once the delay reaches the bound, the user's satisfaction becomes zero, indicating that the user is not willing to pay for the data service. Figure 2 shows an example of the satisfaction function $S(t)$ of a specific user for a specific data, where t_{bound} is the upper bound of the user's delay tolerance. p is the original charge for the data service, and the satisfaction curve represents the user's expected price for the data as the delay increases. For example, with delay t_1 the user is only willing to pay p_1 instead of p. $p - p_1$ is the satisfaction loss caused by delay t_1.

Fig. 2. Satisfaction function

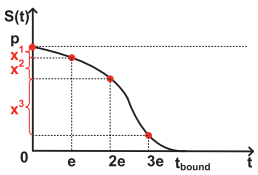


Fig. 3. Private value

3.3 Auctions

In economics, auction is a typical method to determine the value of a commodity that has an undetermined and variable price. It has been widely applied to many fields. Most auctions are *forward auction* which involves a single seller and multiple buyers, and the buyers send bids to compete for obtaining the commodities sold by the seller. In this paper, we use *reverse auction* [19] which involves a single buyer and multiple sellers, and the buyer decides its purchase based on the bids sent by the sellers. To begin with, we introduce some notations.

Bid (b_i): It is submitted by bidder i to express i's valuation on the resource for sale, which is not necessarily true.

Private value (x_i): It is the true valuation made by bidder i for the resources; i.e., the true price that i wants to obtain for selling the resource. This value is only known by i.

Market-clearing price (p_i): It is the price actually paid by the buyer to bidder i. This price cannot be less than the bids submitted by i.

Utility (u_i): It is the residual worth of the sold resource for bidder i, namely the difference between i's market-clearing price p_i and private value x_i.

The bidders in the auction are assumed to be rational and risk neutral. A common requirement for auction design is the so-called *individual rationality*.

Definition 1: An auction is with individual rationality if all bidders are guaranteed to obtain non-negative utility.

The rational bidders decide their bidding strategy to maximize their utility. Let $\mathcal{N}$ denote the set of all bidders. The concept of *weakly dominant strategy* is defined as:

Definition 2: $b_i = \beta_i$ is a weakly dominant strategy for user i if and only if: $u_i(\beta_i, \beta_{-i}) \geq u_i(\beta_i', \beta_{-i}),\ \forall \beta_i' \neq \beta_i$.

Here $\beta_{-i} = \{\beta_1, \beta_2, \cdots, \beta_{i-1}, \beta_{i+1}, \cdots, \beta_{|\mathcal{N}|}\}$ denotes the set of strategies of all other bidders except for bidder i. We can see a weakly dominant strategy maximizes i's utility regardless of the strategies chosen by all other bidders. If for every bidder, truthfully setting its bid to its private value is a weakly dominant strategy, the auction is *truthful* (strategyproof).

Definition 3: An auction is truthful if each bidder, say i, has a weakly dominant strategy, in which $b_i = x_i$.

The truthfulness eliminates the expensive overhead for bidders to strategize against other bidders and prevents the market manipulation. Also, it assures the efficient allocation by encouraging bidders to reveal their true private values. Vickrey-Clarke-Groves (VCG) [20] [21] [22] is the most well-studied auction format, due to its truthful property. However, VCG only ensures truthfulness when the optimal allocation can be found, and it usually cannot assure the truthfulness when applied to the approximation algorithms [23]. Unfortunately, the allocation problem in Win-Coupon is NP-hard. It is known that an allocation algorithm leads to be truthful if and only if it is monotone [24]. In order to maintain the truthfulness property, we design an approximation algorithm and make it monotone in a deterministic sense. Therefore, our incentive mechanism possesses three important properties: 1) truthfulness, 2) individual rationality, and 3) low computational complexity.

4 Main Approach of Win-Coupon

In this section, we illustrate the details of Win-Coupon. In the reverse auction based Win-Coupon, the buyer is the network operator who pays coupon in exchange for longer delay of the users. The sellers are the cellular users who sell their delay tolerance to win coupon. The right side of Figure 1 shows the flow chart of Win-Coupon. At first, the network operator collects the bids to derive the delay tolerance of the bidders, and predicts their offloading potential. Then, based on the derived information, a reverse auction is conducted, which includes two main steps: allocation and pricing. Finally, the network operator returns the auction outcome to the bidders.

In the rest of this section, we first introduce the bidding. Then, we present auction mechanism and prove its properties. Finally, we illustrate how to predict bidders' offloading potential for both DTN and WiFi cases.

4.1 Bidding

To obtain coupon, the users attach bids with their data requests to reveal their delay tolerance. For each user, the upper bound t_{bound} of its delay tolerance can be viewed as the resources that it wants to sell. The user can divide t_{bound} into multiple time units, and submit multiple bids $\mathbf{b} = \{b^1, b^2, \cdots, b^l\}$ to indicate the value of coupon it wants to obtain for each additional time unit of delay, where l equals $\lfloor \frac{t_{bound}}{e} \rfloor$, and e is the length of one time unit. By receiving these bids, the network operator knows that the user wants to obtain coupon with value no less than $\sum_{k=1}^{k_i} b^k$ by waiting for k_i time units. The length of time unit e can be flexibly determined by the network operator. Shorter time unit results in larger bids with more information, which increases the performance of the auction, but it also induces more communication overhead and higher computational complexity. To simplify the presentation, in the rest of the paper delay t is normalized by time unit e.

As shown in Figure 2, $p - S(t)$ is the satisfaction loss of the user due to delay t. Then, $p - S(t)$ represents the private value of the user to the delay, namely the user wants to obtain the coupon with value no less than $p - S(t)$ for delay t. Thus, the private value of the user to each additional time unit of delay is $\mathbf{x} = \{x^1, x^2, \cdots, x^l\}$, where x^k ($k \in \{1, \cdots, l\}$), equals $S(k-1) - S(k)$. For example, as shown in Figure 3, the user wants to obtain the coupon with value no less than x^1 if it waits for one time unit, $x^1 + x^2$ for two time units, and $x^1 + x^2 + x^3$ for three time units. Generally, the user can set its bids with any value at will, however we will prove that the auction in Win-Coupon is truthful, which guarantees that the users would bid their private value; that is, $b^k = x^k$, for all k.

4.2 Auction Algorithms

Win-Coupon is run periodically in each auction round. Usually, the auction would result in an extra delay for the bidders to wait for the auction outcome. However, different from other long-term auctions, such as the FCC-style spectrum leasing, the auction round in our scenario is very short, since hundreds of users may request cellular data service at the same time. Also, because the bidders who are willing to submit bids are supposed to have a certain degree of delay tolerance, the extra delay caused by auction can be neglected. Next, we describe two main steps of the auction: allocation and pricing.

4.2.1 Allocation

In traditional reverse auction, the allocation solution is purely decided by the bids; i.e., the bidders who bid the lowest price win the game. However, in our scenario, besides the bids which express the bidders' delay tolerance, the offloading potential of the bidders should also be considered. Let $\{t_1, t_2, \cdots, t_{|\mathcal{N}|}\}$ represent the allocation solution, where t_i denotes the length of delay that network operator wants to buy from bidder i.

Note that since each bidder is asked to wait for integer multiples of time unit, t_i is an integer. If t_i equals zero, bidder i loses the game. The allocation problem in Win-Coupon can be formulated as follows:

Definition 4: The allocation problem is to determine the optimal solution $\{t_1, t_2, \cdots, t_{|\mathcal{N}|}\}$ which minimizes the total incentive cost, subject to a given offloading target.

$$\min_{t_i} \quad \sum_{i\in\mathcal{N}} \sum_{k=1}^{t_i} b_i^k \tag{1}$$

$$s.t. \quad \sum_{i\in\mathcal{N}} V_i^d(t_i) \geq v_0 \tag{2}$$

$$\forall i,\ t_i \in \{0, 1, 2, \cdots, l_i\}. \tag{3}$$

In Eq.(1), $\sum_{k=1}^{t_i} b_i^k$ denotes the value of the coupon that the network operator needs to pay bidder i in exchange for its delay t_i. $V_i^d(t)$ in Eq.(2) denotes the expected traffic that can be offloaded, if bidder i downloads data d and is willing to wait for delay t. We will provide the details on how to predict $V_i^d(t)$ in Section 4.3 and 4.4 for both DTN and WiFi cases respectively. We assume that within a short auction round, each bidder only requests one data item, so that each i is mapped to a single d. Thus, this constraint ensures that the total expected offloaded traffic is no less than the offloading target v_0. Eq.(3) ensures that the delay that each bidder i waits does not exceed l_i, the maximum number of time units that i is willing to wait.

It is easy to prove that our allocation problem can be reduced to the 0-1 knapsack problem, under the assumption that $l_i = 1$, for all i. The 0-1 knapsack problem is proved to be NP-hard, and thus our problem is also NP-hard. Next, we transform the original problem, and derive the optimal solution of the new problem by dynamic programming (DP).

We replace constraint (2) with $\sum_{i\in\mathcal{N}} \lfloor V_i^d(t_i)M \rfloor \geq \lfloor v_0 M \rfloor$, where $M = 10^n$ is a common scalar, to transform $V_i^d(t_i)$ and v_0 into integers. In this way, a table for DP can be formed and the values in the table can be resolved gradually. With a larger M, the optimal solution of the new problem becomes closer to that of the original problem, and the former converges to the latter when M increases to infinity. On the other hand, larger M increases the computational complexity of the algorithm, and when M is infinite, the approximation algorithm has pseudo-polynomial complexity. The operator needs to select a proper scaler M to balance the accuracy and the computational complexity of the allocation algorithm. We define $\hat{V}_i^d(t_i) = \lfloor V_i^d(t_i)M \rfloor$, and $\hat{v}_0 = \lfloor v_0 M \rfloor$.

Let T_i^v denote the minimum time units of delay that bidder i needs to wait to offload v volume of traffic, and C_i^v denote the corresponding value of coupon that i requests. Note that here and in the rest of this section, traffic volume v is scaled by M. Then, we have:

$$T_i^v = \arg\min_k \{\hat{V}_i^d(k) \geq v\} \tag{4}$$

$$C_i^v = \sum_{k=1}^{T_i^v} b_i^k \tag{5}$$

We use $\mathcal{B} = \{\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_{|\mathcal{N}|}\}$ to denote the bid set including all the bids sent by the bidders in set $\mathcal{N}$, and use $\mathcal{B}_i = \{\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_i\}$ to denote the bid set including all the bids sent by the first i bidders in $\mathcal{N}$. Assume only the first i bidders join the auction, we define $\mathcal{C}_{\mathcal{B}_i}^v$ to be the minimal incentive cost incurred to achieve a given offloading target v with the bid set $\mathcal{B}_i$, and define $\mathcal{T}_{\mathcal{B}_i}^v = \{t_1, t_2, \cdots, t_i\}$ to be the corresponding optimal allocation solution. Our allocation algorithm is illustrated in Algorithm 1 with $\mathcal{T}_{\mathcal{B}}^{\hat{v}_0}$ giving the optimal allocation solution. In Algorithm 1, line 4 to 8 update $\mathcal{T}_{\mathcal{B}_i}^v$, $\mathcal{C}_{\mathcal{B}_i}^v$ to include a new bidder at each iteration. Line 6 searches for the optimal allocation solution $\mathcal{T}_{\mathcal{B}_i}^v$ to obtain minimal $\mathcal{C}_{\mathcal{B}_i}^v$. The complexity of the algorithm is $O(|\mathcal{N}|\hat{v}_0^2)$.

4.2.2 Pricing

The VCG-style pricing is generally used in forward auction, which involves single seller with limited resources for sale, and multiple buyers. The bidders who have the highest bid win the game, and each winning bidder pays the "opportunity cost" that its presence introduces to others. It is proved that this pricing algorithm provides bidders with the incentives to set their bids truthfully. Based on the basic idea, in our pricing algorithm, the network operator also pays bidder i the coupon with value equal to the "opportunity cost" exerted to all the other bidders due to i's presence. Given the offloading target $\hat{v}_0$, let $c1 = \mathcal{C}_{\mathcal{B}\setminus\{\mathbf{b}_i\}}^{\hat{v}_0}$ denote the total value of coupons requested by all the bidders under the optimal allocation solution without the presence of i. Let $c2 = (\mathcal{C}_{\mathcal{B}}^{\hat{v}_0} - \sum_{k=1}^{t_i} b_i^k)$ denote the total value of coupons requested by all the bidders except for i under the current optimal allocation solution. Then, i's "opportunity cost" is defined as the difference between $c1$ and $c2$. Thus, i's market-clearing price can be derived as:

$$p_i = c1 - c2 = \mathcal{C}_{\mathcal{B}\setminus\{\mathbf{b}_i\}}^{\hat{v}_0} - (\mathcal{C}_{\mathcal{B}}^{\hat{v}_0} - \sum_{k=1}^{t_i} b_i^k). \tag{6}$$

Algorithm 1: Win-coupon-Allocation $(\mathcal{N}, \mathcal{B})$

1 **for** $v = 0$ *to* $\hat{v}_0$ **do**
2 $\mathcal{T}_{\mathcal{B}_1}^v = \{T_1^v\}$;
3 $\mathcal{C}_{\mathcal{B}_1}^v = C_1^v$;
4 **for** $i = 2$ *to* $|\mathcal{N}|$ **do**
5 **for** $v = 0$ *to* $\hat{v}_0$ **do**
6 $s* = arg\min_{s\in[0,v]}\{\mathcal{C}_{\mathcal{B}_{i-1}}^s + C_i^{v-s}\}$;
7 $\mathcal{T}_{\mathcal{B}_i}^v = \mathcal{T}_{\mathcal{B}_{i-1}}^{s*} \cup \{T_i^{v-s*}\}$;
8 $\mathcal{C}_{\mathcal{B}_i}^v = \mathcal{C}_{\mathcal{B}_{i-1}}^{s*} + C_i^{v-s*}$;
9 **return** $\mathcal{T}_{\mathcal{B}}^{\hat{v}_0}$, $\mathcal{C}_{\mathcal{B}}^{\hat{v}_0}$;

The pricing algorithm is illustrated in Algorithm 2, and the computational complexity of the algorithm is $O(A|\mathcal{N}|\hat{v}_0^2)$, where A is the number of winning bidders.

4.2.3 Properties

In Section 4.2.1, 4.2.2, we have shown that Win-Coupon can be solved in polynomial time, if a suitable scalar M

Algorithm 2: Win-coupon-Pricing $(\mathcal{N}, \mathcal{B}, \mathcal{T}_{\mathcal{B}}^{\hat{v}_0}, \mathcal{C}_{\mathcal{B}}^{\hat{v}_0})$

```
for i = 1 to |N| do
    if i is the winning bidder then
        Win-Coupon-Allocation(N \ {i}, B \ {b_i});
        p_i = C^{v̂0}_{B\{b_i}} − (C^{v̂0}_B − Σ_{k=1}^{t_i} b_i^k);
    else
        p_i = 0;
return p_i, for all i;
```

is selected. Next, we prove that Win-Coupon also has the properties: truthfulness and individual rationality.

Theorem 1: In Win-Coupon, for each bidder, say i, setting its bids truthfully, i.e., $\mathbf{b}_i = \mathbf{x}_i$, is a weakly dominant strategy.

Proof: We assume that when bidder i sets its bids truthfully, i.e., $\mathbf{b}_i = \mathbf{x}_i$, network operator would buy delay t_i from it, and its market-clearing price is $p_i = \mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} - (\mathcal{C}^{\hat{v}_0}_{\mathcal{B}} - \sum_{k=1}^{t_i} b_i^k)$. Then, the utility obtained by i is $u_i = \mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} - (\mathcal{C}^{\hat{v}_0}_{\mathcal{B}} - \sum_{k=1}^{t_i} b_i^k) - \sum_{k=1}^{t_i} x_i^k$. Now, suppose that bidder i sets its bids untruthfully, i.e., $\mathbf{b}'_i \neq \mathbf{x}_i$. Then, the length of delay t'_i that network operator would buy from i falls into two cases: 1) $t'_i = t_i$ and 2) $t'_i \neq t_i$.

In case 1), the market-clearing price paid to bidder i would become $p'_i = \mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}} - (\mathcal{C}^{\hat{v}_0}_{\mathcal{B}} - \sum_{k=1}^{t'_i} {b'}_i^k)$. Due to the sub-problem optimality in deriving the incentive cost $\mathcal{C}^{\hat{v}_0}_{\mathcal{B}}$, $\mathcal{C}^{\hat{v}_0}_{\mathcal{B}} = \mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t'_i)}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}} + \sum_{k=1}^{t'_i} {b'}_i^k$. Then we have $p'_i = \mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}} - \mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t'_i)}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}}$, where $\mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}}$ and $\mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t'_i)}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}}$ are independent of the bids sent by bidder i. Therefore, if $t'_i = t_i$, then $p'_i = p_i$, which is unaffected and the utility of bidder i has no change.

In case 2), similarly the market-clearing price paid to bidder i would be changed to $p'_i = \mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}} - \mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t'_i)}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}}$. Then, the new utility obtained by i equals $u'_i = \mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}} - \mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t'_i)}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}} - \sum_{k=1}^{t'_i} x_i^k$. The utility gain obtained by bidder i by setting $\mathbf{b}'_i \neq \mathbf{b}_i$ can be calculated as:

$$\begin{aligned}\Delta u_i &= u'_i - u_i = (\mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}} - \mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t'_i)}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}} - \textstyle\sum_{k=1}^{t'_i} x_i^k) \\ &-(\mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} - \mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t_i)}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} - \textstyle\sum_{k=1}^{t_i} x_i^k) \\ &= (\mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t_i)}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} + \textstyle\sum_{k=1}^{t_i} x_i^k) - (\mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t'_i)}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}} + \textstyle\sum_{k=1}^{t'_i} x_i^k).\end{aligned}$$

When bidder i sets its bids truthfully as $\mathbf{b}_i = \mathbf{x}_i$, Buying delay with length t_i from it is the optimal solution of the network operator to minimize the incentive cost. Therefore, keeping other settings unchanged, the solution with buying delay t'_i instead of t_i from bidder i leads to larger incentive cost. Thus we have $(\mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t_i)}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} + \sum_{k=1}^{t_i} x_i^k) < (\mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t'_i)}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} + \sum_{k=1}^{t'_i} b_i^k)$. Since $\mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t'_i)}_{\mathcal{B}\setminus\{\mathbf{b}_i\}}$ is independent of $\mathbf{b}_i$, and $\mathbf{b}_i = \mathbf{x}_i$, we have $\mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t'_i)}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} + \sum_{k=1}^{t'_i} b_i^k = \mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t'_i)}_{\mathcal{B}\setminus\{\mathbf{b}'_i\}} + \sum_{k=1}^{t'_i} x_i^k$. Thus $\Delta u_i < 0$, under this case, bidder i also cannot obtain higher utility by setting $\mathbf{b}_i \neq \mathbf{x}_i$. □

Theorem 2: In Win-Coupon, all bidders are guaranteed to obtain non-negative utility.

Proof: We have proved that for each bidder, say i, if it participates the auction game, setting its bids truthfully as $\mathbf{b}_i = \mathbf{x}_i$, is a weakly dominant strategy. The utility that i obtains equals $u_i = \mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} - \mathcal{C}^{\hat{v}_0 - \hat{V}_i^d(t_i)}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} - \sum_{k=1}^{t_i} x_i^k = \mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} - \mathcal{C}^{\hat{v}_0}_{\mathcal{B}}$, where t_i is the optimal length of delay that the network operator would buy from i to minimize the incentive cost. Since $\mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}_i\}}$ is the incentive cost incurred by the solution with network operator buying delay with length of 0 instead of t_i from bidder i, we have $\mathcal{C}^{\hat{v}_0}_{\mathcal{B}\setminus\{\mathbf{b}_i\}} \geq \mathcal{C}^{\hat{v}_0}_{\mathcal{B}}$. Therefore, Win-Coupon guarantees that all bidders would obtain non-negative utility. □

4.2.4 Reserve Price

In forward auction, the seller has the option to declare a *reserve price* for its resources. The reserve price means that the seller would rather withhold the resources if the bids are too low (lower than the reserve price). In Win-Coupon, to guarantee the network operator obtaining non-negative profit, we also provide it with the option to set a reserve price to indicate the highest incentive cost it is willing to pay for offloading one traffic unit. If the value of coupon asked by the bidders exceeds the reserve price, the network operator would rather not trade with them. Suppose that the network operator sets a reserve price c_0, which means that it is willing to spend at most c_0 for offloading one traffic unit. Adding the reserve price c_0 can be understood as adding a virtual bidder in the auction round. The bids sent by the bidder is $\{c_0, c_0, \cdots, c_0\}$, and it can offload one traffic unit per one time unit of delay.

4.3 Prediction of Offloading Potential: the DTN case

By motivating users to wait for some time, part of the cellular traffic can be offloaded to other intermittently available networks. One such example is DTN which generally coexists with cellular networks, and does not rely on any infrastructure. Mobile users can share data via DTNs by contacting each other. In urban area with higher user density, mobile users have more chances to contact other users who have their requested data. Large data requests such as video clips tend to drain most of the cellular network resource, and such requests can also tolerate some delay. By offloading them via DTNs, the payload of cellular network can be significantly reduced. In this section, we illustrate how to predict the potentials of the users to offload their traffic via DTNs.

4.3.1 Models

Due to high node mobility, large data items are hard to be completely transmitted when two nodes contact. In [25], it has been proved that the Random Linear Network Coding (RLNC) techniques can significantly improve the data transmission efficiency, especially when the transmission bandwidth is limited. Thus, in our model, RLNC is adopted to encode the original data into a set of coded packets. As long as the requester collects enough

number of any linearly independent coded packets of its requested data, the data can be reconstructed. Due to page limit, we omit the details of RLNC and suggest interested readers to refer to [26]. Besides, when the data item is large, multi-generation network coding is usually adopted. To balance the data transmission efficiency, the computational, and the transmission cost, how to decide the generation size and how to schedule their generation transmissions should be carefully considered. Since this is not the focus of this paper, we will not discuss it in the paper.

In the following analysis to simplify the presentation, we assume that the contact process between each node pair follows i.i.d. Poisson distribution with rate λ, and exactly one packet can be transmitted when two nodes contact. Our analysis based on these assumptions can be extended to more general cases such as node pairs follow contact processes other than Poisson, and they can transmit arbitrary number of packets during a contact.

4.3.2 The Main Idea of Prediction

We describe the rationale of prediction in one auction round. The starting time of this round is denoted by t_0. The objective of the prediction is to calculate the expected volume of traffic $V_i^d(t)$ that can be offloaded to DTNs, if node i requests data item d and is willing to wait for delay t. By using RLNC, data item d has been encoded into a set of coded packets, and any s_d linear independent packets can be used to reconstruct d. We say that a node retrieves an *innovative* coded packet, if the packet is linearly independent to all the coded packets cached in the node. It has been proven that as long as the subspace spanned by the sender's code vectors does not belong to receivers, the probability to obtain an innovative packet from the sender is at least $1 - 1/|E|$, where $|E|$ is the size of Galois field to generate coding coefficients which is generally set to 2^8 [27]. Therefore, we assume that when a node contacts another node which has cached some coded packets of the requested data, it can always retrieve an innovative packet with a very high probability. This assumption has been commonly used in prior works [25] [28]. In practice, if the size of the finite field to generate the coding coefficients is large enough, the probability is very close to 1.

Node i can retrieve one packet by contacting a node which has some coded packets of data item d, until it has collected all s_d packets. We use variable T_r ($1 \leq r \leq s_d$) to represent the time that node i retrieves r packets of d, and let $F_{T_r}(t)$ denote the Cumulative Distribution Function (CDF) of T_r. Thus, $V_i^d(t)$ can be computed as follows:

$$V_i^d(t) = h \int_0^t R(t)(1 - F_{T_{s_d}}(t))\mathrm{d}t \tag{7}$$

where h is the size of one coded packet. $1 - F_{T_{s_d}}(t)$ is the probability that node i has not received all s_d packets at time t. $R(t)$ represents the receiving rate of node i at time t. Due to the i.i.d Poisson contact processes with rate λ between node pairs, $R(t)$ equals $\lambda N_d(t)$, where $N_d(t)$ denotes the total number of nodes that has at least one packet of data d at time t. Next, we describe how to calculate $N_d(t)$ and $F_{T_{s_d}}(t)$.

4.3.3 Calculation of $N_d(t)$

Based on nodes' interests to data d, all the nodes in the network except for node i can be divided into two classes: $\mathcal{D}$ and $\mathcal{I}$, where $\mathcal{D}$ contains all the non-interesters and $\mathcal{I}$ contains all the interesters. The interesters include both the nodes which are downloading the data, and those which have already downloaded the data. To facilitate our analysis, we further divide class $\mathcal{I}$ into $s_d + 2$ subclasses: $\mathcal{I}_0, \mathcal{I}_1, \cdots, \mathcal{I}_{s_d}, \mathcal{I}_E$, based on the nodes' current downloading progress of data d. Specifically, $\mathcal{I}_j$ ($j \in [0, s_d]$) includes all the nodes in the network other than node i which have already downloaded j packets of data d, and $\mathcal{I}_E$ includes all the nodes which have finished data downloading before and already deleted the data from their buffer.

Based on our description of Win-Coupon in Section 4, each node in class $\mathcal{I}$ has a promised delay. When the delay ends, the network operator would automatically push the remaining data packets to the node. For the nodes which lose the auction or choose to directly download data without bidding, their waiting delay is zero. To characterize the different waiting delays of the nodes, we further decompose each class $\mathcal{I}_j$ ($j \in [0, s_d - 1]$) into $g + 1$ subclasses $\mathcal{I}_{j1}, \mathcal{I}_{j2}, \cdots, \mathcal{I}_{jg}, \mathcal{I}_{j\infty}$, where g denotes the maximal remaining delay of the current downloading nodes. $\mathcal{I}_{jk}$ ($j \in [0, s_d - 1], k \in [1, g]$) includes the nodes in class $\mathcal{I}_j$ whose remaining delay is k time slots. For the new requesters which transit from class $\mathcal{D}$ to class $\mathcal{I}$ after time t_0, we assume they prefer waiting a long delay to retrieve the complete data d via DTNs. Such new requesters in class $\mathcal{I}_j$ are classified into the subclass $\mathcal{I}_{j\infty}$. Under this assumption, the derived $V_i^d(t)$ is a lower bound of the actual value, due to the following reason. If the delays of the new requesters are limited, after the delay, the network operator would directly push the traffic to them, which potentially increases the data copies in the network, and results in a larger $V_i^d(t)$.

Next, we analyze how the network states vary with time. Let $N_{\mathcal{C}}(t)$ denote the number of nodes in class $\mathcal{C}$ at time $t > t_0$. For example, $N_{\mathcal{I}_{jk}}(t)$ denotes the number of nodes in class $\mathcal{I}_{jk}$ at time t. The class transition can be modeled as two types: *active* and *passive* transition.

- Active transition: A node would actively transit from one class to another class by three ways: 1) The node is in class $\mathcal{D}$, and transits to class $\mathcal{I}_{0\infty}$ by generating a request for data d; 2) The node is in class $\mathcal{I}_j$ ($j \in [0, s_d - 1]$) and transits to class $\mathcal{I}_{j+1}$ by retrieving a packet from a contacted node; 3) The node is in class $\mathcal{I}_{s_d}$, and transits to class $\mathcal{I}_E$ by deleting d from its buffer. The active transition processes are marked as the black arrows in Figure 4.
- Passive transition: A node would passively transit from class $\mathcal{I}_{jk}$ ($j \in [0, s_d - 1], k \in [2, g]$) to class $\mathcal{I}_{j(k-1)}$,

and transit from class $\mathcal{I}_{j1}$ ($j \in [0, s_d-1]$) to class $\mathcal{I}_{s_d}$, when one time slot passes. Note that the latter transition is caused by the network operator pushing the remaining traffic to the node when its promised delay ends. The passive transition processes are marked as the blue dotted arrows in Figure 4.

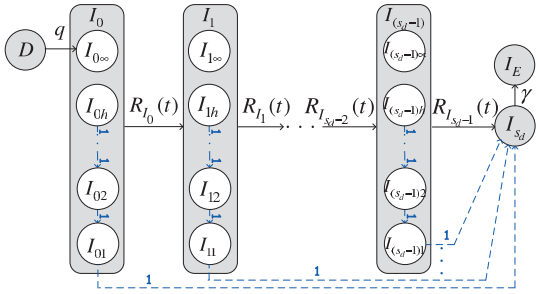


Fig. 4. Class transition processes

In the following, we use Ordinary Differential Equations (ODEs) to first analyze the active transition process. We assume that there are qdt portion of the nodes in class $\mathcal{D}$ that transit to class $\mathcal{I}_{0\infty}$ between time t and $t+dt$, where dt is infinitesimal, and q is the query rate decided by the popularity of data d. As a node in class $\mathcal{I}_j$ ($j \in [0, s_d-1]$) contacts another node in class $\mathcal{I}_{j'}$ ($j' \in [1, s_d]$), the former node retrieves a packet from the latter and transit into class $\mathcal{I}_{j+1}$. Let $R_{\mathcal{I}_j}(t)$ ($j \in [0, s_d-1]$) denote the receiving rate of the node in class $\mathcal{I}_j$ at time t, and we have:

$$R_{\mathcal{I}_0}(t) = \lambda(\sum_{y=1}^{s_d} N_{\mathcal{I}_y}(t)) \quad (8)$$

$$R_{\mathcal{I}_j}(t) = \lambda(\sum_{y=1}^{s_d} N_{\mathcal{I}_y}(t) - 1), \forall j \in [1, s_d-1] \quad (9)$$

where 1 in Eq.(9) represents the node itself, since the node cannot retrieve new packet from itself. After a node has completely downloaded data d, it may delete it from its local buffer. We assume that there are γdt portion of the nodes in class $\mathcal{I}_{s_d}$ that delete data d and transit to class $\mathcal{I}_E$, between time t and $t+dt$. Given all the initial value of the number of nodes in each classes at the starting time, $N_{\mathcal{I}_{jk}}(t)$ ($j \in [0, s_d], k \in [1, g] \cup \infty$) can be computed by solving the following ODEs.

$$\frac{d(N_{\mathcal{I}_{0\infty}}(t))}{dt} = N_{\mathcal{D}}(t)q - N_{\mathcal{I}_{0\infty}}(t)R_{\mathcal{I}_0}(t) \quad (10)$$

$$\frac{d(N_{\mathcal{I}_{0k}}(t))}{dt} = -N_{\mathcal{I}_{0k}}(t)R_{\mathcal{I}_0}(t), \forall k \in [1, g] \quad (11)$$

$$\frac{d(N_{\mathcal{I}_{jk}}(t))}{dt} = N_{\mathcal{I}_{(j-1)k}}(t)R_{\mathcal{I}_{j-1}}(t) - N_{\mathcal{I}_{jk}}(t)R_{\mathcal{I}_j}(t), \forall j \in [1, s_d-1], k \in [1, g] \cup \infty \quad (12)$$

$$\frac{d(N_{\mathcal{I}_{s_d}}(t))}{dt} = \sum_{\forall k} N_{\mathcal{I}_{(s_d-1)k}}(t)R_{\mathcal{I}_{s_d-1}}(t) - N_{\mathcal{I}_{s_d}}(t)\gamma. \quad (13)$$

Eq.(10) characterizes the varying rate of $N_{\mathcal{I}_{0\infty}}(t)$ which is composed of two parts: 1) $N_{\mathcal{D}}(t)q$ nodes transit to this class from class $\mathcal{D}$ by generating a request for d, 2) $N_{\mathcal{I}_{0\infty}}(t)R_{\mathcal{I}_0}(t)$ nodes transit from the class to class $N_{\mathcal{I}_{1\infty}}(t)$ by retrieving a packet from its contacted node. Eq.(11) depicts the varying rate of $N_{\mathcal{I}_{0k}}(t)$. $N_{\mathcal{I}_{0k}}(t)R_{\mathcal{I}_0}(t)$ nodes transit from class $\mathcal{I}_{0k}$ to class $\mathcal{I}_{1k}$ by retrieving a packet from others. Eq.(12) shows the varying rate of $N_{\mathcal{I}_{jk}}(t)$ ($j \in [1, s_d-1], k \in [1, g] \cup \infty$), which also consists of two parts: 1) $N_{\mathcal{I}_{(j-1)k}}(t)R_{\mathcal{I}_{j-1}}(t)$ nodes join the class from class $\mathcal{I}_{(j-1)k}$, 2) $N_{\mathcal{I}_{jk}}(t)R_{\mathcal{I}_j}(t)$ nodes leave from the class to class $\mathcal{I}_{(j+1)k}$. Eq.(13) shows the varying rate of $N_{\mathcal{I}_{s_d}}(t)$, where the first term denotes the number of nodes that join the class from class $\mathcal{I}_{(s_d-1)k}$ ($k \in [1, g] \cup \infty$), and the second term denotes the number of nodes which delete the data and transit to class $\mathcal{I}_E$.

The passive transition would happen at the end of each time slot. At the end of each time slot, we update the number of nodes in each class as follows:

$$N_{\mathcal{I}_{jk}}(t) = N_{\mathcal{I}_{j(k+1)}}(t^-), \forall j \in [0, s_d-1], k \in [1, g-1] \quad (14)$$

$$N_{\mathcal{I}_{s_d}}(t) = N_{\mathcal{I}_{s_d}}(t^-) + \sum_{j=0}^{s_d-1} N_{\mathcal{I}_{j1}}(t^-). \quad (15)$$

The number of nodes in the rest of the classes which are not listed in Eq.(14) and (15) remains the same. Also, at the end of each time slot, the maximal delay of the existing downloading nodes would minus 1 (i.e., $g = g-1$). By combining the active and passive transition processes, the network state at any time t ($t > t_0$) can be derived. Thus, we can calculate $N_d(t)$, the number of nodes which has at least one packet of data d at time t, as $N_d(t) = \sum_{j=1}^{s_d} N_{\mathcal{I}_j}(t)$.

4.3.4 Calculation of $F_{T_{s_d}}(t)$

The derivative of $F_{T_r}(t)$ ($r \in [2, s_d]$) is represented as follows by using ODEs:

$$\frac{dF_{T_r}(t)}{d(t)} = \frac{Pr(T_r \le t+dt) - Pr(T_r \le t)}{dt} \quad (16)$$

$$= \frac{R(t)dt(Pr(T_{r-1} \le t) - Pr(T_r \le t))}{dt} \quad (17)$$

$$= R(t)(F_{T_{r-1}}(t) - F_{T_r}(t)), \forall r \in [2, s_d]. \quad (18)$$

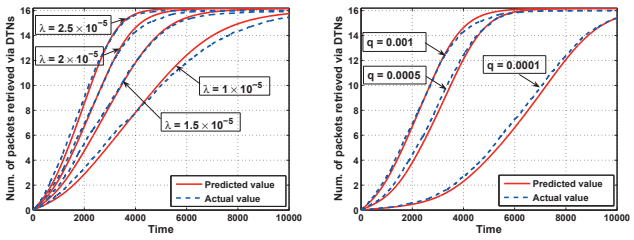
We ignore the probability that node i receives more than one packet during a very short time interval dt. Thus, the probability that T_r, the time for node i receives r packets, is between the range of $[t, t+dt]$ equals the probability that node i exactly receives $r-1$ packets before time t, and receives the rth packet during time t to $t+dt$. Thus, we derive Eq.(17) from Eq.(16). Similarly, we also derive $\frac{dF_{T_1}(t)}{d(t)} = R(t)(1-F_{T_1}(t))$. Therefore, given the initial values that $F_{T_r}(t_0) = 0$ ($r \in [1, s_d]$), $F_{T_{s_d}}(t)$ can be derived by solving the following ODEs:

$$\frac{dF_{T_1}(t)}{d(t)} = R(t)(1 - F_{T_1}(t)) \quad (19)$$

$$\frac{dF_{T_r}(t)}{d(t)} = R(t)(F_{T_{r-1}}(t) - F_{T_r}(t)), \forall r \in [2, s_d]. \quad (20)$$

4.3.5 Numerical Results

To verify the accuracy of our DTN based prediction model and analyze the impacts of the system parameters, we numerically solve the ODEs and compare the prediction results to the actual values derived from the Monte-Carlo simulations. In the simulations, we generate 300 nodes following i.i.d. Poisson contact process, and one data item with 16 packets and query rate $q = 0.001$. The same set of parameters is imported to the ODEs. We focus on the number of downloaded packets along time t on a specific node, and compare the results derived in the simulation with that from solving the ODEs. The results given by the simulation are averaged over 200 runs. Figure 5(a) shows the results with different contact rate λ. We can see that the prediction results are very close to the values given by the simulations, which verifies the accuracy of our prediction model. The larger the contact rate is, the earlier the node collects all 16 packets. We further compare the results when the query rate q varies, as shown in Figure 5(b). The prediction also achieves results close to that of the simulations. As the query rate increases, the node collects more packets from other peers as time passes. This implies that if a node requests a popular item, its offloading potential is large.



(a) With different contact rate λ (b) With different data query rate q

Fig. 5. Numerical results - DTN

4.4 Prediction of Offloading Potential: the WiFi case

Similar to the DTN case, by motivating mobile users to wait for some time, part of their cellular traffic may be redirected to WiFi networks when they contact some WiFi hotspots. In urban areas with wide deployment of WiFi networks, WiFi offloading can significantly mitigate the cellular network overload problem. In this section, we illustrate how to predict the potential of the users to offload their data traffic via WiFi networks.

4.4.1 Models

Most mobile users have some diurnal patterns (e.g., following the same commute path each day), and thus we can formulate their mobility based on the Markov model. Due to high node mobility, we also consider the contact duration limits in the WiFi case. That is, a large data item may not be completely downloaded when a node contacts a WiFi hotspot. To predict the offloading potential, both steady and transient behavior of node mobility should be considered. Therefore, we model node mobility by a Semi Markov Process, in which arbitrary distributed sojourn times are allowed. To avoid state space explosion, each Markov state represents a geographical area with a fixed size. The process of a user moving from a geographical area to another is modeled as a transition of Markov processes between two states. We assume that the average downlink bandwidth for each state is pre-calculated, and the average downlink data rate of state j is denoted as r_j.

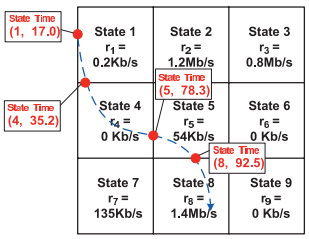


Fig. 6. Markov model

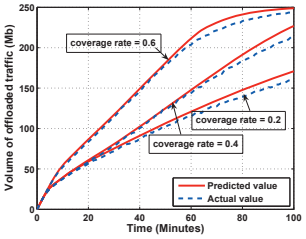


Fig. 7. Numerical results - WiFi

Figure 6 shows an example of the Markov model. There are nine states and each state represents a uniform sized geographical area. The value below each state shows its average downlink bandwidth. As shown in the figure, node i moves and will successively transits to state 1, 4, 5, and 8. For example, it transits to state 1 at time 17.0, and will leave state 1 and transit to state 4 at time 35.2.

We model the node mobility as a Markov renewal process $\{(X_n^i, T_n^i) : n \geq 0\}$ with a discrete state space $\mathbf{S} = \{1, \cdots, m\}$. $X_n^i \in \mathbf{S}$ is the state of node i's nth transition, and T_n^i is the time instance of this transition. We consider a first order Markov process and assume that the Markov process is *time homogeneous*; i.e., the distribution of these variables does not change over time.

4.4.2 The Main Idea of Prediction

Similar to Section 4.3, the objective of the prediction is to calculate the expected traffic $V_i^d(t)$ that can be offloaded to WiFi, if node i requests data item d and is willing to wait for delay t. We assume that node i's initial state is j; i.e., the node is in state j when it submits the bid. To simplify the presentation, we drop the superscript of X_n^i and T_n^i, and use node i as the default target node in the following analysis. The associated time homogeneous semi Markov kernel Q is defined as:

$$\begin{aligned} Q_{jk}(t) &= Pr(X_{n+1} = k, T_{n+1} - T_n \leq t | X_n = j) \\ &= p_{jk} S_{jk}(t). \end{aligned} \quad (21)$$

where $p_{jk} = Pr(X_{n+1} = k | X_n = j) = \lim_{t\to\infty} Q_{jk}(t)$ is the state transition probability from state j to k, and $\mathbf{P} = [p_{ij}]$ is the transition probability matrix of the embedded Markov chain. $S_{jk}(t)$ is the sojourn time distribution at state j when the next state is k; i.e., $S_{jk}(t)$ is the probability that the node will move from state j to k within sojourn time t, which can be derived as:

$$S_{jk}(t) = Pr(T_{n+1} - T_n \leq t | X_{n+1} = k, X_n = j). \quad (22)$$

Let $S_j(t) = Pr(T_{n+1} - T_n \leq t | X_n = j)$ denote the probability that the node will leave the current state j to another state within sojourn time t, which represents the probability distribution of the sojourn time in state j regardless of the next state. Then, $S_j(t) = \sum_{k=1}^{m} Q_{jk}(t)$.

We assume the time is discrete in our model, and define the homogeneous semi Markov process as $\mathbf{Z} = (Z_t, t \in \mathbb{N}^*)$, which describes the state of node at time t. The transition probability of $\mathbf{Z}$ is defined by $\phi_{jk}(t) = Pr(Z_t = k | Z_0 = j)$, which can be calculated as:

$$\begin{aligned} \phi_{jk}(t) &= Pr(Z_t = k | Z_0 = j) \\ &= (1 - S_j(t))\delta_{jk} + \sum_{l=1}^{m} \sum_{\tau=1}^{t} \dot{Q}_{jl}(\tau)\phi_{lk}(t-\tau). \end{aligned}$$

where δ_{jk} is the Kronecker delta function, which equals to 1 if and only if $j = k$; otherwise it is zero. $(1 - S_j(t))$ is the probability that the node stays at state j between time 0 and t without any transition. $\sum_{l=1}^{m} \sum_{\tau=1}^{t} \dot{Q}_{jl}(\tau)\phi_{lk}(t-\tau)$ represents the probability that the node transits at least once between time 0 to t, where $\dot{Q}_{jl}(\tau) = Q_{jl}(\tau) - Q_{jl}(\tau - 1)$ which is the probability that the node will transit from state j to state l at time t.

Given the transition probability of $\mathbf{Z}$, we can calculate $V_i^d(t)$, the expected traffic that can be transmitted to WiFi networks within time t when node i requests data d and moves in the network. The size of data d is denoted as s_d. We define $D_{jk}(t)$ as the expected traffic that can be transmitted within time t with the initial state j and the final state k. Then we obtain:

$$\begin{aligned} V_i^d(t) &= \sum_{k=1}^{m} D_{jk}(t)\phi_{jk}(t) = \sum_{k=1}^{m} (\min(tr_j, s_d)(1 - S_j(t))\delta_{jk} \\ &+ \sum_{l=1}^{m} \sum_{\tau=1}^{t} \min(\tau r_j + D_{lk}(t-\tau), s_d)\dot{Q}_{jl}(\tau)\phi_{lk}(t-\tau)). \end{aligned}$$

where $\min(tr_j, s_d)$ represents the traffic that can be offloaded if the node stays at state j with no transition before time t, and the traffic is bounded by the total amount of data requested by the node. $\min(\tau r_j + D_{lk}(t-\tau), s_d)$ is the traffic that can be offloaded if the node transits at least once before time t. Thus, we derive the offloading potential in WiFi case. Next, we describe how to calculate the transition probability matrix $\mathbf{P}$ and the sojourn time probability distribution $S_{jk}(t)$.

4.4.3 Calculation of $\mathbf{P}$ and $S_{jk}(t)$

To calculate $\mathbf{P}$ and $S_{jk}(t)$, node's mobility histories are needed. Mobile users can upload their mobility information to the network operator periodically through WiFi interfaces on their moibile phones or through wired networks by using their PCs.

$\mathbf{P}$ is the transition probability matrix of the embedded Markov chain. Each element $p_{jk} \in \mathbf{P}$ represents the probability that node i will transit from state j to k. We define p_{jk} as the observed transition frequency in the node mobility trace. Then, we obtain $p_{jk} = num_{jk}/num_j$, where num_{jk} is the number of transitions from state j to state k, and num_j is the number of transitions from state j without considering the next transition state.

$S_{jk}(t)$ is the sojourn time probability distribution at state j when the next transition state is k. Based on the node mobility history, $S_{jk}(t)$ can be estimated as:

$$\begin{aligned} S_{jk}(t) &= Pr(t_{jk} \leq t) \\ &= \frac{num_{jk}(t_{jk} \leq t)}{num_{jk}}. \end{aligned} \quad (23)$$

where t_{jk} is the sojourn time at state j when followed by state k, and $num_{jk}(t_{jk} \leq t)$ is the number of transitions from state j to state k with the sojourn time less than t.

In this way, the cellular network operator can derive the transition probability matrix and the sojourn time probability distribution for each node based on their uploaded mobility history.

4.4.4 Numerical Results

To verify the accuracy of our prediction model in the WiFi case, we numerically solve the Markov model and compare the predicted results and the actual results derived by performing the Monte-Carlo simulations. In the simulations, we generate a map which is divided into 100 (10×10) geographical grids with a node moving among these grids. We further generate a transition probability matrix and the corresponding power-law-like sojourn time probability distributions for the node. The node can choose four directions (up, down, left and right) to move at each transition state, and the probabilities for choosing the four directions are decided based on the transition probability matrix. The time that the node stays at the each state is set according to the corresponding sojourn time probability distribution. Some WiFi hotspots are randomly distributed on the map, and each geographical grid has an average WiFi downlink data rate. If there is no hotspot placed in the grid, the average downlink data rate is set to zero. We randomly generate data rates within the range of 50kbps and 1Mbps for each grid that contains the WiFi hotspots. In the simulation, the node is downloading a data item of 250Mb and we consider three WiFi coverage rates: 0.2, 0.4, and 0.6.

Figure 7 shows the comparison results, in which the red curves are the expected traffic that can be offloaded as predicted by our prediction model, and the blue dotted curves are the actual traffic that has been offloaded as derived by the Monte-Carlo simulations. As can be seen, the predicted results are very close to the actual results, which demonstrates the effectiveness of our WiFi based prediction model. The larger the WiFi coverage rate, the more traffic can be offloaded via WiFi. When the coverage rate is set to 0.6, almost all data can be downloaded via WiFi if the node is willing to wait for 100 minutes.

5 Performance Evaluation

In this section, we evaluate the performance of WinCoupon through trace-driven simulations for both DTN and WiFi cases. For each case, we first introduce the

simulation setup, and then evaluate the performance of Win-Coupon under various system parameters. In the evaluation, the following performance metrics are used:

• *Offloaded traffic*: The total amount of traffic that is actually offloaded.

• *Allocated coupon*: The total incentive cost spent by the network operator for offloading purpose.

• *Average downloading delay*: The average time a bidder spends to get the complete data after sending the request.

5.1 The DTN case

5.1.1 Simulation Setup

Our performance evaluation in the DTN case is conducted on the UCSD trace [29], which records the contact history of 275 HP Jornada PDAs carried by students over 77 days. Based on the trace, we generate 50 data items, and each contains 8 packets. The query rate q_d for each data d is generated following Zipf distribution, and the default skewness parameter w is set to 1.5. The delete rate γ_d for each data d is randomly generated within the range of $[1.0 \times 10^{-8}, 5.0 \times 10^{-8}]$ following the uniform distribution. When nodes request data, they can choose to attach bids with the request message based on their satisfaction function. In the simulations, we model the user satisfaction function as: $S(t) = p - at^b$, where p is the original data service charge, and we assume that all the data items have the same charge $p = 0.8$. a determines the scale of $S(t)$, and a smaller a results in higher delay tolerance. b determines the shape of $S(t)$. When $b > 1$, $b = 1$ and $b < 1$, $S(t)$ is a concave, linear and convex function respectively. In the simulations, we randomly generate parameters a and b within the range of $[0.04, 0.08]$ and $[0.8, 1.2]$ respectively following the uniform distribution for each node to each data unless specified differently. In the simulations, the trace for the first five days is used for warmup, during which some nodes can directly download data without bidding. The presented results are averaged over 10 runs.

The scale of the trace, in terms of the number of users and their contact frequencies, is rather small. This results in long auction rounds for the network operator to collect enough bids, as well as long downloading delay experienced by the users. In a university there would probably be a larger number of users, thus we further generate a large-scale trace by replicating the nodes in the original trace 10 times, which seems like a more reasonable network scale. The evaluation results on the large-scale trace are given in Section 5.1.5.

5.1.2 Simulation results – impact of number of bidders

First, we evaluate the performance of Win-Coupon for different number of bidders in the DTN case. The results are shown in Figures 8(a), 8(b), and 8(c). The number of bidders is set to 30, 60, and 90 by varying the length of the auction round. The reserve price is set to 0.2, i.e., the network operator is willing to pay at most 0.2 for offloading one traffic unit. As shown in Figure 8(a), the actual offloaded traffic by adopting Win-Coupon keeps close to the offloading target, until a certain upper bound reaches. The bound represents the upper limit of the traffic that can be offloaded by fully utilizing the delay tolerance and the offloading potential of the bidders given the reserve price. More traffic can be potentially offloaded if more bidders participate in the auction.

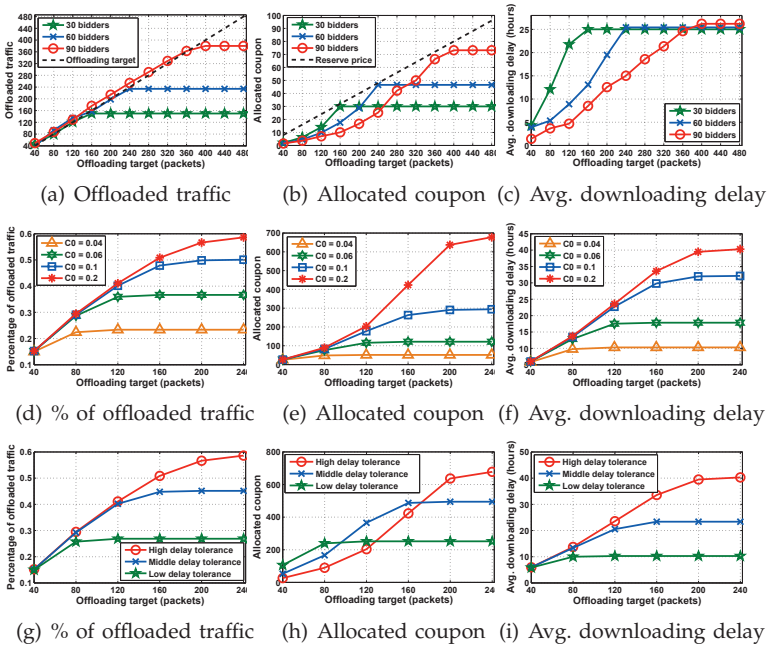
As can be seen from Figures 8(b) and 8(c), with the increase of offloading target, the allocated coupon and the average downloading delay increase accordingly, until reaching the offloaded traffic bound. The total value of coupon allocated by the network operator is strictly controlled by the reserve price which is marked as the black dotted line in Figure 8(b). With the same amount of traffic that is actually offloaded, the increase in the number of bidders results in less allocated coupon and shorter average delay. For example, when the number of bidders set to 30, 60 and 90, the network operator spends 6.3, 4.5, and 3.6 to actually offload 80 traffic units, and the average downloading delay is 12.1, 5.3, and 3.7 hours. The reason behind this phenomenon is that when more bidders participate in the auction, it is more likely to have more bidders with high delay tolerance or large offloading potential. To offload the same amount of traffic, the bidders with high delay tolerance request less coupon to compensate their satisfaction loss, and the bidders with large offloading potential need to wait for shorter time. Thus, the incentive cost and the delay decrease when more bidders participate in the auction.

5.1.3 Simulation results – impact of reserve price

To evaluate the impact of reserve price, we fix the duration of one auction round to be ten minutes, and set the reserve price to 0.04, 0.06, 0.1, and 0.2 respectively. We run the simulations for 20 consecutive auction rounds. The results are shown in Figures 8(d), 8(e), and 8(f). As can be seen, with the increase of reserve price, more traffic can be offloaded. This is because higher reserve price indicates larger willingness of the network operator to pay for offloading one unit of traffic, and then potentially motivates more users for offloading. When the reserve price is set to 0.2, almost 60% of the traffic can be offloaded as shown in Figure 8(d). However, higher reserve price results in higher incentive cost as shown in Figure 8(e). To balance this tradeoff, the network operator can set the reserve price appropriately according to its budget. Also, as shown in Figure 8(f), the average delay increases as the reserve price increases, since more users are selected as the winning bidder and motivated to wait.

5.1.4 Simulation results – impact of delay tolerance

To evaluate the impact of users' delay tolerance, we generate three scenarios with high, middle and low delay tolerance, by randomly setting the parameter a within the range of $[0.04, 0.08]$, $[0.08, 0.16]$, and $[0.16, 0.32]$. The reserve price is set to 0.2, and other settings remain the same as that used in the last subsection. The simulation

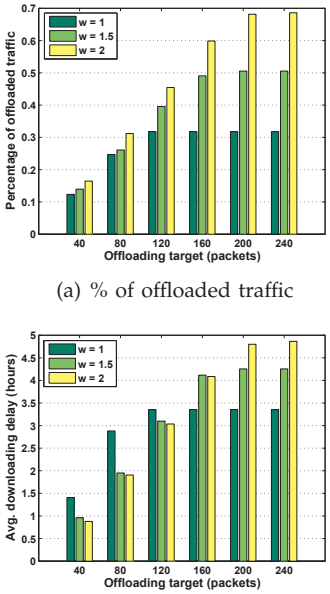


(a) Offloaded traffic (b) Allocated coupon (c) Avg. downloading delay

(d) % of offloaded traffic (e) Allocated coupon (f) Avg. downloading delay

(g) % of offloaded traffic (h) Allocated coupon (i) Avg. downloading delay

Fig. 8. Impact of bidder number, reserve price and delay tolerance - DTN



(a) % of offloaded traffic

(b) Avg. downloading delay

Fig. 9. Large-scale DTN trace

results are shown in Figures 8(g), 8(h), and 8(i). As can be seen from Figures 8(g) and 8(i), when the delay tolerance becomes larger, more traffic can be offloaded, and the average downloading delay increases. When the offloading target is set to 240, about 26.8%, 45.2%, and 58.6% of the traffic is offloaded in the scenario with low, middle and high delay tolerance respectively.

Figure 8(h) shows the value of allocated coupon with different offloading targets. When the offloading target is low, e.g., less than 80, as the users' delay tolerance gets higher, the incentive cost of the network operator drops, since less coupon is requested by the bidders. As the offloading target further increases, the amount of traffic that is actually offloaded remains almost the same in the low delay tolerance scenario. This is because the users in this scenario are not willing to wait longer and the traffic being offloaded is bounded. Then, the value of the allocated coupon in this scenario remains the same. However, in the scenarios with middle and high delay tolerance, as the offloading target increases more traffic can be offloaded by better exploiting users' delay tolerance, and then the allocated coupon increases.

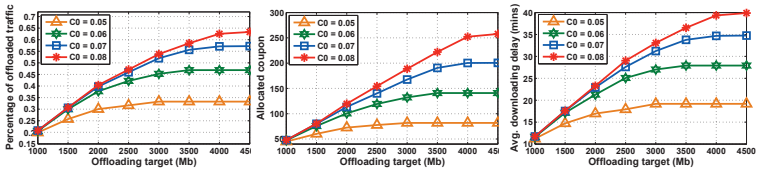
5.1.5 Simulation results – large-scale trace

In the above simulations, the duration of the auction round and the downloading delay are pretty long. This is due to the small scale of the UCSD trace. In reality, however, the network scale will be much larger, and then it will be easier for the network operator to collect enough bids and for the users to contact more peers within short time. Therefore, we further generate a large-scale trace including 2750 nodes by replicating the nodes in the UCSD trace 10 times. The contact rate between the nodes in the same copy remains the same as in the original trace, and the contact rate between the nodes in different copies is set to the average aggregated contact rate derived in the original trace. The duration of an auction round is set to only one minute. Figures 9(a) and 9(b) show the evaluation results when the skewness parameter w of the data query distribution is set to 1, 1.5, and 2. As can be seen, the larger the w is, the more traffic can be offloaded. With large w, more queries are for the popular data, and then it is easier for the requester to retrieve data from other contacted peers. When the offloading target reaches 240, almost 70% of the traffic can be offloaded in the case of $w = 2$. Also, as shown in Figure 9(b) when the offloading target is relatively small, larger w results in shorter delay. This is also due to the fact that the skewer data query distribution benefits more for the cellular traffic offloading. When the offloading target continues to increase, smaller w results in shorter delay, since the offloading target exceeds the offloading potential and many users directly download data via cellular network. More importantly, we can see that the average delay decreases significantly and becomes more reasonable for practical use in the large-scale scenario.

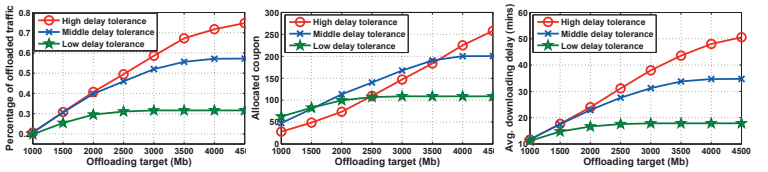
5.2 The WiFi case

5.2.1 Simulation Setup

To evaluate the performance of Win-Coupon in the WiFi case, we use the UMass DieselNet trace [30], which includes the mobility histories of 32 buses. In the trace, each bus is equipped with a GPS device, and periodically records its GPS location. To apply our prediction model, the map is divided into 10×15 uniform-sized geographical grids. Based on the mobility information provided by the trace, we further add synthetic WiFi information. We

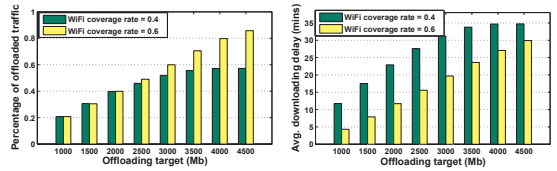


(a) % of Offloaded traffic (b) Allocated coupon (c) Avg. downloading delay

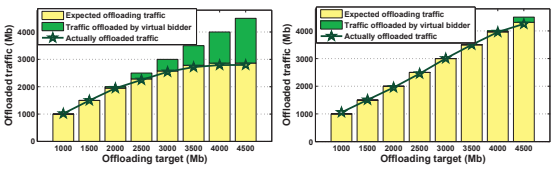


(d) % of offloaded traffic (e) Allocated coupon (f) Avg. downloading delay

Fig. 10. Impact of reserve price and delay tolerance - WiFi



(a) % of offloaded traffic (b) Avg. downloading delay



(c) Coverage rate = 0.4 (d) Coverage rate = 0.6

Fig. 11. Impact of WiFi coverage rates

assume that some WiFi hotspots are distributed on the map. We preset a WiFi coverage rate, which represents the ratio of the number of grids with some WiFi hotspots to the total number of grids. The downlink data rate for those grids with WiFi hotspots are randomly generated within the range of 50Kbps and 1Mbps.

To derive the transition probability matrix and the corresponding sojourn time probability distributions for each node, we take two-week traces as the training data. We pick up one day trace (11-06-2007) which has relatively high network density to perform Win-Coupon. The first auction round begins at 8:30 AM and the auction is performed for 10 consecutive rounds with the interval of one hour. Since the total number of nodes in the trace is quite limited, we assume that each node will participate in the auction to increase the number of participants. The size of data requested by nodes are randomly generated within the range of 100Mb and 500Mb.

Similar to the DTN case, we also define the user satisfaction function as $S(t) = p - at^b$ to model user delay tolerance. We randomly generate parameters a and b within the range of $[0.2, 0.3]$ and $[0.8, 1.2]$ respectively following the uniform distribution for each node to each data unless specified differently. The presented results are averaged over 10 runs.

5.2.2 Simulation results – impact of reserve price

To evaluate the impact of reserve price, we set the reserve price to 0.05, 0.06, 0.07, and 0.08, where setting it to 0.05 means that the network operator is willing to pay at most 0.05 for offloading 1Mb traffic. The results are shown in Figures 10(a), 10(b), and 10(c). Similar to DTN-based results shown in Figures 8(d), 8(e), and 8(f), larger reserve price motivates more users to wait longer. Then, more traffic can be offloaded at the expense of longer average delay and higher incentive cost. When the reserve price is set to 0.08, almost 65% traffic can be offloaded, and the average delay is about 40 minutes.

5.2.3 Simulation results – impact of delay tolerance

We further evaluate the impact of users' delay tolerance. Three scenarios with high, middle and low delay tolerance are generated by randomly setting the parameter α in the satisfaction function within the range of $[0.1, 0.2]$, $[0.2, 0.3]$, and $[0.3, 0.4]$ respectively. The reserve price is set to 0.07. The evaluation results are shown in Figures 10(d), 10(e), and 10(f). As can be seen, similar trend is captured in the WiFi case compared to the DTN case. Higher delay tolerance results in better offloading performance at the expense of longer downloading delay. To offload the same amount of traffic, less coupon is allocated in the high delay tolerance scenario. For example, to offload 20% traffic, 62.6, 47.8, and 28.1 coupon is allocated in the low, middle and high delay tolerance scenarios, respectively. In the high delay tolerance scenario, almost 75% of the traffic can be offloaded and the average downloading delay is only about 50 minutes.

5.2.4 Simulation results – impact of WiFi coverage rate

We set the WiFi coverage rate to 0.4 and 0.6 respectively to evaluate the impact of WiFi availability on the performance of Win-Coupon. The evaluation results are shown in Figure 11. Figures 11(a) and 11(b) show the percentage of offloaded traffic and the average delay respectively when offloading target increases from 1000Mb to 4500Mb. When the offloading target is set to be low, e.g., less than 2000Mb, different WiFi coverage rate results in similar percentage of offloaded traffic. This is because the relatively low offloading target can be easily achieved in both network scenarios. However, to offload the same amount of traffic, the average delay is much shorter when the WiFi coverage rate is higher. When the WiFi coverage rate is 0.6, almost 85% of the traffic can be offloaded, and the average delay is about 30 minutes.

Figures 11(c) and 11(d) show the comparisons of the expected offloaded traffic predicted by our prediction model and the actual. The yellow parts shown in the bars represent the expected traffic to be offloaded based on our prediction model and the lines drawn in the figure

denote the traffic that has been actually offloaded. As can be seen, the expected results are close to the actual results in both network scenarios. The green parts shown in the bars represent the volume of traffic that is expected to be offloaded by the virtual bidder. As explained in Section 4.2.4, the virtual bidder is added to ensure the network operator gaining non-negative profit. In other words, if the actual bidders have low delay tolerance or small offloading potential, the network operator would not trade with them and ask them to directly download data via cellular network (letting the virtual bidders win the game), even if the offloading target cannot be achieved. As can be seen when the WiFi coverage rate reaches 0.6, the offloading target can be almost achieved, due to the large offloading potential of the bidders.

6 Discussions

In this paper, we mainly focused on the downloading scenario since the majority of cellular traffic is on the downlink [31]. We also separate WiFi and DTN when discussing Win-Coupon design. Actually, our framework is very general, and can be extended to fit many other scenarios. Win-Coupon consists of two parts: auction based incentive mechanism and prediction. As long as the volume of offloaded traffic $V_i^d(t)$ can be predicted, the incentive mechanism can be adopted for coupon allocation and pricing under various scenarios such as uploading, downloading, DTN only, WiFi only, or hybrid of DTN and WiFi. The only difference under various scenarios is in the prediction part.

Uploading Scenario: In the WiFi case, since only the contact between the user and the WiFi hotspot affects offloading, the same prediction method can be used in the uploading scenario. In the DTN case, the current prediction method is based on the assumption that multiple users request the same popular items to share them through DTNs. Thus, it cannot be directly applied to the uploading scenario, since users generally upload different items. Hence, we need to design other offloading strategies and the prediction methods for the DTN case. For example, the uploading traffic can be offloaded by jointly using DTN and WiFi. Then, the node which generates data can transmit it via DTN to a contacted node which has large potential to have a WiFi connection in the near future, and upload the data through WiFi.

Hybrid Network Scenario: In the hybrid scenario which consider both DTN and WiFi, the user offloading potential should be re-calculated. A naive way is to simply treat them as two independently coexisting networks; i.e., mobile users can get data pieces from both networks during their waiting period. Then, the prediction is to find the "expected offloaded traffic" of DTN and WiFi separately using the current prediction methods, and sum them together. However, there are other better solutions. For example, instead of only downloading the data to satisfy their own demand, users can pro-actively download the popular data items from WiFi, then cache and share them with others via DTN. In this case, a joint prediction model is necessary. Also, more advanced caching mechanisms can be applied. Based on the techniques in [33][32], socially active nodes can use WiFi to pro-actively fetch and cache items with high popularity and low availability in their social communities and share them with other nodes via DTNs.

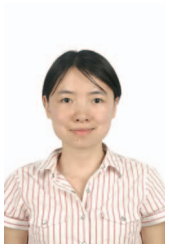
7 Conclusion

In this paper, we proposed a novel incentive framework for cellular traffic offloading. The basic idea is to motivate the mobile users with high delay tolerance and large offloading potential to offload their traffic to other intermittently connected networks such as DTN or WiFi hotspots. To capture the dynamic characteristics of users' delay tolerance, we design an incentive mechanism based on reverse auction. Our mechanism has been proved to guarantee truthfulness, individual rationality, and low computational complexity. Moreover, we design two accurate models to predict the offloading potential of the users for both DTN and WiFi cases. Extensive trace-driven simulations validate the efficiency and practical use of our incentive framework.

References

[1] M. Reardon, "Cisco Predicts Wireless-Data Explosion," [Online] Available: http://news.cnet.com/8301-30686_3-10449758-266.html.
[2] I. Trestian, S. Ranjan, A. Kuzmanovic, and A. Nucci, "Taming the Mobile Data Deluge with Drop Zones," *IEEE/ACM Trans. on Networking*, 2011.
[3] B. Han, P. Hui, V. Kumar, M. Marathe, J. Shao, and A. Srinivasan, "Mobile Data Offloading through Opportunistic Communications and Social Participation," *IEEE Trans. on Mobile Computing*, 2011.
[4] K. Lee, I. Rhee, J. Lee, S. Chong, and Y. Yi, "Mobile Data Offloading: How Much Can WiFi Deliver?" *Proc. of ACM CoNEXT*, 2010.
[5] A. Balasubramanian, R. Mahajan, and A. Venkataramani, "Augmenting Mobile 3G Using WiFi," *Proc. of ACM MOBISYS*, 2010.
[6] N. Ristanovic, J.-Y. L. Boudec, A. Chaintreau, and V. Erramilli, "Energy Efficient Offloading of 3G Networks," *Proc. of IEEE MASS*, 2011.
[7] J. Whitbeck, Y. Lopez, J. Leguay, V. Conan, and M. D. Amorim, "Relieving the Wireless Infrastructure: When Opportunistic Networks Meet Guaranteed Delays," *Proc. of IEEE WoWMoM*, 2011.
[8] C. Boldrini, M. Conti, and A. Passarella, "Modelling Data Dissemination in Opportunistic Neworks," *Proc. of ACM CHANTS*, 2008.
[9] P. Costa, C. Mascolo, M. Musolesi, and G. Picco, "Socially Aware Routing for Publish Subscribe in Delay-Tolerant Mobile Ad Hoc Networks," *IEEE JSAC*, vol. 26, no. 5, pp. 748-760, 2008.
[10] W. Gao, Q. Li, B. Zhao, and G. Cao, "Multicasting in delay tolerant networks: a social network perspective," *Proc. of ACM MOBIHOC*, 2009.
[11] A. J. Nicholson and B. D. Noble, "Breadcrumbs: Forecasting Mobile Connectivity," *Proc. of ACM MOBICOM*, 2008.
[12] M. R. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely, "Energy-Delay Tradeoffs in Smartphone Applications," *Proc. of ACM MOBISYS*, 2010.
[13] P. Cramton, "Spectrum Auctions,", *Handbook of Telecommunications Economics*, pp. 605-639, 2002.
[14] P. Xu, S. Wang, and X. Li, "SALSA: Strategyproof Online Spectrum Admissions for Wireless Networks," *IEEE Trans. on Computers*, pp. 1691-1702, 2010.
[15] X. Zhou, S. Gandhi, S. Suri, and H. Zheng, "Ebay in the Sky: Strategy-Proof wireless spectrum auctions," *Proc. of ACM MOBICOM*, 2008.

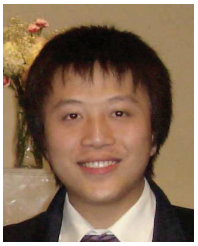
[16] L. Anderegg and S. Eidenbenz, "Ad hoc-VCG: A Truthful and Cost-Efficient Routing protocol for Mobile Ad hoc Networks with Selfish Agents," *Proc. of ACM MOBICOM*, 2003.
[17] W. Wang, X. Li, and Y. Wang, "Truthful Multicast in Selfish Wireless Networks," *Proc. of ACM MOBICOM*, 2004.
[18] X. Zhuo, W. Gao, G. Cao, and Y. Dai, "Win-Coupon: An Incentive Framework for 3G Traffic Offloading," *Proc. of IEEE ICNP*, 2011.
[19] D. P. Bertsekas, D. A. Castanon, and H. Tsaknakis, "Reverse Auction and the Solution of Inequality Constraned Assignment Problems," *SIAM Journal on Optimization*, pp. 268-299, 1993.
[20] W. Vickrey, "Counterspeculation, Auction and Competitive Sealed Tenders," *Journal of Finance*, pp. 8-37, 1961.
[21] E. Clarke, "Multipart pricing of public goods," *Public Choice*, pp. 17-33, 1971.
[22] T. Groves, "Incentives in Teams," *Econometrica*, pp. 617-631, 1973.
[23] N. Nisan and A. Ronen, "Algorithmic Mechanism Design," *Games and Economic Behavior*, pp. 166-196, 2001.
[24] A. Archer, C. Papadimitriou, K. Talwar, and E. Tardos, "An Approximate Truthful Mechanism for Combinatorial Auctions with Single Parameter Agents," *Internet Mathematics*, vol. 1, no. 2, pp. 129-150, 2004.
[25] Y. Lin, B. Li, and B. Liang, "Stochastic Analysis of Network Coding in Epidemic Routing," *IEEE JSAC*, vol. 26, no. 5, pp. 794-808, 2008.
[26] P. Chou, Y. Wu, and K. Jain, "Practical Network Coding," *Proc. of Annual Allerton Conf. on Comm., Control, and Comput.*, 2003.
[27] S. Deb, M. Medard, and C. Choute, "Algebraic Gossip: A Network Coding Approach to Optimal Multiple Rumor Mongering," *IEEE Trans. on Information Theory*, 2006.
[28] C. Zhang, Y. Fang, and X. Zhu, "Throughput-Delay Tradeoffs in Large-Scale MANETs with Network Coding," *Proc. of IEEE INFOCOM*, 2009.
[29] M. McNett and G. Voelker, "Access and mobility of Wireless PDA users," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 9, no. 2, pp. 40-55, 2005.
[30] "CRAWDAD metadata: umass/diesel," [Online] Available: http://crawdad.cs.dartmouth.edu/meta.php?name=umass/diesel.
[31] H. Falaki, D. Lymberopoulos, R. Mahajan, S. Kandula, and D. Estrin, "A First Look at Traffic on Smartphones," *Proc. of ACM IMC*, 2010.
[32] X. Zhuo, Q. Li, W. Gao, G. Cao, and Y. Dai, "Contact Duration Aware Data Replication in Delay Tolerant Networks," *Proc. of IEEE ICNP*, 2011.
[33] X. Zhuo, Q. Li, G. Cao, Y. Dai, B. Szymanski, and T. L. Porta, "Social-Based Cooperative Caching in DTNs: A Contact Duration Aware Approach," *Proc. of IEEE MASS*, 2011.



Xuejun Zhuo received the BE degree in computer science from Huazhong University of Science and Technology in 2007 and the PhD degree in computer science from Tsinghua University in 2012. She is currently a staff researcher in IBM China Research Lab. Her research interests include opportunistic mobile network, mobile social network, cloud computing, and network security.

Wei Gao received the BE degree in electrical engineering from the University of Science and Technology in 2005 and the PhD degree in computer science from the Pennsylvania State University in 2012. He is currently an assistant professor in the Department of Electrical Engineering and Computer Science at the University of Tennessee, Knoxville. His research interests include wireless and mobile network systems, mobile social networks, cyber-physical systems, and pervasive and mobile computing. He is a member of the IEEE.

Guohong Cao received the BS degree in computer science from Xian Jiaotong University and received the PhD degree in computer science from the Ohio State University in 1999. Since then, he has been with the Department of Computer Science and Engineering at the Pennsylvania State University, where he is currently a Professor. He has published more than 150 papers in the areas of wireless networks, wireless security, vehicular networks, wireless sensor networks, cache management, and distributed fault tolerant computing. He has served on the editorial board of IEEE Transactions on Mobile Computing, IEEE Transactions on Wireless Communications, IEEE Transactions on Vehicular Technology, and has served on the organizing and technical program committees of many conferences, including the TPC Chair/Co-Chair of IEEE SRDS'2009, MASS'2010, and INFOCOM'2013. He was a recipient of the NSF CAREER award in 2001. He is a Fellow of the IEEE.



Sha Hua received the BE degree from Huazhong University of Science and Technology in 2007 and the PhD degree in Electrical Engineering from Polytechnic Institute of New York University in 2012. He is currently a senior engineer in Qualcomm. His research interests include the optimization and protocol design in wireless video transmission, next-generation cellular system, cognitive radio networks and network economics.