Throughput-Efficient Channel Allocation in Multi-Channel Cognitive Vehicular Networks

You Han*, Eylem Ekici*, Haris Kremo†, and Onur Altintas †
*Department of Electrical and Computer Engineering
The Ohio State University, Columbus, Ohio 43210
Email: han.639@osu.edu, ekici@ece.osu.edu
†Toyota Info Technology Center Co., Ltd, Tokyo, Japan
Email: {hkremo, onur}@jp.toyota-itc.com

***Abstract*—Recent studies show that the Dedicated Short Range Communication (DSRC) band allocated to vehicular networks is insufficient to carry the wireless traffic load generated by emerging applications for vehicular systems. A promising bandwidth expansion possibility presents itself through the release of large TV band spectra by FCC for cognitive access. One of the primary challenges of the so-called TV White Space (TVWS) access in vehicular networks is the design of efficient channel allocation mechanisms in face of high vehicular mobility and spatial-temporal variations of TVWS. In this paper, we address the channel allocation problem for multi-channel cognitive vehicular networks with the objective of system-wide throughput maximization. We show that the problem is a NP-hard combinatorial optimization problem, to which we present two solution approaches. We first propose a probabilistic polynomial-time $(1 - 1/e)$-approximation algorithm based on linear programming. Next, we prove that our objective function can be written as a submodular set function, based on which we develop a deterministic polynomial-time constant-factor approximation algorithm with a more favorable time complexity. Finally, we show the efficacy of our algorithms through numerical examples.**

I. Introduction

Recent studies show that the DSRC band allocated to vehicular networks is unlikely to meet the bandwidth demands of emerging wireless applications in vehicular networks [1] [2]. One of the challenges of using TVWS in vehicular networks is the design of efficient channel allocation mechanisms that can cope with high vehicular mobility and spatial-temporal variations of TVWS. However, IEEE 802.11p, which is designed for vehicular networks based on CSMA/CA principle, is not suitable for Cognitive Radio Networks (CRNs) for the lack of cognitive radio capability.

The channel allocation problem in CRNs has been studied extensively in the literature. A number of works model the primary user (PU) activity with discrete time Markov chains and channel allocation is made in each time slot. In [3], the spectrum sensing and access process of secondary users (SUs) is modeled as a Partially Observable Markov Decision Process (POMDP). Based on this POMDP model, the authors derive optimal and sub-optimal strategies for SUs to decide which channel to sense and access to maximize system throughput. Similarly, the availability of primary channels is modeled as an alternating renewal process in [4]. However, these models only apply to static CRNs. Due to the high vehicular mobility and spatial-temporal variations of TVWS, new and low time complexity channel allocation mechanisms are needed that utilize both statistical and instantaneous channel availability information.

A joint rate control and channel allocation problem is studied in [5] to maximize long-term system throughput of single-hop CRNs. This work has been extended to address the same issue for multi-hop CRNs to maximize the throughput of SUs while stabilizing the CRNs in [6]. However, these methods focus on maximizing long-term system throughput and can fail to meet vehicles' short-term throughput maximization requirements. Moreover, they assume that each SU is associated with a uniform and fixed priority while the priority of a vehicle varies with the priority of its packets. In vehicular networks, short-term throughput maximization requirement is very important due to the real time requirements of associated applications. Therefore, algorithms for long-term utility maximization are not well suited for vehicular networks.

In this paper, we propose a throughput-efficient channel allocation framework for multi-channel cognitive vehicular networks. In contrast to existing works, we study the channel allocation problem with the objective of maximizing vehicular short-term utility. We also consider high vehicular mobility, spatial-temporal variations of TVWS as well as collision probability constraints imposed by PUs. More specifically, we consider a TDMA model where available channels are allocated to vehicles at the beginning of each scheduling cycle aiming to cope with rapidly changing vehicle locations. The allocation is based on vehicles' packet priorities and packet sizes, expected remaining idle time of each TVWS channel as well as collision probability constraint of each channel.

The theoretical contributions of our work are threefold. First, we formulate the problem of short-term utility maximization via channel allocation for vehicular networks in TVWS and show that the problem is NP-hard by reducing it to the Generalized Assignment Problem. Second, we propose a probabilistic polynomial time $(1-1/e)$-approximation algo-

978-1-4799-3360-0/14/$31.00

978-14799-3360-0/14/$31.00

rithm based on linear programming. Third, we prove that our objective function can be written as a submodular set function, based on which we develop a deterministic polynomial-time constant-factor approximation algorithm with a more favorable time complexity.

The rest of the paper is organized as follows. The system model and problem formulation are described in Section II. In Section III, we propose our method based on linear programming and analyze its performance and time complexity. Then, our objective function is proved to be a submodular set function and a deterministic polynomial time algorithm is developed in Section IV. We present simulation results in Section V and conclude our work in Section VI.

II. System Model and Problem Formulation

We consider a cognitive vehicular network consisting of N vehicles. We assume that there are M available TVWS channels in the network. In vehicular networks, packets are classified into four Access Categories with decreasing priority: $AC[0] \cdots AC[3]$. Therefore, in our model, we associate each priority class $AC[i]$ with a weight A_i subject to $A_i > A_j, \forall i < j$. In our model, we assume that Roadside Channel Monitors (RCMs) constantly sense and estimate usage patterns of all available TVWS channels. Time is partitioned into equal scheduling cycles with length T and RCMs are also responsible for assigning channels to vehicles at the beginning of every scheduling cycle. The value of T is determined by properties and requirements of a specific vehicular network.

The TVWS occupancy is modeled through a random variable t_j, which is defined as the residual time until the return of a PU to channel j. We assume that both the probability density function (PDF) $f_j(t_j)$ and cumulative distribution function (CDF) $F_j(t_j)$ of t_j are known to RCMs. Since exact behaviors of PUs are unknown to RCMs, vehicular transmissions scheduled on a channel can conflict with PU transmissions with non-zero probabilities. However, in our model, we assume that PUs can tolerate a certain collision probability, i.e., the collision probability caused by scheduled vehicles on a channel j must not exceed a predetermined bound γ_j, which is a system parameter determined by the PU network. With this collision probability constraint, RCMs can compute the maximum allowed scheduling time period on each channel which is denoted by T_j^r. Then, the collision probability P_{coll} can be computed as follows:

$$P_{coll} = Pr\{t_j \le T_j^r\} = \int_0^{T_j^r} f_j(t_j)dt_j = F_j(T_j^r) \le \gamma_j, \tag{1}$$

and thus the maximum T_j^r can be computed by solving Equation (1). Therefore, the maximum allowed scheduling duration on channel j is $\min\{T_j^r, T\}$.

In our model, we assume that the transmissions of a vehicle before a PU returns are successful and all remaining packets are thought to be lost (one possible reason is that high transmit power of PUs starves vehicular transmitters). The vehicle would try to transmit the lost packets in following cycles. Let L_i be the number of packets vehicle i tends to transmit in the current cycle. Note that L_i is not necessarily the real packet queue length of vehicle i. For example, L_i can be the amount of emergent safety-related packets vehicle i wants to transmit in the current cycle. Instead of tracking the packet queue length of each vehicle, we assume that L_i value is included in the vehicle's requesting messages to the RCM. Then, the RCM determines required transmission time of vehicle i on channel j by the following formula:

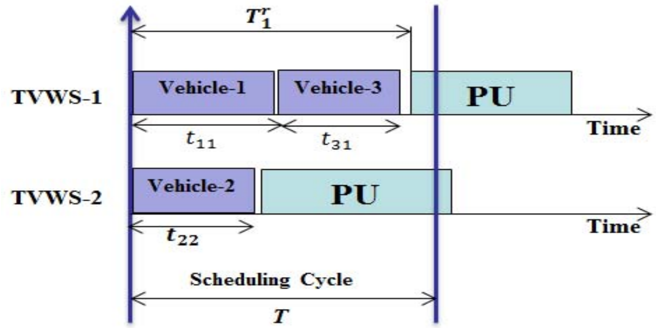

Figure 1. An example of the channel allocation model

$$t_{ij} = \min\{L_i/R_j, T_j^r, T\},$$

where R_j denotes the transmission rate of vehicles on channel j. Let x_{ij} be our channel assignment variable, i.e., $x_{ij} = 1$ if vehicle i is scheduled on channel j and $x_{ij} = 0$ otherwise. We use U_{ij} to denote the expected weighted throughput, i.e., utility of vehicle i on channel j in the current scheduling cycle:

$$\begin{aligned} U_{ij} &= \frac{1}{T}A_iR_j \int_0^{\infty} (t_{ij}x_{ij} \cdot 1_{\{t_j \ge \sum_{k=1}^{i} t_{kj}x_{kj}\}} \\ &+ (t_j - \sum_{k=1}^{i-1} t_{kj}x_{kj}) \cdot 1_{\{\sum_{k=1}^{i-1} t_{kj}x_{kj} \le t_j \le \sum_{k=1}^{i} t_{kj}x_{kj}\}}) f_j(t_j)dt_j \\ &= \frac{1}{T}A_iR_j \left[t_{ij}x_{ij} - \int_{\sum_{k=1}^{i-1} t_{kj}x_{kj}}^{\sum_{k=1}^{i} t_{kj}x_{kj}} F_j(t_j)dt_j \right]. \end{aligned} \tag{2}$$

Note that we let $\sum_{k=1}^{0} t_{kj}x_{kj} = 0$ when $i = 1$, and thus $U_{1j} = \frac{1}{T}A_1R_j \left[t_{1j}x_{1j} - \int_0^{t_{1j}x_{1j}} F_j(t_j)dt_j\right]$. An example of our channel allocation model is shown in Figure 1.

Our goal is to maximize the total weighted throughput of all vehicles, i.e., $\sum_{i,j} U_{ij}$, by scheduling disjoint set of vehicles on available TVWS channels. Since more than one vehicle can be scheduled on a channel, the expected throughput of a vehicle is affected by vehicles scheduled ahead of it according to Equation (2). Therefore, the transmission order of vehicles on a channel will affect the total weighted throughput of these vehicles. Lemma 1 states that priority-ordered transmission maximizes the total weighted throughput on a channel.

Lemma 1: Given a set of vehicles to be scheduled on a channel, higher priority vehicles must be scheduled earlier than lower priority vehicles in order to maximize the total weighted throughput, while the transmission order of vehicles of the same priority does not affect the total weighted throughput.

Proof. First, we prove that for any two vehicles v_1, v_2 scheduled on the same channel, scheduling the vehicle with higher priority ahead of the other vehicle is more rewarding in terms of total weighted throughput of the two vehicles. Let A_1 and A_2 be their priority weights with $A_1 > A_2$ and t_1, t_2 be their scheduled transmission durations. Suppose time duration $(0, t_0)$ has been assigned to other vehicles, therefore the first transmitter of the two vehicles starts transmitting at t_0. Let U_{1-2} be their total utility when scheduling v_1 ahead of v_2 and U_{2-1} be the total utility for the reverse order. Let U^1_{1j} and U^1_{2j} be the expected throughput of the two vehicles when v_1 is scheduled ahead of v_2, and U^2_{1j} and U^2_{2j} be the expected throughput of the two vehicles when v_2 is scheduled ahead of v_1. According to Equation (2), we compute U_{1-2} and U_{2-1} in Equation (3) and (4), respectively:

$$\begin{aligned} U_{1-2} &\triangleq U^1_{1j} + U^1_{2j} \\ &= \frac{1}{T} R_j (A_1 t_1 + A_2 t_2 - A_1 \int_{t_0}^{t_0+t_1} F_j(t_j) dt_j \\ &- A_2 \int_{t_0+t_1}^{t_0+t_1+t_2} F_j(t_j) dt_j)), \end{aligned} \tag{3}$$

$$\begin{aligned} U_{2-1} &\triangleq U^2_{1j} + U^2_{2j} \\ &= \frac{1}{T} R_j (A_1 t_1 + A_2 t_2 - A_2 \int_{t_0}^{t_0+t_2} F_j(t_j) dt_j \\ &- A_1 \int_{t_0+t_2}^{t_0+t_2+t_1} F_j(t_j) dt_j)). \end{aligned} \tag{4}$$

Then we compare the values of U_{1-2} and U_{2-1} by subtracting U_{2-1} from U_{1-2} and get Equation (5):

$$\begin{aligned} U_{1-2} - U_{2-1} = \frac{1}{T} R_j (A_1 - A_2) (\int_{t_0+t_2}^{t_0+t_2+t_1} F_j(t_j) dt_j \\ - \int_{t_0}^{t_0+t_1} F_j(t_j) dt_j). \end{aligned} \tag{5}$$

Since $A_1 > A_2$, we now compare the two integral parts which have the same integral function and same interval length t_1. Because $F_j(t_j)$ is a CDF, it is non-decreasing. Since the starting point of the first integral's interval is larger, i.e. $t_0 + t_2 > t_0$, we have $\int_{t_0+t_2}^{t_0+t_2+t_1} F_j(t_j) dt_j \geq \int_{t_0}^{t_0+t_1} F_j(t_j) dt_j$. Therefore, we have $U_{1-2} \geq U_{2-1}$, which proves the first part of Lemma 1. Furthermore, if the two vehicles have the same priority, i.e., $A_1 = A_2$, we can immediately conclude from Equation (5) that $U_{1-2} = U_{2-1}$, which proves the second part of Lemma 1. ∎

In our model, vehicles are ordered and renamed as $\{v_1, v_2, \cdots v_N\}$ such that $A_1 \geq A_2 \cdots \geq A_N$ and $L_i \geq L_j, \forall i \leq j$ when $A_i = A_j$. It means that high priority vehicles are scheduled ahead of low priority vehicles and vehicles with the same priority are ordered with the decreasing order of their L_i values.

Now our channel allocation problem can be formulated as follows:

$$\begin{aligned} \max_{x_{ij} \in \{0,1\}} & \sum_{i=1}^{N} \sum_{j=1}^{M} \frac{1}{T} A_i R_j (t_{ij} x_{ij} - \int_{\sum_{k=1}^{i-1} t_{kj} x_{kj}}^{\sum_{k=1}^{i} t_{kj} x_{kj}} F_j(t_j) dt_j) \\ s.t. & \sum_{i=1}^{N} t_{ij} x_{ij} \leq c_j, \forall j \in \{1, 2 \cdots M\} \\ & \sum_{j=1}^{M} x_{ij} \leq 1, \forall i \in \{1, 2 \cdots N\} \\ & t_{ij} = \min\{L_i / R_j, T^r_j, T\}; c_j = \min\{T^r_j, T\}. \end{aligned} \tag{6}$$

In this work, we schedule vehicles based on their priorities and L_i values and not on the long-term evolution of their queues. Therefore, our approach is suitable for highly mobile vehicular networks where message lifetimes are short and messages are expunged from queues if they are not transmitted by their deadlines. Moreover, our approach only requires A_i and L_i values for each of the N vehicles in each scheduling cycle, which can be easily obtained even under high mobility scenarios.

Theorem 1: The channel allocation Problem (6) is NP-hard.

Proof. This theorem is proved by reducing Problem (6) to the Generalized Assignment Problem (GAP). GAP is to assign different sets of items to bins subject to bin capacity constraints to maximize the overall profit, where both the size s_{ij} and profit p_{ij} are constant metrics. GAP has been proved to be NP-hard [7]. In Problem (6), we consider the case where $Pr\{t_j < T\} = 0$. This is a simplified version of our problem where PUs always return at least T time after the beginning of a scheduling cycle, i.e., the TVWS channels are always available. In this case, Problem (6) is reduced to the following problem:

$$\begin{aligned} \max_{x_{ij} \in \{0,1\}} & \sum_{i=1}^{N} \sum_{j=1}^{M} \frac{1}{T} A_i R_j t_{ij} x_{ij} \\ s.t. & \sum_{i=1}^{N} t_{ij} x_{ij} \leq T, \forall j \in \{1, 2 \cdots M\} \\ & \sum_{j=1}^{M} x_{ij} \leq 1, \forall i \in \{1, 2 \cdots N\} \\ & t_{ij} = \min\{L_i / R_j, T\}. \end{aligned} \tag{7}$$

This reduced problem is a GAP, where T is the capacity of all bins, t_{ij} is the size of item i in bin j, and $\frac{1}{T} A_i R_j t_{ij}$ is the profit of assigning item i to bin j. This mapping proves that Problem (6) is NP-hard. ∎

III. Solution 1: Linear Programming Algorithm

The general form of Problem (6) is much harder than GAP since the profit values p_{ij} in GAP are fixed. In contrast, the utility function of each vehicle in Problem (6) is coupled with other vehicles. Here, "coupling" means that the expected throughput of a vehicle is affected by the vehicles scheduled ahead of it. For a given set of vehicles assigned to a channel, however, the utility of each vehicle on the channel (i.e., the "profit" in GAP) is fixed due to the throughput maximization transmission ordering rule (see Lemma 1).

With this observation, our linear programming based (LP) algorithm works as follows. We first formulate an equivalent integer programming (IP) problem for the initial channel allocation problem, i.e., choosing exactly one set of vehicles S_j for each channel j such that the total utility on all channels is maximized. Let $X_j(S_j) \in \{0,1\}$ denote the decision variable indicating the assigned set of vehicles to channel j. By relaxing this constraint to be $X_j(S_j) \in [0,1]$, we obtain a LP problem. However, this LP problem has exponential number of variables due to exponential number of possible sets of vehicles to be assigned to a channel. Even though this LP problem cannot be solved in polynomial time, dealing with its dual problem helps to reduce the number of its variables with no performance loss. Although the dual problem has exponential number of constraints, it is proved in [8] that such problems can be solved in polynomial time by using ellipsoid algorithm associated with a separation oracle (to be defined later). When solving the dual problem with the ellipsoid algorithm iteratively, one is guaranteed to obtain at most one feasible variable for the primal LP problem in each iteration. Since the ellipsoid algorithm is proved to terminate in polynomial iterations, one can obtain at most polynomial number of variables. These variables are sufficient to solve the primal problem with no performance loss [9]. In this way, the primal LP problem becomes a new LP problem with polynomial number of variables, which can be solved in strongly polynomial time [10]. After solving the LP problem, we obtain a fractional solution $\{X_j(S_j)\}$. Then, we propose a rounding algorithm to round the fractional solution into an integer solution with the approximation factor greater than or equal to $(1-1/e)$. Since the optimal solution of the LP problem is an upper bound of the optimal solution of the IP problem, we are guaranteed to achieve at least $(1-1/e)$ of the IP optimum. The detailed procedure of the LP algorithm is as follows.

A. Formulating the LP Problem

Let $G_j(S_j)$ be the utility of assigning the set of vehicles S_j to channel j and I_j be the set of all feasible assignments of vehicles to channel j. Problem (6) can be reformulated as a LP problem as follows:

$$\begin{aligned} \max_{X_j(S_j)\in[0,1]} & \sum_{j=1}^{M}\sum_{S_j\in I_j} G_j(S_j)X_j(S_j) \\ s.t. & \sum_{j=1}^{M}\sum_{S_j\in I_j: i\in S_j} X_j(S_j) \le 1, \forall i \in \{1,2,\cdots N\} \\ & \sum_{S_j\in I_j} X_j(S_j) = 1, \forall j \in \{1,2,\cdots M\}, \end{aligned} \tag{8}$$

where $G_j(S_j)$ can be computed as follows:

$$G_j(S_j) = \sum_{i\in S_j} \frac{1}{T} A_i R_j(t_{ij}x_{ij} - \int_{\sum_{k=1}^{i-1} t_{kj}x_{kj}}^{\sum_{k=1}^{i} t_{kj}x_{kj}} F_j(t_j)dt_j). \tag{9}$$

The first constraint means that each vehicle can be scheduled on at most one channel and the second constraint means only one set of vehicles can be scheduled on a channel.

B. Reducing the Number of Primal Variables Using the Dual Problem

The Lagrangian equation and dual problem of Problem (8) are as follows:

$$\begin{aligned} L(\boldsymbol{X},\boldsymbol{\mu},\boldsymbol{\lambda}) = & \sum_{j=1}^{M}\sum_{S_j\in I_j} G_j(S_j)X_j(S_j) \\ & + \sum_{i=1}^{N}\mu_i(1-\sum_{j=1}^{M}\sum_{S_j\in I_j: i\in S_j} X_j(S_j)) + \sum_{j=1}^{M}\lambda_j(1-\sum_{S_j\in I_j} X_j(S_j)), \end{aligned} \tag{10}$$

$$\begin{aligned} \min_{\boldsymbol{\mu},\boldsymbol{\lambda}} & \sum_{i=1}^{N}\mu_i + \sum_{j=1}^{M}\lambda_j \\ s.t. & \sum_{i\in S_j}\mu_i + \lambda_j \ge G_j(S_j), \forall i, S_j \in I_j \\ & \mu_i \ge 0, \forall i \in \{1,2,\cdots N\}. \end{aligned} \tag{11}$$

Even though the dual Problem (11) has exponential number of constraints, it is guaranteed to be solved in polynomial time by using the ellipsoid algorithm. Before we solve this LP problem, we give some definitions and theorems on the ellipsoid algorithm. Here, we only introduce the general case of the ellipsoid algorithm in solving existence problems, i.e., whether there exists an $\boldsymbol{x}$ in a convex set defined by $P \in \{\boldsymbol{x}|A\boldsymbol{x} \le \boldsymbol{b}\}$ where $A \in R^{m\times n}, \boldsymbol{b} \in R^m$. For the optimization problem $\{\max \boldsymbol{c}^T\boldsymbol{x}|A\boldsymbol{x} \le \boldsymbol{b}\}$, the idea is very similar to the existence problem and a detailed procedure of the ellipsoid algorithm can be found in [8].

The ellipsoid algorithm works as follows to solve the existence problem. It starts with a big ellipsoid E that is guaranteed to contain P. It then checks whether the center of the ellipsoid $\boldsymbol{a}$ is in P. If it is, the algorithm terminates. Otherwise, it finds an inequality $A_i\boldsymbol{x} \le b_i$ violated by the center, where A_i is the i-th row of matrix A and b_i is the i-th element of $\boldsymbol{b}$. The ellipsoid is split into two parts by the hyperplane defined by $A_i\boldsymbol{x} \le A_i\boldsymbol{a}$. By choosing the feasible part, the new set becomes $E\bigcap\{\boldsymbol{x}|A_i\boldsymbol{x} \le A_i\boldsymbol{a}\}$. The procedure iterates in polynomial time until a feasible $\boldsymbol{x}$ is found or it determines that there is no such $\boldsymbol{x} \in P$.

Definition (Separation Oracle [8]): A separation oracle for a linear programming problem with n variables is an algorithm, such that for a given solution $\boldsymbol{x} \in R^n$, the algorithm is able to decide whether $\boldsymbol{x}$ is a feasible solution of the linear programming problem, and finds a constraint that is violated by $\boldsymbol{x}$ if it is not feasible.

Theorem 2 [8]: For the linear programming problem $\{\max \boldsymbol{c}^T \boldsymbol{x} | A\boldsymbol{x} \leq \boldsymbol{b}\}$ where $A \in R^{m\times n}$ and $\boldsymbol{b} \in R^m, \boldsymbol{c} \in R^n$, let Q be the maximal bit-encoding length of any value in A and $\boldsymbol{b}$, and S be the maximal bit-encoding length of any value in $\boldsymbol{c}$, then given a separation oracle of constraints $A\boldsymbol{x} \leq \boldsymbol{b}$ running in polynomial time, the ellipsoid method can optimize the the linear programming problem in time polynomial in n, Q and S.

Proof. A detailed proof is given in [8]. ■

According to Theorem 2, we can solve Problem (11) once we are able to find a polynomial time separation oracle of the constraints, i.e., given a set of solutions $(\lambda_j, \boldsymbol{\mu})$, decide whether these exists $S_j \in I_j$ such that $\sum_{i\in S_j} \mu_i + \lambda_j \geq G_j(S_j)$ and return the violated constraint if the solution is not feasible. Note that U_{ij} is the utility of vehicle i on channel j such that $G_j(S_j) = \sum_{i\in S_j} U_{ij}$, the constraint can be written as follows:

$$\lambda_j \geq \sum_{i\in S_j} (U_{ij} - \mu_i), \forall j \in \{1, 2, \cdots M\}. \tag{12}$$

The separation oracle now becomes equivalent to finding a feasible set S_j such that $\sum_{i\in S_j}(U_{ij} - \mu_i)$ is maximized for a given $\boldsymbol{\mu}$. This problem is equivalent to a knapsack problem: assigning a set of items into a knapsack to maximize the overall profit subject to a capacity constraint. Each item has a size s_i and profit p_i, and the bin has a capacity of W.

There are three general algorithms to solve the knapsack problem in polynomial time. The first one is an exact pseudo polynomial time algorithm which is applicable to knapsack problems with integer item size and bin capacity [11]. Another $(1/2)$-approximation algorithm works by sorting items according to their profit-to-size ratio p_i/s_i [11]. The third one is a β-approximation algorithm which is used in [12]. The algorithm works by scaling down and rounding the profit p_i into small integers and further using the pseudo polynomial algorithm mentioned at the beginning of this paragraph to solve the reduced knapsack problem. Both the second and third algorithms require the profits of each item to be fixed.

Note that the profit functions in our problem are coupled and thus the second and third algorithms cannot be used. Therefore, the first pseudo polynomial time algorithm becomes our only choice. It can be shown that the algorithm applies to knapsack problems with coupled profit functions. A detailed description of the algorithm is given in [11], and thus we only show why the algorithm applies to our problem.

The pseudo polynomial time algorithm is a dynamic programming algorithm and the core idea is to solve the sub-problems recursively to finally solve the initial knapsack problem. Let N and W be defined as the total number of items and the knapsack capacity of a knapsack problem, respectively. Let $V[i, w]$ be the maximum profit achieved by choosing a subset of items with at most i elements, $1 \leq i \leq N$, where the sum of their size is smaller than or equal to w, $0 \leq w \leq W$. The sub-problem is to determine the values of $V[i, w]$ recursively and in the last step determine $V[N, W]$. The values of $V[i, w]$ can be computed using the following lemma.

Lemma 2: For $1 \leq i \leq N$ and $0 \leq w \leq W$

$$V[i, w] = \max\{V[i-1, w], p_i + V[i-1, w-s_i]\} \tag{13}$$

Proof. A detailed proof can be found in [11]. ■

The values $V[i, w]$ can be obtained with time complexity $O(NW)$. The optimal value of the initial knapsack problem is $V[N, W]$ and its corresponding set can be found by tracking back the procedure of computing $V[N, W]$ with time complexity $O(NW)$. Therefore, the total time complexity of this algorithm is also $O(NW)$.

Next we show that the algorithm can be used to solve our knapsack problem with coupled profit functions. The key observation is that, although the profit function of each vehicle is coupled with other vehicles, the profit of a vehicle becomes fixed when the set of vehicles scheduled ahead of it is known. It is because their transmission order is determined by Lemma 1. In Lemma 2, $V[i-1, w-s_i]$ denotes the maximum sum profit achieved by choosing vehicles from $\{v_1, v_2 \cdots v_{i-1}\}$ with total size at most $w - s_i$. Let S_{i-1} be the optimal set attaining $V[i-1, w-s_i]$. Thus S_{i-1} must be a subset of $\{v_1, v_2 \cdots v_{i-1}\}$ and must be scheduled ahead of v_i. Since S_{i-1} is known in the current iteration, the profit of of v_i becomes fixed. Therefore, we can further compute $V[i, w]$ using Lemma 2. By using Lemma 2 for $O(NW)$ iterations, we are guaranteed to solve our knapsack problem [11].

One issue of using the dynamic programming algorithm as our separation oracle is that t_{ij} and c_j in Problem (6) are not necessarily integers. This problem can be solved by requiring that the system is time slotted and and thus each vehicle can only request integer number of slots in each scheduling cycle. Slotted time approach is widely used in many scheduling problems in the literature such as [5] and [6]. Another issue with the algorithm is that it is pseudo polynomial instead of polynomial and its time complexity is $O(NW)$. However, this algorithm can be bounded to be polynomial in our case. First, the algorithm is fully polynomial in N, i.e., the number of vehicles. Second, although the algorithm is pseudo polynomial in W (i.e., c_j in our case), we can bound the value of W. Let W_j be the number of maximum scheduling slots on channel j. Given the length of a time slot t_0, we can bound W_j as follows:

$$W_j = \frac{c_j}{t_0} = \frac{\min\{T, T_j^r\}}{t_0} \leq \frac{T}{t_0}. \tag{14}$$

With this bound, we can make sure our separation oracle is a polynomial-time oracle (see Equation (12)). According to Theorem 2, we can solve Problem (11) in polynomial time.

C. Solving the Primal LP Problem

By using the ellipsoid algorithm, we can reduce the number of variables in the primal LP without performance loss [9]. Since the ellipsoid algorithm always runs for polynomial number of iterations and only one set S_j is returned in each iteration, the total number of sets for the primal Problem

(8) is also polynomial. Therefore, Problem (8) becomes a typical linear programming problem with polynomial number of variables and constrains, and thus can be solved much more easily in polynomial time in N.

D. Rounding the Fractional Solution

Since we can obtain a fractional solution $\{X_j(S_j)\}$ by solving Problem (8), we continue to round it to an integer solution with certain performance guarantees. The rounding method works as follows. First, for a given solution to the LP problem, we schedule one set of vehicles S_j to its corresponding channel j with probability $P_j(S_j)$ which is computed as follows:

$$P_j(S_j) = \frac{X_j(S_j)}{\sum_{S_j \in I_j} X_j(S_j)}. \tag{15}$$

It is possible that some vehicles are scheduled on more than one channel since they may be contained in multiple sets. In this case, let g_{ij} be the expected throughput of vehicle i given it is scheduled on channel j which is computed in Equation (17). If a vehicle is scheduled on more than one channel, it is assigned to the channel with the maximum g_{ij} value. Before computing g_{ij}, we define $f_{ij}(S_j)$ as the throughput of v_i on channel j when it is in the set S_j. The utility of v_i is affected by the vehicles scheduled ahead of it. However, when the set of vehicles ahead of it is known, its utility is also fixed and can be computed in $O(N)$ as follows:

$$f_{ij}(S_j) = \frac{1}{T} A_i R_j (t_{ij} x_{ij} - \int_{\sum_{k=1}^{i-1} t_{kj} x_{kj}}^{\sum_{k=1}^{i} t_{kj} x_{kj}} F_j(t_j) dt_j). \tag{16}$$

Given v_i is scheduled on channel j, its expected throughput can be computed as follows:

$$g_{ij} = \sum_{S_j \in I_j : i \in S_j} f_{ij}(S_j) \frac{X_j(S_j)}{\sum_{S_j \in I_j : i \in S_j} X_j(S_j)}. \tag{17}$$

Since the number of sets obtained by the LP problem is always polynomial in both the number of vehicles and number of channels, and the time complexity of computing $f_{ij}(S_j)$ for a given S_j is $O(N)$, the time complexity of computing g_{ij} is polynomial, as well.

Before we proceed to prove the performance guarantee of our rounding algorithm, we recall the following lemma:

Lemma 3: For any $Y_j \geq 0, l > 1, \sum_j Y_j \leq 1$ and $g_{i1} \geq g_{i2} \geq \cdots \geq g_{il} \geq 0$, the following inequality holds:

$$\begin{aligned} & Y_1 g_{i1} + (1 - Y_1) Y_2 g_{i2} + \cdots (\prod_{j=1}^{l-1} (1 - Y_j) Y_l g_{il}) \\ & \geq (1 - (1 - \frac{1}{l})^l) \sum_{j=1}^{l} g_{ij} Y_j. \end{aligned} \tag{18}$$

Proof. The proof can be found in [12]. ■

Define OPT_{LP} as the maximum total weighted throughput obtained by solving Problem (8), and we have the following theorem.

Theorem 3: The total weighted throughput of our rounded solution is at least $(1 - 1/e) OPT_{LP}$.

Proof. For any vehicle v_i, let Y_j be the probability that it is scheduled on channel j computed as: $Y_j = \sum_{S_j \in I_j : i \in S_j} X_j(S_j)$.

Let l_i be the number of channels including v_i. Then order and rename all the channels including v_i in the non-increasing order of g_{ij} such that $g_{i1} \geq g_{i2} \geq \cdots \geq g_{il_i}$. According to our rounding method, vehicle i always chooses the channel with the maximum g_{ij} value, hence the final throughput of vehicle i is g_{i1} with probability Y_1. Similarly, the final throughput of vehicle i is g_{i2} with probability $(1 - Y_1) Y_2$. In this way, we can compute the expected throughput of vehicle i as follows:

$$\begin{aligned} E[g_{ij}] &= Y_1 g_{i1} + (1 - Y_1) Y_2 g_{i2} + \cdots (\prod_{j=1}^{l_i - 1} (1 - Y_j) Y_{l_i} g_{il_i}) \\ &\geq (1 - (1 - \frac{1}{l_i})^{l_i}) \sum_{j=1}^{l_i} g_{ij} Y_j \\ &= (1 - (1 - \frac{1}{l_i})^{l_i}) \sum_{j=1}^{l_i} \sum_{S_j \in I_j : i \in S_j} f_{ij}(S_j) X_j(S_j) \\ &\geq (1 - \frac{1}{e} + \frac{1}{32 l_i^2}) \sum_{j=1}^{l_i} \sum_{S_j \in I_j : i \in S_j} f_{ij}(S_j) X_j(S_j) \\ &\geq (1 - \frac{1}{e}) \sum_{j=1}^{l_i} \sum_{S_j \in I_j : i \in S_j} f_{ij}(S_j) X_j(S_j) \end{aligned} \tag{19}$$

The first inequality is due to Lemma 3 and the second inequality is proved in [12]. Note that the contribution of vehicle i in the objective function of Problem (8) is $\sum_{j=1}^{l_i} \sum_{S_j \in I_j : i \in S_j} f_{ij}(S_j) X_j(S_j)$. Therefore, each vehicle is guaranteed to achieve at least $(1 - 1/e)$ of the throughput achieved in the LP problem and thus the total throughput is also guaranteed to have an approximation ratio of $(1 - 1/e)$, which proves Theorem 3. ■

E. Time Complexity

The time complexity of the LP algorithm is largely determined by the ellipsoid algorithm. The ellipsoid algorithm requires $O(n^2 L)$ iterations [13], where $n = M + N$ is the dimension of our problem and $L = NT$ is the length of the input data in bits in our case. Each iteration requires $O(MN + 2(M+N)^2)$ operations [13]. Therefore, the number of variables in the primal LP problem is $O(NT(M+N)^4)$. As discussed earlier (see part B of this section, Equation (14)), the value of T can be bounded, and thus Problem (8) becomes an LP problem with $O(N(M+N)^4)$ number of variables.

IV. Solution 2: Submodular Set Function Based Algorithm

Although the LP algorithm can be guaranteed to run in polynomial time, the time complexity is still high for large problem instances. Thus, more time efficient algorithms are needed. Submodular set functions have been used to solve many NP-hard combinatorial optimization problems as well as resource allocation problems at low time complexity [14]. In this section, we first prove that our objective function of Problem (6) can be written as a non-decreasing submodular function. Then, we propose a deterministic constant-factor approximation algorithm based on the work in [15]. Below are some basic definitions and properties of submodular set functions from [16].

Definition 1 (Submodularity): Let S be a non-empty finite set and f be a mapping from the power set of S to non-negative real numbers. f is said to be submodular if it satisfies $f(A \cup \{v\}) - f(A) \geq f(B \cup \{v\}) - f(B)$ for all $v \in S \setminus B$ and all $A \subseteq B \subseteq S$.

Definition 2 (Monotonicity): Let S be a non-empty finite set and f be a mapping from the power set of S to non-negative real numbers. Then f is non-decreasing if it satisfies $f(A) \leq f(B)$ for all $A \subseteq B \subseteq S$.

Lemma 4: A positive linear combination of submodular functions is submodular.

Now we proceed to show the submodularity and monotonicity of our objective function by proving the following theorem.

Theorem 4: The objective function in Problem (6) can be written as a non-decreasing submodular set function.

Proof. Recall that S_j has been defined as the set of vehicles scheduled on channel j. We further define $f_j(S_j) \triangleq \sum_{i \in S_j} \frac{1}{T} A_i R_j (t_{ij} x_{ij} - \int_{\sum_{k=1}^{i-1} t_{kj} x_{kj}}^{\sum_{k=1}^{i} t_{kj} x_{kj}} F_j(t_j) dt_j)$ and write the objective function as follows:

$$G(\boldsymbol{x}) = \sum_{i=1}^{N} \sum_{j=1}^{M} \frac{1}{T} A_i R_j (t_{ij} x_{ij} - \int_{\sum_{k=1}^{i-1} t_{kj} x_{kj}}^{\sum_{k=1}^{i} t_{kj} x_{kj}} F_j(t_j) dt_j) = \sum_{j=1}^{M} f_j(S_j). \tag{20}$$

Since the objective function is a positive linear combination of $f_j(S_j)$, we only need to prove $f_j(S_j)$ is submodular for all j to prove the submodularity of the initial objective function according to Lemma 4. Similarly, if we prove $f_j(S_j)$ is non-decreasing for all j, then the initial objective function is also non-decreasing.

First, we prove $f_j(S_j)$ is non-decreasing. Suppose A and B to be two sets of vehicles scheduled on channel j with $A \subseteq B \subseteq S_j$. Set B can be further partitioned into two parts: A and $C = B \setminus A$. When scheduling vehicles in set B, assume vehicles in part A are scheduled ahead of vehicles in part C and let $f_j(C|A)$ be the overall weighted throughput of vehicles in C after A has been scheduled. Hence, according to Lemma 1, $f_j(B) \geq f_j(A) + f_j(C)$ because there may be some vehicles in C whose priorities are higher than some vehicles in A. Therefore, $f_j(B) \geq f_j(A) + f_j(C) \geq f_j(A)$, which proves the monotonicity of $f_j(S_j)$.

Now we continue to prove the submodularity of $f_j(S_j)$. Let A and B be two sets of vehicles scheduled on channel j with $A \subseteq B \subseteq S_j$. Both vehicles in A and B can be partitioned into four parts based on the priorities of their packets and higher priority vehicles are always scheduled ahead of lower priority vehicles. Let t_l^A and t_l^B be the total duration of packets with priority l in A and B respectively where $l \in \{0, 1, 2, 3\}$. Since $A \subseteq B$, it is easy to see that $t_l^A \leq t_l^B, \forall l \in \{0, 1, 2, 3\}$. Therefore, we can rewrite $f_j(A)$ and $f_j(B)$ as follows:

$$\begin{aligned} f_j(A) &= \sum_{i \in A} \frac{1}{T} A_i R_j (t_{ij} x_{ij} - \int_{\sum_{k=1}^{i-1} t_{kj} x_{kj}}^{\sum_{k=1}^{i} t_{kj} x_{kj}} F_j(t_j) dt_j) \\ &= \frac{1}{T} R_j \sum_{l=0}^{3} A_l (t_l^A - \int_{\sum_{k=0}^{l-1} t_k^A}^{\sum_{k=0}^{l} t_k^A} F_j(t_j) dt_j), \\ f_j(B) &= \frac{1}{T} R_j \sum_{l=0}^{3} A_l (t_l^B - \int_{\sum_{k=0}^{l-1} t_k^B}^{\sum_{k=0}^{l} t_k^B} F_j(t_j) dt_j). \end{aligned} \tag{21}$$

Now we schedule a new vehicle v on channel j whose duration and priority are t_v and l_v, respectively. First, suppose the new vehicle is added to set A. According to Lemma 1, we can always schedule v at the end of vehicles of priority l_v. Thus the remaining scheduled time interval $(\sum_0^{l_v} t_l^A, \sum_0^3 t_l^A)$ is shifted backwards by t_v and thus the new time interval becomes $(\sum_0^{l_v} t_l^A + t_v, \sum_0^3 t_l^A + t_v)$. We first consider the case where $t_v \leq t_{l_v+1}^A$ and we will show our theorem also holds when $t_v > t_{l_v+1}^A$. In the first case, the marginal utility of v consists $(4 - l_v)$ parts and in each part the corresponding interval is replaced with higher priority packets. Since there are no packets of priority "4", we let $A_4 = 0$. Therefore, we can evaluate the marginal utility of v for both A and B as follows:

$$\begin{aligned} & f_j(A \cup \{v\}) - f_j(A) \\ &= \frac{1}{T} R_j \sum_{l=l_v}^{3} (A_l - A_{l+1}) (t_v - \int_{\sum_{k=0}^{l} t_k^A}^{\sum_{k=0}^{l} t_k^A + t_v} F_j(t_j) dt_j), \\ & f_j(B \cup \{v\}) - f_j(B) \\ &= \frac{1}{T} R_j \sum_{l=l_v}^{3} (A_l - A_{l+1}) (t_v - \int_{\sum_{k=0}^{l} t_k^B}^{\sum_{k=0}^{l} t_k^B + t_v} F_j(t_j) dt_j). \end{aligned} \tag{22}$$

Hence, we can obtain the following result:

$$\begin{aligned} & [f_j(A \cup \{v\}) - f_j(A)] - [f_j(B \cup \{v\}) - f_j(B)] \\ &= \frac{1}{T} R_j \sum_{l=l_v}^{3} (A_l - A_{l+1}) (\int_{\sum_{k=0}^{l} t_k^B}^{\sum_{k=0}^{l} t_k^B + t_v} F_j(t_j) dt_j \\ &- \int_{\sum_{k=0}^{l} t_k^A}^{\sum_{k=0}^{l} t_k^A + t_v} F_j(t_j) dt_j)) \\ &\geq 0. \end{aligned} \tag{23}$$

The last inequality is due to the facts that (1) $A_i > A_j, \forall i < j$, (2) $t_l^A \leq t_l^B, \forall l \in \{0, 1, 2, 3\}$ and (3) the CDF $F_j(t_j)$ is non-decreasing. Inequality (23) proves the theorem when $t_v \leq t_{l_v+1}^A$. When $t_v > t_{l_v+1}^A$, it is easy to see from the equality in (20) that more parts of the initial scheduled time interval are replaced by higher priority packets, which further enlarges the difference between the two integral parts. The proof follows the same steps and details are omitted here due to space limitation. ∎

Our problem becomes the maximization of a monotone submodular set function under linear packing constraints. We first show that the algorithm proposed in [15] applies to our problem. In [15], the authors study the problem of maximizing a monotone submodular set function $f(S)$ subject to linear packing constraints $A\boldsymbol{x} \leq \boldsymbol{b}$, where $f : 2^{[n]} \to R, A \in [0,1]^{m\times n}, \boldsymbol{b} \in [1,\infty)^m$. Since we have proved the objective function of Problem (6) is a monotone submodular set function, we now proceed to show that the constraints of Problem (6) can be written as $A\boldsymbol{x} \leq \boldsymbol{b}$, where $A \in [0,1]^{m\times n}, \boldsymbol{b} \in [1,\infty)^m$. Recall the capacity constraints of the initial problem are $\sum_{i=1}^{N} t_{ij}x_{ij} \leq c_j, \forall j \in \{1,2\cdots M\}$. We define the new constraint matrix as follows: $A_{ij} = \frac{t_{ij}}{\max_k\{t_{kj}\}}, b_j = \frac{c_j}{\max_k\{t_{kj}\}}, \forall i,j$. According to the network model, we have $c_j = \min\{T, T_j^r\} \geq t_{ij} = \min\{T, T_j^r, L_i/R_j\}$ and thus $A \in [0,1]^{m\times n}, \boldsymbol{b} \in [1,\infty)^m$. By writing the set S as $\{x_{11}, x_{21}, \cdots x_{N1}, x_{12}, x_{22}, \cdots x_{N2}, x_{1M}, \cdots x_{NM}\}, \forall x_{ij} \in \{0,1\}$, we have transformed the assignment constraints $\sum_{j=1}^{M} x_{ij} \leq 1, \forall i \in \{1,2\cdots N\}$ into a set constraint. Therefore, Problem (6) becomes a standard problem of maximizing a monotone submodular set function subject to linear packing constraints as studied in [15].

The algorithm in [15] is based on a multiplicative update method and works as follows. The algorithm maintains a set of weight factors that are updated in multiplicative way. The factors are designed to indicate the extend to which each constraint is close to be violated under the current solution. The algorithm has only one loop and it extends the current solution with a non-selected element that minimizes a normalized sum of weights in each iteration. After the loop terminates, the final solution is returned as long as it is feasible, otherwise, either the last selected element or the resulting solution without the last solution is returned. The detailed algorithm is given in Algorithm 1.

Theorem 5: Algorithm 1 is a deterministic polynomial-time algorithm that achieves an approximation ratio of $1/(e \times M^{(1/P)} + 1)$ for maximizing a monotone submodular function subject to linear packing constraints, where $P = \min(b_j/A_{ij} : A_{ij} > 0)$. The algorithm attains the approximation factor $(1-4\varepsilon)(1-1/e)$ for any $\varepsilon > 0$, by using update factor $\lambda = e^{\varepsilon P}$ in the algorithm when $P \geq \max\{(lnM)/\varepsilon^2, 1/\varepsilon\}$.

Proof. A detailed proof is given in [15]. ∎

The time complexity of the submodular set function-based algorithm can be computed as follows. Algorithm 1 will terminate after at most N iterations since the algorithm selects at least one element from the N elements in each iteration. Moreover, in each iteration, there are $O(NM)$ operations since the algorithm computes the $\sum_{j=1}^{M} A_{ij}w_i/G_S(i)$ value of each element. Therefore, the time complexity of Algorithm 1 is $O(MN^2)$.

Algorithm 1 Submodular Set Function Method

Require: The total weighted throughput function $G(\boldsymbol{x}) = \sum_{j=1}^{M} f_j(S_j)$ and its marginal value function $G_S(i) = G(S \cup \{i\}) - G(S)$, knapsack constraints $A\boldsymbol{x} \leq \boldsymbol{b}$, an update factor $\lambda = e^P M$, where $P = \min\{b_i/A_{ij} : A_{ij} > 0\}$

$S \leftarrow \varnothing$
for $j = 1$ **to** M **do**
  $w_j = 1/b_j$
end for
while $\sum_{j=1}^{M} b_j w_j \leq \lambda$ and $|S| < N$ **do**
  Let $i \in [N] \setminus S$ be the element with minimal $\sum_{j=1}^{M} A_{ij}w_i/G_S(i)$
  $S \leftarrow S \cup \{i\}$
  for $j = 1$ **to** M **do**
    $w_j = w_j \lambda^{A_{ij}/b_j}$
  end for
end while
if $A\boldsymbol{x} \leq \boldsymbol{b}$ **then**
  STOP
else
  if $G(S \setminus \{i\}) \geq G(\{i\})$ **then**
    $S \leftarrow S \setminus \{i\}$
  else
    $S = \{i\}$
  end if
end if
return A subset S of $[N] = \{x_{11}, x_{21}, \cdots x_{N1}, x_{12}, x_{22}, \cdots x_{N2}, x_{1M}, x_{2M}, \cdots x_{NM}\}$

V. Numerical Results

We simulated a cognitive vehicular network with N vehicles and M TVWS channels, where $N \in \{10, 20\cdots 80\}$ and $M \in \{5, 15\}$. For each given pair of (N, M) values, we run the two channel allocation algorithms for 100 iterations, each iteration consisting of 100 scheduling cycles. We then compare the performance of the two algorithms in terms of vehicles' average utility per cycle.

Our network parameters are set as follows. The scheduling cycle T is five seconds and the duration of each time slot is set to 0.2 seconds in the LP algorithm. The transmission rate of all channels is set to $8Mbps$. The arrival rates of packets in four access categories follow from Poisson processes with different λ values $\{100, 150, 200, 150\}$ (packets per second) with 1280-byte packets. PU activities in each channel are modeled by the probability that the channel is unavailable at the beginning of each scheduling cycle. The probability is set to 0.1 for all channels. All 15 channels are classified into three groups with the same set of parameters. Below are parameters of TVWS channels in each group. Residual idle time of the channels are all set to follow from Gamma distributions with $k_j = 2, \forall j \in \{1,2,3,4,5\}$ and $\beta \in \{5, 10, 10, 15, 20\}$. The collision probability constraints γ_j of five channels are set to $\{0.03, 0.06, 0.06, 0.08, 0.06\}$. Priority weights of four access categories are set to $\{8, 4, 2, 1\}$. At each scheduling cycle, a

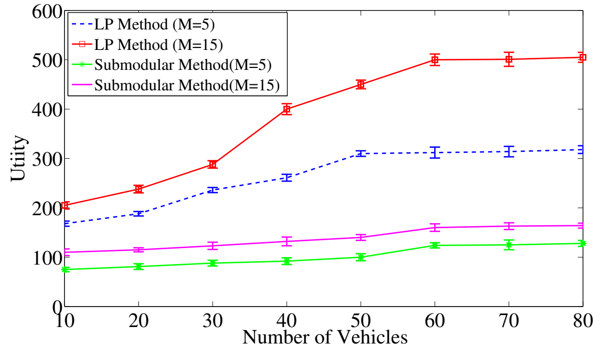

Figure 2. Comparison of LP and Submodular Methods in terms of Utility Per Cycle

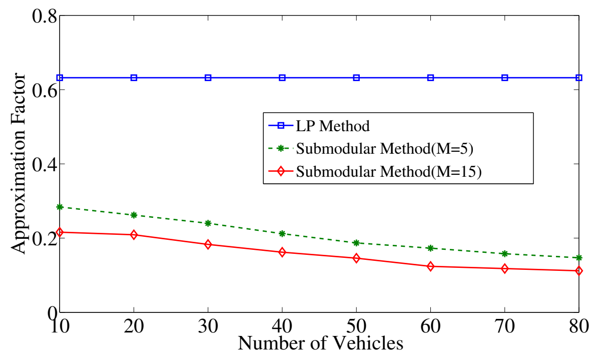

Figure 3. Comparison of LP and Submodular Methods in terms of Approximation Factor

vehicle transmits one of the four priority packets with equal probabilities. Each vehicle can only transmit packets of single priority in each scheduling cycle.

In Figure 2, the system utility is plotted as a function of the number of vehicles. Along with the average utility values, we compute their 95% confidence intervals. It can be seen that the LP method performs much better than the submodular method in terms of system utility. Although the submodular algorithm is a deterministic constant factor approximation method, its constant factor is significantly smaller than that of LP. In contrast, although LP method is a probabilistic method, its practical performance is still satisfying. In Figure 3, we compare the two algorithms in terms of their approximation factors. The approximation factor of LP method is fixed while that of submodular method is determined by system parameters P (see Theorem 5). Note that all these factors are lower bounds of the algorithms. It can be seen from Theorem 5 that the constant factor is an increasing function of P. In our system setting, the value $P = \min(b_j/A_{ij} : A_{ij} > 0)$ decreases with both the number of channels and vehicles, and thus the constant factor also decreases with the number of channels and vehicles as shown in Figure 3.

VI. Conclusion

In this paper, we studied the throughput-efficient channel allocation problem in cognitive vehicular networks in TVWS spectrum. We show that our problem is NP-hard and give two constant-factor approximation algorithms with polynomial running time. Our simulation results support our theoretical analysis. The design of algorithms with both good approximation factor and low time complexity is an important topic for our future work. Further research topics also include study of the channel allocation problem in multi-hop cognitive vehicular networks and design of distributed channel allocation algorithms.

References

[1] M. Hassan, H. Vu, and T. Sakurai, "Performance analysis of the ieee 802.11 mac protocol for dsrc safety applications," *Vehicular Technology, IEEE Transactions on*, vol. 60, no. 8, pp. 3882–3896, 2011.

[2] H. Kremo and O. Altintas, "On detecting spectrum opportunities for cognitive vehicular networks in the tv white space," *Journal of Signal Processing Systems*, pp. 1–12, 2013.

[3] Q. Zhao, L. Tong, and A. Swami, "Decentralized cognitive mac for dynamic spectrum access," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, 2005, pp. 224–232.

[4] H. Kim and K. Shin, "Efficient discovery of spectrum opportunities with mac-layer sensing in cognitive radio networks," *Mobile Computing, IEEE Transactions on*, vol. 7, no. 5, pp. 533–545, 2008.

[5] R. Urgaonkar and M. Neely, "Opportunistic scheduling with reliability guarantees in cognitive radio networks," in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, 2008, pp. 1301–1309.

[6] D. Xue and E. Ekici, "Guaranteed opportunistic scheduling in multi-hop cognitive radio networks," in *INFOCOM, 2011 Proceedings IEEE*, 2011, pp. 2984–2992.

[7] M. L. Fisher, R. Jaikumar, and L. N. Van Wassenhove, "A multiplier adjustment method for the generalized assignment problem," *Manage. Sci.*, vol. 32, no. 9, pp. 1095–1103, Sep. 1986.

[8] M. Grtschel, L. Lovsz, and A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization," *Combinatorica*, vol. 1, no. 2, pp. 169–197, 1981.

[9] D. Chakrabarty and G. Goel, "On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and gap," in *In Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science-Volume 00*, 2008, pp. 687–696.

[10] E. Tardos, "A strongly polynomial algorithm to solve combinatorial linear programs," *Oper. Res.*, vol. 34, no. 2, pp. 250–256, Mar. 1986.

[11] G. L. Nemhauser and L. A. Wolsey, *Integer and combinatorial optimization*. New York, NY, USA: Wiley-Interscience, 1988.

[12] L. Fleischer, M. X. Goemans, V. S. Mirrokni, and M. Sviridenko, "Tight approximation algorithms for maximum general assignment problems," in *Proc. of ACM-SIAM SODA*, 2006, pp. 611–620.

[13] D. Goldfarb and M. Todd, "Modifications and implementation of the ellipsoid algorithm for linear programming," *Mathematical Programming*, vol. 23, no. 1, pp. 1–19, 1982.

[14] N. Golrezaei, K. Shanmugam, A. Dimakis, A. Molisch, and G. Caire, "Femtocaching: Wireless video content delivery through distributed caching helpers," in *INFOCOM, 2012 Proceedings IEEE*, 2012, pp. 1107–1115.

[15] Y. Azar and I. Gamzu, "Efficient submodular function maximization under linear packing constraints," in *Proceedings of the 39th international colloquium conference on Automata, Languages, and Programming - Volume Part I*, ser. ICALP'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 38–50.

[16] G. Nemhauser, L. Wolsey, and M. Fisher, "An analysis of approximations for maximizing submodular set functionsi," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.