Optimal and Efficient Approximate Algorithms for New One-Dimensional RSU Deployment Problem

Zhenguo Gao, Danjie Chen, Shaobin Cai, Hsiao-Chun Wu, *Fellow, IEEE*

***Abstract*—Because of the excessive cost of Road Side Units (RSUs), optimal RSU deployment is of crucial importance to VANETs. RSU Deployment Problem (RDP) models in the literature usually lack of ability to describe curve-shaped roads with non-uniform statistics. We had proposed a more realistic model for the RDP problem. taking into consideration these features. However, our understanding to the RDP problem with the new model, even to the simpler one-dimensional RDP problem, is still weak. In this paper, we focus on the one-dimensional RDP problem with our new model. We firstly analyze the properties of optimal solutions of the RDP problem. Then, suspecting that the one-dimensional RDP problem with n RSUs of different radii is intractable, we propose two greedy-based algorithms (named as Greedy2P3 and Greedy2P3E) and show that Greedy2P3E's approximation ratio is at least $1-(\frac{n-1}{n})^2$. And next, for the one-dimensional RDP problem with n RSUs of identical radii, we found that it can be transformed to a problem being a subset of the well-known maximum coverage problem, which is NP-hard. By exploiting the properties of optimal solutions of the original problem, we propose an optimal algorithm based on greedy idea and dynamic programming, so it is abbreviated as OptGreDyn. At last, we proved that, if applied to the one-dimensional RDP problem with n RSUs of identical radii, the approximation ratios of Greedy2P3 and Greedy2P3E are at least 2/3. Furthermore, Greedy2P3's approximation ratio of 2/3 is tight when $n=3*i$, here i is any positive integer. Numerical simulations validate the correctness of the analyses results. Simulation results verify the optimality of OptGreDyn, meanwhile show that Greedy2P3 and Greedy2P3E usually return near-optimal solutions with more than 98% of optimal solutions, and they are preferable than existing approximate algorithms.**

***Index Terms*—Road side unit (RSU) deployment problem, optimal algorithm, approximate algorithm, approximation ratio, vehicular ad hoc networks(VANETs).**

I. Introduction

Due to the envisioned potential in improving road safety, traffic control as well as infotainment, Vehicular Ad Hoc Networks (VANETs) have attracted great interests in both industry and academia [1]–[3]. A VANET consists of On-Board Units (OBUs) and Roadside Units (RSUs). OBUs are installed on vehicles to provide wireless communication capability, and RSUs are deployed along roads. Acting as the network infrastructure for VANETs and providing internet access to OBUs, RSUs play a key role in improving service quality of VANETs [3]. However, being estimated that a simplistic RSU may cost $13000-$15000 [4], the excessive cost of densely deploying RSUs was identified as a major hinderance to make VANET service ubiquitously accessible [5], where the authors also showed that maximizing the profit of fixed RSUs is challenging.

The RSU deployment problem can be regarded as an extension of the node placement problem widely researched in the realm of wireless sensor networks [6]. Since that the vehicles in VANETs are usually restricted to move along roads, RSUs are expected to be deployed at road side, which is quite different from the case of wireless sensor networks, where nodes are placed with much fewer restrictions. Therefore, results on the node placement problem in wireless sensor networks are usually not suitable for the RSU deployment problem in VANETs.

In past years, many efforts have been devoted to the RSU deployment problem, where RSUs are usually assumed to be fixed at particular locations [2], [4], [5], [7]–[12]. These problem models in these works are different in many aspects, such as the factors and the restrictions considered, targeted scenarios, optimization objectives, and type of candidate positions. Some researches took into considerations of mobile RSUs, where they are used to facilitate message dissemination [13] or store-carry-forward mode message routing [14]. In [3], Kim *et al.* proposed a framework for the joint deployment of fixed RSUs, mobile RSUs on public vehicles and those on dedicated vehicles. However, optimal deployment is more crucial for fixed RSUs than for mobile RSUs since that repositioning fixed RSUs is more difficult.

By inspecting the road network models in existing works, we noticed that road shape and the heterogeneity of different road segments (e.g., different number of lanes, traffic statistics) along a single road are usually neglected. Hence, in [15], a more realistic model which considers these issues was proposed, and a genetic based algorithm was proposed for solving the RDP problem with the new problem model. Later we use phrase *the new RDP problem* to represent *the RDP problem with the new model*. Genetic algorithms can return approximate solutions, but they are relatively less powerful in deepening our understandings about the new problem. RDP problem can be sub-categorized into one-dimensional, two-dimensional, and three-dimensional problems according to the dimension of the underlying road-network model. In one-dimensional RDP problem, the road network model is indeed a straight line, and thus the model's ability to describe curve shaped roads is disabled, however its ability to describe roads with heterogeneous property is kept.

Z. G. Gao, and S. B. Cai are with the College of Computer Science and Technology, Huaqiao University, Xiamen, 361021, CHINA e-mail:gaozhenguo@hqu.edu.cn

D. J. Chen is with the College of Civil Engineering, Huaqiao University, Xiamen, 361021, CHINA.

H.-C. Wu is with the School of Electrical Engineering and Computer Science, Louisiana State University, LA, USA.

Manuscript received April XX, XXXX; revised December XX, XXXX.

In this paper, we focus on the new one-dimensional RDP problem for optimally deploying a certain number of fixed RSUs along a straight road line. To be specific, with a given number n, the problem is to find the positions of n fixed RSUs with fixed constant radii such that the total profit of deploying the RSUs is maximum. At the first sight, our new one-dimensional RDP problem here looks similar to the well known one-dimensional Geometric Set Cover (GSC) problem [16] and the Weighted Maximum Coverage (WMC) problem [17]. In the GSC problem, we are given a set of points on real line and some intervals, the task is to cover all points with minimum intervals. However, further inspection show that they are different: (1)what should be covered in the GSC problem are some discrete points on the real line, whereas they are regions with non-uniform weight densities in the RDP problem; (2)the objective of the GSC problem is to cover all given discrete points with minimum number of intervals, whereas the objective of the RDP problem is to obtain maximum coverage profits with a given number of intervals with given lengthes. The differences makes the two problems different. Although one-dimensional GSC problem can be solved in polynomial time using a simple greedy algorithm [16], we haven't found existing work that could solve the one-dimensional RDP problem. In the WMC problem, we are given a collection of elements with weights, several sets and a number n, the task is to select n sets such that total weights of the elements in these sets are maximized. The WMC problem is NP-hard, and cannot be approximated within $1-\frac{1}{e}+o(1)\approx 0.632$ under standard assumptions [17]. Although with these similar problems already researched, the new one-dimensional RDP problem is new, and our understanding even to the new one-dimensional RDP problem is still much lack. Hence, additional research efforts for the new RDP problem are required.

We describe our results on this new one-dimensional RDP problem in this paper. We firstly analyze the properties of optimal solutions of the RDP problem. Then, suspecting that the one-dimensional RDP problem with n RSUs of different radii is intractable, we propose two greedy-based algorithms (named as Greedy2P3 and Greedy2P3E) and show that Greedy2P3E's approximation ratio is least $1-(\frac{n-1}{n})^n$. Next, for the one-dimensional RDP problem with n RSUs of identical radii, we found that it can be transformed to the well-known NP-hard WMC problem [17]. By exploiting the properties of optimal solutions of the original problem, we propose an optimal algorithm based on greedy idea and dynamic programming, which is abbreviated as OptGreDyn. At last, we proved that, if applied to the one-dimensional RDP problem with n RSUs of identical radii, the approximation ratios of Greedy2P3 and Greedy2P3E are at least 2/3. Furthermore, Greedy2P3's approximation ratio of 2/3 is tight when $n{=}3{*}i$, here i is any positive integer. Numerical simulations validate our analysis results and show the efficiency of the proposed algorithms. Greedy2P3 has an approximation ratio of 2/3, and Greedy2P3E is an improved version of Greedy2P3, hence come their names.

Our main contributions in the paper are as follows

- For the new one-dimensional RDP problem with one RSU, we determine the set of candidate positions of the optimal solutions.
- For the new one-dimensional RDP problem with multiple RSUs of different coverage radius, we prove that there must be non-overlapped optimal solutions, and determine the set of candidate positions of the optimal solutions. These enable us to consider only non-overlapped solutions. We propose two greedy-based algorithms Greedy2P3 and Greedy2P3E and prove that Greedy2P3E's approximation ratio is at least $1-(\frac{n-1}{n})^n$.
- For the new one-dimensional RDP problem with multiple RSUs of identical coverage radius, we propose OptGreDyn and prove its optimality. We also proved that, when applied to one-dimensional RDP problem with multiple RSUs of identical coverage radius, approximation ratios of Greedy2P3 and Greedy2P3E are at least 2/3, and Greedy2P3's approximation ratio of 2/3 is tight when $n{=}3{*}i$.

The rest of the paper is organized as follows. Related work is discussed in Section II. The new RDP problem model as well as its road-network model and deployment benefit model is given in Section III. In Section IV, we prove some basic properties of the optimal solutions for one-dimensional RDP problem. In Section V, we propose Greedy2P3 and Greedy2P3E, and prove the approximation ratio of Greedy2P3E. In Section VI, we propose the optimal algorithm OptGreDyn and proves its optimality. In Section VII, we prove Greedy2P3's tight constant approximation ration of 2/3 when applied to RDP problem with multiple identical RSUs. In Section VIII, simulations results are provided and analyzed. A conclusion is given in Section IX. Detailed steps for the main result in VII are provided as appendix in Section A.

II. Related Work

Many schemes have been proposed for the RDP problem, where RSUs are assumed to be fixed at particular locations. These works are different in many aspects, such as the factors and restrictions taken into considerations(e.g, vehicle density [5],budget constraint [8], communication delay-bound [7]–[9]), targeted scenario(e.g., highway [9], [18],urban [4]), optimization objectives (e.g., network connectivity [10], traffic coverage [2], information dissemination [5]), type of candidate positions(road intersections, middle roads,regular grid points [4]), solution approaches(e.g., numerical optimization [4], greedy based algorithm [2], [3], clustering [19], [20], game theory [21], even heuristic schemes including genetic algorithm [11], harmony search algorithm [12]).

Based on the deployment objective, we categorize these existing works into 3 types, including network connectivity oriented schemes, traffic coverage oriented schemes, and information dissemination oriented schemes [4]. However, no works suitable for our new RDP problem have not been found, as far as we know.

- **Network Connectivity Oriented Schemes**

Network connectivity oriented schemes focus on improving network connectivity and reducing the probability and time period of network partition with limited number of RSUs.

In 2009, Zheng et al. [22] introduced the concept of α-coverage for measuring intermittent coverage for accessing the Internet through RSUs. Lee et al. [23] proposed a greedy based scheme for improving network connectivity and reducing the network disconnection interval, where every intersection of the road network is a candidate location for a RSU. In [10], the scheme tries to maximize the number of road intersections covered.

- **Traffic Coverage Oriented Schemes**

The traffic coverage oriented schemes aim at maximizing the number of vehicles that get in contact with the RSUs deployed in the considered region.

In 2012, Oscar et al. [24] formulated the RDP problem as a mixed-integer quadratic programming problem with the objective to maximize traffic volume for internet downloading services. In 2014, Silva et al. [25] proposed a scheme which utilizes partial mobility information, and then formulated the RDP problem as a probabilistic maximum coverage problem. That scheme tries to figure out locations maximizing the number of vehicles covered by at least one RSU. In 2015, Cheng et al. [2] proposed a geometry-based sparse coverage scheme taking into consideration the shape of road segments in urban scenarios. Although curve-shaped roads were considered there, the optimization objective there was still to maximize the regions covered, and the benefits are also calculated based on the regions covered instead of the roads covered.

- **Information Dissemination Oriented Schemes**

Information dissemination oriented schemes try to improve the information dissemination performance, such as reducing the end-to-end delay, increasing the packet throughput, or decreasing the packet loss ratio.

Sun et al. [26] formulated the RDP scheme as a set-covering problem, which considers the wireless communication range, the driving time and the extra overhead time, and then proposed a cost-efficient scheme. Abdrabou et al. [7] proposed an analytical framework by considering the randomness of the vehicle data traffic and the statistical variation of the intermittently connected communication channel. The framework can approximately estimate the minimum number of RSUs for covering a road segment with a probabilistic Vehicle-to-Infrastructure (V2I) transmission delay. They also inspected the maximum distance between RSUs with some given limitations. Aslam et al. [27] proposed two schemes for the RDP problem: Binary Integer Programming (BIP) algorithm and Balloon Expansion Heuristic (BEH) algorithm. The objective there was to minimize the average reporting time for a given number of RSUs and area overage. Their works consider factors including vehicle density, vehicle speed and the occurrence likelihood of an incident/event. Barrachina et al. [5] proposed a density-based RSU deployment scheme (D-RSU) specially designed for providing emergency alerting services with the lowest possible cost in case of accidents.

III. RSU Deployment Problem Model

A. RDP Problem Model

We review the RDP problem model focused in this paper briefly. For detailed introduction, please refer to [15].

Our new RDP problem model consists of two embedded models: a super graph based road-network model $G(V,E)$ and a RSU deployment profit model $B(G(V,E),p_{\rm s},n)$. The road network model is an abstract representation of a regional road map. In $G(V,E)$, V is the set of nodes representing road intersections and E is the set of curves representing road segments. A curve $e{\in}E$ is defined as a 4-tuple $e(v_h,v_t,f_c(\cdot),f_w(\cdot))$, which means that the curve starts at node v_h, ends at node v_t, its shape is expressed as function $f_c(\cdot)$ (it can be either in analytical form or in vectorial form), and $f_w(\cdot)$ is its profit density function defined along the curve. Notations $f_{e,c}(\cdot)$ and $f_{e,w}(\cdot)$ are used to explicitly represent curve e's shape function and profit density function, respectively. It should be noticed that the super graph concept here is different from the normal definition of a super graph.

The RSU deployment profit model $B(G(V,E),p_{\rm s},n)$ determines how the deployment profit is calculated for a solution p_s of deploying n RSUs in the road network $G(V,E)$. Here $p_{\rm s}{=}\{p_1,p_2,\cdots,p_n\}$ represents the solution to the RDP problem with n RSUs, where the i-th RSU is deployed at position p_i, which is the abbreviation of $p_i(x_i,y_i)$. Here x_i and y_i are two coordinate values of position p_i. For the i-th RSU, we use u_i to denote the set of fully-covered road segments, and use s_i to denote the set of partially-covered road segments. Additionally, we make definitions of $U{\stackrel{\rm def}{=}}\cup_{i=1}^{n}u_i$ and $S{\stackrel{\rm def}{=}}\cup_{i=1}^{n}s_i$. The profit of solution $p_{\rm s}$ of deploying n RSUs in the road network, denoted as $B(G(V,E),p_{\rm s},n)$, can then be calculated as Eq.(1). Here $M_e{\stackrel{\rm def}{=}}e{\cap}O$ represents the part of road curve e covered by O. Here O represents the union of the coverages of all RSUs. The integrals in Eq.(1) are line integrals where $\mathrm{d}s$ is the integral variable.

$$B(G(V,E),p_{\rm s},r_{\rm s},n){\stackrel{\rm def}{=}}\begin{array}{l}\displaystyle\sum_{e\in{\rm U}}\int_{e}f_{e,w}(x,y)\mathrm{d}s\\ \quad+\displaystyle\sum_{M_e\in S}\int_{M_e}f_{e,w}(x,y)\mathrm{d}s\end{array}\tag{1}$$

For a given road-network $G(V,E)$, deployment region A, and the set of radii $r_{\rm s}$ of the coverages of the RSUs, we denote the RDP problem as $RDPM(G(V,E),B,A,r_{\rm s},n)$. Then, the RDP problem of finding an optimal deployment solution with maximum profit for n RSUs can be expressed as Eq.(2). Here $P_{\rm S}$ represents the set of all valid solutions.

$$\begin{array}{rl}p_{\rm s}^{*}{=}\mathbf{arg\,max}\limits_{p_{\rm s}\in P_{\rm S}} & B(G(V,E),p_{\rm s},r_{\rm s},n)\\ \text{subject to}: & p_{\rm s}{\stackrel{\rm def}{=}}\{p_1,p_2,\cdots,p_n\};\\ & p_i{\stackrel{\rm def}{=}}p_i(x_i,y_i),\ \ i{=}\{1,2,\cdots,n\};\\ & (x_i,y_i){\in}A,\ \ i{=}\{1,2,\cdots,n\};\end{array}\tag{2}$$

B. One-Dimensional RDP Problem Model

We classify RDP problems according to the dimensions of the related road networks. In the previous subsection, the RDP problem is described based on a two-dimensional road network, so the problem is called two-dimensional RDP problem. Otherwise, if the road network is a line, then the corresponding problem is called a one-dimensional RDP problem.

Looks more intractable and having no clear ideas for optimally solving the two-dimensional RDP problem, we restrict to the one-dimensional RDP problem here. For expression clarity, in later text, **we use phrase *RDP problem* to refer to *one-dimensional RDP problem* by default**.

In the one-dimensional RDP problem, the road network is indeed a road line consisting of several road segments. Shape of a road reduces to a line. Profit density function of the road curve degrades to a one-dimensional function defined on the corresponding line segment. The integrals in Eq.(1) become normal single-variable integrals. An example of a road line is shown in Fig.1. In this figure, the belt with light blue background represents the road line. The two end points of a road segment are called segment ends. Taking the road line as x-axis, using profit density as y-axis, the red curves in Fig.1 represent profit density functions of the segments.

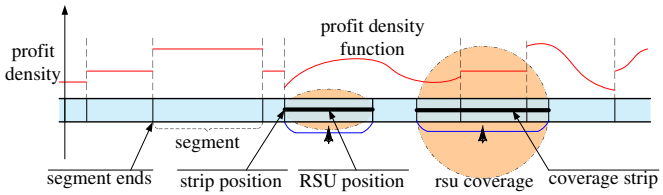


Fig. 1. An example of the road line of a one-dimensional RDP problem.

For a segment e with ends $x_{e,0}$ and $x_{e,1}$, its profit density function $f_{e,w}(x)$ can be expressed as a function in coordinate system (x, w) as shown in Eq.(3).

$$f_{e,w}(x)\stackrel{\text{def}}{=}\begin{cases} g_e(x), & x\in[x_{e,0}, x_{e,1}] \\ 0, & x\notin[x_{e,0}, x_{e,1}] \end{cases} \tag{3}$$

If the road line consists of L segments, then their profit density functions can be combined into one function as described in Eq.(4), which can then be regarded as the profit density function of the whole road line. Here $e_{e,i}$, $i\in\{0, 1, \ldots, L\}$, are segment ends, and they satisfy $x_{e,i-1}<x_{e,i}$. Function $g_{e,i}(x)$ can be any non-negative differentiable function, whereas the whole function $g_e(x)$ is not required to be continuous at the segment ends. Such a L-segment piecewise differentiable function is general enough to describe any practical profit density functions. The differentiable assumption indeed is not a restriction, since that for any function with some non-differentiable at some points, we can split it into more segments at these non-differentiable points. This operation leads to a piecewise differentiable function with more segments.

$$f_{e,w}(x){:}w{=}g_e(x){=}\begin{cases} g_{e,0}(x){:=}0, & x\in(-inf, x_{e,0}) \\ g_{e,1}(x), & x\in[x_{e,0}, x_{e,1}] \\ \ldots, & \ldots \\ g_{e,L}(x), & x\in(x_{e,L-1}, x_{e,L}] \\ g_{e,L+1}(x){:=}0, & x\in(x_{e,L}, inf) \end{cases} \tag{4}$$

As shown in Fig.1, in one-dimensional RDP problem, whatever shape the coverage of a RSU has, its actual effect on covering the road line is equivalent to a line strip. The strip's length equals the length of the road line in the coverage. To maximize profit, the strip with maximum possible length should be used. For a circular coverage, the maximum length of the coverage strip equals the coverage's diameter, and the RSU should be placed at road side. Hence, the one-dimensional RDP problem is indeed to find the positions of a given number of RSUs along the road line such that accumulated profits of road parts covered by these RSUs are maximized.

In the following text, the concept **strip** is used to represent the effective coverage of a RSU on the road line, and the concept **strip length** means the length of the strip, which is denoted as d. If a strip covers a road segment of $[x, x{+}d]$, then we say that the strip is at x. For a RSU with circular coverage region of radius $r{=}d/2$, "place the strip at x" has the same meaning as "place the RSU at $x{+}r$". In the following text, x is used as strip position by default, and $B(G(V, E), x{+}r, r, 1)$ is abbreviated to $B(x)$.

C. Comparison of Our RDP Problem with GSC and WMC

Our RDP problem here looks similar to the well known Geometric Set Cover (GSC) problem [16] and the Weighted Maximum Coverage (WMC) problem [17].

The GSC problem is the special case of the set cover problem in geometric settings [16]. The input is a range space $\Sigma{=}(X, \mathcal{R})\}$ where X is a universe of points in $\mathbb{R}^d$ and $\mathcal{R}$ is a family of subsets of X called regions, defined by the intersection of X and geometric shapes such as disks and axis-parallel rectangles. The goal is to select a minimum-size subset $\mathcal{C}\subseteq\mathcal{R}$ of regions such that every point in the universe X is covered by some range in $\mathcal{C}$.

Although looks similar, Our RDP problem and the GSC problem are different: (1)what should be covered in the GSC problem are some discrete points in $\mathbb{R}^d$, whereas they are curves with non-uniform weight densities in the RDP problem; (2)the objective of the GSC problem is to cover all given discrete points with minimum number of regions, whereas the objective of the RDP problem is to obtain maximum coverage profits with a given number of regions with given shapes and sizes. In the one-dimensional case, where X contains points on the real line and $\mathcal{R}$ is defined by intervals, the GSC problem can be solved in polynomial time using a simple greedy algorithm. In higher dimensions where $d{\geq}2$, the GSC problem is known to be NP-complete even for simple shapes, i.e., when $\mathcal{R}$ is induced by unit disks or unit squares [28].

The WMC problem is the weighted version of the maximum coverage (MC) problem. MC problem is also called as max k-cover problem [29]. In the WMC problem, we are given a collection of elements with weights, several sets and a number n, the task is to select n sets such that total weights of the elements in these sets are maximized. The WMC problem is NP-hard, and cannot be approximated within $1{-}\frac{1}{e}{+}o(1){\approx}0.632$ under standard assumptions [17].

Our RDP problem can be regarded as the geometric extension of the maximum coverage (MC) problem. Although no proofs were found, we suspect that our RDP problem may be at least as hard as the GSC problem and the WMC problem. We also suspect that the one-dimensional RDP problem is also intractable, and we have't found existing works that solve the

one-dimensional RDP problem. Our understanding even to the new one-dimensional RDP problem is still much lack.

IV. Properties of One-Dimensional RDP problem

A. One-Dimensional RDP problem with One RSU

For a one-dimensional RDP problem of one RSU with radius $r{=}d/2$, if the profit density function of the road line is a L-segment piecewise differentiable function, and if $x_{e,L}{-}x_{e,0}{\leq}d$, then it is easy to notice that all $x{\in}[x_{e,0}{+}r, x_{e,L}{-}r]$ are optimal solutions with the same profit. Hence, we will omit this trivial case in later analyses.

For optimal solutions of one-dimensional RDP problem with one RSU, we have the following Lemma 1. For expression clarity, we define several sets as in Eq.(5). Here $g_{e,i+1}(x_{e,i}^{+})$ represents the limit of $g_e(x)$ as x infinitely approaches to $x_{e,i}$ from the right side, $g'_e(x)$ is the derivative of $g_e(x)$ with respect to x. $g'_{e,i+1}(x_{e,i}^{+})$ represents the limit of $g'_{e,i+1}(x)$ as x infinitely approaches to $x_{e,i}$ from the right side. $g'_{e,i}(x_{e,i}^{-})$ represents the limit of $g'_{e,i}(x)$ as x infinitely approaches to $x_{e,i}$ from the left side. Other symbols are defined similarly. It is easy to notice that the positions in $X_{\text{cand}}(d)$ are all local maximum points.

$$\begin{aligned}
X_{\text{dif}}(d) &\overset{\text{def}}{=} \left\{x\big|^{g_e(x+d)=g_e(x),g'_e(x+d)-g'_e(x)\leq 0,}_{x\in[x_{e,0},x_{e,L}-d]}\right\};\\
X_{\text{moveL1}}(d) &\overset{\text{def}}{=} \{x|g_e((x{+}d)^{-}){>}g_e(x^{-}), x{\in}[x_{e,0}, x_{e,L}{-}d]\};\\
X_{\text{moveL2}}(d) &\overset{\text{def}}{=} \left\{x\big|_{g'_e((x+d)^{-})\geq g'_e(x^{-}),x\in[x_{e,0},x_{e,L}-d]}^{g_e((x+d)^{-})=g_e(x^{-}),}\right\};\\
X_{\text{moveR1}}(d) &\overset{\text{def}}{=} \{x|g_e((x{+}d)^{+}){<}g_e(x^{+}), x{\in}[x_{e,0}, x_{e,L}{-}d]\};\\
X_{\text{moveR2}}(d) &\overset{\text{def}}{=} \left\{x\big|_{g'_e((x+d)^{+})\leq g'_e(x^{+}),x\in[x_{e,0},x_{e,L}-d]}^{g_e((x+d)^{+})=g_e(x^{+}),}\right\};\\
X_{\text{moveL}}(d) &\overset{\text{def}}{=} X_{\text{moveL1}}(d){\cup}X_{\text{moveL2}}(d);\\
X_{\text{moveR}}(d) &\overset{\text{def}}{=} X_{\text{moveR1}}(d){\cup}X_{\text{moveR2}}(d);\\
X_{\text{move}}(d) &\overset{\text{def}}{=} X_{\text{moveL}}(d){\cap}X_{\text{moveR}}(d);\\
X_{\text{cand}}(d) &\overset{\text{def}}{=} X_{\text{move}}(d);
\end{aligned} \tag{5}$$

Lemma 1: For a one-dimensional RDP problem of one RSU with strip length d, if the profit density function of the road line is L-segment piecewise differentiable, and if $x_{e,L}{-}x_{e,0}{>}d$, then all optimal strip positions must be in set $X_{\text{cand}}(d)$.

Proof: The profit $B(x)$ of placing the strip at x can be expressed as Eq.(6) by using Eq.(1).

$$B(x){=}\int_{x}^{x+d} g_e(s)\mathrm{d}s \tag{6}$$

To determine the properties that optimal positions of the strip should have, we consider the following two cases depending on whether the ends of the strip are at segment ends.

- **Case 1: Neither ends of the strip are at segment ends**. In this case, we conduct analyses in the following three subcases. Here we define function $id(x)\overset{\text{def}}{=}\{i|x{>}x_{e,i}, x{\leq}x_{e,i+1}\}$ for expression clarity.
  - **Subcase 1:** $id(x{+}d){=}id(x)$.
    In this case, Eq.(6) can be calculated as Eq.(7).

$$B(x){=}\int_{x}^{x+d} g_{e,id(x)}(s)\mathrm{d}s \tag{7}$$

Differentiate $B(x)$ with respect to x, we obtain

$$\frac{\mathrm{d}(B(x))}{\mathrm{d}x}{=}g_{e,id(x)}(x{+}d){-}g_{e,id(x)}(x) \tag{8}$$

  - **Subcase 2:** $id(x{+}d){=}id(x){+}1$.
    In this case, Eq.(6) can be calculated as Eq.(9).

$$\begin{aligned}\textstyle\int_{x}^{x+d} g_e(x)dx{=}&\textstyle\int_{x}^{x_{e,id(x)+1}} g_{e,id(x)}(s)\mathrm{d}s\\ &{+}\textstyle\int_{x_{e,id(x+d)}}^{x+d} g_{e,id(x+d)}(s)\mathrm{d}s\end{aligned} \tag{9}$$

Differentiate $B(x)$ with respect to x, we obtain

$$\tfrac{\mathrm{d}(B(x))}{\mathrm{d}x}{=}g_{e,id(x+d)}(x{+}d){-}g_{e,id(x)}(x) \tag{10}$$

  - **Subcase 3:** $id(x{+}d){>}id(x){+}1$.
    In this case, Eq.(6) can be calculated as Eq.(11).

$$\begin{aligned}\textstyle\int_{x}^{x+d} g_e(s)\mathrm{d}s{=}&\textstyle\int_{x}^{x_{e,id(x)+1}} g_{e,id(x)}(s)\mathrm{d}s\\ &{+}\textstyle\sum_{i=id(x)+1}^{id(x+d)-1}\int_{x_{e,i}}^{x_{e,i+1}} g_{e,i}(s)\mathrm{d}s\\ &{+}\textstyle\int_{x_{e,id(x+d)}}^{x+d} g_{e,id(x+d)}(s)\mathrm{d}s\end{aligned} \tag{11}$$

Differentiate $B(x)$ with respect to x, we obtain

$$\frac{\mathrm{d}(B(x)))}{\mathrm{d}x}{=}g_{e,id(x+d)}(x{+}d){-}g_{e,id(x)}(x) \tag{12}$$

In whichever of the three cases above, the derivative of $B(x)$ with respect to x can be expressed as

$$\begin{aligned}\tfrac{\mathrm{d}(B(x))}{\mathrm{d}x} &{=}g_{e,id(x+d)}(x{+}d){-}g_{e,id(x)}(x)\\ &{=}g_e(x{+}d){-}g_e(x)\end{aligned} \tag{13}$$

Each x that maximizes $B(x)$ must satisfy Eq.(14). Such positions are in set $X_{\text{dif}}(d)$.

$$\frac{\mathrm{d}(B(x))}{\mathrm{d}x} {=}g_e(x{+}d){-}g_e(x){=}0; \tag{14a}$$

$$\frac{\mathrm{d}^2(B(x))}{\mathrm{d}x^2} {=}g'_e(x{+}d){-}g'_e(x){\leq}0; \tag{14b}$$

- **Case 2: One or two ends of the strip are at segment ends**.
  In this case, either x or $x{+}d$ or both of them are at segment ends, and the derivatives at these ends do not exist. The analyzing method used in case 1 is not applicable. However, we can analyze in another way as follows. For a strip at x that maximizes $B(x)$, it must satisfy that moving the strip toward left or right for an infinitely small step should both decreases or at least not increases the profit of the strip.
  For the case of moving left, the requirement of not increasing the profit leads to Eq.(15), which leads to $X_{\text{moveL}}(d)$.

$$g_e((x{+}d)^{-}){>}g_e(x^{-}); \tag{15a}$$

$$g'_e((x{+}d)^{-}){\geq}g'_e(x^{-}),\ \text{if}\ g_e((x{+}d)^{-}){=}g_e(x^{-}); \tag{15b}$$

For the case of moving right, the requirement of not increasing the profit leads to Eq.(16), which leads to $X_{\text{moveR}}(d)$.

$$g_e((x+d)^+) < g_e(x^+); \quad (16a)$$

$$g'_e((x+d)^+) \leq g'_e(x^+), \text{ if } g_e((x+d)^+) = g_e(x^+); \quad (16b)$$

Since that x that maximizes $B(x)$ should satisfy both Eq.(15) and Eq.(16), all such x must be in set $X_{\text{move}}(d) = X_{\text{moveL}}(d) \cap X_{\text{moveR}}(d)$.

Combining above two cases, we obtain that all x maximizing $B(x)$ must be in set $X_{\text{dif}}(d) \cup X_{\text{move}}(d)$. Furthermore, we can notice that each $x \in X_{\text{dif}}(d)$ must also satisfy both Eq.(15b) and Eq.(16b), hence we have $X_{\text{dif}}(d) \subseteq (X_{\text{moveL2}} \cap X_{\text{moveR2}}) \subseteq X_{\text{move}}$. As a result, we have $X_{\text{dif}}(d) \cup X_{\text{move}}(d) = X_{\text{move}}(d)$. The positions in $X_{\text{move}}(d)$ are called as candidate positions for the strip. To make symbols more meaningful, we define the set $X_{\text{cand}}(d)$ in Eq.(5) as an alias of $X_{\text{mode}}(d)$. The Lemma follows. ■

Using Lemma 1, we can obtain the optimal positions of the RSU directly by (1) calculating $B(x)$ for all x in $X_{\text{cand}}(d)$; (2) get the x with maximum $B(x)$. The set $X_{\text{cand}}(d)$ may contain continuous regions in form of $[x_1, x_2]$, and thus the size of $X_{\text{cand}}(d)$ can be infinite. In this case, we can let $X_{\text{cand}}(d)$ contain only the end points of such regions, and then get the x with maximum $B(x)$. If the x obtained in this way is originally an end point of a region, then any position in the corresponding region will also be an optimal position.

B. One-Dimensional Problem with Multiple RSUs

If the total length of the RSUs' coverages is not smaller than the length of the road line, an obvious optimal solution is to place the strips along the road line such that the whole road is completed covered by the union of the strips. There may be many such optimal solutions. In later text, we omit this trivial case and focus on the normal non-trivial case. In the pseudo code of our algorithm in later text, code lines corresponding to the trivial case are also omitted.

When there are multiple RSUs, if the distance between two RSUs is smaller than the sum of their coverage radii, then the coverages of the two RSUs will overlap each other, or in other words, the corresponding strips overlap with each other. If there are overlapped strips in a solution, then we call it an overlapped solution, otherwise a non-overlapped solution. Maximizing each strip's profit may help to maximize the total profit of a solution, but the overall profit of overlapping strips are not independent, i.e., covering a road segment already covered by other strips brings no benefit. Helpfully, the one-dimensional RDP problem has the property in Lemma 2, which enables us to design efficient algorithms.

Lemma 2: For any one-dimensional RDP problem with multiple RSUs, there must be non-overlapped optimal solutions.

Proof: For any overlapped solution of a RDP problem instance, we can obtain a non-overlapped solution by stretching apart the overlapped strips meanwhile guaranteeing that the road segments covered by the overlapped solution are still covered by the non-overlapped solution. Thus, the profit of the non-overlapped solution must be not smaller than that of the overlapped solution. This property also applies to overlapped optimal solutions. Hence, if there are optimal solutions where strips overlap, then we must can obtain a new non-overlapped solution meanwhile it is optimal. The Lemma follows. ■

In a non-overlapped solution, some strips may be adjacent to each other. By the adjacency relations, the strips in a solution form several groups, where the strips in each group are adjacent one-by-one, whereas the strips in different groups are separated. For a strip group contains m strips, assume that the strips from left to right in the group have length $d_1, d_2, \ldots, d_m$, then we can regard the strip group as a virtual strip s_v with length $d_v = \sum_{j=1}^{m} d_j$. For expression clarity, we can also regard an isolated strip as a virtual strip which has only one strip. If the virtual strip s_v is placed at x, then the positions of the strips in the virtual strip are also determined, where the k-th strip is placed at $x_k = x + \sum_{j=1}^{k-1}(d_j)$, $1 \leq k \leq m$. We denote the position set $\{x_1, x_2, \ldots, x_m\}$ as $X_{\text{VS}}(d_v, x)$. Given a RDP problem instance with n different RSUs, we denote the set of all possible lengths of the virtual strips as S_{LVS}. Then we define a set of positions $X_{\text{cand},n} \stackrel{\text{def}}{=} \cup_{d \in S_{\text{LVS}}}(\cup_{x \in X_{\text{cand}}(d)}(X_{\text{VS}}(d, x)))$.

It is obvious that swapping positions of the strips in a virtual strip dose not change the overall profit of the virtual strip. Hence, if a virtual strip can be transformed to another virtual strip by only rearranging the sequences of the strips, then we regard that **the two virtual strips are identical**. Given two solutions, if the virtual strips in one solution have a one-to-one identical relation with the virtual strips in the other solution, then we regard that **the two solutions are identical**. With these concepts and definitions, we have the following lemma.

Lemma 3: For any one-dimensional RDP problem with n RSUs having different coverage diameter, for any non-overlapped optimal solution, there must be an identical solution which is also optimal and the positions of the strips in this solution are all in set $X_{\text{cand},n}$.

Proof: We will prove it by contradiction. Assume that there is an optimal solution where positions of some strips are not in set $X_{\text{cand},n}$. These strips may form virtual strips by the adjacency relations. For any of these virtual strips, we assume that its length is d, then we must have $d \in S_{\text{LVS}}$. According to Lemma 1, the position x of this virtual strip must be in $X_{\text{cand}}(d)$. Otherwise, we can increase the profit of the virtual strip by moving the virtual strip as a whole toward left or right, which contradicts with the assumption that this solution is optimal. As a result, this virtual strip must have an identical virtual strip where the positions of the strips in this later virtual strip must all be in set $X_{\text{cand}}(d)$. Similarly, there must be a solution with strip positions all in $X_{\text{cand}}(d)$ meanwhile it is identical to the optimal solution. The lemma follows. ■

If $X_{\text{cand},n}$ contains continuous regions, then it can be treated using the method applied to $X_{\text{cand}}(d)$ in the previous section.

If the strips have different lengths, then $|S_{\text{LVS}}|$ can be as larger as 2^n, and the number of all possible different combinations of virtual strips can roughly be expressed as $\sum_{i=1}^{n} \frac{i^{n-1} - (i-1)^n}{(i-1)!}$.

To search for an optimal solution for a RDP problem with multiple RSUs, a straightforward way is to traverse all possible solutions. To do so, profits of all possible solutions may need to be checked thoroughly, where each solution consists of a combination of several virtual strips, and each virtual strip

s_v has its own corresponding set $X_{\text{cand}}(d_v)$. According to Lemma 2, we need only to consider non-overlapped solutions. However, when there are multiple RSUs, if the strips have different lengths, then $|S_{\text{LVS}}|$ can be as larger as 2^n, and the number of all possible different combinations of virtual strips can roughly be expressed as $\sum_{i=1}^{n} \frac{i^{n-1}-(i-1)^n}{(i-1)!}$. Thus, the number of possible solutions may be prohibitively large to be tackled. Contrastively, if all the strips have the same length, then we have $S_{\text{LVS}}{=}n$, and the number of all possible different combinations of virtual strips can roughly be expressed as 2^{n-1}. Thus, the one-dimensional RDP problem with identical RSUs is possibly more tractable. In later text, we focus on the later problem where all strips have length d. Although greedy or heuristic algorithms can be used to find quasi-optimal solutions, it is non-trivial to find an optimal solution.

C. One-Dimensional Problem with Multiple Identical RSUs

If all strips have the same length d, then each possible position x of a virtual strip consisting of m strips must be in set $X_{\text{cand}}(m{*}d)$. This x actually corresponds to a list $[x, x{+}d, \ldots, x{+}(i{-}1)d]$, which are the positions of the strips in the virtual strip. Using notation $X{+}d \stackrel{\text{def}}{=} \{x{+}d | x{\in}X\}$, the set of all candidate positions of strips determined by $X_{\text{cand}}(m{*}d)$ can be expressed as $X_{\text{cand}}(m,d){=}\bigcup_{i=0}^{m-1}(X_{\text{cand}}(m{*}d){+}i{*}d)$, so we have $|X_{\text{cand}}(m,d)|{=}m{*}|X_{\text{cand}}(m{*}d)|$. Here $|s|$ denotes the size of set s. Making a symbol definition $X_{\text{cand,c}}(i,d) \stackrel{def}{=} \bigcup_{j=1}^{i} X_{\text{cand}}(j,d)$, the set of possible positions of all strips corresponding to all possible virtual strips is indeed $X_{\text{cand,c}}(n,d)$. We have $|X_{\text{cand,c}}(n,d)| {\le} \sum_{i=1}^{n}(i{*}|X_{\text{cand}}(i{*}d)|)$. If $|X_{\text{cand}}(i{*}d)| = |X_{\text{cand}}((i{+}1){*}d)|{=}N,\ i{\in}\{1,2,\ldots,n{-}1\}$, then we have $|X_{\text{cand,c}}(n,d){\le}\frac{n(n+1)}{2}N$. However, if most road segments have constant profit density function, Eq.5 implies that most elements in $X_{\text{cand}}(i,d)$ are also elements of $X_{\text{cand}}(i{+}1,d)$. Thus, we can roughly assume that $X_{\text{cand}}(i,d){\subseteq}X_{\text{cand}}(i{+}1,d)$. As a result, $|X_{\text{cand,c}}(n,d)|{\approx}n{*}X_{\text{cand}}(1,d)|{=}\mathcal{O}(nL)$.

Each position in $X_{\text{cand,c}}(n,d)$ is a candidate position for a strip, and the ends of all the strips with position in $X_{\text{cand,c}}(n,d)$ divides the whole road line into small pieces, later we call these pieces as **slices**.

With these preparations, the main work to solve a one-dimensional RDP problem is actually to select the positions of n non-overlapped strips from the set $X_{\text{cand,c}}(n,d)$ such that the overall profit of these selected strips is maximum.

For a one-dimensional RDP problem with multiple identical RSUs, the overlap relationships among the strips with positions in $X_{\text{cand,c}}(n,d)$ can be expressed as a weighted graph. In this graph, each node represents a candidate strip with its corresponding position, the weight of the node is the profit of the strip. If two strips are overlapped, a link is inserted between the two corresponding nodes. This graph is indeed the interval graph of the strips in $X_{\text{cand,c}}(n,d)$.

With this graph, the RDP problem is equivalent to the problem of finding n independent nodes with maximum total weight. This problem has relation to a well known Non-deterministic Polynomial Complete(NPC) problem in graph theory, named Maximum Weighted Independent Set (MWIS) [30]. In MWIS, the number of nodes to be selected is a variable to be solved. Different from the MWIS problem, the number of nodes to be selected here is a given parameter.

V. Greedy based Approximate Algorithms to the RDP Problem with Heterogenous RSUs

Suspecting that finding an optimal solution for the one-dimensional RDP Problem with multiple RSUs of different radii is intractable, greedy-based approach is a natural way for obtaining approximate solutions. We propose two greedy-based approximate algorithms Greedy2P3 and Greedy2P3E. According to Lemma 2, our algorithms only consider non-overlapped solutions, i.e., solutions where the strips are non-overlapped.

A. Greedy2P3 Algorithm

Pseudo code of the Greedy2P3 algorithm is shown in Algorithm 1. In this code, structural variable $roadNet$ contains raw information about the problem, structural variable $dataMat$ contains information such as list of candidate strip positions, list of strip profits, strip overlap information. $dataMat$ is generated by function `getDataMatfromRoad`($roadNet$). Function `getPosMaxFree`($dataMat, posList, dList(i)$) return the strip with maximum profit among all strips uncovered by the strips in $optList$. Function `sortDescending`($dList$) returns the sorted list of the diameters of the RSUs in descending order. Function `UpdateRoadNet`($dataMat, dList(i)$) updates road line information, i.e, resets the profit density function of the road parts covered by the strip at $dList(i)$ to 0, and obtain the set $X_{cand,1}(dList(i))$ corresponding to the updated road line using Lemma 1.

Algorithm 1 The Greedy2P3 Algorithm

Require: n: number of RSUs to be deployed;
$roadNet$: road network;
$dList$: the list of the diameters of the RSUs;
Ensure: $posList$: the list of the selected strips;
$maxProfit$:the profit of the best solution found;

```
posList=[ ]; maxProfit=0;
dList=sortDescending(dList);
dataMat=getDataMatfromRoad(roadNet);
for i=1 : n do
    [dataMat]=UpdateRoadNet(dataMat, dList(i));
    [pos, val]=getPosMaxFree(dataMat, posList, dList(i));
    posList=[posList, pos];
    maxProfit=maxProfit+val;
end for
return [posList, maxProfit];
```

B. Greedy2P3E Algorithm

In Greedy2P3, function getPosMaxFree($dataMat$, $optList, dList(i)$) only considers strips non-overlap with the strips in $optList$, which neglects possible better strips. Such an example is shown in Fig.3 in Section VII. In this instance, n=3, and the whole road line is divided into 11 slices. The symbol in a slice indicates the profit obtained by covering the slice. Here a,b are non-negative constants and satisfy $a>b$, and ε is a non-negative constant which can be infinitely small. For this instance, Greedy2P3 will select three strips cg_1, cg_2, and cg_3 one by one. However, it is easy to notice that the strip cg_3' is a better option when selecting the third strip. This instance implies that, Greedy2P3 may neglect some simple chances that lead to better profit. This motivates us to propose Greedy2P3E. Greedy2P3E exploits such chances when selecting new strips, and then after selecting enough strips, the strips are moved so as to obtain a final better non-overlapped solution.

Algorithm 2 Greedy2P3E Algorithm

Require: n: number of RSUs to be deployed;
$roadNet$: road network;
$dList$: the list of the diameters of the RSUs;
Ensure: $posList$: the list of the selected strips;
$maxProfit$:the profit of the best solution found;
1: $posList$=[ ]; $maxProfit$=0;
2: $dList$=sortDescending($dList$);
3: $dataMat$=getDataMatfromRoad($roadNet$);
4: **for** i=1 : n **do**
5: [$dataMat$]=UpdateRoadNet($dataMat, dList(i)$);
6: [$pos, addiVal$]=getPosMaxAddiProfit(
7: $dataMat, posList, dList(i)$);
8: $posList$=[$posList, pos$];
9: $maxProfit$=$maxProfit$+$addiVal$;
10: **end for**
11: **while** isExist(overlapped strips in $posList$) **do**
12: [ccs, n_{ccs}]=selContinueSegOverlap($posList$);
13: $posList$=solveOverlaps($roadNet, posList, ccs, n_{ccs}$);
14: **end while**
15: $maxProfit$=calProfit($posList$);
16: **return** $posList, maxProfit$;

The Greedy2P3E algorithm solves a RDP problem instance in two stages. Code lines 4-10 forms Stage1, code lines 11-14 forms Stage2. Stage1 is similar to Greedy2P3 except that function getPosMaxFree($\cdot$) is replaced by function getPosMaxAddiProfit($\cdot$). The latter function selects a strip with maximum additional profit. Additional profit of the strip represents the total profit of road part only covered by this strip, whereas the profits of the road part covered by previously selected strips are not considered.

If the solution obtained in Stage1 has overlapped strips, the additional Stage2 will be performed. In stage2, each overlapped strip set is replaced with a continuous non-overlapped strip set, where the left most strip in the overlapped strip set is kept unchanged, whereas all other strips in the overlapped strip set are extended towards right one by one. To be specific, function selContinueSegOverlap($posList$) selects one overlapped strip set, and function solveOverlaps($\cdot$) replaces the selected overlapped strip set with its corresponding continuous non-overlapped strip set.

C. Property Analysis on the Algorithms

Lemma 4: For any RDP problem instance, a solution found by Greedy2P3E must be not worse than one found by Greedy2P3.

Proof: Both algorithms select new strips repeatedly by selecting one in each loop. For the first two loops, the strips selected by Greedy2P3 and Greedy2P3E must be the same. In the later loops, it can be noted easily that the set of candidate strips inspected by function getPosMaxAddiProfit($\cdot$) in Greedy2P3E must be a superset of that inspected by function getPosMaxFree($\cdot$) in Greedy2P3P. Hence, the profit of the strip selected in Greedy2P3E must be not smaller than one selected in Greedy2P3. Therefore, total profit of solution found by Greedy2P3PE after Stage1 must be not smaller than one by Greedy2P3. Furthermore, the optional Stage2 in Greedy2P3E will replace overlapped solutions with a non-overlapped one, meanwhile guaranteeing that the profit of the non-overlapped solution is not smaller than that of the overlapped one. As a conclusion, the lemma follows. ■

Although Greedy2P3 is not better than Greedy2P3E, we prove in later sections that Greedy2P3 has a tight constant approximation ratio for one-dimensional RDP problems with multiple RSUs of identical diameter, thus provides a lower bound for the approximation ratio of Greedy2P3E.

Running time of the greedy algorithms depends on $|X_{\text{cand}}(dList(i))|$ (i=1, 2, ..., n). $|X_{\text{cand}}(dList(i))|$ depends on segment number L of the whole road line. Eq.(5) implies that, if all segments have constant profit density function, then we have $|X_{\text{cand}}(dList(i))|\leq 2*L$. Otherwise, $|X_{\text{cand}}(dList(i))|$ may be boundlessly large, but such cases are unusual in practice. Ignoring such less practical cases, we denote that $|X_{\text{cand}}(dList(i))|=N$. Then, for time complexity of the two algorithms, we have the following Lemma 5.

Lemma 5: Time complexity of Greedy2P3 and Greedy2P3E are both $\mathcal{O}(nN)$.

Proof: Main operation in the loop in Greedy2P3 is to obtain set $X_{\text{cand}}(dList(i))$, and calculates their corresponding profits. This operation requires time $\mathcal{O}(N)$. Function getPosMaxFree($\cdot$) returns the best free strip in time $\mathcal{O}(N)$. Hence, each loop requires time $\mathcal{O}(N)$. The loop repeats n times, hence total running time will be $\mathcal{O}(nN)$. Stage1 of Greedy2P3E also has time complexity $\mathcal{O}(nN)$. Stage2 of Greedy2P3E can be completed in time $\mathcal{O}(n)$. Hence, time complexity of Greedy2P3E is $\mathcal{O}(nN)+\mathcal{O}(n)=\mathcal{O}(nN)$. The lemma follows. ■

Greedy algorithms are approximate algorithms, which usually return approximate solutions. For a certain problem instance, the ratio of the profit of a solution returned by an approximate algorithm to that of an optimal solution is named **performance ratio** of the approximate algorithm with the given instance. For different problem instances, the performance ratio is usually different. In the literature, another concept of **approximation ratio** [3], [8] is used to measure

the approximation ability of approximate algorithms, which is defined as the minimum of all possible performance ratios.

Lemma 6: For approximation ratio of Greedy2P3E, denoted as α_{2P3E}, we have $\alpha_{\text{Greedy2P3E}} \geq 1-(\frac{n-1}{n})^n$.

Proof: Results in [29] shows that, for max n-cover problem, if the profit function f defined on the covers meets Eq.(17), then the greedy algorithm, which iteratively selects the set that has the maximum additional profit, has an approximation ration of at least $1-(\frac{n-1}{n})^n$. Obviously our deployment profit model ensures that the profit of strips in our RDP problem satisfies Eq.(17). Furthermore, function $\texttt{sortDescending}(dList)$ ensures that the strips are treated in descending order, so our Greedy2P3E algorithm works exactly greedily. Thus, the result about the approximation ratio also applies to Greedy2P3E. The lemma follows. ∎

$$\begin{aligned} f(U)+f(V) \geq \quad & f(U \cup V)+f(U \cap V), \\ & U \text{and} V \text{are any covers;} \\ f(\emptyset)=0; & \end{aligned} \tag{17}$$

VI. Optimal Algorithm OptGreDyn for One-dimensional RDP Problem with identical RSUs

A. Optimal Algorithm OptGreDyn

According to Lemma 3 and Lemma 2, set $X_{\text{cand,c}}(n,d)$ must contain at least one optimal solution. Thus, the task of the RDP problem is indeed to find a combination of n different values of x such that the corresponding strips are non-overlapped meanwhile their total profit is maximum. A straight forward way is by traversing all $\mathcal{O}((M)^n)$ possible combinations of strips, here M represents $|X_{\text{cand,c}}(n,d)|$. As M becomes much large, this approach will be very time consumptive.

For the one-dimensional RDP problem with multiple identical RSUs, we found the following property.

Lemma 7: For the best strip s_1 in $X_{\text{cand,c}}(n,d)$, there must be an optimal non-overlapped solution that satisfies one of the two conditions: (Cond1)the optimal solution contains s_1; (Cond2)the optimal solution contains two other strips s_2 and s_3 such that s_2 and s_3 are non-overlapped with each other, but they all overlap with s_1.

Proof: We will prove by contradiction. According to Lemma 2, there must be non-overlapped optimal solutions. Assume that all these optimal solutions do not satisfy the two conditions. Then for each such optimal solution S_1, we can replace any one strip s_4 in the solution with strip s_1, thus obtain a new solution S_2. The assumption implies that s_4 non-overlap with s_1. Since that s_1 is the best one in $X_{\text{cand,c}}(n,d)$, we have $B(S_2)-B(S_1)=B(s_1)-B(s_4) \geq 0$. Hence, S_2 must also be an optimal solution, which satisfies Cond1, thus we have a contradiction. The lemma follows. ∎

We use X to represent $X_{\text{cand,c}}(n,d)$ for expression brevity. For a strip s_1, we denote the set of the pairs (s_2,s_3) satisfying cond2 as $C(s_1)$. Given X and n, we use $\mathcal{S}(X,n)$ to denote the profit of an optimal solution, then Lemma 7 implies Eq.(18). Here $X-s$ represents the set obtained from X by removing all strips that overlap with strip s.

$$\mathcal{S}(X,n)=\max\left(\begin{matrix} B(s_1)+\mathcal{S}(X-s_1,n-1)), \\ \{B(s_2)+B(s_3)+\mathcal{S}(X-s_2-s_3,n-2)\}_{(s_2,s_3)\in C(s_1)} \end{matrix}\right) \tag{18}$$

Eq.(18) enable us an efficient optimal algorithm combing greedy and dynamic programming techniques, hence comes the name OptGreDyn. Its pseudo code is shown in Algorithm 3. Function $\texttt{getOptSol}(X,n)$ implements Eq.(18). Function $\texttt{getPosMaxFree}(X)$ returns the best strip. Function $\texttt{sortProfitDesc}(X)$ sorts the strips in descending order of profit. Function $\texttt{getStripPairCond2}(X,x)$ returns the set of (s_2,s_3) pairs satisfy cond2. Here $B(x)$ denotes the profit of strip x.

Algorithm 3 OptGreDyn Algorithm

Require: n: number of RSUs to be deployed;
X: abbreviation of $X_{\text{cand,c}}(n,d)$;
d: length of each coverage strip;

```
[optList, maxP]=getOptSol(X, n);
Function [sol]=getOptSol(X, i)
maxP=0;
if i==1 then
    [x]=getPosMaxFree(X);
    return {x, B(x)};
else if i==2 then
    [xList]=sortProfitDescUpdate(X);
    for i=1:|X| do
        [optList, v]=getOptSol(X−xList(i), i−1);
        if B(xList(i))+v>maxP then
            maxP=v + B(xList(i));
            optList=[xList(i), optList];
        end if
        if B(xList(i))<v then
            break;
        end if
    end for
    return {optList, maxP};
else
    [xList]=sortProfitDesc(X);
    [optList, v]=getOptSol(X−xList(1), i−1);
    if B(xList(1))+v>maxP then
        maxP=B(xList(1))+v;
        optList=[xList(1), optList];
    end if
    [C]=getStripPairCond2(X, xList(1));
    for each(s2, s3)∈C do
        [optList, v]=getOptSol(X−s2−s3, i−2);
        if B(s2)+B(s3)+v>maxP then
            maxP=B(s2)+B(s3)+v;
            optList=[s2, s3, optList];
        end if
    end for
    return {optList, maxP};
end if
End Function
```

Lemma 8: A solution obtained by the OptGreDyn algorithm must be optimal.

Proof: We prove it by mathematical induction. It is obvious that when n=1 and n=2, a solution found by OptGreDyn must be optimal. Now assume that it is also true for n=i and n=i−1. Since that the code lines 21-35 implement Eq.(18), then using Lemma 7, the solution obtained at line 35 must be an optimal solution for n=i+1. The lemma follows. ■

Assume that the average size of $\mathcal{C}$ obtained by code line 27 in OptGreDyn is C, the average number of loops executed in the for loop implemented by code lines 9-18 is F. We usually have $C \ll n^2N$, $F \ll n^2N$.

Lemma 9: Time complexity of OptGreDyn is $\mathcal{O}(C^{(n-2)/2}F+M\log(M))$ when n=2∗i, and $\mathcal{O}(C^{(n-3)/2}(C+F)+M\log(M))$ when n=2∗i+1, where i is an integer.

Proof: In the pseudo code of the OptGreDyn algorithm, function `sortProfitDesc`(X) is called in several code lines with slightly different X. Hence, in practical implementation, we can sort X once and then reuse the sorted list later with some trivial updating operation. Thus, total time of all calls to function `sortProfitDesc`(X) can be assumed to be $T(s)$=$\mathcal{O}(M\log(M))$.

Denoting the running time of recursive iterations with n as $T(n)$, we have $T(n)$=$T(n-1)$+C∗$T(n-2)$. With $T(1)$=$\mathcal{O}(1)$ and $T(2)$=F∗$T(1)$=$\mathcal{O}(F)$, we obtain $T(2*i)$=$\mathcal{O}(C^{i-1}F)$ and $T(2*i+1)$=$\mathcal{O}(C^{i-1}(C+F))$.

Combining $T(s)$ and $T(n)$, the lemma follows. ■

B. Dynamic Limiting Technique

Another possible approach to the one-dimensional RDP problem with multiple identical RSUs is called as dynamic limiting technique. This technique works in a depth-first exhaustively-search-like way, but tries to reduce search space size in the searching process by dynamically setting the lower limits of the orders of the strips that can be used to construct candidate solutions. Here the order of a strip is its index in the sorted list of the strips in $X_{\text{cand,c}}(n,d)$ sorted in descending order of profit. Assume that we use $t[1{:}n] \stackrel{\text{def}}{=} [t_1, t_2, \ldots, t_n]$ to denote the order limits, then using the dynamic limiting technique, only the strips with orders smaller than t_i can be selected as the i-th strip of a candidate solution. Further more, when all RSUs have identical diameter, swapping positions of the strips in a solution do not make a difference. Hence, we need only search through solutions where the i-th strip has order smaller than the j-th strip, j>i. Once a new candidate solution is constructed and checked for optimality, the limits $t[1{:}n]$ will be adjusted when possible for reducing search space size. The underlying idea for adjusting limits is simple and can be cleared in the example. Assume an instance of the RDP problem with n=3, if we find a candidate solution $s[1{:}3]$=$[1,3,7]$, then we can set $t[1{:}3]$=$[s[3]-2, s[3]-1, s[3]]$=$[5,6,7]$ by the fact that any candidate solution, where the order of the first element is at least 5, must have profit not greater than $B_o([1,3,7])$. Hence, we can omit all solutions with $s[1]$≥5. Limits of other elements are determined similarly.

We have proposed an algorithm named as OptDynLim using the dynamic limiting technique. Simulations shows that OptDynLim is less efficient than OptGreDyn, so we just provide the idea here. In performance evaluation section, OptDynLim is used as a reference for other algorithms.

VII. Approximation Ratio of Greedy2P3 and Greedy2P3E when RSUs Having Identical Radii

The approximation ratio of the Greedy2P3 algorithm to RDP problem with multiple identical RSUs is 2/3, and it is tight. Before proving this, we first make some definitions as follows.

- **Greedy solution and greedy strip**: a solution returned by our Greedy2P3 algorithm is called a greedy solution. The strips in a greedy solution are called as greedy strips.
- **Optimal solution and optimal strip**: Solutions with maximum profit are called as optimal solutions. Here we only consider non-overlapped optimal solutions. The strips in an optimal solution are called as optimal strips.
- **Overlapped strips and non-overlapped strips**: If the joint of the coverages of two strips is non-empty, we say that they are overlapped, otherwise they are non-overlapped. **Adjacent strips** are regarded as non-overlapped.
- **Completely overlapped and partly overlapped**: For overlapped strips, there are two sub cases: if their coverages are identical, then they are completely overlapped, otherwise they are partly overlapped.
- **Free strips**: Strips those are non-overlapped with other strips are called as free strips.
- **direct connection, indirect connection, path**: For the pair of a greedy strip and an optimal strip, if the two strips are overlapped, we say that they have direct connection, or they are directly connected. For any two strips c_1 and c_n, if there is a list $[c_1, c_2, \cdots, c_n]$ such that c_j and c_{j+1} are directly connected, $j \in \{1, 2, \cdots, n-1\}$, then we say that c_1 and c_n are indirectly connected. This list is called a path. If two strips are either directly connected or indirectly connected with each other, we say that they are **connected** with each other.
- **Strip cell and Cell style**: For analysis purpose, small set of connected strips with typical structure is called a cell. Respect to connection relationships as well as relative superiority of the strips in profit, cells are classified into cell styles.
- **Stable, unstable, and invalid cell style**: If a cell style can actually exist without connecting to any other strips, we say that this style is stable. Otherwise, if a cell style can only exist by connecting to other strips, we say that it is unstable. Cell styles that could not actually exist is called invalid cell style.
- **Strip group and group style**: For a set of connected strips, if the strips are not connected with other strips not belonging to the set, then such a strip set is called as a strip group. According to the structural characteristics and relative priority of the strips in profit, strips groups are classified into group styles.
- **balanced group and unbalanced group**: A strip group with equal number of greedy strips and optimal strips is called a balanced group, otherwise it is a unbalanced group.

- **Logic group and logic group style**: By logically rearranging (e.g, removing from or inserting into) strips in unbalanced groups, we can make unbalanced groups become balanced. Balanced groups obtained by using **logic rearrangement operations** are called as logic groups. In a logical group, we say that the newly logically inserted strip and the previously contained strips in the group are **logically connected**. Logic groups are classified into **logic group styles**.
- **Greedy profit and optimal profit of a logic group**: For a logic group, the total profit of the greedy strips in the group is called the greedy profit of the group. Similarly, the total profit of the optimal strips are called as optimal profit of the group.

Cell style, group style, and logic group style are all abstract concepts. They are category representations of cells, groups, and logic groups, respectively.

The whole process for proving the approximation ratio of the Greedy2P3 algorithm can be roughly expressed in 5 steps as follows.

Step1: We totally identify 15 cell styles, and determine whether they are stable, unstable, invalid. (Refer to Appendix A-A).

Step2: Based on the properties of the cell styles, and using the fact that an unstable cell must join a group by connecting to stable cells through paths, we identify 8 group styles (Refer to Appendix A-B).

Step3: We make all unbalanced groups become balanced by using logic rearrangement operations. We identify 6 logic group styles (Refer to Appendix A-C).

Step4: For each logic group style, we prove its performance ratio(Refer to Appendix A-D).

Step5: We prove the approximation ratio of the Greedy2P3 algorithm by combining performance ratios of the logic group styles, and show that it is tight by providing some problem instances which can approach this approximation ratio.

The 6 logic group styles are shown in the sub figures in Fig.2. These logic group styles are denoted as LGS_a, LGS_b, LGS_c, LGS_d, LGS_e and LGS_f, respectively. In these sub figures, a solid arrow represents direct connection between the two corresponding strips, whereas a dashed arrow represents logic connection. An arrow from strip a to strip b means $B(b)\geq B(a)$, here $B(x)$ denotes the profit of strip x. A double ended arrow means that this arrow can have either direction. Parallel solid and dashed arrows between a pair of strips mean that the connection between the two strips can be either a direct connection or a logic connection.

In the following text, we only provide the detailed prove process for step 5. Detailed process for other steps are given in Appendix A.

***Theorem** 1:* The approximation ratio of the Greedy2P3 algorithm, denoted as α_{G2P3}, is at least 2/3, and it is tight for $n{=}3{*}i$, here i is any positive integer.

Proof: We will prove it by firstly showing that its performance ratio on any problem instance should be at least 2/3, and then showing that there are problem instances with performance ratio infinitely approach 2/3, thus the proof can be completed.

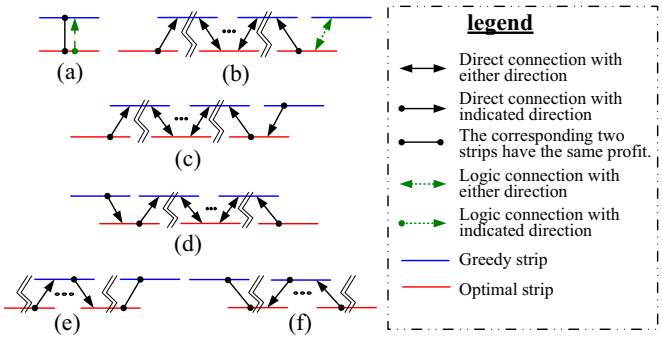


Fig. 2. Logic group styles.

As shown in Appendix A, for any instance of one-dimensional RDP problem with multiple identical RSUs, for any certain greedy solution and any non-overlapped optimal solution, the greedy strips and the optimal strips can must be grouped into logic groups of the 6 logic group styles. Without loss of generality, we assume that the numbers of logic groups of style LGS_a, LGS_b, LGS_c, LGS_d, LGS_e and LGS_f,are $n(a)$, $n(b)$, $n(c)$, $n(d)$, $n(e)$ and $n(f)$, respectively.

For any logic group $LG_{a,i}$ of style LGS_a, we denote its greedy profit and optimal profit as $B_{\text{G2P3}}(a,i)$ and $B_{\text{Opt}}(a,i)$, respectively. Notations for other logic groups are defined similarly. About the performance ratios of Greedy2P3 on logic groups of the 6 styles, results in section A-D shows that we have $\beta_{\text{LGS}}(b)\geq 2/3$, $\beta_{\text{LGS}}(c)\geq 2/3$, $\beta_{\text{LGS}}(d)\geq 2/3$, $\beta_{\text{LGS}}(e)\geq 1$ and $\beta_{\text{LGS}}(f)\geq 1$. Hence, the performance ratio of Greedy2P3 on the given problem instance, denoted as β_{G2P3}, can then be calculated as Eq.(19), which shows that $\beta_{\text{G2P3}}\geq 2/3$.

$$\begin{aligned}\beta_{\text{G2P3}}=\frac{B_{\text{G2P3}}}{B_{\text{Opt}}} &= \frac{\sum_{x\in\{a,b,c,d,e,f\}}\left(\sum_{i=1}^{n_x} B_{\text{2P3}}(x,i)\right)}{\sum_{x\in\{a,b,c,d,e,f\}}\left(\sum_{i=1}^{n_x} B_{\text{Opt}}(x,i)\right)} \\ &\geq \frac{\sum_{x\in\{a,b,c,d,e,f\}}\left(\sum_{i=1}^{n_x} \beta_{\text{LGS}}(x) B_{\text{Opt}}(x,i)\right)}{\sum_{x\in\{a,b,c,d,e,f\}}\left(\sum_{i=1}^{n_x} B_{\text{Opt}}(x,i)\right)} \\ &\geq \frac{\sum_{x\in\{a,b,c,d,e,f\}}\left(\sum_{i=1}^{n_x} 2/3{*}B_{\text{Opt}}(x,i)\right)}{\sum_{x\in\{a,b,c,d,e,f\}}\left(\sum_{i=1}^{n_x} B_{\text{Opt}}(x,i)\right)} \\ &= 2/3\end{aligned} \tag{19}$$

Now we will show that there are some RDP problem instances where the performance ratios of Greedy2P3 infinitely approach 2/3. Such an instance is shown in Fig.3. In this instance, $n{=}3$, and the whole road line is divided into 11 slices. The symbol in a slice indicates the profit obtained by covering the slice. Here a,b are non-negative constants, ε is a non-negative constant which can be infinitely small. Without loss of generality, we assume that $a{>}b$. For this instance, it is easy to check that the strip set $\{co_1, co_2, co_3\}$ is its optimal solution, whereas the greedy solution will be $\{cg_1, cg_2, cg_3\}$. Performance ratio β_{G2P3} of Greedy2P3 on this instance can be calculated as Eq.(20), which shows that $\beta_{\text{G2P3}}\to 2/3$ when $\varepsilon\to 0^+$.

$$\beta_{\text{G2P3}} = \frac{B(cg_1)+B(cg_2)+B(cg_3)}{B(co_1)+B(co_2)+B(co_3)} = \frac{2(a+b)+3\varepsilon}{3(a+b)+\varepsilon} \overset{\varepsilon\to 0^+}{=} 2/3 \tag{20}$$

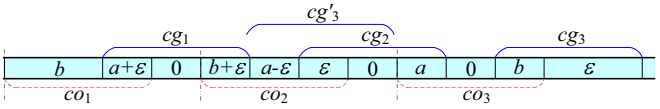


Fig. 3. An instance of the RDP problem where performance ratio of Greedy2P3 approaches 2/3.

For any $n{=}3*i$ (i is an integer), we assume a road line consists of i isolated road segments all identical to the example road line in Fig.3. Then for this road line, it is easy to notice that the greedy strips and the optimal strips must forms i strip groups identical to the strip group in Fig.3. The performance ratio of Greedy2P3 on this problem instance will be 2/3.

Combining Eq.(19) and Eq.(20), we obtain that the performance ratio of Greedy2P3 on any RDP instance is at least 2/3, and this value is reachable. Hence, Greedy2P3's approximation ratio α_{G2P3} is at least 2/3, and it is tight for $n{=}3{*}i$. ■

Comment: Although the approximation ratio is not very high, simulation results show that Greedy2P3 usually return quasi-optimal solutions with profit usually more than 98% of optimal solutions.

Combining Theorem 1 and Lemma 4, we can obtain Theorem 2 directly.

***Theorem** 2:* The approximation ratio of Greedy2P3E for one-dimensional RDP problem with n identical RSUs is at least $\max\{1{-}(\frac{n-1}{n})^n, 2/3\}{=}\begin{cases}1{-}(\frac{n-1}{n})^n, \text{ If } n{\leq}5;\\ 2/3, \text{ If } n{\geq}6;\end{cases}$.

VIII. Performance Evaluation

A. Performance Metrics

In order to verify the analysis results and evaluate the performance of the proposed algorithms, we conduct numerical simulations using Matlab 2015b on a computer with Win7-bit64, 2GHz CPU, and 4GB Memory. Three performance metrics are used: **Solution profit, Run time(s), and Try number**. The solution profit metric represents the profit of a solution. The run time metric measures the average time used to solve a RDP problem instance. The metric of try number measures the number of candidate solutions checked by an algorithm, which is used to evaluate the effectiveness of optimal algorithms in reducing search space. Here we compare our optimal algorithm OptGreDyn with two other optimal algorithms named as OptAll and OptDynLim. OptDynLimit utilizes the dynamic limiting technique introduced in Section VI-B. OptAll uses the exhaustive search approach.

Quite a few works have been conducted in the literature, where greedy based approach is widely used. These greedy algorithms are adapted to variants of the RDP problem with particular restrictions and assumptions. Most works assume that RSUs can only be deployed at road intersections [31] or middle roads [32]. Hence, for comparative evaluation on our approximate algorithms Greedy2P3 and Greedy2P3E, we add two typical previous algorithms into our simulation experiments, marked respectively as GreedyMiddle and BEP. GreedyMiddle is a representation of existing greedy based algorithms, which determines RSUs' positions greedily one by one, but only middle points of segments are considered as valid candidate positions. The BEP algorithm represents the Balloon Expansion Heuristic(BEH) algorithm proposed in [27], which is adapted to our one-dimensional problem here. In BEP, only segment ends are valid candidate positions for RSUs. BEP works as follows. At first, a RSU is assumed at each candidate positions, then this RSUs expand their coverage radius from zero to $d/2$ gradually. In this process, RSUs with overlapped coverages are replaced with one representative RSU selected from the RSUs. At last, the best n RSUs in term of profit form the final solution.

BEP first assumes that at The idea underlie BEP is very distinctive, so it is also included here.

B. Simulation Setup

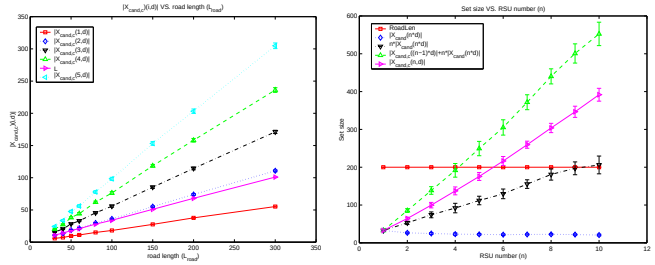
The profit density functions of the segments make difference only in the construction of $X_{\text{cand,c}}(n,d)$ and the profit of the strips in $X_{\text{cand,c}}(n,d)$. Later operations in an algorithm need only work on $X_{\text{cand,c}}(n,d)$, whereas the profit density functions are not needed any longer. The construction of X_{cand} is straightforward, so main results in our work need to be verified are on one-dimensional RDP problem with multiple identical RSUs. Hence, without deceasing the credibility of our experiments, we only perform simulations on cases where roads have piecewise constant functions.

Main parameters used for simulating our RDP problem with identical RSUs include (1) number of RSUs n, (2) strip length d, i.e., diameter of the circular coverage of the RSUs, (3) length of the whole road line L_{road}, (4) average length of a road segment L_{seg}, and (5) the upper limit $w_{\max}$ of the profit density function value of a road segment. A set of particular values for these parameters is called a simulation configuration. For each configuration, 100 problem instances are generated and then treated using the tested algorithms in turn. In each problem instance, round$(\frac{L_{\text{road}}}{L_{\text{seg}}})-1$ points are generated randomly in range $(0, L_{\text{road}})$ following uniform distribution. These points are used as segment end points, which divide the whole road line into $L{=}$round$(\frac{L_{\text{road}}}{L_{\text{seg}}})$ segments. These segments all have constant profit density functions, where the constants take integer values randomly from [0,$w_{\max}$] following uniform distribution. Data for the performance metrics are collected and averaged to obtain the final results for the configuration. Hence, each point in the figures in later subsections showing simulation results corresponds to 100 simulations. The 95% confidence intervals of the performance metrics are also obtained.

The effect of a parameter on an algorithm's performance is usually tested by performing a simulation set consisting of several simulation configurations. In these configurations, only the parameter to be tested take different values, whereas the other parameters usually take default values. In our simulation, the parameters' default values are $n{=}5$, $d{=}4$, $L_{\text{road}}{=}200$, $L_{\text{seg}}{=}3$, and $w_{\max}{=}10$.

C. Size of Candidate Strip Sets

It is obvious that $|X_{\text{cand,c}}(n,d)|$ has a great effect on the execution time of the algorithms. We first inspect the relation between $|X_{\text{cand,c}}(n,d)|$ and the parameters. We only provide



(a) (b)

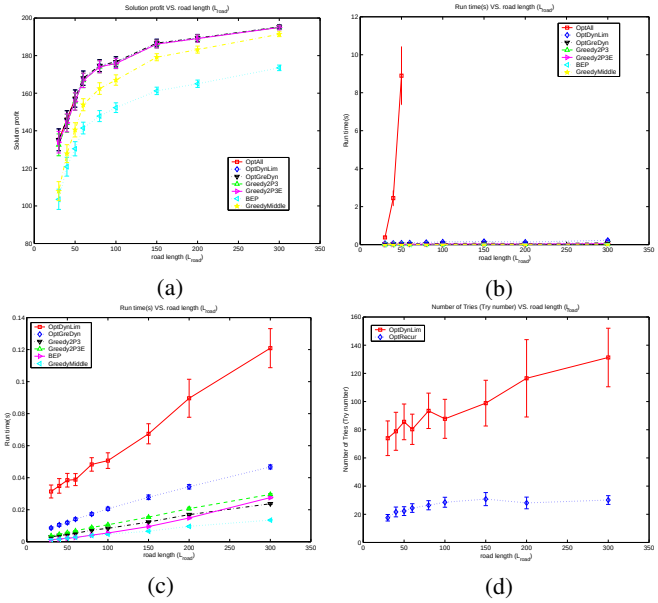
Fig. 4. The effects of L_{road} and n on the size of candidate strip sets.

the results for L_{road} and n in Fig.4. The results for other parameters show no new insights and are thus omitted for space limitation. Fig.4(a) shows the effect of L_{road}, where it takes values from the list $\{30, 40, 50, 60, 80, 100, 150, 200, 300\}$. Fig.4(b) shows the effect of n, where it takes values from 1 to 10. To get better understandings, some additional information are also collected. Fig.4(a) also show the results of $|X_{\text{cand,c}}(i,d)|$, $i \in \{1, \ldots, n\}$. In Fig.4(b), other data as indicated by the legends are shown. The results show that, $|X_{\text{cand,c}}(i,d)|$ increase both linearly along L and n, which consists with the rough analysis result of $|X_{\text{cand,c}}(n,d)| = \mathcal{O}(n*L)$ in Section IV-C.

D. Simulation Results

The results of the performance metrics in the previous experiment with different L_{road} are shown in Fig.5. Fig.5(a) shows that, profits of the solutions obtained by OptGreDyn, OptDynLim are both identical to that of OptAll, which confirm the optimality of OptGreDyn and OptDynLim. In the simulated cases, our two approximate algorithms Greedy2P3 and Greedy2P3E both return quasi-optimal solutions, with profit more than 98% and 99% of optimal solutions. Contrastively, GreedyMiddle and BEP are much less efficient. When L_{road}=30, profits of their solutions are both less than 80% of optimal solutions, and GreedyMiddle solutions are slightly better than those of BEP. As L_{road} increases, solutions of GreedyMiddle approach optimal solutions gradually, whereas the differences between optimal solutions and those of BEP show no distinctive change. Results on run time metric are shown in Fig.5(b). As run time of OptAll increases to sharply, we omit the tests on OptAll when $L_{\text{road}} \geq 60$. To show the results on the run time metric of other algorithms more clearly, we provide a new view in Fig.5(c), where the results of OptAll is removed. Compared with OptDynLim, OptGreDyn saves run time more than 50%. Compared with OptGreDyn, Greedy2P3, Greedy2P3E, BEP save runtime by more than 40%. Among the tested algorithms, GreedyMiddle is the most time efficient one, whose run time is only about 30% of OptGreDyn. Fig.5(d) shows the results on try number metric. The results show that, compared with OptDynLim, OptGreDyn reduces search space size by more than 75%. As a conclusion, OptGreDyn is an efficient optimal algorithm, and Greedy2P3 and Greedy2P3E are more preferable than BEP and GreedyMiddle when considering both solution profit and run time.

We have conduct other experiments for inspecting the



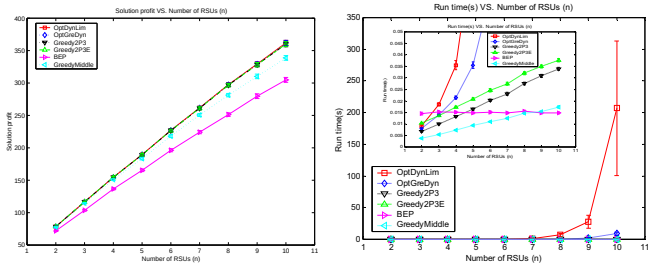
(a) (b) (c) (d)

Fig. 5. The effects of L_{road} on algorithms' performances, (a)solution profit, (b)run time, (c)run time of the algorithms without OptAll, (d)try number

effects of all other main parameters (n, d, L_{seg}, $W_{\max}$). In these experiments, OptAll is omitted since that the optimality of OptGreDyn and OptDynLim have been verified in the previous simulations. Simulation results all demonstrate that Greedy2P3 and Greedy2P3E usually return quasi-optimal solutions with profit more than 98% of optimal ones. Greedy2P3 and Greedy2P3E are preferable than BEP and GreedyMiddle. For space limitation, we only provide the results corresponding to parameter n in Fig.6, where n takes values from 2 to 10 with step size 1. The solution profit results in Fig.6(a) show that Greedy2P3 and Greedy2P3E both return nearly optimal solutions with profit more than 99% of optimal ones. GreedyMiddle's solutions are less optimal with profit less than 93% of optimal ones. Furthermore, as n increases, the difference between GreedyMiddle's solution profit and that of the optimal solutions increases gradually. In the test cases, solutions of BEP are always the worst one. Fig.6(b) shows the run time metric of the algorithms. As n increases, run time of both OptGreDyn and OptDynLim increases quickly, but the increasing speed of OptGreDyn is obviously slower than that of OptDynLim. Contrastively, run time of the approximate algorithms are trivial. To show the results for the approximate algorithms more clearly, a zoomed view is provided in Fig.6(b). Run time of Greedy2P3, Greedy2P3E, and GreedyMiddle all increase near linearly as n increases, which is reasonable considering how they work. BEP's run time does not vary much as n increases, which is reasonable since that its operation procedure doesn't rely on n heavily. The curves for the run time metric and the try number metric show similar trends, so the figure corresponding to the try number metric is omitted here.

IX. Conclusions

In this paper, we focus on the one-dimensional RDP problem with our new model. We firstly analyze the properties



(a) (b)

Fig. 6. The effect of n on performance, (a) solution profit, (b) run time.

of optimal solutions of the RDP problem. Then, suspecting that the one-dimensional RDP problem with n RSUs of different radii is intractable, we propose two greedy-based algorithms (named as Greedy2P3 and Greedy2P3E) and show that Greedy2P3E's approximation ratio is at least $1-(\frac{n-1}{n})^2$. And next, for the one-dimensional RDP problem with n RSUs of identical radii, we found that it can be transformed to a problem being a subset of the well-known maximum coverage problem, which is NP-hard. By exploiting the properties of optimal solutions of the original problem, we propose an optimal algorithm based on greedy idea and dynamic programming, so it is abbreviated as OptGreDyn. At last, we proved that, if applied to the one-dimensional RDP problem with n RSUs of identical radii, the approximation ratios of Greedy2P3 and Greedy2P3E are at least 2/3. Furthermore, Greedy2P3's approximation ratio of 2/3 is tight when $n=3*i$, here i is any positive integer. Numerical simulations validate the correctness of the analyses results. Simulation results verify the optimality of OptGreDyn, meanwhile show that Greedy2P3 and Greedy2P3E usually return near-optimal solutions with more than 99% of optimal solutions, and they are preferable than existing approximate algorithms.

Several open issues about the new RDP problem require further research efforts, such as problem complexity analysis to one-dimensional RDP problem and two RDP problem, tighter approximation ratio of better greedy algorithms including our Greedy2P3E. Efforts on better approximate algorithms for the new RDP problem with multiple heterogenous RSUs are also needed.

APPENDIX A
PROOF PREPARATIONS OF OF THEOREM 1

The whole process for proving the approximation ratio of Greedy2P3 can be roughly expressed in 5 steps as follows.

Step1: We totally identify 15 cell styles, and determine whether they are stable, unstable, invalid. (Refer to Appendix A-A).

Step2: Based on the properties of the cell styles, and using the fact that an unstable cell must join a group by connecting to stable cells through paths, we identify 8 group styles (Refer to Appendix A-B).

Step3: We make all unbalanced groups become balanced by using logic rearrangement operations. We identify 6 logic group styles (Refer to Appendix A-C).

Step4: For each logic group style, we prove its performance ratio(Refer to Appendix A-D).

Step5: We prove the approximation ratio of the Greedy2P3 algorithm by combining performance ratios of the logic group styles, and show that it is tight by providing some problem instances which can approach this approximation ratio.

In this section, we will provide detailed process for the first four steps in the following four subsections, respectively.

A. Cell Styles

Restricted to at most 3 strips, there are totaly 15 cell styles, as shown in Fig.7. In this figure, each sub-figure represents a cell style. Belts with lighter blue background represent road lines. Red brackets under road line represent optimal strips, whereas blue brackets over road line represent greedy strips. An arrow from strip a to strip b represents that $B(b)\geq B(a)$. The cell style in Fig.7(a) is denoted as CS_a, other cell styles are denoted similarly. Some cell styles looks similar, such as CS_{d1}, CS_{d2}, CS_{d3}, CS_{d4}, thus we can also regard them as sub-styles of a virtual super style CS_d.

For cell styles, we obtain properties in terms of stable, unstable, and invalid. Main properties are as follows.

***Property** 1:* If $B(cg_1)\neq B(co_1)$, then cell styles CS_a and CS_b can not co-exist with each other.

Proof: We prove it by contraction. For any greedy solution S_g and optimal solution S_o, we assume that some cells of style CS_a and CS_b co-exists. Then, for any greedy strip cg_1 of style CS_a and any optimal strip co_1 of style CS_b, we have the following two cases.

- Case: $B(cg_1)>B(co_1)$

In this case, by replacing co_1 in S_o with cg_1, we can obtain a new solution S_{o2}. It is obvious that $B(S_{o2})>B(S_o)$, so S_o must not be an optimal solution. This contradicts with the fact that solution S_o is optimal.

- Case: $B(cg_1)<B(co_1)$

In this case, cg_1 will not be selected by Greedy2P3 since that cg_1 is not the best free strip because that $B(cg_1)<B(co_1)$. This contradicts with the fact that cg_1 is a greedy strip.

Combining the above results, we complete the proof. ■

***Property** 2:* For the two cell styles CS_a and CS_b, If only CS_a exists, then any optimal strips must have profit not smaller than any of the CS_a strips. If only CS_b exists, then any greedy strips must have profit not smaller than any of the CS_b strips.

Proof: We only provide the proof for the case that only CS_a exists. The proof for the other case can be completed similarly. We prove it by contradiction. If only CS_a exists, we assume that there is an optimal strip os_1 that satisfies $B(os_1)<B(gs_1)$. In this case, we can replace os_1 in the optimal solution(denoted as S_1) with strip gs_1, we thus obtain a solution S_2 with $B(S_2)>B(S_1)$, which contradicts with the fact that solution S_1 is optimal. ■

***Property** 3:* If $B(cg_1)\neq B(co_1)$, then cell styles CS_{d1},CS_{d2},CS_{d3} and CS_{d4} are unstable.

Proof: Here we only provide the proof for cell style CS_{d1}. The results for the other three styles can be proved similarly.

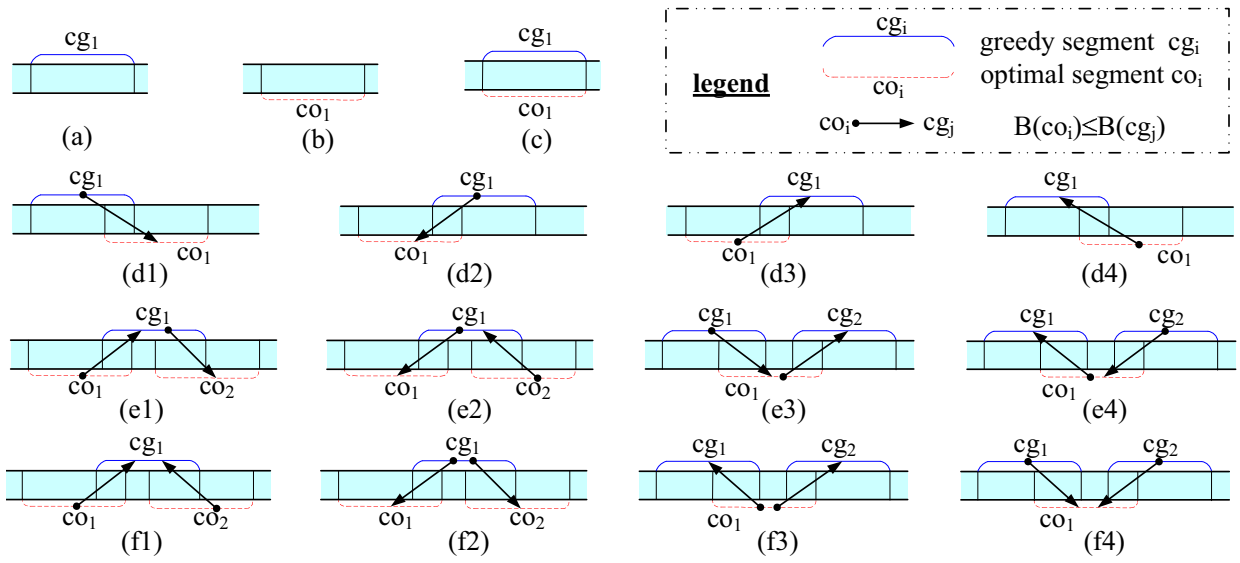


Fig. 7. Cell styles.

As shown in style CS_{d1} in Fig.7, we have $B(cg_1)\leq B(co_1)$. We will prove it with the help of Fig.8. If $B(cg_1)\neq B(co_1)$, then we have $B(cg_1)<B(co_1)$. Since that Greedy2P3 will select the maximum free strip at each time, the fact of $B(cg_1)<B(co_1)$ implies that strip co_1 must be connected with another greedy coverage, say cg_2, as shown in Fig 8(a), which prevents Greedy2P3 from selecting co_1. Otherwise, cg_1 will not be the maximum free strip at the time to select cg_1. We step further, if $B(co_1)<B(cg_2)$, then the strip set $\{cg_1, co_1, cg_2\}$ should also be connected to other strips. Otherwise, co_1 will not be an optimal strip since that replacing co_1 with cg_2 will obtain a better solution. Hence, there must be another optimal strip co_2 to which cg_2 directly connects. Such analysis process can continue until it meets one of the two cases: (1) $B(cg_i)\geq B(co_i)$, (2)$B(cg_{i+1})=B(co_i)$.

- Case $B(cg_i)\geq B(co_i)$:

In this case, the strip set $\{co_{i-1}, cg_i, co_i\}$ forms an instance of cell style CS_{f1}, which is stable as shown in Property 5.

- Case $B(cg_{i+1})=B(co_i)$:

In this case, the strip set $\{cg_{i+1}, co_i\}$ forms an instance of cell style CS_{d2} or CS_{d3}. They all are stable when $B(cg_{i+1})=B(co_i)$.

As a conclusion, cell style CS_{d1} is unstable when $B(cg_1)\neq B(co_1)$. Cells of style CS_{d1} can only exist by connecting to other stable cells. The property follows. ■

The following two properties can be proved similarly.

***Property** 4:* If $B(cg_1)$, $B(cg_2)$, $B(co_1)$, $B(co_2)$ are all different, then cell styles CS_{e1},CS_{e2},CS_{e3} and CS_{e4} are unstable.

***Property** 5:* If $B(cg_1)<B(co_1)$ or $B(cg_1)<B(co_2)$, then sell style CS_{f2} is unstable. If $B(co_1)<B(cg_1)$ or $B(co_1)<B(cg_2)$, then sell style CS_{f3} is unstable.

Cell style CS_{f4} has the following property.

***Property** 6:* If $B(cg_1)<B(co_1)$ and $B(cg_2)<B(co_1)$, then cell style CS_{f4} is invalid.

Proof: We prove it by contradiction. Without lost generality, we assume that Greedy2P3 selects cg_1 before cg_2. However, Greedy2P3 will actually not select cg_1 since that cg_1 is not the best free strip, as implied by the fact $B(co_1)>B(cg_1)$. Thus there is a contradiction. The property follows. ■

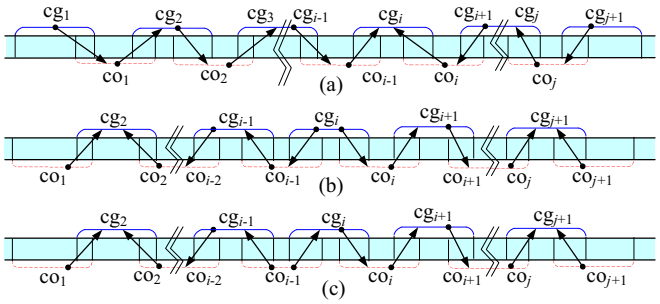


Fig. 8. Extension of unstable cells.

It is obvious that all other cell styles are stable. Furthermore, when the formulae represented by the arrows in these cell styles take equality, the cell styles are all stable. For example, if $B(cg_1)=B(co_1)$, then all cell styles CS_{di}, $i\in\{1,2,3,4\}$, are stable. If $B(cg_1)=B(co_2)$, then we can reverse the directions of the corresponding arrows in cell styles $CS_{e,1}$ and $CS_{e,2}$. These cell styles thus become cell styles CS_{f1} and CS_{f2}, respectively. Similarly, if $B(cg_1)=B(co_2)$, then cell styles $CS_{e,3}$ and $CS_{e,4}$ can be transformed to cell styles $CS_{f,3}$ and $CS_{f,4}$ by reversing the directions of the corresponding arrows.

B. Group Styles

Unstable cells can only exist actually by connecting to stable cells directly or indirectly. So, for any unstable cell, there must be one or two paths that connect the unstable cell to stable ones. For example, in Fig.8(a), the unstable cell $\{cg_1, co_1\}$ of style CS_{d1} connects to stable cell $\{co_{i-1}, cg_i, co_i\}$ of style CS_{f1} through a path. In fact, such paths are formed by unstable cells. Meanwhile, another unstable cell $\{co_j, cg_j{+}1\}$ of style CS_{d2} connects to the same stable cell at the right side. In Fig.8(b), an unstable cell $\{co_{i-1}, cg_i, co_i\}$ of style

CS_{f2} is connected to two stable cells of style CS_{f1} (they are $\{co_1, cg_2, co_2\}$ and $\{co_j, cg_{j+1}, co_{j+1}\}$) from the left side and right side, respectively. In Fig.8(c), the paths connecting an unstable cell $\{cg_{i-1}, co_{i-1}, cg_i\}$ of style CS_{f3} to two stable cells of style CS_{f1} (they are $\{co_1, cg_2, co_2\}$ and $\{co_j, cg_{j+1}, co_{j+1}\}$) are shown.

Thus, through paths, cells are grouped into strip groups. There are totally eight group styles, as shown in Fig.9. The group style in Fig.9(a) is denoted as GS_a. Other group styles are denoted similarly. For expression brevity, the latter five group styles from GS_d to GS_h are illustrated using abbreviated symbols, where red bars in the lower layers represent optimal strips, blue bars in the upper layers represent greedy strips. A bi-directional arrow represents an arrow with either direction. A line without arrow ends represents the only case that the two strips have exactly the same profit. A part marked with three continuous small circles means that the part can happen zero or more times.

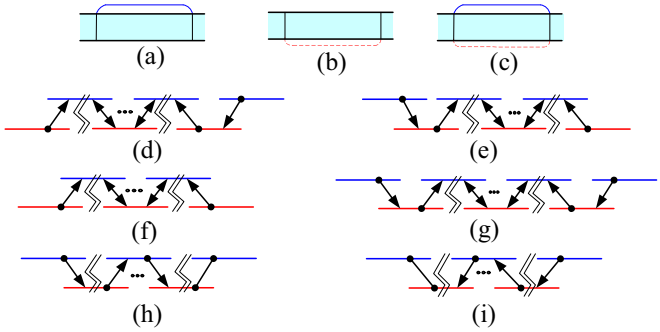


Fig. 9. Group styles.

Group styles are abstract representations of many strip groups that can actually exist. For example, group style GS_c can represent groups formed by cell style CS_c, or groups formed by stable cells of styles CS_{di} when $B(cg_1)$=$B(co_1)$, $i\in\{1,2,3,4\}$. The group in Fig.8(a) is an instance of group style GS_g. The groups in Fig.8(b) and Fig.8(c) are both instances of group style GS_f. As in the proof of Property 3, if $B(co_{i-1})$=$B(cg_i)$, the strip set $\{cg_1, co_1, \cdots, cg_{i-1}, co_{i-1}, cg_i\}$ forms a group of style GS_h.

C. Logic Group Styles

To facilitate performance analysis, balanced groups are preferred. For unbalanced group styles of GS_a, GS_b, GS_f, GS_g, GS_h and GS_i, we can make them become balanced by logically rearranging some strips among different unbalanced groups. In the logic rearrangement operation, we do not move strips physically, instead, we just logically regard the strips as a member of the other group where it is logically inserted. Hence, final balanced groups are called as logic groups. Logic groups are be categorized into totally five logic group styles as shown in Fig.2. These logic group styles are denoted as LGS_a, LGS_b, LGS_c, LGS_d, and LGS_f, respectively.

The whole process of making all unbalanced groups become balanced is called as **logic rearrangement process**. The whole process can be roughly described as three steps. This process is demonstrated in Fig.10, where in group of styles GS_g, GS_h and GS_i, the strips marked with pentagons are the strips to be split off.

Step1: Logically split off one greedy strip from each group of styles GS_g, GS_h, GS_i, and collect all these strips together with possible GS_a strips into a set S. These groups are thus become balanced logic groups of style LGS_d, LGS_e, LGS_f, respectively.

Step2: For each group of style GS_f, we randomly select a strip from the set S and logically connect it to the rightmost optimal strip in the group, thus we obtain logic groups of style LGS_b. .

Step3: If groups of style GS_b exist, then some strips will be left in set S. In this case, the number of the strips must be identical to that of GS_b groups. So we can match GS_b groups and strips in S into pairs. According to Property 1, in each pair, the greedy strip must have profit not smaller than that of the optimal strip. Thus, each pair will be a logic group of style LGS_a. If groups of style GS_b do not exist, then set S after Step2 must become empty.

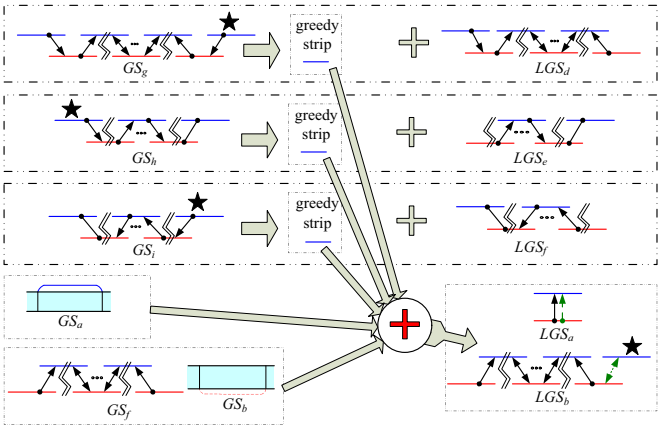


Fig. 10. Logic rearrangement operation.

After the logic rearrangement process, all groups will be changed to logic groups, and they are all balanced groups.

D. Performance Ratio of Greedy Algorithm on Logic Groups

In this section, we will analyze the performance ratios of Greedy2P3 respect to the logical groups. We use $\beta_{\text{LGS}}(i)$ to denote the performance ratio of Greedy2P3 on logic groups of style LGS_i, $i\in\{a,b,c,d,e,f\}$. It is obvious that $\beta_{\text{LGS}}(a)\geq 1$. The following lemmas show that $\beta_{\text{LGS}}(b)\geq 2/3$, $\beta_{\text{LGS}}(c)\geq 2/3$, $\beta_{\text{LGS}}(d)\geq 2/3$, $\beta_{\text{LGS}}(e)\geq 1$ and $\beta_{\text{LGS}}(f)\geq 1$.

***Lemma** 10:* For any logical group of style LGS_b,LGS_c and LGS_d, there must be $\beta_{\text{LGS}}(b)\geq 2/3$, $\beta_{\text{LGS}}(c)\geq 2/3$ and $\beta_{\text{LGS}}(d)\geq 2/3$.

Proof: We only provide the proof for style LGS_b. The proof for the other two styles can be completed similarly. Without loss of generality, we assume that the number of greedy strips and that of the optimal strips in a logic group of style LGS_b is m. The detailed structure of the strip group is shown in Fig.11, where the ends of the strips divides the

road section into slices. For expression brevity, we roughly group the slices into $m+1$ sections, and each section contains 4 slices tagged with a, b, c and d, respectively. The profit of slice j in section i is denoted as $x_{i,j}$. For example, $x_{2,b}$ represents the profit of the slice b in section 2. This example is general enough to represent cases where all slices have different lengths. It can also represent cases where strips are adjacent. For example, $x_{i,b}=0$ can represent the case that two greedy strips are adjacent.

The total profit of the greedy strips in the logical group, denoted as B_{G}, can be calculated as Eq.(21). Here G to represent the profit of the rightmost greedy strip, which is logically connected to the rightmost optimal strip.

$$B_{\mathrm{G}}=G+\sum_{i=1}^{m-1}(x_{i,c}+x_{i,d}+x_{i+1,a}) \tag{21}$$

Similarly, the total profit of the optimal strips in the logical group, denoted as B_{Opt}, can be calculated as Eq.(22).

$$B_{\mathrm{Opt}}=\sum_{i=1}^{m}(x_{i,a}+x_{i,b}+x_{i,c}) \tag{22}$$

Simple mathematical transformations show that, the task of proving that $\beta_{\mathrm{LGS}}(b)\geq 2/3$ is equivalent to prove Eq.(23).

$$\begin{aligned} 2(x_{1,a}+x_{m,c}+\textstyle\sum_{i=1}^{m}x_{i,b})\leq\ & \textstyle\sum_{i=1}^{m-1}(x_{i,c}+x_{i+1,a}) \\ & +3(G+\textstyle\sum_{i=1}^{m-1}x_{i,d}) \end{aligned} \tag{23}$$

No matter what directions are taken by the bi-directional arrows in the structural diagram of LGS_b, we have Eq.(24), which can be checked easily. For example, when $i=2$, Eq.(24a) represents $x_{2,b}\leq(x_{2,d}+x_{3,a})$, which must be true. Otherwise, we have $B(cg'_2)\geq x_{2,b}+x_{2,c}>x_{2,d}+x_{3,a}+x_{2,c}=B(cg_2)$. Thus, Greedy2P3 will select strip cg'_2 rather than cg_2. Since that all strips have the same length, strip cg'_2 must be able to cover the two slices of $x_{2,b}$ and $x_{2,c}$ completely.

$$x_{i,b}\leq x_{i,d}+x_{i+1,a},\quad 2\leq i\leq m-1 \tag{24a}$$

$$x_{i,b}\leq x_{i-1,c}+x_{i-1,d},\quad 2\leq i\leq m-1 \tag{24b}$$

According to the meaning of the leftmost arrow in Fig.11, we have Eq.(25).

$$x_{1,a}+x_{1,b}\leq x_{1,d}+x_{2,a} \tag{25}$$

For the rightmost arrow, we must have Eq.(26), otherwise strip cg'_m will be selected by Greedy2P3 instead of the logically inserted strip with profit G.

$$x_{m,b}+x_{m,c}\leq G \tag{26}$$

Furthermore, we must also have Eq.(27), otherwise strip cg'_0 will be selected by Greedy2P3 instead of the logically inserted one with profit G.

$$x_{1,a}+x_{1,b}\leq G \tag{27}$$

Accumulating both sides of the Equations in (24), Eq.(25), Eq.(27), and 2 times of Eq.(26), we obtain Eq.(28).

$$\begin{aligned} 2(x_{1,a}+x_{m,c}+\sum_{i=1}^{m}x_{i,b})\leq & (\sum_{i=1}^{m-2}x_{i,c})+(\sum_{i=2}^{m}x_{i,a}) \\ & +x_{m-1,d}+2(\sum_{i=2}^{m-1}x_{i,d})+3G \end{aligned} \tag{28}$$

Defining $f(x)$ as Eq.(29), we have $f(x)\geq 0$ because that all variables in right side of $f(x)$ are non-negative.

$$f(x)\overset{\text{def}}{=}x_{m-1,c}+x_{m-1,d}+\sum_{i=1}^{m-1}x_{i,d} \tag{29}$$

Adding $f(x)\geq 0$ to Eq.(28), we obtain Eq.(23). The Lemma follows. ■

Lemma 11: For logic groups of style LGS_e and LGS_f, we have $\beta_{\mathrm{LGS}}(e)\geq 1$ and $\beta_{\mathrm{LGS}}(f)\geq 1$.

Proof: We only provide the proof for LGS_e. The case for LGS_f can be proved similarly. Without loss of generality, we assume that there are m greedy strips $\{gs_1, gs_2, \cdots, gs_m\}$ and m optimal strips $\{os_1, os_2, \cdots, os_m\}$ in a logic group of style LGS_e. According to the meanings of the arrows in LGS_e, we have $B(gs_i)\geq B(os_i)$, $i\in\{1, 2, \cdots, m\}$. Hence, $\beta_{\mathrm{LGS}}(e)$ can be calculated as Eq.(30). The Lemma follows. ■

$$\beta_{\mathrm{LGS}}(e)=\frac{\sum_{i=1}^{m}B(gs_i)}{\sum_{i=1}^{m}B(os_i)}\geq\frac{\sum_{i=1}^{m}B(os_i)}{\sum_{i=1}^{m}B(os_i)}=1 \tag{30}$$

ACKNOWLEDGMENT

This research was funded by Natural Science Foundation of China (No.61671169).

REFERENCES

[1] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, "Vehicular networking: a survey and tutorial on requirements, architectures, challenges, standards and solutions," *IEEE Communications Surveys & Tutorials*, vol.13, no.4, pp.584-616, 2011.

[2] H. Cheng, X. Fei, A. Boukerche, and M. Almulla, "GeoCover: an efficient sparse coverage protocol for RSU deployment over urban VANETs," *Ad Hoc Networks*, vol.34, no.24, pp.85-102, 2015.

[3] D. Kim, Y. Velasco, W. Wang, R. N. Uma, R. Hussain, and S. Lee, "A new comprehensive RSU installation strategy for cost-efficient VANET deployment," *IEEE Transactions on Vehicular Technology*, vol.66, no.5, pp.4200-4211, 2017.

[4] L. X. Xue, Y. C. Yang, and D. C. Dong, "Roadside infrastructure planning scheme for the urban vehicular networks," *Transportation Research Procedia*, vol.25, pp.1380-1396, 2017.

[5] J. Barrachina, P. Garrido, M. Fogue, F. J. Martinez, J. C. Cano, C. T. Calafate, and P. Manzoni, "Road side unit deployment: a density-based approach," *IEEE Intelligent Transportation Systems Magazine*, vol.5, no.3, pp.30-39, 2013.

[6] H. Liu, X. Chu, Y. W. Leung, and R. Du, "Minimum-Cost sensor placement for required lifetime in wireless sensor-target surveillance networks," *IEEE Transactions on Parallel and Distributed Systems*, vol.24, no.9, pp.1783-1796, 2013.

[7] A. Abdrabou, and W. H. Zhuang, "Probabilistic delay control and road side unit placement for vehicular ad hoc networks with disrupted connectivity," *IEEE Journal on Selected Areas in Communications*, vol.29, no.1, pp.129-139, 2011.

[8] P. Li, C. H. Huang, and Q. Liu, "BCDP: budget constrained and delay-bounded placement for hybrid roadside units in vehicular ad hoc networks," *Sensors*, vol.14, no.12, pp.22564-22594, 2014.

[9] C. Y. Liu, H. J. Huang, and H. W. Du, "Optimal RSUs deployment with delay bound along highways in VANET," *Journal of Combinatorial Optimization*, vol.33, no.4, pp.1168-1182, 2017.

[10] J. Chi, J. Yeongwon, P. Hyunsun, H. Taehyeon, and P. Soyoung, "An effective RSU allocation strategy for maximizing vehicular network connectivity," *International Journal of Control and Automation*, vol.6, no.4, pp.259-270, 2013.

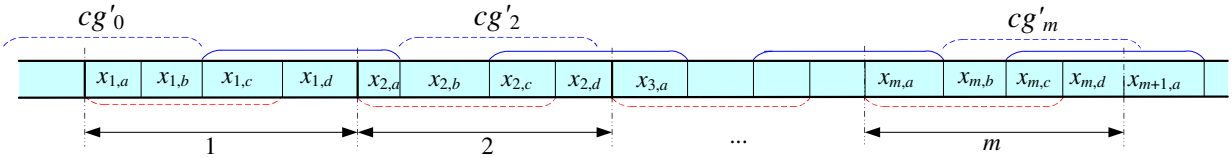


Fig. 11. An example logical group of style LGS_b.

[11] M. F. Faraj, J. F. M. Sarubbi, C. M. Silva, and F. V. C. Martins, "A hybrid genetic algorithm for deploying RSUs in VANETs based on inter-contact time," in *Proc. Genetic and Evolutionary Computation Conference Companion(GECCO17)*, 2017, pp.193-194.

[12] C. C. Lin, P. C. Chen, and L. W. Chang, "On different-dimensional deployment problems of hybrid VANET-sensor networks with QoS considerations," *Mobile Networks & Applications*, vol.22, no.1, pp.125-138, 2017.

[13] Y. Xu, Y. Wu, J. Xu, and L. Sun, "Efficient detection scheme for urban traffic congestion using buses," in *Proc. 26th Int. Conf. Adv. Infor. Netw. Appl. Workshops*, 2012, pp.287-293.

[14] Y. Huang, X. Guan, Z.Cai, and T. Ohtsuki, "Multicast capacity analysis for social-proximity urban bus-assisted VANETs," in *Proc. IEEE Int. Conf. Commun.*, 2013, pp.6138-6142.

[15] Z. G. Gao, D. J. Chen, N. M. Yao, Z. M. Lu, B. C. Chen, "A novel problem model and solution scheme for roadside unit deployment problem in VANETs," *Wireless Personal Communications*, Avaliable: https://doi.org/10.1007/s11277-017-4888-6.

[16] Wikipedia, "Geometric set cover problem," https://en.wikipedia.org/wiki/Geometric_set_cover_problem,2017.

[17] Wikipedia, "Maximum coverage problem," https://en.wikipedia.org/wiki/Maximum_coverage_problem,2017.

[18] T.-J. Wu, W. J. Liao, and C.-J. Chang, "A cost-effective strategy for road-side unit placement in vehicular networks," *IEEE Transactions on Communications*, vol.60, no.8, pp.2295-2303, August 2012.

[19] Y. Kim, S. Park, and J. Chi, "Absorbing markov chain-based roadside units deployment," *Contemporary Engineering Sciences*, vol.9, no.12, pp.579-586, 2016.

[20] J. Whang, S. Park, and J. Chi, "Influence maximized MCL basd RSU deployment," *International Journal of Future Generation Communication and Networking*, vol.9, no.11, pp.229-238, 2016.

[21] I. Filippini, F. Malandrino, G. Dán, M. Cesana, C. Casetti, and I. Marsh, "Non-cooperative RSU deployment in vehicular networks," in *Proc. 9th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, 9-11, Jan.2012, pp.79-82.

[22] Z. Z. Zheng, P. Sinha, and S. Kumar, "Alpha coverage: bounding the interconnection gap for vehicular internet access," in *Proc. IEEE International Conference on Computer Communications (INFOCOM)*, 2009, pp.2831-2835.

[23] J. Lee, and C. M. Kim, "A roadside unit placement scheme for vehicular telematics networks," in *Proc. Advances in Computer Science and Information Technology (LNCS.6059)*, 2010, pp.196-202.

[24] T. C. Oscar, M. Fiore, and J. M. Barcelo-Ordinas, "Cooperative download in vehicular environments," *IEEE Transactions on Mobile Computing*, vol.11, no.4, pp.663-678, 2012.

[25] C. M. Silva,L. L. A. Andre, and J. W. Meira, "Deployment of roadside units based on partial mobility information," *Computer Communications*, vol.60, pp. 28-39, 2015.

[26] Y. P. Sun, R. X. Lu, X. D. Lin, J. Su, and X. M. Shen, "Roadside units deployment for efficient short-time certificate updating in VANETs", in *Proc. IEEE International Conference on Communications*, 2010, pp.1-5.

[27] B. Aslam, F. Amjad, and C. C. Zou, "Optimal roadside units placement in urban areas for vehicular networks," in *Proc. IEEE Symposium on Computers and Communications*, 2012, pp.423-429.

[28] R. J. Fowler, M. S. Paterson, and S. L. Tanimoto, "Optimal packing and covering in the plane are NP-complete," *Information Processing Letters*, vol.12, no.3, pp.133-137, 1981.

[29] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions-I,", *Mathematical Programming*, vol.14, pp.265-294, 1978.

[30] Wikipedia, "Independent set (graph theory)," https://en.wikipedia.org/wiki/ Independent_set_(graph_theory),2017.

[31] O. Trullols, M. Fiore, C. Casetti, C. F. Chiasserini, and J. M. B. Ordinas, "Planning roadside infrastructure for information dissemination in intelligent transportation systems," *Computer Communications*, vol.33, no.4, pp.432-442, 2010.

[32] M. Kafsi, P. Papadimitratos, O. Dousse, T. Alpcan, and J. P. Hubaux, "VANET connectivity analysis," in *Proc. IEEE Workshop on Automotive Networking and Applications*, 2008. pp.1-10.

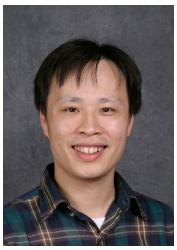
Zhenguo Gao Ph.D., Professor. He was now a Professor in Huaqiao University, Xiamen, China. He has been a visiting scholar in University of Illinois at Urbana-Champaign and University of Michigan in 2010 and 2011. He received his BS and MS degree in Mechanical and Electrical Engineering from Harbin Institute of Technology, Harbin, China, in 1999 and 2001, respectively. Then he received his Ph.D. degree in Computer Architecture from Harbin Institute of Technology, Harbin, China, in 2006. His research interests include wireless ad hoc network, cognitive radio network, network coding.

He is a senior member of China Computer Federation. He received National Science Foundation Career Award of China in 2007 and Outstanding Junior Faculty Award of Harbin Engineering University in 2008. He is severing as a reviewer for project proposals to National science foundation of China, Ministry of Education of China. He is also serving as a reviewer for some refereed Journals including IEEE/ACM Transactions on Networking, IEEE Transactions on Mobile Computing, etc

Danjie Chen was now a lecturer in Huaqiao University, Xiamen, China. She received her M.S. degree in Computer Software from Beijing University of Technology. Her research interests include cooperative communication, cognitive radio networks.

PLACE PHOTO HERE

Shaobin Cai was now a Professor in Huaqiao University. He was a Post Doctor of computer department, UCLA, Member of China Computer Federation. His primary research interests include ad hoc networks, wireless sensor network, and underwater acoustic sensor network.



Hsiao-Chun Wu (M'00-SM'05-F'15) received a B. S. E. E. degree from National Cheng Kung University, Taiwan, in 1990, and the M. S. and Ph. D. degrees in electrical and computer engineering from University of Florida, Gainesville, in 1993 and 1999 respectively. From March 1999 to January 2001, he had worked for Motorola Personal Communications Sector Research Labs as a Senior Electrical Engineer. Since January 2001, he has joined the faculty in Department of Electrical and Computer Engineering, Louisiana State University, Baton Rouge, Louisiana, USA. From July to August 2007, Dr. Wu had been a visiting assistant professor at Television and Networks Transmission Group, Communications Research Centre, Ottawa, Canada. From August to December 2008, he was a visiting associate professor at Department of Electrical Engineering, Stanford University, California, USA.

Dr. Wu has published more than 120 peer-refereed technical journal and conference articles in electrical and computer engineering. His research interests include the areas of wireless communications and signal processing. He currently serves as an Associate Editor for IEEE Transactions on Broadcasting, IEEE Signal Processing Letters, IEEE Communications Magazine, etc. He used to serve as an Associate Editor for IEEE Transactions on Vehicular Technology. He has also served for numerous textbooks, IEEE/ACM conferences and journals as the technical committee, symposium chair, track chair, or the reviewer in signal processing, communications, circuits and computers.