An Auction Based Task Dispatching and Pricing Mechanism in Bike-sharing

Bing Shi (bingshi@whut.edu.cn)
Wuhan University of Technology
Yaping Deng
Wuhan University of Technology
Han Yuan
Wuhan University of Technology

Research Article

Keywords: Bike-sharing, Mechanism design, Task dispatching, Pricing

Posted Date: April 26th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-261715/v1

Version of Record: A version of this preprint was published at Knowledge-Based Systems on January 1st, 2022. See the published version at https://doi.org/10.1016/j.knosys.2021.107631.

Noname manuscript No.
(will be inserted by the editor)

An Auction Based Task Dispatching and Pricing Mechanism in Bike-sharing

Bing Shi · Yaping Deng · Han Yuan

Received: date / Accepted: date

Abstract As a green and low-carbon transportation way, bike-sharing provides lots of convenience in the daily life. However, the daily usage of sharing bikes results in dispatching problems, i.e. dispatching bikes to the specific destinations. The bike-sharing platform can hire and pay to workers in order to incentivize them to accomplish the dispatching tasks. However, there exist multiple workers competing for the dispatching tasks, and they may strategically report their task accomplishing costs (which are private information only known by themselves) in order to make more profits, which may result in inefficient task dispatching results. In this paper, we first design a dispatching algorithm named GDY-MAX to allocate tasks to workers, which can achieve good performance. However it cannot prevent workers strategically misreporting their task accomplishing costs. Regarding this issue, we further design a strategy proof mechanism under the budget constraint, which consists of a task dispatching algorithm and a worker pricing algorithm. We theoretically prove that our mechanism can satisfy the properties of incentive compatibility, individual rationality and budget balance. Furthermore we run extensive experiments to evaluate our mechanism based on a dataset from Mobike. The results show that the performance of the proposed strategy proof mechanism and GDY-MAX is similar to the optimal algorithm in terms of the coverage ratio of accomplished task regions and the sum of task region values, and our

Bing Shi
Wuhan University of Technology, Wuhan, 430070, China
E-mail: bingshi@whut.edu.cn

Yaping Deng
Wuhan University of Technology, Wuhan, 430070, China

Han Yuan
Wuhan University of Technology, Wuhan, 430070, China

mechanism has better performance than the uniform algorithm in terms of the total payment and the unit cost value.

Keywords Bike-sharing · Mechanism design · Task dispatching · Pricing

1 Introduction

Bike-sharing as a low-carbon travelling way, can solve the problem of "the last mile" in the public transportation system. There exist many bike-sharing platforms, such as HelloBike, Mobike and so on, where HelloBike has provided bike-sharing services in more than 460 cities in China. However, the bike dispatching problem emerges with the flourishment of the bike-sharing market. For example, bikes can be parked anywhere when riders reach their destinations, and then other riders may not find available bikes in the nearby area. The platform needs to dispatch these bikes regularly to specific locations in order to satisfy riders' demands (e.g. dispatching bikes to the exit of subway station). However, the bikes are usually distributed anywhere within a specific geographical area, and thus it is inconvenient and costive to use trucks to convey them. Actually, in the real world, some riders might be willingly to accomplish these dispatching tasks in their spare time to make money, and thus the bike-sharing platform can pay to incentivize these riders (i.e. workers) to dispatch bikes to specific locations. In this paper, we intend to analyze how the bike-sharing platform hires workers to dispatch bikes efficiently.

In more detail, the bike dispatching tasks in different ares may have different values. For example, the value of dispatching bikes to the subway station is high since potentially it can satisfy plenty of riders' demands. Furthermore, the platform usually has a budget constraint for paying to workers who accomplish the tasks. In this context, the bike-sharing platform needs to hire workers to dispatch bikes efficiently given the budget constraint in order to maximize the total values of dispatching tasks. Similar to some existing works, we consider that the platform adopts an auction based mechanism to allocate dispatching tasks to workers. In this mechanism, the platform divides the overall area into a number of small regions. In each region, there exist a number of bikes to be dispatched. Workers bid for the dispatching tasks by reporting their accomplishing costs of dispatching all bikes in that region, i.e. instead of bidding for dispatching an individual bike, workers bid for a region, which includes a number of bikes to be dispatched. The platform then determines how to match workers with regions, and determines the payments to the workers in order to maximize the total values of all regions. In this situation, it may happen that multiple workers compete with each other for the bike dispatching regions in order to make profits, and they may untruthfully report their costs of accomplishing tasks in order to make more profits. This may result in inefficient task dispatching results for the platform. Therefore, how to design an strategy proof[1] auction mechanism to allocate dispatching tasks and pay

[1] Strategy proof means that workers have no incentives to misreport their private information, i.e. satisfying incentive compatibility (workers can maximize their own profits by

to workers under the budget constraint to maximize the total accomplishing values of regions is a key issue.

Specifically, this paper advances the state of art in the following ways. In order to maximize the region values given the budget constraint, we first design a task dispatching algorithm, named GDY-MAX, which consists of GDY operation and MAX operation. The experimental results show that this task dispatching algorithm can achieve good performance. However, GDY-MAX cannot meet strategy proof. Then according to Myerson's theorem [2], we further design a strategy proof mechanism where the task dispatching algorithm is monotonous and the worker pricing algorithm is based on critical price. In more detail, we use linear programming to improve the classical greedy algorithm by introducing Max operation in the submodule of maximization problem, and then design the task dispatching algorithm satisfying the monotonicity. In addition, we design the worker pricing algorithm with the budget constraint based on the critical price. We theoretically prove that this mechanism can satisfy the properties of incentive compatibility, individual rationality and budget constraint. Furthermore, we run extensive experiments to evaluate GDY-MAX and the strategy proof mechanism based on a Mobike dataset. The results show that the performance of the mechanism and GDY-MAX is similar to the optimal algorithm in terms of the coverage ratio of accomplished task regions and the sum of task region values, and our mechanism has better performance than the uniform algorithm in terms of the total payment and the unit cost value.

The rest of the paper is structured as follows. We describe the related work in Section 2. We then introduce the basic settings in Section 3. In Section 4, we introduce GDY-MAX dispatching algorithm and the strategy proof mechanism. We run experiments to evaluate our methods in Section 5, and conclude the paper in Section 6.

2 Related work

There exist plenty of works on analyzing bike-sharing systems. For example, in [3], the authors use a survey based approach to improve the bike-sharing system to reduce the traffic congestion. In [7], a framework integrating artificial immune system(AIS) and the artificial neural network forecasting technique is developed to predict the bike rental demands, and in [10], the authors propose a hierarchical prediction model that predicts the number of rents/returns to each cluster in a future period to achieve redistribution. Furthermore, a data mining method is proposed to predict the hourly rental demand of bikes, which can help to provide stable bicycle rental supply for the public transportation service [4]. In [5], the authors propose a network optimization approach by considering multiple factors to increase the quality and efficiency of the bike sharing service. In [6], a Distributed RL (DiRL) approach with reinforcement

truthfully reporting their information) and individual rationality (workers' profits should be non-negative) [1].

learning and transfer learning is proposed to dynamically satisfy users' demand at the minimum cost.

How to dispatch bikes is a key issue in the bike-sharing system. For example, a multi-objective optimization method is proposed to reposition bikes over times [8]. In [9], the balancing problem of bike stations is investigated. In [11], a deep reinforcement learning based framework is proposed to incentivize users to rebalance the bike distribution and thus improve service levels. In [12], an approach integrating multiobjective optimization with artificial bee colony algorithm is used to reposition the bikes over time such that enough bikes and open parking slots are available to users. Furthermore, hiring users to accomplish the bike dispatching tasks can be regarded as a typical crowdsourcing issue, where auction based methods are widely used to allocate tasks to users efficiently. For example, in [13], an auction model under a crowdsourcing framework is proposed to allocate bike relocation tasks to users in order to balance the bike supply and demand.

However, in such crowdsourcing related issues, workers may untruthfully report their information to make more profits. In order to prevent strategic behavior of reporting information, mechanism design [2] is introduced to ensure workers report information truthfully. In [14], an incentive mechanism called TruPreTar is proposed to incentivize users to park bicycles at locations desired by the platform toward rebalancing bike supply and demand. In [15], the authors introduce a reverse auction framework to model the interactions between the platform and the smartphone users who perform sensing tasks. A mechanism called TRAC is proposed which can guarantee that the submitted bids of users reflect their real costs of performing sensing tasks. In [16], an asynchronous and distributed task selection (ADTS) algorithm is proposed to help the smartphone users plan their task selections on their own. In [17], a mobile crowdsensing framework is proposed to selects workers who are more likely to provide reliable data, and compensates their costs for both sensing and privacy leakage. In [18], a Lyapunov-based VCG auction mechanism is proposed to increase users' engagement in the crowdsourcing tasks.

However, these works do not consider the budget constraint in when hiring workers to accomplish the crowdsourcing task. Actually, to the best of our knowledge, existing works usually do not consider how to prevent strategy behavior of workers when bidding to dispatch bikes in the bike-sharing system, or do not consider the budget constraint. In this paper, we intend to design a strategy proof mechanism to solve the task dispatching issue in order to maximize the value of task regions.

3 Basic Settings

In this section, we introduce the basic settings of the bike-sharing platform, workers and tasks, and provide the problem definition.

As we have discussed previously, we divide the overall area into several small regions. The bike-sharing platform will regularly publish the dispatch-

ing tasks of each region. Instead of bidding for the specific dispatching task, workers bid for the specific region according to their current locations. Then the platform determines the matching between the regions and workers, and computes the payment to workers. The worker matched with the specific region needs to dispatch all unsorted bikes in that region to the destinations, and then gets the payment from the platform. Note that different regions may have different values, such as that the region of the subway station may have higher values. In this paper, we intend to maximize the sum of region values given the budget constraint.

3.1 Task Region

In this paper, we divide the overall area D into k small regular hexagon regions which do not cross and overlap with each other, i.e. $D = \{D_1, D_2, \ldots, D_k\}$. The task regions are represented by a set $\mathcal{T} = \{t_1, t_2, \ldots, t_k\}$, where $t_j = (l_j, v_j)$, l_j is the centric location of that region, and v_j is the value of that region.

3.2 Workers

Workers bid for task regions, and then accomplish all dispatching tasks in that matched region. We use $M = \{m_1, m_2, \ldots, m_n\}$ to represent the set of workers, and use pos_i to represent the location of worker m_i. The task region which can be accomplished by the worker depends on the geographical region D_{m_i} and the worker's activity radius rad_i. Now the set of task regions accomplished by worker i is $\mathcal{T}_i = \{t_j \mid \mathrm{dis}\,(pos_i, l_j) \leq rad_i, t_j \in \mathcal{T}\}$.

Each worker has a cost for accomplishing the dispatching tasks of that region, consisting of the time cost and energy cost, which is denoted by $c_i = \alpha \times H + \beta \times S$, where H represents the total time to complete the task, S represents the total walking distance, α and β are time and distance parameters respectively. For worker $m_i \in M$, its expected utility u_i is defined as the payment p_i received from the platform minus the cost c_i, i.e. $\mu_i = p_i - c_i$.

When the platform publishes the task information, workers compete with each other to bid for the task region. The bid submitted by worker m_i is denoted as $b_i = (\mathcal{T}_i, c_i)$, including the set of regions $\mathcal{T}_i$ that can be done by the worker and the total cost c_i. The bids of all workers are denoted as $b = (b_1, b_2, \ldots, b_M)$. Note that the worker may lie about cost c_i (i.e. its bid may be different from the true cost) in order to make more profits.

3.3 Problem Definition

In this paper, the platform needs to match regions with workers and make the payment in order to maximize the total values of all accomplished task regions given the budget constraint. We use $\mathcal{M}$ to represent the set of workers who win the regions, and then the allocated regions are denoted as the set

$\mathcal{T}_{\mathcal{M}} = \cup_{m_i \in \mathcal{M}} \mathcal{T}_i$. Note that the payment should be greater than the worker's cost, and the total payment should not be greater than the budget constraint B. We use $p = \{p_1, p_2, \ldots, p_M\}$ to represent the payments to all workers. We intend to maximize the overall region values, which is defined as follows.

Definition 1 (Bike Task Value Maximization Problem) Given the set of task regions $\mathcal{T}$, the set of workers M and the set of bids b, the platform will dispatch task regions to workers $\mathcal{M} \subseteq M$ (i.e. task dispatching algorithm) and make the payment to workers (i.e. pricing algorithm) under the budget constraint B, in order to maximize the sum of values $V(\mathcal{M}) = \sum_{t_j \in \mathcal{T}_{\mathcal{M}}} v_j$, i.e:

$$\max : V(\mathcal{M}), \text{ s.t. } \forall m_i \in \mathcal{M}, \sum_{\mathrm{m_i} \in \mathcal{M}} p_i \leq B$$

This problem is a typical set cover problem [19], which is an NP-hard Problem. Therefore an approximation algorithm is needed. We adopt a submodular function in this paper.

Definition 2 (Submodular functions) N is a finite set. Function $f : 2^N \mapsto R$ is a submodular function if and only if $S_1 \subseteq S_2 \subseteq N$ and $x \in N \backslash S_2$:

$$f(S_1 \cup \{x\}) - f(S_1) \geq f(S_2 \cup \{x\}) - f(S_2) \tag{1}$$

According to the definition of submodular function, for any set of workers $M_1 \subseteq M_2 \subseteq M$ and worker $m_i \in M \backslash M_2$, $M_1 \subseteq M_2$, there is $V(M_1) \leq V(M_2)$. The value is:

$$V(M_1 \cup \{m_i\}) - V(M_1) = \sum_{t_j \in \mathcal{T}_i \cup_{m_k \in M_1} \mathcal{T}_k} v_j \tag{2}$$

Because of the diminished marginal effect, we have:

$$\sum_{t_j \in \mathcal{T}_i \cup_{m_k \in M_1} \mathcal{T}_k} v_j \geq \sum_{t_j \in \mathcal{T}_i \backslash \cup_{m_k \in M_1} \mathcal{T}_k} v_j \tag{3}$$

According to Equations 2 and 3, we now have:

$$V(M_1 \cup \{m_i\}) - V(M_1) \geq \sum_{t_j \in \mathcal{T}_i \backslash \cup_{m_k \in M_1} \mathcal{T}_k} v_j = V(M_2 \cup \{m_i\}) - V(M_2) \tag{4}$$

4 Task Dispatching and Pricing

In this section, we first describe the task dispatching algorithm GDY-MAX, and then introduce the strategy proof mechanism which ensure workers bid truthfully.

4.1 Task Dispatching Algorithm (GDY-MAX)

To solve the task dispatching issue, we first design a greedy based dispatching algorithm. The reason of doing this is that the strategy proof mechanism is based on this algorithm. In addition, we also find that this algorithm can achieve good performance in the experimental analysis. We call it as **GDY-MAX**, which includes GDY operation and MAX operation.

In more details, GDY operation chooses workers with low accomplishing cost to complete high-value task regions. We use set M_k to represent winning workers. When adding worker m_i into set M_k, the marginal contribution is $f_i = f_{i|M_k} = V(M_k \cup \{m_i\}) - V(M_k)$. We select the worker with the largest unit marginal contribution $m_{i*} \leftarrow \text{argmax}_{m_i \in M \setminus M_k} \frac{f_i}{c_i}$ until all workers are selected, or the payments to workers reach the budget constraint, i.e. $c_i* \leq \frac{B \cdot f_i^*}{2V(M_k \cup \{m_{i^*}\})}$.

Furthermore, we use MAX operation to choose the region having the largest value, i.e. $m^* \leftarrow \text{argmax}_{m_i \in M} V(\{m_i\})$. We then select the largest set of task regions by comparing $V(M_k)$ with $V(\{m^*\})$, where $\{m^*\}$ is the set of workers who can get the largest sum value of task regions.

Although the MAX operation overcomes the deficiency on the approximation of GDY operation, **GDY-MAX** algorithm is not monotonous. For example, there is a worker $m_i \in M_k$ with cost c_i. **GDY-MAX** algorithm can generate the set of workers M_k. Let worker m_i report its cost as $c^{'}$ and $c^{'} < c_i$. According to the unit margin contribution, when the cost of worker m_i is lower than a certain value, the set of workers selected by **GDY-MAX** algorithm could be $\{m^*\}$, or $\{m_i\}$. Therefore, **GDY-MAX** algorithm does not satisfy monotonicity. Therefore, according to Myerson's theorem [2], it is not strategy proof.

4.2 A Strategy Proof Mechanism

In this section, we intend to design a strategy proof mechanism (named **BMT VMP**) under the budget constraint to maximize the value of task regions while ensuring workers bid truthfully. According to Myerson's theorem, the task dispatching algorithm in the mechanism needs to satisfy monotonicity, and the worker pricing algorithm should be based on the critical price.

4.2.1 Task Dispatching

We adopt linear rounding [20] to address the monotonic issue. First, we compute $V(\{m^*\})$, and then compute the values of integer programming solution, named V. The set with the largest values will be selected. The task dispatching algorithm can guarantee monotonicity with only a slight sacrifice of performance. In more details, for the linear rounding process of **BMTVMP** in the integer programming, **BMTVMP**$(\frac{B}{2}, M^-)$ means that variable x_i (which

shows whether the worker is selected by the platform as the winning worker) changes from a discrete value 0,1 to continuous value [0,1] with the budget constraint changed to $B/2$. To solve the linear programming problem, worker m^* with the largest region value is eliminated since the final linear programming solution needs to be compared with worker m^*. Workers with task accomplishing cost greater than $B/2$ should be excluded. Those workers are denoted as a set $M_{\frac{B}{2}}$. Then the rest of workers is expressed as $M^- \triangleq M \backslash \left(\{m^*\} \cup M_{\frac{B}{2}}\right)$. Finally, the optimal solution of linear programming can be denoted as GDY-LP-MAX$\left(\frac{B}{2}, M^-\right)$. Similarly, the optimal solution of the integer programming is expressed as GDY-IP-MAX$\left(\frac{B}{2}, M^-\right)$.

Similar to **GDY-MAX** algorithm, **BMTVMP** $\left(\frac{B}{2}, M^-\right)$ is also based on greedy method. However, **BMTVMP** $\left(\frac{B}{2}, M^-\right)$ has different MAX operation. The task dispatching algorithm compares the maximum region value with the value of the optimal linear programming solution GDY-LP-MAX$\left(\frac{B}{2}, M^-\right)$. Instead of comparing values directly, we set a parameter $\theta = \frac{6e^2}{(e-1)^2}$, which is related to the approximation ratio. The reason of setting such a parameter value is as follows. From the degree of similarity between GDY-LP-MAX$\left(\frac{B}{2}, M^-\right)$ and $V(M_k)$, we have:

$$\text{GDY-IP-MAX}\left(\frac{B}{2}, M^-\right) \leq \frac{3e}{e-1} \max\left\{V\left(M_k\right), V\left(m^*\right)\right\} \tag{5}$$

According to pipage rounding [21] in the linear programming, we have:

$$\text{GDY-LP-MAX}\left(\frac{B}{2}, M^-\right) \leq \frac{2e}{(e-1)} \text{GDY-IP-MAX}\left(\frac{B}{2}, M^-\right) \tag{6}$$

Based on Equations 5 and 6, we have:

$$\text{GDY-LP-MAX}\left(\frac{B}{2}, M^-\right) \leq \frac{6e^2}{(e-1)^2} \max\left\{V\left(M_k\right), V\left(m^*\right)\right\} \tag{7}$$

Therefore, the parameter θ is set to $\frac{6e^2}{(e-1)^2}$.

Finally, the task dispatching algorithm of **BMTVMP** mechanism is shown in Algorithm 1.

In Algorithm 1, lines 3 to 6 indicate how to sort workers based on the marginal contributions. Then lines 8 to 11 compare $V(m^*)$ with the optimal solution of linear programming GDY-LP-MAX$\left(\frac{B}{2}, M^-\right)$. When GDY-LP-MAX$\left(\frac{B}{2}, M^-\right) < \theta \times V\left(m^*\right)$, we choose $\{m^*\}$ as the winning worker, otherwise, we select M_k as the winner. The time complexity of this algorithm is $O(|M|^2)$.

Lemma 1 *The task dispatching algorithm of **BMTVMP** mechanism satisfies monotonicity.*

Proof From lines 8 and 11 of Algorithm 1, we can see that there are two possible outputs. We now discuss them in turn.

Algorithm 1: Task dispatching algorithm of BMTVMP

Input: Set of workers M; Set of task regions $\mathcal{T}$; Budget constraint B; Set of bids b
Output: Set of winning workers M^*

1 $M_k \leftarrow \phi$;
2 $m^* \leftarrow \text{argmax}_{m_i \in M} V(\{m_i\})$, $m_{i^*} \leftarrow \text{argmax}_{m_i \in M} \frac{f_i}{c_i}$;
3 **while** $M \backslash M_k \neq \phi$ and $c_{i^*} \leq \frac{(B \cdot f_{i^*})}{2V(M_k \cup \{m_{i^*}\})}$ **do**
4 $M_k \leftarrow M_k \cup \{m_{i^*}\}$;
5 $m_{i*} \leftarrow \text{argmax}_{m_i \in M \backslash M_k} \frac{f_i}{c_i}$;
6 **end**
7 $M^- \leftarrow M \backslash (\{m^*\} \cup M_B)$;
8 **if** GDY-LP-MAX$\left(\frac{B}{2}, M^-\right) > \frac{6e^2}{(e-1)^2} \times V(\{m^*\})$ **then**
9 $M^* \leftarrow M_k$;
10 **else**
11 $M^* \leftarrow \{m^*\}$;
12 **return** M^*;

- If the output is $M^* = \{m^*\}$, it shows that GDY-LP-MAX$\left(\frac{B}{2}, M^-\right) \leq \theta \times V(m^*)$. In this case, the cost of worker m^* accomplishing the task only needs to satisfy the budget constraint B. Therefore, the bid of worker m^* cannot affect the final result, which means that this worker will always be selected as the winner.
- If the output is $M^* = M_k$, we have GDY-LP-MAX $\left(\frac{B}{2}, M^-\right) > \theta \times V(m^*)$. Assuming that there is no change in the task accomplishing cost for the remaining workers other than worker m_i, worker $m_i \in M_k$ will submit new bid where the task cost will be reduced from c_i to $c_{i'}$. In this case, the value of GDY-LP-MAX$(\frac{B}{2}, M^-)$ will be increased due to the reduced cost of worker m_i, satisfying the statement in line 8. For the marginal contribution, since the task cost is reduced to c_i', the position of worker m_i will be advanced in the overall workers. We assume that the new position is j and $j \leq i$. Because of the decreased marginal effects, we have $f_{i|M_{i-1}} \leq f_{i|M_{j-1}}$. The sum of values is, however, increased with the increased workers, which can be expressed as $V(M_{i-1} \cup \{m_i\}) \geq V(M_{j-1} \cup \{m_i\})$. In combination with budget constraint, we have

$$c_i' < c_i \leq B \times \frac{f_{i|M_{i-1}}}{V(M_{i-1} \cup \{m_i\})} \leq B \times \frac{f_{i|M_{j-1}}}{V(M_{j-1} \cup \{m_i\})} \tag{8}$$

Based on the above analysis, if the original winning worker reduces the task accomplishing cost, it still satisfies the budget constraint and the marginal contribution condition, and thus is selected as the winning worker. In summary, the task dispatching algorithm in **BMTVMP** mechanism satisfies monotonicity.

4.2.2 Pricing

According to Algorithm 1, there will be two possible outputs. When the final result is worker m^* with the maximum value, the platform can take the entire

budget as its payment. Otherwise, the platform needs to calculate the payment to each winning worker separately and ensure that the total payment does not exceed the budget constraint B.

Definition 3 (Critical Cost) p is the worker's critical task cost. If the task accomplishing cost submitted by the worker is greater than its critical cost p, the worker will not be assigned to the task by the platform, otherwise it will be assigned.

When the payment to worker m_i is calculated, we can remove m_i from the set of sorted winning workers. Then, the payment that worker m_i receives at each position is calculated separately. Obviously, the payments are different for workers at different positions. We then choose the largest price as the payment to worker m_i. This method calculates the payment p_i as the critical cost of worker $m_i \in M^*$. Note that the payment to unassigned workers is set to 0, while the payment to worker m^* with the maximum value is set to B. In the case of multiple winning workers, we first remove worker m_i, and get the remaining workers $M^{-i} = M \backslash m_i$. According to the greedy method, we compute the payment which can defeat worker from first to $k^{-i}+1$ position, denoted as $c^{-i}_{i(j)}$. $c^{-i}_{i(j)}$ has two limitations. First, the marginal contribution of worker m_i's unit cost is greater than m_j's unit cost:

$$\frac{f^{-i}_i}{c^{-i}_{i(j)}} \geq \frac{f^{-i}_j}{c_j} \Rightarrow c^{-i}_{i(j)} \leq \frac{f^{-i}_i \times c_j}{f^{-i}_j} = \varphi_{i(j)} \tag{9}$$

Second, $c^{-i}_{i(j)}$ should not be greater than budget constraint B:

$$c^{-i}_{i(j)} \leq B \times \frac{f^{-i}_i}{V\left(M^{-i}_{j-1} \cup \{m_i\}\right)} = \lambda_{i(j)} \tag{10}$$

From Equations 9 and 10, we know that $c^{-i}_{i(j)} \leq \min\left\{\varphi_{i(j)}, \lambda_{i(j)}\right\}$. The pricing algorithm use the smallest value between $\varphi_{i(j)}$ and $\lambda_{i(j)}$ as the payment $p^{-i}_{i(j)}$ to worker m_i. When the position index j increases, f^{-i}_j is monotonically increased, and $\frac{c_j}{f^{-i}_j}$ is monotonically decreased according to the submodularity of the value function. Therefore, we can select the maximum value of $p^{-i}_{i(j)}$ and set it as the payment to worker m_i.

$$p_i = \max_{1 \leq j \leq (k'+1)} p^{-i}_{i(j)} \tag{11}$$

The pricing algorithm is shown in Algorithm 2. Lines 2 to 3 mean that worker m^* has the maximum task region value and the total budget B is the worker's payment. Lines 6 and 7 sort other workers by marginal contribution when removing m_i and put them into the set M^{-i}. Then, lines 8 to 13 use the new sorted sequence to calculate the possible payment p_i for worker m_i, and then select the maximum value from all possible values as the payment to m_i. The time complexity of Algorithm 2 is $O(|M|^3)$.

Algorithm 2: Worker pricing algorithm of BMTVMP

Input: Set of workers M; Set of task regions $\mathcal{T}$; Budget constraint B; Set of bids b; Set of winning workers M^*
Output: Payment p

$p \leftarrow 0$;
if $M^* = \{m^*\}$ **then**
  $p \leftarrow B$;
return p;
for $m_i \in M^*$ **do**
  $M^{-i} \leftarrow M \backslash \{m_i\}, j \leftarrow 1, M_{j-1}^{-i} \leftarrow \emptyset, p_i \leftarrow c_i$;
  $m_j \leftarrow \text{argmax}_{m_t \in M^{-i}} \frac{f_t^{-i}}{c_t}$;
  while $M^{-i} \backslash M_{j-1}^{-i} \neq \varnothing$ and $c_j \leq \frac{B \times f_j^{-i}}{V\left(M_{j-1}^{-i} \cup (m_i\}\right)}$ **do**
    $\varphi_{i(j)} \leftarrow \frac{f_i^{-i} \times c_j}{f_j^{-i}}, \lambda_{i(j)} \leftarrow \frac{B \times f_j^{-i}}{V\left(M_{j-1}^{-i} \cup \{m_i\}\right)}, p_i \leftarrow \max\left\{p_i, \min\left\{\varphi_{i(j)}, \lambda_{i(j)}\right\}\right\}$;
    $M_j^{-i} \leftarrow M_{j-1}^{-i} \cup \{m_j\}, j \leftarrow j+1, m_j \leftarrow \text{argmax}_{m_t \in M^{-i} \backslash M'_{j-1}} \frac{f_t^{-i}}{c_t}$;
  end
  $\varphi_{i(j)} \leftarrow \frac{f_i^{-i} \times c_j}{f_j^{-i}}, \lambda_{i(j)} \leftarrow \frac{B \times f_i^{-i}}{V\left(M_{j-1}^{-i} \cup \{m_i\}\right)}$;
  $p \leftarrow \max\left\{p_i, \min\left\{\varphi_{i(j)}, \lambda_{i(j)}\right\}\right\}$;
end
return p;

Lemma 2 *The payment calculated by the pricing algorithm meets the critical cost requirement.*

Proof The proof can be shown in two cases:

- After worker m_i submits a new bid, where the task accomplishing cost of m_i is less than the critical cost p_i, $c_i \leq p_i = p_{i(r)}^{-i}$, we have $c_i \leq \theta_{i(r)}$ and $c_i \leq \lambda_{i(r)}$. The worker can be assigned to the task at position r.
- After worker m_i submits a new bid, where the task accomplishing cost is greater than or equal to the critical cost p_i, the platform will no longer select the worker as the winner. We discuss it in the following two cases.
  1) $\theta_{i(r)} \leq \lambda_{i(r)}$, we get $p_{i(r)}^{-i} = \theta_{i(r)}$ from $p_i \leftarrow min\{\theta_{i(j)}, \lambda_{i(j)}\}$. When m_i's task cost is not less than p_i, $c_i > \theta_{i(r)}$ can be obtained. The cost of m_i increases while the marginal contribution remains, which means the marginal contribution per unit cost of worker m_i is smaller than the marginal contribution per unit cost of worker m_r. At the same time, we sort the workers by marginal contribution of unit cost. Let $j \in \left[r+1, k^{-i}+1\right]$ be below position r, we have $\theta_{i(r)} \geq \theta_{i(j)}$. Then we have $c_i > \theta_{i(r)} > \theta_{i(j)}$, and m_i will not be selected as the winner. Suppose that at position j, we have $\varphi_{i(j)} > p_{i(r)}^{-i} = \varphi_{i(r)} > p_{i(j)}^{-i} = \lambda_{i(j)}$ from $\theta_{i(r)} < \theta_{i(j)}$. Since the cost of worker m_i has increased, we have $c_i > \theta_{i(r)} > \lambda_{i(j)}$, and it means that the budget constraint is not satisfied and m_i is not selected as the winner.
  2) $\theta_{i(r)} > \lambda_{i(r)}$, $p_{i(r)}^{-i} = \lambda_{i(r)}$ can be deduced from $p_i \leftarrow min\{\theta_{i(j)}, \lambda_{i(j)}\}$. Assuming that position $j \in [0, k^{-i}+1]$, then $\lambda_{i(r)}$ is greater than $\lambda_{i(j)}$.

Worker m_i's cost increases $c_i > p_{i(r)}^{-i} = \lambda_{i(r)} > \lambda_{i(j)}$, and it will not be selected as the winner because of the budget constraint. Assuming that $\lambda_{i(r)} < \lambda_{i(j)}$, we can get $\lambda_{i(j)} > p_{i(r)}^{-i} = \lambda_{i(r)} > p_{i(j)}^{-i} = \varphi_{i(j)}$. Furthermore, we have $c_i > p_{i(r)}^{-i} = \lambda_{i(r)} > \varphi_{i(j)}$, it has conflict with $\theta_{i(r)} > \lambda_{i(r)}$, m_i cannot be selected.

In conclusion, the payment p_i calculated by the worker pricing algorithm is the critical cost of m_i.

4.2.3 Theoretical analysis

In this section, we theoretically prove that our mechanism can satisfy incentive compatibility, individual rationality and budget constraint.

Theorem 1 ***BMTVMP*** *mechanism satisfies incentive compatibility under budget constraint.*

Proof From Lemma 1 and Lemma 2, we know that the task dispatching algorithm is monotonous and the payment is the critical cost of the worker. According to Myerson's theorem, **BMTVMP** mechanism satisfies incentive compatibility.

Theorem 2 ***BMTVMP*** *mechanism satisfies individual rationality under budget constraint.*

Proof In the worker pricing algorithm (Algorithm 2), the worker who does not participate in the dispatching task is paid zero. According to $p_i \leftarrow max\{p_i, min\{\theta_{i(j)}, \lambda_{i(j)}\}\}$, we know that the payment to m_i is $p_i \geq p_{i(j)}^{-i}$. Bids submitted by other workers are unchanged when calculating the remuneration of m_i. In this case, other winning workers will remain unchanged except m_i, which can be represented by the set $M_{i-1} = M_{j-1}^{-i}$. Worker m_j can win by eliminating m_i. In the list that does not exclude m_i, worker m_i is selected by the platform as the winner, which indicates that m_i's task cost meets the budget constraint, and it can be inferred that:

$$c_i \leq \frac{B \times f_{i|M_{i-1}}}{2V\left(M_{i-1} \cup \{i\}\right)} = \frac{B \times f_{i|M_{j-1}}^{-i}}{2V\left(M_{j-1}^{-i} \cup \{i\}\right)} = \lambda_{i(j)} \tag{12}$$

The list is sorted by the marginal contribution of unit cost, and the position of worker m_i precedes that of worker m_j:

$$\frac{f_{i|M_{i-1}}}{c_i} \geq \frac{f_{j|M_{i-1}}}{c_j} \Rightarrow c_i \leq \frac{f_{i|M_{j-1}^{-i}}^{-i} \times c_j}{f_{j|M_{i-1}}^{-i}} = \varphi_{i(j)} \tag{13}$$

From Equations 12 and 13, we have the task accomplishing cost of worker m_i:

$$c_i \leq \min\left\{\varphi_{i(j)}, \lambda_{i(j)}\right\} \leq p_{i(j)}^{-i} \leq p_i \tag{14}$$

The above equations indicates that the task accomplishing cost of worker m_i is less than or equal to payment p_i. Then the utility of worker m_i is greater than or equal to zero. To sum up, in either cases, the utility obtained by the worker is non-negative. Therefore, **BMTVMP** mechanism satisfies individual rationality.

Theorem 3 *Given the dispatching task assigned to worker $m_i \in M^*$, the payment p_i limited to $\frac{f_i}{V(M^*)\times B}$, the total payment calculated by **BMTVMP** mechanism meets the budget constraint.*

Proof We assume that the output of the task dispatching algorithm is $M^* = \{m^*\}$. Now the worker's pricing algorithm sets B as the payment, which meets the budget constraint. In another case, the output of the task dispatching algorithm is $M^* = M_k$. Assuming that the worker pricing algorithm gives $m_i \in M_k$ a large payment, which satisfies $p_i > \frac{f_i}{V(M^*)\times B}$. Let position $j \in [0, k^{-i}+1]$, the equation $r = argmax p_{i(j)}^{-i}$ is established. As the payment p_i is limited by the marginal contribution of unit cost and the budget, we have:

$$p_i \leq \frac{f_{i|M_{r-1}^{-i}}^{-i} \times c_r}{f_{r|M_{r-1}^{-i}}^{-i}} = \frac{B \times f_{i|M_{r-1}^{-i}}^{-i}}{2V\left(M_{r-1}^{-i} \cup \{i\}\right)} \tag{15}$$

According to individual rationality, the utility obtained by worker is non-negative, and $c_i \leq p_i$. Because worker m_i is not selected within the former $(i-1)$ positions, when $0 \leq j \leq (i-1)$, we have $p_{i(j)}^{-i} < c_i$. Then we have $p_{i(j)}^{-i} < p_{i(r)}^{-i}$. Furthermore, $M_{i-1} \in M_{r-1}^{-i}$, when $M_{r-1}^{-i} \cup \{m_i\} = M_{r-1}^{-i} \cup M_k$, it can be concluded that:

$$\frac{f_{i|M_{i-1}}}{p_i} \geq \frac{f_{i|M_{r-1}}^{-i}}{p_i} \geq \frac{2V\left(M_{r-1}^{-i} \cup \{m_i\}\right)}{B} = \frac{2V\left(M_{r-1}^{-i} \cup M_k\right)}{B} \geq \frac{V(M_k)}{B} \tag{16}$$

From Equation 16, it can be obtained that $p_i \leq \frac{f_i}{V(M^*)\times B}$. The original assumption is not true. Therefore, the payment p_i to worker $m_i \in M_k$ meets $p_i \leq \frac{f_i}{V(M^*)\times B}$. We have that:

$$\sum_{m_i \in M^*} p_i \leq \frac{B \times \Sigma_{m_i \in M^*} f_i}{V(M^*)} = B \tag{17}$$

Therefore, the total payment of **BMTVMP** mechanism will not exceed the budget constraint.

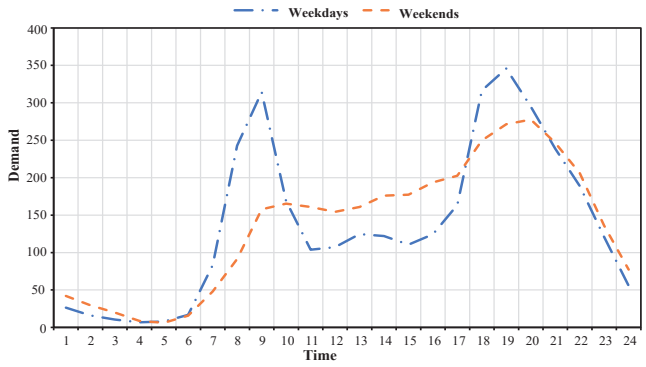

Fig. 1 Demand of weekdays and weekends

5 Experimental Analysis

We now run experiments to evaluate our methods. In the experiment, we use a Mobike dataset including worker tracks in Shanghai from August 1 to September 1, 2016. Each data record contains order ID, bike ID, worker ID, start time, start location (longitude and latitude), end time, end location, and trace. The total amount of orders is 102361. In the experiment, we assume that the bikes needs to be dispatched periodically.

5.1 Experimental Settings

We first pre-process all order data and distinguish them into weekdays and weekends, as shown in Figure 1. We can see that there is a huge difference between the number of orders on weekdays and weekends. In order to make a reliable and consistent experimental results, we exclude the order data of weekends, and run the experiments on the order data of weekdays.

Fig. 2 Task dispatching diagram

Table 1 Experimental Parameters

Parameters	*Description*
Average number of bikes	from 1 to 100
Workers' moving radius	1.5, 2, 2, 3, 3.5
The unit price	[0.01, 0.1]
Basic price	[10, 20]
The number of workers	from 50 to 600
Budget constraints	from 500 to 2000

We set the experimental area as a two kilometers by two kilometers square, divide it into a number of small regions. By looking into the dataset, the number of bikes in a region ranges from 1 to 100. We categorize the regions into 5 levels, and each level corresponds to different unit cost information. The bike-sharing platform sets different values for tasks. The task value of the bustling regions is higher, which is shown in red, followed by yellow and green on the map. When there are fewer workers and the region is faraway from city centre, the region has low values, which is shown in blue, as shown in Figure 2. Specifically, the region value is equal to the number of bikes multiplied by the average time of accomplishing tasks. We then use a scaling factor 0.1 to compute the final value. The worker's radius of activity is divided into 5 levels, namely 1.5, 2, 2, 3 and 3.5 kilometers. We set 2 twice because there exist plenty of regions with 50 to 100 bikes. The higher the bike density is, the smaller the worker's radius of activity is. The worker cost is calculated by the basic price plus the total price. The total price is obtained by multiplying the unit price with the number of bikes. Since the value factor 0.1 is set, in order to make cost value at the same magnitude, the unit price is randomly selected from [0.01, 0.1] and the basic price is set [10, 20] at random.

In the experiment, we randomly select different workers from the dataset at each time, and set the number of workers between 50 and 600 with step size 50. For each worker, the set of accomplishing task region depends on radius, and the cost of task accomplishment is uniformly distributed in the range. The experiment sets the budget between 500 and 2000 with step size 250. The experimental parameters are shown in Table 1.

Furthermore, we evaluate our mechanism and GDY-MAX dispatching algorithm against two benchmark algorithms.

UNIFORM[22]: It sets the probability of 2/5 to assign the single worker with the maximum value, and the probability of 3/5 to choose the workers according to greedy algorithm. This probabilistic selection method enables the UNIFORM algorithm to have a constant approximation ratio in the worst environment. The comparison between **UNIFORM** algorithm and **BMTVMP** mechanism can intuitively shows the influence of payments on the maximization of task region values.

OPTIMAL: The optimal algorithm of the set coverage problem can obtain the optimal solution for the value maximization problem. Therefore **OPTI-**

MAL algorithm is selected. However, it cannot be used in real life since the computation is heavy and cannot guarantee strategy proof.

We evaluate **BMTVMP** and **GDY-MAX** against the above two algorithms in terms of the following metrics.

- The coverage ratio of accomplished task regions, which is the proportion of the regions completed by workers to the total number of regions.
- The sum of task region values, which is the accumulation of values of accomplished task regions.
- The total payment, which is paid to workers by the platform.
- The unit cost value, which is the ratio of sum of task region values to the total payment of platform.

5.2 Experimental Results

The experiments are repeated for 100 times, and we compute the average results for the analysis. We now discuss the experimental results in detail.

The coverage ratio of accomplished task regions: The coverage ratio of the four algorithms are shown in Figure 3, where the number of workers is set to 300, and the corresponding budget constraint is set from 500 to 2000. From Figure 3, we can see that the coverage ratios of the accomplished task regions of **GDY-MAX**, **OPTIMAL** algorithm and **BMTVMP** mechanism increase with the increased budget since the platform is able to hire more workers to complete more tasks, and thus increasing the regional coverage.

In general, three algorithms can achieve more than 70% coverage given the high budget except **UNIFORM** algorithm. Furthermore, the performance of **GDY-MAX** algorithm is very similar to that of **OPTIMAL** algorithm. In contrast, the task coverage of **BMTVMP** mechanism is slightly lower, but the ratio of coverage growth is stable. Note that the **UNIFORM** algorithm shows that the coverage tends to decline when the budget increases. It may be caused by the random selection of winning workers with a certain probability. Workers with high cost are selected to spend more budget and the amount of accomplished task regions is reduced, thus resulting in the decreased coverage ratio. On the other hand, we find that when the budget increases to a high value, **BMTVMP** mechanism is still unable to cover all regions. This is because the workers' locations are randomly selected and thus cannot cover all regions.

Figure 4 shows the coverage ratio of four algorithms when the budget is set to 1000 and the number of workers is set from 50 to 600. The experimental results show that the overall coverage of **GDY-MAX** algorithm, **OPTIMAL** algorithm and **BMTVMP** mechanism increase with the increased number of workers. Compared with Figure 3, the coverage ratio increases faster, implying that the number of workers has a greater impacts on the coverage ratio than the budget.

The sum of the task region values: Figure 5 shows the sum of task region values with budgets from 500 to 2000 when the number of workers is

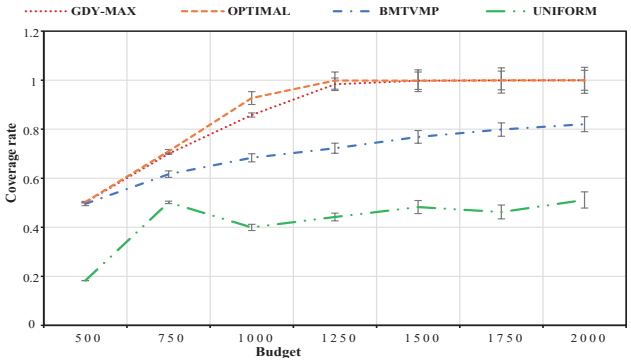

Fig. 3 The coverage ratio when workers fixed at 300

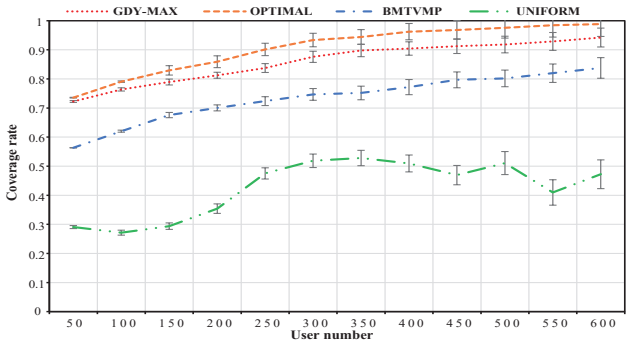

Fig. 4 The coverage ratio when budget fixed at 1000

300. We can see that when the budget increases, the sum of task region values increase for all algorithms. The reason is that when the budget increases, bike-sharing platform can hire more workers to complete more tasks. When the budget exceeds 1000, **GDY-MAX** and **OPTIMAL** algorithms can almost complete all tasks and obtain high values compared with other algorithms. Note that when the budget reaches a relatively large value, **BMTVMP** mechanism can not only prevents the strategic behavior, but also achieves near-optimal performance.

Figure 6 shows the sum of task region values when the budget is 1000 and the number of workers varies between 50 and 600. Task values obtained by those four algorithms go up with the increased workers. When the number of workers increases, bike-sharing platform can select workers with lower cost to complete tasks, but still win the same value. We can see that **BMTVMP** mechanism can achieve similar performance to the optimal algorithm. Although **BMTVMP** mechanism has a loss of small amount of values, it can prevent strategic behavior.

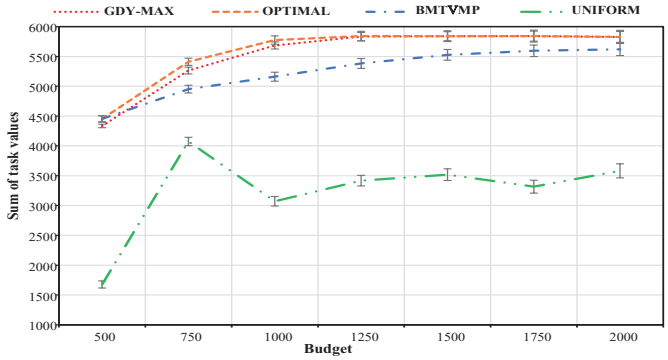

Fig. 5 The sum of task region value when workers fixed at 300

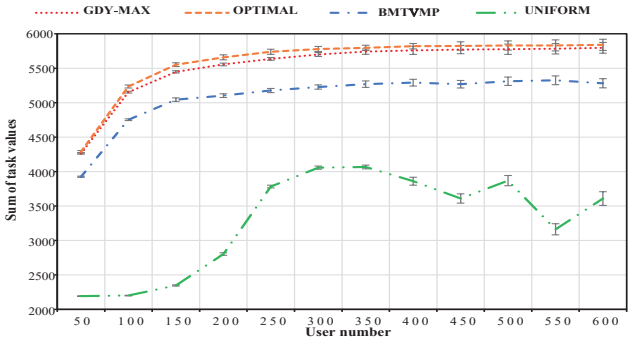

Fig. 6 The sum of task region value when budget fixed at 1000

The total payment: Since **GDY-MAX** algorithm and **OPTIMAL** algorithm do no include the pricing algorithm, we only evaluate the total payment of **BMTVMP** mechanism against **UNIFORM** algorithm. The number of workers is fixed at 300 and the budget ranges from 500 to 2000. The results are shown in Figure 7. We can see that the total payment increases with the increased budget. When the number of workers is large and the budget is increased, the bike-sharing platform can use more budget to hire more workers to complete tasks. The total payment of **BMTVMP** mechanism is less than **UNIFORM** algorithm, which implies that **BMTVMP** mechanism can save budget and pay to workers more reasonably.

Next, the budget is fixed at 1000 and the number of workers is set between 50 and 600. The total payments of the bike-sharing platform in **BMTVMP** mechanism and **UNIFORM** are shown in Figure 8. The results show that the total payments of **BMTVMP** mechanism and **UNIFORM** algorithm decrease with the increased workers. The reason is that when the budget is fixed, the competition among workers becomes fiercer when the number of workers increases. The platform will choose workers with lower cost to complete tasks, reducing the total payment. Similarly, the results show that the

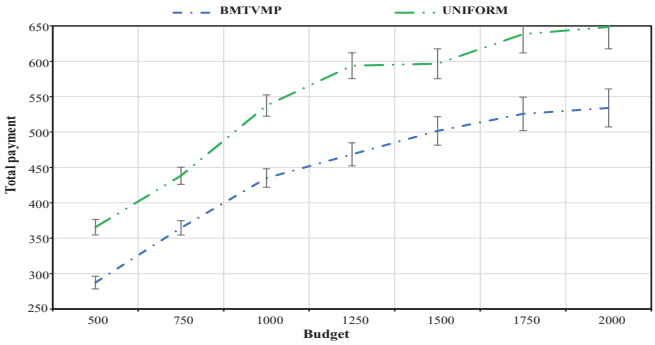

Fig. 7 The total payment when workers fixed at 300

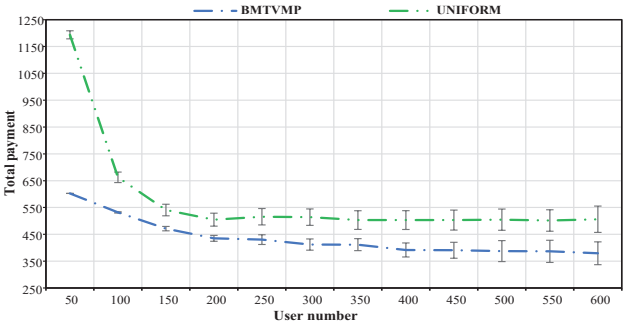

Fig. 8 The total payment when budget fixed at 1000

total payment of **BMTVMP** mechanism is less than **UNIFORM** algorithm. Compared with Figure 7, the total payments of both mechanisms decrease, implying that the number of workers has a greater impact on the total payment than the budget constraint.

The unit cost value: Finally, we use the unit cost value to measure the performance of **BMTVMP** mechanism and **UNIFORM** algorithm. The number of workers is fixed at 300 and the budget is set between 500 and 1000. The unit cost value of the bike-sharing platform in **BMTVMP** mechanism and **UNIFORM** algorithm are shown in Figure 9. The unit cost values of **BMTVMP** mechanism and **UNIFORM** algorithm decrease as budget increases. We find that when the budget increases, the sum of task region values increases and the total payment also increases. However, the experiment results show that the ratio decreases. That means when the budget is increased, the amount of increased payment is greater than the sum of task region values, which is consistent with Figures 5 and 7.

Next, the budget is fixed at 1000, while the number of workers is set between 50 and 600. The unit cost value of the bike-sharing platform in **BMTVMP** mechanism and **UNIFORM** algorithm are shown in Figure 10.

The unit cost values of **BMTVMP** mechanism and **UNIFORM** algorithm increase with the increased workers. By comparing Figure 6 with Figure 8, we know that when the number of workers increases, the sum of task region values increases, but the total payment decreases, and thus the ratio should increase. This means that when the number of workers increases, the mechanism can find workers with lower cost to complete high-value tasks, improving the unit cost value. We know that the unit cost value of **BMTVMP** mechanism are always greater than **UNIFORM** algorithm, which means that **BMTVMP** mechanism can also achieve better performance under the impacts of the number of workers. Comparing Figure 9 with 10, we find that attracting more workers to participate can obtain more values than simply increasing the budget.

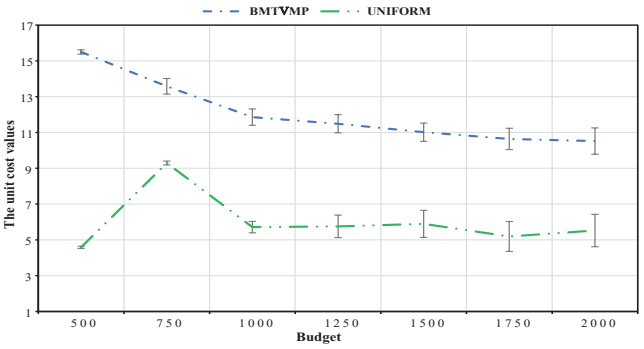

Fig. 9 The unit cost value when workers fixed at 300

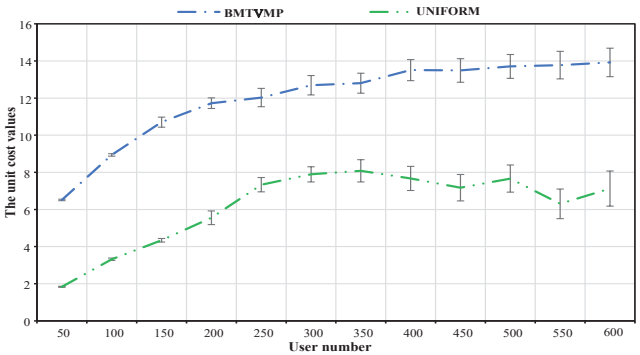

Fig. 10 The unit cost value when budget fixed at 1000

6 CONCLUSION

In this paper, we first design a task dispatching algorithm **GDY-MAX** to allocate tasks to workers to maximize the value of accomplished task regions. Since this algorithm cannot prevent strategic behavior of workers, we further propose a strategy proof mechanism to maximize the values of task regions under the budget constraint. In more detail, this mechanism consists of a task dispatching algorithm and a worker pricing algorithm. We use a technique of linear programming to improve the classical greedy algorithm by introducing Max operation in the submodule maximization problem, and thus design a task dispatching algorithm satisfying the monotonicity. Furthermore, we design a worker pricing algorithm satisfying the budget constraint. We theoretically prove that our mechanism is strategy proof. We then run extensive experiments to evaluate our mechanism based on a Mobike dataset. The results show that compared with the **UNIFORM** pricing algorithm and the **OPTIMAL** algorithm, our mechanism and GDY-MAX algorithm can achieve performance similar to the **OPTIMAL** algorithm in terms of the coverage ratio of accomplished task regions and the sum of task region values. We also show that our mechanism has better performance than **UNIFORM** algorithm in terms of the total payment and the unit cost value.

Funding

This paper was funded by the Humanity and Social Science Youth Research Foundation of Ministry of Education (Grant No. 19YJC790111), the Philosophy and Social Science Post-Foundation of Ministry of Education (Grant No.18JHQ060) and Shenzhen Fundamental Research Program (Grant No.JCYJ 20190809175613332).

Conflict of interest

The authors declare that they have no conflict of interest.

Availability of data and material

The data that support this study is available from
Kesci, [https://www.kesci.com/mw/dataset/5eb6787e366f4d002d77c331/file]

Code availability

The code is available in [https://github.com/Autumncow/code]

References

1. M.A. Satterthwaite, H. Sonnenschein, The Review of Economic Studies **48**(4), 587 (1981)
2. R.B. Myerson, Mathematics of operations research **6**(1), 58 (1981)
3. M. Holieninová, Z. Kádeková, T. Holota, U. Nagyová, Mobile Networks & Applications **25**(1) (2020)
4. V. Sathishkumar, J. Park, Y. Cho, Computer Communications **153**, 353 (2020)
5. J. Liu, Q. Li, M. Qu, W. Chen, J. Yang, H. Xiong, H. Zhong, Y. Fu, in *International Conference on Data Mining* (IEEE Computer Society, 2015), pp. 883–888
6. I. Xiao, CoRR **abs/1810.04058** (2018)
7. P. Chang, J. Wu, Y. Xu, M. Zhang, X. Lu, Soft Computing **23**(2), 613 (2019)
8. H. Jia, H. Miao, G. Tian, M. Zhou, Y. Feng, Z. Li, J. Li, IEEE Transactions on Automation Science and Engineering **17**(2), 909 (2019)
9. A.A. Kadri, I. Kacem, K. Labadi, Soft Comput. **23**(14), 5945 (2019)
10. W. Jia, Y. Tan, L. Liu, J. Li, H. Zhang, K. Zhao, Knowledge-Based Systems **178**, 84 (2019)
11. L. Pan, Q. Cai, Z. Fang, P. Tang, L. Huang, (AAAI, 2019), pp. 1393–1400
12. H. Jia, H. Miao, G. Tian, M. Zhou, Y. Feng, Z. Li, J. Li, IEEE Trans Autom. Sci. Eng. **17**(2), 909 (2020)
13. H. Lv, F. Wu, T. Luo, X. Gao, G. Chen, Theoretical Computer Science **803**, 105 (2020)
14. H. Lv, C. Zhang, Z. Zheng, T. Luo, F. Wu, G. Chen, (AAAI, 2020), pp. 2144–2151
15. Z. Feng, Y. Zhu, Q. Zhang, L.M. Ni, A.V. Vasilakos, in *Infocom, IEEE* (2014)
16. M.H. Cheung, R. Southwell, F. Hou, J. Huang, CoRR **abs/1503.06007** (2015). URL http://arxiv.org/abs/1503.06007
17. H. Jin, L. Su, H. Xiao, K. Nahrstedt, in *Acm International Symposium* (2016)
18. L. Gao, F. Hou, J. Huang, in *2015 IEEE Conference on Computer Communications, INFOCOM 2015, Kowloon, Hong Kong, April 26 - May 1, 2015* (IEEE, 2015), pp. 2803–2811
19. G. Bergantiños, M. Gómez-Rúa, N. Llorca, M. Pulido, J. Sánchez-Soriano, European Journal of Operational Research **284**(3), 1074 (2020)
20. Y. Azar, I. Gamzu, in *International Colloquium on Automata, Languages, and Programming* (2008), pp. 833–844
21. A.A. Ageev, M.I. Sviridenko, Journal of Combinatorial Optimization **8**(3), 307 (2004)
22. Z. Zheng, F. Wu, X. Gao, H. Zhu, S. Tang, G. Chen, IEEE Transactions on Mobile Computing **16**(9), 2392 (2016)

Figures

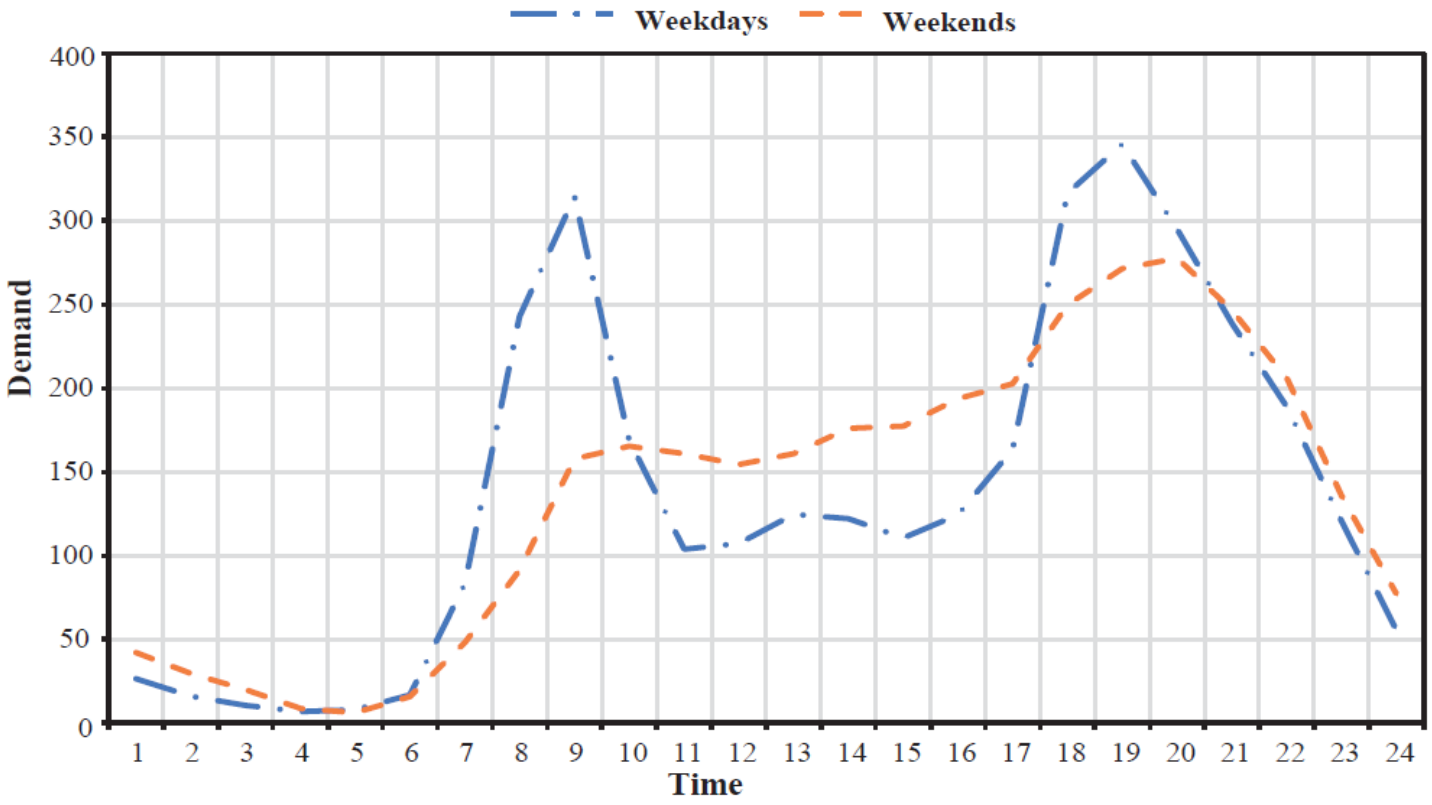

Figure 1

Demand of weekdays and weekends

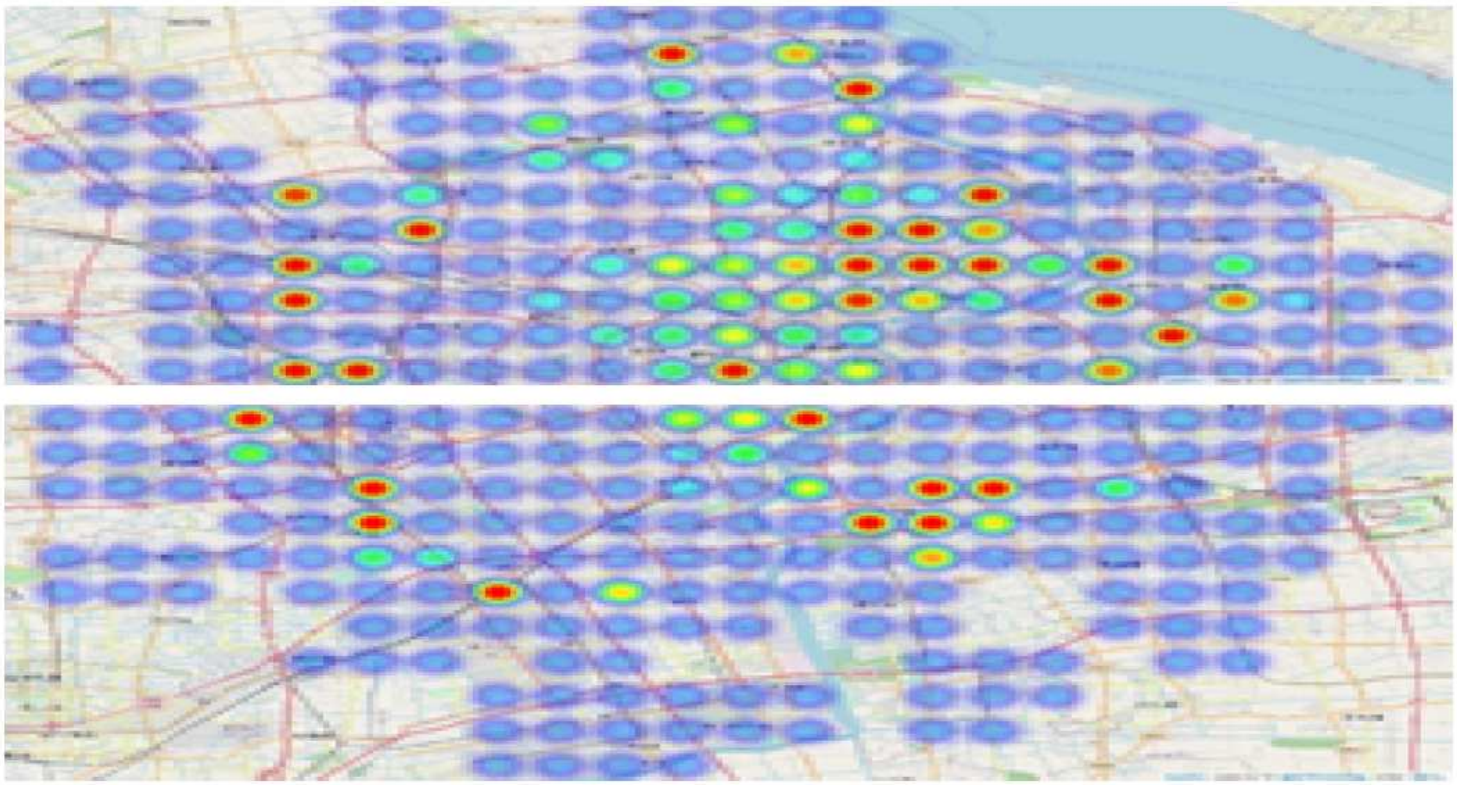

Figure 2

Task dispatching diagram

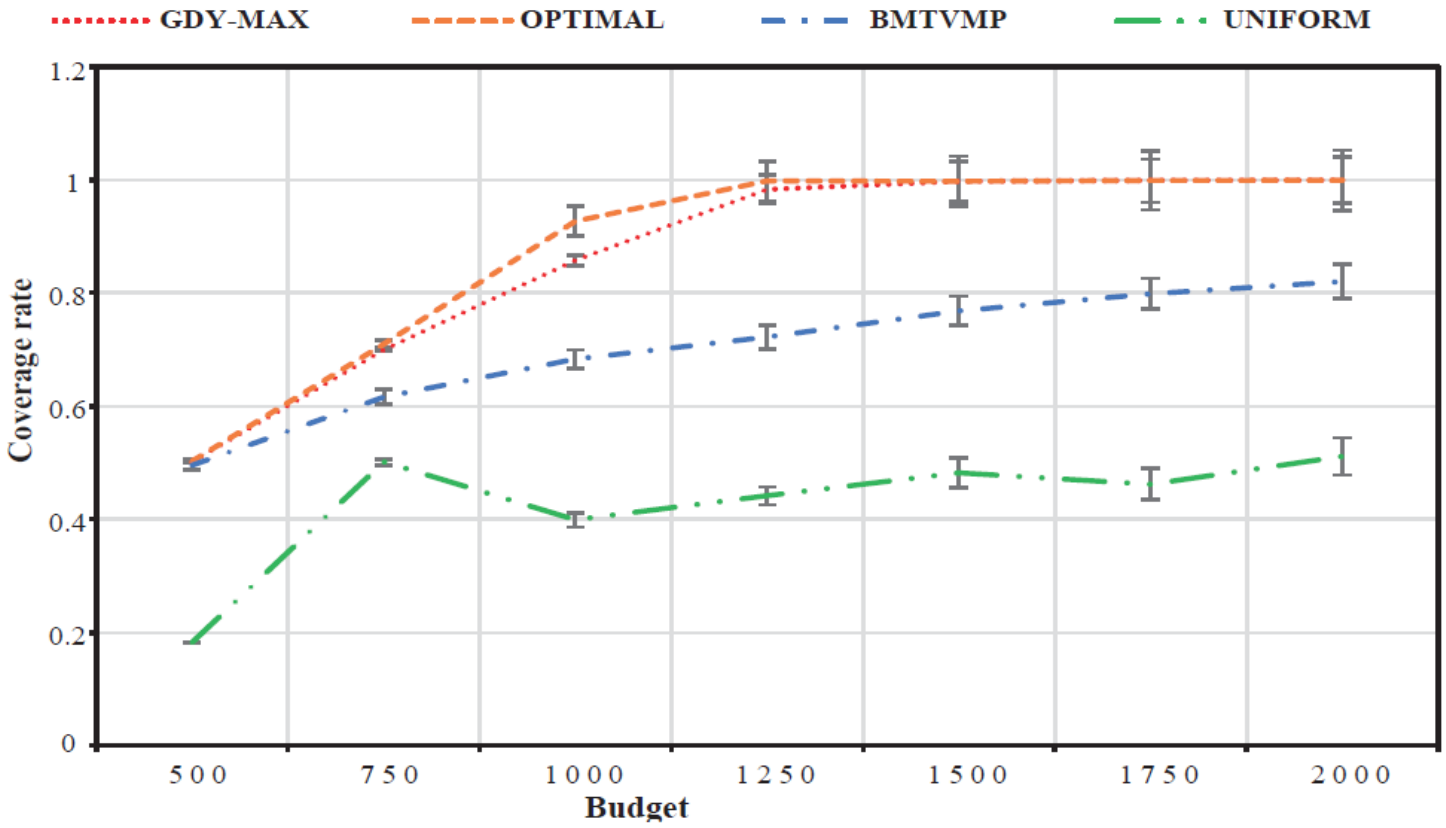

Figure 3

The coverage ratio when workers fixed at 300

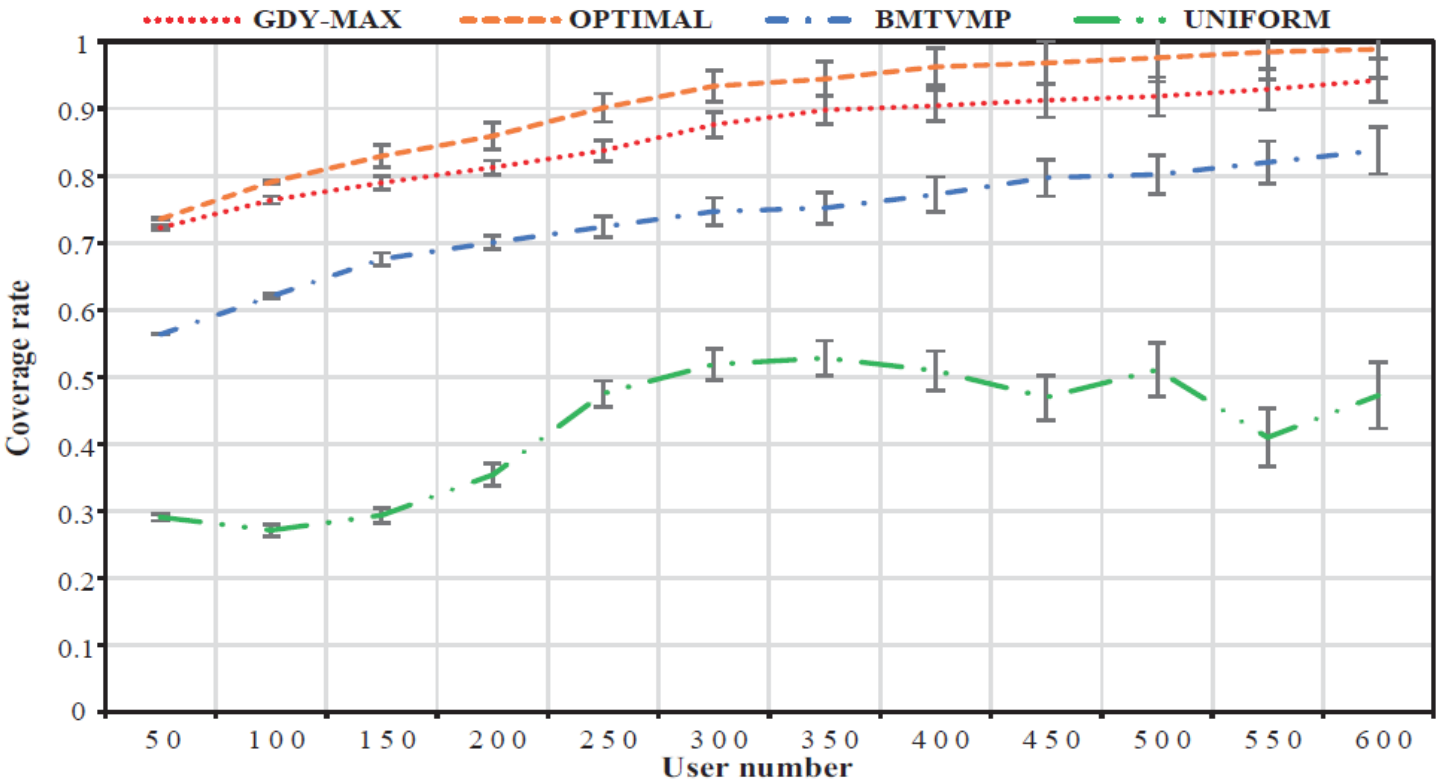

Figure 4

The coverage ratio when budget fixed at 1000

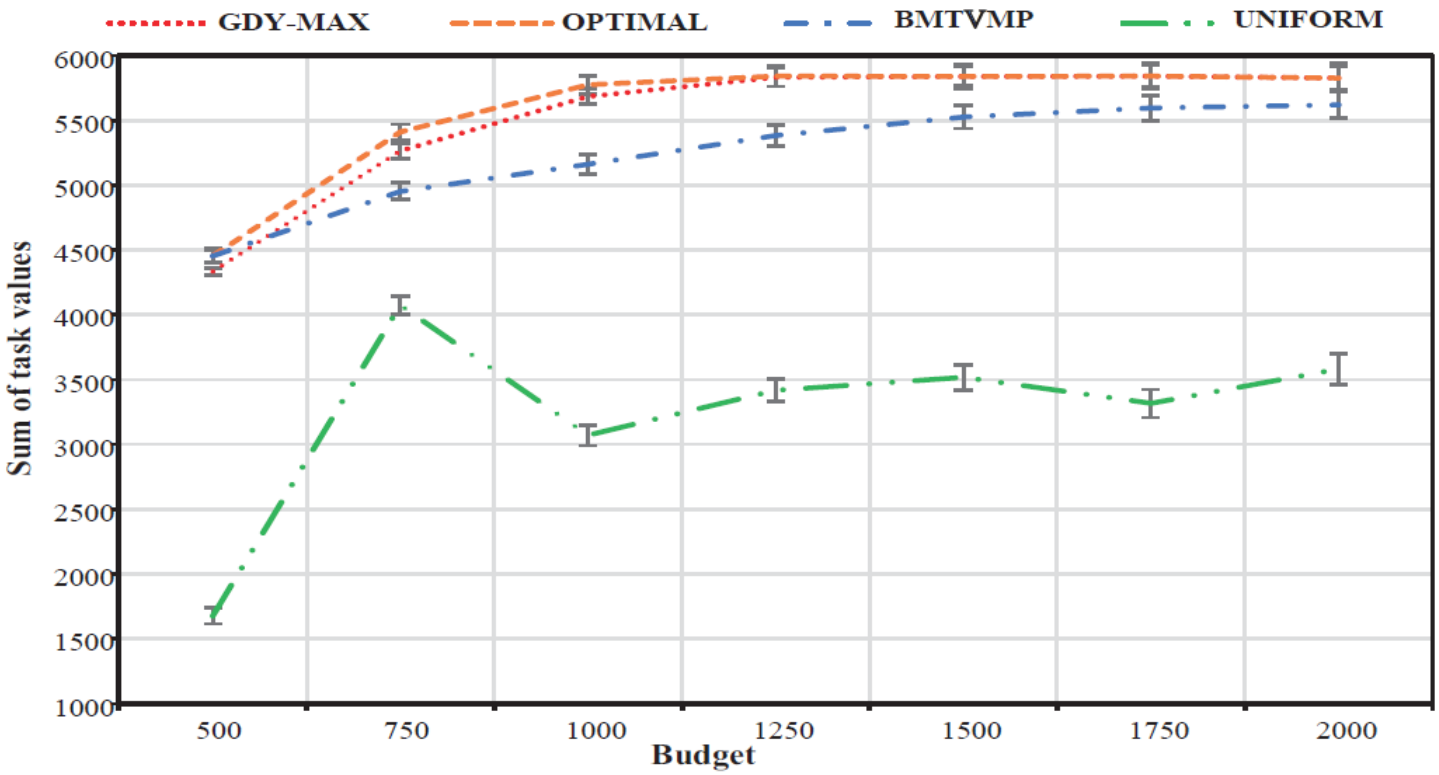

Figure 5

The sum of task region value when workers fixed at 300

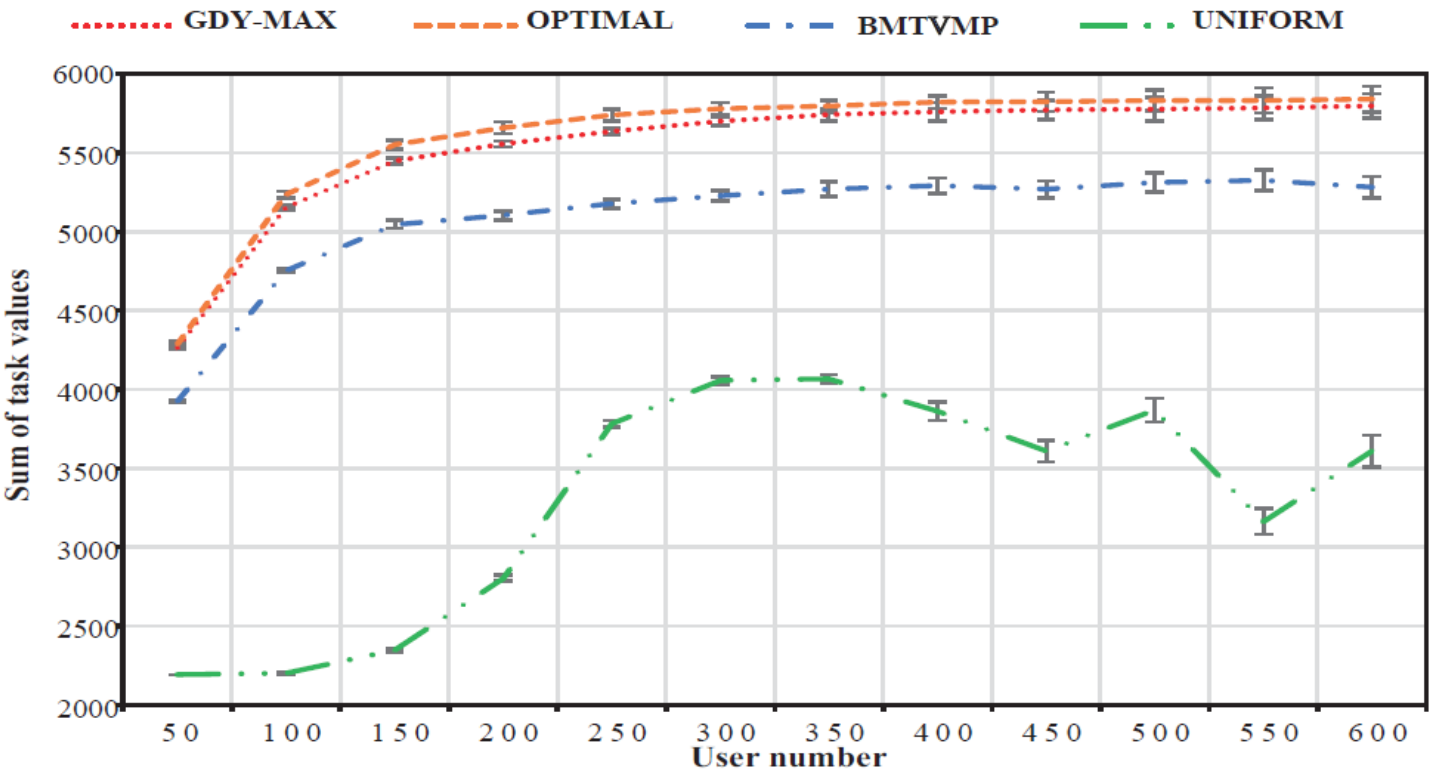

Figure 6

The sum of task region value when budget fixed at 1000

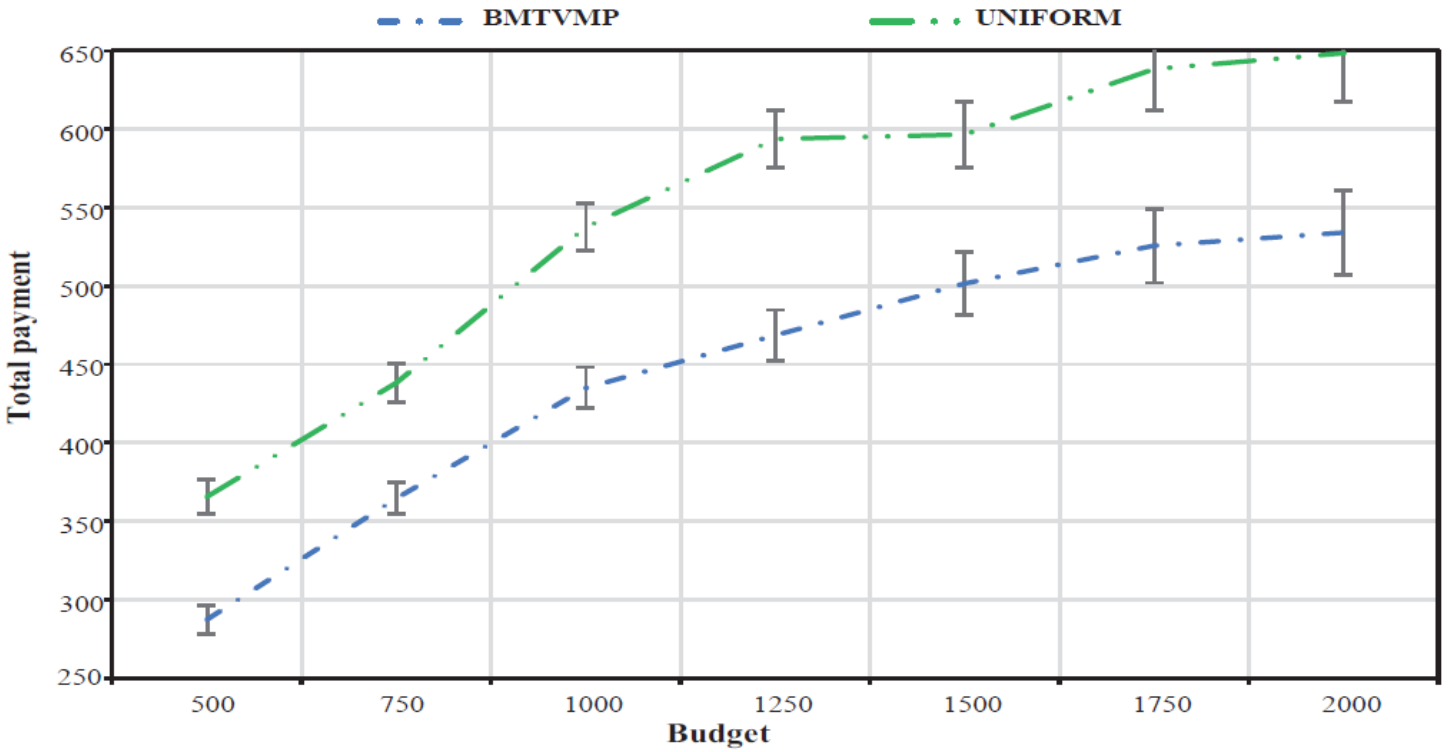

Figure 7

The total payment when workers fixed at 300

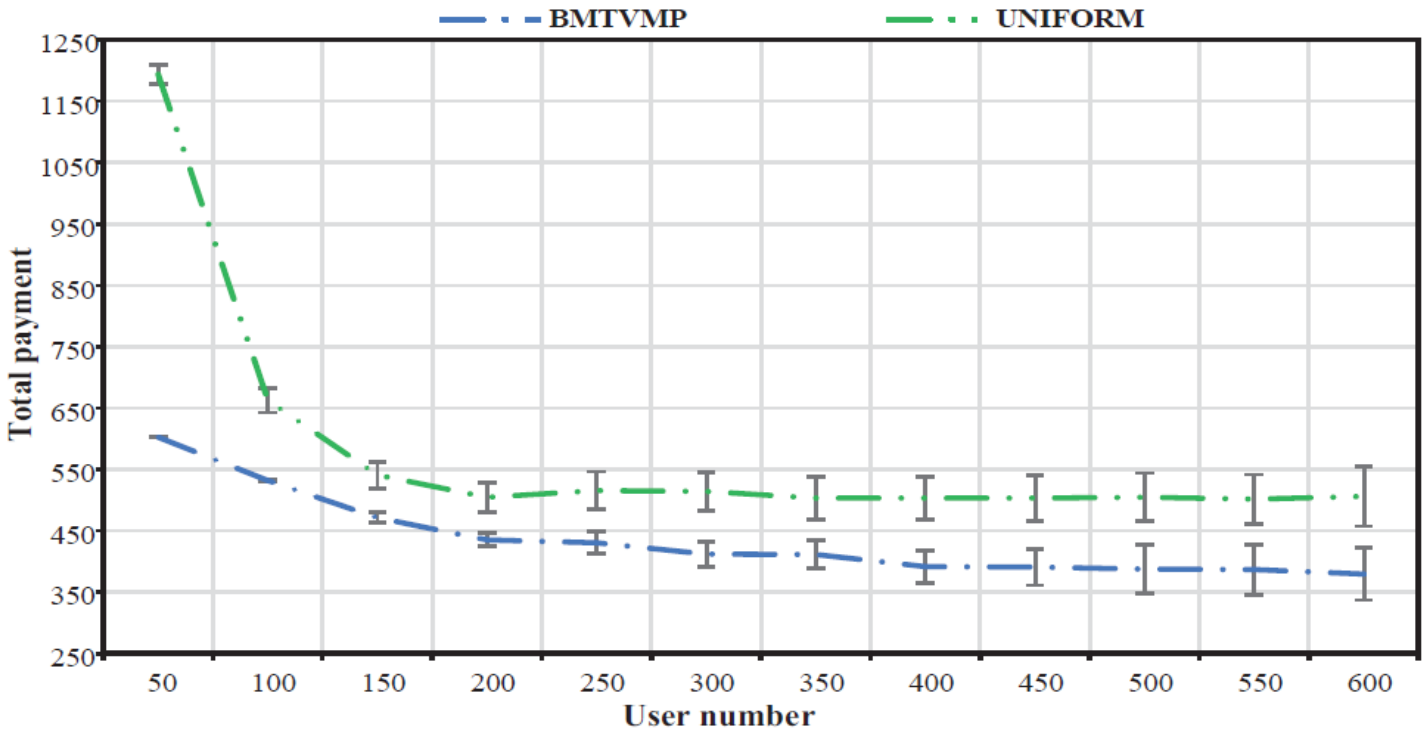

Figure 8

The total payment when budget fixed at 1000

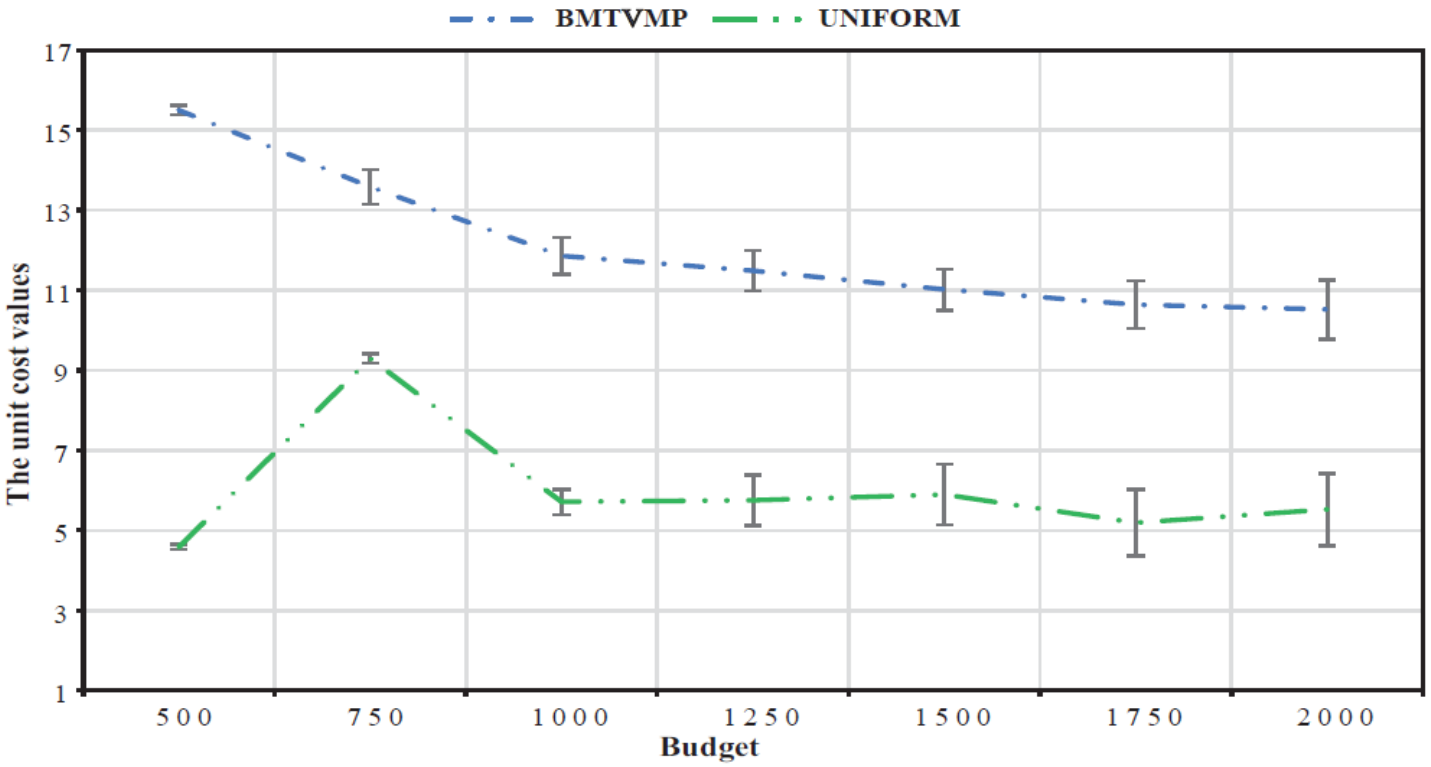

Figure 9

The unit cost value when workers fixed at 300

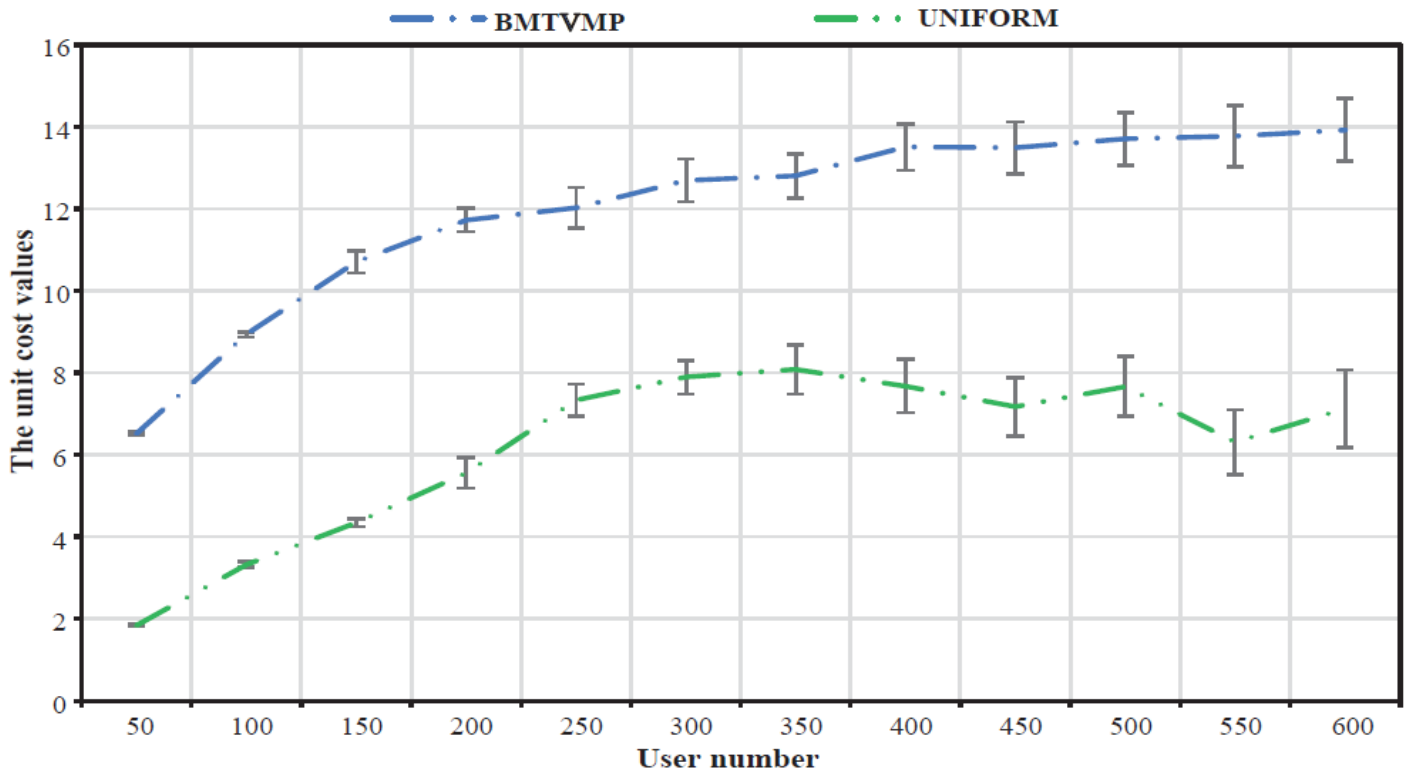

Figure 10

The unit cost value when budget fixed at 1000