

When Reinforcement Learning Meets Zero-Determinant: Social Welfare Control in Iterated Game Under Spatial Crowdsourcing

Jiqing Gu, Chao Song , *Member, IEEE*, Jie Wu , *Fellow, IEEE*, Li Lu , *Member, IEEE*,
and Ming Liu , *Member, IEEE*

Abstract—Spatial crowdsourcing is a prospective category of crowdsourcing that takes advantage of various sensors and resources of mobile devices in the area of mobile internet. The participants in spatial crowdsourcing interact with each other, such as requestors, platform, and workers, and they collaborate to complete the crowdsourcing tasks. The Internet Service Provider (ISP) plays a role in transmitting crowdsensing data and supports spatial crowdsourcing. The interactions of the participants form a game in crowdsourcing, but the game between the two players (platform and ISP) is paid little attention by researchers. The crowdsourcing platform assigns sensing tasks and the ISP selects a portion of the Road-Side Units (RSUs) to power on for data transmissions. The interaction between the platform and ISP is crucial, because it impacts the data quality in spatial crowdsourcing. In this paper, we investigate the iterated game between crowdsourcing platform and ISP in Spatial Crowdsourcing (SC). The objective is to achieve a possible maximal social welfare between the platform and ISP in the long term, which is defined as the weighted sum of their expected utilities. We first propose a Reinforcement Learning based strategy of Spatial Crowdsourcing (RLSC) to maximize the reward (i.e., expected utility) from the iterated game in the long term. To control the social welfare without considering the opponent's strategy, we investigate the Zero-Determinant strategy of Spatial Crowdsourcing (ZDSC) in a discrete case, which involves two actions and multi-actions, respectively. We model the iterated game by using the Markov Decision Process (MDP) and analyze how to theoretically obtain RLSC and ZDSC strategies with the objective of maximal social welfare control. Furthermore, we analyze the two strategies against each other as a Markov Game using the

MDP. Experimental results show RLSC can achieve global maximal social welfare sometimes, but it cannot control the social welfare when it plays against classical strategies, and the social welfare may fall into a local maximum. The proposed ZDSC strategy can unilaterally control social welfare to achieve a stable and possible maximal value, regardless of the opponent's strategies.

Index Terms—Crowdsourcing, iterated game, zero-determinant strategy, reinforcement learning.

I. INTRODUCTION

THE popularity of mobile devices has far expanded the application scenarios of Spatial Crowdsourcing (SC), due to its ability to provide fine-grained multi-dimensional sensor readings associated with location information. With the unprecedented popularity of various mobile devices, spatial crowdsourcing has recently been a fast-growing paradigm for gathering sensing data in the Internet of Things (IoT) applications (such as Waze¹). It takes advantage of mobile devices and provides services with fine-grained multi-dimensional sensor readings associated with location information, such as urban monitoring [1], air quality monitoring [2], wildlife habitat monitoring [3], and cross-space public information sharing [4].

In the spatial crowdsourcing, the infrastructure-based transmission is one of fundamental transmission paradigms [5], which reports sensing data via Internet provided by Internet Service Provider (ISP). Infrastructure availability is a key factor in determining proper crowdsensing management [6]. The ISP acts as a role for transmitting the crowdsensing data to support spatial crowdsourcing [7], [8]. The interactions between the crowdsourcing platform and ISP are investigated to learn how to obtain high quality sensing data via different transmitting scenarios provided by the ISP. The ISP provides different transmitting scenarios, such as setting different available RSUs each day, which impacts the transmission of sensing data, influencing the data quality. The sensing tasks are assigned to different areas, which also impacts the transmission profits of ISP under different cases of available RSUs. The interactions between the crowdsourcing platform and the ISP is crucial, which can directly influence economic outcomes. For instance, the ISP can reduce energy costs by optimizing the activation strategy of RSUs, while the platform can enhance the commercial value of

Received 28 September 2022; revised 19 October 2024; accepted 7 September 2026. Date of publication 11 September 2026; date of current version 29 September 2026. This work was supported in part by the National Natural Science Foundation of China under Grant 62472070, Grant 82241060, and Grant 62020106013, in part by the Natural Science Foundation of Sichuan Province of China under Grant 2024NSFC0492, in part by the National Key R&D Program of China under Grant 2021YFB3101303, and in part by the Science and Technology Key Research and Development of Sichuan Province of China under Grant 2023YFG0115, Grant 2023YFG0101, and Grant 2023YFG0100. Recommended for acceptance by M. Zhao. (*Corresponding author: Chao Song.*)

Jiqing Gu is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with the School of Computer Science, Chengdu University of Information Technology, Chengdu 610225, China (e-mail: gujiqing@cuit.edu.cn).

Chao Song, Li Lu, and Ming Liu are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: chaosong@uestc.edu.cn; luli2009@uestc.edu.cn; csmlu@uestc.edu.cn).

Jie Wu is with the China Telecom Cloud Computing Research Institute, Beijing 100089, China, and also with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122 USA (e-mail: jjewu@temple.edu).

Digital Object Identifier 10.1109/TPDS.2026.3733276

¹<https://www.waze.com/>

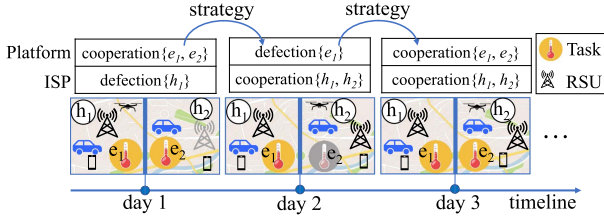


Fig. 1. Iterated game in spatial crowdsourcing.

data through effective task allocation. The interactions also can enhance the end-user experience. For example, if the ISP provides fast and reliable data transmission services, the platform can receive high-quality crowdsourced data promptly, thereby improving user satisfaction. In a spatial crowdsourcing application, the interactions between the crowdsourcing platform² and ISP are playing more than once in succession; we treated their interactions in the spatial crowdsourcing scenario as an iterated game. The iterated game between the platform and ISP exists in vehicular crowdsensing [9], content caching [10], mobile crowdsensing [11] and so on. Vehicular crowdsensing is one of the ways to complete crowdsensing tasks such as collecting sensing data [12], [13], [14]. We take vehicular crowdsensing in Fig. 1 as a case study of the game. Each round (e.g., a day), the platform publishes the selected tasks, and the ISP powers on a portion of RSUs for uploading data as their actions independently. According to their utilities and actions of the previous day, both of them will adjust their actions the next day. In the iterated game, we define the *social welfare* as the weighted sum of expected utilities of the platform and ISP.

Reinforcement learning (RL) [15] is appropriate to be used in the iterated game between platform and ISP, because platform and ISP involve different actions the game. So reinforcement learning is a powerful tool that helps to choose actions under different states. Reinforcement learning tries various actions, obtains the rewards, and learns the optimal action at each state, so as to achieve the optimal value in the iterated game. In reinforcement learning, the platform and ISP are agents or players in the iterated game, and they interact with each other each day by publishing different tasks and transmitting the crowdsourcing data via different available RSUs controlled by ISP. With adequate iterations, both platform and ISP record all the reward of every action under each state, and then choose the action with the largest reward under the current state. Besides, Both platform and ISP add the reward gained from future states to the current state's reward, impacting the present decision by the expected reward in the future effectively.

However, reinforcement learning can sometimes get stuck in a locally-optimal solution while a zero determinate based strategy can keep the total reward stable at a good performance whatever strategy its opponent use [16]. That is a participant adopts a zero determinate based strategy, and can control the total payoff of all participants without considering its opponents' strategy, which is called social welfare control. Social welfare control in the iterated games has been discussed in many previous

works: such as social welfare (total quality of data) control in mobile crowdsourcing [17], social welfare control to solve the crowdsourcing dilemma [18], [19], [20], total Quality of Services (QoS) control of all participants in wireless communication [21], unilaterally control on the expected payoffs of its opponent and itself in the block withholding attacks [22], an administrator unilaterally controls the total utilities of all players in the resource management of wireless networks [23], etc. Therefore, social welfare is an important metric in iterated games; one dominant player in the game concerns with the total social welfare and is responsible for controlling it in the long term. Maintain a high social welfare in the long term promotes a more balanced and cooperative environment, which benefits all participants and sustains the system in the long run. Therefore, the dominant player seeks to optimize the overall social welfare to ensure the system's stability. So we study the social welfare control problem with different strategies to help operators make decisions.

In this paper, we investigate the iterated game of spatial crowdsourcing between the platform and ISP, and the objective is to control a possible maximal social welfare between the platform and ISP. The possible maximal social welfare refers to a bigish stable value that the social welfare tries to achieve. The iterated game is an extension of Markov decision process (MDP) to the multi-agent case, which is defined by Markov Games. We use MDP to analyze and compare the proposed two strategies. First, we propose a Reinforcement Learning based strategy of Spatial Crowdsourcing (RLSC), which aims to learn how to choose the assignment of tasks or RSUs under the outcome of the previous round, so as to maximize the expected reward from the environment in the long term. Q-learning is a model-free reinforcement learning algorithm, which is to learn the value of an action in a particular state and widely used [24]. We utilize Q-learning to update the RLSC in each iteration, and the details are presented in Section IV. Furthermore, we consider the platform belongs to the government [25], who concerns about the social welfare on the whole, so the platform is responsible for controlling the social welfare. We discuss such social welfare control to reach a possible maximal stable value of the iterated game, and propose a Zero-Determinate strategy of Spatial Crowdsourcing (ZDSC). We analyze ZDSC in the discrete cases of two players with two actions and multi-actions, respectively. The basic idea of ZDSC is that it derives a linear relationship between the expected payoffs of the platform and ISP, then we can set proper parameters to obtain ZDSC. The details are presented in Section V. More specifically, we analyze the ZDSC and RLSC play against each other in the iterated game by Markov process in Section VI. In summary, the main contributions are as follows:

- We formulate the interactions between the platform and ISP as an iterated game under spatial crowdsourcing, and verify that the game exists in equilibrium in each iteration.
- We propose two strategies (RLSC and ZDSC) for the iterated game to maximize the social welfare in spatial crowdsourcing, and analyze the iterated game by Markov process.

²“platform” refers to “crowdsourcing platform”

- To the best of our knowledge, we are the first to investigate ZDSC against RLSC in the iterated game, and verify ZDSC can control social welfare when his opponent adopts RLSC.
- We implement the ZDSC and RLSC with real trace-driven simulations. By setting proper parameters for the to-be-adopted ZDSC, experimental results show that the platform can control social welfare to achieve a high and stable value, regardless of ISP's strategies.

This paper is organized as follows: In Section II, we survey the related work; Section III discusses the game between crowdsourcing platform and ISP, and analyses the game by MDP. Section IV introduces the iterated game by Reinforcement Learning based strategy; Section V presents the ZDSC strategy in the discrete model; Section VI discusses the two strategies ZDSC and RLSC. Section VII evaluates the performance and comparison of the ZDSC and RLSC strategies; the last section draws conclusions and presents our future work.

II. RELATED WORK

We briefly review related work on spatial crowdsourcing, games in spatial crowdsourcing, Reinforcement Learning in game, and zero-determinant strategy.

Spatial Crowdsourcing: Spatial crowdsourcing is a sensing paradigm where humans and mobile devices actively or passively involved in the procedure of sensing [26], [27]. The mobile Internet and shared economy promote the popular of spatial crowdsourcing, where the crowdsourcing tasks are spatiotemporal and completed in the constraint time and particular location. In general, a spatial crowdsourcing includes tasks, platform, workers and transmission process. The physical world makes spatial crowdsourcing a common way of sensing and collecting data for daily applications widely including many services, such as vehicular crowdsensing [28], [29], [30], task assignment [31], crowdsensing air quality [2], urban services (Gigwalk [32] and Uber [33]) and spatio-temporal data collection (Waze [34] and OpenStreetMap [35]). The main difference between spatial crowdsourcing and web-based crowdsourcing lies in that the spatial crowdsourcing requires the workers to finish the tasks while they are moving.

Games in Spatial Crowdsourcing: Game includes repeated (iterated) and static game. The games in spatial crowdsourcing often involves different parties [36]. The work [37] introduced the repeated interactions between the crowdsensing server and independent vehicles as a dynamic vehicular crowdsensing game [27], [37]. [38] modeled the evolutionary dynamics of human sensing behaviors via the rounds of iterated social dilemmas, and they validated the methodology in a vehicular crowdsensing scenario. Hu et al; [17] study the game between requestor and worker, so as to control the total quality of data in mobile crowdsourcing. The work [39] presented an incentive mechanism for vehicular crowdsensing in autonomous vehicles game, so as to address the problem of sensing coverage of regions located out of the AVs planned trajectories. However, the work [17], [37], [38], [39] focused on the interactions between task requestor and workers (users, vehicles), neglecting the role and action of ISP.

RL in Game: Reinforcement Learning (RL) is concerning how to map states to actions to maximize the cumulative rewards in the game. Some work investigated RL in games with many applications, such as [37] proposed the Q-learning-based strategies for the server and dynamic vehicular in the crowdsensing game to make the optimal decisions; [40], [41], [42] applied RL-based algorithms to obtain the optimal policy in the game; [40], [41] applied RL-based algorithms to obtain the optimal policy in the game. [42] proposed the RL-based cell selection algorithm for the sparse mobile crowdsourcing game. [41] utilized the RL-based advertising strategy by using the crowdsensing vehicular data, to determine the advertising policy for each digital billboard and maximize the commercial profit in the game. [43] developed an intelligent crowdsourcing algorithm by leveraging the power of multi-agent RL. Nevertheless, these work did not consider the social welfare control problem.

Zero-Determinant Strategy: Zero-determinant strategy can achieve different goals with different parameter settings, such as a player adopted zero-determinant strategy can set the expected utility of an opponent, set a ratio between their expected payoff or control social welfare, regardless of the opponent's strategies. [16] proposed a zero-determinant strategy in 2×2 iterated Prisoner's Dilemma game; the zero-determinant strategy is extended into the any 2×2 iterated games in many applications [18]. The works [21] investigated the power control problem in resource sharing among wireless users and network operators; the network operator applied zero-determinant strategy to control social welfare, which modeled the resource sharing among participants as an iterated game. [17], [20] formulated the interaction between a requestor and workers as an iterated game, improving data quality in mobile crowdsourcing. The work [23] studied the equilibrium for zero-determinant strategy in resource management in the wireless network. [18] proposed a zero-determinant strategy to address the malicious attack problem in crowdsourcing. Furthermore, zero-determinant strategy is studied in the evolutionary game [44].

However, these work focus on the interactions between task requestor and workers (users, vehicles, or other mobile devices), neglecting the role and action of ISP; they applied the strategy obtained from reinforcement learning [37] or a greedy method [39] to maximize their utility, which may fall into local optimum. Moreover, these work do not involve the iterated game between a ZDSC player and an RLSC player.

III. SYSTEM MODEL

In this section, we first present the spatial crowdsourcing model, then we introduce the game formulation and discuss the iterated game in MDP. Frequently-used notations in this paper are listed in Table I.

A. Spatial Crowdsensing Model

This subsection introduces the participants and their interactions in the spatial crowdsourcing scenario. As shown in Fig. 2, spatial crowdsourcing systems usually include the SC task, SC platform, SC workers, and the transmission process. Usually, the SC workers are mobile devices such as vehicles

TABLE I
DESCRIPTION OF FREQUENTLY-USED NOTATIONS

Notation	Description
$E = \{e_1, \dots, e_m\}$	The set of m candidate sensing tasks.
$H = \{h_1, \dots, h_n\}$	The set of n candidate RSUs.
a_1^h/a_2^h	Highest / lowest number of RSUs.
a_1^e/a_2^e	Highest / lowest number of tasks.
W	Weighted task-RSU traffic matrix.
$f(\cdot)/g(\cdot)$	Payoff function of platform / ISP.
$u_d/u_t/u_e/u_h$	Parameters of payoff function.
C	Operating cost to vehicles.
$\mathbf{P}/\mathbf{Q}$	The payoff matrix of platform / ISP.
P_i, Q_i	The elements in $\mathbf{P}$ and $\mathbf{Q}$.
$\mathbf{p}/\mathbf{q}$	The mixed strategy of platform / ISP.
$\mathbf{v}_s/\mathbf{f}$	The stationary vector / any vector.
$U^e/U^h/U_{all}$	Utility of platform / ISP / both.
$\mathbf{M}$	The Markov state transition matrix.
α, β, γ	Parameters in ZDSC.
η, θ	Parameters in updating RLSC.
$\pi^e/\pi^h, a^e/a^h, s$	Policy, action, state in RLSC.

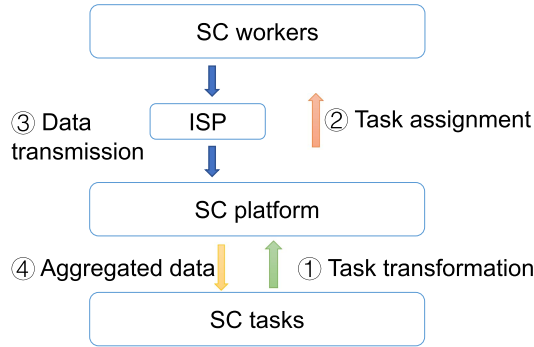


Fig. 2. Framework of spatial crowdsourcing.

and individuals with mobile devices. In the transmission process, ISP provides RSUs for transmitting crowdsourcing data. The SC task providers, such as Uber, DiDi, Yelp, Waze, etc., transfer the tasks to the SC platform, which assigns the tasks to the recruited SC workers. The SC workers collect the crowdsourcing data and send them to the SC platform via RSUs provided by ISP. Then the platform aggregates the obtained data and sends them to the SC task providers. The word 'spatial' represents that vehicles connect the RSUs to upload the sensing data, so vehicles and RSUs correlate with one another in the spatial scenario. Spatial crowdsourcing is usually applied in data collection or query answering tasks.

The platform is responsible for publishing the crowdsourcing tasks (e.g., monitoring the temperatures [4]) in the SC, and assigning tasks to the recruited mobile devices. Such as Waze [34] is a crowdsourcing platform that leverages the equipped sensors in users' mobile devices to collect and share data. In Fig. 1, each grid is assigned a sensing task of the area, and we denote the set of m sensing tasks as $E = \{e_1, e_2, \dots, e_m\}$. The platform could belong to the government [25] or a service platform company (such as Uber [33]). The recruited mobile devices are responsible for sensing data; these mobile devices could be mobile phones, taxis, buses, cars, robots, and so on, which are managed by the platform. The platform's income comes from the quality of sensing data, which is calculated by the quantity of sensing

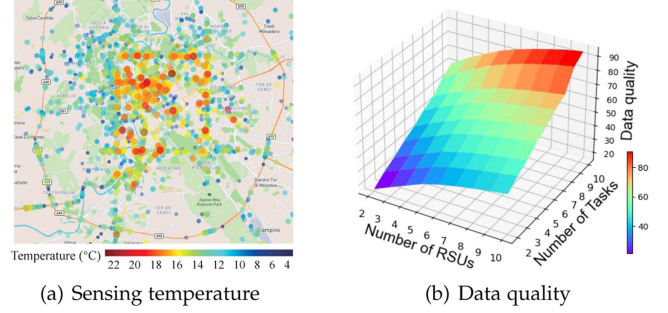


Fig. 3. Roma temperature data with spatial crowdsourcing.

data and the diversity of tasks. We assume the platform pays a constant operating cost (denoted as C) to the fixed recruited mobile devices every day. The mobile devices sense data passively when they pass the sensing areas; instead of actively driving to the sensing areas; the incentive mechanism for mobile devices, proposed in [19], [45], [46], is out of the scope of this paper.

The RSUs, such as Wi-Fi, femtocell, and Smart Pole,³ are installed by ISP and provide communication for mobile devices in the SC, which are denoted by $H = \{h_1, h_2, \dots, h_n\}$. These n RSUs are responsible for uploading the sensing data generated by the mobile devices [47]. To save the energy cost, the switch-mode of cell activation is widely used [48], so the ISP schedules to partly power on RSUs. Generally, compared to a cell-tower in cellular systems, the coverage of each RSU is relatively small [49], about 100-500 meters. Thus, it is hard to provide seamless roaming for vehicles [50], so the sensing data's offloading can be delayed during its lifetime through an access point acting as a gateway. If there is no available RSU, the sensing data can be delayed instead of being immediately sent over the RSU; this kind of technology has been extensively investigated [51].

This paper focuses on the interactions between the platform and ISP, which is the fundamental support for transmitting sensing data. We take vehicular crowdsourcing as a case study to illustrate the iterated spatial crowdsourcing; Fig. 3(a) is an example of a temperature monitoring scenario in Rome, Italy by the real data trace [52]. The points in Fig. 3(a) are the temperature data and their positions sensed by the vehicles, which is also called sensing data. A red point refers to a place with a higher temperature, and a blue one refers to a place with a lower temperature. The sensing data (such as temperature) are generated by the recruited vehicles and wait to be offloaded when they leave the sensing area, the infrastructure operators (such as ISP) power on part of RSUs to upload the sensing data. On each day, the platform selects to publish several tasks of sensing the temperatures at some areas like the grids in Fig. 1, and the ISP powers on a portion of RSUs for uploading the sensing data to save energy costs. When a mobile vehicle passes by an area that needs to be sensed, it will generate a sensing data packet with records, and this packet waits for uploading by the way of mobile data offloading. When a vehicle leaves the current sensing area with sensing data and moves to the communication range of a

³<https://www.omniflow.io/smartpole>

working RSU, the data packet is uploaded to a remote cloud server through the RSU.

B. Game Formulation

In this subsection, we formulate the payoffs in the iterated game and propose the problem of social welfare control.

We use a^e and a^h to denote the number of selected tasks and RSUs, respectively. A task refers to a sensing task, such as measuring the temperature, humidity, air quality index (AQI), or noise levels in environmental monitoring in specific areas of a city. These sensing tasks are issued by demanders who post them on the crowdsensing platform. Generally, the sensing tasks of different areas have different priorities, and the RSUs are powered on by a matching algorithm [53]. The detail setting of the selection tasks by priority is in Section VII-A. Based on the task priority, the platform selects top a^e sensing tasks and ISP matches a^h RSUs. In our vehicular crowdsourcing scenario, the platform earns values from the quality of harvested sensing data and the diversity of tasks, and pays for the data transmission and recruitment of vehicles. We assume the size of each sensing data packet is the same. The data quality is defined as:

$$O_d(a^e, a^h) = \sum \left(\log(W_{i,j}^{a^e, a^h} + 1) - K_{ij} \log K_{ij} \right), \quad (1)$$

where W^{a^e, a^h} refers to the traffic distribution matrix of a^e sensing tasks under a^h RSUs, $W_{i,j}^{a^e, a^h}$ represents the traffic count of the i -th sensing task under the j -th RSUs, $\log(W_{i,j}^{a^e, a^h} + 1)$ reflects the growth rate of platform's quality decreases (diminishing return) as the increment of sensing data [54]; the $\log$ function also reflects the redundant sensing data cannot contribute much to the data quality. We utilize entropy $-K_{ij} \log K_{ij}$ to model the diversity of sensing data [55], which reflects the diversity of transmission and sensing data. The matrix $K_{ij} = W_{i,j}^{a^e, a^h} / \sum W_{i,j}^{a^e, a^h}$ represents the sum of all the elements in a matrix.

The utility of the platform depends on the data quality of the sensing data, payment of transmitting the data and recruiting vehicles. We formulate the payoff function of the platform as the value of sensing data minus the costs of data transmission and represented as:

$$f(a^e, a^h) = u_d \cdot O_d(a^e, a^h) - u_t \cdot \sum W^{a^e, a^h} - C, \quad (2)$$

where u_d is the benefit of per data quality could bring, u_t is the transmission price of per data traffic; C refers to the cost of recruiting the fixed vehicles. Note that the platform hopes to publish more tasks, in order to cover more areas with more sensing data uploaded by RSUs and obtain more utility.

Fig. 3(b) shows the trend of data quality with the selected a^e sensing tasks and a^h RSUs by the real data trace [52]. The detailed setting about the selection of tasks by priority is introduced in Section VII-A. We notice that the data quality is increased with more sensing tasks and RSUs, which determines the income of the platform. To transmit more sensing data, the ISP is required to power on more RSUs, which causes higher energy costs. The ISP earns the traffic income of transmitting the sensing data, and pays the energy costs of operations and

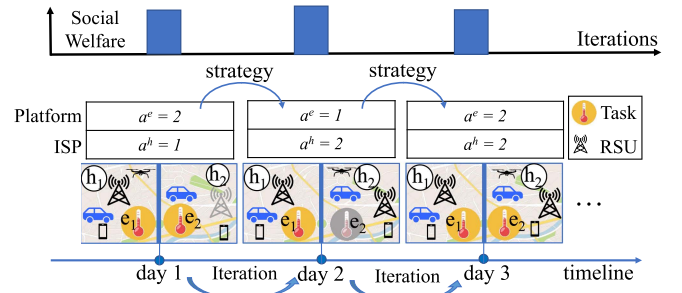


Fig. 4. Social welfare control in spatial crowdsourcing.

transmissions by RSUs. Thus, we formulate the payoff function of ISP as the traffic income of sensing data minus the energy costs on RSUs as follows:

$$g(a^e, a^h) = (u_t - u_e) \cdot \sum W^{a^e, a^h} - a^h u_h, \quad (3)$$

where a^h represents the number of RSUs powered on and u_h refers to the per cost of operating an RSU; u_e refers to the energy cost to transmit per sensing data.

For each day, the platform first select the tasks and then publishes them, the ISP powers on a portion of RSUs, both the platform and the ISP are independent. In the above two payoff functions, the cost of uploading sensing data of the platform is $u_t \cdot \sum W^{a^e, a^h}$, which is also the income of the ISP. In (3), when the number of sensing tasks reaches a certain level, the utility of the traffic from the sensing data is greater than the energy costs; thus, the ISP is willing to power on more RSUs. This is an iterated game with incomplete information, because the strategy of the platform is private for ISP before the real implement, and vice versa.

Problem 1 (Social Welfare Control): We aim to control a maximal social welfare of the iterated game at a stable and maximal possible value, which indicates the social welfare shows little changes and sustains around a maximal possible value as the increase of the game iteration.

$$\begin{aligned} \max \quad & U_{all} = \alpha U^e + \beta U^h, \\ \text{s.t.} \quad & U^e + U^h = -\gamma. \end{aligned} \quad (4)$$

where U_{all} is the weighted sum of the expected payoff of the platform and ISP; α and β refers to the weights of expected utilities U^e and U^h , respectively; γ is a variable that refers to social welfare.

The constraint in (4) indicates the social welfare is a constant. The objective function and the constraint determine that the goal is to maintain the social welfare at a table and possible maximal value.

Fig. 4 is an example to illustrate social welfare control in the spatial crowdsourcing iterated game. On each day, the platform and ISP choose their action by their strategies, such as the action are chosen as $a^e = 1, a^h = 2$ on the second day. Under such strategies and actions, the social welfare is expected to be possible maximal and stable in the long term.

Theorem 1: The game between the platform and ISP exists equilibrium in each round.

Proof: For the simple case, both the crowdsourcing platform and ISP have two actions in the 2×2 game; the actions of the platform and ISP are finite by the requirement. If there is no existence of pure actions, mixed actions must exist because every finite strategic-form game has a mixed-strategy equilibrium [56]. $\square$

Zero-determinant strategy and reinforcement learning based strategy are two kinds of methods to achieve the goal of (4) in the iterated games. Below we introduce these two solutions separately.

IV. SOCIAL WELFARE CONTROL WITH RLSC

In this section, we first model the iterated game in Markov Decision Process (MDP), then we analyze the iterated game by Reinforcement Learning based strategy.

A. MDP in the Iterated Game

The iterated game is viewed as a Markov Decision Process (MDP) [57], which is modeled as follows:

- *Action space:* The action space of the platform is denoted as A^e , where the platform's action ($a^e \in A^e$); the action space of ISP is denoted as A^h , where the ISP's action ($a^h \in A^h$). The highest and lowest number of tasks is denoted as a_1^e and a_2^e , respectively;
- *State space:* The state space is the action pairs in the previous round, and denoted as the set $S = \{(a^e, a^h) | a^e \in A^e, a^h \in A^h\}$, $s_t \in S$.
- *Reward:* We denote r^e or r^h as the immediate reward received after transitioning from state s to state s' for the platform and ISP, respectively. The reward of the platform and ISP are calculated by $f(a^e, a^h)$ in (2) and $g(a^e, a^h)$ in (3), respectively.
- *State transition probabilities:* $P_a(s, s') = \Pr(s_{t+1} = s' | s_t = s, a_t = a)$ is the probability that action a in state s at time t will lead to state s' at the $t + 1$ round.

A policy function π is a (potentially probabilistic) mapping from state space S to action space A . Once the MDP combines with a policy, a Markov Chain forms. Since the action chosen in state s is completely determined by $\pi(s)$ and $\Pr(s_{t+1} = s' | s_t = s, a_t = a)$ reduces to $\Pr(s_{t+1} = s' | s_t = s)$ (a Markov transition matrix). Briefly speaking, in the MDP, a player repeatedly observes the current state s_t of the environment and takes an action from all available actions in this state. Then, the state will transfer to s_{t+1} and the player will get a reward r_t from the environment for its action.

Below we analyze the social welfare control problem by RLSC strategy and ZDSC strategy under the MDP, respectively. The two strategies have the same action space and state space, but their calculation methods of long-term reward are different due to their different strategies.

B. Analyze Social Welfare Control With RLSC

In this subsection, we analyze the iterated game by Reinforcement Learning based strategy. In general, the actions of the

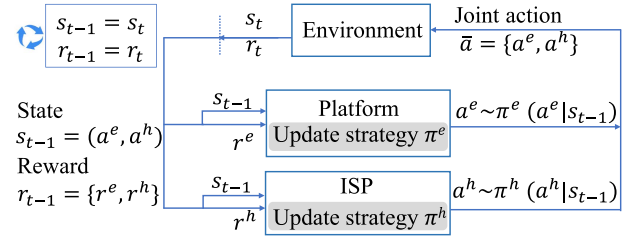


Fig. 5. Framework of RLSC.

player adopted RLSC are usually discrete, which could be two distinct action or multi-actions.

For the iterated game of spatial crowdsourcing, we propose a Reinforcement Learning based strategy of Spatial Crowdsensing (RLSC), whose framework is shown in Fig. 5. In practice, the platform and ISP's actions are not known to each other before the sensing tasks are completed. In the iterated game, the platform and ISP are viewed as the agents in reinforcement learning scenario. The joint action is $\bar{a} = \{a^e, a^h\}$. We denote the policy (or strategy) of the platform as $\pi^e(a^e | s_{t-1})$, which means the probability for choosing a^e in the current round under the state s_{t-1} of the previous round. Similarly, we write the policy of ISP as $\pi^h(a^h | s_{t-1})$, which is the probability for choosing a^h under the state s_{t-1} . With the chosen action, the state at s_t forms, and the reward pair r_t of platform and ISP is calculated. Then the process is repeated that the platform and ISP choose their action at the current state.

RL is learning how to map states to actions, so as to maximize the rewards from the environment in the long term. So we need to learn policies π^e and π^h to achieve the goal. The details of the policy update is shown in Algorithm 1. We first randomly initialize π^e , π^h , initial state and parameters. When both the platform and ISP adopt RLSC in the game, they will sample an action a^e and a^h based on the policy π^e and π^h in each round, respectively (line 2); this process contains the exploration of action space, because the action are generated with probability by the platform and ISP's strategies. Then, the platform and ISP transform to a new state and obtain their rewards (line 3), their rewards depend on their joint action. The platform maintains a Q-table $Q_e(s, a^e)$, which represents the value of an action in a particular state s ; it also records the value of each state and is denoted as $V_e(s)$, which refers to the expected reward of possible actions at state s , and can be calculated as follows:

$$Q_e(s, a^e) = \eta(r_e + \theta \max_{a^e} Q_e(s', a^e) - Q_e(s, a^e)), \quad (5)$$

$$V_e(s) = r_e + \theta V_e(s'), \quad (6)$$

where $\theta \in [0, 1]$ is the discount factor indicating a myopic view of the Q-learning regarding the future reward; $\eta \in (0, 1]$ is the learning rate. At the end of each round in the game, the platform updates its Q-table by (5) and update the value of a particular state by (6) (line 4). If $Q_e(s, a^e) > V_e(s)$, it represents the action a^e may bring more rewards in the long term, and we increase the probability of choosing a^e . Otherwise, we decrease this probability (lines 6-8). Similarly, ISP maintains $Q_h(s, a^h)$ and

Algorithm 1: Update RLSC in Discrete Model.

Input: Randomly initialize $\pi^e, \pi^h, s, \theta, \eta$;
Output: π^e, π^h

```

1 repeat
2    $a^e \sim \pi^e(a^e|s), a^h \sim \pi^h(a^h|s);$ 
3    $s, a^e, a^h \rightarrow s', r_e, r_h;$ 
4   Update  $Q_e(s, a^e)$  and  $V_e(s)$  by Eq.(5) and Eq.(6);
5   if ( $Q_e(s, a^e) > V_e(s)$ ) then
6      $\pi^e(a^e|s) \leftarrow \pi^e(a^e|s) + \theta(\max \pi^e - \pi^e(a^e|s));$ 
7   else
8      $\pi^e(a^e|s) \leftarrow \pi^e(a^e|s) - \theta(\max \pi^e - \pi^e(a^e|s));$ 
9   Update  $Q_h(s, a^h), V_h(s)$  by Eq.(7) and Eq.(8);
10  if ( $Q_h(s, a^h) > V_h(s)$ ) then
11     $\pi^h(a^h|s) \leftarrow \pi^h(a^h|s) + \theta(\max \pi^h - \pi^h(a^h|s));$ 
12  else
13     $\pi^h(a^h|s) \leftarrow \pi^h(a^h|s) - \theta(\max \pi^h - \pi^h(a^h|s));$ 
14   $s \leftarrow s';$ 
15 until  $\pi^e, \pi^h$  converge;
16 return  $\pi^e, \pi^h;$ 

```

$V_h(s)$ (line 9), the update of its policy (lines 11-13) is as follows:

$$Q_h(s, a^h) \leftarrow \eta(r_h + \theta \max_{a^h} Q_h(s', a^h) - Q_h(s, a^h)), \quad (7)$$

$$V_h(s) = r_h + \theta V_h(s'). \quad (8)$$

The updates of (5)-(8) are processed in each round of the iterated game. After the policies π^e and π^h are converged, the platform and ISP choose their actions based on their policies, and will maximize the total reward (social welfare) in the long term.

In scenarios of large action space, Q-table is too large and increase the running time. Function approximation techniques can be employed to address this issue. For example, Deep Q-Networks (DQN) [58] are well-suited for scenarios with discrete action spaces, as they approximate the value function without the need to explicitly maintain a large Q-table.

V. SOCIAL WELFARE CONTROL WITH ZDSC

In this section, we analyze the iterated game with the aim of social welfare control by zero-determinant strategy, then we discuss the discrete strategy model.

A. Game Analysis With ZDSC

To achieve the goal of social welfare control, the crowdsourcing platform takes advantage of the zero-determinant strategy to control social welfare. Zero-determinant strategy [16] is utilized in a general memory-one iterated game. The iterated game starts with an initial action, and players' strategies decide a stochastic process. It is proven that a long-memory player in the iterated game has no advantage over a short-memory player, since players in each iteration determine their moves based on the outcome of the previous round [16]. This means the corresponding stochastic process can be represented by a Markov chain, where the state transitions are joint probabilities

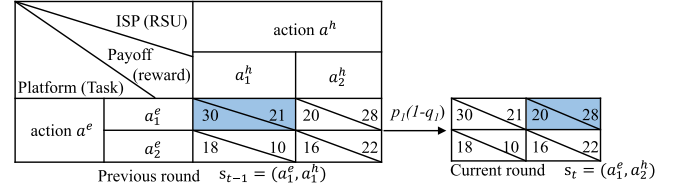


Fig. 6. Probability transition process in iterated game.

calculated from the probabilistic strategies of players. The actions of players in each iteration are obtained from the strategy. A zero-determinant strategy is a probability vector, where each element is the probability for choose a specified action in the current round under the actions of the previous round, and the probability is calculated by the payoff functions of the platform and ISP in the game. Fig. 6 shows the transition process of the previous round to the current round. In the iterated game of each day, the number of sensing tasks published and RSUs powered on are calculated by the strategies and the outcome of previous round; the platform can control the social welfare without considering ISP's strategy. p_1 and q_1 are the probabilities of choosing action a_1^e and a_1^h of the platform and ISP in the current round (e.g. a day) under previous state (a_1^e, a_1^h) , respectively.

We will analyze the game with multi-actions in the iterated game. To better understand the discrete strategy of ZDSC, we first analyze ZDSC with two actions, where the platform and ISP adopt either an extremely amicable or a vicious action; then we present the ZDSC with multi-actions.

B. ZDSC in the Discrete Model

In this subsection, we first introduce the two-action ZDSC is to provide a simpler example, to explain how to derive the linear relationship between the expected payoff of platform and ISP, making it easier for readers to grasp the core idea behind the Zero-Determinant strategy in spatial crowdsourcing. Then we jumping directly into the multi-action case.

1) **ZDSC With Two Actions:** In the discrete model with the two actions, such as $a^e \in \{a_1^e, a_2^e\}$ and $a^h \in \{a_1^h, a_2^h\}$. At each round, the player's action is private; thus, there are four outcomes of each iteration in the game. The four possible outcomes of each iteration are labeled as 1, 2, 3, and 4 associate with the outcomes $a^e a^h \in (a_1^e a_1^h, a_1^e a_2^h, a_2^e a_1^h, a_2^e a_2^h)$ respectively. We assume that the platform and ISP only have the state memory of the previous round. As mentioned in [16], it is demonstrated that a long-memory player in the iterated game is without any advantage over a short-memory player when the game in each stage is identically repeated infinite times. In the game, both players have mixed strategies at each round, denoting the probabilities the specified action (such as a_1^e and a_1^h) under the four possible states in the previous round. Accordingly, we define the mixed strategy of the platform as $\mathbf{p} = (p_1, p_2, p_3, p_4)$, which is a zero-determinant strategy of the platform. The mixed strategy of ISP is $\mathbf{q} = (q_1, q_2, q_3, q_4)$. Here, p_1, p_2, p_3, p_4 and q_1, q_2, q_3, q_4 are the probabilities of choosing a_1^e or a_1^h in the current round when the state of previous round is $a^e a^h = (a_1^e a_1^h, a_1^e a_2^h, a_2^e a_1^h, a_2^e a_2^h)$. Fig. 6 shows the probability transition process of previous round's action to the

current round's action. In the current round, we denote the possibilities of the four potential states as $\mathbf{v} = [v_1, v_2, v_3, v_4]^T$, where $\sum_{i=1}^4 v_i = 1$. So the expected payoffs of the platform and ISP are $U^e = \mathbf{v}^T \mathbf{P}$ and $U^h = \mathbf{v}^T \mathbf{Q}$, respectively. Here, $\mathbf{P}$ and $\mathbf{Q}$ are the payoff matrices. $\mathbf{P} = [P_1, P_2, P_3, P_4]^T = [f(a_1^e, a_1^h), f(a_1^e, a_2^h), f(a_2^e, a_1^h), f(a_2^e, a_2^h)]^T$, and $\mathbf{Q} = [Q_1, Q_2, Q_3, Q_4]^T = [g(a_1^e, a_1^h), g(a_1^e, a_2^h), g(a_2^e, a_1^h), g(a_2^e, a_2^h)]^T$.

Based on the definitions of $\mathbf{p}$ and $\mathbf{q}$, we denote the Markov state transition matrix as $\mathbf{M}$, and the stationary vector as $\mathbf{v}_s$. $\mathbf{v}_s^T \mathbf{M} = \mathbf{v}_s^T$, where

$$\mathbf{M} = \begin{bmatrix} p_1 q_1 & p_1(1-q_1) & (1-p_1)q_1 & (1-p_1)(1-q_1) \\ p_2 q_2 & p_2(1-q_2) & (1-p_2)q_2 & (1-p_2)(1-q_2) \\ p_3 q_3 & p_3(1-q_3) & (1-p_3)q_3 & (1-p_3)(1-q_3) \\ p_4 q_4 & p_4(1-q_4) & (1-p_4)q_4 & (1-p_4)(1-q_4) \end{bmatrix}.$$

where each element in $\mathbf{M}$ represents the transition probability from one state to another state. For example, the state $p_1(1-q_1)$ refers to the transition probability of the game from the state $a_1^e a_1^h$ to the state $a_1^e a_2^h$.

Inspired by [16], we suppose $\mathbf{M}' = \mathbf{M} - \mathbf{I}$, then $\mathbf{v}_s^T \mathbf{M}' = 0$. According to Cramer's rule, we obtain $\text{Adj}(\mathbf{M}')\mathbf{M}' = \det(\mathbf{M}')\mathbf{I} = 0$, where $\text{Adj}(\mathbf{M}')$ denotes the adjoint matrix. Comparing the above two equations, we obtain every row of $\text{Adj}(\mathbf{M}')$ is proportional to $\mathbf{v}_s^T$. Thus, the dot product of any vector $\mathbf{f}$ with the stationary vector $\mathbf{v}_s$ is calculated as follows,

$$\mathbf{v}_s^T \cdot \mathbf{f} = D(\mathbf{p}, \mathbf{q}, \mathbf{f}) = \det \begin{bmatrix} p_1 q_1 - 1 & p_1 - 1 & q_1 - 1 & f_1 \\ p_2 q_2 & p_2 - 1 & q_2 & f_2 \\ p_3 q_3 & p_3 & q_3 - 1 & f_3 \\ p_4 q_4 & p_4 & q_4 & f_4 \end{bmatrix}. \quad (9)$$

It's clear that the second column of (9) can be determined by the platform alone, which is denoted as $\tilde{\mathbf{p}} = (p_1 - 1, p_2 - 1, p_3, p_4)^T$. Hence, when $\mathbf{f} = \alpha \mathbf{P} + \beta \mathbf{Q} + \gamma \mathbf{1}$, where α and β are weight factors, we obtain $\mathbf{v}_s^T \cdot \mathbf{f} = \mathbf{v}_s^T \cdot (\alpha \mathbf{P} + \beta \mathbf{Q} + \gamma \mathbf{1}) = \alpha U^e + \beta U^h + \gamma$, where γ is a scalar, U^e and U^h are the expected utility in the final stable state. Based on (9), we also have $\alpha U^e + \beta U^h + \gamma = D(\mathbf{p}, \mathbf{q}, \alpha \mathbf{P} + \beta \mathbf{Q} + \gamma \mathbf{1})$. Namely, when $\tilde{\mathbf{p}} = \phi(\alpha \mathbf{P} + \beta \mathbf{Q} + \gamma \mathbf{1})$ ($\phi \neq 0$), the corresponding matrix's second column is proportional to the fourth column. Based on the properties of matrix determinant, we obtain a linear relationship between the expected payoff of two players:

$$\alpha U^e + \beta U^h + \gamma = 0. \quad (10)$$

Therefore, (10) shows U^e and U^h form a linear relationship. The strategy $\mathbf{p}$ adopted by the platform is ZDSC, which facilitates the formation of linear relationship.

From (10), for example, the expected utility U^h could be controlled if we set $\alpha = 0$. With different parameter settings of α and β , (10) can generate various strategies. Such strategies, which are called as zero-determinant strategies, are realized if the game process is formulated as a one-step Markov process.

Theorem 2: When platform's ZDSC strategy $\mathbf{p}$ satisfies

$$p_i = \begin{cases} \phi(\alpha P_i + \beta Q_i + \gamma_{\min}) + 1, & i = 1, 2, \\ \phi(\alpha P_i + \beta Q_i + \gamma_{\min}), & i = 3, 4, \end{cases} \quad (11)$$

the weighted social welfare $U_{all} = \alpha U^e + \beta U^h$ could be unilaterally controlled by the platform in a possible maximal and stable value.

Proof: We define the weighted social welfare of this game as follows:

$$U_{all} = \alpha U^e + \beta U^h = -\gamma. \quad (12)$$

The above analysis implies that when the platform adopts ZDSC strategy, the platform can control over the social welfare at a desired value no matter what strategy the ISP adopts. This provides the platform a powerful tool to maintain the stability of the total utility. The maximum and stable social welfare that the platform maintains regardless of ISP's strategy can be attained by addressing the problem below:

$$\begin{aligned} \max \quad & U_{all} = \alpha U^e(\mathbf{p}, \mathbf{q}) + \beta U^h(\mathbf{p}, \mathbf{q}), \quad \forall \mathbf{q}, \\ \text{s.t.} \quad & \begin{cases} \mathbf{0} \leq \mathbf{p} \leq \mathbf{1}, \\ \alpha U^e + \beta U^h + \gamma = 0. \end{cases} \end{aligned} \quad (13)$$

The terms $U^e(\mathbf{p}, \mathbf{q})$ and $U^h(\mathbf{p}, \mathbf{q})$ refer to the expected payoffs of the platform and ISP are determined by $\mathbf{p}$ and $\mathbf{q}$. Accordingly, it is equivalent to solve the following problem:

$$\begin{aligned} \min \quad & \gamma \\ \text{s.t.} \quad & \begin{cases} \mathbf{0} \leq \mathbf{p} \leq \mathbf{1}, \\ \tilde{\mathbf{p}} = \phi(\alpha \mathbf{P} + \beta \mathbf{Q} + \gamma \mathbf{1}), \\ \phi \neq 0. \end{cases} \end{aligned} \quad (14)$$

Next, we discuss how to calculate each element of $\mathbf{p}$ from (14); specifically, we divide the discussion by $\phi > 0$ and $\phi < 0$, respectively.

When $\phi > 0$, we feed the constraint $\mathbf{p} \geq 0$ into the second constraint of (14), then we get

$$\begin{aligned} \gamma_{\min} &= \max(\tau_i) \quad \forall i \in \{1, 2, 3, 4\}, \\ \tau_i &= \begin{cases} -\alpha P_i - \beta Q_i - 1/\phi, & i = 1, 2, \\ -\alpha P_i - \beta Q_i, & i = 3, 4. \end{cases} \end{aligned}$$

Correspondingly, we feed the constraint $\mathbf{p} \leq 1$ into the second constraint of (14) and obtain

$$\begin{aligned} \gamma_{\max} &= \min(\tau_j) \quad \forall j \in \{5, 6, 7, 8\}, \\ \tau_j &= \tau_{i+4} = \begin{cases} -\alpha P_i - \beta Q_i, & i = 1, 2, \\ -\alpha P_i - \beta Q_i + 1/\phi, & i = 3, 4. \end{cases} \end{aligned}$$

Note that γ is a feasible value only if it satisfies $\gamma_{\min} \leq \gamma_{\max}$, that is $\max(\tau_i) \leq \min(\tau_j)$, $\forall i \in \{1, 2, 3, 4\}, \forall j \in \{5, 6, 7, 8\}$. As ϕ is positive, we obtain $\gamma_{\min}$ as follows:

$$\begin{aligned} \gamma_{\min} &= \max(-\alpha P_1 - \beta Q_1 - 1/\phi, -\alpha P_2 - \beta Q_2 - 1/\phi, \\ &\quad -\alpha P_3 - \beta Q_3, -\alpha P_4 - \beta Q_4). \end{aligned} \quad (15)$$

That is $\gamma_{\min}$ is the maximal value of a set.

Similarly, when $\phi < 0$, while considering the constraint $\mathbf{p} \geq 0$, we have $\gamma_{\min} = \max(\tau_j)$, $\forall j \in \{5, 6, 7, 8\}$; while when considering $\mathbf{p} \leq 1$, we have $\gamma_{\max} = \min(\tau_i)$, $\forall i \in \{1, 2, 3, 4\}$. Thus, γ is a feasible value when $\gamma_{\min} \leq \gamma_{\max}$, that is $\max(\tau_j) \leq \min(\tau_i)$, $\forall i \in \{1, 2, 3, 4\}, \forall j \in \{5, 6, 7, 8\}$. As ϕ is negative and normalizes $\mathbf{p}$ in the range $[0, 1]$, we have the minimum value of γ as follows:

$$\begin{aligned} \gamma_{\min} &= \max(-\alpha P_1 - \beta Q_1, -\alpha P_2 - \beta Q_2, \\ &\quad -\alpha P_3 - \beta Q_3 + 1/\phi, -\alpha P_4 - \beta Q_4 + 1/\phi). \end{aligned} \quad (16)$$

According to (15) and (16), when γ reaches $\gamma_{\min}$, each element of $\mathbf{p}$ is represented as follows:

$$p_i = \begin{cases} \phi(\alpha P_i + \beta Q_i + \gamma_{\min}) + 1, & i = 1, 2, \\ \phi(\alpha P_i + \beta Q_i + \gamma_{\min}), & i = 3, 4. \end{cases} \quad (17)$$

Thus, p_i is calculated by (17), and the ZDSC strategy $\mathbf{p}$ of the platform meets $\tilde{\mathbf{p}} = \phi(\alpha \mathbf{P} + \beta \mathbf{Q} + \gamma \mathbf{1})$ ($\phi \neq 0$). $\square$

That is the ZDSC strategy $\mathbf{p}$ meets $\tilde{\mathbf{p}} = \phi(\alpha \mathbf{P} + \beta \mathbf{Q} + \gamma \mathbf{1})$. The maximal stable value refers to the stable value that ZDSC can control and reach to the possible maximal value under the stable condition.

We take the payoff matrices in Fig. 6 as an example, here $\mathbf{P} = [P_1, P_2, P_3, P_4] = [30, 20, 18, 16]$ and $\mathbf{Q} = [Q_1, Q_2, Q_3, Q_4] = [21, 28, 10, 22]$. If we set $\phi = -0.05, \alpha = \beta = 1$; In this case, $\gamma_{\min} = (-51, -48, -28 - 20, -38 - 20) = -48$, and $\gamma_{\max} = (-51 + 20, -48 + 20, -28, -38) = -28$, so $\gamma_{\min} < \gamma_{\max}$ is feasible when $\phi = -0.05$. Note that ϕ is set to guarantee each element in $\mathbf{p}$ is in the range of $[0, 1]$. According to (17), $p_1 = -0.05(51 - 48) + 1 = 0.85, p_2 = -0.05(48 - 48) + 1 = 0, p_3 = -0.05(28 - 48) = 1, p_4 = -0.05(38 - 48) = 0.5$. So the ZDSC strategy is represented as $\mathbf{p} = (0.85, 0, 1, 0.5)$; when $\mathbf{p}$ is adopted by the platform in the iterated game, the social welfare can achieve to a maximum stable value $-\gamma_{\min} = 48$ in the long term.

2) *ZDSC With Multi-Actions*: Based on the detailed analysis of ZDSC of the player with two actions, it is easy to extend ZDSC into player with multi-actions.

We denote there are n actions of platform and ISP, where $a^e \in A^e, a^h \in A^h$ and $|A^e| = |A^h| = n$. In each game iteration, we define the mixed strategy $\mathbf{p}$ of platform is denoted as:

$$\mathbf{p} = [p_{1,1}, p_{2,1}, \dots, p_{n,1}, p_{1,2}, \dots, p_{j,s}, \dots, p_{n,n^2}]^T, \quad (18)$$

where $p_{j,s}$ ($j \in \{1, 2, \dots, n\}, s \in \{1, 2, \dots, n^2\}$) refers to the probability that the platform chooses the action $a^e = j$ and the ISP chooses the action a^h the same as the s -th outcome of the previous iteration. The utility vector of platform is defined as:

$$\mathbf{u}^e = [u_{1,1}^e, u_{2,1}^e, \dots, u_{n,1}^e, u_{1,2}^e, u_{2,2}^e, \dots, u_{j,s}^e, \dots, u_{n,n^2}^e]^T, \quad (19)$$

where $u_{j,s}^e$ is the utility that platform chooses the action $a^e = j$ in the s -th outcome, and can be calculated by $f(\mathbf{a}^{(j,s)}), \mathbf{a}^{(j,s)}$ denotes the action pair $\mathbf{a}$ corresponding to s -th action-pair that $a^e = j$. The utility vector of ISP is denoted as $\mathbf{u}^h$.

Similar as the deduction in Section V-B, for the stationary vector $\mathbf{v}$, the platform's expected utility in the stationary state is:

$$U^e = \frac{\mathbf{v}^T \cdot \mathbf{u}^e}{\mathbf{v}^T \cdot \mathbf{1}} = \frac{\det(\mathbf{p}, \mathbf{q}, \mathbf{u}^e)}{\det(\mathbf{p}, \mathbf{q}, \mathbf{1})}. \quad (20)$$

Then, from (20), we obtain the linear combination of the sum expected utilities of platform and ISP as follows:

$$\alpha U^e + \beta U^h + \gamma = \frac{\det(\mathbf{p}, \mathbf{q}, \alpha \mathbf{u}^e + \beta \mathbf{u}^h + \gamma \mathbf{1})}{\det(\mathbf{p}, \mathbf{q}, \mathbf{1})}, \quad (21)$$

where α and β are the weighted coefficients of U^e and U^h , respectively. γ is a variable in a range.

When the platform sets its strategy $\mathbf{p}$ that satisfies $\tilde{\mathbf{p}} = \phi(\alpha \mathbf{u}^e + \beta \mathbf{u}^h + \gamma \mathbf{1})$, where ϕ is a non-zero coefficient. $\tilde{\mathbf{p}}^i$ is

proportional to the last column of $\det(\mathbf{p}, \mathbf{q}, \alpha \mathbf{u}^e + \beta \mathbf{u}^h + \gamma \mathbf{1})$, so the (21) is written as:

$$\alpha U^e + \beta U^h + \gamma = 0, \quad (22)$$

that is $\alpha U^e + \beta U^h = -\gamma$. The elements in $\tilde{\mathbf{p}}$ can be represented by the elements in $\mathbf{p}$:

$$\tilde{\mathbf{p}} = [-1 + p_{1,1}, -1 + p_{2,1}, \dots, -1 + p_{n,1}, -1 + p_{1,2}, \dots, -1 + p_{n,n}, p_{1,n+1}, \dots, p_{n,n^2}]. \quad (23)$$

Let the platform take ZDSC strategy, and the social welfare is denoted as $\alpha U^e + \beta U^h = -\gamma$. To obtain the stable and possible maximum social welfare regardless of the ISP's strategy, we need to solve the following constrained optimization problem:

$$\begin{aligned} \min \quad & \gamma \\ \text{s.t.} \quad & \begin{cases} 0 \leq p_{j,s} \leq 1, j \in \{1, 2, \dots, n\}, s \in \{1, 2, \dots, n^2\} \\ \tilde{\mathbf{p}} = \phi(\alpha \mathbf{u}^e + \beta \mathbf{u}^h + \gamma \mathbf{1}), \\ \phi \neq 0. \end{cases} \end{aligned} \quad (24)$$

To simplify the description, we denote the k -th element in $\mathbf{u}^e$ as $u_k^e, k \in \{1, 2, \dots, n^2\}$. ϕ is a scale coefficient that controls the convergence rate to the stable state. In the social welfare control, the value of ϕ impacts the calculation of $\gamma_{\max}$ and $\gamma_{\min}$. Thus we discuss the optimization problem by dividing $\phi > 0$ and $\phi < 0$, respectively.

Case 1: $\phi > 0$: When ϕ is positive, in order to satisfy the constraint $p_{j,s} \geq 0$, the lower bound of γ is:

$$\begin{aligned} \gamma_{\min} &= \max(Y_k) \\ Y_k &= \begin{cases} -\frac{1}{\phi} - \alpha u_k^e - \beta u_k^h, k = 1, 2, \dots, n, \\ -\alpha u_k^e - \beta u_k^h, k = n + 1, \dots, n^2, \end{cases} \end{aligned} \quad (25)$$

where Y_k is a set including n^3 elements.

Next we consider the constraint $p_{j,s} \leq 1$ and obtain the upper bound of γ :

$$\begin{aligned} \gamma_{\max} &= \min(T_l) \\ T_l &= \begin{cases} -\alpha u_k^e - \beta u_k^h, k = 1, 2, \dots, n, \\ \frac{1}{\phi} - \alpha u_k^e - \beta u_k^h, k = n + 1, \dots, n^2, \end{cases} \end{aligned} \quad (26)$$

where T_l is a set including n^3 elements.

Note that γ is a feasible value only if it satisfies $\gamma_{\min} \leq \gamma_{\max}$, that is $\max(Y_k) \leq \min(T_l), \forall k, l \in \{1, 2, \dots, n^3\}$. As ϕ is a positive value, we obtain $\gamma_{\min}$ as follows:

$$\gamma_{\min} = \max \left\{ -\frac{1}{\phi} - \alpha u_1^e - \beta u_1^h, \dots, -\frac{1}{\phi} - \alpha u_n^e - \beta u_n^h, -\alpha u_{n+1}^e - \beta u_{n+1}^h, \dots, -\alpha u_{n^2}^e - \beta u_{n^2}^h \right\}. \quad (27)$$

That is, $\gamma_{\min}$ is the maximal value of the set Y_k , and $\gamma_{\max}$ is the minimal value of the set T_l .

Case 2: $\phi < 0$: Similarly, when ϕ is negative, we consider $p_{j,s} \geq 0$ and obtain $\gamma_{\min} = \max(T_l), l \in \{1, 2, \dots, n^3\}$. Taking $p_{j,s} \leq 1$, we obtain $\gamma_{\max} = \min(Y_k), k \in \{1, 2, \dots, n^3\}$. γ is a feasible value when $\gamma_{\min} < \gamma_{\max}$, that is $\max(T_l) \leq \min(Y_k)$,

$\forall k, l \in \{1, 2, \dots, n^3\}$. So we obtain the following result:

$$\gamma_{\min} = \max \left\{ -\alpha u_1^e - \beta u_1^h, \dots, -\alpha u_n^e - \beta u_n^h, \right. \\ \left. \frac{1}{\phi} - \alpha u_{n+1}^e - \beta u_{n+1}^h, \dots, \frac{1}{\phi} - \alpha u_{n^3}^e - \beta u_{n^3}^h \right\}. \quad (28)$$

In this case, $\gamma_{\min}$ is the maximal value of the set T_l , and $\gamma_{\max}$ is the minimal value of the set Y_k .

Based on the above analysis, the platform adopts ZDSC strategy $\mathbf{p}$ and can control the expected social welfare at a stable and maximal value $\gamma_{\min}$, where $\mathbf{p}$ satisfies $\tilde{\mathbf{p}} = \phi(\alpha \mathbf{u}^e + \beta \mathbf{u}^h + \gamma \mathbf{1})$, and each element in $\mathbf{p}$ is calculated as follows:

$$p_m = \begin{cases} \phi(\alpha u_m^e + \beta u_m^h + \gamma_{\min}) + 1, & m=1, 2, \dots, n, \\ \phi(\alpha u_m^e + \beta u_m^h + \gamma_{\min}), & m=n+1, \dots, n^3, \end{cases} \quad (29)$$

where p_m refers to the m -th element in $\mathbf{p}$.

In practice, when the platform is chosen to adopt the ZDSC strategy, the platform owns the following prior knowledge: (1) The platform has the global view of the action space of its own and ISP. (2) The payoff functions of the platform and ISP is known. the platform utilizes its prior knowledge to calculate each element (probability) of the ZDSC strategy, then it selects the action by the probability based on the previous outcome in the iterated game.

In sum, for the discrete case, when the elements in the ZDSC strategy $\mathbf{p}$ adopted by the platform are set by (17) in the two-action case, or (29) in the multi-action case, social welfare can be unilateral controlled by the ZDSC strategy $\mathbf{p}$ and reaches a possible maximal and stable value, without considering the ISP's strategy.

VI. DISCUSSION

This section first discusses RLSC and ZDSC together. Then we analyze a possible method to escape a local optimal in the case of two RLSC players playing against each other.

A. RLSC and ZDSC Strategies

The strategy precalculated by (17) and (29) can unilaterally control over the social welfare at a possible maximal and stable value no matter what strategy the ISP adopts. The strategy obtained from reinforcement learning [37] to maximize their utility may fall into a local maximum; because the initial policies of two players are set randomly, which may impact the final learned policies of players.

Apart from the impact of the initial policy of two RLSC players, the learning rate η is also a factor that impacts the coverage of the policies, which may lead to a local maximum. Here is a solution to help RLSC players escape a local optimal. When two RLSC players play against each other, each time the policies π^e , π^h converge, we save the initial policy, learning rate, the converged policies, and the value of social welfare. We then restart the optimization with a large learning rate to escape the current local maximum [59], [60]. After a few iterations, we obtain the policies π^e and π^h with maximal social welfare. This

idea will jump out of the local optimum to find the maximum value, but it is not guaranteed to find the maximum value.

To the best of our knowledge, we are the first to investigate RLSC plays against ZDSC in the iterated game, and prove ZDSC can control social welfare even if the opponent adopts RLSC. The iterated game of the platform and ISP in spatial crowdsourcing can be treated as an extension of Markov Decision Process to the multi-agent case, which can be defined by Markov Games. The solutions about strategies RLSC and ZDSC have similarities and differences. The similarity is both of them care about the long run expected utility. The differences between them are: (1) The goal of RLSC is trying to update its strategy and maximize its expected rewards in the long run. (2) ZDSC derives a linear relationship between the expected payoff of the platform and that of the ISP, and it focuses on the unilateral control for social welfare to reach a stable and a possible maximal value.

B. Complexity Analysis of ZDSC and RLSC Scheme

The time and space complexity of the ZDSC scheme is $O(T)$ and $O(|A^e||A^h|)$, respectively. The time and space complexity of RLSC scheme is $O(T)$ and $O(|A^e||A^h|^2)$, respectively. The details are explained as follows:

Time complexity of ZDSC scheme: The time complexity of ZDSC scheme is determined by two main components: computing the strategy and executing the strategy during the game. (1) Computing the ZD Strategy: ZDSC strategies involve setting up a linear relationship between the players' payoffs. This relationship can be defined by a set of linear equations with a few parameters. Solving these equations involves basic linear algebra operations, and because there are only a few parameters, this can be done in constant time, resulting in a time complexity of $O(1)$. (2) Executing the Strategy: In each iteration of the game, the ZDSC strategy uses the set parameters to calculate the probabilities for the player's actions based on the game's history. These calculations involve simple arithmetic operations, which are also done in constant time, $O(1)$, per iteration. If the game is repeated T times, the total time complexity becomes $O(T)$ because each iteration requires a constant amount of time. Overall time complexity of ZDSC scheme is $O(T)$, where T is the number of iterations of the repeated game.

Space complexity of ZDSC scheme: The space complexity of ZDSC strategies is largely determined by the amount of information the strategy needs to store, which includes: (1) Strategy parameters and payoff matrices: The ZDSC strategy requires storing constant number of parameters (e.g., α , β) and payoff matrices. This takes $O(|A^e||A^h|)$ space. (2) Game history: In some implementations, ZDSC strategies need to keep track of the opponent's actions over the immediate past round to make decisions. However, this still only requires a constant amount of space because it does not depend on the total number of iterations. Therefore, the space complexity remains $O(1)$. Overall space complexity of ZDSC scheme is $O(|A^e||A^h|)$.

Time complexity of RLSC scheme: Time Complexity of RLSC scheme is $O(T)$, where T is the number of iterations. In each iteration, Q-Learning updates the Q-value for a state-action pair based on the reward received and the maximum Q-value of the next state. This update operation takes $O(1)$ time. However,

the total time complexity depends on the number of updates. Therefore, the total time complexity for T iterations is $O(T)$.

Space complexity of RLSC scheme: The space complexity of Q-Learning primarily depends on maintaining the Q-table: (1) Q-Table size: Q-Learning stores a Q-value for each state-action pair, resulting in a Q-table with a size of $|A^e||A^h|^2$. (2) Additional variables: Apart from the Q-table, Q-Learning uses a few other variables like the learning rate, the discount factor, the current state, and the reward. These take up constant space $O(1)$. Thus, the overall space complexity of RLSC is: $O(|A^e||A^h|^2)$.

VII. EXPERIMENTS

This section reports the experimental settings, comparison methods and evaluation results.

A. Experiment Setup

We validate the RLSC and ZDSC based methods in a vehicular crowdsourcing scenario as a case study. We perform experiments on the Taxi-Roma dataset [52]. The dataset includes the information about the GPS coordinates of 320 taxi drivers that work in the center of Rome. These GPS coordinates are collected over 30 days. The dataset is preprocessed to filter out some outliers. The traces cover an area with a range of $66\text{km} \times 59\text{km}$, and we divide the area into 10×10 grids. We assume each RSU in the grid serving as a gateway, each grid is viewed as an area with a sensing task. We calculate the priority of the sensing tasks by the number of GPS coordinates in the grid, the top a^e sensing tasks are selected, a^r RSUs are selected by a matching algorithm. From the traces analysis, we obtain the task-RSU weighted traffic matrix, each element in it is the number of sensing data uploaded to an RSU for a task. In the discrete model, we set both the lowest and highest numbers of task and RSU as 2 and 8. That is $\forall a^e, a^h \in \{2, 3, 4, 5, 6, 7, 8\}$. The parameters are set as $\alpha = \beta = 1$, $u_d = 30$, $u_t = 1$, $u_e = 0.01$, $u_h = 50$, $C = 500$. We implement the RLSC and ZDSC strategy in Python 3.8. All experiments were conducted on a computer with Intel Core i7-6700 CPU and 8 G RAM.

B. Comparison Methods

We describe here the baselines for comparisons.

- *ALLC* [17], [19], [21]: It is an all cooperation strategy $\mathbf{p} = [1, 1, \dots, 1]$, which means whatever the opponent player has done in the previous round, it always chooses the cooperate action.
- *ALLD* [17], [21]: It is an all defection strategy, that is $\mathbf{p} = [0, 0, \dots, 0]$, which means whatever the opponent has done in the previous round, it always chooses defection.
- *Random* [17], [19], [21]: It is a random strategy $\mathbf{p} = [0.5, 0.5, \dots, 0.5]$, which means whatever the opponent has done in the previous round, it randomly chooses cooperate action.
- *TFT (tit-for-tat)* [19], [61]: A TFT player cooperates in the first round and then does whatever the opponent player has done in the previous round. Hence, TFT strategy embodies

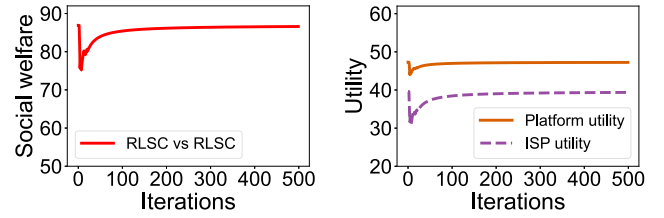


Fig. 7. Global optimal of RLSC vs. RLSC.

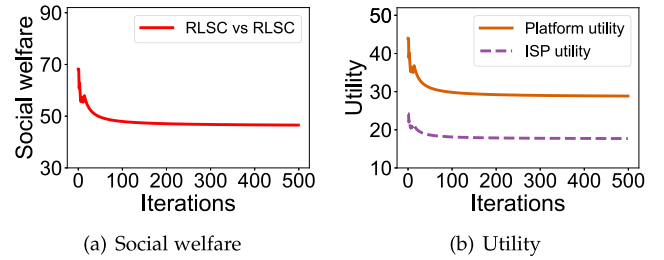


Fig. 8. Local maximum of RLSC vs. RLSC.

the simple principle “an eye for an eye and a tooth for a tooth.”

- *Evolved* [16]: it’s an evolved strategy that an evolutionary player starts his strategy $\mathbf{q} = [0, 0, \dots, 0]$ and his opponent adopts ZDSC strategy in the iterated games. $\mathbf{q}$ is updated in each iteration and will be stable at the end of the iterations; it is a special case with $\theta = 0$, that is the player only considers the current reward when it updates strategy.
- *Pavlov (win-stay-lose-shift strategy)* [19], [62]: If a Pavlov player receives a higher payoff, it will repeat the same action next round, this is “win-stay.” If Pavlov player receives a lower payoff, it will switch to the opposite action, this is “lose-shift.”

C. Performance Evaluation

In this subsection, we first evaluate the RLSC player plays against RLSC, simple strategies, and classical strategies. Then we evaluate ZDSC player plays against simple strategies and classical strategies. Finally, we evaluate the performance of RLSC player plays against ZDSC player.

1) *Performance of RLSC:* We validate the performance of RLSC, by setting the learning rate $\eta = 0.005$ and the discount factor $\theta = 0.5$, which are the empirical values and set based on the specific application scenarios [42]. We perform the experiments when both the platform and ISP adopt RLSC. Figs. 7(a) and 8(a) show the global and local maximum social welfare, respectively. Figs. 7(b) and 8(b) show the expected utility of the platform and ISP in the two cases, which shows the utility may fall into a local maximum. The reason is that the initializations of RLSC adopted by the platform and ISP are random, their policies are dynamically updated in each round, an RLSC player aims to maximize his own rewards that may lead to a local maximum social welfare. If the platform takes RLSC, the ISP adopts other strategies, i.e., ALLC, ALLD, Random, TFT, Pavlov; As shown in Fig. 9, the social welfare is determined by the strategies of

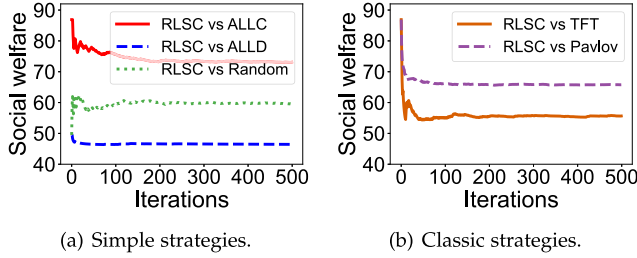


Fig. 9. Social welfare in RLSC solution (platform vs. ISP).

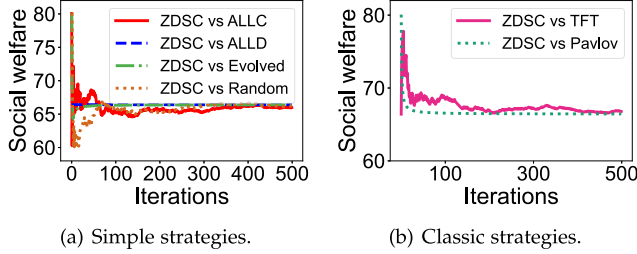


Fig. 10. Social welfare in ZDSC solution (platform vs. ISP).

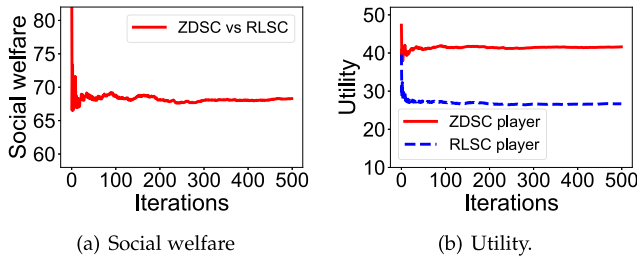


Fig. 11. Platform adopts ZDSC vs. ISP adopts RLSC.

both the platform and the ISP, indicating that the RLSC can not dominate the control the social welfare over other strategies; because RLSC player aims to maximize his own rewards in the game.

2) *Performance of ZDSC*: According the definition of social welfare in (4), which represents the sum of platform and ISP's payoffs when we set $\alpha = \beta = 1$. (α and β can be set as other values, which represent different goals of social welfare control). Then we compare the ZDSC with six other strategies that ISP may adopt. We display the results when the platform adopts ZDSC, and ISP adopts all cooperation ($\mathbf{p} = [1, 1, \dots, 1]$, denoted as ALLC), all defection ($\mathbf{p} = [0, 0, \dots, 0]$, denoted as ALLD), random ($\mathbf{p} = [0.5, 0.5, \dots, 0.5]$, denoted as Random), TFT and Pavlov strategies. Moreover, we also adopt an evolved strategy to ISP, which is a special case of RLSC when the discount factor $\theta = 0$, denoted as it only considers the current reward. We view ISP as an evolutionary player adopted evolved strategy. Fig. 10 shows when the platform adopts ZDSC, whatever the ISP adopts, the social welfare is maintained stable and achieves its maximum, which means the ZDSC unilaterally controls the social welfare and reaches to the maximum as much as possible. The social welfare of RLSC vs. ALLC in Fig. 11(a) is larger than that of Fig. 10, because the ALLC player always chooses cooperation, and the RLSC player updates policy to maximize

its reward, which yields their cooperation and achieve global maximal social welfare. Moreover, the social welfare controlled by ZDSC does not fall into a local maximum. We assume ISP as an evolutionary player, and he doesn't know the ZDCS strategy. So the case that both the platform and ISP adopt ZDCS is out the scope of this paper.

3) *Performance of RLSC Vs. ZDSC*: We investigate the performance when the platform adopts ZDSC and ISP adopts RLSC. Fig. 11 shows the results. We compare the results together with Fig. 10, which shows ZDSC adopted by the platform can unilaterally control and achieve a high and stable social welfare in long run, regardless of ISP's strategies. Thus, Figs. 10 and 11 represent ZDSC strategy can control the social welfare. The social welfare that ZDSC unilaterally controlled in Figs. 10 and 11 are smaller than that of Fig. 7(a), because Fig. 7(a) adopts two RLSC players in the game, and each of them updates their policy dynamically to maximize its reward; while a precalculated ZDSC is to achieve social welfare control, the theoretical social welfare can be precalculated, but RLSC can not. In summary, when a ZDSC player plays against a player adopted other strategies in the iterated game, ZDSC player can unilaterally control social welfare without considering the ISP's strategies, even the strategy is RLSC. That is to say, ZDSC can control social welfare in the iterated game on the whole. But the shortcoming of ZDSC is the controlled stable social welfare may not be the global maximum.

D. Discussion of Scalability

The iterated game can be extended to a continuous action space from a discrete action space. The continuous action space of each player can be divided into multiple small ranges, the space is continuous when the range is small enough. In the continuous case, ZDSC is a precalculated function to obtain ZDSC player's action in each round. However, the complexity of RLSC is larger than that of ZDSC because of the large state and space.

The ZDSC can unilaterally control over the social welfare at a possible maximal and stable value no matter what strategy the ISP adopts. The strategy obtained from RL [37] may achieve maximize utility sometimes, but it may fall into a local maximum sometimes; because the initial policy of two RLSC payers are set randomly, which impact the final learned policies. In this paper, we do not deny the effectiveness of RLSC, and just analyze the two kinds of strategies in the iterated game. If the state and action space are known in advance, that means it is a controllable game, and RLSC is dominated by ZDSC in this case. But if unknown states and actions exist, RLSC performs better than ZDSC, which will be discussed in our future work.

VIII. CONCLUSION

In this paper, we introduce the iterated game between the platform and ISP under spatial crowdsourcing, and address the social welfare control problem in this game. To the best of our knowledge, we are the first to investigate an RLSC player plays against ZDSC player in the iterated game, and prove ZDSC unilaterally controls the social welfare with a possible

maximal stable value regardless of ISP's strategies in theory and experiment. Experimental results verify that the platform using ZDSC controls the social welfare with a possible maximal and stable value; while RLSC sometimes achieves the global maximum and sometimes may lead to a local maximum. In the future, we will explore applications on the variants of ZDSC, and investigate the continuous action space of ZDSC.

REFERENCES

- [1] C. Song and J. Wu, "Vehicular Ad Hoc/sensor networks in smart cities," in *Mission-Oriented Sensor Networks and Systems: Art and Science*. Berlin, Germany: Springer, 2019, pp. 287–310.
- [2] Z. Pan, H. Yu, C. Miao, and C. Leung, "Crowdsensing air quality with camera-enabled mobile devices," in *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 4728–4733.
- [3] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: Current state and future challenges," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 32–39, Nov. 2011.
- [4] B. Guo, H. Chen, Z. Yu, X. Xie, S. Huangfu, and D. Zhang, "Fliermeet: A mobile crowdsensing system for cross-space public information reposting, tagging, and sharing," *IEEE Trans. Mobile Comput.*, vol. 14, no. 10, pp. 2020–2033, Oct. 2015.
- [5] H. Ma, D. Zhao, and P. Yuan, "Opportunities in mobile crowd sensing," *IEEE Commun. Mag.*, vol. 52, no. 8, pp. 29–35, Aug. 2014.
- [6] V. Ayu, "Revisiting mobile crowdsensing: An open challenge," in *Proc. Conf. Comput. Sci. Inf. Technol.*, 2021, pp. 93–101. doi: 10.5121/csit.2021.111509.
- [7] F. Wamser, A. Seufert, A. Hall, S. Wunderer, and T. Hofffeld, "Valid statements by the crowd: Statistical measures for precision in crowdsourced mobile measurements," *Network*, vol. 1, no. 2, pp. 215–232, 2021.
- [8] M. C. Lucic, X. Wan, H. Ghazai, and Y. Massoud, "Leveraging intelligent transportation systems and smart vehicles using crowdsourcing: An overview," *Smart Cities*, vol. 3, no. 2, pp. 341–361, 2020.
- [9] X. Yang, B. Gu, B. Zheng, B. Ding, Y. Han, and K. Yu, "Toward incentive-compatible vehicular crowdsensing: An edge-assisted hierarchical framework," *IEEE Netw.*, vol. 36, no. 2, pp. 162–167, Mar./Apr. 2022.
- [10] Z. Rashidi, V. Hakami, P. Geranmayeh, and S. Rajaei, "Multi-agent learning algorithms for content placement in cache-enabled small cell networks: 4G and 5G use cases," *Neural Comput. Appl.*, vol. 34, pp. 11641–11668, 2022.
- [11] R. Esmailyfard and R. Esmaili, "A privacy-preserving mechanism for social mobile crowdsensing using game theory," *Trans. Emerg. Telecommun. Technol.*, vol. 33, 2022, Art. no. e4517.
- [12] S. D. Martino and L. L. L. Starace, "Vehicular crowd-sensing on complex urban road networks: A case study in the city of porto," *Transp. Res. Procedia*, vol. 62, pp. 350–357, 2022.
- [13] D. Agarwal et al., "Understanding driver-passenger interactions in vehicular crowdsensing," *Proc. ACM Hum.-Comput. Interaction*, vol. 5, no. CSCW2, pp. 1–24, 2021.
- [14] C. Piao and C. H. Liu, "Energy-efficient mobile crowdsensing by unmanned vehicles: A sequential deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6312–6324, Jul. 2020.
- [15] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," in *Handbook of Reinforcement Learning and Control*, vol. 325, Berlin, Germany: Springer, 2021, pp. 321–384.
- [16] W. H. Press and F. J. Dyson, "Iterated prisoner's dilemma contains strategies that dominate any evolutionary opponent," in *Proc. Nat. Acad. Sci. USA*, vol. 109, no. 26, 2012, pp. 10409–10413.
- [17] Q. Hu, S. Wang, R. Bie, and X. Cheng, "Social welfare control in mobile crowdsensing using zero-determinant strategy," *Sensors*, vol. 17, no. 5, 2017, Art. no. 1012.
- [18] Q. Hu, S. Wang, L. Ma, R. Bie, and X. Cheng, "Anti-malicious crowdsourcing using the zero-determinant strategy," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst.*, 2017, pp. 1137–1146.
- [19] C. Tang, X. Li, M. Cao, Z. Zhang, and X. Yu, "Incentive mechanism for macrotasking crowdsourcing: A zero-determinant strategy approach," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8589–8601, Oct. 2019.
- [20] Q. Hu, S. Wang, P. Ma, X. Cheng, W. Lv, and R. Bie, "Quality control in crowdsourcing using sequential zero-determinant strategies," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 5, pp. 998–1009, May 2020.
- [21] H. Zhang, D. Niyato, L. Song, T. Jiang, and Z. Han, "Zero-determinant strategy for resource sharing in wireless cooperations," *IEEE Trans. Wireless Commun.*, vol. 15, no. 3, pp. 2179–2192, Mar. 2016.
- [22] Q. Hu, S. Wang, and X. Cheng, "A game theoretic analysis on block withholding attacks using the zero-determinant strategy," in *Proc. Int. Symp. Qual. Service*, 2019, pp. 1–10.
- [23] H. Zhang, D. Niyato, L. Song, T. Jiang, and Z. Han, "Equilibrium analysis for zero-determinant strategy in resource management of wireless network," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2015, pp. 2002–2007.
- [24] X. Chen, C. Wang, Z. Zhou, and K. W. Ross, "Randomized ensembled double Q-learning: Learning fast without a model," in *Proc. Int. Conf. Learn. Representations*, 2021.
- [25] E. Keremoglu and N. B. Weidmann, "How dictators control the internet: A review essay," *Comp. Political Stud.*, vol. 53, no. 10–11, pp. 1690–1703, 2020.
- [26] Y. Tong, Z. Zhou, Y. Zeng, L. Chen, and C. Shahabi, "Spatial crowdsourcing: A survey," *VLDB J.*, vol. 29, no. 1, pp. 217–250, 2020.
- [27] T. Ren, X. Zhou, K. Li, Y. Gao, J. Zhang, and K. Li, "Efficient cross dynamic task assignment in spatial crowdsourcing," in *Proc. IEEE 39th Int. Conf. Data Eng.*, 2023, pp. 1420–1432.
- [28] L. Pu, X. Chen, G. Mao, Q. Xie, and J. Xu, "Chimera: An energy-efficient and deadline-aware hybrid edge computing framework for vehicular crowdsensing applications," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 84–99, Feb. 2019.
- [29] F. Morselli, F. Zabini, and A. Conti, "Environmental monitoring via vehicular crowdsensing," in *Proc. IEEE 29th Annu. Int. Symp. Pers. Indoor Mobile Radio Commun.*, 2018, pp. 1382–1387.
- [30] F. Campioni, S. Choudhury, K. Salomaa, and S. G. Akl, "Improved recruitment algorithms for vehicular crowdsensing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1198–1207, Feb. 2019.
- [31] Y. Zhao et al., "Preference-aware task assignment in spatial crowdsourcing," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 2629–2636.
- [32] Gigwalk, 2022. [Online]. Available: <http://www.gigwalk.com>
- [33] Uber, 2022. [Online]. Available: <https://www.uber.com/>
- [34] Waze, 2022. [Online]. Available: <http://www.waze.com/>
- [35] OpenStreetMap, 2022. [Online]. Available: <https://www.openstreetmap.org/>
- [36] H. Wang, W. Liu, A. Liu, T. Wang, H. Song, and S. Zhang, "Sqcs: A sustainable quality control system for spatial crowdsourcing via three-party evolutionary game: Theory and practice," *Expert Syst. Appl.*, vol. 238, 2024, Art. no. 122132.
- [37] L. Xiao, T. Chen, C. Xie, H. Dai, and H. V. Poor, "Mobile crowdsensing games in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1535–1545, Feb. 2018.
- [38] A. Di Stefano, M. Scatà, B. Attanasio, A. La Corte, P. Liò, and S. K. Das, "A novel methodology for designing policies in mobile crowdsensing systems," *Pervasive Mobile Comput.*, vol. 67, 2020, Art. no. 101230.
- [39] A. Chakeri, X. Wang, Q. Goss, M. I. Akbas, and L. G. Jaimes, "A platform-based incentive mechanism for autonomous vehicle crowdsensing," *IEEE Open J. Intell. Transp. Syst.*, vol. 2, pp. 13–23, 2021.
- [40] L. Xiao, Y. Li, G. Han, H. Dai, and H. V. Poor, "A secure mobile crowdsensing game with deep reinforcement learning," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 1, pp. 35–47, Jan. 2018.
- [41] K. Lou, Y. Yang, E. Wang, Z. Liu, T. Baker, and A. K. Bashir, "Reinforcement learning based advertising strategy using crowdsensing vehicular data," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4635–4647, Jul. 2021.
- [42] W. Liu, L. Wang, E. Wang, Y. Yang, D. Zeghlache, and D. Zhang, "Reinforcement learning-based cell selection in sparse mobile crowdsensing," *Comput. Netw.*, vol. 161, pp. 102–114, 2019.
- [43] Y. Chen and H. Wang, "Intelligentcrowd: Mobile crowdsensing via multi-agent reinforcement learning," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 5, no. 5, pp. 840–845, Oct. 2021.
- [44] C. Hilbe, M. A. Nowak, and K. Sigmund, "Evolution of extortion in iterated prisoner's dilemma games," *Proc. Nat. Acad. Sci. USA*, vol. 110, no. 17, pp. 6913–6918, 2013.
- [45] W. Liu, Y. Yang, E. Wang, and J. Wu, "Dynamic user recruitment with truthful pricing for mobile crowdsensing," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 1113–1122.
- [46] E. Wang, Y. Yang, J. Wu, K. Lou, D. Luan, and H. Wang, "User recruitment system for efficient photo collection in mobile crowdsensing," *IEEE Trans. Human-Mach. Syst.*, vol. 50, no. 1, pp. 1–12, Feb. 2020.
- [47] A. Okic, L. Zanzi, V. Sciancalepore, A. Redondi, and X. Costa-Pérez, "π-road: A learn-as-you-go framework for on-demand emergency slices in v2x scenarios," in *Proc. IEEE Conf. Comput. Commun.*, 2021, pp. 1–10.

- [48] C. Liang, Y. He, F. R. Yu, and N. Zhao, "Energy-efficient resource allocation in software-defined mobile networks with mobile edge computing and caching," in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2017, pp. 121–126.
- [49] N. Lyamin, A. Vinel, M. Jonsson, and J. Loo, "Real-time detection of denial-of-service attacks in IEEE 802.11 p vehicular networks," *IEEE Commun. Lett.*, vol. 18, no. 1, pp. 110–113, Jan. 2014.
- [50] T.-J. Wu, W. Liao, and C.-J. Chang, "A cost-effective strategy for road-side unit placement in vehicular networks," *IEEE Trans. Commun.*, vol. 60, no. 8, pp. 2295–2303, Aug. 2012.
- [51] C. Zhou, C.-K. Tham, and M. Motani, "Online auction for scheduling concurrent delay tolerant tasks in crowdsourcing systems," *Comput. Netw.*, vol. 169, 2020, Art. no. 107045.
- [52] L. Bracciale et al., "CRAWDAD data set roma/taxi (v. 2014-07-17)," 2014. [Online]. Available: <http://crawdad.org/roma/taxi/>
- [53] W. Chen, Z. Su, Q. Xu, T. H. Luan, and R. Li, "VFC-based cooperative UAV computation task offloading for post-disaster rescue," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 228–236.
- [54] J. Xu, S. Yang, W. Lu, L. Xu, and D. Yang, "Incentivizing for truth discovery in edge-assisted large-scale mobile crowdsensing," *Sensors*, vol. 20, no. 3, 2020, Art. no. 805.
- [55] Z. Zheng, Y. Peng, F. Wu, S. Tang, and G. Chen, "Trading data in the crowd: Profit-driven data acquisition for mobile crowdsensing," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 2, pp. 486–501, Feb. 2017.
- [56] J. F. Nash et al., "Equilibrium points in n-person games," *Proc. Nat. Acad. Sci. USA*, vol. 36, no. 1, pp. 48–49, 1950.
- [57] T. Burkhardt, A. Leopold, and U. Leopold-Wildburger, "Markov simulation of an iterated prisoners' dilemma experiment," in *Proc. Int. Conf. Operations Res.*, 2012, pp. 223–228.
- [58] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
- [59] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hoppercroft, and K. Q. Weinberger, "Snapshot ensembles: Train 1, get m for free," in *Proc. Int. Conf. Learn. Representations, Poster*, 2017.
- [60] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in *Proc. Int. Conf. Learn. Representations*, 2017.
- [61] N. Lacey, *The Prisoners' Dilemma*. Cambridge, U.K., Cambridge Univ. Press, 2008.
- [62] L. A. Imhof, D. Fudenberg, and M. A. Nowak, "Tit-for-tat or win-stay, lose-shift?," *J. Theor. Biol.*, vol. 247, no. 3, pp. 574–580, 2007.



Jiqing Gu received the PhD degree in computer science from the University of Electronic Science and Technology of China (UESTC), China, in 2022. In 2021, she was a visiting student with the University of Toronto, under the supervision of Dr. Baochun Li. She is currently a lecturer with the School of Computer Science, Chengdu University of Information Technology. She has authored or coauthored in AAAI, ACM TKDD, KBS, TKDE, JCST, IEEE MSN, IEEE ICPADS, and IEEE ISPA. Her research interests include crowdsensing, recommender system, graph stream mining.



Chao Song (Member, IEEE) received the PhD degree in computer science from the University of Electronic Science and Technology of China (UESTC), China, in 2009. In 2013, he was a visiting scholar with Temple University, under the supervision of Dr. Jie Wu. He is currently an associate professor with the School of Computer Science and Engineering, UESTC. He has authored or coauthored more than 40 papers in referred journals and conferences, including AAAI, IEEE PerCom, IEEE GlobeCom, IEEE MASS, IEEE ICPADS, IEEE MSN, ACM TKDD, Journal of Parallel and Distributed Computing, and Computer Communications. His research interests include mobile computing, vehicular networks and distributed computing. He is a member of the IEEE Communication Society.



Jie Wu (Fellow, IEEE) is currently chief scientist of China Telecom and Director of China Telecom Cloud Computing Research Institute. He was Laura H. Carnell Professor with Temple University and the Director of the Center for Networked Computing (CNC). His research interests include mobile computing and wireless networks, routing protocols, network trust and security, distributed algorithms, applied machine learning, and cloud computing. He serves on several editorial boards, including *IEEE/ACM Transactions on Networking*. He is/was general chair/co-chair for IEEE IPDPS'23, ACM MobiHoc'23, and IEEE CCGrid'24 as well as program chair/cochair for IEEE INFOCOM'11 and CCF CNCC'13. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). He is a Fellow of the AAAS. He was a recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award. He is a member of the Academia Europaea.



Li Lu (Member, IEEE) received the BE and MS degrees in automation control from Zhejiang University, Hangzhou, China, in 2000 and 2003, respectively, and the PhD degree from the Key Laboratory of Information Security, Chinese Academy of Science, Beijing, China, in 2007. He is a professor with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China. He has authored or coauthored more than 80 papers in journals and conferences, including ACM Mobicom, USENIX NSDI, *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Wireless Communications*, IEEE Infocom, IEEE PerCom, IEEE RFID, IEEE IPDPS, and IEEE ICPADS. His research interests include battery-free systems and RFID technology, wireless networks, and network security. He is a member of the IEEE Communication Society and ACM.



Ming Liu (Member, IEEE) received the MS degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2003, and the PhD degree from Nanjing University, Nanjing, China, in 2006. He is currently a full professor with the School of Computer Science and Engineering, University of Electronic Science and Technology of China. He has authored or coauthored more than 200 papers in referred journals and conferences, including AAAI, IEEE PerCom, IEEE GlobeCom, IEEE MASS, IEEE ICPADS, IEEE MSN, IEEE ICDCS, ACM MobiHoc, IEEE INFOCOM, ICPP, IEEE ICC, IEEE GLOBECOM, ACM TKDD, Journal of Parallel and Distributed Computing, and Computer Communications. His research interests include mobile computing, wireless sensor networks, distributed computing, deep learning, and Big Data analysis.