Resource Provision for Cloud-Enabled Automotive Vehicles With a Hierarchical Model

Kaixiang Zhang, Zhaojian Li, *Senior Member, IEEE*, Xiang Yin, *Member, IEEE*, and Liang Han, *Member, IEEE*

***Abstract*—Cloud computing is an emerging paradigm to enable computation and data-intensive automotive systems for improved safety and drivability. In this article, we propose a hierarchical, decentralized, and auction-based resource allocation model for cloud-enabled automotive vehicles. In this model, cloud-enabled vehicles bid for resources at a high level, inducing a multi-player game; at a low level, each vehicle performs an onboard resource optimization to allocate its obtained resources to its cloud-based applications. The Nash equilibrium of the induced game is defined, and we show the existence and uniqueness of the equilibrium. A constrained optimization problem is solved for onboard resource allocation. A distributed update mechanism is considered: asynchronized update where only a subset of vehicles updates their bid at each iteration. This mechanism shares desired features of requiring little communication and being secure. Convergence to Nash equilibrium is proved for the proposed update mechanism. Furthermore, the robustness to stochastic task arrival rate is characterized in terms of total variance distance. Numerical simulations are presented to demonstrate the efficacy of the proposed framework.**

***Index Terms*—Cloud computing, game theory, Nash equilibrium, resource allocation.**

I. Introduction

THERE is a growing interest in employing cloud computing for automotive applications [1], [2]. Ready access to distributed information and computing resources can enable computation and data intensive vehicular applications for improved safety, drivability, and entertainment. Several cloud-based automotive applications have been identified. For instance, a cloud-based driving speed optimizer is studied in [3] to improve fuel economy for everyday driving. In [4], a cloud-aided safety-based route planner is prototyped to boost driving safety by considering both travel time and road risk level. A cloud-based semi-active suspension control is studied in [5] to enhance suspension performance by utilizing road preview and great computation power from the cloud.

As such, cloud computing has been both an immense opportunity and a crucial challenge for the automotive industry: opportunity because of the great potential to improve driving safety, comfort, and entertainment; challenge because cybersecurity and resource allocation are critical issues that need to be carefully considered. A cloud resource allocation scheme concerns how a cloud server or multiple cloud servers, such as Amazon's EC2 and Google Cloud distributes resources to its many clients (vehicles in our context) efficiently, effectively, and profitably. These (conflicting) objectives make a good resource allocation design a nontrivial, yet important task.

Not surprisingly, resource allocation problems have been extensively studied in the general utility computing and cloud computing frameworks. The existing approaches can be categorized as centralized and decentralized. In general, centralized approaches [6], [7], [8] assume that there is a central resource allocator that can monitor all clients' information and thus can allocate the resource via a centralized manner. Although centralization can ensure high system performance, it has inherent single-point-of-failure drawback. To avoid this issue, different decentralized approaches [9], [10], [11] have been proposed. In particular, the bidding-based decentralized techniques are widely used to formulate various resource allocation problems [12], [13], [14], [15], [16], [17] from an economic-theoretical perspective, where each client makes a bid for the underlying resource. Aside from avoiding the single-point-of-failure drawback inherent in centralized resource allocation schemes, these decentralized schemes also have the important practical benefit that each client is not incentivized to lie to the resource allocator in terms of the resource needs, since the allocation is purely provisioned based on how much one pays. Several studies have also focused on the implementation perspective of this class of decentralized bidding schemes [18], [19], [20] and confirmed the advantages.

Cloud resource allocation for vehicle applications has also recently emerged. For example, several works [21], [22], [23], [24] use Markov decision process to formulate the resource allocation for vehicular cloud computing. In [22] and [23], semi-Markov decision process (SMDP) is applied to design the resource allocation to maximize the long-term expected total reward and the expected average reward of the cloud computing system, respectively. An extended SMDP model is proposed in [24] for resource

Manuscript received 28 March 2022; accepted 9 August 2022. Date of publication 2 September 2022; date of current version 16 February 2023. This work was supported by the National Science Foundation under Award 2045436. This article was recommended by Associate Editor X. Ge. *(Corresponding author: Zhaojian Li.)*

Kaixiang Zhang and Zhaojian Li are with the Department of Mechanical Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: zhangk64@msu.edu; lizhaoj1@egr.msu.edu).

Xiang Yin is with the Department of Automation and the Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: yinxiang@sjtu.edu.cn).

Liang Han is with the School of Sino-French Engineering, Beihang University, Beijing 100191, China (e-mail: liang_han@buaa.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSMC.2022.3200452.

Digital Object Identifier 10.1109/TSMC.2022.3200452

2168-2216 © 2022 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

allocation which additionally considers heterogeneous vehicles and roadside units. Reinforcement learning (RL) [25], [26], [27] is another effective tool to address the resource allocation for vehicle applications. An RL approach is studied in [26] that trains a bidding policy by modeling all other vehicles as an unknown environment. To support different vehicular applications, the work in [27] considers two computing architectures and employs RL techniques to address the derived multidimensional resource optimization problems. Furthermore, a multiobjective optimization is formulated in [28] to optimally allocate the resource from both the provider's and user's perspectives. Note that most of the aforementioned approaches require the cloud to collect and monitor all related information, such as onboard tasks and utility functions, and allocate the resource with centralized manner, which not only shows low robustness to single-point-of-failure but also raises concerns in vehicle privacy and security since information leaking in connected vehicles can lead to various malicious activities [29], [30].

In this article, we propose a novel hierarchical model to achieve robust and secure resource allocation for cloud-aided automotive systems. Specifically, at a high level, a decentralized, auction-based allocation scheme is considered where each vehicle bids for the cloud resource under a proportional-sharing mechanism. The obtained resource is then optimally allocated for onboard applications that incorporate the queuing nature of the applications. We formulate the interplay between the vehicles as a multiplayer game and examine the system response from the perspective of Nash equilibrium. Results on the existence and uniqueness of the Nash equilibrium are then established. An asynchronized bid update mechanism is proposed and its convergence to the unique Nash equilibrium is proved. The proposed auction-based bidding mechanism is secure and requires little communication efforts, making it suitable for vehicle applications that are privacy-/security-sensitive.

The contributions of this article include the following. First, we propose a novel and practical resource allocation model for cloud-aided automotive systems that combines a high-level auction-based resource bidding with a low-level onboard optimization. Second, we characterize the auction-based interplay as a multiplayer game and establish the existence and uniqueness of the Nash equilibrium. Furthermore, a decentralized scheme that is secure and requires little communication is proposed. Results on the convergence to the unique Nash equilibrium are derived. Last but not least, numerical simulations are presented to demonstrate the efficacy of the developed framework. In particular, we characterize the scheme's robustness to stochastic environment using total variance distance (TVD) and statistically show their stochastic stability under random task arrival rate. There are important differences between our work and the existing studies. The works [22], [23], [24], [28] consider the resource allocation in vehicular cloud computing, which require the cloud to collect privacy-sensitive information, such as onboard tasks and utility functions. While in this work, we focus on optimally allocating the multiserver-based cloud resources for vehicle applications, and the proposed hierarchical model and bidding mechanism are light-weight in communication efforts and can avoid privacy leakage of the vehicle. Moreover, due to the special problem structure of resource allocation for the cloud-enabled vehicles, our formulation has the crucial difference from the existing literature on bidding-based decentralized schemes [12], [13], [14], [15], [16], [17]. The existing literature focused exclusively on a single server resource allocation, whereas we study network-based resource allocation due to the nature of vehicle application. Further, our formulation is different in the aspect that each vehicle/participant also needs to perform another level of optimization (to allocate resources efficiently for its own set of tasks internally), which interacts with and hence needs to be taken into account when deciding the amount to bid. These differentiating aspects require us to pose a different resource allocation model from the existing work and hence require us to understand its properties differently.

The remainder of this article is organized as follows. In Sections II and III, the resource allocation problem for cloud-aided automotive systems is described and a hierarchical allocation model is developed, respectively. In Section IV, results on the existence and uniqueness of Nash equilibrium of the induced game are established. Dynamics of reaching the equilibrium is presented in Section V, where asynchronized update scheme is proposed with proved convergence to the unique Nash equilibrium. Numerical simulations are presented in Section VI whereas Section VII concludes this article.

II. Problem Formulation

This article considers a hierarchical and decentralized framework of divisible resource allocation for cloud-enabled vehicles, where the objective is to optimally allocate the distinct server resource to subscribed vehicles with multiple onboard applications by minimizing a combined backlog and cloud usage-incurred cost (see Section III-C for details on the cost term). A schematic of cloud resource allocation for vehicle-to-cloud-to-vehicle (V2C2V) [2] applications is illustrated in Fig. 1. Based on Fig. 1, a simple yet flexible formulation of resource allocation is provided in this section.

A. Task Flow

The cloud consists of a network of M servers, and each server $\mathcal{S}_k$, $k = 1, \ldots, M$, provides a distinct computational resource that is needed for completing different jobs generated from vehicles. The total amount of resource at sever $\mathcal{S}_k$ is denoted by γ^k and we assume it can be arbitrarily allocated to different cloud tasks. The resource γ^k can be viewed as the service rate: when the available resource is γ^k, the amount of time it takes to serve a job follows an exponential distribution with parameter γ^k [and hence mean processing time is $(1/\gamma^k)$] [15]. Furthermore, we consider a fleet of N cloud-enabled vehicles, and each vehicle $\mathcal{V}_i$, $i = 1, \ldots, N$, runs a (possibly different) set of applications/tasks, $T_{i,1}, T_{i,2}, \ldots, T_{i,m_i}$, where the subscripts i and m_i, respectively, denote the vehicle index and the total number of cloud-based applications running on $\mathcal{V}_i$. The jth task in vehicle i, $T_{i,j}$, $i = 1, \ldots, N$, $j = 1, \ldots, m_i$, has a stream of jobs that need to be processed using the resources in

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

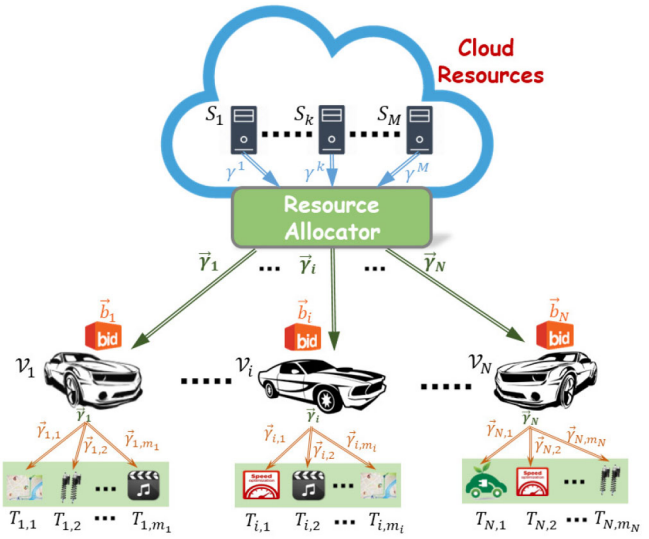


Fig. 1. Schematic of cloud resource allocation for V2C2V applications.

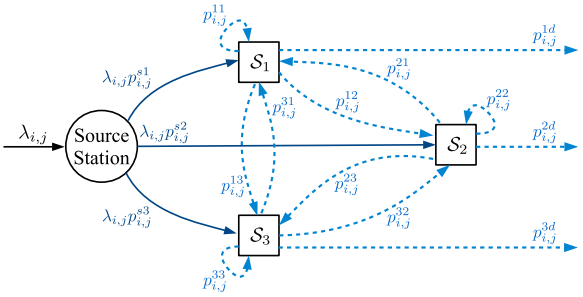


Fig. 2. Queuing network for task $T_{i,j}$ with the total server amount M being 3.

the network of servers. For instance, a task could be a virtual intelligent assistant application, where the application needs to constantly take the human voice as input and run various speech recognition, classification and speech synthesis jobs to assist the human driver. As such, we can view a task as a stream of incoming jobs, each of which needs to be processed by (possibly different) servers in the cloud and flows through the network of servers via a (possibly random) path.

The entry point for all tasks' streams of jobs is a source station s (distinct from the M servers). Similar to [22], [23], and [24], it is assumed that jobs from task $T_{i,j}$ arrive at the source station s via a Poisson process with arrival rate $\lambda_{i,j}$. The source station performs preliminary processing and then routes $T_{i,j}$ to server $\mathcal{S}_k$ with probability $p_{i,j}^{sk} \in [0, 1]$. Without loss of generality, we assume that this preprocessing time and the task transmission time are negligible. After server $\mathcal{S}_k$ finishes processing task $T_{i,j}$, $T_{i,j}$ will be routed to server $\mathcal{S}_{k'}$ with probability $p_{i,j}^{kk'} \in [0, 1]$, for all $k, k' \in \{1, \ldots, M\}$ and will also exist the network (representing the event that the task is completely finished) with probability $p_{i,j}^{kd} \in [0, 1]$. Note that loops are allowed, i.e., k may be equal to k'. Furthermore, these probabilities satisfy the following flow-conservation constraints:

$$\sum_{k'=1}^{M} p_{i,j}^{kk'} + p_{i,j}^{kd} = 1 \quad \forall k \in \{1, \ldots, M\}. \tag{1}$$

The queuing network for task $T_{i,j}$ among three servers is presented in Fig. 2 for reference.

Based on the above description, it can be found that the considered cloud-enabled vehicles are heterogeneous with different onboard applications, and the characteristics of the onboard applications are reflected by the Poisson arrival rate model and routing probabilities. We now explain the reason for choosing such characteristics to formulate the queuing model. Although automotive applications are in general scheduled with fixed periods, due to variations in execution time and higher-priority task interruptions, task arrivals are stochastic with Poisson process being an appropriate modeling [22], [23], [24]. Due to computation variation and random communication delays, the processing time is also stochastic in nature and exponential processing rate is a sensible assumption. Different from the conventional M/M/N queue in which multiple servers have the same computational resource and are used to complete the same tasks, we consider that each task will be processed by multiple servers with different computational resource and flows through the network of servers via routing probabilities. The routing probabilities offer considerable flexibility in modeling. For instance, a common scenario in the cloud-enabled vehicle application is that a task needs to be processed by following a predetermined path of servers. This scenario corresponds to setting all-but-one routing probabilities to be 0, and the remaining one routing probability to be 1, which is a situation known as "routepinning" in queuing theory [31]. In addition, the routing probabilities can also model the situation where the state of a task is random after each service stage. Specifically, when a task is processed in server $\mathcal{S}_k$, whether it needs to be further processed and by which server depends on a certain random event in the current service stage. For example, if an error has occurred, then the task needs to be routed back to the current server for reprocessing. If there are dependencies between two service stages, then a current failure may require the task to be routed to an earlier server on which the processing already took place. In these scenarios, at each stage, there is inherent uncertainty regarding the next service step for a task, and such uncertainty can be precisely captured by the routing probabilities.

B. Single Task's Perspective and Backlog

The resource provided by server $\mathcal{S}_k$ to process task $T_{i,j}$ is denoted by $\gamma_{i,j}^k$, and the service procedure follows the first-come-first-serve protocol. We note that the resulting queuing network for each task $T_{i,j}$ is an open Jackson network [32]. Specifically, for each task, the following list of conditions holds.

1) Task $T_{i,j}$'s jobs arrive at server $\mathcal{S}_k$ according a Poisson process with rate $\lambda_{i,j} p_{i,j}^{sk}$.
2) All service times are independently and exponentially distributed: the service time at server $\mathcal{S}_k$ is $\exp(\gamma_{i,j}^k)$.
3) A task, upon completion of service at server $\mathcal{S}_k$, enters server $\mathcal{S}_{k'}$ with probability $p_{i,j}^{kk'}$ and exits the system with probability $p_{i,j}^{kd}$.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

4) The exit probabilities $p_{i,j}^{kd}$ are nonzero for a nonempty subset of $\{1, \ldots, M\}$.

With the above notation, we can compute the effective arrival rates to every server, including both external arrivals and internal transitions. Specifically, let $\alpha_{i,j}^k$ denote task $T_{i,j}$'s effective arrival rate to server $\mathcal{S}_k$, we have for each $k \in \{1, \ldots, M\}$

$$\alpha_{i,j}^k = \lambda_{i,j} p_{i,j}^{sk} + \sum_{k'=1}^{M} \alpha_{i,j}^{k'} p_{i,j}^{k'k}.$$

We can write this compactly using matrix notation as follows:

$$\vec{\boldsymbol{\alpha}}_{i,j} = \left(\boldsymbol{I} - \boldsymbol{P}_{i,j}\right)^{-1} \vec{\boldsymbol{\lambda}}_{i,j} \tag{2}$$

where $\vec{\boldsymbol{\alpha}}_{i,j} = \left[\alpha_{i,j}^1, \alpha_{i,j}^2, \ldots, \alpha_{i,j}^M\right]^T$, $\boldsymbol{I}$ is an identity matrix, $\boldsymbol{P}_{i,j}$ is a matrix with its (k, k') element being $p_{i,j}^{k'k}$, and $\vec{\boldsymbol{\lambda}}_{i,j} = \left[\lambda_{i,j} p_{i,j}^{s1}, \lambda_{i,j} p_{i,j}^{s2}, \ldots, \lambda_{i,j} p_{i,j}^{sM}\right]^T$. Note that $(\boldsymbol{I} - \boldsymbol{P}_{i,j})^{-1}$ always exists since all eigenvalues of $\boldsymbol{P}_{i,j}$ are strictly less than 1.

By using Jackson's Theorem [32], it can be concluded that the mean/expected backlog of task $T_{i,j}$ at server $\mathcal{S}_k$ is given by

$$B_{i,j}^k := \begin{cases} \frac{\alpha_{i,j}^k}{\gamma_{i,j}^k - \alpha_{i,j}^k}, & \text{if} \gamma_{i,j}^k > \alpha_{i,j}^k \\ +\infty, & \text{otherwise.} \end{cases} \tag{3}$$

From (3), it follows that the total backlog cost of vehicle $\mathcal{V}_i$ at server $\mathcal{S}_k$ is defined as

$$B_i^k := \begin{cases} \sum_{j=1}^{m_i} w_{i,j}^k \frac{\alpha_{i,j}^k}{\gamma_{i,j}^k - \alpha_{i,j}^k}, & \text{if } \gamma_{i,j}^k > \alpha_{i,j}^k \quad \forall j = 1, \ldots, m_i \\ +\infty, & \text{otherwise} \end{cases} \tag{4}$$

where $w_{i,j}^k > 0$ is the weighting factor that describes the relative importance of different task of vehicle $\mathcal{V}_i$ at server $\mathcal{S}_k$. Higher priority task corresponds to larger weighting factor.

C. Main Question

The goal of each (selfish) vehicle is to obtain more resource from the cloud so that the total backlog of all running applications/tasks, i.e., $B_i = \sum_{k=1}^{M} B_i^k$, is kept as small as possible. This immediately raises the central question faced by the administrator of the network: how should the resource γ^k at each server $\mathcal{S}_k$ be allocated to the vehicles with multiple applications?

To address this problem, we propose a hierarchical, decentralized, and auction-based resource allocation scheme. As shown in Fig. 3, at a high level, each vehicle $\mathcal{V}_i$ bids for resource at server $\mathcal{S}_k$ by submitting a price b_i^k. How much resource each vehicle gets from each server depends on how much bid it pays as well as on how much other vehicles pay. That is, each server $\mathcal{S}_k$ severally allocates its resource γ^k based on the received bid $\{b_1^k, \ldots, b_N^k\}$. Let γ_i^k be the resource that vehicle $\mathcal{V}_i$ gets from server $\mathcal{S}_k$ during the bidding process. Then, at a low level, vehicle $\mathcal{V}_i$ will further optimally distributes each kind of server resource γ_i^k into m_i portions $\{\gamma_{i,1}^k, \ldots, \gamma_{i,m_i}^k\}$ with $\gamma_{i,j}^k$ being the resource provided by server $\mathcal{S}_k$ to process task $T_{i,j}$. Since there are M different servers in the cloud, each vehicle needs to submit a bidding vector $\vec{\boldsymbol{b}}_i = \left[b_i^1, \ldots, b_i^M\right]^T$ to the cloud for resource allocation and will receive a portion of the overall server resource $\vec{\boldsymbol{\gamma}}_i = \left[\gamma_i^1, \ldots, \gamma_i^M\right]^T$.

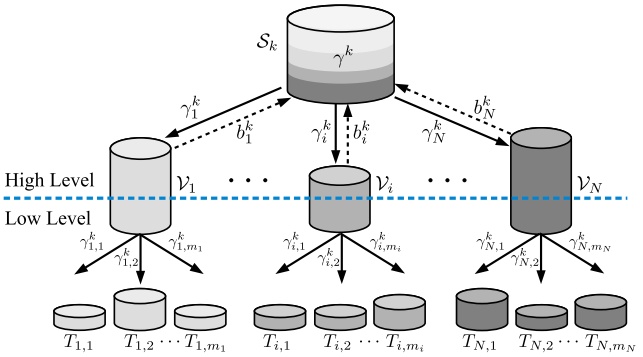


Fig. 3. Hierarchical model for resource allocation at server $\mathcal{S}_k$.

We describe the resource allocation scheme in detail in the next section. Before proceeding, note that if $\sum_{i=1}^{N} \sum_{j=1}^{m_i} \alpha_{i,j}^k > \gamma^k$, then at least one task will have an infinite job queue at server $\mathcal{S}_k$, no matter what the resource allocation scheme is. In this article, we assume that $\sum_{i=1}^{N} \sum_{j=1}^{m_i} \alpha_{i,j}^k < \gamma^k \ \forall k = 1, \ldots, M$, meaning each server offers enough resources so that there always exists a partition of the resources to keep all the backlogs finite.

III. Hierarchical Resource Allocation

A. Bidding Mechanism

If the cloud is trusted by all vehicles and can thus elicit job-related information such as $\alpha_{i,j}^k$ and the average backlog function from all vehicles, then the cloud can in principal compute the optimal allocation of server resources. However, there are several issues with this centralized scheme.

First, the cloud needs to decide metrics for the "social welfare" by aggregating the backlogs from all vehicles. However, no vehicles would like to downplay its own importance. For example, if a linear weighted sum of individual backlogs is adopted as the overall "social" cost, the cloud will unlikely get unanimous weights among the vehicles. Second, the cloud should not expect the vehicles to provide truthful information in the first place, since doing so would put the vehicle in a disadvantageous position in terms of getting resources. Therefore, the vehicles have incentives to lie.

To address these issues, we consider a proportional-sharing bidding scheme to allocate the multiserver resources: each vehicle $\mathcal{V}_i$ makes a bidding vector $\vec{\boldsymbol{b}}_i = \left[b_i^1, \ldots, b_i^M\right]^T$, and given $b_i^k \geq 0$ the resource that vehicle gets from server $\mathcal{S}_k$ is

$$\gamma_i^k = \begin{cases} \gamma^k \times \frac{b_i^k}{\sum_{l=1}^{N} b_l^k}, & \text{if } \sum_{l=1}^{N} b_l^k > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

While there are many other bidding schemes, we adopt linear-proportional sharing, a commonly used bidding scheme whose effectiveness has been demonstrated in a variety of different applications [12], [33], [34], [35], [36]. In our case, each vehicle only submits the bidding vector without any other auxiliary

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

information to the cloud, and thus allocating the resource with the bidding proportion is a suitable way to ensure fairness. Furthermore, it takes $O(N)$ operations to calculate the linear proportional sharing, which can be easily implemented in the cloud.

B. Vehicle-Level Resource Optimization

Essentially, the vehicle needs to make two decisions: 1) the bidding price $\vec{\boldsymbol{b}}_i$ and 2) allocation of the obtained resource γ_i^k to its onboard applications. After submitting a bidding vector $\vec{\boldsymbol{b}}_i$, vehicle $\mathcal{V}_i$ will receive a portion of the overall sever resource, $\vec{\boldsymbol{\gamma}}_i = \left[\gamma_i^1, \ldots, \gamma_i^M\right]^T$, based on the proportion-sharing scheme in (5). Then the vehicle will distribute the obtained resources γ_i^k among its applications $T_{i,1}, \ldots, T_{i,m_i}$. Let $\vec{\boldsymbol{\gamma}}_i^k = \left[\gamma_{i,1}^k, \ldots, \gamma_{i,m_i}^k\right]^T$ be the partitioning vector of resource γ_i^k, then the optimal vehicle-level resource partitioning for vehicle $\mathcal{V}_i$ can be formulated as the following constrained optimization problem:

$$\begin{aligned} \underset{\vec{\boldsymbol{\gamma}}_i^1, \ldots, \vec{\boldsymbol{\gamma}}_i^M}{\text{minimize}} \quad & B_i \coloneqq \sum_{k=1}^{M} B_i^k \\ \text{subject to} \quad & \sum_{j=1}^{m_i} \gamma_{i,j}^k \le \gamma_i^k \quad \forall k = 1, \ldots, M \\ & \gamma_{i,j}^k \ge \alpha_{i,j}^k \quad \forall j = 1, \ldots, m_i, k = 1, \ldots, M \end{aligned} \tag{6}$$

where the cost function is the summation of all backlogs; the first constraint characterizes the resource limit at each server and the second constraint characterizes the minimum required resource for each application to avoid infinite backlog as shown in (3). Note that $\gamma_i^k \ge \sum_{j=1}^{m_i} \alpha_{i,j}^k \ \forall k = 1, \ldots, M$, is needed to make (6) feasible. This is a reasonable assumption as the servers are viewed as a pool of enormous resources. We can always limit the initial bidding in a reasonable range to satisfy $\gamma_i^k \ge \sum_{j=1}^{m_i} \alpha_{i,j}^k \ \forall k = 1, \ldots, M$. The solution of this optimization problem is summarized in the following lemma.

Lemma 1: Let $\alpha_i^k = \sum_{j=1}^{m_i} \alpha_{i,j}^k$ be the minimal required resource for vehicle $\mathcal{V}_i$ at server $\mathcal{S}_k$. If $\gamma_i^k > \alpha_i^k \ \forall k = 1, \ldots, M$, then the optimal onboard resource partition in (6) is $\gamma_{i,j}^{k*} = ((\gamma_i^k - \alpha_i^k)/(\sum_{j=1}^{m_i} \sqrt{w_{i,j}^k \alpha_{i,j}^k}))\sqrt{w_{i,j}^k \alpha_{i,j}^k} + \alpha_{i,j}^k$ and the associated minimal cost is $B_i^* = \sum_{k=1}^{M} B_i^{k*} = \sum_{k=1}^{M} (\sum_{j=1}^{m_i} \sqrt{w_{i,j}^k \alpha_{i,j}^k})^2/(\gamma_i^k - \alpha_i^k)$.

Proof: From (6), we define the Lagrangian

$$\mathcal{L}\left(\vec{\boldsymbol{\gamma}}_i^1, \ldots, \vec{\boldsymbol{\gamma}}_i^M, \vec{\boldsymbol{\mu}}\right) = \sum_{k=1}^{M} \left(B_i^k + \mu_0^k \left(\sum_{j=1}^{m_i} \gamma_{i,j}^k - \gamma_i^k \right) + \sum_{j=1}^{m_i} \mu_j^k \left(\alpha_{i,j}^k - \gamma_{i,j}^k \right) \right)$$

where $\mu_j^k \ge 0 \ \forall j = 0, \ldots, m_i, k = 1, \ldots, M$. The Karush–Kuhn–Tucker conditions for the optimality in (6) are

$$\frac{\partial \mathcal{L}}{\partial \gamma_{i,j}^k} = \frac{-w_{i,j}^k \alpha_{i,j}^k}{\left(\gamma_{i,j}^{k*} - \alpha_{i,j}^k\right)^2} + \mu_0^k - \mu_j^k = 0$$

$$\forall j = 1, \ldots, m_i, k = 1, \ldots, M \tag{7}$$

$$\sum_{j=1}^{m_i} \gamma_{i,j}^{k*} \le \gamma_i^k \ \forall k = 1, \ldots, M \tag{8}$$

$$\gamma_{i,j}^{k*} \ge \alpha_{i,j}^k \ \forall j = 1, \ldots, m_i, k = 1, \ldots, M \tag{9}$$

$$\mu_0^k \left(\sum_{j=1}^{m_i} \gamma_{i,j}^{k*} - \gamma_i^k \right) = 0 \ \forall k = 1, \ldots, M \tag{10}$$

$$\mu_j^k \left(\alpha_{i,j}^k - \gamma_{i,j}^{k*} \right) = 0 \ \forall j = 1, \ldots, m_i, \ k = 1, \ldots, M \tag{11}$$

$$\mu_0^k \ge 0, \quad \mu_j^k \ge 0 \ \forall j = 1, \ldots, m_i, \ k = 1, \ldots, M. \tag{12}$$

Note from (3), for each task $T_{i,j}$, the obtained resource $\gamma_{i,j}^k$ from server $\mathcal{S}_k$ needs to be *strictly* greater than $\alpha_{i,j}^k$ to keep the backlog cost finite. Therefore, $\mu_j^k = 0 \ \forall j = 1, \ldots, m_i, k = 1, \ldots, M$. From (7), it follows that $\mu_0^k \ne 0$, meaning

$$\sum_{j=1}^{m_i} \gamma_{i,j}^{k*} = \gamma_i^k \tag{13}$$

and $(w_{i,1}^k \alpha_{i,1}^k)/(\gamma_{i,1}^{k*} - \alpha_{i,1}^k)^2 = (w_{i,2}^k \alpha_{i,2}^k)/(\gamma_{i,2}^{k*} - \alpha_{i,2}^k)^2 = \cdots = (w_{i,m_i}^k \alpha_{i,m_i}^k)/(\gamma_{i,m_i}^{k*} - \alpha_{i,m_i}^k)^2$, which is equivalent to

$$\frac{\gamma_{i,1}^{k*} - \alpha_{i,1}^k}{\sqrt{w_{i,1}^k \alpha_{i,1}^k}} = \frac{\gamma_{i,2}^{k*} - \alpha_{i,2}^k}{\sqrt{w_{i,2}^k \alpha_{i,2}^k}} = \cdots = \frac{\gamma_{i,m_i}^{k*} - \alpha_{i,m_i}^k}{\sqrt{w_{i,m_i}^k \alpha_{i,m_i}^k}} = \sigma \tag{14}$$

with σ being a positive constant. Also note that $\sum_{j=1}^{m_i} (\gamma_{i,j}^{k*} - \alpha_{i,j}^k) = \gamma_i^k - \sum_{j=1}^{m_i} \alpha_{i,j}^k = \sigma \sum_{j=1}^{m_i} \sqrt{w_{i,j}^k \alpha_{i,j}^k}$, which indicates

$$\sigma = \frac{\gamma_i^k - \sum_{j=1}^{m_i} \alpha_{i,j}^k}{\sum_{j=1}^{m_i} \sqrt{w_{i,j}^k \alpha_{i,j}^k}}. \tag{15}$$

Combining (14) and (15), we have

$$\gamma_{i,j}^{k*} = \frac{\gamma_i^k - \sum_{j=1}^{m_i} \alpha_{i,j}^k}{\sum_{j=1}^{m_i} \sqrt{w_{i,j}^k \alpha_{i,j}^k}} \sqrt{w_{i,j}^k \alpha_{i,j}^k} + \alpha_{i,j}^k. \tag{16}$$

Plugging (16) into (4), we have $B_i^{k*} = (\sum_{j=1}^{m_i} \sqrt{w_{i,j}^k \alpha_{i,j}^k})^2/(\gamma_i^k - \sum_{j=1}^{m_i} \alpha_{i,j}^k)$, indicating that $B_i^* = \sum_{k=1}^{M} B_i^{k*} = \sum_{k=1}^{M} (\sum_{j=1}^{m_i} \sqrt{w_{i,j}^k \alpha_{i,j}^k})^2/(\gamma_i^k - \sum_{j=1}^{m_i} \alpha_{i,j}^k)$. ■

C. Induced Game

We consider that the total cost of vehicle $\mathcal{V}_i$ consists of two parts: 1) the overall backlogs among all applications and 2) the bidding price related to $\vec{\boldsymbol{b}}_i$. Each vehicle can weigh these two costs differently and we use a weighted sum to represent the total cost function

$$C_i\left(\vec{\boldsymbol{b}}\right) = B_i^*\left(\vec{\boldsymbol{b}}\right) + \vec{\boldsymbol{\varepsilon}}_i^T \vec{\boldsymbol{b}}_i \tag{17}$$

where $\vec{\boldsymbol{b}} = (\vec{\boldsymbol{b}}_1, \ldots, \vec{\boldsymbol{b}}_N)$ denotes the bidding vector of the all N vehicles and $\vec{\boldsymbol{\varepsilon}}_i = \left[\varepsilon_i^1, \ldots, \varepsilon_i^M\right]^T$ $(\varepsilon_i^k > 0)$ is the weighting factor that characterizes the relative importance placed on the

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

bidding cost compared to waiting time from $\mathcal{V}_i$. Based on the fact that $B_i^* = \sum_{k=1}^{M} B_i^{k*}$, (17) can be rewritten as

$$C_i\left(\vec{\boldsymbol{b}}\right) = \sum_{k=1}^{M}\left(B_i^{k*} + \varepsilon_i^k b_i^k\right) = \sum_{k=1}^{M} C_i^k\left(\vec{\boldsymbol{b}}^k\right) \tag{18}$$

where $\vec{\boldsymbol{b}}^k = \left[b_1^k, \ldots, b_N^k\right]^T$ is the bidding vector of the N vehicles submitted to server $\mathcal{S}_k$, and $C_i^k(\vec{\boldsymbol{b}}^k) = B_i^{k*} + \varepsilon_i^k b_i^k$ is the subcost function with respect to server $\mathcal{S}_k$. Note that the M subcost functions $C_i^k(\vec{\boldsymbol{b}}^k)$ are independent to each other, indicating that minimizing each $C_i^k(\vec{\boldsymbol{b}}^k)$ is equivalent to minimizing the total cost function $C_i(\vec{\boldsymbol{b}})$. Since the subcost function $C_i^k(\vec{\boldsymbol{b}}^k)$ in (18) is interdependent on the bid $\vec{\boldsymbol{b}}^k$, a game is naturally induced with the Nash equilibrium defined as follows.

Definition 1: A biding vector $\vec{\boldsymbol{b}}^{k*}$ is a Nash equilibrium, if $C_i^k(b_i^{k*}, \vec{\boldsymbol{b}}_{-i}^{k*}) \le C_i^k(b_i^k, \vec{\boldsymbol{b}}_{-i}^{k*})\ \forall b_i^k \in \mathbb{R}_+\ \forall i = 1, \ldots, N$, where $\vec{\boldsymbol{b}}_{-i}^{k*}$ is an $(N-1)$-dimensional vector by removing the ith element from $\vec{\boldsymbol{b}}^{k*}$.

Immediate questions pertaining to the existence and uniqueness of Nash equilibrium arise: the asymptote at α_i^k due to the queuing nature of the system induces the cost values to be unbounded (and hence a noncompact domain), rendering the standard game theory results not applicable here. A second concern also arises here: how can an equilibrium be reached? We will investigate these two issues in the next two sections in order.

D. Best Response Dynamics

The best response function is not only a useful tool to characterize the existence and uniqueness of Nash equilibrium but also offers rationales for the dynamics to reach an equilibrium.

Definition 2: Given a bidding vector $\vec{\boldsymbol{b}}^k \in \mathbb{R}_+^N$, the best response function g_i^k for vehicle $\mathcal{V}_i$ is defined as

$$g_i^k\left(\vec{\boldsymbol{b}}^k\right) = \arg\min_{b_i^k \in \mathbb{R}_+} C_i^k\left(b_i^k, \vec{\boldsymbol{b}}_{-i}^k\right). \tag{19}$$

With the best-response function definition, the Nash equilibrium $\vec{\boldsymbol{b}}^{k*}$ can be stated as $\vec{\boldsymbol{g}}^k(\vec{\boldsymbol{b}}^{k*}) = \vec{\boldsymbol{b}}^{k*}$, that is, the Nash equilibrium is a fixed point of $\vec{\boldsymbol{g}}^k(\cdot)$. Here, $\vec{\boldsymbol{g}}^k(\vec{\boldsymbol{b}}^{k*}) = \left[g_1^k(\vec{\boldsymbol{b}}^{k*}), g_2^k(\vec{\boldsymbol{b}}^{k*}), \ldots, g_N^k(\vec{\boldsymbol{b}}^{k*})\right]^T$.

IV. Existence and Uniqueness of Nash Equilibrium

In this section, we answer the question on the existence and uniqueness of the induced game. Toward that end, we first show several properties of the best response function $\vec{\boldsymbol{g}}^k$.

Lemma 2: The best response function $\vec{\boldsymbol{g}}^k$ defined in Definition 2 has the following properties.

1) The best response function is well-defined and continuous.
2) For any $\vec{\boldsymbol{x}}, \vec{\boldsymbol{y}} \in \mathbb{R}_+^N$, $\vec{\boldsymbol{x}} \ge \vec{\boldsymbol{y}} \Rightarrow \vec{\boldsymbol{g}}^k(\vec{\boldsymbol{x}}) \ge \vec{\boldsymbol{g}}^k(\vec{\boldsymbol{y}})$.
3) For any $\beta > 1$, $\vec{\boldsymbol{g}}^k(\beta\vec{\boldsymbol{x}}) < \beta\vec{\boldsymbol{g}}^k(\vec{\boldsymbol{x}})\ \forall \vec{\boldsymbol{x}} \in \mathbb{R}_+^N$.
4) There exist $\vec{\boldsymbol{x}}, \vec{\boldsymbol{y}} \in \mathbb{R}_+^N$ with $\vec{\boldsymbol{x}} < \vec{\boldsymbol{y}}$ such that $\vec{\boldsymbol{x}} \le \vec{\boldsymbol{g}}^k(\vec{\boldsymbol{x}})$, $\vec{\boldsymbol{y}} \ge \vec{\boldsymbol{g}}^k(\vec{\boldsymbol{y}})$.

Here, $\vec{\boldsymbol{x}} \ge \vec{\boldsymbol{y}}$ represents the vector inequality, that is, $x_i \ge y_i$ for all $i = 1, \ldots, N$, and $\vec{\boldsymbol{x}} \ne \vec{\boldsymbol{y}}$.

Proof: From (18), it follows that

$$\begin{aligned}\frac{\partial C_i^k}{\partial b_i^k} &= \frac{\partial B_i^{k*}}{\partial \gamma_i^k}\frac{\partial \gamma_i^k}{\partial b_i^k} + \varepsilon_i^k \\ &= \frac{-\left(\sum_{j=1}^{m_i}\sqrt{w_{i,j}^k \alpha_{i,j}^k}\right)^2}{\left(\gamma_i^k - \alpha_i^k\right)^2} \cdot \frac{\gamma^k \sum \vec{\boldsymbol{b}}_{-i}^k}{\left(b_i^k + \sum \vec{\boldsymbol{b}}_{-i}^k\right)^2} + \varepsilon_i^k\end{aligned} \tag{20}$$

where $\sum \vec{\boldsymbol{b}}_{-i}^k = \sum_{l=1}^{N} b_l^k - b_i^k$. The second-order derivative is thus

$$\begin{aligned}\frac{\partial^2 C_i^k}{\partial^2 b_i^k} &= 2\frac{\left(\sum_{j=1}^{m_i}\sqrt{w_{i,j}^k \alpha_{i,j}^k}\right)^2}{\left(\gamma_i^k - \alpha_i^k\right)^3} \cdot \frac{\left(\gamma^k \sum \vec{\boldsymbol{b}}_{-i}^k\right)^2}{\left(b_i^k + \sum \vec{\boldsymbol{b}}_{-i}^k\right)^4} \\ &\quad + 2\frac{\left(\sum_{j=1}^{m_i}\sqrt{w_{i,j}^k \alpha_{i,j}^k}\right)^2}{\left(\gamma_i^k - \alpha_i^k\right)^2} \cdot \frac{\gamma^k \sum \vec{\boldsymbol{b}}_{-i}^k}{\left(b_i^k + \sum \vec{\boldsymbol{b}}_{-i}^k\right)^3}.\end{aligned} \tag{21}$$

Note that to keep the cost finite, the best response must have $\gamma_i^k > \alpha_i^k$. Therefore, we have $(\partial^2 C_i^k / \partial^2 b_i^k) > 0$, meaning that $C_i^k(\vec{\boldsymbol{b}}^k)$ is a convex function with respect to b_i^k. The convexity of C_i^k therefore implies that the necessary and sufficient condition for the minimizer is $(\partial C_i^k / \partial b_i^k)(b_i^{k*}) = 0$. By defining $\vec{\boldsymbol{Z}}^k(\vec{\boldsymbol{b}}^k) : \mathbb{R}_+^N \to \mathbb{R}_+^N$ with the ith output as

$$Z_i^k\left(\vec{\boldsymbol{b}}^k\right) := \frac{\left(\sum_{j=1}^{m_i}\sqrt{w_{i,j}^k \alpha_{i,j}^k}\right)^2}{\left(\gamma_i^k - \alpha_i^k\right)^2} \cdot \frac{\gamma^k \sum \vec{\boldsymbol{b}}_{-i}^k}{\left(b_i^k + \sum \vec{\boldsymbol{b}}_{-i}^k\right)^2} \tag{22}$$

the optimality $(\partial C_i^k / \partial b_i^k) = 0$ is thus equivalent to $Z_i^k(b_i^{k*}, \vec{\boldsymbol{b}}_{-i}^k) = \varepsilon_i^k$. Note that $\gamma_i^k(b_i^k, \vec{\boldsymbol{b}}_{-i}^k)$ is increasing in b_i^k so $(\sum_{j=1}^{m_i}\sqrt{w_{i,j}^k \alpha_{i,j}^k})^2 / (\gamma_i^k - \alpha_i^k)^2$ is decreasing in b_i^k. Also, $\gamma^k \sum \vec{\boldsymbol{b}}_{-i}^k / (b_i^k + \sum \vec{\boldsymbol{b}}_{-i}^k)^2$ is decreasing in b_i^k. Therefore, $Z_i^k(b_i^k, \vec{\boldsymbol{b}}_{-i}^k)$ is monotonically decreasing in b_i^k with the following two limits:

$$\lim_{b_i^k \to \left(\frac{\alpha_i^k \cdot \sum \vec{\boldsymbol{b}}_{-i}^k}{\gamma^k - \alpha_i^k}\right)^+} Z_i^k\left(b_i^k, \vec{\boldsymbol{b}}_{-i}^k\right) = +\infty \tag{23}$$

$$\lim_{b_i^k \to +\infty} Z_i^k\left(b_i^k, \vec{\boldsymbol{b}}_{-i}^k\right) = 0. \tag{24}$$

This implies that there exists a unique b_i^{k*} that satisfies $Z_i^k(b_i^{k*}, \vec{\boldsymbol{b}}_{-i}^k) = \varepsilon_i^k$. Note that $Z_i^k(b_i^k, \vec{\boldsymbol{b}}_{-i}^k)$ is smooth and continuous so $g_i^k(b_i^k, \vec{\boldsymbol{b}}_{-i}^k)$ is also continuous. 1) is thus established.

In order to prove 2), we first show that $\forall l \ne i$

$$\begin{aligned}\frac{\partial Z_i^k}{\partial b_l^k} &= -2\frac{\left(\sum_{j=1}^{m_i}\sqrt{w_{i,j}^k \alpha_{i,j}^k}\right)^2}{\left(\gamma_i^k - \alpha_i^k\right)^3} \cdot \frac{-\gamma^k b_i^k}{\left(b_i^k + \sum \vec{\boldsymbol{b}}_{-i}^k\right)^2} \cdot \frac{\gamma^k \sum \vec{\boldsymbol{b}}_{-i}^k}{\left(b_i^k + \sum \vec{\boldsymbol{b}}_{-i}^k\right)^2} \\ &\quad + \frac{\left(\sum_{j=1}^{m_i}\sqrt{w_{i,j}^k \alpha_{i,j}^k}\right)^2}{\left(\gamma_i^k - \alpha_i^k\right)^2} \cdot \frac{\gamma^k\left(b_i^k - \sum \vec{\boldsymbol{b}}_{-i}^k\right)}{\left(b_i^k + \sum \vec{\boldsymbol{b}}_{-i}^k\right)^3}\end{aligned}$$

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

$$= \frac{\gamma^k\left(\sum_{j=1}^{m_i}\sqrt{w_{i,j}^k\alpha_{i,j}^k}\right)^2}{\left(\gamma_i^k-\alpha_i^k\right)^3\left(b_i^k+\sum\vec{\boldsymbol{b}}_{-i}^k\right)^4} \times\left(\left(\gamma^k-\alpha_i^k\right)\left(b_i^k\right)^2+\gamma^k\sum\vec{\boldsymbol{b}}_{-i}^k b_i^k+\alpha_i^k\left(\sum\vec{\boldsymbol{b}}_{-i}^k\right)^2\right) > 0 \quad (25)$$

which means that $Z_i^k(b_i^k,\vec{\boldsymbol{b}}_{-i}^k)$ is monotonically increasing in b_l^k $\forall l\neq i$.

Let $y_i^*=g_i^k(\vec{\boldsymbol{y}})$, $x_i^*=g_i^k(\vec{\boldsymbol{x}})$. Then for each i, we have

$$Z_i^k\left(y_i^*,\vec{\boldsymbol{y}}_{-i}\right)=\varepsilon_i^k. \quad (26)$$

From $\vec{\boldsymbol{x}}\geq\vec{\boldsymbol{y}}$ and the monotonicity (25), we have $Z_i^k(y_i^*,\vec{\boldsymbol{x}}_{-i})\geq\varepsilon_i^k$.

Since $Z_i^k(x_i^*,\vec{\boldsymbol{x}}_{-i})=\varepsilon_i^k$ and $Z_i^k(\vec{\boldsymbol{b}}^k)$ is monotonically decreasing in b_i^k, we must have $x_i^*\geq y_i^*$. Therefore, 2) is established.

For 3), let $x_i^*=g_i^k(\vec{\boldsymbol{x}})$ and $\bar{x}_i^*=g_i^k(\beta\vec{\boldsymbol{x}})$. We then need to prove $\bar{x}_i^*<\beta x_i^*$ $\forall\beta>1$. Using the optimality condition, we have

$$Z_i^k\left(x_i^*,\vec{\boldsymbol{x}}_{-i}\right)=\varepsilon_i^k \quad (27)$$

and

$$Z_i^k\left(\bar{x}_i^*,\beta\vec{\boldsymbol{x}}_{-i}\right)=\varepsilon_i^k. \quad (28)$$

By virtue of contradiction, we assume that $\exists\beta>1$, such that $\bar{x}_i^*\geq\beta x_i^*$. It then follows from the monotonic decrease of $Z_i^k(\cdot,\vec{\boldsymbol{x}}_{-i})$ that

$$Z_i^k\left(\beta x_i^*,\beta\vec{\boldsymbol{x}}_{-i}\right)\geq\varepsilon_i^k. \quad (29)$$

On the other hand, from (22) and the fact that $\gamma_i^k(\vec{\boldsymbol{b}}^k)=\gamma_i^k(\beta\vec{\boldsymbol{b}}^k)$ due to the proportional sharing scheme in (5), it follows that:

$$Z_i^k\left(\beta x_i^*,\beta\vec{\boldsymbol{x}}_{-i}\right)=\frac{1}{\beta}Z_i^k\left(x_i^*,\vec{\boldsymbol{x}}_{-i}\right)<Z_i^k\left(x_i^*,\vec{\boldsymbol{x}}_{-i}\right)=\varepsilon_i^k \quad (30)$$

for all $\beta>1$. It is clear that (30) contradicts (29). 3) is thus established.

We now prove 4). It is straightforward from (22) that

$$Z_i^k\left(\beta\cdot\vec{\boldsymbol{b}}^k\right)=\frac{1}{\beta}\cdot Z_i^k\left(\vec{\boldsymbol{b}}^k\right)\ \forall\beta>0.$$

Then for a fixed $\vec{\boldsymbol{b}}^k\in\mathbb{R}_+^N$, we can find two constants: $\beta_-\in\mathbb{R}_+$ small enough and $\beta_+\in\mathbb{R}_+$ large enough, such that

$$Z_i^k\left(\beta_-\cdot\vec{\boldsymbol{b}}^k\right)>\varepsilon_i^k$$
$$Z_i^k\left(\beta_+\cdot\vec{\boldsymbol{b}}^k\right)<\varepsilon_i^k. \quad (31)$$

Set $\vec{\boldsymbol{g}}^k(\beta_-\cdot\vec{\boldsymbol{b}}^k)=\vec{\boldsymbol{b}}^{k-}$ and $\vec{\boldsymbol{g}}^k(\beta_+\cdot\vec{\boldsymbol{b}}^k)=\vec{\boldsymbol{b}}^{k+}$, meaning that

$$Z_i^k\left(b_i^{k-},\beta_-\cdot\vec{\boldsymbol{b}}_{-i}^k\right)=\varepsilon_i^k$$
$$Z_i^k\left(b_i^{k+},\beta_+\cdot\vec{\boldsymbol{b}}_{-i}^k\right)=\varepsilon_i^k. \quad (32)$$

Given that $Z_i^k(b_i^k,\vec{\boldsymbol{b}}_{-i}^k)$ is monotonically decreasing in b_i^k, combing (31) and (32) leads to

$$b_i^{k-}\geq\beta_-\cdot b_i^k\ \forall i=1,\ldots,N$$
$$b_i^{k+}\leq\beta_+\cdot b_i^k\ \forall i=1,\ldots,N. \quad (33)$$

Consequently, $\vec{\boldsymbol{g}}^k(\beta_-\cdot\vec{\boldsymbol{b}}^k)=\vec{\boldsymbol{b}}^{k-}\geq\beta_-\cdot\vec{\boldsymbol{b}}^k$ and $\vec{\boldsymbol{g}}^k(\beta_+\cdot\vec{\boldsymbol{b}}^k)=\vec{\boldsymbol{b}}^{k+}\leq\beta_+\cdot\vec{\boldsymbol{b}}^k$. Setting $\vec{\boldsymbol{x}}=\beta_-\cdot\vec{\boldsymbol{b}}^k$ and $\vec{\boldsymbol{y}}=\beta_+\cdot\vec{\boldsymbol{b}}^k$ concludes the proof. ■

We are now ready to show the existence and uniqueness of the Nash equilibrium of the induced game in Section III-C.

Theorem 1: The Nash equilibrium of the induced game in Section III-C uniquely exists.

Proof: The existence of a Nash equilibrium follows from Lemma 2 and the fixed point theorem developed in [37]. More specifically, by Property 4) of Lemma 2, there exist $\vec{\boldsymbol{x}},\vec{\boldsymbol{y}}\in\mathbb{R}_+^N$ with $\vec{\boldsymbol{x}}<\vec{\boldsymbol{y}}$ such that $\vec{\boldsymbol{x}}\leq\vec{\boldsymbol{g}}^k(\vec{\boldsymbol{x}})$ and $\vec{\boldsymbol{y}}\geq\vec{\boldsymbol{g}}^k(\vec{\boldsymbol{y}})$. Consider the sequence $\{(\vec{\boldsymbol{g}}^k)^n(\vec{\boldsymbol{x}})\}$, where the function $(\vec{\boldsymbol{g}}^k)^n$ is recursively defined

$$\left(\vec{\boldsymbol{g}}^k\right)^0(\vec{\boldsymbol{x}})=\vec{\boldsymbol{x}},\ \left(\vec{\boldsymbol{g}}^k\right)^n(\vec{\boldsymbol{x}})=\vec{\boldsymbol{g}}^k\left(\left(\vec{\boldsymbol{g}}^k\right)^{n-1}(\vec{\boldsymbol{x}})\right)\ \forall n\in\mathbb{N}_+.$$

Since the best-response function $\vec{\boldsymbol{g}}^k$ is continuous and monotonic (by Lemma 2) and $\vec{\boldsymbol{x}}\leq\vec{\boldsymbol{g}}^k(\vec{\boldsymbol{x}})$, we have $(\vec{\boldsymbol{g}}^k)^{n-1}(\vec{\boldsymbol{x}})\leq(\vec{\boldsymbol{g}}^k)^n(\vec{\boldsymbol{x}})$ $\forall n\in\mathbb{N}_+$. Therefore, $\{(\vec{\boldsymbol{g}}^k)^n(\vec{\boldsymbol{x}})\}_{n\in\mathbb{N}_+}$ forms an increasing chain. Moreover, $\vec{\boldsymbol{y}}>\vec{\boldsymbol{x}}$ implies that $(\vec{\boldsymbol{g}}^k)^n(\vec{\boldsymbol{y}})\geq(\vec{\boldsymbol{g}}^k)^n(\vec{\boldsymbol{x}})$ and $\vec{\boldsymbol{y}}\geq\vec{\boldsymbol{g}}^k(\vec{\boldsymbol{y}})$ implies that $\vec{\boldsymbol{y}}\geq(\vec{\boldsymbol{g}}^k)^n(\vec{\boldsymbol{y}})$ $\forall n\in\mathbb{N}_+$. It then implies that $\vec{\boldsymbol{y}}\geq(\vec{\boldsymbol{g}}^k)^n(\vec{\boldsymbol{x}})$, leading to that the chain $\{(\vec{\boldsymbol{g}}^k)^n(\vec{\boldsymbol{x}})\}$ is bounded; hence it has a supremum $\vec{\boldsymbol{x}}^*$

$$\vec{\boldsymbol{x}}^*=\sup_{n\in\mathbb{N}_+}\left(\vec{\boldsymbol{g}}^k\right)^n(\vec{\boldsymbol{x}}). \quad (34)$$

Therefore, $\vec{\boldsymbol{x}}^*$ is a fixed point of $\vec{\boldsymbol{g}}^k$ by [37, Th. 2], which establishes the existence of Nash equilibrium.

We next show the uniqueness of Nash equilibrium by contradiction. Suppose there exist two distinct Nash equilibria $\vec{\boldsymbol{x}},\vec{\boldsymbol{y}}\in\mathbb{R}_+^n$. Let $i^*=\arg\max_l(y_l/x_l)$ and $\beta=(y_{i^*}/x_{i^*})$. Without loss of generality, we assume $\beta>1$. Then we have $\beta\vec{\boldsymbol{x}}\geq\vec{\boldsymbol{y}}$ and $\beta x_{i^*}=y_{i^*}$. Note from Property 3) in Lemma 2 and the definition of Nash equilibrium, we have

$$g_{i^*}^k(\beta\vec{\boldsymbol{x}})<\beta g_{i^*}^k(\vec{\boldsymbol{x}})=\beta x_{i^*}. \quad (35)$$

On the other hand, from Property 2) in Lemma 2 and the fact that $\beta\vec{\boldsymbol{x}}\geq\vec{\boldsymbol{y}}$, we have

$$g_{i^*}^k(\beta\vec{\boldsymbol{x}})\geq g_{i^*}^k(\vec{\boldsymbol{y}})=y_{i^*}. \quad (36)$$

Combining (35) and (36) leads to $y_{i^*}<\beta x_{i^*}$, which contradicts the fact that $\beta x_{i^*}=y_{i^*}$. Hence, $\vec{\boldsymbol{x}}=\vec{\boldsymbol{y}}$ and the uniqueness of Nash equilibrium is established. ■

With some derivations it can be shown that the best response functions can be explicitly written as $g_i^k(\vec{\boldsymbol{b}}^k)=(1/(\gamma^k-\alpha_i^k))(\alpha_i^k\sum\vec{\boldsymbol{b}}_{-i}^k+\sqrt{(\gamma^k\sum\vec{\boldsymbol{b}}_{-i}^k)/\varepsilon_i^k(\sum_{j=1}^{m_i}\sqrt{w_{i,j}^k\alpha_{i,j}^k})})$, and then based on $b_i^{k*}=g_i^k(\vec{\boldsymbol{b}}^{k*})$, the best bid b_i^{k*} can be expressed with $\vec{\boldsymbol{b}}_{-i}^{k*}$. The analytic form of the best response function can be used to show the existence of a Nash equilibrium. However, the proof method using the fixed point theory can work with more general backlog function than the queue formulation in our application, e.g., $B_{i,j}^k$ in (3) is decreasing, convex, and twice differentiable, with $\lim_{x\to\alpha_{i,j}^k}B_{i,j}^k=+\infty$ on the interval $(\alpha_{i,j}^k,+\infty)$ [37].

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

V. Dynamics of Reaching the Equilibrium

A. Asynchronized Best Response Dynamics: Convergence to the Unique NE

The best-response function (19) suggests a natural update scheme (i.e., synchronized best response update): at each iteration, each vehicle submits the best bidding vector based on other vehicles' bid from the previous iteration. In a decentralized update setting, it may not be easy to enforce a fixed time step by which every vehicle updates simultaneously. Hence, we consider a more practical and easily implementable scheme (i.e., asynchronized best response update), where not every vehicle is necessary updating its bid at each iteration. To implement the update scheme, the cloud can distribute the bid of all other vehicles to each vehicle. However, this is not necessary. Note due to the proportional bidding scheme (5), each vehicle $\mathcal{V}_i$ can infer the total bid of all other vehicles at sever $\mathcal{S}_k$, i.e., $\sum \vec{\boldsymbol{b}}_{-i}^k$, based on its bid b_i^k and obtained resource γ_i^k from last iteration

$$\sum \vec{\boldsymbol{b}}_{-i}^k = \frac{\gamma^k - \gamma_i^k}{\gamma_i^k} b_i^k. \tag{37}$$

The vehicle can then adjust its bid using the best-response function (19).

This update scheme has several desired features. First, no communication is required to share the bidding information. Each vehicle can infer the total bid from other vehicles and then adapt its bid, making the scheme completely decentralized. Second, this scheme is secure. The cloud receives bid and allocate resources, with no capability of deducing the relevant information about the vehicles, such as backlog function, number of tasks, and weights. Moreover, no vehicle will be able to deduce such information from other vehicles since no information is shared between the vehicles. This provides an important basis for all vehicles to submit the true bidding vectors. Last but not least, this update scheme will be guaranteed to converge to the equilibrium, which we will show next. In the standpoint of the cloud, this bidding update is simply a fixed-point iteration where the iterates are the submitted bidding vectors. This iteration process is summarized in Algorithm 1. Note that in Algorithm 1, if $\mathcal{U}^t = \{1, 2, \ldots, N\}$ at every iteration t, i.e., every vehicle updates its bid at each iteration, then the asynchronized best response update scheme changes to the synchronized one.

We next show the convergence of asynchronized iteration in Algorithm 1, that is, as long as each vehicle updates its bid infinitely often, the bid iterate will converge to the unique Nash equilibrium. Toward that end, we introduce a proposition that will be used for the proof.

Proposition 1: Under Algorithm 1, let each vehicle $\mathcal{V}_i$ update its bid infinitely often, that is, $|\{t|i \in \mathcal{U}^t\}| = +\infty$. Then the tth bid iterate $\vec{\boldsymbol{b}}^k(t)$ converges to the unique Nash equilibrium irrespective of the initial bidding vector $\vec{\boldsymbol{b}}^{k0}$ if the following conditions hold.

1) There exists a sequence of nonempty subsets $\{\mathcal{G}(t)\}$ with $\mathcal{G}(t+1) \subset \mathcal{G}(t)$, $t = 0, 1, 2, \ldots$, and is such that if $\{\vec{\boldsymbol{x}}(t)\}$ is any sequence with $\vec{\boldsymbol{x}}(t) \in \mathcal{G}(t)$, then $\{\vec{\boldsymbol{x}}(t)\}$ converges pointwise to the unique Nash equilibrium.

Algorithm 1: Asynchronized Best Response Bidding Iteration

```
Each vehicle V_i chooses an arbitrary initial bidding vector b_i ∈ R_+^M ;
for iteration t = 0, 1, 2, ... do
    Let U^t ⊂ {1, 2, ..., N} be a possibly empty set of updating vehicles at iteration t;
    for i ∈ U^t do
        for k = 1, ··· , M do
            Observe received resource γ_i^k(t) from server S_k at step t;
            Compute the sum of the bid from all other vehicles at server S_k as
            Σ b_{-i}^k(t) = ((γ^k − γ_i^k(t)) / γ_i^k(t)) b_i^k(t);
            Update the bid b_i^k(t+1) = g_i^k(b^k(t));
        end
    end
end
```

2) *Synchronous Convergence Condition:* $\forall (\vec{\boldsymbol{g}}^k)^t(\vec{\boldsymbol{b}}^{k0}) \in \mathcal{G}(t)$, we have

$$\left(\vec{\boldsymbol{g}}^k\right)^{t+1}\left(\vec{\boldsymbol{b}}^{k0}\right) \in \mathcal{G}(t+1), \quad t = 0, 1, 2, \ldots$$

3) *Box Condition:* $\mathcal{G}(t)$ is a Cartesian product of the form

$$\mathcal{G}(t) = \mathcal{G}_1(t) \times \cdots \times \mathcal{G}_N(t), \quad t = 0, 1, 2, \ldots$$

where $\mathcal{G}_i(t)$ is a set of real-valued functions.

Proof: See [38, Proposition 2.6.1]. ■

Theorem 2: For all $k = 1, \ldots, M$, if each vehicle $\mathcal{V}_i$ updates its bid infinitely often, that is, $|\{t|i \in \mathcal{U}^t\}| = +\infty$, then Algorithm 1 terminates with $\vec{\boldsymbol{b}}_{\text{final}}^k$ being the unique Nash equilibrium.

Proof: To prove Theorem 2, we now need to show that there exits a sequence of nonempty subsets $\{\mathcal{G}(t)\}$ such that conditions 1)–3) in Proposition 1 are satisfied.

By the proof of Property 4), Lemma 2, for any initial bidding vector $\vec{\boldsymbol{b}}^{k0}$, there exist $\beta_- > 0$ small enough and $\beta_+ > 0$ large enough such that $\beta_-\vec{\boldsymbol{b}}^{k0} \leq \vec{\boldsymbol{g}}^k(\beta_-\vec{\boldsymbol{b}}^{k0})$ and $\beta_+\vec{\boldsymbol{b}}^{k0} \geq \vec{\boldsymbol{g}}^k(\beta_+\vec{\boldsymbol{b}}^{k0})$ with

$$\beta_-\vec{\boldsymbol{b}}^{k0} \leq \vec{\boldsymbol{b}}^{k0} \leq \beta_+\vec{\boldsymbol{b}}^{k0}. \tag{38}$$

Therefore, by repeatedly applying $\vec{\boldsymbol{g}}^k$ to the three sides of (38), it follows from (34) in the proof of Theorem 1 that both $(\vec{\boldsymbol{g}}^k)^t(\beta_+\vec{\boldsymbol{b}}^{k0})$ and $(\vec{\boldsymbol{g}}^k)^t(\beta_-\vec{\boldsymbol{b}}^{k0})$ converge as $t \to \infty$ (setting $\vec{\boldsymbol{x}} = \beta_-\vec{\boldsymbol{b}}^{k0}$ and $\vec{\boldsymbol{y}} = \beta_+\vec{\boldsymbol{b}}^{k0}$). The corresponding limits are fixed points of $\vec{\boldsymbol{g}}^k$ and are therefore Nash equilibria. Furthermore, by the uniqueness of Nash equilibrium established in Theorem 1, the two equilibria must coincide and it then follows that $(\vec{\boldsymbol{g}}^k)^t(\vec{\boldsymbol{b}}^{k0})$ also converges to the unique Nash equilibrium since $(\vec{\boldsymbol{g}}^k)^t(\beta_-\vec{\boldsymbol{b}}^{k0}) \leq (\vec{\boldsymbol{g}}^k)^t(\vec{\boldsymbol{b}}^{k0}) \leq (\vec{\boldsymbol{g}}^k)^t(\beta_+\vec{\boldsymbol{b}}^{k0})$. Based on the above analysis, the sequence of nonempty subsets $\{\mathcal{G}(t)\}$ can be constructed as

$$\mathcal{G}(t) = \left\{\vec{\boldsymbol{x}} \middle| \left(\vec{\boldsymbol{g}}^k\right)^t\left(\beta_-\vec{\boldsymbol{b}}^{k0}\right) \leq \vec{\boldsymbol{x}} \leq \left(\vec{\boldsymbol{g}}^k\right)^t\left(\beta_+\vec{\boldsymbol{b}}^{k0}\right)\right\} \tag{39}$$

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

which guarantees conditions 1) and 2) in Proposition 1 are satisfied.

In addition, from the definition of $\vec{\boldsymbol{g}}^k(\cdot)$, it can be obtained that $\mathcal{G}_i(t) = \{x | g_i^k((\vec{\boldsymbol{g}}^k)^{t-1}(\beta_- \vec{\boldsymbol{b}}^{k0})) \leq x \leq g_i^k((\vec{\boldsymbol{g}}^k)^{t-1}(\beta_+ \vec{\boldsymbol{b}}^{k0}))\}$, indicating condition 3) is satisfied. Therefore, we can conclude that Algorithm 1 terminates with $\vec{\boldsymbol{b}}^k_{\text{final}}$ being the unique Nash equilibrium. ■

Denote t_e as the time for running steps 6–8 of Algorithm 1 (i.e., the time for updating the bid b_i^k) and T_{final} as the iteration step when the Nash equilibrium is achieved. Then the main factor for the complexity of Algorithm 1 is the number of total iterations taken to update b_i^k. Note that the exact number of total iterations cannot be determined as $\mathcal{U}^t$ is a random subset of $\{1, 2, \ldots, N\}$ at each iteration t. Therefore, we consider the maximum iteration number as the implementation complexity (i.e., the worst-case complexity). From the view of the entire algorithm, the maximum iteration number taken to update the bid b_i^k is $T_{\text{final}}MN$, indicating that the maximum computation time and worse-case complexity of Algorithm 1 are $T_{\text{final}}MNt_e$ and $O(T_{\text{final}}MN)$, respectively.

The update mechanism in Algorithm 1 is myopic processes where each vehicle optimizes its bid based on the bid from the previous iteration. During the procedure to reach the Nash equilibrium, some vehicles maybe cannot bid for enough source to process the onboard applications. However, the vehicle-level resource allocation takes the importance of different application into consideration, which guarantees that more important task will have higher priority to get the resources. To make sure most of the vehicles can bid for enough source, we can specify a reasonable range of bidding value or add soft margins in the optimization problems. In fact, we note that as long as the update mechanism converges to the unique Nash equilibrium and is computationally feasible, the mechanism can be adopted. The long-term cost of the vehicles only depends on the derived resources from the bidding equilibrium, and the process by which the equilibrium is reached can be viewed as a "transient" process that has negligible impact on the vehicle's long-term utility.

B. Stochastic Stability of Best Response Updates Under Random Environments

So far the existence and uniqueness of the Nash equilibrium for the induced cloud resource game are treated with fixed task arrival rates. In practice, the task arrival rates may vary as the cloud applications may run different sets of features under different road or traffic conditions. For example, the cloud-aided semi-active suspension application [5] needs to perform more frequent control tasks on rough roads as compared to flat and smooth roads. The objective of this section is to characterize the behavior of the best response dynamics we proposed earlier under a stochastic environment.

Note when the task arrival rate varies at every time step, the bidding vector will not converge to a fixed point, i.e., the Nash equilibrium, as shown before. Therefore, we need a new metric to quantify the stability of the proposed best response dynamics (synchronized and asynchronized) proposed in the preceding sections. Specifically, we exploit the TVD with the following definition.

TABLE I
SIMULATION PARAMETERS

# of servers (M)	# of cars (N)	# of Apps (m_i)
3	100	$\sim \mathcal{U}\{5, 15\}$
arrival rate ($\lambda_{i,j}$)	weights ($w_{i,j}^k/\varepsilon_i^k$)	total resources at sever $\mathcal{S}_k$ (γ^k)
$\sim \mathcal{U}(0.01, 1)$	$\sim \mathcal{U}(0.8, 1.5)/\mathcal{U}(5, 10)$	$2 \cdot \sum_{i=1}^{N} \sum_{j=1}^{m_i} \alpha_{i,j}^k$

Definition 3 [39]: Let μ and η be two probability measures over a finite set Ω. The total variation distance between μ and η (also called statistical distance) is the normalized l_1-distance between the two probability measures

$$\|\mu - \eta\|_{\text{tv}} \coloneqq \frac{1}{2} \sum_{x \in \Omega} |\mu(x) - \eta(x)|. \tag{40}$$

Moreover, if X and Y are two random variables drawn from μ and η, respectively, we will write $\|X - Y\|_{\text{tv}}$ to denote $\|\mu - \eta\|_{\text{tv}}$.

Let $P_i(K)$ be the random bid the vehicle i submits at time step K by following a best response dynamics (e.g., synchronized or asynchronized). The stability metric (SM) of the best response dynamics for vehicle i can be characterized by

$$\text{SM}_i(K) = \|P_i(K) - P_i(K+1)\|_{\text{tv}}, i = 1, \ldots, N \tag{41}$$

with a high enough K. Intuitively, this metric captures how different the bid is distributed between step K and step $K+1$. We will perform comprehensive simulations to compute the SM (41) in the next section.

VI. NUMERICAL EXAMPLES

In this section, we present numerical simulations to demonstrate the developed resource allocation framework. We will first consider the scenario with fixed task arrival rates and verify the convergence results we proved earlier. We will then perform simulations under a stochastic environment with random task rates and demonstrate the bidding stability of the proposed best response dynamics.

A. Bidding and Cost Convergence With Fixed Arrival Rates

We consider a network of three servers and a fleet of 100 vehicles with each vehicle running a random number of applications ranging from 5 to 15. For each task $T_{i,j}$, the arrival rate $\lambda_{i,j}$ is sampled from a uniform distribution between 0.01 and 1, and the routing matrix $\boldsymbol{P}_{i,j}$ is randomly generated. Based on (2), the effective arrival rate $\alpha_{i,j}^k$ (i.e., the minimum required resources of task $T_{i,j}$ at server $\mathcal{S}_k$) can be obtained. The weights $w_{i,j}^k$ that characterize the relative importance of different tasks are sampled from a uniform distribution between 0.8 and 1.5, while the weights ε_i^k that tradeoff the bid price and backlog cost are sampled from a uniform distribution between 5 and 10. The total resource of sever $\mathcal{S}_k$ is set as twice of the summed minimum resources from all vehicles. These parameters are summarized in Table I.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

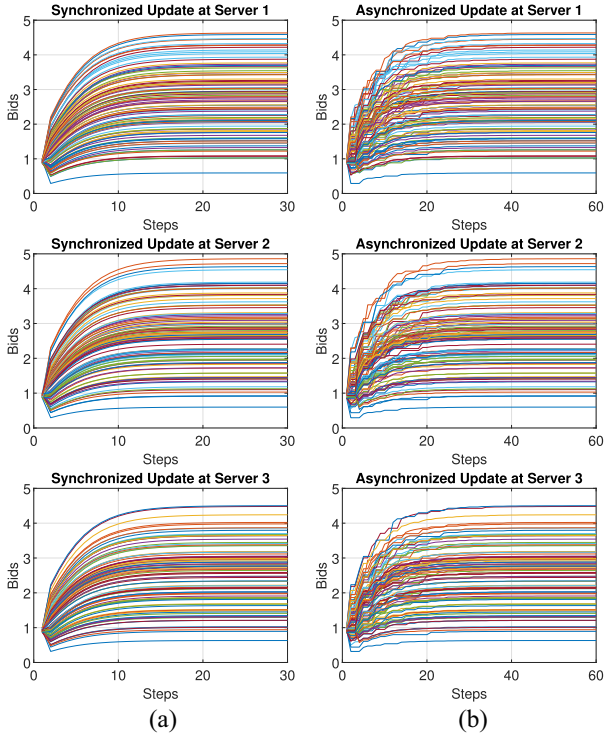


Fig. 4. Bidding evolution of synchronized update (left) and asynchronized update (right).

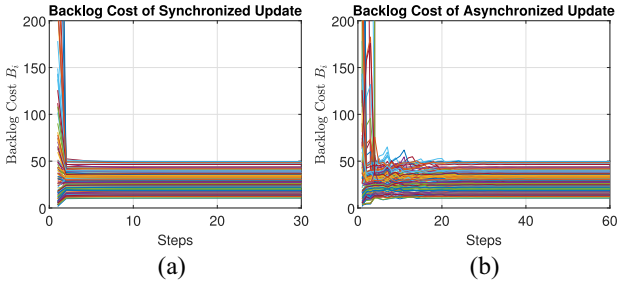


Fig. 5. Backlog evolution of synchronized update (left) and asynchronized update (right).

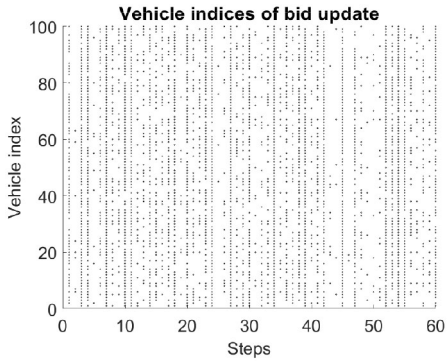


Fig. 6. Indices of vehicles that performed bid update at each time step.

The bidding trajectory and the backlog cost of each vehicle under the synchronized update mechanism are shown in Figs. 4(a) and 5(a), respectively. For the synchronized update mechanism, every vehicle updates its bidding vector at each iteration based on its bid and corresponding obtained resource from the last iteration. From Figs. 4(a) and 5(a), it is clear that both the bidding vectors and the backlog cost converge. Moreover, under the MATLAB environment (i.e., MATLAB 2019b, Intel Core i7-10710U CPU@1.10 GHz), the synchronized update mechanism takes 0.1455 s to reach the Nash equilibrium.

The bidding evolution of the asynchronized update mechanism (Algorithm 1) and its corresponding backlog cost are shown in Figs. 4(b) and 5(b), respectively. At each iteration, a random subset of the vehicle fleet update their bidding vector based on the best-response function. The random subset is illustrated in Fig. 6. It is clear that the bidding vectors and backlog cost also converge to the same Nash equilibrium as in the synchronized update. Note that the asynchronized update mechanism does have a slower convergence, which is reasonable since the synchronized approach updates the bid of all vehicles at each step whereas the asynchronized approach only updates a subset of the vehicles; less frequent update thus leads to slower convergence. The asynchronized update mechanism takes 0.1689 s to converge to the Nash equilibrium.

To better evaluate the performance of the Nash equilibrium solution achieved by Algorithm 1, three alternative allocation methods are chosen to conduct the comparison. Specifically, Method 1 assumes that the vehicles mistrust the cloud and no bidding mechanism is introduced to facilitate the resource allocation. Under this circumstance, each server in the cloud evenly distributes its resources to subscribed vehicles, and then each vehicle will optimally allocate its obtained resources to onboard applications as discussed in Section III-B. To evaluate the efficacy of the linear-proportional sharing scheme (5), Method 2 utilizes the following nonlinear sharing scheme to allocate the resource:

$$\gamma_i^k = \begin{cases} \gamma^k \times \frac{\ln\left(1+b_i^k\right)}{\sum_{l=1}^{N} \ln\left(1+b_l^k\right)}, & \text{if } \sum_{l=1}^{N} b_l^k > 0 \\ 0, & \text{otherwise} \end{cases} \tag{42}$$

and the remaining is the same to the proposed method. The logarithm function $\ln(1+b_i^k)$ in (42) is monotonically increasing and grows very slowly for large b_i^k, which can restrain the impact of overlarge bids on the sharing model. For Method 3, it is assumed that the cloud is trusted by all vehicles, and each vehicle will submit all application-related information to the cloud. Based on the detailed vehicular information, the cloud adopts a centralized scheme to find the optimal allocation resource by minimizing the total vehicle backlog cost, i.e., $\sum_{i=1}^{N} B_i$. This centralized scheme is essentially the optimal strategy that we use to compare with our proposed approach and evaluate the performance gap. The comparison results are presented in Table II. It can be seen that the proposed approach is more effective in reducing the backlog cost compared to Methods 1 and 2, indicating that the proportional sharing bidding scheme is effective in making proper resource allocations. Furthermore, the efficiency of the Nash equilibrium solution is then evaluated by comparing the equilibrium performance with the performance of centralized optimal solution (Method 3). It can be found from Table II

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

TABLE II
COMPARISON OF TOTAL BACKLOG COST

	Method 1	Method 2	Method 3	Proposed
$\sum_{i=1}^{N} B_i$	5640.6	3067.7	2916.1	2958.2

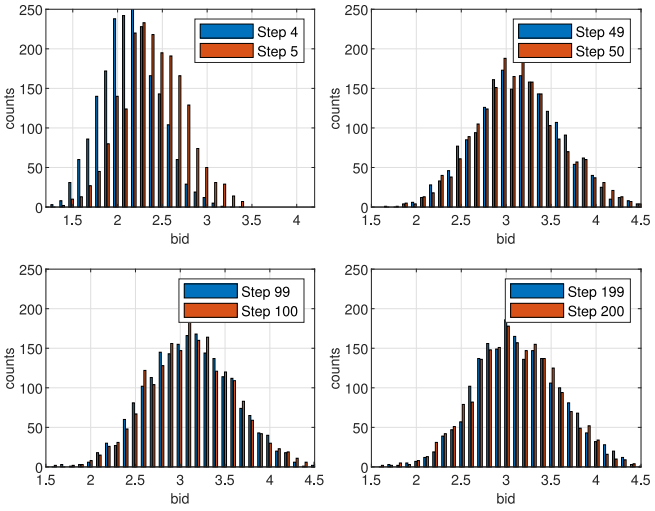


Fig. 7. Histogram of vehicle 1's bid submitted to server 1 at steps 4, 5, 49, 50, 99, 100, 199, and 200 under i.i.d. task arrival rate with a bin size of 0.01.

that the backlog cost of the proposed Nash equilibrium solution is quite close to the one of centralized optimal approach (i.e., Method 3). In the meantime, the proposed framework is hierarchical and decentralized, which is more secure and light-weight in communication.

B. Stochastic Stability Under Random Task Arrival Rates

In this section, we perform simulations and quantitatively characterize the stochastic stability when the task arrival rates vary randomly at each time step. We use the same set of the simulation parameters as in Table I except we now use a fleet of ten vehicles ($N = 10$) for the simplification of visualization and the total resource of each server is set at 160 ($\gamma^k = 160$) to ensure the cloud can provide enough resource for task processing. Instead of the arrival rate $\lambda_{i,j}$ being held constant across the entire time horizon, we consider random task arrival rate at each time step with two stochastic modeling: 1) identically and independent distributed (i.i.d.) and 2) Markov chains (MCs). For the i.i.d. case, at each time step, we sample $\lambda_{i,j}$ from a uniform distribution between 0.1 and 1, i.e., $\lambda_{i,j} \sim \mathcal{U}(0.1, 1)$. For the MC case, we have three Markov states for $\lambda_{i,j}$, and each state corresponds to a value randomly sampled from a uniform distribution between 0.1 and 1. The transition matrices of Markov states are also randomly generated using MATLAB function *mcmix*(3).

We first simulate the bidding process under the i.i.d. case using the synchronized best response dynamics over a horizon of 200 steps for 2000 times. The histograms of the bid submitted to server 1 at steps 4, 5, 49, 50, 99, 100, 199, and 200 for vehicle 1 with a bin size of 0.1 are shown in Fig. 7. It can be seen that the distributions at the initial two sequential steps

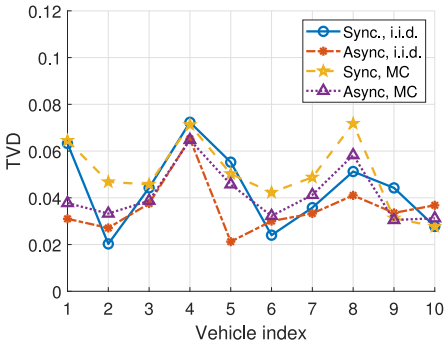


Fig. 8. Summary of TVD under different simulation setups.

are very different. While as the step increases, the distributions at two sequential steps become similar, which indicates that the bid vector will converge to a stationary distribution. With the histogram in Fig. 7, we normalize the counts and translate it to the probability of falling into each of the bins. We then apply (40) and (41) to compute the TVD between the two distributions at step 199 and step 200. Furthermore, we perform simulations for the case that the task arrival rates are subject to MC. Similarly, we simulate the bidding process under the MC formulation over a 200 step horizon for 2000 times, draw the histogram, and compute the TVD. In addition, we follow the same stochastic environment setup and implement the asynchronized best response dynamics. A summary of the TVD for the simulations is shown in Fig. 8. Specifically, under four different simulation setups, each vehicle's TVD between the sequential distributions of step 199 and step 200 is presented. From Fig. 8, it is clear that the synchronized and asynchronized best response dynamics have low TVD under both i.i.d. and MC stochastic arriving rates, demonstrating their robustness to the task arrival rate variations.

VII. CONCLUSION

In this article, we developed a hierarchical resource allocation model for cloud-aided automotive systems. At high level, an auction-based proportional-sharing scheme is exploited for cloud resource allocation, which is induced to a multiplayer game. At low level, a constrained optimization problem is solved to optimally allocate the obtained resource to onboard applications. The induced multiplayer game is studied from the perspective of Nash equilibrium, which facilitates the design of an asynchronized bid update mechanism. Simulations were presented to demonstrate the efficacy of proposed framework. In particular, the convergence of the bids in the stochastic setting indicates promise as a useful model, as the cloud settings and vehicle applications are complex and time varying. In the future, we will extend the current theoretical results to more general stochastic environment. Furthermore, it would also be interesting to consider more characteristics of cloud-enabled vehicle applications (e.g., mobility, time-delay, and energy constraints), when establishing the resource allocation model and adopt learning-based approaches to simultaneously learn the model parameters and perform resource assignment.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] D. Filev, J. Lu, and D. Hrovat, "Future mobility: Integrating vehicle control with cloud computing," *Mech. Eng.*, vol. 135, no. 3, pp. 18–24, 2013.

[2] Z. Li, "Developments in estimation and control for cloud-enabled automotive vehicles," Ph.D. dissertation, Dept. Aerosp. Eng., Univ. Michigan, Ann Arbor, MI, USA, 2016.

[3] E. Ozatay *et al.*, "Cloud-based velocity profile optimization for everyday driving: A dynamic-programming-based solution," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 6, pp. 2491–2505, Dec. 2014.

[4] Z. Li, I. Kolmanovsky, E. Atkins, J. Lu, D. P. Filev, and J. Michelini, "Road risk modeling and cloud-aided safety-based route planning," *IEEE Trans. Cybern.*, vol. 46, no. 11, pp. 2473–2483, Nov. 2016.

[5] Z. Li, I. Kolmanovsky, E. Atkins, J. Lu, D. Filev, and J. Michelini, "Cloud aided semi-active suspension control," in *Proc. IEEE Symp. Comput. Veh. Transp. Syst.*, 2014, pp. 76–83.

[6] I. Menache, A. Ozdaglar, and N. Shimkin, "Socially optimal pricing of cloud computing resources," in *Proc. Int. ICST Conf. Perform. Eval. Methodol. Tools*, 2011, pp. 322–331.

[7] F. Farahnakian *et al.*, "Using ant colony system to consolidate VMs for green cloud computing," *IEEE Trans. Services Comput.*, vol. 8, no. 2, pp. 187–198, Mar./Apr. 2015.

[8] W. Song, Z. Xiao, Q. Chen, and H. Luo, "Adaptive resource provisioning for the cloud using online bin packing," *IEEE Trans. Comput.*, vol. 63, no. 11, pp. 2647–2660, Nov. 2014.

[9] W. Wang, Y. Jiang, and W. Wu, "Multiagent-based resource allocation for energy minimization in cloud computing systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 2, pp. 205–220, Feb. 2017.

[10] D. Ardagna, B. Panicucci, and M. Passacantando, "A game theoretic formulation of the service provisioning problem in cloud systems," in *Proc. Int. Conf. World Wide Web*, 2011, pp. 177–186.

[11] D. Ardagna, B. Panicucci, and M. Passacantando, "Generalized Nash equilibria for the service provisioning problem in cloud systems," *IEEE Trans. Services Comput.*, vol. 6, no. 4, pp. 429–442, Oct.–Dec. 2013.

[12] R. Johari, S. Mannor, and J. N. Tsitsiklis, "Efficiency loss in a network resource allocation game: The case of elastic supply," *IEEE Trans. Autom. Control*, vol. 50, no. 11, pp. 1712–1724, Nov. 2005.

[13] A. Dimakis, R. Jain, and J. Walrand, "Mechanisms for efficient allocation in divisible capacity networks," in *Proc. IEEE Conf. Decis. Control*, 2006, pp. 1264–1269.

[14] R. T. Maheswaran and T. Başar, "Auctions for divisible resources: Price functions, Nash equilibrium, and decentralized update schemes," in *Proc. Int. Workshop Agent Mediated Electron. Commerce*, 2002, pp. 87–102.

[15] Z. Zhou and N. Bambos, "A general model for resource allocation in utility computing," in *Proc. Amer. Control Conf.*, 2015, pp. 1746–1751.

[16] K. Schubert, N. Master, Z. Zhou, and N. Bambos, "Asynchronous best-response dynamics for resource allocation games in cloud computing," in *Proc. Amer. Control Conf.*, 2017, pp. 4613–4618.

[17] A. Ward, Z. Zhou, and N. Bambos, "Bidding-based dynamic power pricing scheme in smart grids," in *Proc. Int. Conf. Comput. Netw. Commun.*, 2019, pp. 729–734.

[18] X. Sun, S. Su, P. Xu, S. Chi, and Y. Luo, "Multi-dimensional resource integrated scheduling in a shared data center," in *Proc. Int. Conf. Distrib. Comput. Syst. Workshops*, 2011, pp. 7–13.

[19] B. N. Chun and D. E. Culler, "Market-based proportional resource sharing for clusters," Dept. Electr. Eng. Comput. Sci., Univ. California, Berkeley, CA, USA, Rep. CSD-1092, 2000.

[20] K. Lai, B. A. Huberman, and L. R. Fine, "Tycoon: A distributed market-based resource allocation system," 2004, *arXiv:cs/0412038*.

[21] H. Meng, K. Zheng, P. Chatzimisios, H. Zhao, and L. Ma, "A utility-based resource allocation scheme in cloud-assisted vehicular network architecture," in *Proc. IEEE Int. Conf. Commun. Workshop*, 2015, pp. 1833–1838.

[22] K. Zheng, H. Meng, P. Chatzimisios, L. Lei, and X. Shen, "An SMDP-based resource allocation in vehicular cloud computing systems," *IEEE Trans. Ind. Electron.*, vol. 62, no. 12, pp. 7920–7928, Dec. 2015.

[23] R. I. Meneguette, A. Boukerche, and A. H. M. Pimenta, "AVARAC: An availability-based resource allocation scheme for vehicular cloud," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3688–3699, Oct. 2019.

[24] C.-C. Lin, D.-J. Deng, and C.-C. Yao, "Resource allocation in vehicular cloud computing systems with heterogeneous vehicles and roadside units," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3692–3700, Oct. 2018.

[25] H. R. Arkian, R. E. Atani, A. Diyanat, and A. Pourkhalili, "A cluster-based vehicular cloud architecture with learning-based resource management," *J. Supercomput.*, vol. 71, no. 4, pp. 1401–1426, 2015.

[26] Z. Li, T. Chu, I. V. Kolmanovsky, X. Yin, and X. Yin, "Cloud resource allocation for cloud-based automotive applications," *Mechatronics*, vol. 50, pp. 356–365, Apr. 2018.

[27] H. Peng and X. Shen, "Deep reinforcement learning based resource management for multi-access edge computing in vehicular networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2416–2428, Oct.–Dec. 2020.

[28] W. Wei, R. Yang, H. Gu, W. Zhao, C. Chen, and S. Wan, "Multi-objective optimization for resource allocation in vehicular cloud computing networks," *IEEE Trans. Intell. Transp. Syst.*, early access, Aug. 3, 2021, doi: 10.1109/TITS.2021.3091321.

[29] J. P. Hubaux, S. Capkun, and J. Luo, "The security and privacy of smart vehicles," *IEEE Security Privacy*, vol. 2, no. 3, pp. 49–55, May/Jun. 2004.

[30] M. Xue, W. Wang, and S. Roy, "Security concepts for the dynamics of autonomous vehicle networks," *Automatica*, vol. 50, no. 3, pp. 852–857, 2014.

[31] J. F. Shortle, J. M. Thompson, D. Gross, and C. M. Harris, *Fundamentals of Queueing Theory*. Hoboken, NJ, USA: Wiley, 2018.

[32] J. R. Jackson, "Jobshop-like queueing systems," *Manag. Sci.*, vol. 10, no. 1, pp. 131–142, 1963.

[33] R. D. Yates, "A framework for uplink power control in cellular radio systems," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 7, pp. 1341–1347, Sep. 1995.

[34] R. Johari and J. N. Tsitsiklis, "Efficiency of scalar-parameterized mechanisms," *Oper. Res.*, vol. 57, no. 4, pp. 823–839, 2009.

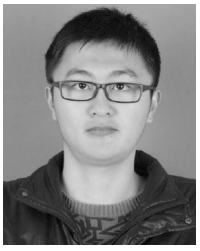
[35] B. Duncan, "Pumpkin pies and public goods: The raffle fundraising strategy," *Public Choice*, vol. 111, nos. 1–2, pp. 49–71, 2002.

[36] F. Teng and F. Magoules, "Resource pricing and equilibrium allocation policy in cloud computing," in *Proc. IEEE Int. Conf. Comput. Inf. Technol.*, 2010, pp. 195–202.

[37] Z. Zhou, N. Bambos, and P. Glynn, "Deterministic and stochastic wireless networks games: Equilibrium, dynamics and price of anarchy," *Oper. Res.*, vol. 66, no. 6, pp. 1498–1516, 2018.

[38] D. P. Bertsekas, *Abstract Dynamic Programming*. Belmont, MA, USA: Athena Sci., 2018.

[39] C. Daskalakis. "Probability and computation." 2011. [Online]. Available: http://people.csail.mit.edu/costis/6896sp11/



Kaixiang Zhang received the B.Eng. degree in automation from Xiamen University, Xiamen, China, in 2014, and the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2019.

From November 2017 to August 2018, he was a Visiting Scholar with the National Institute for Research in Computer Science and Automation, Rennes, France. He is currently a Postdoctoral Fellow with the Department of Mechanical Engineering, Michigan State University, East Lansing, MI, USA. His research interests include visual servoing, robotics, and nonlinear control.

Zhaojian Li (Senior Member, IEEE) received the B.Eng. degree in civil aviation from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2010, and the M.S. and Ph.D. degrees in aerospace engineering (flight dynamics and control) from the University of Michigan at Ann Arbor, Ann Arbor, MI, USA, in 2013 and 2015, respectively.

He is currently an Assistant Professor with the Department of Mechanical Engineering, Michigan State University, East Lansing, MI, USA. His research interests include learning-based control, nonlinear and complex systems, and robotics and automated vehicles.

Dr. Li is a recipient of NSF CAREER Award.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.

Xiang Yin (Member, IEEE) was born in Anhui, China, in 1991. He received the B.Eng. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2012, and the M.S. and Ph.D. degrees in electrical engineering from the University of Michigan at Ann Arbor, Ann Arbor, MI, USA, in 2013 and 2017, respectively.

Since 2017, he has been with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China, where he is an Associate Professor. His research interests include formal methods, discrete-event systems, and cyber–physical systems.

Dr. Yin also received the IEEE Conference on Decision and Control Best Student Paper Award Finalist in 2016. He is the Co-Chair of the IEEE CSS Technical Committee on Discrete Event Systems. He is also a member of the IEEE CSS Conference Editorial Board.

Liang Han (Member, IEEE) received the B.Eng. degree in automation from the Nanjing University of Science and Technology, Nanjing, China, in 2011, and the Ph.D. degree in control science and engineering from Beihang University, Beijing, China, in 2017.

He was a Research Scholar with the Department of Aerospace Engineering, University of Michigan at Ann Arbor, Ann Arbor, MI, USA, from 2013 to 2014. He is currently an Associate Professor with the Sino-French Engineer School, Beihang University, Beijing, China. His research interests include aircraft guidance, navigation and control, and cooperative control of multiagent systems.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on July 31,2023 at 09:00:48 UTC from IEEE Xplore. Restrictions apply.