# Dynamic Resource Allocation and Pricing for Edge-Assisted Metaverse

Valensia Sebastiani*
*ECE department*
*National University of Singapore*
Singapore
valensia@u.nus.edu

Alakesh Kalita*
*ECE department*
*National University of Singapore*
Singapore
alakesh@nus.edu.sg

Mohan Gurusamy
*ECE department*
*National University of Singapore*
Singapore
gmohan@nus.edu.sg

***Abstract*—Metaverse has become an increasingly popular virtual service platform, leading to a growing demand for computational resources to support it. However, there remains a challenge for efficient allocation of resources between *virtual service providers*(VSPs) and *edge service providers* (ESPs) as the existing works on Metaverse did not consider such resource allocation problem. Furthermore, we note that there is a need for dynamic pricing for getting service from the ESPs by the VSPs based on the quality of service provided by the ESPs and type of service *i.e.,* either *high-priority* or *low-priority* requested by the VSPs. To address these issues, we propose a *Dynamic Resource Allocation and Pricing* (`DRAP`) framework. The proposed framework considers the impact of VSPs and ESPs' behaviours on resource allocation, including the implicit involvement of end users for efficient resource utilization. Later, we leverage the multi-player *Stackelberg gaming model*, where VSPs act as leaders and ESPs act as followers to find the optimal solutions for the VSPs and ESPs of the proposed framework. Through numerical experiments, we demonstrate the effectiveness of our proposed framework in improving resource utilization, making it a promising solution for Metaverse.**

***Index Terms*—Metaverse, Resource allocation, Stackelberg Game, Edge Computing**

## I. Introduction

Metaverse has been gaining tremendous popularity as it is capable of merging the physical world and digital virtuality that lies beyond the current universe [1]–[3]. With established tech giants such as Google, Meta, Microsoft, and Unity taking an interest in Metaverse publicly, it is reasonable to expect even further advancements to it in the years to come [4]. As the Metaverse is in a nascent stage and has started expanding day by day, researchers have to deal with several issues, including efficient resource allocation in terms of computation and bandwidth in various stages to ensure that end users have an enjoyable and seamless experience [5], [6]. Edge computing brings computation and data storage closer to the end user of Metaverse system so it can improve response times in the Metaverse by reducing latency, caching content locally, distributing processing tasks, and optimizing network bandwidth. It enables real-time interactions, secure transactions, and immersive experiences. To support the operations of the Metaverse, Virtual Service Providers (VSPs) require a significant amount of computational resources. VSPs directly interact with the end users for quick response to the users' requests by managing resources from Edge Service Providers (ESPs) in the edge-assisted Metaverse [7], [8].

*A. Motivation*

The VSPs provide the necessary computational resources, interaction interfaces, and infrastructures to create and maintain the virtual environment for the end users. On the other side, there exist ESPs that support VSPs in the operation of the Metaverse by providing the required resources near the users, such as computing power, storage, and network bandwidth. So, how well the Metaverse operates for the end users depends greatly on this interaction of effective resource allocation between the VSPs and ESPs. Traditionally, resources are pre-allocated with overestimating the demands to allow room for peaking demands. This approach leads to under-utilization and wastage of resources, severely affecting the revenue generated by the Metaverse. Additionally, pre-allocating resources in the Metaverse can hinder scalability by imposing fixed limits, leading to inefficiencies and inflexibility. To achieve scalability, dynamic resource allocation and on-demand provisioning are crucial for adapting to changing user demands and optimizing resource utilization. In brief, static and pre-allocation of resources affect the scalability of Metaverse. To counter the issues of static and pre-allocation, dynamic resource allocation (*i.e.,* allocating resources depending on requirement) can improve resource utilization and enhance the overall performance of the Metaverse. By dynamically allocating resources from ESPs, VSPs optimize their resource utilization and reduce operational costs while providing high-quality services to end users. Furthermore, some of the Metaverse applications require real-time computing, which we can refer to as high-priority jobs, for which resources should be allocated immediately, while some Metaverse applications do not require such real-time allocation, which we can refer to as low-priority jobs. Even though dynamic resource allocation schemes for Metaverse have been proposed by different researchers, the existing works [9]–[13] neither consider the resource allocation between the VSPs and ESPs nor consider high- and low-priority jobs based allocation.

Apart from this resource allocation problem, the pricing for the allocating resources from ESPs by the VSPs should be

*Authors contributed equally to this work.

2023 IEEE Future Networks World Forum (FNWF) | 979-8-3503-2458-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/FNWF58287.2023.10520653

based on the service provided by the ESPs and types of service (*i.e.,* high- and low-priority) requested by the VSPs to increase the revenue of both individual entities *i.e.,* VSPs and ESPs. Otherwise, ESPs would not deliver better quality of service (*e.g.,* quick processing of data, high sampling rate, enough bandwidth) requested by the VSPs and VSPs would not make sufficient payment to the ESPs. Therefore, there is a need of dynamic pricing between the VSPs and ESPs during resource allocation so that both entities can optimize their profits while providing better services to the end users.

### B. Our contribution

To deal with the above-mentioned issues, in this work, we propose a framework for dynamically allocating resources between the VSPs and ESPs in Metaverse to implicitly satisfy the end users' requirements. In our framework, the VSPs can leverage the resources from ESPs to complete paid jobs from end users and, concurrently, the ESPs can earn based on their contribution of resources to the VSPs. Specifically, we handle different services such as both high- and low-priority jobs from VSPs (*i.e.,* requested by the end users) to ESPs for the corresponding resource allocation considering $M/M/1$ queue model. We also incorporated dynamic pricing between the VSPs and ESPs which varies depending on the quality of service provided by ESPs and service requested by the VSPs. In order to find the optimal solutions *i.e.,* optimal resource allocation with optimized price for the VSPs and ESPs, we incorporate our model to a Stackelberg gaming model, where the VSPs behave as leaders and ESPs behave as followers. None of the works considered dynamic pricing for high and low-priority jobs together in the literature. We solve the Stackelberg gaming model to find the optimal solutions for VSPs and ESPs. Finally, we perform numerical analysis to show efficacy of our proposed scheme. The major contributions of this work are as follows,

- We design a framework to handle different services at the VSPs such as both high- and low-priority jobs from end users and based on the service requests, VSPs request ESPs for the corresponding resource allocation.
- We incorporate dynamic pricing between the VSPs and ESPs in our proposed framework.
- Next, we leverage the Stackelberg gaming to find optimal solutions for VSPs and ESPs of the proposed framework.
- We perform numerical analysis to demonstrate the effectiveness of our proposed scheme `DRAP` compared to the Greedy approach.

The rest of the paper is organized as follows: Section II discusses the works on resource allocation in Metaverse. Section III briefly discusses the proposed system model for our framework, and Section IV discusses the `DRAP` framework. In Section V, we discuss the Stackelberg gaming-based solution approach and numerical analysis in Section VI. Finally, we conclude this work in Section VII.

## II. Related Work

Only a few works have been carried out in the literature related to resource allocation and service pricing in Metaverse. The work in [9] focuses on the Metaverse's virtual education system and offers a stochastic optimal resource allocation strategy for VSPs and cyber resources like edge or cloud services. The authors' numerical research showed that their suggested strategy allocates resources efficiently considering customer demand. However, the complexity of environments with numerous VSPs and cyber resources was not taken into consideration in their study. Researchers investigated how smart devices and VSPs allocated resources while gathering data from the physical environment in [14], [15]. The authors used a hybrid evolutionary dynamics approach in [14] and QoE-aware resource allocation in [15] in which the owners of smart devices modify their tactics over time in order to maximise their revenue. In [11], authors presented a sharding strategy to improve the intricate interactions between VSPs and a collection of Mobile Users (MUs) in a blockchain-based Metaverse architecture. Based on Stackelberg game theory, they created an incentive system that considered the MUs' contributions to the Metaverse's computational capacity. Researchers examined the interaction between VSPs and network infrastructure providers in [13], taking into account the usefulness of the VSPs based on the requirements of MUs for Quality of Experience (QoE). Compared to traditional schemes, their suggested one showed improvement in delivering QoE with constrained resources.

None of the existing works considers the resource allocation and efficient pricing between the VSPs and ESPs in an edge-assisted Metaverse system, which is considered in this work.

## III. System Model

Let us consider there exists $\mathcal{M} = \{1, ..., m, ..., M\}$ ESPs and $\mathcal{N} = \{1, ..., n, ..., N\}$ VSPs in a Metaverse system to provide services to end users. We assume that each VSP has its own separate non-overlapping set of end users and they are considered as one entity $E$ of a VSP, so the end users subscribed to VSP $n$ is $E_n$.

### A. High- and Low-Priority Jobs

The $J_n$ jobs assigned to VSP $n$ by end users $E_n$ can be divided into two categories *i.e., high-priority* and *low-priority*.

*1) High-Priority Jobs:* These jobs have a tight deadline, meaning they must be done within a certain amount of time. The rate end users need to pay to VSPs for the service and the price VSPs need to pay ESPs for the resources for high-priority jobs is higher than that of low-priority. If $J_n^H$ and $\alpha_n^H$ denote the amount of high-priority jobs and part of the total jobs that are high-priority, respectively, then we can write,

$$\alpha_n^H = \frac{J_n^H}{J_n} \tag{1}$$

*2) Low-Priority Jobs:* In contrast, low-priority jobs have a soft deadline, meaning they can take longer to complete. So, the rate end users need to pay to VSPs for the service and the price VSPs need to pay ESPs for the resources is lower. If $J_n^L$ and $\alpha_n^L$ denote the amount of low-priority jobs and part of the total low-priority jobs, respectively, then we can write,

$$\alpha_n^L = \frac{J_n^L}{J_n} \tag{2}$$

Based on Eqs. (1) and (2) of high- and low-priority jobs, we can write $J_n^H + J_n^L = J_n$ and $\alpha_n^H + \alpha_n^L = 1$.

TABLE I: Notations and Definitions

| Notation | Definition |
|---|---|
| $\alpha^H$ | Part of jobs that are high-priority jobs |
| $\alpha^L$ | Part of jobs that are low-priority jobs |
| $S$ | Fixed subscription rate end users $E$ pays VSP |
| $r^H$ | Variable rate end users $E$ pays VSP for high-priority jobs |
| $r^L$ | Variable rate end users $E$ pays VSP for low-priority jobs |
| $T^H$ | Time restriction for VSP to complete high-priority jobs |
| $T^L$ | Time restriction for VSP to complete low-priority jobs |
| $\lambda^H$ | Arrival rate of high-priority job requests at VSP |
| $\lambda^L$ | Arrival rate of low-priority job requests at VSP |
| $\mu^H$ | Servicing rate of VSP for high-priority job requests |
| $\mu^L$ | Servicing rate of VSP for low-priority job requests |
| $\mu^T$ | Total servicing rate of VSP for all job requests |
| $\omega_{nm}$ | Probability of VSP $n$ choosing ESP $m$ |
| $x_{nm}$ | Resource allocated to VSP $n$ by ESP $m$ |
| $p_{nm}$ | Price VSP $n$ pays ESP $m$ for resource allocation |
| $\mathbb{P}$ | Overall profit of VSP for completing jobs |
| $c$ | Cost ESP pay for operations and maintenance |
| $G$ | Maximum resource capacity of ESP |

### B. M/M/1 Queuing Model

Assuming the use of the M/M/1 queuing model at the VSPs, each VSP can be modelled as a single server, with incoming jobs from their respective end users represented as the queue. The arrival rate of job requests from end users $E_n$ into VSP $n$ that implements the M/M/1 queue follows a Poisson distribution, denoted as $\lambda_n$. Specifically, we denote the arrival rate of requests into VSP $n$ as $\lambda_n^H$ for high-priority jobs and $\lambda_n^L$ for low-priority jobs. The service rate of VSP $n$ to deliver services to end users based on the requests follows an exponential distribution, denoted as $\mu_n$.

The time restriction imposed by the end users to get resources from VSP $n$ to complete high- and low-priority jobs, $T_n^H$ and $T_n^L$ respectively, can be calculated using Queuing theory as follows.

$$T_n^H = \frac{1}{\mu_n^H - \lambda_n^H} \tag{3a}$$

$$\mu_n^H = \frac{1 + T_n^H \lambda_n^H}{T_n^H} \tag{3b}$$

Similarly, we derive the following for the low-priority jobs:

$$\mu_n^L = \frac{1 + T_n^L \lambda_n^L}{T_n^L} \tag{4a}$$

The service rate of the VSPs need to be larger than the rate at which the job requests come in to deliver the Metaverse services to the end user.

$$\mu_n^T \geq \mu_n^H + \mu_n^L \tag{5a}$$

### C. End Users Payment to VSP for Services

End users incur a fixed subscription rate, $S$, regardless of the amount or type of jobs they request. So, end users $E_n$ need to pay VSP $n$ a fixed amount of $S_n$. Aside from the subscription rate, VSP $n$ receive from end users $E_n$ for high-priority jobs and low-priority jobs for a unit of service to be $r_n^H$ and $r_n^L$, respectively, such as $r_n^H > r_n^L$. As there is a need to move high-priority jobs up the queue and service them within a shorter, stricter deadline, the rate end users pay for them will be higher. Therefore, the total amount that VSP $n$ received from end users $E_n$ is,

$$R_n = S_n + r_n^H \mu_n^H + r_n^L \mu_n^L \tag{6}$$

### D. Probability of VSP choosing ESP

The VSPs can choose the ESPs to get better service. The probability of VSP $n$ requesting resource allocation from ESP $m$ is denoted by $\omega_{nm}$ and takes into consideration two parameters of ESP $m$, price competitiveness $PC_{nm}$ and reputation $\mathbb{R}_{nm}$. Explicitly, VSP $n$ is more likely to choose ESP $m$ amongst all other ESPs to get its service from if the price ESP $m$ set for the service is less. The price competitiveness of ESP $m$ compared to all other ESPs using which VSP $n$ considers which ESP to get its service from is calculated by:

$$PC_{nm} = 1 - \frac{p_{nm}}{\Sigma_{k=1}^{M} p_{nk}} \tag{7}$$

Again, we adapt a reputation-based incentive mechanism where VSP $n$ is considering which ESP it should get its service from. We assume the quality of service such as quick processing of data, high sampling rate, and sufficient bandwidth for communication provided by ESP $m$ to VSP $n$ as $\phi_{nm}$ while the total required resources VSP $n$ expected is $\Phi_n$. Please note that any and more parameters can be considered for calculating the reputation, which is basically implementation-based. The reputation of ESP $m$ judged by VSP $n$ is calculated as follows,

$$RP_{nm} = \frac{\phi_{nm}}{\Phi_{nm}} \tag{8}$$

As resource allocation supply and demand are dynamic end ever-changing. The VSP does not calculate and fix a certain reputation $RP_{nm}$ of an ESP indefinitely. Instead, it will periodically calculate a new $RP_{nm}$ with updated values to keep itself up-to-date. However, it does not simply ignore the previous reputation values and instead takes them into consideration. Each of the VSPs keeps a weight parameter $\beta$ that determines the importance of the historical reputation versus the current reputation using which it calculates the running reputation $\mathbb{R}$. VSP $n$ with weight parameter $\beta_n$ will calculate the running capability of ESP $m$ as:

$$\mathbb{R}_{nm}(t) = \beta_n \mathbb{R}_{nm}(t-1) + (1-\beta_n) RP_{nm}(t) \tag{9}$$

In the above equation, $\mathbb{R}_{nm}(t)$ represents the running ranking of ESP $m$ at time $t$. Considering the weight parameter $\beta_n$, each VSP $n$ will each decide separately this value which determines whether the $\mathbb{R}$ value should consider more of responsiveness or stability in choosing an ESP.

For an ESP to be preferred by VSPs and receive more income, said ESP need to maintain not only a comparatively low price but also a good reputation amongst all other ESPs from which the VSPs choose to get its resource. We denote this relative reputation of ESP $m$ in VSP $n$ POV as the ranking $RK_{nm}$ which is calculated as:

$$RK_{nm} = \frac{\mathbb{R}_{nm}}{\Sigma_{k=1}^{M}\mathbb{R}_{nk}} \tag{10}$$

After calculating the price competitiveness $PC_{nm}$ using (6) and the reputation $RC_{nm}$ for every candidate ESP, VSP $n$ will then come up with a set of probabilities of choosing each ESP $\Omega_n = \{\omega_{n1}, ..., \omega_{nm}, ...\omega_{nM}\}$ where each probability $\omega_{nk}$ is calculated using a weighted sum of the price competitiveness and ranking as:

$$\omega_{nk} = \nu_1 \times PC_{nk} + \nu_2 \times RK_{nk} \tag{11a}$$

$$\nu_1 + \nu_2 = 1 \tag{11b}$$

Again, each VSP $n$ needs to decide on its own which of the two considerations is more significant to the decision of choosing an ESP. If both are equally important, VSP $n$ can set both $\nu_1$ and $\nu_2$ as 0.5 in Eq. (11a).

### *E. VSP Payment to ESPs for Resource Allocation*

As VSPs receive the resources from ESPs, VSPs need to pay a certain amount of price to ESPs based on the amount of allocated resources. Note that a single VSP does not get all of it resources allocated by a single ESP, instead, it gets the necessary resources from many different ESPs. Thus, VSP needs to pay all of these ESPs for the corresponding amount of resources it gets from them. VSP $n$ gets $x_{nm}$ amount of resources from ESP $m$ and in return, it will pay $p_{nm}^H$ and $p_{nm}^L$ for high and low priority jobs for a unit of service to ESP $m$. So to all the ESPs, VSP $n$ will be paying a total of:

$$P_n = \Sigma_{k=1}^{M}\omega_{nk}x_{nk}(zp_{nm}^H + (1-z)p_{nm}^L) \tag{12}$$

where, $z = 1$, when $x_{nk}$ is a high priority job, otherwise, $z = 0$.

### *F. Profit and Cost of ESPs*

While VSPs make the decision to choose which ESP to get their resources allocated from, ESPs also have their own considerations to make in choosing the VSP. For a specific ESP $m$ with a capacity of $G_m$, we have a constraint for it to allocate resources to all VSPs as:

$$\Sigma_{j=1}^{N}\omega_{jm}x_{jm} \leq G_m \tag{13}$$

As it receives payment from VSPs for resource allocation, ESPs also need to spend an amount of money for operational and maintenance costs $c$ for them. Each different ESP may need to pay different costs depending on the vendors that are involved in their daily operations. We denote this operational and maintenance cost for ESP $m$ per unit of resource as $c_m$, and so, the total cost ESP $m$ needs to pay based on the resources it allocates is calculated by:

$$C_m = c_m\Sigma_{j=1}^{N}\omega_{jm}x_{jm}\phi_{jm} \tag{14}$$

where, $C_m$ is increased with the increasing value of $\phi_{nm}$ i.e., quality of service. So, the overall profit that ESP $m$ gets is,

$$\mathbb{Q}_m = \Sigma_{j=1}^{N}\omega_{jm}x_{jm}(p_{jm} - c_m\phi_{jm}) \tag{15}$$

## IV. Dynamic Resource Allocation and Pricing (DRAP) Framework

The challenge is to allocate resources in a way that maximizes the overall utility of all VSPs, provides the best possible service to end-users, and minimizes the cost and energy consumption of the computing infrastructure of the ESPs. For such efficient allocation of computing resources to high- and low-priority jobs based on the demands of end-users, the utility functions of VSPs and ESPs are discussed below.

The utility functions of VSP $n$ in regards to high- and low-priority jobs to find the optimized maximum revenue with subject to a few constraints is as follows.

$$\begin{aligned} \max_{U_n} \quad & S_n + \Sigma_{k=1}^{M}\omega_{nk}x_{nk}(z(r_n^H - p_{nm}^H) + (1-z)(r_n^L - p_{nm}^L)) \\ s.t. \quad & r_n^H + r_n^L > p_{nm}^H + p_{nm}^L \\ & \Sigma_{k=1}^{M}\omega_{nk}x_{nk} < \mu_n^T \end{aligned}$$

Similarly, the utility function of ESP $m$ is,

$$\begin{aligned} \max_{U_m} \quad & \Sigma_{j=1}^{N}\omega_{jm}x_{jm}(p_{jm} - c_m\phi_{jm}) \\ s.t. \quad & \Sigma_{j=1}^{N}\omega_{jm}x_{jm} \leq G_m \\ & p_{jm} > c_m\phi_{jm} \end{aligned}$$

## V. Stackelberg Gaming Model

We leverage the Stackelberg gaming model to find the optimal values for our framework, where the VSPs act as leaders and ESPs as followers. At first, the VSPs announce their requirement to all the ESPs and the ESPs find their optimal resource allocation strategies using the announced information by the VSPs and announce back again to all the VSPs. Next, the VSPs update their optimal resource allocation strategy using the information received from ESPs and subsequently announce back to the ESPs. This exchanging of information continues til the VSPs do not change their profits. Algorithm 1 shows the above-mentioned steps in detail.

## VI. Numerical Analysis

We carry out a performance study to evaluate the effectiveness of the proposed scheme DRPA in terms of resource allocation and profit earned. We also compare its performance with the Greedy approach.

**Algorithm 1:** DRAP Framework

Set the variable $changedProfit \leftarrow$ True
VSPs announce their high- and low-priority job requirements and their respective prices.
**while** $changedProfit$ **do**
  **for** each $ESP$ in $ESPs$ **do**
    Find optimal values for $ESP$ and announce them.
  **end for**
  **for** each $VSP$ in $VSPs$ **do**
    Find optimal values for $VSP$ and announce them.
  **end for**
  Calculate the new profits for VSPs.
  **if** new profits are the same as previous profits **then**
    $changedProfit \leftarrow$ False
  **end if**
**end while**
Final optimal values for VSPs and ESPs.

### A. Initialization

We initialize the relevant parameters of the 4 VSPs and 4 ESPs with the values provided in Tables II and III respectively. To model realistic behaviour, we consider below points:

1) We set $VSP3$ to have a lower resource price compared to the other VSPs, which will result in ESPs allocating fewer resources to it due to lower profit margins.
2) We set one $ESP4$ to have a lower reputation score compared to the other ESPs, indicating poorer performance, which discourages VSPs from requesting resource allocation from it.

TABLE II: Leader VSPs' Initial Input

| Leaders | Subscription rate ($) | High-Priority Jobs | | Low-Priority Jobs | | Price ($/RU) | | | | Reputation | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Rate ($/RU) | Resource required (RU) | Rate ($/RU) | Resource required (RU) | $ESP1$ | $ESP2$ | $ESP3$ | $ESP4$ | $ESP1$ | $ESP2$ | $ESP3$ | $ESP4$ |
| $VSP1$ | 120 | 15 | 10 | 12 | 20 | 9 | 8 | 10 | 5 | 0.6 | 0.7 | 1 | 0.1 |
| $VSP2$ | 100 | 14 | 15 | 13 | 20 | 5 | 5 | 12 | 6 | 0.75 | 0.75 | 0.8 | 0.2 |
| $VSP3$ | 75 | 20 | 25 | 16 | 20 | 2 | 3 | 4 | 3 | 0.9 | 0.8 | 1 | 0.1 |
| $VSP4$ | 120 | 18 | 15 | 14 | 20 | 8 | 8 | 10 | 7 | 0.8 | 0.8 | 1 | 0.2 |

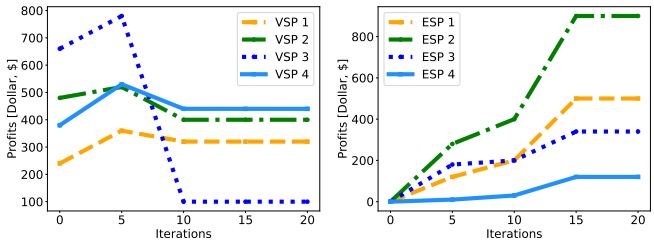
TABLE III: Follower ESPs' Initial Input

| Followers | Capacity (RU) | Price ($/RU) | | | | Cost ($/RU) |
|---|---|---|---|---|---|---|
| | | $VSP1$ | $VSP2$ | $VSP3$ | $VSP4$ | |
| $ESP1$ | 40 | 9 | 5 | 2 | 8 | 1.5 |
| $ESP2$ | 50 | 8 | 5 | 3 | 8 | 1 |
| $ESP3$ | 45 | 10 | 12 | 4 | 10 | 0.75 |
| $ESP4$ | 40 | 5 | 6 | 3 | 7 | 0.5 |

### B. Implementation and Results

We implement the leader-follower Stackelberg game on MATLAB with the utilization functions of VSP and ESP described in Section IV to obtain the optimal resource allocation. We analyze the performance of the proposed DRAP by evaluating the value of the utility functions of VSPs and ESPs over iterations to find the final resource allocation (*i.e.,* which VSP receive resource allocated by which ESP and respective allocation prices) after the game has reached an equilibrium.

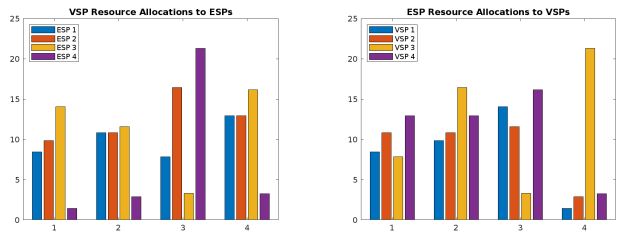
Figs. 1a and 1b show the profit of all the VSPs and ESPs against the iterations. The Stackelberg equilibrium refers to the points after which profits of the VSPs and ESPs do not




(a) Leaders' profit. (b) Follwers' profit.

Fig. 1: Leaders and Followers' profit over the iterations

change *i.e.,* where the leaders' decisions are optimal, given that the followers respond to their decisions in the best way possible. Analyzing further Fig. 1a, we are able to get more insight to what is happening. We observe the trend-line of $VSP3$ in blue that has a continuous dip in profits up until it reaches the optimum point at iteration 10. To explain this, we recall that $VSP3$ is the VSP that has bad initial price point for ESPs as compared to the other VSPs. At first, we can see that the profit of $VSP3$ is very high because the earning it gets from the end users is much higher than what it is spending to pay the ESPs. However, this results in ESPs being reluctant to allocate resources to $VSP3$. Not getting enough resources, $VSP3$ has to try to increase its price point to appeal to the ESPs to allocate it resources. This causes the gap between the earning and the expense, *i.e.,* the net profit, to decrease as the iterations go. On the other hand, we also observe in more detail Fig. 1b that shows the profit of the ESPs over the iteration. ESPs 1, 2, and 3 steadily increase over the iterations as it allocates resources to the requesting VSPs. The profit of $ESP4$ however is grounded at $0 until iteration 9. This is because, with its low reputation ranking, VSPs avoid requesting resources from it as they deem it to be not reliable unless there is no other choice for the VSP but to choose it to fulfil the end users' requirements. With the price point to be paid to $ESP3$ being the highest, we expect that it will have the largest profit, but that is not the case. This high price point is not attractive to the VSPs, resulting in the VSPs not choosing $ESP3$ as the provider of the resources.

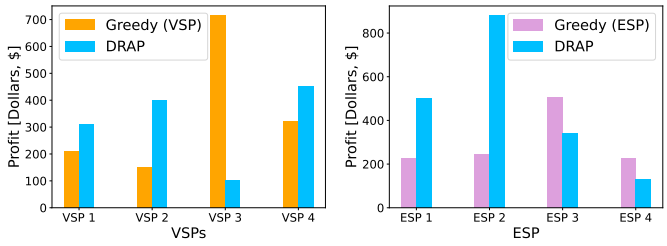



(a) VSPs' RA Distribution. (b) ESPs' RA Distribution.

Fig. 2: Resource Allocation (RA) Distribution from VSP to ESP and vice versa.

Figs. 2a and 2b show the resource allocation to each VSP from different ESPs. From Fig. 2a and Table II, we can

observe that the final allocation to each VSP, regardless of the allocating ESP, is more than the required resource which aligns with the intention of VSPs to get enough resources to service the end users. We can also see in Fig. 2b that VSPs are reluctant to request $ESP4$ for resource allocation due to its bad reputation as compared to the other ESPs. The very small amount that VSPs 1, 2, and 4 request from $ESP4$ is very likely due to the other ESPs reaching their capacity and not being able to allocate more resources. This excludes $VSP3$ whose price point to be paid to the ESPs is lower as compared to the other VSPs, which leads to no resource allocation from the other high-reputation ESPs leading to $VSP3$ having to resort to low-reputation $ESP4$ to avoid not being able to fulfil the required services for its end users.




(a) (b)

Fig. 3: Comparison of DRAP with greedy approach.

In Fig 3, we compare DRAP with the greedy approach – Greedy (VSP) denotes VSPs give the highest priority to the ESPs that have the highest reputations and in this case, ESPs do not have control over resource allocation. On the other hand, Greedy (ESP) denotes ESPs are greedy about their profit and VSPs stay idle. In Figs. 3a and 3b, we can see that using DRAP, all the VSPs can earn more profit compared to the Greedy (VSP) approach and all the ESPs can earn more compared to Greedy (ESP) approach, respectively. So, we can claim that DRAP satisfies both the VSPs and ESPs expectations in terms of profit while efficiently allocating resources.

## VII. Conclusion

In this work, we investigated the problem of resource allocation between VSPs and ESPs in a Metaverse system. For this, we proposed the (DRAP) framework, which allocates resources based on the QoS provided by the ESPs and type of services *i.e., high-* and *low-priority* requested by the VSPs. Furthermore, we varied the pricing for resource allocation over time-based on the reputation of the ESPs. In brief, ESPs get incentives *i.e.,* more service offers from VSPs, if they offer better service. In order to determine the optimal profit for both VSPs and ESPs, we used a multi-leader and -follower based Stackelberg gaming approach, with VSPs as leaders and ESPs as followers. We carried out numerical analysis to simulate real-life Metaverse system using which we were able to determine the optimal solutions for the leaders and followers. The optimal solutions show that the proposed scheme DRAP is able to reach a stable point of resource allocation that meets the requirements of VSPs, and ultimately the end users, while also satisfying both the VSPs and ESPs' expectations in terms of profit.

## Acknowledgment

This work is supported by the National University of Singapore under MoE AcRF Tier-2 Grant, NUS WBS No. A00094520100 (MoE T2EP50120-0021)

## References

[1] L. Lee, T. Braud, P. Zhou, L. Wang, D. Xu, Z. Lin, A. Kumar, C. Bermejo, and P. Hui, "All One Needs to Know about Metaverse: A Complete Survey on Technological Singularity, Virtual Ecosystem, and Research Agenda," *arXiv preprint arXiv:2110.05352*, 2021. [Online]. Available: https://arxiv.org/abs/2110.05352

[2] M. Qu, Y. Sun, and Y. Feng, "Digital Media and VR Art Creation for Metaverse," in *2nd Asia Conference on Information Engineering (ACIE)*, 2022, pp. 48–51.

[3] J. D. N. Dionisio, W. G. B. III, and R. Gilbert, "3D Virtual Worlds and the Metaverse: Current Status and Future Possibilities," *ACM Computing Surveys*, vol. 45, no. 3, jul 2013.

[4] Steve Kovach, "Next for the metaverse: convincing you it's not just for kids." Accessed on Aug. 25, 2022. [Online]. Available: https://www.cnbc.com/2021/12/22/here-are-the-companies-building-the-metaverse-meta-roblox-epic.html

[5] S.-M. Park and Y.-G. Kim, "A Metaverse: Taxonomy, Components, Applications, and Open Challenges," *IEEE Access*, vol. 10, pp. 4209–4251, 2022.

[6] H. Ning, H. Wang, Y. Lin, W. Wang, S. Dhelim, F. Farha, J. Ding, and M. Daneshmand, "A Survey on Metaverse: the State-of-the-art, Technologies, Applications, and Challenges," *arXiv preprint arXiv:2111.09673*, 2021.

[7] W. Y. B. Lim, Z. Xiong, D. Niyato, X. Cao, C. Miao, S. Sun, and Q. Yang, "Realizing the Metaverse with Edge Intelligence: A Match Made in Heaven," *IEEE Wireless Communications*, pp. 1–9, 2022.

[8] L. U. Khan, Z. Han, D. Niyato, E. Hossain, and C. S. Hong, "Metaverse for Wireless Systems: Vision, Enablers, Architecture, and Future Directions," 2022. [Online]. Available: https://arxiv.org/abs/2207.00413

[9] W. C. Ng, W. Yang Bryan Lim, J. S. Ng, Z. Xiong, D. Niyato, and C. Miao, "Unified Resource Allocation Framework for the Edge Intelligence-Enabled Metaverse," in *IEEE International Conference on Communications*, 2022, pp. 5214–5219.

[10] Y. Han, D. Niyato, C. Leung, D. I. Kim, K. Zhu, S. Feng, S. X. Shen, and C. Miao, "A Dynamic Hierarchical Framework for IoT-assisted Digital Twin Synchronization in the Metaverse," *IEEE Internet of Things Journal*, pp. 1–1, 2022.

[11] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, and E. Dutkiewicz, "MetaChain: A Novel Blockchain-based Framework for Metaverse Applications," in *IEEE Vehicular Technology Conference*, 2022.

[12] X. Huang, W. Zhong, J. Nie, Q. Hu, Z. Xiong, J. Kang, and T. Q. S. Quek, "Joint User Association and Resource Pricing for Metaverse: Distributed and Centralized Approaches," 2022. [Online]. Available: https://arxiv.org/abs/2208.06770

[13] H. Du, J. Liu, D. Niyato, J. Kang, Z. Xiong, J. Zhang, and D. I. Kim, "Attention-aware Resource Allocation and QoE Analysis for Metaverse xURLLC Services," 2022. [Online]. Available: https://arxiv.org/abs/2208.05438

[14] Y. Han, D. Niyato, C. Leung, C. Miao, and D. I. Kim, "A Dynamic Resource Allocation Framework for Synchronizing Metaverse with IoT Service and Data," in *IEEE International Conference on Communications*, 2022, pp. 1196–1201.

[15] A. Kalita and M. Gurusamy, "D2R2: Distributed and Dynamic Reputation based Resource Allocation in Metaverse," 11 2022. [Online]. Available: 10.36227/techrxiv.21395316.v1