UAV-assisted Wireless Power Charging for Efficient Hybrid Coded Edge Computing Network

Jer Shyuan Ng[1,2], Wei Chong Ng[1,2], Wei Yang Bryan Lim[1,2],
Zehui Xiong[3], Dusit Niyato[4], Cyril Leung[5,6] and Chunyan Miao[2,4,5]
[1]Alibaba Group [2]Alibaba-NTU JRI [3]SUTD, Singapore
[4]SCSE, NTU, Singapore [5]LILY Research Center, NTU, Singapore [6]ECE, UBC, Canada

ICC 2022 - IEEE International Conference on Communications | 978-1-5386-8347-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICC45855.2022.9838931

***Abstract*—With the ubiquitous sensing enabled by the Internet-of-Things (IoT), massive amount of data is generated every second, transforming the way we interact with the world. To manage big data and enable analytics at the edge of the network, large amount of computation power is required to perform the computation intensive tasks. However, the energy-constrained IoT devices are not able to perform the computation tasks without compromising the quality-of-service of the applications. In this paper, we propose a hybrid network in which users can offload their computation tasks to edge servers through coded edge offloading or perform local computation with the wireless power transfer derived from coalitions of unmanned aerial vehicles (UAVs) serving as mobile charging stations. We consider a two-level optimization approach where an optimal UAV coalitional structure that minimizes the network cost is formed. In the performance evaluation, we provide extensive sensitivity analyses to study the performance of the cost minimization approach amid varying network parameters.**

***Index Terms*—Edge computing, coded computation, unmanned aerial vehicles, stochastic integer programming, coalition**

I. Introduction

The proliferation of the increasingly capable and interconnected Internet-of-Things (IoT) devices generates enormous amounts of data that drive the development of smart applications by using Artificial Intelligence (AI) techniques. The data-driven AI models rely heavily on the large quantities of training data, which in turn requires large amounts of computation power. Due to the energy constraints, the IoT devices face challenges in meeting the required quality-of-service (QoS) and satisfying the complex demands of the consumers, especially for latency-sensitive applications.

To enable the IoT devices to perform the computation-intensive tasks while meeting the required QoS, unmanned aerial vehicles (UAVs) are increasingly used as mobile charging stations to provide wireless power charging so as to guarantee the continuity of services. However, there are various factors that can lead to the failure of the UAVs in performing their operations. For example, the UAVs may not able to take off from their depots due to motor or hardware failure. Even when the UAVs are able to take off, their engines may fail due to the adverse weather conditions, i.e., strong wind or heavy rain [1]. Besides, there may be malicious UAVs that cause disruptions to the infrastructure, where unintended operations are performed, hence threatening the performance of the UAV network [2]. Specifically to the UAVs as mobile charging stations, the variability in the efficiency of magnetic coupling resonant causes uncertainties in the actual amount of charging energy to the IoT devices, which may lead to the failure of the UAVs in providing the energy required by the IoT devices.

Apart from relying solely on UAV enabled wireless power charging, which may be prone to the aforementioned disruptions, the resource-constrained IoT devices can also offload their computation tasks to the edge servers. Offloading computation-intensive tasks from the IoT devices to the edge servers helps to enhance the performance of the IoT devices while minimizing the on-board energy usage [3]. The IoT devices can leverage the resources of multiple edge servers to perform the distributed computation tasks collaboratively. In order to mitigate the straggler effects in the distributed computing network where the computation tasks are often delayed by the slower edge servers, coding techniques [4] can be used to split the computation tasks into subtasks, which are in turn allocated to the edge servers.

In this paper, we consider a hybrid network in which IoT devices, i.e., users, can perform their computation tasks using the following approaches: i) *Full local computation:* Each cell of users is supported by a coalition of UAVs which powers the users through wireless power transfer, thereby enabling the users to complete the computation tasks, ii) *Full coded offloading:* Each user utilizes coding techniques to allocate the computation subtasks to edge servers for computation, thereby completing their task while reducing the computation latency, and iii) *Hybrid approach:* Each user partially offloads while simultaneously performing local computation, e.g., due to energy constraints or the unreliability of UAV wireless power transfer. To reduce the cost of the network, it is crucial to determine the computation approach adopted by each user. In addition, the coalition of UAV which minimizes the network cost also has to be derived. To this end, we consider a two-level optimization approach:

- *Upper level stochastic offloading optimization:* We derive the computation decision of each user, subjected to constraints of computation capacities of the UAV coalitions and edge servers. Moreover, to account for the stochastic nature of wireless power transfer reliability, we formulate the computation decision problem into a two-stage stochastic integer programming (SIP) model.
- *Lower level UAV coalition formation:* Since multiple UAVs can work together, e.g., as UAV swarms or coalitions, we adopt the coalition formation game to model the cooperation among the UAVs. Then, we derive the optimal coalitional structure that can derive the minimum

cost in the network.

II. System Model

We consider a UAV-assisted wireless power charging enabled coded edge computing network with a coverage area that is divided into I cells, represented by a set $\mathcal{I} = \{1, \ldots, i, \ldots, I\}$. Each cell i contains W_i IoT devices, i.e., users, where the total number of users in each cell i may be different. The term d_{iw} denotes the w^{th} user in cell i. The users aim to perform machine learning computation tasks using the data collected from their surroundings. Given the large amounts of data collected, the users may not have sufficient computation resources, e.g., energy, to complete the machine learning tasks individually.

In general, the hybrid network enables users to complete the computation tasks using the below approaches:

1) *Full local computation:* The I cells are supported by J UAVs, the set of which is denoted by $\mathcal{J} = \{1, \ldots, j, \ldots, J\}$ which can power the users through wireless power transfer. The UAVs can form swarms, i.e., coalitions, that are capable of flexible deployment, thereby allowing the users to have sufficient computation power to complete the machine learning tasks locally. While UAV coalitions are able to transfer their power to the users in a wireless manner, they may not always be reliable due to the variability in the efficiency of the magnetic coupling resonant. More specifically, the fluctuation of resonance frequency affects the efficiency of power transfer from the UAVs to the users [5].
2) *Full coded offloading:* The I cells are supported by K edge servers, e.g., micro base stations, which are represented by a set $\mathcal{K} = \{1, \ldots, k, \ldots, K\}$, that provides coded computation offloading support for the users. Specifically, the users use coding techniques, e.g., Polynomial codes [6], to allocate the distributed computation subtasks to the edge servers that have sufficient resources. The use of coding techniques minimizes the computation latency by reducing the recovery threshold, which is defined as the number of edge servers that need to return their computed results for the users to reconstruct the final result. While the edge servers are more expensive to employ than the UAVs, the edge servers have larger computation capacities and are more reliable.
3) *Hybrid approach:* The users may adopt a combined approach in some cases, e.g., adopt wireless charging for *partial* local computation while simultaneously offloading the subtasks partially to the edge servers.

A. UAV Charging Model

To provide wireless power transfer, each UAV j has a maximum energy of e_j^{max} and is equipped with a wireless power charger that has a charging power of p_j^c. The UAVs spend their energy mainly in three activities: (i) flying, (ii) hovering, and (iii) charging. To ensure that the UAVs have sufficient energy to return to their bases after transferring their power to the users, the maximum charging time of UAV j for cell i, which is denoted as t_{ij}, can be calculated based on (1) as follows:

$$e_j^{max} \geq \frac{2d_{ij}}{v} p_j^f + t_{ij}(p_j^c + p_j^h), \tag{1}$$

where v is the travelling speed of the UAVs, which is assumed to be the same for all UAV $j \in \mathcal{J}$, p_j^f and p_j^h are the flying and hovering energies of UAV j respectively. $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ refers to the distance between UAV j and cell i, where the 3D-coordinates of UAV j and cell i are represented by $\{x_j, y_j, h\}$ and $\{x_i, y_i, 0\}$ respectively and h is the height of the UAV, which is assumed to be the same for all UAVs.

The power of the UAVs is transferred to the users through magnetic resonant coupling technique which allows high transmit efficiency over a relatively large distance. However, the performance of the UAVs may be hindered by the uncertainty in the efficiency of the magnetic coupling resonant, which in turn affects the amount of energy that can be gained by the users. There are several factors that affect the efficiency of the magnetic coupling resonant such as the change in resonance frequency and the load value of the users [5]. In this paper, we do not consider the effects of each individual factors. Without loss of generality, we denote the efficiency of the magnetic coupling resonant when UAV j charges user i by β_j^i. The values of β_j^i, $\forall i \in \mathcal{I}$ and $\forall j \in \mathcal{J}$, are between 0 and 1. For example, $\beta_j^i = 0.4$ indicates that 40% of the charging power of UAV j can be transferred efficiently to user i.

The efficiency of the magnetic coupling resonant cannot be known precisely before wireless charging is completed. In a large-scale heterogeneous network, there may exist several *scenarios* of varying combinations of charging efficiencies across the different UAVs. Specifically, each scenario λ_y refers to a particular scenario indexed y, and is in turn represented by a matrix of charging efficiencies $\beta_j^i(\lambda_y)$, $\forall i \in \mathcal{I}$ and $\forall j \in \mathcal{J}$. This is expressed as follows:

$$\lambda_y = \begin{bmatrix} \beta_1^1(\lambda_y) & \beta_1^2(\lambda_y) & \cdots & \beta_1^I(\lambda_y) \\ \beta_2^1(\lambda_y) & \beta_2^2(\lambda_y) & \cdots & \beta_2^I(\lambda_y) \\ \vdots & \vdots & \ddots & \vdots \\ \beta_J^1(\lambda_y) & \beta_J^2(\lambda_y) & \cdots & \beta_J^I(\lambda_y) \end{bmatrix}. \tag{2}$$

The set of scenarios are represented by $\Lambda = \{\lambda_1, \ldots, \lambda_y, \ldots, \lambda_Y\}$ where Y is the total number of possible scenarios. The probability that scenario λ_y happens is denoted as $P(\lambda_y)$. For the rest of the paper, we use $\beta_j^i(\lambda_y)$ and β_j^i interchangeably where a confusion does not arise from dropping the (λ_y) notation. Accordingly, the available power to be gained by cell i from UAV j, which is denoted as p_{ij}^v, is expressed as follows:

$$p_{ij}^v = \beta_j^i p_j^c. \tag{3}$$

The actual charging energy of cell i by UAV j, which is represented by e_{ij}, depends on the maximum storage energy

e_i^{max} and remaining energy e_i^r of cell i. Specifically, e_{ij} is expressed as follows:

$$e_{ij} = \begin{cases} p_{ij}^v t_{ij} & \text{if } e_i^{max} - e_i^r > p_{ij}^v t_{ij}, \\ e_i^{max} - e_i^r & \text{if } e_i^{max} - e_i^r \leq p_{ij}^v t_{ij}. \end{cases} \quad (4)$$

For the first case, cell i is not fully charged even after the UAV transfers all the possible power to cell i. For the second case, the cell is fully charged. This happens when the UAV has sufficient power to be transferred to the cell until it is fully charged. In other words, the UAV stops charging before it utilizes all its charging power.

B. Coded Distributed Computing Model

The users can also partially or completely offload their subtasks to the edge servers, where the resources of multiple edge servers are leveraged to complete the distributed computation tasks collaboratively. In order to mitigate the straggler effects, coding techniques are used to split the computation tasks into smaller tasks, i.e., subtasks, and distribute them to the edge servers for computation. The objective of the coding schemes is to minimize the recovery threshold α_{iw}, which is defined as the number of edge servers that are required to return their computed results to the users in order to reconstruct the final result.

In this paper, we consider that each of the user d_{iw} adopts the Polynomial codes [6] to perform the distributed matrix multiplication computations, i.e., $\mathbf{C}^{iw} = \mathbf{A}^{iw\top}\mathbf{B}^{iw}$ where $\mathbf{A}^{iw}$ and $\mathbf{B}^{iw}$ are input matrices of user d_{iw}, $\mathbf{A}^{iw} \in \mathbb{F}_q^{s\times r}$ and $\mathbf{B}^{iw} \in \mathbb{F}_q^{s\times t}$ for integers s, r, and t and a sufficiently large finite field $\mathbb{F}_q$. In order to perform the coded distributed computation by using the Polynomial codes, there are four important steps:

1) *Distribution of computation tasks:* Given that the edge servers are able to store up to $\frac{1}{m_{iw}}$ and $\frac{1}{n_{iw}}$ fractions of matrices $\mathbf{A}^{iw}$ and $\mathbf{B}^{iw}$ respectively, user d_{iw} divides the input matrices into submatrices $\tilde{\mathbf{A}}_k^{iw} = f_k(\mathbf{A}^{iw})$ and $\tilde{\mathbf{B}}_k^{iw} = g_k(\mathbf{B}^{iw})$, where $\tilde{\mathbf{A}}_k^{iw} \in \mathbb{F}_q^{s\times \frac{r}{m_{iw}}}$ and $\tilde{\mathbf{B}}_k^{iw} \in \mathbb{F}_q^{t\times \frac{r}{n_{iw}}}$ respectively. Specifically, $\mathbf{f}$ and $\mathbf{g}$ represent the vectors of functions such that $\mathbf{f} = (f_1, \ldots, f_k, \ldots, f_K)$ and $\mathbf{g} = (g_1, \ldots, g_k, \ldots, g_K)$, respectively. Then, the user distributes the submatrices to the edge servers over the wireless channels for computations.
2) *Computation by the edge servers:* Each edge server k is allocated submatrices $\tilde{\mathbf{A}}_k^{iw}$ and $\tilde{\mathbf{B}}_k^{iw}$ by the user. Based on the allocated submatrices, the edge servers perform the matrix multiplication, i.e., $\tilde{\mathbf{A}}_k^{iw\top}\tilde{\mathbf{B}}_k^{iw}$.
3) *Transmission of computed results:* Upon completion of the local computations, each edge server transmits its computed results, i.e., $\tilde{\mathbf{C}}_k^{iw} = \tilde{\mathbf{A}}_k^{iw\top}\tilde{\mathbf{B}}_k^{iw}$ to the user over the wireless communication channels.
4) *Reconstruction of final result:* By using the Polynomial codes, the user is able to reconstruct the final result upon receiving α_{iw} computed results by using decoding functions. In other words, although the computation task is split and distributed to more than α_{iw} edge servers, the user does not need to wait for all the edge servers to return their computed results. The user only needs α_{iw} computed results where $\alpha_{iw} \leq K$. Note that although there is no constraint on the decoding functions to be used, a low-complexity decoding function such as the Reed-Solomon decoding algorithm [7] ensures the efficiency of the overall matrix multiplication computations.

Given that each edge server is able to store up to $\frac{1}{m_{iw}}$ and $\frac{1}{n_{iw}}$ fractions of matrices $\mathbf{A}^{iw}$ and $\mathbf{B}^{iw}$ respectively, the optimum recovery threshold that can be achieved by user d_{iw} using the Polynomial codes [6] is expressed as follows:

$$\alpha_{iw} = m_{iw} n_{iw}. \quad (5)$$

The cost of employing edge server k to complete the computation subtask is denoted as c_k. Hence, in order to complete its CDC task, user d_{iw} needs to pay a total amount of $\sum_{k\in\mathcal{L}_{iw}} c_k$, where $\mathcal{L}_{iw}$ represents the set of edge servers that successfully complete the computation subtasks allocated by user d_{iw}. For simplicity, we consider the cost of employing edge server k the same for all edge servers, i.e., $c_k = c$, $\forall k \in \mathcal{K}$. Hence, the total payment of user d_{iw} to complete its CDC task is given by $\alpha_{iw} c$.

III. Lower Level UAV Coalition Formation

For multiple UAVs to charge the users in the cells, the UAVs in a coalition need to cooperate with one another. In particular, they need to communicate with each other to determine the allocation of UAVs to the users in a cell. The total cooperation cost incurred by UAV j in coalition S_l is denoted as $\gamma_j(S_l)$. The cooperation cost is only incurred when there are two or more UAVs in a coalition. Otherwise, there is no cooperation cost incurred by singleton coalitions. Specifically, $\gamma_j(S_l)$ is defined as follows:

$$\gamma_j(S_l) = \begin{cases} |S_l| b_j, & \text{if } |S_l| \geq 2, \\ 0, & \text{if otherwise,} \end{cases} \quad \forall S_l \in \Pi, \quad (6)$$

where $|S_l|$ is the number of coalition members in coalition S_l and b_j is the cooperation cost of UAV j with any other UAV j', $j \neq j'$. In other words, $|S_l|$ is the coalition size of coalition S_l. The cost of the coalition formation is a non-decreasing function of the coalition size.

Since multiple UAVs are allowed to charge a cell of users, the actual charging energy of cell i by coalition S_l which is denoted by $e_i(S_l)$ is expressed as follows:

$$e_i(S_l) = \begin{cases} \sum_{j\in S_l} p_{ij}^v t_{ij} & \text{if } e_i^{max} - e_i^r > \sum_{j\in S_l} p_{ij}^v t_{ij}, \\ e_i^{max} - e_i^r & \text{if } e_i^{max} - e_i^r \leq \sum_{j\in S_l} p_{ij}^v t_{ij}. \end{cases} \quad (7)$$

The total cost of UAVs in coalition S_l to charge the users in cell i is given as follows:

$$c_i(S_l) = \theta_l e_i(S_l) + \sum_{j\in S_l} \gamma_j(S_l), \quad (8)$$

where θ_l is the unit cost of energy charged by coalition S_l.

Proposition 1. *The formation of a grand coalition, where all UAVs provide power to a single cell of users, is not always stable. Instead, smaller coalitions are formed in the network.*

Proof. The proof is omitted due to space constraints. □

Following Proposition 1 that proves the non-subadditivity of the coalitional game and the emptiness of core, the UAVs do not form a grand coalition. Instead, smaller and disjoint UAV coalitions are expected to form in the network.

In order to modify a coalitional structure $\Pi \in \mathcal{J}$, there are two important operations that are used following the merge-and-split algorithm [8].

- **Merge Rule:** Merge any set of coalitions $\{S_1, \ldots, S_l, \ldots, S_L\}$ where $\mathcal{C}(\Pi_{new}) < \mathcal{C}(\Pi)$, thus $\{S_1, \ldots, S_l, \ldots, S_L\} \rightarrow \bigcup_{l=1}^{L} S_l$.
- **Split Rule:** Split any set of coalitions $\{\bigcup_{l=1}^{L} S_l\}$ where $\mathcal{C}(\Pi_{new}) < \mathcal{C}(\Pi)$, thus $\bigcup_{l=1}^{L} S_l \rightarrow \{S_1, \ldots, S_l, \ldots, S_L\}$

IV. Upper Level Stochastic Offloading Optimization

To account for the stochastic nature of wireless power transfer reliability, we formulate the cost minimization problem as a two-stage SIP model [9].

The two stages are as follows:

- *First stage:* At the first stage, the network makes the *ex-ante* decision between i) full local computation, ii) full coded offloading, and iii) a hybrid approach for each user's computation task. This decision is made before the scenarios are known. Note that the scenarios refer to the varying combinations of charging efficiencies across the UAV coalitions, as described in Section II-A.
- *Second stage:* At the second stage, the scenarios are known since the wireless charging is completed. In other words, each user now knows the shortfall in computation as the charging efficiencies have been realized. To account for this shortfall, the user makes the *ex-post* decision to offload the remaining incomplete computation subtasks to the edge servers. Note that the decision to offload comes at a higher cost since extra time has elapsed for the realization of the first stage. This cost is known as the penalty cost.

In our problem formulation, the decision variables are as follows:

- δ_{il} is a binary variable that indicates whether coalition S_l is allocated to cell i. When $\delta_{il} = 1$, coalition S_l is allocated to cell i whereas $\delta_{il} = 0$ indicates otherwise.
- $\bar{\delta}_{il}$ is a non-negative variable that indicates the number of subtasks that are computed locally by the cell i. When $\bar{\delta}_{il} = 1$, cell i computes 1 subtask locally.
- μ_{ik} is a binary variable to indicate whether the subtask is offloaded to edge server k by cell i. Specifically, $\mu_{ik} = 1$ indicates that cell i offloads 1 subtask to the edge server k whereas $\mu_{ik} = 0$ indicates otherwise.
- $\mu_{ik}^{(2)}(\lambda_y)$ is a binary correction variable that indicates whether a correction action at stage 2 is performed after scenario λ_y has been realized. If $\mu_{ik}^{(2)}(\lambda_y) = 1$, this implies a subtask has to be offloaded to edge server k for computation at stage 2, whereas $\mu_{ik}^{(2)}(\lambda_y) = 0$ indicates otherwise.
- μ_i^p is a binary variable to indicate that a penalty cost c_i^p has been incurred at stage 2. When $\mu_i^p = 1$, this indicates that cell i has to perform a corrective edge offloading at stage 2, whereas $\mu_i^p = 0$ otherwise.

The objective function in (9) aims to minimize the total cost of the network. The expressions in (9) and (10) represent the first and the second stage objectives respectively. $\mathbb{E}\Big[\mathcal{Q}(\mu_{ik}^{(2)}(\lambda_y))\Big]$ is the expectation over all scenarios $\lambda_y \in \Lambda$. $P(\lambda_y)$ denotes the probability if scenario $\lambda_y \in \Lambda$ is realized. The SIP formulation can be expressed as follows:

Minimize:

$$\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}}\delta_{il} + \sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}}\bar{\delta}_{il}c_i(S_l) + \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{K}}\mu_{ik}c + \mathbb{E}\Big[\mathcal{Q}(\mu_{ik}^{(2)}(\lambda))\Big], \quad (9)$$

where:

$$\mathcal{Q}(\mu_{ik}^{(2)}(\lambda_y)) = \sum_{\lambda\in\Lambda} P(\lambda_y) \sum_{i\in\mathcal{I}} \Big(\mu_i^p(\lambda_y)c_i^p + \sum_{k\in\mathcal{K}} \mu_{ik}^{(2)}(\lambda_y)c\Big), \quad (10)$$

subject to:

$$\sum_{l\in\mathcal{L}}\bar{\delta}_{il}\sum_{w\in W_i}\alpha_{iw} + \sum_{k\in\mathcal{K}}\mu_{ik} \geq \sum_{w\in W_i}\alpha_{iw}, \qquad \forall i \in \mathcal{I}, \quad (11)$$

$$\bar{\delta}_{il} \leq \square\delta_{il}, \qquad \forall i \in \mathcal{I}, \forall l \in \mathcal{L}, \quad (12)$$

$$\sum_{k\in\mathcal{K}}\mu_{ik}^{(2)}(\lambda) \leq \square\mu_i^p(\lambda), \qquad \forall i \in \mathcal{I}, \forall \lambda \in \Lambda, \quad (13)$$

$$\sum_{i\in\mathcal{I}}\delta_{il} \leq 1, \qquad \forall l \in \mathcal{L}, \quad (14)$$

$$\sum_{l\in\mathcal{L}}\delta_{il} \leq 1, \qquad \forall i \in \mathcal{I}, \quad (15)$$

$$\sum_{i\in\mathcal{I}}\mu_{ik} \leq 1, \qquad \forall k \in \mathcal{K}, \quad (16)$$

$$\sum_{i\in\mathcal{I}}\mu_{ik}^{(2)}(\lambda) \leq 1, \qquad \forall k \in \mathcal{K}, \forall \lambda \in \Lambda, \quad (17)$$

$$\bar{\delta}_{il}p^s\sum_{w\in W_i}\alpha_{iw} \leq \sum_{j\in\mathcal{S}_l}p_j^c, \qquad \forall i \in \mathcal{I}, \forall l \in \mathcal{L}, \quad (18)$$

$$\sum_{l\in\mathcal{L}} \bar{\delta}_{il}\beta_l^i(\lambda) \sum_{w\in W_i} \alpha_{iw} + \sum_{k\in\mathcal{K}} \mu_{ik} + \sum_{k\in\mathcal{K}} \mu_{ik}^{(2)}(\lambda) \geq \sum_{w\in W_i} \alpha_{iw}, \quad \forall i \in \mathcal{I}, \forall \lambda \in \Lambda, \tag{19}$$

$$\delta_{il}, \mu_{ik}, \mu_{ik}^{(2)}(\lambda) \in \{0,1\}, \quad \forall i \in \mathcal{I}, \forall l \in \mathcal{L}, \forall k \in \mathcal{K}, \forall \lambda \in \Lambda, \tag{20}$$

$$\bar{\delta}_{il} \in \{0, 0.1, \ldots, 1\} \quad \forall i \in \mathcal{I}, \forall l \in \mathcal{L}. \tag{21}$$

The constraint in (11) ensures that the total number of subtasks that are computed should be greater than the total recovery threshold in the first stage. The constraint in (12) ensures that if any cell performs local computation, a coalition should be allocated to that cell, where $\square$ is any large number. The constraint in (13) ensures that the penalty cost is paid when any cell performs re-offloading. The constraints in (14) and (15) ensure a one-to-one matching between the cells and the UAV coalitions, i.e., only a single UAV coalition is allocated to a cell, and a cell cannot choose another coalition that has already been allocated. The constraint in (16) ensures that the cells do not choose any occupied edge servers. The constraint in (17) ensures that the cells do not choose repeated edge servers when performs re-offloading. The constraint in (18) ensures that the amount of energy that the cells require do not exceed the maximum energy that allocated coalitions provide, where p^s is the energy required to compute each subtask. The constraint in (19) ensures that if any cell performs local computation and when the wireless power transfer is unreliable, the correction action is performed, i.e., the subtasks have to offload to the edge servers. This enforces the completion of computation tasks. The constraint in (20) indicates that δ_{il}, μ_{ik} and $\mu_{ik}^{(2)}(\lambda)$ are binary variables. The last constraint in (21) indicates that $\hat{\delta}_i$ is a non-negative variable that has a value between 0 and 1 with a step-size of 0.1.

V. Performance Evaluation

We consider a UAV-assisted IoT network on a Cartesian grid of size 1000×1000 as shown in Fig. 1. Unless otherwise specified, the simulation parameter values used subsequently are summarized in Table I.

TABLE I: System Simulation Parameter Values.

Parameter	Values
Maximum storage energy of cell i, e_i^{max}	$[1000, 4000]J$
Remaining energy of cell i, e_i^r	$[500, 1000]J$
Flying energy of UAV j, p_j^f	$[10, 50]W$
Hovering energy of UAV j, p_j^h	$[1, 20]W$
Charging energy of UAV j, p_j^c	$[10, 100]W$
Maximum energy of UAV j, e_j^{max}	$[1000, 5000]J$
Unit cooperation cost of UAV j, b_j	$[1, 3]$
Unit charging cost of coalition S_l, θ_l	$[1, 10]$

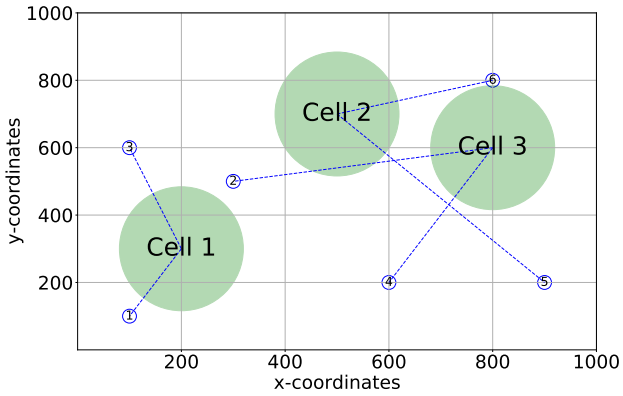

Fig. 1: Optimal allocation of UAV coalitions where the UAVs are represented by {1,2,3,4,5,6}.

The extensive experiments are conducted with two hundred identical edge servers. The value of α_{iw} is chosen randomly between 1 and 5, whereas the value of p^s is set to be 8. We consider two scenarios, i.e., $|\Lambda| = 2$, of wireless power charging efficiency. In the first scenario, the wireless power charging from all coalitions is not efficient λ_1, i.e., $\beta_l^i < 1$. In the second scenario, all the wireless power charging are efficient λ_2, i.e., $\beta_l^i = 1$. The probabilities of wireless power charging efficiency are $P(\lambda_1) = 0.3$ and $P(\lambda_2) = 0.7$. Through the coalition formation and SIP algorithm, we obtain {1,3}, {2,4}, {5,6} as the optimal coalitional structure and we use this coalitional structure for the rest of the simulations. Since the offloading cost is high, all cells want to compute the subtasks locally. With the energy constraint that limits the number of subtasks to compute locally, the optimal coalitional structure is formed such a way that all cells are able to compute some subtasks locally and then offload the remaining subtasks to the edge servers. For the presented experiments, we implement the SIP model using GAMS script [10], [11].

A. Edge server employing cost c

We study the effect of varying values of c. The value of β_l^i and c_i^p are kept constant at 0.8 and 200 respectively. Fig. 2 shows the cost break down when we vary c. Clearly, as the value of c increases, the stage 1 cost increases. Moreover, at high values of c, the stage 2 cost increases as well. The reason is that cells are less willing to use the edge servers at stage 1 and are switching to the hybrid computation approach. Under the hybrid computation approach, there is a likelihood of wireless charging inefficiency and hence a shortfall and the corresponding corrective offloading cost at stage 2.

B. Energy per subtask p^s

We vary the energy required to complete each subtask p^s in this simulation, and set $c = 2000$. The rest of the settings are kept the same as that of Section V-A. Figure 3 shows the cost break down of the network. The offloading cost of this network is high throughout the simulation. When $p^s = 2$, the

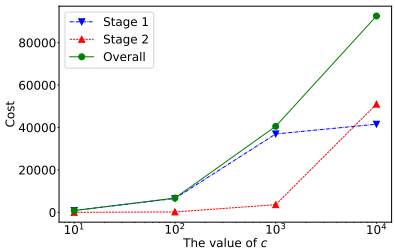

Fig. 2: The cost of the network by varying c.

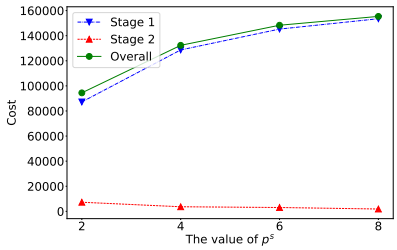

Fig. 3: The cost of the network by varying energy per subtask p^s.

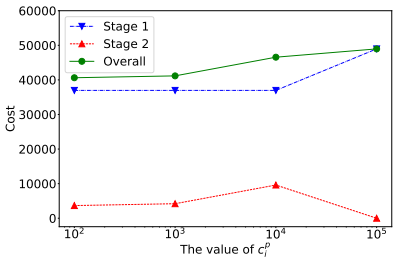

Fig. 4: The cost of the network by varying c_i^p.

cells try to compute as many subtasks in local computation and then offload the remaining subtasks to the edge servers. As p^s increases, the number of subtasks computed locally is reduced due to the charging energy constraints of the UAV coalitions. In other words, the number of subtasks that are offloaded to the edge servers increases. This explain why the cost at stage 1 increases. Moreover, with fewer subtasks computed locally, the likelihood that there is a shortfall to be corrected at stage 2 decreases. As such, the cost at stage 2 decreases.

C. Penalty cost c_i^p

Next, we set $p^s = 0.5$, $c = 1500$ and vary the penalty cost, while keeping the rest of the settings same as that of Section V-A. When $c_i^p = 100$, cells 1, 2 and 3 compute all the subtasks locally by using the wireless power charging from coalitions 2, 1 and 3 respectively. Moreover, given that the penalty cost is low, the cells only perform the corrective action, i.e., edge offloading, when there is inefficient wireless charging. As such, the cost is incurred at stage 2. Similarly, the above explanation applies for $c_i^p = 1000$ and $c_i^p = 10000$. However, when $c_i^p = 100000$, the penalty cost is very high. As a result, to avoid the penalty cost, the network is more prone to choosing the hybrid local computation approach where edge offloading is performed at the first stage. This ensures that even if local computation fails, the redundancy from offloading at the first stage enforces task completion. As a result, the cost incurred at stage 2 is zero since the potential shortfall from wireless charging inefficiency is avoided.

VI. Conclusion

In this paper, we proposed the use of UAVs as mobile charging station for the resource-constrained IoT devices in a hybrid coded edge computing network. Given the uncertainties of the UAVs in providing their power wirelessly, we derived the optimal UAV coalitional structure and computation approach of users in the network. For our future work, we can consider the optimization of the travelling path of the UAVs.

Acknowledgements

This research is supported, in part, by the programme DesCartes and is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme, Alibaba Group through Alibaba Innovative Research (AIR) Program and Alibaba-NTU Singapore Joint Research Institute (JRI), the National Research Foundation, Singapore under the AI Singapore Programme (AISG) (AISG2-RP-2020-019), WASP/NTU grant M4082187 (4080), Singapore Ministry of Education (MOE) Tier 1 (RG16/20). This research is also supported in part by the SUTD SRG-ISTD-2021-165, in part by the SUTD-ZJU IDEA under Grant (SUTD-ZJU (VP) 202102) and in part by the SUTD-ZJU IDEA Seed under Grant (SUTD-ZJU (SD) 202101).

References

[1] S. Zermani, C. Dezan, and R. Euler, "Embedded Decision Making for UAV Missions," in *2017 6th Mediterranean Conference on Embedded Computing (MECO)*, (Bar, Montenegro), pp. 1–4, 2017.

[2] M. R. Brust, G. Danoy, P. Bouvry, D. Gashi, H. Pathak, and M. P. Gonçalves, "Defending Against Intrusion of Malicious UAVs with Networked UAV Defense Swarms," in *2017 IEEE 42nd Conference on Local Computer Networks Workshops (LCN Workshops)*, (Singapore), pp. 103–111, 2017.

[3] H. Guo, J. Liu, J. Zhang, W. Sun, and N. Kato, "Mobile-Edge Computation Offloading for Ultradense IoT Networks," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4977–4988, 2018.

[4] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding Up Distributed Machine Learning Using Codes," *IEEE Transactions on Information Theory*, vol. 64, no. 3, pp. 1514–1529, 2018.

[5] G. Zhang, C. Li, H. Zhang, and X. Jiang, "Parameters Optimization for Magnetic Resonance Coupling Wireless Power Transmission," *The Scientific World Journal*, vol. 2014, pp. 1–8, 2014.

[6] Q. Yu, M. Maddah-Ali, and S. Avestimehr, "Polynomial Codes: an Optimal Design for High-Dimensional Coded Matrix Multiplication," in *Advances in Neural Information Processing Systems 30 (NIPS 2017)* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), pp. 4403–4413, Curran Associates, Inc., 2017.

[7] F. Didier, "Efficient Erasure Decoding of Reed-Solomon Codes," *arXiv preprint arXiv:0901.1886*, 2009.

[8] K. R. Apt and T. Radzik, "Stable Partitions in Coalitional Games," *arXiv preprint arXiv:cs/0605132*, 2006.

[9] S. Sawadsitang, R. Kaewpuang, S. Jiang, D. Niyato, and P. Wang, "Optimal Stochastic Delivery Planning in Full-Truckload and Less-Than-Truckload Delivery," in *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, pp. 1–5, 2017.

[10] D. Chattopadhyay, "Application of General Algebraic Modeling System to Power System Optimization," *IEEE Transactions on Power Systems*, vol. 14, no. 1, pp. 15–22, 1999.

[11] G. D. Corporation, "General Algebraic Modeling System (GAMS) Release 24.2.1." Washington, DC, USA, 2013.