IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 6, 2020, accepted October 14, 2020, date of publication October 21, 2020, date of current version October 30, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.3032735*

# Privacy-Preserving and Trustworthy Device-to-Device (D2D) Offloading Scheme

**HOSUNG BAEK[1], HANEUL KO[2], (Member, IEEE), AND SANGHEON PACK[1], (Senior Member, IEEE)**

[1]School of Electrical Engineering, Korea University, Seoul 02841, South Korea
[2]Department of Computer Convergence Software, Korea University, Sejong 30019, South Korea

Corresponding author: Sangheon Pack (shpack@korea.ac.kr)

This work was supported in part by National Research Foundation (NRF) funded by the Korean Government (No. 2020R1A2C3006786) and in part by the ITRC (Information Technology Research Center) support program (IITP-2020-2017-0-01633) supervised by the IITP (Institute for Information & communications Technology Planning & Evaluation).

**ABSTRACT** In device-to-device (D2D) offloading, frequent offloading to a specific mobile device (i.e., an offloadee) can violate the privacy of a task owner and proper offloading results cannot always be guaranteed due to untruthful mobile devices. To address these problems, we propose a privacy-preserving and trustworthy D2D offloading scheme (PPTS) with two steps: 1) a privacy-preserving offloading step and 2) a blockchain-based verification step. In the first step, a task owner selects several offloadees and offloads the tasks redundantly to them to obtain reliable offloading results with a minimum task completion delay while preserving its own privacy at a sufficient level. In addition, the task owner chooses another mobile device as a verifier. In the second step, the task owner and verifier exploit blockchain networks to check whether the offloading results are appropriately processed. We formulate a static game to decide an appropriate redundancy ratio of offloading tasks and incentives for the offloadees.The evaluation results demonstrate that PPTS can provide reliable offloading results with a reduced task completion delay and guarantee a sufficient level of privacy of the task owner.

**INDEX TERMS** Blockchain, device-to-device (D2D), offloading, privacy, static game.

## I. INTRODUCTION

With the recent advancement of mobile devices with high computing powers, there is an increasing interest in device-to-device (D2D) offloading [1]–[20] that leverages numerous collaborative mobile devices. In D2D offloading, a mobile device (i.e., task owner) offloads tasks to other mobile devices in its vicinity. Subsequently, the mobile devices process the offloaded tasks by using their idle resources. Because such D2D offloading can work without any infrastructure, it can alleviate the network congestion and connectivity problems.

However, there can be untruthful mobile devices that return fake offloading results without processing tasks and expect only an incentive for the results. Accordingly, a reputation-based scheme [4] was proposed to mitigate this problem. In this scheme, a task owner calculates the reputations of mobile devices based on historical information indicating whether these devices have appropriately processed offloaded tasks. Subsequently, the task owner offloads the task to mobile devices with high reputations. However, it is not trivial for the task owner to maintain this historical information. For example, if mobile devices arrive frequently in the vicinity of the task owner in D2D environments, the task owner cannot easily maintain historical information due to mobility. Furthermore, when the tasks are frequently offloaded to a specific mobile device with a high reputation, the privacy of the task owner cannot be preserved (i.e., the usage patterns of the task owner may be exposed to external mobile devices) [15], [16].

To address this issue, we propose a privacy-preserving and trustworthy D2D offloading scheme (PPTS) with two steps. In the first step, a task owner determines the appropriate mobile devices (i.e., offloadees) and the number of tasks to be offloaded to each of these devices. Subsequently, it offloads the tasks redundantly to them in order to obtain trustful offloading results with the minimum task completion delay,

The associate editor coordinating the review of this manuscript and approving it for publication was Yuan Gao.

while preserving its privacy at a sufficient level. In addition, the task owner chooses another mobile device as a verifier. In the second step, the task owner and verifier exploit blockchain networks to check whether the offloading results are appropriately processed. We formulate a static game to decide an appropriate redundancy ratio of the offloading tasks and incentives for offloadees. The evaluation results demonstrate that PPTS can guarantee a sufficient level of privacy of the task owner and provide reliable offloading results with a significantly reduced task completion time compared with that of a random scheme in which offloadees are randomly selected. In addition, the performance of our proposed privacy-preserving offloading algorithm is comparable to that of the optimal scheme.

The main contributions of this paper can be summarized as follows. First, to the best of our knowledge, this is the first work to consider the privacy of the task owner and the reliability of offloading results simultaneously. Second, we determine the appropriate redundancy ratio of offloading tasks and incentive to make offloadees process the tasks properly based on the static game. Finally, we present the evaluation results under various environments, which are expected to serve as valuable guidelines for designing privacy-preserving and trustworthy D2D offloading schemes in the future.

The remainder of this paper is organized as follows. The related works are summarized in Section II. Subsequently, the system model and PPTS are described in Section III and Section IV, respectively. After that, the static game is formulated in Section V. The evaluation results are presented in Section VI. Finally, our concluding remarks are provided in Section VII.

## II. RELATED WORK

A number of studies on D2D offloading have been conducted in the literature [4], [11]–[16]. Based on their objectives, these works can be classified into the following categories: 1) reduction of task completion delay [11]–[13]; 2) provision of reliable task processing results [4], [14]; and 3) solution to the privacy leakage problem [15], [16].

Chen *et al.* [11] investigated a joint task assignment and task execution problem to optimize the task completion delay by considering energy consumption and task type constraints. Shi *et al.* [12] formulated a task scheduling problem in local mobile clouds by satisfying the task completion delay constraints while maintaining a low energy consumption. Guo *et al.* [13] proposed a resource scheduling policy to shorten task completion delay by comparing the local execution cost at an optimal clock frequency with the cloud execution cost at optimal transmission power. Although these works could reduce the task completion delay of D2D offloading, they did not consider untruthful mobile devices that may return fake computation results only for obtaining incentives [11]–[13].

Chatzopoulos *et al.* [4] proposed a flopcoin framework, which rewards collaborating mobile devices while punishing selfish ones, to obtain reliable offloading results. Similarly, Restuccia and Das [14] proposed a trust-based framework in which mobile security agents rule out incorrect reports and reward reliable users. However, these frameworks cannot always guarantee that the mobile devices (i.e., offloadees) will provide reliable offloading results because the offloadees are selected based on historical (or previous) information, which only indicates whether they had appropriately processed offloaded tasks [4], [14]. In addition, the privacy of the task owner was not examined in these works.

Ni *et al.* [15] proposed a random matrix-based location matching approach for mobile crowdsensing to allocate tasks without disclosing the location of the mobile devices. He *et al.* [16] presented a task offloading scheduling algorithm based on a constrained Markov decision process problem to achieve a low task completion delay and low energy consumption while maintaining a sufficient level of privacy on the location and usage pattern. However, these works did not consider the reliability of the offloading results.

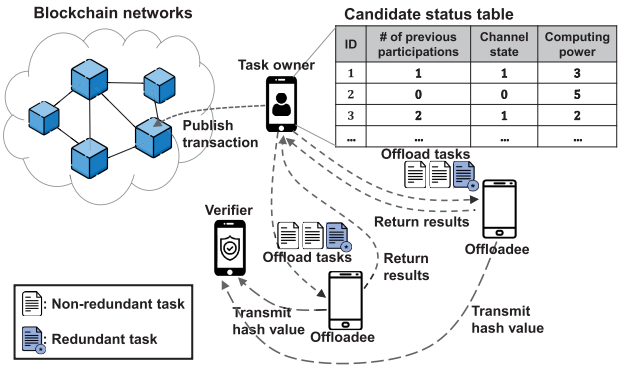



**FIGURE 1.** System model.

## III. SYSTEM MODEL

As shown in Fig. 1, a mobile device (i.e., task owner) offloads tasks to adjacent mobile devices (i.e., offloadees) through a wireless communication technology [21], [22] such as cellular D2D, WiFi-Direct, and Bluetooth. The task is characterized by $\zeta_D$, which is its data size (i.e., the number of bytes), and $\zeta_C$, which is the amount of computing resources required (i.e., the number of CPU cycles) [2]. Specifically, the task owner maintains some information of offloadees, such as number of previous participations in D2D offloading, $N_P$, channel state $H$, and computing power $P_C$ (i.e., CPU cycles per second) in a candidate status table.[1] Based on Shannon theory, the D2D transmission rate $V_i$ from task owner to the $i$th mobile device can be expressed as $V_i = W \log_2(1 + \frac{\epsilon_i \eta_i^2}{\sigma(g) d_i^{\theta} I})$ where $W$ is the bandwidth allocated for the D2D link, $\epsilon_i$ is the D2D transmission power of the $i$th mobile device, $\eta_i$ is the channel fading coefficient, $d_i$ is the distance

[1] Note that, for simplicity, we only considered a simple probability-based good/bad channel model. Therefore, the channel fading coefficient is assumed to be estimated based on the channel state $H$, which has two conditions, i.e., good or bad. That is, if the channel condition is good, $H$ is 1; otherwise, $H$ is 0.

between the task owner and the $i$th mobile device, $\theta$ is the path loss exponent, and $I$ denotes the noise power. $\sigma(g) = -\frac{2\log(5g)}{3}$ represents the SNR margin introduced to meet the desired target bit error rate (BER), denoted by $g$, with a QAM constellation [20].

Based on this information, the task owner selects $N_O$ appropriate offloadees and the number of tasks to be offloaded to each offloadee in order to obtain reliable offloading results with the minimum task completion delay while preserving its privacy at a sufficient level. Note that the task owner selects the offloadees and a verifier through the first step of PPTS (i.e., the privacy-preserving offloading step), which is elaborated in Section IV-A. After selecting the offloadees, the task owner offloads its tasks redundantly to them. The appropriate redundancy ratio of the offloading tasks is decided by the static game, which is described in Section V, and the task owner offloads the same redundant tasks to all the offloadees.

In the second step of PPTS (i.e., the blockchain-based verification step), verification of the task results and incentive transfer are performed using blockchain networks in which the users can transact directly without a trusted authority. Each transaction in blockchain networks works for the monetary transfer and involves the amount of incentives, incentive source (i.e., the task owner), incentive destination (i.e., offloadees and verifier), and condition (i.e., the number of redundantly offloaded tasks). Note that the numbers of transactions required to deliver the incentives from the task owner to $N_O$ offloadees and the verifier are $N_O$ and 1, respectively. Consequently, the task owner needs to construct $(N_O + 1)$ transactions to verify the task results. In addition, the task owner should allocate a certain amount of incentives as a deposit, which will be delivered only when the verification is completed [23]–[25]. The detailed verification procedure will be elaborated in Section IV-B.

## IV. PRIVACY-PRESERVING AND TRUSTWORTHY D2D OFFLOADING SCHEME (PPTS)

PPTS consists of two steps: 1) a privacy-preserving offloading step and 2) a blockchain-based verification step. These two steps are described in the following subsections.

### A. PRIVACY-PRESERVING OFFLOADING STEP

In the privacy-preserving offloading step, the task owner first broadcasts a message to nearby mobile devices (i.e., candidates for offloadees and verifier). When receiving this message, the mobile devices transmit their information (i.e., identifiers (IDs) and computing powers $P_C$) to the task owner if they want to participate in D2D offloading. After that, the task owner selects $N_O$ offloadees and one verifier among the mobile devices, and then offloads its tasks redundantly to the selected offloadees.

We formulate an integer nonlinear programming (INLP) problem to obtain reliable offloading results with the minimum task completion delay while preserving the privacy of the task owner at a sufficient level. The important notations in this paper are summarized in Table 1.

**TABLE 1. Summary of Notations.**

| Parameter | Description |
|---|---|
| $x_i$ | Binary variable to denote whether the mobile device $i$ is selected as an offloadee |
| $X_O$ | Vector denoting selected mobile devices |
| $N_T$ | Vector representing the number of tasks offloaded to each offloadee |
| $N_O$ | Number of offloadees |
| $N_M$ | Number of mobile devices that participate in the offloading process |
| $N_{P,i}$ | Number of previous offloadings to the $i$th offloadee |
| $n_{T,i}$ | Number of tasks offloaded to the $i$th mobile device |
| $N_{T,O}$ | Number of original tasks to be offloaded |
| $N_{T,T}$ | Total number of tasks to be offloaded |
| $N_{v,i}$ | Maximum number of tasks having the same hash values of the $i$th offloadee |
| $\zeta_D$ | Data size of the task |
| $\zeta_C$ | Required computing resources to process the task |
| $\tau_T$ | Task completion time |
| $\tau_i$ | Task completion time of the $i$th mobile device |
| $V_i$ | Data transmission rate between the task owner and the $i$th mobile device |
| $\eta_i$ | Channel fading coefficient between the task owner and the $i$th mobile device |
| $\epsilon_i$ | D2D transmission power of the $i$th mobile device |
| $P_{C,i}$ | Computing power of the $i$th mobile device |
| $R_O$ | Redundancy ratio of the offloading tasks |
| $C_A$ | Combination of $N_O$ mobile devices among $N_M$ mobile devices |
| $\alpha$ | Unit incentive for processing one task |
| $E_P$ | Unit energy consumption to process one task |
| $\rho_P$ | Probability that some (not all) tasks are processed and there is no problem in the verification process |

Note that, in PPTS, offloadees receive the incentives only when they process the redundant tasks appropriately. Therefore, it can be assumed that the reliability of results can be achieved by redundant offloading. In this situation, the objective of the privacy-preserving offloading step is to minimize the task completion delay. Accordingly, we have to select appropriate offloadees (i.e., those with high computing powers and good channel states) and number of tasks offloaded to each offloadee.

Let $X_O$ and $N_T$ be the vectors representing the selected mobile devices and the number of tasks offloaded to each offloadee, respectively. The task completion delay $\tau_T$ is decided by the offloadee that completes its tasks last. Therefore, considering that $\tau_i$ denotes the task completion delay of the $i$th mobile device selected as an offloadee, $\tau_T$ can be expressed as

$$\tau_T = \max_{X_O, N_T} \tau_i \cdot x_i \tag{1}$$

where $x_i$ is a binary variable denoting whether the $i$th mobile device is selected as an offloadee. That is, if the $i$th mobile device is selected as an offloadee, $x_i = 1$; otherwise, $x_i = 0$.

$\tau_i$ consists of 1) the task transmission delay from the task owner to the $i$th mobile device; 2) the processing delay of the $i$th mobile device; and 3) the result transmission delay from the $i$th mobile device to the task owner. However, the result

transmission delay from the $i$th mobile device to the task owner can be neglected because the result is relatively small in size compared with the offloading tasks [26]. Furthermore, when $n_{T,i}$ and $V_i$ denote the number of tasks offloaded to the $i$th mobile device and the data transmission rate between the task owner and the $i$th mobile device, respectively, the task transmission delay can be calculated by $\frac{n_{T,i}\cdot\zeta_D}{V_i}$. In addition, considering that the $i$th mobile device has a computing power of $P_{C,i}$, the processing delay of the $i$th mobile device is $\frac{n_{T,i}\cdot\zeta_C}{P_{C,i}}$. To sum up, $\tau_i$ can be represented as

$$\tau_i = \frac{n_{T,i}\cdot\zeta_D}{V_i} + \frac{n_{T,i}\cdot\zeta_C}{P_{C,i}}. \tag{2}$$

As mentioned above, our objective is to minimize the task completion delay $\tau_T$, which can be expressed as

$$\min_{X_O, N_T} \tau_T. \tag{3}$$

The problem formulated above is an INLP problem due to the multiplication of two decision variables (i.e., $x_i$ and $n_{T,i}$) in its objective function.

The constraints for offloadee selection are as follows. First, the task owner selects $N_O$ offloadees in PPTS. Thus, we have

$$\sum_{i\in M} x_i = N_O. \tag{4}$$

The task owner offloads the same redundant tasks to all the selected mobile devices (i.e., all offloadees) for verification. Therefore, the number of offloaded tasks to each offloadee should be larger than the number of redundant tasks. Thus, the corresponding constraint can be expressed by

$$n_{T,i}\cdot x_i \geq N_{T,O}\cdot R_O, \text{ for } \forall i. \tag{5}$$

where $R_O$ denotes the redundancy ratio. Note that $R_O$ can be decided based on the analysis results of the static game in Section V. Therefore, $R_O$ is assumed to be known in advance for the INLP problem.

Furthermore, the total number of offloaded tasks (including redundant tasks), denoted by $N_{T,T}$, should be equal to the sum of the numbers of tasks offloaded to the offloadees (i.e., $\sum_i n_{T,i}\cdot x_i$). Therefore, we have

$$\sum_i n_{T,i}\cdot x_i = N_{T,T}. \tag{6}$$

If too many tasks are offloaded to a specific offloadee, it is easier to infer the entire data [27]. Thus, the ratio of the tasks offloaded to each offloadee must be less than a given threshold $\delta$, to avoid such a situation. Thus, we have

$$\frac{n_{T,i}\cdot x_i}{N_{T,O}} \leq \delta, \text{ for } \forall i. \tag{7}$$

When the task owner frequently offloads its tasks to the same offloadee, the data usage pattern of the task owner can be easily inferred by this offloadee [16]. Thus, to protect the privacy of the task owner in terms of its usage patterns, an additional constraint is defined as

$$N_{P,i}\cdot x_i \leq \gamma, \text{ for } \forall i. \tag{8}$$

where $N_{P,i}$ is the number of previous offloadings to the $i$th offloadee, and $\gamma$ is the desired level of privacy of the task owner regarding its usage pattern.

If a brute-force approach is used to solve the INLP problem formulated above, the complexity is given by $O(2^{N_M}\cdot {N_{T,O}}^{N_M})$, where $N_M$ is the total number of mobile devices that want to participate in the D2D offloading, because we have to check all the possible combinations of $X_O$ and $N_T$. Since this complexity is too high (i.e., a non-polynomial function), the brute-force approach is not feasible. Consequently, we propose a low-complexity heuristic for privacy-preserving offloading to determine a sub-optimal solution in a practical manner (see **Algorithm 1**).

**Algorithm 1** Privacy-Preserving Offloading Algorithm

1: Obtain all combinations $C_A$ of $N_O$ mobile devices among $N_M$ mobile devices
2: $k \leftarrow 0, j \leftarrow 0$
3: **repeat**
4: $\quad k \leftarrow k+1$
5: $\quad$ **if** $N_{P,i} \leq \gamma$, for $\forall i \in C^k$ **then**
6: $\quad\quad j \leftarrow j+1$
7: $\quad\quad C_U^j = C^k$
8: $\quad\quad S_V^j = \sum_{i\in C_U^j} V_i$
9: $\quad$ **end if**
10: **until** $k = \binom{N_M}{N_O}$
11: Select offloadees in $C_U^{j^*}$, where $j^* = \arg\max_j S_V^j$
12: Select the numbers of tasks to be offloaded to each offloadee in proportion to its computing power while satisfying $\frac{n_{T,i}}{N_{T,O}} \leq \delta$
13: Offload tasks to offloadees

The algorithm is described in detail as follows. First, **Algorithm 1** obtains all combinations $C_A$ of selecting $N_O$ mobile devices among $N_M$ mobile devices (line 1). Note that the total number of combinations is $\binom{N_M}{N_0} = \frac{N_M!}{N_O!(N_M-N_O)!}$. To find the combinations of mobile devices satisfying the usage pattern privacy constraint in (8), **Algorithm 1** checks all combinations $C_A$. If all the mobile devices in the $k$th combination $C^k$ satisfy the aforementioned constraint (line 5), **Algorithm 1** stores this combination into $C_U^j$ (line 7). In addition, **Algorithm 1** calculates the sum of the data transmission rates of the mobile devices in $C_U^j$ (line 8). After finding all combinations of mobile devices satisfying the usage pattern privacy constraint, **Algorithm 1** selects offloadees with the highest summation of the data transmission rates (line 11). Subsequently, **Algorithm 1** selects the offloadees with the highest summation of data transmission rates. Specifically, **Algorithm 1** selects the number of tasks to be offloaded in proportion to the computing power of each offloadee while satisfying $\frac{n_{T,i}}{T} \leq \delta$ (i.e., constraint in (7)) (line 12). Finally, the task owner offloads tasks to the offloadees (line 13).

As mentioned before, **Algorithm 1** finds the set of offloadees that satisfies the usage pattern privacy constraint and

has the highest summation of data transmission rates among $N_M$ mobile devices [28]. Therefore, its complexity is given by $O(\frac{N_M!}{N_O!(N_M-N_O)!})$ and its upper bound can be obtained as $O\left({N_M}^{N_O}\right)$ by Theorem 1 (see Appendix A for its proof).

*Theorem 1:* The complexity of **Algorithm 1** is bounded to $O\left({N_M}^{N_O}\right)$ and the number of offloadees is a system parameter (i.e., constant value). Therefore, **Algorithm 1** has polynomial time complexity.

Since D2D offloading is used among nearby devices and the communication range is limited (e.g., 20 meters in WiFi Direct), the number of offloadees (i.e.,$N_O$) and the number of mobile devices (i.e., $N_M$) are typically small values, e.g., $N_O$ is 1 or 2 and $N_M$ is less than 50 [3], [4]. Consequently, the input size of **Algorithm 1** is also small and it can be run in polynomial time at the task owner. As reported in [29], the energy consumption of an algorithm to solve a problem in polynomial time is not significant. Therefore, the power consumption to run **Algorithm 1** is also low.

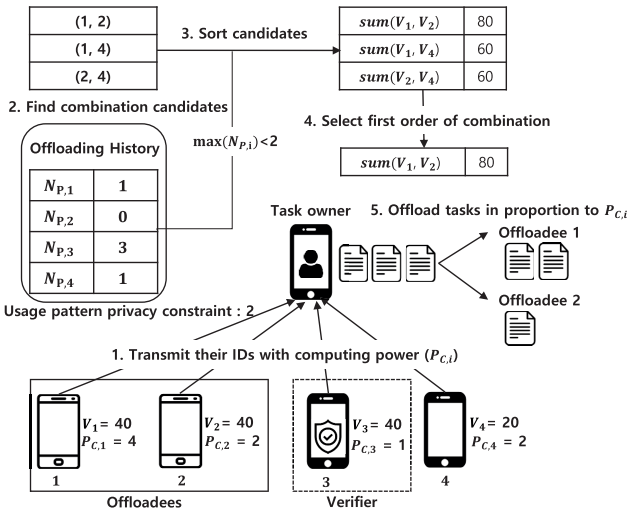



**FIGURE 2.** Operational example of privacy-preserving offloading step.

Fig. 2 presents an example of selecting the offloadees and verifier using the heuristic privacy-preserving offloading algorithm. In this example, four mobile devices want to participate in D2D offloading. In addition, the task owner sets $N_O$ to 2. After the mobile devices send their computing powers and IDs to the task owner (step 1), the task owner finds all combinations of selecting two mobile devices who satisfy the usage pattern privacy constraint (step 2). Subsequently, it obtains the summation of $V_i$ of these combinations and sorts them in descending order (step 3). Mobile devices 1 and 2 are selected as offloadees because they have the highest summations of $V_i$ (step 4). After that, the task owner offloads tasks to the selected offloadees in proportion to $P_{C,i}$. In this example, $N_{T,T}$ is 3 and $P_{C,i}$ of offloadee 1 and 2 are 4 and 2, respectively. Therefore, the task owner offloads two tasks to offloadee 1 and one task to offloadee 2 (step 5). Moreover, the mobile device 3 is selected as the verifier because it has a higher data rate than mobile device 4. Note that since the verifier does not process any task requiring high computing power,[2] the task owner selects the verifier based on the transmission rate without considering its computing power.

### B. BLOCKCHAIN-BASED VERIFICATION STEP

The blockchain-based verification step is performed after the tasks are offloaded. This step is described in detail as follows. First, the task owner constructs $(N_O+1)$ transactions (i.e., transaction $O_i$ $(1 \leq i \leq N_O)$ and transaction $V$ for the verifier) to verify the task results and deliver the incentives to the $i$th offloadee.[3] After constructing transactions, the task owner encrypts the number of redundant tasks (i.e., $T \cdot R_O$) by its own public key to prevent it from being decrypted by other mobile devices (i.e., verifier and offloadees). That is, the task owner obtains the encrypted value $E_O(T \cdot R_O)$, where $E_O(a)$ is an encryption function with an input $a$ based on the task owner's public key. Subsequently, the task owner publishes the encrypted value to blockchain networks. This encrypted value will be used for a comparison with the number of redundant tasks counted by the verifier.

After processing the tasks, offloadees transmit the results to the task owner. After that, they compute the hash values of the task results to reduce the result sizes for a tactical comparison and transmit these values to the verifier. On receiving these values, the verifier counts the number of tasks having the same hash values between all pairs of offloadees and obtains their maximum number for each offloadee to determine the number of redundantly offloaded tasks.[4] Specifically, the verifier compares the hash values from the $i$th offloadee to those from the $j$th offloadee and obtains the number of tasks with the same hash values, denoted by $N_{v,i,j}$ (for $\forall j \in X_O \setminus i$). Subsequently, it can obtain the maximum number of tasks having the same hash value $N_{v,i} = \max_j N_{v,i,j}$ for each offloadee. The verifier performs this procedure for all the offloadees. After that, it encrypts all $N_{v,i}$ (for $\forall i$) by its own public key. That is, it obtains the encrypted values $E_V(N_{v,i})$ (for $\forall N_{v,i}$) where $E_V(a)$ is an encryption function with an input $a$ based on the verifier's public key. After the encryption, the verifier publishes these encrypted values to the blockchain networks.

The miners in the blockchain networks verify the transactions using these published values (i.e., $E_V(N_{v,i})$ and $E_O(T \cdot R_O)$). Specifically, for the transaction $O_i$, the miners check whether $E_V(E_O(T \cdot R_O))$ and $E_O(E_V(N_{v,i}))$ are identical by using a cryptographic primitive. The cryptographic primitive employed in this case is that doubly-encrypted values with two public keys are always the same regardless of their encryption orders [4]. Using this cryptographic primitive, one cannot decrypt the ciphertexts (i.e., $E_V(E_O(T \cdot R_O))$ and

[2]The verifier simply receives the hash values and counts the number of tasks having the same hash values.

[3]Note that the transactions are monetary transfers in the blockchain networks. The offloadees and verifier can obtain some incentives only when prespecified conditions in the transactions are satisfied.

[4]The hash values of redundantly offloaded task results should be identical since they are the same tasks.

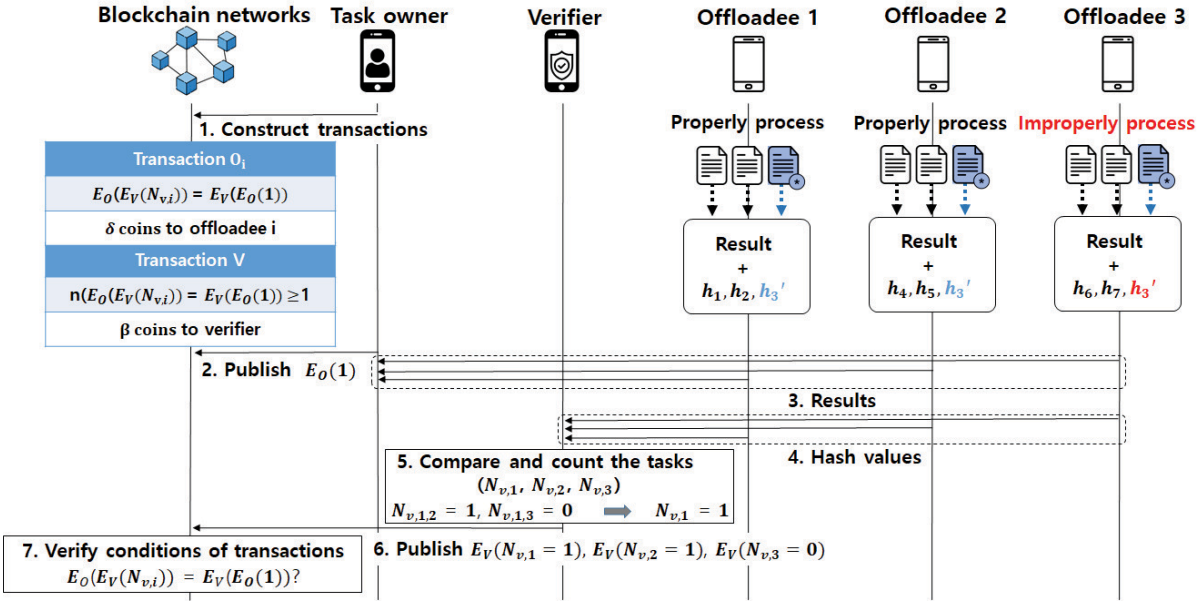



**FIGURE 3.** Operational example of blockchain based verification step.

$E_O(E_V(N_{v,i}))$) without the help of the other user. If these values are identical, it indicates that the $i$th offloadee has processed tasks properly, and thus, it can receive incentives. Otherwise, the $i$th offloadee cannot obtain any incentives. Furthermore, for the transaction $V$, the miners check whether any $E_O(E_V(N_{v,i}))$ and $E_V(E_O(T \cdot R_O))$ have the equivalent values. If these values are identical, it indicates that the verifier performed the verification appropriately, and therefore, it can receive some incentives. Otherwise, no incentives are provided to the verifier. Note that the condition of the transaction $V$ is derived from the assumption that at least one of $N_{v,i}$ has the same value as $(T \cdot R_O)$ if there are at least two offloadees that processed tasks properly.[5]

Fig. 3 shows an example of the verification step. Three offloadees are assumed in this example. Therefore, the task owner constructs four transactions (i.e., transactions $O_1$, $O_2$, $O_3$ for the offloadees and $V$ for the verifier) (step 1). Subsequently, the task owner publishes $E_O(1)$ to blockchain networks (step 2) because it is assumed that the number of redundant tasks (i.e., $T \cdot R_O$) is 1 in this example. Moreover, the task results and the hash values of the results are transmitted to the task owner and verifier, respectively (steps 3-4). After receiving the hash values, the verifier counts the number of tasks having the same hash values between all pairs of offloadees and obtains the maximum numbers for each offloadee (step 5). In this example, it is assumed that the 1st and 2nd offloadees process tasks properly, and the 3rd offloadee does not process tasks properly and transmits fake hash values on the task results. Therefore, in the case of the 1st offloadee, $N_{v,1,2}$ is 1 because the hash values of redundant task results of the 1st and 2nd offloadees are the same. However, $N_{v,1,3}$ is 0 because the hash values of the redundant task result of the 1st and 3rd offloadees are not the same. Accordingly, $N_{v,1}$ is 1 because the maximum number of tasks having the same hash values (i.e., $N_{v,1,2}$ and $N_{v,1,3}$) is 1. Similarly, $N_{v,2}$ is 1. However, $N_{v,3}$ is 0 because there are no task results having the same hash value from the other offloadees (i.e., both $N_{v,3,1}$ and $N_{v,3,2}$ are 0). The verifier encrypts $N_{v,1}$, $N_{v,2}$, and $N_{v,3}$ using its own public key and transmits them to blockchain networks. Then, the miners in blockchain networks encrypt $E_V(N_{v,1})$, $E_V(N_{v,2})$, and $E_V(N_{v,3})$ by the task owner's public key and $E_O(T \cdot R_O)$ by the verifier's public key. Finally, the miners check whether $E_O(E_V(N_{v,i}))$ $(1 \leq i \leq 3)$ and $E_V(E_O(T \cdot R_O))$ are identical. Then, the 1st and 2nd offloadees can obtain $\delta$ incentives because $E_O(E_V(N_{v,1}))$, $E_O(E_V(N_{v,2}))$, and $E_V(E_O(T \cdot R_O))$ are the same. Furthermore, the verifier can obtain $\beta$ incentive because $E_O(E_V(N_{v,i}))$ (for $i = 1$ and $i = 2$) and $E_V(E_O(T \cdot R_O))$ are the same. However, the 3rd offloadee cannot obtain the incentive because $E_O(E_V(N_{v,3})) = 0$ is different from $E_V(E_O(T \cdot R_O))$.

[5]Providing appropriate incentives to the offloadees is believed to encourage them to process the tasks appropriately. Therefore, this assumption is considered reasonable. In addition, the probability that at least two offloadees among 10 offloadees process the tasks properly is larger than 99% since the probability that offloadees do not process the tasks properly is under 0.5 in practical D2D offloading systems [30], [31].

## V. INCENTIVE AND REDUNDANCY RATIO ANALYSIS

In this section, we formulate a static game to decide an appropriate redundancy ratio of the offloading tasks and incentives for the offloadees. This static game consists of three elements: players, actions, and utilities. In our static game, the players are $N_O$ offloadees. In addition, each player $i$ (i.e., the $i$th offloadee) has an action set $A$ that can be expressed as $A = \{0, 1, \ldots, n_{T,i}\}$, where $A = j$ indicates that the $i$th offloadee processes $j$ tasks properly among $n_{T,i}$ tasks. Furthermore, we define a utility function $u_i$ of player $i$ considering its incentive and energy consumption. Let $\alpha$ and $E_P$ be the unit

incentive and the unit energy consumption for processing one task, respectively. Thus, if player $i$ processes all the offloaded tasks properly (i.e., $j = n_{T,i}$) by consuming $E_P \cdot n_{T,i}$ energy,[6] player $i$ always receives $\alpha \cdot n_{T,i}$ incentives. In contrast, in the case of $0 < j < n_{T,i}$, player $i$ consumes $E_P \cdot j$, and it can obtain $\alpha \cdot n_{T,i}$ incentives with the probability $\rho_P$. Here, $\rho_P$ is the probability that player $i$ processes only a part of the tasks (i.e., $j$ tasks) properly and it can obtain the incentive by chance. To sum up, the utility $u_i$ can be defined as

$$u_i = \begin{cases} \omega\alpha \cdot n_{T,i} - (1-\omega)E_P \cdot n_{T,i}, & \text{if } j = n_{T,i} \\ \omega\alpha \cdot n_{T,i} \cdot \rho_P - (1-\omega)E_P \cdot j, & \text{if } 0 < j < n_{T,i} \end{cases} \quad (9)$$

where $\omega$ is a weighted factor to balance incentives and the energy consumption.

The aforementioned $\rho_P$ can be calculated as $\binom{j}{N_{T,O} \cdot R_O} / \binom{n_{T,i}}{N_{T,O} \cdot R_O}$ because the redundantly offloaded tasks should be included in the properly processed tasks to obtain the incentives. If player $i$ processes all the redundant tasks properly, it can receive the incentive because the verification is performed with the redundantly offloaded tasks. Furthermore, if the number of tasks excluding the properly processed task (i.e., $j$) is smaller than the number of redundantly offloaded tasks (i.e., $N_{T,O} \cdot R_O$), the situation where player $i$ processes all the redundant tasks properly cannot occur. Therefore, $\rho_P$ is 0. To sum up, $\rho_P$ can be expressed as

$$\rho_P = \begin{cases} \frac{\binom{j}{N_{T,O} \cdot R_O}}{\binom{n_{T,i}}{N_{T,O} \cdot R_O}}, & \text{if } N_{T,O} \cdot R_O \leq j \\ 0, & \text{otherwise.} \end{cases} \quad (10)$$

In our static game model, we can derive Theorem 2 and guarantee that PPTS operates without any untruthful behaviors of the offloadees if a suitable incentive and redundancy ratio are decided based on Theorem 2. The proof of Theorem 2 can be found in Appendix B.

*Theorem 2:* In our static game, each player processes all the offloaded tasks properly if the appropriate incentive according to the redundancy ratio $R_O$ is provided. That is, $A = n_{T,i}$ is the best response strategy for player $i$ if the unit incentive $\alpha$ is larger than $\frac{(1-\omega)}{\omega} \cdot \frac{E_P \cdot (n_{T,i} - N_{T,O} \cdot R_O - 1) \cdot (n_{T,i} - 1)!}{n_{T,i}! - (n_{T,i} - N_{T,O} \cdot R_O)! (N_{T,O} \cdot R_O + 1)!}$.

## VI. EVALUATION RESULTS

For performance evaluation, we compare the proposed scheme, PPTS, with that of the following three schemes: 1) OPTIMAL, which selects offloadees and the number of tasks to each offloadee by solving the INLP problem; 2) RANDOM, which selects offloadees randomly; and 3) FAST, which selects offloadees having good channel conditions and the highest computing powers. Since the objective of this paper is to minimize the task completion time while preserving the privacy of the task owner at a sufficient level, the task completion time and privacy metrics (i.e., $\delta$ and $\gamma$) are used as performance measures of PPTS. To measure the privacy metrics, we define the data privacy leakage level as $\max_{x_i} \frac{n_{T,i} \cdot x_i}{N_{T,O}}$ and the usage pattern privacy leakage level as $\max_{x_i} N_{P,i} \cdot x_i$. This is because the data privacy leakage level increases as large parts of the original tasks are offloaded to a specific offloadee, and the usage pattern privacy leakage level increases as the offloading attempts are repeatedly performed to the same offloadee. Note that the task completion time consists of the transmission time from the task owner to offloadees and the execution time at the offloadees.

The default parameter settings are as follows. $N_{T,O}$ and $N_{T,O} \cdot R_O$ are set to 7 and 1, respectively. $\zeta_D$ is 2.5 Mb and $\zeta_C$ is 1 Gigacycle. $P_{C,i}$ of the offloadees are uniformly distributed in [3, 8] GHz [32], and the D2D range is 20 m. Furthermore, we have the channel bandwidth $W = 10MHz$, the noise power $I = 5 \times 10^{-5}$, the target BER $g = 10^{-3}$, and the transmission power $\epsilon_i = 0.1\ W$. The distances between the task owner and mobile devices are uniformly set in [0, 20]. $\eta_i$ is $1.5 \times 10^{-1}$ for good channel conditions and $1 \times 10^{-1}$ for bad conditions, most of which are in accordance with the real measurement in [2], [20]. We conducted simulations 1000 times in order to obtain reliable results.

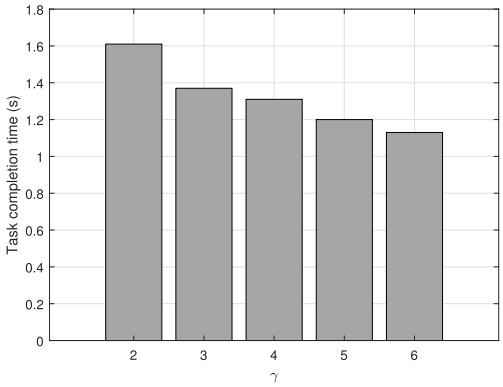



**FIGURE 4.** Effect of $\gamma$.

### A. EFFECT OF $\gamma$

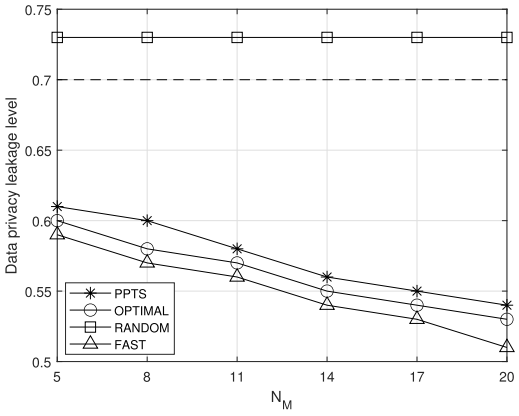
Fig. 4 shows the task completion time according to the desired level of usage pattern privacy, $\gamma$, when $N_O$, $N_M$ and $\delta$ are set to 2, 10 and 0.7, respectively.[7] Considering the mobility of the mobile devices, 10% of the mobile devices are replaced by new mobile devices for each offloading process. As shown in Fig. 4, the task completion time decreases as $\gamma$ increases. This is because as $\gamma$ increases, the number of mobile devices satisfying the usage pattern privacy leakage level also increases, which, in turn, increases the probability of having mobile devices with good channel conditions and high computing powers. Consequently, the task completion time can be reduced by increasing $\gamma$.

### B. EFFECT OF $N_M$

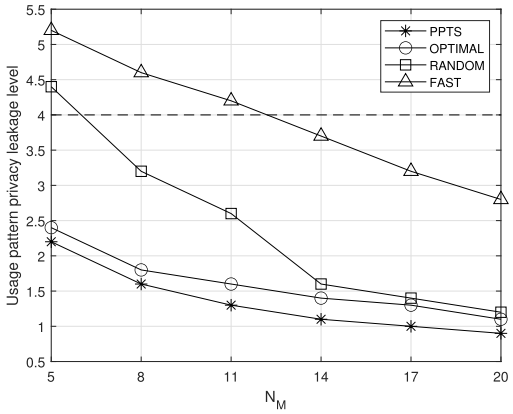
Fig. 5 shows the effect of the number of mobile devices in the offloading process, $N_M$. The dotted lines in Fig. 5(a) and Fig. 5(b) represent $\delta$ and $\gamma$, respectively, which are set

[6]Note that since the task results are considerably smaller in size than the tasks [26], the energy consumed to transmit the task results can be neglected.

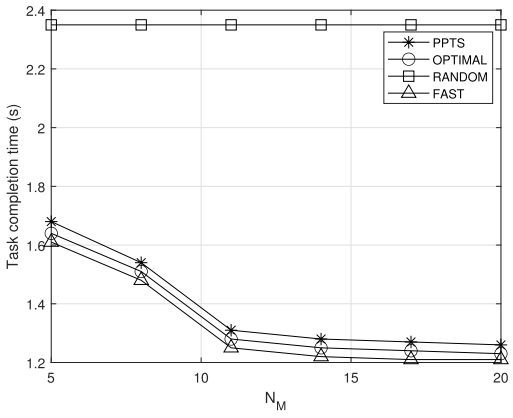
[7]The pre-defined privacy level can be set by the task owner according to the desired privacy level and the task type.




(a) Data privacy leakage level

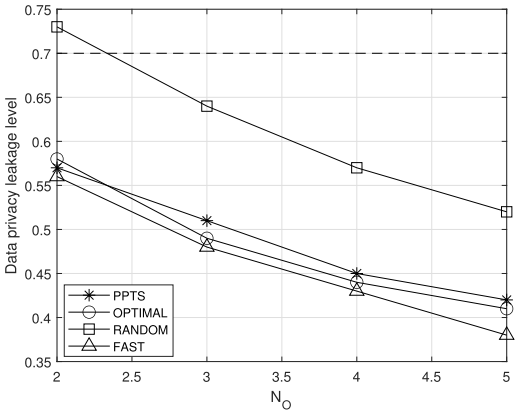



(b) Usage pattern privacy leakage level

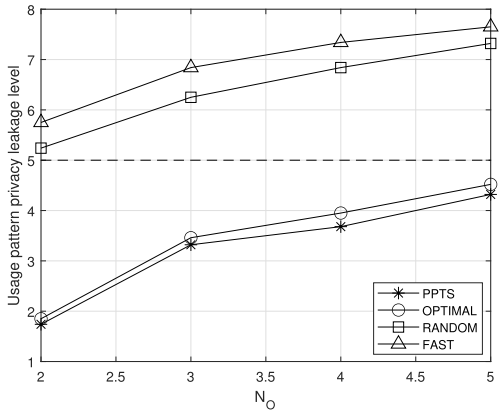



(c) Task completion time

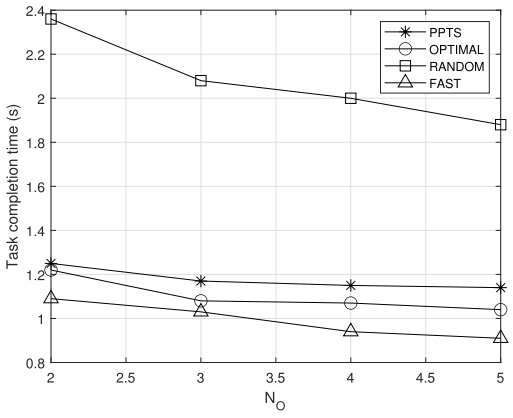
**FIGURE 5.** Effect of $N_M$.




(a) Data privacy leakage level




(b) Usage pattern privacy leakage level




(c) Task completion time

**FIGURE 6.** Effect of $N_O$.

to 0.7 and 4, respectively. Moreover, $N_O$ and the ratio of substituted mobile devices in the offloading process are set to 2 and 0.2, respectively. As shown in Fig. 5, PPTS can preserve the privacy of the task owner at a sufficient level and achieve comparable performance to OPTIMAL in terms of the task completion time. This is because PPTS selects offloadees considering their channel conditions among the mobile devices that satisfy the privacy leakage level and offloads the tasks in proportion to their computing powers. Fig. 5(c) shows that PPTS and OPTIMAL can reduce the task completion time by increasing $N_M$. This is because, when $N_M$ increases, the number of offloading mobile devices satisfying the privacy level also increases.

Furthermore, from Fig. 5(c), it can be found that the task completion time of FAST is the lowest among the compared schemes. However, FAST cannot guarantee the required privacy level for offloading since it considers only the channel conditions and computing powers of the offloadees.

### C. EFFECT OF $N_O$

Fig. 6 shows the effect of the number of offloadees $N_O$. Similarly, the dotted lines in Fig. 6(a) and Fig. 6(b) are $\delta$ and $\gamma$, which are set to 0.7 and 5, respectively. $N_M$ and the ratio of substituted mobile devices in the offloading process are set to 10 and 0.2, respectively. As shown in Fig. 6, all schemes can reduce the task completion time by increasing $N_O$. This is because as $N_O$ increases, the number of offloaded tasks to each offloadee decreases. Additionally, Fig. 6(c), shows that when $N_O$ increases from 2 to 5, PPTS can still achieve a better performance than RANDOM, whereas its gain relative to that of RANDOM decreases slightly with an increase in $N_O$. This can be explained as follows. Initially, the mobile devices with better channel conditions and higher computing powers are actively selected as offloadees in PPTS. However, as time progresses, previously selected mobile devices need to be replaced by devices with worse channel conditions and lower computing powers in order to satisfy the privacy level requirement. In contrast, RANDOM is not affected by the privacy level requirement; thus, its task completion time is evidently reduced as $N_O$ increases.

## VII. CONCLUSION

In this paper, we proposed a privacy-preserving and trustworthy D2D offloading scheme (PPTS) to prevent the privacy of the task owner and guarantee the task results during device-to-device (D2D) offloading. PPTS consists of two steps 1) a privacy-preserving offloading step and 2) a blockchain-based verification step. PPTS achieves an improved task completion time, while preserving a prespecified level of privacy of the task owner. The incentive and redundancy ratio analysis show how to determine the appropriate redundancy ratio and incentive to make offloadees cooperative. Furthermore, the evaluation results demonstrated that PPTS could guarantee a sufficient level of the privacy of the task owner and provide reliable offloading results with a reduced task completion time compared with that of a random scheme in which offloadees were randomly selected. As our future work, we plan to develop an improved offloading scheme considering the mobility of mobile devices as well as the privacy of the task owner.

## APPENDIX A
## PROOF OF THEOREM 1

The complexity of **Algorithm 1** is $O\left(\frac{N_M!}{N_O!(N_M-N_O)!}\right)$. Therefore, to prove that the complexity of **Algorithm 1** is bounded to $O({N_M}^{N_O})$, we need to prove the inequality

$$O\left(\frac{N_M!}{N_O!(N_M-N_O)!}\right) \leq O({N_M}^{N_O}). \tag{A.1}$$

The left-hand side of (A.1) can be rearranged as

$$\frac{\prod_{k=N_M-N_O+1}^{N_M} k}{N_O!}. \tag{A.2}$$

$\prod_{k=N_M-N_O+1}^{N_M} k$ is smaller than ${N_M}^{N_O}$. Therefore, $\frac{N_M!}{N_O!(N_M-N_O)!} \leq {N_M}^{N_O}$.

Furthermore, the number of offloadees is a system parameter (i.e., constant value). Therefore, **Algorithm 1** has polynomial time complexity.

## APPENDIX B
## PROOF OF THEOREM 2

To prove **Theorem 2**, we need to show that action $A = n_{T,i}$ is a dominant strategy for the players since the dominant strategy equilibrium is a Nash equilibrium. That is, if the utility for action $A = n_{T,i}$ is the largest, action $A = n_{T,i}$ is the dominant strategy for the players. Therefore, the following inequality should be satisfied to prove **Theorem 2**:

$$u_i(A = n_{T,i}) - u_i(A = j\ (0 < j < n_{T,i})) > 0. \tag{B.1}$$

Because $\rho_P$ is calculated as $\frac{(n_{T,i}-N_{T,O}\cdot R_O)!j!}{(j-N_{T,O}\cdot R_O)!\cdot n_{T,i}!}$ (see page 6), the left-hand side of (B.1) can be rearranged as

$$\alpha - \frac{(1-\omega)\cdot E_P\cdot(n_{T,i}-j)\cdot(j-N_{T,O}\cdot R_O)!(n_{T,i}-1)!}{\omega\cdot((j-N_{T,O}\cdot R_O)!\cdot n_{T,i}! - (n_{T,i}-N_{T,O}\cdot R_O)!j!)}. \tag{B.2}$$

The proof of (B.1) can be provided by showing that the minimum value of (B.2) is larger than 0. The value of (B.2) is minimum when $j$ is minimum and the minimum value of $j$ is $N_{T,O}\cdot R_O+1$. Therefore, let $j$ be replaced by $(N_{T,O}\cdot R_O+1)$. Then (B.2) can be rearranged as

$$\alpha - \frac{(1-\omega)\cdot E_P\cdot(n_{T,i}-N_{T,O}\cdot R_O-1)\cdot(n_{T,i}-1)!}{\omega\cdot(n_{T,i}! - (n_{T,i}-N_{T,O}\cdot R_O)!(N_{T,O}\cdot R_O+1)!)}. \tag{B.3}$$

It is sufficient to show that if $\alpha$ is larger than $\frac{(1-\omega)}{\omega}\cdot\frac{E_P\cdot(n_{T,i}-N_{T,O}\cdot R_O-1)\cdot(n_{T,i}-1)!}{n_{T,i}!-(n_{T,i}-N_{T,O}\cdot R_O)!(N_{T,O}\cdot R_O+1)!}$, then the utility for action $A = n_{T,i}$ is larger than that of any other action. Therefore, action $A = n_{T,i}$ is the best strategy for every player if $\alpha$ is larger than $\frac{(1-\omega)}{\omega}\cdot\frac{E_P\cdot(n_{T,i}-N_{T,O}\cdot R_O-1)\cdot(n_{T,i}-1)!}{n_{T,i}!-(n_{T,i}-N_{T,O}\cdot R_O)!(N_{T,O}\cdot R_O+1)!}$.

## REFERENCES

[1] Y. Li and W. Wang, "Can mobile cloudlets support mobile applications?" in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Toronto, ON, Canada, Apr. 2014, pp. 1060–1068.

[2] L. Pu, X. Chen, J. Xu, and X. Fu, "D2D fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted D2D collaboration," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3887–3901, Dec. 2016.

[3] G. Hu, Y. Jia, and Z. Chen, "Multi-user computation offloading with D2D for mobile edge computing," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Abu Dhabi, UAE, Dec. 2018, pp. 1–6.

[4] D. Chatzopoulos, M. Ahmadi, S. Kosta, and P. Hui, "FlopCoin: A cryptocurrency for computation offloading," *IEEE Trans. Mobile Comput.*, vol. 17, no. 5, pp. 1062–1075, May 2018.

[5] D. Wu, F. Wang, X. Cao, and J. Xu, "Joint communication and computation optimization for wireless powered mobile edge computing with D2D offloading," *J. Commun. Inf. Netw.*, vol. 4, no. 4, pp. 72–86, Dec. 2019.

[6] N. Lalwani, V. Mehta, and S. N. Merchant, "Efficient resource allocation for crowd-cloud assisted D2D computation offloading," in *Proc. 16th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Las Vegas, NV, USA, Jan. 2019, pp. 1–2.

[7] M. Mehrabi, D. You, V. Latzko, H. Salah, M. Reisslein, and F. H. P. Fitzek, "Device-enhanced MEC: Multi-access edge computing (MEC) aided by end device computation and caching: A survey," *IEEE Access*, vol. 7, pp. 166079–166108, Nov. 2019.
[8] J. Xie, Y. Jia, Z. Chen, Z. Nan, and L. Liang, "D2D computation offloading optimization for precedence-constrained tasks in information-centric IoT," *IEEE Access*, vol. 7, pp. 94888–94898, Jul. 2019.
[9] Y. Lan, X. Wang, D. Wang, Z. Liu, and Y. Zhang, "Task caching, offloading, and resource allocation in D2D-aided fog computing networks," *IEEE Access*, vol. 7, pp. 104876–104891, Jul. 2019.
[10] M. Sun, X. Xu, X. Tao, and P. Zhang, "Large-scale user-assisted multi-task online offloading for latency reduction in D2D-enabled heterogeneous networks," *IEEE Trans. Netw. Sci. Eng.*, early access, Mar. 9, 2020, doi: 10.1109/TNSE.2020.2979511.
[11] W. Chen, C.-T. Lea, and K. Li, "Dynamic resource allocation in ad-hoc mobile cloud computing," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, San Francisco, CA, USA, Mar. 2017, pp. 1–6.
[12] T. Shi, M. Yang, X. Li, Q. Lei, and Y. Jiang, "An energy-efficient scheduling scheme for time-constrained tasks in local mobile clouds," *Pervas. Mobile Comput.*, vol. 27, pp. 90–105, Apr. 2016.
[13] S. Guo, J. Liu, Y. Yang, B. Xiao, and Z. Li, "Energy-efficient dynamic computation offloading and cooperative task scheduling in mobile cloud computing," *IEEE Trans. Mobile Comput.*, vol. 18, no. 2, pp. 319–333, Feb. 2019.
[14] F. Restuccia and S. K. Das, "FIDES: A trust-based framework for secure user incentivization in participatory sensing," in *Proc. IEEE Int. Symp. World Wireless, Mobile Multimedia Netw.*, Sydney, NSW, Australia, Jun. 2014, pp. 1–10.
[15] J. Ni, K. Zhang, X. Lin, Q. Xia, and X. S. Shen, "Privacy-preserving mobile crowdsensing for located-based applications," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, May 2017, pp. 1–6.
[16] X. He, J. Liu, R. Jin, and H. Dai, "Privacy-aware offloading in mobile-edge computing," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Singapore, Dec. 2017, pp. 1–6.
[17] M. I. Ashraf, M. Bennis, C. Perfecto, and W. Saad, "Dynamic proximity-aware resource allocation in vehicle-to-vehicle (V2V) communications," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Washington, DC, USA, Dec. 2016, pp. 1–6.
[18] S. Mumtaz, K. M. S. Huq, J. Rodriguez, and V. Frascolla, "Energy-efficient interference management in LTE-D2D communication," *IET Signal Process.*, vol. 10, no. 3, pp. 197–202, May 2016.
[19] Y. Pan, M. Chen, Z. Yang, N. Huang, and M. Shikh-Bahaei, "Energy-efficient NOMA-based mobile edge computing offloading," *IEEE Commun. Lett.*, vol. 23, no. 2, pp. 310–313, Feb. 2019.
[20] S. Yu, B. Dab, Z. Movahedi, R. Langar, and L. Wang, "A socially-aware hybrid computation offloading framework for multi-access edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 6, pp. 1247–1259, Jun. 2020.
[21] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," 2019, *arXiv:1911.02417*. [Online]. Available: http://arxiv.org/abs/1911.02417
[22] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," 2019, *arXiv:1909.07972*. [Online]. Available: http://arxiv.org/abs/1909.07972
[23] Y. He, H. Li, X. Cheng, Y. Liu, C. Yang, and L. Sun, "A blockchain based truthful incentive mechanism for distributed P2P applications," *IEEE Access*, vol. 6, pp. 27324–27335, Apr. 2018.
[24] M. Andrychowicz, S. Dziembowski, D. Malinowski, and L. Mazurek, "Secure multiparty computations on bitcoin," in *Proc. IEEE Symp. Secur. Privacy*, San Jose, CA, USA, May 2014, pp. 443–458.
[25] I. Bentov and R. Kumaresan, "How to use bitcoin to design fair protocols," in *Proc. CRYPTO*, Berkeley, CA, USA, Aug. 2014, pp. 421–439.
[26] Q.-V. Pham, T. Leanh, N. H. Tran, B. J. Park, and C. S. Hong, "Decentralized computation offloading and resource allocation for mobile-edge computing: A matching game approach," *IEEE Access*, vol. 6, pp. 75868–75885, Nov. 2018.
[27] Z. Xu, X. Liu, G. Jiang, and B. Tang, "A time-efficient data offloading method with privacy preservation for intelligent sensors in edge computing," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 1, p. 236, Oct. 2019.
[28] *Complexity of Combination Algorithm*. Accessed: Oct. 5, 2020. [Online]. Available: https://stackoverflow.com/questions/24643367/whats-time-complexity-of-this-algorithm-for-finding-all-combinations
[29] P. Baptiste, M. Chrobak, and C. Durr, "Polynomial time algorithms for minimum energy scheduling," 2009, *arXiv:0908.3505*. [Online]. Available: http://arxiv.org/abs/0908.3505
[30] T. Wang, Y. Sun, L. Song, and Z. Han, "Social data offloading in D2D-enhanced cellular networks by network formation games," *IEEE Trans. Wireless Commun.*, vol. 14, no. 12, pp. 7004–7015, Dec. 2015.
[31] C. Gao, H. Zhang, X. Chen, Y. Li, D. Jin, and S. Chen, "Impact of selfishness in device-to-device communication underlying cellular networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 9338–9349, Oct. 2017.
[32] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7944–7956, Aug. 2019.

**HOSUNG BAEK** received the B.S. degree from Korea University, Seoul, South Korea, in 2015, where he is currently pursuing the M.S. and Ph.D. integrated degrees with the School of Electrical Engineering. His research interests include 5G networks, programmable dataplane language, mobile cloud computing, and SDN/NFV.

**HANEUL KO** (Member, IEEE) received the B.S. and Ph.D. degrees from the School of Electrical Engineering, Korea University, Seoul, South Korea, in 2011 and 2016, respectively. From 2016 to 2017, he was a Postdoctoral Fellow in mobile network and communications with Korea University. From 2017 to 2018, he was with the Smart Quantum Communication Research Center, Korea University. He was a Visiting Postdoctoral Fellow with the University of British Columbia, Vancouver, BC, Canada. He is currently an Assistant Professor with the Department of Computer Convergence Software, Korea University, Sejong, South Korea. His research interests include 5G networks, network automation, mobile cloud computing, SDN/NFV, and future Internet.

**SANGHEON PACK** (Senior Member, IEEE) received the B.S. and Ph.D. degrees in computer engineering from Seoul National University, Seoul, South Korea, in 2000 and 2005, respectively. From 2005 to 2006, he was a Postdoctoral Fellow with the Broadband Communications Research Group, University of Waterloo, Waterloo, ON, Canada. He joined the Faculty of Korea University, Seoul, in 2007, where he is currently a Professor with the School of Electrical Engineering. His research interests include softwarized networking (SDN/NFV), 5G/6G mobile core networks, mobile edge computing/programmable data plane, and vehicular networking. He was a recipient of the IEEE/Institute of Electronics and Information Engineers (IEIE) Joint Award for IT Young Engineers Award 2017, the Korean Institute of Information Scientists and Engineers (KIISE) Young Information Scientist Award 2017, the Korea University TechnoComplex (KUTC) Crimson Professor 2015, the Korean Institute of Communications and Information Sciences (KICS) Haedong Young Scholar Award 2013, the LG Yonam Foundation Overseas Research Professor Program, in 2012, and the IEEE ComSoc APB Outstanding Young Researcher Award, in 2009. He served as a TPC Vice-Chair for information systems of the IEEE WCNC 2020, a Track Chair of the IEEE VTC 2020-Fall/2010-Fall and the IEEE CCNC 2019, a TPC Chair of the IEEE/IEIE ICCE-Asia 2018/2020, EAI Qshine 2016, and ICOIN 2020, a Publication Co-Chair of the IEEE INFOCOM 2014 and ACM MobiHoc 2015, a Symposium Chair of the IEEE WCSP 2013, a TPC Vice-Chair of the ICOIN 2013, and a Publicity Co-Chair of the IEEE SECON 2012. He is an Editor of the IEEE Internet of Things (IoT) Journal, the *Journal of Communications Networks* (JCN), and *IET Communications*. He is a Guest Editor of the IEEE Transactions on Emerging Topics in Computing (TETC) and the IEEE Transactions on Network Science and Engineering (TNSE).