Security and energy-aware collaborative task offloading in D2D communication

Zhongjin Li[a], Haiyang Hu[a*], Hua Hu[a,b], Binbin Huang[a], Jidong Ge[c], Victor Chang[d]

[a] School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China

[b] School of Information Science and Engineering, Hangzhou Normal University, Hangzhou, China

[c] State Key Laboratory for Novel Software Technology, Software Institute, Nanjing University, Nanjing, China

[d] Artificial Intelligence and Information Systems Research Group, School of Computing & Digital Technologies, Teesside University, Middlesbrough, UK

Abstract

Device-to-device (D2D) communication technique is used to establish direct links among mobile devices (MDs) to reduce communication delay and increase network capacity over the underlying wireless networks. Existing D2D schemes for task offloading focus on system throughput, energy consumption, and delay without considering data security. This paper proposes a Security and Energy-aware Collaborative Task Offloading for D2D communication (Sec2D). Specifically, we first build a novel security model, in terms of the number of CPU cores, CPU frequency, and data size, for measuring the security workload on heterogeneous MDs. Then, we formulate the collaborative task offloading problem that minimizes the time-average delay and energy consumption of MDs while ensuring data security. In order to meet this goal, the Lyapunov optimization framework is applied to implement online decision-making. Two solutions, greedy approach and optimal approach, with different time complexities, are proposed to deal with the generated mixed-integer linear programming (MILP) problem. The theoretical proofs demonstrate that Sec2D follows a $[O(1/V), O(V)]$ energy-delay tradeoff. Simulation results show that Sec2D can guarantee both data security and system stability in the collaborative D2D communication environment.

Keywords: D2D communication; security modeling; collaborative task offloading; Lyapunov optimization; mixed-integer linear programming (MILP)

1. Introduction

Mobile applications (e.g., interactive online gaming, virtual reality (VR), augmented reality (AR), and high-definition movies, etc.) have gained extensive popularity attributed to cellular technologies and mobile services. Moreover, up-to-date mobile devices (MDs) like Apple XS, Samsung S9, and Nokia X7 are equipped with powerful CPU chips. Thus, the current MDs have the potential to process computation-intensive applications [1], [2], [3], [4]. However, only running mobile applications at a single MD is still constrained due to limited computational resources and battery capacity, resulting in longer delay and more energy consumption [5], [6].

One solution to reduce energy consumption and decrease latency is offloading tasks to the evolved NodeB (eNB) of mobile edge computing [7], [8], [9], [10]. Nevertheless, Cisco recently reports that approximately 11.6 billion MDs will be anticipated by 2020 [11]. In order to meet the vast and ever-increasing service and resource demand, additional eNB infrastructures are required, leading to more deployment and maintenance costs.

The other promising solution is device-to-device (D2D) communication, which can establish direct links among physically proximate MDs without the interaction of eNB [12], [13]. It has been viewed as a very useful technology in the mobile network due to the facts that 1) a large number of MDs residing in cellular network environment will stress the availability of network resource, and thus, D2D communication can effectively alleviate the network congestion and the edge node's workload; 2)

D2D communication can make full use of the spectrum resource and improve the spectral efficiency; 3) collaborative task offloading in D2D paradigm can be implemented by computational resource sharing among MDs to achieve lower delay and less energy consumption [1], [4], [14].

Apart from energy consumption and latency, security is another top-prioritized concern in D2D communication [15], [16], [17]. D2D communication is built upon hybrid architecture with a variety of network nodes. So, it faces various malicious threats and attacks, which are also faced by the cellular network. For instance, the report by 3GPP Security Workgroup (SA3) indicates that the D2D environment suffers from six categories of security threats [18]. Thus, efficient security solutions, such as confidentiality and integrity services, are needed to guarantee data exchange security among MDs. However, adding any service to mobile applications will inevitably incur extra time, leading to longer service time as well as more energy consumption. In general, how to balance delay, energy, and security imposes a key challenge in D2D communication.

This paper investigates a Security and Energy-aware Collaborative Task Offloading for D2D communication (Sec2D). The major contributions are summarized as follows:

- We establish a novel security model based on the number of CPU cores and CPU frequency, and data size, which is utilized to measure the execution time of different security services on heterogeneous MDs. According to the security model, the security workload sharping is also devised to calculate the security workload for different security service levels.
- We design a collaborative task offloading architecture with two workload queues, including a front-end queue and a back-end queue. The former is applied to queue the arrived tasks and offload tasks; the latter is used for receiving the offloaded tasks and processing tasks. Moreover, before routing tasks from one MD to another, the corresponding security services should be adopted up to the security demand.
- We formulate the collaborative task offloading problem to minimize the average delay and energy consumption with the demand for security services. In order to address this problem, the Lyapunov optimization framework is applied to implement online decisions making. Additionally, a low-complexity greedy approach and a branch and cut (BAC) based optimal approach is developed to tackle generated mixed-integer linear programming (MILP) problem.
- We give the result of theoretical proof that the Sec2D algorithm meets a $[O(1/V), O(V)]$ energy-delay tradeoff. Simulation results on algorithm comparison, the impact of system parameters, and the impact of security parameters show that Sec2D indeed can guarantee data security and keep the system stability in collaborative D2D communication scenario.

The rest of this paper is organized as follows: Section 2 summarizes the related work about task offloading and security scheduling. Section 3 builds our novel security model on different MD parameters. Section 4 presents the system model and formulates our problem. Section 5 introduces the algorithm design of Sec2D and gives the performance analysis. Section 6 first gives the experimental setting and then elaborates the simulation results. Finally, Section 7 concludes this paper and plans for future work.

2. Related work

Existing MDs are equipped with high-performance multi-core CPUs, which have the potential to implement the collaborative task offloading. For example, Pu et al. [1] propose a mobile task offloading framework, D2D fogging, to minimize the energy consumption of the entire system by incorporating the incentive mechanism. Hong et al. [2] introduce a task outsourcing and scheduling method to allow the cooperative processing of MDs' computation-intensive tasks in a D2D network. Based on the Lyapunov approach, Feng et al. [14] suggest a computation offloading and resource sharing strategy to optimize the average cost of MDs while meeting the constraint of resource budget in the D2D-enabled scenario. Liu et al. [4] propose an edge resource pooling (ERP) framework to pool and share the computational resource for D2D task offloading. By using the adaptive genetic algorithm, Huynh et al. [10] present an energy consumption optimization (ECO) algorithm for D2D multimedia communications. Lai et al. [19] propose a resource sharing and power control approach, called DRAPC, for a pure D2D communication scenario. DRAPC is to allow MDs to process tasks for other MDs, which is implemented based on the graph-coloring solution to improve throughput, fairness, and resource utilization. Fan et al. [20] develop a task offloading approach in a D2D-assisted fog computing environment, where the tasks can be processed by other MDs or offloaded to fog nodes. In order to maximize the total profit, the problems of the offloading scheme, resource allocation, and MD selection are formulated and solved by game theory. Saleem et al. [21] propose a joint resource and task offloading scheme, called JPORA, for minimizing the task execution delay according to multiple condition constraints, where the spectrum sharing is applied to exploit the computation resources of other MDs. Li et al. [22] utilize the reinforcement learning technique to offload tasks in D2D-MEC (mobile edge computing) environment where the objective is to minimize the long-term cost. However, the above offloading schemes focus on energy consumption, delay, and cost, neglecting the data security problem in the D2D environment.

Security is another critical concern when transmitting private data in the wireless environment. In [23], [24], [25], [26], a series of key management approaches are applied to ensure secure connections among MDs and eNBs. The key management, carried out by public key infrastructure, utilizes a shared key or public key to verify the identities of nodes (eNB and MDs). For instance, Shen et al. [24] investigate a key agreement protocol to establish a secure, shared, low cost and secret overhead key without requiring any prior knowledge. Hsu and Lee [25] propose two authentication-based key exchange protocols for network-covered and network-absent D2D communication cases. The group anonymity with authentication is applied for the former the k-anonymity secret handshake scheme is used for the latter case. Zhang et al. [26] present a data sharing protocol to guarantee data security. Specifically, a joint of digital signature and authentication service is used to ensure the entity authentication, data authority, and data integrity. However, lacking essential key infrastructure, it is impracticable to use these key management mechanisms for guaranteeing data security in pure D2D communication. Further, due to the mobility and discrepancy of MDs, the scheme of the pre-allocating secret key in MD is also infeasible when implementing frequent resource sharing and task offloading [18].

Without any key infrastructure, the security services can be employed to form a complete security protection mechanism, which has been extensively applied in the cluster [27], grid computing [28], cloud computing [29], [30], [31], real-time embedded system [32] and mobile edge computing [33]. For example, Xie and Qin [27] devise a security-aware scheduling approach for real-time application processing in the cluster. Song et al. [28] propose three risk-resilient job scheduling mechanisms to offer security guaranteeing in the grid. Li et al. [30] develop a security scheduling method for scientific workflow execution in the cloud. By task replication mechanism, Chen et al. [31] suggest a workflow task scheduling framework for protecting the security of output data in clouds. Elgendy et al. [6] propose a resource allocation algorithm for computation offloading in the MEC environment, where a security technique is applied to protect sensitive data. Further, the authors [34] put forward an efficient and secure task offloading scheme for a multi-user MEC environment. The image and video compression techniques are utilized to reduce time and energy consumption. Then, to guarantee data security, the AES cryptographic algorithm is used to avoid cyberattacks. However, the above two security offloading schemes just consider one cryptographic algorithm to cope with the cyber-attacks. So, they cannot be applied in the scenarios of multiple kinds of attacks and multiple security service requirements. Huang et al. [33] introduce a security-aware task offloading for service workflows to optimize the MD's energy consumption, where multiple security services are considered for various malicious attacks. However, this security model is derived from the perspective of physical servers, which can no longer be applied in a D2D communication environment because the CPU architectures of the physical server and MD are different. Besides the above symmetric cryptographic algorithms, many asymmetric cryptographic approaches also have been developed. For example, Mansour et al. [35] design a Chinese Remainder Theorem (CRT) based RSA encryption technique, which overcomes key exchange and management problems. Further, a lightweight asymmetric cryptographic mechanism, called AMOUN, is proposed for multi-party communication [36]. Ametepe et al. [37] devise a hybrid cryptosystem by combining asymmetric and symmetric cryptographic algorithms to secure the data transmission in the wireless network environment. However, the above works focus on designing a cryptographic algorithm, which is orthogonal to our security services.

In a word, the above security mechanisms cannot be immediately generalized to D2D communication because of two reasons: 1) The existing security model is only based on the computational performance of the physical server, which cannot be directly available to MDs; 2) There exists a large number of heterogeneous MDs in D2D environment, and these MDs have a different number of CPU cores and processing capacities. Thus, the security model should be applied for MDs of any type to calculate the security time and workload.

3. Security Modeling

Previous works like [27], [29], [32] have built and used the security models to protect the security of data. However, these security models are established on the basis of the architecture of the processor or server, and only the time overhead of different data sizes is considered. Thus, the existing security model cannot be directly applied in D2D communication due to the following reasons: 1) the CPU architecture of the server is significantly different from that of MD, and hence even under the same condition (i.e., same CPU frequency and data size), it will cause completely different time overhead; 2) the up-to-date MDs have powerful multi-core CPU and the dynamic voltage and frequency scaling (DVFS) function to adjust working frequency [38]. Thus, security time should also be measured under various CPU cores and frequencies.

We utilize confidentiality and integrity services to build our security model. Seven symmetric cryptographic algorithms (such as ARC4, RC5, Blowfish, AES, IDEA, DES, RC2 [27], [30], [33]) and three asymmetric cryptographic algorithms (i.e., RSA, ECC (Elliptic curve cryptography), and DH (Diffie-Hellman) [36], [35], [37]) are used to fulfill the confidentiality service. Similarly, six hash functions, i.e., MD4, MD5, Tiger, SHA1, RIPEMD, SHA256, are provided to realize the integrity service [27], [39]. In order to measure the time overhead, all the security services are implemented by Java. Moreover, we use Huawei Honor8 as the mobile device to build the security model, which is equipped with 4GB GPU memory and 8 CPU cores (4 low-performance cores and 4 high-performance cores). In detail, the working frequencies of Huawei Honor8 on the low-performance core are $F_L = \{0.8, 1.2, 1.5, 1.8\}$ GHz, and the working frequencies on the high-performance core are

$F_H = F_L + \{2.0, 2.3\}$ GHz.

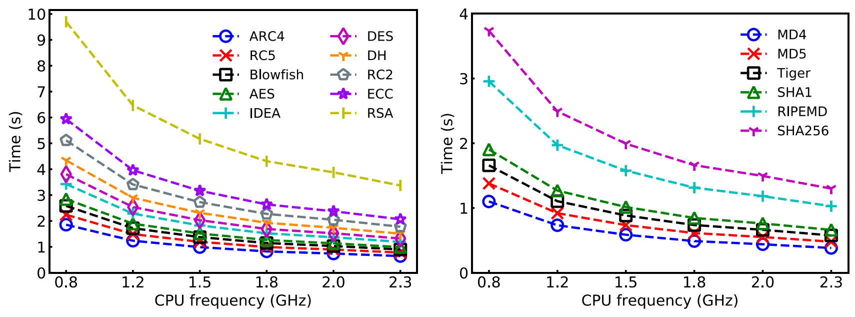
In a word, security modeling in the context of D2D communication remains an open issue. Next, we first construct three security time models on the number of CPU cores N, CPU frequency F, and data size D, respectively. Then, the security level model is built for quantifying the protection capacity of each security service. Finally, we convert security time to security workload for our collaborative task offloading.

3.1 Time overhead model

We investigate the time impact of security service on different CPU frequencies $F \in F_H = \{0.8, 1.2, 1.5, 1.8, 2.0, 2.3\}$ GHz, where we fix the number of CPU cores at $N = 1$ (high-performance core) and the size of data at $D = 1$ MB. Fig. 1 shows that the execution time of all the security services decreases with the CPU frequency. The time overhead is inversely proportional to the value of CPU frequency, i.e., doubled CPU frequency makes half of the security time.

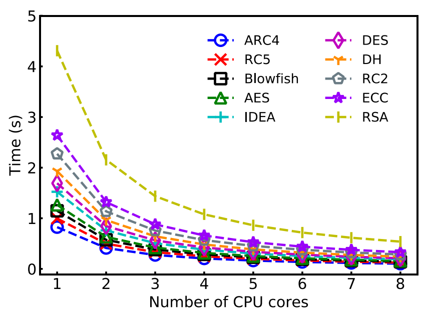
We also examine the impact of execution time with the different number of CPU cores, where $F = 1.8$ GHz and $D = 1$ MB. For processing data with multiple CPU cores, we first divide the data into equally sized data blocks, and the number of blocks is equal to that of CPU cores. For example, if we use the CPU with 2 cores to encrypt the 1 MB data, the data is first divided into two equal parts, i.e., each data block is 0.5 MB. Thus, multi-core CPUs can be completely exploited. The experimental results are shown in Fig. 2. We can see that the time overhead decreases with the increase of the number of CPU cores in all the cases. Furthermore, the service time is inversely proportional to the number of CPU cores, and if the number of CPU cores is doubled, the time is halved, which is similar to the behavior of Fig. 1.

Fig. 3 shows the results of time overhead on different data sizes, i.e., varying D from 0.1 to 1 MB with the increment of 0.1 MB, where we keep the other parameters fixed, $F = 1.8$ GHz and $N = 1$. It can be observed that the time overheads of all security services grow linearly with D. That means the service time is in direct proportion to the size of data, i.e., the time required is double when the size of data is double.

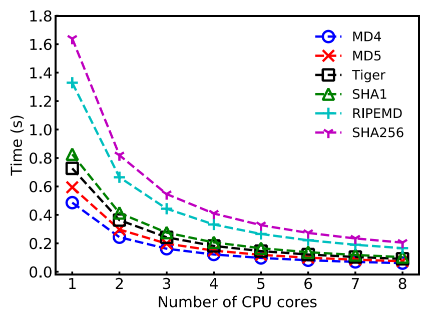


(a) Time overhead for confidentiality service (b) Time overhead for integrity service

Fig. 1 The impact of F on time overhead.

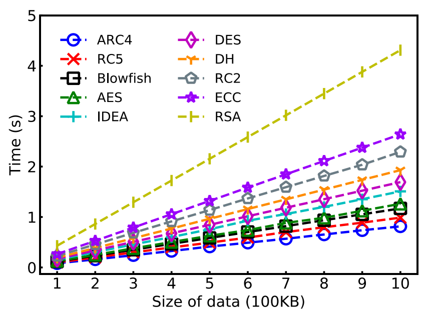


(a) Time overhead for confidentiality service

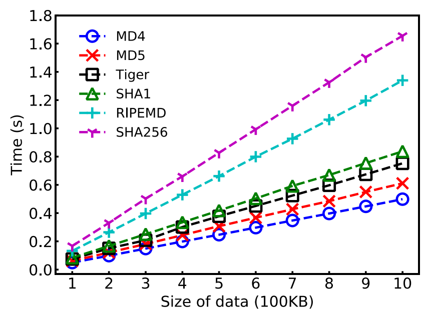


(b) Time overhead for integrity service

Fig. 2 The impact of the N on time overhead.



(a) Time overhead for confidentiality service



(b) Time overhead for integrity service

Fig. 3 The impact of D on time overhead.

3.2 Security level model

Like [27], [33], we allocate the security levels for all the cryptographic algorithms and hash functions to represent the capacity of data protection, shown in Table 1 and Table 2, respectively. Note that the time overheads are collected under $F = 1.8$ GHz, $N = 1$, and $D = 1$ MB. For simplicity, we denote cd and ig as the confidentiality and integrity services, respectively.

We say that the security level is in accordance with its service time, which means the algorithm or function with a longer execution time has a higher security level. For instance, for cryptographic algorithms, we give the highest level of 1.00 to the RC2 algorithm (see Table 1). We then define the other algorithms' security levels by

$$sl_{cd}(k) = T_{cd}^{sc}(k)/2.27, \tag{1}$$

where $T_{cd}^{sc}(k)$ is the security time of k-th confidentiality service.

Using the same method, regarding hash functions, the slowest but strongest, SHA256, is assigned the highest security level 1.00. Thus, the security levels of the rest functions are represented by

$$sl_{ig}(k) = T_{ig}^{sc}(k)/1.64, \tag{2}$$

where $T_{ig}^{sc}(k)$ is the security time of k-th integrity service. All the security levels of hash functions can be seen in Table 2.

Table 1. Cryptographic algorithms for confidentiality service.

k	Cryptographic algorithms	$T_{cd}^{sc}(k)$ (s)	$sl_{cd}(k)$
1	ARC4	0.83	0.19
2	RC5	0.99	0.23
3	Blowfish	1.15	0.27

4	AES	1.27	0.30
5	IDEA	1.52	0.35
6	DES	1.62	0.38
7	DH	1.93	0.45
8	RC2	2.27	0.53
9	ECC	2.64	0.61
10	RSA	4.31	1.00

Table 2. Hash functions for integrity service.

k	Hash Functions	$T_{ig}^{sc}(k)$ (s)	$sl_{ig}(k)$
1	MD4	0.48	0.29
2	MD5	0.59	0.36
3	Tiger	0.73	0.45
4	SHA1	0.82	0.50
5	RIPEMD	1.29	0.79
6	SHA256	1.64	1.00

3.3 Security workload shaping

As discussed above, the security time is highly correlated with CPU frequency F, the number of CPU cores N and data size D. Similar to [40], [41], we make an assumption that F is equal to the processing capacity C. Then, for the security level $sl_v(k), v \in \{cd, ig\}$, if a task with data size D to be protected, the security time on the single-core CPU with processing capacity C is represented by $T_v^{sc}(k)$. In this case, the security workload, in the form 'security time $\times$ processing capacity', can be obtained by

$$W_v^{sc}(k) = D \cdot T_v^{sc}(k)C, v \in \{cd, ig\}, \tag{3}$$

which also indicates that the security workload is highly correlated with the data size D. For example, if $D = 0.1$ MB, $F = C = 1.8$ GHz and $N = 1$, then $T_{cd}^{sc}(k) = 2.27$s and $sl_{cd}(k) = 1.0$. Thus, the security workload is $W_{cd}^{sc}(k) = 0.1 \times 2.27 \times 1.8$ GHz $\cdot$ s $= 0.41$ GHz $\cdot$ s. Thus, the total security workload of a task can be expressed as

$$W^{sc} = \sum_{v\in\{cd,ig\}} W_v^{sc}(k). \tag{4}$$

However, in D2D communication, it is impossible for all the MDs to have the same computational performance (i.e., the number of CPU cores and working frequency). In such a heterogeneous environment, each MD will perform different security times. According to the security workload computed above, for an MD with processing capacity C, the security time on this MD is computed by

$$T^{sc} = W^{sc}/C. \tag{5}$$

Based on the Eqs. (3)-(5), we can derive the security time on any MDs under any data size D and processing capacity C.

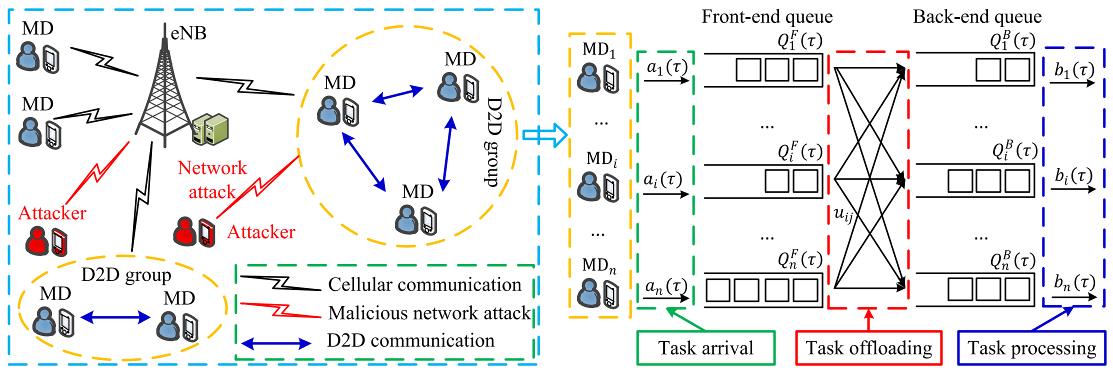
4. System Model and Problem Formulation

Fig. 4 shows the D2D communication scenario and collaborative task offloading architecture. The left part depicts the wireless network scenario with cellular MDs, eNB, D2D groups, and malicious attackers. The MD can communicate with eNB by cellular network and with other MDs by D2D communication. This paper mainly concentrates on the D2D groups, consisting of a set of D2D devices. In D2D communication, the signal strength varies over time due to MDs' mobility. Let $R_{ij}(\tau)$ denote the data transmission rate (i.e., signal strength) from MD_i to MD_j in time slot τ, where τ is the system time. Let $SNR_{ij}(\tau) = P_i^{tx} G_{ij}(\tau)/\sigma_n^2$ denote the signal-to-noise ratio (SNR), where P_i^{tx} represents the transmission power, σ_n^2 is the Gaussian white noise power, and $G_{ij}(\tau) = \gamma[d_{ij}(\tau)]^{-\theta}$ denotes the channel gain of this D2D link, where γ and θ are the path-loss constant and exponent, respectively, and $d_{ij}(\tau)$ is the path distance. Then, $R_{ij}(\tau)$ can be calculated by [1], [3]

$$R_{ij}(\tau) = B_i \log_2[1 + SNR_{ij}(\tau)], \tag{6}$$

where B_i is the bandwidth of MD_i.

The right part of Fig. 4 elaborates on the collaborative task offloading architecture, where the D2D group has n MDs. Let $\mathrm{MD} = \{\mathrm{MD}_1, \dots, \mathrm{MD}_i, \dots, \mathrm{MD}_n\}$ denote the MD set and $\mathcal{N} = \{1,2,\dots,n\}$ denote the index set of MDs. It is worth pointing that each MD_i contains a front-end queue $Q_i^F(\tau)$ and a back-end queue $Q_i^B(\tau)$ [42], respectively. The former is applied to receive and offload tasks, and the latter is utilized to accommodate and process tasks. At each time slot τ, every MD makes decisions on computing resource usage, task offloading, and workload processing. Below, we introduce three models and formulate our optimization problem.



(a) The D2D communication scenario. (b) The collaborative task offloading architecture.

Fig. 4. The D2D communication scenario and collaborative task offloading architecture.

4.1 Workload queueing model

At time slot τ, the number of tasks arriving at $Q_i^F(\tau)$ is represented by $a_i(\tau)$. Let $\boldsymbol{A}(\boldsymbol{\tau}) = (a_1(\tau), \dots, a_i(\tau), \dots, a_n(\tau))$ denote the task arrival vector, and $a_i(\tau) \in \mathcal{A} = \{0,1,\dots,A_{max}\}$. Suppose the task arrival process $\{a_i(\tau)\}$ is independent and identically distributed (i.i.d.) with $\mathbb{E}\{\boldsymbol{A}(\tau)\} = \boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_i, \dots, \lambda_n)$. We make an assumption that each task has a constant data size D and execution workload W^{ex}. This assumption is usually justified in practice, as we can split a task into many equally sized small tasks. Let $u_{ij}(\tau)$ denote the number of tasks routed from MD_i to MD_j. In addition, $u_{ij}(\tau)$ must be taken from the feasible set, i.e., $u_{ij}(\tau) \in U$ and $u_{ij}(\tau) \le U_{max}, \forall i,j \in \mathcal{N}$. Let $\boldsymbol{B}(\tau) = (b_1(\tau), \dots, b_i(\tau), \dots, b_n(\tau))$ denote the workload processed by each MD in time slot τ, and $b_i(\tau) \in \mathcal{B} = [0, B_{max}]$. Then, the evolution equations of two MD's queues are expressed as

$$Q_i^F(\tau+1) = \max\{Q_i^F(\tau) - \textstyle\sum_{j\in\mathcal{N}} u_{ij}(\tau) W^{ex}, 0\} + a_i(\tau) W^{ex}, \tag{7}$$

$$Q_i^B(\tau+1) = \max\{Q_i^B(\tau) - b_i(\tau), 0\} + \textstyle\sum_{j\in\mathcal{N}} u_{ji}(\tau) W^{ex} + \sum_{j\in\mathcal{N}} u_{ji}(\tau) W_{ij}^{sc}. \tag{8}$$

where Eqs. (7) and (8) are the two workload queues of front-queue and backend-queue, respectively. Note that $u_{ij}(\tau) W^{ex}$ and $a_i(\tau) W^{ex}$ are the execution workloads, and $u_{ij}(\tau) W_{ij}^{sc}$ is the security workload. In addition, $u_{ij}(\tau)$ and $a_i(\tau)$ are two integers because they represent the number of tasks. In contrast, due to the workload can be processed continuously, $b_i(\tau)$ is a continuous variable. Let $W_{ii}^{sc} = 0, \forall i \in \mathcal{N}$, which means if a task offloaded to itself, security service is not required.

The task offloading process with security services is shown in Fig. 5. Before transmitting the task from the front-end queue of MD_i to the back-end queue of MD_j, an encryption algorithm (E) and a hash function (H) are successively executed on the data of this task. After receiving this task by MD_j, it will be first put into the back-end queue and served by the first-come-first-serve (FCFS) rule. When this task begins to be processed, it will first be hashed again to verify its integrity (denoted as H as well) and then decrypted (denoted as DE). After that, MD_j starts to process this task. So, MD_j needs to process the security workload and execution workload for an offloaded task. The above analysis explains why Eq. (8) needs the additional security workload in the back-end queue.

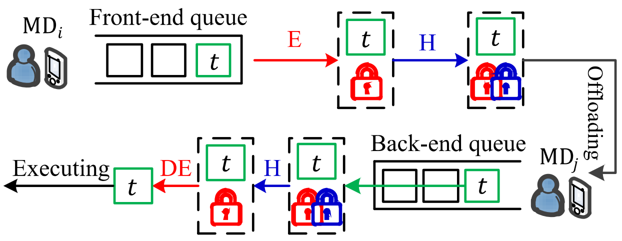


Fig. 5. The task offloading process with security services.

4.2 Security risk model

We quantitatively measure the task's risk probability on different security levels based on the introduced security model. The risk probability follows a Poisson distribution, as its coefficient can be considered as the number of network attacks. Thus, the risk probability with v-th security service follows the exponential distribution [28], [29], [30], which is given by

$$Pr(sl_v) = \begin{cases} 0, & if\ \ sl_v \geq sd_v \\ 1 - \exp\left(-\eta_v(sd_v - sl_v)\right), & otherwise, \end{cases} \tag{9}$$

where sd_v represents security demand. The total risk probability of a task is represented by considering all the security services, and hence we have

$$Pr = 1 - \prod_{v\in\{cd,ig\}}(1 - Pr(sl_v)). \tag{10}$$

From Eq. (9) we find that a task is expected to be secure when satisfying the security assurance condition $sl_v \geq sd_v$. Besides, the security demand sd_v is correlated with the risk coefficient η_v. For example, if $\eta_v = 0$, that means the MD does not experience any malicious attack. Thus, we do not need to deploy any security service to protect the data, and the corresponding security demand sd_v is 0. If η_v becomes larger, the higher sd_v is required to guarantee data security. So, we devise the following formula to describe the relationship between η_v and sd_v.

$$sd_v = \Gamma(\eta_v) = 1 - \exp(-\beta\eta_v), \tag{11}$$

where β is an experience value. To guarantee the security, the security demand of sending data from MD_i to MD_j must satisfy the following equation.

$$sd_v^i = \max\{sd_v^i, sd_v^j\}. \tag{12}$$

Eq. (12) can be explained as follows: in D2D communication, each MD will face various network attacks. Consequently, the security demand should be the maximum one between two MDs. Thus, the security level should be just larger or equal to sd_v^i, i.e.,

$$sl_v^i = \arg\min(sl_v(k) - sd_v^i), sl_v(k) - sd_v^i \geq 0. \tag{13}$$

Then, the risk probability $Pr(sl_v^i) = 0, v \in \{cd, ig\}$, indicating data transmission will not experience any security risk.

4.3 Energy consumption model

Typically, a MD_i is equipped with N_i heterogeneous CPU cores. Suppose $N_{i,L}$ and $N_{i,H}$ are the number of low-performance and high-performance cores, respectively, and $N_{i,L} + N_{i,H} = N_i$. The computing power of this MD_i can be expressed as [7], [9]

$$P_i^{cx}(\tau) = \alpha_{i,L} \sum_{l=1}^{N_{i,L}} F_{i,l}^3(\tau) + \alpha_{i,H} \sum_{h=1}^{N_{i,H}} F_{i,h}^3(\tau), \tag{14}$$

where $\alpha_{i,L}$ and $\alpha_{i,H}$ are two constants related to the chip architecture. The state-of-the-art MD adopts an advanced DVFS technique, allowing automatic CPU frequency regulating. In practice, the level of CPU frequency is bounded by a minimum value and a maximum value, i.e., $F_{i,l}(\tau) \in F_{i,L} = [F_{i,L}^{min}, F_{i,L}^{max}]$ and $F_{i,h}(\tau) \in F_{i,H} = [F_{i,H}^{min}, F_{i,H}^{max}]$. However, in [42], a conclusion has been justified that homogeneous CPU cores should have the same operating frequency to reach the optimal power consumption. So, Eq. (14) can be rewritten as

$$P_i^{cx}(\tau) = \alpha_{i,L} N_{i,L} F_{i,l}^3(\tau) + \alpha_{i,H} N_{i,H} F_{i,h}^3(\tau). \tag{15}$$

The computational performance is also controlled by the CPU frequency. Besides, we have the relation between computation capacity and CPU frequency, i.e., $C = F$, and then we have

$$C_i(\tau) = N_{i,L} F_{i,l}(\tau) + N_{i,H} F_{i,h}(\tau). \tag{16}$$

According to the queueing model in Section 3.2, in time slot τ, $\sum_{j \in \mathcal{N}} u_{ij}(\tau)$ tasks will be offloaded from MD_i to MD_j. Before transmitting these tasks, security services should be performed. Then, the security time is computed by,

$$T_i^{sc}(\tau) = [\sum_{j \in \mathcal{N}} u_{ij}(\tau)\, W_{ij}^{sc}]/C_i(\tau), \tag{17}$$

Note that the security workload is symmetric, i.e., $W_{ij}^{sc} = W_{ji}^{sc}$. After implementing the security services, $\sum_{j \in \mathcal{N}} u_{ij}(\tau)$ tasks will be transmitted with absolute security, and the transmission time is expressed as

$$T_i^{tx}(\tau) = \sum_{j \in \mathcal{N}} u_{ij}(\tau) D / R_{ij}(\tau). \tag{18}$$

where D is the data size of task t_i. As the data do not need to be transmitted to the MD itself, we assume that $R_{ii}(\tau) = \infty, \forall i \in \mathcal{N}$. So, the energy consumption of data transmission is obtained by

$$E_i^{tx}(\tau) = P_i^{tx} T_i^{tx}(\tau). \tag{19}$$

In addition, MD_i needs to process the amount of $b_i(\tau)$ workload from the back-end queue. The corresponding execution time is calculated by

$$T_i^{ex}(\tau) = b_i(\tau)/C_i(\tau). \tag{20}$$

Then, the CPU energy consumption, formed "power $\times$ time", can be derived by,

$$E_i^{cx}(\tau) = P_i^{cx}(\tau)[T_i^{sc}(\tau) + T_i^{ex}(\tau)]. \tag{21}$$

Finally, the total energy consumption of MD_i, in time slot τ, is the sum of CPU and data transmission energy consumption, which is expressed as

$$E_i(\tau) = E_i^{cx}(\tau) + E_i^{tx}(\tau). \tag{22}$$

Moreover, receiving data also causes energy consumption, but the power of receiving data is much less than that of data transmission and CPU working. So, we neglect it in this energy consumption model.

4.4 Problem formulation

This paper interests in minimizing the long-term energy consumption for the D2D group under the data security constraint, which is represented by

$$\bar{E} = \limsup_{t \to \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[E(\tau)], \tag{23}$$

where $E(\tau) = \sum_{i \in \mathcal{N}} E_i(\tau)$. Furthermore, all the MDs' task queues must be stable in the average time sense, i.e., keeping the stability of the D2D system, which is expressed by

$$\bar{Q} = \limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{i=1}^{n}\mathbb{E}[Q_i^F(\tau) + Q_i^B(\tau)]. \tag{24}$$

By Little's law [43], the average delay is in direct proportion to $\bar{Q}$. Therefore, we treat $\bar{Q}$ as the average delay in this work. Thus, our optimization problem **P** is given as follows:

$$\textbf{P}:\ \min\ \bar{E}, \tag{25a}$$
$$\text{s.t. } \bar{Q} < \infty, \tag{25b}$$
$$T_i^{sc}(\tau) + T_i^{ex}(\tau) \le T_{slot},\ i \in \mathcal{N}, \tag{25c}$$
$$T_i^{sc}(\tau) + T_i^{tx}(\tau) \le T_{slot}, i \in \mathcal{N}, \tag{25d}$$
$$\sum_{j\in\mathcal{N}} u_{ij}(\tau) W^{ex} \le Q_i^F(\tau), i \in \mathcal{N} \tag{25e}$$
$$b_i(\tau) \le Q_i^B(\tau), i \in \mathcal{N} \tag{25f}$$
$$F_{i,l}(\tau) \in F_{i,L}, F_{i,h}(\tau) \in F_{i,H}, i \in \mathcal{N}, \tag{25g}$$
$$u_{ij}(\tau) \in \mathbb{Z}, b_i(\tau) \in \mathbb{R}, \forall i,j \in \mathcal{N}. \tag{25h}$$

where T_{slot} the duration of each time slot.

Directly solving **P** requires a set of a priori information, such as $R_{ij}(\tau)$ and $a_i(\tau)$; however, which cannot be obtained in advance. In the next section, we apply the Lyapunov optimization framework, which can make online decisions only according to current state information, to address our proposed problem.

5. Algorithm design and performance analysis

This section mainly focuses on algorithm design and performance analysis. The Sec2D algorithm is proposed to solve the problem P by invoking the Lyapunov optimization framework [44], [45]. More specifically, we first highlight the entire Sec2D algorithm framework. Then, two approaches, a low-complexity greedy approach and an optimal approach are introduced for solving the MILP problem deduced from the Lyapunov optimization technique. Finally, we characterize the performance of Sec2D.

5.1 Algorithm design

The Lyapunov function $L(\tau)$, representing a scalar metric of queue backlog, is defined as

$$L(\tau) = \frac{1}{2}\sum_{i\in\mathcal{N}}\{[Q_i^F(\tau)]^2 + [Q_i^B(\tau)]^2\}, \tag{26}$$

where $L(\tau)$ evolves over the time slot τ. By Eq. (26), we define the Lyapunov drift $\Delta(\tau)$ as below:

$$\Delta(\tau) = \mathbb{E}\{L(\tau+1) - L(\tau)|\boldsymbol{Q}(\tau)\}, \tag{27}$$

where $\boldsymbol{Q}(\tau) = (Q_i^F(\tau), Q_i^B(\tau), i = \mathcal{N})$ is the vector denoting all the workload queues at time slot τ. Then, the drift-plus-penalty of the Lyapunov framework is expressed as $\Delta(\tau) + V\mathbb{E}\{E(\tau)|\boldsymbol{Q}(\tau)\}$, where V is the control parameter which indicates how much the system emphasizes the energy consumption. The explanation of the drift-plus-penalty term is made as follows: we want to keep the system stable by pushing $\Delta(\tau)$ toward a lower value; we also want to make $\mathbb{E}\{E(\tau)|\boldsymbol{Q}(\tau)\}$ smaller to incur less energy consumption. So, the purpose is to minimize $\Delta(\tau) + V\mathbb{E}\{E(\tau)|\boldsymbol{Q}(\tau)\}$, and the corresponding upper bound is given in the following lemma.

Lemma 1. For any possible actions under queue stability condition that can be implemented at time slot τ, we have

$$\Delta(\tau) + V\mathbb{E}\{E(\tau)|\boldsymbol{Q}(\tau)\} \le \Omega + V\mathbb{E}\{E(\tau)|\boldsymbol{Q}(\tau)\}$$
$$+\mathbb{E}\{\sum_{i\in\mathcal{N}}[Q_i^F(\tau)(a_i(\tau)W^{ex} - \sum_{j\in\mathcal{N}} u_{ij}(\tau) W^{ex})]|\boldsymbol{Q}(\tau)\}$$

$$+\mathbb{E}\{\textstyle\sum_{j\in\mathcal{N}} Q_j^B(\tau)[\sum_{i\in\mathcal{N}} u_{ij}(\tau)\,(W^{ex}+W_{ij}^{sc})-b_j(\tau)]\,|\boldsymbol{Q}(\tau)\}, \tag{28}$$

where $\Omega = n\{[nU_{max}W^{sc}]^2 + [A_{max}W^{sc}]^2 + n^2U_{max}^2[W^{sc}+W_{max}^{sc}]^2 + B_{max}^2\}/2$.

Proof. See Appendix A.

The main purpose of the Lyapunov technique is to minimize $\Delta(\tau)+V\mathbb{E}(E(\tau)|\boldsymbol{Q}(\tau))$ at each time slot τ. By doing so, the queue length can be kept at a low level. Meanwhile, the energy consumption of all MDs can be minimized. Rather than directly minimize the drift-plus-penalty term, the Lyapunov framework seeks to minimize the bound given in the right-hand-side of Eq. (28). Thus, as stated above, instead of solving **P** directly, we only need to address its modified version **P1**.

$$\textbf{P1}:\ \min\ VE(\tau)+\textstyle\sum_{i\in\mathcal{N}} Q_i^F(\tau)[a_i(\tau)W^{ex}-\sum_{j\in\mathcal{N}} u_{ij}(\tau)\,W^{ex}]$$

$$+\textstyle\sum_{j\in\mathcal{N}} Q_j^B(\tau)[\sum_{i\in\mathcal{N}} u_{ij}(\tau)\,(W^{ex}+\mathrm{W}_{ij}^{sc})-b_j(\tau)], \tag{29a}$$

$$\text{s.t. (25c-25h).} \tag{29b}$$

P1 is the optimal objective for all the MDs in the D2D group. The following Lemma 2 makes the conclusion that **P1** can be realized by the distributed decision making at each MD_i.

Lemma 2. By the framework of opportunistically minimizing a conditional expectation, **P1** can be equivalently reformulated as,

$$\textbf{P2}:\ \min\ \textstyle\sum_{j\in\mathcal{N}}\rho_{ij}(\tau)u_{ij}(\tau)+\varphi_i(\tau)b_i(\tau), \tag{30a}$$

$$\text{s.t. (25c)-(25h),} \tag{30b}$$

where

$$\rho_{ij}(\tau) = Q_j^B(\tau)[W^{ex}+W_{ij}^{sc}]-Q_i^F(\tau)W^{ex}$$
$$+V[P_i^{cx}(\tau)W_{ij}^{sc}/C_i(\tau)+P_i^{tx}D/R_{ij}(\tau)],$$

and

$$\varphi_i(\tau) = VP_i^{cx}(\tau)/C_i(\tau)-Q_i^B(\tau).$$

Proof. See Appendix B.

We find that there exist four variables in Eq. (30a), such as $u_{ij}(\tau)$, $b_i(\tau)$, $F_{i,l}(\tau)$, and $F_{i,h}(\tau)$. For each MD_i, the numbers of $F_{i,l}(\tau)$ and $F_{i,h}(\tau)$ are finite and is a small integer. To obtain the minimum value, we first fix $F_{i,l}(\tau)$ and $F_{i,h}(\tau)$, and use $g(F_{i,l}(\tau),F_{i,h}(\tau))=\sum_{j\in\mathcal{N}}\rho_{ij}(\tau)u_{ij}(\tau)+\varphi_i(\tau)b_i(\tau)$ to represent the objective function under these CPU frequencies. Note that for the given $F_{i,l}(\tau)$ and $F_{i,h}(\tau)$, the $C_i(\tau)$ and $P_i^{cx}(\tau)$ can be determined. Thus, $\rho_{ij}(\tau)$ and $\varphi_i(\tau)$ are two constants in this case. After traversing all the combinations of $F_{i,l}(\tau)$ and $F_{i,h}(\tau)$, we can choose the minimal $g^*(F_{i,l}(\tau),F_{i,h}(\tau))$ as the optimal value. The framework of the Sec2D algorithm is shown in Algorithm 1. It is worth pointing out that $u_{ij}(\tau)$ is the integer variable and $b_i(\tau)$ is the continuous variable, minimizing $g(F_{i,l}(\tau),F_{i,h}(\tau))$ is a mixed-integer linear programming (MILP) problem.

Algorithm 1. The Sec2D framework.

```
1: At the beginning of each time slot $\tau$, each $\text{MD}_i$ first observes the queue backlog $Q_i^F(\tau)$, $Q_i^B(\tau)$, and task arrival $a_i(\tau)$, broadcasts its back-end queue information to other MDs, and then makes the online decision as follows:
2: for $(F_{i,l}(\tau),F_{i,h}(\tau))\in F_{i,L}\times F_{i,H}$ do
3:     Calculate $\rho_{ij}(\tau)$ and $\varphi_i(\tau)$
4:     Minimize $g(F_{i,l}(\tau),F_{i,h}(\tau))$
5: Update $Q_i^F(\tau+1)$ and $Q_i^B(\tau+1)$ according to Eqs. (7) and (8)
```

5.2 WO-based greedy approach

MILP plays an important role in many real-world problems, including resource sharing [46], service optimization [47], and so on. In this section, we present a greedy approach with low time complexity for MILP problem. To minimize Eq. (30a) in problem **P2**, we first discuss four cases as follows:

- Case 1: $\rho_{ij}(\tau) \geq 0, \forall j \in \mathcal{N}$, and $\varphi_i(\tau) > 0$. In this case, $u_{ij}(\tau) = 0$ and $b_i(\tau) = 0$ will obtain the minimum value $g(F_{i,l}(\tau), F_{i,h}(\tau)) = 0$.
- Case 2: $\rho_{ij}(\tau) \geq 0, \forall j \in \mathcal{N}$, and $\varphi_i(\tau) \leq 0$. We set $u_{ij}(\tau) = 0$, and the problem is to minimize $g(F_{i,l}(\tau), F_{i,h}(\tau)) = \varphi_i(\tau) b_i(\tau)$ with the optimal solution $b_i(\tau) = \min\{C_i(\tau)T_{slot}, Q_i^B(\tau)\}$. It is noted that, when $\varphi_i(\tau) = 0$, for the sake of using already allocated computing resources, we process workload as much as possible. This will not affect the value of the objective function but decrease the length of the back-end queue.
- Case 3: $\rho_{ij}(\tau) < 0, \exists j \in \mathcal{N}$, and $\varphi_i(\tau) \geq 0$. We set $b_i(\tau) = 0$, and our problem is to minimize $g(F_{i,l}(\tau), F_{i,h}(\tau)) = \sum_{j\in\mathcal{N}} \rho_{ij}(\tau) u_{ij}(\tau)$.
- Case 4: $\rho_{ij}(\tau) < 0, \exists j \in \mathcal{N}$, and $\varphi_i(\tau) < 0$. In this case, the objective function is the original one $g(F_{i,l}(\tau), F_{i,h}(\tau)) = \sum_{j\in\mathcal{N}} \rho_{ij}(\tau) u_{ij}(\tau) + \varphi_i(\tau) b_i(\tau)$.

The above analysis process is shown in Algorithm 2, and we give the summary as follows: 1) when $Q_i^F(\tau)$ is too long, or $Q_j^B(\tau)$ is short (leading to $\rho_{ij}(\tau) < 0$), MD$_i$ will migrate more tasks to $Q_j^B(\tau)$ to maintain the system stability; 2) when $Q_j^B(\tau)$ is congested, or the computing resource is sufficiently enough (leading to $\varphi_i(\tau) < 0$), MD$_i$ will process more tasks as much as possible to maintain the stability of D2D system.

Algorithm 2. WO-based Greedy Approach.

```
1: for $j$ in $\mathcal{N}$ do
2:     Calculate $\rho_{ij}(\tau)$
3: Calculate $\varphi_i(\tau)$
4: Case 1:
5:     $u_{ij}(\tau) \leftarrow 0$
6:     $b_i(\tau) \leftarrow 0$
7: Case 2:
8:     $u_{ij}(\tau) \leftarrow 0$
9:     $b_i(\tau) \leftarrow \min\{C_i(\tau)T_{slot}, Q_i^B(\tau)\}$
10: Case 3:
11:     Minimize $\sum_{j\in\mathcal{N}} \rho_{ij}(\tau) u_{ij}(\tau)$
12: Case 4:
13:     Minimize $\sum_{j\in\mathcal{N}} \rho_{ij}(\tau) u_{ij}(\tau) + \varphi_i(\tau) b_i(\tau)$
```

Moreover, to solve the objective function in Cases 3 and 4, we develop a weight ordering (WO) based greedy method, which has low time complexity. Taking Case 4, $g(F_{i,l}(\tau), F_{i,h}(\tau)) = \sum_{j\in\mathcal{N}} \rho_{ij}(\tau) u_{ij}(\tau) + \varphi_i(\tau) b_i(\tau)$, as an example, let $\mathcal{N}_i(\tau)$ denote the set of MDs whose $\rho_{ij}(\tau) < 0, \forall j \in \mathcal{N}$, and $m = |\mathcal{N}_i(\tau)| \leq n$. Let $w_j(\tau)$ be the weight of parameters $\rho_{ij}(\tau)$ and $\varphi_i(\tau)$, which is defined as

$$w_j(\tau) = \begin{cases} \rho_{ij}(\tau)\frac{C_i(\tau)}{W_{ij}^{sc}}\frac{R_{ij}(\tau)}{D}, \forall j \in \mathcal{N}_i(\tau) \\ \varphi_i(\tau)C_i(\tau), \qquad j = m+1, \end{cases} \tag{31}$$

where the former and latter of the right-hand-side of Eq. (31) are defined for $u_{ij}(\tau)$ and $b_i(\tau)$, respectively. For the former case, $C_i(\tau)/W_{ij}^{sc}$ and $R_{ij}(\tau)/D$ can be seen as the ratio of occupied time when offloading one task from MD$_i$ to MD$_j$. When $\rho_{ij}(\tau) < 0$, we hope that offloading the same number of tasks, i.e., $u_{ij}(\tau)$, will take up less time. Thus, MD$_i$ can route more tasks to other MDs at a one-time slot, resulting in lower $g(F_{i,l}(\tau), F_{i,h}(\tau))$. So, the lower $w_j(\tau)$, the smaller $g(F_{i,l}(\tau), F_{i,h}(\tau))$. The same explanation can be applied to the latter case.

The pseudo-code of the WO mechanism is outlined in Algorithm 3. Let T_{re1} and T_{re2} denote the remaining time for constraints Eqs. (25c) and (25d), and let Q_{re}^F denote the remaining workload for the front-end queue. We first compute all the weights and put them into the set S. Then, all the weights in set S are sorted in increasing order based on their value. Next, we take the first item from S, which has the lowest value and get the weight index. If index $j \in \mathcal{N}_i(\tau)$, we need to allocate the number of tasks for $u_{ij}(\tau)$. Due to the constraints T_{re1}, T_{re2} and Q_{re}^F, we get $u_{ij}(\tau) = \min\{\left\lfloor\frac{T_{re1}C_i(\tau)}{W_{ij}^{sc}}\right\rfloor, \left\lfloor\frac{T_{re2}C_i(\tau)R_{ij}(\tau)}{W_{ij}^{sc}R_{ij}(\tau)+C_i(\tau)D}\right\rfloor, Q_{re}^F/W^{ex}\}$. If index $j = m+1$, the workload $b_i(\tau)$ should be determined, i.e., $b_i(\tau) = \min\{T_{re1}C_i(\tau), Q_i^B(\tau)\}$ based on the time constraint and back-end queue. Also, T_{re1}, T_{re2}, Q_{re}^F and S are needed to be updated. For Case 3, computing $w_{m+1}(\tau)$ and allocating $b_i(\tau)$ are not required, and hence the corresponding parts can be removed from Algorithm 3.

By incorporating the WO mechanism into Algorithm 2, our proposed greedy approach can be implemented. Unlike the optimal solution approach, the worst time complexity of this greedy approach is $O(n\log n)$, resulting from sorting the weights in set S. So, the worst time complexity of the Sec2D algorithm is $O(N_L N_H n\log n)$, where $N_L N_H$ is the maximum number of groups $(F_{i,l}(\tau), F_{i,h}(\tau))$ for all MDs.

Algorithm 3. WO Mechanism (for Case 4).

```
1: $S \leftarrow \emptyset$, $T_{re1}, T_{re2} \leftarrow T_{slot}$, $Q_{re}^F \leftarrow Q_i^F(\tau)$
2: for $j$ in $\mathcal{N}_i(\tau)$ do
3:     $w_j(\tau) \leftarrow \rho_{ij}(\tau)\frac{C_i(\tau)}{W_{ij}^{sc}}\frac{R_{ij}(\tau)}{D}$
4:     $S \leftarrow S + \{w_j(\tau)\}$
5: $w_{m+1}(\tau) \leftarrow \varphi_i(\tau)C_i(\tau)$
6: $S \leftarrow S + \{w_{m+1}(\tau)\}$
7: Sort weights in set $S$ in the increasing order
8: while $S \neq \emptyset$, $T_{re1} > 0$ and $T_{re2} > 0$ do
9:     Get the index $j$ of the first weight in $S$
10:    if $j \in \mathcal{N}_i(\tau)$ then
11:        $u_{ij}(\tau) \leftarrow \min\{\left\lfloor\frac{T_{re1}C_i(\tau)}{W_{ij}^{sc}}\right\rfloor, \left\lfloor\frac{T_{re2}C_i(\tau)R_{ij}(\tau)}{W_{ij}^{sc}R_{ij}(\tau)+C_i(\tau)D}\right\rfloor, Q_{re}^F/W\}$
12:        $T_{re1} \leftarrow T_{re1} - u_{ij}(\tau)W_{ij}^{sc}/C_i(\tau)$
13:        $T_{re2} \leftarrow T_{re2} - u_{ij}(\tau)W_{ij}^{sc}/C_i(\tau) - u_{ij}(\tau)D/R_{ij}(\tau)$
14:        $Q_{re}^F \leftarrow Q_{re}^F - u_{ij}(\tau)W^{ex}$
```

```
15:     else if $j = m + 1$ then
16:         $b_i(\tau) \leftarrow \min\{T_{re1} C_i(\tau), Q_i^B(\tau)\}$
17:         $T_{re1} \leftarrow T_{re1} - b_i(\tau)/C_i(\tau)$
18:     $S \leftarrow S - \{w_j(\tau)\}$
```

5.3 BAC-based optimal approach

Let $c = [\rho_{i1}(\tau), \ldots, \rho_{im}(\tau), \varphi_i(\tau)]^{\mathrm{T}}$ represent the coefficient vector and $x = [u_{i1}(\tau), \ldots, u_{im}(\tau), b_i(\tau)]^{\mathrm{T}}$ be the variable vector. Note that only the variables with $\rho_{ij}(\tau) < 0$ are considered that can reduce the dimensions of our problem. Also, if $\varphi_i(\tau) \geq 0$, $\varphi_i(\tau)$ and $b_i(\tau)$ are removed from c and x . Then, $g(F_{i,l}(\tau), F_{i,h}(\tau)) = c^{\mathrm{T}}x$. Similarly, let $\pi = [T_{slot}, T_{slot}, Q_i^F(\tau), Q_i^B(\tau)]$, and

$$\Pi = \begin{bmatrix} \frac{W_{i1}^{sc}}{C_i(\tau)} & \cdots & \frac{W_{im}^{sc}}{C_i(\tau)} & \frac{1}{C_i(\tau)} \\ \frac{W_{i1}^{sc}}{C_i(\tau)} + \frac{D}{R_{i1}(\tau)} & \cdots & \frac{W_{im}^{sc}}{C_i(\tau)} + \frac{D}{R_{im}(\tau)} & 0 \\ W^{ex} & \cdots & W^{ex} & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix}$$

We can rewrite our problem **P2** into the general MILP problem, namely

$$\min \ g(F_{i,l}(\tau), F_{i,h}(\tau)) = c^{\mathrm{T}}x, \tag{32a}$$

$$\text{s.t. } \Pi x \leq \pi, \tag{32b}$$

$$x \in \mathbb{Z}^m \times \mathbb{R}. \tag{32c}$$

Ideally, MILP can be optimally solved by exhaustive search methods, e.g., branch and bound [48], [49] and cutting plane [50], [51]. Nevertheless, it is difficult to apply to our problem. This is because solving this MILP for MD_i results in unacceptable running time. This motivates us to devise a more efficient solution.

The merits and demerits of cutting plane and branch and bound are as follows: cutting plane is fast but unreliable; while branch and bound is reliable but slow. Recently, the branch and cut (BAC) algorithm combines the advantages from these two methods and improves the defects, i.e.; we can solve the MILP problems by taking some cutting planes into the branch and bound process [52], [53]. It has proven to be a very successful approach to solving a wide variety of real-world problems.

The BAC-based optimal approach is outlined in Algorithm 4 [54]. It mainly has seven steps. In the *Initialization* step, upper bound $\bar{z}$ and lower bound $\underline{z}$ are initialized. Particularly, $\bar{z}$ and x^* are set by our greedy approach, which is very useful for the *Pruning* step, as the solution of our greedy algorithm maybe not optimal but feasible. Let Φ denote the set of non-integer variables in the branch and cut tree, and $\Phi = \{\mathrm{P}^0\}$, where P^0 is the original MILP problem **P2**. The *Termination* step is used to determine if the optimal solution is found. Unlike other MILP problems, our problem must have an optimal solution. In the *Node selection* step, generally, there are two search strategies, best first and depth-first, for selecting the next node (subproblem) to be processed. To minimize overall solution time, we apply the best first approach to choose the node with the best bound (lowest upper bound), which can minimize the size of the search tree. The *Relaxation* step is the most important factor for proving a solution is optimal and is applied in all the MILP/ILP problems and methods. Its implementation neglects the integer constraints on the decision variables x and solves the relaxed linear programming (LP) problem (i.e., $x \in \mathbb{R}^n$). We assume that the LP solver is available, called LP_{SOLV}. In particular, we consider this solver that is able to find the unique minimal optimal solution z and x_{LP}. Let $K = \bigcap_{i=1}^{m} K_i$ be the constraint set with each K_i, defined by $\Pi_i x \leq \pi_i$. Further reducing the number of nodes to explore with improved relaxation bounds, the *Cutting plane* step is utilized to generate valid inequalities to the LP relaxation, where h_{MIG} represents the mixed-integer Gomory (MIG) cuts [51]. The *Pruning* step is to leverage the convexity of LP relaxations to prune the enumeration tree that eliminates a problem from further consideration. Let $\text{conv}(\Lambda)$ denote the polyhedral convex hull. If $x_{LP} \in \text{conv}(\Lambda)$, then x_{LP} is the optimal solution. Finally, if our relaxed solution is not integer feasible, we must decide how to partition the search space into smaller subproblems. The strategy for doing this is called a *Branching*. Branching wisely is very important, and its most common approach is changing variable bounds. For instance, if x_i is not integer feasible, we create two problems with additional constraints, i.e., $x_i \leq \lfloor x_i \rfloor$ on one branch and $x_i \geq \lceil x_i \rceil$ on another branch. Then, we add these two subproblems to the set Φ.

Algorithm 4. BAC-based Optimal Approach

```
1: Initialization
   $\bar{z}, x^* \leftarrow$ Call WO-based greedy approach //Upper bound
   $\underline{z} \leftarrow -\infty$ //Lower bound
   $\Phi \leftarrow \{\mathrm{P}^0\}$ //Problem set
2: Termination
   if $\Phi = \emptyset$ then, the current $x^*$ is the optimal solution,
         i.e., $z^* \leftarrow \bar{z}$, and STOP
   if $\underline{z} = z = \bar{z}$ then, set $z^* \leftarrow z$, and STOP
3: Node selection
   Compute the best bound by the best first approach
   Select and delete a problem $\mathrm{P}^i$ in $\Phi$
4: Relaxation
   $z, x_{LP} \leftarrow \mathrm{LP}_{SOLV}$ $(K)$ //Solve the LP relaxation of $\mathrm{P}^i$
5: Cutting plane
   $h_{MIG} \leftarrow MIG(x_{LP})$ // Gomory cutting plane
   $K \leftarrow K \cap h_{MIG}$, and go to Relaxation
6: Pruning
   if $\underline{z} \geq \bar{z}$ then, go to Termination
   if $\underline{z} < \bar{z}$ and $x_{LP} \in \mathrm{conv}(\Lambda)$ then
      Set $\bar{z} \leftarrow z$
      Remove possibly dominated problem from $\Phi$
      Go to Termination
7: Branching
   $\mathrm{P}^{i1} \leftarrow K \cap \{x_i \leq \lfloor x_i \rfloor\}$, and $\mathrm{P}^{i2} \leftarrow K \cap \{x_i \geq \lceil x_i \rceil\}$
   $\Phi \leftarrow \Phi \cup \{\mathrm{P}^{i1}, \mathrm{P}^{i2}\}$, and go to Termination
```

5.4 Performance analysis

Lyapunov optimization method can derive a performance bound on energy-delay tradeoff. To prove this theory, we first introduce an existing lemma, Lyapunov stability [44], that if there exists a positive constant $\epsilon > 0$, such that for all time slots τ, we have

$$\Delta(\tau) \leq \Omega - \epsilon \textstyle\sum_{i \in \mathcal{N}} \mathbb{E}[Q_i^F(\tau) + Q_i^B(\tau)]. \tag{33}$$

Moreover, the following Lemma 3 makes the objective function close to the optimal solution.

Lemma 3: For any $\delta > 0$, there exists a stationary and randomized policy that chooses $u_{ij}(\tau)$, $b_i(\tau)$, $F_{i,l}(\tau)$ and $F_{i,h}(\tau)$ each time slot τ, and achieves the following results:

$$\mathbb{E}\{E(\tau)\} \le E^* + \delta, \tag{34a}$$

$$\lambda_i \le \mathbb{E}\{\sum_{j\in\mathcal{N}} u_{ij}(\tau)\} + \delta, \forall i \in \mathcal{N}, \tag{34b}$$

$$\mathbb{E}\{\sum_{j\in\mathcal{N}} u_{ji}(\tau)\ [W^{ex} + W_{ij}^{sc}]\} \le \mathbb{E}\{b_i(\tau)\} + \delta, \forall i \in \mathcal{N}. \tag{34c}$$

Proof. The proof process can be found from the Theorem 4.5 of [45].

Based on Eq. (33) and Lemma 3, we can prove the performance bounds and energy-delay tradeoff of our proposed Sec2D algorithm.

Theorem 1. Suppose that $\epsilon > 0$ and the data request arrival rate $\boldsymbol{\lambda}$ is strictly within the network capacity region $\boldsymbol{\Lambda}$. For any control parameter $V > 0$, under the Sec2D algorithm, we have

$$\bar{E} = \lim_{t\to\infty} \sup \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[E(\tau)] \le E^* + \Omega/V, \tag{35}$$

$$\bar{Q} = \lim_{t\to\infty} \sup \frac{1}{t} \sum_{\tau=0}^{t-1} \sum_{i=1}^{n} \mathbb{E}[Q_i^F(\tau) + Q_i^B(\tau)] \le (\Omega + VE^*)/\epsilon. \tag{36}$$

Proof. See Appendix C.

Theorem 1 shows the result of the energy-delay tradeoff $[O(1/V), O(V)]$. For example, as V becomes larger for any V>0, the average energy consumption decreases and approaches the optimal value E^*, whereas the average queue length increases linearly with V. On the other hand, when V becomes smaller, the average queue length is close to its lower bound Ω/ϵ, while the average energy consumption increases with the increment of Ω/V.

6. Simulation Experiments

This section first introduces the experimental parameters, performance metrics, and comparison algorithms. Then, three groups of simulation results and corresponding analysis are given. Finally, we summarize the experimental results.

6.1 Experiment setup

Suppose n MDs in D2D communication, and the distance between two MDs is a random number with maximum value $d_{max} = 200$ m. Besides, each MD's bandwidth $B = 10$ MHz, noise power $\sigma^2 = -174$ dbm/Hz, path-loss constant $\gamma = 0.01$, and path-loss exponent $\theta = 4$ [3], [7]. Also, suppose each MD has the same transmit power $P_i^{tx} = 100$ mW [55]. We consider the maximum working frequency of each CPU core of an MD is 2.0 GHz, and its computational power is 900 mW. According to $P^{cx} = \alpha F^3$, α is computed as $900/2^3 = 112.5$ mW/(GHz)^3 [1], [8]. We use the above parameter setting for all the MDs. The other parameters of heterogeneous MDs are given in Table 4. Moreover, we can adjust the CPU frequency level in the range [0.3,1] with the increment of 0.1 for low and high-performance cores by the DVFS technique.

The task arrival rate of each MD follows the Poisson distribution with mean $\lambda_i \in [1,10]$, and the size of arrived data is 10 KB with workload $W = 1$ GHz · s. For the security service, we assume that $\beta = 0.5$ and $\eta_v \in [0,5], \forall s \in \{cd, ig\}$. In addition, we set $T_{slot} = 1$ and the number of time slots $N_{slot} = 10000$, which is long enough for obtaining stable results.

The performance metrics evaluated in our experiments are as follows:

- *Average energy consumption* (*AEC*) and average queue length (*AQL*): These two metrics are computed by Eqs. (23) and (24), respectively.
- *Average running time (ART)*: It is the average execution time of an algorithm on each time slot, reflecting the time complexity, which is calculated by $\sum_{\tau=1}^{N_{slot}}\sum_{i\in\mathcal{N}}[T_i^{run}(\tau)/(N_{i,L}+N_{i,H})]/(nN_{slot})$. $T_i^{run}(\tau)/(N_{i,L}+N_{i,H})$ is the running time for MD_i at time slot τ, where the denominator $N_{i,L}+N_{i,H}$ indicates that the multiple CPU cores are used to search the optimal solution from $(F_{i,l}(\tau),F_{i,h}(\tau))$ groups simultaneously.
- *Average risk probability (ARP)*: The average risk probability is calculated by $\sum_{\tau=1}^{N_{slot}}\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}}u_{ij}(\tau)\,Pr(i,j)/\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}}u_{ij}(\tau)$, where $i\neq j$ and $Pr(i,j)$ is the total risk probability of transmitting data from MD_i to MD_j.

Also, we conduct the following comparison algorithms to compare the performance with our Sec2D algorithm.

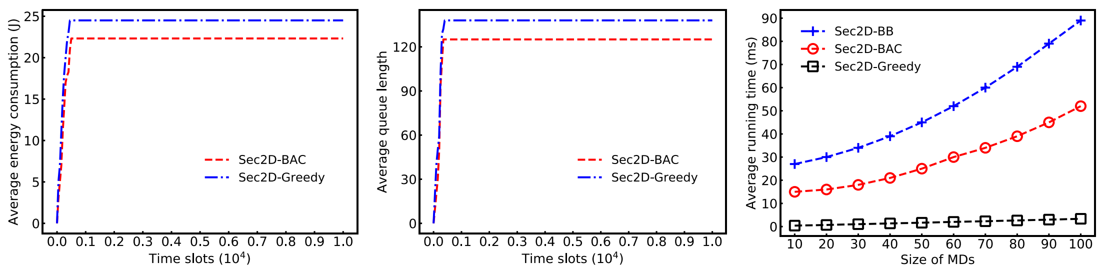
- *No security service*: This method does not adopt any security service when offloading tasks, i.e., exposing the data to malicious attacks, leading to high risk probability.
- *Local computation*: All the tasks arrived at Q_i^F are routed to Q_i^B, i.e., $u_{ii}(\tau)=a_i(\tau)$, and $u_{ij}=0$ for $i\neq j$. So, this case will not exist any cooperation for task processing among MDs.

Table 4: The parameters of heterogeneous MDs.

	MD (CPU chip)	N_L	N_H	F_L^{max} (GHz)	F_H^{max} (GHz)
1	iPhone 7 (A10)	2	2	1.05	2.34
2	iPhone X (A11)	4	2	1.6	2.38
3	iPhone XS (A12)	4	2	1.6	2.5
4	Samsung S8 (Exynos 8895)	4	4	1.7	2.5
5	Samsung S9 (Exynos 9810)	4	4	1.7	2.9
6	Samsung S10 (Exynos 9820)	4	2+2	1.9	2.3+2.7
7	Nokia X7 (Snapdragon 710)	6	2	1.7	2.2
8	Huawei Honor8 (Kirin 950)	4	4	1.8	2.3
9	Huawei p20 (Kirin 970)	4	4	1.8	2.4
10	Huawei Mate20 (Kirin 980)	4	4	1.8	2.6

6.2 Algorithms comparison

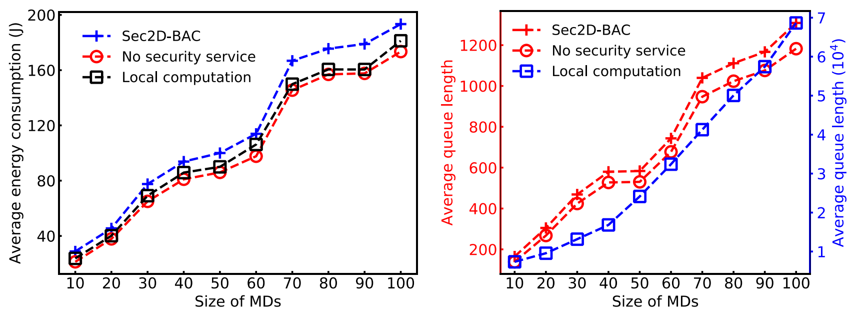
We first evaluate the performance of our proposed algorithms. Let Sec2D-BB, Sec2D-BAC and Sec2D-Greedy represent the problem **P2** solved by branch and bound, branch and cut, and greedy approaches, respectively. Note that we apply the Cplex 12.0.0 for Sec2D-BB and Sec2D-BAC. Fig. 6 shows the performance comparison on the convergences of AEC and AQL, and the ART on different n. It is observed from Figs. 6(a) and 6(b) that for Sec2D-BAC and Sec2D-Greedy, the AEC and AQL increase at the beginning and stabilize within approximately 500-time slots, where we keep $n=10$. Besides, we can see that the AEC and AQL of Sec2D-BAC are all less than that of Sec2D-Greedy, as Sec2D-Greedy is not the optimal solution approach. Fig. 6(c) shows the ARTs of all comparison algorithms. Sec2D-Greedy has the least ART. Sec2D-BAC has less ART than Sec2D-BB. This is because Sec2D-BAC has fewer variables than Sec2D-BB, the upper bound initialized by Sec2D-Greedy, and the cutting plane is integrated into the branch and bound framework, resulting in speeding up the optimal solution searching.



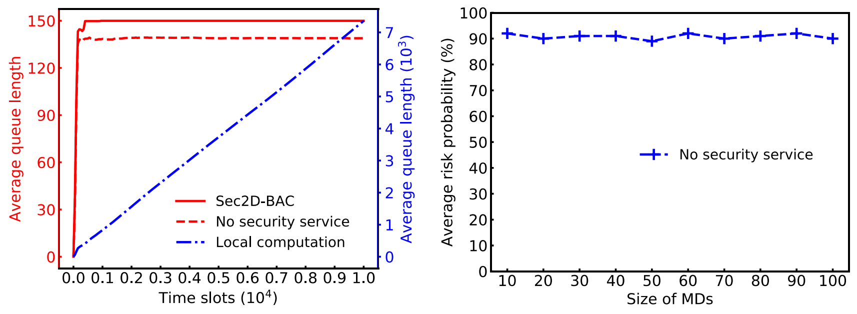
(a) Convergence of energy consumption (b) Convergence of queue length (c) ART

Fig. 6. Comparison results among Sec2D-BB, Sec2D-BAC and Sec2D-Greedy.

Fig. 7 gives the comparison results among Sec2D-BAC, No security service, and Local computation under different n. From Figs. 7(a) and 7(b), we can see that with the increase of n, the AECs and AQLs of all algorithms increase. This is because the AEC and AQL are the time average of the sum of all MDs' energy consumption and queue length on each time slot, respectively. So, the more MDs in the D2D group, the larger AEC and the longer AQL. Further, No security service performs best on AEC and AQL under all the size of MDs, as it does not have any energy consumption resulting from security service, and also it has no security workload in the back-end queue. However, lacking security protection, the data transmission in collaborative task offloading will experience high risk, shown in Fig. 7(d). Additionally, Sec2D-BAC has almost the same AEC as Local computation, but extremely less AQL. For example, when $n = 10$, the AQL of Sec2D-BAC is less than 200, while the AQL of Local computation is about 6500. The corresponding convergence behavior of AQL under $n = 10$ is given in Fig. 7(c). We find that the queues of Sec2D-BAC and No security service are quickly stable. In contrast, the AQL of Local computation always increases with the number of time slots, indicating that Local computation cannot maintain system stability. The reason is that if an MD with low computation capacity but has a high task arrival rate without collaborative task offloading, it is not capable of processing all the workload even under the maximum $F_{i,L}^{max}$ and $F_{i,H}^{max}$. In this case, the AQL will become longer as the time slot evolving. Hence, under Local computation, a huge amount of workload is congested, and hence many tasks cannot be processed, resulting in less energy consumption.



(a) AEC (b) AQL

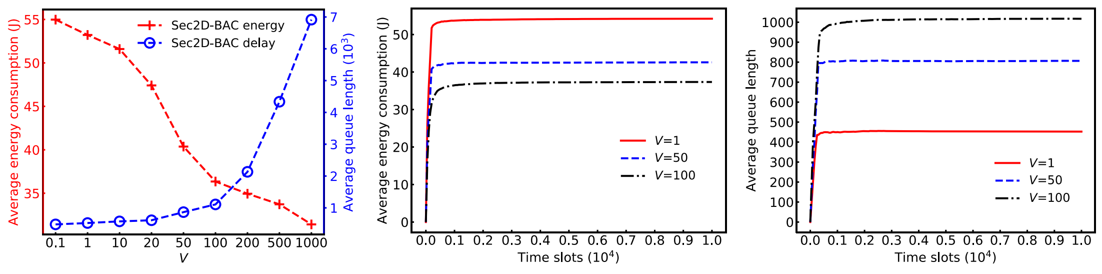


(c) Convergence of queue length (d) ARP

Fig. 7. Comparison results among Sec2D-BAC, No security service, and Local computation under different n.

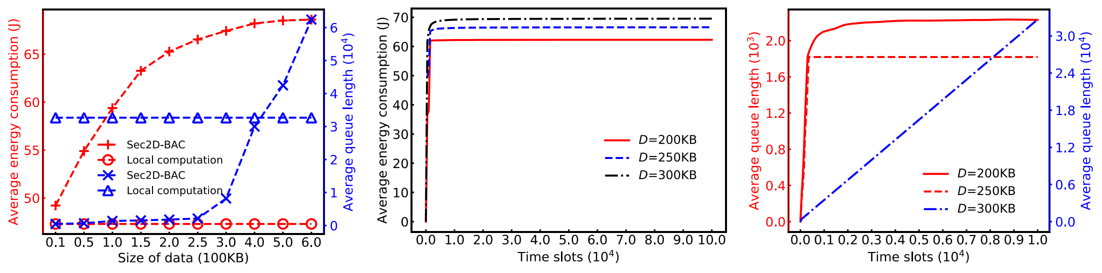
6.3 The impact of systemparameters

Fig. 8 explores the energy-delay tradeoff of Sec2D-BAC under different V, where $n = 10$, $\beta = 0.5$, and $\eta_v = 0.5$. From Fig. 8(a), we can see that as V goes from 0.1 to 1000, the AEC of Sec2D-BAC decreases, but the AQL increases. According to drift-plus-penalty, the small V means the system emphasizes queue backlog, resulting in low AQL but high AEC. On the contrary, when V is very large, this indicates the system focuses on energy consumption instead of queue delay. As a result, Sec2D-BAC gains high AQL but low AEC. This phenomenon is consistent with our theoretical analysis (see Theorem 1). Also, we plot the convergence behaviors of AEC and AQL under three situations ($V = 1, 50, 100$) in Figs. 8(b) and 8(c). We observe that all the AECs and AQLs can be stable in all cases, and different V will incur different AECs and AQLs. This suggests that the control parameter V indeed implements the energy-delay tradeoff. Thus, a proper V can be chosen in a practical scenario to satisfy self-defined requirements on energy consumption and delay.



(a) Energy-delay tradeoff (b) Convergence of energy consumption (c) Convergence of queue length

Fig. 8. Impact of different V.



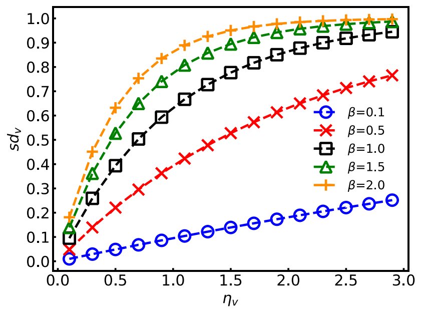
(a) AEC and AQL (b) Convergence of energy consumption (c) Convergence of queue length

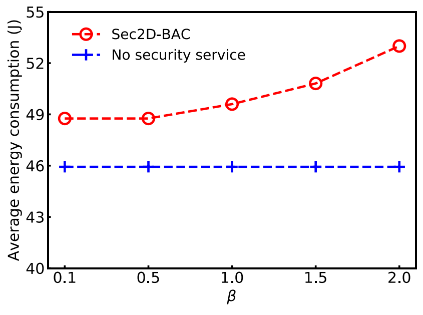
Fig. 9. Impact of different D.

To investigate the impact of D, we conduct the experiments on Sec2D-BAC and Local computation with D varying from 10 to 600 KB. The related results are plotted in Fig. 9, where $n = 10$, $a_i = 10$, $\beta = 0.5$, $\eta_v = 0.5$ and $V = 1$. We make the following observations from Fig. 9(a). For Local computation, the curves of AEC and AQL are flat, indicating it is independent of D. This is because the data size only impacts directly on security service and data transmission, i.e., the larger D, the larger $T_i^{sc}(\tau)$ and $T_i^{tx}(\tau)$ (see Eqs. (17) and (18)). So, for Sec2D-BAC, with the increase of D, its AEC and AQL increase. Moreover, the AEC of Sec2D-BAC grows rapidly with D, and then the curve becomes nearly flat when $D = 300$ KB. In contrast, the AQL of Sec2D-BAC increases slowly and then goes up fast when $D = 300$ KB. The main reason behind this phenomenon is that, when $D = 300$ KB, the queue of Sec2D-BAC is unstable. This can be explained that: 1) the larger D will produce less offloading tasks due to wireless bandwidth limitation; 2) the larger D results in more security workload, making the back-end queue longer. The corresponding convergences of AEC and AQL on $D = 200, 250, 300$ KB are shown in Figs. 9(b) and 9(c). From Fig. 9(b), it seems that the traces of AEC in all cases are stable, even when $D = 300$ KB. This is because, when $D = 300$ KB, the energy consumption of the whole system almost reaches the maximum value. Particularly, from Fig. 9(c), the AQLs of $D = 200$ KB and 250KB are stable, but the AQL of $D = 300$ KB appears to grow linearly with time slots. This indicates that the Sec2D-BAC cannot keep the queue stable when $D = 300$ KB. Recall that from Fig. 9(a), when $D = 500$ KB, the AQL of Sec2D-BAC is even longer than that of Local computation. In a word, the above results in Fig. 9 suggest that the data size has a great impact on Sec2D-BAC.

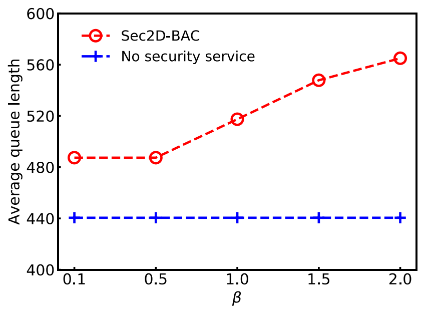
6.4 The impact of security parameters

To reveal the performance impact of β, we compare the various results of Sec2D-BAC and No security service by varying β from 0.1 to 2.0. Fig. 10 gives the corresponding results, where we fix $n = 10$, $a_i = 10$, $V = 1$, and $\eta_v = 0.5$. Fig. 10(a) shows the relationship among β, η_v and sd_v. For a specified η_v, a larger β leads to a larger sd_v. So, MD should apply a higher level of security service to protect data security, resulting in more security workload. Besides, the more security workload, the more energy consumption, and a longer queue length. Therefore, the AEC and AQL of Sec2D-BAC increase with β, which are shown in Figs. 10(b) and 10(c), respectively. Also, we notice that when $\beta = 0.1$ and 0.5, Sec2D-BAC has the same AEC and AQL. The rationale is that, when β is a small value, the lowest security level will meet the security demand for the above two cases. Lacking security protection, the AEC and AQL of No security service are independent of β. However, we can see from Fig. 10(d) that the larger β incurs, the higher the security risk. We know that data security is very critical. As a consequence, our proposed Sec2D-BAC can ensure data security under various β by costing more energy and tolerating longer delays.

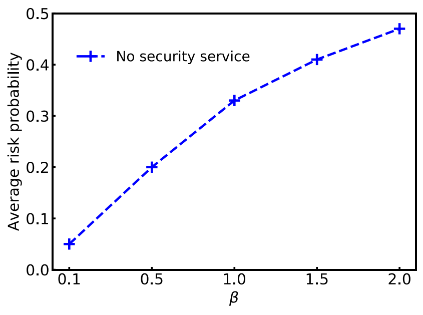




(a) Relation among β, sd_v and η_v (b) AEC



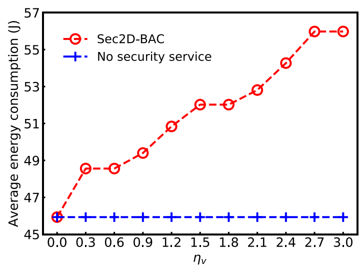
(c) AQL



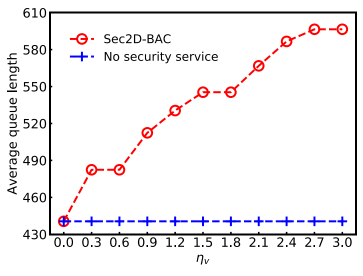
(d) ARP

Fig. 10. Impact of different β.

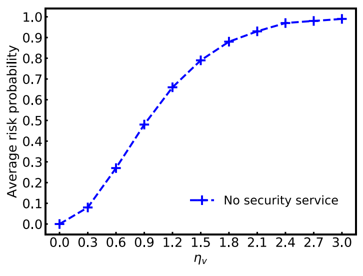
Fig. 11 shows the results of the impact of $\eta_v (v \in \{cd, ig\})$ varying from 0 to 3 with the increment of 0.3, where $n = 10$, $a_i = 10$, $V = 1$, and $\beta = 0.5$. From Figs. 11(a) and 11(b), we find that AEC and AQL of Sec2D-BAC increase with η_v, because the larger η_v will incur higher sd_v and more security workload. In particular, when $\eta_v = 0$, it means that there is no security risk in D2D communication. In this case, Sec2D-BAC and No security service have the same AEC and AQL. In addition, sometimes, for example, when $\eta_v = 0.3$ and 0.6, Sec2D-BAC has the same AEC and AQL. This is primarily due to the reason that the security demands under different η_v can be satisfied by the same security level, resulting in no variation on AEC and AQL. In similar, due to the lack of security services, the AEC and AQL of No security service are independent of η_v, and the corresponding curves are flat. According to Eq. (9), the larger η_v will generate higher ARP. The related result is plotted in Fig. 11(c). When $\eta_v = 3$, the system will experience almost 100% risk. Thus, we need to deploy security services to guarantee data security.



(a) AEC



(b) AQL



(c) ARP

Fig. 11. Impact of different η_v.

6.5 Results summary

Sec2D-BAC outperforms Sec2D-BB on running time and performs better than Sec2D-Greedy on AEC and AQL but with more running time. Although No security service has less AEC and AQL than Sec2D-BAC, it will experience high ARP. For some MDs with low computing capacity but high task arrival rate, Local computation cannot maintain the queue stability. Our proposed Sec2D-BAC can stabilize the workload queue and ensure the data security by the collaborative task offloading mechanism, even varying various parameters, i.e., n, V, β and η_v. Nevertheless, when the data size of a task is too large, Sec2D-BAC will consume more energy to implement security service, even making the queue unstable. Overall, Sec2D-BAC can achieve security and collaborative task offloading when D is not very large, under the existing parameters set. The rationale is that data security is achieved at the expense of other performance.

7. Conclusions and future work

This paper investigates the security and energy-aware collaborative task offloading in D2D communication. We first build a novel security model for mobile devices by considering the number of CPU cores, CPU frequency, and data size. Thus, this security model can be used in the heterogeneous D2D scenario. Then, we formulate the collaborative task offloading problem where the objective is to optimize the time-average delay and energy consumption of MDs with the security guarantees. Moreover, the Lyapunov optimization framework is applied to implement online decisions making, and a low-complexity WO-based greedy approach and a BAC-based optimal approach are proposed to solve the generated MILP problem. Theoretical proof shows that the Sec2D algorithm meets a $[O(1/V), O(V)]$ energy-delay tradeoff. Moreover, the simulation results validate the theoretical analysis and indicate that Sec2D can stabilize the system, as validated by our experiments. In a word, our proposed algorithm can ensure data security and keep the system stable for collaborative D2D communication.

For our future work, we will apply security services for task offloading in the mobile edge computing environment, where the edge servers and MDs should build their own security models. Another direction is to use the model-free method, deep reinforcement learning, to implement the computation or task offloading for D2D communication, which can meet different performance requirements compared to the Lyapunov technique.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61802095, 61572162), the Zhejiang Provincial Key Science and Technology Project Foundation (No. 2018C01012), the National Key R&D Program of China (2016YFC0800803), the VC Research (No. VCR 0000126) for Prof Chang. Hua Hu is the co-corresponding author.

Appendix A. Proof of Lemma 1

According to Eq. (26), we have

$$L(\tau+1) - L(\tau) = \frac{1}{2}\sum_{i\in\mathcal{N}}\{[Q_i^F(\tau+1)]^2 - [Q_i^F(\tau)]^2\}$$
$$+\frac{1}{2}\sum_{j\in\mathcal{N}}\{[Q_j^B(\tau+1)]^2 - [Q_j^B(\tau)]^2\} \tag{37}$$

Taking $[Q_i^F(\tau+1)]^2 - [Q_i^F(\tau)]^2$ as an example,

$$[Q_i^F(\tau+1)]^2 - [Q_i^F(\tau)]^2 \le \left[\sum_{j\in\mathcal{N}} u_{ij}(\tau)\, W^{ex}\right]^2$$
$$+[a_i(\tau)\, W^{ex}]^2 + 2Q_i^F(\tau)[a_i(\tau)\, W^{ex} - \sum_{j\in\mathcal{N}} u_{ij}(\tau)\, W^{ex}] \tag{38}$$

which can be derived in terms of the fact that for any $Q \ge 0$, $a \ge 0$ and $b \ge 0$, we have:

$$(\max[Q-b, 0] + a)^2 \le Q^2 + a^2 + b^2 + 2Q(a-b)$$

Repeating above steps for the back-end queue, we get

$$[Q_j^B(\tau+1)]^2 - [Q_j^B(\tau)]^2 \le \left[\sum_{i\in\mathcal{N}} u_{ij}(\tau)\, (W^{ex} + W_{ij}^{sc})\right]^2 + \left[b_j(\tau)\right]^2$$
$$+2Q_j^B(\tau)[\sum_{i\in\mathcal{N}} u_{ij}(\tau)\, (W^{ex} + W_{ij}^{sc}) - b_j(\tau)] \tag{39}$$

Moreover, as $a_i(\tau) \le A_{max}$, $b_i(\tau) \le B_{max}$ and $u_{ij}(\tau) \le U_{max}$, we have

$$\left[\sum_{j\in\mathcal{N}} u_{ij}(\tau)\, W^{ex}\right]^2 + [a_i(\tau)W^{ex}]^2 + \left[\sum_{i\in\mathcal{N}} u_{ij}(\tau)\, (W^{ex} + W_{ij}^{sc})\right]^2 + \left[b_j(\tau)\right]^2$$
$$\le [nU_{max}W^{ex}]^2 + [A_{max}W^{ex}]^2 + [nU_{max}(W^{ex} + W_{max}^{sc})]^2 + B_{max}^2 \tag{40}$$

Then,

$$L(\tau+1) - L(\tau) \le \Omega + \sum_{i\in\mathcal{N}}\{Q_i^F(\tau)[a_i(\tau)W^{ex} - \sum_{j\in\mathcal{N}} u_{ij}(\tau)\, W^{ex}]\}$$
$$+ \sum_{j\in\mathcal{N}}\{Q_j^B(\tau)[\sum_{i\in\mathcal{N}} u_{ij}(\tau)\, (W^{ex} + W_{ij}^{sc}) - b_j(\tau)]\} \tag{41}$$

Now adding $V\mathbb{E}\ \{E(\tau)|\boldsymbol{Q}(\tau)\}$ to the both sides of Eq. (41) that proves the lemma 1.

Appendix B. Proof of Lemma 2

In terms of Eq. (22), at time slot τ, the energy consumption of all the MDs can be represented by,

$$E(\tau) = \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} P_i^{cx}(\tau)\,[u_{ij}(\tau)W_{ij}^{sc}/C_i(\tau)]$$
$$+ \sum_{i\in\mathcal{N}} P_i^{cx}(\tau)[b_i(\tau)/C_i(\tau)]$$
$$+ \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} P_i^{tx}[u_{ij}(\tau)D/R_{ij}(\tau)] \tag{42}$$

Based on this, we extract the items related to $u_{ij}(\tau)$ from Eq. (29a), that is

$$\sum_{j\in\mathcal{N}} Q_j^B(\tau)\sum_{i\in\mathcal{N}} u_{ij}(\tau)\ [W^{ex} + W_{ij}^{sc}] - \sum_{i\in\mathcal{N}} Q_i^F(\tau)\sum_{j\in\mathcal{N}} u_{ij}(\tau)\, W^{ex}$$
$$+ \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} V[P_i^{cx}(\tau)u_{ij}(\tau)W_{ij}^{sc}/C_i(\tau)] + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} V[P_i^{tx}u_{ij}(\tau)\,D/R_{ij}(\tau)] \tag{43}$$

Moreover, we have the following two equalities,

$$\sum_{j\in\mathcal{N}} Q_j^B(\tau)\sum_{i\in\mathcal{N}} u_{ij}(\tau)\ [W^{ex} + W_{ij}^{sc}] = \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} u_{ij}(\tau)Q_j^B(\tau)[W^{ex} + W_{ij}^{sc}] \tag{44}$$

$$\sum_{i\in\mathcal{N}} Q_i^F(\tau)\sum_{j\in\mathcal{N}} u_{ij}(\tau)\, W^{ex} = \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} u_{ij}(\tau)Q_i^F(\tau)W^{ex} \tag{45}$$

Then, Eq. (43) can be rewritten by,

$$\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} u_{ij}(\tau)[Q_j^B(\tau)(W^{ex} + W_{ij}^{sc}) - Q_i^F(\tau)W^{ex}]$$
$$+ \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} V[P_i^{cx}(\tau)u_{ij}(\tau)\,W_{ij}^{sc}/C_i(\tau)]$$
$$+ \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} V[P_i^{tx}u_{ij}(\tau)\,D/R_{ij}(\tau)] \tag{46}$$

Similarly, we also extract the items related to $b_j(\tau)$ from Eq. (29a), i.e.,

$$\sum_{i\in\mathcal{N}} V[P_i^{cx}(\tau)\, b_i(\tau)/C_i(\tau)] - \sum_{j\in\mathcal{N}} Q_j^B(\tau)b_j(\tau) \tag{47}$$

We know that $\sum_{j\in\mathcal{N}} Q_j^B(\tau)b_j(\tau) = \sum_{i\in\mathcal{N}} Q_i^B(\tau)b_i(\tau)$, and let

$$\rho_{ij}(\tau) = Q_j^B(\tau)[W^{ex} + W_{ij}^{sc}] - Q_i^F(\tau)W^{ex} + V[P_i^{cx}(\tau)\,W_{ij}^{sc}/C_i(\tau) + P_i^{tx}D/R_{ij}(\tau)]$$

and

$$\varphi_i(\tau) = VP_i^{cx}(\tau)/C_i(\tau) - Q_i^B(\tau)$$

Then, Eq. (29a) can be rewritten as

$$\min\ \sum_{i\in\mathcal{N}}[\sum_{j\in\mathcal{N}} \rho_{ij}(\tau)u_{ij}(\tau) + \varphi_i(\tau)b_i(\tau)] \tag{48}$$

We can see from Eq. (48) that, for each MD_i, as both the objective function and constraints can be decomposed for individual (i.e., $u_{ij}(\tau)$, $b_i(\tau)$, $F_{i,l}(\tau)$, and $F_{i,h}(\tau)$), the online optimal decisions can be done separately at each MD_i. Then, we reformulate Eq. (48) as follows:

$$\min\ \sum_{j\in\mathcal{N}} \rho_{ij}(\tau)u_{ij}(\tau) + \varphi_i(\tau)b_i(\tau) \tag{49}$$

This completes the proof of Lemma 2.

Appendix C. Proof of Theorem 1

For current time slot τ, because of Eq. (33) and Lemma 3 hold, we can take expectations of both sides and use the law of iterated expectations to yield

$$\Delta(\tau) + V\mathbb{E}\{E(\tau)\} \le \Omega + VE^* - \epsilon\sum_{i\in\mathcal{N}}\mathbb{E}\ [Q_i^F(\tau) + Q_i^B(\tau)] \tag{50}$$

Summing over $\tau \in \{0,1,\ldots,t-1\}$ and using the law of telescoping sums yields

$$\mathbb{E}\{L(\boldsymbol{Q}(t))\} - \mathbb{E}\{L(\boldsymbol{Q}(0))\} + V\sum_{\tau=0}^{t-1}\mathbb{E}\{E(\tau)\} \leq (\Omega + VE^*)t - \epsilon\sum_{\tau=0}^{t-1}\sum_{i\in\mathcal{N}}\mathbb{E}\ [Q_i^F(\tau) + Q_i^B(\tau)] \quad (51)$$

The above equation uses the fact that $\sum_{\tau=0}^{t-1}\Delta(\tau) = \mathbb{E}\{L(\boldsymbol{Q}(t))\} - \mathbb{E}\{L(\boldsymbol{Q}(0))$. Rearranging terms and neglecting non-negative terms, we can get

$$\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{E(\tau)\} \leq (\Omega + VE^*)/V + \mathbb{E}\{L(\boldsymbol{Q}(0))\}/Vt \quad (52)$$

and

$$\frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{i\in\mathcal{N}}\mathbb{E}[Q_i^F(\tau) + Q_i^B(\tau)] \leq (\Omega + VE^*)/\epsilon + \mathbb{E}\{L(\boldsymbol{Q}(0))\}/\epsilon t \quad (53)$$

Taking limits of the above as $t \to \infty$ and suppose $L(\boldsymbol{Q}(0)) = 0$, that proves the Eqs. (35) and (36).

REFERENCES

[1] L. Pu, X. Chen, J. Xu, X. Fu, D2D fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted D2D collaboration, IEEE Journal on Selected Areas in Communications 34 (12) (2016) 3887-3901.

[2] Z. Hong, Z. Wang, W. Cai, V.C.M. Leung, Connectivity-aware task outsourcing and scheduling in D2D networks, In: 26th International Conference on Computer Communication and Networks (ICCCN), 2017, pp. 1-9.

[3] W. Jiang, G. Feng, S. Qin, T.P. Yum, Efficient D2D content caching using multi-agent reinforcement learning, In: IEEE Conference on Computer Communications Workshops (INFOCOM Workshops), 2018, pp. 511-516.

[4] J. Liu, K. Luo, Z. Zhou, X. Chen, A D2D offloading approach to efficient mobile edge resource pooling, In: 16th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), 2018, pp. 1-6.

[5] J.M. Liang, P.Y. Chang, J.J. Chen, C.F. Huang, Y.C. Tseng, Energy-efficient DRX scheduling for D2D communication in 5G networks, Journal of Network and Computer Applications 116 (2018) 53-64.

[6] I.A. Elgendy, W. Zhang, Y.C. Tian, K. Li, Resource allocation and computation offloading with data security for mobile edge computing, Future Generation Computer Systems 100 (2019) 531-541.

[7] Y. Mao, J. Zhang, S.H. Song, K.B. Letaief, Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems, IEEE Transactions on Wireless Communications 16 (9) (2017) 5994-6009.

[8] TQ. Dinh, J. Tang, Q.D. La, TQS. Quek, Offloading in mobile edge computing: Task allocation and computational frequency scaling, IEEE Transactions on Communications 65 (8) (2017) 3571-3584.

[9] F. Wang, J. Xu, X. Wang, S. Cui, Joint offloading and computing optimization in wireless powered mobile-edge computing systems, IEEE Transactions on Wireless Communications 17 (3) (2018) 1784-1797.

[10] D.T. Huynh, M. Chen, T.T. Huynh, C.H. Hai, Energy consumption optimization for green Device-to-Device multimedia communications, Future Generation Computer Systems, 92 (2019) 1131-1141.

[11] Cisco visual networking index: Global mobile data traffic forecast update, Technique Report, 2019.

[12] M.K. Pedhadiya, R.K. Jha, H.G. Bhatt, Device to device communication: A survey, Journal of Network and Computer Applications 129 (2019) 71-89.

[14] J. Feng, L. Zhao, J. Du, X. Chu, F.R. Yu, Computation offloading and resource allocation in D2D-enabled mobile edge computing, In: 2018 IEEE International Conference on Communications (ICC), 2018, pp. 1-6.

[15] S. Debroy, P. Samanta, A. Bashir, M. Chatterjee, SpEED-IoT: Spectrum aware energy efficient routing for device-to-device IoT communication, Future Generation Computer Systems 93 (2019) 833-848.

[16] P. Gandotra, R.K. Jha, S. Jain, A survey on device-to-device (D2D) communication: Architecture and security issues, Journal of Network and Computer Applications 78 (2017) 9-29.

[17] F. Jameel, Z. Hamid, F. Jabeen, S. Zeadally, M.A. Javed, A survey of device-to-device communications: Research issues and challenges, IEEE Communications Surveys and Tutorials 20 (3) (2018) 2133-2168.

[18] M. Haus, M. Waqas, A.Y. Ding, Y. Li, S. Tarkoma, J. Ott, Security and privacy in device-to-device (D2D) communication: A review, IEEE Communications Surveys and Tutorials 19 (2) (2017) 1054-1079.

[23] Y. Jung, E. Festijo, M. Peradilla, Joint operation of routing control and group key management for 5G ad hoc D2D networks, In: 2014 International Conference on Privacy and Security in Mobile Systems (PRISMS), 2014, pp. 1-8.

[24] W. Shen, W. Hong, X. Cao, B. Yin, DM Shila, Y. Cheng, Secure key establishment for device-to-device communications, In: IEEE Global Communications Conference (GLOBECOM), 2014, pp. 336-340.

[25] R. Hsu. J. Lee, Group anonymous D2D communication with end-to-end security in LTE-A, In: 2015 IEEE Conference on Communications and Network Security (CNS), 2015, pp. 451-459.

[26] A. Zhang, J. Chen, R.Q. Hu, Y. Qian, SeDS: Secure data sharing strategy for D2D communication in LTE-advanced networks, IEEE Transactions on Vehicular Technology 65 (4) (2016) 2659-2672.

[27] T. Xie, X. Qin, Scheduling security-critical real-time applications on clusters, IEEE Transactions on Computers 55 (7) (2006) 864-879.

[28] S. Song, K. Hwang, Y. Kwok, Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling, IEEE Transactions on Computers 55 (6) (2006) 703-719.

[29] L. Zeng, B. Veeravalli, X. Li, SABA: A security-aware and budget-aware workflow scheduling strategy in clouds, Journal of Parallel and Distributed Computing 75 (2015) 141-151.

[30] Z. Li, J. Ge, H. Yang, L. Huang, H.Y. Hu, H. Hu, B. Luo, A security and cost aware scheduling algorithm for heterogeneous tasks of scientific workflow in clouds, Future Generation Computer Systems 65 (2016) 140-152.

[31] H. Chen, X. Zhu, D. Qiu, L. Liu, Z. Du, Scheduling for workflows with security-sensitive intermediate data by selective tasks duplication in clouds, IEEE Transactions on Parallel and Distributed Systems 28 (9) (2017) 2674-2688.

[32] W. Jiang, K. Jiang, X. Zhang, Y. Ma, Energy optimization of security-critical real-time applications with guaranteed security protection, Journal of Systems Architecture-Embedded Systems Design 61 (7) (2015) 282-292.

[34] I.A. Elgendy, W.Z Zhang, Y. Zeng, H. He, YC Tian, Y. Yang, Efficient and Secure Multi-User Multi-Task Computation Offloading for Mobile-Edge Computing in Mobile IoT Networks, IEEE Transactions on Network and Service Management 17 (4) 2020 2410-2422.

[33] B. Huang, Z. Li, P. Tang, S. Wang, J. Zhao, H. Hu, W. Li, V.I. Chang, Security modeling and efficient computation offloading for service workflow in mobile edge computing, Future Generation Computer Systems 97 (2019) 755-774.

[38] X. Lin, Y. Wang, Q. Xie, M. Pedram, Task scheduling with dynamic voltage and frequency scaling for energy minimization in the mobile cloud computing environment, IEEE Transactions on Services Computing 8 (2) (2015) 175-186.

[39] T. Xie, X. Qin, Security-aware resource allocation for real-time parallel jobs on homogeneous and heterogeneous clusters, IEEE Transactions on Parallel and Distributed Systems 19 (5) (2008) 682-697.

[40] X. Lyu, W. Ni, H. Tian, R.P. Liu, X. Wang, G.B. Giannakis, A. Paulraj, Optimal schedule of mobile edge computing for internet of things using partial information, IEEE Journal on Selected Areas in Communications 35 (11) (2017) 2606-2615.

[41] Y. Sun, S. Zhou, J. Xu, EMM: Energy-aware mobility management for mobile edge computing in ultra dense networks, IEEE Journal on Selected Areas in Communications 35 (11) (2017) 2637-2646.

[42] Y. Yao, L. Huang, A.B. Sharma, L. Golubchik, M.J. Neely, Power cost reduction in distributed data centers: A two-time-scale approach for delay tolerant workloads, IEEE Transactions on Parallel and Distributed Systems 25 (1) (2014) 200-211.

[43] Ross, S.M., 2014. Introduction to Probability Models. Cambridge, MA, USA: Academic.

[44] L. Georgiadis, M.J. Neely, L. Tassiulas, Resource allocation and cross-layer control in wireless networks, Foundations and Trends in Networking 1 (1) (2006) 1-149.

[45] M.J. Neely, Stochastic Network Optimization with Application to Communication and Queueing Systems, Synthesis Lectures on Communication Networks, Morgan & Claypool Publishers, 2010.

[46] W. Zhao, S. Wang, Resource sharing scheme for device-to-device communication underlaying cellular networks, IEEE

Transactions on Communications 63 (12) (2015) 4838-4848.

[47] T. Guerout, Y. Gaoua, C. Artigues, G.D. Costa, P. Lopez, T. Monteil, Mixed integer linear programming for quality of service optimization in clouds, Future Generation Computer Systems 71 (2017) 1-17.

[48] I. Chakroun, Parallel heterogeneous branch and bound algorithms for multi-core and multi-GPU environments, Ph.D. Dissertation, Lille University of Science and Technology, 2013.

[49] P.G. Saghand, H. Charkhgard, C. Kwon, A branch-and-bound algorithm for a class of mixed integer linear maximum multiplicative programs: A bi-objective optimization approach, Computers & Operations Research 101 (2019) 263-274.

[50] S.J. Tyber, Cutting planes in mixed integer programming: Theory and algorithms, Ph.D. Dissertation, Georgia Institute of Technology, 2013.

[51] A. Testa, A. Rucco, G. Notarstefano, A finite-time cutting plane algorithm for distributed mixed integer linear programming, In: 56th IEEE Annual Conference on Decision and Control (CDC), 2017, pp. 3847-3852.

[52] Y. Song, J.R. Luedtke, Branch-and-cut approaches for chance-constrained formulations of reliable network design problems, Mathematical Programming Computation 5 (4) (2013) 397-432.

[53] M. Miltenberger, T.K. Ralphs, D.E. Steffy, Exploring the numeric of branch-and-cut for mixed integer linear optimization, In: Operations Research Proceedings, 2017, pp. 151-157.

[54] J.E. Mitchell, Branch-and-cut algorithms for combinatorial optimization problems, Handbook of Applied Optimization, Oxford University Press, 2000.

[55] C. Wang, C. Liang, F.R. Yu, Q. Chen, L. Tang, Computation offloading and resource allocation in wireless cellular networks with mobile edge computing, IEEE Transactions on Wireless Communications 16 (8) (2017) 4924-4938.

[13] M. Waqas, Y. Niu, Y. Li, M. Ahmed, D. Jin, S. Chen, Z. Han, A comprehensive survey on mobility-aware D2D communications: principles, practice and challenges, IEEE Communications Surveys & Tutorials 22 (3) 2020 1863-1886.

[19] W.K. Lai, Y.C. Wang, H.C. Lin, J.W. Li, Efficient resource allocation and power control for LTE-A D2D communication with pure D2D model, IEEE Transactions on Vehicular Technology 69 (3) 2020 3202-3216.

[20] N. Fan, X. Wang, D. Wang, Y. Lan, J. Hou, A collaborative task offloading scheme in D2D-assisted fog computing networks, IEEE Wireless Communications and Networking Conference (WCNC), pp. 1-6, 2020.

[21] U. Saleem, Y. Liu, S. Jangsher, X. Tao, Y. Li, Latency minimization for D2D-enabled partial computation offloading in mobile edge computing, IEEE Transactions on Vehicular Technology 69 (4) (2020) 4472-4486.

[22] G. Li, M. Chen, X. Wei, T. Qi, W. Zhuang, Computation offloading with reinforcement learning in D2D-MEC Network, International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 69-74, 2020.

[35] A. Mansour, A. Davis, M. Wagner, R. Bassous, H. Fu, Y. Zhu, Multi-asymmetric cryptographic RSA scheme, The 12th Annual Conference on Cyber and Information Security Research (CISRC), pp. 9:1-9:8, 2017.

[36] A. Mansour, K.M. Malik, N. Kaso, AMOUN: Lightweight scalable multi-recipient asymmetric cryptographic scheme, IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), pp. 838-846, 2019.

[37] A.F.X Ametepe, SARM Ahouandjinou, E.C. Ezin, Secure encryption by combining asymmetric and symmetric cryptographic method for data collection WSN in smart agriculture, IEEE International Smart Cities Conference (ISC2), pp. 93-99, 2019.