Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2021) 29: 2374 – 2384

doi:10.3906/elk-2011-151

An ADMM-based incentive approach for cooperative data analysis in edge computing

Weiwei FANG[1,*], **Xue WANG**[1], **Qingli WANG**[1], **Yi DING**[2]
[1]School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China
[2]School of Information, Beijing Wuzi University, Beijing, China

Received: 29.11.2020 • **Accepted/Published Online:** 19.04.2021 • **Final Version:** 23.09.2021

Abstract: Edge computing is a new paradigm that provides data processing capabilities at the network edge. In view of the uneven data distribution and the constrained onboard resource, an edge device often needs to call for a number of neighboring devices as followers to cooperate on data analysis tasks. However, these followers may be rational and selfish, having their private optimization objectives such as energy efficiency. Therefore, the leader device needs to incentivize the followers to achieve a certain global objective, e.g., maximizing task accomplishment, rather than their own objectives. In this paper, we model the aforementioned challenges in edge computing as a Stackelberg game, and integrate this Stackelberg game with alternating direction method of multipliers (ADMM) to solve the conflict between the global and individual objectives. Through rigorous analysis and extensive experiments, we verify that the proposed approach can quickly converge to the optimum regardless of the number of followers and is very robust to parameter variations.

Key words: Edge computing, ADMM, game theory, computation offloading

1. Introduction

Cloud computing has dramatically changed how people process data and obtain information. However, this computing paradigm usually suffers unpredictably long latency and incurs high bandwidth occupation [1, 2]. Meanwhile, it also raises privacy and security issues. As data is increasingly generated at the network edge, it would be more efficient to move computation and analysis to a close proximity of data sources, using edge devices such as smart phones, laptops, and edge servers [1]. It has been envisioned that such an edge computing paradigm will become a promising supplement to cloud computing and have as significant impacts on our society as the latter [3].

Unlike cloud servers, edge devices are usually constrained by computation and storage resources due to deployment limitation or power provision. Thus, a single edge device may have only a small fraction of the data and cannot perform edge analytics all on its own [4]. To overcome these problems, the cooperative edge computing was recently introduced as a promising solution, in which a leader device can call for a number of qualified neighboring devices to work together on computation and analysis tasks [5]. For example, Gong et al. [4] proposed a privacy-preserving solution for the logistic regression model training based on distributed sensing data from edge devices. Xiao et al. [6] designed an alternating direction method of multipliers (ADMM)-based cooperation strategy that enables an edge device to forward its workload to neighboring devices to maximize

*Correspondence: wwfang@bjtu.edu.cn

users' quality of experience. Sahni et al. [7] developed the Edge Mesh computing paradigm to distribute computation tasks among multiple devices within the edge system. Some studies [4–7] assume that all involved edge devices are willing to cooperate with each other to achieve a common goal. However, in reality, different devices usually have their own individual optimization objectives, while these objectives may conflict with each other. Thus, the follower devices have to be incentivized by the leader device for compensating their loss in cooperation. Liu et al. [8] proposed a Stackelberg game-based strategy for the cloud to stimulate edge servers to participate in computation offloading. However, it costs too much time for payment negotiations between edge devices, which is unacceptable in large-scale systems [9]. Chen et al. [11] designed a distributed algorithm based on merge-and-split rules to incentivize edge devices to form coalitions and cooperate with others to maximize system utility. It was revealed in [9, 10] that scalability is hard to guarantee in this game theoretic approach.

In this paper, we study the issue of designing a fast convergent and scalable incentive approach for cooperative data analysis in edge computing. Specifically, an edge computing system with one leader device and multiple follower devices are considered. It is very idealistic to assume that the followers are always willing to cooperate with the leader on computation freely. Actually, the followers often have their own optimization objectives, which may contradict with that of the leader. Thus, the follower has to incentivize these followers for compensating their loss in cooperation on data analysis tasks. Naturally, we model such a problem as a Stackelberg game [12], where the leader device determines the payments to compensate the followers for their energy consumption on data processing. Then, this problem is solved in a distributed manner by exploiting the advantages of both game theory and ADMM [9]. Note that ADMM is a simple yet very effective algorithm that can decompose a large-scale optimization problem into a series of small-scale subproblems which can be solved quickly and efficiently [13]. Through rigorous analysis and extensive simulations, we verify the performance of the proposed work in terms of convergence, scalability, and stability.

The rest of this paper is organized as follows. We introduce the system model and formulate the problem in Section 2. Then, the Stackelberg-based ADMM algorithm is discussed in detail in Section 3. Numerical results are presented in Section 4. Finally, we conclude this paper in Section 5.

2. Problem formulation

In this paper, we consider a system consisting of one leader device (LD) and N follower devices (FDs). The system operates in slotted time with fixed slot length T (in seconds). The LD can request for cooperation from these FDs on data analysis, such as model training [4] or model inference [1]. When an FD $i \in \{1, ..., N\}$ receives such a request, it will execute the analytical task based on its own local data at an average processing rate of x_i (in bits/s), and then feed back the results to the LD. The LD's objective is to maximize its revenue on cooperative data analysis, so the utility function of LD is defined as

$$U_{LD}(\mathbf{x}) = \sum_{i=1}^{N} \log(1 + x_i T), \tag{1}$$

where the log function is used to model the diminishing returns on FD i's data [14]. To obtain the resulting data with enough diversity [4], the following constraint should be satisfied

$$\sum_{i=1}^{N} \mu_i x_i T \geq C, \tag{2}$$

where $\mu_i \in (0,1]$ is the empirical parameter that statistically measures the data quality and the contribution of FD i [9, 14], and C is the threshold given by the LD for measuring the participant's contribution.

The FDs are assumed to be rational and selfish, and each of them tries to maximize its own utility. To define the utility function for an FD i, we first characterize the FD's energy consumption on computation. Following [15], we use processing density γ_p (in cycles/bit) to model the amount of CPU processing resources required per bit for computation workload. Let f_i (in cycles/s) denote the CPU-cycle frequency of FD i, and the CPU could adaptively adjust f_i in the range of f_i^{min} to f_i^{max} by using dynamic voltage scaling (DVS) technology. Then, we know that

$$x_i = \frac{f_i}{\gamma_p} \in [\frac{f_i^{min}}{\gamma_p}, \frac{f_i^{max}}{\gamma_p}]. \tag{3}$$

The CPU power consumption of FD i is modeled as $P_i = qf_i^3$, where q is a parameter determined by chip architecture [15]. Finally, we define the utility function of FD i as

$$u_i(x_i) = \alpha_i x_i T - \beta_i E_i, \tag{4}$$

where $E_i = P_i T$ denotes the energy consumption of i in the slot, and $\alpha_i, \beta_i \in [0,1]$ denote a collection of normalizing weights. This objective is a weighted linear combination of the LD's contribution and expenditure in the cooperation. By choosing appropriate values for α_i and β_i, we can assign different priorities between throughput and energy.

It is obvious that U_{LD} and all u_i have different optimal results since the LD and FDs have their respective optimization objectives. The LD has to offer incentives to the FDs to induce them to change their actions to help maximize the global objective instead of their own objectives. Our work is to propose such an incentive mechanism to achieve the optimal result for the LD's objective in (1) under the constrains in (2) and (3). This is done by designing a set of incentive functions as optimization objectives for LDs. Specifically, FD i optimizes a new incentive function $\Psi_i(u_i(x_i), \theta_i)$ rather than $u_i(x_i)$, where θ_i is the incentive parameter controlled by the LD to influence $u_i(x_i)$. Based on what is discussed above, we can formulate the problem as a Stackelberg game as

$$\begin{aligned}
\text{Leader's game:} \quad & \{\theta_i^*\} = \min_{\theta_i} \sum_{i=1}^{N} -\log(1 + x_i T), \\
\text{Followers' game:} \quad & x_i^* = \min_{x_i} \Psi_i((-\alpha_i x_i T + \beta_i E_i), \theta_i^*), \quad \forall i \in \{1, \ldots, N\}, \\
\text{Constraints:} \quad & \sum_{i=1}^{N} \mu_i x_i T \geq C, \\
& x_i \in [\frac{f_i^{min}}{\gamma_p}, \frac{f_i^{max}}{\gamma_p}], \quad \forall i \in \{1, \ldots, N\},
\end{aligned} \tag{5}$$

where the LD and FDs are taken as the leader and the followers in the game, respectively, and Z^* denotes the optimal value of Z when the Stackelberg equilibrium is achieved. Our work is to design the incentive function $\Psi_i(u_i(x_i), \theta_i)$ and the parameter θ_i for each $i \in \{1, ..., N\}$ so that the leader and the followers finally have the same optimal point in the game. However, it is often very difficult to achieve a global optimum because the

update of variables can only be accomplished in a distributed manner. That is, the LD can only make decisions on the incentive parameter $\{\theta_i\}$, as the leader's strategy to incentivize each FD i to make decisions on its own computation resource f_i and x_i, as the follower's strategy. To achieve a consensus between the LD and the FDs, it is natural to solve the problem in (5) by an iterative algorithm with reasonable complexity.

3. Algorithm design

In this section, we first introduce the design of incentive function and describe the iteration process of our algorithm based on the basic framework proposed in [9]. Then, we prove that the basic conditions in our design are able to guarantee the convergence of the proposed algorithm.

3.1. Stackelberg game-based ADMM

At each iteration k, the LD offers its incentive θ_i to FD i, and the incentive function for FD i is constructed as

$$\Psi_i((-\alpha_i x_i T + \beta_i E_i), \theta_i^k) = L_i(x_i, \lambda) + \Psi_i^k(x_i), \tag{6}$$

where $L_i(x_i, \lambda)$ is a Lagrangian function of x_i, and $\Psi_i^t(x_i)$ is the independent part of the FD i after being stimulated. Specifically, these two parameters are respectively given as

$$L_i\left(x_i, \lambda\right) = -\log\left(1 + x_i T\right) - \lambda \mu_i x_i T, \tag{7}$$

$$\Psi_i^k\left(x_i\right) = -\alpha_i x_i T + \beta_i E_i - \theta_i^k x_i, \tag{8}$$

where λ is the Lagrange multiplier. We can interpret the incentive function in (6) as follows. The LD uses $\theta_i^k x_i$ as a payment to FD i for optimizing $L_i(x_i, \lambda)$, where θ_i^k is the price for the contribution of FD i. Under this incentive, FD i not only considers its independent part $\Psi_i^k(x_i)$ but also helps the LD to achieve the global objective by minimizing $L_i(x_i, \lambda)$.

The operation of our proposed Stackelberg-based ADMM is an iterative process as depicted in Figure 1. At each step k, given incentive factors $\{\theta_i^k\}$ from the LD, each FD i updates its own processing rate x_i^k to minimize the incentive function in (6). Based on the results from all FDs, the LD adjusts the incentive factors by $\{\theta_i^{k+1}\}$ so that the FDs are willing to cooperate with the LD on data analysis. This iterative process continues until the convergence is achieved. However, it is difficult for the FDs to achieve the current optimal processing state $\mathbf{x}^k$ in one step due to the constraints (2) from the LD and (3) from the FDs. Thus, the FDs have to interact with the LD iteratively to ensure that these two constraints can be satisfied. As a result, we design a two-tier iteration process to solve the problem.

In the inner loop, the LD and FDs attempt to reach an optimal point (x_i^k, λ^k) for solving the followers' game in (5) by using the ADMM algorithm. The ADMM algorithm takes advantage of the decomposability of dual ascent and the fast convergence properties of the method of multipliers to solve the large-scale problems

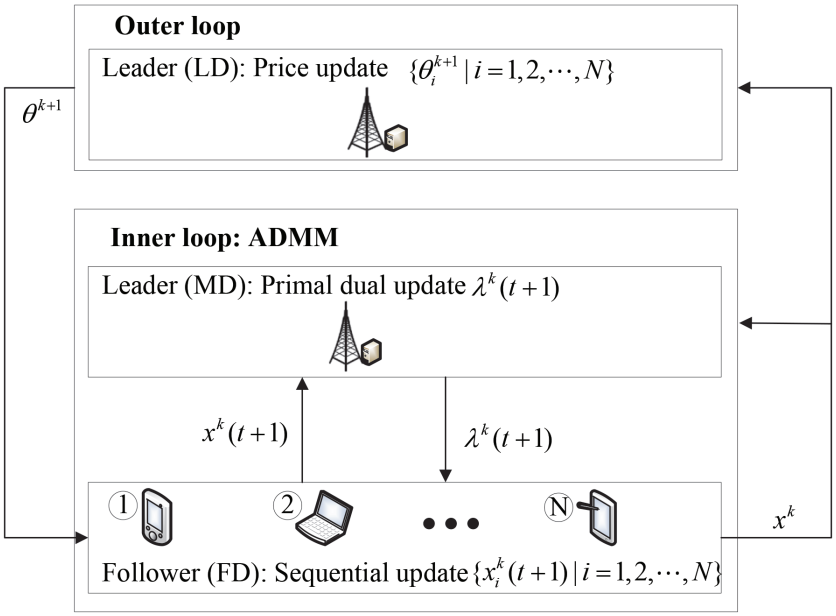

Figure 1. An overview of the Stackelberg-based ADMM scheme.

efficiently. Specifically, we first update $x_i^k(t+1)$ sequentially by solving the following problems:

$$\begin{aligned} \min_{x_i} \quad & L_i(x_i, \lambda^k(t)) + \Psi_i^k(x_i) + \frac{\rho}{2} \left\| \sum_{j=1}^{i-1} \mu_j x_j^k(t+1)T + \mu_i x_i T + \sum_{j=i+1}^{N} \mu_j x_j^k(t)T - C \right\|_2^2, \\ \text{s.t.} \quad & x_i \in [\frac{f_i^{min}}{\gamma_p}, \frac{f_i^{max}}{\gamma_p}], \end{aligned} \tag{9}$$

where t is the iteration step index in the inner loop, $\rho > 0$ is a damping factor in ADMM.

Then, the LD updates its dual variable as

$$\lambda^k(t+1) = \lambda^k(t) - \rho \left(\sum_{i=1}^{N} \mu_i x_i^k(t+1)T - C \right). \tag{10}$$

The above steps for the inner loop are repeated until each x_i^k and λ^k do not change significantly, i.e. converge to the current optimal point at iteration k. After that, in the outer loop, each FD feeds back its own marginal cost to the LD for price update as

$$\theta_i^{k+1} = \nabla_{x_i}^k \left(-\alpha_i x_i T + \beta_i E_i \right). \tag{11}$$

It is rational for the LD to set the price equal to or lager than θ_i^{k+1}, when the LD wants FDs to cooperate on data analysis. The outer loop terminates when the stopping criterion is satisfied as

$$\left\| L\left(\{x_i^{k+1}\}, \lambda^{k+1}\right) - L\left(\{x_i^k\}, \lambda^k\right) \right\| \leq \varepsilon_0, \tag{12}$$

Algorithm 1 Stackelberg game-based ADMM for the LD and FDs

Input:

$k=0,\ t=0,\ \{\theta_i=0|i=1,\ldots,N\},\ \{x_i=0|i=1,\ldots,N\},\ \lambda=0$

Output:

$\theta_i^*,\ x_i^*,\ i=1,\ldots,N$

Procedure

while $\left\|L\left(\left\{x_i^{k+1}\right\},\lambda^{k+1}\right)-L\left(\left\{x_i^k\right\},\lambda^k\right)\right\|>\varepsilon_0$ **do**

(a) **ADMM-based Optimization in the Inner Loop:**

Update $x_i^k(t)$ and $\lambda^k(t)$ according to (9) and (10), respectively, until convergence is achieved

(b) **Leader's Price Design based on the Followers' Feedback:**

The LD adjust the price for FD i according to (11)

k = k + 1

end while

Result:

Optimal Variable: $x_i^*=x_i^k,\ 1\le i\le N$;

Optimal Price: $\theta_i^*=\theta_i^k,\ 1\le i\le N$.

where ε_0 is a predefined threshold. The augmented Lagrangian function $L\left(\{x_i\},\lambda\right)$ can be given as

$$L\left(\{x_i\},\lambda\right)=\sum_{i=1}^{N}L_i\left(x_i,\lambda\right)+\lambda C+\frac{\rho}{2}\left\|\sum_{i=1}^{N}\mu_i x_i T-C\right\|_2^2. \tag{13}$$

The algorithm for the above two-layer iteration process is summarized in Algorithm 1.

3.2. Convergence guarantee

According to [9, 10], the following proposition should be proved to guarantee the convergence of Algorithm 1.

Proposition 1 *To guarantee the convergence of our Stackelberg game-based ADMM algorithm, the following conditions should be satisfied:*

(1) Each part of the LD's objective function, i.e. $-\log(1+x_iT)$*, is a strongly convex function;*

(2) The objective function of each FD, i.e. $-\alpha_i x_i T+\beta_i E_i$*, is a strongly convex function.*

(3) The objective function of each FD, i.e. $-\alpha_i x_i T+\beta_i E_i$*, satisfies a uniform Lipschitz gradient condition [16].*

Proof (1) The first-order and second-order derivative for $-\log(1+x_iT)$ can be calculated as

$$\nabla_{x_i}\left(-\log\left(1+x_iT\right)\right)=-\frac{1}{\ln 10}\cdot\frac{T}{1+x_iT}.$$

$$\nabla_{x_i}^2\left(-\log\left(1+x_iT\right)\right)=\frac{1}{\ln 10}\cdot\frac{T^2}{\left(1+x_iT\right)^2}\ge\frac{1}{\ln 10}\cdot\frac{T^2\gamma_p^2}{\left(\gamma_p+f_i^{mx}T\right)^2}>0.$$

According to convex optimization theory [17], $-\log(1+x_iT)$ is a strongly convex function.

(2) The first-order and second-order derivative for $-\alpha_i x_i T+\beta_i E_i$ can be calculated as

$$\nabla_{x_i}\left(-\alpha_i x_i T+\beta_i E_i\right)=-\alpha_i T+3\beta_i T q\left(x_i\gamma_p\right)^2.$$

$$\nabla_{x_i}^2 \left(-\alpha_i x_i T + \beta_i E_i\right) = 6\beta_i T k \gamma_p^2 x_i \geq 6\beta_i T q \gamma_p f_i^{min} > 0.$$

According to convex optimization theory [17], $-\alpha_i x_i T + \beta_i E_i$ is a strongly convex function.

(3) The Lipschitz gradient condition generally requires the first-order gradient of the target function changes with a speed below a positive real value [9, 17], which can be proved as follows:

$$\frac{\left|\nabla_{x_i}\left(-\alpha_i x_i' T + \beta_i E_i\right) - \nabla_{x_i}\left(-\alpha_i x_i T + \beta_i E_i\right)\right|}{|x_i' - x_i|} = 3\beta_i T q \gamma_p^2 (x_i' + x_i) \leq 6\beta_i T q \gamma_p f_i^{max}.$$

□

Based on the above conditions, our algorithm can converge linearly with an average number of iterations bounded by $O(1/\varepsilon_0)$ [9]. The proof is similar to that of [10] and omitted here for brevity.

4. Numerical results

4.1. Experimental settings

In this section, we evaluate the performance of our Stackelberg game-based ADMM through simulation experiments, with parameter values suggested by [10, 15]. Table lists the default values for simulation parameters in our experiments, if not otherwise specified. All the experiments are conducted on the Matlab platform [17].

Table . Simulation parameters.

Notation	Definition	Default value
N	Total number of FDs	10
α_i	Weight for throughput	0.5
β_i	Weight for energy	0.016
ε_0	Threshold for ADMM convergence	$1e-4$
T	Time slot in seconds	1
μ_i	Data quality indicator	$\mu \sim N(0,1]$
C	Threshold for measuring FD's contribution	0.5
q	Parameter for CPU power consumption	$1e-26$
f_i^{max}	Maximal CPU frequency	2 GHz
γ_p	CPU processing density	1000 cycles/bit
ρ	Damping factor in ADMM	1.0
λ	Lagrange multiplier	1.0

4.2. Comparisons on incentive approaches

Figure 2 compares the LD's objective function of the proposed algorithm with that of the optimal objective value and that without any incentive mechanism. It can be found that the proposed algorithm can quickly approach to the optimal value within 40 iterations and finally converges to the minimal point of the LD's objective after 80 iterations. Notice that the initial value of the LD's objective is equal to the value with no incentive mechanism. The results clearly verify the fast convergence of our incentive approach.

Figure 3 compares the convergence speed of the proposed algorithm with the naive approach in game theory [12]. Similar to that in [10], the naive approach here adopts a simple incentive function $\theta_i^{k+1} = \theta_i^k + \Delta$, where $\Delta = 0.01$ when $\nabla_{x_i}^k \left(-\alpha_i x_i T + \beta_i E_i\right) > 0$, and $\Delta = -0.01$ otherwise. As shown in the figure, this naive approach has a relatively slower speed to approach the optimal value and at least cannot achieve convergence within 100 iterations. That is because the constraints in (5) are not always satisfied in the naive approach, while our algorithm can always guarantee the satisfaction of constraints in (5) by carefully selecting the incentive factors in (11).

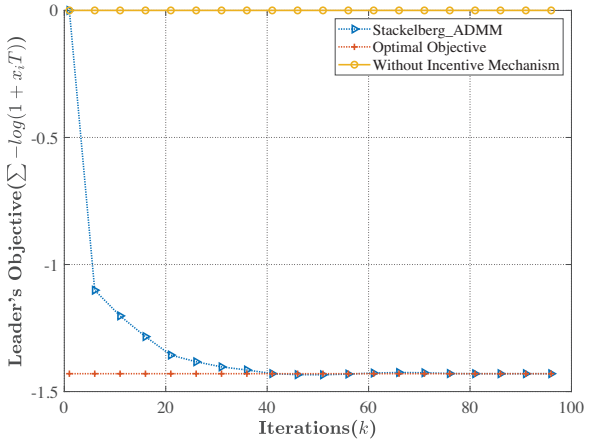

Figure 2. The convergence of LD's objective function.

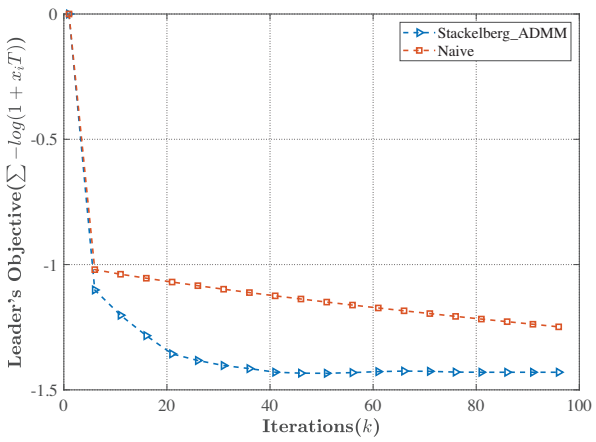

Figure 3. Impact of different incentive approaches.

4.3. Impacts of system parameters

In Figures 4 and 5, we show the impacts of varying the stopping threshold ε_0 on system performance. From Figure 4, the proposed algorithm can always converge toward the optimal value of the LD's objective. However, the performance gap would be more noticeable when we choose a relatively larger ε_0, e.g., $\varepsilon_0 = 10^{-1}$. Besides, the proposed algorithm can converge to the actual optimal value of LD's objective only when ε_0 is small enough, i.e. $\varepsilon_0 \leq 10^{-3}$. Correspondingly, it will inevitably take more number of iterations for our algorithm to achieve the final convergence. From Figure 5, we can observe that the relation between the number of iterations and $\log_{10}(1/\varepsilon_0)$ is linear, which confirms our analysis on linear convergence in Section 3.2. Thus, we should carefully choose a proper value for this stopping threshold to balance the trade-off between performance and cost in algorithm execution.

In Figures 6 and 7, we show the impacts of varying the number of FDs, N, on system performance. From Figure 6, as N increases from 10 to 50, the proposed algorithm can always converge to a corresponding optimal objective value in dozens of iterations. The results are consistent with the definition of LD's objective in (5). Meanwhile, we can see from Figure 7 that the required number of iterations has not increased significantly or prohibitively but is well controlled within the order of tens to hundreds of iterations (about 60 to 110 iterations). All in all, the optimum can be guaranteed, and the convergence rate is linear with or independent of the system size. These results clearly indicate the capability of our algorithm for large-scale systems, which is superior to the traditional game theoretic approaches [11].

In Figures 8 and 9, we show the impacts of varying the key parameters, i.e. f_i^{max} or γ_p, on system

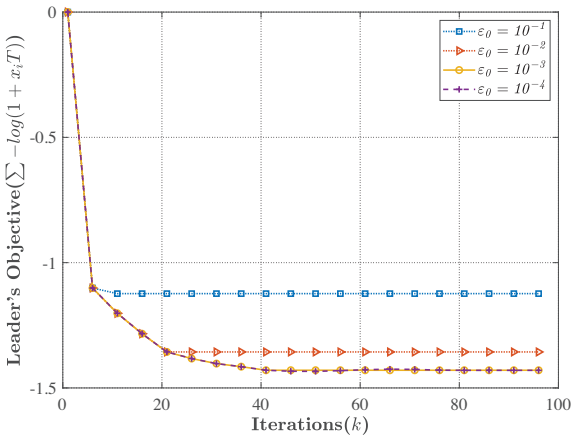

Figure 4. LD's objective under different ε_0.

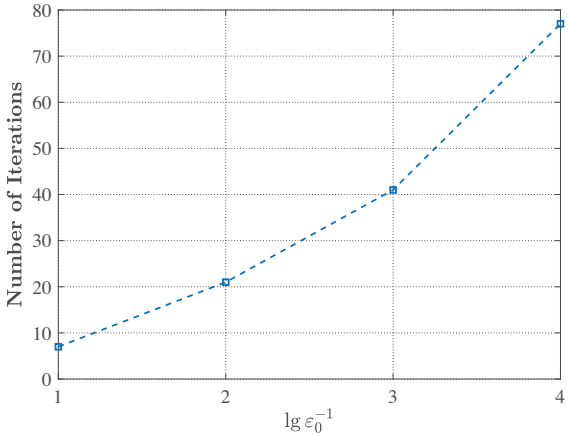

Figure 5. Total number of iterations under different ε_0.

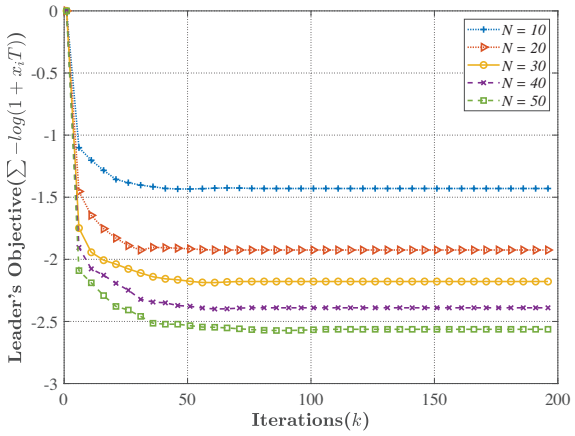

Figure 6. LD's objective under different N.

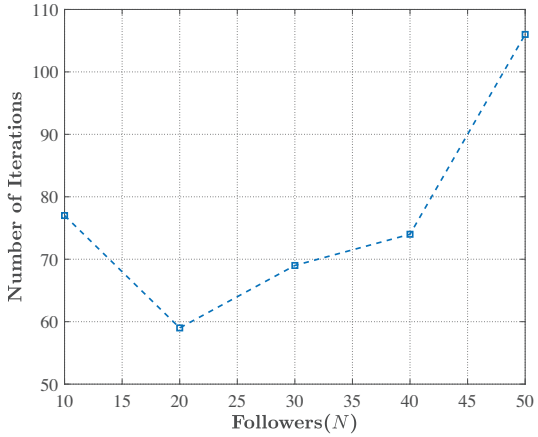

Figure 7. Total number of iterations under different N.

performance. As shown in Figure 8, as f_i^{max} increases from 1 to 3 GHz, the FDs will share more computing resources to cooperate with the LD, resulting in lower objective values. However, the result gap becomes much smaller when f_i^{max} is relatively higher, i.e. $f_i^{max} = 3$. According to the CPU power consumption model, the energy consumption on computation has currently become the dominant factor in the FD's objective function, which prevents the task cooperation between LD and FDs to some extent. Interestingly, this is consistent with the real fact that people are not willing to cooperate with others when the task would most probably influence the normal usage of their devices.

As shown in Figure 9, when the target task becomes more complex and requires more computation, the FDs are constrained by their processing capability (i.e. f_i^{max}) and may not be able to provide enough support to help the LD. As a result, we can notice that the objective value tends to converge to be closer to 0.

5. Conclusion

In this paper, we study the problem of designing an effective and efficient incentive approach for an edge computing system with one leader and multiple followers. The leader device provides incentives to the rational

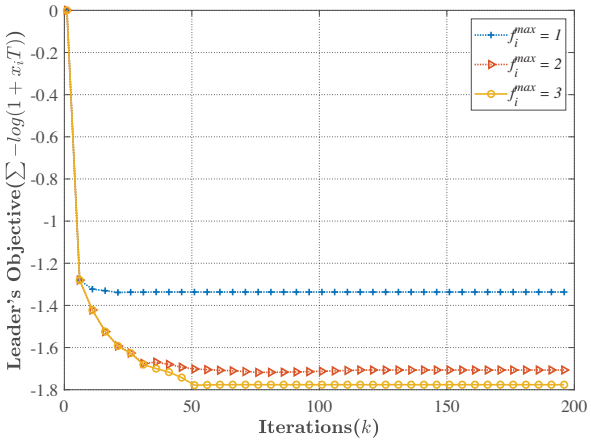

Figure 8. LD's objective under different f_i^{max}.

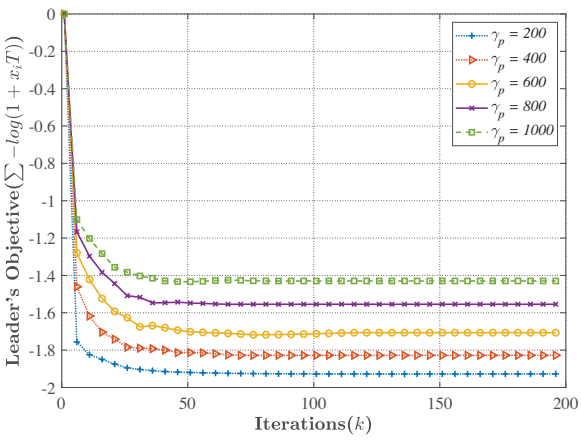

Figure 9. LD's objective under different γ_p.

and selfish following devices so that they are willing to provide cooperation on data analysis tasks. The problem is formulated as a Stackelberg game, which is decomposed into two subproblems and solved in a distributed manner by ADMM. Thorough analysis and extensive experiments demonstrate its rapid convergence, good scalability, and high robustness to parameter variations. In the future, we plan to extend the proposed approach for more general objective functions as well as other game theoretic models [12].

Acknowledgments

This work is supported by the Beijing Municipal Natural Science Foundation under Grant L191019, the Open Project of Key Laboratory of Industrial Internet of Things & Networked Control, Ministry of Education under Grant 2019FF03, the Open Project of Beijing Intelligent Logistics System Collaborative Innovation Center under Grant BILSCIC-2019KF-10, the CERNET Innovation Project under Grant NGII20190308, the Research Base Project of Beijing Municipal Social Science Foundation under Grant 18JDGLB026 and the Science and Technique General Program of Beijing Municipal Commission of Education under Grant KM201910037003.

References

[1] Shi W, Cao J, Zhang Q, Li Y, Xu L. Edge computing: vision and challenges. IEEE Internet of Things Journal 2016; 5: 637-646.

[2] Fang W, Zhou W, Li Y, Yao X, Xiong N. A distributed ADMM approach for energy-efficient resource allocation in mobile edge computing. Turkish Journal of Electrical Engineering & Computer Sciences 2018; 6: 3335-3344.

[3] Wang S, Zhang X, Zhang Y, Wang L, Yang J, Wang W. A survey on mobile edge networks: Convergence of computing, caching and communications. IEEE Access 2017; 5: 6757-6779.

[4] Gong Y, Fang Y, Guo Y. Private data analytics on biomedical sensing data via distributed computation. IEEE Transactions on Computational Biology and Bioinformatics 2016; 3: 431-444.

[5] Tran T, Hajisami A, Pandey P, Pompili D. Collaborative mobile edge computing in 5G networks: new paradigms, scenarios, and challenges. IEEE Communications Magazine 2017; 4: 54-61.

[6] Xiao Y, Krunz M. QoE and power efficiency tradeoff for fog computing networks with fog node cooperation. In: IEEE 2017 Conference on Computer Communications; 1-4 May, 2017; Atlanta, GA, USA. New York, NY, USA: IEEE. pp. 1-9.

[7] Sahni Y, Cao J, Zhang S, Yang L. Edge mesh: a new paradigm to enable distributed intelligence in Internet of Things. IEEE Access 2017; 5: 16441-16458.

[8] Liu Y, Xu C, Zhan Y, Liu Z, Guan J, Zhang H. Incentive mechanism for computation offloading using edge computing: a Stackelberg game approach. Computer Networks 2017; 2: 399-409.

[9] Zheng Z, Song L, Han Z, Li G, Poor HV. Game theoretic approaches to massive data processing in wireless networks. IEEE Wireless Communications 2018; 1: 98-104.

[10] Zheng Z, Song L, Han Z, Li G, Poor HV. Game theory for big data processing: Multileader multifollower game-based ADMM. IEEE Transactions on Signal Processing 2018; 15: 3933-3945.

[11] Chen L, Xu J. Socially trusted collaborative edge computing in ultra dense networks. In: ACM/IEEE 2017 Symposium on Edge Computing; 12-14 October, 2017; San Jose, CA, USA. New York, NY, USA: IEEE. pp. 1-11.

[12] Fudenberg D, Tirole J. Game Theory. Cambridge, MA, USA: MIT Press, 1993.

[13] Boyd S, Parikh N, Chu E, Peleato B, Eckstein J. Distributed optimization and statistical learning via the alternating direction method of multipliers. Found Trends Mach Learn 2011; 3: 1-122.

[14] Koh JY, Peters GW, Leong D, Nevat I, Wong WC. Privacy-aware incentive mechanism for mobile crowd sensing. In: IEEE 2017 International Conference on Communications; 21-25 May, 2017; Paris, France. New York, NY, USA: IEEE. pp. 1-6.

[15] Kwak J, Kim Y, Lee J, Chong S. DREAM: dynamic resource and task allocation for energy minimization in mobile cloud systems. IEEE Journal on Selected Areas in Communications 2015; 12: 2510-2523.

[16] Hong MY, Luo ZQ. On the linear convergence of the alternating direction method of multipliers. Mathematical Programming 2017; 162: 165-199.

[17] Boyd S, Vandenberghe L. Convex Optimization. New York, NY, USA: Cambridge University Press, 2004.