FedUC: A Unified Clustering Approach for Hierarchical Federated Learning

Qianpiao Ma, Yang Xu, *Member, IEEE*, Hongli Xu, *Member, IEEE*, Jianchun Liu
Liusheng Huang, *Member, IEEE*

Abstract—Federated learning (FL) is an effective approach to train models collaboratively among distributed edge nodes (*i.e.*, workers) while facing three crucial challenges, edge heterogeneity, resource constraint, and Non-IID data. Under the parameter server (PS) architecture, a single parameter server may become the system bottleneck and cannot well deal with the edge heterogeneity, while the peer-to-peer (P2P) architecture causes significant communication consumption to achieve satisfactory training performance. To this end, hierarchical aggregation (HA) architecture is proposed to cluster workers to tackle the edge heterogeneity and reduce communication consumption for FL. However, the existing researches on HA architecture cannot provide a unified clustering approach for various inter-cluster aggregation patterns (*e.g.*, centralized or decentralized structure, synchronous or asynchronous mode). In this paper, we explore the quantitative relationship between the convergence bounds of different inter-cluster patterns and several factors, *e.g.*, data distribution, frequency of clusters participating in inter-cluster aggregation (for asynchronous modes), and inter-cluster topology (for decentralized structures). Based on the convergence bounds, we design a unified clustering algorithm FedUC to organize workers for different patterns. Experimental results on classical models and datasets show that FedUC can greatly accelerate the model training of different patterns by 1.79-7.39$\times$ compared with the state-of-the-art clustering methods.

Index Terms—Federated learning, Edge computing, Clustering optimization, Heterogeneity, Resource constraint, Non-IID.

1 Introduction

With the increasing popularity of Internet of Things (IoT), a massive amount of data are generated from physical worlds each day [1][2][3]. Traditionally, these data are forwarded to the remote cloud for training or processing, which will lead to potential privacy leakage and massive bandwidth consumption due to long-distance transmission. To this end, *edge computing* is proposed to push more computation capacity to the network edge, enabling efficient data processing locally. Besides, it motivates the application of federated learning (FL), which implements distributed machine learning over edge nodes (also called workers) [4][5][6].

To implement highly efficient FL in edge computing, we should take into account the following factors and challenges.

- **Edge Heterogeneity:** Various edge nodes with diverse geographic locations, data volume, CPU capacities, and network connections, will act as workers in practical applications [7]. As a result, the time for performing local updating and delivering models may vary greatly.
- **Resource Constraint:** On the one hand, training models are usually computation-intensive, while the computation resource on each worker is usually limited [8]. On the other hand, the frequent model transmission between workers consumes enormous communication bandwidth. As a result, the edge network may be easily congested due to limited communication resources [9][10].
- **Non-IID Data:** Since a worker collects data from its physical location directly, its local data often cannot be regarded as the samples drawn uniformly from the overall distribution. In other words, the data among workers are usually non-independent-and-identically-distributed (Non-IID) [11]. The previous works [12][13] have pointed out that the performance of FL will be significantly degraded over Non-IID data.

There are three types of architectures, parameter server (PS), peer-to-peer (P2P), and hierarchical aggregation (HA) as dominant solutions for FL. Under the PS architecture [11][14], a centralized parameter server aggregates the local models from workers and then distributes the aggregated global model back. On suffering from the enormous amount of traffic workload, the parameter server will become the system bottleneck, leading to the risk of single point failure and poor scalability [15]. Under the P2P architecture [16–19], workers transmit their models to each other through peer-to-peer communication. Although the P2P architecture can achieve better scalability compared with the PS architecture, each worker needs to share its model with all the others to achieve the same training performance as the PS architecture, which causes significant communication consumption [16]. To reduce communication consumption, some previous P2P solutions transmit models through sparse communication

- *This work was supported in part by the National Science Foundation of China (NSFC) under Grant 92367104, Grant 92267301 and Grant 62132019. (Corresponding author: Hongli Xu.)*

- *Q. Ma is with the Purple Mountain Laboratories, Nanjing, Jiangsu, China, 211111. E-mail: maqianpiao@pmlabs.com.cn*
- *Y. Xu, H. Xu, J. Liu and L. Huang are with the School of Computer Science and Technology, University of Science and Technology of China, Hefei, Anhui, China, 230027, and also with Suzhou Institute for Advanced Study, University of Science and Technology of China, Suzhou, Jiangsu, China, 215123. E-mail: xuyangcs@ustc.edu.cn, xuhongli@ustc.edu.cn, jcliu17@ustc.edu.cn, lshuang@ustc.edu.cn*

links, *e.g.*, by constructing subgraphs for the original topology [18][19]. However, sparse topologies may generate the high-variance problem among workers and the model may be hard to converge, especially under Non-IID data among workers [20].

To mitigate the disadvantages of both PS and P2P architectures, the HA architecture [21–28] has been proposed, where the workers are organized into multiple clusters. Each cluster is deployed with an aggregator, which may be a base station or a selected leader worker within the cluster. The HA architecture contains two tiers of model aggregation. One is intra-cluster aggregation, where each aggregator aggregates the local models of the workers within its cluster into a cluster model. The other is inter-cluster aggregation, where the aggregators collaborate to derive new cluster models from their current cluster models. After that, aggregators distribute their new cluster models to the workers within the corresponding clusters. Compared with the PS architecture, HA disperses the communication consumption on the centralized server over multiple aggregators, achieving better scalability. Meanwhile, since the local models of workers within the same cluster are consistent with the received cluster model, the high-variance problem among workers will be relieved. Thus, HA can effectively accelerate convergence by contrast with the P2P architecture.

In this paper, we explore the quantitative relationship between the convergence bounds of different inter-cluster aggregation patterns (*i.e.*, synchronous or asynchronous, centralized or decentralized) and several factors, *e.g.*, data distribution among clusters, frequency of clusters participating in asynchronous inter-cluster aggregation, and decentralized inter-cluster topology. Moreover, for intra-cluster aggregation, we propose an optimal time-sharing scheduling strategy to address edge heterogeneity and communication resource constraint. Based on both the convergence analysis for inter-cluster patterns and the optimal intra-cluster strategy, we design a unified clustering algorithm FedUC to construct clusters that can be adapted to different inter-cluster patterns of HA. Our FedUC is orthogonal to HA architectures with different inter-cluster patterns, so their training performance can be greatly improved by deploying our clustering algorithm.

The main contributions of this paper are as follows:

- We explore the quantitative relationship between the convergence bounds of four different inter-cluster patterns (*i.e.*, CenSyn, CenAsy, DecSyn and DecAsy) and several factors, *e.g.*, data distribution among clusters, frequency of clusters participating in inter-cluster aggregation (for CenAsy and DecAsy), and inter-cluster topology (for DecSyn and DecAsy).
- For intra-cluster aggregation, we propose an optimal time-sharing scheduling algorithm as the aggregation strategy, which can minimize the completion time of intra-cluster aggregation.
- Based on both the convergence analysis for inter-cluster patterns and the optimal intra-cluster strategy, we design a unified clustering algorithm FedUC to solve the cluster construction problem given the time constraints for intra-cluster aggregation in HA.
- Experimental results on the classical models and datasets show that, by deploying our clustering algorithm, the model training of CenSyn (1.79×), CenAsy (5.88×), DecSyn (7.39×) and DecAsy (5.32×) can be greatly accelerated.

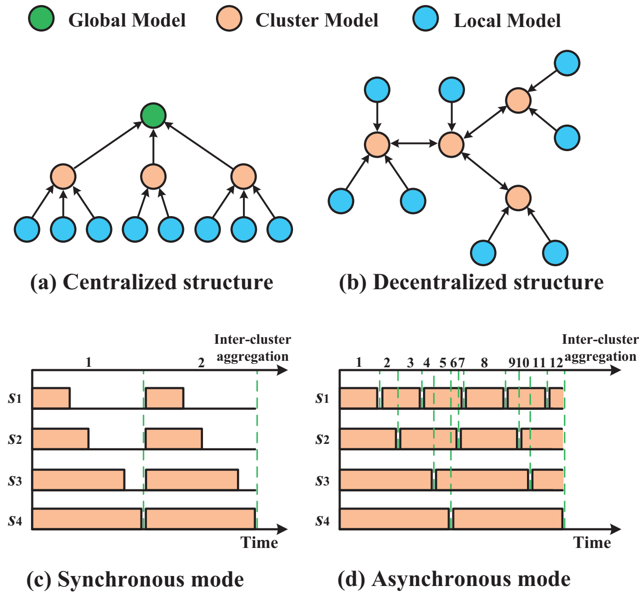


Fig. 1: Inter-cluster structures and communication modes.

TABLE 1: Inter-cluster structures and communication modes.

Mode / Structure	Synchronous	Asynchronous
Centralized	[21][22]	[23][24][25]
Decentralized	[26][27]	[28]

2 Related Works

The inter-cluster aggregation of the existing HA researches is generally considered in terms of two dimensions: structure and communication mode. The inter-cluster structure can be either centralized or decentralized. In a centralized structure, a central parameter server aggregates the cluster models from aggregators and distributes the global model back to them, while in a decentralized structure, each aggregator exchanges its cluster model with its neighboring aggregators bidirectionally. The inter-cluster communication mode can be either synchronous or asynchronous. In synchronous mode, all aggregators perform one inter-cluster aggregation in each round, while in asynchronous mode, each aggregator performs inter-cluster aggregation immediately after completing its local updating. Fig. 1 illustrates the inter-cluster structures (centralized and decentralized) and communication modes (synchronous and asynchronous). These two dimensions result in four inter-cluster patterns: centralized-synchronous pattern (CenSyn) [21][22], centralized-asynchronous pattern (CenAsy) [23][24][25], decentralized-synchronous pattern (DecSyn) [26][27] and decentralized-asynchronous pattern (DecAsy) [28], as summarized in Table 1.

In fact, even with the same clustering method, different inter-cluster patterns will lead to diverse training performance, whereas none of the existing clustering methods pay attention to this importance. For example, the authors in [22–28] cluster each worker to the aggregator with the shortest

TABLE 2: Performance comparison for inter-cluster patterns.

Clustering \ Pattern	CenSyn	CenAsy	DecSyn	DecAsy
Communication -aware [22–28]	Medium	Poor	Poor	Poor
Data-aware [21]	Good	Medium	Medium	Medium
FedUC	Good	Good	Good	Good

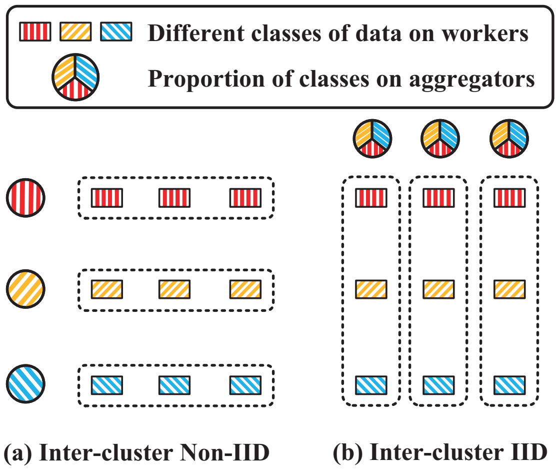

Different classes of data on workers
Proportion of classes on aggregators
(a) Inter-cluster Non-IID
(b) Inter-cluster IID

Fig. 2: Non-IID and IID data among clusters.

communication time (or distance). However, due to Non-IID data among workers, a communication-aware clustering method likely causes a high degree of data Non-IID among clusters, as shown in Fig. 2(a). In this case, each cluster model is trained from samples of a small subset of classes, so it will be biased towards these classes [29]. As a result, it may significantly slow down the convergence rate or even fail to converge under CenAsy and DecSyn patterns.

The authors in [21] propose a data-aware clustering method, where workers are clustered based on their data distribution, such that the inter-cluster data distribution is close to IID, as shown in Fig. 2(b). However, under the asynchronous patterns (*i.e.*, CenAsy and DecAsy), the frequency of each aggregator participating in inter-cluster aggregation may be diverse greatly, which makes the global model biased towards the cluster models with high participation frequency [30]. The clustering method [21] may cause a large number of workers to be clustered to low-frequency aggregators, reducing training accuracy. Similarly, under the decentralized patterns (*i.e.*, DecSyn and DecAsy), since the clustering method [21] does not consider the inter-cluster topology, a large number of workers may be clustered to aggregators with sparse links, also reducing training accuracy [18]. The performance of different clustering methods for different inter-cluster patterns is summarized in Table 2.

The rest of this paper is organized as follows. Section 3 introduces the hierarchical aggregation federated learning. Section 4 gives the convergence analysis of four inter-cluster aggregation patterns. A unified clustering algorithm is proposed in Section 5. The Experimental results are shown in Section 6. We conclude this paper in Section 7.

TABLE 3: Key Notations.

Symbol	Semantics
$\mathcal{V}$	The set of workers $\{v_1, v_2, ..., v_N\}$
$\mathcal{S}$	The set of aggregators $\{s_1, s_2, ..., s_M\}$
$\mathcal{V}_j$	The set of workers whose aggregator is s_j
$\mathcal{S}_j$	The set of s_j's neighboring aggregators including itself (only for DecSyn and DecAsy)
$d_i/D_j/D$	The data size on worker v_i/ cluster $\mathcal{V}_j$/ all workers
α_i/β_j	The proportion of the data size of worker v_i/ cluster $\mathcal{V}_j$ to the total data size
ϕ_j^i	The proportion of the data size of worker v_i to the data size of cluster $\mathcal{V}_j$
$f_i/F_j/F$	The loss function of worker v_i/cluster $\mathcal{V}_j$/ global
$f_i^*/F_j^*/F^*$	The optimal value of $f_i/F_j/F$
$\mathbf{w}_t$	The global model at round t
$\mathbf{v}_t^i$	The local model of worker v_i at round t
$\tilde{\mathbf{v}}_t^i$	The trained local model of worker v_i at round t
$\mathbf{w}_t^j$	The cluster model of aggregator s_j at round t
$\tilde{\mathbf{w}}_t^j$	The cluster model of aggregator s_j derived by intra-cluster aggregation at round t
$\hat{\mathbf{w}}_t$	The cluster model that the parameter server receiving at round t (for CenAsy and DecAsy)

3 Hierarchical Aggregation Federated Learning

In this section, we first introduce the concept of federated learning (Section 3.1). Then we describe four procedures of the hierarchical aggregation federated learning architecture (Section 3.2). For ease of expression, some key notations are listed in Table 3.

3.1 Federated Learning (FL)

We perform federated learning over a set of workers $\mathcal{V} = \{v_1, v_2, ..., v_N\}$, with $|\mathcal{V}| = N > 1$. Each worker v_i trains a model on its local dataset $\not{d}_i$, with the size of $d_i \triangleq |\not{d}_i|$. Then the loss function of worker v_i is defined as

$$f_i(\mathbf{w}) \triangleq \frac{1}{d_i} \sum_{\xi \in \not{d}_i} f_i(\mathbf{w}; \xi), \tag{1}$$

where $\mathbf{w}$ is the parameter vector, and $f_i(\mathbf{w}; \xi)$ is the loss over a sample ξ in dataset $\not{d}_i$.

The global dataset over all workers is $\mathcal{D}$, with size $D = |\mathcal{D}| = \sum_{v_i \in \mathcal{V}} d_i$. Let $\alpha_i = d_i/D$ denote the proportion of worker v_i's data size to the total data size. The global loss function on all the distributed datasets is defined as

$$F(\mathbf{w}) \triangleq \sum_{v_i \in \mathcal{V}} \frac{d_i}{D} f_i(\mathbf{w}) = \sum_{v_i \in \mathcal{V}} \alpha_i f_i(\mathbf{w}). \tag{2}$$

The learning problem is to find the optimal parameter vector $\mathbf{w}^*$ so as to minimize $F(\mathbf{w})$, *i.e.*, $\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} F(\mathbf{w})$.

3.2 Hierarchical Aggregation (HA) Federated Learning

We introduce the worker side, the aggregator side and the parameter server side of the HA architecture respectively as described in Alg. 1. The HA architecture consists of four procedures: *worker clustering*, *local training*, *intra-cluster aggregation* and *inter-cluster aggregation*.

3.2.1 Worker Clustering

In addition to a set of workers, the HA architecture includes a set of aggregators $\mathcal{S} = \{s_1, s_2, ..., s_M\}$, with $M = |\mathcal{S}|$. Each worker sends its local model to a selected aggregator. If s_j is the aggregator of worker v_i, $x_{i,j} = 1$; otherwise, $x_{i,j} = 0$. As a result, workers in $\mathcal{V}$ are organized into M clusters $\mathcal{V}_1, ..., \mathcal{V}_M$, satisfying $\bigcup_{j=0}^{M} \mathcal{V}_j = \mathcal{V}$ and $\mathcal{V}_j \bigcap \mathcal{V}_{j'} = \varnothing, \forall j \neq j'$. Let $\mathcal{D}_j$ denote the dateset of cluster $\mathcal{V}_j$, with size $D_j = |\mathcal{D}_j| = \sum_{v_i \in \mathcal{V}_j} d_i = \sum_{v_i \in \mathcal{V}} x_{i,j} d_i$. The loss function on cluster $\mathcal{V}_j$ is defined as

$$F_j(\mathbf{w}) \triangleq \sum_{v_i \in \mathcal{V}_j} \frac{d_i}{D_j} f_i(\mathbf{w}) = \sum_{v_i \in \mathcal{V}_j} \phi_j^i f_i(\mathbf{w}), \tag{3}$$

where ϕ_j^i denotes the proportion of the data size of worker v_i in cluster $\mathcal{V}_j$. It is obviously that

$$F(\mathbf{w}) = \sum_{s_j \in \mathcal{S}} \frac{D_j}{D} F_j(\mathbf{w}) = \sum_{s_j \in \mathcal{S}} \beta_j F_j(\mathbf{w}), \tag{4}$$

where β_j denotes the proportion of the data size of cluster $\mathcal{V}_j$ to the total data size is $\beta_j = D_j / D$.

3.2.2 Local Training

Worker v_i performs local updating at round t by

$$\tilde{\mathbf{v}}_t^i = \mathbf{v}_{t-1}^i - \eta \nabla f_i(\mathbf{v}_{t-1}^i; \xi_{t-1}^i), \tag{5}$$

where η is the learning rate, ∇ is the gradient operator, ξ_{t-1}^i is a sample uniformly chosen from the local data and $\mathbf{v}_{t-1}^i$ is the local model of worker v_i at round $t-1$. Let T'' denote the number of local updating iterations before each intra-cluster aggregation. If t is an integer multiple of T'', $\tilde{\mathbf{v}}_t^i$ is uploaded to its aggregator (Line 5). Otherwise, the local model $\mathbf{v}_t^i$ is set as $\tilde{\mathbf{v}}_t^i$ (Line 9), and then v_i performs local updating for the next round $t+1$.

3.2.3 Intra-cluster Aggregation

Aggregator s_j receives the updated local model $\tilde{\mathbf{v}}_t^i$ from each worker v_i in cluster $\mathcal{V}_j$, and aggregates them as

$$\tilde{\mathbf{w}}_t^j = \sum_{v_i \in \mathcal{V}_j} \frac{d_i}{D_j} \tilde{\mathbf{v}}_t^i = \sum_{v_i \in \mathcal{V}_j} \phi_j^i \tilde{\mathbf{v}}_t^i. \tag{6}$$

Assume that there are T' intra-cluster aggregations on each aggregator before inter-cluster aggregation. If the intra-cluster aggregation index $h = t/T''$ is an integer multiple of T', aggregator s_j performs inter-cluster aggregation to obtain its new cluster models $\mathbf{w}_t^j$ as described in Section 3.2.4. Otherwise, its cluster model $\mathbf{w}_t^j$ is set as $\tilde{\mathbf{w}}_t^j$ (Line 28).

Subsequently, aggregator s_j distributes its cluster model $\mathbf{w}_t^j$ back to the workers in cluster $\mathcal{V}_j$ (Line 29). On the worker side, each worker v_i sets its local model as the received cluster model from its aggregator, *i.e.*, $\mathbf{v}_t^i = \mathbf{w}_t^j, v_i \in \mathcal{V}_j$ (Line 7).

3.2.4 Inter-cluster Aggregation

Aggregators perform inter-cluster aggregation to derive their new cluster models. We introduce four inter-cluster aggregation patterns accordingly.

Centralized-synchronous (CenSyn): This pattern performs synchronous inter-cluster aggregation in the centralized structure. There is a centralized parameter server in the HA architecture. Each aggregator s_j uploads its aggregated cluster model $\tilde{\mathbf{w}}_t^j$ to the parameter server at each inter-cluster aggregation. The parameter server performs global aggregation[1] as

$$\mathbf{w}_t = \sum_{s_j \in \mathcal{S}} \frac{D_j}{D} \tilde{\mathbf{w}}_t^j = \sum_{s_j \in \mathcal{S}} \beta_j \tilde{\mathbf{w}}_t^j, \tag{7}$$

and then distributes the global model $\mathbf{w}_t$ to all aggregators (Line 34). After receiving the global model, each aggregator s_j updates its cluster model, *i.e.*, $\mathbf{w}_t^j = \mathbf{w}_t$ (Line 18).

Centralized-asynchronous (CenAsy): This pattern performs asynchronous inter-cluster aggregation in the centralized structure. On receiving the aggregated cluster model from any aggregator, the parameter server performs global aggregation and returns the updated global model. Let s_{j_r} denote the aggregator participating in the r^{th} global aggre-

Algorithm 1 Hierarchical Aggregation Federated Learning

```
1:  Processing at Each Worker $v_i$
2:  for $t = 1$ to $T$ do
3:      Obtain local model $\tilde{\mathbf{v}}_t^i$ by Eq. (5)
4:      if $t$ mod $T'' == 0$ then
5:          Upload $\tilde{\mathbf{v}}_t^i$ to its aggregator $s_j$
6:          Receive $\mathbf{w}_t^j$ from $s_j$
7:          $\mathbf{v}_t^i = \mathbf{w}_t^j$
8:      else
9:          $\mathbf{v}_t^i = \tilde{\mathbf{v}}_t^i$
10: Processing at Each Aggregators $s_j$
11: for $h = 1$ to $T/T''$ do
12:     Receive $\tilde{\mathbf{v}}_t^i$ from each worker $v_i \in \mathcal{V}_j$
13:     Obtain cluster model $\tilde{\mathbf{w}}_t^j$ by Eq. (6)
14:     if $h$ mod $T' == 0$ then
15:         if Inter-cluster pattern is CenSyn or CenAsy then
16:             Send $\tilde{\mathbf{w}}_t^j$ to the parameter server
17:             Receive $\mathbf{w}_t$ from the parameter server
18:             $\mathbf{w}_t^j = \mathbf{w}_t$
19:         if Inter-cluster pattern is DecSyn then
20:             Send $\tilde{\mathbf{w}}_t^j$ to each aggregator $s_{j'} \in \mathcal{S}_j$
21:             Receive $\tilde{\mathbf{w}}_t^{j'}$ from each aggregator $s_{j'} \in \mathcal{S}_j$
22:             Update model $\mathbf{w}_t^j$ by Eq. (10)
23:         if Inter-cluster pattern is DecAsy then
24:             Obtain $\tilde{\mathbf{w}}_t^{j'}$ of each aggregator $s_{j'} \in \mathcal{S}_j$ from caches
25:             Update model $\mathbf{w}_t^j$ by Eq. (11)
26:             Send $\tilde{\mathbf{w}}_t^j$ to each aggregator $s_{j'} \in \mathcal{S}_j$
27:     else
28:         $\mathbf{w}_t^j = \tilde{\mathbf{w}}_t^j$
29:     Distribute $\mathbf{w}_t^j$ to each worker $v_i \in \mathcal{V}_j$
30: Processing at Parameter Server
31: if Inter-cluster pattern is CenSyn then
32:     Receive $\tilde{\mathbf{w}}_t^j$ from each aggregator $s_j \in \mathcal{S}$
33:     Update global model $\mathbf{w}_t$ by Eq. (7)
34:     Distribute $\mathbf{w}_t$ to each aggregator $s_j \in \mathcal{S}$
35: if Inter-cluster pattern is CenAsy then
36:     Receive $\tilde{\mathbf{w}}_t^j$ from any $s_j$
37:     Update global model $\mathbf{w}_t$ by Eq. (8)
38:     Distribute $\mathbf{w}_t$ to $s_j$
```

1. The term "global aggregation" is equal to "inter-cluster aggregation" for the patterns with centralized inter-cluster structures, *e.g.*, CenSyn and CenAsy.

gation. Obviously, when aggregators perform inter-cluster aggregation, the index of inter-cluster agregations satisfies $r = t/(T' \cdot T'')$. Specifically, on receiving a cluster model $\hat{\mathbf{w}}_t$ from s_{j_r} at round $t = rT'T''$, the parameter server performs global model aggregation as

$$\begin{aligned}\mathbf{w}_t =&(1-\theta_r)\mathbf{w}_{t-T'T''} + \theta_r\hat{\mathbf{w}}_t\\ =&(1-\theta_r)\mathbf{w}_{(r-1)T'T''} + \theta_r\hat{\mathbf{w}}_{rT'T''},\end{aligned} \tag{8}$$

where θ_r is the weight of the received model at the r^{th} global aggregation. The global model $\mathbf{w}_t$ is distributed to aggregator s_{j_r} (Line 38) and set as s_{j_r}'s cluster model subsequently, *i.e.*, $\mathbf{w}_t^{j_r} = \mathbf{w}_t$ (Line 18).

Note that since the aggregators participate in global model aggregation asynchronously, the received cluster model $\hat{\mathbf{w}}_t$ on the parameter server is not necessarily equal to $\tilde{\mathbf{w}}_t^{j_r}$ [31]. Let τ_r be the interval (called the *staleness* [32]) between the current inter-cluster aggregation r, and the last received global model version by aggregator s_{j_r}. The received cluster model actually satisfies

$$\hat{\mathbf{w}}_t = \hat{\mathbf{w}}_{rT'T''} = \tilde{\mathbf{w}}^{j_r}_{(r-\tau_r)T'T''}. \tag{9}$$

Decentralized-synchronous (DecSyn): This pattern performs synchronous inter-cluster aggregation in the decentralized structure. There is no system-wide centralized parameter server in the decentralized inter-cluster structure. As an alternative, each aggregator s_j exchanges its cluster model $\tilde{\mathbf{w}}_t^j$ with its one-hop neighboring aggregators at each inter-cluster aggregation. Let $\mathcal{S}_j$ denote the set of s_j's neighboring aggregators including itself, and $\sigma_j^{j'} = D_{j'}/\sum_{s_{j'}\in\mathcal{S}_j} D_{j'}$ denote the proportion of the data size of cluster $s_{j'}$ to the total data size of clusters in S_j. After receiving other aggregators' cluster models $\tilde{\mathbf{w}}_t^{j'}, \forall s_{j'} \in \mathcal{S}_j$, aggregator s_j performs model aggregation by

$$\mathbf{w}_t^j = \frac{\sum_{s_{j'}\in\mathcal{S}_j} D_{j'}\tilde{\mathbf{w}}_t^{j'}}{\sum_{s_{j'}\in\mathcal{S}_j} D_{j'}} = \sum_{s_{j'}\in\mathcal{S}_j} \sigma_j^{j'}\tilde{\mathbf{w}}_t^{j'}. \tag{10}$$

Decentralized-asynchronous (DecAsy): This pattern performs asynchronous inter-cluster aggregation in the decentralized structure. Unlike the DecSyn pattern where each aggregator needs to wait for cluster models from its neighbour aggregators, under the DecAsy pattern, each aggregator maintains caches for the received cluster models from its neighbouring aggregators. If a new model is received from a neighbour aggregator, the corresponding original model is overwritten. At the r^{th} inter-cluster aggregation, aggregator s_{j_r} aggregates the cluster model $\hat{\mathbf{w}}_t^j, \forall s_j \in \mathcal{S}_{j_r}$ in its caches by

$$\mathbf{w}_t^{j_r} = \sum_{s_j\in\mathcal{S}_{j_r}} \sigma_{j_r}^j \hat{\mathbf{w}}_t^j, \tag{11}$$

and sends $\mathbf{w}_t^{j_r}$ to its neighbour aggregators (Line 26). Similar to the case under CenAsy, let τ_r^j be the staleness between the current inter-cluster aggregation r and the version of the cluster model $\hat{\mathbf{w}}_t^j$ in s_{j_r}'s cache. Thus the cluster model actually satisfies

$$\hat{\mathbf{w}}_t^j = \hat{\mathbf{w}}_{rT'T''}^j = \tilde{\mathbf{w}}^j_{(r-\tau_r^j)T'T''}. \tag{12}$$

4 Convergence Analysis

In this section, we first give several general assumptions in federated learning (Section 4.1). Then we obtain the convergence bounds of the CenSyn, CenAsy, DecSyn and DecAsy patterns through theoretical analysis (Section 4.2).

4.1 Assumptions

We make the following assumptions on the loss functions $f_i(\mathbf{w}), \forall v_i \in \mathcal{V}$ in Eq. (1) for convergence analysis, which are widely used in the existing literatures [8][33].

Assumption 1 (Smoothness): *$f_i(\mathbf{w};\xi)$ is L-smooth for every realization of ξ with $L > 0$, i.e., $\forall \mathbf{w}_1, \mathbf{w}_2$, $\|\nabla f(\mathbf{w}_1;\xi) - \nabla f(\mathbf{w}_2;\xi)\| \le L\|\mathbf{w}_1 - \mathbf{w}_2\|$. Then, by the property of L-smooth function, we have, $\forall \mathbf{w}_1, \mathbf{w}_2$, $f_i(\mathbf{w}_2) - f_i(\mathbf{w}_1) \le \langle \nabla f_i(\mathbf{w}_1), \mathbf{w}_2 - \mathbf{w}_1\rangle + \frac{L}{2}\|\mathbf{w}_2 - \mathbf{w}_1\|^2$.*

Assumption 2 (Strong convexity): *$f_i(\mathbf{w})$ is μ-strongly convex with $\mu > 0$, i.e., $\forall \mathbf{w}_1, \mathbf{w}_2$, $f_i(\mathbf{w}_2) - f_i(\mathbf{w}_1) \ge \langle \nabla f_i(\mathbf{w}_1), \mathbf{w}_2 - \mathbf{w}_1\rangle + \frac{\mu}{2}\|\mathbf{w}_2 - \mathbf{w}_1\|^2$.*

Note that models with convex loss functions, such as linear regression and support vector machines, satisfy Assumption 2. The evaluation results in Section 6 show that our mechanism can also work well for models (*e.g.*, CNN) with non-convex loss functions.

4.2 Analysis of Convergence Bounds

4.2.1 Convergence Analysis of Intra-cluster Aggregation

For ease of expression, we abbreviated $F(\mathbf{w}^*)$ as F^*, which represents the optimal value of the global loss function F. Similarly, F_j^* denotes the optimal value of the loss function F_j on cluster $\mathcal{V}_j$, and f_i^* denotes the optimal value of the loss function f_i on worker v_i.

For the HA architecture, inter-cluster aggregation usually causes more communication consumption than intra-cluster aggregation [23]. So we can reduce the communication consumption by increasing the frequency of intra-cluster aggregation (decreasing T'') while decreasing that of inter-cluster aggregation (increasing T') [34]. To this end, we reasonably set $T'' = 1$ for the rest of this paper [21]. We give the convergence analysis for performing T' intra-cluster aggregations in cluster $\mathcal{V}_j$ by the following lemma.

Lemma 1 : *Taking $\eta < \frac{\mu}{2L^2}$, for $\forall s_j \in \mathcal{S}, t = rT', r \in \{0, 1, ..., R\}$, it holds that*

$$\mathbb{E}[F(\tilde{\mathbf{w}}^j_{rT'})] - F^* \le \rho_0^{T'}(\mathbb{E}[F(\mathbf{w}_{(r-1)T'})] - F^*) + \delta_j,$$

where $\rho_0 = 1 - \mu\eta$, $\delta_j = \frac{1-\rho_0^{T'}}{1-\rho_0}\Gamma_j$ and $\Gamma_j = \frac{\eta}{2}\xi_j + 2L^2\eta^2F_j^* - 2L^2\eta^2\sum_{v_i\in\mathcal{V}_j}\phi_j^i f_i^* + L\eta^2\sum_{v_i\in\mathcal{V}_j}\phi_j^i g_i^*$.

Proof: According to Eqs. (5) and (6), for $\forall s_j \in \mathcal{S}$, it holds that

$$\begin{aligned}\tilde{\mathbf{w}}_t^j =& \sum_{v_i\in\mathcal{V}_j}\phi_j^i\tilde{\mathbf{v}}_t^i\\ =& \sum_{v_i\in\mathcal{V}_j}\phi_j^i(\mathbf{v}_{t-1}^i - \eta\nabla f_i(\mathbf{v}_{t-1}^i;\xi_{t-1}^i))\\ =&\mathbf{w}_{t-1}^j - \eta\sum_{v_i\in\mathcal{V}_j}\phi_j^i\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i).\end{aligned} \tag{13}$$

According to Assumption 1, it is obvious that F is L-smooth. It follows

$$\begin{aligned}F(\tilde{\mathbf{w}}_t^j) - F^* \le& F(\mathbf{w}_{t-1}^j) + \langle\nabla F(\mathbf{w}_{t-1}^j), \tilde{\mathbf{w}}_t^j - \mathbf{w}_{t-1}^j\rangle\\ &+ \frac{L}{2}\|\tilde{\mathbf{w}}_t^j - \mathbf{w}_{t-1}^j\|^2\\ =&F(\mathbf{w}_{t-1}^j) - F^* - \eta\langle\nabla F(\mathbf{w}_{t-1}^j), \sum_{v_i\in\mathcal{V}_j}\phi_j^i\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i)\rangle\end{aligned}$$

$$+ \frac{L\eta^2}{2}\|\sum_{v_i\in\mathcal{V}_j}\phi_j^i\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i)\|^2. \tag{14}$$

We derive the expectation of the third term of Eq. (14) as

$$\begin{aligned}&\mathbb{E}[\langle\nabla F(\mathbf{w}_{t-1}^j), \sum_{v_i\in\mathcal{V}_j}\phi_j^i\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i)\rangle]\\ =&\langle\nabla F(\mathbf{w}_{t-1}^j), \sum_{v_i\in\mathcal{V}_j}\phi_j^i\mathbb{E}[\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i)]\rangle\\ =&\langle\nabla F(\mathbf{w}_{t-1}^j), \nabla F_j(\mathbf{w}_{t-1}^j)\rangle\\ =&\frac{1}{2}\Big(\|\nabla F(\mathbf{w}_{t-1}^j)\|^2+\|\nabla F_j(\mathbf{w}_{t-1}^j)\|^2\\ &-\|\nabla F(\mathbf{w}_{t-1}^j)-\nabla F_j(\mathbf{w}_{t-1}^j)\|^2\Big).\end{aligned} \tag{15}$$

According to Assumption 2, it is obvious that F is μ-strongly convex. It follows

$$\|\nabla F(\mathbf{w}_{t-1}^j)\|^2 \geq 2\mu(F(\mathbf{w}_{t-1}^j)-F^*). \tag{16}$$

By using the AM-GM Inequality and the Jensen's Inequality, we derive the expectation of the last term of Eq. (14) as

$$\begin{aligned}&\mathbb{E}[\|\sum_{v_i\in\mathcal{V}_j}\phi_j^i\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i)\|^2]\\ =&\mathbb{E}[\|\sum_{v_i\in\mathcal{V}_j}\phi_j^i\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i)-\sum_{v_i\in\mathcal{V}_j}\phi_j^i\nabla f_i(\mathbf{v}_i^*;\xi_{t-1}^i)\\ &+\sum_{v_i\in\mathcal{V}_j}\phi_j^i\nabla f_i(\mathbf{v}_i^*;\xi_{t-1}^i)\|^2]\\ \leq&2\mathbb{E}[\|\sum_{v_i\in\mathcal{V}_j}\phi_j^i(\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i)-\nabla f_i(\mathbf{v}_i^*;\xi_{t-1}^i))\|^2]\\ &+2\mathbb{E}[\|\sum_{v_i\in\mathcal{V}_j}\phi_j^i\nabla f_i(\mathbf{v}_i^*;\xi_{t-1}^i)\|^2]\\ \leq&2\sum_{v_i\in\mathcal{V}_j}\phi_j^i\mathbb{E}[\|\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i)-\nabla f_i(\mathbf{v}_i^*;\xi_{t-1}^i)\|^2]\\ &+2\sum_{v_i\in\mathcal{V}_j}\phi_j^i\mathbb{E}[\|\nabla f_i(\mathbf{v}_i^*;\xi_{t-1}^i)\|^2],\end{aligned} \tag{17}$$

where $\mathbf{v}_i^*$ is the optimal solution of f_i. According to Lemma 3 of [35], we have

$$\begin{aligned}&\mathbb{E}[\|\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i)-\nabla f_i(\mathbf{v}_i^*;\xi_{t-1}^i)\|^2]\\ \leq&2L(f_i(\mathbf{w}_{t-1}^j)-f_i^*).\end{aligned} \tag{18}$$

Let $g_i^*=\mathbb{E}[\|\nabla f_i(\mathbf{v}_i^*;\xi_1^i)\|^2]=\mathbb{E}[\|\nabla f_i(\mathbf{v}_i^*;\xi_2^i)\|^2]=...=\mathbb{E}[\|\nabla f_i(\mathbf{v}_i^*;\xi_T^i)\|^2]$, since $\{\xi_t^i\}_{t\geq 0}$ are IID random variables for each worker v_i [35]. Then we have

$$\begin{aligned}&\mathbb{E}[\|\sum_{v_i\in\mathcal{V}_j}\phi_j^i\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i)\|^2]\\ \leq&\sum_{v_i\in\mathcal{V}_j}\phi_j^i[4L(f_i(\mathbf{w}_{t-1}^j)-f_i^*)+2g_i^*]\\ =&4LF_j(\mathbf{w}_{t-1}^j)-4L\sum_{v_i\in\mathcal{V}_j}\phi_j^i f_i^*+2\sum_{v_i\in\mathcal{V}_j}\phi_j^i g_i^*.\end{aligned} \tag{19}$$

Since F_j is also μ-strongly convex for $\forall s_j\in\mathcal{S}$, we have

$$F_j(\mathbf{w}_{t-1}^j)\leq\frac{1}{2\mu}\|\nabla F_j(\mathbf{w}_{t-1}^j)\|^2+F_j^*. \tag{20}$$

Therefore, Eq. (19) can be transformed as

$$\begin{aligned}&\mathbb{E}[\|\sum_{v_i\in\mathcal{V}_j}\phi_j^i\nabla f_i(\mathbf{w}_{t-1}^j;\xi_{t-1}^i)\|^2]\\ \leq&\frac{2L}{\mu}\|F_j(\mathbf{w}_{t-1}^j)\|^2+4LF_j^*-4L\sum_{v_i\in\mathcal{V}_j}\phi_j^i f_i^*+2\sum_{v_i\in\mathcal{V}_j}\phi_j^i g_i^*.\end{aligned} \tag{21}$$

We define $\xi_j\triangleq\max_{t\in[T]}\|\nabla F(\mathbf{w}_t^j)-\nabla F_j(\mathbf{w}_t^j)\|$ as the maximum of the gradient divergence [8][36] of cluster $\mathcal{S}_j$ for $\forall t\in[T]$. By taking Eqs. (15), (16) and (21) into Eq. (14), we derive that

$$\begin{aligned}&\mathbb{E}[F(\tilde{\mathbf{w}}_t^j)]-F^*\leq(1-\mu\eta)(\mathbb{E}[F(\mathbf{w}_{t-1}^j)]-F^*)\\ &-(\frac{\eta}{2}-\frac{L^2\eta^2}{\mu})\|\nabla F_j(\mathbf{w}_{t-1}^j)\|^2+\frac{\eta}{2}\xi_j\\ &+2L^2\eta^2F_j^*-2L^2\eta^2\sum_{v_i\in\mathcal{V}_j}\phi_j^i f_i^*+L\eta^2\sum_{v_i\in\mathcal{V}_j}\phi_j^i g_i^*.\end{aligned} \tag{22}$$

Since $\eta<\frac{\mu}{2L^2}$, we have

$$\mathbb{E}[F(\tilde{\mathbf{w}}_t^j)]-F^*\leq\rho_0(\mathbb{E}[F(\mathbf{w}_{t-1}^j)]-F^*)+\Gamma_j, \tag{23}$$

where $\rho_0=1-\mu\eta$ and $\Gamma_j=\frac{\eta}{2}\xi_j+2L^2\eta^2F_j^*-2L^2\eta^2\sum_{v_i\in\mathcal{V}_j}\phi_j^i f_i^*+L\eta^2\sum_{v_i\in\mathcal{V}_j}\phi_j^i g_i^*$. For $\forall s_j\in\mathcal{S}$, when $t=rT', r\in\{0,1,...,R\}$, it holds that

$$\begin{aligned}\mathbb{E}[F(\tilde{\mathbf{w}}_{rT'}^j)]-F^*\leq&\rho_0(\mathbb{E}[F(\mathbf{w}_{rT'-1}^j)]-F^*)+\Gamma_j\\ =&\rho_0(\mathbb{E}[F(\tilde{\mathbf{w}}_{rT'-1}^j)]-F^*)+\Gamma_j,\end{aligned} \tag{24}$$

where $\mathbf{w}_{rT'-1}^j=\tilde{\mathbf{w}}_{rT'-1}^j$ because the index $rT'-1$ of the current round is not an integer multiple of T'. By using the recursive relation Eq. (24) from round rT' to round $(r-1)T'$, we obtain that

$$\begin{aligned}&\mathbb{E}[F(\tilde{\mathbf{w}}_{rT'}^j)]-F^*\\ \leq&\rho_0(\mathbb{E}[F(\tilde{\mathbf{w}}_{rT'-1}^j)]-F^*)+\Gamma_j\\ \leq&\rho_0^2(\mathbb{E}[F(\tilde{\mathbf{w}}_{rT'-2}^j)]-F^*)+[\rho_0+1]\Gamma_j\\ &...\\ \leq&\rho_0^{T'-1}(\mathbb{E}[F_j(\tilde{\mathbf{w}}_{rT'-T'+1}^j)]-F^*)+\frac{1-\rho_0^{T'-1}}{1-\rho_0}\Gamma_j\\ \leq&\rho_0^{T'}(\mathbb{E}[F_j(\mathbf{w}_{(r-1)T'}^j)]-F^*)+\frac{1-\rho_0^{T'}}{1-\rho_0}\Gamma_j\\ =&\rho_0^{T'}(\mathbb{E}[F_j(\mathbf{w}_{(r-1)T'})]-F^*)+\frac{1-\rho_0^{T'}}{1-\rho_0}\Gamma_j,\end{aligned} \tag{25}$$

where $\mathbf{w}_{(r-1)T'}^j=\mathbf{w}_{(r-1)T'}$ because the cluster models are synchronized to the global model when the index of the current round is an integer multiple of T'. Let $\delta_j=\frac{1-\rho_0^{T'}}{1-\rho_0}\Gamma_j$, we complete the proof. □

Based on Lemma 1, we analyze the convergence bounds of the following four inter-cluster aggregation patterns: CenSyn, CenAsy, DecSyn and DecAsy, respectively.

4.2.2 Convergence Bound of CenSyn

Theorem 1 : *$\mathbf{w}_0$ is the initial global model. After inter-cluster aggregation Eq. (7) is performed $R=T/T'$ times, the trained global model $\mathbf{w}_T$ satisfies*

$$\mathbb{E}[F(\mathbf{w}_T)]-F^*\leq\rho^T(F(\mathbf{w}_0)-F^*)+\delta,$$

where $\rho=1-\mu\eta$, $\delta=\frac{1-\rho^T}{1-\rho}\Gamma$ and $\Gamma=\frac{\eta}{2}\sum_{s_j\in\mathcal{S}}\beta_j\xi_j+2L^2\eta^2\sum_{s_j\in\mathcal{S}}\beta_jF_j^*-2L^2\eta^2\sum_{v_i\in\mathcal{V}}\alpha_if_i^*+L\eta^2\sum_{v_i\in\mathcal{V}}\alpha_ig_i^*$

Proof: Since F is convex and $\beta_j\in(0,1]$, according to Lemma 1, we can deduce that

$$\begin{aligned}&\mathbb{E}[F(\mathbf{w}_T)]-F^*=\mathbb{E}[F(\mathbf{w}_{RT'})]-F^*\\ \leq&\sum\nolimits_{s_j\in\mathcal{S}}\beta_j(\mathbb{E}[F(\tilde{\mathbf{w}}_{RT'}^j)]-F^*))\\ \leq&\rho_0^{T'}\sum_{s_j\in\mathcal{S}}\beta_j(\mathbb{E}[F(\mathbf{w}_{(R-1)T'})]-F^*)+\sum_{s_j\in\mathcal{S}}\beta_j\delta_j\\ =&\rho_0^{T'}(\mathbb{E}[F(\mathbf{w}_{(R-1)T'})]-F^*)+\frac{1-\rho_0^{T'}}{1-\rho_0}\Gamma,\end{aligned} \tag{26}$$

where $\Gamma = \frac{\eta}{2}\sum_{s_j\in\mathcal{S}}\beta_j\xi_j + 2L^2\eta^2\sum_{s_j\in\mathcal{S}}\beta_j F_j^* - 2L^2\eta^2\sum_{v_i\in\mathcal{V}}\alpha_i f_i^* + L\eta^2\sum_{v_i\in\mathcal{V}}\alpha_i g_i^*$. Thus we can obtain the difference between $F(\mathbf{w}_T)$ and F^* by the global updating of R times in Eq. (7) as follows

$$\begin{aligned}&\mathbb{E}[F(\mathbf{w}_T)] - F^*\\ \le&\rho_0^{T'}(\mathbb{E}[F(\mathbf{w}_{(R-1)T'})] - F^*) + \frac{1-\rho_0^{T'}}{1-\rho_0}\Gamma\\ \le&\rho_0^{2T'}(\mathbb{E}[F(\mathbf{w}_{(R-2)T'})] - F^*) + (\rho_0^{T'}+1)\frac{1-\rho_0^{T'}}{1-\rho_0}\Gamma\\ &\cdots\\ \le&\rho_0^{RT'}(F(\mathbf{w}_0) - F^*) + \frac{1-\rho_0^{RT'}}{1-\rho_0^{T'}}\frac{1-\rho_0^{T'}}{1-\rho_0}\Gamma\\ =&\rho^T(F(\mathbf{w}_0) - F^*) + \delta,\end{aligned} \tag{27}$$

where $\rho = \rho_0 = 1 - \mu\eta$ and $\delta = \frac{1-\rho^T}{1-\rho}\Gamma$. □

4.2.3 Convergence Bound of CenAsy

Before convergence analysis, we first state a key lemma for our statement. For ease of expression, we denote $\omega_r = r - \tau_r - 1 \ge 0$ as the version of the received global model on s_{j_r} before the rth global aggregation. $\tau_{max} = \max_r\{\tau_r\}$ denotes the maximum staleness.

Lemma 2 [37]: *Let $Q(r)$ be a sequence of real numbers for $r \ge 0$. x, y and z are three nonnegative constants, satisfying $x + y < 1$. If $Q(r) \le xQ(r-1) + yQ(\omega_r) + z$, then*

$$Q(r) \le \rho^r Q(0) + \delta, \tag{28}$$

where $\rho = (x+y)^{\frac{1}{1+\tau_{max}}}$ and $\delta = \frac{z}{1-x-y}$.

We prove this lemma by mathematical induction. The details are provided in Appendix A.

Next, we derive the specific values of x, y and z in the CenAsy pattern and obtain the convergence bound of CenAsy by applying Lemma 2. For asynchronous inter-cluster aggregation, we denote ψ_j as the relative frequency of aggregator s_j participating in the global aggregation, satisfying $\sum_{s_j\in\mathcal{S}}\psi_j = 1$. In addition, we reasonably set $\theta_r = \beta_{j_r} = \frac{D_{j_r}}{D}$ in Eq. (8) similar to the existing works [30][38], since the larger the volume of data in cluster $\mathcal{V}_{j_r}$, the greater its impact on the global model.

Theorem 2 : *$\mathbf{w}_0$ is the initial global model. After inter-cluster aggregation Eq. (8) is performed $R = T/T'$ times, the trained global model $\mathbf{w}_T$ satisfies*

$$\mathbb{E}[F(\mathbf{w}_T)] - F^* \le \rho^T(F(\mathbf{w}_0) - F^*) + \delta,$$

where $\rho = [1-(1-\rho_0^{T'})\sum_{s_j\in\mathcal{S}}\psi_j\beta_j]^{\frac{1}{1+\tau_{max}}}$, $\delta = \frac{\sum_{s_j\in\mathcal{S}}\psi_j\beta_j\Gamma_j}{\mu\eta\sum_{s_j\in\mathcal{S}}\psi_j\beta_j}$ and $\rho_0 = 1 - \mu\eta$.

Proof: Combining Eq. (9) and Lemma 1, for $\forall t = rT', r \in \{0, 1, ..., R\}$, it holds that

$$\begin{aligned}\mathbb{E}[F(\hat{\mathbf{w}}_t)] - F^* =&\mathbb{E}[F(\hat{\mathbf{w}}_{rT'})] - F^*\\ =&\mathbb{E}[F(\tilde{\mathbf{w}}^{j_r}_{(r-\tau_r)T'})] - F^*\\ \le&\rho_0^{T'}(\mathbb{E}[F(\mathbf{w}_{(r-\tau_r-1)T'})] - F^*) + \delta_{j_r}\\ \le&\rho_0^{T'}(\mathbb{E}[F(\mathbf{w}_{\omega_r T'})] - F^*) + \delta_{j_r},\end{aligned} \tag{29}$$

By the global aggregation Eq. (8) at the parameter server, we can deduce that

$$\begin{aligned}&F(\mathbf{w}_{rT'}) - F^*\\ =&F((1-\beta_{j_r})\mathbf{w}_{(r-1)T'} + \beta_{j_r}\hat{\mathbf{w}}_{rT'}) - F^*\\ \le&(1-\beta_{j_r})F(\mathbf{w}_{(r-1)T'}) + \beta_{j_r}F(\hat{\mathbf{w}}_{rT'}) - F^*\\ =&(1-\beta_{j_r})(F(\mathbf{w}_{(r-1)T'}) - F^*) + \beta_{j_r}(F(\hat{\mathbf{w}}_{rT'}) - F^*).\end{aligned} \tag{30}$$

Combining Eqs. (29) and (30), the expectation of the difference between $F(\mathbf{w}_{rT'})$ and F^* can be calculated as

$$\begin{aligned}\mathbb{E}[F(\mathbf{w}_{rT'})] - F^* \le& (1 - \sum_{s_j\in\mathcal{S}}\psi_j\beta_j)(\mathbb{E}[F(\mathbf{w}_{(r-1)T'})] - F^*)\\ &+ \sum_{s_j\in\mathcal{S}}\psi_j\beta_j\rho_0^{T'}(\mathbb{E}[F(\mathbf{w}_{\omega_r T'})] - F^*) + \frac{1-\rho_0^{T'}}{1-\rho_0}\sum_{s_j\in\mathcal{S}}\psi_j\beta_j\Gamma_j.\end{aligned} \tag{31}$$

Let $Q(r) = \mathbb{E}[F(\mathbf{w}_{rT'})] - F^*$. Then $\mathbb{E}[F(\mathbf{w}_{(r-1)T'})] - F^* = Q(r-1)$ and $\mathbb{E}[F(\mathbf{w}_{\omega_r T'})] - F^* = Q(\omega_r)$. The recursive relation in Eq.(31) is transformed into

$$\begin{aligned}Q(r) \le& \underbrace{(1 - \sum_{s_j\in\mathcal{S}}\psi_j\beta_j)}_{x}Q(r-1)\\ &+ \underbrace{\sum_{s_j\in\mathcal{S}}\psi_j\beta_j\rho_0^{T'}}_{y}Q(\omega_r) + \underbrace{\frac{1-\rho_0^{T'}}{1-\rho_0}\sum_{s_j\in\mathcal{S}}\psi_j\beta_j\Gamma_j}_{z}.\end{aligned} \tag{32}$$

According to Lemma 2, if $(1-\rho_0^{T'})\sum_{s_j\in\mathcal{S}}\psi_j\beta_j \in (0, 1)$, then

$$\mathbb{E}[F(\mathbf{w}_T)] - F^* \le \rho^T(F(\mathbf{w}_0) - F^*) + \delta, \tag{33}$$

where $\rho = [1-(1-\rho_0^{T'})\sum_{s_j\in\mathcal{S}}\psi_j\beta_j]^{\frac{1}{1+\tau_{max}}}$ and $\delta = \frac{\sum_{s_j\in\mathcal{S}}\psi_j\beta_j\Gamma_j}{\mu\eta\sum_{s_j\in\mathcal{S}}\psi_j\beta_j}$. □

4.2.4 Convergence Bound of DecSyn

Since no global model is maintained in the DecSyn pattern, we analyze the weighted model $\overline{\mathbf{w}}_T = \sum_{s_j\in\mathcal{S}}\beta_j\mathbf{w}_T^j$ of all cluster models. For ease of expression, we define several matrices. $\mathbf{B} = [\beta_1, ...\beta_M]$ is the vector representing the proportion of the clusters' data sizes. Recall that $\sigma_j^{j'} = D_{j'}/\sum_{s_{j'}\in\mathcal{S}_j}D_{j'}$ denote the proportion of the data size of cluster $s_{j'}$ to the total data size of clusters in S_j. Then we define

$$\mathbf{S} = \begin{pmatrix}\sigma_1^1 & \cdots & \sigma_1^M\\ \vdots & \ddots & \vdots\\ \sigma_M^1 & \cdots & \sigma_M^M\end{pmatrix} \tag{34}$$

as the data size weight matrix of the neighboring clusters.

Before analyzing the convergence of DecSyn, we first state a lemma to describe the quantitative relations that two matrices $\mathbf{B}$ and $\mathbf{S}$ satisfy.

Lemma 3 : *$\mathbf{BS}^r\mathbf{1}_M = 1$ for $\forall r \ge 0$, where $\mathbf{1}_M = [1, ..., 1]^\top$ denotes a vector with M dimensions all 1's.*

We prove Lemma 3 by mathematical induction. The details are provided in Appendix B. Then we analyse the convergence bound of DecSyn pattern as follows.

Theorem 3 : *$\mathbf{w}_0$ is the initial model on each worker. After inter-cluster aggregation Eq. (10) is performed $R = T/T'$ times, the weighted model $\overline{\mathbf{w}}_T$ satisfies*

$$\mathbb{E}[F(\overline{\mathbf{w}}_T)] - F^* \le \rho^T(F(\mathbf{w}_0) - F^*) + \delta,$$

where $\rho = 1 - \mu\eta$, $\delta = \sum_{r=0}^{R-1}\rho^{rT'}\mathbf{BS}^{r+1}\mathbf{\Delta}$ and $\mathbf{\Delta} = \frac{1-\rho^{T'}}{1-\rho}[\Gamma_1, \Gamma_2, ...\Gamma_M]^\top$.

Proof: According to Lemma 1, for $\forall s_j \in \mathcal{S}, t = rT', r \in \{0, 1, ..., R\}$, it holds that

$$\mathbb{E}[F(\tilde{\mathbf{w}}_t^j)] - F(\mathbf{w}^*) = \mathbb{E}[F(\tilde{\mathbf{w}}_{rT'}^j)] - F(\mathbf{w}^*)$$

$$\leq \rho^{T'}(\mathbb{E}[F(\mathbf{w}^j_{(r-1)T'})] - F(\mathbf{w}^*)) + \delta_j, \quad (35)$$

where $\rho = \rho_0 = 1 - \mu\eta$. Combining Eqs. (10) and (35), we deduce that

$$\begin{aligned}&\mathbb{E}[F(\mathbf{w}^j_{rT'})] - F(\mathbf{w}^*)\\ \leq &\sum_{s_{j'} \in \mathcal{S}_j} \sigma_j^{j'}(\mathbb{E}[F(\tilde{\mathbf{w}}^{j'}_{rT'})] - F(\mathbf{w}^*))\\ \leq &\rho^{T'} \sum_{s_{j'} \in \mathcal{S}_j} \sigma_j^{j'}(\mathbb{E}[F(\mathbf{w}^{j'}_{(r-1)T'})] - F(\mathbf{w}^*)) + \sum_{s_{j'} \in \mathcal{S}_j} \sigma_j^{j'} \delta_{j'}.\end{aligned} \quad (36)$$

Let $Q_j(r) = \mathbb{E}[F(\mathbf{w}^j_{rT'})] - F^*$ and $\mathbf{Q}(r) = [Q_1(r), ..., Q_M(r)]^\top$. The recursive relation for $\forall s_j \in \mathcal{S}$ is transformed into

$$\mathbf{Q}(r) \leq \rho^{T'}\mathbf{S}\mathbf{Q}(r-1) + \mathbf{S}\boldsymbol{\Delta}, \quad (37)$$

where $\boldsymbol{\Delta} = \frac{1-\rho^{T'}}{1-\rho}[\Gamma_1, \Gamma_2, ...\Gamma_M]^\top$. By iterating Eq. (37) for R times, we have

$$\begin{aligned}\mathbf{Q}(R) \leq &\rho^{T'}\mathbf{S}\mathbf{Q}(R-1) + \mathbf{S}\boldsymbol{\Delta}\\ \leq &\rho^{2T'}\mathbf{S}^2\mathbf{Q}(R-2) + [\rho^{T'}\mathbf{S} + \mathbf{E}]\mathbf{S}\boldsymbol{\Delta}\\ &...\\ \leq &\rho^{RT'}\mathbf{S}^R\mathbf{Q}(0) + \sum_{r=0}^{R-1}\rho^{rT'}\mathbf{S}^{r+1}\boldsymbol{\Delta}.\end{aligned} \quad (38)$$

For $\forall s_j \in \mathcal{S}$, it follows $Q_j(0) = F(\mathbf{w}_0^j) - F^* = F(\mathbf{w}_0) - F^*$, so $\mathbf{Q}(0) = \mathbf{1}_M(F(\mathbf{w}_0) - F^*)$. Therefore, the weighted model of clusters satisfies

$$\begin{aligned}&\mathbb{E}[F(\overline{\mathbf{w}}_T)] - F^*\\ \leq &\sum_{s_j \in \mathcal{S}} \beta_j(\mathbb{E}[F(\mathbf{w}^j_{RT'})] - F^*)\\ = &\sum_{s_j \in \mathcal{S}} \beta_j Q_j(R)\\ = &\mathbf{B}\mathbf{Q}(R)\\ \leq &\rho^T\mathbf{B}\mathbf{S}^R\mathbf{1}_M(F(\mathbf{w}_0) - F^*) + \sum_{r=0}^{R-1}\rho^{rT'}\mathbf{B}\mathbf{S}^{r+1}\boldsymbol{\Delta}.\end{aligned} \quad (39)$$

By Lemma 3, $\mathbf{B}\mathbf{S}^R\mathbf{1}_M = 1$. So, we have

$$\mathbb{E}[F(\overline{\mathbf{w}}_T)] - F^* \leq \rho^T(F(\mathbf{w}_0) - F^*) + \sum_{r=0}^{R-1}\rho^{rT'}\mathbf{B}\mathbf{S}^{r+1}\boldsymbol{\Delta}. \quad (40)$$

Setting $\delta = \sum_{r=0}^{R-1}\rho^{rT'}\mathbf{B}\mathbf{S}^{r+1}\boldsymbol{\Delta}$, we complete the proof. □

4.2.5 Convergence Bound of DecAsy

For ease of expression, we construct three sequences of matrices $\mathbf{X}(r)$, $\mathbf{Y}_j(r)$ and $\mathbf{Z}(r)$ for $r \geq 1$, where $\mathbf{X}(r)$ and $\mathbf{Y}_j(r)$ are $M \times M$ diagonal matrices, and $\mathbf{Z}(r)$ is a M-dimensional vector. Specifically, the j_r^{th} diagonal element of $\mathbf{X}(r)$ is x_r, and others are 1; the j_r^{th} diagonal element of $\mathbf{Y}_j(r)$ is y_j^r, and others are 0; the j_r^{th} element of $\mathbf{Z}(r)$ is z_r, and others are 0. That is,

$$\mathbf{X}(r) = \begin{pmatrix}1 & 0 & \cdots & \cdots & 0\\ \vdots & \ddots & & & \vdots\\ 0 & & x_r & & 0\\ \vdots & & & \ddots & \vdots\\ 0 & 0 & \cdots & \cdots & 1\end{pmatrix},$$

$$\mathbf{Y}_j(r) = \begin{pmatrix}0 & 0 & \cdots & \cdots & 0\\ \vdots & \ddots & & & \vdots\\ 0 & & y_j^r & & 0\\ \vdots & & & \ddots & \vdots\\ 0 & 0 & \cdots & \cdots & 0\end{pmatrix},$$

and $\mathbf{Z}(r) = [0, 0, ...z_r, ...0]^\top$. Note that x_r, y_j^r and z_r are are on the same row in $\mathbf{X}(r)$, $\mathbf{Y}_j(r)$ and $\mathbf{Z}(r)$, respectively, satisfying $\theta_r = x_r + \sum_{s_j \in S_{j_r}} y_r^j \leq 1$. Similar to the case under CenAsy, we first state a key lemma for our statement. Let $\omega_r^j = r - \tau_r^j - 1 \geq 0$, $\tau_{max} = \max_{r,j}\{\tau_r^j\}$ and $\theta_{max} = \max_r\{\theta_r\}$.

Lemma 4 : *Let $\mathbf{Q}(r)$ be a sequence of matrices for $r \geq 0$. If $\mathbf{Q}(r) \leq \mathbf{X}(r)\mathbf{Q}(r-1) + \sum_{s_j \in \mathcal{S}_{j_r}} \mathbf{Y}_j(r)\mathbf{Q}(\omega_r^j) + \mathbf{Z}(r)$, then*

$$\mathbf{Q}(r) \leq \prod_{i=0}^{r}\mathbf{P}(i)\mathbf{Q}(0) + \sum_{i=0}^{r}\boldsymbol{\Delta}(i), \quad (41)$$

where $\mathbf{P}(r)$ is a $M \times M$ diagonal matrix and its j_r^{th} diagonal element is ${\theta_{max}}^{\frac{1}{1+\tau_{max}}}$, and others are 1. That is,

$$\mathbf{P}(r) = \begin{pmatrix}1 & 0 & \cdots & \cdots & 0\\ \vdots & \ddots & & & \vdots\\ 0 & & {\theta_{max}}^{\frac{1}{1+\tau_{max}}} & & 0\\ \vdots & & & \ddots & \vdots\\ 0 & 0 & \cdots & \cdots & 1\end{pmatrix},$$

$\boldsymbol{\Delta}(r)$ is a M-dimensional vector

$$\boldsymbol{\Delta}(r) = \begin{cases}[0, 0, ..., 0]^\top, r = 0\\ (\mathbf{X}(r) + \sum\limits_{s_j \in \mathcal{S}_r}\mathbf{Y}(r) - \mathbf{E})\sum\limits_{i=0}^{r-1}\boldsymbol{\Delta}(i) + \mathbf{Z}(r), r \geq 1.\end{cases}$$

We prove this lemma by mathematical induction. The details are provided in Appendix C.

Theorem 4 : *$\mathbf{w}_0$ is the initial model on each worker. After inter-cluster aggregation Eq. (11) is performed $R = T/T'$ times, the weighted model $\overline{\mathbf{w}}_T$ satisfies*

$$\mathbb{E}[F(\overline{\mathbf{w}}_T)] - F^* \leq \kappa(F(\mathbf{w}_0) - F^*) + \delta,$$

where $\kappa = \sum_{s_j \in \mathcal{S}} \beta_j \rho_0^{\frac{\psi_j T}{1+\tau_{max}}}$, $\delta = \mathbf{L} \odot \mathbf{B}\mathbf{S}\boldsymbol{\Delta}$, $\mathbf{L} = [\frac{1-\rho_0^{\psi_1 T}}{1-\rho_0^{T'}}, \frac{1-\rho_0^{\psi_2 T}}{1-\rho_0^{T'}}, ..., \frac{1-\rho_0^{\psi_M T}}{1-\rho_0^{T'}}]$, $\mathbf{B} = [\beta_1, ...\beta_M]$, $\boldsymbol{\Delta} = \frac{1-\rho_0^{T'}}{1-\rho_0}[\Gamma_1, \Gamma_2, ...\Gamma_M]^\top$ and $\rho_0 = 1 - \mu\eta$. $\odot$ is the Hadamard product symbol.

Proof: According to Lemma 1 and Eq. (12), for $\forall s_j \in \mathcal{S}_{j_r}, t = rT', r \in \{0, 1, ..., R\}$, it holds that

$$\begin{aligned}\mathbb{E}[F(\hat{\mathbf{w}}_t^j)] - F^* = &\mathbb{E}[F(\tilde{\mathbf{w}}^j_{(r-\tau_r^j)T'})] - F^*\\ \leq &\rho_0^{T'}(\mathbb{E}[F(\mathbf{w}^j_{(r-\tau_r^j-1)T'})] - F^*) + \delta_j.\end{aligned} \quad (42)$$

Combining Eqs. (11) and (42), we derive that

$$\begin{aligned}&\mathbb{E}[F(\mathbf{w}_t^{j_r})] - F^* = \mathbb{E}[F(\mathbf{w}^{j_r}_{rT'})] - F^*\\ \leq &\sum_{s_j \in \mathcal{S}_{j_r}} \sigma^j_{j_r}(\mathbb{E}[F(\tilde{\mathbf{w}}^j_{(r-\tau_r^j)T'})] - F^*)\\ \leq &\rho_0^{T'}\sum_{s_j \in \mathcal{S}_{j_r}} \sigma^j_{j_r}(\mathbb{E}[F(\mathbf{w}^j_{(r-\tau_r^j-1)T'})] - F^*) + \sum_{s_j \in \mathcal{S}_{j_r}} \sigma^j_{j_r}\delta_j.\end{aligned} \quad (43)$$

Let $Q_j(r) = F(\mathbf{w}^j_{rT'}) - F^*$ and $\mathbf{Q}(r) = [Q_1(r), ..., Q_M(r)]^\top$. The recursive relation for $\forall s_j \in \mathcal{S}$ is transformed into

$$\mathbf{Q}(r) \leq \mathbf{X}(r)\mathbf{Q}(r-1) + \sum_{s_j \in \mathcal{S}_r}\mathbf{Y}_j(r)\mathbf{Q}(\omega_r^j) + \mathbf{Z}(r), \quad (44)$$

where

$$\mathbf{X}(r) = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & & & \vdots \\ 0 & & \rho_0^{T'}\sigma_{j_r}^{j_r} & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & 1 \end{pmatrix},$$

$$\mathbf{Y}_j(r) = \begin{pmatrix} 0 & 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & & & \vdots \\ 0 & & \rho_0^{T'}\sigma_{j_r}^{j} & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & 0. \end{pmatrix},$$

and $\mathbf{Z}(r) = [0, 0, ... \sum_{s_j \in \mathcal{S}_{j_r}} \sigma_{j_r}^{j}\delta_j, ..., 0]^\top$. According to Lemma 4,

$$\mathbf{Q}(R) \leq \prod_{r=0}^{R} \mathbf{P}(r)\mathbf{Q}(0) + \sum_{r=0}^{R} \mathbf{\Delta}(r), \quad (45)$$

where $\mathbf{P}(r)$ is a $M \times M$ diagonal matrix. Specifically, the j_r^{th} diagonal element of $\mathbf{P}(r)$ is $(\rho_0^{T'} \sum_{s_j \in \mathcal{S}_{j_r}} \sigma_{j_r}^{j})^{\frac{1}{1+\tau_{max}}} = \rho_0^{\frac{T'}{1+\tau_{max}}}$, and others are 1. That is,

$$\mathbf{P}(r) = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & & & \vdots \\ 0 & & \rho_0^{\frac{T'}{1+\tau_{max}}} & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & 1 \end{pmatrix}.$$

$\mathbf{\Delta}(r)$ satisfies the following recursive relation:

$$\mathbf{\Delta}(r) = \begin{cases} [0, 0, ..., 0]^\top, r = 0 \\ \begin{pmatrix} 0 & 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & & & \vdots \\ 0 & & \rho_0^{T'} & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & 0 \end{pmatrix} \sum_{i=0}^{r-1} \mathbf{\Delta}(i) + \begin{pmatrix} 0 \\ \vdots \\ \sum_{s_j \in \mathcal{S}_{j_r}} \sigma_{j_r}^{j}\delta_j \\ \vdots \\ 0 \end{pmatrix} \\ \qquad , r \geq 1. \end{cases} \quad (46)$$

From Eq. (46), we can derive that

$$\mathbf{B}\sum_{r=0}^{R} \mathbf{\Delta}(r) = [\beta_1, ..., \beta_M] \begin{pmatrix} \frac{1-\rho_0^{\psi_1 T}}{1-\rho_0^{T'}} \sum_{s_j \in \mathcal{S}_1} \sigma_1^j \delta_j \\ \frac{1-\rho_0^{\psi_2 T}}{1-\rho_0^{T'}} \sum_{s_j \in \mathcal{S}_2} \sigma_2^j \delta_j \\ \vdots \\ \frac{1-\rho_0^{\psi_M T}}{1-\rho_0^{T'}} \sum_{s_j \in \mathcal{S}_M} \sigma_M^j \delta_j \end{pmatrix}$$

$$= \sum_{s_j \in \mathcal{S}} \beta_j \frac{1-\rho_0^{T'\psi_j T}}{1-\rho_0^{T'}} \sum_{s_{j'} \in \mathcal{S}_j} \sigma_j^{j'} \delta_{j'}$$

$$= \mathbf{L} \odot \mathbf{BS\Delta}, \quad (47)$$

where $\mathbf{\Delta} = \frac{1-\rho_0^{T'}}{1-\rho_0}[\Gamma_1, \Gamma_2, ...\Gamma_M]^\top$. Similar to the case under DecSyn, we have

$$\mathbb{E}[F(\overline{\mathbf{w}}_T)] - F^* \leq \sum_{s_j \in \mathcal{S}} \beta_j (\mathbb{E}[F(\mathbf{w}_{RT'}^j)] - F^*)$$

$$= \sum_{s_j \in \mathcal{S}} \beta_j Q_j(R)$$

$$= \mathbf{BQ}(R)$$

$$\leq \mathbf{B}\prod_{r=0}^{R} \mathbf{P}(r)\mathbf{1}_M(F(\mathbf{w}_0) - F^*) + \mathbf{B}\sum_{r=0}^{R} \mathbf{\Delta}(r)$$

$$= \sum_{s_j \in \mathcal{S}} \beta_j \rho_0^{\frac{\psi_j T}{1+\tau_{max}}} (F(\mathbf{w}_0) - F^*) + \mathbf{L} \odot \mathbf{BS\Delta}. \quad (48)$$

Setting $\kappa = \sum_{s_j \in \mathcal{S}} \beta_j \rho_0^{\frac{\psi_j T}{1+\tau_{max}}}$ and $\delta = \mathbf{L} \odot \mathbf{BS\Delta}$, we complete the proof. □

5 Unified Clustering Algorithm

In this section, we first give the unified form of the convergence bounds for different inter-cluster aggregation patterns (Section 5.1), and then propose a time-sharing strategy for intra-cluster aggregation (Section 5.2). Based on both the discussions of convergence bounds and the proposed intra-cluster aggregation strategy, we describe the problem formulation (Section 5.3) and propose a unified clustering algorithm for accelerating HA (Section 5.4).

5.1 Unified Form of the Convergence Bounds

Recall that $x_{i,j}$ is the indicator for whether worker v_i belongs to cluster $\mathcal{V}_j$ or not. If s_j is the aggregator of worker v_i, $x_{i,j} = 1$; otherwise, $x_{i,j} = 0$. The clustering strategy in the whole edge network is denoted as $\boldsymbol{x} = \{x_{i,j}\}_{v_i \in \mathcal{V}, s_j \in \mathcal{S}}$. Let $\mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T)$ denote the convergence bound after T rounds of training from the initial model $\mathbf{w}_0$ under the clustering strategy $\boldsymbol{x}$. Therefore, according to Theorems 1, 2, 3 and 4, the convergence bound of four inter-cluster aggregation patterns can be uniformly expressed as

$$\mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T) \triangleq \kappa(\boldsymbol{x})(F(\mathbf{w}_0) - F^*) + \delta(\boldsymbol{x}). \quad (49)$$

where $\kappa(\boldsymbol{x})$ is called the convergence factor and represents the convergence rate of the loss function, and $\delta(\boldsymbol{x})$ is called the residual error, represents that the loss function can converge to a δ-neighborhood of the optimal value.

Specifically, the specific expressions of $\kappa(\boldsymbol{x})$ and $\delta(\boldsymbol{x})$ under CenSyn, CenAsy, DecSyn and DecAsy patterns can be obtained by Theorems 1, 2, 3 and 4 respectively, which are concluded in Table 4.

TABLE 4: $\kappa(\boldsymbol{x})$ and $\delta(\boldsymbol{x})$ under different patterns.

Pattern	$\kappa(\boldsymbol{x})$
CenSyn	$(1-\mu\eta)^T$
CenAsy	$\{1-[1-(1-\mu\eta)^{T'}]\sum_{s_j \in \mathcal{S}} \psi_j \beta_j(\boldsymbol{x})\}^{\frac{T}{1+\tau_{max}}}$
DecSyn	$(1-\mu\eta)^T$
DecAsy	$\sum_{s_j \in \mathcal{S}} \beta_j(\boldsymbol{x})(1-\mu\eta)^{\frac{\psi_j T}{1+\tau_{max}}}$
Pattern	$\delta(\boldsymbol{x})$
CenSyn	$\frac{1-(1-\mu\eta)^T}{\mu\eta}\sum_{s_j \in \mathcal{S}} \beta_j(\boldsymbol{x})\Gamma_j(\boldsymbol{x})$
CenAsy	$\frac{\sum_{s_j \in \mathcal{S}} \psi_j \beta_j(\boldsymbol{x})\Gamma_j(\boldsymbol{x})}{\mu\eta \sum_{s_j \in \mathcal{S}} \psi_j \beta_j(\boldsymbol{x})}$
DecSyn	$\sum_{r=0}^{R-1}(1-\mu\eta)^{rT'}\mathbf{B}(\boldsymbol{x})[\mathbf{S}(\boldsymbol{x})]^{r+1}\mathbf{\Delta}(\boldsymbol{x})$
DecAsy	$\mathbf{L} \odot \mathbf{B}(\boldsymbol{x})\mathbf{S}(\boldsymbol{x})\mathbf{\Delta}(\boldsymbol{x})$

Without causing confusion, we denote $\Gamma_j(\boldsymbol{x})$, $\mathbf{\Delta}(\boldsymbol{x})$, $\beta_j(\boldsymbol{x})$, $\mathbf{B}(\boldsymbol{x})$ and $\mathbf{S}(\boldsymbol{x})$ as the values or matrices of Γ_j, Δ, β_j, $\mathbf{B}$ and $\mathbf{S}$ under the clustering strategy $\boldsymbol{x}$, respectively.

Obviously, the convergence bound $\mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T)$ is subject to the convergence factor $\kappa(\boldsymbol{x})$ and the residual error $\delta(\boldsymbol{x})$, whose values may depend on the clustering strategy $\boldsymbol{x}$. Then we can draw some meaningful corollaries as follows.

5.1.1 The Impact of $\rho(\boldsymbol{x})$

For the CenSyn and DecSyn patterns, the convergence factor $\kappa(\boldsymbol{x})$ is independent of the clustering strategy $\boldsymbol{x}$, *i.e.*, $\kappa(\boldsymbol{x}) = (1-\mu\eta)^T$. However, $\kappa(\boldsymbol{x})$ is related to $\boldsymbol{x}$ for CenAsy and DecAsy patterns, *i.e.*, $\{1-[1-(1-\mu\eta)^{T'}]\sum_{s_j\in\mathcal{S}}\psi_j\beta_j(\boldsymbol{x})\}^{\frac{T}{1+\tau_{max}}}$ and $\kappa(\boldsymbol{x}) = \sum_{s_j\in\mathcal{S}}\beta_j(\boldsymbol{x})(1-\mu\eta)^{\frac{\psi_j T}{1+\tau_{max}}}$ for CenAsy and DecAsy, respectively. We can draw the following corollary.

Corollary 1 : *For the CenAsy and DecAsy patterns, if the participation frequency of aggregators s_1 and s_2 satisfies $\psi_1 < \psi_2$, the convergence factor $\kappa(\boldsymbol{x})$ can be reduced by clustering some workers from s_1 to s_2.*

We prove Corollary 1 by comparing the value of $\kappa(\boldsymbol{x})$ before and after clustering some workers from s_1 to s_2. The details are provided in Appendix D. Corollary 1 shows that workers can be clustered to high-frequency aggregators as much as possible to accelerate convergence for the CenAsy and DecAsy patterns. For example, without considering communication resource constraints, clustering all workers to the aggregator with the highest participation frequency will achieve the fastest convergence rate (minimum κ). However, if workers are clustered to aggregators farther away from them, it may decelerate FL training instead due to long communication time, which will be further discussed in Section 5.3.

5.1.2 The Impact of $\delta(\boldsymbol{x})$

Recall that $\Gamma_j = \frac{\eta}{2}\xi_j + 2L^2\eta^2F_j^* - 2L^2\eta^2\sum_{v_i\in\mathcal{V}_j}\phi_j^i f_i^* + L\eta^2\sum_{v_i\in\mathcal{V}_j}\phi_j^i g_i^*$ and $\xi_j \triangleq \max_{t\in[T]}\|\nabla F(\mathbf{w}_t^j) - \nabla F_j(\mathbf{w}_t^j)\|$. For four different patterns, the residual error $\delta(\boldsymbol{x})$ partly depends on the data distribution among clusters ($\Gamma_j(\boldsymbol{x})$ and $\beta_j(\boldsymbol{x})$, $\forall s_j \in \mathcal{S}$). We can intuitively draw the following corollary.

Corollary 2 : *The greater the degree of data Non-IID among clusters, the larger the value of $\xi_j(\boldsymbol{x})$ for each cluster s_j, and the higher residual error $\delta(\boldsymbol{x})$. Given IID data among clusters, then $\nabla F(\mathbf{w}_t^j) = \nabla F_j(\mathbf{w}_t^j)$ and $\xi_j(\boldsymbol{x}) = 0$ for $\forall s_j \in \mathcal{S}$, the residual error $\delta(\boldsymbol{x})$ can be reduced.*

Corollary 2 shows that workers can be clustered to make the inter-cluster data distribution close to IID, so as to reduce the residual error $\delta(\boldsymbol{x})$ and improve training performance.

In addition, for DecSyn and DecAsy patterns, it is obvious that the residual error $\delta(\boldsymbol{x})$ is also related to $\mathbf{S}(\boldsymbol{x})$, which depends on the clusters' data sizes and the inter-cluster topology. However, it is difficult to intuitively figure out the influence of inter-cluster topology on training performance from the expression, which will be explored through experiment in Section 6.

Remark 1 : *The residual error $\delta(\boldsymbol{x})$ is also related to the learning rate η, and it approaches 0 as η decreases. This means that a diminishing learning rate can eliminate the residual error and achieve the optimal solution. However, a diminishing learning rate may also slow down the convergence rate, and it is difficult to choose an appropriate learning rate for each round a-priori. In this paper, we focus on the effect of clustering strategies in hierarchical federated learning on the convergence performance, and we assume a fixed learning rate for simplicity and clarity. In our future work, we will explore how to improve the convergence efficiency by selecting a suitable diminishing learning rate.*

5.2 Deploy Time-sharing Strategy for Intra-cluster Aggregation

For FL aggregations, workers usually follow a time-sharing (TS) [39][40][41] or frequency-sharing (FS) [42][43] strategy. However, with frequency-sharing strategies, the frequency allocated for the workers is static, resulting in a waste of bandwidth resources [41]. Therefore, we deploy the time-sharing strategy for the uplink communication of intra-cluster aggregation, *i.e.*, only one worker is uploading its model in any time period. Similar to the existing works, this paper does not consider the model distributing time caused by downlink communication, which is negligible compared to the model uploading time. The reason is that the downlink bandwidth is much larger than uplink bandwidth, and the aggregators can distribute models to workers by broadcasting.

Due to edge heterogeneity, workers in a cluster may have diverse computation capabilities and communication quality with the aggregator, which results in different model training and transmission time. Let L_i^c denote the model training time on worker v_i. $L_{i,j}^u$ denote the model uploading time between worker v_i and aggregator s_j. Suppose that the full knowledge of L_i^c and $L_{i,j}^u, \forall v_i \in \mathcal{V}_j$ can be obtained according to aggregator s_j's previous measurements [30]. Since workers' model training can be performed in parallel, different scheduling strategies for model uploading in uplink communication will result in different completion times of intra-cluster aggregation.

Without confusion, given R workers $v_1, v_2, ..., v_R$ in cluster $\mathcal{V}_j$. Let $v_{i_1}, v_{i_2}, ..., v_{i_R}$ denote the uploading sequence of the scheduling strategy, where sequence $i_1, i_2, ..., i_R$ is a rearrangement of sequence $1, 2, ..., R$. Let $L_{i_r}^p$ denote the completion time of each worker v_{i_r}'s model uploading. It is obvious that the start time of v_{i_r}'s uploading is equal to the larger of $L_{i_{r-1}}^p$ and $L_{i_r}^c$. Therefore, we derive the recursive formula as

$$L_{i_r}^p = \begin{cases} 0, & x = 0 \\ \max\{L_{i_{r-1}}^p, L_{i_r}^c\} + L_{i_r,j}^u, & 1 \le x \le X. \end{cases} \quad (50)$$

We can derive the following lemma from Eq. (50).

From the following Theorem, we can obtain the optimal time-sharing strategy, which minimizes the completion time of the intra-cluster aggregation.

Theorem 5 : *The completion time of the scheduling strategy will be minimized by ordering the uploading sequence as $v_{i_1}, v_{i_2}, ...v_{i_R}$, such that $L_{i_1}^c \le L_{i_2}^c \le ... \le L_{i_R}^c$.*

The above problem is essentially a single machine scheduling problem $1|r_j|C_{max}$, which is described in [44], and it is proven the optimal strategy is to schedule in order of the release times (*i.e.*, training times in this paper). Then the optimal completion time of the intra-cluster aggregation for

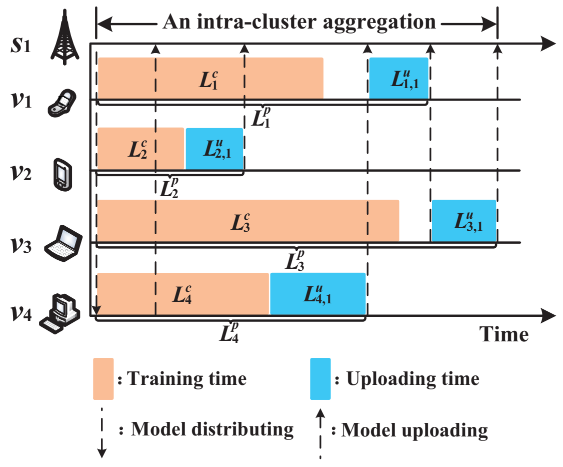


Fig. 3: Illustration of the intra-cluster time-sharing strategy.

cluster $\mathcal{V}_j$ is equal to the completion time of worker v_{i_R}'s model uploading, *i.e.*,

$$L_j^{opt} = L_{i_R}^p. \tag{51}$$

To better illustrate the impact of time-sharing scheduling strategy on the completion time of intra-cluster aggregation, we give an example with four workers v_1-v_4 clustered by aggregator s_1 in Fig. 3. As shown, the local training times of workers satisfy $L_2^c < L_4^c < L_1^c < L_3^c$, therefore the uploading sequence v_2, v_4, v_1, v_3 minimizes the completion time of the intra-cluster aggregation. For example, since the first worker to upload its local model is v_2, the completion time of its model uploading is $L_2^p = L_2^c + L_{2,1}^u$. Besides, since the completion time of the v_4's uploading is larger than that of local training on v_1, the completion time of v_1's model uploading is $L_1^p = \max\{L_4^p, L_1^c\} + L_{1,1}^u = L_4^p + L_{1,1}^u$. The optimal completion time of the intra-cluster aggregation for s_1 is L_3^p.

5.3 Problem Formulation

Based on both the discussions of convergence bounds in Section 5.1 and the proposed intra-cluster aggregation strategy in Section 5.2, we define the cluster construction problem for accelerating HA. Specifically, we determine a clustering strategy $\boldsymbol{x} = \{x_{i,j}\}_{v_i \in \mathcal{V}, s_j \in \mathcal{S}}$ to minimize the convergence bound $\mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T)$ in Eq. (49) with given intra-cluster aggregation completion time constraints.

On the one hand, minimizing the convergence bound $\mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T)$ is equivalent to improving the training performance by jointly considering several critical factors, *e.g.*, data distribution among clusters, aggregators' participation frequency (for asynchronous patterns), and inter-cluster topology (for decentralized patterns). Note that the values of some parameters depend on the data distribution among workers, such as μ, L and ξ_j, and we employ the method in [8] for estimation.

On the other hand, if workers are clustered into aggregators with long communication time, the benefit of minimizing the convergence bound on accelerating FL training may be reduced or even nullified. Therefore, it is necessary to limit the completion time of intra-cluster aggregation, *i.e.*,

$$L_j^{opt}(\boldsymbol{x}) \le L_j^{max} \tag{52}$$

Algorithm 2 Unified Clustering Algorithm for Hierarchical Federated Learning (FedUC)

```
Input: Data size $D_i$, $D_i^k$, $\forall v_i \in \mathcal{V}$, $\forall c_k \in \mathcal{C}$; completion time
    threshold $L_j^{max}$, $\forall s_j \in \mathcal{S}$
Output: Final clustering strategy $\boldsymbol{x}$
 1: for each $v_i \in \mathcal{V}$ do
 2:     for each $s_j \in \mathcal{S}$ do
 3:         $x_{i,j} = 0$
 4: Sort each worker $v_i \in \mathcal{V}$ in descending order by $D_i$ as $\mathcal{Q}$
 5: $\mathcal{F}_{temp} = +\infty$
 6: for each $v_i \in \mathcal{Q}$ do
 7:     for each $s_j \in \mathcal{S}$ do
 8:         $x_{i,j} = 1$
 9:         if $L_j^{opt}(\boldsymbol{x}) > L_j^{max}$ then
10:             $x_{i,j} = 0$
11:             continue
12:         if $\mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T) < \mathcal{F}_{temp}$ then
13:             $\mathcal{F}_{temp} = \mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T)$
14:             $j^* = j$
15:         $x_{i,j} = 0$
16:     $x_{i,j^*} = 1$
17: return final clustering strategy $\boldsymbol{x}$
```

for $\forall s_j \in \mathcal{S}$. L_j^{max} is the completion time threshold of cluster $\mathcal{V}_j$'s intra-cluster aggregation and $L_j^{opt}(\boldsymbol{x})$ is the optimal completion time under clustering strategy $\boldsymbol{x}$, which can be obtained by Eq. (51). Therefore, we formulate the cluster construction problem as follows:

$$\textbf{(P1)}: \min \mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T) \tag{53a}$$
$$\textbf{s.t.} \quad L_j^{opt}(\boldsymbol{x}) \le L_j^{max}, \qquad s_j \in \mathcal{S} \tag{53b}$$
$$\sum_{s_j \in \mathcal{S}} x_{i,j} = 1, \qquad \forall v_i \in \mathcal{V} \tag{53c}$$
$$x_{i,j} \in \{0, 1\}, \qquad \forall v_i \in \mathcal{V}, s_j \in \mathcal{S}. \tag{53d}$$

The first set of inequalities (53b) represents that the completion time of each intra-cluster aggregation does not exceed the threshold. The second set of equalities (53c) represents that each worker belongs to a unique cluster. Our target is to minimize the convergence bound, *i.e.*, $\min \mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T)$.

Remark 2 : *In the above description, we assumed that the workers are relatively stationary, so that a fixed clustering strategy can be made by FedUC. However, FedUC can also handle worker mobility by using a logger to track the historical locations of the moving workers [30]. In this way, the clustering strategy can be periodically updated by applying FedUC algorithm.*

5.4 Unified Clustering Algorithm

We introduce the unified clustering algorithm (FedUC) to solve **P1** for hierarchical federated learning, which is formally described in Alg. 2.

The main idea of our algorithm is to greedily determine which aggregator each worker belongs to one by one, so as to minimize the current objective function $\mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T)$. Specifically, we first initialize the states of all workers, *i.e.*, set $x_{i,j} = 0$, for all $v_i \in \mathcal{V}, s_j \in \mathcal{S}$, then sort the workers in the descending order of their data sizes as a queue $\mathcal{Q}$ (Lines 1-4). Note that the purpose of sorting the workers is to make the algorithm preferentially traverse the workers with more data, and in this way, the performance of the algorithm is usually better than traversing in random order. Then we decide which aggregator each worker in $\mathcal{Q}$ is clustered to in turn. For each worker v_i, we attempt to organize it to each aggregator $s_j \in \mathcal{S}$, *i.e.*, set $x_{i,j} = 1$ (Line 8), and calculate the values of $L_j^{opt}(\boldsymbol{x})$ and $\mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T)$. On the one hand, if the intra-cluster completion time of $\mathcal{V}_j$ exceeds the threshold L_j^{max}, worker v_i will not be organized to aggregator s_j, *i.e.*, resetting $x_{i,j} = 0$ (Lines 9-11). On the other hand, if the values of $L_j^{opt}(\boldsymbol{x})$ satisfy the constraints (53b), we traverse all aggregators $s_j \in \mathcal{S}$ and organize worker v_i to the aggregator s_{j^*} that minimizes the value of $\mathcal{F}(\boldsymbol{x}, \mathbf{w}_0, T)$ (Lines 7-16). The algorithm will terminate until all workers in $\mathcal{Q}$ are organized to the aggregators. Since we first sort the N workers according to the amount of data, and then traverse the clustering of each worker to M aggregators, the algorithm FedUC has polynomial time complexity $O(N \log N + NM)$.

6 Performance Evaluation

In this section, we first describe the system setup (Section 6.1). Then we explore the clustering results under different patterns (Section 6.2). Finally, we compare the training performance of our proposed FedUC with that of other clustering algorithms through extensive simulations (Section 6.3).

6.1 System Setup

6.1.1 Environment

To implement federated learning in a large-scale edge system, we conduct an experimental environment using PySyft (version 0.2.9) [45]. PySyft is a Python library for privacy-preserving deep learning under the PyTorch framework, which can create virtual workers for FL training. The virtual workers are deployed on an AMAX deep learning workstation with an 8-core Intel(R) Xeon(R) CPU (E5-2620v4) and 4 NVIDIA GeForce RTX 2080Ti GPUs with 11GB GDDR6, and each of them can be regarded as an individual machine and trains local model on its own dataset. Our experiments are performed on Ubuntu 18.04, CUDA v10.0, cuDNN v7.5.0.

6.1.2 Topology and Transmission Model

We consider a typical edge computing system [46] with a 40m×40m region divided by 4×4 grids as shown in Fig. 4, and an aggregator is deployed in the center of each grid, so totally M=16 aggregators s_1-s_{16} are deployed. Under the DecSyn pattern, each aggregator will send/receive cluster models to/from its neighbor aggregators, *i.e.*, aggregators closest to it in the up, down, left and right directions. For example, aggregator s_6 can exchange its cluster model with aggregators s_2, s_5, s_7 and s_{10}. A centralized PS is deployed at the junction of grids 1, 2, 5 and 6, such that the required time for transferring models between the PS and aggregators varies. In this way, we simulate the CenAsy pattern where aggregators participate in the global aggregation with different frequency. 100 workers v_1-v_{100} with data of different sizes and distributions are randomly deployed across the region, and each of them will be clustered to a unique aggregator.

We simulate the model distributing time $L_{i,j}^d$, training time L_i^c and uploading time $L_{i,j}^u$ for each worker v_i and aggregator s_j. The model training time can be calculated by $L_i^c = \zeta_i D_i$, where ζ_i is the training time of one sample on worker v_i. $h_{i,j}$ denotes the channel gain from worker v_i to aggregator s_j, and p_i denotes the transmission power of worker v_i. As a result, the transmission rate from v_i to s_j can be given by the Shannon capacity [49], $r_{i,j}^u = W_j \log_2(1 + \frac{p_i^u h_{i,j}}{\sigma^2})$, where W_j is channel bandwidth on aggregator s_j for model delivering, p_i^u is the transmission power of each worker v_i and σ^2 is the noise power. The channel gain for calculating the wireless transmission is modeled by the pass-loss $h_{i,j} = h_0 d_{i,j}^{-4}$ model [50], where h_0 is the path-loss constant and $d_{i,j}$ is the distance between worker v_i and aggregator s_j. Let ξ denote the model size. Thus the model uploading time from worker v_i to aggregator s_j can be calculated by $L_{i,j}^u = \frac{\xi}{r_{i,j}^u}$. Similarly, we can obtain the model distributing time $L_{i,j}^d$. The settings of some important parameters are listed in Table 5.

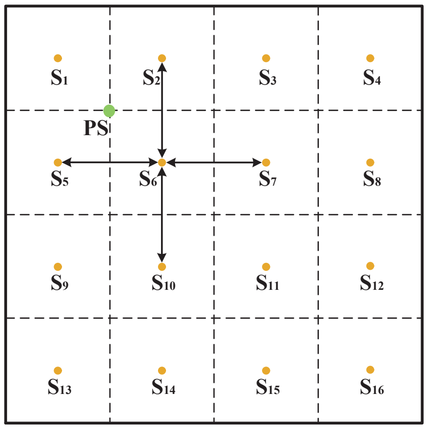


Fig. 4: Grid Network.

TABLE 5: Settings of Some System Parameters.

Parameters	Values
Channel bandwidth W_j	10 MHz [47]
Aggregators' transmission power p_j^d	33 dBm
Workers' transmission power p_i^u	50-100 mWatts [48]
Noise power σ^2	-100 dBm [48]
path-loss constant h_0	-40 dB [49]

6.1.3 Models and Datasets

The experiments are conducted with two classical models (*i.e.*, LR [51] and CNN [52]) and on two datasets (MNIST [53] and CIFAR-10 [54]). MNIST consists of 60,000 handwritten digits for training and 10,000 for testing, while CIFAR-10 includes 50,000 images for training and 10,000 for testing, and both of them have ten different types of objects.

Similar to the existing works [30][55][56], we implement the Non-IID data among workers by label skewed partition. Specifically, the data in MNIST (or CIFAR-10) labeled as '0' are distributed to workers v_1-v_{10}, the data labeled as '1' are distributed to workers v_{11}-v_{20},..., and the data labeled as '9' are distributed to workers v_{91}-v_{100}.

The LR network architecture consists of three fully-connected layers with 784, 512 and 512 units respectively, and a softmax layer with 10 units. The CNN network architecture consists of two 5×5 convolution layers (20, 50 channels for MNIST and 32, 64 channels for CIFAR-10), each of which is followed by 2×2 max pooling, two fully-connected layers (800, 500 units for MNIST and 1600, 512 units for CIFAR-10), and a softmax layer with 10 units. The model sizes are 2.56 MB (LR on MNIST), 1.64MB (CNN on MNIST) and 3.35MB (CNN on CIFAR-10), respectively. We adopt the same mini-batch size 64 for all workers. The learning rate is set as $\eta = 0.01$ for MNIST, and $\eta = 0.05$ for CIFAR-10.

6.1.4 Benchmarks and Performance Metrics

We compare our clustering algorithm FedUC with two typical clustering benchmarks.

- Communication-aware clustering (ComAw) [24]: Each worker is clustered to the aggregator with the shortest communication time.
- Data-aware clustering (DatAw) [21]: Workers are clustered based on the statistical properties of data distribution among them.

ComAw focuses solely on minimizing communication time, while DatAw aims to optimize the clustering based on the characteristics of the data distribution. In contrast, our FedUC algorithm takes into account additional factors, such as the frequency of clusters participating in inter-cluster aggregation and the inter-cluster topology. By comparing FedUC with these baseline schemes, we can effectively evaluate the performance and effectiveness of our proposed algorithm in considering a broader range of factors beyond communication and data distribution in the clustering process.

We implement FedUC, ComAw and DatAw under different inter-cluster aggregation patterns, *i.e.*, CenSyn, CenAsy, DecSyn and DecAsy. We set $T' = 5$, *i.e.*, each aggregator performs 5 intra-cluster aggregations before each inter-cluster aggregation.

To evaluate the training performance, we adopt three performance metrics. 1) *Loss Function* reflects the training process of the model and whether convergence has been achieved or not. 2) *Accuracy* is the most common performance metric in classification problems, which is defined as the proportion of the right data classified by the model to all test data. 3) *Training Time* is adopted to evaluate the training speed.

6.2 Clustering Results

In this section, we explore some values of key indexes (*e.g.*, the degree of Non-IID, intra-cluster aggregation time) after the execution of different clustering algorithms. In addition, we explore the impacts of some parameters (*e.g.*, aggregators' participation frequency under CenAsy, the number of communication links to aggregators under DecSyn) on clustering for FedUC. The exploration of these factors is beneficial for understanding the evaluation results in Section 6.3.

6.2.1 The Degree of Non-IID

We apply the earth mover distance (EMD) [12] to represent the difference of data distribution between cluster $\mathcal{V}_j$'s dataset and the global dataset, *i.e.*, $P_j = \text{EMD}(\mathcal{D}, \mathcal{D}_j)$. Fig. 5 shows the average of EMD $\overline{P} = \sum_{s_j \in \mathcal{S}} P_j$ for FedUC, ComAw and DatAw under different inter-cluster aggregation patterns. $\overline{P}$ can be regarded as a index to evaluate the degree of data Non-IID among clusters. As is shown, the values of $\overline{P}$ for ComAw and DatAw remain 0.394 and 0.182 unchanged under different patterns, since the clustering strategies of ComAw and DatAw are consistent under different patterns. The values of $\overline{P}$ for FedUC are 0.191, 0.242, 0.223 and 0.256 under CenSyn, CenAsy DecSyn and DecAsy, respectively, which are slightly larger than those in DatAw. This shows that by deploying FedUC clustering algorithm, the degree of data Non-IID among clusters can close to that by deploying DatAw.

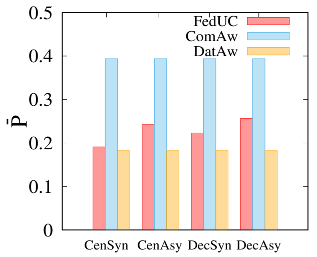


Fig. 5: Average EMD under Different Pattern.

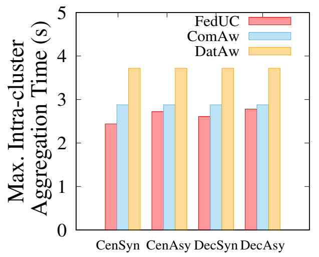


Fig. 6: Maximum Intra-cluster Aggregation Time under Different Pattern.

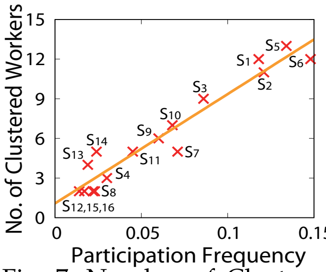


Fig. 7: Number of Clustered Workers vs. Participation Frequency (CenAsy).

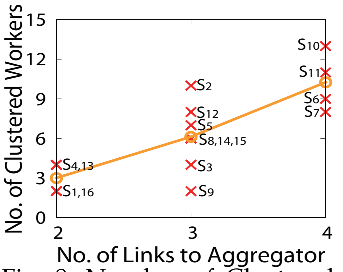


Fig. 8: Number of Clustered Workers vs. Number of Links to Aggregator (DecSyn).

6.2.2 Intra-cluster Aggregation Time

Fig. 6 shows the maximum intra-cluster aggregation time of all clusters. As is shown, the maximum intra-cluster aggregation time for ComAw and DatAw remain 2.88s and 3.72s unchanged under different patterns. The maximum intra-cluster aggregation time for FedUC are 2.44s, 2.72s, 2.61s and 2.78 under CenSyn, CenAsy DecSyn and DecAsy, respectively. The reason FedUC is superior to ComAw in terms of the intra-cluster aggregation time is that FedUC restricts the intra-cluster aggregation time as shown in Eq. (52). Whereas, in ComAw, workers are clustered to the aggregators with the shortest communication time, which may cause some aggregators to communicate with excessive workers and increase the intra-cluster aggregation time.

6.2.3 The Impact of Participation Frequency under CenAsy

Fig. 7 shows the number of workers clustered to aggregators s_1-s_{16} after the FedUC clustering algorithm is executed under CenAsy pattern. As shown by the general trend of the fitting curve, the number of clustered workers increases with the aggregators' participation frequency. For example, there are 12 workers clustered to aggregator s_6 with participation frequency 0.148, while only 2 workers clustered to aggregator s_{12} with participation frequency 0.014. This indicates that workers tend to be clustered to aggregators with larger participation frequency under CenAsy pattern.

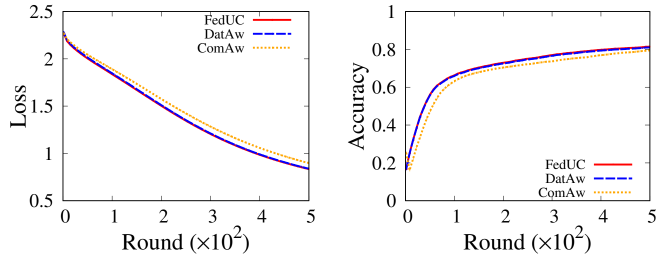


Fig. 9: Loss/Accuracy vs. Round (LR on MNIST, CenSyn). *Left*: Loss; *Right*: Accuracy.

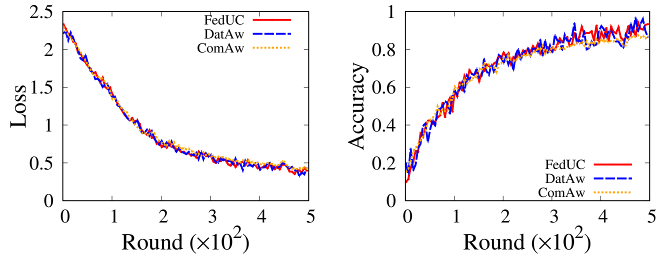


Fig. 10: Loss/Accuracy vs. Round (CNN on MNIST, CenSyn). *Left*: Loss; *Right*: Accuracy.

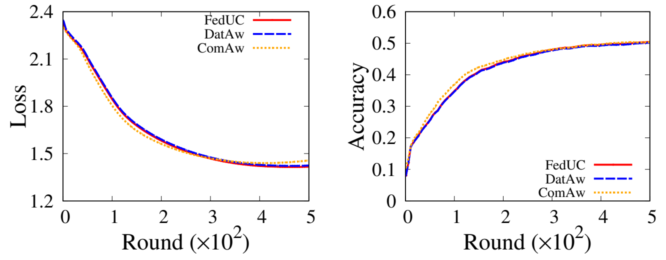


Fig. 11: Loss/Accuracy vs. Round (CNN on CIFAR-10, CenSyn). *Left*: Loss; *Right*: Accuracy.

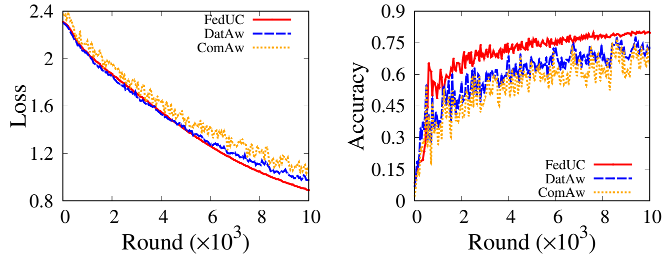


Fig. 12: Loss/Accuracy vs. Round (LR on MNIST, CenAsy). *Left*: Loss; *Right*: Accuracy.

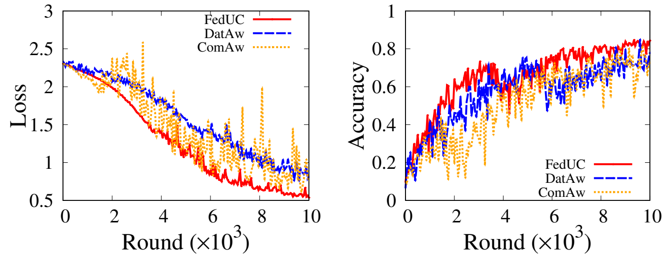


Fig. 13: Loss/Accuracy vs. Round (CNN on MNIST, CenAsy). *Left*: Loss; *Right*: Accuracy.

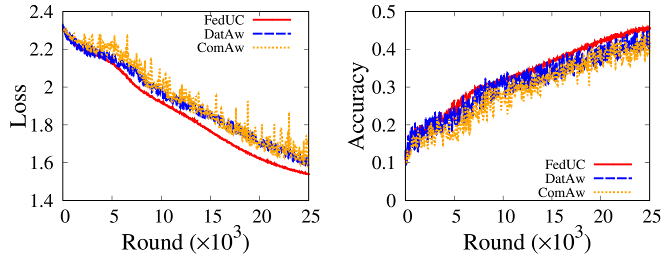


Fig. 14: Loss/Accuracy vs. Round (CNN on CIFAR-10, CenAsy). *Left*: Loss; *Right*: Accuracy.

6.2.4 *The Impact of Topology under DecSyn*

Fig. 8 shows the number of workers clustered to aggregators s_1-s_{16} after the FedUC clustering algorithm is executed under DecSyn pattern. As shown in this figure, the average numbers of workers clustered to aggregators with links numbers of 2, 3, and 4 are 3, 6.125 and 10.25, respectively, which shows that the expected number of workers clustered to an aggregator is positively correlated with the number of links. For example, there are 13 workers clustered to aggregator s_{10} with 4 communication links, while only 2 workers clustered to aggregator s_1 with 2 communication links. This indicates that workers tend to be clustered to aggregators with dense communication links under DecSyn pattern.

6.3 Performance Evaluation

6.3.1 *Convergence Performance*

For different inter-cluster aggregation patterns, *e.g.*, CenSyn, CenAsy, DecSyn and DecAsy, our clustering algorithm FedUC can always achieve better convergence performance than ComAw and DatAw, but the degree of performance improvement varies with different patterns.

Figs. 9-11 show that, under the CenSyn pattern, the convergence performance of FedUC is slightly better than that of ComAw, while is similar to that of DatAw. This is because ComAw clusters workers without considering the data distribution, and its convergence performance is worse than that of both FedUC and DatAw. For example, in Fig. 9, the convergence accuracy of FedUC, DatAw and ComAw is 81.4%, 81.0%, 79.4% after 500 rounds, respectively. The number of training rounds to achieve 75% accuracy is 254, 259 and 337, respectively. FedUC can reduce the number of required training rounds by about 24.6% compared with ComAw.

Figs. 12-14 show that during the training process under CenAsy, the loss and accuracy curves jitter more violently compared with that under CenSyn due to the asynchronism of global aggregation, but the jitter degree varies with different clustering algorithms. Since ComAw clusters workers without considering the data distribution, it is easy to cause the inter-cluster data highly Non-IID (large $\overline{P}$). Unfortunately, it has been pointed out that asynchronous aggregation aggravates the negative effect of Non-IID issue on convergence performance [30]. Although DatAw clusters workers based on the data distribution among workers, it does not consider the participation frequency of aggregators in global updating, which may cause a large number of workers to be clustered to low-frequency aggregators, reducing convergence performance. In FedUC, not only the data distribution is considered, which makes inter-cluster data distribution as close to IID as possible, but also the participation frequency of aggregators is taken account. Therefore, the jitter of FedUC is greatly reduced and the convergence performance can be significantly improved compared with that of ComAw and DatAw. For example, in Fig. 12, after 10000 rounds of training, the training

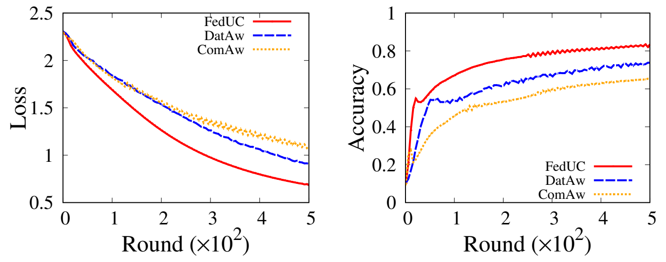


Fig. 15: Loss/Accuracy vs. Round (LR on MNIST, DecSyn). *Left*: Loss; *Right*: Accuracy.

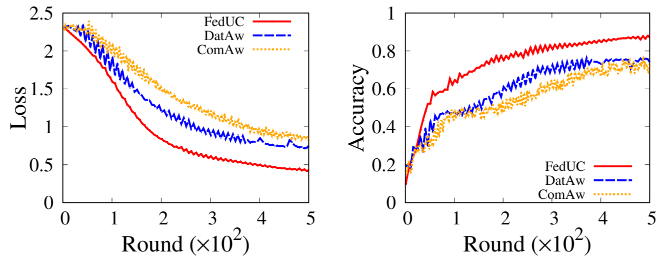


Fig. 16: Loss/Accuracy vs. Round (CNN on MNIST, DecSyn). *Left*: Loss; *Right*: Accuracy.

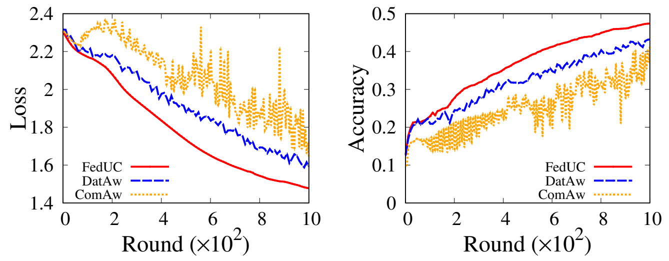


Fig. 17: Loss/Accuracy vs. Round (CNN on CIFAR-10, DecSyn). *Left*: Loss; *Right*: Accuracy.

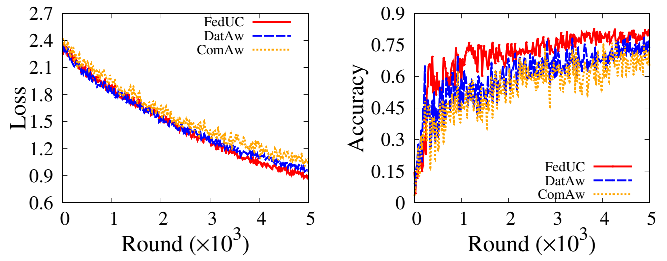


Fig. 18: Loss/Accuracy vs. Round (LR on MNIST, DecAsy). *Left*: Loss; *Right*: Accuracy.

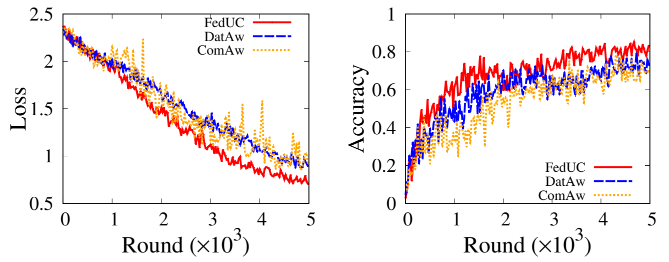


Fig. 19: Loss/Accuracy vs. Round (CNN on MNIST, DecAsy). *Left*: Loss; *Right*: Accuracy.

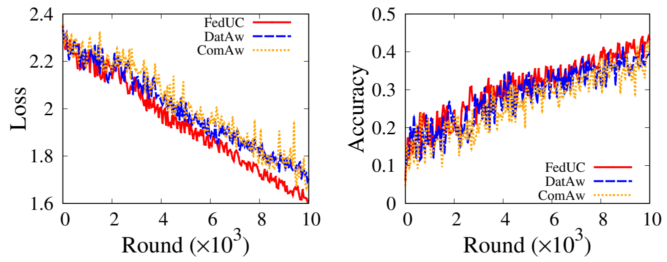


Fig. 20: Loss/Accuracy vs. Round (CNN on CIFAR-10, DecAsy). *Left*: Loss; *Right*: Accuracy.

accuracy of ComAw varies between 63% and 73%, that of DatAw varies between 66% and 77%, while that of FedUC varies between 79% and 80%. Besides, the number of training rounds to achieve stable 60% accuracy is 9201, 5702 and 1913 for ComAw, DatAw and FedUC, respectively. FedUC reduces the number of training rounds by about 79.2% and 66.5% compared with ComAw and DatAw, respectively.

Figs. 15-17 show that, under DecSyn, the convergence performance of FedUC is much better than that of ComAw and DatAw. Similar to the case under CenAsy, the clustering method of DatAw does not consider the inter-cluster topology, and a large number of workers may be clustered to aggregators with sparse links, reducing convergence performance. For example, in Fig. 15, the training accuracy of FedUC, DatAw and ComAw is 81.8%, 73.7%, 65.2% after 500 rounds, respectively. The number of training rounds to achieve 65% accuracy is 85, 256 and 491, respectively. FedUC can reduce the number of training rounds by about 82.7% and 66.8% compared to ComAw and DatAw, respectively.

Figures 18-20 demonstrate the comparison of FedUC with ComAw and DatAw under the DecAsy pattern. Similar to the observations made under the CenAsy and DecSyn patterns, DatAw clusters workers without considering the inter-cluster topology and the participation frequency of aggregators in global updating, which may lead to a suboptimal convergence performance. In contrast, FedUC considers a broader range of factors, including the above two factors, beyond communication and data distribution in the clustering process. Consequently, under the DecAsy pattern, FedUC showcases superior convergence performance compared to ComAw and DatAw. For example, in Figure 18, after 5000 rounds of training, the training accuracy of ComAw varies between 61% and 75%, that of DatAw varies between 70% and 78%, while that of FedUC varies between 79% and 82%. Besides, the number of training rounds to achieve stable 60% accuracy is 4487, 3301 and 884 for ComAw, DatAw and FedUC, respectively. FedUC reduces the number of training rounds by about 80.3% and 73.2% compared with ComAw and DatAw, respectively.

6.3.2 *Impact of the Number of Clusters*

Figs. 21-24 show the impact of different number M of clusters on the convergence performance of CenSyn, CenAsy, DecSyn and DecAsy.

Fig. 21 shows that, under CenSyn, the number of training rounds for the target accuracy has little correlation with the number of clusters. For example, when the number M of clusters in FedUC is set as 9, 16 and 25, the number of training rounds to achieve stable 80% accuracy is 310, 307 and 302, respectively, while that in ComAw is 318, 320 and 355, respectively.

Fig. 22 shows that the convergence performance decreases with the increasing number of clusters for CenAsy. For example, given the number of clusters of 9, 16, 25 in FedUC, the number of training rounds required to achieve stable 60%

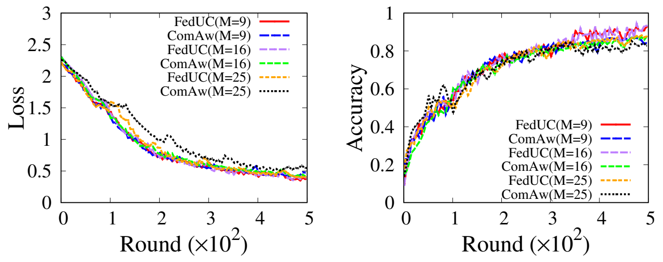


Fig. 21: Loss/Accuracy vs. Round (CNN on MNIST, CenSyn). *Left*: Loss; *Right*: Accuracy.

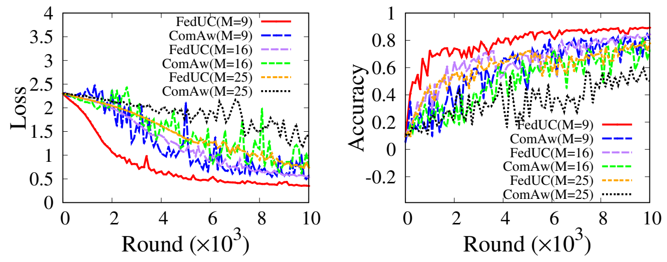


Fig. 22: Loss/Accuracy vs. Round (CNN on MNIST, CenAsy). *Left*: Loss; *Right*: Accuracy.

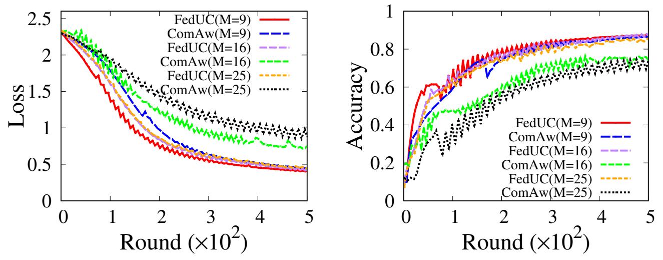


Fig. 23: Loss/Accuracy vs. Round (CNN on MNIST, DecSyn). *Left*: Loss; *Right*: Accuracy.

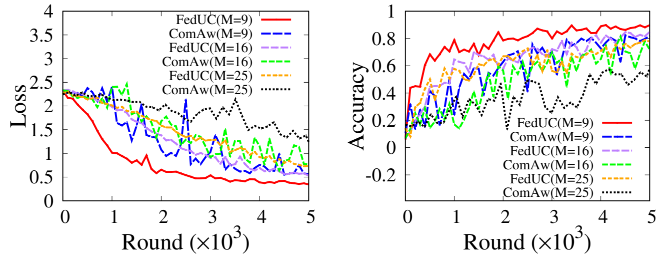


Fig. 24: Loss/Accuracy vs. Round (CNN on MNIST, DecAsy). *Left*: Loss; *Right*: Accuracy.

accuracy is 713, 2385 and 3039, respectively. While in ComAw it separately needs 5227, 8285 and 9954 training rounds to achieve stable accuracy of 50%. Meanwhile, the convergence performance can be greatly improved for different number of clusters by deploying FedUC. For example, given 9 clusters, it requires 713 and 5227 training rounds to achieve stable 50% accuracy for FedUC and ComAw, while given 25 clusters requires 3039 and 9954 rounds. By deploying FedUC under CenAsy, the number of required training rounds can be reduced by 69.5%-87.1% compared with deploying ComAw.

Fig. 23 shows that, under DecSyn, the convergence performance decreases with the increasing of the number of clusters in ComAw. For example, when the number of clusters in ComAw is set as 9, 16 and 25, the number of training rounds to achieve 70% accuracy is 151, 349 and 498, respectively. Nevertheless, the impact of the increasing number of clusters has little effect on the convergence performance for FedUC. For example, when the number of clusters in FedUC is set as 9, 16 and 25, the number of training rounds required to achieve 80% accuracy is 261, 285 and 302, respectively. This indicates that the convergence performance of FedUC does not decrease obviously with more clusters in the system, which enhances the scalability for DecSyn. When the number of clusters is small, whether to deploy FedUC or not has little impact on the convergence performance, but when the number of clusters increases, the performance improvement of deploying FedUC gets more and more significant. For example, given 9 clusters, it requires 146 and 151 rounds of training to achieve stable 70% accuracy for FedUC and ComAw, respectively, while the numbers of training rounds reach 158 and 495 when M=25.

Fig. 24 shows that the convergence performance decreases with the increasing number of clusters for CenAsy. For example, given the number of clusters of 9, 16, 25 in FedUC, the number of training rounds required to achieve stable 60% accuracy is 563, 2189 and 3424, respectively. While in ComAw it separately needs 1723, 3464 and 4439 training rounds to achieve stable accuracy of 50%. Meanwhile, the convergence performance can be greatly improved for different number of clusters by deploying FedUC. For example, given 9 clusters, it requires 433 and 563 training rounds to achieve stable 50% accuracy for FedUC and ComAw, while given 25 clusters requires 1753 and 3424 rounds. By deploying FedUC under CenAsy, the number of required training rounds can be reduced by 23.1%-48.9% compared with deploying ComAw.

TABLE 6: Training time required to achieve target accuracy.

Accuracy		ComAw	DatAw	FedUC
CenSyn	50%	198s	153s	107s
	60%	295s	222s	158s
	70%	659s	537s	367s
	80%	1841s	1533s	1026s
CenAsy	50%	521s	276s	86s
	60%	1170s	626s	140s
	70%	2367s	1632s	340s
	80%	5388s	4133s	916s
DecSyn	50%	503s	163s	42s
	60%	1165s	620s	153s
	70%	3063s	1396s	327s
	80%	7010s	3621s	948s
DecAsy	50%	470s	184s	37s
	60%	812s	472s	103s
	70%	1553s	1126s	273s
	80%	4274s	2832s	804s

6.3.3 *Comparison of Training Time*

In this section, we evaluate the training time of ComAw, DatAw and FedUC over LR on MNIST. As shown in Table 6, FedUC greatly accelerates HA compared with the other two methods under the patterns CenSyn, CenAsy, DecSyn and DecAsy. For example, under CenSyn, the time required to reach 80% accuracy is 1841s, 1533s and 1026s for ComAw, DatAw and FedUC, respectively. FedUC can accelerate HA by about 1.79× and 1.49× compared with ComAw and DatAw. Under CenAsy, the time required to reach 80% accuracy is 5388s, 4133s and 916s for ComAw, DatAw and FedUC, respectively. FedUC can accelerate HA by about 5.88× and 4.51× compared with ComAw and DatAw, respectively. Under DecSyn, the time required to reach 80% accuracy is 7010s, 3621s and 948s for ComAw, DatAw and FedUC, respectively. FedUC can accelerate HA by about 7.39× and 3.82× compared with ComAw and DatAw. Under DecAsy, the time required to reach 80% accuracy is 4274s, 2832s and 804s for ComAw, DatAw and FedUC, respectively. FedUC can accelerate HA by about 5.32× and 3.52× compared with ComAw and DatAw. The reason FedUC can greatly accelerate model training is that FedUC not only has better convergence performance (as shown in Fig. 9-24), but also has shorter intra-cluster aggregation time (as shown in Fig. 6) compared with ComAw and DatAw under different patterns.

7 Conclusion

In this paper, we have explored the cluster construction problem under different patterns for accelerating hierarchical federated learning. We have theoretically analyzed the quantitative relationship between the convergence bounds of four different inter-cluster aggregation patterns (CenSyn, CenAsy, DecSyn and DecAsy) and several factors, *e.g.*, data distribution among clusters, frequency of clusters participating in inter-cluster aggregation, and inter-cluster topology. Based on the convergence analysis, we have designed a unified clustering algorithm FedUC to solve the cluster construction problem given the time constraints for intra-cluster aggregation. The experimental results indicate that the model training of CenSyn, CenAsy, DecSyn and DecAsy can be significantly accelerated by deploying our FedUC algorithm.

References

[1] A. Pantelopoulos and N. G. Bourbakis, "A survey on wearable sensor-based systems for health monitoring and prognosis," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 1, pp. 1–12, 2009.

[2] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones." in *Esann*, 2013.

[3] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Towards an intelligent edge: Wireless communication meets machine learning," *arXiv preprint arXiv:1809.00343*, 2018.

[4] H. B. McMahan and D. Ramage., "http://www.googblogs.com/federated-learning-collaborative-machine-learning-without-centralized-training-data/," Google, 2017.

[5] X. Wei, Q. Li, Y. Liu, H. Yu, T. Chen, and Q. Yang, "Multi-agent visualization for explaining federated learning," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 2019, pp. 6572–6574.

[6] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–7.

[7] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine Learning and Systems*, vol. 2, pp. 429–450, 2020.

[8] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[9] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[10] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, pp. 6757–6779, 2017.

[11] B. McMahan, E. Moore, D. Ramage, S. Hampson, and A. y. B. Arcas, "Communication-efficient learning of deep networks from decentralized data," *AISTATS*, pp. 1273–1282, 2017.

[12] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[13] F. Sattler, S. Wiedemann, K.-R. Muller, and W. Samek, "Robust and communication-efficient federated learning from non-iid data," *IEEE transactions on neural networks and learning systems*, pp. 1–14, 2019.

[14] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, "Scaling distributed machine learning with the parameter server," in *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, 2014, pp. 583–598.

[15] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[16] J. Wang and G. Joshi, "Adaptive communication strategies to achieve the best error-runtime trade-off in local-update sgd," *arXiv: Learning*, 2018.

[17] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," *arXiv preprint arXiv:1705.09056*, 2017.

[18] J. Wang, A. K. Sahu, Z. Yang, G. Joshi, and S. Kar, "MATCHA: Speeding Up Decentralized SGD via Matching Decomposition Sampling," in *2019 Sixth Indian Control Conference (ICC)*, 2019, pp. 299–300.

[19] H. Xu, M. Chen, Z. Meng, Y. Xu, L. Wang, and C. Qiao, "Decentralized machine learning through experience-driven method in edge networks," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 2, pp. 515–531, 2022.

[20] J.-w. Lee, J. Oh, S. Lim, S.-Y. Yun, and J.-G. Lee, "Tornadoaggregate: Accurate and scalable federated learning via the ring-based architecture," *arXiv preprint arXiv:2012.03214*, 2020.

[21] N. Mhaisen, A. Awad, A. Mohamed, A. Erbad, and M. Guizani, "Optimal user-edge assignment in hierarchical federated learning based on statistical properties and network topology constraints," *IEEE Transactions on Network Science and Engineering*, pp. 1–1, 2021.

[22] M. S. H. Abad, E. Ozfatura, D. Gunduz, and O. Ercetin, "Hierarchical federated learning across heterogeneous cellular networks," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 8866–8870.

[23] Z. Wang, H. Xu, J. Liu, H. Huang, C. Qiao, and Y. Zhao, "Resource-efficient federated learning with hierarchical aggregation in edge computing," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[24] Z. Chai, A. Ali, S. Zawad, S. Truex, A. Anwar, N. Baracaldo, Y. Zhou, H. Ludwig, F. Yan, and Y. Cheng, "TiFL: A Tier-based Federated Learning System," in *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing*, 2020, pp. 125–136.

[25] Z. Wang, H. Xu, J. Liu, Y. Xu, H. Huang, and Y. Zhao, "Accelerating federated learning with cluster construction and hierarchical aggregation," *IEEE Transactions on Mobile Computing*, 2022.

[26] Y. Sun, J. Shao, Y. Mao, J. H. Wang, and J. Zhang, "Semi-decentralized federated edge learning for fast convergence on non-iid data," in *2022 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2022, pp. 1898–1903.

[27] Z. Zhong, Y. Zhou, D. Wu, X. Chen, M. Chen, C. Li, and Q. Z. Sheng, "P-FedAvg: Parallelizing Federated Learning with Theoretical Guarantees," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[28] Y. Sun, J. Shao, Y. Mao, and J. Zhang, "Asynchronous semi-decentralized federated edge learning for heterogeneous clients," in *ICC 2022-IEEE International Conference on Communications*. IEEE, 2022, pp. 5196–5201.

[29] A. Bellet, A.-M. Kermarrec, and E. Lavoie, "D-cliques: Compensating noniidness in decentralized federated learning with topology," *arXiv preprint arXiv:2104.07365*, 2021.

[30] Q. Ma, Y. Xu, H. Xu, Z. Jiang, L. Huang, and H. Huang, "FedSA: A Semi-Asynchronous Federated Learning Mechanism in Heterogeneous Edge Computing," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3654–3672, 2021.

[31] S. Zheng, Q. Meng, T. Wang, W. Chen, N. Yu, Z.-M. Ma, and T.-Y. Liu, "Asynchronous stochastic gradient descent with delay compen-

sation," in *International Conference on Machine Learning*. PMLR, 2017, pp. 4120–4129.

[32] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv preprint arXiv:1903.03934*, 2019.

[33] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," *arXiv preprint arXiv:1907.02189*, 2019.

[34] J. Wang, S. Wang, R.-R. Chen, and M. Ji, "Local averaging helps: Hierarchical federated learning and convergence analysis," *arXiv preprint arXiv:2010.12998*, 2020.

[35] L. Nguyen, P. H. Nguyen, M. Dijk, P. Richtárik, K. Scheinberg, and M. Takác, "Sgd and hogwild! convergence without the bounded gradients assumption," in *International Conference on Machine Learning*. PMLR, 2018, pp. 3750–3758.

[36] X. Cao, G. Zhu, J. Xu, and S. Cui, "Transmission power control for over-the-air federated averaging at network edge," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 5, pp. 1571–1586, 2022.

[37] H. R. Feyzmahdavian, A. Aytekin, and M. Johansson, "A delayed proximal gradient method with linear convergence rate," in *2014 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2014, pp. 1–6.

[38] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-iid data," in *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020, pp. 15–24.

[39] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 1387–1395.

[40] X. Mo and J. Xu, "Energy-efficient federated edge learning with joint communication and computation design," *Journal of Communications and Information Networks*, vol. 6, no. 2, pp. 110–124, 2021.

[41] B. Luo, X. Li, S. Wang, J. Huang, and L. Tassiulas, "Cost-effective federated learning in mobile edge networks," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3606–3621, 2021.

[42] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, pp. 2457–2471, 2021.

[43] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 3, pp. 1935–1949, 2021.

[44] J. Blazewicz, K. Ecker, E. Pesch, G. Schmidt, and J. Weglarz, *Handbook on scheduling*. Springer, 2019.

[45] A. Ziller, A. Trask, A. Lopardo, B. Szymkow, B. Wagner, E. Bluemke, J.-M. Nounahon, J. Passerat-Palmbach, K. Prakash, N. Rose *et al.*, "Pysyft: A library for easy federated learning," in *Federated Learning Systems*. Springer, 2021, pp. 111–139.

[46] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Joint service placement and request routing in multi-cell mobile edge computing networks," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 10–18.

[47] D. Jiang and L. Delgrossi, "Ieee 802.11 p: Towards an international standard for wireless access in vehicular environments," in *VTC Spring 2008-IEEE Vehicular Technology Conference*. IEEE, 2008, pp. 2036–2040.

[48] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974–983, 2014.

[49] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, 2016.

[50] T. S. Rappaport *et al.*, *Wireless communications: principles and practice*. prentice hall PTR New Jersey, 1996, vol. 2.

[51] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*. John Wiley & Sons, 2013, vol. 398.

[52] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[53] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

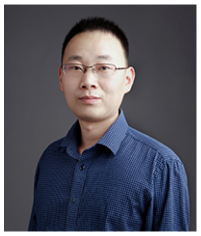
[54] A. Krizhevsky, G. Hinton *et al.*, *Learning multiple layers of features from tiny images*. Citeseer, 2009.

[55] J. Liu, H. Xu, L. Wang, Y. Xu, C. Qian, J. Huang, and H. Huang, "Adaptive asynchronous federated learning in resource-constrained edge computing," *IEEE Transactions on Mobile Computing*, 2021.

[56] K. Hsieh, A. Phanishayee, O. Mutlu, and P. Gibbons, "The non-iid data quagmire of decentralized machine learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 4387–4398.

Qianpiao Ma received his Ph.D. degree in computer software and theory from the University of Science and Technology of China in 2022. Prior to that, he received his B.S. degree in computer science from the University of Science and Technology of China in 2014. He is currently a postdoctoral researcher at Purple Mountain Laboratories. His primary research interests include federated learning, mobile edge computing and distributed machine learning.

Yang Xu (Member, IEEE) is currently an associate researcher in the School of Computer Science and Technology, University of Science and Technology of China. He got his Ph.D. degree in computer science and technology from University of Science and Technology of China in 2019. He got his B.S. degree in Wuhan University of Technology in 2014. His primary research interests include ubiquitous computing, deep learning and mobile edge computing.



Hongli Xu (Member, IEEE) received his Ph.D. degree in computer software and theory from the University of Science and Technology of China, China, in 2007. He is a professor at the School of Computer Science and Technology, University of Science and Technology of China. He has won the best paper awards in several famous conferences. He has published more than 100 papers in famous journals and conferences. His primary research interest is software defined networks, edge computing and Internet of Thing.

jianchun Liu received his Ph.D. degree in computer software and theory from the University of Science and Technology of China in 2022. He is currently an associate researcher in the School of Computer Science and Technology, University of Science and Technology of China. His primary research interests include federated learning, mobile edge computing and distributed machine learning.

Liusheng Huang (Member, IEEE) received his M.S. degree in computer science from the University of Science and Technology of China in 1988. He is currently a Senior Professor and a Ph.D. Supervisor with the School of Computer Science and Technology, University of Science and Technology of China. He has authored or co-authored six books and over 300 journal/conference papers. His research interests are in the areas of the Internet of Things, vehicular Ad-Hoc networks, information security, and distributed computing.