通过间接服务器-客户端通信实现的联邦学习

Jieming Bian 电子与计算机工程系 迈阿密大学 科勒尔盖布尔斯，FL 33146
jxb1974@miami.edu

Cong Shen 电子与计算机工程系 弗吉尼亚大学 夏洛茨维尔，VA 22904
cong@virginia.edu

Jie Xu 电子与计算机工程系 迈阿密大学 科勒尔盖布尔斯，FL 33146
jiexu@miami.edu

arXiv:2302.07323v1 [cs.DC] 14 Feb 2023

摘要—联邦学习（FL）是一种通信高效且具有隐私保护能力的分布式机器学习框架，近年来受到广泛关注。尽管联邦学习算法形式多样（例如同步联邦学习、异步联邦学习），底层优化方法也各不相同，但几乎所有现有研究都默认存在通信基础设施，以支持服务器与客户端之间直接通信并进行模型数据交换。然而，在许多能够从分布式学习中受益、却缺乏适当通信基础设施的现实应用中（例如偏远地区的智能感知），这一假设并不成立。本文提出一种名为 FedEx（通过模型快递传输实现的联邦学习的简称）的新型联邦学习框架，利用移动运输工具（例如无人机）在服务器与客户端之间建立间接通信信道。根据运输器采用同步调度还是异步调度，我们开发了两种算法，分别称为 FedEx-Sync 和 FedEx-Async。尽管间接通信会在全局模型分发和本地模型收集过程中，为客户端引入异构延迟，我们证明了 FedEx 两个版本的收敛性。随后，收敛分析为如何将客户端分配给不同运输器以及如何设计客户端之间的路线提供了启示。我们通过仿真网络中的实验，在两个公开数据集上评估了 FedEx 的性能。

一、引言

近年来，联邦学习（FL）已成为一种广受欢迎的分布式机器学习框架。在该框架中，多个分布式客户端可在参数服务器的协调下协同训练同一个机器学习模型，而无需向其他方暴露自己的数据。随着智能手机和物联网（IoT）设备等边缘设备生成的数据量空前增长，以及人们对上传数据所带来的隐私问题日益关注，

由于其分布式特性和隐私保护优势，联邦学习如今被广泛视为下一代机器学习范式，可为医疗保健、农业、交通运输、工业物联网和移动应用等众多领域提供支持。

联邦学习区别于其他分布式学习框架的一个主要特点，是特别关注客户端与服务器之间的通信效率。具体而言，本地随机梯度下降（SGD）算法[1] 和 FedAvg 算法 [2]允许客户端先在各自的数据集上执行多次

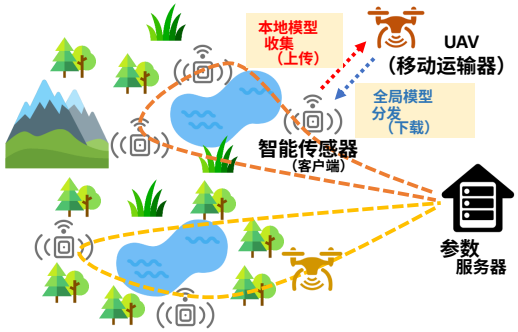

图 1. FedEx 应用于偏远地区智能感知的示意图，这些地区没有通信基础设施。

本地随机梯度下降迭代，然后将结果上传到参数服务器进行聚合。与早期的分布式学习算法（如分布式随机梯度下降 [3]）相比，后者要求每次迭代后都将本地计算结果上传到服务器，本地随机梯度下降（或 FedAvg）在通信效率方面具有明显优势，同时仍能保证收敛。这一理念启发了许多基于不同优化方法开发的后续联邦学习算法，也使联邦学习成为通信受限学习场景中的首选方案；在这些场景中，客户端与服务器之间的带宽可能受限，或者通信模式具有随机性且时断时续。

尽管采用的优化方法和关注的场景各不相同，几乎所有现有研究都默认客户端可以直接 与服务器通信。在异步联邦学习领域，一些研究放宽了这一通信假设 [4], [5] ，允许客户端与服务器之间呈现随机且零散的通信模式，但客户端仍然可以直接与服务器通信，只是通信频率较低。然而，在许多现实系统中，由于缺乏适当的通信基础设施，客户端可能完全无法与服务器直接通信。因此，亟需了解在没有直接服务器-客户端通信的情况下，联邦学习能否正常运行，以及如何在这些场景中优化联邦学习算法。

在本文中，我们提出了一种新的联邦学习框架，称为 FedEx（通过模型快速交付实现的联邦学习），用于

所考虑的没有服务器与客户端直接通信的联邦学习系统，提出了一种名为 FedEx（通过模型快递实现的联邦学习）的新型联邦学习框架。为应对无法直接通信这一挑战，FedEx 设计了一种主动移动机制，利用移动运输工具（例如无人机）在服务器与客户端之间建立间接通信信道，以促进模型信息交换。换言之，移动运输工具充当服务器与客户端之间的中间媒介，用于分发全局模型并收集本地模型更新，其作用类似于传统包裹快递系统中的送货卡车。图 1 展示了 FedEx 在没有通信基础设施的偏远地区应用于智能感知的情形。

然而，间接通信也给 FedEx 的收敛分析与优化带来了重大挑战。首先，在全局模型分发阶段和本地模型收集阶段，移动运输器从一个地点移动到另一个地点需要时间，因此间接通信不可避免地会引入延迟。目前尚不清楚 FedEx 在这种延迟下是否仍能收敛，以及如果能够收敛，收敛速度如何。其次，根据运输工具调度策略的不同，运输器层面的学习可能是同步的，也可能是异步的，这进一步增加了 FedEx 收敛分析的复杂性。第三，客户端会因所在位置以及运输器选择的路线不同而经历异构延迟。因此，FedEx 的性能也取决于如何将客户端分配给运输器，以及运输器如何规划路线。下面总结本文的主要贡献。

- 据我们所知，我们提出了首个通过间接服务器-客户端通信实现的联邦学习框架。根据运输器是否在分配给它们的客户端之间同步巡回，我们提出了两种算法，分别称为 FedEx-Sync 和 FedEx-Async。
- 我们采用虚拟序列技术证明了 FedEx-Sync 和 FedEx-Async 的收敛性。
- 基于收敛界的具体形式，我们提出了一种双层优化算法，用于求解客户端分配与路径设计联合问题。
- 基于两个公开数据集的实验结果验证了 FedEx 的有效性，并与我们的理论结果一致。

II. 问题描述

我们考虑一个包含一台参数服务器和 N个客户端的联邦学习系统，这些客户端分布在广阔区域内，且不具备直接通信能力。换言之，没有任何客户端能够与服务器建立直接通信信道，任意两个客户端之间也无法相互通信。为简化符号表示，我们将服务器编号为 0，并用集合$\mathcal{N} = \{1, 2, \cdots, N\}$为客户端编号。服务器和客户端部署在固定位置，不会移动。已知服务器和客户端的位置坐标后，可以轻松计算出（对称）距离矩阵 $D \in \mathbb{R}^{(N+1)\times(N+1)}$，该矩阵描述任意两台设备之间的距离。具体而言，$D_{0i} = D_{i0}$表示服务器与客户端 i 之间的距离，$D_{ij} = D_{ji}$表示客户端 i 与客户端 j之间的距离。

每个客户端 i 都拥有一个数据集，所有客户端必须协同训练一个机器学习模型，并由服务器进行协调，具体通过求解以下分布式优化问题实现：

$$\min_x f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{\zeta_i}[F_i(x, \zeta_i)], \quad (1)$$

其中，$f_i : \mathbb{R}^d \to \mathbb{R}$ 是客户端i的非凸损失函数，F_i 是基于从客户端 i的数据集中抽取的小批量数据样本 ζ_i 得到的估计损失函数，$x \in \mathbb{R}^d$ 是待学习的模型参数。

为了训练此类机器学习模型，传统的联邦学习框架要求客户端与服务器之间进行周期性或非周期性通信。然而，在我们考虑的场景中，客户端与服务器之间的所有直接通信信道均不存在，因此这些联邦学习框架都无法运行。在下一节中，我们将提出一种新颖的联邦学习框架，以实现在这种极端通信场景（即没有直接通信）下的联邦学习。

III. FedEx：通过模型快递传输实现的联邦学习

为了解决客户端与服务器之间缺乏直接通信的问题，我们的想法是利用移动运输工具（例如无人机），在服务器与客户端之间建立间接通信信道。这些移动运输工具在服务器与客户端之间传输全局/本地模型，就像送货卡车在仓库与客户之间运送包裹一样。我们将这一新的联邦学习框架称为 FedEx，即通过模型快递传输实现的联邦学习。

设 FedEx 中可以使用 K 个移动运输工具。K 的具体取值取决于可用的系统资源，本文将其视为已知量。不失一般性，我们假设所有客户端具有相同的计算速度，且所有运输器具有相同的移动速度。我们将时间离散化为时隙，编号为 $t = 0$、1、2、$\cdots$，其中每个时隙对应于一个客户端完成一个本地训练步骤所需的时长。此外，为简化记号，我们略微滥用符号，将矩阵 D 表示为移动运输器从一个位置移动到另一个位置所需的时间（以时隙计），而非距离。

FedEx 的运行方式是先将客户端划分为 K 互不重叠的子集，再将每个子集分配给一台移动运输器。令 $\mathcal{R}_k \subset \mathcal{N}$ 表示由运输器 k负责的客户端子集。我们有 $\mathcal{R}_k \cap \mathcal{R}_{k'} = \emptyset$，$\forall k \neq k'$且 $\cup_{k=1}^{K} \mathcal{R}_k = \mathcal{N}$。此外，令 $R_k = |\mathcal{R}_k|$ 表示子集 $\mathcal{R}_k$中的客户端数量。对于每台运输器 k，它都会确定一条经过服务器和 $\mathcal{R}_k$中的客户端的往返路线。令z_{ij}，$\forall i$，$j \in \mathcal{R}_k \cup \{0\}$ 表示一个二元变量，用于指示从设备 i 到设备 j 的路径是否包含在巡回中，则往返时间（RTT）可以轻松计算为

$$\Delta_k = \sum_{i=0}^{N} \sum_{j\neq i, j=0}^{N} D_{ij} z_{ij}. \quad (2)$$

为确保巡回恰好经过每台设备一次，我们施加如下约束：每台设备恰有一条入向路径和一条出向路径，可表示为 2($N+1$) 个线性方程：

$$\sum_{i=0,i\neq j}^{N} z_{ij} = 1, \sum_{j=0,j\neq i}^{n} z_{ij} = 1, \forall j = 0, \cdots, N. \quad (3)$$

为给定的一组客户端 $\mathcal{R}_k$ 找到最短巡回，本质上是一个旅行商问题。我们将在了解 FedEx 的收敛行为后，于第五节再优化客户端在运输器之间的分配。目前，我们将 $\mathcal{R}_k$, $\forall k = 1, \cdots, K$ 视为已确定，同时假设相应巡回的往返时间为 RTT Δ_k。

A. FedEx-Sync

在 FedEx 的同步版本 FedEx-Sync 中，每次运输器开始新一轮巡回其分配到的客户端时，它们都会同时从服务器出发。由于运输器的巡回往返时间不同，往返时间较短的运输器需要等待往返时间较长的运输器返回，才能开始下一轮巡回。因此，FedEx-Sync 自然由同步的学习轮次构成，每轮包含 $\Delta \triangleq \max_k \Delta_k$ 个时隙。在每一轮中（将该轮的第一个时隙记为 t_0），会发生以下事件。

- 在每轮开始时，每个运输器都会从服务器下载当前的全局模型 x^{t_0}。随后，运输器根据预先确定的客户端访问顺序，在其分配的客户端之间开始巡回。
- 当某个运输器（例如运输器 k）在时间 $t > t_0$遇到某个客户端（例如客户端 i）时，客户端 i 会下载该运输器 k 当前携带的全局模型，即 x^{t_0}。随后，运输器离开，客户端 i 使用 $x_k^t = x^{t_0}$ 作为初始模型，利用自身的本地数据集训练新的本地模型，直到下一次与运输器相遇。由于运输器需要经过 Δ 个时隙才能再次访问客户端i，因此本地训练将持续 Δ 个时隙。本地训练采用小批量随机梯度下降方法：

$$x_i^{s+1} = x_i^s - \eta g_i^s, \forall s = t, \cdots, t+\Delta-1, \quad (4)$$

其中，$g_i^s = \nabla F_i(x_i^s, \zeta_i^s)$ 是从随机抽取的小批量 ζ_i^s 上计算得到的随机梯度，η 是学习率。设 $m_i^t \in \mathbb{R}^d$ 为客户端 i 自上次与运输器相遇以来、在时间 s 的累积本地更新（CLU），其递归更新方式如下

$$m_i^s = \sum_{s'=t}^{s-1} \eta g_i^s, \forall s = t, \cdots, t+\Delta. \quad (5)$$

- 当运输器（例如运输器 k）在时间 $t > t_0$ 遇到客户端（例如客户端 i）时，客户端 i 也会将其当前的 CLU 上传至运输器 k。然而请注意，此 CLU 是基于上一轮的全局模型得到的，即 $x^{t_0-\Delta}$。运输器 k 会维护一个聚合累积本地更新 u_k^t 在当前巡回期间，以节省存储空间，并在收到新的客户端 CLU 时，根据以下内容对其进行更新

$$u_k^t = u_k^{t-1} + m_i^t. \quad (6)$$

- 当运输器返回服务器时，聚合累积本地更新将用于更新全局模型。在 FedEx-Sync 中，每轮结束时都会同步更新全局模型，具体如下

$$x^{t_0+\Delta} = x^{t_0+\Delta-1} - \frac{1}{N}\sum_{k=1}^{K} u_k^{t_0+\Delta-1}. \quad (7)$$

B. FedEx-Async

FedEx-Sync 中的同步机制是让较快的运输器等待较慢的运输器。不过，这会给较快的运输器带来额外延迟。如果最慢的运输器执行一次往返时间非常长的巡回，那么其他所有运输器都必须等待很久才能开始下一次巡回。在 FedEx-Async 中，我们通过让运输器在完成上一次巡回后立即开始新一轮巡回，消除了这种等待时间。这样一来，更多客户端就能与服务器进行更频繁的全局/本地模型交换。FedEx-Async 与 FedEx-Sync 有许多相似之处，最大的区别在于，每个运输器都有个性化 的学习轮次，这些轮次不一定与其他运输器同步。对于运输器 k，其学习轮次持续 Δ_k 个时隙，每轮都会发生以下事件（将第一个时隙记为 t_0）。

- 在每轮开始时，运输器 k 会从服务器下载当前模型 x^{t_0}。随后，它开始在分配给自己的客户端之间巡回。
- 当运输器 k 在时间 $t > t_0$与客户端 i 会合时，客户端 i 会从运输器 k 下载 x^{t_0}，并将其用作初始模型，训练新的本地模型，直到下次与该运输器会合。与 FedEx-Sync 不同，本地训练将持续 Δ_k 个时隙，而且不同运输器的训练时长各不相同。
- 当运输器 k 在时间 $t > t_0$与客户端 i 会合时，客户端 i还会将其当前的 CLU（基于上一轮全局模型 $x^{t_0-\Delta_k}$ 得到）上传给运输器 k。随后，运输器 k 更新其聚合累积本地更新 u_k^t。
- 运输器返回服务器时，全局模型将按以下方式更新

$$x^{t_0+\Delta_k} = x^{t_0+\Delta_k-1} - \frac{1}{N} u_k^{t_0+\Delta_k-1}. \quad (8)$$

同样，这与 FedEx-Sync 不同，因为服务器无需等待所有运输器都更新全局模型。请注意，多个运输器可能在同一时隙返回服务器（例如t）。在这种情况下，全局模型更新规则改为

$$x^{t_0+\Delta_k} = x^{t_0+\Delta_k-1} - \frac{1}{N}\sum_{k'\in\mathcal{S}^{t_0+\Delta_k}} u_{k'}^{t_0+\Delta_k-1}, \quad (9)$$

其中，$\mathcal{S}^{t_0+\Delta_k-1}$ 是指在时隙 $t_0+\Delta_k-1$返回服务器的客户端集合。

IV. 收敛分析

本节分析 FedEx 的收敛性。由于 FedEx-Sync 可以视为 FedEx-Async 的一种特殊情况，即所有 $\Delta_k, \forall k$ 取值相同，因此我们将重点分析 FedEx-Async 的收敛性。

A. 客户端训练对齐

在分析 FedEx 的收敛性之前，我们先介绍一种等效的 FedEx 视角，在该视角下，同一子集中的客户端本地训练是对齐的。考虑运输器 k 承载全局模型 x^{t_0}的一轮学习。由于 $\mathcal{R}_k$中客户端的位置各不相同，客户端会在不同的时隙接收到 x^{t_0}，并在不同的时隙开始新一轮本地训练。完成当前一轮本地训练后，它们基于 x^{t_0} 生成的 CLU 将通过运输器在时隙 $t_0 + 2\Delta_k$上传至服务器。此时，全局模型将利用这些 CLU 进行更新。

客户端未对齐的本地训练，即使由同一运输器覆盖，也会给 FedEx 的收敛分析带来重大挑战。幸运的是，存在一种等效（但假想）的客户端训练流程，能够生成完全相同的全局模型序列。具体而言，设想 $\mathcal{R}_k$ 中的客户端在时隙 t_0立即接收全局模型 x^{t_0}，并进行 Δ_k 个时隙的本地训练。然后，它们的 CLU 会延迟一轮上传至服务器。也就是说，在时隙 $t_0 + 2\Delta_k$，全局模型将进行更新。显然，这一改变完全不会影响全局模型更新，但同一子集$\mathcal{R}_k$ 中客户端的本地训练现在已完全对齐。由于我们关注的是全局模型的收敛性，因此在收敛性证明中采用这种等效的对齐客户端训练流程。该流程对齐了客户端的本地训练，同时不影响全局模型的演变。从本质上说，这种对齐将下载延迟转移到了上传阶段，但总延迟保持不变。经过这一改变，FedEx-Sync 就成为一种熟悉的同步联邦学习算法，只是其 CLU 上传延迟为一轮。除了 CLU 上传延迟外，Fed-Async 在不同客户端子集之间仍采用异步学习。

B. 假设

我们的收敛分析将采用以下标准假设。

假设 1（Lipschitz平滑性）。 存在一个常数$L > 0$，使得 $\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|$，$\forall x,\ y \in \mathbb{R}^d$且 $\forall i = 1,\ \cdots,\ N$。

假设 2（无偏局部梯度估计）。 局部梯度估计是无偏的，即 $\mathbb{E}_\zeta F_i(x,\ \zeta) = \nabla f_i(x)$，$\forall x$且 $\forall i = 1,\ \cdots,\ N$。

假设3 (有界梯度)。 存在一个常数$G > 0$，使得 $\mathbb{E}\|\nabla F_i(x,\ \zeta)\|^2 \leq G^2$，$\forall x \in \mathbb{R}^d$ 且$\forall i = 1,\ \cdots,\ N$。

假设4 (有界方差)。 存在一个常数 σ >0，使得 $\mathbb{E}_\zeta\|\nabla F_i(x,\ \zeta) - \nabla f_i(x)\|^2 \leq \sigma^2$，$\forall x \in \mathbb{R}^d$ 且$\forall i = 1,\ \cdots,\ N$。

C. 收敛界

我们的收敛分析依赖于理解全局模型的两个序列之间的关系及其演变。全局模型的真实序列 是指服务器随时间维护的实际全局模型，可根据 FedEx 计算如下：

$$x^t = x^0 - \frac{1}{N}\sum_{i=1}^{N}\sum_{s=0}^{\phi_i(t)}\eta g_i^s, \forall t, \tag{10}$$

其中，我们将 $\phi_i(t)$ 定义为截至时间 t已收到客户端 i 所有相应梯度的时隙。换句话说，在时隙 t，服务器已收到来自客户端 i（通过运输器）的梯度 g_i^0、$\cdots$、$g_i^{\phi_i(t)}$。在 FedEx-同步中，所有客户端与服务器具有相同的间接通信模式，因此 $\phi_i(t) = \phi_j(t)\ \forall i,\ j \in \mathcal{N}$。在 FedEx-异步中，属于同一运输器的客户端与服务器具有相同的间接通信模式，因此，$\phi_i(t) = \phi_j(t)\ \forall i,\ j \in \mathcal{R}_k,\ \forall k$。

全局模型的虚拟序列定义于这样一种假想情形：所有客户端的梯度在计算完成后立即上传至服务器。 [6]中也采用了类似的虚拟序列。然而，我们的收敛性证明针对本文中的特定问题进行了设计，与所有先前的工作均不同。具体而言，虚拟序列定义如下

$$v^t = x^0 - \frac{1}{N}\sum_{i=1}^{N}\sum_{s=0}^{t-1}\eta g_i^s, \forall t. \tag{11}$$

显然，由于客户端梯度延迟上传，真实序列与虚拟序列之间存在差异。对于 FedEx-Sync，该延迟的上界为

$$(t-1) - \phi_i(t) \leq 2\Delta, \forall i \in \mathcal{N}. \tag{12}$$

对于 FedEx-Async，该延迟的上界为

$$(t-1) - \phi_i(t) \leq 2\Delta_k, \forall i \in \mathcal{R}_k, \forall k. \tag{13}$$

引理1。 真实全局模型与虚拟全局模型之间的差异可界定如下

$$\mathbb{E}\|v^t - x^t\|^2 \leq \frac{4\eta^2 G^2}{N}\sum_{k=1}^{K} R_k\Delta_k^2. \tag{14}$$

所有客户端的本地模型与虚拟全局模型之间的平均差异有如下界

$$\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\|v^t - x_i^t\|^2 \leq \frac{18\eta^2 G^2}{N}\sum_{k=1}^{K} R_k\Delta_k^2. \tag{15}$$

定理1。 通过设置学习率 $0 < \eta \leq 1/L$，我们得到

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(x^t)\|^2$$

$$\leq \frac{4}{\eta T}(f(x^0)-f^*)+\frac{44\eta^2G^2L^2}{N}\sum_{k=1}^{K}R_k\Delta_k^2+\frac{2L\eta\sigma^2}{N}. \quad (16)$$

注1。对于 $T \geq N^3$，将学习率设置为 $\eta = \frac{\sqrt{N}}{L\sqrt{T}}$ 后，收敛界可恢复经典同步联邦学习 [7] 的相同 $O(\frac{1}{\sqrt{NT}})$ 收敛速率。

注释 2. 对于 FedEx-Sync，由于真实序列与虚拟序列会周期性地彼此重合，因此收敛界可以进一步略微收紧。此外，由于所有运输器的有效往返时间均为 $\max_k \Delta_k$，收敛界可简化为

$$\frac{2}{\eta T}(f(x^0)-f^*)+18\eta^2G^2L^2(\max_k \Delta_k)^2+\frac{L\eta\sigma^2}{N}. \quad (17)$$

V. 客户端分配与路径设计

本节中，我们研究联合客户端分配与路径设计问题，以优化 FedEx 的收敛界。

A. 问题描述

鉴于运输器可用数量有限以及物联网设备可能的大规模部署，我们考虑一种典型设置：客户端数量远大于运输器数量，即 $N \gg K$。令 $a_i \in \{1,\cdots,K\}$ 为客户端 i 的分配变量，用于指示其被分配给哪个运输器。我们还将所有客户端的分配变量汇集于 $\boldsymbol{a}=(a_1, \cdots, a_N)$ 中。显然，$\mathcal{R}_k = \{i : a_i = k\}$。

给定为每个运输器 k 分配的客户端 $\mathcal{R}_k$，我们可以设计一条路线以最小化往返时间。设 $\Delta_k(\mathcal{R}_k)$ 为给定一组客户端 $\mathcal{R}_k$ 时的最小往返时间。或者，$\Delta_k(\mathcal{R}_k)$ 也可以写作 $\Delta_k(\boldsymbol{a})$，因为 $\mathcal{R}_k$ 由 $\boldsymbol{a}$ 决定。客户端分配需要解决以下优化问题

$$\text{FedEx-Sync}: \min_{\boldsymbol{a}} \max_k \Delta_k(\boldsymbol{a}), \quad (18)$$

$$\text{FedEx-Async}: \min_{\boldsymbol{a}} \sum_k R_k(\boldsymbol{a})\Delta_k^2(\boldsymbol{a}). \quad (19)$$

上述问题是一个难以求解的组合优化问题。接下来，我们提出一种新算法来解决这个问题。

B. 双层优化

我们提出了一种双层优化算法，称为 CARD（客户端分配与路径设计的简称）。

1) 内层优化：内层优化是在给定客户端集合 $\mathcal{R}_k$ 的情况下，为每个运输器求解最小往返时间，即对于给定的 $\boldsymbol{a}$，计算 $\Delta_k(\boldsymbol{a})$。这是一个经典的旅行商问题（TSP） [8], [9], [10]。考虑到动态规划的高复杂度（即 $O(2^n n^2)$，其中 n 为节点数），在我们的实现中，我们使用时间复杂度为 $O(n^2)$ 的启发式算法 2-OPT [11], , 来计算给定 $\boldsymbol{a}$ 下的 $\Delta_k(\boldsymbol{a})$。

2) 外层优化：为解决用于确定最优客户端分配的外层优化问题，我们采用吉布斯采样。为简化符号表示，我们使用统一的代价函数 $C(\boldsymbol{a})$；对于 FedEx-Sync，该函数等于 $\max_k \Delta_k(\boldsymbol{a})$，而对于 FedEx-Async，则等于 $\sum_k R_k(\boldsymbol{a})\Delta_k^2(\boldsymbol{a})$。CARD 算法按照预定义的顺序访问每个客户端，在保持其他客户端的分配决策不变的情况下，生成其分配决策的概率分布，并根据该分布采样新的分配决策。重复这一过程足够多次后，可以确保分配以较高概率收敛到最优解。

VI. 实验

在本节中，我们使用仿真网络环境和标准公开数据集评估 FedEx 的性能。

A. 实验设置

网络。我们模拟一个没有直接通信的网络，其中一台参数服务器和40个客户端分布在一个区域内，如图2所示。整个区域首先被划分为10个大小相等的区块（即4个单位 $\times$ 10 单位），每个区块内随机分布4个客户端。在大多数实验中，我们模拟 $K = 4$ 个移动运输工具，其移动速度均为每时隙4个单位。

深度神经网络模型和数据集。我们在两个公开数据集上进行联邦学习实验，即 FMNIST [12]和 SVHN [13]。对于这两个数据集，我们都采用 LeNet [14]作为骨干模型。

数据集划分。在 FMNIST 实验中，每个客户端拥有60个训练数据样本。在 SVHN 实验中，每个客户端拥有800个训练数据样本。我们模拟了两种数据分布。

- 独立同分布：客户端的数据分布为独立同分布。
- 非独立同分布：我们使用狄利克雷方法创建非独立同分布数据集，该方法广泛应用于联邦学习研究，例如 [15]。在 FMNIST 实验中，我们使用参数为 0.3 的狄利克雷分布；在 SVHN 实验中，使用参数为 0.5 的狄利克雷分布。

运输器路线。我们评估了采用不同目标函数来解决客户端分配问题的多种运输器路线设计：

- 最小-最大：$\max_k \Delta_k(\boldsymbol{a})$。
- 加权平方和（SWS）：$\sum_k R_k(\boldsymbol{a})\Delta_k^2(\boldsymbol{a})$。
- 最短总距离：$\sum_k \Delta_k(\boldsymbol{a})$。

B. 结果

独立同分布数据下的 FedEx-Sync 与 FedEx-Async 比较。我们首先比较独立同分布数据下 FedEx-Sync（采用最小-最大路线）与 FedEx-Async（采用 SWS 路线）的性能。图3和图4分别展示了它们在 FMNIST 和 SVHN 上的收敛曲线。可以看出，在独立同分布数据下，两种算法都能收敛，但 FedEx-Async 的表现优于 FedEx-Sync。这是合理的，因为 FedEx-Sync 需要额外花时间等待速度最慢的移动运输工具。

非独立同分布数据下的 FedEx-Sync 与 FedEx-Async。接下来，我们比较非独立同分布数据下的 FedEx-Sync 和 FedEx-Async。图

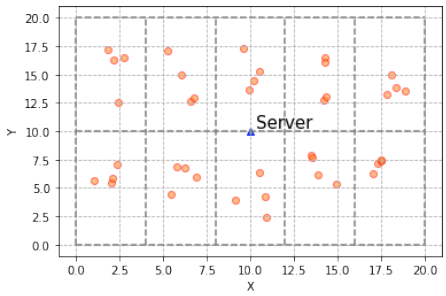

图2. 网络示意图

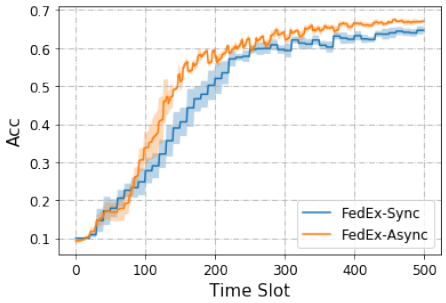

图3. FedEx 在 FMNIST 的 i.i.d 数据上的表现

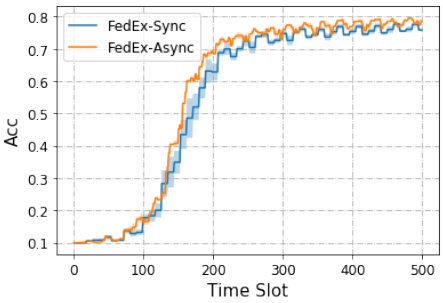

图4. FedEx 在 SVHN 的 i.i.d 数据上的表现

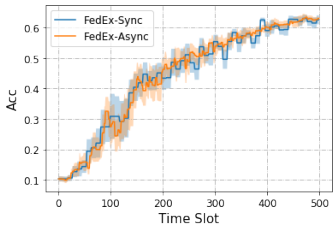

图5. FMNIST 上的非 i.i.d 数据

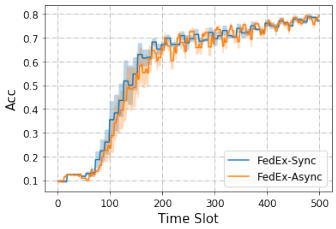

图6. SVHN 上的非 i.i.d 数据

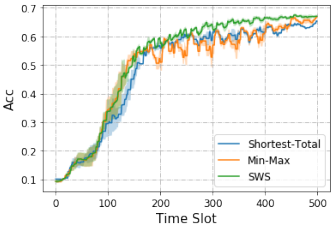

图7. 路线的影响（异步）

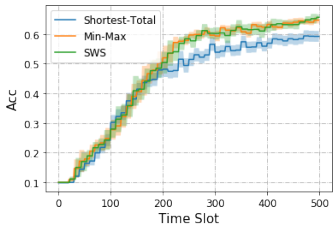

图 8. 路线的影响（同步）

5 和图 6 表明，FedEx-Sync 的表现与 FedEx-Async 相近。尽管各个客户端的数据可能高度非独立同分布，但当运输器路线上的客户端数量足够多时，运输器所汇集的客户端数据整体仍可能与全部数据分布相似。

路线的影响。我们通过实证研究考察了不同运输器路线设计的影响。图 7 比较了不同路线设计下 FedEx-Async 的收敛情况。结果与定理1中的理论分析一致，即 SWS 设计可实现最佳收敛性能。图 8 展示了 FedEx-Sync 的结果。在这种情况下，SWS 中得到的最长单个往返时间与最小-最大中得到的相同，因此 SWS 也是最小-最大问题的另一种解。因此，最小-最大路线和 SWS路线的收敛性能相近，且优于最短总距离，这与定理1中的理论分析一致。

VII. 结论

本文提出了一种基于间接服务器-客户端通信的新型联邦学习框架，以支持在没有通信基础设施的场景下进行分布式机器学习。我们提出了两种新颖的算法，利用移动运输工具通过设备到设备通信分发全局模型并收集本地模型。我们针对非凸损失函数和非独立同分布数据分布，在任意运输器路线下对这些算法进行了新颖的收敛分析。研究结果为联合客户端分配与路线设计提供了有原则的指导。

参考文献

[1] S. U. Stich，“本地随机梯度下降收敛快、通信开销小”，*arXiv*预印本 *arXiv:1805.09767*，2018。

[2] B. McMahan、E. Moore、D. Ramage、S. Hampson 和 B. A. y Arcas，“基于去中心化数据的深度网络通信高效学习”，载于人工智能与统计。PMLR，2017，第1273–1282页。[3] M. F. Balcan、A. Blum、S. Fine 和 Y. Mansour，“分布式学习、通信复杂度与隐私”，载于学习理论会议。JMLR研讨会与会议论文集，2012，第26–1页。[4] D. Avdiukhin 和 S. Kasiviswanathan，“任意通信模式下的联邦学习”，载于国际机器学习会议。PMLR，2021，第425–435页。[5] D. Basu、D. Data、C. Karakus 和 S. Diggavi，“Qsparse-local-sgd：采用量化、稀疏化和本地计算的分布式随机梯度下降”，神经信息处理系统进展，第32卷，2019。[6] K. Yuan、Q. Ling 和 W. Yin，“关于去中心化梯度下降的收敛性”，*SIAM*优化期刊，第26卷，第3期，第1835–1854页，2016。[7] H. Yu、S. Yang 和 S. Zhu，“采用更快收敛和更少通信的并行重启随机梯度下降：揭示模型平均为何适用于深度学习”，载于*AAAI*人工智能会议论文集，第33卷，第01期，2019，第5693–5700页。[8] S. Lin 和 B. W. Kernighan，“一种有效的旅行商问题启发式算法”，运筹学，第21卷，第2期，第498–516页，1973。[9] M. M. Flood，“旅行商问题”，运筹学，第4卷，第1期，第61–75页，1956。[10] M. J¨unger、G. Reinelt 和 G. Rinaldi，“旅行商问题”，运筹学与管理科学手册，第7卷，第225–330页，1995。[11] G. A. Croes，“一种求解旅行商问题的方法”，运筹学，第6卷，第6期，第791–812页，1958。[12] H. Xiao、K. Rasul 和 R. Vollgraf，“Fashion-MNIST：用于机器学习算法基准测试的新型图像数据集”，*arXiv*预印本 *arXiv:1708.07747*，2017。[13] Y. Netzer、T. Wang、A. Coates、A. Bissacco、B. Wu 和 A. Y. Ng，“利用无监督特征学习读取自然图像中的数字”，2011。[14] Y. LeCun、L. Bottou、Y. Bengio 和 P. Haffner，“应用于文档识别的基于梯度的学习”，*IEEE*论文集，第86卷，第11期，第2278–2324页，1998。[15] H.-Y. Chen 和 W.-L. Chao，“FedBE：使贝叶斯模型集成适用于联邦学习”，*arXiv*预印本 *arXiv:2009.01974*，2020。