

Context: The model training participation rate of client devices is low while the high participation rate is crucial for the global model performance.

Problem: Although the system relies greatly on the participation of clients to produce high-quality global models, clients are not mandatory to join the model training and contribute their data/resources.

Forces: The problem requires to balance the following forces:

- *Incentive scheme.* It is challenging to formulate the form of rewards to attract different clients with different participation motives. Furthermore, the incentive scheme needs to be agreed upon by both the learning coordinator and the clients, e.g., performance-based, data-contribution-based, resource-contribution-based, and provision-frequency-based.
- *Data privacy.* To identify the contribution of each client device, the local data and client information is required to be studied and analysed by the central server. This exposes the client devices' local data to privacy threats.

Solution: An incentive registry records all client's contributions and incentives based on the rate agreed. There are various ways to formulate the incentive scheme, e.g., deep reinforcement learning, blockchain/smart contracts, and the Stackelberg game model.

Consequences:

Benefits:

- *Client motivatability.* The incentive mechanism is effective in attracting clients to contribute data and resources to the training process.

Drawbacks:

- *System security.* There might be dishonest clients that submit fraudulent results to earn rewards illegally and harm the training process.

Related patterns: *Client Registry, Client Selector*

Known uses:

- *FLChain* [5] is a federated learning blockchain providing an incentive scheme for collaborative training and a market place for model trading.
- *DeepChain* [42] is a blockchain-based collaborative training framework with an incentive mechanism that encourages parties to participate in the deep learning model training and share the obtained local gradients.
- *FedCoin* [29] is a blockchain-based peer-to-peer payment system for federated learning to enable Shapley Value (SV) based reward distribution.

3.4. Model Aggregation Patterns

Model aggregation patterns are design solutions of model aggregation used for different purposes. *Asynchronous aggregator* aims to reduce aggregation latency and increase

system efficiency, whereas *decentralised aggregator* targets to increase system reliability and accountability. *Hierarchical aggregator* is adopted to improve model quality and optimises resources. *Secure aggregator* is designed to protect the models' security.

3.4.1. Pattern 11: Asynchronous Aggregator

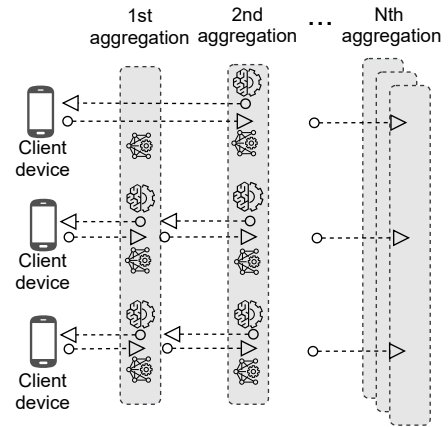


Figure 14: Asynchronous Aggregator.

Summary: To increase the global model aggregation speed every round, the central server can perform model aggregation asynchronously whenever a model update arrives without waiting for all the model updates every round. In Fig. 14, the asynchronous aggregator pattern is demonstrated as the first client device asynchronously uploads its local model during the second aggregation round while skipping the first aggregation round.

Context: In conventional federated learning, the central server receives all local model updates from the distributed client devices synchronously and performs model aggregation every round. The central server needs to wait for every model to arrive before performing the model aggregation for that round. Hence, the aggregation time depends on the last model update that reaches the central server.

Problem: Due to the difference in computation resources, the model training lead time is different per device. Furthermore, the difference in bandwidth availability, communication efficiency affects the model's transfer rate. Therefore, the delay in model training and transfer increases the latency in global model aggregation.

Forces: The problem requires the following forces to be balanced:

- *Model quality.* There will be possible bias in the global model if not all local model updates are aggregated in every iteration as important information or features might be left out.
- *Aggregation latency.* The aggregation of local models can only be performed when all the model updates are collected. Therefore, the latency of the aggregation process