Logical foundations for correct communication of distributed machine learning

Silvia Ghilezan

University of Novi Sad & Mathematical Institute SASA

LAP 2025
IUC, Dubrovnik
24-28 September 2025

The talk is based on joint work with

Miodrag Djukić, Ivan Kaštelan, Miroslav Popović, Marko Popović, Ivan Prokić, Simona Prokić (U. Novi Sad), Alceste Scalas (TU Copenhagen) and Nobuko Yoshida) (U. Oxford).

Federated Learning + Formal Verification

- **Federated learning (FL)** – decentralised machine learning setting where clients
  - keep training data decentralised (private data) while
  - collaboratively train a model (local data)

- **Formal verification (FV)** – a process of mathematically checking that the behaviour of a system satisfies a given property

Sounds great ✓ but there was zero previous work to build upon ✗

To build a common language of the two working communities ??

Roadmap

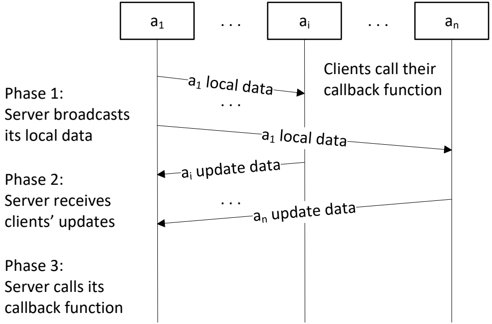
1. Federated Learning: Distributed machine learning
2. Verification: Untyped calculus and model checking
3. Verification: Typed calculus and correctness-by-construction

Outline

1. Federated Learning: Distributed machine learning
2. Verification: Untyped calculus and model checking
3. Verification: Typed calculus and correctness-by-construction

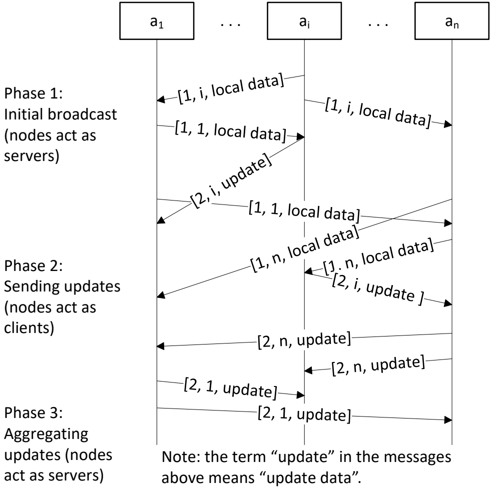
FL Centralised - star topology

- The algorithm goes in 3 phases where:
  - local data is the local machine learning model
  - private data is training data
- At this point our focus was on the communication pattern:
  - broadcasting
  - receiving from clients in any order!



FL Decentralised - clique topology

- The algorithm goes in 3 phases where:
  - local data is local machine learning model
  - private data is training data
- At this point our focus was on the communication pattern:
  - broadcasting
  - receiving from clients in any order!



PTB-FLA

- Python Testbed for Federated Learning Algorithms (PTB-FLA)
- The primary intention is to use PTB-FLA as **a framework for developing federated learning algorithms** on small smart devices (swarm, edge computing)
- PTB-FLA supports both
  - the generic **centralised** one-shot FLA execution
  - the generic **decentralised** one-shot FLA execution

Popovic, M., Popovic, M., Kastelan, I., Djukic, M., Ghilezan, S.: A simple python testbed for federated learning algorithms. In: ZINC 2023. pp. 148-153 (2023).

Outline

Untyped calculus and model checker

- Calculus: Communicating Sequential Processes (CSP)
  - Formal language for describing patterns of interaction in concurrent systems
  - Introduced in the late 1970s by Tony Hoare
- Model checker: Process Analysis Toolkit (PAT)
- Other process calculi: Calculus of Communicating Systems (CCS), π-calculus

Formal verification of FL protocols

- Specification: FL protocols are modelled in CSP
- Verification: PAT model checker is used to prove
  - **deadlock freedom** and
  - **termination**

  of the two CSP models (top down)

CSP Model for Generic Centralised FLA Algorithm

```
// PTB-FLA
...
channel server2client[NoNodes-1] 1;
channel clients2server NoNodes-1;

FlCentralised(noNodes, nodeId, flSrvId, ldata, pdata) =
  if(nodeId == FlSrvId) {
    CeServer(noNodes, nodeId, flSrvId, ldata, pdata)
  } else {
    CeClient(noNodes, nodeId, flSrvId, ldata, pdata)
  };

CeServer(noNodes, nodeId, flSrvId, ldata, pdata) =
  {terminated = False} ->
  CeBroadcastMsg(0, noNodes, nodeId, ldata);
  CeRcvMsgs(0, noNodes-1);
  {terminated = True} -> Skip;

CeBroadcastMsg(id, noNodes, nodeId, ldata) =
  if(id != nodeId) {
    server2client[id]!ldata -> Skip
  };
  if(id < noNodes-1) {
    CeBroadcastMsg(id+1, noNodes, nodeId, ldata)
  };

CeRcvMsgs(i, noMsgs) =
  if(i < noMsgs) {
    clients2server?update -> CeRcvMsgs(i+1, noMsgs)
  };
```

Model checking with PAT

Centralised

```
####assert SysCentralised() deadlockfree;
####define Terminated (terminated == True);
####assert SysCentralised() reaches Terminated;
####assert SysCentralised() |= []<> Terminated;
```

- CSP models generated **manually** from the Python code
  - I. Prokić, S. Ghilezan, S. Prokić, M. Popovic, M. Popovic, I. Kaštelan. Correct orchestration of Federated Learning generic algorithms: formalisation and verification in CSP
    ECBS 2023 - Engineering of Computer-Based Systems LNCS 14390, 274–288 (2023)
- CSP models generated **automatically** from the Python code
  - M. Djukic, I. Prokic, M. Popovic, S. Ghilezan, M. Popovic, S. Prokic. Correct orchestration of federated learning generic algorithms: Pyton translation to CSP and verification in PAT. *International Journal on Software Tools and Technology Transfer* 27(1), 21-34 (2025).

Outline

1. Federated Learning: Distributed machine learning
2. Verification: Untyped calculus and model checking
3. Verification: Typed calculus and correctness-by-construction

Multiparty session types (MPST)

- Typed calculus: Multiparty session types (MPST)
  - π-calculus like formal language for the verification of message-passing programs
  - Introduced by Honda, Yoshida, Carbone (2008,2016)
- MPST express protocols-as-types and use type checking to verify whether one or more communicating processes correctly implement some desired protocols
- MPST enjoy (guarantee) safety, deadlock-freedom, liveness and other good communication properties
- Corectness-by-construction

MPST

Process calculi	π-calculus, CCS, CSP,...	Process
Session types	2 participants communication	$!p$-send $?q$-receive
MPST	multiple participants	$r \triangleleft !p.?q...,\ p \triangleleft !s.?r....$
Subtyping	enables flexibility	reconfiguration of communication

Advantages

- Subject reduction
- Progress
- Liveness
- Deadlock freedom
- Different extensions for real-life problems

Disadvantages

- insufficient to model arbitrary order of message arrivals

MPST for FL

- extension of MPST, specially for this purpose
- as a novelty, MPST supports
  input/output operations directed towards multiple participants at the same time
- the extension is proven to enjoy safety, deadlock-freedom, liveness, and session fidelity properties
- FL algorithms are modelled in MPSTs
- this approach paves the way for more scalable and efficient techniques for verification and analysis of distributed machine learning algorithms based on correctness-by-construction.

I. Prokić, S. Prokić, S. Ghilezan, A. Scalas, N. Yoshida. On Asynchronous Multiparty Session Types for Federated Learning. ICTAC 2025.

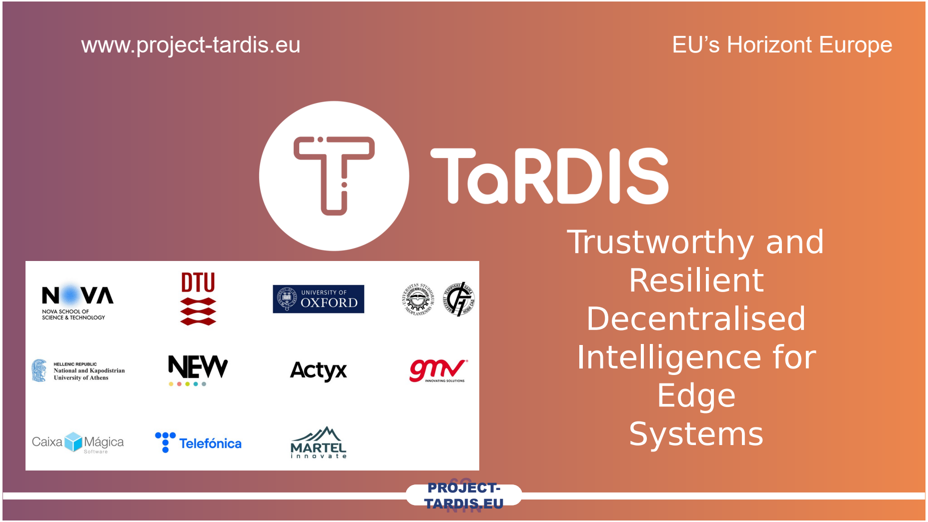
Conclusion and further work

Done

- to automatise the translation of the Python code into the CSP model
- to develop MPST for FL

Further investigations

- to verify PTB-FLA CSP models in Maude
- to identify and prove more properties relevant to FL
- Curry-Howard correspondence paradigm new challenges


www.project-tardis.eu
EU’s Horizont Europe
TaRDIS
Trustworthy and Resilient Decentralised Intelligence for Edge Systems
NOVA
NOVA SCHOOL OF SCIENCE & TECHNOLOGY
DTU
UNIVERSITY OF OXFORD
HELLENIC REPUBLIC
National and Kapodistrian University of Athens
NEW
Actyx
gmv
INNOVATING SOLUTIONS
Caixa Mágica
Software
Telefónica
MARTEL
innovate
PROJECT-TARDIS.EU