

Security, Privacy and Ethics by Design in Decentralized Computing

Josep Domingo-Ferrer

Universitat Rovira i Virgili, Tarragona, Catalonia
CYBERCAT-Center for Cybersecurity Research of Catalonia



*Chair in
Data Privacy*

josep.domingo@urv.cat

Edinburgh, May 24, 2022

Contenido

- 1 Introduction
- 2 Some limitations in decentralized computing
- 3 Rationality and co-utility
- 4 Co-utile approach to ethics-by-design in decentralized computing
- 5 Hints on how to address the above limitations
- 6 Conclusions

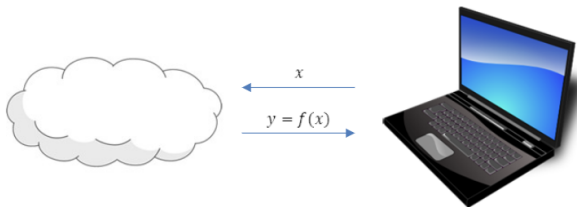
Introduction

- The history of computing is marked by a succession of centralization and decentralization periods.
- Distributed computing, edge computing and fully decentralized computing herald today's decentralization.
- **Technical advantages:** scalability, latency reduction, efficiency, load balancing, avoidance of single points of failure, etc.
- **Ethical advantages:** privacy, fairness, autonomy.

Hurdles to decentralization

- How to guarantee that all agents involved in a computation (peers, work requesters or workers) perform as expected?
- Accidental failure can be mitigated technically.
- Intentional deviation by *malicious attackers* or *free-riders* is problematic.
- Incentivizing the *ethical* behavior of agents would be a turning point.

Computation outsourcing



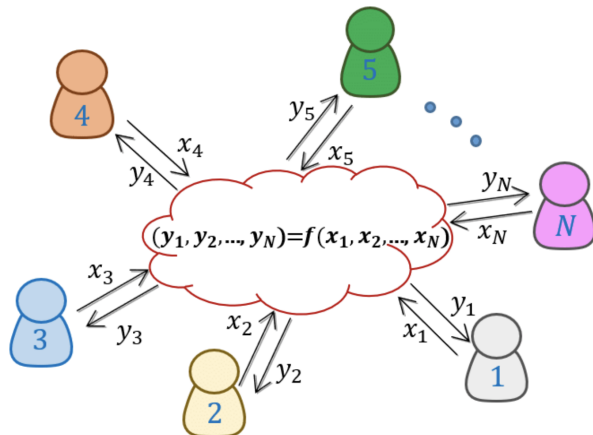
Computation outsourcing: security and privacy problems

- Edge and fog computing allow, *inter alia*, a computation requester to assign tasks to workers with free resources.
- In general, requesters do not trust workers.
- **Security problem:** How to make sure computations are correct?
- **Privacy problem:** How to prevent workers from learning the requester's sensitive data?

Computation outsourcing: state of the art and limitations

- *Verifiable computing* uses worker replication, trusted hardware or correctness proofs (SNARGs, SNARKs). **Limitations:** redundancy causes overhead, special hardware excludes a lot of edge workers, cryptographic proofs are *ad hoc*.
- *Accountable cryptography*. **Limitation:** it requires enforceable *ex post* punishment.
- *Game-theoretic approaches* rely on external payments or blockchain. **Limitations:** external infrastructures are beyond the participants' control and they may not be available or not be rationally sustainable.

Multiparty computation (MPC)

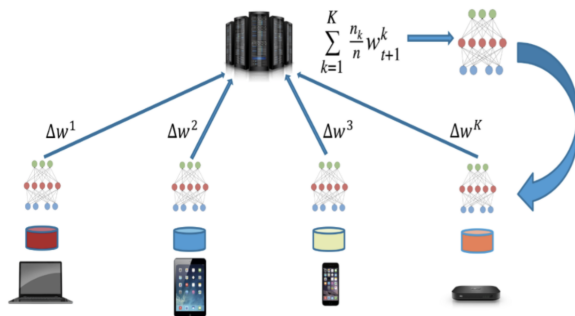


Multiparty computation: state of the art and limitations

- MPC consists in several parties engaging in joint computation so that each party's input and output remain private to that party.
- Thanks to compilers, MPC is general-purpose nowadays.
- **Limitations:** Since compilers need to translate the code of computation to circuits, substantial programming effort is required to unroll loops and recursive calls, which must be bounded.

Decentralized learning

Federated learning (FL)



Decentralized learning: security and privacy problems

- Federated learning (FL) and fully decentralized machine learning (FDML) provide scalability and some worker privacy (workers do not need to share their local data).
- **Security problem:** Vulnerability to Byzantine attacks (prevent convergence) or poisoning attacks (converge to a wrong global model).
- **Privacy problem:** Model updates sent by workers may allow inferences on their local data.

Decentralized learning: state of the art and limitations

- Security defenses are based on the model manager detecting outlying updates or assessing model degradation.
- Privacy defenses are based on the workers securely aggregating their updates (via MPC) or adding noise to them (via differential privacy, DP).
- Limitations: Security defenses are based on the manager seeing updates, whereas privacy defenses either prevent it (MPC) or cause accuracy loss (DP). Security-privacy-accuracy conflict!

A pragmatic security model for protocols

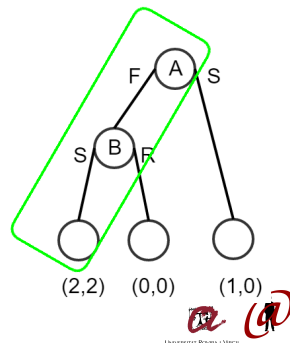
Most protocols defining the interaction among agents work in one of two security models:

- *Semi-honest model*: Agents follow the protocol but may be curious to learn about other agents;
- *Malicious model*: Agents may arbitrarily deviate from the protocol rules.

I contend the *rational model* is more pragmatic: protocols should be designed to incentivize the adherence of agents.

Rationality and co-utility

- The above limitations can be greatly alleviated through adequate incentivization of agents.
- Co-utility is a type of interaction between rational agents in which the best option for each agent to reach her own goal is to help other agents reach theirs.
- E.g., F-S is a co-utile protocol.



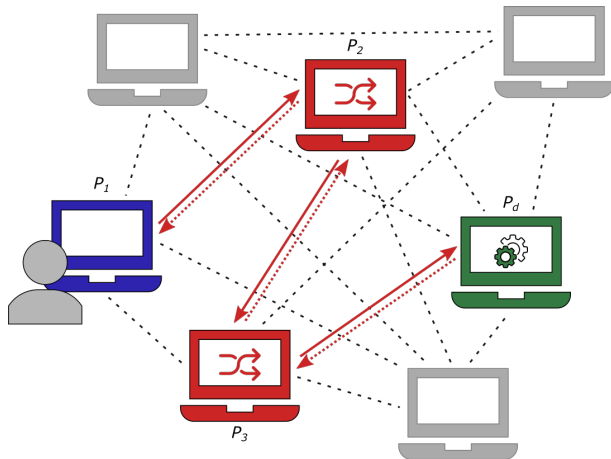
Co-utility

- Protocol Π is co-utile if all the following holds:
 - 1 It is self-enforcing (sticking to the next protocol step is an equilibrium of the remaining subgame);
 - 2 The utility (payoff) derived by each agent participating in Π is greater than the utility of not participating;
 - 3 Π is Pareto-optimal (there is no alternative protocol Π' whereby all agents are better off).
- E.g., if two agents P_1 and P_2 want to protect against profiling by a WSE, it is rational for P_1 to ask P_2 to submit P_1 's queries and it is rational for P_2 to accept if P_1 's queries are useful noise for P_2 's profile of interests.

A co-utile approach to ethics by design

- 1 Identify the **ethical values** (autonomy, privacy, security, fairness, transparency, etc.) relevant to a particular decentralized computing scenario.
- 2 Devise **artificial incentives** to make ethical behavior co-utile when it is not naturally co-utile (e.g., tit-for-tat, co-utile decentralized reputation, etc.).
- 3 Construct a **co-utile anonymous communication channel**, based on multi-hop forwarding of messages among peers, which provides **privacy** as **unlinkability**.

The co-utile anonymous channel



Co-utile computation outsourcing

- The main ethical value in computation outsourcing is **security** (correct results).
- Computing correctly must bring a reputation increase to the worker.
- A worker can become a requester only after accumulating a high reputation (so that reputation is useful).
- Reputation is less expensive than replication and heavy cryptographic protocols, and more flexible than *ad hoc* protocols¹.

¹J. Domingo-Ferrer, A. Blanco-Justicia, D. Sánchez and N. Jebreel. "Co-utile peer-to-peer decentralized computing", in *20th IEEE/ACM Intl. Symp. on Cluster, Cloud and Internet Computing – CCGrid 2020*, pp. 31-40, 2020.

Co-utile multiparty computation

- Let $(O_1, \dots, O_m) = C(I_1, \dots, I_m)$ be a computation to be carried out by parties $P_1, \dots, P_m$, where C is the computation code, I_i is P_i 's private input and O_i is P_i 's private output.
- Each P_i can prune C into the part C_i needed to obtain O_i .
- Then, P_i outsources C_i to peer worker P_{w_i} using the **co-utile anonymous channel**, so that P_{w_i} does not know who P_i is (**privacy**).
- All $P_1, \dots, P_m$ also use the co-utile anonymous channel to send their inputs $I_1, \dots, I_m$ to every other peer (in particular to P_{w_i}).
- MPC entails **autonomy**, and the co-utile anonymous channel provides **security** and **privacy**.
- The code C can be a normal programming code with non-bounded loops and recursions².

²J. Domingo-Ferrer and J. Manjón, "Circuit-free general-purpose multi-party computation via co-utile unlinkable outsourcing", *IEEE Trans. Dependable and Secure Computing*, to appear.

Co-utile decentralized learning

- Workers can use the co-utile anonymous channel to send updates to the manager (**privacy**).
- The manager still sees exact individual updates, so that it can apply **security** defenses.
- Good worker updates can be incentivized using co-utile outsourcing (with reputation). Reputation rewards must be awarded without breaking worker privacy ³.

³J. Domingo-Ferrer, A. Blanco-Justicia, J. Manjón and D. Sánchez, "Secure and privacy-preserving federated learning via co-utility", *IEEE Internet of Things Journal*, 9(5):3988-4000, 2022.

Conclusions and future research

- Decentralized computing is attractive for many reasons, but agents are autonomous and adherence to protocols cannot be taken for granted.
- If agents can be assumed rational, protocols can be designed so that adherence to prescribed ethical values is the rational agent behavior.
- Co-utility is a useful design principle.
- We have sketched how to address current limitations in computation outsourcing, MPC and decentralized learning.
- Future research can investigate how to enforce values that are not interactively peer-checkable (e.g., **fairness**) and tackle other decentralized computing scenarios.