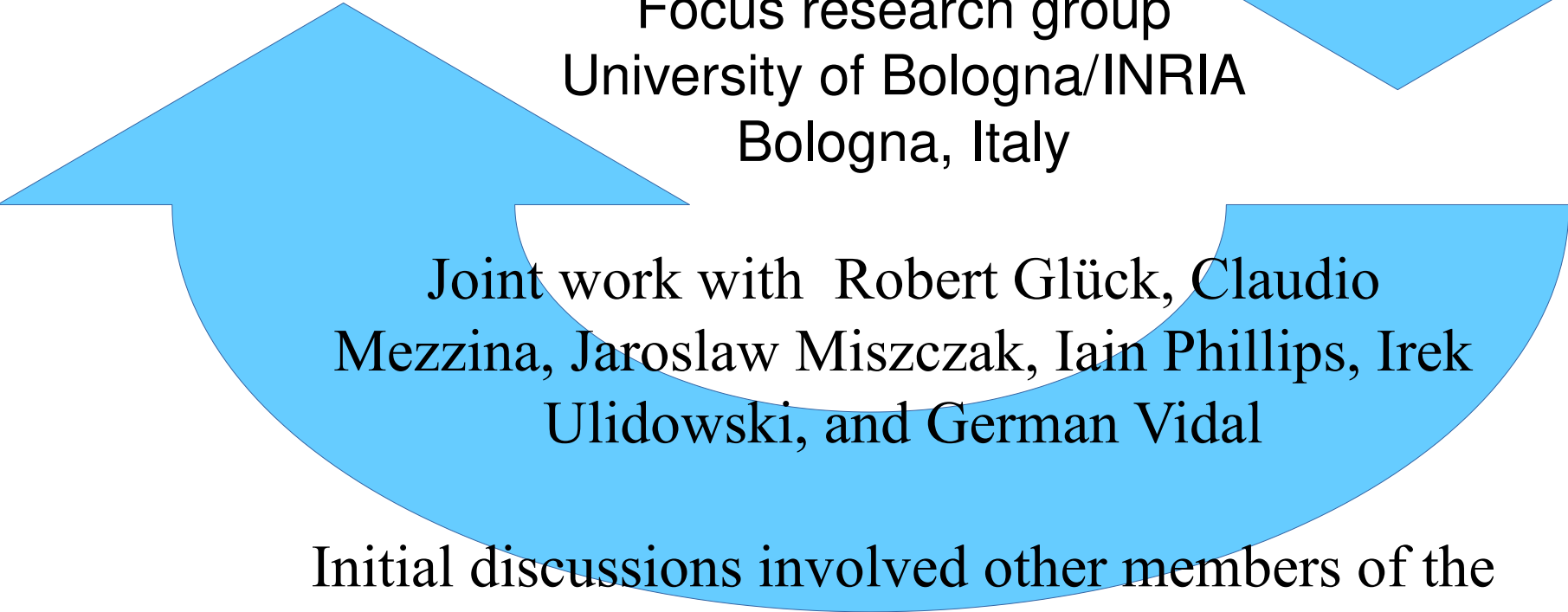




# **Towards a Taxonomy for Reversible Computation Approaches**

Ivan Lanese  
Focus research group  
University of Bologna/INRIA  
Bologna, Italy



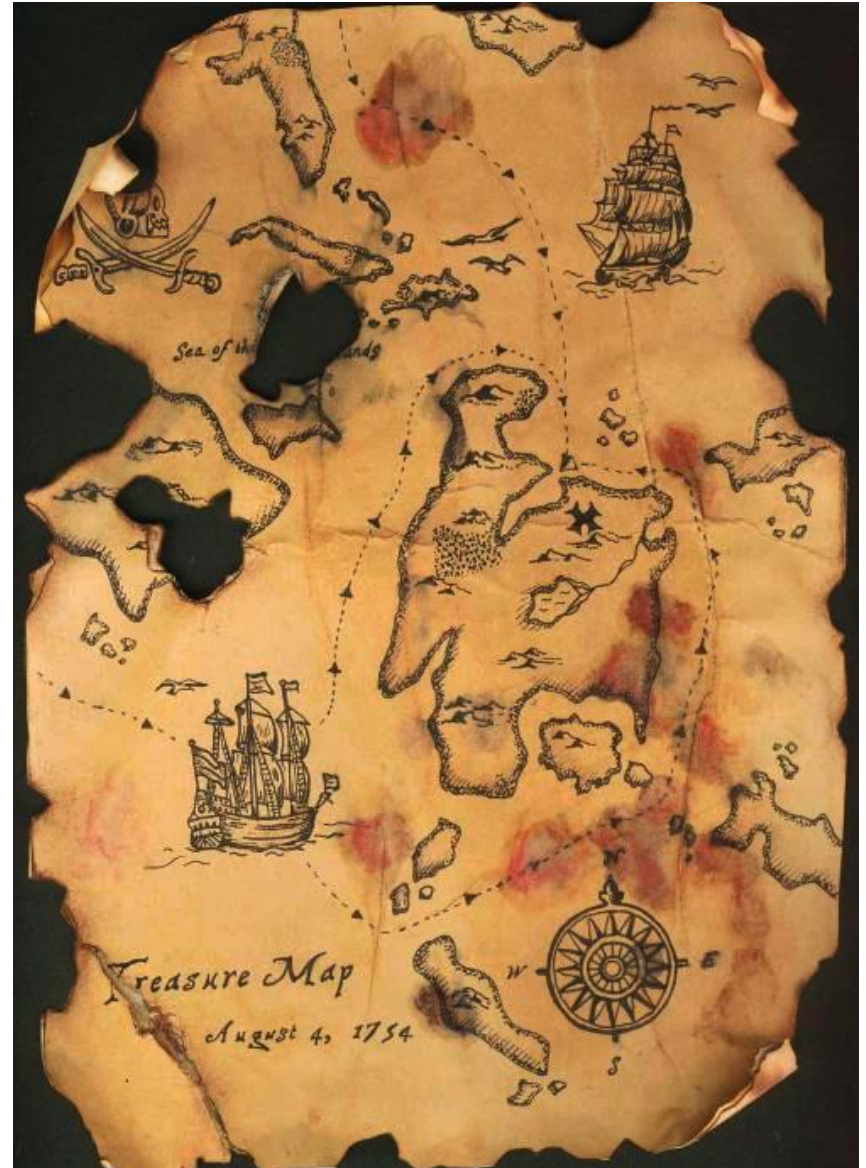
Joint work with Robert Glück, Claudio Mezzina, Jaroslaw Miszczak, Iain Phillips, Irek Ulidowski, and German Vidal

Initial discussions involved other members of the  
COST Action IC1405

# Roadmap

---

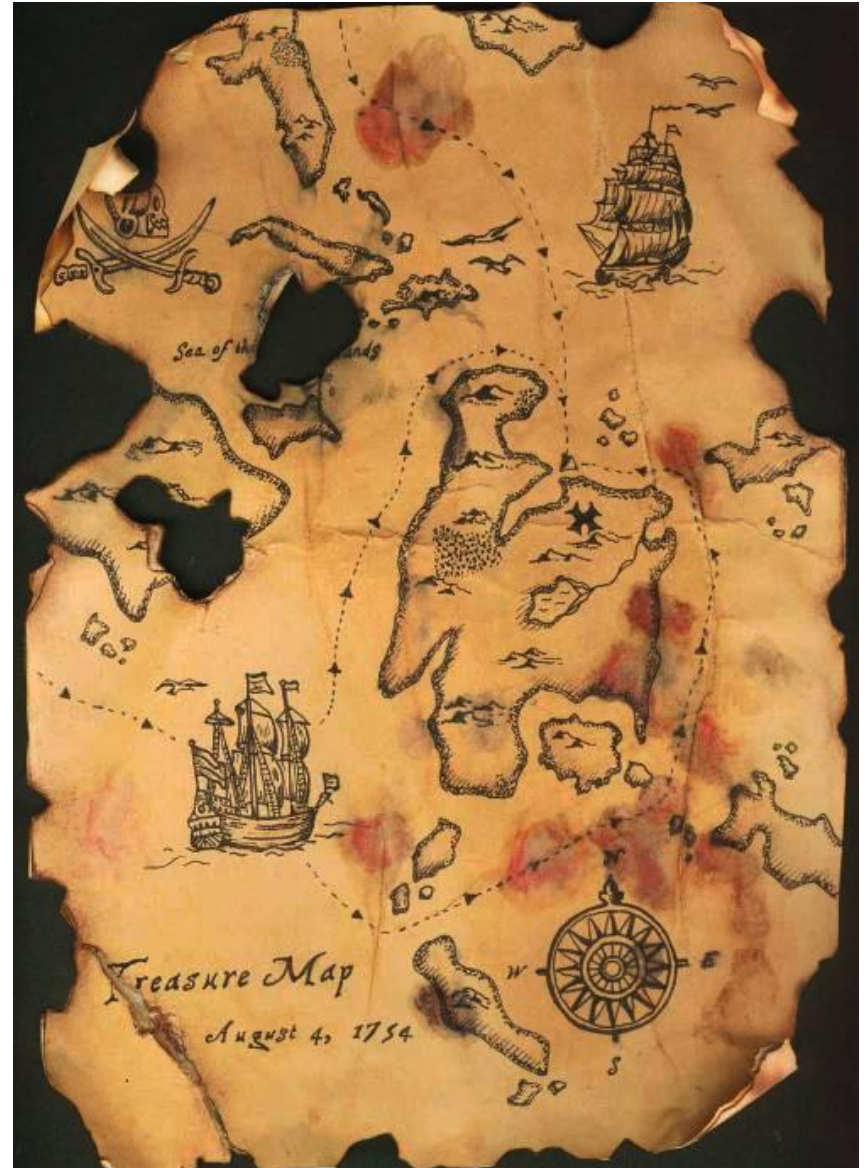
- Why a taxonomy?
- The six dimensions
- Some examples
- Conclusion



# Roadmap

---

- Why a taxonomy?
- The six dimensions
- Some examples
- Conclusion



# Reversible debugging zoo

---

- Reversible computing has been used in a plethora of settings, including hardware circuits, programming languages, formal models, algorithms, ...
- Reversible computing targets a plethora of application areas: low-power computing, debugging, robotics, simulation, ...
- While the different definitions share some aspects, they are not identical

# How to put some order?

---



- Which are the commonalities and differences between the existing approaches?
- Which are the ones closer to each other?
- Could we transfer concepts and techniques?
  
- We propose a taxonomy of the different approaches, as a first step towards answering these questions
- Only a preliminary proposal, with no aim of being the final word on the topic, neither of completeness

# Taxonomy structure

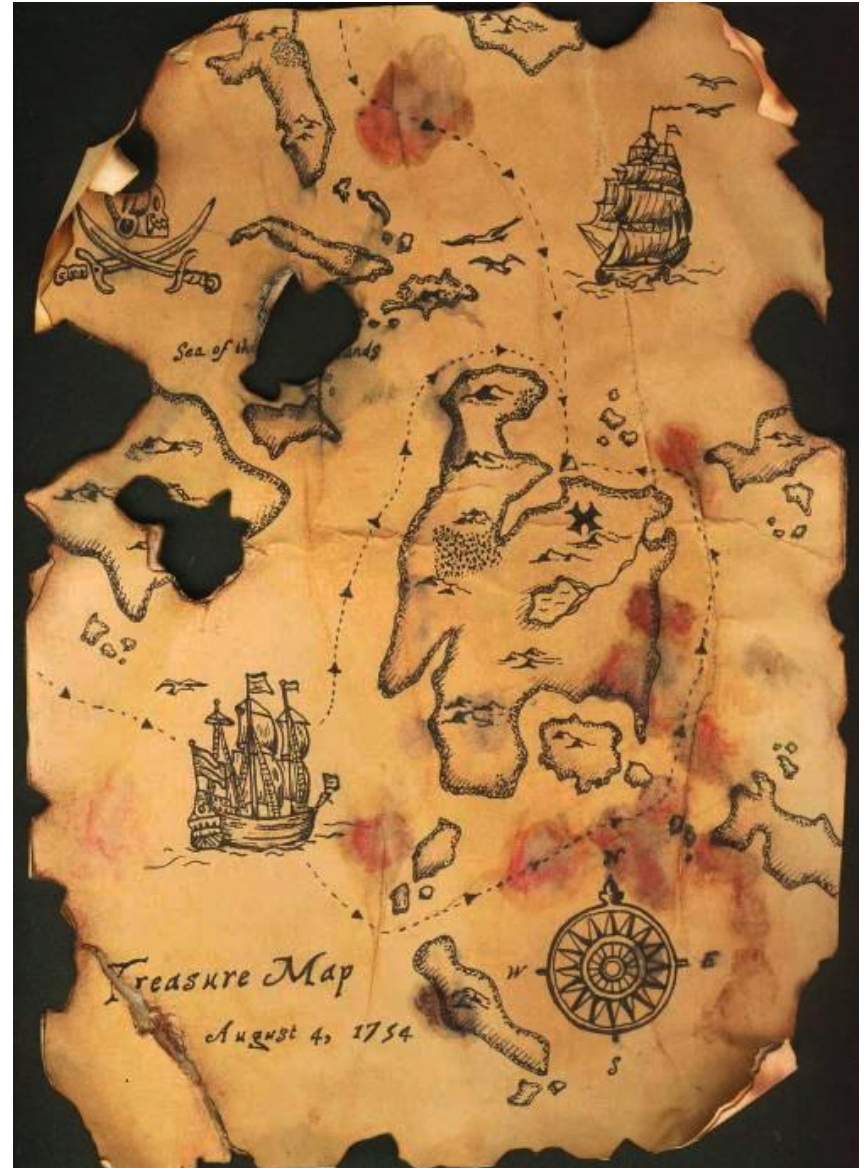
---

- We consider six dimensions, and for each dimension different positions
- In many dimensions, positions can be seen as ordered, from more specific to more general
- The dimensions aim at capturing features of reversibility, abstracting away from:
  - the underlying model
  - the target application area

# Roadmap

---

- Why a taxonomy?
- The six dimensions
- Some examples
- Conclusion



# Dimension **FOC**: reversibility focus

---

- Functional behavior: a system is reversible if it computes injective functions (**Janus**, circuits, Turing machines, ...)
- Reachable states: a system is reversible if it can go back to past states (checkpointing, SVN, ...)
- Undoing steps: a system is reversible if it can undo steps (reversible calculi, reversible Erlang, **Janus**, ...)



# Dimension **RES**: resources for reversibility

---

- None: the model is naturally reversible, and has no need of additional resources (Janus, circuits, Turing machines, ...)
  - One can compute backwards without computing forwards first
- Inside the model: resources are needed, and are in the same formalism as the original system (Petri nets, ...)
- Outside the model: resources are needed, and one needs to extend the model to represent them (reversible calculi, reversible Erlang, ...)
- Ordered, the position depends on the abstraction level

# Dimension **WHE**: when reversibility is enabled

---

- Always: any action can be undone (Janus, RCCS, ...)
- Sometimes: some actions can be undone, others cannot (RCCS with irreversible actions, robotics, ...)
- Ordered

# Dimension **ORD**: order of undoing

---

- This dimension applies to models where there is a notion of undoing (cf. dimension FOC)
- Reverse order: only the last action can be undone (Janus, Turing machines, ...)
  - These models are backward deterministic
- Causal order: any action can be undone provided that its consequences have been undone first (most calculi and languages for concurrency, ...)
- Out of causal order: there is no constraint on which action can be undone (models for biology, ...)
- Ordered

# Dimension **STR**: state reachability

---

- This dimension applies to models where there is a notion of state (cf. dimension FOC)
- Only past states: only states in the past of the system (Janus, Turing machines, ...)
- Only past states up-to concurrency: states that could have been reached by swapping the order of concurrent actions (most calculi and languages for concurrency, ...)
- Forward reachable states: states that are reachable by going forward from the initial state (some Petri nets, ...)
- Also states not forward reachable (models for biology, ...)
- Ordered, roughly correspond to dimension ORD

# Dimension **PRE**: preciseness of reversibility

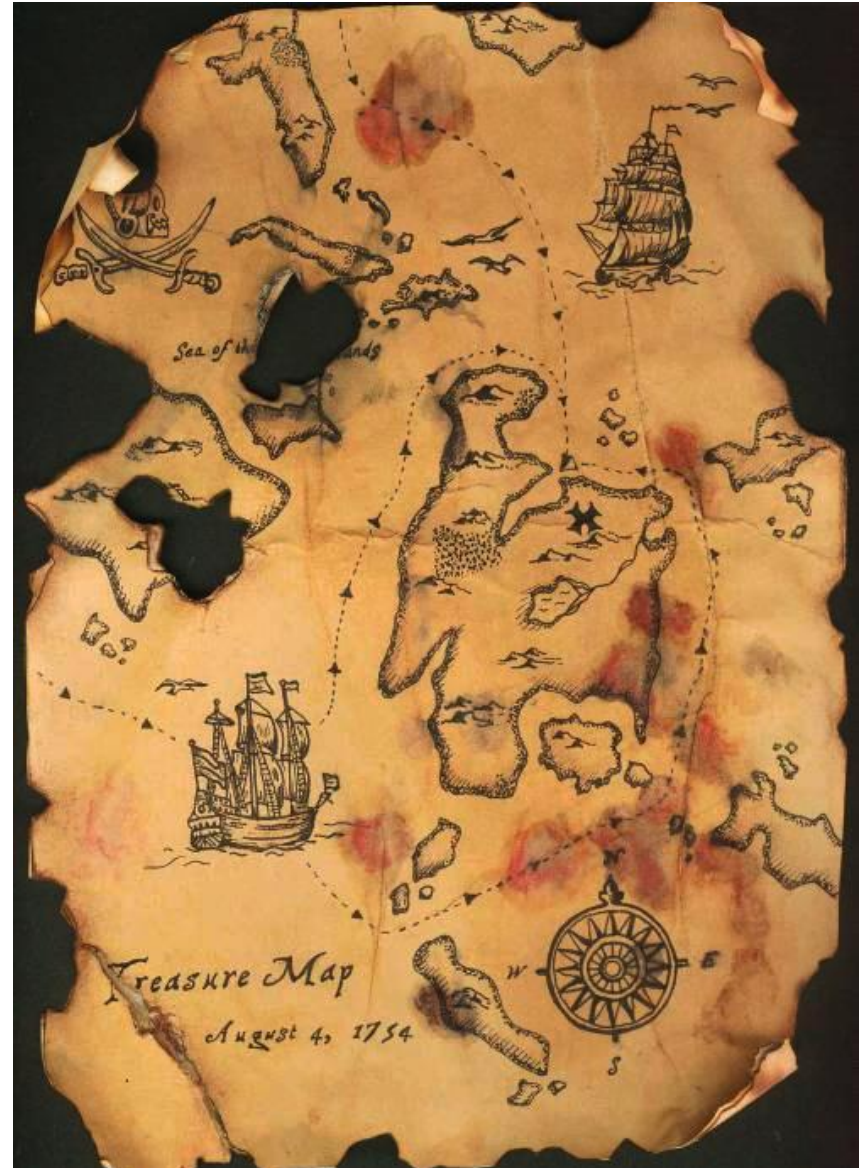
---

- Precise: classical reversibility (Janus, Turing machines, ...)
  - Captured by the Loop Lemma in concurrency
- With additional information: when going backwards one keeps information on the undone actions (local search with backtracking, ...)
- Approximate: one goes back to a state close to the original one (transactions with compensations, robotics, ...)
- Ordered

# Roadmap

---

- Why a taxonomy?
- The six dimensions
- Some examples
- Conclusion



# Janus (and traditional reversible models)

---

- Reversible and quantum circuits fit here as well
- Focus on injective functional behavior and possibility of undoing actions (by computing backwards)
- Naturally reversible, no additional resources needed
  - Obtained by restricting classical models
- All their components are reversible
- Backward execution in reverse order
- Only past states are reachable (but one can go before the beginning of the computation)
- Reversibility is precise

# RCCS (and formalisms for concurrency)

---

- Focus on possibility of executing back and forward
- Memory to remember past interactions and causality
  - Outside the model: RCCS is not CCS, and it is not even clear whether an encoding is possible
- All their components are reversible
- Backward execution in causal order
- Only past states up-to concurrency are reachable
- Reversibility is precise
- This approach takes the name of causal-consistent reversibility



# Petri nets

---

- Different works take different approaches to reversibility
- Some focus on possibility of executing back and forward, others on state reachability
- Some use additional places as memories or add reverse transitions, others do not
- All their components are reversible
- All possible orders have been considered
- As a result, different approaches allow to reach different states
- Reversibility is precise

# Sagas (and transactions with compensations)

---

- Actions are undone by executing ad-hoc compensations
- Focus on possibility of executing back and forward
- Information on the past is kept inside the model
- Sagas allow for irreversible actions
- Backward execution in causal order
- Compensations can bring to states not forward reachable
- Reversibility is approximate
  - Ideally compensations are approximate undos, even if there is no precise characterization of this

# SVN and GIT

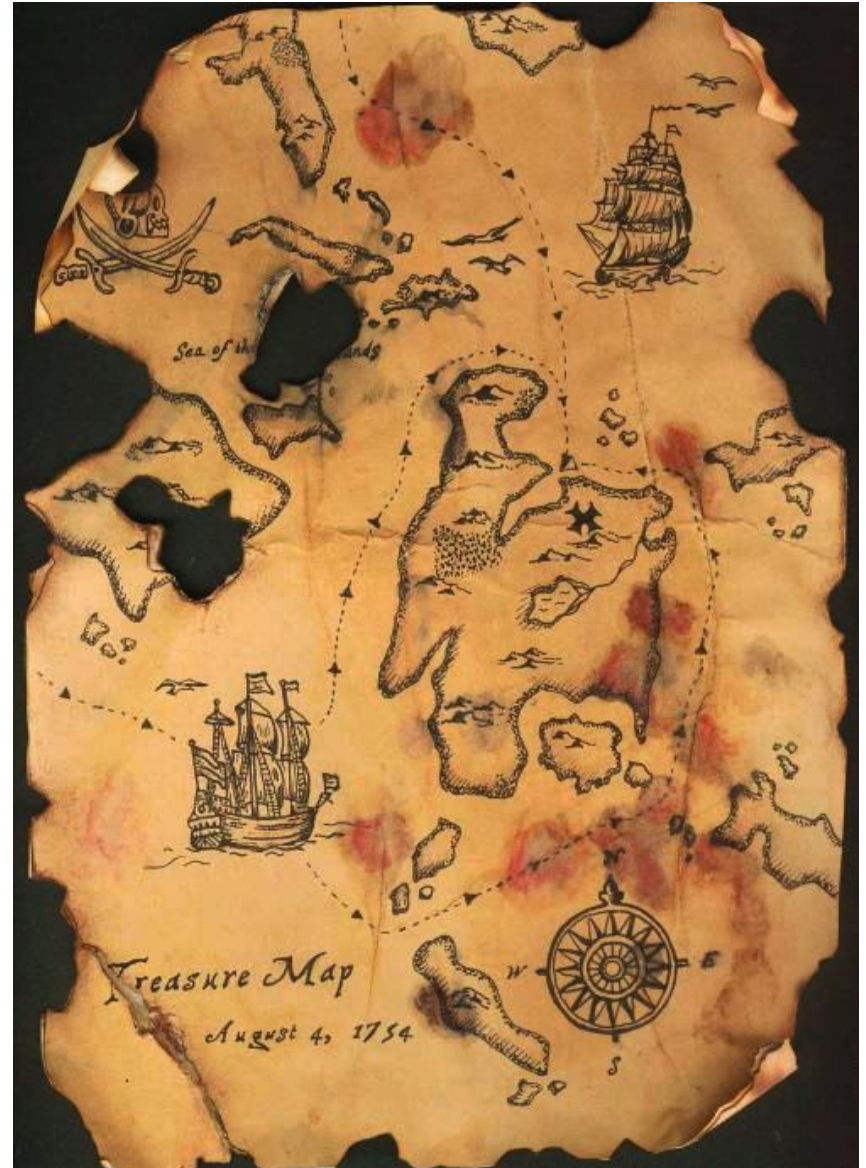
---

- Focus on possibility of recovering past states
- Information on the past is kept inside the model
- All (saved) past states are reachable
- Order of undoing is not meaningful, since there is no notion of undoing
- Only past states can be reached
  - ... but merges are possible in case of contrasting futures
- Reversibility is precise

# Roadmap

---

- Why a taxonomy?
- The six dimensions
- Some examples
- Conclusion



# Summary

---

- First taxonomy of approaches to reversibility
- Mainly tailored to formal models and languages, due to the expertise of the authors
- We tried to stress-test the taxonomy by considering approaches outside classical reversibility (like Sagas)
- Some positions ensure some properties
  - Loop Lemma, backward determinism

# Future directions

---



- Refining the taxonomy
- Applying it to other models
- Is it possible to give an axiomatic characterization of (part of) the taxonomy?
- Can we use Petri nets as a single setting to contrast different approaches?
  - Some works already consider a few possibilities

Finally

---

**Thanks!**

**Questions?**