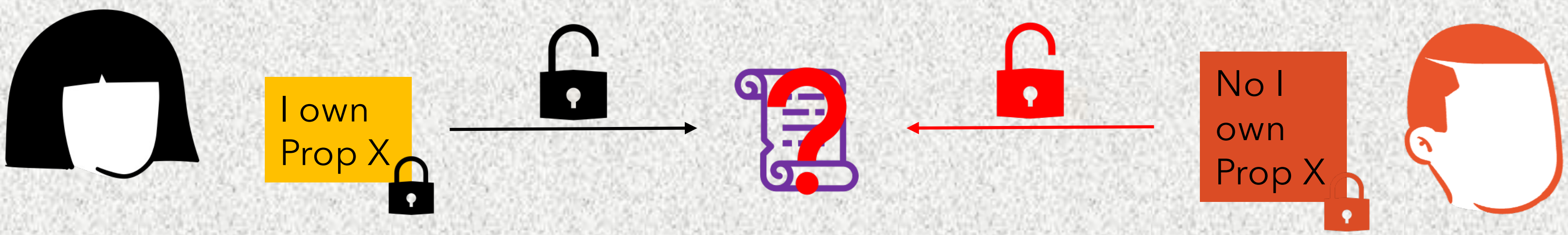
Uncloneable Cryptography

A tale of two paradigms

Ghada Almashaqbeh (UConn), Rohit Chatterjee (Stony Brook)

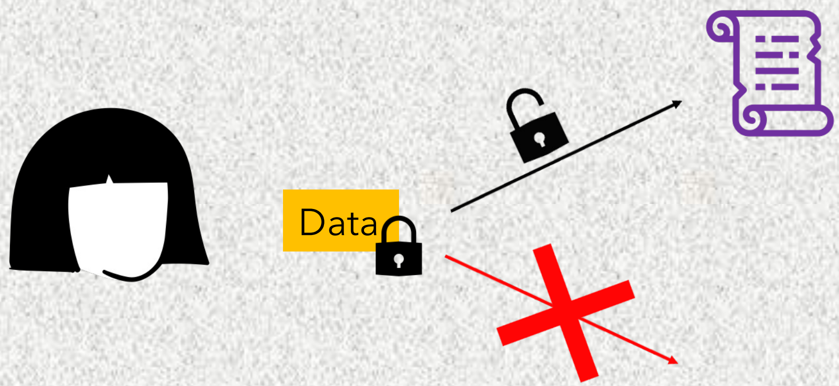
What is Uncloneable Crypto?



- **Secrecy/ Authenticity is not always sufficient**
- **Multiplicity of authorized sources is the problem**

GOAL: Control ability of users to 'copy' info!

Need for Uncloneability



Turns out to be a natural and useful guarantee

- Watermarking type applications
- Associates naturally with minting of digital currencies!
- Very close to what NFTs set out to do

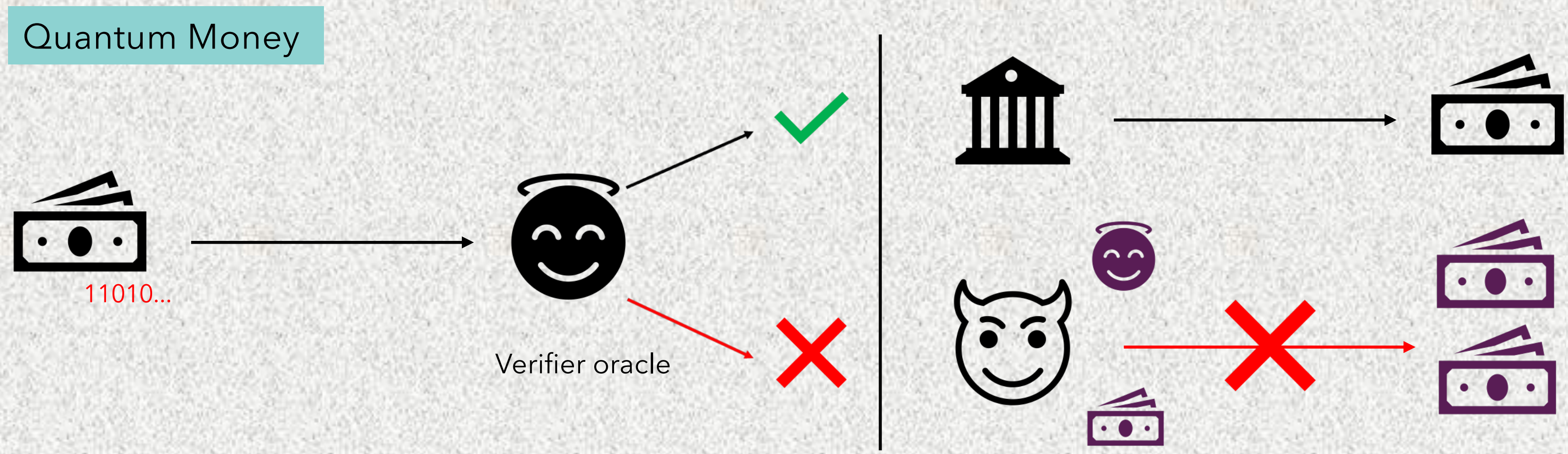
Overview

- Two major themes:
  - Quantum state-based constructions
  - Polymer-based constructions

- Our contributions:
  - Classification of Uncloneable Primitives
  - Comparison and identifying properties unique to either setting
  - New constructions in the polymer setting
  - Directions for Future Work

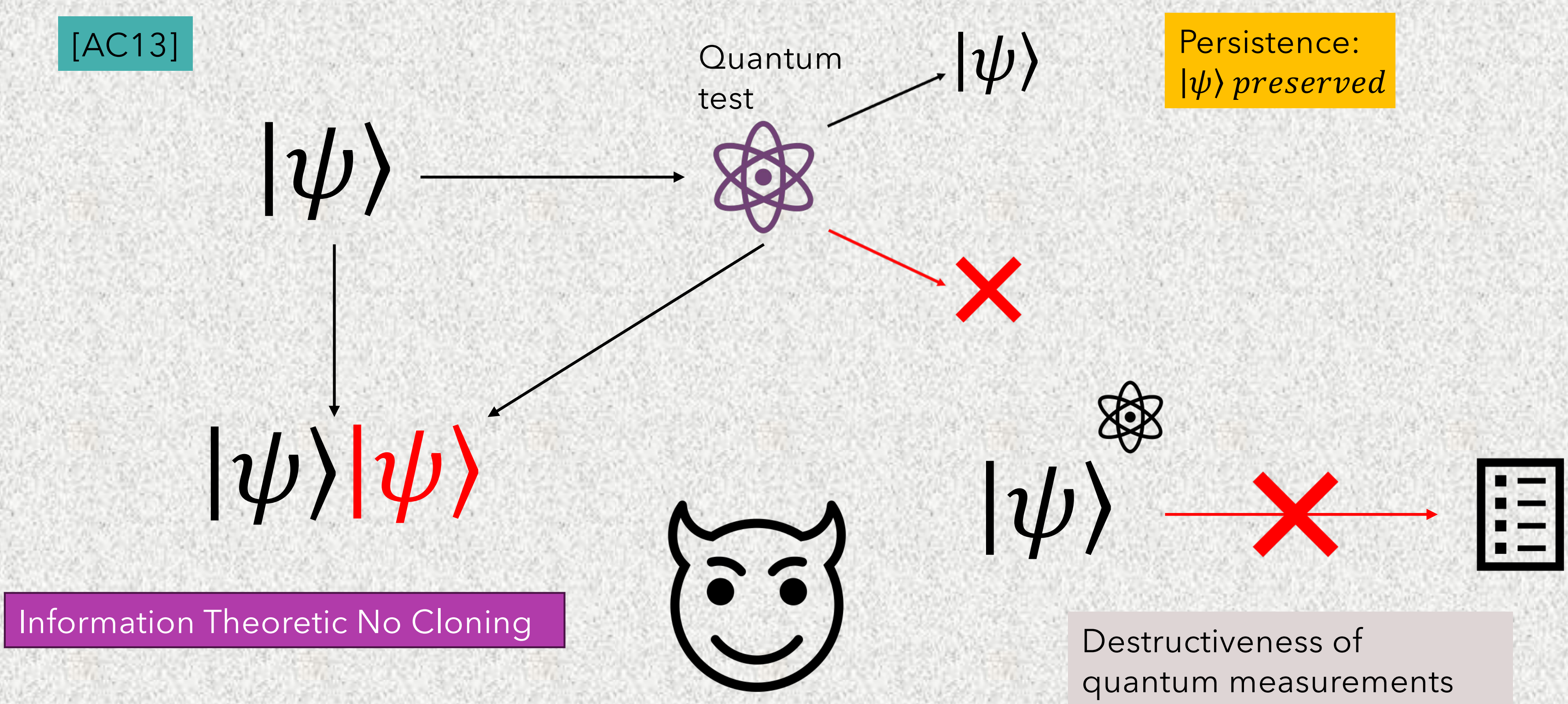

Models for Uncloneable Crypto

Uncloneability from Quantum States



- Money states verifiable by a (publicly accessible) interface
- Only bank mints currency
- Cannot create new money from existing notes

How Quantum Money Works

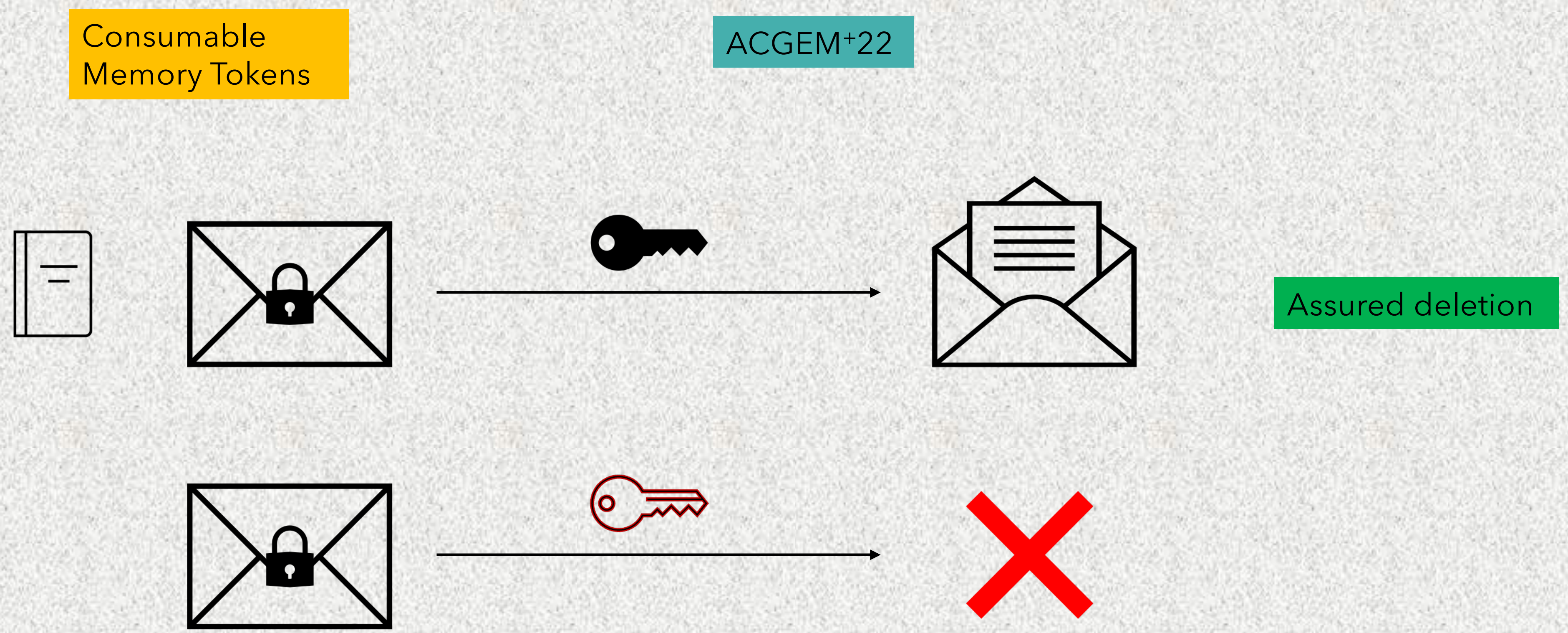


Uncloneable Crypto from Quantum States

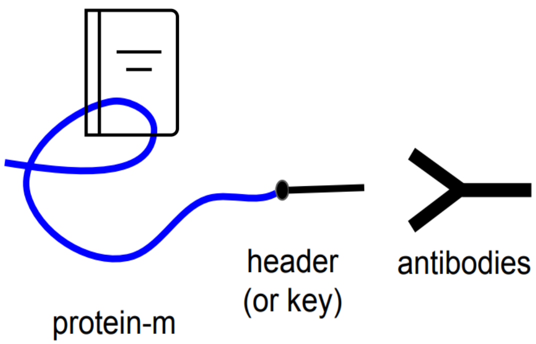
- Quantum Money
- One-Shot Signatures/ Tokenized Signatures
- Uncloneable Encryption
- Secure Software Leasing
- Copy Protected Programs

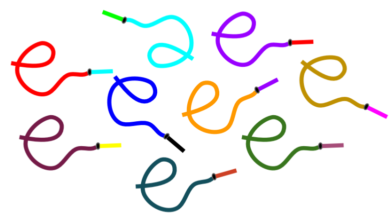
- Typically, we need (alongside standard crypto/QROM etc):
  - Information-theoretic No-Cloning theorem
  - Post - Quantum Indistinguishability obfuscation

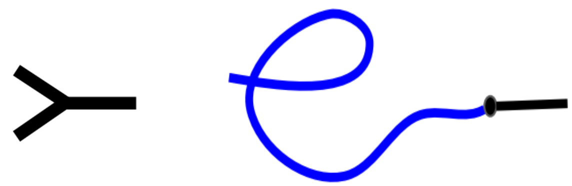
Uncloneability from Polymer Constructs



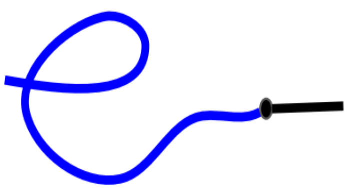
How Memory Tokens work (roughly)



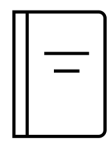




Key – matching errors



(1,n): Adversary power gap



- Data unrecoverable without correct key!
- Data is destroyed in read attempts
- Protein sample *cannot be cloned* (Central Dogma of molecular biology)

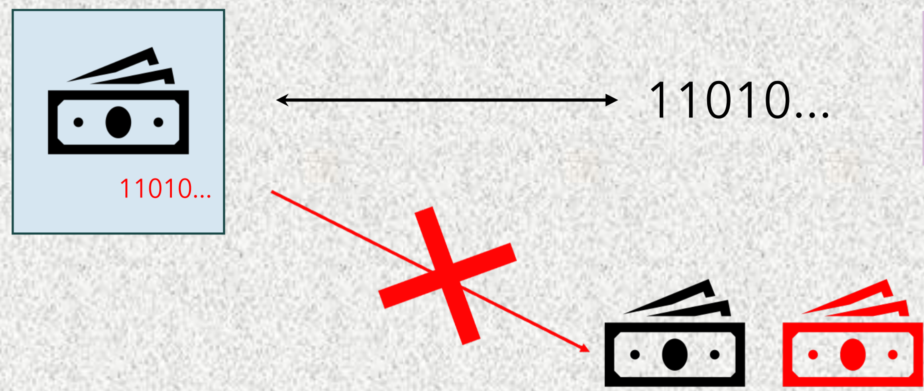
Uncloneable Crypto from Polymers

- Consumable Memory Tokens
- Digital Lockers
- Bounded Execution/ k-time Programs

- Typically, we need (alongside standard crypto/QROM etc):
  - Hardness of Protein Reading
  - Impossibility of cloning proteins (Central Dogma)
  - Indistinguishability Obfuscation

Classification and Comparison

Tier 1: Uncloneable Entities



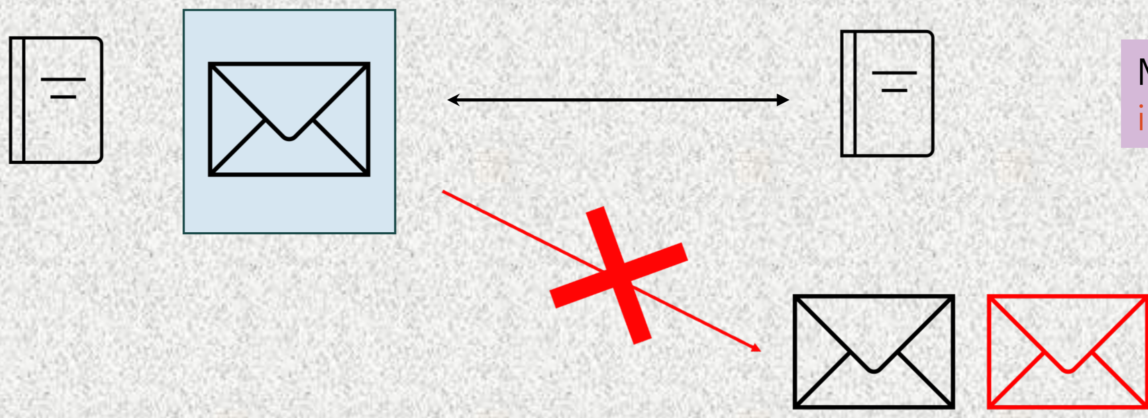
Metadata is procedure induced and not in explicit control of generator (e.g. generation randomness)

Can only really use for verification

Primitives:

- Quantum Money
- Signature Tokens

Tier 2: Uncloneable Data



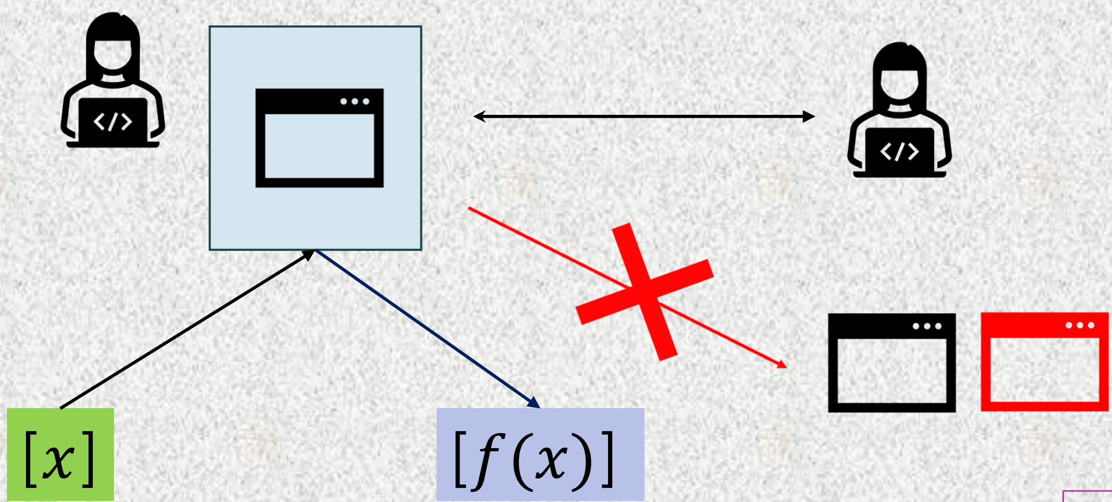
Message is explicitly included in the encoding

Can only recover message by revoking uncloneability

Primitives:

- Uncloneable Encryption
- Digital Lockers

Tier 3: Uncloneable Programs



Program is explicitly defined in the encoding. Inputs need to be appropriately encoded as well.

Typically requires some sort of obfuscation

Primitives:

- Secure Software Leasing
- Copy – Protected Programs

Setting	**Paradigm**	**Existing Primitives**	**Additional Assumptions**
Quantum	Unclonable states	Quantum money	qOWF, $qi\mathcal{O}$, qLWE
	Unclonable programs	Copy protection	$qi\mathcal{O}$, qOWF, qLWE
	Unclonable programs	Secure software leasing	CRS, $qi\mathcal{O}$, qLWE
	Unclonable states	One-shot signatures	$qi\mathcal{O}$, any secure classic signature scheme
	Unclonable data	Unclonable encryption	qOWF, $qi\mathcal{O}$, qROM
	Unclonable programs	Unclonable decryption	$qi\mathcal{O}$, qOWF, qLWE
Polymers	Unclonable data	Digital lockers	ROM
	Unclonable programs	$(1, n)$-time programs	OWF, $i\mathcal{O}$

Contrasting the two paradigms

Quantum Model

- Persistence → Reusable constructions
- Typically requires oracles
- Requirement: Quantum Computers/ Networks

Polymer Model

- Guaranteed destruction → Bounded # of execs
- Uncloneability is direct
- Requirements: (Ongoing) Biochemical techniques, physical devices

Comparing the two paradigms

- Protein → Quantum: Difficult to get Guaranteed Deletion
- (Lower bounds: Bdd Exec Programs [even w/ power gap] need hardware assumptions even w/ quantum computing)
- Quantum → Protein: Possible, but with caveats: based around (limited) Bdd exec programs.
- Need to account for adversary power gap (1 vs n tries).
- Persistent applications (e.g., copy protection) are also not yet achievable through proteins.

Primitive to realize	**Using k-time programs?**	**Using $(1, n)$-time programs?**
Quantum money	Yes (with $k = 1$)	No—a coin can be spent n times
Software copy protection (and secure software leasing)	Yes (including learnable functions)—but a program can be executed only k times	Yes—but permitting domain splitting attacks and the power gap between the honest party and the adversary
One-shot signatures	Yes (with $k = 1$)	No—an attacker can sign up to n messages instead of one
Unclonable encryption	Yes	Yes—but a weaker security notion covering $n + 1$ attackers instead of two
Unclonable decryption	Yes	Yes—same constraint as above
Digital lockers	Yes—k trials for honest party	Yes

Directions for Future Work

- Q1: Strengthening the polymer-based model.
- Caveat: realizes very strong primitives like non-interactive oblivious transfer.
- Q2: Combining both approaches in a 'Hybrid Model'.
- Are there stronger primitives we can get from combining both kinds of assumptions?
- No obvious obstacles or caveats to doing this.
- Both approaches are speculative, requires further work.

Thank You!

Eprint: 2023/702