14.129
Spring, 2025
Lecture 5

Tokenized and Programmable Assets

Dynamic Ledgers, Platforms, Atomic Trade and Settlement, Legacy Systems and Trade Fails

Robert M. Townsend

Elizabeth & James Killian Professor of Economics, MIT

Massachusetts Institute of Technology

Outline

- Asset exchanges programmed in advance, dynamic ledgers
- Legacy systems and trade fails
- Problems nevertheless with programmed assets
- Further solutions
- Interoperability
- Tokenization
- Exchange and contract platforms
- Central bank innovations

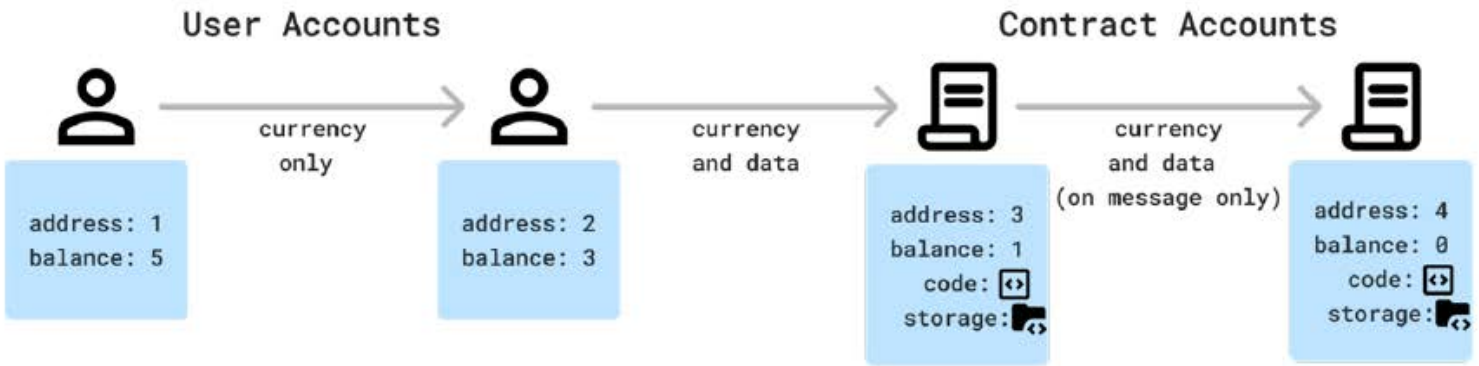
Asset Exchanges Programmed in Advance

- Dynamic Ledger
  - Ownership and transfers over time and states of the world
- Example: Trade agreement for asset lending
- Atomic Swaps
  - Trade and settlement instantaneous despite separation in time

Ethereum as Smart Contract

Ethereum Structure

- Two types of accounts, externally owned (user) accounts and contract accounts
  - externally owned accounts have an address and balance
  - contract accounts also contain code and data storage
- Contract accounts receive transactions and update their state and send other transactions
  - can only initiate transactions as a response to another transaction
  - using computational resources to update state incurs "gas" fees



Transaction Structure

Sender	Recipient	Currency	Data	Start Gas
User address	User or contract address	Amount of Ether to transfer	key, value pairs of data to send to contract (can be empty)	extra currency to fund computation (can be empty)
0×1	0×2	3 eth	key1: value1 key2: value2 key3: value3	0.0001 eth

- All transactions have a sender, recipient, and currency field (can be 0)

- Transactions can **optionally** have data and start gas to allow message passing and computation

EVM

- Ethereum Smart Contracts compute on the Ethereum Virtual Machine
- Virtual Machines have a CPU, memory, disk storage, and a network interface
- They perform tasks identical to a physical machine, but allows administrators to change the resources of the machine based on workload intensity
  - Allows a single server to operate many separate machines, each with their own resources
- The EVM is a Turing-complete software that runs on the Ethereum network, allowing anyone to execute code in a trustless ecosystem
- When Smart Contracts are executed on the EVM, every node executes the contract code using their EVM instance
  - This ensures transaction outcomes are consistent across the network
  - The virtual machine is isolated from the rest of the physical machine, so execution failures cannot harm the host computer
- The EVM has its own native cryptocurrency, Ether, which is used to pay for transaction fees and computational services

EVM, cont.

- Smart contracts are self-executing contracts with the terms of the agreement directly written into code
- They run on the EVM and automatically enforce and execute the terms of a contract when predetermined conditions are met
- Contracts can be reached using special transactions called messages.
  - A message call can transfer Ether, change the state of data within a contract, or create a new contract
- Smart Contracts are written in a high level, human-readable language (Solidity), then compiled down to bytecode, an intermediary code that bridges the gap between the high-level source code and low-level machine code
- The EVM turns each bytecode instruction into binary bits, 1s and 0s, which the (EVM) virtual processor can read and execute

Lee, Martin and Townsend (2024) “Optimal Design of Tokenized Markets”

Resolving Settlement Uncertainty in Token System

- DLT-based settlement system as technological solution, **token system**
- Tokenize financial assets on DLT
- Programmability of assets
  - Progammability enables traders to commit to settlement
  - "Collapse" between trade and settlement

$\therefore$ **Eradicate** potential for fails

Model

- Three risk-neutral traders $i = \{A, B, C\}$
- Single long-lived asset, initially owned by A
- Traders have period-dependent payoffs for holding asset

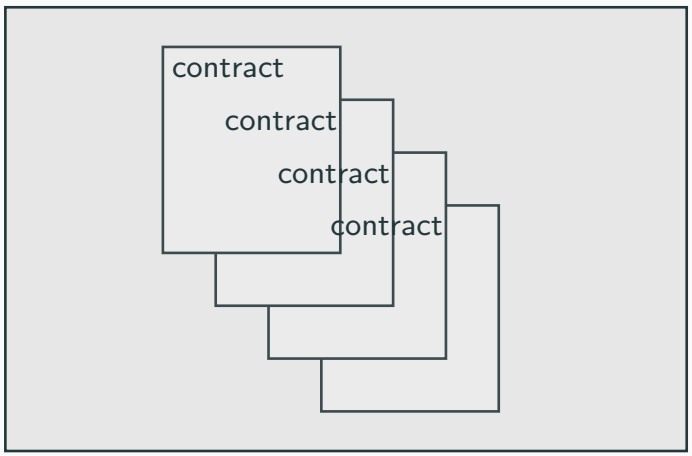
Model Timeline

Two stages

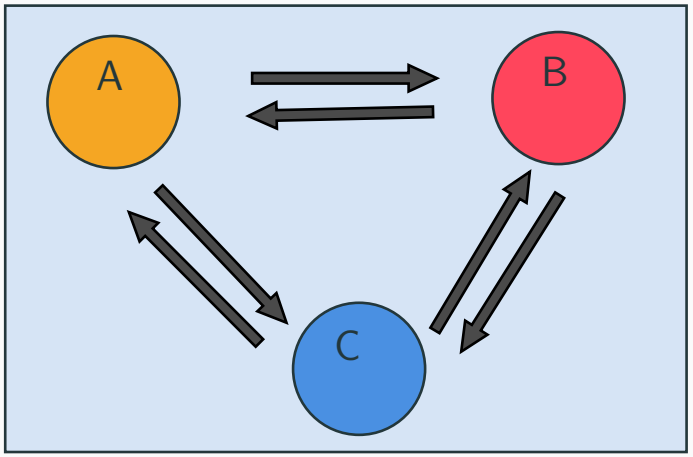
- **Trading stage.** $t = m1, m2$
  Two sequential bilateral meetings between traders

- **Settlement stage.** $t = 1, 2, 3$
  Actual transfer of asset between traders

Trading Stage
$t = m1, m2$



Settlement Stage
$t = 1, 2, 3$



Trading Stage $t = m1, m2$

- Two meetings sequentially occur at $t = m1$ and $t = m2$

[A meets with B] [B meets with C]

- Order of meetings random, **known only to** B
  - A and C **never** meet
  - B acts as intermediary between A and C

Settlement Stage $t = 1, 2, 3$

- Up to two trades successfully occur in trading stage
- Trades specify "transfer" of asset between A, B, and C over $t = 1, 2, 3$

B borrows asset from A in period $t = 1$ for price P

  - A transfers asset to B at the beginning of $t = 1$
  - B transfers asset to A at the end of $t = 3$

- In the settlement stage, settlement actions take place

Settlement in Token System

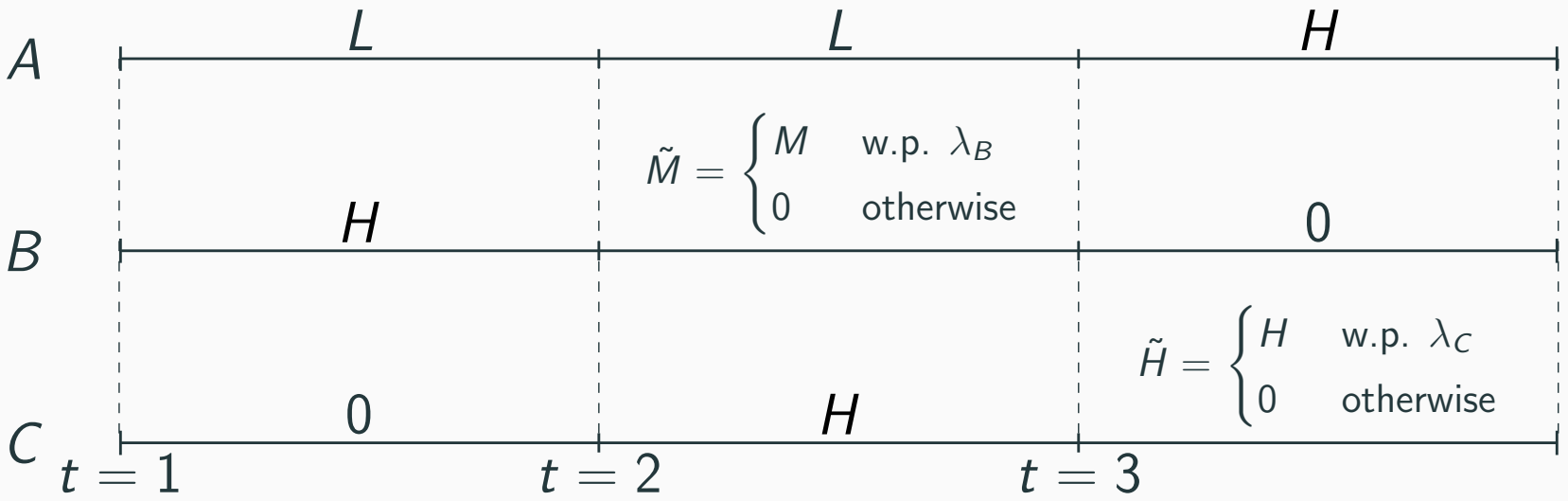
Bargaining and programming occurs simultaneously, as if "immediately settled"

- Asset "programmed" during meetings in trading stage
- Transfer instructions are self-executing
  - Assets are transferred without any further trader action
  - No trader able to prevent programmed transfer from occurring
- **Requires** trader to be current holder of asset at the time contract specifies it to be transferred

Implication: Both parties know that conditions of the trade are satisfied, e.g. A must own the asset to lend the asset, **and** B **knows**

Traders' Payoffs

- Payoff from holding asset differ
- $H > M > L > 0$
- B and C privately learn $\tilde{M}$ and $\tilde{H}$ after trading stage



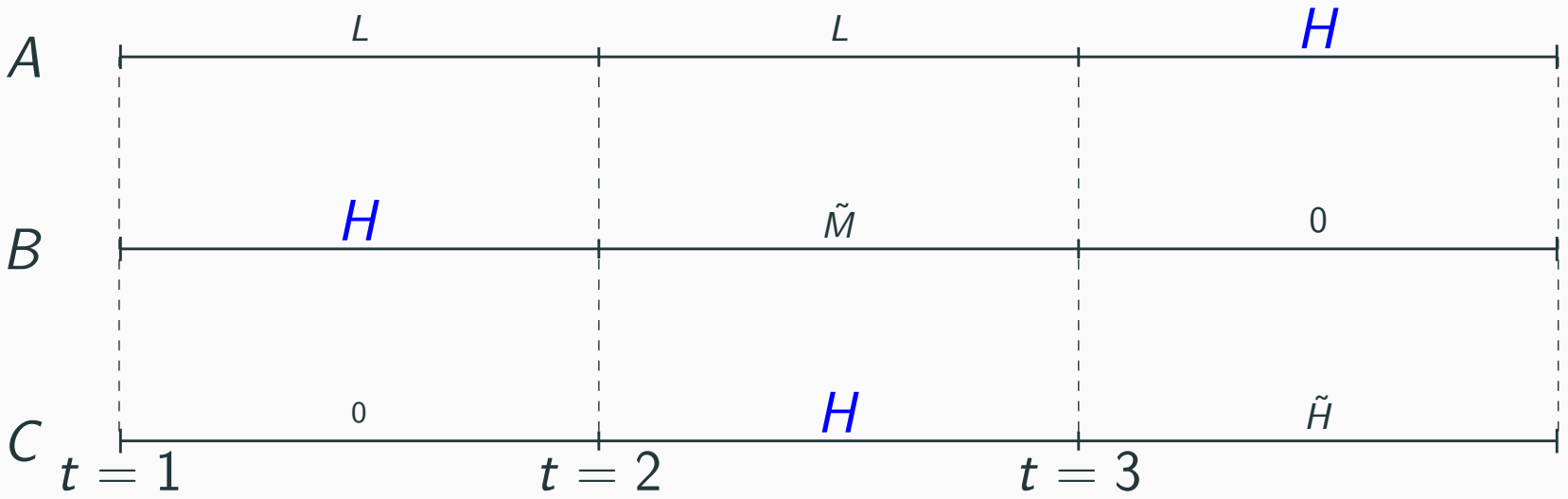
Trader's Payoffs

- Maximum payoff from holding asset:

$$A \text{ in } t = 3$$

$$B \text{ in } t = 1$$

$$C \text{ in } t = 2$$



Legacy System and Trade Fails

- How the current financial system operates
- The extent of fails
- Causes including cascades

Without Commitment

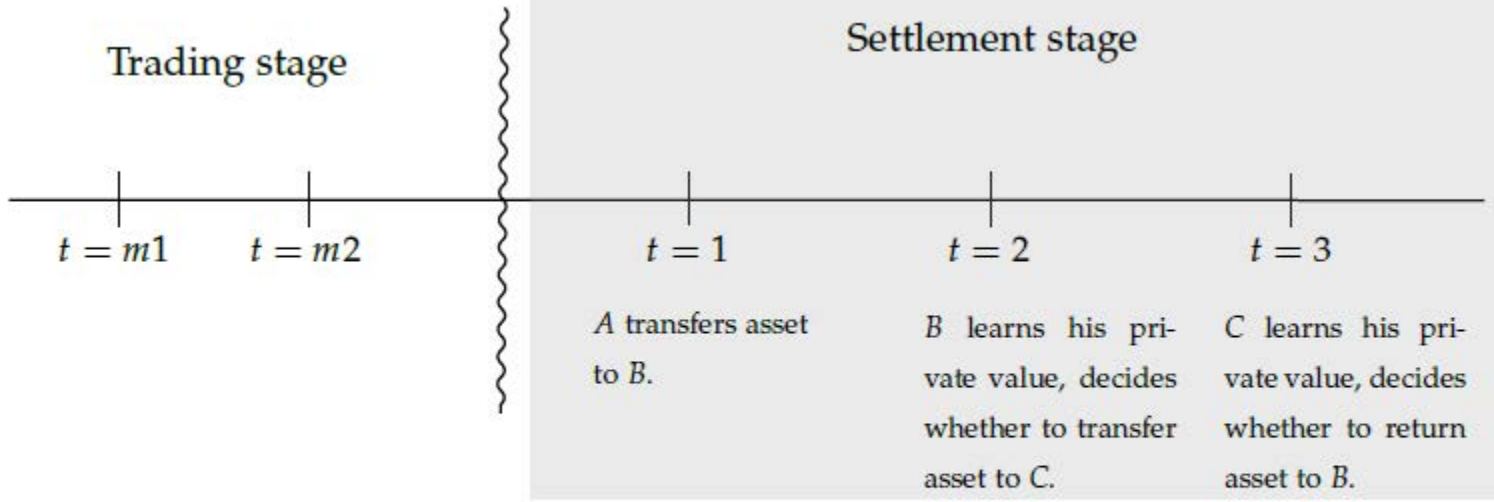


Figure 3: **Settlement in Legacy System.** This figure highlights key decisions that take place in the settlement stage conditional on $\mathcal{C}_{AB}^{13}$ and $\mathcal{C}_{BC}^{23}$. At $t = 3$, $\tilde{H}$ is realized, and C decides on whether to return the asset to B (which is then returned to A). At $t = 2$, $\tilde{M}$ is realized, and B decides on whether to transfer the asset to C.

Lee, Martin and Townsend (2024)

Paths With and Without Fails

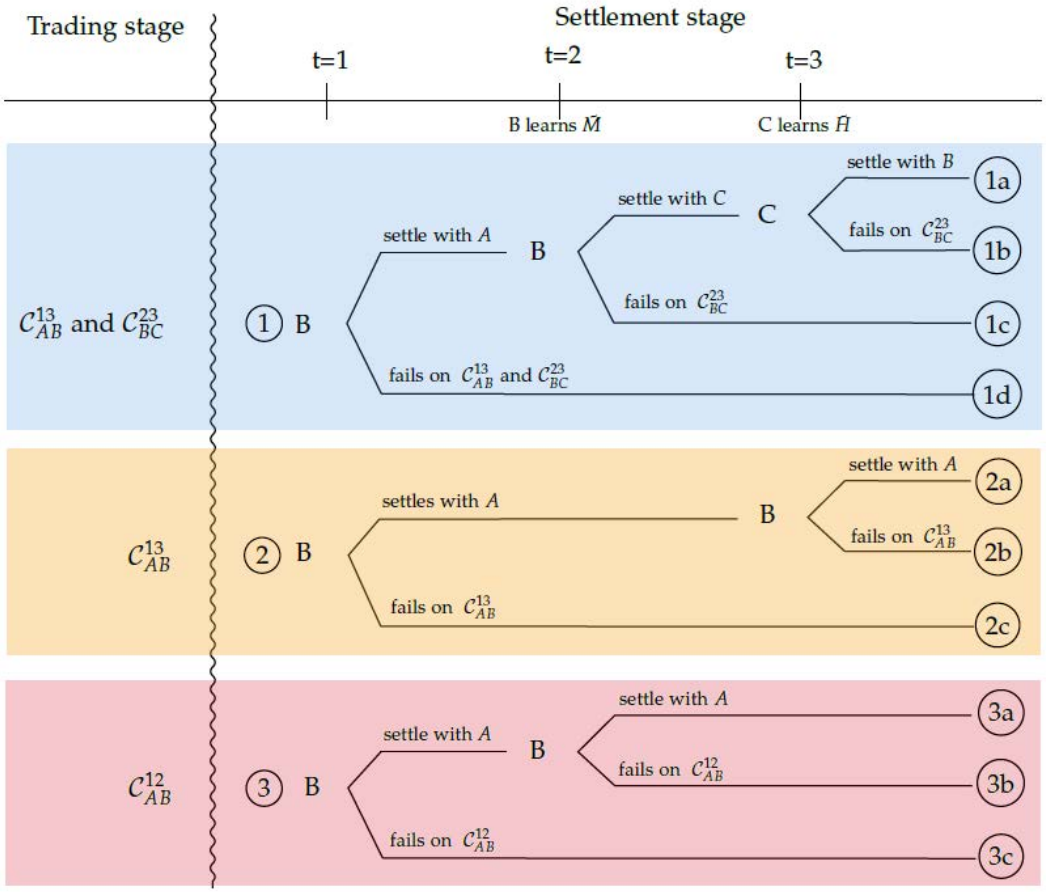


Figure 4: **Contracts and settlement actions under legacy system.** This figure summarizes the key settlement actions that arise at $t = 1, 2, 3$ given a set of contracts, specified on the left. B and C privately learn their $t = 2$ and $t = 3$ payoffs $\tilde{M}$ and $\tilde{H}$ in the beginning of $t = 2$ and $t = 3$, respectively. Private values factor into their settlement strategies, where $\tilde{M} = M$ with probability λ_M, and 0 otherwise; and $\tilde{H} = H$ with probability λ_H and 0 otherwise. The terminal payoffs for A, B, and C at the end of $t = 3$ is provided in Figure 5.

Lee, Martin and Townsend (2024)

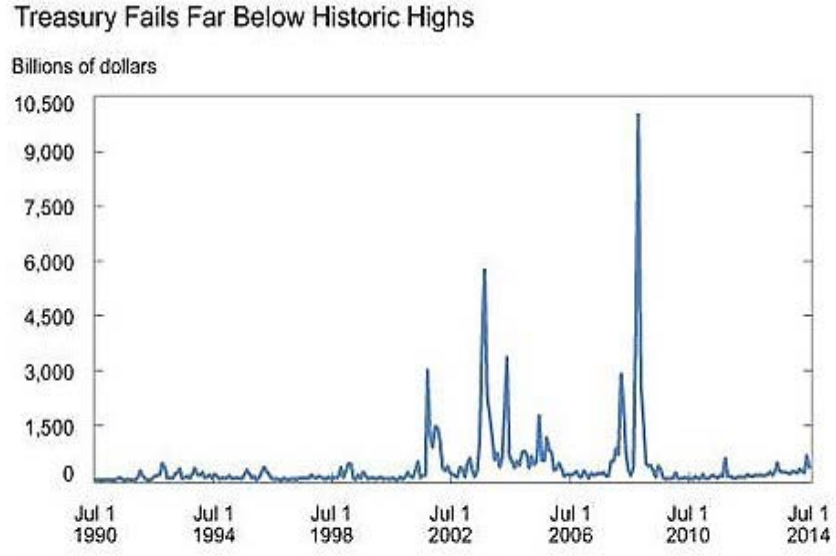
Legacy Systems and Trade Fails: In Practice

- Settlement fails in modern financial markets
  - Settlement fails still occur
    - At peak of GFC, $400 billion per day in settlement fails in Treasury market
    - A fails charge introduced to reduce this
    - Settlement fails still occur!
- Settlement risk in legacy system
  - Persistence of settlement fails
    - Institutional and technological feature of (current) legacy system
    - Individual traders to submit instructions corresponding to contractual obligations arising from trading activity
    - Incentives break down → Settlement breaks down

Trade Fails: GFC

- Beneficial impact of expanded Fed balance sheet on payments
- Trade fails: Previously high but then down

- Fed now makes available current and historical data on trades in US Treasury and other securities that fail to settle as scheduled
- Substantial variation in the frequency of fails
- Surges in fails sometimes result from operational disruptions
- But often reflect market participants' insufficient incentive to avoid failing



Source: Authors' calculations, based on data from the Federal Reserve Bank of New York.
Note: The chart plots cumulative fails to deliver by month.
http://libertystreeteconomics.newyorkfed.org/2014/09/measuring-settlement-fails.html

Settlement Fails are Commonplace in Contemporary Financial Markets, Today

- Either the agreement to trade securities or the actual settlement of money fails
  - Miscommunication: A buyer and seller may not have a common understanding of the terms of a trade, or fail to communicate settlement instructions to its custodian, or communicate incorrect instructions.
  - System Failure: Well-known instance occurred on Thursday, November 21, 1985, with a computer outage at the Bank of New York. Also, after 9/11, there was an initial surge in fails due to massive operational disruptions, but insufficient incentive to resolve fails contributed to their persistence.
  - Seller does not have the requisite securities in its commercial book-entry account, but it is usually avoided at other times by borrowing securities and delivering the borrowed securities.
  - Cascading failures: Sellers' failure to pay money lead to others' lack of liquidity in the chain. Likewise, failure to pass the security means others will not get it even if as in repo previous agreements they have traded it away.

Problems Nevertheless with Programmed Assets

Hold up problems in trade with intermediaries

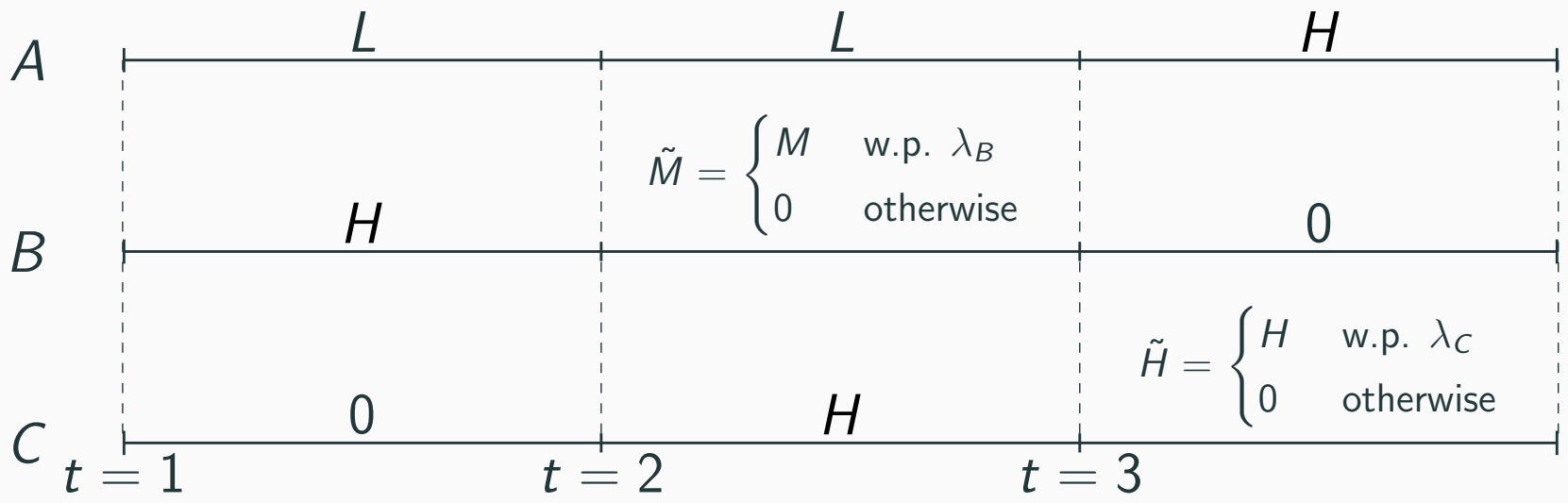
Two Tensions

- Limited Commitment Problem
- **Latent Hold-up Problem**

B's Consideration

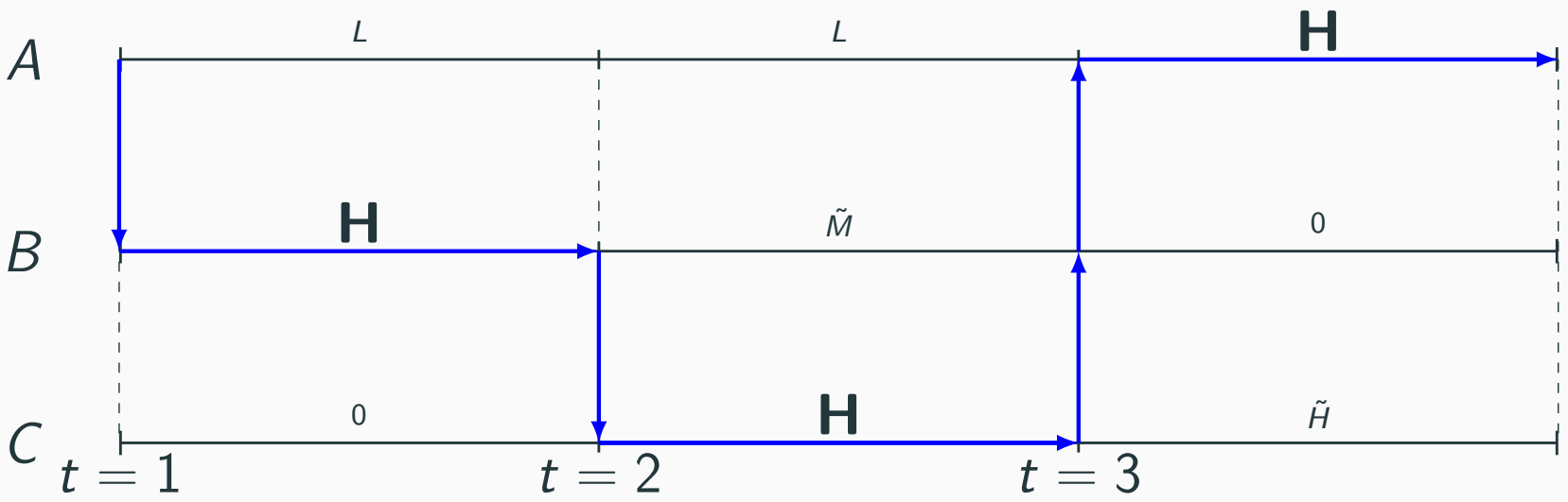
B must agree to lend asset to C for $t = 2$

- B's valuation of asset in $t = 1$ is high
- B's valuation of asset in $t = 2, 3$ is not



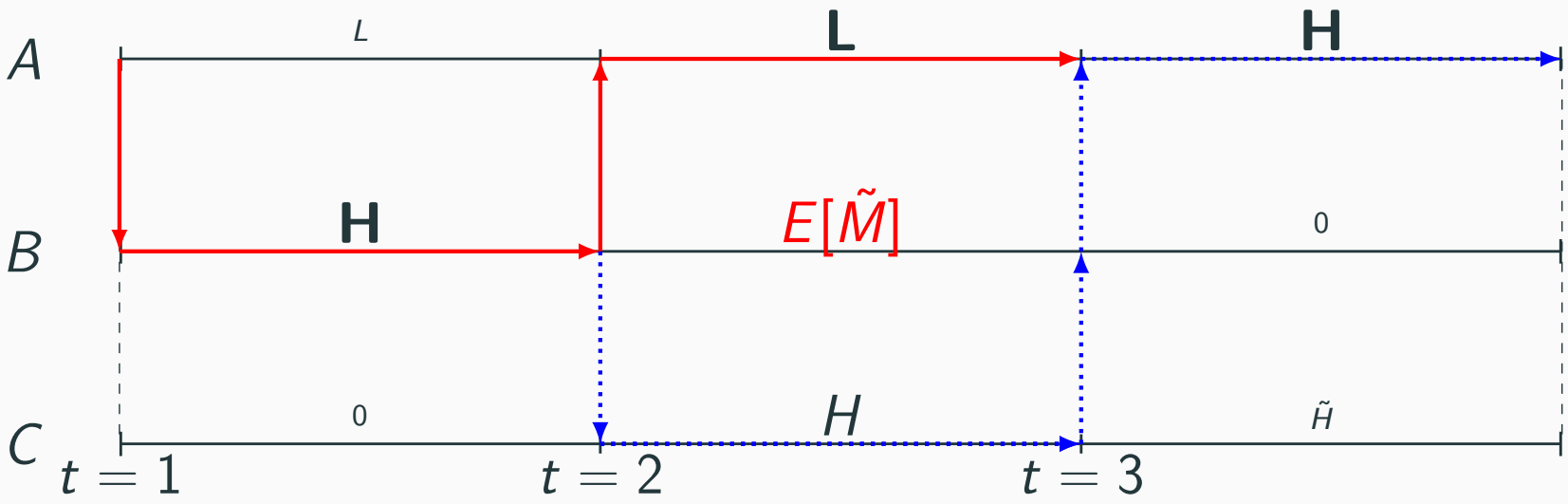
Latent Hold-up Problem

- Only B is matched with both lender A, borrower C
- C can only get asset through B
- Meetings occur asynchronously $\rightarrow$ B "makes markets" by completing one side of the chain in advance of other, anticipating future trade.



Latent Hold-up Problem

- B may need to pay A in excess of his own valuation to obtain asset for C.
- Potential for C to make "low-ball" offer to B, knowing B might have already acquired the asset from A.

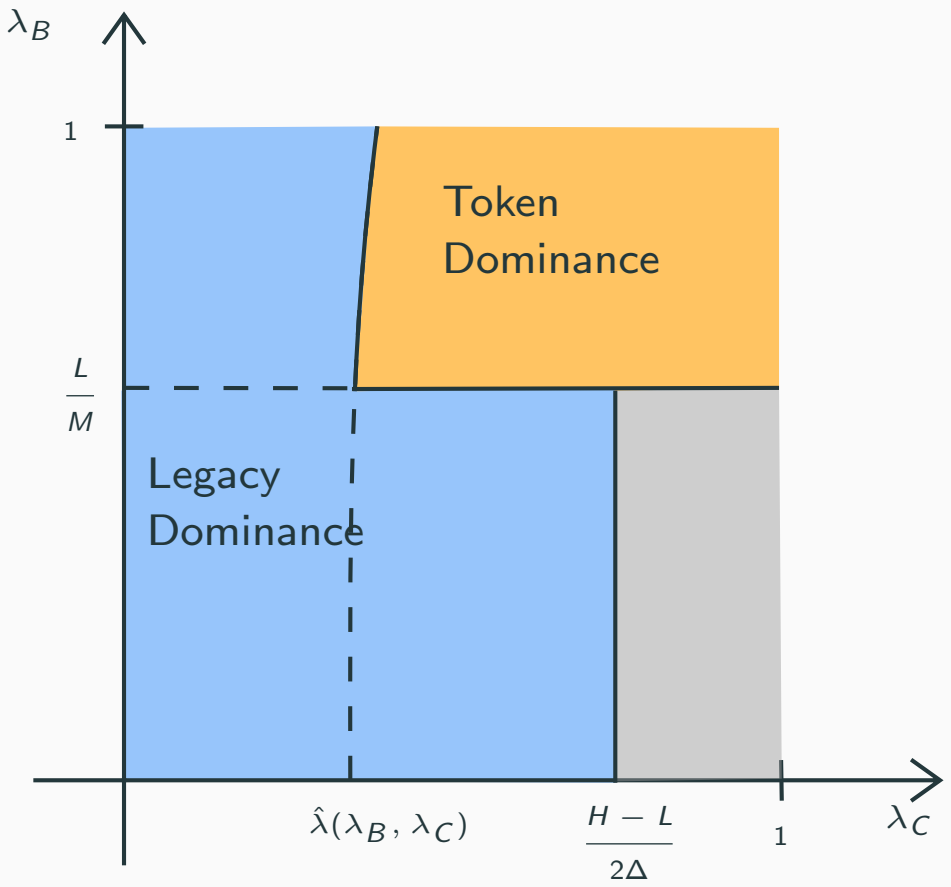


Strong Feature of Legacy System

- Complete decoupling between trade and settlement
  - Traders can enter any contract without explicit proof that they can fulfill
  - B can hide from C whether he has met with A before
- Interplay between the trading system and the settlement system
  - Trading occurs asynchronously, no transparency over the history of trades.
  - Taken in isolation, opaque trading + incentive-based settlement sub-optimal
- Decoupling actually facilitates efficient transfer of ownership between multiple traders

Relative Efficiency

Result. Token system improves efficiency over legacy system only if hold-up problem does not bind ($E[\tilde{M}] \geq L$), and limited commitment problem is sufficiently high.



State-contingent contract

Further Solutions

- Direct centralized trading
- Encryption will be considered subsequently

- Baseline model considers *intermediated trading*
  - Rationale: intermediated trading to overcome (unmodeled) frictions.

- Intermediated trading synergizes with legacy settlement.
- Less compatible with the token system, due to both the trade asynchronicity and information problems.

$\Rightarrow$ Joint consideration of trade + settlement in the design of financial systems.

- Consider alternative trading system *direct trading* that directly matches end-seller and end-buyers for each unit-period claim.

Direct Trading

- Alternative trading stage.

$t = m1$ B and C each simultaneously post ultimatum offers on $t = 1$ and $t = 2$ ownership to A

$t = m2$ A chooses the offers (if any) to accept

- Assume A incurs a cost $\phi \geq 0$ whenever trading directly with C.

Observations.

1. Direct trading resolves *both* issues arising in token system with intermediated trading

2. Legacy system with direct trading still prone to fails by B and C

3. (Direct-Token) always dominates (Direct-Legacy)

Tokenization

- Fungible and non-fungible tokens on EVM
  - Recovering interoperability for some purposes
- Qualifications: Interface with legacy assets, trust

Fungible and Nonfungible Tokens, On EVMs

What Is ERC-20?

ERC-20 is the technical standard for fungible tokens created using the Ethereum blockchain. A fungible token is one that is exchangeable with another token, whereas the well-known ERC-721 non-fungible tokens (NFTs) are not.

ERC-20 allows developers to create smart-contract-enabled tokens that can be used with other products and services. These tokens are a representation of an asset, right, ownership, access, cryptocurrency, or anything else that is not unique in and of itself but can be transferred.

Investopedia.com

In Lieu of Direct Trading: Exploiting the Enhanced Capabilities of Tokens, Approve and Transfer Functions

1. Assuming the tokenized asset resides on the blockchain as an ERC20 token (just for simplicity and to link to privacy-preserving ERC20 examples from the web below - and as it's the most popular tokenization standard. But one could change to any other ERC standard without loss of generality or impact on how feasible the privacy-preserving schemes are, as they work on all EVM compatible blockchains)

2.

A	B	C
X amount	0	0

with Nicolas Zhang--As in the Lee Martin Townsend paper, A owns that token initially. E.g. in the ERC20 smart contract ledger (seeing this not as a token owned only by one participant, but as a ledger in the smart contract is important, as that's what technically happens, and as that explains better the *approve() and transferFrom() function.* See https://metaschool.so/articles/what-are-erc20-approve-erc20-allowance-methods*)*

3. A and B meet. They can agree on conditions for B to potentially sell to C at a later stage, but say B doesn't want to commit to it, or to buy outright the asset. Then A can just tell the ERC20 smart contract representing the tokenized asset that "A approves B to transfer up to 1 unit of the asset when B chooses to do so". Then in the ERC20's ledger, it looks like

A	B	C
X amount (but with 1 amount that B can transfer unilaterally out of A's account to anyone)	0	0

4. B and C later meet. If they reach a good deal for this asset, B can just tell the ERC20 smart contract to transfer From A's account to C 1 unit (which he previously got approval for). Again see https://metaschool.so/articles/what-are-erc20-approve-erc20-allowance-methods. If they don't reach a deal B can just leave the asset in A's account and do nothing. It can later tell A to revoke this approval, or at step 3 A can set a timer for this approval to expire. There are many options.

ERC20 Approval in More Detail

- This method empowers a token holder to authorize a designated smart contract to spend a predefined quantity of tokens from their balance.
- The ERC20 approve method enables token holders to selectively allow spending for specific purposes or transactions, enhancing the flexibility of token usage.
- Token holders retain the ability to revoke or modify approvals granted through the approve method, providing dynamic control over the smart contracts they authorize.
- **Function Signature**: *function approve(address spender, uint256 amount) external returns (bool)*
- **Parameters**: There are two parameters in ERC20 approve method
  - *spender*: The address of the permitted smart contract or any Ethereum address.
  - *amount*: The maximum token amount the spender is permitted to use on behalf of the token holder.
- **Return Value**: A Boolean signaling the success of the approval. In computer science, a **Boolean** or **Bool** is a data type with two possible values: **true** or **false**.

Encryption, Preview of Coming Attractions

5. All these can be easily kept private, e.g., the table always looks scrambled to participants. That's called homomorphic encryption (HE), and was raised as a solution in the IMF XC blueprint. Basically HE is a math property so that computations *f* can be performed on top of encrypted data *Enc*(). And once this result is decrypted, it still gives the correct math answer, e.g., *f*(*Enc*()) = *Enc*(*f*())

   On EVM platforms several open source libraries and implementations already allow such privacy preserving smart contracting to happen; they are listed below. Many give examples with how ERC20 ledgers can be kept scrambled, so that only a participant knows how much assets he has, but the smart contract can still do these *transfer*, *approve*, *transferFrom* functions, still preventing double spending, and preventing the balance from going negative if that's needed.

   e.g., the table at step 2 is something like

A	B	C
kljahsdiouHSIUDHHv	AJSHGDHGJGHJSD	jknKDUAJSHIK

   and the table at step 3 is then something like

A	B	C
sdaHJJHHG	AJSHGDHGJGHJSD	jknKDUAJSHIK

   See more explanations and examples of HE on EVM smart contracts (so any ERC token, not just ERC20. And beyond any ERC token, HE can be used for any EVM smart contract)

   https://www.circle.com/blog/confidential-erc-20-framework-for-compliant-on-chain-privacy
   https://www.zama.ai/post/confidential-erc-20-tokens-using-homomorphic-encryption
   https://blog.sunscreen.tech/building-a-truly-dark-dark-pool-2/
   https://www.fhenix.io/

The Interface with Legacy Assets, Tokenization

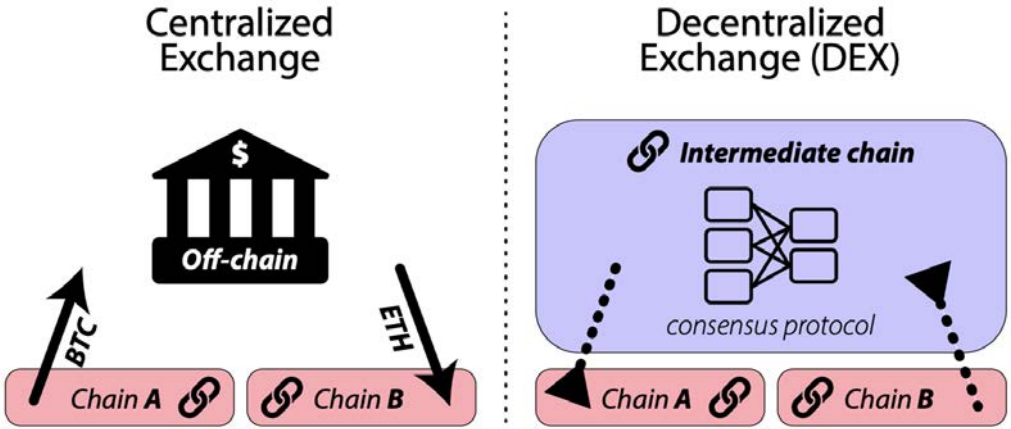
- "…a non-exhaustive list of questions that token developers may need to consider. Important considerations include availability; issuance and redemption; access; underlying assets/funds and claims; transfer mechanism; privacy and regulatory compliance; and interoperability.
- Depending on the design choices made, there could be a number of implications for safety and efficiency of the token arrangement.
- One important element is the nature of the claims on the underlying assets or funds.
- …some wholesale token proposals have attempted to create a safer settlement asset by employing a model that seeks to "back" the tokens "one for one" with corresponding deposits of central bank money, commercial bank money, or some other sufficiently safe and reliable asset.
- For wholesale tokens to derive value or safety from the asset or fund backing they represent, there should be clarity as to on what or on whom a tokenholder has a claim or other right, or how the "backing" relates to that right.
- The assets backing the token, and linkages between the underlying assets/funds and the token arrangement, may have implications for other design choices, such as availability, or hours, in which the tokens can be transferred; the issuance and redemption mechanism; access; and interoperability."

Limited Interoperability, Limited Centralization, Again

Problem

- Blockchain technology is fragmented into different chains that in general cannot communicate or exchange assets
- Existing crypto exchanges are typically not decentralized
  - exchanges maintain liquidity pools on various chains and exchange balances of agents
  - Defeats the decentralized idea of the chain
- Some cross-chain DEXs exist, but require conversion between an intermediate currency, and consensus on an intermediate blockchain
  - Adds unwanted additional overhead and complexity

Current Technology

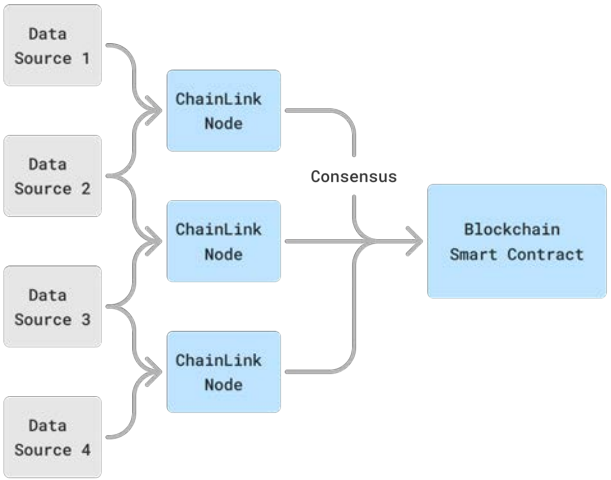


- Centralized exchanges operate much like banks, requiring trust from a user that a transaction will be made
- Decentralized exchanges require no trust, but are computationally expensive from the overhead of an intermediate blockchain which handles consensus in the conversion of currency

ChainLink

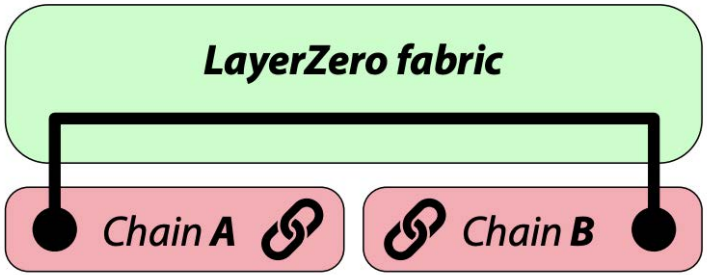
ChainLink is a decentralized ***oracle*** network for off chain events

- Oracle: gives information to on-chain nodes, requires trust in the source
- Nodes gather information from several sources, then perform consensus
- Limited to data retrieval, no mechanism for data transmission



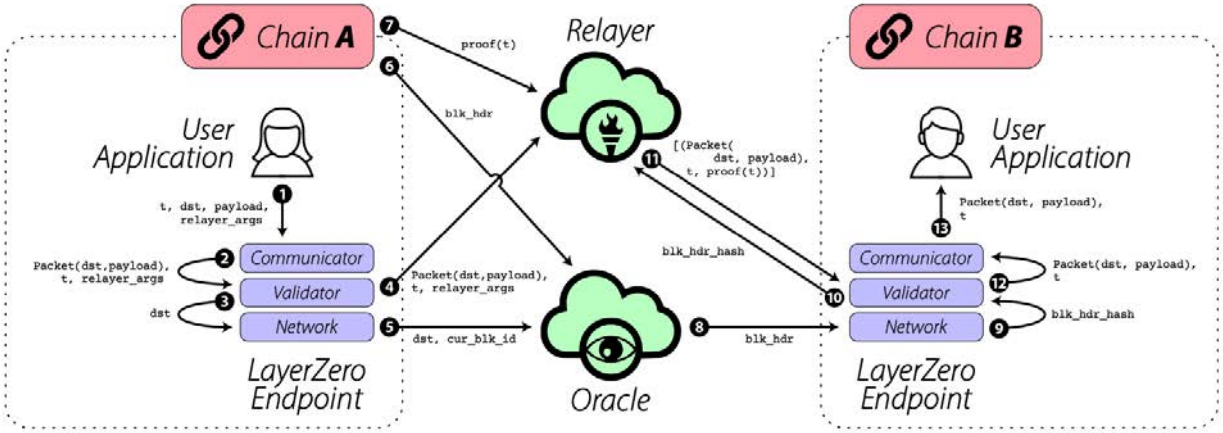
LayerZero

- A completely decentralized messaging system that does not require an intermediate blockchain
- Enables currency exchange and arbitrary messaging between two arbitrary chains
- Guarantees a message will be delivered to chain B if and only if it is committed to the state on chain A



LayerZero Implementation

- Each participating blockchain must contain a LayerZero Endpoint, which is implemented in a smart contract
- Additionally LayerZero requires 2 intermediaries, the Relayer and the Oracle, which both serve to pass information between the two chains



Exchange and Contract Platforms

- The IMF envisions an exchange and contract platform, XC, for cross-border FX
- BIS envisioned a unified ledger within countries, a critical review
- FRB Board, need a coherence guarantee, storge of digital value and programmability
- Interoperability is required, including for contracts

Exchange and Contract Platforms

Summary

Cross-border payments can be slow, expensive, and risky. They are intermediated by counterparties in different jurisdictions which rely on costly trusted relationships to offset the lack of a common settlement asset as well as common rules and governance. In this paper, we present a vision for a multilateral platform that could improve cross-border payments, as well as related foreign exchange transactions, risk sharing, and more generally, financial contracting. The approach is to leverage technological innovations for public policy objectives. A common ledger, smart contracts, and encryption offer significant gains to market efficiency, completeness, and access, as well as to transparency, transaction and compliance costs, and safety. This paper is a first step aiming to stimulate further work in this space.

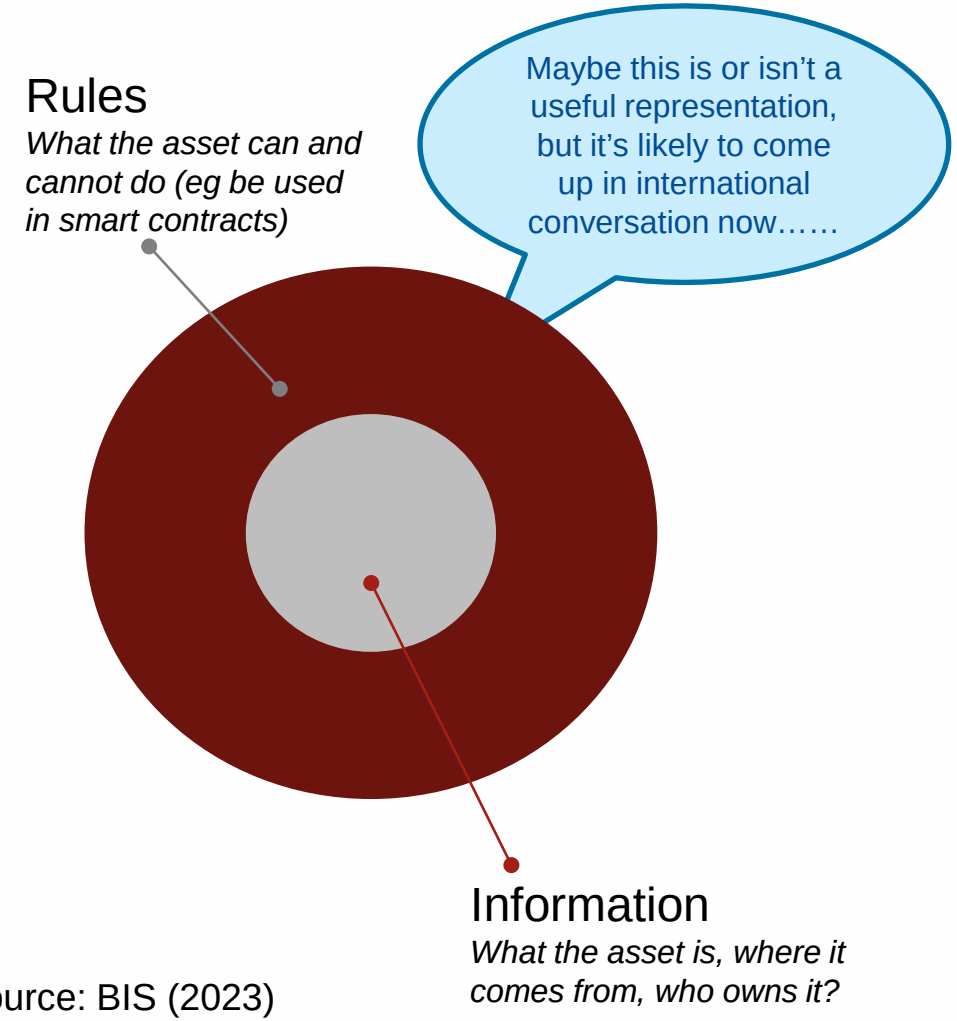
IMF.org

BIS Envisioned a Unified Ledger within Countries: A Critical Review

Discussion of the "BIS – unified ledger"

2023

Tokenization



Source: BIS (2023)

- **Tokenization:** recording/mirroring claims on real or financial assets that exist on a traditional ledger onto a programmable platform.
- **Tokens:** records in a database with the rules and logic (native or not)

Claims on Tokenization advantages

- It combines messaging, reconciliation, and settlement in a single operation, improving efficiency and unlocking new features
- It enables atomic settlement (reciprocal transfers occur simultaneously – one being pre-condition to the other)
- It enables contingent performance of actions in more complex cases (multiple parties)
- It could expand the universe of contractual outcomes for more productive use of resources and better risk-sharing

This works if you are in ONE blockchain

What happens when information comes from “outside”?

other ledgers? “real life” information?

Who custodies non-native assets? Who synchronizes states?

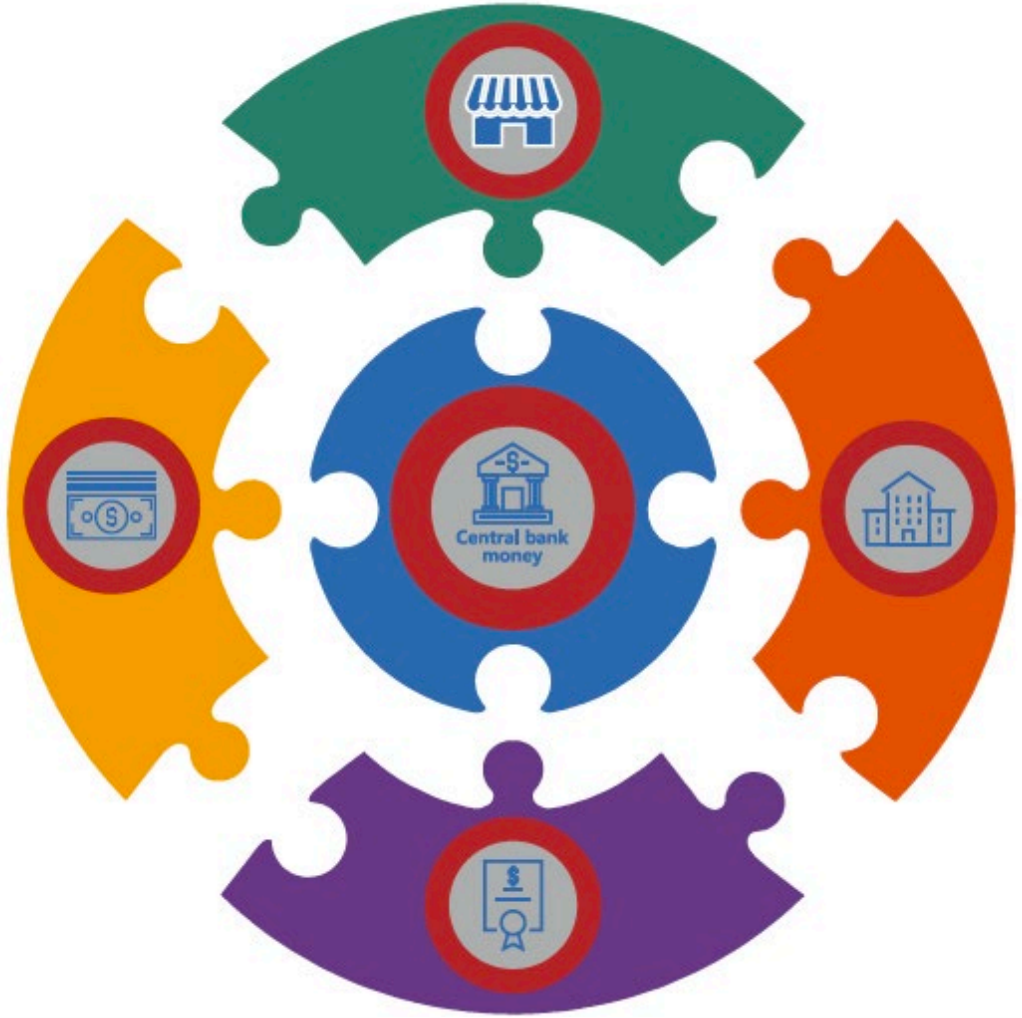
Claims about tokenized deposits

- The paper make **three claims to why tokenized deposits are better compared to the alternative of asset-backed stablecoins**:
  - *"**First**, tokenised deposits would help preserve the singleness of money." [p 90]*
  - *"**Second**, payments in tokenised deposits settled in wholesale CBDC would ensure finality. By using its own balance sheet as the ultimate means of settlement, the central bank provides the means for ensuring the finality of wholesale payments." [p 91]*
  - *"**Third**, tokenised deposits would ensure that banks could continue to offer credit and liquidity in a flexible way." [p 92]*
- ***In contrast** to tokenised deposits, stablecoins represent a transferrable claim on the issuer, akin to a digital bearer instrument. As stablecoins are tradable, their prices can deviate from par, thus undermining the singleness of money. In addition, stablecoins do not allow for elastic liquidity.*

Why even have the CB do anything? (eg why connect CB money to this?)

How BIS defines unified ledger

A more fundamental route is to bring together CBDCs, tokenised deposits and other tokenised assets on a shared programmable platform – a unified ledger



+ the only other definition from their presentation:

"bringing in the same venue tokenized assets, tokenized deposits, CBDCs"

- What would be that "venue"?

Their diagram seems to suggest role for both private platforms and central banks

- How to bring these assets there? They mention "API"

What could be considered "unified ledgers"? What not? (1/2)

These *could* all be "unified ledger"

Via a public platform

- Connecting DLTs to existing CB infrastructure *(RTGS/FPS/…)* → here CBDC = existing central bank reserve systems
- Via new CB DLTs to programmably bridge private and public tokens → here CBDC = tokenized CB reserves/Tbills/…

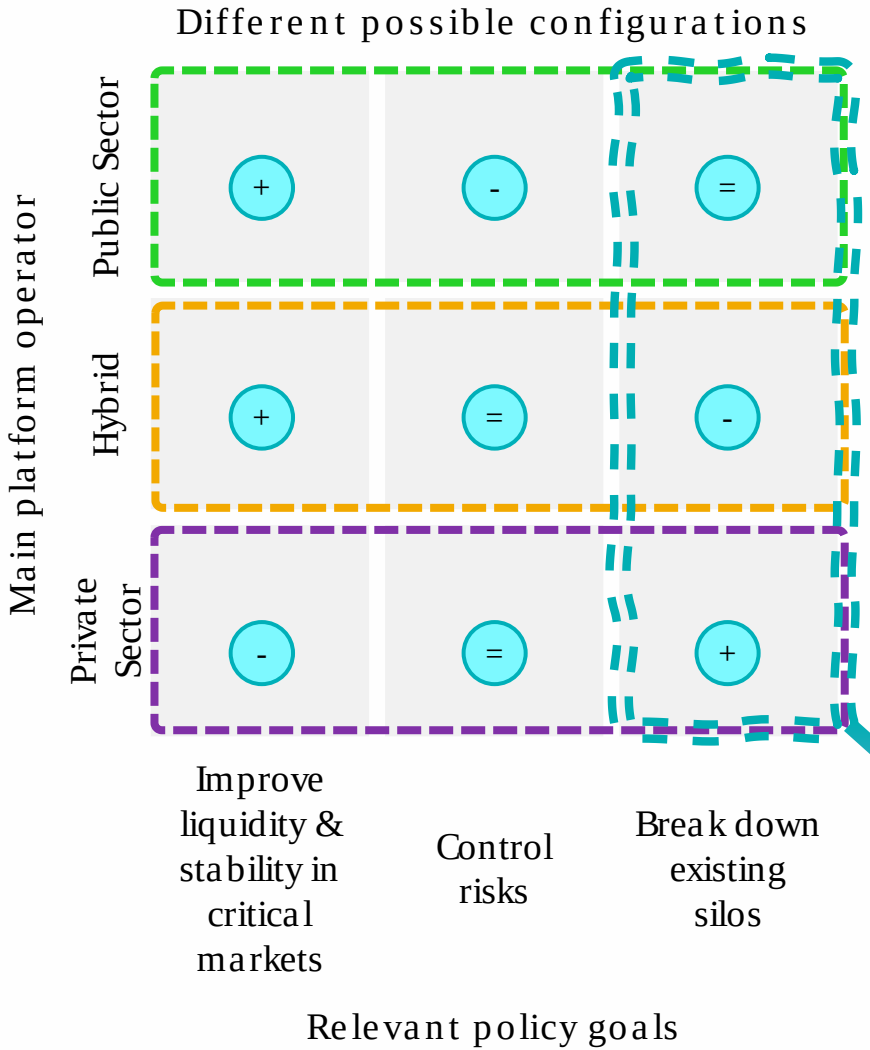
Via regulations/partnership

Existing assets are tokenized, custodied and reconciled by someone (either public or private, as in "RLN")

- These tokenized monies can be ported to a public DLT platform
- Or exist solely on private, regulated DLT platforms (here there's no CBDC!)

Why does this framework / design analysis matter? (1/4)

- Each design defines CBDCs very differently; "*unified ledger*" bundles too much – might be a dangerous shortcut



Could retail CBDC be used as wholesale CBDC? (*wCBDC definition1: "unified CBDC"*)

Should the Fed / Treasury tokenize native objects, to be used on private platforms? (*wCBDC definition2: "Fed reserve CBDC"*)

Should the Fed regulate private platforms/provide public one? (*definition3 – alternative to wCBDC: "regulated liability network"*)

Combinations of definitions above are also possible – e.g. native digital UST *and/or* rCBDC , *on* a Fed run unified platform

Necessary for Programmability: Coherence Guarantee

- “…programmable money as a unified, coherent product that encapsulates both the storage of digital value and programmability of that value.
- This note uses the term coherence guarantee to refer to a mechanism guaranteeing that the technical components of the programmable money product are "inseparable" and that those components are consistently functional, such that the product is stable and coherent for users.
- While nominally "separable" technologies (including traditional technologies that predate DLT) may be used to implement the two core technical components depicted above, the coherence guarantee must ensure that they are not separable as far as the overall product is concerned.
- Without this guarantee, the individual components are nothing novel: Digital money as a product has existed in many forms for years, as has the ability to write computer software.
- With the guarantee, programmable money may be seen as a new product category sitting alongside existing money products such as central bank deposits, demand deposits, nonbank money, and cash.
- The benefit of stability that this guaranteed inextricability provides users of the product may not be present in other systems relying on services that can be altered by their providers and which may be subject to outages.”

FRB Board of Governors