A decentralized implementation of the IMF XC proposal, to anchor exchange rates between cryptocurrencies (and with fiat!)

By TOWNSEND, ROBERT M. AND ZHANG, NICOLAS X.*

This paper is a technical illustration of ideas expressed first in the IMF's XC proposal, written in the context of fiat currencies and central banks. A blog post motivating a transposition of these ideas to crypto currencies can be found here, as part of a broader agenda expressed in the 2023 American Economic Association's *Papers&Proceedings*, and supported by a presentation at their 2023 Annual Meetings.

Actual implementation could look very different, but we hoped to provide for concrete architectures and designs which would cast light on how such a scheme could work.

I. Goal: let smart contracts create (in a decentralized fashion) pseudo-central banks (from a FX point of view) that could prevent cryptocurrency crashes

A common criticism of cryptocurrency projects, especially after FTX's downfall, is that issuing a token is in a way creating a Ponzi scheme, as participants are incentivized monetarily to create mechanisms to prop up the prices of the crypto assets they hold (for instance by hoarding tokens, by burning supplies as in Ethereum after the London fork, by forcing a security token to also be used an utility token so that users of a service have to acquire these as Chainlink or Compound do). These perverse incentives create wide swings in crypto asset prices, and create ever stronger incentives to fraud (even if sometimes the initial intention was benevolent — I will refer here to the psychological cycle of fraud example, which sees someone making an initial mistake doubling down to try to cover it up and repair the mistake — as the Hollywood "Rogue Trader" movie of how an obscure Singapore based trader brought down the "the oldest and the Queen of England"'s Bank illustrates). A second critic to crypto markets is that because there is so much leverage, a shock spills easily and contaminate the entire industry very quickly and very dangerously (as summer 2022's Luna crash brought down FTX).

The two issues are linked, as the multiple generations of models of (fiat) currency crisis illustrate - McKinnon Pill (1996), Krugman (1998), Corsetti, Pesenti, Roubini (1998) suggested that "over borrowing" by banks to fund moral hazard lending was a form of hidden debts which could lead to currency crisis. Radelet Sachs (1998) suggested that self-fulfilling panics can hit financial intermediaries and force liquidation of long run assets, which then "confirms" the panics as in a bank run. Chang and Velasco (2000) argue that a currency crisis may cause a banking crisis if local banks have debts denominated in foreign currency. All these are transposable to cryptocurrency crisis, as are the (fiat) solutions to currency crisis: (1) a lender of last resort, (2) large amounts of reserves and sane balance sheets, and (3) strong coordination with other currencies including swap lines and FX intervention measures.

We would like to illustrate below that it's not impossible, nor absurd, to create *pseudo*-central banks even in cryptocurrencies, and that these could be beneficial to the industry as a whole, as to participants creating them. Indeed, what would be core drivers of a cryptocurrencies' price (relative to other currencies)? The 2 (and only 2) factors are: (A) the beliefs of people holding both this cryptocurrencies and the other currencies against which it's being priced against, and (B) how much of each of these currencies holders sell - which will in turn determine the price direction of that currency pair (see this paper for the clearest model of the determinants of FX rate for fiat currencies).

As an astute reader would note, (A) can easily be aggregated (even privacy-preservingly as detailed later) by code (we will discuss below if on-chain or off-chain), and these beliefs can

* Townsend: MIT Economics, 50 Memorial Dr, Cambridge, MA, rtownsen@mit.edu. Zhang: MIT CSAIL, 32 Vassar St, Cambridge, MA, nxyzhang@mit.edu.

be quantified by amounts of reserves committed by participants to back their beliefs (akin to an auction or a voting process). And once code determined (privacy-preservingly again) the "winning belief" (a price level between two cryptocurrencies) that bears the highest amount of committed reserves, these highest amount of committed reserves can then be used to defend this currency's price in (B) (most central banks actually perform versions of this "systematic managed float", as described in this empirical survey paper).

In fact, to the contrary of fiat currencies' prices which are still constrained by a country's trade flows and economic fundamentals, nothing but beliefs guide the prices of cryptocurrencies (relative to one another). So if a major part of any two cryptocurrencies holders participate in the scheme described just above, they can just "force" their collective beliefs on the rest of the market, and make it the only existing price between the two cryptocurrencies! And in the process generate a lot of profits (from arbitraging other exchanges with other prices), that can be shared among them. So this is another incentive (beyond the stability and anchoring ones laid out previously) for token holders to participate in this scheme, which we will describe in further details below.

II. How to ensure fairness so that no one can game the system - using privacy-preserving computation so that no one knows what the value of the "winning belief" is

To ensure that know one can see the value of the winning bid (a price level between two cryptocurrencies), so that no ones tries to game it (by front-running it, by trying to break it with a larger amount of reserves backing a different belief and win over the profits from arbitraging...) we propose the use of privacy-preserving computation, so that the code operates on *encrypted* beliefs and values, and so that no one can interpret, nor intercept, this information. For that we make use of homomorphic encryption (HE), and multiparty computation (MPC).

The key property of homomorphic encryption, HE, is that a function f can operate on a true underlying space or equivalently operate on the space of encrypted (i.e. encoded) values. That is, in order to utilize a function f in some application, one does not need to input the "plaintext" information which reveals the original and true values of a given agent, but rather, one can encrypt the agent's data, so that data can be kept private, apply the same function f on encrypted input data and get a result in the encrypted space. That is f(Enc(m)) = Enc(f(m)). Equivalently: Decipher[f(Enc(m))] = f(m). The distributivity allows operations outside the parenthesis to be "distributed" to the elements within, whereas a non distributive scheme doesn't allow such treatment of the parenthesis. This allows us to perform a series of composed transitive functions g all on top of the encrypted message, such as: g[f(Enc(m))] = g[Enc(f(m))] = Enc[g(f(m))]. Equivalently decipher(g[f(Enc(m))]) = g[f(m)]. Thus, the agent or contract performing all the functions never gets to see the actual content of the message, nor the true outcome of f in the normal space

The key property of multiparty computation, MPC, is that to determine the value of a function f with inputs from multiple agents, one can run f on encrypted private inputs (with each private input being encrypted differently by each agent) and still get the correct output value of f after decryption. Namely, for J agents Decipher[$f(c_1,c_2,...,c_J)$] = $f(m_1,m_2,...,m_J)$ where $(c_1,c_2,...,c_J) = (Enc_1(m_1),Enc_2(m_2),...,Enc_J(m_J))$ are the encrypted values of the inputs from the J agents, each encrypted using a different key. Distributivity of MPC fails, and we can't compose transitive functions g on top of f - ie Decipher[$g(f(c_1,c_2,...,c_J))$] $\neq$ $g(f(m_1,m_2,...,m_J))$. But combining MPC with HE as in Asharov, Jain, and Wichs 2012, we have MPC plus distributivity - ie the not equal sign above becomes an equality.

Overall, let's note that with HE and MPC, encrypted inputs can be kept private and still contribute to operations performed on top of them. We will call that a "private-but-contributing-state-of-information". What private information to reveal (or not) even as a computation is run on it, and who can decide on it, become a new choice element of mechanism design. For illustration, here below is a list

of the main steps followed by each agent in a privacy-preserving computation such as that of de Castro et al 2020. Note that these are just specific implementations, but these tools can be tailored to other problems.

1) Each agent generates its own key pair, where each key pair contains a public encryption key and a private decryption key
2) All agents submit their public keys to (a) central server(s) chosen to perform the homomorphic operations - ie to run computation on encrypted data.
3) The trusted server(s) combines all agents' public keys into a single joint/shared public key, which is distributed to all agent
4) Each agent encrypts its private data using this new key, generating a ciphertext (an encrypted block of data) that cannot be decrypted by anyone
5) Each agent sends the ciphertext of its private data to the trusted server(s). This ciphertext completely hides the agent's data.
6) The trusted server(s) run computations on all the encrypted data, producing an encrypted result of the computation.
7) This encrypted result can be compared by the trusted server(s) to a similarly encrypted value of the current price. Depending on the sign of this comparison, the code will then know if it should buy or sell the reserves committed to achieve the "winning belief".

Figure 1: main steps followed in an HE-MPC computation adapted to this problem

III. What can be on-chain, what has to be off-chain

Ideally all steps above should be on-chain, but currently HE-MPC doesn't exist on-chain yet (it should be feasible to make a version compatible with evm, just no one did it yet, mostly because it requires trusted servers that introduce centrality). If for immediate implementation, one option would be to have a DAO maintain and ensure proper running of an off-chain codebase doing the above, with the DAO members generating the key pairs of step 1). Aggregating amounts committed by participants are to be transferred to the smart contracts, which then take as oracle input the "buy" or "sell" order from the off-chain code, so that this smart contract can buy or sell from other exchanges and conduct arbitrage depending on whether *Enc(price on an exchange)* is superior or inferior to *Enc(the price resulting from the offchain computations of this DAO)* (or equivalently thanks to homomorphic encryption - whether *Enc(price on an exchange is superior or inferior to the price resulting from the offchain computations of this DAO)*).

To be clearer in the details, an agent submits to the offchain codebase (with Enc() the encryption operator, making the content of the message secret to everyone else, and in an example between ETH and BTC):

Enc([array of the agent's band preference pair])

With, within the array, each (preference pair) representing, for an agent, the pair of his

{*[desired exchange rate band 1 between ETH and BTC], [X amount of ETH and Y of BTC to be dedicated to its defense should this band 1 be the one selected by the smart contract]*}.

Now the full array for this agent will then encompass all bands and all preference pairs, ie:

[(desired band 1 rate, amount dedicated for it),
(desired band 2 rate, amount dedicated for it),
(. . . ,. . .),
(desired band n rate, amount dedicated for it)]

This participant also send to the smart contract (eg to an on-chain account)

$$Sum_{AllDesiredBands}(amounts_{foraband})$$

There are then cryptographic checks to ensure consistency between the sum of the amounts announced by the participant to the off-chain database, and the amount that he com-

mitted to the smart contract (all in one transaction, so that no one can infer the decomposition of his allocation to different bands even if this aggregated amount is visible on the public blockchain). These cryptographic checks can be as simple as computing correspond to

$$Enc(Sum_{AllDesiredBands}(amounts_{forABand}))$$

on the off-chain codebase, and check that it matches

$$Enc(ammount_{sentToSmartContract})$$

IV. Incentives to participate via remuneration from arbitrage

A scheme to distribute profits from arbitrage could be paired too, for instance for every token deposited a soulbound (ie non transferrable) receipt token is given to the participant, so that it receives a proportional share of the profits.

Using homomorphic encryption (definitions and tutorials here), the off-chain codebase looks for

$$Enc(\max_{acrossAllBands}(\sum_{acrossAllAgents} amountsdedicated))$$

and remembers the Enc(max([band value]) corresponding to it (it's like selectioning a winning vote object among all participants, with the dedicated amounts being the bidding units!). Ie the off-chain codebase is aggregating all encrypted inputs, in a still encrypted format (but understandable by the smart contract — and only by him), so that the smart contract can act on the encrypted result (without anyone else being able to decrypt this encrypted result). As the smart contract performs the comparision on *Enc(price on an exchange)* vs *Enc(the price resulting from the offchain computations of this DAO)* and buys/sells this way, if the Enc operation is done using a private key that only the smart contract has (so that it's the only one who can perform this comparison) then no one but the smart contract has this preferential information on which to arbitrage other exchanges, so that outside of the smart contract no one (including members from the DAO) can leverage this preferrential information to arbitrage or front-run himself. In fact, if the reserves committed to the smart contracts are large enough compare to the depth of the order book then the smart contract should be able to arbitrage the other DEXes until they converge to the Enc(max([band value]) resulting from the off-chain privacy-preserving computations!

The same would work for several cryptocurrencies, in a way so that all conversions are equivalent (ie converting ETH to BTC is equivalent to convert via vehicle cryptocurrencies ETH to LTC to ... to BTC as in a grid of exchange rates). There agents would submit multiple copies of the array above, one for each currency pair they want to be involved in.

V. Further/future improvement to on-chain verification of the off-chain computation

To check that the off-chain computations that compared each of the {amounts for a band} resulting in the Enc(max([band value]) were done correctly, participants can leverage zero-knowledge verifiable computation. There, one can prove that f(x)=y where y is a public output (Enc(max([band value])) but x is private (here it can be everyone's {amounts for a band} except that of the agent wishing to verify the computation). The off-chain codebase could integrate zero-knowledge verifiable computation so that during the computation it generates a circuit for each of the different agent of the form above, so that each agent could verify with his own input (that only he knows) that the computation is correct.

VI. Is this type of thinking desirable

We hope to convince readers that this type of monetary anchoring and financial stabilization of cryptocurrencies (using only cryptocurrency protocols) is possible - ie that we can provide a strong tie and stable exchange rate among cryptocurrencies, and with fiat currencies as well, as we could even include one fiat currency in the multi-currency mix just above. We welcome discussion and debates around whether that is desirable; for not, the pros/cons we could think of are listed below:

- Decreased volatility between cryptocurrencies, for instance if we only allow agents to select

price bands in the vicinity of the current exchange rates (or to only select volatility bands and not price levels).

- Increase financial safety: indeed if we are in the grid of exchange rate situation presented above, then catastrophic crashes like that of Terra or FTX would be more gradual then sudden: a "smoothing/leaning against the wind" is being paid for by those who believed in these currencies and wanted to define its value, committing some of their own reserves to its defense *(actions, not just words!)*

- By having a stronger, collective framework smoothening over cryptoasset prices, we would limit the growth speed of any specific token (as to make its price relative to other currencies go up it would need to convince holders of these other currencies to commit some of it to prop up/defend a higher price), and also provide more resiliency so that an individual error will less likely trigger cascading failures (similarly to currency crisis in the fiat world)

- If truly successful at arbitraging away other exchanges then it might mean the end of AMMs, as AMMs provide other exchange rates. That would be the end of quite a lot of experimentation (that produced great technical innovations and creative solutions), and created stable jobs in the industry, so it's not to be taken lightly. But on a purely economic design front, we could argue that these are just how maturing designs replace prior experimentation as newer designs become safer, more stable, and probably more fair for users (both for those wanting to use cryptocurrencies as means of payments and not just for speculation, and those retail investors being ripped off by more sophisticated institutional traders currently making a profit on DeFi, as described in this empirical survey paper)

- In the long-run/steady state, if arbitrage is really successful, then there probably won't be much divergence between prices left, so probably that will decrease exchanges' margins and market makers' rents, and so lower conversion cost for users.

VII. Conclusion

There are then, as one can see above, many great debates to have in this DeFi space, with big definitional and consequential decisions to be taken on how crypto-markets are organized, how innovative and flexible they can be, but also how safe, how fair, how predictive, and how credible they should be. Now is probably a good timing to have these discussions, as many are now gloating over how unsafe cryptomarkets were, as the past summer's Terra/Luna crash (partly) lead to FTX's downfall (and they say, potentially many others to come!). We hope that this discussion, and subsequent initiative, could safeguard crypto experiments while taking out some of the incentives for speculation and deception in cryptomarkets, and encourage designers of crypto applications to focus more on the initial philosophy of decentralization and collective improvement.