Special thanks to some discussions from Dominic Williams from 2015 that brought up the idea of multi-issuer collateral-risk-reduced stablecoins.

One major type of “stablecoin” that already exists, and will likely continue to become more popular, on public blockchains is the issuer-backed token (eg. Tether, and likely soon Digix Gold). In this model some centralized issuer maintains a reserve (likely a bank or brokerage account or vault depending on context) of some underlying backing asset, and issues a quantity of tokens on the blockchains equal to the quantity of the backing asset. They promise that each unit of the blockchain token will be redeemable for a unit of the backing asset.

This kind of token is attractive because it avoids thefinancial “black swan” risk of stablecoins like DAI that are purely backed by crypto, but it comes with its own set of problems, chief among which is that it brings back the spectre of counterparty risk. Issuers of backed tokens are often met with suspicion, and the tokens are often avoided as a result.

The following is a proposal for how to mitigate this risk, effectively creating 1-of-N issuer-backed stablecoins that only fail if most or all of the issuers fail.

Suppose that we have N issuers of USD on the blockchain, I[1] ... I[n]

, and there exists a DAO into which M coins from each issuer have been deposited. The DAO releases N new assets, which we call “slice 1” … “slice N”, effectively ordered low risk to high risk. The goal is as follows: buyers of slice 1 will be able to redeem a dollar if at least one, any one, of the issuers continues to be solvent. Buyers of slice 2 will be able to redeem a dollar as long as at least two issuers are solvent. And so on and so forth until slice N, which will be able to redeem a dollar only if all issuers are solvent.

The expected loss component of counterparty risk can never be reduced or removed - if the issuers collectively lose $X, that loss of $X has to be paid by someone

. But what this does let us do is channel the risk toward those who are most willing to bear it, and give those who are not an asset that is highly robust, losing value only if a very large number of issuers fail.

To compensate those who are willing to bear risk (or those with insider knowledge that allows them to trust the issuers more than the general public does), the holders of slices closer to N would be paid interest rates, which would come out of the pockets of the holders of slices closer to 1.

Now, on to implementation. The coins are issued at time T, and have a pre-determined duration D. Before T, anyone is allowed to specify a bid, of the form “I want to buy a unit of slice i

, and I am willing to pay x / N

units of every

coin in the basket to purchase it”. The system then keeps track of the bids in highest-to-lowest sorted order for each slice, and once the bidding period ends it starts processing the bids. It looks at the top bid for each slice, and sees if the top bids sum to at least 1. If they do, then it accepts the bids, and if they do not then it terminates.

At the end of the process, everyone who bid for slice i

pays the same price as the last (ie. lowest) accepted bid for slice i

. This mechanism ensures that, for every set of bids the system accepts, the system issues one coin for each slice and receives at least one coin from each issuer as backing, so it will be able to meet all obligations.

At time T + D, comes the claiming phase, which is split into N periods. In the first period, everyone who has a coin of slice 1 can redeem it in exchange for a coin from any issuer of their choosing. In the second period, everyone who has a coin of slice 2 can do the same, though if there is some issuer whose coins have already been fully drained by the redeeming process they can naturally no longer be claimed. This continues for all N slices.

This mechanism removes the need to have any kind of fancy dynamically adjusted/controlled interest rate, or an oracle to tell which issuers are insolvent. If k

of the N issuers are insolvent, then holders of coins in slices 1…N-k would redeem all of the solvent coins first, leaving the holders of coins in slices N-k+1…N with worthless coins; the need for an oracle is substituted with market-based preference revelation.

To create an infinite-duration coin on top of this, one can simply imagine a DAO that creates rounds of this game with duration 2D every D (ie. there are always two overlapping games) and another DAO which buys tokens of some specific slice on the open market a quarter of the way through their period and sells them three quarters of the way through to buy the coins from the next game.

Variations

- Have one of the “issuers” be a contract that holds ETH and has a redemption process that allows holders of a coin to claim an amount of ETH equivalent to 1 USD. The contract’s USD liabilities would be half the value of its ETH holdings at the start, and if the contract becomes insolvent it would simply give each token holder an equal share of its entire quantity of ETH. Any undistributed ETH would be given to a second class of token holder, who would thus be holding “ETH at 2x leverage”
- Come up with more complex combinatorial mechanisms that allow people to express through the market opinions like “I think issuers 1, 4 and 11 are solvent but have no idea about any of the others”
- Have one of the “issuers” be DAI