Checker

Arthur Breitman
@arthurb

September 30, 2019

Disclaimer

- These slides explore ideas for the design of a software technology
- Like all exploratory ideas these are subject to change
- I may use loose vocabulary and metaphors to better convey ideas
- These metaphors break down past a certain point
- The system should be understood based on what the actions its actually programmed to take

What is Checker?

Checker is a software technology for stabilizing a token with respect to an externally provided index.
Concretely:

- Defined as a Tezos smart-contract
- Index can be any peg, e.g.
  - Price of 1 oz t of gold in tez
  - Price of 1 HKD in tez
  - Viewcount of baby shark videos on January 1st 2030

Goals

Primariy design goal is to minimize trust:

- Minimize discretionary inputs: minimalistic governance
- Minimize reliance on centralized point of failures: no centralized collateral
- Minimize reliance on external actors

Design limits:

- Peg can break
- Parameter tuning is still worth having
- Oracle feeds are a weak point

Model Design

- Modular design to make it easier to reason about the system.
- Relies on three different tokens
  - Draught token
  - Checker token
  - Kits

Locking tez

Locking tez in Draught. User selects:

- Locking time (e.g 1 day, 1 month, 1 year)
- Unlocking amount (e.g. spend 100 kit to unlock)
- Forfeit threshold (e.g. 120 kit worth of tez)

Draught contract offers some amount of kits to the user based on the above variables and an internal model.

- Draught contract sends kits (fewer than the unlocking amount)
- Tez is locked. *User still controls delegation.*

Draught

Users locking tez interact with a Draught contract

- Draught model determined by tunable parameters
- Draught is itself tokenized in draught tokens
- Draught targets fixed kit holding, e.g. 5%
  - Cannot create or destroy kit
  - Maintain enough kit to interact with users
  - Create draught tokens and auction for kit
  - Or auction kit for draught and burn
- If user forfeits tez, tez held by Draught is auctionned off for kit

Checker

Checker is also a tokenized contract. Tokens have a dual purpose.

- Governance
  - Used to vote on Draught parameters
  - Used to select an oracle feed
- Stabilization
  - If kit above beg, mint kit, auction off for checker tokens, burn
  - If kit below beg, mint checker tokens, auction off for kit, burn
  - *Acquire draught tokens...*

Checker and draught

Would this stabilization mechanism work by itself? Sort of …

- Checker holders may want to minimize dilution
- Rational course of action:
  mint checker tokens, exchange for draught tokens
- Don't rely on business acumen, automate the process

How to automate?

- Draught exports kit unlock amount per token
- Checker automatically mints checker tokens and auctions them off for draught tokens
- Checker has no privileged access to Draught tokens!

Auctions

How does a smart-contract exchange one kind of token for another? Auctions

- Simple to analyze
- Descending auction is gas sparing
- Typical horizon on the order of days
- Alternatives (e.g. Dexter) assume liquidity
- Downside: slow

Oracle feed

Contract needs two oracle feed

- tez with respect to peg
- kit with respect to tez

A few remarks

- Second one observable on-chain
- Oracle construction is out of scope for Checker
  - Median of 5 reputable sources
  - Decentralized Oracle à la Augur

What about MakerDAO?

- Similar to MakerDAO with a few key differences as outlined above
- Pretty much the only reasonable category of design for this
- Dai has survived very wild periods relatively unscathed

Failure scenarios

Here are a few ways in which this can break down

- Malicious mispricing from oracle feed
- Forfeitures spiral
- Demand for kit can outstrip interest for locking tez
  this is the weird / important one
- Draught token cornered
- Checker governance picking loose Draught parameters
- Implementation bugs

Questions

Questions ?