Dripper Specification

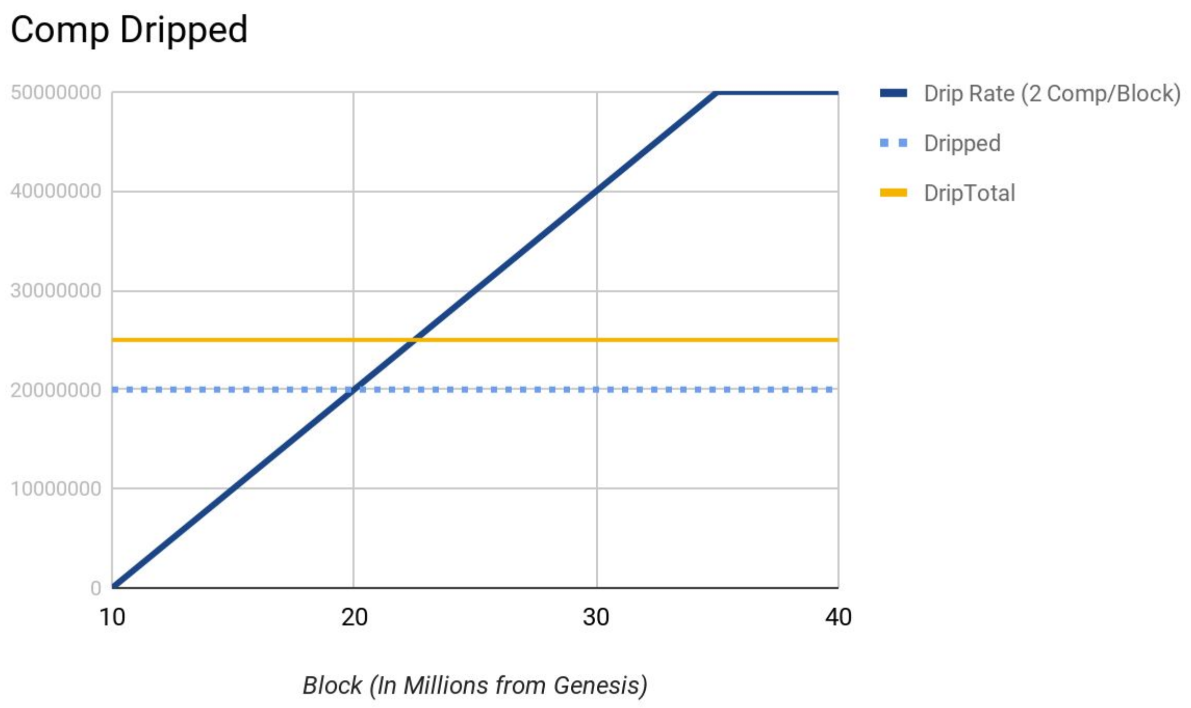
Compound Engineering

Overview

Comp will be distributed to the community through the Flywheel mechanism in the Comptroller. To prevent the community from making a proposal to speed up the Flywheel (or simply gift the Comp to some other address), we propose a Dripper that will give the Comptroller its Comp tokens over a period of years. The Dripper will be a separate smart contract with a large Comp balance and a mechanism to transfer that Comp to the Flywheel given some constraint on the rate of transfer. We aim to make this contract as simple as possible.

Mechanism

We propose a mechanism that simply tracks how much Comp has been moved versus an immutable rate function. Effectively, we take the following graph and track our previous y-coordinate (*Dripped*). For each call to Drip(), we look at our expected y-coordinate (*DripTotal*), subtract the previous Dripped value, and send that delta amount of Comp to the Flywheel.



This graph shows how much Comp should have been Dripped from some starting point (here we start at block 10MM and we drip 2 Comp/Block).

Algorithm

Static Configuration:

- $DripStart$: A block number when dripping should start
- $DripRate$: Amount of Comp which should be dripped per block
- $Token$: Address of Token to Distribute
- $Target$: Target of Dripper

State:

- $Dripped$: Amount of Token that has already been transferred to Flywheel

Externals:

- $Token.balanceOf(Dripper)$: Dripper's own Token balance
- $Token.transfer(Target,\ amount)$: Side-effecting function to transfer Token to Target
- $Block()$: Current Ethereum block number

d

```
$Define\ Initialize(DripRate^*,\ Token^*,\ Target)$
  $SET\ Dripped = 0$
  $SET\ DripStart\ =\ Block()$
  $SET\ DripRate = DripRate^*$
  $SET\ Token\ =\ Token^*$
  $SET\ Target\ =\ Target^*$

$Define\ Drip()$
  $DripTotal\ = DripRate \times (Block() - DripStart)$
  $DeltaDrip\ = DripTotal\ - Dripped$
  $ToDript = min(CALL\ Token.balanceOf(Dripper),\ DeltaDrip)$
  $Assert\ ToDrip\ \geq 0$
  $SET\ Dripped = Dripped + ToDrip$
  $CALL\ Token.transfer(Target,\ ToDrip)$
```

Correctness

To prove the previous algorithm correct, first we note that DripTotal is always the exact amount that should have been dripped since the start of the contract as it matches the equation listed above. Assume, for the moment, Dripper's balance of Comp is sufficiently large. We break into an inductive proof. In the base case, Dripped is zero and we transfer exactly DripTotal tokens, which is the correct amount. In the inductive case, we have already transferred Dripped tokens, which we assert from inductive was the correct amount that should have been dripped. We are now calculating the amount owed since the beginning and subtracting the amount we have dripped. This is the exact amount we need to transfer to match DripTotal. This is thus also correct and completes our proof.

A Quick Aside

First, we note that our inductive case will succeed *even if* Dripped was not the correct amount to be dripped previously, so long as Dripped is less than or equal to DripTotal. That is, it doesn't matter how much we previously dripped, so long as we haven't dripped too much, since the next call to Drip will correct any issues.

With a Variable Comp Balance

Now we consider the case where the Comp balance is less than sufficient in some cases. Here, we revise our proof using the aside above to realize that the proof still holds so long as we can show that Dripped is never greater than DripTotal in our inductive case. We show this by our *min* statement, which says that we never set ToDrip to be greater than DeltaDrip, though it may be less than DeltaDrip. Finally, we relax the constraint in our proof that we always Drip to the correct DripTotal and instead say that "given a sufficient Comp balance, we always Drip to the correct DripTotal." This completes our (modified) proof.

Assertions

- $Block() \geq DripStart$ - Blocks always increase
- $DripRate \times (Block() - DripStart)$ does not overflow
- Note: $DripTotal + ToDrip$ cannot overflow since it is strictly less than or equal to DripTotal.