Tracer: Perpetual Pools Litepaper

Ryan Garner,
Lachlan Webb,
Harrison Coates
Mycelium

Tim Jiang
Paperclip

Version 1.0
August 27, 2021

ABSTRACT

We propose a simple derivative infrastructure to create leveraged tokens, whereby long and short users have changing claims on a pool of collateral. As long and short collateral tends to parity, due to the implicit **rebalancing rate**, users' returns are expected to simulate a constantly rebalancing leveraged position. These positions are non-liquidatable, fully collateralised, and can exist perpetually without upkeep. This new financial primitive is referred to as Perpetual Pools.

OVERVIEW

Until now, demand for maintenance-free access to fungible leveraged exposure has been met by leveraged exchange traded funds (LETFs). Perpetual Pools trump existing LETFs. Perpetual Pools allow users to acquire low-cost, leveraged exposure to any asset with zero risk of liquidation, zero trust in a centralized party, and zero limits set on leverage.

Perpetual Pools is a financial contract for the transfer of value between **long** and **short** sides of a collateral pool, based on a price feed. Ownership of the pool is divided into shares represented by fungible ERC20 **tokens**. Long side tokens are labelled **L-tokens** and short side tokens are labelled **S-tokens**. The value of these shares is determined by the proportion of collateral held in each side of the pool, which updates according to a transfer function we call power leverage (see: *Power Leverage*).

DESIGN

A Perpetual Pool has two sides, long and short. These sides hold users' collateral and transfer an amount of it from one side to the other when a pre-defined condition has been met (e.g. time period, price change (%), momentum threshold). The amount, and direction, of the transfer is calculated using the power leverage function, which accounts for values provided by a price feed. Any tokenised asset can be chosen as collateral.

POWER LEVERAGE

This design enables fully collateralised, non-liquidatable leveraged tokens if the transfer function chosen a) amplifies the sensitivity of **value transfers** to price changes, and b) cannot return 100% or greater. The function that performs best under these conditions is:

$$\text{Value Transfer (\%)} = \begin{cases} 1 - \left(\frac{P_0}{P_1}\right)^{\text{leverage}}, & P_1 > P_0 \\ 1 - \left(\frac{P_1}{P_0}\right)^{\text{leverage}}, & P_1 < P_0 \end{cases}$$

where P is a value provided by the price feed. Note that the value (P) is not restricted to price feeds, and could feasibly be any data feed.

We call this transfer function **power leverage**. Power leverage gives returns almost equivalent to "times" leverage[1] for typical price movements, but dampens returns to extreme price movements so users can never lose 100% of their collateral.

[1] Traditional leverage that amplifies returns by a multiple e.g., return = 1% * 2x = 2%.

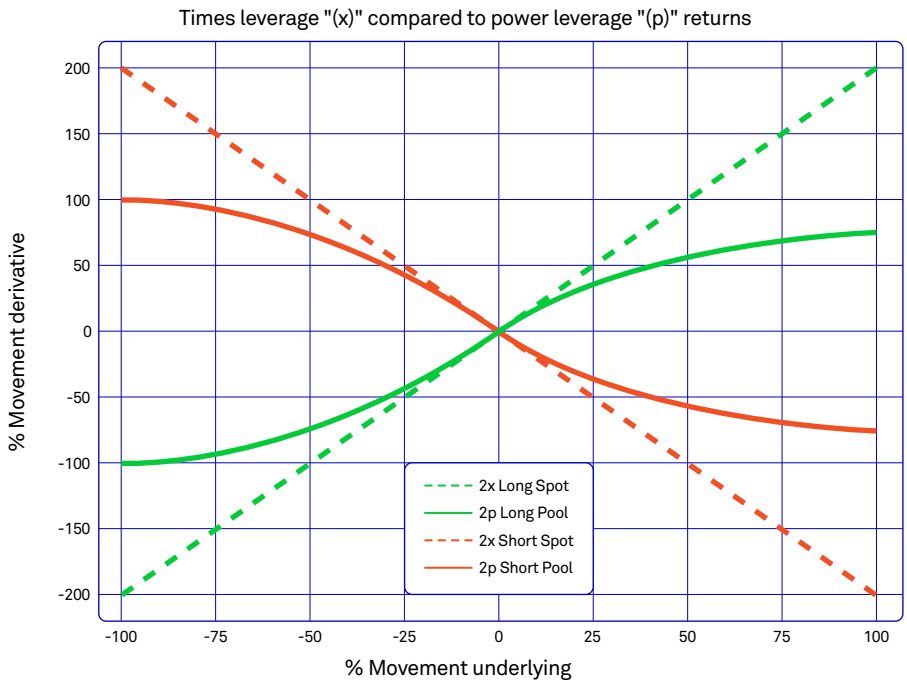

Figure 1: Times leverage "(x)" compared to power leverage "(p)" returns. Notice the asymptotic value transfers for power leverage. The x-axis is the underlying price movement (%), where the y-axis is the corresponding derivative movement (%).

REBALANCING

Users can commit to **mint** (**burn**) to (from) the pool at the upcoming value transfer. A **keeper** executes the value transfer, which updates the proportion of collateral held in each side of the pool and mints (burns) tokens. The percentage transferred is calculated by the power leverage function, using the underlying price feed (given by an **oracle**) at the **rebalancing event**. Collateral is then moved from the short side to the long side in the case of price appreciation or from the long to the short in the case of price depreciation.

Minting (burning) activity does not change the price of the existing tokens, but allows for users to add (remove) collateral to (from) a pool. New tokens are created so that the ratio of pool tokens to collateral remains constant or, if there is no existing collateral of the type deposited, at a 1:1 ratio.

REBALANCING RATE

In the case where long and short collateral is at parity, an x% transfer from one side will equate to an x% gain for the other side. However, in the case where long and short collateral is not equal, an x% transfer from one side will not result in an x% gain for the other side. Instead, the side's gain is adjusted by the rebalancing rate. A favourable rebalancing rate presents asymmetric upside for the less collateralised side. The formula describing rebalancing rate follows:

$$\text{Rebalancing Rate (\%)} = \left(\frac{\text{Long Collateral}}{\text{Short Collateral}}\right) - 1$$

We expect this rebalancing rate to tend towards 0% in a Perpetual Pool, as it has done (historically) for explicit funding rates in Perpetual Swaps. Asymmetric upside naturally attracts collateral inflow, causing a return to parity. The function below can be used to calculate a side's gain:

$$\text{Gain (\%)} = \begin{cases} \left(\dfrac{1}{1 + \text{Rebalancing Rate (\%)}} \right) \times \text{Value Transfer (\%)}, & \text{Long} \\ \left(1 + \text{Rebalancing Rate (\%)} \right) \times \text{Value Transfer (\%)}, & \text{Short} \end{cases}$$

QUALIFICATIONS

1. Claims (deposits) must be made prior to the front running interval to receive collateral (tokens) for that period, else the collateral (tokens) remains in escrow until the next rebalancing event. The front running interval is a parameter measured in Ethereum blocks.
2. Keeper payments are taken from the pool collateral and may be a non-trivial amount long term. The keeper is reimbursed the gas required to execute the job and receives a **tip** for performing the transaction. The tip is set to 5% of the **variable fee** (fast gas + 20 gwei), increasing by 5% for each block the job is not performed.

CONCLUSION

The simple bilateral structure of Perpetual Pools can accommodate any transfer function. Power leverage is an as-yet unused transfer agreement that fits the conditions required for leveraged token creation. It is here described assuming a spot price input, though we expect it to perform similarly (if not better) under different inputs[2].

[2] 0xb2bc296da1a7cb99e6f9078427b020cd66c3223aef855bed620599d9ae7e5329

GLOSSARY

(in order of mention)

Rebalancing Rate
an implicit rate affecting value transfers.
Calculated as (Long Collateral/Short Collateral) - 1.

Long
position that profits when the price appreciates.

Short
position that profits when the price depreciates.

Token
tradable digital asset that represents a position in a pool.

L-token
token that represents a claim on long side collateral.

S-token
token that represents a claim on short side collateral.

Value Transfer
a percentage of collateral transferred from one side to the other.
Calculated by the power leverage function.

Power Leverage
function of price that determines the percentage of collateral transferred on Rebalancing Event.

Mint
creation of pool token/s representing claim on collateral.

Burn
destruction of pool token/s to claim collateral.

Keeper
entity that performs maintenance tasks for smart contracts.

Oracle
entity that streams external data for smart contracts, e.g. market pair.

Rebalancing Event
Smart contract transaction. Includes value transfer and token mint/burn.

Tip
an incentive payment for the keeper additional to the variable fee.

Variable Fee
the gas price reimbursed to a keeper, paid in a pool's collateral asset.
Calculated as the fast gas price + 20 gwei.

ACKNOWLEDGEMENTS

The authors would like to thank Chris Berg, Sinclair Davidson, Jason Potts, Yamashita, Michael Anderson, and Vance Spencer for their contributions to this paper.