CanSwap

A decentralised liquidity network for CanYa.

Overview

CanSwap uses ethereum-based continuous liquidity pools to allow on-chain conversions of tokens and ether into and out of CAN. The continuous liquidity pools are permissionless; anyone can add or remove liquidity and anyone can use the pools to convert between assets. The pools rely on permissionless arbitrage to ensure correct market pricing of assets at any time.

Comparison to Bancor

Bancor also has an on-chain liquidity network, but has the following issues:

- No ability to include the conversion as part of a transfer (assets are always returned back to the user)
- Pools are not permissionless for staking (only the token owner can add more liquidity)
- No incentives for staking
- High gas cost to use the network

CanSwap has the following benefits:

- Users can do both conversions as well as transfers (convert and pay to another wallet)
- Anyone can add liquidity to the network
- Clear incentives for staking
- Much lower gas cost
- CAN is used as the settlement currency of the network
- A clear market for traders to arbitrage

Once deployed, the following will be the benefits to the CanYa ecosystem:

- Conversions and transfers from any token or ether to/from CAN with one atomic transaction
- Trustless on-chain price feeds of ETH/CAN and USD/CAN
- High community engagement with a dynamic, permissionless and engaging market to be part of
- Reduced stock-to-flow ratio with CAN held in liquidity pools

Comparison to UniSwap

Uniswap has the following issues:

- Ether is the settlement currency
- Liquidity fee is fixed
- Each pool is a separate contract, and
- Stakes are tracked by issuing a ERC-20 token for each pool.

CANSwap has the following benefits:

- CAN is the settlement currency
- Liquidity fee is proportional to trade slip and has the correct incentives
- All pools are in a single contract, and
- Stakes are tracked in the contract, and not with an ERC-20.

Theory

Tokens on one side of the pool are bound to the tokens on the other. We can now determine the output given an input and pool depth.

$$X * Y = K$$

$$\frac{y}{Y} = \frac{x}{x+X} \;=>\; y = \frac{xY}{x+X}$$

DEFINITIONS

X	*Balance of TKN in the input side of the pool*	*x*	*Input*
Y	*Balance of TKN in the output side of the pool*	*y*	*Output*
K	*Constant*		

Prices and Slip

We can now determine the expected slip trade and the pool, based only on the input and the depth of the input side of the pool.

$$P_0 = \frac{X}{Y}, \quad P_1 = \frac{X+x}{Y-y}$$

$$outputSlip = \frac{x/P_0 - t}{x/P_0} = 1 - \frac{Xy}{xY} = \frac{x}{x+X}$$

$$poolSlip = \frac{P_1 - P_0}{P_0} = \frac{xY + Xy}{XY - Xy} = \frac{x(2X+x)}{X^2}$$

DEFINITIONS

P0	*Starting Price*	*outputSlip*	*Slip of the output compared to input*
P1	*Final Price*	*poolSlip*	*Slip of the pool after the output is removed*

Example:

$0.10	CAN	TKN	$0.83	
State0	100	12	8.33333333333333	**Price**
Trade in ->	100000	11.99	99.90%	**outputSlip**
State1	100100.00	0.01	8350008.33	**Final Price**
		*	100200000%	**poolSlip**

Liquidity Fee

Stakers stake symmetrically and earn liquidity fees, which is proportional to slip. Slip is proportional to trade size and liquidity depth. Thus staking is incentivised in pools with out-sized trades.

$$liqFee = tradeSlip * tokensOutputted$$

$$liqFee = \frac{x}{x+X} * \frac{xY}{x+X} = \frac{x^2Y}{(x+X)^2}$$

$$tokensEmitted = tokensOutputted - liqFee$$

$$tokensEmitted = \frac{xY}{x+X} - \frac{x^2Y}{(x+X)^2}$$

$$tokensEmitted = \frac{xYX}{(x+X)^2}$$

$$swapSlip = \frac{xY/X - tokensEmitted}{xY/X} = \frac{x(2X+x)}{(x+X)^2}$$

DEFINITIONS

tokensOutputted	*Tokens outputted from the formula before the fee is applied.*	*outputSlip*	*The slip of price between input and output*
tokensEmitted	*Tokens emitted from the pool after the fee is applied.*	*swapSlip*	*The slip of price between input and emission*
		poolSlip	*The slip of price in the pool after the swap*

Thus the final price that the user receives, as well as the final price of the pool, are:

$$Price_{user} = Price_0 * (1 - swapSlip)$$

$$Price_{pool} = Price_0 * (1 - poolSlip)$$

Atomic Swap Calculations

We have a single pool, TKN1, paired to CAN. We wish to swap TKN1 to CAN.

DEFINITIONS

X	*Balance of TKN1 in the input side of the pool*	x	*Input of TKN1*
Y	*Balance of CAN in the output side of the pool*	y	*Output of CAN*

$$tokensOutputted = \frac{xY}{x+X} \qquad outputSlip = \frac{x}{x+X}$$

$$liqFee = \frac{x^2Y}{(x+X)^2}$$

$$swapSlip = \frac{x(2X+x)}{(x+X)^2} \qquad poolSlip = \frac{x(2X+x)}{X^2}$$

$$tokensEmitted = \frac{xYX}{(x+X)^2}$$

	TKN1	CAN		
State0	12000	300000	0.04	**Price**
Trade in ->	70	1739.85	0.58%	**outputSlip**
Liquidity Fee		10.09	Retained as Fee	**$0.40**
TokensEmitted		1729.76	1.16%	**swapSlip**
State1	12070.00	298260.15	0.04	**Final Price**
			1.17%	**poolSlip**

Atomic swaps over two pools:

We have a two pools, TKN1 & TKN2, both paired to CAN. We wish to swap TKN1 to TKN2.

DEFINITIONS

X	*Balance of TKN1 in the input side of the pool*	x	*Input of TKN1*
Y	*Balance of CAN in the output side of the pool*	y	*Output of CAN*
C	*Intermediary Input Balance (CAN)*		
Z	*Final Output Balance (TKN2)*	z	*Final Output*

$$swapSlip_1 = \frac{x(2X+x)}{(x+X)^2} \quad liqFee_1 = \frac{x^2Y}{(x+X)^2} \qquad int_{emission} = y = \frac{xYX}{(x+X)^2}$$

Then:

$$swapSlip_2 = \frac{y(2C+y)}{(y+C)^2} \quad liqFee_2 = \frac{y^2Z}{(y+C)^2} \qquad tokensEmitted_2 = \frac{yZC}{(y+C)^2}$$

Using just the pool depths, and the input, we can calculate the final output and slip:

$$tokensEmitted_2 = z = \frac{xXYCZ(x+X)^2}{(xXY+Cx^2+2CxX+CX^2)^2}$$

$$P_{X0} = \frac{X}{Y}, \quad P_{Z0} = \frac{C}{Z}, \quad P_0 = P_{X0} * P_{Z0} = \frac{XC}{YZ}$$

$$finalSlip = \frac{x/P_0 - z}{x/P_0} = \frac{\frac{xYZ}{XC} - z}{\frac{xYZ}{XC}} = 1 - \frac{C^2X^2(x+X)^2}{(C(x+X)^2 + xXY)^2}$$

	TKN1	CAN1		CAN2	TKN2			
State0	12000	300000	0.04	300000	1500	200	**Price**	**$8.00**
Trade in ->	50	1244.81	0.41%	1239.65	6.17	0.41%	**outputSlip**	
Liquidity Fee		5.17	$0.21		0.03			**$0.20**
Output		1239.65	0.83%		6.15	0.82%	**swapSlip**	**$0.41**
State1	12050.00	298760.35	0.04	301239.65	1493.85	201.65	**Final Price**	
Final	1.00		117614025	191325321	6.15			
			188182440	191325321	0.0164	1.64%	**finalSlip**	

Pool Share

Stakers stake assets to earn a share of the pool. Stake average is the average of their two stakes (of CAN and TKN) at the time they staked.

DEFINITIONS

C	*Balance of CAN in the pool*	*T*	*Balance of TKN in the pool*
stakeC	*Stake of CAN*	$stake_T$	*Stake of TKN*
poolFeesCAN	*Total accumulated fees in CAN*	*poolFeesTKN*	*Total accumulated fees in TKN*
$stakeAve_{Xi}$	*Averaged stake for staker X*	$stakeAve_X$	*Sum of averaged stakes for staker X*
$poolStake_X$	*Sum of all stakes for staker X*	*poolTotal*	*Sum of pool stakes for all stakers*
$poolShare_X$	*Share of the pool for staker X*	*poolShareX*	*Share of the pool for staker X*
CANFeesX	*Share of the CAN fees for staker X*	*TKNFeesX*	*Share of the TKN fees for staker X*
CANStakeX	*Share of the CAN fees for staker X*	*TKNStakeX*	*Share of the TKN fees for staker X*

$$stakeAve_{Xi} = \left(\frac{stake_C}{C + stake_C} + \frac{stake_T}{T + stake_T}\right) * \frac{1}{2}$$

$$poolStake_{Xi} = stakeAve_{Xi} * (T + stake_T)$$

Users can only withdraw their stake partially or fully, or add more. This is tracked as the same all all stakes for that user:

$$poolStake_X = [poolStake_{X0} + poolStake_{X1} + \ldots n]$$

A further number tracks the sum of every averaged stake from every user that has been made into the pool, including the first:

$$poolTotal = \sum_{i=0}^{n} poolStake_i \quad \text{or } poolTotal = [poolStake_X + poolStake_Y + \ldots n]$$

At any stage, any user's share of the pool and its fees is thus given by the proportion between their averaged stake and the pool total:

$$poolShare_X = \frac{poolStake_X}{pool_{total}}$$

Thus when a user *X* withdraws either the fees or their share of the pool, the following are the equations:

$$TKNFees_X = poolShare_X * poolFees_{TKN}$$
$$CANFees_X = poolShare_X * poolFees_{CAN}$$
$$TKNStake_X = poolShare_X * bal_{TKN}$$
$$CANStake_X = poolShare_X * bal_{CAN}$$

For simplicity, stakes and fees can be set to be fully distributed when a staker withdraws.

Example:

	C	T	Notes	
Stake0	100	100	1	Price
PoolShare			100.000	
Stake-UserX	200	100		
StakeAve	0.67	0.50	58.33%	
PoolShare	175.00	116.67	116.667	216.667
State1	300.00	200.00	1.50	
Stake-UserY	200	50		
StakeAve	0.40	0.20	30.00%	
PoolShare	150.00	75.00	75.000	291.667
State2	500.00	250.00	2.00	
UserX - Withdraw Stake	200.0	100.0	0.40	
UserY - Withdraw Stake	128.6	64.3	0.26	
User0 - Withdraw Stake	171.4	85.7	0.34	
	500.0	250.0	1.0	

Tracking Pool Metrics

A single smart contract holds all information regarding all pools.

DEFINITIONS

bal_{CAN}	*Total balance of CAN in the contract*	*CANpooli*	*Balance of CAN reserved for pool i*
balTKN1	*Total balance of TKN1 in the contract*	*TKNpooli*	*Balance of TKN reserved for pool i*

The total balance of CAN represents the total balance of CAN held for each pool which matches the total balance held in the contract.

The balance for each pool is held separately, and is tracked across a static variable. It should equal the balance of the contract. The balance of every pool is the sum of all stakes and inputs, minus all emissions.

$$bal_{CAN} == self.balance(CAN)$$

$$bal_{CAN} = [CAN_{pool_1} + CAN_{pool_2} + \dots n],$$

$$CAN_{pool1} = \sum_{i=0}^{n} stake_i + \sum_{i=0}^{n} inputs_i - \sum_{i=0}^{n} emissions_i$$

The balance for each pool is held separately, and is tracked across a static variable. The balance of every pool is the sum of all stakes and inputs, minus all emissions.

$$bal_{TKN1} == TKN_{pool1}$$
$$TKN_{pool1} = \sum_{i=0}^{n} stake_i + \sum_{i=0}^{n} inputs_i - \sum_{i=0}^{n} emissions_i$$

Thus when a trade across a pool occurs (CAN -> TKN), CAN and TKN balances are changed atomically:

$CAN_{pool1_1} = CAN_{pool1_0} + x,$
and $bal_{CAN_1} == bal_{CAN_0} + x$

$TKN_{pool1_1} = TKN_{pool1_0} - y,$
and $bal_{TKN_1} == bal_{TKN_0} - y$

Across two pools, (TKNA -> TKNB via CAN), all balances are changed atomically:

$TKN_{pool1_1} = TKN_{pool1_0} + x,$
and $bal_{TKN1_1} == bal_{TKN1_0} + x$

$CAN_{pool1_1} = CAN_{pool1_0} - y,$
$CAN_{pool1_1} = CAN_{pool1_0} + y,$
and $bal_{CAN_1} == bal_{CAN_0}$

$TKN_{pool1_1} = TKN_{pool1_0} - z,$
and $bal_{TKN1_1} == bal_{TKN1_0} - z$

The fees are recorded in a global variable, which sums up all transaction fees for that pool, minus every time they were distributed by stakers. A global variable tracks the total fees for that pool every accumulated.

$poolFees_{TKN1} = \sum_{i=0}^{n} liqFee_{TKN1_i} - \sum_{i=0}^{n} FeesDistributed_{TKN1_i}$, where

$poolFees_{TKN1} = \sum_{i=0}^{n} liqFee_{TKN1_i}$

Since CAN is held in a global balance, an array of fees for each pool must also be tracked:

$poolFees_{CAN_{1_i}} = \sum_{i=0}^{n} liqFee_{CAN_{1_i}} - \sum_{i=0}^{n} FeesDistributed_{CAN_{1_i}}$, where

$poolFees_{CAN_1} = [liqFee_{CAN_{1_0}} + liqFee_{CAN_{1_1}} + \ldots n]$, for n transactions, and

$poolFees_{CAN} = [liqFee_{CAN_1} + liqFee_{CAN_2} + \ldots n]$, for n pools.

Arbitrage

Arbitrage is important. A premium value is calculated compared the external price to the internal price:

$$P_1 = P_M \qquad P_1 = \frac{X_1}{Y_1}$$

$$Y_1 = Y_0 - y, \qquad X_1 = X_0 + x$$

$$P_M = \frac{X_0 + x}{Y_0 - y} \qquad tokensEmitted = \frac{xYX}{(x+X)^2}$$

$$\frac{X + x}{Y - \frac{xYX}{(x+X)^2}} = P_M$$

$$1 - \frac{X + x}{PY - \frac{xYX * P}{(x+X)^2}}$$

The real root of x is:

x = -(-2 P^3 Y^3 + 3 P^2 X Y^2 + 6 P^2 Y^3 + sqrt(81 P^4 X^2 Y^4 - 108 P^3 X^3 Y^3 - 270 P^3 X^2 Y^4 + 540 P^2 X^4 Y^2 - 108 P^2 X^3 Y^3 + 297 P^2 X^2 Y^4 + 108 P X^5 Y - 324 P X^4 Y^2 + 324 P X^3 Y^3 - 108 P X^2 Y^4) - 24 P X^2 Y + 3 P X Y^2 - 6 P Y^3 - 2 X^3 + 6 X^2 Y - 6 X Y^2 + 2 Y^3)^(1/3)/(3 2^(1/3)) + (2^(1/3) (-3 (P X Y - X^2 - 2 X Y) - (P Y - 2 X - Y)^2))/(3 (-2 P^3 Y^3 + 3 P^2 X Y^2 + 6 P^2 Y^3 + sqrt(81 P^4 X^2 Y^4 - 108 P^3 X^3 Y^3 - 270 P^3 X^2 Y^4 + 540 P^2 X^4 Y^2 - 108 P^2 X^3 Y^3 + 297 P^2 X^2 Y^4 + 108 P X^5 Y - 324 P X^4 Y^2 + 324 P X^3 Y^3 - 108 P X^2 Y^4) - 24 P X^2 Y + 3 P X Y^2 - 6 P Y^3 - 2 X^3 + 6 X^2 Y - 6 X Y^2 + 2 Y^3)^(1/3)) + 1/3 (P Y - 2 X - Y);

This is a significantly complex solution, which means arbitrage will only be an approximate science in practise.

A significantly easier solution is to attempt to slip the price in the opposite direction but same amount as the premium, using the trade slip, instead of the pool slip. Under 20% premiums, this results in a final price within 1% of the desired price.

$$P_1 = P_M$$

$$P_1 = \frac{X_1}{Y_1}$$

$$Y_1 = Y_0 - tradeSlip, \qquad X_1 = X_0 + x$$

$$\frac{X_0 + x}{Y_0 - y} = P_M \qquad y = \frac{xY_0}{x + X_0}$$

$$\frac{X_0 + x}{Y_0 * P_M - \frac{xY_0 * P_M}{x + X_0}}$$

$$x = \sqrt{X}(\sqrt{P}\sqrt{Y} + \sqrt{X})$$

Example

CAN Price	CAN	TKN	Price	Market	Premium	Value
$0.10	2000	220	9.091	10	10.000%	$1.00
	97.62	9.76	10.0000%			
State1	2097.62	210.24	9.977	$10.00	-0.2%	
	9.76		$1.00			