An Analysis of Uniswap and Senswap Models

Anh-Tuan Nguyen, Tu Phan, Thanh-Phuong Nguyen
{tuannguyen, tuphan, phuong nguyen}@descartes.network

November 8, 2020

1 Logarithmic market scoring rules

Let $q = (q_1, \ldots, q_n)$ where q_i represents the number of outstanding tokens for the token category i. The Logarithmic market maker keeps track of the cost function as follows:

$$C(q) = b \ln \sum_{i=1}^{n} \exp^{q_i}$$

and the price function of each token is described by

$$P_i(q) = \frac{\partial C(q)}{\partial q_i} = \frac{e^{q_i/b}}{\sum_{j=1}^{n} e^{q_j/b}}$$

An automated market makers using LMSR is described as follows:

Assume that b = 1 and the patron sets up an automated market marker $q_0 = (1000, 1000)$ by depositing 1000 coins of token A and 1000 coins of token B. The initial market cost is $C(q_0) = \ln(\exp^{1000} + \exp^{1000}) = 1000.693147$. The instantaneous prices for a coin of tokens are

$$P_A(q_0) = \frac{\exp^{1000}}{\exp^{1000} + \exp^{1000}} = 0.5$$

Similarly, $P_B(q_0) = 0.5$. The relative price between A and B equal to $P_A(q_0)/P_B(q_0) = 1$. If a trader uses 0.689772 coins of token B to buy 5 coins of token A from market q_0, then the market moves to a state $q_1 = (995, 1000.689772)$ with a total market cost $C(q_1) = 1000.693147 = C(q_0)$. The instantaneous prices for a coin of tokens in q_1 are $P_A(q_1) = 0.00336897524$ and $P_B(q_1) = 295.8261646$.. Now a trader can use 0.0033698 coins of token B to purchase 995 coins of token A from the automated market maker q1 with a resulting market maker state $q_2 = (0, 1000.693147)$ and a total market cost $C(q_2) = 1000.693147 = C(q_0)$. The instantaneous prices for a coin of tokens in market maker q_2 are $P_A(q_2) = 2.537979907 \times 10^{-435}$ and $P_B(q_2) = 1$.

This shows that LMSR based automated market maker works well only when the outstanding shares of the tokens are evenly distributed (that is, close to 50/50). When the outstanding shares of the tokens are not evenly distributed, a trader can purchase all coins of the token with lesser outstanding shares and let the price ratio $P_A(q)/P_B(q)$ change to an arbitrary value with a negligible cost.

2 Uniswap Bounded Loss

Uniswap is an automated market maker based on *constant function market maker*. In this model, the cost function is described as follows:

$$C(x) = x_1 * x_2 = k, \text{where k is a constant.}$$

For this market, the price function for each token is defined as:

$$P_1(x) = \frac{\partial C(x)}{\partial x_1} = x_2.$$

$P_2(x)$ is defined in similar way. Hence, if x_1 grow up as fast then the price of x_2 as small.

No-arbitrage conditions: The marginal price of coin α in Uniswap is defined as the price of an infinitesimally small trade. In other words, the marginal price is the exchange rate at the trading time. This price is described by the following formula:

$$\frac{d\Delta_2}{d\Delta_1} = \frac{1}{\gamma}\frac{x_2}{x_1} = \gamma^{-1}m_u,$$

where $m_u = x_2/x_1$. From the above formula, we see that $d\Delta_2 = \gamma^{-1}m_u d\Delta_1$. Thus, we can always make a nonzeoro profit if the Uniswap marginal price of α, $\gamma^{-1}m_u$ is smaller than the market marginal price m_p. In other words, the following inequality holds: $m_p\Delta_1 \geq \gamma^{-1}m_u\Delta_1$ after performing a small enough trade. Assume there is no arbitrage, this means we must have:

$$m_u \geq \gamma m_p.$$

Similarly, by swapping α for β we get the following bounds on the Uniswap market price:

$$\gamma m_p \leq m_u \leq \gamma^{-1}m_p.$$

In the case of no transaction fee, this means $\gamma = 1$, to obtain the no arbitrage conditions, we must have $m_p = m_u$.

2.1 Bounded Loss Condition

Bounded loss is the condition that makes a market maker can control the loss in the worst-case. It is normally defined as:

$$\sup[\max_i(x) - C(x)] < \infty$$

The worse-case loss of a cost function C which begins from initial payout vector x^0 is

$$\sup[\max_i(x_i) - C(x) + C(x^0)]$$

In the Uniswap model, the market maker can't be suffered loss from trading exclude the possible loss from the difference of exchange rate between two tradings at two different times. Therefore, if we want to define a bounded loss for Uniswap, we can regard the payout which pays for traders of the market maker as the number of tokens that are sent for traders after each trading.

With the above-defining payout along with the Uniswap formula, we noticed that the maximum payout that the market maker has to pay for traders in the worst case is to equal with either all reserve of x_1 or x_2. Hence, using the definition of bounded loss condition, we have:

$$\sup[\max_i(x_i) - C(x)]$$

either equal to

$$x_1 - C(x) = x_1 - k < \infty, \text{ with } k \text{ is the constant}$$

or

$$x_2 - C(x) = x_2 - k < \infty, \text{ with } k \text{ is the constant}$$

and the worse-case loss is to equal to either x_1 or x_2.

3 Senswap Bounded Loss

It's similar to Uniswap, our model Senswap is also based on the constant functions class. Concretely, we chose the constant ellipse function for our model rather than the constant product function for its scalability and highly adapted options. The constant-ellipse cost function is defined as follows:

$$C(x) = (x_1 - a)^2 + (x_2 - a)^2 + bx_1x_2 = k.$$

Thus, the price formula for each token has the following form:

$$P_1(x) = \frac{\partial C(x)}{\partial x_1} = 2(x_1 - a) + bx_2.$$

Similarly, $P_2(x) = 2(x_2 - a) + bx_1$. The rate exchange between two coins is

$$P_{12} = \frac{2(x_2 - a) + bx_1}{2(x_1 - a) + bx_2}$$

3.1 Liquidity

We will survey the liquidity of this model. Assume that in the initial state, the reserves for coins A and B respectively are x_1 and x_2. Then, the marginal price between A and B might be

$$P_{12} = \frac{2(x_2 - a) + bx_1}{2(x_1 - a) + bx_2} = k$$

We always expect this rate exchange to be constant number k at all the time. Hence, if there are some trades which happened then the reserves x_1^t, x_2^t should satisfy the following equation:

$$P_{12}^t = \frac{2(x_2^t - a) + b^t x_1^t}{2(x_1^t - a) + b^t x_2^t} = k$$

By transforming, we have

$$a = \frac{x_2(2k - b^t) + x_1(b^t k - 2)}{2k - 2}$$

When the marginal price/ reaches to 1, the value of a can not be defined. Therefore, in some cases, we should fix the value of a and adjust the value of b as desired. But in reality, the same price between two coins is barely occurrence. Thus in this model, we just consider the stable of liquidity based on the change of variable a.

3.2 Bounded Loss and Impermanent Loss

Since Senswap is constructed from constant-ellipse function, it's sure that the Bounded Loss Condition always exists as the same as the Uniswap model. And the remaining part, we should evaluate the impermanent loss of this model at some times.