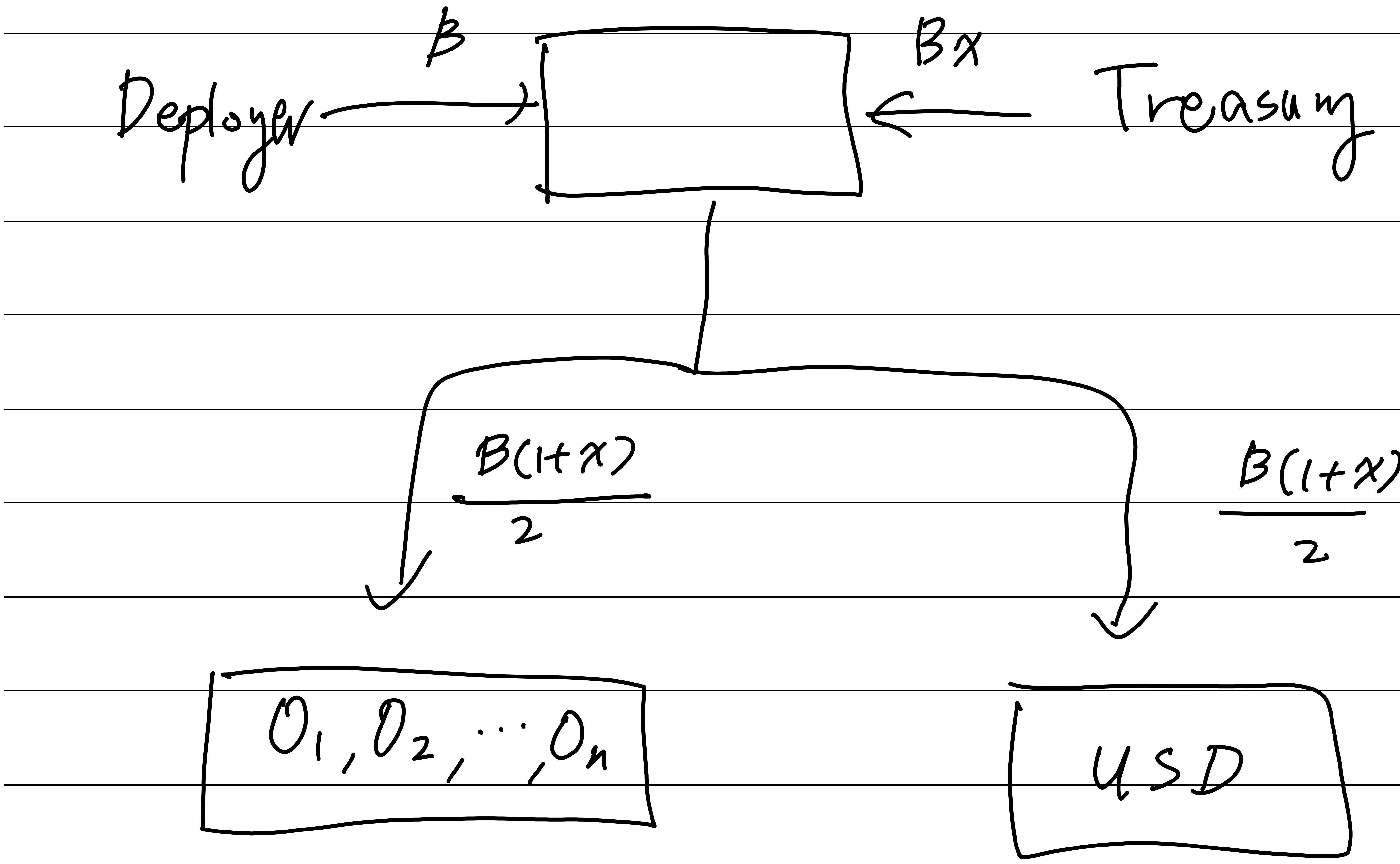
AMM for Prediction Market : CPMM.



$$\frac{B(1+x)}{2}\ O_1 \quad \& \quad \frac{1}{n}\cdot\frac{B(1+x)}{2}\ \text{USD}$$

$$\frac{B(1+x)}{2}\ O_2 \quad \& \quad \frac{1}{n}\cdot\frac{B(1+x)}{2}\ \text{USD}$$

$$\frac{B(1+x)}{2}\ O_3 \quad \& \quad \frac{1}{n}\cdot\frac{B(1+x)}{2}\ \text{USD}$$

$$\vdots$$

Only one Survives

$$\Rightarrow \frac{B(1+x)}{2\sqrt{n}}\ O_i \quad \& \quad \frac{B(1+x)}{2\sqrt{n}}\ \text{USD}$$

$$\frac{B(1+x)}{\sqrt{n}} \ge Bx$$ (Treasury remains profitable)

$$\iff 1+x \ge \sqrt{n}\,x$$

$$\iff x \le \frac{1}{\sqrt{n}-1} = \frac{\sqrt{n}+1}{n-1}$$

e.g.) 2 options $\Rightarrow$ deposit $(\sqrt{2}+1)B \approx 2.41\,B$

4 options $\Rightarrow$ deposit B

9 options $\Rightarrow$ deposit $\frac{B}{2}$

16 options $\Rightarrow$ deposit $\frac{B}{3}$

Generalization

$$Bx \le B(1+x)\sum \sqrt{p_i}\cdot p_i = \mathbb{E}[LP].$$

Since $\sum p_i^{\frac{3}{2}} \ge \frac{1}{\sqrt{n}}$, we are assured that the previous bound is profitable in expectation.

Pool Reserves : $\left(\frac{L_i}{\sqrt{P_i}}, L_i \sqrt{P_i} \right), \ i = 1, 2, \cdots, n.$

Let x be small enough. Trader wants to buy O_i (token for i-th outcome) w/ x USD.

Option 1)

x USD

$\boxed{i\text{-th pool}}$

$\downarrow$

$\frac{x}{p_i} \ O_i$

Option 2)

x USD

$\boxed{\text{mint}}$

$\downarrow$

$(x O_1, x O_2, \cdots, x O_n)$

$\boxed{j\text{-th pool}} \ j \neq i$

$\downarrow$

$(x O_i, \ \sum_{j \neq i} p_j x \text{ USD})$

$\boxed{i\text{-th pool}}$

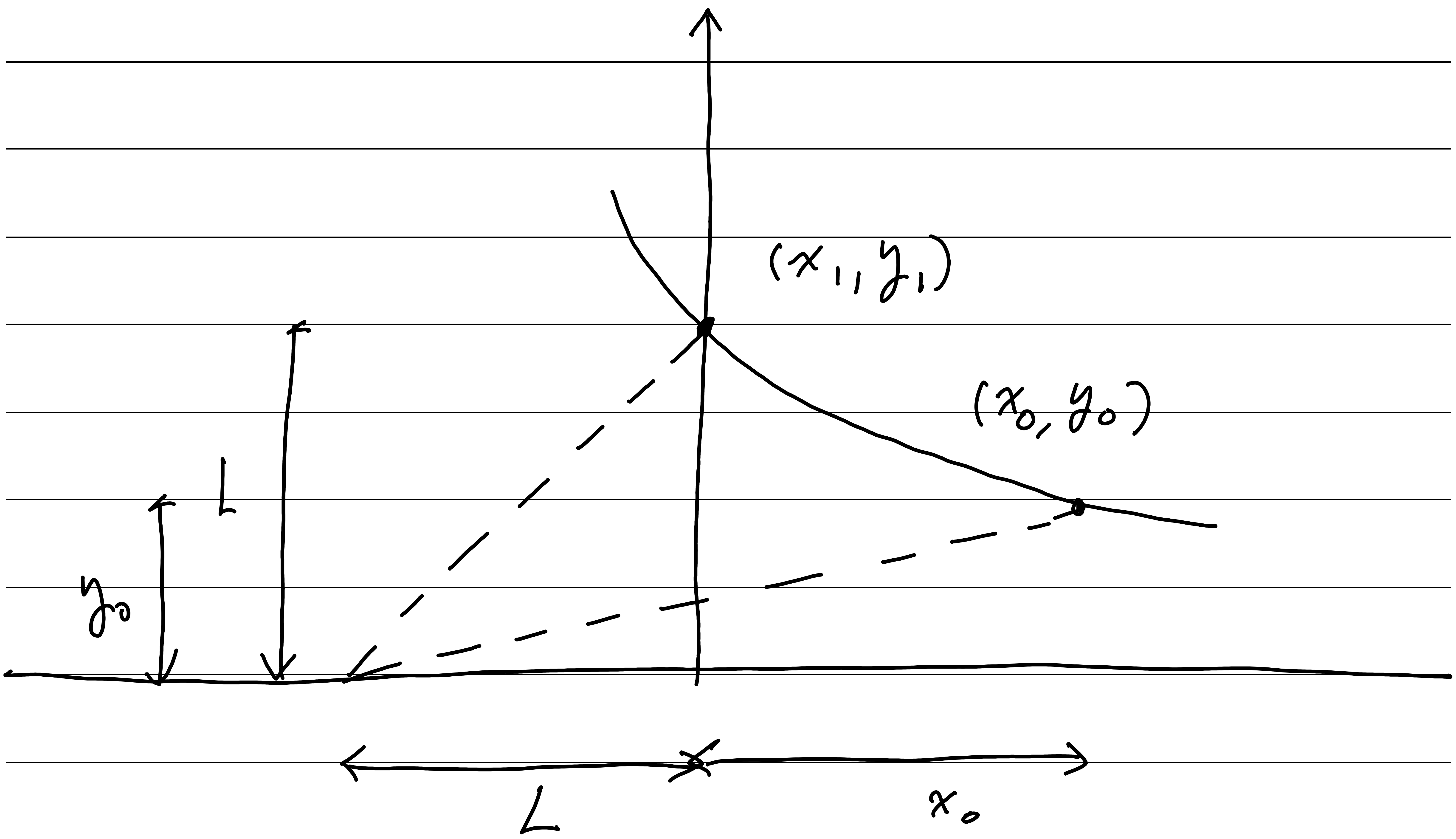
$\downarrow$

$x \left(1 + \frac{1}{p_i} \sum_{j \neq i} p_j x \right) O_i$

Due to the arbitrageurs

$$\frac{x}{p_i} = x \left(1 + \frac{1}{p_i} \sum_{j \neq i} p_j x \right) \Rightarrow \sum_{i=1}^{n} p_i = 1.$$

AMM for Prediction Markets : CLMM.



Main Eq)

$$\left(x + \frac{L}{\sqrt{P_u}}\right)\left(y + L\sqrt{P_l}\right) = L^2$$

Put $(P_u, P_l) \leftarrow (1, 0) \Rightarrow (x+L)y = L^2$

$$\begin{cases} x_0 + L = \dfrac{L}{\sqrt{P}} \\ y_0 = L\sqrt{P} \\ x_1 = 0 \\ y_1 = L \end{cases} \Rightarrow \begin{cases} L = x_0 \cdot \dfrac{\sqrt{P}}{1-\sqrt{P}} \\ y_0 = x_0 \cdot \dfrac{P}{1-\sqrt{P}} \\ x_1 = 0 \\ y_1 = x_0 \cdot \dfrac{\sqrt{P}}{1-\sqrt{P}} \end{cases}$$

$$\text{Deployer} \xrightarrow{?} \boxed{\text{Total:}} \xleftarrow{?} \text{Treasury}$$

$$X\ O_1 \quad \& \quad X \cdot \frac{P_1}{1-\sqrt{P_1}}\ \text{USD}$$

$$X\ O_2 \quad \& \quad X \cdot \frac{P_2}{1-\sqrt{P_2}}\ \text{USD}$$

$$\vdots$$

$$X\ O_n \quad \& \quad X \cdot \frac{P_n}{1-\sqrt{P_n}}\ \text{USD}$$

$$\text{total:}\quad X\ \text{USD} \quad \& \quad X \cdot \sum \left(\frac{P_i}{1-\sqrt{P_i}} \right)\ \text{USD}.$$

$$\mathbb{E}[LP] = X \cdot \sum \left(\frac{P_i\sqrt{P_i}}{1-\sqrt{P_i}} \right)\ \text{USD}.$$

$$\geq X \cdot \frac{\frac{1}{n} \cdot \frac{1}{\sqrt{n}}}{1-\frac{1}{\sqrt{n}}} \cdot n \quad (\text{convexity})$$

$$= X \cdot \frac{1}{\sqrt{n}-1}$$

$$\geq (\text{Payment from Treasury})$$

$$\text{Treasury: } X \cdot \frac{1}{\sqrt{n}-1}$$

$$\text{Deployer: } X \cdot \left(1 + \sum_{i=1}^{n} \frac{P_i}{1-\sqrt{P_i}} - \frac{1}{\sqrt{n}-1} \right).$$

In short, if Deployer's Bid is B and governance's estimation on probability distribution is $(p_i)_{i \in [n]}$, the payment from treasury T should be:

$$T = B \cdot \frac{\frac{1}{\sqrt{n}-1}}{\left(1 + \sum_{i=1}^{n} \frac{p_i}{1-\sqrt{p_i}} - \frac{1}{\sqrt{n}-1}\right)}.$$

X becomes:

$$X = (T+B) \cdot \frac{1}{\left(1 + \sum_{i=1}^{n} \frac{p_i}{1-\sqrt{p_i}}\right)}.$$

Then provide liquiditiy to each pair (O_i, USD) by $\left(X, X \cdot \frac{p_i}{1-\sqrt{p_i}}\right)$. Range will be 0 to 1 with initial price being p_i, which is, the probability of outcome i.

e.g.)

n options, $\left(\frac{1}{n}\right)_{i \in [n]}$

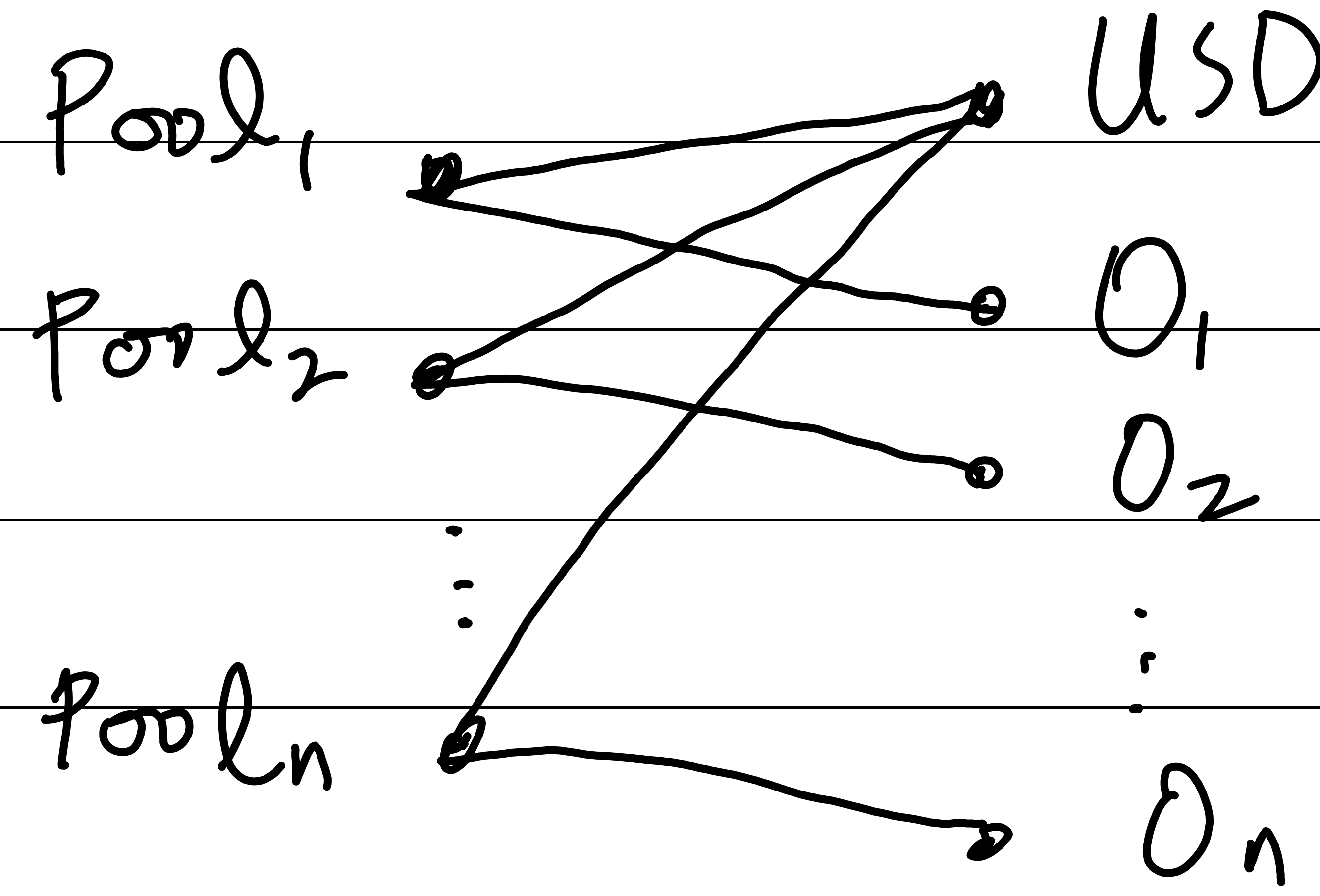
$\Rightarrow \frac{B}{2(\sqrt{n}-1)}$ from Treasury

$\Rightarrow$ find X

$\Rightarrow$ provide liquidity.

Miscellaneous.

[Optimal Routing between CPMMs]



refer Angeris et al. (2022). Can be easily solved by using cvxpy.

[Optimal Routing between CLMMs]

$$x^* = \underset{x}{\operatorname{argmax}} \; Route_1(x) + Route_2(1-x).$$

since R_1 and R_2 are concave, $R_1(x) + R_2(1-x)$ is unimodal or monotonic. Hence x^* can be easily found by using ternary (or golden-section) search.