Le but du document est d'établir des formules concernant les multi-swap dans un échangeur décentralisée ayant des pools de liquidité du type produit constant 50 /50. On souhaites donner des formules explicites pour une condition d'artbitrage d'un cycle de longeur n, la taille du trade optimal et le gain. Pour l'implémentation, il faut réfléchir si ces formules exactes sont plus efficace que des formules itératives mais d'un point de vue théorique obtenir des formules optimales de manière assez clean me semble être assez intéréssant en vue d'optimisation.

Formule pour 1 swap Soit $\mathcal{A}_0$ et $\mathcal{A}_1$ deux tokens et $\mathcal{L}_1$ une pool de liquidité permettant l'échange entre ces deux tokens, on note f les fees de la pool, ceux-ci sont prélévées par rapport au token qui est vendu i.e $\mathcal{A}_0$ ici. On notera ℓ'_{in} le nombre de token de la pool, $\ell_{in} = f \times \ell'_{in}$ et ℓ_{out} le nombre de token de type $\mathcal{A}_1$.

Soit a un nombre de token de type $\mathcal{A}_0$ que l'on souhaite échanger contre $\mathcal{A}_1$. L'algorithme de la pool consiste à dire que lors d'un swap le nombre $\ell'_{in} \times \ell_{out}$ est constant, ceci induit les calculs suivant :

- $k = \ell'_{in} \times \ell_{out}$.
- $a' = a/f$ (les fees sont prélevés sur le token de départ).
- $\text{New}\ell'_{in} = \ell'_{in} + a'$ (apport de a' token dans la pool).
- $\text{New}\ell_{out} = k/\text{New}\ell'_{in}$ (le produit est constant).
- $b = \ell_{out} - \text{New}\ell_{out}$.

On obtient

$$b = \frac{a\ell_{out}}{a + \ell_{in}}$$

est la fonction qui prend en entrée un nombre de token et retourne le nombre de token reçu. Cette fonction est une homographie et nous lui associons la matrice :

$$\begin{bmatrix} \ell_{out} & 0 \\ 1 & \ell_{in} \end{bmatrix}$$

que l'on appelle matrice du swap $\mathcal{A}_0 \to \mathcal{A}_1$ dans la pool $\mathcal{L}$.

formule du n-swap Commençons par introduire les notations :

- On a $n+1$ token que l'on note $\mathcal{A}_0, \ldots, \mathcal{A}_n$.
- On a n pool de liquidité $\mathcal{L}_1, \ldots, \mathcal{L}_n$ tel que pour tout $i \in \{1, \ldots, n\}$ la pool $\mathcal{L}_i$ contient les deux tokens $\mathcal{A}_{i-1}$ et $\mathcal{A}_i$.
- On note f_i le fee de la pool $\mathcal{L}_i$, celui-ci est prélevé sur le token que l'on vend dans la pool i.e $\mathcal{A}_{i-1}$.

- $$\ell_{i,in} = (\text{nombre de token de type } \mathcal{A}_{i-1} \text{ dans } \mathcal{L}_i) \times f_i$$
$$\ell_{i,out} = \text{nombre de token de type } \mathcal{A}_i \text{ dans } \mathcal{L}_i$$

- la matrice de l'échange $\mathcal{A}_{i-1} \to \mathcal{A}_i$ dans la pool $\mathcal{L}_i$:

$$M_i = \begin{bmatrix} \ell_{i,out} & 0 \\ 1 & \ell_{i,in} \end{bmatrix}$$

- La matrice du n-swap $\mathcal{A}_0 \to \cdots \to \mathcal{A}_n$:

$$\mathcal{M}_n = M_n \times M_{n-1} \times \cdots \times M_1$$

- Pour tout $j \in \{1, \ldots, n\}$, on note

$$\Pi_{j,in} = \ell_{1,in} \times \ldots \ell_{j,in} \qquad \Pi_{j,out} = \ell_{1,out} \times \cdots \times \ell_{j,out}$$

Il est clair que la matrice $\mathcal{M}_n$ à la forme :

$$\begin{bmatrix} \Pi_{n,out} & 0 \\ c_n & \Pi_{n,in} \end{bmatrix}$$

Le coefficient c_n est inconnu pour le moment mais on peut obtenir une relation de récurence, en utilisant la définition de $\mathcal{M}_n$. En effet, $\mathcal{M}_{n+1} = M_{n+1} \times \mathcal{M}_n$ et on obtient alors :

$$c_{n+1} = \Pi_{n,out} + c_n \times \ell_{n+1,in} \qquad c_1 = 1$$

On peut alors expliciter c_n en introduisant la suite $\gamma_n = \frac{c_n}{\Pi_{n,in}}$ de sorte que :

$$\gamma_{n+1} = \frac{\Pi_{n,out}}{\Pi_{n+1,in}} + \gamma_n$$

de sorte que :

$$\gamma_n = \gamma_1 + \sum_{j=1}^{n-1} \frac{\Pi_{j,out}}{\Pi_{j+1,in}} \qquad \gamma_1 = \frac{1}{\ell_{1,in}}$$

Posons alors $\Pi_{0,out} = 1$ de sorte que :

$$\gamma_n = \sum_{j=0}^{n-1} \frac{\Pi_{j,out}}{\Pi_{j+1,in}}$$

et on obtient :

$$\boxed{c_n = \Pi_{n,in} \sum_{j=0}^{n-1} \frac{\Pi_{j,out}}{\Pi_{j+1,in}}}$$

Maintenant que l'on a obtenu une forme explicite nous allons étudier un peu les homographies d'un point de vue optimisation.

Optimisation Soit $f = \frac{ax+b}{cx+d}$, alors $f' = \frac{\det(f)}{((cx+d)^2}$. On cherche a optimiser la fonction de gain :

$$g = f - x \qquad\qquad g' = \frac{\det(f)}{((cx+d)^2} - 1$$

On a alors $g' = 0$ si et seulement si $x = x_{opt} := \frac{\sqrt{\det(f)}-d}{c}$ De plus :

$$f(x_{opt}) = \frac{a - \sqrt{\det(f)}}{c}$$

Et

$$f(x_{opt}) - x_{opt} = \frac{\mathrm{tr}(f) - 2\sqrt{\det(f)}}{c}$$

Pour notre application, nous avons que $b = 0$ et l'on peut simplifier les formules :

$$f(x_{opt}) - x_{opt} = \frac{\mathrm{tr}(f) - 2\sqrt{\det(f)}}{c} = \frac{(\sqrt{a} + \sqrt{d})^2}{c} \geq 0$$

Retour sur le n-swap On obtient :

$$x_{opt} = \frac{\sqrt{\Pi_{n,in}\Pi_{n,out}} - \Pi_{n,in}}{\Pi_{n,in} \sum_{j=0}^{n-1} \frac{\Pi_{j,out}}{\Pi_{j+1,in}}}$$

$$x_{opt} = \frac{1}{\sum_{j=0}^{n-1} \frac{\Pi_{j,out}}{\Pi_{j+1,in}}} \left(\sqrt{\frac{\Pi_{n,out}}{\Pi_{n,in}}} - 1 \right)$$

En particulier, on voit que

$$\boxed{x_{opt} \geq 0 \qquad \text{si et seulement si} \qquad \Pi_{n,out} \geq \Pi_{n,in}}$$

ce qui est la condition d'arbitrage et on a :

$$f(x_{opt}) - x_{opt}) = \frac{1}{\sum_{j=0}^{n-1} \frac{\Pi_{j,out}}{\Pi_{j+1,in}}} \left(1 + \sqrt{\frac{\Pi_{n,out}}{\Pi_{n,in}}} \right)^2$$

J'explicite les choses dans le cadre $\mathcal{A}_0 \to \mathcal{A}_1 \to \mathcal{A}_0$. On trouve :

$$x_{opt} = \frac{1}{\frac{1}{\ell_{1,in}} + \frac{\ell_{1,out}}{\ell_{1,in}\ell_{2,in}}} \left(\sqrt{\frac{\ell_{1,out}\ell_{2,out}}{\ell_{1,in}\ell_{2,in}}} - 1 \right)$$

En simplifiant :

$$x_{opt} = \frac{1}{\ell_{2,in} + \ell_{1,out}} \left(\sqrt{\ell_{1,out}\ell_{2,out}\ell_{1,in}\ell_{2,in}} - \ell_{1,in}\ell_{2,in} \right)$$

avec les notations du premier pdf :

$$\ell_{1,in} = A_1 f_1 \qquad \ell_{1,out} = B_1 \qquad \ell_{2,in} = B_2 f_2 \qquad \ell_{2,out} = A_2$$

On trouve :

$$x_{opt} = \frac{\sqrt{B_1 A_2 f_1 A_1 f_2 B_2} - A_1 f_1 B_2 f_2}{B_2 f_2 + B_1}$$