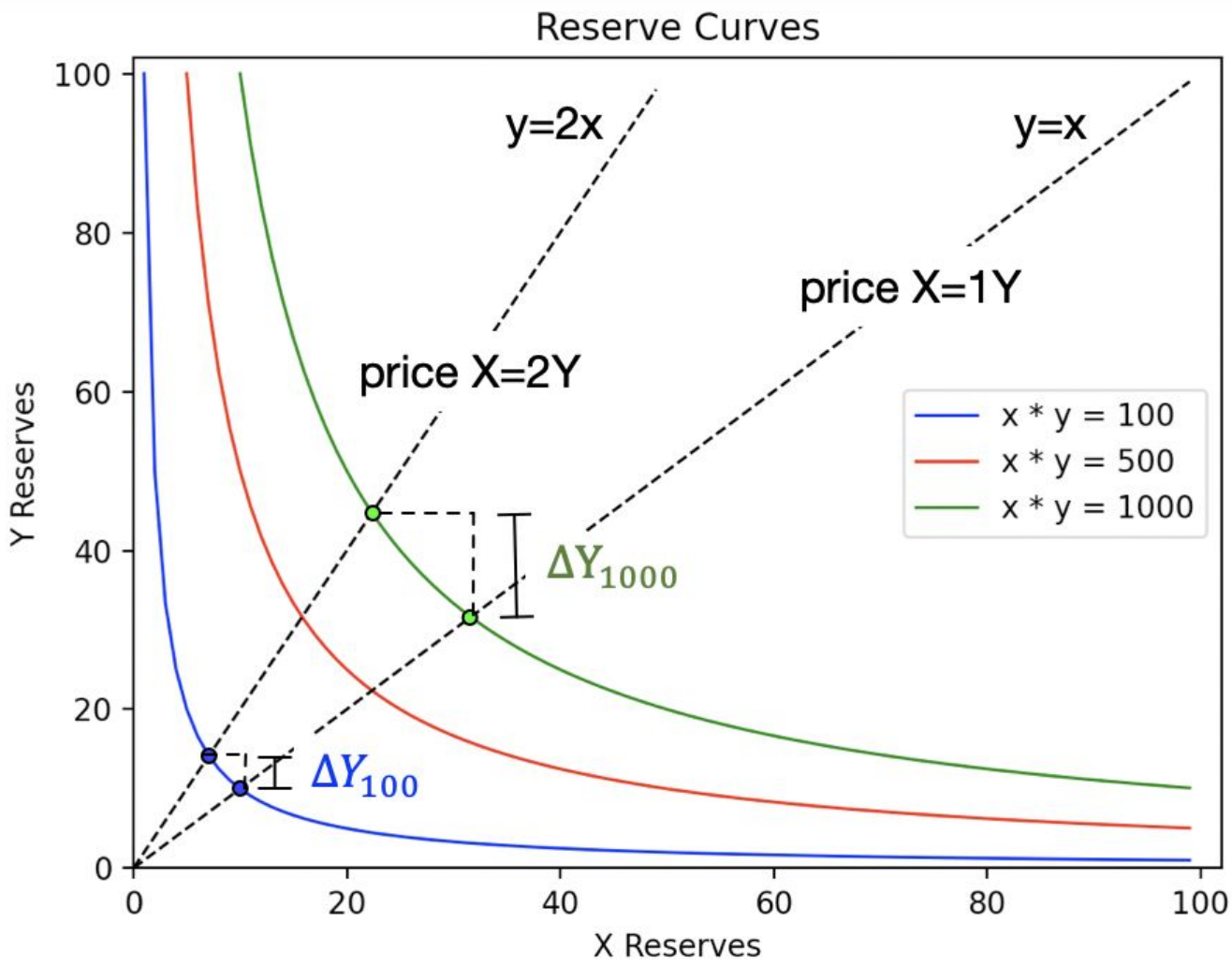


Uniswap v3 设计详解

JasonW
喜欢思考，博而不专的铲屎官

Uniswap v3版本发布也有一段时间了，它的日交易量也逐渐超越了v2版本，之前对v2版本的设计原理进行了一个科普，对Uniswap本身或v2版本的技术原理不清楚的，可以参考Uniswap深度科普，在v3版本中针对v2运行过程中已发现的问题，做了一些改进，这也导致了的v3版本的复杂度有了极大的提升，这篇文章主要针对v3的整体设计做一个介绍。

在协议设计上最核心的改变就是引入了**集中流动性**，来解决v2版本中资金利用率不高的问题，机制本身的设计并不难理解，比较复杂的是整个交易或者计算体系的设计和实现，接下来就从设计原理和具体实现上分别介绍一下，主要参考还是来自于白皮书和开源代码。

v2中资金使用率低的问题

先看一下v2版本中资金使用率低的问题，在v2的版本里流动性沿储备金曲线均匀分布，储备金曲线方程式如下：

$$x \times y = k, \quad k\text{为常数}$$

在这个方程式中，x和y分别代表两种代币X和Y的储备量，除了k是一个常数之外，对x和y的值没有

地聚合流动性，

假设WETH/USDC交易对的流动性池中共有资金：9000 USDC 和 2 WETH，即WETH的价格为4500 USDC

根据 x · y = k（以x代表WETH，y代表USDC），可以计算出：

k = 9000 * 2 = 18000

当WETH价格下降到4000 USDC时，根据 x · y = 18000，y / x = 4000 可以计算出此时USDC的资金约为：

y′ = 8485

资金使用率为：

(9000 - 8485) / 9000 = 5.72%

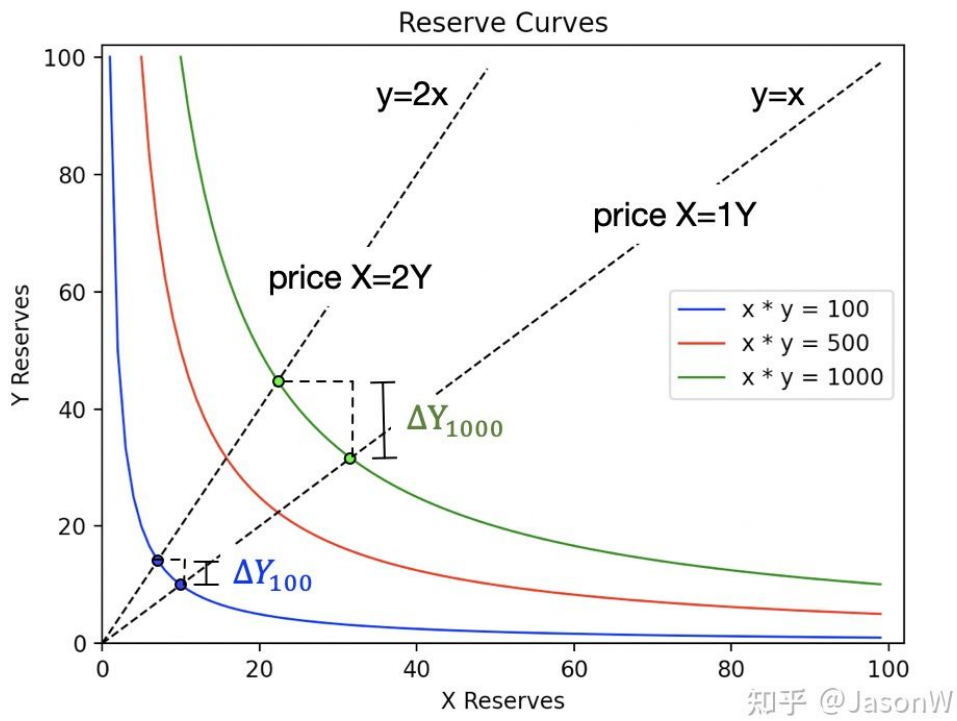
也就是说在价格有较大波动的情况下，资金使用率也没有超过10%，那么这个问题对于稳定币池（如USDC/DAI）来说就会更严重。

集中流动性

为了解决v2中资金使用率的问题，v3中引入了集中流动性，**集中流动性**的定义：流动性提供者（LPs）可以将其提供的流动性"限制"在任意的价格区间内来集中其流动性。

也就是在v3版本里允许LPs把其提供的流动性集中在一段价格区间（position）里，这样不仅能有效的提升资金利用率，同时由于流动性集中在了更小的区间，使得这个价格区间里有更深的交易深度，即更大的流动性。

首先我们来看看更大的流动性有哪些好处，这个原理对v2和v3版本来说都同样适用，下图是不同k值的储备金曲线：

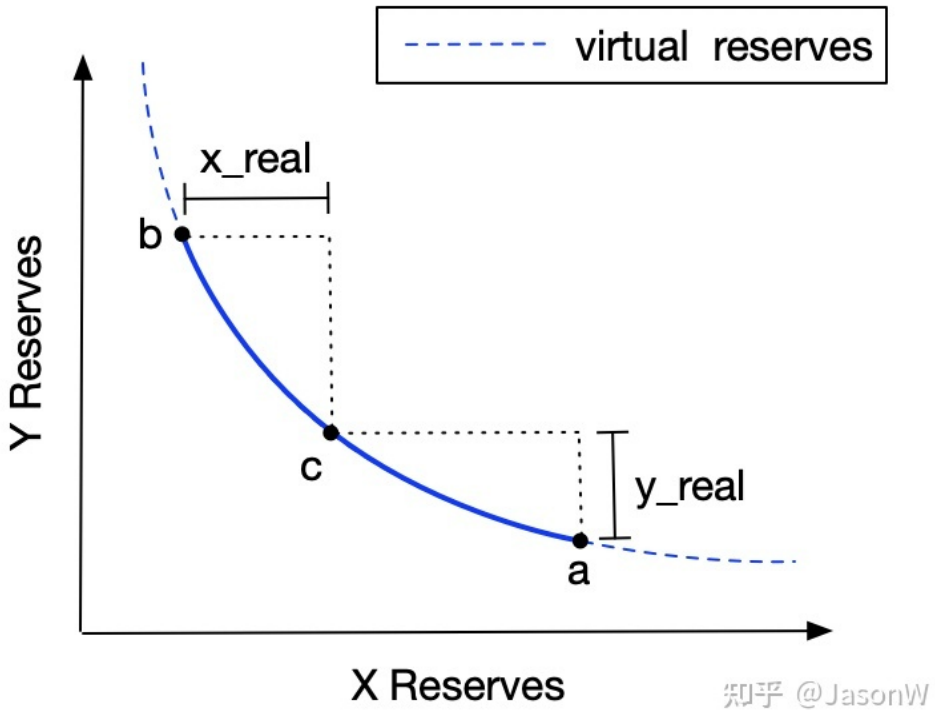


X）使得X的价格上涨，从1Y变成2Y，那么绿色曲线上Y的数量的变化要远大于蓝色曲线上，即：

$$\Delta Y_{1000} > \Delta Y_{100}$$

也就是说在产生相同的价格变化时，绿色曲线上需要的交易量要远远大于蓝色曲线上的交易量，即绿色曲线上的交易深度更大，流动性更好，反向来解释的话，就是相同的交易量之下，绿色曲线上价格的变化要小于蓝色曲线，即用户承受的滑点损失会比较小。

接下来看看分区间提供流动性之后带来的好处，假设只有一个区间时，所有流动性都集中在这一个区间中，如下图所示：



在上图中，用户只在[p_a, p_b]价格区间提供流动性，在这个区间里只需要保证有足够的资产X来覆盖价格变动到上限p_b，有足够的资产Y来覆盖价格变动到下限p_a即可，例如在价格p_c时，用户实际的流动性提供为x_real和y_real，在[a,b]区间所有的点都满足有如下方程式：

$$(x_{real} + x_{virtual}) \times (y_{real} + y_{virtual}) = k$$

其中x_real和y_real是用户真实提供的X token和Y token的数量，x_virtual和y_virtual代表流动性池虚拟出的X token和Y token数量，为了能保证价格计算的一致性 $x \cdot y = k$，并不参与实际的交易，而且只限于[a, b]这个区间，当价格游移出[a, b]这个区间时，x_virtual和y_virtual会被移除。

我们来看一下x_virtual和y_virtual的值，若将Y视为基准币（例如USDC）则价格p定义为：1X = pY，即1单位X可以换取p个Y，x和y分别代表两种代币的数量，则a点和b点的价格满足方程：

$$p_a x_a = y_a \quad \Rightarrow \quad p_a = \frac{y_a}{x_a}$$

$$p_b x_b = y_b \quad \Rightarrow \quad p_b = \frac{y_b}{x_b}$$

用L来代表流动性的量，则:

$$L = \sqrt{k}$$

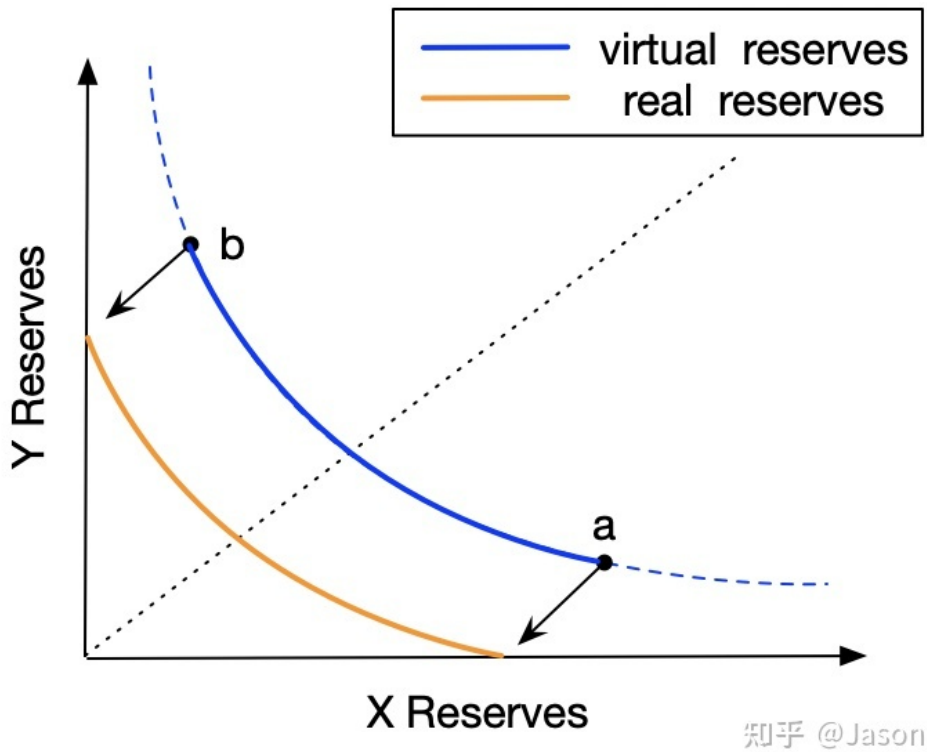
带入到曲线方程中进行一些推导来获得实际的储备金曲线方程：

$$\left.\begin{aligned} \frac{y_a}{x_a} &= p_a \\ x_a \times y_a &= L^2 \end{aligned}\right\} \Rightarrow y_a = L\sqrt{p_a}$$

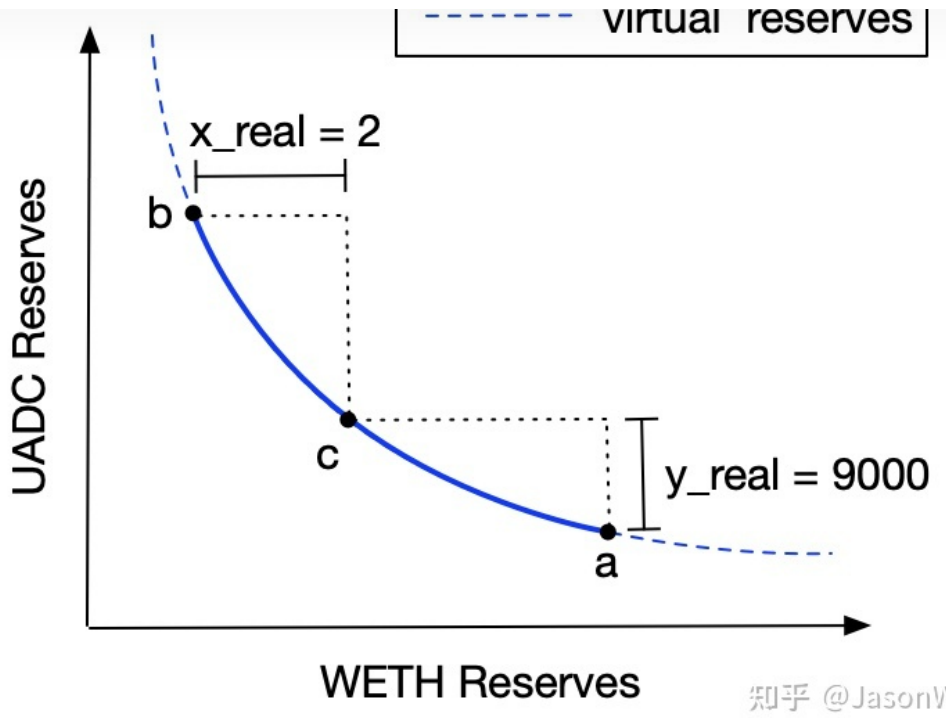
$$\left.\begin{aligned} \frac{y_b}{x_b} &= p_b \\ x_b \times y_b &= L^2 \end{aligned}\right\} \Rightarrow x_b = \frac{L}{\sqrt{p_b}}$$

$$\Rightarrow \quad \left(x_{real} + \frac{L}{\sqrt{p_b}}\right) \times \left(y_{real} + L\sqrt{p_a}\right) = L^2$$

在流动性不改变的情况下，由于p_a和p_b是固定的，所以x_virtual和y_virtual也是固定值，根据上面推导出来的方程，虚拟储备金和实际储备金曲线如下图所示：



公式的推导太抽象了一点，现在我们回到之前提到的 WETH/USDC 交易对池的例子，我们假设把流动性集中在 [4000, 5000] 这个价格区间内，最初提供的WETH和USDC仍然是2 和 9000，则虚拟储备金曲线如下图所示：



其中 p_a = 4000， p_b = 5000，因为在a点y_real = 0，在b点x_real = 0，所以最初提供的x_real和 y_real对应中间的一个点，这样可以计算出虚拟储备金曲线中的流动性：

$$\left(2+\frac{L}{\sqrt{5000}}\right)\times\left(9000+L\sqrt{4000}\right)=L^2 \;=>\; L\approx 2314$$

因此虚拟储备金方程为：

$$x\times y=2314^2=\;5354596$$

对应到实际的储备金方程为：

$$(x_{real}+33)\times(y_{real}+148058)=5354596$$

当价格为 WETH = 4500 USDC 时，可根据虚拟储备金方程计算出:

$$\left.\begin{array}{l} x\times y=5354596 \\ \dfrac{y}{x}=4500 \end{array}\right\} => y\approx 155228$$

则：

$$y_{real}=155228\;-148058\;=7170$$

当 WETH= 4000 USDC 时，即图中的a点，此时y_real = 0，当 WETH = 5000 USDC 时，即图中b点，x_real = 0，则 y_real = 14202

那么WETH价格从4500 USDC变化到4000 USDC时，资金使用率即为:

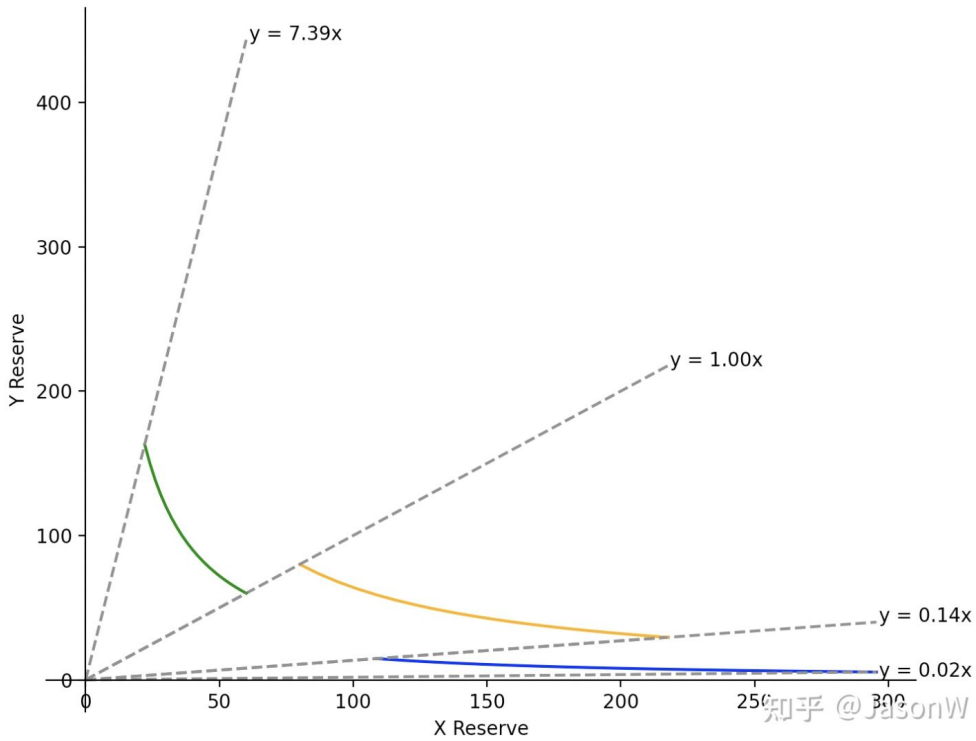
(7170 - 0) / 14202 = 50.49%（此值由于计算过程中取值精度问题会略有误差）

要远大于之前的5.72%，如果进一步把区间缩小到 [4000, 4500]，则资金使用率为 100%。

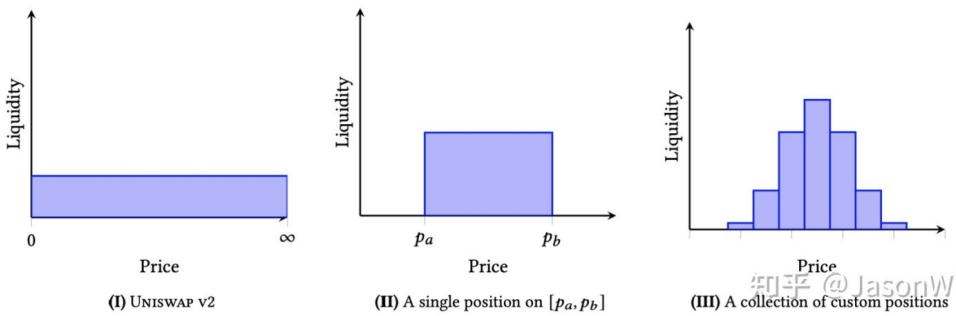
现做一些拆解介绍，但由于篇幅所限，不会对细节做过多解释，有兴趣的读者可以通过开源出来的代码来做更深入的了解。

集中流动性实现设计

集中流动性的引入，使用户可以把资金放到任意价格区间内提供流动性，上文为了方便解释原理，只用了一个区间的场景，实际场景中不太可能只有一个价格区间，假设有几个连续的价格区间各自有不同的流动，则如下如所示：



我们可以看到，储备金曲线不再连续了，在交易过程中，价格变化到不同区间时要分段独立计算，但是在X token-Y token直角坐标系中每个价格区间的边界是一条直线，X token和Y token的量是同时变化的，而且还可能存在价格区间有重叠的情况，这就导致整个的计算过程复杂度非常高，在v3版本中，针对这个问题的解决办法是做了一次坐标系转换，或者说做了一个降维处理，不再用X token - Y token的坐标系了，而是改用 Price - Liquidity的坐标系，这样价格区间的边界就从一个二维的直线变成了Price轴上的一个点了，而且由于增加/减少流动性不会改变当前价格，swap交易过程会导致价格变化但不会改变流动性的值，因此Price和Liquidity在同一时间只有一个值变化，进一步降低了计算复杂度，Uniswap白皮书中的举例也都是在这个坐标轴体系内：



当然，没有绝对完美的解决方案，那么采用Price - Liquidity会带来哪些问题呢？对用户和应用侧来说，还是对 X token和 Y token进行的操作，所以在实际计算过程中要在 (X,Y) <—> (Price, Liquidity)之间做互相的转换，这就涉及到白皮书中非常重要的几个公式：

$$\Delta y = \Delta\sqrt{P} \cdot L \tag{6.14}$$

$$\Delta\frac{1}{\sqrt{P}} = \frac{\Delta x}{L} \tag{6.15}$$

$$\Delta x = \Delta\frac{1}{\sqrt{P}} \cdot L \tag{6.16}$$

基于这些公式，整个计算过程只需要关注流动性L和价格P，为了减少计算过程中开根号的运算，v3使用全局状态存储了价格的平方根sqrt_P，和常数k的平方根L（即流动性）。

Tick和Position

由于v3版本支持用户创建任意的价格区间，所以可能会出现两个价格区间会有部分重叠，也有可能出现两个价格之间没有流动性等等情况，针对这些复杂场景，v3引入tick和position（价格区间）的设计来解决。

与区块链上很多其他的设计或者概念类似，tick并不是Uniswap首创的，还都是沿用的传统金融领域的概念，只不过在实际应用中做了改动和创新，我们看一下Investopedia中是如何定义的：

> **What Is a Tick?**
>
> A tick is a measure of the minimum upward or downward movement in the price of a security. A tick can also refer to the change in the price of a security from one trade to the next trade. Since 2001 and the advent of decimalization, the minimum tick size for stocks trading above $1 is one cent.

Tick也就是证券价格向上或向下变化的最小度量，在Uniswap中为了实现自定义流动性提供，任意的价格区间position就是由离散的ticks来划定的，也就是流动性提供者可以在任意的两个ticks（不必相邻）之间的范围内提供流动性，由于不同币种的价值相差很大，因此用相对的百分比来移动价格区间更为合理，在Uniswap中把价格基准点设置为0.01%，这样每一个价格都是1.0001的整数指数，以i为索引指数，则在每个tick的价格p可表示为：

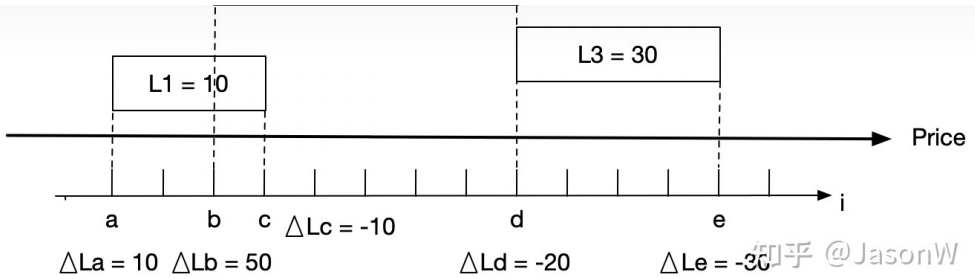
$$p(i) = 1.0001^i$$

$$\mathrm{i} = \lfloor log_{\sqrt{1.0001}}\sqrt{P} \rfloor$$

使用索引指数i来表示一个价格区间的时候比较方便，例如 [100, 101] 就可以表示1.0001的100次方和101次方之间的价格区间，相较于实际价格 [1.0100496621, 1.0101506671] 表示上会简便很多。

这样我们把价格进行了离散化处理，每一个position都对应到两个tick，tick_lower和tick_upper，而且不同的position可以共用同一个tick，但由于v3支持的tick的精度已经很高，所以并不需要记录每个tick的所有信息，只有被选做某个position的价格边界的tick才会被实例化，同时在tick上记录流动性的变化。

这部分在白皮书中介绍的并不是很充分，接下来具象化的解释一下，我们用tick的索引i来表示价格，任意创建几个position，如下图：



我们假设整个交易过程中，价格从左到右（从小到大）依次穿透三个position的边界，并且在a点之前和e点之后没有position，我们来看一下穿过每个点时的相关状态数值的变化：

1. 从左到右穿过a点时，流动性增加L1=10，总流动性为10；
2. 从左到右穿过b点时，流动性增加L2=50，总流动性为60；
3. 从左到右穿过c点时，流动性减少L1=10，总流动性为50；
4. 从左到右穿过d点时，流动性减少L2=50，增加L3=30，净增加-20，总流动性为30；
5. 从左到右穿过e点时，流动性减少L3=30，总流动性为0；

穿过每一个tick时，流动性增加的净值（可为负）记录在tick中，从左到右穿过tick时，增加净值，从右到左穿过tick时，减少净值，通过上图的流程，我们可以看到在实际的swap过程中，是分区间来计算交易的，如 [a, b]，[b, c]，[c, d]，[d, e]，在每个区间里流动性是一定的，通过输入的token数量和价格变化来计算出输出token数量，但这个区间并不一定和某个position重合，有可能是两个不同position的边界，如 [b, c] 和 [c, d]，在每个区间里可以通过白皮书中6.13~6.16的公式进行交易计算，包括手续费的计算等。

总结

以上就是Uniswap v3整个计算体系设计实现的原理，除了引入集中流动性的概念之外，v3版本还支持同一个交易对可以有不同的手续费的流动性池等特性，由于属于附属功能，在此未做过多介绍，那么在v3这些新特性之下，我们可以看看使用上带来哪些不同：

1. 范围订单（range order），如果我们把价格区间限制的非常小，那么范围订单就类似传统order book中的一个限价订单；
2. 流动性提供者可以创建任意大小、任意数量的区间（position），可以根据市场行情来调整流动性分布以获取最大收益。

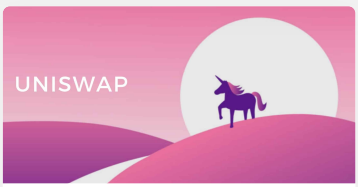
有利就有弊，带来这些好处的同时，同样也带来了一些问题：

1. 协议复杂度提升，使用门槛高；
2. 手续费、收益计算复杂，不易理解；
3. 交易手续费提升。

Uniswap v3版本有很多创新性的尝试，引入集中流动性之后也带来了很多潜在的商机，例如最近出圈的DEX聚合服务商1inch等等，但就目前市场来看，v3版本的发布并没有达到所有人的预期，v2版本依然有很大的交易量，而且可以预见在相当长一段时间内两个版本依然会共存，后续的发展我们拭目以待吧。

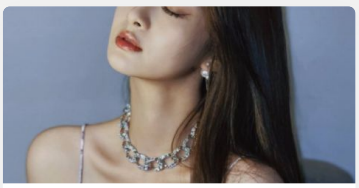
编辑于 2022-01-18 08:50

区块链(Blockchain) DeFi Uniswap



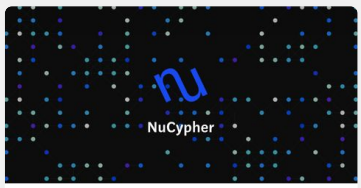
去中心化交易所：Uniswap v2 白皮书中文版

我是个AI



Uniswap V3 到底是什么鬼？一文带你了解V3新特性

一路向北



NuCypher 深度评测：顶级区块链投资机构为何都要钟情它?

周徒子 发表于NPC源计...

还没有评论

写下你的评论...

▲ 赞同 7 ▼ 添加评论 分享 喜欢 收藏 申请转载 ···