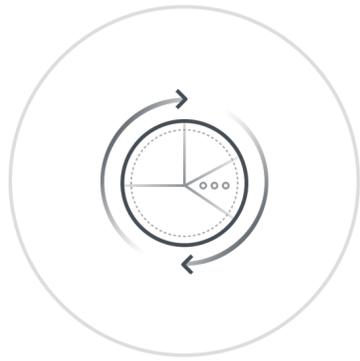


Midaswap的 NFT AMM设计思想



NFT AMM 需求分析



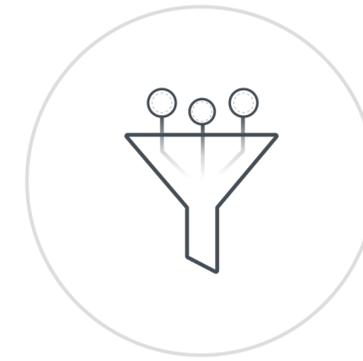
AMM 产品核心要素

流动性利用效率、无偿损失、可组合性



NFT 交易的核心需求

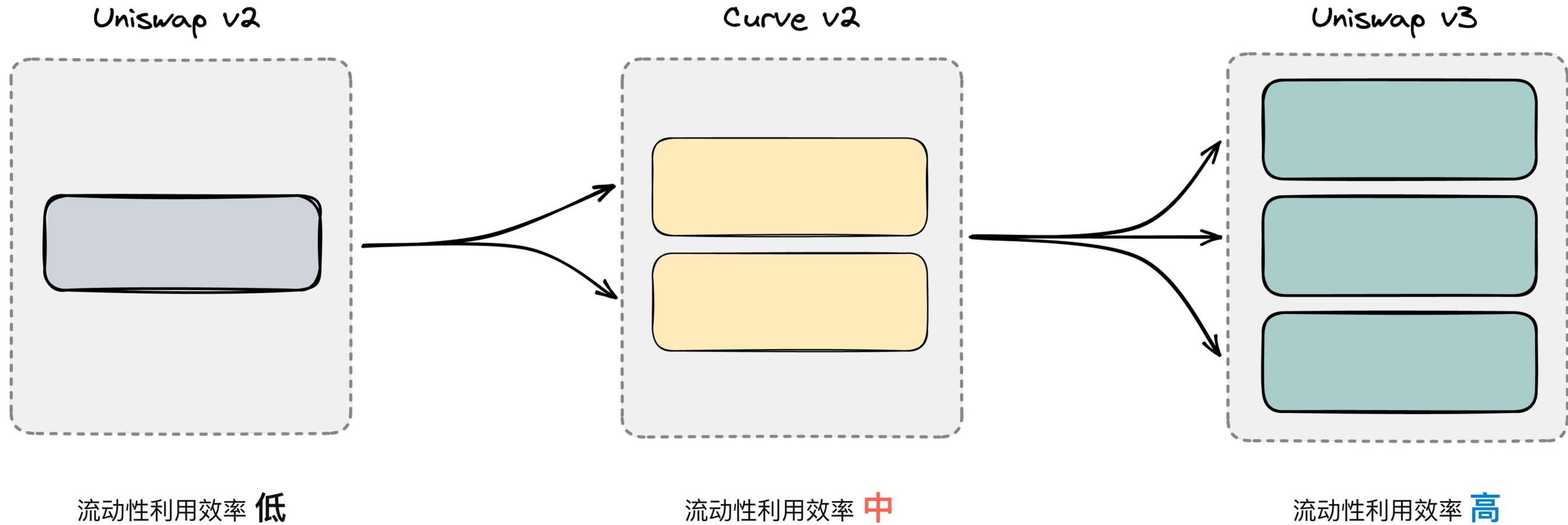
流动性 & Tokenid的唯一性



NFT AMM 产品优先级

极致的流动性利用效率

NFT AMM 算法选型

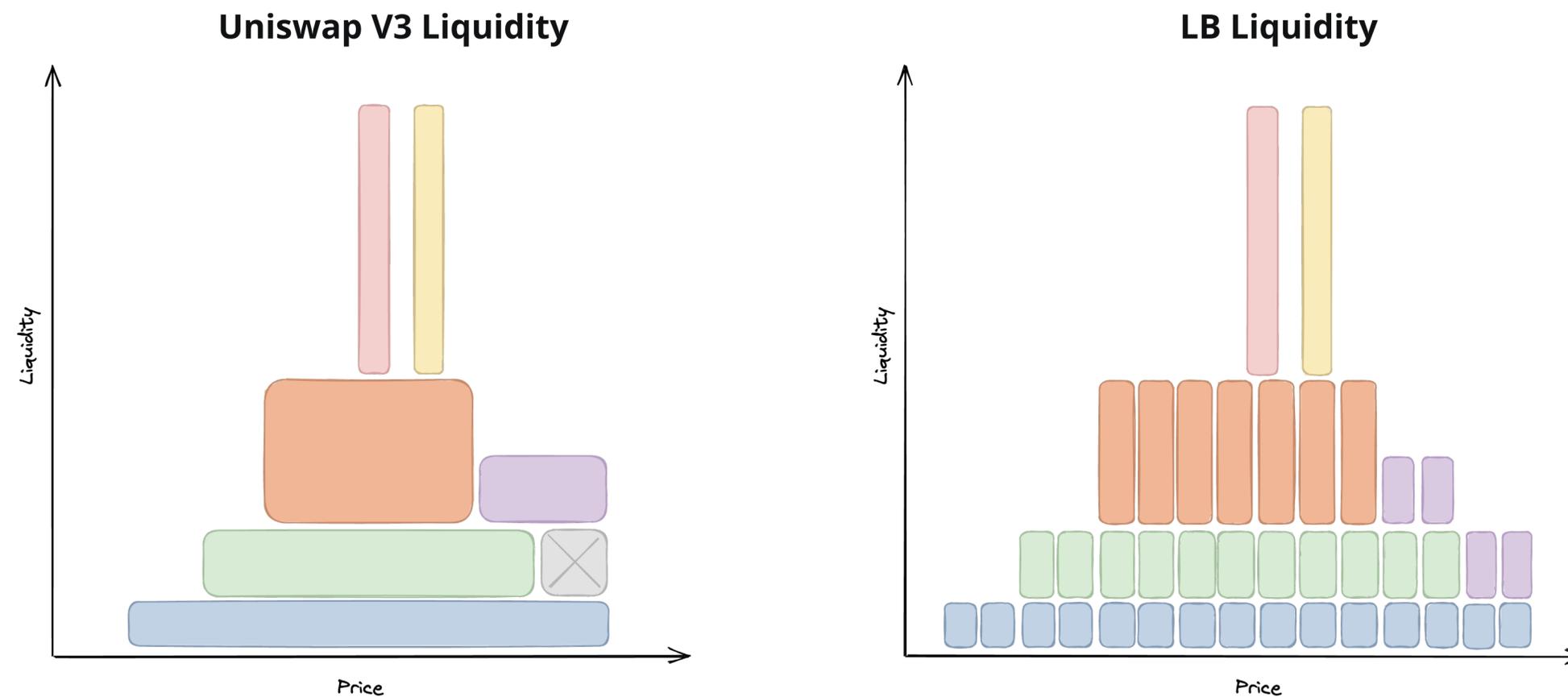


NFT AMM 架构设计

- Uniswap v3 AMM 算法核心: $X*Y=K$ & Tick 不适配NFT交易场景
- Uniswap v3 适合浮点数计算, NFT交易场景是整数计算
- 如果使用碎片化的方案会丢失NFT Tokenid唯一性同时Gas很高

NFT AMM 架构设计

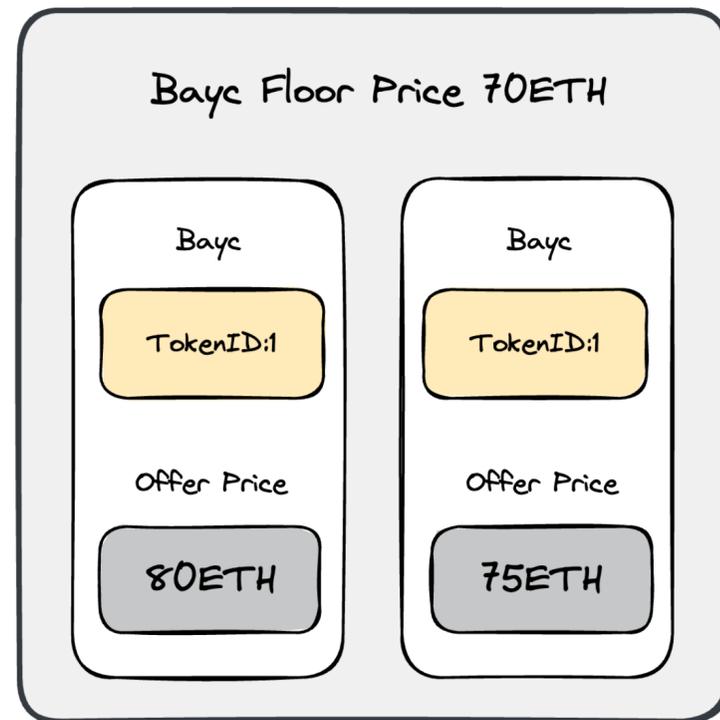
- NFT AMM 需求：既可以继承Uniswap v3 流动性利用效率，也适配NFT交易过程中整数场景
- Liquidity Book：将Uniswap v3 $X*Y=K$ 替换为 $X+Y=K$ ，将Tick替换为Bin



NFT AMM 架构设计

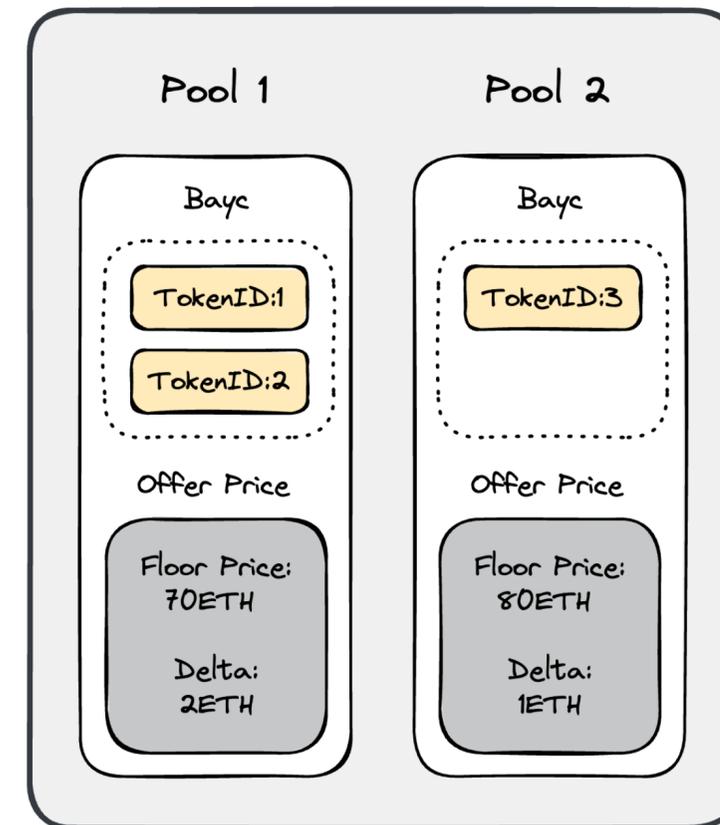
- **Liquidity Book 算法可以解决NFT AMM资产计价问题，但是无法解决NFT Tokenid 唯一性的问题**
- **当LPs添加流动性进Pool时，如何记录LP与提供的NFT流动性Tokenid唯一性的绑定关系**
- **当Trader与Pool交易时，交易后的NFT应该交于哪位LPs**
- **核心问题： NFT Tokenid的结算问题**

NFT AMM 架构设计



Opensea

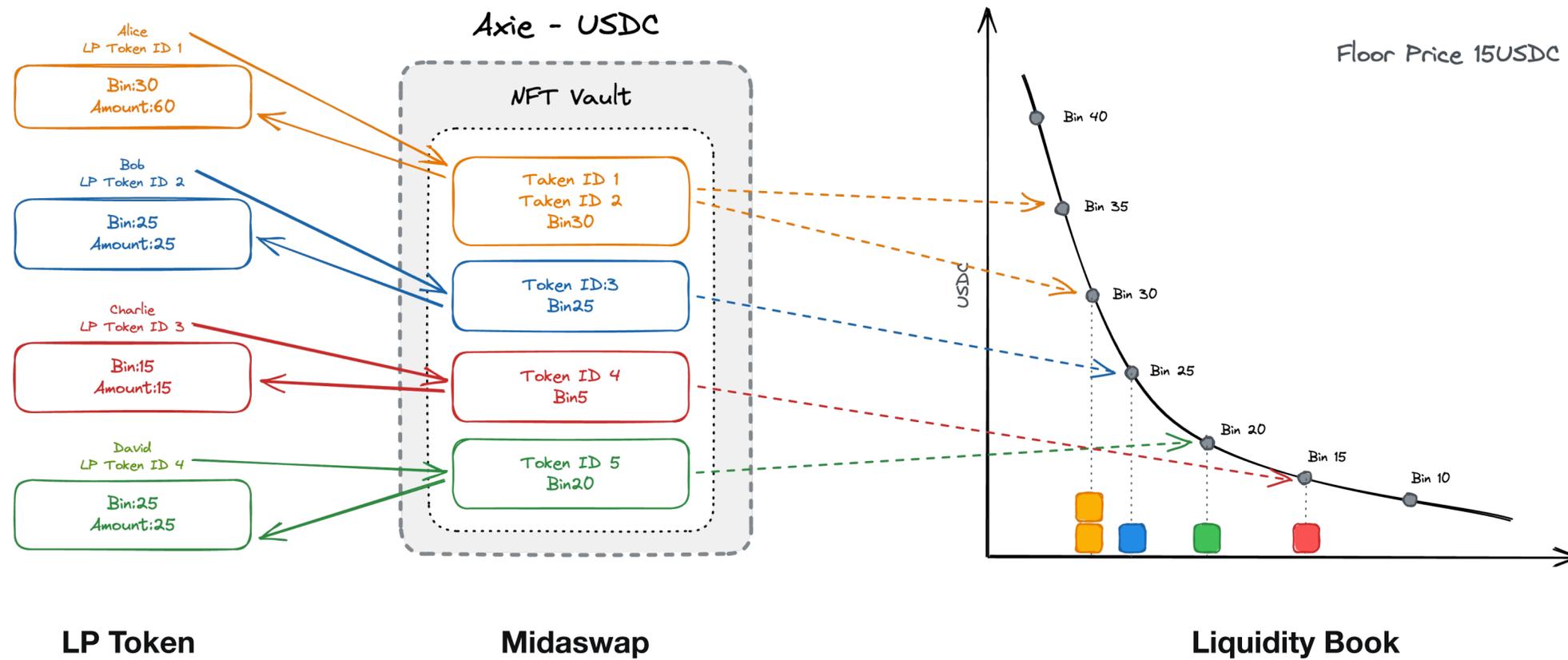
- 采用Tokenid与Trader绑定的方式结算



Sudoswap

- 采用Bonding Curve & 多Pool的方式构建产品
- 使用Pool Address来结算NFT资产

NFT AMM 架构设计

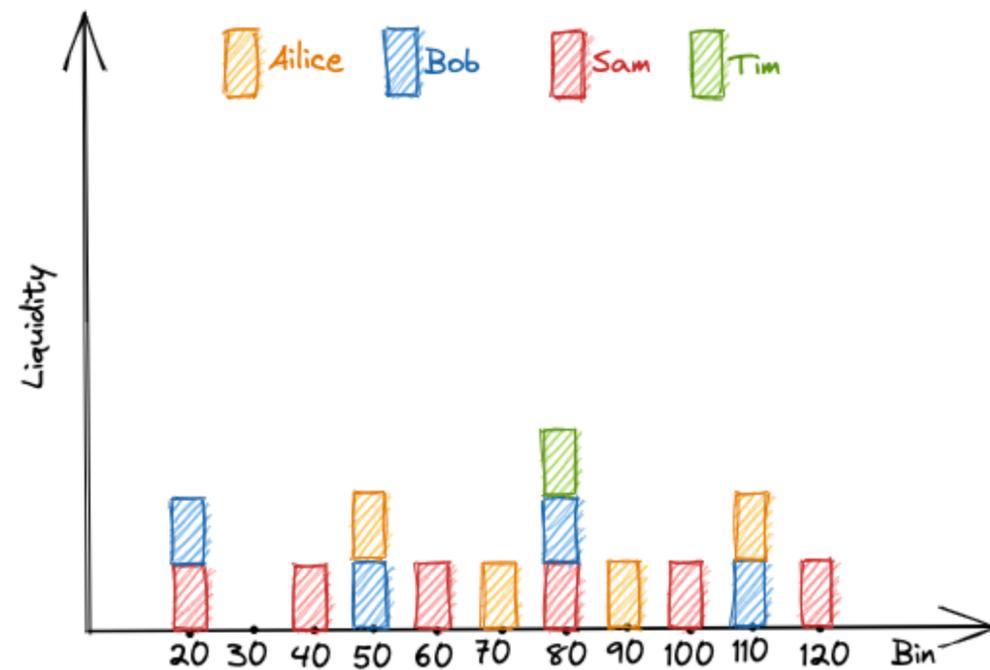


- 在Liquidity Book中一个LP = Sudoswap 中的一个Pool
- 在Liquidity Book中使用LP TokenID来 结算NFT资产
- 将Liquidity Book原先的ERC1155 LP Token升级为ERC721 LP Token

NFT AMM 架构设计

- NFT AMM中LPs如何添加流动性?

- NFT与FT流动性要如何分配?



$$\Delta = \frac{Bin_{upper} - Bin_{down}}{N - 1}$$

Bin_{upper} LP输入价格区间上限

Bin_{down} LP输入价格区间下限

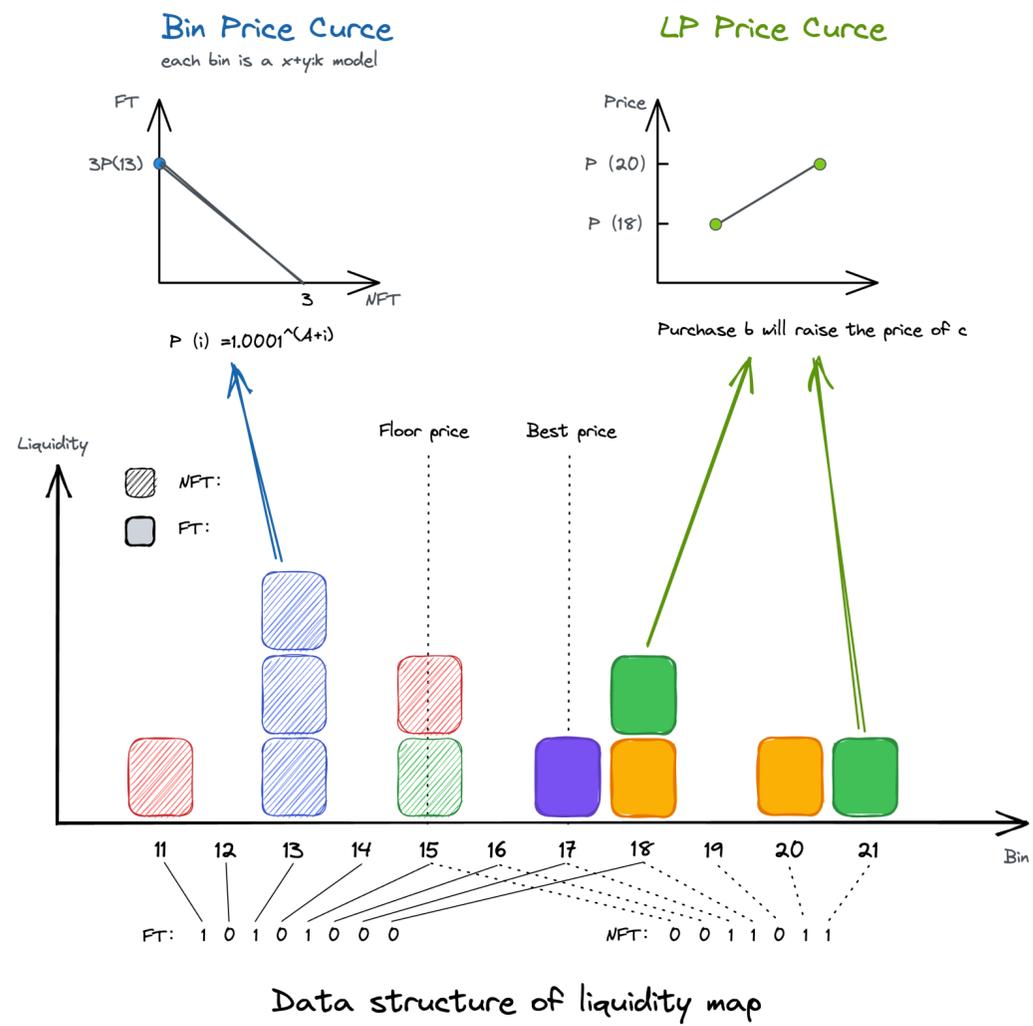
N LP输入NFT数量

Δ Bin间隔数

$$Sam \Delta = \frac{120 - 20}{5} = 20$$

采用等差Bin序列号的方式分配流动性

NFT AMM 实现原理



LP Token Data

Limited Order	1	Tokenid 1
LP1	2	26 ETH
LP2	4	Tokenid 2 Tokenid 5
LP3	6	Tokenid 3 Tokenid 4
LP4	8	39 ETH

Azuki-ETH

Item	Price
Tokenid 1	17 ETH
Tokenid 2	18 ETH
Tokenid 3	18 ETH
Tokenid 4	18 ETH
Tokenid 5	18 ETH

- 1. 使用Liquidity Book算法，将Tick修改为Bin，每一个Bin里面的算法是 $X+Y=K$
- 2. LPs添加流动性可以获得ERC721 LP Token，根据LP Token Tokenid清算NFT资产
- 3. 对于NFT资产根据Tokenid进行结算，对于FT资产根据先到先得时间排序规则进行结算
- 4. LPs的流动性分布采用等差Bin序列的方式进行排布
- 5. FT的流动性只能添加在Best Offer之下，NFT流动性只能添加Floor Price之上

Demo演示

