EPISODE 1247

[00:00:00] JM: QuidMarket market enables individuals or groups to quickly buy and sell assets. Decentralized platforms can struggle to execute trades when their platform does not have much liquidity for a specific token. Newer tokens or tokens with limited supply are most often the least liquid because there might be an imbalance of buyers and sellers. You can't sell token A for price X without a consenting buyer on the other end.

The company Uniswap is a decentralized protocol for creating liquidity and trading ERC-20 tokens on Ethereum. Uniswap encourages users to be liquidity providers, whereby they pool their assets into funds that enable people to complete trades without an opposite party. Instead, they swap against the liquidity pool created by the liquidity providers. Every swap incurs a small fee which is distributed proportionally to liquidity providers when they decide to pull their funds. Uniswap prices coins based on the simple formula X times Y equals K. In this episode, we talk with Noah Zinsmeister, engineering leader Uniswap. Noah also maintains Web3 React, a framework for building blockchain applications. We discussed cryptocurrency liquidity, the Ethereum blockchain and how Uniswap is building a community of liquidity providers and traders.

[INTERVIEW]

[00:01:12] JM: Noah, welcome to the show.

[00:01:14] NZ: Thanks so much for having me, Jeff.

[00:01:16] JM: You work on Uniswap. Uniswap is a liquidity pool. Can you explain what that means?

[00:01:23] NZ: Definitely. So it uses some traditional financial words but in a very untraditional setting. So what a liquidity pool is, is a smart contract, which is a piece of code that lives on the Ethereum blockchain. And this smart contract dictates how people can interact with it and the rules that people have to abide by if they want to participate with it. And so a liquidity pool allows users to come to the table with some assets, two assets in our case, asset A and asset B. And they can give those assets to the smart contract. The smart contract then can make them available for trading. So other users can come to the smart contract and offer one asset in exchange for another at some price. And, again, the

pool dictates both how the user, the price the user is getting when they're trading asset A for asset B as well as the rewards and the potential upside for the liquidity provider. And so there're some fees that are being collected in the assets that are being traded. And so those fees accrue as additional tokens that the liquidity provider can then withdraw at a later date. And so ultimately, a liquidity pool is a smart contract, where the rules of the system are dictated not by an individual or an institution, but rather just by code and then participants are free to interact with it in a way that they see fit.

[00:02:38] JM: How is this different from a market maker or an exchange?

[00:02:43] NZ: That's a great question. Well, it's somewhat of a form of both. So in a traditional exchange, there's a platform typically, right, like, your Geminis, your Coinbases, and that platform offers tools for market makers and for liquidity providers. And so those tools are typically in the form of API's, order books, things like that. And as a market maker, you typically have some strategy that you're executing against, it's usually pretty algorithmic these days. There are sort of manual ways to market make and to trade. But for the most part, you're sort of operating in this highly professionalized environment on a particular platform.

So in Uniswap's case, it's a bit different. So again, there are these traditional participants. So there are market makers, there are liquidity providers, there are arbitrage orders as well, which play a slightly more prominent part in Uniswap than in a in a traditional setting. And so let's just go through each of those in particular. So a liquidity provider, as I mentioned, is someone who comes to the table with assets. And so in a traditional setting, right, those assets are held sort of in custody by the platform and you can put typically orders on either side of the order book. And those are executed pro rata as the price changes. In Uniswap, you're required to upfront sort of give your funds over as a sort of active custodianship to this smart contract. And once your funds have been given to the smart contract, they're available for trading, and you no longer actually have control over those funds until you choose to withdraw. So the moment you deposit your funds, users are able to trade against them immediately. And at any point, you can exit the pool and withdraw your funds.

And so there's a there's a risk profile as a liquidity provider, right? You're making a bet that there's going to be some amount of trading volume, where some small portion of that trading volume is accrued as fees, which are then collected by you. But there's also this risk, right? Where if the price changes quite dramatically, then you're potentially exposed to some loss, right? Because you've essentially been

market making all the way as this price changes from where it is now to where it's going and you've been sort of selling on the way up is the easiest way to put it. And so that's a risk, but then there's this upside that you're getting with fees.

And so, again, it's analogous to a liquidity provider on an order book because you stand to earn fees and, again, do stand exposure to price movements if you're only able to sort of pull your order quickly before price changes. But there are again these differences, because you're not actively really market making, you're actually passively market making, which it's term we use it Uniswap. And so it's a very interesting way of participating in this marketplace that, again, is typically highly professionalized, but now is sort of becoming a bit more open with algorithmic market makers like Uniswap.

And then just to close this thought out, from a trader's perspective, it's actually quite similar that the analogies between a centralized exchange and something like Uniswap are much more similar. You're getting quoted a rate for a given amount of input that you're looking to trade for it for an output amount. And that process on a centralized exchange is just typically eating through orders in order book, and Uniswap, it's sort of continuously traversing a path that's determined by the market making function. And so in Uniswap case, that function is quite simple. It's X times Y equals K. And so this function is what dictates the rate at which you're able to trade assets for one another. And we can go into that probably more at a later date. And there are other sort of options, other competitors to Uniswap, and different market makers in this space use different ways of calculating prices. But ultimately, it's determined by an algorithm by a continuous function, as opposed to the sum of orders that are manually placed by individuals.

[00:06:22] JM: Is there anything like a liquidity pool in traditional finance?

[00:06:27] NZ: It's more rare than you might think, and there is some novelty to what we're doing here. Now, there are sort of pro rata ways that you can earn fees. You can participate in a market making firm, and you can be sort of an LP for a market maker. And there is a similar risk profile that you're taking on here, right? Like, ultimately, it's about being smart about being in the markets when a lot of trades are coming through and a lot of fees are being generated, when there's a lot of chop and volatility in the markets. And it's about being careful that you're not, again, exposed to these big price movements, and making sure that you're doing your inventory management correctly.

But ultimately, I think – And again, I'm not actually the expert on traditional finance here, so I won't sort of stray too far from my domain. But I will say that for the average participant potentially in these markets, market making is really not your domain. You're going to get eaten if you're a small fish. And the market is just highly professionalized and it's become very efficient in many ways, which is good, right? Like you want efficient market making across different exchanges, you want there to be sort of few arbitrage opportunities. So you're just not going to be able to keep up really, as an individual, if you're trying to get into the market making game. You need sort of an institution that's backing you or a highly sophisticated strategy that's backed by algorithms and data and servers and things like that. But in Uniswap, you can sort of be a participant in this market making strategy that the whole pool is sharing, right? And so you can participate in this upside, as well as take on the risk profile. And you know, you can do that in a way that's sort has less overhead and sort of a – There's less logistical as well as mental overhead. You don't really have to keep track of the way that the market is going day to day. You're just sort of making this long-term sort of high-level bet on what the prices of these assets are going to do in the future and how many fees are going to be generated. So that's the sort of extent to which it's a bit of a novel experiment, really, relative to a traditional exchange.

[00:08:30] NZ: Tell me a little bit about how Uniswap gets used in modern crypto markets.

[00:08:37] NZ: Sure thing. So something that people talk about a lot in crypto and in Ethereum in particular is Money Legos. So the idea behind this is that in the past, financial institutions have been pretty siloed. So innovation comes sort of in spurts and bursts and it's typically concentrated at the firm level, like a new product offering comes out, a new credit card, Stripe starts offering aggregation and payment processing and things like that. And it's often quite, quite hard to interoperate between the systems. And that's due to a variety of factors. There're regulations at play here that prevent the mixing of products across different industries, across different risk profiles, across different investor classes, things like that. And additionally, there's just – Ultimately, we're just very sort of wedded to this structure of financial offerings, again, at a firm level, at a company level. And of course, the Fed has a lot of programs that sort of are meant to level rates and level market making opportunities across a sort of an array of financial institutions, and those types of behaviors exists. But again, there's a lot of silos that are still around and that creates inefficiencies, and it makes it hard to interoperate and offer truly innovative products across the spectrum of financial opportunities.

And so what Ethereum offers a sort of centralized – Well, of course, it's decentralized, right? But maybe I should just say a shared settlement layer and logic layer for all of these financial applications to interoperate with each other. And so you don't have these silos anymore. And so there're, of course, a lot of tradeoffs that come with this, right? Like, if you imagine this shared global ledger where people are sort of competing to get their transaction. And you can imagine that that comes with a lot of overhead. There're some costs associated with participating in this network. But the upside is that, again, you get this shared layer where any application that's offering a financial product can sort of immediately out of the box interoperate with any other application.

And so let's be concrete about what that means. So Uniswap, as I've mentioned, is a way just to swap asset, swap A to B. And so this is a quite simple primitive, really, building block. And, of course, there are uses that don't require other moving parts. You might just want to buy and sell an asset as part of an investment strategy. And that's totally fine. That happens. But there are sort of mechanisms which require the exchange of assets as part of a larger system. And so Uniswap is really meant to be one of those building blocks. And so it lets you swap assets for one another in a trustless way without needing to go to an intermediary. And it can be done as part of – Again, as part of a larger system.

And so when liquidations need to happen, for example, you can sell assets for their “fair market value” on Uniswap at any point, and this can be done sort of on demand at any point by any protocol that needs to have like a liquidation layer, for example, in their system. And so, yeah, we're really excited about Uniswap not as necessarily a platform for trading, which it is, and that is a great use case. But we're excited for it to be, again, a building block, a layer that can be plugged in at any level from within anyone else's stack as long as they're building on Ethereum. So that's what's exciting to us.

[00:12:01] JM: Ethereum has a concept of atomic and non-atomic actions. Can you explain this a little bit more and describe how it has relevance to Uniswap?

[00:12:14] NZ: Absolutely. So this is an important concept for understanding the setting that Uniswap exists in. So let's take an example and then we can go through some of the implications of this example. So one thing that Ethereum offers that the traditional financial sector really can't because of the technology involved is something called a flash loan or flash swap. And so what this is, is it's basically an optimistic loan of assets to an individual with the expectation of repayment after the fact, right?

And so in the real world, this is pretty risky, right? You can't really give someone money and say, “I'm just trusting that you'll pay me back later,” because you don't know if they will, right? And this isn't just a matter of personal finance. This problem exists in the traditional sector as well. Asset-backed loans are very real. And so you need to – And unsecured loans are risky, and there's a whole variety of products and technology solutions that have evolved to sort of address different aspects of this unsecured loan technology, right?

But in crypto, there is a twist here, and we can actually ensure that payment happens after the initial loan of assets. And so let's walk through a very quick example. So Uniswap actually offers this somewhat as a feature. So let's say that we're operating in the Ethereum Dai pool on Uniswap. Meaning there's a pool, which has a bunch of Ethereum and a bunch of Dai dye in it. Dai is a token, it's pegged to the price of the US dollar. Ethereum is the base unit of Ethereum itself. And so these two assets are available for trading. What you can do is you can go to this pool. And, again, by the rules that are dictated by the smart contract itself, you can ask it for some Eth. You say, “Hey, I want 10 Eth. Can you loan me this?” And the contract will say, “Sure,” and it will give you 10 Eth. With that 10th, you can then do whatever you'd like, literally whatever you'd like on Ethereum within the space of what's called an atomic transaction.

And then at the very end of this transaction, what the contract does is it simply demands that you either A, repay the Eth, or B, give back an equivalent amount of Dai. And so remember, this is an Eth Dai pool. So if you've repaid your Eth, the pool is happy. You've sort of returned what you’ve owed it. And equivalently, you can also pay back the equivalent value of Dai. So if you borrowed 10 Eth and Eth is trading to $1,000 Dai, you have to send back $10,000 Dai or something, or 100 dye or whatever the conversions are.

And so, what this lets you do is essentially act on arbitrage opportunities without having to put up an initial capital allocation. So, in a typical arbitrage scenario you need to have capital across, let's say, two competing exchanges, and to take advantage of price opportunities you trade in one direction on the one and then the other direction on the other. And to do that you sort of need accounts and capital locked up in both of these systems.

But in Uniswap, if there's an arbitrage opportunity, external to Uniswap on this Eth-Dai pair, the price in Uniswap is different than the price externally, you don't actually need the capital to take advantage of this, because what you can do is you can ask the Uniswap pair for an optimistic transfer essentially, a flash loan, so that it'll give you the money up front. You can use that money to execute this arbitrage opportunity, earn your profit, return as much as you need to the pool and then pocket the rest, right? And so this is a really powerful tool to make markets more efficient. And really, it's impossible to think about in a traditional financial setting, because again, the technology stack involved here is just not replicable in a traditional setting. And so that's a really interesting application and benefit of this atomicity behavior.

And then just to back up maybe a little bit after having dived into that example, the basic idea behind an atomic transaction is that you can execute something on Ethereum and basically roll back those changes if something doesn't go the way you wanted it to, right? So again, to take a traditional analog. Let's say you wire some money to your broker. You're trying to buy a house or you're trying to get an apartment, and you ended up putting the wrong wire amount. Well, that's pretty bad, because that's sort of an irreversible transaction in a lot of ways. And so on Ethereum what can happen is maybe you don't make a mistake. But let's say there's some price movement ahead of your trade that you were not anticipating. And let's say it would be adverse to your trading. You sort of don't want to execute that trade. What you can do is essentially roll back that transaction and act as if the state changes that occurred as a result of your behaviors never happened.

And so, in a flash loan scenario, you can ask for the money up front, you can try to execute your arbitrage opportunity. But if the ARB doesn't exist anymore, and it's not possible to profitably trade, you can basically just cancel that operation. The flash loan won't be able to be repaid, but you can just roll back the state and so it's as if it never happened. And so there is a small fee involved with this, of course. You can't sort of get this for free. But it is this really unique, interesting behavior that you can get on Ethereum. And so, when you're thinking about an atomic transaction, essentially, it's just a bundle of state changes that either all go through or none go through. So it's an all or nothing situation, and that can drive a lot of really powerful behavior.

[00:17:11] JM: I'd like to know a little bit more about the process of building Uniswap. Can you tell me about the engineering process that went into building the first version of Uniswap, but like the architecture and the planning and the implementation? How you tested it even?

[00:17:30] NZ: Absolutely. So this is a great question. And I think I'll probably frame this in the context of v3. So for those of you who are unfamiliar with Uniswap, we're on our third iteration. Uniswap v1 was the sort of initial MVP launch product. Uniswap v2 was released about a year ago now in 2020. And it was a sort of evolution of the original product. And then Uniswap v3 is our most ambitious project to date. It's currently in its latest stages of development, where basically we finalize the code, and we're waiting to deploy it pretty soon now.

So Ethereum is a pretty interesting setting to do software engineering. And the sort of base environment that we're living in is called the Etherium virtual machine, the EVM. And so it's a tailor made execution environment for code that is suited in particular to blockchain use cases. So there's a variety of tradeoffs and security concerns and data storage models that have to be sort of tailored to blockchain in particular.

And so the AVM has some funky rough edges. But it is this environment where all the code that is on Ethereum sort of lives within and operates within. And so we're developing code for the EVM. And so we do that in a language called Solidity, which is sort of a mashup of Java, JavaScript, C++ and it's a bit of a young language, let's just say. So it doesn't have a lot of the fully fledged features that some of these more traditional languages have. And so there are often a lot of rough edges that you run into and you do have to be quite, quite careful with the patterns that you're using.

So to dive into a little bit of the sort of tech stack of v3, we're basically writing these contracts in Solidity, and they are responsible for costing a lot of assets, right. And so we have to be pretty sure that they work. And so we essentially have two levels at which we run tests. We have unit tests, which are just very typical sort of tests of particular behaviors. We want to make sure that the math operations are being rounded correctly. We want to make sure that the recipients are actually getting their assets. We want to make sure that fee collection is pro rata across all existing liquidity providers, things like that. So we have a very sort of extensive suite of unit tests that test these specific behaviors.

And then we have what are called fuzz tests. And so fuzz testing for those who don't know, and you can use this in traditional settings as well, not just blockchain. It basically searches a vast array of possible inputs and tests some conditions on an output. So you have a function that takes an input and transforms it and it produces an output.

In Ethereum, the range of potential inputs is it can often be quite, quite large. And you actually have to be prepared and capable of accepting all possible inputs because the system is adversarial, right? Like, an attacker is just as free to use your software as anyone else, right? And so, you can't rely on like an API to sort of prevent DDoS'rs or bad actors. You really have to be, again, fully prepared to accept the worst possible case of this function being called, right?

And so that is a pretty unique property. And so fuzz testing really helps us with this, because what it does is it simulates a given transaction call against, again, as I mentioned, a wide, wide range of possible inputs, and it basically checks some conditions on the output. And it makes sure that this function is operating as intended. And so we also have an extensive amount of fuzz testing. And that has actually been really, really helpful. Because often, the code that you write seems intuitively correct to you. But often, you're missing something very crucial. And so we found that there have been numerous cases where this fuzz testing has expose a vulnerability or a flaw in our reasoning that we just wouldn't have caught without these automated tooling layers. So that has been very, very helpful.

And then I guess there is a third point to mention. Beyond unit tests and fuzz tests, there are some formal verification tools that we have been using that we've used a lot in v2 and that we're also using in v3. And what those are, are sort of more high-powered mathematical models that let you prove sort of beyond a shadow of a doubt properties about your code. And so this is reserved for the most low-level functions, because it's extremely hard to prove properties about complex systems, but you can break those systems down into their constituent parts and prove small scoped properties about the parts themselves. And so there's some work we've been doing around formal verification, which is probably slightly out of scope, maybe we can dig into it a little bit. But that's just something to note. And we have been running those horrible verifiers against the v3 code. And so that's the general testing process.

And then ultimately, what we have is just a piece of bytecode that we're going to be deploying to the blockchain. And then once it's live on chain, again, it just becomes its own sort of self-sovereign piece of code. And the actions that it can take are fully dictated by the parameters that we wrote into the system. And so it's a very high stakes developing environment, right? Like, you really need to make sure that you've sort of done every – You've crossed all your T's and dotted all your I's because there's no going back once it's deployed and once capital has been allocated to it. And so it's a very high stakes, but also very exciting and very interesting domain.

[00:22:49] JM: I'd like to go through some of the terminology around Uniswap. First, let's simply define what is a swap?

[00:23:00] NZ: Right? So as a swap is just very simply an exchange of one asset for another. And so, if you imagine someone with apples and oranges, they have 10 of each, right? You maybe come with two apples, and the apple and orange dealer says, “Ah! Two apples, that's about equivalent to one orange,” Because they're not going to give you two, because they need to take a little fee, right? So they'll give you one orange for two apples. And now they have 12 apples and nine oranges. And now someone comes with two apples. That's no longer worth a full orange, potentially, right? Because the ratio of assets in the pool is now unbalanced. There're more apples and oranges. So they're less valuable. So a swap is essentially just exchange of one asset for another at a market price that's determined by the relative ratio of assets in a pool.

[00:23:43] JM: And could you also define what a pool is?

[00:23:46] NZ: Of course, yeah. So the pool, as I've alluded to, is this smart contract that is essentially the custodian of the apples and oranges, in our case, right? It's the permissionless, sort of ownerless piece of code, which owns the assets, for as long as the underlying liquidity providers haven't trusted them, the assets to it. And so, the pool is what ultimately decides how many apples you get for a given amount of oranges at any point. And because of the atomicity feature that we discussed earlier, there's always a market price for apples and oranges, right? There's never any ambiguity. There's never any, like pending orders. There's never any settlement that needs to be waited upon at any moment, because a transaction either goes through or it doesn't. There's a ratio of apples and oranges in a pool. And so you're always able to get a quote for an output amount against any input amount.

[00:24:35] JM: What is an Oracle?

[00:24:37] NZ: So, an Oracle is a way of accessing data on a blockchain that cannot be sort of immediately gathered from the surrounding environment. So, blockchains are somewhat of a sequestered system. This EVM that I was discussing earlier, the virtual machine, which smart contracts all live in, really has no access to the outside world. It can't fetch API results, it doesn't know the weather. It doesn't even know the price of the dollar or Bitcoin or Eth. And that's basically just part of the

security properties of this chain. And so that presents pretty unique challenges. So when you're building financial products, it's often quite necessary to be able to know the dollar price of an asset, right? You need to be able to compute collateralization ratios. You need to be able to track the dollar value of portfolios, right?

And so, to do this in Ethereum, requires an Oracle. And this Oracle is some party that is telling you how much an asset is worth, or it's telling you the weather last week in Spain. It's just a piece of – It's a reporter of data that has various security properties, right? And so, the security of Oracle data cannot be guaranteed by the chain itself by the base layer. The security rather needs to be ensured via some other means. So there are a variety of ways to get real world, "real world" data into a blockchain setting. And all of those various ways encompass the different methods of being an Oracle or retrieving Oracle data on Ethereum. And so we can go into those at any level of detail you'd like. But that's the general setting that we're discussing here.

[00:26:26] JM: So why are oracle's so relevant to Uniswap?

[00:26:30] NZ: Right. So Uniswap is actually in a pretty unique place with respect to oracles. We actually don't need an oracle to offer – The smart contracts that we put out don't need an oracle in order to offer, swaps of one asset for another. And that's a pretty unique set. At most, I would say most DeFi, decentralized finance, applications on Ethereum do in fact need an oracle for the proper functioning of their systems, but Uniswap does not.

So let's go into a little bit why and then maybe sort of what opportunities this presents to other people that are integrating with Uniswap. So why doesn't Uniswap need an oracle? Well, as I mentioned, the pools in question are always able to offer one asset for another. And so whenever the price of Eth and Dai in the Eth-Dai pool and Uniswap diverges from the Eth-Dai price elsewhere on centralized exchanges, on Coinbase, for example, that represents an arbitrage opportunity.

And so what that means is that there's a profit to be made by trading, again, in one direction in Uniswap, in another on a centralized exchange. And so that's going to be exploited, right? Someone's going to take advantage of that arbitrage opportunity. And so, ultimately, the price on Uniswap tracks the real world price whenever there's enough profit to be made from equilibrating that price.

And so while this might seem sort of extractive, it's actually the real core reason why Uniswap works at all, and why the prices that are offered by Uniswap are even close to something that people would take advantage of. And so this behavior actually lets us be fully agnostic of any kind of oracle situation, because we don't need to know what the price is off-chain. We're actually just relying on arbitragers taking a profit from the system whenever there's a deviation.

And so this is a pretty powerful system for us. And it simplifies a lot of the security assumptions around Uniswap, because money on the table is a very powerful motivator. And so you can make pretty strong assumptions about the properties of the price on Uniswap, based on the amount of money at stake to be made if price differences exist.

And so then let's move on to what this means for the rest of the ecosystem, right? Because the price on Uniswap, as we just discussed, tends toward the true off-chain price at any given moment, you can actually use the Uniswap price as an Oracle input to other systems, right? So like the Eth-Dai price in the Eth-Dai Uniswap pool – And remember, I said that Dai was pegged to $1. So the Eth-Dai price is really the dollar price of Ethereum. And that you can access natively from any other application on Ethereum without needing to trust some off chain source of what the dollar price of Ethereum is at any given moment.

And so, that is really powerful, because it basically brings this what used to be a problem of fetching data external to Ethereum into the Ethereum world itself, and it becomes a matter of fetching data just from Uniswap instead, and that is a method of getting data that can be much more scoped and involves far fewer trust assumptions. And so the only real concern here now is not about the security of who's reporting this data to you, because you don't need to trust anyone anymore. The code itself is offering this data to you. But you do need to be worried about manipulation and attackers.

And so, because Ethereum is an adversarial system, the price – You can choose to manipulate the price of a pool, and it'll cost you a lot of money, but no one's going to stop you, right? And so, if an external system is using the Uniswap price as an oracle for what the dollar price of Ethereum is and then they're using that as an input to their system, there's a possibility to more value is locked in this system than is available for arbitrage profits on Uniswap. And so it might be more profitable to sort of quote “attack” the price on Uniswap by pushing it far from the true value. And of course, they'd be burning money to do so because the price will be constantly getting arbitraged back to its true price.

And you'll have to, again, trade it up to its manipulated value. So you'll be burning money, but the potential payoff might be quite high, because this other system might be now using an incorrect price as part of its calculations and there's maybe some greater profit opportunity to be made there.

And so that's just a bit of a window into – There's a really wacky world of decentralized finance that involves, again, a really different tradeoff space than traditional applications, right? Like, there's all these trust assumptions. There's manipulation resistance that you have to think about. And there are, of course, ways to address this on Uniswap. But it's a really interesting setting. And this oracle problem is one of the biggest that's involved in building decentralized applications. So to the extent that Uniswap can inform that oracle process for other applications, we think it's a pretty valuable tool.

[00:31:21] JM: So to go inside the mechanism of Uniswap smart contracts. Each contract manages a pool of two ERC-20 tokens. And the balance of these tokens is maintained by a constant product formula, which is X times Y equals K. And people who have looked at Uniswap may have seen that formula. They may not know exactly what it means. Can you explain what that formula means? The X times Y equals K and why that's relevant?

[00:31:52] NZ: Definitely. So this is the constant product formula exactly that I mentioned earlier in the podcast, and you basically –You nailed it. The root of Uniswap is really X times Y equals K. And so what that means is the X represents one asset, the Y represents another. So in our apples and oranges example where the marketplace has 10 apples and 10 oranges in a pool, X is 10, Y is 10, and K is this sort of unitless constant, which represents the scale of this pool. And so in our case, it's X times Y equals K. So 10 times 10 equals 100. So K is 100. And K isn't really meant to be – There's no real world analogy. It's just a sort of – It's like a chit, like a little token that represents the relative magnitude of the pool irrespective of the amount of underlying assets in the pool.

And so in our example, someone comes with two apples and they say, “Hey, can I trade this for an orange?” And the pool says, “Sure.” And the reason that's okay is because let's imagine the new X and the new Y now, right? New X is going to be 12, because someone gave two apples. And the new Y is going to be 9. So we have now instead of 10 and 10, we have 12 and 9. And so what's 10 times 10? 100. What's 12 times 9? Well, it's 108. Right?

And so K has actually gone up in this trade. And if we imagine, instead of two to one, let's say we were doing two to two. So someone came with two apples. And the marketplace instead had given them two oranges. Well, what's the balance of the pool B after that trade? Well, it would be 12 and 8, and 12 times 8 is 96, which is actually less than our original K of 100. And so what that means is that if the pool keeps executing trades like this, ultimately, it's going to be left without any value, right?

And so what has to happen is that this K value, which represents the sort of unitless scale of the liquidity pool always has to be going up, right? And so the trades that happen against a marketplace, against a Uniswap pool, must cause this K value to increase. And so that actually is what dictates the market price at any given point. It's just making sure that in some generic sense, irrespective again of the of the actual amounts of the underlying assets in the pool, that the pool is sort of growing in an abstract sense. And so of course, as I mentioned, the relative amounts are always changing, because you're never giving both assets and getting both back. You're always getting one in exchange for the other and then you're changing the price. But in sort of geometric terms, you're actually increasing the size of the pool, always in Uniswap, and that's what helps it maintain its security properties.

[00:34:30] JM: Why do you have different contracts for different currency pairs rather than one big contract?

[00:34:38] NZ: That's a great question. So for those of you who aren't familiar, Uniswap has a sort of factory pool setup. So we have a factory contract, which is responsible for creating liquidity pools for any number of assets. And so what that means is that we don't control the assets that are listed on Uniswap. Anyone who wishes to become a liquidity provider for a given pool can simply create a pool, deposit their initial assets, and then anyone can immediately start trading those assets on Uniswap.

And so there's a variety of sort of tradeoffs involved here. There're some costs associated with creating pools, and that cost has to be borne by liquidity providers, potentially by traders. And so the reason really why separate pools exist is because a shared pool has some risks associated with it. It means that if there is a bug in one of the tokens involved, in any Uniswap pool, that can potentially bleed over and affect the tokens in any other pool because there's this shared contract at all token and all logic is executed against in Ethereum.

And so having separate pools that are sort of partitioned and we're only responsible for trading one asset for another, and that's it, sort of its scopes and compartmentalizes, again, the logic and the custodianship of assets in a way that is far easier to audit, far easier to understand and intuit as a developer. And ultimately, it's just, from our perspective, a bit of a safer way to go about development. And there are some I think cost savings. It's fair to be had if we were to say instead of distributed network of X times Y equals K pairs, we instead just have a single canonical trading contract that handled all possible trades between assets. It ultimately becomes far harder to do the accounting appropriately for that big shared pool and there's this security property where you have to worry about the security of any individual token in the system potentially bleeding over and affecting the security of the entire system as opposed to only affecting pairs in which it is a constituent asset.

[00:36:49] JM: So for each pool, there are liquidity providers, and those liquidity providers receive LP tokens or shares. What is the value of those shares? What are they receiving in return for the shares?

[00:37:06] NZ: That's a great question. And that really gets to the root of what are the units of this K constant that we discussed earlier? So as I mentioned, K is this unitless number, and it represents the “size” of a liquidity pool? Well, it turns out that you can actually formulate LP tokens as a sort of manipulation or a reformulation of K. And so what an LP token is, again, it's actually unitless. So there's no unit to it. But what it does is it represents your pro rata share of assets in the pool. And so let's take a pool that has just been created between Dai and Eth and the very first liquidity provider is coming along and are giving some assets. They're going to give, let's say, one Eth and 1000 Dai. They're going to be issued in exchange for that an LP token, right? And this token represents their claim on the underlying assets, right? Because remember, at any point, a liquidity provider is free to sort of remove their assets at will and then get them back.

So this LP token is just an IOU, essentially, that can be given back to the contract in exchange for the underlying value at any point. So someone contributed their Eth and their Dai and they got this LP token and return. Now let's say someone else comes along and instead of contributing one and 1000, they contribute .1 and 100, right? So much less. They're issued another unit of LP tokens, but rather than – And let's say the first person got 100 tokens. Rather than getting also 100, this second contributor, this second liquidity provider gets 10, because they've contributed a 10th of the liquidity. And so now, the total liquidity in the pool is 110, 100 of which is owned by the first LP, 10 of which is

owned by the second. So you can basically apportion fees now pro rata to liquidity providers based on how much underlying they staked.

And so if someone comes and pays – They want to swap Eth for Dai and they end up paying like a 50 Dai fee, let's say. 10 over 100 and 10th of that, whatever that is, one over 11th of that, will go to the smaller liquidity provider, and then 10/11th, the lion's share of the fee, goes to this initial liquidity provider who ultimately is putting more capital at risk and more capital at stake.

And so what lets us do that pro rata calculation of fees to liquidity providers, LPs, is this LP token. So it's basically the unit of account that lets you both redeem your stake in the pool for the amount of underlying assets you've contributed. And it lets you sort of calculate ongoing fee growth and ensures that you get fees sort of proportional and relative to the amount that you've contributed to the pool.

[00:39:52] JM: Tell me more about the dynamics of the shares. Like how are the amount of shares determined and how is the incentive structure maintained?

[00:40:02] NZ: So the very first liquidity provider, as I mentioned, is actually issued shares according to the square root of the product of the asset values they're contributing. So, in X times Y equals K, basically, for a given amount of X and Y, you take the square root of K, and then issue that many token to this provider. And so that if you sort of do the math, that's the thing that actually makes sense here, and lets you do this pro rata tracking that I was discussing earlier.

And then at least in v2, subsequent liquidity providers would actually just be issued shares relative to the current value and the sort of total supply of LP tokens. So I think it's easiest to think about an example. So let's say that there's, again, one Eth and 1000 Dai in a pool and then there's some amount of liquidity provider tokens, right? The very first liquidity provider put in this much money and they they've been issued the square root of 1 times 1000 tokens, right? So they actually have 10 LP tokens. And then some other liquidity provider comes along, and what they're doing is they're providing a fraction of the underlying value of the pool, right? So if they provide .1 Eth, they're providing a 10th of the current value of the pool. If they're providing 1 Eth, they're doubling the pool, right? They're providing 100% of the current value in the pool.

And so what happens when you provide more of the underlying asset is that you dilute existing LP tokens by issuing new tokens, right? And so let's say that the new entrant is providing exactly 1 Eth and then 1000 Dai, right? So they're literally doubling the amount of assets in the pool. What you can basically do is just take the existing total supply of LP tokens and double it. You can just issue another 10 tokens such that the total supply now is 20 and each LP has 10 in turn. And so that's a pretty easy sort of scenario to run through if you're just considering people entering at this exact price. It gets a bit more complicated when fees are earned, because let's say the first liquidity provider enters. They're issued their LP tokens, and then someone trades, right? That changes the relative asset ratio in the pool. And now a little bit of fees have actually accrued to this first liquidity provider.

And so now when a new entrant comes, it's no longer this clean calculation. But what you can actually still do is basically just calculate the proportion of underlying assets that the new LP is contributing and then mint that same proportion of liquidity tokens, LP tokens. And it turns out that, again, if you do the math, that basically ensures that this pro rata fee distribution works appropriately.

And so in v3, there is some complexity here. We've introduced some additional sort of accounting layers that complicate the story. But ultimately, it's as simple as what I described. And there're just some technical bells and whistles on top to ensure that everything works out appropriately. But the math underlying is the same.

[00:42:46] JM: Tell me about how Uniswap fits into the future of the DeFi ecosystem and how you see it functioning as time goes on.

[00:42:57] NZ: Right. So we're very interested at Uniswap and really pushing the frontiers of simplicity, as well as sort of scalability and generalizability. So what made Uniswap v1 and v2 special is that it was ultimately really simple. It was just X times Y equals K, right? And we discussed, there's no oracle that you need to run Uniswap. Uniswap just sort of works, right? And it's incentive compatible. LPs want to join typically because they want to earn fees. Traders want this ability to sort of permissionlessly get quoted input and output amounts without having to go through any sort of KYC rigmarole, or have to worry about token listings or things like that. And, ultimately, arbitrageurs are earning profits from the system and in exchange for keeping the prices in line with centralized alternatives.

And so, ultimately, that system turned out to be quite robust and has worked really well for the past couple of years. Ultimately, though, crypto is as a space evolving, and it's becoming, again, a bit more professionalized. So it's no longer possible – It's possible, but no longer is it going to be the case that Uniswap v1 is sort of scalable and powerful enough to deal with all the possible DeFi applications that are being built on Ethereum. And so I think what we're excited about is continuing to evolve this core idea of what makes Uniswap Uniswap, righ? X times Y equals K. And expanding it in a way that remains true to our original ethos, but offers tools and becomes sort of a bit more of a powerful building block that meets people's needs in DeFi. And so that basically v3 is an attempt to do that. It's taking X times Y equals K. And I know we haven't discussed v3, but what it's doing is it's basically just bringing it to another level and it's trying to appeal a bit more to the professional side of the market and to offer our product that really could potentially be on par with the capital efficiency of centralized exchanges.

And so I think that we want Uniswap to continue to evolve, while remaining true to its ethos and, again, really push the boundaries of what it means to be an automated market maker on Ethereum. Because, typically, people thought of Uniswap as a very good product, but maybe only good relative to its competition, right? Like, because the EVM and the blockchain sort of competing environment is so constrained, there are real limits on what you can do relative to a centralized exchange. And so some people said, “Oh, Uniswap is great for X, Y, and Z. But ultimately, it can never compete on volume. It can never compete on capital efficiency. The market making tools that are offered by these centralized exchanges are just too powerful, right? It's too fast. There're too many benefits.

But I think that what Uniswap v3 is trying to prove, and what we're going to continue to try to prove as a company as an ecosystem, is basically that we can compete along almost every dimension that centralized exchanges can compete on. And so we're basically trying to be the best that we can and serve that sort of defect community broadly by like offering this oracle support. By giving people the tools to, again, run automatic liquidations over Uniswap or swap token sort of on demand on the spot while also being a powerful enough building block and base layer that can actually compete with these centralized alternatives. And so I think that's the long term vision. And what that means for the future of our products, of our team, is a bit uncertain, just because the space is evolving so quickly. But we're in it for the long run. And we think that v3 is a really powerful step to, again, just like leveling up Uniswap and making it that much more powerful and capital efficient. And so I think we are, we're really looking forward to the future of what crypto and defy becomes, and I think that Uniswap is going to be a big part of that.

[00:46:42] JM: Awesome. Well, Noah, thank you so much for coming on the show. It's been a pleasure.

[00:46:46] NZ: Jeff, this has been great. Thanks for having me.

[END]