Blockchain y Criptomonedas II

UCEMA - QUANt

Clase III: 17/11/25

Matias Selser

Agenda

- Módulo X: Discusión de la tarea
- Módulo XI: *Flash Loans* y margin trading
- Módulo XII: Modelos de arbitraje para AMMs de UNIV2
- Break
- Módulo XIII: Riesgos de ejecución
- Módulo XIV: Otros productos financieros

Módulo X: Discusión de la tarea

Módulo XI: *Flash loans* y *Margin trading*

11.1 ¿Cómo podemos hacer funcionar estas ideas?

- Restricción: custodia de tokens
  - Cexes en general son *custodials* de los tokens, mientras que los dexes no lo son
- Principal restricción: manejo de capital.
- Para hacer un arbitraje cex-dex, habría que disponer de tokens en un mercado y usar esos tokens en el otro, todo en el mismo espacio temporal.
- Complejo operativamente

11.2 Cexes: *Margin Trading*

- Cada cex tiene una forma distinta de implementarlo, pero la idea es la siguiente
- Se collateraliza una posición con algún token (puede ser una stable coin, por ejemplo) y se opera el token deseado
- Referencias: binance

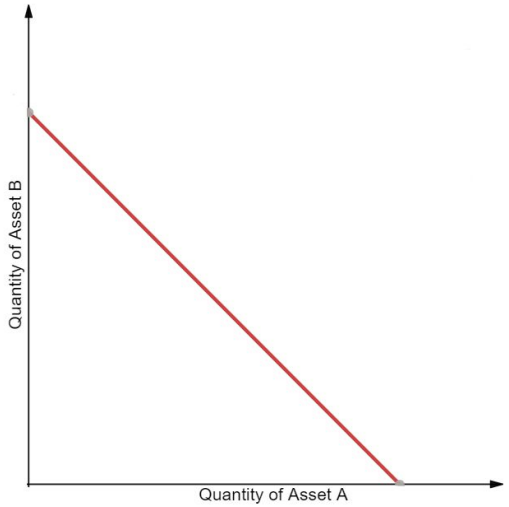
11.3 *Flash Loans* o *Borrow*

- En defi, las opciones son un poco más amplias.
- *Flash loans*:
  - Préstamos intra transacción. Permiten pedir prestado activos sin collateralizar nada, con la condición de que se devuelvan en la misma transacción
  - Solo implementables con smart contracts
- Borrowing:
  - Existen protocolos como AAVE o Compound que permiten pedir préstamos sobrecollateralizados de tokens.
  - Más transacciones, más complejidad para cerrar el círculo

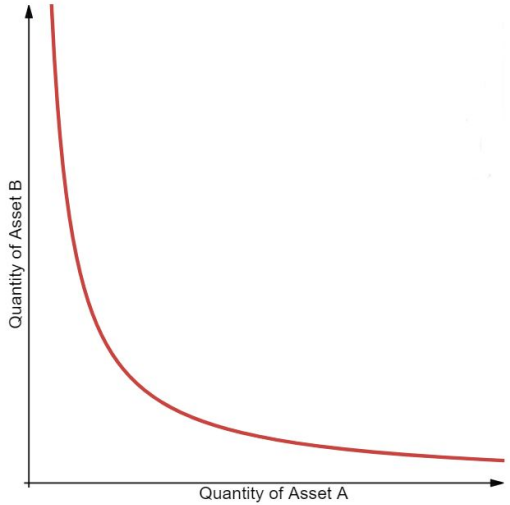
Módulo XII: Modelos de arbitraje para AMMs de UNIV2

12.1 *CPMM* y *CSMM*

- *CSMM*: *constant sum market maker*
  - Consiste en un modelo en el cual la suma de los tokens es constante: $x + y = k$



- *CPMM*: *constant product market maker*
  - Consiste en un modelo en el cual el producto de los tokens es constante: $xy = k$



- *CPMM vs CSMM:*
  - *CPMM* tiene liquidez infinita
  - CSMM permite drenar las reservas, mal modelo
    - usado en stable coins en algunos casos
  - Problemas de esta generación:
    - *Impermanent loss*: recordemos, diferencia entre quedarse el token y e invertirlo en un AMM. Producido por la divergencia hacia algún extremo del pool
    - Baja eficiencia de capital:
      - Se necesitan muchos tokens para generar un *price impact* similar al observado en los *orderbooks* tradicionales.
      - Provee liquidez en todo el rango de precios, y no en el rango de precios trascendente (revisar UNI V3)

12.2 Maximización

- Volviendo al caso implementado en múltiples exchanges, se pueden observar divergencias en los precios absolutos de los activos
- ¿Son aprovechables? Veamos
- Recordemos:
  - Pools uni v2, CPMM
  - Para un par determinado, en un exchange determinado: $x_1 y_1 = k_1$
  - Para el mismo par, en otro exchange: $x_2 y_2 = k_2$
- Para todos los pares, podemos obtener (y ya lo hicimos!)
  - La dirección del pool asociado
  - Las reservas de ambos tokens, entonces podemos obtener k

- ¿Qué pasa si intentamos agregar una cantidad dx al pool?

$$(x + dx)(y - dy) = k$$

- Para que el producto se mantenga constante, al agregar una cantidad dx vamos a sacar una cantidad dy. ¿Y cómo calculamos dy?

$$(y - dy) = \frac{k}{x + dx}$$

$$y - \frac{k}{x + dx} = dy$$

- ¡Determinístico! Excelente. Qué pasa si, entonces, intentamos encadenar dos pools entre sí. Es decir, colocar dx1 en el primer pool, obtener dy1, y poner dy1 en el segundo pool para obtener dx2.

$$y_1 - \frac{k_1}{x_1 + dx_1} = dy_1$$

$$(x_2 - dx_2)(y_2 + dy_1) = k_2$$

$$(x_2 - dx_2)(\frac{y_1 + y_2 - k_1}{x_1 + dx_1}) = k_2$$

- Entonces, dx2 es igual a

$$(x_2 - dx_2) = \frac{k_2(x_1 + dx_1)}{y_1 + y_2 - k_1}$$

$$x_2 - \frac{k_2(x_1 + dx_1)}{y_1 + y_2 - k_1} = dx_2$$

- Por lo tanto, ¿cuál sería el problema que estamos buscando optimizar?
  - ¡Obtener la mayor cantidad de tokens a la salida!

$$argmax_{dx_1} \; F(x_1, y_1, x_2, y_2, dx_1)$$

$$F(x_1, y_1, x_2, y_2, dx_1) = dx_2 - dx_1$$

- ¿Cómo resolver?
  - ¡Podemos derivar con respecto a dx1 y hallar ese valor!
  - Tarea propuesta, resolución la semana que viene!

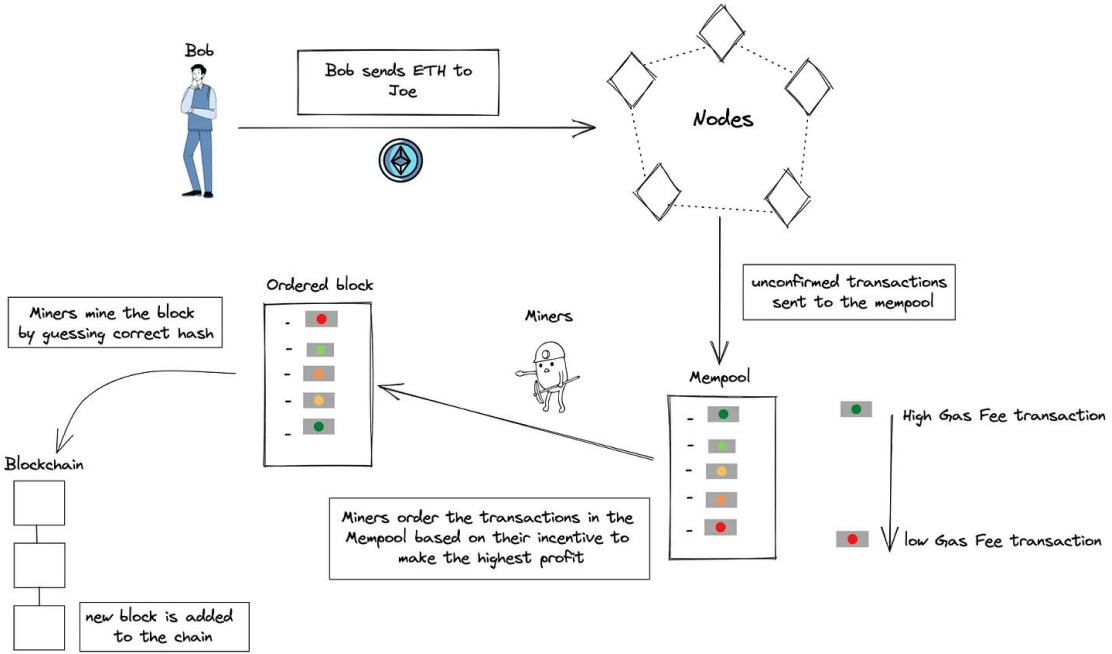
12.3 ¿Qué pasaría si…?

- Detalles a tener en cuenta respecto al setup:
  - *Fees*:
    - ¿Se están considerando los fees de *trading* de los pooles en este set up?
    - ¿Cómo cambian las ecuaciones si los fuéramos a considerar?
  - Restricciones:
    - En el problema original, no está acotado el dominio de las variables asociadas
    - ¿Qué pasa si tenemos en cuenta que, por ejemplo, tenemos un máximo número de tokens para operar? Es decir, nuestra *wallet* tiene 1 eth y no más para poner en el pool. Restricciones de capital
  - Ecuaciones:
    - En este caso, este problema se puede plantear porque conocemos el comportamiento de las curvas. ¿Qué se puede hacer si las ecuaciones no son derivables?
    - ¿Y si en vez de un pool tenemos un *orderbook* involucrado?

Módulo XIII: Riesgos de ejecución

13.1 *Mempool*

- ¿Cómo es el flujo de desarrollo de una transacción en Ethereum?



- Usuario envía la transacción a su nodo
- El nodo comparte la transacción al resto de los nodos
- Las transacciones se ordenan por los mineros o validadores
- El ganador de la lotería o el que adivina el hash **arma el bloque**
- El bloque se propaga a la red

- Ahora bien, ¿cómo se arma el bloque?
  - ¡Los validadores tienen un rol activo en esa decisión!
  - Si uno fuera dueño de un nodo, cuando gana el sorteo podrá elegir de qué forma se ordenan las transacciones
  - Por ejemplo:
    - Se observa que en la mempool alguien está intentando hacer un swap entre el token A y el token B
    - Cómo se conocen los parámetros del *swap*, es fácil (dado que la ecuación es determinista) conocer el resultado de las reservas después de esta transacción
    - Resolviendo entonces el problema anterior, se podría computar cual es el *input* óptimo para ese pool que se desarbitra para maximizar la ganancia! (si es que hay)
    - Se envía entonces un “doble swap” atrás del swap que desarbitra las reservas

- Si fuera tan sencillo…
  - Muy bien, entendemos cómo funciona el arbitraje entre pools
  - Si es tan sencillo de implementar, ¿qué dificultades se pueden llegar a presentar para tomar esos arbitrajes? ¿Ideas?
    - ¡Sistema de lotería! Si controlamos un nodo, solo vamos a poder ordenar las transacciones con muy baja frecuencia (proporcional al *share* de tokens *stakeados* que tengamos)
    - Si usamos un *provider*, quizás tengan un mejor share. Pero competimos por el mismo arbitraje con el resto de los arbitradores. Para obtener ese arbitraje, tenemos que mejorar el gas price o prima para el validador para que nuestra transacción se ordene como queremos
    - Por otro lado, este modelo de arbitraje es intensivo en capital. Se necesita tener ambos tokens para poder hacer el *swap*. Esto suele implicar hundir capital, y tiene un costo de oportunidad asociado

- Si fuera tan sencillo… (continuación)
  - Asumamos que podemos resolver lo anterior asociado al capital, y tenemos suerte de que podemos ordenar nuestra transacción en el lugar adecuado del bloque. ¡En realidad necesitamos hacer dos transacciones! Un swap en el primer pool, y otro swap en la dirección contraria en el segundo pool.
    - Solución a esto: smart contract que en una transacción haga los dos swaps! Se puede resolver en Remix
  - Si no hay disponibilidad de capital, hay que hacer un borrow. Otra transacción.
  - Infraestructura…

13.2 Privacidad

- Existen providers de mempool que nos permiten tener acceso a las transacciones pendientes de los distintos usuarios
- ¿Todas las transacciones son públicas?
- ¡No! Existen servicios de nodos privados, sobre los cuales no se tiene acceso a la mempool
- Flashbots: nodo RPC que NO brodcastea las transacciones pendientes. ¡Más seguro!

13.3 Ataques

- Como vimos anteriormente, existe un parámetro en los swaps del modelo de UNIV2 que se llama *minAmountOut*. Esto responde pura y exclusivamente a la necesidad de proteger los trades de los usuarios.
- Si uno fuera a enviar un trade desprotegido (*minAmountOut* = 0), podría quedar expuesto a *frontrunning*, *backrunning*, o *sandwich trading*.

- *Frontrunning*
  - Consiste en ver una transacción pendiente, procesarla, y enviar una transacción para que se ejecute antes que esa
    - Ejemplo: liquidaciones
- *Backrunning*
  - Consiste en ver una transaccion pendiente, procesarla, y enviar una transacción para que se ejecute justo después que esa
    - Ejemplo: arbitraje
- *Sandwitch trading*
  - Consiste en ver una transacción pendiente, procesarla, y enviar una transacción para que se ejecute justo antes y otra que se ejecute justo despúes
    - Ejemplo: *riskless profit* de un swap desprotegido

Módulo XIV: Otros productos financieros

14.1 Staking

- Consiste en:
  - Tomar un token determinado
  - Colocarlo en un depósito
  - A cambio de un rendimiento, puede ser fijo o variable, en tokens
- Incentiva:
  - Al usuario, porque recibe tokens simplemente por tener tokens
  - Al protocolo, porque es una estrategia de incremento de usuarios
- Desincentiva:
  - Al trading excesivo: funciona como una manera de reducir el circulante en el ambiente
  - Al usuario, porque sus tokens están bloqueados (a veces por períodos fijos)
  - Al protocolo, porque “debe emitir” para cubrir los intereses a pagar

14.2 *Liquidity providing*

- Consiste en “hacer market making” en los pools
- Uno deposita tokens en una proporción determinada (depende del punto de operación del pool) y recibe los fees prorrateados por la proporción de la liquidez que representa.
- *Impermanent loss*
- Dependiendo el modelo de pool, la liquidez puede ser concentrada (Uniswap V3), o distribuida (Uniswap V2) entre otros modelos

14.3 *Yield Farming*

- Consiste en:
  - Tomar un token de liquidity providing
  - Stakear ese token de liquidity providing para obtener mayor share del pool
- Sirve para:
  - Aumentar la liquidez en los pools, ya que incentiva a los usuarios a ofrecer liquidez
  - También a mantener a los usuarios ofreciendo liquidez por más tiempo
- Riesgos:
  - Nuevamente, *impermanent loss*
  - Muy competitivo. ¡Requiere mucho seguimiento!

14.4 *Lending / Borrowing*

- Existen protocolos como AAVE y Compound que permiten tomar préstamos en tokens.
- Son préstamos sobrecolateralizados
- También permiten prestar los tokens para obtener un rendimiento
- Se puede integrar esta idea a smart contracts, trading, etc.

Conclusiones y resumen

Resumen

- Módulo X: Discusión de la tarea
  - Modos de suscripción, nuevo bloque y evento
- Módulo XI: *Flash Loans* y margin trading
  - ¿Cómo resolvemos la cuestión operativa de tomar estos arbitrajes?
- Módulo XII: Modelos de arbitraje para AMMs de UNIV2
  - ¿Cómo resolvemos la cuestión matemática de tomar estos arbigrajes?
- Módulo XIII: Riesgos de ejecución
  - No es tan fácil como parece…
- Módulo XIV: Otros productos financieros
  - Distintas ideas del mundo defi

Conclusión

- Existen limitaciones para operar entre el mundo cex y el mundo dex
- Pudimos implementar casi end to end un sistema de monitoreo de arbitrajes, tanto entre dexes, entre cexes, y cex-dex
- Discutimos riesgos operativos asociados
- Revisión general de otros productos financieros, para quien tenga interés

¡Muchas gracias!