FELIPY PENHA BOTEGA

ESTUDO SOBRE ARBITRAGEM E TRANSAÇÕES DE ALTA FREQUÊNCIA EM EXCHANGES DESCENTRALIZADAS

LAVRAS-MG
2023

FELIPY PENHA BOTEGA

ESTUDO SOBRE ARBITRAGEM E TRANSAÇÕES DE ALTA FREQUÊNCIA EM EXCHANGES DESCENTRALIZADAS

Monografia apresentada à Universidade Federal de Lavras, como parte das exigências do Curso de Engenharia de Controle e Automação, para obtenção do título de Bacharel.

Prof. Dr. Arthur de Miranda Neto
Orientador

LAVRAS-MG
2023

FELIPY PENHA BOTEGA

ESTUDO SOBRE ARBITRAGEM E TRANSAÇÕES DE ALTA FREQUÊNCIA EM EXCHANGES DESCENTRALIZADAS

STUDY ON ARBITRAGE AND HIGH FREQUENCY TRADING IN DECENTRALIZED EXCHANGES

Monografia apresentada à Universidade Federal de Lavras, como parte das exigências do Curso de Engenharia de Controle e Automação, para obtenção do título de Bacharel.

APROVADA em 08 de março de 2023.
Prof. Dr. Arthur de Miranda Neto - UFLA
Prof. Dr. Danilo Alves de Lima - UFLA
Prof. Dr. Renato Vieira dos Santos - UFLA
MSc. Leomar Santos Marques - UFLA

Prof. Dr. Arthur de Miranda Neto
Orientador

LAVRAS-MG
2023

AGRADECIMENTOS

É com grande satisfação e gratidão que apresento este Trabalho de Conclusão de Curso (TCC) e gostaria de exprimir meus mais sinceros agradecimentos a todos aqueles que, de alguma forma, contribuíram para a sua realização. Em primeiro lugar, quero agradecer à Universidade Federal de Lavras pela oportunidade de realizar meu TCC nesta prestigiada instituição, bem como pelas excelentes condições acadêmicas e estruturais proporcionadas para o desenvolvimento deste trabalho. Aos meus colegas de turma, agradeço pelo apoio constante e incentivo ao longo desta jornada acadêmica, e por todos os momentos de troca de ideias e aprendizado compartilhados. E, por fim, mas não menos importante, ao meu orientador, quero expressar minha mais profunda gratidão pela dedicação, paciência e valiosos ensinamentos prestados. Sua orientação foi fundamental para a consecução deste trabalho e eu lhe sou profundamente grato. Espero que este trabalho contribua para o avanço do conhecimento na área e que possa ser útil para futuros estudiosos. Agradeço a todos novamente por seus valiosos contributos.

RESUMO

Com a evolução tecnológica e o surgimento de redes distribuídas e descentralizadas, que possibilitam a criação de ativos digitais como alternativa e complemento ao mercado financeiro tradicional, as tecnologias já existentes e amplamente utilizadas em negociações, foram adaptadas ao novo mercado de criptoativos. O *High Frequency Trading* (HFT) é uma técnica de negociação que utiliza a capacidade computacional no processo de tomada de decisão e ação, com base em diversas estratégias fundamentadas na condição de rentabilidade. Neste trabalho, investiga-se de forma sistemática a estrutura tecnológica envolvida nas exchanges descentralizadas, destacando sua importância para o mercado de ativos digitais e como as tecnologias de HFT podem e estão sendo aplicadas diante das oportunidades desse mercado. Para isso, foi explorado os modelos de HFT existentes, os principais fatores que influenciam nas transações, diante das consequências das tecnologias envolvidas, como a blockchain. Além disso, foi abordado o desenvolvimento de um dos modelos de HFT, a arbitragem, com o qual foi possível identificar as principais dificuldades e os pontos a serem melhorados afim de obter um algoritmo competitivo e rentável para ser aplicado ao mercado das finanças descentralizadas.

Palavras-chave: Ativos digitais; High Frequency Trading; Exchange descentralizadas; Arbitragem; Blockchain.

ABSTRACT

With technological evolution and the emergence of distributed and decentralized networks, which allow the creation of digital assets as an alternative and complement to the traditional financial market, existing technologies widely used in trading have been adapted to the new crypto asset market. High Frequency Trading (HFT) is a trading technique that uses computational power in the decision-making and action process, based on various profitability-based strategies. In this paper, we systematically investigate the technological structure involved in decentralized exchanges, highlighting their importance to the digital asset market and how HFT technologies can and are being applied in the face of opportunities in this market. To do so, we explored existing HFT models, the main factors that influence transactions, given the consequences of the technologies involved, such as blockchain. Additionally, we addressed the development of one of the HFT models, arbitrage, which allowed us to identify the main difficulties and points to be improved in order to obtain a competitive and profitable algorithm to be applied to the decentralized finance market.

Keywords: Digital assets; High Frequency Trading; Decentralized exchanges; Arbitrage; Blockchain.

LISTA DE ILUSTRAÇÕES

LISTA DE SIGLAS

AMM	Automated Market Makers
API	Application Programming Interface
AWS	Amazon Web Services
BASH	Bourne Again Shell
CEX	Centralized Exchanges
CSS	Cascading Style Sheets
DAOs	Decentralized Autonomous Organizations
DApps	Descentralized Applications
DeFi	Decentralized Finances
DEX	Descentralized Exchanges
HFT	High Frequency Trading
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
ICO	Initial Coin Offering
IDO	Initial Dex Offering
JSON	JavaScript Object Notation
LP	Liquidity Provider
MEV	Maximum Extractable Value
NPM	Node Package Manager
PGAs	Priority Gas Auctions
RPC	Remote Procedure Call

SUMÁRIO

1 INTRODUÇÃO

1.1 Contextualização

A prática da negociação tem seus relatos desde a mitologia, com o uso de técnicas persuasivas que buscavam maximizar vantagens diante de seus interesses. Com o passar dos séculos e a expansão do comércio, novas oportunidades surgiram e as técnicas foram adaptadas aos novos cenários. Atividades financeiras, como as iniciadas na Holanda durante o século XVII, com o comércio de contratos de tulipas, marcam o início de uma nova era de negociações. Conhecidas como especulações, essas atividades se baseiam no comércio de ativos com previsão de venda futura, diante da promessa de altos rendimentos. Essa nova prática de negócios ganhou força e se desenvolveu, de forma a ser presente até os dias atuais (FISHER; URY, 1991).

Com o surgimento de novos tipos de ativos, atrelados a diversas áreas e com diferentes promessas, os locais onde eram realizadas as operações também tiveram que se adaptar. Nesse contexto, surgem as bolsas de valores, as quais possuem como principal característica a concentração de uma ampla variedade de ativos e o poder de realizar operações de compra e venda com baixa distorção do valor. Criada em 1531, na Bélgica, a Bolsa de Antuérpia é considerada a primeira bolsa oficial, onde as operações eram realizadas de forma física em locais nos quais se aglomeravam as partes interessadas nos papéis. Esse modelo de mercado se expandiu e se perpetuou até o século XX. Nesse período, novas ferramentas de negociação foram implementadas e adaptadas ao cenário do momento (ROLNIK, 2010).

Com o desenvolvimento tecnológico, as metodologias utilizadas para realizar transações evoluíram. Operações que antes eram realizadas de forma presencial passaram a ser efetuadas por linhas telefônicas. Esse novo modelo de mercado permitiu que as operações fossem agilizadas, contribuindo ainda mais para a ampliação do mercado financeiro. Em 2008, ocorreu uma nova ruptura nos paradigmas da bolsa, com a transição do modelo de pregão viva-voz para o eletrônico, o qual modificou consideravelmente a forma de realizar operações. As novas negociações deixaram de ter processos que antes eram efetuados de forma manual e suscetíveis a erros humanos, dando espaço a um modelo escalável a nível global, com o uso da internet e do protocolo TCP/IP (LO; MACKINLAY, 1990).

A partir do momento em que o mercado financeiro passou a operar de forma digital, novas técnicas foram criadas e utilizadas para negociações. A possibilidade de realizar operações e obter dados de forma instantânea motivou o uso de técnicas matemáticas e estatísticas como forma de prever a tendência do comportamento dos preços. Além disso, o uso de ferramentas de transação conectadas diretamente à bolsa de valores em tempo real auxiliou

na automatização dos processos e na implementação de novas técnicas baseadas em alta frequência, iniciando, dessa forma, uma nova corrida por infraestruturas de alto poder de processamento e baixa latência (ALMGREN; CHRISS, 2000).

Em 2015, surgiu um novo tipo de plataforma, criada por um grupo de desenvolvedores, entre eles Vitalik Buterin, conhecida como rede Ethereum. Utilizando tecnologias de blockchain semelhantes às do Bitcoin, e com o grande diferencial de possuir uma máquina virtual de Turing Complete, iniciou-se a aplicação dos conceitos de finanças descentralizadas. Com a Ethereum, a execução de contratos inteligentes e o início das aplicações descentralizadas tornaram-se possíveis. Não demorou muito e os primeiros ativos digitais começaram a aparecer, alguns inclusive com representação em moedas fiduciárias como o dólar (BUTERIN, 2014).

O crescente interesse pela tecnologia, atrelado ao desejo de altos rendimentos sem a influência de um órgão central, favoreceu a criação de diversos tipos de aplicações descentralizadas, oferecendo soluções para problemas de diversas áreas da sociedade. Junto a essas aplicações, tokens atrelados a elas foram criados, como promessa de serem ativos de pagamento, utilidade ou como representatividade de algo único. Dentro desse contexto, surgiram os aplicativos de finanças descentralizadas ou DeFi (descentralized finance), sendo as DEX (exchanges descentralizadas), um dos tipos de aplicação, que desenvolveu uma grande importância dentro deste ecossistema.

1.2 Justificativa

A negociação de ativos criptográficos em exchanges descentralizadas (DEX) tem crescido significativamente, oferecendo aos usuários a oportunidade de negociar seus ativos sem a necessidade de transferir a custódia para terceiros. Isso é possível graças à interação com contratos inteligentes que garantem segurança, resistência à censura e imutabilidade nas transações.

A arquitetura das DEX é baseada em tecnologia blockchain, o que as diferencia das exchanges centralizadas (CEX), que normalmente fazem uso de uma arquitetura baseada em banco de dados. Além disso, a transações de alta frequência em DEX requer diferentes algoritmos baseados em novos tipos de estratégias, tornando essa área promissora diante das oportunidades. Os algoritmos de HFT podem melhorar a eficiência e a liquidez do mercado de ativos digitais, ajudando a reduzir os spreads de oferta e demanda e a aumentar a profundidade do mercado, o que torna a compra e venda de ativos mais rápida e eficiente.

Quando combinados com as DEX, os HFT podem impulsionar ainda mais o desenvolvimento do mercado, criando mais oportunidades para os usuários e facilitando a execução das transações. Assim, a utilização de HFT em DEX pode ser uma estratégia viável e promissora para investidores interessados em transações de alta frequência no mercado de ativos digitais. Diante disso, a necessidade de um estudo sobre as transações de alta frequência em exchange descentralizadas é ainda mais relevante e justificável.

1.3 Objetivos

Este trabalho tem como objetivo principal investigar e compreender o funcionamento das transações de alta frequência quando aplicadas em exchanges descentralizadas (DEX). Para isso, será feito um estudo das tecnologias envolvidas, como as redes blockchain e os contratos inteligentes. Será realizada uma análise das estratégias de transação de alta frequência, com ênfase na arbitragem, com a finalidade de compreender como podem ser aplicadas nas DEX e quais os fatores que influenciam o desempenho dos algoritmos e os diferenciam das exchanges centralizadas (CEX), em virtude do uso da tecnologia blockchain.

Buscando identificar os principais desafios na implementação das estratégias de HFT, será abordado o desenvolvimento de um algoritmo de arbitragem entre exchanges descentralizadas. A partir de então, será possível compreender os desafios e identificar os principais pontos a serem melhorados para obter um algoritmo de alto desempenho.

Nesse contexto, é importante destacar que a utilização de transações de alta frequência em exchanges descentralizadas tem se tornado cada vez mais comum. Por isso, é fundamental que sejam realizados estudos sobre o tema, a fim de garantir uma maior segurança e eficiência nesse ambiente. A análise dessas transações pode contribuir para a criação de novos modelos de negócios e para o fortalecimento do mercado de finanças descentralizadas.

2 REFERENCIAL TEÓRICO

O presente capítulo aborda os temas fundamentais no contexto das transações de alta frequência em exchanges descentralizadas. Com o crescente uso de criptomoedas e tecnologias blockchain, o mercado de criptomoedas tem visto um aumento significativo na atividade de negociação de alta frequência (HFT). Nesse sentido, é importante compreender as tecnologias subjacentes a esse mercado, incluindo as redes blockchain e a plataforma Ethereum, as quais fornecem um ambiente seguro e transparente para a realização dessas transações.

As aplicações descentralizadas (DApps) também têm desempenhado um papel importante, oferecendo uma ampla gama de serviços financeiros sem a necessidade de intermediários. Entre eles, destaca-se as exchanges descentralizadas que são fundamentais para a compra e venda de criptomoedas, permitindo que os usuários realizem transações diretamente em um ambiente seguro e transparente. Ao explorar esses tópicos em detalhes, este capítulo fornece uma visão abrangente das tecnologias e processos subjacentes às transações de alta frequência em exchanges descentralizadas.

2.1 Redes blockchain

O entendimento do funcionamento das redes blockchain é de grande importância, especialmente quando se trata de transações de criptomoedas, realizadas por meio de exchanges descentralizadas. Esses sistemas distribuídos permitem o registro e validação de transações de forma descentralizada, aumentando assim a transparência e a segurança. Desse modo, é essencial compreender o funcionamento das redes blockchain para entender o potencial dessa tecnologia em outras áreas, como em contratos inteligentes e governança descentralizada.

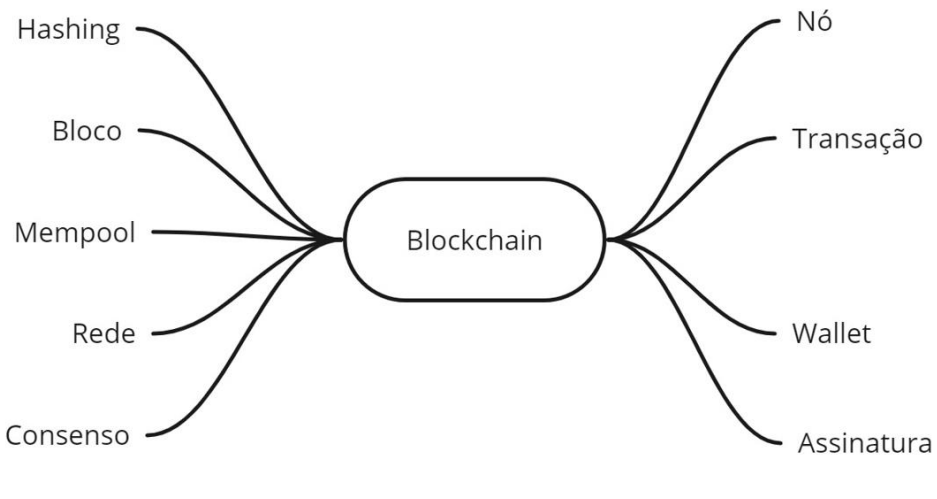
A blockchain pode ser definida como um grande livro-razão, que permite a inserção e distribuição de registros de forma permanente e irreversível. A cadeia de blocos, ou blockchain, pode ser comparada a um grande banco de dados digital, com características de ser pública, distribuída, descentralizada, remota, inviolável e com a capacidade de inserir dados de diversos tipos. No caso das criptomoedas, como o Bitcoin, os dados referem-se às transações de quantidades de ativos e possuem informações que garantem a autenticidade das transações com base nos protocolos da rede (NAKAMOTO, 2008).

O sistema blockchain é formado por uma arquitetura de "cadeia de blocos", na qual um conjunto de transações é inserido dentro de um desses blocos, que são salvos e fechados por uma chave criptográfica. Em seguida, um novo bloco é criado e irreversivelmente vinculado aos blocos existentes. Os registros inseridos podem ser consultados gratuitamente, atribuindo, dessa forma, características como transparência e rastreabilidade ao sistema. A velocidade e a

eficiência também se tornam uma vantagem significativa, uma vez que, a partir dos contratos inteligentes, as ações podem ser realizadas de forma automatizada, dependendo apenas do tempo de mineração dos blocos (TAPSCOTT, D.; TAPSCOTT, A., 2016).

A arquitetura das redes blockchain pode ser explicada por meio de camadas, informadas na figura 1, que se relacionam e possuem seus respectivos comportamentos, de forma a caracterizar e garantir o funcionamento da rede de acordo com seus protocolos estabelecidos.

Figura 1 – Camadas que compõem a tecnologia blockchain.



Fonte: Do Autor (2023).

2.1.1 *Hashing*

O processo de *hashing* consiste na conversão de dados, em que uma entrada de tamanho arbitrário gera uma saída de tamanho fixo. Assim, independentemente da quantidade de dados envolvidos no processo, o hash gerado sempre será de um mesmo tamanho. Esse processo é realizado por meio de funções matemáticas conhecidas como funções hash, normalmente utilizadas para verificação de autenticidade de informações (NASH, 2020).

Um exemplo de destaque é o algoritmo criptográfico SHA-256, amplamente utilizado em redes blockchains. Isso ocorre, devido ao fato de seu resultado ser sempre uma sequência alfanumérica de tamanho 64, do tipo determinístico e com uma codificação de 256 bits, o que garante alta segurança e resistência ao fenômeno de colisões, que consiste na geração de um mesmo *hash* para entradas diferentes. Além disso, o SHA-256 possui a característica de ser unidirecional, não podendo ser facilmente revertido sem o uso de uma grande capacidade computacional em um amplo período de tempo (NASH, 2020).

Figura 2 - Processo de hashing.

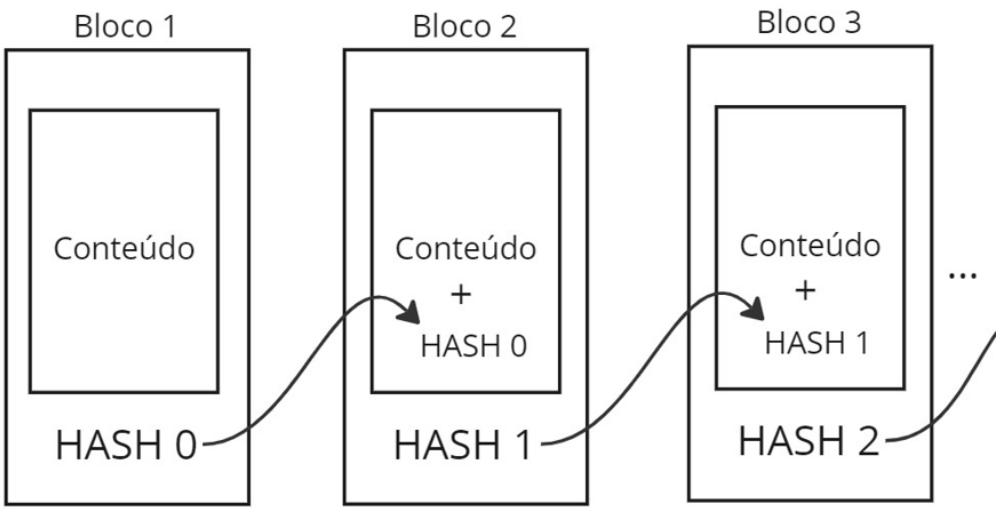
Fonte: (NASH, 2020).

Quando se refere às redes blockchains, as funções *hash* criptográficas têm uma grande importância. Além dos aspectos de segurança, as funções são fundamentais no processo de mineração, sendo essenciais na geração de novos endereços, chaves e no vínculo criptográfico entre blocos. Além disso, a capacidade de lidar e condensar uma grande quantidade de informação, torna essa tecnologia essencial no processo de armazenamento de dados (NASH, 2020).

2.1.2 Bloco

Os blocos de uma blockchain podem ser entendidos como um container, no qual são armazenados os dados referentes às transações. Cada bloco é construído a partir do anterior e inclui informações de forma a estabelecer um vínculo e garantir a relação entre eles. O relacionamento entre blocos é realizado por meio das funções *hash*. Sempre que um bloco é finalizado, é gerado um *hash* referente a ele, que no bloco seguinte será utilizado em conjunto aos dados das transações para gerar um novo *hash*. Como as chances de encontrar um conjunto de dados que resultem nas mesmas saídas são extremamente baixas e comprovadas pelo fato de não existirem colisões SHA-256 conhecidas, ao incluir o *hash* do bloco anterior no atual, tentativas de edição e violação dos blocos anteriores se tornam imediatamente perceptíveis na rede, esse processo é exemplificado na Figura 3 (NARAYANAN et al., 2016).

Figura 3 - Encadeamento de blocos.



Fonte: Do Autor (2023).

2.1.3 Mempool

O mempool consiste na contração das palavras "*Memory Pool*", e é constituído por um nó, ou seja, uma máquina conectada à rede que armazena e compartilha informações sobre ela. O nó mempool é um mecanismo de memória temporária intermediária, na qual são armazenadas informações sobre as transações não confirmadas, que estão pendentes na rede e ainda não foram incluídas em um bloco. Esse mecanismo é de grande importância, pois as transações não ocorrem de forma imediata e é necessário um tipo de zona de *buffer* para o armazenamento das informações (SOMPOLINSKY; ZOHAR, 2015).

Quando uma transação é iniciada, as informações referentes a ela são propagadas a partir de um nó da rede, e então as informações são repassadas para os nós pareados com ele, que posteriormente realizam o mesmo processo até que tenham sido amplamente propagadas. As transações são verificadas diversas vezes, por diversos nós, para garantir a autenticidade das assinaturas e dos ativos a serem transacionados. Se não atenderem às condições, a transação é rejeitada (SOMPOLINSKY; ZOHAR, 2015).

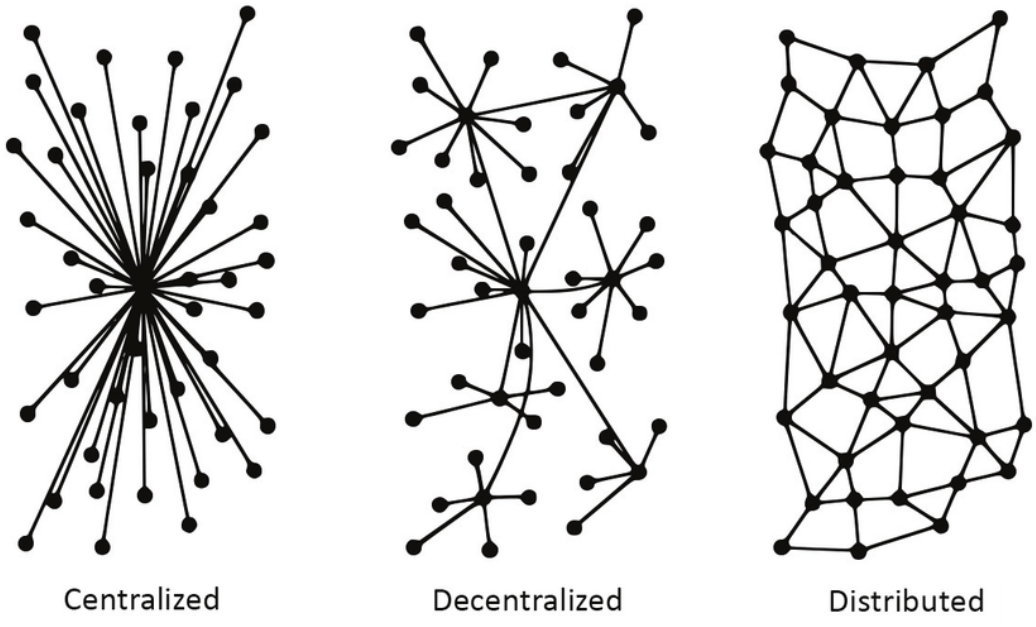
Os mineradores são motivados pelo valor das taxas. Dessa forma, transações com taxas mais altas são priorizadas no momento de serem gravadas no bloco. As taxas são definidas pelo usuário que iniciou a transação e o valor segue a lei da oferta e demanda. Como os blocos normalmente possuem tamanho fixo, em um período de pico, com alta demanda por transações, o valor tende a se elevar, e o contrário também acontece (SOMPOLINSKY; ZOHAR, 2015).

2.1.4 Rede

As redes podem ser definidas como um conjunto de dois ou mais dispositivos interligados, com capacidade de se comunicar e compartilhar informações por meio de um conjunto de regras definidas, os protocolos de redes. Diante desse conceito, é possível caracterizar as redes conforme sua arquitetura.

As redes centralizadas possuem seus dados armazenados em um único servidor, enquanto os demais nós atuam como clientes. Já nas redes descentralizadas, os dados não estão armazenados em um ponto central, mas sim em diversos servidores, nos quais os clientes podem estabelecer a conexão. Junto ao conceito de descentralização, também é possível estabelecer o conceito das redes distribuídas, em que cada nó pertencente à rede possui uma cópia dos dados que são compartilhados de forma sincronizada e possuem o mesmo nível de autoridade, os tipos de redes estão exemplificados na Figura 4 (TANENBAUM; STEEN, 2017).

Figura 4 - Tipos de rede.



Fonte: (TANENBAUM; STEEN, 2017).

As redes *peer-to-peer*, presentes nas blockchains, são redes distribuídas e descentralizadas. Não existindo um administrador central, é possível que, ao requisitar um serviço, este seja feito em qualquer nó presente na rede, que posteriormente propagará as informações para os demais nós pertencentes à ela. Esse tipo de arquitetura permite que o

funcionamento seja contínuo e, mesmo que alguns nós deixem a rede, ainda é garantido o funcionamento e a integridade da mesma (TANENBAUM; STEEN, 2017).

2.1.5 Consenso

O mecanismo de consenso é a forma de se tomar decisões dentro da blockchain, por meio de um acordo entre os nós pertencentes à rede que seguem regras pré-estabelecidas com o uso de algoritmos. Sem um consenso, é impossível que uma decisão seja tomada em uma rede distribuída, na qual todos os nós têm o mesmo nível de autoridade. Diante dessa ideia, algoritmos foram criados como forma de implementação do conceito, buscando obter sucesso e eficiência, sem comprometer a segurança da rede (TANENBAUM; STEEN, 2017).

Os algoritmos de consenso são elementos crucias para o bom funcionamento de uma rede blockchain, pois garantem que as ações sejam tomadas de forma a beneficiar todos os envolvidos na rede. Os principais algoritmos de consenso aplicados são chamados de *Byzantine fault tolerance* (tolerância a falhas bizantinas). O chamado Problema dos Generais Bizantinos descreve um conflito referente ao consenso dos generais ao tomar a decisão de atacar uma cidade cercada ou recuar. A decisão de atacar somente pode ser feita de forma unânime. Diante disso, surge o problema das interferências maliciosas, caso um dos participantes seja malicioso e tome decisões contrárias ao grupo, todos os envolvidos são comprometidos. Para superar uma potencial falha em uma blockchain, diversos protocolos de consenso foram criados e denominados de *Byzantine fault tolerance*, nos quais o critério de consenso é baseado na maioria e não mais na unanimidade (NARAYANAN et al., 2016).

O primeiro algoritmo de consenso implementado para blockchains de criptoativos foi o *Proof of Work*, utilizado na rede do Bitcoin. O *Proof of Work* deu início ao processo conhecido como mineração, que consiste no ato de resolver algoritmos matemáticos definidos pelo protocolo da rede. No processo de mineração, um alto poder computacional é requisitado, a fim de resolver os problemas rapidamente e se tornar o nó validador do bloco. O responsável por gravar as transações, obtém as recompensas estabelecidas pelo protocolo e a quantia referente às taxas pagas pelas transações solicitadas pelos usuários da rede (NAKAMOTO, 2008).

O *Proof of Work* possui grandes vantagens, principalmente devido ao fato de ter sido aprovado e testado ao longo do tempo, a descentralização e a facilidade de se tornar um nó validador também é uma característica importante do protocolo. Porém, devido ao alto custo energético gasto no processo de mineração, esse tipo de protocolo de consenso tem sido altamente criticado por ambientalistas. Além disso, as barreiras de entrada se tornam cada vez

maiores à medida que mais mineradores se juntam à rede, e ocorre uma corrida por equipamentos com poder computacional cada vez maior (NAKAMOTO, 2008).

Nos protocolos *Proof of Work*, o incentivo a agir honestamente e a favor da rede é o alto valor investido no processo de mineração, o minerador não receberá as recompensas caso não minere blocos válidos. Além disso, caso ocorra falhas na rede, o risco de se investir nos ativos relacionados a ela aumenta, causando uma fuga de investidores que compromete diretamente o retorno dos envolvidos no processo de mineração (NAKAMOTO, 2008).

O *Proof of Stake* consiste em um protocolo de consenso, no qual o poder computacional não é tão relevante e o conceito de mineração não existe. Em vez disso, é fornecido recursos internos, como o próprio ativo da rede, como prova de participação. As regras variam de acordo com o protocolo da rede, mas geralmente seguem o conceito de colocar uma quantidade de ativos bloqueado por um período de tempo, esse processo é chamado de *staking*. Nesse mecanismo de consenso, o próprio protocolo escolhe o nó validador, por meio de um tipo de sorteio entre os participantes, e as chances de ganhar são proporcionais à quantidade de ativo depositado.

O *Proof of Stake* apresenta benefícios, como o custo reduzido de infraestrutura e maior dificuldade de fraude, já que o fraudador necessitaria possuir 51% das moedas da rede, o que seria caro e o próprio ato de fraude poderia prejudicar a precificação do ativo. Contudo, há um ponto negativo, que é a tendência para a concentração dos ativos, uma vez que os detentores de maior quantidade têm maiores chances de se tornarem validadores e obter as recompensas da validação dos blocos. Recentemente, o protocolo foi implementado em redes maiores e altamente descentralizadas, como a Ethereum, que efetuou a migração do *Proof of Work* para o *Proof of Stake* em um processo de *fork* realizado no segundo semestre de 2022. Apesar do sucesso na migração, o curto período de operação ainda gera incertezas quanto ao seu desempenho futuro (SOMPOLINSKY; ZOHAR, 2015).

Além dos algoritmos de consenso mencionados, existem outros, como o *Practical Byzantine Fault Tolerance*, *Delayed Proof of Work, Delegated Proof of Stake*, *Proof of Authority*, *Proof of Activity* e o *Proof of Burn*. Cada protocolo tem suas vantagens e desvantagens que se destacam de acordo com cada necessidade. No entanto, vale ressaltar que o protocolo *Proof of Work* continua sendo amplamente aceito, devido ao fato de ser utilizado na maior rede atual, Bitcoin, e tem sido comprovado ao longo do tempo (NAKAMOTO, 2008).

2.1.6 Nó

A definição de um nó pode variar de acordo com o contexto da aplicação, de forma geral, pode ser entendido como um ponto responsável pela transmissão, recepção ou criação de uma mensagem. No contexto das redes blockchain, os nós constituem um dos pilares fundamentais na arquitetura do sistema, sendo responsáveis por executar o software do protocolo, de forma sincronizada com os demais nós participantes. Assim, é garantido o funcionamento e segurança da rede e é introduzido o conceito da descentralização (NARAYANAN et al., 2016).

As funções dos nós também podem variar de acordo com o protocolo de funcionamento da rede. Em redes como a Bitcoin, que possui um protocolo de consenso do tipo *Proof of Work*, alguns nós exercem a função de minerador, sendo responsáveis pela mineração e validação dos blocos. Uma das características principais desse tipo de nó é o alto poder de processamento que as máquinas que o executam possuem. Esse tipo de nó pode ser executado de forma individual ou em grupos, conhecidos como *miner pools* (piscinas de minerações), que constituem um grupo de computadores que executam a mineração de forma paralelizada, buscando otimizar e aumentar a capacidade de processamento para a resolução de algoritmos de *Proof of Work* (NARAYANAN et al., 2016).

Os *Full Nodes* são responsáveis por dar suporte e segurança às redes blockchain, sendo responsáveis pelo processo de verificação de transações e formação dos blocos da rede, que seguem as regras do protocolo de consenso. Normalmente, os *Full Nodes* possuem uma cópia completa da blockchain da rede a qual fazem parte, essa cópia contém todo o histórico de transações, desde o início da operação da rede, necessitando assim, de uma maior capacidade de armazenamento de dados e um maior tempo de download e sincronização dos dados (SCHRIJVERS et al., 2019).

Os *Full Nodes* desempenham um papel fundamental para o ecossistema da rede, contribuindo diretamente para o nível de segurança. Alguns usuários operam *Full Nodes* como forma de manter a integridade do ecossistema, podendo ser de forma pública, permitindo receber requisições, ou invisíveis, em casos de operarem por trás de um firewall ou fazendo uso de protocolos como o Tor, não permitindo, dessa forma, receber conexões (SCHRIJVERS et al., 2019).

2.1.7 Transação

A blockchain pode ser definida como um sistema baseado em transações. O propósito das ações dentro de uma rede blockchain envolve a transferência de ativos entre duas partes, que é realizada seguindo o fluxo de operação da rede.

Para iniciar uma transação, é necessário inicialmente que sejam identificadas as partes envolvidas, e que o processo possa ser verificado de forma a garantir que apenas o detentor dos ativos possa realizar a transferência. Diante disso, o uso do sistema de criptografia assimétrica se torna de grande importância.

Também chamado de criptografia de chave assimétrica, ele consiste no uso de chaves públicas e privadas, que são formadas por um conjunto de caracteres gerados aleatoriamente. O sistema de chave pública é utilizado para encriptar informações, enquanto as chaves privadas fazem o processo inverso, permitindo que as informações sejam lidas. As chaves possuem entre 1024 e 2048 bits, o que torna quase impossível gerar a chave privada a partir da pública (NARAYANAN et al., 2016).

O algoritmo mais conhecido para gerar as chaves é denominado RSA (Rivest-Shamir-Adleman), em que as chaves são geradas com base na multiplicação de dois números primos. O resultado é público, mas o processo de fatoração pode demandar anos. Quando uma nova carteira é criada, um par de chaves público e privado é gerado, a partir desse par uma nova camada de segurança é aplicada e um endereço público é criado, o qual é utilizado para a identificação das partes envolvidas na transação (NARAYANAN et al., 2016).

O endereço pode ser compartilhado com segurança, enquanto a chave privada deve ser mantida em segurança e é utilizada para criar assinaturas digitais que serão aplicadas ao executar uma transação. Ao verificar a autenticidade das transações pelo ECDSA (Algoritmo de Assinatura Digital de Curva Elíptica), ela poderá ser inserida no bloco pelo validador. Esse mecanismo de assinaturas digitais garante que apenas o detentor da carteira possa realizar a transferência dos ativos (RIVEST; SHAMIR; ADLEMAN, 1978).

As transações de ativos são realizadas a partir de *inputs* (entradas) e *outputs* (saídas). Quando uma nova transação é realizada, o usuário faz uso de uma ou mais UTXOs (*Unspent Transaction Outputs*), que são saídas de transação não gastas que podem ser usadas como entradas de uma nova transação. Quando o processo é iniciado, uma busca é feita na carteira do usuário por UTXOs que superem o valor a ser enviado. Caso a UTXO tenha um valor acima, duas UTXOs são geradas, uma referente ao valor a ser enviado e uma segunda referente ao troco que é devolvido ao usuário. As UTXOs consumidas não podem ser mais utilizadas em

uma nova transação, e as saídas geradas se tornam novos UTXOs (RIVEST; SHAMIR; ADLEMAN, 1978).

2.1.8 Carteiras

As carteiras consistem em uma ferramenta que permite que os usuários interajam com a rede blockchain, possibilitando que os ativos sejam armazenados e transacionados. Por meio das carteiras, é feita a conexão com os nós da rede, possibilitando, então, que sejam realizadas operações. Em algumas carteiras, como a "Metamask", a conexão ao nó é feita de forma indireta, com o uso de RPC (*Remote Procedure Call*), no qual uma interface gráfica, via extensão do navegador, realiza a requisição para um servidor que posteriormente estabelecerá a comunicação com o nó (COINMAMA..., 2021).

As carteiras podem ser divididas em grupos, sendo eles: software, hardware e *paper wallets*, e classificadas de acordo com seu funcionamento, sendo as do tipo "*hot*" aquelas que geram as chaves através da conexão com a internet, e do tipo "*cold*" as que conseguem gerar de modo offline (BLOCKGEEKS..., c2018a).

As carteiras de software são o tipo mais comum e podem ser caracterizadas como:

- Web: Com interface via browser, sendo de simples acesso, porém de menor segurança devido ao fato de ser gerenciada por terceiros.
- Desktop: Que necessita a instalação do software localmente, as chaves são armazenadas na própria máquina, possuindo, dessa forma, maior segurança.
- Móvel: Semelhantes ao tipo desktop, porém de acesso via dispositivos móveis.
- Hardware: Consistem em dispositivos físicos que possuem o mecanismo de gerar e armazenar as chaves sem a necessidade de conectar à internet. Apesar de serem menos acessíveis e exigir um custo para adquirir, atualmente, são uma das alternativas mais seguras.
- Paper: As *paper wallets* são basicamente um pedaço de papel no qual são gravadas as chaves e endereço referente à carteira. Devido à dificuldade de ser utilizada, por necessitar de um alto conhecimento técnico ao realizar as operações diretamente nos nós da rede, tem sido pouco utilizada.

2.1.9 Assinatura

Uma transação é iniciada em uma rede blockchain a partir do momento em que a carteira realiza a assinatura, confirmando ser realmente o responsável pela origem do ativo. O processo

de incluir uma assinatura em um bloco é feito a partir de algoritmos complexos, destacando-se o RSA, ECDSA e Schnorr. O algoritmo RSA é o padrão para criptografia há muitos anos, porém com o surgimento do ECDSA, a hegemonia de uso passou a decair. Isso se deve ao fato de ambos oferecerem o mesmo nível de segurança, de 112 bits. Porém, isso é possível com apenas 224 bits de tamanho das chaves públicas ECDSA, enquanto o RSA necessita de 2048 bits (GEEKSFORGEEKS..., 2021).

O algoritmo de Schnorr possui um funcionamento semelhante ao ECDSA, mas apresenta algumas vantagens, sendo mais simples e atraente para transações com processo do tipo *signature aggregation* (agregação de assinaturas). Algumas redes, como a do Bitcoin, iniciaram seu funcionamento optando pelo ECDSA, devido às patentes do Schnorr terem expirado em 2008, poucos meses antes do lançamento da rede. A rede continuou operando utilizando o algoritmo até novembro de 2021, data em que houve a atualização da rede, denominada Taproot, passando então a fazer o uso de assinaturas por Schnorr (BITCOIN MAGAZINE..., 2018).

2.2 Ethereum

A Ethereum é uma rede que se baseia na tecnologia blockchain. Ao contrário da rede Bitcoin, a Ethereum tem em sua arquitetura uma máquina virtual que pode executar algoritmos em uma linguagem de programação *Turing Complete*, ou seja, significa que é possível executar qualquer etapa lógica em uma função computacional, por mais complexa que seja, com tempo e memória suficiente. Compreender o funcionamento da rede Ethereum e das tecnologias envolvidas, assim como os tipos de aplicações desenvolvidas, é crucial para quem deseja entender e aproveitar as vantagens oferecidas pela plataforma, especialmente no contexto de transações de alta frequência.

A Ethereum pode ser definida como uma rede ponto a ponto de máquinas virtuais e é considerada a segunda geração das redes blockchain de ativos digitais. A partir da rede Ethereum, qualquer usuário pode interagir com algoritmos, que foram implementados na rede e que possuem funções para executar ações de forma automatizada. Esses algoritmos são denominados de contratos inteligentes e constituem a principal diferença em comparação com as redes de primeira geração. Além disso, os contratos inteligentes possuem características semelhantes às das transações, como o fato de serem públicos e poderem ser vistos e auditáveis por qualquer interessado (BUTERIN, 2014).

A Ethereum possui sua própria blockchain pública descentralizada, e todas as transações e interações com os contratos passam por todas as camadas de segurança e criptografia da rede.

Além disso, cada nó pertencente à rede contém uma máquina virtual, denominada de Ethereum *Virtual Machine*, responsável pela execução dos contratos inteligentes. Assim como as demais redes blockchain, todas as ações ocorridas na rede e escritas em blocos, demandam alocação de recursos, e o risco assumido sobre é recompensado por meio de pagamentos com a criptomoeda da rede, o ether, por meio do processo de mineração (BLOCKGEEKS..., c2018b).

A ideia principal da Ethereum é fornecer um protocolo construído em cima de uma estrutura blockchain, para a criação de aplicações descentralizadas, fornecendo um conjunto de ferramentas favoráveis à construção rápida e segura e de forma escalável. Tudo isso é feito a partir de uma camada de linguagem de programação de alto nível, construída para abstrair todo o funcionamento dos contratos inteligentes (INVESTOPEDIA..., 2021).

2.2.1 Plataforma de desenvolvimento

A EVM (Ethereum Virtual Machine), assim como as máquinas virtuais em geral, é responsável por criar uma camada de abstração entre o código e a máquina em execução. A EVM consiste de uma pilha virtual de software, capaz de executar os bytecodes gerados por meio da compilação de um código em linguagem Solidity, que é uma linguagem Turing Complete, de alto nível, baseada em linguagens como Java, JavaScript e C++. Isso significa que o algoritmo de máquina é separado da rede, do disco e de outras operações do computador *host*. Cada nó na rede Ethereum executa uma instância EVM, permitindo que eles concordem com o mesmo conjunto de instruções a serem executadas. Além disso, a EVM, além de garantir a portabilidade do software para diferentes máquinas e sistemas operacionais, também é responsável pelo gerenciamento de pilhas, memórias e threads, sendo diretamente responsável pela otimização de recursos computacionais (CONSENSYS..., 2020).

2.2.1.1 Ethereum é *Turing Complete*

O conceito de "*Turing Complete*" surgiu na primeira metade do século XX, com Alan Turing, levando à hipótese de que um dia haveriam máquinas capazes de resolver qualquer problema. Turing afirmou que os computadores não pensam ou processam informações da mesma maneira que os humanos, mas são capazes de resolver problemas por meio de um conjunto de fundamentos de processamento de dados (CONSENSYS..., 2018).

Alan imaginou sua máquina como uma longa fita, na qual as informações são escritas na forma de códigos binários (1 e 0). A máquina também possui um cabeçote de leitura/gravação que se move ao longo da fita e lê cada quadrado individualmente. O código pede à máquina um problema de matemática e a fita leva esse tempo para encontrar a solução.

Quando o cabeçote se move sobre a fita, a máquina segue instruções simples de como reagir. Ele lê a fita, executa as instruções e executa uma ação específica, tudo isso enquanto escreve um novo código. Este novo exemplo de código é a resposta para o problema (CONSENSYS..., 2018).

Uma máquina de Turing hipotética poderia resolver qualquer problema computacional que pudesse ser expresso em código (e que tivesse uma resposta computável). Um dispositivo ou linguagem de programação é dito "*Turing Complete*" se ele pode reproduzir uma máquina de Turing, executando qualquer programa ou resolvendo um problema que uma máquina de Turing pode executar ou resolver. Se um dispositivo ou linguagem de programação não pode fazer isso, é chamado de "*Turing Incomplete*". Diante disso, pode-se afirmar que a principal função da EVM é descobrir qual será o estado geral da Ethereum para cada bloco validado em sua blockchain. Esses conceitos são essenciais para compreender a EVM, pois formam a base de sua arquitetura de funcionamento (99BITCOINS, 2019).

Máquina de Estado Distribuído é a nomenclatura dada à camada dos contratos inteligentes. De forma simplificada, o estado do Ethereum consiste de um grande banco de dados, incluindo todas as contas e saldos em Ether. Simultaneamente, o estado da Ethereum é um estado de máquina, capaz de alterar cada novo bloco e executar qualquer tipo de código de máquina que seguem os protocolos da rede. A EVM é responsável por controlar como a máquina altera seu estado em cada novo bloco, sendo um componente crítico do sistema (BLOCKGEEKS..., c2018c).

2.2.2 Contratos inteligentes

Os contratos inteligentes são programas de computador que permitem a execução de acordos e contratos entre duas ou mais partes, sem a necessidade de intermediários. Esses contratos são autoexecutáveis, o que significa que são executados automaticamente quando as condições definidas em código são atendidas.

Os contratos inteligentes foram introduzidos pela primeira vez em 1994 por Nick Szabo, um pesquisador em criptografia, e foram popularizados pela rede blockchain Ethereum. Eles são projetados para serem transparentes, confiáveis, imutáveis e seguros, o que torna possível automatizar muitos processos de negócios e garantir a integridade dos dados (INVESTOPEDIA..., 2021).

Os contratos inteligentes são utilizados em uma ampla variedade de setores, incluindo finanças, saúde, energia, imobiliário e logística. Na área financeira, por exemplo, os contratos inteligentes são usados para automatizar processos como empréstimos, seguros e investimentos.

Eles também são usados para criar *tokens* digitais, que são ativos digitais emitidos por empresas ou organizações e podem representar tudo, desde moedas virtuais a ações e commodities.

Os contratos inteligentes podem ser programados para executar uma ampla variedade de funções, desde a transferência de ativos digitais até a execução de comandos específicos quando determinadas condições são atendidas. Eles são escritos em linguagens de programação específicas, como por exemplo, a Solidity no caso da rede Ethereum.

A principal vantagem dos contratos inteligentes é a sua capacidade de reduzir a necessidade de intermediários, tais como advogados ou bancos, na execução de contratos. Isso torna os processos mais eficientes, mais rápidos e mais baratos, já que não há necessidade de pagar taxas para intermediários ou aguardar a conclusão de transações que podem levar dias ou semanas para serem concluídas (INVESTOPEDIA..., 2021).

No entanto, os contratos inteligentes também têm algumas desvantagens. Por exemplo, eles são altamente dependentes da precisão dos dados fornecidos para sua execução. Além disso, como são auto executáveis, os erros de programação podem resultar em erros de execução que podem ter consequências graves e até mesmo irreversíveis.

2.2.3 *Tokens*

Os *tokens* consistem em ativos digitais construídos a partir de uma rede blockchain específica, com um conjunto de regras codificadas e implementadas no protocolo da rede a partir dos contratos inteligentes. Cada *token* é essencialmente um ativo digital e, diferentemente das criptomoedas, como o bitcoin e o ether, que são nativos da rede, os *tokens* podem apresentar diversas propriedades e funções que normalmente estão relacionadas a algum projeto específico.

De acordo com a documentação da rede Ethereum (2023), os padrões mais utilizados para a construção de *tokens* são:

- ERC-20: Uma interface padrão para *tokens* fungíveis (intercambiáveis), como *tokens* de votação, *tokens* de *staking* ou moedas virtuais.
- ERC-1363: Define uma interface de *token* para *tokens* ERC-20 que suporta a execução do código do destinatário após a transferência ou o código do remetente após a aprovação
- ERC-721: Uma interface padrão para *tokens* não fungíveis (NFT), como uma escritura de arte ou uma música.

- ERC-2309: Um evento padronizado emitido ao criar/transferir um ou muitos *tokens* não fungíveis usando identificadores de *token* consecutivos.
- ERC-4400: Extensão de interface para função de consumidor EIP-721 ERC-4907 - Adiciona uma função de tempo limitado com permissões restritas aos *tokens* ERC-721.
- ERC-777: Permite que as pessoas criem funcionalidades extras em cima de *tokens*, como um contrato de mixagem para melhorar a privacidade das transações ou uma função de recuperação de emergência para salvá-lo se você perder suas chaves privadas.
- ERC-1155: Permite negociações mais eficientes e agrupamento de transações – economizando custos. Esse padrão de *token* permite a criação de *tokens* utilitários (como $BNB ou $BAT) e *tokens* não fungíveis (NFTs) como CryptoPunks.
- ERC-4626: Um padrão de cofre "tokenizado" projetado para otimizar e unificar os parâmetros técnicos de cofres com rendimento.

2.2.4 Gás

O conceito de gás foi introduzido na rede Ethereum, como uma forma de incentivar o processo de mineração e mitigar os riscos da rede. O gás consiste em uma unidade de medida, utilizada para mensurar a quantidade de trabalho realizada pela rede durante uma operação. O valor de gás para cada tipo de operação é conhecido, possui valor fixo tabelado e depende exclusivamente da quantidade de poder computacional gasto para executar a tarefa. Algoritmos de complexidade maior tendem a possuir um valor mais elevado de gás (ETHEREUM..., 2021).

Na rede Ethereum, o valor do gás é calculado a partir do valor do ether e, para cada transação, o usuário especifica o preço que está disposto a pagar por cada unidade de gás. O valor da taxa final paga pelo usuário é definido multiplicando a quantidade total de gás que será gasto na transação pelo valor da unidade de gás em Ether.

Após as transações serem inseridas na mempool da rede, a seleção de quais transações e em quais ordens serão inseridas no bloco é feita de acordo com o maior valor pago pela unidade de gás. Isso ocorre porque o tamanho de cada bloco é limitado. No caso da Ethereum, cada bloco possui um limite alvo de 15.000.000 de gás, mas esse tamanho pode variar de acordo com a demanda da rede, chegando até o limite de 30.000.000 de gás. Isso garante que o tamanho dos blocos não ultrapasse o limite físico da rede e não comprometa o desempenho diante dos quesitos de espaço e velocidade (CONSENSYS..., 2020).

2.2.5 Binance Smart Chain

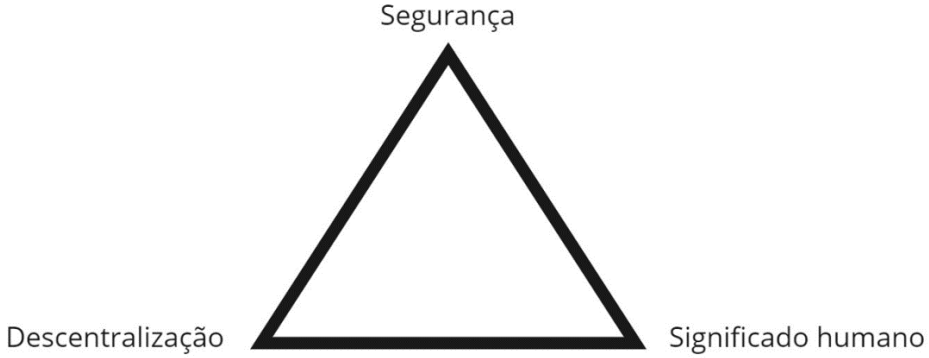
A Binance Smart Chain (BSC) consiste em uma rede blockchain mantida pela maior corretora de ativos digitais do mundo, a Binance. A rede foi construída por meio de um *hard fork* realizado a partir do protocolo da Ethereum, ou seja, toda a sua estrutura foi construída por meio dos algoritmos da Ethereum, apresentando, dessa forma, semelhanças significativas com o seu protocolo de origem (BINANCE..., 2020).

A maior diferença da BSC em relação à Ethereum está no mecanismo de consenso da rede, que foi modificado de forma a permitir que transações sejam realizadas de forma mais rápida e com menor custo. Isso foi possível principalmente devido à centralização da rede e ao menor número de nós, que facilitam o processo de validação dos blocos e contribuem para a escalabilidade da rede (BINANCE..., 2020).

A rede BSC, diferentemente da Ethereum, faz uso do mecanismo de PoSA (*Proof of Stake Authority*), que combina aspectos do *Delegated Proof of Stake* (DPoS) e do *Proof of Authority* (PoA). O protocolo foi criado de forma que 21 validadores se revezem em turnos para produzir blocos. Em troca, eles recebem como recompensa, taxas de transação em BNB, a criptomoeda nativa da rede. Para ser um validador, primeiramente o usuário deve ser um candidato elegível, executando um nó e colocando em *stake* pelo menos 10.000 BNB. Outro usuário, conhecido como delegador, apostará BNB em nome do candidato escolhido. Os 21 melhores competidores escolhidos pelo valor da aposta se revezam na entrega dos blocos. Todo esse processo é repetido a cada 24 horas. Os delegados também recebem uma parte dos prêmios distribuídos aos validadores (BINANCE..., 2020).

De acordo com o Triângulo de Zooko's, demonstrado na Figura 5, que consiste de um trilema de três propriedades desejáveis para uma rede blockchain, é possível estabelecer com maestria duas delas. As propriedades consistem no significado humano, que se refere ao propósito da rede, a segurança e a descentralização. No caso da rede BSC, as propriedades buscadas são a segurança e a escalabilidade como propósito humano, o que a distância da característica de descentralização. Diferentemente, a rede Ethereum possui a segurança e a descentralização como seus pilares, o que a torna distante de características como a escalabilidade, que é um dos principais motivos de crítica à rede, e o grande desafio que os desenvolvedores buscam solucionar (ZOOKO, 2021).

Figura 5 - Triangulo de Zooko.



Fonte: Do Autor (2023).

De acordo com dados coletados em 2021, a rede BSC tem se mostrado a escolha mais popular, diante da alta velocidade e baixo custo das taxas. Dados mostram que os picos de transações ocorridos na BSC chegam a cerca de 12 milhões de transações diárias, valor muito acima do pico de 1,75 milhão da rede Ethereum. O número de endereços ativos também se apresenta superior, com cerca de 2,1 milhões de endereços na rede BSC, contra cerca de 800 mil da rede Ethereum (BINANCE..., 2020).

Outro grande motivo, que favoreceu a expansão da rede BSC, é o fato da compatibilidade entre elas. Por se tratarem de tecnologias semelhantes, as aplicações podem ser facilmente replicadas entre elas. Alguns exemplos podem ser citados, como o protocolo de DeFi da PancakeSwap, principal exchange descentralizada presente na rede BSC, que foi criado a partir de um hard fork da UniSwap, principal exchange descentralizada de rede Ethereum. (PANCAKESWAP..., 2023).

2.3 Aplicações decentralizadas

As aplicações descentralizadas, também conhecidas como DApps, são uma das principais inovações trazidas pela tecnologia blockchain. Ao contrário das aplicações centralizadas tradicionais, os DApps são executados em uma rede descentralizada, sem um ponto central de controle. Isso traz inúmeras vantagens, como maior segurança, transparência e resistência à censura.

Os DApps (Decentralized Applications) são aplicativos executados em sistemas distribuídos, como uma rede blockchain, e podem possuir características como governança democrática, transparência, prova de fraude, acesso global e construção colaborativa. Ao contrário dos aplicativos convencionais, que são hospedados por uma entidade única usando uma arquitetura centralizada de armazenamento, os DApps não são suscetíveis a um único

ponto de falha e não são comprometidos caso um dos nós da rede seja interrompido (TAPSCOTT, D. e TAPSCOTT, A., 2016).

Existe uma ampla variedade de DApps que foram e estão sendo criados com o objetivo de resolver diversos problemas da sociedade, muitos semelhantes a aplicações já existentes, mas sem a dependência de um órgão centralizador. Entre as aplicações mais utilizadas estão jogos, colecionáveis, mídias sociais, organizações autônomas descentralizadas (DAOs) e finanças descentralizadas (DeFi) e seus protocolos (TAPSCOTT, D. e TAPSCOTT, A., 2016).

As aplicações descentralizadas geralmente criam e alimentam seus ecossistemas por meio de uma economia própria, usando um sistema de *tokens* criados a partir de seus contratos inteligentes. Esses *tokens* são utilizados para realizar ações dentro da plataforma, como tomar decisões no caso dos *tokens* de governança ou como forma de pagamento por produtos ou serviços, disponíveis nas aplicações através dos *tokens* de utilidade (TAPSCOTT, D. e TAPSCOTT, A., 2016).

2.3.1 WEB 3.0

A WEB 3.0 é a próxima evolução da internet, que busca torná-la mais descentralizada e segura. A WEB 3.0 utiliza tecnologias como blockchain, criptografia e contratos inteligentes para permitir que as pessoas realizem transações financeiras, armazenem dados e se comuniquem diretamente, sem a necessidade de intermediários. Além disso, a WEB 3.0 também promove a privacidade dos usuários, permitindo que eles controlem seus próprios dados e determinem quem tem acesso a eles. Essa nova web está trazendo um grande potencial para inovações em vários setores, como finanças, saúde, educação e governança. Embora ainda existam desafios a serem enfrentados, como questões regulatórias, de segurança e escalabilidade, a Web 3.0 está abrindo caminho para um futuro mais justo e inclusivo, onde as pessoas têm mais controle sobre seus próprios dados e ativos.

O surgimento da internet ocorreu em 1969 nos Estados Unidos, em um meio acadêmico e militar cujas aplicações eram exclusivamente em projetos de pesquisas avançadas envolvendo a comunicação entre computadores. Logo no início dos anos 70, nasceu o primeiro protocolo de comunicação, o TCP/IP, que constituiu a base da internet como é conhecida hoje. Posteriormente, no início dos anos 90, com o surgimento da linguagem de marcação de HiperTexto (HTML) e da World Wide Web (WWW), por Tim Berners-Lee, iniciaram os acessos a páginas estáticas. As interações nesse tipo de página eram limitadas à exibição de informações. Esse tipo de experiência foi denominado por Darci DiNucci em 1999 como WEB 1.0 (DINUCCI, 1999).

No início dos anos 2000, o termo WEB 2.0 foi popularizado pela editora O'Reilly, que o utilizou para designar a segunda evolução da internet. Nesse período, a internet começou a se modificar, tornando-se mais interativa com o usuário. A estrutura cliente e servidor se tornou mais robusta e veloz, o que aumentou de forma exponencial a evolução das tecnologias. Novas linguagens e paradigmas de programação foram criados, bancos de dados passaram a ser utilizados junto com as aplicações e possibilitou que a experiência web passasse a ser dinâmica. A WEB 2.0 tinha a característica de ser mais participativa, possibilitando a interação do usuário junto às aplicações. É nesse cenário que surgiram grandes empresas de tecnologia, em áreas como redes sociais, marketplaces, serviços de streamers, entre outros (O'REILLY, 2005).

Embora a WEB3.0 ainda esteja em processo de construção e adoção, seus principais conceitos e características já foram definidos. Seguindo de forma analítica o histórico da evolução da internet, é coerente que ocorra uma transição de uma semântica dinâmica para um modelo mais inteligente, visando a automatizar processos e tornar familiar e personalizada a experiência do usuário. Tecnologias como inteligência artificial (IA), visualização e apresentação 3D interativa e as redes *peer-to-peer* descentralizadas, como as blockchains, se apresentam como aspectos essenciais para a WEB 3.0, sendo uma das bases para esse novo ecossistema de desenvolvimento (BERNERS-LEE, 1989).

A partir das redes blockchain, torna-se possível a remoção de partes centralizadoras e intermediadores de processos do sistema, eliminando o controle de terceiros sobre os dados dos usuários e reduzindo o risco de censura. Um dos setores que tende a se beneficiar desse novo sistema são as finanças, por meio das carteiras digitais de criptoativos, é possível ter controle total sobre bens e serviços oferecidos por DApps (Decentralized Application). Além disso, a compatibilidade de DApps on-chain com dados externos tem se tornado cada vez maior, possibilitando a construção de um ecossistema completo de forma descentralizada (INTERNET HALL OF FAME..., 2019).

2.3.2 *Decentralized Autonomous Organizations (*DAOs)

As DAOs *(Decentralized Autonomous Organizations*) são organizações que possuem a capacidade de operar de forma autônoma, descentralizada e ininterrupta. As DAOs são implementadas a partir de contratos inteligentes e, uma vez criadas, não podem ser controladas por uma única entidade. Cada DAO possui seu conjunto de regras específicas, que são desenvolvidas e elaboradas em código. Essas regras, normalmente, seguem uma filosofia de cooperação, tendo as decisões tomadas pela comunidade de participantes que buscam, de forma

conjunta, soluções para os problemas e desafios apresentados pela organização (BINANCE..., c2021a).

Normalmente, as DAOs seguem princípios semelhantes em relação à tomada de decisão. Para fazer parte da comunidade responsável e ter poder de voto, é necessário possuir *tokens* de governança da organização. Esses *tokens* desempenham o papel de utilidade dentro do sistema e o poder de voto é proporcional à quantidade do ativo que o usuário possui. As DAOs permitem que as comunidades sejam organizadas de forma democrática e anônima. O valor agregado ao seu *token* motiva os usuários possuidores a buscarem e tomarem as melhores decisões, o que contribui ainda mais para um crescimento sustentável e seguro do projeto (BINANCE..., c2021a).

2.3.3 *Decentralized Finance* (DeFi)

A *Decentralized Finance* (DeFi) é uma alternativa ao sistema financeiro global, que possibilita o uso de instrumentos financeiros sem a dependência de uma entidade intermediária. Atualmente, a oferta monetária e o conjunto de regras a serem seguidos, pelas instituições financeiras centralizadas de cada estado, são executadas por entidades conhecidas como bancos centrais. O DeFi desafia o sistema tradicional e surge como uma proposta de democratizar e facilitar o uso do mercado financeiro, possibilitando que o uso de serviços financeiros seja feito por qualquer pessoa, em qualquer lugar do mundo (BINANCE..., 2019).

A partir do sistema DeFi, é possível ter controle e visibilidade total sobre os próprios ativos. Por serem construídos sobre uma infraestrutura de rede blockchain, os códigos são abertos e auditáveis, o que proporciona maior transparência e confiança ao sistema. Diferentemente do sistema financeiro tradicional, que depende do lucro para manter a operabilidade dos serviços, por meio da DeFi, os custos são significativamente reduzidos. Qualquer indivíduo pode fazer o uso, independentemente de sua situação financeira, o que beneficia principalmente pessoas de baixa renda que não possuem acesso ao sistema bancário. Por se tratar de uma tecnologia operante a nível global, as operações podem ser realizadas sem qualquer barreira territorial, em minutos, sendo uma grande vantagem em casos de operações internacionais (BINANCE..., 2019).

2.3.3.1 Serviços financeiros descentralizados

A descentralização oferece alternativas para a maioria dos serviços financeiros prestados por instituições centralizadas, incluindo serviços bancários digitais, mercados de

ativos eletrônicos e plataformas de empréstimo *peer-to-peer*. A ampla variedade desses serviços é acessível graças à tecnologia de contratos inteligentes em redes blockchain.

Dentre os principais serviços bancários monetários, disponíveis em aplicações DeFi, está a emissão de *stablecoins*. Com a evolução da indústria blockchain e abertura do mercado tradicional ao mercado de criptoativos, a demanda por *stablecoins* tem se tornado maior. As *stablecoins* são um tipo de ativo, cujo valor está atrelado a uma moeda estável fiduciária, ou seja, um ativo do mundo real. Como os ativos digitais possuem uma alta volatilidade, as *stablecoins* são uma ótima opção para manter o capital em meio às grandes flutuações de valor.

Os seguros e proteções de ativos também têm se tornado promissores no mercado DeFi. Com o surgimento de diversas organizações DeFi, vulnerabilidades presentes nos códigos dos protocolos têm sido usadas por hackers e causado diversos prejuízos ao setor. Por meio dos contratos inteligentes, alguns protocolos visados à segurança e à cobertura de riscos, como o "Armor", têm sido criados, o que permite transferir parte do risco assumido ao interagir com protocolos novos e menos conhecidos, de forma mais barata e rápida, quando comparado com seguradoras do mundo real (BINANCE..., c2021a).

Além dos serviços financeiros citados, uma das aplicações DeFi mais importantes são as exchanges descentralizadas (DEXs). Essas plataformas permitem que usuários negociem ativos digitais sem a necessidade de um intermediário confiável. As transações são realizadas diretamente entre carteiras de usuários, usando contratos inteligentes. Por exigirem menos trabalho de operação e manutenção, as exchanges descentralizadas geralmente têm taxas de negociação mais baixas quando comparadas as exchanges centralizadas (BINANCE..., c2021a).

2.3.3.2 Empréstimos

Atualmente, o processo de emprestar ou tomar um ativo como empréstimo, a partir de instituições financeiras, requer vários processos de verificações pessoais e fiscais, a fim de comprovar que o indivíduo tem a real capacidade de realizar o pagamento do valor acordado. Diferentemente, os empréstimos descentralizados ocorrem sem que alguma das partes envolvidas precise se identificar ou passar por qualquer tipo de burocracia comprobatória. Os provedores descentralizados de empréstimos podem ser de dois tipos, *peer-to-peer*, no qual um mutuário obtém o empréstimo diretamente de um credor específico ou por meio de *pools*, em que credores fornecem fundos a um *pool* de liquidez e posteriormente os mutuários solicitam os empréstimos a partir dele (BINANCE..., c2021a).

2.3.3.2.1 Empréstimos com privacidade

Os empréstimos descentralizados funcionam sem que as partes envolvidas no processo precisem se identificar ou apresentar comprovações financeiras. Em vez disso, todo o processo ocorre por meio dos contratos inteligentes, de forma automatizada, necessitando apenas que o mutuário ofereça algum ativo como garantia, cujo valor supere o valor do empréstimo, podendo inclusive, em alguns protocolos, ser aceitos NFTs como garantia. Nos protocolos mais comuns, o valor disponível para empréstimo é de 50% do valor depositado como garantia. Em casos de o ativo depositado possuir alta volatilidade ou por algum motivo ocorrer a queda do valor, ele poderá ser liquidado, a fim de garantir que o credor não perca sua posição inserida no protocolo de empréstimo (BINANCE..., 2021).

A partir do empréstimo é possível ter acesso a fundos sem que seja necessário vender ativos, o que normalmente são transações tributáveis em alguns estados. Os empréstimos também se tornam uma ótima opção, quando se deseja obter algum ativo com baixa volatilidade e que gere fluxo de caixa, como *stablecoins*, que são o tipo de ativo com valor mais próximo ao de moedas fiduciárias como o dólar (BINANCE..., c2021a).

2.3.3.2.2 Flash loans

O *flash loan* consiste em um tipo de empréstimo em que não é necessário fornecer garantias ou informações pessoais. Diferentemente dos empréstimos com garantia, eles não são amplamente acessíveis e necessitam de conhecimento técnico aprofundado para serem realizados. Como o próprio nome sugere, eles são um tipo de empréstimo rápido, que ocorre em um mesmo bloco de transação. Ou seja, o fundo obtido por meio de um credor precisa ser pago na mesma transação. Se não puder ser pago, toda a transação é revertida como se nada tivesse acontecido, tendo como prejuízo apenas o valor da taxa da transação (BINANCE..., c2022a).

Esse tipo de empréstimo ocorre a partir de pools de liquidez, que são formadas para serem usadas como empréstimos. Esses *pools* contêm ativos que não estão sendo usados, o que abre oportunidade para serem emprestados em troca de uma pequena taxa de juros. Como tudo acontece na mesma transação em um período muito curto de tempo, isso favorece que volumes maiores de ativos sejam emprestados e que vários empréstimos aconteçam ao longo do tempo, beneficiando ambas as partes envolvidas (BINANCE..., c2022a).

O *flash loan* é realizado a partir de um contrato inteligente, que executa todo o processo de forma automatizada. Assim, é necessário que lógicas sejam criadas e personalizadas de acordo com o tipo de operação planejada. O objetivo ao fazer uso de um algoritmo de *flash loan*

é obter lucro de forma rápida, com risco minimizado e com alto grau de alavancagem em situações de baixo capital disponível. Existem diversos tipos de aplicações, com diversos tipos de protocolos DeFi que podem ser usados em conjunto com a operação, sendo de responsabilidade do programador desenvolver a estratégia que melhor lhe atende (BINANCE..., c2022a).

Um dos exemplos mais populares que usam a estratégia *flash loan* é a arbitragem. A partir de uma estratégia de arbitragem entre exchanges, poderiam ocorrer cenários como o seguinte:

1. Tomar emprestado o valor X de ativo a $1,00 em um *pool* A.
2. Vender X do ativo em uma Exchange B por $1,10.
3. Comprar X do ativo em uma Exchange C por $1,00.
4. Pagar o empréstimo para o *pool* A.
5. Obter o lucro das diferenças de valores menos as taxas de transação.

Em um mercado de finanças tradicionais, a arbitragem se torna possível apenas para usuários com uma grande quantia de dinheiro, o que torna inacessível para a maior parte do mercado. Assim, o *flash loans* se tornam um grande exemplo de um futuro em que ter dinheiro não é necessariamente um pré-requisito para obter lucratividade. Porém, existem desafios nesse tipo de operação a alta competitividade em conjunto com as taxas de transação da rede, taxas de juros e *slippage*, torna desafiador encontrar uma estratégia gerenciável e que permita obter lucros.

2.4 Exchanges

As exchanges são plataformas digitais que revolucionaram a negociação de ativos financeiros nos últimos anos. Elas surgiram como uma solução para as limitações das transações *peer-to-peer* e democratizaram o acesso ao mercado financeiro. Essas plataformas oferecem recursos avançados que permite ao investidor tomar decisões, automatizar processos e negociar ativos em alta velocidade. A compreensão das exchanges é fundamental para quem deseja entender como o mercado financeiro funciona e como os investidores podem se beneficiar dele.

A partir de exchanges, é possível realizar transações entre diferentes pares, de forma rápida e simples, com uma ampla variedade de recursos que auxiliam nas tomadas de decisões e na automatização de processos dentro do mercado financeiro. Com o surgimento dos primeiros ativos financeiros, as exchanges surgiram como uma forma de minimizar as barreiras e democratizar a comercialização, que antes eram realizadas de forma *peer-to-peer*, sem uma organização intermediadora, dependendo apenas da confiança de ambas as partes envolvidas.

As exchanges desempenham uma função vital no ecossistema de criptoativos. A partir dos serviços oferecidos por essas plataformas, uma ampla base de usuários consegue iniciar e realizar operações envolvendo os ativos digitais e moedas fiduciárias. Além disso, a liquidez que as exchanges conseguem fornecer, diante da grande base de usuários, favorece com que a variação de preço dos ativos seja menor, o que incentiva ainda mais a entrada de usuários com operações de alto volume no mercado.

Devido à simplicidade e à qualidade dos serviços oferecidos, tradicionalmente, as exchanges centralizadas (CEXs) possuem uma dominância sobre o setor. Porém, com o surgimento dos protocolos DeFi, originados a partir de contratos inteligentes, houve um rápido avanço das ferramentas para as trocas descentralizadas, sendo uma delas as exchanges descentralizadas (DEXs). Elas possuem semelhanças com as CEXs, porém com algumas diferenças, sendo a maior delas o fato de os usuários permanecerem com a custódia de seus ativos.

2.4.1 Exchanges centralizadas

Em uma CEX, de forma semelhante ao mercado financeiro tradicional, como as bolsas de valores, a partir de um depósito em moeda fiduciária na plataforma, o usuário consegue ter acesso aos recursos e realizar operações conforme desejado. Além do depósito em moedas fiduciárias, é possível realizar depósito de ativos digitais.

Os ativos disponibilizados pelas exchanges passam por um processo de listagem, sendo assim, deve-se verificar anteriormente se a CEX possui suporte para o ativo desejado. Ao realizar o depósito de um fundo, diferentemente das exchanges centralizadas, o usuário não possui acesso à chave privada, o acesso se torna exclusivo da CEX. Do ponto de vista de usabilidade, o usuário permanece com total controle, porém com a suscetibilidade da ocorrência de ações externas sobre seus fundos (BINANCE..., c2021b).

Uma das grandes vantagens das CEXs é o fluxo de trabalho simplificado. Diferente das operações on-chain, as operações não ocorrem diretamente nas redes blockchain, as transações são realizadas internamente, no sistema da exchange, em um banco de dados próprio, semelhante ao que ocorre em corretoras do mercado tradicional. Esse tipo de sistema favorece para que as transações aconteçam de forma mais veloz e simples, além de estar livre das taxas de mineração, sendo uma grande vantagem em momentos de pico e congestão nas redes (BINANCE..., c2021b).

As ferramentas de análise gráfica em tempo real também se tornam grandes aliadas dos usuários das plataformas, sendo de grande uso por analistas técnicos no processo de tomada de

decisões. Por meio das CEXs é possível ter acesso a ferramentas como o livro de ofertas, o qual consiste de uma lista das ordens de compra e venda de um determinado ativo. Esses tipos de dados são de grande utilidade para traders, pois ajudam a avaliar o interesse do comprador e do vendedor em níveis de preços específicos. Esses dados podem fornecer informações valiosas sobre os níveis potenciais de suporte e resistência, além de auxiliar na previsibilidade da cotação dos valores, sendo de grande uso em algoritmos de trading (BINANCE..., c2021b).

2.4.2 Exchanges descentralizada

Assim como as CEXs, as DEXs têm como objetivo possibilitar trocas entre diferentes ativos, porém apresentam diferenças significativas. As exchanges descentralizadas são mercados construídos a partir de uma rede blockchain com suporte aos contratos inteligentes. A partir das DEXs é possível realizar a negociação de ativos digitais sem a necessidade de uma organização centralizada intermediadora, para a proteção dos fundos (BINANCE..., 2020).

Por meio dos contratos inteligentes, todas as ações e os procedimentos de segurança são automatizados. Além disso, por se tratar de um sistema construído sobre uma infraestrutura global, há menores restrições geográficas, o que possibilita a negociação de ativos em quaisquer localidades do mundo, ao contrário das CEXs, cujas permissões de atuação estão sob as legislações locais (BINANCE..., c2020b).

Existem diferentes tipos de DEXs que atuam em diferentes tipos de redes blockchain, porém todas têm como semelhança o fato de suas ordens serem executadas *on-chain*. Sendo assim, os usuários nunca deixam de possuir a custódia sobre seus fundos e todas as ações ocorrem de forma pública e anônima. Atualmente, apesar dos avanços em tecnologias *cross-chain*, entre blockchains, as maiores DEXs estão disponíveis nas redes Ethereum e BSC, sendo as principais a UniSwap e a PancakeSwap (BINANCE..., c2020b).

2.4.2.1 *Book* de ofertas *on-chain*

Nas principais exchanges descentralizadas, todas as operações são realizadas *on-chain*. Sendo assim, todas as ordens são registradas na blockchain, o que torna uma abordagem transparente e imune a censuras. Porém, torna o processo menos prático, pois para a efetivação de uma transação ela deve ser gravada em um bloco, sendo necessário que seja requerido a um nó para que a transação seja acionada. Esse tipo de procedimento apresenta uma maior complexidade e necessita de maior conhecimento, quando comparado às CEXs (BINANCE..., c2020b).

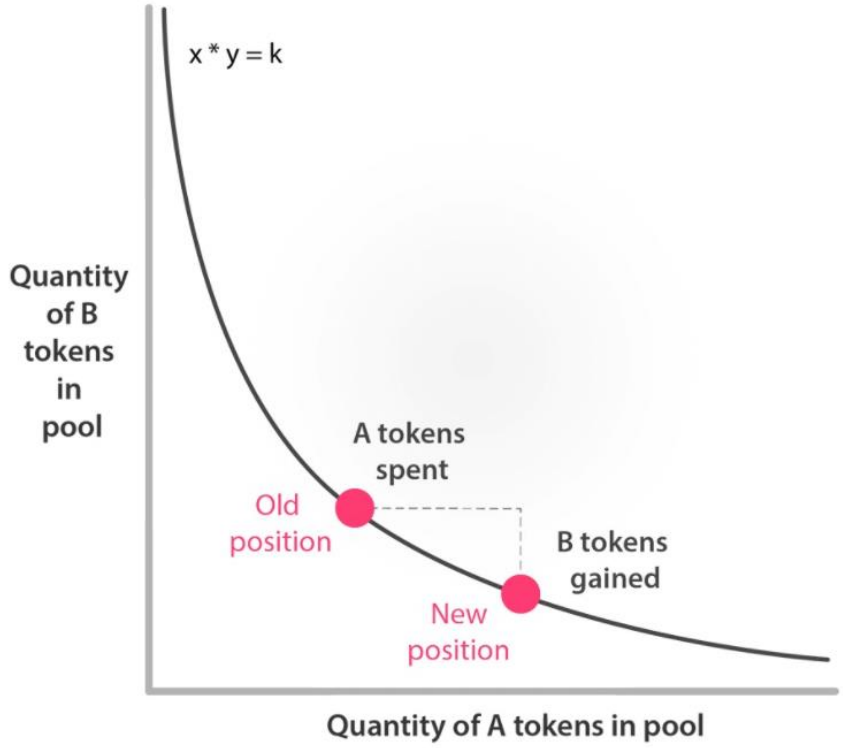
Ao solicitar uma transação, o tempo e a ordem da execução dependem de alguns fatores, como o tempo de validação do bloco e o valor do gás ofertado pelo serviço. Além disso, esse tipo de transação é mais lento quando comparado às CEXs, deixando a transação suscetível a altas variações nas taxas de *slippage*. Outro fator a ser considerado é o *front running*, devido ao fato das transações permanecerem na mempool, até que sejam efetuadas, alguns usuários podem analisar as transações pendentes e fazer o pagamento do gás com um valor maior, a fim de realizar uma negociação antes que a pendente seja efetuada, sendo um mecanismo muito utilizado em ferramentas de HFT (BLOCKCHAIN.INFO, 2023).

2.4.2.2 *Automated Market Makers* (AMM)

O AMM (*Automated Market Makers*) pode ser definido como um tipo de protocolo de DEX, um algoritmo de automatização da cotação de preço entre dois ativos. Algumas organizações fazem uso desse tipo de tecnologia com base em algumas fórmulas matemáticas, podendo ser de níveis de complexidade diferentes, dependendo do tipo de problema a ser solucionado. O AMM, além de possibilitar a realização de trades sem a necessidade de algum tipo de cadastro pessoal, permite que usuários possam participar do protocolo de forma colaborativa, por meio do *pool* de liquidez. Com isso, tornando-se *Market Makers* e sendo recompensados com partes das taxas cobradas ao serem realizadas algum tipo de negociação pela plataforma (BINANCE, c2020b).

Diferentemente de uma exchange tradicional que faz uso do livro de ordens com transações *peer-to-peer*, com um AMM, os ativos são precificados a partir de algoritmos. Um dos maiores exemplos é a DEX UniSwap, na qual o valor de um ativo é calculado conforme exemplificado na Figura 6, a partir da expressão X * Y = K, sendo X e Y os ativos a serem negociados e K uma constante. Isso significa que a liquidez total de um par de ativos permanecerá sempre a mesma, modificando apenas a relação de proporcionalidade entre os ativos, conforme ocorrem as negociações. A liquidez é fornecida por usuários, os *Market Makers*, e são denominados LPs (*Liquidity Providers*) (BINANCE, c2020b).

Figura 6 - Relação da quantidade de tokens.



Fonte: (BINANCE, c2020b).

2.4.2.3 *Pool* de liquidez

Os pools de liquidez são, basicamente, um tipo de contrato inteligente que possibilita que um conjunto de fundos seja bloqueado e utilizado em serviços, como negociação e empréstimos, em protocolos DeFi. Os *pools* de liquidez são os componentes principais das exchanges descentralizadas, pois fornecem toda a liquidez necessária para que as trocas sejam efetuadas, sem a necessidade de um segundo usuário como contraparte da negociação. Os provedores de liquidez são os responsáveis por adicionar fundos aos *pools*, e, em troca, recebem uma parte proporcional da participação do *pool* sobre as taxas pagas pelos usuários que realizaram negociações na plataforma. Nas principais exchanges descentralizadas, os ativos depositados seguem uma proporção de valor de 50% para cada ativo (BINANCE, c2022b).

Criar formas de atrair fundos é de grande importância. Devido à forma como o AMM funciona, conforme a liquidez aumenta, os pedidos de grande valor são menos propensos a *slippage*, o que atrai um maior volume para a plataforma, incentivando seu crescimento. Os problemas de *slippage* variam entre diferentes projetos de AMM, mas é algo a ser considerado em todos eles. Basicamente, o preço é determinado pela mudança na relação entre ativos no

pool de liquidez após a transação. Se o valor relativo entre os ativos variar muito, haverá um maior *slippage* (BINANCE, c2022b).

2.4.2.4 LP tokens

Após realizar o depósito dos pares de ativos, nas proporções definidas, o usuário receberá *tokens* LP (*Liquidity Provider*). Esse tipo de *token* funciona como um tipo de recibo e representa a parte do usuário no pool de liquidez. A partir dos *tokens* LP, o usuário poderá remover os seus ativos do *pool* e coletar sua participação nos juros obtidos durante o período. Os *tokens* LP são depositados na carteira do usuário. Em alguns tipos de protocolos, os *tokens* LP e suas propriedades no *pool* de liquidez podem ser transferidos para outras carteiras sem quaisquer problemas (BINANCE..., c2022b).

A UniSwap, talvez seja a maior referência de uso dos *tokens* LP. Em suas versões V1 e V2, os *tokens* eram recebidos no padrão ER-20 e davam direito a 0,3% das taxas de negociação na exchange. Porém, com as novas atualizações do protocolo, para a versão V3, a dinâmica do fornecimento de liquidez seguirá uma curva de preço diferente, e o *token* que antes era fungível passará a ser não fungível seguindo o padrão ERC-721 (BINANCE..., c2022b).

2.4.2.5 *Impermanete loss*

De forma simplificada, *impermanente loss* (perda impermanente) pode ser entendida como a diferença de valor entre os ativos, caso permanecidos em uma carteira em *hold* ou aplicados em um *pool* de liquidez. Quanto maior a volatilidade e variação dos ativos, maior a probabilidade de estar exposto a perdas impermanentes. Uma perda neste caso significa que o valor total dos ativos, quando comparado ao dólar, valerá menos do que no momento do depósito no pool (BINANCE..., c2020c).

Os riscos das perdas impermanentes podem ser mitigados com as taxas de transações. Mesmo em situações de alto risco, em um mercado de alto fluxo de negociações, a relação de risco e retorno pode ser benéfica para o fornecedor de liquidez. No entanto, isso depende de diversos fatores relacionados ao protocolo, como *pools* específicas, ativos depositados e condições de mercado, e devem ser analisados cuidadosamente antes da aplicação de capital nesse tipo de mercado (BINANCE..., c2020c).

2.4.2.6 *Initial Dex Offering* (IDO)

Os IDOs consistem em um método de captação de recursos a partir de uma DEX, é uma abordagem de captação de recursos que reúne capital de investidores de varejo. As IDOs foram

originadas buscando compensar as deficiências do modelo "tradicional" de *crowdfunding* de criptomoedas, como os ICO, com o uso das redes blockchain, permitindo que qualquer usuário faça parte do projeto (Binance..., c2020d).

Uma IDO faz uso de uma exchange descentralizada (DEX), para realizar a venda de *tokens*. Um projeto fornece seus *tokens* para a DEX e os usuários oferecem seus fundos por meio da plataforma. Esses processos são automatizados e executados por contratos inteligentes na blockchain. Os fundos arrecadados durante a IDO acabam em um pool de liquidez. Quando o projeto estiver ativo, o *pool* de liquidez será aberto para negociação. Os investidores que compraram o *token* antecipadamente podem começar a negociar imediatamente (Binance..., c2020d).

2.4.2.7 UniSwap

UniSwap é uma das principais plataformas de negociação descentralizada (DEX) baseada em Ethereum. Ela permite que os usuários troquem *tokens* ERC-20 sem a necessidade de intermediários tradicionais, como corretoras centralizadas. A UniSwap utiliza um modelo de negociação automático baseado em algoritmos, em vez de ordens de compra e venda tradicionais (BINANCE..., 2020).

Desde o seu lançamento em 2018, a UniSwap evoluiu significativamente, com o lançamento da UniSwap v2 em 2020 e a UniSwap v3 em maio de 2021. A versão 2 incluiu várias melhorias, como uma melhor integração de liquidez, melhor eficiência de preço e uma interface de usuário atualizada. Já a versão 3 introduziu o conceito de "concentração de liquidez", permitindo que os provedores de liquidez se concentrem em intervalos de preço específicos para maximizar seus retornos (BINANCE..., 2020).

A UniSwap se tornou um importante componente do ecossistema DeFi (finanças descentralizadas), com um volume diário de negociação que ultrapassa US $1 bilhão em muitos dias. Além disso, a plataforma tem atraído muita atenção do mercado devido ao seu modelo de governança descentralizado, que permite que os detentores de *token* UniSwap (UNI) votem em decisões importantes relacionadas à plataforma. Com o crescimento constante do DeFi e a crescente popularidade das DEXs, a UniSwap está bem posicionada para desempenhar um papel cada vez mais importante no futuro dos mercados financeiros descentralizados.

2.4.2.8 SushiSwap

A SushiSwap é uma plataforma descentralizada de troca de criptomoedas que foi lançada em setembro de 2020, como um *fork* da UniSwap. O projeto foi criado por uma equipe

anônima de desenvolvedores que buscavam criar uma plataforma de troca mais justa e descentralizada do que a UniSwap. A principal diferença entre SushiSwap e UniSwap é que a primeira oferece *tokens* da plataforma, denominado SUSHI, como recompensa, incentivando os usuários a fornecerem liquidez para o protocolo. Além disso, a SushiSwap apresenta taxas de troca menores, em relação à UniSwap, o que torna a plataforma mais atraente para usuários que buscam economizar.

2.5 High Frequency Trading (HFT)

O *High Frequency Trading* (HFT) é um método avançado de negociação que utiliza algoritmos complexos, para analisar grandes quantidades de dados e executar negociações em alta velocidade. Ele surgiu inicialmente nos mercados financeiros tradicionais, mas com o aprimoramento da infraestrutura em exchanges centralizadas, como a Binance, acabou entrando no espaço dos ativos digitais. Os HFTs também são possíveis de serem utilizados em exchanges descentralizadas e existem diversas empresas renomadas, incluindo Jump Trading, DRW, DV Trading e Hemeier que utilizam essa técnica, de acordo com o Financial Times (ALDRIDGE, 2013).

Por meio dos HFTs, é possível analisar diversos mercados e executar um grande número de transações em questão de milissegundos, o que muitas vezes é um fator crucial para o sucesso nas negociações. Esse tipo de negociação elimina os spreads estreitos de compra e venda, permitindo a execução rápida de volumes elevados. Além disso, os HFTs possibilitam com que os participantes do mercado aproveitem os movimentos de preços antes que eles sejam totalmente refletidos no livro de ofertas, tornando-o rentável mesmo em mercados voláteis ou com baixa liquidez (ALDRIDGE, 2013).

2.5.1 Principio de funcionamento em DEX

As exchanges descentralizadas (DEX) operam de maneira diferente das exchanges centralizadas (CEX). Em uma CEX, os usuários depositam seus ativos em uma carteira da exchange e, em seguida, fazem negociações por meio da própria plataforma. Diferentemente, em uma DEX, os usuários mantêm a custódia de seus ativos em suas próprias carteiras, e as negociações ocorrem por meio de contratos inteligentes executados em uma blockchain. Esse modelo de funcionamento traz benefícios em termos de segurança, já que os usuários têm controle total sobre seus ativos. Além disso, as DEX não estão sujeitas às mesmas regulamentações que as CEX, o que as torna mais atraentes para aqueles que buscam privacidade e liberdade financeira.

No entanto, as operações em DEX apresentam alguns desafios em relação à velocidade de execução das negociações. Como as negociações ocorrem por meio de contratos inteligentes, a execução pode levar mais tempo do que em uma CEX, devido ao tempo de validação dos blocos. Isso pode ser um obstáculo para os *traders* de alta frequência, que buscam aproveitar pequenas flutuações de preços em um curto período de tempo. Contudo, torna-se uma área promissora para o desenvolvimento de novas estratégias, uma vez que ainda é pouco explorada em comparação ao mercado de finanças tradicionais (ESMA..., 2015).

Para contornar esses problemas, os HFTs são desenvolvidos com algoritmos sofisticados que operam em velocidades extremamente altas para analisar continuamente os ativos digitais em diversas DEX. Esses algoritmos são projetados para encontrar gatilhos e tendências de negociações que não são facilmente detectados a olho nu, permitindo que o HFT seja o primeiro a aproveitar novas tendências de mercado. Além disso, os HFTs também utilizam ferramentas de monitoramento de mempool, que são áreas de espera nas quais as transações aguardam para serem validadas pelos mineradores. Ao monitorar a mempool, os *traders* podem identificar quais transações irão ser processadas e, assim, ajustar suas estratégias para aproveitar as flutuações de preços com maior precisão (ESMA..., 2015).

Embora as operações em DEX possam apresentar desafios em termos de velocidade de execução e tomada de decisão, as vantagens em termos de segurança e liberdade financeira estão atraindo cada vez mais usuários e traders para essas plataformas descentralizadas. A medida que a tecnologia e os algoritmos de negociação de alta frequência continuam a evoluir, espera-se que as negociações em DEX se tornem ainda mais eficientes no futuro, o que beneficia o mercado, aumentando a liquidez entre os pares transacionados e permitindo a negociação de grandes volumes de ativos.

2.5.2 Principais estratégias de HFT.

Apesar de existirem diversos tipos de estratégias de HFT, que poderiam ser listadas, a maioria delas já existe há algum tempo e é aplicada apenas em mercados tradicionais. Diante do mercado de ativos digitais, é importante compreender as principais estratégias que podem ser aplicadas, seus princípios de funcionamento e sua importância.

2.5.2.1 *Colocation*

Colocation refere-se à prática de colocar os servidores de negociação o mais próximo possível do *data center* da exchange, em alguns casos, dentro da mesma instalação, para garantir os menores atrasos possíveis na transmissão de dados do mercado (ALDRIDGE, 2013).

Em operações normais, todos os usuários enfrentam riscos associados a atrasos de comunicação. Para a maioria dos pequenos investidores, esses pequenos atrasos no processamento não terão um impacto significativo. Mas para operadores de alta frequência, milissegundos podem definir a obtenção de lucro ou não. Uma maneira óbvia de aproveitar isso é com melhores equipamentos, outra maneira é ter seus próprios servidores de negociação fisicamente localizados perto do data center que alimenta o *hub* da exchange escolhida (ALDRIDGE, 2013).

Algumas exchanges oferecem servidores privados em seus sistemas para os clientes interessados. No entanto, também há casos em que usuários instalam seus algoritmos diretamente nos servidores principais, permitindo acesso direto aos *data centers* da exchange. Os clientes que optarem por essas opções evitarão a conexão pela internet, o que reduzirá consideravelmente os atrasos de transmissão. Essa prática é conhecida como "*colocation*" e reduz de forma significativa a latência na transferência de dados, sendo uma grande vantagem para todos os que a utilizam (ALDRIDGE, 2013).

Existem diversas exchanges de criptomoedas que oferecem esse tipo de conexão. Exchanges como HitBTC, Gemini e ErisX oferecem opções de *colocation* para seus clientes há mais de um ano. Isso permite que as organizações que não estão fisicamente próximas o suficiente dos data centers e que precisem acessar, aproveitem e façam uso da estratégia HFT. A *colocation* geralmente é um fator crítico na realização desse tipo de negociação, pois sempre dará uma vantagem sobre aqueles que estão mais distantes do ponto de acesso (ALDRIDGE, 2013).

2.5.2.2 *Market Making*

A estratégia de *Market Making* (criação de mercado) envolve colocar simultaneamente ordens de compra e venda de um ativo e lucrar com a diferença entre eles, conhecido como *spread*. Grandes empresas, chamadas de formadoras de mercado, fornecem liquidez e estabilidade aos mercados e são conhecidas principalmente no mercado tradicional de ações. A estratégia também pode ser aplicada às exchanges de criptomoedas, o que garante maior qualidade e segurança ao mercado. No entanto, existem também formadoras de mercado que não possuem acordos com plataformas de negociação e utilizam seus algoritmos com o objetivo de lucrar com os *spreads* (ALDRIDGE, 2013).

As formadoras de mercado estão constantemente comprando e vendendo ativos e definindo seus spreads de compra e venda para obter um pequeno lucro em cada negociação. Por exemplo, elas podem comprar Bitcoin por US$ 37.100 de alguém que deseja vender e

oferecer para vendê-lo por US$ 37.102. A diferença de US$ 2,00 entre os preços de compra e venda é chamada de spread e é a forma principal das formadoras de mercado obterem lucro. Embora a diferença entre os preços de compra e venda possa parecer pequena, a negociação diária em grandes volumes pode resultar em uma fatia significativa de lucro.

2.5.2.3 *Ping*

O *ping* é um processo no qual os usuários de HFT utilizam uma série de pedidos menores, feitos muito rapidamente para procurar pedidos maiores que foram "segmentados", ou divididos em partes pequenas, para não afetar significativamente o preço de mercado. Esse método testa essencialmente uma faixa de preços, usando pequenas encomendas, para descobrir as faixas alta e baixa pelas quais um grande trader está tentando vender. Como os meios de detectar essas ordens são codificados em algoritmos, eles podem procurar essas oportunidades várias vezes por segundo. O software pode então utilizar sua velocidade aumentada para comprar o ativo imediatamente na extremidade inferior desse intervalo e vendê-lo rapidamente na extremidade superior, gerando um lucro instantâneo (ALDRIDGE, 2013).

É certo que essa técnica se alimenta principalmente de grandes *traders* e é frequentemente utilizada em locais conhecidos como *dark pools* “piscinas escuras". Piscinas escuras são exchanges privadas ou fóruns que não informam sua carteira de pedidos em tempo real. As regulamentações geralmente exigem que as informações da transação sejam divulgadas, mas podem ser adiadas por tempo suficiente para que grandes usuários institucionais sejam capazes de realizar grandes operações sem afetar imediatamente o mercado. No entanto, esses sistemas estão propensos a serem vitimados por comerciantes de alta frequência que podem utilizar a técnica de *ping* para identificar e negociar contra outros usuários da *dark pool*. Isso pode reduzir os lucros dos comerciantes que não são de alta frequência e prejudicar os benefícios do uso de uma *dark pool* (ALDRIDGE, 2013).

2.5.2.4 Trading baseado em notícias

A negociação baseada em notícias é uma estratégia que consiste em comprar ou vender ativos de acordo com as informações divulgadas pela imprensa. É comum que os traders utilizem as notícias como parte de suas estratégias de negociação, pois as informações são geralmente acessíveis para o público em geral. No entanto, é importante lembrar que o uso de informações não divulgadas publicamente é considerado "abuso de informações privilegiadas" e é ilegal em muitos mercados (ALDRIDGE, 2013).

A forma como a *High Frequency Trading* (HFT) usa notícias é por meio do uso de software avançado para analisar rapidamente as notícias assim que elas são lançadas, e então tomar decisões de compra ou venda imediatamente. O software é capaz de identificar não apenas qual ativo está sendo afetado pela notícia, mas também se essa notícia é positiva ou negativa. Embora o uso dessas notícias não seja considerado "abuso de informações privilegiadas", ainda pode oferecer uma vantagem significativa para aqueles que conseguem processar e reagir às notícias mais rapidamente do que outros (ALDRIDGE, 2013).

2.5.2.5 Arbitragem

Arbitragem é o método utilizado para aproveitar as diferenças de preço, para um mesmo ativo em diferentes exchanges. Por exemplo, se um Bitcoin custa $ 30.050 na exchange A e $ 30.100 na exchange B, pode-se comprá-lo na primeira exchange e vendê-lo imediatamente na segunda exchange para obter um lucro rápido (DOE, 2022).

Os comerciantes que lucram com essas inconsistências do mercado são chamados de arbitradores. Usando algoritmos HFT eficientes, eles podem aproveitar as discrepâncias antes de qualquer outra pessoa, e ao fazê-lo, contribui para estabilizar os mercados, ao equilibrar os preços (DOE, 2022).

O HFT é altamente benéfico para os arbitradores, pois a janela de oportunidade para conduzir estratégias de arbitragem é geralmente muito pequena (menos de um segundo). Para aproveitar rapidamente as oportunidades de mercado de curto prazo, os HFTs contam com sistemas de computadores robustos que podem varrer os mercados rapidamente. Além disso, as plataformas HFT não apenas descobrem oportunidades de arbitragem, mas também podem fazer negociações até centenas de vezes mais rápidas do que um operador humano (DOE, 2022). Existem diferentes tipos de arbitragem, sendo elas:

- Arbitragem entre exchanges: A compra é realizada em uma exchange e a venda é feita em uma segunda exchange da mesma rede.
- Arbitragem espacial: Semelhante à arbitragem entre exchanges, porém as exchanges estão localizadas em redes ou regiões diferentes.
- Arbitragem triangular: Loops de negociação e movimentação de fundos entre dois ou três ativos digitais para capitalizar as discrepâncias de preços de um ou mais ativos, as negociações são feitas em uma mesma exchange entre *pools* de liquidez diferentes.

2.5.2.6 Oportunidade de curto prazo

As negociações de alta frequência não se destinam a *swing traders* e *buy-and-holders*. Em vez disso, são empregadas por especuladores que querem apostar nas flutuações de preços de curto prazo. Como tal, os *traders* de alta frequência se movem tão rapidamente que o preço pode não ter tempo de se ajustar antes de agirem novamente (DOE, 2022).

Por exemplo, quando uma "baleia" (nome dado aos proprietários de grandes quantidades de criptomoedas) despeja criptomoeda, o seu preço normalmente cai por um curto período de tempo antes que o mercado se ajuste para atender ao equilíbrio entre oferta e demanda. A maioria dos traders manuais perderá essa queda porque pode durar apenas alguns minutos (ou até segundos), mas os traders de alta frequência podem capitalizar isso. Eles têm tempo para deixar seus algoritmos funcionarem, sabendo que o mercado acabará se estabilizando (DOE, 2022).

2.5.2.7 Negociação de volume

A negociação de volume em HFT (*High Frequency Trading*) é uma estratégia que envolve a realização de transações de acordo com o volume de ações negociadas em um determinado período. Os traders que utilizam essa estratégia rastreiam o número de ações que são negociadas em um determinado período de tempo, e, em seguida, fazem negociações com base nas informações obtidas (DOE, 2022).

A lógica por trás dessa estratégia é que, à medida que o volume de ações negociadas aumenta, aumenta também a liquidez do mercado, o que pode facilitar a compra ou venda de um grande número de ações sem afetar significativamente o preço de mercado. A negociação de volume é uma das muitas estratégias utilizadas por traders de HFT para explorar oportunidades de lucro em mercados financeiros altamente líquidos e voláteis (DOE, 2022).

2.6.3 Fatores

A velocidade de negociação sempre foi uma marca registrada das técnicas de HFT. Isso tem sido uma bênção e uma maldição, pois o mercado HFT evoluiu em torno da tentativa de obter vantagens minúsculas em velocidade, em vez de inovações tecnológicas fundamentais. No caso do DeFi, a velocidade de negociação permanece altamente relevante, mas está longe de ser a única dominante das estratégias HFT bem-sucedidas. A natureza programável e *on-chain* do DeFi introduziu novas dimensões que determinam o sucesso ou o fracasso das estratégias de HFT (DOE, 2022).

Embora existam muitas diferenças entre HFT em DeFi quando comparadas às estruturas tradicionais de mercado de capitais, é possível citar alguns fatores que adicionam novas dimensões ao design dos softwares que executam estratégias de HFT em DeFi.

2.6.3.1 Tempo de validação dos blocos

O tempo de validação dos blocos pode ter um impacto significativo nos algoritmos de HFT, pois a velocidade de processamento das transações é fundamental para a execução rápida das ordens de compra e venda. Se o tempo de validação dos blocos for muito longo, pode haver atrasos na execução das ordens, o que pode levar a perdas financeiras. Por outro lado, se o tempo de validação dos blocos for muito curto, pode haver um aumento na volatilidade do preço dos ativos, o que também pode levar a possíveis prejuízos. Portanto, é importante levar em consideração o tempo de validação dos blocos ao desenvolver estratégias de negociação em alta frequência.

Além disso, o tempo de validação dos blocos pode afetar a liquidez do mercado, o que pode tornar mais difícil encontrar contrapartes para negociações. Quando a liquidez é baixa, os spreads entre os preços de compra e venda podem ser maiores, o que pode diminuir a rentabilidade das estratégias de negociação com HFT. Por outro lado, quando a liquidez é alta, os spreads tendem a ser menores, o que pode tornar as estratégias de negociação com HFT mais lucrativas. Por isso, o tempo de validação dos blocos e a liquidez do mercado são fatores importantes que devem ser considerados por quem executa algoritmos de HFT (DOE, 2022).

2.6.3.2 Transparência comercial

A transparência é um fator chave que diferencia as estratégias de HFT em DeFi dos mercados tradicionais de capitais. Isso porque, devido à natureza transparente da infraestrutura DeFi, os comerciantes devem ser capazes de construir estratégias competitivas para lidar com outras estratégias de HFT, ou simplesmente competir com estratégias alternativas. Isso é possível devido ao fato de todos poderem ver as negociações que já foram realizadas e as que estão em fila para serem executadas (DOE, 2022).

2.6.3.3 Custo

Os leilões prioritários de gás (PGAs), em que os participantes fazem lances de transação, têm sido um dos fatores associados ao aumento dos custos de gás na blockchain Ethereum. Isso tem levado ao crescimento na adoção de outras blockchains, como Binance Smart Chain, Solana e Polygon. Do ponto de vista das negociações de alta frequência, as estratégias não só

precisam considerar a lógica algorítmica por trás das negociações específicas, mas também os custos associados a elas. Em muitos cenários, as estratégias de HFT podem não ser economicamente viáveis devido aos custos associados à execução das transações (DOE, 2022).

2.6.3.4 *Miner Extrable Value* (MEV)

O valor extraível do minerador (MEV) tornou-se um dos conceitos mais debatidos em DeFi nos últimos anos. Inicialmente cunhado por Phil Daian et al. no artigo “Flash Boys 2.0”, o MEV descreve o lucro que um minerador pode obter com base em sua capacidade de realizar transações em um bloco em uma ordem específica. O MEV é um conceito importante na economia de criptoativos e tem profundas implicações nas estratégias HFT-DeFi. Ele impõe um vetor de restrição nas estratégias, contando com o interesse econômico do minerador para determinar a colocação final de uma transação em um bloco. Negociações de HFT simples e perfeitamente viáveis em um protocolo DeFi podem fracassar, devido ao fato de um minerador colocar a transação em uma ordem que favoreça outro arbitrador (DAIAN, 2019).

Além disso, o MEV é completamente obscuro e faz com que todos os negócios dependam de uma parte cujo interesse econômico pode não estar alinhado com uma determinada estratégia de HFT. Nos últimos meses, protocolos como Flashbots, ArcherDAO e outros vêm tentando criar dinâmicas mais transparentes e quantificáveis para mitigar o impacto do MEV.

2.6.3.5 Protocolo subjacente

Nos mercados de capitais tradicionais, os algoritmos de HFT interagem com uma infraestrutura relativamente consistente em diferentes classes de ativos que foram estabelecidas há anos. No espaço DeFi, eles precisam interagir com uma infraestrutura que está em constante mudança, com novos protocolos e tempos de execução variável. Jogar com uma infraestrutura instável e em constante mudança apresenta desafios para as estratégias de HFT em DeFi, mas também cria ondas de novas oportunidades devido às ineficiências dos novos protocolos que entram no mercado (DOE, 2022).

3 MATERIAIS E MÉTODOS

O desenvolvimento de sistemas automatizados, para a negociação de criptomoedas tornou-se cada vez mais relevante e necessário em um mercado em que as tecnologias estão profundamente enraizadas. Neste capítulo, é apresentado o desenvolvimento de um algoritmo de arbitragem entre duas exchanges descentralizadas, utilizando as principais tecnologias de desenvolvimento de software para a WEB 3.0. O sistema foi projetado para permitir que os usuários negociem com eficiência em exchanges da rede Ethereum, explorando as diferenças de preços entre elas. Será explicado em detalhes o funcionamento do algoritmo e suas principais características.

O método aplicado é qualitativo, no qual, de acordo com Creswell (2014), não busca representar números, mas sim explicar o fenômeno ou o contexto em que a pesquisa foi realizada, aprofundando o entendimento sobre um determinado assunto. De acordo com a proposta do trabalho, busca-se explicar o motivo do uso de determinadas tecnologias, apresentando os fatos sobre os aspectos da automatização do processo de análise e tomada de decisão. Além disso, busca-se demonstrar os procedimentos para a implementação e execução do algoritmo, buscando, assim, o objetivo principal do trabalho, que é aplicar uma das estratégias de HFT, a saber, a arbitragem entre exchanges descentralizadas.

3.1 Tecnologias

Nesta seção, são apresentadas as ferramentas e tecnologias que foram empregadas no desenvolvimento da aplicação em questão. Esse conjunto de materiais foi escolhido com base nas necessidades específicas do projeto e em sua viabilidade técnica, visando alcançar os objetivos estabelecidos de forma eficiente e eficaz. As tecnologias selecionadas incluem linguagens de programação, frameworks, bibliotecas, softwares de desenvolvimento e outras ferramentas relevantes para a construção da aplicação. O conhecimento desses materiais é essencial para uma compreensão completa do processo de desenvolvimento da aplicação e para sua eventual manutenção, evolução e replicação por parte dos interessados.

3.1.1 JavaScript

No desenvolvimento do algoritmo em questão, optou-se por utilizar a linguagem de programação JavaScript. Essa escolha foi motivada pelo fato de ser adepta ao paradigma da orientação a objetos, desempenho elevado e, sobretudo, pela disponibilidade da biblioteca web3.js, fundamental para as funcionalidades requeridas.

3.1.2 Node.Js

"Node.js" é uma plataforma de desenvolvimento baseada em JavaScript que permite a execução de código JavaScript no lado do servidor. Com uma arquitetura de entrada/saída não bloqueante, ele é capaz de lidar com várias solicitações ao mesmo tempo, graças ao uso do poderoso engine V8 do Chrome. Além disso, Node.js oferece uma ampla gama de módulos e bibliotecas prontas para uso, tornando-se uma escolha popular para aplicações web em tempo real, serviços de back-end e APIs RESTful. O "Node.js" é mantido pela comunidade open-source e conta com uma rica variedade de bibliotecas e frameworks disponíveis para ajudar os desenvolvedores a construir soluções robustas e eficientes.

3.1.3 Bibliotecas

No processo de desenvolvimento do algoritmo, além da linguagem de programação JavaScript junto ao Node.js, foram utilizadas algumas bibliotecas que desempenharam um papel fundamental na disponibilização de recursos e funções que auxiliaram na construção do projeto. Essas bibliotecas contribuíram significativamente para tornar o algoritmo mais enxuto e reduzir a sua complexidade, resultando em um código mais organizado e fácil de manter. Entre as bibliotecas empregadas, destacam-se aquelas voltadas para a interação com a rede Ethereum, como a web3.js, que possibilitaram a integração da aplicação com a tecnologia blockchain de forma eficiente e confiável. Também foram utilizadas bibliotecas que possuem a estrutura base de diversos tipos de contratos, que serão necessários ao longo do desenvolvimento.

3.1.3.1 Web3.Js

A ferramenta web3.js é uma coleção de bibliotecas que permite a interação com um nó local ou remoto da Ethereum, diretamente em arquivos de extensão ".Js", utilizando um canal de comunicação através de um provedor *Remote Procedure Call* (RPC) ou *Inter-process Communication* (IPC). Para trabalhar com essa ferramenta, é necessário estar conectado a um nó da rede. Utilizou-se a API para interagir com os contratos e coletar os dados utilizados para a análise e tomada de decisão. Além disso, a web3.js permite executar operações diretamente por meio do código em JavaScript, como a compra e venda de ativos, o que contribui para a automatização do processo de arbitragem.

3.1.3.2 UniSwap V-2 Core

A biblioteca UniSwap V-2 Core contém diversos arquivos no formato JSON que são referentes à ABI dos protocolos dos contratos utilizados pelas exchanges. Alguns desses contratos são:

- ERC-20: Padrão utilizado pelos *tokens* negociados nas exchanges.
- Factory: Utilizado pela UniSwap e SushiSwap, responsável pela criação de cada *pool* de liquidez de cada par de tokens negociado na Exchange. Existe uma função chamada "createPair", na qual é passado o endereço de cada um dos tokens que compõe o par, e esse contrato inteligente cria outro contrato inteligente chamado "Pair Contract".
- Pair Contract: refere-se ao contrato inteligente do pool de liquidez dos pares de tokens contidos nas exchanges.

3.1.3.3 UniSwap V-2 Periphery

A biblioteca UniSwap V-2 Periphery contém a estrutura da ABI dos contratos "Router" das exchanges, utilizada principalmente para coletar informações dos contratos e interagir diretamente com os pares, através da web3.js.

3.1.3.4 Dotenv

A biblioteca dotenv é uma dependência utilizada para carregar variáveis de ambiente em aplicações Node.js. As variáveis de ambiente são valores configurados em um arquivo ".env", no diretório raiz do projeto, e podem ser usadas para configurar diferentes aspectos da aplicação, como credenciais de banco de dados, chaves de API, endereços de servidor, entre outros. A vantagem de usar variáveis de ambiente é que elas permitem alterar configurações sem precisar mexer no código-fonte do aplicativo e sem precisar fazer commits de alterações indesejadas, o que facilita a manutenção e a ocultação de dados sensíveis, garantindo uma maior segurança.

3.1.4 Node as a service

O serviço "Node as a Service" em criptografia refere-se a uma oferta em que um provedor de serviços fornece aos usuários um nó (ou múltiplos nós) de blockchain em funcionamento para uso. Existem duas maneiras de se comunicar com um nó na rede Ethereum: a primeira é através de um nó remoto, utilizando o *Remote Procedure Call* (RPC), a segunda é utilizando um nó instanciado localmente, utilizando o *Inter-process Communication* (IPC). Com o RPC, é possível acessar o nó remotamente, enviar solicitações e receber respostas por

meio de uma interface JSON. Já com o IPC, a comunicação ocorre entre processos no mesmo sistema operacional, oferecendo um desempenho melhor do que o RPC. A escolha entre as duas opções depende das necessidades e dos requisitos do projeto em questão.

3.1.4.1 *Remote Procedure Call* (RPC)

Neste trabalho, foi feito o uso de um nó RPC. Um nó *Remote Procedure Call* (RPC) é um tipo de software que permite que outros programas executem funções ou procedimentos em um sistema remoto. O protocolo RPC é utilizado para enviar solicitações de um cliente para um servidor e para o servidor retornar uma resposta ao cliente. Os nós RPC são comumente utilizados em sistemas distribuídos e ambientes de rede para permitir a comunicação e a troca de dados entre diferentes sistemas e aplicativos, conforme mencionado por Larimer (2018).

A comunicação com um nó Ethereum via RPC (*Remote Procedure Call*) é realizada por meio de uma API (*Application Programming Interface*), que permite que outros programas se comuniquem com o nó. Isso é feito por meio de uma URL de *endpoint*, que normalmente é "http://localhost:8545" para nós locais e pode ser configurado para se conectar a nós remotos através de protocolos como HTTP ou WebSockets.

A API da Ethereum oferece uma variedade de métodos que permitem aos usuários obter informações sobre contas, enviar transações e verificar o estado da rede. Para se comunicar com um nó Ethereum por meio do RPC, é necessário utilizar uma biblioteca de cliente RPC, como a web3.js. Essa biblioteca fornece uma interface JavaScript que permite aos usuários acessar as funcionalidades do nó da rede, como enviar transações, verificar o saldo de contas e interagir com contratos inteligentes. Além disso, a web3.js torna fácil e seguro o acesso às informações do nó Ethereum, eliminando a necessidade de lidar diretamente com as complexidades do protocolo RPC.

3.1.4.2 Protocolos de requisição e resposta

O HTTP é um protocolo baseado em requisições e respostas. Quando um cliente envia uma requisição HTTP para um servidor, o servidor retorna uma resposta. Essa comunicação é geralmente "assíncrona" e é usada para transferir arquivos estáticos (como imagens, HTML, CSS) e dados dinâmicos (como JSON) entre o cliente e o servidor.

WebSockets, por outro lado, é um protocolo bidirecional que permite comunicação em tempo real entre o cliente e o servidor. Ele é projetado para permitir que os dados sejam transmitidos de forma eficiente e contínua, ao invés de uma comunicação baseada em

requisições e respostas. Isso é útil para aplicações como jogos, chats, e sistemas de monitoramento em tempo real.

Como o processo de arbitragem necessita de um monitoramento constante e em tempo real, o protocolo adotado foi o *WebSockets*.

3.1.4.3 QuickNode

A QuickNode é uma plataforma de hospedagem de nós de redes blockchains, como a Ethereum, e permite que os usuários implantem e gerenciem seus próprios nós facilmente. Com a QuickNode, é possível implantar nós Ethereum em diferentes regiões geográficas, escolher entre diferentes tipos de nós, como *full nodes* ou *light nodes*, e gerenciar suas configurações de forma fácil e intuitiva. Além disso, a plataforma fornece uma API de nós, que permite aos usuários acessar e interagir por meio de chamadas RPC (*Remote Procedure Call*), sendo uma escolha ideal para o desenvolvimento do projeto (QUICKNODE, 2022).

Com o serviço de nó remoto, tornou-se possível concentrar-se no desenvolvimento da aplicação e na interação com a rede Ethereum, sem a necessidade de se preocupar com a configuração e gerenciamento dos nós subjacentes. Isso resultou em economia de tempo e esforço, além de proporcionar mais segurança e escalabilidade em comparação com o gerenciamento de um nó Ethereum local.

3.1.5 Configuração do ambiente de desenvolvimento

Inicialmente, foi necessário preparar o ambiente de desenvolvimento. O sistema operacional escolhido foi o Ubuntu, pois ele oferece mais flexibilidade e suporte durante o uso das ferramentas utilizadas. Os códigos foram escritos na interface de desenvolvimento VSCode, onde é possível baixar extensões para destacar as linguagens, neste caso, o JavaScript. O web3.js e outras dependências do node.js foram gerenciadas pelo Node Package Manager (NPM).

3.2 Implementação do sistema

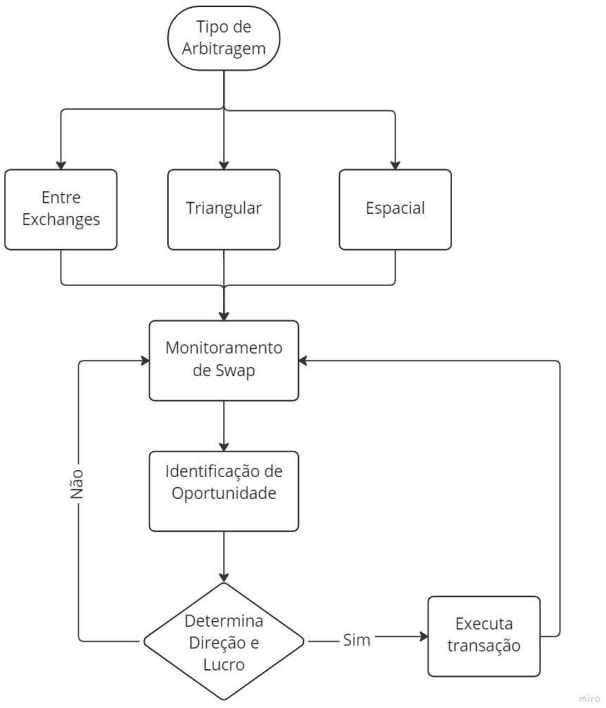
Nesta seção, serão descritos os procedimentos adotados para a implementação do sistema proposto. Com base nos conceitos descritos no referencial teórico e nas tecnologias informadas na seção anterior. Com isso, é objetivado fornecer uma visão completa e detalhada do processo de implementação do sistema, permitindo aos leitores entender como o algoritmo funciona e como ele foi desenvolvido de forma a atender aos objetivos propostos.

3.2.1 Idealização do algoritmo

O projeto de um algoritmo de arbitragem pode ser dividido em alguns passos, exemplificado pela Figura 7 e sendo eles:

1. Determinação do tipo de arbitragem.
2. Monitoramento de *Swap*: O algoritmo deve ser capaz de monitorar eventos e identificar quando ocorreu uma transação.
3. Identificação de oportunidades: O algoritmo deve ser capaz de identificar as oportunidades de arbitragem, ou seja, quando o preço de uma criptomoeda é diferente em duas ou mais exchanges.
4. Determinação da direção da transação.
5. Determinação da possibilidade de lucro na operação.
6. Execução da operação.
7. Verificação de lucro.

Figura 7- Fuxo de desenvolvimento.



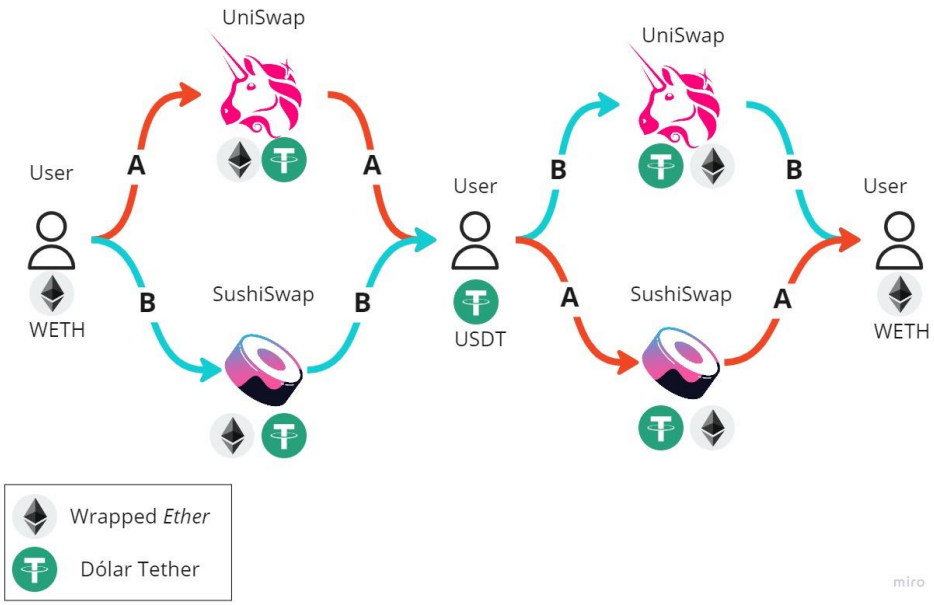
Fonte: Do Autor (2023).

Para este trabalho, foi escolhida a arbitragem entre exchanges descentralizadas de uma mesma rede blockchain, a rede Ethereum. A arbitragem ocorrerá entre as exchanges UniSwap e SushiSwap, sendo a SushiSwap considerada um *fork* do projeto da UniSwap, apresentando

características similares. Uma das grandes vantagens de realizar a arbitragem entre as duas exchanges é a compatibilidade entre as dependências UniSwap V-2 Core e UniSwap V-2 Periphery, facilitando o uso dos métodos das ABI dos contratos inteligentes factory, router e pair para ambas as exchanges.

A partir da Figura 8 é possível analisar o fluxo de compra e venda do algoritmo. Inicialmente o usuário irá conter o *token* WETH (Wrapped Ether). Posteriormente será verificado em qual exchange é possível obter a maior quantidade do token USDT (Dólar Tether), e a troca será efetuada. Em seguinda será realizada a troca na exchange oposta, obtendo então uma maior quantidade do token WETH. Ao finalizar o fluxo, é esperado que o usuário tenha uma quantidade de WETH superior ao do início do fluxo.

Figura 8- Fluxo de compra e venda.



Fonte: Do Autor (2023).

3.2.2 Algoritmo

Dividir um algoritmo em vários arquivos pode ser uma técnica muito útil para simplificar a manutenção e a evolução do código, tornando o processo de desenvolvimento mais eficiente e menos propenso a erros. O algoritmo apresentado neste trabalho foi dividido conforme os arquivos descritos a seguir:

- .env: Presente no Apêndice A, contém os principais dados sensíveis e variáveis de ambiente utilizadas no projeto. A partir dele é possível modificar dados, como o endereço dos ativos a serem arbitrados e a chave da API do serviço de node, em um

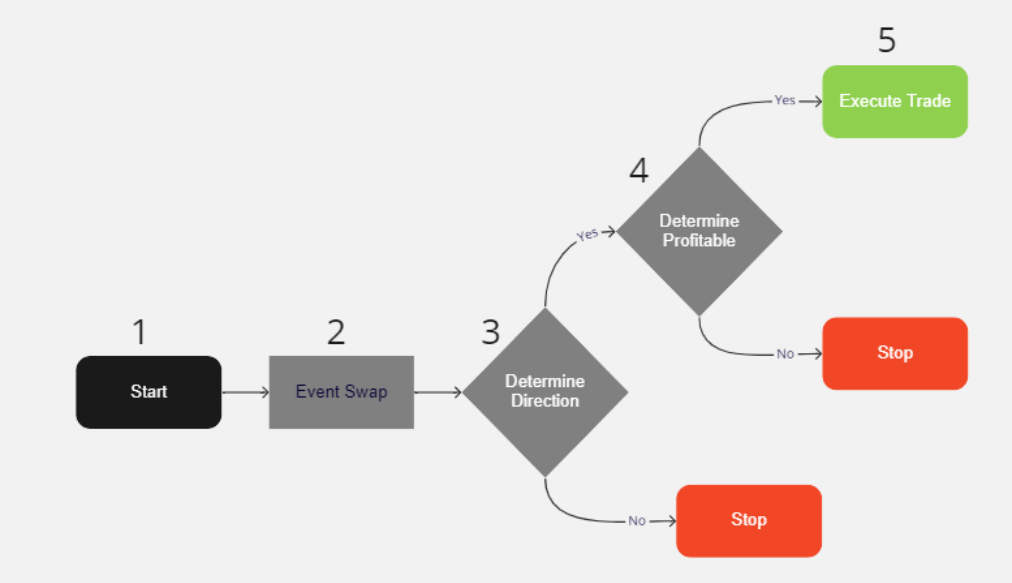
único local, sem que seja necessário alterar nas chamadas que ocorrem em trechos do algoritmo.

- config.json: Presente no Apêndice B, de forma semelhante ao ".env", também possui dados que são utilizados nos algoritmos, porém na extensão ".json". As informações contidas são referentes ao endereço do contrato factory e router de ambas as exchanges.
- inicializa.js: Presente no Apêndice C, possui a finalidade de estabelecer a conexão com o serviço de nó da quikNode e instanciar os contratos factory e router de ambas as exchanges. Os contratos e a classe web3 são exportados e podem ser importados posteriormente por classes internas.
- auxiliares.js: Presente no Apêndice D, contém as principais funções que serão utilizadas pela classe "main.js". Algumas funções como a "getPairAddress", "getPairContract" podem ser utilizadas não só em algoritmos de arbitragem, mas também em diversos tipos de algoritmos que necessitam de consultas diretas na rede blockchain.
- main.js: Presente no Apêndice E, contém a função principal do algoritmo, a partir dela todo o script entra em funcionamento. A "main.js" importa os demais arquivos apresentados anteriormente e com isso consegue fazer uso de seus atributos e métodos.

3.2.3 Fluxo de execução

De acordo com o diagrama apresentado na Figura 9, é possível descrever o fluxo de execução e condicionamento do algoritmo.

Figura 9 - Fluxo de execução.



Fonte: Do Autor (2023).

1. O algoritmo é inicializado por meio do comando "node .\main.js" que deverá ser executado na pasta de origem do projeto. Posteriormente, a função assíncrona "main" é chamada.
2. Após a inicialização do projeto, passa a ser monitorado os eventos de troca no contrato dos pares de *tokens* em cada uma das exchanges. Assim que um evento de troca ocorre, é chamada a próxima função.
3. A função "checkPrice" é chamada, a qual obtém a diferença percentual de preço dos ativos entre as exchanges UniSwap e SushiSwap. Após obter a diferença de preços, é chamada então a função "determineDirection" que retornará a direção de compra e venda dos ativos, ou seja, em qual Exchange deverá ser comprado o ativo com WETH e em qual deverá ser vendido o ativo por WETH.
4. Após obter a direção, serão verificados o lucro e a viabilidade de executar a transação. Isso será feito através da função "determineProfitability" que realizará os cálculos baseando-se nas taxas de gás que deverão ser pagas e nas taxas cobradas pelas exchanges. Se a operação tiver previsão de lucratividade, é passado para a etapa de execução do trade.
5. A execução do trade será a última etapa a ser realizada. Após a execução, será exibido um balanço dos retornos obtidos. A execução poderá ser realizada de diferentes formas, como o uso de contratos de empréstimo como *flash loans*, ou com execução direta com os próprios ativos que possuir em carteira.

4 RESULTADOS E DISCUSSÃO

Para testar o software, foi criado um cenário fictício. Nessa simulação, foram selecionados dois *tokens* para serem arbitrados: o USDT, um ativo estável atrelado ao dólar americano, e o WETH, que é uma versão do ether compatível com o padrão ERC-20, permitindo sua utilização em aplicações descentralizadas na rede Ethereum, como as exchanges descentralizadas.

Ao iniciar a aplicação é exibido no terminal BASH as informações referentes ao endereço do contrato dos pares em cada uma das exchanges, conforme a Figura 10.

Figura 10 - Endereço do par.

```
Pair Token Uniswap: 0x0d4a11d5EEaaC28EC3F61d100daF4d40471f1852
Pair Token SushiSwap: 0x06da0fd433C1A5d7a4faa01111c044910A184553
```

Fonte: Do Autor (2023).

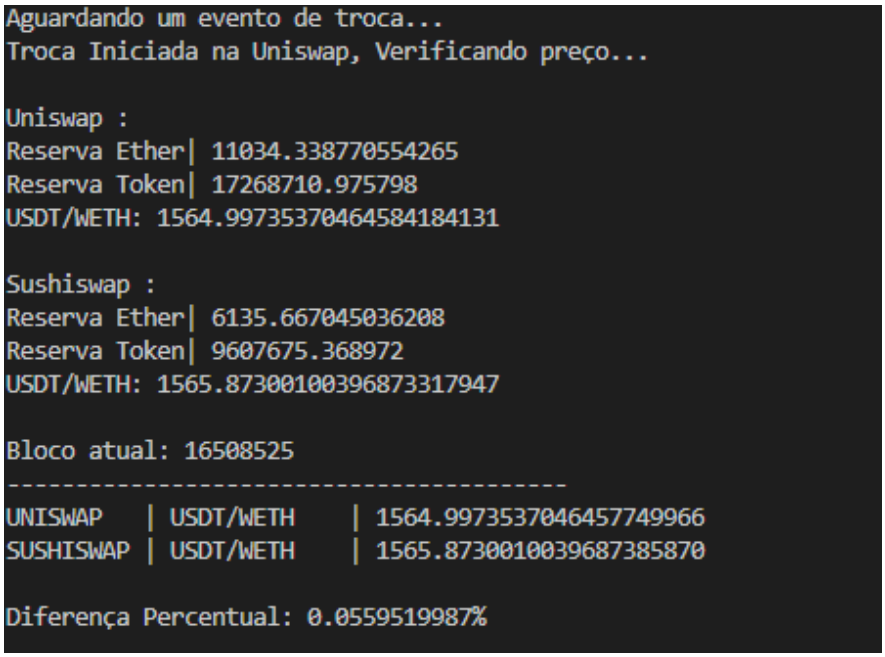
No momento em que a aplicação é inicializada, uma função do tipo assíncrona é executada, e a partir dela os contratos referentes aos pares dos *tokens* que provém a liquidez em cada uma das exchanges são monitorados.

Sempre que um novo bloco é validado, um novo conjunto de transações é efetuada, caso alguma das transações envolva os contratos monitorados, uma verificação será feita, para identificar possíveis oportunidade de arbitragem.

A verificação consiste em obter a quantidade da reserva de cada *token* presente nos contratos do pool de liquidez de cada Exchange e calcular a relação entre eles. A partir de então é feito o cálculo percentual da relação entre cada exchange, conforme demonstrado na Figura 11. Caso o valor percentual seja maior que o valor definido na constante “PRICE_DIFFERENCE” do arquivo “.env” o algoritmo chama a função referente a próxima a etapa, a determinação da direção da arbitragem.

A constante “PRICE_DIFFERENCE” refere ao valor mínimo da diferença de valor entre as exchanges para que seja viável a execução da arbitragem. Para definir esse valor deve ser considerado algumas taxas, como a cobrada pelas exchanges que são de aproximadamente 0,3% e a taxa de *slippage*, que consiste na diferença entre o preço que é esperado pagar ou receber em uma ordem de compra ou venda e o preço real. A previsão da taxa de *slippage* requer cálculos mais complexos, e não foi aplicada neste contexto. Para fins experimentais foi definido um valor extremamente baixo para “PRICE_DIFFERENCE”, com a finalidade de buscar oportunidades de arbitragem em um tempo menor.

Figura 11 - Identificação de oportunidade.

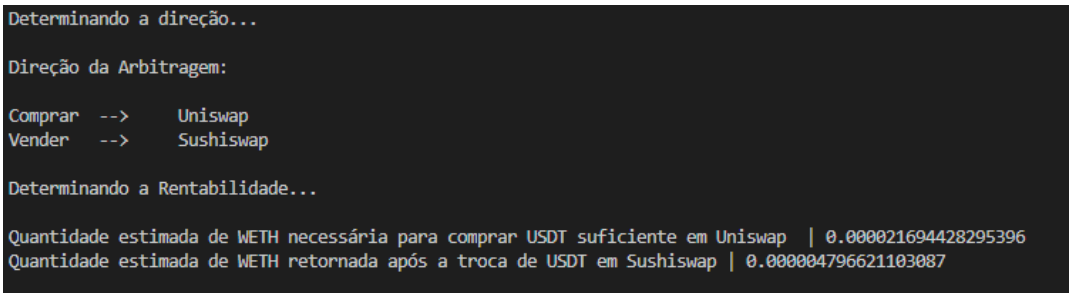


Fonte: Do Autor (2023).

Após a identificação de uma oportunidade de arbitragem, é determinado então a direção da arbitragem, onde deverá ser comprado e posteriormente vendido os ativos.

Com a direção identificada, o próximo passo consiste em identificar a quantidade a ser arbitrada, ou seja, a quantidade de USDT que deverá ser comprado com WETH e o valor estimado do retorno após a venda na segunda exchange, conforme a Figura 12.

Figura 12 - Direção e rentabilidade.



Fonte: Do Autor (2023).

Além das informações citadas, uma tabela é apresentada com os saldos da criptomoeda da rede, o ETH (ether), que é utilizada para pagamento das taxas cobradas pela execução das transações pela rede Ethereum, e os saldos em *token* WETH, o utilizado nas negociações nas exchanges. Como o ETH e o WETH possui valor corresponde, é possível estabelecer uma

relação e identificar o saldo que será perdido ou ganhado após a execução das transações, o resultado simulado está presente na Figura 13.

Figura 13 - Estimativa de saldo em ETH e WETH.

(index)	Values
Saldo ETH Antes	'0.030164112963815492'
ETH gasto (gás)	'0.0000093'
Saldo ETH após	0.030154812963815492
-	
Saldo WETH Antes	'0'
Saldo WETH após	1.4258286487800196e-7
WETH Gained/Lost	1.4258286487800196e-7
Total ganho/perdido	-0.000009157417135121999

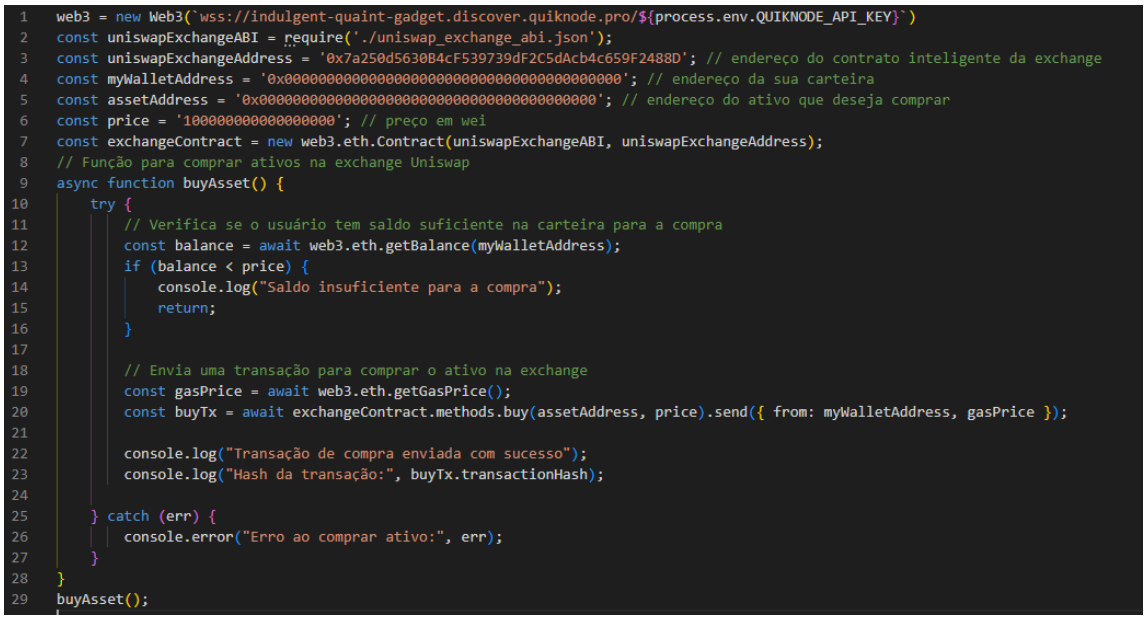
Fonte: Do Autor (2023).

4.1 Execução

A simulação ilustrada não consegue efetuar o processo de arbitragem devido ao fato do o algoritmo ter sido desenvolvido e testado com as configurações da rede principal. Isto significa que a sua execução requereria recursos reais, tornando-se inviável em um ambiente de teste.

Existem duas alternativas para a realização da execução. A primeira é através de contratos de empréstimo com "*Flash Loans*", que não requerem uma quantidade significativa de recursos presentes na carteira, mas precisam de uma quantidade de ether para o deploy do contrato na rede Ethereum. A segunda opção é utilizar recursos próprios em todas as etapas do processo de arbitragem, um exemplo de algoritmo que auxiliaria nesse tipo de abordagem é apresentado na Figura 14.

Figura 14 - Algoritmo de execução.



```
web3 = new Web3(`wss://indulgent-quaint-gadget.discover.quiknode.pro/${process.env.QUIKNODE_API_KEY}`)
const uniswapExchangeABI = require('./uniswap_exchange_abi.json');
const uniswapExchangeAddress = '0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D'; // endereço do contrato inteligente da exchange
const myWalletAddress = '0x0000000000000000000000000000000000000000'; // endereço da sua carteira
const assetAddress = '0x0000000000000000000000000000000000000000'; // endereço do ativo que deseja comprar
const price = '10000000000000000'; // preço em wei
const exchangeContract = new web3.eth.Contract(uniswapExchangeABI, uniswapExchangeAddress);
// Função para comprar ativos na exchange Uniswap
async function buyAsset() {
    try {
        // Verifica se o usuário tem saldo suficiente na carteira para a compra
        const balance = await web3.eth.getBalance(myWalletAddress);
        if (balance < price) {
            console.log("Saldo insuficiente para a compra");
            return;
        }

        // Envia uma transação para comprar o ativo na exchange
        const gasPrice = await web3.eth.getGasPrice();
        const buyTx = await exchangeContract.methods.buy(assetAddress, price).send({ from: myWalletAddress, gasPrice });

        console.log("Transação de compra enviada com sucesso");
        console.log("Hash da transação:", buyTx.transactionHash);

    } catch (err) {
        console.error("Erro ao comprar ativo:", err);
    }
}
buyAsset();
```

Fonte: Do Autor (2023).

Este exemplo utiliza a função "buy()" do contrato inteligente da UniSwap para realizar uma transação de compra na exchange. Antes de enviar a transação, é verificado se o usuário possui saldo suficiente na carteira. Caso haja algum erro durante a transação, ele é identificado e exibido na tela. Para vender um ativo, basta usar a função "sell()" do contrato inteligente da UniSwap, informando o endereço do ativo e o valor desejado como argumentos. A lógica do algoritmo é semelhante, mas chama a função "sell()" ao invés de "buy()".

5 CONCLUSÃO

A presente monografia alcançou com êxito seus objetivos iniciais ao apresentar um referencial teórico sobre as tecnologias intrinsecamente relacionadas às exchanges descentralizadas. Ademais, foi realizado o estudo das principais estratégias de transação de alta frequência, enfatizando a arbitragem, o que permitiu estudar e compreender os principais fatores que afetam diretamente o desempenho dos algoritmos e os diferenciam das exchanges centralizadas.

O referencial teórico apresentado foi fundamental para o desenvolvimento do algoritmo de arbitragem, pois possibilitou o entendimento do fluxo de execução das transações e dos principais termos envolvidos nas transações de alta frequência quando aplicadas às exchanges descentralizadas. Isso auxiliou na identificação dos principais fatores que podem influenciar o processo de análise e execução das transações e possibilitou a elaboração de estratégias para minimizar os riscos envolvidos.

Com base nos resultados obtidos, é possível confirmar que a aplicação desenvolvida pode servir como base para um modelo de arbitragem entre exchanges descentralizadas. Entretanto, é importante destacar que o algoritmo apresenta diversas limitações que precisam ser mitigadas para obter um desempenho lucrativo. Portanto, em trabalhos futuros relacionados ao tema, é fundamental que essas limitações sejam eliminadas, a fim de se aproximar de uma estratégia eficaz.

Em síntese, este estudo contribui para o entendimento e o fortalecimento do mercado de ativos digitais ao apresentar uma análise das transações de alta frequência nas exchanges descentralizadas. Contribuindo dessa forma para estudos futuros e favorecendo o desenvolvimento de novos modelos de negócios nesse ambiente.

5.1 Trabalhos futuros

Existem diversas ideias de melhorias para trabalhos futuros, que surgiram durante o desenvolvimento do projeto. Algumas delas incluem:

- Nó de conexão: Com a adoção do protocolo de consenso *Proof of Stake* pela rede Ethereum, buscando garantir o funcionamento constante, hoje a maioria dos nós é hospedada em plataformas de computação em nuvem, como a AWS (*Amazon Web Services*). Uma alternativa para garantir o funcionamento contínuo dos algoritmos HFT e reduzir a latência na obtenção de informações dos blocos, seria hospedar o algoritmo em um servidor semelhante e próximo à localidade da maioria dos nós. Além disso,

executar um nó próprio é importante, pois elimina a dependência de serviços de nós externos, que são suscetíveis a vários tipos de atrasos na transmissão de informações.

- Monitoramento de *swap*: Uma alternativa que mudaria completamente a dinâmica do algoritmo e seria significativamente útil, é o monitoramento da mempool e a tomada de decisões a partir dela. Assim, as decisões seriam em tempo real, dependendo apenas da latência entre os nós da rede e da capacidade de processamento da máquina hospedada. O algoritmo apresentado toma decisões baseado em blocos já minerados, o que torna sua competitividade extremamente baixa.
- Determinação do valor a ser arbitrado: A modelagem e implementação de uma equação para otimizar o valor a ser arbitrado seria de grande importância. As taxas da rede estão em constante mudança, além das taxas de *slippage*, que podem comprometer a arbitragem em caso de distorções de preço significativas. É importante definir cuidadosamente o valor utilizado para garantir um maior rendimento.

Além dos pontos citados, é importante analisar e estudar outros tipos de estratégias de arbitragem e modelos de algoritmos de transação de alta frequência. Os protocolos e serviços estão em constante evolução e manter-se atualizado e buscar novas oportunidades é fundamental para o sucesso na utilização dos HFTs.

REFERÊNCIAS

FISHER, R.; URY, W. **Getting to yes: Negotiating agreement without giving in**. Penguin, 1991.

ROLNIK, R. **A história das bolsas de valores**. São Paulo: Editora FGV, 2010.

LO, A. W.; MACKINLAY, A. C**. An econometric analysis of the limit order book and the order flow in the NYSE**. Journal of Financial Economics, 1990.

ALMGREN, R.; CHRISS, N. **Optimal execution of portfolio transactions**. Journal of Risk, 2000.

BUTERIN, V. **A Next-Generation Smart Contract and Decentralized Application Platform.** [S.I.]. Ethereum Foundation, 2014. Disponível em https://github.com/ethereum/wiki/wiki/White-Paper, Acesso em: 21 jan. 2023.

BACK, A.; CORALLO, M.; DASHJR, L.; FRIEDENBACH, M.; MAXWELL, G.; MILLER, A.; POELSTRA, A.; TIMN, J.; WUILLE, P. **Enabling Blockchain Innovations with Pegged Sidechains**. Blockstream, 2018.

GÓMEZ; RAMÍREZ, J.; LIZCANO, D. **High-Frequency Trading in Decentralized Exchanges**. Journal of Trading, 2019.

NAKAMOTO, S. **Bitcoin: A Peer-to-Peer Electronic Cash System**. [S.I.]. Bitcoin, 2008. Disponível em: https://bitcoin.org/bitcoin.pdf. Acesso em: 21 jan. 2023.

TAPSCOTT, D.; TAPSCOTT, A. **Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world**. Penguin, 2016.

NASH, M. **The Basics of Hashing Algorithms.** [S.I.]. Medium, 2020. Disponível em: Medium.com. Acesso em: 21 jan. 2023.

NARAYANAN, A.; BONNEAU, J.; FELTEN, E.; MILLER, A.; GOLDFEDER, S. **Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction.** Princeton University Press, 2016.

SOMPOLINSKY, Y.; ZOHAR, A. **Secure high-rate transaction processing in Bitcoin.** Financial Cryptography and Data Security, FC 2015 International Workshops, BITCOIN, WAHC, 2015.

TANENBAUM, A. S., & STEEN, M. V. **Computer networks.** Pearson, 2017.

SCHRIJVERS, O., BONNEAU, J., BONEH, D., BÜNZ, B., CIALDINI, J., & VAN RENESSE, R. **A survey of attacks on Ethereum smart contracts.** Cryptology ePrint Archive, 2019.

RIVEST, R. L., SHAMIR, A., & ADLEMAN, L. **A method for obtaining digital signatures and public-key cryptosystems.** Communications of the ACM, 1978.

COINMAMA. **What is a Wallet in Cryptocurrency?**. [S.I.]. Coinmama, 2021. Disponível em https://www.coinmama.com/blog/cryptocurrency-wallet-explained/. Acesso em: 21 jan. 2023.

BITDEGREE. **Types of Crypto Wallets: A Beginner's Guide.** [S.I.]. Bitdegree, 2020. Disponível em: https://www.bitdegree.org/tutorials/types-of-crypto-wallets/. Acesso em: 21 jan. 2023.

BLOCKGEEKS. **Hot vs Cold Wallets: Which is the Best Option for Storing Cryptocurrency?.** [S.I.]. Blockgeeks, c2018a. Disponível em: https://blockgeeks.com/guides/hot-vs-cold-wallets/. Acesso em: 21 jan. 2023.

BLOCKGEEKS. **A Beginner's Guide to Ethereum**. [S.I.]. Blockgeeks, c2018b. Disponível em: https://blockgeeks.com/guides/ethereum/. Acesso em: 21 jan. 2023.

BLOCKGEEKS. **What is a Turing Complete Language?.** [S.I.]. Blockgeeks, c2018c. Disponível em: https://blockgeeks.com/guides/turing-complete-language/. Acesso em: 21 jan. 2023.

GEEKSFORGEEKS. **Understanding RSA Algorithm with Example**s. [S.I.]. GeeksforGeeks, 2021. Disponível em: https://www.geeksforgeeks.org/understanding-rsa-algorithm-with-examples/. Acesso em: 21 jan. 2023.

BITCOIN MAGAZINE. **Schnorr Signatures: The Future of Bitcoin Transactions**. [S.I.]. Bitcoin Magazine, 2018. Disponível em: https://bitcoinmagazine.com/articles/schnorr-signatures-future-bitcoin-transactions/. Acesso em: 21 jan. 2023.

INVESTOPEDIA. **Ethereum Mining: How It Works and What to Know**. [S.I.]. Investopedia, 2021. Disponível em: https://www.investopedia.com/terms/e/ethereum-mining.asp. Acesso em: 21 jan. 2023.

CONSENSYS. **Ethereum Virtual Machine: An Overview**. [S.I.]. Consensys, 2020. Disponível em: https://consensys.net/blockchain-101/ethereum-virtual-machine-overview/. Acesso em: 21 jan. 2023.

99BITCOINS. **The Turing Complete Concept Explained** . [S.I.]. 99Bitcoins, 2019. Disponível em: https://99bitcoins.com/turing-complete-concept-explained/. Acesso em: 21 jan. 2023.

ETHEREUM. **Ethereum Token Standards**. [S.I.]. Ethereum, 2023. Disponível em: https://ethereum.org/en/developers/docs/standards/tokens/#top. Acesso em: 21 jan. 2023.

ZOOKO, Z. **Trilema de Zooko**. [S.I.]. Zooko, 2021. Recuperado de https://zooko.com/trilema/. Acesso em: 21 jan. 2023.

PANCAKESWAP**. Sobre PancakeSwap**. [S.I.]. PancakeSwap, 2023. Recuperado de https://pancakeswap.finance/. Acesso em: 21 jan. 2023.

DINUCCI, D. **Fragmented Future.** [S.I.]. Webstyleguide, 1999. Disponível em: https://www.webstyleguide.com/wsg3/3-information-architecture/4-site-organization/fragmented-future.html. Acesso em: 21 jan. 2023.

O'REILLY, T. **What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software**. [S.I.]. O'Reilly, T., 2005. Disponível em: https://www.oreilly.com/library/view/what-is-web-20/9780596526848/. Acesso em: 21 jan. 2023.

BERNERS-LEE, T. (1989). **Information Management: A Proposal**. [S.l.]. Disponível em: https://www.w3.org/History/1989/proposal.html. Acesso em: 21 jan. 2023.

INTERNET HALL OF FAME. **The Birth of the Internet**. [S.l.]. Internet Hall of Fame, 2019. Disponível em: https://www.internethalloffame.org/the-birth-of-the-internet. Acesso em: 21 jan. 2023.

TAPSCOTT, D., & TAPSCOTT, A. **Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world.** Penguin, 2016.

BLOCKCHAIN INFO. **Decentralized Exchanges Explained**. Blockchain.info, 2023. Disponível em: https://www.blockchain.com/learn/decentralized-exchanges. Acesso em: 21 jan. 2023.

BINANCE. **The Complete Beginner's Guide to Decentralized Finance (DeFi).** [S.l.]. Binance Academy 2019. Disponível em: https://academy.binance.com/pt/articles/the-complete-beginners-guide-to-decentralized-finance-defi. Acesso em: 21 jan. 2023.

BINANCE. **An Introduction to Binance Smart Chain (BSC).** [S.l.]. Binance Academy, c2020a. Disponível em: https://academy.binance.com/pt/articles/an-introduction-to-binance-smart-chain-bsc. Acesso em: 21 jan. 2023.

BINANCE. **Automated Market Maker.** [S.l.]. Binance Academy, c2020b. Disponível em: https://academy.binance.com/pt/articles/what-is-an-automated-market-maker-amm22. Acesso em: 21 jan. 2023.

BINANCE. **What is a Decentralized Exchange (DEX)**. [S.l.]. Binance Academy, c2020b. Disponível em: https://academy.binance.com/pt/articles/what-is-a-decentralized-exchange-dexem. Acesso em: 21 jan. 2023.

BINANCE. **Impermanent Loss Explained**. [S.l.]. Binance Academy, c2020c. Disponível em: https://academy.binance.com/en/articles/impermanent-loss-explained. Acesso em: 21 jan. 2023.

BINANCE. **What is an IDO Initial Dex Offering**. [S.l.]. Binance Academy, c2020d. Disponível em: https://academy.binance.com/en/articles/what-is-an-ido-initial-dex-offering. Acesso em: 21 jan. 2023.

BINANCE. **What Are Investment DAOs.** [S.l.]. Binance Academy, c2021a. Disponível em: https://academy.binance.com/pt/articles/what-are-investment-daos. Acesso em: 21 jan. 2023.

BINANCE. **Centralized Exchanges: An Overview.** [S.l.]. Binance Academy, c2021b https://academy.binance.com/pt/articles/centralized-exchanges-an-overview, Acessado em: 22 Janeiro de 2021.

BINANCE. **What Are Flash Loans in DeFi?.** [S.l.]. Binance Academy, c2022a. Disponível em: https://academy.binance.com/pt/articles/the-complete-beginners-guide-to-decentralized-finance-defi. Acesso em: 21 jan. 2023.

BINANCE. **What Are Liquidity Pool LP Tokens.** [S.l.]. Binance Academy, c2022b. Disponível em: https://academy.binance.com/en/articles/what-are-liquidity-pool-lp-tokens. Acesso em: 21 jan. 2023.

ALDRIDGE, Irene. **High-Frequency Trading: A Practical Guide to Algorithmic Strategies and Trading Systems**. Ed. Wiley, 2013.

ESMA**. High-Frequency Trading and its Impact on Market Quality**. The European Securities and Markets Authority, 2015.

DOE, J. **Fatores que afetam a estratégia de HFT em DeFi**. Revista de Finanças Decentralizadas, 2022.

DAIAN, P .et al. **Flash Boys 2.0: Frontrunning, Transactions Reordering, and Consensus.** Cornell University, 2019.

LARIMER, D. **Mastering Ethereum: Building Smart Contracts and Dapps**. Packt Publishing, 2018.

QUICKNODE. **QuickNode: Plataforma de Hospedagem de Nós de Blockchains**. [S.l.]. QuickNode ,2022. Disponível em: https://www.quicknode.com/pt-br/nodes/ethereum. Acesso em: 21 jan. 2023.

APÊNDICE A — ARQUIVO DOTENV

```
QUIKNODE_API_KEY="XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX/"
ARB_FOR="0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2"
ARB_AGAINST="0xdac17f958d2ee523a2206206994597c13d831ec7"
ACCOUNT="XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
PRICE_DIFFERENCE=0.03
UNITS=18
GAS_LIMIT=600000
GAS_PRICE=0.0000000355
```

APÊNDICE B — ARQUIVO CONFIG JSON

```
{
    "UNISWAP": {
        "V2_ROUTER_02_ADDRESS": "0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D",
        "FACTORY_ADDRESS": "0x5C69bEe701ef814a2B6a3EDD4B1652CB9cc5aA6f"
    },
    "SUSHISWAP": {
        "V2_ROUTER_02_ADDRESS": "0xd9e1cE17f2641f24aE83637ab66a2cca9C378B9F",
        "FACTORY_ADDRESS": "0xC0AEe478e3658e2610c5F7A4A2E1777cE9e4f2Ac"
    }
}
```

APÊNDICE C — ARQUIVO INICIALIZA

```
//Importa a dependência "dotenv".
require("dotenv").config();

//requer o endereço do Router e do Factory das duas exchanges contidos no arquivo "config.json".
const config = require('./config.json')

//requer a biblioteca "web3.js".
const Web3 = require('web3')
let web3

//Instancia a classe Web3 e realiza a conexão com o provedor.
web3 = new Web3(`wss://indulgent-quaint-gadget.discover.quiknode.pro/${process.env.QUIKNODE_API_KEY}`)

// abi uniswapRouter
const IUniswapV2Router02 = require('@uniswap/v2-periphery/build/IUniswapV2Router02.json')

// abi uniswapFactory
const IUniswapV2Factory = require("@uniswap/v2-core/build/IUniswapV2Factory.json")

// Instância do Factory e do Router Uniswap
const uFactory = new web3.eth.Contract(IUniswapV2Factory.abi, config.UNISWAP.FACTORY_ADDRESS) // UNISWAP FACTORY CONTRACT
const uRouter = new web3.eth.Contract(IUniswapV2Router02.abi, config.UNISWAP.V2_ROUTER_02_ADDRESS) // UNISWAP ROUTER CONTRACT

// Instância do Factory e do Router Sushiswap
const sFactory = new web3.eth.Contract(IUniswapV2Factory.abi, config.SUSHISWAP.FACTORY_ADDRESS) // SUSHISWAP FACTORY CONTRACT
const sRouter = new web3.eth.Contract(IUniswapV2Router02.abi, config.SUSHISWAP.V2_ROUTER_02_ADDRESS) // SUSHISWAP ROUTER CONTRACT


//Exporta as instâncias do arquivo initialization.js
module.exports = {
    // exporta instância uniswap Factory
    uFactory,
    // exporta instância uniswap Router
    uRouter,
    // exporta instância sushiswap Factory
    sFactory,
    // exporta instância sushiswap Router
    sRouter,
    // exporta web3
    web3
}
```

APÊNDICE D — ARQUIVO AUXILIARES

```
//requer a biblioteca "big.js, utilizada para aritmética decimal de precisão arbitrária.
const Big = require('big.js');

//requer o objeto "web3" do arquivo "initialization.js"
const { web3 } = require('./inicializa')

//requer o Enum "ChainId" e a classe "Token" da biblioteca "uniswap-sdk"
const { ChainId, Token } = require("@uniswap/sdk")

// abi da interface do pair smartContract.
const IUniswapV2Pair = require("@uniswap/v2-core/build/IUniswapV2Pair.json")

// Abi de um token ERC20.
const IERC20 = require('@openzeppelin/contracts/build/contracts/ERC20.json')
```

```

// recebe o endereço de dois tokens.
// instancia os dois contratos dos tokens.
async function getTokenAndContract(_token0Address, _token1Address) {

    // instância do smart contract do token0, fazendo uso da abi de um token padrão ERC20.
    const token0Contract = new web3.eth.Contract(IERC20.abi, _token0Address)

    // instância do smart contract do token1, fazendo uso da abi de um token padrão ERC20.
    const token1Contract = new web3.eth.Contract(IERC20.abi, _token1Address)

    /*
    -Instancia um objetivo Token do sdk da uniswap.
    -Coleta o Id do token referente a Main Net da Ethereum.
    -Coleta o endereço do token.
    -Obtem uma instância do token para trabalhar com a sdk da uniswap.
    */
        const token0 = new Token(
            ChainId.MAINNET,
            _token0Address,
            18, //Numero de decimais do token.
            await token0Contract.methods.symbol().call(), //Simbolo do token, exemplo: WETH.
            await token0Contract.methods.name().call()  //Nome do token, exemplo: Wrapped Ether.
        )

        const token1 = new Token(
            ChainId.MAINNET,
            _token1Address,
            18,
            await token1Contract.methods.symbol().call(),
            await token1Contract.methods.name().call()
        )

        // retorna o contrato e o objeto instanciado de cada token.
        return { token0Contract, token1Contract, token0, token1 }
}

// Obtem o endereço do par, passando o endereço do factory e os dois tokens.
async function getPairAddress(_V2Factory, _token0, _token1) {

    // utiliza o método getPair do factory para obter o endereço do par.
    const pairAddress = await _V2Factory.methods.getPair(_token0, _token1).call()
    return pairAddress
}
```

```
60
61
62  //Retorna uma instância do pairContract.
63  async function getPairContract(_V2Factory, _token0, _token1) {
64
65      //Obtem o endereço do par dos token, é passado o contrato do factory referente a exchange.
66      const pairAddress = await getPairAddress(_V2Factory, _token0, _token1)
67
68      //Instancia um objeto referente ao contrato do pair.
69      const pairContract = new web3.eth.Contract(IUniswapV2Pair.abi, pairAddress)
70      return pairContract
71  }
72
73
74  //Obtem a reserva de cada token do par.
75  async function getReserves(_pairContract) {
76
77      const reserves = await _pairContract.methods.getReserves().call()
78      return [reserves.reserve0, reserves.reserve1]
79  }
80
81  //Calcula o dos tokens baseado nas reservas.
82  async function calculatePrice(_pairContract, exchange) {
83
84      //salva as reservas no array.
85      const [reserve0, reserve1] = await getReserves(_pairContract)
86      let ether = Number(web3.utils.fromWei(reserve0.toString(),'ether'))
87      let token = Number(web3.utils.fromWei(reserve1.toString(),'mwei'))
88      //Imprime a relação dos valores no console.
89      console.log(`${exchange} :`)
90      console.log(`Reserva Ether| ${ether}`)
91      console.log(`Reserva Token| ${token}`)
92      console.log(`USDT/WETH: ${Big(token).div(Big(ether)).toString()} \n`)
93      return (Big(token).div(Big(ether)))
94  }
95
96  //calcula a porcentagem da diferença dos valores.
97  function calculateDifference(uPrice, sPrice) {
98
99      if (uPrice > sPrice){
100         return (((uPrice - sPrice) / sPrice) * 100).toFixed(10)
101     }
102     else{
103         return (((sPrice - uPrice) / uPrice) * 100).toFixed(10)
104     }
105 }
```

```
106
107 // calcula o retorno estimado
108 async function getEstimatedReturn(amount, _routerPath, _token0, _token1) {
109
110     // utiliza o getAmountsOut para calcular o retorno esperado
111     const trade1 = await _routerPath[0].methods.getAmountsOut(amount, [_token0.address, _token1.address]).call()
112     const trade2 = await _routerPath[1].methods.getAmountsOut(trade1[1], [_token1.address, _token0.address]).call()
113
114     // transforma de wei para ether
115     const amountOut = Number(web3.utils.fromWei(trade1[0], 'ether'))
116     const amountIn = Number(web3.utils.fromWei(trade2[1], 'ether'))
117
118 return { amountIn, amountOut }
119 }
120
121 //Exporta as funções.
122 module.exports = {
123     getTokenAndContract,
124     getPairAddress,
125     getPairContract,
126     getReserves,
127     calculatePrice,
128     calculateDifference,
129     getEstimatedReturn
130 }
```

APÊNDICE E — ARQUIVO PRINCIPAL

```
require("dotenv").config();
const config = require('./config.json')
const { getTokenAndContract, getPairContract, calculatePrice, getEstimatedReturn, getReserves, calculateDifference } = require('./auxiliares')
const { uFactory, uRouter, sFactory, sRouter, web3, arbitrage } = require('./inicializa')

//Coleta os valores do arquvio ".env"
const arbFor = process.env.ARB_FOR //Este é o endereço do token que estamos tentando arbitrar (WETH) e aumentar o volume.
const arbAgainst = process.env.ARB_AGAINST //Endereço do segundo token do par a ser negociado.
const account = process.env.ACCOUNT //Enderço da conta que executará as transações.
const difference = process.env.PRICE_DIFFERENCE
const units = process.env.UNITS // Usado para exibição/relatório de preços
const gas = process.env.GAS_LIMIT // Limite de gás a ser pago
const estimatedGasCost = process.env.GAS_PRICE // Estimativa de Gas: 0.008453220000006144 ETH + ~10%

let uPair, sPair, amount
let isExecuting = false


//Função assincrona principal, monitora os eventos de troca.
const main = async () => {

    //Obtem o endereço dos contratos dos tokens e a instancia do objeto de cada token que foi criado.
    const { token0Contract, token1Contract, token0, token1 } = await getTokenAndContract(arbFor, arbAgainst)

    //Obtem o endereço do contrato do pair dos tokens na Uniswap.
    uPair = await getPairContract(uFactory, token0.address, token1.address)

    //Obtem o endereço do contrato do pair dos tokens na Sushiswap.
    sPair = await getPairContract(sFactory, token0.address, token1.address)

    //Imprime no console os endereços dos pares obtidos.
    console.log(`Pair Token Uniswap: ${uPair._address}`)
    console.log(`Pair Token SushiSwap: ${sPair._address}`)

```

```

    //Função assincrona que monitara quando a ocorrencia de uma trasação envolvendo o contrato do par na Uniswap.
    uPair.events.Swap({}, async () => {
        if (!isExecuting) {
            isExecuting = true

            //Obtem a diferença de preço dos ativos entre as exchanges Uniswap e Sushiswap.
            const priceDifference = await checkPrice('Uniswap', token0, token1)

            //Determina a direção da arbitragem e se é uma arbitragem economicamente viável.
            const routerPath = await determineDirection(priceDifference)

            //Se se não é uma arbitragem viável, imprime no cosole e sai da função
            if (!routerPath) {
                console.log(`Nenhuma arbitragem disponivel\n`)
                console.log(`--------------------------------------\n`)
                isExecuting = false
                return
            }

            //Se for uma arbitragem viável, verifica qual será o retorno esperado
            const isProfitable = await determineProfitability(routerPath, token0Contract, token0, token1)

            if (!isProfitable) {
                console.log(`Nenhuma arbitragem disponivel\n`)
                console.log(`--------------------------------------\n`)
                isExecuting = false
                return
            }

            //Se o retorno esperado estiver dentro das condições estabelecidas, executa as transações.
            const receipt = await executeTrade(routerPath, token0Contract, token1Contract)

            isExecuting = false
        }
    })

```

```
106  //Verirfica o relação de preço entre os token em uma exchange.
107  const checkPrice = async (exchange, token0, token1) => {
108      isExecuting = true
109
110      console.log(`Troca Iniciada na ${exchange}, Verificando preço...\n`)
111
112      const currentBlock = await web3.eth.getBlockNumber()
113
114      const uPrice = await calculatePrice(uPair, "Uniswap")
115      const sPrice = await calculatePrice(sPair, "Sushiswap")
116
117      const uFPrice = Number(uPrice).toFixed(units)
118      const sFPrice = Number(sPrice).toFixed(units)
119
120      console.log("uFPrice",uFPrice)
121      console.log("sFPrice",sFPrice)
122
123      const priceDifference = calculateDifference(uFPrice, sFPrice)
124
125      console.log(`Bloco atual: ${currentBlock}`)
126      console.log(`----------------------------------------`)
127      console.log(`UNISWAP   | ${token1.symbol}/${token0.symbol}\t | ${uFPrice}`)
128      console.log(`SUSHISWAP | ${token1.symbol}/${token0.symbol}\t | ${sFPrice}\n`)
129      console.log(`Diferença Percentual: ${priceDifference}%\n`)
130
131      return priceDifference
132  }
133
```

```
134  const determineDirection = async (priceDifference) => {
135      console.log(`Determinando a direção...\n`)
136
137      if (priceDifference >= difference) {
138
139          console.log(`Direção da Arbitragem:\n`)
140          console.log(`Comprar\t -->\t Uniswap`)
141          console.log(`Vender\t -->\t Sushiswap\n`)
142          return [uRouter, sRouter]
143
144      } else if (priceDifference <= -(difference)) {
145
146          console.log(`Direção da Arbitragem:\n`)
147          console.log(`Comprar\t -->\t Sushiswap`)
148          console.log(`Vender\t -->\t Uniswap\n`)
149          return [sRouter, uRouter]
150
151      } else {
152          return null
153      }
154  }
155
156
157  const determineProfitability = async (_routerPath, _token0Contract, _token0, _token1) => {
158      console.log(`Determinando a Rentabilidade...\n`)
159
160      // É aqui que você pode personalizar suas condições sobre se uma negociação lucrativa é possível.
161      // Este é um exemplo básico de negociação WETH/USDT...
162
163      let reserves, exchangeToBuy, exchangeToSell
164
165      if (_routerPath[0]._address == uRouter._address) {
166          reserves = await getReserves(sPair)
167          exchangeToBuy = 'Uniswap'
168          exchangeToSell = 'Sushiswap'
169      } else {
170          reserves = await getReserves(uPair)
171          exchangeToBuy = 'Sushiswap'
172          exchangeToSell = 'Uniswap'
173      }
174
```

```
175
176     try {
177
178         //Retorna o valor mínimo do ativo de entrada necessário para comprar o valor do ativo de saída fornecido (contabilizando as taxas) das reservas fornecidas.
179         let result = await _routerPath[0].methods.getAmountsIn(reserves[0], [_token1.address, _token0.address]).call()
180
181         const token0In = result[0] // WETH
182         const token1In = result[1] // USDT
183
184
185         //Dado um valor de ativo de entrada , retorna o valor máximo de saída do outro ativo (contabilizando taxas) com reservas fornecidas.
186         result = await _routerPath[1].methods.getAmountsOut(token1In, [_token0.address, _token1.address]).call()
187
188         console.log(`Quantidade estimada de WETH necessária para comprar USDT suficiente em ${exchangeToBuy}\t| ${web3.utils.fromWei(token0In, 'ether')}`)
189         console.log(`Quantidade estimada de WETH retornada após a troca de USDT em ${exchangeToSell}\t| ${web3.utils.fromWei(result[1], 'ether')}\n`)
190
191         //Obtem as estimativas de compra e venda.
192         const { amountIn, amountOut } = await getEstimatedReturn(token0In, _routerPath, _token0, _token1)
193
194         //Coleta o saldo de ETH da conta.
195         let ethBalanceBefore = await web3.eth.getBalance(account)
196         ethBalanceBefore = web3.utils.fromWei(ethBalanceBefore, 'ether')
197
198         //Faz a diferença entra o saldo da conta e a estimativa de custo de gás
199         const ethBalanceAfter = ethBalanceBefore - estimatedGasCost
200
201         //Lucro entre os ativos que serão comprados e posteriomente serão vendidos.
202         const amountDifference = amountOut - amountIn
203
204         //Obtem o saldo em WETH da conta.
205         let wethBalanceBefore = await _token0Contract.methods.balanceOf(account).call()
206         wethBalanceBefore = web3.utils.fromWei(wethBalanceBefore, 'ether')
207
208         const wethBalanceAfter = amountDifference + Number(wethBalanceBefore)
209         const wethBalanceDifference = wethBalanceAfter - Number(wethBalanceBefore)
210
211         const totalGained = wethBalanceDifference - Number(estimatedGasCost)
212
213         const data = {
214             'Saldo ETH Antes': ethBalanceBefore,
215             'ETH gasto (gás)': estimatedGasCost,
216             'Saldo ETH após': ethBalanceAfter,
217             '-': {},
218             'Saldo WETH Antes': wethBalanceBefore,
219             'Saldo WETH  após': wethBalanceAfter,
220             'WETH Gained/Lost': wethBalanceDifference,
221             '-': {},
222             'Total ganho/perdido': totalGained
223         }
```

```
224
225             console.table(data)
226
227             if (amountOut < amountIn //|| totalGained <= 0
228                 ) {
229                 console.log("Arbitragem não viável, previsão de perda.\n")
230                 console.log(`----------------------------------------\n`)
231                 return false
232             }
233
234             amount = token0In
235             return true
236
237         } catch (error) {
238             console.log(error.data.stack)
239             console.log(`\nOcorreu um erro ao tentar determinar a lucratividade...\n`)
240             console.log(`Isso geralmente pode acontecer devido a um problema com reservas.\n`)
241             return false
242         }
243     }
244
245     const executeTrade = async (_routerPath, _token0Contract, _token1Contract) => {
246
247         console.log(`Tentativa de Arbitragem...\n`)
248
249         // Buscar saldo de token antescomi
250         const balanceBefore = await _token0Contract.methods.balanceOf(account).call()
251         const ethBalanceBefore = await web3.eth.getBalance(account)
252
253         console.log(`Negociação concluída:\n`)
254
255         //Chama a função que realizara a trade
256
257         // Buscar saldo de token após a negociação
258         const balanceAfter = await _token0Contract.methods.balanceOf(account).call()
259         const ethBalanceAfter = await web3.eth.getBalance(account)
260         const balanceDifference = balanceAfter - balanceBefore
261         const totalSpent = ethBalanceBefore - ethBalanceAfter
```