UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

MAURÍCIO SUGIMOTO POLLONI

KriptoWatcher: Robô de investimentos em criptomoedas utilizando algoritmo de arbitragem triangular

BAURU
2022

MAURICIO SUGIMOTO POLLONI

KriptoWatcher: Robô de investimentos em criptomoedas utilizando algoritmo de arbitragem triangular

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Orientador: Prof. Dr. Kleber Rocha de Oliveira

Bauru
2022

P777k

Polloni, Mauricio Sugimoto

KriptoWatcher : Robô de investimentos em criptomoedas utilizando algoritmo de arbitragem triangular / Mauricio Sugimoto Polloni. -- Bauru, 2022

35 p.

Trabalho de conclusão de curso (Bacharelado - Ciência da Computação) - Universidade Estadual Paulista (Unesp), Faculdade de Ciências, Bauru

Orientador: Prof. Dr. Kleber Rocha de Oliveira

1. Ciência da computação. 2. Robô de investimentos. 3. Criptomoedas. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Faculdade de Ciências, Bauru. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

Maurício Sugimoto Polloni

KriptoWatcher Robô de investimentos em criptomoedas utilizando algoritmo de arbitragem triangular

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Banca Examinadora

Prof. Dr. Kleber Rocha de Oliveira
Orientador
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Engenharia e Ciências - Câmpus de Rosana
Coordenadoria de Curso de Engenharia de Energia

Profa. Dra. Simone Domingues Prado
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Ciência da Computação

Prof. Dr. Renê Pegoraro
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Ciência da Computação

Bauru, _____ de _____ de.

Dedico este trabalho a minhã mãe e ao meu finado pai, meus guias neste mundo, que sempre me apoiaram e me acompanharam nessa jornada.

Agradecimentos

Agradeço inicialmente aos meus pais, que me apoiaram e me incentivaram desde o começo em minhas decisões

Agradeço ao meu orientador, Prof Dr. Kleber Rocha de Oliveira por me guiar e auxiliar no desenvolvimento deste trabalho do inicio ao fim.

Agradeço ao meu grande amigo Rafael Cherubin Hurdington, Bacharel em Ciências Contábeis pela FEARP, que me inspirou nas ideias iniciais e auxiliou no desenvolvimento do projeto.

Agradeço também aos meus amigos e colegas de trabalho da Intuitive Care, empresa onde trabalho, pela inspiração e apoio.

Por fim, agradeço aos meus professores e colegas da Unesp, por todo o conhecimento, experiencia e bons momentos que tive nesta grande universidade.

Resumo

Por possuírem alta volatilidade, criptomoedas, como Bitcoin, Ethereum, Dogecoin em um curto espaço de tempo e em alta frequência, tem expressiva variação de preços.
O processo de negociações, envolvendo o atos de compra ou venda destas criptomoedas, denominado *trade*, pode se mostrar ineficiente e inseguro se executado de forma manual. Neste cenário, foi produzido um robô de investimentos para executar estas negociações de forma automática, ininterrupta e que conseguisse acompanhar a volatilidade das criptomoedas. Este robô foi desenvolvido utilizando a linguagem de programação Python e alocado na plataforma de computação em nuvem AWS. Nele foi implementado o algoritmo de arbitragem triangular, com objetivo de ganho tanto na alta, quanto na baixa. Como resultado foi feito um protótipo para monitorar os preços e executar ações de *trade* através da API da corretora de investimentos em criptomoedas Binance.
Palavras-chave: Criptomoedas. Algoritmo de Arbitragem. Arbitragem Triangular.

Abstract

Because they have high volatility, cryptocurrencies such as Bitcoin, Ethereum, Dogecoin in a short time and at high frequency, have significant price variations. The negotiation process, involving the act of buying or selling these cryptocurrencies, called trade, can prove to be inefficient and insecure if performed manually. In this scenario, an investment robot was produced to execute these negotiations automatically, uninterrupted and that could monitor the volatility of cryptocurrencies. This robot was developed using the Python programming language and allocated on the AWS cloud computing platform. In it, the triangular arbitrage algorithm was implemented, with the objective of gaining both high and low. As a result, a prototype was made to monitor prices and execute trade actions through the API of the cryptocurrency investment broker Binance.
Keywords: Cryptocurrencies. Arbitrage Algorithm. Triangular Arbitration.

Lista de ilustrações

Lista de quadros

Lista de tabelas

Sumário

1 Introdução

A automação já é uma realidade nos mercados financeiros, sendo aplicada através de robôs de investimentos.

Um robô de investimento é definido como:

> Um programa de computador (software) que executa uma Estratégia de Investimento Automatizada. O intuito de um robô investidor é executar, com total fidelidade, a Estratégia de Investimento para qual ele foi programado. (SMARTTBOT, 2021)

Segundo Paraná (2018), em uma publicação do Instituto de Economia e Pesquisa Aplicada (IPEA), estima-se que nos Estados Unidos, mais de 50% das transações em mercado foram executadas por robôs de investimentos. No cenário nacional, esses robôs representam uma modalidade que vem adquirindo destaque, pois apresentam uma proposta de custo reduzido, disponibilizando ao pequeno investidor uma visão bem estruturada de seus investimentos.

Esses robôs de investimentos também podem ser aplicados no mercado de cripto ativos, que são ativos digitais, ou seja, podem ser negociados entre pessoas de forma livre, sem a intermediação de alguma entidade central, como as bolsas de valores. Dessa forma, não há lei que estabeleça ou assegure um valor de mercado, e sua cotação é determinada apenas pela oferta e demanda.

De acordo com Brito (2021), criptomoedas são uma categoria de cripto ativos que surgiu no ano de 2008, com a criação do Bitcoin por Satoshi Nakamoto. Elas funcionam de forma descentralizada, não possuem um lastro oficial, e também não são atreladas a nenhum tipo de papel moeda ou outro ativo.

Para definir o valor de uma criptomoeda, de acordo com a Coinext (2020) o valor de cada criptomoeda é conhecido como capitalização de mercado. A cotação de cada criptomoeda é estabelecida unicamente pela oferta e demanda.

Ao longo do tempo, com a popularidade crescente do Bitcoin, outras criptomoedas foram surgindo, como é o caso do , Apecoin, Ripple e Tron. Até algumas corretoras passaram a desenvolver suas próprias criptomoedas, como é o caso da Binance Coin, criptomoeda desenvolvida pela Binance, representada pelo simbolo BNB.

As corretoras de cripto ativos atuam unicamente como intermediárias entre usuários durante o ato de compra ou venda. Devido a negociação ocorrer de forma simultânea em diferentes países e moedas, é possível existir diferenças nos preços entre cada mercado.

Algumas destas corretoras, como a Binance e a Coinext, disponibilizam uma série de rotinas e recursos para acesso de um determinado aplicativo conhecida como *Application Programming Interface* (API), para que seus usuários possam acessar seus sistemas de forma personalizada. Utilizando os recursos desta API, é possível a criação de um robô de investimento para criptomoedas.

Diferente do mercado de ativos comuns, como ações, que esta ativo apenas em

horário comercial, o mercado de criptomoedas fica ativo constantemente, 24 horas por dia, sem pausa. A fim de acompanhar o mercado, o robô de investimentos necessita ser executado constantemente. Tecnologias de computação em nuvem, como a Amazon Web Services (AWS) e a Microsoft Azure, possuem recursos que possibilitam a execução constante desses robôs, sem a necessidade de se preocupar com configurações relacionadas a infraestrutura. A AWS, escolhida para este projeto, possui recursos, como o Lambda e o S3, que possibilitam, respectivamente, executar o robô e registar relatórios de suas negociações em arquivos.

1.1 Justificativa

Automatizar processos reduz parte dos problemas ligados à limitação humana. Um sistema automatizado pode ser executado constantemente, sem restrições de tempo, podendo acompanhar e analisar variações de dados de forma mais rápida, precisa e segura, possuindo menos riscos de falha.

O uso de “robôs” no setor de investimentos já são usados amplamente, principalmente nos processos relacionados a fundos de investimentos. De acordo com um estudo da GLOBAL (2017), a indústria de robôs de investimentos nos Estados Unidos deve chegar a US$450 bilhões ao final de 2021.

Utilizando conceitos de *webService* é possível desenvolver uma aplicação em nuvem, de forma com que esta seja executada periodicamente, sem a necessidade de ação manual. Com o mercado de cripto ativos em constante crescimento, aplicar este sistema nesse mercado pode ser uma forma de reduzir riscos e aumentar ganhos.

1.2 Objetivos

Desenvolver um robô de investimentos com foco para criptomoedas, atuando na corretora de cripto ativos Binance, este robô deve ser capaz de atuar 24 horas por dia, nos 7 dias da semana, vigiando o mercado em tempo real, e utilizando o algorítimo de arbitragem como estrategia para realizar as operações de negociação.

1.3 Organização da Monografia

Este trabalho foi estruturado em 5 capítulos. No capítulo 1 é tratada a introdução, contextualização do projeto, justificativa e objetivos. No capítulo 2, a fundamentação teórica é apresentada, onde é abordado os conceitos iniciais sobre criptomoedas e algoritmo de arbitragem triangular. Já no capítulo 3, é apresentado todo o conjunto de ferramentas utilizadas no desenvolvimento do projeto. Em sequência, o capítulo 4 aborda o desenvolvimento do projeto, demonstrando como as ferramentas citadas no capítulo anterior foram utilizadas.

Por fim, o capítulo 5 se refere a conclusão, mostrando os resultados finais do projeto e possíveis trabalhos futuros.

2 Fundamentação Teórica

2.1 Cripto ativos

De acordo com Brito (2021):

> Cripto ativos são ativos financeiros digitais, protegidos por um código criptografado, que geram uma representação digital dos valores transacionados. Os cripto ativos são a classe que engloba tudo que diz respeito ao universo cripto, como as criptomoedas, os *tokens*, *stablecoins*, protocolos de consenso, *blockchains* e plataformas de *blockchains*.

2.1.1 Criptomoeda

Criptomoedas, como por exemplo Bitcoin criado em 2008 por Satoshi Nakamoto, são uma categoria de cripto ativos. Este ativo foi criado a partir do conceito de moeda digital decentralizada, que não possui lastro oficial e que não esteja representada de forma física (papel moeda). As criptomoedas são uma opção em relação às moedas fiduciárias, como Euro, Dólar ou Real. (BRITO, 2021)

Do ponto de vista juridico/financeiro, de acordo com Stella (2017):

> Criptomoeda, ou moeda criptografada, é um ativo digital denominado na própria unidade de conta que é emitido e transacionado de modo descentralizado, independente de registro ou validação por parte de intermediários centrais, com validade e integridade de dados assegurada por tecnologia criptográfica e de consenso em rede. Trata-se de instrumentos desenhados para viabilizar transferências de valores em rede de maneira segura e independente de um sistema de intermediação financeira

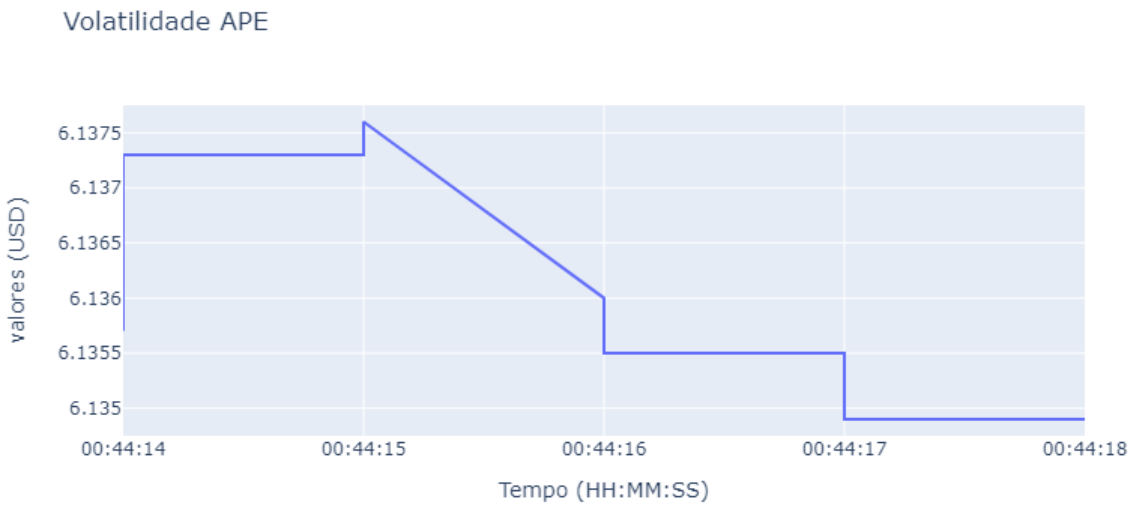
Neste projeto, foram utilizadas nos testes as criptomoedas Tron e Ripple, representadas respectivamente por TRX e XRP. Estas criptomoedas foram escolhidas devido ao seu baixo preço, como informado na Figura 5, e a sua alta volatilidade, cujo conceito é explicado na Seção 2.1.2.

2.1.2 Volatilidade

Volatilidade é o termo que descreve o quão rápido e o quanto varia o preço de um ativo. Este índice é calculado utilizando o desvios padrão no retorno anual de um ativo durante um determinado período de tempo. Por ser um conceito que analisa a velocidade, frequência e intensidade na mudanças de preços, a volatilidade é geralmente usada como uma métrica para se analisar o risco de se investir em determinado ativo. (BINANCE, 2021)

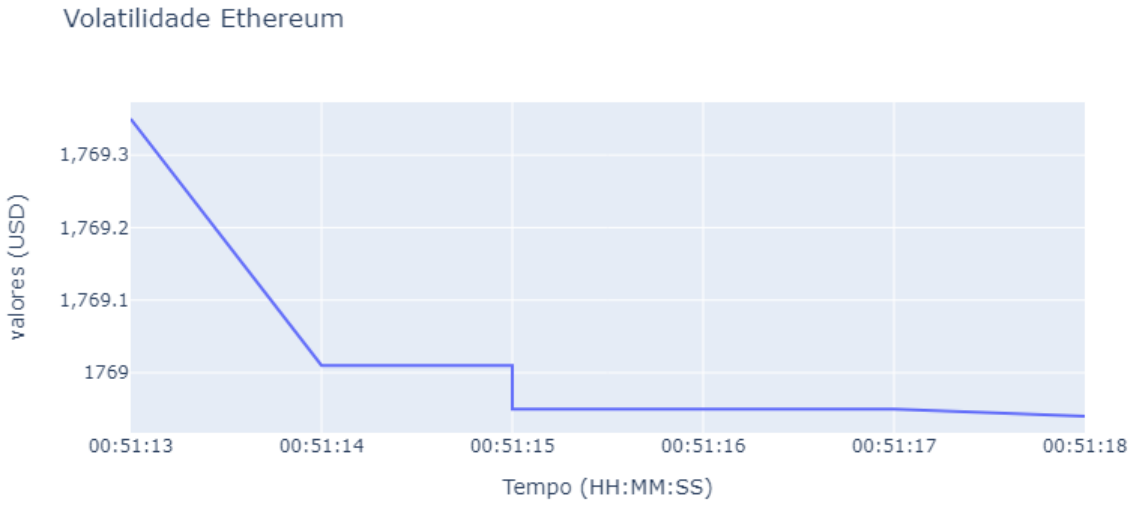
Diferente do mercado de ações tradicional, o mercado de cripto ativos se caracteriza por, de maneira geral, ter alta volatilidade, sendo possível observar grandes variações de volatilidade ao se comparar diferentes criptomoedas, por exemplo Ethereum, demonstrado na Figura 2, se mostra com uma volatilidade menor se comparadas a moedas recentes ou de menor valor, como a Apecoin, conforme foi apresentado na Figura 1.

Figura 1 – Volatilidade da criptomoeda Apecoin em cinco segundos



Elaborado pelo autor

Figura 2 – Volatilidade da criptomoeda Ethereum em cinco segundos



Elaborado pelo autor

2.1.3 *Stablecoins*

Para evitar grandes níveis de volatilidade nas criptomoedas, foram criadas as *stablecoin*. Estes ativos nada mais são do que cripto ativos associados a um ativo estável, moeda fiduciária, como por exemplo o Dólar, Euro, ou um metal precioso. No decorrer do projeto, serão apresentadas algumas *stablecoins*, tais como USDT (BINANCE, 2020)

2.2 *High frequency trade*

High frequency trade (negociações em alta frequência) é uma categoria de negociação algorítmica onde um grande número de ordens (compra ou venda de ativos) são enviados ao mercado em uma frequência extremamente alta. Devido a grande quantidade e frequência, o tempo de execução entre cada ordem é medida geralmente em milissegundos. (DEUTSCHE BANK, 2011)

2.3 Algorítimo de Arbitragem

Arbitragem pode ser definida como um momento no mercado financeiro onde ativos semelhantes podem ser comprados e vendidos simultaneamente a preços diferentes para obter lucro. De maneira mais simplificada, o ato de compra um ativo mais baratos em um mercado e vende-lo em outro mercados por um preço mais caro, porem executando ambas ações de compra e venda com um espaço de tempo muito curto, podendo ser praticamente considerada como uma operação simultânea. (FXCM RESEARCH TEAM, 2016)

De acordo com a hipótese dos mercados tradicionais, as oportunidades de arbitragem não deveriam existir, pois em condições normais, os preços dos ativos tendem a estar em equilíbrio entre os mercados, ou seja, o mesmo ativo em diferentes mercados deveriam possuir o mesmo preço. As condições para a arbitragem surgem na prática devido a ineficiências da comunicação entre mercados. Quando ocorrem os casos de arbitragem, denominados oportunidades, as moedas podem ser precificadas incorretamente, esse acontecimento ocorre devido a atrasos na cotação de preços do mercado ou a informações assimétricas. Porém, devido às negociação de alta frequência, a ocorrência de oportunidades se tornam cada vez mais raras, pois a informatização dos processos tornaram a precificação mais eficiente, reduzindo o índice de frequência e a duração de tempo das oportunidades. (FXCM RESEARCH TEAM, 2016)

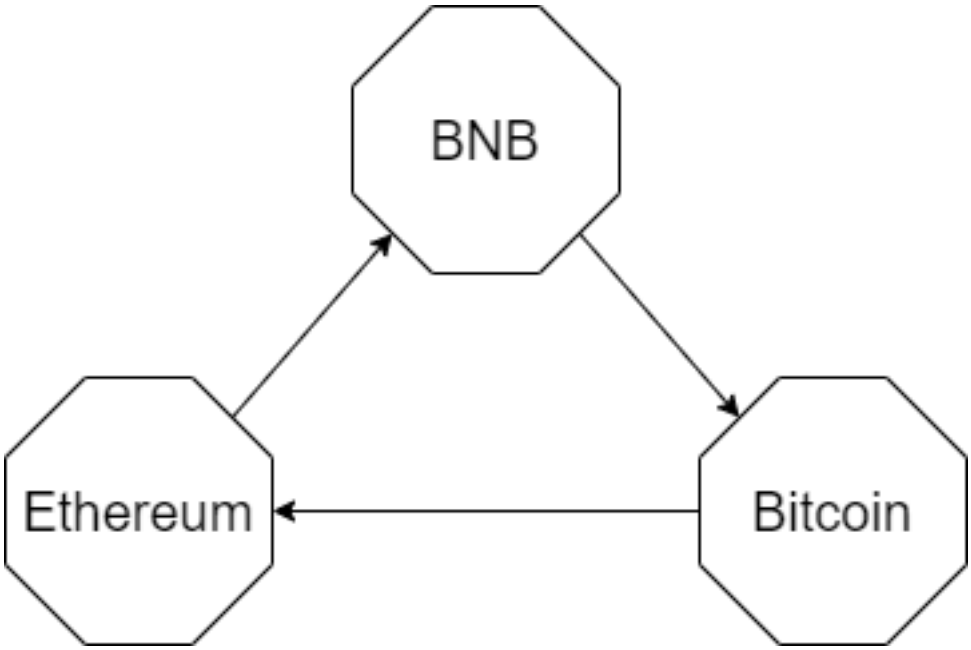
2.3.1 Arbitragem triangular

A arbitragem triangular (também conhecida como arbitragem de três pontos ou arbitragem de moeda cruzada) é uma variação da estratégia de arbitragem, podendo oferecer melhores chances. Envolve a negociação de três ou mais moedas diferentes, aumentando assim a probabilidade de que as ineficiências de comunicação do mercado apresentem oportunidades de lucros. Nesta estratégia, os *traders* procurarão situações em que uma moeda específica esteja supervalorizada em relação a uma moeda, mas subvalorizada em relação à outra. (FXCM RESEARCH TEAM, 2016)

Por exemplo, inicialmente é comprado Bitcoin com BNB, depois comprar Ethereum com Bitcoin e, finalmente, comprar de volta BNB com Ethereum. Se o valor relativo entre

Ethereum e o Bitcoin não corresponde ao valor que cada uma dessas moedas tem em BNB, existe uma oportunidade de arbitragem.

Figura 3 – Esquema do algoritmo de arbitragem triangular



Elaborado pelo autor

A arbitragem triangular representa uma das formas de arbitragem mais simples, porem, devido a alta liquidez do preço do mercado, as atualizações ocorrem em frequências extremamente altas, portanto, é necessário um conjunto de dados de frequência igualmente alta para testar oportunidades de arbitragem triangular. (FENN, 2008)

3 Ferramentas

3.1 HTTP

De acordo com MDN CONTRIBUTORS (2022):

> HTTP (Hiper Text protocol) é um protocolo que permite a obtenção de recursos, como documentos HTML. É a base de qualquer troca de dados na Web e um protocolo cliente-servidor, o que significa que as requisições são iniciadas pelo destinatário, geralmente um navegador da Web. Um documento completo é reconstruído a partir dos diferentes sub-documentos obtidos, como por exemplo texto, descrição do layout, imagens, vídeos e scripts.

3.1.1 HTTP *Request*

De acordo com MDN CONTRIBUTORS (2021) o protocolo HTTP define um conjunto de métodos de requisição responsáveis por indicar a ação a ser executada para um dado recurso.

3.1.1.1 GET

MDN CONTRIBUTORS (2021) define que o método GET solicita a representação de um recurso específico. Requisições utilizando o método GET devem retornar apenas dados.

3.1.1.2 POST

De acordo com MDN CONTRIBUTORS (2021):

> O método POST é utilizado para submeter uma entidade a um recurso específico, frequentemente causando uma mudança no estado do recurso ou efeitos colaterais no servidor. Uma solicitação POST geralmente é enviada por meio de um formulário HTML e resulta em uma alteração no servidor.

3.2 API

3.2.1 Definições

Application Programming Interface (API), é um conjunto de regras que definem como aplicativos ou dispositivos podem se conectar e se comunicar uns com os outros. Uma API de REST é uma API que se modela aos princípios de projeto do REST ou o estilo de arquitetura do *Representational State Transfer.* (IBM CLOUD EDUCATION, 2021)

De acordo com IBM CLOUD EDUCATION (2021):

> As APIs de REST se comunicam via solicitações de Protocolo de hipertexto de para executar funções padrão do banco de dados como criar, ler, atualizar e excluir registros (também conhecidos como CRUD) em um recurso. Por exemplo, uma API de REST usaria uma solicitação para recuperar um registro, uma solicitação para criar um registro, uma solicitação para atualizar um registro e uma solicitação

> para excluir um registro. Todos os métodos deste protocolo podem ser usados em chamadas da API. Uma API de REST bem projetada é semelhante a um website em execução em um navegador da web com funcionalidade deste protocolo integrada.
>
> O estado de um recurso em qualquer horário específico, ou registro de data e hora, é conhecido como representação de recursos. Estas informações podem ser entregues a um cliente em praticamente qualquer formato, incluindo JavaScript Object Notation (JSON), HTML, XLT, Python, PHP ou texto simples. O JSON é bastante usado, porque é legível tanto por humanos quanto por máquinas, além de ser uma programação de linguagem independente.
>
> Os cabeçalhos e parâmetros de solicitação também são importantes em chamadas de API de REST, porque incluem informações importantes do identificador como metadados, autorizações, identificadores de recursos uniformes (URIs), armazenamento em *cache*, *cookies* e mais. Cabeçalhos de solicitação e cabeçalhos de resposta, juntamente aos códigos de *status* do Protocolo de hipertexto convencionais, são usados dentro de APIs de REST bem projetadas.

3.2.2 Binance API

A corretora de cripto ativos disponibiliza aos seus usuários uma API *Restful*, podendo acessa-la por meio de HTTP *Request*. Através desta API é possível negociar entre diversas criptomoedas.

3.3 AWS

Escolhida para este trabalho por ser uma das principais plataformas de computação em nuvem, oferecendo serviços de execução e registro de histórico de código, além de monitoramento de recursos computacionais utilizados. A Amazon Web Services (AWS) possui servidores em diversos locais do planeta, denominados zonas. Uma das principais zonas fica no Norte da Virgínia, nos Estados Unidos. (AMAZON WEB SERVICES INC, 2022a)

3.3.1 Definições

A Amazon Web Services (AWS) é uma plataforma de computação em nuvem desenvolvida e fornecida pela Amazon. A plataforma combina os modelos de serviços *cloud* IaaS, PaaS e SaaS.

3.3.1.1 Infraestrutura como serviço (IaaS)

O Infraestrutura como serviço (IaaS) é o modelo de serviço onde os recursos de computação ficam hospedados em nuvens privadas, públicas ou hibridas.

De acordo com (ORACLE, 2022a):

> Empresas podem usar o modelo IaaS para transferir parte ou todo o uso da infraestrutura *on-premises*[1] ou colocada do *data center* para a nuvem, onde pertence e é gerenciado por um provedor de nuvem. Esses elementos de infraestrutura de baixo custo podem incluir computação, rede e *hardware* de armazenamento, bem como outros componentes e *software*.

3.3.1.2 Plataforma como serviço (PaaS)

A Plataforma como serviço (PaaS) é definida por um grupo de serviços voltados para gerenciamento e criação de aplicativos. (ORACLE, 2022b)

De acordo com IBM (2022):

> O provedor PaaS hospeda de tudo em seu data center tal como servidores, redes, armazenamento, software de sistema operacional, bancos de dados e ferramentas de desenvolvimento. Geralmente os clientes podem pagar uma taxa fixa para fornecer uma quantia especificada de recursos para um número especificado de usuários ou escolher a precificação 'pré-paga' para pagar apenas pelos recursos usados. Qualquer uma das opções permite que os clientes de PaaS desenvolvam, testem, implementem, executem, atualizem e ajustem a escala dos aplicativos de maneira mais rápida e mais barata do que poderiam se tivessem que desenvolver e gerenciar sua própria plataforma on-premises [2].

3.3.1.3 *Software* como serviço (SaaS)

O modelo de Saas é definido como um formato de entrega de *software* onde quem desenvolve, gerencia atualizações e disponibiliza o produto é o próprio provedor de computação em nuvem.

De acordo com (EQUIPE TOTVS, 2022):

> A fornecedora/desenvolvedora da aplicação o disponibiliza mediante pagamento mensal, trimestral, semestral etc., como um serviço, não como produto. Geralmente, não é preciso instalá-lo em computadores, pois funciona de maneira *online*. A empresa ou pessoa que contrata essa forma de programa, normalmente, paga pela utilização das funcionalidades das quais necessita, pois nem sempre é preciso adquirir todos os recursos do sistema.

3.3.2 Lambda

O Lambda é um serviço de computação em nuvem onde é possível executar códigos sem a necessidade de configurar servidores. Através dele é possível executar o código em uma infraestrutura de computação de alta disponibilidade sem a necessidade de gerenciar os recursos computacionais e as demais configurações para manutenção do servidor e do sistema operacional. O serviço também possui escalabilidade automática da capacidade, monitoramento de recursos computacionais utilizados e registro dos histórico de ocorrências (*log*) do código.

[1] Tipo de sistema de informação em que os dados e os processos de um negócio são armazenados e gerenciados na própria organização

[2] Tipo de sistema de informação para ERP (Enterprise Resource Planning) em que os dados e os processos de um negócio são armazenados e gerenciados na própria organização

Atravez do Lambda, é possivel executar código para diversos tipo de aplicações ou serviços de *backend*, sendo necessário apenas fornecer o código em uma das linguagens compatíveis (sendo elas Python, Java, Ruby, Node.js, Go e C#). Porém este serviço é mais indicado para processos de curtas duração, pois as funções do Lambda poder ser executadas por até 15 minutos por invocação. (AMAZON WEB SERVICES INC, 2022b)

O Lambda fornece as seguintes opções de arquiteturas de conjuntos de instruções:

- arm64: arquitetura ARM de 64 bits para o processador AWS Graviton2.
- x86_64: arquitetura x86 de 64 bits para processadores baseados em x86.

3.3.3 Amazon S3

O Amazon S3 (*Simple Storage Service*) é um serviço de armazenamento de objetos, podendo ser usado para armazenar e proteger qualquer volume de dados para uma variedade de casos de uso, como sites, aplicações móveis, *backup* e restauração, arquivamento, IoT e *big data*. Também fornece recursos de gerenciamento, sendo possível organizar e configurar o acesso aos seus dados para atender aos seus requisitos específicos de negócios, organizacionais e de compatibilidade.

No S3, todo objeto é a composição de arquivo e seus receptivos metadados. Estes objetos são armazenados em um contêiner de objetos denominado *bucket*. Para armazenar seus dados é necessário criar um *bucket*, informando seu respectivo nome e a região da AWS. Ao carregar seus dados para esse *bucket*, seus dados serão considerados como objetos do *bucket*. Cada objeto tem uma chave responsável por identificar o objeto no *bucket*.

Este serviço também possui diversos recursos para oferecer suporte para casos de uso específico. É possível utilizar versionamento dos arquivos para manter várias versões de um objeto no mesmo *bucket*, permitindo restaurar objetos excluídos anteriormente. (AMAZON WEB SERVICES INC, 2022c)

Os *buckets* e os objetos neles são privados e poderão ser acessados seguintes meios:

- Políticas do AWS IAM (*Identity and Access Management*)
- Listas de controle de acesso (ACLs)
- Credenciais de acesso de administrador

3.3.4 Amazon CloudWatch

O Amazon *CloudWatch* é um serviço de monitoramento, em tempo real, de recursos da AWS.

O *CloudWatch* é usado para rastrear e coletar métricas utilizadas medir para seus recursos e aplicativos. Estas informações ficam presentes em sua página inicial ou painéis personalizados para exibir métricas sobre seus aplicativos. Também é possível criar alarmes que monitoram as métricas e enviam notificações ou fazem alterações automaticamente nos recursos que você está monitorando quando um limite é violado. Por exemplo, monitorar o uso da CPU e as leituras e gravações de disco de suas instâncias do Amazon EC2 e, em seguida, usar esses dados para determinar como se deve executar instâncias adicionais para lidar com o aumento da carga.(AMAZON WEB SERVICES INC, 2022e)

3.3.5 Amazon EventBridge

O Amazon EventBridge é um serviço de barramento de eventos que facilita a conexão de seus aplicativos com dados de várias fontes. Este serviço fornece um fluxo de dados em tempo real de seus aplicativos e serviços da AWS. É possível configurar regras de roteamento para determinar para onde enviar seus dados para criar arquiteturas de aplicativos que reagem em tempo real a todas as suas fontes de dados. O *EventBridge* permite que você crie arquiteturas orientadas a eventos que são fracamente acopladas e distribuídas. (AMAZON WEB SERVICES INC, 2022d)

3.4 Python

Python é uma linguagem *open source* poderosa e fácil de aprender. Tem estruturas de dados de alto nível e eficientes, além de uma abordagem simples mas efetiva, de programação orientada a objetos. A elegância de sintaxe e a tipagem dinâmica de Python aliadas com sua natureza interpretativa, o fazem a linguagem ideal para programas e desenvolvimento de aplicações rápidas em diversas áreas e na maioria das plataformas. Python também é conhecida por ser uma linguagem de propósito geral, usado bastante em *data-science*, *machine learning*, desenvolvimento de *web*, desenvolvimento de aplicativos e automação de scripts. (DOWNEY, 2012; ROSSUM, 2003; PYTHON SOFTWARE FOUNDATION, 2021)

3.5 Boto3

Boto3 é um SDK (*Software Development Kit*) da AWS (*Amazon Web Services*) para *Python*, esta ferramenta permite aos desenvolvedores, através de um código em *Python*, acessar diversos produtos e recursos da AWS, como por exemplo ler ou carregar arquivos no S3, cujo conceito é explicado na Seção 3.3.3, ou executar funções no Lambda, cujo conceito é explicado na Seção 3.3.2. (AMAZON WEB SERVICES INC, 2021)

4 Desenvolvimento

4.1 Utilizando a API da Binance

Todo o projeto é baseado nas requisições disponíveis na API *Restful* da corretora *Binance*, ou seja, fornece dados em um formato padronizado baseado em requisições HTTP, podendo ser no formato GET ou POST, retornando todos as suas requisições em formato JSON. As requisições são subdivididas de acordo com a área onde atuam, podendo ter a necessidade de passar em seu cabeçalho, duas chaves de segurança, sendo possível gera-las no site da corretora.

As requisições que não necessitam das chaves de segurança acessam as informações publicas, como preço dos cripto ativos, histórico da quantidade de negociações de determinado cripto ativo, dados do *status* dos servidores, entre outros, geralmente são executadas no formato GET, podendo ter exceções. Já requisições relacionadas diretamente com sua conta necessitam das chaves, como por exemplo acessar dados da conta, limites e histórico de negociações executadas pelo usuário, e principalmente as requisições responsáveis pelas negociações de compra e venda.

O *endpoint* base é https://api.binance.com, se houver problemas de desempenho com o *endpoint* acima, a Binance API também disponibiliza os seguintes *endpoints*: (BINANCE, 2022)

- https://api1.binance.com;
- https://api2.binance.com;
- https://api3.binance.com.

4.1.1 Chaves de Segurança

Para efetuar qualquer requisição que esteja relacionada a conta, como a requisição citada na Seção 4.1.4, é necessário colocar no cabeçalho da requisição uma chave de acesso, denominada *API Key*, e no parâmetro das requisições, uma assinatura criptografada no padrão SHA-256, composta por todos os outros parâmetros em junção com uma chave de segurança, denominada *Security Key*. Para gerar essas chaves , é necessário acessar sua conta na corretora Binance e solicitar a criação. (BINANCE, 2022)

4.1.2 Pesos e limites

A fim de evitar diversos problemas relacionados ao “bombardeio de requisições”, a *Binance* inseriu um limite requisições para cada IP ou conta, esta medição é feita através de um peso dado a cada requisição, ou seja, cada requisição possui no mínimo 1 de peso.

Existem dois limites de requisições:

- Limite de peso, cada IP pode executar até 1200 de peso por minuto, ou seja, é possível executar até 1200 requisições de peso 1 em 1 minuto.
- Limite de ordens de negociação, sendo que cada conta pode executar até 50 ordens (compra ou venda de cripto ativos) num tempo 10 segundos. (BINANCE, 2022)

4.1.3 *Symbol Price Ticker*

Para capturar os preços de uma determinada criptomoeda na corretora Binance, é necessário efetuar uma requisição do tipo *Get,* utilizando um dos *endpoints* informados na Seção 4.1 junto a URL especifica, como informada no Quadro 1.

Esta requisição, retorna os preços da corretora, tendo duas variações: para mostrar apenas um mercado é necessário informar seu simbolo na URL da requisição, como referenciado no Código 4.1. Caso este dado seja omitido, a requisição retornara todas os valores disponíveis da corretora, como representado no Código 4.2. Todas as respostas das funções de requisição passam por uma função de formatação, referenciada no Código 4.3, onde é adicionado também o tempo de resposta da requisição, o *status* da requisição, e o peso, citado também na Seção 4.1.2.

Quadro 1 – Symbol Price Ticker

Tipo	GET
URL	/api/v3/ticker/price
Exemplo de resposta	{ "symbol": "LTCBTC", "price": "4.00000200" }

Fonte: BINANCE (2022)

Código 4.1 – Função de resgate do preço de determinado criptoativo

```
1 def last_price_symbol(end_point, symbol):
2     get_params = {"symbol": symbol}
3     response = req.get(end_point + Path.symbol_price_ticker.value
          , params=get_params)
4     return parse_response(response)
```

Código 4.2 – Função de resgate do preço de todos os criptoativos da corretora

```
1  def last_price(end_point):
2     response = req.get(end_point + Path.symbol_price_ticker.value
          )
3     return parse_response(response)
```

Código 4.3 – Função que padroniza resposta de requisição

```
1 def parse_response(response):
2     time = "{:.3f}".format(response.elapsed.total_seconds())
3     info = {"Response": response.text,
4             "status": status.verifica_status(response),
5             "Weight": response.headers['x-mbx-used-weight-1m'],
6             "Time": time}
7     return info
```

4.1.4 *New Order*

Requisição responsável por executar um pedido de compra ou venda. Nesta requisição, referenciada no Quadro 2, é necessário informar as chaves de segurança da API no cabeçalho da requisição, seu corpo é composto pelos parâmetros referenciados no Quadro 2, na linha Corpo.

Quadro 2 – New Order

Tipo	POST
URL	/api/v3/order

Corpo	• **timestamp**
	  ◦ Tempo atual em milissegundos
	• **symbol**
	  ◦ Símbolo que faz referência ao mercado, por exemplo BTCUSDT (Bitcoin para dólar)
	• **side**
	  ◦ Símbolo referente a compra ou venda
	• **quantity**
	  ◦ Quantidade, em criptoativos, a ser negociada
	• **type**
	  ◦ Referente ao tipo de ordem criada, porém no projeto será usada apenas o tipo MARKET (ordem de compra ou venda executada no momento da criação do pedido)

Exemplo de resposta	{ "symbol": "BTCUSDT", "orderId": 28, "orderListId": -1, "clientOrderId": "6gCrw2kRUAF9CvJDGP16IP", "transactTime": 1507725176595}
	• **symbol**
	  ◦ Símbolo que faz referência ao mercado onde foi feito a negociação
	• **orderId**
	  ◦ Identificador da ordem destro da conta
	• **orderListId**
	  ◦ Identificador de execução imediata
	• **clientOrderId**
	  ◦ Identificador da ordem dentro do servidor da Binance
	• **transactTime**
	  ◦ Momento em que a ordem foi criada (em milisegundos)

Fonte: BINANCE (2022)

Código 4.4 – Exemplo de função de compra ou venda de determinado criptoativo

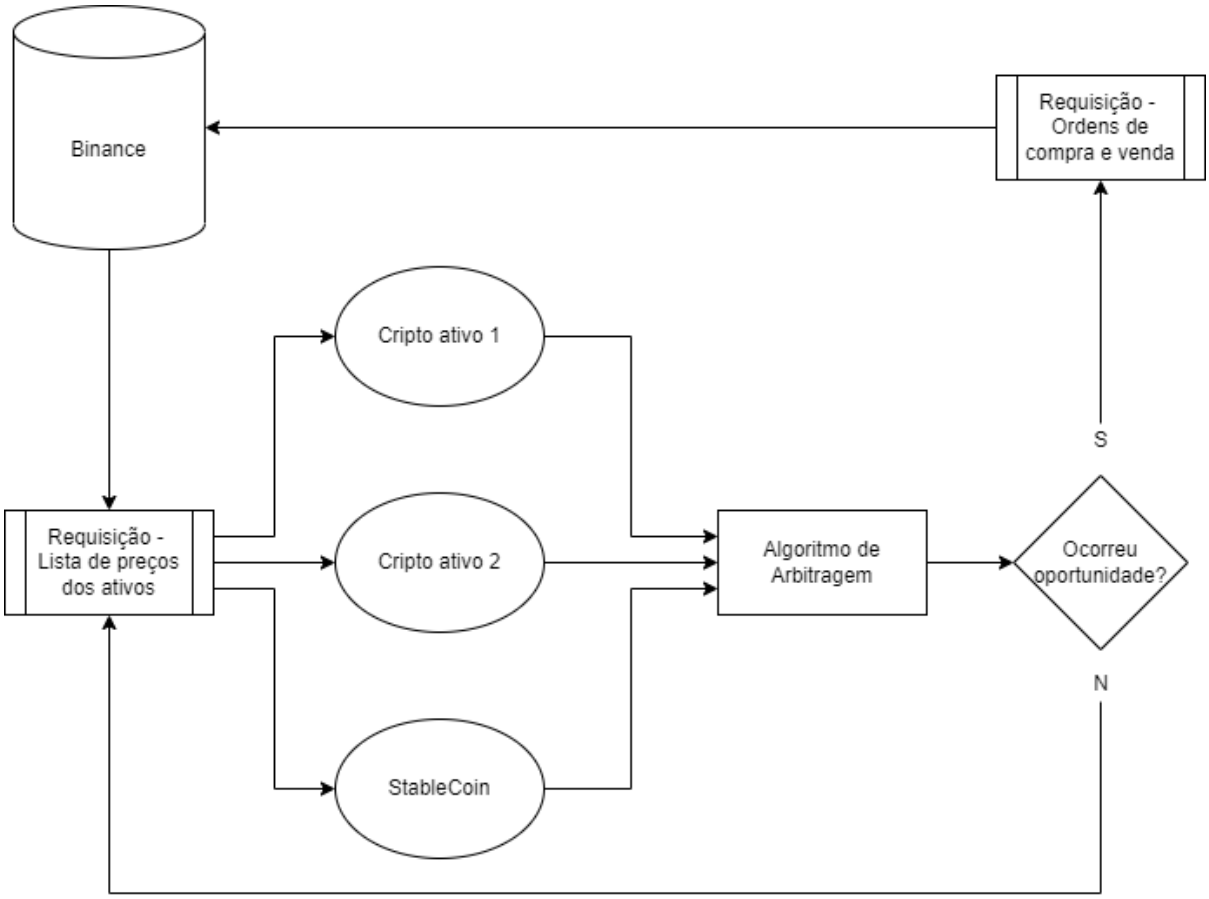
```
1 def new_order(end_point, symbol, side, type_order, value):
2     server_time_int = get_timestamp(end_point)
3     params = {
4         "timestamp": server_time_int,
5         "symbol": symbol,
6         "side": side,
7         "type": type_order,
8     }
9     if side == 'BUY':
10        params.update("quoteOrderQty", value)
```

```
11      else:
12          params.update("quantity", value)
13
14      response = req.post(end_point + Path.new_order.value,
15                          params=get_base_params(params, get_hash_
                                sha256(params, secret)),
16                          headers=get_headers_API(apikey))
17
18      return parse_response(response)
```

4.2 Protótipo

Conforme esquematizado na Figura 4, o protótipo é composto por três etapas, sendo a primeira o processo de captura dos preços das criptomoedas, tanto o valor entre eles (por exemplo, o preço do Ethereum sendo medido em Bitcoin) quanto seus preços em uma *stablecoin* especifica (por exemplo, o preço do Bitcoin em USDT, *stablecoin* que corresponde ao Dolar); a segunda se refere a execução do algoritmo de arbitragem triangular, onde ocorrera a análise dos preços capturados na etapa anterior, caso esta analise resulte em uma janela de oportunidade de lucro, o protótipo executará a terceira etapa, sendo a requisição de criação das ordens de compra e venda definidos pelo algoritmo de arbitragem. Se a analise dos preços não retornar nenhuma janela de oportunidade, o protótipo retornará para a etapa 1, reiniciando o processo.

Figura 4 – Esquema do robô de investimentos



Elaborado pelo autor

4.3 Implementação na AWS

4.3.1 S3

No momento que alguma ordem de compra ou venda for executada com sucesso, o resultado das requisições será salvo em um arquivo JSON, nome do arquivo é um *timestamp* para não ocorrer duplicidade de dados. Todos os arquivos gerados no mesmo dia serão salvos em uma pasta cujo nome está no formato da data (AAAA-MM-DD).

4.3.2 EventBrigde

Este serviço possui o recurso de criar regras para executar outros serviços a cada determinado período de tempo, de forma permanente. Foi criado uma regra para executar o protótipo a cada 1 minuto. Ao final de cada execução do protótipo, o serviço o executará no minuto seguinte.

4.3.3 Lambda

Considerado o núcleo estrutural do protótipo, o Lambda é o serviço onde o código do projeto será alocado e executado. Recebendo o código em Python, ele irá executar as funções referentes as requisições para a API da Binance, avaliando suas respostas e *status* de retorno. A função do Lambda será reexecutada pela regra do serviço EventBridge.

4.4 Tratativa de Erros

Sendo uma API *Restful*, a Binance API possui diversos *status* de retorno de erro referente as requisições, como citado na Tabela 1.

Tabela 1 – Principais erros das requisições para a API da Binance

Código do *Status*	**Descrição**
403	*WAF Limit (Web Application Firewall)*
429	*breaking a request rate limit*
418	*IP has been auto-banned for continuing to send requests after receiving*

Fonte: BINANCE (2022)

O Protótipo esta preparado para desabilitar a regra responsável por reexecutar a cada 1 minuto caso qualquer requisição retorne um *status* diferente de 200 (sucesso), impedindo um bloqueio de IP por parte da API da Binance.

4.5 Extração e análise de dados

Para efetuar a análise do protótipo, este foi executado de forma continua por aproximadamente 18 horas, extraindo a cada 1 minuto os preços das criptomoedas TRX e XRP, representados na Figura 5, analisando os valores entre si e em relação a *stablecoin* USDT.

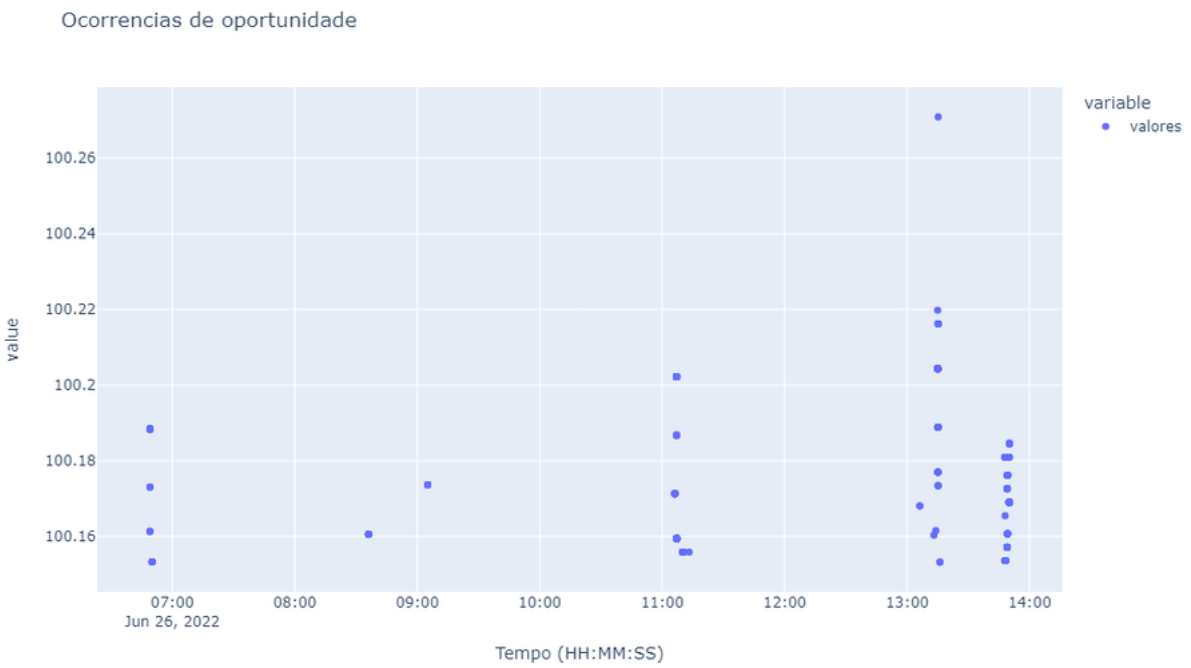
A execução do protótipo foi feita através da AWS, utilizando o recurso Lambda para execução, S3 para registrar o resultado de cada requisição, e o CloudWatch para monitoramento de recursos e histórico do código. Diferente de executar em uma maquina local, requisições feitas via AWS Lambda podem possuir melhor velocidade de resposta, pois o servidor se localiza no Norte da Virgínia, nos Estados Unidos, possuindo melhor infraestrutura de rede, de acordo com os referencias e descrições da AWS.

Figura 5 – Variação do preço das criptomoedas TRX e XRP para USDT num período de 18 horas aproximadamente

Elaborado pelo autor

Aplicando o algoritmo de arbitragem triangular, citado na Seção 2.3.1, sobre os dados, usando um valor inicial fictício de 100 USDT (equivale a 100 USD), 0.1% do valor negociado, em cada operação, como taxa, é possível observar que ocorre a existência de Oportunidades de aplicação do algoritmo em alguns momentos dentro de um espaço de tempo muito pequeno, como é demonstrado na Figura 6, onde cada ponto representa uma ocorrência de oportunidade.

Figura 6 – Número de ocorrências de Oportunidades encontradas após a analise dos dados



Elaborado pelo autor

Durante este processo, as ordens de *trade* foram apenas simuladas, aplicando as taxas de maneira fictícia, não ocorrendo negociações reais. O processo foi executado desta forma devido a necessidade de possuir a quantia equivalente ao valor fictício utilizado na aplicação.

5 Conclusão

Devido a alta volatilidade, baixa ocorrência de oportunidades e pequena margem de lucro por execução da ordem de negociações do algoritmo, a forma manual de negociação de criptomoedas se mostra ineficaz e possivelmente impossível de ser executada com precisão e segurança.

A execução do protótipo de investimento se mostra promissora. Mesmo efetuando uma análise de mercado em um espaço de tempo curto de 18h, o numero de ocorrências de oportunidades pode se mostrar significativo. No cenário apresentado na Figura 6, a margem de lucro seria de aproximadamente 0,12% por execução do algoritmo, sendo uma margem de lucro abaixo dos parâmetros de uma *trade* convencional. Porem, se aplicada constantemente, interruptamente pelo mesmo período de tempo, há possibilidade de lucro relevante.

5.1 Trabalhos futuros

A ideia inicial para trabalhos futuros é implementar melhorias no algoritmo de arbitragem triangular, a fim de resultar em melhor eficiência do robô.

Como o protótipo já possui a base do algoritmo de arbitragem triangular, propoem-se determinar os possíveis melhores horários de execução, pois a quantidade de negociações pode variar de acordo com o horário, e como citado na Seção 2.3, o volume e a frequência de negociações podem influenciar a quantidade de ocorrências de oportunidade.

Propoem-se também determinar melhores escolhas de criptomoedas para aplicar no algoritmo, analisando seu preço e sua volatilidade, tanto entre si quanto em relação a uma determinada *stablecoin,* a fim de determinar combinações mais eficientes para o Algoritmo de arbitragem triangular.

Referências

AMAZON WEB SERVICES INC. **Boto3 documentation**. 2021. Disponível em: https://boto3.amazonaws.com/v1/documentation/api/latest/index.html. Acesso em: 10/06/2022.

AMAZON WEB SERVICES INC. **Infraestrutura global da AWS**. 2022a. Disponível em: https://aws.amazon.com/pt/about-aws/global-infrastructure/?hp=tile&tile=map. Acesso em: 10/08/2022.

AMAZON WEB SERVICES INC. **O que é o AWS Lambda?** 2022b. Disponível em: https://docs.aws.amazon.com/pt_br/lambda/latest/dg/welcome.html. Acesso em: 06/05/2022.

AMAZON WEB SERVICES INC. **O que é o Amazon S3?** 2022c. Disponível em: https://docs.aws.amazon.com/pt_br/AmazonS3/latest/userguide/Welcome.html. Acesso em: 06/05/2022.

AMAZON WEB SERVICES INC. **What Is Amazon EventBridge?** 2022d. Disponível em: https://docs.aws.amazon.com/pt_br/eventbridge/latest/userguide/eb-what-is.html. Acesso em: 07/05/2022.

AMAZON WEB SERVICES INC. **What is Amazon CloudWatch?** 2022e. Disponível em: https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/WhatIsCloudWatch.html. Acesso em: 07/05/2022.

BINANCE. **General API Information**. 2022. Disponível em: https://binance-docs.github.io/apidocs/spot/en/#api-library. Acesso em: 07/05/2022.

BINANCE. **O que é uma Stablecoin?** 2020. Disponível em: https://academy.binance.com/pt/articles/what-are-stablecoins. Acesso em: 24/05/2022.

BINANCE. **Volatility**. 2021. Disponível em: https://academy.binance.com/pt/glossary/volatility. Acesso em: 01/06/2022.

BRITO, L. **Criptoativos: O que é e qual a sua aplicação**. 2021. Disponível em: https://coinext.com.br/blog/o-que-e-criptoativo. Acesso em: 10/11/2021.

COINEXT. **O que é criptomoeda**. 2020. Disponível em: https://coinext.com.br/blog/o-que-e-criptomoeda. Acesso em: 02/11/2021.

DEUTSCHE BANK. **High frequency trading**. 2011. Disponível em: https://c.mql5.com/forextsd/forum/168/high-frequency_trading_-_better_than_its_reputation.pdf. Acesso em: 07/05/2022.

DOWNEY, A. **Think python**. [S.l.]: O'Reilly Media, Inc., 2012.

EQUIPE TOTVS. **Software as a Service (SaaS): como funciona, vantagens e exemplos**. 2022. Disponível em: https://www.totvs.com/blog/negocios/software-as-a-service-saas/. Acesso em: 03/07/2022.

FENN, D. J.; HOWISON, S. D.; MCDONALD, M.; WILLIAMS, S.; JOHNSON, N. F. **THE MIRAGE OF TRIANGULAR ARBITRAGE IN THE SPOT FOREIGN EXCHANGE MARKET**. 2008. Disponível em: https://arxiv.org/pdf/0812.0913. Acesso em: 07/05/2022.

FXCM RESEARCH TEAM. **Triangular Arbitrage**. 2016. Disponível em: https://www.fxcm.com/markets/insights/triangular-arbitrage/. Acesso em: 22/05/2022.

GLOBAL, S. **U.S. Digital Adviser Forecast: AUM To Surpass $450B By 2021**. 2017. Disponível em: https://www.spglobal.com/en/research-insights/articles/US-Digital-Adviser-Forecast-AUM-To-Surpass-450B-By-2021. Acesso em: 05/11/2021.

IBM. **PaaS (Platform-as-a-Service)**. 2022. Disponível em: https://www.ibm.com/br-pt/cloud/learn/paas. Acesso em: 03/07/2022.

IBM CLOUD EDUCATION. **APIs de REST**. 2021. Disponível em: https://www.ibm.com/br-pt/cloud/learn/rest-apis. Acesso em: 04/06/2022.

MDN CONTRIBUTORS. **Métodos de requisição HTTP**. 2021. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Methods. Acesso em: 28/05/2022.

MDN CONTRIBUTORS. **Uma visão geral do HTTP**. 2022. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Overview. Acesso em: 28/05/2022.

ORACLE. **O que é o IaaS - Infraestrutura como Serviço?** 2022a. Disponível em: OqueéoIaaS-InfraestruturacomoServiço? Acesso em: 03/07/2022.

ORACLE. **O que é plataforma como serviço (PaaS)?** 2022b. Disponível em: https://www.oracle.com/br/cloud/what-is-paas/. Acesso em: 03/07/2022.

PARANÁ, E. **A DIGITALIZAÇÃO DO MERCADO DE CAPITAIS NO BRASIL: TENDÊNCIAS RECENTES**. 2018. Disponível em: http://repositorio.ipea.gov.br/bitstream/11058/8280/1/TD_2370.PDF. Acesso em: 20/11/2021.

PYTHON SOFTWARE FOUNDATION. **O tutorial de Python**. 2021. Disponível em: https://docs.python.org/pt-br/3/tutorial/. Acesso em: 11/07/2022.

ROSSUM, G. V. **An introduction to Python**. [S.l.]: Bristol: Network Theory Ltd, 2003.

SMARTTBOT. **O que é um robô investidor?** 2021. Disponível em: https://ajuda.smarttbot.com/hc/pt-br/articles/360007603814-O-que-é-um-robô-investidor-. Acesso em: 10/11/2021.

STELLA, J. C. **Moedas Virtuais no Brasil: como enquadrar as criptomoedas**. 2017. Revista da Procuradoria-Geral do Banco Central. Disponível em: https://revistapgbc.bcb.gov.br/index.php/revista/issue/view/26/A9%20V.11%20-%20N.2. Acesso em: 24/05/2022.