PUCRS

ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

EDUARDO VIANA PEREIRA

DESENVOLVIMENTO DE UMA PLATAFORMA DE SIMULAÇÃO PARA AVALIAÇÃO DE ESTRATÉGIAS DE MERCADO BASEADA NO PUMA TRADING SYSTEM

Porto Alegre
2024

PÓS-GRADUAÇÃO - *STRICTO SENSU*

Pontifícia Universidade Católica
do Rio Grande do Sul

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

DESENVOLVIMENTO DE UMA PLATAFORMA DE SIMULAÇÃO PARA AVALIAÇÃO DE ESTRATÉGIAS DE MERCADO BASEADA NO PUMA TRADING SYSTEM

EDUARDO VIANA PEREIRA

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Ciência da Computação na Pontifícia Universidade Católica do Rio Grande do Sul.

Orientador: Prof. Dr. Fernando Gehm Moraes
Co-Orientador: Prof. Dr. Rafael Fraga Garibotti

Porto Alegre
2024

FICHA CATALOGRÁFICA

EDUARDO VIANA PEREIRA

DESENVOLVIMENTO DE UMA PLATAFORMA DE SIMULAÇÃO PARA AVALIAÇÃO DE ESTRATÉGIAS DE MERCADO BASEADA NO PUMA TRADING SYSTEM

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação, Escola Politécnica da Pontifícia Universidade Católica do Rio Grande do Sul.

Aprovado(a) em 5 de March de 2024.

BANCA EXAMINADORA:

Prof. Dr. Vanderlei Bonato (USP)

Prof. Dr.César Augusto Missio Marcon (PPGCC/PUCRS)

Prof. Dr. Rafael Fraga Garibotti (Vector Trading- Co-Orientador)

Prof. Dr. Fernando Gehm Moraes (PPGCC/PUCRS - Orientador)

DESENVOLVIMENTO DE UMA PLATAFORMA DE SIMULAÇÃO PARA AVALIAÇÃO DE ESTRATÉGIAS DE MERCADO BASEADA NO PUMA TRADING SYSTEM

RESUMO

As negociações automatizadas conduzidas por meios eletrônicos revolucionaram o comportamento dos mercados financeiros. Dois fatores chave surgem como críticos para alcançar a rentabilidade: a latência do sistema de negociação e a eficácia da estratégia concebida pelos participantes no mercado. Os sistemas de negociação tradicionais baseados em software muitas vezes sofrem de alta latência devido à análise de dados de mercado e à comunicação em rede. No entanto, eles se destacam na implementação de algoritmos de aprendizado de máquina para discernir padrões e prever tendências de mercado, tornando assim as estratégias de negociação dinâmicas. Por outro lado, hardware dedicado é empregado para melhorar vários componentes do sistema, como a pilha de protocolos de rede ou a pré-análise do protocolo financeiro. Neste contexto, o presente trabalho visa contribuir para a divulgação de informações a respeito de sistemas de negociação aplicados ao mercado financeiro. A ênfase está no estudo do novo ambiente de negociação e nos novos protocolos de comunicação que estão sendo implementados no sistema PUMA da B3, principal bolsa do Brasil. O objetivo é desenvolver um ambiente de simulação abrangente utilizando os novos protocolos. Este ambiente de simulação permitirá aos desenvolvedores aplicar suas estratégias de mercado e testar a capacidade de resposta de suas implementações.

Palavras-Chave: Mercado Financeiro, Protocolos de Comunicação, Plataforma de Simulação de Mercado Financeiro, Sistema PUMA, B3.

A SIMULATION PLATFORM DEVELOPMENT FOR ASSESSMENT OF MARKET STRATEGIES BASED ON THE PUMA TRADING SYSTEM

ABSTRACT

Automated negotiations driven by electronic means have revolutionized the behavior of financial markets. Two key factors emerge as critical for achieving profitability: the trading system latency and the efficacy of the strategy devised by market participants. Traditional software-based trading systems often suffer from high latency due to market data analysis and network communication. However, they excel in implementing machine learning algorithms to discern patterns and forecast market trends, thus rendering trading strategies dynamic. On the other hand, dedicated hardware is employed to enhance various system components, such as the network protocol stack or financial protocol pre-analysis. In this context, the present work aims to contribute to the dissemination of information regarding trading systems applied to the financial market. The emphasis lies in studying the novel trading environment and the new communication protocols that are being implemented in the PUMA trading system of B3, the main exchange in Brazil. The objective is to develop a comprehensive simulation environment utilizing the new protocols. This simulation environment will enable developers to apply their market strategies and test the responsiveness of their implementations.

Keywords: Financial Market, Communication Protocols, Financial Market Simulation Platform, PUMA Trading System, B3.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

API – Application Programming Interface

B3 – Brasil, Bolsa e Balcão

CME – Chicago Mercantile Exchange

COMEX – Commodity Exchange

DMA – Direct Market Access

DME – Dubai Mercantile Exchange

EUREX – European Derivatives Exchange

GUI – Graphical User Interface

FAST – FIX Adapted for Streaming

FIFO – First In, First Out

FIX – Financial Information Exchange

FPGA – Field-Programmable Gate Array

HFT – High-Frequency Trading

IPO – Initial Public Offering

MBP – Market by Price

MDB – Market Data Binary

NYMEX – New York Mercantile Exchange

OEB – Order Entry Binary

PUMA – Plataforma Unificada de Multiativos

RIP – Routing Information Protocol

ROS – Robotic Operation System

S.A. – Sociedade Anônima

SBE – Simple Binary Encoding

SSL – Secure Sockets Layer

TCP – Transmission Control Protocol

TOB – Top of Book

UDP – User Datagram Protocol

UMDF – Unified Market Data Feed

UTC – Coordinated Universal Time

SUMÁRIO

1. INTRODUÇÃO

Ao longo da última década, os mercados financeiros passaram por uma transformação significativa, indo de locais físicos tradicionais e negociações baseadas em pregões, que dependiam da tomada de decisões humanas, para negociações automatizadas baseadas em meios eletrônicos, impulsionadas por algoritmos avançados, sem intervenção humana. Uma grande parte destas negociações automatizadas baseia-se na identificação e exploração do *spread* do mercado; as diferenças transitórias entre os principais preços de venda e de compra de títulos [Boutros et al., 2017]. Um exemplo de negociações que exploram os *spreads* são as negociações de alta frequência (HFT, do inglês *High-Frequency Trading*). Referem-se ao uso de atrasos extremamente baixos e programas de computador sofisticados para realizar uma enorme quantidade de execuções comerciais em um curto espaço de tempo e acumular lucros através de transações extremamente frequentes e pequenas *spreads* [Kao et al., 2022].

A latência de resposta em um sistema de negociação e a eficácia de uma estratégia de negociação são pontos cruciais na lucratividade de uma empresa que opera no mercado financeiro. Em relação ao tempo de resposta em uma negociação, as soluções de software convencionais geralmente sofrem com altas latências decorrentes de diversas fontes, como a pilha de protocolos de rede ou latências imprevisíveis causadas pela instabilidade do sistema operacional [De et al., 2009]. Contudo, soluções baseadas em Field-Programmable Gate Array (FPGA) estão ganhando cada vez mais popularidade, pois podem atingir latência baixa e consistente, processando pacotes diretamente em hardware [Fu et al., 2017]. Aproveitando a natureza reprogramável de FPGAs, é possível otimizar o desempenho do sistema para aplicações específicas, resultando em um processamento mais rápido e eficiente de dados de mercado. Portanto, há uma tendência inevitável de transferir gradualmente tarefas críticas de latência de implementações de software para aceleração de hardware [Wang et al., 2020, Abeyrathne et al., 2021, Tan et al., 2021, Klaisoongnoen et al., 2022]. Por outro lado, a situação se inverte quando diz respeito à maioria das aplicações de estratégias de negociação. Nestes casos, a tendência atual é a utilização de aprendizado de máquina (em inglês, *machine learning*) e algoritmos de aprendizado profundo (do inglês, *deep learning*) em nível de software para revelar padrões, reduzir significativamente o risco de previsão de tendências, ou ajustar uma determinada estratégia de negociação de forma dinâmica [Nabipour et al., 2020, Rosati et al., 2020].

Levando em consideração os fatores temporais e lógicos como as principais influências na obtenção de lucro no mercado financeiro, este trabalho tem como foco o novo protocolo que está em implementação na B3, principal bolsa de valores do Brasil, com o objetivo de dar suporte à aplicação e validação de estratégias de forma eficiente. Para tanto,

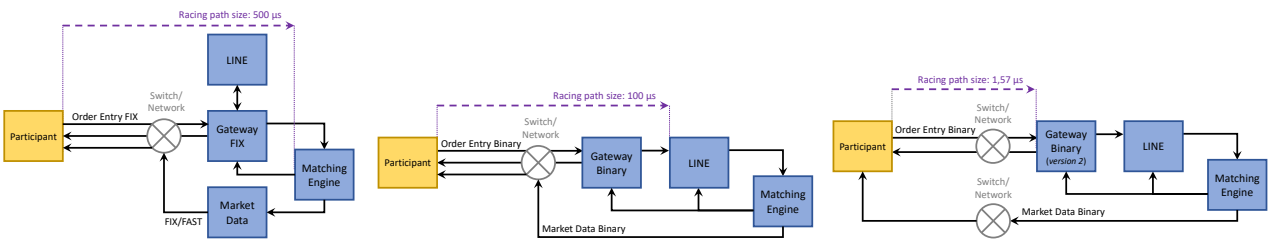
desenvolve-se um ambiente de simulação que permite testar estratégias e medir o tempo de execução.

O ambiente de simulação inclui participantes (do inglês, *players*), que simulam empresas e pessoas físicas no mercado; estes participantes são responsáveis pela decodificação e codificação de pacotes para a correta comunicação com o mercado financeiro. Após este processo, cria-se um livro de ofertas para relacionar as ofertas de cada produto de acordo com sua chegada. Este ambiente de simulação é extremamente útil, pois permite a desenvolvedores testar estratégias de mercado num contexto onde o tempo gasto na codificação e decodificação é minimizado, habilitando manter um foco exclusivo na aplicação de estratégias de mercado.

Esta abordagem de implementação de um ambiente de simulação é importante para maximizar a eficiência das operações financeiras, proporcionando aos participantes do mercado uma plataforma para testar e refinar as suas estratégias. Ademais, a capacidade de medir o tempo de execução das estratégias e suas interações com outros participantes permite uma análise precisa da sua viabilidade e eficácia, contribuindo para uma tomada de decisão mais informada e assertiva.

1.1 Motivação

Os clientes da B3 apresentam uma demanda crescente por maior previsibilidade, liquidez e resiliência nos ambientes de negociação. Em resposta a essa tendência, a B3 oferece o *PUMA Trading System*, uma plataforma que integra diversas funcionalidades, capacidades de gerenciamento de riscos e transações operacionais, garantindo um ambiente operacional seguro para seus clientes. Porém, para aprimorar a infraestrutura e trazer maior determinismo e transparência nas transações realizadas em seu sistema, a B3 está realizando uma ampla modificação em sua plataforma de negociação PUMA existente, conforme ilustra a Figura 1.1.



(a) Arquitetura inicial (anterior a Dezembro 2023)

(b) Arquitetura atual (a partir de Dezembro 2023)

(c) Planejamento da arquitetura futura

Figura 1.1 – Evolução do *PUMA Trading System* da B3. Adaptado de [B3, 2023c].

A Figura 1.1 é dividida em participante (laranja) e o *PUMA Trading System*, representado por seus diferentes blocos de cor azul. Ela mostra 3 momentos distintos da

evolução da infraestrutura do *PUMA Trading System* que impactam diretamente essa dissertação de mestrado. O primeiro momento (Figura 1.1.a) mostra uma infraestrutura com suporte a protocolos de comunicação que são padrões neste segmento, por exemplo o FIX/FAST. Já no momento atual (Figura 1.1.b) apresenta-se um protocolo de comunicação proprietário da B3, e os próximos passos dessa evolução (Figura 1.1.c) mostram uma redução da latência de corrida na utilização de *switches* dedicados e na troca de onde acaba o caminho da corrida (em inglês, *racing path size*).

A motivação deste trabalho decorre das mudanças que estão sendo implementadas pela B3 em seu sistema PUMA. Uma das principais mudanças está no protocolo de comunicação, que no início dessa dissertação utilizava o FIX/FAST e que está sendo migrado para UMDF – *Unified Market Data Feed*. Com esta transição, torna-se crucial fornecer plataformas que permitam aos desenvolvedores de estratégias de mercado testar as suas implementações neste novo ambiente, utilizando este protocolo.

Este trabalho reconhece a necessidade de fornecer um ambiente propício para desenvolvedores testarem suas estratégias de mercado no novo sistema PUMA, considerando as mudanças no protocolo de comunicação. O processamento e a análise das mensagens, incluindo descompressão e compressão, torna-se um fator crítico para o sucesso das estratégias de negociação. Portanto, é essencial fornecer uma plataforma que permita simular a eficácia das estratégias relativas a este novo ambiente, onde o correto processamento das mensagens é de grande importância.

Ao fornecer uma plataforma de simulação adequada, o desenvolvimento e o aprimoramento de estratégias de mercado são facilitados, permitindo que desenvolvedores se concentrem especificamente na adaptação de suas implementações ao novo protocolo de comunicação. Isto contribui para a preparação dos participantes no mercado para as mudanças iminentes, garantindo que possam aproveitar ao máximo as oportunidades oferecidas pelo novo sistema PUMA, com especial atenção à eficiência das transações.

1.2 Objetivos e Contribuições

O objetivo geral deste Mestrado é propor um ambiente de simulação para avaliação de estratégias de mercado baseado no novo protocolo que está sendo implementado pela B3. Este será executado em nível de *software*, para que empresas e negociantes do mercado possam testar suas estratégias, e possam observar os resultados em uma operação completa simulada, desde a chegada do *feed*, até a chegada da ordem no livro de ofertas.

O objetivo geral será alcançado através dos seguinte conjunto de objetivos específicos:

1. Estudo dos conceitos relacionados às negociações realizadas em bolsas de mercadorias e futuros;
2. Estudo dos protocolos de comunicação utilizados em bolsas de mercadorias e futuros;
3. Estudo do sistema de negociações utilizado na bolsa de mercadorias e futuros do Brasil (B3), e sua devida atualização;
4. Estudo e integração de um codificador/decodificador [Oliveira and Bonato, 2023] com o livro de ofertas;
5. Desenvolvimento de um multiplexador para controle e recepção de mensagens dos participantes (codificadores/decodificadores);
6. Integração das várias partes do ambiente a nível de software;
7. Simulação de operações completas do mercado brasileiro através de estudos de caso;
8. Avaliação de desempenho do ambiente em software.

Note-se que a realização de cada um destes objetivos específicos são contribuições deste mestrado. Desta forma, este volume descreve a proposta de um ambiente de simulação robusto e eficiente utilizando o novo protocolo da B3. O propósito é permitir que desenvolvedores testem suas estratégias de mercado e avaliem o tempo de resposta de suas implementações, contribuindo para que melhores decisões sejam tomadas pelos participantes do mercado.

1.3 Metodologia

O processamento eficiente de mensagens em um sistema de negociação desempenha um papel crucial na obtenção da rentabilidade para certos tipos de produtos do mercado financeiro. Consequentemente, há uma demanda crescente por sistemas mais fidedignos; por isto, o simulador proposto fornece um ambiente completo e que dá suporte a protocolos mais recentes do mercado financeiro brasileiro.

Com base em um sólido conhecimento sobre o novo protocolo da B3, aplica-se o mesmo ao ambiente *Encoder/Decoder* com base em referências anteriores da literatura (por exemplo, [Oliveira and Bonato, 2023]). Usando o mecanismo codificador/decodificador com o novo protocolo UMDF, desenvolve-se um livro de ofertas com base na referência [Moscon, 2023]. Com estes mecanismos, desenvolveu-se um multiplexador responsável por controlar a chegada dos dados e adiciona-se estes a uma fila para que o livro de ofertas possa compilar as informações.

Com o ambiente de software disponível, testa-se este em um contexto de negociações do mercado através de estudos de caso. Essa abordagem permite a validação abrangente do ambiente de desenvolvimento, proporcionando a engenheiros a oportunidade de aplicar e validar suas estratégias em um ambiente compatível com o novo protocolo de comunicação da B3. Com esta abordagem, pretende-se contribuir para a compreensão e disseminação de informações sobre o desenvolvimento de sistemas no contexto de negociações baseados no novo *PUMA Trading System*.

1.4 Organização do Documento

O restante deste volume está organizado em cinco capítulos:

- O Capítulo 2 descreve conceitos básicos necessários para acompanhar o trabalho realizado;
- O Capítulo 3 apresenta e discute um conjunto de trabalhos relacionados a sistemas de negociação;
- O Capítulo 4 apresenta o ambiente de simulação do sistema de negociação desenvolvido, comparando o mesmo com o ambiente de negociação utilizado pela B3;
- O Capítulo 5 apresenta resultados do exercício com o ambiente de simulação, iniciando pela análise do desempenho dos protocolos de comunicação, e finalizando com um estudo de caso para a aplicação de estratégias do mercado;
- O Capítulo 6 conclui o texto, apontando direções para trabalhos futuros.

2. CONCEITOS BÁSICOS

Este Capítulo descreve o que são as bolsas de mercadorias e futuros, a fim de introduzir o ambiente onde este trabalho se encaixa, mostrando também alguns exemplos específicos destas. Na sequência apresenta-se os sistemas de negociação que estas bolsas utilizam, que estão relacionados com o sistema proposto neste trabalho. Em seguida, mostra-se um pouco das estratégias de negociação utilizadas nas bolsas de mercadorias e futuros, com foco nas estratégias orientadas por pedidos e uma apresentação de negociação de alta frequência (do inglês *high-frequency trading* - HFT). Finalizando o Capítulo, abordam-se os principais protocolos de comunicação utilizados nas bolsas de mercados e futuros, com destaque nos protocolos específicos utilizados na bolsa brasileira.

2.1 BOLSAS DE MERCADORIAS E FUTUROS

2.1.1 Diferença entre Bolsa de Valores e Bolsa de Mercadorias e Futuros

A bolsa de valores é um ambiente onde são negociadas ações, que representam pequenas parcelas de uma empresa. Essas ações são emitidas por empresas de capital aberto. No Brasil, as empresas conhecidas como sociedade anônima (S.A.), isto é, têm “S.A.” em sua razão social, podem oferecer e ter suas ações negociadas na bolsa de valores.

Para que uma empresa de sociedade limitada se torne uma S.A. e tenha suas ações negociadas na bolsa de valores, ela deve realizar uma oferta pública inicial (do inglês *initial public offering* - IPO). Geralmente, as empresas de capital fechado realizam IPO quando buscam um grande volume de dinheiro para expandir seus negócios, como a abertura de filiais ou aquisição de novas máquinas.

Por outro lado, existe a bolsa de mercadorias e futuros que realiza negociação de contratos futuros e de opções, incluindo *commodities* como ouro, petróleo, grãos, entre outros, através de derivativos e ativos financeiros, como taxas de juros e câmbio.

A maioria dos contratos é apenas financeiro, para estabelecer um preço de compra e venda. Isso é uma grande diferença em relação à bolsa de valores, onde o resultado de lucro ou prejuízo só ocorre quando a ação é vendida.

2.1.2 Tipos de Mercadorias

As mercadorias que são negociadas no mercado financeiro são bens que têm valor econômico e que podem ser padronizados, como produtos agrícolas, minerais, metais, energia, entre outros. Elas são negociadas em bolsas de mercadorias e futuros, onde os compradores e vendedores estabelecem os preços e os contratos futuros ou à vista. As mercadorias podem ser usadas como forma de proteção contra a inflação, a variação cambial ou a escassez de oferta.

Os principais produtos negociados nas bolsas de mercadorias e futuros são petróleo, ouro, milho, gás natural, soja, café, açúcar, algodão, cobre e prata [Stumpf, 2023].

Cada mercadoria tem suas características, seus fatores de influência e suas estratégias de negociação. Além disso, essas mercadorias variam de acordo onde elas são negociadas. Abaixo são apresentadas as principais mercadorias negociadas em bolsas relevantes pelo mundo:

- **B3:** contratos futuros de *commodities* como café arábica, etanol hidratado, açúcar, gado, milho e soja [B3, 2023a];
- **CME:** contratos futuros de taxas de juros, moedas estrangeiras, metais, energia, entre outros [CME, 2023];
- **NYMEX:** contratos futuros de *commodities* como petróleo, gás natural, ouro, prata, entre outros [Reis, 2020];
- **Eurex:** contratos futuros de *commodities*, incluindo metais, energia e algumas mercadorias agrícolas [EUREX, 2023a];
- **DME:** negociação de contratos de futuros de petróleo Oman Crude Oil [DME, 2023b].

2.2 BOLSAS PELO MUNDO

Esta Seção apresenta algumas bolsas de mercadorias e futuros pelo mundo. As bolsas selecionadas tem relevância para este trabalho, como a brasileira B3 [B3, 2019], usada como exemplo para aplicação deste trabalho. Outras são a CME Chicago [CME, 2024] e a NYMEX Nova York [NYMEX, 2023], importantes bolsas de mercadorias e futuros dos Estados Unidos e onde a B3 se inspirou para melhorar a sua infraestrutura de negociação. Além destas aborda-se a Eurex [Ricardo, 2020] e a DME [DME, 2023a], situadas na Alemanha e nos Emirados Árabes Unidos, respectivamente.

2.2.1 B3 – Brasil

A Brasil, Bolsa e Balcão (B3) [B3, 2019], anteriormente conhecida como BM&FBOVESPA, é a principal bolsa do Brasil. Como instituição financeira, a B3 possibilita e facilita a negociação de uma ampla gama de ativos financeiros, incluindo ações, títulos, derivativos, moedas e *commodities*. Por meio de sua plataforma eletrônica, a B3 proporciona um ambiente seguro e regulamentado para compradores e vendedores realizarem transações. Além das atividades de negociação, a B3 também desempenha funções de gestão pós-negociação. Isto envolve a manutenção de registros, compensação, liquidação e custódia das transações, garantindo a integridade e segurança das operações.

Vale destacar que a B3 é uma das principais bolsas de valores e mercadorias globais, proporcionando uma vasta variedade de produtos, como ações, moedas estrangeiras e *commodities*. Ela oferece uma variedade de contratos futuros e está constantemente criando novos contratos em resposta à identificação de demandas de mercado. Os contratos futuros na B3 estão distribuídos em quatro grandes segmentos: juros, moedas, índices e *commodities*. Em relação às *commodities* negociadas, o Brasil se destaca por ser um país agrícola. Neste sentido, são negociadas na B3 café, açúcar, gado, milho e soja.

Como entidade responsável pela divulgação dos principais índices de mercado do Brasil, como o Ibovespa, a B3 oferece acesso a diversos tipos de investidores, incluindo instituições financeiras, corretoras e investidores individuais. Por meio de diversos canais e interfaces de acesso, a B3 permite que esses participantes se envolvam no mercado financeiro de acordo com suas necessidades e interesses.

A B3 desempenha um papel importante no desenvolvimento econômico do País, facilitando o financiamento corporativo e proporcionando oportunidades de investimento. Além disto, a presença da B3 no mercado ajuda a atrair capital estrangeiro para o Brasil, contribuindo para o crescimento e fortalecimento da economia nacional.

2.2.2 CME – Chicago

A Bolsa de Mercadorias de Chicago [CME, 2024], conhecida como *Chicago Mercantile Exchange* ou CME, é uma instituição localizada na cidade de Chicago, nos Estados Unidos. Sua origem data de 1898, quando foi estabelecida com o nome de *Chicago Butter and Egg Board*.

A trajetória da CME teve um redirecionamento significativo em 2002, quando a instituição passou por um processo de IPO, abrindo seu capital para a participação de novos investidores [CME, 2024].

Em 2007, um marco relevante foi alcançado por meio da fusão com a *Chicago Board of Trade*, resultando na criação do CME Group. Nasceu assim uma das maiores bolsas de valores do mundo, mesmo que sua notoriedade seja inferior às tradicionais NYSE e NASDAQ, inclusive devido à natureza dos ativos que a integram.

Atualmente, a instituição está imersa em um processo contínuo de modernização de ativos e procedimentos. Por exemplo, em 2017, incorporou a negociação de Bitcoins à sua lista de ativos. Essa medida é notável, uma vez que as cripto-moedas têm demonstrado significativa valorização ao longo dos últimos anos.

Outro indicativo de sua capacidade de adaptação reside no método de negociação empregado pelos investidores para transacionar instrumentos na *Chicago Mercantile Exchange*. A bolsa utiliza uma plataforma eletrônica denominada CME Globex (ver a Sessão 2.3.2), embora ainda seja possível participar do mercado por meio dos convencionais comandos de voz.

A *Chicago Mercantile Exchange* também ganhou reconhecimento por ser a pioneira entre as Bolsas de Mercadorias e Futuros a se transformar em uma empresa de capital aberto, permitindo a participação de investidores.

Atualmente, embora tenha iniciado como uma entidade sem fins lucrativos, a CME destaca-se como uma relevante Bolsa de Mercadorias e Futuros, concentrando-se especialmente em contratos futuros e opções. É importante ressaltar que esses instrumentos apresentam níveis mais elevados de risco e volatilidade em comparação com ativos convencionais [Equipe Mais Retorno, 2020].

2.2.3 NYMEX – Nova York

O *New York Mercantile Exchange* (NYMEX) [NYMEX, 2023] é uma das maiores bolsas de *commodities* do mundo, tendo sido estabelecida em 1872. Reconhecido como o principal mercado físico para a negociação de contratos futuros de *commodities* em escala mundial, o NYMEX tem sua sede em Manhattan, Estados Unidos, e mantém escritórios adicionais em Boston, Washington, Atlanta e San Francisco nos EUA, além de presença internacional em Dubai, Londres e Tóquio.

Especializado na negociação de contratos futuros de energia, notadamente de petróleo bruto e seus derivados, como o contrato futuro de petróleo do WTI (*West Texas Intermediate*), que é amplamente utilizado como referência global nos mercados de petróleo, o NYMEX também abrange contratos futuros de gás natural, gasolina, óleo de aquecimento e outros produtos energéticos.

Diante da crise financeira de 2008, o NYMEX, enfrentando desafios significativos, optou por se associar ao CME Group, que adquiriu a bolsa nesse mesmo ano, incluindo

também o COMEX (*Commodity Exchange*). Essa transação resultou na expansão da lista de ativos negociados, incorporando contratos relacionados a metais, energia e agricultura.

A tradicional negociação face a face, padrão por décadas, foi substituída progressivamente por sistemas eletrônicos, a partir de 2006, tornando o processo mais eficiente e econômico. Em 30 de dezembro de 2016, o "pit", local de negociação presencial, foi oficialmente encerrado devido à diminuição do volume de transações nesse formato. Como o NYMEX faz parte do grupo CME, utiliza-se o sistema GLOBEX (Sessão 2.3.2), o mesmo utilizado na Bolsa de Chicago.

2.2.4 Eurex – Alemanha

A Eurex [Ricardo, 2020] foi estabelecida em 1998 como uma junção entre a *Deutsche Börse AG* e a *SIX Swiss Exchange*. Este período coincidiu com a transição do sistema de negociação tradicional, baseado em voz, para os emergentes sistemas de negociação eletrônica. A Eurex destacou-se como uma das pioneiras ao oferecer uma plataforma de negociação totalmente eletrônica, antecipando as mudanças tecnológicas no setor.

Em 2012, a *Deutsche Börse AG* adquiriu as ações da SIX na Eurex, consolidando-se como a única proprietária da bolsa [Ricardo, 2020].

Embora seu foco seja os mercados europeus, a Eurex mantém uma presença global, atraindo participantes e investidores de diversos países. A bolsa desempenha um papel importante no cenário global de derivativos.

Atualmente, a Eurex diversificou sua oferta de produtos, abrangendo derivativos de taxas de juros, ações, dividendos, câmbio estrangeiro, *commodities* e imóveis. A *Eurex Clearing AG*, uma subsidiária da Eurex, desempenha um papel crucial ao fornecer serviços de compensação para as transações conduzidas na plataforma Eurex, contribuindo para a integridade e estabilidade do mercado financeiro.

O sistema de plataforma de negociação adotado pela Eurex é o T7 (Sessão 2.3.3), desenvolvido pelo *Deutsche Börse Group*. Vale destacar que essa tecnologia é compartilhada com outras entidades, como a *European Energy Exchange* (EEX) e a *Powernext*, para a negociação de derivativos.

2.2.5 DME – Dubai

A *Dubai Mercantile Exchange* (DME) [DME, 2023a] é a principal bolsa de *commodities* focada em energia no Oriente Médio e a terceira maior referência mundial para o petróleo bruto. Lançada em 2007, a DME introduziu a negociação de preços justos e trans-

parentes, além de eficiente gestão de risco para o mercado da região. O transação principal da DME é o contrato futuro de petróleo bruto de Omã, proporcionando uma referência para os preços oficiais de venda do petróleo bruto na região, como Omã e Dubai. Esses preços historicamente orientam as transações de exportação de petróleo bruto do Oriente Médio para a Ásia.

Localizada no Centro Financeiro Internacional de Dubai (DIFC), uma zona financeira projetada para impulsionar os serviços financeiros nos Emirados Árabes Unidos, a DME assegura que todas as negociações realizadas em sua plataforma sejam compensadas e garantidas pela CME.

2.3 SISTEMAS DE NEGOCIAÇÃO

Esta Seção descreve alguns dos sistemas de negociação utilizados nas bolsas de mercados e futuros apresentados na Sessão 2.2.

2.3.1 PUMA

A Plataforma Unificada de Multiativos (PUMA) é o sistema utilizado pela B3. Caracterizada pela eficiência e segurança, o PUMA oferece uma gama de funcionalidades essenciais para as operações realizadas na bolsa [B3, 2019].

Com interface avançada para envio e execução de ordens, o sistema PUMA permite ao investidor realizar transações de compra ou venda com rapidez e precisão. A plataforma foi desenvolvida para atender um grande volume de transações simultâneas, garantindo eficiência operacional e reduzindo o tempo de resposta. Além disto, o sistema oferece diversas funcionalidades avançadas, permitindo aos investidores personalizar suas estratégias de negociação e definir condições específicas para suas ordens.

O PUMA está integrado ao sistema de pós-negociação da B3, garantindo a transição das operações executadas no sistema para os processos de registro, compensação, liquidação e custódia. Isso torna o fluxo operacional mais seguro e eficiente. Além disso, o sistema PUMA fornece informações em tempo real sobre cotações, volumes de negociação e outras métricas relevantes, permitindo que os investidores monitorem o mercado de forma dinâmica e tomem decisões com base em dados atualizados.

O sistema PUMA opera de acordo com os seguintes procedimentos:

- **Envio de ordens:** os participantes do mercado utilizam suas interfaces de acesso para enviar ordens de compra ou venda de ativos financeiros ao sistema;

- **Correspondência de ordens:** o sistema emprega um mecanismo de correspondência de ordens para analisar as ordens de compra e venda recebidas e identificar aquelas que podem ser executadas. Quando uma ordem de compra atende a uma ordem de venda com condições compatíveis, ocorre uma transação;
- **Execução de ordens:** após ser encontrada uma correspondência entre as ordens de compra e venda, o sistema executa as ordens, resultando na compra e venda dos ativos negociados. Estas transações são devidamente registradas e incorporadas ao histórico de transações;
- **Monitoramento e atualizações em tempo real:** o sistema monitora continuamente pedidos e transações, fornecendo informações em tempo real sobre cotações, volumes de negociação, e outras métricas relevantes.

Segurança e integridade são elementos essenciais do sistema PUMA e diversas medidas são tomadas para garantir a sua proteção. Entre os mecanismos implementados destacam-se a criptografia de dados, controles de acesso e sistemas de redundância.

Vale destacar também que a B3 realiza auditorias internas e externas para avaliar a conformidade do sistema PUMA com as regulamentações vigentes e as melhores práticas de segurança. Estas auditorias desempenham um papel importante na identificação de potenciais vulnerabilidades e na garantia da adesão aos padrões de segurança estabelecidos.

A B3 também disponibiliza múltiplas interfaces de acesso ao sistema PUMA, permitindo que os participantes do mercado se conectem à plataforma de acordo com suas necessidades e exigências específicas. Estas interfaces incluem opções como acesso à web, interfaces de programação de aplicativos (APIs) e soluções de conectividade dedicadas.

2.3.2 CME Globex

CME Globex é a primeira plataforma eletrônica de negociação de futuros do mundo, criada pelo CME Group, dedicada à execução de uma ampla variedade de produtos financeiros. Operando praticamente 24 horas por dia, o sistema GLOBEX da CME se destaca não apenas pela amplitude de sua operação contínua, mas também pela oferta abrangente de serviços e ferramentas essenciais para os participantes do mercado.

O sistema GLOBEX não se limita apenas à execução de ordens, ele é projetado para proporcionar uma experiência completa de negociação eletrônica. Uma característica crucial é a oferta de serviços e ferramentas de gerenciamento de risco, proporcionando aos participantes do mercado as ferramentas necessárias para administrar os riscos e evitar

exposições indesejadas. Além disso, a plataforma garante acesso a dados de mercado em tempo real, fornecendo informações críticas para tomada de decisões informadas. A conectividade direta e indireta oferecida pelo sistema amplifica a acessibilidade, permitindo uma participação flexível nos mercados de derivativos do CME Group.

O funcionamento do sistema GLOBEX se dá por meio de uma plataforma eletrônica que atua como a ponte entre os clientes e os mercados de derivativos do CME Group. Os clientes têm a flexibilidade de utilizar uma interface gráfica (GUI) ou uma API para enviar, modificar ou cancelar ordens eletrônicas ao longo de cinco sessões diárias distintas. A eficiência do sistema é evidenciada pela execução de ordens em milissegundos, destacando a importância da velocidade nas transações financeiras modernas. Além disso, o sistema conta com algoritmos avançados para garantir a execução de negócios conforme as regras de prioridade de preço e tempo.

O sistema GLOBEX da CME conta com atualizações frequentes. As atualizações regulares do sistema são direcionadas para aprimorar a velocidade, aumentar a capacidade de processamento, lidar com volumes substanciais de transações e introduzir novos produtos aos seus usuários.

As principais características do sistema GLOBEX são [Group, 2023]:

- **Rapidez de execução:** o sistema é capaz de processar negócios em milissegundos, o que significa que os clientes podem interagir com o mercado em tempo real;
- **Disponibilidade:** o sistema opera praticamente 24 horas por dia (exceto aos sábados e alguns feriados), permitindo que os clientes negociem produtos derivativos em diferentes fusos horários e regiões;
- **Diversidade:** o sistema abrange uma grande variedade de produtos derivativos, como *commodities*, energia, metais, moedas e taxas de juros. Os clientes podem negociar contratos de referência mundial em uma única plataforma;
- **Segurança:** o sistema fornece transparência, anonimato e integridade do mercado, além de ferramentas de gerenciamento de risco para proteger os clientes de exposições indesejadas;
- **Inovação:** o sistema é constantemente aprimorado para oferecer aos clientes maior velocidade, volume, capacidade e novos produtos. Ele também permite que os clientes criem seus próprios *spreads* e usem preços implícitos para melhorar a liquidez. Além disto, opera com antigos e novos protocolos, como o FIX/FAST (Sessão 2.5.1) e os protocolos binários SBE (Sessão 2.6.1).

2.3.3 T7 Trading System

O *T7 trading system* é uma plataforma eletrônica de negociação desenvolvida pelo Grupo Deutsche Börse, que visa revolucionar a forma como os *traders* e investidores acessam as oportunidades de mercado em todo o mundo [EUREX, 2023c]. Ele tem como uma de suas das principais características a baixa latência, fazendo com que os participantes possam operar no mercado em tempo real. Além disso, possui um sistema robusto, podendo lidar com um alto volume de transações e uma grande flexibilidade para introduzir novos produtos e novas funcionalidades rapidamente.

O *T7 trading system* opera por meio de uma arquitetura que conecta os clientes com os mercados de derivativos e de ações do Grupo Deutsche Börse. Assim como em outros sistemas, os clientes podem utilizar uma GUI ou uma API para realizar suas operações no mercado.

O *T7 trading system* utiliza os seguintes protocolos para a comunicação entre os clientes e o sistema [EUREX, 2023b]:

- **Enhanced Trading Interface (ETI):** é o protocolo nativo do T7, que oferece velocidade e funcionalidade para os clientes. Ele permite o envio e a modificação de ordens, a recepção de confirmações de negócios, a consulta de dados de mercado e de referência, e o gerenciamento de risco;
- **FIX Gateway:** é o protocolo padrão da indústria, que oferece uma interface simples e padronizada para os clientes. Ele permite o envio e a modificação de ordens, a recepção de confirmações de negócios, e a consulta de dados de mercado e de referência (Sessão 2.5.1);
- **Market Data Interface:** é o protocolo que fornece os dados de mercado em tempo real para os clientes, incluindo as ofertas, os negócios, os instrumentos, as estatísticas e os eventos do mercado;
- **Reference Data Interface (RDI):** é o protocolo que fornece os dados de referência para os clientes, incluindo as informações sobre os produtos, os segmentos, os calendários e as sessões de negociação.

2.4 MECANISMOS E ESTRATÉGIAS DE NEGOCIAÇÃO

2.4.1 Mecanismos de Negociação: Orientados por Pedidos/Ordens

Em bolsas de mercadorias e futuros os mecanismos de negociação são orientados por pedido/ordens, isto é, esses sistemas permitem as interações diretas entre os compradores e vendedores de ativos financeiros, como moedas e derivativos. Nestes sistemas, os participantes enviam ordens de compra ou de venda, que são registradas em um livro de ofertas e executadas de acordo com as regras pré-estabelecidas, com os preços de cada negociação sendo determinados pelas ofertas e demandas dos ativos do mercado.

As negociações podem ser classificadas em diferentes tipos, como leilão, mercado contínuo e mercado de balcão, de acordo com o grau de liquidez, eficiência e regulação do mercado.

- **Leilão:** ocorre em horários específicos, onde as ordens são acumuladas e executadas ao mesmo tempo, ao preço que maximiza o volume negociado. Por exemplo, o leilão de abertura e fechamento da bolsa;
- **Mercado Contínuo:** ocorre durante todo o horário de funcionamento do mercado, onde as ordens são executadas assim que encontram uma parte compatível. Por exemplo, o mercado a vista de ações;
- **Mercado de Balcão:** ocorre fora da bolsa, onde as ordens são negociadas diretamente entre as partes, sem a divulgação pública dos preços e volumes. Por exemplo, o mercado de títulos privados.

2.4.2 Estratégias de Negociação

Existem diferentes tipos de operações no mercado financeiro, elas variam de acordo com o tempo de permanência na posição, objetivo de lucro e o nível de risco. Algumas das estratégias de negociação são:

- **Scalping:** é uma operação de curtíssimo prazo, que busca aproveitar pequenas variações nos preços dos ativos. Um *scalper*, por exemplo, costuma realizar de 15 a 70 operações diariamente, usando gráficos de minutos ou segundos;
- **Day trading:** é uma operação de curto prazo, que consiste em comprar e vender um ativo no mesmo dia, sem deixar posições abertas ao final do pregão. O objetivo do

day trader é lucrar com as movimentações diárias do mercado, usando gráficos de 5 a 60 minutos. Essa modalidade requer atenção, experiência e gestão de riscos do *trader*;

- **Swing trading:** é uma operação de médio prazo, que busca capturar uma tendência de alta ou baixa de um ativo, mantendo a posição por alguns dias ou semanas. O *swing trader* usa gráficos de 60 minutos ou diários, e tem uma expectativa de lucro maior do que o *day trader*. Essa modalidade demanda paciência, análise e planejamento do *trader*;
- **Position trading:** é uma operação de longo prazo, que visa investir em um ativo com base em sua valorização futura, permanecendo na posição por meses ou anos. O *position trader* usa gráficos semanais ou mensais, e tem uma visão mais ampla do mercado. Essa modalidade envolve conhecimento, diversificação e alocação de ativos.

2.4.3 Negociação de Alta Frequência (*High Frequency Trading* – HFT)

A negociação de alta frequência (do inglês *high frequency trading* – HFT) é uma estratégia de negociação que utiliza computadores e algoritmos de alta velocidade para executar negociações em um ritmo rápido. O HFT é caracterizado pelo uso de tecnologia avançada, como *colocation*, conexões de rede de baixa latência e algoritmos sofisticados, para processar dados de mercado e executar negociações em tempo real. As empresas de HFT confiam em sua capacidade de analisar grandes quantidades de dados e tomar decisões rapidamente para obter uma vantagem competitiva no mercado.

Uma das principais características de HFT é a sua velocidade. Os algoritmos de HFT podem executar negociações em questão de microssegundos, permitindo que os *traders* respondam rapidamente às mudanças nas condições do mercado e explorem as ineficiências do mercado. Os algoritmos de HFT normalmente usam modelos matemáticos complexos e análises estatísticas para identificar padrões em dados de mercado e tomar decisões comerciais com base nesta análise.

Outra característica importante de HFT é a sua escalabilidade. As estratégias de HFT podem ser usadas para negociar uma variedade de instrumentos financeiros, incluindo ações, opções, futuros e moedas. Os algoritmos de HFT também podem ser adaptados a diferentes mercados e ambientes de negociação, permitindo que os *traders* capitalizem as oportunidades de mercado em todo o mundo.

HFT oferece diversas vantagens competitivas em relação às estratégias de negociação tradicionais. Uma das principais vantagens é a sua capacidade de executar negociações de forma rápida e eficiente, o que pode resultar em lucros mais elevados e menor

exposição ao risco. Os algoritmos de HFT também podem ser programados para tomar decisões com base em dados de mercado em tempo real, permitindo que os *traders* respondam rapidamente às mudanças nas condições do mercado e adaptem as suas estratégias conforme necessário.

Outra vantagem de HFT é a sua capacidade de reduzir os custos de negociação. As empresas de HFT utilizam frequentemente ligações de rede de baixa latência e serviços de *colocation* para reduzir o tempo necessário para receber e processar dados de mercado, o que pode resultar em custos de transação mais baixos e maior rentabilidade. Além disso, os algoritmos de HFT podem ser usados para executar negociações em grandes volumes, permitindo que os *traders* aproveitem as economias de escala e reduzam ainda mais os custos de negociação.

Além de utilizar a estratégia de *scalping* (Sessão 2.4.2), as operações de alta frequência utilizam outras estratégias para operar no mercado:

- **Arbitragem:** explora as diferenças de preços entre dois ou mais mercados, comprando um ativo em um deles e vendendo em outro, obtendo lucro com a diferença. Essa estratégia requer uma rápida execução e uma baixa latência, ou seja, o menor tempo possível entre a emissão e a recepção de uma ordem;
- **Tape reading:** analisa o fluxo de ordens de um determinado ativo, observando o livro de ofertas, o volume e a direção das negociações. O objetivo é identificar as tendências e os padrões de comportamento do mercado, antecipando-se aos movimentos dos preços;
- **Criação de mercado:** oferece liquidez ao mercado, colocando ordens de compra e venda simultaneamente, com uma pequena margem de lucro entre elas. O objetivo é ganhar com o *spread*, ou seja, a diferença entre o preço de compra e o preço de venda de um ativo;
- **Negociações baseadas em notícias:** reage rapidamente às notícias e aos eventos que podem afetar o mercado, aproveitando as oportunidades de lucro que surgem com as variações de preços. Essa estratégia requer uma alta capacidade de processamento de dados e de interpretação de linguagem natural.

2.5 Protocolos de Comunicação

Nesta seção apresenta-se protocolos de comunicação suportados nos principais sistemas de negociação (Sessão 2.3).

2.5.1 FIX/FAST

FAST (*FIX Adapted for Streaming*) é um protocolo de comunicação usado para *streaming* de dados financeiros em tempo real. É baseado no protocolo *Financial Information Exchange* (FIX), amplamente utilizado no setor financeiro para roteamento e execução de pedidos. O FAST foi projetado para fornecer transmissão de dados de alta velocidade e baixa latência para *feeds* de dados de mercado, permitindo que os *traders* respondam rapidamente às mudanças no mercado.

Observe que os protocolos de comunicação são essenciais para a transmissão de dados eficiente e confiável em uma ampla gama de aplicações. Ethernet, TCP/IP, UDP e FAST são apenas alguns exemplos dos muitos protocolos de comunicação usados em vários setores e aplicações. Cada protocolo possui seu próprio conjunto de recursos e vantagens, dependendo dos requisitos específicos da aplicação.

2.5.1.1 Arquitetura do Protocolo FIX

A arquitetura FIX consiste em duas camadas principais e os tipos de conexão.

- **Camada de Sessão**: esta camada trata da inicialização, manutenção e encerramento de conexões. Além disso, ele lida com o *handshake* entre transferências ordenadas de mensagens, o que também abrange integridade de dados, entrega de mensagens e sequenciamento dos dados;
- **Camada de Aplicação**: esta camada lida com funções específicas do negócio, como criação, cancelamento e substituição de pedidos. Ademais, é responsável pela assinatura dos dados de mercado, além de processar e analisar os relatórios de execução;
- **Tipos de conexão**: para as informações de preços não é preciso o uso de uma camada de soquetes seguros (do inglês *Secure Sockets Layer* – SSL), já para negociações é obrigatório o uso via *SSL Tunneling*.

2.5.1.2 Estrutura de Mensagem

No protocolo FIX, as mensagens seguem um formato estruturado, normalmente na forma de *strings* baseadas em texto. O formato permite a transmissão de diversos tipos de informações financeiras de forma padronizada. Cada mensagem FIX é composta por diferentes campos de dados, que servem como contêineres para informações específicas. Esses campos são delimitados por caracteres especiais que funcionam como separadores para distinguir um campo de outro na mensagem. Os campos dentro de uma mensagem FIX são usados para representar vários tipos de dados, como identificadores de pedidos,

preços, quantidades e informações de data/hora. Eles fornecem uma maneira padronizada de transmitir detalhes específicos sobre uma negociação ou pedido.

A estrutura de mensagens do protocolo FIX segue o seguinte formato:

- **Cabeçalho (*Header*)**: contém campos que fornecem informações gerais sobre a mensagem, como a versão do protocolo, o remetente da mensagem e o destinatário pretendido. Esta informação garante que a mensagem seja encaminhada com precisão e identificada corretamente pela parte receptora;
- **Corpo (*Body*)**: este campo contém informações essenciais de negócios ou de mercado. O conteúdo do corpo da mensagem pode variar dependendo do tipo específico de mensagem. Por exemplo, uma mensagem de pedido incluirá campos relativos aos detalhes do pedido, enquanto que uma mensagem de execução conterá campos específicos aos detalhes de execução do pedido;
- **Rodapé (*Trailer*)**: é composto por campos adicionais que servem para fornecer informações de controle e validação de mensagens. Estes campos abrangem códigos de verificação e sequência, que desempenham um papel crucial na garantia da integridade e autenticidade da mensagem transmitida.

A Figura 2.1 apresenta a estrutura da mensagem codificada no protocolo FIX. Observe que o protocolo FIX não abrange apenas a estrutura da mensagem, mas também estabelece um conjunto abrangente de códigos e convenções padronizados. Esses códigos e convenções são usados para representar dados em campos, incluindo códigos de instrumentos financeiros, códigos de moeda e formatos de data e hora. Ao aderir a estes códigos e convenções padronizados, o protocolo FIX permite uma comunicação eficiente e confiável entre os participantes do mercado financeiro. Isto garante a consistência e a interoperabilidade nas informações trocadas, aumentando a eficácia global do protocolo.

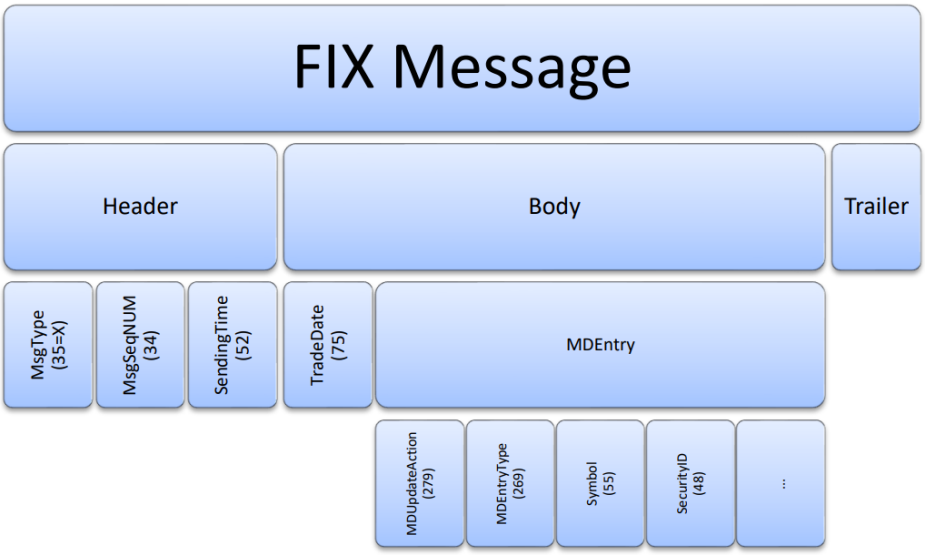


Figura 2.1 – Estrutura de uma mensagem FIX [B3, 2019].

Um exemplo ilustrativo de mensagem FIX é fornecido na Tabela 2.1, mostrando os campos junto com seus valores correspondentes. O campo inicial exibido na Tabela 2.1 é indicado pelo valor numérico 8, seguido pela string “FIX.4.4”. Esta informação inicial reside no cabeçalho da mensagem e pertence à versão do protocolo utilizado.

Tabela 2.1 – Exemplo de mensagem FIX [Darwinex, 2019].

8=FIX.4.4	9=110	35=D	34=3	49=CLIENT
52=20190424-11:19:16.885			56=SERVER	
11=ID	21=3	40=1	54=1	55=EUR/USD
60=20190424-11:19:16.885			10=054	

Outro campo importante é o *MsgType*, que segue a seguinte padronização:

- **D** (*Order Single*): classifica a ordem como ordem limite. Uma ordem com limite é uma diretiva explícita para comprar ou vender um ativo a um preço predeterminado ou a um preço mais favorável. Este tipo específico de ordem estabelece um limite máximo para o preço máximo para uma ordem de compra ou um limite mínimo para o preço mínimo para uma ordem de venda;
- **0** (*Heart Beat*): esta mensagem serve como meio de verificar a conectividade entre sistemas envolvidos na troca de mensagens FIX. Esta mensagem específica é enviada em intervalos regulares para garantir a atividade contínua da ligação e para identificar quaisquer potenciais perturbações na comunicação;
- **1** (*Test Request*): é uma mensagem utilizada para verificar a disponibilidade e o correto funcionamento do sistema de negociação ou aplicação alvo. Ao transmitir uma mensagem “Solicitação de Teste”, o remetente solicita uma resposta do destinatário para validar o status ativo do sistema. Isso garante que o sistema esteja funcionando conforme esperado e confirma sua prontidão operacional.

Contudo, esses valores apresentam variação com base na implementação específica do protocolo FIX, e os participantes do mercado têm flexibilidade para adotar valores personalizados para atender às suas necessidades específicas. Nesse sentido, a Tabela 2.2 apresenta as tags, juntamente com suas descrições correspondentes e valores padrões adotados no setor financeiro.

Tabela 2.2 – Resumo das descrições das tags do protocolo FIX [Darwinex, 2019].

Tag	Tag Description	Value	Value Description
8	BeginString	FIX.4.4	
9	BodyLength	110	
10	CheckSum	54	
11	ClOrdID	ID	
21	HandInst	3	Manual Order
34	MsgSeqNum	3	
35	MsgType	D	Order Single
40	OrdType	1	Market
49	SenderCompID	Client	
52	SendingTime	20190424-11:19:16.885	
54	Side	1	Buy
55	Symbol	EUR/USD	
56	TargetCompID	Server	
60	TransactTime	20190424-11:19:16.885	

2.5.2 UDP

O Protocolo de Datagrama de Usuário (do inglês *User Datagram Protocol* – UDP) é um protocolo de comunicação que pertence à camada de transporte na pilha de protocolos da internet. Sua finalidade é a transmissão de mensagens encapsuladas, como datagramas em pacotes entre *hosts* em uma rede IP. Ao operar em uma rede IP, o UDP não requer uma comunicação prévia para estabelecer canais ou caminhos de dados, adotando um modelo de comunicação simples e não orientado à conexão, caracterizado por uma quantidade mínima de mecanismos protocolares.

O UDP inclui verificações de integridade, através de um *checksum*, aplicadas ao cabeçalho e ao *payload*, porém não assegura a entrega, ordenação ou prevenção contra duplicidade de mensagens. No caso de transmissões que demandam confiabilidade, essas garantias devem ser implementadas no nível da aplicação do usuário.

Este protocolo é particularmente apropriado para contextos nos quais verificações e correções de erros não são imperativas ou são executadas na aplicação. Dessa forma, o UDP evita sobrecarregar o processamento na pilha de protocolos. Em aplicações sensíveis ao tempo, o UDP é preferido, uma vez que a eliminação de pacotes é preferível à espera por pacotes atrasados resultantes de retransmissões, particularmente em sistemas em tempo real.

Os primeiros 8 bytes do pacote compreendem todas as informações necessárias do cabeçalho, enquanto a parte subsequente consiste nos dados. Os campos de número de porta do UDP possuem 16 bits de comprimento cada, sendo utilizados para distinguir diversas funções ou solicitações do usuário tanto na origem quanto no destino do datagrama, mostrado na Figura 2.2.

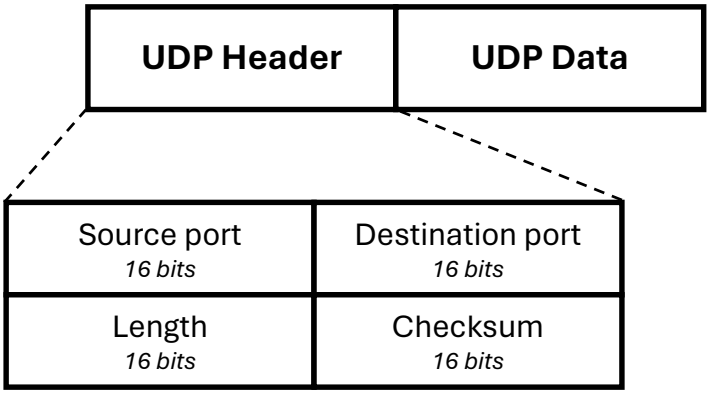


Figura 2.2 – Formato do pacote UDP.

O UDP é empregado em comunicações de requisição e resposta de natureza simples, especialmente quando o tamanho dos dados é reduzido, minimizando assim as preocupações com controle de fluxo e erro. Sua utilização é apropriada em cenários de *multicast*, dado que o UDP suporta a comutação de pacotes. O UDP é empregado em protocolos de atualização de roteamento, como o Protocolo de Informações de Roteamento (do inglês *Routing Information Protocol* – RIP), sendo adotado em aplicações de tempo real que não podem tolerar atrasos irregulares entre as partes de uma mensagem recebida.

2.5.3 TCP

O Protocolo de Controle de Transmissão (do inglês *Transmission Control Protocol* – TCP) é um dos protocolos de comunicação fundamentais que forma a infraestrutura da internet. Com a tarefa de assegurar a transmissão de dados de maneira confiável, ordenada e livre de erros entre computadores e servidores, o TCP opera em conjunto com o IP, que é responsável por identificar os dispositivos na rede.

O TCP é caracterizado como um protocolo orientado à conexão, estabelecendo uma conexão entre os dispositivos antes de iniciar a transmissão de dados. Essa conexão é mantida até que todos os dados tenham sido entregues ou até que a conexão seja devidamente encerrada. O TCP exerce controle sobre o fluxo de dados, ajustando a taxa de envio conforme a capacidade da rede e a disponibilidade do receptor.

No TCP, os dados são divididos em segmentos, os quais são numerados e transmitidos sequencialmente. Cada segmento incorpora um cabeçalho contendo informações

cruciais, tais como origem, destino, número de sequência, número de confirmação, tamanho da janela, *checksum*, e outras opções, como mostra a Figura 2.3. Essas informações ajudam o TCP a verificar se os segmentos foram recebidos de maneira correta, na sequência adequada e sem duplicações. Caso algum segmento seja perdido, corrompido ou sofra atraso, o TCP solicita a retransmissão do segmento ao emissor.

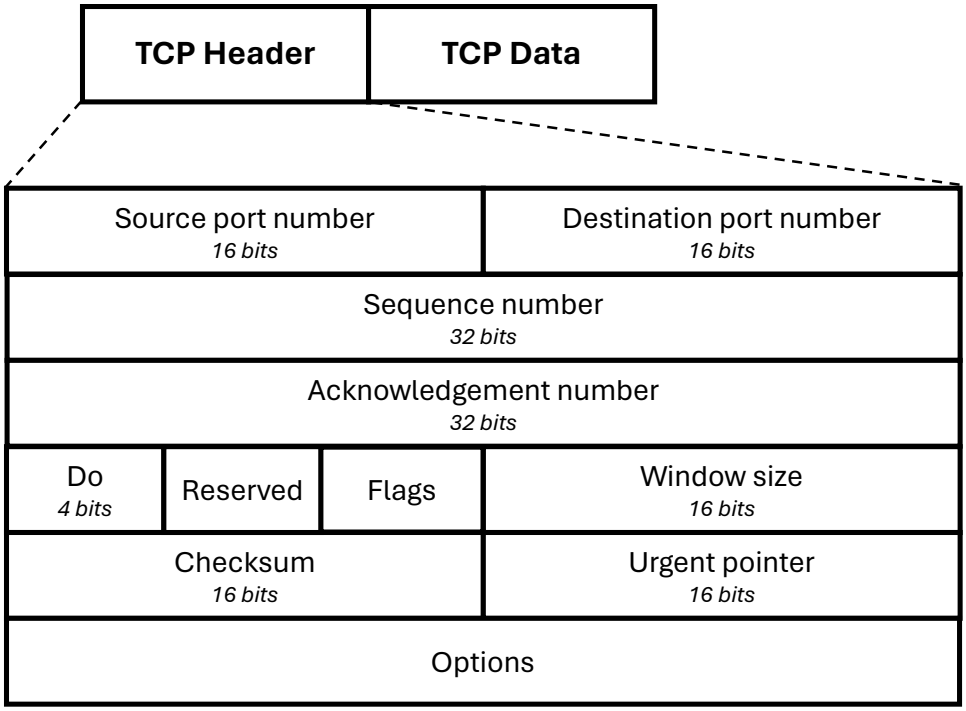


Figura 2.3 – Formato do pacote TCP.

O TCP utiliza portas como meio de identificar os serviços ou aplicações que fazem uso desse protocolo. Cada porta é designada por um número situado entre 0 e 65535, estando associada a um processo específico no dispositivo. Um exemplo é a porta 80 que é reservada para o protocolo HTTP, muito usado na navegação web. Assim, quando um navegador solicita uma página a um servidor, utiliza a porta 80 para estabelecer comunicação com o TCP do servidor.

O TCP, devido à sua capacidade de confiabilidade, controle de fluxo e correção de erros, é extensamente utilizado na internet. Este protocolo é usado por diversas aplicações, tais como SSH, FTP, HTTP, HTTPS, SMTP, IMAP, POP, entre outras.

2.6 Protocolos de Comunicação Específicos da B3

Este seção aborda os protocolos de comunicação específicos da B3. Por exemplo, a B3 com seu novo sistema adotou alguns novos protocolos para a atualização de *feed* de mercado e solicitações de ordens.

2.6.1 Market Data Binary

O *Market Data Binary* (MDB) é baseado no *Unified Market Data Feed* (UMDF), plataforma eletrônica desenvolvida pela B3 para divulgar informações sobre ativos negociados na bolsa de valores.

O UMDF tem como objetivo fornecer dados em tempo real sobre preços, negociações e outras informações relevantes sobre ativos financeiros negociados na bolsa. O novo sistema que está sendo implementado pela B3 prevê a utilização do UMDF, mas em formato binário. A principal discrepância entre UMDF e UMDF binário reside na abordagem de codificação adotada para mensagens de dados de mercado. Enquanto o UMDF emprega o protocolo FIX com codificação FAST, o UMDF Binary usa o protocolo *Simple Binary Encoding* (SBE). O SBE é um protocolo de codificação binário mais contemporâneo e eficiente, caracterizado pela sua simplicidade e agilidade tanto na codificação quanto na decodificação comparado ao FAST [B3, 2023b].

O binário UMDF é igualmente capaz de suportar eventos orientados por ordem. No entanto, o UMDF cobre uma extensa gama de mensagens de dados de mercado, como *Market by Price* (MBP) e *Top of Book* (TOB), que não estão disponíveis no SBE. Além disso, o UMDF Binary opera com base em UDP Multicast para disseminar informações de dados de mercado, enquanto o UMDF utiliza conexões TCP. Resumindo, o UMDF Binary adota um protocolo notavelmente semelhante ao FIX/FAST, mas com uma abordagem binária, onde os campos serão separados e codificados em formato binário. O mapeamento dos campos do protocolo FIX/FAST para o protocolo SBE é mostrado na Figura 2.4.

Observe que a estrutura da mensagem SBE permanece bastante semelhante à da mensagem FIX/FAST, como fica evidente ao comparar a Figura 2.5 com a Figura 2.1. Ele é composto por um cabeçalho contendo campos essenciais relacionados ao tamanho da mensagem e outras informações, e um corpo onde o SBE é aplicado seguindo as orientações apresentadas na Figura 2.4.

Para fins ilustrativos, a Figura 2.6 representa um exemplo de mensagem codificada usando o UMDF Binary [B3, 2023b]. Junto com isso, a Tabela 2.3 apresenta a mensagem decodificada. Observe que a Tabela 2.3 é muito semelhante a Tabela 2.2, mas não existe mais o campo TAG e o valor do campo é codificado em binário.

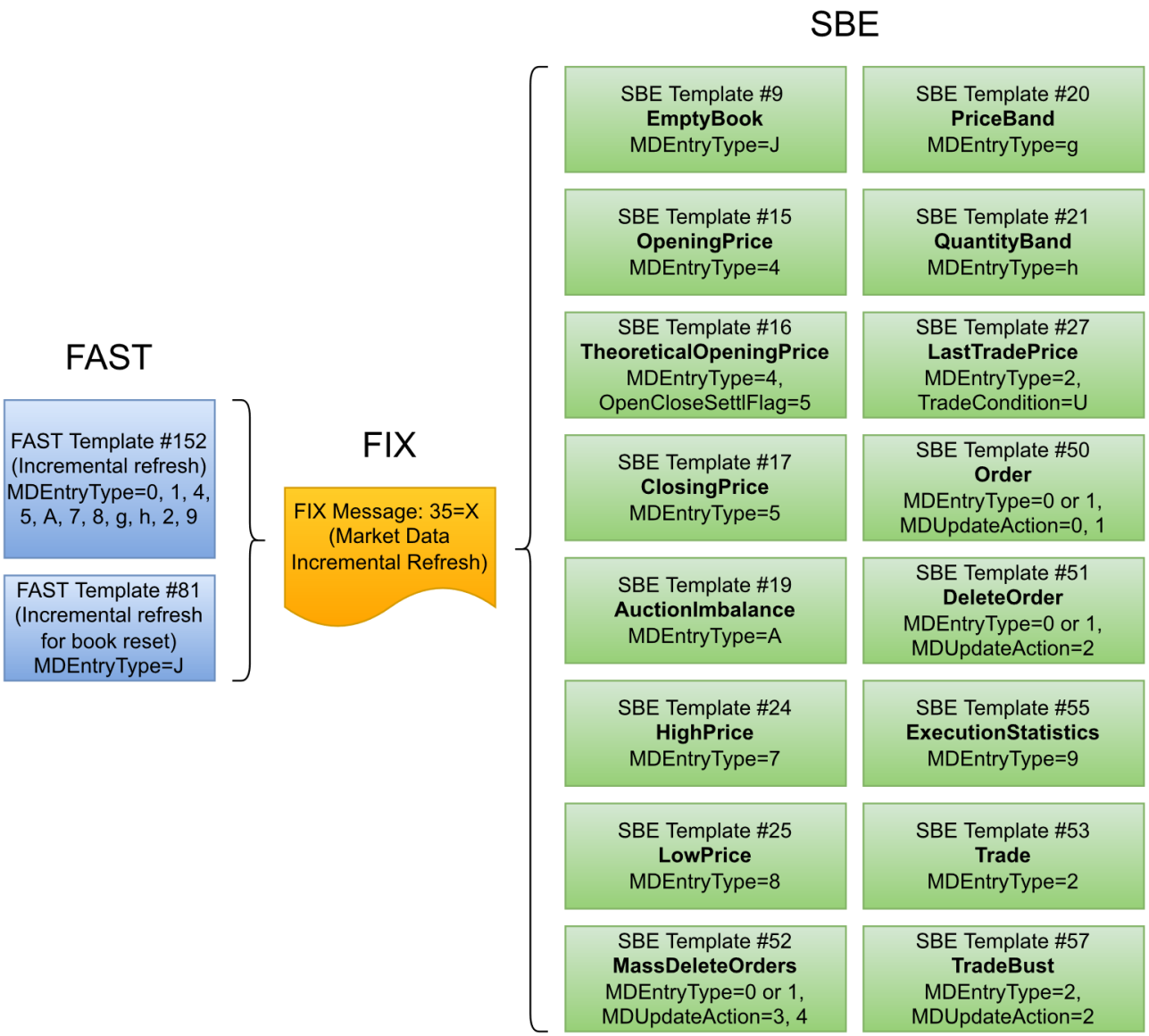


Figura 2.4 – Mapeamento de mensagens FIX/FAST para mensagens FIX/SBE [B3, 2022].

Figura 2.5 – Estrutura da mensagem SBE [B3, 2022].

```
39 00 50 EB 06 00 09 03 01 00 00 00 49 42 4F 56 00 00 12 00
02 50 45 54 52 34 00 CC A5 9B 06 00 00 00 00 18 55 17 03 56
41 4C 45 35 00 D0 9B CD 02 00 00 00 00 9D 01 2E 01
```

Figura 2.6 – Exemplo de *UMDF Binary*.

Tabela 2.3 – Exemplo de mensagem de decodificação do *UMDF Binary* [B3, 2023b].

#	Field	Value (Bytes)	Decoded Value
	messageLength	39 00	57
	encodingType	50 EB	60240
	blockLength	06 00	6
	templateID	09 03	777
	schemaID	01 00	1
	schemaVersion	00 00	0
	symbol (55)	49 42 4F 56 00 00	“IBOV”
	noUnderlyings (711): GroupSizeEncoding.blockLength	12 00	18
	noUnderlyings (711): GroupSizeEncoding.numInGroup	02	2
1	underlyingSymbol (311)	50 45 54 52 34 00	“PETR4”
1	indexPct (6919)	CC A5 9B 06 00 00 00 00	1.10863820 (00 00 00 00 06 9B A5 CC = 110,863,820; divide by 100,000,000)
1	indexTheoreticalQty (37021)	18 55 17 03	51,860,760 (03 17 55 18 = 51860760)
2	underlyingSymbol (311)	56 41 4C 45 35 00	“VALE5”
2	indexPct (6919)	D0 9B CD 02 00 00 00 00	0.4702920 (00 00 00 00 02 CD 9B D0 = 47,029,200; divide by 10,000,000)
2	indexTheoreticalQty (37021)	9D 01 2E 01	19,792,285 (01 2E 01 9D = 19792285)

2.6.2 Order Entry Binary

O *Order Entry Binary* (OEB) é uma interface de gerenciamento de ordens do sistema de negociação PUMA, bem semelhante ao MDB (Sessão 2.6.1). Contudo, ao invés de ser utilizado para informar sobre o *feed* do mercado, o OEB é utilizado pelo participante para enviar as requisições ou pela B3 para enviar os relatórios sobre as ordens. O *Order Entry Binary* é baseado no protocolo FIX 4.4. Assim como o MDB, ele usa o formato *Simple Binary Encoding* (SBE) para codificar e decodificar os dados, o que reduz o tamanho das mensagens e aumenta a velocidade de transmissão. Além disso, o OEB permite a entrada, o cancelamento, a alteração e a consulta de ordens, bem como o recebimento de relatórios de execução. O OEB suporta dois segmentos de mercado da B3, derivativos e ações. Ele usa portas para identificar os serviços ou aplicações que estão usando o protocolo. Além disso, o OEB requer uma certificação prévia para se conectar ao *gateway* de entrada de ordens da B3. As características do *Order Entry Binary* incluem:

- **Protocolo Binário**: onde os dados são representados por sequências de bits, sem caracteres humanamente legíveis. Isso torna o protocolo mais compacto e eficiente, mas também mais difícil de depurar e analisar;
- **Orientado a Sessão**: diferente do MDB, ele estabelece uma conexão lógica entre o cliente e o servidor antes de enviar as mensagens. Essa conexão é mantida até que seja encerrada explicitamente ou por algum motivo de falha. O protocolo usa um mecanismo de *heartbeat* (isto é, um envio periódico de mensagens ou pacotes de

dados entre um servidor e um cliente) para verificar se a conexão está ativa e um mecanismo de retransmissão para recuperar mensagens perdidas;

- **Protocolo Assíncrono**: o que significa que o OEB não espera uma resposta imediata para cada mensagem enviada. Isso permite que o cliente e o servidor possam enviar e receber múltiplas mensagens simultaneamente, sem bloquear o fluxo de comunicação. Vale ressaltar que o protocolo usa um campo de sequência para identificar e ordenar as mensagem;
- **Baseado em Eventos**: o que significa que o OEB envia e recebe mensagens de acordo com as ações realizadas pelos participantes do mercado. Por exemplo, quando um cliente envia uma ordem, ele recebe uma mensagem de confirmação, uma mensagem de rejeição ou uma mensagem de execução, dependendo do resultado da ordem.

O *Order Entry Binary* usa o formato SBE, o mesmo usado no MDB. O SBE define uma estrutura de dados baseada em blocos, que contém um cabeçalho, um corpo e um rodapé. Cada bloco pode ter um ou mais campos, que podem ser de tipos primitivos, compostos ou enumerados. O SBE também define um esquema XML para descrever a estrutura de dados e facilitar a geração de código.

As principais mensagens utilizadas no formato SBE, são:

- ***NewOrderSingle***: é uma mensagem usada para enviar uma nova ordem ao sistema de negociação. Ela contém campos como identificador da ordem, símbolo do instrumento, lado da ordem, tipo da ordem, preço da ordem, quantidade da ordem, validade da ordem, entre outros;
- ***ExecutionReport***: é uma mensagem usada para informar o status ou o resultado de uma ordem ao cliente. Ela contém campos como identificador da ordem no sistema, identificador da execução, tipo da execução, status da ordem, preço da última execução, quantidade da última execução, quantidade acumulada, quantidade restante, entre outros;
- ***OrderCancelRequest***: é uma mensagem usada para solicitar o cancelamento de uma ordem existente. Ela contém campos como identificador original da ordem, identificador do pedido de cancelamento, símbolo do instrumento, lado da ordem, entre outros.

3. TRABALHOS RELACIONADOS

A literatura científica envolvendo o mercado financeiro é pequena, isso porque dificilmente as empresas revelam suas implementações ou divulgam alguma abordagem que utiliza um algoritmo para melhorar sua estratégica de negociação. Tais informações podem ser o diferencial para uma empresa ganhar ou perder dinheiro no mercado financeiro. O que se sabe são os fatores críticos que fazem uma empresa alcançar uma maior rentabilidade, que são principalmente a latência do sistema de negociação e a eficácia da estratégia concebida pelos participantes no mercado financeiro. Neste contexto, este capítulo apresenta estudos que abordem tanto a latência como os tipos de algoritmos usados no mercado financeiro.

3.1 Latência no Mercado Financeiro

A preocupação com a latência do sistema de negociação e uso de sistemas baseados em FPGA é uma tendência recente em negociação de alta frequência. No entanto, como é um tema direcionado principalmente às empresas, existem apenas alguns trabalhos disponíveis na literatura que cobrem esse assunto.

Por exemplo, Funie *et al.* [Funie et al., 2015] propõem uma abordagem de aceleração reconfigurável para melhorar o processo de avaliação nas estratégias de negociação. Os autores demonstram a eficácia de sua abordagem usando uma estratégia de negociação de referência, e os resultados mostram uma aceleração significativa em comparação às implementações tradicionais de software. Este trabalho destaca o potencial do uso de hardware reconfigurável para acelerar os aplicativos financeiros.

Tang *et al.* [Tang et al., 2016] propõem uma arquitetura de processamento de dados de mercado que pode superar o problema de dependência de dados financeiros que existe nos sistemas tradicionais. A arquitetura proposta usa uma série de módulos de hardware, como FPGAs e GPUs, para processar dados financeiros de maneira pipeline, alcançando maior desempenho e menor latência do que as soluções tradicionais baseadas em software. Os resultados da avaliação mostram que a arquitetura proposta pode processar grandes quantidades de dados financeiros com baixa latência, tornando-o adequado para aplicativos de negociação de alta frequência.

Um trabalho mais recente é proposto por Klaisoongnoen *et al.* [Klaisoongnoen et al., 2022], que discutem o desenvolvimento de uma arquitetura de aceleração de hardware para o cálculo da opção *Greeks* usando FPGAs. A arquitetura proposta fornece uma solução rápida e eficiente em termos de energia para a análise de riscos em tempo real dos derivativos, essencial para as instituições financeiras. Os autores implementaram a arquite-

tura proposta nos *FPGAs Xilinx* e *Intel* e alcançaram melhorias significativas na velocidade e na eficiência de energia em comparação com as abordagens baseadas em software.

Diferente dos trabalhos anteriores, algumas abordagens se concentram em implementações híbridas. Yi-chieh *et al.* [Kao et al., 2022] descrevem o desenvolvimento de um sistema HFT baseado em FPGA que atinge baixa latência (isto é, 433 ns) para uma comunicação *Ethernet* de 10 *gigabit*. O sistema usa uma abordagem de co-design de *hardware/software* e emprega um protocolo de comunicação personalizado e um pipeline de processamento de dados para transmissão e processamento eficientes de dados. Os resultados experimentais mostram que o sistema proposto supera uma solução baseada em software em latência e rendimento, demonstrando o potencial de sistemas baseados em FPGA para aplicações de HFT.

Fu *et al.* [Fu et al., 2017] propõem um *design* de tabela híbrida que pode gerar dados de mercado financeiro com precisão no nível de nanossegundos. Esse *design* combina abordagens baseadas em *software* e *hardware* para obter geração de dados de alto desempenho e baixa latência. O *design* da tabela híbrida é avaliado por meio de experimentos, e os resultados mostram que ele supera as soluções existentes em termos de precisão, velocidade e flexibilidade.

Leber *et al.* [Leber et al., 2011] também propõem uma arquitetura para sistemas HFT que combinam *hardware* e *software* para obter alto desempenho e baixa latência. O FPGA é usado para implementar algoritmos críticos, como o interpretador dos dados do mercado, enquanto o *software* lida com tarefas menos críticas de latência, como a manutenção do livro de ofertas. O sistema resultante atinge uma latência significativamente menor, isto é, uma atraso 4x menor em comparação com um sistema puramente baseado em *software*.

Boutros *et al.* [Boutros et al., 2017] emploram os benefícios do uso de sistemas baseados em FPGA para negociação de alta frequência através do uso de síntese de alto nível. Os autores propõem uma abordagem eficiente e simplificada para a criação de sistemas de negociação baseados em FPGA que podem fornecer baixa latência, alta taxa de transferência e flexibilidade para atender às demandas dos mercados financeiros modernos.

Vale ressaltar que a latência de comunicação do FPGA também é uma preocupação de alguns trabalhos. Wang *et al.* [Wang et al., 2020] propõem uma nova abordagem para acelerar a comunicação com a rede. A abordagem envolve o uso de placas de interface de rede inteligentes (*SmartNICs*), o que reduz a carga na CPU e melhora o desempenho geral da rede. O artigo apresenta resultados experimentais que demonstram a eficácia da abordagem proposta em termos de latência e taxa de transferência.

Tan *et al.* [Tan et al., 2021] propõem uma arquitetura pipeline de alto desempenho para a aceleração da classificação de pacotes em unidades de processamento de dados (DPUs). A arquitetura proposta combina paralelismo de granularidade fina e grossa para

obter alta taxa de transferência e baixa latência, e é baseada em um algoritmo de árvore de decisão com uso eficiente de memória para classificação de pacotes. A arquitetura proposta é avaliada usando um conjunto de dados de referência, e os resultados mostram que essa arquitetura pode atingir uma taxa de transferência superior a 250 Gbps para pacotes Ethernet pequenos de 64 bytes, o que significa uma alta taxa de transferência de classificação e baixa latência em comparação com as soluções existentes.

Gao *et al.* [Gao et al., 2020] propõem uma arquitetura que suporta multissessões TCP de forma escalável, a fim de melhorar o desempenho de aplicações sensíveis à latência. A solução proposta usa uma arquitetura hierárquica para otimizar o processamento de várias sessões, alcançando uma latência de 262 ns para o recebimento e aproximadamente 180 ns para envio de um pacote de 48 bytes de carga útil (*payload*). Ademais, as latências crescem linearmente com a quantidade de dados a uma taxa de 12,8 ns a cada 8 bytes. Esses resultados experimentais mostram que a solução proposta supera as implementações tradicionais e ainda pode lidar com muitas sessões TCPs com eficiência.

Continuando na mesma direção, Huang *et al.* [Huang et al., 2022] apresentam um *design* de *hardware all-in-one* (AIOC) para sistemas de negociação de mercado financeiro, que integra várias funcionalidades como interfaces de rede, FPGAs e memórias em um único cartão. O *design* da AIOC visa melhorar o desempenho e reduzir a latência dos sistemas de negociação financeira, minimizando o atraso de transmissão de dados e o tempo de processamento, permitindo assim uma execução mais rápida de estratégias de negociação de alta frequência. Os resultados experimentais mostram que o sistema baseado em AIOC atinge uma latência baixa e estável, fornecendo uma solução de alto desempenho para as aplicações voltadas ao mercado financeiro.

Oliveira e Bonato [Oliveira and Bonato, 2023] apresentam um componente de *hardware* baseado em FPGA que implementa um decodificador de mensagens FAST para acessar dados de livro de ofertas usando um template da B3, a bolsa de valores do Brasil. Primeiramente, este trabalho desenvolve um decodificador/codificador do protocolo FIX/FAST em *software* para fins de validação. Em seguida, o componente foi desenvolvido em *hardware* para acelerar o processo de decodificação e codificação da mensagem e ter uma vantagem na corrida do mercado. O componente de hardware foi implementado em uma placa FPGA Stratix V e testado com logs de mensagens reais da B3. Os resultados mostram que o componente foi capaz de decodificar mensagens com latência média de 0,72μs e vazão de 1,4M mensagens FAST por segundo. Vale ressaltar que o componente também apresentou baixo consumo de recursos da FPGA, usando apenas 3147 blocos lógicos (*LookUp Table* – LUT) e 354 registradores (*flip-flop* – FF).

Outro ponto relevante em relação a latência no mercado financeiro é o entendimento das características únicas dos sistemas de negociação. Nesse sentido, Kohda *et al.* [Kohda and Yoshida, 2021] examinam as características da negociação de alta frequência e seu potencial impacto nos mercados financeiros. Ele fornece uma visão geral das

estratégias de HFT, a tecnologia e a infraestrutura usadas na HFT e o impacto da HFT na liquidez, volatilidade e eficiência do mercado. Além disso, o artigo revisa os modelos de previsão de HFT e discute os desafios de sistemas HFT.

3.2 Algoritmos aplicados no Mercado Financeiro

Esta seção apresentará alguns trabalhos que abordam a utilização de algoritmos para previsão de informações voltadas ao mercado de *commodities* e futuros. He e Wen [He and Wen, 2019] apresentam uma abordagem para prever o estado sem risco em negociações de *commodities* aplicando técnicas de aprendizado de máquina. Note que o risco de uma arbitragem impacta diretamente na quantidade de contratos que um participante (cliente) pode negociar no mercado, onde a bolsa têm que retirar margem dos clientes que cheguem no caso de risco máximo. A abordagem apresentada neste trabalho integra dados que vão desde a cotação do contrato até os parâmetros do contrato. Os resultados mostram que essa abordagem supera os métodos existentes, além de ser um complemento aos mesmos, uma vez que ao contrário dos modelos tradicionais, esta técnica explora a gestão do risco de arbitragem sob a ótica do risco mínimo.

Já Usha *et al.* [Usha et al., 2019] apresentam um agente para automatizar a negociação de uma determinada mercadoria ou moeda em um mercado simulado com os objetivos de maximizar retornos e minimizar perdas para o *trader*. O modelo aprende com as tendências dos dados históricos do mercado e é capaz de comprar, vender ou manter uma negociação em uma determinada instância. O modelo é validado executando o agente em dados de mercado não vistos de um período posterior e os retornos gerados são analisados.

Ramakrishnan *et al.* [Ramakrishnan et al., 2017] usam técnicas de aprendizado de máquina para prever a taxa de câmbio da Malásia com base nos preços de *commodities*. Este trabalho compara três técnicas de aprendizado de máquina: *Support Vector Machine* (SVM), Redes Neurais e *RandomForest*. Os resultados experimentais demonstram que a *RandomForest* é comparativamente melhor que a SVM e as redes neurais, em termos de precisão e desempenho. Além disso, este trabalho revela que os preços das matérias-primas específicas da Malásia (petróleo bruto, óleo de palma, borracha e ouro) são os fortes parâmetros dinâmicos que influenciam a taxa de câmbio da Malásia. Com base nesses resultados, é possível criar estratégias de negociação mais eficiêntes.

Suman *et al.* [Suman et al., 2022] desenvolveram um modelo de algoritmo *Long Short-Term Memory* (LSTM) para prever o preço de mercado de *commodities*, como ouro, alumínio e soja. Este trabalho utilizou um conjunto de dados de acesso gratuito para mercados de *commodities* com preços de abertura, máximo, mínimo e de fechamento a partir de dados históricos. O objetivo é que *traders* possam usar esse modelo de previsão do

mercado de commodities para obter ganhos monetários, mantendo ao mesmo tempo um ambiente sustentável no mercado de matérias-primas.

Outro trabalho que utiliza o algoritmo LSTM é apresentado por Ke*et al.* [Ke et al., 2023], que propõem um método de previsão de *commodities* aplicado à bolsa de mercadorias e futudos da China. Inicialmente, este método aplica nos preços futuros das *commodities* o algoritmo *ensemble empirical mode decomposition* (EEMD). O resultado obtido e o índice de fractal adaptativo de Hurst calculado usando dados intradiários do mercado de *commodities* são as entradas do modelo LSTM que analisa a correlação com o mercado externo para detectar mudanças nas condições de mercado. Os resultados mostram que este método tem um melhor desempenho preditivo em comparação com outros modelos de aprendizagem profunda.

Por outro lado, Sonare *et al.* [Sonare et al., 2023] exploram a eficácia de modelos de algoritmos de aprendizado de máquina na previsão do preço do Bitcoin, analisando um conjunto diversificado de dados históricos. Este trabalho compara o desempenho dos algoritmos de aprendizado de máquina em termos de precisão para descobrir os melhores algoritmos para previsão de preços de Bitcoin de curto e longo prazo.

Já o trabalho apresentado por Hu [Hu, 2023] tem o objetivo de modelar e prever o preço de mercado de futuros de *commodities* e descobrir estratégias de investimento mais lucrativas para negociação de futuros através de modelos de aprendizado de máquina. Os resultados mostram que na regressão dos mínimos quadrados, os preços futuros passados são o fator mais influente na previsão, e que escolher os contratos futuros com os maiores retornos diários previstos para negociação é uma estratégia de investimento relativamente lucrativa.

3.3 Discussão

A literatura apresentada tem dois focos, a melhora da latência dos sistemas de negociação, e a utilização de algoritmos (majoritariamente de aprendizado de máquina) para a melhora das estratégias de negociação utilizadas no mercado financeiro. Por um lado, os trabalhos da literatura sobre latência visam implementações específicas, muitas vezes em FPGAs, que prometem melhorar o sistema de negociação de um participante. Contudo, pouco é aplicado de modo geral a outros participantes. Por outro lado, os estudos em relação aos algoritmos aplicados no mercado financeiro, visam principalmente a previsão do valor de alguma *commodity* para que se apliquem estratégias de negociação que tragam lucro financeiro ao participante do mercado. Neste sentido, tais estudos são complementares ao apresentado nesta dissertação de mestrado, e que podem ser aplicados em conjunto para uma melhor aplicação de uma estratégia de negociação.

Desta forma, diferente da literatura revisada, este trabalho visa desenvolver um ambiente de simulação que permita que os desenvolvedores do mercado financeiro apliquem e testem suas estratégias em um ambiente rápido e atualizado, incorporando o novo protocolo que a B3 está implementando no sistema PUMA. Essa iniciativa se concentra em fornecer um ambiente que se alinha às métricas atualizadas do mercado financeiro brasileiro, permitindo uma avaliação eficiente e trazendo suporte para aplicação das estratégias de negociação.

4. AMBIENTE DE SIMULAÇÃO DO SISTEMA PUMA

Para obter vantagem competitiva no mercado financeiro, a consideração de diversos fatores na criação de cada estratégia de negociação revela-se crucial. Neste contexto, este capítulo descreve o desenvolvimento de um ambiente de simulação para o mercado financeiro baseado na evolução do sistema de negociação brasileiro que está em estágio de implantação (*PUMA Trading System* – Sessão 2.3.1). O ambiente de simulação proposto visa ser utilizado em uma fase pré-mercado, o que permitirá avaliações futuras de estratégias de mercado de diferentes *players*, além de disseminar conhecimento e possibilitar que mais *startups* participem deste mercado.

4.1 Visão Geral dos Blocos do Sistema de Negociação

A Figura 4.1 mostra a nova estrutura prevista para o sistema de negociação PUMA. Para o desenvolvimento do ambiente de simulação, foi reproduzido uma infraestrutura semelhante a desenvolvida pela B3, reproduzindo cada bloco com suas funcionalidades.

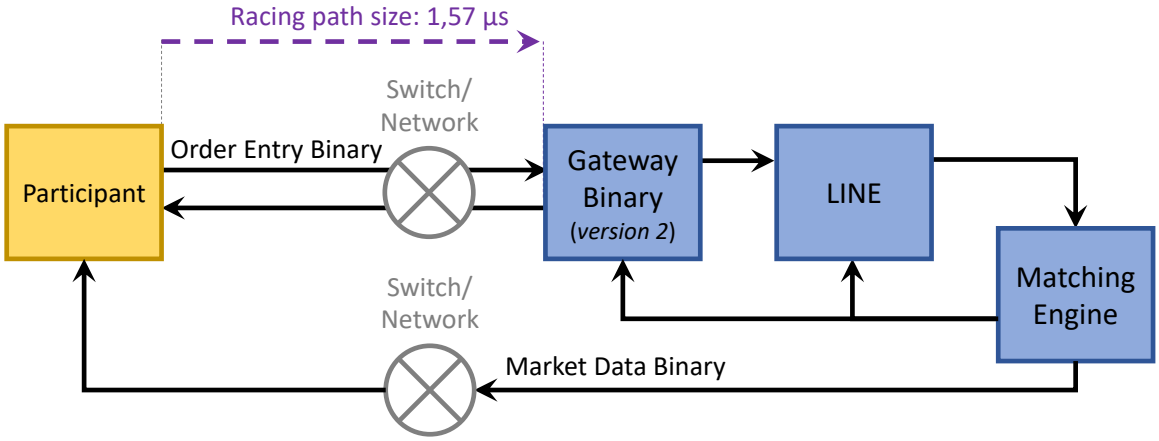


Figura 4.1 – Nova estrutura prevista para o sistema de negociação PUMA.

4.1.1 Participante

O participante, mostrado à esquerda da Figura 4.2, representa as corretoras (do inglês *brokers*), empresas e *startups* que operam na bolsa. Note que as pessoas físicas podem operar também e serem consideradas participantes na Figura 4.2, mas somente através de um dos participantes primários citados.

Na Figura 4.2 os participantes fazem uso do serviço de *co-location* da B3. Este serviço possibilita aos contratantes a hospedagem completa de sua infraestrutura tecnoló-

gica, podendo utilizá-la não só para acessar as plataformas eletrônicas da B3, mas também os serviços e plataformas tecnológicas de outros contratantes através de interconexões (*cross connections*), reduzindo consideravelmente o custo total para os contratantes destes serviços e garantindo uma menor latência do mercado para o *PUMA Trading System*.

O participante utiliza *switches* Ethernet para se comunicar com o *gateway* através do protocolo *Order Entry Binary* (Sessão 2.6.2), que também é utilizado pelo *gateway* para responder a um participante específico. Já para o participante receber os *feeds* do mercado, também são utilizadas *switches* Ethernet para enviar as mensagens. Contudo, o protocolo utilizado é o *Market Data Binary* (Sessão 2.6.1).

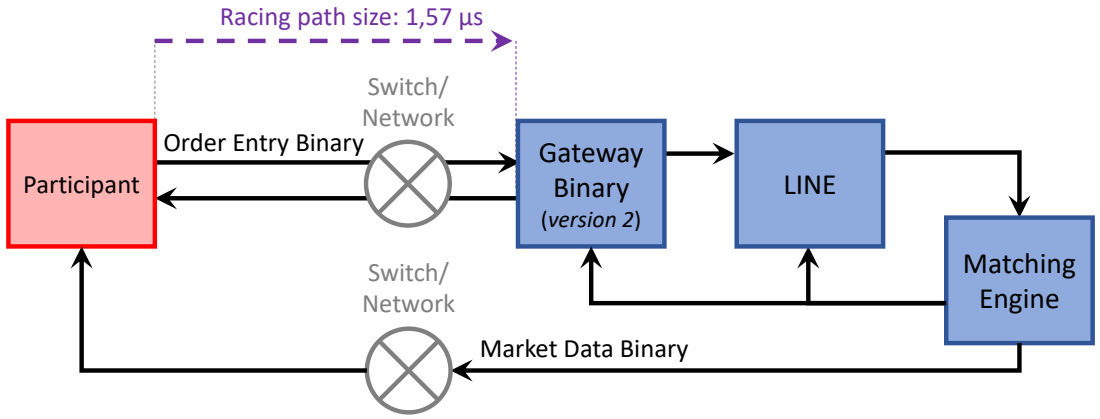


Figura 4.2 – Bloco dos participantes.

4.1.2 Gateway

O *gateway* binário em destaque na Figura 4.3 é a interface de comunicação entre os participantes e o sistema de negociação. O *gateway* recebe as ordens de entradas, e o recebimento de dados do mercado de forma mais rápida e segura. Ele também é responsável por fazer a marcação de tempo nas ordens de entrada, a fim de que o sistema identifique o momento exato em que uma determinada ordem chegou. Desta forma, o *order book* pode fazer a ordenação por ordem de chegada de cada requisição. Vale ressaltar que o *gateway* mostrado na Figura 4.3 utiliza o protocolo *Order Entry Binary* para se comunicar com o participante. Contudo, como previamente descrito na Sessão 2.6.2, o *gateway* envia somente mensagens específicas para cada participante, não enviando mensagens em *broadcast* para todos os participantes.

A Figura 4.4, mostra como funciona a comunicação do participante com o *Gateway*, onde é necessário estabelecer uma conexão entre as duas partes, a confirmação dessa negociação, a troca de mensagens em nível de aplicação, que acontece durante todo o dia de operação, e por final o término da conexão.

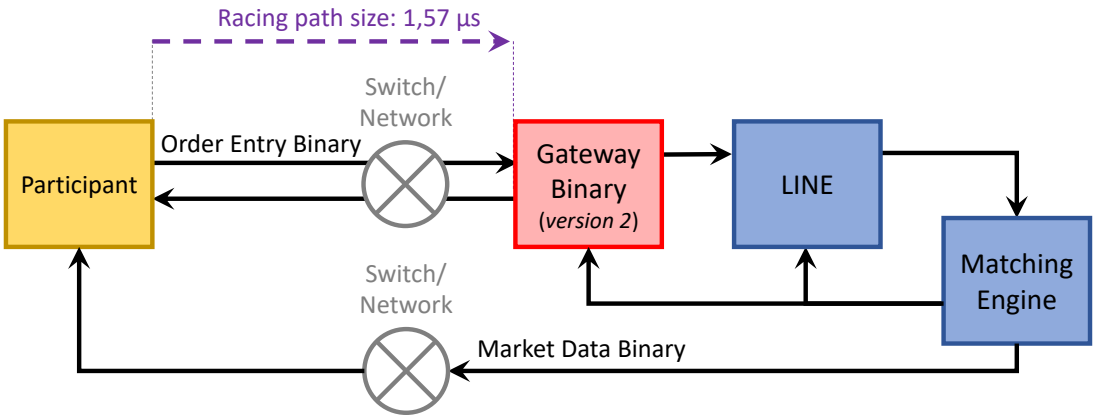


Figura 4.3 – Bloco *Gateway*.

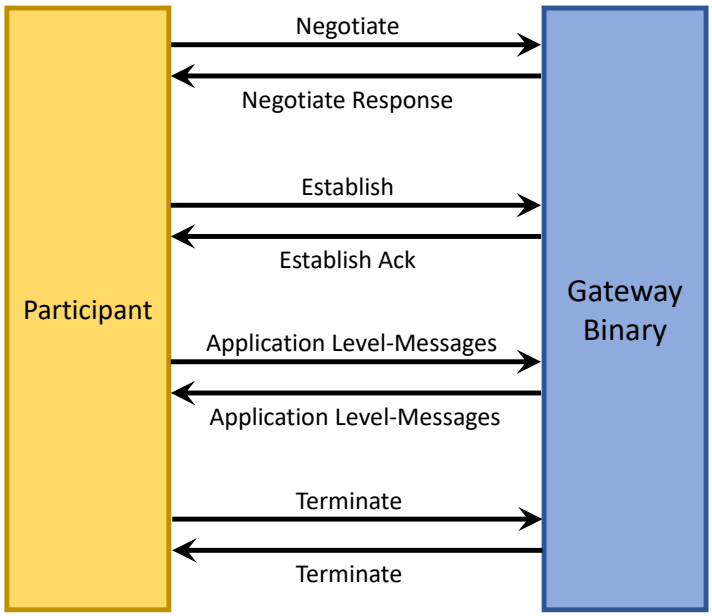


Figura 4.4 – Linha de tempo de comunicação entre participante e Gateway.

4.1.3 Line

O bloco *Line* mostrado na Figura 4.5 é uma ferramenta de gerenciamento de risco de pré-negociação. Esta ferramenta faz com que as corretoras possam estabelecer limites de negociação para clientes específicos, os "*heavy users*", ou seja, o *Line* verifica se o cliente pode operar (comprar ou vender) o número de ordens enviadas e se podem operar aquela determinada mercadoria (por exemplo, petróleo, milho, soja), verificando histórico, limites, entre outros registros.

O *Line* também monitora as atividades de negociação em tempo real, garantindo que as operações sejam realizadas dentro dos parâmetros de risco pré-estabelecido. Além disso, o *Line* garante que as operações estejam em conformidade com as regulamentações do mercado financeiro, evitando penalidades e infrações.

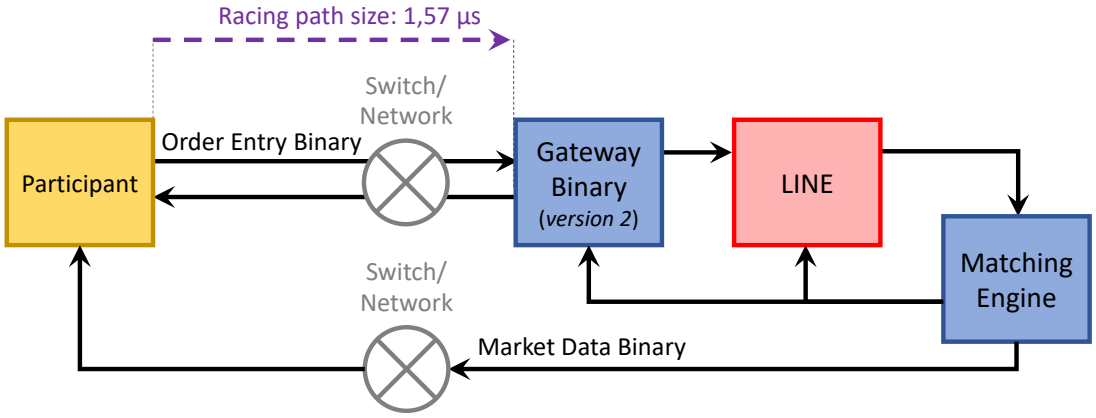


Figura 4.5 – Bloco *Line*.

4.1.4 Matching Engine

O *Matching Engine* é um dos principais blocos do *PUMA Trading System*, o qual deve lidar com o processamento das operações de negociação. O *Matching Engine* é responsável por garantir a execução rápida e eficiente das ordens, sendo essencial em um ambiente de sistema de negociação, onde cada milissegundo é crucial para as operações.

As principais características do *Matching Engine* são garantir o processamento e executar as ordens com alta velocidade e precisão, normalmente executado por um componente chamado de *Order Book* (Sessão 4.1.5); garantir o fluxo de mensagens do *Market Data* (Sessão 4.1.6), a fim de que todos os participantes recebam as informações atualizadas em tempo real; e ter uma boa escalabilidade para lidar com alto volume de negociações sem comprometer o desempenho.

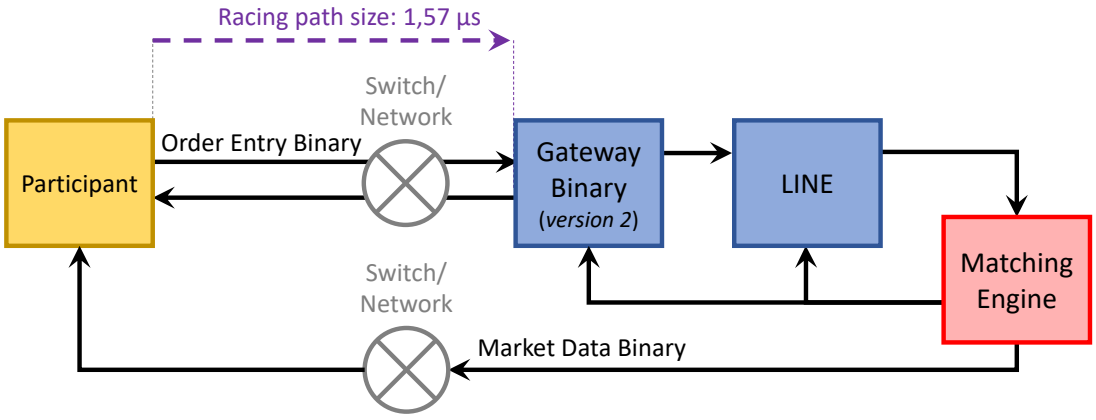


Figura 4.6 – Bloco de *Matching Engine*, que inclui o *Order Book* e o *Market Data*.

4.1.5 Order Book

O livro de ofertas ou livro de ordens (do inglês *Order Book*) é o principal componente do mercado financeiro, que registra e organiza as ordens de compra e venda de ativos financeiros dos participantes. No *PUMA Trading System*, o livro de ofertas está localizado no *Matching Engine* (Sessão 4.1.4). Funciona como um livro-razão, documentando as intenções transacionais dos investidores. No livro de ofertas, as ordens são listadas e organizadas com base na ordem de chegada. Isto significa que o primeiro pedido recebido tem prioridade sobre os pedidos subsequentes. Portanto, a ordem de chegada desempenha um papel crucial na determinação da preferência para executar transações.

Os livros de ofertas são amplamente utilizados nos mercados financeiros, pois proporcionam transparência e eficiência nas negociações. Eles permitem que os participantes visualizem as melhores ofertas de compra e venda disponíveis, bem como a quantidade de ativos ofertados a cada preço. Ao organizar os pedidos por ordem de chegada, o livro de oferta garante um processo justo e imparcial de correspondência entre compradores e vendedores. Além disso, fornece informações importantes aos participantes do mercado, permitindo-lhes tomar decisões informadas com base nas condições de oferta e procura.

4.1.6 Market Data

O *Market Data* é o conjunto de dados do mercado que conta com informações como preços, volumes e outras informações relevantes sobre os ativos negociados. Dentro do bloco *Matching Engine* (Sessão 4.1.4), o *market data* é processado e disseminado em tempo real para os participantes, permitindo com que eles tomem decisões baseadas em informações atualizadas do mercado. Este bloco utiliza o protocolo *Market Data Binary* (Sessão 2.6.1), para disseminar as informações do mercado em *broadcast* para todos os participantes.

O funcionamento do *Market Data* é descrito abaixo:

1. Coleta os dados de todas as negociações realizadas no *PUMA Trading System*, incluindo preços de execução, volumes negociados e ofertas de compra e venda;
2. Estes dados são processados e distribuídos em tempo real para todos os participantes do mercado;
3. Garante a entrega rápida e confiável desses dados, minimizando a latência e maximizando a eficiência das transações realizadas.

4.2 Descrição dos Blocos do Ambiente de Simulação do Sistema de Negociação

Esta seção apresenta a descrição dos blocos do ambiente de simulação do sistema de negociação, se baseando fortemente nos blocos reais do *PUMA Trading System*. No ambiente de simulação proposto, é criado uma base de dados que inclui diversas ofertas codificadas no protocolo *Market Data Binary*, usada no mercado financeiro brasileiro. Então, mensagens serão enviadas aos participantes, o qual será responsável pelo processo de decodificação e codificação da informação por meio de um codificador/decodificador. O resultado desta operação será então enviada para um livro de ofertas (simulando a aplicação de uma estratégia de mercado do participante). Pelo lado que simula a B3, esta listará as ofertas por ordem de chegada, permitindo com que o participante seja como se posicionou na fila. Por sua vez, esta livro de ofertas emula o funcionamento de um produto em uma bolsa de mercadorias e futuros, onde a rapidez na execução das ordens proporciona uma vantagem tanto para compradores como para vendedores, permitindo-lhes ser os primeiros a atuar no mercado financeiro. Ademais, a simulação por um tempo mais longo pode replicar uma situação do mercado, fazendo com que os participantes possam testar diversas estratégias de negociação.

4.2.1 Visão Geral do Simulador

O ambiente de simulação proposto consiste em um processo no qual um codificador/decodificador lê dados de um arquivo. Esses dados são codificados em um protocolo baseado no novo protocolo binário de dados de mercado. O codificador/decodificador assume a responsabilidade de decodificar os dados e posteriormente codificá-los novamente. Esta funcionalidade é particularmente relevante para permitir aos utilizadores deste ambiente de simulação aplicarem as suas estratégias de compra e venda após a decodificação, seguida da codificação dos dados resultantes da aplicação destas estratégias.

A Figura 4.7 apresenta o ambiente de simulação do sistema de negociação proposto. Ele é semelhante a nova estrutura do *PUMA Trading System*, mas contendo três blocos: (1) os blocos dos participantes; (2) um bloco emulando o *gateway* e o *line*; e (3) o bloco do *matching engine*.

4.2.2 Participante

O primeiro bloco do ambiente de simulação é relativo aos participantes. A Figura 4.7 mostra os blocos de participantes em laranja, que se conectam com o bloco azul,

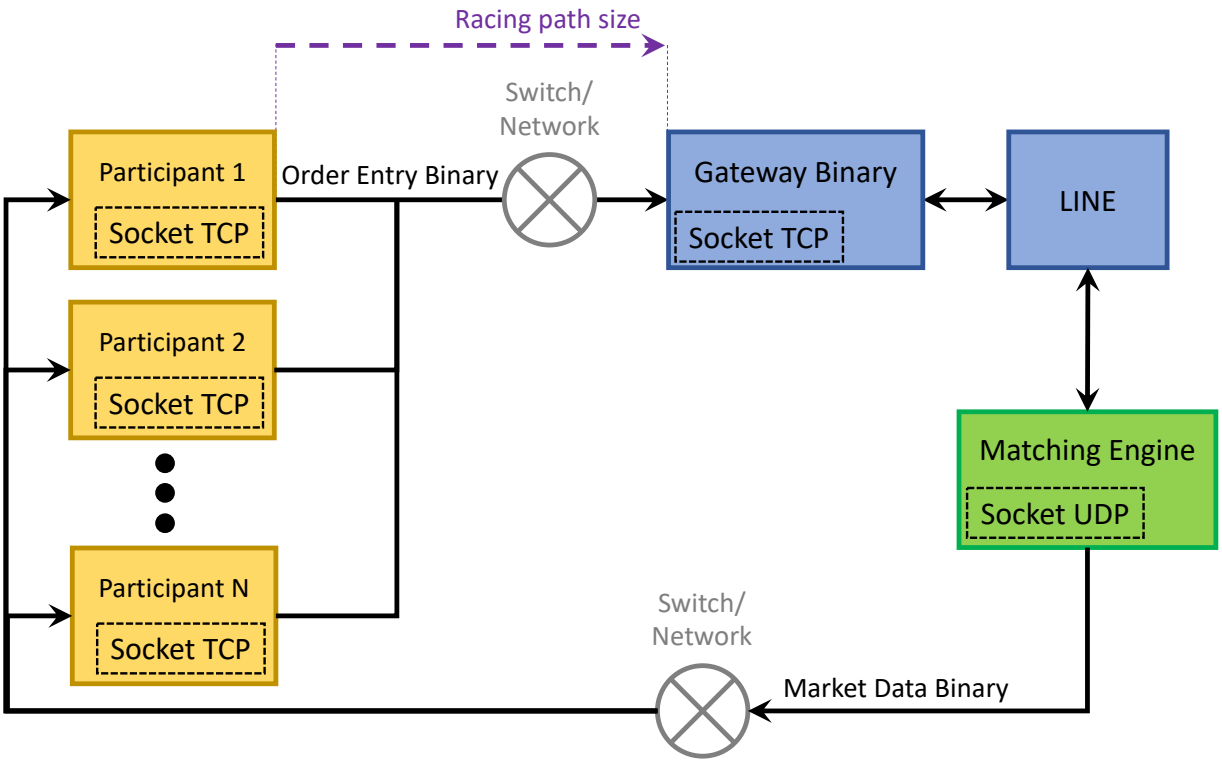


Figura 4.7 – Ambiente de simulação do sistema de negociação proposto.

o qual contém o *gateway* e o *line*. Inicialmente, o participante envia uma requisição de conexão, então o *gateway* avalia se a requisição é válida, e abre uma conexão através de um *socket* TCP, entre cada participante e o próprio *gateway*. O TCP é escolhido nesta parte do simulador pois é um protocolo orientado a conexão, tendo assim garantia de envio e chegada das ordens de compra e venda e também para que possa ser feito um *login* entre as partes no inicio da conexão. Em seguida, após ser realizada esta conexão, o participante abre outra conexão, desta vez UDP, com o *matching engine* e fica aguardando os dados de *feed* do mercado. Nesta parte do simulador é escolhido o UDP, pois a mensagem do *feed* de mercado para o participante não precisa ter garantia de chegada. Além disso, também não é necessário nenhum tipo de solicitação de conexão. Quando estes dados (*market data*) são recebidos pela conexão UDP provindos do *matching engine*, o participante decodifica estes dados, que estão codificados no protocolo MDB (Sessão 2.6.1), faz as alterações necessárias para realizar uma negociação, e então codifica uma mensagem no formato *Order Entry Binary* e envia através do *socket* TCP para o *gateway*.

4.2.3 Gateway/Line

A Figura 4.7 destaca em azul o bloco composto pelo *gateway* e o *Line*. No início do processo, o *gateway* fica aguardando os participantes se conectarem. Então, ele cria uma *thread* para cada participante que envia uma requisição de conexão. Por sua vez, o

Line analisa esta solicitação, e se for válida o *gateway* mantém aberta a conexão. Caso contrário, se a solicitação for inválida, o *gateway* fecha a conexão TCP.

Uma vez que as conexões com os participantes forem realizadas, o *gateway* fica escutando os participantes através de cada *thread* criada com cada participante. A partir do momento em que o *gateway* recebe qualquer dado do participante, ele coloca uma *tag* de tempo (em milissegundos) na mensagem determinando o tempo de chegada da mensagem, a fim de que o *order book* possa posteriormente ordenar todas as ordens dos participantes de acordo com sua ordem de chegada.

4.2.4 Matching Engine

O bloco *Matching Engine* é mostrado em verde na Figura 4.7. Por sua vez, ele é responsável por começar enviando o *feed* de abertura do mercado para os participantes, para que eles possam realizar suas ordens de compra e venda. Após esse primeiro envio de abertura do mercado, o bloco *Matching Engine* fica esperando as mensagens que o *gateway* recebeu e colocou a *tag* de tempo para realizar a organização das ordens. Então o *Matching Engine* envia mensagens do *feed* para os participantes, contendo os "*matches*" de compra e venda, informações de execução de um produto, entre outros.

Vale ressaltar que as mensagens do *market data* são enviadas através de um socket UDP, onde cada participante tem uma porta dedicada. Desta forma, o *market data* é enviando de forma não síncrona para os participantes. Logo, cada vez que o bloco *Matching Engine* vai enviar um novo *feed* de mercado, ele realiza uma randomização das portas configuradas e envia aleatoriamente a informação para cada participante. Isso simula um ambiente real de comunicação, onde cada participante receberá as informações em tempos variados, simulando infraestruturas diferentes, distâncias diferentes entre os participantes (corretora, empresas, etc.), entre outros aspectos que possam influenciar a latência de chegada dos dados de mercado.

4.3 Descrição dos Protocolos Implementados

No simulador proposto temos a utilização de dois protocolos de comunicação provenientes da B3, o *Market Data Binary* (Sessão 2.6.1), para a atualização do *feed* de mercado, e também o *Order Entry Binary* (Sessão 2.6.2), que é usado para os participantes enviarem as ordens para o mercado.

Os protocolos *Order Entry Binary* e *Market Data Binary* são derivados do SBE. Desta forma, os *templates* de ambos são bem semalhantes, o que altera é a ordem dos

campos, e a inclusão ou remoção de alguns campos. Portando, será apresentado alguns exemplos do *Market Data Binary*.

Em relação ao *Market Data Binary*, o ambiente de simulação suporta alguns dos principais *templates* que são utilizados para os clientes realizarem as ordens de compra e venda. Os *templates* suportados são:

- ***ExecutionSumary* [55]**: retransmite as informações execução de um produto (Figura 4.8);
- ***Trade* [53]**: retransmite informações comerciais sobre um produto (Figura 4.9);
- ***OpeningPrice* [15]**: transporta informações resumidas sobre eventos de abertura de sessão de negociação por fluxo de dados de mercado (Figura 4.10);
- ***HighPrice* [24]**: o preço mais alto negociado pelo título no pregão (Figura 4.11);
- ***LowPrice* [25]**: o menor preço negociado pelo título no pregão (Figura 4.12);
- ***DeleteOrder* [51]**: divulga a exclusão de um novo pedido (Figura 4.13);
- ***ExecutionStatistics* [56]**: retransmite informações estatísticas resumidas de execução de um produto (Figura 4.14).

Nas figuras apresentadas abaixo, podemos ver exemplos dos *templates*, o detalhamento de sua estrutura mostrando os bytes de cada *template*, e também exemplos de dados em hexadecimal contidos nos *templates*.

Existem alguns campos principais que se repetem em alguns *templates*, estes são:

- ***SecurityID***: é usado para identificar de forma única um instrumento financeiro, como uma ação ou contrato de derivativos, dentro do sistema;
- ***MDEntryType***: é usado para indicar o tipo de entrada de dados de mercado. Por exemplo, pode indicar se a entrada é um preço de oferta, um preço de venda ou um preço de negociação;
- ***MDEntryPX***: é usado para indicar o preço associado à entrada de dados de mercado. Por exemplo, pode ser o preço de oferta, o preço de venda ou preço de negociação;
- ***MDEntryPositionNo***: é usado para indicar a posição de uma entrada de dados de mercado dentro de um livro de pedidos.

ExecutionSummary [55]	securityID *64 bits*	padding *16 bits*	aggressorSide *8 bits*	padding *8 bits*	lastPx *64 bits*	fillQty *64 bits*	tradedHiddenQty *64 bits*	cxlQty *64 bits*	aggressorTime *64 bits*	rptSeq *32 bits*	mDEntryTimestamp *64 bits*
✓ Example:	4a58774817000000	0000	01	00	400d030000000000	0a00000000000000	0000000000000000	0000000000000000	c7f9b21dac336d17	40000000	01e1511fac336d17

Figura 4.8 – *Execution Sumary* (Template 55).

Trade [53]	securityID *64 bits*	matchEventIndicator *8 bits*	tradingSessionID *8 bits*	tradeCondition *16 bits*	mDEntryPx *64 bits*	mDEntrySize *64 bits*	tradeID *32 bits*	mDEntryBuyer *32 bits*	mDEntrySeller *32 bits*	tradeDate *16 bits*	trdSubType *8 bits*	padding *8 bits*	mDEntryTimestamp *64 bits*	rptSeq *32 bits*
✓ Example:	4a587748 17000000	00	01	0120	a0860100 00000000	0a000000 00000000	0a00 0000	64000000	64000000	514c	00	00	51f7391fac336d17	3a00 0000

Figura 4.9 – *Trade* (Template 53).

OpeningPrice [15]	securityID *64 bits*	matchEventIndicator *8 bits*	mDUpdateAction *8 bits*	openCloseSettlFlag *8 bits*	padding *8 bits*	mDEntryPx *64 bits*	netChgPrevDay *64 bits*	tradeDate *16 bits*	mDEntryTimestamp *64 bits*	rptSeq *32 bits*	padding *16 bits*
✓ Example:	4a587748 17000000	00	00	00	00	a086010000000000	00ca9a3b00000000	514c	51f7391fac336d17	3b000000	0000

Figura 4.10 – *Opening Price* (Template 15).

HighPrice [24]	securityID *64 bits*	matchEventIndicator *8 bits*	mDUpdateAction *8 bits*	tradeDate *16 bits*	mDEntryPx *64 bits*	mDEntryTimestamp *64 bits*	rptSeq *32 bits*
✓ Example:	4a58774817000000	00	00	514c	a086010000000000	51f7391fac336d17	3c000000

Figura 4.11 – *High Price* (Template 24).

LowPrice [25]	securityID *64 bits*	matchEventIndicator *8 bits*	mDUpdateAction *8 bits*	tradeDate *16 bits*	mDEntryPx *64 bits*	mDEntryTimestamp *64 bits*	rptSeq *32 bits*
✓ Example:	4a58774817000000	00	00	514c	a086010000000000	c912591fac336d17	3d000000

Figura 4.12 – *Low Price* (Template 25).

Em relação ao *Order Entry Binary* (Sessão 2.6.2), o ambiente suporta os dois principais pacotes utilizados pelo participante, que são os *templates* de nova ordem e cancelamento de uma ordem.

DeleteOrder [51]	securityID *64 bits*	matchEventIndicator *8 bits*	padding *8 bits*	mDEntryType *8 bits*	padding *8 bits*	mDEntryPositionNo *32 bits*	mDEntrySize *64 bits*	secondaryOrderID *64 bits*	mDEntryTimestamp *64 bits*	rptSeq *32 bits*
✓ Example:	4a58774817000000	00	00	30	00	01000000	0000000000000080	f601000000000000	51f7391fac336d17	3e000000

Figura 4.13 – *Delete Order* (Template 51).

ExecutionStatistics [56]	securityID *64 bits*	matchEventIndicator *8 bits*	tradingSessionID *8 bits*	tradeDate *16 bits*	tradeVolume *64 bits*	vwapPx *64 bits*	netChgPrevDay *64 bits*	numberOfTrades *32 bits*	mDEntryTimestamp *64 bits*	rptSeq *32 bits*
✓ Example:	4a58774817000000	80	01	514c	0a00000000000000	a086010000000000	00ca9a3b00000000	01000000	9305441fac336d17	3f000000

Figura 4.14 – *Execution Statistics* (Template 56).

- ***NewOrderSingle* [102]**: faz a solicitação de uma nova ordem (Figura 4.15);
- ***OrderCancelRequest* [105]**: solicita o cancelamento de uma ordem (Figura 4.16).

NewOrderSingle [102]	ClOrdID *64 bits*	EnteringFirm *32 bits*	EnteringTrader *80 bits*	SenderLocation *8 bits*	OrdTagID *8 bits*	MarketSegmentID *? bits*	Side *8 bits*	TransactTime *64 bits*	OrderQty *64 bits*	SecurityID *64 bits*	TimeInForce *8 bits*	OrdType *8 bits*

Figura 4.15 – *NewOrderSingle* (Template 102).

OrderCancelRequest [105]	ClOrdID *64 bits*	EnteringFirm *32 bits*	EnteringTrader *80 bits*	SenderLocation *8 bits*	TransactTime *64 bits*	MarketSegmentID *8 bits*	SecurityID *64 bits*	Side *8 bits*

Figura 4.16 – *OrderCancelRequest* (Template 105).

4.4 Estudo de caso de uma operação

Esta seção apresenta a descrição de um estudo de caso onde há troca de mensagens entre os blocos para melhor compreensão do funcionamento do simulador. Na Figura 4.17 podemos ver um exemplo passo a passo para que uma transação possa acontecer.

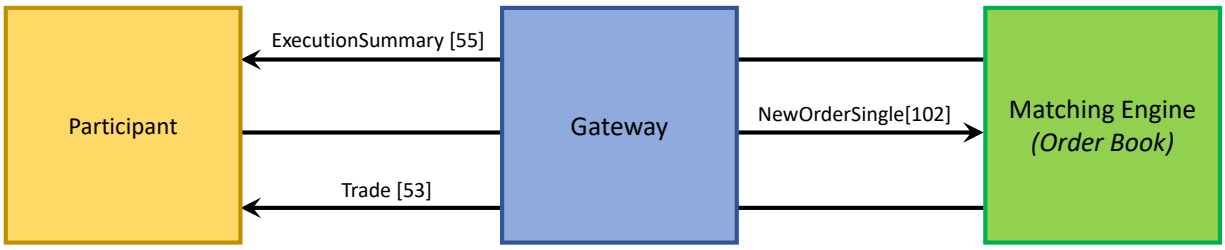


Figura 4.17 – Passo a passo de um exemplo de transação.

Em um primeiro instante, o *Matching Engine* (Sessão 4.1.4) envia o *feed* do mercado através da mensagem *ExecutionSummary [55]*.

Os campos da mensagem *ExecutionSummary [55]* são descritos abaixo:

- *securityID*: ID de segurança que identifica exclusivamente um instrumento (isto é, os ativos disponíveis para os investidores comprar ou vender, no nosso caso, são os contratos) na plataforma PUMA. No caso, se o participante tiver especificado um valor, o sistema PUMA retornará mensagens de definição de segurança apenas para o único instrumento cujo ID de segurança corresponde ao valor especificado. Contudo, se o valor especificado for 48, indica que nenhuma outro valor será necessário para identificar o instrumento;
- *aggressorSide*: indica o lado do agressor, podendo ser 0 para indicar que a negociação não tem agressor, 1 para compra e 2 para venda;
- *lastPx*: preço do último contrato executado;
- *fillQty*: quantidade de todos os contratos executados;
- *tradedHiddenQty*: quantidade total de ordens passivas que não são exibidas ao mercado. Por exemplo, as ordens *iceberg*[1], onde os participantes negociam um grande lote de um determinado produto sem expor todo o lote no mercado de uma só vez;
- *cxlQty*: quantidade total cancelada durante o processo de conciliação (do inglês, *matching process*). Por exemplo, compra e venda de um contrato um mesmo participante;

[1] Os pedidos *iceberg* são pedidos únicos grandes que foram divididos em pedidos com limites menores com o objetivo de ocultar a quantidade real do pedido.

- *aggressorTime*: data/hora de uma ordem agressiva resultante de um processo de conciliação. Por exemplo, em um caso de uma ordem agressiva, este campo refletirá o instante em que esta ordem entra na fila (estilo FIFO, isto é, a primeira ordem que entra é a primeira que sai) no fluxo de entrada de ordens;
- *rptSeq*: representa o valor do *sequence number*;
- *mDEntryTimestamp*: data e hora do instante em que a ordem chegou no bolsa (isto é, *market data entry*).

Em um segundo momento, o participante (Sessão 4.1.1) recebe esse pacote, decodifica, e a partir de alguns dados do pacote *ExecutionSummary* [55], monta o pacote *NewOrderSingle* [102], onde os campos são mostrados abaixo:

- *ClOrdID*: é o principal identificador da ordem do lado do participante. É atribuído inicialmente na entrada da ordem e pode ser posteriormente alterado ao longo do ciclo de vida da ordem em modificações e solicitações de cancelamento. Deve ser único entre todas as ordens ativas em um determinado instrumento enviadas através de uma sessão específica;
- *EnteringFirm*: identifica a corretora;
- *EnteringTrader*: identificador do trader;
- *SenderLocation*: identifica o originador da ordem e a categoria de acesso direto ao mercado (DMA), que é a funcionalidade que permite aos clientes finais, como fundos de investimentos, acessar diretamente a bolsa eletronicamente, sem a necessidade de passar pela infraestrutura física da corretora. O valor deste campo para os *traders* sempre é “BVMF”, já as conexões DMA devem utilizar o valor atribuído pela B3;
- *OrdTagID*: é a identificação da ordem;
- *MarketSegmentId*: identifica o segmento de mercado. Obrigatório para todos os instrumentos negociáveis na bolsa de mercadorias e futuros. Não presente em índices de ações, índices de ETF, BTB e exercício de opções, que são instrumentos não negociáveis;
- *Side*: indica compra ou venda;
- *TransactTime*: tempo de execução/criação da ordem (expresso em UTC);
- *OrderQty*: indica a quantidade de contratos da ordem;
- *securityID*: ID de segurança;

- *TimeInForce*: indica a duração da ordem, por exemplo, 0 indica que a ordem está disponível durante o dia até serem executados ou cancelados, já 3 indica que a ordem requer execução imediata e a quantidade não executada é automaticamente cancelada;
- *OrdType*: é o tipo da ordem. Por exemplo, 2 significa ordens limitados, o qual especifica o pior preço pelo qual a ordem pode ser executada, ou seja, uma ordem para comprar um produto deve estar no preço ou abaixo de um preço declarado, ou para vender um produto a um preço igual ou superior ao valor declarado. Se a ordem não for executada, ela permanecerá no livro de ofertas.

Por fim o *order book* recebe a ordem e monta o pacote *Trade* [53], com a informação dessa nova ordem feita, para o *Matching Engine* eviar como *feed* de mercado para os participantes, os campos do pacote *Trade*, são descritos abaixo:

- *securityID*: ID de segurança;
- *matchEventIndicator*: identifica se é uma retransmissão;
- *tradingSessionID*: identificador da sessão de negociação;
- *tradeCondition*: conjunto de condições que descrevem uma negociação, isto é, cada negociação possui um conjunto de propriedades (“condições”). Por exemplo, o *OpeningPrice* é uma das mensagens enviadas durante a negociação de abertura do mercado, indicando quando um produto (instrumento) é negociado pela primeira vez no pregão em andamento, ou o *RegularTrade* que indica se aquele é um trade regular ou especial (no caso de especial, deve-se verificar o campo trdSubType);
- *mDEntryPx*: preço da ordem;
- *mDEntrySize*: quantidade da ordem;
- *tradeID*: contém o identificador exclusivo de uma negociação por instrumento + data de negociação, conforme atribuído pela bolsa. Obrigatório se estiver relatando uma negociação;
- *mDEntryBuyer*: campo usado para reportar negociações (parte compradora);
- *mDEntrySeller*: campo usado para reportar negociações (parte vendedora);
- *tradeDate*: usado para especificar a data de negociação à qual se aplica um conjunto de dados de mercado;
- *trdSubType*: subtipo de negociação atribuída a uma negociação que não for regular. Por exemplo, *RFQ_TRADE* que indica solicitação de cotação de negociação ou *MULTI_ASSET_TRADE* que indica a negociação de múltiplos ativos;

- *mDEntryTimestamp*: data e hora do instante em que a ordem chegou no bolsa;
- *rptSeq*: representa o valor do *sequence number*.

4.5 Discussão

Neste capítulo foi apresentado a arquitetura planejada que está sendo implementada no sistema *PUMA*. Detalhamos cada um dos blocos dessa arquitetura fazendo um *link* com os blocos implementados no ambiente de simulação. O ambiente proposto pela B3 foi minuciosamente analisado para garantir que o ambiente desenvolvido se assemelhe o máximo possível ao ambiente real. Isso é crucial para permitir que as estratégias dos participantes sejam testadas com a maior fidelidade possível, permitindo que os participantes possam melhorá-las antes de serem aplicadas no autêntico mercado financeiro.

Também foi fornecido uma visão abrangente dos protocolos utilizados pela B3 que foram implementados no ambiente de simulação, descrevendo os principais modelos de mensagens, tanto os enviados pelo mercado quanto os utilizados pelos participantes para executar e cancelar ordens. Além disso, explicamos os principais campos desses *templates* de mensagens que são analisados pelos participantes. Vale ressaltar que o entendimento desses protocolos e a forma como são decodificados podem ajudar os participantes a futuramente otimizarem seus sistemas de negociação, por exemplo, montando um pacote de resposta antes mesmo de receber todo um pacote do mercado.

Vale destacar que através da Sessão 4.4, apresentamos um estudo de caso que ilustra o funcionamento esperado do ambiente de simulação. Este estudo de caso aborda todo o processo de compra de um produto, desde a chegada do *feed* até a atualização do *feed* com a nova compra realizada. Note que este estudo de caso será executado no ambiente de simulação no próximo capítulo (Sessão 5.2.1), isso ajudará o leitor a ter um entendimento completo do ambiente de simulação. Neste sentido, serão utilizados os mesmos valores, e os resultados esperados devem coincidir com aqueles descritos no estudo de caso.

5. RESULTADOS

Este Capítulo expõe alguns dos resultados obtidos, que abrangem tanto análises de protocolos de comunicação - de crucial importância para o ambiente proposto - quanto exemplos de negociações realizadas no mesmo.

Na Sessão 5.1 detalha-se o estudo conduzido para avaliar alguns protocolos de comunicação. Especificamente, comparam-se as velocidades de diferentes protocolos em relação à quantidade de dados transmitidos. Vale ressaltar que foi utilizando o sistema operacional de robôs (ROS). Esta escolha se deve ao fato do ROS permitir uma maior flexibilidade na manipulação dos protocolos de comunicação e fidelidade na obtenção dos resultados e sua posterior avaliação.

Já na Sessão 5.2, apresentam-se exemplos concretos de negociações efetuadas no ambiente de simulação utilizando os protocolos providos pela B3. Discutem-se exemplos de dados enviados com base nos *templates* estudados, demonstrando como o ambiente decodifica, monta os pacotes recebidos e envia de volta para os blocos relativos a B3. Um dos exemplos é a reprodução de um caso de corrida entre dois participantes. Vale ressaltar que durante esses estudos de casos também exemplifica-se como utilizar efetivamente o ambiente de simulação.

5.1 Avaliação da Latência dos Protocolos de Comunicação

Inicialmente, foi realizado um estudo para avaliar o desempenho e entender as características dos principais protocolos de comunicação utilizados.

Um dos principais desafios desse estudo, era encontrar um ambiente que fosse flexível, tanto do lado do *software*, como do *hardware*, a fim de que fosse possível fazer manipulações e implementações em ambos os lados. Neste contexto, foi escolhido o sistema operacional de robôs (ROS, do inglês *Robotic Operation System*), um framework amplamente usado para desenvolver aplicações robóticas.

Vale contextualizar que nas últimas décadas foi visto um impressionante desenvolvimento industrial e uma maior adoção de sistemas robóticos. O aumento das capacidades dos sistemas robóticos trouxe desafios que antes eram secundários se tornarem primários, como a redução da latência de comunicação de sistemas robóticos de coprocessamento ser um desses desafios. Isto se deve ao grande aumento no poder de processamento do núcleo da CPU, auxiliado por sistemas de coprocessamento, como unidades de processamento gráfico (GPUs) e aceleradores de hardware especializados. Esse aumento nas capacidades de processamento pode tornar o sistema geral ineficiente, se a velocidade de comunicação não for escalável quanto a velocidade de computação.

Neste sentido, por mais que os temas pareçam divergentes, o resultado deste estudo pode ser aplicado tanto no ambiente robótico, como na análise do desempenho e entendimento dos protocolos de comunicação voltado ao mercado financeiro, pois em ambos os casos temos dois lados de processamento e a análise da latência durante a comunicação (isto é, durante a troca de mensagens). Ademais, vale ressaltar que este estudo gerou um artigo [Pereira et al., 2023] apresentado na conferência *NEWCAS* e publicado na *IEEE Xplore*.

Na prática, este estudo consistiu em definir um conjunto de cenários de coprocessamento, variando o número de nós, o tamanho da mensagem e o protocolo de transporte (TCP ou UDP). Em seguida, foram realizados experimentos registrando os tempos de envio e recebimento das mensagens em cada cenário. Os cenários deste estudo [Pereira et al., 2023] que são pertinentes à analise do desempenho dos protocolos TCP e UDP usados no mercado financeiro são descritos abaixo:

A. *ROS to ROS:* Este protocolo relaciona dois dispositivos com ROS, por exemplo, dois robôs se comunicando. Ele é baseado no modelo de publicação/subscrição fornecido pelas funções da biblioteca da ROS, onde um nó ROS (denominado *Publisher)* envia mensagens para um determinado tópico (ou seja, um objeto de dados armazenado em um espaço compartilhado que qualquer aplicativo ROS pode acesso). Em seguida, outro nó do ROS (denominado *Subscriter*) assina o tópico e começa a receber mensagens sempre que uma mensagem for publicada sobre o tópico. Nesta comunicação ROS-para-ROS, o protocolo TCP é usado por ambos os lados;

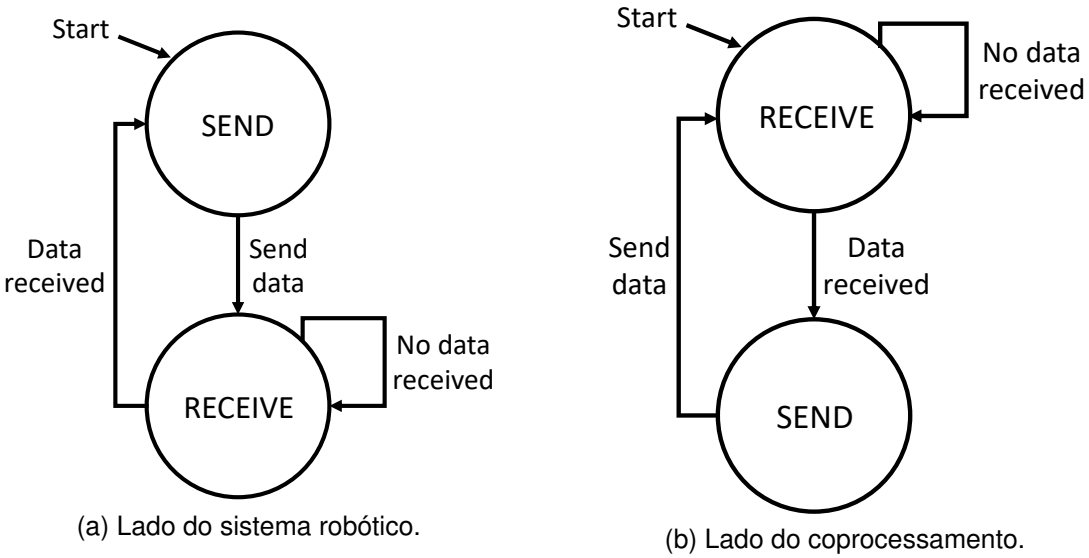
B. *ROS to ROS Bridge:* Uma *bridge* ROS é dividida em protocolo e implementação. Este protocolo de comunicação permite que aplicativos não-ROS trocem mensagens com aplicativos ROS. São usadas aqui três variantes e dois protocolos de comunicação efetivamente distintos. Em relação à implementação, o mais conhecido é o Roslibpy[1], uma biblioteca que permite que o uso do Python interaja com o ROS através do WebSockets, C1. Esta biblioteca ainda é baseada no protocolo TCP e fornece publicação, assinatura e outras funcionalidades essenciais da ROS. Outro protocolo depende do uso do UDP. Esta segunda opção compreende duas abordagens, dependendo do esquema de codificação escolhido: codificação JSON (C2) ou CBOR-RAW (C3);

C. *ROS para rede Ethernet:* Esse protocolo é o único que não se baseia no modelo de publicação/inscrição, mas na troca de mensagens na rede usando apenas *sockets* UDP. A estratégia fornece menos intermediação, reduzindo conversões ou ferramentas que podem adicionar custos de comunicação. No entanto, como o método não usa ferramentas de rede ROS (principalmente com base no TCP), ele não garante nenhuma confiabilidade para trocas de dados.

[1] https://roslibpy.readthedocs.io/en/latest/

Para avaliar adequadamente a latência dos protocolos de comunicações, todos os resultados foram extraídos de um único computador com a seguinte configuração: (*i*) Intel Core i7 9700kf, uma CPU de 8 núcleos com cache de 12 MB; (*ii*) Os núcleos da CPU são executados a 3,6 GHz (4,9 GHz max turbo); (*iii*) 16 GB 2666 MHZ DDR4 RAM; e (*iv*) Ubuntu 20.04.5 LTS OS.

Em relação ao experimento, o lado do sistema robótico é configurado com ROS Noetic e Python 3.8. Para simular o lado robótico, *roscore* deve primeiro ser instanciado para que os nós da ROS se comuniquem. Em seguida, os scripts são lançados para iniciar o processo de comunicação. A Figura 5.1 ilustra o processo de comunicação entre o sistema robótico e os lados de coprocessamento. Observe que, independentemente do protocolo de comunicação, a simulação tem o mesmo comportamento. Primeiro, o lado robótico começa, enviando um pacote de dados e aguardando até que a mensagem enviada retorne. Por outro lado, o coprocessamento começa em um *loop* de espera que consome dados e responde com a mesma mensagem.



(a) Lado do sistema robótico.

(b) Lado do coprocessamento.

Figura 5.1 – Máquinas de estado do processo de comunicação.

Os resultados de latência dos protocolos TCP e UDP são vistos na Figura 5.2 e Figura 5.3, respectivamente. A Figura 5.2 mostra uma característica intrigante. Quando o tamanho dos dados aumenta de 512B para 1kB, a latência salta de 10,5ms para 41,62ms, ou seja, quase 4$\times$ mais. Portanto, fica claro que enviar duas mensagens de 512B é mais rápido do que enviar uma de 1kB. No entanto, esta abordagem não é aplicável a outros tamanhos de mensagens, digamos 2kB.

A Figura 5.3 apresenta os resultados de latência que utiliza o protocolo UDP em ambos os lados. Primeiro, é surpreendente que a latência tenha caído cerca de 3 ordens de grandeza, passando da faixa de ms para a faixa de μs. Além disso, o intervalo geral é muito pequeno, indo de 7,06 a 13,58μs. Note que a camada de transporte UDP para ROS é baseada no pacote de datagrama UDP padrão, ou seja, o tamanho máximo dos dados é

ligeiramente inferior a 64kB, por causa do cabeçalho. Portanto, mensagens maiores devem ser quebradas.

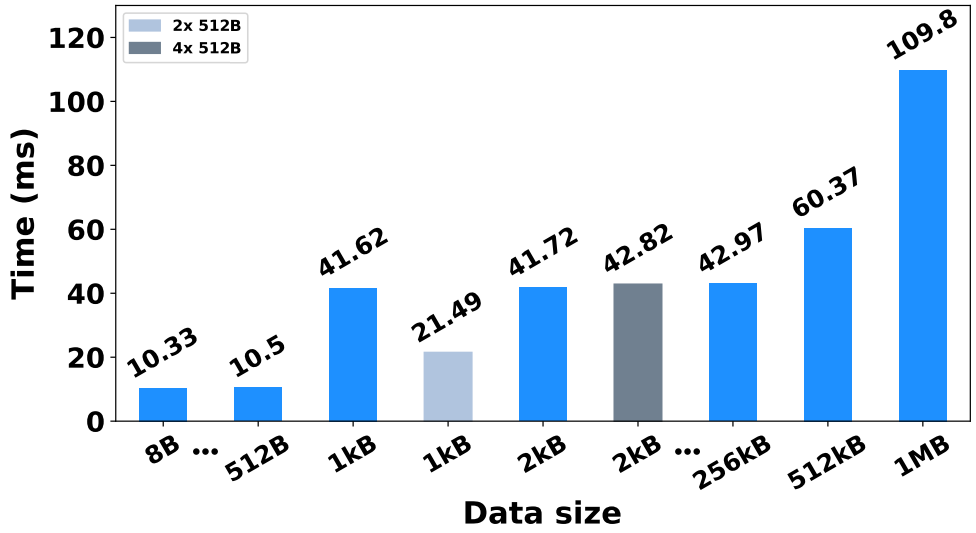


Figura 5.2 – Resultados de latência dos cenários baseados no protocolo TCP (A e B). Fonte: [Pereira et al., 2023].

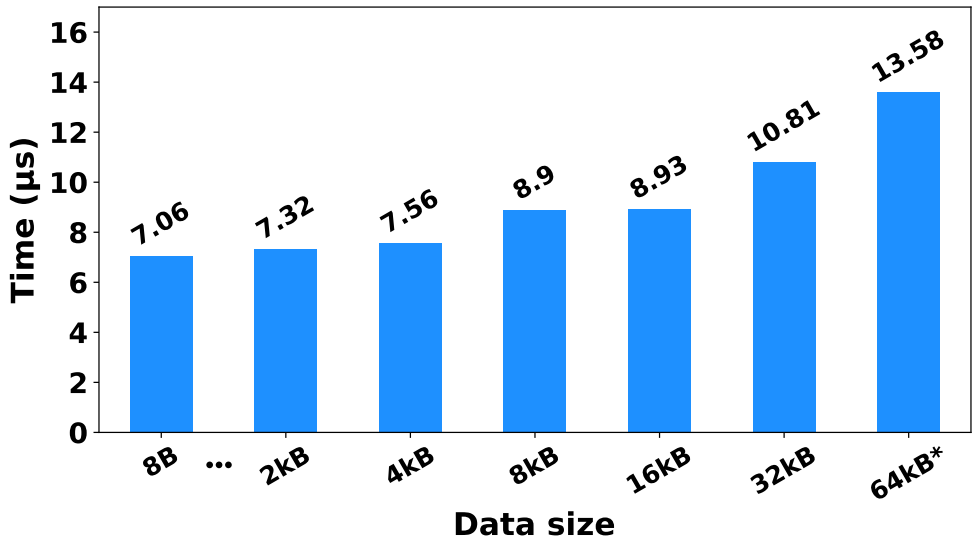


Figura 5.3 – Tempo de comunicação para o protocolo baseado em UDP (C). Observe que, devido ao cabeçalho UDP, o tamanho máximo dos dados é um pouco menor que 64kb (64kb*). Fonte: [Pereira et al., 2023].

Nota-se uma grande diferença de tempo do experimento realizado com o protocolo TCP para o experimento com o protocolo UDP, sendo o do protocolo TCP na ordem dos milissegundos e o protocolo UDP na ordem dos microssegundos. O protocolo TCP é mais lento pois este é um protocolo orientado a conexão, com confirmação de entregas de pacotes. Por outro lado, o protocolo UDP é mais simples, menos confiável, e com um tamanho menor de dados devido ao seu cabeçalho, isso o torna mais rápido em termos de latência de comunicação.

Logo, após a análise desses resultados, foi visto que o protocolo UDP é o mais recomendado quando não necessita de garantia de entrega e sim somente velocidade, pois é o mais rápido para o transporte. Já quando é necessário fazer o *handshake*, e a garantia de entrega de mensagens, o mais recomendado é o protocolo TCP.

5.2 Avaliação da Plataforma de Simulação

Nesta seção apresentamos simulações através de estudos de casos utilizando a plataforma desenvolvida, a fim de apresentar ao leitor como pode-se reproduzir situações reais dentro da plataforma de simulação.

5.2.1 Realização de uma compra

A partir do estudo de caso apresentado na Sessão 4.4, realizamos uma simulação semelhante na plataforma de simulação desenvolvida. Como primeiro passo iniciamos o *Gateway* para que os participantes do mercado financeiro possam se conectar, como mostrado na Figura 5.4.

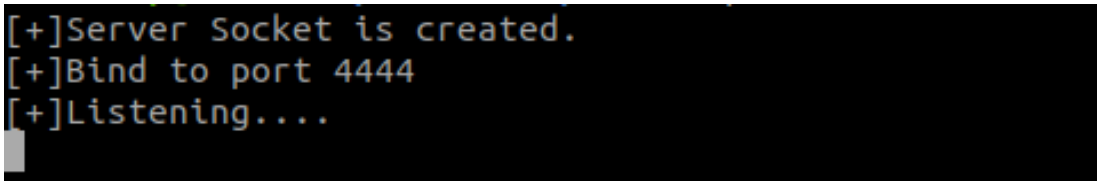


Figura 5.4 – Iniciando o *Gateway*.

Logo em seguida, o participante se conecta ao *Gateway*, enviando seu *login*. O *Gateway* analisa o *login* recebido e se o *login* for válido, a conexão é aceita, como mostrado na Figura 5.5.

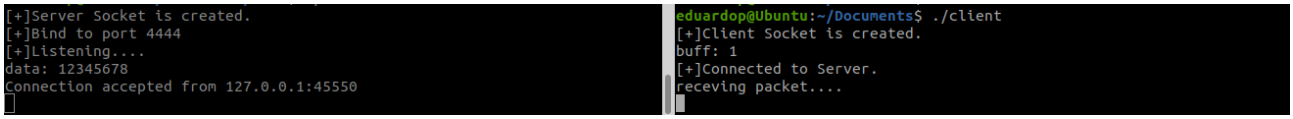


Figura 5.5 – Participante se conectando com o *Gateway*.

Uma vez que a conexão entre um participante e o *Gateway* é aceita, pode-se trocar mensagens do mercado. Na Figura 5.6 é mostrado o envio da primeira atualização de *feed* pelo *Matching Engine* com o pacote *ExecutionSummmary* [55]. A Tabela 5.1 apresenta os dados deste pacote.

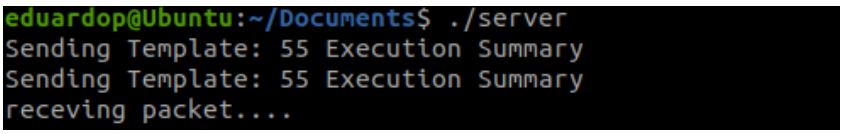


Figura 5.6 – *Matching Engine* enviando primeiro pacote para o participante.

Tabela 5.1 – Dados do pacote *ExecutionSummary* [55] usados na primeira atualização de *feed*.

Campo	Valor (hex)	Valor
securityID	4a58774817000000	100000028746
padding	0000	0
aggressorSide	01	1
padding	00	0
lastPx	400d030000000000	200000
fillQty	0a00000000000000	10
tradedHiddenQty	0000000000000000	0
cxlQty	0000000000000000	0
aggressorTime	c7f9b21dac336d17	498268615
rptSeq	40000000	40
mDEntryTimestamp	01e1511fac336d17	525459713

Nesta simulação, a Tabela 5.1 exemplifica que o pacote enviado para o participante tem o *securityID* de 100000028746, com preço de 200000, além de 10 unidades disponíveis.

Após o participante receber a primeira atualização do *feed* com o pacote *ExecutionSummmary* [55], o participante decodifica a mensagem. A decodificação é realizada com *structs* para cada pacote e baseado no tamanho específico de cada informação. A *struct* do pacote *ExecutionSummmary* [55] é mostrado na Figura 5.7.

A Figura 5.8 ilustra o simulador fazendo esta decodificação, a fim de montar o pacote *NewOrderSingle* [102]. Para montar o pacote *NewOrderSingle* [102], é utilizado uma *struct* semelhante à Figura 5.7, porém com os campos respectivos do template [102]. Esses campos podem ser visto na Tabela 5.2, que mostra também os valores usados nesta simulação. Note que para a montagem do pacote *NewOrderSingle* [102] (Figura 5.9), o participante utilizou suas informações (baseadas em credenciais e na estratégia de negociação que ele quer aplicar), além de informações que vieram do pacote [55], como o *securityID*.

Na Tabela 5.2 é mostrada como o participante montou o seu pacote de acordo com alguma estratégia de negociação que ele tenha interesse (vale ressaltar que a estratégia em si não é o escopo desta dissertação, mas sim prover meios para que os usuários dessa plataforma possam o fazer). Neste pacote, o participante envia seu ID para a bolsa, o nome da sua corretora, o *trader* que está realizando a negociação, etc.

Na sequência da simulação, o *Gateway* recebe a ordem de compra e mostra a *TAG* de tempo do momento em que a ordem chegou, como mostrado na Figura 5.10.

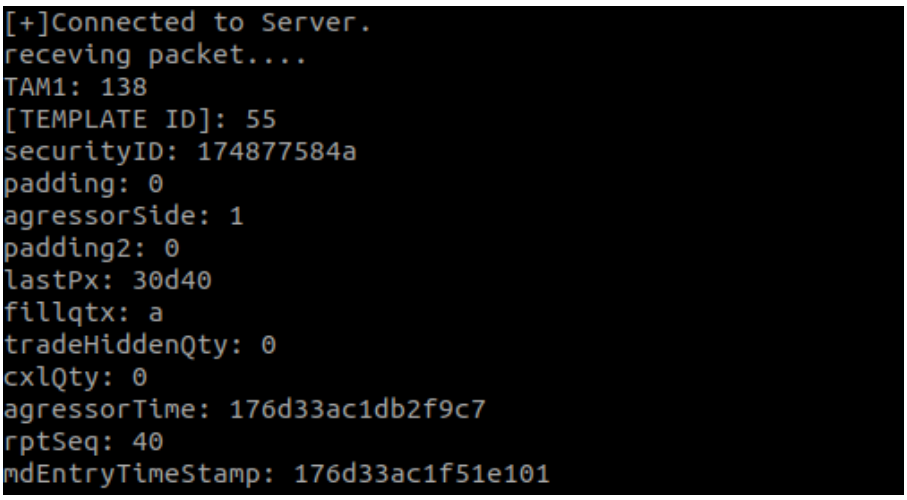


Figura 5.8 – Realização de uma compra e enviando para o *Gateway*.

```
struct pkt55
{
    uint64_t UDP1;
    uint64_t UDP2;
    uint64_t UDP3;
    uint64_t UDP4;
    uint64_t UDP5;
    uint32_t UDP6;
    uint16_t UDP7;
    uint8_t chNumber;
    uint8_t reserved;
    uint16_t seqVersion;
    uint32_t seqNumber;
    uint64_t sendingTime;
    uint16_t msgLength;
    uint16_t encondingType;
    uint16_t blockLength;
    uint16_t templateId;
    uint16_t schemaId;
    uint16_t version;
    uint64_t securityID;
    uint16_t padding;
    uint8_t agressorSide;
    uint8_t padding2;
    uint64_t lastPx;
    uint64_t fillqtx;
    uint64_t tradeHiddenQty;
    uint64_t cxlQty;
    uint64_t agressorTime;
    uint32_t rptSeq;
    uint64_t mdEntryTimeStamp;
} __attribute__((packed));
```

Figura 5.7 – *Struct* para decodificar o pacote *ExecutionSummmary* [55].

```
msg102->templateId = 0x66;
msg102->ClOrdID = 0x01;
msg102->EnteringFirm = 0x00505543;
msg102->EnteringTrader = 0xF5BA50;
msg102->SenderLocation = 0X00;
msg102->OrdTagID = 0X01;
msg102->MarketSegmented = 0XAF;
msg102->side = 0X01;
msg102->TransactTime = 0X65DCF0B6;
msg102->OrderQty = 0X05;
msg102->securityID = msg55->securityID;
msg102->TimeInForce = 0x00;
msg102->OrdType = 0x02;
```

Figura 5.9 – Pacote *NewOrderSingle* [102] sendo montado pelo participante.

Tabela 5.2 – Exemplo do estudo de caso do pacote *NewOrderSingle* [102].

Campo	**Valor (hex)**	**Valor**
ClOrdID	0000000000000001	1
EnteringFirm	00505543	PUC
EnteringTrader	00000000000000F5BA50	16104016
SenderLocation		
OrdTagID	01	1
MarketSegmentId	AF	175
Side	01	Buy
TransactTime	0000000065DCF0B6	1708978358
OrderQty	0000000000000005	5
securityID	4a58774817000000	100000028746
TimeInForce	00	0
OrdType	02	2

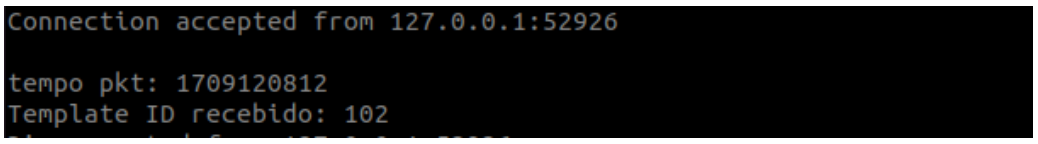


Figura 5.10 – Recebimento de uma ordem de compra pelo *Gateway*.

Logo em seguida, o *Order Book* monta o pacote *Trade* [53], com a ordem de compra recebida e envia a informação para o *Matching Engine*, a fim de o mesmo enviar um novo *feed* para os participantes. Logo, a informação de negociação é mostrada através do pacote *Trade* [53].

Na Tabela 5.3 é mostrado como foi montado o pacote *Trade* [53], exemplificando como a bolsa manda uma nova mensagem de *feed* para os participantes com a informação da compra de um ativo. Os principais campos preenchidos na Tabela 5.3 são o *securityId*, que é do produto negociado, quantas ordens desse produto foram compradas (*mDEntry-Size*), o preço pago por cada quantidade (*mDEntryPx*), e qual corretora fez a compra

(*mDEntryBuyer*). Note que como não foi realizada nenhuma venda, o campo *mDEntrySeller* está vazio.

Tabela 5.3 – Exemplo do estudo de caso do pacote *Trade* [53].

Campo	Valor (hex)	Valor
securityID	4a58774817000000	100000028746
matchEventIndicator	00	00
tradingSessionID	01	1
tradeCondition	2001	8193
mDEntryPx	400d030000000000	200000
mDEntrySize	0000000000000005	5
tradeID	0000000a	10
mDEntryBuyer	00505543	PUC
mDEntrySeller	00000000	00
tradeDate	4c51	19537
trdSubType	01	*RFQ_TRADE*
padding	00	00
mDEntryTimestamp	0000000065DF1D2C	1709120812
rptSeq	0001	1

A Figura 5.11 mostra o participante decodificando o pacote *Trade* [53]. Nele, podemos ver, por exemplo, a *TAG* de tempo que foi colocada pelo *Gateway* no campo *mdEntryTimestamp*, o qual é o mesmo mostrado pelo *Gateway* na Figura 5.10. Essa informação é útil para que o participante consiga montar ele próprio uma lista de ordens ordenada, que auxilia na análise de performance do sistema em negociações do tipo corrida.

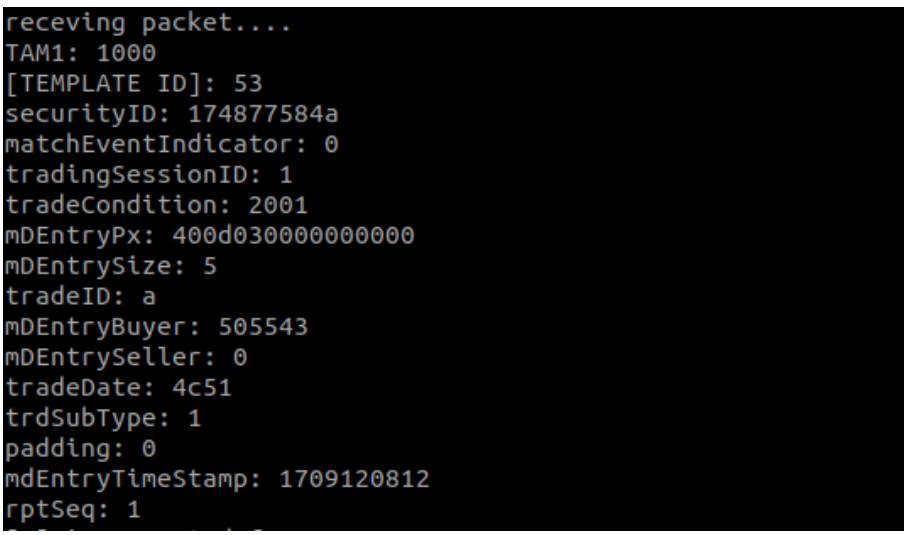


Figura 5.11 – Participante recebendo nova atualização do mercado.

5.2.2 Corrida entre participantes

Existem produtos negociados no mercado de produtos e futuros que tem uma característica especial, onde a estratégia de negociação não é somente baseada em compra e venda de produtos, mas sim, a principal característica que vai trazer rentabilidade ao participante é a velocidade em que uma ordem chega no *Matching Engine*. Para estes casos, chamaremos nesta dissertação de *corrida entre participantes*.

A Figura 5.12 ilustra um exemplo onde temos uma corrida de tempo entre dois participantes, onde o processo de compra de um produto realizado no exemplo apresentado na Sessão 5.2.1 é feito por dois participantes simultaneamente. Neste cenário, onde temos o simulador do mercado financeiro com um *gateway* e dois participantes, cada um enviando uma requisição de compra ao *gateway*, o objetivo é determinar qual participante chega primeiro ao *gateway* para realizar a compra. Isto significa que apenas um dos dois participantes conseguirá fazer a transação (o mais rápido) e o outro não (o mais lento).

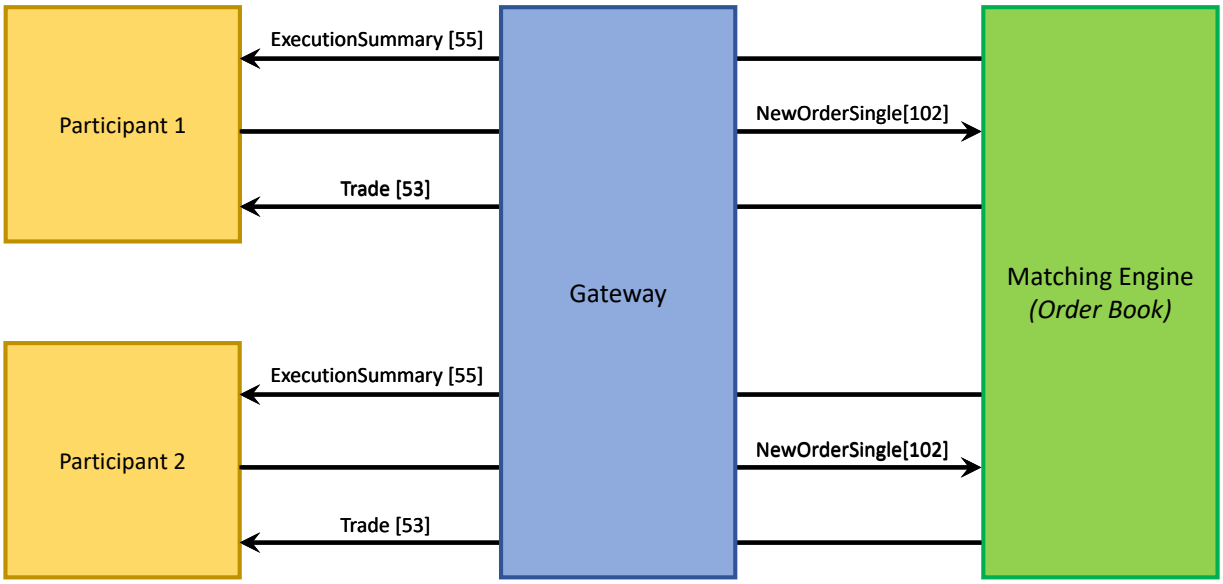


Figura 5.12 – Simulação de uma corrida entre participantes.

Como no primeiro exemplo, o *Gateway* é inicializado e os dois participantes precisam se conectar ao mercado utilizando suas credenciais. A Figura 5.13 ilustra esse processo de conexão. Note que o *Participante 1* é visto no *Gateway* pela identificação: *127.0.0.1:41764*, enquanto o *Participante 2* é visto pela identificação *127.0.0.1:41772*.

O *gateway* é composto por duas *threads*, cada *thread* se conecta a um participante específico, como descrito na Sessão 4.2.3. Assim, quando as requisições de compra são enviadas pelos participantes, ambas as *threads* do *gateway* ficam aguardando a chegada das requisições. Quando uma requisição chega, a *thread* correspondente a esse participante registra o tempo de chegada da requisição.

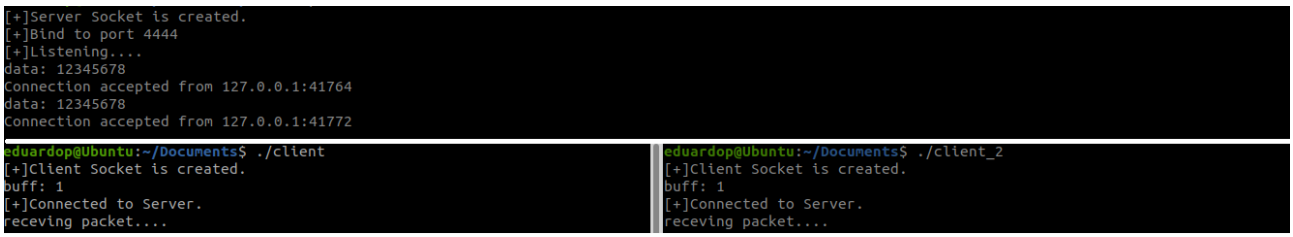


Figura 5.13 – Processo de dois participantes se conectando ao *Gateway*.

A determinação de qual participante chegou primeiro ao *gateway* é feita comparando os tempos registrados pelas *threads*. A *thread* que registrou o tempo mais cedo indica que o respectivo participante chegou primeiro. Isso é possível porque, mesmo que as *threads* sejam executadas de forma concorrente, o sistema operacional garante uma ordem de execução determinística.

Assim, o participante cuja requisição foi registrada com o tempo mais cedo é considerado o primeiro a chegar ao *gateway* e, consequentemente, realiza a compra. Este processo permite simular a competição entre os participantes, garantindo uma execução justa e precisa do simulador de mercado financeiro.

Na Figura 5.14 é mostrado o *Gateway* recebendo os pacotes dos dois participantes e mostrando qual chegou primeiro. Observa-se que o *Timestamp* é o mesmo para as duas requisições. Contudo, o tempo em milissegundos mostra que o participante de endereço 127.0.0.1:41772 (*Participante 2*) chegou na frente do outro participante (*Participante 1*) em 48457 ms.

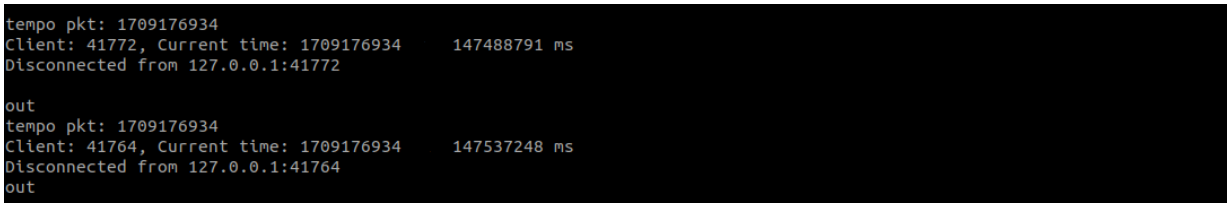


Figura 5.14 – Corrida de tempo entre dois participantes.

Vale ressaltar que nesta simulação, pelo fato dos participantes não estarem usando nenhuma estratégia, e serem exatamente iguais, o *Matching Engine* influência diretamente nesta corrida entre participantes. Isto se deve ao fato que o *Matching Engine* não consegue mandar ao mesmo tempo as informações para os dois participantes. Portanto, o *Matching Engine* sorteia em cada execução qual será o participante que ele irá enviar primeiro a mensagem. Note que esta abordagem não deixa o simulador com menos fidelidade, pois no mundo real, cada participante vai ter um *delay* diferente para receber os *feeds*. Mesmo que os sistemas dos participantes estejam dentro do *Datacenter* da bolsa (este serviço é chamado de *Colocation*), existem vários parâmetros que trazem essa variância na latência das informações do mercado, que vão desde se o participante conecta o cabo em um *switch* ou diretamente em um FPGA, até o tamanho do próprio cabo utilizado influência no tempo em que cada participante irá receber o *feed* de mercado.

6. CONCLUSÃO E TRABALHOS FUTUROS

Esta dissertação de mestrado produziu uma plataforma de simulação que vem de encontro com as transformações que vem ocorrendo nos mercados financeiros ao longo da última década, destacando a transição de negociações baseadas em pregões físicos para negociações automatizadas impulsionadas por algoritmos avançados. Em particular, este trabalho focou no sistema usado na bolsa de mercadorias e futuros do Brasil, e na importância da latência de resposta do sistema de negociação e eficácia das estratégias aplicadas pelos participantes no ambiente de negociação atual.

Foi observado que a latência de resposta é um fator crucial na lucratividade das empresas que operam nos mercados financeiros. Nesse contexto, destacamos nos trabalhos relacionados a crescente popularidade das soluções baseadas em FPGA, que oferecem baixa latência através de um hardware dedicado. Além disso, reconhecemos o papel fundamental do aprendizado de máquina e dos algoritmos de aprendizado profundo no ajuste dinâmico das estratégias de negociação.

Ao explorar os conceitos fundamentais relacionados às bolsas de mercadorias e futuros, sistemas de negociação e protocolos de comunicação, estabelecemos uma base sólida para compreender as complexidades envolvidas na implementação de estratégias de mercado. Destacamos a motivação por trás deste trabalho, que surge das mudanças implementadas pela B3 em seu sistema de negociação PUMA, incluindo a transição do protocolo FIX/FAST para o protocolo UMDF.

Os objetivos e contribuições desta dissertação de mestrado tiveram foco no desenvolvimento de um ambiente de simulação robusto e eficiente para que possa ser possível a aplicação de estratégias de mercado no contexto do novo protocolo UMDF. Por meio de uma metodologia planejada, que envolveu desde o estudo dos conceitos dos novos protocolos até a implementação e teste do ambiente de simulação, buscando preencher uma lacuna importante na preparação dos participantes do mercado para as mudanças iminentes na B3. A plataforma de simulação desenvolvida está disponível em: https://github.com/duduvpereira/AmbienteSimPUMA.

A descrição detalhada do ambiente de simulação do sistema PUMA de negociação, juntamente com os resultados obtidos da avaliação da latência dos protocolos de comunicação e da plataforma de simulação, demonstra o impacto significativo que este trabalho pode ter no desenvolvimento e aprimoramento das estratégias de mercado.

Vale ressaltar que uma das principais dificuldades enfrentadas durante a realização deste trabalho foi a escassez de referências acadêmicas que abordassem o estado da arte deste tema específico. Tal cenário pode ser atribuído tanto à novidade do protocolo em questão, o qual começou a ser amplamente utilizado apenas recentemente, como

na falta de interesse das empresas de apresentarem suas soluções, uma vez isso implica diretamente no diferencial tecnológico da empresa.

6.1 Trabalhos Futuros

A partir do desenvolvimento desta dissertação de mestrado, destacamos alguns pontos que podem ser realizados como trabalhos futuros:

- Como trabalho futuro, sugere-se implementar a decodificação de outros *templates*, além dos mostrados na Sessão 4.3;
- Outro ponto importante a ser implementado no futuro seria colocar diferentes estratégias de mercado nos participantes no ambiente, para que a corrida seja feita com mais precisão em relação ao mercado real;
- Uma melhoria que poderia ser realizada futuramente é na criação de um novo *feed* feito a cada ordem executada, isso pode ser feito através de um melhor entendimento das estratégias usadas no mercado, e de dados reais utilizados nas transações e negociações;
- Por fim, outro trabalho futuro seria portar o ambiente desenvolvido, pelo menos a parte dos participantes, para um ambiente no nível RTL (do inglês, *register-transfer level*), a fim de que possa ser comparado as estratégias tanto em *software*, quanto em *hardware*.

REFERÊNCIAS BIBLIOGRÁFICAS

[Abeyrathne et al., 2021] Abeyrathne, P. T., Dewasurendra, S. D., and Elkaduwa, D. (2021). Offloading specific performance-related kernel functions into an FPGA. In *IEEE International Symposium on Industrial Electronics (ISIE)*, pages 1–6. https://doi.org/10.1109/ISIE45552.2021.9576256.

[B3, 2019] B3 (2019). UMDF – Unified Market Data Feed. https://www.b3.com.br/data/files/A4/11/B5/27/B1C6C6106B9896C6DC0D8AA8/UMDF_MarketDataSpecification_v2.1.7.pdf, March 2024.

[B3, 2022] B3 (2022). Market data B3: Binary UMDF Message Reference. https://www.b3.com.br/data/files/22/64/31/B1/4D2408104532BBF7AC094EA8/Binary%20UMDF%20-%20Message%20Reference%20-%20v.1.3.2.pdf, March 2024.

[B3, 2023a] B3 (2023a). ETFs ligados a commodities negociados na Bolsa e seus desempenhos. https://borainvestir.b3.com.br/tipos-de-investimentos/renda-variavel/etfs/o-desempenho-no-ano-dos-5-etfs-ligados-a-commodities-negociados-na-bolsa/, March 2024.

[B3, 2023b] B3 (2023b). Market Data B3: Binary UMDF Messaging Specification Guidelines. https://www.b3.com.br/data/files/1D/B4/67/8C/69A31810C493CD08AC094EA8/Binary%20UMDF%20-%20Message%20Specification%20Guidelines%20-%20v.1.4.0.pdf, March 2024.

[B3, 2023c] B3 (2023c). Roadmap - Resumo do 3° trimestre de 2023. https://clientes.b3.com.br/c/document_library/get_file?uuid=ba072b66-f343-cc10-9adf-82db03cf956a, March 2024.

[Boutros et al., 2017] Boutros, A., Grady, B., Abbas, M., and Chow, P. (2017). Build Fast, Trade Fast: FPGA-based High-Frequency Trading using High-Level Synthesis. In *International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, pages 1–6. https://doi.org/10.1109/RECONFIG.2017.8279781.

[CME, 2023] CME (2023). CME Group – Produtos de futuros e opções. https://www.cmegroup.com/pt/products.html, March 2024.

[CME, 2024] CME (2024). Chicago Mercantile Exchange. https://www.cmegroup.com/, March 2024.

[Darwinex, 2019] Darwinex (2019). Anatomy of the FIX Protocol | FIX API for Algorithmic Trading. https://www.youtube.com/watch?v=wSgAwJyev2Y, March 2024.

[De et al., 2009] De, P., Mann, V., and Mittaly, U. (2009). Handling OS jitter on multicore multithreaded systems. In *IEEE International Symposium on Parallel and Distributed Processing (IPDPS)*, pages 1–12. https://doi.org/10.1109/IPDPS.2009.5161046.

[DME, 2023a] DME (2023a). About Dubai Mercantile Exchange Limited (DME). https://www.dubaimerc.com/about-dme, March 2024.

[DME, 2023b] DME (2023b). DME Contracts. https://dubaimerc.com/List-Contracts, March 2024.

[Equipe Mais Retorno, 2020] Equipe Mais Retorno (2020). CME – Chicago Mercantile Exchange. https://maisretorno.com/portal/termos/c/cme-chicago-mercantile-exchange, March 2024.

[EUREX, 2023a] EUREX (2023a). Commodity derivatives. https://www.eurex.com/ex-en/markets/com, March 2024.

[EUREX, 2023b] EUREX (2023b). T7 System. https://www.eurex.com/ex-en/support/technology/t7, March 2024.

[EUREX, 2023c] EUREX (2023c). T7 System Trading Architecture. https://www.eurex.com/t7/, March 2024.

[Fu et al., 2017] Fu, H., He, C., Luk, W., Li, W., and Yang, G. (2017). A Nanosecond–Level Hybrid Table Design for Financial Market Data Generators. In *IEEE Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pages 227–234. https://doi.org/10.1109/FCCM.2017.30.

[Funie et al., 2015] Funie, A. I., Grigoras, P., Burovskiy, P., Luk, W., and Salmon, M. (2015). Reconfigurable acceleration of fitness evaluation in trading strategies. In *IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, pages 210–217. https://doi.org/10.1109/ASAP.2015.7245736.

[Gao et al., 2020] Gao, J., Yin, W., Luk, W.-S., and Wang, L. (2020). Scalable Multi-Session TCP Offload Engine for Latency-Sensitive Applications. In *China Semiconductor Technology International Conference (CSTIC)*, pages 1–3. https://doi.org/10.1109/CSTIC49141.2020.9282453.

[Group, 2023] Group, C. (2023). CME Globex - Negociação eletrônica. https://www.cmegroup.com/pt/globex.html, March 2024.

[He and Wen, 2019] He, F. and Wen, Y.-D. (2019). PRAM: A Novel Approach for Predicting Riskless State of Commodity Future Arbitrages With Machine Learning Techniques. *IEEE Access*, 7:159519–159526. https://doi.org/10.1109/ACCESS.2019.2950858.

[Hu, 2023] Hu, X. (2023). Using machine learning models to forecast future prices – an exploration of investment strategies. In *International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA)*, pages 468–474. https://doi.org/10.1109/ICIBA56860.2023.10165490.

[Huang et al., 2022] Huang, B., Huan, Y., Jia, H., Ding, C., Yan, Y., Huang, B., Zheng, L.-R., and Zou, Z. (2022). AIOC: An All-in-One-Card Hardware Design for Financial Market Trading System. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 69(9):3894–3898. https://doi.org/10.1109/TCSII.2022.3167312.

[Kao et al., 2022] Kao, Y.-C., Chen, H.-A., and Ma, H.-P. (2022). An FPGA-Based High-Frequency Trading System for 10 Gigabit Ethernet with a Latency of 433 ns. In *International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, pages 1–4. https://doi.org/10.1109/VLSI-DAT54769.2022.9768065.

[Ke et al., 2023] Ke, H., Zuominyang, Z., Qiumei, L., and Yin, L. (2023). Predicting Chinese Commodity Futures Price: An EEMD-Hurst-LSTM Hybrid Approach. *IEEE Access*, 11:14841–14858. https://doi.org/10.1109/ACCESS.2023.3239924.

[Klaisoongnoen et al., 2022] Klaisoongnoen, M., Brown, N., and Brown, O. T. (2022). Fast and energy-efficient derivatives risk analysis: Streaming option Greeks on Xilinx and Intel FPGAs. In *IEEE/ACM International Workshop on Heterogeneous High-performance Reconfigurable Computing (H2RC@S)*, pages 18–27. https://doi.org/10.1109/H2RC56700.2022.00008.

[Kohda and Yoshida, 2021] Kohda, S. and Yoshida, K. (2021). Characteristics of High-Frequency Trading and Its Forecasts. In *IEEE Computers, Software, and Applications Conference (COMPSAC)*, pages 1496–1501. https://doi.org/10.1109/COMPSAC51774.2021.00222.

[Leber et al., 2011] Leber, C., Geib, B., and Litz, H. (2011). High Frequency Trading Acceleration Using FPGAs. In *International Conference on Field Programmable Logic and Applications (FPL)*, pages 317–322. https://doi.org/10.1109/FPL.2011.64.

[Moscon, 2023] Moscon, B. (2023). Orderbook. https://github.com/bmoscon/orderbook, March 2024.

[Nabipour et al., 2020] Nabipour, M., Nayyeri, P., Jabani, H., S., S., and Mosavi, A. (2020). Predicting Stock Market Trends Using Machine Learning and Deep Learning Algorithms Via Continuous and Binary Data; a Comparative Analysis. *IEEE Access*, 8:150199–150212. https://doi.org/10.1109/ACCESS.2020.3015966.

[NYMEX, 2023] NYMEX (2023). New York Mercantile Exchange. https://www.cmegroup.com/company/nymex.html, March 2024.

[Oliveira and Bonato, 2023] Oliveira, C. C. S. and Bonato, V. (2023). A FAST Hardware Decoder Optimized for Template Features to Obtain Order Book Data in Low Latency. *Journal of Signal Processing Systems*, 95(4):559–567. https://doi.org/10.1007/s11265-023-01850-2.

[Pereira et al., 2023] Pereira, E., Luza, L., Moura, N., Ost, L., Calazans, N., Moraes, F. G., and Garibotti, R. (2023). Assessment of Communication Protocols' Latency in Co-processing Robotic Systems. In *IEEE Interregional NEWCAS Conference (NEWCAS)*, pages 1–5. https://doi.org/10.1109/NEWCAS57931.2023.10198085.

[Ramakrishnan et al., 2017] Ramakrishnan, S., Butt, S., Chohan, M. A., and Ahmad, H. (2017). Forecasting Malaysian exchange rate using machine learning techniques based on commodities prices. In *International Conference on Research and Innovation in Information Systems (ICRIIS)*, pages 1–5. https://doi.org/10.1109/ICRIIS.2017.8002544.

[Reis, 2020] Reis, T. (2020). New York Mercantile Exchange (NYMEX). https://www.suno.com.br/artigos/nymex/, March 2024.

[Ricardo, 2020] Ricardo, J. (2020). Como funciona o Eurex? https://economiaenegocios.com/como-funciona-o-eurex/, March 2024.

[Rosati et al., 2020] Rosati, R., Romeo, L., Goday, C. A., Menga, T., and Frontoni, E. (2020). Machine Learning in Capital Markets: Decision Support System for Outcome Analysis. *IEEE Access*, 8:109080–109091. https://doi.org/10.1109/ACCESS.2020.3001455.

[Sonare et al., 2023] Sonare, B., Patil, S., Pise, R., Bajad, S., Ballal, S., and Chandre, Y. (2023). Analysis of Various Machine Learning and Deep Learning Algorithms for Bitcoin Price Prediction. In *International Conference on Recent Advances in Electrical, Electronics, Ubiquitous Communication, and Computational Intelligence (RAEEUCCI)*, pages 1–5. https://doi.org/10.1109/RAEEUCCI57140.2023.10134467.

[Stumpf, 2023] Stumpf, K. (2023). Mercado Financeiro: o Que É, Quais os Tipos e Como Funciona. https://www.topinvest.com.br/mercado-financeiro-o-que-e/, March 2024.

[Suman et al., 2022] Suman, S., Kaushik, P., Challapalli, S. S. N., Lohani, B. P., Kushwaha, P., and Gupta, A. D. (2022). Commodity Price Prediction for Making Informed Decisions while trading using Long Short-Term Memory (LSTM) Algorithm. In *International Conference on Contemporary Computing and Informatics (IC3I)*, pages 406–411. https://doi.org/10.1109/IC3I56241.2022.10072626.

[Tan et al., 2021] Tan, J., Lv, G., Ma, Y., and Qiao, G. (2021). High-performance pipeline architecture for packet classification accelerator in DPU. In *International Conference on Field-Programmable Technology (ICFPT)*, pages 1–4. https://doi.org/10.1109/ICFPT52863.2021.9609841.

[Tang et al., 2016] Tang, Q., Jiang, L., Su, M., and Dai, Q. (2016). A pipelined market data processing architecture to overcome financial data dependency. In *IEEE International Performance Computing and Communications Conference (IPCCC)*, pages 1–8. https://doi.org/10.1109/PCCC.2016.7820632.

[Usha et al., 2019] Usha, B. A., Manjunath, T. N., and Mudunuri, T. (2019). Commodity and Forex trade automation using Deep Reinforcement Learning. In *International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE)*, pages 27–31. https://doi.org/10.1109/ICATIECE45860.2019.9063807.

[Wang et al., 2020] Wang, S., Meng, Z., Sun, C., Wang, M., Xu, M., Bi, J., Yang, T., Huang, Q., and Hu, H. (2020). SmartChain: Enabling High-Performance Service Chain Partition between SmartNIC and CPU. In *IEEE International Conference on Communications (ICC)*, pages 1–7. https://doi.org/10.1109/ICC40277.2020.9149136.

Pontifícia Universidade Católica do Rio Grande do Sul
Pró-Reitoria de Pesquisa e Pós-Graduação
Av. Ipiranga, 6681 – Prédio 1 – Térreo
Porto Alegre – RS – Brasil
Fone: (51) 3320-3513
E-mail: propesq@pucrs.br
Site: www.pucrs.br