Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 21.Б11-мм

Инструмент для технического анализа торговли на биржах

Шангареев Рустам Ильшатович

Отчёт по производственной практике
в форме «Решение»

Научный руководитель:
ассистент кафедры системного программирования В.И. Гориховский

Санкт-Петербург
2023

Оглавление

Введение

В последние годы мир финансов претерпевает бурное развитие благодаря революционным технологиям и широкому распространению интернета. Одним из наиболее значимых и динамично растущих сегментов данной сферы является трейдинг, особенно с учетом появления и популяризации криптовалют и различных валютных инструментов. Эта деятельность привлекает не только профессиональных участников рынка, но и множество частных инвесторов, стремящихся к увеличению своего капитала с помощью эффективных торговых операций на бирже.

С возрастанием числа трейдеров и инвесторов возникает потребность в комплексном и научно обоснованном подходе к анализу рынков. Технический анализ, как инструмент предсказания будущего движения цен на основе исторических данных, не теряет своей актуальности и постоянно развивается. Его эффективность и применимость доказаны множеством исследований и успешных торговых стратегий, которые делают его незаменимым для современных трейдеров. Рассматривая графики ценовых движений, индикаторы и объемы торгов, трейдеры могут выявить закономерности и использовать их для составления прогнозов.

Однако высокая степень волатильности и множество влияющих факторов требуют от аналитических инструментов высокой точности и производительности. В этом контексте значимую роль играют программные решения с открытым исходным кодом, которые обеспечивают гибкость, модифицируемость и высокую скорость обработки данных. Сообщества разработчиков всего мира сотрудничают над созданием и совершенствованием инструментов анализа данных, облегчая таким образом доступ к качественным рыночным аналитическим системам.

Целью данной работы является разработка высокопроизводительного инструмента с открытым исходным кодом для анализа данных с бирж, который не только предоставит требуемую аналитическую глубину и широту, но и будет доступен для использования широкой аудиторией пользователей, включая трейдеров-любителей и опытных анали-

тиков рынка. Принципиально важными характеристиками такого инструмента являются надежность, скорость работы, а также возможность интеграции с существующими системами и инструментами для обеспечения полноты и многосторонности проводимого анализа.

1. Постановка задачи

Целью данной работы является разработка высокопроизводительного инструмента для технического анализа данных с бирж.

1. Провести общий обзор предметной области и существующих решений
2. Определить требования к инструменту
3. Спроектировать архитектуру инструмента и выбрать технологии для реализации
4. Разработать систему структур данных для эффективного хранения и обработки сделок с бирж
5. Реализовать программный интерфейс для установления соединения и аггрегации сделок в реальном времени с использованием WebSocket и API ByBit
6. Провести модульное тестирование инструмента
7. Проанализировать полученные результаты, выявить и устранить ошибки, составить список задач на следующий семестр

2. Обзор

В данном разделе будет произведен общий обзор предметной области и существующих решений.

2.1. Обзор предметной области

Трейдинг представляет собой активный подход к покупке и продаже финансовых инструментов, таких как акции, облигации, валюты (включая криптовалюты), фьючерсы, опционы и другие производные продукты. Цель трейдера – извлечение прибыли из краткосрочных колебаний рынка. Данный процесс требует быстрого принятия решений, анализа больших объемов информации и глубокого понимания рыночных механизмов.

2.1.1. Технический анализ в трейдинге

Технический анализ — это метод прогнозирования будущего движения цен на финансовых рынках на основе изучения прошлых рыночных данных, в основном цен и объемов торгов. Эти данные отображаются в виде графиков, которые технические трейдеры анализируют с целью обнаружения закономерностей и трендов.

Виды технического анализа:

1. Графический анализ (charting) –использование линейных графиков, столбчатых и японских свечных графиков для визуального выявления тенденций и образований (например, ”голова и плечи”, ”двойная вершина” или ”флаг”)

2. Индикаторы и осцилляторы – математические преобразования ценовых данных для создания дополнительных инструментов анализа (например, скользящие средние, MACD, индекс относительной силы – RSI, стохастик)

3. Теории волн и циклов – концепции, предложенные такими аналитиками, как Ральф Нельсон Эллиотт и Уильям Делберт Ганн,

основанные на повторяющихся циклах и волновых паттернах

4. Исследование объемов – анализ объемов торгов для понимания силы или слабости тренда.

Технический анализ помогает трейдерам идентифицировать тренды и моменты разворота рынка, разрабатывать стратегии входа и выхода из позиций, а также управлять рисками.

Также стоит особо выделить вклад машинного обучения. Эта инновационная область применения искусственного интеллекта открывает перед трейдерами новые возможности для анализа финансовых рынков и принятия более информированных решений.

Машинное обучение обогащает традиционные методы технического анализа, предоставляя инструменты для автоматизированного распознавания сложных шаблонов и зависимостей, которые могут быть неочевидны человеческому глазу. При помощи алгоритмов машинного обучения можно анализировать огромные массивы исторических данных по ценам, объемам и другим рыночным индикаторам, чтобы выявлять закономерности, способные повторяться в будущем.

2.2. Обзор существующих решений

Для поиска существующих решений была проанализирована первая страниц поиска в Google по запросам "Technical trading analysis tool" и "Open source technical trading analysis tool".

Были опредлелены недостатки существующих решений:

- Недостаточная функциональность: TA-Lib (консольное приложение, использование низкопроизводительного Python) [8], OpenCharts (генерация фото вместо графического интерфейса, отсутствие Bar plot, использование низкопроизводительного Python) [6].

- Закрытый исходный код, платный доступ к полной функциональности инструмента: TradingView [9], CoinMarketCap [1], CryptoCompare [3], CoinMetrics [2].

На основе вышеуказанных аргументов можно сделать вывод о значимых недостатках существующих решений для технического анализа данных с бирж. Наличие открытых и более доступных альтернатив может значительно повысить доступность и эффективность биржевого анализа для всех участников рынка. Открытый исходный код не только избавляет от вышеперечисленных недостатков, но и способствует формированию сильного сообщества, которое в совокупности может эффектно развивать и улучшать инструменты, делая их более надежными и функциональными.

3. Проектирование инструмента

В данном разделе описана архитектура и дизайн разрабатываемого инструмента. Цель этой части работы – обеспечить технический фундамент, который послужит основой для последующей разработки, тестирования и внедрения инструмента.

3.1. Требования к инструменту

Требования к инструменту технического анализа торговли на биржах на осенний семестр

1. Хранение данных о сделках с использованием концепции баров (японских свечей)
2. Возможность импорта информации о сделках из csv формата
3. Возможность экспорта информации о сделках в csv формат
4. Агрегация сделок в объекты согласно правилам отбора: временным и тиковым
5. Получение и обработка данных о сделках с помощью WebSocket и ByBit API

3.2. Сценарий использования инструмента конечным пользователем

Для иллюстрации сценариев использования нашего инструмента технического анализа данных с бирж была выбрана диаграмма вариантов использования (use-case diagram) на рис. 1.

3.3. Архитектура инструмента

Для демонстрации архитектуры инструмента была выбрана UML-диаграмма классов на рис. 2, так как она позволяет взглянуть на си-

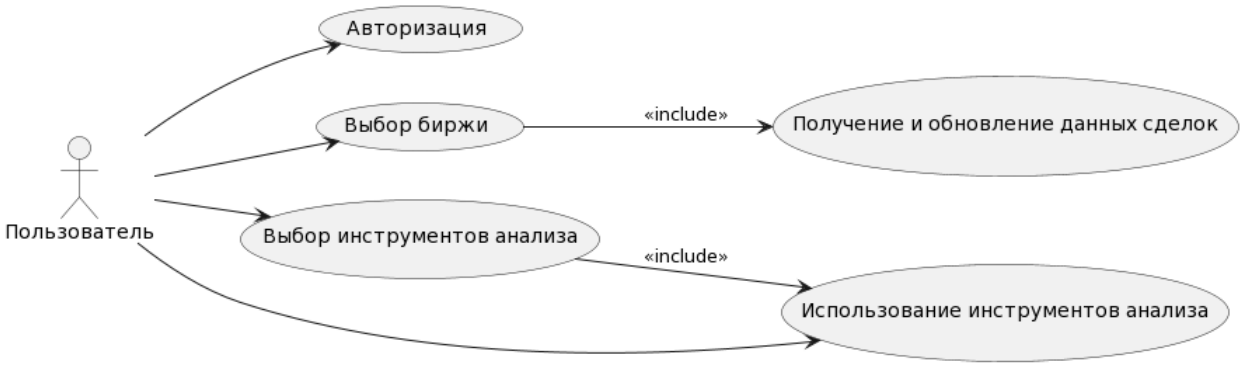

Рис. 1: Use-case диаграмма. На диаграмме указаны пользователь, а также функциональность в виде овалов с пояснением

стему в целом, облегчая понимание взаимодействия между различными частями программы.

3.4. Выбор технологий

Для разработки инструмента был выбран язык C++ по следующим причинам:

- **Производительность и контроль**: C++ известен своей высокой производительностью и эффективностью, что является критически важным для обработки больших объемов биржевых данных и выполнения операций в реальном времени. Язык дает разработчику контроль над низкоуровневыми аспектами системы, такими как управление памятью и системные ресурсы.

- **Популярность и зрелость**: C++ – это зрелый язык с богатой экосистемой и широким сообществом. Имеются множество стабильных и проверенных библиотек для различных аспектов финансового программирования, что уменьшает риски и ускоряет процесс разработки.

Для работы с непрерывным подключением к биржам была взята библиотека WebSocket++ [10], которая представляет собой многопоточный инструмент для достижения наших целей, при этом этот инстру-

Рис. 2: UML диаграмма. Реализованные классы, приватные поля помечены красным цветом, публичные – зеленым, связи в стандартной UML-нотации

мент обладает хорошей документаций, что сильно упрощает разработку и интеграцию.

Библиотека для работы с данными в формате JSON была взята JsonCpp [11]. Это надежное и легковесное решение для наших задач, что является очень важным критерием в контексте целей высокой производительности нашего инструмента.

Для Unit-тестирования была отобрана популярная библиотека для тестирования C++ проектов GoogleTest [4].

4. Ход работы

4.1. Реализация стуктур для хранения данных с бирж

Для хранения данных о финансовых сделках был реализован класс Trade. Класс Trade описывает отдельную сделку на финансовом рынке и содержит следующие поля:

- **timestamp** – значение временной метки Unix, которое отражает время совершения сделки
- **price** – цена сделки, по которой была проведена операция
- **size** – количество активов, участвующих в сделке
- **volume** – общий объем сделки, рассчитанный как произведение цены на количество
- **direction** – направление сделки, где положительное значение означает покупку, а отрицательное – продажу

Для удобства ввода и вывода данных о сделках были реализованы функции операторов ввода и вывода, позволяющие считывать сделки прямо из файла.

Эти функции упрощают процесс сериализации и десериализации объектов Trade, делая класс гибким и удобным инструментом для работы с данными о сделках на финансовых рынках.

Концепция баров относится к методу графического представления данных о торговле на рынке за определённый период времени и является одним из фундаментальных подходов в техническом анализе. Каждый бар (или свеча, если рассматривать свечной анализ) представляет собой информацию о ключевых характеристиках сделок в заданный промежуток времени, количества сделок, объема сделок и т.д. Именно поэтому мы используем этот способ сбора статистики для анализа в нашем инструменте.

Был реализован класс Bar. Класс Bar представляет собой агрегатную информацию за определенный временной, тиковый, долларовый интервал и содержит следующие характеристики для анализа

- **open** – цена открытия
- **close** – цена закрытия
- **high** – максимальная цена сделки за интервал
- **low** – минимальная цена сделки за данный период
- **volume** – общий объем торгов за интервал
- **size** – суммарное количество активов, участвующих во всех сделках
- **signed_volume, signed_size** – объем и размер сделок с учетом роста/падения цены
- **directed_volume, directed_size** – объем и размер сделок с учетом купли/продажи
- **vwap** – объемно-взвешенная средняя цена (VWAP) для интервала

Класс Bar облегчает работу с различными группами сделок и анализ финансовых данных, предоставляя возможность наблюдения за динамикой рынка в торговых интервалах.

4.2. Реализация аггрегаторов сделок

Для агрегирования данных о торговых операциях в рамках курсовой работы было решено использовать абстрактный класс BarAggregator, основанный на шаблоне проектирования "Aggregator" из enterprise integration patterns [5].

Шаблон проектирования "Aggregator" представляет собой механизм, который получает входящие из различных источников сообщения, обрабатывает их и формирует одно агрегированное сообщение на основе полученной информации [3]

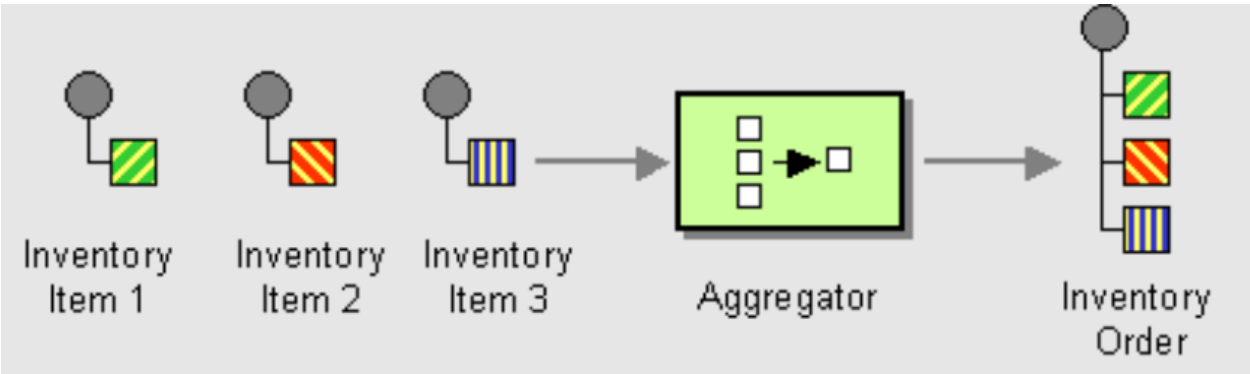

Рис. 3: шаблон Aggregator.

Класс BarAggregator является абстрактным классом, что подразумевает, что методы его не реализованы на данном уровне, они предназначены для переопределения в подклассах. Использование абстрактного класса позволяет определить общий интерфейс для различных видов агрегаторов без привязки к конкретной логике агрегирования. Это обеспечивает гибкость и расширяемость системы, позволяя легко добавлять новые способы агрегирования данных.

Структура класса BarAggregator включает в себя следующие элементы:

- **Поле bar_queue** – очередь, хранящая сформированные объекты Bar, готовые для публикации или дальнейшей обработки
- **Виртуальный метод receive** обрабатывает и агрегирует информацию о сделках
- **Виртуальный метод ready**, определяющий, готов ли агрегатор к публикации
- **Виртуальный метод publish** публикует бары из очереди
- **Статический метод process**, которые сопсобен обработать все сделки из файла, путь до которого передается в функцию

Итого, благодаря шаблону "Aggregator", разработанный класс упрощает обработку и агрегацию больших массивов торговых данных, позволяя адаптировать логику агрегирования в зависимости от потребностей пользователей или особенностей торгового процесса.

4.3. Реализация получения данных с бирж

WebSocket – это протокол связи, который предоставляет полнодуплексный канал связи по единственному TCP-соединению. В контексте торговых платформ, таких как Bybit, WebSocket играет важную роль, так как он позволяет передавать данные о торговых сделках в реальном времени, что критично для актуального отображения рыночной информации и для инструментов, осуществляющих анализ сделок. Использование WebSocket обеспечивает немедленный приём данных о цене, объёме сделок и иных важных торговых параметрах, следовательно, трейдеры и аналитики могут мгновенно реагировать на изменения рынка.

Для взаимодействия с торговой платформой Bybit был реализован класс BybitWebSocketClient. Этот класс отвечает за установление и управление WebSocket соединением, а также за приём данных о сделках и их агрегацию. Структура класса представляет собой следующие ключевые элементы:

- **Метод connect** – запускает процесс подключения к серверу Bybit по заданному URI и запускает клиент WebSocket
- **Метод get_trades** – возвращает вектор с агрегированными торговыми операциями

Реализация класса BybitWebSocketClient позволяет обеспечить надёжную и своевременную доставку данных о торговых сделках с Bybit, что важно для обеспечения точности и своевременности выполняемого анализа сделок

4.4. Модульное тестирование компонентов

Юнит-тестирование было выбрано как метод тестирования на уровне отдельных модулей исходного кода программного продукта с целью проверки правильности их работы отдельно от других частей системы. В качестве фреймворка для реализации юнит-тестов использовался Google Test (gtest).

Информация о покрытии кода тестами [4]

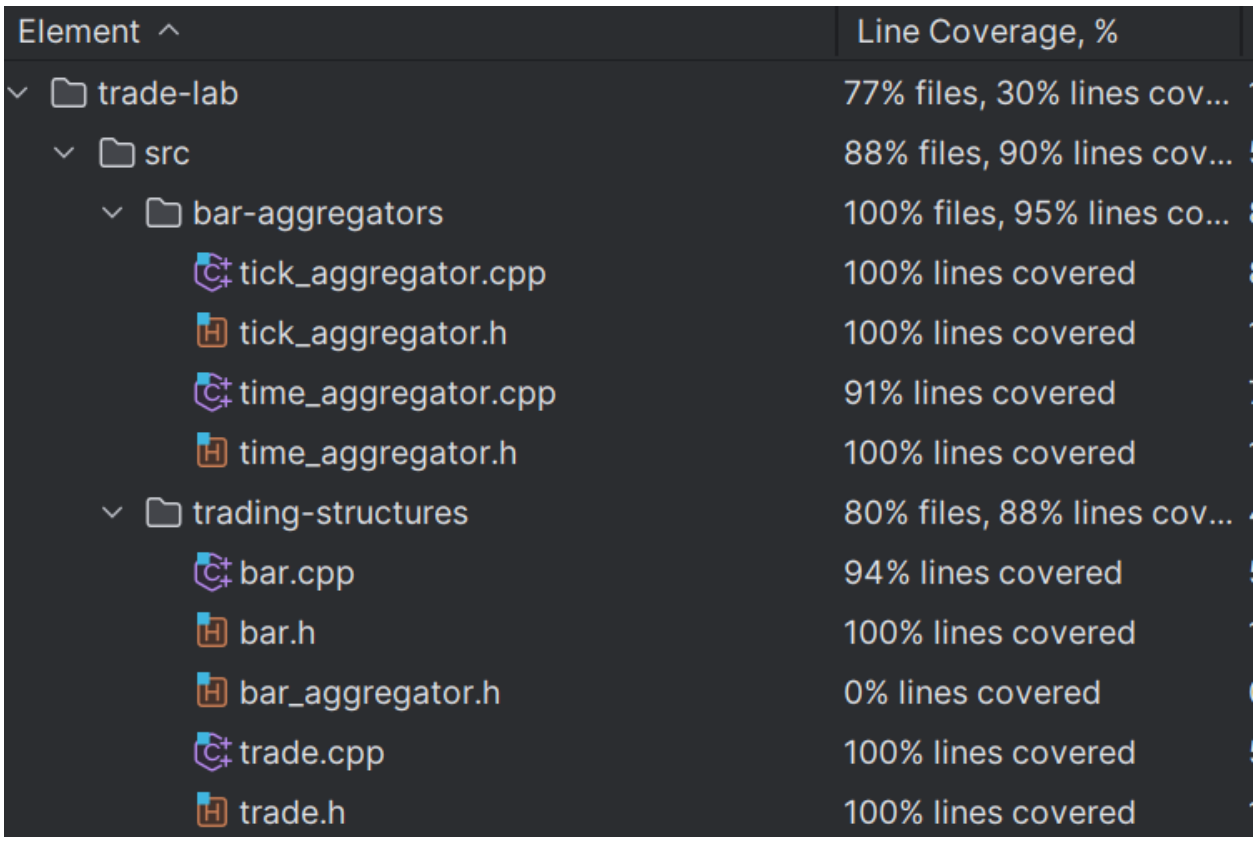

Рис. 4: Покрытие кода тестами.

5. Реализация

Реализация инструмента доступна в репозитории GitHub. [7].

Заключение

В течение осеннего семестра в рамках производственной практики были выполнены следующие задачи.

1. Проведён общий обзор предметной области и существующих решений
2. Определены требования к инструменту
3. Спроектирована архитектура инструмента, а также выбраны технологии для реализации
4. Разработана система структур данных для эффективного хранения и обработки информации с бирж
5. Реализован программный интерфейс для установления соединения и агрегации данных в реальном времени с использованием WebSocket
6. Проведено модульное тестирование иструмента

Таким образом, в будущем предстоит улучшить инструмент:

1. Реализовать возможность получения данных о сделках с различных сервисов
2. Отобрать и реализовать различные средства технического анализа бирж
3. Спроектировать и имплементировать графический интерфейс для взаимодействия с инструментом
4. В перспективе добавить возможность использования машинного обучения для дополнительного технического анализа

Список литературы

[1] CoinMarketCup. CoinMarketCup. — URL: https://coinmarketcap.com/ru/.

[2] CoinMetrics. CoinMetrics. — URL: https://coinmetrics.io/.

[3] CryptoCompare. CryptoCompare. — URL: https://www.cryptocompare.com/.

[4] Google. GoogleTest Library. — URL: https://github.com/google/googletest.

[5] Hohpe Gregor. Enterprise Aggregation Pattern. — URL: https://www.enterpriseintegrationpatterns.com/patterns/messaging/Aggregator.html.

[6] OpenCharts. OpenCharts. — URL: https://github.com/bodealamu/opencharts.

[7] Shangareev Rustam. Github repository with implementations. — URL: https://github.com/letit6E/trade-lab.

[8] Ta-Lib. Ta-Lib. — URL: https://github.com/TA-Lib/ta-lib-python.

[9] TradingView. TradingView. — URL: https://ru.tradingview.com/.

[10] Websocket++ Library. — URL: https://github.com/zaphoyd/websocketpp.

[11] source parsers Open. JsonCpp Library. — URL: https://github.com/open-source-parsers/jsoncpp.