Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Институт естественных и точных наук
Факультет математики, механики и компьютерных технологий
Кафедра прикладной математики и программирования
Направление подготовки: 01.04.02 Прикладная математика и информатика

РАБОТА ПРОВЕРЕНА
Рецензент профессор кафедры УМФ,
д.ф.-м.н. доцент
_____/ Н.А. Манакова
«____» _____20___г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой, д.ф.-м.н.,
доцент
_____/А.А. Замышляева
«____» _____20___ г.

Автоматизация торговли на криптовалютной бирже

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ–01.04.02.2019.035.ПЗ ВКР

Руководитель работы, доцент
_____/ А.А. Замышляева
«____» _____2019 г.
Автор работы
Студент группы ЕТ-222
_____/ А.В. Кених
«____» _____2019 г.
Нормоконтролер, ассистент
_____/ Н.С. Мидоночева
«____» _____2019 г.

Челябинск
2019

АННОТАЦИЯ

Кених А.В. Автоматизация торговли на криптовалютной бирже. – Челябинск: ЮУрГУ, ЕТ-222, 64 с., 25 ил., 4 табл., библиогр. список – 16 наим., 5 прил.

Целью данной работы является автоматизации торговли на криптовалютной бирже. В работе выполнен обзор криптовалют и криптовалютных бирж, были рассмотрены основные принципы торговли на криптовалютных биржах. Выполнено математическое моделирование торговли на криптовалютной бирже. Спроектирована и разработана архитектура системы.

Программа реализована на языке программирования TypeScript с использованием СУБД MongoDB.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность данной темы очевидна. Накопление денежного капитала играет важную роль в капиталистическом хозяйстве. Непосредственно самому процессу накопления денежного капитала предшествует этап его производства. После того как денежный капитал создан или произведен, его необходимо разделить на часть, которая вновь направляется в производство, и ту часть, которая временно высвобождается. Последняя, как правило, и представляет собой сводные денежные средства, аккумулируемые на рынке.

Возникновение и обращение капитала, представленного в виде криптовалюты, тесно связанно с функционированием рынка реальных активов, т. е. рынка, на котором происходит купля-продажа материальных ресурсов. С появлением криптовалюты происходит как бы раздвоение капитала. С одной стороны, существует реальный капитал, представленный производственными фондами, с другой стороны – его отражение в виде криптовалюты.

На современном этапе развития экономики актуализируются проблемы повышения эффективности управления финансовыми ресурсами – в том числе и криптовалютными активами. Весьма целесообразно и необходимо рассмотреть возможные способы использования криптовалюты посредством операций с ними на специализированных криптовалютных биржах.

1 ОСНОВНЫЕ ПРИНЦИПЫ ТОРГОВЛИ НА КРИПТОВАЛЮТНОЙ БИРЖЕ

1.1 Постановка задачи

Целью данной работы является создание автоматизированной системы для торговли на криптовалютной бирже. Для достижения поставленной цели необходимо решить следующие задачи:

- ознакомиться с криптовалютами и криптовалютными биржами;
- рассмотреть правила торговли на криптовалютных биржах;
- разработать требования к приложению;
- смоделировать поведение системы;
- разработать архитектуру приложения в целом;
- реализовать и отладить программу;
- продемонстрировать работу разработанного приложения.

1.2 Криптовалюты и криптовалютные биржи

Криптовалюта – это разновидность цифровой валюты, создание и контроль за которой базируются на криптографических методах. Криптовалюты в отличие от настоящих средств не имеют физического выражения. Ключевой особенностью криптовалют является отсутствие какого-либо внутреннего или внешнего администратора. Поэтому банки, налоговые, судебные и иные государственные или частные органы не могут воздействовать на транзакции каких-либо участников платёжной системы.

Существует два основных способа получения той или иной криптовалюты:

- покупка – приобрести за деньги электронную валюту можно на специальных сервисах по обмену или криптовалютных биржах;
- майнинг – процесс получения криптовалюты путем использования вычислительных данных мощностей.

Для того, чтобы разобраться в сущности криптовалют, стоит выделить их преимущества и недостатки.

Перечислим положительные черты криптовалют.

1. Анонимность. В отличие от классических электронных денег, операции с которыми легко отслеживаются, получить информацию о хозяине криптовалютного кошелька не получится. Доступен только номер бумажника и ограниченные данные по сумме на счету.

2. Децентрализация. Криптовалюты являются независимой денежной единицей. Их эмиссию никто не регулирует, никто не контролирует движение средств на счету. Именно эта особенность привлекает многих участников сети.

3. Ограниченность. Как правило, криптовалюта выпускается в ограниченном объеме, что исключает риски инфляции из-за чрезмерной активности эмитента.

4. Надежность. Взломать, подделать или осуществить другие подобные манипуляции с виртуальной валютой не выйдет – она надежно защищена.

Теперь перечислим отрицательные черты криптовалют.

1. Отсутствие гарантий. Каждый пользователь персонально несет ответственность за свои сбережения. Здесь нет регулирующих механизмов, поэтому в случае кражи доказать что-либо и вернуть деньги не получится.

2. Риск запрета. Государственные структуры с опаской подходят к криптовалютам. Многие страны ввели ограничения по их использованию, а нарушители могут нарваться на штраф или даже тюремный срок. При этом ряд европейских государств еще находится на пути к поиску компромисса в вопросе использования таких денег.

3. Опасность потери. «Ключом» доступа к электронным деньгам является специальный пароль. Если его потерять, находящиеся в кошельке криптовалютные средства становятся недоступны.

Ещё одной чертой криптовалют, которую сложно отнести как к положительной черте, так и к отрицательной, является волатильность.

Криптовалюта является непредсказуемой, ведь зависит от текущего спроса, который, в свою очередь, может меняться на фоне изменений в законодательстве, текущих мнений и прочих факторов. По этой причине имеют место сильные колебания цены виртуальных денег, что может являться как положительным, так и отрицательным фактором.

На сегодняшний день существует несколько тысяч самых разнообразных криптовалют, ниже перечислены первые пять по рыночной капитализации (по данным CoinMarketCap [1]).

Bitcoin [2] – наиболее популярная криптовалюта и, образно выражаясь, родоначальник жанра. Является пиринговой платёжной системой, использующая одноимённую единицу для учёта операций. Для обеспечения функционирования и защиты системы используются криптографические методы, но при этом вся информация о транзакциях между адресами системы доступна в открытом виде. Своим появлением биткоин положил начало развития всех остальных подобных валют. Разработчиком является скрытая группа программистов. Стоит отметить, что создатели оставили свободный код своей разработки, что дало возможность другим специалистам создавать на данной основе уже новые типы криптовалют. Одна из главных особенностей системы – полная децентрализация: нет центрального администратора или какого-либо его аналога. Необходимым и достаточным элементом этой платёжной системы является базовая программа-клиент (имеет открытый исходный код). Запущенные на множестве компьютеров программы-клиенты соединяются между собой в одноранговую сеть, каждый узел которой равноправен и самодостаточен. Невозможно государственное или частное управление системой, в том числе изменение суммарного количества биткойнов, эмиссия биткоина ограничена количеством в 21 миллион и на данный момент этот предел еще не достигнут.

Ethereum [3] – это платформа для создания децентрализованных онлайн-сервисов на базе блокчейна, работающих на базе умных контрактов. Реализована как единая децентрализованная виртуальная машина. Был

предложен Виталиком Бутериным в конце 2013 года, сеть была запущена 30 июля 2015 года. Являясь открытой платформой, эфириум значительно упрощает внедрение технологии блокчейн, что объясняет интерес со стороны не только новых стартапов, но и крупнейших разработчиков ПО, таких как Microsoft, IBM и Acronis. Заметный интерес к платформе проявляют и финансовые компании, включая Сбербанк. В отличие от других криптовалют, авторы не ограничивают роль эфира платежами, а предлагают его, например, в качестве средства для обмена ресурсами или регистрации сделок с активами при помощи умных контрактов, в частности авторы назвали эфир «криптотопливом» для исполнения умных контрактов одноранговой сетью.

Litecoin [4] – форк Bitcoin, пиринговая электронная платёжная система, использующая одноимённую криптовалюту. Данная денежная единица была основана программистом Чарли Ли как альтернатива для Bitcoin, программный код которого был взят за основу. Выпуск лайткоин был начат в 2011 году. Основной замысел данного творения в том, чтобы стать своеобразным аналогом серебра на рынке цифровых финансов. это одноранговая интернет-валюта, которая обеспечивает мгновенные, практически нулевые платежи для всех в мире. Litecoin отличается более быстрым временем подтверждения транзакций и более высокой эффективностью хранения, чем ведущая валюта Bitcoin. Эмиссия Litecoin ограничена 84 миллионами.

Zcash [5] – это криптовалюта с открытым исходным кодом, разработанная компанией Zerocoin Electric Coin Company, обеспечивающая конфиденциальность и выборочную прозрачность транзакций. Как и Bitcoin, данные транзакций Zcash публикуются в общедоступной цепочке блоков, но в отличие от биткойнов, Zcash гарантирует, что ваши личные данные и данные транзакций останутся полностью конфиденциальными. Доказательства с нулевым разглашением позволяют проверять транзакции, не раскрывая сумму отправителя, получателя или транзакции. Функции выборочного раскрытия в Zcash позволяют пользователю обмениваться

некоторыми деталями транзакции в целях обеспечения соответствия или аудита. Как и Bitcoin, Zcash имеет фиксированный общий запас в 21 миллион единиц. Впервые валюта анонсирована 20 января 2016 года.

Ripple [6] – цифровой актив для проведения платежей, внутренняя криптовалюта платформы Ripple. Её создавали для предприятий, а в итоге XRP стал любимчиком банков. Объяснить сложившуюся ситуацию просто: монета является палочкой-выручалочкой при проведении трансграничных платежей. Вдобавок XRP снижает расходы на обмены валют и помогает компаниям заходить на новые рынки.

Как уже говорилось, один из способов получения той или иной криптовалюты является покупка за реальные деньги. Приобрести можно как на специализированных обменных системах, так и на криптовалютных биржах.

Современная биржа цифровых валют – это определенная площадка, специализированно предназначающаяся для торговли, обмена цифровых денег одного типа на другие, либо для обмена на общепринятые мировые валюты. На всех биржах есть пары криптовалюта/криптовалюта, например, можно купить Bitcoin за Ethereum, но далеко не все биржи работают с общепринятыми мировыми валютами, то есть нельзя купить Bitcoin за рубли.

Стоит отметить, что объем торгов на бирже может сказаться на волатильности. Чем выше объем торгов, тем ниже волатильность и меньше возможностей для манипулирования рынком. Если мы пытаемся купить один биткоин по цене, скажем, $ 1000, и объемы торгов на бирже велики, высока вероятность, что сделка произойдет почти сразу. Если мы попытаемся приобрести криптовалюту на площадке с низкими объемами, цена покупки может значительно вырасти, поскольку мы скупим все предложение по текущей цене и перейдем на следующий уровень котировок. В результате мы потратим больше денег, а цена биткоина на этой бирже возрастет.

Также стоит отметить, что ни одна криптовалютная биржа не защищена от хакерских атак полностью. За короткую историю криптовалютного рынка случались различные взломы, однако во многих

случаях биржи компенсировали потери клиентов из своего кармана. Децентрализованные биржи невозможно взломать, однако пользователям приходится самим отвечать за сохранность капиталов. Популярные централизованные площадки не уступают по безопасности.

Рассмотрим несколько функционирующих криптовалютных бирж, которые сегодня пользуются наибольшей популярностью и опытных трейдеров.

Binance [7] – достаточно молодая криптовалютная биржа, её запуск состоялся весной 2017 года. Но менее чем за год она приобрела популярность у широкого круга трейдеров по всему миру, и на данный момент входит в ТОП-5 крупнейших криптовалютных бирж мира. Ежедневный торговый оборот биржи составляет более 2,5 миллиардов долларов. Главные достоинства: комиссия за сделку всего 0,1%; отсутствует комиссия на ввод любой криптовалюты; верификация не обязательна; биржа поддерживает торговлю 87 криптовалютами; наличие приложения для торговли на мобильном устройстве; удобный и быстрый интерфейс; наличие API и документации к нему.

Bittrex [8] – работает с 2015 года и позволяет работать с тысячами криптовалютных пар, имеет двухфакторную авторизацию и холодное хранение активов большинства пользователей для защиты от возможных системных сбоев. Главные достоинства: взимает комиссию за сделки в размере 0,25% от их суммы; большой суточный объем торгов; наличие API и документации к нему.

Poloniex [9] – является одной из самых крупных бирж криптовалют в мире. Она обеспечивает самые высокие объемы торгов для большинства альтернативных монет. Главные достоинства: простой и доступный интерфейс, несмотря на отсутствие русифицированной версии сайта; большие обороты торгов, свыше 140 направлений обмена; низкие комиссии системы за транзакции – от 0 до 0,2%; высокий уровень безопасности, двухфакторная

аутентификация; удобные аналитические и технические инструменты; наличие API и документации к нему.

YoBit [10] – это криптовалютная биржа с большим количеством торговых пар. Имеет множество разных дополнительных функций. Главные достоинства: поддерживает более 400 валютных пар; низкие комиссии за ввод и вывод как криптовалют, так и рублей и долларов; мгновенный ввод/вывод криптовалют; удобное меню, поддержка русского языка; наличие API и документации к нему.

Exmo [11] – является лидирующей биржей по торговле криптовалютами на территории СНГ. А также она является единственной биржей, на которой пары торгуются к рублю и имеют высокие обороты торговли. Такая активность применения рубля обусловлена в основном русскоязычной аудиторией биржи, хотя география пользователей биржи огромна: это и Европа, и американские страны, и азиатские участники. Биржа удобная и простая в использовании, поддерживает более 10 языков интерфейса и, что очень важно, работает с фиатными валютами и платежными системами России и Украины. Как и другие биржи имеет API и документацию к нему.

Это не весь перечень популярных криптовалютных бирж, а только наиболее популярные. Есть и другие наименее популярные биржи со своими минусами и плюсами, но стоит отметить, что практически каждая крупная биржа имеет API и документацию к нему, что позволяет автоматизировать процесс торговли на любой из них.

1.3 Торговля криптовалютой

XXI век – уникальная эпоха новых возможностей, стремительных перемен. В жизнь каждого все больше проникают разнообразные технологии, все активно компьютеризируется, буквально во всех жизненных сферах можно встретить вездесущий Интернет.

Естественно, что деятельно развивается и финансовая сфера. При этом после появления различных онлайн валют, наиболее популярной из которых стал Биткоин (Bitcoin), произошла настоящая революция.

Данные средства пока еще сильно подвержены скачкам курса. Такая их нестабильность на руку тем, кто желает получить дополнительную прибыль, зарабатывая, например, торговлей на бирже.

Задавшись вопросом, как заработать на бирже криптовалют, стоит узнать подробности о данной процедуре.

Сегодня выделяется несколько видов работы в сфере активной торговли:

- классическая торговля, повторяющая привычные стратегии валютных, фондовых рынков;
- вложение денежных средств на этапе старта или до него в криптовалютный форк или проект;
- использование всевозможных производных биткоина;
- арбитраж между функционирующими системами.

Процесс деятельности на современных рынках цифровых валют аналогичен стандартным валютным, фондовым рынкам. Опытному трейдеру потребуется потратить лишь немного времени для изучения наличествующих особенностей, после чего он сможет сразу начать работать в новых для него условиях. А начинающий пользователь при этом довольно серьезно рискует. Чтобы быстрее освоиться, при этом не получив внушительного убытка, новичку в криптотрейдинге необходимо для начала тщательно изучить парутройку простых стратегий, попробовать использовать которые можно на площадке любой из функционирующих бирж.

2017 год ознаменовался экспоненциальным ростом рынка криптовалют, динамика которого поражает воображение. Многие в прошлом году в очередной раз убедились в том, что доходность вложений в новые активы зачастую привлекательнее результатов инвестирования в акции даже наиболее успешных компаний. Перспективы получения значительной

прибыли на вложенные средства привлекают на криптовалютный рынок все новых участников, включая известных игроков мира традиционных финансов.

Перечислим основные преимущества этого рынка для кратко- и средне-срочной торговли.

1. Работа бирж в режиме 24/7. В отличие от традиционного финансового рынка, криптовалютный рынок лишен многих ограничений. Так, от фондового рынка его отличает то, что криптовалютные биржи работают в режиме 24/7, не имеют высоких барьеров для входа, при этом торговые комиссии сравнительно невелики. Поскольку рынок криптовалют работает круглосуточно и без перерывов, отсутствует необходимость закрывать позиции в конце торгового дня или недели.

2. Низкий порог для входа на рынок. Для входа на валютный рынок следует обладать значительным депозитом, что проблематично для новичков трейдинга. Еще больший капитал требуется для входа на рынок ценных бумаг (для американского фондового рынка это от нескольких десятков тысяч долларов). Для торговли биткоином минимально возможный ордер относительно невелик (обычно от 0,0005 до 0,001 BTC, в зависимости от правил той или иной криптовалютной биржи).

3. Высокая волатильность. Значительные колебания цен привлекают на этот рынок толпы трейдеров. Ни для кого не секрет, что криптоактивы гораздо сильнее подвержены волатильности нежели общепринятые мировые валюты. Например, при торговле ликвидными валютными парами, скажем EUR/USD, волатильность будет составлять примерно 1–2 % в месяц. В то же время внутридневная волатильность биткоина может достигать, 5, 10 и более процентов, что очень привлекательно для трейдера.

Таким образом, торговля криптовалютой позволяет получать прибыль в краткосрочной перспективе. Однако требуется рациональный выбор активов, дисциплина и взвешенность в принятии решений.

Необходимо обозначить перечень наиболее важных правил для осуществления торговли на современном рынке онлайн валют, позволяющих оставаться в прибыли. Из-за постоянно растущей популярности криптовалют, несмотря на наличие определенных решений Центробанка, у множества людей появляется желание заработать, торгуя данными средствами. Однако подавляющее большинство быстро теряют собственные денежные средства, потому как не придерживаются основных правил осуществления рабочего процесса на функционирующих ныне площадках.

Приобретать – при падении, реализовать – на росте. На любом функционирующем рынке криптовалют имеются манипуляторы с большим количеством средств, которые тем или иным образом могут резко и внушительно повлиять на ход курса. Поэтому приобретать требуется до старта процесса роста, при фиксировании краткосрочного или долгосрочного падения, иначе велики шансы оказаться в расставленных манипуляторами ловушках. Продавать имеющиеся активы необходимо при наличии роста, когда уровень стоимости оказывается во второй половине возможного пика.

Не стоит гоняться за максимумами или минимумами. Управление курсом, как уже и говорилось, осуществляется манипуляторами, объемы активов которых дают возможность опускать/поднимать стоимостный уровень буквально за пару минут деятельных торгов на десятки процентов. Максимальное ценовое падение либо рост прямо зависим от крупнейших игроков, однако даже они зачастую на некоторых моментах процесса не способны предугадать силу позитивного либо панического настроения остальных пользователей. Соответственно, не стоит выжидать достижения минимальной точки падения, гоняться за мечтой о моментальном высоком росте. Лучшим решением будет занятие фиксированием прибыли на «выше-средней» планке, в противном случае можно вовсе не успеть реализовать сделку.

Не стоит слушать и безосновательно верить в биржевой чат. Подавляющее большинство опубликованных в биржевом чате прогнозов формируется

на основе индивидуальных ожиданий, либо, что еще хуже, проведенного технического анализа, попросту не работающего в данных условиях. Специализированная аналитика от известных мировых рынков, имеющих многомиллиардные обороты, разнообразные форекс стратегии с криптовалютами не работают, ведь всего одному манипулятору, имеющему «в кармане» $ 25 миллионов, доступно в определенный момент обрушить, либо поднять текущий курс на несколько десятков процентов лишь за полчаса.

1.4 Функциональные требования

В ходе знакомства с криптовалютами и криптовалютными биржами было выявлено, что существует большое и разнообразное количество криптовалют, а также существует немалое количество различных криптовалютных бирж, почти каждая из которых имеет свой API и документацию к нему. Следовательно, система не должна ограничиваться одной криптовалютной парой или одной биржей. Также не стоит ограничивать систему одной стратегией торговли.

Ввиду того, что торговля ведется в Интернет сети на криптовалютных биржах, не стоит забывать о том, что возможно появление неполадок с Интернет-соединением. Помимо этого, возможно неожиданное завершение работы по некоторым причинам. Таким образом, система должна иметь возможность возобновлять работу после остановки или неожиданного завершения работы.

, требуется разработка системы, которая:

- позволит взаимодействовать с криптовалютными биржами;
- позволит выбирать произвольные криптовалютные пары;
- позволит подключать и использовать различные стратегии торговли, которые могут быть разработаны отдельно от данной системы;

- будет иметь возможность возобновлять работу после остановки или неожиданного завершения работы.

1.5 Выбор и обоснование средств разработки

Для реализации поставленной задачи мы будем использовать Node.js [12], TypeScript [13] и MongoDB [14].

Node.js – программная платформа, транслирующая JavaScript в машинный код. Сам по себе Node.js является средой выполнения JavaScript, построенной на JavaScript-движке Chrome V8, который позволяет компилировать исходный код JavaScript непосредственно в собственный машинный код, минуя стадию промежуточного байт-кода. Node.js использует управляемую событиями, неблокирующую модель ввода/вывода, которая делает его легким и эффективным. Node.js имеет самую большую экосистему библиотек с открытым исходным кодом в мире – npm, а также позволяет подключать другие внешние библиотеки, написанные на разных языках программирования. Npm – менеджер пакетов, входящий в состав Node.js. Это библиотеки, построенные сообществом, которые можно добавить в свои приложения, чтобы сделать разработку более быстрой и эффективной. Один из главных плюсов Node.js – он понимает JSON. JSON является, вероятно, самым важным форматом обмена данных в Интернете и взаимодействия с NoSQL базами данных. Данный факт очень важен при решении поставленной задачи.

В связке с Node.js мы будем использовать TypeScript.

Во-первых, следует отметить, что TypeScript – это строго типизированный и компилируемый язык. Хотя на выходе компилятор создает все тот же JavaScript, однако строгая типизация уменьшает количество потенциальных ошибок, которые могли бы возникнуть при разработке на JavaScript. Строгая типизация имеет несколько преимуществ: описав сигнатуры через типы, можно отловить много ошибок на момент компиляции. Например, если мы

описали, что метод принимает число, а в метод передаем строку, то TypeScript компилятор нам сразу об этом сообщит. Кроме этого, описание типов позволяет IDE давать более точные подсказки и помогает писать код быстрее. Для других языков программирования строгая типизация улучшает производительность, но для TypeScript это не актуально, так как в результате компиляции генерируется JavaScript код, в котором аннотации типов отсутствуют. Строгая типизация имеет также и свои недостатки: больше времени надо на описание типов, типы иногда ухудшают читабельность кода, иногда есть проблемы с преобразованием одного типа в другой. Однако в TypeScript типизация опциональна. На уровне компилятора у каждой переменной/параметра есть тип, при этом если в коде он не указан, то предполагается тип any, что значит переменная может принимать значение любого типа и TypeScript это допускает.

Во-вторых, TypeScript реализует многие концепции, которые свойственны объектно-ориентированным языкам, как, например, наследование, полиморфизм, инкапсуляция и модификаторы доступа и так далее.

В-третьих, потенциал TypeScript позволяет быстрее и проще писать большие сложные комплексные программы, соответственно их легче поддерживать, развивать, масштабировать и тестировать, чем на стандартном JavaScript.

В-четвертых, TypeScript является кроссплатформенным, а это значит, что для разработки мы можем использовать как Windows, так и MacOS или Linux.

В TypeScript можно использовать все те конструкции, которые применяются в JavaScript – те же операторы, условные, циклические конструкции. Более того код на TypeScript компилируется в JavaScript. В конечном счете, TypeScript – это всего лишь инструмент, который призван облегчить разработку приложений.

TypeScript ориентируется прежде всего на стандарт ECMAScript 3, хотя TypeScript также поддерживает и стандарты ECMAScript 5 и ECMAScript

2015/2017, а в процессе разработки мы можем сами задать целевой стандарт ECMAScript.

В качестве базы данных была выбрана MongoDB. Традиционно под базой данных понимается система управления реляционной базой данных, однако в последние годы в моду вошли документированные базы данных. Документированные базы данных лучше подходят для хранения объектов, что делает их естественным дополнением для Node.js. MongoDB – ведущая документированная база данных, общепризнанная и очень надежная. Для работы с MongoDB мы будем использовать Mongoose [15], который предоставляет простое решение на основе схем для моделирования данных приложения. Он включает встроенное приведение типов, проверку, построение запросов, хуки бизнес-логики и многое другое.

1.6 Выводы по разделу

Стремительный технический прогресс открывает нам огромное количество технологий и возможностей. Одним из новшеств технического прогресса является криптовалюта – разновидность цифрой валюты, создание и контроль за которой базируются на криптографических методах. Не удивительно, что образовалось большое количество криптовалютных бирж – это место, где люди обменивают одни криптовалюты на другие, либо на основные мировые валюты (доллар, евро, рубли и др.). Криптовалюты пока еще сильно подвержены скачкам курса – и это ни для кого не секрет. Такая их нестабильность – отличная возможность для того, чтобы заработать. К тому же, криптовалютные биржи, в отличии от фондового рынка, работают в режиме 24/7, а также многие популярные криптовалютные биржи имеют собственное API и документацию к нему, что позволяет автоматизировать процесс торговли. Придерживаясь несложных правил торговли, а также ознакомившись с API той или иной биржи, можно создать собственную стратегию и автоматизировать её. Таким образом получится инструмент, который будет работать постоянно в режиме реального времени,

придерживаясь той или иной стратегии. В данном разделе мы сформулировали функциональные требования для приложения и выбрали средства для разработки: Node.js, TypeScript и MongoDB.

2 МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ ТОРГОВЛИ НА КРИПТОВАЛЮТНОЙ БИРЖЕ

2.1 Общий принцип работы стратегии

В определенное время курс той или иной пары может иметь восходящий и нисходящей тренд. Стоит учитывать, что направление тренда вовсе не обозначает строгий рост или падание курса и сопровождается колебаниями курса. Именно эти колебания и позволяют фиксировать прибыль. Поскольку имеется два направления тренда (восходящий и нисходящий), то мы можем реализовать два подтипа стратегии, у которых будет следующее поведение:

- сначала продать, затем приобрести;
- сначала приобрести, затем продать.

Рассмотрим первый случай. Логично, что в данном случае необходимо будет продать валюту по высокому курсу, а затем приобрести её по более низкому курсу. Для такого поведения подходит восходящий тренд: рост на определенное количество процентов будет сменяться небольшим спадом, после которого вновь пойдет рост (рисунок 2.1). Таким образом, мы сможем сначала продавать по более высокому курсу, а затем приобретать по более низкому курсу.

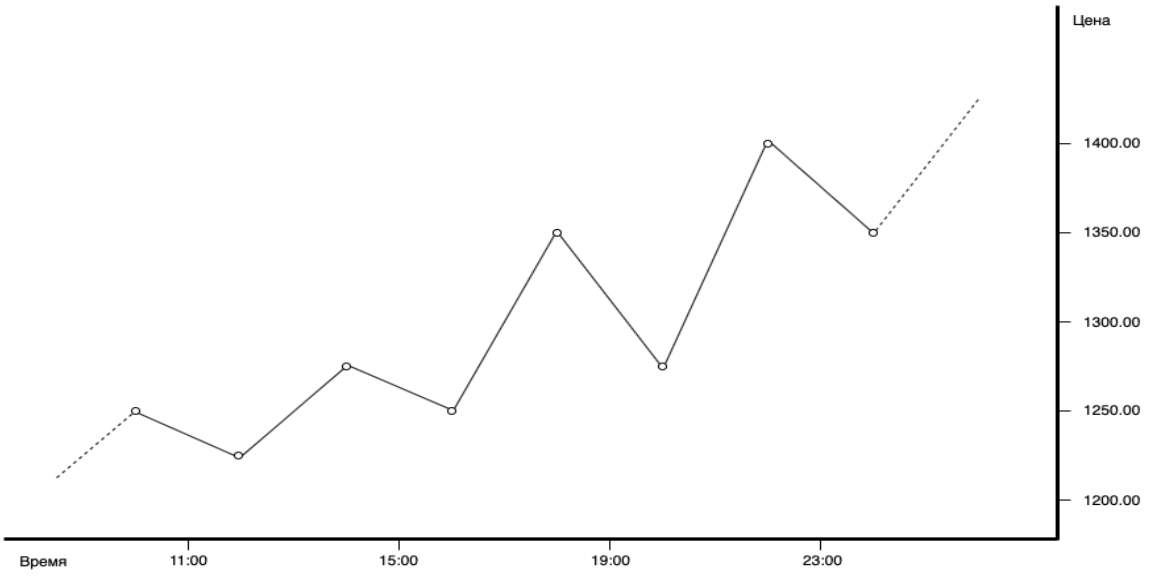

Рисунок 2.1 – Пример восходящего тренда

Рассмотрим второй случай. Логично, что в данном случае необходимо будет приобретать валюту по низкому курсу, а затем продавать её по более высокому курсу. Для такого поведения подходит нисходящий тренд: спад на определенное количество процентов будет сменяться небольшим ростом, после которого вновь пойдет спад (рисунок 2.2). Таким образом, мы сможешь сначала приобретать по более низкому курсу, а затем продавать по более высокому курсу.

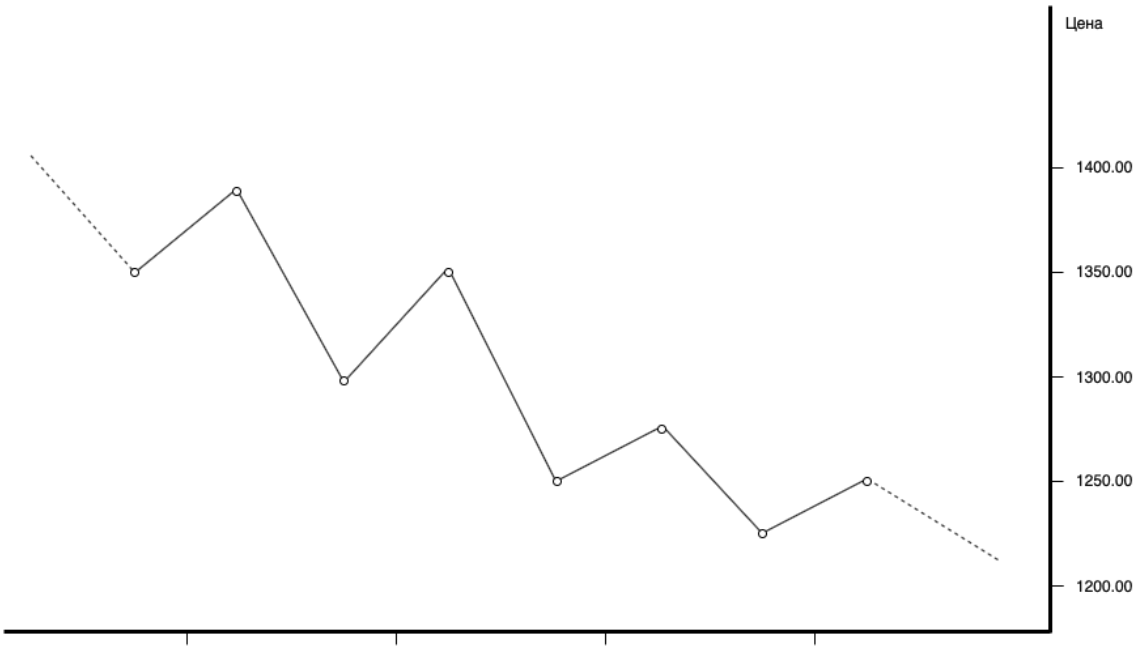

Рисунок 2.2 – Пример нисходящего тренда

Стоит обратить внимание на то, что торговля одним ордером не очень эффективна. Например, покупка актива одним ордером и выставление его на продажу по более высокой цене тоже одним ордером. В таком случае подразумевается, что цена всегда должна подскакивать вверх от закупки, а в случае падения цены, такой алгоритм бессилен. Поэтому более целесообразней выполнять торговлю не одним ордером, а несколькими. Для этого мы можем выполнить декомпозицию используемого депозита, что позволит создать несколько ордеров по направлению тренда. Что это даст? После исполнения одного из таких ордеров, мы создадим реализующий ордер и будем пересоздавать его по мере исполнения остальных ордеров, учитывая их исполненный объем. Таким образом, по мере отдаления цены на

рынке от цены первого ордера, наш реализующий ордер будет словно следовать за ценой на рынке и будет исполнен при отклонении курса.

Рассмотрим поэтапно условный пример торговли несколькими ордерами, в котором мы создадим три ордера на продажу: по цене $ 1250, $ 1300 и $ 1350 (рисунок 2.3).

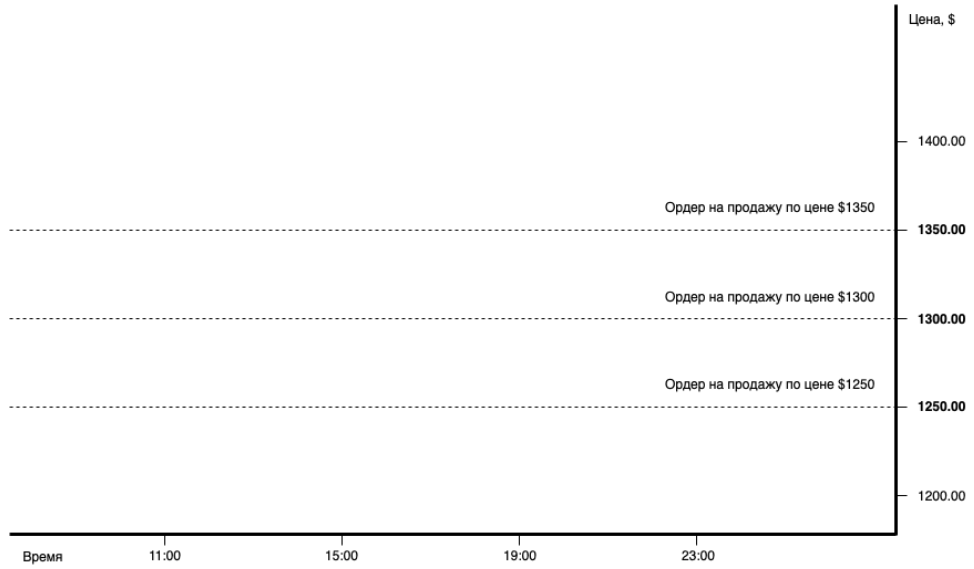

Рисунок 2.3 – Первый этап примера торговли несколькими ордерами

Предположим, что после создание трех ордеров на продажу курс поднялся до цены первого ордера продажи, в результате чего тот был исполнен, а затем был создан реализующий ордер на покупку, но уже по более низкой цене (рисунок 2.4).

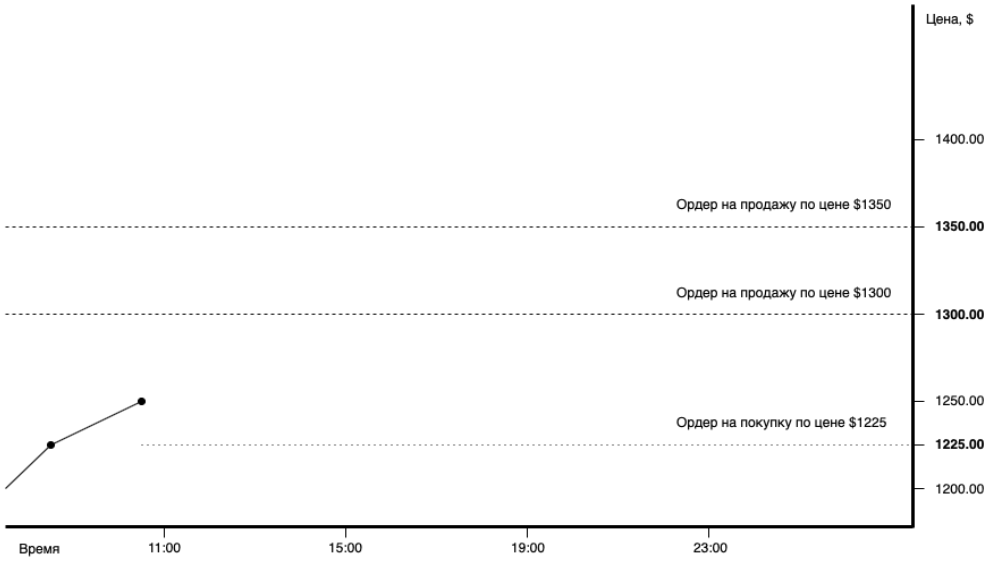

Рисунок 2.4 – Второй этап примера торговли несколькими ордерами

Курс продолжил расти и последовало исполнение следующего ордера продажи по цене $ 1300. Текущий реализующий ордер покупки по цене $ 1225 был отменен и был создан новый, который учитывает объемы уже двух ордеров продажи, соответственно его цена будет выше предыдущего (рисунок 2.5).

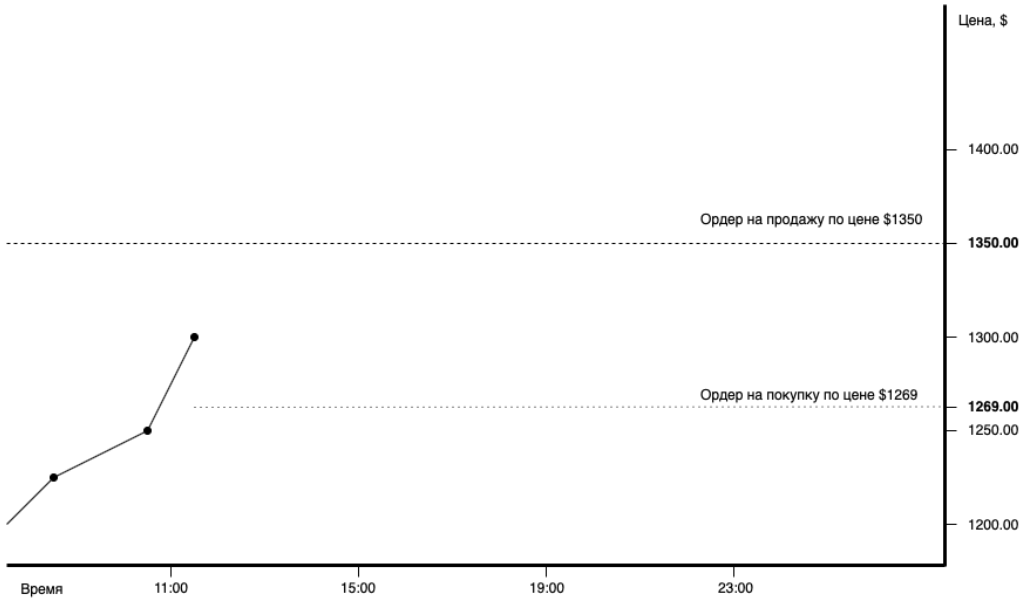

Рисунок 2.5 – Третий этап примера торговли несколькими ордерами

Дальнейшее развитие событий отображено на рисунке 2.6.

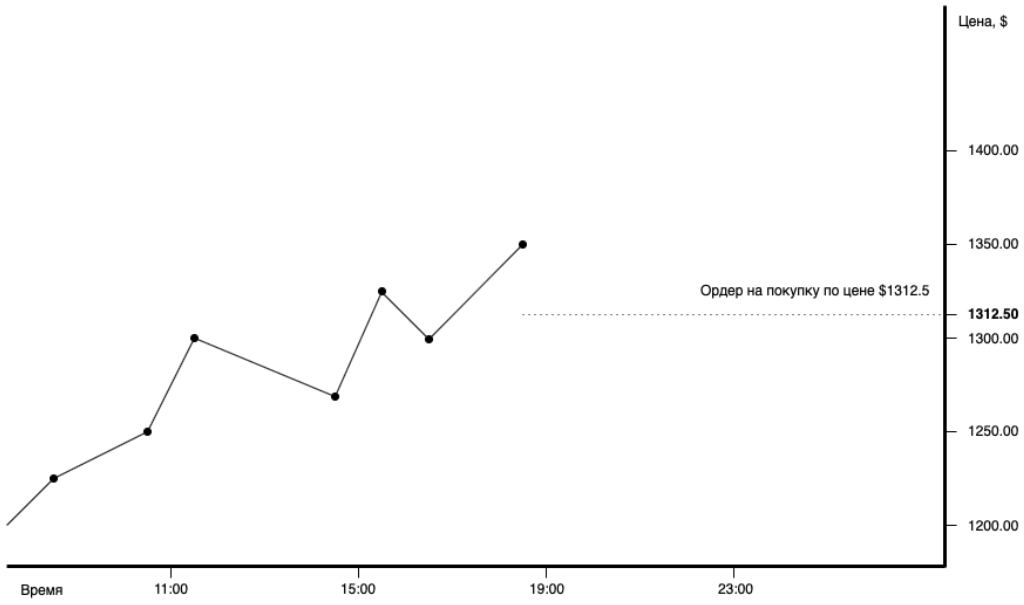

Рисунок 2.6 – Четвертый этап примера торговли несколькими ордерами

При колебаниях курса реализующий ордер покупка не был исполнен, после чего курс достиг цены последнего ордера продажи, в результате чего тот был исполнен. После исполнения последнего ордера продажи, текущий

реализующий ордер покупки по цене $ 1269 был отменен, а вместо него был создан новый реализующий ордер, который учитывает объемы уже всех трех ордеров продажи, соответственно новая цена реализующего ордера покупки будет вновь выше предыдущего.

И наконец, после очередного отклонения курса был исполнен реализующий ордер покупка по цене $ 1312,5. Данные всех исполненных ордеров (трех ордеров продажи и реализующего ордера покупки) изображены на рисунке 2.7.

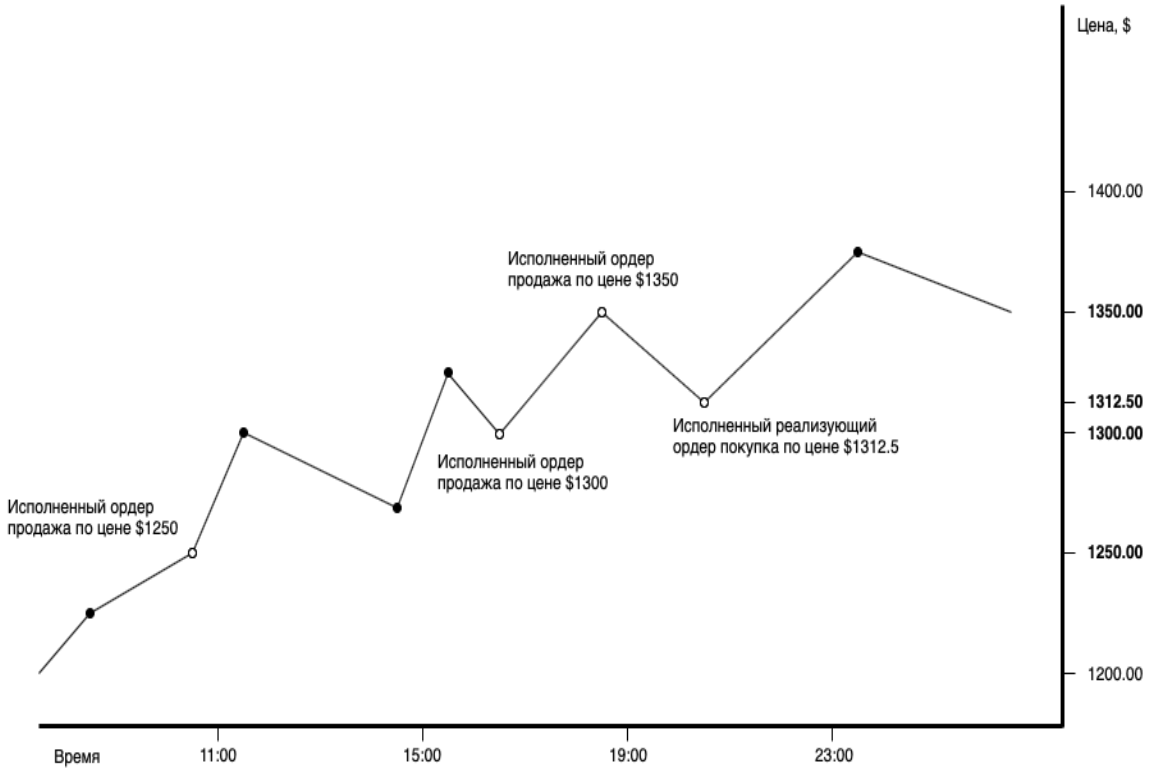

Рисунок 2.7 – Результат примера торговли несколькими ордерами

А теперь рассмотрим, чтобы было, если бы был создан единственный ордер продажа по цене $ 1250. После исполнения данного ордера, был бы создан реализующий ордер покупка, который имел бы цену ниже $ 1250 и не был бы исполнен, судя по графику на рисунке 2.7.

Таким образом, данный пример наглядно демонстрируют эффективность торговли несколькими ордерами по сравнению с торговлей одним ордером.

2.2 Расчёт ордеров

Поскольку торговля одни ордером не эффективна, то мы будем выполнять торговлю несколькими ордерами, которые получаются следующим образом.

1. Выполняется декомпозиция используемого депозита на N частей.
2. Выполняется вычисление N цен в направлении тренда.
3. Выполняется сопоставление частей депозита с ценами.

Выполнение декомпозиции используемого депозита на N частей. Для того, чтобы более сильней стимулировать следование реализующего ордера за текущей ценой на рынке, мы будем выполнять декомпозицию используемого депозита с помощью геометрической прогрессии. Таким образом каждый последующий объем ордера будет больше предыдущего. Для выполнения декомпозиции депозитам с помощью геометрической прогрессии нам потребуются следующие значения:

- объем используемого депозита (сумма геометрической прогрессии);
- количество ордеров (количество элементов геометрической прогрессии);
- параметр, указывающий во сколько раз объем следующего ордера будет больше предыдущего.

Знаменатель прогрессии получим следующим образом:

(1)

где – знаменатель прогрессии;

– значение параметра, указывающего во сколько раз объем следующего ордера будет больше предыдущего, выраженного процентами в долях.

Для того, чтобы получить -ый член геометрической прогрессии, воспользуемся следующей формулой:

(2)

где – n-ый член геометрической прогрессии;

– первый элемент геометрической прогрессии;

– знаменатель геометрической прогрессии;

– количество элементов прогрессии.

Для построения геометрической прогрессии, нам не хватает только первого челна геометрической прогрессии. Воспользуемся формулой суммы первых членов геометрической прогрессии:

(3)

где – сумма первых n членов геометрической прогрессии;

– первый элемент геометрической прогрессии;

– знаменатель прогрессии;

– количество элементов прогрессии.

Из формулы 3 выражаем первый член геометрической прогрессии:

(4)

где – сумма первых n членов геометрической прогрессии;

– первый элемент геометрической прогрессии;

– знаменатель прогрессии;

– количество элементов прогрессии.

Используя все вышеописанные формулы, мы можем выполнить декомпозицию депозита на частей.

Выполнение вычисления цен в направлении тренда. Для вычисления цен мы будем использовать арифметическую прогрессию, за счёт чего мы сможем равномерно покрыть изменение цены. В зависимости от тренда, арифметическая прогрессия будет либо возрастающей, либо убывающей. Для вычисления цен с помощью арифметической прогрессии нам потребуются следующие значения:

- начальная цена, соответствующая цене первого ордера (первый элемент арифметической прогрессии);
- конечная цена, соответствующая цене последнего ордера (последний элемент арифметической прогрессии);

- количество ордеров (есть количество элементов арифметической прогрессии).

Для того, чтобы получить -ый член арифметической прогрессии, воспользуемся следующей формулой:

(5)

где – -ый элемент прогрессии;

– первый элемент прогрессии;

– разность арифметической прогрессии;

– количество элементов прогрессии.

Имея количество элементов прогрессии, первый и последний элемент прогрессии, можно получить разность арифметической прогрессии, которая выражается из вышеописанной формулы и имеет следующий вид:

(6)

где – -ый элемент прогрессии;

– первый элемент прогрессии;

– разность арифметической прогрессии;

– количество элементов прогрессии.

Используя все вышеописанные формулы, мы можем выполнить вычисление цен в направлении тренда.

И наконец, выполняется сопоставление каждой -ой части депозита с вычисленной -ой ценой, в результате чего получается ордеров.

2.3 Первый подтип стратегии

Сначала продать, затем приобрести – подтип стратегии для восходящего тренда. Продажа будет выполняться не одним ордером, а несколькими. Другими словами, если мы хотим продать определенное количество средств, то мы будем разбивать их на несколько

из которых дороже предыдущего. Таким образом, если цена возрастает, то мы продолжаем продать все больше и больше актива. Тем

больше, чем больше возрастает цена. Но в то же время ордер на покупку всегда остается один, не зависимо от того, сколько ордеров на продажу было исполнено. При этом объем ордера покупки и его цена будут изменяться от количества исполненных ордеров продажи. Здесь также стоит учитывать комиссии биржи. При исполнении ордера продажи на получаемый объем (вторая валюта пары) будет наложена комиссия (например, продавая 1 BTC по цене $ 5000, мы получим чуть меньше, чем $ 5000 из-за комиссии биржи). Поскольку при покупке первой валюты пары за вторую, на полученный объем (первая валюта пары) будет накладываться комиссия, то приобретать нужно больше, чем продавали. Таким образом в объеме для ордера покупки будет заложен процент комиссии биржи и необходимый процент выгоды, а цена покупки будет равна отношению затраченных средств к этому объему.

Объем для покупки мы будем рассчитывать следующий образом:

(7)

где – вычисляемый объем для ордера покупки;

– количество проданного актива;

– процент комиссии биржи в долях;

– процент прибыли в долях.

Цену для покупки мы будем рассчитывать следующим образом:

(8)

где – вычисляемая цена для ордера покупки;

– полученные средства при продаже;

– вычисленный объем для ордера покупки.

Нередким случаем является резкий обвал цены, за которым следуют резкое возвращение цены на прежний уровень (рисунок 2.8). Если в момент обвала произвести расчет ордеров, то в дальнейшем велика вероятность того, что этих ордеров не хватит для перекрытия обратного и дальнейшего роста.

Для предотвращения такой ситуации, перед расчетом ордеров мы будем сравнивать последнюю цену на бирже с ценой-ограничителем, которая будет вычисляться следующий образом:

(9)

где – вычисляемая цена-ограничитель;

– наименьшая цена за последние 24 часа;

– требуемый процент отступа от наименьшей цены, выраженный в долях.

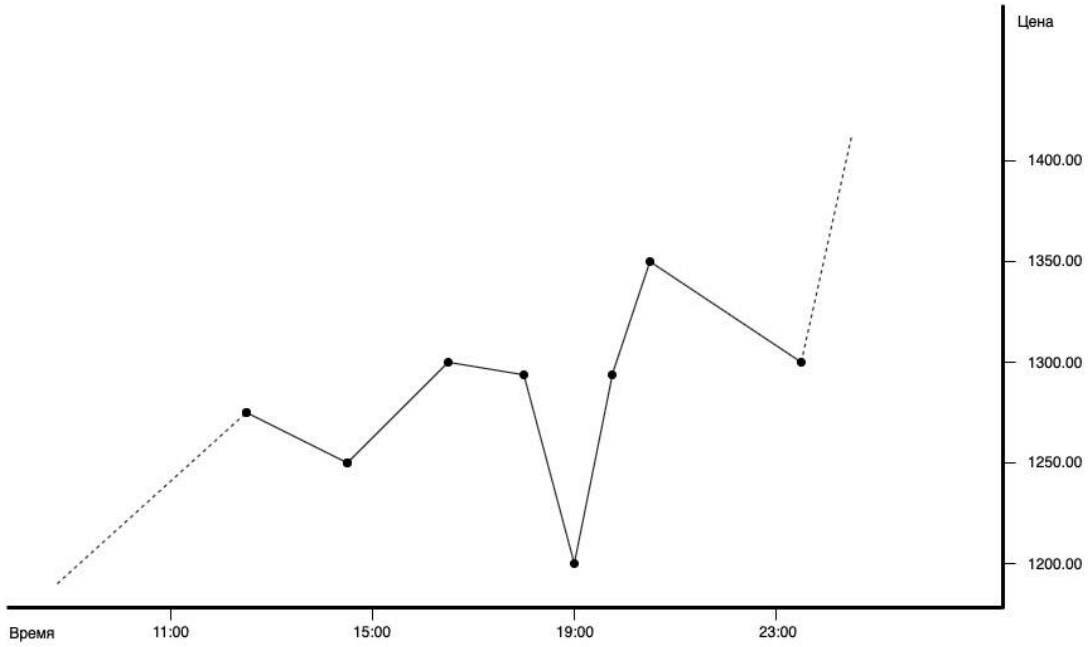

Рисунок 2.8 – Резкий обвал цены с последующим возвращением на прежний уровень

После расчета цены-ограничителя, она сравнивается с последней ценой на бирже. В случае, если последняя цена на бирже меньше цены-ограничителя, то будет происходить ожидание обратного роста цены, в противном случае можно приступить к расчету ордеров.

Для расчёта ордеров потребуется определить цену первого ордера и цену последнего ордера.

Цена первого ордера будет рассчитываться следующим образом:

(10)

где – рассчитываемая цена первого ордера;

– наивысшая цена покупки на бирже;

– процент отступа от последней цены, выраженный в долях.

Используя процент отступа, мы зададим цену первого ордера выше последней цены на бирже.

Цена последнего ордера будет рассчитываться следующим образом:

(11)

где – рассчитываемая цена последнего ордера;

– рассчитанная цена первого ордера;

– процент перекрытия хода цены, выраженный в долях.

Используя цену первого ордера и процент перекрытия хода цены, мы получим цену последнего ордера.

2.4 Второй подтип стратегии

Сначала приобрести, затем продать – подтип стратегии для нисходящего тренда. Закупка будет выполняться не одним ордером, а несколькими. Другими словами, если мы хотим потратить на закупку определенное количество средств, то мы будем разбивать их на несколько более мелких ордеров, каждый из которых дешевле предыдущего. Таким образом, если цена падает, то мы продолжаем закупать все больше и больше актива. Тем больше, чем больше падает цена. Но в то же время ордер на продажу всегда остается один, не зависимо от того, сколько ордеров на покупку было исполнено. При этом объем ордера продажи и его цена будут изменяться от количества исполненных ордеров покупки. Не стоит забывать о комиссиях биржи. При исполнении ордера покупки на получаемый объем (первая валюта пары) будет наложена комиссия (например, приобретая 1 BTC, мы получим чуть меньше из-за комиссии биржи), таким образом объем для ордера продажи будет равен количеству закупленных средств, а цена продажа будет равна цене купленного актива плюс необходимая выгода, также в эту цену закладывается комиссия.

Объем актива для продажи равен приобретённому объему актива с учётом комиссии:

(12)

где – объем актива для продажи;

– количество приобретенного актива с учетом комиссии.

Цену продажи для купленного актива мы будем подсчитывать следующим образом:

(13)

где – вычисляемая цена продажи;

– объем актива для продажи;

– расходы на приобретенный объем;

– процент комиссии биржи, выраженный в долях;

– процент необходимой прибыли, выраженный в долях.

Нередким случаем является резкий рост цены, за которым следуют резкое возвращение цены на прежний уровень (рисунок 2.9). Если на пике такого роста произвести расчет ордеров, то в дальнейшем велика вероятность того, что этих ордеров не хватит для перекрытия обратного и дальнейшего снижения цены.

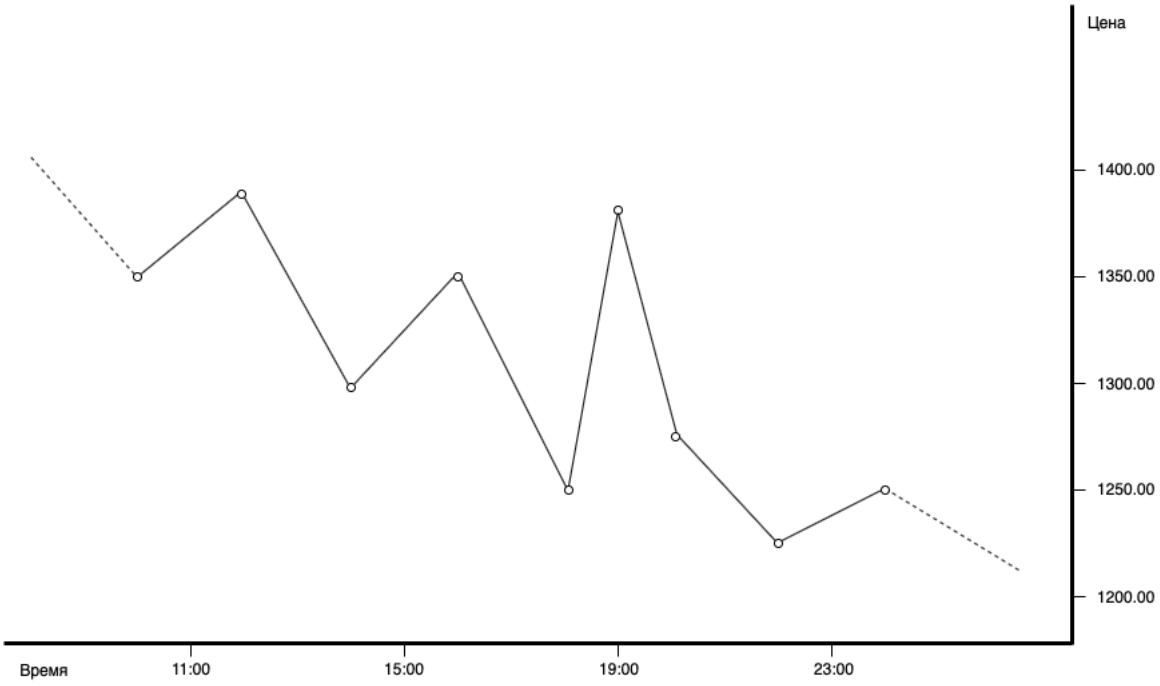

Рисунок 2.9 – Резкий рост цены с последующим возвращением на прежний уровень

Для предотвращения такой ситуации, перед расчетом ордеров мы будем сравнивать последнюю цену на бирже с ценой-ограничителем, которая будет вычисляться следующий образом:

(14)

где – вычисляемая цена-ограничитель;

– наивысшая цена за последние 24 часа;

– требуемый процент отступа от наивысшей цены, выраженный в долях.

После расчета цены-ограничителя, она сравнивается с последней ценой на бирже. В случае, если последняя цена на бирже больше цены-ограничителя, то будет происходить ожидание обратного спада цены, в противном случае можно приступить к расчету ордеров.

Для расчёта ордеров потребуется определить цену первого ордера и цену последнего ордера.

Цена первого ордера будет рассчитываться следующим образом:

(15)

где – рассчитываемая цена первого ордера;

– наименьшая цена продажи на бирже;

– процент отступа от последней цены, выраженный в долях.

Используя процент отступа, мы зададим цену первого ордера ниже последней цены на бирже.

Цена последнего ордера будет рассчитываться следующим образом:

(16)

где – рассчитываемая цена последнего ордера;

– рассчитанная цена первого ордера;

– процент перекрытия хода цены, выраженный в долях.

Используя цену первого ордера и процент перекрытия хода цены, мы получим цену последнего ордера.

2.5 Выводы по разделу

В данном разделе было проведено математическое моделирование торговли. В первую очередь был рассмотрен общий принцип работы стратегии и были выделены два подтипа этой стратегии. Каждый подтип стратегии мы рассмотрели по отдельности и описали все необходимые вычисления для каждого из них.

Также мы рассмотрели условный пример торговли несколькими ордерами, где наглядно убедились в эффективности данного способа торговли по сравнению с торговлей одним ордером. Мы рассмотрели процесс расчёта ордеров и описали все необходимые вычисления, которые используются при их расчёте.

3 РАЗРАБОТКА СИСТЕМЫ ДЛЯ ТОРГОВЛИ НА КРИПТОВАЛЮТНОЙ БИРЖЕ

3.1 Разработка базового класса

Для взаимодействия с той или иной криптовалютной биржей потребуется разработка базового класса, который предоставит набор функций для работы с биржей, которые будут использоваться в стратегиях торговли. Также в базовом классе будут реализованы некоторые дополнительные возможности, не связанные с взаимодействием с криптовалютными биржами.

Для работы с криптовалютными биржами потребуются следующие базовые возможности:

- инициализация работы с указанной криптовалютной биржей;
- установка валютной пары;
- получение баланса по текущей паре;
- получение данных по текущей паре;
- получение открытых ордеров;
- создание ордера на покупку;
- создание ордера на продажу;
- отмена созданного ордера;
- получение истории сделок.

Инициализация работы с указанной криптовалютной биржей и установка валютной пары будут происходить в конструкторе базового класса. В рамках дипломного проекта будет реализована инициализация работы одной криптовалютной биржи – Poloniex. Для того, чтобы достичь неизменяемости кода в случае смены криптовалютной биржи или торговой пары, было принято решение указывать эти параметры в переменных окружения (environment variables). Это некие глобальные значения, расположенные на уровне операционной системы, доступные программам. Кроме названия

криптовалютной биржи и торговой пары могут понадобиться и другие данные, например, секретные ключи для взаимодействия с криптовалютной биржей. Различные криптовалютные биржи могут использовать один или два секретных ключа: публичный и приватный. Таким образом, конструктор базового класса также будет извлекать следующие данные из переменных окружения: название криптовалютной биржи, название валютной пары и публичный и приватные секретные ключи.

Остальные пункты базовых возможностей будут реализованы в методах, которые будут определены в базовом классе. Описание этих методов представлено ниже:

- fetchBalance () – метод, позволяющий получить баланс по текущей торговой паре, включающий в себя общее количество валюты, свободное и используемое количество ордерами;
- fetchTicker () – метод, позволяющий получить информацию по текущей валютной паре;
- fetchOpenOrders () – метод, позволяющий получить список открытых ордеров по текущей валютной паре;
- createBuyOrder (amount: number, price: number) – метод, принимающий два аргумента (объем и цену) и создающий ордер на покупку, который приобретет первую валюту пары, использовав для этого вторую валюту пары;
- createSellOrder (amount: number, price: number) – метод, принимающий два аргумента (объем и цену) и создающий ордер на продажу, который приобретёт вторую валюту пары, использовав для этого первую валюту пары;
- cancelOrder (id: number) – метод, принимающий один аргумент (уникальный идентификатор ордера) и позволяющий отменить созданный ордер на покупку или на продажу;

- fetchTradesHistory (since: number = 0) – метод, принимающий один опциональный аргумент (метка времени, от начала которой требуются данные) и позволяющий получить историю сделок по текущей валютной паре.

Стоит обратить внимание на то, что каждая криптовалютная биржа имеет торговую комиссию, ограничения при создании ордера, а также точность при округлении, причем эти значения могут изменяться в зависимости от той или иной биржи. Что касается торговой комиссии, то некоторые криптовалютные биржи используют прогрессивную шкалу, что делает данный параметр динамическим, а значит он не может быть определен в базовом классе.

Таким образом, было решено создать ещё три метода:

- loadFees() – абстрактный метод, устанавливающий торговую комиссию, который переопределяется при реализации той или иной стратегии;
- loadLimits() – метод, устанавливающий значения ограничений;
- loadPrecisions() – метод, устанавливающий значения точности округлений.

На рисунке 3.1 представлены интерфейсы, которые будут использоваться этими методами и базовым классом в целом.

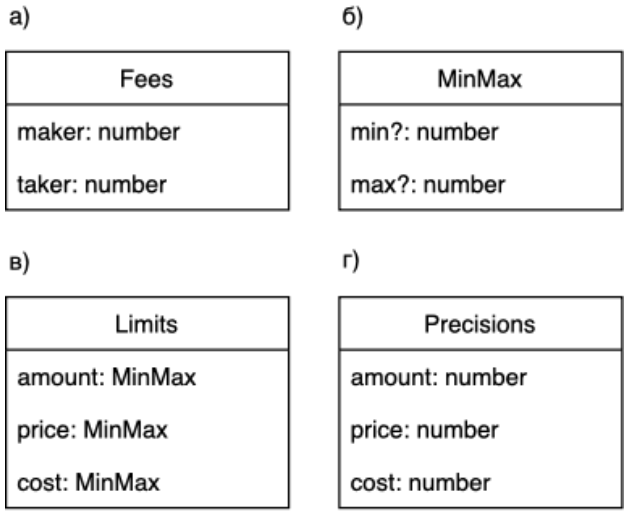

Рисунок 3.1 – Интерфейсы базового класса: а) интерфейс Fess; б) интерфейс MinMax; в) интерфейс Limits; г) интерфейс Precisions

Интерфейс Fees. Некоторые криптовалютные биржи для торговой комиссии используют модель maker-taker. Что обозначает maker-taker? Каждая сделка происходит между двумя сторонами: производителем (maker), чей заказ существует в книге заказов до сделки, и потребителем, который размещает заказ, который соответствует заказу производителя (taker). Производители названы так потому, что их заказы обеспечивают ликвидность на рынке. Потребители – те, кто удаляет эту ликвидность, сопоставляя заказы производителей с их собственными. Таким образом получаются две шкалы торговой комиссии, для производителей (maker) и для потребителей (taker), при этом комиссия для потребителей (taker) выше, чем для производителей (maker), поскольку последние обеспечивают ликвидность на рынке. Модель maker-taker стимулирует рыночную ликвидность, вознаграждая создателей этой ликвидности сниженной комиссией. Таким образом, интерфейс Fees будет иметь два поля: maker и taker. В случае, если работа системы будет происходить на криптовалютной бирже, не использующей модель maker-taker, то оба параметра будут установлены одинаковыми, а использовать можно будет любой из них.

Интерфейс Limits. Содержит минимальное и максимальное значения для объема, цены и стоимости.

Интерфейс MinMax. Данный интерфейс используется в описании интерфейса Limits. Имеет два опциональных поля: min и max. Используется для хранения минимального и максимального значений.

Интерфейс Precisions. Содержит точность округления для объема, цены и стоимости.

Помимо реализованных нами интерфейсов, мы будем использовать интерфейсы, определенные в библиотеке ccxt [16], а именно: Balance – интерфейс баланса учетной записи, Exchange – интерфейс используемой криптовалютной биржи, Ticker – интерфейс данных используемой валютной пары, Trade – интерфейс сделок на криптовалютной бирже и Order – интерфейс ордера.

Ниже представлены поля, которые будут определены в базовом классе:

- exchange – экземпляр ccxt;
- name – название выбранной криптовалютной биржи;
- symbol – выбранная криптовалютная биржа;
- fees – задаваемые комиссии биржи;
- limits – задаваемые ограничения биржи;
- precisions – задаваемые точности округлений биржи.

Также мы определим два геттера: firstCurrency и secondCurrency, возвращающие название первой и второй валюты пары соответственно.

Ещё один вспомогательный метод, который будет определен в базовом классе, это метод toPrecision(type: 'amount', 'price', 'cost', value: number), который будет округлять переданное значение по типу (объем, цена или стоимость), используя точности, указанные в объекте precisions.

Последний метод, который будет реализован в базовом классе, это метод init(), который при вызове будет загружать данные маркетов биржи, которые, например, могут пригодиться для проверки наличия криптовалютной пары на бирже, затем в этом методе будет производиться установка параметров комиссии, ограничений и точности округлений с помощью методов, которые мы описали выше (один из которых абстрактный и должен быть переопределен при реализации стратегии), и в конце выполняет подключение к базе данных.

В совокупности при разработке у нас получился базовый класс Base, который изображен на рисунке 3.2.

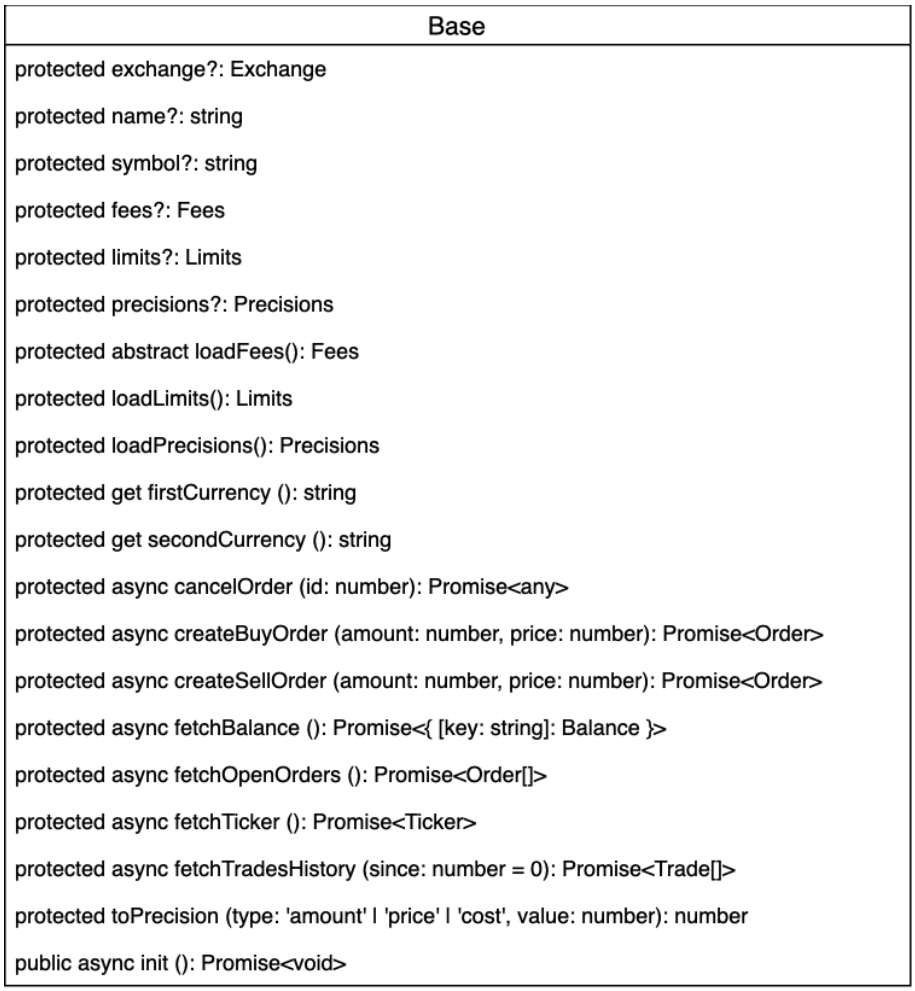

Рисунок 3.2 – Базовый класс Base

В этом классе были реализованы базовые возможности, описанные выше. В дальнейшем, при разработке той или иной стратегии торговли, класс реализовываемой стратегии будет наследовать класс Base, что позволит ему использовать все реализованные базовые возможности. Таким образом, при разработке той или иной стратегии торговли не потребуется каждый раз реализовывать функционал взаимодействия с криптовалютной биржей и иные вспомогательные функции.

3.2 Разработка стратегий торговли

В определенное время курс определенной пары может иметь восходящий или нисходящей тренд. Таким образом можно реализовать две стратегии с перекрытием хода цены в разные стороны.

Стратегия для восходящего тренда. Суть стратегии, простыми словами, заключается в том, чтобы сначала купить, а затем продать подороже. На примере пары BTC/USDT. Сначала мы будем приобретать, а затем продавать. Поскольку создание ордера на покупку по паре означает купить первую валюту пары за вторую, то для начала работы по этой стратегии нам понадобится вторая валюта пары (USDT в данном примере). Используя данную стратегию, мы будем приобретать BTC за USDT, а затем продадим купленное количество BTC по более высокой цене, таким образом наращивая вторую валюту пары.

Стратегия для нисходящего тренда. Суть стратегии, простыми словами, заключается в том, чтобы сначала продать, а затем купить подешевле. На примере пары BTC/USDT. Сначала мы будем продавать, а затем приобретать. Поскольку создание ордера на продажу по паре означает продать первую валюту пары за вторую, то для начала работы по этой стратегии нам понадобится первая валюта пары (BTC в данном примере). Используя данную стратегию, мы будем продавать BTC, получая за это USDT, а затем приобретём BTC на полученное количество USDT, но уже по более низкой цене, таким образом будем наращивать первую валюту пары.

Как видно из краткого описания этих двух стратегий, они имеют одну и ту же суть, поэтому мы реализуем их как подтипы одной стратегии по умолчанию, которую назовём «Default», а подтипы назовем «buy-then-sell» и «sell-then-buy» соответственно.

3.2.1 Стратегия по умолчанию

Поскольку описанные выше стратегии имеют одну и ту же суть и отличаются только в логике выполнения, то было решено реализовать для них один общий класс, который они будут наследовать, а уже сама логика каждой стратегии будет реализована отдельно.

В обоих подтипах нам нужно будет рассчитывать ордера. Для их расчета нам потребуются следующие параметры:

- объем депозита – процент свободного депозита, которым мы будем оперировать при расчете;
- стоп-рейт – требуемый процент отступа текущей цены от наибольшей/наименьшей (в зависимости от подтипа стратегии) цены за последние 24 часа, который должен быть соблюден для начала работы;
- отступ первого ордера – на сколько процентов цена первого ордера будет отличаться от текущей цены на бирже;
- количество ордеров – на сколько ордеров будет разбит используемый депозит;
- перекрытие хода цены – на сколько процентов будет покрывать таблица изменение цены относительно текущей;
- увеличение объема – на сколько процентов объем следующего ордера будет больше предыдущего;
- прибыль – процент чистой прибыли.

Для того, чтобы сохранить неизменяемость кода в случае, если нам потребуется изменить тот или иной параметр, данные параметры так же будут указываться переменных окружения (environment variables). Проценты для параметров будут задаваться строкой, например, «50 %», а затем будут представлены как проценты в долях.

На основе вышеописанного, был спроектирован интерфейс Parameters, который изображен на рисунке 3.3 и будет использоваться в классе Default.

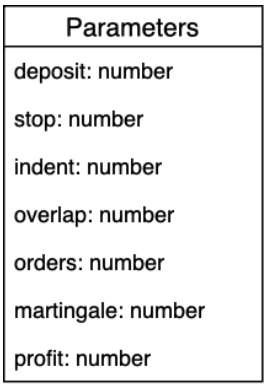

Рисунок 3.3 – Интерфейс Parameters

Одно из требований к системе – возможность возобновлять работу после остановки или неожиданного завершения работы, таким образом нам потребуется сохранять текущее состояние в базу данных. Для достижения этой цели, нам потребуется сохранять в базу данных следующие данные:

- название выбранной криптовалютной биржи;
- подтип стратегии;
- выбранная валютная пара;
- данные исполнения ордеров – это исполненный объем и затраты на исполненный объем;
- данные цен: цена, относительно которой рассчитывалась таблица ордеров, первая и последние цены ордеров в таблице;
- данные созданных ордеров покупки и продажи;
- временная метка последнего исполнения ордера.

Кроме этого, нам понадобятся два флага, которые также будут добавлены в состояние. Первый флаг – это флаг, указывающий на то, что требуется рассчитать таблицу ордеров и произвести создание ордеров. Второй флаг – это флаг, указывающий на необходимость сбросить текущее состояние. Это нужно, например, если понадобится начать работу с нуля.

Для того, чтобы можно было понять, когда создано состояние и когда оно обновилось, мы будем сохранять две даты: дату создания и дату обновления состояния.

Определим данные ордеров, которые нам потребуется сохранять в состоянии. Стоит учесть то, что ордер может быть исполнен частично. Например, если мы создали ордер на продажу 1 BTC по цене 5000 USDT, а кто-то создал ордер на покупку 0,3 BTC по такой же цене, то наш ордер исполнится частично, а именно 0,3 BTC будут проданы, а оставшиеся 0,7 BTC останутся в заказе на продажу.

Таким образом, следующие данные ордеров нам потребуется сохранять в состоянии:

- уникальный идентификатор ордера;
- объем ордера;
- цена ордера;
- исполненный объем ордера;
- оставшийся объем для исполнения;
- временная метка создания ордера.

Получился следующий интерфейс данных ордеров, изображенный на рисунке 3.4, который мы будем использовать в состоянии.

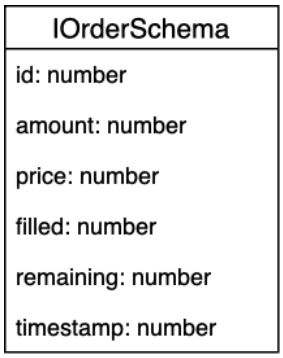

Рисунок 3.4 – Интерфейс IOrderSchema

Остальные интерфейсы, которые мы будем использовать в состоянии представлены на рисунке 3.5.

Интерфейс Requirements имеет два поля: grid и reset, указывающие на необходимость в расчёте таблицы ордеров и на необходимость сбросить текущее состояние соответственно.

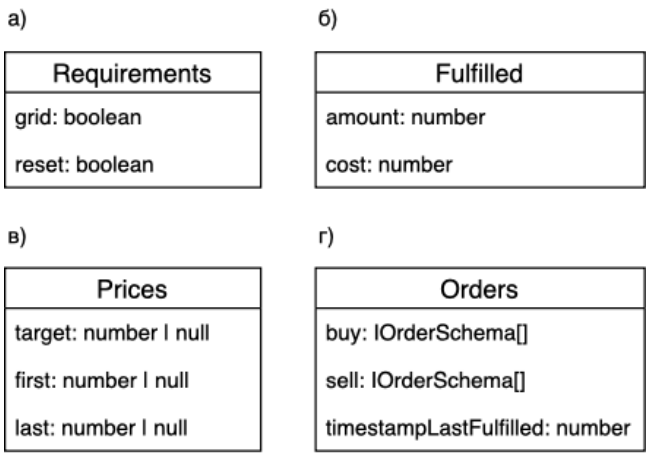

Рисунок 3.5 – Интерфейсы класса Default: а) интерфейс Requirements б) интерфейс Fulfilled в) интерфейс Prices г) интерфейс Orders

Интерфейс Fulfilled имеет два поля: amount и cost, отражают исполненный объем ордеров и затраты на исполненный объем соответственно.

Интерфейс Prices имеет три поля: target, first и last, отражают цену, относительно которой рассчитывалась таблица ордеров, цену первого ордера таблицы ордеров и цену последнего ордера таблицы ордеров соответственно. Поскольку могут быть ситуации, в которых таблица ордеров не рассчитывается, например, не выполняется требование отступа (параметр indent) текущей цены от минимальной/максимальной цены за последние 24 часа и ожидается рост/падение цены, то поля буду заполнены как null.

Интерфейс Orders имеет три поля: buy, sell и timestampLastFulfilled, первые два отражают данные ордеров покупки и продажи соответственно, а последнее поле отражает временную метку последнего исполненного ордера (в том числе и частично исполненного).

Определив вспомогательные интерфейсы, теперь можно определить интерфейс состояния в целом. Интерфейс состояния изображен на рисунке 3.6.

Для того, чтобы спроектировать класс Default, необходимо понимать общий принцип работы обоих подтипов стратегии.

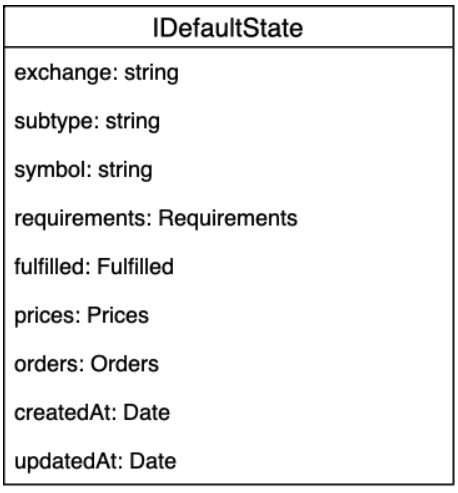

Рисунок 3.6 – Интерфейс IDefaultState

В первую очередь нам понадобятся два метода:

- метод, позволяющий создать новое состояние;
- метод, позволяющий получить существующее состояние.

Далее нам необходимо будет рассчитать ордера, а значит для этого нам понадобятся следующие методы:

- метод, загружающий свежую сводную информацию по валютной паре;
- метод, возвращающий массив арифметического распределения от начальной цены к конечной;
- метод, возвращающий массив геометрического распределения используемого депозита;
- метод, возвращающий массив ордеров, используя массив арифметического распределение цен, массив геометрического распределения используемого депозита и предполагаемый тип таблицы ордеров (таблица ордеров покупки или продажи), данный метод будет возвращать массив данных IGridOrderData, интерфейс которого изображён на рисунке 3.7.

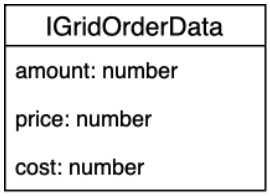

Рисунок 3.7 – Интерфейс IGridOrderData

После расчёта ордеров и их выставления, необходимо будет следить за развитием событий.

В первую очередь нам понадобится определить, не было ли совершено исполнение ордеров. Для этого нам понадобится метод, который будет получать историю сделок с момента последнего исполнения ордера, исключая его. В случае, если мы только создали ордера, то данный метод должен возвращать историю сделок с момента создания состояния, поскольку история сделок, которая происходила до создания состояния нас абсолютно не интересует.

В случае, если мы определили, что не было совершено исполнение ордеров, то нам понадобится два следующих метода в зависимости от подтипа стратегии:

- метод, проверяющий не ушла ли цена вверх относительно цены, от которой строилась таблица ордеров;
- метод, проверяющий не ушла ли цена вниз относительно цены, от которой строилась таблица ордеров.

Будем считать, что курс ушел выше/ниже, если цена изменилась на удвоенный процент отступа (параметр indent) от цены, относительно которой рассчитывалась таблица ордеров.

Если мы определили, что цена ушла вверх или вниз, мы отменяем созданные ордера, вновь рассчитываем ордера и создаем их, таким образом подтягиваем ордера за текущей ценой и вновь начинаем следить за развитием событий. В противном случае, если цена не ушла вверх или вниз, мы продолжаем следить за развитием событий.

В случае, если мы определили, что было совершено исполнение ордеров, нам понадобится:

- геттер, возвращающий список идентификаторов ордеров, сохраненных в нашем состоянии;
- метод, получающий список открытых ордеров, причем этот метод должен возвращать список только тех открытых ордеров, которые были созданы нашей стратегией, а точнее её подтипом, для этого и понадобится вышеописанный геттер;
- геттер, возвращающий список идентификаторов открытых ордеров, полученных вышеописанным методом.

Исключая идентификаторы списка открытых ордеров из списка идентификаторов, сохраненных в состоянии, мы получаем идентификаторы исполненных ордеров в текущей проверке.

Поскольку класс Default будет наследовать базовый класс Base, в котором определен абстрактный метод loadFees() (см. рисунок 3.2), то потребуется переопределить этот метод в классе Default.

И наконец, в классе Default будут определены следующие поля:

- набор параметров, описанных интерфейсом Parameters;

- состояние стратегии, описанное интерфейсом IDefaultState;
- данные по валютной паре;
- массив истории сделок;
- массив открытых ордеров.

Исходя из всего вышеописанного, был спроектирован класс Default, который изображен на рисунке 3.8.

Конструктор класса Default будет получать необходимые параметры из переменных окружения (environment variables), проверит наличие каждого параметра, преобразует их к нужному формату (например, указанное количество процентов нужно преобразовать в проценты в долях), затем сохранит их в поле parameters, а так же задаст начальные значения для полей state, ticker, trades и orders.

Метод distributePrices (from: number, to: number) – метод, который возвращает арифметическое распределение от начальной цены к конечный, соответственно, принимает два аргумента: начальную и конечную цены. Количество элементов прогрессии – есть количество создаваемых ордеров.

Метод decomposeDeposit (deposit: number) – метод, который возвращает геометрическое распределение используемого количества депозита, соответственно, принимает один аргумент: количество используемого депозита. Количество элементов прогрессии – есть количество создаваемых ордеров.

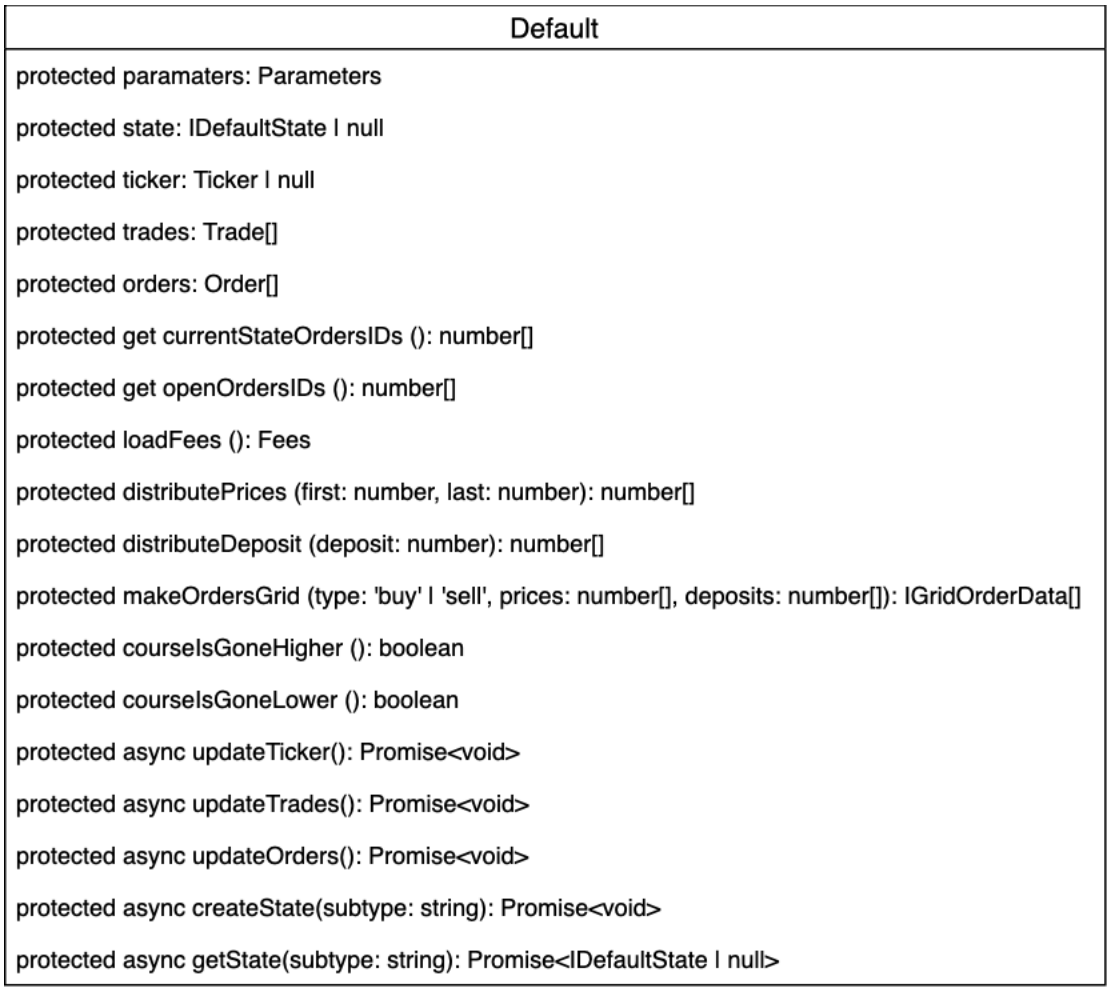

Рисунок 3.8 – Класс Default

Метод makeOrdersGrid (type: 'buy' | 'sell', prices: number[], deposits: number[]) – метод, который возвращает массив ордеров для покупки или для продажи, в зависимости от выбранного типа, используя арифметическую прогрессию распределения цен и геометрическую прогрессию распределения используемого депозита, соответственно, принимает три аргумента: тип построения таблицы ордеров, арифметическую прогрессию распределения цен и геометрическую прогрессию распределения используемого депозита.

Рассмотрим, как данный метод сопоставит арифметическое и геометрическое распределения в зависимости от выбранного типа построения таблицы.

Сопоставление для ордеров закупки.

Значения amount будет обозначать количество покупаемой валюты (1-ая валюта пары). Очередное -ое значение amount есть отношение -го

элемент геометрического распределения депозита к -му элементу арифметического распределения.

Значение price будет обозначать цену, по которой будет производиться покупка (будет выражаться во 2-ой валюте пары). Очередное -ое значение price будет равно -му элементу арифметического распределения цен.

Значение cost будет обозначать затраченное количество (2-ая валюта пары) на покупку. Очередное -ое значение cost будет равно -му элементу геометрического распределения используемого депозита.

Сопоставление для ордеров продажи.

Значения amount будет обозначать количество продаваемой валюты (2-ая валюта пары). Каждое -ое значение amount будет равно -му элементу геометрического распределения используемого депозита.

Значение price будет обозначать цену, по которой будет производиться продажа (будет выражаться в 1-ой валюте пары). Очередное -ое значение price будет равно -му элементу арифметического распределения цен.

Значение cost будет обозначать полученное количество (1-ая валюта пары) от продажи. Очередное -ое значение cost есть произведение -го элемент геометрического распределения депозита и -го элемента арифметического распределения.

3.2.2 Первый подтип стратегии по умолчанию

Работа первого подтипа стратегии основана на принципе покупай частями при падении цены (начальная часть цикла) и продай дороже все купленное при росте цены (финальная часть цикла). Идея заключается в том, что имея определенный депозит, будет производится покупка не на всю сумму сразу, а частями при падении цены, используя для этого рассчитанную заранее сетку ордеров. Первый ордер на покупку в сетке самый близкий к текущей цене и самый маленький по объему. Каждый последующий ордер на покупку объемней и дешевле. Таким образом, при падении цены мы будем покупать все больше и больше, но за меньшую цену. Это позволяет

совершить итоговую прибыльную сделку, продав все купленное по цене гораздо ниже, чем мы начинали покупать. После выставления сетки ордеров на бирже, будет выполняться отслеживание их выполнения. При падении цены выполнится первый ордер в сетке на покупку, который был ближе всего к текущей цене. После этого, будет на биржу будет выставлен фиксирующий прибыль ордер на продажу (объем ордера будет такой же, как у только что выполненного первого ордера в сетке на покупку, а в цене будет заложен профит). При дальнейшем падении цены выполнится второй ордер в сетке на покупку. Текущий фиксирующий прибыль ордер на продажу будет отменен и будет выставлен новый, который будет нести в себе объем двух выполненных ордеров на покупку, цена его будет ниже, чем цена только что отмененного ордера на продажу, но в ней также будет заложена прибыль. Таким образом алгоритм будет продолжаться, пока цена не вырастет и не выполнится фиксирующий прибыль ордер на продажу. Отметим, что по мере падения цены и выполнения первичных ордеров на покупку, цена фиксирующего прибыль ордера на продажу будет тоже снижаться, что и позволит в итоге продать все купленное дешевле, чем начинали покупать. Также отметим, что фиксирующий прибыль ордер всегда один, он в себе несет объем всех выполненных первичных ордеров, а цена его составляет цену купленного плюс профит. При выполнении фиксирующего прибыль ордера на продажу заканчиваем цикл и начинаем новый цикл работы.

Работа данного подтипа стратегии была реализована с помощью отдельного класса, который наследует класс Default (см. рисунок 3.8), включающий в себя четыре метода:

- метод start(), начинающий работу стратегии;
- метод direction(), определяющий по текущему состоянию необходимые дальнейшие действия;
- метод observe() – основной метод, внутри которого происходит наблюдение за ордерами, их создание и так далее;

- метод calcSellValues(), необходимый для вычисление объема и цены реализующего ордера продажи (интерфейс данных для этого метода изображен на рисунке 3.9).

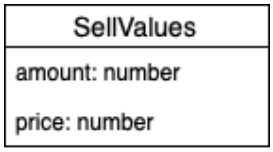

Рисунок 3.9 – Интерфейс SellValues

Для создания и сохранения состояния в базу данных, используя метод createState() класса Default, необходимо определить поле с подтипом стратегии, что выполняется в конструкторе класса. Класс первого подтипа стратегии изображен на рисунке 3.10.

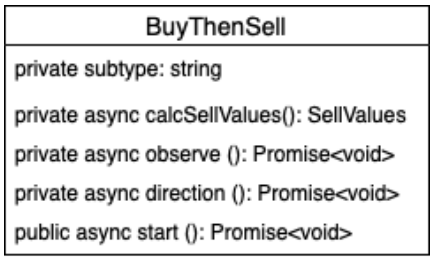

Рисунок 3.10 – Класс BuyThenSell

Алгоритм работы метода start() представлен в приложении 1. После выполнения метода start() происходит вызов метода direction(), алгоритм работы которого представлен в приложении 2. И наконец, происходит вызов метода observe(), алгоритм работы которого представлен в приложении 3, после чего вновь происходит вызов метода direction(), и так до тех пор, пока программа не будет остановлена.

Таким образом получается, что алгоритм работы зациклен. После начала работы программы, происходит вызов метода direction(), после которого происходит вызов метода observe(). После ожидания определенного интервала времени, вновь происходит поочередный вызов этих двух методов.

3.2.3 Второй подтип стратегии по умолчанию

Принцип работы зеркален первому подтипу и основан на принципе продавай частями при росте цены (начальная часть цикла) и купи дешевле

все проданное при падении цены (финальная часть цикла). Первый ордер на продажу в сетке самый близкий к текущей цене и самый маленький по объему. Каждый последующий ордер объемней и дороже. Таким образом, при росте цены мы будем продавать все больше и больше, но за большую цену. Это позволяет совершить итоговую прибыльную сделку, купив все проданное по цене гораздо выше, чем мы начинали продавать После выставления сетки ордеров на бирже, будет выполняться отслеживание их выполнения и, в случае выполнения, будет выставлен ордер на покупку по цене ниже, чем они были проданы.

Работа данного подтипа стратегии была реализована с помощью отдельного класса, который наследует класс Default (см. рисунок 3.8), включающий в себя четыре метода:

- метод start(), начинающий работу стратегии;
- метод direction(), определяющий по текущему состоянию необходимые дальнейшие действия;
- метод observe() – основной метод, внутри которого происходит наблюдение за ордерами, их создание и так далее;
- метод calcBuyValues(), необходимый для вычисление объема и цены реализующего ордера покупки (интерфейс данных для этого метода изображен на рисунке 3.11).

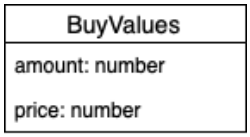

Рисунок 3.11 – интерфейс BuyValues

Для создания и сохранения состояния в базу данных, используя метод createState() класса Default, необходимо определить поле с подтипом стратегии, что выполняется в конструкторе класса. Класс второго подтипа стратегии изображен на рисунке 3.12.

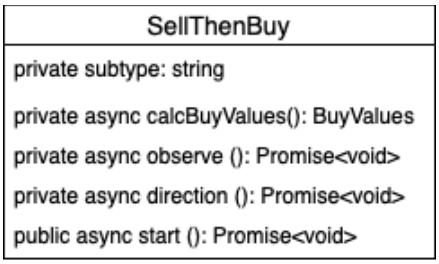

Рисунок 3.12 – Класс SellThenBuy

Алгоритм работы метода start() идентичен алгоритму из предыдущего подтипа стратегии (см. приложение 1). После выполнения метода start() происходит вызов метода direction(), алгоритм работы которого представлен в приложении 4. И наконец, происходит вызов метода observe(), алгоритм работы которого представлен в приложении 5, после чего вновь происходит вызов метода direction(), и так до тех пор, пока программа не будет остановлена.

Таким образом здесь тоже получается, что алгоритм работы зациклен. После начала работы программы, происходит вызов метода direction(), после которого происходит вызов метода observe(). После ожидания определенного интервала времени, вновь происходит поочередный вызов этих двух методов.

3.3 Тестирование работы на экспериментальных данных

Выполним тестирование для первого подтипа стратегии (сначала купить, затем продать) на экспериментальных данных со следующими параметрами:

- валютная пара – BTC/USDT;
- количество доступного депозита – 1000 USDT;
- «текущая» наименьшая цена продажи – 7000,00000001 USDT;
- процент отступа от наименьшей цены продажи – 0,25%;
- количество создаваемых ордеров – 10;
- процент перекрытия хода цены – 5%;
- процент увеличения объема каждого последующего ордера – 9,5%;
- процент чистой прибыли – 1%;

- процент комиссии биржи – 0.15%;

Рассчитанные ордера по указанным параметрам приведены в таблице 1.

Таблица 1 – Рассчитанные ордера покупки

№	Приобретаемое кол-во, BTC	Цена, USDT	Стоимость, USDT
1.	0,00920389	6982,50000001	64,26616193
2.	0,01013456	6943,70833334	70,37142873
3.	0,01115969	6904,91666668	77,05672948
4.	0,01228890	6866,12500001	84,37712351
5.	0,01353280	6827,33333334	92,39293653
6.	0,01490309	6788,54166668	101,17024743
7.	0,01641267	6749,75000001	110,78141933
8.	0,01807576	6710,95833334	121,30567220
9.	0,01990803	6672,16666668	132,82969417
10.	0,02192678	6633,37500001	145,44855428

Опираясь на данные таблицы 1, можно сделать следующие выводы по поводу расчета ордеров покупки:

- правильно рассчитывается цена первого ордера, при расчете которой используется «текущая» наименьшая цена продажи и процент отступа от наименьшей цены продажи;
- правильно рассчитывается цена последнего ордера, при расчете которой используется цена первого ордера и процент перекрытия хода цены;
- правильно рассчитываются цены остальных ордеров, при расчете которых используется арифметическая прогрессия;
- правильно рассчитываются объемы (стоимость) для всех ордеров, при расчете которых используется геометрическая прогрессия: объем каждого последующего ордера больше предыдущего на 9,5%, а суммарный объем затрат равен количеству доступного депозита.

Таким образом, можно сделать вывод о том, что расчет ордеров покупки для первого подтипа стратегии (сначала купить, затем продать) выполняется правильно.

Выполним проверку расчета реализующего ордера продажи для первого подтипа стратегии. Для проверки реализующего ордера будем считать, что исполнились все десять рассчитанных ордеров покупки. Рассчитанный реализующий ордер продажа отображен в таблице 2.

Таблица 2 – Рассчитанный реализующий ордер продажа

Продаваемое кол-во, BTC	Цена, USDT	Стоимость, USDT
0,14732486	6865,89673945	1011,51727591

После исполнения реализующего ордера продажи, будет продано 0,14732486 BTC по цене 6865,89673945 USDT, с учётом комиссии будет получено 1010 USDT. Разница между полученным количеством (1010 USDT) и потраченным количеством (1000 USDT) является чистой прибылью и соответствует 1% от использованного количества.

Таким образом, можно сделать вывод о том, что расчёт реализующего ордера продажи для первого подтипа стратегии выполняется правильно.

На рисунке 3.13 представлены результаты эмуляции работы первого подтипа стратегии по выше представленным параметрам.

Теперь выполним тестирование для второго подтипа стратегии (сначала продать, затем купить) на экспериментальных данных со следующими параметрами:

- валютная пара – BTC/USDT;
- количество доступного депозита – 1 BTC;
- «текущая» наибольшая цена покупки – 6999,99999999 USDT;
- процент отступа от наименьшей цены продажи – 0,25%;
- количество создаваемых ордеров – 10;
- процент перекрытия хода цены – 5%;
- процент увеличения объема каждого последующего ордера – 9,5%;

- процент чистой прибыли – 1%;
- процент комиссии биржи – 0,15%.

```
> cryptocurrency-exchange-trader@1.2.0 emulate:buy-then-sell /Users/kenikh/Projects/cryptocurrency-exchange-trader
> node ./dist/emulation/buy-then-sell.js

Наименьшая цена продажи на бирже: 7000.00000001.
1. Исполнен ордер покупка 0.00920389 BTC x 6982.50000001 USDT (потрачено 64.26616193 USDT).
2. Исполнен ордер покупка 0.01013456 BTC x 6943.70833334 USDT (потрачено 70.37142873 USDT).
3. Исполнен ордер покупка 0.01115969 BTC x 6904.91666668 USDT (потрачено 77.05672948 USDT).
4. Исполнен ордер покупка 0.0122889 BTC x 6866.12500001 USDT (потрачено 84.37712351 USDT).
5. Исполнен ордер покупка 0.0135328 BTC x 6827.33333334 USDT (потрачено 92.39293653 USDT).
6. Исполнен ордер покупка 0.01490309 BTC x 6788.54166668 USDT (потрачено 101.17024743 USDT).
7. Исполнен ордер покупка 0.01641267 BTC x 6749.75000001 USDT (потрачено 110.78141933 USDT).
8. Исполнен ордер покупка 0.01807576 BTC x 6710.95833334 USDT (потрачено 121.3056722 USDT).
9. Исполнен ордер покупка 0.01990803 BTC x 6672.16666668 USDT (потрачено 132.82969417 USDT).
10. Исполнен ордер покупка 0.02192678 BTC x 6633.37500001 USDT (потрачено 145.44855428 USDT).
Всего куплено 0.14732486 BTC (с учётом комиссии).
Всего потрачено 1000 USDT.
Исполнен ордер продажа 0.14732486 BTC x 6865.89673945 USDT (получено 1011.51727591) USDT.
Продано 0.14732486 BTC.
Получено: 1010 USDT (с учётом комиссии).
BTC: 0 -> 0.14732486 -> 0 (ожидалось: 0).
USDT: 1000 -> 0 -> 1010 (ожидалось: ~ 1010).
```

Рисунок 3.13 – эмуляция работы первого подтипа стратегии

Рассчитанные ордера по указанным параметрам приведены в таблице 3.

Таблица 3 – Рассчитанные ордера продажи

№	Продаваемое кол-во, BTC	Цена, USDT	Стоимость, USDT
1.	0,06426615	7017,49999999	450,98770762
2.	0,07037143	7056,48611110	496,57501841
3.	0,07705672	7095,47222221	546,75381629
4.	0,08437711	7134,45833332	601,98497558
5.	0,09239293	7173,44444443	662,77555041
6.	0,10117026	7212,43055555	729,68347454
7.	0,11078144	7251,41666666	803,32238037
8.	0,12130567	7290,40277777	884,36719353
9.	0,13282971	7329,38888888	973,56060059
10.	0,14544858	7368,37499999	1071,71968066

Опираясь на данные таблицы 3, можно сделать следующие выводы по поводу расчета ордеров продажи:

- правильно рассчитывается цена первого ордера, при расчете которой используется «текущая» наибольшая цена покупки и процент отступа от наименьшей цены продажи;

- правильно рассчитывается цена последнего ордера, при расчете которой используется цена первого ордера и процент перекрытия хода цены;
- правильно рассчитываются цены остальных ордеров, при расчете которых используется арифметическая прогрессия;
- правильно рассчитываются объемы (продаваемое количество) для всех ордеров, при расчете которых используется геометрическая прогрессия: объем каждого последующего ордера больше предыдущего на 9,5%, а суммарный объем затрат равен количеству доступного депозита.

Таким образом, можно сделать вывод о том, что расчет ордеров продажи для второго подтипа стратегии (сначала продать, затем купить) выполняется правильно.

Выполним проверку расчета реализующего ордера покупки для второго подтипа стратегии. Для проверки реализующего ордера будем считать, что исполнились все десять рассчитанных ордеров продажи. Рассчитанный реализующий ордер покупка отображен в таблице 4.

Таблица 4 – Рассчитанный реализующий ордера покупка

Приобретаемое кол-во, BTC	Цена, USDT	Стоимость, USDT
1,011515	7128,80956032	7210,89780241

После исполнения реализующего ордера покупки, будет потрачено 7210,89780241 USDT на приобретение 1,011515 BTC по цене 7128,80956032USDT, с учётом комиссии будет получено 1,00999773 BTC. Разница между полученным количеством (1,00999773 BTC) и изначальным количеством (1 BTC) является чистой прибылью и практически соответствует 1% от изначального количества.

Таким образом, можно сделать вывод о том, что с учётом небольшой погрешности расчёт реализующего ордера покупки для второго подтипа стратегии выполняется правильно.

На рисунке 3.14 представлены результаты эмуляции работы второго подтипа стратегии по выше представленным параметрам.

```
> cryptocurrency-exchange-trader@1.2.0 emulate:sell-then-buy /Users/kenikh/Projects/cryptocurrency-exchange-trader
> node ./dist/emulation/sell-then-buy.js

Наибольшая цена покупки на бирже: 6999.99999999.
1. Исполнен ордер продажа 0.06426615 BTC x 7017.49999999 USDT (получено 450.98770762 USDT).
2. Исполнен ордер продажа 0.07037143 BTC x 7056.4861111 USDT (получено 496.57501841 USDT).
3. Исполнен ордер продажа 0.07705672 BTC x 7095.47222221 USDT (получено 546.75381629 USDT).
4. Исполнен ордер продажа 0.08437711 BTC x 7134.45833332 USDT (получено 601.98497558 USDT).
5. Исполнен ордер продажа 0.09239293 BTC x 7173.44444443 USDT (получено 662.77555041 USDT).
6. Исполнен ордер продажа 0.10117026 BTC x 7212.43055555 USDT (получено 729.68347454 USDT).
7. Исполнен ордер продажа 0.11078144 BTC x 7251.41666666 USDT (получено 803.32238037 USDT).
8. Исполнен ордер продажа 0.12130567 BTC x 7290.40277777 USDT (получено 884.36719353 USDT).
9. Исполнен ордер продажа 0.13282971 BTC x 7329.38888888 USDT (получено 973.56060059 USDT).
10. Исполнен ордер продажа 0.14544858 BTC x 7368.37499999 USDT (получено 1071.71968066 USDT).
Всего продано 1 BTC.
Всего получено 7210.89780241 USDT (с учётом комиссии).
Исполнен ордер покупка 1.011515 BTC x 7128.80956032 USDT (потрачено 7210.89780241 USDT).
Куплено 1.00999773 BTC (с учётом комиссии).
Потрачено: 7210.89780241 USDT.
BTC: 1 -> 0 -> 1.00999773 (ожидалось: ~ 1.01).
USDT: 0 -> 7210.89780241 -> 0 (ожидалось: 0).
```

Рисунок 3.14 – Эмуляция работы второго подтипа стратегии

3.4 Демонстрация работы приложения

Криптовалютная биржа Poloniex предоставляет удобный инструмент для проведения анализа торгов.

Так на рисунке 3.15 изображены результаты торговли за неделю для первого подтипа стратегии (сначала купить, затем продать).

Тестирование программы на первом подтипе стратегии выполнено со следующими параметрами:

- биржа – Poloniex;
- пара – LTC/BTC;
- объем используемого депозита – 50%;
- стоп-рейт – 0,5%;
- процент отступа от наименьшей цены продажи – 0,25%;
- количество создаваемых ордеров – 25;
- процент перекрытия хода цены – 12,5%;
- процент увеличения объема каждого последующего ордера – 9,5%;
- процент чистой прибыли – 1%.

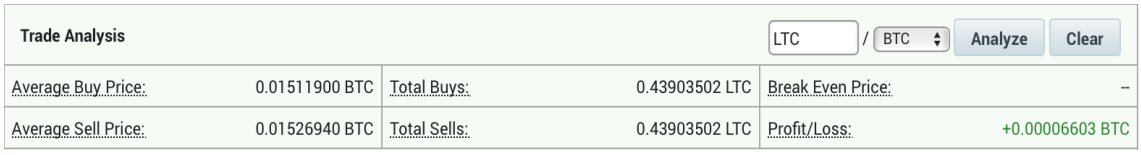

Рисунок 3.15 – Результат работы первого подтипа стратегии

На рисунке 3.15 видно, что было продано 0,43903502 LTC и столько же было куплено, при этом продано было в плюс, и прибыль составила 0,00006603 BTC.

На рисунке 3.16 изображены результаты торговли за два дня для второго подтипа стратегии (сначала продать, затем купить).

Рисунок 3.16 – Результат работы второго подтипа стратегии

Тестирование программы на втором подтипе стратегии было выполнено со следующими параметрами:

- биржа – Poloniex;
- пара – BTC/USDT;
- объем используемого депозита – 50%;
- стоп-рейт – 0,5%;
- процент отступа от наименьшей цены продажи – 0,25%;
- количество создаваемых ордеров – 20;
- процент перекрытия хода цены – 9,5%;
- процент увеличения объема каждого последующего ордера – 5,5%;
- процент чистой прибыли – 1%.

В данном подтипе стратегии купленное количество должно быть больше проданного количества, при этом количество расходов второй валюты пары должно быть близко к нулю (близко, поскольку возможна небольшая погрешность).

Так на рисунке 3.27 видно, что было продано 0,01124089 BTC, а куплено 0,01126153 BTC, при этом количество USDT не уменьшилось, а увеличилось на 0,00000004 из-за погрешности в вычислениях. Таким образом прибыль составила примерно 0,00002064 BTC.

3.5 Выводы по разделу

В данном разделе была произведена разработка приложения. Сначала был спроектирован и реализован базовый класс, который в дальнейшем будет использоваться реализуемыми стратегиями торговли. Был спроектирован и реализован класс стратегии торговли, который включает в себя два подтипа.

Для того, чтобы убедиться в том, что наша математическая модель и программа в целом работают правильно, мы провели тестирование на экспериментальных данных.

Убедившись в том, что наша математическая модель корректна и на экспериментальных данных программа работает правильно, было принято решение провести тестирование на живом рынке и продемонстрировать результаты работы программы.

ЗАКЛЮЧЕНИЕ

В данной работе был проведен обзор криптовалют и криптовалютных бирж, были выделение основных принципов торговли криптовалютой и были сформулированы функциональные требования к системе.

В ходе работы было выполнено математическое моделирование торговли на криптовалютной бирже, в результате чего была получена стратегия торговли, включающая в себя два подтипа стратегии.

Первый подтип стратегии торговли на криптовалютной бирже – сначала продать, затем приобрести. Работа первого подтипа стратегии основана на принципе покупай частями при падении цены (начальная часть цикла) и продай дороже все купленное при росте цены (финальная часть цикла).

Второй подтип стратегии торговли на криптовалютной бирже – сначала приобрести, затем продать. Принцип работы зеркален первому подтипу и основан на принципе продавай частями при росте цены (начальная часть цикла) и купи дешевле все проданное при падении цены (финальная часть цикла).

И наконец, был разработан базовый класс, который отвечает за взаимодействие с криптовалютными биржами, разработаны и реализованы алгоритмы торговли. Было проведено тестирование алгоритмов на экспериментальных данных и тестирование разработанной системы в целом.

Таким образом, все поставленные задачи были успешно выполнены.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 CoinMarketCap – Cryptocurrency Market Capitalization. URL: https://coinmarketcap.com (дата обращения: 04.04.2018).

2 Bitcoin – Open source P2P money. URL: https://bitcoin.org/ (дата обращения: 04.04.2018).

3 Ethereum. URL: https://www.ethereum.org (дата обращения: 04.04.2018).

4 Litecoin – Open source P2P digital currency. URL: https://litecoin.org (дата обращения: 04.04.2018).

5 Zcash – Privacy-protecting digital currency. URL: https://z.cash (дата обращения: 04.04.2018).

6 Ripple – One Frictionless Experience To Send Money Globally. URL: https://ripple.com (дата обращения: 04.04.2018).

7 Binance – Cryptocurrency exchange. URL: https://binance.com (дата обращения: 12.05.2018).

8 Bittrex International. URL: https://bittrex.com (дата обращения: 12.05.2018).

9 Poloniex – Digital Asset Exchange. URL: https://poloniex.com (дата обращения 12.05.2018).

10 Yobit – Crypto Currency Exchange. URL: https://yobit.net/ (дата обращения: 12.05.2018).

11 Exmo – International Cryptocurrency Exchange. URL: https://exmo.me (дата обращения: 12.05.2018).

12 Node.js – Is a JavaScript runtime built on Chrome's V8 JavaScript engine. URL: https://nodejs.org/ (дата обращения: 10.10.2018).

13 TypeScript. URL: https://www.typescriptlang.org (дата обращения: 10.10.2018).

14 MongoDB – The most popular database for modern apps. URL: https://www.mongodb.com (дата обращения: 10.10.2018).

15 Mongoose ODM – Elegant MongoDB object modeling for Node.js. URL: https://mongoosejs.com (дата обращения: 10.10.2018).

16 CCXT – CryptoCurrency eXchange Trading Library. URL: https://github.com/ccxt/ccxt (дата обращения: 05.12.2018).

ПРИЛОЖЕНИЕ 1

Алгоритм работы метода start() обоих подтипов стратегии

Рисунок П1.1 – Алгоритм работы метода start() обоих подтипов стратегии

ПРИЛОЖЕНИЕ 2

Алгоритм работы метода direction() первого подтипа стратегии

Рисунок П2.1 – Алгоритм работы метода direction() первого подтипа стратегии

ПРИЛОЖЕНИЕ 3

Алгоритм работы метода observe() первого подтипа стратегии

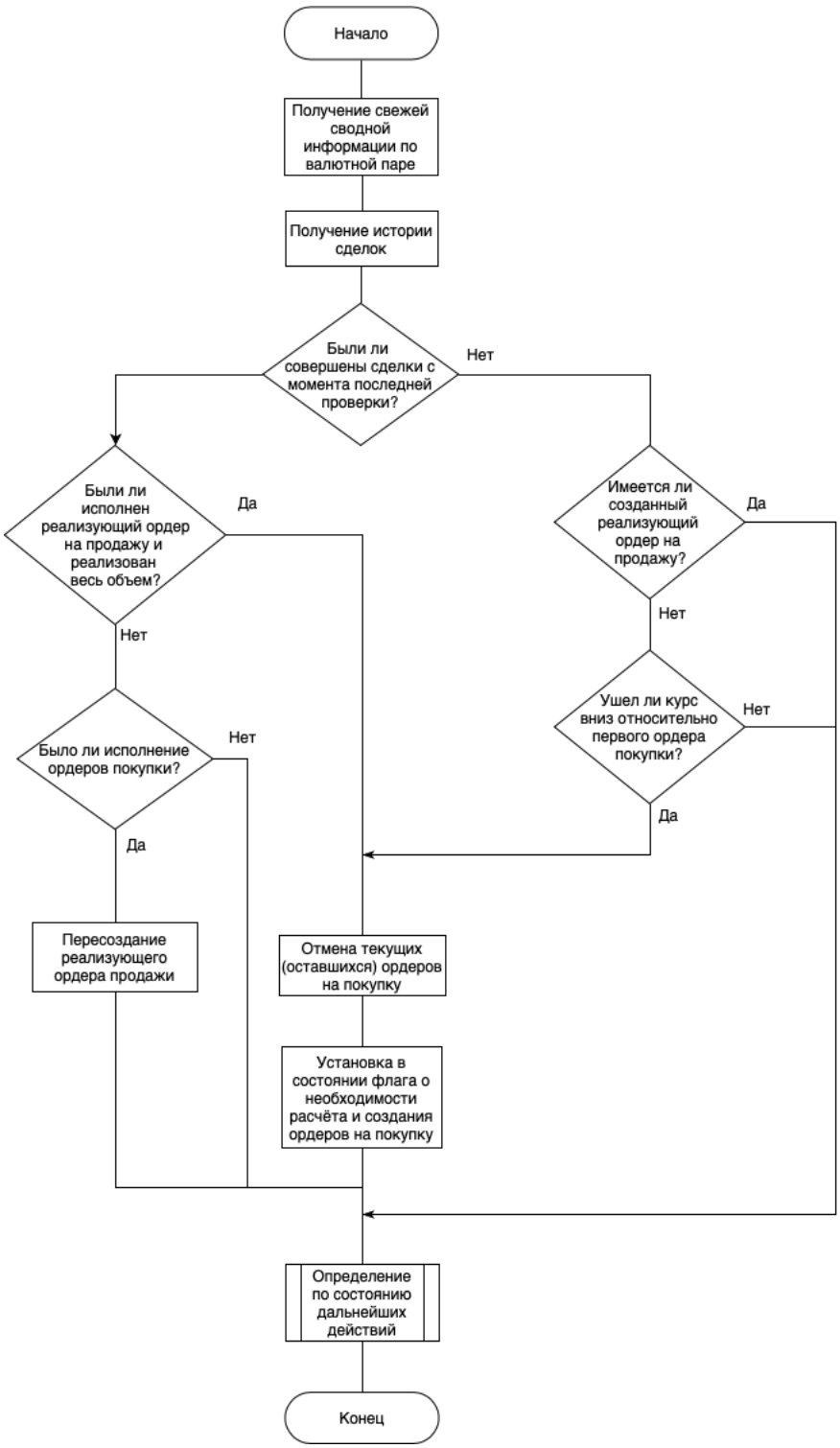

Рисунок П3.1 – Алгоритм работы метода observe() первого подтипа стратегии

ПРИЛОЖЕНИЕ 4

Алгоритм работы метода direction() второго подтипа стратегии

Рисунок П4.1 – Алгоритм работы метода direction() второго подтипа стратегии

ПРИЛОЖЕНИЕ 5

Алгоритм работы методы observe() второго подтипа стратегии

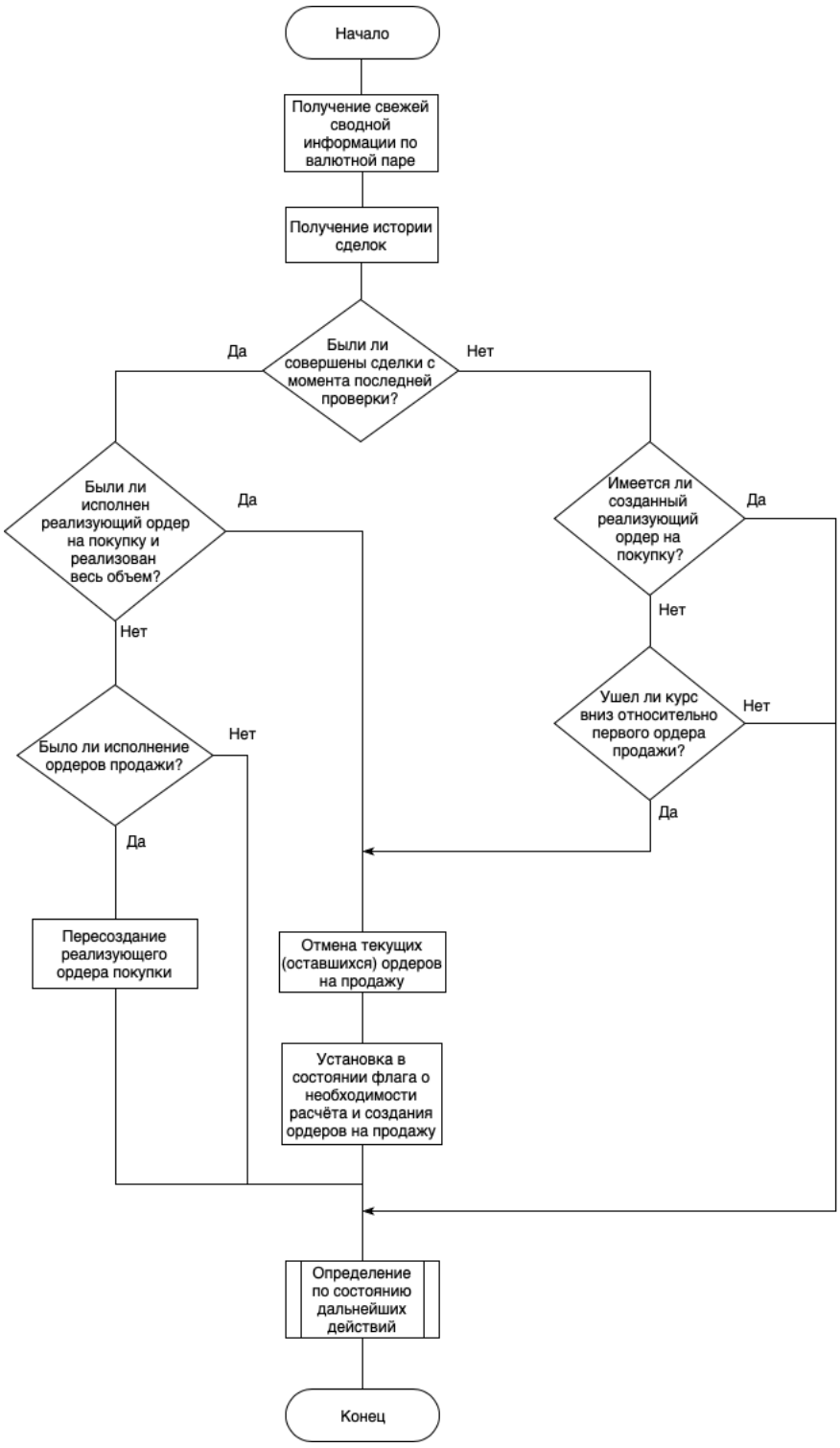

Рисунок П5.1 – Алгоритм работы метода observe() второго подтипа стратегии