ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА ИНФОРМАЦИОННЫХ И РОБОТОТЕХНИЧЕСКИХ СИСТЕМ

АВТОМАТИЧЕСКАЯ СИСТЕМА ТОРГОВЛИ НА ВАЛЮТНОМ РЫНКЕ

Выпускная квалификационная работа студента
обучающегося по направления подготовки 09.03.02 Информационные системы и технологии
очной формы обучения, группы 07001407
Дворяшина Кирилла Сергеевича

Научный руководитель
к.т.н., доцент
Гахов Р.П.

БЕЛГОРОД 2018

РЕФЕРАТ

Автоматическая система торговли на валютном рынке – Дворяшин Кирилл Сергеевич, выпускная квалификационная работа бакалавра, Белгород, Белгородский государственный национальный исследовательский университет (НИУ «БелГУ»), количество страниц 52, включая приложения 64, количество рисунков 23, количество использованных источников 30.

ОБЪЕКТ ИССЛЕДОВАНИЯ: процесс оптимизации торговли на валютном рынке.

ПРЕДМЕТОМ ИССЛЕДОВАНИЯ: является спроектированная и реализованная автоматическая система торговли на валютном рынке.

ЦЕЛЬ РАБОТЫ: оптимизация процесса торговли на валютном рынке.

ЗАДАЧИ ИССЛЕДОВАНИЯ: провести анализ предметной области, исследовать рыночные неэффективности, провести обзор существующих торговых стратегий, спроектировать и протестировать на истории торговый алгоритм, разработать автоматическую торговую систему, разработать веб-приложение с возможностью посмотреть информацию о торговом счете и результатах торговли, протестировать все компоненты системы.

ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ: произведен анализ предметной области; разработана и протестирована автоматическая система торговли на валютном рынке.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В течении последних нескольких десятилетий финансовые рынки прошли через все стадии развития. За последнее десятилетие мир увидел фазы роста и падения. С развитием рынков рос интерес и к их прогнозированию, что поспособствовало к появлению большого количества инструментов и методик анализа изменения цен на финансовые активы. Так появился фундаментальный и технический анализ. К последнему относится анализ форм и фигур, сформированных ценой. Участники торгов записывали цены активов и строили на их основе графики, на которых отмечали закономерности, изучали характер движения, стараясь найти похожие модели, с целью предсказать дальнейшее направление цены. Таким образом, появилось множество так называемых фигур технического анализа. В дальнейшем трейдеры стали анализировать динамику движения цен с помощью построения рыночных индикаторов, которые представляли собой некоторую функцию от цены. Классическим примером такой функции является простое скользящее среднее.

В связи с развитием информационных технологий, методы анализа и прогнозирования финансовых рынков видоизменяются. Еще только в 20 веке участникам рынка приходилось выписывать значения котировок на котировочные доски в торговых залах, а трейдерам приходилось находится непосредственно в этих залах, чтобы иметь представление о текущих ценах на активы. Особо успешные трейдеры могли позволить себе купить телеграф, который печатал цены на специальной бумаге [1].

Спустя несколько десятилетий доступ к рыночной информации значительно упростился. Сейчас достаточно иметь компьютер или смартфон с доступом в интернет, чтобы получать самые актуальные цены на практически любые финансовые активы. Расчет индикаторов, построение сложных графиков – все это не занимает много времени, а стоимость услуг за

использование сервисов, предоставляющих доступ, к подобной информации стремится к нулю.

В настоящее время существуют сотни индикаторов технического анализа, тысячи ценовых паттернов, в той или иной мере используемых участниками торгов при анализе цен финансовых активов. Совокупность методов анализа и прогнозирования движения цен с помощью индикаторов и поиск закономерностей на графике, называется техническим анализом. Данный вид прогнозирования является одним из самых популярных инструментов для принятия торговых решений, особенно у начинающих трейдеров, т.к. позволяет визуально оценить ситуацию на рынке, определить текущее направление цены и попробовать предсказать ее будущее поведение без углубленных знаний экономики и финансов.

Однако, при всех плюсах технического анализа, есть один существенный недостаток: многие классические стратегии, описанные в книгах по трейдингу, а также все остальные публичные торговые стратегии со временем перестают работать и на смену прибыльным сделкам приходит регулярный убыток. Чем известнее становится стратегия и чем больше трейдеров начинают ее использовать в своей торговле, тем меньше ее эффективность. В связи с этим рабочие алгоритмы и идеи зачастую остаются в тайне, а авторы таких исследований предпочитают зарабатывать сами, либо брать средства сторонних инвесторов под свое управление.

Примером этому служит известная на весь мир стратегия Черепах, принесшая миллионы долларов трейдерам по всему миру, начиналась как эксперимент между двумя успешными друзьями [2]. Спустя 30 лет об этой стратегии знает каждый, однако она едва ли может принести хоть какой-то разовый доход.

Также, одной из самых распространенных причин получения убытков на рынке является отсутствие торговой системы или плана торговли.

Бессистемность в принятии торговых решений — довольно частое явление среди трейдеров (особенно среди начинающих). Даже если трейдер

торгует рабочую стратегию, но делает это бессистемно, то это обязательно приводит к убыткам.

Торговая система представляет собой группу конкретных правил и параметров, которые объединяются в одно целое для создания сигналов на покупку и продажу финансового актива. Подобные системы могут быть спроектированы и разработаны с использованием множества различных технологий, включая Microsoft Excel, MATLAB®, TradeStation, R, Python. Кардинальным отличием торговой стратегии от ручной торговли является то, что в первом случае, каждая сделка совершается строго по заданному алгоритму. Таким образом из процесса торговли полностью исключается человеческий фактор, который зачастую и приводит к убыткам [3].

Существует множество различных инструментов, которые можно использовать при проектировании торгового алгоритма. В классических торговых системах используется два или более технических индикатора, которые объединяются и используются как фильтр, для отбора качественных точек входа в сделку. Как видно технические индикаторы являются наиболее распространенными, однако многие торговые системы включают в себя и фундаментальный анализ. В его основе лежит анализ различных фундаментальных рыночных данных, таких как доход (в случае анализа компаний), анализ денежного потока, задолженность к капиталу, учет изменения ключевых ставок, сезонность и т.д. Некоторые, особо интеллектуальные системы, могут использовать для анализа даже новости, сообщения из социальных сетей и другие данные со всего интернета. Любая информация, которая может быть представлена в цифровом виде, может оказаться полезной.

В данной выпускной квалификационной работе главным образом рассматривается вопрос создания такой автоматической торговой системы, которая будет приносить максимальную прибыль с ограниченным риском. Система должна получать данные о ценах с биржи, проводить их анализ, принимать решение и совершать сделки.

Цель работы заключается в оптимизации процесса торговли, путем разработки автоматической системы торговли на валютном рынке. Разработка данного программного обеспечения имеет большое значение: оно исключит влияние человеческого фактора на процесс торговли, избавит трейдера от ответственности за принятые решения и позволит улучшить свои торговые результаты.

В связи с этим были поставлены следующие задачи:

- провести анализ предметной области;
- исследовать рыночные неэффективности;
- провести анализ существующих торговых систем;
- спроектировать и протестировать на истории торговый алгоритм;
- разработать автоматическую торговую систему на основе полученного алгоритма;
- разработать веб-приложение;
- протестировать все компоненты результирующего проекта.

Методологической основой выпускной квалификационной работы стал системный подход и принцип научной объективности, позволившие рассмотреть существующие торговые системы, проанализировать рыночные неэффективности, а также разработать и протестировать различные торговые алгоритмы.

Структура работы отвечает поставленным целям и задачам. Работа состоит из введения, трех глав, 9 пунктов, заключения и списка использованных источников.

1 Анализ предметной области

1.1 Алгоритмический трейдинг

Прежде чем приступить к рассмотрению данной предметной области, необходимо определить несколько понятий.

Алгоритмический трейдинг (автоматический, алготрейдинг) – это процесс использования компьютеров, запрограммированных для выполнения определенного набора инструкций (алгоритма) для размещения сделок на бирже с целью получения прибыли, со скоростью и частотой, которой невозможно добиться при ручной торговле [4]. Определенные наборы правил основаны на сроках, цене, количестве или любой другой математической модели. Помимо возможностей и статистического преимущества, алгоритмическая торговля делает рынки более ликвидными, превращает торговлю в более систематический процесс, исключая влияние человеческих эмоций на торговые операции. В США, как на самом ликвидном рынке, доля автоматических торговых систем составляет более 80% [5]. То есть 4 из 5 заявок выставляет робот. В России алготрейдинг занимает только 25-30% операций суммарно на всех торговых площадках. Однако, по данным Московской Биржи, доля алгоритмических сделок на срочном рынке достигает 80-85%. Связано это с тем, что сделки на срочном рынке носят спекулятивный характер, в отличие от секции фондового рынка, где преимущественно торгуются акции российских компаний.

Бета – среднерыночный доход, который инвестор или трейдер может получить путем займа своих средств надежному заемщику. Обычно в роли такого надежного заемщика является государство или известный банк. Допустим, условная безрисковая ставка дохода равна 7.5% годовых (то есть равна доходности государственных облигаций). Если инвестор возьмет на себя больше риска и вложит свои средства в компанию класса А, то сможет получить доходность на уровне 8.5-9% годовых. Компании с более низким

кредитным рейтингом и более высокой долговой нагрузкой могут предложить заемщикам 10-13% годовых. Для инвестора этот доход зарабатывает заемщик, которому были переданы вложенные средства. Бету в основном зарабатывают крупные инвестиционные банки, индексные фонды, долгосрочные институциональные инвесторы. Как известно, чем больший риск несет в себе инвестиция, тем больший потенциальный доход она может принести (рисунок 1.1). Более активным трейдерам, которые совершают сделки спекулятивного характера, данный уровень доходности не представляет интереса, поскольку представляет собой некий пассивный источник дохода, который слегка покрывает уровень инфляции [6].

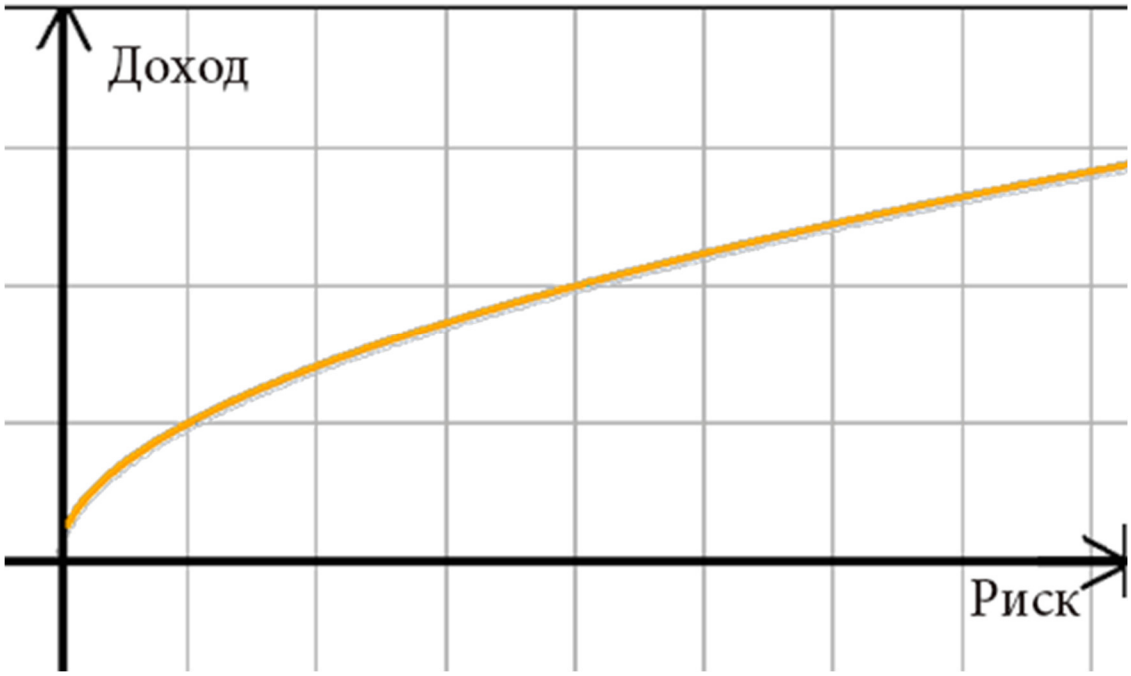

Рисунок 1.1 – Зависимость дохода инвестиций от уровня риска

Альфа – это статистический показатель, отражающий изменение стоимости результатов торговли или портфеля к индексу. Это доход выше среднерыночного (беты). Проще говоря, альфой измеряют рост портфеля (или результатов торговли), который не связан с общим ростом рынка и индексов. Например, если российский фондовый рынок, а соответственно и индекс, за последний год вырос на 20%, а управляющий фондом показал

результат в виде роста на 30%, то говорят, что альфа фонда равна 10 процентам. К сожалению, в данном случае, доход не возникает из воздуха. Чтобы заработать альфу, необходимо, чтобы эти деньги кто-то проиграл. Если фонд или частный трейдер стабильно получает доход выше среднерыночного, обгоняя индекс и получая альфу, значит кто-то этот доход теряет. Отметим, что заработок альфы – относится к так называемом игре с отрицательной суммой. Так как биржа и брокеры получают комиссии от сделок на каждом этапе торговли, то сумма всех выигрышей на бирже меньше, чем сумма проигрышей [7].

Очевидно, что суть использования данных показателей заключается в том, чтобы иметь представление об эффективности торговли, как частного трейдера, так и управляющих фондом. Сравнивая оба показателя, между собой, становится видно сколько фонд или управляющий заработал без (помимо) влияния рыночного роста или падения. В случае, если альфа и бета фонда или частного трейдера равны, либо альфа всего на несколько процентов превосходит, то возникает вопрос в том, стоит ли останавливать выбор на данной инвестиции, если можно просто приобрести пай в индексном фонде и получить среднерыночный результат без участия посредников. Очевидно, альфа зачастую используется для измерения успешности торговли, чтобы понять, насколько они лучше, чем среднестатистический рыночный портфель.

Мы определили, что любой доход выше среднего (альфа) достигается путем, в прямом смысле, отнимания денежных средств у других участников рынка – конкурентов. Необходимо иметь ввиду, что в каждой торговой сделке конечный трейдер пытается заработать деньги и каждая ситуация имеет следующие свойства: текущий торговый объем (ликвидность) – количество средств, которые потенциально могут превратиться в доход по сделке. А также количество конкурентов. У каждого из этих свойств может быть два состояния: много и мало. Соответственно в каждый момент времени на каждом рынке может сформироваться следующая комбинация:

- мало денег, мало конкурентов;
- мало денег, много конкурентов;
- много денег, мало конкурентов;
- много денег, много конкурентов.

Логично предположить, что рентабельность торговли будет максимальной только лишь в одной из перечисленных комбинаций: когда на рынке много денег и конкуренция находится на минимальном уровне. К сожалению, подобная ситуация если и появляется, то не сохраняется слишком долго. И в конечном итоге рынок всегда стремиться лишь к одному состояния, когда на нем большая конкуренция и большой торговый объем [7].

Этим обусловлен большой спрос на автоматические торговые системы, позволяющие торговать подобные ситуации, которые длятся в течение короткого промежутка времени и, следовательно, которые не всегда способен отследить человек.

1.2 Преимущества и недостатки алгоритмических систем

Используя автоматическую систему торговли, конечный пользователь должен осознавать, что существуют некоторые преимущества и недостатки данной конкретной системы. В целом, можно выделить следующие преимущества использования алгоритмических систем в сравнении с ручной торговлей:

- снимает ответственность с трейдера. Когнитивные предубеждения сильно сказываются на торговых доходах и торговых системах. Трейдеры, которые не могут справиться с потерями, жалеют о своих решениях, а те, кто недавно потерял деньги, могут упустить новые возможности. Торговые системы избавляют трейдеров от фактического принятия решений о покупке и продаже и формируют более предсказуемые результаты;

– экономит время. Торговые системы, которые грамотно разработаны и оптимизированы, требуют гораздо меньше усилий для поддержания, чем ручная торговля. Во втором случае трейдеру придется сидеть за рабочим местом в течение всей торговой сессии, самостоятельно анализируя рыночные данные и вручную совершать сделки. Некоторые торговые сессии доступны только в другом часовом поясе, так например, американский рынок работает до 12 часов ночи, по московскому времени. Другие рынки торгуются только в ночное для нас время. Не многие трейдеры могут позволить себе проводить торговли в подобное время суток;

– возможность передать свой алгоритм сторонним разработчикам. Многие разработчики программного обеспечения специализируются на разработке автоматических торговых систем. Если трейдер, не владеющий навыками программирования, опишет алгоритм своей торговли и передаст его на разработку, то в результате сможет автоматизировать свои действия, проанализировать стратегию на истории и возможно оптимизировать параметры системы.

К недостаткам можно отнести:

– наличие особых навыков. Для разработки собственных автоматических торговых систем требуется четкое понимание технического анализа и наличие навыков программирования. Несмотря на наличие сторонних разработчиков, готовых взять этап разработки на себя, конечному трейдеру по-прежнему придется иметь дело с поддержкой автоматической системы в рабочем состоянии, регулярного мониторинга и исправления недочетов в случае наличия таковых;

– сложности реализации. Алгоритмические торговые системы должны включать в себя множество различных дополнительных функций, помимо самого торгового алгоритма, таких как проскальзывание, транзакционные издержки, обработка ошибок со стороны биржи, наличие системы контроля против сбоев и т.д. Тщательно протестировав алгоритм на истории перед запуском на реальном счете не удается продумать наперед все возможные

исходы и исключительные ситуации. Это означает, что существует некоторая степень неопределенности. Многие проблемы могут возникать в реальной торговле, что может является достаточно дорогостоящим и трудно исправимым процессом.

– требует больших первоначальных инвестиций. Разработка и тестирование торговой системы занимает достаточно много времени и первоначальных инвестиций. В течение первого времени, даже при условии прибыльной торговли, конечный трейдер не будет получать доход с вложений, т.к. понадобиться достаточно много времени, чтобы окупить изначальные вложения. Подобные системы также требуют постоянного обслуживания для точной настройки параметров и устранения любых изменений на рынке.

Подводя итоги, стоит отметить, что несмотря на преимущества и недостатки, нет сомнений в том, что в прошлом было много успешных торговых систем, и в будущем их будет намного больше. Самым известным примером успешной торговой системы была система, спроектированная и реализованная Ричардом Деннисом и Биллом Экхардтом – авторами стратегии Черепах [8]. Совершенно очевидно, что для проектирования прибыльного алгоритма и разработки на его базе автоматической торговой системы, необходимо иметь глубокие знания не только в области программирования, но и в понимание биржевого рынка.

1.3 Виды алгоритмических систем

Существует несколько методологий торговли, которые подходят под термин трейдинг в широком смысле этого слова. Каждый из которых подходит определенному виду темперамента конечного трейдера. С точки зрения технического анализа алгоритмические системы можно классифицировать следующим образом (рисунок 1.2):

Рисунок 1.2 – Классификация автоматических торговых систем с точки зрения технического анализа

Как видно, торговые системы основанные на техническом анализе можно условно поделить на два больших класса: трендовые и контртрендовые системы. Эта противоположных друг другу по своей сути класса и соответствуют разным состояниям рынка: направленное и не направленное. Поэтому зачастую при проектировании алгоритмической системы ориентируются только на один из этих классов, т.к. рынок может находится только в одном из них в каждый момент времени [9].

С точки зрения торговых стратегий, условно выделяют следующие виды:

- скальпинг;
- алгоритмический трейдинг;

- позиционный трейдинг;
- инвестирование;
- продажа своих инвестиционных услуг.

Так как цель данной работы состоит в разработке именно автоматической торговой системы, рассмотрим подробнее их классификацию и виды торговых стратегий, лежащих в их основе. Как было отмечено выше, алгоритмическая торговля – это такой вид торговли, в котором задействован непосредственно не сам трейдер, а система, которая, без какого-либо человеческого вмешательства, занимается анализом биржевых данных, информацией о финансовом активе и на основе данного анализа принимает торговые решения. В основе таких систем лежат передовые технологии, сложные математические модели и формулы, которые позволяют получить статистическое превосходство перед другими участниками рынка. Впрочем, часто превосходство достигается не за счет сложных математических вычислений, а, банально, в следствие большой скорости выполнения транзакций, недостижимых при ручной торговле. Некоторые инвестиционные и торговые стратегии, такие как арбитраж, следование тренду, торговля перед ребалансировкой крупных фондов, возврат к среднему или скальпинг могут быть улучшены посредством алгоритмической торговли. Как видно существует большое разнообразие торговых стратегий, которые применимы в алготрейдинге и на основе которых строятся современные алгоритмические системы. Данный пункт посвящен краткому обзору видов таких систем.

Арбитраж – это торговая стратегия в основе которой лежит покупка и перепродажа финансового актива с целью получения прибыли от кратковременной разницы в цене между двумя рынками [10]. Данный вид неэффективности возникает в случае, если финансовый актив, торгуемый в двух независимых друг от друга местах, имеет достаточно большую разницу в цене. Таким образом, если купить актив задешево в одном месте и сразу же продать его в другом по цене дороже, образуется доход равный разнице в

цене покупки и продажи (спред). Другим примером служит покупка и одновременная продажа фьючерса на один и тот же актив, в случае, когда спред между ними отклоняется от среднего значения полученного на некотором промежутке времени [12]. Подобные отклонения спреда обычно носят краткосрочный характер, поэтому покупка данной конструкции и ее продажа в будущем позволяет получить прибыль с достаточно ограниченными рисками (это так называемая дельта-нейтральная торговля). Тем не менее, обычно арбитраж, это гораздо более сложный процесс, чем описанный в данном примере и может быть применен к самым разнообразным финансовым инструментам, включая акции, сырьевые товары, валюту, производные инструменты.

Торговля перед ребалансировкой крупных фондов – относиться к более спокойному виду торговли и скорость исполнения сделок здесь не так важна. В корне этой стратегии лежит принцип следования за крупными биржевыми игроками. В их роли могут выступать пенсионные фонды, паевые инвестиционные фонды, индексные фонды [13]. Пожалуй, самым ярким примером служит MSCI Russia – фондовый индекс российского рынка, входящий в группу индексов развивающихся рынков MSCI Emerging Markets. В состав данного индекса входят 20 лучших российских компаний (по версии их аналитической компании). Все эмитенты входящие в данный индекс имеют разный вес, который в свою очередь зависит от капитализации, количества акций в свободном обращении (free float) и возможности привлечения иностранного капитала. На рисунке 1.3 изображен состав индекса MSCI Russia на 1 июня 2018 года.

MSCI Code	Ценная бумага	Цена	Валюта	Shares FIF Adjusted	Вес,%
1896804	SBERBANK RUSSIA COM (RUB)	155.39000	RUB	10,793,474,000	19,00
1884205	GAZPROM (RUB)	120.00000	RUB	10,653,080,805	14,48
1874804	LUKOIL HOLDING (RUB)	2738.00000	RUB	425,281,628	13,19
6071002	MAGNIT GDR	35.75000	USD	307,324,404	7,06
2056003	TATNEFT COMMON (RUB)	392.10000	RUB	1,525,083,700	6,77
3241302	NOVATEK GDR	112.40000	USD	91,089,180	6,58
2491705	NORILSK NICKEL MMC (RUB)	7955.00000	RUB	55,385,917	4,99
3481902	ROSNEFT (RUB)	300.00000	RUB	1,165,799,560	3,96
3692203	VTB BANK (RUB)	0.06592	RUB	5,184,216,534,935	3,87
2375301	MOBILE TELESYS ADR	8.81000	USD	499,595,394	2,83
6994201	ALROSA (RUB)	89.18000	RUB	2,577,737,970	2,60
1885706	SURGUTNEFTEGAZ COMN (RUB)	30.20000	RUB	7,145,198,941	2,44
1885703	SURGUTNEFTEGAZ PREF (RUB)	28.17500	RUB	6,931,798,412	2,21
7143701	MOSCOW EXCHANGE (RUB)	101.03000	RUB	1,481,113,720	1,70
3099803	NOVOLIPETSK STEEL (RUB)	112.50000	RUB	1,198,645,448	1,53
6187102	INTER RAO UES (RUB)	3.96800	RUB	31,320,000,000	1,41
2517402	SEVERSTAL (RUB)	739.20000	RUB	167,543,732	1,40
6971101	PHOSAGRO GDR	14.20000	USD	116,561,656	1,06
6086401	RUSHYDRO (RUB)	0.78000	RUB	96,563,866,222	0,85
2788702	TRANSNEFT PREF (RUB)	157000.00000	RUB	466,462	0,83
1885503	ROSTELECOM COMMON (RUB)	71.20000	RUB	901,220,234	0,73
3058501	AFK SISTEMA GDR	4.50000	USD	168,875,000	0,49

Рисунок 1.3 – Состав индекса MSCI Russia на 1 июня 2018

Решение о составе пересматривается ежеквартально. Логично предположить, что акции компаний, которые добавляются в данный индекс начинает сильно расти в цене. Это связано как с фундаментальными причинами, так и с психологическими: инвесторы проявляют начинают проявлять большее доверие и добавлять их в свои портфели [14].

Скальпинг – по некоторым мнениям, самый быстрый и высокодоходный метод торговли с достаточно ограниченными рисками. Особенностью данной стратегии является: большая частота сделок, максимально короткое время удержания открытой позиции и закрытие сделки при достижении минимального дохода. Риск у таких сделок так же ограничен определенным уровнем (так называемый стоплосс) при достижении которого позиция закрывается с убытком. Несмотря на простоту и доступность данного метода он несет в себе множество недостатков и зачастую балансирует на уровне небольшой доходности [15]. Принимая тот

факт, что сделки совершаются очень часто большая часть прибыли уходят на покрытие комиссий брокера, а быстрая фиксация минимальной прибыли ограничивает потенциальную доходность каждой сделки.

Возврат к среднему – покупка финансового актива в случае, если наблюдается его падение в надежде, что цена вернется на прежний уровень и наоборот продажа в случае роста с фиксацией прибыли при падении. В некоторых источниках данная стратегия называется контр-трендовой торговлей [16]. Контр-трендовый рынок – рынок, который стоит на месте. Такое состояние наблюдается, когда на рынке нет доминирующей стороны, крупного продавца или покупателя, который двигал бы цену. Стратегии возврата к среднему являются неоднозначными и более рискованным методом торговли на бирже. Однако, несмотря на это – это один из самых любимых и часто используемых методов торговли среди начинающих трейдеров. Причиной этому служит человеческая психология, так как психологически легче покупать падающий актив в надежде на то, что подобное падение носит краткосрочный характер и в цена будущем обязательно вернется на прежний, более высокий уровень. Человеческий мозг устроен таким образом, что люди подсознательно ищут скрытые закономерности, привязываются к круглым и средним числам и дают субъективную оценку происходящему [17].

Тестирование данной стратегии на истории показывает, что доходность торговли напрямую зависит от состояния рынка: при наличии ярко выраженного трендового (однонаправленного) движения подобный метод торговли приносит большие убытки, которые не сравнимы с получаемой прибылью. Однако в случае отсутствия тренда, когда цена имеет пилообразный характер убыток несравнимо мал по отношению к прибыли. Результаты стратегии возврата к среднему на двух различных стадиях рынках представлены на рисунке 1.4 – сравнение доходностей на трендовом и спокойном рынке.

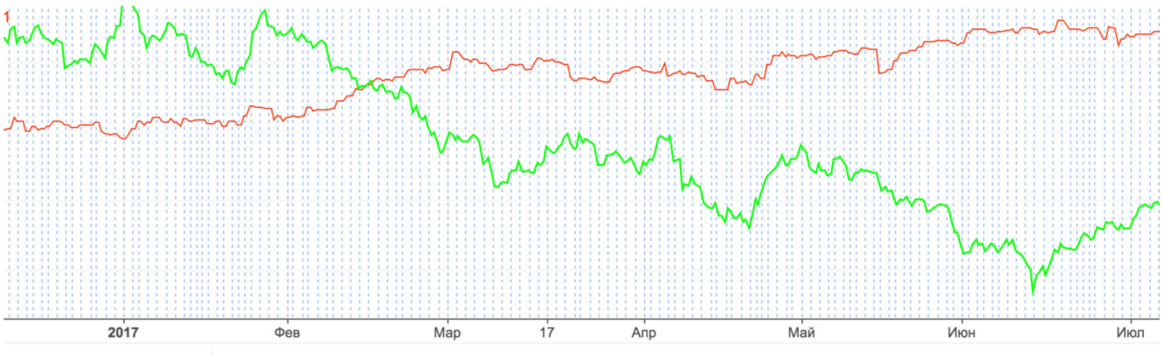

Рисунок 1.4 – Сравнение доходностей на трендовом и спокойном рынке

Зеленым цветом представлена доходность стратегии на трендовом рынке с ярко выраженным движением цены. Красным цветом изображена доходность стратегии на рынке с противоположным состоянием. Как видно, в разные фазы рынка доходность по одной и той же стратегии значительно отличается. Знаменитые трейдеры Лукас и Лебо в своей книге «Компьютерный анализ фьючерсных рынков» [18] в ходе своих исследований сделали вывод, что контртрендовые стратегии рано или поздно приводят к неожиданным большим убыткам и категорически не рекомендуют подобный метод к торговле.

Однако, существует иной способ использовать метод возврата к среднему. Если предположить существование такого рынка, который всегда пребывает в устойчивом не трендовом состоянии и на котором есть достаточно большая ликвидность, чтобы совершать сделки, то данный вид торговли может принести несравнимые результаты, в условиях ограниченного риска. Математики называют такое состояние стационарным – свойство процесса сохранять свои свойства с течением времени. К сожалению, не существует такого рынка и финансового актива, который подходил бы под подобные условия. Тем не менее, в условиях современной биржевой торговли, у трейдеров появилась возможность создать синтетический инструмент, который будет подходить под заданные условия.

Лукас и Лебо не учитывали торговлю стратегий возврата к среднему с точки зрения синтетического инструмента, когда рекомендовали избежать использование данной стратегии в своей торговле, что дает возможность провести собственное исследование и проверить можно ли разработать такую автоматическую систему, которая будет использовать алгоритм, реализующий данную рыночную неэффективность, и зарабатывающую доход выше среднерыночного.

1.4 Тестирование и анализ публичных стратегий

Прежде чем приступить к проектирование собственного торгового алгоритма необходимо провести анализ публичных стратегий и алгоритмических систем. В ходе данной выпускной квалификационной работы был проведен анализ публичных стратегий и их тестирование на истории. Введем несколько понятий.

Тестирование – моделирование стратегии на исторических данных. Моделирование работы торгового алгоритма на рыночных данных в прошлом, с целью оценки эффективности торговли.

Доходность – показатель эффективности вложений, например, проценты или дивиденды, полученные от определенной защиты. Как правило, доходность выражается в виде годовой процентной ставки, основанной на стоимости инвестиций, текущей рыночной стоимости или номинальной стоимости. Доходность может быть известной или ожидаемой в зависимости от сроков и вида финансового актива [19]. Например, инвестиции в облигации считают инвестицией с фиксированной купонной доходностью. В рамках алгоритмических систем, доходностью считается показатель эффективности заработка от вложенных в систему средств.

Фактор восстановления – характеристика торговой системы или результатов частного трейдера, которая определяется как отношение абсолютной прибыли к максимальной просадке. Фактор восстановления

является одним из показателей, которые используют для сравнения нескольких торговых алгоритмов между собой. Фактор восстановления оказывает, насколько прибыль превышает глубину максимальной просадки. Чем выше данный показатель, тем быстрее система восстанавливается после просадки. При сравнении нескольких систем, лучше будет та, у которой фактор восстановления выше.

Максимальная просадка – это максимальное зафиксированное снижение средств в валюте депозита от своего локального максимума. Максимальная просадка показывает разницу между максимумом и минимумом средств, достигнутых в результате торговли.

Z-счет – коэффициент, показывающий насколько случайно система или трейдер совершают сделки. Проведя достаточно долгое время в поисках, проектировании и тестировании торговой системы, которая на практике уже дала реальный доход на небольшом промежутке времени, трейдер получает подтверждение правильности найденного подхода к рынку. Однако, возможно, что это просто случайность и система или трейдер получили доход не благодаря рыночной неэффективности, а просто стечением обстоятельств [20]. Расчет данного показателя тесно связан с нормальным распределением. Не вдаваясь в подробности, для оценки эффективности стратегии, нужно знать, что если z-счет близок к нулю, то распределение торговых сделок отличается от нормального распределения. Z-счет последовательности сделок может дать нам информацию о возможной зависимости между результатами подряд идущих сделок. При этом значения Z трактуются так же, как и вероятность отклонения от нуля случайной величины, распределенной по закону стандартного нормального распределения (среднее=0, sigma=1). Если вероятность попадания нормально распределенной случайной величины в диапазоне ±3σ равна 99.74 %, то попадание этого значения за пределы этого интервала с той же вероятностью 99.74% говорит о том, что эта случайная величина не принадлежит данному нормальному распределению. Поэтому "правило трех сигм" читают так:

нормальная случайная величина отклоняется от своего среднего не более, чем на три сигмы. Знак Z говорит нам о типе зависимости. Положительное говорит нам о том, что за прибыльной сделкой наиболее вероятна убыточная, а отрицательное - что за выигрышем последует выигрыш, а проигрыш повлечет за собой также проигрыш.

Рассмотрим несколько классических публичных торговых стратегий, которые часто лежат в основе автоматических торговых систем. Любимый многими трейдерами метод торговли с помощью переворота индикатора MACD. MACD – индикатор дивергенции конвергенции среднего скользящего среднего. Это трендовый индикатор, следующий за импульсом, который показывает взаимосвязь между двумя скользящими средними цен. MACD рассчитывается путем вычитания 26-дневной экспоненциальной скользящей средней (EMA) из 12-дневной EMA [21]. Также рассчитывается девятидневная EMA, называемая «сигнальной линией». Линия расположена поверх MACD и выступает в роли триггера для сигналов покупки и продажи.

Торговлю с помощью индикатора MACD можно разделить на три группы. Пересечение – когда гистограмма MACD падает ниже сигнальной линии, это медвежий сигнал, который указывает, что настало время продавать. И наоборот, когда MACD поднимается над сигнальной линией, индикатор дает бычий сигнал, который предполагает, что цена актива, скорее всего, будет иметь импульс вверх. Многие трейдеры ждут подтвержденного переворота над сигнальной линией, прежде чем войти в позицию, чтобы избежать ложного сигнала или слишком рано войти в позицию. На рисунке 1.5 показан классический переворотный паттерн.

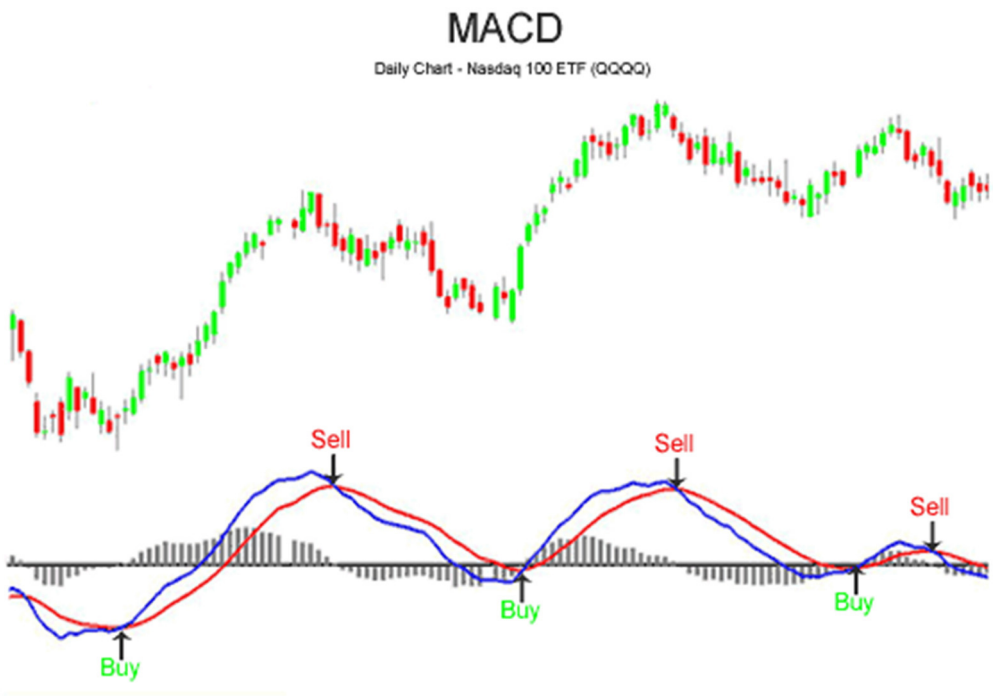

Рисунок 1.5 – Классический переворотный паттерн MACD

Дивергенция – когда цена финансового актива расходится с направлением индикатора, это сигнализирует о смене текущего тренда. Например, цена растущая цена акции и падающие гистограммы MACD могут означать, что текущее движение цены движется к завершению и цена готовится к развороту. И наоборот, если цена акций падает и MACD растет, это может означать, что бычий разворот может произойти в ближайшей перспективе. Трейдеры часто используют дивергенцию в сочетании с другими техническими индикаторами, чтобы найти возможности для прибыльной сделки.

Резкий рост – когда MACD резко возрастает, т.е. более короткое скользящее среднее отходит от более долгосрочного скользящего среднего – это сигнал о том, что актив перекуплен и скоро вернется к нормальным уровням. Трейдеры часто объединяют этот анализ с индексом относительной силы (RSI) или другими техническими индикаторами для проверки перекупленности или перепроданности. Некоторые аналитики также наблюдают за движением выше или ниже нулевой линии, поскольку это сигнализирует положение краткосрочного среднего значения относительно

долгосрочного среднего. Когда MACD выше нуля, кратковременное среднее значение превышает долгосрочное среднее значение, которое сигнализирует о повышении цены. Противоположность истинна, когда MACD находится ниже нуля.

Проведем тестирование переворотной стратегии для того, чтобы оценить насколько выгодно она торговала на истории и можно ли ее использовать в наши дни. Используя публичную информацию о ценах на валютную пару BTC/USD был получен результат тестирования в период с 2012 по 2018 год. На рисунке 1.6 изображена доходность данной переворотной стратегии на таймфрейме 1 час, а также некоторые статистические показатели.

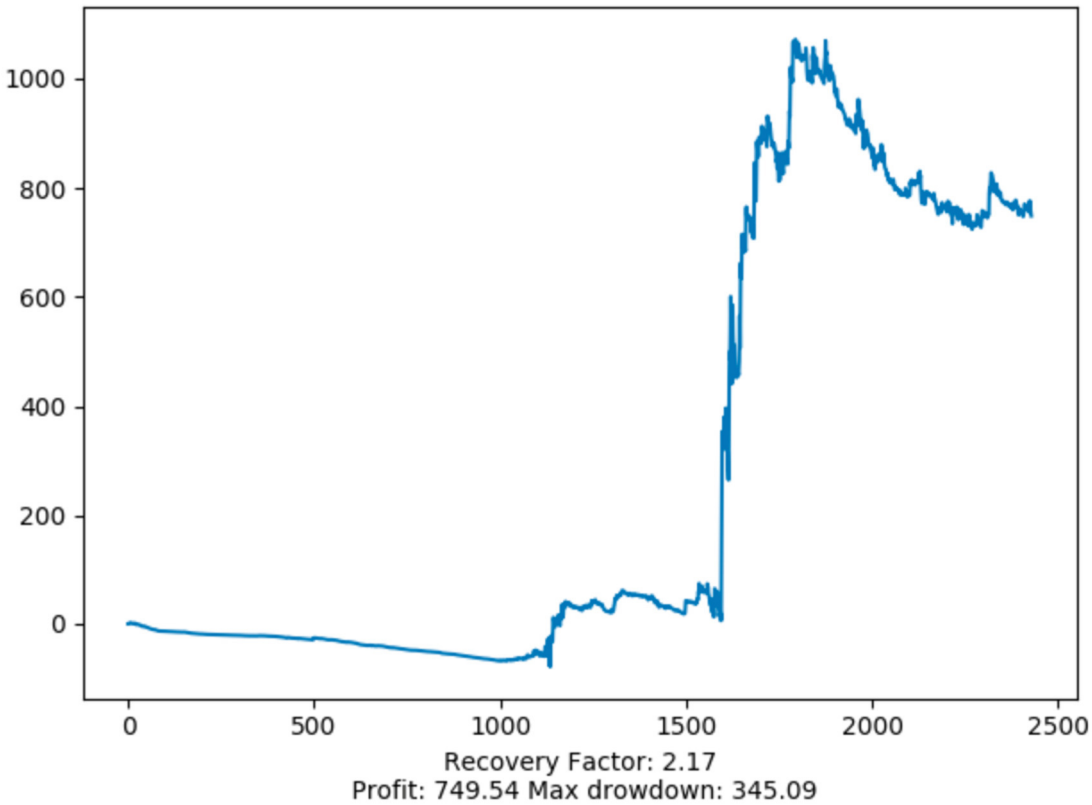

Рисунок 1.6 – Результаты тестирования переворотной стратегии. Таймфрейм 1 час

Как видно, выборка для анализа достаточно большая. Более 6 лет данных и 2500 сделок дают возможность справедливо оценить данную стратегию. На рисунке 1.7 представлена подробная статистика торговли.

```
| Recovery Factor: 2.17
| Profit: 749.54
| Max drowdown: 345.09
| Z-score: -2.8
| Avg. winning trade: 7.37 and Avg. losing trade: -2.18
| Probability to win in each trade: 25.65% (74.35% to lose)
| Ratio Avg. Win:Avg. Loss: 3.38
```

Рисунок 1.7 – Подробная статистика торговли

Анализируя полученные результаты можно сразу сказать, что это – трендовая торговая система. Отношение средней прибыльной сделке к среднему убытку составляет 3.38, что значит прибыльные сделки приносят в 3 раза больше дохода, чем убыточные. Однако, вероятность получить прибыльную сделку в среднем составляет 25.65%, что достаточно мало, поэтому принимая во внимание остальные фактора, использовать данную стратегию торговли в чистом виде не рекомендуется.

Рассмотрим иную торговую стратегию, также относящуюся к классическим: пересечение нескольких скользящих средних. На рисунке 1.8 представлен пример данной стратегии на графике.

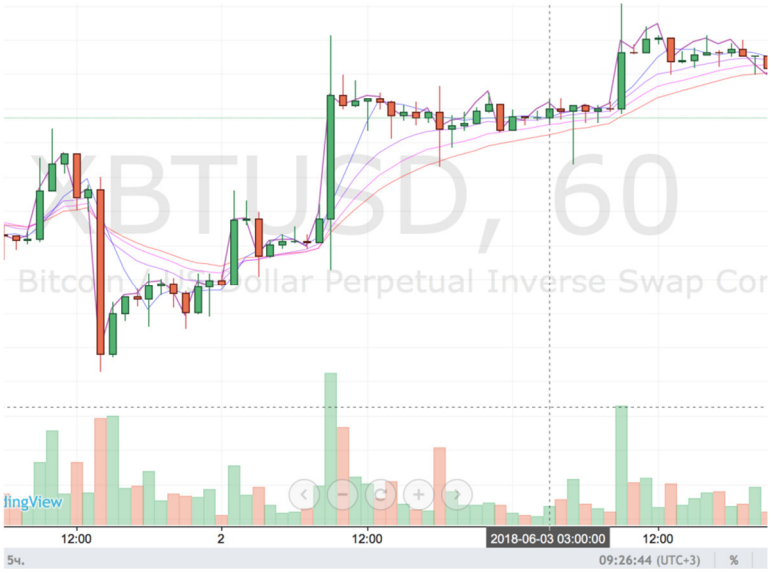

Рисунок 1.8 – Пример использования 4-х скользящих средних

Результаты тестирования изображены ниже на рисунках 1.9 и 1.10. Тестирование также было произведено на свечах с часовым периодом. Стоит отметить, что периоды для средних скользящих были оптимизированы и подобраны таким образом, чтобы максимизировать прибыльность.

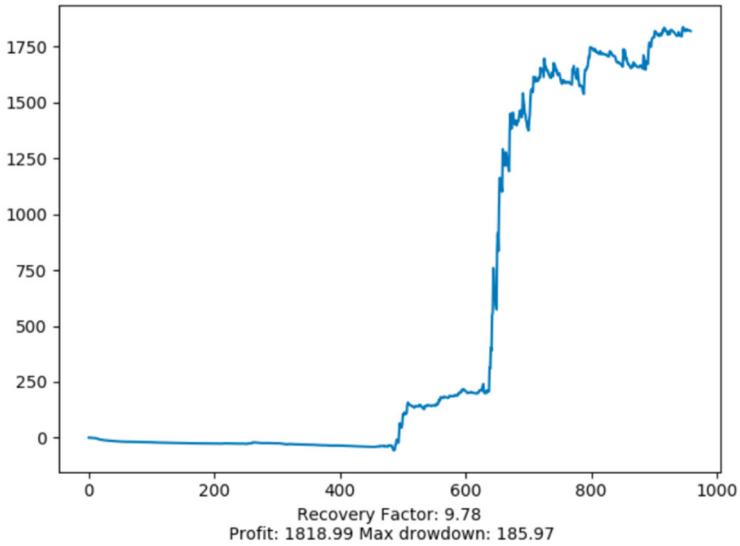

Рисунок 1.9 – Результаты тестирования

```
-------------------------------------------------------------
| Recovery Factor: 9.78
| Profit: 1818.99
| Max drowdown: 185.97
| Z-score: -1.75
| Avg. winning trade: 16.79 and Avg. losing trade: -3.22
| Probability to win in each trade: 25.44% (74.56% to lose)
| Ratio Avg. Win:Avg. Loss: 5.21
-------------------------------------------------------------
```

Рисунок 1.10 – Подробная статистика

Выводы по первому разделу.

Существует большое количество публичных стратегий. Многие из них цитируются в литературе по классическому анализу, некоторые пришли из современных наработок и в целом можно найти стратегию, которая показывала доход на исторических данных. Стоит отметить, что это всего лишь тестирование и торговля на реальном счете не гарантирует подобной доходности.

2 Проектирование автоматической системы торговли

2.1 Обзор существующих инструментов проектирования и разработки автоматических торговых систем

Рассмотрим существующие инструменты, предназначенные для тестирования торговых алгоритмов и создания автоматизированных механических торговых систем:

- инструменты, встроенные в терминал QUIK – Qpile и LUA. Встроенные языки программирования для реализации торговых алгоритмов в терминале QUIK [22]. Оба языка достаточно просты в освоении, что является большим плюсом. Тем не менее, оба языка имеют ограниченный функционал, сложность откладки программ и некоторые другие минусы. Как итог, данный вариант подходит для разработки простых торговых систем, не подразумевающих анализ больших данных и более сложные проектные решения;
- специализированное ПО от частных компаний – зачастую очень схожие по функционалу. В большинстве случаем имеют встроенную возможность тестирования стратегий на исторических данных, написание торговых систем на языках C# (и C подобных), интеграция с торговым терминалом;
- TradeMatic – система для создания торговых систем со встроенным механизмом тестирования на истории;
- TSLab – система аналогичная TradeMatic. Имеет визуальный редактор для проектирования торговой стратегии, расширенный функционал тестирования стратегий. На рисунке 2.1 представлен интерфейс программы, отображающий окно с результатами тестирования стратегии;
- LiveTrade RobotLab – терминал и визуальный конструктор роботов с возможностью написания кода самому;

– индивидуальное ПО. В данном случае язык программирования зависит от личных предпочтений разработчика. Некоторое время назад была популярна связка терминала QUIK и Excel с системой, реализованной на Visual Basic [23]. В настоящий момент большим спросом пользуется библиотека Stock# с открытым исходным кодом [24].

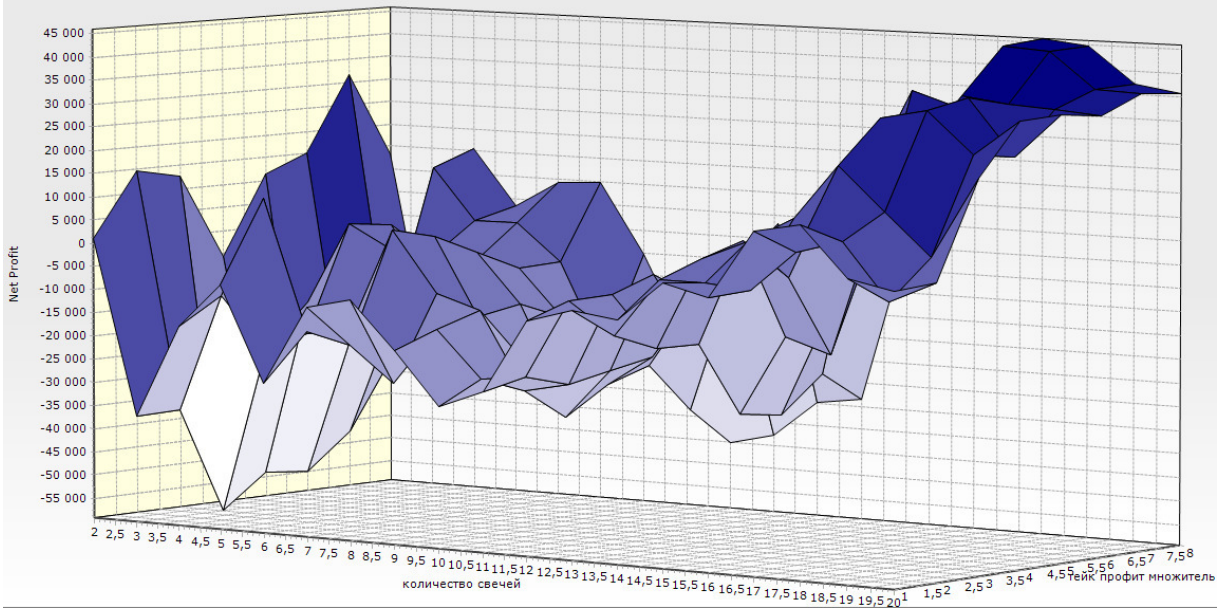

Рисунок 2.1 – Интерфейс программы TSLab с результатами тестирования стратегии

К сожалению, ни один из описанных выше инструментов не подходит для тестирования и разработки, рассматриваемой в данной работе, стратегии. Учитывая тонкость и специфику данного исследования было решено использовать индивидуальные решения, на базе языка Python, а также библиотек: CCXT, Ta-Lib, Pandas и MatplotLib.

2.2 Описание и проектирование торгового алгоритма

Одним из направлений алгоритмической торговли является высокочастотная торговля – метод торговли, при котором торговая система производит тысячи сделок в день. С развитием информационных технологий растет и внимание к подобному виду торговли. В настоящее время алгоритмическая торговля и HFT составляют большую часть от общего торгового объема в мире [25]. Основной целью этого исследования является проектирование и разработка торгового алгоритма использующего модель статистического арбитража. В качестве инструмента были выбраны два фьючерса на биткоин с разными сроками экспирации. Стратегия арбитража основана на получении прибыли за счет использования ценовых различий между ближним и дальним фьючерсными контрактами одного и того же базового актива. Открываются длинные и короткие позиции, когда спред между контрактами изменяется в надежде, что данное состояние является краткосрочным отклонением и цены сблизятся в ближайшем будущем.

Несмотря на большое количество обсуждений подобного метода торговли, в мире финансов, научных исследований доступных широкой публике критически мало. Причины данного феномена были описаны в предыдущих главах. Тем не менее, к минусам также можно добавить сложность реализации технологии и высокий порог входа. Подобный вид торговли предполагает чрезвычайно быстрое исполнение сделок, беспрерывный анализ рыночных данных, качественное принятие решений, мгновенный контроль сделок и т.д. Как было отмечено выше 70-80% ликвидности на всех современных мировых биржах [26] предоставляют торговые системы. HFT широко распространена на всех торговых секциях современных бирж, таких как: акции, валюты, фьючерсы, опционы. Помимо того, что подобные системы предоставляют ликвидность, они также используются как инструмент арбитража, что приводит к уравновешиванию цен на рынках и сохранению справедливых цен, избавляя рынок от разного

рода неэффективностей. Исходя из всего вышеперечисленного данный вид торговли (статистический арбитраж) был взят за основу в данной выпускной работе.

Первые упоминания о статистическом арбитраже появились в 1950 году [27]. Стратегия была привлекательна для хедж-фондов тех времен, так как несла в себе достаточно хорошую доходность с ограниченным риском. В портфелях крупных фондов и управляющих были одновременно открыты длинные и короткие позиции, что значительно уменьшало рыночные риски. Подобные стратегии построены на том, что доход по длинной позиции перекрывает убыток по короткой, либо наоборот. Таким образом позволяя извлекать практически безрисковую прибыль на долгосрочной основе.

Основой статистического арбитража является парный трейдинг [28]. Стратегия использует преимущества и неэффективности рынка на основе двух кореллирующих между собой активов, например валют. Для этого необходимо найти две валюты, которые движутся синхронно и одновременно занять две противоположные позиции, в моменты расхождения их спреда. Ожидается, что цены таких финансовых активов в ближайшем будущем вернутся к своему обычному состоянию. Очевидно, что разрабатываемая система должна использовать высококоррелированные активы, например фьючерсные контракты разных месяцев экспирации.

Рассмотрим методологию алгоритма более подробно. Алгоритм разрабатываемой торговой системы наблюдает за выбранными инструментами и ожидает, пока разница (спред) между ценами контрактов не отклонятся от своих средних значений. При возникновении подобной ситуации происходит открытие позиций. Сигналом выхода из сделки является возвращение спреда к исходному состоянию.

Фьючерсы выбираются с ближайшим сроком погашения, так как в них сосредоточена наибольшая ликвидность. Нормализация данных была выполнена следующим образом: для каждой цены контракта P (i, t) было

рассчитано эмпирическое среднее μ (i, t) и стандартное отклонение σ (i, t), а затем применено следующее уравнение (формула 1.1).

$$p(i,t) = \frac{P(i,t) - \mu(i,t)}{\sigma(i,t)} \tag{1.1}$$

Полученное p (i, t) является нормализованной ценой актива **i** в момент **t**. Нормализация необходима для реализации метода торговли парами. Подобный метод нормализации был реализован М.Перлином [29].

Торговая система рассмотренная М.Перлином, генерирует торговый сигнал при отклонении данный на заданный порог d. Данный параметр должен быть тщательно подобран, так как от него зависит частота и качество торговых входов. В данном исследовании брались цены покупок (бид) по более близкому контракту и цены продаж (аск) по дальнему. Таким образом, при увеличении расстояния до заданного d, торговая система открывает короткую позицию в размере одного контракта и длинную на другом. В ходе исследований было выявлено, что при увеличении порогового значения d результаты торговли системы были значительно лучше. На рисунке 2.3 изображены результаты тестирования данного алгоритма на истории.

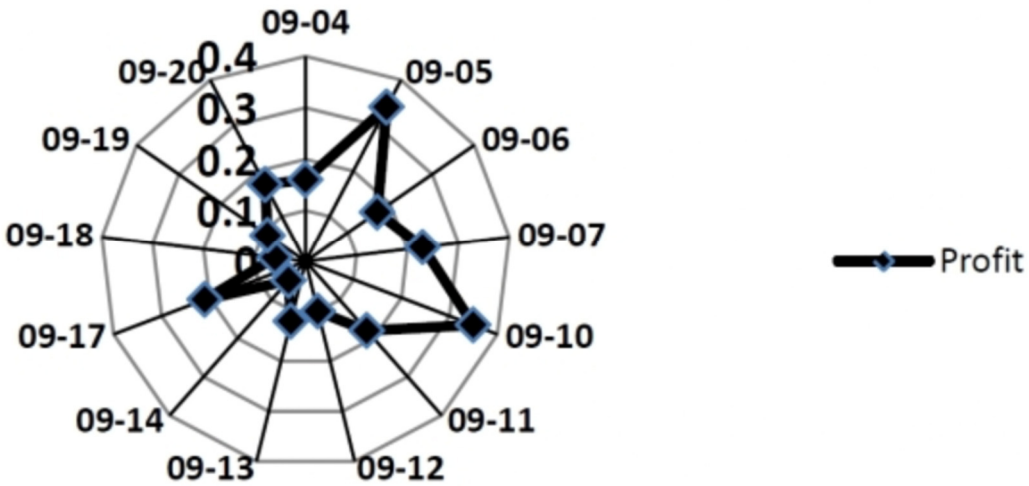

Рисунок 2.3 – Результаты тестирования алгоритма на истории

Чёрная линия показывает прибыль, а цифры 06-05, 06-06 и так далее – в это даты (5 июня, 6 июня). В выборку попали данные за 13 дней. Цифры 0.1, 0.2, 0.3, 0.4 — это прибыль в процентах. Самый удачный день принёс 0,4%, а самый неудачный 0,05%. В данном случае подсчитана чистая доходность, без учета комиссии биржи, брокера и других расходных средств. Очевидно, что с добавлением дополнительных издержек, проскальзывания и комиссий результаты торговли значительно ухудшатся. Подобные издержки обязательно должны быть учтены с запасом, при проектировании системы, т.к. в реальной торговле торговая может столкнуться с огромным количеством неопределенностей, в ходе которых можно понести незапланированный убыток [30].

Рассмотрим более простую модель. Принцип торговли остается тем же, однако точки входа будут определяться методом отклонения от среднего. Схема алгоритма представлена на рисунке 2.4.

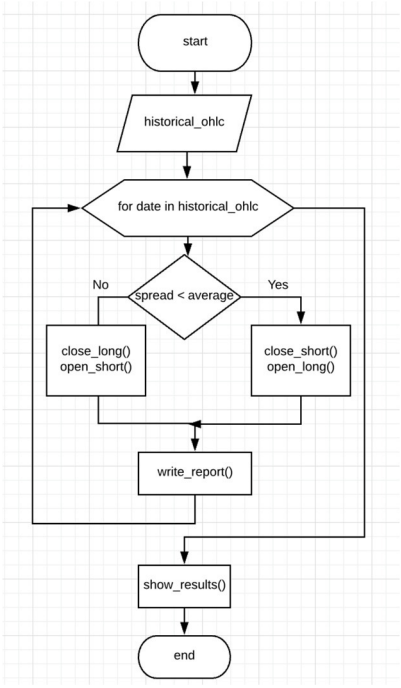

Рисунок 2.4 – Схема торгового алгоритма

Рассмотрим процесс тестирования более подробно. Прежде всего необходимо получить исторические котировки выбранных инструментов. На

рисунке 2.5 изображена часть кода, позволяющего скачать данные с биржи и рассчитать спред на основе полученных данных.

```
bitmex = ccxt.bitmex()

since = (int(time.time()) - 95000) * 1000
xbtu = pd.DataFrame(bitmex.fetch_ohlcv('XBTU18', '1m', since=since, limit=500),
                    columns=['timestamp', 'open', 'high', 'low', 'close', 'volume'])

xbtm = pd.DataFrame(bitmex.fetch_ohlcv('XBTM18', '1m', since=since, limit=500),
                    columns=['timestamp', 'open', 'high', 'low', 'close', 'volume'])

result = xbtu.copy()
result.open = xbtu['open'] - xbtm['open']
result.high = xbtu['high'] - xbtm['high']
result.low = xbtu['low'] - xbtm['low']
result.close = xbtu['close'] - xbtm['close']
```

Рисунок 2.5 – Листинг программы

Далее, исходя из полученных данных о ценах и разнице между инструментами необходимо сформировать цикл, моделирующий работу алгоритма на истории, рассчитать средний спред (рисунок 2.6), и его с историческими значениями. В случае если выполняется условие открытия или закрытия позиции информация о сделке записывается в отчет, для дальнейшего анализа.

```
_open = np.array(quotes['open'].values, dtype=float)
_high = np.array(quotes['high'].values, dtype=float)
_low = np.array(quotes['low'].values, dtype=float)
_close = np.array(quotes['close'].values, dtype=float)
ma = talib.MA(_open, 20)
```

Рисунок 2.6 – Расчет среднего спреда между двумя фьючерсами

Результаты тестирования алгоритма отображены на рисунке 2.7 – кривая доходности.

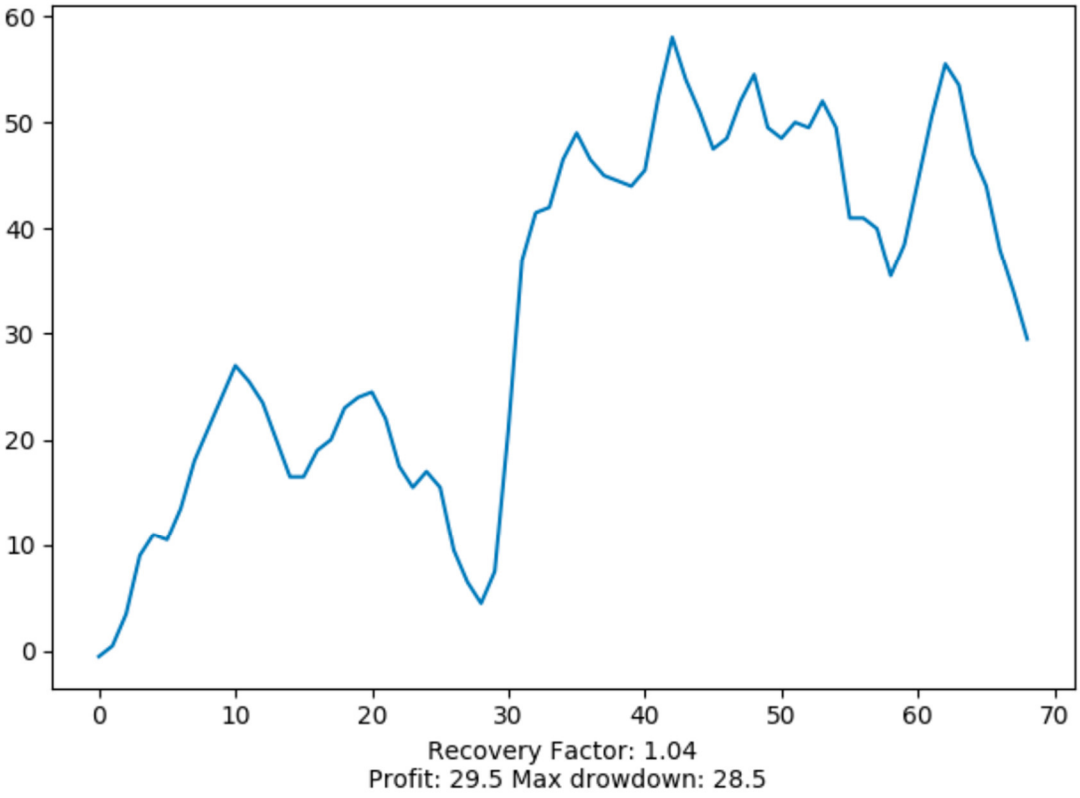

Рисунок 2.7 – Кривая доходности

Изменим параметры скользящей средней в попытке оптимизировать результат (рисунок 2.8).

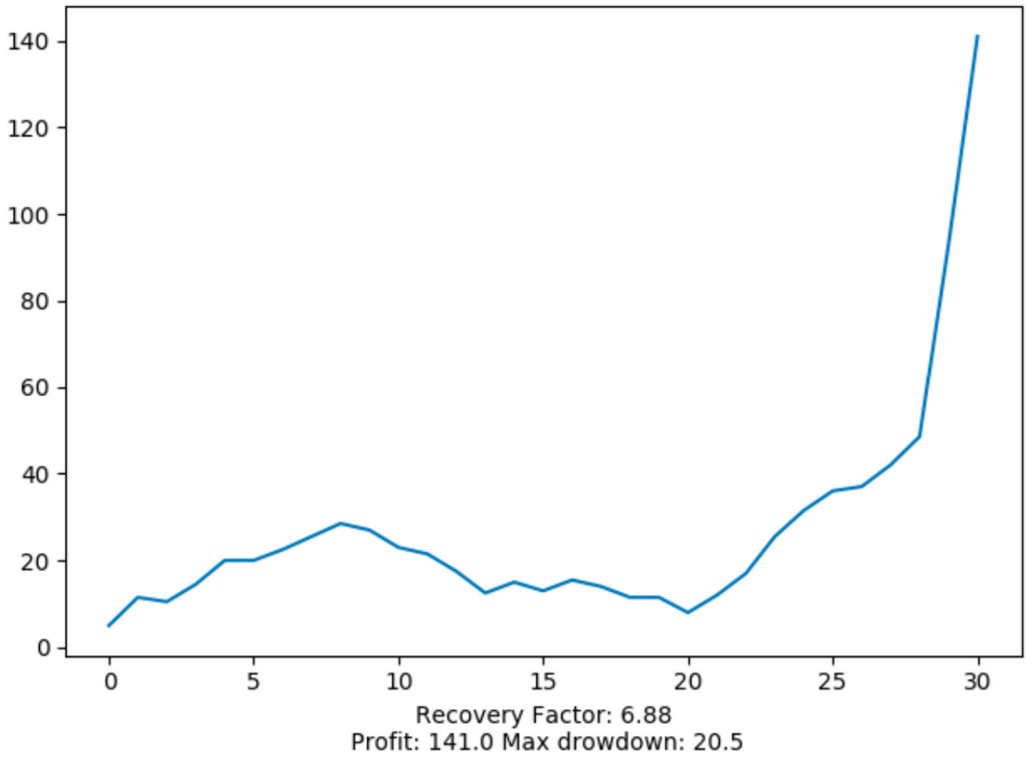

Рисунок 2.7 – Кривая доходности после оптимизации

В ходе тестирования была использована выборка данных длиной в 12 часов. В результате было получено 30 сделок, чистая прибыль 141$ (рисунок 2.9), при стартовом капитале в 2000$.

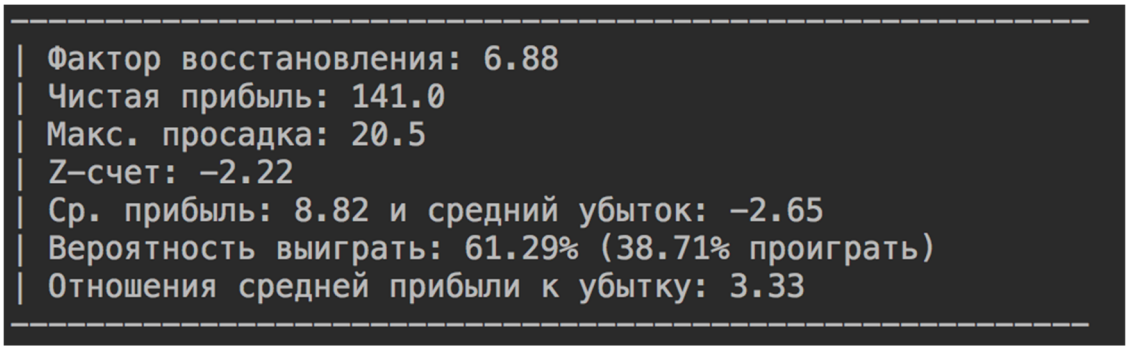

Рисунок 2.9 – Подробная статистика результатов тестирования

Стоит отметить, что алгоритм был протестирован на достаточно маленьком отрезке времени. Зачастую этого недостаточно, чтобы сделать корректные выводы. Тем не менее, протестировав алгоритм на большем количестве исторических данных, были получены похожие результаты, что только доказывает гипотезу статистического арбитража.

Рассмотрим подробнее процесс оптимизации данной торговой стратегии. Зачастую оптимизация проводится на заключительном этапе тестирования алгоритма, с целью улучшить уже имеющиеся результаты. Существует большое количество методов оптимизации: от простого перебора до генетических алгоритмов. На рисунке 2.10 представлены результаты оптимизации периода для скользящей средней, простым перебором, по параметру фактор восстановления.

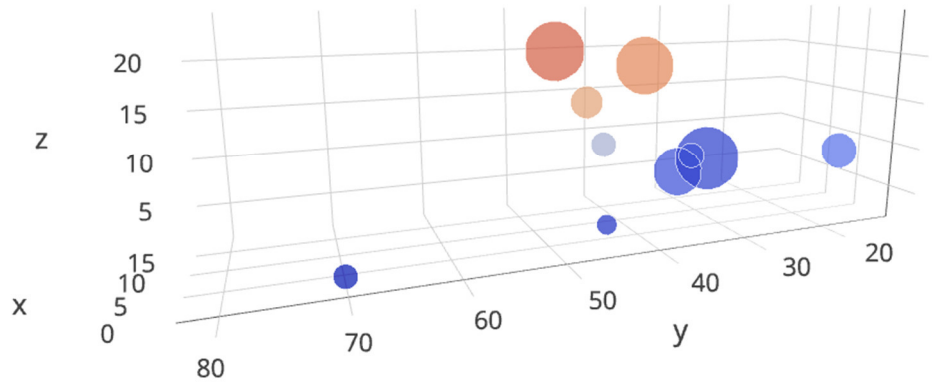

Рисунок 2.10 – Результаты оптимизации периода, скользящей средней

Как видно на рисунке, стратегия показывает максимальную прибыль с периодом скользящей средней в 20 минут. Протестируем стратегию еще раз, чтобы оценить результаты. Результаты тестирования показаны на рисунках 2.11, 2.12.

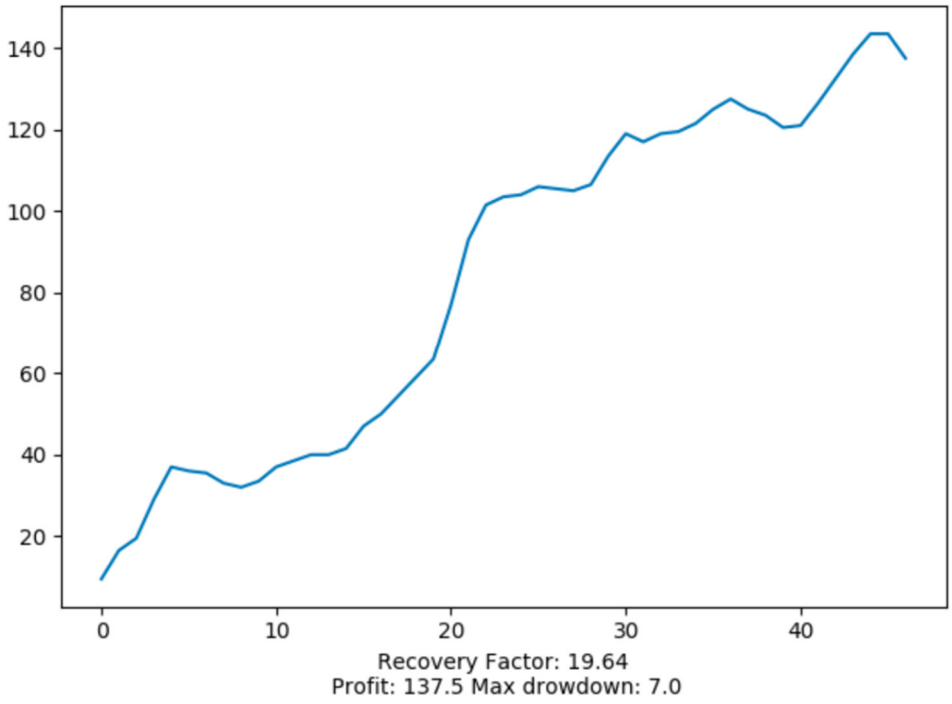

Рисунок 2.11 – Результаты тестирования после оптимизации

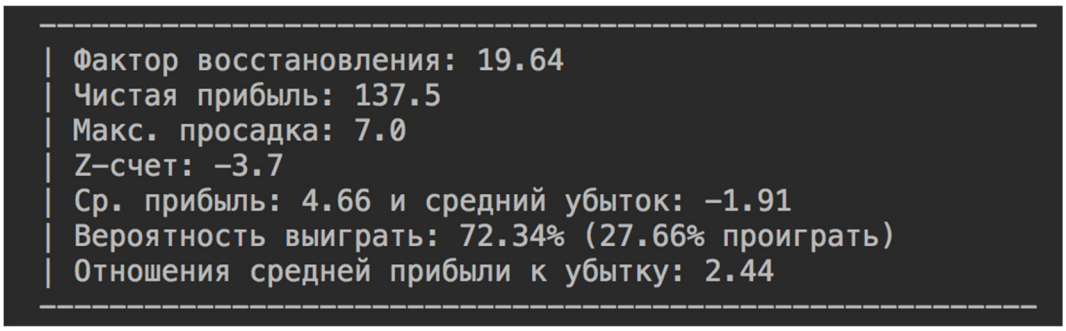

Рисунок 2.12 – Подробная статистика после оптимизации

В данном пункте был описан, спроектирован и протестирован на исторических данных алгоритм будущей автоматической системы торговли. Также система была оптимизирована с целью увеличения доходности и уменьшению возможного риска потери капитала.

2.3 Диверсификация торговой стратегии

Зачастую на этапе проектирования торговых систем разработчики акцентируют внимание на прибыльности системы. Однако, в погоне за прибыльностью трейдеры часто забывают о сокращении возможных рисков, выраженных в максимальной просадке. Простой, но сравнительно надежный метод оценки эффективности торговли – расчет отношения доходности к максимальной просадке на тестируемой выборке. Например, если в результате тестирования система показывает результат чистой прибыли в 45% годовых, а максимальная просадка составляет 15%, фактор восстановления будет равен 3. Таким образом, при сравнении двух различных систем с разными показателями просадки и доходности, логично предположить, что наиболее привлекательной будет та, у которой выше фактор восстановления. Торговая система, дающая 30% годовых и с просадкой в 5%, представляет собой больший интерес по сравнению с системой, дающей 100% годовых и просадкой в 40%. Трейдеру стоит понимать, что желаемую доходность легко можно получить, используя так называемое плечо (заемные средства, маржинальная торговля), а вот долю

риска при этом остается неизменным фактором, на который нельзя повлиять. Эта та часть рыночных рисков, к которым трейдер может только присмотреться, но не взять под контроль. Стоит помнить также, что, увеличивая доходность с помощью заемных средств, соответственно увеличивается и риск.

Однако, существуют способы сократить риск в целом по портфелю из нескольких стратегий. Речь идет о диверсификации. То есть, если сформировать портфель, состоящий не из одной стратегии, а из нескольких и разделив капитал между системами, то риск соответственно тоже будет распределен. В таком случае, убыток каждой отдельной стратегии не обаятельно будет приводить к общей просадке в портфеле, так как полученный убыток может быть скомпенсирован доходом от другой системы. В то же время изменится и общая доходность. В лучшем случае в результате получится усредненная доходность всех стратегий в портфеле. Логично предположить, что если торговые алгоритмы в достаточной мере независимы друг от друга (торгуются разные инструменты с обратной корреляцией, используются точки входа в разные состояния рынка и т.д.), то падение прибыльности в одной из систем скорее всего будет компенсировано ростом другой.

Выводы по второму разделу.

Чем более независимы стратегии и инструменты, тем сильнее размывается максимальная просадка. Возможны даже ситуации, когда имеет смысл трейдеру добавить в портфель заведомо убыточную стратегию. Теоретически, если сформировать портфель из чрезвычайно большого количества стратегий разного рода, то можно получить сколь угодно малый риск, и, соответственно, сколь угодно большую эффективность. Однако, на практике такое намерение неизбежно столкнется с проблемой корреляции между различными стратегиями и инструментами.

3 Программная реализация

3.1 Разработка автоматической торговой системы

В данной работе, в качестве основного языка программирования был использован Python. Все вычисления, взаимодействие с биржей через API, расчет индикаторов, веб приложение и т.д. были выполнен с помощью данного языка программирования и дополнительных библиотек, позволяющих все это реализовать. Python – это язык программирования с широким спектром возможностей. Философия данного языка – простота чтения и написания кода, скорость разработки, легкий подход. Идеально подходит для быстрого создания прототипов и написания небольших программ. Учитывая все перечисленные преимущества можно с уверенностью сказать, что порог вхождения достаточно низкий, а результирующий код во многом лаконичный и понятный даже не опытным программистам. За счёт простоты кода, дальнейшее сопровождение программ, написанных на Python, становится легче и приятнее по сравнению с Java или C#. С точки зрения крупных компаний это влечёт за собой сокращение расходов и увеличение производительности труда разработчиков. Для демонстрации лаконичности на рисунке 3.1 представлен листинг двух программ языках Python и Java. Показанный ниже код открывает файл и считывает его содержимое в переменную.

Python:

```python
file = open('file.txt')
content = file.read()
```

Java:

```java
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
public static void main(String[] args) throws IOException {
    String content = new String(Files.readAllBytes(Paths.get("file.txt")));
}
```

Рисунок 3.1 – Сравнение лаконичности кода Python и Java

Подобная простота – не единичный случай. В целом, многие функции, написанные на Python, занимают значительно меньше места. Стоит также отметить растущее с каждым днем сообщество разработчиков. Сообщество вокруг Python одно из самых сильных в мире IT и представляет из себя сложный, но организованный организм. Помимо сотни тысяч индивидуальных разработчиков и небольших компаний, Python предпочитают использовать такие крупные игроки как: Google, Mozilla, Yandex, Facebook и даже Microsoft. Youtube, первые версии Google, Dropbox, Reddit, Instagram и многие другие проекты были реализованы на данном языке. Все это говорит о том, что действительно большие компании не бояться использовать данный инструмент в качестве основного языка разработки.

Однако, стоит помнить и о значительных минусах: скорость исполнения и динамическая типизация становится критическим для многих разработчиков. Несмотря на то, что для многих программистов динамическая типизация только на руку (начинающие разработчики особенно любят этот факт), с ростом кодовой базы следить за типом переменных становится очень сложно. В случаях, когда код разрастается до больших масштабов, переменные названы отлично от стандарта, а архитектура приложения построена не должным образом – динамическая типизация превращается в большой минус. Конечно, существуют различного рода обходные пути данной проблемы, например типовые аннотации, mypy и статический анализ кода. В связи с ограничениями языка, появляются альтернативные реализации интерпретаторов: PyPy, Pyston, Jython, Cython и многие другие. Но в конечном итоге, все это только лишь попытки решить проблему, которые к тому же портят читаемость кода.

Как было отмечено выше, вторым значительным минусом языка является скорость работы. Пожалуй, это самый главный аргумент со стороны разработчиков, которые недолюбливают Python. Относительно низкая скорость выполнения, динамическая типизация и запрет на ручное

управление памятью также входят в число их доводов против. Первые версии Google были написаны на Python, однако при росте популярности сервиса и соответственно появлении огромных нагрузок на сервера компании пришлось переосмыслить подход и переписать весь код на чистый C++. Действительно, скорость работы программ написанных на Python не может сравниться с C подобными языками. Впрочем, если сравнить язык с конкурентами: Ruby, JavaScript, PHP, то все на их фоне Python показывает отличные результаты. Так как целью данной работы является разработка автоматической торговой системы, которая должна совершать сделки с очень большой частотой, скорость выполнения результирующего кода должна быть на очень высоком уровне. Однако, даже несмотря на это, взвесив все преимущества и недостатки, в качестве основного инструмента разработки, был выбран именно Python, с поправкой на то, что в некоторых местах могут использоваться оптимизированные фрагменты кода написанные на C.

Стоит отметить, что в качестве среды разработки было использовано программное обеспечение от компании JetBrains под названием PyCharm. PyCharm – коммерческий продукт, однако к использованию доступна Community Edition версия, распространяемая бесплатно. По функциональности данное ПО намного превосходит существующие на рынке аналоги. Из видимых преимуществ моно отметить: встроенную мощную систему отладки кода с поддержкой трассировки кода, достаточно серьезные инструменты для юнит-тестирования, анализатор кода, поддержка популярных VCS сервисов и многое другое. В ходе исследования одного из популярных сообществ программистов в интернете, был проведен опрос, в результате которого были выявлены наиболее используемые среды разработки. Результаты представлены на рисунке 3.2.

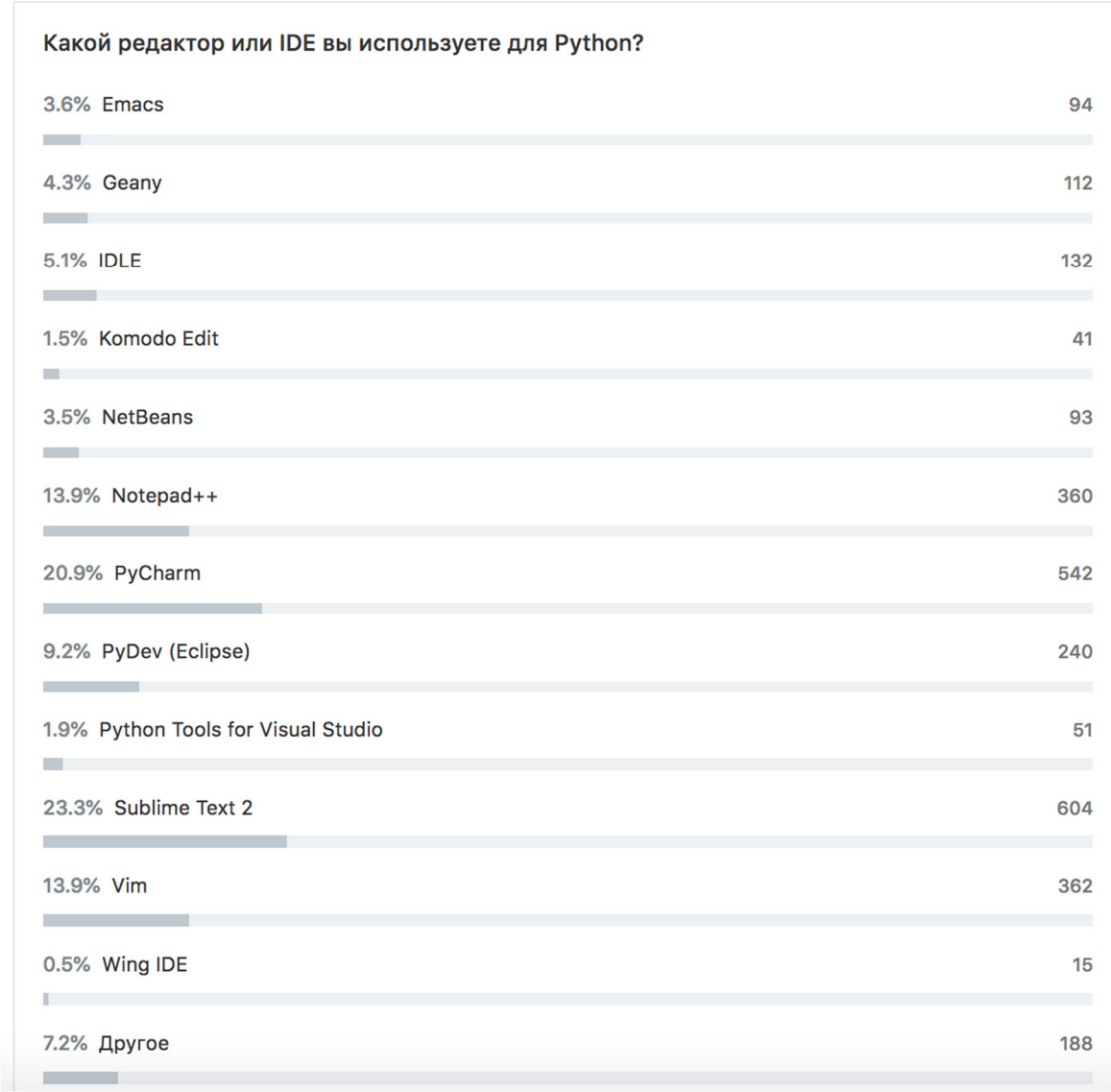

Рисунок 3.2 – Наиболее используемые среды разработки

Как видно на рисунке, сообщество русскоязычных разработчиков действительно предпочитает использовать PyCharm в качестве основного редактора кода. На втором месте находится редактор Sublime Text второй версии. Рассмотрим данный список программного обеспечения с точки зрения поддерживаемых платформ (рисунок 3.3).

IDE	Лицензия	Windows	Linux	macOS	Другие платформы
PyDevelop IDE	GPL	Да	Нет	Нет	
Boa Constructor	GPL	Да	Да	Да	
Eric	GPL	Да	Да	Да	
Geany	GPL	Да	Да	Да	
IDLE	PSFL	Да	Да	Да	
ActiveState Komodo	MPL, GPL, LGPL	Да	Да	Да	
NetBeans	CDDL, GPL, LGPL	Да	Да	Да	OpenBSD, Solaris
Ninja-IDE	GPL	Да	Да	Да	
PyCharm	ASL	Да	Да	Да	
PyDev	EPL	Да	Да	Да	
PyScripter	MIT	Да	Нет	Нет	
Stani's Python Editor	GPL	Да	Да	Да	
Spyder	MIT	Да	Да	Да	
Eclipse + PyDEV	EPL	Да	Да	Да	
Microsoft Visual Studio + Python Tools	Проприетарная, ASL	Да	Нет	Нет	
Pyzo[4]	Лицензия BSD	Да	Да	Да	

Рисунок 3.3 – Список поддерживаемых платформ средами разработки

Исходя из всего вышеперечисленного, в качестве основной среди разработки был выбран редактор PyCharm.

3.2 Разработка автоматической торговой системы

В данном пункте рассмотрен вопрос программной реализации автоматической торговой системы, работающей на основе алгоритма, спроектированного ранее. Помимо самой торговой системы предусмотрена разработка личного кабинета, где пользователь может отслеживать результаты торговли, следить за состоянием счета, сделками и т.д. Чтобы войти в систему, и получить доступ к личным данным, пользователю необходимо ввести пароль авторизации. После сравнения данных с данными базы, пользователь либо входит в систему, либо получает сообщение о том, что введенные данные не верны. Не зарегистрированным пользователям предлагается создать аккаунт.

Рассмотрим архитектуру разрабатываемого программного обеспечения подробнее. Приложение построено по принципу MVC (модель, представление, контроллер). При проектировании был также использован паттерн Singleton. В результате были получены следующие компоненты:

- AlgorithmManager – класс, ответственный за управление экземплярами. Данный класс обеспечивает корректную работу алгоритма с остальными компонентами приложения, также здесь происходит контроль за графиками и состоянием торговой системы на сервере;
- AlgorithmLogManager – данный компонент отвечает за запись логов и ведение отчетов, для дальнейшего анализа работоспособности системы;
- AlgorithmStrategy – этот класс отвечает за модели прогнозирования и представляет из себя непосредственно торговый алгоритм.

Все компоненты связаны между собой и в целом представляют архитектуру автоматической торговой системы. На рисунке 3.4 представлена UML схема результирующей архитектуры.

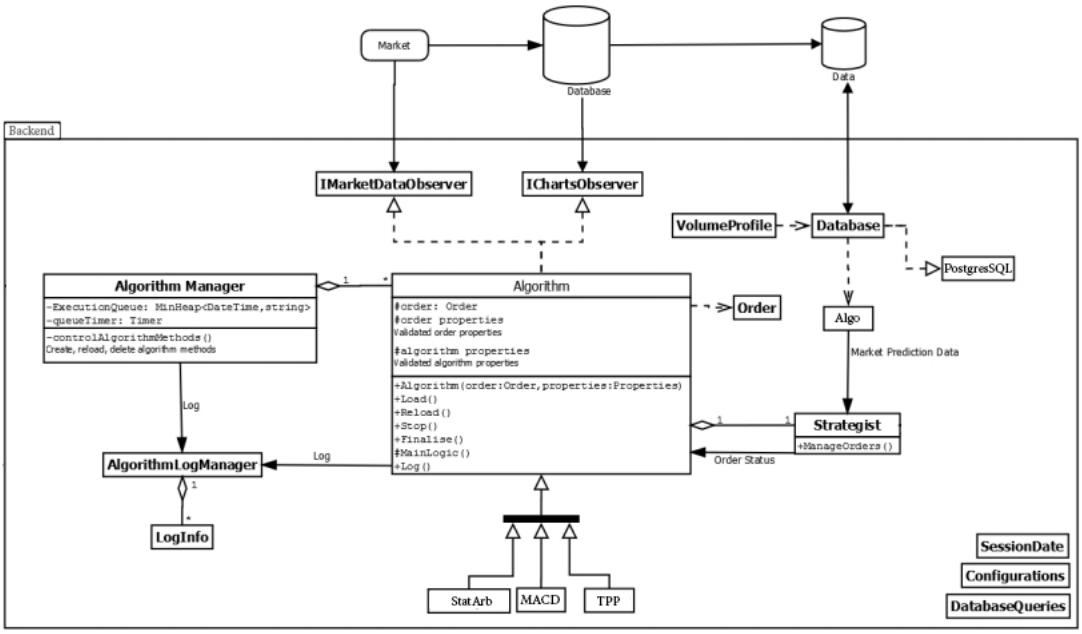

Рисунок 3.4 – UML схема приложения

Система спроектирована таким образом, чтобы при необходимости подключить различные торговые стратегии, не изменяя остальные компоненты системы. На представленной выше схеме система наследует от главного класса три различные стратегии с разной торговой логикой: StatArb, MACD и TPP (True Price Prediction). Рассмотрим основной алгоритм торговли, разрабатываемый в ходе данной работы – StatArb. Это алгоритм статистического арбитража, полученный во второй главе. Стратегия заключается в одновременной покупке и продаже ближнего и дальнего соответственно, фьючерсного контракта с разным сроком экспирации. Точкой входа является ситуация, при которой спред между фьючерсами отклоняется от некого среднего значения, заданного периодом скользящей средней.

Логично, что приложение не может напрямую подключиться к внешней базе данных биржи для получения информации о ценах на финансовые активы. Подключение к бирже происходит посредством REST API и все полученные данные сохраняются в локальной базе данных. Для отслеживания исполнения ордеров и работы системы в целом была добавлена система ведения журнала событий. Данная система не только записывает события торговой системы, но также контролирует их, проверяя работоспособность всей системы каждые 30 секунд.

Архитектура базы данных веб-приложения изображена на рисунке 3.5.

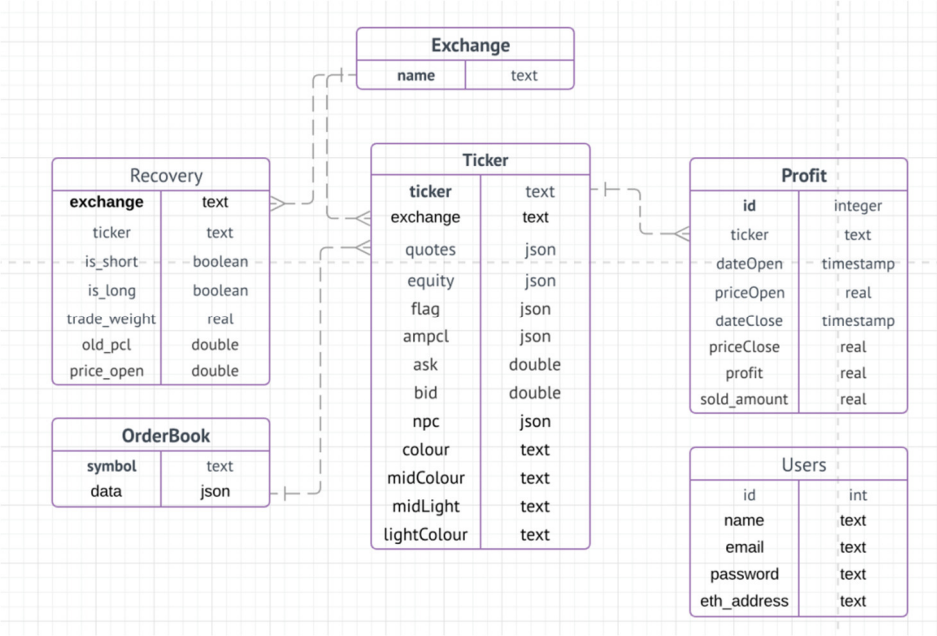

Рисунок 3.5 – Схема базы данных веб-приложения

На рисунке 3.6 отображен базовый интерфейс веб-приложения.

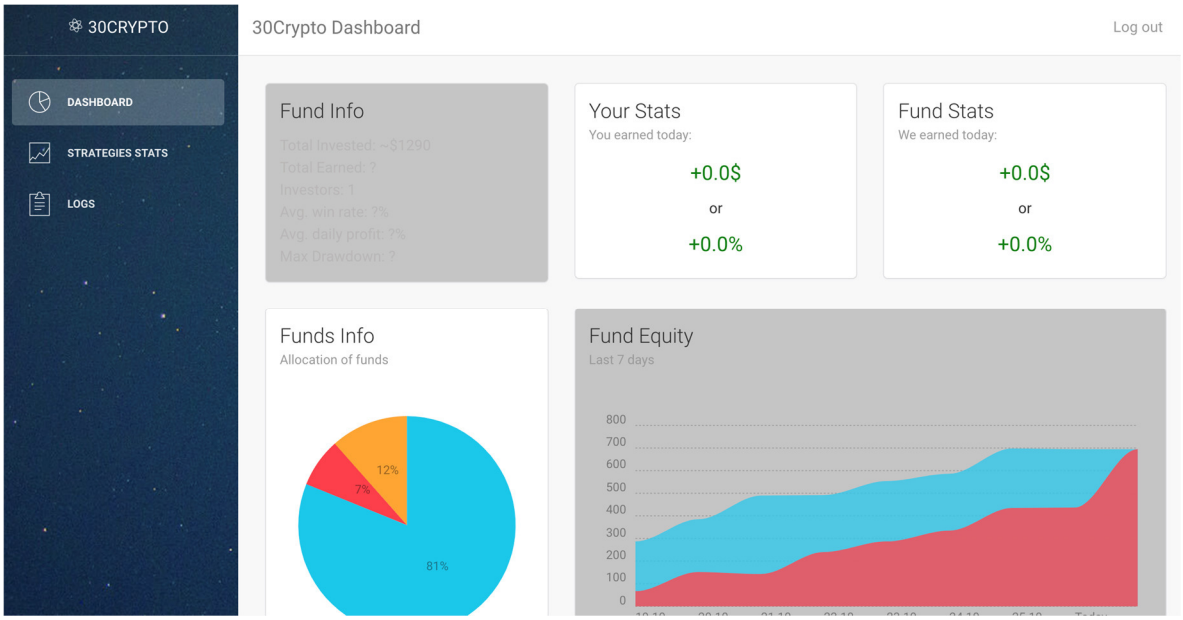

Рисунок 3.6 – Главная страница

В случае, если торговая стратегия предусматривает базовую настройку параметров алгоритма, пользователю предлагается ввести значения переменных вручную (рисунок 3.7).

Рисунок 3.7 – Страница настроек торгового алгоритма

Для удобства пользователя на сайте отображается график с исторической ценой выбранного инструмента (рисунок 3.8).

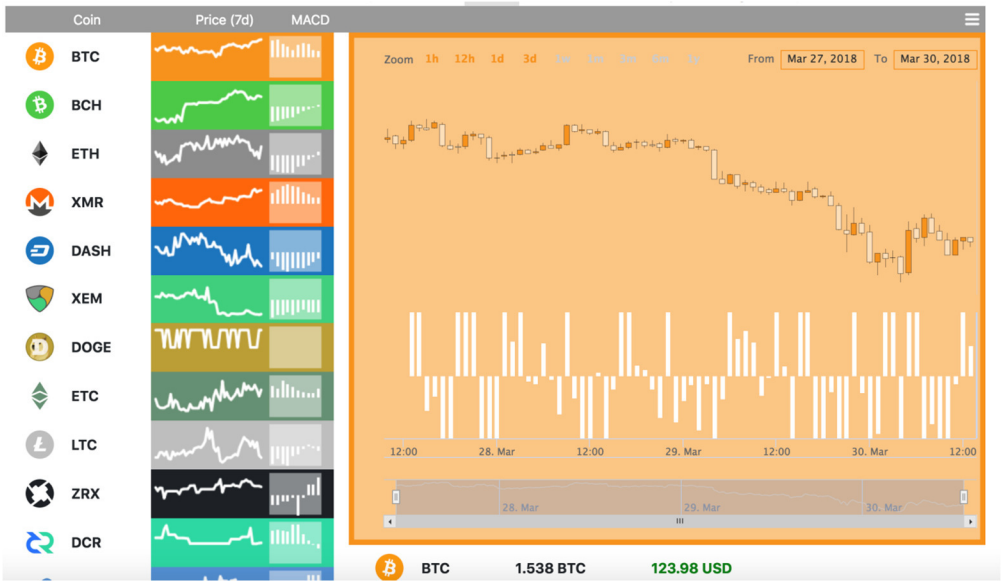

Рисунок 3.8 – Отображение графика цены выбранного инструмента

Выводы по третьему разделу.

Была выполнена задача по разработке автоматической системы торговли на валютном рынке, а также веб-приложение, с возможностью просмотра результатов торговли. Все функции соответствуют описанным.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной выпускной квалификационной работы была достигнута поставленная цель оптимизации процесса торговли на валютном рынке. Все поставленные для решения цели задачи были успешно выполнены. Проведен анализ рыночных неэффективностей и публичных торговых стратегий. Спроектирована и разработана автоматическая система торговли на валютном рынке. Также разработано веб-приложение, позволяющее получить доступ к журналу сделок, торговым результатам, возможность посмотреть текущее состояние на рынке валют, проанализировать эффективность торговли с помощью автоматической системы. Также была описана функциональная схема предметной области, спроектированные логическая и физическая схемы модели базы данных.

Из результата проведенных работ можно сделать вывод, что разработка данного программного обеспечения является актуальной, так как спрос на продукт достаточно велик. Исследования подтверждают, что с каждым годом количество клиентов на Московской бирже стремительно растет, что только доказывает актуальность данных исследований. Главное отличие данного проекта от уже существующих вариантов – это совмещение в себе удобного веб интерфейса и алгоритмической торговли, что является очень удобным для пользователей, не владеющих в достаточной мере навыками работы в консоли.

Информационная система позволяет структурировать и автоматизировать процесс торговли на валютном рынке, а также снять с трейдера ответственность за принятые торговые решения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Лефевр Э. Воспоминания биржевого спекулянта. – Olympus Business, 2012.
2. Faith C. Way of the Turtle: The Secret Methods that Turned Ordinary People into Legendary Traders. – McGraw Hill Professional, 2007.
3. Lehoczky J., Schervish M. Overview and History of Statistics for Equity Markets //Annual Review of Statistics and Its Application. – 2018. – Т. 5. – №. 1.
4. Викулин А. В. Экономико-математические методы в обосновании ожидаемой доходности и риска отдельных акций и портфеля из них //новейшие достижения и успехи развития экономики. – 2018. – С. 37.
5. Лоран Ж. Опасные игры с деривативами: Полувековая история провалов от Citibank до Barings, Societe Generale и AIG. – Альпина Паблишер, 2018.
6. Саркисов В. Г., Парамонов Р. А. Контртрендовая стратегия на рынке сверхкраткосрочных бинарных опционов //Проблемы экономики и менеджмента. – 2014. – №. 12 (40).
7. Weissman R. L. Mechanical trading systems //Hoboken: Wiley. – 2005.
8. LeBeau C., Lucas D. W. Computer analysis of the futures market. – McGraw-Hill, 1992.
9. Burgess A. N. et al. A computational methodology for modelling the dynamics of statistical arbitrage : дис. – University of London, 2000.
10. Broussard J. P., Vaihekoski M. Profitability of pairs trading strategy in an illiquid market with multiple share classes //Journal of International Financial Markets, Institutions and Money. – 2012. – Т. 22. – №. 5. – С. 1188-1201.

11. Рубцов Б. Б. и др. Мировой и российский рынки IPO: анализ тенденций и перспектив развития //Эффективное антикризисное управление. – 2011. – №. 5.

12. Володин С. Н., Коченков И. А. Статистический арбитраж на российском фондовом рынке //Аудит и финансовый анализ. – 2013. – №. 6. – С. 237-244.

13. Мазаев Н. Ю. Использование фундаментального анализа в управление портфелем ценных бумаг //Инновации и инвестиции. – 2015. – №. 1. – С. 34-37.

14. Паттерсон С. Кванты. Как волшебники от математики заработали миллиарды и чуть не обрушили фондовый рынок //М.: Манн, Иванов и Фербер. – 2014.

15. Lin V. Trading Strategy, Cost Efficiencies in FX Algo Trading //Market Commentary, Portfolio Strategy, Credit Suisse, AES. – 2010.

16. Arnuk S. L., Saluzzi J. Toxic equity trading order flow on wall street //Themis Trading LLC White Paper, http://www. themistrading. com/article_files/0000/0524/Toxic_Equity_Trading_on_Wall_Street_-- _FINAL_2__12. – 2008. – Т. 17.

17. Sarika K. B., Sreekumar R., Shilja M. S. An Analytical Approach for Algotrading.

18. Kia F. Developing a fully automated algo-trading system. – 2016.

19. Fecht F., Hackethal A., Karabulut Y. Is proprietary trading detrimental to retail investors? //The Journal of Finance. – 2018. – Т. 73. – №. 3. – С. 1323-1361.

20. Yamamoto A. Cryptocurrency Bible-vol 2: Includes 3 Cryptocurrency Books-Bitcoin Hacking? Bitcoin Why Not to Invest? Cryptocurrency Trading & Investing (Volume 2). – 2018.

21. Glodjo A., Bronson N. D., Harrington S. E. Trading system with individualized order books : заяв. пат. 14846230 США. – 2015.

22. Xing X., Xue Y. Trading mechanisms and market quality: Limit-order books versus dealership markets //Economics Letters. – 2017. – Т. 154. – С. 35-44.

23. Ha Y. Algorithmic Trading in Limit Order Books for Online Portfolio Selection. – 2018.

24. Bouricius P., Dyer M., Robinson L. Molasses and Marshmallow: Food and Trading in New England Account Books. – 2015.

25. Майнер Р. Торговые стратегии с высокой вероятностью успеха: тактики входа и выхода на рынках акций, фьючерсов и валют. – Альпина Паблишер, 2018.

26. Чеботарев Ю. Торговые роботы на российском фондовом рынке. – Litres, 2017.

27. Крюков П. А. Теоретические основы совершенствования финансового механизма ведения торговых операций на валютном рынке Forex //Вестник Кузбасского государственного технического университета. – 2014. – №. 4 (104).

28. Семенкова Е. Операции с ценными бумагами. – Litres, 2017.

29. Ананченко И. В., Мусаев А. А. Торговые роботы и управление в хаотических средах: обзор и критический анализ //Труды СПИИРАН. – 2014. – Т. 3. – №. 34. – С. 178-203.

30. Белова Е., Окороков Д. Технический анализ финансовых рынков. Учебное пособие. – Litres, 2017.

ПРИЛОЖЕНИЕ А

Листинг основного класса.

```
import logging
import sys
import time
import logic.config as config
from database.tables import Recovery as db
from logic.Strategy_XBTUSD import Strategy as Strategy
XBT_USD = None
def init():
    # ### Setup logging settings
    if config.SAVE_LOGS is False:
        logging.FileHandler('logs.log', mode='w')  # remove previous logs

    logging.basicConfig(filename='logs.log', level=logging.INFO, format='%(asctime)s :>>>  %(message)s',
                        datefmt='%d/%m/%Y %H:%M:%S')
    log = logging.getLogger(__name__)
    fh = logging.FileHandler('logs.log')
    fh.setLevel(logging.INFO)
    sh = logging.StreamHandler(sys.stdout)
    sh.setLevel(logging.INFO)
    log.addHandler(sh)
    log.addHandler(fh)

    logs = 'Bot launched'
    logging.info(logs)
    print(logs)

    # ### Initialize trading symbols
    global XBT_USD

    recovery = db.read()

    if config.IS_RECOVERY_XBT_USD:
        # start recovery system
        symbol = 'XBTUSD'
        count_buy = recovery[symbol]['count_buy']
        opened_buys = recovery[symbol]['opened_buys']
        bars_counter = recovery[symbol]['bars_counter']
        sum_amount = recovery[symbol]['sum_amount']
```

```
                range_bars = recovery[symbol]['range_bars']
                price_position = recovery[symbol]['price_position']
                opened_short = recovery[symbol]['opened_short']
                buys_total = recovery[symbol]['buys_total']

                XBT_USD = Strategy(symbol, count_buy, opened_buys, bars_counter,
                                   sum_amount, range_bars, price_position, buys_total, opened_short)

                logs = 'Strategy XBT_USD recovered with parameters:\n\t' \
                    'Count Buys: {};\n\tInTrade: {};\n\t Price Position: {}'.format(count_buy, sum_amount,
price_position)
                logging.info(logs)
                print(logs)

            else:
                XBT_USD = Strategy('XBTUSD')
            logs = 'Strategy XBT_USD initialized'
            logging.info(logs)
            print(logs)

            # check if IS_DEBUG == False --> bot will trade for real money (!)
            logs = 'DEBUG mode enabled'
            if config.IS_DEBUG is False:
                logs = 'DEBUG mode disabled. Bot will trade for real money!'
            logging.info(logs)
            print(logs)

            # report = open('report.csv', 'w+')
            # report.write('symbol;dateOpen;priceOpen;reason;dateClose;priceClose;profit;direction\n')
            # report.close()

            # end of init

            logs = '===== Start trading ====='
            logging.info(logs)
            print(logs)

        class Strategy:
            def __init__(self, symbol, count_buy=0, opened_buys=False, bars_counter=0, sum_amount=0,
                         range_bars=config.Buy_Range_Step, price_position=0, buys_total=config.Buys_Total,
opened_short=False):
                self.Symbol = symbol
```

```
            self.bitmex = BitMEX(symbol=self.Symbol, api_key=config.API_KEY_BITMEX,
api_secret=config.API_SECRET_BITMEX,
                                 base_url='https://www.bitmex.com/api/v1/', orderid_prefix='30k_bot_')
            self.Count_Buy = int(count_buy)
            self.Opened_Buys = opened_buys
            self.Bars_Counter = int(bars_counter)
            self.SUM_AMOUNT = float(sum_amount)
            self.Buy_Range_Bars_Var = int(range_bars)
            self.Sum_Orders_Price = 0
            self.Price_Open = -1
            self.Date_Open = None
            self.Price_position = float(price_position)
            self.Buys_Total = buys_total
            self.Opened_Short = opened_short
            self.SL = 0
            self.five_minute = -1
            self.one_minute = -1

            post = "Bot launched"
            bot.send_message(CHANNEL_NAME, post)

        @staticmethod
        def check_can_buy(quotes, b_low, ma, b_up):
            return quotes['low'].iloc[-2] > b_low[-1] > quotes['low'].iloc[-1] and (not (b_up[-1] > ma[-2] > b_low[-
1]))

        @staticmethod
        def additional_buy(quotes, ma, bands_main, bands_low, bands_up):
            ma3 = float(ma[-3])
            high1 = float(quotes['high'].iloc[-1])
            low1 = float(quotes['low'].iloc[-1])

            return (((ma3 > high1 and ma3 > bands_up[-1]) or (ma3 < low1) and (high1 < bands_main[-1])
                    or (low1 < bands_low[-1])) or (quotes['low'].iloc[-2] > bands_low[-1] > low1))

        def reset_vars(self):
            self.Count_Buy = 0
            self.Opened_Buys = False
            self.SUM_AMOUNT = 0
            self.Price_position = 0.0
            self.Sum_Orders_Price = 0.0
            self.Buy_Range_Bars_Var = config.Buy_Range_Step
```

```
        self.Price_Open = -1
        self.Buys_Total = config.Buys_Total
        self.SL = 0

    def check_vars(self):
        # check if count of opened orders at exchange equal to bot
        contracts_now = self.bitmex.get_total_contracts_amount()
        if contracts_now != self.SUM_AMOUNT:
            post = """
            Contracts now (on exchange): {}
            Contracts should be: {}
            """.format(contracts_now, self.SUM_AMOUNT)
            bot.send_message(CHANNEL_NAME, post)

    def strategy(self):
        try:
            min_now = datetime.datetime.now().minute

            # 1 minute bar control
            if self.one_minute == min_now:
                return False
            self.one_minute = min_now

            # ### Close Buy by SL
            if self.Opened_Buys:
                bid = self.bitmex.get_price('bids')
                if bid <= self.SL:
                    # 1) Close Buy Position
                    if self.bitmex.sell(self.SUM_AMOUNT, bid):  # note: bid or self.SL ?
                        price_close = bid
                        logs = '### {}: Sold by SL {} at {}'.format(self.Symbol, self.SUM_AMOUNT, price_close)
                        logging.info(logs)
                        print(logs)

                        # Approximately profit
                        profit = round((price_close - self.Price_position) / self.Price_position * 100, 2)
                        log = ' #-#-# {} Approximately profit: {}%'.format(self.Symbol, profit)
                        logging.info(log)
                        print(log)

                        # reset vars/counters
                        self.reset_vars()
```

```
            # 2) Open Reverse SHORT Position
            rate = self.bitmex.get_price('asks')
            rate -= 0.5
            amount = AMOUNT_CONTRACTS
            self.bitmex.set_leverage(LEVERAGE_LVL)  # setup leverage
            if self.bitmex.sell(amount, rate):
                self.Opened_Short = True
                self.Price_Open = rate
                self.Date_Open = time.time()
                logs = '### {}: Opened Reversed Short {} at {}'.format(self.Symbol, amount, price_close)
                logging.info(logs)
                print(logs)

# 5 minute bar control
sec_now = datetime.datetime.now().second
while sec_now <= 30:
    time.sleep(10)
    sec_now = datetime.datetime.now().second
if not (min_now % 5 == 0 and self.five_minute != min_now):
    return False
self.five_minute = min_now

quotes_json = self.bitmex.get_quotes(self.Symbol, time_frame='5m')
quotes = pd.DataFrame(quotes_json)

# region ### Get Indicators
close = np.array(quotes['close'].values, dtype=float)
high = np.array(quotes['high'].values, dtype=float)
low = np.array(quotes['low'].values, dtype=float)
bbands = talib.BBANDS(close, Bands_Period, Bands_Deviation_Up, Bands_Deviation_Dn)
bands_low = bbands[2]
bands_main = bbands[1]
bands_up = bbands[0]
ma = talib.SMA(close, config.MA_Period)
atr = talib.ATR(high, low, close)
# endregion

# Close short
if self.Opened_Short and self.check_can_buy(quotes, bands_low, ma, bands_up):
    rate = self.bitmex.get_price('bids')  # get BID (not Ask) price
    rate += 0.5
```

```
        amount = AMOUNT_CONTRACTS
        self.bitmex.set_leverage(LEVERAGE_LVL)  # setup leverage lvl
        if self.bitmex.buy(amount, rate):
            price_close = rate
            sold_amount = amount
            logs = '### {}: Closed Short {} at {}'.format(self.Symbol, sold_amount, price_close)
            logging.info(logs)
            print(logs)

            # Approximately profit
            profit = round((self.Price_Open-price_close) / self.Price_Open * 100, 2)
            log = ' #-#-# {} Approximately profit: {}%'.format(self.Symbol, profit)
            logging.info(log)
            print(log)

            self.Opened_Short = False
            self.Price_Open = 0

    # ### Close buy POSITION
    if self.Opened_Buys:
        if quotes['high'].iloc[-1] > bands_up[-1]:

            # set the bid price
            rate = self.bitmex.get_price('asks')
            # rate -= (rate * 0.0001)  # and add 0.3 percent to the Bid price
            rate -= 0.5

            # close buy position
            if self.bitmex.sell(self.SUM_AMOUNT, rate):
                price_close = rate
                sold_amount = self.SUM_AMOUNT
                logs = '### {}: Sold {} at {}'.format(self.Symbol, sold_amount, price_close)
                logging.info(logs)
                print(logs)
                # Approximately profit
                profit = round((price_close - self.Price_position) / self.Price_position * 100, 2)
                log = ' #-#-# {} Approximately profit: {}%'.format(self.Symbol, profit)
                logging.info(log)
                print(log)

                # reset vars/counters
                self.reset_vars()
```

```
                    # Open Short position
                    if profit < 0:
                        rate = self.bitmex.get_price('asks')
                        rate -= 0.5
                        amount = AMOUNT_CONTRACTS
                        self.bitmex.set_leverage(LEVERAGE_LVL)  # setup leverage lvl
                        if self.bitmex.sell(amount, rate):
                            self.Opened_Short = True
                            self.Price_Open = rate
                            self.Date_Open = time.time()
                            logs = '### {}: Opened AL Short {} at {}'.format(self.Symbol, amount, price_close)
                            logging.info(logs)
                            print(logs)

                # ### Buy
                if not self.Opened_Short and not self.Opened_Buys and self.check_can_buy(quotes, bands_low,
ma, bands_up):
                    rate = self.bitmex.get_price('bids')  # get BID (not Ask) price
                    rate += 0.5
                    amount = AMOUNT_CONTRACTS
                    self.bitmex.set_leverage(LEVERAGE_LVL)  # setup leverage lvl
                    if self.bitmex.buy(amount, rate):
                        self.Opened_Buys = True
                        self.Count_Buy = 1
                        self.Price_Open = rate
                        self.Date_Open = time.time()
                        self.SUM_AMOUNT = amount
                        self.Price_position = self.Sum_Orders_Price = rate
                        self.SL = self.Price_Open - atr[-1]*10
                        logs = '### {}: Bought {} at {}. SL at {}'.format(self.Symbol, amount, self.Price_Open,
self.SL)
                        logging.info(logs)
                        print(logs)

                # ### Additional Buys
                if self.Opened_Buys and self.Buys_Total > self.Count_Buy > 0 > \
                        float(quotes['open'].iloc[-1]) - self.Price_position:
                    self.Bars_Counter += 1
                    if self.additional_buy(quotes, ma, bands_main, bands_low, bands_up):
                        if self.Bars_Counter > self.Buy_Range_Bars_Var:
                            rate = self.bitmex.get_price('bids')
```

```
                    # rate += (rate * 0.0003)  # and add 0.3 percent to1 the Bid price
                    rate += 0.5
                    amount = AMOUNT_CONTRACTS
                    self.bitmex.set_leverage(LEVERAGE_LVL)  # setup leverage lvl again
                    if self.bitmex.buy(amount, rate):
                        amount_bought = amount
                        self.Count_Buy += 1
                        self.Bars_Counter = 0
                        self.Price_Open = rate
                        self.SUM_AMOUNT += float(amount_bought)
                        self.Sum_Orders_Price += rate
                        self.Price_position = self.Sum_Orders_Price / self.Count_Buy
                        self.SL = self.Price_position - atr[-1]*10
                        logs = '### {}: Bought Additional {} at {}. SL at {}'.format(self.Symbol, amount_bought,
                                                                                     self.Price_Open, self.SL)
                        logging.info(logs)
                        print(logs)
                        if rate - self.Price_position < 0:
                            self.Buys_Total += 1

                # save current state of important variables to DB (table Recovery)
                save_vars(self.Symbol, self.Count_Buy, self.Opened_Buys, self.Bars_Counter,
self.SUM_AMOUNT,
                          self.Buy_Range_Bars_Var, self.Price_position)

                if datetime.datetime.now().minute == 30 or datetime.datetime.now().minute == 0:
                    self.check_vars()

                return True
```

Обозначение	Наименование	Дополнительные сведения
	Текстовые документы	
1. 110700026.09.03.02. 595.ПЗВКР	Пояснительная записка	52 с.
	Графические документы	
2. 110700026.09.03.02. 595.ДМВКР	Демонстрационные материалы (презентация)	
	Демонстрационные материалы (пл. ф. А4)	12 л *5 экз
	Другие документы	
3.	Документы на компакт-диске	1 CD

Изм.	Лист.	Номер докум.	Подп.	Дата	110700026.09.03.02. 595.ПЗВКР		
Разработал		Дворяшин К.С.			Автоматическая система торговли на валютном рынке Ведомость ВКР	Лит.	Лист 62 / Листов 63
Проверил		Гахов Р.П.				У	
Н.контр.		Нестерова Е.В.				НИУ «БелГУ» гр.07001407	
Утвердил		Иващук О.А.					

Выпускная квалификационная работа выполнена мной совершенно самостоятельно. Все использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

«___» _____ _____ г.

_____ _____