Санкт–Петербургский государственный университет

Карпий Павел Евгеньевич

Выпускная квалификационная работа

Разработка блокчейн-сети для тестирования экономической модели

Уровень образования: бакалавриат
Направление 01.03.02 «Прикладная математика и информатика»
Основная образовательная программа СВ.5005.2016 «Прикладная математика, фундаментальная информатика и программирование»
Профиль «Исследование и проектирование систем управления и обработки сигналов»

Научный руководитель:
доцент, кафедра компьютерных технологий
и многопроцессорных систем,
к.ф. - м.н. Корхов Владимир Владиславович

Санкт-Петербург
2020 г.

Содержание

Термины и определения

Геймификация — это использование игровых подходов, для неигровых процессов, которое позволяет повысить вовлеченность участников в решение прикладных задач, использование продуктов, услуг, усилить лояльность клиентов.

Децентрализация — это процесс перераспределения, рассеивания функций, сил, власти, людей или вещей от центрального местоположения или управляющего органа.

Доверительная среда — это среда, в которой каждый участник доверяет другому вследствие имеющихся систем или правил, которые исключают возможность недобросовестного поведения.

Контейнеризация — это метод виртуализации, при котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя вместо одного.

Криптовалюта — это разновидность цифровой валюты, учет внутренних расчетных единиц которой обеспечивает децентрализованная платежная система.

Нода — это любой компьютер, подключенный к блокчейн-сети. Ноды децентрализованной сети контактируют посредством P2P-протоколов для обмена информацией о блоках и транзакциях.

Публичный ключ — это уникальный индивидуальный адрес, который виден всем в блокчейн-сети, а также, который используется для в качестве адреса для входящих переводов.

Смарт-контракт — это компьютерная программа или протокол транзакции, который предназначен для автоматического выполнения, контроля или документирования соответственно юридически значимых событий и действий в соответствии с условиями контракта, соглашения или переговоров.

Фиатная валюта — это узаконенное платежное средство, ценность которого устанавливается правительством и выпускается им, необеспеченное физическим товаром или продуктом.

Docker — это программное обеспечение для автоматизации развер-

тывания и управления приложениями в средах с поддержкой контейнеризации.[15]

Перечень сокращений и обозначений

Админ-аккаунт — это профиль, который обладает бóльшими привилегиями по сравнению с обычными пользователями.

DLT — a distributed ledger (распределенный реестр).

Ethereum — это открытая общедоступная платформа распределенных вычислений разрабатываемая Ethereum Foundation.[14]

ERC — Ethereum Request for Comment. Это технические документы, используемые разработчиками смарт-контрактов в Ethereum[10].

MongoDB — это кроссплатформенная документально-ориентированная база данных[6].

SQL — Structured Query Language. Предметно-ориентированный язык, используемый в программировании и предназначенный для управления данными, хранящимися в системе управления реляционными базами данных.

NoSQL — это термин, обозначающий ряд подходов, направленных на реализацию систем управления базами данных, имеющих существенные отличия от моделей, используемых в традиционных систем управления реляционными базами данных.

HLF — Hyperledger Fabric.

PoC — Proof of Concept. Реализация определенного метода или идеи для демонстрации его осуществимости или демонстрация в принципе с целью проверки того, что некоторая концепция или теория имеет практический потенциал.

VPS — Virtual Private Server. Виртуальный выделенный сервер.

BasicAuth — способ представления пользователя и пароля при выполнении запроса для пользовательского агента HTTP (например, веб-браузера)[17].

Введение

Блокчейн — выстроенная по определенным правилам непрерывная последовательная цепочка блоков, содержащих информацию. Связь между блоками обеспечивается не только нумерацией, но и тем, что каждый блок содержит свою собственную хеш-сумму и хеш-сумму предыдущего блока. Для изменения информации в блоке необходимо редактировать и все последующие блоки. Чаще всего копии цепочек блоков хранятся на множестве разных компьютеров независимо друг от друга. Этот факт делает крайне затруднительным внесение изменений в информацию, уже включенную в блоки, что может быть использовано в различных сферах, где может существовать потенциальное недоверие между взаимодействующими участниками, например – в финансовых системах. Блокчейн это новая технология, которая уже сильно изменила финансовую сферу, сильно упростила передачу активов третьим лицам и их привлечение на развитие проектов.

Блокчейн на сегодняшнем этапе своего развития не повлиял существенно на жизнь обычных людей, но уже доказал, как кардинально может изменить бизнес-процессы[2]. Компании, использующие эту технологию, становятся трансграничными: у них появляется возможность получить новых клиентов по всему миру. Сейчас блокчейн главным образом используют компании, предоставляющие виртуальные услуги.

Учитывая низкую вероятность успешной атаки децентрализованной сети[13] – подмены или удаления информации из общего реестра –, а также возможность запрограммировать исполнение транзакций (выплат) и хранения критически важной служебной информации в смарт-контрактах, чье поведение (исполнение) является полностью контролируемым и легко проверяемым каждым участником сети, блокчейн технологии могут быть использованы для оплаты труда, внутри компаний, а также на платформах, где оплата труда происходит не на постоянной основе, а по факту выполнения исполнителем оговоренных заранее условий, а сами взаимоотношения между заказчиком и исполнителем могут носить исключительно формальный характер, и могут не обязывать к личной встрече сторон и даже не обязывать к разглашению личной информации сторон друг о друге (за

исключением публичного ключа). Более того, при определенных соглашениях на этапе начала сотрудничества сторон, смарт-контракты позволяют заказчику запрограммировать качественную проверку результатов выполнения обязательств стороной исполнителя. Ввиду описанных особенностей таких платформ, естественным образом возникает не доверительная среда. Использование блокчейн технологий позволяет урегулировать отношения сторон на подобного рода платформах и минимизировать недобросовестное исполнение своих обязательств сторонами.

Однако человеку свойственно не доверять и относится с опасением к валютам, чей обменный курс не контролируется каким-либо центральным регулятором и не привязан к чему-нибудь конкретному, как это сделано с фиатными валютами.

Поэтому возникает вопрос о разработке системы, которая бы поддерживала оборот криптовалюты и ее обмен на фиатные активы на описанных выше платформах, т.е. система, которая бы совмещала внутри себя как крипто, так и фиатные активы и которая также позволяла бы обменивать их, подчиняясь рыночным законам, регулирующим обменный курс. Также важно отметить, что ключевым фактором такой системы должно являться вознаграждение пользователя за выполненные задачи в обеих валютах, то есть выплаты в фиатных активах не должны быть полностью заменены на вознаграждение в виртуальных валютах. Этот факт позволит системе функционировать (продолжать выплаты в фиатной валюте напрямую от заказчика к исполнителю) даже при ее «полной несостоятельности» в плане прогнозирования поведения пользователей или инвесторов, или – что равносильно – при полном обесценивании токена.

Также сохранение выплат в фиатных валютах объясняет, почему имеется возможность экспериментировать с различными подходами регулирования криптовалюты внутри сети такими, как уменьшение количественного эквивалента выплат, введение порога, после которого выплаты закончатся, введение уникальных для каждого пользователя понижающих и повышающих коэффициентов на выплаты и тому подобные: криптовалюта в данной системе не является основной валютой, и ее целью не является замена привычных людям денежных активов. Она используется как

«бонус» при выполнении заданий внутри платформы, а также повышает лояльность пользователей, так как в случае успешного функционирования сети ценность токенов будет увеличиваться со временем, вызывая логичную стратегию их сохранения и накопления и, как следствие, повышение лояльности пользователей.

Цели работы

- Изучить предложенную экономическую модель (с возможностью внесения своих идей)
- Изучить имеющиеся в открытом доступе децентрализованные платформы и выбрать наиболее подходящую под требования, обусловленные экономической моделью
- Проанализировать дополнительные инструментальные требования для реализации тестовой модели
- Разработать тестовую модель со следующим функционалом:
  - Реализация своей собственной криптовалюты с использованием DLT
    - Контейнеризированное приложение
    - Функционирующие согласно модели токены
  - Серверная часть
    - Контейнеризированное приложение
    - Интеграция с блокчейн-платформой
    - Возможность регистрации и дальнейшей авторизации пользователей
    - Наличие привилегированного админ-аккаунта
    - Реализация математической части экономической модели
    - Развертывание на VPS
  - Клиентская часть
    - Контейнеризированное приложение
    - Возможность регистрации и дальнейшей авторизации пользователей
    - Наличие аккаунт страницы с возможностью выполнения операций согласно экономической модели

    * Визуализация состояния сети для админ-аккаунта

– Тестирование (наличие тестового режима работы)

    * Моделирование роста числа новых пользователей в системе согласно моделям
    * Моделирование поведения пользователей в сети с возможностью изменения параметров

Глава 1. Обзор литературы

1.1 Модель ERC20 токена в сети

ERC20 – это разработанный в 2015 году стандарт токенов, который используется в смарт-контрактах Ethereum блокчейн-сети[3]. ERC20 определяет общий лист правил, который должен реализовывать токен в Ethereum сети, предоставляя разработчикам возможность программировать то, как новые токены будут функционировать в экосистеме Ethereum.

Стандарт токена ERC20 описывает функции и события, которые должен реализовать смарт-контракт токена Ethereum[20].

Язык, на котором написаны смарт-контракты в сети, – Solidity[12].

Перед описанием методов и событий в контракте необходимо определить два map-объекта:

1. **mapping(address ⇒ uint256) balances**;
2. **mapping(address ⇒ mapping (address ⇒ uint256)) allowed**;

Они хранят в себе информацию о текущем состоянии токенов и возможности их перевода. Первый объект – **balances** – хранит в себе информацию о том, сколько токенов находится у каждого пользователя. Второй – **allowed** – будет включать переводы, утвержденные для снятия с данного счета, на другие счета вместе с разрешенной для каждого конкретного счета суммой снятия.

В дополнение к стандартным функциям ERC20 многие токены ERC20 также имеют дополнительные поля, а некоторые стали де-факто частью стандарта ERC20, если не в задокументированной форме, то, как минимум, на практике. Вот несколько примеров таких полей:

- **string public constant name;**

  Имя токена, например «TestToken».

- **string public constant symbol;**

  Символ токена, например «TT».

- **uint8 public constant decimals;**

Количество десятичных знаков, которые использует токен – например, 8, означает, что возвращаемое количество токенов необходимо разделить на 100000000, чтобы получить его действительное представление в системе.

У объекта контракта должен быть реализован следующий набор методов:

1. **function totalSupply() public view returns (uint256);**

   метод, который будет в дальнейшем возвращать общее количество токенов, выпущенных этим смарт-контрактом, независимо от того, кто вызвал этот метод.

2. **function balanceOf(address tokenOwner) public view returns (uint);**

   метод, который вернет текущий баланс токена учетной записи, идентифицируемой по адресу ее владельца (**tokenOwner**).

3. **function allowance(address tokenOwner, address spender) public view returns (uint);**

   метод, который возвращает количество токенов, заявленных **tokenOwner**, как число, которое может быть переведено на счет **spender**.

4. **function transfer(address to, uint tokens) public returns (bool);**

   метод, который используется для перемещения количества токенов **tokens** с баланса их владельца на баланс другого (**to**) пользователя или получателя. Передающий владелец – тот, кто вызывает исполнение метода смарт-контракта, его адрес не требуется явно при вызове метода, это регулируется внутренними механизмами сети, что подразумевает, что только владелец токенов может передать их другим участникам сети. Также внутри метода необходимо реализовать проверку на наличие запрашиваемого числа токенов **tokens** у пользователя, который вызывает метод.

5. **function approve(address spender, uint tokens) public returns (bool);**

   метод, который позволяет разрешить в будущем перевод делегатом **tokens** токенов со счета пользователя, который вызывает метод, на **spender** счет. Это позволяет сети завершить сделку, не дожидаясь предварительного одобрения операции.

6. **function transferFrom(address from, address to, uint tokens) public returns (bool);**

   метод, который должен быть использован совместно с методом **approve**. Это позволяет делегату, утвержденному на снятие средств, переводить средства владельца на сторонний счет. При этом внутри метода должны быть реализованы проверки на наличие у **from** достаточного количества токенов **tokens**, а также наличие разрешения перевода со счета **from** на счет **to** соответствующего числа токенов. Разрешение при этом должно быть выдано ранее с помощью вызова метода **approve** с соответствующими аргументами.

Также смарт-контракт должен определять два специально определенных события (**event**):

1. **event Approval(address indexed tokenOwner, address indexed spender, uint tokens);**

2. **event Transfer(address indexed from, address indexed to, uint tokens);**

Эти события будут вызываться или генерироваться, когда пользователь предоставит права на перевод токенов с его счета, и после того, как токены будут фактически переведены.

1.2 Моделирование роста числа пользователей сети

Для качественного и количественного анализа процесса внедрения новых технологий в обществе необходимо учитывать большое количество параметров таких, как емкость рынка, наличие или отсутствие предыдущих версий технологии, наличие или отсутствие конкуренции в этой технологической нише, репутационная составляющая продукта, степень информирования населения, наличие уже приобретенных аналогов у целевой аудитории продукта, экономическая составляющая в регионе и так далее.

Однако, для моделирования процессов диффузии инноваций, существуют и широко и успешно применяются упрощенные модели, которые учитывают лишь часть всех необходимых параметров, которые поддаются относительно легкому анализу.

Эти модели можно разделить на две категории:

- Модели, основанные на биологических моделях динамики конкурентного взаимодействия биологических популяций.
- Классическая модель Ф. М. Басса и ее обобщения

1.3 Модели, основанные на биологических моделях

Аргументация возможности применения биологических моделей динамики популяций в экономике основана на наличии схожих черт в определениях понятий конкуренции в смысле межвидовой или внутривидовой конкуренции живых организмов, которые обязаны в ней участвовать в виду ограниченности тех или иных биологических или природных ресурсов, и в смысле конкуренции различных производителей товаров, для которых, популяция потребителей, рынки сбыта, различные сырьевые запасы, области выгодного вложения своих активов тоже являются в определенном смысле ограниченными ресурсами, за который им необходимо бороться, предлагая более выгодные по сравнению с конкурентами решения.

1.3.1 Модель Т. Р. Мальтуса.

Модель Т. Р. Мальтуса может быть записана в следующем виде:

$$\frac{dN(t)}{dt} = rN(t), \tag{1}$$

где $t \in \mathbb{R}_{t_0} \equiv [t_0, \infty)$ - время, $r > 0$ — *мальтузианский параметр* или же скорость роста численности популяции.

Также необходимо добавить начальное условие:

$$N(t)|_{t=t_0+0} = N_0 > 0 \tag{2}$$

Из (1) и (2) получается, что численность популяции в зависимости от времени высчитывается по формуле:

$$N(t) = N_0 e^{r(t-t_0)},$$

где $t \in \mathbb{R}_{t_0} \equiv [t_0, \infty)$.

Эта модель может быть использована при моделировании численности однородной популяции, однако, при некоторых условиях демонстрирует нереалистичное поведение, например: $N(t) \to \infty$, $t \to \infty$ что, конечно, не соответствует действительности: максимальная численность пользова-

телей технологии (как и в целом популяции) ограничена естественными природными факторами.

Уравнение (1) можно обобщить до уравнения:

$$\frac{dN(t)}{dt} = G(N(t)), \tag{3}$$

где также $t \in \mathbb{R}_{t_0} \equiv [t_0, \infty)$, а начальные условия те же: (2).

Обычно принято за $G(N)$ брать функцию $M(N)N$, где $M(N)$ – *мальтузианская функция*, однако это необязательно. Предположим, что в среде обитания популяции (или новой технологии) есть какой-то физический ограничитель (m): ресурс, который необходим для существования живых организмов популяции (условие, необходимое для использование популяцией новой технологии соответственно), причем размерность этого ресурса сопоставима с численностью популяции и, например, каждому организму необходимо использовать какое-то количество ресурса для своей успешной жизнедеятельности, то есть это необходимое условие существования данной популяции. Тогда получаем, что $m - N > 0$ — количество оставшегося ресурса, определяющее способность популяции к размножению.

В описанном случае уравнение (3) можно переписать в виде:

$$\frac{dN(t)}{dt} = r[m - N(t)],$$

где также $t \in \mathbb{R}_{t_0} \equiv [t_0, \infty)$, а начальные условия те же: (2).

Решение получившегося уравнения, удовлетворяющее начальным условием (2), записывается в виде:

$$N(t) = m(1 - e^{-r(t-t_0)}) + N_0 e^{-r(t-t_0)}, \tag{4}$$

где также $t \in \mathbb{R}_{t_0} \equiv [t_0, \infty)$, но при этом $N(t) \to m$, $t \to \infty$.

1.3.2 Модель Ферхюльста – Пирла – Рида (логистическое уравнение)

Модель Ферхюльста – Пирла – Рида или логистическое уравнение - модифицированное уравнение (1) с отсутствующим недостатком в ви-

де устремления к бесконечности численности популяции при аналогичном стремлении времени. Записывается оно в виде:

$$\frac{dN(t)}{dt} = r(1 - \frac{N(t)}{m})N(t), \tag{5}$$

где также $t \in \mathbb{R}_{t_0} \equiv [t_0, \infty)$.

Решение уравнения (9) записывается следующим образом:

$$N(t) = \frac{N_0 e^{r(t-t_0)}}{1 + \frac{N_0}{m}[e^{r(t-t_0)} - 1]}, \tag{6}$$

где также $t \in \mathbb{R}_{t_0} \equiv [t_0, \infty)$, но при этом $N(t) \to m$, $t \to \infty$.

Величина в скобках в уравнении (9) является дополнением к мальтузианскому параметру в уравнении (1), которая объясняет увеличение конкуренции при увеличении числа популяции, то есть описывает процесс, называемый «внутривидовой конкуренцией».

1.3.3 Модель Басса

Модель Басса была разработана Фрэнком Бассом. Она состоит из простого дифференциального уравнения, которое описывает процесс принятия новых продуктов в популяции. Модель представляет обоснование того, как взаимодействуют нынешние и потенциальные пользователи нового продукта. Основная предпосылка модели заключается в том, что новые пользователи могут быть классифицированы как новаторы или подражатели, а скорость и сроки принятия продукта в популяции зависят от «степени их новаторства» (то есть то, с каким желанием новые пользователи начинают использовать новую технологию) и «степени имитации» (то есть то, каким образом потенциальные новые пользователи готовы попробовать новый продукт под воздействием влияния пользователей уже использующих его) среди потенциальных пользователей продукта. Модель Басса широко используется в прогнозировании, особенно в прогнозировании продаж новых продуктов и прогнозировании развития технологий. Математически основная диффузия Баса представляет собой уравнение Риккати с постоянными коэффициентами.

Уравнение Риккати с постоянными коэффициентами:

$$y' = ay + by^2 + c,$$

где a, b, c — константы.

Уравнение Басса:

$$\frac{dN(t)}{dt} = \left[p + q\frac{N(t)}{m}\right][m - N(t)], \tag{7}$$

где также $t \in \mathbb{R}_{t_0} \equiv [t_0, \infty)$, а начальные условия те же: (2).

В (7):

$N(t)$ – число пользователей в популяции, который уже пользуются новой технологией,

$m - N(t)$ – число пользователей, который не пользуются новой технологией, но потенциально могут начать это делать,

p – «степень новаторства», то есть характеристика, которая определяет, насколько быстро принимается новая технология в популяции. Этот параметр является собирательным и состоит в том числе из репутации нового продукта, степени его разрекламированности в популяции, экономической ситуации в регионе (в том числе то, насколько велика покупательная способность у популяции), отношения Средств Массовой Информации к продукту и так далее. Например, в обществе, где производитель товара по какой-то причине не организовал маркетинговую компанию по продвижению своего товара, а продукт можно купить только заказав его со склада (то есть на полках магазина его нельзя увидеть обычному покупателю), этот коэффициент равен нулю.

q – «степень имитации», то есть характеристика, которая определяет, насколько сильно подвержены пользователи общественному мнению уже приобретших технологию индивидов. Например, в полностью рациональном обществе, где каждый индивид пользуется исключительно рациональными доводами о новой технологии и не подвергается общественному давлению со стороны уже имеющих новых продукт пользователей и люди не рассказывают друг другу о новых приобретениях (так же, для наглядности

предположим, что количество уже реализованных продуктов новой технологии не влияет на опыт использования, то есть пользоваться продуктом, который есть у всех также удобно, как и пользоваться продуктом, которого нет ни у кого), этот коэффициент будет равен нулю.

Решение задачи (7), (2):

$$N(t) = m\frac{(pm + qN_0) - p(m - N_0)e^{-(p+q)(t-t_0)}}{(pm + qN_0) + q(m - N_0)e^{-(p+q)(t-t_0)}} = m\frac{1 - p\Delta e^{-(p+q)(t-t_0)}}{1 + q\Delta e^{-(p+q)(t-t_0)}}, \quad (8)$$

где $\Delta = \frac{m-N_0}{pm+qN_0}$, и $N(t) \to m$, $t \to \infty$.

Существует также большое количество обобщений модели Басса другими авторами с дополнительными параметрами, добавленными эвристически вследствие статистических наблюдений, либо же с помощью распространения модели на случай более одной компании на рынке (увеличение размерности дифференциального уравнения). Например:

$$\frac{dN(t)}{dt} = \left[p + \left(q\frac{N(t)}{m}\right)^{\delta}\right][m - N(t)], \quad (9)$$

где $\delta > 0$ — некоторая положительная константа.

Глава 2. Основные результаты работы

2.1 Выбор блокчейн-платформы

Учитывая, что первоочередная задача была сформулирована как PoC и что вероятные результаты не предполагались к полноценному коммерческому использованию, а система должна выступать исключительно в роли платформы для тестирования, было принято решение о нецелесообразности разработки собственной распределенной системы ввиду очень больших временных и энергетических затрат процесса ее реализации.

Поэтому было решено выбрать одну из существующих в открытом доступе децентрализованную платформу.

Определяющими критериями были установлены:

- Существование публичного режима работы сети
- Наличие смарт-контрактов в платформе
- Комиссия за проведение транзакций (стоимость транзакций)

Также учитывался косвенный критерий при отборе платформ: популярность (число использований в мире), т.к. при разработке важно также учитывать, насколько часто использовалось ранее это решение и насколько быстрым будет поиск дополнительной информации.

Результаты приведены в таблице:

Таблица 1

Блокчейн	Публичность	Смарт-контракты	Комиссия
Ethereum	Да	Да	Настраиваемая
Stellar	Да	Да	0.00001 Lumens
Neo	Да	Да	Нет
IOTA	Да	Нет	Нет
HLF	Нет	Да	Нет
EOS.IO	Да	Да	Нет

После анализа был выбран Ethereum. Выбор был обоснован популярностью Ethereum блокчейн-сети в обществе на момент проведения анализа, а также имеющимися хорошо описанными стандартами реализации собственных токенов в сети[14].

2.2 Реализация токенов с помощью смарт-контракта

Так как система предполагает, что токены являются исключительно частью системы лояльности, и могут выдаваться только в качестве поощрения за выполняемые задания на платформе, обмениваться на фиатную валюту, а также не могут быть переданы от пользователя к пользователю, то полноценный ERC20 стандарт не может быть применен для описанной задачи, следовательно, некоторые методы необходимо изменить или удалить.

Методы **Transfer()** и **TransferFrom()** необходимо переписать в методы **TransferFromAdmin()** и TransferToAdmin(), которые должны вы-

полняться при любых операциях с токенами. Также необходимо ввести проверку на вызывающего эти методы, так как только администратор должен обладать правами на перевод токенов со своего счета на счет пользователей, это достигается написанием дополнительного класса, который обладает функциями проверки вызывающего его пользователя сети, а также обладает функциональностью передачи своих привилегированных прав при необходимости.

Событие **Transfer()** сохраняет свою пользу и применимость для данной задачи. Однако в то же время событие **Approval()** не имеет осмысленного применения для тестовой экономической модели, поэтому было принято решение его не использовать.

Конструктор основного контракта также должен учитывать возможность быть инициализированным с параметром, который представляет начальное значение выпущенного количества токенов, так как контракт будет разворачиваться многократно в виду частого запуска модели для тестирования.

2.3 Выбор базы данных

Для корректной работы модели и поддержания состояния зарегистрированных пользователей необходимо использование базы данных. При выборе учитывались следующие критерии:

- Запросы к базе данных не планируются быть комплексными
- Скорость разработки и простота работы приоритетнее функционала
- При тестировании возможны запись и чтение большого объема данных и важна скорость этих операций
- Производительные мощности, выделяемые для работы некоммерческого проекта, ограниченны
- Так как изначально была известна только экономическая модель, без конкретного плана ее реализации, то с высокой долей вероятности

структуры, хранимые в базе данных, будут меняться со временем, как и отношения между таблицами базы данных

Учитывая вышеописанные критерии, был выбран NoSQL подход реализации систем управления базами данных, а внутри этого подхода – MongoDB.

2.4 Серверная часть

Вся серверная часть была написана с использованием языка Golang[4] и контейнезирована в docker-контейнер[7].

На сервере происходит обработка запросов с клиентской части[9]. Авторизация происходит по методу BasicAuth, так как перед приложением не стояло задачи быть максимально защищенным на текущем этапе разработки.

Так же серверная часть отвечает за:

- Все операции с базой данных, в том числе миграцию данных при каждом запуске
- Все операции с Ethereum
- Выполнение математической части экономическом модели
  - Начисление токенов за выполнение заданий на платформе
  - Расчет геймификации (для повышения лояльности пользователя)
  - Расчет дивидендов (для повышения лояльности пользователя)
  - Подготовку статистики для информации админ-аккаунта
  - Изменение параметров модели со страницы админ-аккаунта, и/или полный перезапуск экономической модели, а также управление временем в моделируемом экономическом процессе

2.5 Клиентская часть

Клиентская часть была написана с использованием Vue.js фреймворка и контейнзирована в docker-контейнер с Nginx веб-сервером[18][19].

Глава 3. Тестирование

3.1 Моделирование роста числа пользователей

Было предложено тестирование по следующему сценарию.

Перед запуском системы в тестовом режиме необходимо заполнить конфигурационный файл, в котором нужно указать параметры моделирования платформы.

Прежде всего, это выбор функции, согласно которой развивается система. Сюда относится выбор одной из заранее запрограммированных функций, определяющих, по какому закону будет увеличиваться число новых пользователей системы. Этой функцией может выступать функция, являющаяся решением уравнения обобщенной модели Мальтуса (4) или решением уравнения модели Ферхюльста – Пирла – Рида (логистического уравнения) (6) в случае, если выбрано моделирование, основанное на биологических моделях. В противном случае – то есть в случае выбора модели Басса – используется решение (8).

Для моделирования развития системы обязательно указание значения параметров m и r для биологических моделей, а также дополнительных параметров p и q для модели Басса. Общим параметром для всех моделей является N_0 — начальное значение пользователей сети.

3.2 Моделирование поведения пользователей

Дополнительно было предложено использовать параметр $k \in [0, 1)$. Этот параметр отвечает за процентное соотношение «разумных» пользователей сети, то есть тех пользователей, которые при планировании своих дальнейших действий пользуются здравым смыслом: если есть основания полагать на основе уже известных общих данных, что система будет

продолжать успешно развиваться, то такие пользователи продолжат сохранять активы в виде токенов с целью ожидания увеличения обменного курса и, как следствие, увеличения своего капитала, и наоборот, если есть тенденции к снижению обменного курса, то такие пользователи предпочтут избавляться от активов. Так как прогнозирование обменного курса является очень сложной задачей на долгосрочную перспективу и зависит от большого количества факторов даже в упрощенных моделях, не говоря уже о реальном мире[9], для прогнозирования «разумным» пользователям было предложено использовать простую линейную аппроксимацию обменного курса методом наименьших квадратов по значениям обменного курса на начало пяти последних расчетных периодов. В случае ожидания увеличения курса по предложенному методу, пользователь не будет обменивать токены на фиатную валюту, а также будет более лояльным к платформе и продолжит выполнять на ней задания. В случае ожидания понижения курса пользователь предпримет попытку вывести часть своих токенов из платформы (процентное число выведенных токенов будет также определяться исходя из ожидания, т.е. от модуля знака производной аппроксимирующей функции), а также в зависимости от степени ожидания будет стараться брать краткосрочные работы на платформе, либо же откажется от работы на платформе в ближайший расчетный период совсем. Пороговые значения модуля производной для принятия решений о выводе также определяются конфигурационным файлом.

Соответственно, $1 - k \in (0, 1]$ – процентное соотношение «неразумных» пользователей, которые поступают псевдослучайно. К ним же и относятся пользователи, поведение которых невозможно спрогнозировать и невозможно отнести к «разумным» или «неразумным» действиям: предложенное моделирование не учитывает большого количества внешних факторов.

Под работой на платформе подразумевается выполнение заданий, оплата которых начисляется и в токенах, и в фиатных валютах. При этом оплата в токенах является фиксирована в фиатном эквиваленте, то есть сумма выплаты токенов зависит от текущего обменного курса. Для моделирования процесса выполнения заданий, на платформе уже имеется 5

шаблонных заданий, оплата которых варьируется от 1000 до 5000 условных фиатных единиц с шагом в 1000. Выполнение заданий у пользователя занимает, соответственно, от 1 до 5 расчетных периодов.

3.3 Моделирование временных процессов

В конфигурационном файле также необходимо указать число расчетных периодов, в течение которых будет моделироваться система. Фактическая продолжительность расчетного периода в моделируемом процессе – 20 секунд, то есть приблизительное время, необходимое для добавления двух блоков в сеть Ethereum. В случае моделирования процесса длящегося 10 лет (120 расчетных периодов) понадобится приблизительно 40 минут фактического времени.

3.4 Возникнувшие проблемы

В ходе тестирования различных подходов (различных моделей) и различных параметров были получены следующие результаты: при выборе хотя бы в какой-то мере приблизительно отражающих реальность параметров система показала себя вполне успешно и демонстрировала практически стабильный рост в смысле повышения обменного курса. Однако, как и предполагалось до начала моделирования и проведения первых тестовых запусков, существует очевидная проблема, которая в некоторых случаях не позволяла системе функционировать даже в течении одного смоделированного года: так как запас токенов является фиксированным и строго определяемым до начала всего имитационного моделирования, то чрезмерная скорость развития может оказать негативное влияние на платформу. Она продолжит функционировать, так как криптовалюта не была единственным поощрением за выполнение заданий, однако нарушается система лояльности, общий набор благ, получаемых за сотрудничество с платформой, снижается, система теряет свои конкурентные стороны.

Выпуск дополнительных токенов – очевидное на первый взгляд решение. Но такой подход приведет к неминуемой инфляции – потере ценности токенов, что только может еще более отпугнуть потенциальных и тем более

уже лояльных пользователей.

Еще одним менее очевидным недостатком является отсутствие антимонопольной системы. Нет никаких механизмов, которые бы препятствовали или хотя бы сдерживали более трудолюбивых или более старых в плане времени нахождения на платформе пользователей. Если у человека во владении находится слишком большая часть общего числа выпущенных токенов, то он де-факто становится потенциальным монополистом. Вся система становится слишком зависимой от него и в случае его ухода (что равносильно полной распродаже токенов и выводу большого количества фиатных денег из системы) может отрицательно отреагировать лавинной продажей всей криптовалюты пользователями, которые могут принять решение о несостоятельности платформы в смысле хранения активов.

3.5 Предложенное решение

Криптовалюта не является полноценной валютой в рассматриваемой системе. За нее не предлагают товары, она не может быть передана корыстно или, наоборот, безвозмездно кому-либо, она исключительно выполняет стимулирующую и сберегательную роль. Основываясь на этой идее, было предложено решение описанных выше проблем: неявный контроль за числом токенов, который бы сделал невозможным наличие монополиста, а также исключил бы (максимально отсрочил, сдержал) ситуацию нулевого числа свободных токенов в системе.

Пусть у $\forall$ i-го пользователя имеется свой уникальный множитель

$$g_i \in (0, 1)$$

Пусть он подчиняется следующему правилу:

$$g(x) \to 0, x \to \infty, \tag{10}$$

а

$$x = \left| \frac{T_a}{T_a - t_a} \right|,$$

где T_a — максимальное число токенов, которое, предполагается, мо-

жет иметь пользователь, высчитывается оно следующим образом:

$$T_a = \frac{T_{total}}{N},$$

где T_{total} – общее число токенов, которое планируется выдать пользователям, N - текущее число пользователей.

В качестве функции $g(x)$ можно, например, использовать смещенный и инвертированный гиперболический тангенс:

$$\tanh +1$$

так как он подходит по условиям (10)(см. рис. 1).

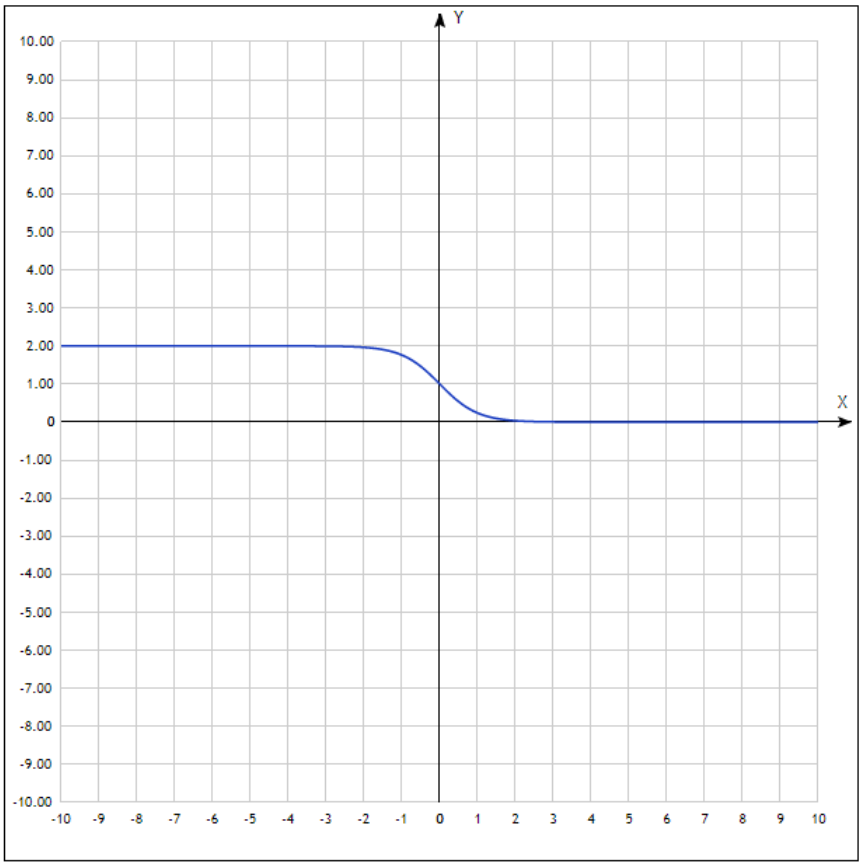

Рис. 1

Таким образом, «разумный» пользователь будет решать задачу максимизации прибыли:

$$(t_a - t)a - \frac{1000}{exchRate}(1 - g(t_a - t)) \to max \tag{11}$$

при условиях $t \in [0, t_a]$ – количество токенов, которое можно продать в следующем месяце. (t_a — текущее состояние, а $\frac{1000}{exchRate}$ — это крипто валютный эквивалент, который пользователь должен получать).

В (11) уменьшаемое имеет смысл «пассивного» дохода, который пользователь получает, если не будет продавать токены. Коэффициент a характеризует все тот же угол наклона аппроксимирующей прямой по пяти последним известным точкам обменного курса. Вычитаемое же, наоборот описывает потери в случае бездействия (отсутствия действий по продаже токенов), ведь теперь на все заработанные токены накладывается «штраф», определяемый функцией $g()$.

Производную функции (11) можно упростить следующим образом:

$$-a - \frac{1000}{exchRate}(sch(t_a - t))$$

Исходя из вида функции, становится понятно, что в случае положительности коэффициента a «разумный» пользователь будет стремиться минимизировать свои убытки и поэтому продаст все токены, так как $t = t_a$ будет оптимальной точкой на заданном интервале. А вот в случае отрицательности коэффициента возможны три случая:

1. a по модулю все еще слишком мал на рассматриваемой области, тогда все еще оптимальная точка — это продажа всех имеющихся токенов

2. a по модулю частично превосходит монотонный гиперболический секанс на рассматриваемой области и тогда точка, где они пересекаются, – точка минимума, следовательно, необходимо смотреть значение функций на концах отрезка и искать наибольших среди них

3. a по модулю строго больше секанса на рассматриваемой области, тогда оптимальной точкой является «точка бездействия» - продавать токены невыгодно.

С практической точки зрения, это означает, что разумнее сразу искать значения функции на двух концах отрезка и выбирать наибольший из них без дополнительного исследования.

Глава 4. Заключение

В ходе работы была воспроизведена экономическая модель, полностью готовая для тестирования и проверки гипотезы о своей состоятельности.

Модель состоит из четырех docker контейнеров:

- Серверная часть
- Клиентская часть
- База данных
- Ethereum

Были написаны конфигурационные файлы для docker-compose утилиты, позволяющие с помощью одной команды в командной строке развернуть систему локально.

Также были предложены усовершенствования модели, которые решают две основные проблемы, а именно: проблема траты всей криптовалюты на выплату пользователям (следовательно, автоматическая инфляция в случае выпуска дополнительных), а также проблема несостоятельности системы в случае возврата пользователями всей криптовалюты обратно в смарт-контракт (следовательно, вывод всех фиатных активов).

Следующим логичным продолжением тестирования является углубление в изучение целевой аудиторий практическое отыскание необходимых для моделирования параметров, разработка более детальной модели поведения пользователя в тестируемой экономической сети. Далее имеет смысл смоделировать поведение системы при различных параметрах пользователей на многолетнем промежутке времени.

Список литературы

[1] Merkel D. Docker: lightweight linux containers for consistent development and deployment. Linux journal. 2014 Mar 1;2014(239):2.

[2] Tapscott A, Tapscott D. How blockchain is changing finance. Harvard Business Review. 2017 Mar 1;1(9):2-5.

[3] ERC20 – A standard interface for tokens [Электронный ресурс]. — URL: https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md

[4] Go - an open source programming language [Электронный ресурс]. — URL: https://golang.org/

[5] Bitbucket – Git code management. [Электронный ресурс]. — URL: https://bitbucket.org/product/

[6] MongoDB – cross-platform document-oriented database [Электронный ресурс]. — URL: https://www.mongodb.com/

[7] Docker – enterprise application container platform [Электронный ресурс]. — URL: https://www.docker.com/

[8] net/http – HTTP client and server implementations [Электронный ресурс]. — URL: https://golang.org/pkg/net/http/

[9] httptest – utilities for HTTP testing [Электронный ресурс]. — URL: https://golang.org/pkg/net/http/httptest/

[10] Ethereum – a global, open-source platform for decentralized applications [Электронный ресурс]. — URL: https://ethereum.org/

[11] https://www.researchgate.net/publication/331904191_Neural_networks_perfor

[12] Документация Solidity URL: https://solidity.readthedocs.io/en/develop/commo patterns.html

[13] Gencer AE, Basu S, Eyal I, Van Renesse R, Sirer EG. Decentralization in bitcoin and ethereum networks. InInternational Conference on Financial Cryptography and Data Security 2018 Feb 26 (pp. 439-457). Springer, Berlin, Heidelberg.

[14] Iyer K, Dannen C. First Steps with Ethereum. InBuilding Games with Ethereum Smart Contracts 2018 (pp. 37-56). Apress, Berkeley, CA.

[15] Merkel D. Docker: lightweight linux containers for consistent development and deployment. Linux journal. 2014 Mar 1;2014(239):2.

[16] Andrawos M, Helmich M. Cloud Native Programming with Golang: Develop microservice-based high performance web apps for the cloud with Go. Packt Publishing Ltd; 2017 Dec 28.

[17] Reschke J. The'basic'http authentication scheme. Work in Progress, draft-ietf-httpauth-basicauth-update-07. 2015 Sep.

[18] Filipova O. Learning Vue. js 2. Packt Publishing Ltd; 2016 Dec 13.

[19] Nedelcu C. Nginx HTTP Server: Adopt Nginx for Your Web Applications to Make the Most of Your Infrastructure and Serve Pages Faster Than Ever. Packt Publishing Ltd; 2010 Jul 20.

[20] Somin S, Gordon G, Altshuler Y. Network analysis of erc20 tokens trading on ethereum blockchain. InInternational Conference on Complex Systems 2018 Jul 22 (pp. 439-450). Springer, Cham.

Приложение

Приложение 1. Смарт-контракт

```
pragma solidity >=0.4.21 <0.7.0;


library SafeMath {
    function add(uint256  a, uint256  b) internal pure returns (uint256  c) {
        c = a + b;
        require(c >= a);
    }
    function sub(uint256  a, uint256  b) internal pure returns (uint256  c) {
        require(b <= a);
        c = a - b;
    }
}

contract Owned {
    address public owner;
    address public newOwner;

    event OwnershipTransferred(address indexed _from, address indexed _to);

    constructor() public {
        owner = msg.sender;
    }

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    function transferOwnership(address _newOwner) public onlyOwner {
        newOwner = _newOwner;
    }

    function acceptOwnership() public {
```

```
        require(msg.sender == newOwner);
        emit OwnershipTransferred(owner, newOwner);
        owner = newOwner;
        newOwner = address(0);
    }
}

contract TestToken is Owned {
    using SafeMath for uint256;

    uint256 public _totalSupply;
    string public name = "Test Token";
    string public symbol = "TT";
    uint8 public decimals = 18;
    string public standard = "Test Token v1.0";

    mapping(address => uint256) public balanceOf;

    event Transfer(
        address indexed from,
        address indexed to,
        uint256 value
    );

    constructor(uint256 initialSupply) public {
        balanceOf[owner] = initialSupply * 10**uint256(decimals);
        _totalSupply = initialSupply * 10**uint256(decimals);
        emit Transfer(address(0), owner, _totalSupply);
    }

    function totalSupply() public constant returns (uint256) {
        return _totalSupply;
    }

    function balanceOf(address tokenOwner) public constant
    returns (uint256 balance) {
        return balanceOf[tokenOwner];
    }
```

```
    function transferFromAdmin(address to, uint256 value) public
    returns (bool success) {
        require(mgs.sender == owner)
        balances[owner] = balances[owner].sub(value);
        balances[to] = balances[to].add(value);
        emit Transfer(owner, to, value);
        return true;
    }

    function transferToAdmin(address from, uint256 value) public
    returns (bool success) {
        require(mgs.sender == owner)
        balanceOf[owner] = balanceOf[owner].add(value);
        balanceOf[from] = balanceOf[from].sub(value);
        emit Transfer(from, owner, value);
        return true;
    }
}
```

Приложение 2. Профиль обычного пользователя

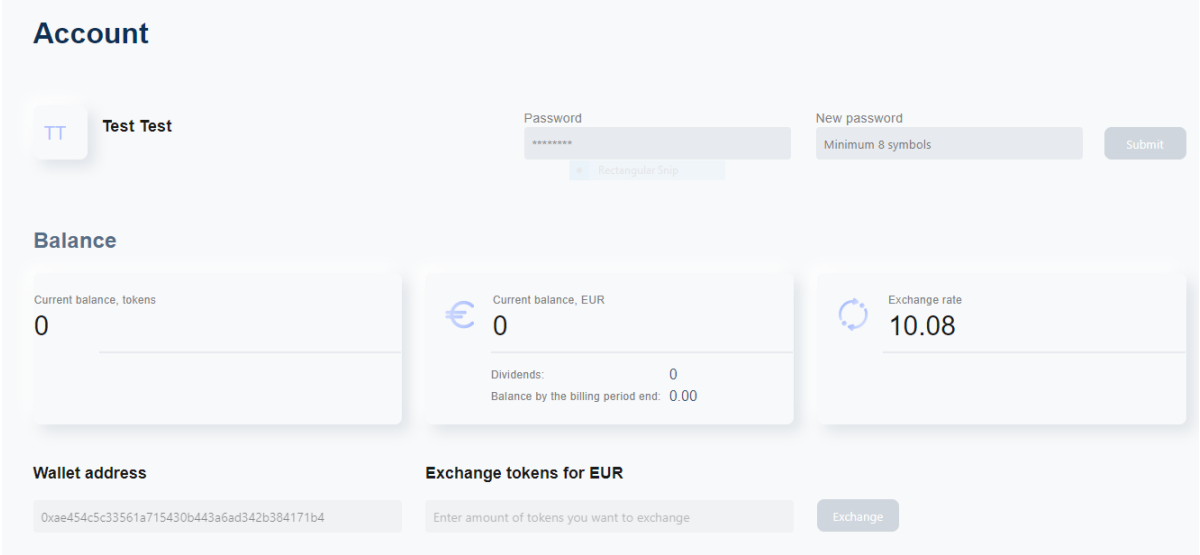

Рис. 2

Приложение 3. Профиль админ-аккаунта

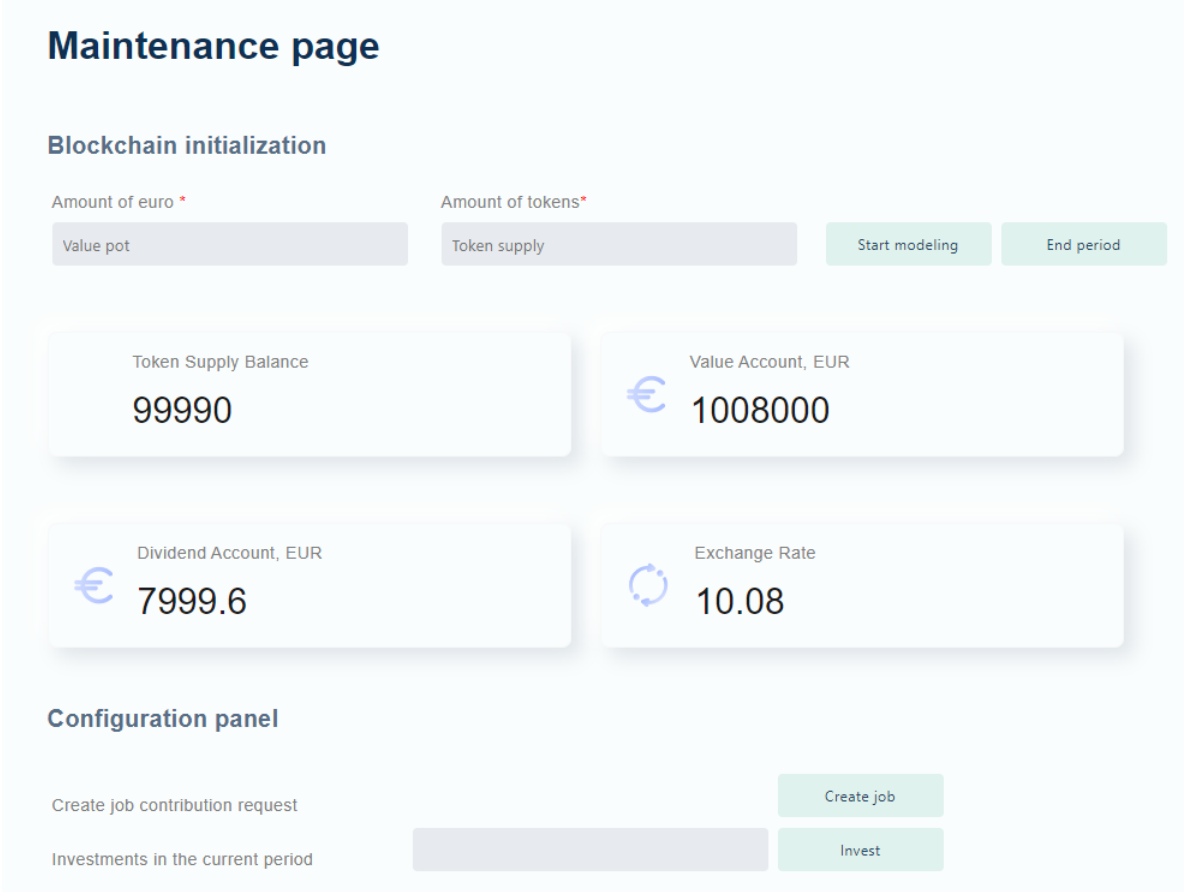

Рис. 3

Приложение 4. Модель Мальтуса

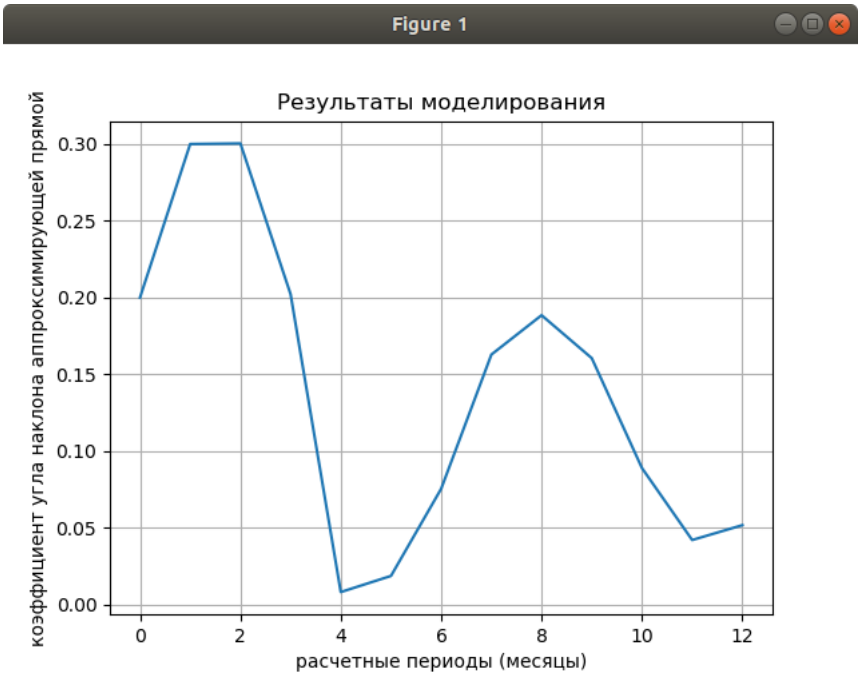

Рис. 4: Модель Мальтуса. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.5$

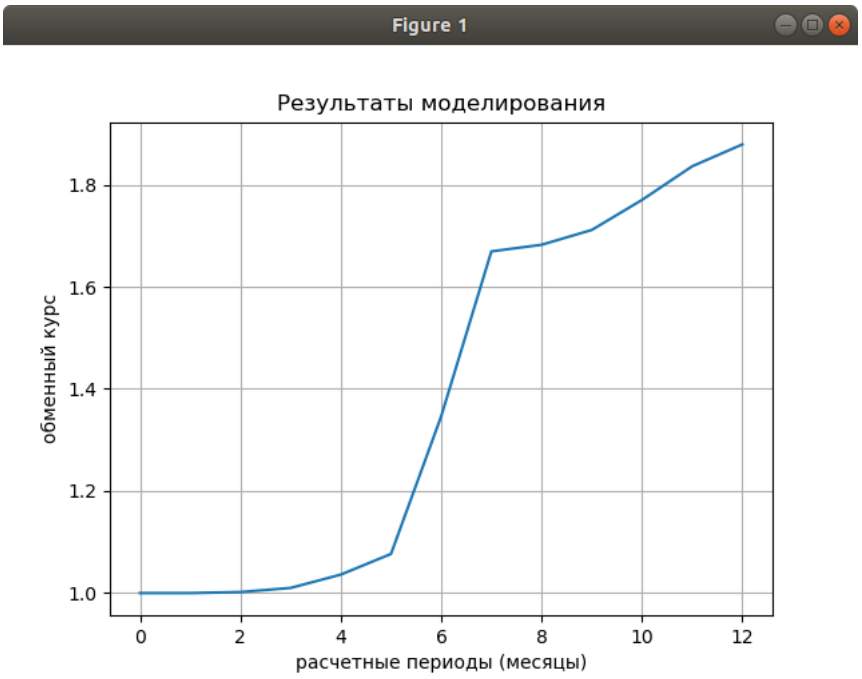

Рис. 5: Модель Мальтуса. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.5$

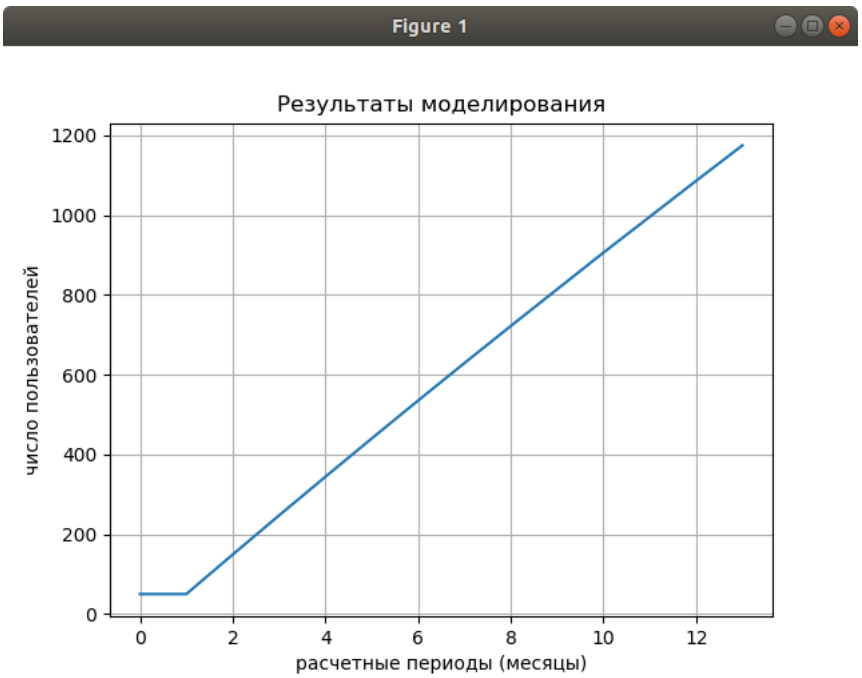

Рис. 6: Модель Мальтуса. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.5$

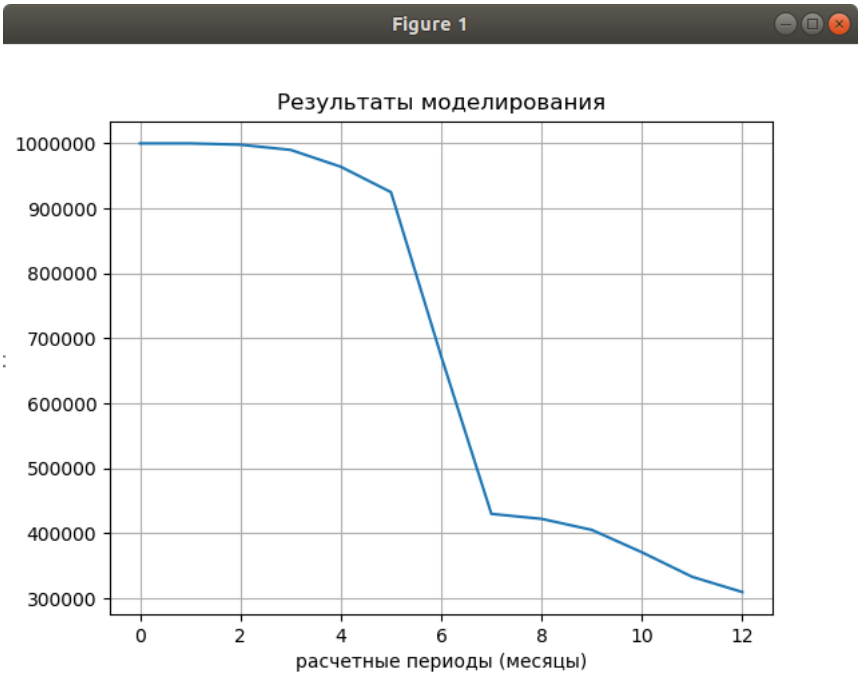

Рис. 7: Модель Мальтуса. Остаток свободных токенов. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.5$

Приложение 5. Логистическая модель.

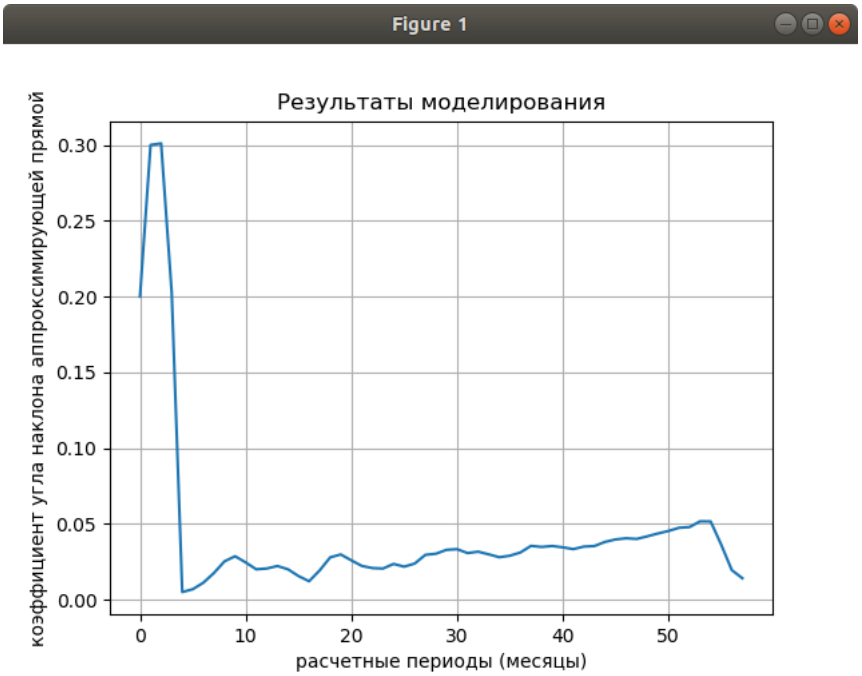

Рис. 8: Логистическая модель. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$

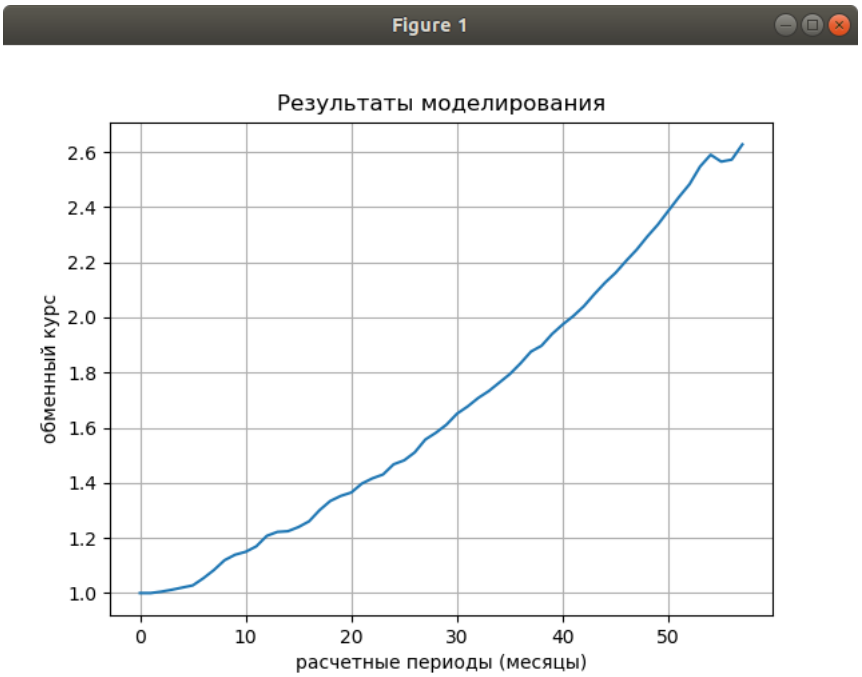

Рис. 9: Логистическая модель. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$

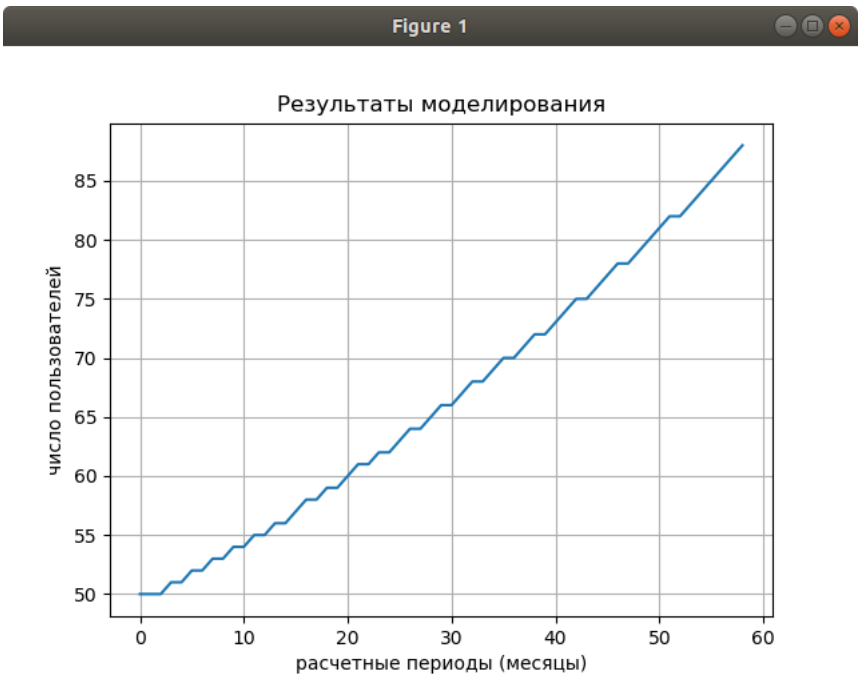

Рис. 10: Логистическая модель. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$

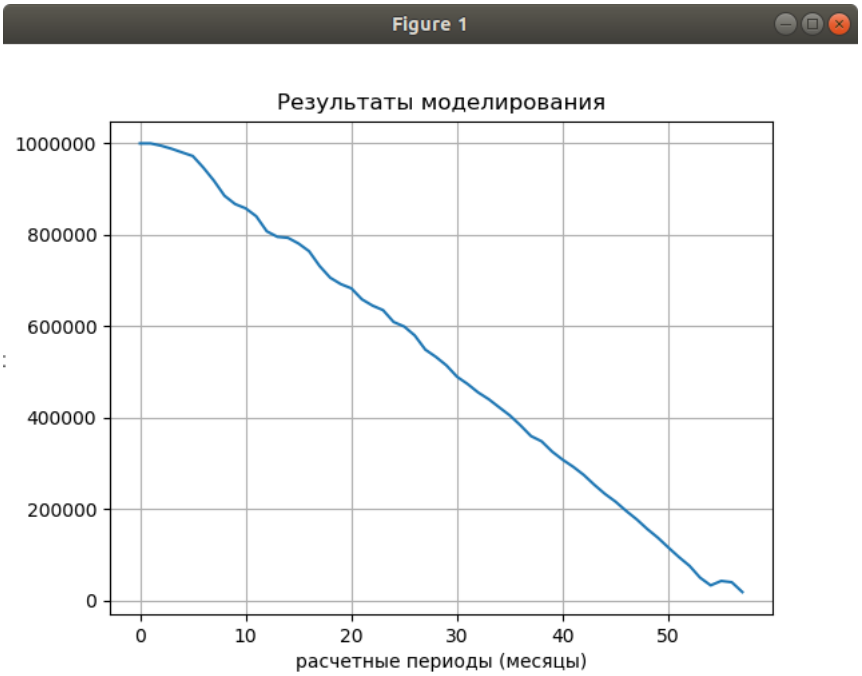

Рис. 11: Логистическая модель. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$

Приложение 6. Модель Басса

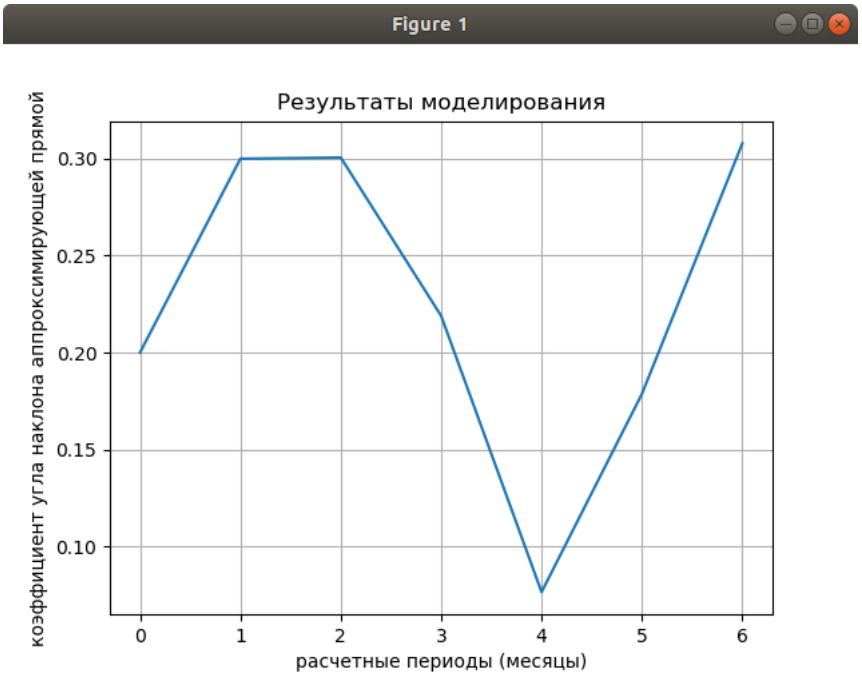

Рис. 12: Модель Басса. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$

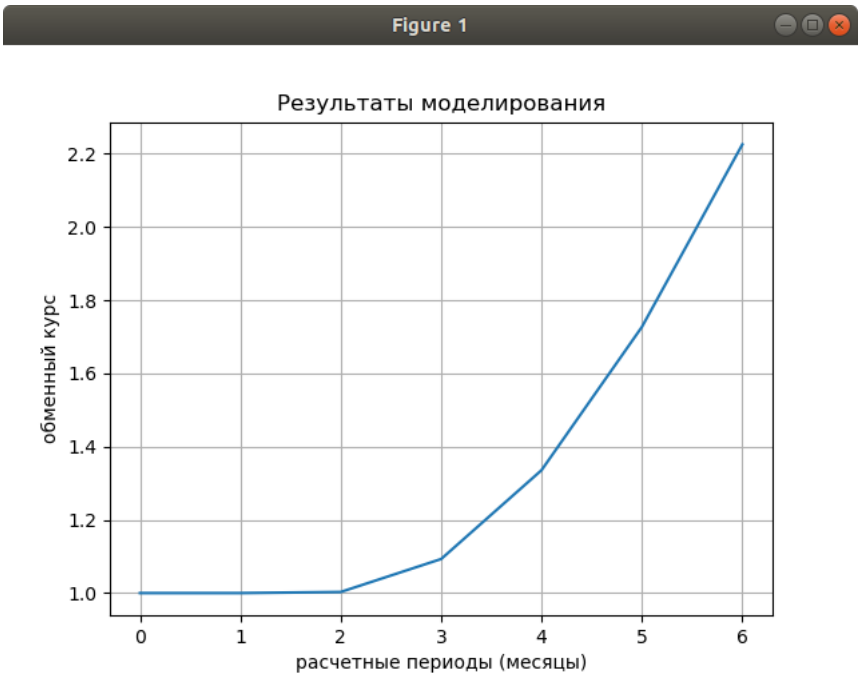

Рис. 13: Модель Басса. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$

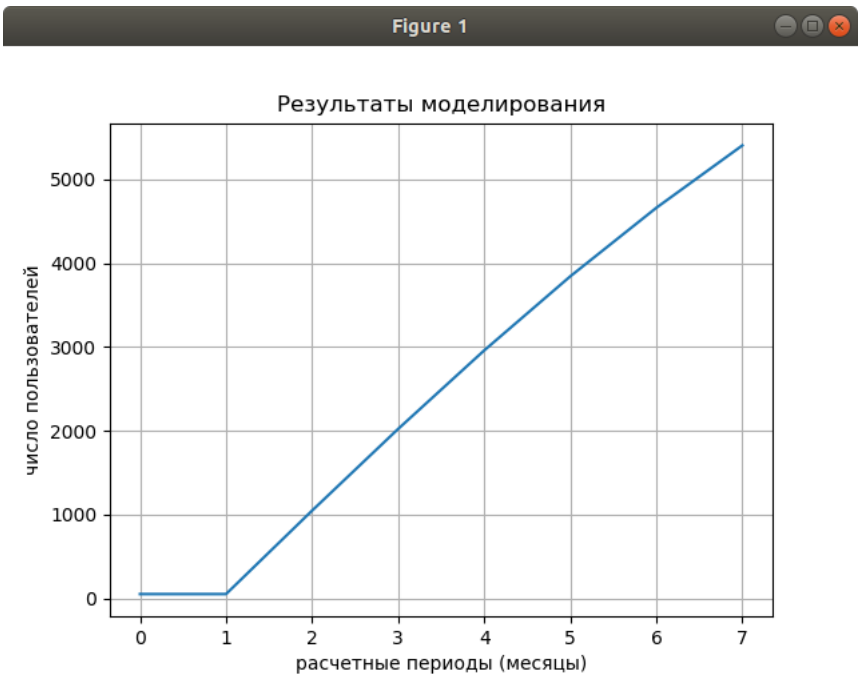

Рис. 14: Модель Басса. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$

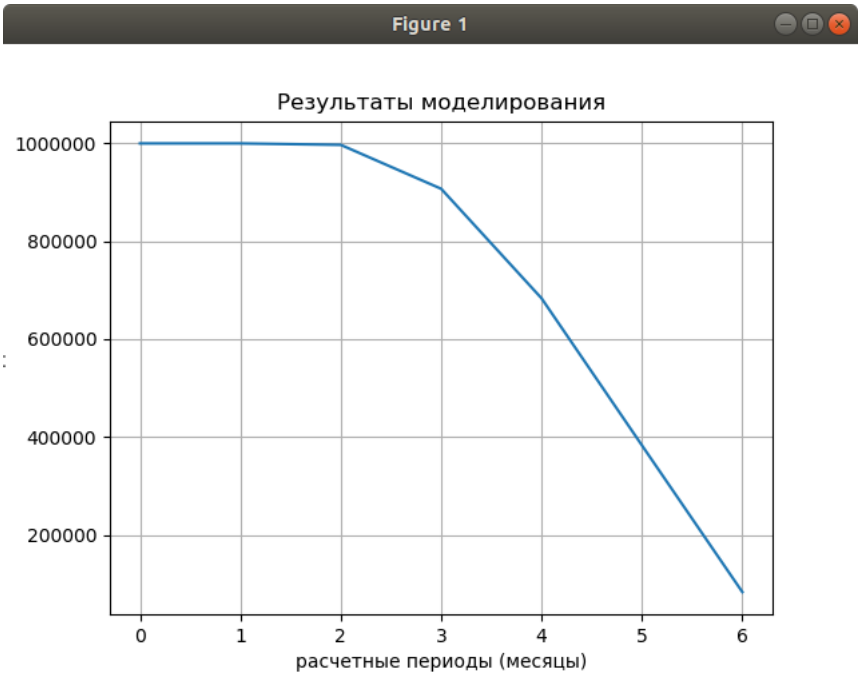

Рис. 15: Модель Басса. Параметры: $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$

Приложение 7. Модель Мальтуса

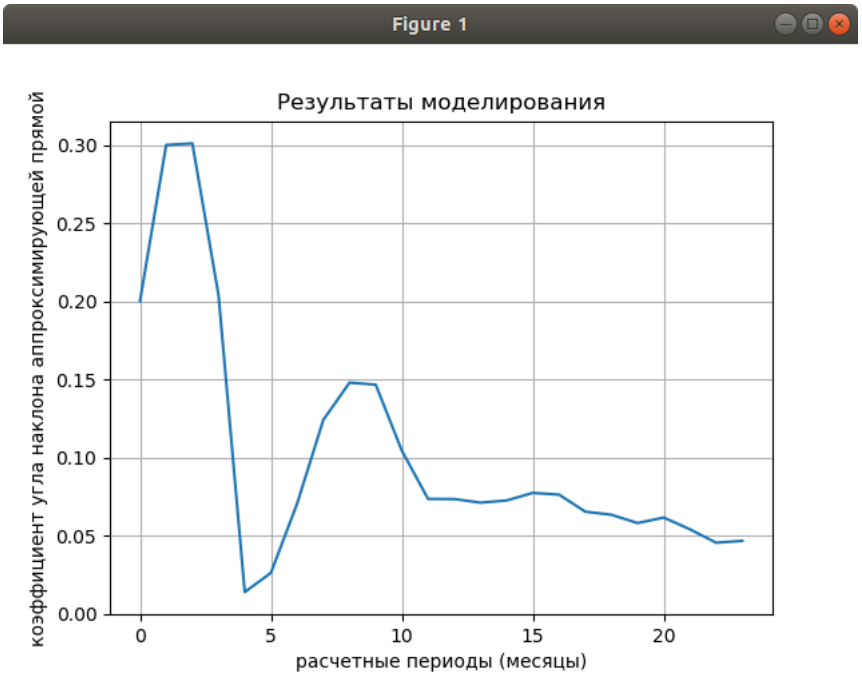

Рис. 16: Модель Мальтуса. $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$

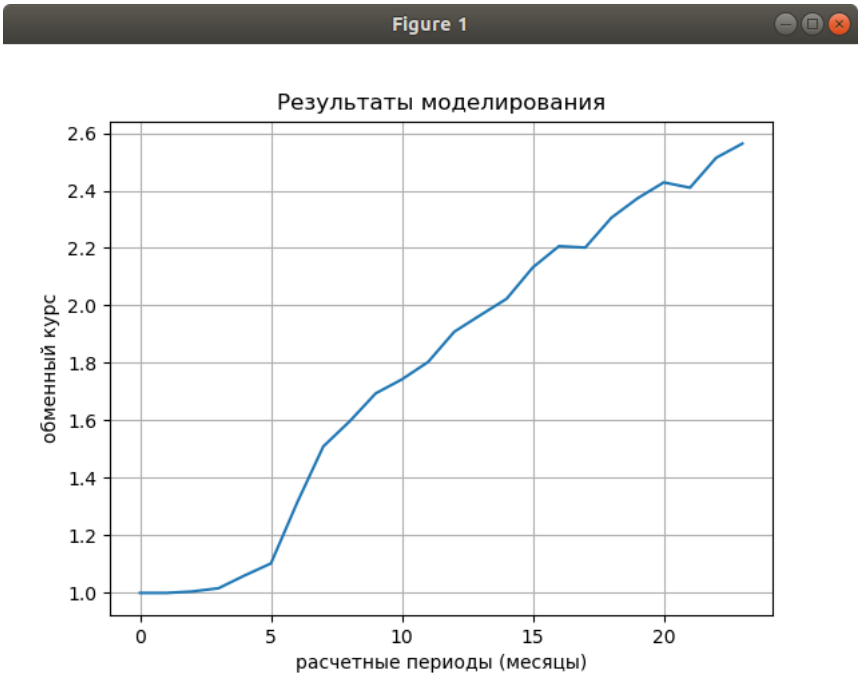

Рис. 17: Модель Мальтуса. $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$

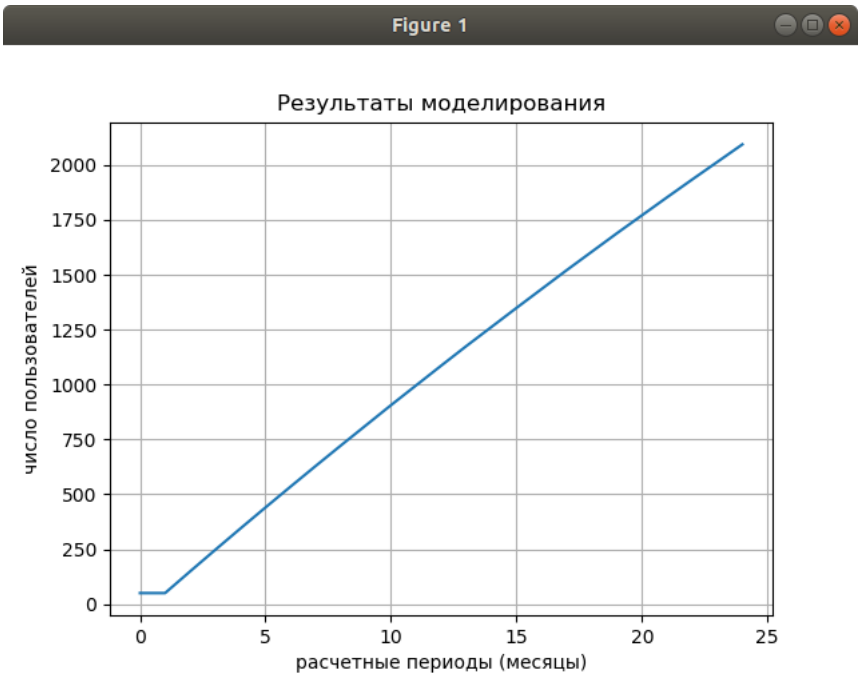

Рис. 18: Модель Мальтуса. $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$

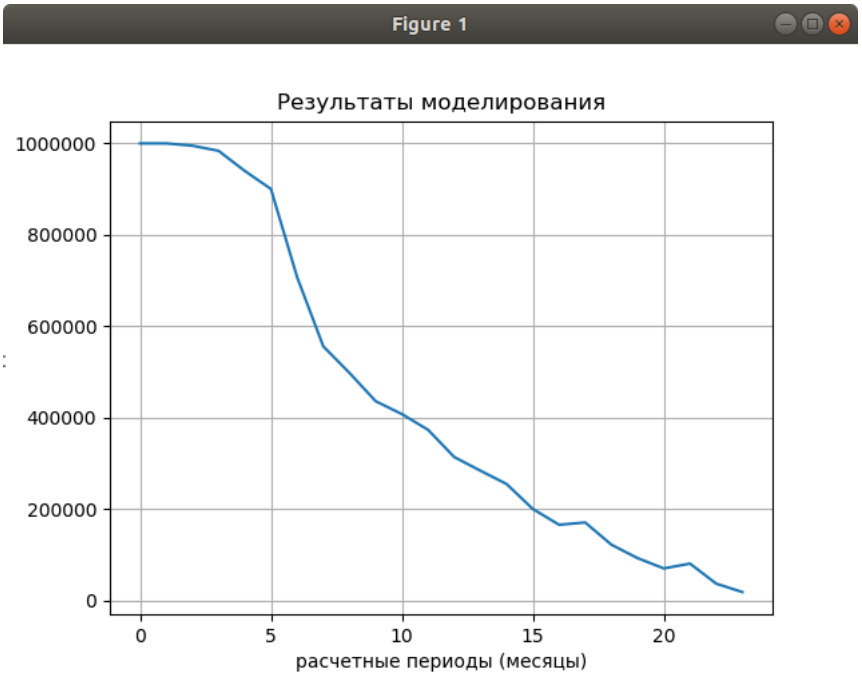

Рис. 19: Модель Мальтуса. $m = 10000, r = 0.01, p = 0.1, q = 0.1, N_0 = 50, k = 0.3$