Санкт-Петербургский государственный университет

Глушков Егор Александрович

Выпускная квалификационная работа

Адаптация консенсуса и экосистемы для фреймворка Crowdfunding.BGX

Уровень образования бакалавриат

Направление 02.03.02

«Фундаментальная информатика и информационные технологии»

Основная образовательная программа СВ.5003.2016

«Программирование и информационные технологии»

Научный руководитель:

доктор физ.-мат. наук, профессор

Богданов А.В.

Рецензент:

кандидат физ.-мат. наук, доцент

Корхов В.В.

Санкт-Петербург

2020

Содержание

Введение

Цифровизация современной экономики позволяет рассматривать практически любую организацию как информационную фабрику, создающую добавленную стоимость благодаря информационному обмену, использующую такие инновационные технологии как, например, блокчейн, искусственный интеллект, большие данные.

Первое поколение блокчейн-систем[1] концентрировалось на вопросах криптовалюты. Такие сети были заведомо публичными: любой узел мог присоединиться к сети и попробовать выполнить транзакцию. В сети Ethereum, например, среднее время жизни одного узла составляет несколько минут [1]. Многие узлы образованы электронными кошельками, и такое поведение узлов типично для публичных сетей с акцентом на криптовалюты. Фактически, главная задача блокчейн-сетей — создать устойчивую среду доверия в условиях тотального недоверия: любой узел может быть источником злонамеренного поведения.

Корпоративные блокчейн-сети (частные или консорциум-базированные) не предполагают высокой динамики времени жизни узлов. При этом всё еще остаются вероятности взлома узлов, нарушения их уровня доступности и производительности (Service Level Agreement, SLA). Поэтому часто на такие сети распространяют все архитектурные решения, работающие в публичных сетях. Это определяется в основном специальными правилами согласования транзакций — механизмами консенсуса, например, Proof-of-Work (PoW), Proof-of-Stake и

[1] Вообще говоря, блокчейн является частным случаем реализации технологии распределённых реестров (Distributed ledger technology, DLT), однако для краткости здесь и далее будет использовано понятие «блокчейн», хотя многое из сказанного можно обобщать и до DLT

прочими. Более подробное описание подобных консенсус систем будет рассмотрено в параграфе 1.1. Платой за надежность (устойчивость к атакам) упомянутых механизмов консенсуса является снижение производительности сети, её тяжеловесность, необходимость специальных методов настройки.

В блокчейн-системах, ориентированных на корпоративный рынок, вертикальная и горизонтальная интеграции приводят к объединению более слабых игроков вокруг сильных, к обмену данными между надежными и оснащенными системами и более легкими узлами. Одним из примеров подобных систем являются краудфандинг в частности и инвестиционные платформы в более общем случае. Технологии распределенных реестров в инвестиционных платформах позволяют обеспечить прозрачность и интеграцию всех участников. Главными вопросами в таких системах являются риск и доверие к различным участникам. В такой ситуации можно учесть вес каждого узла в части его надежности и адаптировать механизм консенсуса в части объема проверок. Такая модернизация позволит значительно увеличить производительность распределенных вычислений и выравнять вычислительную нагрузку за счет перехода от модели «полного недоверия» к учёту вычислительной репутации узлов, а также осуществит привязку исторических данных к текущему потоку транзакций.

В качестве примера системы распределенного реестра в данной работе выступает децентрализованная платформа DGT (BGX)[2] [2]. Более подробное описание организации её работы представлено в параграфе 1.2. В настоящей работе представлена попытка спроектировать функцию доверия к узлу (Trust-функцию) и включить ее в общий механизм консенсуса. В качестве примера алгоритма консенсуса выступает F-BFT, основанный на двух других

[2] На момент согласования темы данный программный продукт имел название BGX, однако с выходом новой версии v0.5 был произведен ребрендинг, текущее название платформы — DGT. Далее в работе будет использовано новое название платформы

алгоритмах: pBFT и Raft, которым будет уделено особое внимание в параграфе 1.1. Выбор механизма консенсуса не случаен: именно F-BFT консенсус лежит в основе платформы DGT. Каждый из представленных алгоритмов обладает свойством немедленной (абсолютной) завершенности [3], основан на голосовании внутри группы узлов с лидером, однако в отличие от других F-BFT адаптирован для использования в неоднoранговых сетях с произвольной топологией.

Применение функции доверия (Trust Score) к узлу позволит адаптировать механизм консенсуса для распределенных сетей вида «экосистема», когда основная тяжесть ведения инфраструктуры ложится на системообразующие организации, а основным источником риска являются присоединившиеся организации. Типовым примером такой структуры является SberX — экосистема Сбербанка [4].

Постановка задачи

Целью данной работы является построение функции доверия к узлу (Trust Score) в задаче голосования для алгоритма консенсуса F-BFT и адаптация архитектуры платформы DGT, в основе которой лежит описанный алгоритм консенсуса, для работы данной функции.

Для достижения обозначенной цели необходимо выполнить следующие задачи:

1) анализ существующих консенсус систем;
2) описание организации работы платформы DGT;
3) построение функции доверия на основе имеющихся показателей работы узлов;
4) встраивание построенной функции доверия в систему DGT.

Обзор литературы

Оценивание значимости, доверия, репутации узлов в децентрализованной сети — актуальная тема, которая находит отражение во множестве научных работ.

В статье R. Di Pietro et al. «A blockchain-based trust system for the Internet of Things» [5] приводится построение распределенной модели доверия для сети Интернета вещей (Internet of Things, IoT). В качестве типичного для IoT сценария работы представляется схема взаимодействия «поставщик-потребитель». Для устранения посредников и уменьшения временных затрат на поставку какой-либо услуги, а также для возможности потреблять эту услугу «здесь и сейчас» с оплатой после использования вводится дополнительная цепочка обязательств. Также вводится специальный протокол, позволяющий быстро и безопасно осуществлять взаимодействие между участниками — предоставление услуг или отказ — с использованием введенной блокчейн-структуры, хранящей все предыдущие обязательства.

S. Tamang в работе «Decentralized Reputation Model and Trust Framework Blockchain and Smart contracts» [6] показывает использование технологии блокчейн как интерактивной системы, гарантирующей неизменный рейтинг доверия участников. Каждый участник строит список узлов (других участников), которым он доверяет. Полученные списки агрегируются на основе приведенной в работе метрики. Для подсчета, хранения и распространения рейтинга доверия используются смарт-контракты и сеть Ethereum. Приведенные в работе результаты показывают устойчивость предложенного метода к различным моделям угроз, а также обнаружение злонамеренного поведения участников сети.

Нередко в работах используются различные математические методы. Так, в [7] приводится метод PageRank, использующий веса ребер графа, которые представляют некоторые зависимости узлов друг от друга, для подсчета важности узлов. В [8] также используется граф и метрики, связанные с весами ребер, для подсчета предвзятости и престижа узлов в сети.

N. Griffiths в [9] представляет общее доверие как некоторую многомерную величину, состоящую из успеха, стоимости, своевременности и качества. Каждое из этих измерений представляет собой вещественное число, отражающее некую сравнительную ценность без сильного семантического значения. Измерения обновляются на основе эвристического правила, базирующегося на непосредственном опыте отдельного агента. Полученные значения агрегируются специальной функцией «значение производительности», результат которой затем используется для выбора поставщиков.

В работе S. Kamvar et al. «The EigenTrust Algorithm for Reputation Management in P2P Networks» [10] кроме вводимых метрик, основанных на данных о доверии узлов друг другу, заслуживает внимания и список требований к репутационной системе одноранговой сети:

1. Саморегулируемость сети (контроль со стороны самих узлов сети, а не центрального органа).
2. Анонимность участников достигается за счет выдачи непрозрачных идентификаторов в противовес публичным IP-адресам.
3. Новички не получают преимуществ (предотвращение атак Сивиллы с созданием множества сущностей, управляемых одним участником).
4. Минимальные накладные расходы на вычисления и хранение.

5. Устойчивость к вредоносным коллективам узлов, действующих как единое целое.

Работа [11] представляет собой описание COTI — децентрализованной высокопроизводительной криптовалютной экосистемы. В архитектуре COTI заложены разные инновационные решения, например, в качестве реестра используется дерево (Directed Acyclic Graph, DAG), для нахождения злонамеренных узлов используется машинное обучение и служба арбитров. Однако отдельного внимания заслуживает формула подсчета значения доверия узлов, представленная на рисунке 1. Стоит заметить, что значение доверия представляет собой сумму из переменных, характеризующих некоторые показатели работы узла, убывающих экспоненциально с течением времени. Кроме того, при подсчете некоторые показатели умножаются на весовые коэффициенты, зависящие от времени, что позволяет придать более поздним событиям бóльший вес.

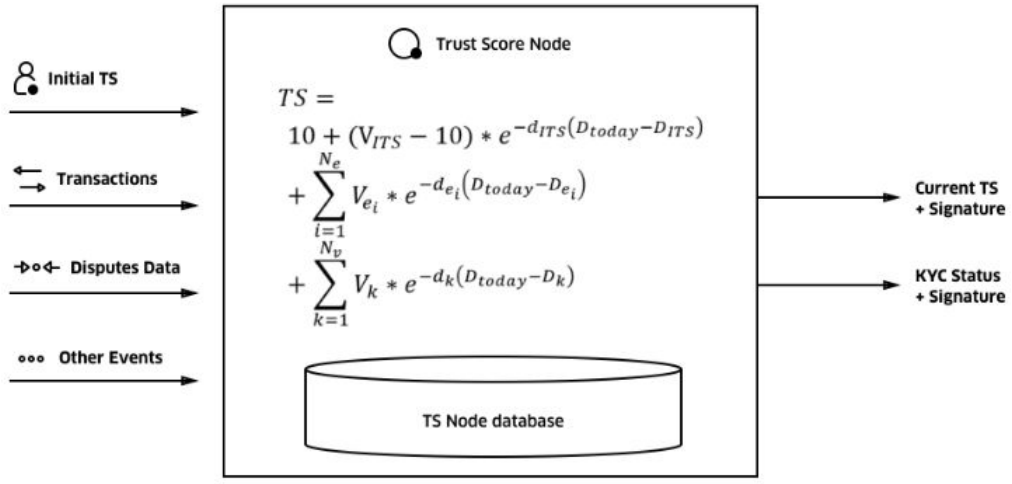

Рисунок 1 — Процесс обновления значения доверия узла в системе COTI

Источник: COTI Technical White Paper, V.4.0 [11, с. 22]

Таким образом, в обзоре литературы были приведены наиболее интересные работы, представляющие разные подходы по построению значения доверия или важности узлов. Кроме того, некоторые решения могли бы быть полезны при достижении цели текущей работы в рамках построения функции доверия для консенсуса F-BFT и платформы DGT.

Глава 1. Алгоритмы консенсуса и организация работы платформы DGT

1.1. Анализ консенсус систем

Протокол[3] консенсуса — это механизм или набор правил, с помощью которых сеть приходит к согласию, и именно каким образом узлы согласовывают валидацию транзакций [3]. Протокол консенсуса является основой технологии распределенного реестра (distributed ledger technology, DLT).

Консенсус сети состоит в том, что все узлы содержат один и тот же распределенный реестр. В традиционной архитектуре программного обеспечения сложно говорить о консенсусе, так как обычно существует центральный сервер, контролирующий работу всей сети. Однако в распределенной сети на основе DLT каждый узел является как клиентом, так и сервером, и ему необходимо обмениваться информацией с другими узлами для достижения консенсуса. Иногда некоторые узлы могут быть недоступны, не иметь доступа для передачи информации по сети (быть оффлайн) или иметь значительную задержку при отправке или получении сообщений. Кроме этого, часть узлов может носить вредоносный характер, имея цель нарушить работу сети. Потому хорошо спроектированный алгоритм консенсуса должен сводить к минимуму наносимый таким поведением вред, чтобы обеспечить финальный результат, согласованный добросовестными узлами сети.

[3] Вместо слова «протокол» в этой работе в этом же значении могут быть использованы слова «алгоритм», «механизм», «система», когда речь идет о консенсусе

Протокол консенсуса должен быть адаптирован под конкретные требования приложения, его сценарии работы. Распределенные реестры могут быть классифицированы по возможности участия в работе сети:

- публичный реестр — открытая сеть, в которой каждый узел принимает участие в работе сети, в том числе может читать и дополнять реестр;
- консорциум-базированный реестр — состав его участников ограничен и заранее определен (хотя и существуют механизмы изменения числа узлов в такой сети, но не каждый узел может стать участником), при этом сам реестр может быть публичным для чтения; участники имеют ограниченные права на его изменение;
- приватный реестр — у каждого участника свой уровень доступа, сам реестр доступен только для участников.

Также распределенные реестры классифицируются по возможности верификации транзакций:

- permissionless — инклюзивный реестр, новые транзакции может регистрировать любой участник;
- permissioned — эксклюзивный, только строго ограниченная группа участников может вносить изменения в реестр.

В распределенных системах не существует идеального протокола консенсуса: он должен находить компромисс между согласованностью, доступностью и отказоустойчивостью к разделению, жертвуя чем-то одним (CAP теорема). Кроме того, консенсус должен решать Задачу византийских генералов, когда существуют вредоносные узлы, намеренно подрывающие работу системы.

Proof-of-Work (PoW) — доказательство работы — один из первых алгоритмов консенсуса, нашедший применение в криптовалюте Биткоин, ставшей революцией в сфере блокчейн/DLT. PoW выбирает один узел для создания нового блока в каждом раунде консенсуса с помощью конкуренции

узлов в вычислении некоторого хэша. Узел, который первым решит эту «криптографическую головоломку», имеет право на создание нового блока и его присоединение к самой длинной цепочке из существующих. Соответственно, для увеличения шансов получения нового блока требуется бóльшая вычислительная мощность. Потенциально существует ненулевая вероятность, что злоумышленник может переписать блок или иметь скрытую более длинную цепочку, и тогда именно его версия блокчейна будет верной, что является угрозой для остальных пользователей. Однако с ростом числа пользователей такая вероятность стремится к нулю, так как требует огромных вычислительных затрат: больше половины всех вычислительных мощностей (что известно как «атака 51%»). Таким образом, в PoW завершенность (finality) носит лишь вероятностный характер, то есть транзакция может быть отменена, но с течением времени вероятность отмены значительно уменьшается, хотя и остается ненулевой. Следовательно, PoW выбирает доступность и отказоустойчивость к разделению; согласованность обеспечивается лишь в конечном счете (eventual consistency). Использование PoW ведет к огромным тратам электроэнергии и по сути бессмысленным вычислениям хэш-функций, вычисленные результаты которых нигде более не используются.

Proof-of-Stake (PoS) — доказательство доли владения — обеспечивает бóльшие шансы на право создания нового блока тем, кто имеет бóльшую долю имеющейся криптовалюты от общего её количества (в отличие от большей доли вычислительных мощностей в PoW). В PoS также применяется решение «криптографической головоломки», однако её вычисление менее затратно и зависит от баланса валюты на счете. Это позволяет экономить значительные вычислительные и энергоресурсы, однако такая схема ведёт к увеличению доли уже наиболее обеспеченных участников, что сказывается на всё большей централизации сети. Кроме того, существует вероятность

сговора участников, обладающими большими долями, что даёт им возможность навязывать свои условия вне зависимости от мнения большого числа участников.

Delegated Proof-of-Stake (DPoS) — делегированное доказательство доли владения — позволяет узлам, имеющих долю, голосовать за выбор создателей (верификаторов) блоков, т.е. делегировать право создания блоков другим узлам, тем самым снижая собственные вычислительные затраты до нуля. Вес голоса участника при делегировании пропорционален той доле, что он имеет. Делегаты получают некоторую награду за каждую валидированную транзакции. Существует несколько вариантов реализации данного механизма консенсуса: в одном число делегатов фиксировано, поэтому при подсчете голосов с учетом их весов выбираются первые N участников, что может вести к централизации с ростом сети; в другом число делегатов N может меняться и зависит от того, сколько делегатов с наибольшими набранными голосами необходимо, чтобы их общий вес был больше половины от всего числа голосов. Далее узлы из числа выбранных делегатов создают блок в циклическом порядке (round-robin fashion), что позволяет балансировать нагрузку и уменьшает предопределенность выбора узла, что в свою очередь уменьшает вероятность злонамеренного поведения. В случае неудачи в создании блока за определенное время выбирается следующий узел. Как только все делегаты сделают свой ход, список узлов в выбранной группе также перемешивается. Таким образом, DPoS заметно более вычислительно- и энергоэффективный по сравнению с PoW и PoS. Также этот алгоритм предлагает больше влияния узлам, которые имеют не такую большую долю, как другие узлы. Кроме того, подобная схема работы предотвращает возможность атаки двойной траты (Double-spending) ввиду постоянной смены делегата-создателя после завершения производства блока, что отличает его от PoW, где сеть на начальном этапе при малом числе

участников может допускать проведение подобной атаки. К слову, PoS также достаточно защищен от двойной траты, потому что в случае неправомерного поведения узел-валидатор теряет свою долю (так как его доля разделяется на две части). PoS и DPoS имеют вероятностную завершенность, то есть действующая на данный момент цепочка блоков может быть изменена, однако вероятность этого мала. Тем не менее, сеть останется доступной и продолжит работу, даже если она позволяет проводить неточные транзакции.

В противовес механизмам, основанным на вычислительной мощности или доле владения, существуют алгоритмы консенсуса, базирующиеся на голосованиях [12]. Последние делятся на две категории: BFT (Byzantine Fault Tolerance — отказоустойчивость к византийским ошибкам) и CFT (Crash Fault Tolerance — отказоустойчивость к падениям узлов). Если алгоритмы второй категории не чувствительны к падениям узлов по чисто техническим причинам, то первые также работают и при «византийском» поведении некоторых участников, когда те ведут себя злонамеренно и пытаются ввести в заблуждение другие узлы.

Одной из простейших реализаций BFT является алгоритм pBFT (Practical BFT) [13]. Данный алгоритм обладает низкой вычислительной сложностью и высокой практичностью в распределенных системах. В данном протоколе узлы выбирают узел-лидер в рамках циклического механизма (round-robin). Лидер является таковым до тех пор, пока сам не выйдет из строя, после чего выбирается новый лидерский узел. Корректная работа алгоритма лежит в предположении, что все честно работающие узлы имеет одинаковый реестр. Число злонамеренных и выведенных из строя узлов должно быть строго меньше ⅓ от общего количества участников. Таким образом, чем больше узлов в сети, тем менее вероятно, что треть или более узлов будут мошенническими или выведенными из строя. Сам механизм содержит пять фаз работы:

1) запрос (request);
2) предварительная готовность (pre-prepare);
3) готовность (prepare);
4) фиксация/утверждение (commit);
5) ответ (reply).

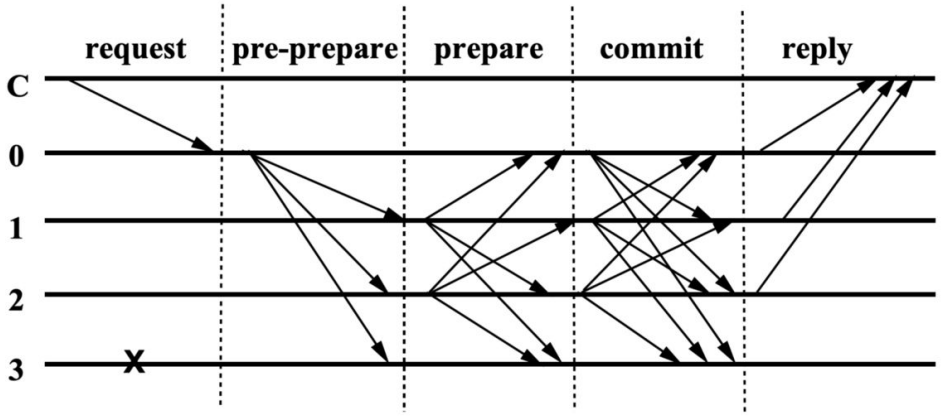

Рисунок 2 — Процесс принятия pBFT консенсуса

*Источник: https://miro.medium.com/max/1400/0*F9jozhrlqOy4XYlz*

Процесс генерации блока в pBFT консенсусе (рисунок 2) может быть описан следующим образом. Клиент отправляет запрос узлу, который был выбран лидером, для выполнения транзакции. Лидер собирает запросы на транзакции и группирует их в блок. Затем блок рассылается всем остальным узлам в сети. Каждый узел верифицирует транзакции в блоке, создает другой блок с действительными (valid) транзакциями, вычисляет хэш-функцию от этого блока и рассылает это другим узлам. Узел ожидает ответа от ⅔ узлов с тем же хэшем, после чего блок добавляется в реестр этого узла.

Такой механизм обеспечивает поддержание одинакового для всех узлов реестра и строгой согласованности (обращаясь к упомянутой выше CAP теореме), что позволяет классифицировать pBFT как алгоритм абсолютной завершенности, когда однажды добавленный в цепочке блок уже не может быть недействительным. Выбор лидерского узла обозначает некоторый уход от полной децентрализации, но такой подход является отличным решением для консорциум-базированных и приватных сетей. Большое число пересылок

сообщений показывает pBFT как алгоритм, эффективный в системах с малой задержкой, но чувствительный к числу участников.

Примером CFT алгоритма может служить протокол Raft [12, 14]. Проблема издержек связи в виде большого числа пересылок при pBFT устраняется в алгоритме Raft в предположении, что коммуникация в сети происходит только через лидерский узел. Как и любой CFT алгоритм, Raft обеспечивает безопасность только в ситуациях возможных сбоев узлов без защиты от злонамеренных атак. Правильная работа механизма обеспечивается при более чем 50% рабочих узлов. Все узлы могут быть в одной из следующих ролей: лидер (только один), кандидат в лидеры (в случае отсутствия такового) или последователь. Именно лидер отвечает за генерацию записей реестра при получении транзакций от клиентов. Весь процесс работы сети может быть представлен как три этапа: избрание лидера, репликация реестра и выполнение транзакции.

1. Избрание лидера.

   Время в алгоритме Raft представляет собой некоторые промежутки разной длины, известные как сроки (terms). Срок представляет из себя натуральное число, которое увеличивается на 1 в случае выбора нового лидера. Когда последователь перестаёт получать сообщения от лидера в течение некоторого периода времени (timeout), то он предполагает, что в сети нет лидера и выдвигает свою кандидатуру, увеличивая срок на 1 и рассылая другим участникам запрос на избрание лидера и свой срок. Другие узлы-последователи отвечают на запрос об утверждении кандидатуры. Результатом может быть одна из трех возможных ситуаций:

   1) кандидат получил голоса от большинства узлов сети, он признается победителем и должен отправить другим узлам сообщение о начале своего срока в качестве лидера;

2) пока узел ожидает ответа от других узлов, к нему может прийти сообщение от другого кандидата, претендующего на роль лидера. Если срок (term index) другого кандидата меньше, чем у рассматриваемого узла, тогда последний отправляет сообщения другим узлам об отклонении своей кандидатуры; если срок другого кандидата больше, чем текущий инкрементированный срок рассматриваемого узла, то это значит, что новый лидер был избран раньше, чем рассматриваемый узел, и снова переходит в состояние последователя, запоминая нового лидера и значение его срока. В случае совпадения сроков узел продолжает ожидать;
3) при наличии нескольких кандидатов ни один из них может не получить большинство голосов. В таком случае каждый из кандидатов ожидает некоторое время (timeout) и заново выдвигает свою кандидатуру, также увеличивая срок на 1. Последователи голосуют за тот узел, от которого получили запрос на лидерство раньше других. Во избежание повторения совпадения значений сроков рекомендуется предоставлять случайное время ожидания для кандидатов.

2. Репликация реестра.

   Выбранный лидер обрабатывает приходящие от клиентов транзакции, проверяя их (процесс валидации) и присваивая им индексы для поддержания порядка. Такая транзакция добавляется в журнал и отправляется на голосование всем последователям в сети. Узлы также проверяют полученные транзакции и отправляют лидеру своё решение о принятии или отказе.

3. Выполнение транзакции.

После получения положительных ответов от большинства последователей лидер выполняет транзакцию и уведомляет остальные узлы о её выполнении.

Raft консенсус не работает со злонамеренным поведением узлов. Если такое поведение наблюдается у лидера или у большинства узлов, то реестр становится недействительным. Так как только один узел может дополнять реестр (но не изменять предыдущие записи), то Raft не подвержен атаке двойной траты. Raft предпочтителен для небольших сетей, так как с ростом числа участников экспоненциально растет число сообщений, что определяет данный консенсус как применимый в приватных и консорциум-базированных сетях [15]. Для нормальной работы сети требуется, чтобы время доставки сообщений от узлов к лидеру и наоборот было намного меньше времени голосования, которое в свою очередь должно быть намного меньше времени стабильной работы узлов в сети (то есть времени работы без отказов). Данный алгоритм консенсуса обладает свойством абсолютной завершенности.

Federated Byzantine Fault Tolerance (F-BFT) — федеративный BFT — реализуется в приватных и консорциум-базированных сетях, где вся сеть представляет собой набор кластеров (некоторых групп узлов) [16, 17]. F-BFT консенсус основан на алгоритмах pBFT и Raft, общая его схема представляет собой голосование о действительности транзакции сначала внутри кластера, затем результаты проверяются вовне. Участники данной сети могут выполнять следующие роли:

1) инициатор — узел, инициирующий транзакцию (например, после запроса подключенного к этому узлу клиента);
2) лидер — узел, собирающий транзакции внутри кластера и подсчитывающий результаты голосования узлов; узлы внутри кластера ведут коммуникацию именно через лидера; смена лидера может

происходить как после определенного числа раундов голосований, так и в результате циклического выбора (round-robin), как в pBFT, притом в различных модификациях выбор лидера может быть не случаен, а с учетом некоторых показателей работы узла;

3) арбитр — узел вне кластера, проверяющий голосование внутри кластера и добавляющий транзакцию в реестр.

Сам процесс работы сети выглядит следующим образом. В сеть через узел-инициатор приходит транзакция, которая верифицируется (сверяется на предмет корректности формы) и после успешной проверки пересылается лидеру, который в свою очередь рассылает её всем узлам кластера. Участники сети проводят валидацию транзакции (соответствие формальным правилам, определенным для транзакций такого типа, например, на наличие денег на счете при переводе валюты) и подписывают результат работы, отправляя его обратно лидеру. Лидер подсчитывает результаты голосований, проверяет корректность подписей и отправляет транзакцию на проверку одному или нескольким арбитрам, находящимся вне кластера при числе голосов больше ⅔. Арбитры проверяют правильность голосования, подписи и саму транзакцию, после успешного завершения добавляют её в свой реестр и пересылают результаты лидеру, который распространяет принятую транзакцию уже внутри кластера. Каждый из узлов добавляет транзакцию в свой реестр.

Как видно из приведенного выше механизма работы, возможность записи лишь одним узлом защищает от атаки двойной траты, а проверка внешними узлами (арбитрами) — от злонамеренных действий лидера кластера или всего кластера в целом. Как и консенсусы pBFT и Raft, F-BFT имеет абсолютную завершенность: однажды принятая транзакция не может быть впредь отменена. Проблемы с масштабируемостью двух приведенных выше алгоритмов, на которых основан F-BFT, решаются кластерной

(федеративной) архитектурой сети и возможным использованием реестра, отличного от цепочки блоков (например, древовидный реестр — DAG, directed acyclic graph — открывающий возможности для шардирования, что будет затронуто в параграфе 2.1).

Таблица 1 — Сравнение основных механизмов консенсуса

Свойство	PoW	PoS/DPoS	pBFT	Raft	F-BFT
Завершенность	Вероятностная		Абсолютная		
Отказоустойчивость	50%	50%	33%	50%	33%
Энергозатратность	Высокая	Низкая	Незначительная		
Масштабируемость	Высокая		Низкая		Средняя
Приложения (виды сетей)	Публичные		Приватные и консорциум-базированные		

Таким образом, в данном параграфе были рассмотрены наиболее популярные алгоритмы консенсуса, в том числе pBFT и Raft, которые послужили основной для алгоритма F-BFT — ключевого элемента платформы DGT, рассматриваемой в данной работе. Некоторые сравнительные результаты представлены в таблице 1.

1.2. Описание организации работы платформы DGT

DGT (до ребрендинга BGX) — децентрализованная процессинговая платформа, построенная на типовых узлах, обменивающихся некоторыми сообщениями-транзакциями. Каждый узел представляет собой виртуальный суперкомпьютер, состоящий из подсистем, реализованных в виде отдельных виртуальных сервисов (контейнеров), и отождествляется с некоторой организацией, развернувшей данный узел и являющейся клиентом. Узлы образуют сеть, разбитую на кластеры (федерации), добавление в сеть определяется особыми «якорными» правилами, например, владелец узла

должен обладать определенной суммой на счете или иметь сертификат, чтобы стать участником сети. Подобное даёт возможность классифицировать сети применения DGT как консорциум-базированные или приватные. Данные о заданной топологии (схеме организации узлов) определяются особым модулем — процессором топологических транзакций, который прописывает связи и положение узлов непосредственно в распределенном реестре, хранящемся на каждом узле. В сети узлы могут быть в одной из трёх ролей по отношению к определенному кластеру: инициатор, лидер и арбитр. Взаимодействие между узлами, а также назначение ролей, определяется выбранным механизмом консенсуса. Платформа DGT использует алгоритм консенсуса F-BFT, специально адаптированный под условия использования платформы и её архитектуру.

Транзакции генерируются клиентами, которые взаимодействуют с узлами сети через API (программный интерфейс приложения). Перед отправкой клиент подписывает транзакцию с использованием системы приватных и публичных ключей и механизма эллиптической криптографии (ECDSA). Отправка транзакций происходит асинхронно, то есть результат не будет немедленным, однако клиент может узнать статус транзакции по её идентификатору, если она была успешно обработана системой. Каждая отправляемая клиентом транзакция имеет тело транзакции (её описание) и типовой заголовок, сообщающий назначение транзакции и адреса тех, кто и кому её отправляет.

В отличие от большой части систем распределенного реестра, использующих в своей основе цепочку блоков (блокчейн), DGT хранит транзакции в древовидной структуре DAG (directed acyclic graph — направленный ациклический граф). При добавлении транзакции в DAG её

заголовок расширяется: хэш[4] тела транзакции, цифровая подпись записывающего в реестр узла, хэш предыдущих транзакций и хэш всего набора данных. Таким образом, DAG представляет собой направленный граф, вершины которого — добавленные транзакции, дуги между ними — ссылки на предыдущие транзакции.

В реестре DAG поддерживаются разные семейства транзакций, обрабатываемые специальным процессором транзакций, имеющим свои правила валидации. Архитектура DGT даёт возможность писать собственные процессоры транзакций, что позволяет оперировать в платформе DGT, например, не только валютой или токенами, но и различными цифровыми ценностями, например, информацией о клиентах в сети, представляющей собой экосистему из компаний со смежной клиентской базой. Копии реестра хранятся на каждом узле и обновляются асинхронно за конечное время (так обеспечивается согласованность «в конечном счете»). Для оптимизации расходов на хранение и транспортные задержки древовидный граф разделяется на сегменты (шарды, «гнезда»), хранимые в кластерах, которые чаще всего и работают с этой частью графа. Таким образом, граф наращивается асинхронно в разных его частях, однако периодически происходит синхронизация, в то же время алгоритм консенсуса гарантирует согласованность имеющихся данных.

Подробное описание работающего в DGT алгоритма консенсуса F-BFT приведено в параграфе 1.1, однако для краткости можно отметить, что сначала транзакция проходит ряд проверок и голосование внутри кластера, в который она была добавлена клиентом, затем верифицируется на внешних для кластера узлах-арбитрах. Такая схема (рисунок 3) позволяет решить

[4] Хэш является результатом применения некоторой функции к набору данных, генерирующей на основе этих данных уникальную строку символов (бит)

проблему различных атак, например, двойной траты или подмены узла (Finney attack).

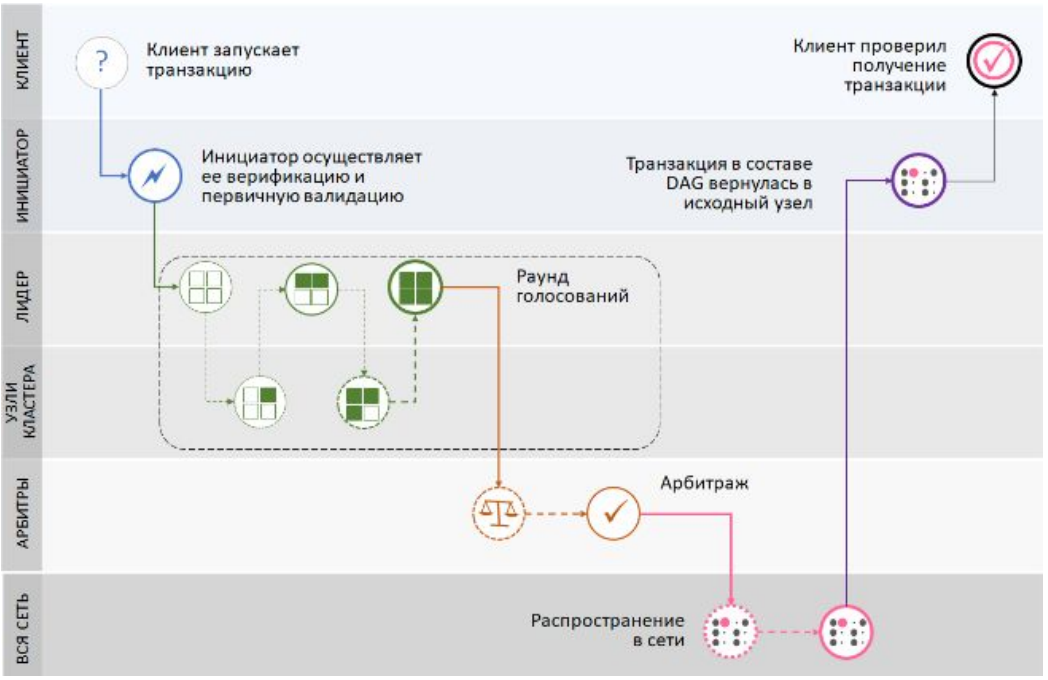

Рисунок 3 — Прохождение транзакции в сети DGT

Источник: документация DGT [16]

Стоит отметить, что программный продукт DGT (в прошлом BGX) является ответвлением фреймворка Hyperledger Sawtooth — открытого программного обеспечения, разработанного компанией IBM [18]. Данный фреймворк представляет собой блокчейн-систему, имеющую модульную архитектуру, что позволяет вносить изменения для адаптации решения под конкретные условия. Среди особенностей Hyperledger Sawtooth можно отметить следующее:

1. Поддержка разных семейств транзакций.
2. Динамический алгоритм консенсуса (возможность его замены, по умолчанию — PoET — Proof of Elapsed Time).
3. Гибкость построения сети узлов (сети могут быть как приватными, так и консорциум-базированными), при этом предполагается одноранговая сеть (p2p, peer-to-peer).
4. Развитый программный интерфейс приложения.

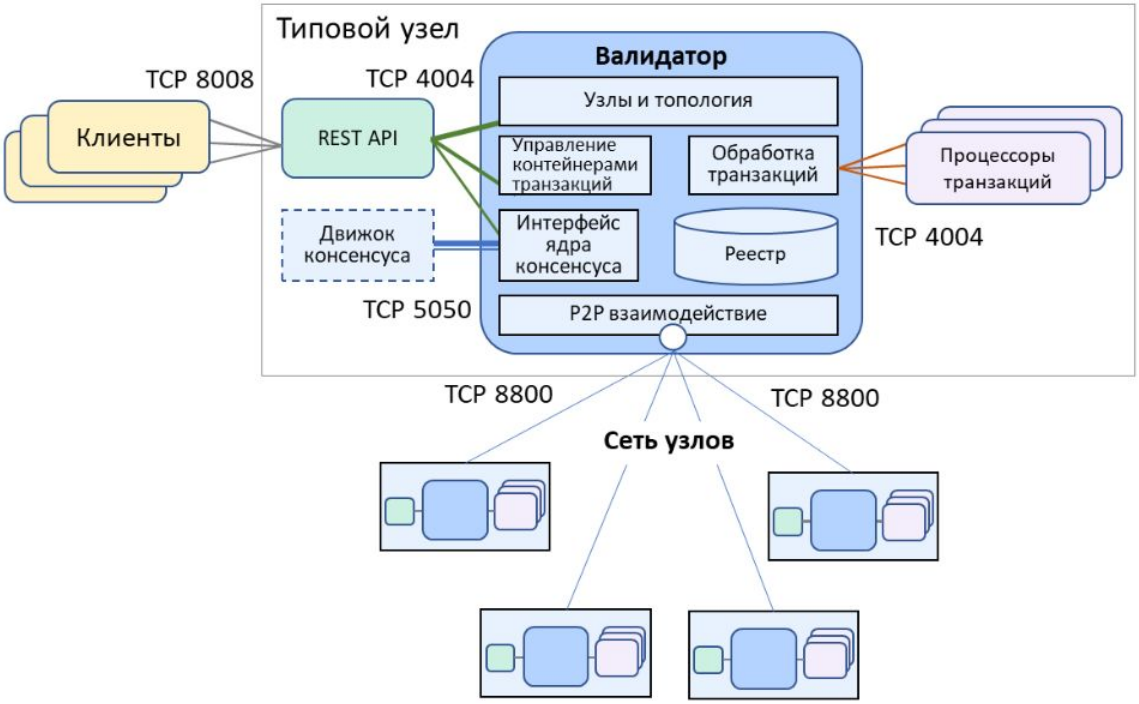

Рисунок 4 — Архитектура типового узла DGT

Источник: документация DGT [16]

Соответственно в DGT были произведены отказ от одноранговых сетей в пользу кластеров и добавление процессора топологий, а также якорный механизм (особые правила для присоединения узлов к кластеру, что положительно влияет на безопасность относительно атак вида 51%) и, как следствие, введены алгоритм консенсуса F-BFT и реестр DAG, что учитывает измененную архитектуру платформы. Архитектура узла представлена на рисунке 4.

Глава 2. Построение функции доверия и её встраивание в платформу DGT

2.1. Построение функции доверия на основе имеющихся показателей работы узлов

Определим понятие «доверие» (trust) как «убежденность в чьей-нибудь честности, порядочности; вера в искренность и добросовестность кого-нибудь»[5]. В приведённом выше обзоре литературы были приведены разные подходы к построению функции доверия/репутации. Некоторые из них основываются на отношении одних узлов к другим, однако это представляется весьма субъективным фактором. В данной работе доверие определяется объективными факторами, которые можно точно измерить, найти, подсчитать. Из имеющейся при работе сети информации об узлах мы можем извлечь такие факторы, как время подключения узла к сети, физическое время жизни (измеряемое, например, в секундах; т.е. время, когда узел был активен), число транзакций, инициированных данным узлом, и число успешных транзакций, которые были инициированы узлом. Можно заметить, что физическое время не так информативно, если активность работы сети низкая или неравномерная. Для иллюстрации данного замечания стоит представить сеть из достаточно большого числа узлов, где один из узлов активен почти всё время, кроме, разве что, времени около полудня, а другой узел почти всё время отключен, кроме того же времени около полудня; притом это сеть, например, компании в сфере питания, и основная нагрузка ложится как раз на обеденное время, когда люди активно совершают оплату в виде транзакций; тогда второй узел принимает участие в

[5] Толковый словарь Ожегова. С.И. Ожегов, Н.Ю. Шведова. 1949-1992

большем числе голосований и более полезен для сети, чем первый узел, который не принимал существенного участия в валидации транзакций. Из приведённого примера следует, что в качестве «времени жизни» может выступать такой относительный параметр, как число участий в голосованиях. Значение этого фактора становится более информативным в сравнении с аналогичными значениями других узлов. Кроме того, из информации о работе сети можно извлечь задержку (latency) как время, проходящее между моментами отправки транзакции лидером узлу на проверку и получением результата (то есть время голосования). Однако этот фактор в большей степени говорит о топологии сети, о физической удаленности одних узлов от других, который будет заметно изменяться при смене лидера. Время подключения узла к кластеру также не является чем-то информативным в контексте доверия к узлу, его надежности.

В результате некоторой комбинации имеющихся факторов (переменных) требуется получить функцию, результат которой описывает доверие к узлу. Доверие выражается в том, что новая транзакция, инициируемая данным узлом, будет успешно добавлена в итоговый реестр. Результатом вычисления функции от приведенных выше переменных (факторов) служит число TrustScore $\in [0, 1]$. Необходимое число голосов за принятие транзакции, инициированной некоторым узлом, снижается пропорционально значению функции доверия TrustScore данного узла. Например, при общем числе узлов в кластере, равном 15, требуется более 10 голосов для принятия транзакции при использовании протокола консенсуса F-BFT (то есть более ⅔ всех голосов). Однако с учетом значения доверия узла, который инициировал данную транзакцию, равного, для условности, 0.8, требуется уже более 8 голосов (вместо 10). Таким образом, мы учитываем историю узла, его поведение в сети для оценки его надежности, доверия к нему с целью уменьшения времени ожидания необходимого числа

голосов, нагрузки на вычислительные мощности (в таком случае мы должны уведомить ещё не проголосовавшие узлы остановить проверку транзакции, тем более, что одним из условий работы консенсусов Raft и F-BFT является то, что время на доставку сообщений между узлами заметно меньше времени голосования, следовательно дополнительное сообщение об остановке дойдет достаточно быстро и до того, как завершиться проверка на узле).

При построении функции доверия также можно учесть и то, что новички не должны иметь высокое значение доверия ввиду недостаточного объема репрезентативной статистики о них, а также для предотвращения желания недобросовестных узлов создавать новые сущности и сразу влиять на работу сети (хотя для борьбы с этим уже имеется якорный механизм в платформе DGT, однако в общем случае его может и не быть). Кроме того, новая транзакция, которая окажется отвергнутой, должна резко понижать доверие к узлу. В дополнение ко всему вышеперечисленному, новые транзакции могут иметь больший вес, чем ранее проведенные транзакции, а также проведенные до момента времени t_0 операции можно не учитывать вовсе.

Таким образом, в нашем распоряжении имеются следующие факторы, описывающие историю поведения узлов:

1. L_i — относительное время жизни i-го узла (измеряется числом участия в голосованиях);
2. N_i — число успешных транзакций, инициированных i-м узлом;
3. M_i — число всех транзакций, инициированных i-м узлом.

$$Acc_i = \frac{N_i}{M_i} \tag{1}$$

Заметим, что в самом простом случае отношение числа успешных транзакций к общему числу, отраженное в формуле (1), представляет собой не что иное, как точность (accuracy). Однако добавление новой невалидной

транзакции слабо отражается на точности, тогда как она должна резко понижать данный показатель. Вместо линейной зависимости проектируемой функции TrustScore, которая слабо чувствительна к новым отвергаемым транзакциям, можно использовать экспоненту, и тогда падение будет значительным даже при минимальном изменении отношения N_i к M_i.

$$TrustScore_i = e^{C\frac{N_i}{M_i}} - 1 \quad (2)$$

Формула (2) отражает экспоненциальную зависимость значения доверия i-го узла от отношения успешных транзакций к общему числу транзакций. Вычитание единицы является переносом в начало координат (0; 0). Параметр C в показателе экспоненты влияет на степень наклона графика экспоненты, его близость к оси абсцисс: чем больше значение C, тем «глубже» график функции, чем значение C ближе к нулю, тем более линейным становится график. Таким образом, параметр C контролирует «степень наказания» за неправильные транзакции. Стоит отметить, что при $M_i = 0$ можно считать значение функции доверия равным ε как малому положительному числу во избежание деления на 0. Далее формула (2) может быть модифицирована.

$$TrustScore_i = \frac{L_i}{\max_k L_k}(e^{C\frac{N_i}{M_i}} - 1) \quad (3)$$

В формуле (3) учитывается время жизни узла относительно других узлов в этом кластере, когда узлы, принявшее участие в большем числе голосований, имеют больший вес, чем новые узлы. Теперь нормируем полученную величину.

$$TS_i = \frac{TrustScore_i}{\max_k TrustScore_k} \quad (4)$$

Значение функции доверия каждого узла в формуле (4) нормируется на величину максимального значения функции среди всех узлов в этом кластере. Заметим, что в таком случае каждое значение $TrustScore_i$

масштабируется на одинаковое для всех максимальное значение L_k среди всех узлов, что теряет смысл при нормировке всех значений в формуле (4). Перепишем формулу (3) как (5):

$$TrustScore_i = L_i(e^{C\frac{N_i}{M_i}} - 1) \tag{5}$$

Придание важности более новым транзакциям можно произвести с помощью модификации величин N_i и M_i.

$$N_i = \sum_j w(t_j)\delta_j, \quad where\ \delta_j = \begin{cases} 1, & \text{if j was correct} \\ 0, & \text{else} \end{cases} \tag{6}$$

$$M_i = \sum_j w(t_j) \tag{7}$$

Пусть каждой транзакции j соответствует время t_j, когда она была выполнена и добавлена в реестр, $w(t_j)$ — значение весовой функции в момент времени t_j, δ_j равно 1, если транзакция j, инициированная узлом i, была верна, иначе 0. Тогда формулы (6) и (7) придают вес транзакциям в зависимости от времени их принятия.

Формулы (8)–(11) являют собой вариации того, как может выглядеть весовая функция в зависимости от требований сети к важности новых транзакций. Формула (8) отражает случай, когда все транзакции имеют равный вес вне зависимости от времени. Именно такой смысл изначально присутствовал в формулах (1)–(3), (5). Формула (9) отсекает все транзакции до определенного момента времени. Формула (10) показывает линейное отношение времени, прошедшего с некоторого момента времени t_0 до момента t выполнения транзакции, к временному промежутку, прошедшему с t_0 до текущего момента (линейная интерполяция). Стоит заметить, что с помощью данной формулы не следует вычислять вес транзакций, происходящих в данный момент, ввиду деления на 0. Формула (11) вместо линейной оценки разбивает временную ось на промежутки времени, каждый

из которых имеет свой вес. Таким образом, формула (9) является частным случаем (11).

$$w(t) = 1 \tag{8}$$

$$w(t) = \begin{cases} 0, & \text{if t} \prec t_0 \\ 1, & \text{else} \end{cases} \tag{9}$$

$$w(t) = \frac{t - t_0}{t_{now} - t_0} \tag{10}$$

$$w(t) = \begin{cases} w_0 & t \prec t_0 \\ w_1 & t_0 \prec t \prec t_1 \\ \dots & \\ w_n & t_{n-1} \prec t \prec t_n \end{cases} \tag{11}$$

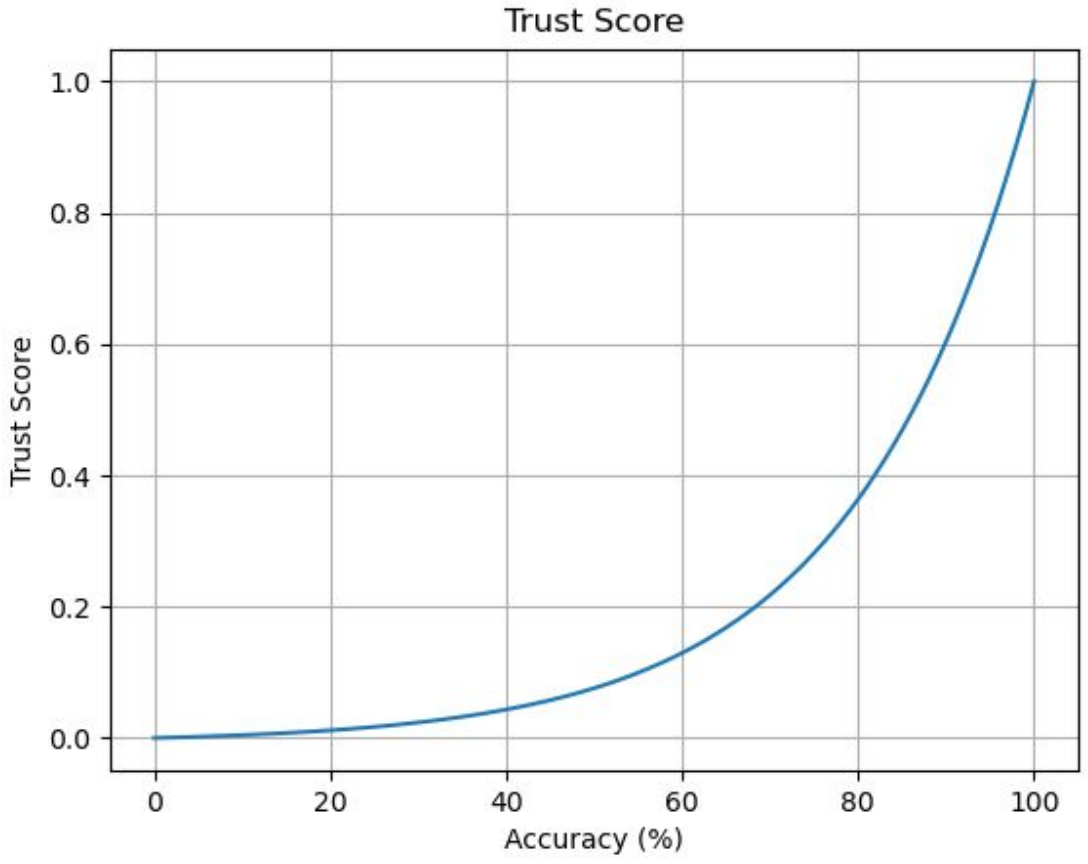

Рисунок 5 — Зависимость функции доверия от доли корректных транзакций

Для наглядности построим график функции доверия, вычисленной по формулам (4) и (5), в зависимости от отношения (1), представляющего собой

точность (т.е. какая доля транзакций была корректной). Параметр C примем равным 5. Результат представлен на рисунке 5.

В качестве примера смоделируем возможных участников сети при помощи параметров L, N, M (таблица 2).

Таблица 2 — Значение параметров при моделировании поведения узлов

№ узла	L	N	M
1	10	2	2
2	202	50	50
3	202	45	50
4	100	17	25
5	100	22	25
6	40	10	10
7	150	37	40
8	150	32	40

Узел 1 представляет собой новичка, недавно присоединившегося к сети, 2 и 3 — два узла, присоединившиеся к сети в момент её создания, но узел 3 имеет несколько непринятых транзакций, 4 — узел, иногда пытающийся провести транзакции, которые в конечном итоге отвергаются (мошенническое поведение), 5 — узел, который недавно инициировал транзакцию, которая не прошла, 6 — узел, который почти не принимает участия в голосованиях, но активно инициирует транзакции (при этом успешно), 7 и 8 — два узла, проведших по одной неудачной транзакции, но в одном случае она была инициирована давно, в другом — недавно. Стоит заметить, что здесь представлены значения N уже после использования весовой функции (поэтому, например, одна неудачная транзакция узла 7 может весить 3 условные единицы).

Таблица 3 — Результаты работы функции доверия для моделируемых узлов

№	1	2	3	4	5	6	7	8
TrustScore	0.05	1.0	0.67	0.11	0.3	0.22	0.56	0.3

Значения функции доверия для моделируемых узлов представлены в таблице 3. Результаты соответствуют ожиданиям: безошибочный узел 2 имеет доверие, равное 1.0, узлы 3 и 7, допускавшие неверные транзакции, имеют доверие, большее 0.5, узел-новичок 1 и часто инициирующий неверные транзакции узел 4 имеют крайне низкие рейтинги. Узлы 5 и 8 в недавнем прошлом инициировали неверную транзакцию, и теперь их значения доверия сильно упали, однако с течением времени они могут подняться при добросовестной работе узлов. Узел 6 практически не принимал участия в голосованиях и не являлся активным участником сети, потому и имеет относительно низкий рейтинг.

Стоит напомнить, что функция доверия призвана снижать число голосов, необходимых для принятия транзакции в кластере, и выступает в некоторой степени поощрением за качественную работу. Даже малое число отвергнутых транзакций должно сильно сказываться на доверии к узлу, так как ожидается практически безошибочная работа узлов для уменьшения числа необходимых проверок. Значение доверия, близкое к нулю, говорит лишь о том, что для принятия требуется ровно столько голосов, сколько требует алгоритм консенсуса без использования функции доверия. Построенная функция доверия — формулы (4)–(11) — удовлетворяет всем перечисленным выше условиям.

2.2. Встраивание функции доверия в систему DGT

Для встраивания построенной в параграфе 2.1 функции доверия в систему DGT требуются небольшие изменения, связанные с устройством и работой сети. Некоторые из факторов, описывающих узлы, можно взять из

распределенного реестра, например, число успешных транзакций, инициированных некоторым узлом. Однако число участий узла в голосованиях (в том числе отрицательные ответы на запрос об утверждении транзакции), количество отвергнутых транзакций, которые временно хранятся в журнале (логах) валидатора, не хранится в общем реестре или получение этих величин затруднительно. В таком случае можно организовать сбор статистики, и делать это может лишь лидерский узел. Именно он может записывать все участия в голосованиях, считать все полученные от инициаторов транзакции. Если после того, как лидер сам выступает инициатором и рассылает транзакции со статистиками узлам, набирается достаточное число голосов от узлов, успешно валидировавших транзакции, то собранная статистика добавляется в распределенный реестр. Добавление в DAG означает и распространение транзакции среди всех узлов, поэтому при смене лидера у нового лидера уже будет вся необходимая ему информация о показателях других узлов. Именно лидер считает значение доверия к узлу, когда получает от последнего новую транзакцию, чтобы узнать, какое число голосов необходимо для принятия транзакции. Добавлять показатели работы узлов в реестр DAG возможно, так как в реестре присутствуют транзакции разных семейств, одним из которых является семейство транзакций топологии. Для этого требуется немного изменить процессор транзакций, обрабатывающий транзакции топологии, но система DGT как раз позволяет писать новые процессоры транзакций для возможности обработки самых разных данных, в том числе и статистик узлов. Открытым остается вопрос, как часто лидер инициирует новые транзакции, касающиеся статистик. Формально изменение TrustScore происходит после каждой новой транзакции, но инициировать новую транзакцию после каждой утвержденной затратно. Поэтому стоит определить некоторые периоды времени, по истечении которых лидер запускает процесс обновления статистик в реестре

при помощи инициации новой транзакции. Выбор такого периода зависит от активности работы сети. Например, если транзакции приходят в сети редко, то не стоит обновлять статистику так часто. Кроме того, с тех пор, как статистика добавляется в реестр с помощью транзакций, уже само лишь обновление показателей работы узлов, изменяющее значение доверия, также изменяет значение доверия. Этот замкнутый круг можно прервать, если, например, не учитывать в показателях работы узлов транзакции семейства топологии, к которым и относится добавляемая статистика.

Альтернативным путем внедрения функции в работу платформы является введение дополнительного реестра, как это было сделано в работе [5], краткое описание которой есть в обзоре литературы. Однако добавление статистик непосредственно в реестр DAG при небольшом изменении процессора топологий выглядит наиболее оптимальным решением.

Стоит отметить, что подсчет лидером статистик узлов, периодическое добавление последних в реестр и вычисление значения доверия к узлу также требуют некоторых ресурсов, однако они заметно меньше, чем потенциально сэкономленные вычислительные мощности и время при использовании функции доверия для уменьшения необходимого для принятия числа голосов.

Выводы

Построенная в параграфе 2.1 с помощью формул (4)–(11) функция доверия действительно отражает ту степень, с которой можно ожидать надежную работу узлов, снижая вычислительную нагрузку на сеть в виде числа голосов пропорционально рейтингу доверия к узлу-инициатору. В доказательство этого можно привести результаты моделирования типового поведения узлов в таблице 3, которые вполне соответствуют ожидаемым. Описанное в параграфе 2.2 встраивание работы функции в платформу DGT действительно требует минимальных изменений последней, при этом процесс сбора необходимой информации, её добавление в реестр и подсчет рейтинга доверия не несут серьезных вычислительных и транспортных затрат, позволяя в то же время значительно экономить эти самые ресурсы за счет уменьшения числа голосов. Подтвердить это непосредственным экспериментом не представляется возможным из-за пока что тестовой работы платформы DGT, отсутствия необходимых данных и малой нагрузки на платформу в данный момент в целом. Кроме того, некоторые заявленные элементы платформы DGT отсутствуют, но ожидаются в следующей версии этой развивающейся платформы.

Заключение

В данной работе были проанализированы существующие механизмы консенсуса и рассмотрена архитектура платформы DGT для понимания их работы и дальнейшей адаптации алгоритма консенсуса F-BFT и платформы DGT в рамках успешного построения функции доверия узлов и её встраивание в рассматриваемую платформу. Создание данной функции является важным шагом для построения краудфандинг-платформы на основе DGT, ведь именно доверие узлов сети является одной из основ инвестиционной платформы. Стоит отметить, что все задачи были успешно решены, а цель достигнута. В качестве дальнейшей работы может служить рассмотрение функции доверия не только как поощрения для добросовестных узлов и улучшения производительности сети, но и как наказания для недобросовестных участников, а также для оценки арбитров для последующего их выбора. Кроме того, можно расширить работу функции доверия на весь кластер, учитывать и другие факторы работы узлов, а также применить современные методы обнаружения недобросовестного поведения узлов, как, например, машинное обучение или нейронные сети.

Список литературы

1. Qianlan Bai, Chao Zhang, Yuedong Xu, Xiaowei Chen, Xin Wang Evolution of Ethereum: A Temporal Graph Perspective. 2020. URL: https://arxiv.org/pdf/2001.05251.pdf (дата обращения: 31.05.2020).
2. Официальный сайт платформы DGT [Электронный ресурс]. URL: https://dgtworld.ru/ (дата обращения: 31.05.2020).
3. Мурзин П.Е. Основные подходы к разработке протокола консенсуса в распределенных реестрах // Вестник современных цифровых технологий. 2019. С. 24–34. URL: www.granit-concern.ru/pdf/Murzin_statia_razrabotka_consensa_rr.pdf (дата обращения: 31.05.2020).
4. Экосистема Сбербанка SberX [Электронный ресурс]. URL: https://www.sberbank.ru/ru/ecosystem (дата обращения: 31.05.2020).
5. R. Di Pietro, X. Salleras, M. Signorini, and E. Waisbard A blockchain-based trust system for the Internet of Things // Proc. of the 23th ACM on Symposium on Access Control Models and Technologies (SACMAT). 2020. doi:10.1145/3205977.3205993.
6. S. Tamang Decentralized Reputation Model and Trust Framework Blockchain and Smart contracts. 2018. URL: http://uu.diva-portal.org/smash/get/diva2:1352089/FULLTEXT01.pdf (дата обращения: 31.05.2020).
7. L. Page, S. Brin, R. Motwani, T. Winograd The PageRank Citation Ranking: Bringing Order to the Web. Technical Report. Stanford InfoLab. 1999. URL: http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf (дата обращения: 31.05.2020).

8. Li, Rong-Hua & Yu, Jeffrey & Huang, Xin & Cheng, Hong A Framework of Algorithms: Computing the Bias and Prestige of Nodes in Trust Networks // PloS one 7(12):e50843. 2012. doi:10.1371/journal.pone.0050843.
9. Griffiths, Nathan Task delegation using experience-based multi-dimensional trust // Proceedings of the International Conference on Autonomous Agents. c. 489-496. 2005. doi:10.1145/1082473.1082548.
10. Kamvar, Sepandar & Schlosser, Mario & Garcia-molina, Hector The EigenTrust Algorithm for Reputation Management in P2P Networks. 2003. URL: https://nlp.stanford.edu/pubs/eigentrust.pdf (дата обращения: 31.05.2020).
11. The Trust Chain Consensus. COTI: a decentralized, high performance cryptocurrency ecosystem optimized for creating digital payment networks and stable coins. Technical White Paper, V.4.0 [Электронный ресурс]. 2018. URL: https://coti.io/files/COTI-technical-whitepaper.pdf (дата обращения: 31.05.2020).
12. Ismail, L. & Materwala, H. A Review of Blockchain Architecture and Consensus Protocols: Use Cases, Challenges, and Solutions // Symmetry. 2019, *11*(10), 1198. doi:10.3390/sym11101198.
13. Castro M. Practical Byzantine Fault Tolerance. 1999. URL: http://www.pmg.csail.mit.edu/papers/osdi99.pdf (дата обращения: 31.05.2020).
14. Ongaro, D. & Ousterhout, J. In search of an understandable consensus algorithm // USENIX. 305-320. 2014. URL: https://raft.github.io/raft.pdf (дата обращения: 31.05.2020).
15. Официальная документация платформы Sawtooth Hyperledger. Raft [Электронный ресурс]. URL: https://sawtooth.hyperledger.org/docs/raft/nightly/master/introduction.html (дата обращения: 31.05.2020).

16. Документация платформы BGX/DGT. Версия 03 [Электронный ресурс]. URL: https://app.box.com/s/ims8y9pv525vkfuewbbtaa4ourq14iay (дата обращения: 31.05.2020).
17. Bogdanov A., Degtyarev A., Korkhov V., Iakushkin O., Khvatov V. On Conceptual Response to Some of the Blockchain Problems. 2018. URL: http://ceur-ws.org/Vol-2267/337-341-paper-64.pdf (дата обращения: 31.05.2020).
18. Официальная документация платформы Sawtooth Hyperledger [Электронный ресурс]. URL: https://sawtooth.hyperledger.org/docs/core/releases/1.1/architecture.html (дата обращения: 31.05.2020).