ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ И СЕТИ

УДК 334.029.3

МОДЕЛЬ СЕТИ БЛОКЧЕЙН, ЗАПИСЕЙ И ТРАНЗАКЦИИ

В.А. ВИШНЯКОВ[1], Д.А. КАЧАН[2]

[1]Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»,
ул. П. Бровки, 6, Минск, 220600, Беларусь
ORCID: https://orcid.org/0000-0003-2929-8958

[2]ОАО «Гипросвязь»,
ул. Сурганова, 24, Минск, 220012, Беларусь
ORCID: https://orcid.org/0000-0003-2229-4022

Поступила в редакцию 24 декабря 2022

В статье приведен краткий обзор использования технологии блокчейн в системе управления образованием, включая разработки авторов. Приведена математическая модель сети блокчейн, включая формальное представление, структуры, учетные записи, транзакции. Особое внимание уделено общей функции стоимости хранения данных, хеш-функции Keccak с переменной длиной выхода, алгоритму рекурсивного префикса длины. Рассмотрены два типа транзакций: традиционные и основанные на механизмах децентрализованных приложений.

Ключевые слова: технология блокчейн, система образования, модель, структура, учетная запись, транзакция.

Введение. При большом многообразии работ в области информационного управления социальными системами актуальным является исследование вопросов развития моделей управления на основе ведущих технологических трендов, в частности технологии распределенных реестров (блокчейн). Технология блокчейн является одним из актуальных технологических решений уже на протяжении более 10 лет [1]. Благодаря заложенному при создании потенциалу и научному заделу, блокчейн имеет широкие границы применения и постоянно совершенствуется, открывая все новые способы использования.

Начиная с 2013 года, с появлением блокчейн Ethereum, применение блокчейн вышло за рамки осуществления финансовых транзакций [2]. Заложенная в Ethereum возможность реализации «умных контрактов» и экосистемы «распределенных приложений» (distributed apps, dApps) значительно расширила применимость технологии, в том числе как средство оптимизации методик информационного управления [3].

Авторами на основе технологии блокчейн разработаны: алгоритмы получения цифрового документа об образовании и подтверждения его достоверности на [4]; структура интернет-маркетинга и алгоритмы для интеллектуальной многоагентной системы [5]; графическая структура взаимосвязи блокчейн и базисных факторов системы образования, исследование которой в процессе когнитивного моделирования показало повышение эффективности до 5 % [6]. В данной работе приведена модель сети блокчейн, включая формальное представление, структуры, учетные записи, транзакции.

Модель сети блокчейн. Выбор блокчейн-сети Ethereum в качестве «опорной» технологии для информационного управления в образовании обосновывается следующим образом: Ethereum предоставляет разработчику интегрированную систему для разработки программного обеспечения с поддержкой контрактов. Работа сети начинается с «нулевого» состояния «genesis», которое благодаря транзакциям преобразуется в некоторое текущее

состояние. Понятие «состояние» может включать информацию: состояние счетов (остатки на счетах), репутация, соглашения, данные, относящиеся к информации из физического мира (связь с IoT) и др. Транзакции делятся на достоверные (valid) и недостоверные (invalid). Достоверные транзакции отражают допустимое состояние блокчейн-сети. К недостоверным транзакциям относятся такие транзакции, как уменьшение баланса счета без равного увеличения в другом месте. Достоверные транзакции отражают допустимый переход состояния блокчейн-сети. Функция перехода может быть выражена следующим образом:

$$\sigma_{t+1} \equiv Y\left(\sigma_t, T\right), \tag{1}$$

где σ_{t+1} – новое состояние сети; Y – функция перехода состояния блокчейн-сети; T – достоверные транзакции в блокчейн-сети; σ отражает достоверное состояние блокчейн-сети, то есть состояние всех счетов, загруженных и вызванных смарт-контрактов.

Все достоверные транзакции объединяются в блоки, которые соединяются в цепочку с использованием дайджеста (криптографического хэша) в качестве ссылки на предыдущие достоверные состояния блокчейн-сети. Блоки формируются как реестр записей, дописывая серию транзакций вместе с предыдущим блоком и идентификатором (дайджестом) конечного состояния. Дополнительно осуществляется дописывание транзакций, касающихся компенсации расходов вычислительных узлов (процесс «оплаты» услуг вычислительных узлом условно называется «чеканкой»). Формальное математическое представление блокчейн-сети:

$$\sigma_{t+1} \equiv P\left(\sigma_t, B\right), \tag{2}$$

где σ_{t+1} – новое состояние сети; P – функция перехода состояния блока, отражающие принятые правила формирования блоков; σ_t – текущее состояние сети; B – блок, включающий в себя серию достоверных транзакций, обеспечивающих переход состояния блокчейн-сети, включая дополнительные компоненты, необходимые для работы блокчейн-сети.

$$B \equiv \left(\ldots, \left(T_0, T_1, \ldots\right), \ldots\right), \tag{3}$$

где B – блок, включающий в себя серию достоверных транзакций, обеспечивающих переход состояния блокчейн-сети; T_0, T_1 – достоверные транзакции.

Функция перехода:

$$P\left(\sigma, B\right) \equiv \Omega\left(B, Y\left(Y\left(\sigma, T_0\right), T_1\right)\ldots\right), \tag{4}$$

где Ω – функция добавления блока в цепь блоков блокчейн-сети с переходом сети в новое состояние и перечислением вознаграждения узлам за проделанные вычисления.

Уравнения (2–4) отражают парадигму децентрализованных транзакционных систем, основанных на механизмах консенсуса. Выделим:

– общую функцию стоимости хранения данных в сети блокчейн (любая транзакция обеспечивает передачу и хранение данных в сети блокчейн, реализуемую вычислительными узлами) C_{sstore};

– функцию хэш-свертки *KEC* (хеш-функция Keccak с переменной длиной выхода);

– кортеж T отражает транзакции (T_n – отдельная транзакция в сети блокчейн).

Состояние блокчейн-сети – это связь между адресами (160-битными идентификаторами) и состояниями учетных записей.

Для обеспечения хранения данных вычислительными узлами используется процедура сериализации данных, работающая в паре с обратной процедурой – десереализацией. Для сериализации в Ethereum применяется особый алгоритм – рекурсивный префикс длины (Recursive Length Prefix, RLP). RLP – это процесс кодирования структуры данных в последовательности байтов и используется для всех типов данных, таких как учетные записи, транзакции и блоки. Одни и те же входные данные всегда преобразуются в одни и те же последовательности байт. Определение множество возможных структур выражается формулами:

$$T \equiv L \text{ в } B; \tag{5}$$

$$L \equiv \left\{ t : t = \left(t[0], t[1], \ldots \right) \wedge \forall n < \|t\| : t[n] \in T \right\}; \tag{6}$$

$$B \equiv \left\{ b : b = \left(b[0], b[1], \ldots \right) \wedge \forall n < \|b\| : b[n] \in O \right\}, \tag{7}$$

где O – набор байт; B – набор всех последовательностей байт (иначе известных как массивы байт); L – это набор всех древовидных структур (подструктур); T – набор всех массивов байт и структурных последовательностей.

Функция RLP определяется с помощью двух подфункций, первая из которых обрабатывает данные, когда их значение представляет собой массив байт, вторая – когда это последовательность дополнительных значений.

Если значение, подлежащее сериализации, представляет собой массив байтов, RLP принимает одну из трех форм:

– если массив байт содержит только один байт, и этот единственный байт меньше 128, то входные данные в точности равны выходным;

– если массив байт содержит менее 56 байт, то выходные данные равны входным с префиксом байта, равным длине массива байт плюс 128;

– выходные данные равны входным, при условии, что они содержат менее 2^{64} байт, с префиксом массива байт минимальной длины, который при интерпретации как большое целое число равное длине массива входных байт, которое в свою очередь имеет префикс равный числу байт, необходимых для сериализации плюс 183.

Массивы байт, содержащие 2^{64} или более байт, не могут быть кодированы. Это ограничение гарантирует, что первый байт кодировки массива байтов всегда меньше 192, и, таким образом, его можно легко отличить от кодировок последовательностей в L. Таким образом, BE – это функция, которая расширяет неотрицательное целочисленное значение до массива байт минимальной длины, обеспечивая конкатенацию последовательностей данных. Если значение, подлежащее сериализации, представляет собой последовательность, то RLP принимает один из двух видов:

– если объединенные сериализации каждого содержащегося элемента имеют длину менее 56 байт, то выходные данные равны этой конкатенации с префиксом байта, равным длине этого массива байтов плюс 192;

– выходные данные равны объединенным сериализациям при условии, что они содержат менее 264 байт, с префиксом массива байт минимальной длины, который при интерпретации как целое число с большим числом байт равен длине массива байт, объединенных сериализацией, который сам по себе имеет префикс требуемого количества байт в виде значения длины плюс 247.

Объединенные последовательности, элементы которых содержат 2^{64} или более байт, не могут быть кодированы. Это ограничение гарантирует, что первый байт кодировки не превышает 255. Если RLP используется для кодирования скалярных величин, определенных только целыми неотрицательными числами (входят в множество N), кодировка осуществляется как для короткого массива байт.

Если входные данные представляют собой один байт в диапазоне *[0x00, 0x7f],* то кодировка не осуществляется (это и есть вид кодировки RLP):

1. Если данные не имеют значения (нулевая величина или байт, пустая строка, пустое множество), кодировка RLP имеет вид *0x80*.

2. Если входные данные представляют собой байт в диапазоне *[0x80, 0xff]* кодировка RLP будет определяться конкатенацией *0x81* с байтом *([0x81, байт]*).

3. Если входные данные представляют собой строку длиной 2–55 байт, кодировка RLP состоит из одного байта со значением *0x80* плюс длина строки в байтах, а затем массива шестнадцатеричного значения строки.

*Например, транзакция (значение, записанное в реестр блоков) текста «hello world» в Ethereum будет иметь вид [***0x8b****, 0x68, 0x65, 0x6c, 0x6c, 0x6f, 0x20, 0x77, 0x6f, 0x72, 0x6c, 0x64]:*

– hello world имеет длину, равную 11 байт в десятичном исчислении или 0x0b в шестнадцатеричном;

– первый байт кодировки RLP равен сумме 0x80 и 0x0b – **0x8b;**

– далее следуют байты фразы «hello world».

4. Если входные данные представляют собой строку длиной более 55 байт, кодировка RLP состоит из 3 частей слева направо:

– первая часть представляет собой один байт со значением *0xb7* плюс длина в байтах второй части;

– вторая часть – это шестнадцатеричное значение длины строки;

– последняя строка – это строка в байтах. Диапазон первого байта равен [0xb8, 0xbf].

Например, строка с 1024 символами «a» имеет вид ***[0xb9, 0x04, 0x00*****, 0x61, 0x61, …]**. *Первая часть –* ***0xb9*** **=** *0xb7 + 0x02 (0x02 это длина второй части); вторая часть – это* ***0x04 и 0x00 (****длина строки* 0x0400 **=** 1024**);** *от четвертого элемента массива* **0x61** *до конца идет строка в байтах (третья часть).*

5. Если ввод представляет собой пустой массив, кодировка RLP имеет один байт *0xc0*.

6. Если входные данные представляют собой список с длиной 0–55 байт, кодирование RLP состоит из одного байта со значением *0xc0* плюс длина списка, а затем конкатенация кодировок RLP элементов в списке. Диапазон первого байта равен *[0xc1, 0xf7]*.

Например [«hello», «world»] будет представлен в виде *[0xcc, 0x85, 0x68, 0x65, 0x6c, 0x6c, 0x6f, 0x85, 0x77, 0x6f, 0x72, 0x6c, 0x64]*. В этом кодировании RLP «hello» представлено в виде *[0x85, 0x68, 0x65, 0x6c, 0x6c, 0x6f]*; кодированием RLP «world» является *[0x85, 0x77, 0x6f, 0x72, 0x6c, 0x64]*; *0xcc* равное *0xc0 + 0x0c0x0c = 0x06 + 0x06* является длиной.

7. Если ввод представляет собой список с общей полезной нагрузкой длиной более 55 байт, кодирование RLP включает 3 части. Первая – это один байт со значением *0xf7* плюс длина в байтах второй части. Вторая часть – это длина общей полезной нагрузки. Последняя часть – это конкатенация RLP-кодировок элементов в списке. Диапазон первого байта равен *[0xf8, 0xff]*.

Учетные записи в блокчейн-сети. Для осуществления транзакций в блокчейн применяется модифицированное дерево Меркла – полное двоичное дерево, в листовые вершины которого помещены дайджесты блоков данных, а внутренние вершины содержат дайджесты от сложения значений в дочерних вершинах [7]. Корневой узел дерева Меркла содержит дайджест от всего набора данных, то есть хеш-дерево является однонаправленной хеш-функцией. Дерево Меркла позволяет получить «отпечаток» всех транзакций в блоке, а также эффективно верифицировать транзакции.

В Ethereum используется усовершенствованный алгоритм формирования дерева Меркла – дерево Меркла-Патриции. Функция вычисления дайджеста обозначается TRIE.

Состояние учетной записи $\sigma[a]$ содержит четыре поля:

– номер (nonce): скалярное значение, равное количеству транзакций, отправленных с этого адреса, или, в случае учетных записей с соответствующим кодом, количеству контрактов, созданных этой учетной записью. Для учета адреса a состояние формально обозначается $\sigma[a]_n$;

– баланс (balance): скалярное значение, равное количеству единиц криптовалюты, принадлежащих этому адресу. Формально обозначается $\sigma[a]_b$. Выражается в Wei (1 Ether равен 10^{18} Wei);

– storageRoot: 256-битный дайджест корневого узла дерева Меркла-Патриции, который кодирует содержимое хранилища учетной записи. Дайджест формально обозначается $\sigma[a]_s$;

– codeHash: дайджест кода EVM для учетной записи (код, который выполняется, если на этот адрес поступает сообщение с вызовом). Этот дайджест формально обозначается $\sigma[a]_s$, код может быть обозначен как b, учитывая, что $KEC(b) = \sigma[a]_s$.

Поскольку проще ссылаться не на корневой дайджест (trie), а на базовый набор пар ключ/значения, хранящиеся внутри, определим математический эквивалент этой пары:

$$TRIE\left(L_I^*\left(\sigma[a]_s\right)\right) \equiv \sigma[a]_s . \quad (8)$$

Функция свертывания для набора пар ключ/значение в trie, L_I^*, определяется как поэлементное преобразование базовой функции L_I и задается:

$$L_I\left((k,v)\right) \equiv \left(KEC(k), RLP(v)\right), \tag{9}$$

$$\text{где } K \in B_{32} \wedge v \in N, \tag{10}$$

$\sigma[a]_s$ является абстракцией для учетной записи и не учитывается при ее последующей сериализации.

Если поле codeHash является дайджестом *Keccak-256* пустой строки, т. е. $\sigma[a]_c = KEC(())$, то учетная запись представляет собой простую учетную запись, называемую «неконтрактной» учетной записью. Функция свертки состояния L_s определяется следующим образом:

$$L_s(\sigma) \equiv \left\{ p(a) : \sigma[a] \neq \varnothing \right\}, \tag{11}$$

$$\text{где } p(a) \equiv \left(KEC(a), RLP\left(\left(\sigma[a]_n, \sigma[a]_b, \sigma[a]_s, \sigma[a]_c\right)\right)\right). \tag{12}$$

Функция L_s используется наряду с функцией trie для вычисления краткого идентификатора (дайджеста) общего состояния.

$$\forall a : \sigma[a] = \varnothing \vee \left(a \in B_{20} \wedge v\left(\sigma[a]\right)\right), \tag{13}$$

где v – функция валидности учетной записи, которая может быть выражена:

$$v(x) \equiv x_n \in N_{256} \wedge x_b \in N_{256} \wedge x_s \in B_{32} \wedge x_c \in B_{32}. \tag{14}$$

Учетная запись пустая, если не содержит код, номера и имеет нулевой баланс:

$$EMPTY(\sigma, a) \equiv \sigma[a]_c = KEC(( )) \wedge \sigma[a]_n = 0 \wedge \sigma[a]_b = 0. \tag{15}$$

Вызываемые предварительно скомпилированные смарт-контракты могут иметь пустое состояние учетной записи. Это связано с тем, что состояния их учетных записей обычно не содержат кода, описывающего их поведение.

Учетная запись считается неактивной, когда ее состояние учетной записи не существует или пусто:

$$DEAD(\sigma, a) \equiv \sigma[a] = \varnothing \vee EMPTY(\sigma, a). \tag{16}$$

Транзакции в блокчейн-сети. Транзакция (T) – это единственная криптографически подписанная инструкция, созданная внешним субъектом по отношению к сети Ethereum – это означает, что отправителем транзакции не может быть смарт-контракт. Необходимо отметить, что человек является абсолютным действующим лицом по своей природе, однако транзакции могут осуществляться и программными алгоритмами по указанию или с ведома субъекта (человека).

Существует два типа транзакций: традиционные (аналогичные транзакциям в сети Bitcoin) и транзакции, основанные на механизмах децентрализованных приложений. Второй тип делится в свою очередь на две подкатегории: транзакции, которые приводят к вызовам сообщений (вызов смарт-контракта); транзакции, которые приводят к созданию новых учетных записей с соответствующим кодом (создание смарт-контракта). Все типы транзакций имеют ряд общих полей:

– номер (nonce): скалярное значение, равное количеству транзакций, отправленных отправителем (T_n);

– gasPrice: скалярное значение, равное количеству *W*ei, подлежащих оплате за единицу газа за все вычислительные затраты, понесенные в результате выполнения этой транзакции (T_p);

– gasLimit: скалярное значение, равное максимальному количеству газа, которое должно быть использовано при выполнении этой транзакции. Оплачивается авансом, до выполнения каких-либо вычислений, и не может быть увеличено позже (T_g);

– кому (to): 160-разрядный адрес получателя вызова сообщения или для транзакции, направленной на создание смарт-контракта (T_t);

– величина (value): скалярное значение, равное количеству Wei, которое должно быть передано получателю сообщения или, в случае создания контракта, в качестве затрат для вновь созданной учетной записи (T_v);

– *r, s*: значения, соответствующие подписи транзакции и используемые для определения отправителя транзакция (T_r и T_s).

Заключение. 1. Дан краткий обзор использования технологии блокчейн в системе управления образованием, включая разработки авторов. Приведена математическая модель сети блокчейн, ее формальное представление, описание структур, учетных записей, транзакций.

2. Приведена общая функция стоимости хранения данных, хеш-функция Keccak с переменной длиной выхода, обсужден алгоритм рекурсивного префикса длины.

3. Рассмотрены два типа транзакций: традиционные и основанные на механизмах децентрализованных приложений. Описаны поля обеих видов транзакций.

BLOCKCHAIN NETWORK MODEL, RECORDS AND TRANSACTIONS

U.A. VISHNIAKOU, D.A. KACHAN

Abstract

The article provides a brief overview of the use of blockchain technology in the education management system, including the authors' developments. The mathematical model of the blockchain network is presented, including the formal representation, structures, accounts, transactions. Special attention is paid to the general data storage cost function, the Keccak hash function with variable output length, and the recursive length prefix algorithm. Two types of transactions are considered: traditional and based on the mechanisms of decentralized applications.

Список литературы

1. Mengelkamp, E. A blockchain-based smart grid: towards sustainable local energy markets / E. Mengelkamp [et al.] // Computer Science-Research and Development. – 2018 – № 1(11). – P. 207–214.

2. Куприяновский, В. П. Цифровые цепи поставок и технологии на базе блокчейн в совместной экономике / В. П. Куприяновский [и др.] // International Journal of Open Information Technologies. – 2017. – Т. 5, № 8. – С. 80–95.

3. Власов, А. И. Системный анализ технологии обмена и хранения данных blockchain / А. И. Власов [и др.] // Современные технологии. Системный анализ. Моделирование. – 2017. – № 3. – С. 75–83.

4. Вишняков, В. А. Управление интернет-маркетингом в системе образования с использованием блокчейн технологий / В. А. Вишняков, Д. А. Качан // Доклады БГУИР. – 2020. – № 2. – С. 30–36.

5. Качан, Д. А. Подход и модели применения технологии распределенных реестров для подтверждения достоверности документов в образовании / Д. А.Качан, В. А. Вишняков // Доклады БГУИР. – 2020. – № 7. – С. 14–23.

6. Качан, Д. А. Оценка воздействия применения технологии распределенных реестров в системе образования с использованием когнитивного моделирования / Д. А. Качан, В. А. Вишняков // Проблемы инфокоммуникаций. – 2021. – № 1. – С. 35–40.

7. Алиев, И. А. Уязвимости смарт-контрактов блокчейн-платформы Ethereum / И. А. Алиев // Научные записки молодых исследователей. – 2019. – № 3. – С. 47–57.