УДК 004.75 DOI: 10.18698/0236-3933-2021-4-27-40

ОБРАБОТКА ОШИБОЧНЫХ СИТУАЦИЙ В БОЛЬШИХ БЛОКЧЕЙН-СЕТЯХ АЛГОРИТМОМ ДОСТИЖЕНИЯ КОНСЕНСУСА, ОСНОВАННОМ НА РЕШЕНИИ ЗАДАЧИ ВИЗАНТИЙСКИХ ГЕНЕРАЛОВ

А.П. Бардин[1] lovvi@mail.ru
А.В. Новицкий[1] a@sumus.team
Ю.Ю. Шумилов[2, 3] shm@bmstu.ru

[1] **Sumus Company Limited, Гонконг, Китайская Народная Республика**
[2] **МГТУ им. Н.Э. Баумана, Москва, Российская Федерация**
[3] **Финансовый университет, Москва, Российская Федерация**

Аннотация

Блокчейн-сети, построенные на основе алгоритмов PoS, DPoS, LPoS, PoE, PoIT, pBFT, имеют определенные ограничения, связанные со снижением скорости включения новых транзакций в блокчейн при увеличении числа узлов блокчейн-сетей, принимающих участие в создании блока. Разработанный алгоритм достижения консенсуса sdBFT позволяет по сравнению с существующими алгоритмами BFT увеличить на несколько порядков число узлов сети, участвующих в достижении консенсуса, сохраняя при этом скорость включения транзакций в блокчейн. Исследованы основные ошибочные ситуации в больших блокчейн-сетях с точки зрения их обработки алгоритмом sdBFT. Приведена обработка семи основных ошибочных ситуаций в блокчейне: мастер-узел недоступен; эскорт-узел недоступен; поступление некорректной транзакции; разное число закрытых блоков на разных узлах блокчейн-сети в один и тот же момент времени; узел отвергает новый блок блокчейна; сеть отвергает новый блок блокчейна; появление в сети двух блоков с идентичными номерами. Показано, что корректное взаимодействие алгоритма sdBFT с блокчейн-сетью позволяет обеспечить не только высокое быстродействие при большом числе узлов и высокой скорости поступления транзакций, но и способность блокчейна, использующего этот алгоритм, отрабатывать все основные ошибочные ситуации, возникающие при работе блокчейн-сети

Ключевые слова

Блокчейн-консенсус, транзакция, узел, блок, хеш-функция, электронная подпись

Поступила 09.12.2020
Принята 28.06.2021

Введение. За последние несколько лет разработаны алгоритмы консенсуса, устраняющие слабые стороны алгоритма доказательства работы PoW (Proof-of-Work) [1]: PoS, DPoS, LPoS, PoE, PoIT, pBFT [2–4]. Позже появились новые требования к блокчейн-сетям: время создания нового блока не более 1 мин; число узлов, участвующих в выработке консенсуса, равно 10^3–10^4; скорость поступления транзакций не менее 10^3 транзакций в секунду; алгоритм не должен требовать значительных энергозатрат и существенных вычислительных мощностей.

Ранее предложенные алгоритмы консенсуса не удовлетворяют перечисленным требованиям: наилучших результатов они достигают при числе узлов не более 40. Скорость поступления транзакций для такого числа узлов равна $2 \cdot 10^4$ транзакций в секунду, при этом время закрытия блока не превышает 40 с. Если число узлов более 60, то скорость транзакций для такого числа узлов составляет не более $5 \cdot 10^3$ транзакций в секунду, при этом время закрытия блока более 100 с. Время выработки консенсуса для 10^4 узлов составляет 6 мин.

Новый подход к решению проблемы производительности в алгоритме pBFT (practical Byzantine Fault Tolerant) реализован в алгоритме консенсуса sdBFT (stake distributed BFT), первый вариант которого для скорейшего ознакомления специалистов в этой области представлен авторами в 2018 г. в электронных ресурсах [5, 6]. Для удобства дальнейшего изложения приведем этот алгоритм в модифицированном виде с 11, а не с 9 шагами.

Постановка задачи. Пусть A_N $(|A_N| = N)$ — множество узлов во всей блокчейн-сети; $B_n \subseteq A_N$ $(|B_n| = n)$ — множество узлов, которые потенциально могут быть привлечены к выработке консенсуса; функция $d : T \mapsto Y_{B_n}$, $d = d(t)$, $t \in T$ — заданная функция, где t — независимая переменная, соответствующая текущему времени, значение $d \in Y_{B_n}$ соответствует текущему состоянию B_n в момент времени *t*.

Алгоритм достижения консенсуса должен обеспечить время закрытия каждого блока не более 20 с при числе узлов в сети порядка ~ 10^2 и скорости поступления транзакций ~ 10^4 транзакций в секунду при криптостойкости выше, чем для алгоритмов семейства pBFT.

Алгоритм должен корректно обрабатывать следующие ошибочные ситуации, возникающие в блокчейн-сетях: мастер-узел недоступен; эскорт-узел недоступен, поступила некорректная транзакция; на разных узлах блокчейн-сети разное число закрытых блоков в один и тот же момент времени; узел отвергает новый блок блокчейна; сеть отвергает новый блок блокчейна; появление в сети двух блоков с идентичными номерами.

Исходные положения алгоритма sdBFT. Достижение консенсуса сводится к получению участниками распределенной системы согласованного решения, если некоторое их число не приняло участие в согласовании решения по следующим причинам: ошибка при передаче сообщения о принятии решения одним участником; слишком медленная передача сообщения о принятии решения одним участником; сбой в работе участника в системе; введение в заблуждение при принятии решения в системе как умышленное, так и неумышленное.

Для первых трех причин принятие решения возможно при условии $n > m + 1$ [7], где m — число участников, не привлекаемых к выработке консенсуса, а n — число участников, принявших решение. В случае четвертой причины получаем задачу византийских генералов, которая имеет решение, когда $n > 3m$.

Запишем функцию

$$f : Y_{B_n} \mapsto J_n, \quad f = f\left(d\right), \quad d \in Y_{B_n},$$

где J_n $\left(\left|J_n\right| = n\right)$ — множество номеров узлов из B_n.

Значения функции f определяются следующим образом: вычисляется двойной хеш от двоичного числа $d \in Y_{B_{n'}}$, строится псевдослучайная битовая последовательность $j_1 = H(H(d)), \quad j_2 = H(H(d+1)), \ldots$ [11, 12], получается следующая битовая запись $R = j_1 \| j_2 \|, \ldots$, разделенная последовательно без пропусков и перекрытий на кортежи по r бит в каждом, первые n' из них являются номерами узлов, образующих множество $B_{n'}$, $j_k \in J_n$, $k = 1, \ldots, n'$.

Мастер-узел всегда будет узлом, номер которого $j_{\hat{k}}$ $(\hat{k} = 1)$ сформируется первым. Если случайно повторится уже полученный номер j_k, то повторно полученное число пропускается. Поскольку из B_n случайным образом выбрано подмножество $B_{n'}$ с заданным n' [8–10], тогда $B_{n'} \subset B_n \subseteq A_N$, причем $Y_{B_{n'}}$ — множество значений d, соответствующих всем текущим состояниям $B_{n'}$, $Y_{B_{n'}} \subseteq Y_{B_n}$, где Y_{B_n} — множество значений d, соответствующих всем текущим состояниям B_n.

Для решения проблемы увеличения времени достижения консенсуса с ростом числа узлов, которые потенциально могут быть привлечены к выработке консенсуса, предлагается из B_n выделять подмножество $B_{n'}\left(\left|B_{n'}\right| = n'\right)$ способом, приведенным ранее. Благодаря применению двойного хеширования, номера узлов во множестве B_n при выборе из него

номеров узлов для построения множества $B_{n'}$ окажутся достаточно хорошо «перемешаны» для того, чтобы номера узлов из B_n можно было считать распределенными равномерно, что является хорошо известным свойством хеш-функций и непосредственно следует из [13]. В этом случае есть возможность при $n \gg n'$ решать задачу не на множестве B_n, а на $B_{n'}$, что позволит уменьшить вычислительную сложность алгоритма достижения консенсуса примерно в $[n/n']$ раз и существенно уменьшить время закрытия блока по сравнению с алгоритмами, разработанными ранее.

Поскольку алгоритмы семейства pBFT имеют приемлемое быстродействие при поступлении в блокчейн порядка 10^4 транзакций в секунду лишь при небольшом числе узлов в блокчейне, то нарушителю достаточно решить криптоаналитическую задачу по получению ключей только для 2/3 этого числа узлов. В случае алгоритма spBFT, когда число узлов возрастает в k раз, то в k раз возрастает и сложность криптоаналитической задачи для нарушителя и, соответственно, увеличивается криптостойкость алгоритма sdBFT.

Если b — блок, в отношении которого в некоторый момент времени t' множество узлов $B_{n'}$ стремится достичь консенсуса, то функцию хеширования SHA-3 [13] над этим блоком обозначим $H(b)$, а ее значения обозначим как h. Для вычисления электронной подписи подойдет практически любой современный алгоритм, поскольку его влияние на характеристики основного алгоритма несущественно. В настоящей работе выбран популярный алгоритм вычисления электронной подписи EdDSA с параметрами эллиптической кривой edwards25519 [14, 15]. В принятых обозначениях результат вычисления подписи будет равен $s = \mathrm{sig}(h)$.

Алгоритм sdBFT. Шаг 1. Пусть в момент времени $\hat{t} \in [t,\ t')$ (где $[t,\ t')$ — полуинтервал, на котором должен быть создан блок) узел с номером k осуществляет запись I в блокчейн.

Шаг 2. Выберем все j_k, включая $j_{\hat{k}}$, с помощью функции f. Выработка множества узлов консенсуса осуществляется на полуинтервале $[t,\ t')$.

Шаг 3. В случае признания мастер-узлом включение записи I в блок b допустимым мастер-узел передает всем узлам из $B_{n'}$ эту запись для проверки и включения в блок b. В противном случае запись I отвергается без уведомления.

Шаг 4. Новая запись включается в блок до наступления момента t'. Мастер-узел рассылает сообщение тем же узлам о фиксации блока b. Все узлы из $B_{n'}$ вычисляют значение хеш-функции $H(b)$, равное, допустим, h.

Шаг 5. Каждый узел вычисляет электронную подпись $s_b = (s_1, ..., s_{j_k})$ и передает ее на мастер-узел.

Шаг 6. Мастер-узел ожидает электронные подписи в течение времени Δt после наступления момента t'. В момент $t' + \Delta t$ на мастер-узле формируется кортеж

$$s_b = (s_1, ..., s_{j_k}),\ 1 \le k < n'. \tag{1}$$

Мастер-узел проверяет каждую подпись из (1) и подсчитывает число корректных подписей. Подписи некоторых узлов из $B_{n'}$ могут оказаться голосующими «против» или некорректными в том случае, когда в $B_{n'}$ окажется узел с некоторым номером $j_{\tilde{k}}$, $1 \le \tilde{k} \le n'$, который признает запись некорректной (*а*); в момент времени $\tilde{t} \in [t,\ t')$ имеется состояние блокчейна $\tilde{d}$, отличное от состояния d для узла $j_{\hat{k}}$ (*б*); исказится запись I при формировании блока блокчейна (*в*).

Шаг 7. Мастер-узел вычисляет число корректных подписей μ и проверяет выполнение неравенства

$$\mu > \left\lceil \frac{2}{3} n' \right\rceil. \tag{2}$$

Если (2) не выполняется, то мастер-узел делает вывод, что консенсус не достигнут и транзакция не включается в блок, в противном случае транзакция включается в блок.

Шаг 8. Если текущее время находится в полуинтервале $[t,\ t')$, то следует переход к шагу 1, если нет, то — к шагу 9.

Шаг 9. Для блока b составляется число $b \| s_{k_1} \| \cdots \| s_\mu$ и вычисляется $H\left(b \| s_{k_1} \| \cdots \| s_\mu\right)$, а также sig (электронная подпись узла с номером $j_{\hat{k}}$).

Шаг 10. Число

$$d' = b \| s_1 \| \cdots \| s_\mu \| \mathrm{sig}\left(H\left(b \| s_1 \| \cdots \| s_\mu\right)\right), \tag{3}$$

соответствующее новому закрытому блоку, будем считать новым состоянием блокчейна d' в момент t'. Мастер-узел рассылает его всем узлам множества A_N.

Шаг 11. На каждом узле из A_N осуществляется проверка s_b и sig. Если проверка пройдена, то блок b признается корректным и блокчейн на узле переходит в состояние $d' = d(t')$. Если этот узел не получил блок для проверки в промежутке времени $[t' + \Delta t,\ t' + \Delta t + \lambda]$, где λ — время задержки передачи информации, то узел сочтет консенсус недостигнутым и выберет новое множество $B_{n'}$ на основе старого состояния d.

Сравнение результатов работы алгоритма sdBFT с другими алгоритмами проводилось на примере алгоритма Honey Badger [4]. Он является характерным представителем алгоритмов семейства pBFT, имеет наиболее полное описание и хорошо исследован. На рис. 1 кривые *1*–*6* показывают зависимость времени закрытия блока t_B от скорости v поступления транзакций в блокчейн для алгоритма Honey Badger (pBFT) при 32, 40, 48, 56, 64 и 104 узлах в блокчейне соответственно.

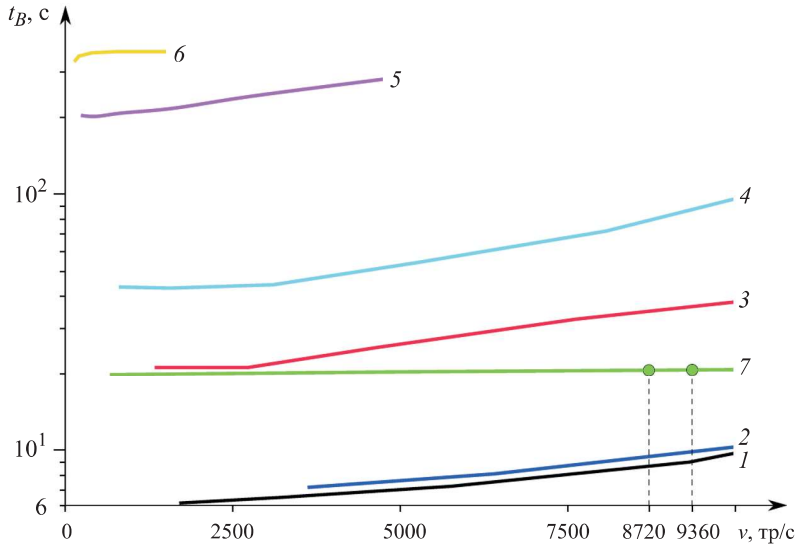

Рис. 1. Зависимости времени закрытия блока от скорости поступления транзакций для алгоритмов Honey Badger (pBFT) и sdBFT

Алгоритм sdBFT исследовался при тех же числах узлов. Результаты во всех шести случаях лежат практически на одной кривой *7*, отклонения от которой для разного числа узлов пренебрежимо малы. Следует обратить внимание на то, что в случае 64 узлов результаты для sdBFT получены при скорости поступления транзакций не более 9360 транзакций в секунду, а в случае 104 узлов — при скорости не более 8720 транзакций в секунду. Это связано с повышением сложности сети передачи данных, что приводит к увеличению времени распространения блоков по сети. В остальном время закрытия блока для алгоритма sdBFT практически не зависит от числа узлов и скорости поступления транзакций (см. рис. 1, кривая *7*) и равно ~ 20 с.

Время t_B для алгоритма sdBFT больше, чем для алгоритма Honey Badger только при довольно малом числе узлов (32 и 40), но, начиная с 48 узлов, алгоритм Honey Badger работает существенно медленнее spBFT: при числе узлов более 56 может обрабатываться не более 5000 транзакций в секунду.

Приблизительное постоянство времени закрытия блока при разном числе узлов в блокчейне и разной скорости поступления транзакций достигнуто благодаря относительно небольшому числу узлов, привлекаемых к выработке консенсуса, что становится возможным в силу равновероятного попадания любого узла из множества B_n в $B_{n'}$.

Пример работы алгоритма sdBFT. 1. Перед началом работы задана компьютерная сеть с узлами, имеющими собственный сетевой адрес и уникальный номер, который знают все участники сети. Допустим, есть 13 участников сети (блоки *1*–*13*, рис. 2). Предположим, что консенсус будут вырабатывать 5 узлов. Будем называть эти условия начальным состоянием компьютерной сети.

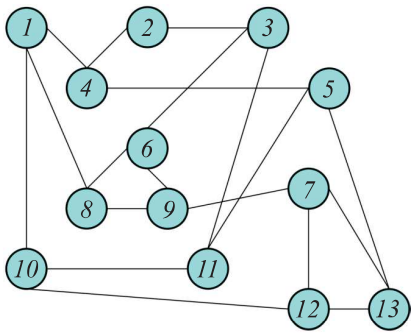

Рис. 2. Диаграмма состояний компьютерной сети

2. Начало работы алгоритма. Пусть все узлы компьютерной сети примут блок *1*. В принятом блоке содержится информация, которая позволит функции f создать случайную последовательность. Пусть эта последовательность будет следующей: *3*, *5*, *7*, *11*, *13*.

3. Узел *3* является мастер-узлом. С этого момента узлы *3*, *5*, *7*, *11*, *13* участвуют в консенсусе.

4. Пусть узел *3* получил новую транзакцию от узла *2*. Узел *3* проверяет, является ли транзакция корректной, если она признается корректной, то узел *3* пересылает ее узлам *5*, *7*, *11*, *13*.

5. Узел *3* продолжает принимать транзакции и рассылать их узлам *5*, *7*, *11* до завершения времени, отведенного на закрытие блока. По завершении этого времени узел *3* пересылает узлам *5*, *7*, *11*, *13* сообщение о закрытии блока.

6. Узлы *5*, *7*, *11* и *13* пересылают хеш дерева Меркла, принятых ими транзакций и свои подписи под хешем узлу *3*.

7. Узел *3* считает подписи, если они корректны и их число удовлетворяет решению задачи византийских генералов и их число не менее трех, то блок считается сформированным. Узел *3* рассылает анонс нового блока всем узлам сети.

8. Узлы принимают блок *2*. Возвращаемся на первый шаг алгоритма, пусть теперь функция f создаст случайную последовательность на основании принятых блоков, пусть это будут номера *6*, *2*, *3*, *11*, *12*. Далее процесс повторяется в соответствии с шагами 1–11 алгоритма.

Обработка ошибочных ситуаций алгоритмом sdBFT

Ситуация 1. Мастер-узел недоступен

1. Компьютерная сеть находится в начальном состоянии.

2. Начало работы алгоритма. Пусть все узлы пиринговой сети примут блок *1*. В принятом блоке содержится информация, которая позволит функции f создать случайную последовательность. Пусть эта последовательность будет следующей: *3*, *5*, *7*, *11*, *13*. Узел *3* недоступен.

3. Узлы эскорта не получат сообщения о закрытии блока, блокчейн-сеть переходит на следующий раунд.

4. Возвращаемся на первый шаг алгоритма, пусть теперь функция f создаст случайную последовательность на основании принятого блока и номера раунда, пусть будут номера *7*, *1*, *4*, *12*, *13*.

5. Далее алгоритм будет исполняться штатным образом в соответствии с шагами 1–11 алгоритма.

Ситуация 2. Эскорт-узел недоступен

1. Компьютерная сеть находится в начальном состоянии.

2. Начало работы алгоритма. Пусть все узлы пиринговой сети примут блок *1*. В принятом блоке содержится информация, которая позволит функции f создать случайную последовательность. Пусть эта последовательность будет следующей: *3*, *5*, *7*, *11*, *13*. Узел *13* недоступен.

3. Узел *13* эскорта не получит сообщения о закрытия блока и не отправит свою подпись для закрытия блока. Если оставшиеся три узла отправят корректные подписи транзакций формируемого блока, то блок будет сформирован. Узлы, участвующие в консенсусе, будут перевыбраны.

4. Если оставшиеся три узла отправят некорректные подписи транзакций формируемого блока, то блок не будет сформирован. Блокчейн перейдет на следующий раунд. Узлы, участвующие в консенсусе, будут перевыбраны.

Ситуация 3. Поступила некорректная транзакция

1. Пусть узел *3* получил новую транзакцию от узла *2*. Узел *3* проверяет транзакцию и признает ее некорректной.

2. Если узел *3* признал транзакцию некорректной, то эта транзакция не обрабатывается.

Ситуация 4. Число блоков в блокчейне разное на разных узлах. Число принятых блоков на разных узлах блокчейна может быть разным в случае, например, если сеть была сегментирована и не все узлы успели синхронизироваться.

1. Компьютерная сеть в начальном состоянии.

2. Начало работы алгоритма. Пусть все узлы компьютерной сети примут блок *1*. В принятом блоке содержится информация, которая позволит функции f создать случайную последовательность. Пусть эта последовательность будет следующей: *3*, *5*, *7*, *11*, *13*, а узлы *7* и *11* имеют отличное от узлов *3*, *5*, *13* число принятых блоков.

3. Узел *3* является мастер-узлом. С этого момента узлы *3*, *5*, *13* участвуют в консенсусе.

4. Пусть узел *3* получил новую транзакцию от узла *2*. Узел *3* проверяет, является ли транзакция корректной, если она признается корректной, то узел *3* пересылает ее узлам *5*, *7*, *11*, *13*. Узлы *7* и *11* отвергают транзакцию.

5. По истечении времени на закрытие блока узел *3* пересылает узлам *5*, *7*, *11*, *13* сообщение о закрытии блока, узлы *7* и *11* отвергают сообщение. Узлы *5* и *13* пересылают хеш транзакций и свои подписи под хешем узлу *3*.

6. Узел *3* проверяет подписи узлов эскорта. Поскольку число подписей узлов эскорта недостаточно для принятия блока, то блокчейн переходит на следующий раунд.

Ситуация 5. Узел отвергает новый блок блокчейна. Узел сети может отвергнуть новый блок блокчейна. Причин, по которым узел отвергает блок, может быть несколько, например, на узле вследствие программного или аппаратного сбоя возникла ошибка чтения из базы данных и балансы кошельков изменились. Повторим шаг 11 алгоритма.

1. Узел *3* рассылает анонс нового блока всем узлам сети.

2. Пусть узел *1* отвергает блок.

3. Узел *1* пытается найти в сети блок с хеш-суммой, отличной от хеш-суммы признанного им ошибочным блока. Если узел не может найти удовлетворяющий его блок, то узел начинает процедуру пересинхронизации блокчейна (шаг 11 алгоритма).

Ситуация 6. Сеть отвергает новый блок блокчейна. При создании нового блока может произойти сознательная попытка группы узлов навязать собственный, ошибочный блок. Пусть узлы *3*, *5*, *7*, *11*, *13* пытаются навязать собственный, неправильный блок сети блокчейна. Повторим шаг 11 алгоритма.

1. Узел *3* рассылает анонс нового блока всем узлам сети.

2. Все узлы сети отвергают новый блок.

3. Блокчейн переходит на следующий раунд и далее, пока не будет корректно сформирован следующий блок.

Ситуация 7. В сети появилось два блока с идентичными номерами. Два блока с идентичными номерами могут возникнуть в сети, только если

будет сформировано два консенсуса из узлов с разными раундами, а это возможно, если в сети возник глобальный сбой. Например, часть узлов находилась на серверах, которые были одновременно перезагружены. В таком случае будет происходить следующее.

1. Предположим, что сформировалось два набора узлов для создания консенсуса — *3*, *5*, *7*, *11*, *13* и *8*, *9*, *6*, *1*, *10*. Узлы *3* и *8* рассылают анонс нового блока всем узлам сети.

2. Узел при принятии нового блока проверяет, входит ли раунд создания блока в доверительный интервал раундов или нет, т. е. насколько сильно раунд нового блока отличается от собственного раунда узла. Если раунд признается узлом корректным, то блок принимается, если признается некорректным, то — отвергается. Далее возможно два пути развития ситуации:

– раунды сформированных блоков находятся в доверительном интервале, в таком случае узлом будет принят блок, пришедший первым. Вероятность попадания в блокчейн блока будет зависеть от того, какой блок был принят большинством узлов сети;

– раунд одного сформированного блока не находится в доверительном интервале у большинства узлов сети. Следовательно, большинством узлов будет принят блок с номером раунда из доверительного интервала.

Заключение. Алгоритм sdBFT достижения консенсуса позволяет увеличить в несколько раз число узлов сети, имеющих возможность участвовать в достижении консенсуса при высокой скорости поступления транзакций, по сравнению с существующими алгоритмами семейства pBFT. Алгоритм sdBFT обеспечивает время закрытия блока, равное 20 с, при поступлении ~ 10^4 транзакций в секунду в сеть с числом узлов ~ 10^2. При этом алгоритм sdBFT корректно обрабатывает следующие ошибочные ситуации, возникающие в больших блокчейн-сетях: мастер-узел недоступен; эскорт-узел недоступен, поступление некорректной транзакции; образование на разных узлах блокчейн-сети разного числа закрытых блоков в один и тот же момент времени; узел отвергает новый блок блокчейна; сеть отвергает новый блок блокчейна; появление в сети двух блоков с идентичными номерами. С учетом высокой криптографической стойкости этого алгоритма можно сделать вывод о том, что на сегодняшний день он является одним из наиболее конкурентоспособных алгоритмов в больших блокчейн-сетях, функционирующих на основе достижения консенсуса.

ЛИТЕРАТУРА

[1] Satoshi N. Bitcoin: a peer-to-peer electronic cash system. *bitcoin.org: веб-сайт*. URL: https://bitcoin.org/en/bitcoin-paper (дата обращения: 20.01.2018).

[2] Transactions speeds: how do cryptocurrencies stack up to Visa or PayPal? *howmuch.net: веб-сайт*. URL: https://howmuch.net/articles/crypto-transaction-speeds-compared (дата обращения: 12.08.2019).

[3] Proof of stake versus proof of work. *bitfury.com: веб-сайт*.
URL: http://bitfury.com/content/5-white-apersresearch/pos-vs-pow-1.0.2.pdf (дата обращения: 02.10.2019).

[4] Miller A., Xia Y., Croman K., et al. The honey badger of BFT protocols. *eprint.iacr.org: веб-сайт*. URL: https://eprint.iacr.org/2016/199.pdf (дата обращения: 26.05.2020).

[5] Consensus algorithm for bigger blockchain networks. *sumustech.com: веб-сайт*. URL: https://files.sumustech.com/doc/1-86cd7815.pdf (дата обращения: 14.04.2018).

[6] Goldmint blockchain solutions: consensus algorithm designed by Sumus team for bigger blockchain networks. *blog.goldmint.io: веб-сайт*.
URL: https://blog.goldmint.io/goldmint-blockchain-solutions-consensus-algorithm-designed-by-sumus-team-for-bigger-blockchain-6ace5fd3ee6d
(дата обращения: 30.10.2020).

[7] Lamport L., Shostak R., Pease M. The byzantine generals' problem. *ACM Trans. Program. Lang. Syst.*, 1982, vol. 4, no. 3, pp. 382–401.
DOI: https://doi.org/10.1145/357172.357176

[8] Деон А.Ф., Меняев Ю.А. Полное факториальное моделирование равномерных последовательностей целых случайных величин. *Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение*, 2017, № 5 (116), с. 132–149.
DOI: http://doi.org/10.18698/0236-3933-2017-5-132-149

[9] Applebaum B. Pseudorandom generators with long stretch and low locality from random local one-way functions. *Proc. 44th Ann. ACM STOC,* 2012, pp. 805–816. DOI: https://doi.org/10.1145/2213977.2214050

[10] Classen K., Palka M.H. Splittable pseudorandom number generators using cryptographic hashing. *Proc. 2013 ACM SIGPLAN Symp. Haskell*, 2013, pp. 47–58. DOI: https://doi.org/10.1145/2503778.2503784

[11] Варфоломеев А.А. Некоторые рекомендации по повышению стойкости шифра с малым размером ключа к методу полного опробования. *Вопросы кибербезопасности*, 2015, № 5, с. 60–62.

[12] Hattab S., Taha Alyaseen I.F. Consensus Algorithms Blockchain: a comparative study. *IJPCC*, 2019, vol. 5, no. 2, pp. 66–71. DOI: https://doi.org/10.31436/ijpcc.v5i2.103

[13] SHA-3 standard: permutation-based hash and extendable-output functions. *NIST*, 2015. DOI: http://doi.org/10.6028/NIST.FIPS.202

[14] Josefsson S., Liusvaara I. Edwards-curve digital signature algorithm (EdDSA). URL: https://datatracker.ietf.org/doc/html/rfc8032 (дата обращения: 08.04.2018).

[15] Варфоломеев А.А. Реализация одной схемы цифровой подписи по доверенности на основе российских стандартов. *Безопасность информационных технологий*, 2010, т. 17, № 1, с. 50–51.

Бардин Алексей Павлович — директор по исследованиям и разработкам компании Sumus Company Limited (2103, 21/F, Tower 1, Lippo Centre, 89 Queensway, Admiralty, Hong Kong).

Новицкий Анатолий Викторович — технический директор компании Sumus Company Limited (2103, 21/F, Tower 1, Lippo Centre, 89 Queensway, Admiralty, Hong Kong).

Шумилов Юрий Юрьевич — д-р техн. наук, профессор кафедры «Прикладная математика» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1); профессор департамента информационной безопасности Финансового университета (Российская Федерация, 125993, Москва, Ленинградский пр-т, д. 49).

Просьба ссылаться на эту статью следующим образом:

Бардин А.П., Новицкий А.В., Шумилов Ю.Ю. Обработка ошибочных ситуаций в больших блокчейн-сетях алгоритмом достижения консенсуса, основанном на решении задачи византийских генералов. *Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение*, 2021, № 4 (137), с. 27–40.
DOI: https://doi.org/10.18698/0236-3933-2021-4-27-40

PROCESSING ERRONEOUS SITUATIONS IN LARGE BLOCKCHAIN NETWORKS BY A CONSENSUS ALGORITHM BASED ON THE BYZANTINE GENERALS' PROBLEM SOLUTION

A.P. Bardin[1] lovvi@mail.ru
A.V. Novitsky[1] a@sumus.team
Yu.Yu. Shumilov[2, 3] shm@bmstu.ru

[1] **Sumus Company Limited, Hong Kong, PRC**
[2] **Bauman Moscow State Technical University, Moscow, Russian Federation**
[3] **Financial University, Moscow, Russian Federation**

Abstract

Blockchain networks built on the basis of PoS, DPoS, LPoS, PoE, PoIT, pBFT algorithms have certain limitations associated with the reduction of the new transactions inclusion rate in the blockchain when the number of blockchain network nodes participating in block creation increases. The developed stake distributed Byzantine Fault Tolerant (sdBFT) algorithm of consen-

Keywords

Blockchain-consensus, transaction, node, block, hash function, digital signature

sus achievement allows to increase by several orders of magnitude the number of network nodes participating in consensus achievement in comparison to existing BFT algorithms, while maintaining the speed f transactions inclusion in the blockchain. The main erroneous situations in large blockchain networks are investigated in terms of their processing by the sdBFT algorithm. The processing of seven main erroneous situations in a blockchain is given: a master node is not available; an escort node is not available; an incorrect transaction is received; different number of closed blocks on different nodes of a blockchain network at the same time; a node rejects a new blockchain block; a network rejects a new blockchain block; two blocks with identical numbers appear in the network. It is shown that the correct interaction of the sdBFT algorithm with the blockchain network enables not only high performance with a large number of nodes and high transaction arrival rate, but also the ability of the blockchain using this algorithm to work out all the major error situations that arise in the operation of the blockchain network

Received 09.12.2020
Accepted 28.06.2021

REFERENCES

[1] Satoshi N. Bitcoin: a peer-to-peer electronic cash system. *bitcoin.org: website*.
Available at: https://bitcoin.org/en/bitcoin-paper (accessed: 20.01.2018).

[2] Transactions speeds: how do cryptocurrencies stack up to Visa or PayPal? *howmuch.net: website*. Available at: https://howmuch.net/articles/crypto-transaction-speeds-compared (accessed: 12.08.2019).

[3] Proof of stake versus proof of work. *bitfury.com: website*.
Available at: http://bitfury.com/content/5-white-apersresearch/pos-vs-pow-1.0.2.pdf (accessed: 02.10.2019).

[4] Miller A., Xia Y., Croman K., et al. The honey badger of BFT protocols. *eprint.iacr.org: website*. Available at: https://eprint.iacr.org/2016/199.pdf (accessed: 26.05.2020).

[5] Consensus algorithm for bigger blockchain networks. *sumustech.com: website*. Available at: https://files.sumustech.com/doc/1-86cd7815.pdf (accessed: 14.04.2018).

[6] Goldmint blockchain solutions: consensus algorithm designed by Sumus team for bigger blockchain networks. *blog.goldmint.io: website*.
Available at: https://blog.goldmint.io/goldmint-blockchain-solutions-consensus-algorithm-designed-by-sumus-team-for-bigger-blockchain-6ace5fd3ee6d (accessed: 30.10.2020).

[7] Lamport L., Shostak R., Pease M. The byzantine generals' problem. *ACM Trans. Program. Lang. Syst.*, 1982, vol. 4, no. 3, pp. 382–401.
DOI: https://doi.org/10.1145/357172.357176

[8] Deon A.F., Menyaev Yu.A. Complete factorial simulation of integer random number uniform sequences. *Herald of the Bauman Moscow State Technical University, Series Instrument Engineering*, 2017, no. 5 (116), pp. 132–149 (in Russ.).
DOI: http://doi.org/10.18698/0236-3933-2017-5-132-149

[9] Applebaum B. Pseudorandom generators with long stretch and low locality from random local one-way functions. *Proc. 44th Ann. ACM STOC*, 2012, pp. 805–816.
DOI: https://doi.org/10.1145/2213977.2214050

[10] Classen K., Palka M.H. Splittable pseudorandom number generators using cryptographic hashing. *Proc. 2013 ACM SIGPLAN Symp. Haskell*, 2013, pp. 47–58.
DOI: https://doi.org/10.1145/2503778.2503784

[11] Varfolomeev A.A. Some recommendations for improving security of the cipher with small key against brute force attack. *Voprosy kiberbezopasnosti* [Cybersecurity Issues], 2015, no. 5, pp. 60–62 (in Russ.).

[12] Hattab S., Taha Alyaseen I.F. Consensus Algorithms Blockchain: a comparative study. *IJPCC*, 2019, vol. 5, no. 2, pp. 66–71. DOI: https://doi.org/10.31436/ijpcc.v5i2.103

[13] SHA-3 standard: permutation-based hash and extendable-output functions. *NIST*, 2015. DOI: http://doi.org/10.6028/NIST.FIPS.202

[14] Josefsson S., Liusvaara I. Edwards-curve digital signature algorithm (EdDSA). Available at: https://datatracker.ietf.org/doc/html/rfc8032 (accessed: 08.04.2018).

[15] Varfolomeev A.A. The realization of one proxy digital signature scheme on the base of Russian standards. *Bezopasnost' informatsionnykh tekhnologiy* [IT Security], 2010, vol. 17, no. 1, pp. 50–51 (in Russ.).

Bardin A.P. — Chief Research and Development Officer, Sumus Company Limited (2103, 21/F, Tower 1, Lippo Centre, 89 Queensway, Admiralty, Hong Kong).

Novitsky A.V. — Chief Technology Officer, Sumus Company Limited (2103, 21/F, Tower 1, Lippo Centre, 89 Queensway, Admiralty, Hong Kong).

Shumilov Yu.Yu. — Dr. Sc. (Eng.), Professor, Department of Applied Mathematics, Bauman Moscow State Technical University (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation); Professor, Department of Information Security, Financial University (Leningradskiy prospekt 49, Moscow, 125993 Russian Federation).

Please cite this article in English as:

Bardin A.P., Novitsky A.V., Shumilov Yu.Yu. Processing erroneous situations in large blockchain networks by a consensus algorithm based on the Byzantine generals' problem solution. *Herald of the Bauman Moscow State Technical University, Series Instrument Engineering*, 2021, no. 4 (137), pp. 27–40 (in Russ.).
DOI: https://doi.org/10.18698/0236-3933-2021-4-27-40