УДК 004.657

© 2014 г. **Ю.А. Григорьев,** д-р техн. наук,
**Е.В. Цвященко**
(Московский государственный технический университет им. Н.Э. Баумана)

# АНАЛИЗ ХАРАКТЕРИСТИК СОГЛАСОВАНИЯ РЕПЛИК В КОНЕЧНОМ СЧЕТЕ В БАЗАХ ДАННЫХ NOSQL

В статье рассмотрены проблемы репликации баз данных NoSQL. Разработана модель доступа к базе данных для случая слабой согласованности (W = R = 1). Получена оценка вероятности, что в процессе обновления N реплик поступит хотя бы одно требование на чтение из необновленных реплик. Приведен анализ полученных результатов.
***Ключевые слова:*** база данных NoSQL, согласованность в конечном счете, сильная согласованность, преобразование Лапласа-Стилтьеса, производящая функция, вероятность доступа.

## Введение

Для повышения производительности, отказоустойчивости и других параметров автоматизированных информационных систем (АИС) в настоящее используются параллельные системы баз данных SQL, построенные на основе реляционной модели. К основным недостаткам хранилищ на основе реляционной модели можно отнести: ограниченную возможность обработки неструктурированных данных (UDF), невысокую масштабируемость (до 100 узлов), небольшое число реплик (до двух), а также необходимость перезапуска системы после сбоя.

В последнее время начали появляться и уже получили широкую известность базы данных, построенные на парадигме распределенных хранилищ «ключ/значение» и получившие название NoSQL (Not-Only-SQL) [1]. Эти системы обеспечивают доступ к неструктурированным данным (текст, видео и др.) посредством выполнения MapReduce заданий. Основными преимуществами хранилищ являются: высокая масштабируемость (Hadoop – 4000 узлов, Riak – 6000 узлов), большое число реплик, а, следовательно, высокая надежность.

Как отмечалось в [2], базы данных NoSQL имеют один серьезный недостаток: они не поддерживают режим ведения транзакций. Поэтому в таких системах возникает проблема обеспечения свойства согласованности данных. В статье исследуется согласованность в конечном счете хранилища NoSQL с параметрами W = R = 1 [2]. В работе приводится вывод формулы для оценки вероятности, что

за время обновления N реплик придет хотя бы одно требование на чтение записей из несогласованных реплик. Выполнен анализ полученных результатов.

## Проблемы репликации данных в NoSQL

В базах данных NoSQL широко используется репликация данных. Степень идентичности данных в каждой из реплик отражает согласованность данных. Существуют различные виды согласованности. При сильной (строгой) согласованности в каждый момент времени гарантируется чтение последних обновлений (достигается за счет выбора параметров W = R = N/2 + 1 [2]). При слабой согласованности гарантируется, что все реплики будут идентичны в конечном счете (параметры W = R = 1). При этом не гарантируется чтение последних обновлений.

На практике согласованностью можно управлять с помощью параметров N, W, R [2], где N – число реплик, на которые данные будут реплицированы в конечном счете (фактор репликации). Величина W определяет, на скольких репликах операция записи должна завершиться успешно, прежде чем можно будет считать успешной операцию в целом (фактор записи). Наконец, R – число реплик, откуда должны бать прочитаны данные, чтобы операция чтения считалась успешной (фактор чтения). Чтение в узле начинается только после заверения обновления соответствующей записи в этом узле. Так как, согласно теореме Брювера [3], одновременное поддержание свойств доступности, согласованности и устойчивости к потере связности невозможно, а от третьего свойства отказаться в NoSQL не представляется возможным, то необходимо искать компромисс между доступностью и согласованностью. Однако часто параметров системы N, W, R недостаточно, чтобы достичь оптимального баланса между производительностью, надежностью и доступностью. Рассмотрим основные проблемы, связанные с использованием репликации для поддержания требуемого уровня (гарантий) согласованности и производительности.

*Проблема согласования данных.* Если число N велико, то имеет место проблема согласования данных.

Рассмотрим случай согласованности в конечном счете. Как показано на рис. 1а, обновления выполняются в фоновом режиме. Следовательно, вероятность чтения устаревших данных возрастает при увеличении общего числа реплик N. Рост вероятности чтения устаревших данных приводит к уменьшению степени согласования данных.

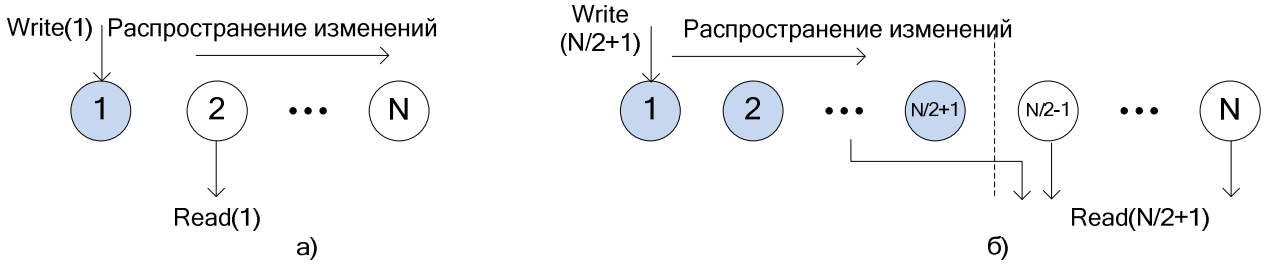


*Рис. 1.* Проблема согласования данных:
а) согласованность в конечном счете; б) сильная согласованность.

Рассмотрим случай сильной согласованности. Как показано на рис. 1б, об-

новление N/2 + 1 реплик может выполняться не в фоном режиме, как в первом случае, а в процессе выполнения запроса на чтение к NoSQL. В этом случае время ожидания начала чтения из обновляемой реплики возрастает при увеличении общего числа реплик N, но при этом свойство согласованности гарантировано.

*Проблема низкой производительности.* Если число N мало – имеет место проблема низкой производительности.

Рассмотрим случай согласованности в конечном счете. Уменьшение общего числа реплик, как показано на рис. 2а, ведет к увеличению нагрузки на узлы при фиксированной интенсивности поступления требований на чтение и, как следствие, – к уменьшению производительности системы.

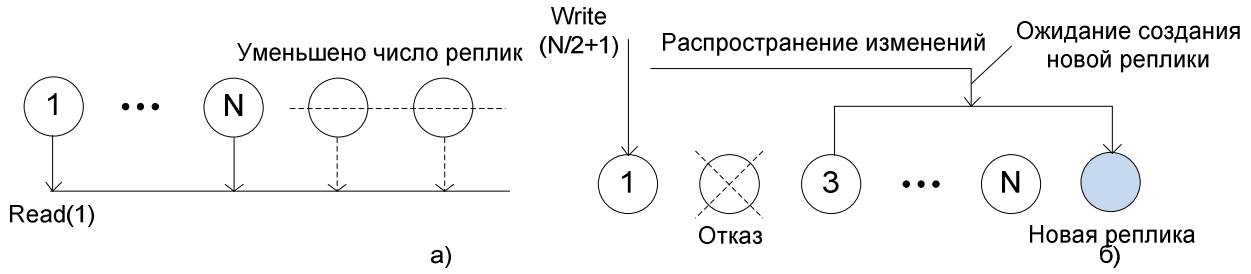


*Рис. 2.* Проблема низкой производительности:
а) согласованность в конечном счете; б) сильная согласованность.

В случае сильной согласованности достигается максимальная идентичность данных в любой момент времени. При фиксированном N отказ какого-либо узла с репликой данных может привести к резкому увеличению времени отклика системы. Это связано с ожиданием создания новой реплики (рис. 2б).

Из приведенных выше примеров можно сделать вывод о сложности выбора параметров N, W, R на стадии проектирования и о необходимости их исследования.

## Модель анализа рассогласования реплик для случая W = R = 1

Ранее в большинстве работ, например, в [4], показатели согласованности измерялись экспериментально. Эксперимент состоял в последовательном считывании записи до того момента, пока устаревшее значение не перестанет возвращаться. Разница между временем последнего считывания устаревшей версии пары <ключ/значение> и временем последнего обновления ключа отражала окно рассогласованности. С целью повышения производительности чтения данных использовались несколько географически распределенных читающих процессов, так как существует вероятность, что все запросы из одной географической зоны могут перенаправляться на одни и те же узлы распределенной БД. Но при использовании нескольких считывающих процессов возникает проблема, связанная с синхронизацией часов на разных узлах. Поскольку окно рассогласованности определяется временным показателем, то часы должны быть точно синхронизированы, что сложно осуществить на практике.

В [5] предлагается подход к количественному измерению показателей согласованности через «вероятностно ограниченное устаревание» записи (пары <ключ/значение>). Вероятностно ограниченное устаревание оценивается сле-

дующими показателями: k – устаревание; t – видимость; (k,t) – устаревание.

Вероятность, что считанный из распределенного хранилища набор версий записей не будет содержать актуальную версию записи является функцией, зависящей от времени. Однако в формуле (1) статьи [5] это постоянная величина. В формуле (2, [5]) вероятность, что считанный из распределенного хранилища набор версий записей будет содержать одну из версий из последних k-обновлений также является постоянной величиной, не зависящей от времени. Эта ошибка следует из формулы (1, [5]). В формуле (3, [5]) не ясно, как получить функцию $P_w(W_r,t)$. Также в формуле содержится элемент $P_s$ из формулы (1, [5]), который уже содержит ошибку. В формуле (4, [5]) ошибка вытекает из описанных выше рассуждений. Таким образом, возникает задача разработки новой модели анализа рассогласования реплик, которая не имела бы указанных недостатков.

На рис. 3 представлена модель рассогласования данных для случая W = R = 1. Предполагается, что используется синхронный режим обновления реплик, т.е. координатор ожидает завершения обновления записи в очередной реплике.

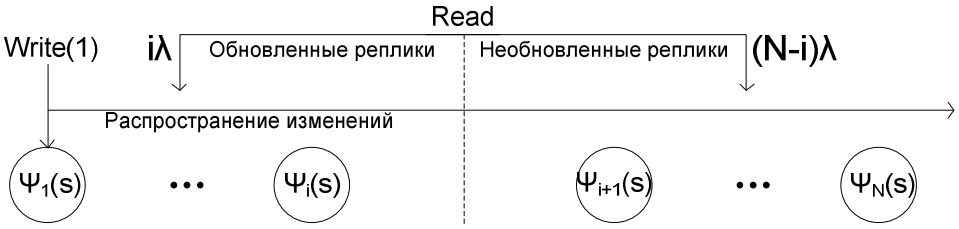


*Рис. 3.* Модель рассогласования данных для случая W=R=1.

Входящий поток требований на чтение из одной реплики принимается пуассоновским с параметром λ, $\Psi_i(s)$ – преобразование Лапласа-Стилтьеса времени обновления *i*-й реплики. В каждый момент времени с суммарной интенсивностью iλ поступают требования на чтение из уже обновленных реплик и с интенсивностью (N – i)λ – из необновленных реплик. Задача состоит в том, чтобы оценить вероятность, что за время обновления N реплик поступит хотя бы одно требование на чтение записей из необновленных реплик.

Для решения поставленной задачи предлагается сначала получить производящую функцию (ПФ) $W_{i+1}(z)$ числа требований на чтение записей из (N – i) необновленных реплик, которые пришли за случайное время обновления (i + 1)-й реплики, i = 0, ..., (N – 1) – число уже обновленных реплик. Основываясь на свойствах пуассоновского входящего потока [6], имеем:

$$W_{i+1}(z) = \sum_{m=0}^{\infty} z^m \cdot g_m = \sum_{m=0}^{\infty} z^m \cdot \int_0^{\infty} p(v_t = m) dF(t) =$$

$$= \int_0^{\infty} \sum_{m=0}^{\infty} z^m \cdot p(v_t = m) dF(t) = \int_0^{\infty} e^{-\lambda(\mathrm{N-i})(1-z)t} dF(t) = \qquad (1)$$

$$= \psi_{i+1}(\lambda(N-i)(1-z)).$$

В соответствии с вероятностным смыслом производящей функции [6] $(1 - W_{i+1}(0))$ – это вероятность, что за время обновления (i + 1)-й реплики поступит хотя бы одно требование на чтение из (N – i) необновленных реплик.

Таким образом, вероятность, что за время обновления N реплик поступит хотя бы одно требование на чтение записей из необновленных реплик равна:

$$P = (1 - W_1(0)) + \sum_{i=2}^{N}((1 - W_i(0)) \cdot \prod_{j=1}^{i-1} W_j(0)). \quad (2)$$

Выражение (2) получено путем рекурсивного использования формулы полной вероятности.

## Оценка времени обновления реплики

Ниже приведено преобразование Лапласа-Стилтьеса (ПЛС) $\Psi_{i+1}(s)$ функции распределения вероятностей времени обновления (i + 1)-й реплики.

Время обновления реплики складывается из двух составляющих: сетевой (передача данных между координатором и репликой) и локальной. Обозначим через $\Lambda(s,r,t)$ ПЛС сетевой составляющей, а через $\Theta(s)$ ПЛС – локальной составляющей времени обновления. Имеем

$$\Lambda_{i+1}(s,t,r) = G_{kv}^{t}(\phi_m^2(s) \cdot \phi_n^2(s) \cdot \phi_{ns}^{r}(s)) \cdot G_{kv}^{1-t}(\phi_m(s)), \quad (3)$$

где параметр $t = 1$, если узел, содержащий (i + 1) реплику, не совпадает с координатором (имеет место передача данных по сети), 0 – иначе (передача данных в памяти). Параметр r = 1, если узел, содержащий (i + 1) реплику, находится в подсети, не содержащей координатор, 0 – иначе. Здесь использованы следующие обозначения: $\varphi_m(s)$ – ПЛС времени чтения одного байта данных из ОП; $\varphi_m^2(s)$ учитывает передачу данных из ОП в буфер сетевого адаптера (СА) до передачи по сети и передачу данных из буфера СА в буфер ОП после передачи по сети; $\varphi_n(s)$ – ПЛС времени перемещения байта данных по локальной сети между станцией и коммутатором; $\varphi_{ns}(s)$ – ПЛС времени перемещения байта данных по сети, соединяющей подсети; $G_{kv}(z)$ – производящая функция (ПФ) объема данных (в байтах) пары ключ/значение. Будем считать, что подсети имеют одинаковую пропускную способность. Ключ – 20-байтное число (RIAK), 16-байтное (Cassandra, Dynamo (MD5)).

По умолчанию в RIAK используется локальное хранилище Bitcask. Остановимся более подробно на локальной составляющей:

$$\Theta(s) = \phi_{crc}(s) \cdot G_{do}(\phi_{do}(s) \cdot \phi_m(s)) \cdot \phi_{kd}(s), \quad (4)$$

где $\varphi_{crc}(s)$ – ПЛС времени подсчета контрольной суммы; $G_{do}(z)$ – ПФ объема данных в байтах, занимаемых информацией <CRC, временная метра, длина ключа, ключ, длина значения, значение>; $\varphi_{do}(s)$ – ПЛС времени дискового вывода одного байта; $\varphi_{kd}(s)$ – ПЛС времени обновления хеш-таблицы (keydir) в ОП, не зависит от размера значения, так как хеш-таблица строится по ключу, значение которого соответствует структуре <идентификатор файла, размер значения, смещение значения, временная метка>.

Таким образом, ПЛС времени обновления реплики можно определить как:

$$\Psi_{i+1}(s) = \Lambda_{i+1}(s,r,t) \cdot \Theta(s), \quad (5)$$

где $\Lambda(s,r,t)$ и $\Theta(s)$ определяются выражениями (3) и (4).

Ниже приведены формулы для описанных ПЛС:

$$\phi_{crc}(s) = G_{crc}(\phi_p^4(s) \cdot \phi_m(s)), \quad (6)$$

$$G_{crc}(z) = G_{kv}(z) \cdot z^{12}, \quad (7)$$

где $\varphi_p(s)$ – ПЛС времени выполнения одной операции процессором.

*Примечание.* Будем считать, что кэш процессора – это «черная дыра», в которой сохраняются данные процессора, необходимые для вычислений.

Поясним (6) и (7). На один байт данных для расчета контрольной суммы требуется 4 процессорные операции [7]. Далее длина кортежа <временная метка, длина ключа, длина значения> составляет 12 байт:

$$\phi_{kd}(s) = G_k(\phi_m(s)) \cdot \phi_p^{PRT}(s), \quad (8)$$

где PRT – количество итераций, необходимых для вычисления хеша ключа. Преобразование строки в число выполняется псевдослучайным преобразованием.

Расчеты показывают, что значение PRT примерно равно 16:

$$G_{do}(z) = G_{kv}(z) \cdot z^{16}, \quad (9)$$

где $G_k(z)$ – ПФ размера ключа (в байтах).

Поясним (9). Длина кортежа <CRC, временная метка, длина ключа, длина значения> составляет 16 байт.

Остальные формулы представлены в табл. 1.

*Таблица 1*

| $\phi_n(s)$ | $\phi_{ns}(s)$ | $\phi_{do}(s)$ | $\phi_m(s)$ | $\phi_P(s)$ |
|---|---|---|---|---|
| $\frac{\mu_n}{\mu_n + s}$ | $\frac{\mu_{ns}}{\mu_{ns} + s}$ | $\frac{\mu_{do}}{\mu_{do} + s}$ | $\frac{\mu_m}{\mu_m + s}$ | $\frac{\mu_P}{\mu_P + s}$ |

Здесь $\mu_P$ – число процессорных циклов в секунду; $\mu_{do}$ – интенсивность вывода данных на диск (байт/сек); $\mu_m$ – интенсивность чтения информации из ОП (байт/сек); $\mu_n$ – интенсивность передачи данных по шине локальной сети (байт/сек); $\mu_{ns}$ – интенсивность передачи данных по шине сети, соединяющей подсети (байт/сек).

## Оценка среднего значения времени обновления реплики

Дифференцируя (1) как сложную функцию по z в 1, получим математическое ожидание числа требований на чтение, которые поступят за время обновления (i+1)-й реплики:

$$M = Q_m \cdot \overline{\phi_m} + Q_p \cdot \overline{\phi_p} + Q_{do} \cdot \overline{\phi_{do}} + Q_n \cdot \overline{\phi_n} + Q_{ns} \cdot \overline{\phi_{ns}}, \quad (10)$$

где для расчета средних величин $\overline{\phi_m}$, $\overline{\phi_p}$, $\overline{\phi_{do}}$, $\overline{\phi_n}$, $\overline{\phi_{ns}}$ используются формулы из табл. 1 (путем дифференцирования в нуле). Коэффициенты при средних величинах представлены в табл. 2, где K – размер ключа; V – размер значения.

*Таблица 2*

| $Q_m$ | $Q_p$ | $Q_{do}$ | $Q_n$ | $Q_{ns}$ |
|---|---|---|---|---|
| (K+V)·(t+1)+3K+2V+28 | 4·(K+V+12)+ +PRT·K | K+V+16 | 2t·(K+V) | t·r·(K+V) |

## Анализ вероятности доступа к рассогласованным данным

Ниже приведены результаты анализа вероятности поступления хотя бы одного требования на чтение записей из необновленных реплик (2).

Характеристики ресурсов (интенсивности обработки) были получены с помощью программы синтетических тестов AIDA64 [8]. Расчеты были выполнены при следующих значениях характеристик ресурсов:

1) процессор – Mobile DualCore Intel Core i5-2450M, 2900 MHz, для которого значение числа процессорных циклов (выполняемых секунду) $\mu_P = 2900 \cdot 10^6$ (1/с);

2) внешняя память Momentus 5400 640423 Seagate <ST9640423AS> 5400rpm 16Mb; интенсивность ввода/вывода данных на диск $\mu_{do} = 85 \cdot 1024 \cdot 1024$ (байт/с);

3) оперативная память – DDR3-1333 PC3-10667; интенсивность чтения данных из ОП $\mu_m = 9842 \cdot 1024 \cdot 1024$ (байт/с);

4) производительность локальной сети внутри сегмента сети равна 1 Гбит/с; интенсивность передачи данных по шине локальной сети $\mu_n = 125 \cdot 10^6$ (байт/с).

5) производительность сети между ее сегментами составляет 64 Мбит/с; интенсивность передачи данных по шине сети, соединяющей подсети $\mu_{ns} = 8 \cdot 10^6$ (байт/с).

На рис. 4 показана зависимость вероятности поступления хотя бы одного требования на чтение из необновленных реплик от интенсивности входящих запросов при различных значениях N. Общее число физических узлов – 20, узлы разделены на два сегмента сети, по 10 узлов в каждом. Число виртуальных узлов – 20. Размер ключа K составляет 20 байтов; размер значения V – 512 байтов.

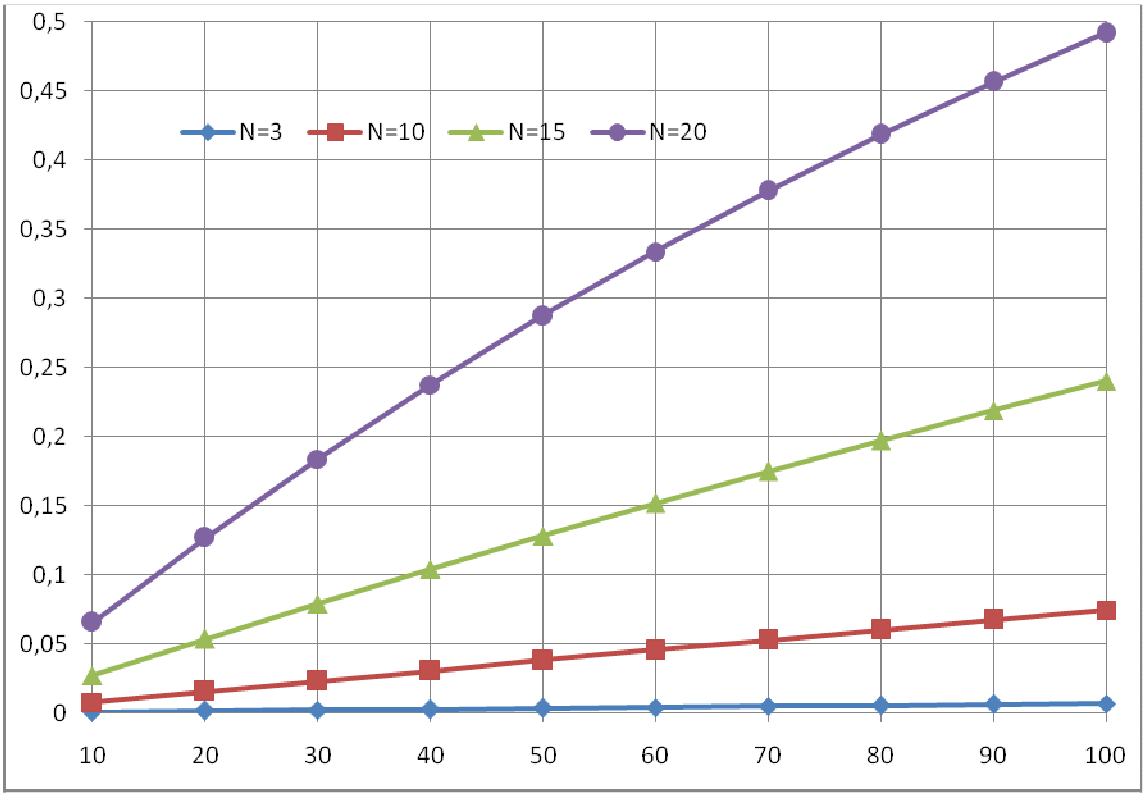


*Рис. 4.* Зависимость вероятности поступления требований на чтение из необновленных реплик в зависимости от интенсивности запросов на чтение λ при различных N.

Системы NoSQL, например, RIAK [9], принимают значение N по умолчанию равным 3. В этом случае даже при больших λ вероятность не превышает 0.007 (надежность согласованности составляет почти две девятки: 0,993).

Предположим, что реплики обновляются параллельно и все реплики одинаково удалены от координатора. Серверы, где хранятся реплики, имеют одинаковые характеристики. В этом случае суммарное время обновления всех реплик можно оценить через время обновления первой реплики (N произвольно).

На рис. 5 показана зависимость вероятности поступления хотя бы одного требования на чтение из необновленных реплик от интенсивности входящих запросов при различной удаленности реплик от координатора. Число физических узлов – 20, число виртуальных узлов – 20. Размер ключа K составляет 20 байтов; размер значения V – 1024 байта.

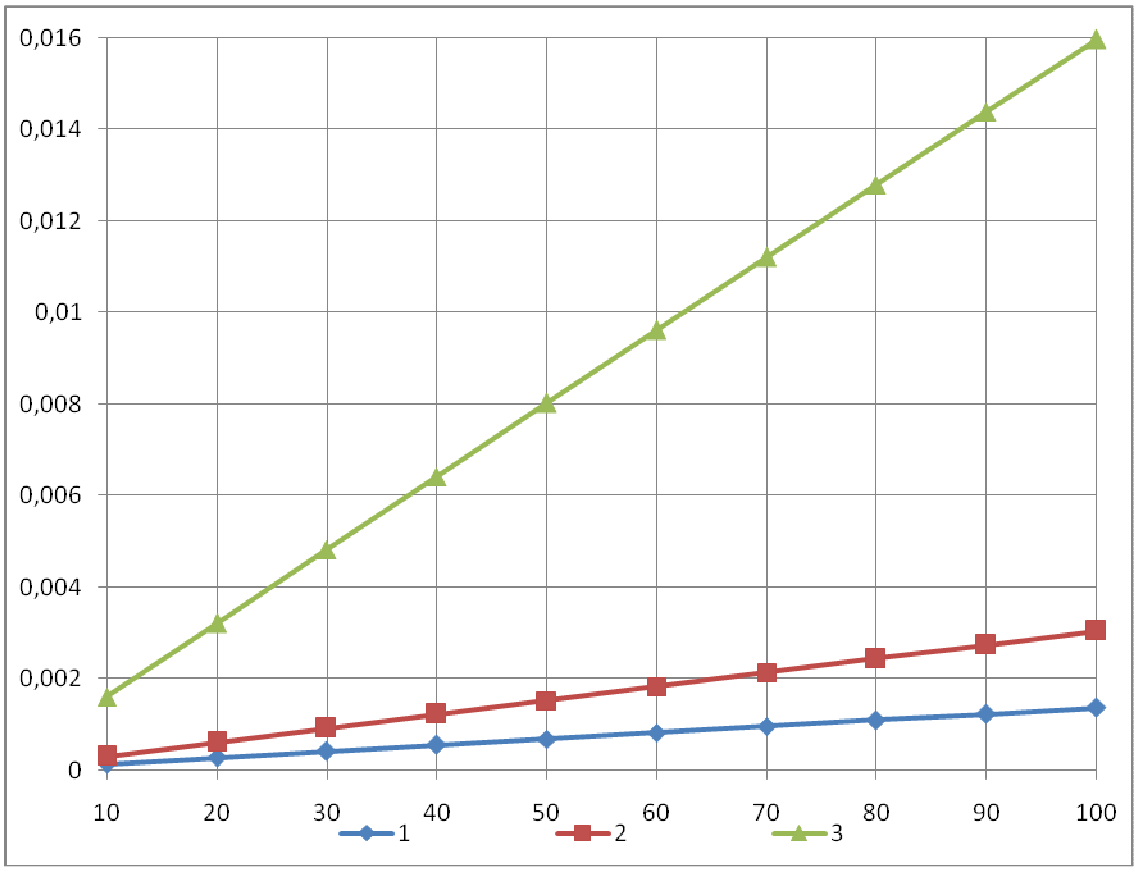


*Рис. 5.* Зависимость вероятности поступления требований на чтение в течение параллельного обновления всех реплик в зависимости от интенсивности запросов на чтение λ при различной удаленности реплик от координатора.

Графики на рис. 5 показывают зависимости: график 1 (легенда 1) – при нахождении всех реплик на координаторе; график 2 – все реплики находятся в том же сегменте сети, что и координатор; график 3 – все реплики находятся в подсети, отличной от подсети координатора. Видно, что наличие реплик в сегменте сети координатора обеспечивает надежность согласованности по чтению записей из БД почти на уровне трех девяток (1 – 0,003 = 0,997) даже при больших значениях λ.

## Заключение

Выведена оценка вероятности, что за время обновления N реплик придет хотя бы одно требование на чтение записей из необновленных реплик.

Получено преобразование Лапласа-Стилтьеса функции распределения вероятностей времени обновления реплики.

Выполнен анализ полученных результатов, из которого следует, что при малых N (N = 3) система NoSQL позволяет обеспечить свойство согласованности по чтению на уровне 0,99 (двух девяток). А наличие реплик в сегменте сети координатора повышает эту надежность почти до 0,999 (т.е. до трех девяток).

ЛИТЕРАТУРА


1. NoSQL. [Электронный ресурс] [http://ru.wikipedia.org/wiki/NoSQL] Проверено 22.03.2014.
2. *Редмон Э., Уилсон Д.Р.* Семь баз данных за семь недель. Введение в современные базы данных и идеологию NoSQL. – М.: ДМК Пресс, 2013.
3. Cap Theorem. [Электронный ресурс] [http://en.wikipedia.org/wiki/CAP_theorem] Проверено 22.03.2014.
4. *Bermbach D., Tai S.* Eventual Consistency: How soon is eventual?, 2011. [Электронный ресурс] [http://dl.acm.org/citation.cfm?id=2093186] Проверено 22.03.2014.
5. *Bailis P., Venkataraman Sh., Franklin M.J., Hellerstein J.M., Stoica I.* Probabilistically Bounded Staleness for Practical Partial Quorums, 2012. [Электронный ресурс] http://www.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-4.pdf] Проверено 22.03.2014.
6. *Ивченко Г.И., Каштанов В.А., Коваленок И.Н.* Теория массового обслуживания. – М.: Высшая школа, 1982.
7. Cyclic redundancy check. [Электронный ресурс] [http://en.wikipedia.org/wiki/Cyclic_redundancy_check] Проверено 22.03.2014.
8. AIDA64 Extreme Edition. [Электронный ресурс] [http://www.aida64.com/product/aida64-extreme-edition/overview] Проверено 22.03.2014.
9. Riak documentation. [Электронный ресурс] [http://docs.basho.com/index.html] Проверено 22.03.2014.



*E-mail:*
*Григорьев Юрий Александрович – grigorev@bmstu.ru;*
*Цвященко Евгений Васильевич – eugene.tsviashchenko@gmail.com.*